Design and Implementation of Integrated Smart Home Energy Management Systems for Clusters of Buildings

A Thesis in Department of

Building, Civil and Environmental Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of
Master of Applied Science (Building engineering) at
Concordia University

Montréal, Québec, Canada

August 2022

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: Reza Rajaei

Entitled: Design and Implementation of Integrated Smart Home Energy Management Systems for Clusters of Buildings

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Building Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

____________________________________ Chair

Dr. Mazdak Nikbakht

____________________________________ Examiner

Dr. Mazdak Nikbakht

____________________________________ Examiner

Dr. Mohamed Ouf

____________________________________ Supervisor

Dr. Ursula Eicker

Approved by

____

Dr. Ashutosh Bagchi

Chair of Department of Building, Civil and Environmental Engineering

____

Dr. Mourad Debbabi

Chair of Gina Cody School of Engineering and Computer Scienc

ABSTRACT

Design and Implementation of Integrated Smart Home Energy Management Systems for Clusters of Buildings

Reza Rajaei

To ensure the balance between power generation and demand at the peak hours, power utilities need to keep additional power generation capacity on standby which usually causes higher operational cost. This will result in variations in hourly to seasonal electricity prices. With a focus on the physical layer integration, we developed a methodology which includes a prototype of a cluster of smart homes with energy management systems (SHEMS) to study and harvest the flexibility in the demand side in the residential building sector by monitoring and controlling the loads at appliances level. For this goal, we developed and fabricated the electronic circuit of five custom-designed smart plugs (DC, MQTT based) and integrated a vendor-based smart plug (AC) to the system. The devices were equally allocated to three home hubs. As opposed to a standalone SHEMS, this methodology is applied at a cluster scale: through awareness of the electricity consumption of all houses, under certain assumptions optimized load patterns can be generated not only to decrease consumers' electricity bills, but also to meet the grid's constraints. We crafted 3 scenarios as showcases of the methodology performance, with two electricity price plans and different load configurations. The results show both smart plug types were successful in measuring the loads and communication with other layers resulting in decreased electricity cost. Additionally, using a hybrid cloud-fog based architecture, a function was designed for saving the smart plugs records during cloud service or internet disconnection to enable later synchronization of the local and cloud database.

Acknowledgement

This project was relied on the support my supervisor, colleagues, friends, and family. First and foremost, I thank my supervisor for her guidance through the course of developing this project. I immensely appreciate her patience and believe in me during the whole program even when things didn't look well. I am grateful for having her as my teacher and supervisor.

I am thankful for our research team’s post-doctoral fellow Dr. Ramannuni Parakkal Menon for the time he spent with me to advice and guide me through the maze of the issues in my thesis. I also benefited from discussions with Dr. El Bayeh, a post-doctoral fellow at the CERC team.

The research reported in this study was partially supported by the Canada Excellence Research Chairs Program funded by the Tri-Agency Institutional Program Secretariat.

My friends helped me to grow and made this path pleasurable and peaceful. I appreciate the friendship of Armin Kiarash, Peter, and Rachid. I am also thankful for friendship with Hassan Bardareh, his sharp attention and energy emission during all the ups and downs of the program.

My family blessed me with their countless backups during my journey in the program, in a city far from home. It’s unthinkable for me to put my gratefulness in words; I cannot imagine myself overcoming the challenges during my studies without their support and patience. I dedicate my efforts in this research work to my parents and my brother.

Table of Contents

List of Figuers

List of Tables

Chapter 1

1 Introduction

1.1 Concept of Smart Home Energy Management Systems

The advances in miniaturization and efficiency of sensors, processing hardware (Drouin et al., 2017), improvements in communication technologies, and development of various standards, have enabled attractive combinations of features and functions to be used for enhancing human well-being. Smart homes, as a type of ambient intelligence application (Cook et al., 2009) are designed to service the residents in different aspects of life such as security, health, comfort and finance. In the literature, there is no unique categorization for smart homes. Based on the focus of their functions, different categorizations are introduced, but energy efficiency can be seen as one of the main categories (Balta-Ozkan et al., 2014; Kamel & Memari, 2019). Such smart homes are equipped with so-called Smart Home Energy Management Systems (SHEMS).

SHEMS function scope can be categorized in three levels: Smart homes only with energy monitoring ability; smart homes with energy monitoring and controlling capacities; and the most intelligent one, the smart homes with data analyzing and consumption optimization capabilities. When it comes to the third level, SHEMS can have a wide variation of abilities and pursue multiple goals. Although, one major goal is to minimize energy consumption and/or its associated cost without trading off the occupant's comfort.

In this thesis, "smart home energy management systems" or SHEMS, are seen to be construed as the set of SHEMS functions that remotely monitor and control the appliances' loads and are able to communicate with one or multiple cloud servers associated with utilities. In electrical grids, there is a need to balance and adjust the electricity generation with variation of electricity consumption. The common way to keep the balance is to increase electricity generation when demand increases, which leads to higher cost of power generation for the utilities. An alternative to increasing the power generation capacity is benefiting from the potential flexibility in the demand side to shift the peak hours. In the residential sector, grids currently measure the total electricity of a house with a single smart meter that reports the electricity consumption to the utilities, periodically (in future smart grids, there might be multiple metering devices). The price plans are also designed based on the total house consumption. However, to unlock higher efficiency in energy costs, a higher resolution of measurements on the demand side is needed. For the residential sector, this translates to monitoring the houses' consumption at the appliances level.

1.2 Importance of Smart Plugs to SHEMS

Many of the residential appliances built in the past and the present time, do not support energy metering and scheduling their usage. To monitor and control some of the appliances within a house, a smart plug can be used. Energy monitoring smart plugs monitor electricity usage of appliances which is useful in discovering energy consumption patterns for demand response.

In this work, the word “smart plug” refers to a device that is placed between the electricity outlet of a house and plug of an electrical appliance. The word, “smart” emphasizes the ability of the device for having processing power, wireless communication as well as sensing and acting on the environmental characteristics (i.e., current and voltage).

1.3 Goal of Work, Research Question and Contribution

The goal of the work is to create, configure and analyze various hardware and software as part of a smart home energy management system that allows shifting of electrical loads according to different price plans and load scenarios. As a direct consequence, a functional prototype must be available that can be freely programmed and that allows communication between a cluster of such smart home components.

Commercially available smart home components such as smart plugs usually integrate with proprietary software solutions or environments, and therefore data records and control are not fully accessible. Also, some of these components are fully or partially dependent on cloud services—for example, Google Nest and Apple HomeKit—which means that any interruption in the connectivity to the cloud can negatively impact their quality of service. In SHEMS smart plugs continuously measure power consumption and stream their data to the main controller. For most of the commercially available smart plugs, the data is solely stored in a cloud-based time series database for visualization of current consumption or a posteriori analysis. However, during a cloud service or internet outage, the measured data is lost. Due to these problems with commercial smart plugs, flexible hardware was designed and implemented in this work.

To achieve the purpose, a working prototype of a cluster of SHEMS was designed and built, with three individual smart homes each equipped with two energy monitoring smart plugs that represent the electrical loads of multiple appliances in a house. The cluster is monitored and controlled by a server as the central energy management system (CEMS). This helps to better aggregate and monitor the houses power consumption. Two types of smart plugs are used in this prototype: (1) custom-designed smart plugs, where the electronic circuit of six smart plugs, mostly using open-source hard- and software, was designed, and fabricated. The goal is to understand the hard- and software detail, constraints, and potentials, and be able to program or modify them based on a requirement. They were designed to measure direct current (DC) mostly for safety issues and better accessibility, although, with some modifications they can be turned into alternating current (AC) measuring plugs. (2) A commercial smart plug, where the company's initial firmware was flashed and changed to an open-source firmware. This now allows to use this plug as a freely configurable commercial smart plug that supports modifications in the communication configuration and integration with the smart homes' hub. This feature is mostly missing among the available products in the market, including the product used product. Both smart plugs use the MQTT protocol for their communication, and together, show the integration of a commercially manufactured smart plug (AC) and custom-designed smart plugs (DC) in one smart home network. The prototype of cluster of SHEMS, with opensource smart plugs, can be further developed to be integrated with multi-level energy management systems.

In addition, a mechanism was designed to increase the resilience in collection of smart plugs data during cloud service interruptions. The feature is enabled by addition of a fog layer to the conventional cloud-based databases.

Based on the context provided in the previous sections, the main research questions of this thesis are:

- How to design a concept for clustering smart plugs and/or disparate metering devices within as part of a SHEMS?
- How to manage the communication between the smart plugs?

- How to increase data collection resilience? and;
- how to design and implement scenarios to operate the smart home clusters and thus validate the concepts devised?

With that in mind, the thesis is structured in the following manner:

Chapter 2 reviews a number of related works; Chapter 3 discusses methodology developed; Chapter 4 presents detailed information about the smart plugs and homes' hard- and software; Chapter 5 describes the energy management scenarios setup and results; and finally, chapter 6 discusses the results and concludes the work.

Chapter 2

2 Review of the state of the art

The related works' review is divided in 3 parts: the first part briefly compares intrusive and non-intrusive load monitoring. The second part discusses the research works that used fog architecture to increase the smart homes performance (e.g., Quality of Service. QoS). Next, the different designs of smart plugs in literature are compared.

2.1 Intrusive and Non-Intrusive Load Monitoring

To harvest the potential flexibility in appliances' use schedules in the residential sector, higher resolution electricity consumption measurements and control is required; the control through switches enables higher level of automation in managing the energy consumption of residential houses. In appliance load monitoring (ALM), this level of measurement is considered as intrusive load monitoring (ILM) as opposed to non-intrusive load monitoring (NILM), which only measure the total electricity consumption of a house by one meter (Ridi et al., 2014). In NILM, a smart meter measures the current and voltage, or total power consumption value of a house from one single point (meter panel) and tries to find each appliance power consumption by disaggregation of the signal patterns seen on the acquired data. In intrusive load monitoring, there is a metering device for measuring the power consumption of every individual appliance. The cost of installation is lower for NILM as there is only one meter needed per house, and addition of a new device in a house does not need a new meter. However, in recent years, the advancements in chip manufacturing have decreased the cost of embedded systems (Barker et al., 2015) including power sensors and RF modules. Having accurate appliance level time-series consumption data enables more opportunities for efficient power consumption. Ridi et al., 2014 conducted a review on 50 papers, and compared ILM and NILM methods. Briefly with ILM it is possible to:

- detect abnormal energy consumption in appliances.
- understand share of individual appliances in total energy consumption cost.
- evaluate NILM algorithms' performances with ILM results (As a reference).
- detect human activity indirectly by analyzing appliances electricity consumption.
- detect low power device consumptions (not commonly available using NILM)

- measure appliances standby power consumption (not commonly available using NILM)

In this research, along with the goal of finding individual appliance energy consumption, having an actuator (relay) to remotely control each appliance's electricity flow, already determines that we are in the category of intrusive load monitoring.

2.2 Fog Based Smart Homes

Gill et al. (2017) systematically reviewed challenges associated with IoT architecture and identified 9 challenges and 7 related solutions. One architectural challenge reported was quality of service, which has been defined by reliability, availability, and service assurance characteristics. As mentioned earlier, internet disconnection or cloud unavailability can disrupt the functions of smart homes components performance such as smart plugs data storage. In this section works with fog or cloud based smart homes architecture are reviewed.

Chen et al. (2018) identified cloud service outage, latency, cost and security as limiting factors of cloud only architectures. As a solution, they built a smart home prototype with a hybrid fog-cloud architecture to compare its performance with a cloud-only architecture in terms of latency, CPU usage and system load average. The prototype had a hierarchical three-layer fog node within each house. The first layer represents the sensors, each node in the second layer handles the sensor within a room and transfers data packets between sensors and the house's hub which is the third layer. The house hub had a higher computational capacity as it does data collection, and data distribution between sensors and connection to the cloud servers. The results showed that overall, the hybrid cloud only architecture had a lower CPU usage, latency, and workload average.

Rahimi et al. (2020) systematically reviewed 22 papers related to fog based smart homes which were published between 2013 to 2019. Depending on the approaches proposed or studied in the papers, they were classified in two groups: resource management based and service management-based approaches. The papers within each group were also compared with each other in terms of approaches, evaluation types, algorithm types and tools. The resource management-based group consisted of the papers related to fog and cloud integration, load balancing, lowering the processing and response time. The papers that fell into service-based approaches, mainly focused on resilience, quality of service, energy management and security. The evaluation factors used for fog-based smart homes in the reviewed papers were time, cost, energy consumption, security, scalability, and availability. Among them, the availability was the closest factor that could cover the cloud unavailability issues, but, with only 3 papers considering it as an evaluation factor, availability was least considered in the 22 reviewed paper. The authors defined the term availability as "To be accessible anywhere and whenever the user is running a task". Among those 3 papers, only Batalla & Gonciarz (2018) mentioned data storage. They proposed a fog-based architecture that brings smart homes configurations to the edge of the network, using a set-top-box hub which was configured, maintained, and controlled by the operator of the network out of the local area network. Through the remote control of the hub, regular updates and encrypted configuration data back-ups restoring the initial state were among the solutions for enhancing the continuous availability. In general, the test results showed the architecture was successful in enhancing the performance in terms of security and availability aspects. However, looking in more details into the functions of the hub, one function stores sensors data out of the local area network, in the management platform. Therefore, at least this function was dependent to the internet and in times of the internet disconnection the historical data on the cloud would have gaps.

Risteska Stojkoska and Trivodaliev (2017) compared a number of commercial smart homes products and concluded that none of them can be integrated at a neighborhood level due to the fact that high purchase cost prevents large scale implementation of smart homes in all houses in one neighborhood. Moreover, they argued that the high cost is mostly due to absence of interoperability. They proposed a fog-based structure with three hierarchical layers for connection of homes in a neighborhood to microgrid with the focus on optimizing the data transmission and its associated energy consumption. In more details, using predictive filters, data transmission was 95 per cent decreased with keeping the same data accuracy. The framework did not discuss a solution for cloud service outage and its effects on data storage.

Al Faruque and Vatanparvar (2016) noted several challenges in the implementation of smart home energy management on a large scale including: economic justification of SHEMS (due to high price of hardware and software), interoperability of heterogeneous devices, scalability, and adaptability. Following that, they proposed a fog based SHEMS platform with features to address the challenges. The platform consisted of low-cost hardware (cost issues), open-source hardware and software (scalability issue) and running EMS as a service over Device Profile Web Services (Interoperability). In the platform, interaction with cloud servers for data transfer and computing was omitted. The platform was prototyped at two levels: at micro grid level, and at home level. In the microgrid level, a transformer controlled three houses. In each house, the sensors were emulated with one router; another router was used for communication with the transformer router (4 routers used in total). The homes were controlled with an energy management system installed in the transformer level, which monitored the homes energy consumption and controlled them using rule-based algorithms. For example, in case that a house consumption passes a threshold a signal can be sent to the house controller to decrease the consumption. However, the details that how this signal effects the appliances and other power consuming systems was not mentioned. In the other prototype one house hosts and monitors multiple sensor values, but the sensor data storage was not discussed. Also, the prototypes are independent from each other; in other words, it was not mentioned how the signal from the transformer for decreasing the energy consumption, can affect the devices inside a house.

In the literature, several efforts have been done to address the problems that happen during a cloud service interruption. (Doan, 2018), studied the effects of cloud disconnection on common functionalities of smart homes—specifically the ones related to security, safety and well-being of home occupants. They also proposed and implemented RES-Hub which handles some of the cloud-dependent functions, once it detects the connection to the cloud server is down. However, RES-Hub does not specifically address the problem of IoT devices' measured data loss during a cloud disconnection.

Similar to cloud-dependent services, fog-based IoT systems are also unreliable and prone to failures. However, solutions proposed in this context, to some extent, could be helpful to address the problems related to cloud connection failure. Ozeer et al., (2019) designed and deployed a resilient Fog-IoT framework through a smart home case study, in order to mitigate the instability and unreliability of Fog-based IoT systems. The main goal of the framework was to prevent the propagation of failures and avoid the entire restart of the system's configurations by recovering the system to its consistent state. In this case study, multiple mechanisms were defined for restoring service or a device state after a failure using the state data already saved in reliable storage.

However, the solution mentioned does not include the saving of the stream of measurements published by sensors through a failure such as an internet connection.

ASID (inspired by ACID notion in transactional database), to ensure meeting its defined goals. Other research works also proposed mechanisms to prevent confusion between smart homes services and resulting failures but did not propose a solution for storage of data during a cloud disconnection (Cooper & James, 2009; Copie et al., 2013).

Ahsan et al. (2019) believe that fog computing can extend the capabilities of cloud computing in facing the tremendous amount of measured data generated in large smart grids. They proposed a 3-level architecture including smart devices, fog servers and cloud servers. Also, for privacy concerns, in the presented model, the details of each appliance's electricity consumption in house are encrypted using a key and stored and accessed through a cloud server. In the hierarchical model, there were 4 types of communication: device to device, device and fog, fog to fog and cloud to fog. The fog nodes were operated by utilities and were connected to the cloud layer through the internet. The authors, however, did not consider a solution for saving the data when the connection to fog or cloud is interrupted for any reasons.

2.3 Smart Plugs Design

Smart plugs can be categorized based on various criteria:

- Based on number of outlets in one device including single outlet smart plug, double outlet smart plugs, smart strips.
- Based on the range of the current they can measure.
- Based on location of use; within house interior, outdoor, industrial use; this categorization is dependent on the type of certification obtained by the manufacture of their devices; for example, the ones used for outdoor, are better to have waterproof certification.

A variety of smart plugs are manufactured and sold in the market worldwide which are different in appearance, functionalities, and certificates they have earned.

Almost all of them can be controlled with a mobile app to be switched ON and OFF. But they might be different in supporting some features including scheduling, energy monitoring, integration with other IoT devices, integration with other mobile applications, smart houses hub, appliance detection, and phantom power termination. Any feature that can increase integration with other IoT devices or central control systems can make smart plugs contribute to a smarter living ambience.

In terms of geometry, some are installed on the wall and the faceplate of the outlet covers the circuit (Lorek et al., 2015*; Radhakrishnan*, 2018). Lorek et al. (2015) introduced a smart energy meter with inductive current sensing. An inductive coil was designed on a printed circuit board (PCB) which was placed between the two plug's blades (conductors connected to the live and neutral wires). Inductive current sensing is more advantageous in comparison with shunt resistors, as 1. there is no need to break the circuit, and 2. the current does not pass through a series of resistors. Shunt resistors with comparatively high resistance dissipate energy while measuring the current, which is not suitable for high currents (10 A). Since in inductive load monitoring, there is no need to break the circuit, the aspect ratio of the PCB becomes much smaller; lower aspect ratio makes energy meters easier to be planted in the wall~~,~~. Using inductive method for sensing current and Zigbee for communication, the total power consumption of the energy monitor is around 35 mW. But it should be noticed that this slim design is possible because of not including an actuator, as adding a relay breaks the circuit and increases the aspect ratio. The other alternative to inductive current sensing Hall Effect current sensing. Both methods sense the magnetic field produced in the main wires, and hence are considered as non-contact methods (no change in the current path of the main wire). Hall effect is useful for both AC and DC current sensing which can be found in a wide variety of current rating and bandwidth, in compact forms (e.g., board mounted current sensors with galvanic isolation-Figure 1).

Figure 1-Hall effect current sensor-ACS 712[1]

Another wall outlet prototype was developed by See and Jing (2019) with 3 main features:1) A safety mechanism to keep the front cover of the outlet locked was introduced. This mainly aims at prevention of electrocuting that involves the wall outlets. The mechanism uses an electromagnetic lock, which can be controlled using an RFID card. 2) For the current sensing, vampire load and overcurrent detection the hall effect current sensor model ACS 712 was used. 3) it had a buzzer for sound alert. The drawback of this prototype was that it did not have a voltage sensor (assuming the voltage is constant, e.g.,120 V).

Smart plugs, despite their initial simple appearance, can turn complicated when it comes to their comparison, as there are many aspects and possible arrangements among them to be compared such as: their programing and functions, communication and messaging protocols, electronic components, user interface, geometrical characteristics, power consumption, cost, and the level of access to the device design. Generally, in terms of hardware, looking on the papers referenced in Table 1, the prototypes of the smart plugs can be categorized in 3 types:

1. Some papers have proposed AC smart plugs that only measures the AC current and there is no voltage sensor. Therefore, the proposed smart plugs assume the voltage to be constant, e.g., 120V. In such designs as the waveform of the voltage is unknown, they may be practical only for resistive loads with sinusoidal waves but have considerable error for inductive loads as their current and voltage waves are not sinusoidal and their True RMS

---

[1] Source: https://www.allegromicro.com/

value is calculated differently. In addition to this error, the voltage is also not always constant, and has some variations, depending on a number of factors such as distance from power distribution node, the number and type of appliances in use at the same time in a house. It should be noticed however, that the error of the voltage sensors should not exceed the percentage of the voltage variation, as it makes the voltage sensor redundant.

2. Other smart plugs with commercially-off-the-shelves AC voltage and current sensors interface with a development board, such as Arduino, ESP32, ESP8266, etc. The computation for the power consumption is processed by the development board for the microcontroller used for data collection and transfer.
3. The smart plugs with PCB design: these smart plugs use a single-phase/three phase power monitoring integrated circuit (PMIC), along with current and voltage sensor for calculating the values associated with the AC electricity such as active, reactive, and apparent power, RMS values of current (I_{rms}), and voltage (V_{rms}). Table 2 provides information on a number of the PMIC commercially available, along with some example prototypes. Weranga et al., 2013 provided details on functions of PMIC such as calculations, RMS current and voltage, active and reactive power for sinusoidal and non-sinusoidal waveforms, and ADC pins details.

Table 1

References	Measurement Unit		Uses a relay?	Dev Board/ Microprocessor	Wireless Communications	Messaging protocol
	Voltage sensor	Current sensor				
Lorek et al., 2015	Inductive current sensing on PCB	inductive current sensor printed into PCB	No	CC2530	Zigbee	
See & Jian Jing, 2019	N/A	ACS 712	Yes	Node MCU ESP8266 v2	N/A	N/A
Žorić et al., 2020	Included in the PZEM-016		Yes	PZEM-016 + Node MCU ESP8266	Wi-Fi	N/A
Figueiredo et al., 2021	ZMPT101b	ACS 712	Yes	NodeMCU ESP8266	Wi-Fi	MQTT
Le & Kim, 2018	MSP 430 + A signal conditioner		No	ESP32	Wi-Fi	MQTT
Afonso et al., 2015	Resistive voltage divider	AC1015	Yes	Custom edited CC2530 +ADE 7753	ZigBee	
Ahmed et al., 2015	Resistive voltage divider	ACS 712	Yes	ZigBee Microcontroller	ZigBee	
Lee & Yang, 2017	Resistive voltage divider (25Vdc)	ACS712	Yes	Arduino Uno	Wi-Fi	N/A
Radhakrishnan, 2018	N/A	Current Transformer	Yes	STM32F205 +ADE7753	Wi-Fi	N/A
Angrisani et al., 2018	ZMPT101b	ACS 712	Yes	LoPy4	Wi-Fi	MQTT

Table 2- An example of single-phase energy management ICs (PMIC) available in the market

Manufacture	**Single Phase Energy Meter IC**	**Example prototype**
Analog Devices	ADE 77XX: • ADE 7753 • ADE 7758 • ADE 7752	-Anbya et al., 2012 -Guimaraes et al., 2015 -Afonso et al., 2015
	ADE 7953	-Sohaib et al., 2016
	ADE9XXX: • ADE 9000 • ADE 9078 • ADE 9153	---
Allegro Microsystems	ACS 37800	---

2.4 Literature Review Conclusion

As seen in section 2.1, hybrid fog-cloud based architecture were proposed to enhance the performance of cloud based smart home architecture. Among them, some works focused on effects of cloud unavailability on performance of the smart homes in security, safety, and quality of service (e.g., conflict resolution). *None of the research above proposed a solution for preserving the sensor data, more specifically the energy monitoring smart switches connected to SHEMS during a cloud service or internet outage*. In section 2.2, the compared smart plugs were developed with different features to address a specific purpose. *But one common pattern that can be seen in literature, is that smart plugs were mostly developed as standalone devices as opposed to prototyping and operating in groups;* the case studies containing implementation of multiple smart plugs in multiple smart homes, with a model for collection and distributing the messages of the smart plugs were less discussed. Therefore, the contributions of this thesis try to cover the two abovementioned gaps.

Chapter 3

3 Methodology

To analyze energy management strategies for clusters of smart homes, in the first step, physical layer's components such as smart plugs needed to be designed, built, calibrated, and experimentally tested. In the second step, a communication infrastructure between the smart plugs and a server infrastructure had to be designed, configured, and tested. As the third step, algorithms for emulating different energy management strategies needed to be created. Finally, the different scenarios for energy management had to be implemented on the chosen software architecture and the entire system had to be tested, including for service failures. In this chapter, some of the smart plugs' fabrication procedure, criteria and assumptions that were effective on the final form of the work are discussed.

3.1 Smart Plug Fabrication

The selection of the electronic modules depends on an application's requirements. In this thesis the criteria for selection of the components were as follows:

1) Accessibility: using rare components prevents the open-source designs to be scalable.
2) Size: smaller components fit better in the common dimensions of a smart plugs and outlets.
3) Better documentation: Being open-source and having richer documentation helps in troubleshooting, and having faster developments.

Baden et al. (2015) identifies four advantages of using open-source hardware and software:

1. The designs are developed and freely shared by the people who need the application of the final product.

2. Developing the designs requires deeper understanding of the principles related to the equipment; this results in knowing the limitations.

3. Building the equipment can be done in locations with difficult accessibility.

4. The open source connects people around the globe; this also includes the involvement of talented programmers and makers out of the academic and scientific platforms.

The second advantage is beneficial in terms of changing the configuration of the devices whenever an extension, integration, or an update is needed. The third one is helpful for geographical scalability of the systems made by open-source designs.

For choosing the software, protocols, and libraries, in addition to being open source, the lightweight ones in terms of required processing power were put in higher priority as they were supposed to be implemented on edge devices with limited computational power. Examples are preferring MQTT messaging protocol, SQLite database to other alternatives.

For the electronic components such as sensors an evaluation was done to confirm their expected behavior. For example, the current sensor's behavior over a specific current range had to be verified (whether it was linear or not), or if the Analog to Digital Converter (ADC) pins readings of the same version of microcontrollers were not noticeably different. Therefore, the performance of the voltage and current sensors, and every microcontroller used (including the ones with the same version) were validated. Also, for better calibration and accuracy, a recursive test-and-improve methodology was implemented for continuous framework development, validation, troubleshooting, and adjustments until the final design was secured.

In addition, since the abovementioned process and assembling had to be done for six circuits, it increased the chance of mistakes in testing and assembling. Therefore, for safety reasons the smart plugs were designed to measure direct current (DC). It should be noted that with similarities the modifications suggested in section 4.2.2, the DC measuring smart plug can be transformed to an AC measuring plug. For the replaced sensors and microcontrollers, a new calibration was needed. Since each smart plug had its own calibration values, a checklist and version control were highly required to keep track of changes and values of the plugs to avoid confusion.

Regarding the platform, initially, three cloud platforms were explored in terms of offered features (thingspeak.com, thingsboard.com and Node-Red) and were connected to the smart plugs for bidirectional packet transfer. However, later, as the scope of the research became clearer, it was concluded that functions needed to run the cluster of SHEMS did not require any IoT development platform, and to reduce complexities, the python code on a server in the cloud layer handles all the functions.

3.2 Demonstration Scenarios

To demonstrate the performance of the developed cluster of smart homes in various aspects including measurements, message transfer between all layers, data logging, and so on, four scenarios have been defined. The first scenario was only a showcase of resilience in data collection during cloud services interruptions; energy management rules are not involved. The other three scenarios are designed to demonstrate the performance of the entire system in monitoring and controlling the loads and adjusting the consumption pattern considering a price plan with the aim of decreasing the electricity cost. In each of these scenarios there were three houses with a smart home energy management: each smart home hub was connected to two smart plugs.

On the software part, to test and debug the performance of the codes on Raspberry Pis and ESP32s, it was needed to run the scenarios multiple times. For this process multiple codes had to run in coordination. For large number of tests, this demanded a large amount of time and increased the chance of mistakes. To alleviate this some checklist and procedures were designed to automize and simplify the testing process, which proved to be effective and reduced occasional mistakes.

3.3 Assumptions

IoT infrastructure have various aspects, which mostly are crucial for a real-world deployment (security, privacy, etc.) However, in this study to better focus on one aspect of the system, we had to make assumptions for the other parts. For example, according to Gill et al. (2017) security is the top reported challenge in IoT literature. As the IoT applications are expanding with a high rate, the potential of lack of security in causing significant disasters is increasing. Hence, IoT devices and services has become attractive targets for cybercriminals. Smart Homes Energy Management Systems as a category of the IoT applications are no exception and are exposed to the same treats. In some cases, the breach in house IoT devices not only can impose a danger to the safety of the house and its occupants, but also can endanger the grid or resources of utilities. Large implementation of the IoT devices comes with a risk, if a kind of vulnerability remains hidden and spreads with the devices. Since the cyberattacks have a wide spectrum of complexity, identifying and fixing the cyber vulnerabilities should be done by cyber security experts. Therefore, we assume cyber defense and immunization was already provided to the system.

Another assumption was that for monitoring and controlling the appliances by a cloud server, the consumer already agreed with a certain level of appliance control and storage of data in a cloud server associated with a trusted party. Other technical assumptions are discussed in Chapter 4.

Chapter 4

4 Hardware and Software Configuration

In this chapter, first, a brief overview on the IoT architecture used is give in section 4.1, and in the next sections, detail of components' hardware and software and their connection is explained.

4.1 IoT Architecture

The network topology and the configuration of devices and protocols and their integration is defined through an IoT architecture used for data collection, transport, storage, and process in a consistent way to satisfy an application's requirements. For the design of the architecture, the main question was, "how can measurement data bidirectionally be transferred between a smart plug and cloud servers". Next, more detailed questions were added around the first question based on the requirements of the applications: "Which messaging protocol is proper for multiple smart homes and smart plugs?" and "How to save data of smart plugs, when the internet is disconnected, for later database synchronization?". Based on the requirements, the solution gradually changed to its final configuration.

4.1.1 Hybrid Cloud-Fog Architecture

The architecture used in this research includes a fog layer and a cloud layer. The cloud layer includes three cloud servers: the first server aggregates and stores data and provides required data to the second; the second server is responsible for analyzing the data, constructing decisions, and sending them back to the smart homes. The third server is an MQTT broker. Although, by installing a local broker on the second server, the same function could be achieved, a server in the cloud is used to simulate the real communication through internet. In the fog layer, a Raspberry Pi (RPi) assumes a smart home hub's role. The main function of the RPi is to collect and store the smart plugs measurements, and to transfer and distribute the signals from central energy management system (CEMS) back to the smart plugs. Figure 2 illustrates the hybrid cloud-fog architecture for one house. For example, one smart plug and the RPi, can find and transfer packets with each other using the Wi-Fi router in the house. Using the same router, the RPi can communicate with the server in the cloud.

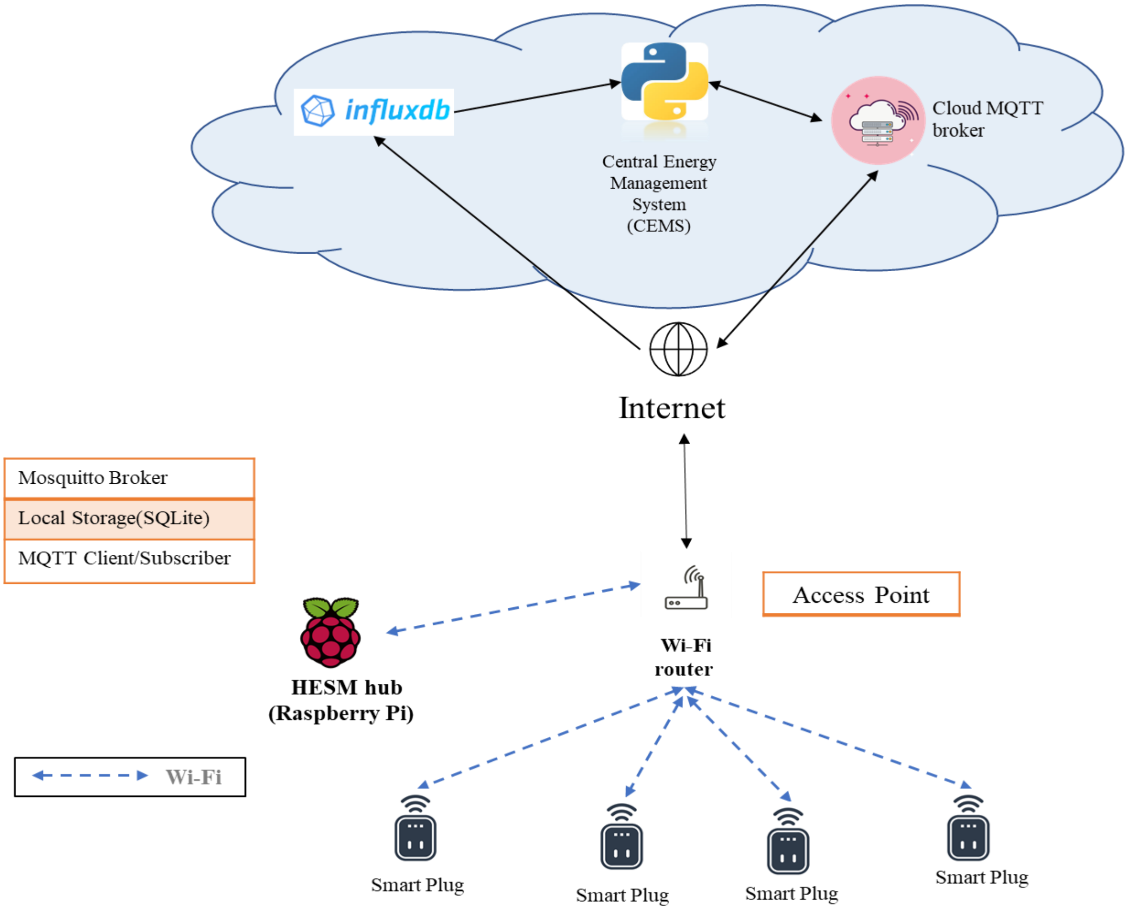


Figure 2-A schematic diagram showing connection of different components and the data transfer path between different components.

4.1.2 The Cloud Layer Services

4.1.2.1 The IoT Cloud Platform. Recently the number of IoT cloud platforms (both open-source and closed-source) has increased, giving a wide spectrum of options to be selected based on required features. For instance, Node-RED is an open-source, browser-based flow editor built on Node.js. It is a light weight platform suitable to be installed on edge devices (Node-RED, n.d.). The selection of the IoT cloud platforms depends on the specifics of an application. In this project, in the cloud layer, the required functions could be executed only using python scripts and libraries. Hence, there was no need for an additional software to control, which prevents redundant complexity.

4.1.2.2 MQTT Brokers. One of the main functions of the MQTT broker is to transfer data packets from publishers (machines that send packet with specific topics) to the subscribers (machine that listen to only specific topics). MQTT brokers is a software that can either be installed on a local computer or a cloud server. As mentioned before Raspberry Pis have a local MQTT broker. For CEMS, a cloud MQTT broker was used for communication of the central energy management system (CEMS) with the smart homes' hub. Figure 3 shows the details of the configuration and the links between the brokers, clients, and subscribers.

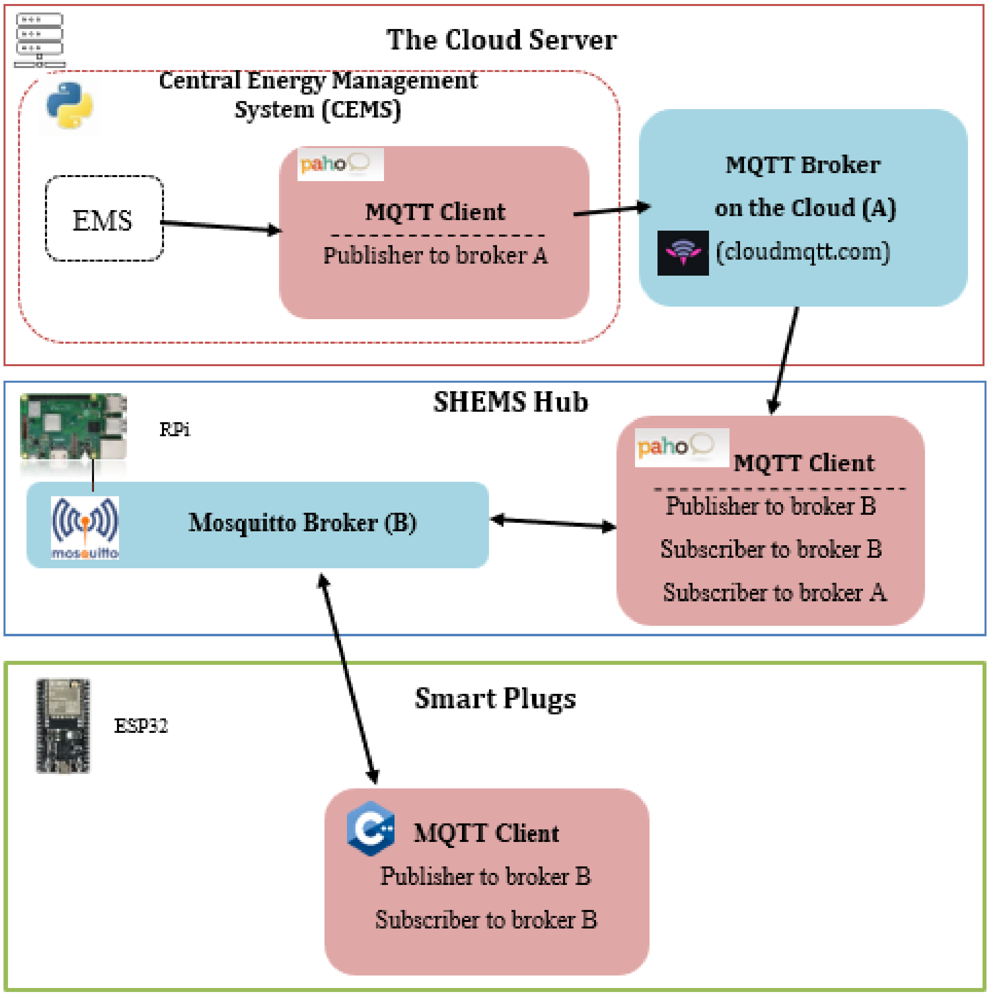


Figure 3-A schematic diagram of MQTT clients and brokers connection

4.1.3 The Wireless Communication

In IoT applications, the wireless communication protocol is one of the configurations that is highly dependent on the specifics of that application. Numerous wireless communication protocol reviews can be found in the literature and over the internet, which can assist designers select the most compatible protocol to their criteria. For instance, Yokotani (2016) compared wired and wireless communication technologies including BLE, ZigBee and Wi-Fi which are common for smart applications. As almost all the houses that have access to the internet use a Wi-Fi router, we used Wi-Fi to decrease the required devices to install the system and their associated cost. ESP32 development boards and Raspberry Pi 4 which are used in this work, both support Wi-Fi protocol with their built-in Wi-Fi module. As smart plugs they take the power of the circuit from the outlet, there is no battery-imposed energy constraint.

4.1.4 The IP Addresses

As shown in Figure 2, the router within each house acts as an access point and uses Ipv4 for communication between all the devices. A static IP is allocated to each Raspberry Pi by the router, and this static IP is hardcoded to the smart plugs of each house which enables them to find and connect to the MQTT broker within the Raspberry Pi.

4.2 Hardware Configuration

This section describes the specifications of smart plugs electronic components, calibration, and solutions to challenges related to electronic parts.

ESP32 is used as the microcontroller unit of the smart plug. It is a well-known microcontroller used for various IoT, mobile, and wearable applications which outperforms many same class MCUs in terms of packed features and cost. The high integration of features, such as Wi-Fi, Bluetooth, power management modules and filters makes it suitable for applications with tight spatial limitations. ESP32 can be programmed with the Arduino IDE which is a simple but practical IDE with a large evolving community. Both ESP32 and Arduino, have user-created libraries that are helpful for many applications (for example MQTT Pub-Sub-Client library written by Nick O'Leary).

The board has 18 analogous to digital converter pins (ADC). The ADC pins are categorized in ADC 1 and ADC 2; since ADC 2 can't be used when the Wi-Fi module is activated, only 6 ADC pins (ADC1) remain for applications which need Wi-Fi. The ADC pins in ESP32 have 12-bit resolution and 3.3 V capacity. This means the ADC's 0 to 3.3 V input voltage is mapped between 0 and 4095. In case that the circuit is made correctly, without any additional components such as 0.1 μF capacitor, the variations in ADC pin were mostly around ±15 Least Significant Bit (LSB) different from the values read by an oscilloscope.

It is noteworthy that ESP32 development boards such as Devkit V1, and Devkit V4, has a step-down voltage regulator (AMS 1117-3.3), which reduces voltage down to 3.3 V with 1.1 V typical voltage dropout. This means that if the input voltage of ESP32 development board decreases below 4.4 V, ESP32 ADCs would give wrong results.

In this work, every ESP32 ADC used has been calibrated using an oscilloscope and a programmable DC power supply with 0.1 V steps. The results are shown in one graph in Figure 4. As it can be seen for the input voltage of more than 2.5 V the ADC reading becomes non-linear.

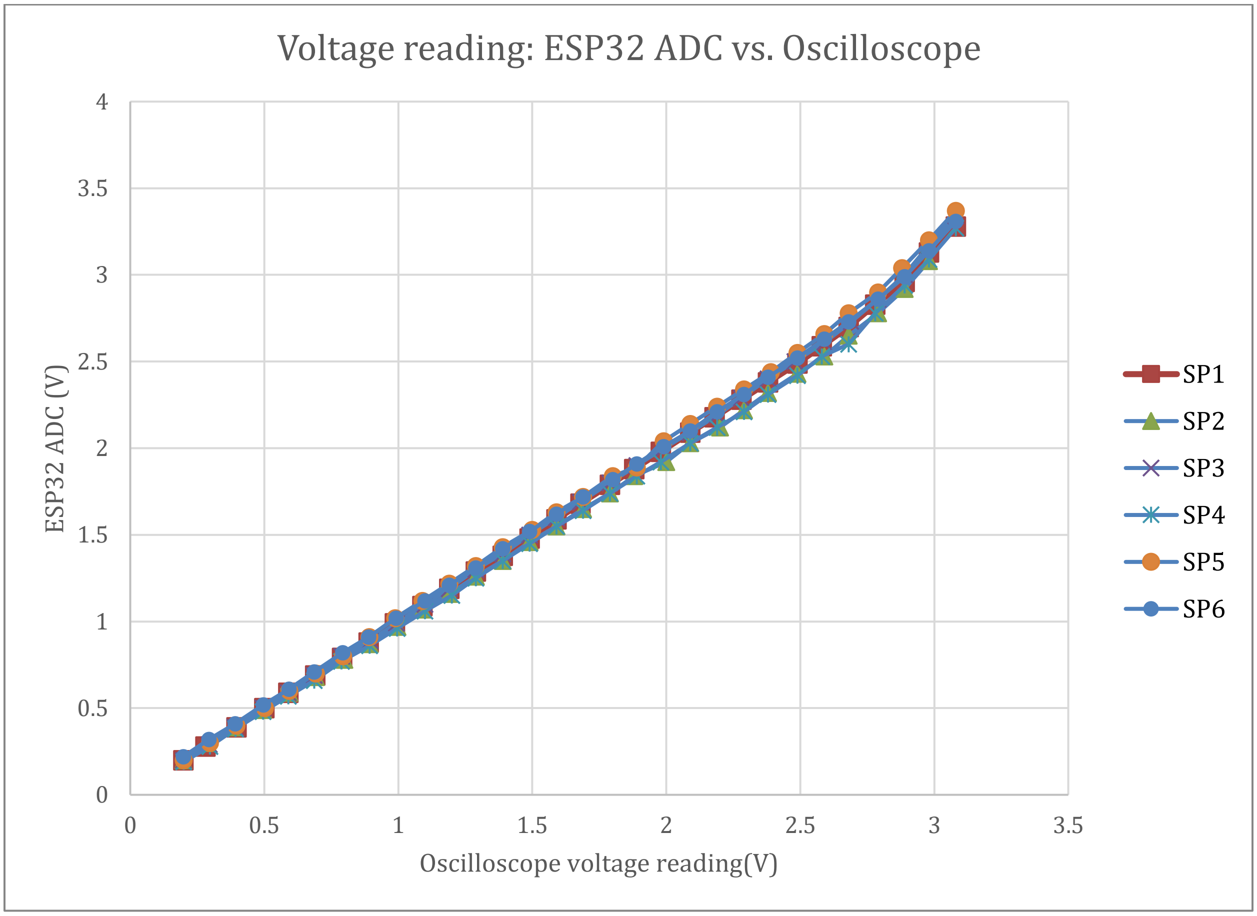


Figure 4-Voltage reading of ESP32 ADC vs. the oscilloscope

Therefore, we only used 150 mV to 2500 mV range of the ADC. Using a linear regression, the drifts from the reference values (achieved by oscilloscope) were adjusted in the programming.

4.2.1 Current Sensor

The sensor used in this work is ACS 712-5A, a hall-effect based linear ratiometric current sensor, that comes in 3 versions: 5 A, 20 A, and 30 A. The successor version of the ACS 712, which is

ACS 723 series is available in the market. ACS723 IC series have approximately the same price as ACS 712, but module-based versions are custom developed by other companies and are more expensive than ACS 712. As ACS 723's better sensitivity is not effective on our work in compared to its higher price, ACS 712-5A is preferred. The 5 A version has a sensitivity of 0.185V/A and the sensor converts -5 A to 5 A to output voltage of around 1.57 V and 3.42 V (Figure 5).

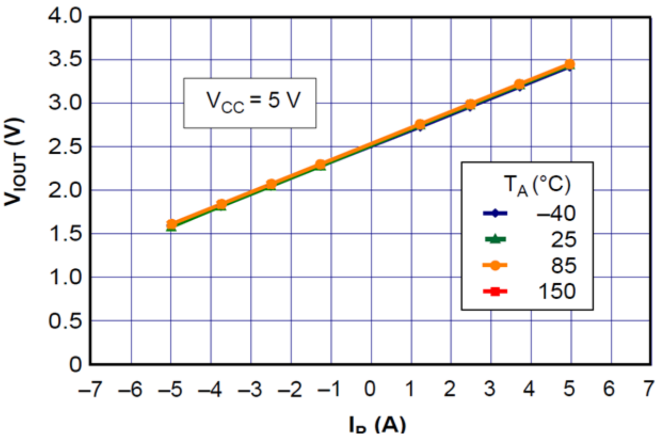


Figure 5- ACS 712 voltage output vs input current (datasheet)

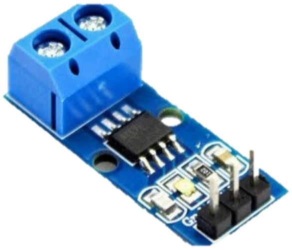


Figure 6- ACS-712-5 A

The graph above, shows changes of the sensor's output voltage against the input current, which is linear and changes in temperature has no noticeable effect on the sensor's performance. To validate the important specifications of the sensor we conducted a calibration with an oscilloscope (Rigol DS1102E-Figure 8) and a programmable DC power supply (Korad KD3005D) for every current sensor used in this research. The results were recorded for 0.1 A steps, up to 2A.

The datasheet of the current sensor IC mentions that the noise of the sensor is 21 mV peak to peak. However, for the module shown in the Figure 6, the noise observed with oscilloscope, shows around 40 mV noise peak to peak. With 0.185 V/A sensitivity this translate into ~0.2 mA peak to peak variations in the measured current.

The other challenge arises since ACS 712 is a ratiometric sensor. In other words, the sensor characteristics such as quiescent voltage (output voltage when there is no input current—usually Vcc/2) and sensor's sensitivity, change with supply voltage of the sensor. Therefore, the ripples and voltage drop in supply voltage affects the sensor readings. Initially, the smart plugs were prototyped in breadboards and jumper wires, and noticeable voltage drop and variations in the current and voltage readings was observed. After a range of trial and error and adjustments on the circuit, the problem mitigated with some modification in hardware:

- Soldering the components on two sided solderable breadboard
- Changing wires to AWG 22
- Adding electrical components (resistors and transistors) to prevent current draw from the microcontroller
- Reversing the current input directions; in this way, the output voltage of the current sensor decreases as the input current increases, and the 150 mV to 2500 mV range of ESP's ADC is not exceeded (Figure 25)
- Changing the initial DC power supply to a more powerful and accurate one

For calibration of current sensors, each module's the input current was read with the power supply and the output voltage was monitored by the oscilloscope. The results are shown in Figure 7.

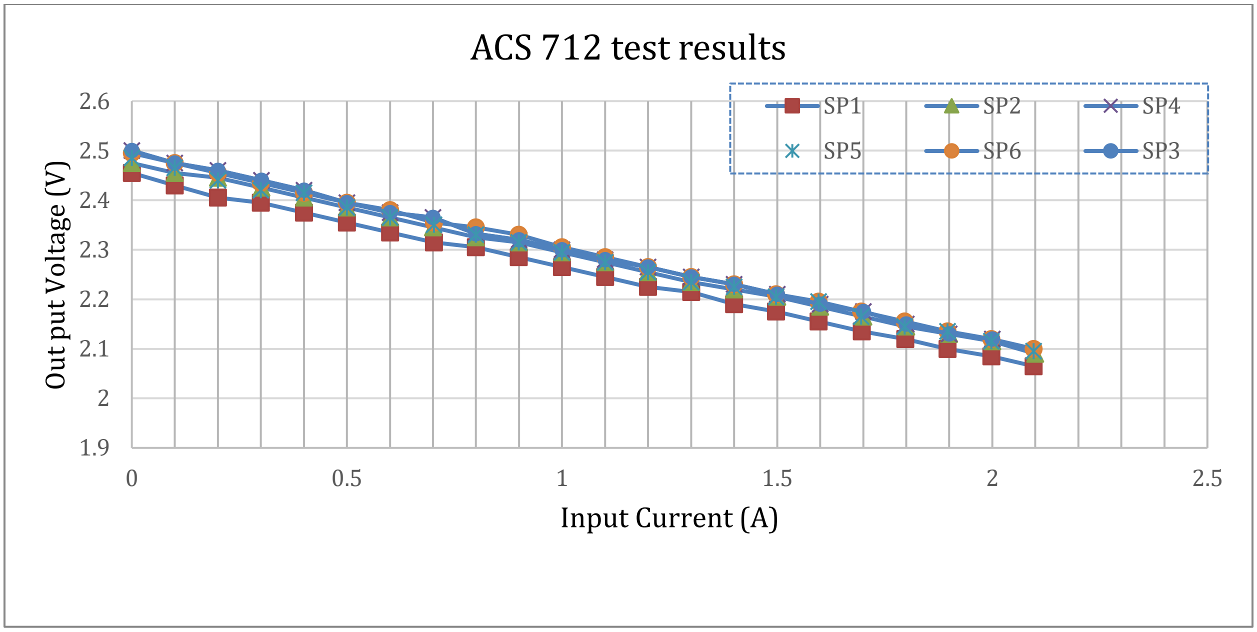


Figure 7-Current sensor test results for all smart plugs

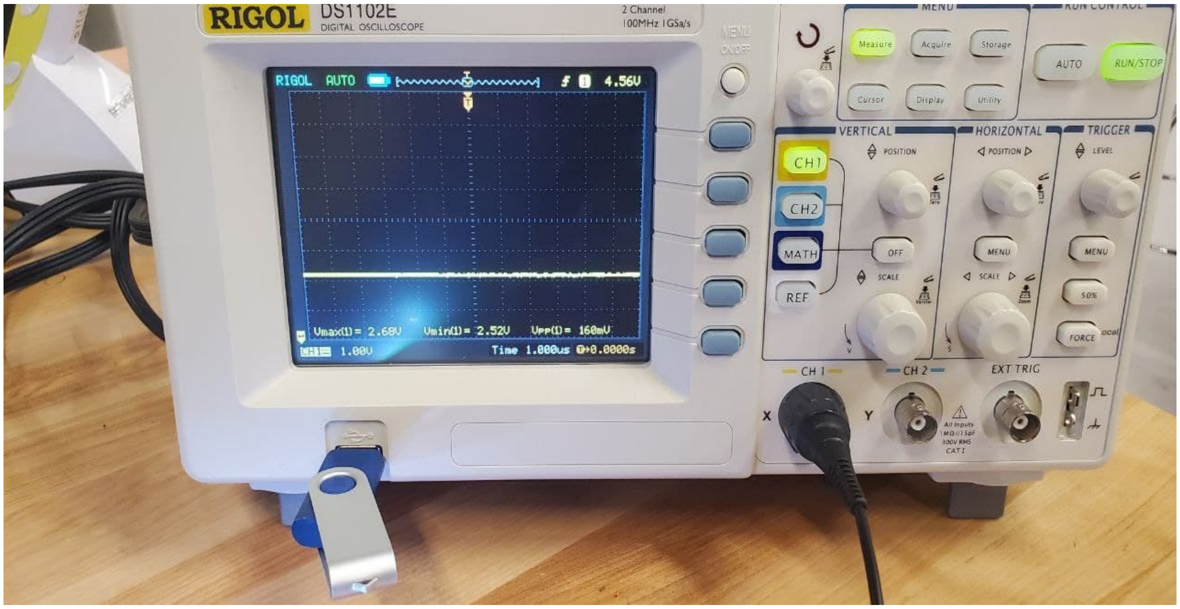


Figure 8-The image shows the oscilloscope used for the calibration of the current sensors.

4.2.2 Voltage Sensor

The DC voltage sensor (Figure 10 and Figure 11) is basically a voltage divider made by two resistors. As same as the current sensor, a calibration was done for the voltage sensor. To validate the division ratio of the DC voltage sensor with its nominal range, a calibration with a power supply and an oscilloscope was conducted in the same way done for the current sensor. As shown in the Figure 9-Voltage Sensor Calibration, the voltage sensor has linear pattern with a slope of around.

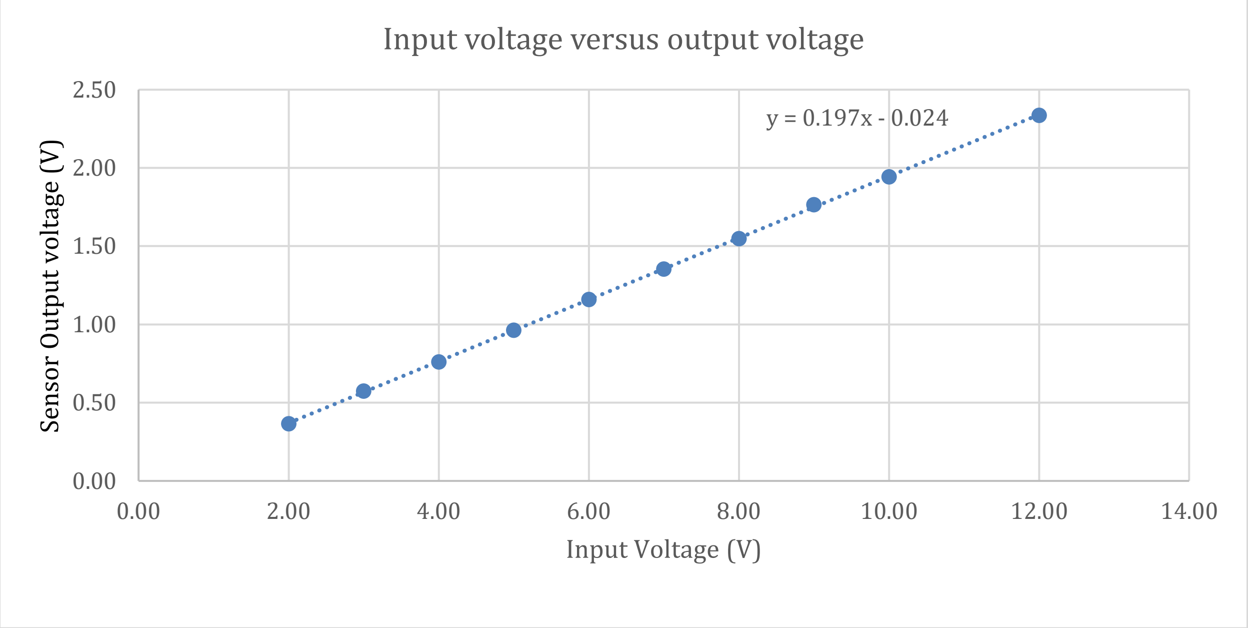


Figure 9-Voltage Sensor Calibration

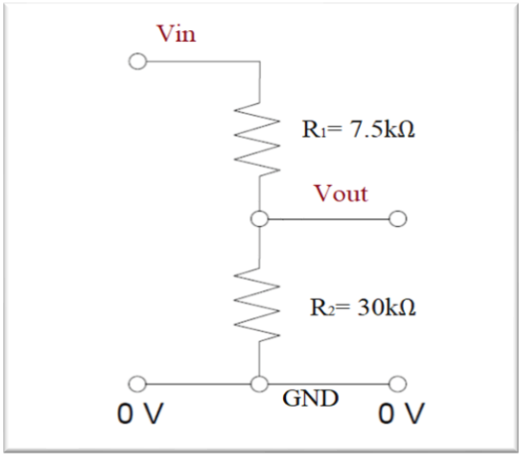


Figure 10- DC voltage sensor or voltage divider

Figure 11

DC Voltage sensor

Figure 12

AC voltage sensor

4.2.3 The DC smart plug's connections

For the connections of the smart plugs USB Type A is used. For connection of the load to the power supply a male USB port is used. As LEDs also use male USB type A output, a female USB port is used to connect the lamp to the DC smart plug. Connecting the power supply to the circuit and the load through USB makes the overall look of the circuit and connections more organized. Figure 14 show the male and female USB type A used.

Figure 13-Female USB (typeA)

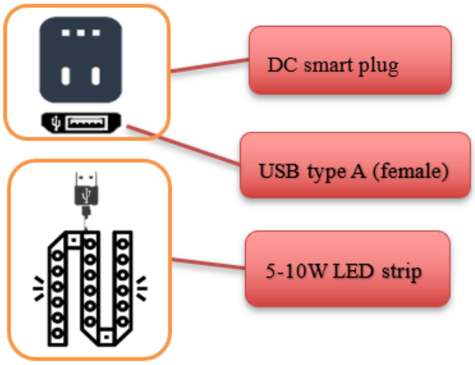


Figure 14-smart plug connections

4.2.4 Possible Adjustments to Transform the DC Plug to AC Plug

Table 3 compares the developed DC smart plug and the possible alternative components for transforming the plug into an AC smart plug. Regarding voltage sensor, ZMPT101b module with maximum 250 Vac replaces the DC voltage sensor (i.e., voltage divider). The current sensor can be the same, as it is bidirectional with 80 kHz sampling frequency. However, there is a wide range of current sensors in the market with different characteristics, which brings more flexibility to the designs. For microcontroller, ESP32 or the similar alternatives, depending on the library used and the functions uploaded may not be enough to handle all calculations. In such case, a single-phase power management integrated circuit (PMIC) such as ADE 7753, can decrease the computational load on microcontroller (refer to Table 2).

Table 3-suggested components to transform the DC into AC smart plug

Component	DC	AC
Voltage Sensor	Resistive Voltage Divider (25Vdc)	ZMPT101b AC Voltage Transformer Module
Current Sensor	ACS 712 series	ACS 712 series
Processor/ Microcontroller	ESP32	ESP32 (+ PMIC)
Power Supply	N/A	Addition of a voltage converter and a fuse

4.2.5 The Final Design

Figure 15 shows the assembled DC measuring smart plugs and illustrates the detail of their circuit. Since a couple of smart plugs burnt and replaced, we kept one smart plug as a spare part for immediate replacement with the damaged one.

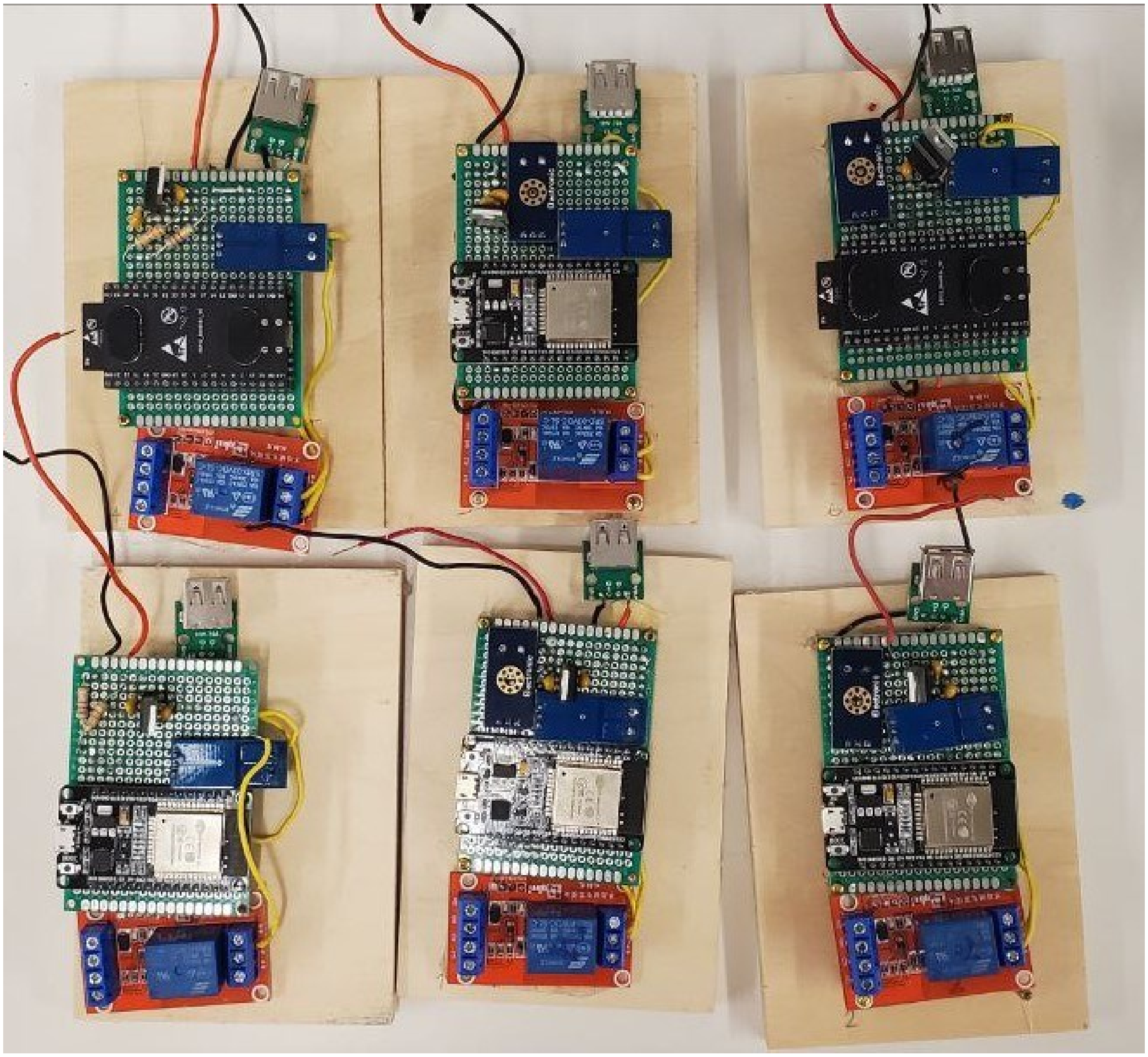


Figure 15-Final Design of DC measuring smart plugs

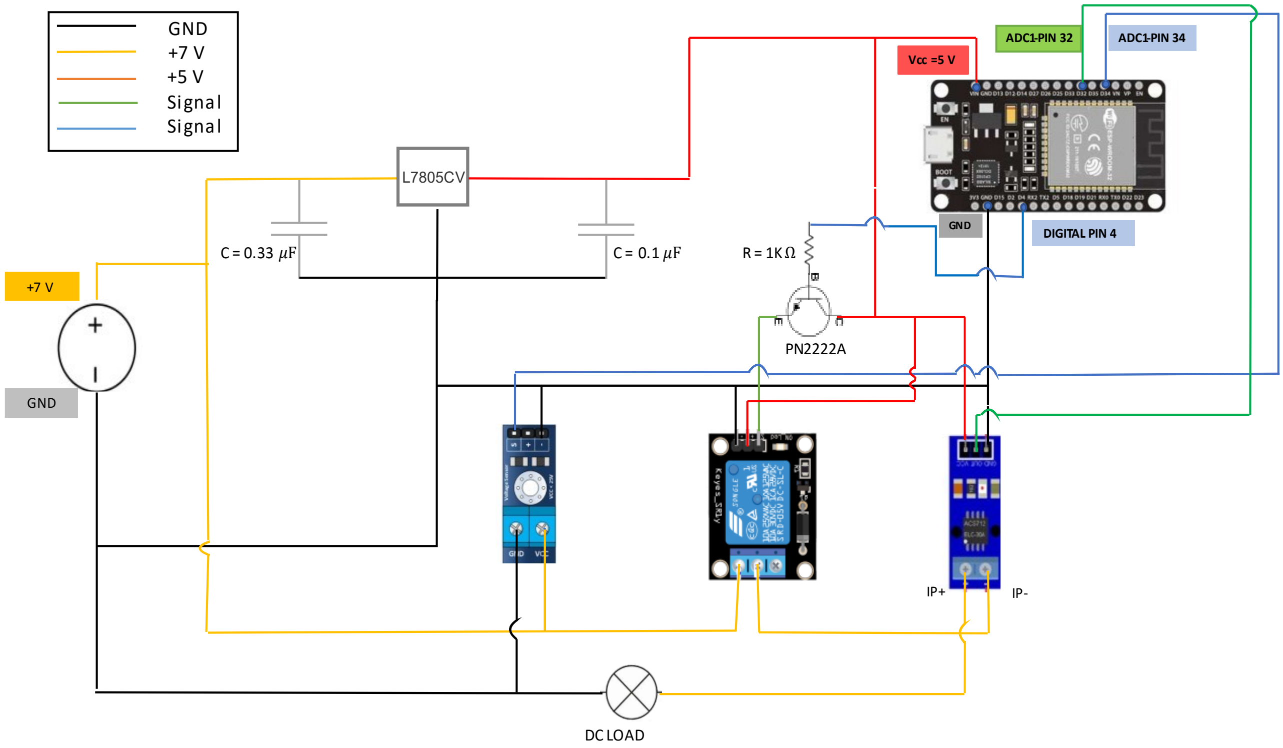


Figure 16-Schematic of the custom-designed smart plug circuit

4.3 Software Configuration

This section provides information on the integration of messaging protocols in all layers. Also, the configuration of local and cloud databases and the formation of data packets is explained.

4.3.1 Messaging Protocol

In this architecture, MQTT is used as the messaging protocol. MQTT was initially invented by IBM and later standardized by one of the world's non-profit standards organization, OASIS; it describes MQTT as follows:

> "MQTT is a client server publish/subscribe messaging transport protocol. It is lightweight, open, simple, and designed to be easy to implement. These characteristics make it ideal for use in many situations, including constrained environments such as for communication in Machine to Machine (M2M) and Internet of Things (IoT) contexts where a small code footprint is required and/or network bandwidth is at a premium."

MQTT allows for topic-based publications and subscription within a pub/sub pattern. Compared to HTTP, features of MQTT allow it to be faster, and more efficient in network bandwidth, processing power and hence less battery usage for resource constrained devices (Yokotani, 2016).

Raspberry Pi as the smart home's hub uses the Eclipse Mosquitto as MQTT broker. Eclipse Mosquitto is a lightweight open-source broker which can be installed on machines with high processing powers or embedded devices with limited resources. According to Eclipse Foundation, Mosquito can have 1000 clients connected with 3MB of RAM. The broker also has the bridge feature which allows it to be connected to other MQTT brokers, enabling a structured network of MQTT brokers (*Eclipse Mosquitto | Projects.Eclipse.Org*, n.d.). In this project, MQTT brokers are installed at both fog layer and cloud layer.

Paho, another project from Eclipse foundation, is a library in Python which provides a client class for applications. Using it, applications can create clients that connect to MQTT brokers to publish and subscribe to topics, for sending and receiving messages. Paho library is installed on Raspberry Pi.

4.3.2 Databases

4.3.2.1 Local Database. SQLite is a developer-friendly, well tested, transactional, lightweight and reliable database, which make it an attractive option to be used in embedded devices (*SQLite Home Page*, n.d.). In Raspberry Pi's python script, connection with an instance of SQLite database is made by "sqlite3" python library.

4.3.2.2 Cloud Database (InfluxDB). InfluxDB is an open-source database specifically designed for time series data. One of the main features of InfluxDB is its wide range of supported libraries and plug-ins which makes integration with other software or platforms easier.

4.3.3 Payload Structure

The data points are initially generated in smart plugs (edge layer) in JSON format. The message generated in the smart plug contains the following attributes:

HouseID, DeviceID, Current, Voltage, and *Relay_State* which are arranged and sent in the following **JSON** format**:**

{“HouseID”: ***“house ID number”*****, “DeviceID”:** ***“device ID number”*****, “Current”:** ***“current value”*****, “Voltage”:** ***“voltage value”*****, “Switch”:** ***“relay state”*****}**

DeviceID*:* DeviceID is unique for each device in a house and is hardcoded within them.

HouseID*:* it is provided by the administrator of the system and is unique for every consumer. Therefore, when users register to the system, a HouseID is allocated to them, which can associate their smart plugs to that house. The assumption is that the registration of the owners and the associated smart plugs are done using a mobile application. Since programming a mobile application for the system is out of the scope of this thesis, the DeviceID, HouseID, and their relations are created manually.

Current & Voltage: the current and voltage values which are measured by current and voltage sensors on both types of smart plugs (custom designed and commercial).

Timestamp*:* Once the MQTT messages are received by the python client in Raspberry Pi (fog layer), it appends a UTC timestamp to the message.

Switch*:* shows the state of the relay.

NFL*:* the attribute is a flag (Network Flag) that shows the status of the connection to the cloud database. During the time that the data is being recorded to the database, A) if the SHEMS hub can connect to the cloud database, then NFL is set to “0”. B) if it cannot connect to the cloud database for any reason, NFL is set to “1”. NFL is appended to the message after timestamping.

NFL is used for database synchronization during internet or cloud services disconnections which are explained in more detail in section 4.4. With exponential increase in data generated in large scale IoT applications, conventional fully dependent cloud services face several issues including network traffic, response time, security, and privacy. Doan (2018) identifies the reasons that a cloud service can become unavailable and the subsequent dangers it imposes on the security and safety of the house. The severity of the impact of these issues varies based on the focus of smart homes applications and the importance of their functions. For example, in smart homes with focus on health, an internet disconnection can endanger the health of occupants. Another example is when a smart home is supposed to activate an alarm (fire alarm, intrusion detection alarm) and due to disconnection from the controller in the cloud, fails to alert the house owners. The purpose of fog computing, which is an extension and complementary to cloud services, is to enhance the performance of architectures that cloud servers and their services have an essential role in.

4.3.4 Commercial Smart Plug Integration

Among the IoT devices sold in the market, a few companies offer smart bulbs and plugs that are cloud free. Since such IoT devices weren't accessible at the time of the research, to represent the integration of a commercial cloud free device to the SHEMS, we selected a Tuya[2]-based smart plug with dependence to the cloud and replaced the firmware with Tasmota, which is a common open-source firmware for ESP32 and ESP82XX. The firmware change was done via over-the-air (OTA) method, using the code developed by "Vtrust.com". The advantage of OTA flashing is that there is no need to open the case of the device.

In the next step, the MQTT topics of Tasmota were adjusted for the smart plug and added to the smart home's python script, allowing to control the relay and read the Power Monitoring IC data (active power, voltage, power factor). Table 4 shows the 3 main Tasmota MQTT topics used for this project. The period for data upload is set 10 seconds (maximum frequency available).

Table 4-Tasmota MQTT topic definition

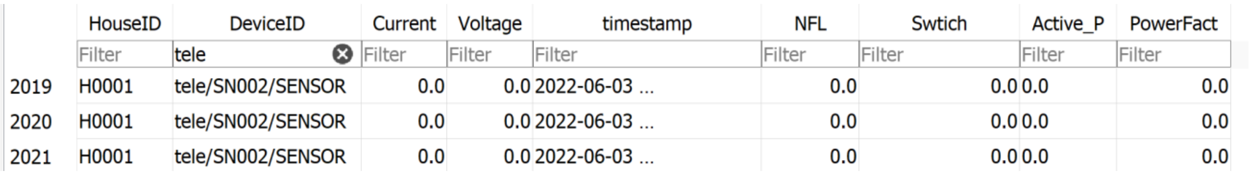
Function	Publish topic	Payload	Subscription topic
Turn the relay ON	cmnd/%deviceID%/POWER	"ON" or 1	cmnd/%devicename%/STATE
Turn the relay OFF	cmnd/%deviceID%/POWER	"OFF or 0	cmnd/%devicename%/STATE
Report Telemetry data	N/A	N/A	Tele/%devicename%/SENSOR

---

[2] Tuya is a commercial firmware used by reportedly large number of many IoT devices under different brands.

Figure 17-The commercial smart plug integrated to the SHEMS

By parsing the JSON file, the current, voltage, and power are retrieved and saved in the databases. Figure 18, shows the final database schema.



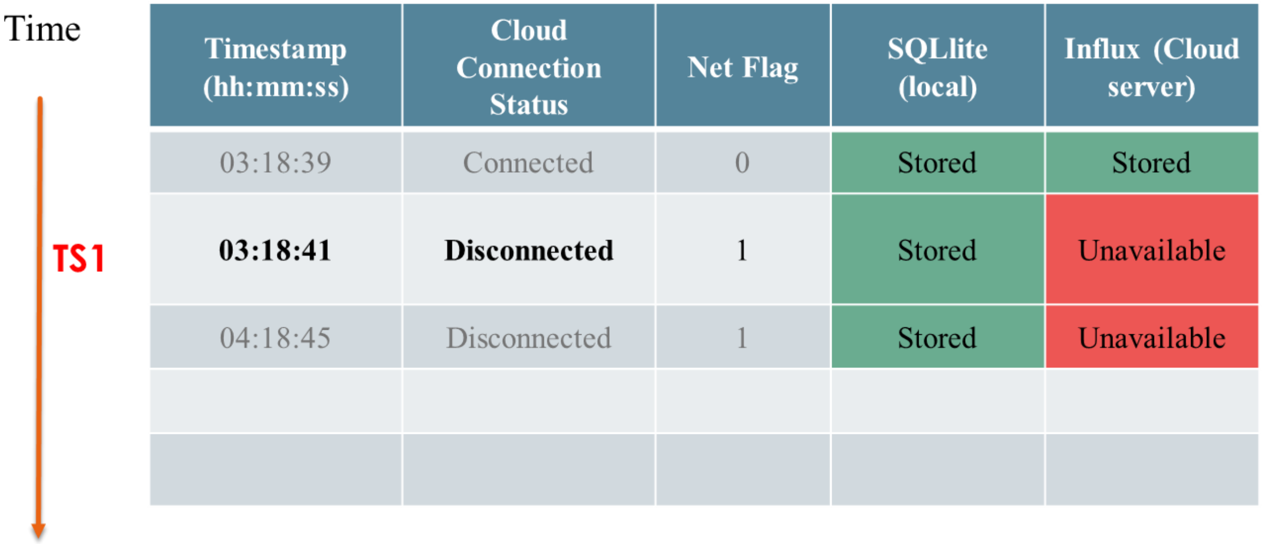
	HouseID	DeviceID	Current	Voltage	timestamp	NFL	Swtich	Active_P	PowerFact
	Filter	tele	Filter	Filter	Filter	Filter	Filter	Filter	Filter
2019	H0001	tele/SN002/SENSOR	0.0	0.0	2022-06-03 ...	0.0	0.0	0.0	0.0
2020	H0001	tele/SN002/SENSOR	0.0	0.0	2022-06-03 ...	0.0	0.0	0.0	0.0
2021	H0001	tele/SN002/SENSOR	0.0	0.0	2022-06-03 ...	0.0	0.0	0.0	0.0

Figure 18-a screenshot of SQLite schema

4.4 Resilience Against Cloud Unavailability

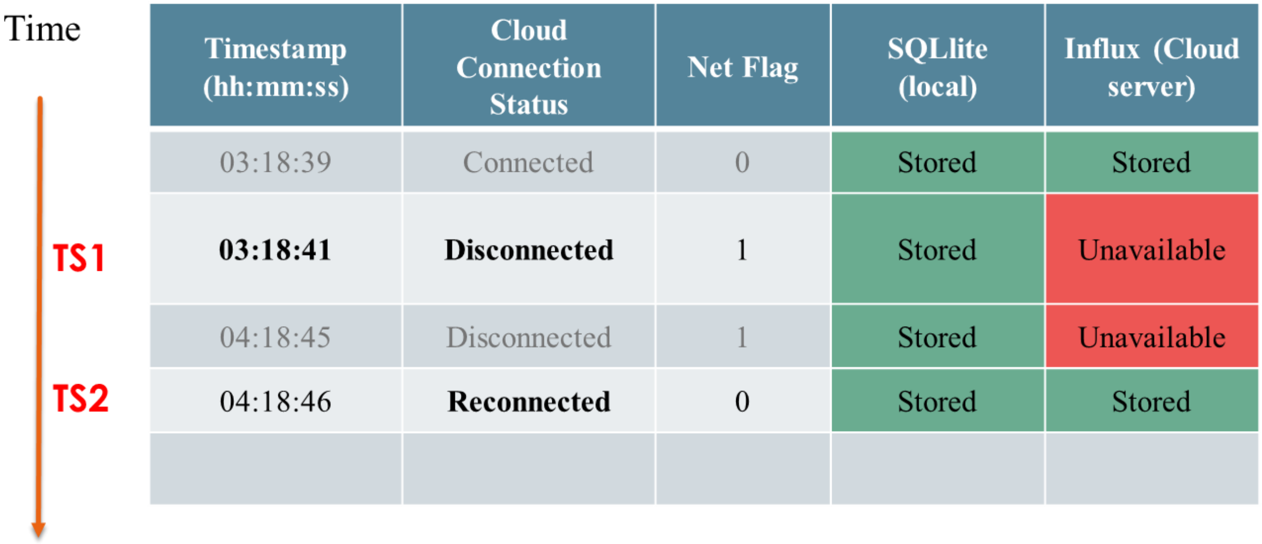
The goal for developing this feature is to increase the resilience of smart homes and central energy management systems in collecting smart plugs data when cloud servers are unavailable for a period of time. This feature saves the sensors' measurements during an interruption of cloud services, and then synchronizes the cloud databases once the connection is re-established. In this work, this feature is specifically used for smart plugs. However, with minor adjustments, it can be extended for other types of sensors.

Once Raspberry Pi receives a message, as mentioned before, it gets timestamped and then is stored in the local database. Next, RPi tries to upload the message to the cloud server. At this stage, if the cloud server is unavailable for any reason, the code records that timestamp (TS1) and raises a flag (NFL = 1), indicating the cloud server is unavailable, and this is shown on all the data recorded during the disconnection (Figure 19). The system keeps recording the messages until the connection to the cloud database is established. Once the connection is back (Figure 20), it records the timestamp of the reconnection. In the next step, it takes a query from the local database for all the data that were recorded during TS1 to TS2. Since the result is multiple data points, the results are turned into a Pandas data frame. Finally, they are batch uploaded to the InfluxDB cloud. The results are shown and discussed in section 5.2.



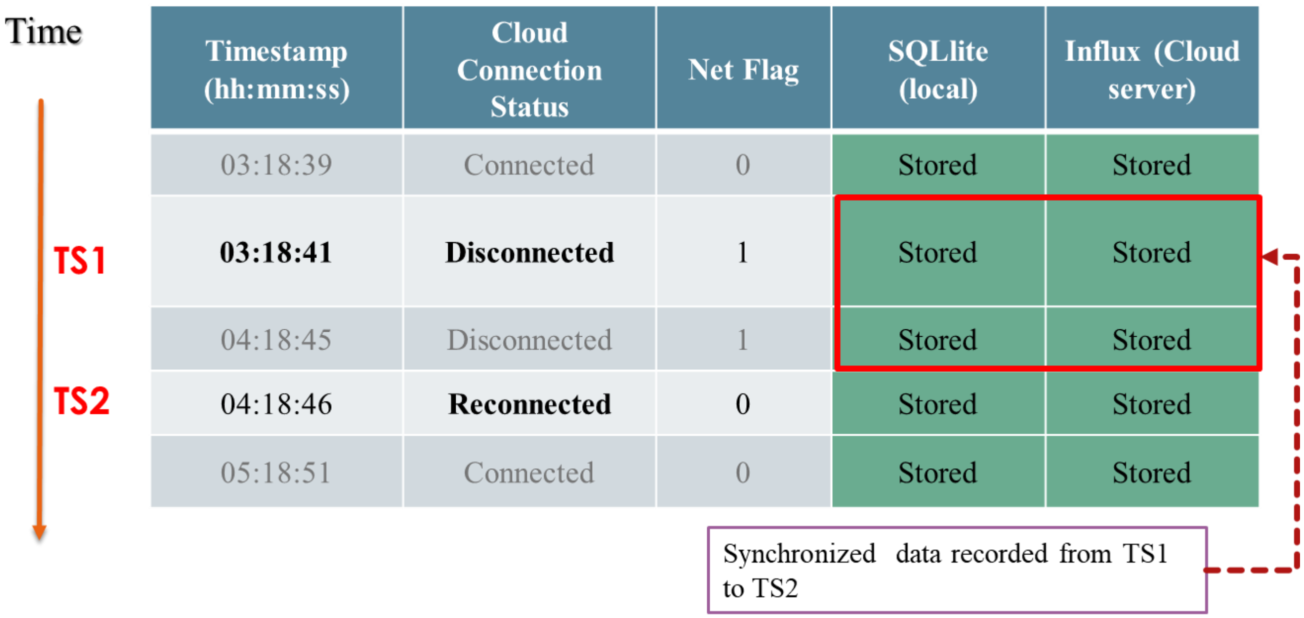
Timestamp (hh:mm:ss)	Cloud Connection Status	Net Flag	SQLlite (local)	Influx (Cloud server)
03:18:39	Connected	0	Stored	Stored
03:18:41	**Disconnected**	1	Stored	Unavailable
04:18:45	Disconnected	1	Stored	Unavailable

Figure 19-The diagram shows the timeline of internet and databases status. After TS1, internet was disconnected, and data was only stored in the local database. The "timestamp" column is just to better clarify the chronological order of the internet disconnection and reconnection.



Timestamp (hh:mm:ss)	Cloud Connection Status	Net Flag	SQLlite (local)	Influx (Cloud server)
03:18:39	Connected	0	Stored	Stored
03:18:41	**Disconnected**	1	Stored	Unavailable
04:18:45	Disconnected	1	Stored	Unavailable
04:18:46	**Reconnected**	0	Stored	Stored

Figure 20(a)-At TS2, internet is reconnected. The diagram shows the moment before synchronization of the databases.



Timestamp (hh:mm:ss)	Cloud Connection Status	Net Flag	SQLlite (local)	Influx (Cloud server)
03:18:39	Connected	0	Stored	Stored
03:18:41	**Disconnected**	1	Stored	Stored
04:18:45	Disconnected	1	Stored	Stored
04:18:46	**Reconnected**	0	Stored	Stored
05:18:51	Connected	0	Stored	Stored

Synchronized data recorded from TS1 to TS2

Figure 20(b)-The diagram shows the moment after the synchronization of the databases.

4.5 Load Modeling

IoT devices have various interacting aspects or parts such as user interface, authentication, etc. To represent the parts that were required for the smart homes to work, but were not in the research focus, and to make the system's performance demonstration more feasible and clearer, some **assumptions and representation** were made:

1. For limitation in fabrication of custom-designed smart plug fabrication, each smart plug represents a group of appliances, having the role of a smart meter (excluding scenario 4–full detail is given in chapter 5).
2. The appliance loads are modeled by LED strips (Figure 22) and LED bars (Figure 21). The reason for choosing LED bars over a single LED is to draw enough current from the power supply, to be detectable by the current sensor. Both LEDs are powered by USB type A, which is compatible with the smart plug's connection.
3. Turning ON or OFF a plug is equivalent to turning ON or OFF an appliance. In other words, the user interface of the system is synchronized with the smart plug. This assumption allows automatic load shifting.

4. The plugs are already registered to the smart home hubs and the central system, therefore their information such as the associated house, and their ID is known to the system.
5. Every hour is scaled down to 30 seconds, as a result 1 day is represented by 12 minutes (This is very helpful in faster troubleshooting as well).

Since each plug represents multiple appliances loads and one plug is not able to show power consumption variation in multiple appliances, The following assumptions and methods are designed to simulate the effect of multiple appliances loads:

1. As a fact, we know that the power consumption of each LED fluctuates and is not a constant value. Fluctuations are between approximately 4.5 W to 5.5 W for LED strips (connected to SN001; Figure 22-LED Strip) and 2.3 W to 2.8 W for LED bars (connected to SN002;Figure 21-LED bar). In the model and optimization calculations, the values were averaged out, and the LED strip was assigned as consuming 5 W and the LED bar as 2.5 W.
2. Smart plug #1 (SN001) power consumption was represented by integer values ranging from 0 to 5 kW, as the appliances power consumption rates mostly are not continuous. For instance, to demonstrate 2 kW consumption in one time slot, the switch is ON only for a proportion of the time slot, which is calculated by the following formula:

   $$\text{The duration of the switch being ON} = \left(\frac{\text{power consumption}}{\text{Maximum power consumption}}\right) \times (\text{time block in seconds})$$

   $$\left(\frac{2\ \text{kW}}{5\ \text{kW}}\right) \times (30\ \text{s}) = 12\ \text{s}$$

   Therefore, the lamp remains on for 12 seconds to represent 2 kW power consumption.
3. For smartplug #2, the minimum power consumption was 2.5 kW and represented with binary values. Table 5 presents a summary of the load model and scales.

Table 5-Load model used with smart plugs

Load Characteristic	Appliance group #1 (SN001)	Appliance group #2 (SN002)
Minimum discrete power consumption (kW)	1 kW	2.5 kW
Smart plug power representation scale [3]	1:1000	1:1000
Hourly CEMS payload	An integer value between 0 to 5 $P = \{0,1,2,3,4,5\}$	Binary values "1" for 2.5 kW, and "0" for 0 kW

Every hour CEMS takes the decisions that which smart plug turns ON or OFF and then sends the decisions to each smart home. The payload for each smart home is an array which contains integer values for each smart plug.

To emulate the household appliances use, the non-deferrable appliances are controlled by CEMS remotely instead of someone manually turning ON or OFF the appliances. Therefore, for Non-deferrable loads, relay signals are sent by the CEMS as well (Table 6).

As mentioned in chapter 3 this is done under the assumption that the consumer agreed to a certain level of load control by a utility. Table 6 shows an example of how the packets are hourly sent to smart homes.

---

[3] 1 Watt consumed by a LED represents 1 kW consumed by appliances

Table 6-Topic tree formation and

Smart Home ID	Plug ID	MQTT Topic Tree	Payload in the 1st hour	Payload in the 2nd hour	Payload in the 24th hour	Payload array after 24 hours
H0001	SN001	H0001/SP/SN001	$\begin{bmatrix}0\\0\end{bmatrix}$	$\begin{bmatrix}1\\0\end{bmatrix}$	$\begin{bmatrix}5\\1\end{bmatrix}$	[0, 1,…,5]
H0001	SN002	H0001/SP/SN002				[0, 0,…,1]
H0002	SN001	H0002/SP/SN001	$\begin{bmatrix}1\\1\end{bmatrix}$	$\begin{bmatrix}1\\0\end{bmatrix}$	$\begin{bmatrix}1\\1\end{bmatrix}$	[1, 1,…,1]
H0002	SN002	H0002/SP/SN002				[1, 0,…,1]
K	SN001	K/SP/SN001	$\begin{bmatrix}1\\ \cdots\\ \cdots\\ P_j{}^{4} = 0\end{bmatrix}$	$\begin{bmatrix}1\\ \cdots\\ \cdots\\ P_j = 0\end{bmatrix}$	$\begin{bmatrix}4\\ \cdots\\ \cdots\\ P_j = 1\end{bmatrix}$	[1,1,…,4]
K	J	**K**/SP/**J**				[0,0,…,1]

Figure 23, shows the custom designed and commercial smart plugs with their assigned ID and LED.

Examples for the appliances with deferrable loads are dishwashers, washing machines, or any type of appliance that its use can be shifted during a day. Non deferrable loads are the appliances that are randomly or always used, and their schedule cannot be changed or predicted such as a television, monitors, refrigerators.

---

[4] Pj is the payload of the smart plug number j

Figure 21-LED bar

Figure 22-LED strip

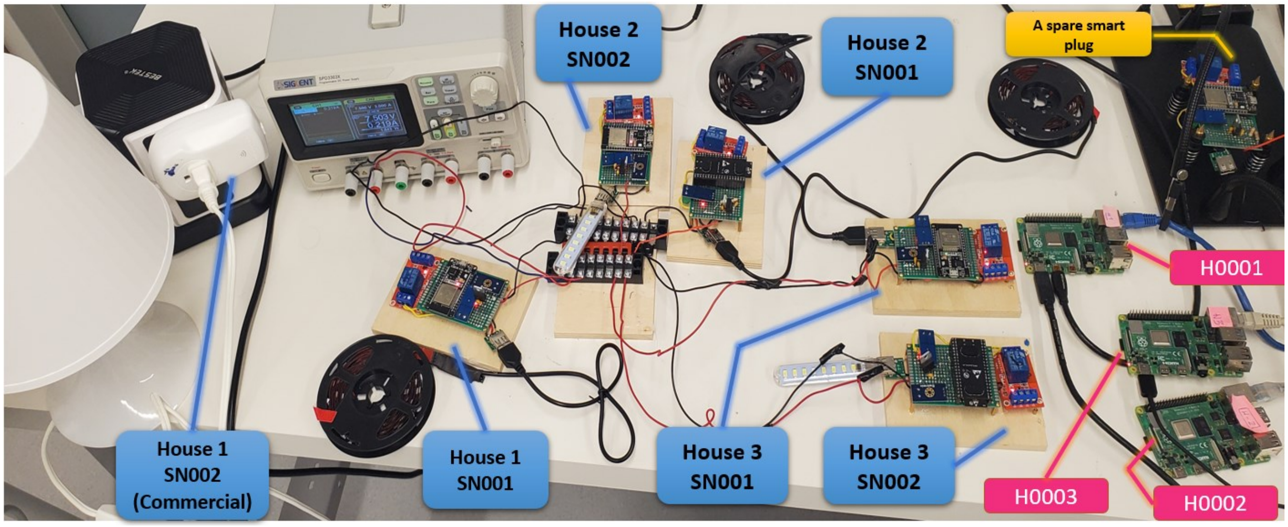


Figure 23-A picture showing smart plugs, and Raspberry Pis, and a power supply

4.6 Energy Management System on the Cloud

The energy management functions are defined in two python scripts running on a server in the cloud. The first script is an optimization algorithm that is already solved in python using GLPK and Gurobi as the solver. The algorithm takes the initial hourly power consumption schedule and minimizes the daily electricity cost as the objective function. In this thesis, the input to the optimization algorithm was generated randomly. However, this can be later replaced by using extracted using pattern recognition algorithms and historical power consumption. The output of the optimization script plugs into the second script's–the second script's role is to distribute the signals related to each plug to its associated smart plug.

Chapter 5

5 Energy Management Scenario Definition and Results

Four scenarios have been defined to demonstrate the performance of the smart homes system. The first scenario is only a showcase of resilience in data collection during cloud services interruptions and energy management rules are not involved. The other three scenarios are designed to demonstrate the performance of the entire system in communicating data packets, monitoring, and controlling the loads and adjusting the consumption pattern based on a price plan with the goal of decreasing the electricity cost. In each of these scenarios there are three houses with smart home systems; each smart home hub is connected to two smart plugs. The smart plugs measure the load of one or multiple *emulated appliances* loads that are either deferrable or non-deferrable (Figure 24 and Figure 25).

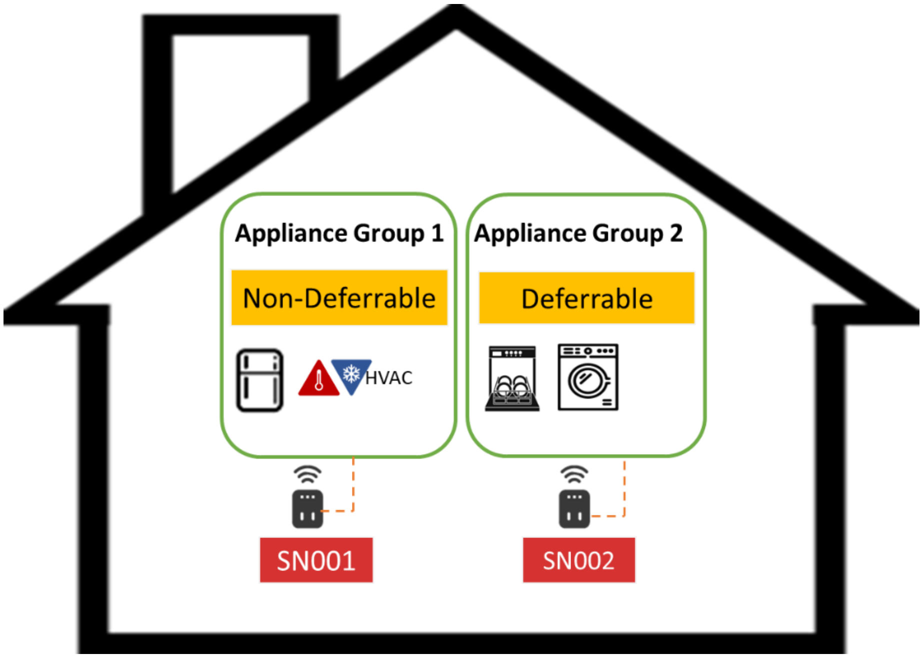


Figure 24-An individual house load detail

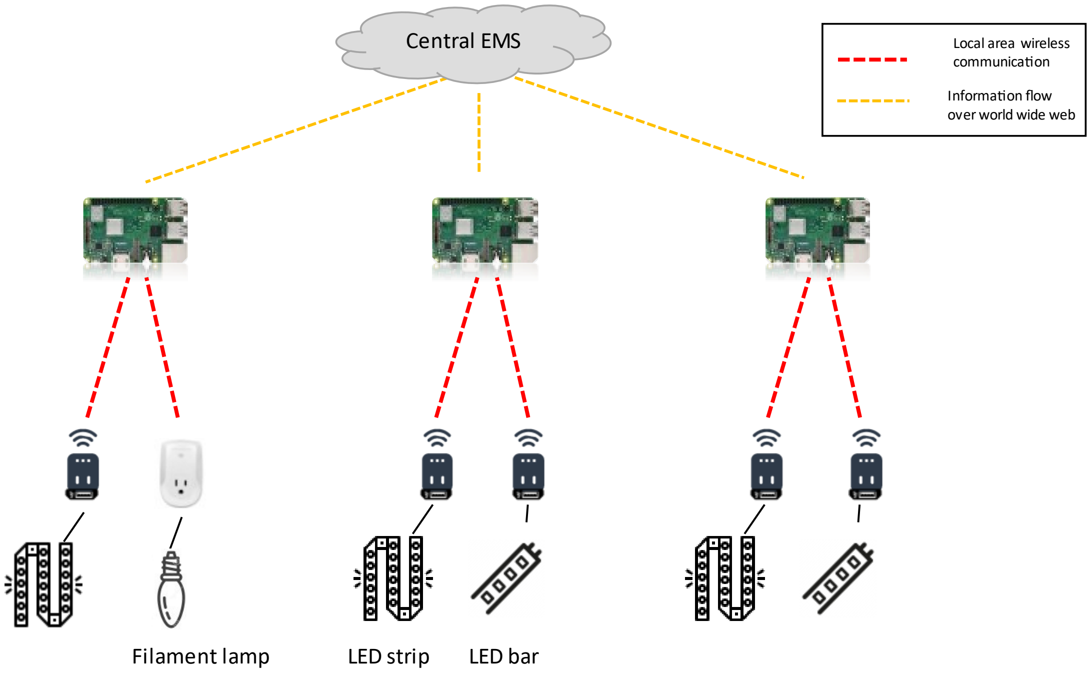


Figure 25-Schematic of smart homes clusters architecture

5.1 Price Plans

In demand response programs, utilities offer dynamic price plans to encourage customers to participate in the reduction of electricity use. Dynamic pricing is one of the factors that increase the flexibility on the demand side and enables more control over the electricity bill of the consumers. In time-of-use (ToU), the electricity price varies with hours, weekdays, or seasons. The electricity price rate is determined based on the price of the electricity generation for the corresponding times (*Electricity Pricing and Costs*, n.d.). During peaks in electricity demand, electricity has a higher price, to discourage consumers for consuming electricity in the peak hours. For example, Hydro-Quebec, in its Rate Flex D pricing, informs its members a day before that the electricity price rises to 50.650 ¢/kWh in the peak hours. In the off-peak hours, however, the electricity is billed less than the Rate D (base rate). Table 7 shows the summary of Hydro Quebec Rate Flex D (*Rate Flex D | Hydro-Québec*, n.d.).

Table 7-Hydro-Quebec Rate Flex D detail

Item	Summer period:April 1–Nov. 30	Winter period:Dec. 1–March 31
System access charge for each day in the consumption period	41.168¢	41.168¢
Price applicable to energy consumed up to 40 kWh times the number of days in the consumption period (1st tier)	6.159 ¢/kWh	4.336 ¢/kWh
Price applicable to remaining energy consumed (2nd tier)	9.502 ¢/kWh	7.456 ¢/kWh
Price applicable to energy consumed during peak demand events	N/A	50.650¢/kWh

In Scenario 2 a tiered plan and in scenarios 3 and 4, time-of-use pricing is used to demonstrate the ability of system (CEMS) in working with different price plans (Table 8). As mentioned in section4.6, the input of the optimization script was generated randomly. This pattern (energy management rules) was sent to smart plugs and the power consumption was measured. The "Reference Scenario" refers to these values. Reference scenario values was compared to Scenarios 2, 3 and 4 for demonstration of the systems' performance.

Table 8-Summary of Scenarios 1 to 4

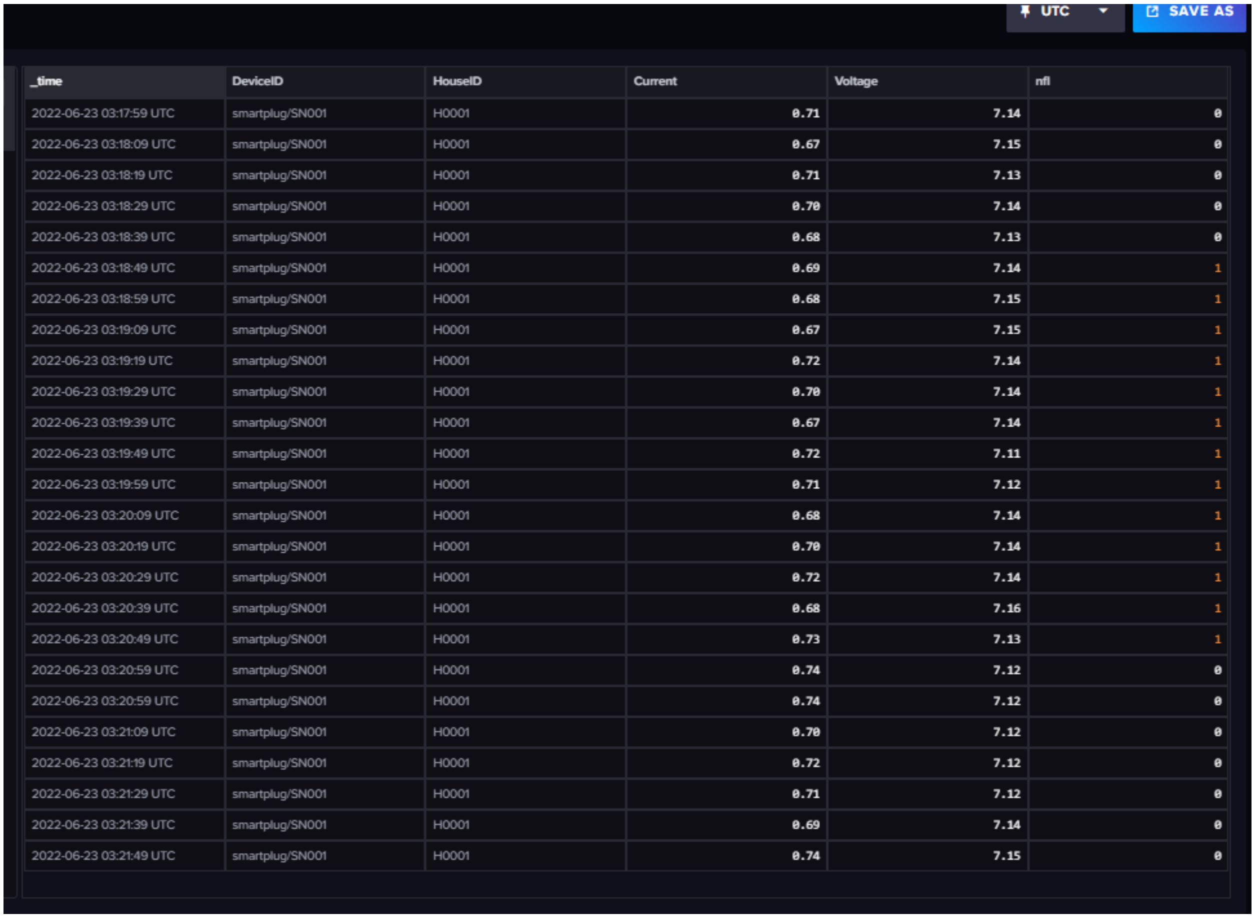
Scenario	Plugs with Fixed Loads	Plugs with Variables Loads	Device Representation	Price Plan	Purpose
Scenario 1	N/A	N/A	N/A	N/A	Demonstration of the resilience in data collection against internet disconnection
Reference Scenario	N/A	N/A	group of appliances in Scenario 2 and 3 A single appliance in Scenario 4	Tiered in Scenario 2 ToU in Scenario 3 and 4	• The input of the optimization script. • The load patterns for all smart plugs are randomly generated and measured. This acts as a reference to compare energy consumption • The random load pattern is constant for all three scenarios.
Scenario 2	SN001	SN002	A group of appliances	Hydro Quebec Flat Rate D (Tiered)	Showcase of the ability of the system in controlling the loads and working with two different price plans
Scenario 3	SN001	SN002	A group of appliances	High day price Low night price (Time-of-use)	
Scenario 4	----	SN001 and SN002	A single appliance	High day price Low night price (Time-of-use)	

5.2 Scenario 1

The purpose of the first scenario is to solely demonstrate the performance of the smart home hub in resilient data collection from smart plugs during a cloud server unavailability (explained in section4.4). One smart home equipped with 2 smart plugs was assumed. The system was tested for approximately 4 minutes, in which for 2 minutes the internet was disconnected. The data during the 2 minutes were stored locally for later synchronization with a InfluxDB cloud. Table 9 shows the detail of the Scenario 1 demonstration.

Table 9-Scenario1 demonstration details

Duration	House ID	Internet status	SN001 and SN002 relay status
~1 min	H0001	Connected	ON
~2 min		Disconnected	ON
~1 min		Connected	ON



_time	DeviceID	HouseID	Current	Voltage	nfl
2022-06-23 03:17:59 UTC	smartplug/SN001	H0001	0.71	7.14	0
2022-06-23 03:18:09 UTC	smartplug/SN001	H0001	0.67	7.15	0
2022-06-23 03:18:19 UTC	smartplug/SN001	H0001	0.71	7.13	0
2022-06-23 03:18:29 UTC	smartplug/SN001	H0001	0.70	7.14	0
2022-06-23 03:18:39 UTC	smartplug/SN001	H0001	0.68	7.13	0
2022-06-23 03:18:49 UTC	smartplug/SN001	H0001	0.69	7.14	1
2022-06-23 03:18:59 UTC	smartplug/SN001	H0001	0.68	7.15	1
2022-06-23 03:19:09 UTC	smartplug/SN001	H0001	0.67	7.15	1
2022-06-23 03:19:19 UTC	smartplug/SN001	H0001	0.72	7.14	1
2022-06-23 03:19:29 UTC	smartplug/SN001	H0001	0.70	7.14	1
2022-06-23 03:19:39 UTC	smartplug/SN001	H0001	0.67	7.14	1
2022-06-23 03:19:49 UTC	smartplug/SN001	H0001	0.72	7.11	1
2022-06-23 03:19:59 UTC	smartplug/SN001	H0001	0.71	7.12	1
2022-06-23 03:20:09 UTC	smartplug/SN001	H0001	0.68	7.14	1
2022-06-23 03:20:19 UTC	smartplug/SN001	H0001	0.70	7.14	1
2022-06-23 03:20:29 UTC	smartplug/SN001	H0001	0.72	7.14	1
2022-06-23 03:20:39 UTC	smartplug/SN001	H0001	0.68	7.16	1
2022-06-23 03:20:49 UTC	smartplug/SN001	H0001	0.73	7.13	1
2022-06-23 03:20:59 UTC	smartplug/SN001	H0001	0.74	7.12	0
2022-06-23 03:20:59 UTC	smartplug/SN001	H0001	0.74	7.12	0
2022-06-23 03:21:09 UTC	smartplug/SN001	H0001	0.70	7.12	0
2022-06-23 03:21:19 UTC	smartplug/SN001	H0001	0.72	7.12	0
2022-06-23 03:21:29 UTC	smartplug/SN001	H0001	0.71	7.12	0
2022-06-23 03:21:39 UTC	smartplug/SN001	H0001	0.69	7.14	0
2022-06-23 03:21:49 UTC	smartplug/SN001	H0001	0.74	7.15	0

Figure 26-A screenshot of InfluxDB cloud user interface, showing uploaded records for SN001 during the internet disconnection.

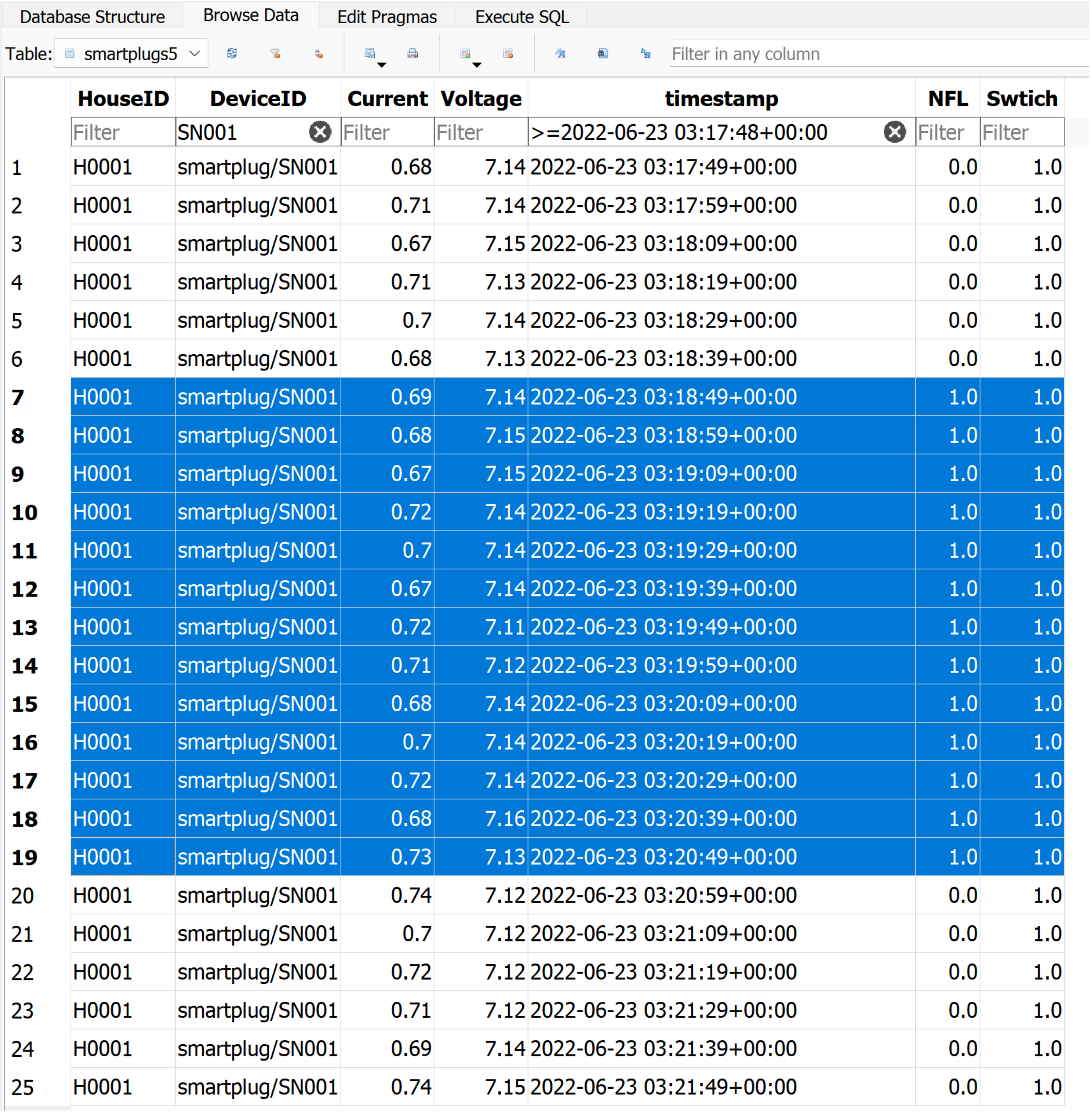
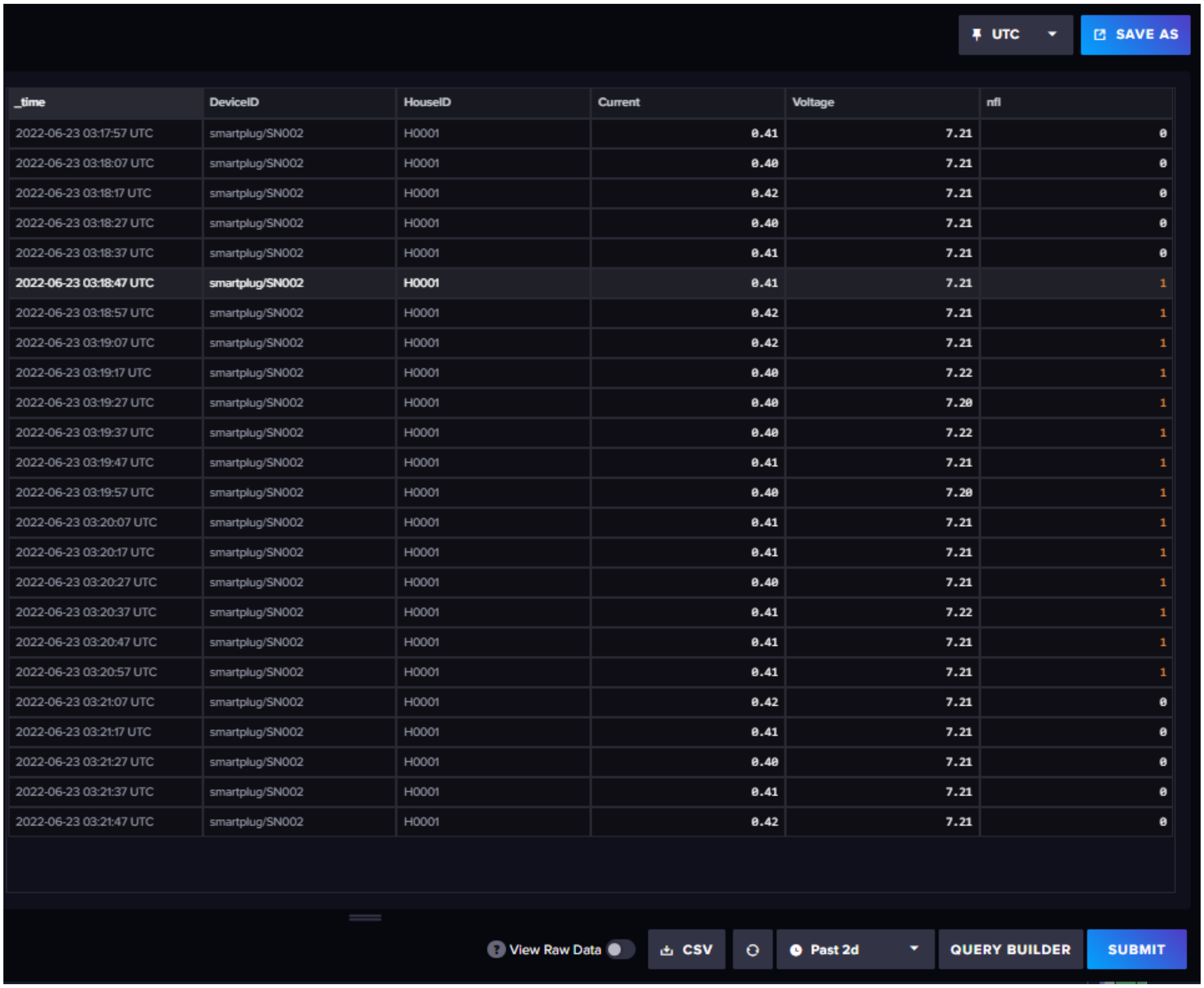


Table: smartplugs5

	HouseID	DeviceID	Current	Voltage	timestamp	NFL	Swtich
	Filter	SN001	Filter	Filter	>=2022-06-23 03:17:48+00:00	Filter	Filter
1	H0001	smartplug/SN001	0.68	7.14	2022-06-23 03:17:49+00:00	0.0	1.0
2	H0001	smartplug/SN001	0.71	7.14	2022-06-23 03:17:59+00:00	0.0	1.0
3	H0001	smartplug/SN001	0.67	7.15	2022-06-23 03:18:09+00:00	0.0	1.0
4	H0001	smartplug/SN001	0.71	7.13	2022-06-23 03:18:19+00:00	0.0	1.0
5	H0001	smartplug/SN001	0.7	7.14	2022-06-23 03:18:29+00:00	0.0	1.0
6	H0001	smartplug/SN001	0.68	7.13	2022-06-23 03:18:39+00:00	0.0	1.0
7	H0001	smartplug/SN001	0.69	7.14	2022-06-23 03:18:49+00:00	1.0	1.0
8	H0001	smartplug/SN001	0.68	7.15	2022-06-23 03:18:59+00:00	1.0	1.0
9	H0001	smartplug/SN001	0.67	7.15	2022-06-23 03:19:09+00:00	1.0	1.0
10	H0001	smartplug/SN001	0.72	7.14	2022-06-23 03:19:19+00:00	1.0	1.0
11	H0001	smartplug/SN001	0.7	7.14	2022-06-23 03:19:29+00:00	1.0	1.0
12	H0001	smartplug/SN001	0.67	7.14	2022-06-23 03:19:39+00:00	1.0	1.0
13	H0001	smartplug/SN001	0.72	7.11	2022-06-23 03:19:49+00:00	1.0	1.0
14	H0001	smartplug/SN001	0.71	7.12	2022-06-23 03:19:59+00:00	1.0	1.0
15	H0001	smartplug/SN001	0.68	7.14	2022-06-23 03:20:09+00:00	1.0	1.0
16	H0001	smartplug/SN001	0.7	7.14	2022-06-23 03:20:19+00:00	1.0	1.0
17	H0001	smartplug/SN001	0.72	7.14	2022-06-23 03:20:29+00:00	1.0	1.0
18	H0001	smartplug/SN001	0.68	7.16	2022-06-23 03:20:39+00:00	1.0	1.0
19	H0001	smartplug/SN001	0.73	7.13	2022-06-23 03:20:49+00:00	1.0	1.0
20	H0001	smartplug/SN001	0.74	7.12	2022-06-23 03:20:59+00:00	0.0	1.0
21	H0001	smartplug/SN001	0.7	7.12	2022-06-23 03:21:09+00:00	0.0	1.0
22	H0001	smartplug/SN001	0.72	7.12	2022-06-23 03:21:19+00:00	0.0	1.0
23	H0001	smartplug/SN001	0.71	7.12	2022-06-23 03:21:29+00:00	0.0	1.0
24	H0001	smartplug/SN001	0.69	7.14	2022-06-23 03:21:39+00:00	0.0	1.0
25	H0001	smartplug/SN001	0.74	7.15	2022-06-23 03:21:49+00:00	0.0	1.0

Figure 27-A screenshot of SQLite database for smart plug #1(SN001)-data rows recorded during the internet disconnection (nfl=1) are shown with blue color.



_time	DeviceID	HouseID	Current	Voltage	nfl
2022-06-23 03:17:57 UTC	smartplug/SN002	H0001	0.41	7.21	0
2022-06-23 03:18:07 UTC	smartplug/SN002	H0001	0.40	7.21	0
2022-06-23 03:18:17 UTC	smartplug/SN002	H0001	0.42	7.21	0
2022-06-23 03:18:27 UTC	smartplug/SN002	H0001	0.40	7.21	0
2022-06-23 03:18:37 UTC	smartplug/SN002	H0001	0.41	7.21	0
2022-06-23 03:18:47 UTC	smartplug/SN002	H0001	0.41	7.21	1
2022-06-23 03:18:57 UTC	smartplug/SN002	H0001	0.42	7.21	1
2022-06-23 03:19:07 UTC	smartplug/SN002	H0001	0.42	7.21	1
2022-06-23 03:19:17 UTC	smartplug/SN002	H0001	0.40	7.22	1
2022-06-23 03:19:27 UTC	smartplug/SN002	H0001	0.40	7.20	1
2022-06-23 03:19:37 UTC	smartplug/SN002	H0001	0.40	7.22	1
2022-06-23 03:19:47 UTC	smartplug/SN002	H0001	0.41	7.21	1
2022-06-23 03:19:57 UTC	smartplug/SN002	H0001	0.40	7.20	1
2022-06-23 03:20:07 UTC	smartplug/SN002	H0001	0.41	7.21	1
2022-06-23 03:20:17 UTC	smartplug/SN002	H0001	0.41	7.21	1
2022-06-23 03:20:27 UTC	smartplug/SN002	H0001	0.40	7.21	1
2022-06-23 03:20:37 UTC	smartplug/SN002	H0001	0.41	7.22	1
2022-06-23 03:20:47 UTC	smartplug/SN002	H0001	0.41	7.21	1
2022-06-23 03:20:57 UTC	smartplug/SN002	H0001	0.41	7.21	1
2022-06-23 03:21:07 UTC	smartplug/SN002	H0001	0.42	7.21	0
2022-06-23 03:21:17 UTC	smartplug/SN002	H0001	0.41	7.21	0
2022-06-23 03:21:27 UTC	smartplug/SN002	H0001	0.40	7.21	0
2022-06-23 03:21:37 UTC	smartplug/SN002	H0001	0.41	7.21	0
2022-06-23 03:21:47 UTC	smartplug/SN002	H0001	0.42	7.21	0

Figure 28- A screenshot of InfluxDB cloud user interface, showing uploaded records for SN001 during the internet disconnection.

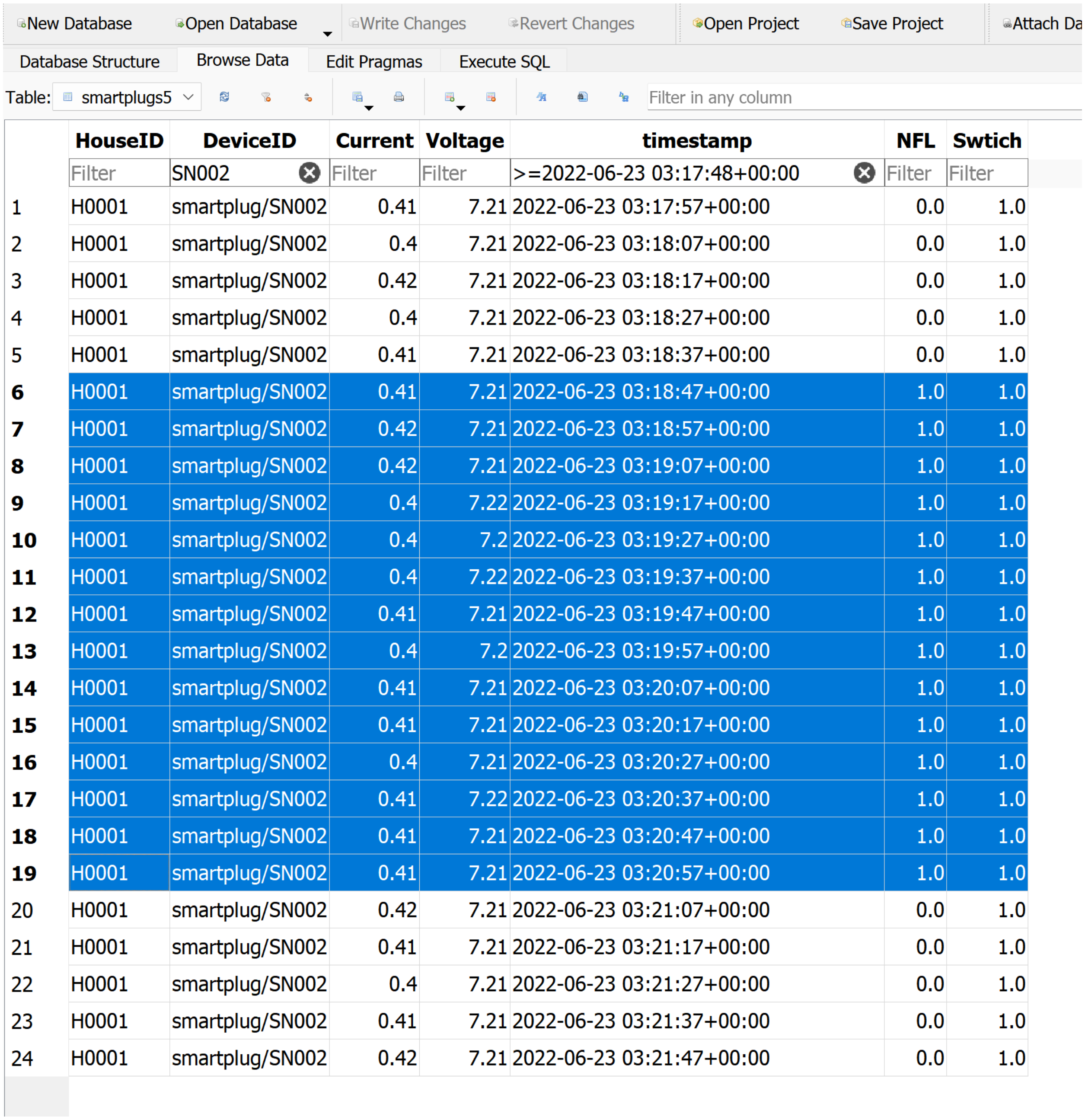
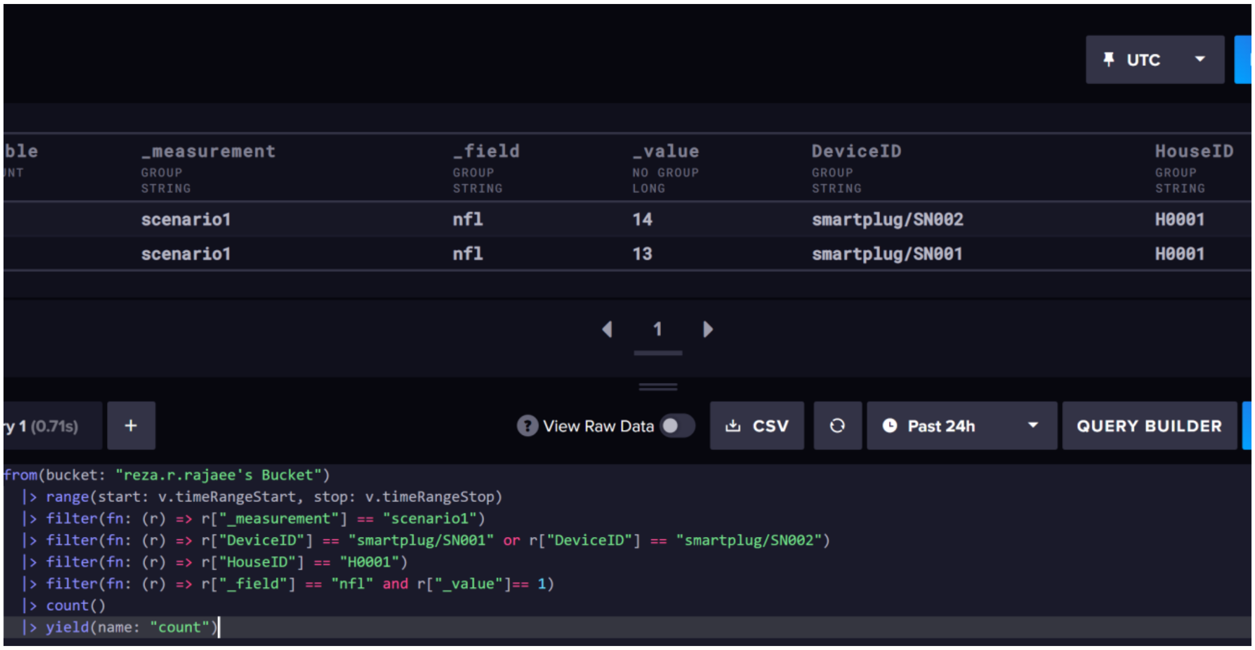


Table: smartplugs5

	HouseID	DeviceID	Current	Voltage	timestamp	NFL	Swtich
	Filter	SN002	Filter	Filter	>=2022-06-23 03:17:48+00:00	Filter	Filter
1	H0001	smartplug/SN002	0.41	7.21	2022-06-23 03:17:57+00:00	0.0	1.0
2	H0001	smartplug/SN002	0.4	7.21	2022-06-23 03:18:07+00:00	0.0	1.0
3	H0001	smartplug/SN002	0.42	7.21	2022-06-23 03:18:17+00:00	0.0	1.0
4	H0001	smartplug/SN002	0.4	7.21	2022-06-23 03:18:27+00:00	0.0	1.0
5	H0001	smartplug/SN002	0.41	7.21	2022-06-23 03:18:37+00:00	0.0	1.0
6	H0001	smartplug/SN002	0.41	7.21	2022-06-23 03:18:47+00:00	1.0	1.0
7	H0001	smartplug/SN002	0.42	7.21	2022-06-23 03:18:57+00:00	1.0	1.0
8	H0001	smartplug/SN002	0.42	7.21	2022-06-23 03:19:07+00:00	1.0	1.0
9	H0001	smartplug/SN002	0.4	7.22	2022-06-23 03:19:17+00:00	1.0	1.0
10	H0001	smartplug/SN002	0.4	7.2	2022-06-23 03:19:27+00:00	1.0	1.0
11	H0001	smartplug/SN002	0.4	7.22	2022-06-23 03:19:37+00:00	1.0	1.0
12	H0001	smartplug/SN002	0.41	7.21	2022-06-23 03:19:47+00:00	1.0	1.0
13	H0001	smartplug/SN002	0.4	7.2	2022-06-23 03:19:57+00:00	1.0	1.0
14	H0001	smartplug/SN002	0.41	7.21	2022-06-23 03:20:07+00:00	1.0	1.0
15	H0001	smartplug/SN002	0.41	7.21	2022-06-23 03:20:17+00:00	1.0	1.0
16	H0001	smartplug/SN002	0.4	7.21	2022-06-23 03:20:27+00:00	1.0	1.0
17	H0001	smartplug/SN002	0.41	7.22	2022-06-23 03:20:37+00:00	1.0	1.0
18	H0001	smartplug/SN002	0.41	7.21	2022-06-23 03:20:47+00:00	1.0	1.0
19	H0001	smartplug/SN002	0.41	7.21	2022-06-23 03:20:57+00:00	1.0	1.0
20	H0001	smartplug/SN002	0.42	7.21	2022-06-23 03:21:07+00:00	0.0	1.0
21	H0001	smartplug/SN002	0.41	7.21	2022-06-23 03:21:17+00:00	0.0	1.0
22	H0001	smartplug/SN002	0.4	7.21	2022-06-23 03:21:27+00:00	0.0	1.0
23	H0001	smartplug/SN002	0.41	7.21	2022-06-23 03:21:37+00:00	0.0	1.0
24	H0001	smartplug/SN002	0.42	7.21	2022-06-23 03:21:47+00:00	0.0	1.0

Figure 29- A screenshot of SQLite database for smart plug #2 (SN002)-data rows recorded during the internet disconnection (nfl=1) are shown with blue color.



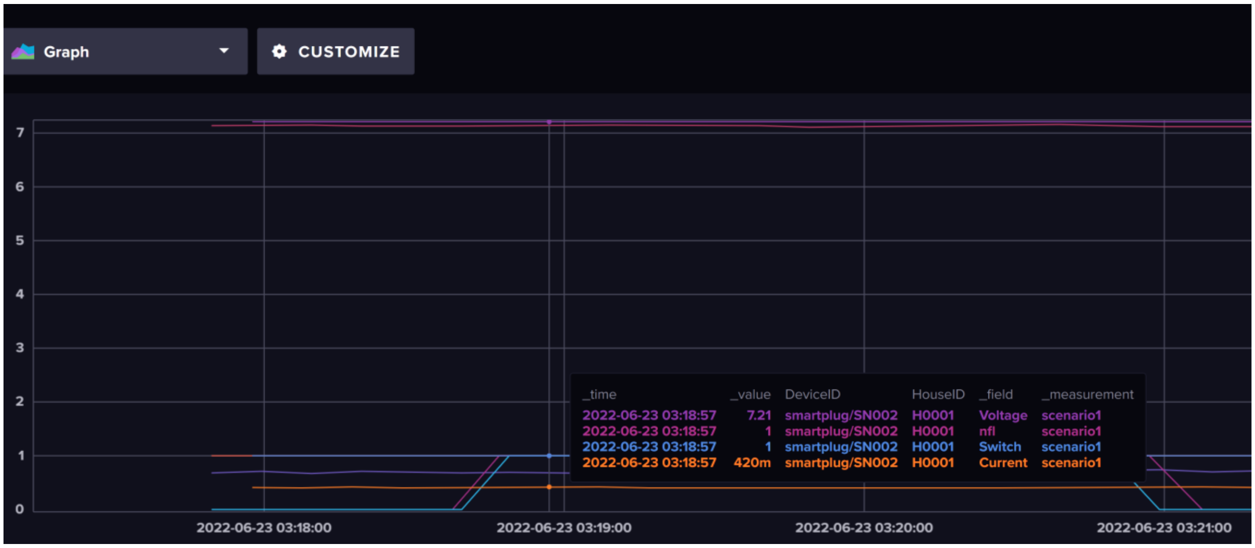
ble NT	_measurement GROUP STRING	_field GROUP STRING	_value NO GROUP LONG	DeviceID GROUP STRING	HouseID GROUP STRING
	scenario1	nfl	14	smartplug/SN002	H0001
	scenario1	nfl	13	smartplug/SN001	H0001

```
from(bucket: "reza.r.rajaee's Bucket")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "scenario1")
  |> filter(fn: (r) => r["DeviceID"] == "smartplug/SN001" or r["DeviceID"] == "smartplug/SN002")
  |> filter(fn: (r) => r["HouseID"] == "H0001")
  |> filter(fn: (r) => r["_field"] == "nfl" and r["_value"]== 1)
  |> count()
  |> yield(name: "count")
```

Figure 30-InfluxDB cloud user interface showing the number of records with "nfl"=1" for each plug. The code block in the bottom was used for the query.

_time	_value	DeviceID	HouseID	_field	_measurement
2022-06-23 03:18:49	7.14	smartplug/SN001	H0001	Voltage	scenario1
2022-06-23 03:18:49	1	smartplug/SN001	H0001	Switch	scenario1
2022-06-23 03:18:49	1	smartplug/SN001	H0001	nfl	scenario1
2022-06-23 03:18:49	690m	smartplug/SN001	H0001	Current	scenario1

Figure 31- InfluxDB cloud explorer showing the records for smartplug#1 (SN001)



_time	_value	DeviceID	HouseID	_field	_measurement
2022-06-23 03:18:57	7.21	smartplug/SN002	H0001	Voltage	scenario1
2022-06-23 03:18:57	1	smartplug/SN002	H0001	nfl	scenario1
2022-06-23 03:18:57	1	smartplug/SN002	H0001	Switch	scenario1
2022-06-23 03:18:57	420m	smartplug/SN002	H0001	Current	scenario1

Figure 32-InfluxDB cloud explorer showing the records for smartplug#2 (SN002)

Comparing the records of Figure 26 (InfluxDB cloud) with Figure 27 (SQLite), and Figure 28 with Figure 29 we can see all the results are the same in both databases, suggesting the local database successfully saved smart plugs #1 and #2 data respectively during the internet disconnection. Figure 31 and Figure 32 present all recorded measurements by smart plug #1 and #2 (current, voltage, relay state, and NFL).

5.3 Scenario 2

Scenario 2 is defined to show:

1) the smart plugs in multiple houses can receive commands from the CEMS and send their measurements to the smart homes (communication unit).

2) to demonstrate the reduction in electricity consumption based on a tiered price plan.

The price plan used was the Rate D of Hydro Quebec[5], which is a tiered price plan. The price plan details are shown in Table 10 (*Rate D | Hydro-Québec*, n.d.) and smart plugs type and assigned loads are presented in Table 11. In this scenario, each SHEMS hub controlled the energy

---

[5] Hydro Quebec is the utility responsible for electricity generation and distribution in Quebec province, Canada.

consumption and as the total energy consumption exceeded 40 kWh the deferrable load (SN002) switched and remained OFF.

Accordingly, the data recorded in the local database (SQLite) shows the same pattern. Figure 33 and Figure 34 show two snapshots of a table which was designed only for debugging and verification purposes. the attribute *Deferrable_FL* shows the status of the controllable loads (SN002); as the total house energy passes 40 kW, the deferrable loads are switched OFF.

Moving to the graphs, Figure 35 shows measured power consumption for the Reference Scenario for both smart plugs (SN001, SN002pattern in house 3 (H0003). Similarly, Figure 36 gives the power consumption details for Scenario 2. By comparing measured power consumption for Reference Scenario with Scenario 2, for all houses the following insights can be realized:

1) The deferrable load represented by SN002 meter, turned OFF (at around the 12th hour) and locked until the rest of the day.
2) Figure 37 presents a comparison of energy cost for the reference and Scenario 2 power consumption in house 3. Both red and green trends, align with each other before the 12th hour when the energy consumption reaches to 40 kWh. After that a difference in cost is visible between the two load patterns. The alignment of the cost trends before the 12th hour suggests the stable performance of smart plugs' measurement unit.

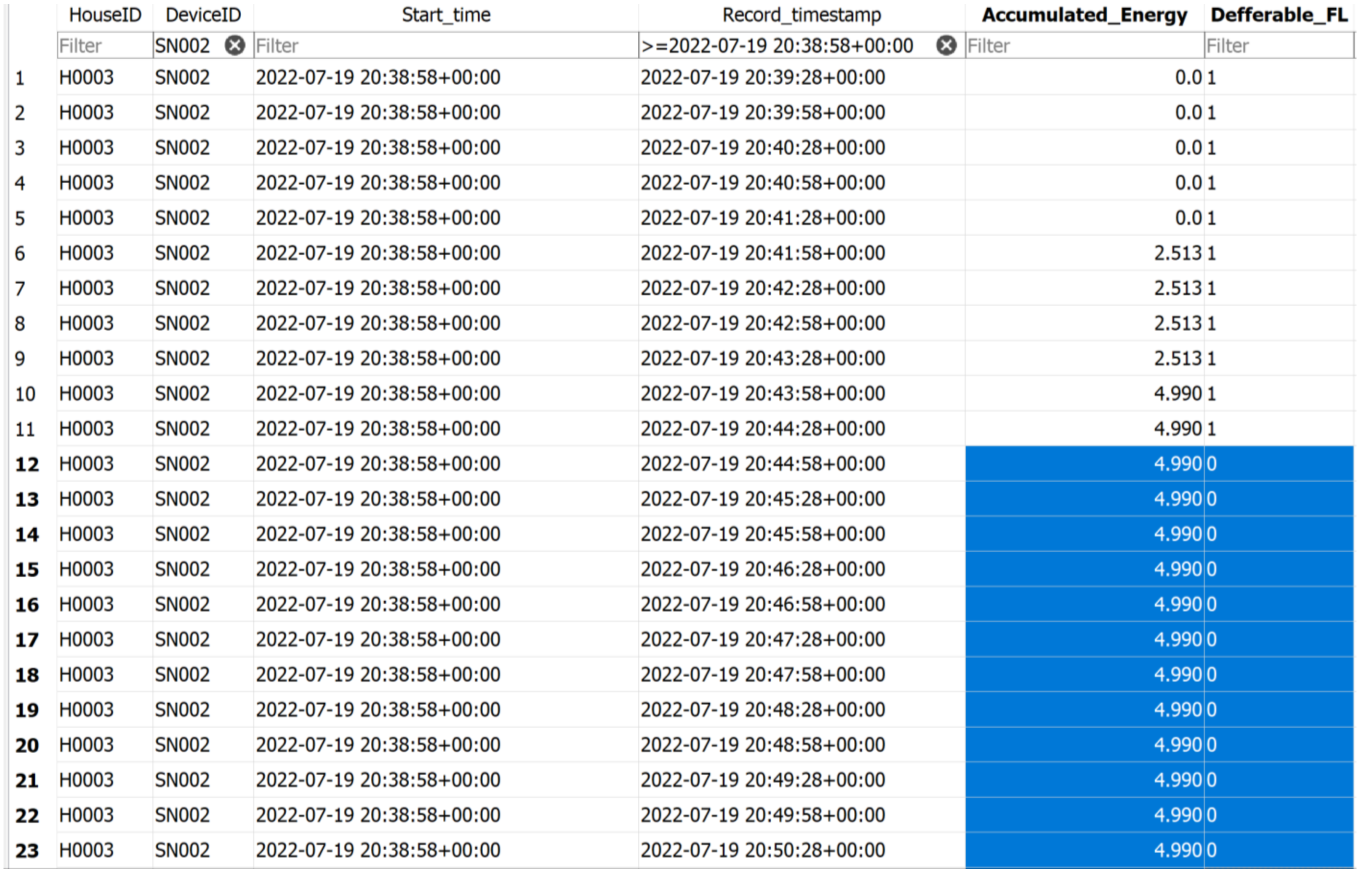
The *measured power consumption* and *cost comparison* graphs for house 1 and 2, can be found in the Appendix (section Appendix A-Scenario 2 graphs).

Table 10-Hydro-Quebec Rate D price plan

Condition	**Price**
Price applicable to energy consumed up to 40 kWh times the number of days in the consumption period (1st tier)	6.159 ¢/kWh
Price applicable to remaining energy consumed (2nd tier)	9.502 ¢/kWh

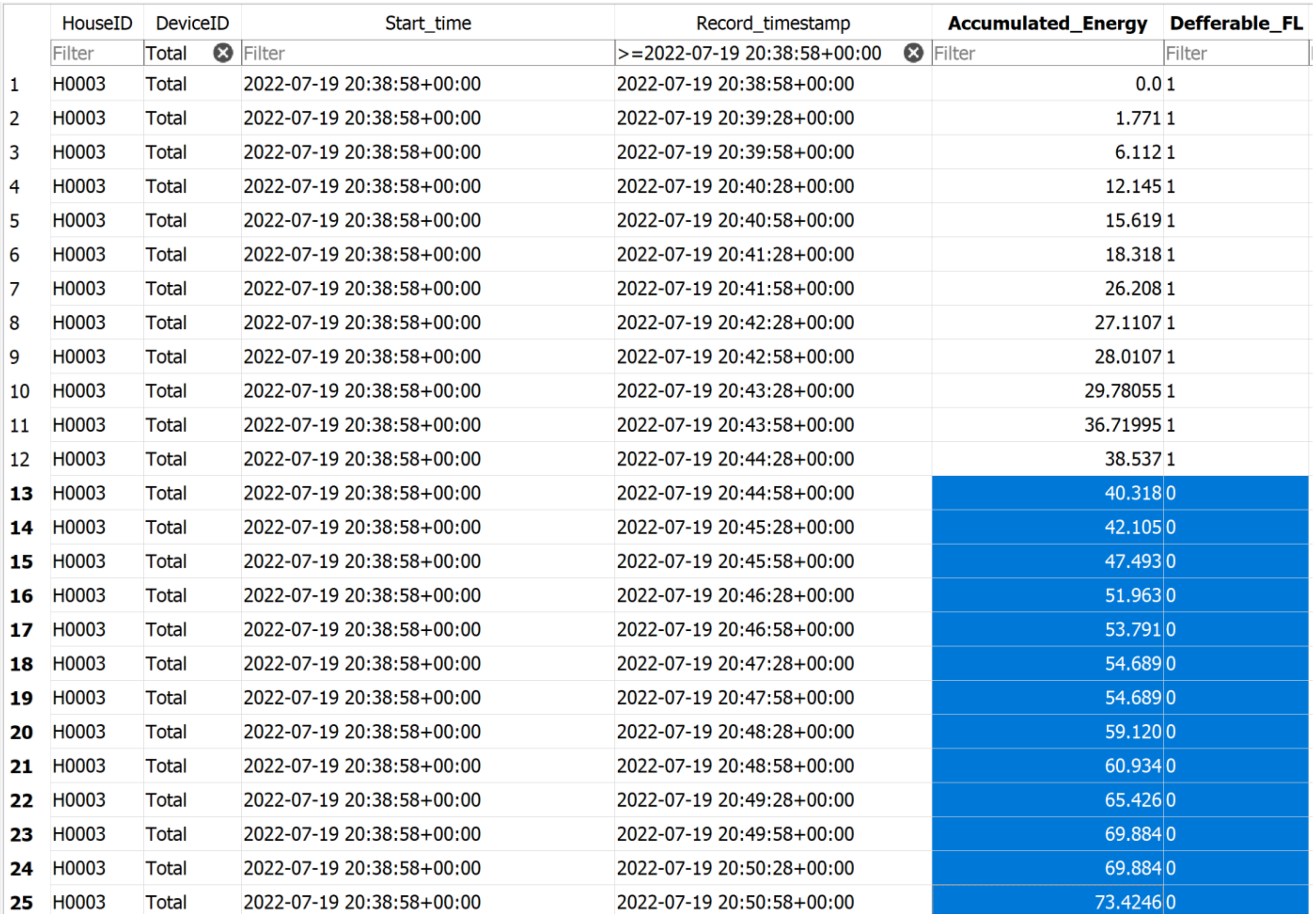
Table 11-Scenario 2 and 3 configurations

House ID	Hub device	Smart plug load representation	Smart plug type	Power Consumer	Smart plug ID
H0001	Raspberry Pi 4	Appliance Group 1 (non-deferrable)	Custom designed	5W LED strip (DC)	SN001
		Appliance Group 2 deferrable	Commercial	2.5 W filament lamp (AC)	SN002
H0002	Raspberry Pi 4	Appliance Group 1 (non-deferrable)	Custom designed	5W LED strip (DC)	SN001
		Appliance Group 2 deferrable	Custom designed	2.5 W LED bar (DC)	SN002
H0003	Raspberry Pi 4	Appliance Group 1 (non-deferrable)	Custom designed	5W LED strip (DC)	SN001
		Appliance Group 2 deferrable	Custom designed	2.5 W LED bar (DC)	SN002



	HouseID	DeviceID	Start_time	Record_timestamp	**Accumulated_Energy**	**Defferable_FL**
	Filter	SN002	Filter	>=2022-07-19 20:38:58+00:00	Filter	Filter
1	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:39:28+00:00	0.0	1
2	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:39:58+00:00	0.0	1
3	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:40:28+00:00	0.0	1
4	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:40:58+00:00	0.0	1
5	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:41:28+00:00	0.0	1
6	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:41:58+00:00	2.513	1
7	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:42:28+00:00	2.513	1
8	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:42:58+00:00	2.513	1
9	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:43:28+00:00	2.513	1
10	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:43:58+00:00	4.990	1
11	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:44:28+00:00	4.990	1
12	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:44:58+00:00	4.990	0
13	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:45:28+00:00	4.990	0
14	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:45:58+00:00	4.990	0
15	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:46:28+00:00	4.990	0
16	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:46:58+00:00	4.990	0
17	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:47:28+00:00	4.990	0
18	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:47:58+00:00	4.990	0
19	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:48:28+00:00	4.990	0
20	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:48:58+00:00	4.990	0
21	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:49:28+00:00	4.990	0
22	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:49:58+00:00	4.990	0
23	H0003	SN002	2022-07-19 20:38:58+00:00	2022-07-19 20:50:28+00:00	4.990	0

Figure 33-A screenshot from SQLite database showing the records for SN002 hourly energy consumption-house 3. Once the energy consumption crossed 40kWh, the deferrable load was switched and remained OFF for the rest of the day.



	HouseID	DeviceID	Start_time	Record_timestamp	**Accumulated_Energy**	**Defferable_FL**
	Filter	Total	Filter	>=2022-07-19 20:38:58+00:00	Filter	Filter
1	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:38:58+00:00	0.0	1
2	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:39:28+00:00	1.771	1
3	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:39:58+00:00	6.112	1
4	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:40:28+00:00	12.145	1
5	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:40:58+00:00	15.619	1
6	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:41:28+00:00	18.318	1
7	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:41:58+00:00	26.208	1
8	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:42:28+00:00	27.1107	1
9	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:42:58+00:00	28.0107	1
10	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:43:28+00:00	29.78055	1
11	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:43:58+00:00	36.71995	1
12	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:44:28+00:00	38.537	1
13	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:44:58+00:00	40.318	0
14	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:45:28+00:00	42.105	0
15	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:45:58+00:00	47.493	0
16	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:46:28+00:00	51.963	0
17	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:46:58+00:00	53.791	0
18	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:47:28+00:00	54.689	0
19	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:47:58+00:00	54.689	0
20	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:48:28+00:00	59.120	0
21	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:48:58+00:00	60.934	0
22	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:49:28+00:00	65.426	0
23	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:49:58+00:00	69.884	0
24	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:50:28+00:00	69.884	0
25	H0003	Total	2022-07-19 20:38:58+00:00	2022-07-19 20:50:58+00:00	73.4246	0

Figure 34-SQLite database screenshot for total house hourly energy consumption-house 3

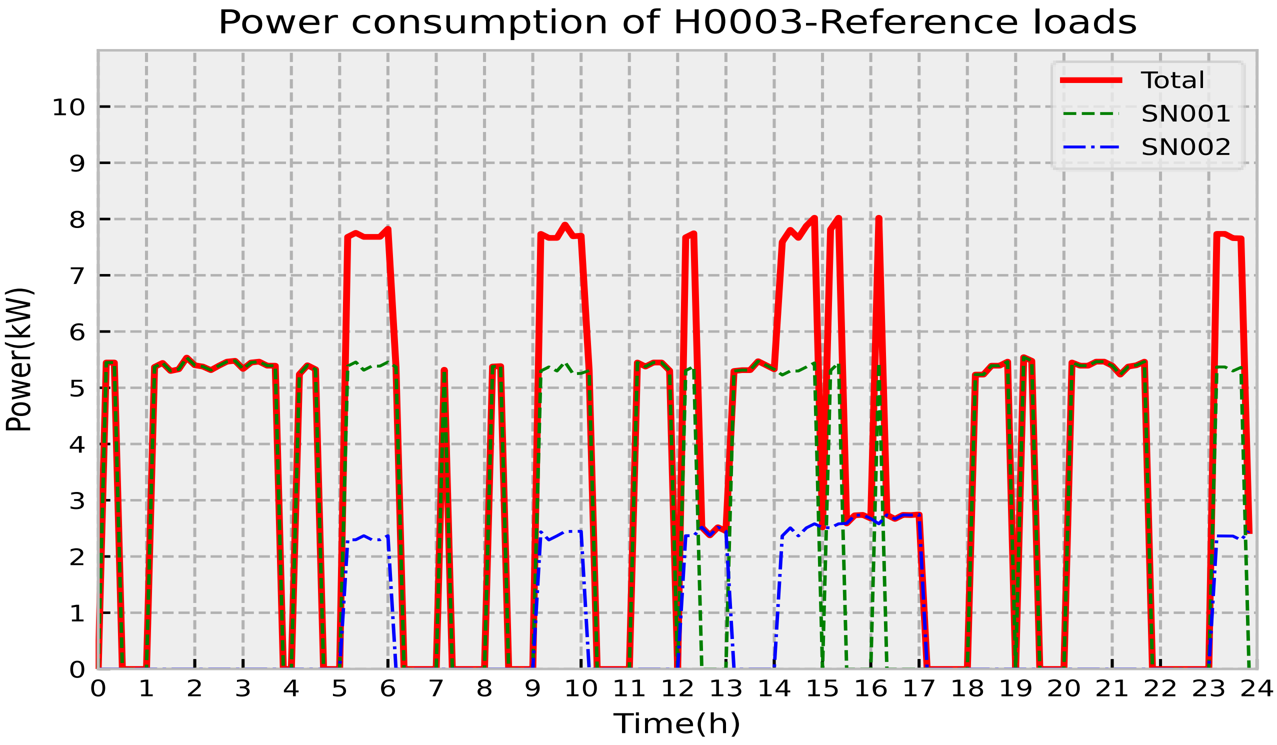


Figure 35-Measured power consumption-H0003-Reference

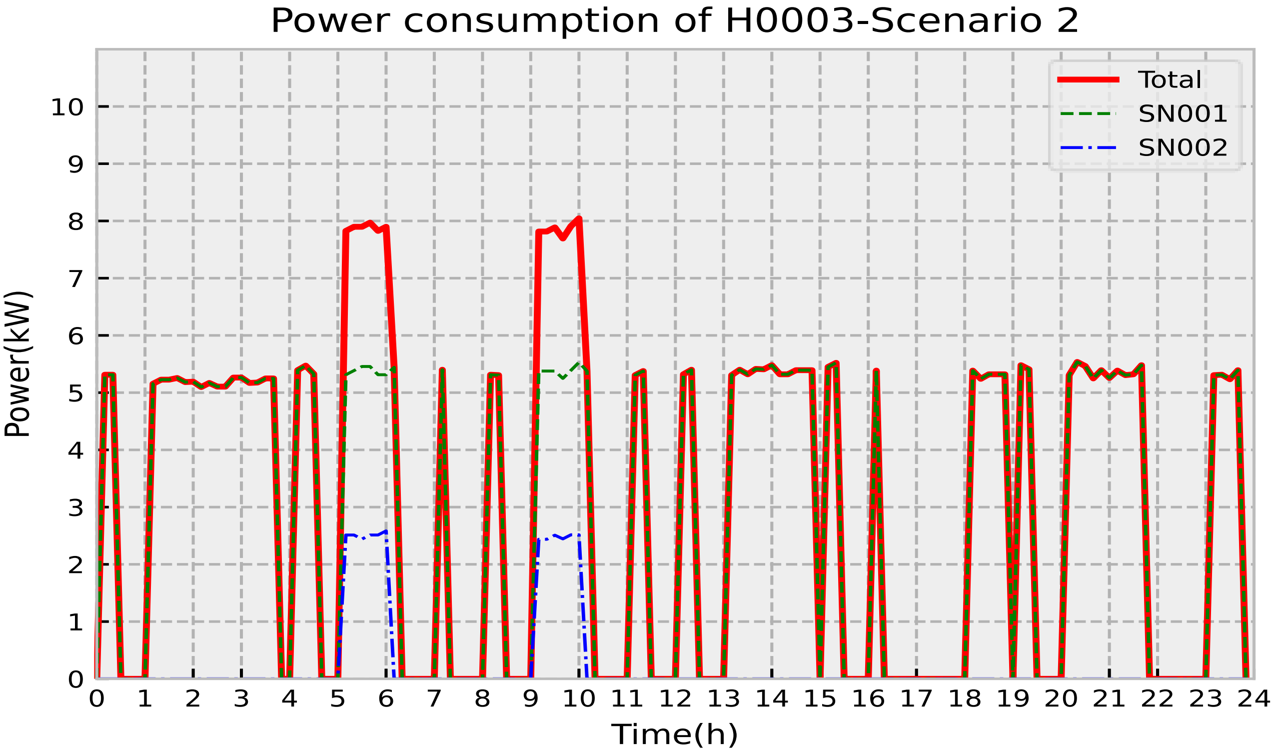


Figure 36-Measured power consumption-H0003-Scenario 2

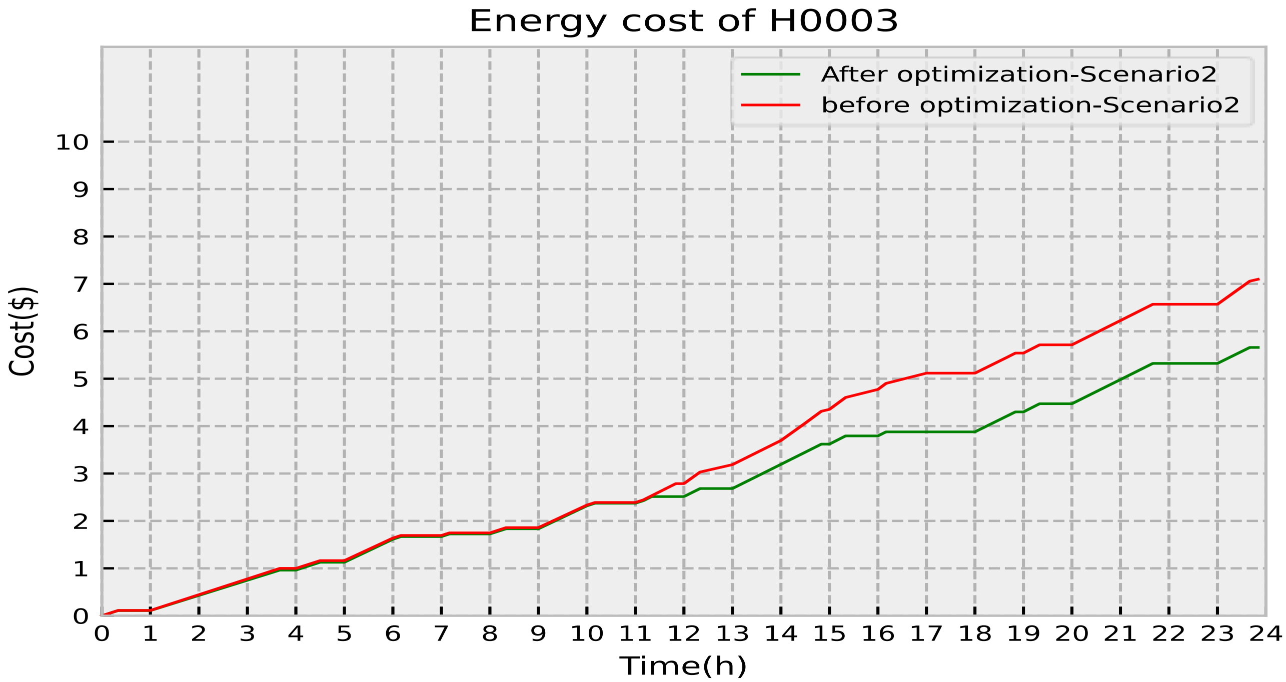


Figure 37-Energy cost comparison H0003-Scenario 2 vs. reference

5.4 Scenario 3

The demonstration configuration for Scenario 2 and 3 is the same, except for the price plans–in Scenario 3 time-of-use (ToU) price plan is used (Table 12). Both scenarios show the ability of the system in working with different price plans.

Table 12- The Time-of-use price plan

Price mode	Duration	The Price
Low night price	00:00 to 07:59 and 19:00 to 23:59	6 ¢/kWh
High day price	8:00 to 18:59	10 ¢/kWh

Similar to Scenario 2, the plug #1 (SN001) is considered as non-deferrable load and plug #2 as deferrable load. Mathematically speaking, if we let set T be defined as: T = {0, 1 2, 3, …., 23} and set H be defined as H = {1,2,3}, the optimization equations would be:

Objective Function:

$$\text{Min } \sum_{t=0}^{23}\left(P_{NDef}^{ht} + P_{Def}^{ht}\right) \times r_t \qquad \forall\, t \in T, and \quad \forall\, h \in H \qquad (1)$$

subject to:

$$P_{NDef}^{ht} + P_{Def}^{ht} < P_{lim-indiv} \qquad \forall\, t \in T \text{ and } \forall\, h \in H \qquad (2)$$

$$\sum_{t=0}^{23} P_{Def\ Before\ Optimization} = \sum_{t=0}^{23} P_{Def\ After\ Optimization} \qquad \forall\, t \in T\ and\ h \in H$$

(3)

Where:

r_t : *the hourly price according to Table 12*

P_{Def}^{ht}: $power\ consumption\ of\ deferrable\ loads$

P_{NDef}^{het}: $power\ consumption\ of\ non-deferrable\ loads$

$P_{\lim-indiv}$: $power\ limit\ of\ each\ house$

As an example, in Scenario 3 power limit of each house is set to 7 kW.

Figure 38 shows the optimized loads pattern generated by CEMS (Titled as “Expected power consumption”). As explained before (Table 6), these signals are sent to each smart home’s hub every hour. Subsequently, smart homes distribute them between the associated smart plugs. The result of this process and power consumption measurements is shown in Figure 39.

By looking at Figure 39 it can be seen that SN001 power consumption (green line) during 6th to 9th hour continuously fluctuated around 5 kW, while between 0 to 6th hour, when the expected power consumption was less than 5 kW, there was a switch OFF at every hour. This pattern appears for the power consumptions less than 5 kW (e.g., 2 kW) and shows that the load model discussed in section 4.5 was implemented successfully.

By comparison of the expected power consumption graphs with measured power consumption values in Scenario 3 the following results can be concluded:

1) the devices turn ON and OFF exactly according to the payload generation time at the central server, demonstrating the consistent performance of the communication layer in transferring the data packets without a noticeable delay.
2) as a result, energy cost decreased in Scenario 3 (Figure 40).

These results are also true for H0002 and H0003; the graphs are presented Appendix (section Appendix B-Scenario 3 graphs).

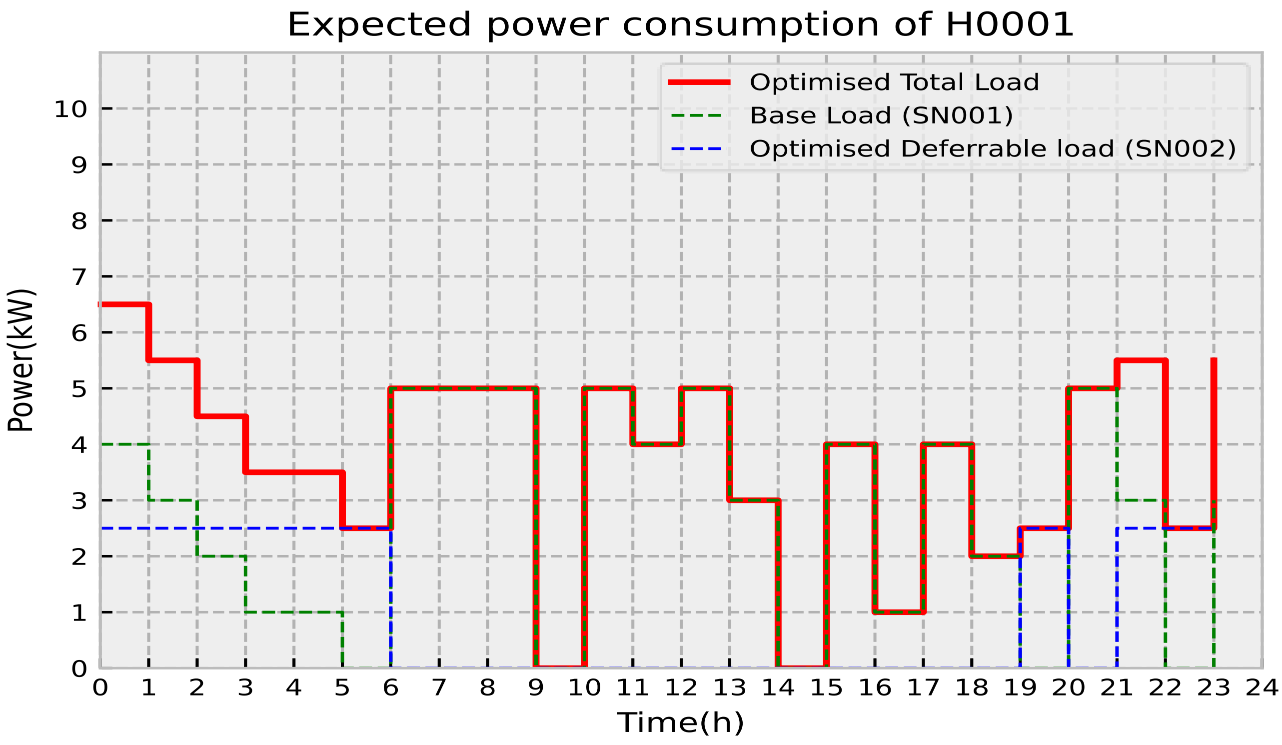


Figure 38-Expected power consumption-H0001-Scenario 3

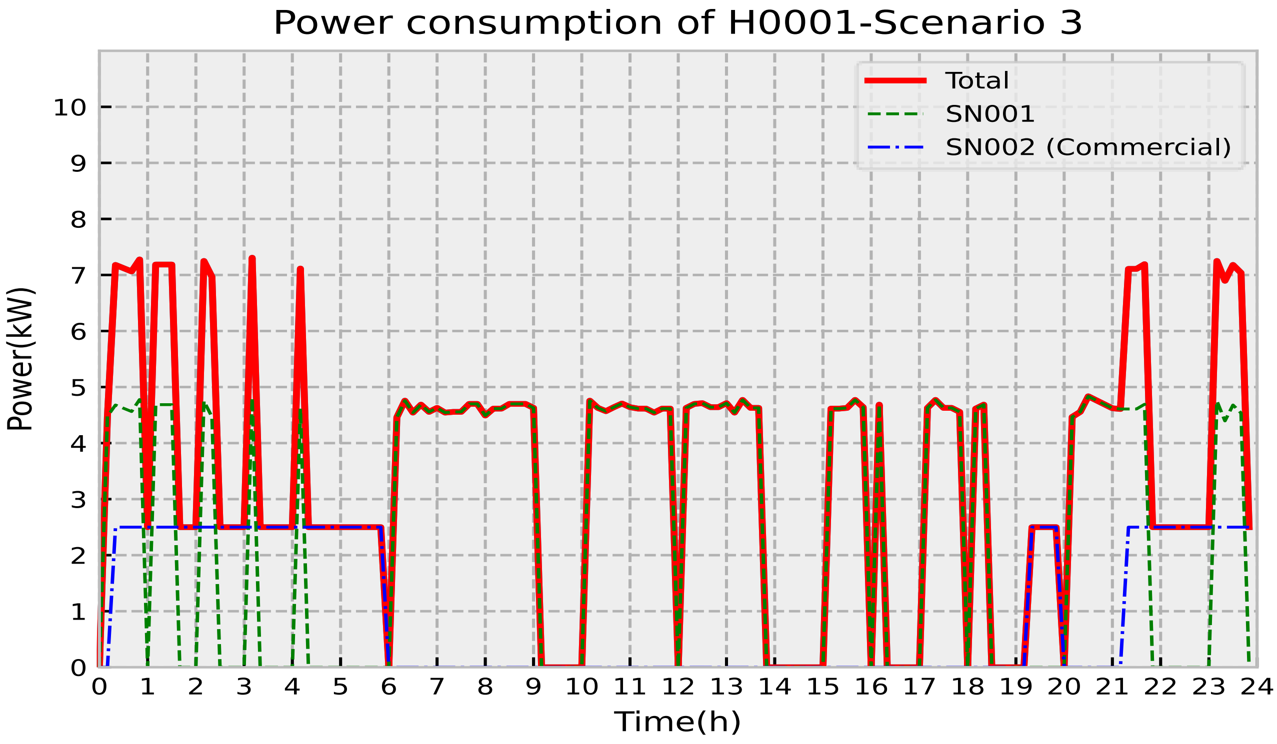


Figure 39-Measured power consumption-H0001-Scenario3

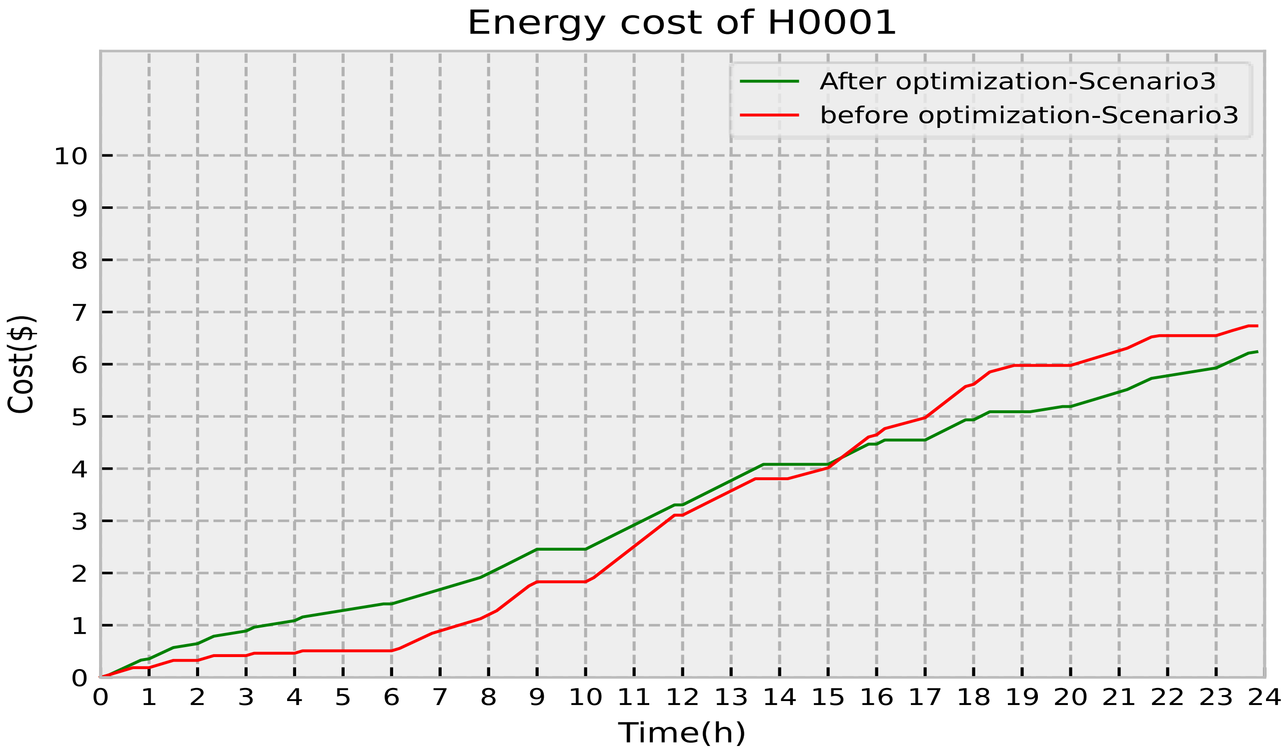


Figure 40-Energy cost comparison H0001-Scenario 3 vs. reference

5.5 Scenario 4

In Scenario 4, the price plan is the same as Scenario 3, but both smart plug #2 and smart plug #1 measure an appliance with a deferrable load (Figure 56).

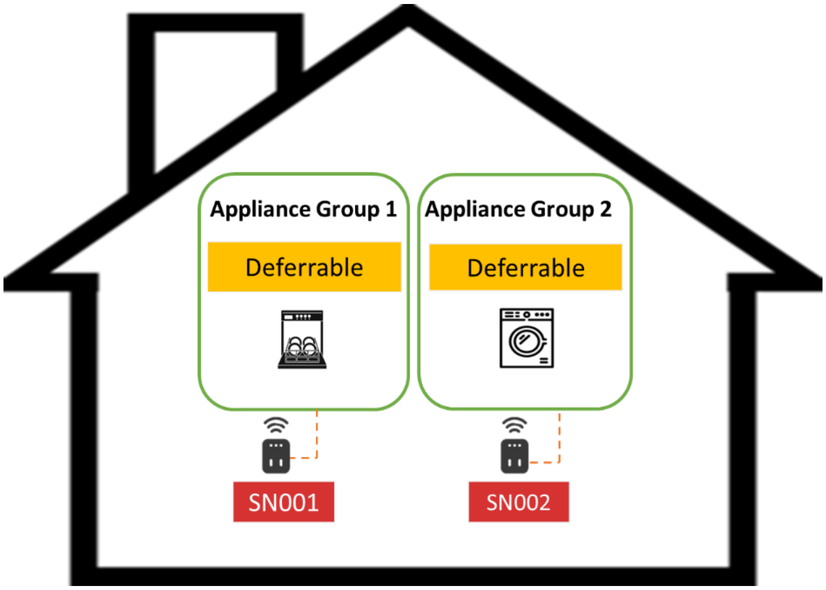


Figure 41-Schematic of the load setup for scenario 4

In this scenario, in addition to the previous scenario's constraints, the loads should be shifted without exceeding the limit of a hypothetical transformer. This constraint requires the appliances power consumption data of *every house in the cluster (houses 1,2 and 3)* to be aggregated in one system (CEMS). In mathematical notation the optimization equations are:

Objective function:

$$\text{Min } \sum_{t=0}^{23} \left(P_{NDef}^{ht} + P_{Def}^{ht}\right) \times r_t \tag{4}$$

Subject to:

$$P_{Def1}^{ht} + P_{Def2}^{ht} < P_{lim-indiv} \quad \forall\, t \in T, and \quad \forall\, h \in H \tag{5}$$

$$\sum_{h=1}^{3} \left(P_{Def1}^{ht} + P_{Def2}^{ht}\right) < P_{lim-total} \quad \forall\, t \in T, and \quad \forall\, h \in H \tag{6}$$

$$\sum_{t=0}^{23} {P_{Def1}^{ht}}_{Before\ Optimization} = \sum_{t=0}^{23} {P_{Def1}^{ht}}_{After\ Optimization} \tag{7}$$

$$\sum_{t=0}^{23} {P_{Def2}^{ht}}_{Before\ Optimization} = \sum_{t=0}^{23} {P_{Def2}^{ht}}_{After\ Optimization} \tag{8}$$

Where:

r_t : the hourly price according to Table 12

$\mathrm{P}_{\mathrm{Def}}^{\mathrm{ht}}$: Power consumption of deferrable loads group1

$\mathrm{P}_{\mathrm{Def2}}^{\mathrm{ht}}$: Power consumption of deferrable load group 2

$\mathrm{P}_{\mathrm{lim-indiv}}$: power limit of the individual house

As an example, the individual house power limit was set 7 kW (Same as Scenario 3) and the total house cluster power limit was set to 18 kW. Table 13 presents the configuration of scenario 4.

Table 13-Scenario 4 configuration

House ID	Hub device	Smart plug load representation	Smart plug type	Power Consumer	Smart plug ID
H0001	Raspberry Pi 4	Appliance 1 deferrable	Custom designed	5W LED strip (DC)	SN001
		Appliance 2 deferrable	Commercial	2.5 W filament lamp (AC)	SN002
H0002	Raspberry Pi 4	Appliance 1 deferrable	Custom designed	5W LED strip (DC)	SN001
		Appliance 2 deferrable	Custom designed	2.5 W LED bar (DC)	SN002
H0003	Raspberry Pi 4	Appliance 1 deferrable	Custom designed	5W LED strip (DC)	SN001
		Appliance 2 deferrable	Custom designed	2.5 W LED bar (DC)	SN002

The comparison of the expected power consumption graph with the measured power consumptions graph in every house, suggests that:

1) the power consumptions of both plugs are shifted to the hours with the lower tariffs. As a result, the energy cost was decreased in all houses.
2) the constraint of not exceeding the hypothetical transformer limit (18 kW) was satisfied at every hour. In other words, the summation of the three houses total power consumption remained below approximately 18 kW. This can be seen in Figure 51, where the average of the datapoints in every hour, remains below 18 kW.

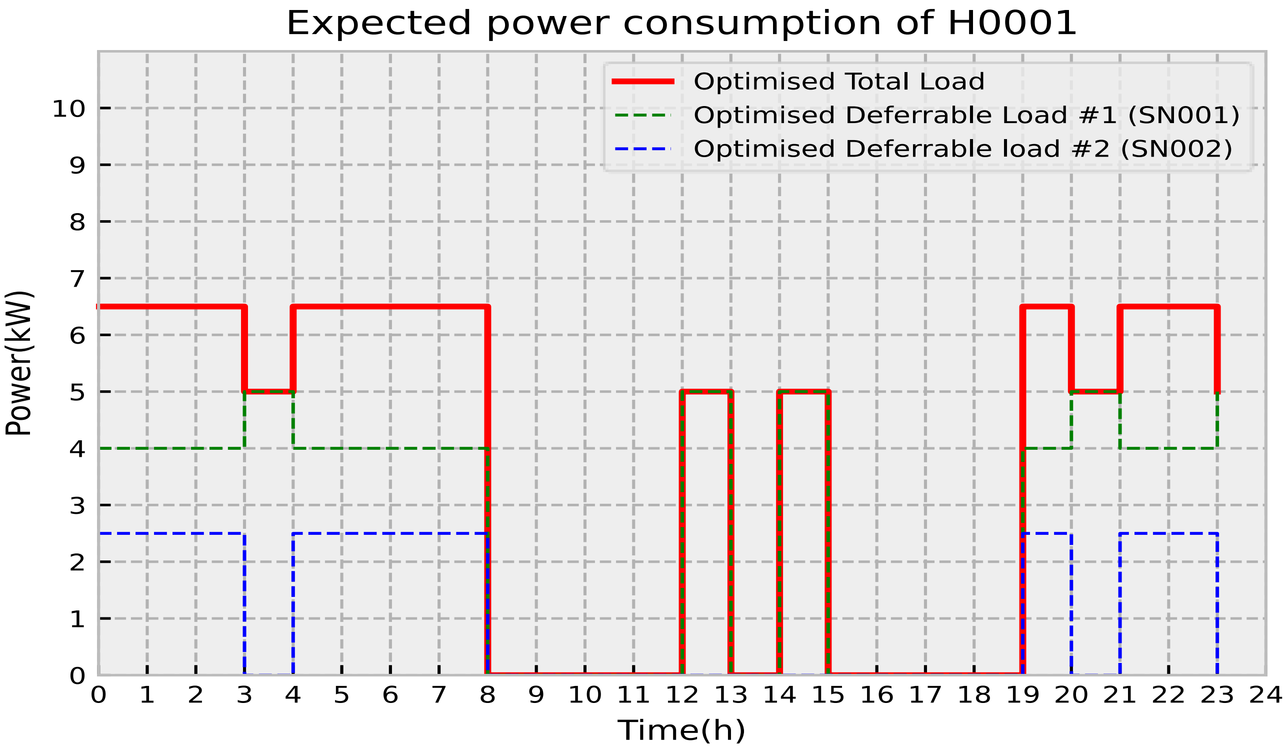


Figure 42-Expected power consumption- H0001-Scenario 4

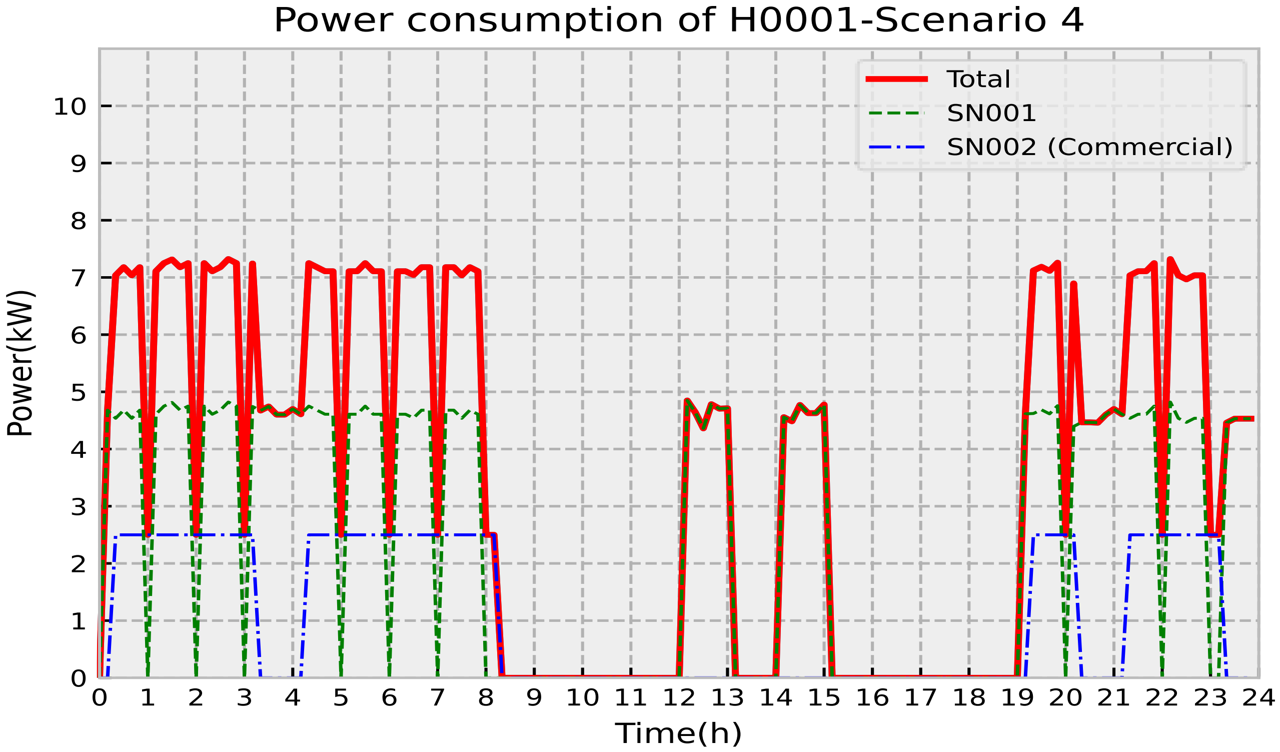


Figure 43-Measured power consumption-H0001-Scenario 4

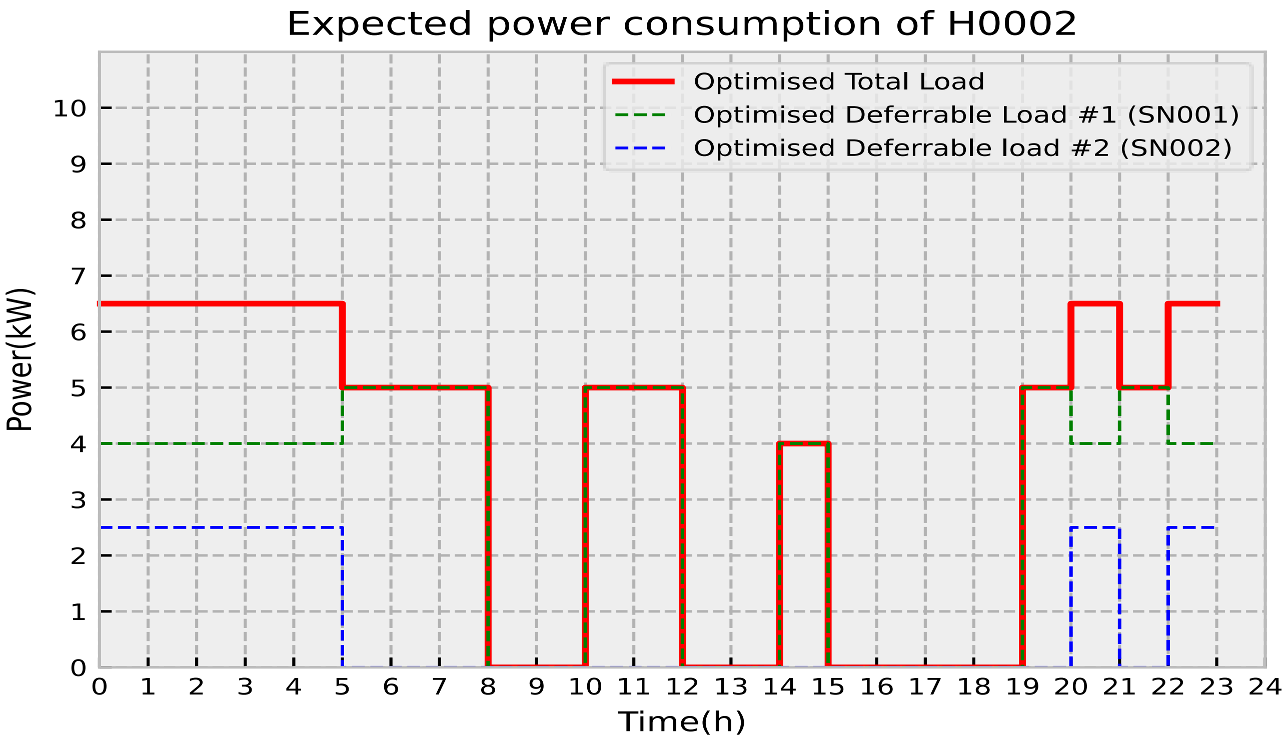


Figure 44-Expected power consumption-H0001-Scenario 4

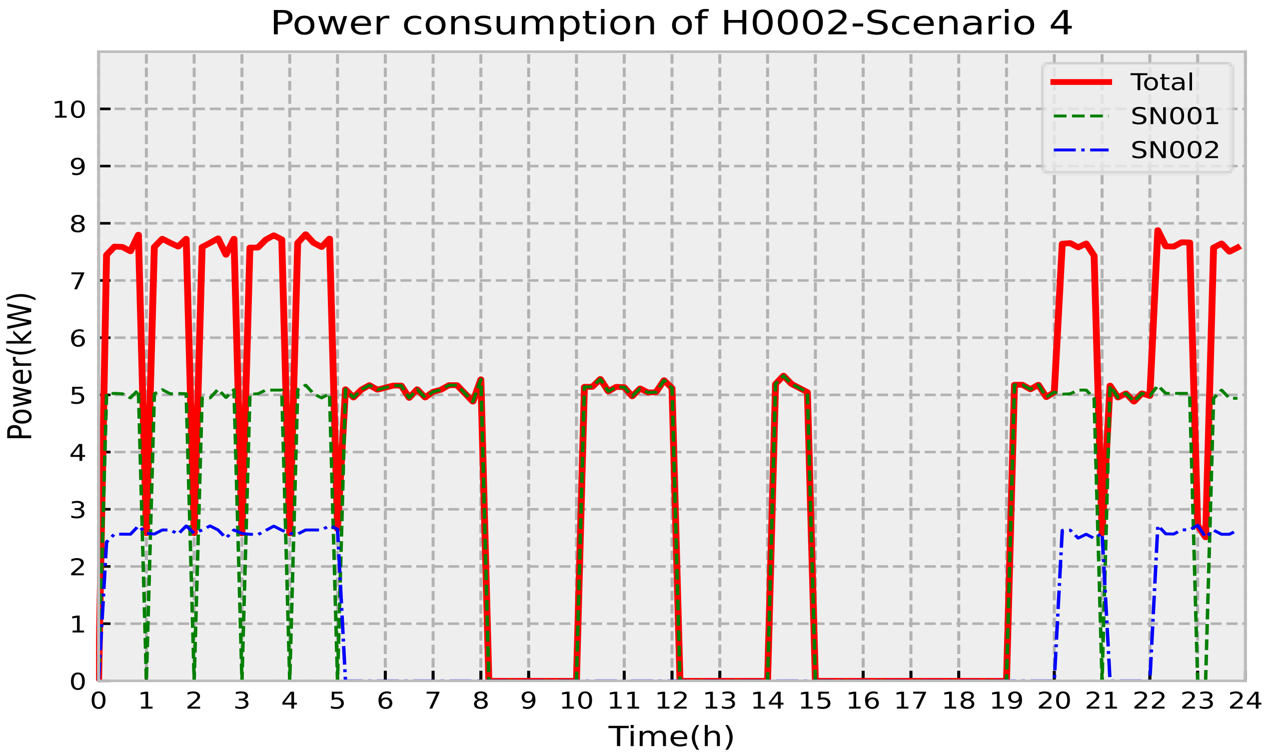


Figure 45-Measured power consumption-H0002-Scenario 4

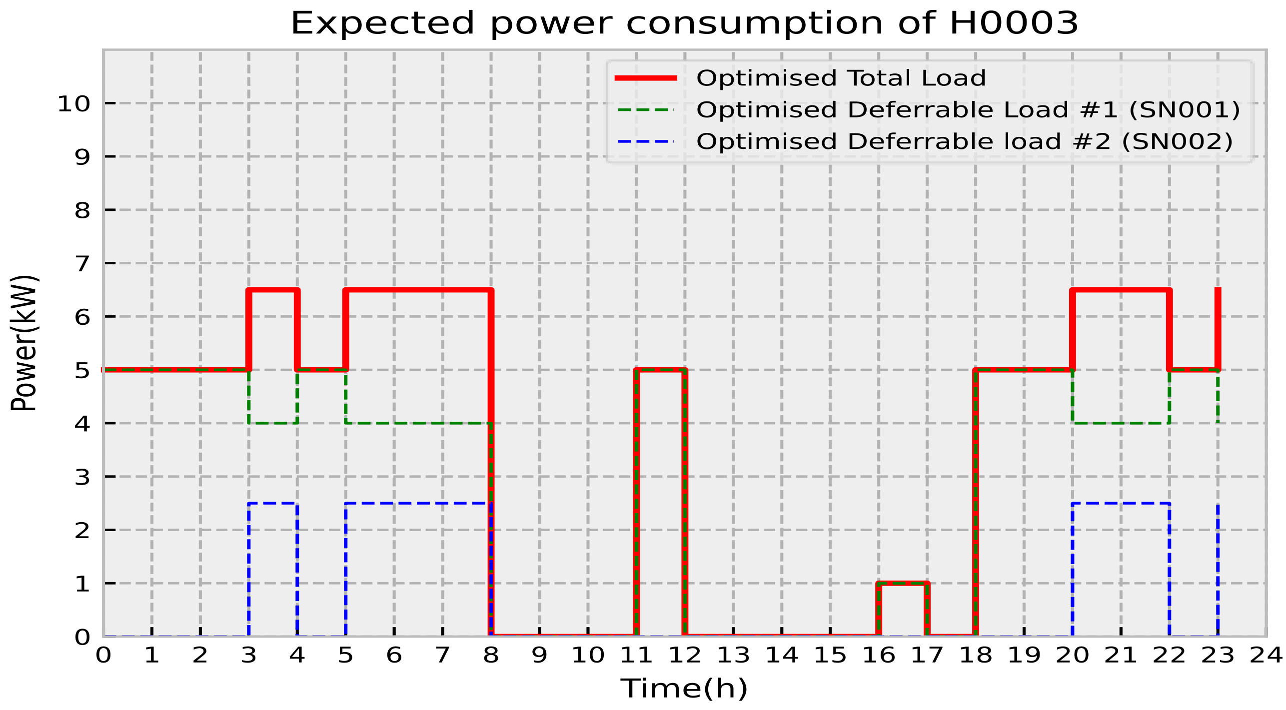


Figure 46-Expected power consumption-H0003-Scenario 4

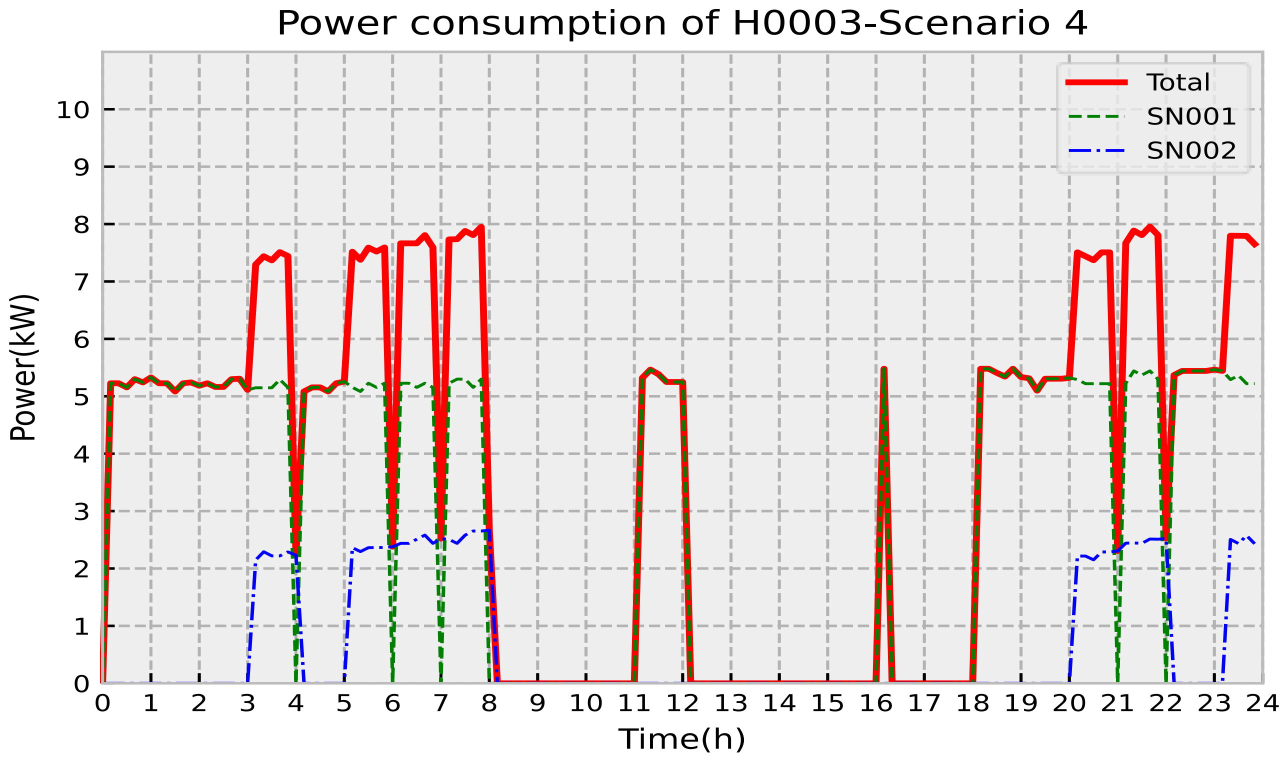


Figure 47-Measured power consumption-H0003-Scenario 4

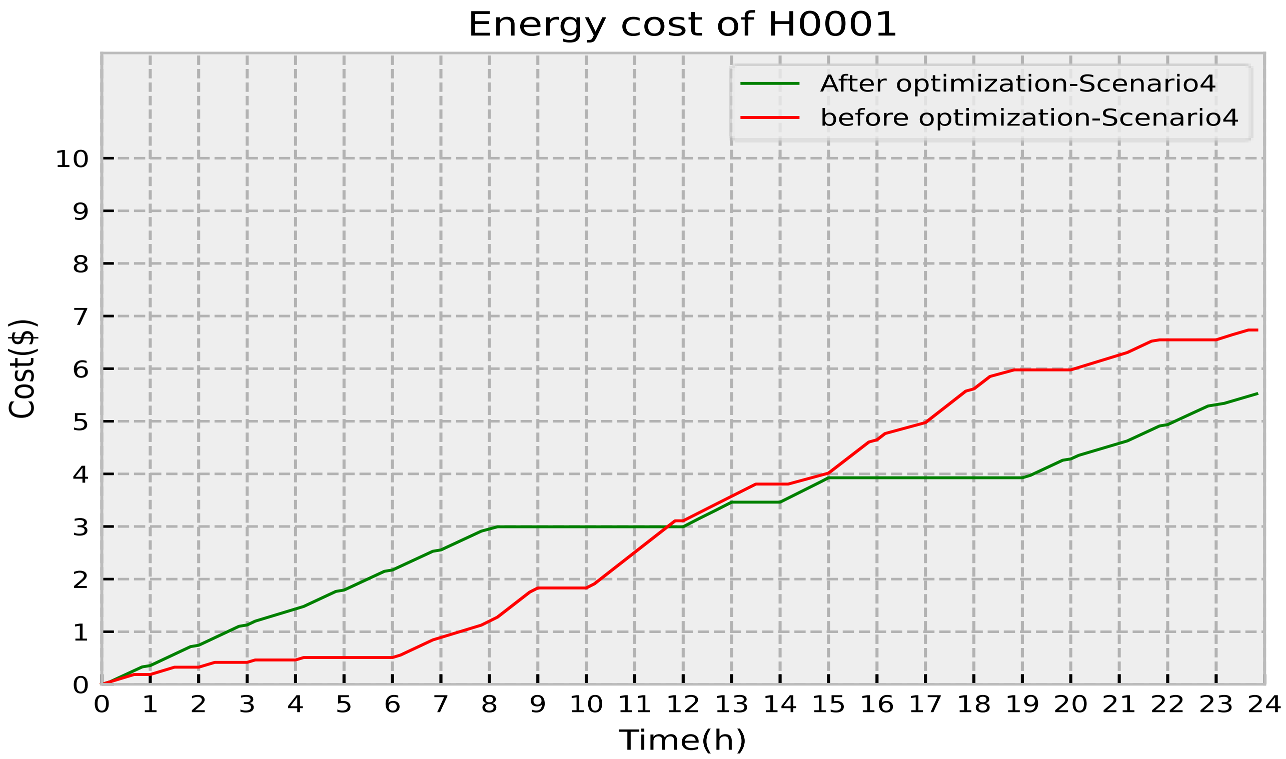


Figure 48-Energy cost comparison H0003-Scenario 4 vs. reference

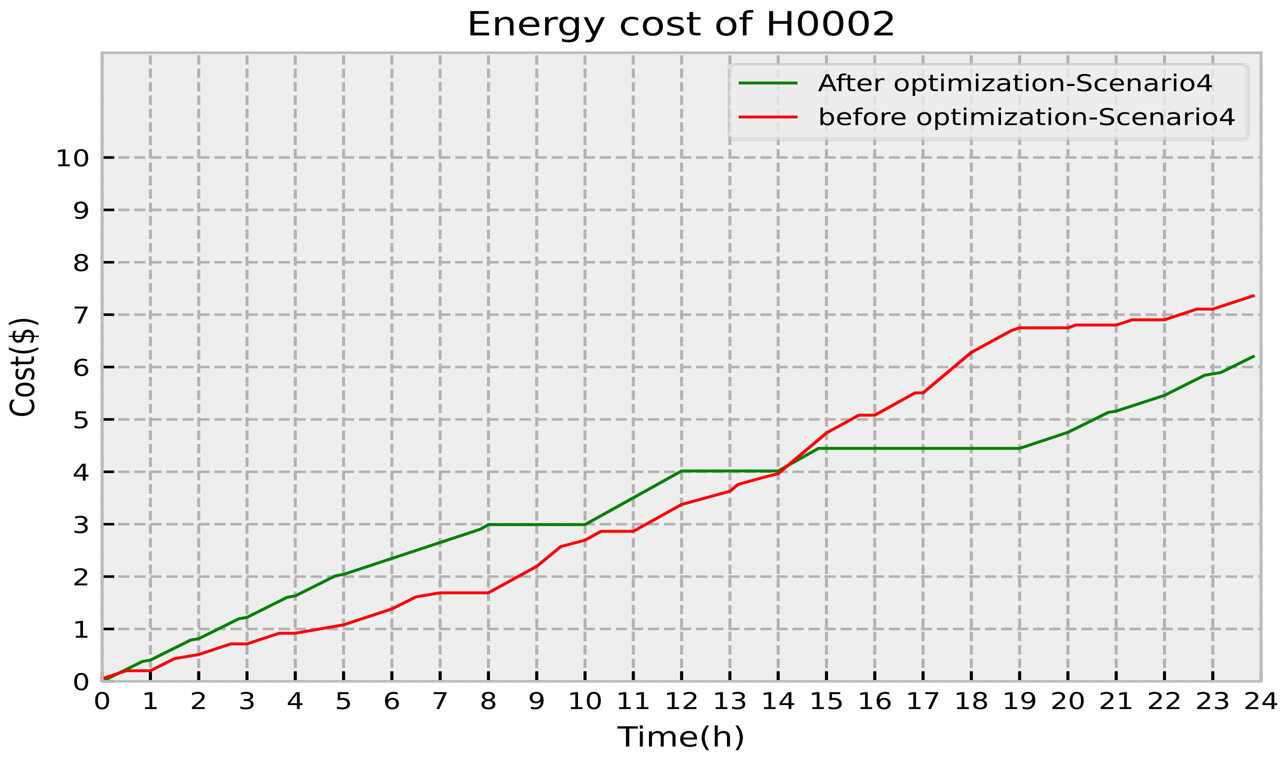


Figure 49-Energy cost comparison H0002-Scenario 4 vs. reference

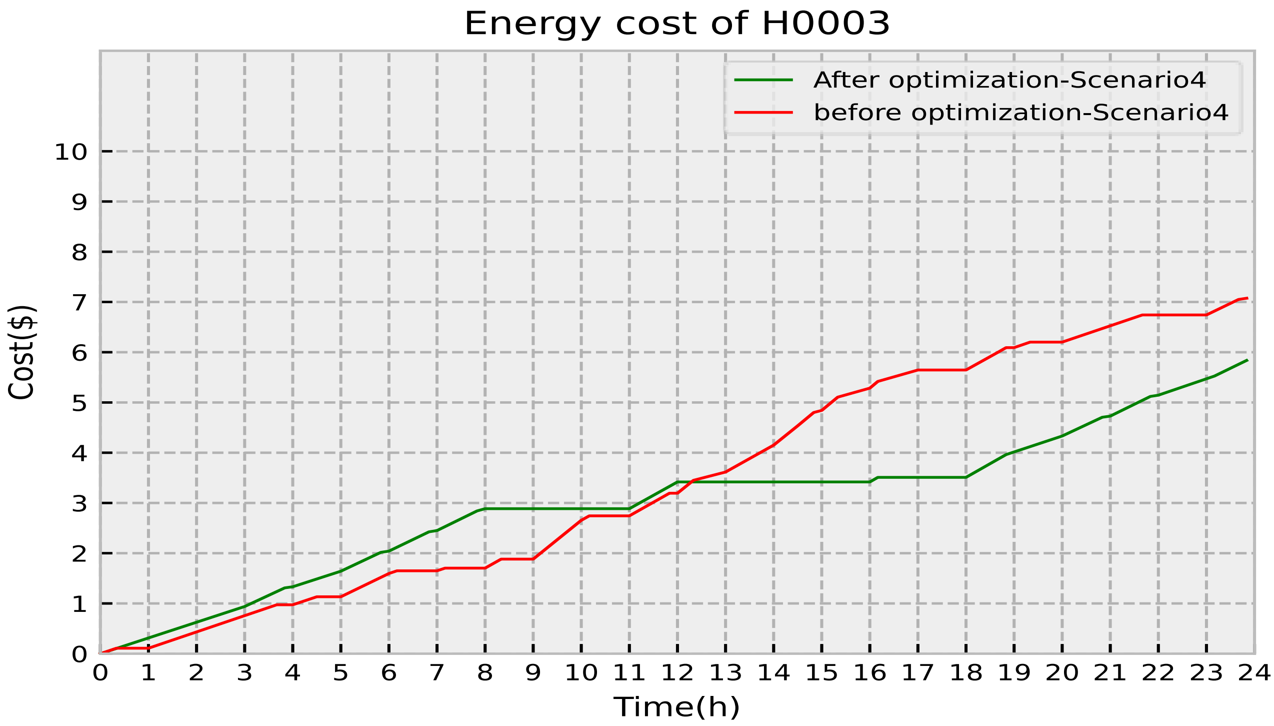


Figure 50-Energy cost comparison H0003-Scenario 4 vs. reference

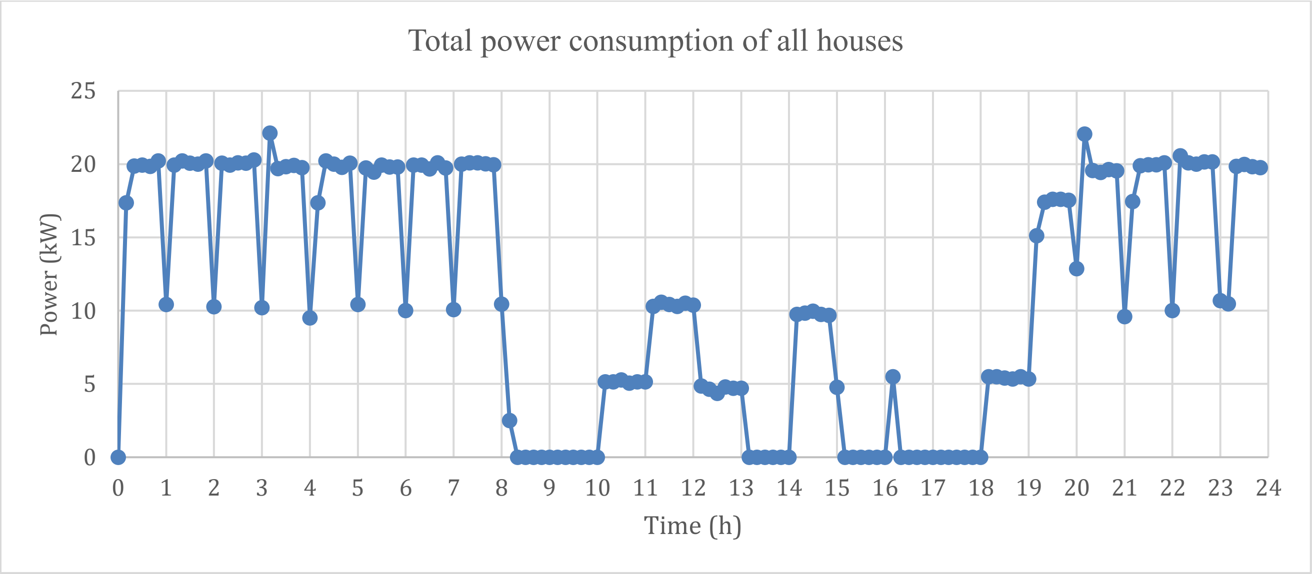


Figure 51-Hourly total power consumption of the three houses-the average power consumption at every hour remains below approximately 18 kW during the entire day.

5.6 Saved Costs in Scenarios 2 to 4

Table 15 compares the energy and cost values of Scenario 2 and the reference scenario, for all three houses in one day (a day is scaled down to 12 minutes). The price plan used was a tiered price plan (Table 14). As mentioned before, the reduction in cost was due to the deferrable loads switch OFF once the total energy consumption exceeded 40 kWh.

Table 16 compares the details of energy consumption in Scenario 3 and 4 with the reference scenario values. The price plan used was time-of-use (Table 14). effect of load shifting in decreasing energy cost depends on the formulation of the examples: the reference loads, the appliances power consumption, and load scales. For example, in Table 16it can be noticed that in Scenario 4 more energy was shifted in comparison with Scenario 3. This was because in Scenario 3 there was one deferrable load group and in Scenario 4 there were two appliances with deferrable loads which resulted in more possible load rescheduling.Recalling a constraint used in both Scenario 3 and 4, in which total energy consumption should have remained constant. However, in Table 16, total energy consumption of reference scenario, compared to Scenarios 3 or 4 had a slight difference. This difference was because of the accumulated effect of the variations in power consumption of the LEDs when they emulate the same power consumption; in other words, the total energy of the same scenario may be slightly different in two test runs. This can be considered as the effect of using physical devices (LED and smart plugs) for emulating appliance power consumption values instead of hypothetical values.

Table 14- Price plans

Energy price level	ToU	Tiered
Higher tariff	8:00 to 18:59 (0.1$/kwh)	Energy consumption more than 40kwh (0.09502 $/kwh)
Lower tariff	00:00 to 07:59 And 19:00 to 23:59 (0.06$/kwh)	Energy consumption up to 40 kwh (0.06159 $/kwh)

Table 15-Summary of scenario 2 results

House ID	Scenario Number	Tiered tariffs		Reduced energy (kWh)	Total daily energy consumption (kWh)	Total daily energy cost-Tiered price plan (C$)	~Energy cost saved (C$)
		Energy consumption calculated with low tariff (kWh)	Energy consumption calculated with high tariff (kWh)				
		0.06159 (C$/kWh)	0.09502 (C$/kWh))				
H0001	Reference Scenario	40	39.9	N/A	79.9	6.255	N/A
	Scenario 2	40	26.86	13.04	66.86	5.016	1.239
H0002	Reference Scenario	40	49.26	N/A	89.26	7.144	N/A
	Scenario 2	40	37.34	11.92	77.34	6.012	1.132
H0003	Reference Scenario	40	48.65	N/A	88.65	7.086	N/A
	Scenario 2	40	33.42	15.23	73.42	5.639	1.447

Table 16-Summary of Scenario 3 and 4 results

House ID	Scenario Number	ToU tariffs		Total shifted loads (kWh)	Total daily energy consumption (kWh)	Total daily energy cost-ToU price plan (C$)	~Cost saved (C$)
		Energy consumed at high tariff hours (kWh)	Energy consumed at low tariff hours (kWh)				
		0.1 (C$/kWh)	0.06 (C$/kWh)				
H0001	Reference Scenario	47.75	32.15	N/A	79.90	6.704	N/A
	Scenario 3	31.00	51.80	16.75	82.80	6.208	0.496
	Scenario 4	9.73	75.52	38.02	85.25	5.504	1.20
H0002	Reference Scenario	50.93	38.33	N/A	89.26	7.393	N/A
	Scenario 3	39.38	54.16	11.55	93.54	7.188	0.205
	Scenario 4	14.55	78.47	36.38	93.02	6.163	1.23
H0003	Reference Scenario	43.88	44.77	N/A	88.65	7.074	N/A
	Scenario 3	31.28	58.43	12.60	89.71	6.634	0.44
	Scenario 4	11.65	78.15	32.23	89.80	5.854	1.22

Chapter 6

6 Conclusion and Discussion

This project started with the overall goal of improving energy consumption and minimizing the cost in the residential sector, using IoT submetering via smart plugs. The project was successful at creating multiple smart homes energy management systems (SHEMS) with appliance level load monitoring using communication protocol, the hardware and software described in the previous chapters. The developed prototype successfully shifts the loads based on the electricity consumption, price plans and load types.

To validate the methodology adopted, the software framework and hardware devices were validated by using them to emulate a cluster of smart homes with EMSs. In addition, a commercially manufactured smart plug was also integrated into the framework to show the flexibility of the developed framework.

The existing smart plugs in the market don't allow arbitrary change in configuration of communication or measurement units. The custom designed plugs were developed using mostly open-source software and hardware to have full access to the hardware design, programming and to better understand their limitation and potentials. The developed framework is currently measuring DC-powered loads, and at present, the large majority of loads in a household are AC-powered. This is a limitation of the present system, however, since there are similarities with AC circuits (e.g., logic circuit, current sensor), with some modifications they can be turned to AC for measuring the appliances load. In addition to the custom smart plugs, the firmware of a commercial smart plug was flashed and changed to an open-source one. This represents the smart plugs in the market with either open-source firmware, or those that support a certain level of modification in their communication setup that leaves a possibility for integration with other software which are currently not common in the market. These two types of plugs, together show: (1) a combination of DC and AC measurements, and (2) hardware wise, a mixture of custom built and a commercially manufactured smart plug in one IoT architecture for monitoring and controlling the electricity consumption.

Four scenarios were defined to demonstrate the performance of the framework. In Scenario 1, using a database on the fog layer, the measurements of the smart plugs were saved during the internet or cloud service disconnection and synchronized with a cloud database after reconnection. Three energy management scenarios were defined to demonstrate the entire framework's performance in communication, load monitoring and shifting. The graphs in all scenarios for the three houses (all smart plugs) show that the smart plugs, smart homes, and the central server were able to successfully measure the loads, integrate, send and receive data packets without an effective delay. As a result, the energy cost was decreased in all houses. For example, in Scenario 4 we showed that the central energy management system (CEMS) residing in the cloud side, with benefiting from available information on the load patterns of the three houses in the cluster, can shift the loads to the hours with cheaper tariffs, without exceeding total power supply (transformer) limit.

6.1 Limitations and Future Works

SHEMS, IoT architectures and their associated devices such as smart plugs, in real applications have a wider range of functions, standards, and interactive interfaces. In this thesis, we were only able to focus on a portion of this realm and, therefore made certain assumptions for the areas that were not in the scope of the work. The limitations on this work are as follow:

1. The mechanism developed for saving data during a cloud/internet service outage only collects data of the smart plugs and cannot transfer packets from the central energy management system (CEMS) to the smart plugs during the same time. This means, energy management commands that are sourced from CEMS will not reach to the smart homes. Therefore, Scenarios 3 and 4 which are dependent on the hourly packets sent by CEMS cannot be implemented during internet disconnections. For controlling energy consumption or cost at such periods, depending on the algorithm, the local processing power of the Raspberry Pi can be used.
2. Smart plugs are assumed to be in synchronization with the function of the appliances. In other words, if a smart plug that is connected to a washing machine, is turned ON, the washing machine starts its washing cycles. This assumption is required to compensate for the lack of interoperability between the current smart devices and smart homes. How to make the smart devices with different user

interface and protocol to work with each other, is related to one of the most important challenges of IoT–Interoperability.

3. We assumed that the appliances are known to the system. Although, in reality the smart plugs need to detect the type of appliances that are connected to them. This function can be added with the existing microcontroller used in various way such as capturing appliances power consumption signature, using RFID tags or NFC devices. For signature detection the analog to digital converters (ADC) on ESP32 can capture sensor values with higher frequency (more than 1 kHz). For the commercial plug used in the research further manipulation in the firmware is required to access high frequency measurements (if the hardware supports). In this way, deferability of an appliance load can be determined, and its associated schedule or user preferences can be assigned to it.
4. For monitoring and controlling the appliances by a cloud server we assumed that the consumer already agreed with a certain level of appliance control and storage of data in a cloud server associated with a trusted party. Yet, addressing which party or organization should own the cloud database, is another research question that should be investigated.

6.2 Future work

The existence of a user interface enables a wide range of possible functions to be added to the framework, such as a new IoT device detection, authentication, and experimenting users' engagement with the SHEMS. Beside the technical part, consumer preferences and behavior are another major factors that play a big role in the capability of SHEMS in decreasing energy consumption and/or cost. A human-machine interface (HMI) can be used as a medium for involving or experimenting psychological factors to make SHEMS experience more attractive, and to unlock higher purchase rate and performance in energy efficiency. This along with many other economic, technical, and sociological factors need to be considered within academic studies for successful implementation of IoT within the urban environment.

Appendix

Appendix A-Scenario 2 graphs

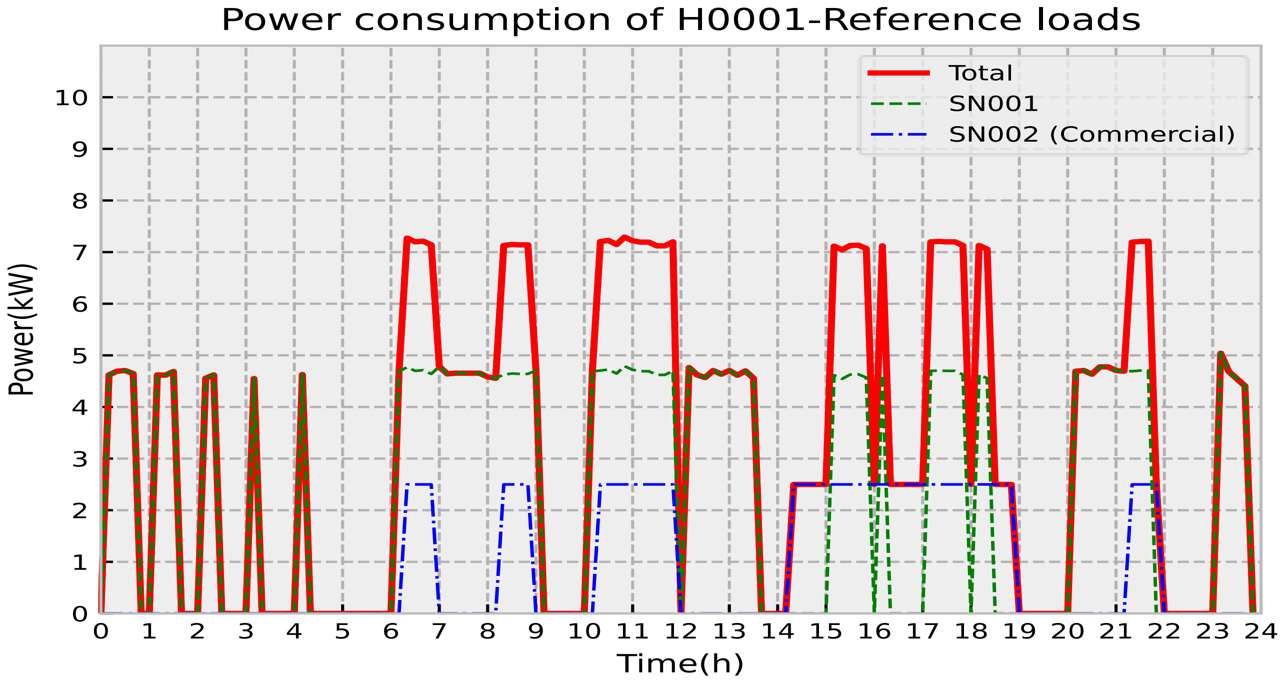


Figure 52-Measured power consumption-H0001- Reference Scenario

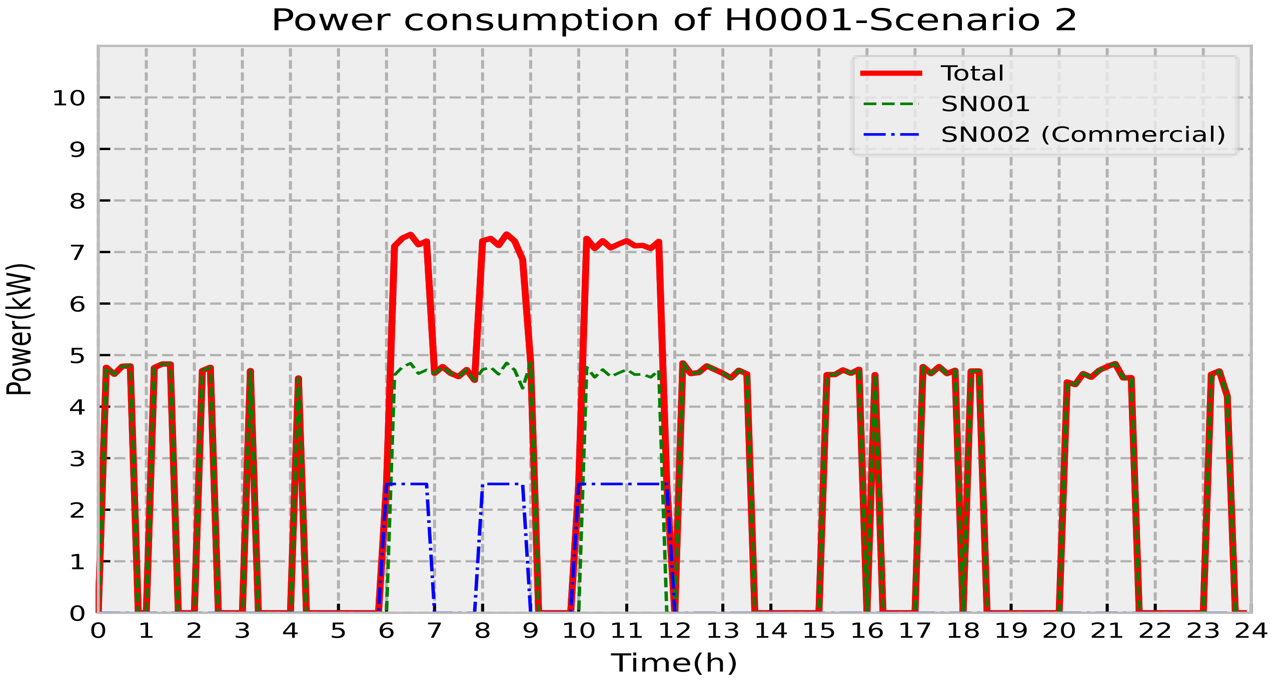


Figure 53-Measured power consumption-H0002-Scenario 2

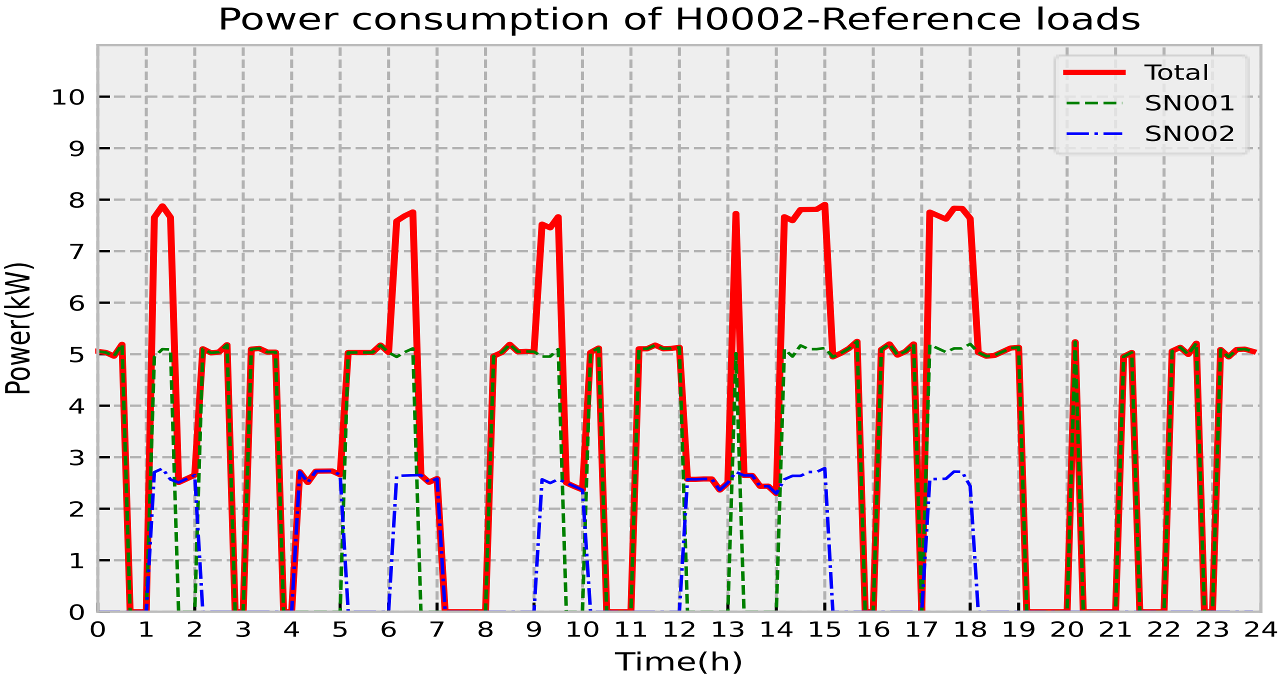


Figure 54-Measured power consumption-H0002-Reference

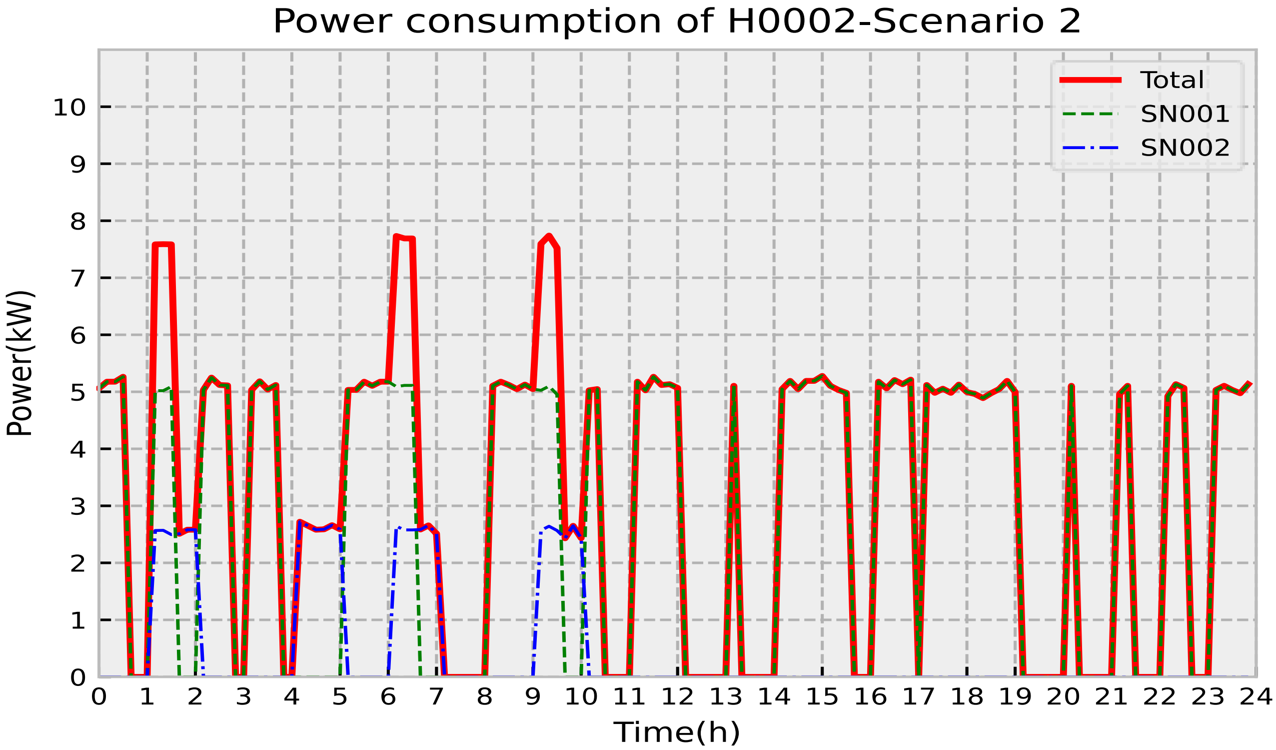


Figure 55-Measured power consumption-H0002-Scenario 2

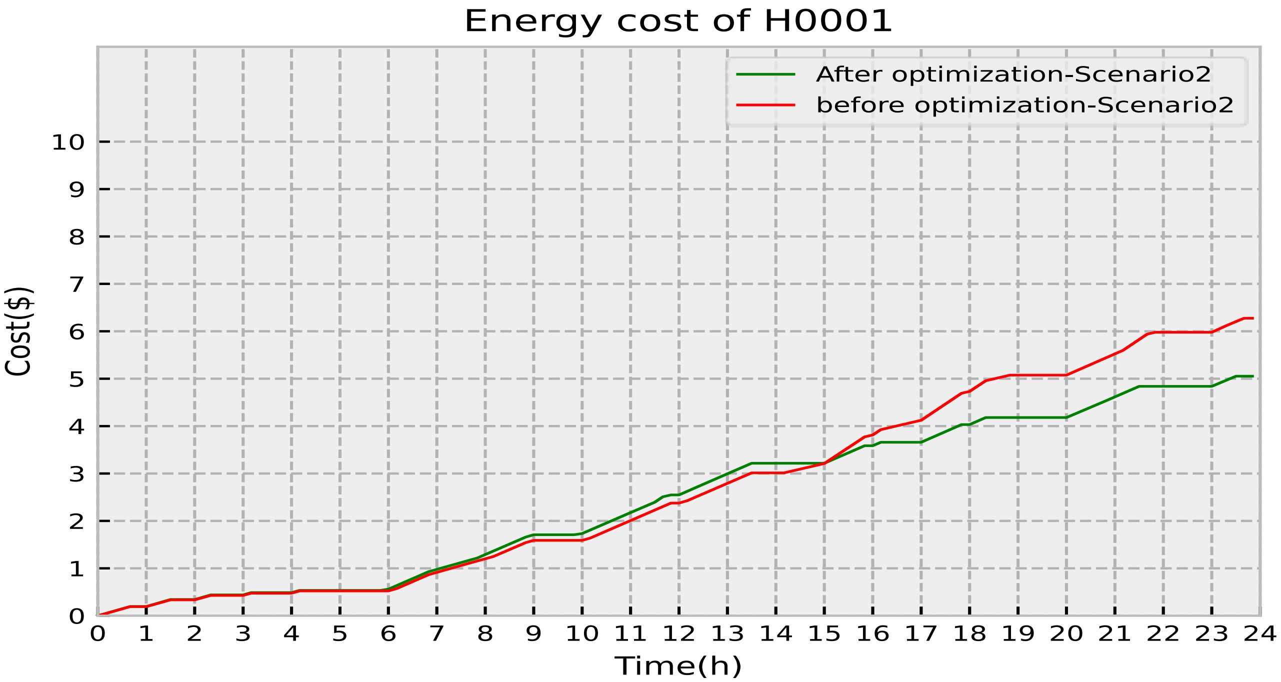


Figure 56-Energy cost comparison H0001-Scenario 2 vs. Reference

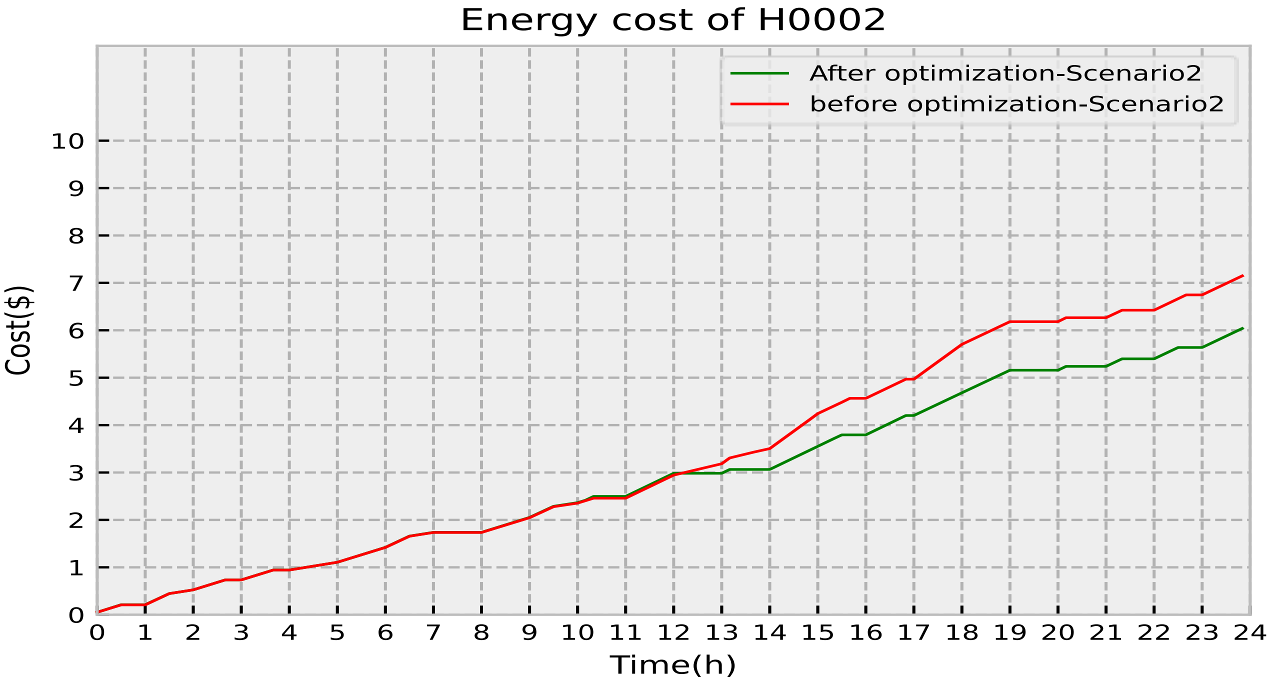


Figure 57-Energy cost comparison H0002-Scenario 2 vs. Reference

Appendix B-Scenario 3 graphs

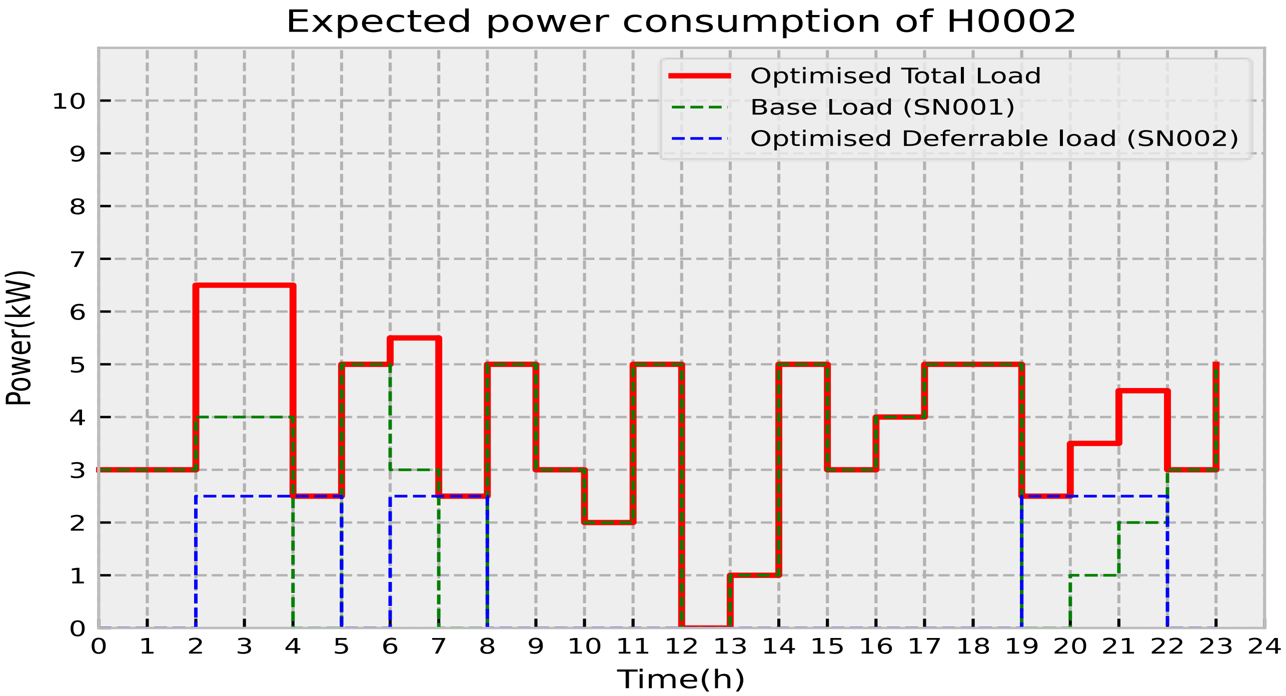


Figure 58-Expected power consumption-H0002-Scenario 3

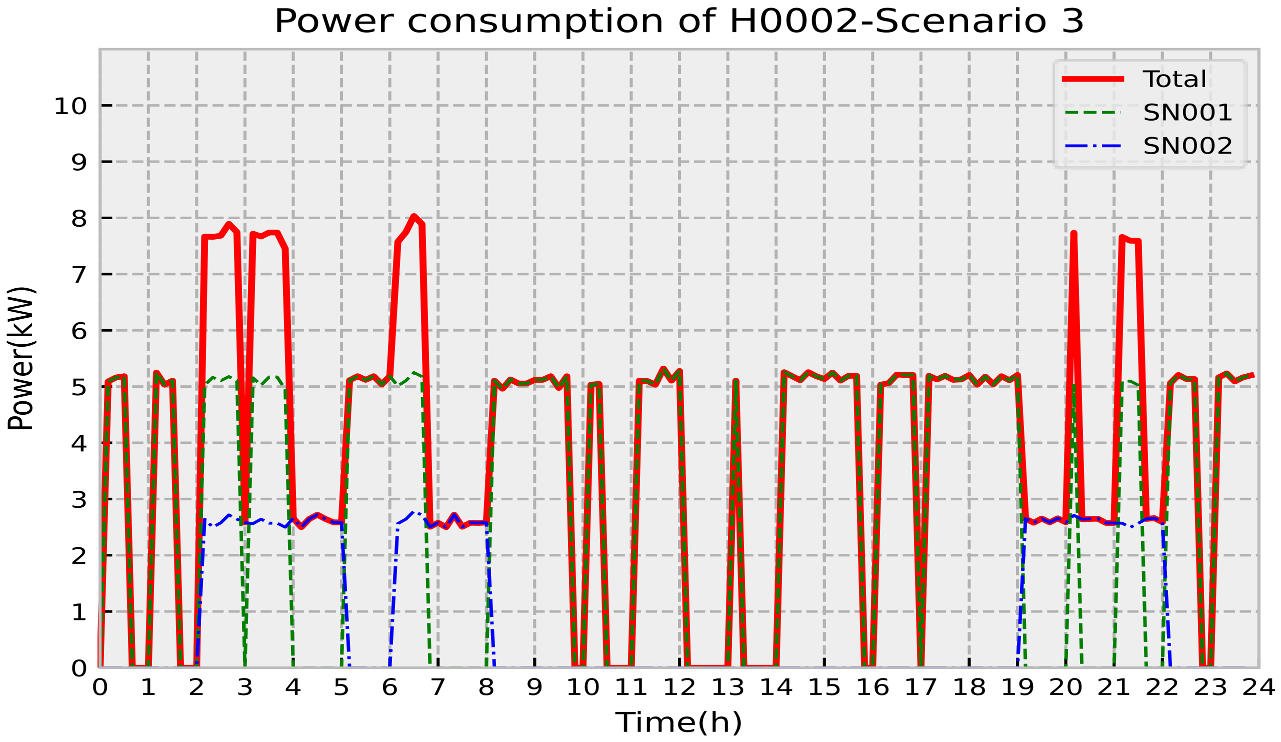


Figure 59-Measured power consumption-H0002-Scenario3

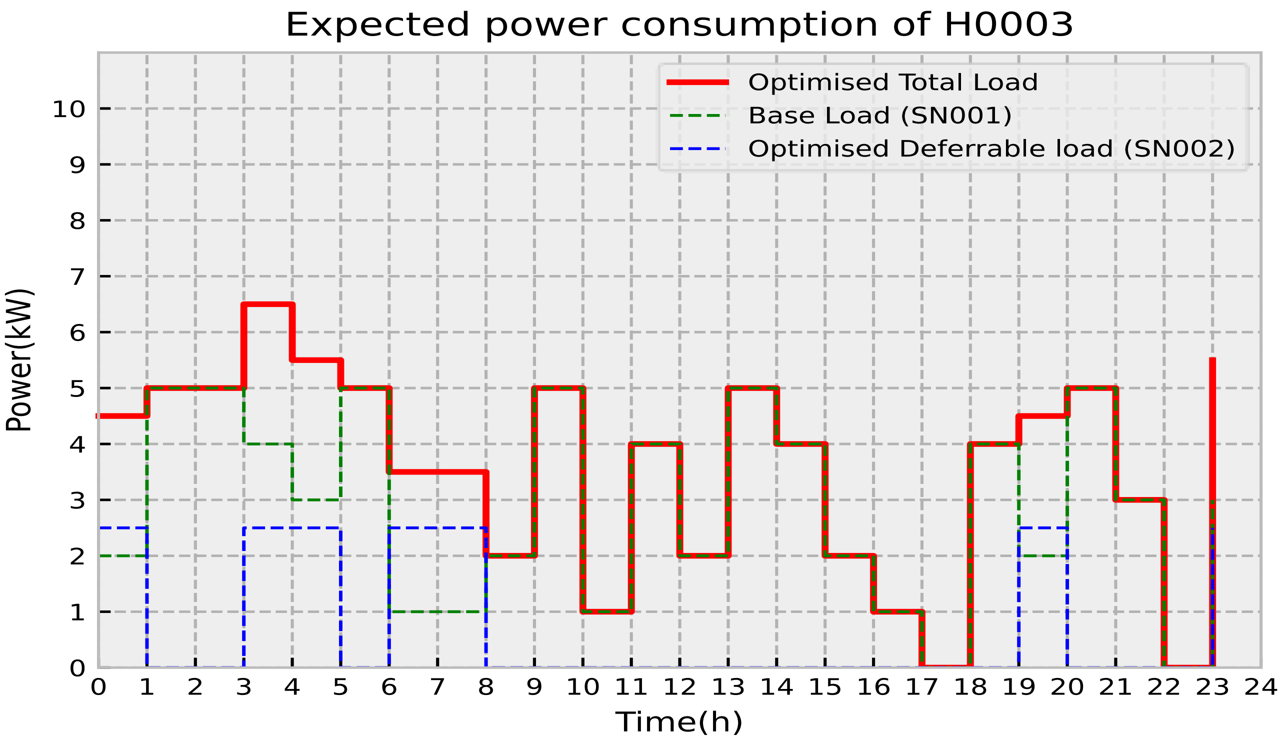


Figure 60-Expected power consumption-H0003-Scenario 3

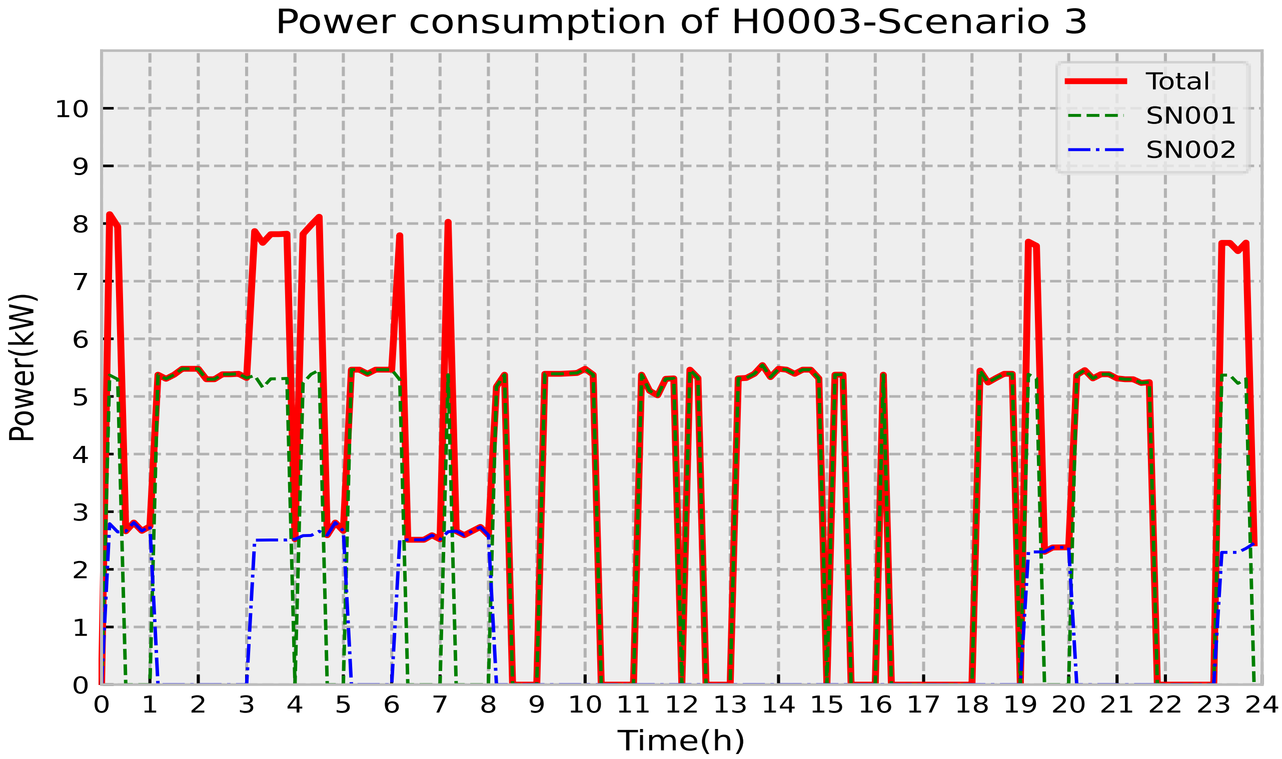


Figure 61-Measured power consumption-H0003-Scenario3

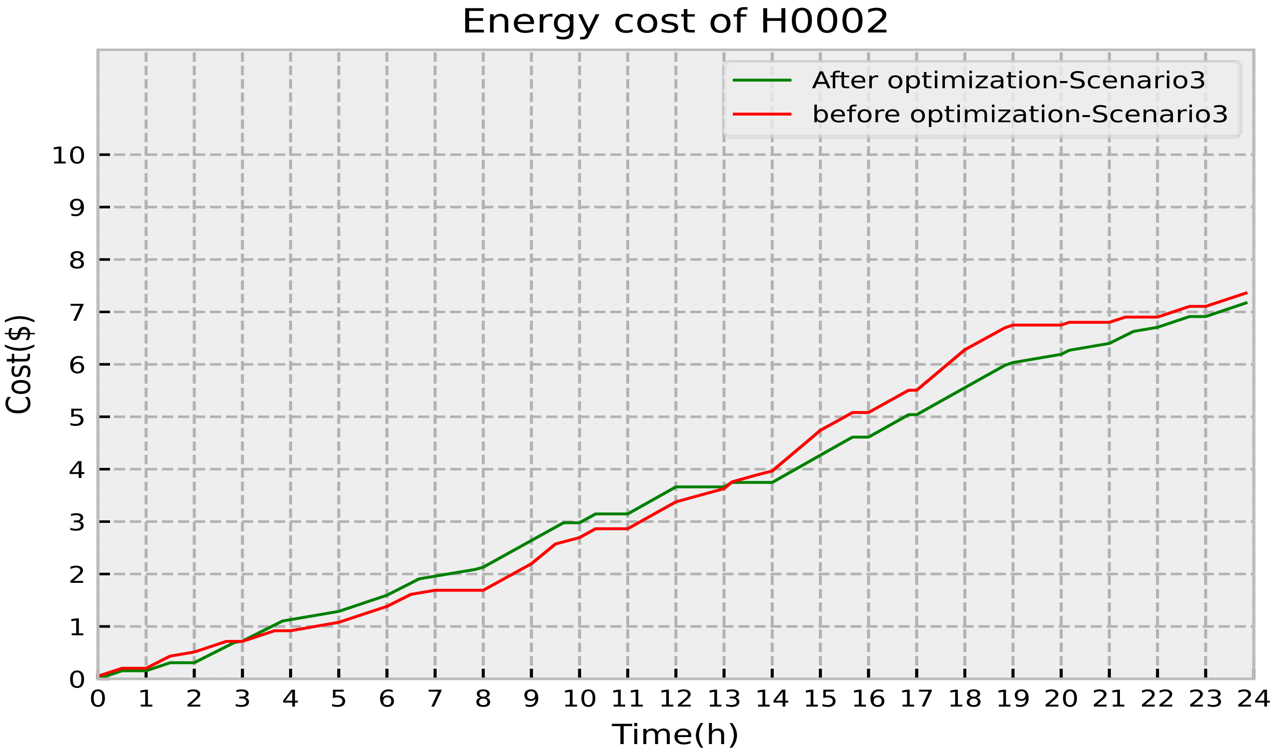


Figure 62- Energy cost comparison H0002-Scenario 3 vs. Reference

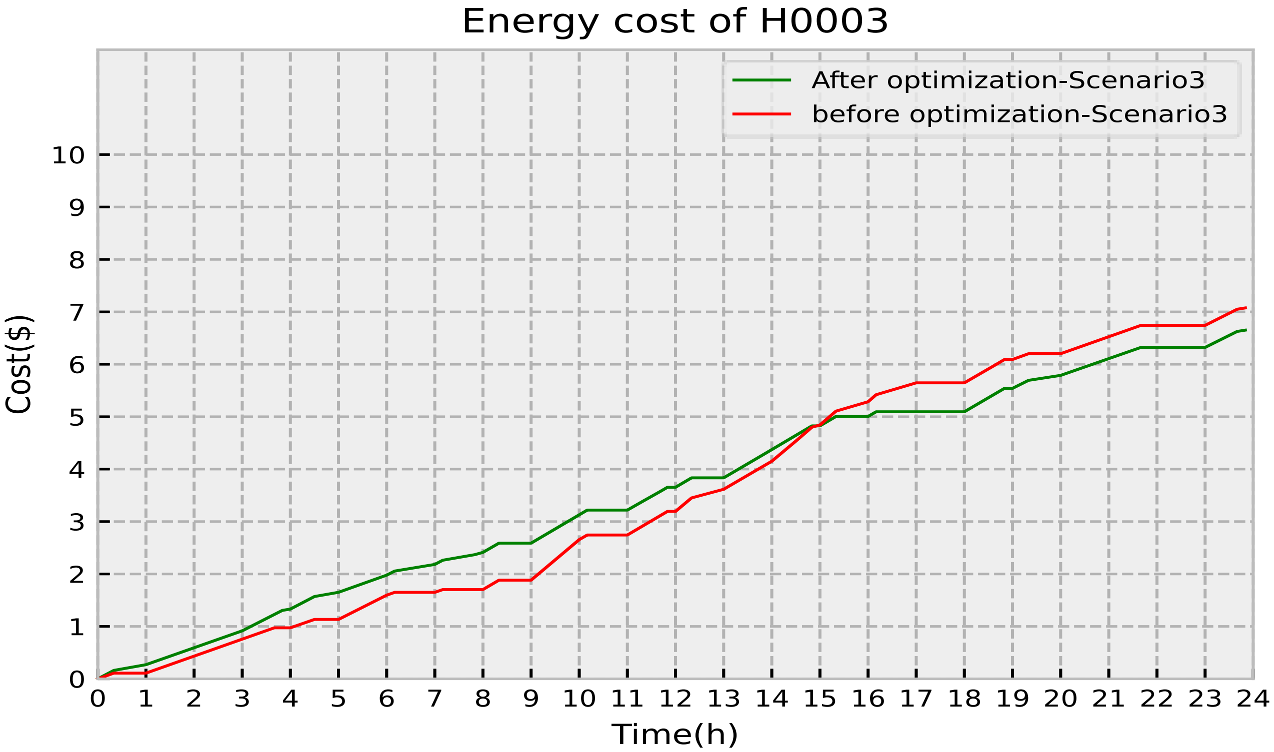


Figure 63-Energy cost comparison H0003-Scenario 3 vs. Reference

Bibliography

Afonso, J. A., Rodrigues, F., Pereira, P., Gonçalves, H., & Afonso, J. L. (2015). Wireless monitoring and management of energy consumption and power quality events. *Lecture Notes in Engineering and Computer Science*, *2217*, 338–343.

Ahmed, M. S., Mohamed, A., Homod, R. Z., Shareef, H., Sabry, A. H., & Bin Khalid, K. (2015). Smart plug prototype for monitoring electrical appliances in Home Energy Management System. *2015 IEEE Student Conference on Research and Development, SCOReD 2015*, 32–36. https://doi.org/10.1109/SCORED.2015.7449348

Ahsan, S. B., Yang, R., Noghabi, S. A., & Gupta, I. (2019). Home, safehome: Ensuring a safe and reliable home using the edge. *2nd USENIX Workshop on Hot Topics in Edge Computing, HotEdge 2019, Co-Located with USENIX ATC 2019*.

Al Faruque, M. A., & Vatanparvar, K. (2016). Energy Management-as-a-Service over Fog Computing Platform. *IEEE Internet of Things Journal*, *3*(2), 161–169. https://doi.org/10.1109/JIOT.2015.2471260

Anbya, M. F. B., Salehuddin, M., Hadisupadmo, S., & Leksono, E. (2012). Wireless sensor network for single phase electricity monitoring system via Zigbee protocol. *Proceedings of 2012 IEEE Conference on Control, Systems and Industrial Informatics, ICCSII 2012*, 261–266. https://doi.org/10.1109/CCSII.2012.6470512

Angrisani, L., Bonavolontà, F., Liccardo, A., Lo Moriello, R. S., & Serino, F. (2018). Smart power meters in augmented reality environment for electricity consumption awareness. *Energies*, *11*(9). https://doi.org/10.3390/en11092303

Baden, T., Chagas, A. M., Gage, G., Marzullo, T., Prieto-Godino, L. L., & Euler, T. (2015). Open Labware: 3-D Printing Your Own Lab Equipment. *PLoS Biology*, *13*(3), 1–12. https://doi.org/10.1371/journal.pbio.1002086

Balta-Ozkan, N., Boteler, B., & Amerighi, O. (2014). European smart home market development: Public views on technical and economic aspects across the United Kingdom, Germany and Italy. *Energy Research and Social Science*, *3*(C), 65–77.

https://doi.org/10.1016/j.erss.2014.07.007

Barker, S., Musthag, M., Irwin, D., & Shenoy, P. (2015). Non-intrusive load identification for smart outlets. *2014 IEEE International Conference on Smart Grid Communications, SmartGridComm 2014*, *October*, 548–553. https://doi.org/10.1109/SmartGridComm.2014.7007704

Chen, Y. Da, Azhari, M. Z., & Leu, J. S. (2018). Design and implementation of a power consumption management system for smart home over fog-cloud computing. *IGBSG 2018 - 2018 International Conference on Intelligent Green Building and Smart Grid*, 1–5. https://doi.org/10.1109/IGBSG.2018.8393553

Consortium, O., & Working, A. (2017). Open fog reference architecture for fog computing. *Open Fog Consortium Architecture Working Group*, *February*, 1–162.

Cook, D. J., Augusto, J. C., & Jakkula, V. R. (2009). Ambient intelligence: Technologies, applications, and opportunities. *Pervasive and Mobile Computing*, *5*(4), 277–298. https://doi.org/10.1016/j.pmcj.2009.04.001

Cooper, J., & James, A. (2009). Challenges for database management in the internet of things. *IETE Technical Review (Institution of Electronics and Telecommunication Engineers, India)*, *26*(5), 320–329. https://doi.org/10.4103/0256-4602.55275

Copie, A., Fortis, T. F., & Munteanu, V. I. (2013). Benchmarking cloud databases for the requirements of the Internet of Things. *Proceedings of the International Conference on Information Technology Interfaces, ITI*, 77–82. https://doi.org/10.2498/iti.2013.0546

Doan, T. T. (2018). *Smart Home with Resilience Against Cloud Disconnection*. https://doi.org/10.11575/PRISM/32955

Drouin, D., Droulers, G., Labalette, M., Lee Sang, B., Harvey-Collard, P., Souifi, A., Jeannot, S., Monfray, S., Pioro-Ladriere, M., & Ecoffey, S. (2017). A Fabrication Process for Emerging Nanoelectronic Devices Based on Oxide Tunnel Junctions. *Journal of Nanomaterials*, *2017*. https://doi.org/10.1155/2017/8613571

Eclipse Mosquitto | projects.eclipse.org. (n.d.). Retrieved November 28, 2021, from https://projects.eclipse.org/projects/iot.mosquitto

Electricity Pricing and Costs. (n.d.). Retrieved November 29, 2021, from https://www.hydroone.com/rates-and-billing/rates-and-charges/electricity-pricing-and-costs

Figueiredo, R. E., Monteiro, V., Ferreira, J. C., Afonso, J. L., & Afonso, J. A. (2021). Smart home power management system for electric vehicle battery charger and electrical appliance control. *International Transactions on Electrical Energy Systems*, *31*(4), 1–19. https://doi.org/10.1002/2050-7038.12812

Gill, A. Q., Behbood, V., Ramadan-Jradi, R., & Beydoun, G. (2017). *IoT architectural concerns*. 1–9. https://doi.org/10.1145/3018896.3025166

GitHub - ct-Open-Source/tuya-convert: A collection of scripts to flash Tuya IoT devices to alternative firmwares. (n.d.). Retrieved June 20, 2022, from https://github.com/ct-Open-Source/tuya-convert#requirements

Guimaraes, A. M. F., Freitas, T. T., Griner, H., & De Almeida, T. H. S. (2015). Smart energy monitoring system with ADE7758 IC. *IYCE 2015 - Proceedings: 2015 5th International Youth Conference on Energy*, 1–5. https://doi.org/10.1109/IYCE.2015.7180759

Kamel, E., & Memari, A. M. (2019). State-of-the-Art Review of Energy Smart Homes. *Journal of Architectural Engineering*, *25*(1), 03118001. https://doi.org/10.1061/(asce)ae.1943-5568.0000337

Le, T. T. H., & Kim, H. (2018). Non-intrusive load monitoring based on novel transient signal in household appliances with low sampling rate. *Energies*, *11*(12). https://doi.org/10.3390/en11123409

Lee, S. H., & Yang, C. S. (2017). An intelligent power monitoring and analysis system for distributed smart plugs sensor networks. *International Journal of Distributed Sensor Networks*, *13*(7). https://doi.org/10.1177/1550147717718462

Lorek, M. C., Chraim, F., & Pister, K. S. J. (2015). Plug-through energy monitor for plug load electrical devices. *2015 IEEE SENSORS - Proceedings*, 1–4. https://doi.org/10.1109/ICSENS.2015.7370666

Mongay Batalla, J., & Gonciarz, F. (n.d.). *Deployment of smart home management system at the edge: mechanisms and protocols*. https://doi.org/10.1007/s00521-018-3545-7

Node-RED. (n.d.). Retrieved December 8, 2021, from https://nodered.org/

Ozeer, U., Letondeur, L., Ottogalli, F. G., Salaun, G., & Vincent, J. M. (2019). Designing and implementing resilient IoT applications in the fog: A smart home use case. *Proceedings of the 2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops, ICIN 2019*, *Icin*, 230–232. https://doi.org/10.1109/ICIN.2019.8685909

Rahimi, M., Songhorabadi, M., & Kashani, M. H. (2020). Fog-based smart homes: A systematic review. *Journal of Network and Computer Applications*, *153*(July 2019), 102531. https://doi.org/10.1016/j.jnca.2020.102531

Rate D | Hydro-Québec. (n.d.). Retrieved November 29, 2021, from https://www.hydroquebec.com/residential/customer-space/rates/rate-d-billing.html

Rate Flex D | Hydro-Québec. (n.d.). Retrieved November 29, 2021, from https://www.hydroquebec.com/residential/customer-space/rates/rate-flex-d-billing.html

Ridi, A., Gisler, C., & Hennebert, J. (2014). A survey on intrusive load monitoring for appliance recognition. *Proceedings - International Conference on Pattern Recognition*, 3702–3707. https://doi.org/10.1109/ICPR.2014.636

Risteska Stojkoska, B. L., & Trivodaliev, K. V. (2017). A review of Internet of Things for smart home: Challenges and solutions. *Journal of Cleaner Production*, *140*, 1454–1464. https://doi.org/10.1016/J.JCLEPRO.2016.10.006

See, Y. C., & Jian Jing, S. (2019). Internet of Things (IoT) based Smart Wall Outlet. *ICRAIE 2019 - 4th International Conference and Workshops on Recent Advances and Innovations in Engineering: Thriving Technologies*, *2019*(November), 1–5. https://doi.org/10.1109/ICRAIE47735.2019.9037646

Smart technologies for plug-load identification, electrical submetering, and power-flow of future building-grids krishnanand kaippilly radhakrishnan. (2018).

Sohaib, S., Sarwar, I., Iftikhar, M. H., & Mahmood, A. (2016). A low cost smart energy monitoring and control system for smart buildings. *IET Conference Publications*, *2016*(CP694). https://doi.org/10.1049/cp.2016.0598

SQLite Home Page. (n.d.). Retrieved December 11, 2021, from https://www.sqlite.org/index.html

Sullivan, J. (2015). *"Fog" computing harnesses personal devices to speed wireless networks*. https://www.princeton.edu/news/2015/12/16/fog-computing-harnesses-personal-devices-speed-wireless-networks

Weranga, K. S. K., Kumarawadu, S., & Chandima, D. P. (2013). *Smart Metering Design and Applications (SpringerBriefs in Applied Sciences and Technology)*. http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20&path=ASIN/9814451819

Yokotani, T. (2016). Comparison with HTTP and MQTT on Required Network Resources for IoT. *2016 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC)*, 1–6. https://doi.org/10.1109/ICCEREC.2016.7814989

Žorić, A. Č., Bejtović, M., Žorić, N., & Jovanović, B. (2020). Improved Iot Energy Meter Applicable in Smart Buildings. *Unitech-Selectedpapers.Tugab.Bg*, *November*, 147–151. https://unitech-selectedpapers.tugab.bg/images/papers/2020/s2/s2_p53_v2.pdf