# IOT-BASED SMART SOCKET SYSTEM

LEOW PEI HUI

UNIVERSITI TUNKU ABDUL RAHMAN

# IOT-BASED SMART SOCKET SYSTEM

**Leow Pei Hui**

**A project report submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering (Honours) Electrical and Electronic Engineering**

**Lee Kong Chian Faculty of Engineering and Science**
**Universiti Tunku Abdul Rahman**

**May 2023**

## DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

| | | |
|---|---|---|
| Signature | : | [illegible] |
| Name | : | Leow Pei Hui |
| ID No. | : | 1802061 |
| Date | : | 16/05/23 |

## APPROVAL FOR SUBMISSION

I certify that this project report entitled **"IOT-BASED SMART SOCKET SYSTEM"** was prepared by **Leow Pei Hui** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Engineering (Honours) Electrical and Electronic Engineering at Universiti Tunku Abdul Rahman.

Approved by,

| | | |
|---|---|---|
| Signature | : | [illegible] |
| Supervisor | : | Ts Dr See Yuen Chark |
| Date | : | 16/05/23 |

| | | |
|---|---|---|
| Signature | : | |
| Co-Supervisor | : | |
| Date | : | |

## ABSTRACT

Electrical Wall Outlet is an essential device for supplying electricity to residential, commercial and industrial users. In light of the escalating demand for energy due to population expansion, minimizing wastage and overconsumption has become a pressing issue. Additionally, malfunctioning wall outlets could be one of the contributing factors to hazardous situations such as fires or electrocution. To address these issues, a Smart Socket System with remote control and real-time current measuring features has been proposed for residential and commercial/industrial settings. The ESP32 microprocessor serves as the system's core, controlling the ACS712-20A current sensor and relay module to enable smart socket functionality. A Node-red user interface (UI) is designed comprising MQTT protocol and Raspberry Pi microcomputer serving as the server. A Long Short-Term Memory (LSTM) AI Model was utilized to forecast hourly power consumption for the succeeding day, based on data collected and stored in Firebase Database. To enhance safety, a child-safe feature was added that notifies the user to switch off the socket remotely during no-load conditions. Additionally, an Over-the-Air (OTA) update was implemented to enable remote system debugging. In this project, the system's performance has been evaluated and found to be satisfactory.

# TABLE OF CONTENTS

## LIST OF TABLES

## LIST OF FIGURES

# LIST OF SYMBOLS / ABBREVIATIONS

| | |
|---|---|
| AC | Alternating Current |
| ADC | Analogue to Digital Converter |
| AI | Artificial Intelligence |
| ANN | Artificial Neural Network |
| API | Application Programming Interface |
| Arduino IDE | Arduino Integrated Development Environment |
| DAC | Digital to Analogue Converter |
| DC | Direct Current |
| ETOU | Enhanced Time-of-Use |
| HTTP | Hypertext Transfer Protocol |
| IoT | Internet of Things |
| IP | Internet Protocol |
| kWh | Kilowatt-hour |
| LSTM | Long Short-Term Memory |
| MQTT | Messaging Queuing Telemetry Transport |
| MAE | Mean Absolute Error |
| MSE | Mean Squared Error |
| NoSQL | Not only Structured Query Language |
| NTP | Network Time Protocol |
| OS | Operating System |
| OTA | Over-the-Air |
| PCB | Printed Circuit Board |
| RMSE | Root Mean Squared Error |
| RNN | Recurrent Neural Network |
| RPI | Raspberry Pi |
| SDK | Software Development Kit |
| SS | Smart Socket |
| SVG | Scalable Vector Graphics |
| TOU | Time-of-Use |
| UI | User Interface |
| Wh | Watt-hour |

## LIST OF APPENDICES

# CHAPTER 1

# INTRODUCTION

## 1.1 General Introduction

Electricity consumption has increased enormously throughout all these years due to the increase in population. According to The World Counts (2022), since 2000, global energy consumption has increased about three times, and it is predicted that in the year 2040, the energy consumption will increase to 740 million terajoules. Figure 1.1 shows the energy consumption from 1965 to 2020 in terawatts-hours (TWh). Hence, the utility has devised a solution, Time-of-Use (TOU) tariff to reduce the energy demand stress during peak hours. It is a method to calculate the energy consumed by customers depending on when the electricity is being used. Besides, it allows the customer to plan their electrical usage on either peak, mid-peak, or off-peak hours with the highest charge rate during Peak hours and vice versa for off-peak (Sunrun, 2021). Consumers are recommended to arrange their electricity use time according to the TOU standard. However, the TOU pricing scheme is only applied to industrial and commercial customers but not to residential customers in Malaysia (TNB, n.d.).

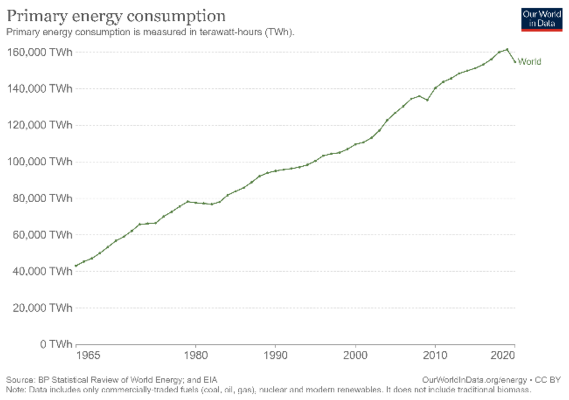


Figure 1.1: Global Energy Consumption Statistics (Ritchie et al., 2020)

Furthermore, the implementation of the Internet of Things (IoT) has increased over the years with the evolution of technology. A few wireless communication protocols used in IoT network systems, such as ZigBee, Wi-Fi, Z-Wave and Bluetooth enable communication between smart devices

(Manyinsa, 2021). According to Business Wire (2021), Smart Meter has been invested globally with the feature of monitoring the real-time energy demand of each consumer, balancing electric loads, and data sharing between utility and consumers. Moreover, Smart Plug was invented to monitor the voltage, and current of the electrical outlet, providing accessibility to the user to control the socket. Thus, a smart socket reduces the monthly electricity bill by cutting off standby power and preventing fire hazards or unforeseeable accidents (Asurion, 2022).

## 1.2 Problem Statement

Energy wastage is one of the problems to be solved by the utility. According to Winchester (2021), switching on plugged-in devices will still drain energy although the devices are not in use. Hence, people should switch off their sockets when not in use. Energy overconsumption will also lead to global warming when more generators are used to produce electricity, thus, releasing more carbon into the atmosphere (Rogers, 2020).

On the other hand, implementing a smart meter helps monitor utility-customer communication. According to Kirk (2021), the communication quality between utility and customers might be affected by the strength of the signal depending on the users' location which will cause delays throughout the data transmission such as electricity bill information. Moreover, rental houses share the same kWh meter between tenants, which causes unfairness to the tenants who use less electricity (Wahyudi et al., 2018). Hence, the Smart socket which consists of the power usage calculating feature, helps provide the individual used power tariff and solve these issues.

Electrical sockets are a common household feature people used in daily life. However, it poses a serious risk to children who are curious about everything around them. A socket without child-safe feature can make things hazardous as children may insert objects such as metal utensils or toys into it and cause serious injuries if the socket is activated with no-load connected. According to Majano (2021), about 2,400 children in the U.S. are injured by electrical outlets each year. From the survey, it is known that nearly one-third of parents are not aware that there are childproof sockets available and use plastic covers instead. Tragically, a two-year-old toddler was found dead after placing a spoon into a running electrical socket (Elicay, 2021). With the implementation of Smart Socket, no-load or excess current conditions can be detected and notify the user to prevent potential tragedies.

## 1.3 Aim and Objectives

The objectives of this project are to design and produce a smart socket with various functions such as remote control, OTA (Over the Air) update and real-time current measurement capabilities. Additionally, the system will utilize Artificial Intelligence (AI) technology to forecast the succeeding 24 hours power consumption and electricity bill. A server will be set up to connect multiple smart sockets, and each socket's calculated power usage will be stored in Cloud for load demand forecasting. Besides, a User Interface (UI) will be created for the end user to access real-time power usage (in kilo-watt hour) in terms of Malaysia's residential and commercial/industrial electricity pricing and tariff. ETOU tariff scheme for commercial and industrial users will also be deployed. Moreover, the UI will send warnings to users when sockets are left switched on under no-load condition.

## 1.4 Scope and Limitation of the Study

In this project, the studies involved are multiple Smart Sockets construction, AI forecasting model development and UI creation on the Node-red platform for status control, real-time monitoring of current, power and bill. Multiple standalone smart sockets will be constructed to simulate a realistic environment for research purposes. A server with MQ Telemetry Transport (MQTT) and Wi-Fi wireless communication protocol will be used to connect all the smart sockets for user accessibility. The real-time measured current will be calculated to obtain electricity tariff (kWh) based on users' selected tariff and sent to the Cloud for storing and monthly electricity bill forecasting using Long Short-Term Memory (LSTM) deep learning algorithm. The AI model will be trained and tested based on an open-source dataset. In addition, connection and data exchange between the Android UI Application and the Smart Sockets will also be established throughout this project.

# CHAPTER 2

# LITERATURE REVIEW

## 2.1 Introduction

In this chapter, smart socket implementation and development will be reviewed. The related work such as ETOU tariff rate and AI forecasting features will also be studied in various journals and research papers.

## 2.2 Smart Socket Operating System

Smart Socket operating system should be equipped with the capabilities of monitoring the electrical load as well as controlling it. Besides, from the literature related to the smart socket, the most-used components are sensors, transmitter/receiver and processor at the hardware aspect, database and coding at the software aspect and also the appliances to be investigated such as light/fans, fridge and also television (Rehman et al., 2020). According to Wahyudi et al. (2018), the smart socket implementation solves the rental house problem and enables the user to monitor the calculated electricity tariff per socket around the house. Figure 2.1 shows the operating system concept used. The ESP8266 microcontroller was used and it has a Wi-Fi feature for the connection to the internet. From Figure 2.1, the smart socket will retrieve the data from Cloud, calculate the total power usage, and display it to the user through the Android application. Besides, Arduino IDE software was used to load the code into the microcontroller, and ANTARES Cloud from Telkom Indonesia was used as Cloud storage.

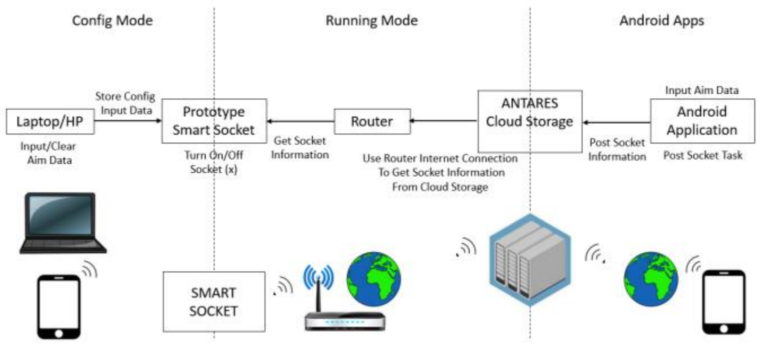


Figure 2.1: Working Principle of Smart Socket IoT System (Wahyudi et al., 2018)

Furthermore, another researcher, Sharmila et al. (2020), presented a 6IN1 Smart Socket system. The Smart socket has the capabilities to remote control the status of the socket via mobile app, time scheduling, implement voice control Artificial Intelligence (AI) and current monitoring feature to indicate over current/voltage across the socket. Figure 2.2 shows the block diagram of the whole system. Wi-Fi module ESP8266 was used as the communication protocol while the Arduino was used as the processor. LCD 16x2 indicator was also used to display real-time current and voltage to the user. When abnormal current and voltage are detected, a red LED will notify the user. Besides, the Time scheduling feature allows the user to set up a timer via a mobile app (Blynk) that automatically switches off the socket at a specific time. Easy VR commander software was used to train the voice control AI model.

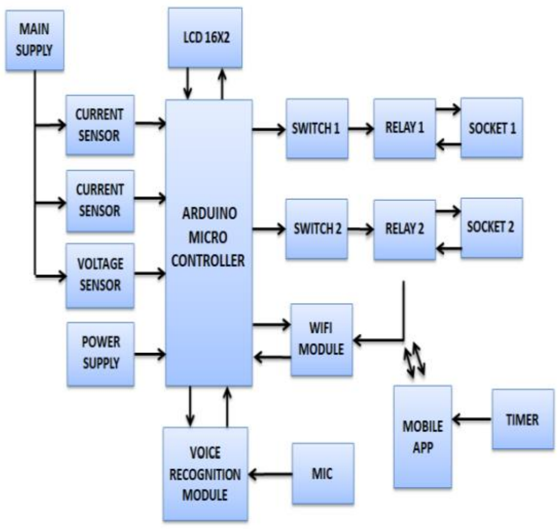


Figure 2.2: Block Diagram of Smart Socket (Sharmila et al., 2020)

However, according to Mateo et al. (2021), the literature proposed a smart socket (SS) that can be used in smart homes, factory automation systems and security systems. Besides, Bluetooth protocol instead of Wi-Fi protocol was chosen as a communication protocol between users and smart sockets. The system architecture is shown in Figure 2.3. The Smart Socket Hub is a server for multiple smart socket connections and provides security between the Cloud and the smart socket communication path. A Bluetooth-meshed system was used on each Smart Socket for communication between the sockets when one is out of range to control by the hub.

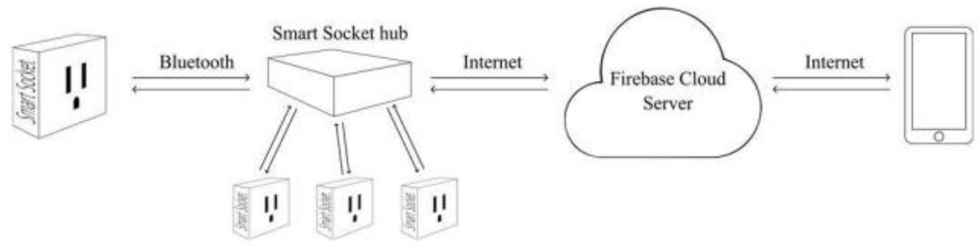


Figure 2.3: SS System Architecture (Mateo et al., 2021)

Moreover, Al-Hassan et al. (2018) have proposed more detailed work where ZigBee wireless protocol and Xbee (ZigBee Module) were used to communicate with the smart sockets. Arduino was used as a processor. When measuring the voltage from the socket, it is found that the voltage read might be too high to input to the Arduino ADC port. Hence, the voltage is stepped down to a suitable value before it is fed into the Arduino. Besides, since Arduino can only detect AC voltage from 0V to a maximum of 5V peak, a DC offset circuit is constructed after the voltage sensor to shift the negative voltage to the positive side, as shown in Figure 2.4. Furthermore, the ACS712 Hall effect-based current sensor with an error of ±1.5% was connected with the load to measure current real-time usage.

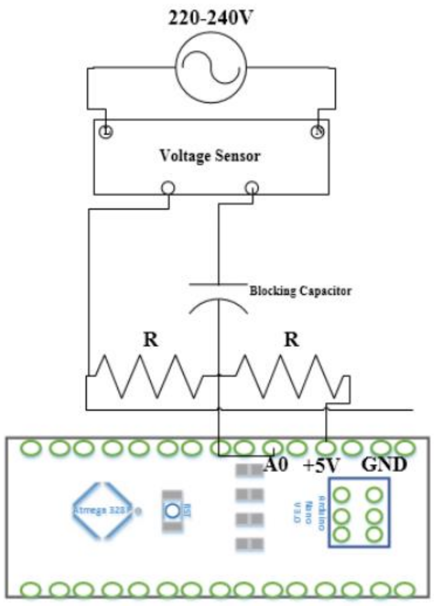


Figure 2.4: DC offset circuit (Al-Hassan et al., 2018)

Moreover, according to the research made by Blanco-Novoa et al. (2017), the Smart Plug implementation can save up to 70 Euros per year for some appliances. The proposed smart socket has two types of networks, an internal network and an external network. The internal network is used between smart sockets while the external network provides remote accessibility for the users. A gateway also existed between these two networks for protocol translation. Besides, the nodes (smart sockets) were connected in the form of star topology which operates in the master and slave concept. The master node will be in charge of sending the collected data to the gateway and then to the server while the slaves are to send their data to the master node as shown in Figure 2.5. The system can determine and change the master node between one another depending on the activation of the nodes. HTTP communication protocol will be used to communicate between each node after all the slave nodes are connected to the master node.

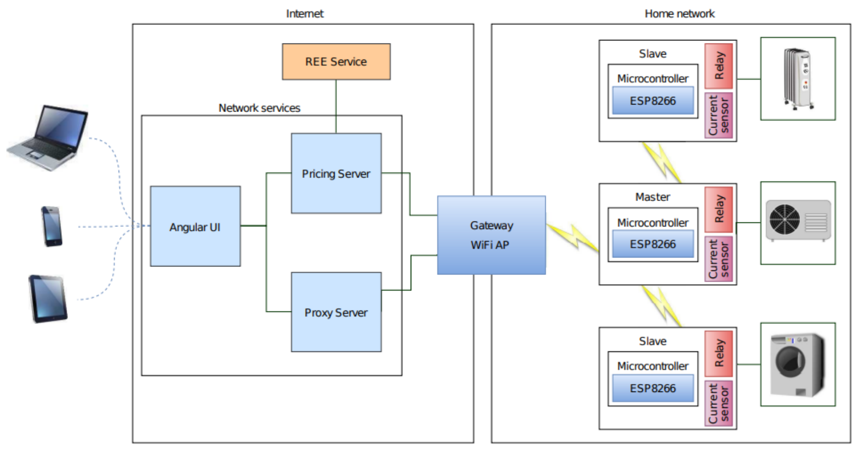


Figure 2.5: Components of the Smart Socket (Blanco-Novoa et al., 2017)

## 2.3 Wireless Communication Protocols for Smart Devices

The implementation of the Internet of Things (IoT) integrates different types of wireless communication protocols for data fetching. The speed and reliability of the protocol play an important role in the overall performance of the IoT system. Most smart homes use ZigBee, Bluetooth, Wi-Fi or Z-Wave as the communication protocol between smart devices. According to Naidu & Kumar (2019), research was conducted to compare Wi-Fi SON, Bluetooth, ZigBee, Z-Wave and Wi-Fi. Wi-Fi SON is an intelligence network that automatically selects and connects smart devices (Qualcomm, n.d.). From Naidu & Kumar’s

(2019) research, the speed of each wireless protocol has been concluded and plotted as shown in Figure 2.6. From the figure, Wi-Fi has the fastest speed compared to other smart home protocols and has the most extensive coverage, which is 100 m out of them. However, Wi-Fi requires high energy compared to other protocols while Bluetooth has the lowest power consumption among them.

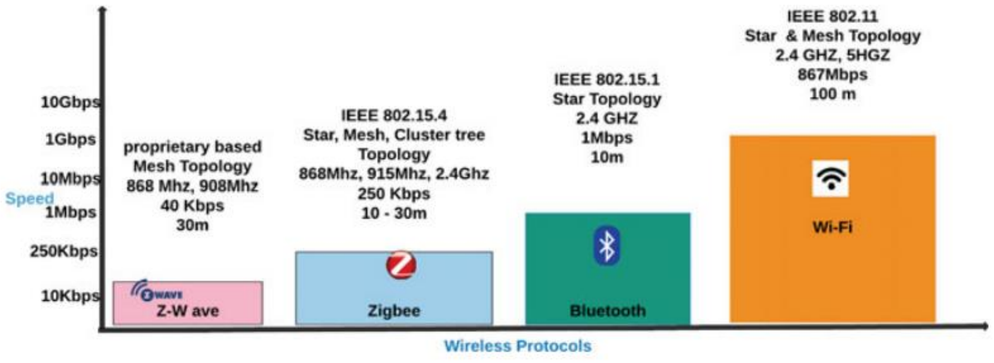


Figure 2.6: Wireless Communication Protocols Comparison (Naidu & Kumar 2019)

Besides, Lee et al. (2007) also researched the performance of each Bluetooth, Wi-Fi, ZigBee and UWB wireless protocol communication, comparing their data rate, distance and bandwidth. The summarized result can be seen in Table 2.0 below. From the table, Wi-Fi has the largest coverage range, and the highest bandwidth indicates that more data can be transferred throughout the transmission. However, Wi-Fi consumed energy the most compared to other protocols.

Table 2.0: Comparison of wireless communication Protocol (Lee et al., 2007)

| **Standard** | **Data Rate** | **Range (m)** | **Bandwidth** |
|---|---|---|---|
| **Bluetooth** | 1 Mb/s | 10 | 1 MHz |
| **UWB** | 110 Mb/s | 10 | 500 MHz – 7.5 GHz |
| **ZigBee** | 250 Kb/s | 10-100 | 2 MHz |
| **Wi-Fi** | 54 Mb/s | 100 | 22 MHz |

## 2.4 Deployment of Artificial Intelligence (AI) in Forecasting

In the olden days, forecasting data such as stock demand or electricity bills is done by a mathematical algorithm through the effort of humans. According to Sadownik & Barbosa (1999), the forecasting of industrial electricity consumption is done using a dynamics non-linear model and also an econometric model. Both of the methods use statistical ways to predict the electricity bill in the future using past data. Also, the forecasting methods are all done by humans instead of machine algorithms. Now, machine learning has become the interest of mankind due to its convenience and even the creation of Artificial Intelligence (AI) that is smart enough to think and make a decision like a human. A few types of machine learning models are used in forecasting such as Artificial Neural Network (ANN) and Long Short-Term Memory Neural Network (LSTM), which are the popular model for predicting. Besides, the machine learning model has better accuracy than traditional methods (Genpact, n.d.).

According to Chupong & Plangklang (2017), an application that is equipped with an electricity bill forecasting feature has been invented. The machine learning model will forecast the bill from the accumulation of power usage per day and notify the user every day with the latest forecasted result. From the research, it was found that the accuracy of the forecasted output is about 96%. On the other hand, Chen et al. (2001) studied the implementation of Artificial Neural Network (ANN) in the Short-Term Load Forecasting System (STLF) for the power system. Three layers of ANN were used in predicting the load usage and better accuracy was achieved. The architecture of the ANN model can be seen in Figure 2.7. According to Marr (2018), Artificial Neural Network is a mimic of the human brain since it consists of many layers. Thus, the network learns more every time the data passes through each layer. From Figure 2.7, the input layer is where input data to be learned or processed are fed. The input layer node is then connected to the hidden layer node, where the data is processed through functions to output the desired data at the output layer.

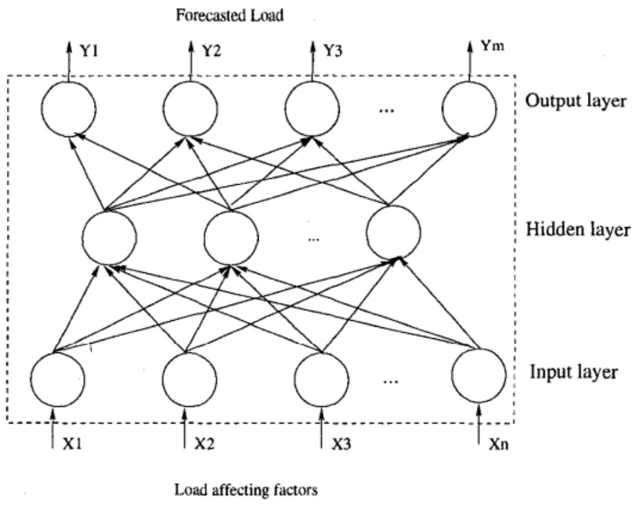


Figure 2.7: ANN model Architecture (Chen et al., 2001)

Besides, another deep neural network was also proposed by Shi et al. (2018). The research focused on household load demand prediction using a Recurrent Neural Network (RNN) whose architecture is shown in Figure 2.8. The architecture shows the looping happening at the hidden layer which is different from the feed-forward ANN model. Besides, each neuron depends on the previous time step. This sharing behaviour improved the RNN model learning process by referring to the previous time step learning outcome, especially in long-term dependencies data. However, if the time step is too long, the model will encounter gradient loss. Hence, the researcher introduced a Long Short-Term Memory (LSTM) cell to the RNN model to improve the lasting time of data flow across each time step. Therefore, the LSTM cell act as a memory cell for the whole system for input time series data with a long propagation time.

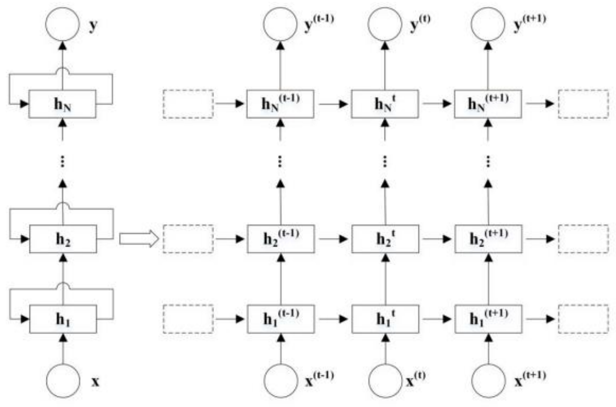


Figure 2.8: RNN model Architecture (Shi et al., 2018)

According to Hasanah et al. (2020), Recurrent Neural Network (RNN) and Vector Autoregressive (VAR) deep neural network has been used for hourly electric load demand forecasting. The comparison was done by investigating the Root Mean Square Error (RMSE) and the Mean Absolute Error (MAE) between the actual and predicted output result. The RMSE and MAE are commonly used evaluation metrics in deep learning models to measure the accuracy of predicted values compared to the actual values. Besides, the hidden neuron numbers were also varied to determine the suitable neuron to be used in the hidden layer of the RNN and VAR architecture. The neuron number with the lowest RMSE and MAE value was preferred. Moreover, this paper concludes that RNN performed better than the VAR model for short-term electric load prediction since the RMSE of RNN is about 40% lesser than the RMSE of VAR value.

According to Muzaffar & Afshari (2019), the researcher deployed the Long Short-Term Memory (LSTM) neural network for power station load forecasting. LSTM deep learning algorithm is a type of RNN neural network designed to address the limitations of RNN or ANN models when dealing with long-term dependencies in sequential data. The study compared the performance of the LSTM deep learning model with traditional statistical models: ARMA, SARIMA and ARMAX in forecasting load for the next 24 hours, 48 hours, 7 days and 30 days accordingly. The researchers computed that the LSTM model provided more consistent and accurate predictions over time and was easier to handle compared to the traditional statistical models in which complex equation is required.

## 2.5 Smart Socket with Time-of-Use (TOU) Scheme

Time-of-Use (TOU) pricing is the electrical tariff pricing based on power usage during peak and off-peak times while Enhanced Time-of-Use (ETOU) has extra mid-peak period allocation compared to the TOU scheme (Sunrun, 2021). According to Tsai et al. (2016), a Residence Energy Control System (RECos) has been established which integrates the following module with a smart socket: peak-time control (PTC), energy-limit control (ELC), automatic control (AC) and user control (UC). The PTC mode enables users to self-define the peak-time power consumption limit. Once the limit is exceeded, the low-priority smart socket defined by the user will switch off first with a warning notifying the user.

Furthermore, the ELC mode will be activated when the user-defined maximum total energy usage of all smart sockets is for about 4 weeks or more. This mode splits the energy usage of each smart socket depending on the past power usage and switches off the power supply when the energy limit is exceeded. The AC and UC module is for the on/off status of the smart socket. AC mode controls the power supply of the socket automatically according to the PTC or ELC signal received. However, UC indicates manual control by the user via the user interface. The whole system operation can be seen in Figure 2.9.

On the other hand, the researchers also proved that energy consumption could be greatly reduced after implementing the energy control mechanism via the RECos system. The power consumption of a water dispenser was recorded in Figure 2.10 and it was found that its energy consumption of it has dropped to 0 kWh due to the automatic switch-off control system.

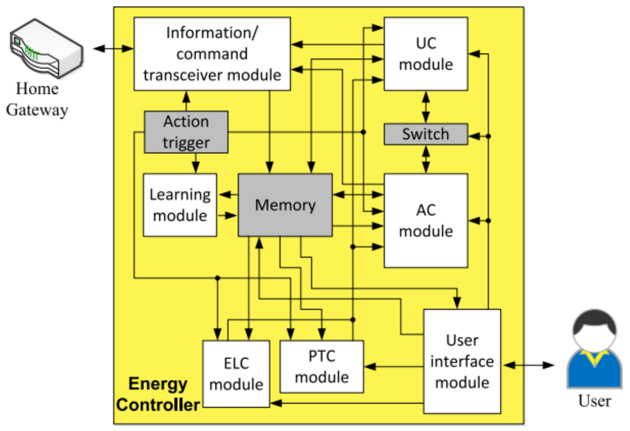


Figure 2.9: RECos System Operation (Tsai et al., 2016)

According to the proposed improved smart socket of Al-Hassan et al. (2018), the TOU scheme was integrated into controlling lighting dimming and refrigerator operation depending on the lux and the temperature respectively. The TOU is then defined according to user preference and the applications were worked depending on the TOU scheduling and algorithms. The LED light will dim or turn off in terms of percentage depending on the luminance limit during peak hours. However, the refrigerator will be turned off during peak hours only when the fridge temperature is within the suitable range. Besides, the paper also concludes that 0.811 kWmin and 0.056$ energy have been reduced with the implementation of TOU scheduling in the smart socket.

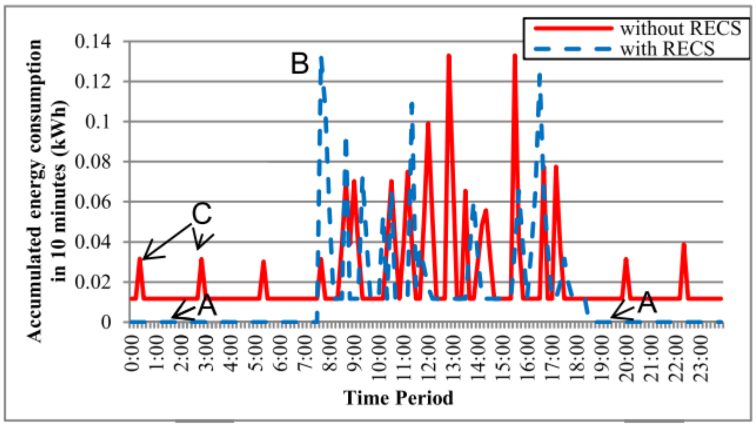


Figure 2.10: Power Consumption of a Water Dispenser (Tsai et al., 2016)

## 2.6 Summary

This chapter studied and reviewed past research on the construction of a smart socket IoT system. Each researcher has their method to construct the system by integrating different communication protocols, Cloud or database, and even an AI model for load demand prediction. Various deep-learning models were examined and compared using metrics like RMSE, MAE, MSE, and others. Besides, the researchers also conclude that energy consumption can be reduced via the features of smart sockets: Remote on/off the socket and TOU scheduling features. Therefore, an actual prototype of a smart socket will be constructed in this project to study its performance practice.

# CHAPTER 3

# METHODOLOGY AND WORK PLAN

## 3.1 Introduction

This chapter provides a detailed discussion of the hardware and software components needed to build a Smart Socket system as well as the implementation approach.

## 3.2 System Overview

This project involves setting up two Smart Sockets (SS) with ESP32 controllers each serving as a standalone node with relays to control sockets on/off and current sensors for current readings. The architecture of the project is presented in Figure 3.1. The ESP32 controllers will be programmed using Arduino IDE software. The data exchange between the Raspberry Pi Server and ESP32 will utilize the MQTT protocol (MQ Telemetry Transport). Node-red and Mosquitto MQTT broker will be integrated into the RPI for MQTT establishment. Cloud database (Firebase) will be used for data storage through Node-red data fetching. Data stored in Firebase will be used as input datasets for AI model prediction. Additionally, the OTA update feature will be integrated to improve the system's user-friendliness by performing remote system debugging. The following sub-section will provide more detail on the proposed work.

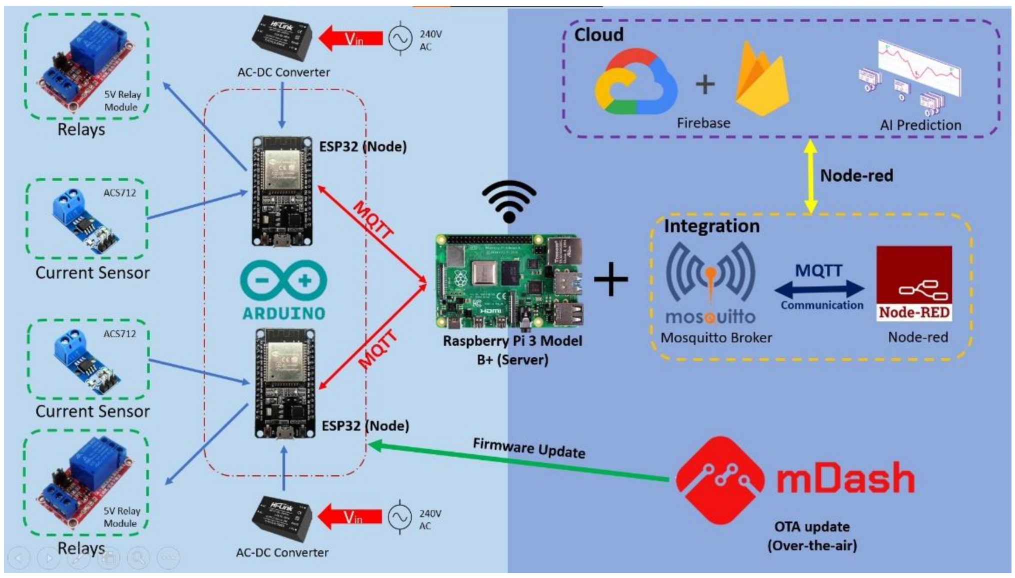


Figure 3.1: Smart Socket (SS) system Architecture

## 3.3 Hardware Description

This section states the detail of the hardware to be used in the Smart Socket system.

### 3.3.1 ESP32 DevKit V1

The ESP32 microcontroller is a development board with built-in Wi-Fi and Bluetooth wireless technologies for connection. It consists of digital pins to input and output digital signal, ADC/DAC and capacitive touch pins. In this project, the controller acts as a core of the smart socket which receives a signal to control the on/off of the socket and sends the sensed current sensor data to the server. It is connected to the Raspberry Pi through the MQTT protocol. Besides, the controller has an operating voltage of 5 $V_{dc}$ and 3.3 $V_{dc}$ for each pin; hence, high AC voltage should be regulated before being fed into the ESP32. The Pin Layout and its specification can be seen in Figure 3.2.

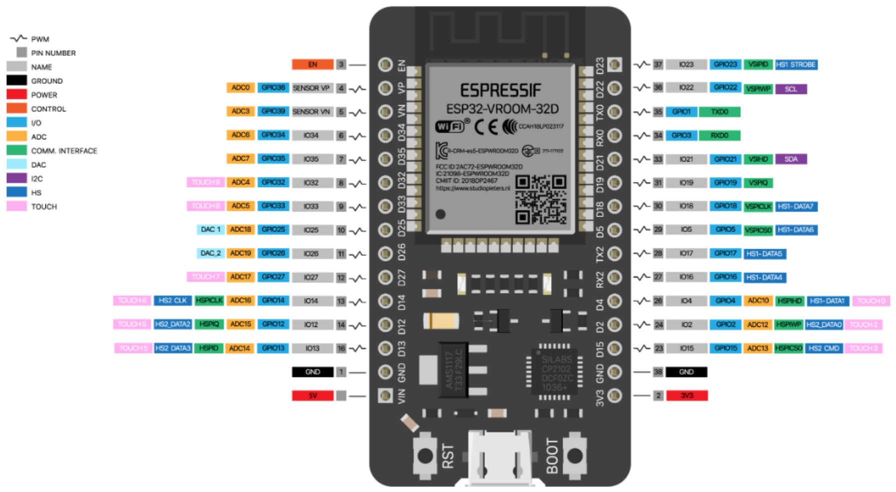

PWM
PIN NUMBER
NAME
GROUND
POWER
CONTROL
I/O
ADC
COMM. INTERFACE
DAC
I2C
HS
TOUCH
EN 3 EN
ADC0 GPIO36 SENSOR VP 4 VP
ADC3 GPIO39 SENSOR VN 5 VN
ADC6 GPIO34 IO34 6 D34
ADC7 GPIO35 IO35 7 D35
TOUCH9 ADC4 GPIO32 IO32 8 D32
TOUCH8 ADC5 GPIO33 IO33 9 D33
DAC 1 ADC18 GPIO25 IO25 10 D25
DAC_2 ADC19 GPIO26 IO26 11 D26
TOUCH7 ADC17 GPIO27 IO27 12 D27
TOUCH6 HS2 CLK HSPICLK ADC16 GPIO14 IO14 13 D14
TOUCH5 HS2_DATA2 HSPIQ ADC15 GPIO12 IO12 14 D12
TOUCH5 HS2 DATA3 HSPID ADC14 GPIO13 IO13 16 D13
GND 1 GND
5V VIN
D23 37 IO23 GPIO23 VSPID HS1 STROBE
D22 36 IO22 GPIO22 VSPIWP SCL
TX0 35 GPIO1 TXD0
RX0 34 GPIO3 RXD0
D21 33 IO21 GPIO21 VSIHD SDA
D19 31 IO19 GPIO19 VSPIQ
D18 30 IO18 GPIO18 VSPICLK HS1-DATA7
D5 29 IO5 GPIO5 VSPICS0 HS1-DATA6
TX2 28 IO17 GPIO17 HS1-DATA5
RX2 27 IO16 GPIO16 HS1-DATA4
D4 26 IO4 GPIO4 ADC10 HSPIHD HS1-DATA1 TOUCH0
D2 24 IO2 GPIO2 ADC12 HSPIWP HS2_DATA0 TOUCH2
D15 23 IO15 GPIO15 ADC13 HSPICS0 HS2 CMD TOUCH3
GND 38 GND
3V3 2 3V3
ESPRESSIF
ESP32-VROOM-32D
RST
BOOT

Figure 3.2: Pinout and Specification of ESP32 (Pieters, 2022)

### 3.3.2 Raspberry Pi 3 Model B+

Raspberry Pi 3 Model B+ (RPI) is a microcomputer featuring a quad-core processor clocked at 1.4 GHz and 1GB RAM. Besides, 16GB ROM is set up and it runs on the Debian-based Raspbian (Bullseye) operating system. The RPI serves as a server utilizing the MQTT protocol to connect all the smart sockets acting as the MQTT broker, while the ESP32 functions as the client. It exchanges messages with the ESP32 to control the smart socket's operation.

Besides, it employs node-red to develop the IoT system. The structure of the Raspberry Pi 3 can be seen in Figure 3.3.

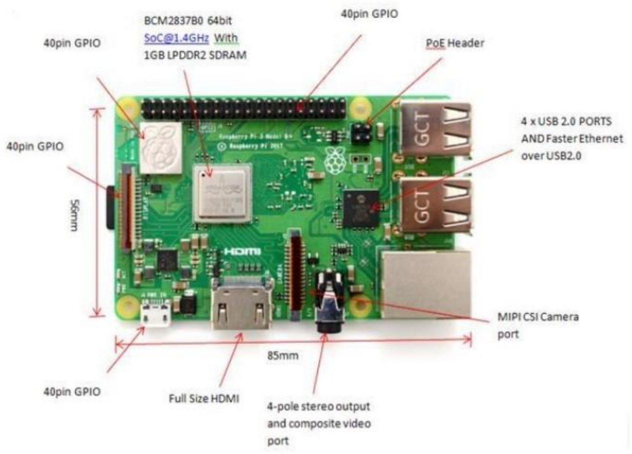


Figure 3.3: Raspberry Pi 3 B+ Structure (Kumbhar et al., 2018)

## 3.3.3 ACS712 Current Sensor (ACS712ELCTR-20A-T)

ACS712 is a Hall Effect-Based Linear Current Sensor that measures both AC and DC current. This sensor detects changes in the magnetic field flowing across it with a sensitivity of 66-185 mV/A. Model ACS712ELCTR-20A-T with a sensitivity of 100 mV/A and rated for a maximum current of 20A will be used in this project. The sensor will be operated at a voltage of 5 V, while the analogue output voltage pin, $V_{IOUT}$, will provide a base value of 2.5 V when no current is flowing. The base value serves as a reference point for the current measured across the ±IP pins. The $V_{IOUT}$ pin will be connected to the ESP32 controller to obtain the current readings.

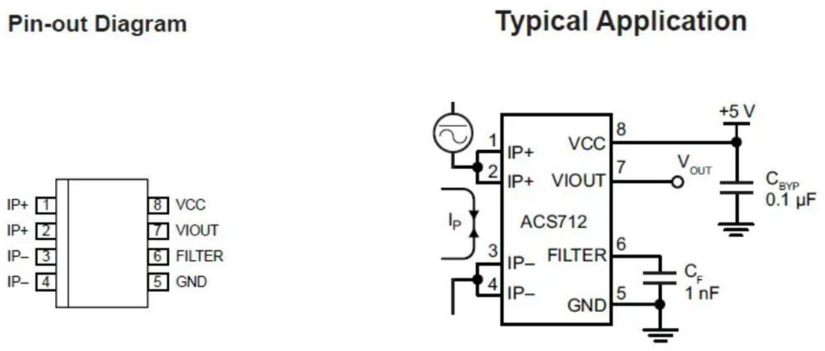


Figure 3.4: ACS712 Pin Diagram and Connection (EG Projects, n.d.)

### 3.3.4 Single Channel 5V Optocoupler Relay Module

The 5V relay module was used to switch on/off the AC supply to the load socket. The optocoupler at the low voltage side controls the high voltage side via a switching transistor, providing safe isolation between the two sides. Depending on the connection at the output terminal, the switching can be in normally open (NO) or normally closed (NC) mode as illustrated in Figure 3.5. The input header will receive a 3.3 V signal and 5 V power supply from the ESP32.

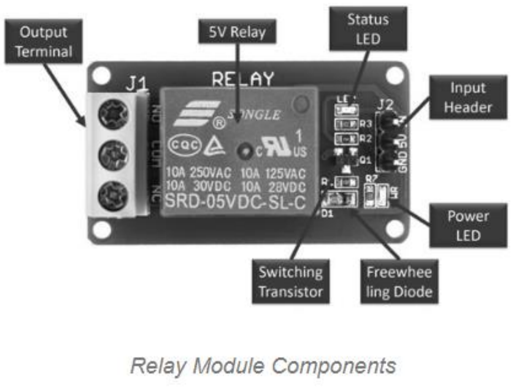


Figure 3.5: 5V Relay Module Structure (Agarwal, 2021)

### 3.3.5 HLK-PM01 AC to DC 5V Power Module

The main power supply for the smart socket which powers up the ACS712, relay, and ESP32 is the HLK-PM01 power converter. This converter is capable of accepting 100-240 Vac, 50-60 Hz, 200 mA, and it provides a 5 Vdc, 3 W, 600-1000 mA output current. Additionally, the converter is equipped with a built-in short circuit and overcurrent protection system ensuring safe and reliable operation. Figure 3.6 shows the component's pinout diagram.

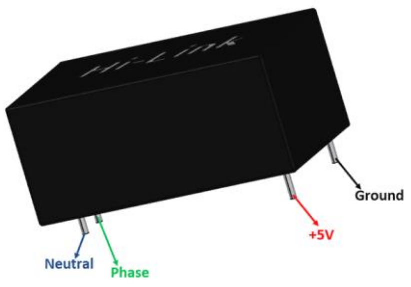


Figure 3.6: HLK-PM01 Pinout Diagram (Components101, 2018)

## 3.4 Software Description

This section states the software to be used in developing the Smart Socket system.

### 3.4.1 Arduino IDE

The Arduino Integrated Development Environment (IDE) is software that contains a built-in text editor or script that enables users to programme and communicate with a controller by coding in C++ or MicroPython programming language. For this project, the ESP32 board is first installed on the Arduino IDE via the Board Manager. Then, the Arduino programming language was utilized to control the ESP32 by driving its GPIO pins to perform desired functions.

### 3.4.2 Firebase Database

The Firebase Database is a cloud-hosted NoSQL database that facilitates the storage and synchronization of real-time data between the server and users. It also supports offline data storing by enabling SDK to store the data in the local cache of the users' device and uploads the data to the cloud once users go online. The Firebase database will be used to store current values and power consumption (kWh) for AI prediction. Moreover, it can communicate with the node-red platform seamlessly and perform data fetching.

### 3.4.3 MQ Telemetry Transport Protocol (MQTT)

MQTT (Message Queuing Telemetry Transport) is a lightweight network protocol that is commonly used for IoT applications. An MQTT broker should be set up to transmit the measured current and voltage value to the Raspberry Pi (Server/broker) through a local network. In this project, both ESP32 microcontroller and the Raspberry Pi server is the publisher/subscriber while the Raspberry Pi will also be the MQTT broker. The Eclipse Mosquitto software will be installed at the RPI to link devices. The IP address of the RPI will be inserted as the IP address for the MQTT broker and the MQTT username and password will be the access key for the subscriber and publisher. Besides, MQTT Client ID will also be assigned to each socket to secure the transmission path.

### 3.4.4 Node-red Platform

Node-Red is a programming tool that visualizes a programme by interconnecting the nodes for hardware connection, APIs and more to create a complete working flow. It is available on Windows, Linux and also Raspberry Pi OS. It is widely used in IoT systems for real-time data collection and is capable to drive the entire data transmission system. In this project, Node-red will be integrated into the Raspberry Pi for data sending/retrieving upon Mosquitto MQTT broker. Additionally, a user interface will be created on Node-red for visualization.

### 3.4.5 Jupyter Notebook

Jupyter Notebook is an open-source web application that capable of running codes, equations or text in many programming languages. The ability to visualize and analyse data in Realtime and modify code accordingly makes it very useful for various data-driven projects. In this project, this platform will be used to develop an AI model through Python programming language for power consumption forecasting in the Cloud. The calculated power consumption stored in the Cloud will be fed into the AI model as model input. Furthermore, the TensorFlow library is one of the open-source libraries for machine learning, and it provides a flexible and powerful platform for building and training various AI models.

### 3.4.6 Over-the-Air Update (OTA)

OTA update is a wireless feature that allows bug fixes and firmware updates on a device without requiring physical access to it. For this project, the OTA update will be integrated into the smart socket system. Among the various methods of implementing OTA update, the remote OTA update through the Cloud method was chosen. To enable this, the mDash software which is built on top of Mongoose OS will be used. It will connect all the devices using a unique mDash device key. The binary file containing the firmware update can be directly uploaded to the device without requiring any interaction from the end user.

## 3.5 Implementation of Project

This section aims to demonstrate the process of integrating the hardware and software components to develop a functional smart socket prototype.

### 3.5.1 Hardware Implementation

This subsection describes the connection of HLK-PM01 power module, ACS712 current sensor as well as the pinout of ESP32 microcontroller.

#### 3.5.1.1 Circuit Connection

The ESP32 controller plays a crucial role in detecting and outputting signal from its GPIO pin. The 5V relay module is controlled using GPIO13 which produces a high (3.3V) or low (0V) signal. GPIO36/VP, an input-only pin, reads the analogue voltage input from the $V_{IOUT}$ of the current sensor. As per the ACS712-20A current sensor datasheet, the $V_{IOUT}$ output range is between 0.5-4.5V, which exceeds the maximum voltage of ESP32, 3.3V. Hence, a voltage divider circuit as shown in Figure 3.7 is required to limit the voltage below 3.3V before being fed into GPIO36. $R_1$ and $R_2$ were selected to be 10 kΩ and 20 kΩ respectively to achieve a maximum of 3.0 V at 20A. The resistance value was chosen based on the voltage and current flow in the circuit. Due to the insufficient current supply to the current sensor, a larger resistance was used to maintain the no-load voltage at 2.5V. A small value of the capacitor, $C_1$, with high reactance was added in parallel with the input pin to filter high-frequency noise and provide stable current sensor reading.

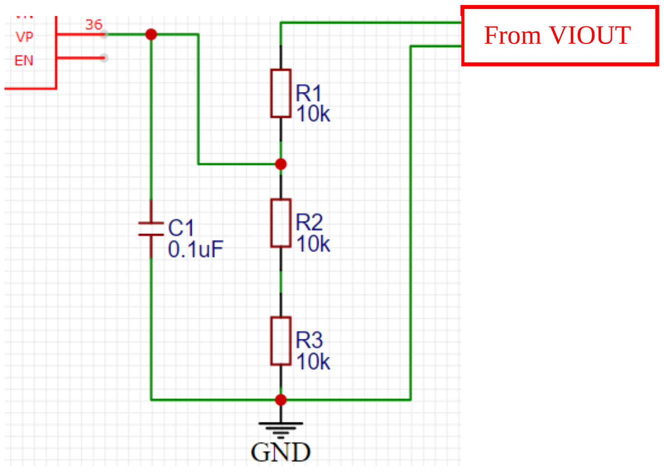


Figure 3.7: Voltage Divider Circuit at GPIO36

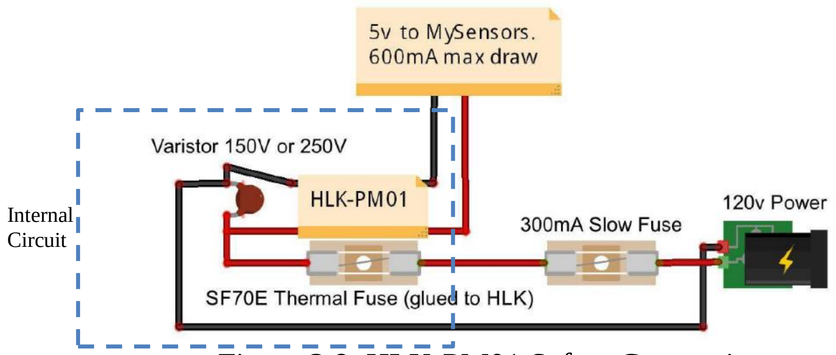


Figure 3.8: HLK-PM01 Safety Connection

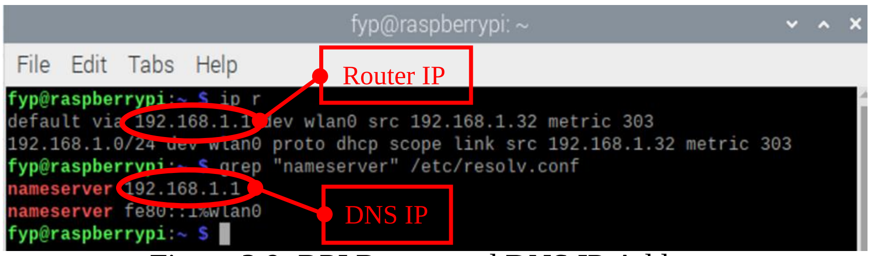
Based on the HLK-PM01 datasheets, the maximum current rating of the power module is 200 mA. In order to protect the power module from failure, a slow blow fuse with a larger current rating of 300 mA was chosen. This fuse was selected to handle scenarios such as inrush current where there is a sudden surge in current that the fuse needs to handle without immediately blowing. Figure 3.8 shows the schematic of the fuse being added in series with the input of the power module.

### 3.5.2 Software Implementation

This subsection shows the MQTT protocol and Firebase setup. Besides, AI model training technique using Jupyter Notebook will also be discussed.

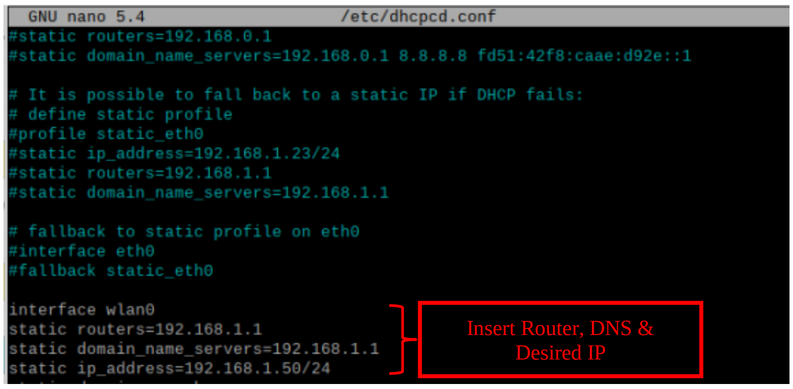
#### 3.5.2.1 Node-red MQTT Protocol and Firebase Setup

A unique MQTT server IP address should be used to link the broker (RPI) and client (SS) together. In this project, the RPI server IP address is used as the MQTT server IP address upon the setup of the Mosquitto broker. However, it was discovered that the RPI tends to use dynamic IP addresses by default. To address this issue, a static IP address can be set up as shown in Figures 3.9 and 3.10. The router and the domain nameserver (DNS) address were obtained and used to configure the WLAN in "/etc/dhcpcd.conf" Dynamic Host Configuration Protocol (DHCP) network configuration file. The desired MQTT server IP address was set to "192.168.1.50" in this project.



```
fyp@raspberrypi:~ $ ip r
default via 192.168.1.1 dev wlan0 src 192.168.1.32 metric 303
192.168.1.0/24 dev wlan0 proto dhcp scope link src 192.168.1.32 metric 303
fyp@raspberrypi:~ $ grep "nameserver" /etc/resolv.conf
nameserver 192.168.1.1
nameserver fe80::1%wlan0
fyp@raspberrypi:~ $
```

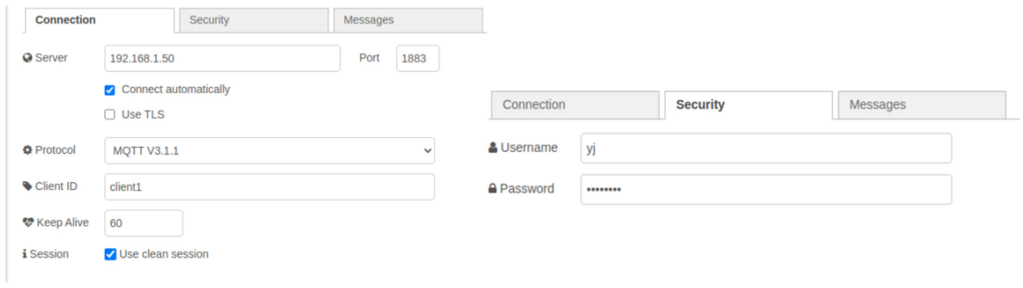
Figure 3.9: RPI Router and DNS IP Address



```
  GNU nano 5.4                    /etc/dhcpcd.conf
#static routers=192.168.0.1
#static domain_name_servers=192.168.0.1 8.8.8.8 fd51:42f8:caae:d92e::1

# It is possible to fall back to a static IP if DHCP fails:
# define static profile
#profile static_eth0
#static ip_address=192.168.1.23/24
#static routers=192.168.1.1
#static domain_name_servers=192.168.1.1

# fallback to static profile on eth0
#interface eth0
#fallback static_eth0

interface wlan0
static routers=192.168.1.1
static domain_name_servers=192.168.1.1
static ip_address=192.168.1.50/24
```

Figure 3.10: RPI Static IP Address Setup

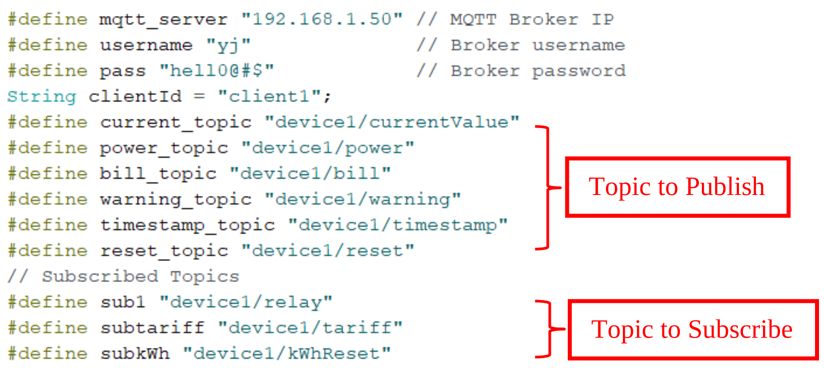
MQTT connection between RPI and SS can be set up by utilizing the Node-red platform. The platform offers built-in MQTT in and MQTT out nodes for data receiving and publishing through MQTT broker communication. The required information to establish the connection is the server address, username and password for security purposes and the ClientID for client's identification as shown in Figure 3.11. Besides, correct information must be uploaded to the ESP32 controller through the Arduino IDE to link with the MQTT server. The code for setting up Smart Socket 1 can be found in Figure 3.12.



Server: 192.168.1.50  Port: 1883
☑ Connect automatically
☐ Use TLS
Protocol: MQTT V3.1.1
Client ID: client1
Keep Alive: 60
Session: ☑ Use clean session

Username: yj
Password: ••••••••

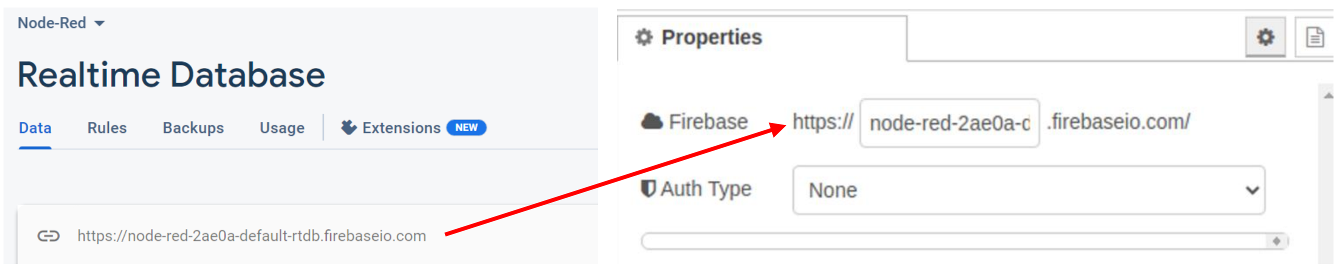
Figure 3.11: Node-red MQTT Set up



```
#define mqtt_server "192.168.1.50" // MQTT Broker IP
#define username "yj"              // Broker username
#define pass "hell0@#$"            // Broker password
String clientId = "client1";
#define current_topic "device1/currentValue"
#define power_topic "device1/power"
#define bill_topic "device1/bill"
#define warning_topic "device1/warning"
#define timestamp_topic "device1/timestamp"
#define reset_topic "device1/reset"
// Subscribed Topics
#define sub1 "device1/relay"
#define subtariff "device1/tariff"
#define subkWh "device1/kWhReset"
```

Figure 3.12: ESP32 programme code for MQTT Connection

Besides, the Node-red Platform is capable to connect with the Firebase Realtime Database by installing the "node-red-contrib-firebase" palette. Before the connection, a project must be created at the Firebase. The reference URL of the Realtime database should be provided in the Firebase node to establish a connection between Node-red and Firebase as shown in Figure 3.13. The data will be appended to the Realtime database based on the specified Child Path.



https://node-red-2ae0a-default-rtdb.firebaseio.com

Firebase: https:// node-red-2ae0a-d .firebaseio.com/
Auth Type: None

Figure 3.13: Connection between Firebase (Left) and Node-Red (Right)

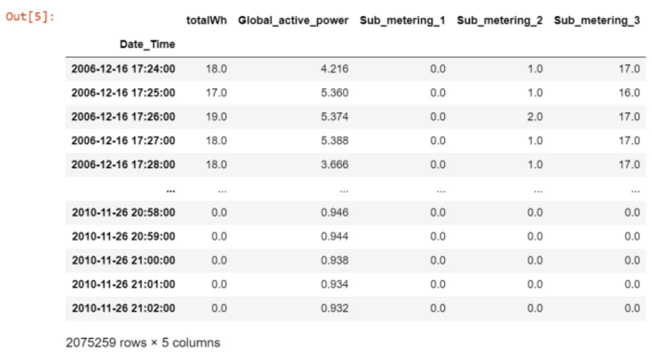
#### 3.5.2.2 Jupyter Notebook AI Model Training Setup

In this project, an AI model was developed to predict one day ahead hourly power consumption (Wh) for users using a dataset that archived France's household power consumptions for 3 sub-metering from December 2006 to November 2010 (UCI, n.d.). The details of the datasets can be seen in Table 3.1.

Table 3.1: Datasets Descriptions

| | |
|---|---|
| **Data Range** | Since 2006-12-16 17:24:00 to 2010-11-26 21:02:00 with 1 minutes timestep |
| **No. of Rows** | 2075259 |
| **No. of Columns** | 7 |
| **Columns Content** | 1. **Global_active_power** – Total kW power consumption of the household<br>2. **Global_reactive_power** – Total kVAR reactive power of the household<br>3. **Voltage** – Supply Voltage (V)<br>4. **Global_intensity** – Total intensity (A) of the household<br>5. **Sub_metering_1** – Kitchen power usage (Wh) from dishwasher, oven, microwave<br>6. **Sub_metering_2** – Laundry Room power usage (Wh) from washing machine, refrigerator, tumble drier & light<br>7. **Sub_metering_3** – Water heater & Air-conditioner power usage (Wh) |

To facilitate data analysis, Python programming language support libraries including Pandas, Numpy, Keras, and Tensorflow, were imported to reshape and manipulate the datasets before feeding into the AI model for training and inferencing. Unnecessary columns such as reactive power, voltage and intensity were dropped, and a "totalWh" column was added for future watt-hour (Wh) prediction purposes. The features that fed into the AI model are shown in Figure 3.14. Besides, resampling and rescaling of data were performed to improve performance by reducing the size of the resources and making it more lightweight. The function to construct a 24 hours prediction time-series data frame for model training is shown in Figure 3.15.

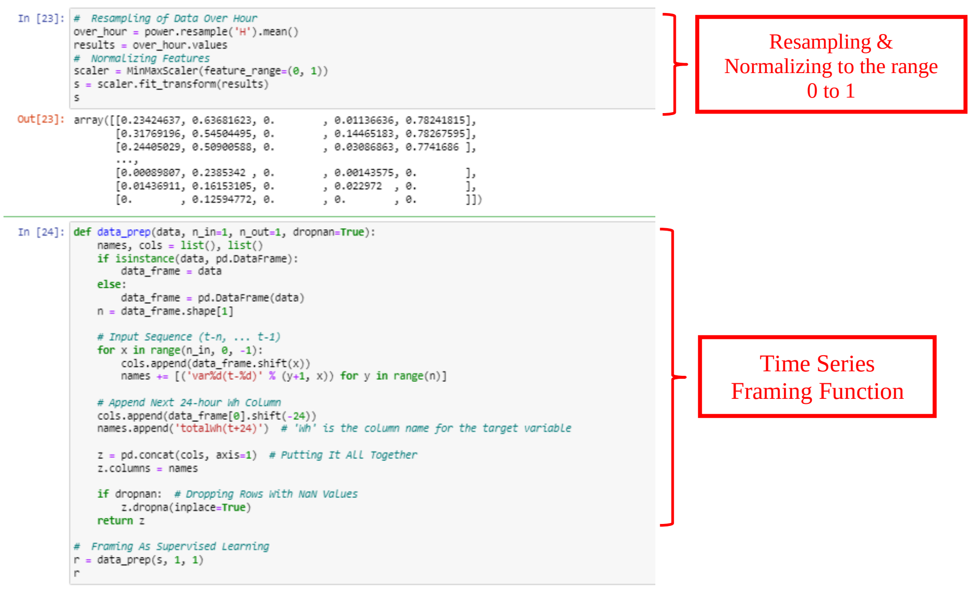


Out[5]:

| Date_Time | totalWh | Global_active_power | Sub_metering_1 | Sub_metering_2 | Sub_metering_3 |
|---|---|---|---|---|---|
| 2006-12-16 17:24:00 | 18.0 | 4.216 | 0.0 | 1.0 | 17.0 |
| 2006-12-16 17:25:00 | 17.0 | 5.360 | 0.0 | 1.0 | 16.0 |
| 2006-12-16 17:26:00 | 19.0 | 5.374 | 0.0 | 2.0 | 17.0 |
| 2006-12-16 17:27:00 | 18.0 | 5.388 | 0.0 | 1.0 | 17.0 |
| 2006-12-16 17:28:00 | 18.0 | 3.666 | 0.0 | 1.0 | 17.0 |
| ... | ... | ... | ... | ... | ... |
| 2010-11-26 20:58:00 | 0.0 | 0.946 | 0.0 | 0.0 | 0.0 |
| 2010-11-26 20:59:00 | 0.0 | 0.944 | 0.0 | 0.0 | 0.0 |
| 2010-11-26 21:00:00 | 0.0 | 0.938 | 0.0 | 0.0 | 0.0 |
| 2010-11-26 21:01:00 | 0.0 | 0.934 | 0.0 | 0.0 | 0.0 |
| 2010-11-26 21:02:00 | 0.0 | 0.932 | 0.0 | 0.0 | 0.0 |

2075259 rows × 5 columns

Figure 3.14: Dataset Fed into the AI Model



```
In [23]: #  Resampling of Data Over Hour
         over_hour = power.resample('H').mean()
         results = over_hour.values
         #  Normalizing Features
         scaler = MinMaxScaler(feature_range=(0, 1))
         s = scaler.fit_transform(results)
         s
Out[23]: array([[0.23424637, 0.63681623, 0.        , 0.01136636, 0.78241815],
                [0.31769196, 0.54504495, 0.        , 0.14465183, 0.78267595],
                [0.24405029, 0.50900588, 0.        , 0.03086863, 0.7741686 ],
                ...,
                [0.00089807, 0.2385342 , 0.        , 0.00143575, 0.        ],
                [0.01436911, 0.16153105, 0.        , 0.022972  , 0.        ],
                [0.        , 0.12594772, 0.        , 0.        , 0.        ]])
```

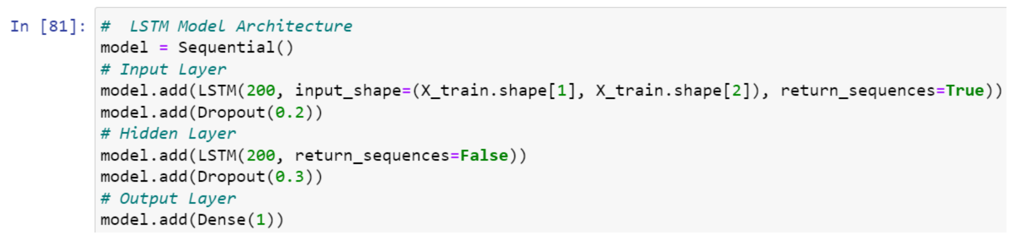
```
In [24]: def data_prep(data, n_in=1, n_out=1, dropnan=True):
             names, cols = list(), list()
             if isinstance(data, pd.DataFrame):
                 data_frame = data
             else:
                 data_frame = pd.DataFrame(data)
             n = data_frame.shape[1]

             # Input Sequence (t-n, ... t-1)
             for x in range(n_in, 0, -1):
                 cols.append(data_frame.shift(x))
                 names += [('var%d(t-%d)' % (y+1, x)) for y in range(n)]

             # Append Next 24-hour Wh Column
             cols.append(data_frame[0].shift(-24))
             names.append('totalWh(t+24)')  # 'Wh' is the column name for the target variable

             z = pd.concat(cols, axis=1)  # Putting It All Together
             z.columns = names

             if dropnan:  # Dropping Rows With NaN Values
                 z.dropna(inplace=True)
             return z

         #  Framing As Supervised Learning
         r = data_prep(s, 1, 1)
         r
```

Resampling & Normalizing to the range 0 to 1
Time Series Framing Function

Figure 3.15: Resampling and Time Series Function

The project utilized a Long Short-Term Memory (LSTM) neural network for load demand forecasting. LSTM is a type of Recurrent Neural Network (RNN) that addresses the issue of gradient vanishing in RNN. The LSTM architecture is similar to RNN but includes a new cell state with forget, input, and output gates that store or discard data based on their model architecture. This allows the network to remember past data while discarding irrelevant data over time, enhancing memory and improving forecasting accuracy. Therefore, this project focused on studying LSTM as a deep learning algorithm for load demand forecasting.

The time-series data frame is split into a training set comprising 80% of the data and a validation/testing set comprising the remaining 20%. To train the model, a 3D array is fed into the LSTM architecture which is composed of one hidden layer to minimize the Mean Squared Error. The LSTM model architecture setup is shown in Figure 3.16. It consists of 3 LSTM layers with two dropout layers that drop out 20% and 30% of data respectively to avoid overfitting issues. Overfitting is when the model tends to memorize instead of learning occurred. The working principle and the performance of the developed AI model will be explained and discussed in the following chapter.



```
In [81]: #  LSTM Model Architecture
         model = Sequential()
         # Input Layer
         model.add(LSTM(200, input_shape=(X_train.shape[1], X_train.shape[2]), return_sequences=True))
         model.add(Dropout(0.2))
         # Hidden Layer
         model.add(LSTM(200, return_sequences=False))
         model.add(Dropout(0.3))
         # Output Layer
         model.add(Dense(1))
```

Figure 3.16: LSTM Model Architecture

## 3.6 Summary

In this chapter, the hardware and software components needed to build the Smart Socket system are discussed as well as the implementation approach and the system architecture. The Smart Socket system consists of ESP32 controllers with relays and current sensors, a Raspberry Pi server and a cloud database. The ESP32 controllers are programmed using the Arduino IDE software and communicate with the Raspberry Pi server through the MQTT protocol. The Raspberry Pi serves as an MQTT broker and employs Node-red to develop the IoT system. The system also utilizes an ACS712 current sensor, a 5V mechanical relay module, and an HLK-PM01 power module. The software used includes the Arduino IDE, Mosquitto MQTT broker, Firebase, Jupyter Notebook and OTA update feature. Moreover, the methods to setup MQTT connection, Firebase and AI Model for prediction are also presented.

# CHAPTER 4

# RESULTS AND DISCUSSION

## 4.1 Introduction

This section shows the final result of the smart socket system prototype. Data obtained will also be tabulated and discussed.

## 4.2 Hardware Prototype

The project includes two hardware prototypes - the central console (server) and two smart socket prototypes. Figure 4.1 shows the central console, Raspberry Pi 3 Model B+. Figure 4.2 shows the final prototype of the smart socket. The detailed smart socket prototype will be presented in the subsequent subsection.

Figure 4.1: Raspberry Pi 4 Central Console with casing

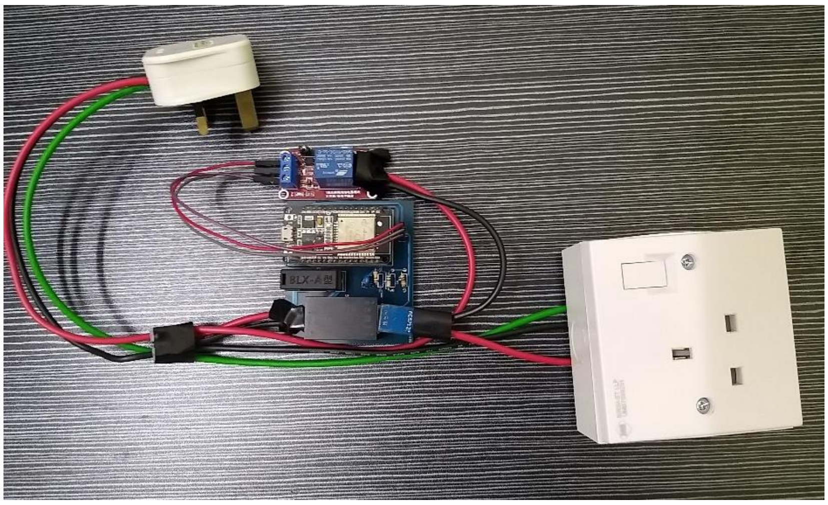


Figure 4.2: Smart Socket Prototype

### 4.2.1 Smart Socket Prototype

The circuit connection schematic is shown in Figure 4.3. The schematic was created on EasyEDA software and is later converted to PCB for fabrication. The PCB consists of two layers where the routing is performed on the top layer while the bottom layer is only used for grounding. Figure 4.4 illustrates the drawing of the PCB while Figure 4.5 shows the top and bottom view of the printed PCB. The routing on the top layer is indicated by the red lines, and the ground plane at the bottom layer is represented by the blue plane.

The size of the PCB board was designed to be 71mm x 64mm which is smaller than the standard size of a wall outlet, approximately 90mm x 90mm. The ACS712 current sensor and 5V relay module are connected to the PCB board through female (P1) and male (H1) connector pins respectively as shown in Figure 4.5.

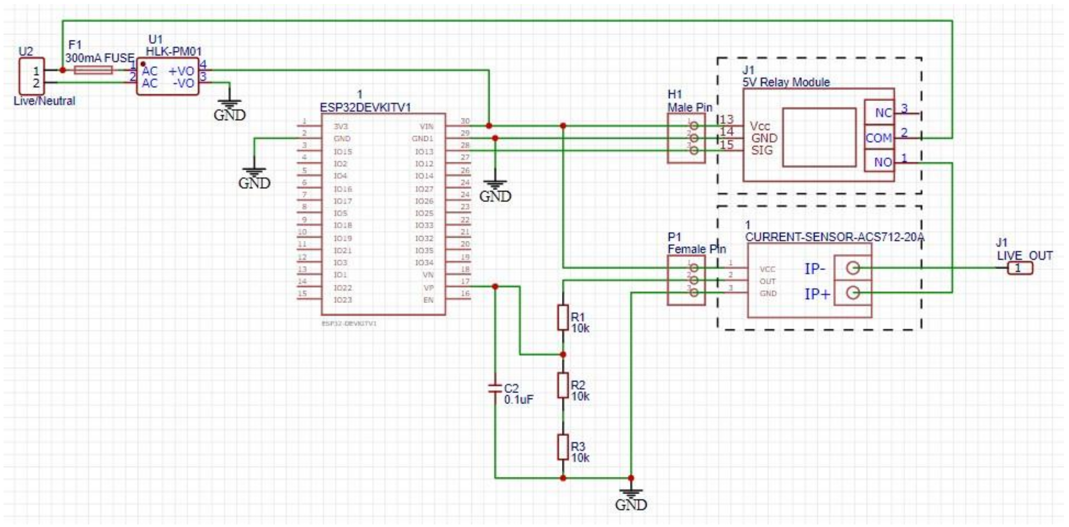


Figure 4.3: Schematic of Smart Socket Circuit

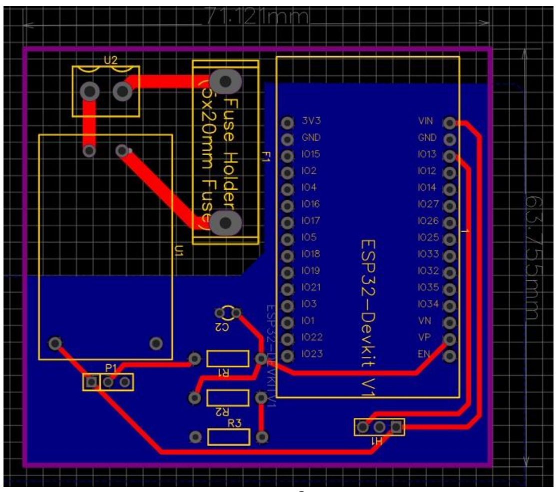


Figure 4.4: Drawing of PCB on EasyEDA

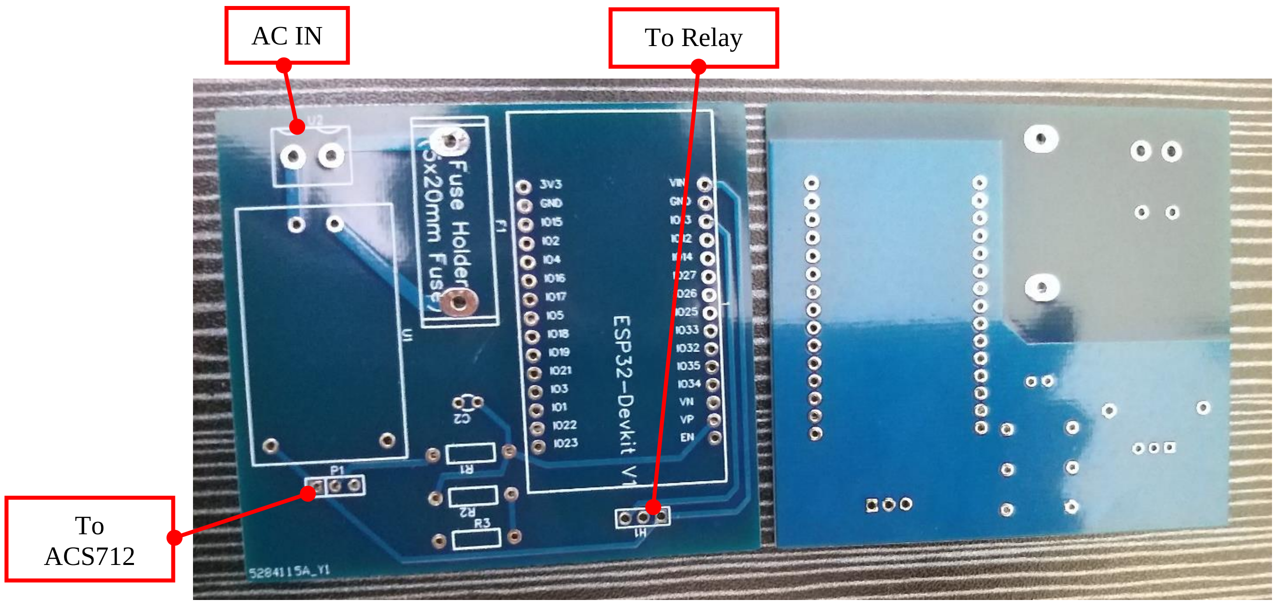


Figure 4.5: Top and Bottom View of PCB Board

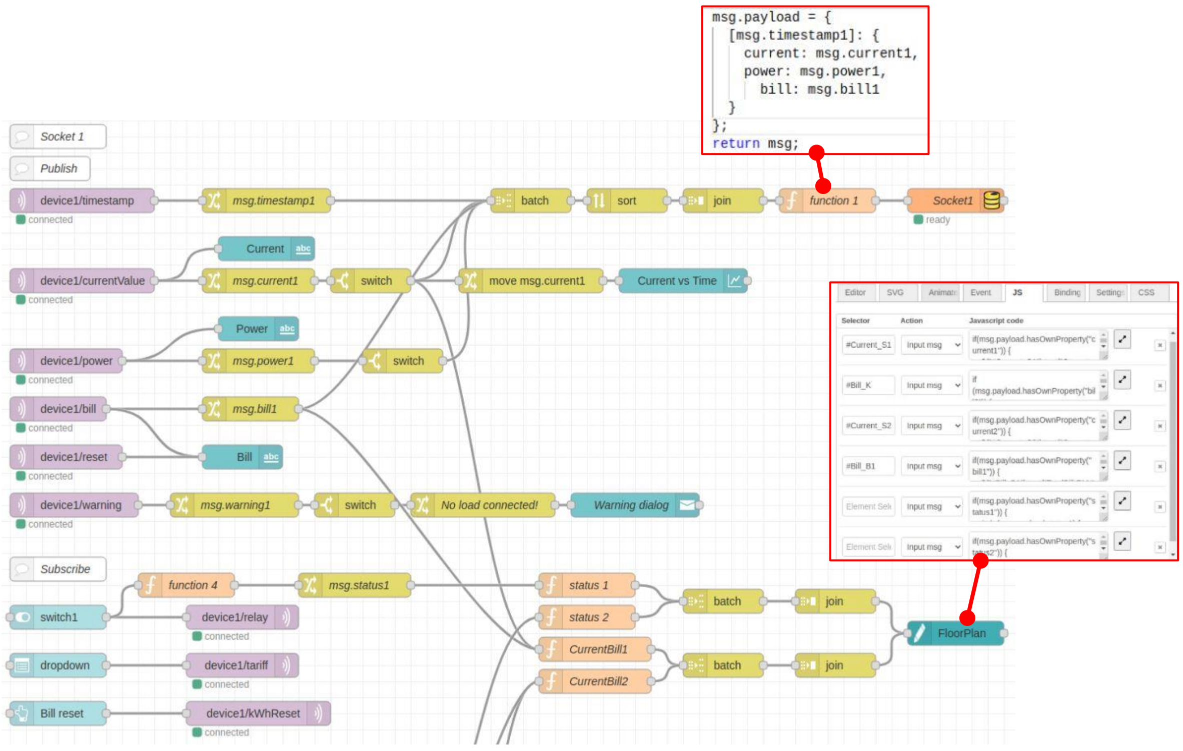
## 4.3 Software Features

This section focuses on the software utilized to control and monitor the smart socket. It provides an overview and discussion of the Simple UI created on Node-RED that utilized a local network, along with various software features that have been implemented on the Smart Sockets.

### 4.3.1 Node-red Data Transfer

The MQTT topic publishing and subscription process is carried out on the Raspberry Pi server using Node-red. Figure 4.6 illustrates the Node-red script of Socket 1 in this project. The MQTT Out nodes (coloured purple) in the "Subscribe" section transmit on/off state, selected tariff, and bill reset condition to the Smart Sockets through the MQTT protocol. In contrast, the MQTT In nodes in the "Publish" section receive real-time timestamp, current, power, and bill calculation based on the selected tariff, and warning status if a no-load condition is detected.

The inputs from MQTT In nodes were converted into specific payload names and combined for ease of data organization before sending to the Firebase database console and the Scalable Vector Graphics (SVG) floorplan node. A function is utilized before the SVG node to define the animation triggering input's on/off state, current, and calculated bill in *msg.payload.<input>* format, aligning with the JavaScript (JS) setup in the SVG node. However, it was discovered that animation clashing may occur within the triggering input. As a result, separate function nodes were implemented to manage the current, bill, and status individually.



```
msg.payload = {
  [msg.timestamp1]: {
    current: msg.current1,
    power: msg.power1,
      bill: msg.bill1
  }
};
return msg;
```

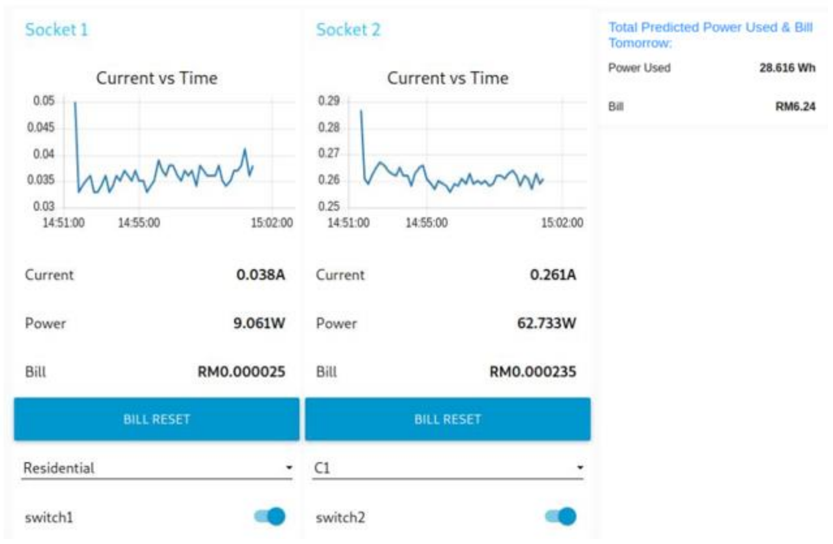
| Selector | Action | Javascript code |
|---|---|---|
| #Current_S1 | Input msg | if(msg.payload.hasOwnProperty("current1")) { |
| #Bill_K | Input msg | if (msg.payload.hasOwnProperty("bil |
| #Current_S2 | Input msg | if(msg.payload.hasOwnProperty("current2")) { |
| #Bill_B1 | Input msg | if(msg.payload.hasOwnProperty("bill1")) { |
| Element Sele | Input msg | if(msg.payload.hasOwnProperty("status1")) { |
| Element Sele | Input msg | if(msg.payload.hasOwnProperty("status2")) { |

Figure 4.6: Node-red Script of Socket 1

### 4.3.2 Node-red User Interface Dashboard

The Node-red dashboard was designed with two tabs. The first tab equipped the chart and text UI node to indicate real-time data of current value, power consumption, and cumulative billing for each socket. Figure 4.7 shows the "Sockets" tab, which contains two connected sockets. The surge at the beginning of the line graph might cause by the inrush current of the closing and opening of contact of the mechanical relay used. The predicted power consumption (Wh) and electricity bill by inserting one-day test data (Open-source Dataset) were also displayed in the UI.

On the other hand, users can choose their preferred tariff rate from the dropdown menu that offers ETOU and Residential rates designed for commercial/industrial and residential users respectively. The tariff rates are defined based on Tenaga Nasional Berhad (TNB), Malaysia as shown in Figures 4.8 and 4.9. The ESP32 in this project is configured to synchronize its clock with the Network Time Protocol (NTP) server to obtain accurate and reliable time information. By connecting to the NTP server, the local Greenwich Mean Time (GMT)+8 time zone is obtained for the bill calculation process of the time-dependent ETOU tariff. Furthermore, the monthly bills are stored in Firebase and can be accessed using Node-Red UI in the "Monthly Bill" Tab as shown in Figure 4.10.



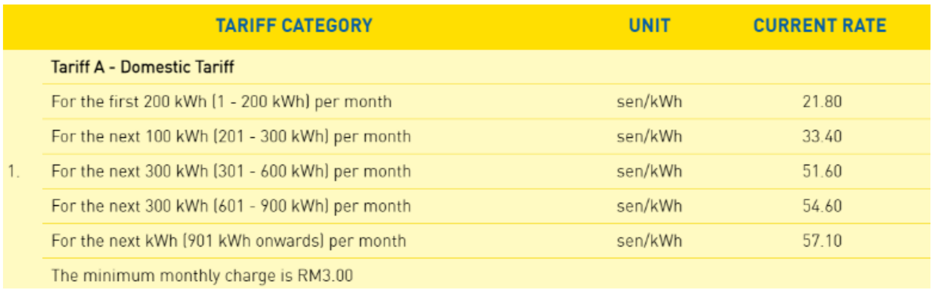
| | |
|---|---|
| Current | 0.038A |
| Power | 9.061W |
| Bill | RM0.000025 |
| Residential | |
| switch1 | |

| | |
|---|---|
| Current | 0.261A |
| Power | 62.733W |
| Bill | RM0.000235 |
| C1 | |
| switch2 | |

Total Predicted Power Used & Bill Tomorrow:

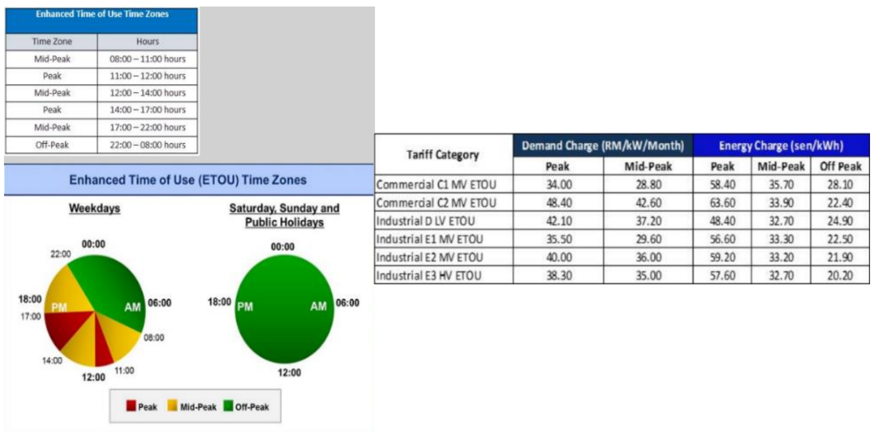
| | |
|---|---|
| Power Used | 28.616 Wh |
| Bill | RM6.24 |

Figure 4.7: Smart Sockets Control Panel



| | TARIFF CATEGORY | UNIT | CURRENT RATE |
|---|---|---|---|
| | Tariff A - Domestic Tariff | | |
| | For the first 200 kWh (1 - 200 kWh) per month | sen/kWh | 21.80 |
| | For the next 100 kWh (201 - 300 kWh) per month | sen/kWh | 33.40 |
| 1. | For the next 300 kWh (301 - 600 kWh) per month | sen/kWh | 51.60 |
| | For the next 300 kWh (601 - 900 kWh) per month | sen/kWh | 54.60 |
| | For the next kWh (901 kWh onwards) per month | sen/kWh | 57.10 |
| | The minimum monthly charge is RM3.00 | | |

Figure 4.8: Residential Tariff Rate (TNB, n.d.)



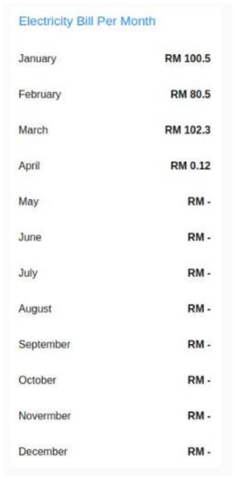
Enhanced Time of Use Time Zones

| Time Zone | Hours |
|---|---|
| Mid-Peak | 08:00 – 11:00 hours |
| Peak | 11:00 – 12:00 hours |
| Mid-Peak | 12:00 – 14:00 hours |
| Peak | 14:00 – 17:00 hours |
| Mid-Peak | 17:00 – 22:00 hours |
| Off-Peak | 22:00 – 08:00 hours |

Enhanced Time of Use (ETOU) Time Zones
Weekdays
00:00
22:00
18:00
17:00
PM
AM
06:00
08:00
14:00
12:00
11:00
Saturday, Sunday and Public Holidays
00:00
18:00
PM
AM
06:00
12:00
🟥 Peak
🟨 Mid-Peak
🟩 Off-Peak

| Tariff Category | Demand Charge (RM/kW/Month) | | Energy Charge (sen/kWh) | | |
|---|---|---|---|---|---|
| | Peak | Mid-Peak | Peak | Mid-Peak | Off Peak |
| Commercial C1 MV ETOU | 34.00 | 28.80 | 58.40 | 35.70 | 28.10 |
| Commercial C2 MV ETOU | 48.40 | 42.60 | 63.60 | 33.90 | 22.40 |
| Industrial D LV ETOU | 42.10 | 37.20 | 48.40 | 32.70 | 24.90 |
| Industrial E1 MV ETOU | 35.50 | 29.60 | 56.60 | 33.30 | 22.50 |
| Industrial E2 MV ETOU | 40.00 | 36.00 | 59.20 | 33.20 | 21.90 |
| Industrial E3 HV ETOU | 38.30 | 35.00 | 57.60 | 32.70 | 20.20 |

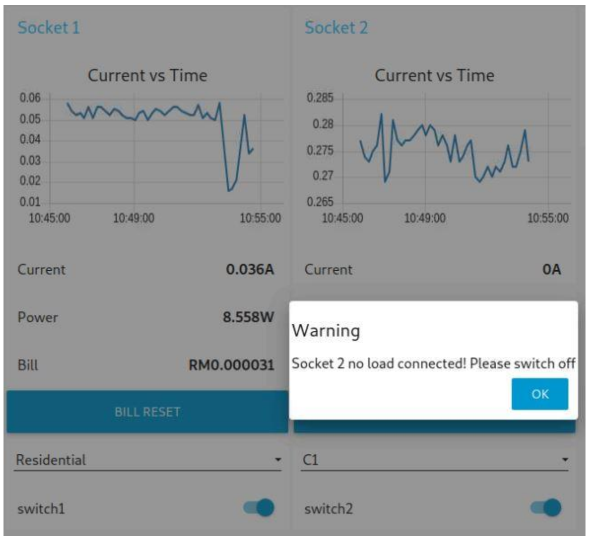
Figure 4.9: Commercial and Industrial ETOU Tariff Rate (TNB, n.d.)



Electricity Bill Per Month

| | |
|---|---|
| January | RM 100.5 |
| February | RM 80.5 |
| March | RM 102.3 |
| April | RM 0.12 |
| May | RM - |
| June | RM - |
| July | RM - |
| August | RM - |
| September | RM - |
| October | RM - |
| Novermber | RM - |
| December | RM - |

Figure 4.10: Monthly Bill Tab



Socket 1

| | |
|---|---|
| Current | 0.036A |
| Power | 8.558W |
| Bill | RM0.000031 |

Residential

switch1

Socket 2

| | |
|---|---|
| Current | 0A |

Warning

Socket 2 no load connected! Please switch off

C1

switch2

Figure 4.11: No-load Condition Warning Dialog

Additionally, the smart socket is equipped with a feature that detects no-load conditions. In the event that a no-load is connected to the socket and the user switches on the relay, a warning dialogue will appear on the dashboard to prompt the user to switch off the socket. This no-load warning feature can be seen in Figure 4.11. Moreover, the Smart Sockets project features a floorplan visualization created using SVG visual elements, which can be accessed on the "FloorPlan" tab of the Node-red dashboard. The floorplan portrays a sample household with two bedrooms, a kitchen, a living room, and two bathrooms. The two connected sockets are represented by socket symbols located in Bedroom 1 (S1) and the kitchen (S2) respectively. Each symbol includes the status of the socket (represented by a lightbulb) and the real-time current value. The "TotalBill:" displayed in each room indicates the accumulated amount of money that each socket in that room has consumed. The floorplan provides improved visualization for the users to monitor the status of each socket and switch them on/off accordingly.

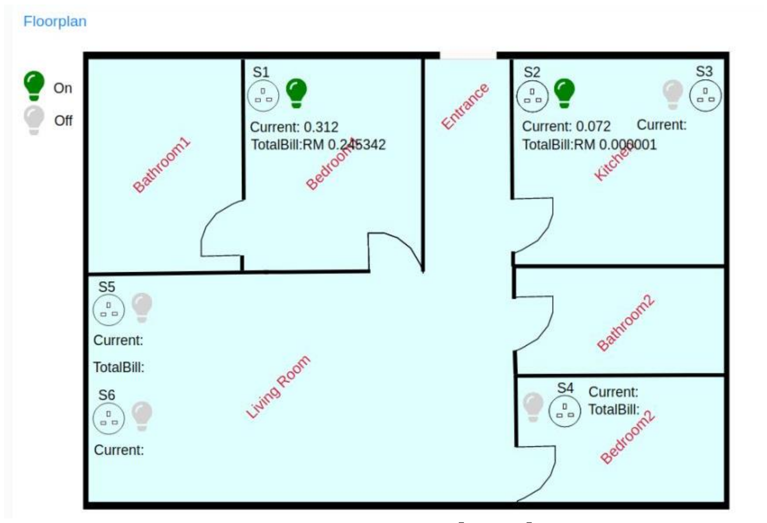
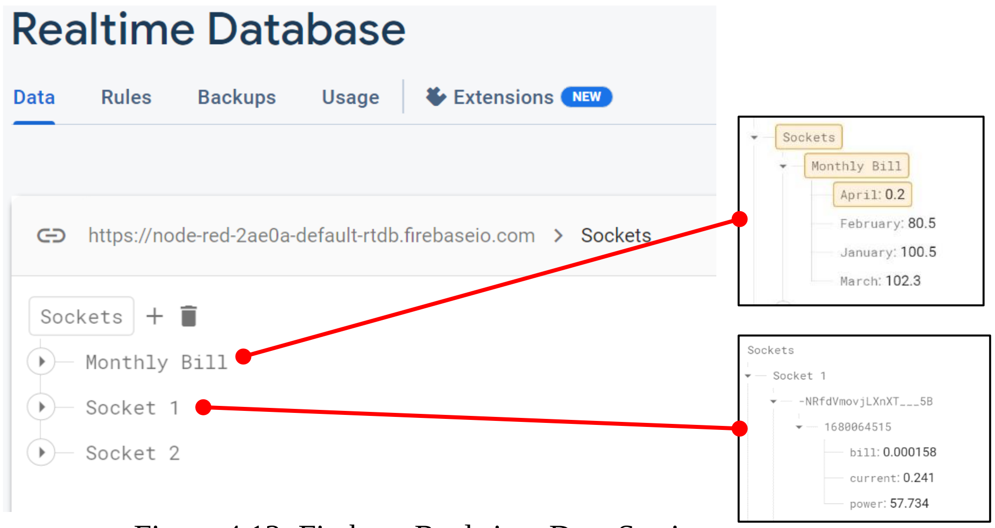


Figure 4.12: SVG Floorplan

### 4.3.3 Firebase Console

This project utilized the Firebase Platform as the Cloud storage for data exchange between the Android User Interface and the smart sockets as well as for future AI prediction. Specifically, the Realtime Database feature was used to store real-time information such as current, active power and bill for each smart socket as well as the monthly bills as shown in Figure 4.13. The real-time data is stored under present timestamp. The unique string before the timestamp layer is generated by Firebase to act as an identifier for each data entry.



```
https://node-red-2ae0a-default-rtdb.firebaseio.com > Sockets

Sockets
  Monthly Bill
  Socket 1
  Socket 2
```

```
Sockets
  Monthly Bill
    April: 0.2
    February: 80.5
    January: 100.5
    March: 102.3
```

```
Sockets
  Socket 1
    -NRfdVmovjLXnXT___5B
      1680064515
        bill: 0.000158
        current: 0.241
        power: 57.734
```

Figure 4.13: Firebase Real-time Data Storing

### 4.3.4 Cloud-Based Over-the-Air (OTA) Update

A Cloud-Based firmware update was performed by utilizing OTA features provided by the mDash software. This was done to enable bug fixes, add new features and improve the overall functionality of the IoT devices. By inserting the unique passKey for each added device provided by mDash software into the ESP32, a secure Transport Layer Security (TLS) connection has been established between the device and the mDash server. This allowed the binary file update process to be carried out wirelessly in an encrypted environment. Figure 4.14 illustrates the binary file update process being performed on Socket 1 while Figure 4.15 shows that both Smart Sockets have been connected to the mDash platform and OTA update can be performed anytime once the devices are online.

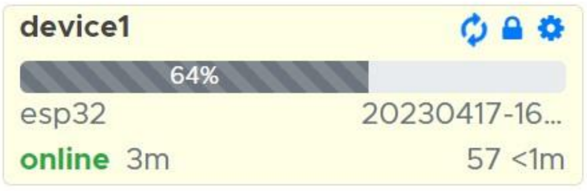
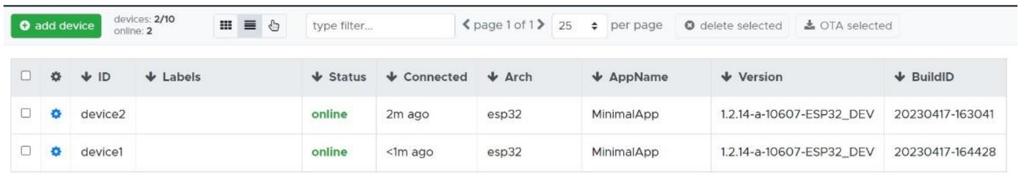


Figure 4.14: Uploading Binary File



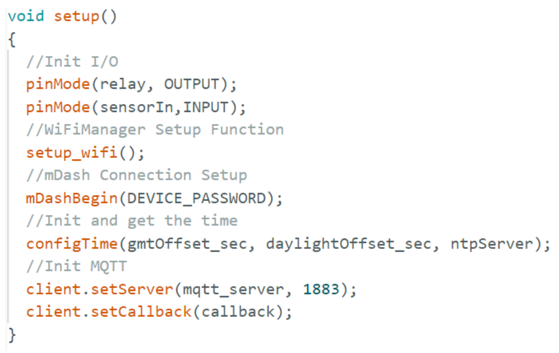
| ID | Labels | Status | Connected | Arch | AppName | Version | BuildID |
|---|---|---|---|---|---|---|---|
| device2 | | online | 2m ago | esp32 | MinimalApp | 1.2.14-a-10607-ESP32_DEV | 20230417-163041 |
| device1 | | online | <1m ago | esp32 | MinimalApp | 1.2.14-a-10607-ESP32_DEV | 20230417-164428 |

Figure 4.15: Two Smart Sockets OTA Status

### 4.3.5 Smart Socket System Operation Flowchart

The flowchart in Figure 4.17 presents the sequence of actions and tasks carried out in the ESP32 program code for establishing and maintaining the connection between the ESP32 and the RPI server. The first part of the flowchart (Part A) shows the essential setup tasks that must be carried out before the void loop() main program including setting up the ESP32 I/O pin for the relay and ACS712 sensor, configuring mDash platform and MQTT server settings, Wi-Fi connection setup using WIFI Manager, and connecting to an NTP server to retrieve local time. These configuration programme codes can be seen in Figure 4.16, where the initialization is located in the void setup() function.

Subsequently, the MQTT broker-client connection is executed in the void loop() repeatedly to ensure continuous topic subscription and publishing. The details of this process can be found in Part B of the flowchart in Figure 4.17 which the relay status and bill reset subscription must be regularly checked before conducting no-load condition checks and data retrieval. The measurement of current value, calculation of power consumption and the accumulated bill will occur only when the relay is in the on state. Finally, the exchange of information between the RPI server and the ESP32 can now be carried out seamlessly. Generally, the flowchart shows the precise breakdown of the procedures involved in establishing a reliable connection between the ESP32 and the RPI server which is essential for the smooth running of this IoT project.



```
void setup()
{
  //Init I/O
  pinMode(relay, OUTPUT);
  pinMode(sensorIn,INPUT);
  //WiFiManager Setup Function
  setup_wifi();
  //mDash Connection Setup
  mDashBegin(DEVICE_PASSWORD);
  //Init and get the time
  configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);
  //Init MQTT
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}
```

Figure 4.16: Setup before Main Programme

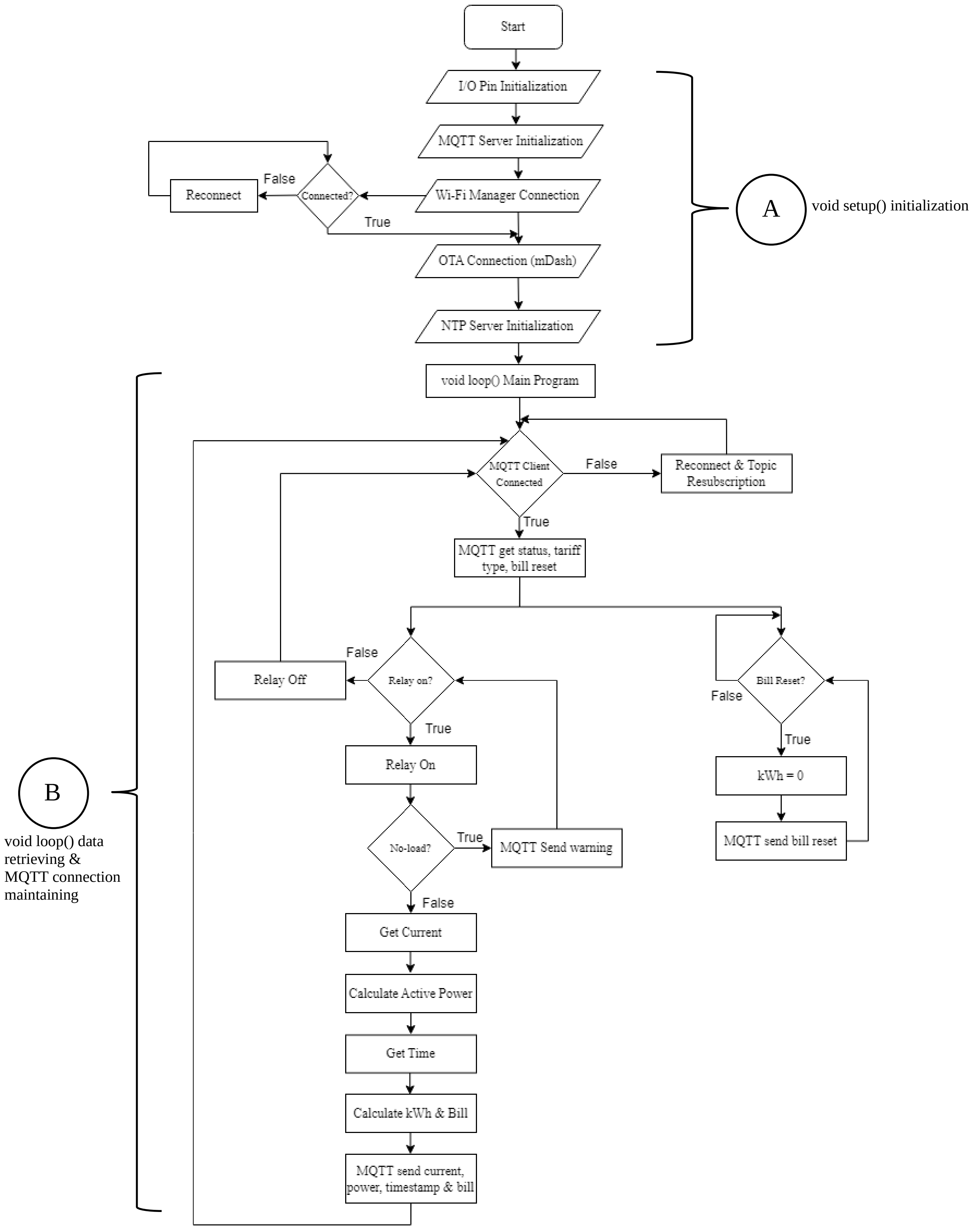


Figure 4.17: Smart Socket System Flowchart

## 4.4 Smart Socket System Performance Evaluation

This section will provide a comprehensive evaluation and discussion on the performance and accuracy of the current sensor along with the analysis of the power consumption of the system.

### 4.4.1 ACS712 Current Sensor Accuracy

The ACS712 current sensor module measured the current based on the changes in the magnetic field around the live wire terminal. It has been observed that the sensor tends to fluctuate frequently which could be caused by external factors such as magnetic field interruptions or instability in the grounding pin. According to the internal structure of the ACS712 current sensor module as shown in Figure 3.4 in Section 3.3.3, two noise suppression capacitors were allocated at the $V_{cc}$ and Gnd pin respectively. However, despite the presence of the internal capacitor, the performance of the sensor is still suboptimal. Therefore, an additional 0.1 μF capacitor has been installed at the $V_{out}$ pin of the current sensor to further reduce the high-frequency noises. Additionally, proper placement of the ACS712 module on the PCB board also contributed to improving the stability of the grounding.

Besides adding a capacitor to establish stable current measurement, a calibration factor and the moving average method were utilized in the ESP32 programme code to achieve accurate and stable output values. The calibration factor is determined through the comparison of the actual current measured by the power meter and the current measured by the ACS712 sensor for various electrical appliances. A calibrating factor of 1.6 was applied for both smart sockets in this project. The results of the comparison between actual and calibrated current for five electrical appliances, including a phone charger, standing fan, small freezer, hair dryer, and water heater, are presented in Table 4.1. The current readings were recorded with an average of three times and the error was calculated using Equation 4.1.

$$Abs\ Error\ (\%) = \frac{Actual\ current - Measured\ current}{Actual\ current} \times 100 \qquad (4.1)$$

Table 4.1: Tabulation of Actual and Measured Electrical Appliances' Current

| **Electrical Appliances** | | **Current (A)** | | **Error (%)** |
|---|---|---|---|---|
| | | **Actual** | **Measured** | |
| 15W Phone Charger | | 0.050 | 0.040 | 20.00 |
| 50W Standing Fan | Speed 1 | 0.172 | 0.148 | 13.95 |
| | Speed 2 | 0.184 | 0.153 | 16.85 |
| | Speed 3 | 0.208 | 0.171 | 17.79 |
| 55W Small Freezer | | 0.278 | 0.310 | 11.51 |
| 1200W Water Heater | Boiling | 5.681 | 5.868 | 3.29 |
| | Warm | 0.046 | 0.03 | 34.78 |
| 1200W Hair Dryer | Cold | 0.308 | 0.272 | 11.69 |
| | Hot 1 | 2.965 | 2.796 | 5.70 |
| | Hot 2 | 5.320 | 5.550 | 4.14 |

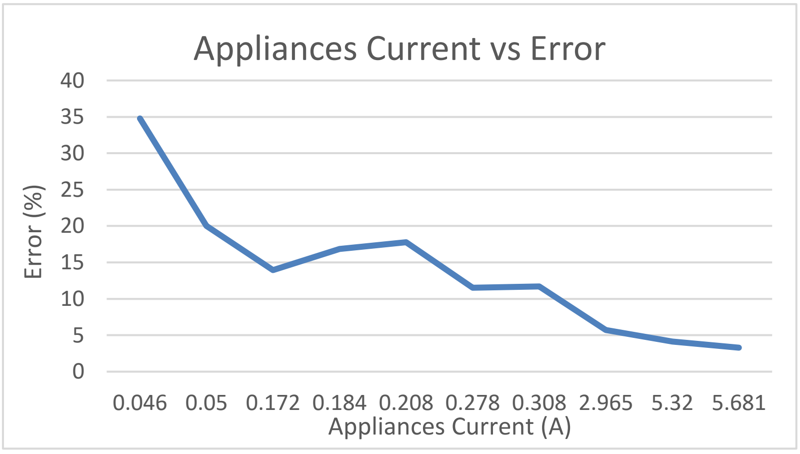


Figure 4.18: Relationship between Error and Current

Based on the findings in Figure 4.18, the absolute error decreases as the appliance's current increases. This indicates the ACS712 current sensor performs better when measuring larger currents. This may be due to the sensitivity and resolution of the sensor as the sensor with a maximum current rating of 20 A and sensitivity of 100 mV/A was used in this project. For example, when detecting a current of 10 mA, the resulting change in output voltage would only be 1 mV which is difficult to accurately differentiate within a small current range. Hence, the error for small current measurement is relatively larger and it is not recommended to use a 20 A current sensor for measuring low-power devices.

Moreover, the active power of each appliance was calculated using Equation 4.2. The actual active power was obtained from the power meter and the comparison between the actual and measured active power of the 5 electrical appliances can be seen in Table 4.2.

$$P = 240 \times I \tag{4.2}$$

where

$P$ = Active Power (W)

$I$ = Measured current (A)

Table 4.2: Tabulation of Actual and Measured Electrical Appliances' Power

| **Electrical Appliances** | | **Power (W)** | | **Power Factor** | **Error (%)** |
|---|---|---|---|---|---|
| | | **Actual** | **Measured** | | |
| 15W Phone Charger | | 6.5 | 11.1 | 0.48 | 70.77 |
| 50W Standing Fan | Speed 1 | 38.5 | 36.6 | 0.89 | 4.94 |
| | Speed 2 | 43.0 | 37.9 | 0.92 | 11.81 |
| | Speed 3 | 51.4 | 41.0 | 0.98 | 20.23 |
| 55W Small Freezer | | 36.5 | 74.6 | 0.52 | 104.38 |
| 1200W Water Heater | Boiling | 1393.0 | 1408.3 | 0.99 | 1.10 |
| | Warm | 11.0 | 6.7 | 0.96 | 39.1 |
| 1200W Hair Dryer | Cold | 69.4 | 65.3 | 0.90 | 5.91 |
| | Hot 1 | 668.8 | 671.0 | 0.91 | 0.34 |
| | Hot 2 | 1305 | 1330.1 | 0.99 | 1.92 |

The absolute error of the actual and calculated active power is not only influenced by the accuracy of the current measurement but also by the power factor of the electrical appliances. The power factor of each appliance varies depending on the type of load (inductive, capacitive, resistive) it represents.

From Table 4.2, the error is smaller for high power devices especially when their power factor is closer to unity, 1. The hair dryer (Hot 2) and water heater (Boiling) have the smallest error as they have a large current and a power factor of 0.99. On the other hand, the 55W small freezer used has a power factor of 0.52 and resulted in a large measurement error. This could be caused by the active power calculation highly depends on the power factor, as shown in Equation 4.3. As the power factor was assumed to be 1 in this project and not directly measured, any smaller power factor values may result in even larger errors, affecting the accuracy of power measurement. Besides, it is known that the input AC voltage can change according to the utilities. However, 240 $V_{ac}$ was assumed in this project for active power calculation. The influences of the input supply voltage in active power calculation can be seen in Equation 4.3.

$$P = V_{ac} \times I \times pf \tag{4.3}$$

where

$P$ = Active Power (W)

$V_{ac}$ = Input Supply AC Voltage (V)

$I$ = Appliances current

$pf$ = Power Factor

### 4.4.2 Power Consumption of the RPI Server

The power consumption of the server was measured using the power meter. The measurement was conducted in three separate 1-hour intervals to obtain an average value for three operating modes. Tabulated data can be seen in Table 4.3 with operating modes:

i. Only Raspberry Pi Server/Broker running
ii. Socket 1 connected to server through MQTT
iii. Sockets 1 and 2 connected to server through MQTT

Table 4.3: Power Consumption of Server per hour

| **Operating Mode** | **Power Consumption Per Hour (W)** | | | **Average (W)** |
|---|---|---|---|---|
| | **1** | **2** | **3** | |
| i | 2.7 | 3.0 | 2.8 | 2.83 |
| ii | 3.1 | 3.3 | 3.0 | 3.13 |
| iii | 3.3 | 3.4 | 3.3 | 3.33 |

As depicted in Table 4.3, the average power consumption of the server increases as the number of connected devices increases. The MQTT connection established between the ESP32 and the broker requires the client to periodically perform connection checking to maintain the connection even when not transmitting any data. The growing workload of the RPI server/broker to handle multiple MQTT connections could be a factor in the increase in power consumption. In addition, the broker may perform authentication for each new connection which could consume processing power on the server.

### 4.4.3 Power Consumption of the Smart Sockets

The power consumption of one of the smart sockets was measured using the power meter. The power consumption was recorded at interval of 1 hour for 3 times to obtain the average. Tabulated data can be seen in Table 4.4 with the operating modes:

i. ESP32 Disconnected with Wi-Fi, MQTT disconnected
ii. ESP32 Connected to Wi-Fi, MQTT connected:
    a. Relay OFF with No-Load
    b. Relay ON with No-Load

Table 4.4: Power Consumption of Smart Sockets per hour

| **Operating Mode** | | **Power Consumption Per Hour (W)** | | | **Average (W)** |
|---|---|---|---|---|---|
| | | **1** | **2** | **3** | |
| i | | 0.4 | 0.5 | 0.4 | 0.43 |
| ii | Relay Off | 0.8 | 0.6 | 0.7 | 0.70 |
| | Relay On | 0.9 | 0.9 | 1.0 | 0.93 |

From Table 4.4, the power consumption of the smart socket without Wi-Fi connection and operations is attributed to the power intake of the components on the PCB board such as the ESP32 microcontroller's memory allocation process and the power supply required by the AC-DC converter, mechanical relay and ACS712 current sensor. When Wi-Fi is connected, the ESP32 is responsible to establish a persistent Wi-Fi and MQTT connection with the MQTT broker by regularly subscribing to the topics. In relay ON mode, the power consumption increases by approximately 0.23 W. This could be attributed to the power required by the relay coil to keep the contact fixed. A small amount of power is needed to create and maintain the electromagnetic field of the coil to keep it energized.

### 4.4.4 Electricity Tariff of the Smart Sockets System

The electricity bill of the smart socket system was calculated based on Malaysia's Domestic tariff as an evaluation. The daily, monthly and yearly power consumption of both the server and smart sockets were computed using Equations 4.4, 4.5 and 4.6 assuming a tariff rate of 0.218 for power consumption below 200 kWh. The power consumption in no-load mode is presented in Table 4.5.

$$Daily\ Cost = Power(kW) * 24\ hours * 0.218 \tag{4.4}$$

$$Monthly\ Cost = Daily\ Cost * 30\ days \tag{4.5}$$

$$Yearly\ Cost = Monthly\ Cost * 12\ months \tag{4.6}$$

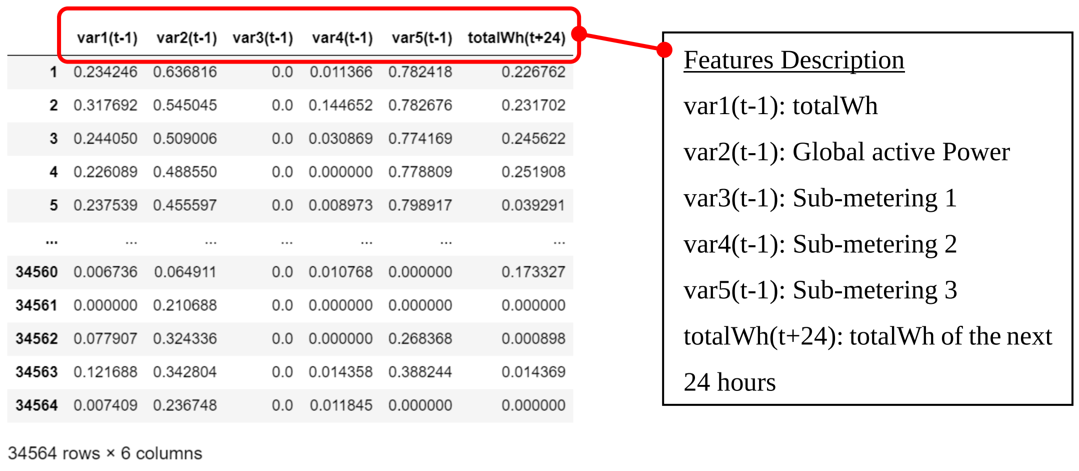
Table 4.5: Electricity Cost of the System

| **Operating Modes** | | **Power (W)** | **Electricity Bill (RM)** | | |
|---|---|---|---|---|---|
| | | | **Daily** | **Monthly** | **Yearly** |
| Server + Two Sockets | Relay OFF | 4.73 | 0.02 | 0.60 | 7.20 |
| | Relay ON | 5.19 | 0.03 | 0.90 | 10.80 |

From Table 4.5, it can be estimated that the yearly cost of the system would be RM 10.80 if the relay is left ON in no-load mode. However, if the no-load detection feature is utilized, the electricity bill can be reduced to RM 7.20 per year. It is important to note that the tabulated data may not be entirely accurate since the server's power consumption is included in the user's electricity bill and a residential tariff rate of 0.218 is assumed. While the data is not exact, the power consumption of the overall smart socket system is still reasonable and cost savings can be realised by putting the no-load detection feature in place.

## 4.5 Prediction of Power Consumption for Appliances

The France household dataset with data resampling to an hourly basis was used, as described in Section 3.5.2.2. The Jupyter Notebook Platform was used for model inferencing and the deep learning LSTM algorithm was employed to perform prediction tasks. To prepare the time-series data frame for the LSTM algorithm, the function shown in Section 3.5.2.2 previously was used. The time-series data frame aims to provide a learning trend to the AI model by taking total Wh, active power and sub-metering 1,2,3 features as references. The desired output was the forecasted power consumption (in Watt-hour) for the next 24 hours. The time-series data frame for the model preparation is illustrated in Figure 4.19.



| | var1(t-1) | var2(t-1) | var3(t-1) | var4(t-1) | var5(t-1) | totalWh(t+24) |
|---|---|---|---|---|---|---|
| 1 | 0.234246 | 0.636816 | 0.0 | 0.011366 | 0.782418 | 0.226762 |
| 2 | 0.317692 | 0.545045 | 0.0 | 0.144652 | 0.782676 | 0.231702 |
| 3 | 0.244050 | 0.509006 | 0.0 | 0.030869 | 0.774169 | 0.245622 |
| 4 | 0.226089 | 0.488550 | 0.0 | 0.000000 | 0.778809 | 0.251908 |
| 5 | 0.237539 | 0.455597 | 0.0 | 0.008973 | 0.798917 | 0.039291 |
| ... | ... | ... | ... | ... | ... | ... |
| 34560 | 0.006736 | 0.064911 | 0.0 | 0.010768 | 0.000000 | 0.173327 |
| 34561 | 0.000000 | 0.210688 | 0.0 | 0.000000 | 0.000000 | 0.000000 |
| 34562 | 0.077907 | 0.324336 | 0.0 | 0.000000 | 0.268368 | 0.000898 |
| 34563 | 0.121688 | 0.342804 | 0.0 | 0.014358 | 0.388244 | 0.014369 |
| 34564 | 0.007409 | 0.236748 | 0.0 | 0.011845 | 0.000000 | 0.000000 |

34564 rows × 6 columns

Features Description

var1(t-1): totalWh

var2(t-1): Global active Power

var3(t-1): Sub-metering 1

var4(t-1): Sub-metering 2

var5(t-1): Sub-metering 3

totalWh(t+24): totalWh of the next 24 hours

Figure 4.19: Inputted Time-series Data Frame

Training, validation and testing datasets were split from the 34564 rows in which the training set allocate 27652 rows while the validation and testing set allocate 3456 rows respectively. The Mean Squared Error (MSE) loss function was used to evaluate the performance of the trained model by comparing the average squared difference between the actual and predicted values. 30 epochs were defined to iterate the training process along with a batch size of 50. The training and validation loss against the epoch graph during model training can be seen in Figure 4.20 which both losses dropped and converged over the epochs. The training loss was slightly larger than the validation loss with a 0.0014 difference, indicating that the model learned well and could generalize new, unseen data represented by the small validation loss. The convergence takes place at approximately epoch 25 which indicates that the model is fully trained and its performance would not improve further.

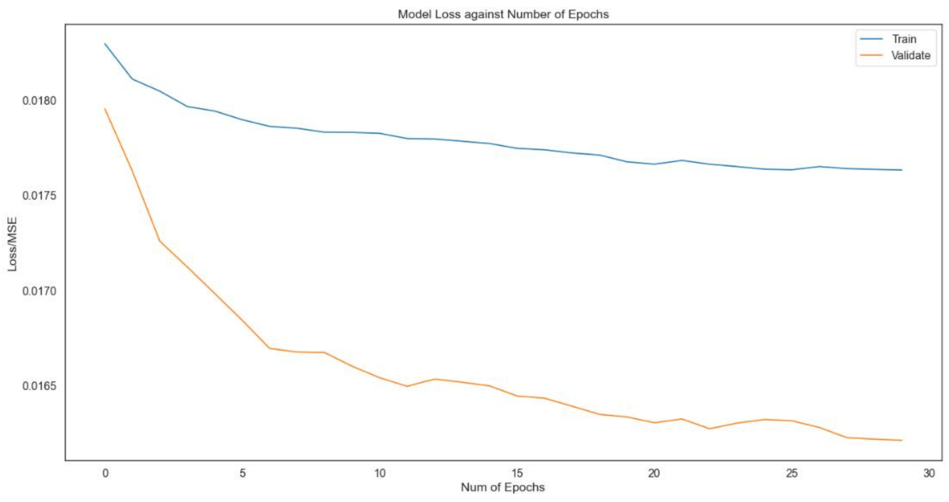


Figure 4.20: Number of Epochs vs Model Loss

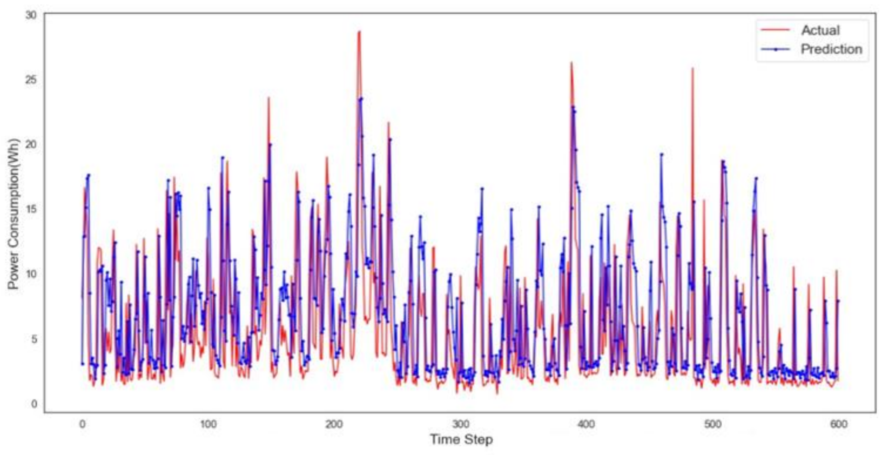


Figure 4.21: Actual vs Predicted Wh for 600 hours

The performance of the trained LSTM model on the test data can be evaluated using the Root Mean Squared Error (RMSE) metric as recommended by Hasanah et al. (2020). MSE and RMSE both are commonly used metrics to measure the mean squared difference between the predicted and actual values. RMSE is a more interpretable metric than the MSE because it is converted to the unit as the predicted and actual values. The RMSE is the square root of MSE as shown in Equations 4.7 and 4.8. The predicted and actual test data (in Wh) are concatenated and 600 hours of data are plotted in Figure 4.21. The predicted values follow the same trend as the actual data, but it performed sub-optimally in predicting high values of power consumption which the lack of high-power reading in the datasheet could be a contributing factor. However, the model might have underfitted the data due to the small training dataset of only 27652 rows and simple features used, resulting in insufficient information for the model to learn and accurately predict values. The RMSE (in Wh) in this project returned an approximate actual value of 8.671 in the data range of [0,30].

$$MSE = \frac{1}{n}\sum(y\_pred - y\_actual)^2 \tag{4.7}$$

$$RMSE = \sqrt{MSE} \tag{4.8}$$

where
*y_pred* = Predicted Wh of Test Data
*y_actual* = Actual Wh of Test Data
*n* = Number of samples

The performance of this AI model should be evaluated based on the application it is meant for as the acceptable error range may vary depending on the particular use case. For example, a small RMSE may not be important if it is used for informational purposes in which a rough estimation of load demand is provided to a homeowner for managing their electricity bill. However, a small RMSE is crucial for industries seeking energy optimization. In this case, the predicted power usage can help to reduce carbon footprint and minimize the risk of equipment failure by indicating the condition of a particular equipment if the predicted power usage is significantly lower or higher than the actual power usage.

## 4.6 Summary

This section presents the hardware prototype, the Node-red dashboard used for visualization, the AI model for predicting the next 24-hour data and the evaluation of the smart socket system's performance and power consumption. It is found that the ACS712 current sensor is sensitive to noise and has limited sensitivity making it unsuitable for accurately measuring low-power appliances. The mechanical relay used in the system consumes more power due to the movement of its contact and the continuous supply of current required to maintain the contact position. Moreover, the power consumption of both the server and smart socket can be influenced by the status of Wi-Fi connectivity and the MQTT protocol. Additionally, the power consumption of the smart socket is higher with the relay turned on than when it is off. Lastly, the developed AI model has an RMSE of 8.671 within the data range of [0,30] and the level of satisfaction with the performance depends on its intended application.

# CHAPTER 5

# CONCLUSIONS AND RECOMMENDATIONS

## 5.1 Conclusions

In conclusion, this project successfully constructed an IoT-based smart socket system that is suitable for domestic, commercial and industrial settings. The system was able to remotely control and monitor appliances connected to the smart sockets. A server was setup to facilitate the connection of multiple smart sockets through the use of the lightweight MQTT network protocol making data transfer and connectivity more accessible. The visualization of the smart socket is also available on the created user interface on the Node-red dashboard. Moreover, the system also deployed the latest ETOU tariff scheme for commercial and industrial users. Besides, a no-load condition detection feature was implemented to improve safety especially for children. Additionally, a deep learning LSTM AI model was developed and trained to predict the hourly power consumption (Wh) of the next 24 hours. Finally, the project was able to create two functional smart sockets with a daily power consumption of 124.56 W each.

## 5.2 Recommendations for future work

The ACS712 current sensor in this project has been found to fluctuate and sensitive to noises particularly when measuring low power appliances. To address the issue, the TLE4971 magnetic current sensor developed by Infineon can be installed in the smart socket to enhance the accuracy of the current measurement. The sensor is equipped with an overcurrent protection feature and has a measurement error of only ±2% with a current measurement range between 25A to 120A, depending on the model chosen. It is a small and compact sensor that saves space; however, it cost a slightly higher price than the ACS712 current sensor since it performs better.

The mechanical relay used in this project consumes more power due to its moving part and the continuous energization of contact. Hence, relays such as the combination of TRIAC and optocoupler can be used to consume less power. However, TRIAC may exhibit leakage current during its OFF state due to its internal material composition. Additionally, it is also sensitive to high temperatures which may escalate the leakage current issue.

Furthermore, future improvements to the system can include the deployment of the AI model directly to the cloud which would enhance the user experience. In addition, the AI model could be retrained using each user's power consumption data allowing it to learn and improve the accuracy of its predictions for each user.

# REFERENCES

Agarwal, T., 2021. 5V relay module: Pin Configuration, circuit, working & its applications. *ElProCus*. Available at: https://www.elprocus.com/5v-relay-module/ [Accessed: March 15, 2023].

Al-Hassan, E. et al., 2018. Improved smart power socket for monitoring and controlling electrical home appliances. *IEEE Access*, 6, pp.49292–49305.

Asurion, 2022. Smart plugs: What they do and how to best use them in your home. *Asurion*. Available at: https://www.asurion.com/connect/tech-tips/smart-plugs-what-they-do-and-how-to-best-use-them-in-your-home/ [Accessed June 30, 2022].

Blanco-Novoa, Ó. et al., 2017. An electricity price-aware open-source smart socket for the Internet of Energy. *Sensors*, 17(3), p.643.

Business Wire, 2021. Global smart electricity meters market 2020-2021 & 2027: Smart meters being the need of the hour will reach $16.4 billion by 2027 - researchandmarkets.com. *Business Wire*. Available at: https://www.businesswire.com/news/home/20210409005132/en/Global-Smart-Electricity-Meters-Market-2020-2021-2027-Smart-Meters-Being-the-Need-of-the-Hour-Will-Reach-16.4-Billion-by-2027---ResearchAndMarkets.com [Accessed July 3, 2022].

Chen, H., Canizares, C.A. & Singh, A., 2001. Ann-based short-term load forecasting in electricity markets. *2001 IEEE Power Engineering Society Winter Meeting. Conference Proceedings (Cat. No.01CH37194)*.

Chupong, C. & Plangklang, B., 2017. Electricity bill forecasting application by Home Energy Monitoring System. *2017 International Electrical Engineering Congress (iEECON)*.

Components101, 2018. *HLK-PM01 AC to DC 5V Power Module Components101*. Available at: https://components101.com/regulators/hlk-pm01-ac-dc-5v-power-module [Accessed: March 15, 2023].

Deng, Z. et al., 2020. Smart plug 2.0: Solid state smart plugs preventing fire and shock hazards in smart homes and offices. *2020 IEEE Energy Conversion Congress and Exposition (ECCE)*.

EG Projects, How to measure current using Arduino and ACS712 current sensor. *Engineers Garage*. Available at: https://www.engineersgarage.com/acs712-current-sensor-with-arduino/ [Accessed August 21, 2022].

Elicay, K. (2021) *Family mourns toddler who died of electrocution after plugging spoon into extension, SmartParenting.com.ph*. Smart Parenting. Available at: https://www.smartparenting.com.ph/health/your-kids-health/child-electrocution-a00228-20210223 [Accessed April 23, 2023].

ESPRESSIF, 2022. ESP32-DEVKITC V4 getting started guide. *ESPRESSIF*. Available at: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/hw-reference/esp32/get-started-devkitc.html#:~:text=ESP32%2DDevKitC%20V4%20is%20a,DevKitC%20V4%20on%20a%20breadboard.&text=male%20or%20female%20pin%20headers. [Accessed August 20, 2022].

Genpact, n.d. The evolution of forecasting techniques. *Genpact*. Available at: https://www.genpact.com/insight/the-evolution-of-forecasting-techniques-traditional-versus-machine-learning-methods [Accessed July 16, 2022].

Hasan, M., 2022. State of IOT 2022: Number of connected IOT devices growing 18% to 14.4 billion globally. *IoT Analytics*. Available at: https://iot-analytics.com/number-connected-iot-devices/#:~:text=In%202021%2C%20IoT%20Analytics%20expects,than%2027%20billion%20IoT%20connections. [Accessed June 30, 2022].

Hasanah, R.N., Ravie O.M.P., R.P. & Suyono, H., 2020. Comparison analysis of electricity load demand prediction using recurrent neural network (RNN) and vector autoregressive model (VAR). *2020 12th International Conference on Electrical Engineering (ICEENG)*.

Josephine, M.M. & Ikechukwu, G.A., 2017. Performance of Surge Arrester Installation to enhance protection. *Advances in Science, Technology and Engineering Systems Journal*, 2(1), pp.197–205.

Kaodim, 2021. 4 reasons why your electricity bill is so high. *Property | free Malaysia today (FMT)*. Available at: https://www.freemalaysiatoday.com/category/category/leisure/property/ [Accessed July 1, 2022].

Kirk, R., 2021. Disadvantages of smart meters: Smart meter problems. *Switchcraft*. Available at: https://www.switchcraft.co.uk/energy/smart-meters/disadvantages-of-smart-meters/ [Accessed July 9, 2022].

Lee, J.-S., Su, Y.-W. & Shen, C.-C., 2007. A comparative study of wireless protocols: Bluetooth, UWB, Zigbee, and Wi-Fi. *IECON 2007 - 33rd Annual Conference of the IEEE Industrial Electronics Society*.

Majano, D. (2021) *How protected are children from electric shock? A tamper resistant receptacle survey, Electrical Safety Foundation International*. Available at: https://www.esfi.org/how-protected-are-children-from-electric-shock-a-tamper-resistant-receptacle-survey/ [Accessed: April 23, 2023].

Manyinsa, D., 2021. What is a smart home protocol? *MUO*. Available at: https://www.makeuseof.com/smart-home-protocols/#:~:text=What%20Does%20a%20Smart%20Home,central%20hub%20or%20controlling%20unit. [Accessed June 30, 2022].

Marr, B., 2018. What are artificial neural networks - a simple explanation for absolutely anyone. *Forbes*. Available at: https://www.forbes.com/sites/bernardmarr/2018/09/24/what-are-artificial-neural-networks-a-simple-explanation-for-absolutely-anyone/?sh=2b6fff981245 [Accessed July 16, 2022].

Mateo, C. et al., 2021. Design flow and implementation of an IOT smart power socket. *2021 IEEE International Conference on Electro Information Technology (EIT)*.

Muzaffar, S. and Afshari, A. (2019) "Short-term load forecasts using LSTM Networks," *Energy Procedia*, 158, pp. 2922–2927. Available at: https://doi.org/10.1016/j.egypro.2019.01.952.

Naidu, G.A. & Kumar, J., 2019. Wireless protocols: Wi-Fi Son, bluetooth, zigbee, Z-wave, and Wi-Fi. *Lecture Notes in Networks and Systems*, pp.229–239.

Qualcomm, n.d. Qualcomm Wi-Fi SON and Distributed Networking. *Qualcomm*. Available at: https://www.qualcomm.com/products/features/wi-fi-son [Accessed July 16, 2022].

Rehman, A.-ur et al., 2020. IOT-enabled smart socket. *Wireless Personal Communications*, 116(2), pp.1151–1169.

Kumbhar, D., Taur, S., Chaudhari, H., & Bhatambrekar, S., 2018. *IoT Based Home Security System Using Raspberry Pi-3*. International Journal of Pure and Applied Mathematics, 119(12), 305-311.

Rogers, C.D., 2020. What are the effects of overusing energy? *Home Guides | SF Gate*. Available at: https://homeguides.sfgate.com/effects-overusing-energy-78753.html [Accessed July 9, 2022].

Sadownik, R. & Barbosa, E.P., 1999. Short-term forecasting of industrial electricity consumption in Brazil. *Journal of Forecasting*, 18(3), pp.215–224.

Sharmila, M. et al., 2020. Designa and development of automated smart socket for Wi-Fi Users. *2020 International Conference on Inventive Computation Technologies (ICICT)*.

Shi, H., Xu, M. & Li, R., 2018. Deep learning for household load forecasting—a novel pooling deep RNN. *IEEE Transactions on Smart Grid*, 9(5), pp.5271–5280.

Sunrun, 2021. Time-of-use (TOU). *Sunrun*. Available at: https://www.sunrun.com/go-solar-center/solar-terms/definition/time-of-use#:~:text=Time%2Dof%2Duse%20metering%20is,vary%20by%20region%20and%20utility. [Accessed June 30, 2022].

The world counts, 2022. Global energy consumption only going up. *The world counts*. Available at: https://www.theworldcounts.com/challenges/climate-change/energy/global-energy-consumption [Accessed June 30, 2022].

theSun, 2019. Electrical malfunctions common cause of residential fires during Raya. *www.thesundaily.my*. Available at: https://www.thesundaily.my/local/electrical-malfunctions-common-cause-of-residential-fires-during-raya-FG932689 [Accessed July 1, 2022].

TNB, TNB enhanced time of use (etou) - tenaga nasional berhad. *TNB Better. Brighter.* Available at: https://www.tnb.com.my/faq/etou/ [Accessed July 1, 2022].

Tsai, K.-L., Leu, F.-Y. & You, I., 2016. Residence Energy Control System based on wireless smart socket and IOT. *IEEE Access*, 4, pp.2885–2894.

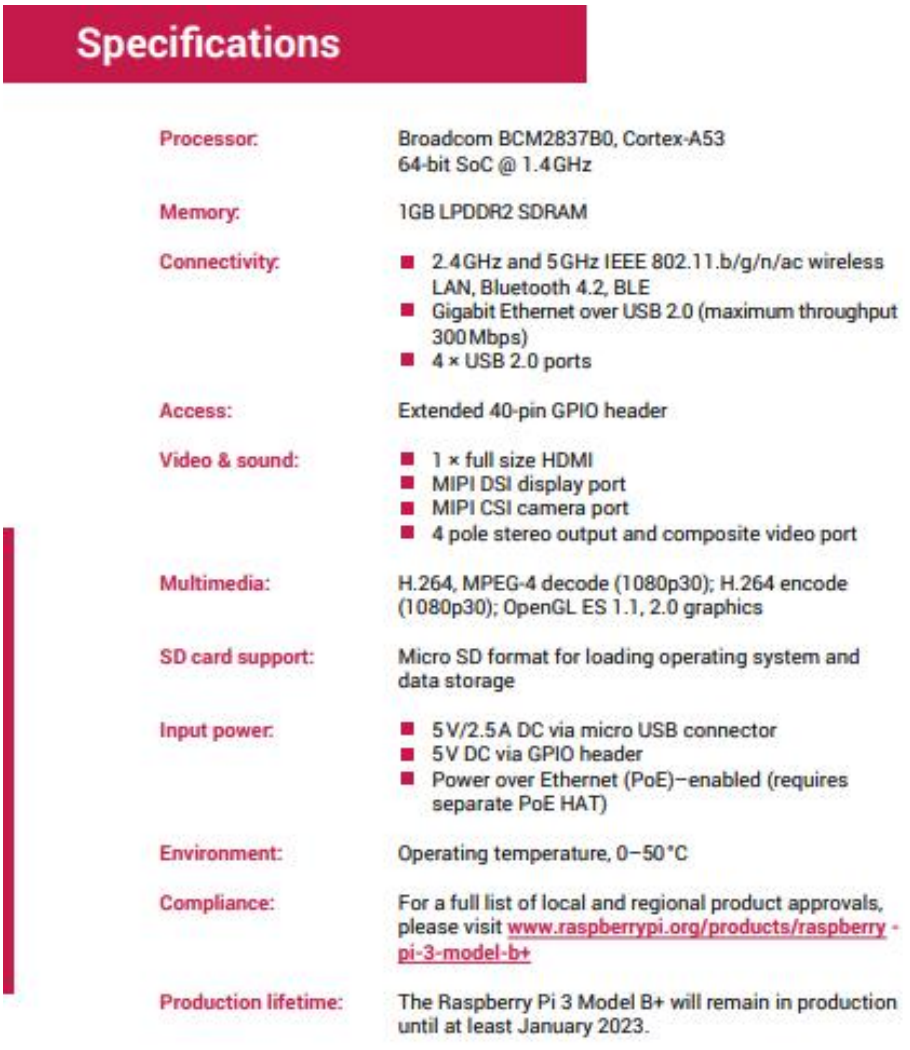
UCI, n.d. *Individual household electric power consumption data set*. *UCI Machine Learning Repository*. Available at: http://archive.ics.uci.edu/ml/datasets/Individual+household+electric+power+consumption (Accessed: March 18, 2023).

Wahyudi, R.A., Saripudin, A. & Budi, A.H., 2018. Simulation of design and implementation of smart socket prototype controlled by Android application. *IOP Conference Series: Materials Science and Engineering*, 384, p.012060.

Winchester, L., 2021. Eight appliances that will drain bills if you don't switch them off by the Plug. *mirror*. Available at: https://www.mirror.co.uk/money/eight-items-drain-your-bills-25260825 [Accessed July 9, 2022].

# APPENDICES

## Appendix A: Raspberry Pi 3 Model B+ Datasheet



### Specifications

| | |
|---|---|
| Processor: | Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4 GHz |
| Memory: | 1GB LPDDR2 SDRAM |
| Connectivity: | ■ 2.4 GHz and 5 GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE ■ Gigabit Ethernet over USB 2.0 (maximum throughput 300 Mbps) ■ 4 × USB 2.0 ports |
| Access: | Extended 40-pin GPIO header |
| Video & sound: | ■ 1 × full size HDMI ■ MIPI DSI display port ■ MIPI CSI camera port ■ 4 pole stereo output and composite video port |
| Multimedia: | H.264, MPEG-4 decode (1080p30); H.264 encode (1080p30); OpenGL ES 1.1, 2.0 graphics |
| SD card support: | Micro SD format for loading operating system and data storage |
| Input power: | ■ 5 V/2.5 A DC via micro USB connector ■ 5 V DC via GPIO header ■ Power over Ethernet (PoE)–enabled (requires separate PoE HAT) |
| Environment: | Operating temperature, 0–50 °C |
| Compliance: | For a full list of local and regional product approvals, please visit www.raspberrypi.org/products/raspberry -pi-3-model-b+ |
| Production lifetime: | The Raspberry Pi 3 Model B+ will remain in production until at least January 2023. |

raspberrypi.org

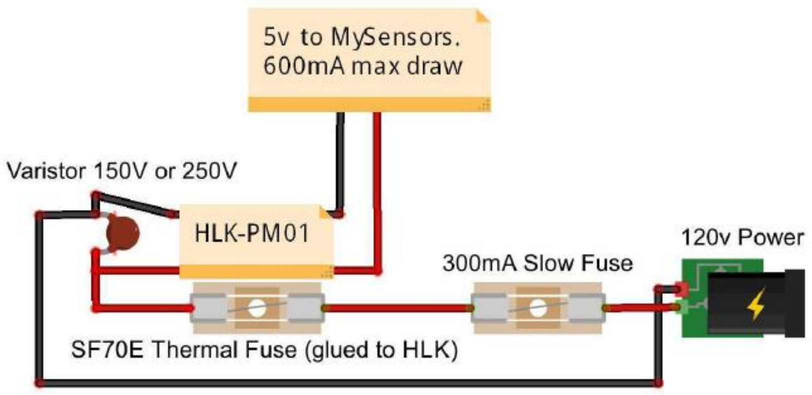
# Appendix B: HLK-PM01 AC-DC Converter Datasheet

**Electrical Characteristics**

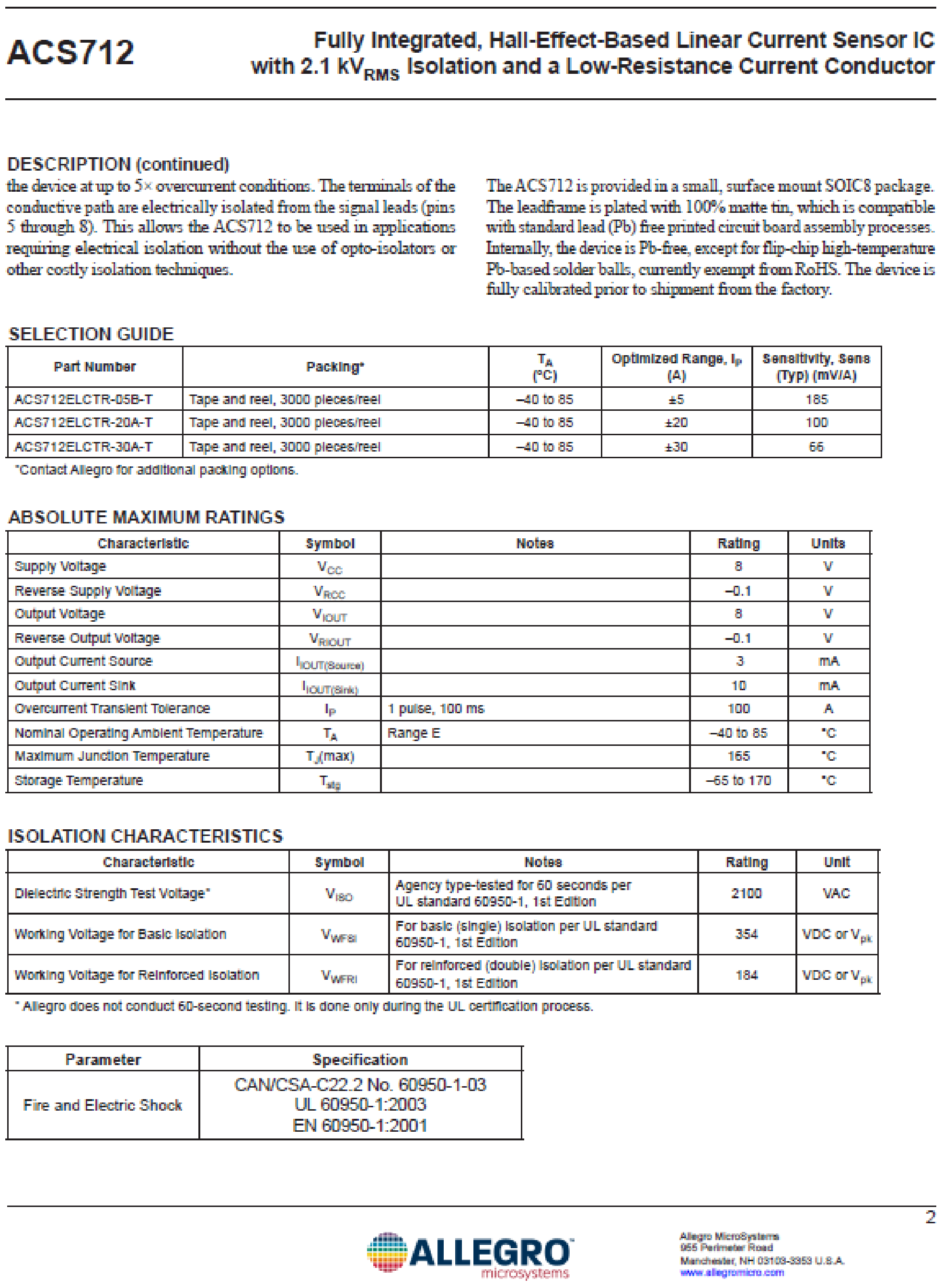
- Rated input voltage: 100-240 VAC
- Input voltage range: 90-264 VAC
- Maximum input voltage: ≤270 VAC
- Maximum input current: ≤0.2 A
- Input current surge: ≤10 A
- Enter slow start: ≤50 ms
- Input Low Voltage Efficiency: Vin=110VAC， Output full-load≥69%
- Input High Voltage Efficiency: Vin=220VAC， output full-load≥70%
- Long-term reliability: MTBF≥100， 000 h
- Load rated output voltage: +5 VDC ±0.1
- Full rated output voltage: +5 VDC ±0.2
- Short-term maximum output current: **≥1000** mA
- The maximum output current for a long time: **≥600** mA
- Voltage Regulation: ±0.2%
- Load Regulation: ±0.5%
- Output ripple and noise (mVp-p): ≤50 mV
    - Rated input voltage, full load. Using 20MHz of bandwidth,
    - The load side **10uF and 0.1uF** capacitor to be tested.

**Safety Characteristics:**

- Designed with the input of 0.5A UL certified insurance;
- PCB board using double-sided copper clad plate production, material for the 94-V0 fire rating level;
- Safety standards: Compliance with UL1012, EN60950, UL60950
- Insulation voltage: I / P-O / P: 2500VAC
- Insulation resistance :I / PO / P> 100M Ohms / 500VDC 25 °C 70% RH
- Conduction and radiation :comply with EN55011, EN55022 (CISPR22)
- Electrostatic discharge :IEC / EN 61000-4-2 level 4 8kV / 15kV
- RF radiation Immunity: IEC / EN 61000-4-3



# Appendix C:  ACS712ELCTR-20A-T Datasheet



**ACS712** Fully Integrated, Hall-Effect-Based Linear Current Sensor IC with 2.1 $kV_{RMS}$ Isolation and a Low-Resistance Current Conductor

## DESCRIPTION (continued)

the device at up to 5× overcurrent conditions. The terminals of the conductive path are electrically isolated from the signal leads (pins 5 through 8). This allows the ACS712 to be used in applications requiring electrical isolation without the use of opto-isolators or other costly isolation techniques.

The ACS712 is provided in a small, surface mount SOIC8 package. The leadframe is plated with 100% matte tin, which is compatible with standard lead (Pb) free printed circuit board assembly processes. Internally, the device is Pb-free, except for flip-chip high-temperature Pb-based solder balls, currently exempt from RoHS. The device is fully calibrated prior to shipment from the factory.

## SELECTION GUIDE

| Part Number | Packing* | $T_A$ (°C) | Optimized Range, $I_P$ (A) | Sensitivity, Sens (Typ) (mV/A) |
|---|---|---|---|---|
| ACS712ELCTR-05B-T | Tape and reel, 3000 pieces/reel | –40 to 85 | ±5 | 185 |
| ACS712ELCTR-20A-T | Tape and reel, 3000 pieces/reel | –40 to 85 | ±20 | 100 |
| ACS712ELCTR-30A-T | Tape and reel, 3000 pieces/reel | –40 to 85 | ±30 | 66 |

*Contact Allegro for additional packing options.

## ABSOLUTE MAXIMUM RATINGS

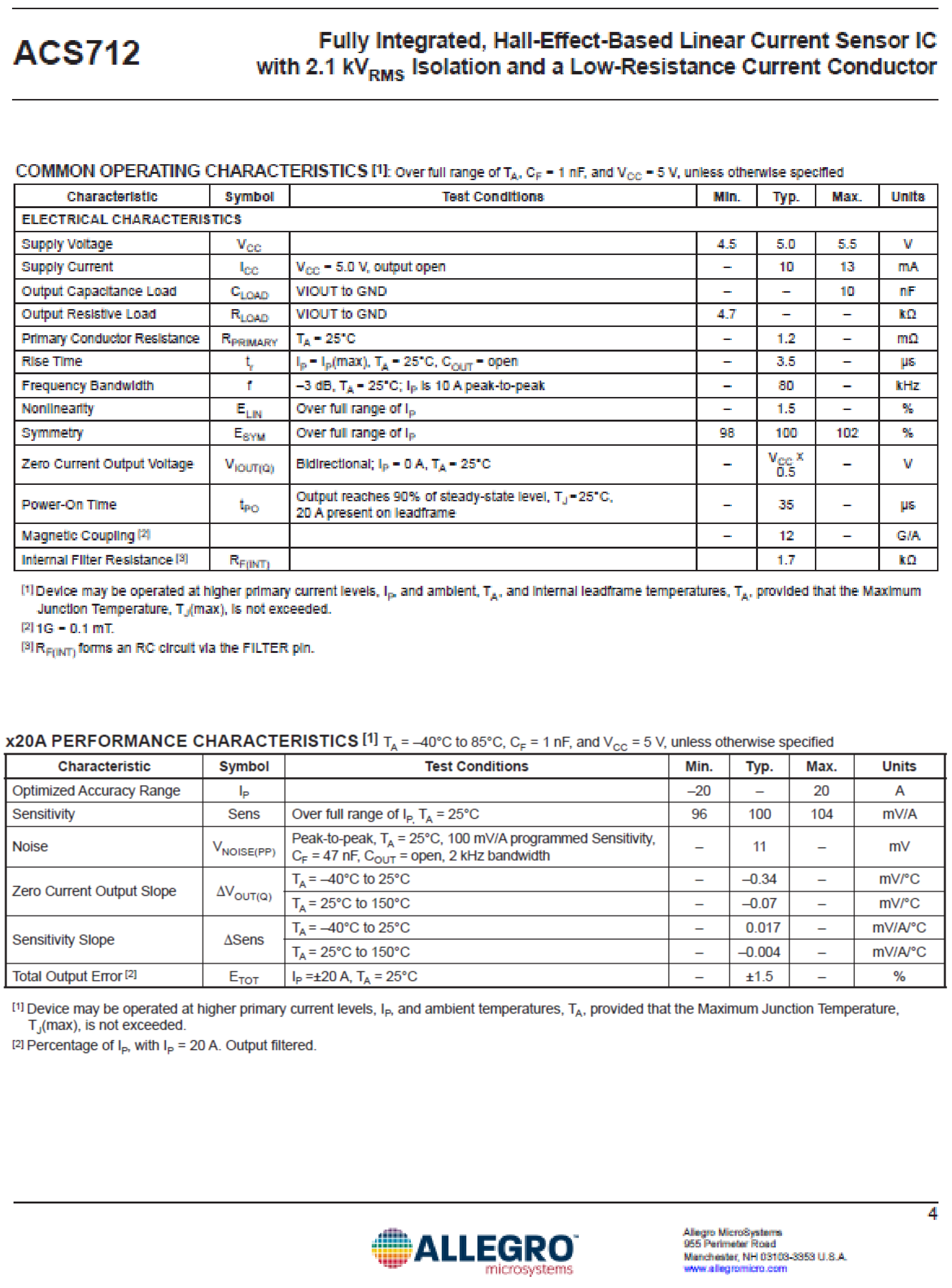
| Characteristic | Symbol | Notes | Rating | Units |
|---|---|---|---|---|
| Supply Voltage | $V_{CC}$ | | 8 | V |
| Reverse Supply Voltage | $V_{RCC}$ | | –0.1 | V |
| Output Voltage | $V_{IOUT}$ | | 8 | V |
| Reverse Output Voltage | $V_{RIOUT}$ | | –0.1 | V |
| Output Current Source | $I_{IOUT(Source)}$ | | 3 | mA |
| Output Current Sink | $I_{IOUT(Sink)}$ | | 10 | mA |
| Overcurrent Transient Tolerance | $I_P$ | 1 pulse, 100 ms | 100 | A |
| Nominal Operating Ambient Temperature | $T_A$ | Range E | –40 to 85 | °C |
| Maximum Junction Temperature | $T_J$(max) | | 165 | °C |
| Storage Temperature | $T_{stg}$ | | –65 to 170 | °C |

## ISOLATION CHARACTERISTICS

| Characteristic | Symbol | Notes | Rating | Unit |
|---|---|---|---|---|
| Dielectric Strength Test Voltage* | $V_{ISO}$ | Agency type-tested for 60 seconds per UL standard 60950-1, 1st Edition | 2100 | VAC |
| Working Voltage for Basic Isolation | $V_{WFSI}$ | For basic (single) isolation per UL standard 60950-1, 1st Edition | 354 | VDC or $V_{pk}$ |
| Working Voltage for Reinforced Isolation | $V_{WFRI}$ | For reinforced (double) isolation per UL standard 60950-1, 1st Edition | 184 | VDC or $V_{pk}$ |

* Allegro does not conduct 60-second testing. It is done only during the UL certification process.

| Parameter | Specification |
|---|---|
| Fire and Electric Shock | CAN/CSA-C22.2 No. 60950-1-03<br>UL 60950-1:2003<br>EN 60950-1:2001 |



**ACS712** Fully Integrated, Hall-Effect-Based Linear Current Sensor IC with 2.1 $kV_{RMS}$ Isolation and a Low-Resistance Current Conductor

**COMMON OPERATING CHARACTERISTICS** [1]: Over full range of $T_A$, $C_F$ = 1 nF, and $V_{CC}$ = 5 V, unless otherwise specified

| Characteristic | Symbol | Test Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| ELECTRICAL CHARACTERISTICS | | | | | | |
| Supply Voltage | $V_{CC}$ | | 4.5 | 5.0 | 5.5 | V |
| Supply Current | $I_{CC}$ | $V_{CC}$ = 5.0 V, output open | – | 10 | 13 | mA |
| Output Capacitance Load | $C_{LOAD}$ | VIOUT to GND | – | – | 10 | nF |
| Output Resistive Load | $R_{LOAD}$ | VIOUT to GND | 4.7 | – | – | kΩ |
| Primary Conductor Resistance | $R_{PRIMARY}$ | $T_A$ = 25°C | – | 1.2 | – | mΩ |
| Rise Time | $t_r$ | $I_P$ = $I_P$(max), $T_A$ = 25°C, $C_{OUT}$ = open | – | 3.5 | – | μs |
| Frequency Bandwidth | f | –3 dB, $T_A$ = 25°C; $I_P$ is 10 A peak-to-peak | – | 80 | – | kHz |
| Nonlinearity | $E_{LIN}$ | Over full range of $I_P$ | – | 1.5 | – | % |
| Symmetry | $E_{SYM}$ | Over full range of $I_P$ | 98 | 100 | 102 | % |
| Zero Current Output Voltage | $V_{IOUT(Q)}$ | Bidirectional; $I_P$ = 0 A, $T_A$ = 25°C | – | $V_{CC}$ × 0.5 | – | V |
| Power-On Time | $t_{PO}$ | Output reaches 90% of steady-state level, $T_J$ = 25°C, 20 A present on leadframe | – | 35 | – | μs |
| Magnetic Coupling [2] | | | – | 12 | – | G/A |
| Internal Filter Resistance [3] | $R_{F(INT)}$ | | | 1.7 | | kΩ |

[1] Device may be operated at higher primary current levels, $I_P$, and ambient, $T_A$, and internal leadframe temperatures, $T_A$, provided that the Maximum Junction Temperature, $T_J$(max), is not exceeded.
[2] 1G = 0.1 mT.
[3] $R_{F(INT)}$ forms an RC circuit via the FILTER pin.

**x20A PERFORMANCE CHARACTERISTICS** [1] $T_A$ = –40°C to 85°C, $C_F$ = 1 nF, and $V_{CC}$ = 5 V, unless otherwise specified

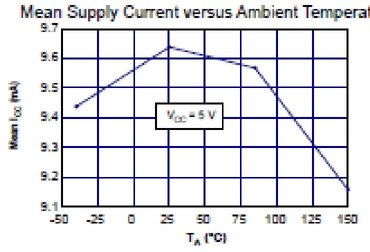
| Characteristic | Symbol | Test Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| Optimized Accuracy Range | $I_P$ | | –20 | – | 20 | A |
| Sensitivity | Sens | Over full range of $I_P$, $T_A$ = 25°C | 96 | 100 | 104 | mV/A |
| Noise | $V_{NOISE(PP)}$ | Peak-to-peak, $T_A$ = 25°C, 100 mV/A programmed Sensitivity, $C_F$ = 47 nF, $C_{OUT}$ = open, 2 kHz bandwidth | – | 11 | – | mV |
| Zero Current Output Slope | $\Delta V_{OUT(Q)}$ | $T_A$ = –40°C to 25°C | – | –0.34 | – | mV/°C |
| | | $T_A$ = 25°C to 150°C | – | –0.07 | – | mV/°C |
| Sensitivity Slope | ΔSens | $T_A$ = –40°C to 25°C | – | 0.017 | – | mV/A/°C |
| | | $T_A$ = 25°C to 150°C | – | –0.004 | – | mV/A/°C |
| Total Output Error [2] | $E_{TOT}$ | $I_P$ = ±20 A, $T_A$ = 25°C | – | ±1.5 | – | % |

[1] Device may be operated at higher primary current levels, $I_P$, and ambient temperatures, $T_A$, provided that the Maximum Junction Temperature, $T_J$(max), is not exceeded.
[2] Percentage of $I_P$, with $I_P$ = 20 A. Output filtered.

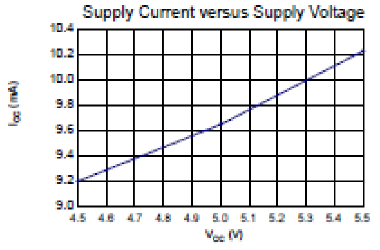
## CHARACTERISTIC PERFORMANCE

$I_P$ = 20 A, unless otherwise specified

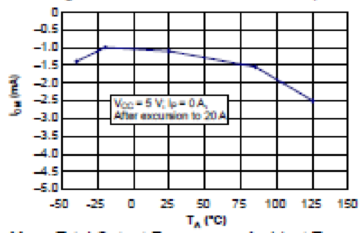
Mean Supply Current versus Ambient Temperature



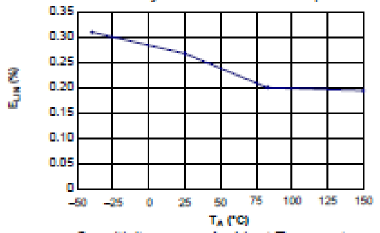
Supply Current versus Supply Voltage



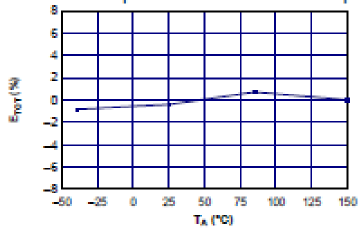
Magnetic Offset versus Ambient Temperature



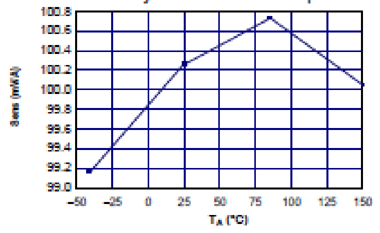
Nonlinearity versus Ambient Temperature



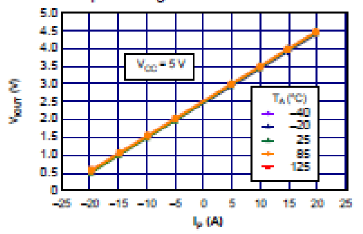
Mean Total Output Error versus Ambient Temperature



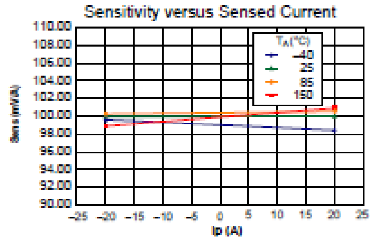
Sensitivity versus Ambient Temperature



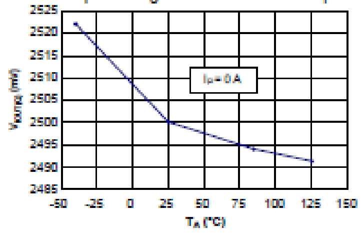
Output Voltage versus Sensed Current



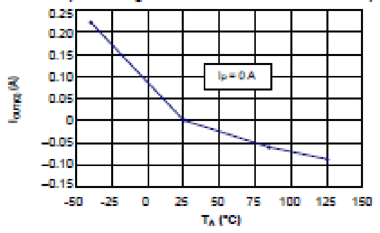
Sensitivity versus Sensed Current



0 A Output Voltage versus Ambient Temperature



0 A Output Voltage Current versus Ambient Temperature


$I_{OUT(Q)}$ (A)
$I_P$ = 0 A
$T_A$ (°C)