



Polytechnic University
of Madrid

**Higher Technical School
of Computer Engineers**



Masters in Software and Systems

Master's Dissertation

**Battery self-management with Artificial
Intelligence in an IoT Architecture**

Author: Andrei García Cuadra
Tutor: Jose Crespo Del Arco

Madrid, 07/2024

This Master's Thesis has been submitted to the School of Computer Science (ETSI) at the Polytechnic University of Madrid for its defense.

Master's Dissertation

Master's Degree in Software and Systems

Title: Battery self-management with Artificial Intelligence in an IoT Architecture

07/2024

Author: Andrei García Cuadra

Tutor: Jose Crespo Del Arco

Departamento de Lenguajes, Sistemas Informáticos e Ingeniería de Software

Grupo de Informática Biomédica

ETSI Informáticos

Universidad Politécnica de Madrid

Resumen

En la era de las tecnologías avanzadas del Internet de las Cosas (IoT), la gestión eficiente de la energía en aplicaciones en tiempo real se ha vuelto fundamental. Esta tesis presenta un enfoque novedoso para la autogestión de baterías utilizando Inteligencia Artificial (IA) dentro de una arquitectura IoT en tiempo real. Centrándose en baterías de Fosfato de Hierro y Litio (LiFePO₄), el estudio aprovecha el poder de la IA y las tecnologías avanzadas de procesamiento de datos, incluidos los protocolos Apache Kafka y WebSocket, para optimizar el uso de energía y promover modelos de energía circular. El sistema propuesto tiene como objetivo mejorar la longevidad y la eficiencia de los sistemas de baterías mediante la monitorización inteligente y el mantenimiento predictivo.

Los componentes clave de la arquitectura incluyen un backend robusto implementado con Kafka para manejar flujos de datos de alto volumen y actualizaciones en tiempo real, y un frontend construido con Angular para una interacción de usuario fluida. La integración de principios de programación reactiva, módulos federados y micro frontends asegura la escalabilidad y mantenibilidad del sistema. Las estrategias de prueba que emplean el marco Jest validan el rendimiento y la fiabilidad de los componentes del sistema.

Esta investigación contribuye al campo de la energía sostenible demostrando cómo la autogestión de baterías impulsada por IA puede conducir a mejoras significativas en la optimización energética y la sostenibilidad en aplicaciones IoT.

Palabras clave: IA, LiFePO₄, IoT, Arquitectura en tiempo real, Kafka, Energía circular, Optimización energética

Abstract

In the era of advancing Internet of Things (IoT) technologies, efficient energy management in real-time applications has become paramount. This thesis presents a novel approach to battery self-management using Artificial Intelligence (AI) within a real-time IoT architecture. Focusing on Lithium Iron Phosphate (LiFePO₄) batteries, the study leverages the power of AI and advanced data processing technologies, including Apache Kafka and WebSocket protocols, to optimize energy usage and promote circular energy models. The proposed system aims to enhance the longevity and efficiency of battery systems through intelligent monitoring and predictive maintenance.

Key components of the architecture include a robust backend implemented with Kafka for handling high-volume data streams and real-time updates, and a frontend built using Angular for seamless user interaction. The integration of reactive programming principles, federated modules, and micro frontends ensures scalability and maintainability of the system. Testing strategies employing Jest framework validate the performance and reliability of the system components.

This research contributes to the field of sustainable energy by demonstrating how AI-driven self-management of batteries can lead to significant improvements in energy optimization and sustainability in IoT applications.

Keywords: IA, LifePO₄, IoT, Realtime architecture, Kafka, Circular energy, energy optimization

Contents

1	Introduction	1
1.1	Motivation	3
1.2	Objectives	4
1.3	Document structure	5
2	Theoretical framework & State of the art	7
2.1	Energy process	9
2.1.1	Energy Production (Fossile)	10
2.1.2	Energy Consumption	11
2.1.3	Energy Generation (Renewables)	12
2.2	Battery chemicals overview	13
2.2.1	Lead-Acid Batteries	14
2.2.2	Nickel-Cadmium (NiCd) Batteries	15
2.2.3	Nickel-Metal Hydride (NiMH) Batteries	16
2.2.4	Lithium-Ion Batteries	17
2.2.5	LiFePO ₄ Batteries: The chosen chemical	18
2.3	Raspberry Pi	22
2.4	Shelly IoT devices	26
2.5	Theoretical photovoltaic electrical system	27
2.5.1	PV installation types	27
2.5.1.1	Grid-Tied PV Systems	28
2.5.1.2	Off-Grid PV Systems	29
2.5.1.3	Hybrid PV Systems	30
2.5.2	Composition	31
2.5.2.1	Solar Panels	31
2.5.2.2	DC to AC Inverter: Victron Multiplus II	31
2.5.2.3	Battery Storage System: LiFePO ₄ and JK BMS	32
2.5.2.4	Charge Controller: Victron SmartSolar MPPT	32
2.5.2.5	Interconnect software devices: CerboGX	34
2.5.2.6	Interconnect electrical systems: Lynx Distributor (DC)	34
2.6	Current researchs	34
3	Project management	35
3.1	System Requirements (User Histories)	37
3.2	Versioning	44
3.3	Gantt Chart	45
3.4	Next upgrades	46
3.5	System design	47

3.6	Programming Principles	49
3.6.1	SOLID Principles	49
3.6.2	Architectural Patterns	49
3.6.3	Importance of Architecture in Battery Management	50
4	Cloud implementation	51
4.1	DevOps	52
4.1.1	Google Cloud	53
4.1.2	Kubernetes and Docker	54
4.1.3	Grafana	55
4.1.4	Jenkins	56
4.1.5	SonarQube	57
4.1.6	GitLab	58
4.1.7	Keycloak	59
4.1.8	OpenApi	60
4.1.9	Redis	61
4.1.10	Kafka	62
4.2	Backend implementation	64
4.2.1	Spring boot with Java implementation	64
4.2.2	Database and storage	65
4.2.2.1	PostgreSQL	65
4.2.2.2	MongoDB	68
4.2.2.3	Nasuni	68
4.2.3	Transactional layer	70
4.2.4	Testing	73
4.3	Frontend implementation	74
4.3.1	Transactional layer	76
4.3.2	Native application	77
4.3.3	Testing	78
5	On-premise implementation	79
5.1	BMS and batteries	79
5.1.1	BMS Configuration	82
5.2	B-Box - Data extraction	84
5.2.1	Enhanced Passive Balancing Using Machine Learning	87
5.2.2	Active Balancing with artificial intelligence	89
5.2.3	Battery degradation prediction with artificial intelligence	91
5.2.4	Battery lifespan prediction with artificial intelligence	93
5.2.5	Battery predictive maintenance with artificial intelligence	95
5.2.6	Full state of charge prediction with artificial intelligence	97
5.2.7	Next days charge prediction with artificial intelligence	99
5.2.8	Expected consumptions with artificial intelligence	101
5.3	IoT	103
5.3.1	MQTT	105
6	Conclusions	107
6.1	Impact analysis	109
A	Appendix	115
A.1	Channels definition	115

A.1.1	AlertChannel	115
A.1.2	BatteryChannel	116
A.1.3	BatteryMetricChannel	116
A.1.4	ConsumptionMetricChannel	117
A.1.5	DcmfChannel	117
A.1.6	ElectricalSystemChannel	117
A.1.7	GenerationMetricChannel	118
A.1.8	SolarChargerChannel	118
A.1.9	UserChannel	118
A.1.10	UserChannel	118
A.2	BMS Screenshoots	119
A.3	DTO implementation	126
A.3.1	AlertChannelTransactionDto	126
A.3.2	BatteryChannelTransactionDto	127
A.3.3	BatteryCellChannelTransactionDto	128
A.3.4	BatteryMetricChannelTransactionDto	128
A.3.5	ConsumptionMetricChannelTransactionDto	128
A.3.6	DcmfChannelTransactionDto	129
A.3.7	ElectricalSystemChannelTransactionDto	129
A.3.8	GenerationMetricChannelTransactionDto	129
A.3.9	SolarChargerChannelTransactionDto	130
A.3.10	UserChannelTransactionDto	130
A.4	Shelly devices overview	131
A.5	Shelly Configuration	140
A.6	Electrical equipment specifications	142
A.7	App Screenshoots	143
A.8	Raspberry PI GPIO Pins	147
A.9	STOMP with spring boot benefits	148
A.10	Shelly IoT Devices overview	149
A.11	Victron communication: RJ-45 Ethernet	149
A.11.1	VE.Direct Protocol	149
A.11.2	VE.Can Protocol	149
A.11.3	VE.Bus Protocol	150
A.12	Junit testing code	151
A.12.1	Failing test (RED)	151
A.12.2	Passing test (GREEN)	151
A.12.3	Refactor the code	151
A.13	MapStruct testing	152
A.14	Reactive programming in Angular	153

List of Figures

2.1	<i>Fossil energy generation by [Science(2024)]</i>	10
2.2	<i>Lead-acid gel chemical schema by [saVRee(2024)]</i>	14
2.3	Nickel-Cadmium chemical schema [ScienceDirect(2024)].	15
2.4	Nickel-metal chemical schema [SemanticScholar(2024b)].	16
2.5	Li-ION chemical schema by [SemanticScholar(2024a)].	17
2.6	Prismatic Lifepo4 chemical schema by [ResearchGate(2024)].	18
2.7	Lifepo4 charge curve by [Florio and Ross(2020)].	19
2.8	Lifepo4 charging stages by [Solar(2024)].	20
2.9	Programmable FPGA image example by [Wikipedia(2024)].	22
2.10	Grid tied system diagram for a PV system by [Parhamfar and Adeli(2022a)].	28
2.11	Off grid system diagram for a PV system by [Parhamfar and Adeli(2022b)].	29
2.12	MPPT diagram by [Mohammed Elgendy (ResearchGate)(2024)]	33
3.1	Sprint lifecycle visual representation for each sprint for a standard way of SCRUM implementation by [Amoniac(2024)].	35
3.2	System operational showcase diagram. Image property of Andrei García Cuadra for this dissertation.	47
4.1	Environment staging. Image property of Andrei García Cuadra	52
4.2	Ideal google cloud platform diagram to handle data. Derived from official documentation of [Google(2024a)].	53
4.3	<i>K8S Pod handling example schema</i> by [Google(2024c)].	54
4.4	Keycloak flow diagram of auth handling by [KeyCloack(2024)].	59
4.5	Architecture diagram for communications. Image property of Andrei García Cuadra.	63
4.6	Class diagram of current project. Image property of Andrei García Cuadra.	64
4.7	Entity relation diagram part 1. Image property of Andrei García Cuadra.	65
4.8	Entity relation diagram part 2. Image property of Andrei García Cuadra.	66
4.9	Current project migrations. Image property of Andrei García Cuadra. . .	67
4.10	Nasuni distributed database example of [Nasuni(2024)]	69
4.11	Data encapsulation strategy with business logic as a reference to the endpoints by [Buddy(2024)].	71
4.12	Reference architecture of <i>Spring Boot MVC</i> by [Medium(2024)].	71
4.13	Data encapsulation strategy with mapper as a reference to the endpoints by [Buddy(2024)].	71
4.14	Federated modules schema, extracted from [smashingtips(2024)].	75
4.15	Electron recompilation for native angular apps by [DevCommunity(2022)].	77
5.1	JK BMS with unwired socket connections by [Nkon(2022)].	79

5.2	Engineering schematic diagram for BMS and LFP by [Yagci et al.(2022)].	81
5.3	Cells 3 and 4 are not properly connected in the given JK Bms configuration. Image taken by Andrei García Cuadra on JK Bms App.	82
5.4	System diagram for deploying IA along with IoT data to Google Cloud. Image property of [Google(2024b)].	103
5.5	MQTT messages (only few of) that are being transacted on the system every second.	105
A.1	Battery bank list (partial covered). Image property of Andrei García Cuadra taken in JK Bms.	119
A.2	Battery current status general overview. Image property of Andrei García Cuadra taken in JK Bms.	120
A.3	Battery settings accesed by tapping tab "settings". This must be unlocked by using the button "Verify PWD" in order to make changes into the configuration. Image property of Andrei García Cuadra taken in JK Bms.	121
A.4	Battery settings when scrolling down, as a continuation of figure A.3. Image property of Andrei García Cuadra taken in JK Bms.	122
A.5	Battery settings when scrolling down, as a continuation of figure A.4. Image property of Andrei García Cuadra taken in JK Bms.	123
A.6	Battery settings when scrolling down, as a continuation of figure A.5. Image property of Andrei García Cuadra taken in JK Bms.	124
A.7	Battery management controls by JK Bms. Image property of Andrei García Cuadra taken in JK Bms.	125
A.8	WIFI Configuration of Shelly <i>DCMF</i> . Image property of Andrei García Cuadra.	140
A.9	Bluetooth Configuration of Shelly <i>DCMF</i> . Image property of Andrei García Cuadra.	140
A.10	MQTT Configuration of Shelly <i>DCMF</i> . Image property of Andrei García Cuadra.	141
A.11	Version Configuration of Shelly <i>DCMF</i> . Image property of Andrei García Cuadra.	141
A.12	Authentication Configuration of Shelly <i>DCMF</i> . Image property of Andrei García Cuadra.	141
A.13	Application main page where data can be analyzed with a first look and easy to read graphics. Image property of Andrei García Cuadra.	143
A.14	General status of all the batteries in the bank. Image property of Andrei García Cuadra.	144
A.15	Battery cells overview status. Image property of Andrei García Cuadra. .	144
A.16	Administration page of PECD. Image property of Andrei García Cuadra.	145
A.17	Battery administration page. Image property of Andrei García Cuadra. .	145
A.18	Administration page of MPPT (solar charge controllers). Image property of Andrei García Cuadra.	145
A.19	Status page of PECD in user mode (non expert). Image property of Andrei García Cuadra.	146

List of Tables

2.1	Comparative Overview of Battery Chemicals Using Likert Scale. Table and scale property of Andrei García Cuadra.	13
2.2	Voltage Chart for LiFePO ₄ Battery (1S and 18S Configurations). Table property of Andrei García Cuadra for this dissertation.	21
2.3	Grid PV installation types summary. Table property of Andrei García Cuadra for this dissertation.	27
3.1	Versioning with specific user histories for each. Property of Andrei García Cuadra for this dissertation.	44
5.1	JK BMS configuration explanation. Property of the table of Andrei García Cuadra for this dissertation.	83
5.2	JK BMS serial data headers. Table property of Andrei García Cuadra . .	86
A.11	JASolar 405W Solar Panel Specifications. Extracted from the side of the bought solar panels.	142
A.12	Raspberry Pi GPIO Pin Configuration. Table property of Raspberry PI corp [Pi(2024)].	147
A.13	Real-Time selection criteria: STOMP with Spring Boot. Extracted from [Boot(2024)] and paraphrased for current dissertation.	148
A.14	Shelly Device Features Comparison, property of [Shelly(2024b)]	149

Chapter 1

Introduction

The integration of Artificial Intelligence (AI) into Internet of Things (IoT) architectures is paving the way for innovative approaches to energy management, particularly in the domain of battery self-management. This convergence aims to optimize energy utilization, improve battery life, and ensure seamless operation of IoT devices, which are increasingly prevalent in our daily lives. The project (encompassing previous work from the final degree project [Cuadra(2024)] and future work for the doctoral thesis) explores the pivotal role of AI in revolutionizing battery self-management within IoT frameworks, focusing on its ability to learn and adapt to various energy consumption patterns and environmental conditions along with predicting battery degradation and ensuring proper functionality during its life cycle.

Taking advantage of AI in battery management systems offers a multitude of functionalities, including predictive maintenance, real-time monitoring, and adaptive energy distribution, which collectively contribute to the sustainability and efficiency of ecosystems IoT. Using AI algorithms and data analytics, systems enabled by AI can predict battery degradation, optimize charging cycles, and prevent potential failures before they occur, thus extending the operational life of batteries, and even more.

In addition, integration of AI into IoT architectures facilitates the development of smart grids and renewable energy systems, where efficient battery management is crucial. Through the analysis of vast amounts of data collected from sensors and devices, AI algorithms can make informed decisions about energy storage and distribution, improving the reliability and performance of IoT networks.

This dissertation delves into the data methodologies and technologies that underlie battery self-management driven by AI, the importance of data quality and quantity, and the challenges associated with integrating AI into IoT architectures. The document also highlights the implementations and current project applications that illustrate the benefits and potential of this cutting-edge approach to battery management.

The fusion of AI with IoT architectures represents a significant leap forward in the field of battery self-management. making it a cornerstone for the future of IoT and energy systems. This is roughly empowered by cloud systems such as Google Cloud that grants the capability of having high availability across with auto-scaling over Dockerization of environments, which causes all the developers and servers to have the same version of the hardware and software making the application easy to scale and maintain.

Dealing with software, managing batteries is really complex in terms of having (near) real-time data over serial protocol which can be corrupted or absent in several ways. This is why using an FGPA such as Raspberry Pi will allow the project to translate complex I/O signals into real-time publisher/subscriber pattern to match the data across all the application. Mentioning cloud environment is near out of the scope of the on-premise running batteries, it is near to distribute the information and compute artificial intelligence for the system.

To face the challenge of making this huge project implementation, the first step is to understand the limits of it and to create a roadmap for all tasks needed to guarantee success in development by making progress in time, calculating deviations, and ensuring proper quality of each module. *Immediately after knowing how to achieve the results, the next step is to set up the cloud environment and develop back-end and front-end applications to ensure that we can handle the data that will be produced by devices on premises (mostly BMS and IoT devices) so that it can be threatened to achieve our objectives as faced in 1.1.*

1.1 Motivation

As we enter the era of the Internet of Things (IoT), the proliferation of connected devices in various sectors has led to a surge in the demand for efficient, reliable, and safe energy solutions. In this context, lithium iron phosphate (LiFePO₄) batteries emerge as a cornerstone for sustainable energy storage, due to their robustness, longevity, and eco-friendliness. However, managing these batteries in complex IoT environments poses significant challenges, which require advanced solutions for monitoring, analysis, and optimization.

This dissertation focuses on the integration of IoT architectures with cloud computing technologies and Raspberry Pi platforms to create a sophisticated system to manage LiFePO₄ batteries. The core of this initiative is to harness real-time data and leverage cloud-based analytics to ensure optimal performance and extend the life cycle of battery systems and reduce consumption.

- **IoT Integration:** By embedding IoT sensors and devices within battery management systems, we can collect detailed operational data in real time. These data are crucial to monitor battery health, predict maintenance needs, and identify potential failures before they occur, among others.
- **Cloud Analytics:** Utilizing cloud platforms enables the aggregation and analysis of a large amount of data collected from IoT devices. Advanced machine learning algorithms and data analytics tools deployed in the cloud provide deep insight into battery usage patterns and efficiency trends, facilitating proactive management strategies.
- **Raspberry Pi Utilization:** The Raspberry Pi serves as an accessible and powerful tool for local data processing and acts as a gateway between IoT devices and the cloud. It enables the execution of edge computing tasks, reducing latency, and ensuring that critical decisions are made swiftly and locally, enhancing system responsiveness. It is essential for development fast-prototyping while saving budget as Raspberry stands for the cheapest out of its competitors.
- **LiFePO₄ Battery Focus:** Special attention is paid to LiFePO₄ batteries due to their unique characteristics. Unlike other lithium-ion batteries, LiFePO₄ offers greater safety, longer service life, and greater thermal stability if properly managed, which are vital for IoT applications that require robust and reliable energy solutions such as this project.

The convergence of these technologies promises to revolutionize battery management systems of energy storage solutions in IoT environments. This project aims to develop a model that not only addresses technical aspects, but also considers environmental impacts, paving the way for a future of sustainable energy.

Regarding [Cuadra(2024)] previous work that is the basis for this dissertation, *key differences are based on the focus of the work.* While the project is just a set of iterations that improve the functionality, the main focus of this project is to manage batteries efficiency and guarantee uptime from time, while previous work was to setup an initial infrastructure (and architecture) which was based on a synchronous architecture and right now it is asynchronous for having IoT and data collection from the system and not only batteries so consumption can be managed and controlled.

1.2 Objectives

The main objective of integrating Battery Self-Management with Artificial Intelligence (AI) in a real-time Internet of Things (IoT) environment is to revolutionize the way battery systems are monitored and optimized. This involves several key goals:

1. **Enhance Battery Efficiency:** *Develop AI algorithms that optimize the charging and discharging processes to extend battery life and improve overall energy efficiency.*
2. **Predictive Maintenance:** *Employ predictive analytics to anticipate and mitigate potential battery issues before they lead to system downtime, thus reducing maintenance costs.*
3. **Real-Time Monitoring and Control:** *Implement systems for the continuous monitoring of battery health and performance metrics, allowing immediate adjustments based on real-time data.*
4. **Energy Optimization:** *Manage and allocate battery resources, enhancing energy use and storage efficiency in response to demand fluctuations and renewable energy availability.*
5. **System Reliability and Safety:** *Strengthen the reliability and safety of battery powered systems by reducing the risk of failure and ensuring consistent performance (especially in variable operational conditions).*
6. **Sustainability and Environmental Impact:** *Promote sustainability by maximizing battery life and reducing the frequency of replacements, which in turn minimizes waste and the environmental footprint of battery systems.*

Achieving these objectives will significantly advance the field of battery management, offering improved performance, reliability, and sustainability for a wide range of applications (from consumer electronics to large-scale renewable energy storage systems). By using software we achieve the state of having the observability and monitoring of the system will ensure it works the way it is expected to, and prevent disasters before it occur. Also, granting full control of the system will allow the electronics to enhance the behaviour by sharing information between consumer electronics (such as BMS, Inverter, IoT devices...) so that the state of the desired device is not only controlled but enhanced using AI algorithms and distributed databases and / or systems. So software will add an external layer for the systems that will communicate them in an efficient way while we achieve less consumptions and more efficiency overall.

1.3 Document structure

The dissertation is organized as follows:

1. Chapter 1 introduces the project, outlining the research background and the importance of integrating IoT with battery management.
2. Chapter 2 reviews the existing literature, highlighting recent advances in IoT, cloud computing, and battery management technologies.
3. Chapter 3 details the methodology used for system design and data analysis.
4. Chapter 4 presents the details of the implementation, including the system setup, configuration, and initial results.
5. Chapter 5 discusses the results, offering insight into battery performance and system efficiency.
6. Chapter 6 concludes the dissertation with a summary of the findings and recommendations for future work.

Chapter 2

Theoretical framework & State of the art

The incorporation of **Artificial Intelligence (AI) into Internet of Things (IoT) frameworks** heralds a novel approach in energy management, specifically in the domain of battery self-management systems. This approach exploits fine-tune energy use, improve battery durability, and ensure smooth *IoT* function. At the core of this is the function in real-time surveillance and flexible energy distribution, coupled with its ability to scrutinize extensive datasets to forecast battery wear, refine charging cycles, and proactively modify operational parameters, thus prolonging battery life and improving device dependability.

Right as [Bairwa et al.(2022)] states:

"Dependency on battery is ever increasing in all sectors. [...] However, the best battery option and the development of electronic circuits and algorithms for more effective battery utilization is still being pursued and is open to researchers around the world. [...] For uninterrupted functioning of the battery bank, all required parameters need to be monitored. [...] The system monitors and stores parameters that provide an indication of battery voltage, current, and temperature in a real-time scenario. The wireless monitoring system is used as the backbone network."

Moreover, **battery management systems** are being amalgamated into larger networks such as smart grids and interconnected vehicle fleets. In addition to energy management, the deployment of AI strengthens device safety by pinpointing and neutralizing safety risks in real time, while this constant gathering and examination of data creates crucial data privacy and security issues as stated by [Mazhar et al.(2023)], which require rigorous safeguards to protect confidential data and maintain user confidence.

This fusion is revolutionizing the field of battery management, with **AI-powered systems capable of forecasting battery wear, refining charging cycles based on usage trends and environmental factors, and proactively modifying operational parameters to prolong battery life and improve device dependability** helping in the creation of smart grids and renewable energy systems taking into account that it is related not only to the engineering field, but also to the social aspects, as stated by

[Kojonsaari and Palm(2023)]. Scrutinize sensor and device data to make intelligent decisions about energy storage and distribution despite offering numerous advantages battery management powered by AI also suffers from crucial data privacy and security issues, requiring robust safeguards to protect confidential data and maintain user confidence that, as is known by [Walczak et al.(2023)], will not only affect the active population, but also a variety of it, with a 55% population living in cities, this is going to be projected to 68% worldwide, making IoT systems necessary to ensure sufficient uptime in smart city systems.

Having known that there is some research that is working proactively in this area, merging artificial intelligence with IoT is an effective solution that is currently not so developed, while the main focus is on one side or the other instead of taunting both at the same time. That is the case of [Sebasthirani et al.(2023a)] which focuses on granting a second life to battery storage systems from several sources, such as cars, to improve sustainability and lower prices for battery banks, or [Fadelallah et al.(2023)] which actively focuses on communication using the protocol ESP32 to manipulate LifePO4 batteries remotely.

2.1 Energy process

Energy processes encompass the complete cycle of energy production, consumption, and generation. They play a central role in the fueling of economies, societies, and daily routines around the world. These processes are vital to understanding how energy is captured, converted and utilized in various sectors such as industry and residential areas. Fundamentally, energy production begins with the extraction or capture of raw energy sources. This energy is then transformed into usable forms, such as electricity or heat, and transported to the point of consumption. The consumption phase refers to the utilization of energy by end users, industries, or for service provision. Lastly, energy generation, often confused because production focuses on fossil energy production (it depends on primary sources), while energy generation focuses on renewable sources (does not depend on primary sources but on environmental statuses), focuses on the methods and technologies employed to generate electricity or other energy forms from primary sources. *As global demand for energy continues to increase, a profound understanding of these processes is essential to devise sustainable, efficient and innovative solutions to meet future energy needs, while fighting climate variability could lead systems such as air purifiers, seawater purifiers and cooling systems to roughly consume 80% of the total energy demand in 2050, as noted by [Tahir and Al-Ghamdi(2023)].*

Taking into account hybrid renewable energy systems is not going to be covered since it involves chemical research, especially when using *waste-to-energy* systems as covered in [Lee et al.(2023)], which are currently in an experimental process to guarantee that it can produce long-term results, while this project needs stability on chemicals to guarantee risk-free results in a short term since it is aimed to be chemical agnostic in the future, thus it needs the first versions to be stable. Also, hydrogen is another actor across the energy process, but as stated in the [Hassan et al.(2023)] evaluations, a long-term effort will be made to implement robust systems that use hydrogen, as stated in his words:

"Consequently, several nations concentrate their research efforts on implementing hydrogen as the catalyst of a clean energy system, issuing various plans and road maps with the cooperation of manufacturers and hydrogen organizations to reduce carbon emissions in the stationary economy [...] divides hydrogen utilization into four stationary uses based on current scientific literature: massive power production; manufacturing sites; residential buildings; and other infrastructures."

Making hydrogen not available as an energy backup, as this dissertation is being written in 2024, but granting a good resource path to be taken for a sustainable future.

2.1.1 Energy Production (Fossile)

Energy production is the process of generating energy from primary sources (without circularity of the energy¹) as seen in figure 2.1. **Fossil energy production remains a cornerstone of global energy supply**, as shown by [Pan and Dong(2023)] (using China as an extrapolation example), based mainly on coal, oil and natural gas. These resources, formed from the remains of ancient plants and animals subjected to heat and pressure over millions of years, are extracted from the Earth and burned to generate electricity and power transport. The combustion process releases significant amounts of carbon dioxide, a greenhouse gas that contributes to climate change, along with other pollutants that can harm air quality and public health. Despite these environmental impacts, fossil fuels are favored for their energy density, reliability, and established infrastructure keeping them a persistent choice in the global energy mix. However, the shift towards renewable energy sources is accelerating as the world seeks to reduce emissions and combat the adverse effects of climate change. As authorized by [Nadeem(2023)] in the specific case of Pakistan:

"The sustainable planning of a country requires extensive investigation in the power and transportation sector. Reliable and sustainable resources for electrification are essential for better living standards."

This will bring about the need for fossil energy for a while, while we implement renewable energy sources through electricity streams across a hypothetical worldwide smart grid that may not be feasible due to international conflicts.

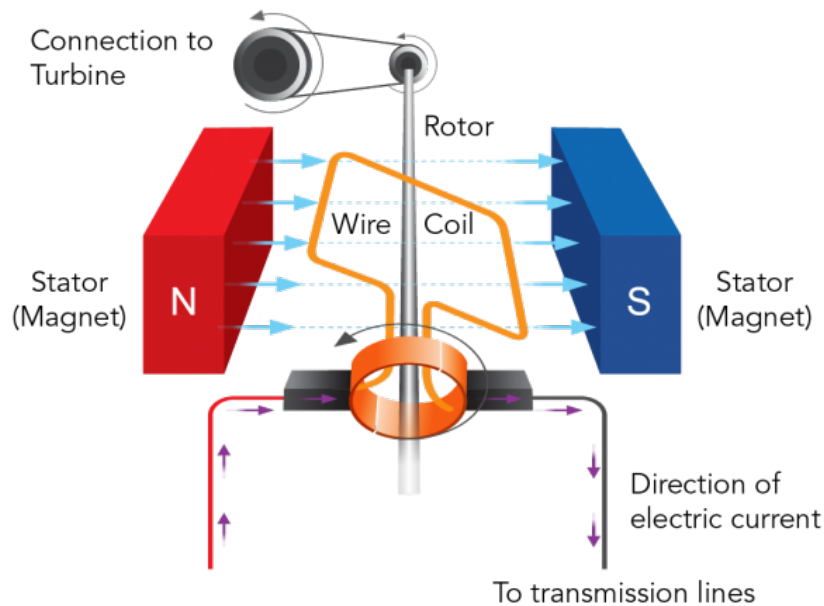


Figure 2.1: Fossil energy generation by [Science(2024)]

¹Circularity aims to optimize energy use, reduce waste, and minimize environmental impact by creating systems where energy efficiency and the reuse of waste energy are prioritized.

2.1.2 Energy Consumption

Energy consumption refers to the use of energy by various sectors:

- **Residential:** Heating, cooling, lighting, and appliances.
- **Commercial:** Building operations, computers and office equipment.
- **Industrial:** Manufacturing processes, machinery and production lines.
- **Transportation:** Fuel for vehicles, airplanes, and ships.

Although not going that deep into the formulas that would ensure our hypothesis, using the study of [González et al.(2024)] focusing on Spain as one of the main gas-dependent countries along with Germany, the determinant factors are stated by [González et al.(2024)]:

As for the determinant factors taken into account in the decomposition, the following driving forces will be considered:

- (a) Population effect, that is, the impact of population growth.*
- (b) Effect of activity, including the impact of economic growth. Assuming a constant (average) coefficient between GDP and CO2 emissions, this effect can be regarded as the theoretical CO2 emissions that come from economic activities (Sun, 1998).*
- (c) Intensity effect, that is, the impact on the emission of energy requirements per unit of value added. It involves energy consumption related to some variables such as energy prices, energy conservation and energy-saving investments, the structure and efficiency of energy systems, technological choices, and socioeconomic behavior.*
- (d) Effect of energy carbon intensity, which is defined as the impact on the mass of gas emitted from each unity of fuel consumed. It is also called the carbonization effect.*

From that, we can infer that consumption will grow as we move to electrical solutions while gas is still one of the main resources needed for our energy consumptions; even knowing that Spain is one of the countries of Europe with more sun hours (if not the one with most hours) we will struggle with the storage [Gómez-Calvet et al.(2023)] of all the production we can make, making the consumption out of sync with it.

2.1.3 Energy Generation (Renewables)

Energy generation focuses on converting primary energy sources into electricity through various renewable sources.

1. **Thermal generation:** Using heat to produce steam that drives turbines. *We can infer that this is such in an experimental state, while being researched by many authors ([Elwardany et al.(2024)], [Olabi et al.(2023)]) it is still not a viable solution for the expected incoming energy consumptions and also not available in all countries as this energy generation is strictly linked to volcanoes, granting a huge advantage to states like Norway or Finland.*
2. **Hydroelectric power:** Using water flow to turn turbines. *This was a common practice in areas with drought and was also a perfect backup for solar energy generation as stated by [Barbón et al.(2023)].*
3. **Solar Panels and Wind Turbines:** *This is a seriously limited way to obtain energy because we directly depend on environmental statuses which may vary really fast, leaving us with no generation. This is why [Lee(2023)] researched the correlation between solar, wind, and hydro power generation in Spain to ensure we never run out of energy (not counting batteries or energy storage, only real-time production).*
4. **Nuclear power:** Utilizing nuclear reactions to generate heat and, subsequently, electricity (using fusion). *This is considered green energy, but social opinion may differ from expectations of energy production and that is why it may not be the most popular even when its efficiency is high and the interdependence with renewable sources is a strong backup option as stated by [Wang et al.(2023)].*

The choice of generation method affects the sustainability, cost, and availability of energy. Focusing on specific cases such as the prediction of the Saudi Arabian input demand by [Samargandi et al.(2024)], minerals will be (if not already) critical resources for all renewable energies along with energy storage, as stated by the previously mentioned author [Gómez-Calvet et al.(2023)].

New renewable sources implies new solutions which consider using another environmental conditions such as compressed air by [Karaca et al.(2023)] although this could be considered a battery by the definition of which a battery "stores energy" and that is what essentially this project aims to achieve, but the prolonged release of energy as it is needed is a perfect backup for renewable energies such as solar panels or wind turbines that directly depend on environmental state and this merges with the next steps needings, which consists directly on finding new energy storage or production solutions that cover that traditional energy generation renewables such as solar or wind such as stated by [Ba-Alawi et al.(2023)] using osmosis.

2.2 Battery chemicals overview

This section provides an *overview* of various battery technologies, focusing on their **chemical compositions**, **applications**, **advantages**, and **limitations**. From traditional *lead-acid batteries* to advanced *lithium ion variants* (including **LiFePO₄**) this overview aims to highlight how the choice of chemical components influences a battery's *performance*, *safety*, and *suitability for specific applications*.

Batteries are essential for storing electricity in a *compact and portable form*. The chemical composition of a battery determines its *capacity*, *voltage*, *reusability*, *cost*, and *environmental impact*, so exploring the most common types of batteries and their chemical properties is **essential** to make this project *efficient* and *feasible* to offer a compelling balance of *safety*, *longevity*, and *environmental friendliness*.

The main focus will be on the sub-section 2.2.5 **LifePO4 batteries**, since this is the chemical chosen for the project.

Below comparative table aims to give an overview before going in-depth:

Chemical	Voltage	Dangerous	Energy	Lifespan	Env. Concerns
Lead-Acid	Moderate	High	Low	Low	Very High
NiCd	Moderate	High	Moderate	Moderate	High
NiMH	Moderate	Moderate	High	Moderate	Low
Li-Ion	High	High	Very High	High	Moderate
LiFePO ₄	High	Low	High	Very High	Low

Table 2.1: Comparative Overview of Battery Chemicals Using Likert Scale. Table and scale property of Andrei García Cuadra.

Likert Scale:

- 1 - Very Low (Good)
- 2 - Low
- 3 - Moderate
- 4 - High
- 5 - Very High (Bad)

2.2.1 Lead-Acid Batteries

Lead-acid batteries are a type of **rechargeable battery** commonly used in various applications due to their **relatively low cost** and **high power density**. They consist of *lead dioxide (positive electrode)*, *sponge lead (negative electrode)*, and *sulfuric acid electrolyte*, as shown in Figure 2.2.1. During discharge, sulfuric acid reacts with lead dioxide and sponge lead, producing *lead sulfate* and **releasing electrons**. The knowledge about Lead-Acid batteries was inferred from [Bode(1977)].

This process **generates electrical energy**. During charging, an external electrical source reverses this reaction, converting lead sulfate back into lead dioxide and sponge lead. Despite widespread use, lead-acid batteries have *limitations*, including **low energy density**, **short lifespan**, and **environmental concerns** due to lead and sulfuric acid, prompting research into alternative battery technologies.

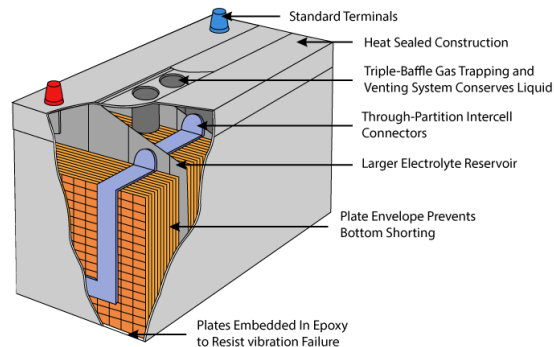


Figure 2.2: Lead-acid gel chemical schema by [saVRee(2024)].

Chemical Composition: Lead acid batteries use *lead (Pb)* and *lead dioxide (PbO₂)* as electrodes and *sulfuric acid (H₂SO₄)* as the electrolyte.

Applications: Commonly used in *automotive starters*, *emergency lighting*, and *backup power systems*.

Advantages and Limitations: *Advantages* include *low cost* and *high power*. *Limitations* include *weight*, *size*, and *environmental concerns*.

2.2.2 Nickel-Cadmium (NiCd) Batteries

Nickel-Cadmium (NiCd) batteries are a type of **rechargeable battery** known for their **high reliability** and **durability**, often used in applications ranging from *portable electronics* to *power tools* as all the knowledge from NiCd chemistry was known thanks to [Jeyaseelan et al.(2020)]. They consist of a **positive nickel oxide hydroxide electrode**, a **negative cadmium electrode**, and an *alkaline electrolyte solution* typically composed of potassium hydroxide as shown in figure 2.3. During discharge, cadmium undergoes an *oxidation reaction* at the negative electrode, releasing electrons and forming *cadmium hydroxide*, while nickel oxide undergoes a *reduction reaction* at the positive electrode, generating electrons and forming *nickel hydroxide*. This flow of electrons produces **electrical energy**. During charging, the reactions are reversed as an external electrical current drives the cadmium and nickel hydroxide back to cadmium and nickel oxide, respectively. NiCd batteries offer advantages such as **high discharge current**, **long cycle life**, and **tolerance to overcharging**, but they are also subject to *memory effect*, have *lower energy density* compared to newer battery technologies like *lithium-ion*, and cadmium raises **environmental concerns**, leading to regulations limiting their use and encouraging the development of more environmentally friendly alternatives.

Chemical Composition: NiCd batteries consist of *nickel* (NiOOH) as the positive electrode and *cadmium* (Cd) as the negative electrode.

Applications: Used in *portable power tools*, *aviation*, and *backup power* because of their robustness.

Advantages and Limitations: They offer *high discharge rates* and *durability*, but they suffer from *memory effects* and *environmental issues*.

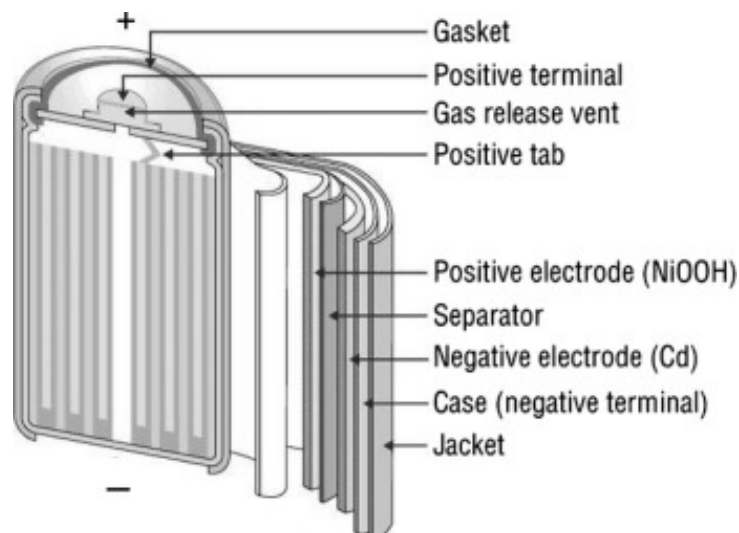


Figure 2.3: Nickel-Cadmium chemical schema [ScienceDirect(2024)].

2.2.3 Nickel-Metal Hydride (NiMH) Batteries

Nickel-Metal Hydride (NiMH) batteries are **rechargeable batteries** widely utilized in *portable electronic devices* and *hybrid vehicles* as demonstrated by their compactness in figure 2.4 as all the information of NiMH batteries was inferred from [Fetcenko et al.(2007)]. They are favored due to their **high energy density** and **relatively low environmental impact** compared to alternatives like *nickel-cadmium*. They consist of a *positive electrode* composed of nickel oxyhydroxide, a *negative electrode* containing a metal hydride alloy, and an *alkaline electrolyte* typically composed of potassium hydroxide. During discharge, hydrogen ions migrate from the negative electrode to the positive electrode through the electrolyte, while electrons flow through the external circuit, producing **electrical energy**. This process causes the metal hydride alloy to release hydrogen ions and electrons, forming metal ions. Simultaneously, nickel oxyhydroxide accepts electrons, forming nickel ions. During charging, the reactions are reversed as an external electrical current drives the nickel ions back to nickel oxyhydroxide and the metal ions back to the metal hydride alloy, effectively storing energy. NiMH batteries offer advantages such as *higher energy density* and *lower toxicity* compared to NiCd batteries, although they are still subject to *self-discharge* and the *memory effect* to a lesser extent. Despite these limitations, NiMH batteries remain a popular choice for various applications, especially where a balance between *performance*, *cost*, and *environmental impact* is desired.

Chemical Composition: NiMH batteries use a chemistry similar to that of NiCd, with a *hydrogen-absorbing alloy* replacing cadmium.

Applications: Widely used in *consumer electronics*, *hybrid vehicles*, and *medical devices*.

Advantages and Limitations: They provide a *higher capacity* than NiCd batteries, are *more environmentally friendly* but have a *shorter lifespan*, and are *sensitive to high temperatures*.

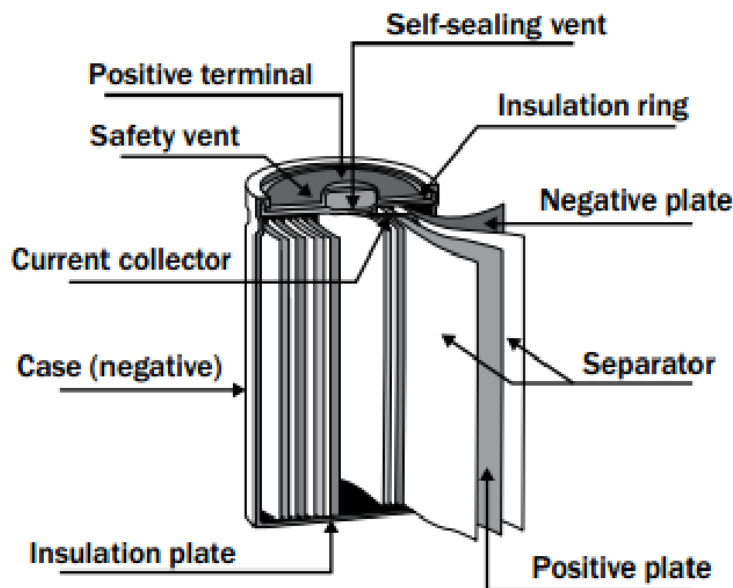


Figure 2.4: Nickel-metal chemical schema [SemanticScholar(2024b)].

2.2.4 Lithium-Ion Batteries

Lithium-ion batteries (**Li-ion**) are **rechargeable energy storage devices** widely used in *portable electronics*, *electric vehicles*, and *renewable energy systems* due to their **high energy density** as shown in figure 2.5 and provide **long cycle life** from design as all the information of Li-ion batteries was acknowledged from [Li et al.(2009)]. They consist of a **positive electrode (cathode)** typically made of lithium cobalt oxide, lithium manganese oxide, or lithium iron phosphate, a **negative electrode (anode)** made of graphite or other lithium-containing compounds, and a *lithium salt electrolyte* dissolved in an organic solvent. During discharge, lithium ions move from the anode to the cathode through the electrolyte, whereas electrons flow through the external circuit, generating **electrical energy**. This process causes the anode material to release lithium ions and electrons, which are then accepted by the cathode material. During charging, an external electrical current drives lithium ions back from the cathode to the anode, effectively storing energy. Li-ion batteries offer advantages such as **high energy density**, **low self-discharge rate**, and *minimal memory effect*, but they also pose *safety concerns* due to the potential for *thermal runaway* and *fire risk*, especially if improperly handled or manufactured. Continuous research and development efforts focus on improving *performance*, *safety*, and *cost-effectiveness* to further expand the applications of lithium ion technology.

Chemical Composition: Lithium ion batteries encompass various chemistries, including LiCoO_2 , LiMn_2O_4 , LiNiMnCoO_2 (NMC), and LiFePO_4 .

Applications: They are prevalent in *portable electronics*, *electric vehicles*, and *renewable energy systems*.

Advantages and Limitations: Advantages include *high energy density*, *low weight*, and *no memory effect*. Limitations involve *safety risks* and *higher costs*.

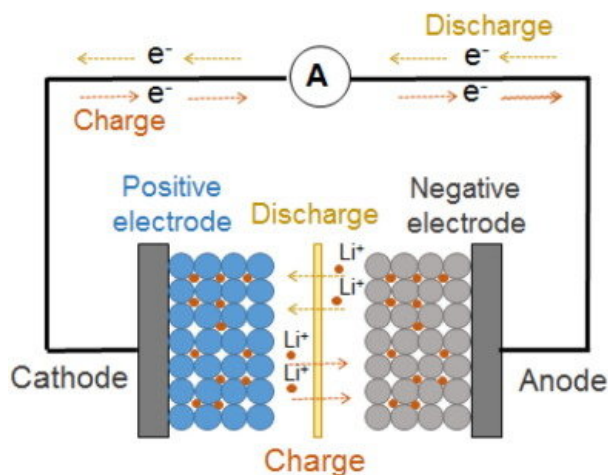


Figure 2.5: Li-ION chemical schema by [SemanticScholar(2024a)].

2.2.5 LiFePO₄ Batteries: The chosen chemical

Lithium Iron Phosphate (LiFePO₄) batteries, often referred to as LFP batteries, are a type of rechargeable lithium-ion battery known for their high energy density, long cycle life, and enhanced safety characteristics as all the information of this section is extracted from [Zhang(2011)]. They feature a cathode made of lithium iron phosphate, an anode typically composed of graphite, and a lithium salt electrolyte. During discharge, lithium ions migrate from the anode to the cathode through the electrolyte, while electrons flow through the external circuit, generating electrical energy.

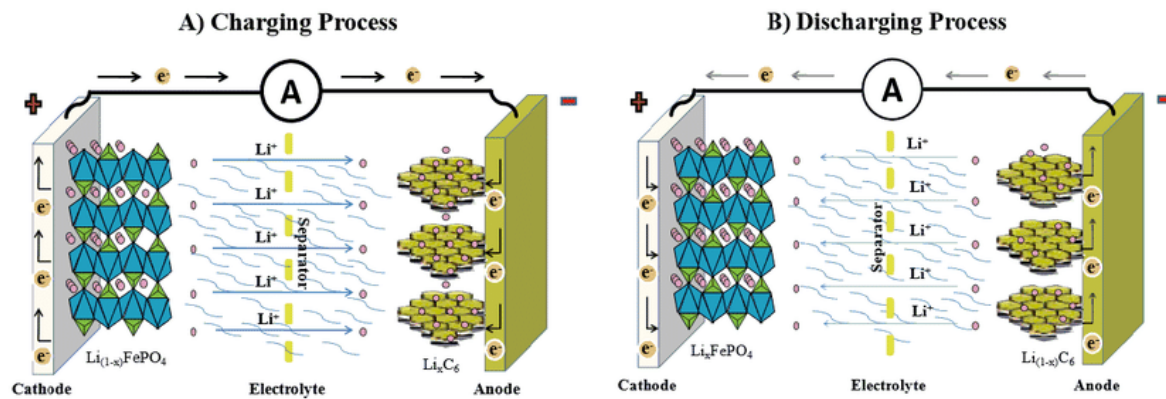


Figure 2.6: Prismatic Lifepo4 chemical schema by [ResearchGate(2024)].

This process results in the reduction of iron phosphate at the cathode and the oxidation of graphite at the anode. LiFePO₄ batteries are distinguished by their stable chemical structure, which significantly reduces the risk of thermal leakage, making them less prone to overheating and fire compared to other lithium-ion chemistries. They also exhibit excellent performance in terms of high charge and discharge rates, making them suitable for applications ranging from electric vehicles to stationary energy storage systems with their prismatic form factor and simplicity of the chemical reaction, as shown in figure 2.6. Despite their slightly lower energy density compared to other lithium ion variants, LiFePO₄ batteries are favored for their superior safety, long lifespan, and environmental friendliness, as they contain no heavy metals such as cobalt.

Chemical Composition: LiFePO₄ batteries use lithium iron phosphate as the cathode material.

Applications: Ideal for electric vehicles, portable power packs, and solar energy storage due to their safety and long life.

Advantages and Limitations: They offer excellent thermal stability, safety, and long cycle life but have a lower energy density compared to other lithium-ion chemistries.

Although LiFePO₄ batteries offer significant advantages, it is essential to compare them with other technologies quimically close to understand their relative benefits and limitations. Compared to lithium cobalt oxide batteries (LiCoO₂) and lithium manganese oxide batteries (LiMn₂O₄), LiFePO₄ batteries have a lower energy density, but offer higher safety and longer service life by having a long cycle life, safety and relatively low cost.

Theoretical framework & State of the art

LiFePO₄ batteries offer several unique chemical and physical properties:

- High thermal stability and safety compared to other lithium-ion chemistries as stated by [Takahashi et al.(2005)].
- Long cycle life, often exceeding 2000 cycles with minimal degradation, as many studies endorse this, such as [Botejara-Antúnez et al.(2024)].
- A stable voltage plateau at 3.2-3.3 volts, which is lower than lithium cobalt oxide (LiCoO₂) batteries, as shown in [Jung and Jung(2016)].

The charge curve of the LiFePO₄ battery is characterized by its flat voltage profile during the bulk of the charging process, which is a result of its stable chemistry, while optimal charging involves adhering to the manufacturer's recommended current and voltage limits, as well as standard battery charge status (SoC) and charging current shown in figure 2.7 to ensure battery health and maximize its useful life. Manufacturers may use different lithium purities and densities, but the standard is 3.2V nominal voltage per prismatic cell. Additionally, temperature management during charging is crucial, as extreme temperatures can affect the battery's performance and safety.

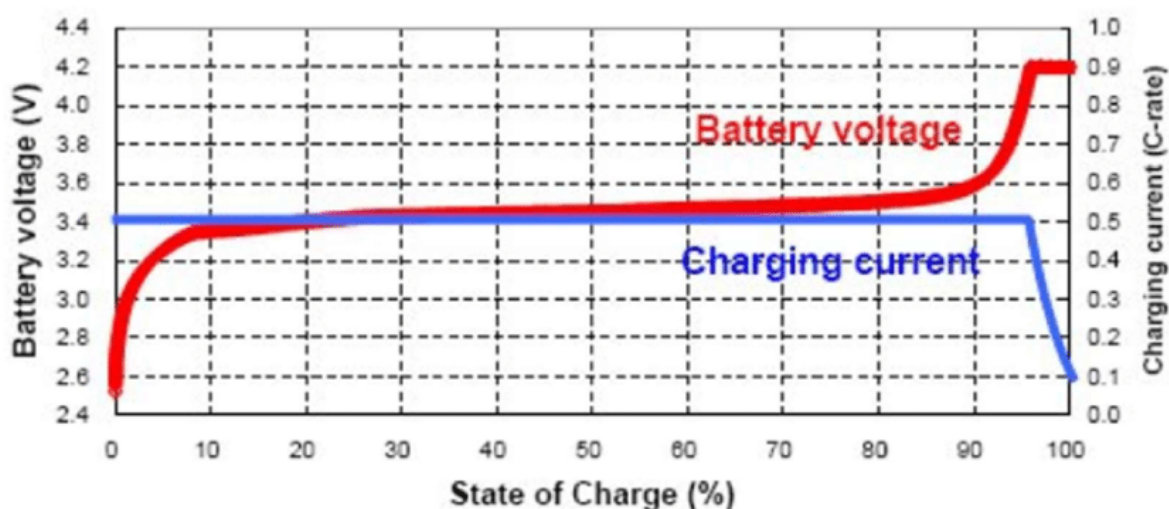


Figure 2.7: Lifepo4 charge curve by [Florio and Ross(2020)].

The charging process of LiFePO₄ batteries involves multiple stages as referred in figure 2.8. It is crucial to follow this stages:

Constant Current (CC) Phase (Bulk). This initial phase involves charging the battery at a constant current until it reaches a set voltage level. The CC phase allows for rapid charging of the battery up to about 70-80% of its capacity.

Constant Voltage (CV) Phase (Absorption). Once the battery reaches the target voltage during the CC phase, the charging process shifts to the CV phase. Here, the voltage is kept constant while the current gradually decreases as the battery approaches full capacity. This phase completes the charging process and ensures the battery is fully charged without overcharging.

Float Charging (Optional) (Float) After the CV phase, some high quality chargers (per example, Victron MPPT was the one used for this project) apply a float charge at

a lower voltage to keep the battery at 100% state of charge without overcharging it. This phase is more common in applications where the battery remains connected to the charger, and it helps maintain the battery at full charge without damaging it.

Equalization (or preventive balancing): This will top balance cells during last charge moments.

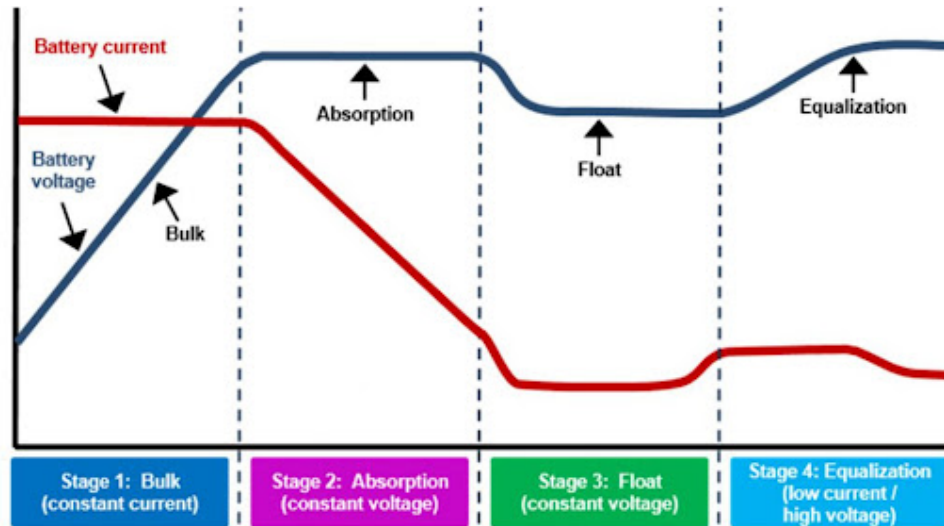


Figure 2.8: Lifepo4 charging stages by [Solar(2024)].

Understanding and implementing the correct charging phases are vital for maximizing the performance and lifespan of LiFePO₄ batteries. It's important to use a charger specifically designed for LiFePO₄ chemistry to ensure that the correct charging algorithm is applied which for this project is Victron SolarCharger MPPT with *BatteryLife* algorithm.

It is also important to know how voltage works and operates in this chemical. At first, there are several voltage nomenclatures that means completely different states of the battery, which was extracted partially from [TutorialsPoint(2024)]:

- **Nominal voltage:** Nominal Voltage is the default, average voltage that a battery is designed to operate at under typical conditions. It is a standardized value used to classify and compare batteries. For LifePO₄ nominal voltage is **3.2V** per cell.
- **Rated Voltage:** Rated Voltage refers to the voltage at which the device is designed to operate optimally. This value is closely related to the nominal voltage but may be specified by the manufacturer to highlight the ideal operating conditions of the device or battery system. For LifePO₄ rated voltage is **3.5V** when idle (if charging or discharging, voltage will differ).
- **Operating Voltage Range:** Operating Voltage Range describes the range of voltages over which the battery can effectively operate. This range includes the maximum voltage when fully charged and the minimum voltage at which the battery can still function before needing recharging or replacement. For LifePO₄ operating voltage is from **2.5V** to **3.65V**.

Since batteries are composed by cells, while this dissertation focuses on 18 string

batteries, below is the table that matches voltages for each cell separately along with 18S battery (nominal voltage of 57,6V when 50% charged and idle). It is important to know that voltage differs when idle, charging or discharging as [Köder et al.(2023)] states. Table 2.2 covers nominal voltage for 1S and 18S battery packs related to SoC:

State of Charge (%)	1S Voltage (V)	18S Voltage (V)
100%	3.65	65.7
90%	3.50	63.0
80%	3.40	61.2
70%	3.30	59.4
60%	3.20	57.6
50%	3.15	56.7
40%	3.10	55.8
30%	3.05	54.9
20%	3.00	54.0
10%	2.95	53.1
0%	2.50	45.0

Table 2.2: Voltage Chart for LiFePO₄ Battery (1S and 18S Configurations). Table property of Andrei García Cuadra for this dissertation.

2.3 Raspberry Pi

A **Field-Programmable Gate Array (FPGA)** is a semiconductor device that is based around a matrix of *configurable logic blocks (CLBs)* connected via *programmable interconnects* as shown physically in figure 2.3 and the knowledge for this section extracted from [Hajjar and Mourad(2019)]. This device can be re-programmed to desired application or functionality requirements after manufacturing, hence the name "field-programmable".

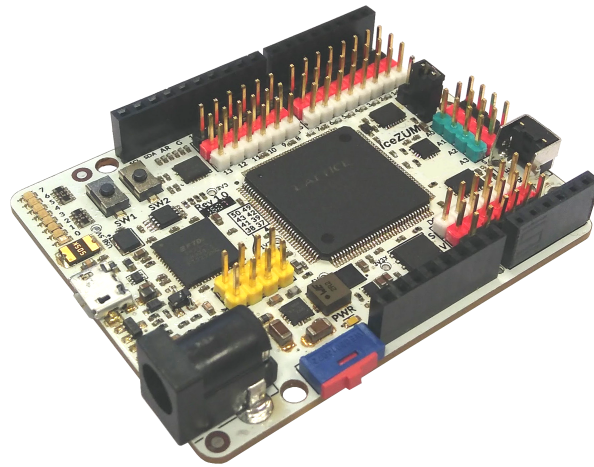


Figure 2.9: Programmable FPGA image example by [Wikipedia(2024)].

FPGAs represent a flexible, powerful technology used across many industries. Their programmability makes them an essential tool for rapid prototyping. The functionality of an FPGA is defined by a *hardware description language (HDL)*, such as VHDL, then the HDL code is synthesized into a configuration file that can be downloaded onto the FPGA, determining how the CLBs, interconnects, and IOBs are configured. A FPGA will be needed to communicate the local system with the cloud as will be stated in chapter 5.2. This section covers the essential knowledge about what an FPGA is.

FPGAs are used in a wide range of applications due to their versatility and reconfigurability such as *Digital Signal Processing (DSP)*, *Telecommunications*, *Aerospace and Defense Systems*, *Automotive Control Systems*, *Medical Imaging*, etc.

While offering several advantages over traditional microprocessors and ASICs such as:

- **Flexibility:** FPGAs can be reprogrammed to desired application or functionality requirements after manufacturing.
- **Speed:** FPGAs can execute complex parallel operations faster than a microprocessor that executes instructions sequentially.
- **Integration:** FPGAs allow the integration of a large amount of logic into a small physical space.

The FPGA architecture consists of a variety of programmable components:

- **Configurable Logic Blocks (CLBs):** These are the fundamental building blocks that provide the logic for the FPGA. Each CLB can perform a certain amount of logic operations, and the complexity of these operations depends on the FPGA's design.

- **Programmable Interconnects:** These allow the CLBs to be interconnected in various ways to perform complex functions. The interconnects can be programmed to create specific pathways between CLBs.
- **Input/Output Blocks (IOBs):** These blocks connect the internal logic of the FPGA to the external pins of the chip, allowing the FPGA to communicate with the outside world.
- **Memory Elements:** FPGAs often include memory elements that can be configured as flip-flops or more complex memory structures like RAM blocks.

For this project the desired selection is the **Raspberry Pi** is a series of small *single-board computers* developed in the United Kingdom by the **Raspberry Pi Foundation** in association with **Broadcom**. The initial intent behind the design of the Raspberry Pi was to promote the teaching of basic computer science in schools and developing countries. Also, according to benefits expressed by [Karthikeyan et al.(2023)]. which convinced me to develop using that platform.

Each Raspberry Pi model provides a small, affordable, yet capable computer that can perform various computing tasks. The standard Raspberry Pi features include a **Broadcom system on a chip (SoC)**, which includes an *ARM compatible central processing unit (CPU)* and *on-chip graphics processing unit (GPU)*, system memory (RAM), and various interfaces and connectors for external devices.

Over the years, several models of the Raspberry Pi have been released, from the original *Raspberry Pi 1 Model B* to the more recent *Raspberry Pi 4 Model B*. Each model varies slightly in terms of processor speed, memory capacity, and connectivity options.

Choosing a Raspberry Pi over other competitors can be advantageous for several reasons. First, its affordability makes it highly accessible for budget-conscious developers and it also boasts a large active community that provides extensive support. Additionally, its low power consumption and small size make it ideal for projects with space constraints or those requiring energy efficiency, while supporting a wide range of operating systems and programming languages such as Python, Java, Perl...

The **GPIO (General Purpose Input/Output) pins** on the Raspberry Pi offer a versatile interface for connecting and controlling a wide range of external devices. Each GPIO pin can function as either an input or output, allowing it to send or receive digital signals. These pins support various forms of communication protocols, including *I2C, SPI, and UART* which can be crucial for more complex tasks and for interfacing with various sensors and actuators when communicating over serial.

In addition to GPIO-based communications, the Raspberry Pi also supports serial communication over USB. This is particularly useful for interacting with external peripherals and devices that require a serial interface. USB serial communication is handled through the USB ports on the Raspberry Pi, which can be configured to act as serial ports. This is achieved by using a USB-to-serial adapter or cable. This setup allows for the transmission and reception of serial data over the USB connection. The complete information on the GPIO pins can be found in the appendix A.8.

All Raspberry Pi models feature an Ethernet port, providing a reliable and fast wired network connection which is essential for applications that require stable and high-speed Internet access as this project requires. Starting with the Raspberry Pi 3,

built-in Wi-Fi support is available, including both 2.4 GHz and 5 GHz bands (5 GHz starting from Raspberry Pi 3 Model B+). This feature enables the Raspberry Pi to connect to the internet and other networked devices wirelessly, which is particularly useful in scenarios where wired connections are impractical. Bluetooth functionality is included in the Raspberry Pi 3 and later models. This allows the Raspberry Pi to connect to various Bluetooth peripherals, which are only needed to set it up this project that uses the 3 channel communications over Bluetooth, Wifi and LAN-Ethernet.

The Raspberry Pi provides a rich programming interface primarily supporting Python, which is widely used due to its simplicity and power. Additionally, the Raspberry Pi can run multiple versions of Linux-based operating systems, such as Raspberry Pi OS, which come with a variety of programming languages such as Java, C++, and Ruby. For direct hardware interaction, the Raspberry Pi offers extensive Python GPIO libraries, enabling users to control physical components such as LEDs, sensors, motors, etc. Moreover, the inclusion of development environments such as Thonny, a Python IDE pre-installed in Raspberry Pi OS, further simplifies code development and testing.

Sensors enable the Raspberry Pi to gather data from its surroundings. These devices can measure various environmental parameters.

Types of Sensors

- **Temperature and Humidity Sensors:** Devices like the DHT11 and DHT22 provide information about environmental conditions, useful in home automation and weather monitoring. This is needed for batteries to be safe and never in contact with the water and/or humidity and also stay at a certain temperature by using air conditioner that is fully controlled by IoT.
- **PIR Motion Sensors:** These are utilized in security systems to detect movements in the environment. It is needed for the project to guarantee access to the battery room to be only done by authorized technicians and to store a record of accesses.
- **Ultrasonic Distance Sensors:** Such as the HC-SR04, these sensors help in distance measurement and are commonly used in robotics. This helps the project by ensuring distance between batteries in order to refrigerate them on the rack.
- **Light Sensors:** Photocells that respond to light intensity are used in projects that require light-based triggers. This is needed to guarantee that the room does not exceed certain lumens so guarantee it is not exposed to heavy environmental conditions.

Plug-in modules enhance the functionality of the Raspberry Pi by providing additional features that are not built into the board.

Plug-in Modules

- **Camera Module:** This module connects to the Raspberry Pi via a dedicated interface, allowing for image capture and video streaming. Useful to record battery room and ensure security on it.
- **HATs (Hardware Attached on Top):** These boards attach directly to the Raspberry Pi's GPIO pins, providing capabilities such as motor control power man-

agement, or specialized functions like weather monitoring. This fits by using shelly sensors to interoperate between GPIO pins and shelly devices.

- **Display Screens:** Various screens can be connected to display information directly from the Raspberry Pi, facilitating the creation of user interfaces or portable devices. It was used in the project, but removed due to performance issues on the device.

Using a Raspberry Pi will allow IoT devices to properly communicate with BMS without the need of proprietary systems from *JK BMS* or other companies that would consume our budget by using devices while limiting the capabilities we can manage over BMS by using a serial protocol. All the knowledge is going to be covered in section 5.2.

2.4 Shelly IoT devices

The **Internet of Things** (IoT) is a revolutionary technology that merges the *physical* and *digital* worlds. IoT devices, known for their internet connectivity and data exchange capabilities, are transforming various industries, including home automation. Among the many IoT options available, **Shelly devices** have become popular due to their flexibility, affordability, and ease of use.

Shelly IoT encompasses a range of smart home gadgets created to give users exceptional command over their living spaces. Crafted by *Allterco Robotics*, Shelly devices are distinguished by their small size, Wi-Fi connectivity, and compatibility with major home automation protocols. These devices empower users to automate daily tasks, track energy consumption, and boost home security via remote management and instant alerts. Shelly IoT gadgets mark a significant progression in smart home technology, providing an easy entryway into home automation. By utilizing these devices, this project enhances convenience and security while effectively managing energy usage. As the IoT field advances, Shelly's inventive solutions are positioned to have a significant impact on the future of interconnected living. A wide range of devices can be found in the comparison table in appendix A.10 and in appendix A.4 for a detailed description of the devices and all the information regarding shelly devices, interactions, panels, and specifications was extracted from its official documentation [Shelly(2024b)] and endpoints and api specification from [Shelly(2024a)].

Shelly devices boast a range of features that cater to the diverse needs:

- **Wi-Fi Connectivity:** Direct connection to home networks without the need for additional hubs.
- **Compatibility:** Supports integration with popular platforms like Google Home, Amazon Alexa, Home Assistant, MQTT and Api REST.
- **Energy Monitoring:** Advanced devices offer real-time energy consumption data, facilitating energy management and savings.
- **Remote Access:** Users can control devices from anywhere, using mobile apps or web interfaces, or limit access through MQTT and API.
- **Customization:** Flexible settings and scripting capabilities allow for personalized automation scenarios.

Shelly IoT devices find applications in various domains of home automation:

1. **Lighting Control:** Automate lighting based on time, occupancy, or other triggers.
2. **Energy Management:** Monitor and control energy consumption of household appliances to reduce bills.
3. **Security Enhancements:** Integrate with sensors and alarms to enhance home security.
4. **Climate Control:** Regulate heating, ventilation, and air conditioning (HVAC) systems for optimal comfort and efficiency.

This IoT devices will provide full control capability over electrical devices that may have high consumption such as thermal water heaters that could drain batteries.

2.5 Theoretical photovoltaic electrical system

A **photovoltaic (PV) electrical system** utilizes sunlight as a *plentiful* and *renewable* energy source to produce electricity in an eco-friendly circular manner. This cutting-edge system includes solar panels constructed from semiconductor materials that transform sunlight directly into **electrical power**. By incorporating elements like inverters, battery storage, and control systems, PV systems can effectively provide electricity for a variety of purposes presenting a *green alternative* to conventional fossil fuel-dependent energy sources. Given technological advancements and the growing emphasis on **sustainable development**, photovoltaic electrical systems are leading the global shift towards green energy while playing a pivotal role in satisfying the escalating energy demands of the world while reducing environmental harm.

2.5.1 PV installation types

This subsection provides an in-depth comparison of the three primary types of electrical photovoltaic (PV) installations: **grid-tied**, **off-grid**, and **hybrid systems**. Each system type is designed to meet specific energy needs and operational requirements, varying significantly in terms of installation complexity, cost, and application scenarios. Choosing the right type of PV installation depends on various factors including location, energy needs, reliability of the local grid, budget, among others.

Grid-tied systems offer simplicity and financial benefits through net metering, **off-grid systems** provide energy independence but at a higher cost, and **hybrid systems** offer a balance between grid reliability and autonomous power provision, making them suitable for a wide range of cases as shown in table 2.3.

Feature	Grid-Tied	Off-Grid	Hybrid
Grid Connection	Yes	No	Yes
Battery Storage	Not required	Required	Required
Power during Grid Outage	No	Yes	Yes
Complexity	Low	High	High
Initial Cost	Lower	Higher	Highest
Operational Cost	Lower	Higher due to battery replacement	Moderate
Best for	Areas with reliable grid	Remote locations without grid access	Areas with intermittent grid outages

Table 2.3: Grid PV installation types summary. Table property of Andrei García Cuadra for this dissertation.

2.5.1.1 Grid-Tied PV Systems

Grid-tied PV systems are connected directly to the public electricity grid and do not require battery storage as shown in figure 2.10. They are characterized by:

- **Operational Efficiency:** High efficiency as they convert solar energy directly into electricity that is either used on-site or fed back into the grid.
- **Net Metering:** Ability to sell excess electricity back to the grid, providing financial benefits and enhancing system ROI.
- **Limitations:** Lack of power during grid outages, as safety protocols disconnect the PV system from the grid.

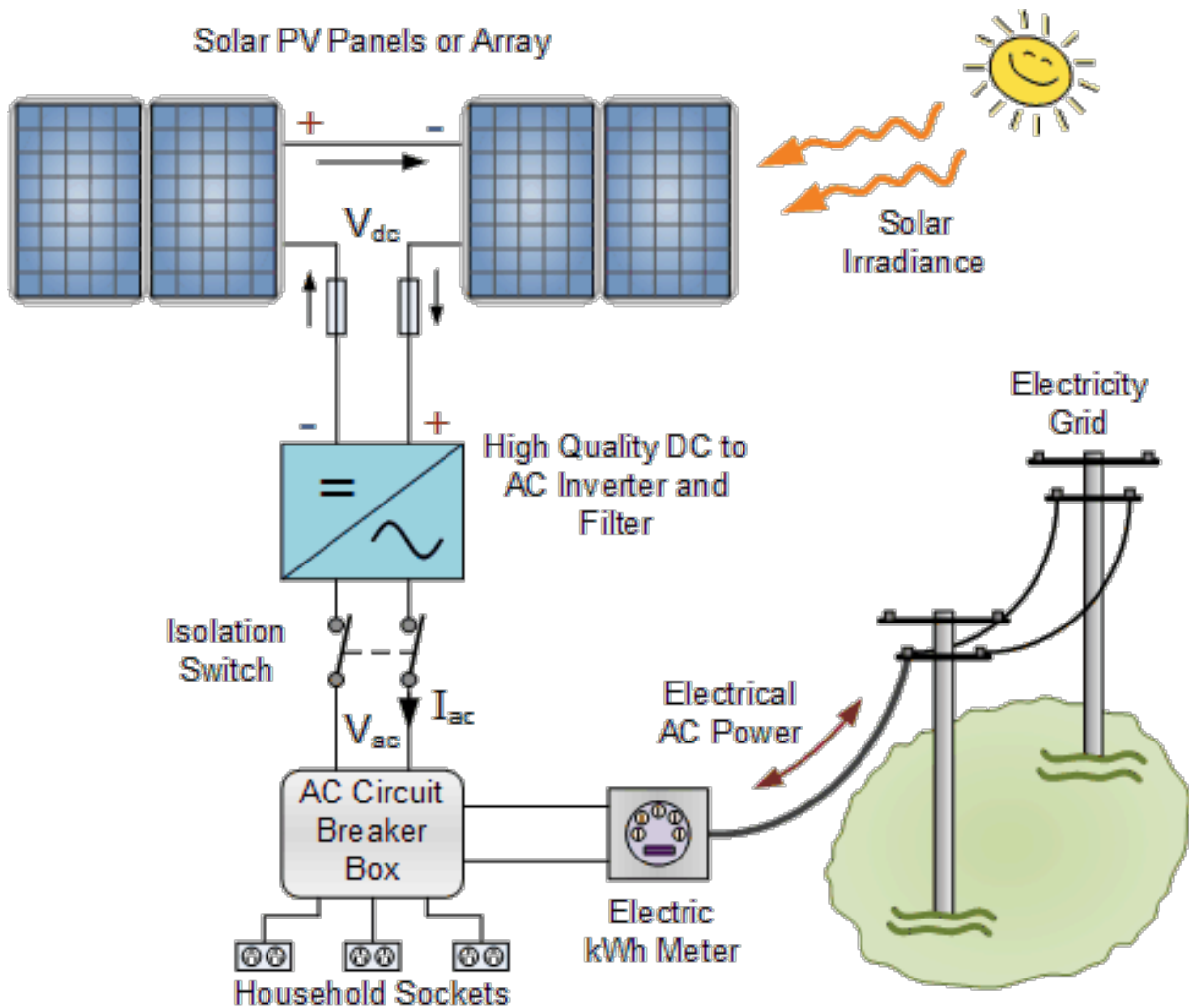


Figure 2.10: Grid tied system diagram for a PV system by [Parhamfar and Adeli(2022a)].

2.5.1.2 Off-Grid PV Systems

Off-grid systems are completely disconnected from the public electricity grid and typically include battery storage as shown in figure 2.11. Their features include:

- **Independence:** Suitable for remote locations where grid connection is impractical or too costly.
- **Storage Requirement:** Necessity for battery storage to ensure power availability during nighttime or cloudy days.
- **System Design:** Typically more complex and costly due to the need for additional components like charge controllers and batteries.

This is the chosen type of installation because it simplifies the final electrical schema while granting full control of the system by not depending on external factors or agents.

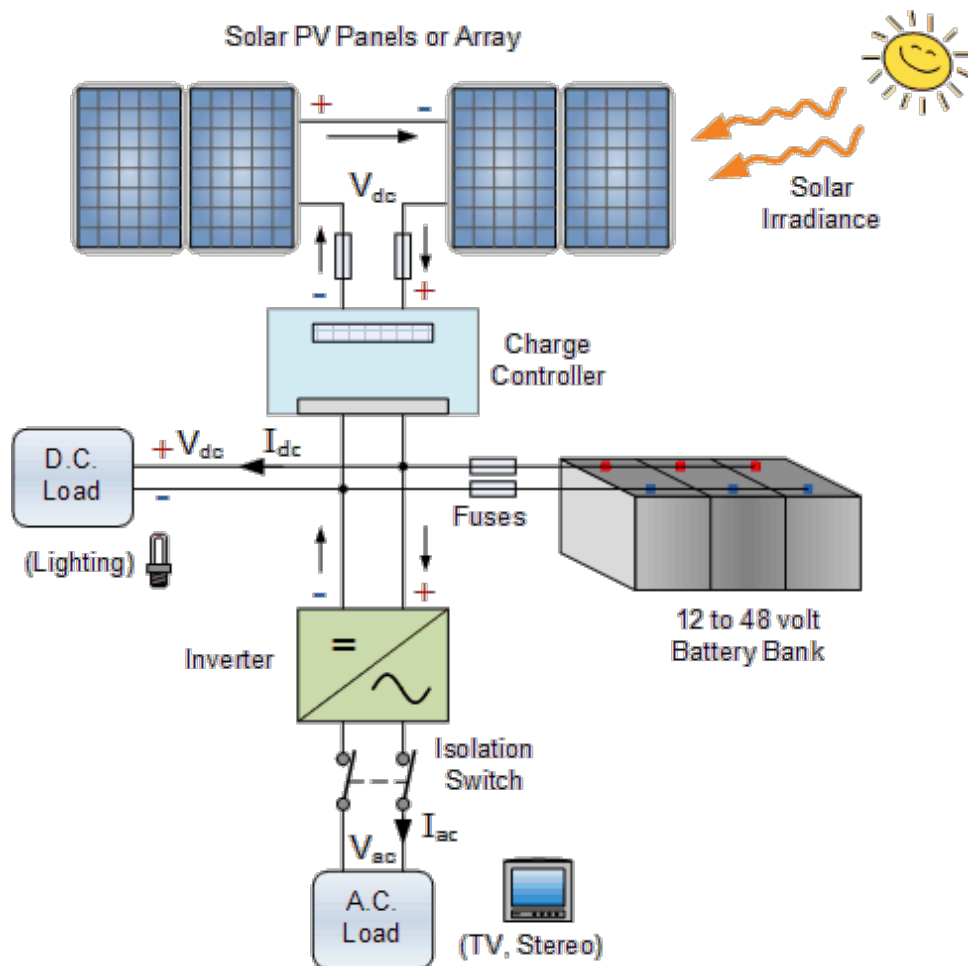


Figure 2.11: Off grid system diagram for a PV system by [Parhamfar and Adeli(2022b)].

2.5.1.3 Hybrid PV Systems

Hybrid systems combine features of both grid-tied and off-grid systems, offering a versatile power solution:

- **Grid Connection with Backup:** Connected to the grid but also include batteries for energy storage to provide power during outages.
- **Energy Management:** Ability to manage energy efficiently by storing excess power and using stored energy during peak demand times or grid outages.
- **Cost and Complexity:** Higher initial investment and complexity due to the integration of grid-tied and off-grid components.

This type of installation is not really covered in this dissertation since it is too complex and would add extra boilerplate to the work, reducing the focus on the other alternatives which are the real focus, just taking note this type of installation exists.

2.5.2 Composition

A *photovoltaic (PV) installation* harnesses sunlight to generate electricity and is a complex integration of various components, each serving a specific function to efficiently convert sunlight into usable electricity. Understanding these components and their functions is essential for designing, installing and maintaining this effective and sustainable solar energy system.

2.5.2.1 Solar Panels

Solar panels, or *PV panels*, are the heart of a PV system. They consist of photovoltaic cells that convert sunlight into direct current (**DC**) electricity. Panels vary in type, size, and efficiency, allowing for customization based on energy needs and available space. Currently, **JASolar 405w/h panels** are being used, with the specifications listed below in the table of appendix A.11 extracted from [JASolar(2024)].

2.5.2.2 DC to AC Inverter: Victron Multiplus II

The inverter is crucial in converting the DC electricity generated by the panels into AC electricity, making it compatible with household appliances and the grid. The **Victron MultiPlus II**, a product of *Victron Energy*, is a highly versatile and sophisticated inverter/charger system, renowned for its ability to provide reliable power solutions in diverse energy systems, including off-grid, backup, and grid-tied configurations. All the information listed below was properly extracted section by section of [Energies(2024c)] and paraphrased below.

This intricate device amalgamates the features of a powerful pure sine wave inverter², a sophisticated battery charger, and a swift AC transfer switch into a singular compact unit. It boasts a unique set of features, including power assist technology to prevent overloading a limited AC source, such as a generator or shore power connection, by supplementing the insufficient generator or shore power with energy from the battery. Furthermore, it can serve as the core of a more comprehensive energy storage system when paired with other Victron components, facilitating both solar and grid energy with seamless energy management. It's designed for scalability, allowing multiple units to be connected in parallel or in three-phase arrangements, thus catering to a wide range of power requirements. Thanks to its superior communication capabilities, it can be seamlessly integrated into existing energy management systems, offering extensive monitoring and control options through Victron's comprehensive array of networking and remote management tools. This establishes as an ideal choice for those seeking a robust, flexible, and future-proof power management system. Both Split phase and Parallel MultiPlus II systems are intricate solutions engineered to improve power management and distribution in various scenarios, such as Split phase providing two Outputs of 120V can be merged to produce 240V, efficiently meeting both low and high voltage needs. This system is perfect for load balancing and power distribution in homes and small businesses. Conversely,

²A pure sine wave inverter is a type of power inverter that converts direct current (DC) into alternating current (AC) and produces a smooth, sinusoidal waveform. This is the same type of waveform typically supplied by electrical utility companies and is ideal for powering sensitive electronic devices that require a high-quality wave, such as medical equipment, audio systems, and certain types of telecommunication devices. Pure sine wave inverters are preferred for their ability to efficiently power devices without causing harmonic distortion, which can lead to noise and heat in some electronics.

2.5. Theoretical photovoltaic electrical system

the Parallel MultiPlus II system involves linking several MultiPlus II inverter/chargers in parallel to boost power capacity and achieve redundancy. This setup enables power solutions to be scaled, with the system capable of expansion based on demand, providing flexibility and dependability for more intricate energy requirements. Each system has its unique advantages, with the split phase being more simple for regular household use, while parallel MultiPlus II setups offer modularity and scalability for larger applications. In this project, six MultiPlus II units are used in parallel to augment power output and redundancy in a power system. The MultiPlus II from Victron Energy is a flexible inverter and charger that supports parallel and three-phase setups. Linking six MultiPlus II units in parallel necessitates meticulous consideration of physical layout and configuration settings since for electrical reasons when wiring DC power cables must fit same length to not have voltage discrepancies. This arrangement boosts power capacity and dependability.

The Multiplus can also charge batteries, but for this project it will be used only as an inverter while the charger will operate only under specific conditions such as batteries being empty. Having an operative inverter that has the capability to communicate with the BMS is crucial to ensure the battery life is maximized.

2.5.2.3 Battery Storage System: LiFePO₄ and JK BMS

Batteries store excess energy produced during the day for use during periods without sunlight (such as at night or on cloudy days) ensuring a continuous power supply. This is covered in Section 2.2.5 with respect to *chemical specifications* and in Chapter 5.1.1 for the BMS and *management system*.

2.5.2.4 Charge Controller: Victron SmartSolar MPPT

The *charge controller* regulates the flow of electricity to and from the battery bank, protecting batteries from overcharging or excessive discharge, thus prolonging their useful life. When comparing charge controller types such as **PWM** (Pulse Width Modulation) and **MPPT** (Maximum Power Point Tracking) charge controllers for solar panels, the main distinction lies in their operational efficiency and technological sophistication as states [Chen et al.(2007)]. PWM controllers are simpler and more cost-effective, modulating the energy flow from solar panels to the battery by rapidly switching the current on and off to maintain the battery voltage, effectively matching the panel output to the battery voltage as stated by [Zhang et al.(2014)]. This results in a straightforward but less efficient system, particularly when the solar panel's voltage is significantly higher than the battery's. In contrast, MPPT controllers are designed to extract the maximum possible power from the solar panels by adjusting their input impedance to find the optimal power point of the panels. This technology allows MPPT controllers to be more efficient, especially in varied environmental conditions and when the panel's voltage is significantly higher than the battery's charging voltage, making them preferable for larger systems where maximizing energy harvest is crucial, albeit at a higher cost.

MPPT controllers are capable of converting excess voltage into amperage, ensuring that the solar panels always operate at their maximum power point, thereby significantly increasing the system's energy efficiency. This technology allows for up to 30% more energy yield compared to PWM (Pulse Width Modulation) charge controllers,

especially during periods of low light or in colder climates where the voltage characteristics of solar panels change. Victron's MPPT range includes models for various system sizes and voltages. In addition, these controllers are equipped with intelligent features such as battery life management and adaptive charging algorithms enhancing battery longevity and performance. With built-in Bluetooth for easy monitoring and configuration via VictronConnect app, and compatibility with a wide selection of battery types (including LiFePO4).

The algorithm used by the charge controller is explained as follows using the sequence diagram in figure 2.12: It is important to note that the algorithm is respon-

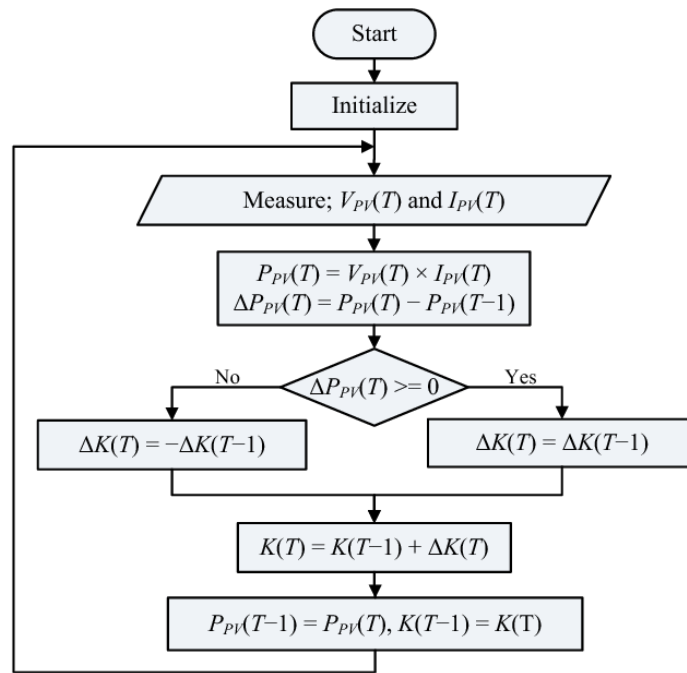


Figure 2.12: MPPT diagram by [Mohammed Elgendy (ResearchGate)(2024)]

sible for granting enough power to the battery bank while also seeking for voltage securities. As [Elgendy et al.(2012)] comments:

«The P&O algorithm continuously perturbs the operating point of the system causing the PV array output voltage to fluctuate around the MPP voltage even if the solar irradiance and the cell temperature are constant. Consequently, similar current and power fluctuations occur. These usually fluctuate between three levels, provided that the perturbation frequency is low enough so that the system can reach a steady state before the next perturbation. The step size must also be high enough so that (a) the control is not affected by noise and (b) the perturbation causes a measurable change in the output power of the array.»

Thus, perturbations could affect the battery bank by degrading it over time, and this is why the MPPT device is a critical selection for the project. Victron MPPT integrates the latest version of the MPPT algorithm as shown in [Elmetennani et al.(2016)].

MPPT is the terminal device that will charge the LifePO4 batteries. Having a proper configuration on it is crucial, especially for the 18s batteries, to charge at 65.7V to guarantee a proper lifespan of the batteries

2.5.2.5 Interconnect software devices: CerboGX

The **Cerbo GX** is an advanced communication center that enhances the monitoring and control capabilities of solar and energy storage systems, designed by *Victron Energy* as all the information is extracted from [Energies(2024a)]. It serves as the core interface between electronic energy components, allowing efficient management and optimization of energy usage. Equipped with a multitude of connectivity options, including **Ethernet, WiFi, and Bluetooth**, it provides real-time insights and detailed system analytics through the *Victron Remote Management (VRM) portal*, which is important to mention that it is not being used in this project for privacy concerns. Its ability to integrate with a wide range of Victron products and third-party devices makes it highly versatile, supporting a broad spectrum of power systems for off-grid and grid-tied configurations. With features such as automatic generator start/stop, load shedding, and system alarms, the Cerbo GX ensures that the energy system performance is maximized.

Contribution to the battery system: The Cerbo GX manages and monitor the health and performance of the battery storage system, ensuring optimal charge cycles and preventing overcharge or deep discharge scenarios, thus prolonging battery life and enhancing overall system efficiency while allowing to monitor the system.

2.5.2.6 Interconnect electrical systems: Lynx Distributor (DC)

The **Lynx Distributor** is an essential component in power systems designed by *Victron Energy*, serving as a central connection point for multiple DC circuits within a power system, and all the information listed in this section is extracted from [Energies(2024b)]. It is engineered to streamline the distribution of power from batteries to various loads and charging sources, accommodating up to four DC devices, such as inverters, chargers, and solar charge controllers. Each connection point is fused individually, enhancing safety by protecting against overcurrent and enabling easy isolation of devices for maintenance or troubleshooting. Constructed with high-quality materials for durability and reliability, the Lynx Distributor is compatible with **12V, 24V, or 48V systems**, while this system uses the base of 48V to maximize it to 65V while charging. Its modular design allows for seamless integration with other Victron components, such as the Lynx Shunt and Lynx Distributor DC.

Contribution to the battery system: The Lynx Distributor facilitates efficient power distribution and management within the battery system, safeguarding against potential electrical hazards, and ensuring stable power delivery to connected devices, which is crucial for maintaining the integrity and longevity of the battery system.

2.6 Current researchs

Current research is mostly based around specific topics such as IoT or AI over the batteries, but since LifePO4 batteries are being industrially used recently, researchs are still in progress as many of the fields have not been covered yet, as they may focus only on specific purposes of the batteries. One example of this is [Sebasthirani et al.(2023b)], who researchs over IoT or [Díaz-Longueira et al.(2023)] who researchs over AI, always without covering purposes of the current work listed in section 1.2.

Chapter 3

Project management

The development of the project will be based on **SCRUM** methodology. All SCRUM knowledge is derived from the original source [Sachdeva(2016)] along with other listed sources. **SCRUM** is a framework for agile project management that is used primarily for software development projects but can also be applied to any complex project or product development. It emphasizes teamwork, accountability, and iterative progress toward a well-defined goal. The **SCRUM** framework is designed to handle projects in a flexible and adaptive way, especially in environments where requirements can change frequently and a fast-paced development cycle is necessary. A cycle is represented in 3.1 and shows the lifecycle of a single sprint.

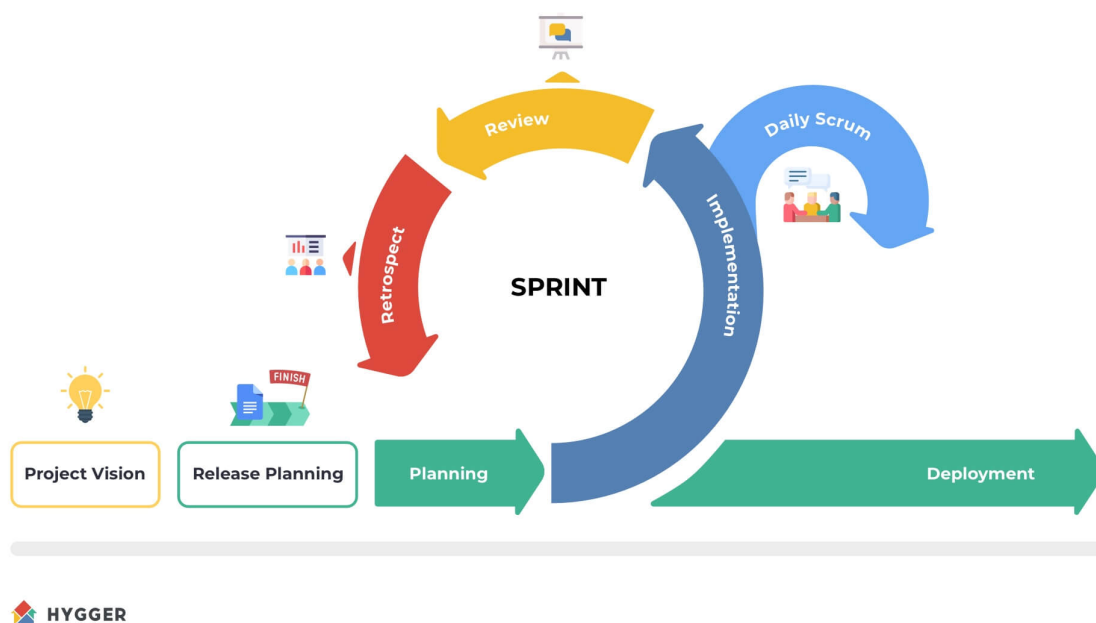


Figure 3.1: Sprint lifecycle visual representation for each sprint for a standard way of SCRUM implementation by [Amoniac(2024)].

Cycles are essential in a scrum process, as stated by [Schwaber(1997)]. **Cycles are composed by:**

- **Sprints:** These are fixed-length iterations of 1 week, during which a specific set of tasks or a functional increment of the product is developed for this project. Each sprint aims at increasing the size of the potentially shippable product. In this project, each iteration will count as one minor version of the development.
- **SCRUM Team:** The team consists of the Product Owner, the Scrum Master, and the Developers. In this case, this will not be assumed since there is only one developer, who is also the owner of the product knowledge.
- **SCRUM Events:** These include Sprint Planning, Daily SCRUM (or Stand-up), Sprint Review, and Sprint Retrospective. In the current project, a sprint review with dissertation director Jose Crespo was only followed to align the knowledge from the time period between syncs.
- **SCRUM Artifacts:** These include Product Backlog, Sprint Backlog, and Scrum Increment. Our backlog is based on user histories, so the increments are going to be versions with specific user histories inside that may not match the same story points or effort but instead constant product incrementations.

3.1 System Requirements (User Histories)

This section will cover the complete implementation of user stories that will be used to compound SCRUM sprints. A **user story** is a brief and simple description of a feature or requirement from the perspective of the end user or the customer as referred to by [Patton and Economy(2014)]. It is a fundamental component within Agile and Scrum methodologies, serving as a tool to capture specific functionalities or needs the user has from the system or product being developed. User stories help ensure that the development team focuses on delivering value to users by understanding their needs and priorities.

User histories will be the strategy to structure the development just as below:

As a < type of user >, I want < some goal > so that < some reason >.

- (HU-1) As a normal user, I want to enable a *PECD* to modify its status.
- (HU-2) As a normal user, I want to disable a *PECD* to prevent its status from being modified.
- (HU-3) As a normal user, I want to view the status of a *PECD* to check its properties.
- (HU-4) As a normal user, I want to view nonsensitive properties of a *PECD* to overview its current characteristics.
- (HU-5) As a normal user, I want to view the activation logs of a *PECD* to visualize the traces of its changes such as activations.
- (HU-6) As a normal user, I want to visualize the production status of the devices.
- (HU-7) As a normal user, I want to see the battery status to ensure the proper functioning of the system.
- (HU-8) As a normal user, I want to receive notifications when the battery is below 20%.
- (HU-9) As a normal user, I want to receive notifications when the energy production of the MPPTs is 0W.
- (HU-10) As a normal user, I want to visualize the total installed capacity to identify errors in production.
- (HU-11) As a normal user, I want to be able to connect to the system to run it.
- (HU-12) As a normal user, I want to turn off the hot water storage heaters but have them restore their state the next day so that they do not consume energy right now.
- (HU-13) As a normal user, I want to turn on the hot water storage heaters but have them restore their state the next day so that they do not consume energy right now.
- (HU-14) As a normal user, I want to turn on the battery room's thermoregulator so that it stays at 25°¹
- (HU-15) As a normal user, I want to visualize the total consumption of the *PECD* to see how much these devices are consuming.

¹25°Celsius is the optimal temperature for stationary lithium ferrophosphate LifePO4 batteries, as described

3.1. System Requirements (User Histories)

- (HU-16) As a normal user, I want to visualize the individual consumption of *PECD* to see how much that device is consuming.
- (HU-17) As a normal user, I want to be able to recover my password to be able to reenter the system if I forget it.
- (HU-18) As a normal user, I want to see the alerts from the *Victron*² / *Cerbo GX*³ systems so I do not have to go to the garage to look at the mini screen of their system.
- (HU-19) As a normal user, I want to visualize the current consumption to see if I can save energy.
- (HU-20) As a normal user, I want to see the remaining battery life if it is being discharged to calculate how many hours I can spend playing on the computer.
- (HU-21) As a normal user, I want to see the remaining time until the battery is fully charged if it is in the *BULK*⁴ to know how many hours of sunlight are left until it is fully charged.
- (HU-22) As a normal user, I want to see the hourly weather forecast for the next 7 days to ensure usability.
- (HU-23) As a normal user, I want to operate with a threshold called twilight ⁵ to activate or deactivate sensors based on solar brightness.
- (HU-24) As a system administrator, I want to disable a *PECD* so that users cannot activate it.
- (HU-25) As a system administrator, I want to enable *PECD* so that users can activate it.
- (HU-26) As a system administrator, I want to delete a *PECD* to remove it from the system.
- (HU-27) As a system administrator, I want to add a new *PECD* to be able to operate it within the system.
- (HU-28) As a system administrator, I want to modify the refresh rate of an existing *PECD* to be able to update its attributes.
- (HU-29) As a system administrator, I want to modify the IPv4/IPv6 address of an existing *PECD* to be able to update its attributes.
- (HU-30) As a system administrator, I want to modify the access password of an existing *PECD* to be able to update its attributes.
- (HU-31) As a system administrator, I want to add a battery to the system so that it can be operated from the system.
- (HU-32) As a system administrator, I want to add an MPPT device to the system to be able to operate it from the system.

²We understand *Victron* system as all electronic equipment provided by *Victron Energy LTD*.

³We understand *Cerbo GX* as a proprietary device from *Victron* that allows operating its electronic systems via the D-Bus protocol.

⁴The bulk phase is the first phase of battery charging in which it is charged as much as possible. It is followed by the absorption and float phases

⁵total production less than 5% of the installed capacity

Project management

- (HU-33) As a system administrator, I want to add thresholds to devices so that the system can integrate rule-based systems⁶.
- (HU-34) As a system administrator, I want to add notifications based on thresholds to devices so that users are alerted in exceptional situations.
- (HU-35) As a system administrator, I want to add users so that they can use the system.
- (HU-36) As a system administrator, I want to remove users so that they cannot use the system.
- (HU-37) As a system administrator, I want to modify users to reset their password when they forget it.
- (HU-38) As a system administrator, I want to view the properties of users to ensure data integrity.
- (HU-39) As a system administrator, I want to activate the emergency mode to stop all domestic consumption.
- (HU-40) As a system administrator, I want to add administrators to help me maintain the system.
- (HU-41) As a system administrator, I want to connect to the system to operate it in a straightforward manner.
- (HU-42) As a normal user, I want to set personalized energy saving goals to motivate a reduction in electricity use.
- (HU-43) As a normal user, I want to receive suggestions on optimizing battery usage based on my consumption patterns.
- (HU-44) As a normal user, I want to compare my energy consumption with that of similar users to gauge my efficiency.
- (HU-45) As a normal user, I want to manually adjust the charge and discharge cycles of the battery to optimize for energy rates.
- (HU-46) As a normal user, I want to receive predictive maintenance alerts for my battery system to prevent failures.
- (HU-47) As a system administrator, I want to integrate the system with local utility grids to optimize energy exchange.
- (HU-48) As a system administrator, I want to implement machine learning algorithms to predict peak load times and optimize battery usage.
- (HU-49) As a system administrator, I want to enable remote firmware updates for all connected devices to ensure that they operate with the latest features and security patches.
- (HU-50) As a system administrator, I want to monitor and manage the energy flow between the battery system, the solar panels and the grid to maximize efficiency and sustainability.
- (HU-51) As a system administrator, I want to configure automated responses to grid demand signals to participate in demand response programs.

⁶Such as rule-based Artificial Intelligence, state managers, etc.

3.1. System Requirements (User Histories)

- (HU-52) As a normal user, I want to access detailed analytics on my energy production, consumption, and battery health over time for better decision making.
- (HU-53) As a system administrator, I want to have a dashboard that provides real-time monitoring of all connected devices' statuses and performance metrics.
- (HU-54) As a system administrator, I want to configure and manage virtual power plants (VPP) functionalities to aggregate distributed energy resources for grid services.
- (HU-55) As a normal user, I want to participate in community energy sharing programs to sell excess energy at peak times.
- (HU-56) As a system administrator, I want to ensure that all user and system data are encrypted and stored securely to protect against data breaches.
- (HU-57) As a normal user, I want to receive automated advice on the best times to charge my electric vehicle according to energy rates and solar production forecasts.
- (HU-58) As a normal user, I want to control and schedule my smart appliances based on battery status and energy production to optimize electricity use.
- (HU-59) As a system administrator, I want to deploy grid stabilization services using the battery storage system to contribute to the overall reliability of the electrical grid.
- (HU-60) As a system administrator, I want to have the ability to perform advanced analytics on energy data to identify trends, predict system demands, and optimize overall performance.
- (HU-61) As a normal user, I want to access a gamified interface that rewards energy-saving behaviors with points or rewards, fostering a community of energy-conscious users.
- (HU-62) As a normal user, I want to have voice command capabilities to interact with the system for convenience.
- (HU-63) As a normal user, I want to receive real-time alerts and guidance during power outages on how to manage my battery system for essential loads.
- (HU-64) As a system administrator, I want to implement a secure multifactor authentication system for user access to protect against unauthorized use.
- (HU-65) As a system administrator, I want to provide APIs for third-party integrations, allowing users to connect their own devices or applications for a more customized experience.
- (HU-66) As a system administrator, I want to enable advanced diagnostic tools for the troubleshooting and maintenance of the battery and connected devices.
- (HU-67) As a normal user, I want to set up automatic energy saving modes that adjust the behavior of the device based on my daily routines and preferences.
- (HU-68) As a normal user, I want to participate in virtual energy audits to receive personalized recommendations to improve home energy efficiency.
- (HU-69) As a system administrator, I want to configure and manage microgrid operations for communities or commercial complexes to ensure energy resilience and sustainability.

Project management

- (HU-70) As a system administrator, I want to analyze and report on the reduction of carbon footprint achieved through the system to support sustainability goals.
- (HU-71) As a normal user, I want the system to automatically adjust the battery charge and discharge rates based on forecasted weather conditions to optimize the anticipated energy production and consumption.
- (HU-72) As a normal user, I want to integrate the battery system with my smart home security system to ensure that essential security features remain powered during outages.
- (HU-73) As a system administrator, I want to use blockchain technology to securely log and verify all transactions related to energy trading within the system.
- (HU-74) As system administrator, I want the system to automatically detect and isolate faulty devices or circuits to prevent system-wide problems and facilitate prompt maintenance.
- (HU-75) As a normal user, I want to customize which devices or loads are prioritized during battery backup power situations to ensure that my most important needs are met first.
- (HU-76) As a normal user, I want to participate in peer-to-peer energy trading platforms to buy and sell excess energy directly with neighbors or within my community.
- (HU-77) As a system administrator, I want to incorporate artificial intelligence to dynamically adjust energy storage and release strategies based on real-time demand and supply conditions.
- (HU-78) As a normal user, I want the system to provide me with learning resources and tips to reduce my energy footprint and maximize the use of renewable energy sources.
- (HU-79) As a system administrator, I want the system to automatically participate in national or regional demand response programs, adjusting energy usage in response to grid requests.
- (HU-80) As a normal user, I want to have a mobile app that gives me full control and monitoring capabilities of the battery system, including remote adjustments and real-time notifications.
- (HU-81) As a system administrator, I want to integrate advanced sensors and IoT devices to better and more accurately monitor system performance and environmental conditions.
- (HU-82) As a normal user, I want the system to automatically switch to the most cost-effective energy source based on real-time prices and my pre-defined preferences.
- (HU-83) As a normal user, I want to establish and participate in local energy cooperatives through the platform, which promotes community engagement and collective energy solutions.
- (HU-84) As a system administrator, I want to receive system health reports regularly, including predictive maintenance alerts, to proactively address potential problems before they escalate.

3.1. System Requirements (User Histories)

- (HU-85) As a normal user, I want the interface to adapt to my usage patterns and preferences, offering a personalized dashboard that highlights the information most relevant to me.
- (HU-86) As a normal user, I want to access a feature that allows me to automatically lend my unused battery storage to the grid during peak demand times in exchange for credits or reduced utility rates.
- (HU-87) As a normal user, I want to receive notifications about solar projects in the local community that I can join to reduce my carbon footprint and save on electricity bills.
- (HU-88) As a system administrator, I want the ability to configure and manage a fleet of batteries as a single virtual power plant, optimizing their collective performance for grid support services.
- (HU-89) As a system administrator, I want to implement a loyalty program that rewards users for their contributions to energy efficiency and community energy projects.
- (HU-90) As a normal user, I want to visualize the impact of my energy use and savings in terms of CO2 emissions avoided, supporting my environmental awareness.
- (HU-91) As a system administrator, I want to integrate the system with national and regional energy databases to access up-to-date information on the energy mix, facilitating more informed energy-sourcing decisions.
- (HU-92) As a normal user, I want to have adaptive learning features in the app that suggest the best times for energy consumption based on my habits and the available renewable energy.
- (HU-93) As a system administrator, I want to deploy drones to inspect remote or difficult-to-access components of the energy system, enhancing maintenance efficiency and system reliability.
- (HU-94) As a normal user, I want to participate in a forum within the app to share tips, discuss energy saving strategies and connect with like-minded people.
- (HU-95) As a normal user, I want to receive personalized energy savings challenges each month, with the system tracking my progress and rewarding achievements to keep me engaged and motivated.
- (HU-96) As a system administrator, I want to have an advanced simulation tool that can model the potential impacts of new policies, technologies, or user behaviors on the system's efficiency and sustainability.
- (HU-97) As a normal user, I want the ability to control and optimize the charging of my electric vehicle based on the availability of renewable energy and my schedule.
- (HU-98) As a system administrator, I want to create detailed reports that analyze the system's performance, user engagement, and environmental impact for stakeholders.
- (HU-99) As a normal user, I want to access a virtual assistant within the app that can answer questions about my energy use and provide guidance on how to improve efficiency.

Project management

- (HU-100) As a system administrator, I want to integrate the system with emerging renewable energy technologies, such as wave and geothermal, as they become commercially viable, ensuring the system remains at the cutting edge of sustainable energy solutions.
- (HU-101) As a user administrator, I want to make prompt questions to the system so that I don't have to use the panel.

3.2 Versioning

For the development, the agile Scrum methodology will be used, which is based on versioning, as the team consists of one developer (dissertation's writer) and one product owner (dissertation's mentor).

As described, following the traditional strategy, the first number of the version indicates the major version, which may involve backward compatibility changes in the software with respect to other major versions. The second number indicates a minor version and maintains backward compatibility with the major version. The third number (if there were one) patches errors on the minor version.

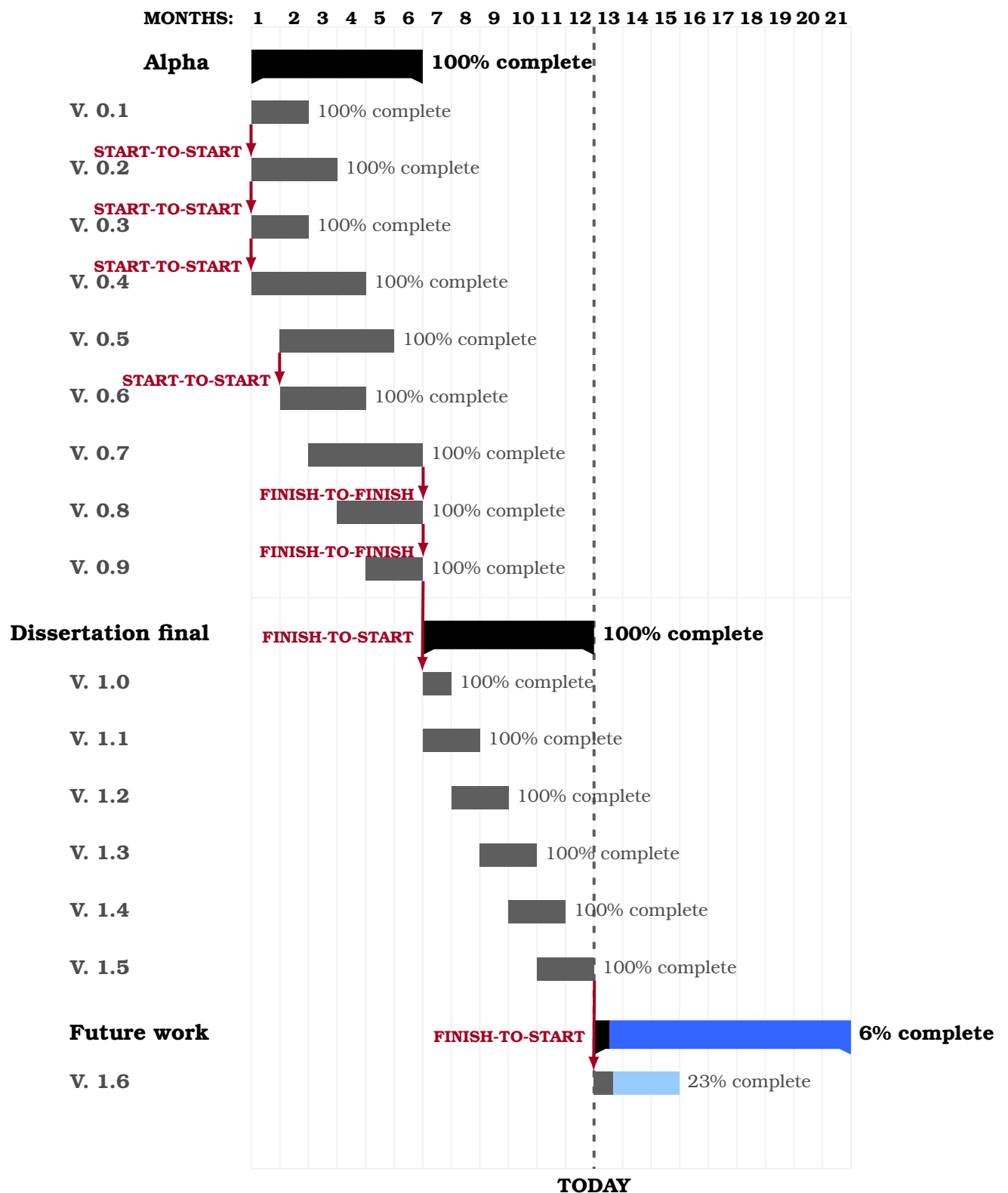
The user stories will be chosen using their unique identifier and assigning such a user story to each version to close it as shown in table 3.1.

Version	Items
0.1 - ALPHA	HU-27, HU-25, HU-24, HU-40, HU-41
0.2	HU-33, HU-32, HU-31, HU-23
0.3	HU-39, HU-11, HU-17, HU-18
0.4	HU-19, HU-20, HU-21, HU-22
0.5	HU-6, HU-7, HU-15, HU-26
0.6	HU-34, HU-16, HU-35, HU-36
0.7	HU-8, HU-9, HU-10, HU-28
0.8	HU-1, HU-2, HU-3, HU-4
0.9	HU-5, HU-12, HU-13, HU-14
1.0 - BETA	HU-29, HU-30, HU-37, HU-38
1.1	HU-87, HU-76, HU-84, HU-92
1.2	HU-60, HU-52, HU-98, HU-88
1.3	HU-40, HU-45, HU-81, HU-74
1.4	HU-42, HU-58, HU-96, HU-100
1.5 - Diss. final	HU-99, HU-39, HU-90, HU-57
1.6	HU-65, HU-62, HU-59, HU-67
1.7	HU-79, HU-80, HU-94, HU-93
1.8	HU-70, HU-89, HU-48, HU-86
1.9	HU-64, HU-44, HU-46, HU-68
2.0	HU-43, HU-53, HU-82, HU-97
2.1	HU-73, HU-72, HU-75, HU-66
2.2	HU-69, HU-95, HU-51, HU-61
2.3	HU-77, HU-50, HU-54, HU-91

Table 3.1: Versioning with specific user histories for each. Property of Andrei García Cuadra for this dissertation.

3.3 Gantt Chart

Below is the Gantt chart given in 3.3 of the given specifications. Note some of the versions are related to future work.



3.4 Next upgrades

This work will serve as the basis for my doctoral thesis, which focuses on real-time communications and operational synchronization. Furthermore, this dissertation is intended to cover only a handful of user stories, as the implementation will continue beyond this point. This project has been showcased not just to ActuaUPM, but also to other innovation competitions, in an effort to secure funding to establish the company. The primary goal of the project is to transition to an industrial setting, rather than being limited to domestic applications.

3.5 System design

First, the system is presented using a **flow operational diagram**. It is essential to note that the system is a mix of *on-premise systems* and *cloud systems*. This is due to the fact that local devices must standardize the way they communicate, lowering operational costs and minimizing transactions while granting robustness to the system. Thus, on-premise systems will be the first to be explained, and by using figure 3.2 it can be assumed that **BMS** is the main data generator here, as we have ignored other data generators such as MPPT controllers or inverters for massive data generation in the current stage since the main focus of this project is related to battery management. Then, these data are synced to the **B-Box** which is fully developed with Raspberry as stated in 5.2 while operating IoT devices based on real-time data as referred to in 5.3.

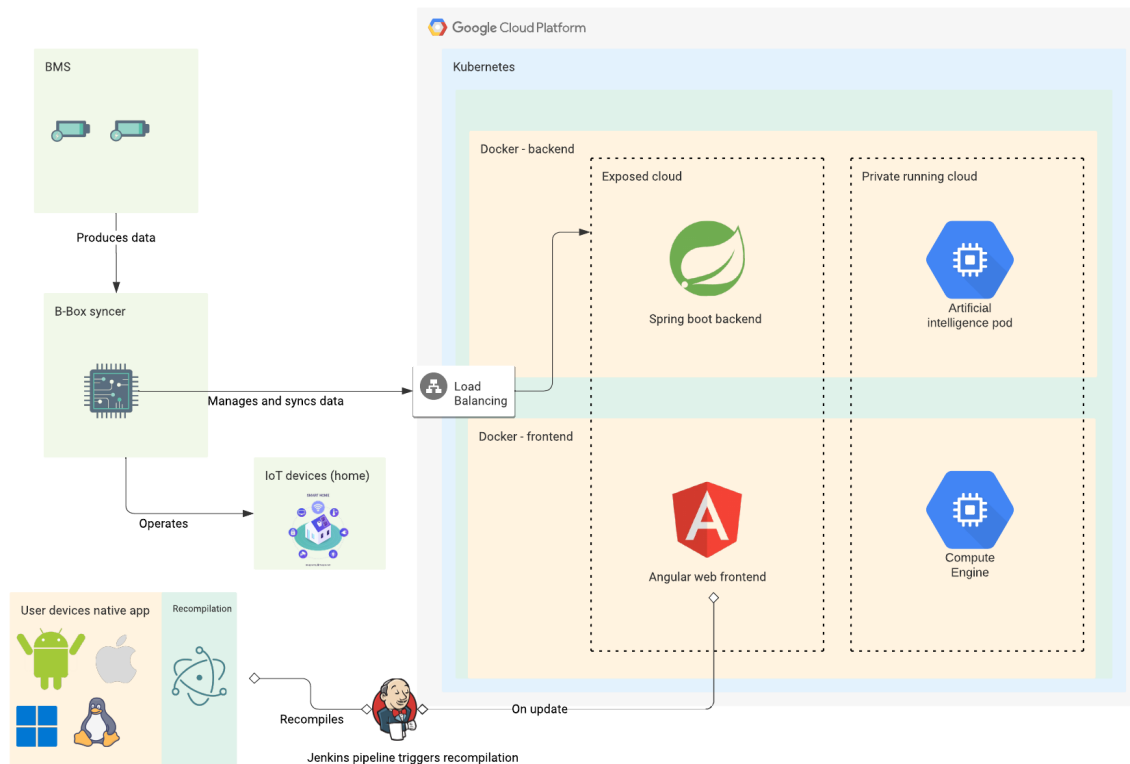


Figure 3.2: System operational showcase diagram. Image property of Andrei García Cuadra for this dissertation.

Focusing now on cloud systems, there are three main differentiations here. At first, there is a *load balancer* that handles every request to the backend (handled in Section 4.2) in order to guarantee a 100% uptime on the API layers while they run inside a Kubernetes cluster that has auto-scaling (operable with load balancing). The load balance is not needed for the Angular frontend handled in section 4.3) because the application self-downloads on the local user browser, so requests are only loaded

when there is an update (the frontend pod auto-scales in that case) or when the user accesses the web for the first time.

Regarding the *artificial intelligence pod*, it is covered in a few chapters regarding battery predictions and management such as **Passive balancing** 5.2.8, **Active balancing** 5.2.2, **Battery degradation prediction** 5.2.3, and **Battery Lifespan Prediction** 5.2.4. Those operations cannot be executed directly from the B-box, so the API acts as an *Apache Kafka broker* which is linked with the artificial intelligence pod to execute compute operations and is responsible for listening to the results from those different operations to sync them on the database (database engines are covered in 4.2.2).

3.6 Programming Principles

Programming principles are crucial for developing robust, maintainable, and scalable software. This section of the document will cover foundational principles in object-oriented design, such as the SOLID principles, and architectural patterns that support complex application developments like microservices and domain-driven design.

3.6.1 SOLID Principles

The SOLID principles are a set of guidelines referred in [Sarcar(2022)] that aim to promote scalability, avoid code smells, and facilitate easier refactoring of software. These principles are essential for developing applications that must be easy to maintain and extend over time. Implementation of these principles is checked during Continuous Integration/Continuous Deployment (CI/CD) processes to ensure all code adheres to these high standards.

- **Single Responsibility Principle (SRP):** Ensures a class has only one reason to change, emphasizing separation of concerns.
- **Open/Closed Principle (OCP):** Entities should be open for extension but closed for modification, preventing changes from introducing errors.
- **Liskov Substitution Principle (LSP):** Subtypes must be substitutable for their base types, enhancing the robustness of the software.
- **Interface Segregation Principle (ISP):** Clients should not be forced to depend on interfaces they do not use.
- **Dependency Inversion Principle (DIP):** High-level modules should not depend on low-level modules; both should depend on abstractions.

3.6.2 Architectural Patterns

Architectural patterns are predefined and reusable solutions to common problems encountered in software architecture. They provide a template for designing and building software systems, ensuring best practices and a solid structure. As stated in the book [Taibi et al.(2018)], below patterns are the critical ones chosen for this project:

- **Model-View-Controller (MVC):** Separates application into three parts, increasing modularity and ease of maintenance.
- **Publisher-Subscriber Pattern:** Allows services to communicate effectively without being directly linked, enhancing scalability.
- **Saga Pattern:** Manages data consistency across microservices using a series of local transactions.
- **Singleton Pattern:** Ensures a class only has one instance and provides a global point of access to it.
- **Facade Pattern:** Provides a simple interface to a complex subsystem, simplifying the interactions.

3.6.3 Importance of Architecture in Battery Management

In the context of battery management systems, implementing these programming principles and patterns helps in creating a structured and efficient architecture. It allows the system to manage energy consumption, extend battery life, and ensure robustness against various operational challenges. Proper application of these principles ensures that the system can adapt to changes in technology and requirements without significant disruptions or redesigns.

Chapter 4

Cloud implementation

The deployment of cloud technologies in the system architecture serves multiple strategic and operational advantages, essential for achieving the ambitious goals of the project. The integration of cloud solutions provides the following benefits:

- **Scalability:** Cloud platforms offer seamless scalability which is vital for handling the fluctuating and increasing data volumes generated by BMS and electrical systems. This scalability ensures that the system can adapt to the growing output of data and the demand of the users without the need for significant hardware upgrades.
- **High Availability and Reliability:** Utilizing cloud services enhances the reliability of the system. Cloud providers implement redundant resources to ensure service continuity even in the event of component failures, thereby supporting the project's requirement for high system uptime.
- **Cost-Effectiveness:** Cloud computing reduces the need for upfront capital investment in hardware. It provides a pay-as-you-go model, which is economically beneficial for managing costs, especially important for a project that might scale in terms of both functionality and geographic distribution.
- **Advanced Analytics and Machine Learning:** The cloud enables the use of sophisticated analytics and machine learning services that can be integrated directly into the cloud environment, facilitating real-time data processing and intelligence extraction vital for predictive maintenance and optimization.
- **Global Deployment:** Cloud platforms can deploy services across multiple geographical regions, reducing latency for users distributed globally and complying with local data regulation policies.

4.1 DevOps

As part of the comprehensive approach to managing the battery self-management system lifecycle, implementation of **DevOps** practices is crucial. **DevOps** integrates software development (**Dev**) and IT operations (**Ops**) into a unified framework, improving productivity of the development process. This project automate the deployment and monitoring while managing both software and infrastructure.

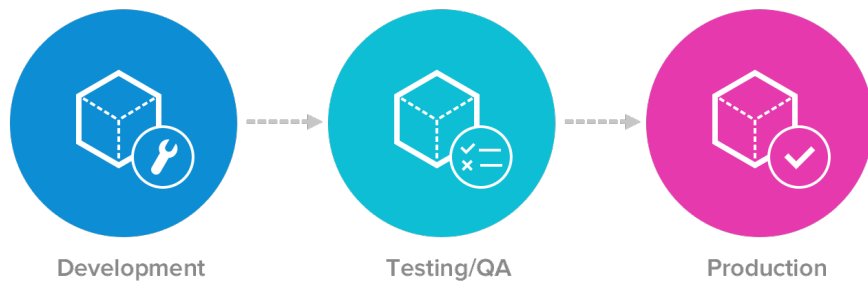


Figure 4.1: Environment staging. Image property of Andrei García Cuadra

It is strongly needed to use DevOps tools to enhance and ensure that IoT system deployments are updated with a full integrity on all devices, while battery management is safe and state-garanteed by giving failover, resilience... That will be detailed in section 4.1. It is important to note that as shown in figure 4.1 project is based on three different kind of environments: development, testing and production.

While development is used for developers to make new functionalities, patches and some manual tests in the local machine, the testing environment is a semi-productive environment deployed on google cloud that reflects the behaviour of production, so in case there is any issue on the deployment this will mirror the behaviour of production without affecting the stability and service of productive environments, which are essentially those that are accesed by clients and final users of the application.

4.1.1 Google Cloud

Google Cloud provides highly scalable cloud infrastructure, which allows our project to dynamically adjust resources based on demand. This is crucial for handling varying loads efficiently without the need for physical hardware investments. All the knowledge of this section with regard to Google Cloud has been extracted from [Krishnan and Gonzalez(2015)] and aims to generate an overview of the technology and then explain the implementation.

Google Cloud offers a pay-as-you-go pricing model which ensures that the project only pays for the computing resources actually used. This approach helps to manage costs more effectively, which is particularly beneficial for projects in which data are generated based on an unexpected demand.

Security is paramount in DevOps practices, and Google Cloud provides robust security features that protect data and maintain privacy. With automatic security updates and threat detection, the platform ensures that the project's data are protected against potential cyber threats and accelerates the development cycle by providing tools for continuous integration and continuous delivery (CI/CD). As shown in figure 4.2, the standard way google cloud handles IoT data is mostly based on asynchronous data pipes across multiple stateless services.

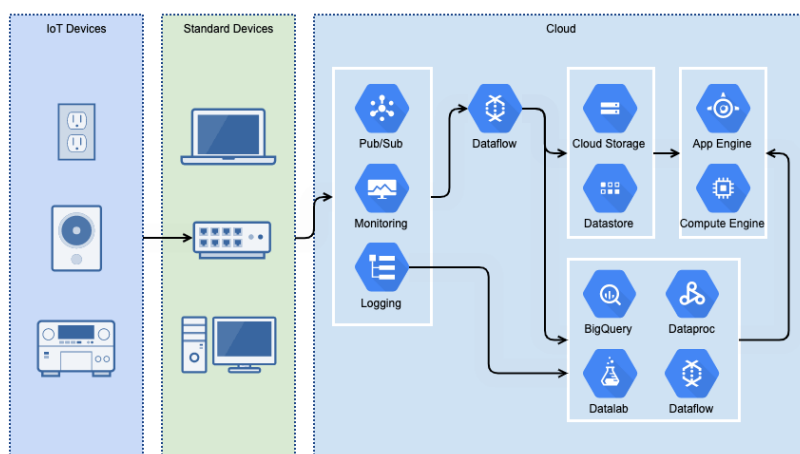


Figure 4.2: Ideal google cloud platform diagram to handle data. Derived from official documentation of [Google(2024a)].

Google Cloud provides robust infrastructure and advanced analytics capabilities that significantly enhance the efficiency and reliability with its powerful data processing tools, enabling real-time monitoring and predictive analytics, which are essential for optimizing battery performance since its global network of data centers ensures high availability and low latency for IoT applications. This, coupled with strong security measures to protect sensitive data, makes the Google Cloud a vital component in modern battery management strategies.

4.1.2 Kubernetes and Docker

Kubernetes, an open-source container orchestration platform, enhances our project's ability to manage containerized applications at scale. Automates the deployment, scaling, and operations of application containers across clusters of hosts. Knowledge of this section has been extracted from [Baier et al.(2019)] and it aims to generate an overview of the technology and then explain the implementation.

Docker containers wrap software in a complete filesystem that contains everything needed to run: code, runtime, system tools, and libraries. This guarantees that the application will always run the same, regardless of the environment in which it runs, leading to fewer bugs and discrepancies between the development and production environments. Both Kubernetes and Docker enhance development efficiency by isolating applications into containers, making them portable and easy to deploy. This isolation simplifies dependency management and conflicts between working environments so that more development is focused on setting up the environment.

Kubernetes excels at managing the scalability and availability of applications. It can automatically adjust the number of running containers based on the demand, without any downtime, ensuring that our application can handle increased loads efficiently and reliably.

Using Docker and Kubernetes together allows for more efficient use of system resources. Containers require less overhead than traditional virtual machines, and Kubernetes ensures that resources are used efficiently, avoiding resource wastage by balancing loads effectively across the infrastructure as shown in figure 4.1.2.

Using Kubernetes for battery implementation is crucial because it enables robust and scalable management of battery systems within IoT frameworks. Kubernetes, a container orchestration platform, allows for the deployment, management, and scaling of applications, which is essential for handling the complex and dynamic workloads associated with battery management systems. This is particularly important for battery systems that require real-time data processing and adaptive management to optimize performance and extend battery life. Additionally, Kubernetes supports high availability, fault tolerance, and resource efficiency. This ensures that battery systems remain operational and efficient, contributing to sustainability and cost savings.

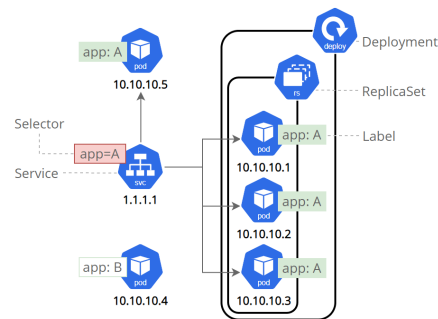


Figure 4.3: *K8S Pod handling example schema* by [Google(2024c)].

4.1.3 Grafana

Grafana is an open source platform used extensively to monitor real-time data on applications and infrastructure. It provides powerful visualization tools that transform raw data into meaningful insights, which are crucial for monitoring the health and performance of systems. All the knowledge of this section with regard to Grafana has been extracted from [Salituro(2020)] and it aims to generate an overview of the technology and then explain the implementation.

Grafana offers highly customizable dashboards that can be tailored to the specific needs of our project. These dashboards allow team members to focus on the metrics that matter most, allowing for quick identification of issues and trends that could affect system performance and stability. One of the key features of Grafana is its robust alerting system, which notifies the team of potential problems before they escalate into critical problems. This proactive approach helps to maintain high availability and performance. Grafana supports integration with a multitude of data sources (such as Prometheus). This flexibility allows it to be a central tool for monitoring across various environments and platforms, providing a unified view of the metrics across different segments of the project. Grafana allows collaboration among team members by allowing shared access to dashboards. Grafana's implementation in our DevOps project significantly improves the monitoring and observability landscape, offering customizable visualizations, effective alerting mechanisms, and broad integration capabilities.

Using Grafana for battery implementation, especially within IoT architectures, is crucial due to its powerful data visualization and monitoring capabilities. Grafana allows for real-time monitoring of battery metrics such as voltage, current, temperature, and capacity, which are essential for ensuring optimal battery performance and longevity. It supports the detection of anomalies and trends that could indicate potential issues, enabling preventive maintenance and reducing downtime. Furthermore, Grafana's ability to integrate with various data sources and its flexible dashboard creation tools provide a comprehensive view of battery operations across different systems and locations.

4.1.4 Jenkins

Jenkins automates the building and testing of software, significantly speeding up development cycles and ensuring that all code commits are automatically built and tested in real-time. This reduces the integration problems commonly associated with commits. All the knowledge of this section regarding to Jenkins has been extracted from [Smart(2011)] and it aims to generate an overview of the technology and then explain the implementation. Jenkins is fundamental in implementing CI/CD pipelines, which are crucial for modern software development practices. By automating the stages of software delivery, Jenkins ensures that software can be released reliably at any time and with reduced risk. This automation extends from code integration to testing, and from deployment to delivery.

Through Jenkins, I receive feedback of my work, which is vital for correcting issues promptly and effectively, notifying detailed reports on the building status, test results, and deployment processes, which helps in identifying and addressing errors quickly. Jenkins supports a vast array of plugins, making it highly adaptable to various tools and technologies. This extensibility allows Jenkins to integrate seamlessly into almost any development environment, supporting tools for version control, build management, containerization, and more.

Jenkins can handle projects of any size and can be scaled up to work across multiple machines for larger projects, increasing its flexibility. It supports both distributed builds and cloud-based architectures.

Integration of Jenkins into our DevOps project streamlines the software development lifecycle by automating builds, tests, and deployments. Its robust plugin system, scalability, and ability to provide quick feedback significantly contribute to increased development efficiency, faster time to market, and enhanced product quality.

Using Jenkins for battery implementation is particularly beneficial due to its robust capabilities in continuous integration and delivery (CI/CD), which are essential for developing and deploying battery management software efficiently. Jenkins automates the testing and deployment processes, ensuring that new code contributions are immediately tested and errors are quickly identified. This is crucial for battery management systems where updates might include critical safety or efficiency improvements. Additionally, Jenkins supports a wide range of plugins, allowing it to integrate seamlessly with various tools and environments used in battery development, from simulation software to hardware testing frameworks. By enabling faster iterations and ensuring high-quality deployments, Jenkins significantly enhances the reliability and performance of battery systems.

4.1.5 SonarQube

SonarQube provides a comprehensive suite of tools to automatically scan and identify code quality issues. By integrating with our continuous integration pipeline, it performs automated reviews with static code analysis to detect bugs, code smells, and security vulnerabilities in the codebase. All the knowledge of this section with respect to SonarQube has been extracted from [Campbell and Papapetrou(2013)] and aims to generate an overview of the technology and then explain the implementation. SonarQube assesses and reports code complexity, duplication, and coding standards. These insights help developers improve the maintainability of code by refactoring complex codes, reducing duplications, and adhering to coding best practices, thus ensuring a clean and manageable codebase. With its robust rule engine, SonarQube identifies security vulnerabilities within the code, thereby preventing potential security breaches before they occur. This proactive security assessment is crucial for maintaining the integrity and security of the software. SonarQube seamlessly integrates into our existing development workflows. It works alongside other tools in our CI/CD pipeline, providing developers with immediate feedback on the quality of their commits. This integration encourages a continuous code improvement mindset among team members. SonarQube offers comprehensive dashboards that provide actionable insights and detailed reports on code quality metrics. These dashboards are essential for project stakeholders to monitor the health of the project codebase and make informed decisions about project management and resource allocation. The integration of SonarQube into our project has profoundly influenced our ability to maintain high standards of code quality. Its continuous analysis capabilities, combined with its extensive security and maintainability checks, have significantly contributed to reducing the overall technical debt and improving the security posture of our applications. As a result, SonarQube has become an indispensable tool in our DevOps toolchain, fostering a culture of quality and excellence in software development.

Using SonarQube for battery implementation in software systems. SonarQube helps ensure code quality and security, which is paramount in systems that manage critical resources such as energy. It automatically scans and analyzes the source code for potential bugs, security vulnerabilities, and code smells, thus preventing possible failures that could lead to system downtimes or unsafe states in battery management systems. Moreover, SonarQube's continuous inspection capability means that it integrates seamlessly into the CI/CD pipelines, ensuring that code quality checks are part of the development process and not an afterthought. This ongoing feedback loop not only enhances the stability and efficiency of battery management software but also supports compliance with industry standards and best practices, reducing technical debt and improving maintainability of the codebase. In essence, using SonarQube is about preempting problems before they manifest in the operational environment, which is essential for maintaining the reliability and safety of battery-dependent systems.

4.1.6 GitLab

GitLab's built-in CI/CD features automate the process of software testing and deployment. By facilitating continuous integration practices, GitLab helps identify integration issues early, reduce bugs, and increase the overall quality of the software. Continuous deployment capabilities allow for seamless and frequent releases, ensuring that updates are delivered quickly to end users. All the knowledge of this section with regard to GitLab has been extracted from [Paiva(2021)] and it aims to generate an overview of the technology and then explain the implementation. GitLab improves team collaboration by integrating issue tracking with the Git repository. This allows team members to track the progress of tasks and bugs directly linked to the corresponding code changes. The Merge Request (MR) feature fosters code review practices, enhancing code quality, and promoting knowledge sharing among team members. GitLab provides extensive security scanning tools, such as static application security testing (SAST), dynamic application security testing (DAST), and dependency scanning, which are integrated into the CI/CD pipelines. These tools help identify and fix security vulnerabilities before the software is deployed, ensuring compliance with security best practices and regulations. Its robust architecture can handle significant loads, which is critical for high transaction volumes and complex projects. This scalability ensures that GitLab grows with the project without compromising performance. The use of GitLab has significantly influenced our project by streamlining DevOps processes and strengthening security practices. These improvements have led to faster development cycles, higher software quality, and a more efficient and secure development environment.

GitLab provides an integrated DevOps platform that supports the entire development lifecycle, from planning and coding to testing, deploying, and monitoring. This integration is vital for battery implementation projects which require continuous integration and deployment (CI/CD) pipelines to ensure that updates and improvements are systematically and safely implemented into the production environment. Additionally, GitLab's robust version control system allows for meticulous tracking of changes, which is essential for maintaining the integrity and reliability of software that manages critical battery operations. This ensures that the battery management systems are not only efficient but also incorporate the latest safety protocols and performance optimizations, ultimately leading to more reliable and sustainable battery usage in IoT devices.

4.1.7 Keycloak

All the knowledge of this section with respect to KeyCloack has been extracted from [Thorgersen and Silva(2021)], and it aims to generate an overview of the technology and then explain the implementation. Keycloak provides a centralized platform for managing user identities and access, simplifying the process of user authentication and authorization across multiple applications and services. This centralization eliminates the need for repeated authentication procedures and reduces the complexity associated with managing multiple identity stores. Keycloak supports a variety of authentication mechanisms, including standard username and password, two-factor authentication, and social logins. This flexibility allows us to implement robust security measures tailored to the needs of the project, enhancing overall security posture. Keycloak facilitates single-sign-on (SSO) across different applications, which enhances user experience by requiring users to log in only once to access multiple services. Furthermore, Keycloak's session management capabilities allow administrators to view and manage user sessions in real time, improving control over user access and security. Keycloak can be easily integrated with existing applications and services using standard protocols such as OpenID Connect and SAML 2.0. This ease of integration ensures that Keycloak can be adopted without significant changes to current systems, making it an ideal choice for enhancing identity management in legacy systems as well as new projects. Keycloak allows for detailed role-based access control (RBAC), which enables precise management of user permissions based on roles within the organization. Administrators can define and manage roles and permissions through an intuitive administrative console, ensuring that users have appropriate access rights to perform their tasks. Implementing Keycloak in our project has brought substantial improvements in managing user identities and access controls. Its robust authentication mechanisms, comprehensive SSO capabilities, and seamless integration with other systems have significantly increased both security and user satisfaction. As a result, Keycloak has become a cornerstone of our identity and access management strategy, supporting a secure and efficient operational environment as shown in flow diagram in figure 4.4.

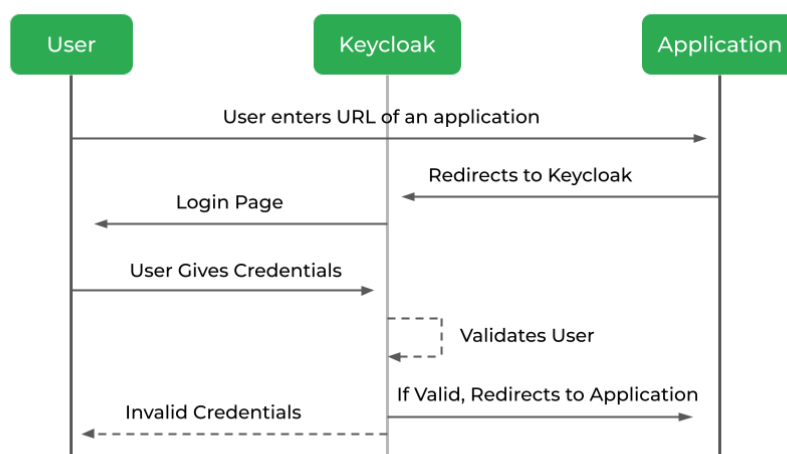


Figure 4.4: Keycloak flow diagram of auth handling by [KeyCloack(2024)].

4.1.8 OpenApi

OpenAPI, formerly known as Swagger, is an API specification framework that is widely used for designing, building, documenting, and consuming RESTful web services. The framework is language-agnostic and is used to create an interface that machines can understand, which details how software components should interact. All the knowledge of this section with respect to OpenApi has been extracted from [Ponelat and Rosenstock(2022)] and is aimed at generating an overview of the technology and then explaining the implementation.

OpenAPI is a specification and complete framework for describing, producing, consuming, and visualizing RESTful Web Services. This interface allows both humans and computers to discover and understand the capabilities of a service without requiring access to source code, additional documentation, or inspecting network traffic.

- **Machine Readable:** The OpenAPI specification is written in JSON or YAML format, making it easy for machines to parse and generate code.
- **Human Readable:** While being machine-readable, it is also structured to be human-readable and editable, bridging the gap between the technical and non-technical documentation.
- **Interactive Documentation:** Tools such as Swagger UI can interpret OpenAPI specifications to generate interactive API documentation that allows developers to perform API calls directly from their browsers.
- **Code Generation:** The OpenAPI specifications can drive code generation tools to produce server stubs, client libraries, and API documentation in various programming languages.
- **Design & Mocking:** enables API design and mocking directly from the specification before any actual coding begins, facilitating design-first API development.

OpenAPI is used in various stages of API development and maintenance:

- **API Design:** Developers can use OpenAPI specifications to plan and design APIs according to business requirements without writing any code.
- **Testing and Validation:** Automated tools can validate both the API documentation against the OpenAPI specification and the API responses against the definitions.
- **Client SDK Generation:** Automatically generate code for clients in various programming languages, enabling easy integration of services.

The modern approach involving systems based on Apache Kafka and real-time data processing frameworks marks a significant improvement over traditional methods:

- **Scalability and Performance:** Kafka provides a high-throughput, low-latency platform capable of handling real-time data feeds that is essential for operating at scale.
- **Data Integrity and Reliability:** Kafka's distributed architecture ensures that data is processed reliably, even in the event of system failures, thus enhancing fault tolerance and system resilience.

- **Stream Processing:** Real-time data streaming allows immediate data manipulation, analysis, and feedback, which is crucial for applications requiring real-time decision making and operational efficiency.

4.1.9 Redis

The integration of Redis into our cloud infrastructure plays a pivotal role in enhancing the performance and scalability of the system. Redis, as an in-memory data structure store, serves various functions such as caching, pub/sub messaging systems, and as a database. All the knowledge of this section with respect to Redis has been extracted from [Da Silva and Tavares(2015)] and aims to generate an overview of the technology and then explain the implementation. The main benefits provided by Redis in our system architecture are outlined below.

- **High Performance:** Redis enhances system performance by providing a high-speed, in-memory data cache. This reduces latency and improves response times for data retrieval, which is crucial for the real-time data processing requirements of our system.
- **Scalability:** With its ability to handle millions of keys per instance and support for clustering, Redis ensures that our system can scale horizontally to accommodate growing data from IoT devices and user demands.
- **Data Persistence Options:** Redis supports various persistence options, allowing us to maintain a balance between durability and performance, essential for data integrity and recovery.
- **Pub/Sub Capabilities:** The pub/sub features of Redis enable effective message broadcasting across different parts of the system, facilitating smooth and scalable communication patterns within the cloud architecture.
- **Atomic Operations:** Redis atomic operations help manage complex data types simply and effectively, ensuring data consistency across operations.

The integration of Redis supports the battery management system by ensuring fast data access and efficient message distribution, which is critical to the system's responsiveness and reliability. This setup is integral to managing real-time data flows and interactions within our IoT ecosystem, crucial for operational efficiency and system robustness.

4.1.10 Kafka

Apache Kafka is a pivotal component of our back-end infrastructure, acting as a distributed event-streaming platform designed to handle high volumes of data and enabling real-time analytics and data integration. In our system, Kafka serves as the central hub for messaging and stream processing. It is particularly crucial in environments where reliability, scalability, and performance are key. The architecture of Kafka allows it to efficiently process streams of data from multiple sources, distributing them across various consumer applications in real-time. All the knowledge of this section with regard to kafka has been extracted from [Shapira et al.(2021)] and aims to generate an overview of the technology and then explain the implementation.

- **Scalability:** Kafka's distributed nature allows it to scale out and handle millions of messages per second. It supports horizontal scaling without downtime, accommodating growing data volumes seamlessly.
- **Durability and Reliability:** Kafka replicates data and can persist data to disk, ensuring that data are not lost and is available for reprocessing if necessary. This is crucial to maintaining data integrity in our IoT ecosystem.
- **Performance:** Kafka boasts high throughput for both publishing and subscribing, with low latency, making it suitable for mission-critical tasks.
- **Real-Time Processing:** Kafka is often used in conjunction with stream processing frameworks such as Apache Flink or Spark Streaming, providing real-time analytics and decision-making capabilities.

In our deployment, Kafka integrates with various components of the system, including IoT devices, microservices, and data storage solutions. Orchestrates the flow of information between these components, enabling efficient and reliable data exchange and processing. Using Kafka, our system achieves enhanced responsiveness and agility, which is essential for the dynamic requirements of battery self-management and AI-driven predictive analytics.

Within the IoT domain of our project, Kafka plays a vital role in collecting and routing sensor data from numerous devices across the network. It ensures that the vast streams of data generated by sensors and devices are efficiently aggregated and available for real-time processing and analysis.

- **Data Aggregation:** Kafka acts as a single source of truth for the incoming data streams, providing a unified platform for data ingestion.
- **Fault Tolerance:** The built-in fault tolerance ensures that the system can handle device failures or network issues without loss of data.
- **Stream Processing:** With Kafka, data streams are processed on the fly, enabling immediate insights and actions based on real-time data.

The integration of Kafka within our system architecture enhances its ability to handle the complexities of modern IoT environments, where timely and reliable data processing is paramount. It provides the backbone for a robust, scalable, and efficient data pipeline that supports advanced analytics and decision-making processes in the context of battery management and energy optimization.

Kafka contributes to battery systems by providing a reliable and scalable framework

Cloud implementation

for data ingestion and processing. This is crucial in battery management for several reasons.

As shown in figure 4.5, communications are handled in a efficient way to keep all the data synced. This section could be all resumed using that diagram, which is self-explanatory having the knowledge acquired from this section.

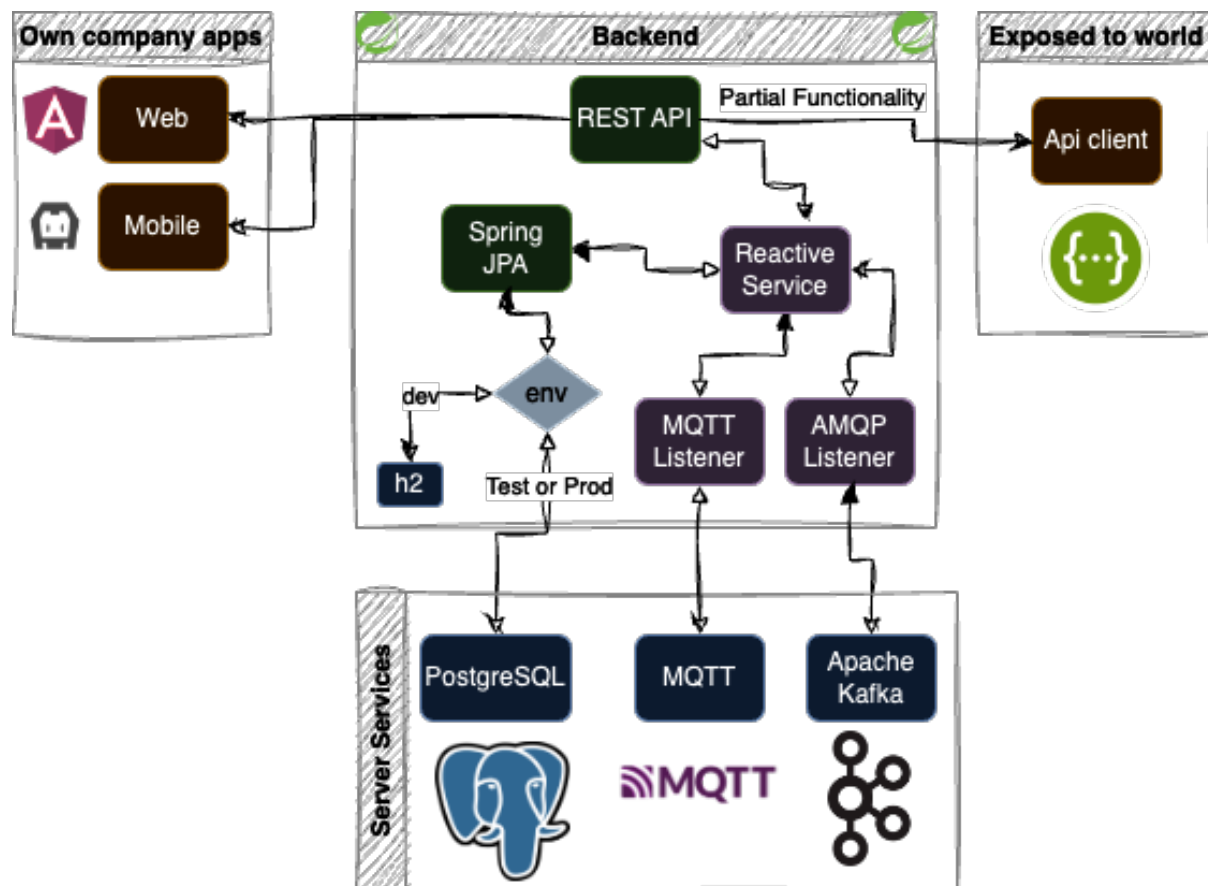


Figure 4.5: Architecture diagram for communications. Image property of Andrei García Cuadra.

4.2 Backend implementation

4.2.1 Spring boot with Java implementation

When considering the factors that will influence the final decision, several aspects of Spring Boot stand out. Firstly, its "convention over configuration" approach significantly reduces development time, enhancing productivity, and facilitating rapid application development through numerous out-of-the-box configurations. Furthermore, Spring Boot benefits from Java's robust ecosystem, providing a vast array of libraries and tools for development and testing, along with integrated support for both relational and NoSQL databases, web services, and security.

In terms of performance and scalability, Java, the language used with Spring Boot, is well known for its capabilities in these areas, making it ideal for high-load backend systems. Applications built with Spring Boot can be easily scaled to accommodate varying demands. Additionally, the platform enjoys extensive community support and an abundant knowledge base, bolstered by Java's large and active developer community and an array of documentation, tutorials, and guides.

Spring Security, integrated within Spring Boot, offers comprehensive security features, supporting authentication and authorization, and protecting against common vulnerabilities. The framework is also optimized for building microservices, with features like service discovery, configuration management, and resilience. The maturity and stability of Java, coupled with Spring Boot's own development, provide a reliable environment for crafting long-lived applications. Cross-platform development is another strong point, as Java applications, including those developed with Spring Boot, are platform independent at both the source and binary levels. Lastly, Spring Boot's comprehensive testing support allows for thorough testing of components, ranging from unit to full integration tests. As shown in below figure 4.6, classes are based on MVP pattern shown in section 3.6 and using Java principles with MVC pattern.

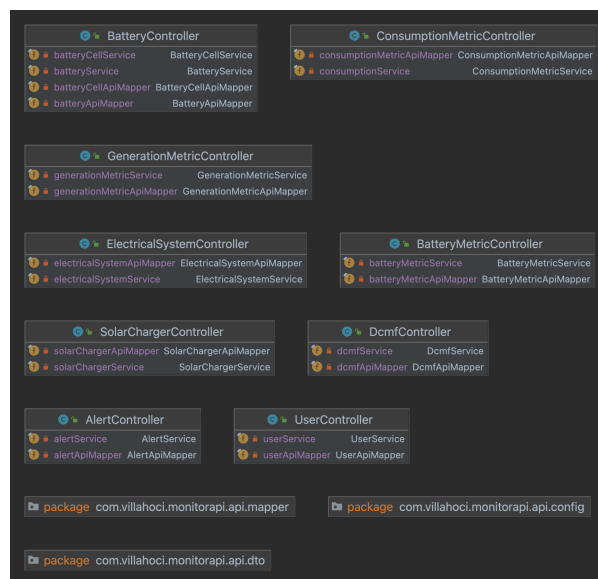


Figure 4.6: Class diagram of current project. Image property of Andrei García Cuadra.

4.2.2 Database and storage

4.2.2.1 PostgreSQL

PostgreSQL, often simply referred to as **Postgres**, is a powerful open-source object relational database system as all this section PostgreSQL knowledge was inferred from [Ginestà and Mora(2012)]. It has earned a strong reputation for its proven architecture, reliability, data integrity, robust feature set, and extensibility. For this project, PostgreSQL serves as the cornerstone for storing and managing the vast amount of data generated by IoT devices and AI algorithms. As shown in figures 4.7 and 4.8 relational model is given through this technology at all.

Key Features and Advantages

PostgreSQL offers several features critical for ensuring efficient data handling in IoT and AI-driven environments:

- **ACID Compliance:** Ensures that all transactions are processed reliably with Atomicity, Consistency, Isolation, and Durability.
- **Extensibility:** Users can define their own data types, add custom functions, and even write code from different programming languages without recompiling the database.
- **Concurrency:** Supports high levels of concurrency through Multiversion Concurrency Control (MVCC), essential for IoT environments where numerous devices perform data operations simultaneously.
- **Full-Text Search:** Integrated full-text search capabilities make it easier to perform complex queries on text data, which is invaluable for analysis purposes.

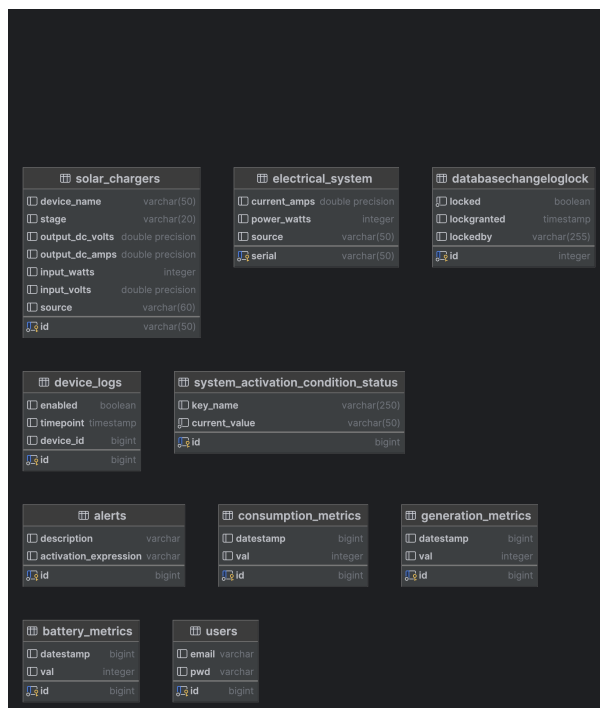


Figure 4.7: Entity relation diagram part 1. Image property of Andrei García Cuadra.

Role in the Project

In our system, **PostgreSQL** is utilized not just as a storage solution but as a dynamic component that interacts seamlessly with the backend applications:

1. **Data Storage:** All sensor data, user information, and transaction records are stored securely in PostgreSQL databases.
2. **Data Retrieval:** The AI algorithms retrieve historical data from PostgreSQL to learn and make informed predictions about battery life and energy usage.
3. **Real-Time Data Analysis:** Using PostgreSQL's advanced analytical functions, the system can perform real-time analysis to optimize energy consumption and predict system failures.

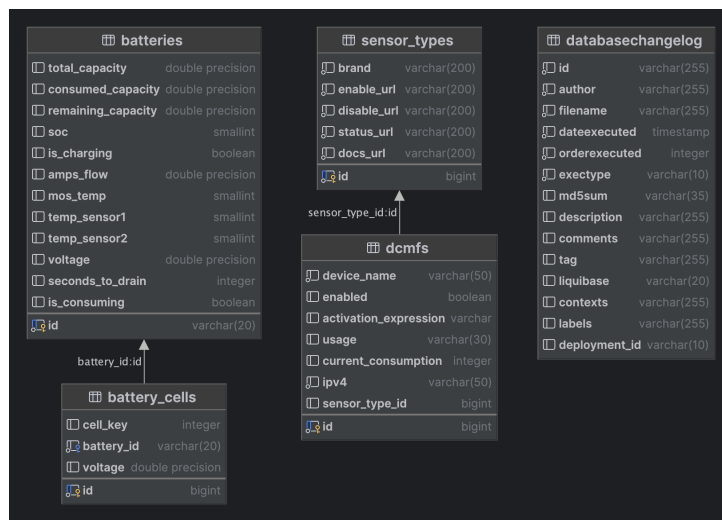


Figure 4.8: Entity relation diagram part 2. Image property of Andrei García Cuadra.

Cloud implementation

Also, there is important to note that some data needs to be persisted between versions and/or environment. This is where Liquibase takes action. Liquibase is an open-source database schema change management tool that streamlines the process of managing and applying database changes across various environments. It uses ChangeLog files, written in XML, YAML, JSON, or SQL, to define database changes in units called changesets, each uniquely identified to ensure accurate tracking. Liquibase maintains a special table, 'DATABASECHANGELOG', to record applied changesets, allowing it to execute only new changesets when a migration is run. It supports rollback operations, enabling easy reversion of changes if necessary. Integrated into CI/CD pipelines, Liquibase ensures consistent and automated application of database changes, enhancing collaboration among development, testing, and production environments as shown by [Dziadosz(2017)]. Current supported migrations are shown in figure 4.9 and based on XML technology since it allows to generate schemas without locking the database engine (interoperable between all SQL-like databases). Having this migrations ensures the project versioning and data will be always synced, reducing significantly errors and bugs overall and granting the capability of controlling code and database with no risks.

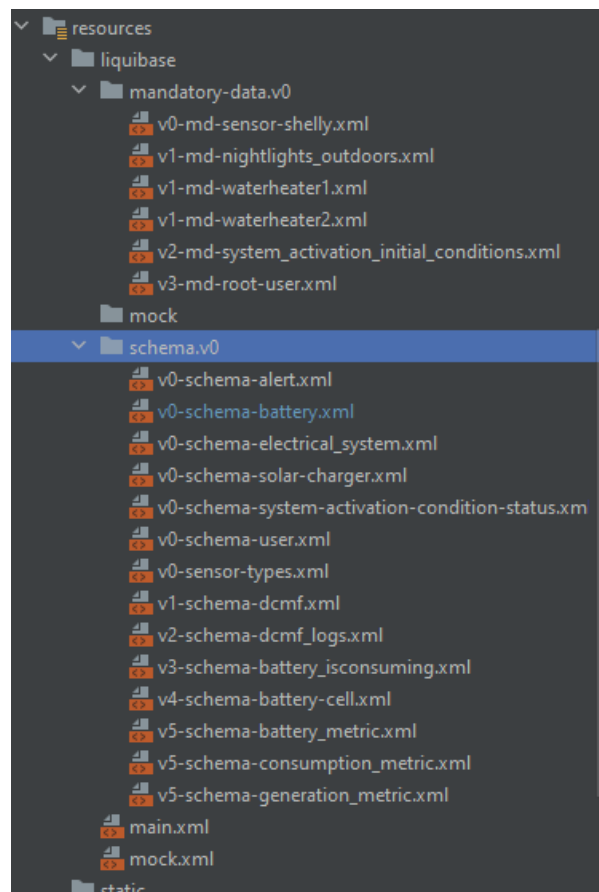


Figure 4.9: Current project migrations. Image property of Andrei García Cuadra.

4.2.2.2 MongoDB

MongoDB is utilized within the system's architecture primarily as a NoSQL database solution, offering flexible data models and scalable operations critical for handling large volumes and diverse forms of data generated by IoT devices. This section outlines the role and configuration of MongoDB in the system, emphasizing its strategic importance and integration details as stated by [Banker et al.(2016)].

MongoDB serves as a dynamic, schema-less database that excels in managing semi-structured data which is typical in IoT environments. Its document-oriented data structure allows for storing varied data types without predefined schema constraints, making it ideal for:

- *Rapid development and iteration:* Changes to the database schema do not require downtime, facilitating agile development and continuous integration/deployment processes (More places to cover with IoT devices).
- *Horizontal scalability:* MongoDB supports sharding, enabling horizontal scaling across multiple servers and improving performance and redundancy (More operations per sensor).
- *High write loads:* It handles high throughput and large volumes of data, essential for telemetry data from numerous IoT devices (More number of sensors).

Deployment Configuration: MongoDB is deployed in a replicated cluster to ensure data redundancy and high availability. This setup includes:

- Multiple replica sets to provide automatic failover and data redundancy.
- Config servers that manage the metadata of the cluster, enabling efficient data querying and sharding operations.

In summary, MongoDB's flexible and robust nature makes it an integral component of the system's data handling capabilities, directly contributing to the system's scalability and resilience. Its deployment is tuned to the specific needs of handling large-scale IoT data, providing a foundation for real-time data processing and analysis critical for the operational success of IoT frameworks.

4.2.2.3 Nasuni

Nasuni is a cloud-native global file system that delivers scalable, secure, and cost-effective storage solutions for businesses dealing with large volumes of unstructured data. By leveraging cloud storage infrastructure, Nasuni provides a unified platform that consolidates Network Attached Storage (NAS) and file server silos, offering unlimited scale, built-in backup, global file sharing, and local server performance. It utilizes caching technology to ensure frequently accessed data is quickly available, while less active data is stored securely in the cloud, optimizing costs and efficiency. The motivation to choose Nasuni often stems from its ability to address the challenges of traditional file storage systems, such as capacity limitations, high costs, complex disaster recovery processes, and the need for seamless collaboration across distributed teams. Organizations are drawn to Nasuni for its innovative approach to file storage that not only meets the demands of modern data growth but also supports digital transformation strategies by enhancing data accessibility, protection, and governance across global enterprises as was inferred from the book [Hu et al.(2013)]

Cloud implementation

Since it is needed to have a bucket where we can upload files for retrieval such as json transactional files, cache files or whatever, Nasuni will cover transient issues for splitting large problems into smaller ones by dividing it. As shown in figure 4.10, our system architecture is quite similar but we won't publish it for safety concerns on security.

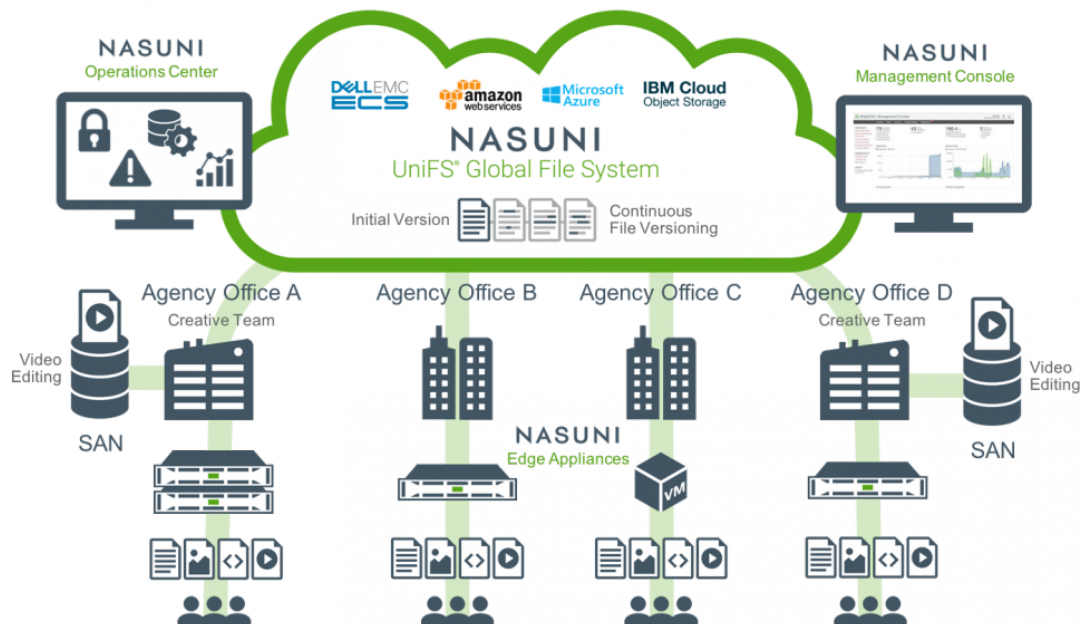


Figure 4.10: Nasuni distributed database example of [Nasuni(2024)]

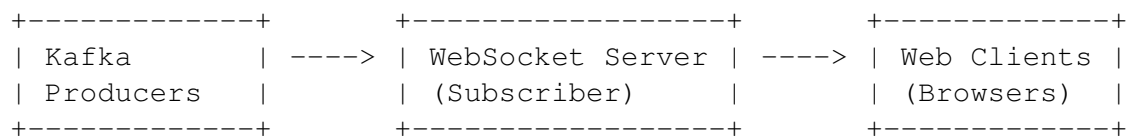
4.2.3 Transactional layer

Apache Kafka is a robust, scalable, and efficient streaming platform capable of handling high-volume data streams. When integrated with WebSocket, Kafka can redistribute messages to web clients in real-time, enabling dynamic data updates in web applications. This section explores the architecture and implementation strategy for combining Kafka with WebSocket to enhance data interaction layers in applications.

- **Publisher-Subscriber Model:** Kafka maintains feeds of messages in categories called topics.
- **Scalability:** It scales horizontally and can handle millions of messages per second.
- **Durability:** Kafka stores streams of records with fault-tolerance and supports data transformations.

WebSocket provides full-duplex communication channels over a single long-lived connection, which allows servers to push data to clients in real-time. Integrating WebSocket with Kafka enables the streaming of messages directly from Kafka to web clients without polling, reducing latency and overhead.

The typical architecture involves Kafka clusters serving as the central messaging hub, and a WebSocket server that subscribes to Kafka topics to fetch new messages. These messages are then pushed to connected web clients through WebSocket connections.



Above is the ASCII diagram for asynchronous data flow.

This setup allows for:

- **Real-Time Interaction:** Immediate data delivery enhances user experience in real-time applications.
- **Reduced Bandwidth and Overhead:** WebSocket eliminates the need for frequent polling, thereby reducing bandwidth usage and server load.
- **Scalability:** Both Kafka and WebSocket are highly scalable, supporting large numbers of users and high volumes of data.

The WebSocket Channel Specification document outlines the design, implementation, and operation of a WebSocket-based communication protocol tailored for real-time, bidirectional data exchange between clients and a server. This specification aims to provide a comprehensive framework that ensures seamless, efficient, and secure communication over WebSocket channels, addressing connection management, message formatting, event handling, and error management. By standardizing the WebSocket communication process, this dissertation serves as a foundational guide for developers and engineers to build and maintain robust real-time applications that leverage the full capabilities of the WebSocket protocol, fostering interoperability, and optimizing performance across diverse computing environments.

Cloud implementation

When exposing data through a channel it is important to use *Mapper* to compact the information through *DTO*. Those *DTO* will operate the Entity 4.11.



Figure 4.11: Data encapsulation strategy with business logic as a reference to the endpoints by [Buddy(2024)].

DTO objects contains an auto-serialization component that are securized without explicit action from the developer.

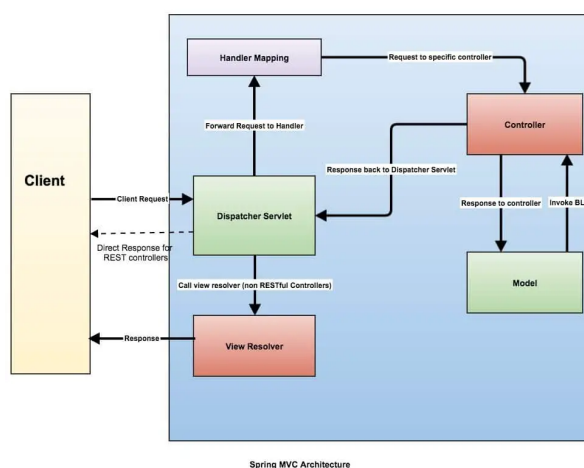


Figure 4.12: Reference architecture of *Spring Boot MVC* by [Medium(2024)].

However, the exception of the controller will be handled through advisor granting transactional integrity for the architecture shown in figure 4.12.

For a further knowledge about implemented channels there is a low level definition in appendix A.1.

Below are listed transactional objects (DTO) used by the I/O channels and their strict typing using Java programming language. DTO is used by the mapper and is the output of it as shown in figure 4.13.



Figure 4.13: Data encapsulation strategy with mapper as a reference to the endpoints by [Buddy(2024)].

For a full reference of DTO implementation, refer to appendix A.3.

MapStruct is a code generator that simplifies the implementation of mappings between Java bean types based on a convention-over-configuration approach. This tool uses annotations to configure mappings, thereby reducing the boilerplate code required to transfer data between object layers (e.g., from DTOs to entity classes and vice versa).

MapStruct offers several key features that make it a preferred choice for object mapping in Java:

- **Annotation-driven:** MapStruct uses annotations to define mappings, which makes the code easy to read and maintain.
- **Compile-time code generation:** It generates mapping code at compile time, ensuring high performance and early error detection.
- **Deep mapping:** MapStruct supports nested properties and complex data structures, allowing for deep copying of objects.
- **Type conversions:** Automatic conversions between types (e.g., from *String* to *Integer*) are provided, and custom type converters can be easily defined.
- **Immutable objects support:** MapStruct can work with immutable objects by calling appropriate constructor or builder patterns.

The MapStruct processor generates the implementation during the compilation phase. An example illustrating the basic usage is available in appendix A.13.

MapStruct allows for customizing mappings of complex types or specifying default values as shown in appendix A.13.

The integration of MapStruct in Java projects provides numerous benefits:

- **Simplicity and Readability:** Reduces the complexity of code for mapping objects, making it easier to understand and review.
- **Reduced Boilerplate:** Eliminates the need to manually write getters and setters for mapping.
- **Performance:** Since the mapping code is generated at compile time, the performance overhead at runtime is minimal compared to reflection-based libraries.
- **Type Safety:** Compile-time code generation also enhances type safety, reducing the risk of runtime errors.

4.2.4 Testing

Test-Driven Development (TDD) is a software development approach where tests are written before the actual code as fully stated by [Koskela(2007)]. This section delves into the practice of TDD using JUnit, one of the most popular testing frameworks in Java. TDD involves a short repetitive development cycle that adheres to the following steps:

The TDD cycle consists of three primary phases, often referred to as **Red**, **Green**, and **Refactor**:

1. **Red: Write a Failing Test**

The cycle begins with the development of tests that fail initially. These tests are designed based on the functionality that needs to be implemented. Writing a failing test first helps to clarify the requirements and ensures that the code will be testable.

2. **Green: Make the Test Pass**

The next step is to write the minimum amount of code necessary to make the test pass. This often results in quick and dirty solutions, which is acceptable at this stage, as the focus is to pass the test and verify the functionality.

3. **Refactor: Clean Up the Code**

Once the test passes, the code is refactored to adhere to clean code practices. This involves removing redundancies, optimizing structures, and applying design patterns where appropriate. Refactoring does not alter the behavior of the code; rather, it improves its internal structure and maintainability.

Here, we use JUnit to develop a simple Stack class following the TDD principles.

Step 1: Write a Failing Test (Red): This test checks if a new stack is empty. As the Stack class does not exist yet, this test will initially fail. This is visible in the implementation referred in the appendix A.12.1.

Step 2: Implement the Code to Pass the Test (Green): The Stack class is implemented with the minimal code to pass the test A.12.2.

Step 3: Refactor the Code (Refactor): The internal implementation of the Stack is refactored to use a more flexible data structure, improving the design without changing the behavior as shown in appendix A.12.3.

Extending the Test Suite New tests are added iteratively following the same TDD cycle.

Using JUnit for TDD offers several benefits:

- **Improved Code Quality:** Tests written in advance improve the design and maintainability of the code, such as shown in appendix A.12.
- **Regression Testing:** Each test suite acts as a regression suite to catch future breaks.
- **Documentation:** The test cases provide documentation that helps new developers understand the codebase.

4.3 Frontend implementation

The front-end contains all the applications that the client will use to communicate with the system in the easiest way. Proof of concept screenshots are available in appendix A.7.

Angular, a platform and framework for building client applications in HTML and TypeScript, is widely recognized for its robust architecture that makes it well-suited for enterprise-level applications. One of the key features of Angular is its built-in support for reactive programming with RxJS, which provides powerful tools for managing asynchronous data streams such as Subjects, BehaviorSubjects, and Subscriptions.

RxJS (Reactive Extensions for JavaScript) is a library for reactive programming using observables that makes it easier to compose asynchronous or callback-based code. This section focuses on three fundamental concepts of RxJS as used in Angular:

- **Subjects**
- **BehaviorSubjects**
- **Subscriptions**

A Subject in RxJS is a special type of Observable that allows values to be multicasted to many Observers. In Angular, Subjects are commonly used to broadcast values shared among multiple components.

In the example provided in appendix A.14 a Subject is used to push values to all subscribers whenever `updateValue` is called. This pattern is especially useful for event handling or data sharing across different parts of an application.

A BehaviorSubject holds the current value and emits it to any new subscribers immediately upon their subscription, making it ideal for representing "displayable" data in Angular applications.

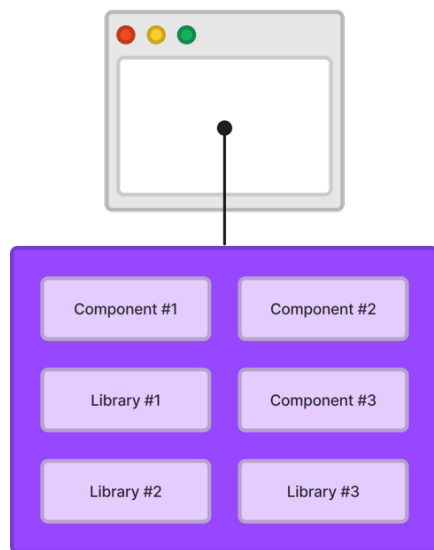
In the given example in A.14 `BehaviorSubject` is used to hold the current string data. The `data$` observable is used in the template with the `async` pipe to automatically subscribe and update the view whenever data changes.

Managing subscriptions is crucial to prevent memory leaks. The code snippet in A.14 demonstrates the best practice for handling subscriptions within Angular components, ensuring they are cleaned up during the component's destruction phase to avoid memory leaks.

Using Subjects, BehaviorSubjects, and managing Subscriptions in Angular allows to efficiently handle data streams and events. Modern web development has seen a shift towards more modular and scalable architectures, particularly with the rise of complex enterprise applications. Two significant concepts that have emerged to address these complexities are federated modules and micro frontends.

Federated Modules refer to a method of sharing code dynamically across various JavaScript applications, which Webpack 5 facilitates through its Module Federation feature. This approach allows for different frontend applications to load shared dependencies dynamically at runtime, rather than bundling them at build time as thesis stated by [Soares(2023)] and visualized in figure 4.14.

Monolith



Module Federation

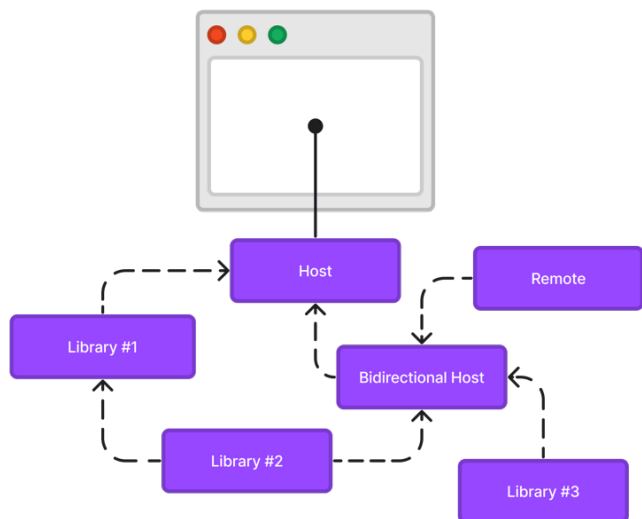


Figure 4.14: Federated modules schema, extracted from [smashingtips(2024)].

Webpack's Module Federation enables disparate builds to construct a cohesive application dynamically. It extends the build process to include dependencies from remote entries dynamically.

Federated Modules streamline the process of code sharing across teams and projects, enabling faster feature development and updates without requiring a full redeployment of the entire application.

Micro frontends extend the principles of microservices to frontend development. This approach breaks down frontend monoliths into smaller, more manageable pieces that can be developed, tested, and deployed independently.

A typical implementation involves setting up multiple frontend applications, each handling different aspects of the business domain, which are then integrated into a cohesive user experience.

- **Autonomy:** Teams can work independently on their respective areas without dependencies on other teams.
- **Scalability:** Each micro frontend can be scaled independently based on demand.
- **Flexibility:** Allows for using different frameworks and technologies suited to the specific needs of each frontend component.

Integrating federated modules with micro frontends provides numerous benefits:

- **Innovation:** Teams can experiment and implement new features without the risk of impacting other parts of the application.
- **Isolation:** Faults in one micro frontend do not affect the entire application, improving overall stability.
- **Maintainability:** Smaller codebases are easier to manage and update.

The combination of federated modules and micro frontends represents a paradigm shift in frontend development. By adopting these architectures enhancing the development workflows along improving deployment processes while delivering better user experiences with greater flexibility and efficiency.

4.3.1 Transactional layer

The Transactional Layer of the Battery Self-Management system is pivotal for ensuring robust and seamless data handling across various components of the system. This layer enhances the capabilities of the WebSocket protocol combined with the Simple Text Oriented Messaging Protocol (STOMP) to facilitate real-time communication between clients and servers as stated by [Ratamaa(2014)].

WebSocket provides a full-duplex communication channel over a single, long-lived connection, which enables continuous data flow without the overhead of repeatedly establishing connections. This is crucial for scenarios where real-time updates are essential, such as monitoring battery status and energy consumption in real-time, as inferred from [Fette and Melnikov(2011)].

- **Low Latency:** WebSocket minimizes the latency observed in traditional HTTP polling connections, providing faster data transmission that is vital for real-time applications.
- **Persistent Connection:** Unlike HTTP, WebSocket keeps the connection open, reducing the resources and time involved in the constant opening and closing of connections.
- **Bi-directional Communication:** Both client and server can send data actively without request-response constraints, enhancing the interactivity of the system.

STOMP, or Streaming Text Oriented Messaging Protocol, adds an additional layer of messaging semantics over the raw WebSocket, making it easier to handle message exchanges. It allows the client and server to communicate with message brokers via a common messaging API as we can infer from [Kumari et al.(2022)].

- **Message Orientation:** STOMP provides a frame-based messaging structure that simplifies the parsing and generation of messages.
- **Subscription-Based:** It supports a subscription model where clients can subscribe to specific topics or queues, which is particularly useful in managing data flow for different user interactions and system states.
- **Flexible Routing:** STOMP allows messages to be routed to various destinations based on the subscription, enabling a modular and scalable system architecture.

Combining WebSocket with STOMP provides a robust solution for handling real-time data in the Battery Self-Management system. This integration not only enhances the responsiveness of the system but also supports the efficient distribution of updates to various components and users, ensuring that all applications have the most current information at their disposal.

4.3.2 Native application

Electron is a framework for creating native applications with web technologies allowing the integration of robust web frameworks such as Angular to enhance the application's interface and functionality. The recompilation process for an Electron application with Angular integration involves monitoring and reflecting changes from the Angular development environment within the Electron app. By leveraging Electron's capabilities and additional development tools creating a seamless development workflow that automatically reflects changes in the Angular front end within the Electron application, enhancing productivity and efficiency in application development as inferred from the great book [Vuika(2019)] while having only one codebase makes the application easy to maintain.

Before diving into the recompilation process, it's essential to establish a development environment that supports both Electron and Angular. This typically involves creating an Angular project within the Electron application's structure and configuring Electron to load the Angular application as its front end as shown in figure 4.15.

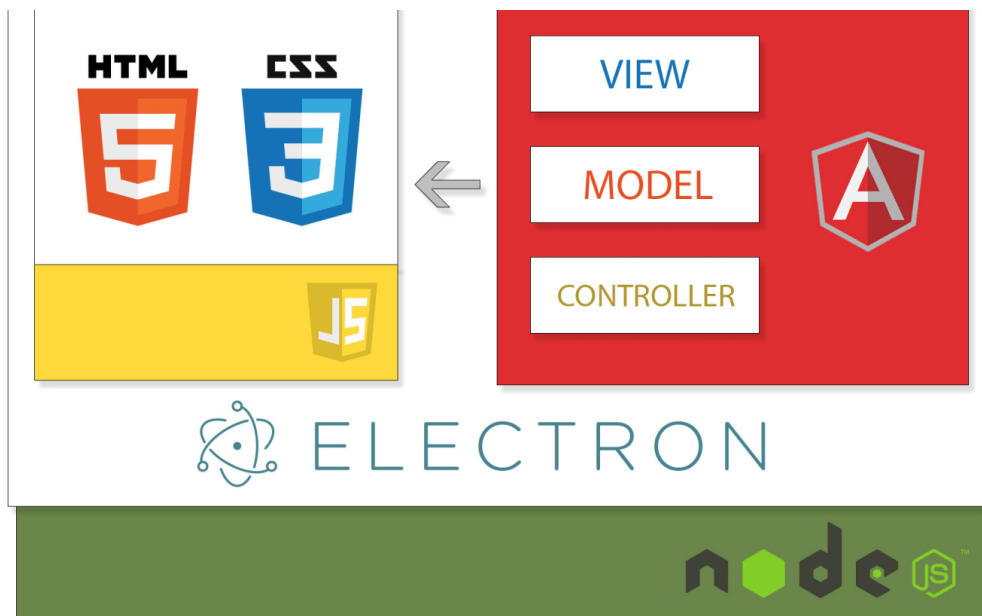


Figure 4.15: Electron recompilation for native angular apps by [DevCommunity(2022)].

4.3.3 Testing

This section outlines the testing strategies implemented in the Battery Self-Management system, focusing on the use of Jest, a delightful JavaScript Testing Framework with a focus on simplicity. It explains how Jest is used for writing both unit and integration tests, ensuring that all components function correctly individually and in combination.

Jest is a popular testing framework maintained by Facebook, designed to be comprehensive and configurable, allowing developers to create robust testing environments. Its zero-configuration setup, instant feedback, and built-in code coverage tools make it particularly suited for complex projects.

- **Zero Configuration:** Jest works out of the box with minimal setup, promoting rapid development cycles.
- **Snapshot Testing:** Ability to capture snapshots of React trees or other serializable values to simplify testing of complex and large structures.
- **Isolated and Fast:** Runs tests in parallel, ensuring that they are fast and flaky-test free.
- **Mocking:** Built-in extensive mocking, spying, and stubbing capabilities.

Unit tests are crucial for validating the individual components of the Battery Self-Management system. Jest provides an easy-to-use API for defining and running unit tests.

Integration tests verify that different modules of the system work together as expected. These tests are essential for ensuring that the integrated components of the Battery Self-Management system interact correctly.

Test checks that the user interface correctly interprets and displays the battery status from the data model as shown in appendix A.14. Note that most of the tests are not public since this could case security issues for the application.

Chapter 5

On-premise implementation

5.1 BMS and batteries

A Battery Management System (BMS)

is an electronic system that manages a rechargeable battery (cell or battery pack), ensuring its safe operation, extending its life, and maintaining its performance over time. The primary functions of a BMS include monitoring the battery's state (such as voltage, current, and temperature), calculating secondary data, reporting that data, controlling the environment, and balancing the battery by using wiries connected to B sockets as shown in figure 5.1. By protecting the battery from operating outside its safe operating area a BMS prevents battery failure optimizing battery performance and increasing the overall efficiency of the battery system.



Figure 5.1: JK BMS with unwired socket connections by [Nkon(2022)].

To wire an 18S LiFePO₄ battery pack, this project aims to connect 18 individual LiFePO₄ cells in series to achieve the desired voltage. Starting by ensuring all cells are at the same state of charge before running together. Connect the positive terminal of the first cell to the negative terminal of the second cell using a conductor with appropriate current-carrying capacity. Continuing this process by connecting the positive terminal of each preceding cell to the negative terminal of the next cell in the sequence, until all 18 cells are connected in series while the overall pack voltage is the sum of the voltages of all 18 cells. The final assembly should have the positive terminal of the first cell in the sequence serving as the pack's positive output, and the negative terminal of the last cell in the sequence serving as the pack's negative output.

Effective management of battery systems requires meticulous attention to temperature balancing and monitoring the degradation of internal resistance over time. Temperature balancing is crucial in preventing thermal runaway and ensuring that all cells within a battery pack operate within a similar temperature range. This unifor-

mity helps to maximize the efficiency and lifespan of the battery by avoiding over-charge and discharge conditions that can arise from temperature variations.

Similarly, the internal resistance of battery cells naturally increases over time due to chemical and physical changes within the cell. This degradation can lead to a decrease in the overall performance and efficiency of the battery. Monitoring this change is vital for predicting battery life expectancy and planning maintenance or replacement before failure occurs.

Furthermore, consistent resistance across all cells within a pack is essential for balanced charging and discharging. If unaddressed, cells with higher resistance may not charge fully or discharge faster than their counterparts, leading to imbalances that can significantly reduce battery effectiveness and safety. Therefore, implementing robust battery management systems that can actively monitor and compensate for these variations is indispensable for maintaining optimal performance and prolonging the operational life of battery systems.

Then... **Why is Balancing Necessary?** We know it thanks to [Daowd et al.(2011)] and below paragraph is fully extracted from that work.

Due to manufacturing variances, aging, and usage patterns, individual cells within a battery pack can exhibit differences in capacity, voltage, and internal resistance, leading to imbalances during charge and discharge cycles.

Imbalances can lead to underutilization of the battery's total capacity, reduced overall lifespan, and in severe cases, safety risks due to overcharging or deep discharging of individual cells.

Passive balancing, also known as bleed-off balancing, involves dissipating excess charge from higher-voltage cells through resistors, ensuring all cells reach the same voltage level during the charging process. While simple and cost-effective, this method is less efficient as it wastes the excess energy as heat.

Active balancing redistributes charge from higher-charged cells to lower-charged ones, either directly or via an energy storage and release mechanism. This method is more efficient than passive balancing, as it conserves energy within the battery pack, but it is also more complex and expensive.

Effective balancing requires sophisticated control systems capable of monitoring and adjusting the state of each cell in real-time. These systems typically include a battery management system (BMS) that ensures safe operation, optimizes performance, and extends the battery pack's lifespan. As shown in figure 5.2 wiring is done directly to cells as shown in diagram (a) and (b). Diagrams B and D are used to top-balance the cells before plugging in the BMS.

BMS alarms can be broadly categorized based on the nature of the detected issues. Key alarm types include:

- **Overcharge Alarm:** Triggered when a cell or the entire battery exceeds its maximum voltage threshold, indicating a risk of overheating or damage.
- **Over-Discharge Alarm:** Activated when a cell or the battery drops below its minimum voltage limit, potentially leading to reduced capacity and lifespan.
- **Over-Temperature Alarm:** Alerts when the temperature of the battery exceeds safe operational limits, posing a risk of thermal runaway.
- **Under-Temperature Alarm:** Indicates when the battery's temperature is too

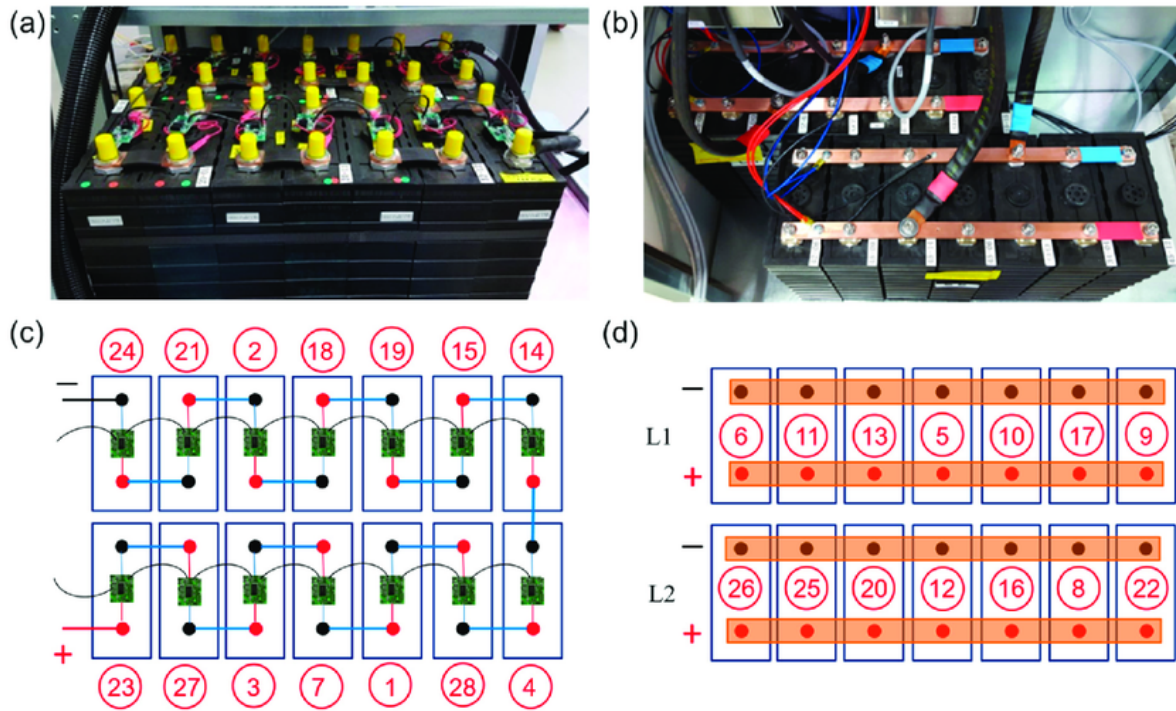


Figure 5.2: Engineering schematic diagram for BMS and LFP by [Yagci et al.(2022)].

low for safe charging, which could lead to plating of metallic lithium, a safety hazard.

- **Short Circuit Alarm:** Detects if a short circuit occurs, posing immediate risks of fire or explosion due to rapid heat generation.
- **Overcurrent Alarm:** Triggered by current flow exceeding safe operational levels during charging or discharging, potentially leading to overheating.

BMS alarms are critical for the proactive management of battery health and safety. They serve several important functions such as:

1. **Prevent Damage:** Alarms enable immediate corrective actions to avoid permanent damage to the battery cells.
2. **Enhance Safety:** By alerting to conditions like overcharging or high temperatures, alarms help prevent dangerous situations such as fires or explosions.
3. **Optimize Performance:** Maintaining operational parameters within safe limits ensures optimal battery performance and longevity.
4. **Support Maintenance:** Alarms can indicate when maintenance or inspection of the battery system is necessary.

5.1.1 BMS Configuration

This section provides a detailed overview of the configurations available in JK Battery Management Systems (BMS), including settings for cell count, current thresholds, voltage parameters, temperature monitoring, balancing configurations, communication settings, and power modes. Proper configuration of these parameters is crucial for the safety, efficiency, and longevity of battery packs managed by JK BMS.

At first, the cell count setting must match the number of cells in the battery pack to ensure accurate monitoring and balancing. As shown in figure 5.1.1, if there is not well configured the bms will not balance, allow to charge or allow to discharge due to security concerns. For a full screenshot reference of the below table 5.1.1 refer to appendix A.2. Having known, LifePO4 voltages are essential and covered at all in section 2.2.5.

The JK BMS configuration parameters are critical for ensuring the efficient and safe operation of battery packs. *Cell Count* refers to the number of cells used in the battery pack, affecting the overall voltage and capacity. *Battery Capacity (AH)* indicates the ampere-hours per cell, dictating the energy storage capacity. *Battery Trig. Volt. (V)* and *Start Balance Volt. (V)* set the thresholds for starting cell balancing based on voltage differences, essential for maintaining cell health and battery efficiency.

Calibrating Volt. (V) and *Calibrating Curr.(A)* are used for setting the current operational metrics of the battery, measured through sensors and calculations, respectively. *Max Balance Cur. (A)* specifies the maximum current allowed during the balancing process, ensuring safety and battery longevity.

Overvoltage and undervoltage protections are critical: *Cell OVP (V)* and *Cell UVP (V)* define the upper and lower voltage cut-off limits to prevent damage from excessive charge or discharge. The corresponding recovery settings, *Cell OVPR (V)* and *Cell UVPR (V)*, specify the voltages at which the battery can resume normal operation after a cut-off event.

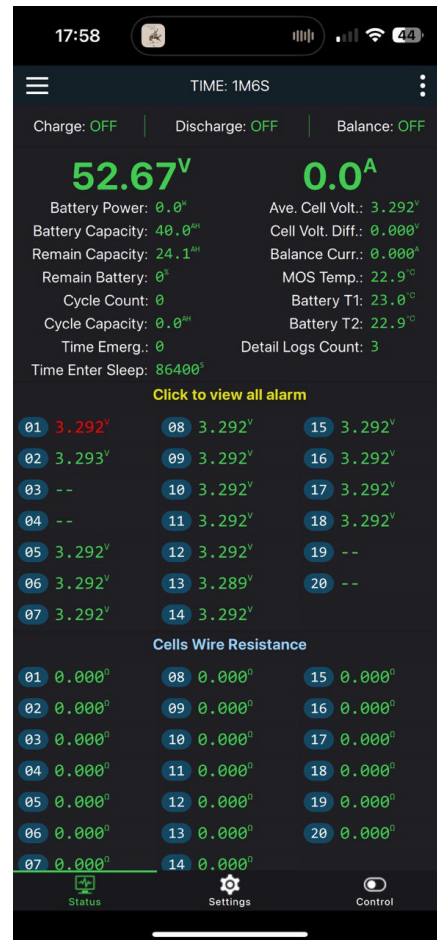


Figure 5.3: Cells 3 and 4 are not properly connected in the given JK Bms configuration. Image taken by Andrei García Cuadra on JK Bms App.

On-premise implementation

Power off volt. (V) is the critical voltage at which the battery will completely shut down, indicating a deep discharge state. Charge and discharge parameters, like *Continued charge curr. (A)* and *Continued Discharge Curr. (A)*, define the maximum current levels for charging and discharging processes. The system's responsiveness to potential faults is controlled by delay and recovery times such as *Charge OCP delay (S)* and *Discharge OCPR Time (S)*, which manage the durations before protective actions are taken.

Temperature thresholds for operation and safety are set by *Charge OTP (°C)* and *Discharge OTP (°C)*, marking the maximum allowable temperatures during charge and discharge to prevent overheating. Lower temperature limits are set by *Charge UTP (°C)*, ensuring the battery does not operate in conditions too cold for safe chemical reactions.

Additional safety features include MOSFET temperature limits (*MOS OTP (°C)*) and self-circuit protection delays (*SCP Delay (μ s)*), which help protect against short circuits and overcurrent situations. The device address (*Device ADDR*) configures network settings for communication protocols like ESP32 or RS485.

Lastly, user-specific settings and measurements like *User private data* and *Cons. Wire Res. 01 ($m \Omega$)* allow for customization and monitoring of internal resistance per cell, providing detailed insights into cell condition and overall battery health.

Configuration	Description	Value
Cell Count	Cell amount used in the battery pack.	18 units
Battery Capacity (AH)	Capacity in amperes per hour per each cell.	305 Amps per hour
Battery Trig. Volt. (V)	Cell difference voltage that enables balancing.	0.003 (volts)
Calibrating Volt. (V)	Current battery pack voltage.	Given by sensor
Calibrating Curr.(A)	Difference between max and min cell charge.	Given by calcs
Start Balance Volt. (V)	Cell difference voltage that enables balancing.	3.30 volts
Max Balance Cur. (A)	Max amperes when balancing.	2.0 Amps
Cell OVP (V)	Cell upper cut off voltage.	3.6 volts
Cell OVPR (V)	Cell max voltage.	3.55 volts
Cell UVPR (V)	Cell min voltage.	2.65 volts
Cell UVP (V)	Cell lower cut off voltage.	2.60 volts
Power off volt. (V)	Voltage where cell "dies".	2.50 volts
Continued charge curr. (A)	Max amperes when charging.	200 Amps
Charge OCP delay (S)	Delay if OVPR is reached when charging (unbalanced cells).	600 seconds
Charge OCPR Time(S)	Battery block if OVPR is reached in a cell repeatedly.	600 seconds
Continued Discharge Curr. (A)	Max amperes of the battery when discharging.	200 Amps
Discharge OCP Delay(S)	Delay if UVPR is reached when discharging (unbalanced cells).	300 seconds
Discharge OCPR Time (S)	Battery block if UVPR is reached in a cell repeatedly.	60 seconds
Charge OTP (°C)	Highest temperature on a cell when charging to shutdown battery.	70°C
Charge OTPR (°C)	Highest temperature on a cell when charging to raise alert.	60°C
Discharge OTP (°C)	Highest temperature on a cell when discharging to shutdown battery.	70°C
Discharge OTPR (°C)	Highest temperature on a cell when discharging to raise alert.	60°C
Charge UTP (°C)	Lowest temperature on a cell when discharging to shutdown battery.	-20°C
Charge UTPR (°C)	Lowest temperature on a cell when charging to raise alert.	-10°C
MOS OTP (°C)	MOSFET maximum temperature to shutdown battery.	100°C
MOS OTPR (°C)	MOSFET maximum temperature to raise alert.	80°C
SCP Delay (μ s)	Delay for self circuit protection on high rates when charging or discharging.	1500 μ s
SCPR Time(S)	Delay when high rates are raised in mosfet.	60 seconds
Device ADDR	ESP32 / RS485 device communication name.	Bx_Lx
User private data	Given when user connects trough serial.	null
Cons. Wire Res. 01 ($m \Omega$)	Cell resistance (one per each cell).	Automatic

Table 5.1: JK BMS configuration explanation. Property of the table of Andrei García Cuadra for this dissertation.

5.2 B-Box - Data extraction

The B box stands for black box. Its responsibilities are to communicate electrical system with the cloud. At first, we must set the data to be extracted by using the RS 485 protocol, also known as TIA/EIA-485 which is a standard defining the electrical characteristics of drivers and receivers for use in serial communications systems. It is an industrial specification designed for multi-point data transmission. RS485 supports long-distance and high-speed data transmission which is used for distributed battery systems that may share battery rooms across different buildings. The main features of RS485 include the following, extracted from [Goldie(1996)]:

- **Differential Signaling:** This method helps to increase immunity to electrical noise and allows for longer cable lengths.
- **Multi-point Configuration:** RS485 can connect up to 32 devices on a single bus without the need for repeaters, making it ideal for networked, on-site environments.
- **Speed and Distance:** It can transmit data at speeds up to 10 Mbps over distances up to 12 meters, and at lower speeds it can reach up to 1200 meters.

RS485 remains a reliable choice for industrial communication, offering flexibility, reliability, and cost-effectiveness. Its ability to function effectively over long distances and in electrically noisy environments ensures its continued relevance in modern industrial applications (just as JK bms are implemented in this case).

This data was extracted from JK Bms by reverse engineering when a USB RS-485 device was plugged into the RS-485 port of the BMS. Then, by extracting and interpreting all the codes, this is the resulting serial header data from JK BMS:

Hex Code	Description
0x00	When reading all data
0x0001	represents the low-volume alarm value, low-capacity alarm
0x01	Activation instruction, when the BMS is dormant, the controller will issue an activation command.
0x05	Pair code instruction to modify the parameters, needs the correct first command before changing
0x06	Read all the data in the ID code table at one time
0x4E	Start frame indicator, first byte of frame
0x57	Second byte of the start frame completes the sequence 0x4E 0x57
0x68	End of identity in the frame structure, indicates the end of the frame
0x79	Single battery voltage read command, returns 3 bytes per cell in mV
0x80	Read power tube temperature, values 0-140 (-40 to 100°C)
0x81	Read the temperature inside the battery box, similar range and encoding as 0x80
0x82	Read battery temperature, using the same format as 0x81
0x84	The current data, redefined scaling in descriptions

On-premise implementation

Hex Code	Description
0x8B	Battery warning message, contains various alarms and status bits.
0x8C	Battery status information, including MOS tube states and equalization switching
0x8E	Total voltage overvoltage protection settings, in mV
0x8F	Total voltage undervoltage protection settings, in mV
0x90	Monomer overvoltage protection voltage
0x91	Overvoltage recovery voltage of the monomer
0x92	Monomer overvoltage protection delay, in seconds
0x93	Unit undervoltage protection voltage
0x94	Monomer undervoltage recovery voltage
0x95	Monomer undervoltage protection delay, in seconds
0x96	Core differential pressure protection value, in mV
0x97	Discharge overcurrent protection value, in Amperes
0x98	Overcurrent delay of discharge, in seconds
0x99	Charge overcurrent protection value, in Amperes
0x9A	Charge overcurrent delay, in seconds
0x9B	Equalizing starting voltage, in mV
0x9C	Equalize the opening differential pressure, in mV
0x9D	Active Equalizing Switch, On or Off
0x9E	Power tube temperature protection value, in °C
0x9F	Power tube temperature recovery value, in °C
0xA0	Temperature protection value in the battery box, in °C
0xA1	Temperature recovery value in the battery box, in °C
0xA2	Temperature difference protection value of battery, in °C
0xA3	Battery charging high temperature protection value, in °C
0xA4	Battery discharge high temperature protection value, in °C
0xA5	Charge low temperature protection value, signed data for reference
0xA6	Charge low-temperature protection recovery value, signed data for reference
0xA7	discharge low temperature protection value, signed data for reference
0xA8	discharge cryogenic protection recovery value, signed data for reference
0xA9	Battery string count settings, range 3-32
0xAA	Battery capacity setting in ampere hours
0xAB	Charge MOS tube switch, on or off
0xAC	Discharge MOS tube switch, on or off
0xAD	Current calibration, in mA
0xAE	Guard plate address, reserved for cascading
0xAF	The battery type, encoded values for different chemistries
0xB0	Dormancy waiting time, in seconds
0xB1	Low capacity alarm value, in percentage
0xB2	Change parameter password, for security
0xB3	Special charger switch, on or off
0xB4	The device ID code

Hex Code	Description
0xB5	Manufacture date
0xB6	system working time, unit points
0xB7	Software version number
0xB8	Whether to start the current calibration
0xB9	Actual battery capacity, in ampere hours
0xBA	Name of Manufacturer ID
0xBB	Restart the system command
0xBC	Factory data reset command.
0xBD	Remote Upgrade Identification
0xBE	The cell turns off GPS with low voltage, in mV
0xBF	Cell low voltage recovery GPS, in mV

Table 5.2: JK BMS serial data headers. Table property of Andrei García Cuadra

The data extracted in table 5.2 is needed to properly set the instructions. This data will be processed and sent to the cloud. The Raspberry Pi collects data (such as sensor readings or system metrics) and sends this information to a Kafka topic as a producer. The data is then consumed by a server in the cloud, which processes, analyzes, or stores the data as required for each channel. In case there is an extreme case of below listed, a notification is generated and populated with highest priority to ensure safety:

- **Overvoltage:** Triggered when the voltage in any single cell or the entire battery exceeds the predefined thresholds.
- **Undervoltage:** Activated when the voltage drops below the minimum operational threshold, indicating potential battery discharge or cell imbalance.
- **High Temperature:** Occurs when the temperature of any cell or the overall system exceeds the safe operating limits.
- **Low Temperature:** Triggered by excessively low temperatures that could affect battery performance and chemistry.
- **Overcurrent:** This alarm is triggered when the current flowing into or out of the battery exceeds the set safety limits, potentially indicating short circuits or external faults.
- **Communication Errors:** Alerts triggered by failures in communication links within the BMS or with external controllers.
- **Sensor Faults:** Alarms due to failures or discrepancies in sensor readings that might affect the reliability of the data.

There is also a maximum of USB interface connections using a converter; the number of parallel connections supported remains dependent on the RS-485 specifications. Each USB to RS-485 converter can handle the same number of devices as a direct RS-485 connection would. This means a single USB port, when connected to an RS-485 network through a converter, can still manage up to 32 devices or more with appropriate hardware enhancements. Since our Raspberry PI Model has only one usb port that accepts serial connections, our limit is 32 batteries.

5.2.1 Enhanced Passive Balancing Using Machine Learning

Effective passive balancing in LiFePO₄ batteries is crucial for maintaining optimal battery health and efficiency. The integration of machine learning (ML) introduces predictive capabilities and automation, significantly enhancing this process.

Data collection is fundamental for training ML models. Key battery parameters to monitor include:

- Cell voltage levels
- Temperatures
- Charge and discharge rates
- Cycle counts

ML models leverage historical data to predict future cell behaviors, enabling preemptive adjustments in the balancing process. The following outlines the process and benefits of implementing ML for passive balancing.

The first step involves collecting relevant battery data. This can be achieved through various sensors and monitoring systems integrated with the battery management system (BMS). Data preprocessing includes normalization, handling missing values, and feature selection to prepare the data for training ML models.

Several ML algorithms can be applied to optimize passive balancing (some are being implemented on incoming sections):

- **Regression Models:** Used for predicting cell voltage levels and temperatures.
- **Classification Models:** For identifying potential cell failures or anomalies.
- **Reinforcement Learning:** To develop policies for real-time balancing based on current battery states.
- **Neural Networks:** For complex pattern recognition and predictive modeling.

Software Implementation: Integrating ML into the BMS involves several software components:

- **Data Acquisition Software:** Collects and stores battery parameters in a database. This is the 5.2
- **ML Training Frameworks:** TensorFlow, PyTorch, or Scikit-learn for developing and training models, as was covered in 4.1.1 using google cloud neural engine.
- **Real-time Data Processing:** Apache Kafka or MQTT for streaming real-time data to the ML models, along with CEP engines such as Flink for high volume processing.
- **Control Algorithms:** Embedded software in the BMS firmware for executing balancing actions based on ML predictions.
- **Feedback Loop:** Continuous data collection and retraining of models to adapt to changing battery conditions.

Optimization of Balancing Strategies: By evaluating real-time data, ML algorithms can determine the most effective strategies for passive balancing. This optimization

ensures that the timing and method of balancing are precisely tailored to the battery's current state.

Advanced Anomaly Detection: Machine learning enhances anomaly detection, identifying cells that deviate from typical behavior patterns. Early detection of anomalies can signal potential cell failures, allowing for proactive maintenance and preventing significant battery issues.

Automated Control Systems: An automated control system, driven by ML, can adjust passive balancing in real time. This dynamic response to changing conditions improves the overall efficiency and responsiveness of the battery management system.

Continuous Learning and Adaptation: Implementing a feedback loop allows the ML system to continuously learn and adapt. This iterative process enhances the accuracy of predictions and the effectiveness of balancing strategies over time, leading to ongoing improvements in battery management.

5.2.2 Active Balancing with artificial intelligence

This chapter discusses the implementation of active balancing in battery management systems (BMS) using artificial intelligence (AI) techniques.

Active balancing refers to the method of redistributing energy within a battery pack to ensure that all cells have equal charge levels, thus maximizing the capacity and lifespan of the battery. Unlike passive balancing, which merely dissipates excess energy as heat, active balancing transfers energy from higher-charged cells to lower-charged ones.

- **Cell Longevity:** Prevents cells from being overcharged or deeply discharged, conditions that can reduce the battery's operational lifespan.
- **Enhanced Efficiency:** Ensures that all cells in the battery pack contribute optimally, thus increasing the overall efficiency of the system.
- **Increased Safety:** Reduces the risk of battery failure due to imbalanced cells, which can lead to overheating and potential hazards.

AI techniques can significantly enhance the effectiveness of active balancing strategies by predicting future cell behaviors and optimizing energy distribution algorithms in real-time.

- **Machine Learning Models:** Predictive models can forecast cell voltages and SOC's under various load conditions.
- **Neural Networks:** Deep learning models can identify complex patterns in battery usage that affect cell imbalance.
- **Optimization Algorithms:** AI-driven algorithms can find optimal transfer strategies to efficiently balance cells while minimizing energy loss.

The implementation of active balancing with AI involves several key components and steps:

1. Collect real-time data from each cell in the battery pack, including voltage, current, temperature, and cycle count that is already managed by Kafka topic */batteries/individual_data*
2. Develop artificial intelligence to predict the future state of each battery cell based on historical and real-time data, such as:

```
from sklearn.ensemble import RandomForestRegressor

# Train a random forest model to predict cell voltage
model = RandomForestRegressor(n_estimators=100)
model.fit(X_train, y_train)
```

3. Create an AI-based control algorithm that uses model predictions to determine the optimal amount of energy to transfer between cells with:

```
def balance_cells(battery_pack):  
    predicted_soc = model.predict(battery_pack.get_data())  
    transfer_amounts = optimization_algorithm(predicted_soc)  
    battery_pack.transfer_energy(transfer_amounts)
```

Implement a real-time monitoring system using a Kafka topic to continuously track the state of the battery and adjust the balancing actions as necessary.

Thoroughly test the AI models and the balancing algorithm in controlled environments and real-world scenarios to validate their effectiveness and safety.

- **Balancing Efficiency:** Measure the time and energy required to achieve balance across the battery pack.
- **Battery Lifespan:** Monitor the impact of active balancing on the overall lifespan of the battery.
- **System Stability:** Assess the stability and robustness of the balancing system under various operating conditions.

5.2.3 Battery degradation prediction with artificial intelligence

Battery degradation prediction is crucial for proactive maintenance and management of battery packs, ensuring they operate efficiently and safely throughout their useful life. By accurately predicting how batteries degrade under different conditions, operators can make informed decisions, such as:

- **Enhanced Reliability:** Predictive insights help in anticipating failures and scheduling maintenance to avoid unexpected downtimes.
- **Cost Efficiency:** Optimizes the lifecycle management of batteries, potentially reducing the costs associated with premature replacements.
- **Safety Improvements:** Early detection of potential degradation can prevent safety issues related to battery malfunctions.

Implementing AI involves selecting appropriate machine learning models that can handle the complexity and variability of battery performance data.

- **Regression Models:** To predict continuous values representing the state of health (SoH) and remaining useful life (RUL) of the battery.
- **Time Series Forecasting:** Utilizing models like ARIMA, LSTM networks for predicting future degradation based on past data.
- **Survival Analysis:** To estimate the time until a battery reaches a critical degradation threshold.

The implementation of an AI-driven battery degradation prediction system consists of several steps:

1. Collect comprehensive data on battery usage, including charge cycles, load profiles, temperature, voltage, current, and any other relevant telemetry data.

```
import pandas as pd
from sklearn.preprocessing import StandardScaler

# Load and scale data
data = pd.read_csv('battery_data.csv')
scaler = StandardScaler()
scaled_data =
scaler.fit_transform(data[['voltage', 'current', 'temperature']])
```

2. Then, extract and select features that significantly impact battery life, such as depth of discharge (DoD), charge rate, and environmental factors.

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor

X_train, X_test, y_train, y_test =
train_test_split(scaled_data, labels, test_size=0.2)
model = RandomForestRegressor(n_estimators=100)
model.fit(X_train, y_train)
```

3. Deploy the trained model into a production environment where it can provide real-time weather energy production predictions. Continuously monitor model performance and update it as more data becomes available or as battery conditions change, saving historical of modifications.

Conduct thorough testing to validate the accuracy of the AI model. Use metrics like Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) to assess performance.

- **Accuracy:** Measure how close the predictions are to actual degradation outcomes.
- **Precision:** Evaluate the model's ability to predict degradation without triggering false alarms.

At the moment this dissertation is being written, accuracy is around 70%, so data is being subject to quality assessments to guarantee that at least 95% of accuracy is reached.

5.2.4 Battery lifespan prediction with artificial intelligence

This chapter focuses on the application of artificial intelligence (AI) to predict the lifespan of batteries, which is critical for optimizing the use.

Predicting the lifespan of batteries involves estimating how long a battery will continue to meet its performance standards before it degrades beyond usability. Accurate lifespan predictions help in planning for replacements.

- **Operational Efficiency:** Ensures that battery-dependent systems operate without unexpected failures, improving uptime and reducing maintenance costs.
- **Cost Management:** Helps in budgeting for battery replacements and in making cost-effective decisions regarding battery refurbishment or recycling.
- **Sustainability:** Supports the environmental aspect by optimizing battery usage and reducing waste.

Utilizing AI and statistical modeling, AI can analyze historical and operational data to predict when a battery will likely fail or fall below certain performance thresholds, some of the examples are:

- **Regression Analysis:** Models like linear regression, support vector regression (SVR), or ensemble methods (e.g., Random Forests) to predict continuous lifespan values based on input features.
- **Deep Learning:** Neural networks, particularly Long Short-Term Memory (LSTM) models, are effective in handling sequences such as time-series data from battery usage cycles.
- **Survival Analysis:** Particularly useful for censored data where the exact time of failure might not be known but can still provide probabilistic estimates of lifespan.

The implementation of an AI-based system for battery lifespan prediction involves several structured steps from data handling to deployment and monitoring.

Gather detailed data from battery operation, including discharge cycles, load characteristics, ambient conditions, and maintenance history.

```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler

# Load data
data = pd.read_csv('battery_usage.csv')
scaler = MinMaxScaler()
features_scaled =
scaler.fit_transform(data[['cycle_count', 'load', 'ambient_temp']])
```

Develop features that correlate strongly with battery health and lifespan, such as the number of charge cycles, average discharge depth, temperature variations, etc.

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import GradientBoostingRegressor

X_train, X_test, y_train, y_test =
train_test_split(features_scaled, lifespan, test_size=0.3)
model = GradientBoostingRegressor(n_estimators=200)
model.fit(X_train, y_train)

# Evaluate model
predictions = model.predict(X_test)
```

Deploy the model into the Google Cloud production environment where it can assess battery data in real-time and provide ongoing lifespan predictions.

Extensive with Kafka validation is necessary to ensure the model's predictions are reliable and accurate across various operating conditions and battery types.

- **Mean Absolute Error (MAE):** Measures the average magnitude of errors in a set of predictions, without considering their direction.
- **Root Mean Squared Error (RMSE):** Provides a measure of the magnitude of error, giving higher weight to larger errors.

Accurately forecasting the end of a battery's useful life plan for maintenance, and make informed financial decisions while extending the operational life of their systems.

5.2.5 Battery predictive maintenance with artificial intelligence

This chapter explores the application of artificial intelligence (AI) in implementing predictive maintenance strategies for battery systems. Predictive maintenance is crucial for maximizing battery life and efficiency, reducing downtime, and minimizing maintenance costs by predicting failures before they occur.

Predictive maintenance for batteries involves monitoring their condition and performance to predict when maintenance should be performed. This approach contrasts with preventive maintenance, which is scheduled at predetermined intervals regardless of the actual condition of the battery.

- **Cost Efficiency:** Reduces unnecessary maintenance, thereby lowering operational costs.
- **Downtime Reduction:** Prevents unexpected battery failures, reducing system downtime and enhancing reliability.
- **Lifecycle Extension:** Proper maintenance can significantly extend the usable life of batteries.

The implementation of AI in predictive maintenance involves several statistical and machine learning techniques that analyze battery data to forecast future performance and identify potential failures with:

- **Classification Models:** Used to determine the likelihood of a failure occurring within a specific future time frame based on current and historical data.
- **Anomaly Detection:** Identifies patterns in battery data that deviate from normal operational behavior, which can be indicative of potential issues.
- **Regression Models:** Predicts the remaining useful life (RUL) of a battery by modeling the degradation path over time.

The predictive maintenance system is implemented through several key phases from data collection to deployment.

Continuous monitoring and collection of data are critical. Data points typically include charge and discharge cycles, voltage, current, temperature, and impedance.

```
import pandas as pd
from sklearn.preprocessing import StandardScaler

# Load and clean data
data = pd.read_csv('battery_data.csv')
data = data.dropna() # Remove missing values

# Feature scaling
scaler = StandardScaler()
scaled_features =
scaler.fit_transform(data[['voltage', 'current', 'temperature']])
```

Features that significantly impact battery health, such as depth of discharge and charge rates, are engineered and selected for model training.

```
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC

X_train, X_test, y_train, y_test =
train_test_split(scaled_features, labels, test_size=0.25)
model = SVC(kernel='linear')
model.fit(X_train, y_train)

# Model evaluation
from sklearn.metrics import classification_report
predictions = model.predict(X_test)
print(classification_report(y_test, predictions))
```

Deploy the AI model to Google Cloud through Jenkins to continuously assess the health of the battery and predict maintenance needs. Implement monitoring tools to evaluate the model's performance and update it as necessary based on Grafana monitoring.

The model is rigorously tested using historical data to validate its accuracy and reliability. Adjustments are made based on performance metrics such as accuracy, precision, and recall.

The use of AI in predictive maintenance allows for significant improvements in the management and operation of battery systems. By predicting maintenance needs before failures occur, battery systems can operate more reliably and efficiently subsequently with reduced costs and extended service lives.

5.2.6 Full state of charge prediction with artificial intelligence

This chapter addresses the implementation of artificial intelligence (AI) techniques in conjunction with the Coulomb counting method to accurately predict the full state of charge (SoC) of battery systems. This approach enhances the precision and reliability of SoC estimations which is crucial for optimal battery management and operation.

State of charge (SoC) is a critical parameter in battery management systems, indicating the current charge level of a battery relative to its capacity. Important areas are:

- **Enhanced Battery Efficiency:** Precise SoC prediction helps in optimizing the charging cycles and energy usage of the battery.
- **Improved Safety:** Reduces the risk of battery failures by preventing conditions that may lead to overheating or damage.
- **Better Energy Management:** Enables more reliable predictions of battery life and power availability for planning and operational purposes, such as next days state of charge (stated in section 5.2.6).

Coulomb counting is a fundamental method for estimating the SoC by integrating the current flowing in and out of the battery over time. It provides a direct measurement of charge movements, which is highly reliable under controlled conditions.

```
def calculate_soc(currents, time_intervals, initial_soc):  
    charge = initial_soc  
    for i in range(len(currents)):  
        charge += currents[i] * time_intervals[i] / 3600  
    return charge
```

While Coulomb counting provides a solid foundation for SoC estimation, integrating AI can compensate for its limitations such as errors due to battery aging and temperature effects enhancing the accuracy and robustness of predictions.

- **Data Collection:** Gather extensive data on battery usage, including current, voltage, temperature, and historical SoC values.
- **Feature Engineering:** Develop features that capture the health and operational patterns of the battery, which affect SoC estimation accuracy.

Using regression models or neural networks to refine SoC estimates derived from Coulomb counting based on observed patterns and anomalies in the data.

```
from sklearn.ensemble import RandomForestRegressor  
  
# Assume features and labels are prepared  
model = RandomForestRegressor(n_estimators=100)  
model.fit(train_features, train_labels)  
  
# Predict SoC  
predicted_soc = model.predict(test_features)
```

Deploy the AI-enhanced Coulomb counting model in a real-time system to continuously update SoC predictions based on live data. Then, next steps are:

1. Integrate the model into the battery management system, ensuring it receives real-time data feeds and provides ongoing SoC estimates to the control system.
2. Conduct comprehensive testing to validate the accuracy and reliability of the SoC predictions, using metrics such as mean absolute error (MAE) and root mean squared error (RMSE).
3. Integrating artificial intelligence with the Coulomb counting method for predicting the state of charge in batteries significantly improves the accuracy, reliability, and operational efficiency of battery management systems. This hybrid approach enables proactive energy management and maintenance strategies that enhance the overall performance and longevity of battery systems, as we will cover in below sections starting from section 5.2.7.

5.2.7 Next days charge prediction with artificial intelligence

The system integrates solar azimuth angle data and meteorological forecasts to optimize charging strategies in environments relying on solar power.

Accurate predictions of daily charge this are vital for maximizing the efficiency of battery systems, especially in contexts where solar energy is a primary power source. These predictions help in planning the energy input needed to fully charge or maintain batteries, thus ensuring consistent power supply and effective energy usage.

- **Solar Power Maximization:** By predicting solar power availability through meteorological and azimuth data, the system can optimize battery charging times to coincide with peak sunlight hours.
- **Operational Efficiency:** Effective prediction helps in avoiding overcharging and deep discharging, conditions that can degrade battery health over time.
- **Energy Management Optimization:** Allows for better grid interaction and energy consumption planning, especially in grid-tied systems.

The prediction system is built on a microservices architecture, which allows for scalable, flexible, and independent deployment of services. Each component of the system — data collection, processing, model execution, and output — is encapsulated into separate microservices.

- **Data Collection Service:** Collects real-time meteorological data and solar azimuth angles. For this, the open api from weather.io was used.
- **Data Processing Service:** Cleanses and transforms data for predictive modeling. For production environment, this is being run in Google Cloud.
- **AI Prediction Service:** Hosts the machine learning models and generates charge predictions.
- **Client Communication Service:** Provides APIs for accessing prediction results and submitting new data points, using Kafka, MQTT and Websocket (the usage of which technology is mainly based on the layer of communication).

The AI models use historical data and forecasts to predict the optimal charge levels for the next day. Techniques such as time series forecasting and regression analysis are employed.

```
# Python pseudo-code for data integration and feature engineering
import pandas as pd
data = pd.merge(meteorological_data, azimuth_data, on='date')
data['solar_potential'] =
data['sunlight_hours'] * data['solar_azimuth']
```

- **Continuous Integration/Continuous Deployment (CI/CD):** Automated pipelines update services with minimal downtime.
- **Monitoring Tools:** Use of Prometheus and Grafana for monitoring the health and performance of services.

Below is the code that runs the base case:

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import GradientBoostingRegressor

# Prepare features and labels
X = data[['temperature', 'cloud_coverage', 'solar_potential']]
y = data['charge_requirement']

# Split data
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.2)

# Train model
model = GradientBoostingRegressor(n_estimators=100)
model.fit(X_train, y_train)

# Evaluate model
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(y_test, model.predict(X_test))
print(f"Model MSE: {mse}")
```

The microservices are containerized and deployed using Docker and orchestrated with Kubernetes. This setup supports auto-scaling based on demand and simplifies the deployment of updates.

Meteorological predictions and solar azimuth data within a microservices architecture the next day's charge prediction system significantly enhances the operational efficiency and sustainability of battery-dependent solar power systems. The architecture not only supports scalability and ease of maintenance but also facilitates the integration of additional data sources and predictive models to further refine the system's accuracy. So the goal for this was achieved by obtaining most demanded frames and interconnecting devices to seamlessly start or stop. Per example, hot water outlets now only run on most producing hours (day) and not on most-demanding hours (night) by overheating the water 20 degrees above recommended temperature (50 degrees) so that the consumption on low-producing hours is near zero.

5.2.8 Expected consumptions with artificial intelligence

The goal of this system is to enhance energy distribution and efficiency. The main points include:

- **Energy Efficiency:** Precise predictions assist in identifying potential savings by optimizing energy usage patterns and minimizing waste.
- **Cost Reduction:** Demand forecasting aids in better tariff and load management, thereby lowering operational and energy expenses.
- **Load Balancing:** Improves the stability of power grids by anticipating peak demand periods and adjusting supply accordingly.

Machine learning equips us with the tools to analyze extensive datasets and discover patterns that can forecast future consumption. The data utilized will include:

- **Historical Consumption Data:** Gather information on previous energy usage across various times and conditions.
- **Environmental Data:** Incorporate temperature, humidity, and other pertinent weather conditions that significantly influence energy consumption.
- **User Behavior Data:** Information on usage patterns at different times of the day or in reaction to specific conditions.

Since at the moment the system has been running for few months there is still no data to analyze, but below is the study of the showcases about the future implementation. Below there are the **algorithms** that will be considered for the **implementation** along with some short code pseudo-codes:

- **Regression Models:** Linear regression, support vector regression, or ensemble methods like random forests or gradient boosting for continuous consumption values.
- **Time Series Models:** ARIMA, Seasonal ARIMA, or LSTM networks for handling data with time-dependent patterns.

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import GradientBoostingRegressor

X = data[['hour_of_day', 'temperature', 'is_weekend']]
y = data['consumption']

X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.3)
model = GradientBoostingRegressor(n_estimators=200)
model.fit(X_train, y_train)

# Evaluate model
from sklearn.metrics import mean_absolute_error
mae = mean_absolute_error(y_test, model.predict(X_test))
print(f"Model MAE: {mae}")
```

Implementing the predictive models in a production setting will need the following actions as it is under development right now as this dissertation is being written:

- **Performance Monitoring:** Continuously assess the model's performance and modify parameters or retrain with updated data as needed.
- **Operational Integration:** Guarantee that the model's outputs are seamlessly incorporated into operational planning and decision-making workflows.

5.3 IoT

The Internet of Things (IoT) significantly augments the management of battery systems by providing advanced connectivity and intelligence capabilities. IoT enables real-time data collection from various sensors embedded within battery systems, which facilitates:

- Continuous monitoring of battery status, including charge levels, temperature, and health status, ensuring optimal operation and preventing failures.
- Predictive maintenance by analyzing data trends to forecast potential issues before they occur, thereby reducing downtime and extending battery life.
- Enhanced decision-making through the aggregation of data across multiple systems, allowing for better energy management and efficiency.

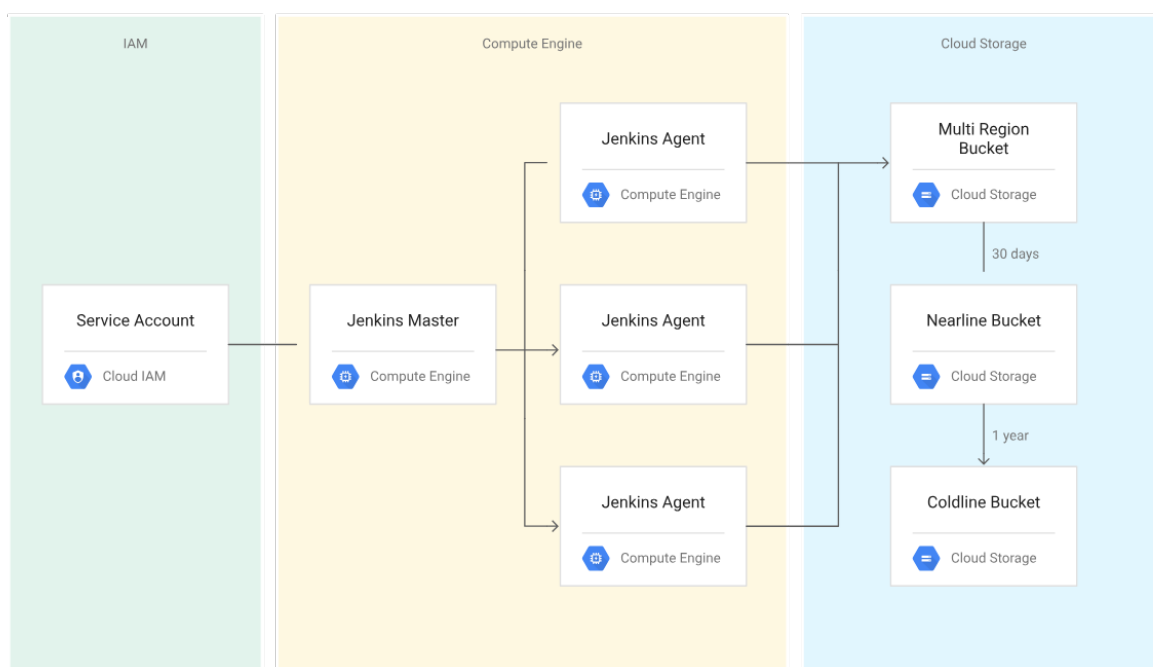


Figure 5.4: System diagram for deploying IA along with IoT data to Google Cloud. Image property of [Google(2024b)].

To optimize the energy usage and identify potential leakages or excesses, conducting energy research to establish preference consumption ranges is essential as shown in figure 5.4. This involves:

1. **Data Collection:** Gather detailed consumption data from the IoT-enabled devices over a significant period to establish baseline energy usage patterns.
2. **Analysis:** Utilize machine learning algorithms to analyze the consumption data to identify typical and atypical energy usage patterns.
3. **Optimization:** Based on the analysis, develop strategies to optimize energy consumption, reduce costs, and enhance battery efficiency.

4. Leakage Detection: Implement anomaly detection techniques to identify and alert any deviations from normal patterns that might indicate leakages or faults.

Implementing this research involves several stages:

- **Setup of Monitoring Tools:** Deploy IoT sensors and smart meters to continuously monitor energy usage across various nodes of the system. Done previously under the devops infrastructure.
- **Data Aggregation and Storage:** Use cloud-based platforms to store and manage the vast amounts of data generated by IoT devices. Done already by handling this with the cloud backend.
- **Analytical Processing:** Apply advanced data analytics and machine learning techniques to process and analyze the data, identifying potential savings and inefficiencies. Done this with the artificial intelligence pods.
- **Reporting and Action:** Generate detailed reports on energy consumption patterns and implement automated systems for real-time adjustments based on the insights gained. Done this with devops monitoring tools such as Grafana.

This structured approach not only enhances the efficiency of battery usage but also aligns with sustainability goals by reducing unnecessary energy consumption and improving the overall management of battery-operated devices.

5.3.1 MQTT

MQTT (Message Queuing Telemetry Transport) is a lightweight messaging protocol designed for constrained devices and low-bandwidth, high-latency, or unreliable networks. In the context of IoT-based battery management, MQTT offers a robust mechanism for device-to-device or device-to-server communication. Here's how MQTT was integrated into the project to improve data transfer efficiency and reliability:

As shown below in figure 5.5, messages are realtime sniffed and shared efficiently using MQTT channels. Taking note that just showing few of them due to privacy concerns.



Figure 5.5: MQTT messages (only few of) that are being transacted on the system every second.

- **Real-time Data Transmission:** MQTT facilitates the real-time transmission of data from IoT devices to the central server or between devices. This is critical for battery management systems where timely data is crucial for operational decisions and predictive maintenance.
- **Reduced Bandwidth Consumption:** Due to its lightweight nature, MQTT minimizes the bandwidth required for transmitting messages. This is essential in IoT environments where multiple devices continuously send data.
- **Persistent Sessions and Message Retention:** MQTT supports persistent sessions, which reduce the need to re-establish connections. This feature, combined with retained messages that store the last relevant message on a topic, ensures that device commands or critical alerts are not lost even if the device temporarily disconnects.
- **Quality of Service (QoS) Levels:** MQTT offers multiple levels of QoS for message delivery:
 1. QoS 0 (At most once) — where messages are delivered according to the best effort of the underlying network.
 2. QoS 1 (At least once) — guarantees that messages are delivered at least

once to the receiver.

3. **QoS 2 (Exactly once)** — ensures that each message is received only once by the counterpart.

This feature is particularly useful in managing energy systems where different types of data might require varied levels of delivery assurance.

- **Subscription Model:** MQTT operates on a publish/subscribe model, which efficiently routes messages to multiple clients that subscribe to a particular topic. This model is ideal for scenarios where battery status updates or energy usage alerts need to be distributed across various operational modules.

Implementing MQTT in battery management systems enhances the overall responsiveness and efficiency of the IoT framework. By ensuring reliable and efficient communication between devices and servers, MQTT plays a pivotal role in the real-time management and monitoring of battery-operated devices within the IoT ecosystem.

Chapter 6

Conclusions

The integration of *Artificial Intelligence* (AI) in battery self-management within a *real-time Internet of Things* (IoT) architecture represents a significant leap forward in optimizing energy storage systems. This advanced approach harnesses the potential of AI to analyze large amounts of data, learn from it, and make **predictive decisions**, thereby improving the **efficiency**, **reliability**, and **useful life** of battery systems. The conclusions drawn from exploring such an integration span several key insights and future directions.

First, the implementation of **AI algorithms** for battery management on IoT devices facilitates *real-time monitoring* and *predictive maintenance* then the system can predict battery failures before they occur, schedule maintenance, and optimize charging cycles according to usage patterns and environmental conditions. This not only extends battery life but also ensures the reliability of IoT devices, which is crucial for critical applications.

Moreover it enables *dynamic energy optimization* in scenarios where IoT devices are part of a larger network, such as smart grids or electric vehicle fleets, real-time decisions about energy allocation, storage, and consumption. This dynamic approach optimizes energy usage throughout the network, reduces operational costs, and minimizes environmental impact by promoting the efficient use of renewable energy sources.

Another significant advantage is the enhancement of safety features along to detect and mitigate risks associated with battery operation, such as overheating or overcharging, in real time. This proactive approach to safety management not only protects the battery and the device but also protects the user and the environment from potential hazards.

The integration of AI into battery self-management also opens new avenues for **sustainability**. By optimizing battery usage and extending battery life, the demand for battery production will be reduced. This contributes to sustainability efforts by minimizing the environmental footprint associated with battery manufacturing and disposal. Within an architecture of IoT highlights the importance of *data privacy* and *security*. As these systems rely on the continuous collection and analysis of data, ensuring the privacy and security of these data becomes paramount. Future developments must prioritize robust security protocols to protect sensitive information and maintain user trust.

Looking ahead, the integration of AI into battery self-management within IoT devices presents fertile ground for innovation. As technologies evolve and IoT devices become more ubiquitous, the potential of AI to revolutionize battery management and energy systems is immense. However, this also requires continued research on AI algorithms, battery technology, and IoT security to overcome challenges and realize the full potential of this integration.

In conclusion, the application of *Artificial Intelligence* in battery self-management within a *real-time* IoT architecture offers transformative benefits, including enhanced **efficiency**, **reliability**, **safety**, and **sustainability**. As we move forward, the continued convergence technologies will play a pivotal role in shaping the future of energy systems and smart devices proning them more intelligent, autonomous and environmentally friendly.

The value of this dissertation will be covered in the next section 6.1. The most difficult part of the dissertation was to properly extract the data from the JK BMS, which is a professional system with no updated documentation all over the network or support in which doing reverse engineering was key to ensure the success of the project. Also, there were many race conditions when communicating the systems since real-time is complex to handle in terms of data integrity and this is why kafka and stomp brokers played a crucial role to play with multi-pod channels. There was a mistake to start the implementation with OpenApi (Section 4.1.8) since this is delved to work in synchronous systems, so this was discarded during the implementation.

6.1 Impact analysis

The integration of Artificial Intelligence (AI) into the management of battery systems within IoT frameworks significantly optimizes energy utilization, improves battery life, and ensures the seamless operation of devices IoT. This impact analysis delves into the various dimensions through which AI-driven battery self-management revolutionizes energy systems in IoT environments. Below are the objectives that have been achieved with the dissertation that were covered in section 1.2. Here we will cover based on the final result obtained, while the result could be combined on several initial objectives such as Cost Efficiency, as belows:

Cost Efficiency Implementation of AI in battery management systems substantially reduces operational and maintenance costs by optimizing charging cycles and predicting potential system failures before they occur.

Market Opportunities The development of battery self-management enabled by AI opens up new market opportunities for smart IoT applications in both the commercial and residential sectors, potentially leading to increased market share and revenue streams for technology providers.

Enhanced User Experience AI-driven systems offer a user-friendly interface for real-time monitoring and control, significantly improving user engagement and satisfaction.

Public Safety and Reliability By enhancing the reliability and safety of battery-operated devices, AI contributes to public safety, especially in critical applications such as medical devices and emergency response systems.

Sustainability Artificial intelligence optimizations lead to better energy management and reduced environmental footprint by prolonging the lifespan of batteries and decreasing the frequency of their replacement.

Reduction of E-Waste Predictive maintenance and optimized battery usage contribute to reducing electronic waste, aligning with global efforts towards environmental sustainability.

Innovation and Advancements The project drives innovation in AI and IoT technologies, setting a benchmark for future developments in intelligent energy solutions.

Challenges and Future Directions Despite its benefits, integrating AI into battery management involves challenges such as data security and privacy, which must be addressed to fully realize the potential of these technologies.

Integration of AI into battery management systems based on IoT not only improves energy efficiency and device longevity, but also offers significant economic, social, and environmental benefits, paving the way for smarter and more sustainable future technologies.

So the most relevance of the dissertation is related the communications between the systems and applying different fields such as IoT, AI and FPGA to a field in which this was never combined, guaranteeing an efficient implementation of domotic and industrial processes.

Glossary

advisor Class that advise or modify the behavior of API requests and responses, without changing the business logic contained in the controllers or services. 71

AI Artificial Intelligence (AI) refers to the branch of computer science focused on creating systems that can perform tasks that would normally require human intelligence. These tasks include reasoning, learning, problem-solving, perception, language understanding, and decision making. 1, 4, 7, 8, 34, 93, 107–109

Angular Angular is a popular open-source web application framework maintained by Google and a community of developers. It's used for building dynamic, single-page web applications (SPAs) and supports the development of large-scale, high-performance applications. 77

autodiscover It operates by providing clients (connections) with the necessary configuration information, including server settings, parameters, and other battery configurations, without requiring manual input from users. 145

BMS A BMS (Battery Management System) is an electronic system that manages and monitors the performance, safety, and lifespan of rechargeable batteries. It is commonly used in various applications, including electric vehicles (EVs), renewable energy storage systems, consumer electronics, and industrial equipment. 4, 25, 32, 51

controller A controller is a component that manages the flow of data between the server and the client. It acts as an intermediary, receiving requests from clients, processing them (often with the help of models and services), and then sending responses back. 71

DTO Encapsulated data through a class that reduces information from a database entity. 71, 72, 112

Electron Open-source framework developed and maintained by GitHub. It allows developers to build cross-platform desktop applications using web technologies such as HTML, CSS, and JavaScript which is the output of an Angular application compiled. 77

Entity Formal representation of a table of a database through a programming class to ensure proper codification. 71

ESP32 The ESP32 is a low-cost, low-power system on a chip (SoC) microcontroller with integrated Wi-Fi and dual-mode Bluetooth capabilities. 8

exception Refers to an error condition or unexpected event that disrupts the normal flow of execution within the API. 71

FPGA An FPGA, or Field-Programmable Gate Array, is a type of integrated circuit (IC) that can be programmed and reprogrammed to perform a wide variety of logic functions after it has been manufactured. 109

I/O Refers to Input Output, which englobes the transaction of the incoming data and the generated data based on it. 71

Inverter An electrical inverter is an electronic device that converts direct current (DC) into alternating current (AC). The primary function of an inverter is to transform the DC electricity, typically from sources like batteries, solar panels, or fuel cells, into AC electricity, which is the standard form of electrical power used to operate household appliances, industrial equipment, and other electrical devices. Inverters play a critical role in renewable energy systems and uninterruptible power supplies (UPS), enabling the use of DC power in AC-dependent applications. 4

IoT The Internet of Things (IoT) refers to a network of interconnected devices that communicate and exchange data with each other over the internet. These devices are embedded with sensors, software, and other technologies, allowing them to collect and act on data they acquire from their environments. 1–5, 7, 8, 24–26, 34, 41, 47, 107–109, 131

Java Programming language used in current project with version 15. 71

JK BMS JK BMS (Battery Management System) is a battery management system developed by JK Power Limited, a company specializing in energy storage solutions and battery management technology. 125

Mapper A function or tool that processes data by transforming input elements into output elements usually called DTO by commonly using a class that has exposing rules. 71

MPPT MPPT stands for Maximum Power Point Tracking. It is a technology used in photovoltaic (PV) solar systems to optimize the power output from solar panels by continuously adjusting the operating conditions to match the maximum power point (MPP) of the solar panel's current-voltage (IV) curve. MPPT stands the same name for the algorithm and the physical device. x, 145

PECD Stands for Physical Environment Control Device. Englobes an actuator, a sensor and a circuit that controls a physical system with a logical controller. x, 37, 38, 140, 145, 146

Product Backlog An ordered list of everything that is known to be needed in the product. It is continually updated by the Product Owner. 36

SCRUM Scrum is based on principles such as transparency, inspection, and adaptation, allowing teams to address challenges and make changes as necessary quickly. It promotes a structured approach to managing projects while providing the flexibility to adapt to emerging business realities and customer needs. ix, 35, 36

Scrum Increment The sum of all the Product Backlog items completed during a sprint and all previous sprints. 36

Sprint Backlog A set of items selected from the Product Backlog to be completed in the sprint, plus a plan for delivering the product Increment and realizing the Sprint Goal. 36

story points Scrum story points are a unit of measure for expressing an estimate of the overall effort required to fully implement a product backlog item or any other piece of work. When we talk about effort, we're considering several factors including complexity, risks, and the amount of work to do. 36

WiFi WiFi is a wireless networking technology that allows electronic devices to connect to a network and the internet using radio waves. It provides high-speed internet and network connections without the need for physical cables. Devices such as smartphones, computers, smart TVs, and IoT devices can connect to a WiFi network to access the internet, share data, and communicate with other connected devices over distances typically up to a few hundred feet indoors and further outdoors. WiFi is based on the IEEE 802.11 family of standards and comes in various versions, offering different speeds and ranges. It has become a ubiquitous technology, found in homes, offices, and public spaces worldwide, providing convenient and flexible access to digital resources and connectivity. 140

Appendix A

Appendix

A.1 Channels definition

A.1.1 AlertChannel

Below are listed all the channels that handle alert statuses.

Channel	Description
_alert/current/all	Path Params: Void Input Body: Void Output: AlertChannelTransactionDto[] Description: Returns all current active alerts.
_alert/closed/all	Path Params: Void Input Body: Void Output: AlertChannelTransactionDto[] Description: Returns all closed alerts.
_alert/all	Path Params: Void Input Body: Void Output: AlertChannelTransactionDto[] Description: Returns all alerts, whether closed or active.
_alert/new	Path Params: Void Input Body: AlertChannelTransactionDto Output: AlertChannelTransactionDto Description: Creates a new alert and returns the created alert.
_alert/mod/{id}	Path Params: ID, long Input Body: AlertChannelTransactionDto Output: AlertChannelTransactionDto Description: Modifies an alert in the database and returns the modified alert.
_alert/del/{id}	Path Params: ID, long Input Body: Void Output: Void Description: Deletes an alert from the database.

A.1.2 BatteryChannel

Below are listed all channels related to batteries.

Channel	Description
_battery/all	Path Params: Void Input Body: Void Output: BatteryChannelTransactionDto[] Description: Returns all batteries.
_battery/by_id/{id}	Path Params: ID, long Input Body: Void Output: BatteryChannelTransactionDto Description: Returns a battery by its ID (without cells).
_battery/cells /by_id/{id}	Path Params: ID, long Input Body: Void Output: BatteryCellChannelTransactionDto[] Description: Returns the cells of a battery by its ID.

A.1.3 BatteryMetricChannel

Below are listed all channels related to battery metric generation.

Channel	Description
_battery_metric/all	Path Params: Void Input Body: Void Output: BatteryMetricChannelTransactionDto[] Description: Returns all battery metrics.

A.1.4 ConsumptionMetricChannel

Below are listed all channels related to consumption metric generation.

Channel	Description
_consumption_metric /all	Path Params: Void Input Body: Void Output: ConsumptionMetricChannelTransactionDto[] Description: Returns all consumption metrics.

A.1.5 DcmfChannel

Below are listed all channels related to DCMF.

Channel	Description
_dcmf/all	Path Params: Void Input Body: Void Output: DcmfChannelTransactionDto[] Description: Returns all DCMFs.
_dcmf/status	Path Params: Void Request Parameters: ID, long; enabled, boolean Input Body: Void Output: DcmfChannelTransactionDto Description: Returns DCMF status.

A.1.6 ElectricalSystemChannel

The following are listed all channels related to the electrical system.

Channel	Description
_electrical_system /all	Path Params: Void Input Body: Void Output: ElectricalSystemChannelTransactionDto[] Description: Returns all electrical parameters.

A.1.7 GenerationMetricChannel

Below are listed all channels related to energy generation metrics.

Channel	Description
_consumption_metric/all	Path Params: Void Input Body: Void Output: GenerationMetricChannelTransactionDto[] Description: Returns all energy generation metrics.

A.1.8 SolarChargerChannel

Below are listed all channels related to solar chargers.

Channel	Description
_solar_charger/all	Path Params: Void Input Body: Void Output: SolarChargerChannelTransactionDto[] Description: Returns all states of solar chargers.
_solar_charger/by_id/{id}	Path Params: ID, long Input Body: Void Output: SolarChargerChannelTransactionDto Description: Returns the state of the solar charger by ID.

A.1.9 UserChannel

Below are listed all channels related to users.

Channel	Description
_user/all	Path Params: Void Input Body: Void Output: UserChannelTransactionDto[] Description: Returns all system users.

A.1.10 UserChannel

Below are listed all channels related to users.

Channel	Description
_user/all	Path Params: Void Input Body: Void Output: UserChannelTransactionDto[] Description: Returns all system users.

A.2 BMS Screenshots

Below are listed partially the mounted batteries in rack A, which shelves stands for the key "Bx", meaning B2 stands for shelf 2, B3 stands for shelf 3, and so on. "Lx" stands for battery count (from left to right) for each shelf. As shown in figure A.1 the list of all the batteries inside given battery bank.

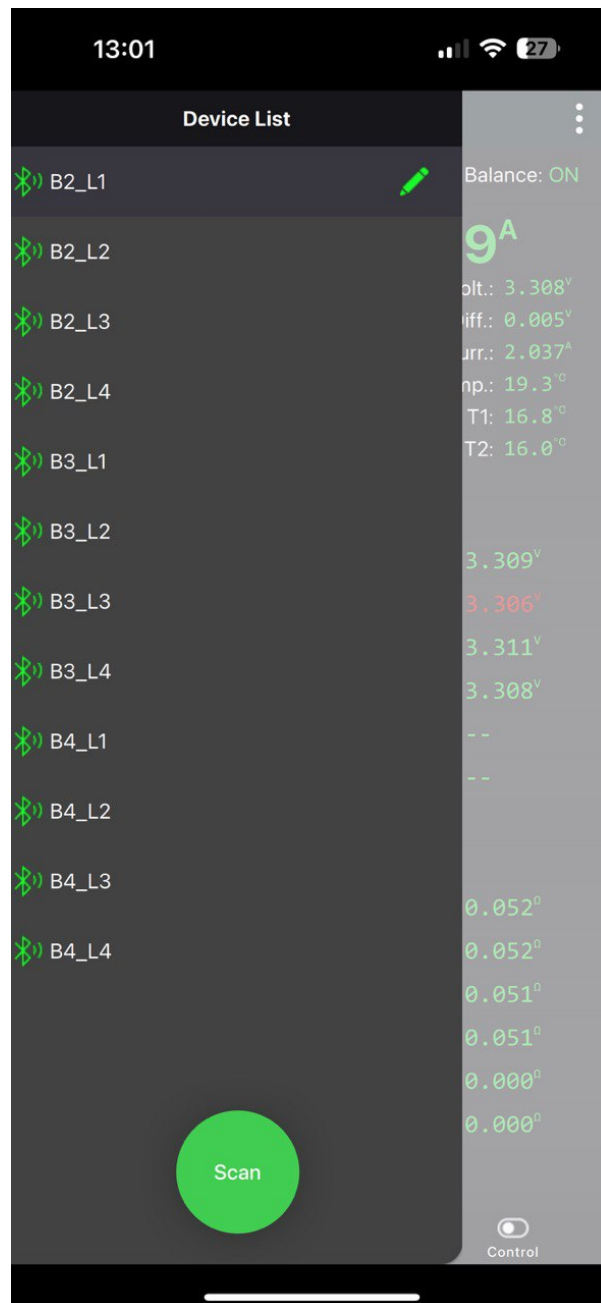


Figure A.1: Battery bank list (partial covered). Image property of Andrei García Cuadra taken in JK Bms.

Whenever accessing a battery by tapping into the name, we have the overview on the "status" tab of the application. From top to bottom as shown in figure A.2, we can review the battery charge/discharge/balance status (which can be on/off for each). Next, JK BMS app will summarize current voltage and amperes along with general battery less relevant information. Next section covers current battery cell voltages and cells resistance per each.

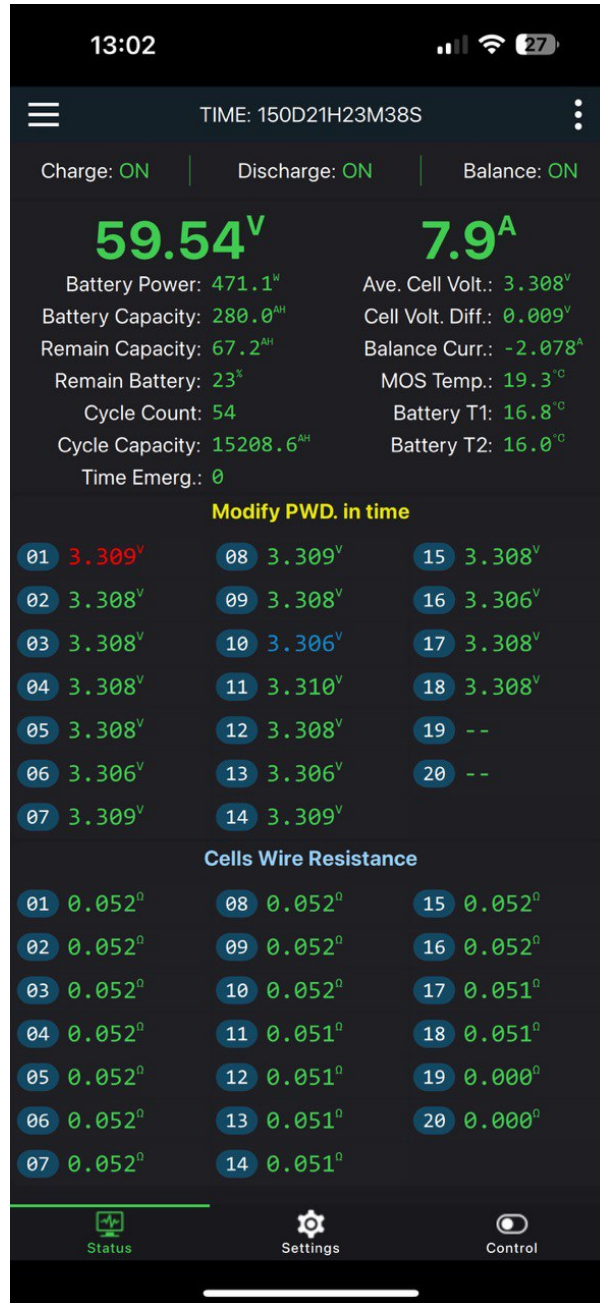


Figure A.2: Battery current status general overview. Image property of Andrei García Cuadra taken in JK Bms.

Appendix

As shown in figures A.3 and A.4 and A.5 and A.6, there are several settings that are taken in table 5.1.1. The configurations must be replicated in each battery as this is critical for the batteries to communicate between them.

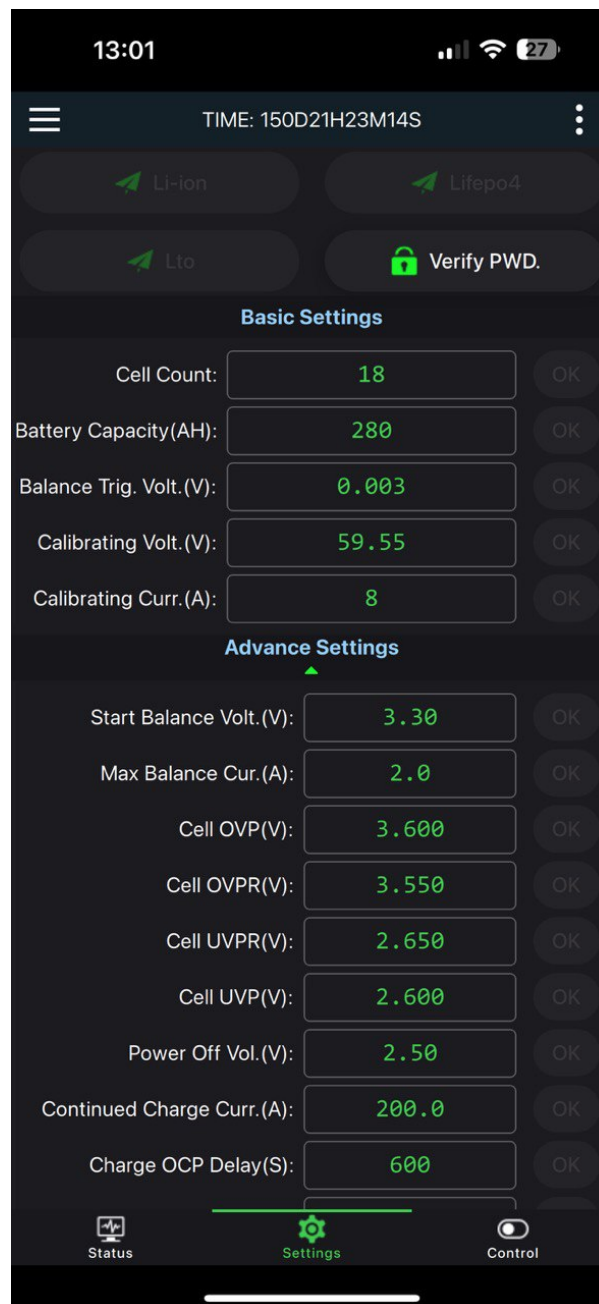


Figure A.3: Battery settings accessed by tapping tab "settings". This must be unlocked by using the button "Verify PWD" in order to make changes into the configuration. Image property of Andrei García Cuadra taken in JK Bms.

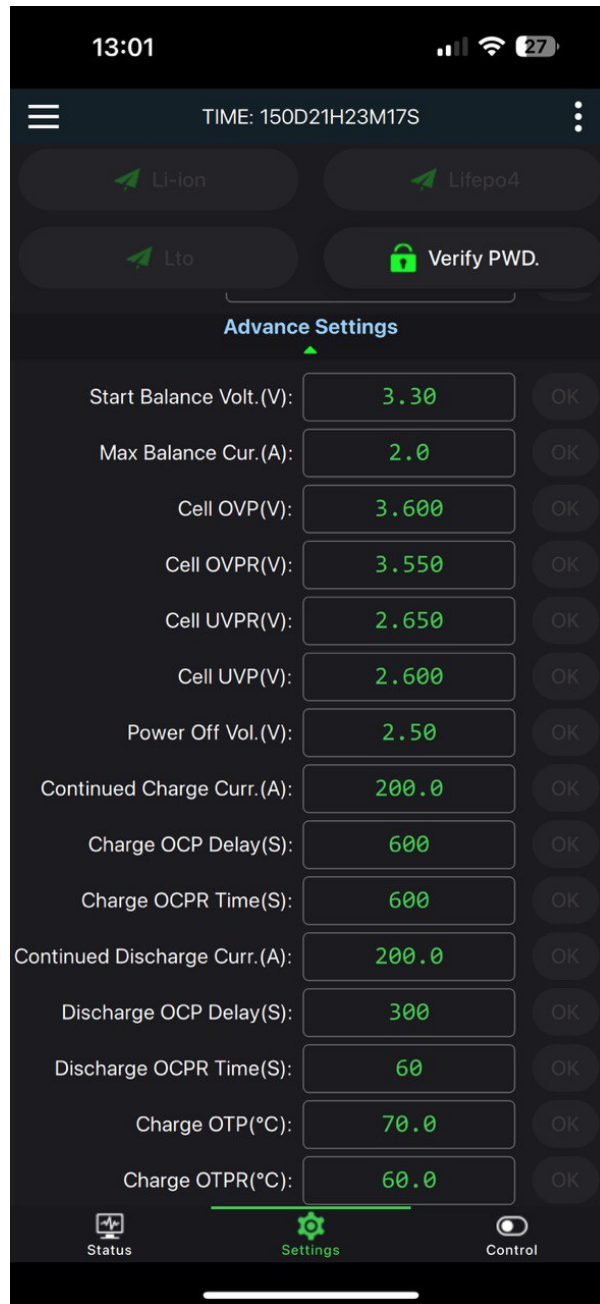


Figure A.4: Battery settings when scrolling down, as a continuation of figure A.3. Image property of Andrei García Cuadra taken in JK Bms.

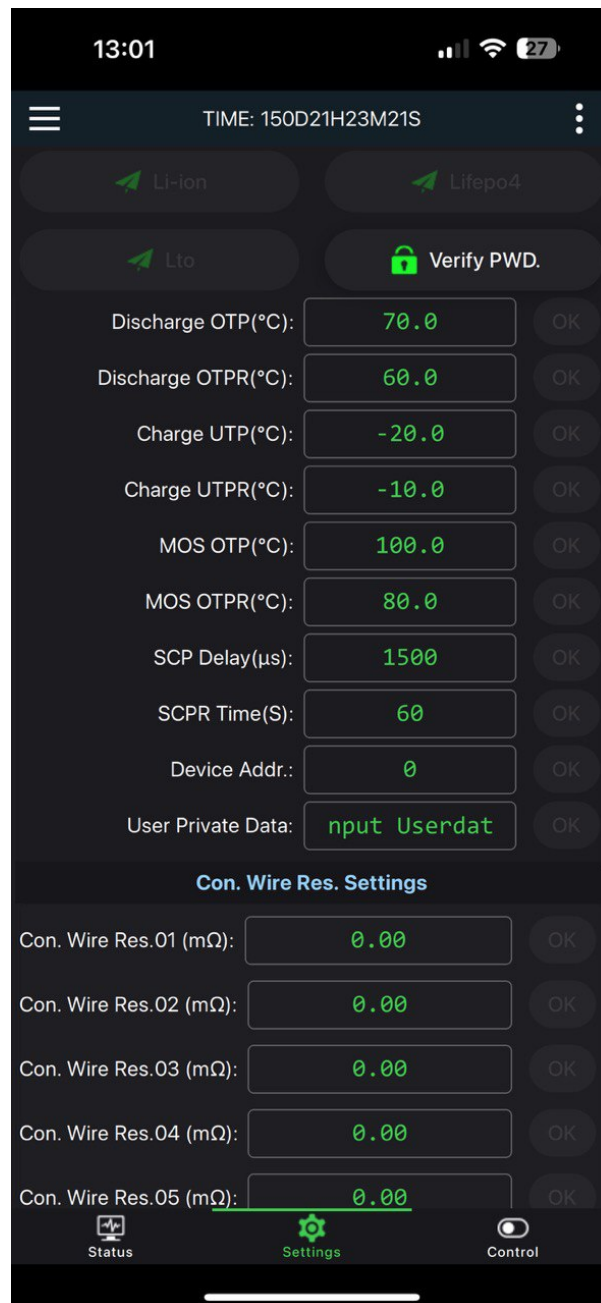


Figure A.5: Battery settings when scrolling down, as a continuation of figure A.4. Image property of Andrei García Cuadra taken in JK Bms.

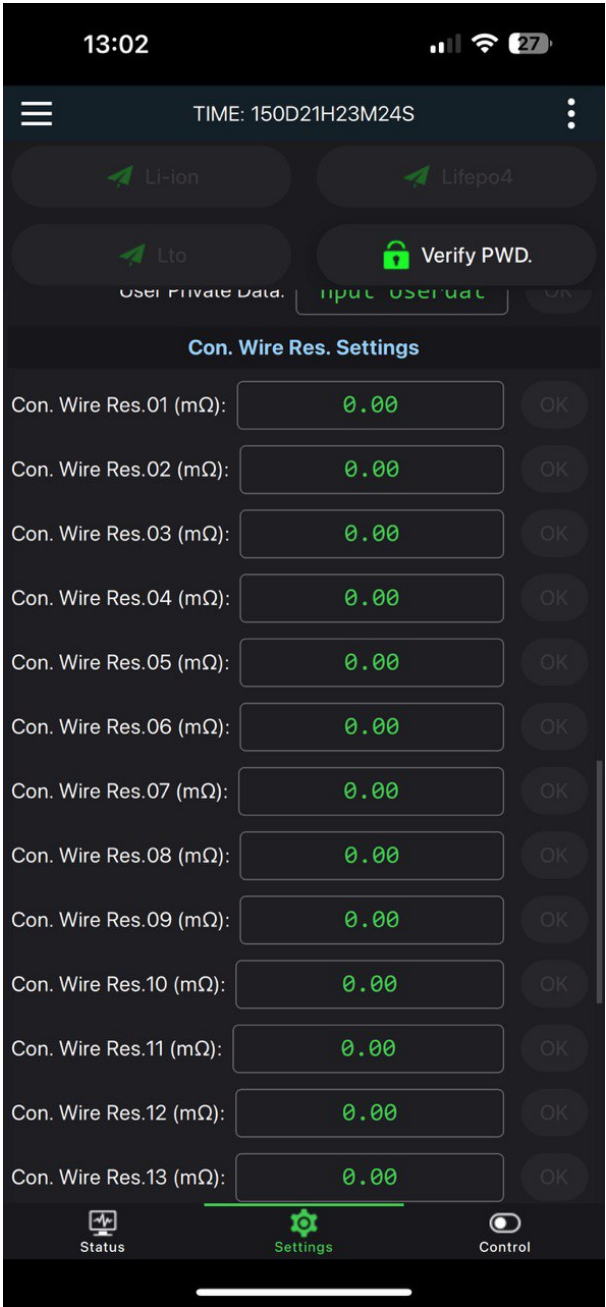


Figure A.6: Battery settings when scrolling down, as a continuation of figure A.5. Image property of Andrei García Cuadra taken in JK Bms.

Appendix

By tapping into "control" we can enable or disable several behaviours of JK BMS as shown in figure A.7 such as Charge, Discharge or Balance which are the primary ones.

Emergency and Disable Temp. sensor are related to use case situations when you need to reset delays or timers to enable battery as soon as possible.

Port switch must be set in RS485 in order to guarantee a good communication with B-Box.

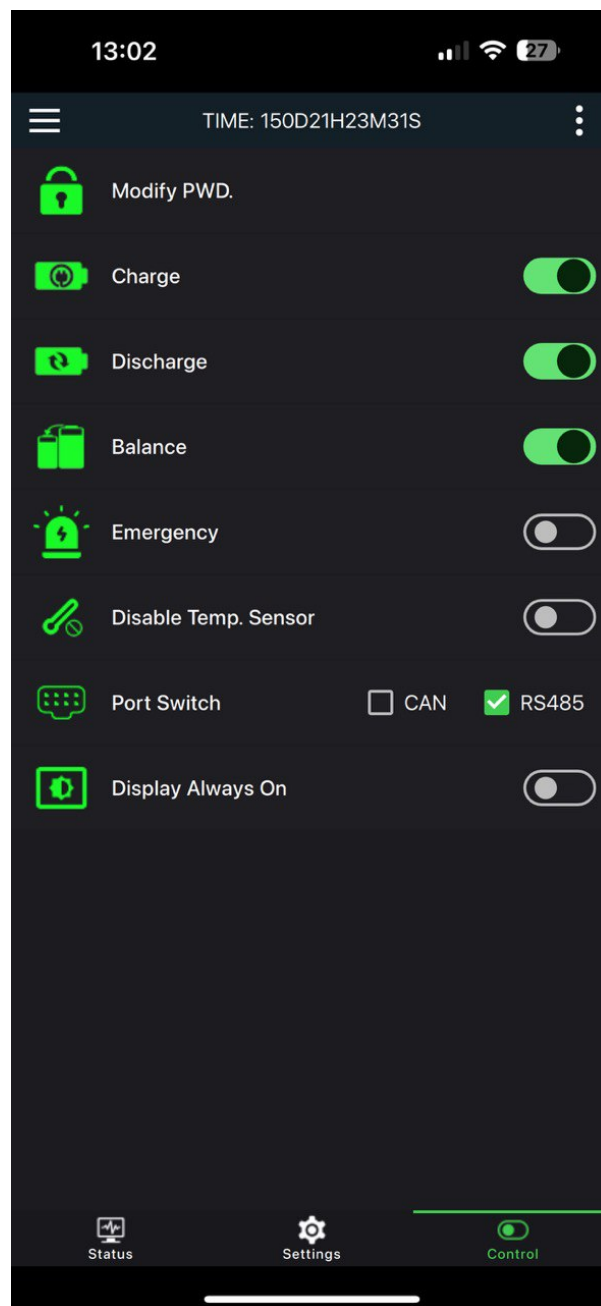


Figure A.7: Battery management controls by JK Bms. Image property of Andrei García Cuadra taken in JK Bms.

A.3 DTO implementation

A.3.1 AlertChannelTransactionDto

The following transactional object is responsible for modeling alerts, providing their Description and the activation expression.

```
1 @Data
2 public class AlertChannelTransactionDto {
3     private long id;
4     private String description;
5     private String activationExpression;
6     private LocalDateTime timestamp;
7 }
```

A.3.2 BatteryChannelTransactionDto

The following transactional object is responsible for modeling batteries. For this, it stores the current state of the battery such as the temperature, voltage, balance state, charging/discharging amperes, among others.

```
9  @Data
10 public class BatteryChannelTransactionDto {
11
12     private String id;
13     private boolean isCharging;
14     private float voltage;
15     private boolean enabled;
16     private String balanceStatus;
17     private float totalCapacity;
18     private float consumedCapacity;
19     private float remainingCapacity;
20     private float ampsFlow;
21     private byte mosTemp;
22     private byte tempSensor1;
23     private byte tempSensor2;
24     private byte soc;
25     private LocalDateTime timestamp;
26 }
```

A.3.3 BatteryCellChannelTransactionDto

The following transactional object is responsible for modeling a specific cell that belongs to a specific battery (battery cells are not shared among batteries). For this, it stores the cell voltage and the cell key. It is related to the battery from the database, being transparent to the DTO.

```
27 @Data
28 public class BatteryCellChannelTransactionDto {
29
30     private byte cellKey;
31     private float voltage;
32 }
```

A.3.4 BatteryMetricChannelTransactionDto

The following transactional object is responsible for modeling a metric for the battery, with its parametrization time and its value.

```
33 @Data
34 public class BatteryMetricChannelTransactionDto {
35     private long datestamp;
36     private int val;
37     private LocalDateTime timestamp;
38 }
```

A.3.5 ConsumptionMetricChannelTransactionDto

The following transactional object is responsible for modeling a metric for consumption, with its parametrization time and its value.

```
39 @Data
40 public class ConsumptionMetricChannelTransactionDto {
41     private long datestamp;
42     private int val;
43     private LocalDateTime timestamp;
44 }
```

A.3.6 DcmfChannelTransactionDto

The following transactional object is responsible for modeling parameters of a device such as the device name, its activation, its current consumption, among others.

```
45 @Data
46 public class DcmfChannelTransactionDto {
47
48     private String id;
49     private String deviceName;
50     private boolean enabled;
51     private String ipv4;
52     private int currentConsumption;
53     private DcmfUsageEnum usage;
54     private LocalDateTime timestamp;
55 }
```

A.3.7 ElectricalSystemChannelTransactionDto

The following transactional object is responsible for modeling the parameters of the electrical consumption of the site. In this phase of the project, it refers only to inverters.

```
56 @Data
57 public class ElectricalSystemChannelTransactionDto {
58     private String serial;
59     private float currentAmps;
60     private int powerWatts;
61     private String source;
62     private LocalDateTime timestamp;
63 }
```

A.3.8 GenerationMetricChannelTransactionDto

The following transactional object is responsible for modeling a metric for energy generation, with its parameterization time and its value.

```
64 @Data
65 public class GenerationMetricChannelTransactionDto {
66     private long datestamp;
67     private int val;
68     private LocalDateTime timestamp;
69 }
```

A.3.9 SolarChargerChannelTransactionDto

The following transactional object is responsible for modeling the parameters of the site's electric chargers. In this phase of the project, it refers only to inverters.

```
70 @Data
71 public class SolarChargerChannelTransactionDto {
72     private String id;
73     private String deviceName;
74     private SolarChargerStageEnum stage;
75     private Double outputDcVolts;
76     private Double outputDcAmps;
77     private Double inputWatts;
78     private Double inputVolts;
79     private LocalDateTime timestamp;
80 }
```

A.3.10 UserChannelTransactionDto

The following transactional object is responsible for modeling user parameters. It only allows viewing the ID and the *email*, in order to protect sensitive data and comply with European regulations.

```
81 @Data
82 public class UserChannelTransactionDto {
83
84     private long id;
85     private String email;
86     private LocalDateTime timestamp;
87 }
```

A.4 Shelly devices overview

All of the content of this appendix section was extracted from [Shelly(2024b)] and [Shelly(2024a)].

Shelly Plus 1

The Shelly Plus 1 is an innovative smart relay designed by Allterco Robotics, part of the Shelly range of IoT (Internet of Things) devices. It is engineered to offer advanced control over home electrical systems. This device is a testament to the evolution of home automation providing an efficient solution to modernize their environments. With its robust feature set and ease of installation was poised to become an essential component in the future of intelligent home design as now it became obsolete due to its sucesors.

The Shelly Plus 1 boasts a robust set of technical features that make it a powerful component in any smart home setup:

- **Power Supply:** 110-230V $\pm 10\%$, 50/60Hz AC or 24-240V DC
- **Max Load:** 16A
- **Wi-Fi Connectivity:** 802.11 b/g/n @ 2.4GHz
- **Energy Monitoring:** Precise current and power measurement for connected appliances
- **Operating Temperature:** -40°C to +40°C
- **Dimensions:** 42mm x 36mm x 14 mm
- **Color:** White

The Shelly Plus 1 is distinguished by its comprehensive feature set, designed to enhance user experience and device functionality:

- **Remote Control:** Users can control connected devices from anywhere in the world through the Shelly Cloud app or web interface.
- **Voice Assistant Compatibility:** Integration with popular voice assistants like Google Assistant and Amazon Alexa.
- **Advanced Scheduling:** Set schedules to automate devices based on time, sunrise/sunset, or other conditions.
- **Energy Monitoring:** Track energy consumption in real-time, allowing for efficient energy use and cost savings.
- **Fast Device Inclusion:** Easy setup and integration into existing home networks without the need for an additional hub.
- **Safety Features:** Over-temperature protection and precise control to ensure safe operation.

The versatility of the Shelly Plus 1 allows it to be used in a variety of applications, enhancing convenience, security, and energy efficiency:

- **Lighting Control:** Automate indoor and outdoor lighting, create scenes, and schedule operations for energy savings and convenience.
- **Appliance Management:** Remotely control and monitor the energy consumption of household appliances.
- **Heating and Cooling:** Integrate with HVAC systems for optimized climate control.
- **Security Systems:** Use in conjunction with sensors and alarms for enhanced security features.

Shelly Plus 1PM

Among these technological advancements, the Shelly Plus 1PM emerges as a versatile and potent gadget crafted to convert any residence into a more intelligent and responsive setting. The Shelly Plus 1PM signifies a remarkable progression in smart home automation technology. With its extensive range of features provides the chance to establish more intelligent living spaces. Developed by Allterco Robotics, the Shelly Plus 1PM is a smart Wi-Fi-enabled relay intended for automating electrical circuits and monitoring energy consumption in real-time.

- **Power Measurement:** Integrated precise power metering functionality for effective energy management.
- **Enhanced Safety:** Overload and temperature protection features ensure safe operation under all conditions.
- **Advanced Scheduling:** Automated control of connected devices based on time, sunrise/sunset, or predefined conditions.
- **Voice Control Compatibility:** Supports voice commands via Google Assistant, Amazon Alexa, and other voice control platforms.
- **Remote Access:** Control and monitor connected devices from anywhere using the Shelly Cloud app.
- **Electrical Rating:** Supports loads up to 16A, with a built-in power meter for real-time monitoring.
- **Connectivity:** Wi-Fi 802.11 b/g/n (2.4 GHz) support for seamless integration into home networks.
- **Operating Temperature:** Designed to operate between -40°C and +40°C, suitable for various environmental conditions.
- **Compact Design:** Small form factor allows for easy installation in junction boxes or behind existing wall switches.

The Shelly Plus 1PM's versatility allows it to be used in a wide range of applications, from simple lighting control to complex automation scenarios:

- **Intelligent Lighting:** Automate lighting based on occupancy, ambient light, or schedules to enhance convenience and energy efficiency.
- **Appliance Control:** Remotely manage appliances, allowing for on/off control and energy usage monitoring.

- **Heating and Cooling:** Optimize HVAC systems for comfort and energy savings by integrating climate control functionalities.
- **Security Systems:** Enhance home security by integrating the Shelly Plus 1PM with alarms and sensors for immediate notifications and actions.

Shelly Pro 1PM

One notable technological advancement in this field is the Shelly Plus 1PM, a versatile and powerful device designed to transform any home into a more intelligent and responsive space as it is designed to automate electrical circuits (mostly oriented in motors) and monitor energy consumption in real-time. It builds upon its predecessors by providing enhanced features, increased reliability, and an improved user interface.

The Shelly Pro 1PM boasts an array of features that distinguish it as a superior choice for smart home automation:

- **Precise Energy Monitoring:** Real-time tracking of energy consumption for connected devices, facilitating efficient energy management.
- **Advanced Load Control:** Ability to manage high-powered appliances up to 16A, with detailed scheduling and automation capabilities.
- **Seamless Integration:** Compatibility with major smart home ecosystems, including Google Home, Amazon Alexa, and MQTT, for comprehensive home automation.
- **Safety and Reliability:** Enhanced safety features such as overload protection, ensuring safe operation within the electrical network.
- **DIN Rail Mountable:** Designed for easy installation on DIN rails, making it suitable for both residential and commercial applications.

Technical Specifications

- **Power Supply:** 110-230V $\pm 10\%$, 50/60Hz AC
- **Max Load:** 16A
- **Wi-Fi Connectivity:** Supports 802.11 b/g/n networks for reliable wireless communication.
- **Energy Measurement:** Integrated power meter with high accuracy for effective energy management.
- **Operating Temperature:** -10°C to $+50^{\circ}\text{C}$, accommodating a wide range of environmental conditions.

Applications The versatility of the Shelly Pro 1PM allows for its application in various scenarios, enhancing the smart home experience:

- **Intelligent Lighting Control:** Automate lighting based on schedules, occupancy, or ambient conditions.
- **HVAC System Management:** Optimize heating, ventilation, and air conditioning for comfort and efficiency.

- **Security System Integration:** Enhance security protocols by integrating with alarms and sensors for immediate action or notification.
- **Energy Management:** Leverage the device's energy monitoring capabilities to reduce consumption and lower utility bills.

Shelly Plus 2PM

The Shelly Plus 2PM is a state-of-the-art device aimed at improving control over home electrical systems and offering real-time energy consumption data with a smart relay module featuring two independently controllable relays and power metering functionality. It is designed for easy integration into existing home networks, enabling smart control of lighting, heating, and other electrical devices. With its compact design and Wi-Fi connectivity along its dual-channel control, accurate energy monitoring, and simple installation make it a valuable asset for homeowners looking to enhance their home's intelligence and energy efficiency.

Key Features

- **Dual Independent Relays:** Control two devices or circuits independently, with a maximum load of 16A per channel.
- **Power Metering:** Integrated precise measurement of energy consumption for connected devices, aiding in energy management and cost savings.
- **Wi-Fi Connectivity:** Direct connection to the home Wi-Fi network, eliminating the need for a hub.
- **Voice Assistant Compatibility:** Support for voice control through integration with platforms like Google Assistant and Amazon Alexa.
- **Comprehensive App Control:** Access and manage the device remotely using the Shelly Cloud app, with options for scheduling, scenes, and automation.

Technical Specifications

- **Power Supply:** 110-230V $\pm 10\%$, 50/60Hz AC or 24-240V DC
- **Max Load:** 16A per channel with a total device maximum of 16A
- **Energy Monitoring:** High accuracy consumption measurement for each channel
- **Connectivity:** Wi-Fi 802.11 b/g/n
- **Operating Temperature:** -10°C to +50°C
- **Dimensions:** Compact design for easy installation in standard electrical boxes

Applications The flexibility of the Shelly Plus 2PM allows it to be utilized in a variety of smart home applications:

- **Lighting Automation:** Create schedules and scenes for indoor and outdoor lighting, enhancing convenience and energy efficiency.
- **Appliance Control:** Manage the operation of appliances remotely, ensuring safety and reducing energy wastage.

- **Heating and Cooling:** Automate HVAC systems for optimal climate control and comfort.
- **Security Systems:** Integrate with security systems for enhanced monitoring and control.

Shelly Pro 2PM

The rise of smart home technology has drastically changed the way we engage with our living spaces. In this realm of innovation, the Shelly Pro 2PM stands out as a prominent player aiming to revolutionize home automation serving as an intelligent relay module connected to Wi-Fi with two channels for managing electrical circuits and offering precise energy monitoring. It is tailored for professional applications in businesses, providing improved connectivity, safety, and automation options. Belonging to the Shelly Pro series renowned for its scalability in smart construction ventures because its sturdy construction together with its advanced energy monitoring and automation functions positions it as an excellent option for professionals seeking to elevate their commercial construction undertakings.

Key Features The Shelly Pro 2PM stands out for its comprehensive set of features designed to meet the demands of sophisticated smart home automation projects:

- **Dual Independent Relays:** Two channels with up to 16A per channel, allowing for the control of two separate electrical circuits.
- **Energy Monitoring:** Precise measurement of power consumption for each channel, enabling effective energy management and cost savings.
- **Wi-Fi and Ethernet Connectivity:** Offers flexible installation options and ensures stable network connections in various environments.
- **Advanced Automation and Control:** Supports complex scripting, scheduling, and conditional automation directly from the device.
- **Secure and Reliable:** Built with industrial-grade components and offers enhanced security features for safe operation.

Technical Specifications

- **Power Supply:** 110-230V $\pm 10\%$, 50/60Hz AC
- **Max Load:** 2x 16A
- **Connectivity:** Wi-Fi 802.11 b/g/n, Ethernet 10/100Mbps
- **Energy Monitoring Accuracy:** High precision monitoring with less than 1% deviation
- **Operating Temperature:** -10°C to +50°C
- **Installation:** DIN rail mountable design for easy integration into electrical panels

Applications The versatility of the Shelly Pro 2PM allows it to serve a wide range of applications in smart home and commercial automation:

- **Automated Lighting Control:** Schedule and manage lighting systems for enhanced comfort and energy efficiency.
- **HVAC Automation:** Optimize heating, ventilation, and air conditioning systems for improved climate control and energy savings.
- **Security and Access Control:** Integrate with security systems for real-time monitoring and control of access points.
- **Energy Management:** Monitor and manage energy consumption of connected devices, reducing operational costs and environmental impact.

Shelly Pro 3

The evolution of smart home technology has led to the development of sophisticated devices designed to enhance the efficiency, security, and convenience of modern living spaces. Among these innovations, the Shelly Pro 3 emerges as a notable advancement, offering a multifaceted solution for managing and automating home electrical systems featuring three independent channels for controlling electrical circuits alongside integrated energy monitoring capabilities. Engineered for commercial settings it provides robust performance making it suitable for sophisticated smart building projects.

Key Features The Shelly Pro 3 distinguishes itself with an array of features tailored to meet the diverse needs of smart automation projects:

- **Three Independent Relays:** Control three separate electrical circuits, enabling complex automation scenarios with a single device.
- **Precise Energy Monitoring:** Track the energy consumption of connected devices, facilitating effective energy management and cost reduction.
- **Wi-Fi and Ethernet Connectivity:** Ensures stable and versatile networking options for seamless integration into existing smart systems.
- **Advanced Automation Capabilities:** Supports scripting, scheduling, and conditional automation directly on the device for enhanced control.
- **Secure Operation:** Incorporates industrial-grade components and security features for reliable and safe use in various environments.

Technical Specifications

- **Power Supply:** 110-230V $\pm 10\%$, 50/60Hz AC
- **Max Load per Channel:** 10A
- **Connectivity:** Wi-Fi 802.11 b/g/n, Ethernet 10/100Mbps
- **Energy Monitoring:** High-precision measurement for accurate energy management
- **Operating Temperature:** -10°C to +50°C
- **Mounting:** DIN rail mountable, fitting standard electrical panels

Applications The Shelly Pro 3's versatility allows it to be used across a wide array of smart automation applications:

- **Intelligent Lighting Solutions:** Automate lighting based on schedules, occupancy, or ambient conditions for enhanced comfort and efficiency.
- **HVAC System Management:** Optimize heating, ventilation, and air conditioning for improved climate control and energy savings.
- **Security System Integration:** Enhance property security by connecting the Shelly Pro 3 with alarms and sensors for real-time monitoring and control.
- **Comprehensive Energy Management:** Utilize energy monitoring features to track and reduce the consumption of connected devices, lowering operational costs and environmental impact.

Shelly Plus i4

The advancement of smart home technology has led to the creation of advanced devices that aim to improve the effectiveness of contemporary living environments. One notable innovation in this field is the Shelly Pro 3 which provides a versatile solution to oversee and automate home electrical systems as it is an intelligent relay module that offers three separate channels to manage electrical circuits and includes built-in energy monitoring capabilities that is designed for use in commercial places.

Key Features The Shelly Plus i4 is equipped with several features that make it a versatile addition to any smart home system:

- **Four Input Channels:** Supports up to four external inputs, allowing for the integration of multiple switches or sensors.
- **Wi-Fi Connectivity:** Ensures easy integration into existing home networks for seamless communication with other smart devices.
- **Compact Design:** Its small footprint allows for easy installation behind existing wall switches or in tight spaces.
- **Advanced Automation Capabilities:** Can be programmed to trigger a wide range of actions and scenarios throughout the smart home ecosystem.
- **Cloud and Local Control:** Offers flexibility in management through cloud services or direct local connection without the need for an internet connection.

Technical Specifications

- **Connectivity:** 802.11 b/g/n Wi-Fi connection
- **Input Type:** 4 potential-free inputs, compatible with up to 12V DC sensors
- **Operating Temperature:** -10°C to +50°C, suitable for various environmental conditions
- **Power Supply:** 12V DC or 24-240V AC, offering flexibility for different installation needs
- **Dimensions:** 42mm x 38mm x 15 mm, ensuring it can fit in confined spaces

Applications The flexibility of the Shelly Plus i4 allows it to serve a multitude of purposes in a smart home environment:

- **Scene Activation:** Use switch presses to activate customized scenes, such as lighting moods, music playlists, or climate settings.
- **Security Enhancements:** Integrate door/window sensors to trigger alarms, notifications, or lighting when unexpected activity is detected.
- **Energy Management:** Connect to power consumption sensors to monitor and manage energy usage in various home appliances.
- **Convenience and Accessibility:** Simplify complex home automation tasks into single-button actions for enhanced usability and accessibility.

Shelly Qubino Wave 1PM

The convergence of smart home technologies has resulted in the creation of highly advanced devices that offer exceptional automation and efficiency in contemporary living spaces. The hypothetical Wave 1PM device is a revolutionary combination of Shelly's user-friendly automation, Qubino's sophisticated control systems, and the dependable Z-Wave communication technology to provide extensive control over household electrical systems while ensuring smooth communication via Z-Wave technology. It embodies the simplicity and efficiency of Shelly's design, the accuracy of Qubino's electrical control functionalities, and the resilience of Z-Wave's mesh network for dependable device communication. By incorporating Shelly's simplicity, Qubino's precise control features, and the reliable communication capabilities of Z-Wave technology, the Wave 1PM represents an innovative solution in smart home automation. Its comprehensive control features, along with detailed energy monitoring and a simple setup process, position it as an outstanding device to improve the intelligence and efficiency of upcoming modern residences.

Key Features The Wave 1PM boasts an array of features that underscore its innovative approach to home automation:

- **Z-Wave Plus Connectivity:** Utilizes the latest Z-Wave Plus technology for secure and reliable wireless communication between devices.
- **Power Metering:** Integrated energy monitoring allows for precise tracking of electricity consumption.
- **Multi-Device Compatibility:** Designed to work seamlessly with a wide range of smart home ecosystems and devices.
- **Advanced Scheduling:** Enables the automation of devices based on time, environmental conditions or user-defined scenarios.
- **Compact Design:** Its small form factor ensures easy installation in any standard electrical box without the need for additional space.

Technical Specifications

- **Power Supply:** 110-230V AC, 50/60Hz
- **Max Load:** 16A
- **Wireless Technology:** Z-Wave Plus (700 series chip)
- **Energy Measurement:** Accurate to within 1%

Appendix

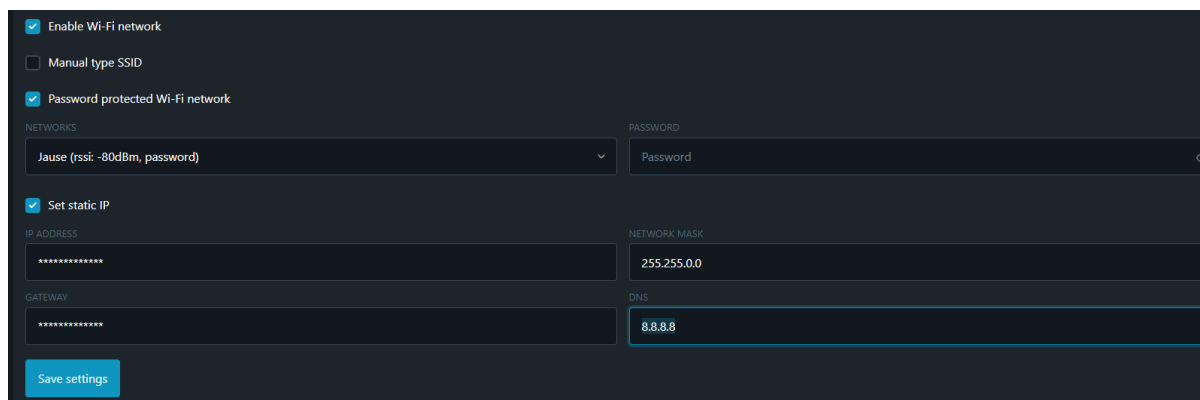
- **Operating Temperature:** -10°C to 40°C
- **Dimensions:** 42 x 37 x 17 mm

Applications The versatility of the Wave 1PM allows it to be utilized across various smart home scenarios:

- **Automated Lighting:** Control and schedule indoor and outdoor lighting systems for enhanced convenience and energy savings.
- **HVAC Management:** Optimize heating and cooling systems for comfort and efficiency.
- **Security Systems Integration:** Enhance home security by integrating alarms and sensors with Wave 1PM for automated responses.
- **Energy Monitoring:** Leverage the device's power metering feature of the device to track and reduce energy consumption in household appliances.

A.5 Shelly Configuration

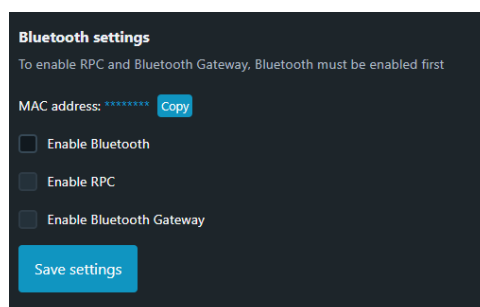
The following are the values used for the device configuration.



The screenshot shows the WiFi configuration interface for a Shelly DCMF device. It features a dark theme with blue accents. At the top, there are three checkboxes: 'Enable Wi-Fi network' (checked), 'Manual type SSID' (unchecked), and 'Password protected Wi-Fi network' (checked). Below these, there are two input fields: 'NETWORKS' with a dropdown menu showing 'Jause (rsst: -80dBm, password)' and 'PASSWORD' with a text input field containing 'Password'. Further down, there are four input fields: 'IP ADDRESS' (containing '*****'), 'NETWORK MASK' (containing '255.255.0.0'), 'GATEWAY' (containing '*****'), and 'DNS' (containing '8.8.8.8'). A 'Save settings' button is located at the bottom left.

Figure A.8: WIFI Configuration of Shelly *DCMF*. Image property of Andrei García Cuadra.

Fields that are empty or hidden behind round symbols are sensitive and have been omitted from this document since they are of a private nature and could pose a risk to the security of the installation. For this purpose, the development through figure A.8 is shown to configure the WiFi connection of the device.



The screenshot shows the Bluetooth settings interface for a Shelly DCMF device. It features a dark theme with blue accents. At the top, there is a section titled 'Bluetooth settings' with a note: 'To enable RPC and Bluetooth Gateway, Bluetooth must be enabled first'. Below this, there is a 'MAC address' field containing '*****' and a 'Copy' button. There are three checkboxes: 'Enable Bluetooth' (unchecked), 'Enable RPC' (unchecked), and 'Enable Bluetooth Gateway' (unchecked). A 'Save settings' button is located at the bottom left.

Figure A.9: Bluetooth Configuration of Shelly *DCMF*. Image property of Andrei García Cuadra.

In the subsequent figure A.9, the Bluetooth connectivity is configured by deactivating it to protect the device, taking the same preventive action as in figure A.10. It must be ensured that we obtain the latest version of our *PECD* and therefore figure A.11 is shown and ultimately, the access to the device must be configured through the settings shown in figure A.12.

The configuration may vary in the next published versions of the Shelly Web Admin Panel. Note that this was locked for version 1-beta.

MQTT settings

☐ Enable MQTT network

CONNECTION TYPE: No SSL

MQTT PREFIX: shellyplus1pm-80646fe4d050

☐ Enable "MQTT Control" ?

☐ Enable RPC over MQTT

☐ RPC status notifications over MQTT

☐ Generic status update over MQTT

SERVER: Eg: my-server.com:1883

CLIENT ID: shellyplus1pm-80646fe4d050

USERNAME: Username

PASSWORD: Password

Save settings

Figure A.10: MQTT Configuration of Shelly *DCMF*. Image property of Andrei García Cuadra.

Firmware information
Check for updates or install new versions of firmware

Firmware ID: 20231114-185726/1.1.0-beta2-g4lbfafd

Web Build ID: 231114184954-a0d728f5-HEAD

Official Shelly updates

Build	Version	ID
beta	1.1.0-beta3	20231213-091952/1.1.0-beta3-g9d8eacd
stable	1.0.8	20231106-160250/1.0.8-gdba0ee3

Firmware file image

Figure A.11: Version Configuration of Shelly *DCMF*. Image property of Andrei García Cuadra.

Authentication
Limit access to your device

☒ Password protected device

PASSWORD

Set or change device password

Empty spaces are not allowed

Save settings

Figure A.12: Authentication Configuration of Shelly *DCMF*. Image property of Andrei García Cuadra.

A.6 Electrical equipment specifications

Specification	Value
Model	JAsolar 405W
Nominal Power (Pmax)	405W
Voltage at Pmax (Vmp)	31.21V
Current at Pmax (Imp)	12.98A
Open-Circuit Voltage (Voc)	37.23V
Short-Circuit Current (Isc)	13.87A
Efficiency	20.5%
Operating Temperature	-40°C to +85°C
Dimensions	1722 x 1134 x 30 mm
Weight	21.5 kg
Warranty	12 years product, 25 years linear power output

Table A.11: JAsolar 405W Solar Panel Specifications. Extracted from the side of the bought solar panels.

A.7 App Screenshots

Figure A.13 allows us to review the current status of the application at a glance.

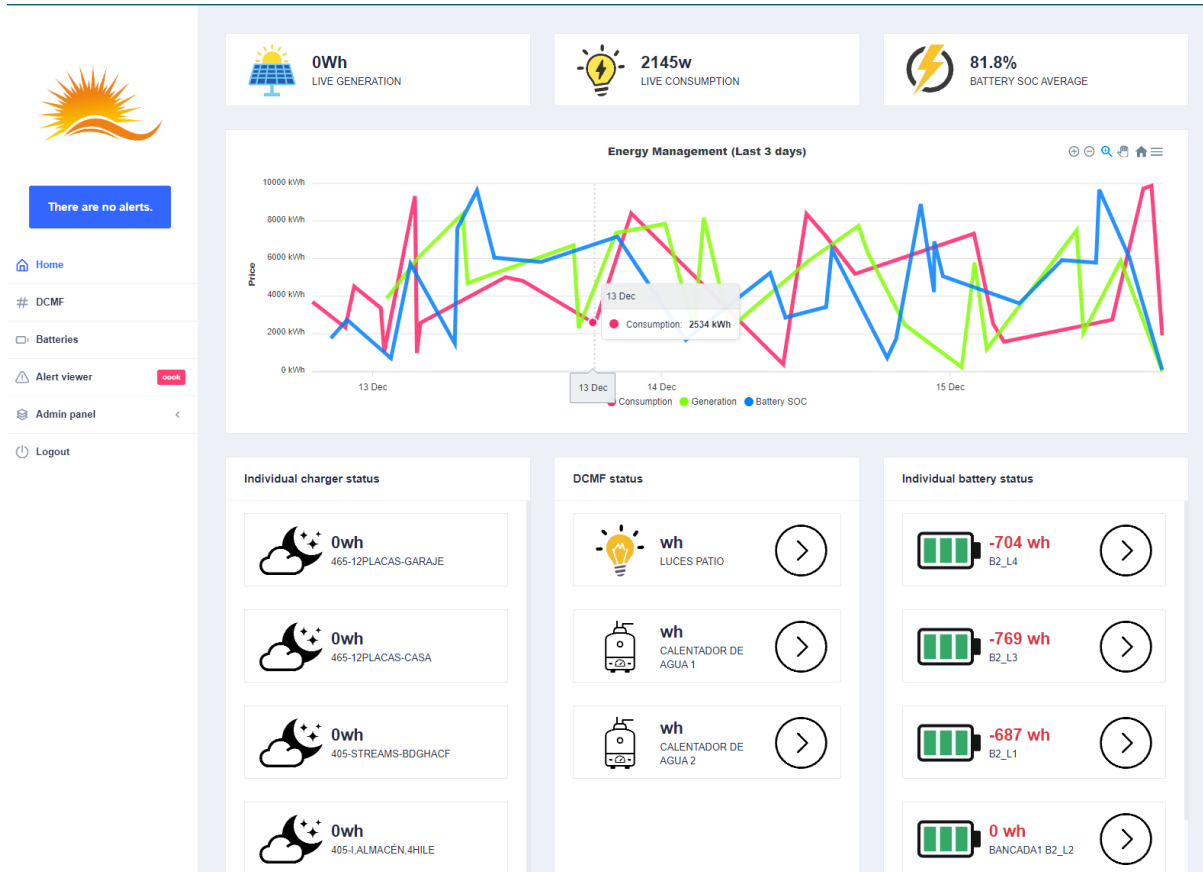


Figure A.13: Application main page where data can be analyzed with a first look and easy to read graphics. Image property of Andrei García Cuadra.

Figure A.14 allows for the evaluation of a battery with simple data of all interconnected batteries.

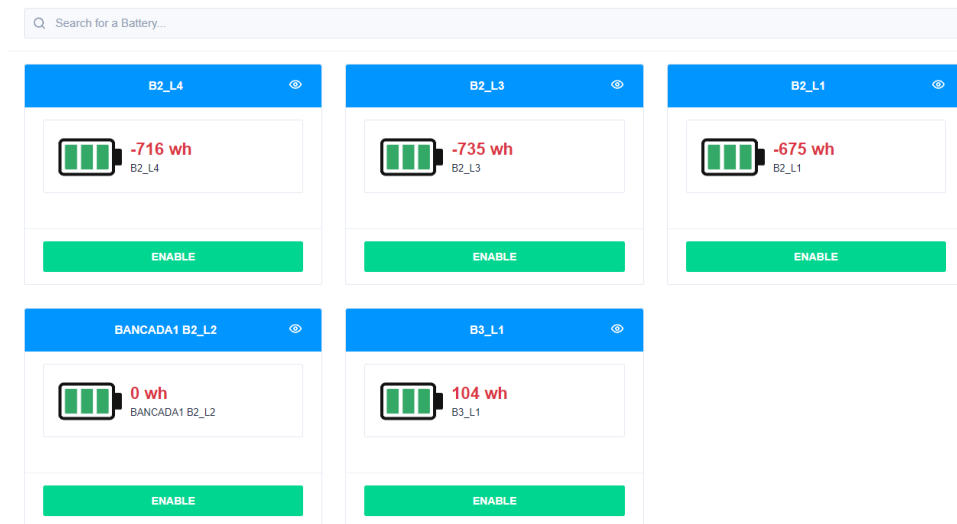


Figure A.14: General status of all the batteries in the bank. Image property of Andrei García Cuadra.

In the next figure A.15, mainly focused on a single battery, technical data is highly focused so that technical users are allowed to properly manage the battery status at a low level. It is easy to read, but the main focus is to allow the technical user to maintain the batteries without restrictions. There is also a grid about cell status of the battery and the management that is being done in real time to guarantee that it is properly balanced, secured and maintained.

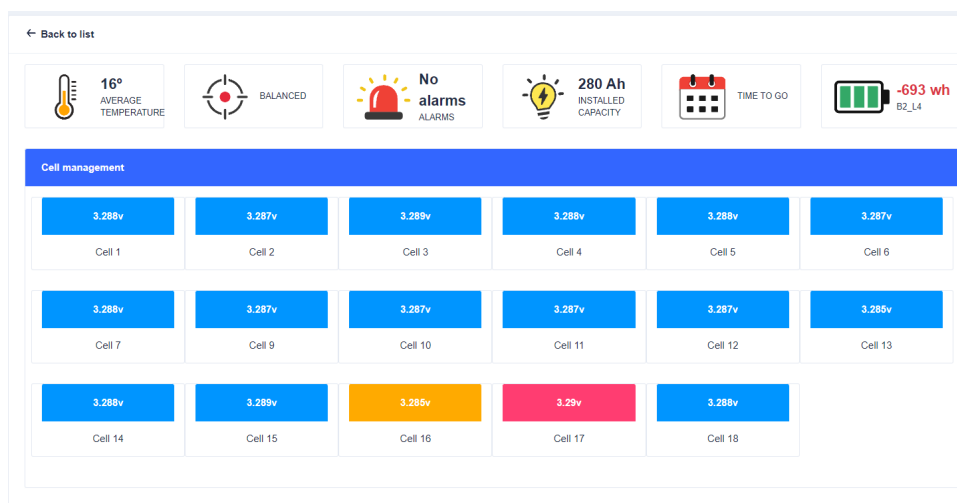


Figure A.15: Battery cells overview status. Image property of Andrei García Cuadra.

It is allowed to manage PECD devices by maintaining, enabling, and disabling them and also adding or removing them from the list as shown in figure A.16.

DCMF Management Crud				
ADD NEW				
Device Name :	Communication ip :	Consumption :	Status :	Actions :
Luces patio	192.168.10.13	0	DISABLE	 
Calentador de agua 1	192.168.10.11	0	ENABLE	 
Calentador de agua 2	192.168.10.12	0	ENABLE	 
3 total				

Figure A.16: Administration page of PECD. Image property of Andrei García Cuadra.

As shown in Figure A.17 we can check battery status in an easy way for non-technical users by using autodiscover protocol that shows or hides the batteries based on artificial intelligence.

Battery :	Gender :	Capacity :
B2_L4	Total ca	20% 250 Ah
B2_L3	Total ca	13% 305 Ah
B2_L1	Total ca	30% 250 Ah
BANCADA1 B2_L2	Total ca	13% 250 Ah
B3_L1	Total ca	17% 250 Ah
5 total		

Figure A.17: Battery administration page. Image property of Andrei García Cuadra.

The figure A.18 allows us to visualize and operate the status of the MPPT charger along with current production and limitations.

ID :	Device name :	Current output :	Current input :	Capacity :
HQ2138DVQGU	465-12placas-garaje	59.58V 0.00A	0W	OFF
HQ2208QVNZ2	465-12placas-casa	59.44V 0.00A	0W	OFF
HQ2203VR2NZ	405-streams-BDGHACF	59.63V 0.00A	0W	OFF
HQ2208M7XXT	405-l.almacén.4hile	59.47V 0.00A	0W	OFF
4 total				

Figure A.18: Administration page of MPPT (solar charge controllers). Image property of Andrei García Cuadra.

From figure A.19 all the PECD with its statuses are shown.

Q Search for a DCMF...		
Luces patio Status: Enabled Last update: 20 mins ago Last activation: 8 hours ago Current consumption: 0 ENABLE	Calentador de agua 1 Status: Disabled Last update: 20 mins ago Last activation: 8 hours ago Current consumption: 0 ENABLE	Calentador de agua 2 Status: Disabled Last update: 20 mins ago Last activation: 8 hours ago Current consumption: 0 ENABLE

Figure A.19: Status page of PECD in user mode (non expert). Image property of Andrei García Cuadra.

A.8 Raspberry PI GPIO Pins

Pin Number	Pin Name	Function
1	3.3V	Power
2	5V	Power
3	GPIO2	SDA1 (I2C)
4	5V	Power
5	GPIO3	SCL1 (I2C)
6	Ground	Ground
7	GPIO4	GPIO
8	GPIO14	TXD (UART)
9	Ground	Ground
10	GPIO15	RXD (UART)
11	GPIO17	GPIO
12	GPIO18	PWM0
13	GPIO27	GPIO
14	Ground	Ground
15	GPIO22	GPIO
16	GPIO23	GPIO
17	3.3V	Power
18	GPIO24	GPIO
19	GPIO10	MOSI (SPI)
20	Ground	Ground
21	GPIO9	MISO (SPI)
22	GPIO25	GPIO
23	GPIO11	SCLK (SPI)
24	GPIO8	CE0 (SPI)
25	Ground	Ground
26	GPIO7	CE1 (SPI)
27	ID_SD	Reserved
28	ID_SC	Reserved
29	GPIO5	GPIO
30	Ground	Ground
31	GPIO6	GPIO
32	GPIO12	PWM0
33	GPIO13	PWM1
34	Ground	Ground
35	GPIO19	MISO (SPI)
36	GPIO16	GPIO
37	GPIO26	GPIO
38	GPIO20	MOSI (SPI)
39	Ground	Ground
40	GPIO21	SCLK (SPI)

Table A.12: Raspberry Pi GPIO Pin Configuration. Table property of Raspberry PI corp [Pi(2024)].

A.9 STOMP with spring boot benefits

Criteria	Justification for STOMP
Compatibility with Spring Boot	STOMP integrates seamlessly with Spring Framework's WebSocket support, enabling easy setup and configuration within Spring Boot applications.
Protocol Support	STOMP provides a simple and extensible protocol for real-time messaging, allowing text-oriented messaging.
Ease of Use	With its simple text-oriented messaging format, STOMP is straightforward to understand and implement.
Scalability	STOMP, used over WebSocket, supports scalable real-time communication.
Interoperability	Being a high-level messaging protocol, STOMP can work on top of any reliable low-level communication protocol.
Message Broker Support	STOMP is supported by a range of message brokers (such as Apache Kafka) facilitating reliable message delivery and scalable architecture designs.
Community and Documentation	There is strong community support and comprehensive documentation available for implementing STOMP in Spring Boot, aiding in troubleshooting and development.
Feature Set	STOMP support a range of essential features for real-time communication, such as message acknowledgment, transactions, and subscription to message destinations.
Performance	While providing a high-level protocol with useful features, STOMP maintains good performance for real-time messaging needs, balancing feature richness with efficiency.

Table A.13: Real-Time selection criteria: STOMP with Spring Boot. Extracted from [Boot(2024)] and paraphrased for current dissertation.

A.10 Shelly IoT Devices overview

As there are many other alternatives in the Shelly ecosystem, we filtered few of them due to pricing and usage. Below is the comparison table as you can check every detail in below indexes of the section. For a full reference of shelly devices, please refer to chapter A.4.

Table A.14: Shelly Device Features Comparison, property of [Shelly(2024b)]

Feature	Plus 1	Plus 1PM	Pro 1PM	Plus 2PM	Pro 2PM	Pro 3	Plus i4
Relays	1	1	1	2	2	3	N/A
Power Metering	No	Yes	Yes	Yes	Yes	Yes	No
Connectivity	Wi-Fi	Wi-Fi	Wi-Fi + Ethernet	Wi-Fi	Wi-Fi + Ethernet	Wi-Fi + Ethernet	Wi-Fi
Max Load	16A	16A	16A	2x16A	2x16A	3x10A	N/A
Applications	Basic control	Energy monitoring	Advanced control	Dual control	Advanced dual control	Triple control	Input triggers

A.11 Victron communication: RJ-45 Ethernet

This section provides an in-depth analysis of the communication protocols developed by Victron Energy, a leader in energy management and automation solutions for renewable energy systems. Victron's protocols, including VE.Direct, VE.Can, and VE.Bus, are designed to ensure seamless and efficient communication between various components of a renewable energy system. Through VE.Direct, VE.Can, and VE.Bus, Victron ensures that its components work seamlessly together, maximizing efficiency and reliability.

A.11.1 VE.Direct Protocol

VE.Direct is a proprietary protocol used in Victron Energy products for point-to-point communication between low-power devices, such as solar charge controllers and battery monitors.

- **Simplicity:** Designed for direct connection between devices without the need for additional networking equipment.
- **Compatibility:** Supports a wide range of Victron devices, facilitating easy setup and integration (it is plug and play).
- **Data Transmission:** Enables the exchange of critical parameters.

A.11.2 VE.Can Protocol

VE.Can is based on the Controller Area Network (CAN bus) protocol, adapted by Victron for communication between higher-power devices and systems.

- **Robustness:** Offers reliable data transmission in electrically noisy environments,.
- **Scalability:** Allows for network expansion without significant increases in wiring complexity.
- **Interoperability:** Facilitates integration with third-party devices supporting the CAN bus standard.

A.11.3 VE.Bus Protocol

VE.Bus is an advanced communication protocol for managing connections between multiple power inverters and inverter/chargers in a single system.

- **System Integration:** Enables complex energy management tasks (phase balancing, power export control, battery management, etc.).
- **High Capacity:** Supports systems with significant power requirements by coordinating multiple units.
- **Flexibility:** Adaptable to various system configurations and expansion needs.

A.12 Junit testing code

A.12.1 Failing test (RED)

```
import static org.junit.Assert.*;
import org.junit.Test;

public class StackTest {

    @Test
    public void testIsEmpty() {
        Stack stack = new Stack();
        assertTrue(stack.isEmpty());
    }
}
```

A.12.2 Passing test (GREEN)

```
public class Stack {
    private int size = 0;

    public boolean isEmpty() {
        return size == 0;
    }
}
```

A.12.3 Refactor the code

```
public class Stack {
    private List<Object> elements = new ArrayList<>();

    public boolean isEmpty() {
        return elements.isEmpty();
    }
}
```

```
@Test
public void testPush() {
    Stack stack = new Stack();
    stack.push("element");
    assertFalse(stack.isEmpty());
}
```

A.13 MapStruct testing

```
@Mapper
public interface BatteryMapper {
    @Mapping(source = "voltage", target = "v")
    BatteryDto entityToDto(Battery battery);
}
```

```
@Mapper
public interface BatteryMapper {
    @Mapping(target = "voltageDate", source = "log.date",
        dateFormat = "dd-MM-yyyy")
    @Mapping(target = "voltage",
        expression = "java(battery.getCurrent().getVoltage())")
    OrderDto entityToDto(Battery b);
}
```

A.14 Reactive programming in Angular

```
import { Subject } from 'rxjs';

@Component({
  selector: 'app-root',
  template: 'Current value: {{ value }}'
})
export class AppComponent {
  value: number;
  private subject = new Subject<number>();

  constructor() {
    this.subject.subscribe({
      next: (v) => this.value = v
    });
  }

  updateValue(newValue: number) {
    this.subject.next(newValue);
  }
}
```

```
import { BehaviorSubject } from 'rxjs';

@Component({
  selector: 'app-data-component',
  template: 'Data: {{ data$ | async }}'
})
export class DataComponent {
  private dataSubject = new BehaviorSubject<string>('Initial Data');
  data$ = this.dataSubject.asObservable();

  updateData(newData: string) {
    this.dataSubject.next(newData);
  }
}
```

```
import { Subscription } from 'rxjs';

@Component({
  selector: 'app-subscribe-component',
  template: `Status: {{ status }}`
})
export class SubscribeComponent implements OnInit, OnDestroy {
  status: string;
  private subscription: Subscription;

  ngOnInit() {
    this.subscription = someObservable.subscribe({
      next: (status) => this.status = status
    });
  }

  ngOnDestroy() {
    this.subscription.unsubscribe();
  }
}
```

```
const userInterface = require('../src/userInterface');
const dataModel = require('../src/dataModel');

test('user interface displays correct battery status', () => {
  dataModel.setBatteryStatus('charging');
  const status = userInterface.displayStatus();
  expect(status).toBe('Battery is charging...');
});
```

Bibliography

- [Amoniac(2024)] Amoniac. 2024. HOW SCRUM WORKS AND WHY YOU SHOULD CHOOSE IT. <https://amoniac.eu/blog/post/how-scrum-works-and-why-you-should-choose-it> [Online; accessed April 27, 2024].
- [Ba-Alawi et al.(2023)] Abdulrahman H Ba-Alawi, Hai-Tra Nguyen, and ChangKyoo Yoo. 2023. Sustainable design of a solar/wind-powered reverse osmosis system with cooperative demand-side water management: A coordinated sizing approach with a fuzzy decision-making model. *Energy Conversion and Management* 295 (2023), 117624.
- [Baier et al.(2019)] Jonathan Baier, Gigi Sayfan, and Jesse White. 2019. *The Complete Kubernetes Guide: Become an expert in container management with the power of Kubernetes*. Packt Publishing Ltd, Packt Publishing Ltd.
- [Bairwa et al.(2022)] Bansilal Bairwa, Kanchana K Reddy, et al. 2022. Design of IoT based battery monitoring system. *AIP Publishing* 2461, 1 (2022), 100.
- [Banker et al.(2016)] Kyle Banker, Douglas Garrett, Peter Bakkum, and Shaun Verch. 2016. *MongoDB in action: covers MongoDB version 3.0*. Simon and Schuster, Simon and Schuster.
- [Barbón et al.(2023)] Arsenio Barbón, Ángel Gutiérrez, Luis Bayón, Covadonga Bayón-Cueli, and Javier Aparicio-Bermejo. 2023. Economic analysis of a pumped hydroelectric storage-integrated floating PV system in the day-ahead Iberian electricity market. *Energies* 16, 4 (2023), 1705.
- [Bode(1977)] Hans Bode. 1977. Lead-acid batteries. *John Wiley and Sons, Inc., New York, NY* 1, 1 (1977), 5–25.
- [Boot(2024)] Spring Boot. 2024. Spring Boot. <https://spring.io/guides/gs/spring-boot> [Online; accessed April 27, 2024].
- [Botejara-Antúnez et al.(2024)] Manuel Botejara-Antúnez, Alejandro Prieto-Fernández, Jaime González-Domínguez, Gonzalo Sánchez-Barroso, and Justo García-Sanz-Calcedo. 2024. Life cycle assessment of a LiFePO₄ cylindrical battery. *Environmental Science and Pollution Research* 1, 1 (2024), 1–17.
- [Buddy(2024)] Jpa Buddy. 2024. Entity to DTO – how-to. <https://jpa-buddy.com/blog/entity-to-dto-how-to/> [Online; accessed April 27, 2024].

- [Campbell and Papapetrou(2013)] G Ann Campbell and Patroklos P Papapetrou. 2013. *SonarQube in action*. Manning Publications Co., Manning Publications Co.
- [Chen et al.(2007)] Wei Chen, Hui Shen, Bifen Shu, Hong Qin, and Tao Deng. 2007. Evaluation of performance of MPPT devices in PV systems with storage batteries. *Renewable Energy* 32, 9 (2007), 1611–1622.
- [Cuadra(2024)] Andrei Garcia Cuadra. 2024. *IoT Architecture for the Optimization of Energy Consumption and Circular Energy Generation*. Ph.D. Dissertation. Universidad Internacional de Valencia.
- [Da Silva and Tavares(2015)] Maxwell Dayvson Da Silva and Hugo Lopes Tavares. 2015. *Redis Essentials*. Packt Publishing Ltd, Redis Essentials.
- [Daowd et al.(2011)] Mohamed Daowd, Noshin Omar, Peter Van Den Bossche, and Joeri Van Mierlo. 2011. Passive and active battery balancing comparison based on MATLAB simulation. In *2011 IEEE Vehicle Power and Propulsion Conference*. IEEE, IEEE, IEEE, 1–7.
- [DevCommunity(2022)] DevCommunity. 2022. Desktop Application Development in ElectronJS Using Angular. <https://dev.to/th3n00bc0d3r/desktop-application-development-in-electronjs-using-angular-4cno> [Online; accessed April 27, 2024].
- [Díaz-Longueira et al.(2023)] Antonio Díaz-Longueira, Álvaro Michelena, Míriam Timiraos, Francisco Zayas-Gato, Héctor Quintián, Carmen Benavides Cuellar, Héctor Alaiz-Moretón, José Luis Calvo-Rolle, and María Teresa García-Ordás. 2023. Comparative Analysis of Intelligent Techniques for Categorization of the Operational Status of LiFePo4 Batteries. *Springer* 1, 1 (2023), 545–553.
- [Dziadosz(2017)] Radoslaw Dziadosz. 2017. Liquibase: Version Control for Database Schema. *Jyvässkylän ammattikorkeakoulu* 1, 1 (2017), 1–15.
- [Elgendy et al.(2012)] Mohammed Elgendy, Bouchaib Zahawi, and D.J. Atkinson. 2012. Evaluation of perturb and observe MPPT algorithm implementation techniques. *ResearchGate* 1 (01 2012), 1–6. <https://doi.org/10.1049/cp.2012.0156>
- [Elmetennani et al.(2016)] S. Elmetennani, T.M. Laleg-Kirati, M. Djemai, and M. Tadjine. 2016. New MPPT algorithm for PV applications based on hybrid dynamical approach. *Journal of Process Control* 48 (2016), 14–24. <https://doi.org/10.1016/j.jprocont.2016.10.001>
- [Elwardany et al.(2024)] Mohamed Elwardany, AM Nassib, and Hany A Mohamed. 2024. Advancing Sustainable Thermal Power Generation: Insights from Recent Energy and Exergy Studies. *Process Safety and Environmental Protection* 1 (2024), 1–100.
- [Energies(2024a)] Victron Energies. 2024a. Cerbo GX full detailed manual. https://www.victronenergy.com/upload/documents/Cerbo_GX/140558-CCGX__Venus_GX__Cerbo_GX__Cerbo-S_GX_Manual-pdf-es.pdf [Online; accessed April 27, 2024].

BIBLIOGRAPHY

- [Energies(2024b)] Victron Energies. 2024b. LYNX full detailed manual. https://www.victronenergy.com/upload/documents/Lynx_Distributor/24531-Lynx_Distributor_Manual-pdf-es.pdf [Online; accessed April 27, 2024].
- [Energies(2024c)] Victron Energies. 2024c. Multiplus II full detailed manual. https://www.victronenergy.com/upload/documents/MultiPlus-II_230V/32424-MultiPlus-II____Quattro-II_120V-230V-pdf-es.pdf [Online; accessed April 27, 2024].
- [Fadelallah et al.(2023)] Issa Fadelallah, Ardeleanu Mihaita Nicolae, and Diaconu Mihai Emil. 2023. *IoT ESP32 device for remote experiment manipulation within a LiFePO₄ energy storage study*. IEEE, 2023 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI). 01–04 pages. <https://doi.org/10.1109/ECAI58194.2023.10194049>
- [Fetcenko et al.(2007)] MA Fetcenko, SR Ovshinsky, B Reichman, K Young, C Fierro, J Koch, A Zallen, W Mays, and T Ouchi. 2007. Recent advances in NiMH battery technology. *Journal of Power Sources* 165, 2 (2007), 544–551.
- [Fette and Melnikov(2011)] Ian Fette and Alexey Melnikov. 2011. *The websocket protocol*. Technical Report. University.
- [Florio and Ross(2020)] Nicholas Florio and Matthew Ross. 2020. CubeSat Electrical Power Subsystem (EPS) Design. *ResearchGate* 1 (02 2020). <https://doi.org/10.13140/RG.2.2.31293.08163>
- [Ginestà and Mora(2012)] Marc Gibert Ginestà and Oscar Pérez Mora. 2012. Bases de datos en PostgreSQL. *SI[sn]* 1, 1 (2012), 1–15.
- [Goldie(1996)] John Goldie. 1996. Ten ways to bulletproof RS-485 interfaces. *EDN* 41, 16 (1996), 141–147.
- [Gómez-Calvet et al.(2023)] Roberto Gómez-Calvet, José M Martínez-Duart, and Ana Rosa Gómez-Calvet. 2023. The 2030 power sector transition in Spain: Too little storage for so many planned solar photovoltaics? *Renewable and Sustainable Energy Reviews* 174 (2023), 113094.
- [González et al.(2024)] Paula Fernández González, María José Presno, and Manuel Landajo. 2024. Tracking the change in Spanish greenhouse gas emissions through an LMDI decomposition model: A global and sectoral approach. *Journal of Environmental Sciences* 139 (2024), 114–122.
- [Google(2024a)] Google. 2024a. GCP IoT reference platform. <https://cloud.google.com/architecture/connected-devices/iot-platform-product-architecture?hl=es-419> [Online; accessed April 27, 2024].
- [Google(2024b)] Google. 2024b. Google. <https://cloud.google.com/architecture/using-jenkins-for-distributed-builds-on-compute-engine?hl=es-419> [Online; accessed April 27, 2024].
- [Google(2024c)] Google. 2024c. Kubernetes Pods. [https://kubernetes.io/docs/concepts/workloads/pods/#:~:text=Pods%20in%20a%20Kubernetes%](https://kubernetes.io/docs/concepts/workloads/pods/#:~:text=Pods%20in%20a%20Kubernetes%20context)

- 20cluster, than%20managing%20the%20containers%20directly. [Online; accessed April 27, 2024].
- [Hajjar and Mourad(2019)] Haissam Hajjar and Hussein Mourad. 2019. Implementation of an FPGA-Raspberry Pi SPI Connection. In *The Twelfth International Conference on Advances in Circuits, Electronics and Micro-electronics*. Hussein, Hussein, 7–12.
- [Hassan et al.(2023)] Qusay Hassan, Ammar M Abdulateef, Saadoon Abdul Hafedh, Ahmed Al-samari, Jasim Abdulateef, Aws Zuhair Sameen, Hayder M Salman, Ali Khudhair Al-Jiboory, Szymon Wieteska, and Marek Jaszczur. 2023. Renewable energy-to-green hydrogen: A review of main resources routes, processes and evaluation. *International Journal of Hydrogen Energy* 1, 0 (2023), 0–10.
- [Hu et al.(2013)] Hongyang Hu, Di Zhang, and Yuezhi Zhou. 2013. A Design of Cloud Storage Gateway Based on Data Dispersal. In *International Conference on Computer, Networks and Communication Engineering (ICCNCE 2013)*. Atlantis Press, Atlantis Press, Atlantis Press, 605–608.
- [JASolar(2024)] JASolar. 2024. *JASolar 405w official specifications*. JASolar. <https://cdn.autosolar.es/pdf/JAM54S30%20390-415.pdf>
- [Jeyaseelan et al.(2020)] Christine Jeyaseelan, Antil Jain, Parul Khurana, Dinesh Kumar, and Sheenam Thatai. 2020. Ni-Cd Batteries. *Rechargeable Batteries: History, Progress, and Applications* 1, 1 (2020), 177–194.
- [Jung and Jung(2016)] Seunghun Jung and Ho-Young Jung. 2016. Charge/discharge characteristics of Li-ion batteries with two-phase active materials: a comparative study of LiFePO₄ and LiCoO₂ cells. *International Journal of Energy Research* 40, 11 (2016), 1541–1555.
- [Karaca et al.(2023)] Ali Erdogan Karaca, Ibrahim Dincer, and Michael Nitefor. 2023. A new renewable energy system integrated with compressed air energy storage and multistage desalination. *Energy* 268 (2023), 126723.
- [Karthikeyan et al.(2023)] S Karthikeyan, R Aakash Raj, Meenalosini Vimal Cruz, Lei Chen, JL Ajay Vishal, and VS Rohith. 2023. A systematic analysis on raspberry pi prototyping: Uses, challenges, benefits, and drawbacks. *IEEE Internet of Things Journal* 10, 16 (2023), 14397–14417.
- [KeyCloack(2024)] KeyCloack. 2024. Identity handling flow diagram. <https://walkingtree.tech/keycloak/> [Online; accessed April 27, 2024].
- [Köder et al.(2023)] Marius Köder, Tobias Winter, Markus Glaser, Marian Loos, and Markus Walter. 2023. Investigating electrochemical impedance spectroscopy for state of health estimation of LiFePO₄ batteries in idle state. *Journal of Energy Storage* 72 (2023), 108637.
- [Kojonsaari and Palm(2023)] A-R Kojonsaari and J Palm. 2023. The development of social science research on smart grids: a semi-structured literature review. *Energy, Sustainability and Society* 13, 1 (2023), 1.
- [Koskela(2007)] Lasse Koskela. 2007. *Test driven: practical tdd and acceptance tdd for java developers*. Simon and Schuster, Simon and Schuster.

BIBLIOGRAPHY

- [Krishnan and Gonzalez(2015)] SPT Krishnan and Jose L Ugia Gonzalez. 2015. *Building your next big thing with google cloud platform: A guide for developers and enterprise architects*. Springer, Springer.
- [Kumari et al.(2022)] Sarita Kumari, Sangeeta Gulia, Vikas Raman, and J Thirupathi. 2022. Enhancing the Security of STOMP Protocol and Its Comparison with other Encryption Methodologies. In *2022 International Conference on Smart and Sustainable Technologies in Energy and Power Sectors (SSTEPS)*. IEEE, IEEE, IEEE, 287–289.
- [Lee(2023)] Jeonghyeon Lee. 2023. ANALYSIS OF THE SPATIAL AND TEMPORAL CORRELATIONS BETWEEN SOLAR, WIND AND HYDRO POWER GENERATION IN SPAIN. *Universidad pontificia comillas* 1, 1 (2023), 200.
- [Lee et al.(2023)] Jechan Lee, Kun-Yi Andrew Lin, Sungyup Jung, and Eilhann E Kwon. 2023. Hybrid renewable energy systems involving thermochemical conversion process for waste-to-energy strategy. *Chemical Engineering Journal* 452 (2023), 139218.
- [Li et al.(2009)] Hong Li, Zhaoxiang Wang, Liquan Chen, and Xuejie Huang. 2009. Research on advanced materials for Li-ion batteries. *Advanced materials* 21, 45 (2009), 4593–4607.
- [Mazhar et al.(2023)] Tehseen Mazhar, Dhani Bux Talpur, Tamara Al Shloul, Yazeed Yasin Ghadi, Inayatul Haq, Inam Ullah, Khmaies Ouahada, and Habib Hamam. 2023. Analysis of IoT security challenges and its solutions using artificial intelligence. *Brain Sciences* 13, 4 (2023), 683.
- [Medium(2024)] Medium. 2024. Spring MVC and Architecture. <https://medium.com/@poorwakesharwani/spring-mvc-and-architecture-812074153d34> [Online; accessed April 27, 2024].
- [Mohammed Elgendy (ResearchGate)(2024)] Mohammed Elgendy (ResearchGate). 2024. MPPT Flow diagram. https://www.researchgate.net/figure/Flowchart-of-P-O-MPPT-algorithm_fig9_261080714 [Online; accessed April 27, 2024].
- [Nadeem(2023)] Anam Nadeem. 2023. Sustainable pathways for electrification development: a case study for Pakistan. *Università Politecnica delle Marche* 1 (2023), 1–10.
- [Nasuni(2024)] Nasuni. 2024. Distributed storage. <https://www.grayhats.com/es/blog/nasuni-a-blockchain-like-system-to-store-information/> [Online; accessed April 27, 2024].
- [Nkon(2022)] Nkon. 2022. JK Smart Active Balance - NKON. <https://eu.nkon.nl/jk-bms-bd6a17s8p.html> [Online; accessed April 27, 2024].
- [Olabi et al.(2023)] AG Olabi, Aasim Ahmed Abdelghafar, Hussein M Maghrabie, Enas Taha Sayed, Hegazy Rezk, Muaz Al Radi, Khaled Obaideen, and Mohammad Ali Abdelkareem. 2023. Application of artificial intelligence for prediction, optimization, and control of thermal energy storage systems. *Thermal Science and Engineering Progress* 39 (2023), 101730.

- [Paiva(2021)] Daicy Patricia Duarte Paiva. 2021. Módulos de reautenticación para el servidor de autenticaciones Keycloak. *Revista sobre estudios e investigaciones del saber académico* 1, 15 (2021), e2021001–e2021001.
- [Pan and Dong(2023)] Yuling Pan and Feng Dong. 2023. Green finance policy coupling effect of fossil energy use rights trading and renewable energy certificates trading on low carbon economy: Taking China as an example. *Economic Analysis and Policy* 77 (2023), 658–679.
- [Parhamfar and Adeli(2022a)] Mohammad Parhamfar and Amir Adeli. 2022a. The Study of Electrical Grid Components After Installing a 10 MW Photovoltaic Power Plant with Large-Scale Batteries at Peak Load by DigSilent Software. *SciencePG* 1, 1 (01 2022). <https://doi.org/10.11648/j.epes.20221105.12>
- [Parhamfar and Adeli(2022b)] Mohammad Parhamfar and Amir Adeli. 2022b. The Study of Electrical Grid Components After Installing a 10 MW Photovoltaic Power Plant with Large-Scale Batteries at Peak Load by DigSilent Software. *SciencePG* 1, 1 (01 2022). <https://doi.org/10.11648/j.epes.20221105.12>
- [Patton and Economy(2014)] Jeff Patton and Peter Economy. 2014. *User story mapping: discover the whole story, build the right product*. " O'Reilly Media, Inc.", " O'Reilly Media, Inc.".
- [Pi(2024)] Raspberry Pi. 2024. Raspberry Pi. <https://projects.raspberrypi.org/en/projects/physical-computing/1> [Online; accessed April 27, 2024].
- [Ponelat and Rosenstock(2022)] Joshua S Ponelat and Lukas L Rosenstock. 2022. *Designing APIs with Swagger and OpenAPI*. Simon and Schuster, Simon and Schuster.
- [Ratamaa(2014)] Tommi Ratamaa. 2014. *Generic Data Transfer Object and Entity Mapping*. Master's thesis. University.
- [ResearchGate(2024)] ResearchGate. 2024. Lithium Iron Phosphate batteries. <https://www.researchgate.net/publication/280886489/figure/fig2/AS:669007095033870@1536515129588/Chemical-reaction-of-a-LiFePO4-cell-21.ppm> [Online; accessed April 27, 2024].
- [Sachdeva(2016)] Sakshi Sachdeva. 2016. Scrum Methodology. *Int. J. Eng. Comput. Sci* 5, 16792 (2016), 16792–16800.
- [Salituro(2020)] Eric Salituro. 2020. *Learn Grafana 7.0: A beginner's guide to getting well versed in analytics, interactive dashboards, and monitoring*. Packt Publishing Ltd, Packt Publishing Ltd.
- [Samargandi et al.(2024)] Nahla Samargandi, Md Monirul Islam, and Kazi Sohag. 2024. Towards realizing vision 2030: Input demand for renewable energy production in Saudi Arabia. *Gondwana Research* 127 (2024), 47–64.
- [Sarcar(2022)] Vaskaran Sarcar. 2022. Understanding SOLID Principles. In *Java Design Patterns: A Hands-On Experience with Real-World Examples*. Springer, Springer, 3–64.
- [saVRee(2024)] saVRee. 2024. Flooded lead acid batteries. [https://savree-storage.s3.amazonaws.com/Articles/optimised/Cutaway%](https://savree-storage.s3.amazonaws.com/Articles/optimised/Cutaway%20Diagram%20of%20a%20Flooded%20Lead-Acid%20Battery.pdf)

BIBLIOGRAPHY

- 20View%20of%20a%20Typical%20Battery%20Used%20For%20Starting%2C%20Lighting%2C%20and%20Ignition%20(SLI).png [Online; accessed April 27, 2024].
- [Schwaber(1997)] Ken Schwaber. 1997. Scrum development process. *Springer* 32, 9 (1997), 117–134.
- [Science(2024)] Let's Talk Science. 2024. Flow of energy fossile production. <https://letstalkscience.ca/educational-resources/backgrounders/generating-electricity-fossil-fuels> [Online; accessed April 27, 2024].
- [ScienceDirect(2024)] ScienceDirect. 2024. Nickel-cadmium batteries. <https://ars.els-cdn.com/content/image/3-s2.0-B9780128139752000065-f06-06-9780128139752.jpg> [Online; accessed April 27, 2024].
- [Sebasthirani et al.(2023a)] K. Sebasthirani, G. Rajesh, Murapaka Dhanalakshmi Bhavani, Niharika Varshney, T. Sathish, and K. Karthigaivel. 2023a. *IoT-Based Second-Life Storage System Prototype for Lifepo4 Batteries*. IEEE, 2023 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI). 1–7 pages. <https://doi.org/10.1109/ACCAI58221.2023.10199869>
- [Sebasthirani et al.(2023b)] K. Sebasthirani, G. Rajesh, Murapaka Dhanalakshmi Bhavani, Niharika Varshney, T. Sathish, and K. Karthigaivel. 2023b. *IoT-Based Second-Life Storage System Prototype for Lifepo4 Batteries*. *IEEE* 1, 1 (2023), 1–7.
- [SemanticScholar(2024a)] SemanticScholar. 2024a. Lithium ion batteries. <https://www.researchgate.net/publication/297719133/figure/fig7/AS:342647265873930@1458704882680/Schematic-drawing-of-a-typical-lithium-ion-battery.png> [Online; accessed April 27, 2024].
- [SemanticScholar(2024b)] SemanticScholar. 2024b. Nickel-metal batteries. <https://d3i71xaburhd42.cloudfront.net/5a915f5bf77684aaa66209969569f3be4592d91b/2-Figure1-1.png> [Online; accessed April 27, 2024].
- [Shapira et al.(2021)] Gwen Shapira, Todd Palino, Rajini Sivaram, and Krit Petty. 2021. *Kafka: the definitive guide*. " O'Reilly Media, Inc.", " O'Reilly Media, Inc."
- [Shelly(2024a)] Shelly. 2024a. Api information. <https://shelly-api-docs.shelly.cloud/> [Online; accessed April 27, 2024].
- [Shelly(2024b)] Shelly. 2024b. Catalogue. https://info.shelly.cloud/documents/Shelly-Catalogue_2023_EU_v04_web_low.pdf?_gl=1*1i12jmq*_ga*MTU1Mjk3NDE2Ny4xNjc2MzU4MjE0*_ga_RG1ZWH9CCQ*MTY3NjQ0NDkwNi4yLjEuMTY3NjQ0NTIxOS42MC4wLjA. [Online; accessed April 27, 2024].
- [Smart(2011)] John Ferguson Smart. 2011. *Jenkins: The Definitive Guide: Continuous Integration for the Masses*. " O'Reilly Media, Inc.", " O'Reilly Media, Inc."

- [smashingtips(2024)] smashingtips. 2024. Federated modules. <https://smashingtips.com/programming/mastering-webpack-module-federation-solve-singleton-issues-code-snippets/> [Online; accessed April 27, 2024].
- [Soares(2023)] Diogo Valentim Carvalho Soares. 2023. *Analysis of Module Federation Implementation in a Micro-Frontend Application*. Ph. D. Dissertation. Porto.
- [Solar(2024)] Fesat Solar. 2024. Etapas de carga de una batería. <https://fesatsolar.com/etapas-de-carga-de-una-bateria-de-ciclo-profundo-acido-plomo-2/> [Online; accessed April 27, 2024].
- [Tahir and Al-Ghamdi(2023)] Furqan Tahir and Sami G Al-Ghamdi. 2023. Climatic change impacts on the energy requirements for the built environment sector. *Energy Reports* 9 (2023), 670–676.
- [Taibi et al.(2018)] Davide Taibi, Valentina Lenarduzzi, and Claus Pahl. 2018. Architectural patterns for microservices: a systematic mapping study. In *CLOSER 2018: Proceedings of the 8th International Conference on Cloud Computing and Services Science; Funchal, Madeira, Portugal, 19-21 March 2018*. SciTePress, SciTePress, 1–100.
- [Takahashi et al.(2005)] M Takahashi, H Ohtsuka, K Akuto, and Y Sakurai. 2005. Confirmation of long-term cyclability and high thermal stability of LiFePO₄ in prismatic lithium-ion cells. *Journal of the Electrochemical Society* 152, 5 (2005), A899.
- [Thorgersen and Silva(2021)] Stian Thorgersen and Pedro Igor Silva. 2021. *Keycloak-identity and access management for modern applications: harness the power of Keycloak, OpenID Connect, and OAuth 2.0 protocols to secure applications*. Packt Publishing Ltd, Packt Publishing Ltd.
- [TutorialsPoint(2024)] TutorialsPoint. 2024. Voltage kinds. <https://www.tutorialspoint.com/nominal-voltage-rated-voltage-and-operating-voltage#:~:text=Conclusion&text=The%20voltage%20value%20of%20electrical,is%20called%20the%20operating%20voltage.> [Online; accessed April 27, 2024].
- [Vuika(2019)] Denys Vuika. 2019. *Electron Projects: Build over 9 cross-platform desktop applications from scratch*. Packt Publishing Ltd, Packt Publishing Ltd.
- [Walczak et al.(2023)] Renata Walczak, Krzysztof Koszewski, Robert Olszewski, Krzysztof Ejmont, and Anikó Kálmán. 2023. Acceptance of IoT edge-computing-based sensors in smart cities for universal design purposes. *Energies* 16, 3 (2023), 1024.
- [Wang et al.(2023)] Qiang Wang, Jiale Guo, Rongrong Li, and Xue-ting Jiang. 2023. Exploring the role of nuclear energy in the energy transition: A comparative perspective of the effects of coal, oil, natural gas, renewable energy, and nuclear power on economic growth and carbon emissions. *Environmental research* 221 (2023), 115290.

BIBLIOGRAPHY

- [Wikipedia(2024)] Wikipedia. 2024. Icezum Alhambra Open FPGA. https://es.m.wikipedia.org/wiki/Archivo:Icezum_Alhambra_Open_FPGA_electronic_board.png [Online; accessed April 27, 2024].
- [Yagci et al.(2022)] Mehmet Yagci, Thomas Feldmann, Elmar Bollin, Michael Schmidt, and Wolfgang Bessler. 2022. Aging Characteristics of Stationary Lithium-Ion Battery Systems with Serial and Parallel Cell Configurations. *Energies* 15 (05 2022), 3922. <https://doi.org/10.3390/en15113922>
- [Zhang(2011)] Wei-Jun Zhang. 2011. Structure and performance of LiFePO₄ cathode materials: A review. *Journal of Power Sources* 196, 6 (2011), 2962–2970.
- [Zhang et al.(2014)] Zheyu Zhang, Fred Wang, Leon M Tolbert, Benjamin J Blalock, and Daniel J Costinett. 2014. Evaluation of switching performance of SiC devices in PWM inverter-fed induction motor drives. *IEEE Transactions on Power Electronics* 30, 10 (2014), 5701–5711.