National College of Ireland

BSHCSD4

Software Development

Academic Year i.e. 2023/2024

Nimrah Memon

x20738271

x20738271@student.ncirl.ie

Project Repo https://github.com/Nimrah12/FinalYearProjectSwitchifyApp- .git

Switchify (Mobile APP)

IOT Smart Home Automation System

Technical Report

Contents

Executive Summary

The proposed home automation system which constitutes an ESP32 microcontroller coupled with a Mobile application serves the purpose of giving users a wide range of options that allow them to control or monitor smart devices relevant to their homes from wherever and whenever they want. The Mobile app enables complex managed devices, and it improves Firebase to show actual sync information in real time. The ESP32 is the main controller and manager of the whole system, establishing a safe and reliable Wi-Fi connection for effective communication.

Some main features such as a pairing algorithm based on secure connection, device controlling for lights switch on and other gears as well as the automation of lighting scene modes are involved. Our app provides easy-to-use Flutter interfaces, making it possible for the software to run on both mobile platforms.

The program comprises of Dart programming for the mobile app, Arduino IDE for ESP32 firmware, and Firebase integration for cloud-based storage. The GUI features a menu bar with options that allow the user to set up time schedules, modify scenes, and give reviews.

The forthcoming improvements stand through the introduction of the energy efficient algorithms and thus the data logging for in-depth analysis. The system presents a scalable and secure solution for smart home automation, which increases user convenience and reduces energy consumption.

1.0 Introduction

1.1. Background

The motivation behind this home automation system project came from the increasing need for smart and efficient solutions in modern living spaces. With the rapid advancements in technology, the desire to create a seamless and user-friendly environment within homes has become important. The integration of an Mobile application with an ESP micro controller offers a versatile and accessible platform for controlling various aspects of a home, ranging from lighting and home appliances to energy management. The project addresses the growing demand for home automation solutions that prioritize user convenience, energy efficiency and accessibility.

One of the primary motivations was the opportunity to create a solution that improves accessibility for individuals with disabilities. It is a well-known fact that technology can improve they quality of life for people with disabilities. And this project aligns with that mission. By developing an Mobile app that allows users to remote home control. We can empower disabled people to independently manage their living environment.

Lastly this project represents an opportunity to explore how technology can seamlessly integrate into everyday life. This project allows me to showcase how technology can improve and simplify daily task. Home automation system is becoming increasingly general, whether it's turning lights on and off, adjusting their brightness, we are making technology a part of user's daily routines.

1.2. Aims

The goal of this project is to make a smart home system that can be controlled using a phone. It will use the ESP32 microcontroller. The main goals are:

- Developing an Android Application: Set up the user-friendly mobile application to work as the central control hub for the home automation system.
- Integration with ESP32 Microcontroller: Set the microcontroller ESP32 in such a way that communication would be possible with the mobile application for wireless control of a wide variety of smart home devices.
- Automate home tasks like lights, security, and energy usage.
- Make sure the system is easy for anyone to use, regardless of their tech skills.
- Add features to save energy and lower electricity costs for users. This is to make the system sustainable and affordable.

By meeting these goals, the project aims to create a smart home system that improves quality of life, saves energy, and meets the needs of homeowners.

1.3. Technology

The chosen technology for this home system to work well together. I pick:

- **Making an app for Android:** We use Android Studio and Dart with flutter to make a strong and easy app. People can use the app to talk to and run the home gadgets.
- **ESP32/8266 chip**: We use this as the main hub to run the devices. ESP32 uses Wi-Fi to link the app and the gadgets.
- **Wi-Fi Link:** We set up a Wi-Fi web using the ESP32 to let the app talk to the gadgets. Wi-Fi helps take out tough wires.
- **Firebase**: We use Firebase as an online base to save and set how users like things, how the gadgets work. This lets the app and the home gear work in real time.
- **Integration of Sensors**: Incorporating sensors, such as motion sensors and ambient light sensors, to enhance automation capabilities. These sensors will provide data to the Esp, allowing the system to respond intelligently to environmental changes.
- **Customized Firmware for ESP32:** Developing custom firmware for the ESP32 microcontroller to interpret signals from the Android application, control

connected devices, and manage the overall functioning of the home automation system.

1.4. Structure

- **Introduction: Sets the background for the project, stressing the nature of the problem, motivation, and what the main aim of the project is - the creation of IOT smart home automation system**

- Technology: The project used the Mobile application development, ESP32 microcontroller, Wi-Fi connectivity, Firebase database, and sensor integration technologies.

- Objectives: Describe the particular project goals and objectives including the development of the Mobile application, ESP32 microcontroller Liaison, and the possibility, of user accessibility, energy savings, and customization.

- State of the Art: Perform comparative research on the existing home automation systems and introduce the specific features and innovations that help to stand out the proposed project in a unique way.
- Technical Approach: It explains the blueprint for the project development process, which includes the creation of the mobile application, UI design, Firebase database setup, and the connection of the ESP32 microcontroller using an agile methodology.
- Technical Details: The educating part is about the implementation language, fundamental libraries, and important approaches or algorithms that are considered in the project.
- Special Resources Required: Best exemplifies these resources in a way that the implementation of a home automation system can be accomplished using hardware, software and a lot of tools.
- Project Plan: The project plan provides an in-depth project plan which includes a Gantt chart with implementation steps, timelines, and milestones.
- Functional Requirements: first of all overwrite the functional requirements and then create a use case diagram with the use cases listed in it and their flows.
- Non-functional Requirements: focus on the nonfunctional aspects or functional requirements of this home automation system.
- Design: it is also important to attach the architecture diagram of the design.
- Implementation, prototype: show a screenshot of the program and the functions for the mobile app as well as the ESP32 controller infrastructure in the video.
- GUI design: the application screenshot and buttons, screenshots of the application, along with wireframes wrapping the application.
- Further Development/ research: The project should have an innovation component. scenarios like in case NFC functionality or user stories.
- Testing: Explains an approach to testing consisting of system tests, integration tests and aspects of how feedback from actual technical information will be used.
- User Profiles: Defines different user profiles and conditions applicable to various user types such as those for elderly people, those with special needs, and those of affordability, to name a few.
- Reflective Journal: It is a retrospective view of the project process, the progress, the challenges and the adjustments made during the months of the project.
- Conclusion: Emphasizes the critical points of the document, expresses their great importance and the prospects of the automated home system project.
- Appendices: all the stuff which it considers to be the Extra ideas such as proposal.
- This structured document presents a comprehensive overview of the home automation system project within which various technological aspects, project objectives, development approaches, testing strategies, ethical considerations and innovative features are included. Every section of the proposal adds up to the comprehensive comprehension of the specific

areas of the given project: development, implementation, and the basis of its effect on society.

2.0 System

2.1. Requirements

2.1.1. Functional Requirements

Mobile App Connectivity (Requirement 1):

Description: The mobile application should ensure a secure and flawless link with the ESP32 microcontroller so as to make the control of the device as easy as possible. Priority: High.

Lighting Control (Requirement 2):

Description: The system must facilitate the users to set the brightness and on/off state of the smart lights through the mobile application. **Priority**: Medium.

Automated Lighting Scenes (Requirement 3):

Description: To achieve this, the system should be software enabled in enabling rendering of automated lighting scenarios hence the user is allowed to create custom lighting settings for different instances. Consumers can choose [lighting] Scene settings as [example] the daytime, evening and bedtime modes. g. "'Night Light,' 'Dim Light,' and so forth are the available options, and you can turn them on through the mobile application. **Priority**: High.

Remote Access and Control (Requirement 4):

Description The system must be designed to facilitate user intervention without physical presence. The users should be able to access the home automation devices remotely from wherever with internet connection. **Priority**: High.

Energy Management (Requirement 5):

Description: The system must provide users with the ability to monitor and manage energy consumption of connected devices through the mobile application. Users can view real-time energy usage data and set preferences for energy-efficient operation. **Priority**: Medium.

Multi-User Account Management (Requirement 6):

Description: The system must support the management of multiple user accounts with varying access levels for shared smart homes. Homeowners can create, modify, and delete user accounts, assigning specific access permissions. **Priority**: High.

User Authentication and Authorization (Requirement 7):

Description: The system must authenticate and authorize users to ensure secure access to the mobile application. Users must enter valid credentials for authentication, gaining access to authorized features. **Priority**: High.

2.1.2. Use Case Diagram for Whole Smart Home Automation System

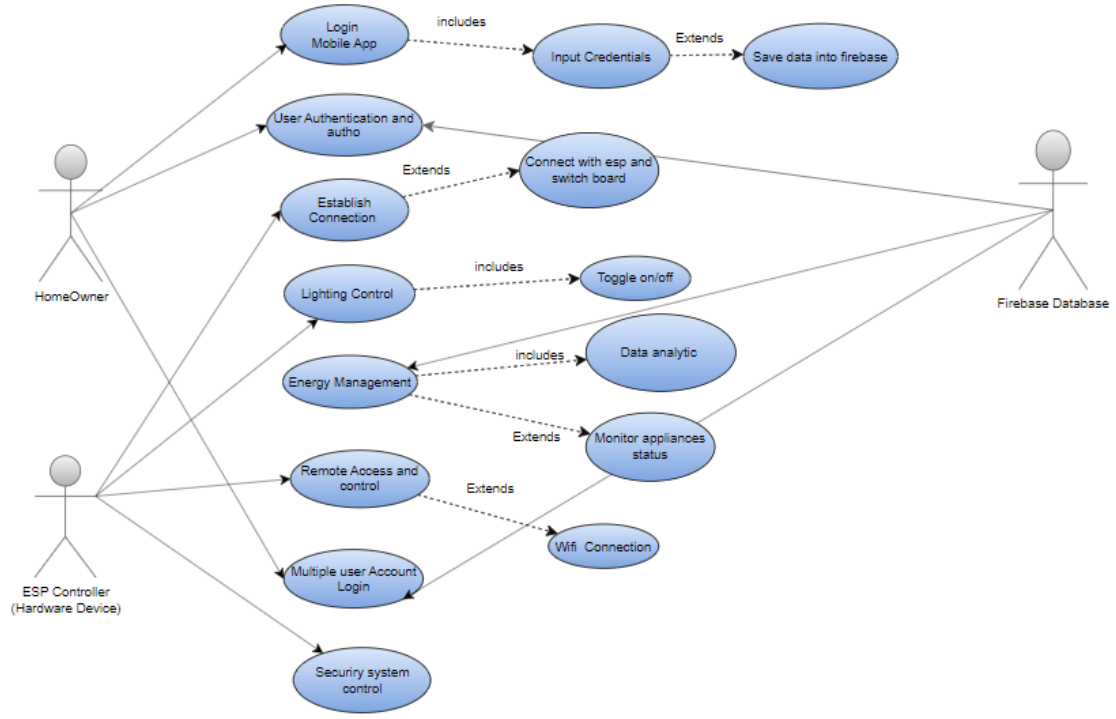


Figure 1: Use Case Diagram for Home Automation System

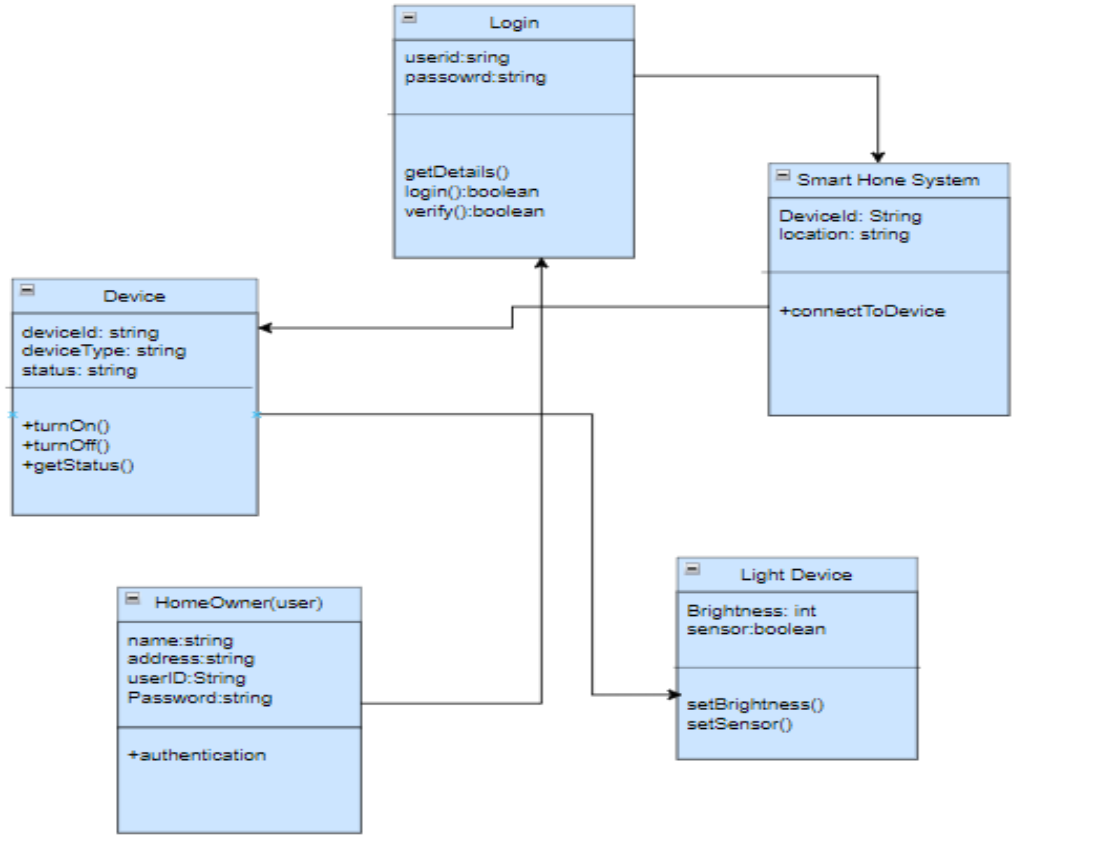


Figure 2: Class Diagram for Home Automation System

2.1.2.1. Requirement 1: Mobile App Connectivity

Use Case ID:	**UC-1**
Use Case Name:	**Mobile App Connectivity**
Use Case Description	The mobile application must establish a secure and reliable connection with the ESP32 microcontroller for seamless control.
Scope	**The scope of this use case is to ensure that users can control smart home devices through the mobile application.**
Description:	This use case describes the process of establishing and maintaining a connection between the Android application and the ESP32 microcontroller.
Pre-conditions:	The mobile app is installed and the ESP32 microcontroller is powered on.
Post-Conditions:	The mobile app is connected to the selected ESP32 microcontroller, enabling device control.
Activation	This use case starts when a user launches the mobile application and attempts to connect to the smart home system.
Actors	**Homeowner, ESP Controller**
Main Flow:	The mobile app identifies available ESP32 microcontrollers. 1. The user selects the desired ESP32 microcontroller for connection. 2. The mobile app sends a connection request to the selected ESP32 microcontroller. 3. The ESP32 microcontroller authenticates the request and establishes a secure connection.
Alternate Flows:	A1 : Connection Failure 1. The mobile app displays an error message. 2. The user can retry the connection process.
Exceptions:	E1 : Authentication Failure • The ESP32 microcontroller denies the connection request. • The mobile app prompts the user to re-enter credentials. • The use case continues at position 3 of the main flow.
Priority:	**High**
Termination	The system presents the user with control options for smart home devices.

Use Case Diagram

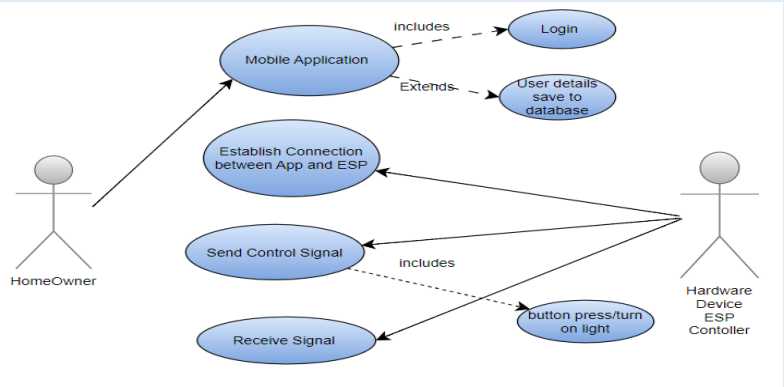


Figure 3: Use Case Diagram: Mobile App Connectivity

2.1.2.2. Requirement 2: Lighting Control

Use Case ID:	**UC-2**
Use Case Name:	**Lighting Control**
Use Case Description:	The system must allow users to control the brightness and on/off state of smart lights through the mobile application
Scope:	This use case concerns to the control of smart lights within the home automation system.
Description:	The user can adjust the brightness and turn on/off smart lights using the mobile application.
Pre-conditions:	The system is in initialisation mode
Post-Conditions:	The system goes into a wait state
Activation	This use case starts when a user initiates a device control command through the mobile app.
Actors	**Homeowner, ESP Controller**
Main Flow:	1. The system identifies the selected device for control. 2. The user issues a command to turn the device on or off (See A1). 3. The system processes the command and activates or deactivates the device. 4. The user receives feedback on the success of the command.
Alternate Flows:	A1 : Retry Command • If the system encounters an issue processing the command. • The user retries the command. • The use case continues at position 3 of the main flow
Exceptions:	E1 : Communication Error 1. If there is a communication error between the mobile app and the ESP32. 2. The system notifies the user of the error. 3. The use case continues at position 4 of the main flow.
Priority:	**Medium**
Termination:	The system presents the next available command for the user
Use Case Diagram	

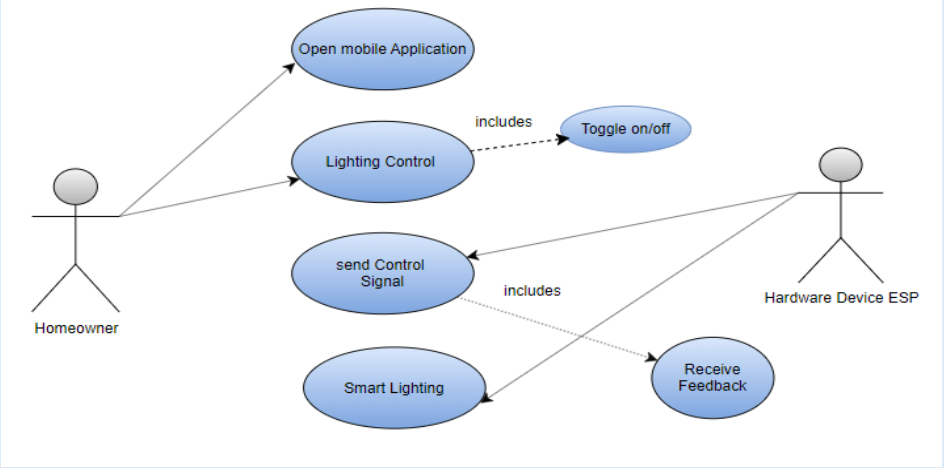


Figure 4: Use Case Diagram: Lighting Control

2.1.2.3. Requirement 3: Automated Lighting Scenes

Use Case ID:	**UC-3**
Use Case Name:	**Automated Lighting Scenes**
Use Case Description:	The system should support the creation and execution of automated lighting scenes, allowing users to define preset lighting configurations for different scenarios. Priority: High.
Scope:	This use case pertains to the creation and execution of automated lighting scenes within the home automation system.
Description:	Users can define lighting scenes, such as “Night Light" or "Dim Light" and activate them through the mobile application.
Pre-conditions:	The system has pre-defined lighting scenes
Post-Conditions:	The system goes into a wait state
Activation	This use case starts when a user (Actor) initiates the activation of a lighting scene through the mobile app.
Actors	**Homeowner, ESP Controller**
Main Flow:	1. The user selects a pre-defined lighting scene to activate. 2. The system processes the request and adjusts connected lights accordingly. 3. The user receives feedback on the successful activation of the scene.
Exceptions:	E1 : Scene not available 1. If the selected scene is not available or recognized. 2. The system notifies the user of the error. 3. The use case continues at position 4 of the main flow.
Priority:	**Medium**
Termination:	The system presents the next available command for the user
Use Case Diagram	

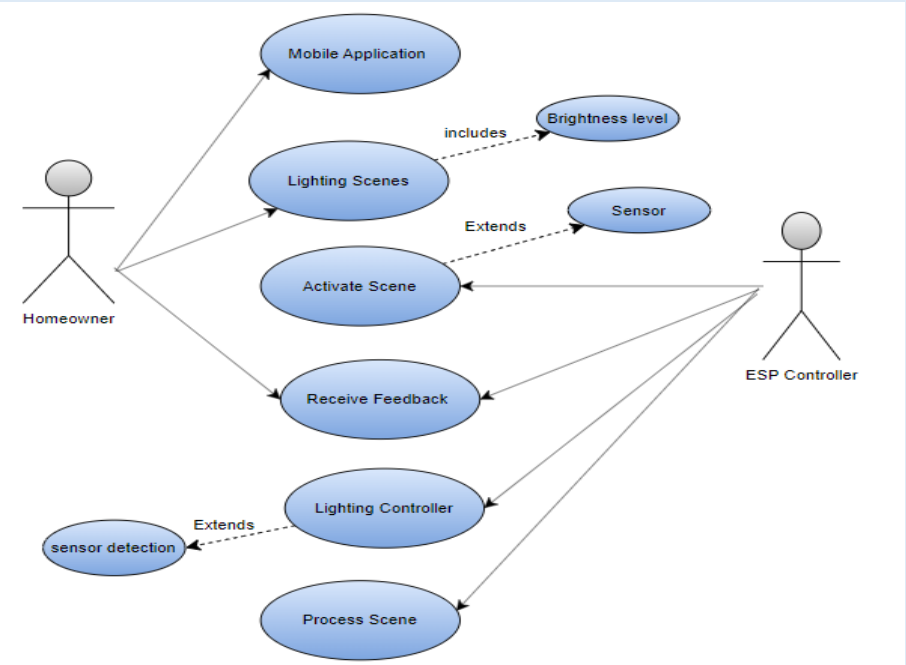


Figure 5: Use Case Diagram: Automated Lighting Scene

2.1.2.4. Requirement 4: Remote Access

Use Case ID:	**UC-4**
Use Case Name:	**Remote Access**
Use Case Description:	The system must provide users with remote access to the home automation features, allowing them to control devices and monitor the status of their smart home from anywhere with an internet connection.
Scope:	This use case focuses on enabling users to access and control their smart home devices remotely.
Description:	Users should be able to remotely log in to the mobile application and control connected devices, ensuring accessibility and control even when not at home.
Pre-conditions:	The mobile application is connected to the internet.
Post-Conditions:	The system goes into a wait state
Activation	This use case starts when a user initiates remote control commands from the mobile app.
Actors	**Homeowner, ESP Controller(hardware Device)**
Main Flow:	1. The user opens the mobile app remotely. 2. The system establishes a secure connection to the home automation system. 3. The user issues control commands for connected devices. 4. The system processes the commands and activates or deactivates devices accordingly.
Exception Flow	E1: Connection Error 1. If there is an issue establishing a secure connection. 2. The system notifies the user of the error. 3. The use case continues at position 4 of the main flow.
Priority:	**High**
Termination:	The system presents the next available command for the user.

Use Case Diagram

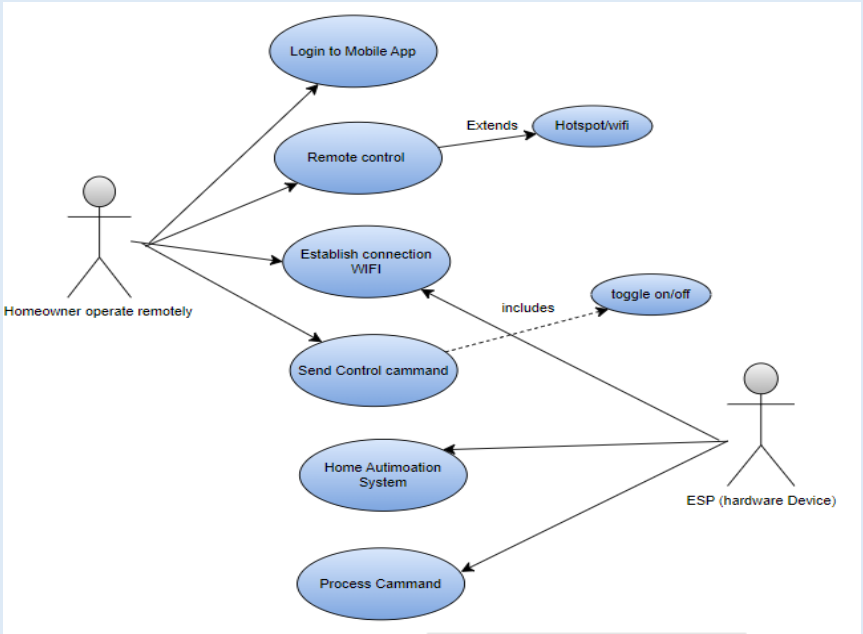


Figure 6: Use Case Diagram: Remote Access

2.1.2.5. Requirement 5: Energy Management

Use Case ID:	**UC-5**
Use Case Name:	**Energy Management**
Use Case Description:	The system must provide users with the ability to monitor and manage energy consumption of connected devices through the mobile application.
Scope:	This use case focuses on the monitoring and management of energy consumption within the home automation system.
Description:	Users can view real-time energy usage data and set preferences for energy-efficient operation of connected devices.
Pre-conditions:	The system is in operation, and energy monitoring is enabled.
Post-Conditions:	The system goes into a wait state, continuously monitoring and updating energy consumption data.
Activation:	This use case starts when a user initiates the energy consumption monitoring feature through the mobile app.
Actors:	**Homeowner, ESP Controller(hardware Device), Database**
Main Flow:	1. The user navigates to the energy management section in App 2. The system retrieves real-time energy consumption data for connected devices. 3. The user views and analyses the energy usage information. 4. The user sets preferences for energy-efficient operation, such as scheduling device usage or setting power-saving modes (See A1).
Alternative Flow:	A1 : Energy-saving Preferences • If the user configures energy-saving preferences. • The system applies the selected preferences to connected devices. • The use case continues at position 3 of the main flow.
Exception Flow:	E1: Data Unavailable • If there is a temporary unavailability of energy consumption data. • The system notifies the user and suggests trying again later. • The use case continues at position 4 of the main flow.
Priority:	**High**
Termination:	The system presents the next available command for the user.

Use Case Diagram

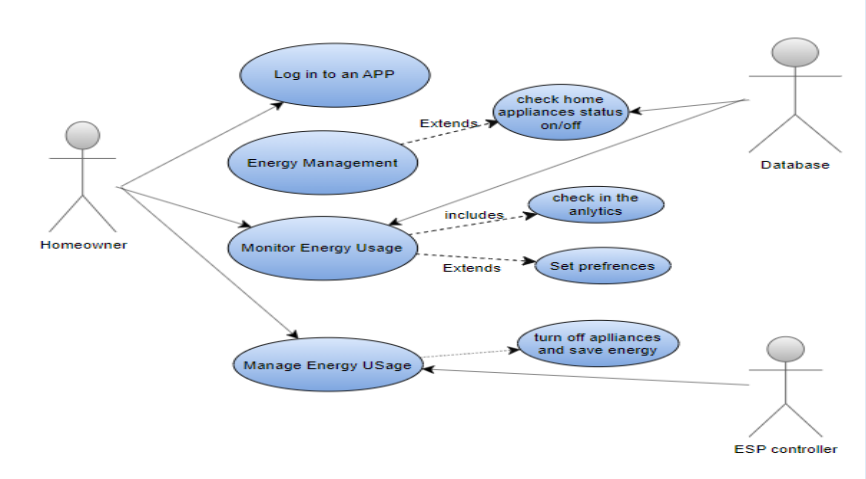


Figure: 7 Use Case Diagram: Energy Management

2.1.2.6. Requirement 6: Multi-User Account Management

Use Case ID:	**UC-6**
Use Case Name:	**Multi-User Account Management**
Use Case Description:	This requirement involves implementing user account management features, allowing homeowners to grant different access levels to family members or trusted individuals. Essential for households with multiple users to ensure personalized control and security. The system must support the management of multiple user accounts with varying access levels for shared smart homes.
Scope:	The scope of this use case is to enable homeowners to manage multiple user accounts within the smart home system.
Description:	This use case describes the process of creating, modifying, and deleting user accounts. Homeowners can assign specific access permissions to each user, controlling their ability to interact with devices and receive notifications.
Pre-conditions:	The system is operational, and the homeowner is logged into the account management section.
Post-Conditions:	The system goes into a wait state, with updated user account information.
Activation:	This use case starts when a homeowner initiates the account management feature through the mobile app.
Actors:	**Homeowner, Firebase Database**
Main Flow:	1. The homeowner navigates to the account management section of the mobile app. 2. The system displays a list of existing user accounts. 3. The homeowner selects an option to create a new user account, modify an existing account, or delete an account (See A1, A2, A3). 4. The system prompts the homeowner to enter or modify user details (e.g., name, email, access permissions). 5. The homeowner confirms the changes, and the system updates the user account information.
Alternative Flow:	A1: Create New User Account • If the homeowner selects the option to create a new user account. • The system guides the homeowner through the process of entering new user details. • Once complete, the system adds the new user account to the list of existing accounts. A2: Modify Existing User Account • If the homeowner selects the option to modify an existing user. • The system displays details of the selected user account for modification. • The homeowner edits the necessary details, and the system updates the user account information. A3: Delete User Account • homeowner selects the option to delete the user account. • The system prompts the homeowner to confirm the deletion.

	• Upon confirmation, the system removes the user account from the list.
Exception Flow:	E1: Account Modification Error • If there is an error in modifying or creating a user account (e.g., invalid input). • The system notifies the homeowner of the issue and provides guidance on resolving it. • The use case continues at position 5 of the main flow
Priority:	**High**
Termination:	The system goes into a wait state, with updated user account information.

Use Case Diagram

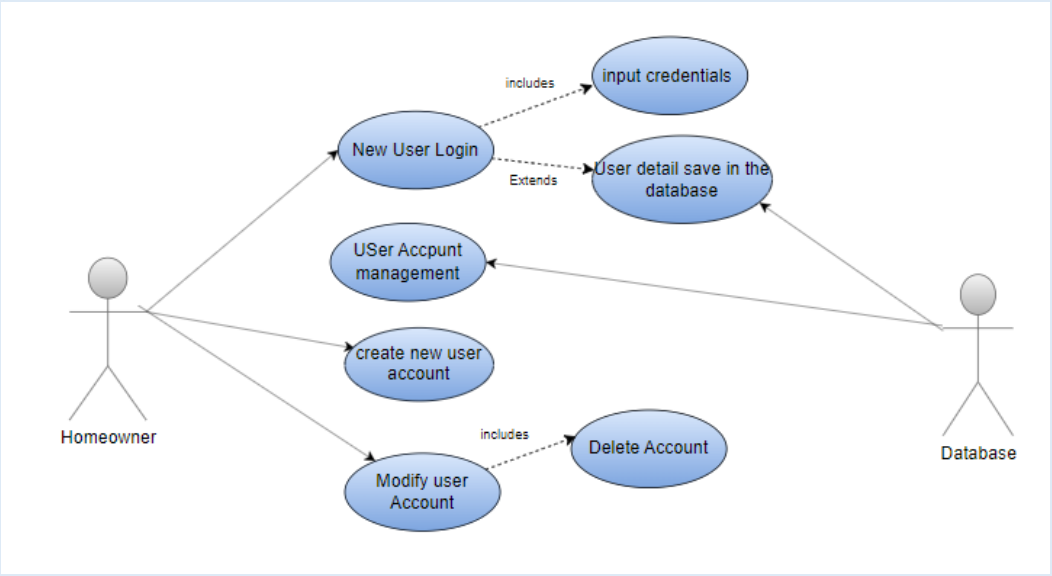


Figure 8: Use Case Diagram: Multi-User Account

2.1.2.7. Requirement 7: User Authentication and Authorisation

Use Case ID:	**UC-7**
Use Case Name:	**User Authentication and Authorisation**
Use Case Description:	The system must authenticate and authorize users to ensure secure access to the mobile application. Priority: High. This requirement is fundamental for maintaining the integrity and security of the smart home system.
Scope:	The scope of this use case is to verify user identity and grant appropriate access levels.
Description:	This use case describes the process of user authentication and authorization when logging in to the mobile application. The system validates user credentials and grants access to authorized features.
Pre-conditions:	The user is attempting to log in to the mobile application.
Post-Conditions:	The system goes into a wait state
Activation	This use case starts when a user provides login credentials
Actors	**Homeowner, ESP Controller(hardware Device)**
Main Flow:	1. The user enters login credentials. 2. The system validates the credentials and grants appropriate access levels. 3. The user gains access to the authorized features of the mobile application.
Exception Flow	E1: Invalid Credentials • If the entered credentials are invalid. • The system notifies the user of the error. • The use case continues at position 3 of the main flow.
Priority:	**High**
Termination:	The user gains access to the authorized features of the mobile application.

Use Case Diagram

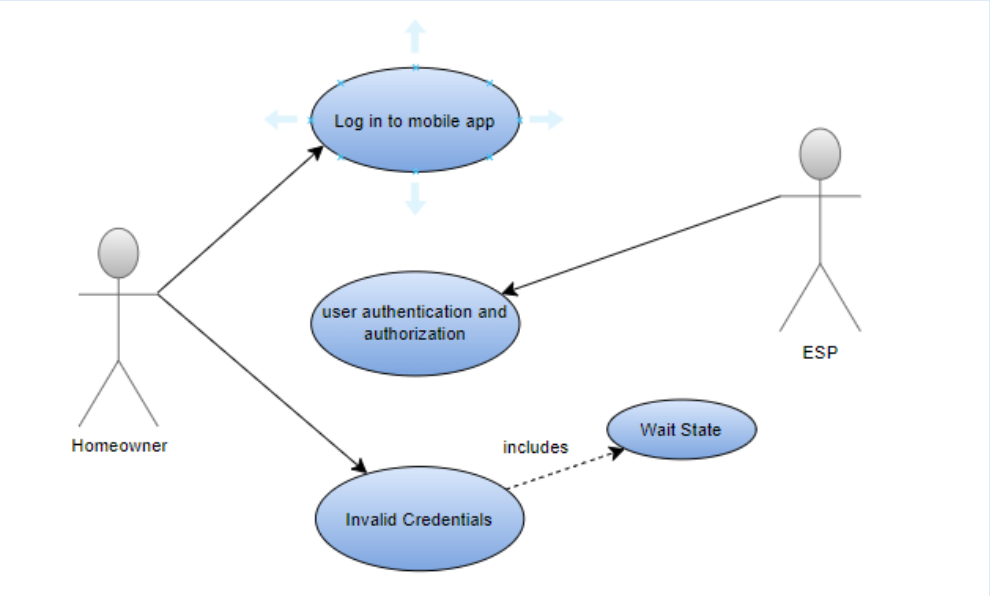


Figure 9: Use Case Diagram: Remote Access

2.1.2 Project Analysis and Planning

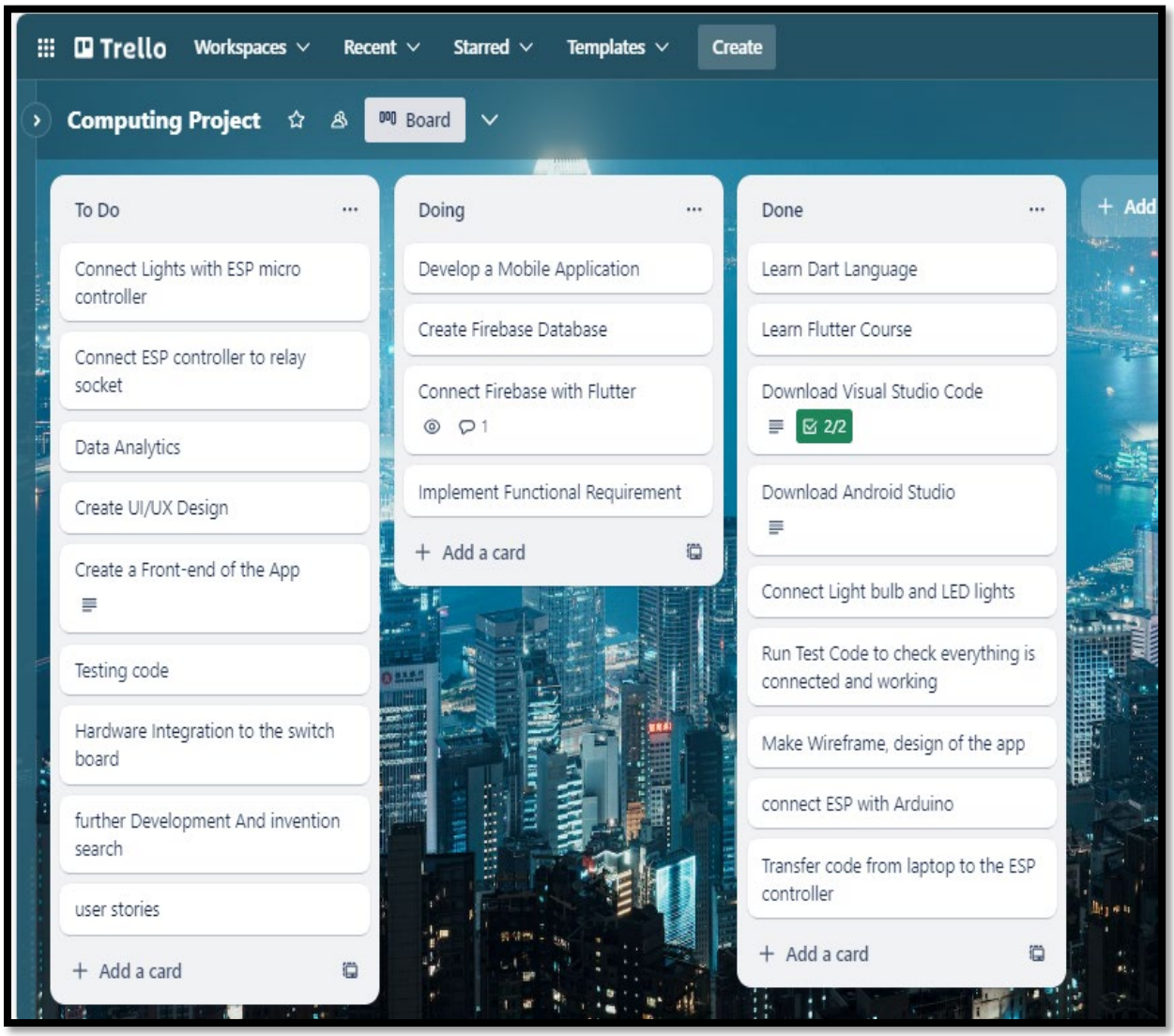


Figure 10: Project Planning on Trello Board

7.0 Project Plan

TASKS	START Date	END Date	DAYS
Project Pitch and Project Proposal			
Project Idea reflection	2023-09-19	2023-10-01	12
Project Pitch	2023-10-02	2023-10-08	6
Project Proposal	2023-10-09	2023-10-29	20
Requirements and Installation			
Learn Flutter and Dart Course	2023-10-30	2023-11-05	6
Install Flutter	2023-11-06	2023-11-08	2
Install Android Studio	2023-11-09	2023-11-11	2
build small app in flutter	2023-11-12	2023-11-15	3
Run some Test code	2023-11-16	2023-11-20	4
Design and Planning			
Make Wireframes	2023-11-21	2023-11-24	3
UI/UX Design	2023-11-25	2023-11-28	3
System Architecture	2023-11-29	2023-12-04	5
Database Setup	2023-12-05	2023-12-07	2
Development and Coding			
Mobile App Development	2023-12-08	2023-12-14	6
Hardware and circuit design	2023-12-15	2023-12-17	2
Hardware Integration	2023-12-17	2023-12-18	1
Mid-point Documentation	2023-12-15	2023-12-20	5
Mid-Point video Presentation	2023-12-19	2023-12-20	1
[illegible]	2023-12-21	2024-01-02	12
Semester 1 Exam Break	2024-01-03	2024-01-13	10
Analytics	2024-01-14	2024-01-25	11
Debug and test the app	2024-01-26	2024-02-05	10
Implement new features	2024-02-06	2024-02-27	21
Quality and Testing			
Unit Testing	2024-02-28	2024-03-03	4
Integration Testing	2024-03-04	2024-03-07	3
Usability Testing	2024-03-08	2024-03-13	5
Performance Testing	2024-03-14	2024-03-20	6
Interface Testing	2024-03-21	2024-03-25	4
Operational Testing	2024-03-26	2024-03-30	4
Documentation			
final Implementation	2024-03-31	2024-04-18	18
Report	2024-04-19	2024-04-25	6
Documentation	2024-04-26	2024-04-30	4
Project review	2024-05-01	2024-05-06	5
Final Testing	2024-05-07	2024-05-10	3
Final Presentation	2024-05-11	2024-05-12	1
Project Showcase	2024-05-28	2024-05-29	1

Figure 11: Table for Project Plan with Dates

2.1.3. Data Requirements

In the context of the IoT Smart Home Automation System, data requirements refer to the necessity of efficient storage, retrieval, and management of information about user profiles, device statuses, energy consumption data. The data architecture should be designed for real-time updates and requirements of data security as well as scalability for both structured and unstructured data types. This includes:

- User Profiles: Retaining and managing user information, image and roles; and preferences.
- Device Statuses: The real-time location monitoring of the connected devices, like the lights, and other home appliances.

2.1.4. User Requirements

As the requirements of the IoT Smart Home Automation System users define the essentials to be provided, design teams must consider providing as much pleasure and ease as possible to the users. These requirements focus on addressing the needs and expectations of diverse users, including: These requirements focus on addressing the needs and expectations of diverse users, including:

- User Authentication: Offering of safe and operation- related logging procedure often seen as primary user authentication and access system protection.
- Intuitive User Interface: Development of an interface that is easy to use and control by the mobile application for the navigation and control of the application.
- Personalization: Giving the ability to users to edit settings, design their schedules and design the room using different devices.
- Multi-User Support: Providing the users of multiple devices in a single household the instrumental support to remotely control the smart home with varying level of accessibility.

2.1.5. Environmental Requirements

The environment requirements for the IoT Smart Home Automation System are the conditions under which the system works. These include:

- Network Stability: Providing a network connection that maintains a stable and secure status for communication via mobile app and IoT units.
- I utilized VS Code to create, code, and run the application. Dart programming language along with the flutter framework is a useful tool for developing apps.
- I stored the data into firebase database.
- It demands internet connection for the user to control the home appliances at the remote corner.

2.1.6. Usability Requirements

Usability needs emphasis making the system more user-friendly and usable by a wide variety of users. For the IoT Smart Home Automation System:For the IoT Smart Home Automation System:

- Accessibility: Creating the mobile app to be available to users with disabilities, the app should have voice commands and screen readers as the features.
- Response Time: Strive for the competent reaction that will carry the cross commands even the user and the implication of the system are having in the general satisfaction.
- Error Handling: Employing meaningful error messages and prompts to guide the users in the event of incorrect inputs or to address the system faults..

- Compatibility: The most important thing is to be compatible with a variety of mobile devices and operating systems to achieve the goal of making the application accessible for users.

2.1.7. Performance Requirements

2.2. Employing meaningful error messages and prompts to guide the users in the event of incorrect inputs or to address the system faults. Performance requirements are vital in order to make sure that the IoT Smart Home Automation System can handle the stipulated number of users and devices and at the same time can keep the system responsive and reliable. These requirements include:

- Scalability: Design the system to be able to handle more traffic rather than becoming slower if the number of users and connected devices increases.
- Concurrency: Support a large number of concurrent users for which the users of the system may carry out their activities all at the same time and the system won't be slowed or compromised, the system can manage enough data and tasks to serve users numbering thousands over time.
- Response Time: Set the time frame by which the user commands and system actions will be accepted. Another example can be to use special functions for common commands like make sure your response time is less than one second.
- Reliability: Promote system high reliability by removing bugs, providing proven fault-tolerant solutions and responding to any failure or disruptions in the system immediately.
- Network Performance: Boost the network performance to reduce the latency and assure that the communication between the mobile application and IoT devices will be smooth.

2.3. Design & Architecture

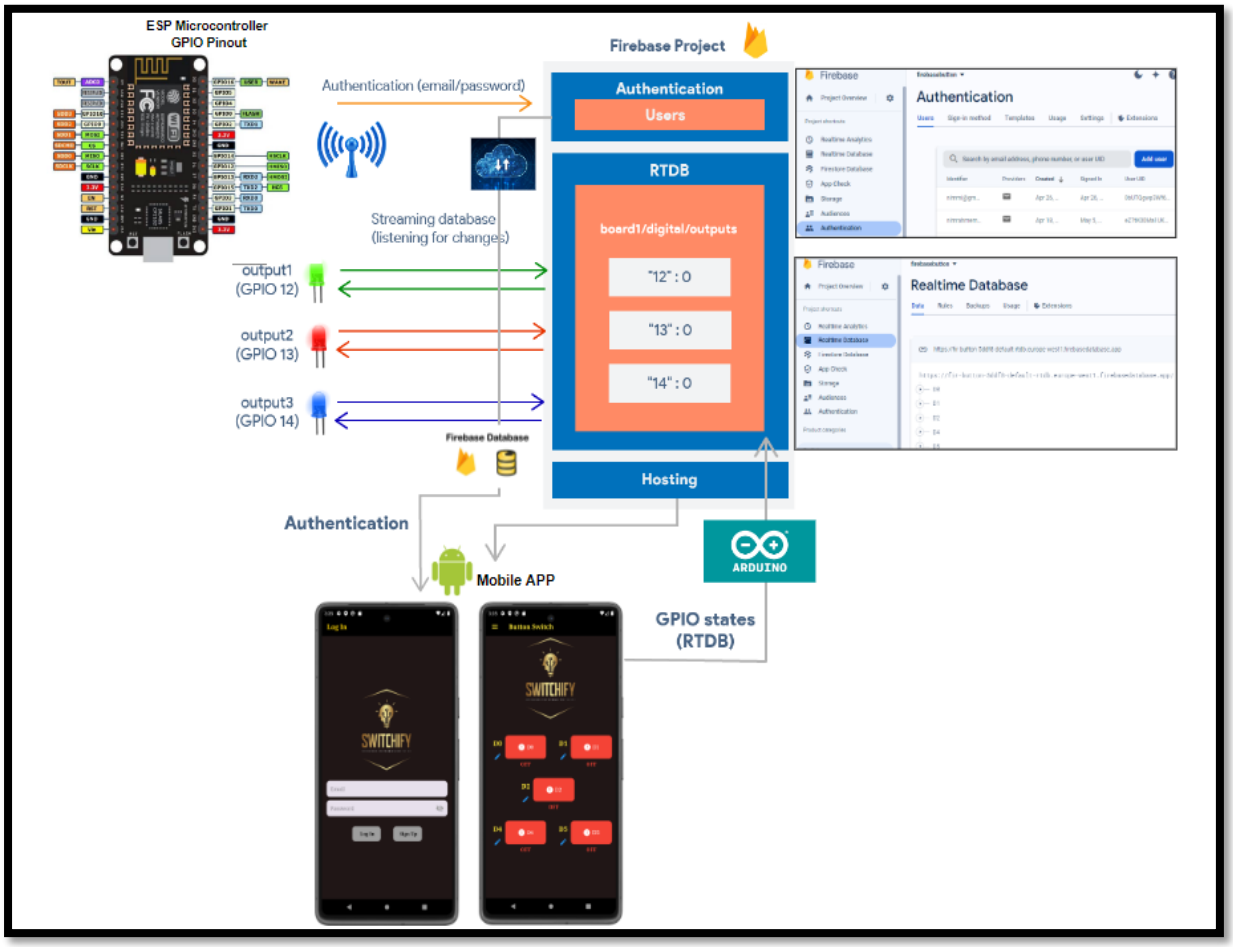


Figure 12: Architectural Diagram

Firebase Hosting and SSL Certificate: Firebase deploys the mobile app worldwide using Firebase Hosting, with SSL certificate to ensure swift access from everywhere while maintaining security.

Authentication: To gain access to the mobile app, users must authenticate by signing in with authorised email addresses and passwords. This functionality enables people to define commands if GPIO is low or high.

Mobile App Interface: The users then authenticate, and they are presented with the mobile app which contains buttons that switch the GPIO states stored in the Firebase real-time database.

Android Device (Mobile App {ESP Switchify}): This represents a user interface of the smart home system; it run on Android devices. The application is developed using Android Studio and VS Code, the mobile app provides users with graphical to interface to interact with their home devices. It allows user to control and monitor various appliances remotely, set schedule and receive notifications. The mobile app communicates with the firebase rea time database to fetch devices status, send control command, and update data in real time database.

Real-time Database Updates: When a user toggles a button to change the GPIO state, the state information for the respective GPIO pin is updated in the firebase real-time database. This leads to the real-time event update live-stream that the ESP8266 sensors are built-on. The backend infrastructure for storing and synchronizing data across devices is provided by the Firebase real-time database. This NoSQL database, which is hosted on the Google Cloud Platform, holds data on user

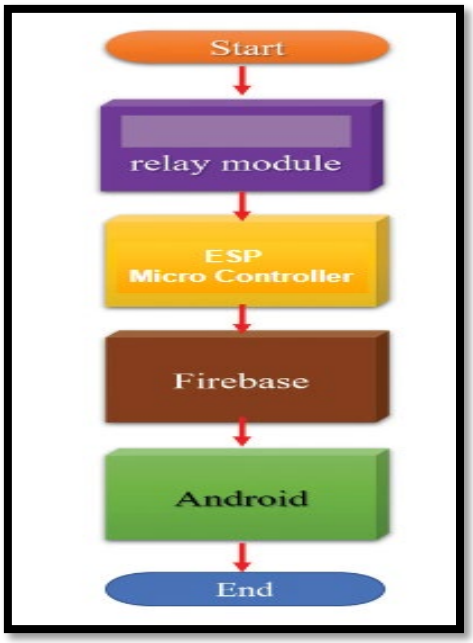
accounts, device setups, and real-time updates about the status of devices. It ensures users obtain the most recent information and control capabilities from any location with an internet connection by facilitating smooth synchronization between the mobile app and the ESP microcontroller. ***See Figure 12.***

ESP32/ESP8266 Integration: The micro controller ESP is on the lookout for changes in the database that's placed continuously. The ESP updates the actual GPIO states when there is a change in the GPIO state in the database, which allows you to control connected devices in real time. The ESP microcontroller serves as the central processing unit within the smart home system, facilitating communication with the Firebase database and hardware components. It commonly runs firmware compatible with Arduino and is based on the ESP8266 or ESP32 platform. In order to provide bidirectional communication with the mobile app, the microcontroller establishes an Ethernet or Wi-Fi connection with the Firebase database. It recognizes commands from the app, manages the connected devices (relays, fans, and LED lights), and updates the database with status information. ***See Figure 12.***

Current State Display: The app shows the present status of the GPIOs, making a representation of the usable devices controlled for the user. This physical setup serves as a representation of the smart home system's hardware. A breadboard can be used as a development platform to connect relays, fans, and LED lights, among other devices. The ESP microcontroller is in charge of these small devices and executes user commands obtained through the mobile app. For instance, the LED bulbs can be used to visually indicate the system's condition, and the relay can be utilized to turn on and off high-power equipment like fans.

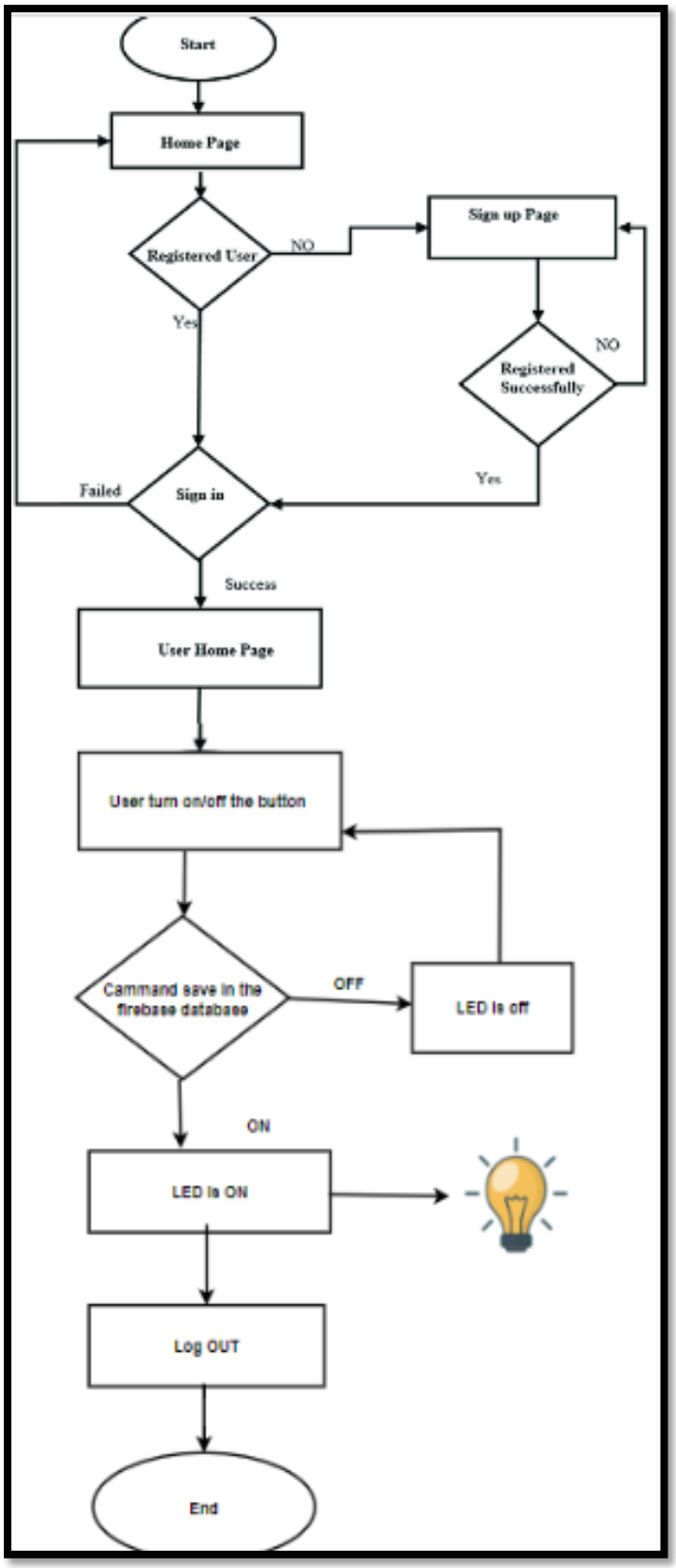
User Logout: The app users are capable of logging out, which would demand users to re-authenticate their access altogether the next time they try to use the mobile app.

System Function Block Diagram



The lights can be turned on and off remotely through an application connected to the internet. After inputting data, the modules will send values to ESP controller. After that, data will be transmitted to Firebase's cloud management platform for further processing. The System function blocam shows the step-by-step process of the syste

Figure 13: System Diagram



Navigation Diagram (flow chart of the sayetm)

User Registration/Login:

- User opens the mobile application.
- If new user, they register with email and password. Otherwise, they log in.

Home Dashboard:

- Upon successful login, user is directed to the home dashboard.
- Dashboard displays status of connected devices and available options.

Device Control:

- User selects a device (e.g., lights, fan) from the dashboard.
- They can control the device by toggling on/off.

Save Data in to Database:

- The button pressed data save in the database
- Led lights and other applinaces work according to the status of the button on/off. ***See figure 14***

Remote Access:

- User accesses the application remotely.
- They log in and connect to the smart home system.
- Remote control of devices and monitoring of home status is available.

Logout:

- User logs out of the application, ending the session.

Figure 14: Navigation Diagram

2.4. Implementation

2.3.1 Splash Screen

```dart
import 'package:flutter/material.dart';

class SplashScreen extends StatefulWidget {
  final Widget? child;
  const SplashScreen({super.key, this.child});

  @override
  State<SplashScreen> createState() => _SplashScreenState();
}

class _SplashScreenState extends State<SplashScreen> {
  final GlobalKey<ScaffoldState> _scaffoldKey = new GlobalKey<ScaffoldState>();
  Color shinyGoldColor = Color.fromRGBO(255, 215, 0, 1);    Use 'const' with

  @override
  void initState() {
    Future.delayed(Duration(seconds: 3), () {    Use 'const' with the construct
      Navigator.pushAndRemoveUntil(
          context,
          MaterialPageRoute(builder: (context) => widget.child!),
          (route) => false);
    }); // Future.delayed
    super.initState();
  }
```

```dart
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      key: _scaffoldKey,
      backgroundColor: const Color.fromARGB(255, 32, 22, 22),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Image.asset(
              'assets/images/switchifylogo.png',
              width: 230,
              height: 230,
            ), // Image.asset
            const SizedBox(height: 20), // Add spacing between image and text
            const Text(
              "Empower you to reach for  the sky",
              style: TextStyle(
                color: Colors.yellow,
                fontWeight: FontWeight.normal,
                fontSize: 18,
                fontFamily: 'Georgia',
              ), // TextStyle
            ), // Text
          ],
        ), // Column
      ), // Center
    ); // Scaffold
  }
}
```

Algorithm 1:
The splash screen code display code listing.

The splash screen code display the name and logo of the app, when the user open this app, the splash screen will run for 3 seconds, see Algorithm 1.

2.3.2 Sign up Page

```dart
class SignUpPage extends StatefulWidget {
  const SignUpPage({super.key});

  @override
  State<SignUpPage> createState() => _SignUpPageState();
}

class _SignUpPageState extends State<SignUpPage> {
  final GlobalKey<ScaffoldState> _scaffoldKey = new GlobalKey<S

  @override
  void dispose() {
    nametextEditingController.dispose();
    emailtextEditingController.dispose();
    passwordtextEditingController.dispose();
    mobiletextEditingController.dispose();
    super.dispose();
  }
```

```dart
  TextEditingController nametextEditingController = TextEditingControl
  TextEditingController emailtextEditingController = TextEditingContro
  TextEditingController passwordtextEditingController = TextEditingCon
  TextEditingController mobiletextEditingController = TextEditingContr

  final FirebaseAuth _firebaseAuth = FirebaseAuth.instance;

  void registerNewUser(BuildContext context) async {
    final User? firebaseUser = (await _firebaseAuth
            .createUserWithEmailAndPassword(
                email: emailtextEditingController.text,
                password: passwordtextEditingController.text)
            // ignore: body_might_complete_normally_catch_error
            .catchError((errMsg) {
      displayToastMessage("Error:$errMsg", context);
    }))
        .user;
    if (firebaseUser != null) {
      //save data to firebase real time database
      Map userDataMap = {
        "name": nametextEditingController.text,
        "email": emailtextEditingController,
        "mobile": mobiletextEditingController,
        "password": passwordtextEditingController,
      };
      usersRef.child(firebaseUser.uid).set(userDataMap);
      displayToastMessage(    Don't use 'BuildContext's across async g
          "Congratulation, your account has been created", context);
      Navigator.of(context).push(_createRoute());    Don't use 'BuildC
    } else {
      // display error toast
      displayToastMessage("New user has NOT been created", context);
    }
  }
```

Algorithm 2:
Signup page display the fields for the username, email, mobile and password

This code defines a Flutter widget called SignUpPage that serves as a form for users to sign up. It utilizes Firebase for **authentication** and real-time **database** storage. The _SignUpPageState class manages the state of the SignUpPage. It includes text editing controllers for capturing user input for **name, email, password, and mobile number.** Upon submission, the registerNewUser function attempts to create a new user account using Firebase Auth, and upon success, it saves user data to Firebase Realtime Database. If successful, it displays a success message and navigates to a new route, otherwise, it displays an error message. See Algorithm 4.

2.3.3 Login Page

```dart
class LoginPage extends StatefulWidget {
  const LoginPage({super.key});

  @override
  State<LoginPage> createState() => _LoginPageState();
}

class _LoginPageState extends State<LoginPage> {
  // final FirebaseAuthService _auth = FirebaseAuthService();
  final GlobalKey<ScaffoldState> _scaffoldKey = GlobalKey<ScaffoldState>();

  @override
  void dispose() {
    emailtextEditingController.dispose();
    passwordtextEditingController.dispose();
    super.dispose();
  }
```

```dart
  TextEditingController emailtextEditingController = TextEditingController();
  TextEditingController passwordtextEditingController = TextEditingController();
  final FirebaseAuth _firebaseAuth = FirebaseAuth.instance;

  void loginAndAuthenticationUser(BuildContext context) async {
    final User? firebaseUser = (await _firebaseAuth
            .signInWithEmailAndPassword(
                email: emailtextEditingController.text,
                password: passwordtextEditingController.text)
            // ignore: body_might_complete_normally_catch_error
            .catchError((errMsg) {
      displayToastMessage("Error:$errMsg", context);
    }))
        .user;
    if (firebaseUser != null) {
      usersRef.child(firebaseUser.uid).once().then((DataSnapshot snap) {
        if (snap.value != null) {
          Navigator.of(context).push(createRoute2());
          displayToastMessage("you are looged in", context);
        } else {
          _firebaseAuth.signOut();
          displayToastMessage(
              "No record exist for this user. please create a anew account",
              context);
        }
      } as FutureOr Function(DatabaseEvent value));
    }
  }

  displayToastMessage(String message, BuildContext context) {
    Fluttertoast.showToast(msg: message);
  }
```

Algorithm 3:

Login Page code display the email and password field and save in Firebase

The word here displays a Flutter widget which has been named LoginPage; as for the users, this widget is the login page. Authentication component relies on firebase. The _LoginPageState class takes care of the state of the LoginPage. It consists of text edit controllers that serve as capturing means, for entering an email and password. On further submitting the data, the function loginAndAuthenticationUser tries to authenticate the user with the help of the Firebase Auth. The process of authentication is successful, which is followed by the verification of the user's existence in the Firebase Realtime Database. When yes, it it goes through the new route and displays a success message; otherwise, it validates the user and sends the user an error message.

2.3.4 Navigation bar

```dart
          ), // DrawerHeader
          ListTile(
            title: const Text('ESP 8266 Pinout'),
            onTap: () {
              Navigator.of(context).push(_createRoute());
            },
          ), // ListTile
          ListTile(
            title: const Text('Switch Analysis'),
            onTap: () {
              Navigator.of(context).push(_createRoutePointer());
            },
          ), // ListTile
          ListTile(
            title: const Text('Graph Analysis for Switches'),
            onTap: () {
              Navigator.of(context).push(_createRouteAnalytics());
            },
          ), // ListTile
          ListTile(
            title: const Text('About Us'),
            onTap: () {
              Navigator.of(context).push(_createRouteAboutus());
            },
          ), // ListTile
          ListTile(
            title: const Text('Sign Out'),
            onTap: () {
              Navigator.of(context).push(_createRouteSignout());
            },
          ), // ListTile
        ],
      ), // ListView
    ), // Drawer
    backgroundColor: const Color.fromARGB(255, 32, 22, 22),
```

```dart
Route _createRouteAnalytics() {
  return PageRouteBuilder(
    pageBuilder: (context, animation, secondaryAnimation) =>
        const Analytics(),
    transitionsBuilder: (context, animation, secondaryAnimation, child) {
      const begin = Offset(0.0, 1.0);
      const end = Offset.zero;
      const curve = Curves.fastOutSlowIn;
      var tween =
          Tween(begin: begin, end: end).chain(CurveTween(curve: curve));
      return SlideTransition(
        position: animation.drive(tween),
        child: child,
      ); // SlideTransition
    },
  ); // PageRouteBuilder
}

Route _createRoutePointer() {
  return PageRouteBuilder(
    pageBuilder: (context, animation, secondaryAnimation) => const Pointer(),
    transitionsBuilder: (context, animation, secondaryAnimation, child) {
      const begin = Offset(0.0, 1.0);
      const end = Offset.zero;
      const curve = Curves.fastOutSlowIn;
      var tween =
          Tween(begin: begin, end: end).chain(CurveTween(curve: curve));
      return SlideTransition(
        position: animation.drive(tween),
        child: child,
      ); // SlideTransition
    },
  ); // PageRouteBuilder
}

Route _createRouteAboutus() {
  return PageRouteBuilder(
    pageBuilder: (context, animation, secondaryAnimation) => Aboutus(),
    transitionsBuilder: (context, animation, secondaryAnimation, child) {
      const begin = Offset(0.0, 1.0);
      const end = Offset.zero;
      const curve = Curves.fastOutSlowIn;
      var tween =
          Tween(begin: begin, end: end).chain(CurveTween(curve: curve));
      return SlideTransition(
```

Algorithm 4

Navigation bar code display the name and routes of different pages

This code snippet represents a navigation bar in a Flutter app, implemented using ListTile widgets. Each ListTile represents a different destination or page in the app. When a user taps on a specific tile, it triggers navigation to the corresponding page using Navigator.of(context).push() along with custom route transitions defined by _createRoute*() functions. The pages included in the navigation bar are: "ESP 8266 Pinout", "Switch Analysis", "Graph Analysis for Switches", "About Us", and "Sign

Out". The "Sign Out" tile is expected to navigate the user to the login page (LoginPage) upon tapping, effectively logging them out of the application.

2.3.5 Button Switch Page (Main Screen)

```
class _WebSocketLed extends State<WebSocketLed> {
  final GlobalKey<ScaffoldState> _scaffoldKey = new GlobalKey<ScaffoldState>();

  late bool D0status; //boolean value to track LED status, if its ON or OFF    T
  late bool D1status;    The variable name 'D1status' isn't a lowerCamelCase ide
  late bool D2status;    The variable name 'D2status' isn't a lowerCamelCase ide
  late bool D4status;    The variable name 'D4status' isn't a lowerCamelCase ide
  late bool D5status;    The variable name 'D5status' isn't a lowerCamelCase ide

  late bool D2modeA = false;    The variable name 'D2modeA' isn't a lowerCamelCa
  late String D2level = "";    The variable name 'D2level' isn't a lowerCamelCas

  late IOWebSocketChannel channel;
  late bool connected; //boolean value to track if WebSocket is connected
  bool d0 = false;
  bool d1 = false;
  bool d2 = false;
  bool d4 = false;
  bool d5 = false;
  final D0Controller = TextEditingController();    The variable name 'D0Controll
  final D1Controller = TextEditingController();    The variable name 'D1Controll
  final D2Controller = TextEditingController();    The variable name 'D2Controll
  final D4Controller = TextEditingController();    The variable name 'D4Controll
  final D5Controller = TextEditingController();    The variable name 'D5Controll

  final DatabaseReference _databaseReference = FirebaseDatabase.instance.ref();
```

Inside the widget's state class **_WebSocketLed**, it initializes several variables to track the status of LEDs (D0status, D1status, etc.), The IOWebSocketChannel (channel) is used to establish a bidirectional communication channel over a WebSocket. The FirebaseDatabase.instance.ref() creates a reference to the Firebase Realtime Database.

Algorithm 5:
button page code displays the variables

```
47  void _updateFirebase(String device, int state) {
48    _databaseReference.child(device).update({
49      'timestamp': DateTime.now().toString(),
50      'Status': state,
51    });
52  }
53
54  void updateTime(String device) async {
55    DatabaseReference dbRef =
56        FirebaseDatabase.instance.ref().child(device).child('Status');
57    DatabaseReference dbRef2 =
58        FirebaseDatabase.instance.ref().child(device).child('time');
59
60    DatabaseEvent dataSnapshot = await dbRef.once();
61    DatabaseEvent dataSnapshot2 = await dbRef2.once();
62
63    int statusdata = int.parse(dataSnapshot.snapshot.value.toString());
64    int timedata = int.parse(dataSnapshot2.snapshot.value.toString());
65
66    print('Status: $statusdata' + ' - Time: $timedata');    Don't invoke 'print
67
68    if (statusdata == 1) {
69      timedata += 1;
70    } else if (statusdata == 0) {
71      timedata += 0;
72    }
73    if (timedata >= 1440) {
74      timedata = 0;
75    }
76    _databaseReference.child(device).update({
77      'time': timedata,
78    });
79    print(timedata);    Don't invoke 'print' in production code. Try using a l
80  }
81  void periodic(String device) {
82    Timer.periodic(Duration(minutes: 1), (timer) {    Use 'const' with the cons
83      updateTime(device);
84    }); // Timer.periodic
85  }
```

In this code, there are three main functions defined.

1. _updateFirebase: This function mutates the db_name firebase on the node_name device with a new timestamp and state. It takes ID, the device identifier and integer state as parameters and performs these two tasks each time: it updates the 'timestamp' field of the corresponding node using the current timestamp and a 'status' field using the provided state.
2. updateTime: This function is in charge of updating the 'time' field of the specified device node in the database according to the state of the device at present. It reads the number, time and status from the database. Status of 1 indicates that the device is ON. When the time incrementing is done, then, only,the field 'time' gets updated in the database. The time exceeding 1440 (24 hours), the show counts down and it resets 0.
3. periodic: This function creates a periodic timer that triggers the updateTime function for the specified device every minute. It seeks to avoid situation where the database time data is stagnant because of the inactive device by synchronizing the database time with the status of the device. See algorithm 6

Algorithm 6:
It displays different function, read the above description for more detail

2.3.6 Switch Analysis Page

Algorithm 7:
Switch Analysis code display the graphs

Below is the Snippet of the Flutter Code 'Pointer' stateful widget which is used to display a Radial Gauge for each the given devices(D0, D1, D2, D4, D5). In the initState method the time data for each device is fetched from the Firebase Realtime Database asynchronously using the fetchTime function. The retrieved data for the times is then further sliced into pieces of 60 to give it the events that occurs in hours.

To build the radial gauge, the SfRadialGauge widget from Syncfusion SF-Gauges package is leveraged. Each dial is made of a radial axis with ranges divided into three sections based on time intervals (0-8 hours, 8-16 hours, and 16-24 hours) represented by different colors. Moreover, a gauge mode preventing visual overlap accessed from Firebase, shows a pointer that indicates the current time fetched and annotation with a corresponding value in hours out of 24 is on either of the gauges. Basically, this widget helps visualize time data in radial gauge not to mention, an at-a-glance way for users to show the over-all status of the device or each device on the web-page within 24-hours. See algorithm 7,

2.3.7 About Us Page

```
class Aboutus extends StatelessWidget {    Constructors for public widgets should have a n
  final GlobalKey<ScaffoldState> _scaffoldKey = new GlobalKey<ScaffoldState>();    Unneces

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      key: _scaffoldKey,
      appBar: AppBar(
        title: const Text(
          "About Us",
          style: TextStyle(
            color: Colors.yellow,
            fontWeight: FontWeight.bold,
            fontSize: 20,
          ), // TextStyle
        ), // Text
        backgroundColor: Color.fromARGB(255, 0, 0, 0),    Use 'const' with the construct
      ), // AppBar
      backgroundColor: const Color.fromARGB(255, 32, 22, 22),
      body: Center(
        child: SingleChildScrollView(
          padding: EdgeInsets.all(20.0),    Use 'const' with the constructor to improve pe
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.center,
            children: <Widget>[
              Image.asset(
                'assets/images/switchifylogo.png',
                width: 200,
                height: 250,
              ), // Image.asset
              const SizedBox(height: 40),
              const Text(
                'Welcome to ESP Switchify - Your Smart Home Automation Solution!',
```

Algorithm 8:
Abouts us code display the logo and description of the app

In this code, Aboutus is given as the name of a stateless widget that represents the page of an app named "About us" It consists of an app bar with a customized title and the background color of your choice. This body part is comprised of a vertical column, that displays an image, a welcome message, an application's usage description written in paragraph format as well as keywords and key features shown in bullet points. With each element having a title and description, there are proper styles to match.

2.3.8 Hardware Implementation

Arduino Sketch Code for ESP that connects with Firebase [Realtime Database (RTDB)]

```
sketch_mar23a §
// Insert Firebase project API Key
#define API_KEY "AIzaSyB_G8FCIQdkj_hZbEKd0gzNyYbVac

// Insert RTDB URLefine the RTDB URL */
#define DATABASE_URL "https://fir-button-5ddf8-defa

//GPIO Pins for buttons
#define D0 16
#define D1 5
#define D2 4
#define D4 2
#define D5 14

//Define Firebase Data object
FirebaseData fbdo;
FirebaseAuth auth;
FirebaseConfig config;
unsigned long sendDataPrevMillis = 0;
bool signupOK = false;

void setup() {
  pinMode(D0, OUTPUT);
  pinMode(D1, OUTPUT);
  pinMode(D2, OUTPUT);
  pinMode(D4, OUTPUT);
  pinMode(D5, OUTPUT);
  Serial.begin(115200);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
```

```
sketch_mar23a §
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();

  /* Assign the api key (required) */
  config.api_key = API_KEY;

  /* Assign the RTDB URL (required) */
  config.database_url = DATABASE_URL;

  /* Sign up */
  if (Firebase.signUp(&config, &auth, USER_EMAIL, USER_PASSWORD)) {
    Serial.println("signup done");
    signupOK = true;
  }
  else {
    Serial.printf("%s\n", config.signer.signupError.message.c_str());
  }

  /* Assign the callback function for the long running token generation task */
  config.token_status_callback = tokenStatusCallback; //see addons/TokenHelper.h

  Firebase.begin(&config, &auth);
  Firebase.reconnectWiFi(true);
}
```

Algorithm 9:
Arduino code display the connection of the Database and GPIO pin

These lines of code are part of an Arduino sketch for the ESP8266 or ESP32 microprocessors which, when connected to the Firebase Real-Time Database (RTDB), enables basic functions such as creating users and uploading the Firebase settings. It commences by specifying the API key of the Firebase project and the RTDB URL, along with the GPIO pins for the buttons. It begins with installing Firebase authentication and configuration, then tries to start registration with the inputs. It configures my Wi-Fi, makes the connection to Firebase, and adds the provided settings into it. Besides, it also provides a callback function for monitoring the token status during the running of long tasks. Eventually, the computer code creates grounds for interacting with a server through Firebase RTDB, which allows for features like data reading, writing and authentication.

```
sketch_mar23a §
}

void D0button() {
  int statD0;
  String sD0;
  if (Firebase.ready() && signupOK && (millis() - sendDataPrevMillis > 100 || sendDataPrevMillis == 0)) {
    sendDataPrevMillis = millis();
    if (Firebase.RTDB.getInt(&fbdo, "/D0/Status")) {
      if (fbdo.dataType() == "int") {
        statD0 = fbdo.intData();
        if (statD0 == 0) {
          sD0 = "Off";
        }
        else if (statD0 == 1) {
          sD0 = "On";
        }
        Serial.println("D0 status = ");
        Serial.print(sD0);
        Serial.println();
      }
    }
    else {
      Serial.println(fbdo.errorReason());
    }

    if (statD0 == 0) {
      digitalWrite(D0, LOW);
      delay(1000);
    }
    else if (statD0 == 1) {
      digitalWrite(D0, HIGH);
      delay(100);
    }
  }
}
```

Algorithm 10:

The code shows the connection between the button and firebase

This function D0button is the one that handles the input from the button connected to the GPIO pin D0 and outputs to the desired device, for example, an LED. In verify_firebase_and_update_db_interval, it tests if the Firebase is ready to use, if user sign up is successful, and a certain time interval passed since the last data collection. It then reads the status D0 from RTDB (Firebase Realtime Database), toggles the output device based on its retrieved status (either on it is , and prints the status to the serial monitor. In general, this feature gives a real-time control of an output device on the basis of the status which is saved in the Firebase RTDB.

2.3.9 Hardware Device (ESP with LEDs)

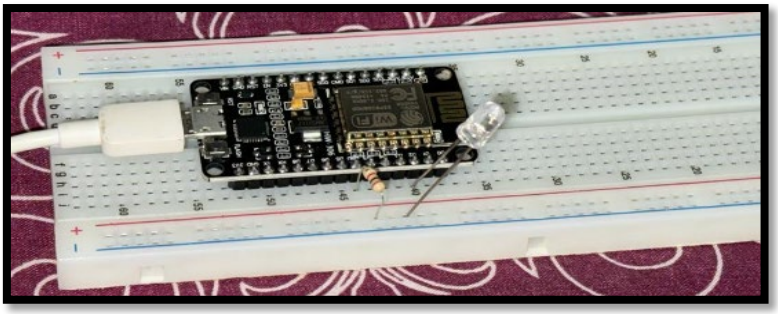


Figure 15: ESP connected with Breadboard

This is my hardware device called ESP microcontroller which is attached to breadboard, and I used arduino IDE and install a library in Arduino called ESP8266 and tranfer the arduino code into the ESP device to turn on the LED light.

2.5. Graphical User Interface (GUI)

2.5.1. Splash Screen

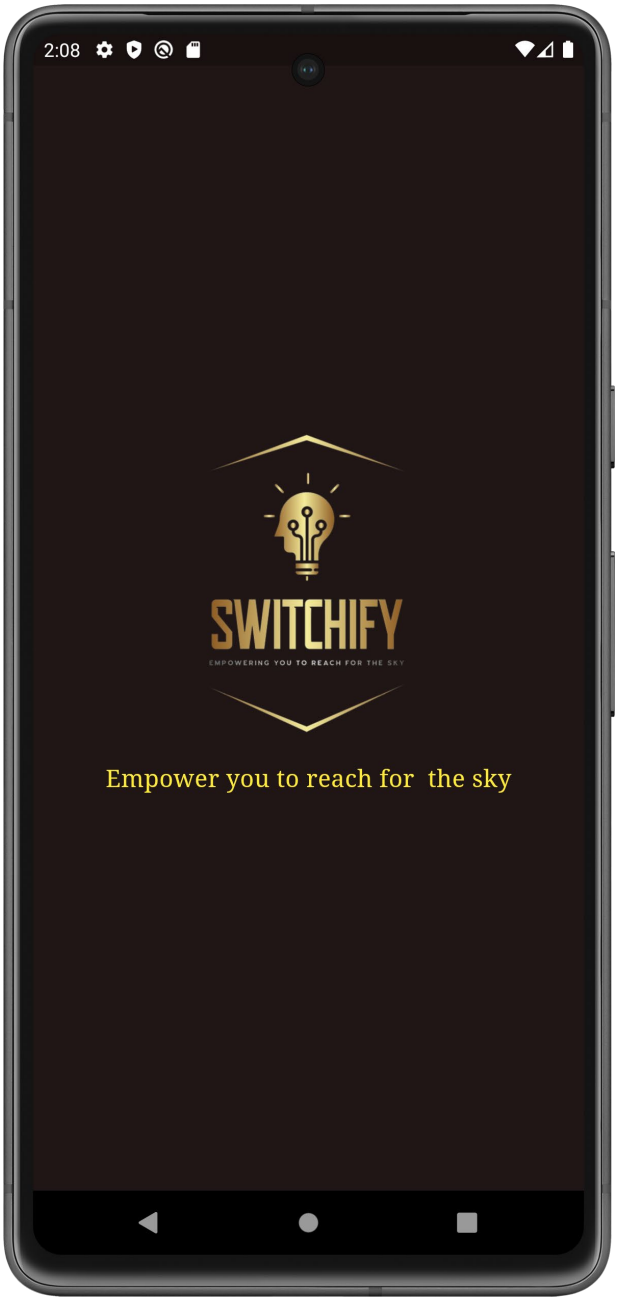


Figure 16

This is the 'Spalsh Screen' of the application. This is the screen that displays when a user runs the app. This screen shows the Logo of the application and slogan and it run for 3 seconds on the screen and then the next page appears.

2.5.2. Sign UP and Login Screen

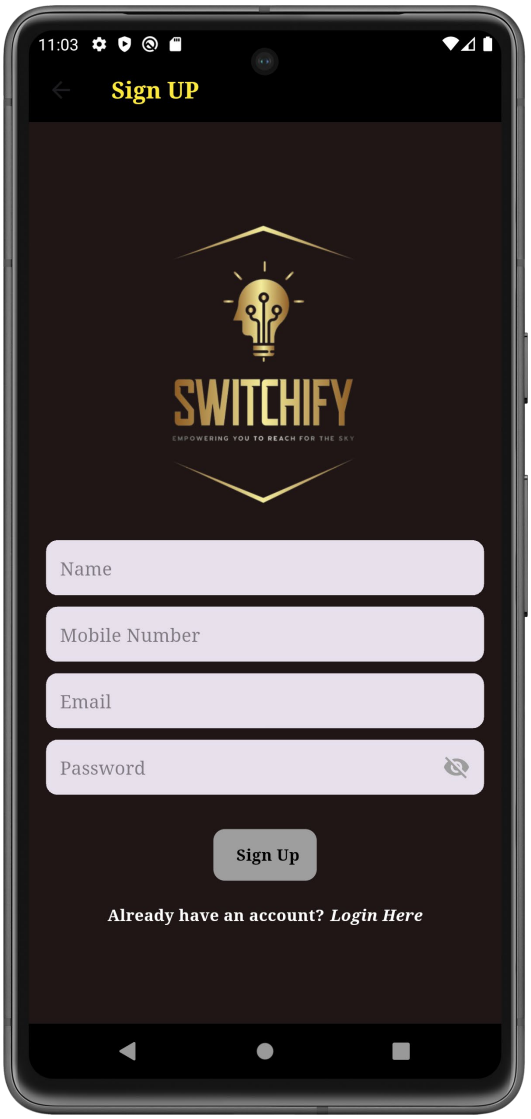


Figure 17: Sign up

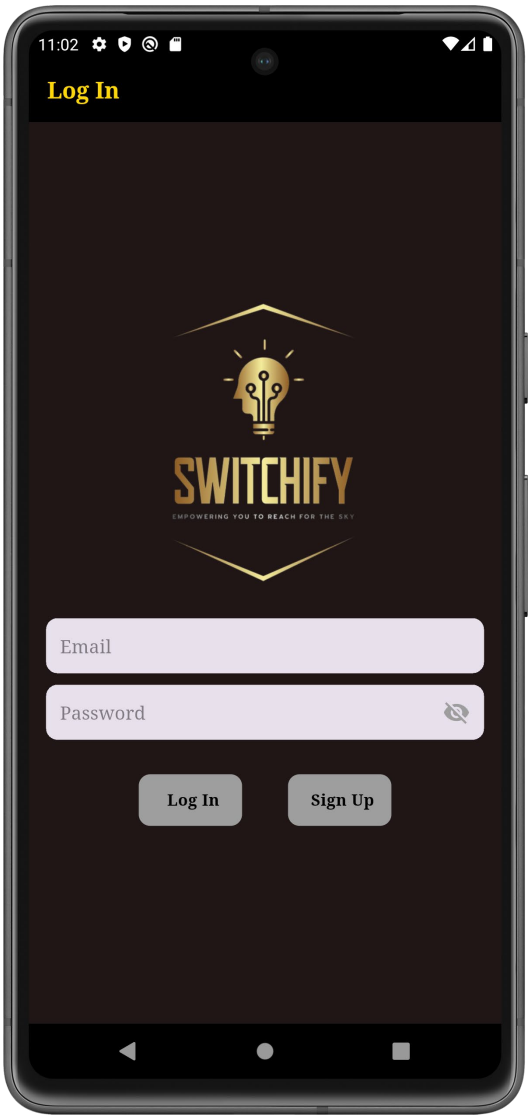


Figure 18: Login

This is the 'Sign Up Screen'. This is where users will navigate to in order to register to the system. A user must fill in all the fields provided and tap the 'Sign Up' button to register. After creating a new account , the user will be able to Log in to their account.

The Login Screen allows registered users to Login into the system by inputting the Valid credentials into the fields provided.

2.5.3. Sign up (display error msg) and Navigation Bar

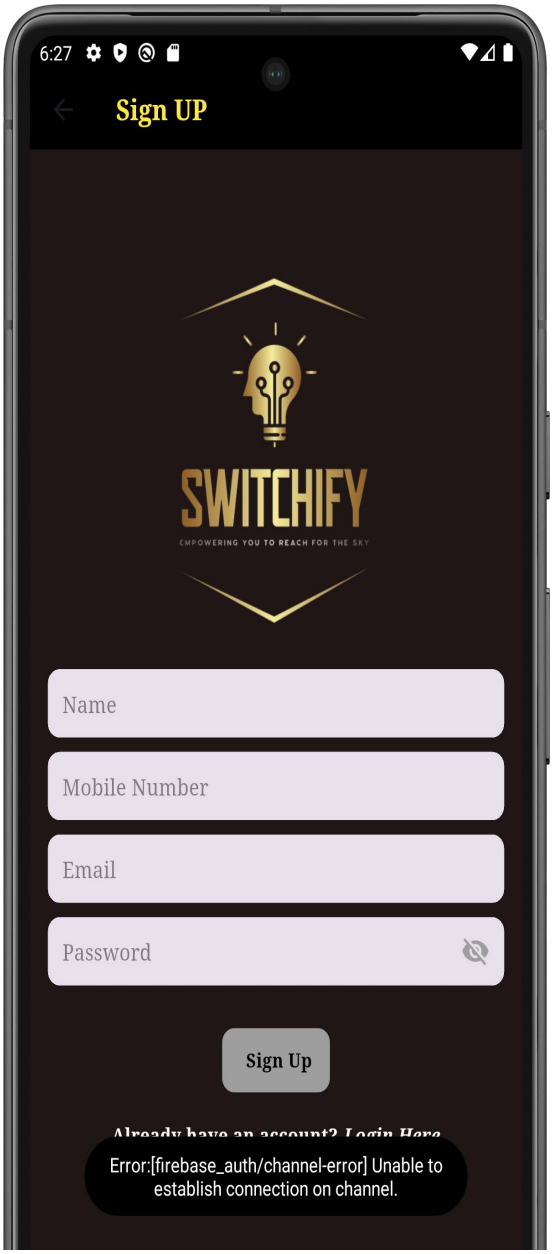


Figure 19: display error msg

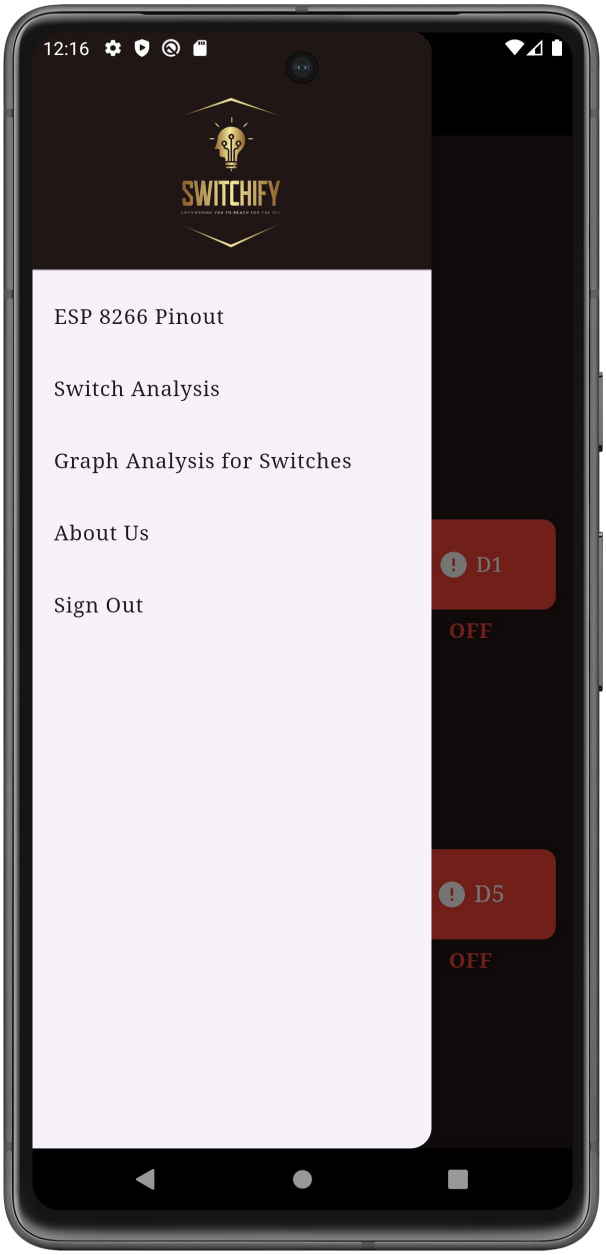


Figure 20: Navigation Bar

- If the user leaves the field empty it will display an error msg, user account is not created. In order to successfully create an account user needs to fill out all the fields and then select signup button.

- Figure 20, shows the navigation bar of the application, all the different pages are shown in the side bar, user can click on any page to navigate to that page.

2.5.4. Button Switch Page

Figure 21: Button Switch Off

Figure 22: Button Switch On

Botton Switch page display the buttons for different led light, you can click the button and it will turn green and Turn on the light on the ESP breadboard

In figure 21 it shows that all the buttons are red that means all the lights are Turned Off
In figure 22 it shows that all the buttons are green that means all the lights are Turned On

2.5.5. Button Switch (with custom name) and Dialog Box

Figure 23: Custom Name for Buttons

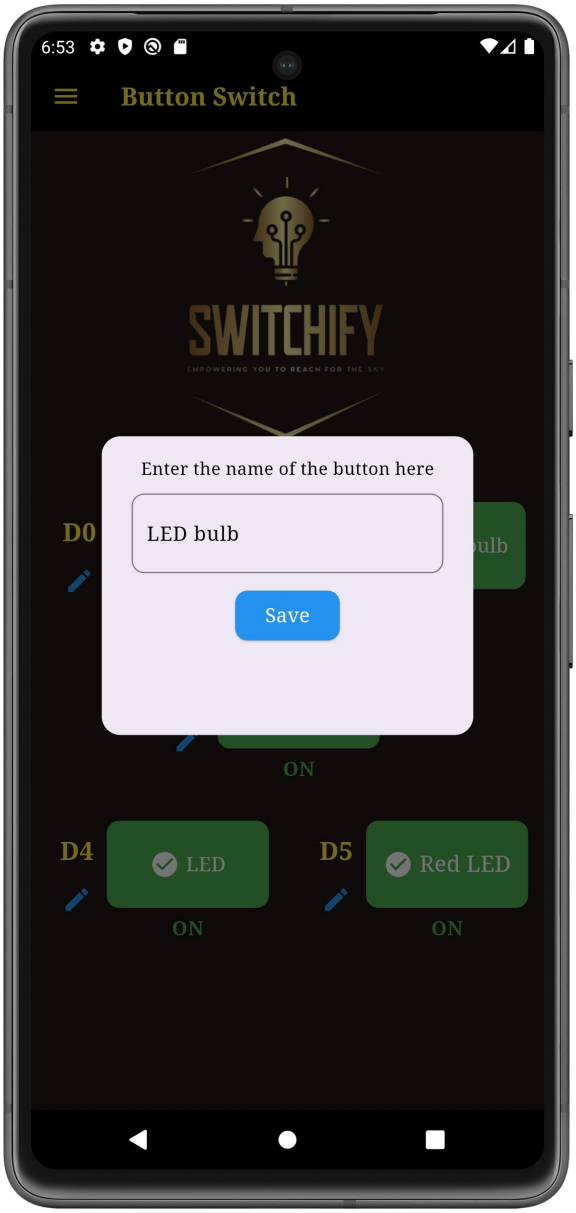


Figure 24: Dialog Box

In Figure 23, it shows that the name on each button is customizable, you can give any name to that button according to your device. Also in figure 24, it display a dialog box where you can write any name and save it.

2.5.6. Pinout Image

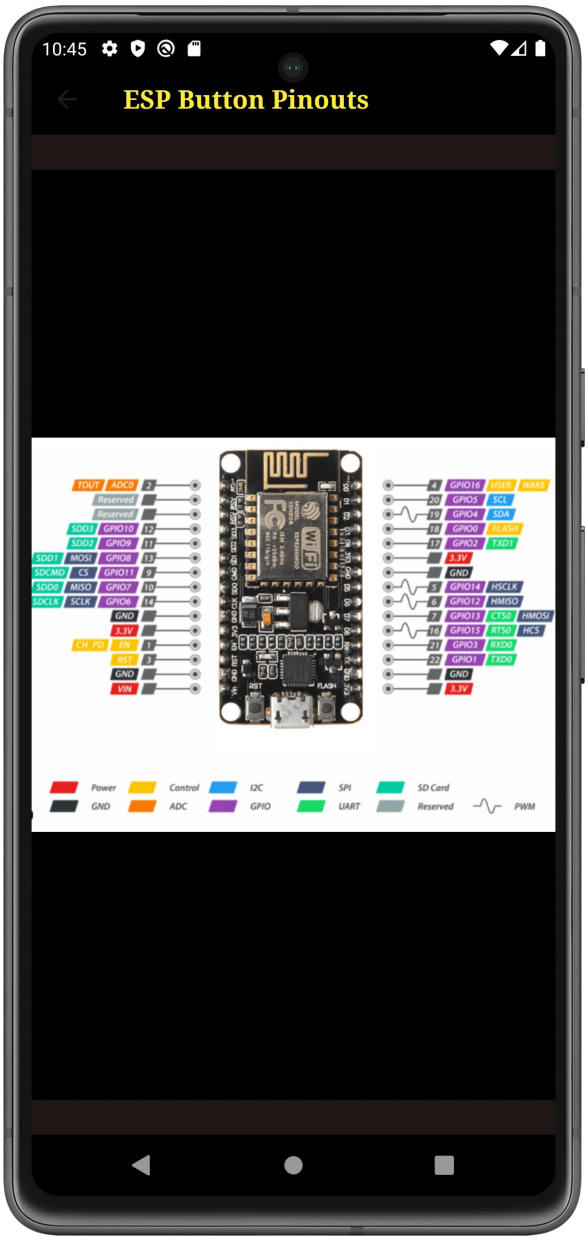


Figure 25: GPIO Pinout

This page shows the GPIO pinout of the Esp Micro controller

2.5.7. Switch Analysis

Figure 26: Switch Analysis D0, D1

Figure 27: Switch Analysis page D4, D5

Switch Analysis page display the total time of the running button, in figure 26, it shows that the D0 button is Turn On for 16 hour out of 24 hours. Which means that this light is turn on for overall 16 hours in different time of the day. The timer will increase when the button is green and it will show on the graph, the white pointer points out the time of the button on the dial.

2.5.8. Bar Graph Analysis for Buttons

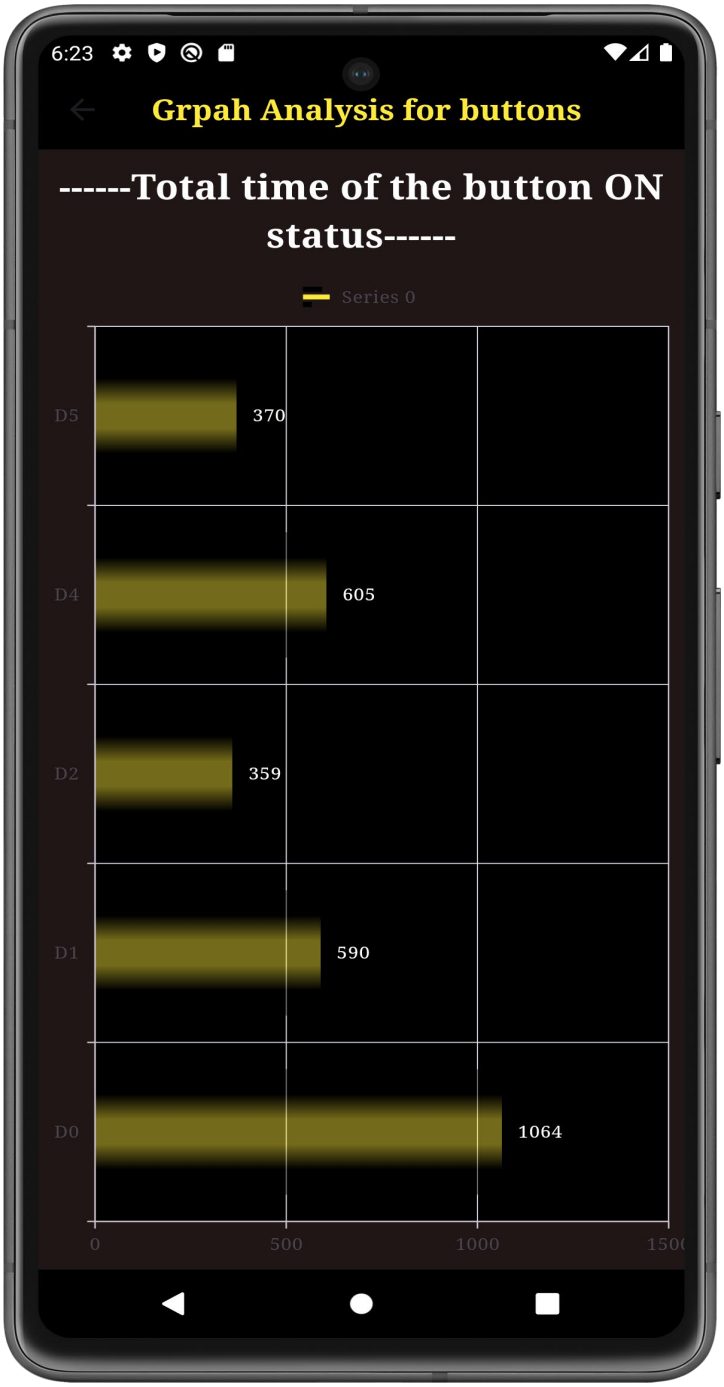


Figure 28: Bar Graph Analysis

In this graph, it shows the total timing of all the running button, this graph shows the timing of each button in minutes. In one day, there is 1440 minutes which means 24 hrs. The graph will display the running time of one day and it will reset to 0 once the 24 hrs is finish, it will start analysing the data for the next day.

As you seen in figure 27, it shows the button running time in hours but this graph shows the timing in minutes.

2.5.9. About Us Page

Figure 29: About Us Page

This is about us page which shows the logo of the Switchify app and explanation of the app, and describe the key features of the app

2.5.10. Hardware configuration with mobile phone and ESP

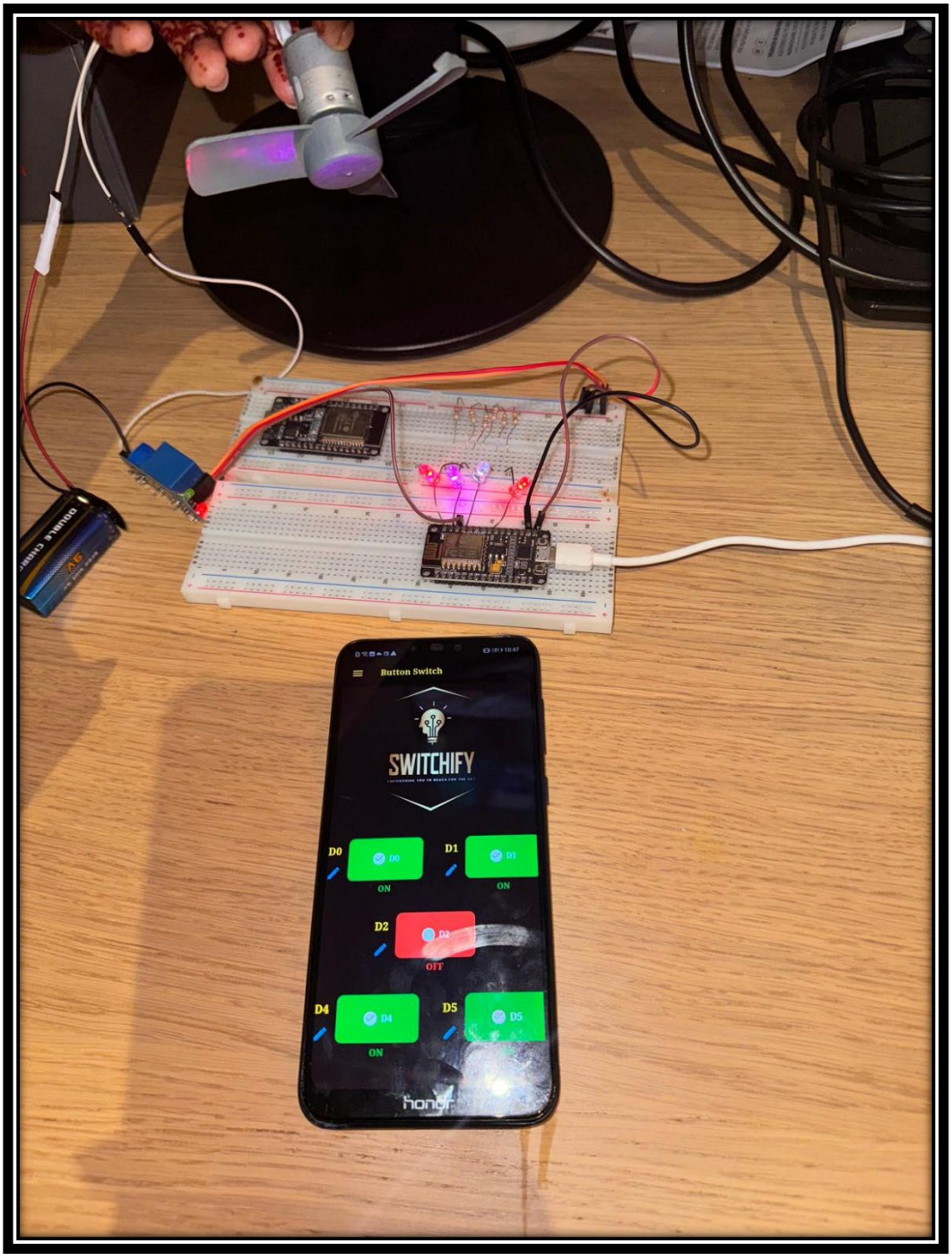


Figure: 30

- In this figure, you can physical android phone, which have the app downloaded and once you click on the button, it will Turn on the LED lights and Fan.
- The ESP microcontroller is connected to breadboard, and the led lights are connected to breadboard with ESP GPIO pinout.
- The fan is connected is to relay, because it requires higher voltage that's why we connected the relay and battery to turn on the fan

2.6. Testing

Test Case ID	Test Objectives	Test Steps	Expected Result	Test Outcome
T1	Login Functionality	1. Enter valid username and password 2. Click on the login button 3. Verify navigation to the home screen	User successfully logs in	**Pass**
T2	Real-time data synchronisation	1. Update data in Firebase database 2. Verify real time synchronisation in the app	Data is updated in the firebase	**Pass**
T3	Command Execution	1. Send command from mobile app to ESP 2. Verify successful execution in hardware	Command is executed by ESP micro controller	**Pass**
T4	Security /Authentication	1. Attempt login with invalid credentials 2. Verify access control to sensitive data	Authentication error message displayed Unauthorized access prevented	**Pass**
T5	LED Bulb Control	1. Turn on/off LED Bulb	LED bulb turns on/off as per command	**Pass**
T6	Fan control via relay	1. Send command to turn on/off via mobile 2. Verify voltage supplied to fan	Fan turn on/off via relay Voltage matches specifications	**Pass**
T7	Integration with Firebase	1. Send data from mobile app to Firebase 2. Verify data integrity in Firebase	Data is stored and retrieved successfully	**Pass**

Test Table 1:

2.5.1. UNIT TESTING

Unit testing allows us to catch bugs and issues I our code at the rly stage of development process. Writing unit testing for different functionality of your code encourage you to follow best practice of coding and leads to better overall code quality. Unit testing verify the behaviour of the individual part of code, you can make changes to your codebase and quicky identify if any existing functionality is broken. IT helps you isolates issues to specific unit of code and making it easier to find the cause of failures, this can save time during debugging and troubleshooting.

Here are some of the Unit testing for my Switchfly app, in terms of functionality

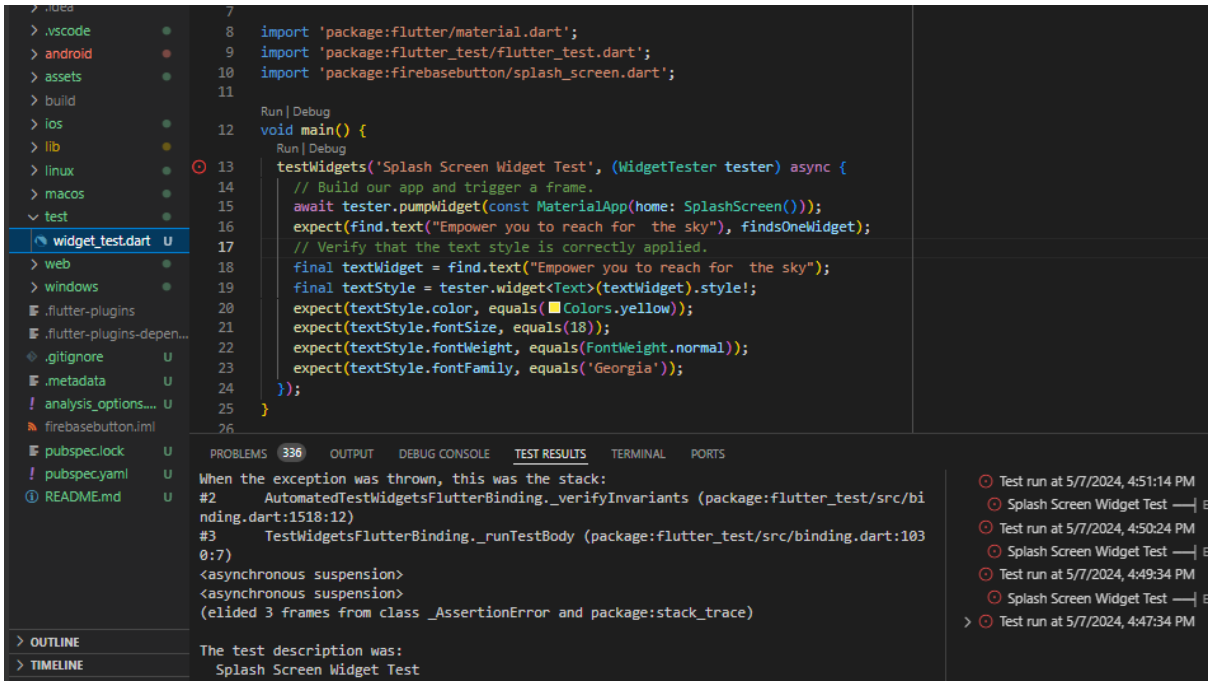


Figure 31:

Unit Testing of the splash screen, this test verifies that the splash screen displays the expected image and text with the correct styling.

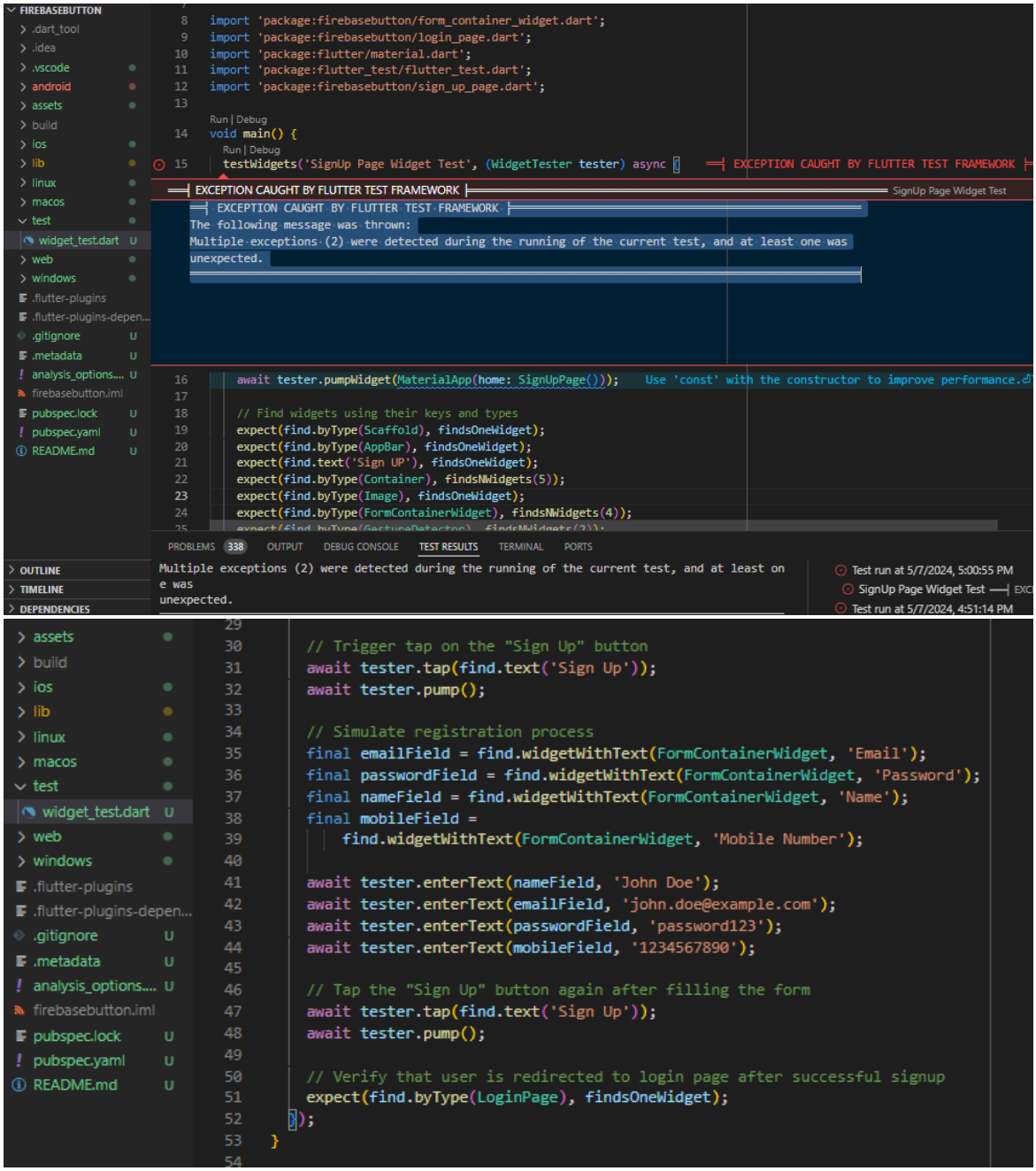


Figure 32:

Unit Testing of the signup page, it throws exception, checks for different features and functionality of the sign up page. In this test, we verify that all the necessary widgets are present on the SignUpPage, simulate a registration process by filling the form fields, and ensure that the user is redirected to the login page after successful signup.

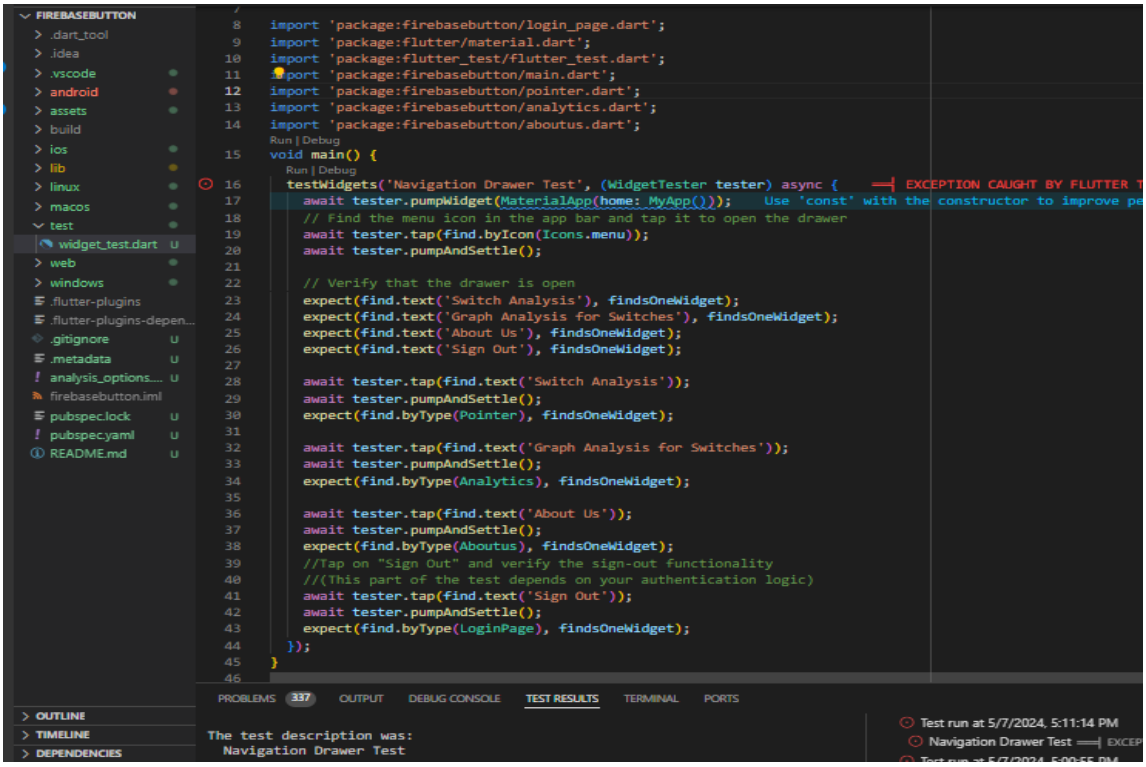


Figure 33:

Unit Test: Navigation Drawer, in this test, we load the myAPP by using widget which then looks for the menu icon on the app bar to open the drawer followed by the tapping each item on the drawer to confirm that it leads to the right routes. In this particular scenario, we are checking the behavior of the navigation drawer in the app by emulating user interactions (taps) and ensuring that the right routes are navigated to as a result. This is very important in order to avoid misuse that the navigation function will be working correctly. See figure 33

2.5.2. System Testing

System testing is the stage of software development during which the whole application is checked to ensure that it is working properly and in real world conditions it meets all the requirements. It's about setting up different examination positions for verification of the app performance, functionality and security. System testing is in fact functionality tests to ensure that all the key features run in full cycle, performance test to make sure that the responsiveness and execution of the system is not affected, and security vulnerability checks to find and fix those security issues. The system testing is the ultimate aim to ensure that the application is strong, stable and security conscious before the users can get to use it.

Test case ID	Description	Expected Outcome	Actual Outcome	Result
1	**End to End Functionality** Verify user can log in to the application Test turning on a smart light LED	User should successfully login Light should turn on	Valid credentials entered, logged in Light is on/off	Pass Pass

2	**Performance Testing** Measure Time taken to load app Measure response time for device control	App should load in 3 secs Control command should execute within 2 sec	App appears in 3 sec Command executed in time	Pass Fail
3	**Security vulnerability assessment** Attempt unauthorized access to data	Access should be denied	Access Denied	Pass

Test Table 2:

2.5.3. Integration Testing

For the Smart Home Automation System case, integration testing will be put into place in order to confirm whether the components are working together as they are supposed to, such as the mobile app, Firebase database and ESP microcontroller. The test cases will make sure that the dat synchronization between the mobile app and Firebase is working correctly and all the changes made in one database are accurately copied to the other database. In this respect, the function is also equivalent of system integration testing which provides a guarantee that the ESP microcontroller can link up with Firebase to send and receive data. This allows testing of integration between the modules that makes it possible to reveal problems and incompatibilities early when the system is still in process of development and the product has chance for success and endorsed by its consumers.

Test case ID	Description	Expected Result	Actual Result	Result
1	Mobile app with Firebase	Data from mobile app should synchronize with firebase on the button pressed	Changes made in the app reflect in Firebase database	Pass
			Changes made in Firebase database reflect in the app	Pass
2	Mobile app with ESP micorcontroleer	ESP should establish connection with Firebase	ESP successfully connects to Firebase	Pass
			ESP fails to establish connection with firebase	Fail

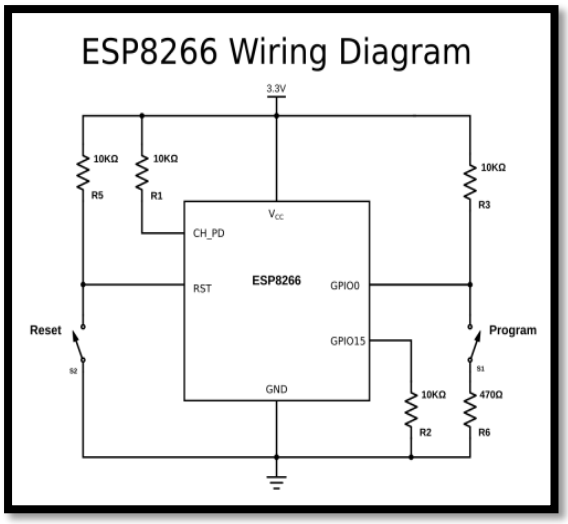
Test Table 3:

2.5.4. Hardware Testing

Test case ID	Description	Expected Outcome	Actual Outcome	Result
1	Device Connectivity	Device should be connected and responsive	All devices are connected successfully	Pass
2	LED BULB Control	LED Bulb should respond to commands from the app	LED bulbs turn on/off according to app commands	Pass
			LED bulb fails to respond to app commands	Fail
3	Relay Fan Control	Relay should regulation power supply to the fan	Fan turn on/off smoothly	Pass
			Fan fails to turn on/off	Fail

Test Table 4:

Circuit diagram (Wiring Diagram) for hardware testing of the LED bulb control



Explanation

The ESP microcontroller has a lot of GPIO pins available, usually with labels like GPIO0, GPIO1, GPIO2, and so on. These pins can be programmed on the ESP interactively through using the software.

The ESP Board connects to the LED Bulb via the circuit.

The ESP32 board sends control signals to the LED bulb for turning it on or off.

The LED lamp luminates in vivid colors in response to a control signal sent by the ESP board, and turns on or off as commanded.

GPIO pins can be configured as digital outputs for the following purposes: control external devices, like LEDs, relays, and motors.

The GND pin is the grounding point for both the ESP board and the LED bulb, thus closing the loop.

This circuit schematic will show how the ESP32 micro-controller works with an LED bulb to drive the lamp operation. It is a simple block diagram of the similar hardware setup that was used for testing of the same feature as the LED bulb control feature in the smart home automation system.

User Acceptance Testing:

In the context of the Smart Home Automation System, UAT involves actual users interacting with the mobile application and hardware components to assess its usability, functionality, and performance. During UAT, users will perform tasks such as logging in, controlling devices, setting up automation scenes, and monitoring the system's status. They will provide feedback on the user interface, ease of navigation, responsiveness, and overall user experience.

2.7. Evaluation

- **Usage data Analysis:** user interaction with the app, including login and sign-up frequency, button control command, timing of the button on state, all were monitored and analysed. This data provided insights into user preferences that helped us to refine the app features and functionality
- **Performance Evaluation:** The app's performance was evaluated in terms of responsiveness, loading times, and overall speed. Tests were conducted to measure the time taken for the app to load, process commands, and synchronize data with the backend server.
- **Correctness Assessment:** The correctness of the app's functionality was verified through extensive testing of its core features, including user authentication, device control, and automation scenes. Tests were designed to detect and address any bugs, errors, or inconsistencies in the app's behaviour.

The evaluation found that the Switchify app was not only usable, reliable, and problem free but also able perform its intended purpose. Users shared high levels of satisfaction relative to the app’s easy to use interface, seamless device manipulation, and the predicted promptness. Performance tests showed that the app can handle concurrent user sessions and devices connectivity without significant change in speed or responsiveness. In addition, this has been a key point in which analysis of usage data was performed and provided some important insights for me about user behaviour patterns and feature usage, which will help me when updating future versions of the app. Lastly, the evaluation results proved that Switchify was capable of finding the objectives it has set for itself and the makes the smart home automation solution reliable and easy to use.

Evaluation Metric	Value	Interpretation
App loading Time	3 sec	Fast
Device Control Response	2 sec	Very Responsive
User Login Response	5 sec	Medium
Firebase Sync Time	1 min	Acceptable

Result Table 5:

3.0 Conclusions

Advantages
• Remote Accessibility: Switchify gives users the option to manage the GPIO states by switching them on/off remotely, allowing them control over devices while having an internet connection • Real time Updates: Integration with Firebase facilitates the delivery of instantaneous GPIO state updates, so that users have a current feedback on the condition of the managed devices. • User Authentication: Authentication system makes only the verified people to gain the control on GPIOs improving the overall security of the system. • User Friendly Interface: The mobile app interface offers the users a simple way to interact with the GPIO controls which makes it accessible even to those users who lack in technical expertise.

Disadvantages
• Dependency on Internet Connection: Because Switchify uses a remote access and real-time updated features which both requires the internet connection, any interruption in internet connectivity will impact its operation. • Security Risks: Although user authentication might be seen as the most obvious security measure, the risk of unauthorized access or data breaches is still there, especially if passwords or other authentication methods are hacked. • Reliability on Firebase: The fact that Firebase is the hosting and database management system implies that if there are any problems or outages in Firebase services, then they will also affect the availability and performance of the Switchify system.

Strength	Limitation
• Integration with Modern Technologies: Switchify utilizes innovative engineering technologies for example Firebase, ESP32, ES8266, and various mobile development frameworks to create an all-encompassing and simple to control IoT solution. • Flexibility: The system's modular design enables seamless customization and adaptation to varying scenarios, both in terms of use cases and environment. • Real-time Feedback: Users get an immediate reaction on their actions which makes the process enjoyable and users also feel they are the ones that control the console as well as its responsiveness.	• Cost: Because the hardware parts of the Switchify system are inexpensive or free to use, a user may not be subject to most costs. Nevertheless, such hardware costs and cloud services may become relevant to larger-scale deployments. • Complexity for Beginners: While Switchify may be tricky to set up and configure for individuals with limited exposure to IoT and mobile development.

4.0 Further Development or Research

Enhanced Security Measures: The further stage of the developmental efforts consists of the deployment of modern security measures to strengthen the system's defences against threats. The solution could encapsulate the integration of additional authentication methods like two-factor authentication, encryption of valuable data and a self-audit for the security protection against the potential vulnerabilities. Moreover this approach must also include investigating commands for safe data transfer between the application and ESP32/ESP8266 devices, like SSL/TLS operator, in which case end users' personal communication and privacy will be enhanced.

Integration of analytics and machine learning solutions becomes the next research angle that the Switchify system will explore. We can get all sort of information about user engagement with the system through data collection and analysis. Knowing frequency of people's visits, other devices they use and their behaviour becomes much easier. This data can prove beneficial for extracting deep insights about the system in order to enhance system performance, give custom user experiences, and make recommendations ahead of time on energy efficiency and device administration. In addition, machine learning algorithms could be used to relieve the workers of their time-consuming tasks like predictive maintenance, anomaly detection, and adaptive device control, consequently raising the level of intelligence and the autonomy of IoT ecosystem.

5.0 References

- Syncfusion. (n.d.). *Customizable Flutter Chart Axis*. [online] Available at: https://www.syncfusion.com/flutter-widgets/flutter-charts/chart-axis [Accessed 3 May 2024].

- Dart packages. (n.d.). *syncfusion_flutter_gauges | Flutter package*. [online] Available at: https://pub.dev/packages/syncfusion_flutter_gauges#about-syncfusion [Accessed 4 May 2024].

- Anon, (2021). *ESP32: Getting Started with Firebase (Realtime Database) | Random Nerd Tutorials*. [online] Available at: https://randomnerdtutorials.com/esp32-firebase-realtime-database/#esp32-read-data-firebase [Accessed 02 March 2024].

- accounts.google.com. (n.d.). *Sign in - Google Accounts*. [online] Available at: https://console.firebase.google.com/project/fir-button-5ddf8/database/fir-button-5ddf8-default-rtdb/data [Accessed 4 Feb 2024].

- Android Developers. (n.d.). *sdkmanager | Android Studio*. [online] Available at: https://developer.android.com/tools/sdkmanager . [Accessed 30 November 2023].

- www.youtube.com. (n.d.). *Firebase Authentication in Flutter 2023 | Sign up & Login | Tutorials | #firebase #flutter* [online] Available at: https://www.youtube.com/watch?v=YVYIPIow418

- www.youtube.com. (n.d.). *Why ESP32's Are The Best Microcontrollers (ESP32 + Arduino series)*. [online] Available at: https://www.youtube.com/watch?v=A5CB4t9sukM&t=19 s [Accessed 28 February 2024].

6.0 Appendices

This section should contain information that is supplementary to the main body of the report.

6.1. Project Proposal

Project Proposal

Switchify

IOT Smart Home Automation System

(Mobile Application)

29/10/2023

BSHCSD4

Software Development

Academic Year i.e. 2023/2024

Nimrah Memon

x20738271

x20738271@student.ncirl.ie

Contents

6.1.1. Objectives

The objective of this project is to design a Smart Home Automation System. The aim of this project is to build an Mobile application which can control lights through wireless connection from Wi Fi. I will be using ESP32 micro controller which will receive signal from mobile app.

A mobile application will be developed that will connect seamlessly with an ESP micro-controller, allowing the users to remotely manage their smart home devices. By doing so we provide homeowners with a powerful tool to reduce energy wastage, lower electricity bills. Also, this app will be helpful for disabled people who won't be able to do home task easily. This app will help them control their home system from a mobile app from anywhere.

Also, this app will be helpful for people who are autistic, they can control the level of lightening from their mobile app. The purpose of this project is to make people's life easy and make their daily task easier.

Main objectives are

- Connect home appliances with ESP micro controller
- Development of an Mobile App: User-friendly app that connect with ESP micro controller. The primary focus is home appliances.
- Wireless control of lights: user can remotely turn lights on and off, adjust the brightness.
- Accessibility for disabled individuals: offer a level of independence who may face challenges in performing everyday home task.
- Support for individuals with Autism: they can control the level of lighting according to their comfort and wellbeing.
- Cost Reduction: user can make informed decisions about lighting and energy consumption; this will contribute to cost savings.
- Improved quality of life: create a comfortable environment, make peoples lives easier and simplify daily tasks.

6.1.2. Background

I choose to do his project is to make a meaningful impact on people's lives by leveraging technology to address specific needs and challenges. There are several reasons that motivate me to pursue this project

One of the primary motivations was the opportunity to create a solution that improves accessibility for individuals with disabilities. It is a well-known fact that technology can improve they quality of life for people with disabilities. And this project aligns with that mission. By developing an Mobile app that allows users to remote home control. We can empower disabled people to independently manage their living environment.

This project also serves to make a positive difference in the lives of individuals with autism. Autism often comes with a heightened sensitivity to environmental factors like lightning. Our system's ability to control lighting levels can help create a comfortable home environment, reducing stress and improving wellbeing of individuals.

Lastly this project represents an opportunity to explore how technology can seamlessly integrate into everyday life. This project allows me to showcase how technology can improve and simplify daily task. Home automation system ix becoming increasingly general, whether it's turning lights on and off, adjusting their brightness, we are making technology a part of user's daily routines.

This project offers a unique and compelling opportunity to develop home automation system that serve multiple purposes. It aligns with my passion for technology and my commitment to creating solution that improve daily task easier. The idea of this is a combination of my personal interest, technological trends, and a desire to address specific needs and making positive difference in users' life.

To meet the objectives my approach involves a combination of strategies and design features. We will develop Mobile app that will interface for the smart home system. The application will offer users the convenience of wireless control over their home appliances remotely.

6.1.3. State of the Art

There are several applications that are currently available in the market. These system offers remote control of home devices. For example, Philips Hue: Phillips offers a range of smart lighting solutions, allowing users to control the colour and brightness of their lights through a mobile app, another example is Google Home and Amazon Alexa, these voice activated systems enable users to control various smart home devices with voice commands.

My projects stand out in several ways

- **Accessibility**: while many smart home systems offer features that can be adapted for accessibility, my project place a primary emphasis on accessibility from the outset. It is specifically designed to serve to the need of individuals with disabilities, also individuals on the autism spectrum. The system interface and controls are designed with accessibility in mind, make it more user friendly for those with special needs.

Control over the wifi

- **Affordability**: the use of ESP32 micro controller distinguishes my project by its affordability. Many existing smart home systems are relatively very expensive. The project aims to provide a cost-effective solution that is accessible to a large user including those who may be on a budget.(cheaper solution , socket need no specific product required).
- **User Interaction Analytics:** The project incorporates advanced analytics features that enable users to gain insight of their energy consumption of appliance usage. User can make informed decisions about their energy usage which will leads to greater efficiency and cost savings.

These differences set the project apart from similar work by others, these features will make it a distinctive and valuable contribution to the field of smart home automation.

6.1.4. Technical Approach

For my project I will be using agile methodology, where I will outline the requirements and scope of the project. This step will involve identifying problem an outline requirement that needed to be solve. After that I will follow by the design phase which will focus on the designing of the application to implement the requirement that's been specified in the last step. Once the design phase is done then I will start coding to build an app. after that i will do the testing part which aims to ensure that are all the requirements meet the expectation not. If anything goes wrong, I will be able to go back and

change few requirements because agile methodology gives you opportunity to go back and make changes to it.

- Develop mobile application: development of mobile app with a focus on user friendly control and accessibility features.
- Create a UI Design for the app: creation of a user interface design that prioritize ease of use and accessibility.
- Create firebase database: set up and configure of the firebase database to store user data, device configuration and analytics information.
- Connect Hardware ESP controller with relay socket: the connection of ESP micro controller with relay socket to enable wireless control of lights, ensuring real-time responsiveness and reliability .

Each sprint will have defined milestones and activities such as completion of specific features or task. Continuous testing and will be integral to each sprint. After each sprint I will reflect in the sprint's success and areas of improvement. This continuous improvement process ensures that the project remain on track and user cantered.

In conclusion, In the development of Smart Home automation system, I will adopt an Agile methodology, which provides a flexible framework which is well-suited for my project dynamics. In summary our agile development approach of the project, adapting users' needs and throughout the development process. The advantage of sprint planning and milestone-driven progress to ensure that the smart home automation system is developed with strong focus on accessibility, usability, and reliability.

6.1.5. Technical Details

Language and principal libraries.

Development language: the development language for the mobile app will be Dart and the framework will be Flutter. Flutter allows for multiple platform development, creating a single codebase that runs on several devices for example, Desktop, web, Android, and iOS. It is known for its robust UI capabilities and efficient development process.

Development Environment: Android studio will be primary IDE (integrated Development Environment) used for developing Mobile application. It provides a comprehensive set of tools for android tool development using Flutter and dart

Database management: firebase will be used for database management, firebase offers real time data synchronization, and it is well suited for application that require data storage, user authentication and secure data transfer.

Wireless communication: the Wi-Fi library for the ESP32 will be used to establish wireless communication between mobile app and micro controller. This library provides the functions for connecting the ESP32 to a wi-fi network and handling data transfer.

Network communication and socket programming: network communication between the mobile app and the Esp will involve socket programming. This will allow data to be exchanges over a network connection, enabling the mobile app to send commands to the ESP controller.

Algorithm and approach

Wireless lighting control: the system will use the Wi-Fi connection to send commands to mobile app to the ESP micro controller. Socket programming will be employed to manage the data transfer which ensure the control of lights.

Data analytics: algorithm for data analysis will be used to provide users with insight into their energy consumption patterns. These involve trend analysis, visualization techniques.

6.1.6. Special Resources Required

Hardware Resources:

- ESP micro controller: ESP32 micro controller will be needed to control lighting system in various part of home
- Relay Socket: These are necessary for controlling electrical appliances including lights and these will integrate with the ESP micro controller.
- Light Bulb and Led lights: These will be used in the testing and implementation of the lighting control system.
- Led buzzer: This may be used for signalling and notifications within the system.

Software Resources:

- Flutter course and Dart: Training course in Flutter and dart.
- Firebase: The firebase platform will be used for database management, storage and data synchronization.
- Git/; version control with Git for managing and tracking code changes.
- Android studio: the IDE is necessary for Android app development using Flutter.
- Laptop: Required a suitable development machine for coding and testing.
- Android smart phone: Android devices will be required for testing the mobile application to ensure compatibility and usability.
- Wi-Fi Network: a stable secure network Wi-Fi required for the successful operation of the system.

Libraries

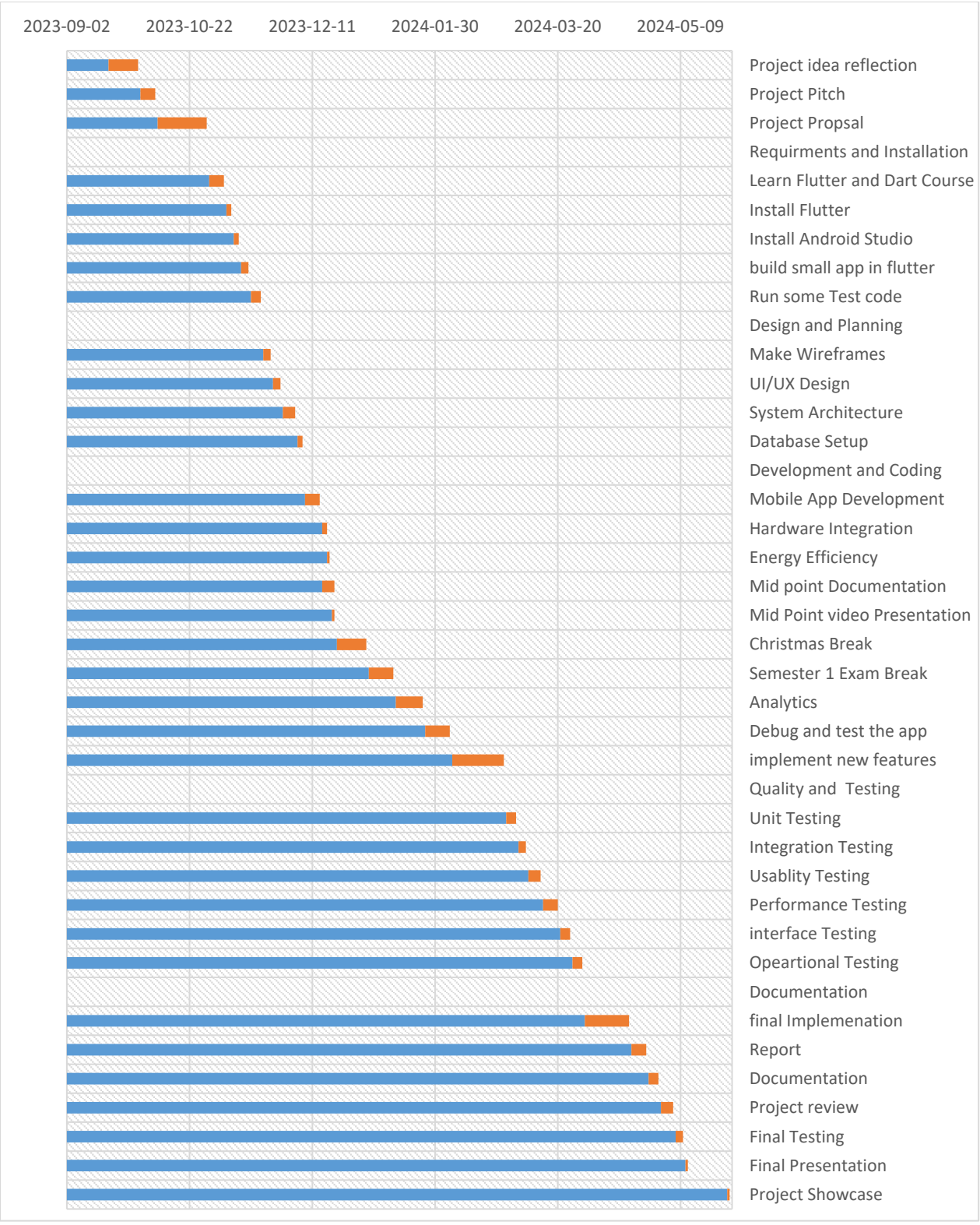
- Firebase Arduino client Library for ESP8266 and ESP32
- Firebase_core.dart
- Firebase_database.dart
- Firebase Auth
- Dart:core
- Flutter SDK
- WIfi Library ESP8266

6.1.7. Project Plan

TASKS	START Date	END Date	DAYS
Project Pitch and Project Proposal			
Project idea reflection	2023-09-19	2023-10-01	12
Project Pitch	2023-10-02	2023-10-08	6

Project Proposal	2023-10-09	2023-10-29	20
Requirements and Installation			
Learn Flutter and Dart Course	2023-10-30	2023-11-05	6
Install Flutter	2023-11-06	2023-11-08	2
Install Android Studio	2023-11-09	2023-11-11	2
build small app in flutter	2023-11-12	2023-11-15	3
Run some Test code	2023-11-16	2023-11-20	4
Design and Planning			
Make Wireframes	2023-11-21	2023-11-24	3
UI/UX Design	2023-11-25	2023-11-28	3
System Architecture	2023-11-29	2023-12-04	5
Database Setup	2023-12-05	2023-12-07	2
Development and Coding			
Mobile App Development	2023-12-08	2023-12-14	6
Hardware and circuit design	2023-12-15	2023-12-17	2
Hardware Integration	2023-12-17	2023-12-18	1
Mid-point Documentation	2023-12-15	2023-12-20	5
Mid-Point video Presentation	2023-12-19	2023-12-20	1
Christmas Break	2023-12-21	2024-01-02	12
Semester 1 Exam Break	2024-01-03	2024-01-13	10
Analytics	2024-01-14	2024-01-25	11
Debug and test the app	2024-01-26	2024-02-05	10
implement new features	2024-02-06	2024-02-27	21
Quality and Testing			
Unit Testing	2024-02-28	2024-03-03	4
Integration Testing	2024-03-04	2024-03-07	3
Usability Testing	2024-03-08	2024-03-13	5
Performance Testing	2024-03-14	2024-03-20	6
interface Testing	2024-03-21	2024-03-25	4
Operational Testing	2024-03-26	2024-03-30	4
Documentation			
final Implementation	2024-03-31	2024-04-18	18
Report	2024-04-19	2024-04-25	6
Documentation	2024-04-26	2024-04-30	4
Project review	2024-05-01	2024-05-06	5
Final Testing	2024-05-07	2024-05-10	3
Final Presentation	2024-05-11	2024-05-12	1
Project Showcase	2024-05-28	2024-05-29	1

Gantt Chart


2023-09-02
2023-10-22
2023-12-11
2024-01-30
2024-03-20
2024-05-09
Project idea reflection
Project Pitch
Project Propsal
Requirments and Installation
Learn Flutter and Dart Course
Install Flutter
Install Android Studio
build small app in flutter
Run some Test code
Design and Planning
Make Wireframes
UI/UX Design
System Architecture
Database Setup
Development and Coding
Mobile App Development
Hardware Integration
Energy Efficiency
Mid point Documentation
Mid Point video Presentation
Christmas Break
Semester 1 Exam Break
Analytics
Debug and test the app
implement new features
Quality and Testing
Unit Testing
Integration Testing
Usablity Testing
Performance Testing
interface Testing
Opeartional Testing
Documentation
final Implemenation
Report
Documentation
Project review
Final Testing
Final Presentation
Project Showcase

6.1.8. Testing

- **Unit Testing:** I will perform unit testing to verify the functionality of components. Such as mobile app, database operations, and ESP32 micro controllers. This testing will help me identify and address software bugs and hardware issue at an early stage.
- **Integration Testing:** it will be conducted to verify that different system components work together. The testing includes the interaction between the mobile app and ESP32 controllers, with firebase and communication over the Wi-Fi network.
- **UI (user interface) Testing:** this testing is to ensure that the user interface is user friendly. It involves verifying the elements like buttons, menus functions and that UI design aligns with the project's objectives.
- **Response Testing:** this will includes evaluating the system's response time. It ensures that user interactions are smooth and responsive.
- **Performance Testing:** The system's performance will be evaluated to ensure that it can handle multiple requests within time frames. The testing includes the real time control of lights and responsiveness to user commands.
- **Usability Testing:** This will involve end users interacting with the system to assess the user interface. This will provide insights into how to build the system and identify areas of improvement.
- **Interface Testing:** The testing of menu options, buttons and navigation and flow of the application between activities.

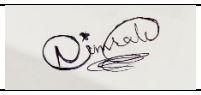
6.2. Reflective Journals

Supervision & Reflection Template

Student Name	**Nimrah Memon**
Student Number	**20738271**
Course	**BSHCSD4 [BSc. In Computing]**
Supervisor	**William Clifford**

Month: First [October]

What? In this month, I have a prepare a project pitch, and wrote a detailed project proposal that outlined the key component and objectives of the project . I done the research into further details in the project proposal of Smart Home automation system.
So What? The completion of project pitch and proposal gives me clear idea of the project. I meet up with my supervisor and discuss the objectives, the meeting was very beneficial and gives me foundation to start work on my project. the challenge I was facing was to how to implement the objectives that I have mentioned in the project proposal.
Now What? I need to do bit more research on my project and choose the suitable technology to implement the app. I will keep gathering feedback from the project supervisor to identify areas that need more clarification, and I will use this feedback to enhance the projects overall quality and alignment with expectation.

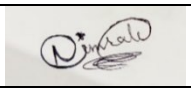
Student Signature	

Supervision & Reflection Template

Student Name	**Nimrah Memon**
Student Number	**20738271**
Course	**BSHCSD4**
Supervisor	**William Clifford**

Month: Second [November]

What? In the second month of the project, I had meetings with my supervisor where we discuss the project proposal in detail, and I got feedback and recommendations on the project proposal that I submit last month. In this month, I also done research on similar projects ideas and searching for tools that I will be using for this project.
So What? **Success**:

Downloads: Successfully download the Android studio, fixed all the errors and version issues, and test some codes to see output. **Feedback Integration:** Actively incorporating feedback received from the project supervisor has proven beneficial. Made adjustment to the project plan and technical specification were made to meet the projects alignment with objectives **Challenges:** **Technical Complexity:** having difficulty connecting the flutter with firebase. Integration between flutter and firebase revealed unforeseen challenges. Addressing these complexities required additional research and tutorial from online platform like YouTube videos. **Time Management:** balancing time effectively emerged as a challenge. The need for a more organised strategy and prioritise the task to overcome this challenge. As this month I get few other assignments so its gets harder to manage time.
Now What? **Regular progress Evaluation:** implement a systematic approach to evaluate progress regularly. This includes setting specific milestones, prioritise task and reassessing the project timeline based on ongoing developments. **Continuous Communication:** Maintain communication with the supervisor and seek feedback. Share progress update in weekly meriting and discuss challenges and seek input to ensure that the project stays on course. **Technical Problem solving** establish a dedicated time for problem solving session to address technical challenges. Engage with classmate who are using similar tools and explore online resources and seek guidance from the project supervisor to overcome challenges efficiently.
Student Signature 

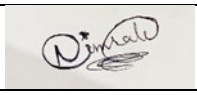
Supervision & Reflection Template

Student Name	Nimrah Memon
Student Number	x20738271
Course	BSHCSD4
Supervisor	William Clifford

Month: Third [December]

What? This month in the development of the IOT Smart Home Automation System, significant progress has been made in the initial stages of the project. The project requirements and technical approach have been prepared and listed out. Completed the login page and establish connection between a hardware device is bee made. The success lies in defining clear project objectives, outlining the technical approach, and specifying necessary resources.
So What? One notable success is the formulation of a comprehensive project plan, including a Gantt chart that outlines implementation steps and timelines. The structured plan serves as a roadmap for the project, enhancing coordination and ensuring that tasks are aligned with the project's goals. Challenges: However, challenges persist, especially in the technical approach and the choice of methodologies. While the agile methodology has been selected, the detailed implementation steps and task breakdowns still need further refinement. Addressing this challenge requires a more in-

depth consideration of development approaches, requirement identification methods, and effective task breakdowns. While outlining the functional requirements for the Smart Home Automation System, one notable challenge was ensuring the balance between comprehensiveness and simplicity. The system needed to cover a wide range of functionalities, from basic lighting control to more advanced features like automated scenes and energy management. Striking the right balance was crucial to avoid unnecessary complexity for end-users while providing a robust and feature-rich solution.
Now What? Moving forward, the focus will be on refining the technical approach through agile methodologies, particularly in identifying and breaking down requirements. Weekly discussion with supervisor and perhaps seeking external expertise can aid in overcoming these challenges. Additionally, continuous monitoring and adjustment of the project plan will be crucial to maintaining a dynamic and responsive development process. This reflective analysis will guide me towards addressing outstanding challenges and ensuring a more streamlined and effective project execution in the coming months.

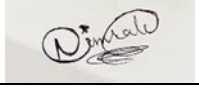
Student Signature	

Supervision & Reflection Template

Student Name	Nimrah Memon
Student Number	20738271
Course	BSHCSD4
Supervisor	William Clifford

Month: Fourth (January)

What? Progress made in this month is connect the database to the application and complete the login page and start the frontend of the main page and creating buttons and give styling to the button and make it more visible for the user who will use the app.
So What? So far, the login page is done, working on the frontend of the application creating buttons and navigation bar by following the wireframe that I created last month. Also try to do the backend and connect to the real tome database. Success: successful implementation of hardware configuration and connect the led bulb to see the state of led. Send code from Arduino to ESP controller and run the program to transfer the code to the device. Challenge: having issues with Realtime database connectivity with Firebase database, there is version issues that the versions are not supported.
Now What? Addressing challenges related to device commination and fixing the backend database to continue working on the project and ensuring that the application align closely with the need and expectations of its end users.

Student Signature	

Supervision & Reflection Template

Student Name	Nimrah Memon
Student Number	X20738271
Course	BSHCSD4
Supervisor	William Clifford

Month: Fifth (February)

What?

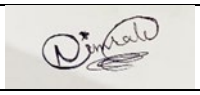
- In the fifth month of the Smart Home Automation System project, significant progress has been made in establishing the connection between the frontend and backend of the application.
- This involves creating a project in Firebase to integrate real-time database functionality into the system.

So What?

- Successfully implementing connectivity between the app and Firebase marked a major milestone in the project.
- It lays the foundation for application integration with a reliable and scalable database.
- However, this process is not without challenges, especially regarding version compatibility issues.
- To address this issue, upgrades have been made including updating the Flutter software as well as integrating core and authentication modules to establish strong connections.

Now What?

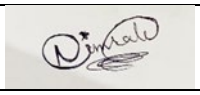
- To address remaining challenges, continuous testing and debugging is essential. Ensuring seamless communication between the app and the Firebase database is essential for overall system reliability.
- Regular updates and maintenance are required to stay consistent with the latest versions of the software and address any emerging compatibility issues.
- Additionally, this phase opens opportunities for further innovation and integration of advanced features, thereby improving the overall functionality of the smart home automation system.

Student Signature	

Supervision & Reflection Template

Student Name	Nimrah Memon
Student Number	X20738271
Course	BSHCSD4
Supervisor	William Clifford

Month: Sixth (March)

What? By the end of March, I included the application in the ESP microcontroller as well as the Firebase successfully. Further on, a number of Firebase auth issues and Realtime database difficulty was effectively managed. Firstly, the mobile app that controlled the LED light is one of our achievements that prove I made real progress.
So What? These improvements are the top events so far in the project's efforts to be successfully completed. Achieving the connection of the application with both the ESP microcontroller and the firebase database matters when you need a smart home automation device that works effectively. Facing a set of challenges arising from integration with Firebase, which is an authentication method and the real-time synchronization of data, shows my ability to solve problems and the degree of my determination. Moreover, however, issues persist; more specifically the issue of ensuring the connection between the application, ESP microcontroller, and Firebase database is reliable and effective. The establishment of smooth communication shall be a major point of interest, being the effective exchange of data between these components. This process requires extensive testing and fine-tuning. Moreover, we will be focusing on improving the quality of the app's user interface and the experience base that covers multiple user preferences and situations as one of my main tasks.
Now What? In order to ensure the success of the project, performing thorough testing of the integrate the system should be a priority in finding and listing out any flaws or inconsistencies. Testing user sessions totally will show a path for the app to make better in the functions and usability side.
Student Signature 

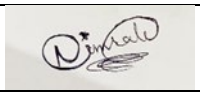
Supervision & Reflection Template

Student Name	Nimrah Memon
Student Number	x20738271
Course	BSHCSD4
Supervisor	William Clifford

Month: Last (April)

What? There have been a lot of noteworthy developments and project milestones this month. I successfully finished the testing phase, making sure that the application satisfies the required criteria in terms of functionality and usability. The completed architectural diagram, which offers a concise summary of the components and structure of the system, was another accomplishment. A demonstration of the application was conducted, showcasing its capabilities and features to supervisor. Hardware testing and configuration were also carried out to ensure seamless communication between the application and the physical devices.
So What?

The achievement of these tasks is clear evidence of the success of our project. The functionality test went well and our application and hardware operated according to plan, reflecting our aim in the project scope, that was to ensure the required application performance. The circular nature of the data flow diagram passes a clear understanding of the system hierarchy to easily communicate among the constituent devices. The experiment showed us the project's performance to the supervisor and we gathered the needed feedback. This recent milestone in the implementation phase of our new application brings us, therefore, getting one step close to the deployment of the product as well as its release for the users. Even though, some roadblocks standing in the way, such as preserving and perfecting some details by the software part and a full acceptance of the interconnected hardware pieces, remain.
Now What? In order to overcome the remaining problems, we will concentrate on the resolution of any issues that have not been cleared during the testing process. It can confuse with debugging code, performance optimization, and addressing the bug when users comment and provides a feedback during demo. Lastly, as we go on, we will not forget to do a rigorous hardware testing and configuration to complete compatibility and dependability, both. The cooperation of the supervisor and the classmate will be the main factor in the solving of these challenges to deliver our smart home automation system successfully.

Student Signature	

6.3. Invention Disclosure Form

1. Title of Invention

ESP Switchify (Wireless communication between mobile app and Hardware home Aplinaces)

2. Inventors

Nimrah Memon	National College of Ireland	Student	Lucan, Co. Dublin X20738271@student.ncirl.ie	70%

3. Contribution to the Invention

My role was key during the decision-making and planning, and also in the stages when the plans were being put into action. I carried out the assignment by doing research on IoT technologies, mainly ESP32 microcontrollers and the integration of the real-time database Firebase. On the other hand, I deployed as well the parts of the hardware, such as fine-tuning and coordinating Arduino code that determines the light bulbs, fans, and relays. It was dynamical process that resulted in working on the application and identification, troubleshooting, and improving user interface of the mobile app, and connection between application and hardware devices. My roles were very important for

the whole project to succeed in its goal of making a smart home automation system that was user-friendly and efficient.

4. Description of Invention

The ESP Switchify is an intelligent smart home automation system that gives the user access to remote controls and monitoring devices on their home via their mobile phones or computers. The invention uses ESP32 microcontrollers, Arduino code, and integration of the real-time database Firebase to make possible the effortless communication between the mobile application and the hardware devices like LED bulbs, fans, and relays. The innovative feature of the software is the Firebase implementation that continuously synchronizes all the devices, enabling the users to view and manipulate pertinent information from their mobile application.

5. Why is this invention more advantageous than present technology?

Resources provided by the Switchify demo Allow us to highlight many of its features and advancements over the existing home automation systems. New features like real-time synchronized, are also installed which ensure seamless communication, updating and execution in fraction of time. This point tackles the problems of delay and latency in the traditional systems, which is eliminated, and the user gets a smooth and natural human interaction. Besides, ESP Switchify's compatibility with Firebase reduces installation and setup time, hence, taking away the need for complex server infrastructure, in turn making the platform scalable and stable.

6. What is the current stage of development / testing of the invention?

The ESP Switchify is currently in the implementation and testing phase, with hardware components configured and integrated with the mobile application. Testing is being conducted to ensure reliability, performance, and user-friendliness.

7. List the names of companies which you think would be interested in using, developing, or marketing this invention

- Smart home device manufacturers
- Home automation solution providers
- Technology companies specializing in IoT and connected devices.

8. Where was the research carried out?

The research and development of the ESP Switchify were carried out at home.

9. What is the potential commercial application of this invention?

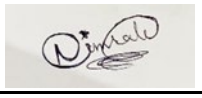
The ESP Switchify has potential commercial applications in the smart home market, offering homeowners a convenient and customizable solution for controlling and monitoring their home devices remotely.

10. Is a model or prototype available? Has the invention been demonstrated practically?

A prototype of the ESP Switchify has been developed and demonstrated practically to showcase its functionality and capabilities.

I/we acknowledge that I/we have read, understood and agree with this form and the Institute's *Intellectual Property and Procedures* and that all the information provided in this disclosure is complete and correct.

I/we shall take all reasonable precautions to protect the integrity and confidentiality of the IP in question.

Inventor: Nimrah Memon  10/05/2024

signature Date

Signature