Smart Life Application

By

Name: Tonmoy Baishnob
ID: 1830295
Spring, 2023

Supervisor:

Mr. Abu Sayed

Lecturer

Department of Computer Science & Engineering

Independent University, Bangladesh

June 01, 2023

Dissertation submitted in partial fulfillment for the degree of Bachelor of Science in Computer Science

Department of Computer Science & Engineering

Independent University, Bangladesh

Attestation

Based on the tasks that I was given during my internship, I, Tonmoy Baishnob (ID: 1830295), attest that this report titled "Smart Life Application" represents my work. I have submitted this report in part fulfillment of the requirements for the Independent University of Bangladesh's (UB) Computer Science and Engineering degree. Under the guidance of Mr. Abu Sayed, my university supervisor, it was finished. Additionally, I have given credit to all of the information and sources used in this research.

Additionally, I certify that I have not plagiarized any work done by other students or people and that I have not previously submitted this report for any evaluation in any other class. However, this work adheres to the widely accepted academic standard that all references to the written works of others must be properly acknowledged.

Signature Date

Acknowledgement

First and foremost, I want to thank Almighty God for His mercy and blessings that have allowed me to get this far. Being an intern with Flow Technology has been a fantastic privilege. From the staff members of the organization who have years of experience, I have gotten so much support and inspiration. Additionally, I would like to express my gratitude to the Flow Technology members for contributing their time and expertise, both of which were crucial to the preparation of this report.

My appreciation goes out to Mr. Abu Sayed, my academic supervisor at the Independent University of Bangladesh (IUB), Department of Computer Science and Engineering, for his important advice, ongoing direction, support, and inspiration throughout my internship and the writing of this report.

Also deserving of my gratitude is Md. Jahid Hasan (CTO at Flow Technology), the mentor and supervisor of my organization, for his unwavering support and direction during the project's final stages.

Finally, I humbly express my gratitude for my family, friends, and their kind wishes, moral support, insightful counsel, motivation, and inspiration.

Tonmoy Baishnob

June 1, 2023

Letter of Transmittal

02 June 2023

Mr. Abu Sayed

Lecturer

School of Computer Science and Engineering

Independent University, Bangladesh

Subject: Letter of Submission for the Internship Report

Dear Sir,

With all due respect, I, Tonmoy Baishnob, ID -1830295, from Section 1 of the internship course for the Spring 2023 Semester, would like to submit my Internship report. This report is based on the project I worked on and my internship program. My internship with Flow Technology, which ran from February 18 to May 25, is already complete. This report is based on my work and experiences I had while an intern at Flow Technology. My internship's main objectives were to familiarize myself with the software engineering field, learn about the various technology-related departments at the company, such as research and development, documentation, and software development, and gain experience working in the field. I had to study and adjust to the changing technologies utilized in various scenarios and requirements throughout my internship at Flow Technology in order to be able to employ them in actual projects.

I will be very thankful if you can read this report and offer your insightful opinion on it. I would be grateful if you found this report to be insightful and helpful in gaining knowledge about the subject at hand.

Sincerely,

Tonmoy Baishnob

ID - 1830295

Email: tanmoystui008@gmail.com

Evaluation Committee

....

Signature

....

Name: **Ajmiri Sabrina Khan**

....

Panel Member-1

....

Signature

....

Name: **Dr. Ashraful Islam**

....

Panel Member-2

....

Signature Asayed

....

Name: **Md. Abu Sayed**

....

Supervisor of the intern

....

Signature

....

Name: **Dr. Mahady Hasan**

....

Head, Department of Computer Science & Engineering

Abstract

Project Name: Smart life

This project is a mobile application, which is connected to IOT devices. The App will show real time data from the devices and control the devices and switches. The app will show real time temperature as well. The app shall contain the following features to get data from the microcontroller using a cloud database. The app can be controlled from anywhere with the data connection. created to give users easy access to control over a variety of smart devices, including Light, Fan, Wi-Fi, AC, cameras, TVs, and more. By providing seamless device administration and control through a single interface, the program seeks to improve user experience.

The application gives customers the ability to control devices remotely, including the ability to change settings, choose modes or channels, and switch them on or off as well as collect media. which improves energy effectiveness and ease. Users receive helpful information on the state of their devices from real-time device monitoring. The application enables scalability for future expansions and ensures interoperability with a wide range of devices. Prioritizing security and privacy ensure secure authentication and data protection. This application's goal is to provide a seamless and effective user experience for controlling smart devices by fusing the flexibility of Flutter with the IoT's capabilities.

Table of Contents

Acknowledgement iii

Letter of Transmittal iv

Evaluation Committee v

Abstract vi

Table of Contents vii

List of Figures ix

Introduction 1

1.1 Overview/Background of the Work 1

1.2 Objectives 1

1.3 Scopes 2

Literature Review 3

2.1 Relationship with undergraduate studies 3

2.2 Related work 4

Project Management & Financing 5

3.1 Work Breakdown Structure 5

3.2 Process/Activity wise Time Distribution 5

3.3 Gantt Chart 6

3.4 Process/Activity wise Resource Allocation 6

3.5 Estimated Costing 7

Methodology 8

Body of the Project 9

5.1 Work Description 9

5.2 Requirement Analysis 9

5.2.1 Rich Picture 11

5.2.2 Functional and Non-Functional Requirements 11

5.3 System Analysis 15

5.3.1 Six Element Analysis 15

5.3.2 Feasibility Analysis 16

5.3.3 Problem Solution Analysis 17

5.3.4 Effect and Constraints Analysis 18

5.4 System Design 19

5.4.1 UML Diagrams 19

5.4.2 Architecture 22

5.5 Implementation 23

5.6 Testing 24

5.6.1 Input 24

5.6.2 Output 24

5.6.3 Designing Test Cases 24

5.6.4 Test Results 24

Results & Analysis 25

Project as Engineering Problem Analysis 27

7.1 Sustainability of the Project/Work 27

7.2 Social and Environmental Effects and Analysis 28

7.3 Addressing Ethics and Ethical Issues 29

Lesson Learned 30

8.1 Problems Faced During this Period. 30

8.2 Solution of those Problems 30

Future Work & Conclusion 31

9.1 Future Works 31

9.2 Conclusion 31

List of Figures

Figure 2.1: Tuya 4
Figure 2.2: Blink 5
Figure 2.3: Blynk 5
Figure 3.1: Work Breakdown Structure 6
Figure 3.2: Grantt Chart 8
Figure 4.1: Agile Methodology 10
Figure 5.1: Flutter 12
Figure 5.3: Rich Picture 14
Figure 5.4: Use Case Diagram 29
Figure 5.5: Activity Diagram 30
Figure 5.6: ERD 31
Figure 5.7: Home page 32
Figure 5.6: Login page 32
Figure 5.9: Bedroom page 32
Figure 5.8: Login page 32
Figure 5.12: Kitchen page 33
Figure 5.10: Living Room 33
Figure 5.11: Dining Room 33
Figure 5.13: Bathroom page 33
Figure 5.14: Garage 34
Figure 5.15: User authentication page from the Firebase 34
Figure 5.18: ESP8266 setup 35
Figure 5.17: ESP8266 setup 35
Figure 5.16: Total storage page from the Firebase 35
Figure 5.19 & 5.20: The live demo of device switch. 36

List of Table

Table 3.1: Process/Activity wise Time Distribution 7
Table: 3.2: Process/Activity wise Resource Allocation 8
Table 3.3: Approximate Cost Estimation. 9
Table 5.1: Functional Requirement 1 16
Table 5.2: Functional Requirement 2 17
Table 5.3: Six Element Analysis 24

Chapter 1

Introduction

1.1 Overview/Background of the Work

Several gadgets, including TVs, air conditioners, cameras, and Wi-Fi networks, are monitored and managed by an IOT-based Flutter application. Customers may efficiently manage and communicate with their connected devices from a single, centralized platform provided by this service. The program uses the Internet of Things (IoT) to interface with the devices seamlessly. Users can do this to control and keep an eye on their devices from a single location. The following gives an overview of how a typical user of this application could feel:

Device Integration: Using the target devices' specific communication protocols, the application integrates with them. For instance, it can connect to the TV by Wi-Fi or infrared, the AC via Bluetooth or IP, the camera via Wi-Fi or IP, and the Wi-Fi router using its specific management protocol. These gadgets can establish connections with the program and communicate with it.

User interface: Using the Flutter application's user-friendly UI, users can manage and control their gadgets. The user interface is designed to be intuitive to use and aesthetically pleasing, including features like device status monitoring and control switches.

Device Control: Using the program, users can send commands to connected devices to complete a variety of tasks. They can control all linked devices' switches. There is a Master Switch that a user can use to turn on or off the lights and other gadgets in the entire area.

Real-time Device Feedback: The software can display real-time device feedback, such as the temperature or camera feed.

In general, an IoT-based Flutter application offers a smooth and unified control platform for a variety of devices, giving consumers ease of use, remote accessibility, and the capacity to change device interactions to suit their individual needs and preferences.

1.2 Objectives

The major objective of this project is to employ a mobile application to make life easier for people and save energy. Users of the Smart Life program can manage their living or working environment. Another important goal of this application is energy conservation. Users can turn off all the switches from the application to save electricity if they fail to turn off any equipment before leaving the room. Users essentially can control their devices remotely, including turning them on and off and changing settings. The software eliminates the need for several device-specific apps or physical remote controls by acting as a universal remote. Users get access to notifications and live information on the state of their devices, their energy usage, and other crucial details. Users may find their patterns of device usage with the aid of the application, which can offer insights and analytics, and then make wise decisions to reduce energy consumption or change device settings. The application's straightforward and user-friendly interface allows users to navigate and engage with it fast. The interface includes straightforward controls, a comprehensive list of all connected devices, and a clear presentation of device status and feedback. Last but not least, it places a great value on safeguarding user data privacy as well as how they interact with technology. User data can be protected from unwanted access or misuse by implementing strong authentication procedures, secure data storage, and secure data transport.

1.3 Scopes

The app's capabilities include the ability for users are:

- Remotely operate devices and the ability to see the current temperature on the homepage.
- Users can manage multiple devices at once.
- Users will have access to an aesthetically pleasing interface.
- The user will have secure authentication methods to guarantee that only authorized users can access the application and related devices.

Chapter 2

Literature Review

2.1 Relationship with undergraduate studies

The development of the “Smart Life” application has benefited from the knowledge and abilities learned in undergraduate courses. It would have proven more difficult if these courses were not covered before working on this project. Some of the courses are:

Introduction to Computer Programming (CSC101): This course teaches and provides the necessary information related to the introduction to a new programming language and the basics of programming from declaring variables, iteration, conditional statements, switch-case, arrays, and functions.

Object Oriented Programming (CSE213): The course goes a step ahead by introducing the concept of classes and objects, variables inside the classes, types of classes. A Graphical User Interface of a real-life problem.

Database Management (CSE303): This was the first course which taught how to design and plan a project. It covered popular planning and strategy practices such as System Development Life Cycle, Rich Picture, Requirement Analysis, Relationship Diagram, Business Process Model and Notation Diagram and many more. These techniques helped in the development planning and strategy of this project.

System Analysis and Design (CSE307): This course introduces the tools and techniques to design a system with proper analysis of information systems. Systems and models, project management, tools for identifying system requirements, data flow diagrams, six element analysis, feasibility Analysis, UML diagrams are some of the subjects covered.

Web Application & Internet (CSE309): This course introduces a comprehensive overview of technologies related to the web and their uses. The discussions in the course are held on front-end languages: HTML and Cascading Style Sheet (CSS) and backend languages: PHP and MySQL followed by the responsiveness for devices with different sizes by jQuery.

Software Engineering (CSE451): This course taught me about Software Development Life Cycle (SDLC). So that while I am making a project or developing software, I can follow all the rules properly. The SDLC defines and explains a comprehensive plan with stages, or phases, each of which includes its own procedure and outputs. Following the SDLC reduces project risks and expenses and speeds up development while increasing the efficiency of production which is helpful and useful.

2.2 Related work

The project that I am doing for my internship is a remotely IOT device controlling app, although there are other IOT device controlling apps and sites on the internet. Here are a few of the examples that are provided.

Tuya Smart: An app for smartphones called Tuya Smart was created to control and manage electronic appliances in the home. Users can Control and manage smart home devices. Users can connect and control a variety of smart home devices, including lighting, security, and temperature control, create automation scenarios, Monitor, and receive alerts, Schedule device operation, Use voice commands.

Figure 2.1: Tuya

Blink Home Monitor: Amazon Blink is a home security camera and video doorbell system that comes with a mobile app to remotely control and monitor the system. Users can Set up and configure their Blink devices, live stream video and audio from their cameras, create and manage multiple Blink systems, share access to their Blink system with family and friends, use voice commands to control their Blink devices with Amazon Alexa and Google Assistant.

Figure 2.2: Blink

Blynk IOT: With the help of the mobile app Blynk, users may construct unique user interfaces to manage and control linked devices in IoT projects. With drag-and-drop widgets, real-time data monitoring, remote device control, notifications, widget interactivity, cloud connection, and energy management, it delivers a user-friendly interface. The Blynk app makes it easier for individuals without a deep understanding of programming or networking to create and manage IoT projects. Also, the app is paid. The cost for using its advanced services is $999 per month.

Figure 2.3: Blynk

Chapter 3

Project Management & Financing

3.1 Work Breakdown Structure

The Work Breakdown Structure (WBS) is a technique for segmenting a project into smaller, simpler components. It is a visual tool that aids project managers in organizing and planning the activities required to meet the objectives of a project. The WBS develops a hierarchical structure by breaking the project scope into more manageable, smaller tasks and subtasks. Frequently, the project is divided into smaller components, which are subsequently divided into smaller, more specialized jobs. The project is typically broken down into more manageable components, starting with the primary objective of the project. The WBS, which is divided into tiers that reflect increasing information levels, gives a clear picture of what must be done to successfully complete the project's aims to make complex jobs simpler by dividing them into smaller, more manageable components. The team's capacity to work on multiple features at once boosts its productivity. We appear to apply a Work Breakdown Structure (WBS) in a top-down manner to some extent. We can quickly coordinate the entire project using WBS, and we can effectively protect job quality.

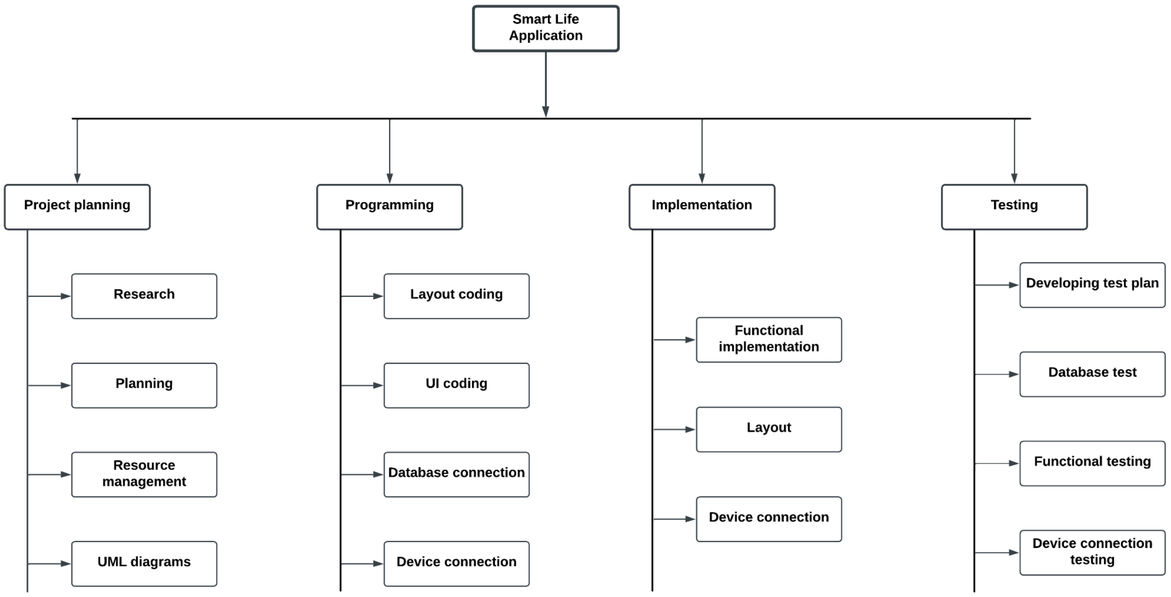


Figure 3.1: Work Breakdown Structure

3.2 Process/Activity wise Time Distribution

A list of all the tasks involved in the project is included in the WBS. We tried as a group to complete these tasks in the allowed time. Estimates state that the project should be completed in 90 days.

Activity	Start Date	End Date	Duration (days)
Planning	18-Feb	24-Feb	7
Requirement Analysis	25-Feb	6-Mar	10
Design	7-Mar	19-Mar	13
Implementation	20-Mar	2-May	44
Testing	3-May	18-May	16

Table 3.1: Process/Activity wise Time Distribution

3.3 Gantt Chart

One of the most common and effective ways to demonstrate activities presented against time is the Gantt Chart, a graphical representation of a project plan that is often used in project management. It helps to estimate the project's duration, identify the resources required, and schedule the activities' completion in order. The Gantt Chart displays the deadline for completing the Work Breakdown Structure activities. Monitoring the project's development once it has begun is also helpful.

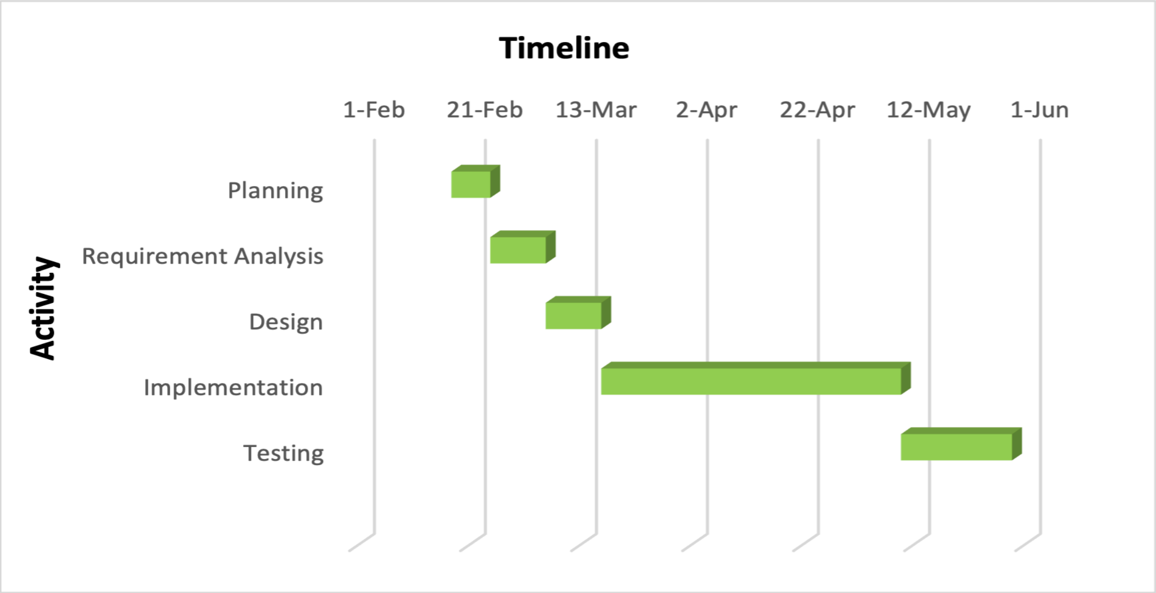


Figure 3.2: Grantt Chart

3.4 Process/Activity wise Resource Allocation

Activity	Start Date	End Date	Duration (days)	Percentage
Planning	18-Feb	24-Feb	7	7.8%
Requirement Analysis	25-Feb	6-Mar	10	11.12%
Design	7-Mar	19-Mar	13	14.45%
Implementation	20-Mar	2-May	44	48.89%
Testing	3-May	18-May	16	17.78%

Table: 3.2: Process/Activity wise Resource Allocation

3.5 Estimated Costing

Based on the company's needs for the system, an estimate of the cost has been made.

No	Requirements	Amount (BDT)
01	Internet Bill (3 Months)	3,600
02	Electric Bill (3 Months)	60,000
03	Hardware	1,00,000
04	Development (3 Months)	1,20,000
Total		2,83,600

Table 3.3: Approximate Cost Estimation.

Chapter 4

Methodology

Agile Methodology

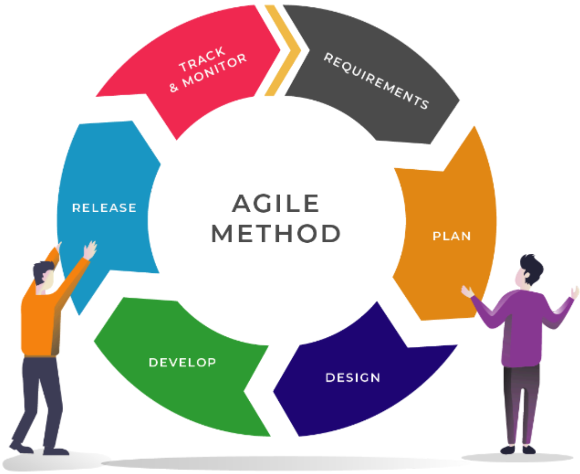


Figure 4.1: Agile Methodology

Agile methodology stresses teamwork, adaptability, and customer satisfaction in its incremental and iterative approach to software development. The early release of functional software is encouraged, along with adaptive planning and ongoing improvement. The main features of the Agile methodology are as follows:

- Iterative and Incremental Development: Agile projects are broken down into brief iterations or sprints, which typically last between one and four weeks. Every sprint entails the development, testing, and delivery of a working software increment that consists of a subset of features or user stories. Continuous feedback and improvement are made possible by this iterative process.
- The development team members collaborate closely throughout the process, fostering open communication and knowledge sharing. The team as a whole oversees the project's success.
- Customer Collaboration: Agile approach places a strong emphasis on working closely with the client or product owner throughout the course of the project. The client participates in the prioritizing of features, offers input, and evaluates

the work produced in each iteration. This makes it possible to guarantee that the provided software satisfies the client's requirements and demands.

- Planning that is flexible acknowledges that requirements might vary over time and that change is inevitable. Agile initiatives. Agile teams prioritize making a flexible and adaptive strategy rather than doing extensive upfront planning. Planning is done at the start of every iteration, allowing the team to react to shifting priorities and fresh insights as the project develops.

- Agile teams strive towards continuous delivery of usable software and frequent integration of code changes. As a result, any problems will be discovered as soon as possible, and stakeholders will be able to monitor the project's progress and offer input as it moves along. Practices for continuous delivery and integration serve to lower the risk connected to significant integration activities after the project's conclusion.

- Continuous Improvement: Teams are urged by the agile methodology to evaluate their operations and make ongoing improvements. To find areas for improvement and implement corrective measures, regular retrospectives are held. Agile teams have a mindset that encourages them to continuously learn, modify, and improve the methods they use to increase output and quality.

Therefore, I have followed agile methodology while developing the Smart Life Application. Because the Agile methodology allows development teams to provide software in a way that is more cooperative, adaptable, and customer focused.

Chapter 5

Body of the Project

5.1 Work Description

To develop the “Smart Life” app I used Flutter which is the framework of dart. So that Android and iOS users both can use it. I have created a flutter project on Android Studio and designed the UI of the project. And I connected Firebase as the backend. The Smart Life app is a function which can control the IoT devices, and it can control a whole room such as WIFI, AC, TV, temperature. By using this app people can control their home’s device. So, to take all the services User must install the app and sign up on the app. And then users can connect devices with the app and control them.

Figure 5.1: Flutter

Figure 5.2: Dart

Our goal is to:

- Create an app that is simple to use.
- Create the app with the least amount of friction.
- User friendly sign up and login process.
- Make a simple design so that the user finds it easy to use.
- Make runtime faster.

5.2 Requirement Analysis

Device Integration: Integration of Devices: The app must be interoperable with a wide range of IoT devices, including air conditioners, cameras, TVs, lighting, fans, and weather sensors. These gadgets need to be able to connect to one another and to control each other using well-known communication protocols like Bluetooth and Wi-Fi. To support specialized device features or acquire weather-related data, integration with third-party APIs or ecosystems may be necessary.

Device Control and Monitoring: From the app, users should be able to control specific devices or groups of devices, including turning them on or off, modifying parameters (such as temperature, fan speed, or brightness), and starting predefined actions or scenarios. The app should offer real-time device status monitoring, including data on things like battery life, temperature measurements, and camera feeds.

Interface and users experience: The app's UI should be simple to use and allow users to effortlessly switch between devices and access their control choices. Icons, labels, or color-coded indicators are good examples of clear and aesthetically pleasing displays of device status. Different user preferences and access levels could necessitate support for various user accounts, roles, or permissions.

Weather Integration: The app must include weather-related functionality, such as displaying forecasts and current weather information. Real-time weather data depending on user location or selected regions must be fetched through integration with weather APIs or services. Users may be able to establish weather-related triggers or rules, for example, automatically modifying thermostat settings based on temperature or turning on rain sensing for outside equipment.

Security and privacy: Strict security controls must be put in place to safeguard user information, device communications, and stop illegal access.
A secure data transmission protocol should be used, together with user authentication techniques. Concerns about privacy should be considered, together with measures to protect the privacy of individuals and ensure that data collection and use are transparent.

Analysis of energy use and efficiency: By providing information on device energy consumption and presenting suggestions for energy-use optimization, the app should encourage energy-saving behaviors. For customers to receive use patterns, energy efficiency statistics, or cost estimates, historical data collecting, and analytics capabilities can be implemented.

Performance and Scalability: The app should be made as scalable as possible, with low latency in device management and data retrieval. This will ensure responsiveness.
Without noticeably degrading in performance, it should be scalable to accommodate an increase in the number of users, devices, and data traffic.

Platform Compatibility: To reach a wider audience, the program should be created for a variety of operating systems, such as iOS and Android. Maintaining feature parity and a constant user experience across many platforms is essential.

Regulatory Compliance: In the regions where it operates, the app must abide by all applicable data protection, privacy, and IoT security laws. To guarantee data privacy and the security of user information, it should adhere to industry best practices and rules.

5.2.1 Rich Picture

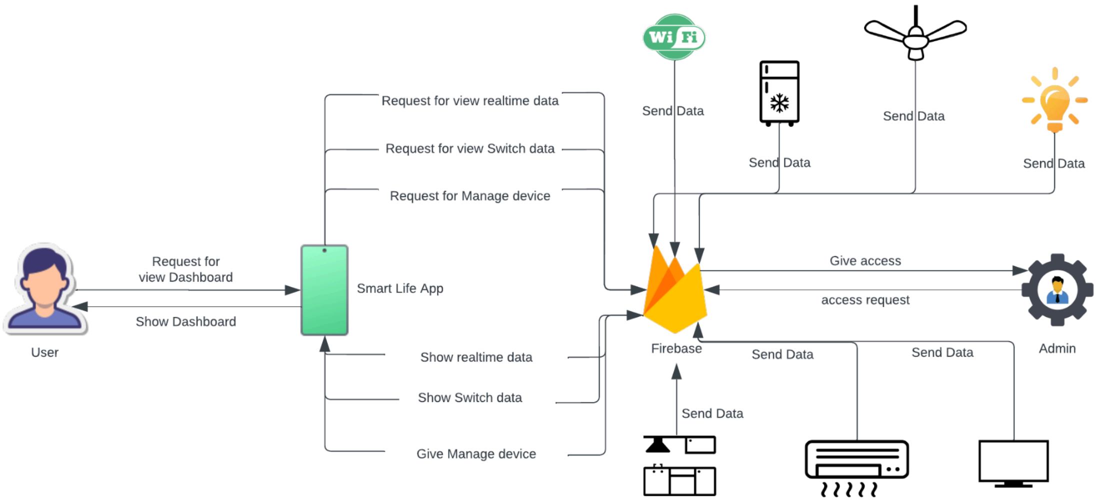


Figure 5.3: Rich Picture

5.2.2 Functional and Non-Functional Requirements

Functional Requirements

The functional requirement frequently defines a task that a system must complete. Features that must be present in an information system are known as functional requirements. Business requirements are satisfied in part by functional requirements, which also need to be user acceptable. The actions and tasks that a system must be able to carry out are known as functional requirements. Functional requirements specify the nature and purpose of an application's components as well as their nature and purpose. Using methods for gathering requirements, the following functional requirements were put together. Below is a discussion of the functional requirements' inputs, processes, outputs, preconditions, and postconditions:

Name of the Function: User can connect devices with the app to control		
Input: Enter user email and password. Select the room and view switches. Connect devices with the app and its switches to control them.	Process: Verify user email and password. Load room and switches. Find the selected device and switch. Store user data in the database.	Output: Successfully logged in. Display room and switches. The device has been successfully connected.
Precondition: The system should be familiar to the users. Users must be in possession of active accounts. Users must have an android device/IOS device.		
Post condition: Proper database connection. All details are recorded in the system. The system should return to the main option.		

Alternate Options: The system responds to the user if the user's email address is invalid. The system issues an error message if the password is invalid. The user system offers a new set up password option if the user forgets the password. If there is no information and data, then the system returns a message.
Side Effects: N/A

Table 5.1: Functional Requirement 1

Name of the Function: User can disconnect a whole room by turning off a single switch (Master Switch).		
Input:	Process:	Output:
Enter user Email and password.	Verify user Email and password.	Successfully logged in.
Select the room and view switches.	Load room and switches.	Display room and switches.
Click the Master switch.	Turn off all the switches.	System will display all the switches off in the selected room.

Precondition: The system should be familiar to the users. Users must be in possession of active accounts. Users must have an android device/IOS device.
Post condition: Proper database connection. All details are recorded in the system. The system should return to the main option.
Alternate Options: The system responds to the user if the user's email address is invalid. The system issues an error message if the password is invalid. The user system offers a new set up password option if the user forgets the password. If there is no information and data, then the system returns a message.
Side Effects: N/A

Table 5.2: Functional Requirement 2

Non-Functional Requirements

A project's non-functional requirement is a quality requirement. Based on factors including portability, security, usability, and responsiveness, the software system is evaluated. The following is a description of non-functional requirements:

Reliability: The system is regarded as reliable. This app is very simple to use for both admin and users. Users can clearly understand the software's interface and process because it is designed to be so straightforward.

Performance: The main goal of using complicated frameworks to develop this simple website is to increase performance. It must demonstrate and accommodate the user's needs. It talks about a satisfactory reaction time and throughput rate. When a website loads quickly, people are more likely to stay on it for longer, therefore this application should offer a seamless user experience to all users.

Security: Another prevalent category of non-functional requirements is security requirements. The system must guarantee that all data contained within the system, or a portion of it, will be safe from virus assaults and illegal access. Every piece of information is safe. Although the website only interacts with a small number of data, the architecture is up to date with security standards. The only individuals with access to the core code are designated developers and admins. Consequently, the system is secured.

Scalability: Scalability is a system's ability to continue operating under the most challenging conditions. The fact that our solution is portable and works with all operating systems, including iOS and Android, is a benefit.

Compatibility: Applications that work on mobile devices will also be created by us. Applications must function on iPhones with OS versions > 13.0 and Android smartphones with OS versions > 8.0.

Availability: The system must always be available to users.

Maintainability: The system must specify how long it will take to address a bug, improve a solution's functionality, or modify another element of it to adapt to a changing environment

5.3 System Analysis

5.3.1 Six Element Analysis

Process	Human	Non-computing hardware	Computing Hardware	Software	Database	Connectivity
Register new user to the Smart Life application	**Admin:** a. Select the sign-up option to register new user to the Smart Life application to take the services. b. Put all the required information of a new user to create the account. c. Click on the sign-up button	None	**Smart phone (Android, iOS)** a. Used by the user to install the Smart Life application to their Smart phone to control IoT devices and their switches. **Computer (Laptop, Desktop)** a. Used by the Admin to	Smart Life Application	**Firebase** a. Only admin has access to this and can manage all the data from the database .	**Internet** a. Used by the Admin to access the real time database and to manage it.

	after giving all the information to register a new user. **User** Give all the necessary and valid information to registration to the application.		control and monitor the real time database ‘Firebase’.			
Manage User of the Smart Life application	**Admin** a. Check all the users from the real-time database. b. Check all the user emails to see that they are valid or not. c. Delete unknown user if	None	**Smart phone (Android, iOS)** a. Used by the user to install the Smart Life application to their Smart phone to control IoT devices and their switches.	Smart Life Application	**Firebase** a. Only admin has access to this and can manage all the data from the database.	**Internet** a. Used by the Admin to access the real time database and to manage it.

	found any. **User** a. To take the services of the Smart Life application login with the valid email and password.		**Computer (Laptop, Desktop)** a. Used by the Admin to control and monitor the real time database ‘Firebase’.			
View the real time temperature from the Smart Life application.	**User** a. Login to the system with the valid email password. b. Go to the home page of the Smart Life application to view the real time temperature.	None	**Smart phone (Android, iOS)** a. Used by the user to install the Smart Life application on to their Smart phone to control IoT devices and its switches.	Smart Life Application	**Firebase** a. Only admin has access to this and can manage all the data from the database.	**Internet** a. Used by the Admin to access the real time database and to manage it.

Connect with the devices and the switches from Smart Life application	**Admin** a. Connect the devices with the real-time database through the ESP8266 code and provide all the necessary information for that user. **User** a. Give all the required information like personal Wi-Fi name and password, user email and password to connect the ESP8266 with the real time database.	**ESP8266** a. To connect the switches of the app with the real time database.	**Smart phone (Android, iOS)** a. Used by the user to install the Smart Life application to their Smart phone to control IoT devices and their switches. **Computer (Laptop, Desktop)** a. Used by the Admin to control and monitor the real time database ‘Firebase’.	Smart Life Application	**Firebase** a. Only admin has access to this and can manage all the data from the database.	**Internet** a. Used by the Admin to access the real time database and to manage it.

Control Light, Fan, WIFI, Camera’s switch	User a. Login to the Smart Life application. b. Select any particular room to control it’s switches. c. Turn on/off of the switches on basis on need.	**ESP8266** a. To connect the switches of the app with the real time database.	**Smart phone (Android, iOS)** a. Used by the user to install the Smart Life application to their Smart phone to control IoT devices and their switches. **Computer (Laptop, Desktop)** a. Used by the Admin to control and monitor the real time database ‘Firebase’	Smart Life Application	**Firebase** a. Only admin has access to this and can manage all the data from the database.	**Internet** a. Used by the Admin to access the real time database and to manage it.
Log out From the Smart Life	User a. Log out from	None	**Smart phone (Android, iOS)**	Smart phone (Android, iOS)	**Firebase** a. Only admin	**Internet** a. Used by the

Application	the application or once user close the app user will be automatic logged out from the Smart life app.		a. Used by the user to install the Smart Life application to their Smart phone to control IoT devices and its switches.		has access to this and can manage all the data from the database.	Admin to access the real time database and to manage it.

Table 5.3: Six Element Analysis

5.3.2 Feasibility Analysis

One of the four essential steps in the software project management process is a proof of concept. A feasibility study, as its name implies, evaluates a software product's viability in terms of the advantages that product development offers an organization from a practical standpoint. A feasibility study is a study that explains specific project elements and shows how they can advance the project's development. An analysis of the project's feasibility criteria aids in spotting potential issues. The following list includes the system's viability components that we observed at:

Operational Feasibility: Operational feasibility evaluates whether the program will function successfully after it is designed and depends on human resources. The operational feasibility carries out the following duties: Identifying whether a problem with user requirements is anticipated is of great priority. The Smart life application has been created in a way that makes it very simple to use. The public will without a doubt accept it. The system is appropriately thought out. It will be accessible to people of all ages. Users of this system don't require a lot of technical expertise. Users can comprehend every instruction with ease. We think that this system will be able to accommodate all user requirements.

Technical Feasibility: The development of a project is examined and evaluated in terms of the hardware, software, and technological resources that are available. This technical feasibility study provides information on whether the appropriate resources and technology required for use in project development are in place.

I have developed a Smart Life application using flutter, dart, firebase. This helps me to build an application which is much more efficient and faster. Moreover, these technologies are very popular in the modern industry and widely used by the growing community.

Economic Feasibility: One of a project's essential components is its economic viability because it offers a wealth of information about the project, including the site needed, the tools and supplies needed, the raw materials needed, and the labor force needed for manufacturing. To figure out the project's costs and benefits, this analysis is conducted. The price to construct an application has been evaluated in detail. It is economically beneficial for the company.

Scheduling Feasibility: The research examined the project's timetable to make sure it would be finished on time. The project will be accomplished if the deadline is met. The project was scheduled to be completed in 90 working days. The project will be completed on schedule if the specified milestones are met on time.

5.3.3 Problem Solution Analysis

- Constrained device compatibility

Ensure worldwide compatibility by integrating with widely adopted IoT protocols and standards, such as Wi-Fi. As a result, several devices, including TVs, air conditioners, cameras, and Wi-Fi enabled devices, may be connected to and controlled by the program.

- Difficult device setup procedure

Provide simple setup instructions and user-friendly user interfaces to make configuring devices easier. Use auto-discovery techniques to find and automatically connect with adjacent devices. To speed up the setup process, support well-liked device provisioning techniques like NFC tags and QR code scanning.

- Insufficient security measures

To safeguard user data and maintain the privacy of device control, incorporate security elements into the program. Adhere to established practices for safe IoT connectivity, implement authentication procedures, and encrypt data transmission. Update the program frequently to patch up any security flaws.

- Restricted control possibilities

The solution is to give users a wide range of control options within the application, including fundamental features like power-on/power-off, volume control, temperature management, and device status monitoring. To improve user convenience and control flexibility, support advanced capabilities like scheduling, automation, and custom scene creation.

- Limited ability to manage devices.

Include device management features in the program, such as device grouping, renaming, and organization. Give consumers the option to create profiles or rooms so they may effectively manage and control many devices at once. In order to facilitate proactive troubleshooting, further give real-time device status updates and problem notifications.

- Insufficient cross-platform support

Utilize the cross-platform capabilities of the Flutter framework to create the application for both iOS and Android. This will guarantee a greater audience and a constant user experience across various devices. To deliver a native-like experience, follow the best practices and rules unique to the platform.

- Lacking guidance and support.

To help consumers use and debug the application, provide thorough documentation that includes setup directions, troubleshooting advice, and FAQs. Additionally, provide quick responses to consumer questions and concerns through channels like email, chat, or forums.

5.3.4 Effect and Constraints Analysis

Effects

Efficiency and Convenience: Through this program, users may easily operate various devices from a single interface. Users may quickly change settings and check on the status of their smart devices, increasing management efficiency.

Enhanced User Experience: The program can provide a better user experience by providing real-time device status updates, intuitive user interfaces, and intuitive, frictionless, and pleasant device control.

Integration and Interoperability: The application may interface with a variety of devices from different manufacturers, promoting interoperability and lowering vendor lock-in. This is made possible by supporting major IoT protocols and standards.

Constraints

Device Compatibility: Due to varying communication protocols or a lack of support for common IoT interfaces, some devices might not be compatible with the program. The number of devices that the app can manage may be lowered as a result.

Privacy and Security: It's important to safeguard user data privacy and security as well as the operating instructions for devices. To guard against unauthorized access, data breaches, and the misuse of personal information, the application must have strong security features.

Network connectivity: If the network is down, the application cannot be used.
If there are problems with the internet connection, such as network outages or weak signal strength, users may experience restrictions or interruptions in device control.

User Learning Curve: To make the program's learning curve for users shorter, a user-friendly interface should be created. For customers who are unfamiliar with IoT principles or using mobile applications for device control, there may still be a little learning curve.

Software Updates and Maintenance: To maintain compatibility, security, and performance as the IoT landscape changes and new devices or communication protocols appear, the application will need regular updates and maintenance. The development team must continue to put in effort and resources in this regard.

5.4 System Design

5.4.1 UML Diagrams

1. Use case Diagram.

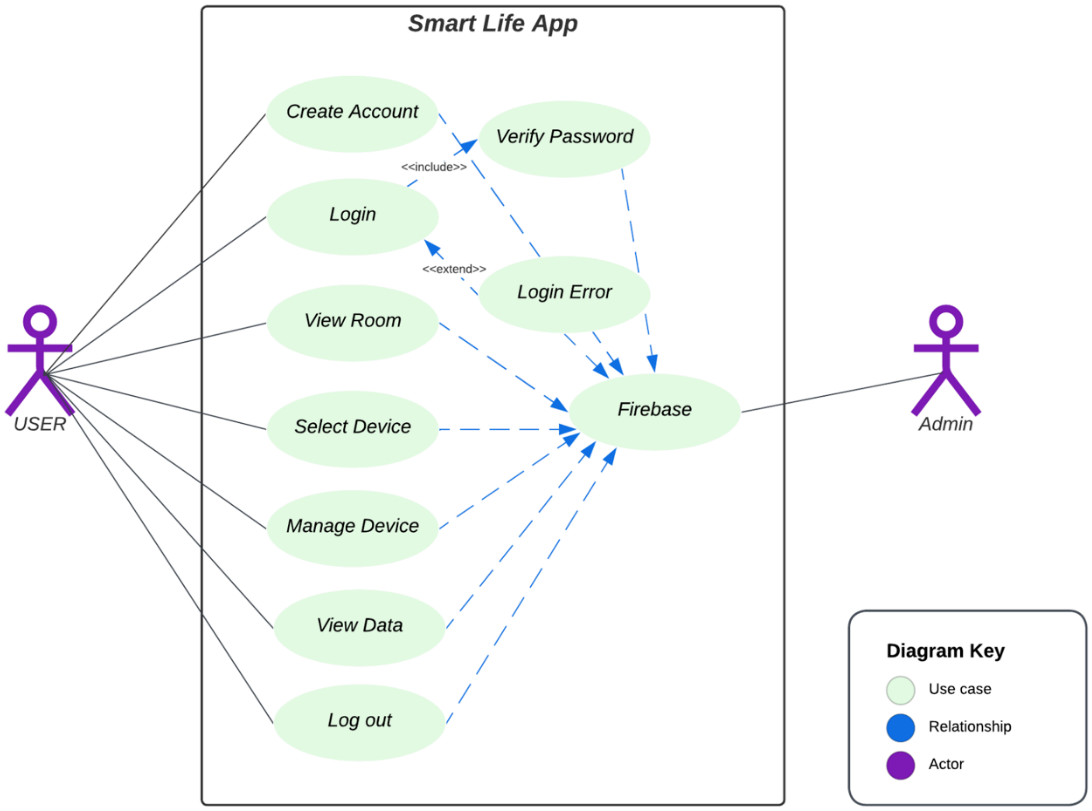


Figure 5.4: Use Case Diagram

2. Activity Diagram

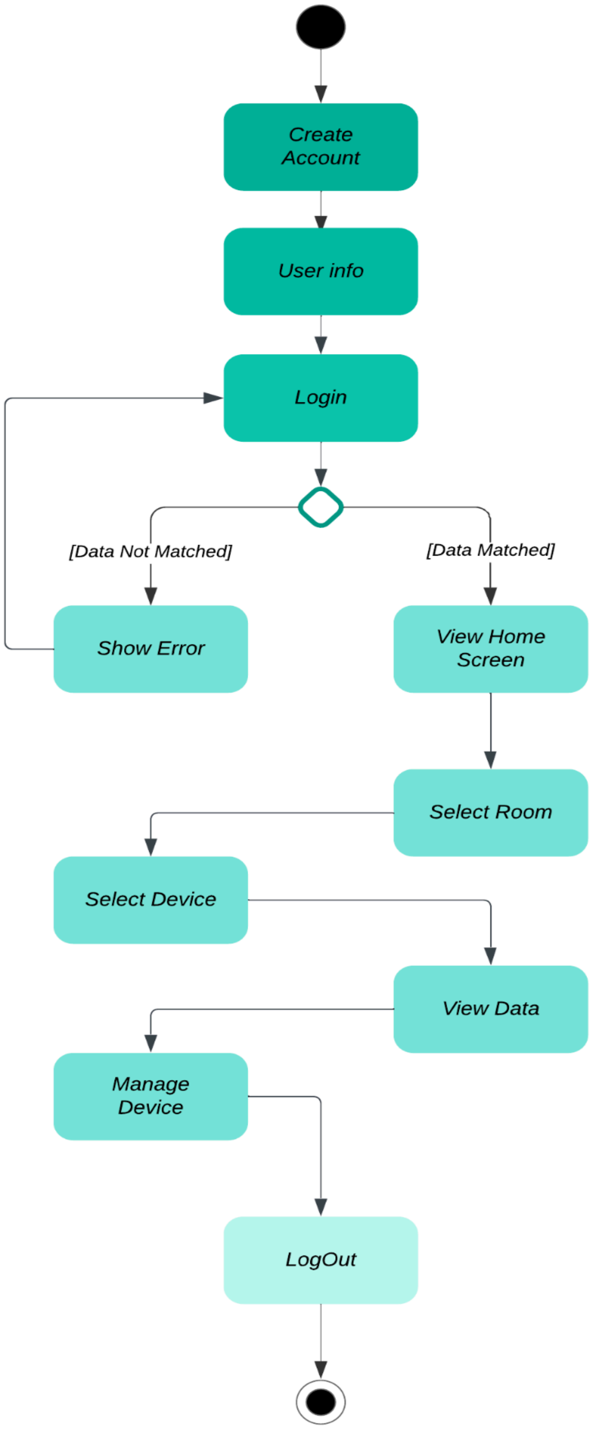


Figure 5.5: Activity Diagram

ERD

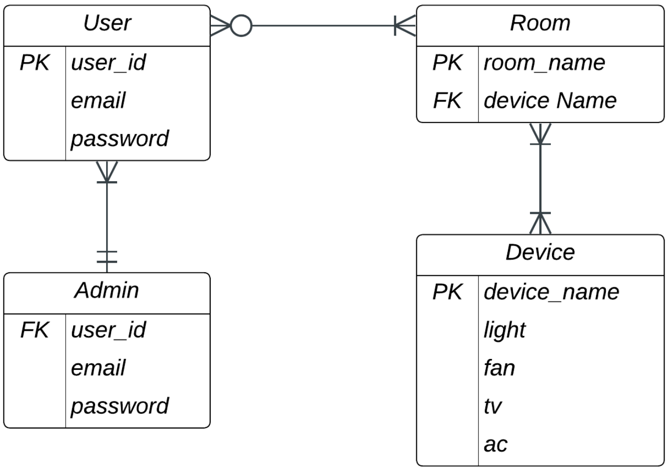


Figure 5.6: ERD

5.4.2 Architecture

User Interface: The front-end user interface was created using the Flutter framework, which offers a wide range of UI components and cross-platform interoperability.
• Components of the user interface: I created screens and widgets that let users manage and keep an eye on connected devices. This may involve switches, sliders, and buttons.

Application Backend: Cloud architecture: To control device connectivity, data processing, and user authentication, I set up a cloud-based backend architecture using Firebase.
• APIs and Web Services: To facilitate communication between the frontend and backend components, APIs and web services were developed. User queries, device commands, and data retrieval are all included in this.

Data management and analytics: • Database: Use a database system to store and manage user preferences, device data, and other pertinent details. This can comprise user profiles, device configurations, and previous data.

Privacy and security:
• Authentication and Authorization: To confirm user identification and manage access to the application and devices, I implemented secure authentication techniques.
• Access restrictions: Establish fine-grained access restrictions to make sure that only authorized users can manage devices and carry out permitted actions.
• Logging and Monitoring: I implemented logging and monitoring methods to follow the performance of devices and applications, spot mistakes or other anomalies, and make troubleshooting easier.

5.5 Implementation & Testing

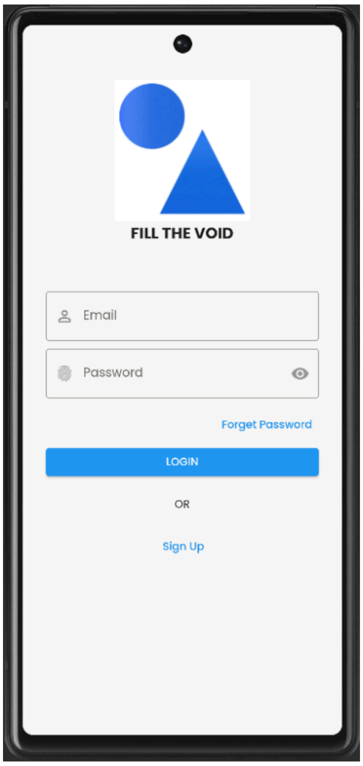


Figure 5.6: Login page

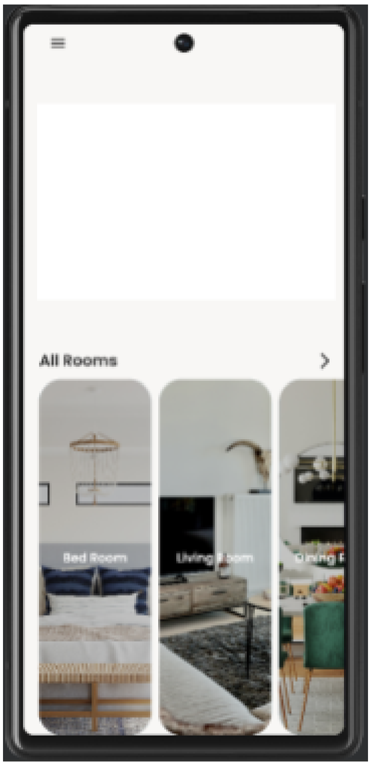


Figure 5.7: Home page

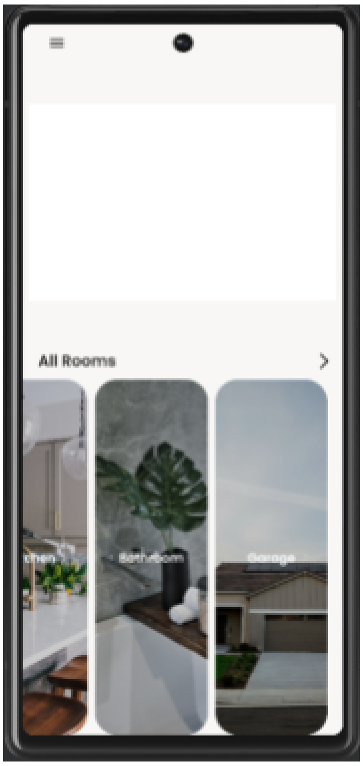


Figure 5.8: Login page

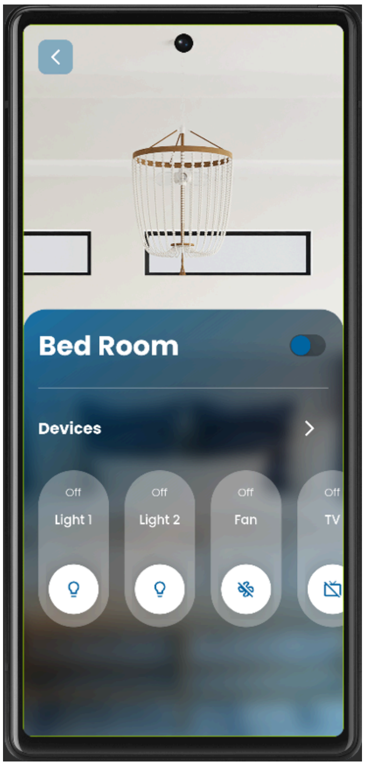


Figure 5.9: Bedroom page

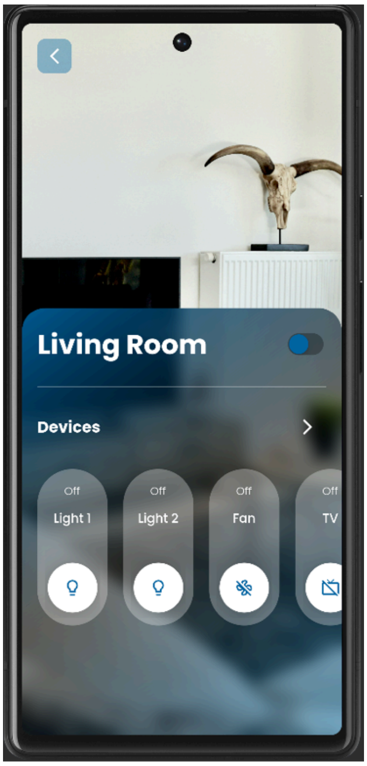


Figure 5.10: Living Room page.

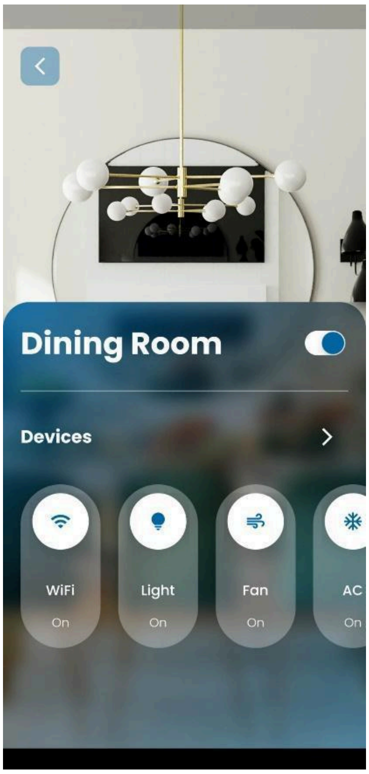


Figure 5.11: Dining Room page

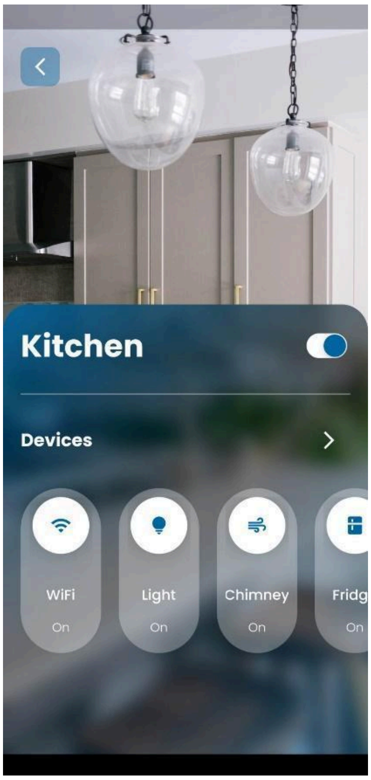


Figure 5.12: Kitchen page

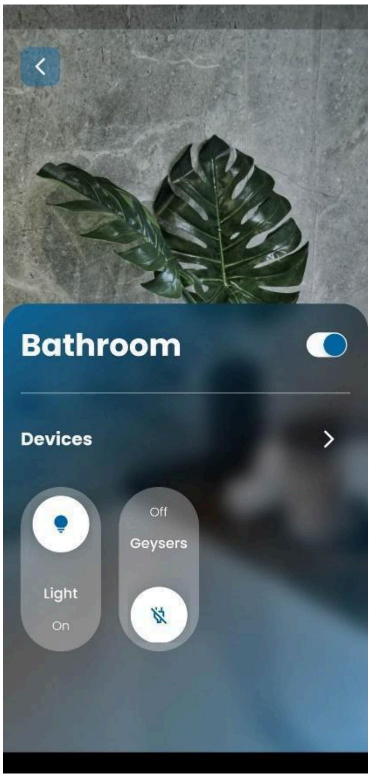


Figure 5.13: Bathroom page

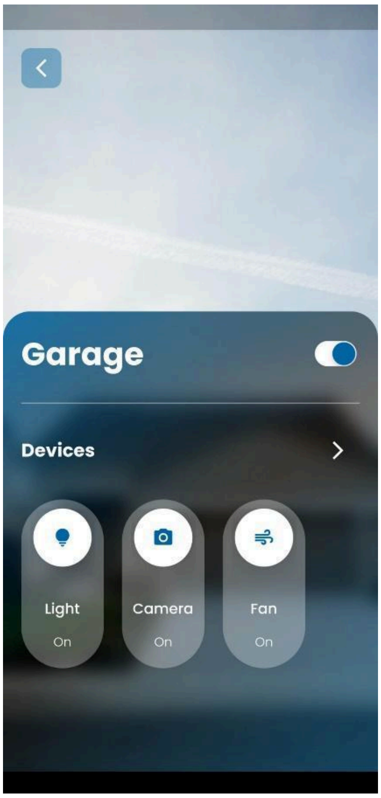
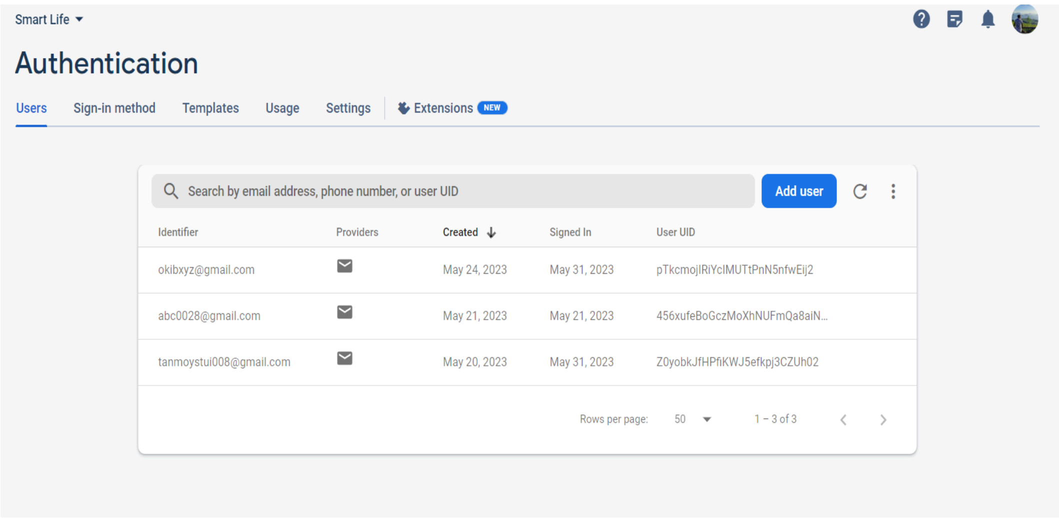


Figure 5.14: Garage page



Identifier	Providers	Created ↓	Signed In	User UID
okibxyz@gmail.com	✉	May 24, 2023	May 31, 2023	pTkcmojIRiYcIMUTtPnN5nfwEij2
abc0028@gmail.com	✉	May 21, 2023	May 21, 2023	456xufeBoGczMoXhNUFmQa8aiN...
tanmoystui008@gmail.com	✉	May 20, 2023	May 31, 2023	Z0yobkJfHPfiKWJ5efkpj3CZUh02

Rows per page: 50 1 – 3 of 3

Figure 5.15: User authentication page from the Firebase

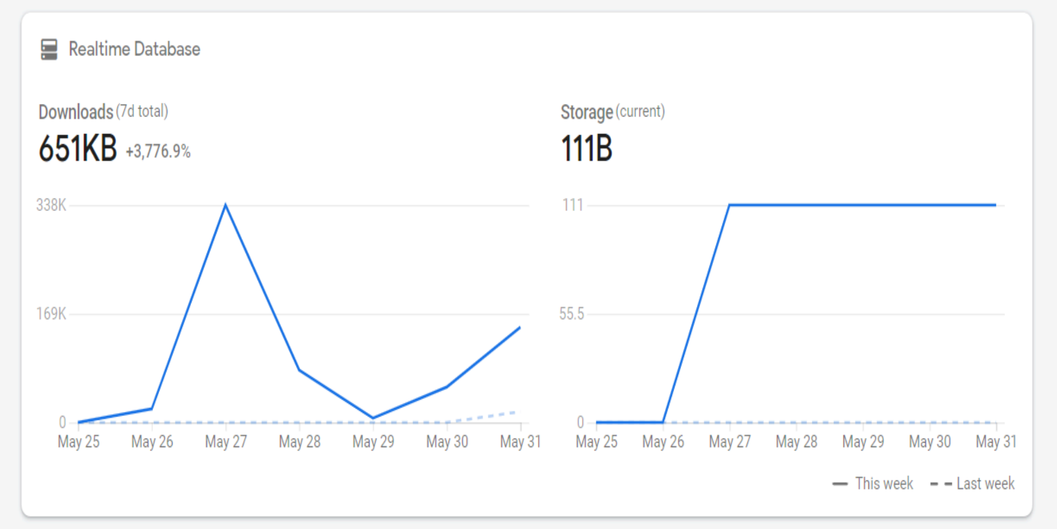
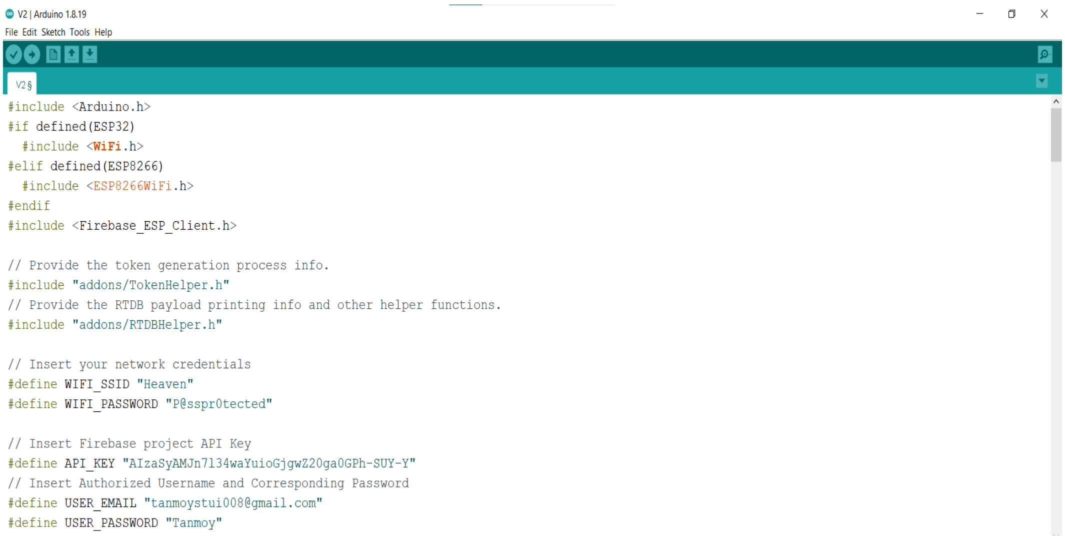


Figure 5.16: Total storage page from the Firebase



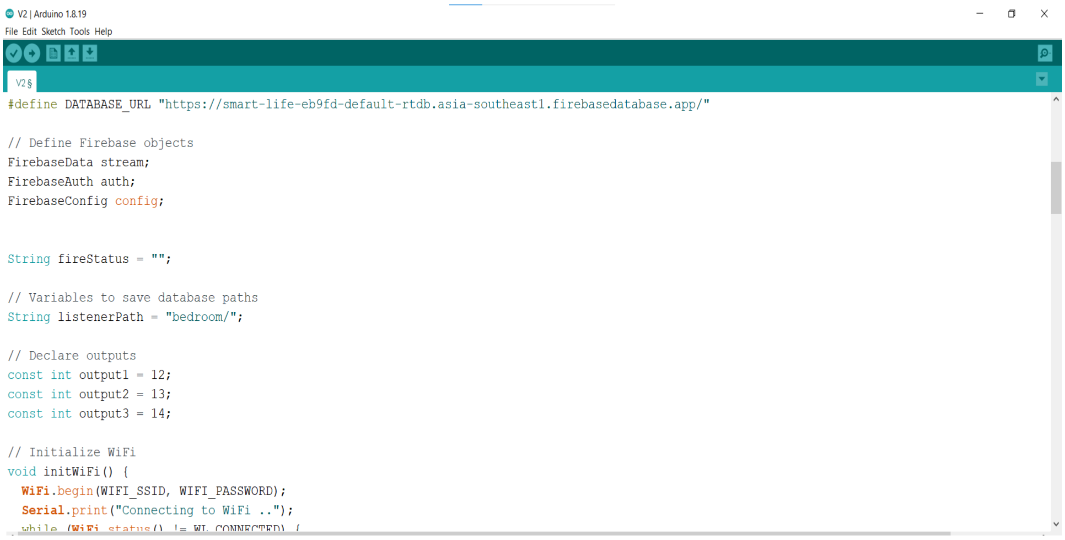
```
#include <Arduino.h>
#if defined(ESP32)
  #include <WiFi.h>
#elif defined(ESP8266)
  #include <ESP8266WiFi.h>
#endif
#include <Firebase_ESP_Client.h>

// Provide the token generation process info.
#include "addons/TokenHelper.h"
// Provide the RTDB payload printing info and other helper functions.
#include "addons/RTDBHelper.h"

// Insert your network credentials
#define WIFI_SSID "Heaven"
#define WIFI_PASSWORD "P@sspr0tected"

// Insert Firebase project API Key
#define API_KEY "AIzaSyAMJn7l34waYuioGjgwZ20ga0GPh-SUY-Y"
// Insert Authorized Username and Corresponding Password
#define USER_EMAIL "tanmoystui008@gmail.com"
#define USER_PASSWORD "Tanmoy"
```

Figure 5.17: ESP8266 setup



```
#define DATABASE_URL "https://smart-life-eb9fd-default-rtdb.asia-southeast1.firebasedatabase.app/"

// Define Firebase objects
FirebaseData stream;
FirebaseAuth auth;
FirebaseConfig config;


String fireStatus = "";

// Variables to save database paths
String listenerPath = "bedroom/";

// Declare outputs
const int output1 = 12;
const int output2 = 13;
const int output3 = 14;

// Initialize WiFi
void initWiFi() {
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to WiFi ..");
  while (WiFi.status() != WL_CONNECTED) {
```

Figure 5.18: ESP8266 setup

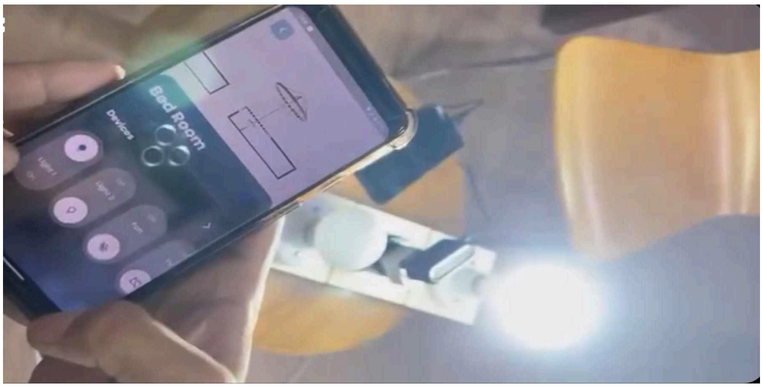
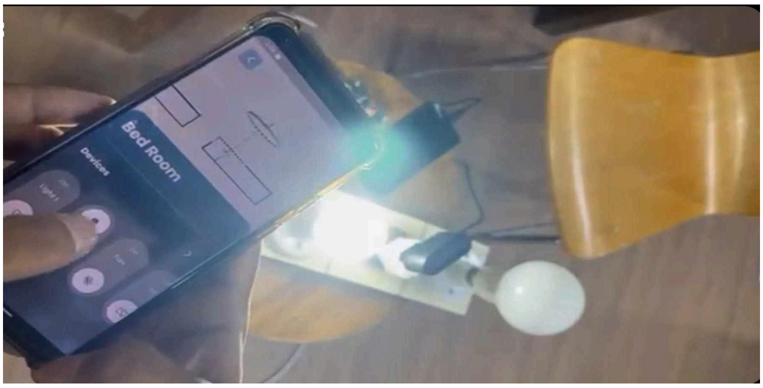


Figure 5.19 & 5.20: The live demo of device switch.



Chapter 6

Results & Analysis

Result

Device Control: Using a single user interface, IoT app users may remotely manage their IoT devices. Users can interact with other connected devices, watch cameras, turn on and off lights, change the temperature of an air conditioner, and change TV channels.

Automation and Convenience: The software enables automation by giving centralized control over various devices. When they leave the house, users can program scenarios or automation processes to change the thermostat and turn off the lights. Even the TV can be programmed to come on at a certain hour.

Energy Conservation: By allowing users to monitor and manage their devices' energy usage, the program can help with energy conservation.
Users can reduce energy consumption by setting the AC's maximum temperature cap or scheduling the lights to turn off automatically at specific times.

Enhanced Security: The integration of cameras and security tools enables remote monitoring of a facility. They can view real-time video feeds and get instant notifications to increase their sense of security and tranquility at home.

Weather Integration: Customers get access to the most recent weather data by integrating hardware or APIs for weather into the app. By receiving temperature updates, weather forecasts, and other information, users can adjust the settings on their smart thermostats based on the weather.

Analysis

User Experience: Providing a flawless and straightforward user experience is essential for the IoT app to be successful. An intuitive user interface, simple device setup, and smooth communication with a variety of IoT devices should all be features of the app. To keep the user experience at its best, regular upgrades and bug fixes are required.

Device Compatibility: The application must work with many different IoT devices from different manufacturers. For it to be more widely utilized and acceptable to consumers, the software must be able to support and work with a range of protocols. (such as Bluetooth, Wi-Fi, and Zigbee).

Privacy and security: Always prioritize one's safety and privacy. Strict security measures must be implemented to safeguard user data. Privacy protection and preventing illegal access require the use of encryption, two-factor authentication, and secure data transmission mechanisms.

Scalability: The app needs to be able to manage heavier loads and increase with the number of IoT users and devices. Strong backend infrastructure and effective coding are needed to keep the app responsive and reliable.

Partnerships and Integration: Increasing the number of supported devices and improving the app's functionality can be achieved by collaborating with service providers and device manufacturers. By forming connections with weather services, energy suppliers, and other IoT ecosystems, the user experience can be enhanced and more functionalities can be made available.

Data analytics and insights: By using anonymized user data that has been obtained in accordance with user consent and privacy controls, it is possible to get knowledge that will improve performance, shape future iterations of the app, and better understand user preferences.

IoT apps can successfully manage and control a range of IoT devices to offer convenience, energy efficiency, security, and enhanced user experiences. Special consideration must be given to the user experience, device compatibility, security, scalability, and integration to guarantee the product's economic success.

Chapter 7

Project as Engineering Problem Analysis

7.1 Sustainability of the Project/Work

Market Demand: The long-term viability of the app is significantly influenced by consumer adoption of IoT devices and demand for smart home technology. If you want your app to remain popular over time, it's critical to monitor market trends, comprehend user preferences, and modify the app's functionality to satisfy changing needs.

Device compatibility and integration: As the IoT ecosystem matures and new technologies are embraced, it is essential to ensure that outdated devices can connect with contemporary ones.

Regular updates and improvements that support a range of devices, protocols, and standards will ensure that the app remains useful and draws in a broader audience.

User Retention and Engagement: For sustainability, encouraging user engagement and retaining current users are crucial. By continually adding new features to the app, providing personalized experiences, and paying attention to user feedback, it is feasible to boost user satisfaction and loyalty. To retain customers, incentives might be used like exclusive discounts, presents, or offers.

Sustainability and energy efficiency: The app's emphasis on energy efficiency features is in keeping with the growing significance of sustainable practices. Promoting energy-saving settings, providing information on energy consumption, and encouraging users to choose environmentally friendly options can all help the app become more sustainable.

Collaborations and Ecosystem Expansion: Building a strong ecosystem can be facilitated by collaboration with IoT platform developers, service providers, and other connected device manufacturers. By forming new alliances, connecting with complementing services, and ensuring seamless interoperability, the app may increase consumer value and increase its sustainability.

Money Streams and Data Monetization: By looking into how to monetize the data the app produces; additional money streams may be created (with the right user consent and privacy controls in place). It is possible to analyze aggregated and anonymized data to derive insights for tailored recommendations, energy optimization services, or focused advertising.

Security and Privacy: Sustainability requires strong security safeguards and the protection of user privacy. A data breach or privacy violation could have a negative impact on the app's reputation and user confidence. Continuous investment in security measures, current knowledge of industry best practices, and prompt correction of shortcomings are requirements for long-term viability.

Regulatory Compliance: It's essential to follow any rules and regulations that may be in force, including IoT security requirements and data protection laws. Complying with industry standards promotes user trust by lowering potential hazards to the business's brand and legal standing.

Support and Maintenance: Bug fixes, ongoing maintenance, and customer support are crucial for survival. Users need a dependable and responsive app experience, and maintaining user satisfaction and loyalty requires promptly addressing issues or aiding.

By taking these factors into consideration, aggressively responding to market developments, and placing a focus on user satisfaction, an IoT software managing many devices can work toward sustainability over the long term.

7.2 Social and Environmental Effects and Analysis

The feasibility of the project depends on several variables. The ability of a product to be updated and maintained is known as sustainability. Understanding sustainability ideas and developing a strong sustainability plan are crucial to preventing project failure. There are three types of long-term viability for the product:

Community Sustainability: How much the community, or the system's intended users, will support the project's long-term viability, is referred to as "community sustainability". The goal of this project is Flow Technology. This application is really easy to use. Everyone may use it effortlessly, and new users can easily become a part of the user community.

Financial Sustainability: This concerns both whether the application will produce sufficient money to earn a respectable profit as well as how its ongoing operating costs will be kept under control once it has been implemented. The running costs of a program also include expenses for hardware, database storage, third-party APIs, etc.

Organizational Sustainability:
Organizational sustainability refers to an organization's ability to run its operations and accomplish its objectives over a protracted period of time while limiting adverse effects on the environment and society. The company frequently maintains an application with either its present team, an expanded team, or a whole new team after it has been deployed. Organizations can also continue working on other projects, grow their teams, form new teams, and refresh their project by including fresh features.

7.3 Addressing Ethics and Ethical Issues

Privacy and Security: Data and personal information belonging to users must be protected from infiltration and hacking by the software. The program should use encryption to safeguard data while it is being transmitted and stored.

Data Ownership: The program must ensure that all data generated by the devices is the user's. The user must be able to control who has access to and uses their data.

Bigotry and Discrimination: The application shouldn't function in a way that encourages prejudice or bigotry. For instance, based on the user's socioeconomic level, gender, or color, it shouldn't allow one user to have more influence over the gadgets than another.

Transparency: The application should run openly and transparently and provide users with clear instructions on how to utilize the gadgets. A user-friendly user interface is also required so that users can comprehend the types of data being gathered, their uses, and data governance procedures.

Accessibility: No matter what skills or limitations someone may have, they should be able to use the application. Users with disabilities should have no issue using the program due to its outstanding design.

Impact on the Environment: When controlling devices, the application must consider the environment. For example, users should be able to manage and reduce their energy use to halt climate change.

Ethics are vital for any IoT-based Flutter application that controls devices like TVs, air conditioners, cameras, Wi-Fi, etc. Developers must prioritize the privacy, security, transparency, accessibility, and environmental impact of their apps to prevent harm to users or society.

Chapter 8

Lesson Learned

8.1 Problems Faced During this Period.

Since I was new with this platform, it was challenging for me to prepare for the assignments that were provided to me through study and practice. I had to learn a ton of new file kinds, libraries, and APIs to complete the task. Working on the project's routing component presented several challenges for me because I was unfamiliar with the flutter framework. Dart was my primary choice for the programming language when I was creating the project's features The hardest thing I had to do during my internship was figured out how to blend in and build relationships with those around me. Considering my shy disposition, I came across it troublesome to get along and be transparent with everyone. As a result, I started to work carefully, especially while refining and finishing the concepts. Due to the interruptions, I lost a lot of productivity, therefore I had to work harder to complete the task before the deadline.

8.2 Solution of those Problems

First off, I had no prior knowledge of Flutter or Dart's framework. I had to learn darts and flutter and its packaging during my internship period. I then had to put it into action. Firebase, a repository for smart living apps, was something I understood very little about at the time. I had to learn every single one of those things by that moment. But more than anything else, I had to overcome problems during my internship that were personal. I learned that I need to be more friendly and chatty while starting work in a new place. This calls for confidence in the task I'll be finishing as well. I saw that work was getting done more quickly and that everyone was willing to provide a helping hand, especially to an intern, when I opened up and started talking openly. I spent an increasing amount of time doing research and accumulating knowledge to get beyond the challenges. Most of my time was also spent online reading blog posts and library documentation for various issues I encountered. I've learnt during my internship to handle a lot of work without procrastinating or slacking off. as the strain became especially heavy.

Chapter 9

Future Work & Conclusion

9.1 Future Works

- Profile customization.

- Connect with other users.

- The application's integration with voice assistants. For instance, using Google Home or Amazon Alexa enables customers to effortlessly and hands-free control their devices.

- Security is a significant issue for IoT applications since cyber threats are increasing in frequency. Future work on the app can concentrate on enhancing the security measures to prevent unauthorized access and protect user privacy.

- The software can be improved further by implementing machine learning and AI algorithms to assess usage trends, give personalized recommendations, and automate particular tasks in accordance with user preferences.

- Integration with well-known smart home ecosystems like Apple HomeKit or Samsung SmartThings. Users will be capable to operate each of the smart home devices they personalized from a single app courtesy of the app's accessibility with recognized smart home ecosystems like Apple HomeKit and Samsung SmartThings.

- The software can be set up to use the least amount of energy possible by automatically turning off gadgets when they are not in use or changing the air conditioner's temperature based on the weather forecast.

- The program might be improved to offer real-time device tracking, giving consumers the ability to remotely check the health of their devices and get notifications when something goes wrong.

- Enhance current features.

9.2 Conclusion

This was a two-person group project. Based on our respective areas of expertise, we equally divided the work and finished the "Smart Life Application" as a team.

I developed my understanding and abilities in Flutter and dart during my internship. I was given the chance to observe the workplace culture throughout this internship. I ended by thanking Dean and my boss. And the project I was given the chance to work on is an Internet of Things-based flutter application that would manage numerous devices. It will manage the technology at home or at business in a nutshell.

The many benefits that such an application may offer include convenience, increased effectiveness, and improved user experience. For instance, users may use their cellphones to operate many devices quickly and simply in their homes or offices without having to physically interact with each one.

A Flutter application built on the Internet of Things might also provide better functionality and automation options. Users can program their devices with automatic routines or schedules that do things like turn off the lights when they leave the room or change the thermostat based on the weather.

If an IoT-based Flutter application for managing devices is correctly developed and put into use, it might provide considerable advantages and enhance both consumer and commercial user experiences. The dependability of the underlying IoT infrastructure, the application's usability and accessibility, and the security and privacy of the data being transmitted between the devices and the application are just a few of the aspects that will determine the success of the application.

Bibliography:

[1] Das, S. R., Chita, S., Peterson, N., Shirazi, B. A., & Bhadkamkar, M. (2011). *Home automation and security for mobile devices. 2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops).*

[2] Hoflinger, F., Gamm, G. U., Albesa, J., & Reindl, L. M. (2014). *Smartphone remote control for home automation applications based on acoustic wake-up receivers. 2014 IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings.*

[3] Nunes, R. J. C., & Delgado, J. C. M. (n.d.). An Internet application for home automation. 2000 10th Mediterranean Electrotechnical Conference. Information Technology and Electrotechnology for the Mediterranean Countries. Proceedings. MeleCon 2000 (Cat. No.00CH37099).

[4] Vaishnavi S. Gunge, Pratibha S. Yalagi. Smart Home Automation: A Literature Review. International Journal of Computer Applications (0975 – 8887) National Seminar on Recent Trends in Data Mining (RTDM 2016).

[5] Brush, A. J. B., Lee, B., Mahajan, R., Agarwal, S., Saroiu, S., & Dixon, C. (2011). *Home automation in the wild. Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems - CHI '11.*

[6] Amul Jadhav* S. Anand Nilesh Dhangare K.S. Wagh. Universal Mobile Application Development (UMAD) On Home Automation. Network and Complex Systems www.iiste.org ISSN 2224-610X (Paper) ISSN 2225-0603 (Online) Vol 2, No.2, 2012

[7] Tuya Smart - Global IoT Development Platform Service Provider.

[8] Blynk: a low-code IoT software platform for businesses and cloud.

Consent Form

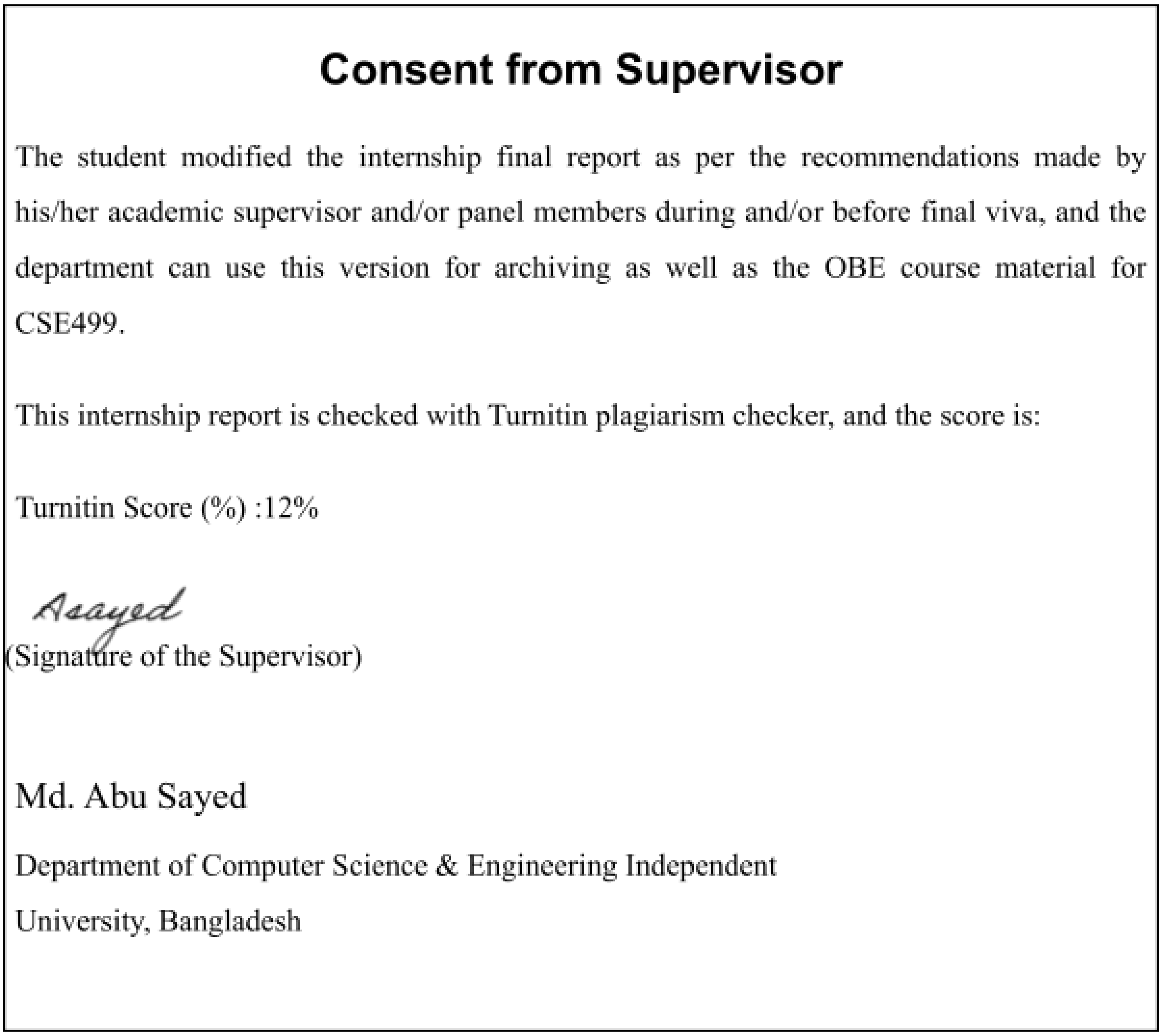
An Undergraduate Internship on Smart Life Application

By

Tonmoy Baishnob

Student ID: **1830295**

Spring, 2023



Consent from Supervisor

The student modified the internship final report as per the recommendations made by his/her academic supervisor and/or panel members during and/or before final viva, and the department can use this version for archiving as well as the OBE course material for CSE499.

This internship report is checked with Turnitin plagiarism checker, and the score is:

Turnitin Score (%) :12%

Asayed

(Signature of the Supervisor)

Md. Abu Sayed

Department of Computer Science & Engineering Independent
University, Bangladesh