ANDROID-BASED
SMART ENERGY MONITORING
SYSTEM

GOH YOU SHENG

UNIVERSITI TUNKU ABDUL RAHMAN

ANDROID-BASED SMART ENERGY MONITORING SYSTEM

GOH YOU SHENG

A project report submitted in partial fulfilment of the requirements for the award of Bachelor of Science (Honours) Software Engineering

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

June 2022

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 :

Name : Goh You Sheng

ID No. : 2101659

Date : 28/4/2023

APPROVAL FOR SUBMISSION

I certify that this project report entitled **"ANDROID-BASED SMART ENERGY MONITORING SYSTEM"** was prepared by **GOH YOU SHENG** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Software Engineering with Honours at Universiti Tunku Abdul Rahman.

Approved by,

Signature :

Supervisor : See Yuen Chark

Date : 28 April 2023

Signature :

Co-Supervisor :

Date :

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to everyone for providing me with the resources and support necessary to complete this project. I would like to thank my supervisor, Dr. See Yuen Chark, for their guidance and support throughout the research process. His expertise and feedback have been invaluable in shaping this project.

In addition, I would also like to thank my friend, Leow Pei Hui, for her development of smart wall outlets. Her contributions have been essential to the success of this project. I would like to acknowledge the invaluable contribution of our testers who took the time to participate in this project, providing valuable feedback and insights that helped to improve the quality of this project.

Lastly, I am deeply grateful to my family and friends for their unwavering support and encouragement throughout this final year project journey. Their love and support have been a constant source of motivation.

ABSTRACT

With the increasing awareness of the need to protect the environment, promoting energy saving has become an urgent concern for individuals and businesses. As a household electricity user, it can be challenging to avoid energy waste at all times. This can lead to excessive energy consumption and harmful carbon emissions. Therefore, this project aims to increase energy efficiency by managing smart wall outlets via a mobile application. The project focuses on the iterative development of a mobile application using the React Native and NodeJS frameworks. This application allows users to remotely control the status of the wall outlet and receive notifications regarding the status of the wall outlet, resulting in improved energy management and conservation. The usability of the mobile application has been a vital aspect of the project, and the app features voice recognition to facilitate its use in different scenarios by different users. This project has far-reaching implications for energy management and the environment, as users no longer need to keep track of the status of dozens or even hundreds of sockets in person, leading to better energy conservation and carbon reduction. Moreover, planning the status of the wall outlets in advance through task scheduling allows for even greater energy efficiency and conservation. Overall, this project represents a step towards better environmental stewardship and responsibility. By developing an app that effectively manages smart wall outlets and reduces unnecessary energy usage, this project contributes to the ongoing efforts to protect the environment and preserve our planet for future generations.

TABLE OF CONTENTS

DECLARATION **i**

APPROVAL FOR SUBMISSION **ii**

ACKNOWLEDGEMENTS **iv**

ABSTRACT **v**

TABLE OF CONTENTS **vi**

LIST OF TABLES **x**

LIST OF FIGURES **xii**

LIST OF SYMBOLS / ABBREVIATIONS **xiv**

LIST OF APPENDICES **xv**

CHAPTER

1 **INTRODUCTION** **1**

1.1 General Introduction 1

1.2 Problem Statement 2

1.3 Aim and Objectives 3

1.4 Scope and Limitation of the Study 4

1.5 Proposed Solution 4

1.6 Proposed Methodology 5

2 **LITERATURE REVIEW** **6**

2.1 Introduction 6

2.2 Similar applications and related works 6

2.2.1 MyTNB 6

2.2.2 Mi home 9

2.3 Speech recognition 11

2.4 Machine learning for power consumption forecasting. 13

2.5 Software Development Methodology 15

2.5.1 Waterfall Methodology 15

2.5.2 Agile Methodology 17

2.5.3 Spiral Methodology 18

2.5.4 Iterative Methodology 19

2.5.5 Comparison among SDLC methodologies 20

3 METHODOLOGY AND WORK PLAN 22

3.1 Introduction 22

3.2 System Development Methodology 22

3.3 Work Plan 25

3.3.1 Work Breakdown Structure 25

3.3.2 Gantt Chart 30

3.4 Development Tools 32

3.4.1 JavaScript 32

3.4.2 React Native 32

3.4.3 Firebase 32

3.4.4 Visual Studio Code 32

3.4.5 Android Studio 33

3.4.6 Git 33

3.4.7 Picovoice 33

3.5 Conclusion 33

4 PROJECT SPECIFICATION 34

4.1 Introduction 34

4.2 Requirements Specification 34

4.2.1 Functional Requirements 34

4.2.2 Non-Functional Requirements 35

4.3 Use Case Diagram 36

4.4 Use Case Description 37

4.4.1 Login 37

4.4.2 Register 38

4.4.3 Add Houses / Factories 39

4.4.4 Design House Floor Map 40

4.4.5 Add New Wall Outlet 41

4.4.6 Set Wall Outlet Location in Room 42

4.4.7 Scan Wall Outlet QR Code 42

4.4.8 Control Wall Outlet (On & Off) 43

4.4.9 Schedule Wall Outlet On & Off 44

4.4.10 Set Energy Consumption Alert Level 45
4.4.11 Make Voice Command 46
4.4.12 View House's / Room's / Wall Outlet's Energy Consumption 47

5 SYSTEM DESIGN 49
5.1 Introduction 49
5.2 System Architecture Design 49
5.3 Database Design 51
5.3.1 Entity Relationship Diagram 51
5.3.2 Data Dictionary 53
5.4 User Interface Design 62
5.4.1 Mock Up 62

6 IMPLEMENTATION 83
6.1 Introduction 83
6.2 Frontend Implementation 83
6.2.1 Profile Module 83
6.2.2 Control Module 84
6.2.3 Monitoring Module 85
6.2.4 Voice Recognition Module 86
6.2.5 Notification Module 87
6.2.6 Property Management Module 87
6.3 Backend Implementation 88
6.3.1 Authentication 89
6.3.2 Real-time usage 89
6.3.3 Schedule 90
6.3.4 API List 91
6.4 Summary 98

7 SYSTEM TESTING 99
7.1 Introduction 99
7.2 Test Plan 99
7.3 Unit Test 102
7.3.1 Mobile Application Unit Test 104
7.3.2 APIs Unit Test 115
7.4 User Acceptance Test 123

7.5 Conclusion 124

8 CONCLUSION AND RECOMMENDATION 125

8.1 Conclusion 125

8.2 Limitations 126

8.3 Recommendation for Future Work 127

REFERENCES 129

APPENDICIES 133

LIST OF TABLES

Table 2.1 Pros and Cons of Waterfall Methodology 16

Table 2.2 Pros and Cons of Agile Methodology 17

Table 2.3 Pros and Cons of Spiral Methodology 19

Table 2.4 Pros and Cons of Iterative Methodology 20

Table 2.5 Comparison of SDLC methodologies 21

Table 5.1 Description of Database Tables 53

Table 5.2 Users Entity Data Dictionary 54

Table 5.3 Properties Entity Data Dictionary 55

Table 5.4 Rooms Entity Data Dictionary 56

Table 5.5 Wall Outlets Entity Data Dictionary 57

Table 5.6 Wall Outlet Schedules Entity Data Dictionary 58

Table 5.7 Notifications Entity Data Dictionary 59

Table 5.8 Task Schedules Entity Data Dictionary 60

Table 5.9 Property Floor Plans Entity Data Dictionary 61

Table 5.10 Room Floor Plans Entity Data Dictionary 61

Table 6.1 Keywords to execute action with examples of commands. 87

Table 6.2 Things to Adjust for Each Item in Floor Plan Editor 88

Table 6.3 APIs List of Different Backend Services 91

Table 7.1 Modules or services to be tested. 100

Table 7.2 Modules not to be tested. 101

Table 7.3 Test Case APP-1 - Register 104

Table 7.4 Test Case APP-2 - Login 106

Table 7.5 Test Case APP-3 - Edit Username 107

Table 7.6 Test Case APP-4 - Edit Password 108

Table 7.7 Test Case APP-5 - Switch Wall Outlet State 109

Table 7.8 Test Case APP-6 - Set Countdown Timer 109

Table 7.9 Test Case APP-7 - Set Alert Value 110

Table 7.10 Test Case APP-8 - View Energy Consumption Data 111

Table 7.11 Test Case APP-9 - Create New Property 112

Table 7.12 Test Case APP-10 - Create New Room 113

Table 7.13 Test Case APP-11 - Create New Wall Outlet 114

Table 7.14 Test Case API-1 - Register 115

Table 7.15 Test Case API-2 - Login 116

Table 7.16 Test Case API-3 - Create Property 117

Table 7.17 Test Case API-4 - Create Room 118

Table 7.18 Test Case API-5 - Create Wall Outlet 119

Table 7.19 Test Case API-6 - Read Wall Outlet Usage 120

Table 7.20 Test Case API-7 - Switch On Wall Outlet 120

Table 7.21 Test Case API-8 - Create Notification 121

Table 7.22 Test Case API-9 - Create Scheduled Task 122

Table 7.23 User Acceptance Test Template 123

LIST OF FIGURES

Figure 1.1 2019 Greenhouse Gas Emissions by Sector in Malaysia (Climate Watch Historical GHG Emissions, 2022) 2

Figure 1.2 Overview of the Smart Energy Monitoring System 5

Figure 2.1 User Interface of MyTNB 7

Figure 2.2 User Interface of Mi Home 9

Figure 2.3 Sample output of spectrogram 12

Figure 2.4 Waterfall Methodology (Trunkett, 2020) 16

Figure 2.5 Agile Methodology (Trunkett, 2020) 18

Figure 2.6 Spiral Methodology (Trunkett, 2020) 19

Figure 2.7 Iterative Methodology (Trunkett, 2020) 20

Figure 3.1 Iterative Methodology (Trunkett, 2020) 23

Figure 3.2 Work Breakdown Structure 28

Figure 3.3 Work Breakdown Structure (Continued) 29

Figure 3.4 Gantt Chart 30

Figure 3.5 Gantt Chart (Continued) 31

Figure 4.1 Use Case Diagram of Smart Energy Mobile System 36

Figure 5.1 System Architecture Design 50

Figure 5.2 Entity Relationship Diagram 52

Figure 5.3 UI Design of Welcome Pages 62

Figure 5.4 Login Page 63

Figure 5.5 Register Page 64

Figure 5.6 Forgot Password Page 64

Figure 5.7 Reset Password Page 65

Figure 5.8 One Time Password Page 66

Figure 5.9 Home Page 67

Figure 5.10 Property Page 67

Figure 5.11 Room Page 68

Figure 5.12 Wall Outlet Page 69

Figure 5.13 Set Schedule for Wall Outlet 69

Figure 5.14 Countdown Setting Page for Wall Outlet 70

Figure 5.15 Wall Outlet Energy Consumption Alert Setting Page 71

Figure 5.16 Energy Consumption Summary Page 72

Figure 5.17 Energy Consumption (in Price) Summary Page 73

Figure 5.18 Add Page 74

Figure 5.19 Add Property Page 74

Figure 5.20 Add Room Page 75

Figure 5.21 Scan Wall Outlet QR Code Page 76

Figure 5.22 Add Wall Outlet Page 76

Figure 5.23 Voice Assistant Page 77

Figure 5.24 Notification List Page 78

Figure 5.25 Notification Detail Page 78

Figure 5.26 Profile Page 79

Figure 5.27 Edit Profile Page 79

Figure 5.28 Profile Security Page 80

Figure 5.29 Property Floor Plan Editor Page 81

Figure 5.30 Room Floor Plan Editor Page 81

Figure 5.31 Room Floor Plan Editor Page (Locate Wall Outlet Position) 82

LIST OF SYMBOLS / ABBREVIATIONS

ADC	Analog-to-Digital Converter
ANN	Artificial Neural Network
API	Application Programming Interface
CRUD	Create, Read, Update and Delete
DBMS	Database Management System
ERD	Entity Relationship Diagram
ETOU	Enhanced Time of Use
HMM	Hidden Markov Model
IDE	Integrated Development Environments
JWT	JavaScript Object Notation Web Token
k-NN	k-Nearest Neighbour
NLP	Natural Language Processing
OTP	One Time Password
POS	Part-of-speech
RAD	Rapid Application Development
RM	Ringgit Malaysia
SDLC	Software Development Life Cycle
SVG	Scalable Vector Graphics
SVM	Support Vector Machine
TNB	Tenaga Nasional Berhad
UAT	User Acceptance Test
XP	Extreme Programming

LIST OF APPENDICES

APPENDIX A : Results of User Acceptance Test 133

CHAPTER 1

INTRODUCTION

1.1 General Introduction

The energy demand worldwide is increasing alongside the growth of the economy and the population. A large number of greenhouse gases are emitted during the generation of electricity, which contributes to global warming. Figure 1.1 shows that according to Climate Watch Historical GHG Emissions (2022), the energy sector contributed 64.3% of greenhouse gas emissions in Malaysia in 2019. To realize the seriousness of global warming, many countries, including Malaysia, sign the Paris Agreement to reduce global greenhouse gas emissions. Therefore, Malaysia should reduce 45% of greenhouse gas emissions by 2030 (United Nations Framework Convention on Climate Change, 2015). Efficient use to reduce the activity of electricity generation can be an effective solution to reach the goal. As a result, Tenaga Nasional Berhad (TNB) has proposed Enhanced Time of Use (ETOU) to encourage customers to manage their electric usage efficiently by consuming less electricity during peak hours. The reason for flattening the electricity demand curve is that the cost of electricity generation disproportionately increases during peak hours, while electricity is lost during low peak hours due to the need to transmit and store the electricity generated (Barker, et al., 2012).

As a household electricity user, it is impractical to remember the tariff scheme of ETOU and walk to the wall outlet one by one to turn it off. However, with the advancement of science and technology, the smart home concept was introduced, and thus the smart wall outlet was born. Unlike a conventional wall outlet, a smart wall outlet can remotely control its status (ON/OFF) while monitoring the energy consumption (Mateo, et al., 2021). With these smart wall outlets, it is no longer an issue to develop an electricity usage scheduling and adjust the status of the smart wall outlets according to the tariff scheme. This arrangement can avoid excess usage of electricity during peak hours.

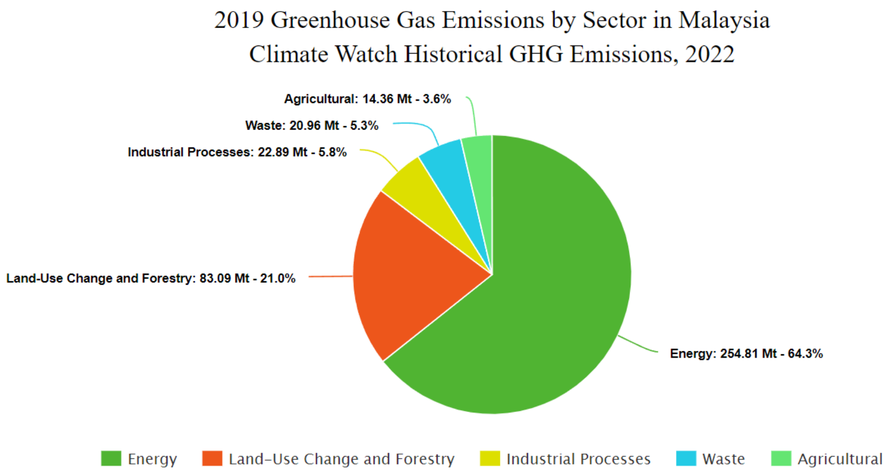


Figure 1.1 2019 Greenhouse Gas Emissions by Sector in Malaysia (Climate Watch Historical GHG Emissions, 2022)

1.2 Problem Statement

A smart energy monitoring application should be accessible anytime and anywhere. Therefore, smart energy monitoring applications must be accessible by smartphones to fulfil this commitment. According to research by Rainie and Zickuhr (2015), 90% of smartphone owners carry their phones with them frequently. This is because smartphones have high portability. There are two solutions to developing smartphone applications: mobile applications and mobile websites. However, mobile websites are restricted in functionality because sometimes they cannot effectively collaborate with smartphone hardware and operating system (iTrobes, 2021). For example, mobile websites are not easy when sending notifications. Consequently, users cannot get a notice when appliances consume unnecessary energy. By lifting the limitation of a mobile website with a mobile application, user can control their home appliances with a better user experience. Therefore, developing a mobile application that obtains real-time information from smart wall outlets is necessary.

Most elders cannot follow technological progress. Some of them cannot use electronic devices such as smartphones. Incapable of using a smartphone has caused them to face many problems in life (Ouyang and Zheng, 2022). Among the issues of elders who live alone, wall outlets are left open for subsequent use due to mobility problems. This results in energy wastage. Even if smart wall outlets are installed, the mobile application may still be some problems for the elders. According to Yusof, Romli, and Yusof (2014), small buttons and text are problems in mobile applications due to elders' poor visual ability. Moreover, the cluttered interface and the countless buttons add to the learning cost. All the extra efforts needed to leave elders with a lack of desire to learn. Therefore, this project will aim to provide an additional operation in the app - speech recognition for voice command - to facilitate users to control smart wall outlets. Speech recognition can assist in completing simple tasks that can be extremely difficult for older people by speaking to a smart device. (AgingInPlace, 2022).

With the application of smart wall outlets, it becomes easier to collect data on energy consumption compared to conventional sockets. Therefore, it is vital to have good use of these data. The collected data can be used to analyze energy efficiency, identify user behaviour patterns, and forecast energy consumption trends. However, based on Goldberg (2018), only 1% of the IoT data obtained has ever been utilized. Energy mobile applications such as myTNB only present the daily electricity bill calculated based on the consumed energy obtained through the Smart Meter. They don't do details data analysis such as time series analysis, data mining, and machine learning. In the survey by Lee (2021), most people said that it doesn't make a big difference whether they use the app or not. Therefore, the machine learning technique will be used in the proposed applications to improve energy efficiency by turning raw data into useful information. As a result, people who use the smart energy monitoring application can gain benefits such as savings in electricity bills.

1.3 Aim and Objectives

The primary goal of this project is to enhance energy efficiency by promoting the utilization of intelligent wall outlets through the integration of mobile applications. In order to achieve this goal, the following objectives had to be met:

- To develop a mobile application that allows remote control of the status (ON/OFF) and monitors the real-time power consumption of smart wall outlets.
- To implement speech recognition for voice command.
- To notify the user when the energy consumption of a wall outlet has exceeded a predetermined level.

1.4 Scope and Limitation of the Study

This project focuses on developing an Android-based smart energy monitoring mobile application that integrates several vital features. Among the features are remote control and monitoring of the real-time power consumption, forecast energy consumption with a machine learning technique, and speech recognition as an alternative way to control the smart wall outlets.

On the other hand, the smart wall outlet's hardware development is not addressed in this project. The smart wall outlets upload the energy consumption data and its status to Firebase's real-time database. The mobile application retrieves data from the database for presentation and analysis purposes.

1.5 Proposed Solution

An Android-based mobile application for smart energy monitoring will be created in this project. to ease the management of smart wall outlets in a house/building. The proposed application will be able to manage the status and energy consumption of multiple smart wall outlets simultaneously. Figure 1.2 shows the overview of the smart energy monitoring system. The smart energy monitoring system can be separated into hardware and software. The energy consumption data will be collected from the smart wall outlets that will later be stored in the home server, Raspberry Pi. At the same time, the data on energy consumption will also be synchronised by the Raspberry Pi with the Firebase real-time database. The proposed application will retrieve the consumption data with the Firebase API gateway and update data based on the user's action. Picovoice speech recognition platform is used to achieve voice command features. Picovoice emphasizes accuracy, privacy, security, and zero latency.

Therefore, this project selected Picovoice to perform speech recognition because it meets the project requirements.

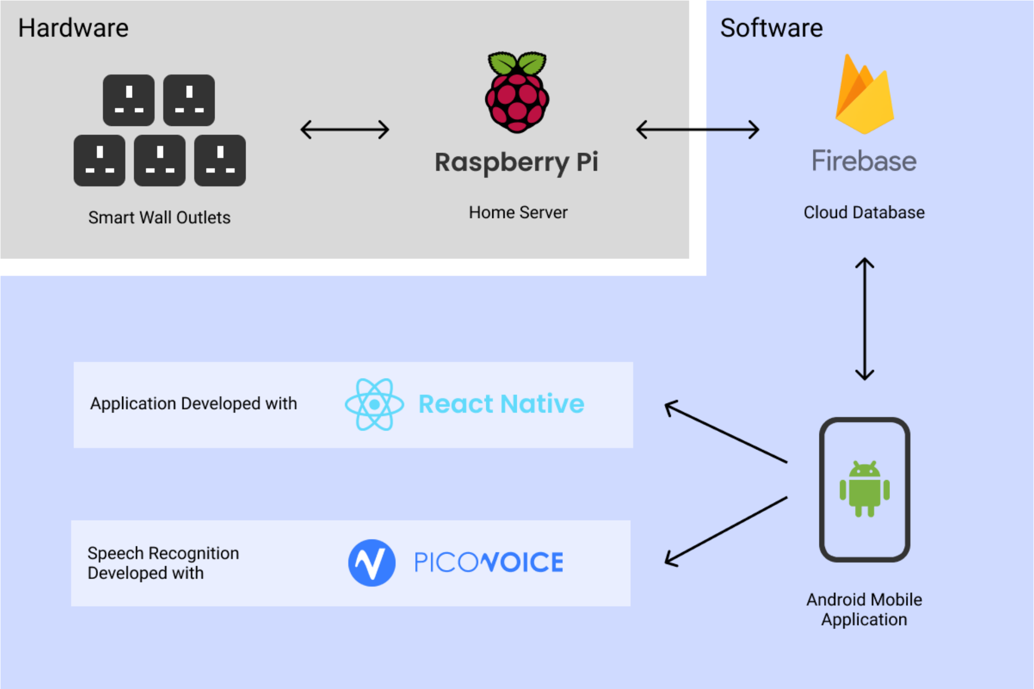


Figure 1.2 Overview of the Smart Energy Monitoring System

1.6 Proposed Methodology

The iterative methodology was proposed for this project. This is because a student from another department is in charge of developing the smart wall outlets' hardware. Therefore, it might be challenging to establish all the requirements at first. The iterative methodology has high flexibility and high adaptability to face changes. So, it is suitable to be used in this project. A more detailed proposed methodology will be discussed in Chapter 3.2.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

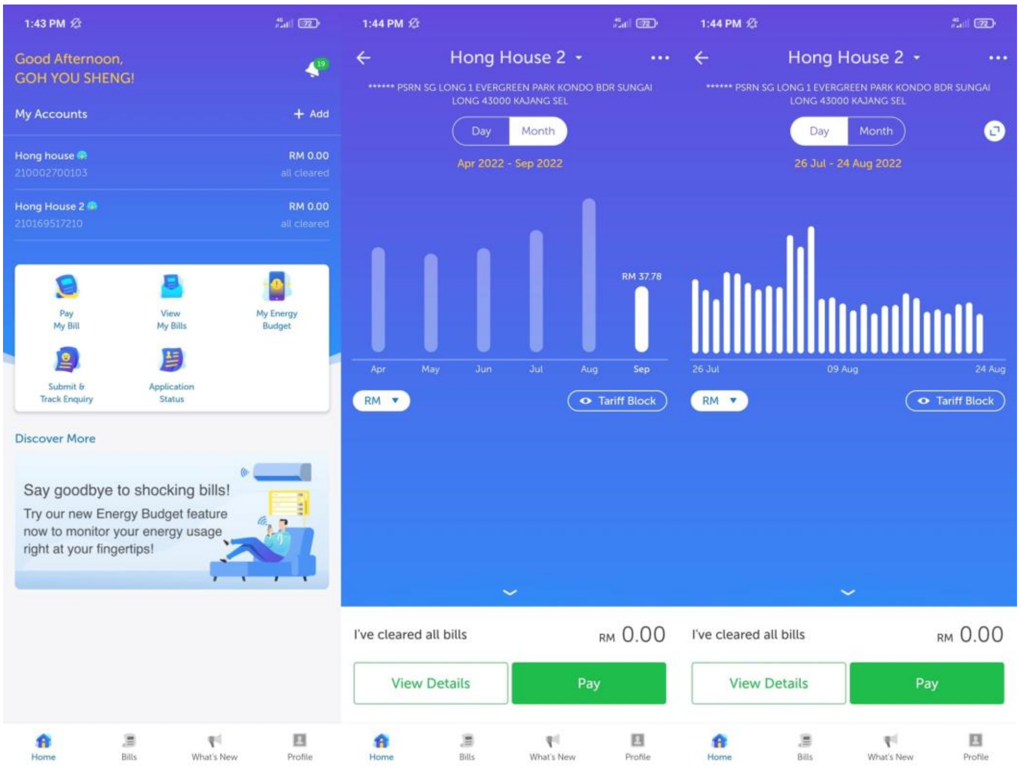
Several important pieces of knowledge need to be explained to achieve this project's objectives. In this chapter, I will first discuss several similar mobile applications available in the market. Next, I will explore speech recognition and deep learning techniques to facilitate the selection of suitable methods to accomplish corresponding features in mobile applications. Finally, several Software Development Life Cycle (SDLC) methodologies will be reviewed and compared. This project will implement the most suitable SDLC methodology after considering the circumstances of the project.

2.2 Similar applications and related works

In this section, two relevant applications were selected from the Android play store, and their design and functionality were analyzed. The project can cover more usage scenarios and finds more functional and non-functional requirements by reviewing other similar applications.

2.2.1 MyTNB

MyTNB is an application developed by Tenaga Nasional Berhad for Malaysian residents to manage TNB electricity account(s). Customers who have smart meters installed in their homes can find out their daily electricity consumption through the mobile application. Moreover, they can view and pay the bill. Figure 2.1 presents the user interface of the MyTNB application.



Say goodbye to shocking bills!
Try our new Energy Budget feature now to monitor your energy usage right at your fingertips!

Figure 2.1 User Interface of MyTNB

Main features

- Manage TNB electricity account(s)

  Users can add electricity accounts for residential or commercial premises to be managed in the application. It is necessary to add at least one account to access other application features. One myTnb account can manage multiple electricity accounts at the same time.

- View electricity consumption of electricity account

  Users can view the electricity consumption of the account on a monthly or daily basis. The electricity consumed is calculated according to the tariff block and is displayed in Ringgit Malaysia.

- Set monthly energy budget

  Users can set their target bill price for each month. The system will calculate the current electricity consumption as a target percentage. When electricity consumption exceeds the target, the application will remind the user.

- Make payment for the electricity account

The application will generate a bill based on the monthly electricity consumption. Therefore, The user can pay the bill through the application, and the supported payment methods are credit/debit cards and online banking.

Evaluation of MyTNB

MyTNB, as an application of the combination of the traditional energy industry and technology, is a good attempt with a successful outcome. By using the application, we can gradually abandon paper bills. In addition, smart meters directly obtain electricity consumption without requiring staff to go door-to-door to record customers' electricity consumption. The most important thing is that mobile apps improve the convenience of users. Several features in myTNB are advantageous and suitable to implement in this project.

The first feature is the presentation of the electricity consumption. It is appropriate and concise with the use of bar graphs. The higher the bar indicates more electricity is consumed in the month or day. Moreover, each bar will be marked with different colours representing the tariff block. Dark blue is for the first 200 units, light blue is for the 201st to 300th units, and yellow is for the 301st to 600th units. This graph presentation makes users realize that the more electricity they use, the more expensive the price per unit. Despite this, there is some delay in the display of daily electricity consumption. Users can only check the records from 2 days ago instead of the current moment.

The second feature is the monthly budget setting. A monthly budget setting feature is supportive because electricity consumption always happens without notice. Many people also lack the concept of electricity price calculation, which causes the price to be higher than expected whenever we receive a bill. Therefore, this feature can proactively remind users before going over budget. The monthly budget setting feature lets users be alert and reduce electricity waste.

However, there are some differences in MyTNB with the application this project intended to develop. Those differences are MyTNB views a house as a whole, and the electricity consumption is measured by a smart meter but not by smart wall outlets. In conclusion, despite the slight difference in requirements, myTNB's user interface design and the features to set a budget can be referenced and adapted to this project.

2.2.2 Mi home

Mi Home is a centralized management and operation application that supports various smart devices. The smart devices include wall outlets, lighting, sensors, cameras, appliances, etc. Regarding smart devices, this section will only focus on reviewing the features that Mi Home offers to smart wall outlets. Figure 2.2 depicts the user interface of the Mi Home application.

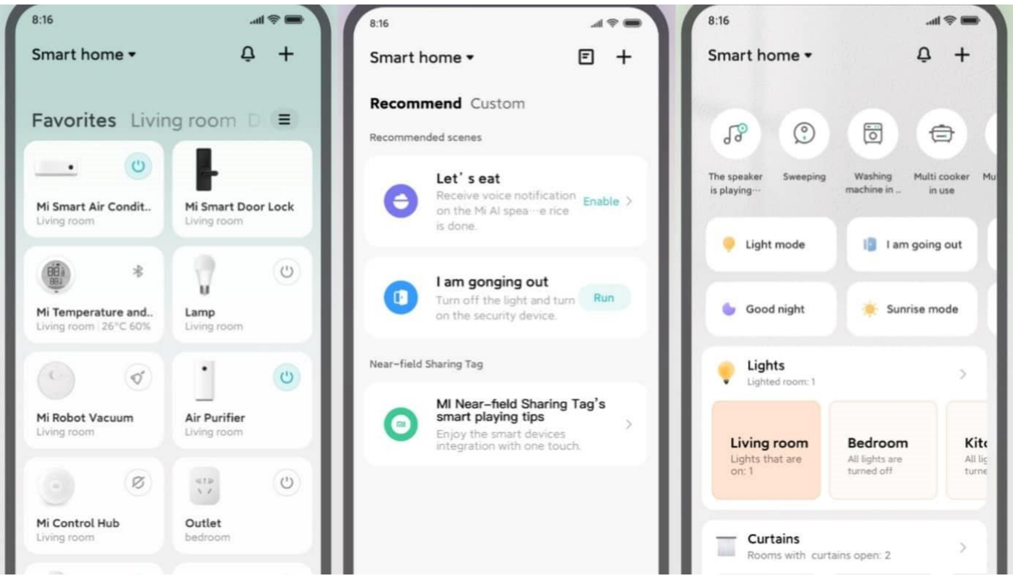


Figure 2.2 User Interface of Mi Home

Main features

- Manage homes and rooms.
  Users can add multiple homes. In each home, users can add various rooms. Thus, users can manage the smart devices according to their location with the house and room classification. This feature also makes it easy to find the corresponding smart devices in the application.
- Add smart devices.

Mi Home application provides two ways to add smart devices. The first method is by scanning the QR code on the smart device. The second method automatically checks for nearby smart devices via Bluetooth, GPS, and Wi-Fi lists.

- Feature for smart wall outlet 1 – Remote control

  Mi Home application will display the current status of the smart wall outlets. Users can change the outlets' status by clicking the app's switch button.

- Feature for smart wall outlet 2 – Timer

  The timer feature allows multiple task events to be set in advance, and each task event contains a start and end time. The user can select which of these tasks to perform. The smart wall outlets will automatically start and close according to the time set for the task.

- Feature for smart wall outlet 3 – Countdown

  The user can turn on the countdown timer and set the countdown length. The smart wall outlet will automatically turn off when the countdown is complete.

- Feature for smart wall outlet 4 – Energy statistic

  The Mi Home app shows the real-time energy consumption data of the smart wall outlet so users can review when the outlet is working and when it is not.

Evaluation of Mi Home

As an application that connects various IoT and smart devices and has a large user base, Mi Home will emphasize the user experience. Users can also set up custom scenes in advance so that when the status of a smart device changes, the status of other smart devices can be changed accordingly. For example, suppose a house is installed with a smart door lock. When the user comes home and unlocks the smart door lock, other smart devices in the house, such as lighting, humidifiers, and other appliances, will be automatically turned on. Conversely, when the user leaves the house, the smart door lock is locked, and all devices that do not need to work will automatically turn off. This custom scene prevents us from forgetting to turn off any outlets, switches, or appliances, thus saving power.

In addition, Mi Home also provides an excellent solution to manage multiple devices with one account. That solution is through house and room. Undoubtedly, by creating house and room floor plans, users can visually find their smart devices' locations. However, considering every home has a different floor plan, simply creating a menu of houses and rooms is enough to reduce the overall complexity of the application when finding a specific smart device. Therefore, these features will be considered to be implemented in this project because they will help users manage the smart wall outlets according to the different rooms.

Other than that, the features provided by Mi Home to the smart wall outlets alone meet most users' usage scenarios. One of the most praiseworthy is the timer system. The timer feature allows the smart wall outlets to perform more tasks autonomously. It is suitable for various scenarios such as battery cars regularly charging, fish tanks oxygenating, and electric lights regularly turning on at night and off at dawn. Without these features, a smart socket is, at best, a conventional outlet with additional remote control capabilities. On the contrary, users no longer need to worry about these trivial things with the timer function.

To conclude, Mi Home is an excellent application for user interface design and features, and these pros are well worth learning. Over time, these characteristics will guarantee the excellence of the application, and the design can meet the aesthetics of most people.

2.3 Speech recognition

Speech recognition is the process where a machine or computer recognizes and converts human languages into machine-understandable forms (Nassif et al., 2019). The first step is converting human speech analogy waves to digital data. Therefore, an analogue-to-digital converter (ADC) is needed. ADC will first chunk down audio into individual sounds and convert these sounds into digital format (How Does Speech Recognition Work? Learn about Speech to Text, Voice Recognition and Speech Synthesis, 2022). In order to recognize these sounds, an individual sound's frequency, intensity, and duration are needed (Goddard, 2022). Thus, digital data of sound will further transform into a spectrogram in which the x-axis represents the time, the y-axis represents the frequency, and the intensity represents the brightness. Figure 2.3 shows the output of the spectrogram.

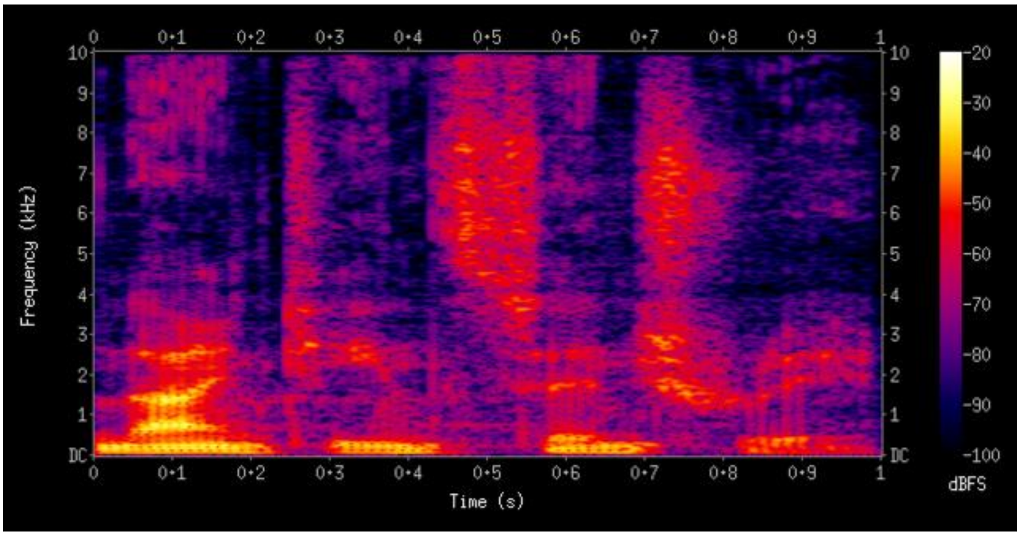


Figure 2.3 Sample output of spectrogram

Phonemes, which include vowels and consonants, are the smallest distinct unit to build up a word. Each phoneme has a different frequency and intensity. Therefore, recognized phonemes will be combined into the word in the dictionary. Finally, different words are then formed into a sentence. To ensure that the phonemes are arranged in the correct order into words and sentences, speech recognition algorithms such as Hidden Markov Model (HMM) and Natural Language Processing (NLP) will be used (Goddard, 2022).

Hidden Markov Model (HMM)

The hidden Markov model (HMM) is a statistical model that can predict the not-direct observable state by evaluating the visual events that depend on the state (Yoon, 2009). The immediately preceding state is always a factor in the present state (i2tutorials, 2019). In speech recognition, the observable events are speech or audio signals, whereas the hidden states are phonemes (i2tutorials, 2019). The model will examine the probability of recognizing a phoneme as correct (Goddard, 2022). Other than that, HMM analyses phonemes near one another and the probability that they should be nearby (Goddard, 2022). Nearby phonemes that HMM verifies will constitute words, and HMM will continue to determine if adjacent words make sense.

Natural Language Processing (NLP)

Natural Language Processing (NLP) is a popular area in artificial intelligence that allows computers to break down and understand human language. Sometimes, a word by itself is not significant and can be confused. For example, ‘leaves’ could mean exiting a place and the plural of the leaf. Big data can make NLP more intelligent and effective by gathering words, phrases, sentences, grammatical rules, and word structure for the machine to learn. For example, what words usually end up next to each other, how a sentence should be formed, and why certain words fit into a sentence better (What is NLP? Learn Natural Language Processing in Artificial Intelligence, 2022). Part-of-speech (POS) tagging is one of the techniques used by NLP to understand human language, where categorize words to a particular part of speech, such as nouns and verbs, depending on its context (Pykes, 2020).

2.4 Machine learning for power consumption forecasting.

In order to make predictions about future energy consumption, we can use smart wall outlets to collect daily or hourly consumption data as the basis of predictions. These are collected as a collection of time series data. A time series is a collection of datasets appearing continuously across time (Hayes, 2022). Unlike cross-sectional data, where each data item is independent, time series data correlates with other data items (What is Time Series Data, 2019). For example, time series data can be a company’s annual revenue, weekly oil prices, and daily stock prices. All these time series data can be used for analysis and forecasting. The techniques used for forecasting can be divided into 2 categories; the first category is the use of traditional statistical models, such as AR, MA, ARMA, ARIMA, and Prophet (Time Series Forecasting with Machine Learning, 2021). The second category is the use of machine learning.

Power consumption forecasting in the statistical model.

A statistical model forecasts future behaviour using historical data (Hayes, 2022). However, some issues still need to be addressed when using statistical models for energy consumption (Shapi, Ramli and Awalin, 2021). The first problem is that the statistical model requires a lot of computing power to make predictions on huge amounts of data (Shapi, Ramli and Awalin, 2021). In addition, the statistical model only performs well for a regular time series (Karim and Alwi, 2012). Xiangyu et al. (2019) also stated that the traditional univariate statistical model relies on only one parameter: the trend of maximum energy demand for modelling and analysis. As a result, additional important features are ignored when developing the model. The traditional statistical model must first predict the first and second days to predict the energy consumption for the subsequent three days. By combining the results of the first day's prediction, the third day's energy consumption can be predicted (Time Series Forecasting with Machine Learning, 2021).

Power consumption forecasting in machine learning

The use of machine learning does not completely solve all the problems. For example, it still has the problem of missing data. However, According to Gerasymov (2022), time series forecasting using machine learning was the most successful in identifying patterns in the sequence data. Many models in the field of machine learning can be applied to forecast energy consumption, including the k-Nearest Neighbour (k-NN), Support Vector Machine (SVM), and Artificial Neural Network (ANN). K-NN machine learning algorithm is widely utilized because of its straightforward rules and ability to forecast complex non-linear patterns (Valgaev, Kupzog and Schmeck, 2016). SVM, sometimes referred to as a maximum margin classifier, addresses classification and regression issues for huge datasets (Ben-Hur et al., 2008). Because ANN can learn complicated behaviour, it is frequently used to generate predictions and identify patterns (Karunathilake and Nagahamulla, 2017).

2.5 Software Development Methodology

The Software Development Lifecycle (SDLC) is a process that every software development goes through. It is divided into several phases: planning, analysis, design, development, testing, deployment, and maintenance. The SDLC aims to ensure that a high-quality product that meets customer expectations is produced within a predetermined time and budget (Gadhavi, 2022). The SDLC methodology is a set of practices to ensure that SDLC is implemented correctly. Many SDLC methodologies can be implemented, but each has advantages and disadvantages. No single SDLC method can be applied to all software development projects. Therefore, many factors need to be taken into consideration. The SDLC methodologies reviewed in this section are Waterfall, Agile, Spiral, and Iterative.

2.5.1 Waterfall Methodology

The waterfall methodology is the most traditional SDLC methodology. The waterfall methodology is a simple sequential approach. The waterfall methodology adheres to a strict and pre-set rule that a previous phase must be finished before moving on to the next phase. This ensures that no multiple phases are overlapped and are performed simultaneously at a point in time. As a result, it is often necessary to wait for confirmation of approval from the decision-making team and the customer. The uncertainty of the wait time may force delays resulting in later phases not following the expected schedule. Figure 2.4 shows the processes of the waterfall methodology.

Since changes are hard to be made once a phase has been confirmed if changes must be made, they require greater costs in terms of money and time. Therefore, the waterfall methodology is inflexible. In today’s complex software development environment, requirements often cannot be determined initially. New requirements are revealed as the project is developed. Consequently, the waterfall methodology is only suitable for developing simple software projects with certain requirements. In addition, since testing can only be done after development is complete, bugs cannot be detected early and therefore require more effort to fix. Table 2.1 lists the pros and cons of the waterfall methodology.

Table 2.1 Pros and Cons of Waterfall Methodology

Pros	**Cons**
Easy to understand and implement	No instant errors repairing mechanism.
Easy to verify that progress is in line with the planned schedule	Unable to adapt to requirements and make changes
The finished product of each phase is determined and can be assessed to confirm whether the next phase can begin	Changes require higher cost and effort
The information of every phase is well documented and can be used in the next phase	Documentation takes a lot of time

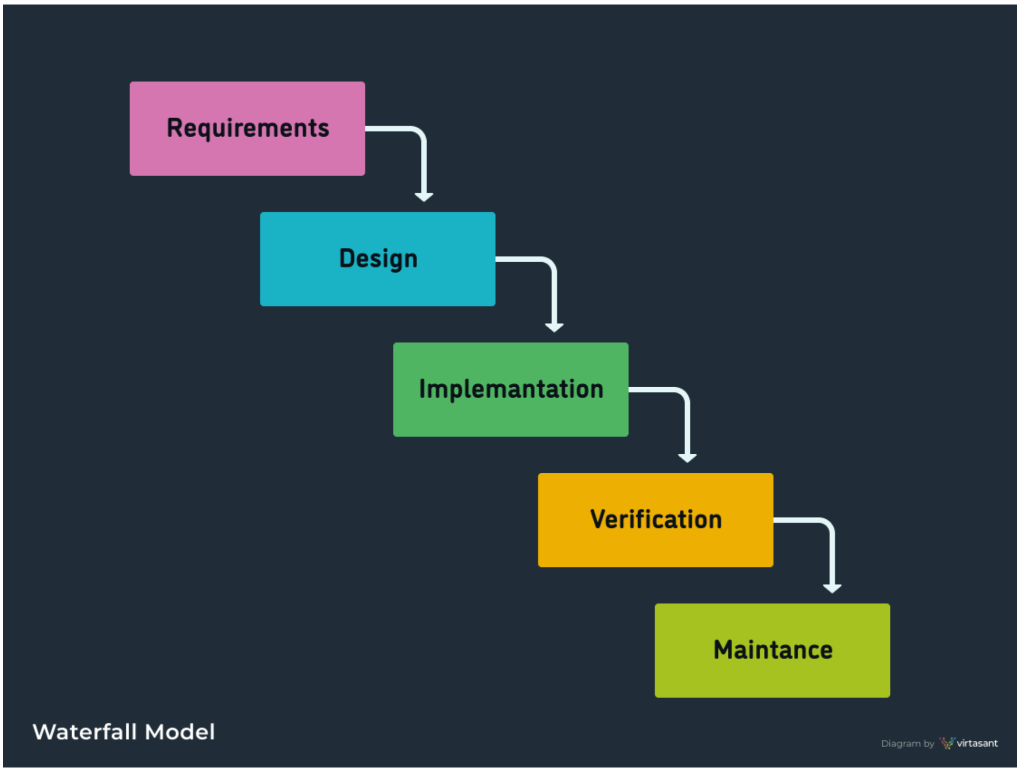


Figure 2.4 Waterfall Methodology (Trunkett, 2020)

2.5.2 Agile Methodology

The Agile methodology is an iterative and incremental model which contains multiple cycles (Software Testing Help, 2022). Each cycle goes through all the SDLC phases and evolves into a better version based on the previous release (Half, 2021). Each cycle will undergo a testing phase to detect small problems in advance to avoid more serious consequences (Half, 2021). Figure 2.5 vividly presents the lifecycle of agile methodology. Therefore, for Agile, earlier failures are better and acceptable. The agile methodology also allows faster and more frequent feedback from users because each cycle will have a finished product that customers can experience directly. The high level of user involvement also ensures that the final product meets or exceeds the customer's expectations. The Agile methodology has several specific frameworks, such as Kanban, Rapid Application Development (RAD), extreme programming (XP), and Scrum. Compared to other SDLC methods, Agile methodology comes with four manifestoes, which are (1) Individuals and interactions over processes and tools, (2) Working software over comprehensive documentation, (3) Customer collaboration over contract negotiation, and (4) Responding to change over following a plan (Beck, 2001). Table 2.2 elaborates on the pros and cons of agile methodology.

Table 2.2 Pros and Cons of Agile Methodology

Pros	**Cons**
Understand customer demand by continuous interaction with them.	Less emphasis on documentation
Quick and regular deployment helps identify the difference between actual product and requirements and helps discover new requirements.	Extra efforts are needed to communicate with the customer. Heavy emphasis on customers sometimes leads in the wrong direction also.
Flexible and adaptive to changes, fewer costs and time required to make the amendment.	Hard to determine how many cycles are needed in the early stages.

Early testing can reduce the risk of the project.	

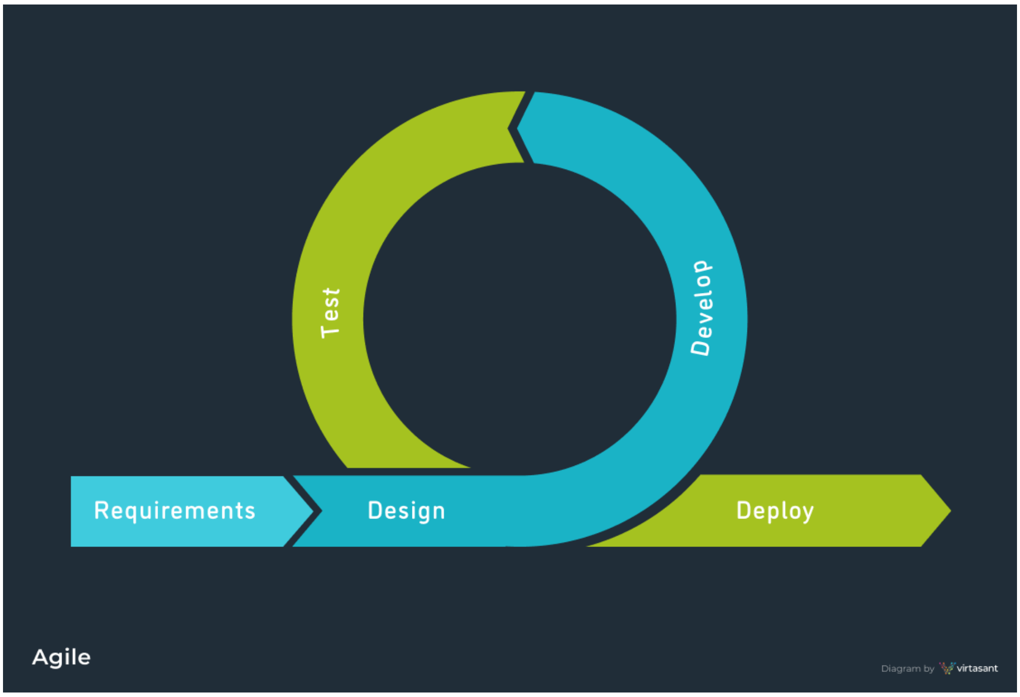


Figure 2.5 Agile Methodology (Trunkett, 2020)

2.5.3 Spiral Methodology

The Spiral methodology is a very flexible SDLC methodology, combining iterative and prototype methods (Software Testing Help, 2022). Four phases in the Spiral methodology: planning, risk analysis, engineering, and evaluation are repeatedly executed in each iteration (Half, 2021). Figure 2.6 illustrates the lifecycle of the spiral methodology. The spiral method is also flexible because new requirements can be added in the next iteration (Software Testing Help, 2022). The spiral method is also risk-driven. In each loop, the spiral methodology focuses on finding potential risks and thinking of the most appropriate way to solve or avoid them. The chosen solution will be tested for effectiveness through a prototype (Software Testing Help, 2022). In the final evaluation phase, the spiral methodology collects user feedback to make continuous improvements. Table 2.3 lists the pros and cons of the spiral methodology.

Table 2.3 Pros and Cons of Spiral Methodology

Pros	Cons
Flexibility to make changes	Only suitable for large projects
Focus on customer feedback	Requires risk management professionals
Lower risk to ensure project success	The uncertain number of cycles can lead to high expenses and longer project periods.

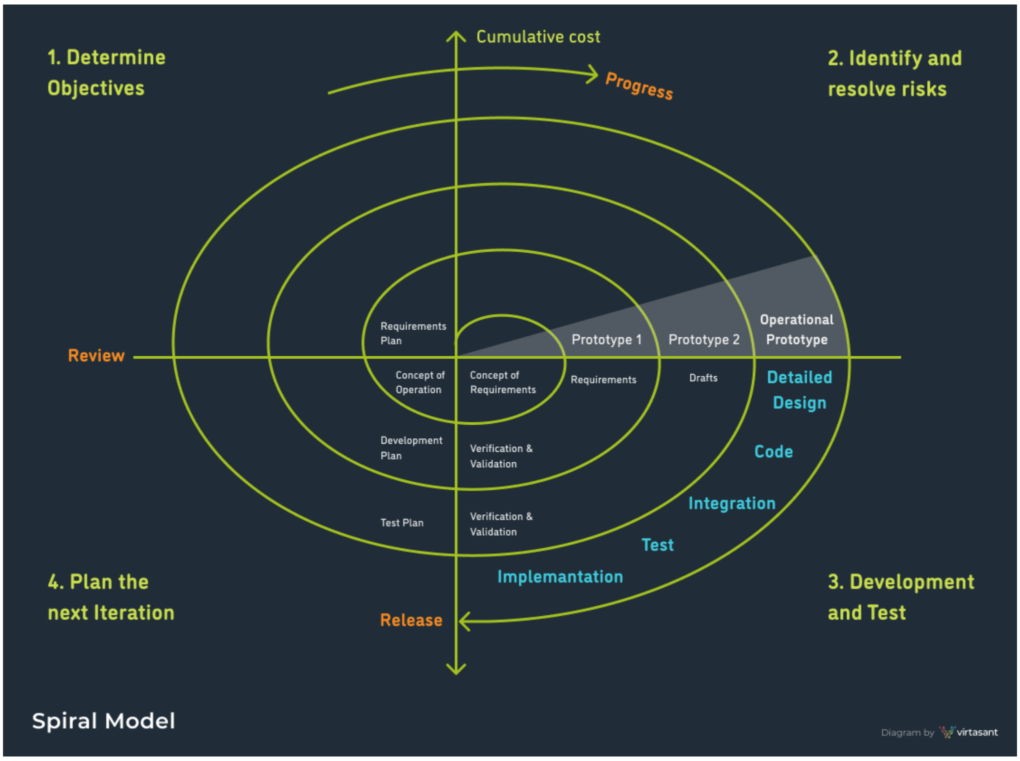


Figure 2.6 Spiral Methodology (Trunkett, 2020)

2.5.4 Iterative Methodology

The iterative methodology does not require knowing all initial requirements but only the primary ones. With these primary requirements, the software can be developed quickly in the first few cycles, and new requirements can be identified later on as knowledge of the project increases. Figure 2.6 illustrates the lifecycle of iterative methodology. Like other modern SDLC methodologies, iterative methodology ensures that defects are detected early in the testing of each iteration. Moreover, customer comments are obtained from the evaluation session in every iteration. This helps to produce newer, more sophisticated, higher quality products in the new iteration. This iteration will continue until the project is completed. Table 2.4 lists the pros and cons of the iterative methodology.

Table 2.4 Pros and Cons of Iterative Methodology

Pros	**Cons**
Problems can be identified early and therefore require less cost to fix.	Failure to identify all requirements at the beginning of the project may result in a faulty initial design
Economical way to find out new requirements	Iteration requires more resources
Flexibility to face changes and high adaptability	

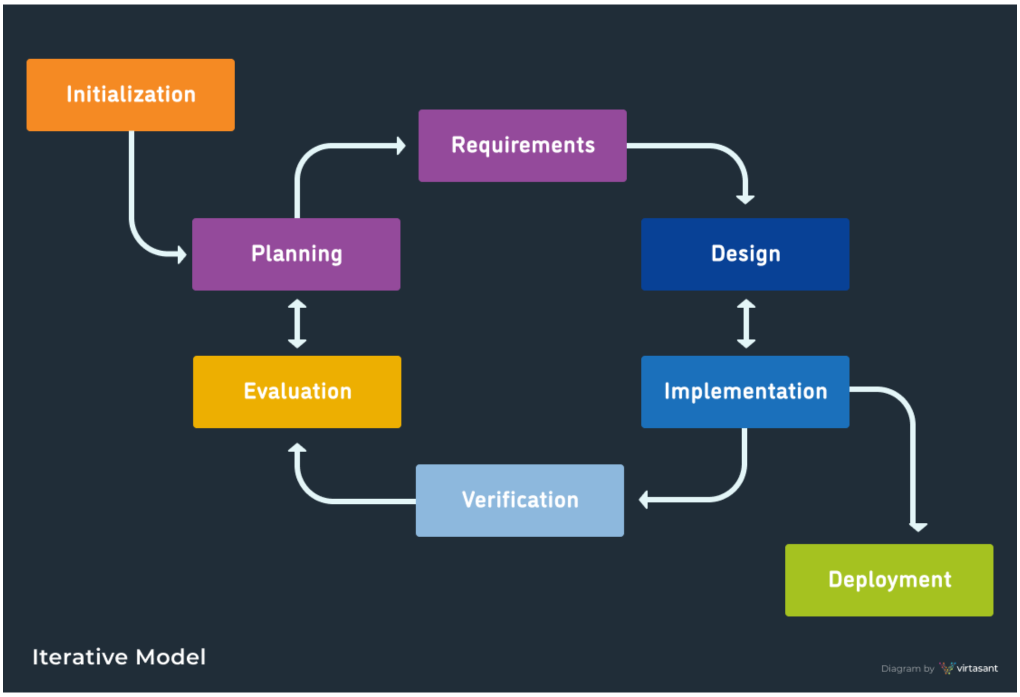


Figure 2.7 Iterative Methodology (Trunkett, 2020)

2.5.5 Comparison among SDLC methodologies

By comparing different SDLC methodologies, the best approach to archive project needs and goals can be determined. Table 2.5 concisely compares four

SDLC methodologies reviewed in this chapter. The features used to compare are the requirement specification gathering stage, the method's flexibility, speed of result delivery, risk factor and frequency of customer feedback.

Table 2.5 Comparison of SDLC methodologies

Methodology / Features	Waterfall	Agile	Spiral	Iterative
Requirements specification	Initial level	Frequently changes	Initial level	Frequently changes
Flexibility	Limited	Excellent	Excellent	Excellent
Result delivery	Delayed	Frequent	Medium	Frequent
Risk factor	High	Low	Low	Low
Customer Feedback	Low	High	High	High

CHAPTER 3

METHODOLOGY AND WORK PLAN

3.1 Introduction

The most appropriate SDLC methodology for this project will be selected from all the methods reviewed in Chapter 2. The reasons for the methodology selection will be discussed, and the detailed implementation flow of the method will be listed in the Work Breakdown Structure. Gantt Chart was provided to state each task's start and end date clearly. Finally, the development tools required for software development will be covered in this chapter.

3.2 System Development Methodology

The iterative methodology was selected after research and review of the different software development lifecycle methodologies were completed in Chapter 2. Figure 3.1 shows the lifecycle of iterative methodology. This is due to the fact that separate teams are responsible for building the hardware of the smart wall outlets. Meanwhile, the software and hardware development are executed concurrently. Therefore, the requirements are difficult to determine at the initial stage. This is because the hardware design may change over time due to other considerations. When the hardware design changes, the software requirements must also be re-determined. The iterative methodology is flexible and adaptive, making changes with less cost.

In addition, smart energy systems and smart wall outlets are strongly related because the software relies on real-time data from the hardware. In contrast, the hardware also depends on the software to make a complete system. Hence, iterative methodology with faster product delivery ensures that the software is developed to meet all the functionality and maximize the use of the information from smart wall outlets. The iterative methodology can evaluate the latest finished mobile application in each iteration, so continuous communication between hardware and software development teams is established. Feedback from the hardware development team guarantees software development is on the right track.

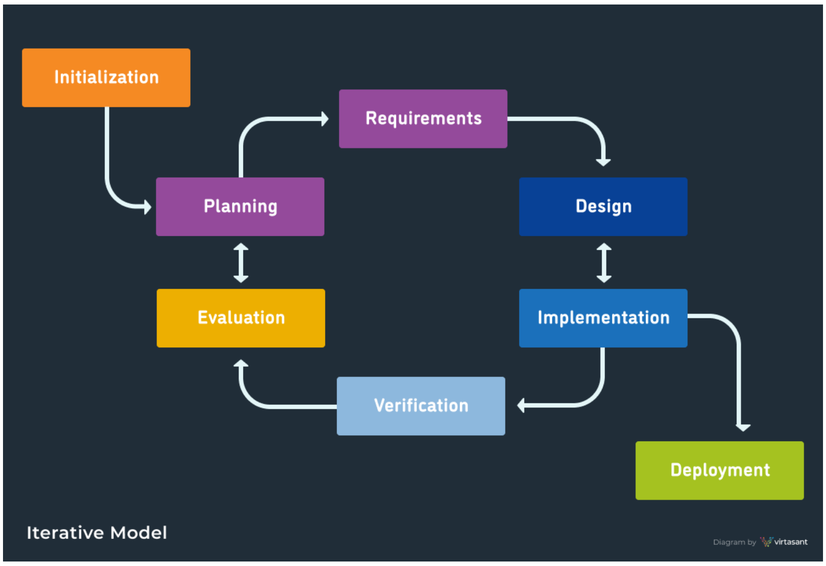


Figure 3.1 Iterative Methodology (Trunkett, 2020)

After the project’s initial planning is completed, the project will enter the iterative phase. The iteration will be performed five times, and each iteration will be approximately 45 to 50 days. Although the same SDLC process is performed in each iteration, different processes will be emphasized.

In the first iteration, more time is spent on obtaining requirements and completing the analysis and design of the project. The use case diagram, the UI design and the system database design will be completed during this iteration. The UI design is then converted to real software to be tested and made available to users for feedback. The UI design will be completed using Figma during this iteration. It is necessary to confirm the selection of colours and fonts for the application and complete the design of all pages. The design of this application must be intuitive and reflect real-world usage scenarios. As such, the page design will showcase the location of the socket through floor plans and use colour to convey the current status of the wall outlet intuitively. Additionally, the pages for controlling the socket switch or sending notifications to users when energy usage exceeds the limit must ensure a good user experience.

Since the feedback from users is received for the first time, there may be multiple points that do not meet the needs and expectations of the users. Therefore, the project will still be able to change the requirements and design in the second iteration. As for the implementation, I will create the main functional module, which controls and monitors the smart wall outlets. These features rely on the database connection to obtain the current data and status of the wall outlet. Therefore, it must be ensured that the hardware can maintain communication with the database. In simple terms, this phase must complete the development of the application’s user interface and enable requests to be sent to the database connected by the hardware when buttons in the application are clicked. This communication between software, database and hardware can fulfil specific requirements, such as obtaining data or modifying the status. Testing will also be done on an ongoing basis to find errors early and reduce risks. Once the main functional module has been established, this phase will continue to complete the remaining modules, such as the floor plan editor, the timing function of the wall outlet, and energy excessive usage alerts. The design of floor plans will be completed using Scalable Vector Graphics (SVG). SVG can draw various shapes within the application, which is appropriate to present the floor plan. The timing function will require building a time-based task scheduler on the backend to trigger specific events at specific times. The alerts for excessive energy usage will need to push notifications to the user's mobile phone.

In the third and fourth iterations, the requirements and design can still be changed based on user feedback from the previous iteration. However, these iterations will focus on completing other modules, such as speech recognition, predicting future energy consumption using machine learning, user management, and property or room management. Speech recognition will be one of the features of this app, but in the initial version, users can only use specific commands to make commands, such as some pre-determined keywords. The speech recognition feature is not designed to intelligently parse and understand the meaning of commands unless some keywords are mentioned. User and property or room management will include user authentication and basic Create, Read, Update and Delete (CRUD) operations. Each completed module will be unit tested to find bugs. Different test cases will be listed and tested one by one to meet as many scenarios as possible.

In the last iteration, all modules will be completed. The test will focus on integration testing, where all the modules can run with each other, and no exceptions are generated. After that, User Acceptance Testing (UAT) will also be carried out to allow users to actually use the application, which can provide more feedback before the official release. The final product will be presented to users. At this point, the application development is complete and will be deployed to a production environment.

3.3 Work Plan

The simplification of complex tasks into smaller tasks for easy execution with Work Breakdown Structure. Figure 3.2 and figure 3.3 portrays the work breakdown structure of the project. The work plan shows how the iterative methodology will be conducted. The specific work content and duration of work in each iteration are planned in Gantt Chart. Figure 3.4 and figure 3.5 display the Gantt Chart of the project.

3.3.1 Work Breakdown Structure

Android-based Smart Energy System

1. Initial Planning
    - 1.1. Project Planning
        - 1.1.1. Background Research
        - 1.1.2. Define Problem Statement
        - 1.1.3. Define Project Objectives
        - 1.1.4. Define Project Solution
        - 1.1.5. Define Project Approach
        - 1.1.6. Define Project Scope
    - 1.2. Literature Review
        - 1.2.1. Review On Similar Application
        - 1.2.2. Review On Speech Recognition
        - 1.2.3. Review On Power Consumption Forecasting
        - 1.2.4. Review On SDLC Methodologies

1.3. Methodology and Workplan

1.3.1. Select SDLC Methodology

1.3.2. Develop Project Workplan

1.3.3. Select Development Tools

2. Iterative Process

2.1. First Iteration

2.1.1. Major Requirements Gathering and Documentation

2.1.2. Analysis & Design

2.1.3. Implementation

2.1.4. Testing

2.1.5. Evaluation

2.2. Second Iteration

2.2.1. New Requirements Gathering and Documentation

2.2.2. Analysis & Design

2.2.3. Implementation

2.2.4. Testing

2.2.5. Evaluation

2.3. Thrid Iteration

2.3.1. New Requirements Gathering and Documentation

2.3.2. Analysis & Design

2.3.3. Implementation

2.3.4. Testing

2.3.5. Evaluation

2.4. Forth Iteration

2.4.1. New Requirements Gathering and Documentation

2.4.2. Analysis & Design

2.4.3. Implementation

2.4.4. Testing

2.4.5. Evaluation

2.5. Fifth Iteration

2.5.1. New Requirements Gathering and Documentation

2.5.2. Analysis & Design

2.5.3. Implementation

2.5.4. Testing

2.5.5. Evaluation

3. Deployment Phase

3.1. System Deployment

4. Report Finalize

4.1. Complete Report Writing

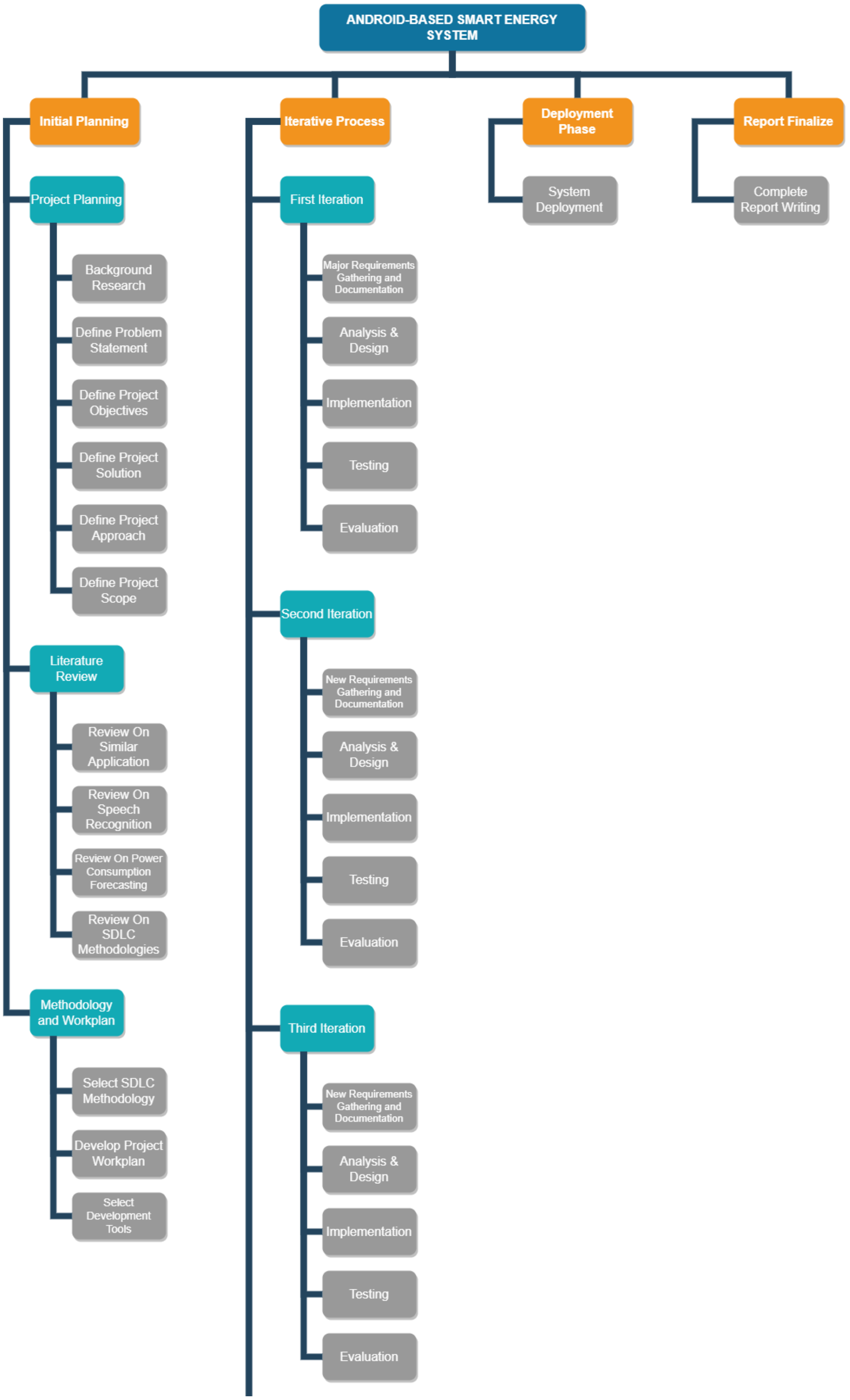


Figure 3.2 Work Breakdown Structure

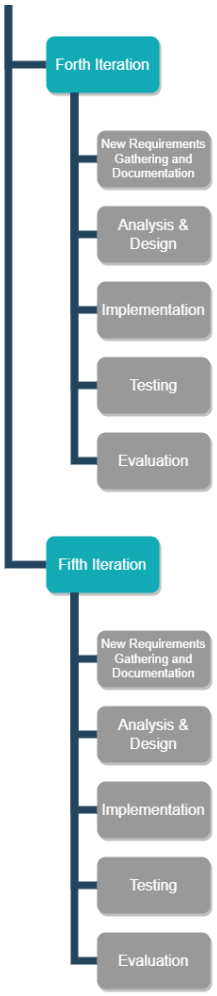
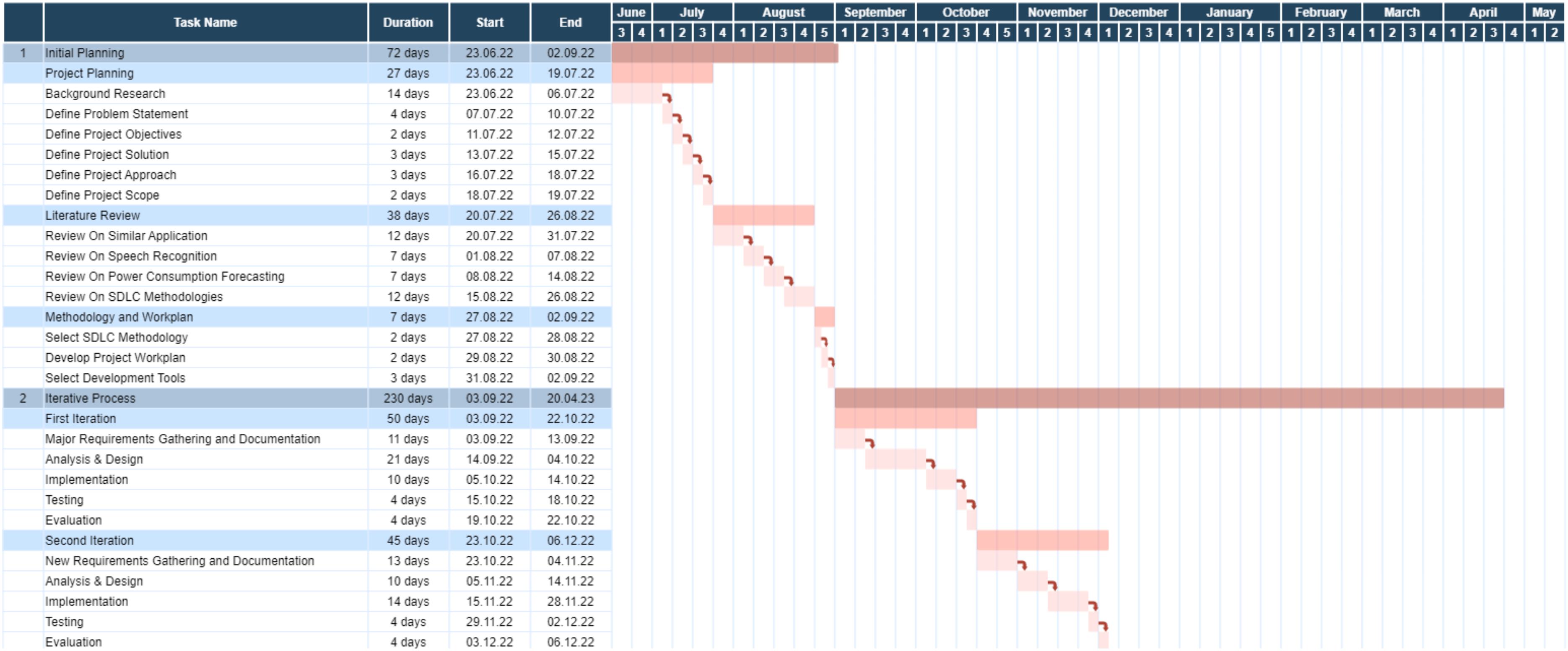


Figure 3.3 Work Breakdown Structure (Continued)

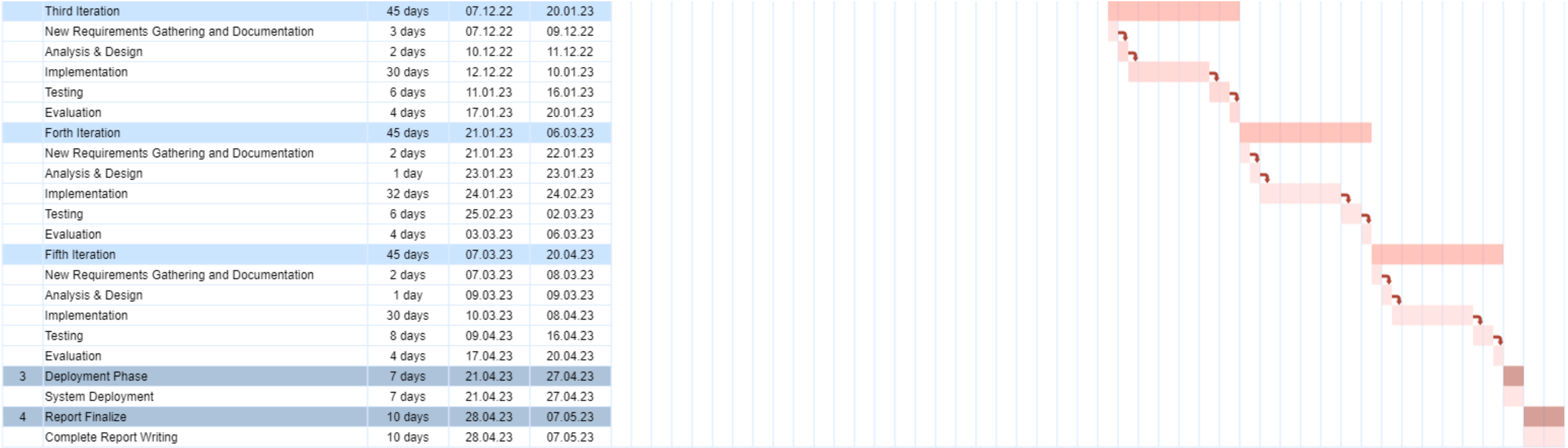
3.3.2 Gantt Chart

Figure 3.4 shows the Gantt chart of the project designed following the iterative methodology.



	Task Name	Duration	Start	End
1	Initial Planning	72 days	23.06.22	02.09.22
	Project Planning	27 days	23.06.22	19.07.22
	Background Research	14 days	23.06.22	06.07.22
	Define Problem Statement	4 days	07.07.22	10.07.22
	Define Project Objectives	2 days	11.07.22	12.07.22
	Define Project Solution	3 days	13.07.22	15.07.22
	Define Project Approach	3 days	16.07.22	18.07.22
	Define Project Scope	2 days	18.07.22	19.07.22
	Literature Review	38 days	20.07.22	26.08.22
	Review On Similar Application	12 days	20.07.22	31.07.22
	Review On Speech Recognition	7 days	01.08.22	07.08.22
	Review On Power Consumption Forecasting	7 days	08.08.22	14.08.22
	Review On SDLC Methodologies	12 days	15.08.22	26.08.22
	Methodology and Workplan	7 days	27.08.22	02.09.22
	Select SDLC Methodology	2 days	27.08.22	28.08.22
	Develop Project Workplan	2 days	29.08.22	30.08.22
	Select Development Tools	3 days	31.08.22	02.09.22
2	Iterative Process	230 days	03.09.22	20.04.23
	First Iteration	50 days	03.09.22	22.10.22
	Major Requirements Gathering and Documentation	11 days	03.09.22	13.09.22
	Analysis & Design	21 days	14.09.22	04.10.22
	Implementation	10 days	05.10.22	14.10.22
	Testing	4 days	15.10.22	18.10.22
	Evaluation	4 days	19.10.22	22.10.22
	Second Iteration	45 days	23.10.22	06.12.22
	New Requirements Gathering and Documentation	13 days	23.10.22	04.11.22
	Analysis & Design	10 days	05.11.22	14.11.22
	Implementation	14 days	15.11.22	28.11.22
	Testing	4 days	29.11.22	02.12.22
	Evaluation	4 days	03.12.22	06.12.22

Figure 3.4 Gantt Chart



	Third Iteration	45 days	07.12.22	20.01.23
	New Requirements Gathering and Documentation	3 days	07.12.22	09.12.22
	Analysis & Design	2 days	10.12.22	11.12.22
	Implementation	30 days	12.12.22	10.01.23
	Testing	6 days	11.01.23	16.01.23
	Evaluation	4 days	17.01.23	20.01.23
	Forth Iteration	45 days	21.01.23	06.03.23
	New Requirements Gathering and Documentation	2 days	21.01.23	22.01.23
	Analysis & Design	1 day	23.01.23	23.01.23
	Implementation	32 days	24.01.23	24.02.23
	Testing	6 days	25.02.23	02.03.23
	Evaluation	4 days	03.03.23	06.03.23
	Fifth Iteration	45 days	07.03.23	20.04.23
	New Requirements Gathering and Documentation	2 days	07.03.23	08.03.23
	Analysis & Design	1 day	09.03.23	09.03.23
	Implementation	30 days	10.03.23	08.04.23
	Testing	8 days	09.04.23	16.04.23
	Evaluation	4 days	17.04.23	20.04.23
3	Deployment Phase	7 days	21.04.23	27.04.23
	System Deployment	7 days	21.04.23	27.04.23
4	Report Finalize	10 days	28.04.23	07.05.23
	Complete Report Writing	10 days	28.04.23	07.05.23

Figure 3.5 Gantt Chart (Continued)

3.4 Development Tools

Development tools can be divided into programming languages, frameworks, integrated development environments (IDE), version control systems, databases, and third-party libraries. This project uses JavaScript as its programming language, React Native as its mobile application development framework, and MySQL and Firebase as its database. This project will be developed using Visual Studio Code and Android Studio IDEs. Git will be used for version control. Finally, a third-party library, Picovoice, will also be utilized.

3.4.1 JavaScript

JavaScript is a client-side scripting language that was first intended to make web pages interactive. Nowadays, JavaScript is also utilized in back-end development because of various advantages such as high speed and less overhead. JavaScript is also a popular programming language because of its relative simplicity, which has led to a large JavaScript community.

3.4.2 React Native

React Native is a JavaScript-based mobile app development framework. React Native provides developers with a faster and more economical way to develop a mobile application. In addition, React Native has a strong community that provides a wealth of information, resources, and assistance. React Native’s complete and comprehensive documentation ensures developers can understand it more deeply.

3.4.3 Firebase

Firebase’s real-time database provides real-time storage of data. Therefore, it is ideal for storing data from smart wall outlets that are getting new data every moment. Firebase is a NoSQL database, so it has a greater advantage than traditional SQL databases when retrieving a lot of time-related data.

3.4.4 Visual Studio Code

The project source code will be written using Visual Studio Code as the primary editor. Visual Studio Code is small and does not consume a lot of memory.

Many plugins can increase the usability of Visual Studio Code, for example, by providing hints for syntax errors, code beautification, and custom shortcuts.

3.4.5 Android Studio

Android Studio is the authorized IDEs built specifically for Android development. Android Studio allows developers to easily manage the SDKs and launch an emulator to preview the application’s development and testing on different Android API levels.

3.4.6 Git

The version control system, Git, can record every change. If an error occurs, the developer can revert to the previous version. Git also ensures that every piece of code is traceable. Syncing code to GitHub for backup also reduces the chance of losing code accidentally.

3.4.7 Picovoice

Picovoice speech recognition platform is used to achieve voice command features. Picovoice emphasizes accuracy, privacy, security, and zero latency. Picovoice provides a keyword wake-up function, which is very important for controlling smart wall outlets. In addition, Picovoice also supports speech recognition in English, German, French, and Spanish. Picovoice is one of the only enterprise-grade speech recognition libraries with a free tier.

3.5 Conclusion

This project will be developed according to the SDLC methodology chosen in this section and the development tools mentioned. The timeline will be strictly adhered to achieve the project objectives.

CHAPTER 4

PROJECT SPECIFICATION

4.1 Introduction

This chapter includes the detailed requirement specification, which defines the functional and non-functional requirements of the application, and the use case analysis, which outlines the interactions and processes of the application from a user's perspective. This information serves as the foundation for the design and development of the application.

4.2 Requirements Specification

Requirement specification defines what the application should do and how it should behave. These requirements were obtained through the review of similar applications and based on the recommendations of the project supervisor.

4.2.1 Functional Requirements

Module 1: Profile

- The application shall allow users to register a new account.
- The application shall allow users to log in using email and password.
- The application shall send a verification email upon successful registration.
- The application shall allow users to edit their usernames.
- The application shall allow users to change their password.

Module 2: Property Management

- The application shall allow users to add different types of property to manage.
- The application shall allow users to tariff type of property.
- The application shall allow users to add multiple rooms to a property.
- The application shall allow users to scan a QR code to add a wall outlet to a room.
- The application shall display the floor plan of a property or room and indicate the wall outlet status in the floor plan by colour.

- The application shall allow users to edit the property and room floor plan.
- The application shall allow users to edit the position of a wall outlet in a room.

Module 3: Control

- The application shall allow users to remote control the smart wall outlet.
- The application shall display the real-time power and current if the wall outlet is switched on.
- The application shall allow users to schedule the smart wall outlet to start and close according to the time settled automatically.
- The application shall allow users to schedule the smart wall outlet to close automatically after a pre-set time limit.

Module 4: Monitoring

- The application shall allow users to view the total energy consumption overall, a specific property, a specific room or a specific wall outlet.
- The application shall allow users to view the total energy consumption in price with unit Ringgit Malaysia (RM).
- The application shall allow users to view the total energy consumption in unit watts per hour, Wh.
- The application shall allow users to view the total energy consumption within a week.
- The application shall allow users to view the total energy consumption within a month.
- The application shall allow users to view the latest events of a wall outlet.

Module 5: Speech Recognition

- The application shall allow the user to use voice commands to control the smart wall outlet's status (ON/OFF).

Module 6: Notification

- The application shall notify the user when energy consumption exceeds the alert level.
- The application shall allow the user to view the list of notifications.

4.2.2 Non-Functional Requirements

- The application shall have an attractive user interface.

- The application shall be built in JavaScript React Native Framework.
- The application shall have the ability to monitor and control smart wall outlets whenever the internet is connected.
- The application must be accessible at all times, 24 hours a day.

4.3 Use Case Diagram

A use case diagram was utilized to analyze and illustrate the users' interactions with the system, providing a better understanding of the requirements. Each user's interaction was depicted as a use case in the diagram. This system consists of one actor and thirteen use cases. Figure 4.1 shows the use case diagram of this smart energy mobile system.

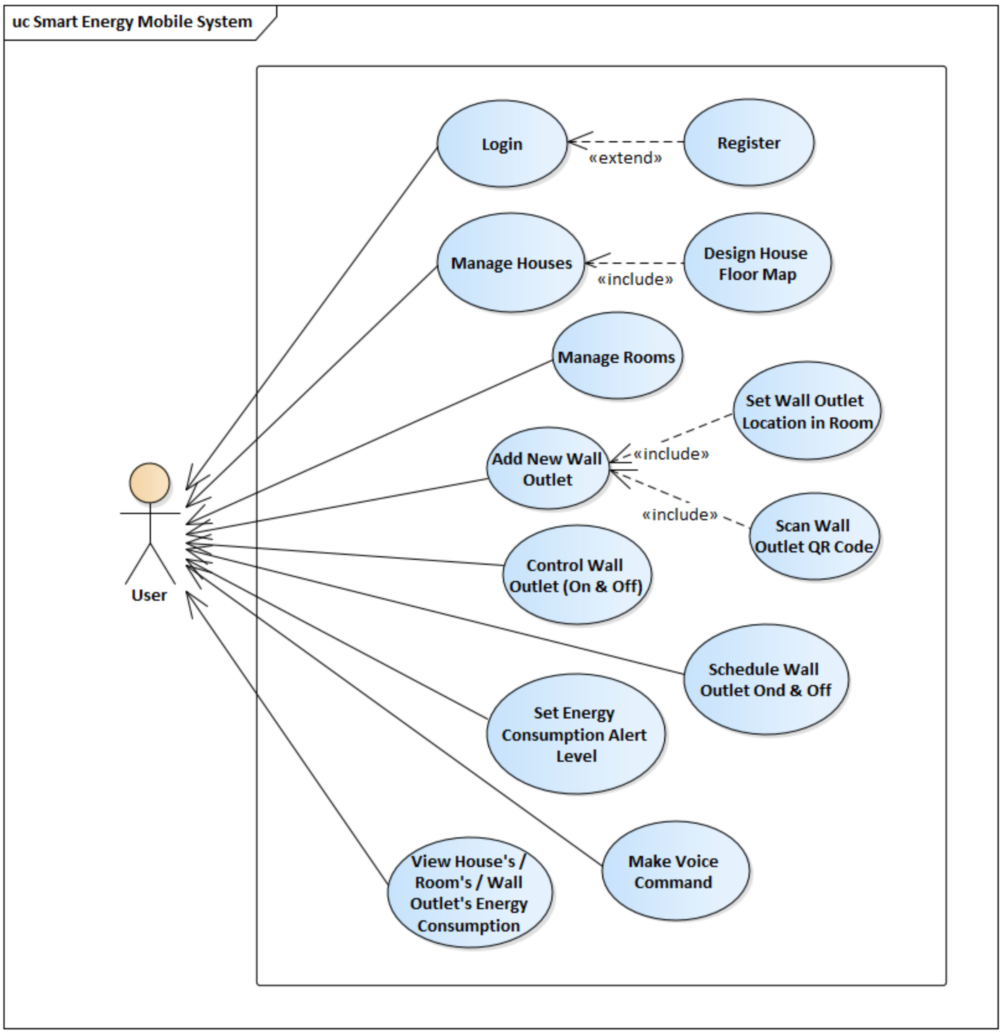

uc Smart Energy Mobile System
Login
«extend»
Register
Manage Houses
«include»
Design House Floor Map
Manage Rooms
Set Wall Outlet Location in Room
Add New Wall Outlet
«include»
«include»
Scan Wall Outlet QR Code
User
Control Wall Outlet (On & Off)
Schedule Wall Outlet Ond & Off
Set Energy Consumption Alert Level
View House's / Room's / Wall Outlet's Energy Consumption
Make Voice Command

Figure 4.1 Use Case Diagram of Smart Energy Mobile System

4.4 Use Case Description

4.4.1 Login

Use Case Name: Login	
ID: 1	**Importance Level:** High
Primary Actor: User	**Use Case Type:** Detail, Essential
Stakeholders and Interests: User - wants to access the application	
Brief Description: The user inserts login credentials to log in to the application.	
Trigger: The user wants to access the application.	
Relationships: Association: User Include: - Extend: Register Generalization: -	
Normal Flow of Events: 1. The user launchs the application. 2. The system cannot find any existing credentials. 3. The system redirects to the login page. 4. The user inserts login information. 5. The system validates the login information. If validation fails, perform exceptional flow 5.1 6. Upon successful validation, the system redirects the user to the application's home page.	
Sub-flows: -	

Alternate/Exceptional Flows: 5.1 Invalid login information feedback 1. System prompts the user that the login information entered is invalid. 2. System prompts the user to re-enter login information.

4.4.2 Register

Use Case Name: Register	
ID: 2	**Importance Level:** High
Primary Actor: User	**Use Case Type:** Detail, Essential
Stakeholders and Interests: User - wants to create a new account for the application.	
Brief Description: The user inserts the information required to create a new application account.	
Trigger: The user wants to register a new account for the application.	
Relationships: Association: User Include: - Extend: - Generalization: -	
Normal Flow of Events: 1. The user inserts the information required for account registration. 2. The user taps on the 'register' button. Perform sub-flow 2.1.	
Sub-flows: 2.1 Validate registration information	

1. System validates that the information entered matches specific formats such as date, phone number and email. 2. System checks the equality of password and password confirmation. 3. If any information is invalid, perform exceptional flow 3.1
Alternate/Exceptional Flows: 3.1 Invalid registration information feedback 1. System prompts the user which information entered was not valid. 2. System prompts the user to re-enter the invalid information.

4.4.3 Add Houses / Factories

Use Case Name: Add Houses / Factories	
ID: 3	**Importance Level:** High
Primary Actor: User	**Use Case Type:** Detail, Essential
Stakeholders and Interests: User - want to add houses or factories in the application	
Brief Description: Users can add houses or factories to manage different properties individually.	
Trigger: The user wants to add houses or factories to the application.	
Relationships: Association: User Include: - Extend: Design Floor Map Generalization: -	
Normal Flow of Events: 1. The user taps on the ‘add’ button in the houses/factories section. 2. The user selects the type of estate and number of rooms and names it. 3. The user taps on the ‘save button to store this property.	

4. The system will display a message confirming that the house or factory has been successfully saved.
Sub-flows: -
Alternate/Exceptional Flows: 5.1 Invalid login information feedback 1. System prompts the user that the login information entered is invalid. 2. System prompts the user to re-enter login information.

4.4.4 Design House Floor Map

Use Case Name: Design House Floor Map	
ID: 4	**Importance Level:** High
Primary Actor: User	**Use Case Type:** Detail, Essential
Stakeholders and Interests: User - wants to design the floor map of the house/factory.	
Brief Description: The user can change room shape, size and position of rooms in the house/factory.	
Trigger: The user wants to design the floor map of the house/factory.	
Relationships: Association: User Include: - Extend: - Generalization: -	
Normal Flow of Events: 1. The user drags a different shape room to the floor map. 2. The user adjusts rooms to the desired size. 3. The user moves rooms to desired position.	

4. The user taps on the ‘Save’ button to store the floor map. 5. The system displays a ‘Floor map saved’ message.
Sub-flows: -
Alternate/Exceptional Flows: -

4.4.5 Add New Wall Outlet

Use Case Name: Add New Wall Outlet	
ID: 5	**Importance Level:** High
Primary Actor: User	**Use Case Type:** Detail, Essential
Stakeholders and Interests: User – wants to add a new wall outlet in a room.	
Brief Description: The user adds a new wall outlet in a room.	
Trigger: The user wants to add a new wall outlet in a room.	
Relationships: Association: User Include: Scan wall outlet QR code, Set wall outlet location in the room Extend: - Generalization: -	
Normal Flow of Events: 1. The user taps on the ‘add’ button to add a wall outlet in a room. 2. The user inserts the wall outlet’s alias name and sets the energy consumption alert level. 3. The user taps on the ‘save’ button to store. 4. The system displays a ‘Wall Outlet saved’ message.	
Sub-flows: -	

Alternate/Exceptional Flows: -

4.4.6 Set Wall Outlet Location in Room

Use Case Name: Set Wall Outlet Location in Room	
ID: 6	**Importance Level:** High
Primary Actor: User	**Use Case Type:** Detail, Essential
Stakeholders and Interests: User - wants to place the wall outlet in a specific location in the room.	
Brief Description: The user can move the wall outlet around the room's floor map to place it in a desired location.	
Trigger: The user wants to add a new wall outlet in the room.	
Relationships: Association: User Include: - Extend: - Generalization: -	
Normal Flow of Events: 1. The user moves the wall outlet on the room floor plan to the desired location. 2. The user taps on the 'ok' button to confirm the wall outlet location. 3. The system exits floor plan editing mode.	
Sub-flows: -	
Alternate/Exceptional Flows: -	

4.4.7 Scan Wall Outlet QR Code

Use Case Name: Scan Wall Outlet QR Code

<table>
<tr><td><b>ID:</b> 7</td><td><b>Importance Level:</b> High</td></tr>
<tr><td><b>Primary Actor:</b> User</td><td><b>Use Case Type:</b> Detail, Essential</td></tr>
<tr><td colspan="2"><b>Stakeholders and Interests:</b>
User - wants to get access to the wall outlet information.</td></tr>
<tr><td colspan="2"><b>Brief Description:</b> The user scans the QR code on the wall outlet hardware to get access to the wall outlet information.</td></tr>
<tr><td colspan="2"><b>Trigger:</b> The user wants to add a new wall outlet in the room.</td></tr>
<tr><td colspan="2"><b>Relationships:</b>
Association: User
Include: -
Extend: -
Generalization: -</td></tr>
<tr><td colspan="2"><b>Normal Flow of Events:</b>
1. The system requests the camera permission from the user.
2. The user uses the camera to scan the QR code on the wall outlet hardware.
3. The system verifies the QR code.
4. The system retrieves the wall outlet information.</td></tr>
<tr><td colspan="2"><b>Sub-flows:</b> -</td></tr>
<tr><td colspan="2"><b>Alternate/Exceptional Flows:</b> -</td></tr>
</table>

4.4.8 Control Wall Outlet (On & Off)

<table>
<tr><td colspan="2"><b>Use Case Name:</b> Control Wall Outlet (On &amp; Off)</td></tr>
<tr><td><b>ID:</b> 8</td><td><b>Importance Level:</b> High</td></tr>
<tr><td><b>Primary Actor:</b> User</td><td><b>Use Case Type:</b> Detail, Essential</td></tr>
</table>

Stakeholders and Interests: User - wants to change the status of a wall outlet.
Brief Description: The user can turn on and off the wall outlet.
Trigger: The user wants to change the status of a wall outlet.
Relationships: Association: User Include: - Extend: - Generalization: -
Normal Flow of Events: 1. The user selects the wall outlet they want to control. 2. The user swipes on the slider controller to change the status of the wall outlet.
Sub-flows: -
Alternate/Exceptional Flows: -

4.4.9 Schedule Wall Outlet On & Off

Use Case Name: Schedule Wall Outlet On & Off	
ID: 10	**Importance Level:** High
Primary Actor: User	**Use Case Type:** Detail, Essential
Stakeholders and Interests: User - wants to schedule a time for the wall outlet to turn on and off automatically.	
Brief Description: The user can insert a time for the wall outlet to turn on and a time for the wall outlet to turn off.	

Trigger: The user wants to schedule a time for the wall outlet to turn on and off automatically.
Relationships: Association: User Include: - Extend: - Generalization: -
Normal Flow of Events: 1. The user selects the wall outlet they want to schedule. 2. The user inserts a time for automatic turn on the wall outlet. 3. The user inserts a time for automatic turn on the wall outlet. 4. The user selects the schedule trigger daily or on a specific day or date. 5. The user taps on the 'ok' button to confirm the schedule. 6. The system stores the schedule.
Sub-flows: -
Alternate/Exceptional Flows: -

4.4.10 Set Energy Consumption Alert Level

Use Case Name: Set Energy Consumption Alert Level	
ID: 10	**Importance Level:** High
Primary Actor: User	**Use Case Type:** Detail, Essential
Stakeholders and Interests: User - wants to set energy consumption alert level for a wall outlet, room or house.	
Brief Description: The user sets an energy consumption alert level to receive a notification when energy consumption is about to exceed the level.	

Trigger: The user wants to set an energy consumption alert level for a wall outlet, room or house.
Relationships: Association: User Include: - Extend: - Generalization: -
Normal Flow of Events: 1. The user taps on the ‘edit’ button to edit alert value. 2. The user inserts the value for warning and forced stop. 3. The user taps on the ‘save’ button to store. 4. The system displays a ‘Alert value saved’ message.
Sub-flows: -
Alternate/Exceptional Flows: -

4.4.11 Make Voice Command

Use Case Name: Make Voice Command	
ID: 11	**Importance Level:** High
Primary Actor: User	**Use Case Type:** Detail, Essential
Stakeholders and Interests: User - wants to use voice command to control the wall outlet.	
Brief Description: The user can call a wall outlet’s alias name to change its status.	
Trigger: The user wants to use voice command to control the wall outlet.	

Relationships: Association: User Include: - Extend: - Generalization: -
Normal Flow of Events: 1. The user speaks to the device to wake the voice assistant. 2. The user says the wall outlet’s alias name and the status want to change. 3. The system provides feedback when the wall outlet’s status is updated.
Sub-flows: -
Alternate/Exceptional Flows: -

4.4.12 View House’s / Room’s / Wall Outlet’s Energy Consumption

Use Case Name: View House’s / Room’s / Wall Outlet’s Energy Consumption	
ID: 12	**Importance Level:** High
Primary Actor: User	**Use Case Type:** Detail, Essential
Stakeholders and Interests: User - wants to view the house’s / room’s / wall outlet’s energy consumption.	
Brief Description: The user can view the house’s / room’s / wall outlet’s energy consumption graphically.	
Trigger: The user wants to view the energy consumption of the house’s / room’s / wall outlet.	

Relationships:

Association: User

Include: -

Extend: -

Generalization: -

Normal Flow of Events:

1. The user taps on the ‘data’ button of a house/room/wall outlet.
2. The system redirects to the energy consumption page.
3. The user views the chart of energy consumption.
4. The user can change the time interval of energy consumption.

Sub-flows: -

Alternate/Exceptional Flows: -

CHAPTER 5

SYSTEM DESIGN

5.1 Introduction

In this chapter, the system architecture, database, and user interface of the project are presented and discussed. The system architecture design focuses on explaining the structure and behaviour of the system, which will include the use of frameworks and their relationship. In addition, the system database design is illustrated with the entity relationship diagram. Lastly, the system's user interface was designed to be efficient with high usability.

5.2 System Architecture Design

The mobile application utilized a client-server architecture that consisted of three tiers. Three-tier client-server architecture is a software architecture pattern in which the user interface (presentation), business logic (application server), and database management (data) are separated into three distinct components. Figure 5.1 shows the diagram of the system architecture of the mobile app. The reason for selecting this architecture is because it allows for greater flexibility and scalability, as changes made in one tier do not affect the other tiers, and each tier can be optimized for its specific role. Additionally, it can improve security by isolating the critical database from direct access by the user and application server. For example, manipulating the wall outlets must go through multiple validation tiers before completion.

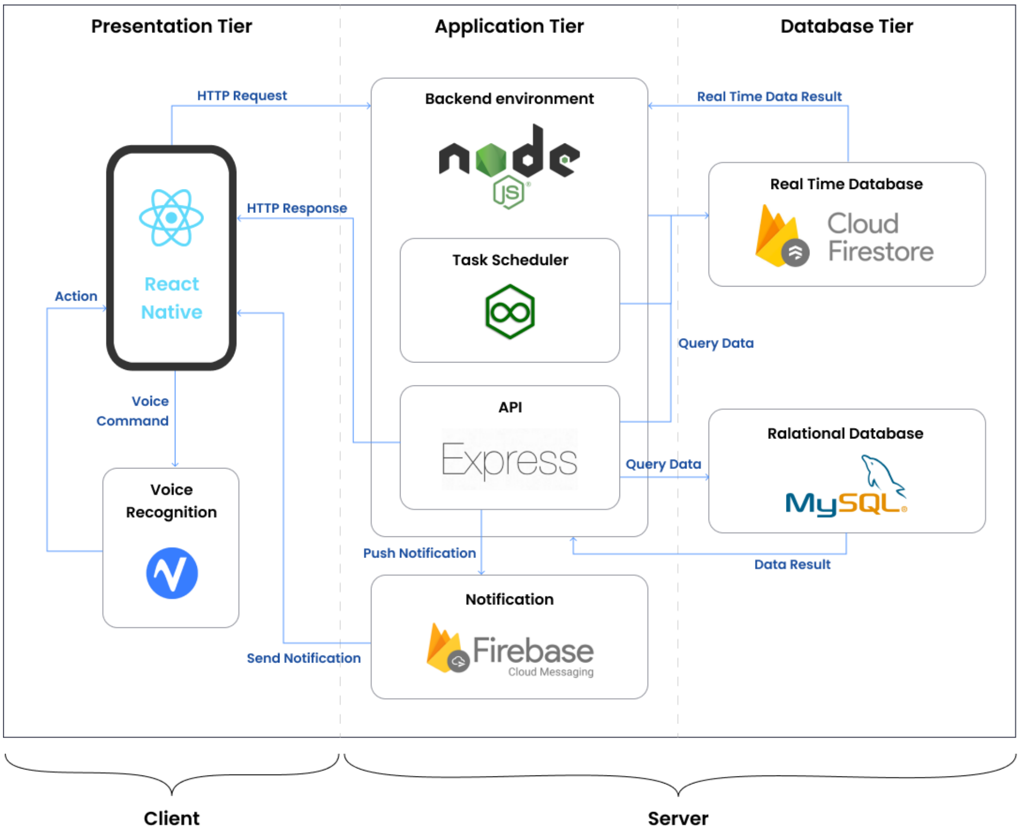


Figure 5.1 System Architecture Design

The presentation tier refers to the frontend layer of the mobile application, will be created using React Native framework. It is responsible for presenting data to the user and receiving user input. The presentation tier will send HTTP requests to specified servers to retrieve the required data. Voice recognition is a sub-system within the presentation tier. The user's voice command will be identified with the support of a third-party library, PicoVoice. As a result, actions will be taken to complete tasks as ordered.

The application tier is the middle layer, responsible for executing the business logic, managing data access and processing, and performing tasks such as triggering push notifications and scheduling some time-based actions. Node.js are used to create a backend server environment in the application tier. This backend server environment will continuously operate to receive HTTP requests at any time. The HTTP requests will be received and processed through the application programming interface written in Express.js. After that, the API will query data in the database and reply with an HTTP response.

Besides that, once the backend server environment is initialized, it will direct retrieve real-time data from the database tier. The real-time data retrieved contain information on wall outlets, such as real-time energy usage. A wall outlet's real-time energy usage will be monitored at all times. Once it exceeds some of the alert levels settled by the user, the user's mobile device will receive a push notification. Since the user can set the schedule of switching wall outlets' status, the schedule will be stored as tasks to be executed in the task scheduler. The task scheduler was implemented with node-cron, a task scheduler library in pure JavaScript for node.js. The task scheduler will trigger functions to interact with the database tier to update the wall outlet state when time conditions meet.

Lastly, the database tier is responsible for storing and retrieving data. It typically includes a database management system (DBMS) that handles data storage, retrieval, and manipulation. Two types of databases are used to store different kinds of data. The first database system is MySQL. MySQL is a relational database management system suitable for storing data that strictly adheres to a relational schema. For example, users, properties, rooms, and wall outlets have affirmed structure and consistent relationships. The first database system is a real-time database by Firebase. The reason for having Firebase is that the data on the energy usage of wall outlets are growing extremely fast. Firebase is a NoSQL database that is more appropriate for storing large datasets.

5.3 Database Design

Database design process such as defining data elements, data relationship and normalising data was conducted before the actual database was built. This paper was only involved in the design of the relational database. Therefore, the design of the real-time database for recording wall outlet data was not included. In this section, Entity Relationship Diagram (ERD) and Data Dictionary were created to illustrate an overview of the structure of the relational database.

5.3.1 Entity Relationship Diagram

An entity Relationship Diagram (ERD) is a valuable tool in database design and development to ensure that the database design meets the requirements. Figure

X shows the ERD of the Smart Energy Monitoring Mobile Application. This relational database contains 9 tables, namely users, properties, rooms, wall outlets, wall outlet schedules, notifications, task schedules, property floor plans and room floor plans.

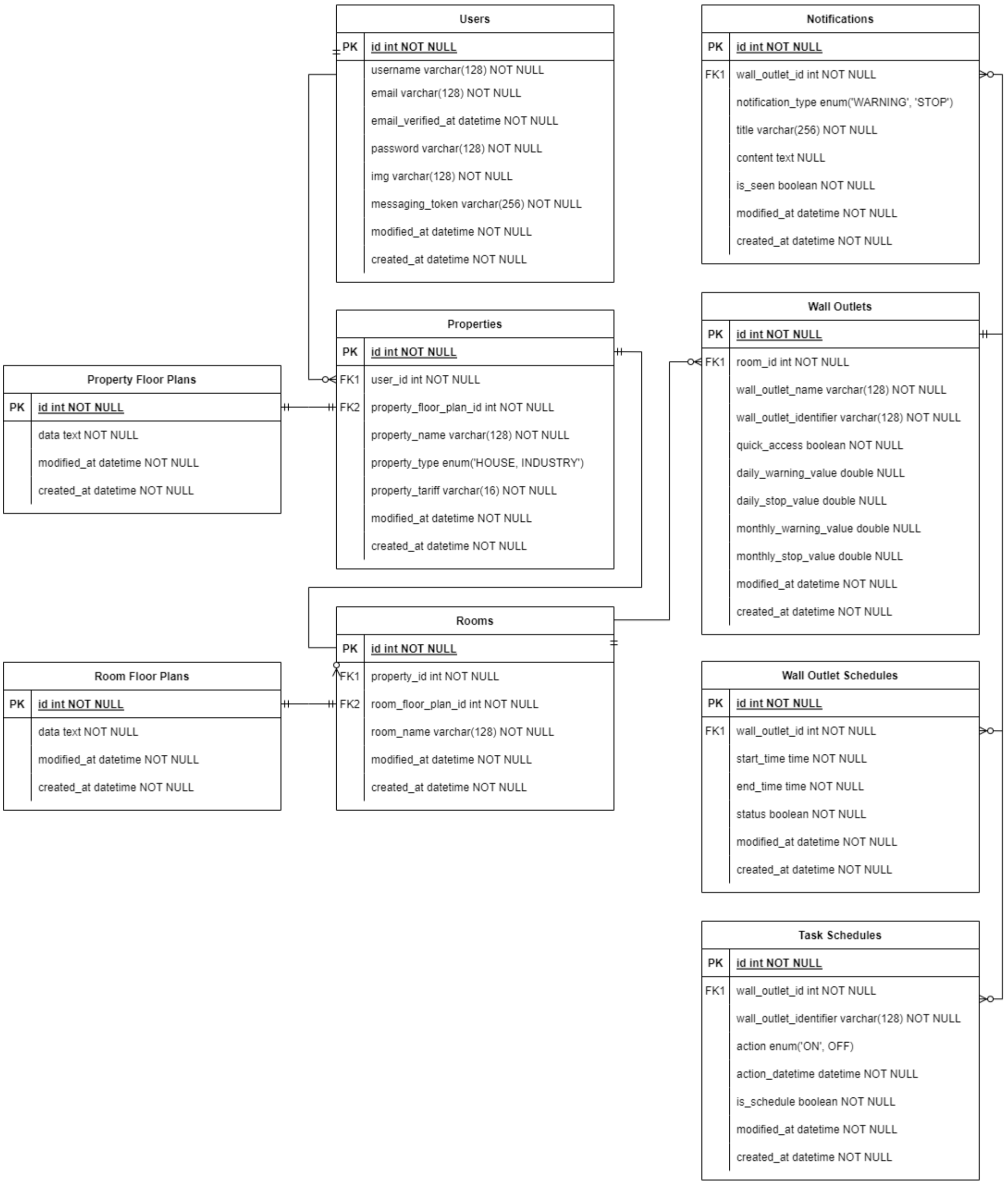

Users
PK id int NOT NULL
username varchar(128) NOT NULL
email varchar(128) NOT NULL
email_verified_at datetime NOT NULL
password varchar(128) NOT NULL
img varchar(128) NOT NULL
messaging_token varchar(256) NOT NULL
modified_at datetime NOT NULL
created_at datetime NOT NULL
Notifications
PK id int NOT NULL
FK1 wall_outlet_id int NOT NULL
notification_type enum('WARNING', 'STOP')
title varchar(256) NOT NULL
content text NULL
is_seen boolean NOT NULL
modified_at datetime NOT NULL
created_at datetime NOT NULL
Property Floor Plans
PK id int NOT NULL
data text NOT NULL
modified_at datetime NOT NULL
created_at datetime NOT NULL
Properties
PK id int NOT NULL
FK1 user_id int NOT NULL
FK2 property_floor_plan_id int NOT NULL
property_name varchar(128) NOT NULL
property_type enum('HOUSE, INDUSTRY')
property_tariff varchar(16) NOT NULL
modified_at datetime NOT NULL
created_at datetime NOT NULL
Wall Outlets
PK id int NOT NULL
FK1 room_id int NOT NULL
wall_outlet_name varchar(128) NOT NULL
wall_outlet_identifier varchar(128) NOT NULL
quick_access boolean NOT NULL
daily_warning_value double NULL
daily_stop_value double NULL
monthly_warning_value double NULL
monthly_stop_value double NULL
modified_at datetime NOT NULL
created_at datetime NOT NULL
Room Floor Plans
PK id int NOT NULL
data text NOT NULL
modified_at datetime NOT NULL
created_at datetime NOT NULL
Rooms
PK id int NOT NULL
FK1 property_id int NOT NULL
FK2 room_floor_plan_id int NOT NULL
room_name varchar(128) NOT NULL
modified_at datetime NOT NULL
created_at datetime NOT NULL
Wall Outlet Schedules
PK id int NOT NULL
FK1 wall_outlet_id int NOT NULL
start_time time NOT NULL
end_time time NOT NULL
status boolean NOT NULL
modified_at datetime NOT NULL
created_at datetime NOT NULL
Task Schedules
PK id int NOT NULL
FK1 wall_outlet_id int NOT NULL
wall_outlet_identifier varchar(128) NOT NULL
action enum('ON', OFF)
action_datetime datetime NOT NULL
is_schedule boolean NOT NULL
modified_at datetime NOT NULL
created_at datetime NOT NULL

Figure 5.2 Entity Relationship Diagram

5.3.2 Data Dictionary

Table 5.1 Description of Database Tables

Table Name	**Description**
Users	Contains the user’s personal information and authentication info.
Properties	Contains property information. One user account can contain multiple properties.
Rooms	Contains room information. The property will be divided into rooms; therefore, one can contain multiple rooms.
Sockets	Contains socket information which will use the information to get specific real-time data from the Firebase database.
Socket schedules	Contains socket switching on and off schedule settled by the user.
Notifications	Contains notification record.
Task Schedules	Contains a record of scheduled tasks by node-cron to act as a backup log to prevent scheduled task loss due to unexpected failure in the application tier.
Property Floor Plans	Contains property floor plan information.
Room Floor Plans	Contains room floor plan information.

Table 5.2 Users Entity Data Dictionary

Fields	**Field’s Description**	**Type**	**Example**
id [PK]	Unique identification number for a user	int	1
username	User's salutation will be displayed in the application	varchar	Sheng
email	Email address of a user	varchar	goh@gmail.com
email_verified_at	Date and time when the email was verified to prevent misuse of someone else's email	datetime	2022-12-25 10:30:00
password	Password for authentication use.	varchar	A2sd##dw
img	User’s profile picture.	varchar	abc.png
messaging_token	Token as device identity to allow push notification	varchar	dBpRIanC
modified_at	Date and time when the last modification was made to this data	datetime	2022-12-25 10:30:00
created_at	Time and date of data creation.	datetime	2022-12-25 10:30:00

Table 5.3 Properties Entity Data Dictionary

Fields	**Field's Description**	**Type**	**Example**
id [PK]	Unique identification number for a property	int	1
user_id [FK]	Unique identifier of the user that has this property	int	1
property_floor_plan_id [FK]	Unique identifier of a property floor plan that stores the floor plan data of this property	int	1
property_name	Property name to display in the application	varchar	My House
property_type	Type of the property either residential or industrial	enum	HOUSE
property_tariff	Property tariff to select to calculate electricity fee based on different charges.	varchar	E1
modified_at	Date and time when the last modification was made to this data	datetime	2022-12-25 10:30:00
created_at	Time and date of data creation.	datetime	2022-12-25 10:30:00

Table 5.4 Rooms Entity Data Dictionary

Fields	**Field’s Description**	**Type**	**Example**
id [PK]	Unique identification number for room	int	1
property_id [FK]	Unique identifier of property that has this room	int	1
room_floor_plan_id [FK]	Unique identifier of the room floor plan that stores the floor plan data of this room	int	1
room_name	Room name to display in the application	varchar	Living Room
modified_at	Date and time when the last modification was made to this data	datetime	2022-12-25 10:30:00
created_at	Time and date of data creation.	datetime	2022-12-25 10:30:00

Table 5.5 Wall Outlets Entity Data Dictionary

Fields	**Field's Description**	**Type**	**Example**
id [PK]	Unique identification number for socket	int	1
room_id [FK]	Unique identifier of a room that has this wall outlet	int	1
wall_outlet_name	Wall outlet name to display in the application	varchar	Wall Outlet A
wall_outlet_identifier	Unique identifier of a socket to query real-time data of the wall outlet from the Firebase database	varchar	2317-sdae-4332-gsdc
quick_access	To define if the wall outlet can be direct access from the homepage	boolean	1
daily_warning_value	A value settled by the user that will trigger a notification when daily energy consumption is more than it	double	1,000
daily_stop_value	A value settled by the user that will trigger a notification and switch off the wall outlet when daily energy consumption is more than it	double	1,200
monthly_warning_value	A value settled by the user that will trigger a notification when monthly energy consumption is more than it	double	10,000
monthly_stop_value	A value settled by the user that will trigger a notification and switch off the wall outlet when monthly energy consumption is more than it	double	12,000
modified_at	Date and time when the last modification was made to this data	datetime	2022-12-25 10:30:00
created_at	Time and date of data creation.	datetime	2022-12-25 10:30:00

Table 5.6 Wall Outlet Schedules Entity Data Dictionary

Fields	**Field’s Description**	**Type**	**Example**
id [PK]	Unique identification number for socket schedule	int	1
wall_outlet_id [FK]	Unique identifier of socket	int	1
start_time	Scheduled time to switch on the wall outlet	time	10:30:00
end_time	Scheduled time to switch off the wall outlet	time	10:30:00
status	Will this schedule need to be executed	boolean	true
modified_at	Date and time when the last modification was made to this data	datetime	2022-12-25 10:30:00
created_at	Time and date of data creation.	datetime	2022-12-25 10:30:00

Table 5.7 Notifications Entity Data Dictionary

Fields	**Field's Description**	**Type**	**Example**
id [PK]	Unique identification number for the notification	int	1
wall_outlet_id [FK]	Unique identifier of socket	int	1
notification_type	Type of notification either a warning or stop	enum	WARNING
title	The title of the information	varchar	Wall Outlet A is exceeding the warning level.
content	The detailed information about this notification	text	Wall Outlet A is exceeding the energy usage of 1,200 kW daily usage.
is_seen	Did the user see this notification	boolean	true
modified_at	Date and time when the last modification was made to this data	datetime	2022-12-25 10:30:00
created_at	Time and date of data creation.	datetime	2022-12-25 10:30:00

Table 5.8 Task Schedules Entity Data Dictionary

Fields	**Field’s Description**	**Type**	**Example**
id [PK]	Unique identification number for task schedule	int	1
wall_outlet_id [FK]	Unique identifier of socket	int	1
wall_outlet_identifier	Unique identifier of a socket to query real-time data of the wall outlet from the Firebase database	varchar	2317-sdae-4332-gsdc
action	Action to be done to the wall outlet, either ON or OFF	enum	ON
action_datetime	Record when the action will be executed.	datetime	2022-12-25 10:30:00
is_schedule	To define is the schedule is countdown or daily schedule	boolean	true
modified_at	Date and time when the last modification was made to this data	datetime	2022-12-25 10:30:00
created_at	Time and date of data creation.	datetime	2022-12-25 10:30:00

Table 5.9 Property Floor Plans Entity Data Dictionary

Fields	**Field's Description**	**Type**	**Example**
id [PK]	Unique identification number for property floor plan	int	1
data	Stringify JSON data that contains coordinates and sizes of a property floor plan	text	{}
modified_at	Date and time when the last modification was made to this data	datetime	2022-12-25 10:30:00
created_at	Time and date of data creation.	datetime	2022-12-25 10:30:00

Table 5.10 Room Floor Plans Entity Data Dictionary

Fields	**Field's Description**	**Type**	**Example**
id [PK]	Unique identification number for room floor plan	int	1
data	Stringify JSON data that contains coordinates and sizes of a room floor plan and also the position of the wall outlet in this room	text	{}
modified_at	Date and time when the last modification was made to this data	datetime	2022-12-25 10:30:00
created_at	Time and date of data creation.	datetime	2022-12-25 10:30:00

5.4 User Interface Design

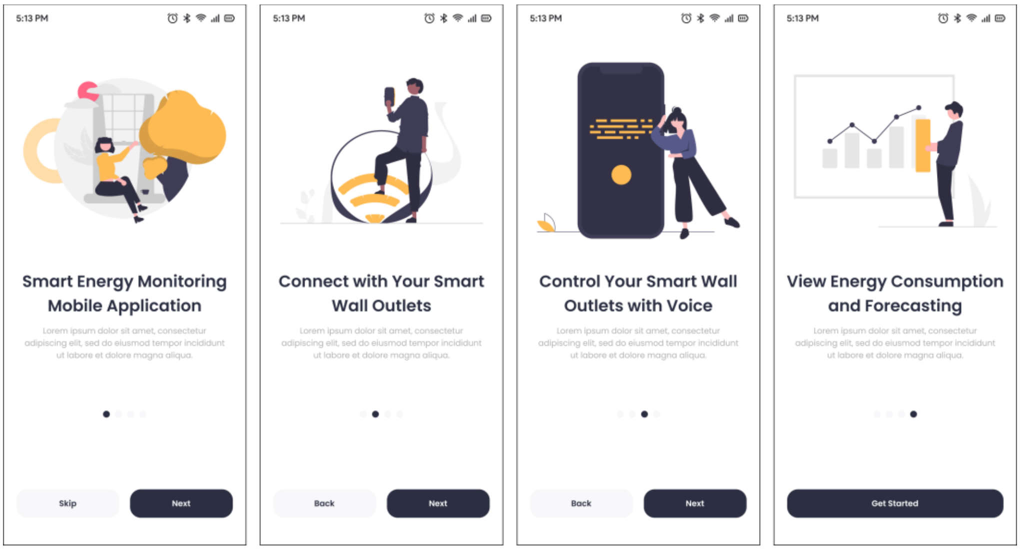
User interface design can present abstract requirements to the user through visual representation, ensuring they meet the user's expectations. Furthermore, when the design is approved, it can speed up the subsequent front-end development and reduce unnecessary changes.

5.4.1 Mock Up

The high-fidelity mock-up was created by using Figma. This mock is divided into nine parts: the welcome pages, user authentication pages, home pages, data display pages, add pages, voice recognition pages, notification pages, profile pages, and floor plan editor pages. The user interface for each part is designed to fulfil specific requirements.

5.4.1.1 Welcome Pages

Welcome pages in a mobile app are necessary because they provide a first impression and set the tone for the user experience. It often serves as an introduction to the app's features and functionality. Figure 5.3 shows the UI design of the welcome pages.



Smart Energy Monitoring Mobile Application

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Connect with Your Smart Wall Outlets

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Control Your Smart Wall Outlets with Voice

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

View Energy Consumption and Forecasting

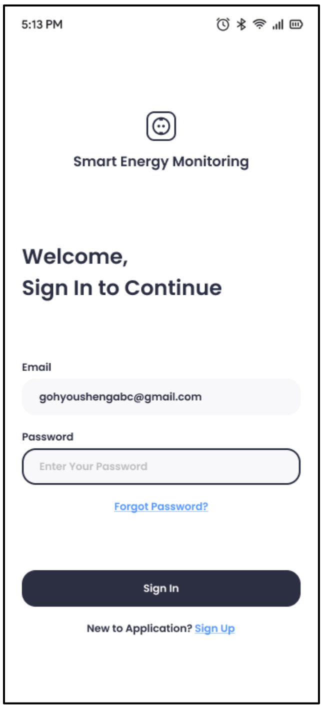
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Figure 5.3 UI Design of Welcome Pages

5.4.1.2 User Authentication Pages

The user authentication module has five pages: login page, register page, forgot password page, reset password page and one-time password page. Figure 5.4

shows the UI design of the login page. The user must fill in a valid email and password to access the application. Hypertext “Forgot Password” and “Sign Up” in light blue allow users to navigate to other pages. Upon successful login, the application will redirect to the One Time Password (OTP) page to perform two-factor authentication for increased security.



5:13 PM

Smart Energy Monitoring

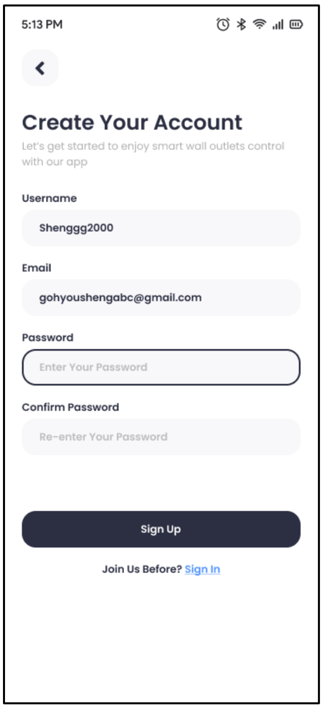
Welcome,
Sign In to Continue

Email: gohyoushengabc@gmail.com

Password: Enter Your Password

Figure 5.4 Login Page

Figure 5.5 shows the layout of the register page. A new user must register a new account to use the application. The user only needs to provide the application with username, email, and password information. The system will verify whether the email has been registered previously. If it hasn't, a confirmation email will be sent to the specified email address.



Create Your Account

Let's get started to enjoy smart wall outlets control with our app

Username: Shenggg2000

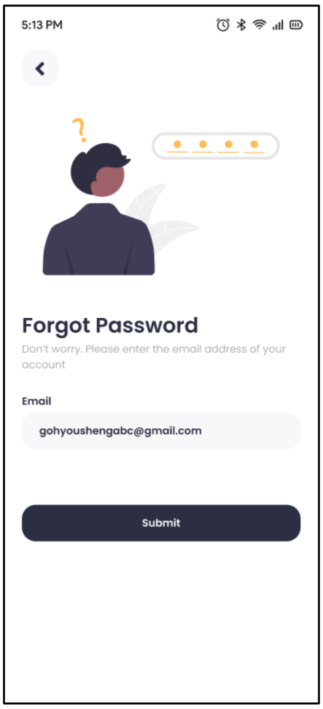
Email: gohyoushengabc@gmail.com

Password: Enter Your Password

Confirm Password: Re-enter Your Password

Figure 5.5 Register Page

Figure 5.6 shows the forgot password page. The user must fill in the email associated with their account. A reset password email will be sent to the email.



Forgot Password

Don't worry. Please enter the email address of your account

Email: gohyoushengabc@gmail.com

Figure 5.6 Forgot Password Page

Figure 5.7 shows the reset password page. A hyperlink in the reset password email will redirect the user to this page. Users can insert a new password for the account.

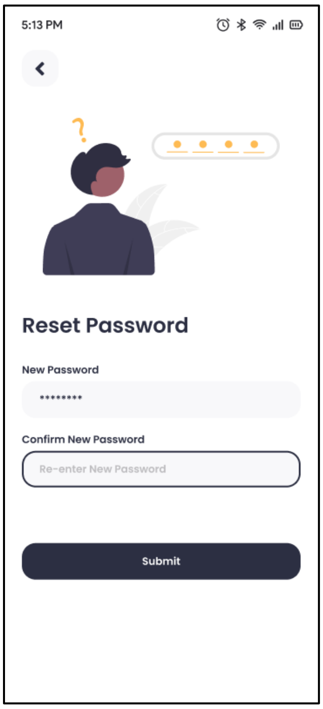
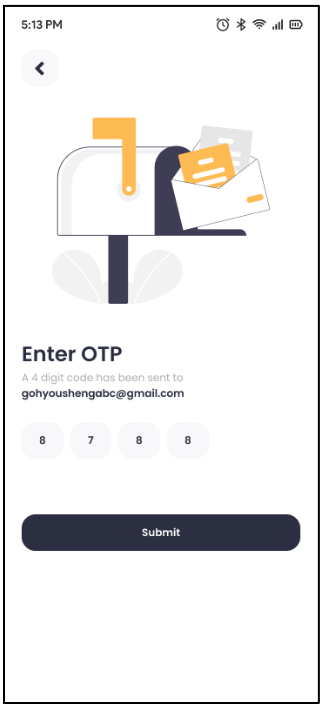


Figure 5.7 Reset Password Page

Figure 5.8 shows the enter OTP page. OTP is required when the user login their account on a new device.



5:13 PM

Enter OTP

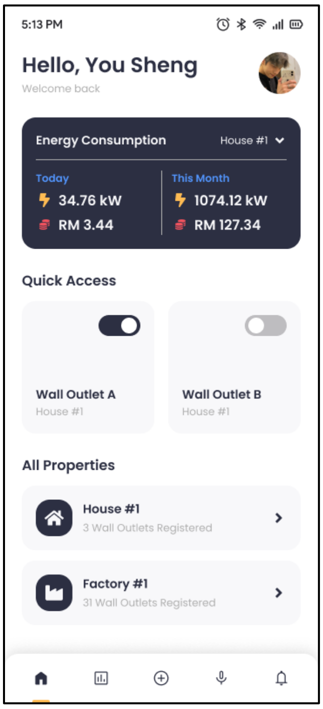
A 4 digit code has been sent to
gohyoushengabc@gmail.com

8 7 8 8

Figure 5.8 One Time Password Page

5.4.1.3 Home Pages

Figure 5.9 shows the homepage of the application. Clicking on the profile picture in the upper right corner will redirect to the profile page. The upper part of the homepage is a small module designed to showcase data on energy consumption. This is followed by quick access buttons that make it easy for users to switch on and off the wall outlets. Lastly, all the properties in this account will be shown. The bottom of the application also has a navigation bar with 5 modules to choose from: Home, Data, Add, Voice Assistant, and Notifications.



5:13 PM

Hello, You Sheng

Welcome back

Energy Consumption House #1

Today
34.76 kW
RM 3.44

This Month
1074.12 kW
RM 127.34

Quick Access

Wall Outlet A
House #1

Wall Outlet B
House #1

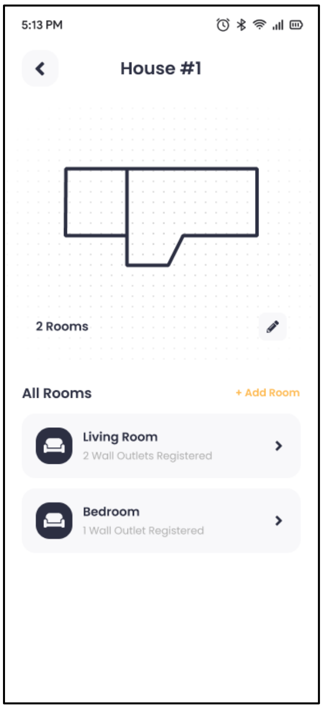
All Properties

House #1
3 Wall Outlets Registered

Factory #1
31 Wall Outlets Registered

Figure 5.9 Home Page

Figure 5.10 shows the property page where the property was a house named House #1. The first thing that catches the eye is the floor plan of the house, allowing the user to edit the floor plan freely. Below that is a list of all the rooms in the house.



5:13 PM

House #1

2 Rooms

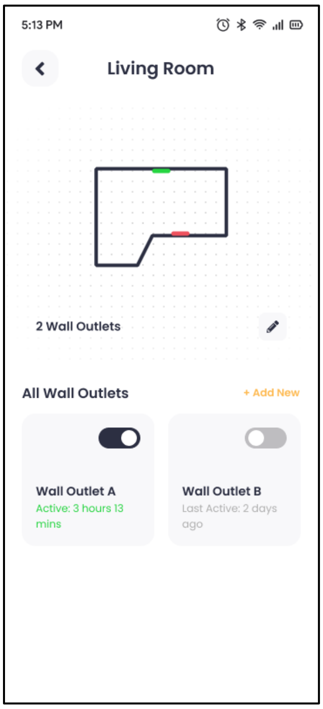
All Rooms + Add Room

Living Room
2 Wall Outlets Registered

Bedroom
1 Wall Outlet Registered

Figure 5.10 Property Page

Figure 5.11 shows the room page where the room was named Living Room. The upper section displays the room's floor plan, and the room's wall outlets will be marked in green or red. Green represents that the outlet is currently in use, while red indicates that the outlet is currently off. Users can switch the wall outlets in the list below or click on the wall outlet for more detailed settings.



5:13 PM

Living Room

2 Wall Outlets

All Wall Outlets

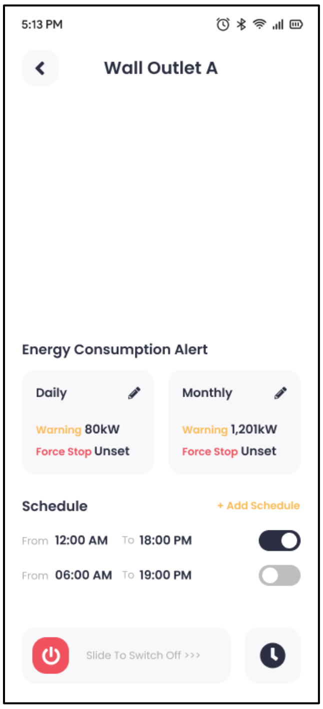
+ Add New

Wall Outlet A
Active: 3 hours 13 mins

Wall Outlet B
Last Active: 2 days ago

Figure 5.11 Room Page

Figure 5.12 shows the wall outlet page of Wall Outlet A. On this page, users can set detailed energy consumption values to trigger notifications and schedule wall outlets to turn on and off automatically. Energy consumption alerts are divided into daily or monthly. For wall outlet schedules, users can add multiple schedules and selectively enable them. Finally, the clock in the lower right corner can set a countdown timer so the wall outlet can be turned on or off after a specified time.



Energy Consumption Alert

Daily
Warning 80kW
Force Stop Unset

Monthly
Warning 1,201kW
Force Stop Unset

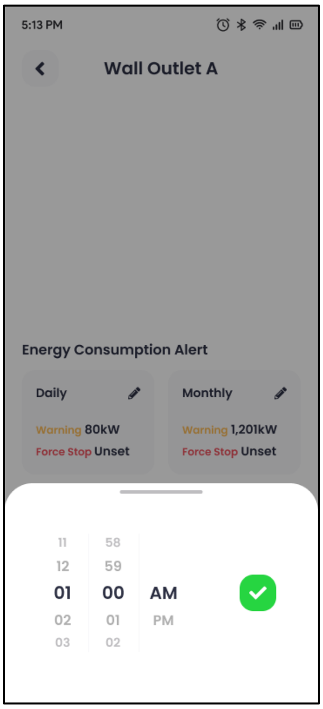
Schedule + Add Schedule

From 12:00 AM To 18:00 PM ●

From 06:00 AM To 19:00 PM ○

Figure 5.12 Wall Outlet Page

Figure 5.13 shows the time selection modal used to set the time of the wall outlet schedule.



Energy Consumption Alert

Daily
Warning 80kW
Force Stop Unset

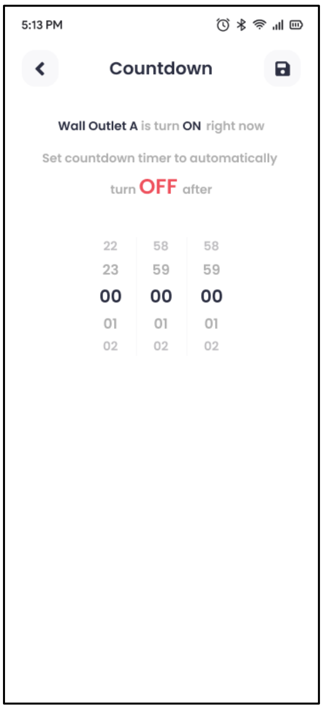
Monthly
Warning 1,201kW
Force Stop Unset

11	58	
12	59	
01	00	AM
02	01	PM
03	02	

✓

Figure 5.13 Set Schedule for Wall Outlet

Figure 5.14 shows the countdown setting page for the wall outlet. If a wall outlet is currently in an on state, the countdown will be used to turn it off. Conversely, if a wall outlet is currently in an off-state, it will be turned on once the countdown is over.

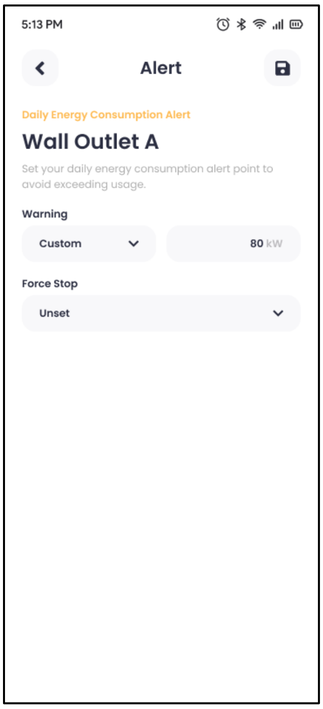


Wall Outlet A is turn ON right now

Set countdown timer to automatically turn OFF after

Figure 5.14 Countdown Setting Page for Wall Outlet

Figure 5.15 shows the wall outlet’s energy consumption alert setting page. The alert is divided into two types: warning and forced stop. A warning will only notify the user through a notification, while a forced stop will directly turn off the outlet to prevent unnecessary consumption. The user can choose whether or not to set this up.



5:13 PM

Alert

Daily Energy Consumption Alert

Wall Outlet A

Set your daily energy consumption alert point to avoid exceeding usage.

Warning

Custom

80 kW

Force Stop

Unset

Figure 5.15 Wall Outlet Energy Consumption Alert Setting Page

5.4.1.4 Data Display Pages

Figure 5.16 shows the energy consumption summary page regarding energy in unit kilowatts. The data summarizes the energy consumption of all properties. If the user wants to get data for an individual property, the user can select it from the list below.

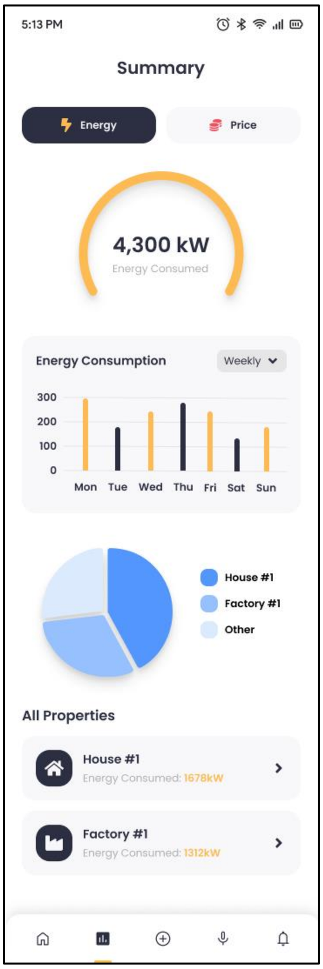


Figure 5.16 Energy Consumption Summary Page

Figure 5.17 shows the energy consumption summary page regarding the price to pay in Ringgit Malaysia (RM) units.

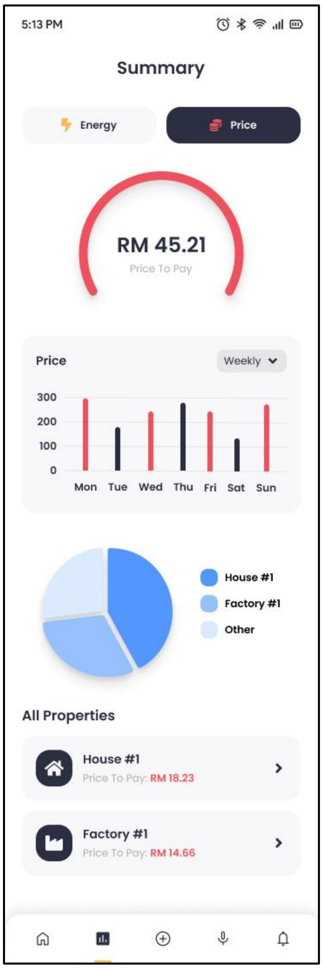


Figure 5.17 Energy Consumption (in Price) Summary Page

5.4.1.5 Add Pages

Figure 5.18 shows the general add page. There are 3 types of items can be added: properties, rooms, and wall outlets. The user can add a new property, add a new room to an existing property, or add a new wall outlet to an existing room.

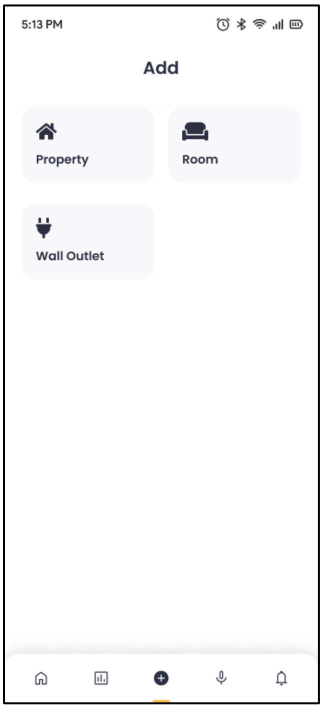
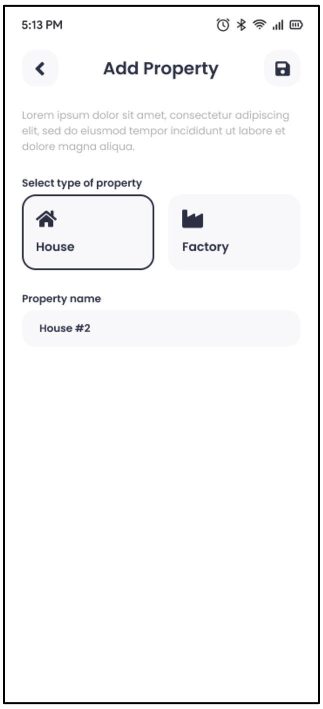


Figure 5.18 Add Page

Figure 5.19 shows the add property page. When creating a new property, the user needs to select the type of property, either for residential or industrial use, and give it a name.



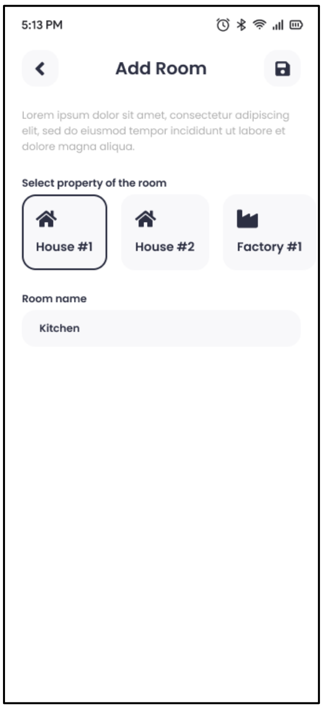
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Select type of property: [✓ House] Factory

Property name: House #2

Figure 5.19 Add Property Page

Figure 5.20 shows the add room page. When creating a new room, the user must select the property the room belongs to and give it a name.



Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Select property of the room: House #1

Room name: Kitchen

Figure 5.20 Add Room Page

Figure 5.21 shows the scan wall outlet QR code page. When adding a new wall outlet, the user needs to get the wall outlet’s identifier by scanning the QR code on the wall outlet.

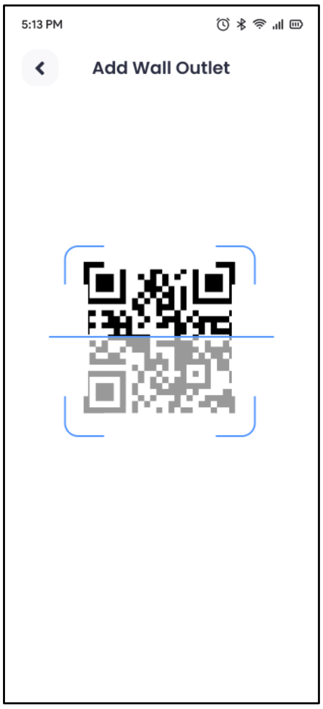
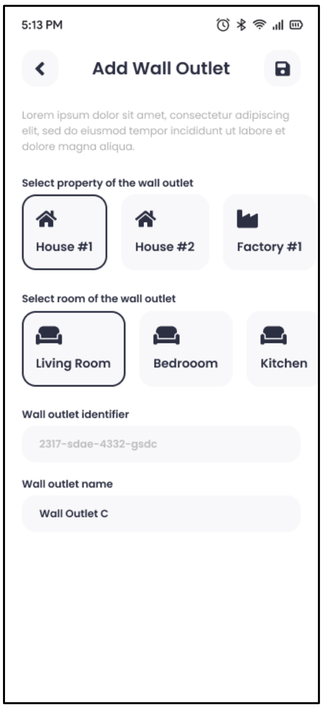


Figure 5.21 Scan Wall Outlet QR Code Page

Figure 5.22 shows the add wall outlet page. When the QR code is successfully recognized, the user can choose the property and room the wall outlet belongs to and name it.



Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Select property of the wall outlet: House #1 | House #2 | Factory #1

Select room of the wall outlet: Living Room | Bedrooom | Kitchen

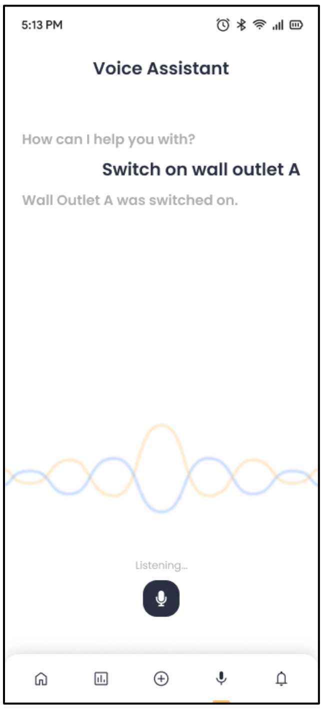
Wall outlet identifier: 2317-sdae-4332-gsdc

Wall outlet name: Wall Outlet C

Figure 5.22 Add Wall Outlet Page

5.4.1.6 Voice Assistant Page

Figure 5.23 shows the voice assistant page. The user can give voice commands, which will be analysed and displayed as text on the page. The application will then perform actions based on the commands.



5:13 PM

Voice Assistant

How can I help you with?

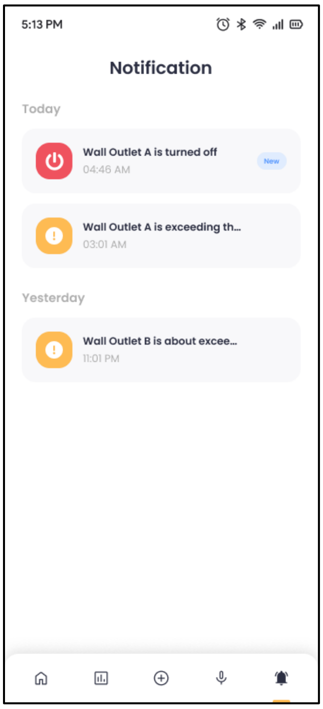
Switch on wall outlet A

Wall Outlet A was switched on.

Figure 5.23 Voice Assistant Page

5.4.1.7 Notification Pages

Figure 5.24 shows the notification list page. Different types of notifications have different icons. A yellow exclamation mark represents a warning that the wall outlet has used more energy than the threshold. A red stop icon represents a wall outlet that has been forced to shut off. Notifications will be sorted from the newest to the oldest.



5:13 PM

Notification

Today

Wall Outlet A is turned off
04:46 AM

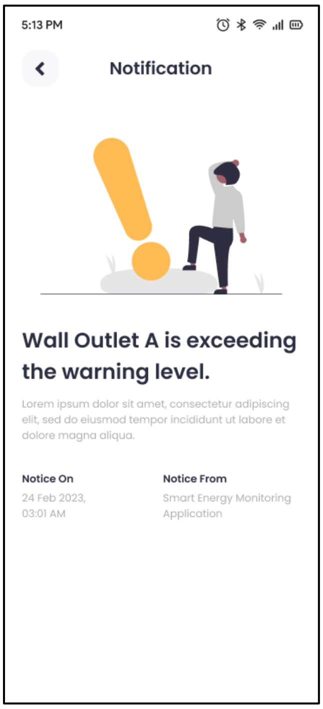
Wall Outlet A is exceeding th...
03:01 AM

Yesterday

Wall Outlet B is about excee...
11:01 PM

Figure 5.24 Notification List Page

Figure 5.25 shows the notification detail page. More information will be displayed on this page.



5:13 PM

Notification

Wall Outlet A is exceeding the warning level.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

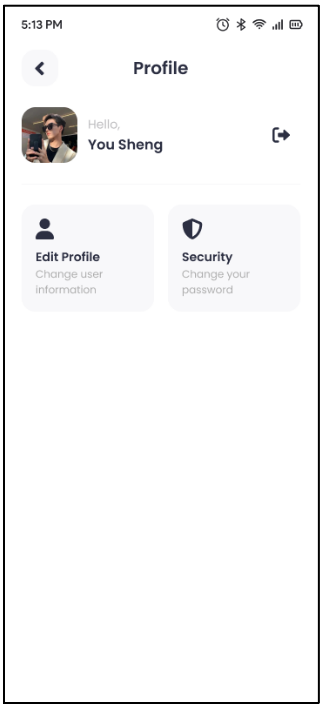
Notice On
24 Feb 2023,
03:01 AM

Notice From
Smart Energy Monitoring Application

Figure 5.25 Notification Detail Page

5.4.1.8 Profile Pages

Figure 5.26 shows the profile page of the application.


5:13 PM

Profile

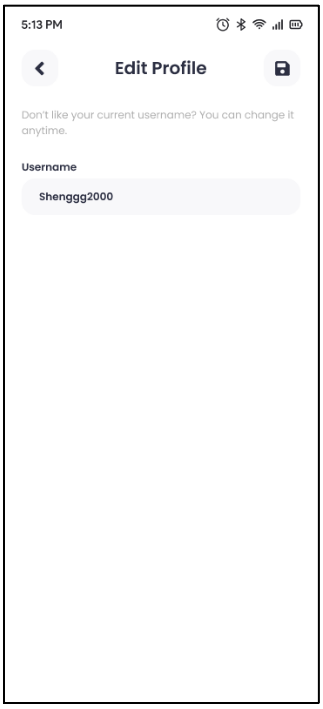
Hello,
You Sheng

Edit Profile
Change user information

Security
Change your password

Figure 5.26 Profile Page

Figure 5.27 shows the edit profile page. Users can change their username.


5:13 PM

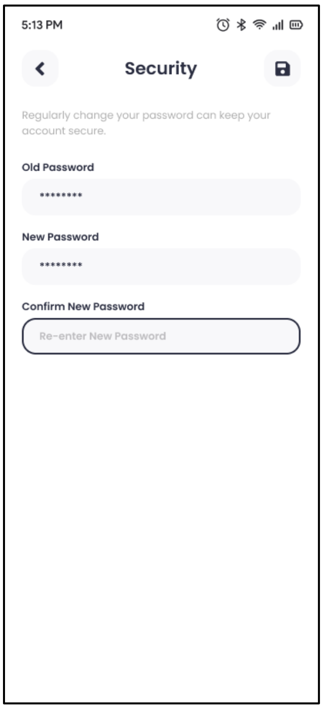
Edit Profile

Don't like your current username? You can change it anytime.

Username
Shenggg2000

Figure 5.27 Edit Profile Page

Figure 5.28 shows the security page in profile module. The current security settings for the application only consist of password changes. Users can periodically change their password to enhance the security level.



Regularly change your password can keep your account secure.

Old Password: ********

New Password: ********

Confirm New Password: Re-enter New Password

Figure 5.28 Profile Security Page

5.4.1.9 Floor Plan Editor Pages

Figure 5.29 shows the property floor plan editor page. On this page, the user cannot adjust the shape of the room; only the size and position can be adjusted. The user can select the room to adjust through the toolbar below.

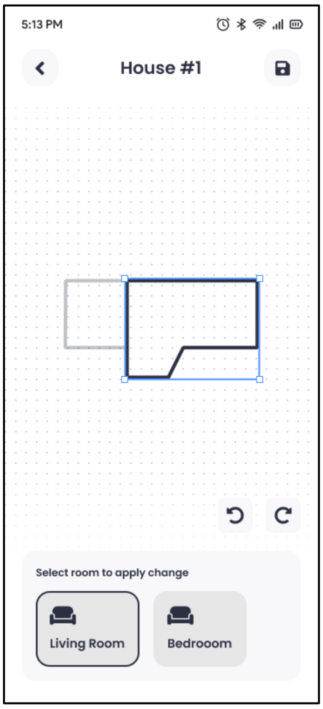


Figure 5.29 Property Floor Plan Editor Page

Figure 5.30 shows the room floor plan editor page. By default, this page is used for adjusting the shape of the room. The user can make changes by dragging the blue dots on the line of the room.

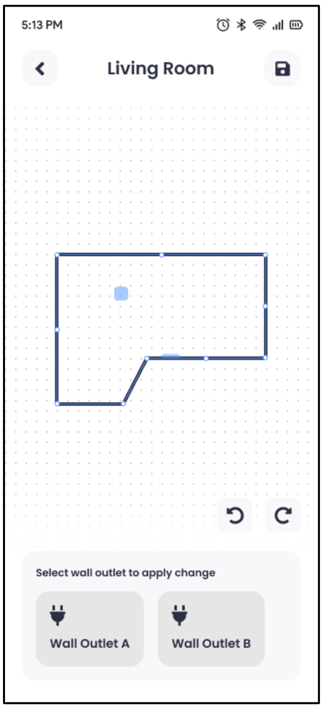


Figure 5.30 Room Floor Plan Editor Page

Figure 5.31 shows the property floor plan editor page in the condition where the user determines the location of a wall outlet. This state will be triggered when the user selects one of the wall outlets from the toolbar. The user will have control over the wall outlet, which can be freely placed anywhere in the room.

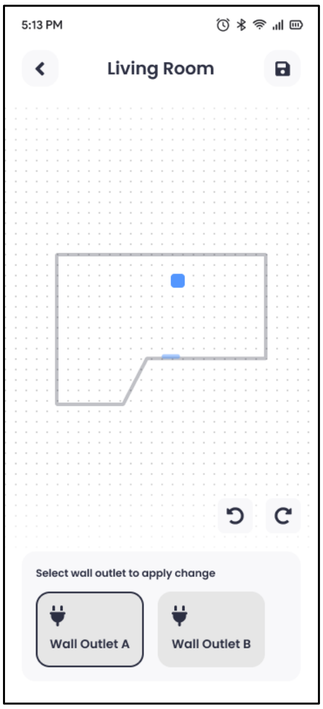


Figure 5.31Room Floor Plan Editor Page (Locate Wall Outlet Position)

CHAPTER 6

IMPLEMENTATION

6.1 Introduction

This section outlines the implementation of the Smart Energy Monitoring Application, separating it into two key areas: frontend and backend. The frontend implementation incorporates the creation of five modules utilizing React Native, encompassing profile, control, monitoring, voice recognition, and notification modules. The backend implementation concentrates on elucidating the authentication method, monitoring real-time usage, scheduling database actions, and developing create, read, update, and delete (CRUD) APIs. A comprehensive overview of the implementation process is presented in this chapter.

6.2 Frontend Implementation

Before the development of each module, the project files undergo a series of initialization settings, which include:

1. Installation of libraries/dependencies of the project.
2. Installation of required fonts.
3. Management of project assets, encompassing necessary icons and images.
4. Creation of components to avoid redundant code and improve project maintainability.
5. Establishment of navigation and routes.
6. The setting of project constants, such as colours used.

These settings are organized into their respective folders to enhance project maintainability.

6.2.1 Profile Module

The Profile Module contains all user-related functions, such as registration, login, email verification, username change, password change, and log in.

JSON Web Token

The application uses JavaScript Object Notation Web Token (JWT) Authentication. Each user will have a unique access token. Upon successful registration or login with the correct email and password, the API will return the access token. The access token will be stored permanently in the application's storage unless the user clears the storage himself. The AsyncStorage library is implemented to help with this task. If an access token is detected in the storage at application startup, the application will automatically redirect the user to the homepage without undergoing the login process repeatedly.

Verify the registered email.

To confirm the legitimacy of an email address, a verification email will be dispatched to the specified email address to ensure that the email belongs to the user. The verification email will be sent whenever registration is completed or when logging in to an unverified account. After completing the verification, the user will be granted access to the home page. If the user does not receive the email, they can re-request a new verification email after 60 seconds. The user only needs to click on the hyperlink inside the email to complete the verification. The hyperlink is encrypted so that it cannot be easily broken.

Register, log in, and change password.

When a user registers, the system verifies that the email address entered by the user has not been registered. The application ensures that all information is filled in and the password and confirmation password are the same. These basic verifications are also applied to login and password change features.

6.2.2 Control Module

The Control Module contains all features related to controlling the wall outlet, such as switching on and off a wall outlet, setting a timer or schedule to control the wall outlet and setting the wall outlet energy usage alert level.

Control wall outlet

When the user comes to the wall outlet page, the most striking thing is the button of the wall outlet controller at the lowermost part of the page. Unlike the regular click button, this button is sliding. Although it is more complicated to implement, it reduces the possibility of incorrect operation. To implement the sliding controller, the first step is to implement the PanGestureHandler component to listen to the user's gestures. In addition, the animated component is also implemented to realise the movement of the button. The threshold value of the sliding controller is set to 50%. The action is not recognised as valid when the user lets go of the button, and the dragging progress is less than 50%. Conversely, it is a valid action if it exceeds 50%.

Set timer and schedule,

Timer setter and schedule time setter two have a similar UI design. Both setters' hour, minute and second are selected by turning the scroll wheel. Compared to the regular input box, this is more interactive and avoids entering values incorrectly. The scrolling of values is achieved by implementing an animated list. The currently selected value is highlighted. The selected time will be set as a scheduled task and saved in the database. The purpose of storing these tasks is to ensure that tasks are run on time, regardless of whether the mobile application is running.

Set wall outlet alert level.

Users can set an alert level to avoid excessive energy consumption. The reason for using the bill's price as the alert unit is that users are usually price sensitive and lack the knowledge and concept of the energy unit, W/h. There are two types of alerts, notification alerts and forced stop alerts. In the case of notification alerts, the socket will be kept running, and the user will have to decide whether to turn the wall outlet off or not by himself.

6.2.3 Monitoring Module

The Monitoring Module graphically displayed the energy consumption data. This module is presented in a hierarchical and cascading manner. First, the total energy consumption of all properties will be displayed, then the energy consumption of a single property, next the energy consumption of a single room and lastly, the energy consumption of a single wall outlet. Despite this, at either stage, the data will be presented in two ways, either in terms of energy or in terms of bill price. In addition, the user can also choose the length of time the data is displayed, which the app currently supports monthly and weekly. The data is visually displayed in the chart, and to achieve this presentation, the project implemented ‘react-native-chart-kit’, a third-party library. The chart was also customised during the implementation process to integrate the chart into the application better and ensure consistency of style.

6.2.4 Voice Recognition Module

The voice recognition module mainly completes converting the user’s voice into commands. First of all, the application needs permission to access the user’s microphone. The user can start the voice recording by tapping the record button at the lowermost part of the page and end the recording by clicking again. To recognise the voice, the application implements a third-party voice recognition library, Picovoice. After that, the audio sampling frequency is set. Audio sampling begins when the record button is clicked, and the audio data is stitched together until the record button is clicked again. The audio data will then form a complete audio file. This file will be analysed by Picovoice’s Leopard tool to obtain the user’s commands. If the command is understood, the application will execute the command and inform the user. Conversely, if the commands are not understood, the application will ask the user to record again to re-analyse it. Currently, the application processes commands by pairing keywords, and the following are the keywords that can currently be understood. Table 6.1 illustrates keywords that the voice recognition module can identify to execute actions with examples of commands.

Table 6.1 Keywords to execute action with examples of commands.

Action	Action keyword	Example command
To switch on a wall outlet	Switch on	Switch on {wallOutletName}.
	Turn on	Turn on {wallOutletName}.
To switch off a wall outlet	Switch off	Switch off {wallOutletName}.
	Turn off	Turn off {wallOutletName}.

6.2.5 Notification Module

This module performs the function of receiving and managing notifications. Users will only receive notifications about energy consumption triggered by exceeding alert values. The application will generate a messaging token for the device to receive notifications when the user successfully logs in. The messaging token was generated by implementing the 'react-native-firebase/messaging' library provided by Firebase. Each device has its messaging token, which will be essential for the system to find the device that wants to receive notifications. The token will be saved in both the application storage and the database. The messaging token in the application storage will be used to record whether or not the current device has generated a messaging token. Whenever the energy consumption exceeds the alert value, the system will look for the messaging token in the database and send a notification to the correspondence device.

6.2.6 Property Management Module

This module completes the function of adding and managing properties.

Add Property, room and wall outlet.

Each user has to add at least one property before adding a room to that property. Likewise, each user must add at least one room before adding an outlet to that room. When adding a property, the user selects the type of property, which will be used to calculate the bill price. Based on the TNB, different properties have different pricing and tariffs. When adding a wall outlet, the user is required to scan the QR code on the wall outlet to get a unique identifier for that wall outlet. The backend server must use this identifier to get its energy consumption data.

Floor Plan Design

This module also contains the function of floor plan design for property and room. As shown in Table 6.2, three items have different things that can be adjusted in the floor plan editor.

Table 6.2 Things to Adjust for Each Item in Floor Plan Editor

Items	Things to adjust
Property	Adjusts the position of the rooms on the property
Room	Adjusts the shape of the room
Wall outlet	Adjusts the position of the wall outlet in the room it belongs to

Scalable Vector Graphics (SVG), a vector file format with the third-party library ‘react-native-svg’, is used in the application to implement this feature. Firstly, if the user never adjusts the room, the room is square by default. Users can adjust the shape of the room by dragging the four corners, or they can click the plus button on the edge of the room to add new corners to form a room in a pentagon shape. After the user clicks on any corner, the application must listen to the location of the user’s touch and re-render the SVG based on the latest location to achieve real-time room shape adjustment. The wall outlet position adjustment is implemented similarly to the room shape adjustment, which also listens to the user’s touch location to re-render the SVG in real-time.

6.3 Backend Implementation

Node.js and express.js are used to implement the backend functionality. The reason for choosing this framework is that it is small and can easily develop flexible interfaces since this project uses Firebase real-time database and a MySQL relational database. It is necessary to have the backend server establish connections to these databases before developing additional functionality. Therefore, it is needed to provide the third-party library ‘mysql2/promise’ and “firebase-admin” with the appropriate certificates for the database.

6.3.1 Authentication

The backend server must generate an access token for the client application to store it after the user has provided the correct authentication information. Therefore, the backend server uses the sign() method provided by ‘jsonwebtoken’ to compose a unique access token from the registered email and the system’s private key. This access token must also be provided again by the client to verify the user’s identity. The backend server will use the verify() method provided by ‘jsonwebtoken’ with the system’s private key to confirm the validity of the access token. The middleware verifies this authentication process, so all CRUD actions are verified before they are executed.

Email verification

Once the user has finished registering, the backend server will dispatch an email to verify the email address. To achieve this, a mailbox for sending emails is registered as “smart.energy.monitoring2023@gmail.com”. For the backend server to access this Google account, the account needs to create a dedicated password for the third-party application to the backend server to access. These steps can be done in the security settings of the Google account. The backend server will log in to the account via “nodemailer” and dispatch an email to the registered email address using the sendMail() method.

6.3.2 Real-time usage

When the wall outlet is turned on, it will add energy consumption data to the Firebase real-time database every 12 seconds. Therefore, the backend server will subscribe and connect with the real-time database. Whenever the latest information is added, the backend server will get the latest bill price. If the bill price exceeds the settled alert price, a notification is sent to the device based on the messaging token of the user to which the outlet belongs. The real-time database uses timestamps as keys for energy consumption data, so the backend server can quickly discover the latest data based on the value of the timestamp, where the larger the timestamp, the latest data is.

6.3.3 Schedule

The mobile application needs to accomplish some tasks that will happen in the future, e.g., turn off a wall outlet after 10 minutes, turn on a wall outlet at 6:00 a.m. and off at 10:00 p.m. every day. Therefore, the backend server needs to create a task schedule. The task schedule is also synchronized to the database to avoid the schedule's loss due to the backend server's service interruption.

Multiple microservices in the backend server access and change the task schedule. This may result in conflicting schedules being obtained by different microservices. To avoid conflicts, the singleton design pattern is used. The singleton design pattern ensures only one instance of the schedule for the entire backend server. Therefore, no matter which microservice requests the schedule, the server will only return the same and only instance.

6.3.4 API List

Mobile applications access databases to store or retrieve information through HTTP requests. Different routes are created to operate on different tables. Table 6.3 shows the breakpoints, HTTP methods, parameters and description of the different routes.

Table 6.3 APIs List of Different Backend Services

Service: Auth				
Endpoint	**HTTP method**	**Parameter**	**Authentication needed**	**Description**
/auth/register	POST	{email, password, username, messaging_token}	False	Create a new user account.
/auth/send-verify-email	POST	{email}	False	Send email address verification email.
/auth/verify/{token}	GET	-	False	Update email address as verified.
/auth/login	POST	{email, password, messaging_token}	False	Log in to account.
/auth/is-verified	POST	{email}	False	Check if the email address verified
/auth	GET	-	True	Retrieve user account information.
/auth	PUT	{username}	True	Update account information.

/auth/password	PUT	{oldPassword, newPassword}	True	Update account password.
Service: Property				
Endpoint	**HTTP method**	**Parameter**	**Authentication needed**	**Description**
/properties	GET	-	True	Retrieve all property records of a user
/properties/{id}	GET	-	True	Retrieve a property record of a user
/properties/summary/{id}	GET	-	True	Retrieve a summary of a property's energy consumption record
/properties/overview	GET	-	True	Retrieve a detail of all properties' energy consumption record
/properties/usage/{id}	GET	-	True	Retrieve a detail of a property's energy consumption record
/properties	POST	{floor_plan_data, property_name, property_type, property_tariff}	True	Create a new property
/properties/{id}	PUT	{property_name, property_type}	True	Update a property information

/properties/floor-plan/{id}	PUT	{floor_plan_data}	True	Update a property’s floor plan information
/properties/{id}	DELETE	-	True	Delete a property
Service: Room				
Endpoint	**HTTP method**	**Parameter**	**Authentication needed**	**Description**
/rooms	GET	-	True	Retrieve all room records of a property
/rooms/{id}	GET	-	True	Retrieve a room record of a property
/rooms/usage/{id}	GET	-	True	Retrieve a detail of a room’s energy consumption record
/rooms	POST	{floor_plan_data, property_id, room_name}	True	Create a new room
/rooms/{id}	PUT	{room_name}	True	Update a room information
/rooms/floor-plan/{id}	PUT	{floor_plan_data}	True	Update a room’s floor plan information
/rooms/{id}	DELETE	-	True	Delete a room

Service: Wall Outlet				
Endpoint	**HTTP method**	**Parameter**	**Authentication needed**	**Description**
/wall-outlets	GET	-	True	Retrieve all wall outlet records of a room
/wall-outlets/{id}	GET	-	True	Retrieve a wall outlet record of a room
/wall-outlets/usage/{id}	GET	-	True	Retrieve a detail of a wall outlet's energy consumption record
/wall-outlets	POST	{wall_outlet_name, wall_outlet_identifier, room_id}	True	Create a new wall outlet
/wall-outlets/schedules	POST	{wall_outlet_id, start_time, end_time}	True	Create a new schedule for a wall outlet
/wall-outlets/{id}	PUT	{wall_outlet_name}	True	Update a wall outlet information
/wall-outlets/daily/{id}	PUT	{daily_warning_value, daily_stop_value}	True	Update a wall outlet daily alert value
/wall-outlets/monthly/{id}	PUT	{monthly_warning_value, monthly_stop_value}	True	Update a wall outlet monthly alert value

/wall-outlets/schedules/{id}	PUT	{start_time, end_time, status}	True	Update a wall outlet schedule time
/wall-outlets/on/{id}	PUT	-	True	Update a wall outlet state as on
/wall-outlets/off/{id}	PUT	-	True	Update a wall outlet state as off
/wall-outlets/likely/on	PUT	{wall_outlet_name}	True	Update a wall outlet state as on where it has a similar wall outlet name.
/wall-outlets/likely/off	PUT	{wall_outlet_name}	True	Update a wall outlet state as off where it has a similar wall outlet name.
/wall-outlets/set-quick-access/{id}	PUT	-	True	Update a wall outlet as a quick accessible wall outlet
/wall-outlets/unset-quick-access/{id}	PUT	-	True	Update a wall outlet become a not quick accessible wall outlet
/wall-outlets/{id}	DELETE	-	True	Delete a wall outlet
/wall-outlets/schedules/{id}	DELETE	-	True	Delete a wall outlet schedule

Service: Notification				
Endpoint	**HTTP method**	**Parameter**	**Authentication needed**	**Description**
/notifications	GET	-	True	Retrieve all notification records of a user
/notifications/{id}	GET	-	True	Retrieve a notification record of a user
/notifications	POST	{wall_outlet_id, notification_type, title, content}	False	Create a new notification
/notifications/seen/{id}	PUT	-	True	Update a notification status as seen.
Service: Tasks Schedule				
Endpoint	**HTTP method**	**Parameter**	**Authentication needed**	**Description**
/task-schedules	GET	-	False	Retrieve all tasks to be done.
/task-schedules	POST	{wall_outlet_id, wall_outlet_identifier, action, action_datetime, is_schedule}	False	Create a new task to be done
/task-schedules/{id}	PUT	{action_datetime}	False	Update task action datetime.
/task-schedules/{id}	DELETE	-	False	Delete a task

/task-schedules/countdown/{id}	DELETE	-	False	Delete a task when the countdown event is cancelled.

6.4 Summary

In conclusion, there is room for improvement in the implementation of the Smart Energy Monitoring Application, particularly in terms of algorithm complexity, coupling, and cohesion. By reducing the complexity of the algorithms, the application can perform computations more efficiently and with less computational power, resulting in shorter response times.

CHAPTER 7

SYSTEM TESTING

7.1 Introduction

This chapter documents the testing-related activities of the project. It includes the test plan, all the tests that have been completed, and the test results. The tests completed for this project are unit tests for the mobile application, APIs, and user acceptance tests.

7.2 Test Plan

This test plan serves as a critical document that aims to enhance the efficiency and effectiveness of the testing process. It entails defining the test scope, item, modules or services to be tested, as well as those that will not be tested, strategies, and exit and entry criteria. The test plan will outline the testing process, ensuring that all activities are carried out accurately and comprehensively. Additionally, this document will identify the resources required, such as environments and tools, to complete the test project. The main goal of this test plan is to mitigate system errors and failures of the smart energy monitoring system. Ultimately, upon test closure, the tested functionality should perform flawlessly, and no bug reports should be issued.

Test Scope

The testing scope encompasses both the frontend and backend components of the application. Since front-end and back-end separation development models are employed, testing for the front-end and back-end will be conducted separately. Moreover, user acceptance tests will be executed after the integration of the project to validate that the user requirements are adequately addressed. The following are the test activity that will be conducted for this project:

1. Unit testing of the mobile application.
2. Unit testing of the API.
3. User acceptance testing.

Test Basis

The following documents will provide the basis for creating test cases:

1. Project’s requirement specification
2. Smart Energy Monitoring System Use Case Diagram
3. Smart Energy Monitoring System Entity Relationship Diagram
4. Smart Energy Monitoring System API List

Test Items

Table 7.1 outlines the modules or services of the system planned for testing. Meanwhile, Table 7.2 outlines the modules that are excluded from testing:

Table 7.1 Modules or services to be tested.

Frontend: Mobile Application	
Module name	**Description**
Profile module	The main validation of this module is to test out the availability to send HTTP requests to the server when valid data is provided during register, login and other profile-related actions. In addition, the unit test should be able to confirm HTTP requests will not send to the server when invalid data is found to reduce the server burden.
Control module	The main validation of this module is to test out the correct user action, and data about controlling a wall outlet was collected before sending an HTTP request.
Monitoring module	This module's main validation is to test the availability of data being displayed in multiple formats correctly.
Property management module	The main validation of this module is to ensure error-free when conducting the property, room and wall outlet creation process.

Backend: APIs	
Service name	**Description**
Auth service	The main validation of this service is to ensure the API can effectively verify user identity, generate authentication tokens, and send verification emails.
Property service	The main validation of this service is to ensure the API can complete the creation, retrieval, modification and deletion of property in multiple scenarios.
Room service	The main validation of this service is to ensure the API can complete the creation, retrieval, modification and deletion of a room in multiple scenarios.
Wall outlet service	The main validation of this service is to ensure the API can complete the creation, retrieval, modification and deletion of wall outlets in multiple scenarios.
Notification service	The main validation of this service is to ensure the API can complete the creation of notifications with the push notification feature.
Task schedule service	The main validation of this service is to ensure the API can complete the creation of a task and execute the task correctly on a determined date and time.

Table 7.2 Modules not to be tested.

Frontend: Mobile Application	
Module name	**Description**
Notification module	The notification module was not being tested because it just simply presents the list of notifications and notification detail with no complex operations involved
Voice Recognition module	The voice recognition module was not tested because this module consists of lots of uncertainty.

	The actual results depend, to a large extent, on the users who use it.

Test Strategy

The testing strategy used in this test is the behavioural testing strategy. Behavioural testing is a testing strategy that focuses on the external behaviour of a system or application without any knowledge of its internal structure. Behavioural testing aims to test how the system responds to different inputs and behaves under different conditions.

Test Condition

The unit tests planned for this project will be carried out manually by the developers with the aid of various testing tools. There will be no automated testing tools or code used. On the other hand, user acceptance testing will be conducted by invited testers who are end-users of the system.

Entry Criteria

The testing can begin when the entry criteria listed are met.

1. The entire test project has already been scheduled in detail.
2. All features to be tested were completed and functioned stably.
3. All relevant resources, such as tools and devices, have been set up in the test environment.

Exit Criteria

The testing can end when the exit criteria listed are met.

1. All test cases have been executed and passed.
2. All defects found during the test phase are corrected and closed.
3. No critical defects were outstanding.
4. No changes to the codes and design.

7.3 Unit Test

Unit testing is a software testing methodology that involves isolating and testing individual units or components of a software system. Unit testing improves the quality and dependability of the software system through early defect detection, enhanced code quality, faster feedback, enhanced documentation, and simplified refactoring. The frontend and backend will be tested separately, which has the advantages that narrowing the scope of detection and identifying the root cause of the problem more quickly.

7.3.1 Mobile Application Unit Test

In order to assure the proper functionality of a mobile app, the project performed unit tests by manually inserting values, clicking buttons, and conducting various actions within an emulator. A total of eleven unit tests were executed on the mobile application, utilizing the test instruction and data outlined within the designated test cases. The testers then documented and compared the actual and expected results of the tests to properly assess and evaluate the system's performance. Table 7.3 until Table 7.13 demonstrate the eleven test cases of the mobile application unit test.

Table 7.3 Test Case APP-1 - Register

Test Case #	APP-1	**Test Case Title**	Register	**Module**	Profile
Test Case Overview	To check a user account able to be created or not.				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Register an account with a valid email, username, password and confirm password.	1. Enter email	test@gmail.com	HTTP request sent.	HTTP request sent.	Pass
	2. Enter username	Test123			
	3. Enter password	12345678			
	4. Enter confirm password	12345678			
	5. Tap register button	-			

Register account with empty input field.	1. Tap register button	-	Error message display without sending HTTP request.	Error message display without sending HTTP request.	Pass
Register account with valid email, username password but not match confirm password.	1. Enter email	test@gmail.com	Error message display without sending HTTP request.	Error message display without sending HTTP request.	Pass
	2. Enter username	Test123			
	3. Enter password	12345678			
	4. Enter confirm password	87654321			
	5. Tap register button	-			

Table 7.4 Test Case APP-2 - Login

Test Case #	APP-2	**Test Case Title**	Login	**Module**	Profile
Test Case Overview	To check if users can login to their account.				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Login account with valid email, username password.	1. Enter email	test@gmail.com	HTTP request sent.	HTTP request sent.	Pass
	2. Enter password	12345678			
	3. Tap login button	-			
Login account with empty input field.	1. Tap login button	-	Error message display without sending HTTP request.	Error message display without sending HTTP request.	Pass

Table 7.5 Test Case APP-3 - Edit Username

Test Case #	APP-3	**Test Case Title**	Edit Username	**Module**	Profile
Test Case Overview	To check if user can edit his username.				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Edit username with valid new username.	1. Enter new username	TestABC	HTTP request sent.	HTTP request sent.	Pass
	2. Tap save button	-			
Edit username with empty input field.	1. Tap save button	-	Error message display without sending HTTP request.	Error message display without sending HTTP request.	Pass

Table 7.6 Test Case APP-4 - Edit Password

Test Case #	APP-4	**Test Case Title**	Edit Password	**Module**	Profile
Test Case Overview	To check if user can edit his password.				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Edit password with valid new password and confirm new password.	1. Enter old password	12345678	HTTP request sent.	HTTP request sent.	Pass
	2. Enter new password	abcdefgh			
	3. Enter confirm new password	abcdefgh			
	4. Tap save button	-			
Edit password with empty input field.	1. Tap save button	-	Error message display without sending HTTP request.	Error message display without sending HTTP request.	Pass
Edit password with valid new password but not match confirm password.	1. Enter old password	12345678	Error message display without sending HTTP request.	Error message display without sending HTTP request.	Pass
	2. Enter new password	abcdefgh			
	3. Enter confirm new password	hgfedcba			
	4. Tap save button	-			

Table 7.7 Test Case APP-5 - Switch Wall Outlet State

Test Case #	APP-5	**Test Case Title**	Switch Wall Outlet State	**Module**	Control
Test Case Overview	To check if user can switch wall outlet state.				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Switch on wall outlet	1. Swipe sliding controller to turn on	-	HTTP request sent with turn on state.	HTTP request sent with turn on state.	Pass
Switch off wall outlet	1. Swipe sliding controller to turn off	-	HTTP request sent with turn off state.	HTTP request sent with turn off state.	Pass

Table 7.8 Test Case APP-6 - Set Countdown Timer

Test Case #	APP-6	**Test Case Title**	Set Countdown Timer	**Module**	Control
Test Case Overview	To check if user can set countdown timer to change wall outlet state.				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Set countdown timer to change wall outlet state	1. Scroll to select countdown hour, minute and second	-	HTTP request sent with correct countdown time.	HTTP request sent with correct countdown time.	Pass
	2. Tap save button	-			

Table 7.9 Test Case APP-7 - Set Alert Value

Test Case #	APP-7	**Test Case Title**	Set Alert Value	**Module**	Control
Test Case Overview	To check if user can set energy consumption alert value.				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Set daily alert value of a wall outlet	1. Tap edit button	-	HTTP request sent with correct parameters.	HTTP request sent with correct parameters.	Pass
	2. Enter warning value	5			
	3. Enter force stop value	10			
	4. Tap save button	-			
Set monthly alert value of a wall outlet	1. Tap edit button	-	HTTP request sent with correct parameters.	HTTP request sent with correct parameters.	Pass
	2. Enter warning value	50			
	3. Enter force stop value	60			
	4. Tap save button	-			

Table 7.10 Test Case APP-8 - View Energy Consumption Data

Test Case #	APP-8	**Test Case Title**	View Energy Consumption Data	**Module**	Monitoring
Test Case Overview	To check if user can inspect energy consumption data in different format.				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
View energy consumption data unit Wh.	1. Tap on the Energy tab.	-	HTTP request sent with energy parameter.	HTTP request sent with energy parameter.	Pass
View energy consumption data in unit RM.	1. Tap on the Price tab.	-	HTTP request sent with price parameter.	HTTP request sent with price parameter.	Pass
View energy consumption data in weekly.	1. Tap on the dropdown button.	-	HTTP request sent with weekly parameter.	HTTP request sent with weekly parameter.	Pass
	2. Tap on the Weekly selection.	-			
View energy consumption data in monthly.	1. Tap on the dropdown button.	-	HTTP request sent with monthly parameter.	HTTP request sent with monthly parameter.	Pass
	2. Tap on the Monthly selection.	-			

Table 7.11 Test Case APP-9 - Create New Property

Test Case #	APP-9	**Test Case Title**	Create New Property	**Module**	Property Management
Test Case Overview	To check if user can create new property				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Add a new property with valid property name.	1. Select property type	-	HTTP request sent.	HTTP request sent	Pass
	2. Enter property name	Sungai Long House			
	3. Tap save button	-			
Create new property with empty input field.	1. Tap save button	-	Error message display without sending HTTP request.	Error message display without sending HTTP request.	Pass

Table 7.12 Test Case APP-10 - Create New Room

Test Case #	APP-10	**Test Case Title**	Create New Room	**Module**	Property Management
Test Case Overview	To check if user can create new room				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Add a new room with valid room name.	1. Select property belong to	-	HTTP request sent.	HTTP request sent	Pass
	2. Enter room name	Living Room			
	3. Tap save button	-			
Create new room with empty input field.	1. Tap save button	-	Error message display without sending HTTP request.	Error message display without sending HTTP request.	Pass

Table 7.13 Test Case APP-11 - Create New Wall Outlet

Test Case #	APP-11	**Test Case Title**	Create New Wall Outlet	**Module**	Property Management
Test Case Overview	To check if user can create new wall outlet				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Add a new wall outlet with valid wall outlet name.	1. Scan QR code	Image of QR code	QR code analyse correctly and HTTP request sent.	QR code analyse correctly and HTTP request sent.	Pass
	2. Select property belong to	-			
	3. Select room belong to	-			
	4. Enter wall outlet name	TV Socket			
	5. Tap save button	-			
Create new wall outlet with empty input field.	1. Tap save button	-	Error message display without sending HTTP request.	Error message display without sending HTTP request.	Pass

7.3.2 APIs Unit Test

For the testing of the APIs, the Postman APIs development tool was used for this project. It allows testers to send HTTP requests to the API to be tested and then observe and analyse the responses. The testers executed nine unit tests for the selected route on the backend based on the test steps and test data provided in the test case. The actual and expected results of the tests were compared and documented. Table 7.14 until Table 7.22 depict the nine test cases of the APIs unit test.

Table 7.14 Test Case API-1 - Register

Test Case #	API-1	**Test Case Title**	Register	**Service**	Auth
Test Case Overview	To check if API can register a new account and send a verification email				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Register with valid email	1. Select POST method	-	New record inserted to SQL database and verification email sent to register email.	New record inserted to SQL database and verification email sent to register email.	Pass
	2. Enter username	gohyoushengabc@gmail.com			
	3. Enter email address	Shenggg2000			
	4. Enter password	12345678			
	5. Send HTTP request.	-			
Register with email that registered before.	1. Select POST method	-	API response with error and error	API response with error and error	Pass
	2. Enter email address	gohyoushengabc@gmail.com			
	3. Enter username	Shenggg2000			

	4. Enter password	12345678	message “email used”	message “email used”	
	5. Send HTTP request.	-			

Table 7.15 Test Case API-2 - Login

Test Case #	API-2	**Test Case Title**	Login	**Service**	Auth
Test Case Overview	To check if API can verify the authentication information with the database and return a token.				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Login with legitimate authentication information.	1. Select POST method	-	API response token without error	API response token without error	Pass
	2. Enter email address	gohyoushengabc@gmail.com			
	3. Enter password	12345678			
	4. Send HTTP request.	-			
Login with invalid authentication information.	1. Select POST method	-	API response with error with error message “login unsuccessful”	API response with error with error message “login unsuccessful”	Pass
	2. Enter email address	gohyoushengabc@gmail.com			
	3. Enter password	abcd			
	4. Send HTTP request.	-			

Table 7.16 Test Case API-3 - Create Property

Test Case #	API-3	**Test Case Title**	Create Property	**Service**	Property
Test Case Overview	To check if API can create a new property				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Add a new property with valid property information.	1. Select POST method	-	New record inserted to SQL database	New record inserted to SQL database	Pass
	2. Enter property name	Sungai Long House			
	3. Enter property type	RESIDENTIAL			
	4. Enter property floor plan data	[[1,140,210]]			
	5. Send HTTP request.	-			

Table 7.17 Test Case API-4 - Create Room

Test Case #	API-4	**Test Case Title**	Create Room	**Service**	Room
Test Case Overview	To check if API can create a new room				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Add a new room with valid room information.	1. Select POST method	-	New record inserted to SQL database	New record inserted to SQL database	Pass
	2. Enter room name	Living Room			
	3. Enter property id	1			
	4. Enter room floor plan data	{"points": [], "midPoints": [], "sockets": []}			
	5. Send HTTP request.	-			

Table 7.18 Test Case API-5 - Create Wall Outlet

Test Case #	API-5	**Test Case Title**	Create Wall Outlet	**Service**	Wall outlet
Test Case Overview	To check if API can create a new wall outlet				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Add a new wall outlet with valid wall outlet information.	1. Select POST method	-	New record inserted to SQL database	New record inserted to SQL database	Pass
	2. Enter wall outlet name	TV Socket			
	3. Enter room id	1			
	4. Enter wall outlet identifier	Socket 1			
	5. Send HTTP request.	-			

Table 7.19 Test Case API-6 - Read Wall Outlet Usage

Test Case #	API-6	**Test Case Title**	Read Wall Outlet Usage	**Service**	Wall outlet
Test Case Overview	To check if API can view wall outlet usage and the latest event				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Read wall outlet usage data	1. Select GET method	-	API response usage data and latest events without error	API response usage data and latest events without error	Pass
	2. Enter wall outlet id at the end of URL	1			
	3. Send HTTP request.	-			

Table 7.20 Test Case API-7 - Switch On Wall Outlet

Test Case #	API-7	**Test Case Title**	Switch On Wall Outlet	**Service**	Wall outlet
Test Case Overview	To check if API can change the wall outlet state to on				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Change wall outlet state to on	1. Select PUT method	-	API response without error and state updated to 1 in real-time database	API response without error and state updated to 1 in real-time database	Pass
	2. Enter wall outlet id at the end of URL	1			
	3. Send HTTP request.	-			

Table 7.21 Test Case API-8 - Create Notification

Test Case #	API-8	**Test Case Title**	Create notification	**Service**	Notification
Test Case Overview	To check if API can create notifications and successfully push notifications to mobile device				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Insert a new notification	1. Select POST method	-	API response without error, new record inserted to SQL database and emulator received notice	API response without error, new record inserted to SQL database and emulator received notice	Pass
	2. Enter wall outlet id	1			
	3. Enter notification type	WARNING			
	4. Enter title	Notice 1			
	5. Enter content	This is a testing notification			
	6. Send HTTP request.	-			

Table 7.22 Test Case API-9 - Create Scheduled Task

Test Case #	API-9	**Test Case Title**	Create Scheduled Task	**Service**	Tasks Schedule
Test Case Overview	To check if API can create a scheduled task and successfully executed the task				
Test Detail	**Test Instruction**	**Test Data**	**Expected Result**	**Actual Result**	**Condition**
Insert a new scheduled task	1. Select POST method	-	API response without error, new record inserted to SQL database and state updated to 1 in real-time database after action_ datetime	API response without error, new record inserted to SQL database and state updated to 1 in real-time database after action_ datetime	Pass
	2. Enter wall outlet id	1			
	3. Enter wall outlet identifier	Socket 1			
	4. Enter action	ON			
	5. Enter action_datetime	2023-04-01 15:11:40			
	6. Enter is_schdule	0			
	7. Send HTTP request.	-			

7.4 User Acceptance Test

User Acceptance Testing (UAT) is the final phase of software testing before the software is released to the end users. UAT aims to ensure that the software is ready for release and meets the end user's expectations. Test scenarios and cases were developed to simulate real-life situations users might encounter. These test scenarios and cases cover all the system's critical functionalities and use cases. Table 7.23 presents the user acceptance test template used to guide the user to complete all nine actions with the blank space to record test results and user comments.

Table 7.23 User Acceptance Test Template

ID	Module	Test Scenario	Result	Comment
1	Profile	1. Register a new account 2. Verify email 3. Log in to the account		
2	Property Management	1. Create a new property 2. Create a new room on the property 2. Create a new wall outlet in the room		
3	Property Management	1. Edit the room floor plan. 2. Edit the wall outlet position in the room		
4	Control	1. Switch on the wall outlet 2. Switch off the wall outlet		
5	Control	1. Set a 5-minute countdown timer to switch on the wall outlet		
6	Control	1. Set a schedule that switches on the wall outlet at 6.00 a.m. and switches off the wall outlet at 10.00 p.m.		

7	Monitor	1. View monthly energy consumption of a wall outlet in unit RM		
8	Voice Recognition	1. Use voice command to switch off the wall outlet.		
9	Profile	1. Logout the account		

Test results for the user acceptance test were included in APPENDIX A: Result of User Acceptance Test. Upon completion of the user acceptance test, there is a strong belief that the project has adequately addressed the user's needs and requirements. Feedback given by users is useful for creating additional features or making enhancements that would benefit the users.

7.5 Conclusion

In conclusion, through rigorous software testing, many potential errors have been identified and resolved. By conducting thorough testing, the overall quality and reliability of the system was improved. The testing process has proven to be an essential step in the software development lifecycle, helping to ensure that the final product meets the required standards and specifications.

CHAPTER 8

CONCLUSION AND RECOMMENDATION

8.1 Conclusion

This project has successfully developed a smart energy monitoring application for Android and successfully addressed several critical problems. Firstly, the issue of being restricted to desktop or web-based applications has been effectively resolved. The project has taken into consideration the difficulties faced by elderly individuals when using complex applications. With the integration of a voice recognition, the app becomes more accessible and user-friendly for this demographic. The project also addressed the problem of users being unaware of the state of a wall outlet where users can receive real-time updates and notifications about the status of their wall outlets.

Furthermore, by using this system, the user can improve energy efficiency by facilitating the use of smart wall outlets. The following project's objectives were achieved.

- To develop a mobile application that allows remote control of the status (ON/OFF) and monitors the real-time power consumption of smart wall outlets.
- To implement speech recognition for voice command.
- To notify the user when a wall outlet's energy consumption has exceeded the predetermined value.

The initial objective has been fulfilled since the development of the mobile application has been accomplished, enabling the user to digitally manage each smart wall outlet within the property. The user does not need to be physically present at the property to know and change the working state of the wall outlet. The application also allows for more options to change the working state of the wall outlets, for example, timing and scheduling. The application also successfully informed the user about the statistical energy consumption data through visual charts.

The second goal has been achieved by implementing a voice recognition module. This module converts the user’s voice into commands and executes them. This feature increases the usability of the application.

Finally, practising a push notification module has accomplished the third goal. This push notification module effectively informs users of the latest information about their wall outlet. This allows the user to know if a wall outlet has exceeded the energy consumption alert level without having to open the application all the time.

8.2 Limitations

Despite the accomplishment of the project's objectives, there remain some limitations in the project that could be enhanced.

The first limitation is that the functionality of the floor plan editor is limited. The floor plan editor only allows one-floor level for a property, and it is impossible to add stairs, balconies, windows or doors, which are the essential components of a property. In addition, there is no undo function, so the user cannot revert to the previous status of the floor plan if they make some mistakes.

The second limitation is the long response time of some features in the application. This limitation was observed in the QR code scanning function and the voice recognition module. The problems that led to this limitation were the algorithm's complexity and the third-party libraries' long loading time. Users usually have to wait for several seconds to continue the operation, and this limitation needs to be improved to increase user satisfaction.

The third limitation is that the voice recognition module needs to be smarter. The currently implemented voice recognition function requires the user to actively press a button to allow the module to listen to the user's speech. In addition, the voice recognition module does not automatically end the recording after the user stops speaking but requires the user to click the button again actively. This greatly increases the number of user actions. Moreover, the voice recognition process is not synchronized with the recording but is analyzed only after the user finishes the sentence. This leads to the inability of this module to respond quickly.

8.3 Recommendation for Future Work

The following are six recommendations for future work that could be explored for this project.

Improve performance.

Optimisations need to be implemented in response to some features' limitation of long response times. Algorithm optimisation and the replacement of more efficient third-party libraries are the possible solutions.

Floor plan editor

Since the floor plan editor only provides simple features at the moment, there is a possibility of adding new features in the future. Some of the features that can be added are undo and floor level. The undo function can improve the error tolerance rate when editing floor plans. The floor function can meet more use cases and even improve user experience and realism with the help of 3D modelling.

Reduce user actions in voice recognition.

Several optimisations can be made to the voice recognition module to reduce the number of user actions. The first is to add a voice wakeup function, which can be done by pre-setting a wakeup keyword. When the user speaks the keyword, voice recognition will start immediately. In addition, when the user finishes the command narration and enters a period of silence, the voice recognition function needs to pause automatically and start the analysis.

Custom time range in displaying energy consumption data.

For the presentation of energy consumption data, a custom range of time is a feature that can be added. Currently, the application only supports monthly and weekly, and the user cannot determine on viewing a specific time range, which would limit the user's use scenarios. This feature would enable the data to be presented in a customized manner that aligns with the user's specific needs, making it more relevant and useful for the user.

Compatible with more smart appliances

For a property, the energy-consuming appliances are not only used through a wall outlet but can also be powered directly by a wire, just like a ceiling fan. However, currently, the application can only add a smart wall outlet, but not other smart appliances. To make the energy consumption statistics more integrated and complete, adding these appliances to the application for monitoring is essential. In addition to monitoring, the control of these new smart appliances also needs to be implemented to enrich the application's usefulness.

Utilize energy consumption data in prediction and prevent energy wastage.

Energy consumption data is a valuable asset that can be utilised to analyse and predict future energy consumption. In addition, the data analysis can also extract valuable information to help users improve their habits of using wall outlets and reduce energy waste.

REFERENCES

AgingInPlace. 2022. Voice Recognition Innovation And The Implications For Seniors. [online] Available at: <https://aginginplace.org/voice-recognition-innovation-and-the-implications-for-seniors/> [Accessed 1 August 2022].

Barker, S., Mishra, A., Irwin, D., Shenoy, P. and Albrecht, J., 2012. Smartcap: Flattening peak electricity demand in smart homes. *2012 IEEE International Conference on Pervasive Computing and Communications*, pp. 67-75. http://dx.doi.org/10.1109/PerCom.2012.6199851.

Beck, K., 2001. Manifesto for Agile Software Development. [online] Agilemanifesto.org. Available at: <http://agilemanifesto.org/> [Accessed 30 August 2022].

Ben-Hur, A., Ong, C., Sonnenburg, S., Schölkopf, B. and Rätsch, G., 2008. Support Vector Machines and Kernels for Computational Biology. *PLoS Computational Biology*, 4(10), p.e1000173.

Climate Watch Historical GHG Emissions. 2022. Washington, DC: World Resources Institute. [online] Available at: <https://www.climatewatchdata.org/ghg-emissions/> [Accessed 1 August 2022].

Gadhavi, M., 2022. 9 SDLC Methodologies: Understand and Choose the Best SDLC Methodology for Your Project. [online] Radixweb. Available at: <https://radixweb.com/blog/best-software-development-life-cycle-methodologies-in-2019> [Accessed 30 August 2022].

Gerasymov, O., 2022. Using Machine Learning for Time Series Forecasting Project. [online] CodeIT. Available at: <https://codeit.us/blog/machine-learning-time-series-forecasting> [Accessed 29 August 2022].

Goddard, W., 2022. Speech Recognition Algorithm. [online] IT Chronicles. Available at: <https://itchronicles.com/artificial-intelligence/speech-recognition-algorithms/> [Accessed 27 August 2022].

Goldberg, S., 2018. Preventing IoT data waste with the intelligent edge. [online] Techtarget. Available at: <https://www.techtarget.com/iotagenda/blog/IoT-Agenda/Preventing-IoT-data-waste-with-the-intelligent-edge> [Accessed 1 August 2022].

Half, R., 2021. 6 Basic SDLC Methodologies: Which One is Best?. [online] Roberthalf.com. Available at: <https://www.roberthalf.com/blog/salaries-and-skills/6-basic-sdlc-methodologies-which-one-is-best#One> [Accessed 30 August 2022].

Hayes, A., 2022. Time Series Definition. [online] Investopedia. Available at: <https://www.investopedia.com/terms/t/timeseries.asp> [Accessed 28 August 2022].

How Does Speech Recognition Work? Learn about Speech to Text, Voice Recognition and Speech Synthesis. 2022. [video] Directed by A. Vishwanath. United States: Acadaimy.

i2tutorials. 2019. Understanding Hidden Markov Model for Speech Recognition | i2tutorials. [online] Available at: <https://www.i2tutorials.com/understanding-hidden-markov-model-for-speech-recognition> [Accessed 27 August 2022].

iTrobes. 2021. What are the advantages and disadvantages of web applications?. [online] Available at: <https://www.itrobes.com/what-are-the-advantages-and-disadvantages-of-web-applications/> [Accessed 1 August 2022].

Karim, S. and Alwi, S., 2012. Electricity load forecasting in UTP using moving averages and exponential smoothing techniques. *Applied Mathematical Sciences*, 7(80), pp. 4003-4014. http://dx.doi.org/10.12988/ams.2013.33149.

Karunathilake, S. and Nagahamulla, H., 2017. Artificial neural networks for daily electricity demand prediction of Sri Lanka. *2017 Seventeenth International Conference on Advances in ICT for Emerging Regions (ICTer).*

Lee, R., 2021. Has TNB’s Smart Meter worked? - SoyaCincau. [online] SoyaCincau. Available at: <https://soyacincau.com/2021/02/11/tnb-tenaga-nasional-smart-meter-opinion/.> [Accessed 25 July 2022].

Mateo, C., Almagro, F., Yi, W.J. and Saniie, J., 2021. Design Flow and Implementation of an IoT Smart Power Socket. *2021 IEEE International Conference on Electro Information Technology (EIT)*, pp. 151-156. http://dx.doi.org/10.1109/EIT51626.2021.9491841.

Nassif, A., Shahin, I., Attili, I., Azzeh, M. and Shaalan, K., 2019. Speech Recognition Using Deep Neural Networks: A Systematic Review. *IEEE Access, 7,* pp.19143-19165. http://dx.doi.org/10.1109/ACCESS.2019.2896880

Ouyang, T. and Zheng, H., 2022. Analysis of the Reasons Why It Is Difficult for the Elderly to Learn to Use Smart Phones. *Advances in Psychology*, 12(05), pp.1900-1906.

Pykes, K., 2020. Part Of Speech Tagging for Beginners. [online] Medium. Available at: <https://towardsdatascience.com/part-of-speech-tagging-for-beginners-3a0754b2ebba > [Accessed 27 August 2022].

Rainie, L. and Zickuhr, K., 2015. Chapter 1: Always on Connectivity. [online] Pew Research Center: Internet, Science & Tech. Available at: <https://www.pewresearch.org/internet/2015/08/26/chapter-1-always-on-connectivity/> [Accessed 1 August 2022].

Shapi, M., Ramli, N. and Awalin, L., 2021. Energy consumption prediction by using machine learning for smart building: Case study in Malaysia. Developments in the Built Environment, 5, p.100037.

Software Testing Help. 2022. SDLC (Software Development Life Cycle) Phases, Process, Models. [online] Available at: <https://www.softwaretestinghelp.com/software-development-life-cycle-sdlc/#1_Waterfall_Model> [Accessed 31 August 2022].

Time Series Forecasting with Machine Learning. 2021. [video] United States: CodeEmporium.

Trunkett, O., 2020. SDLC Methodologies: From Waterfall to Agile | Virtasant. [online] Virtasant.com. Available at: <https://www.virtasant.com/blog/sdlc-methodologies> [Accessed 30 August 2022].

United Nations Framework Convention on Climate Change. 2015. Intented Nationality Determined Contribution of The Government of Malaysia. [online] Available at: <https://www4.unfccc.int/sites/submissions/INDC/Published%20Documents/Malaysia/1/INDC%20Malaysia%20Final%2027%20November%202015%20Revised%20Final%20UNFCCC.pdf> [Accessed 1 August 2022].

Valgaev, O., Kupzog, F. and Schmeck, H., 2016. Low-voltage power demand forecasting using K-nearest neighbors approach. *2016 IEEE Innovative Smart Grid Technologies - Asia (ISGT-Asia).*

What is NLP? Learn Natural Language Processing in Artificial Intelligence. 2022. [video] Directed by A. Vishwanath. United States: Acadaimy.

What is Time Series Data. 2019. [video] Directed by A. LaBarr. United States.

Xiangyu, Z., Qingqing, M., Tao, L., Lei, Z. and Jianquan, Z., 2019. Linear Regression Electricity Prediction Method Based on Clustering of Electric Characteristics. *Proceedings of the 2nd International Conference on Big Data Technologies - ICBDT2019,* pp.171-176. https://doi.org/10.1145/3358528.3358564

Yoon, B., 2009. Hidden Markov Models and their Applications in Biological Sequence Analysis. Current Genomics, 10(6), pp.402-415. http://dx.doi.org/10.2174/138920209789177575

Yusof, M., Romli, N. and Yusof, M., 2014. Design for Elderly Friendly: Mobile Phone Application and Design that Suitable for Elderly. International Journal of Computer Applications, 95(3), pp.28-31.

APPENDICIES

Appendix A Result of User Acceptance Test

Tester #		1		
Testing Date		25 April 2023		
ID	**Module**	**Test Scenario**	**Result**	**Comment**
1	Profile	1. Register a new account 2. Verify email 3. Log in to the account	Pass	
2	Property Management	1. Create a new property 2. Create a new room on the property 2. Create a new wall outlet in the room	Pass	
3	Property Management	1. Edit the room floor plan. 2. Edit the wall outlet position in the room	Pass	
4	Control	1. Switch on the wall outlet 2. Switch off the wall outlet	Pass	
5	Control	1. Set a 5-minute countdown timer to switch on the wall outlet	Pass	
6	Control	1. Set a schedule that switches on the wall outlet at 6.00 a.m. and switches off the wall outlet at 10.00 p.m.	Pass	
7	Monitor	1. View monthly energy consumption of a wall outlet in unit RM	Pass	
8	Voice Recognition	1. Use voice command to switch off the wall outlet.	Pass	
9	Profile	1. Logout the account	Pass	

Tester #		2		
Testing Date		25 April 2023		
ID	**Module**	**Test Scenario**	**Result**	**Comment**
1	Profile	1. Register a new account 2. Verify email 3. Log in to the account	Pass	
2	Property Management	1. Create a new property 2. Create a new room on the property 2. Create a new wall outlet in the room	Pass	
3	Property Management	1. Edit the room floor plan. 2. Edit the wall outlet position in the room	Pass	
4	Control	1. Switch on the wall outlet 2. Switch off the wall outlet	Pass	
5	Control	1. Set a 5-minute countdown timer to switch on the wall outlet	Pass	
6	Control	1. Set a schedule that switches on the wall outlet at 6.00 a.m. and switches off the wall outlet at 10.00 p.m.	Pass	
7	Monitor	1. View monthly energy consumption of a wall outlet in unit RM	Pass	
8	Voice Recognition	1. Use voice command to switch off the wall outlet.	Pass	
9	Profile	1. Logout the account	Pass	

Tester #		3		
Testing Date		25 April 2023		
ID	**Module**	**Test Scenario**	**Result**	**Comment**

1	Profile	1. Register a new account 2. Verify email 3. Log in to the account	Pass	
2	Property Management	1. Create a new property 2. Create a new room on the property 2. Create a new wall outlet in the room	Pass	
3	Property Management	1. Edit the room floor plan. 2. Edit the wall outlet position in the room	Pass	
4	Control	1. Switch on the wall outlet 2. Switch off the wall outlet	Pass	
5	Control	1. Set a 5-minute countdown timer to switch on the wall outlet	Pass	
6	Control	1. Set a schedule that switches on the wall outlet at 6.00 a.m. and switches off the wall outlet at 10.00 p.m.	Pass	
7	Monitor	1. View monthly energy consumption of a wall outlet in unit RM	Pass	
8	Voice Recognition	1. Use voice command to switch off the wall outlet.	Pass	
9	Profile	1. Logout the account	Pass	

Tester #		4		
Testing Date		25 April 2023		
ID	**Module**	**Test Scenario**	**Result**	**Comment**
1	Profile	1. Register a new account 2. Verify email 3. Log in to the account	Pass	

2	Property Management	1. Create a new property 2. Create a new room on the property 2. Create a new wall outlet in the room	Pass	
3	Property Management	1. Edit the room floor plan. 2. Edit the wall outlet position in the room	Pass	
4	Control	1. Switch on the wall outlet 2. Switch off the wall outlet	Pass	
5	Control	1. Set a 5-minute countdown timer to switch on the wall outlet	Pass	
6	Control	1. Set a schedule that switches on the wall outlet at 6.00 a.m. and switches off the wall outlet at 10.00 p.m.	Pass	
7	Monitor	1. View monthly energy consumption of a wall outlet in unit RM	Pass	
8	Voice Recognition	1. Use voice command to switch off the wall outlet.	Pass	
9	Profile	1. Logout the account	Pass	

Tester #		5		
Testing Date		25 April 2023		
ID	**Module**	**Test Scenario**	**Result**	**Comment**
1	Profile	1. Register a new account 2. Verify email 3. Log in to the account	Pass	
2	Property Management	1. Create a new property 2. Create a new room on the property	Pass	

		2. Create a new wall outlet in the room		
3	Property Management	1. Edit the room floor plan. 2. Edit the wall outlet position in the room	Pass	
4	Control	1. Switch on the wall outlet 2. Switch off the wall outlet	Pass	
5	Control	1. Set a 5-minute countdown timer to switch on the wall outlet	Pass	
6	Control	1. Set a schedule that switches on the wall outlet at 6.00 a.m. and switches off the wall outlet at 10.00 p.m.	Pass	
7	Monitor	1. View monthly energy consumption of a wall outlet in unit RM	Pass	
8	Voice Recognition	1. Use voice command to switch off the wall outlet.	Pass	
9	Profile	1. Logout the account	Pass	