# Microcurve Automation System


By

**Aiman Ali**

**Aminah Shahzad**

**Muhammad Falah Naveed**

**Rania Adnan**

Supervised by:

**Dr. Yawar Abbas Bangash**

Submitted to the faculty of the Department of Computer Software Engineering, Military College of Signals, National University of Sciences and Technology, in partial fulfillment of the requirements of B.E Degree in Software Engineering.

May 2022

In the name of ALLAH, the Most benevolent, the Most Courteous

## CERTIFICATE OF CORRECTIONS & APPROVAL

*Certified that work contained in this thesis titled*

*"**Microcurve Automation System**"*

*is carried out by*

***Aiman Ali, Aminah Shahzad, Muhammad Falah Naveed, and Rania Adnan***

*under supervision of*

***Dr. Yawar Abbas Bangash***

*for partial fulfillment of Degree of Bachelor of Software Engineering, in Military College of Signals, National University of Sciences and Technology, Islamabad is correct and approved. The material that has been used from other sources has been properly acknowledged/referred to.*

**Approved by**

**Supervisor**

**Dr. Yawar Abbas Bangash**

**Department of CSE, MCS**

Date: _____

## DECLARATION

**We hereby declare that no portion of the work presented in this thesis has been submitted in support of another award or qualification in either this institute or anywhere else.**

# Plagiarism Certificate (Turnitin Report)

This thesis has been checked for Plagiarism. Turnitin report endorsed by Supervisor is attached.

Signature of Student

Aiman Ali

00000247690

_____

Signature of Student

Aminah Shahzad

00000278778

_____

Signature of Student

Muhammad Falah Naveed

00000192184

_____

Signature of Student

Rania Adnan

00000254480

Signature of Supervisor

# Acknowledgments

We are thankful to our Creator Allah Subhana-Watalah who has guided us throughout this work at every step and for every new thought which He set up in our minds to improve it. Indeed, we could have done nothing without His priceless help and guidance. Whosoever helped us throughout our thesis, whether our parents or any other individual was His will, so indeed none be worthy of praise but Allah.

We are profusely thankful to our beloved parents who raised us when we were not capable of walking and continued to support us throughout every department of our lives. We could not have taken this journey without their tremendous support. Your prayers for us were what sustained us this far.

We would also like to express special thanks to our supervisor **Dr. Yawar Abbas Bangash** for his help throughout our thesis and his tremendous support and cooperation. Without his help, we wouldn't have been able to complete our thesis. We appreciate his patience and guidance throughout the whole thesis.

We are highly thankful to all the faculty and staff of the Computer Software Department of Military College of Signals, NUST for their support and training throughout our coursework. Their guidance helped us to carry out this project.

Finally, we would like to acknowledge the efforts of all our friends, colleagues, and well-wishers, whose prayers and faith helped us in achieving our goal.

***Dedicated to our exceptional parents, adored siblings, and our supervisor Dr. Yawar Abbas Bangash whose tremendous support and cooperation led us to this wonderful accomplishment.***

# Abstract

A normal Wi-Fi based automation system has a limitation of an operational range of 30-50 meters. The appliance working in this system must be within this range. The problem of lesser range can be solved by installing a Microcurve automation system which is an antenna-based automation system. It acts as an additional layer over the Wi-Fi-based automation system with the addition of an antenna-based mechanism attached to the router that will increase the operational range of appliances up to 1 km (LOS). The appliances are operated by Microcurve smart app. When the button is clicked to switch on an appliance from the Microcurve app, data sent by the device reached the Wi-fi router. This data includes the destination address with the IP address of the Ethernet shield attached to the router. The router then routes the data towards the micro-controller via port forwarding. The data is accessed by the controller and sent towards the specific appliance via the antenna transceiver. The transceiver attached to the appliance will convert the incoming analog data to digital form and data is executed, and the appliance is turned on via relay. The same process happens for turning off the appliance.

# Table of Contents

# *Table of Figures*

# Chapter 1. Introduction

## 1.1. Overview

MAS is an additional layer over Wi-Fi-based automation with the addition of an antenna-based mechanism attached to the router that will increase the operational range of appliances up to 1km (LOS). It consists of:

- Microcurve android application
- Microcurve box (attached to Wi-Fi router)
- our antenna-based modules

The user will face the same experience as that of Wi-Fi-based automation. The range here is between the router and the appliance to be controlled. The appliances working with the normal household router must be in the range of 30-50 meters from the Wi-Fi router. MAS will provide a 1 km (LOS) range with that same router without the use of any expensive commercial or industrial routers.

## 1.2. Problem Statement

Nowadays in industrial and commercial sectors, wi-fi-based automation systems are used but with various Wi-Fi extenders/boosters and are costly. A normal Wi-fi based automation system has a limitation of an operational range of 30-50 meters. The appliance working in this system must be within this range. This problem of range limitation can be solved by installing an antenna-based automation system i.e., Microcurve Automation System.

## 1.3. Approach

Being the prototype, it provides a range of 1 km, and a known environment was observed for the system to work in. Layered architecture is applied to make the system flexible to changing needs.

The project was divided into four phases. In the first phase, separate modules for hardware and software were developed. In the second phase, Firebase Cloud Database was designated as Central Database to provide even more flexibility. The automation system was engineered to provide maximum utility and promised range i.e., 1 km (LOS). In the third phase, the hardware was connected to Microcurve smart app, an android application, which controlled the switches. Microcurve automation was fully functional. In the last phase, Microcurve underwent testing.

## 1.4. Scope

The scope of this project will not only be limited to domestic usage, but it will provide a cost-effective yet comprehensive solution by designing a proprietary software and hardware that will benefit the large-scaled smart buildings, and commercial markets, helpful for large-scale irrigation systems, intelligent firefighting, and logistics tracking.

## 1.5. Objectives

The main objective of the Microcurve Automation System is to provide an operational range of 1 km (LOS) to the user. Following are the objectives that are kept in mind:

a. Development of an Android Application to provide an interface of communication for users to interact with the automation system.
b. Communication of Android Application with any appliance over the network. The operational range of 1 km (LOS) is between the router and the appliance.
c. Easy-to-use

During this project, all the aspects of software engineering are covered i.e., survey and feasibility analysis, requirement gathering, architecture, and detailed design, implementation, and testing

along with documentation (SRS, SDD, Test Documentation, Final Report, and User Manual). Group Members are also expected to develop extensive knowledge and technical skills in the fields of IoT and Android development.

## 1.6. Deliverables

| sr | Tasks | Deliverables |
|---|---|---|
| **1** | Literature review | Literature survey and Feasibility Analysis |
| **2** | Requirements Specification | Software Requirements Specification Document (SRS) |
| **3** | Detailed Design | Software Design Document (SDD) |
| **4** | Implementation | Project Demonstration |
| **5** | Testing | Evaluation plan and test document |
| **6** | Training | Deployment plan |
| **7** | Deployment | Complete application with the necessary documentation |

## 1.7. Overview of document

### 1.7.1. Purpose

This document covers a detailed review of all major steps involved in the Software Development Life Cycle (SDLC), Each chapter covers a single step of SDLC and goes from Software Requirement Specification, Software Design Specification to Software Test Plan &

Strategy. These all-involved steps acted as a guide to the development team and now shall provide insight to the reader that how the prototype idea was formulated and then how hardware integration took place, and how software was designed and finally tested.

### 1.7.2. Document Conventions

Headings are prioritized in a numbered fashion, the highest priority heading having a single digit and subsequent headings having more numbers, per their level. All the main headings are titled as follows: a single-digit number and the name of the section (All bold Calibri Light, size 18, Centered). All second-level subheadings for every subsection have the same number as their respective main heading, followed by one a dot and subsequent subheading number followed by name of the subsection (All bold Calibri Light, size 16). Further subheadings, i.e., level three and below, follow the same rules as above for numbering and naming, but different for the font (All bold Calibri, size 14).

#### 1.7.2.1. Figures

All figures in this document have captions and are numbered. All Software Design related diagrams are based on the latest UML standards.

#### 1.7.2.2. References

All references in this document are provided where necessary, however they, were not present, the meaning is self-explanatory. All ambiguous terms have been clarified in the glossary at the end of this document.

#### 1.7.2.3. Links to web pages

All links have been provided with underlined font, and the title of the web page or e-book is

written at the top of the link and the title may be searched on google to pinpoint the exact address.

#### 1.7.2.4. Basic Text

All other basic text appears in regular, size 12 Calibri Light. Every paragraph explains one type of idea.

### 1.7.3. Intended Audience and Reading Suggestions

#### 1.7.3.1. Intended Audience

This document is useful for:

a. Developer
b. Users
c. External Guide
d. Internal Guide
e. Automation and IoT companies
f. Testers
g. Project Evaluators

#### 1.7.3.2. Reading Suggestions

**For better understanding, the document is divided into chapters**

In chapter 1 an introduction to Document and System is provided.

Chapter 2 covers the requirement specifications part and covers Functional, Non-Functional Parts Requirements, resources required, and constraints involved

Chapter 3 covers the Design Specifications which provide an in-depth view of how the systems are developed and how the functionalities are distributed.

Chapter 4 discusses the Testing Phase of SDLC.

Finally, a conclusion is drawn and recommendations for future work are made.

# Chapter 2. Literature Review

## 2.0. Preamble

No project related to an automation system with a range of 1 km (LOS) has been made previously at Military College of Signals, NUST.

Due to rapid development in the IoT field, the needs of users are also changing. Now people want to operate as many devices as they want with a single click. Smart automation systems are growing fast due to applications and systems utilizing these techniques. The world is moving towards an automated and controlled world.

In recent years, a lot of research has been made and surveys are carried out for smart automated systems controlling our homes, offices, and even industries. Each automated system has its range and cost.

## 2.1. Limitations of Current Solutions

The current solutions available in the market have mainly two kinds of limitations:

1. Firstly, the currently available solutions are Wi-Fi-based automation systems which provide a range of 30-50 meters only.
2. Secondly, suppose if a Wi-Fi-based automation system is installed, then to get an operational range of 1 km**, Wi-Fi extenders/boosters** are required which can make the solution expensive.

## 2.2. Design Review

In the market, different automation systems are available but with a lesser operational range. This range is extended by installing more **Wi-Fi extenders/boosters.** So, why not choose an automation system with no nodes and a higher operational range. This was a

more suitable option i.e., to have an antenna-based automation system with an operational range of 1 km (LOS).

## 2.3. Conclusion

Considering the existing solutions and possible limitations, a system is required that requires no Wi-Fi extenders/boosters to operate appliances. So, Microcurve is the best solution to the problem.

# Chapter 3. System Requirement Specifications

## 3.1. Introduction

The Software Requirements Specification (SRS) provides a detailed description of the requirements for the Microcurve Automation System (MAS). This SRS allows for a complete understanding of what is to be expected of the MAS that is to be constructed. The idea is to develop an automation system for domestic and commercial use. It will provide a cost-effective solution to the users. It should be able to operate appliances in the building or outside the building within 1 km. In addition to the basic requirements, this SRS describes the external interface requirements, non-functional requirements, overall description, system features, etc. An ecosystem will be developed by MAS for synchronized communication.

## 3.2. Overall description

### 3.2.1. Project perspective

MAS is an additional layer over Wi-Fi-based automation with the addition of an antenna-based mechanism attached to the router that will increase the operational range of appliances up to 1km (LOS). It consists of:

- MICROCURVE android application
- MICROCURVE box (attached to Wi-Fi router via LAN cable/WIFI)
- our antenna-based modules

The user will face the same experience as that of Wi-Fi-based automation.

Note: The range mentioned here is between the router and the appliance to be controlled. The appliances working with the normal household router must be in the range of 30-50

meters from the Wi-Fi router. MAS will provide a 1km range with that same router without the use of any expensive commercial or industrial router.

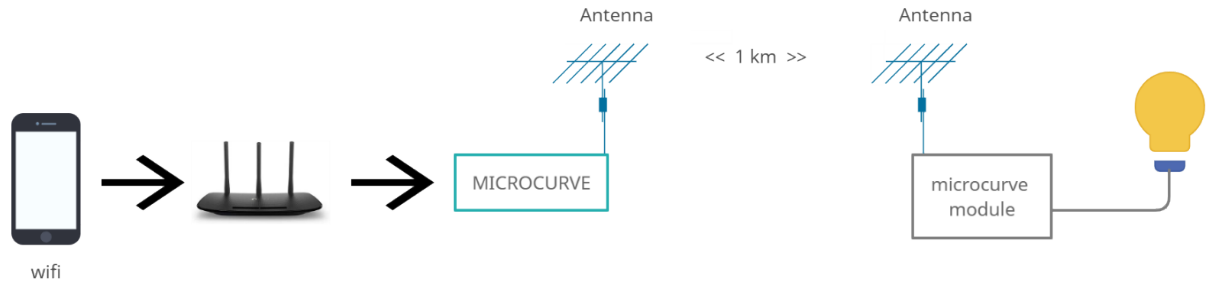


**Figure 1.1 -** *Component Diagram of Microcurve Automation System. Data is sent from the app to the router. The data is transferred to Microcurve box and then to the module which turns an appliance on/off.*

### 3.2.2. Product Features

The key features of the "Microcurve Automation System" are highlighted below:

- Android application for automation system provides an interface to the user for interacting with the automation system. Users shall be able to control arm(lock)/disarm(unlock) appliances from mobile.
- Feedback shall be provided to the user if the appliance has been turned on/off.
- Users shall be able to view the (history of events) and the cameras attached to them.
- Only users with specified authority shall be able to interact with our automation system in enable mode.
- Users shall be able to log in with registered credentials
- Connection of Android application to Arduino over the internet

Below is the component diagram for the project.

## 3.2.3 User Classes and Characteristics

### 3.2.3.1. Summary of User Classes

MAS will be primarily intended for the following user classes:

- Large-scaled smart buildings
- commercial markets
- will be helpful for large scale irrigation systems
- intelligent firefighting
- logistics tracking
- Domestic use

## 3.2.4. Operating Environment

The subsections below give a brief description of the environment, hardware, and software-based requirements for the operation of “Microcurve automation system”.

### 3.2.4.1 Hardware Environment

- NodeMCU ESP8266 (1)
- Arduino UNO (2)
- NRF24L01 2.4 GHz Antenna transceiver (2)
- 5V single channel relay (1)
- A few resistors and LEDs

**3.2.4.2** Software Environment

- Windows
- IDE: Android Studio

- Arduino IDE

### 3.2.5. Design and Implementation Constraints

The device does have its **design and implementation constraints**

- MICROCURVE provides its Eco-system so, it is not compatible with any existing automation system.
- The modules attached with the appliances are also not replaceable with any existing Wi-Fi modules.

### 3.2.6. User Documentations

A user manual will be provided to the users in which separate instructions will be given according to the user i.e., Regular users, developers, and testers. It will include the details of the system's working. Project synopsis will also be a part of the system.

The project report will also be available for the users which will highlight the system features, working, and procedures.

### 3.2.7 Assumptions and Dependencies

Following dependencies and assumptions are being observed for the project:

- The project is dependent on both hardware components and software.
- The system should work on the operating systems that are described
- Software must be in the English language
- The user has a basic knowledge of the system environment
- Microcurve automation system should be installed completely

## 3.3. External Interface Requirements

### 3.3.1. User Interfaces

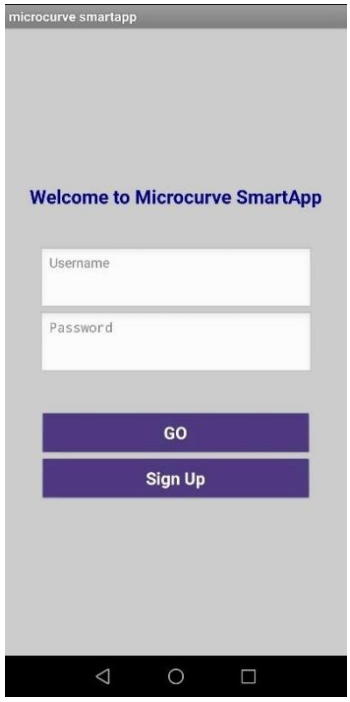


***Figure 1.2*** *Android - Signup and Login Screen*

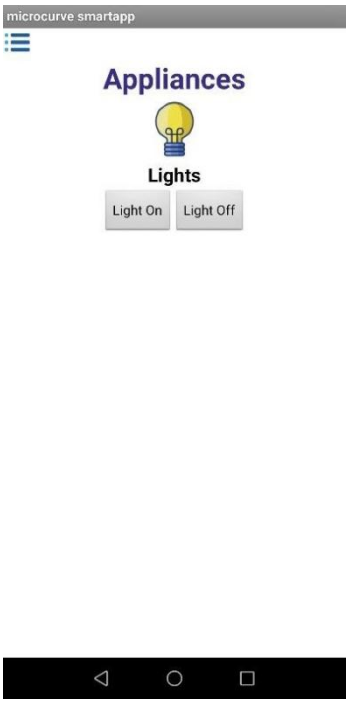


***Figure 1.3*** *Android Application – Home Screen*


***Figure 1.4*** *Android Application – Menu View*

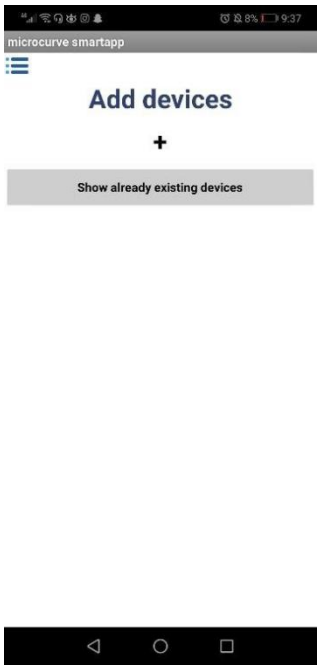


***Figure 1.5*** *Android Application – Add device screen*


***Figure 1.6*** *Android Application – About Us page*


***Figure 1.7*** *Android Application – Remove Device Screen*

## 3.3.2. Hardware Interfaces

- The Wi-Fi router shall be connected to the MICROCURVE using an Ethernet cable/WIFI.
- In MICROCURVE, the antenna module shall be connected to the microcontroller.
- 2.4 GHz GFSK modulated signal shall be transmitted by Antenna.
- In the MICROCURVE module, the antenna module and microcontroller shall be connected to the appliance.
- 2.4 GHz GFSK modulated signal shall be received by Antenna.

### 3.3.3 Software Interfaces

- Android Application shall be used to provide an interface to the user for interacting with the hardware components (Automation system)
- Android Studio shall be used to build an application.
- Database
- Java shall be used to build an application in the android studio

### 3.3.4. Communication Interface

- Android app is connected to WIFI (in our case WIFI is the access point of NodeMCU.
- WIFI information is transferred to the NodeMCU via SPI (Serial Peripheral Interface) communication
- NodeMCU is connected to the Arduino UNO via Arduino serial communication (digital writing and reading of pins)
- Arduino UNO is connected to the NRF24L01+ antenna using SPI (Serial Peripheral Interface)
- NRF24L01+ communicates with another NRF24L01+ (receiver side) via RF radios
- Second NRF24L01+ communicates with second Arduino UNO (receiver side) via SPI
- Second Arduino sends a signal to 5V relay via Arduino serial communication

## 3.4 System Features

This section illustrates an organization of functional requirements of the project "Microcurve Automation System" by System Features:

i. Authentication

ii. Authorization

iii. Addition of new modules/devices

iv. Removing New modules/devices

v. Acquisition of Feedback/Notification

vi. Transmission of data from Wi-fi Module to Antenna Module

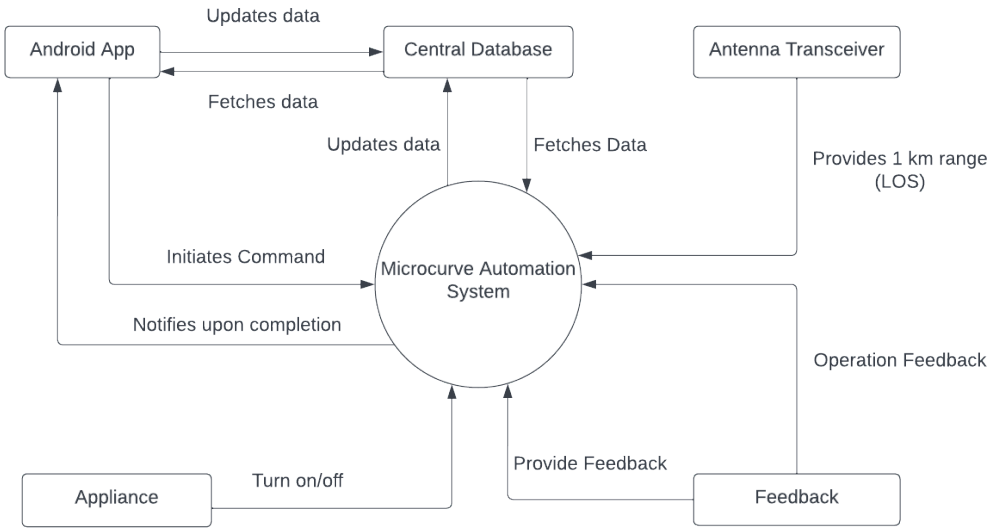


**Figure 1.8** *Context Diagram of Microcurve automation system. The diagram shows how the Android Application generates feedback that turns on/off an appliance.*

## 3.4.1 Authentication

### 3.4.1.1 Description

This feature enables the user to get register with the system and more specifically the Android Application. Also, this feature shall ensure the authenticity of the user.

#### 3.4.1.2 Stimulus/Response Sequences

| *3.4.1.2.1* ***Normal Path: User Registers with Application successfully*** |
|---|
| **Pre-conditions:**<br>User must open the provided Android Application |
| **Interactions:**<br>The user clicks the “Register” button |
| **Post-conditions:**<br>The user shall be taken to the Home Screen of the Application. |
| **Categorization:**<br>**Criticality:** High<br>**Probability of defects**: Low<br>**Risk:** Medium |

| *3.4.1.2.2* ***Exceptional Path: An error message is displayed*** |
|---|
| **Pre-conditions:**<br>The application is not connected to the internet<br>The user enters invalid credentials<br>The user enters an invalid verification code. |

| |
|---|
| **Interactions**<br><br>An exception is thrown into the system |
| **Post-conditions:**<br><br>An error message is displayed |
| **Categorization**<br><br>**Criticality:** High<br><br>**Probability of defects:** Low<br><br>**Risk:** High |

#### 3.4.1.3 Functional Requirements

**REQ-1:** The Application shall enable the user to get himself/herself registered with the System.

**REQ-2:** The Application shall take the User's credentials including Full Name, Email address & a Password, and a Phone number as input.

**REQ-3:** The Application shall authenticate the user's input data by matching the verification code with System's database.

**REQ-4:** Only, in case of successful authentication, the application shall take the user to the Home Screen.

**REQ-5:** In case, the user signs out of the application, the user shall only be able to use the Application's functionality once a user has signed back into the application after providing an email address & password, as input to the application.

## 3.4.2. Authorization

### 3.4.2.1 Description

This feature shall allow the system to store data in the database to authorize it through the data link layer. The system shall keep a record of each user's data and only give access to data if the user meets authentication requirements. The system shall allow encrypted passwords to be stored in the user account database, which will be used to authorize the user every time the user is authenticated to login into the account.

### 3.4.2.2 Stimulus/Response Sequences

| ***3.4.2.2.1 Normal Path: User Logins with Application successfully*** |
|---|
| **Pre-conditions:**<br>User must open the provided Android Application |
| **Interactions:**<br>The user must enter the correct credentials i.e., username and password |
| **Post-conditions:**<br>The user shall be taken to the Home Screen of the Application. |
| **Categorization:**<br>**Criticality:** High<br>**Probability of defects:** Low<br>**Risk:** Medium |

| *3.4.2.2.2* ***Exceptional Path: An error message is displayed*** |
|---|
| **Pre-conditions:** |

| |
|---|
| The application is not connected to the internet<br><br>The user enters invalid credentials |
| **Interactions**<br><br>An exception is thrown into the system |
| **Post-conditions:**<br><br>An error message is displayed |
| **Categorization**<br><br>**Criticality:** High<br><br>**Probability of defects:** Low<br><br>**Risk:** High |

#### 3.4.2.3 Functional Requirements

**REQ-1:** System shall authorize a username for every user who wants to access the account.

**REQ-2:** System shall authorize a password for every user who wants to access the account.

### 3.4.3 Addition of new modules/devices

#### 3.4.3.1 Description

This feature shall allow the system to let the legitimate user add modules/devices.

#### 3.4.3.2 Stimulus/Response Sequences

| |
|---|
| *3.4.3.2.1* ***Normal Path: The user can add modules successfully*** |

| |
|---|
| **Pre-conditions:**<br><br>The user must log in to the application successfully |
| **Interactions:**<br><br>The user shall select the modules system menu<br><br>The user shall be able to add the device by adding a unique name. |
| **Post-conditions:**<br><br>The system shall display already saved devices/modules and various options to change modules.<br><br>The system shall display the new saved devices |
| **Categorization:**<br><br>**Criticality:** High<br><br>**Probability of defects:** Low<br><br>**Risk:** Medium |

| |
|---|
| *3.4.3.2.2* ***Exceptional Path: An error message is displayed*** |
| **Pre-conditions:**<br><br>The application is not connected to the internet |
| **Interactions**<br><br>An exception is thrown into the system |
| **Post-conditions:**<br><br>An error message is displayed |
| **Categorization** |

| |
|---|
| **Criticality:** High<br>**Probability of defects**: Low<br>**Risk:** High |

#### 3.4.3.3 Functional Requirements

**REQ-1:** The user shall be able to add devices to the MAS android application.

**REQ-2:** System shall display all the devices connected to the Wi-Fi module whenever the user adds them on the MAS android application.

## 3.4.4 Removing devices/Modules

#### 3.4.4.1 Description

This feature shall allow the system to let the legitimate user remove modules.

#### 3.4.4.2 Stimulus/Response Sequences

| |
|---|
| *3.4.4.2.1* ***Normal Path: User can remove modules successfully*** |
| **Pre-conditions:**<br>The user must log in to the application successfully |
| **Interactions:**<br>The user shall select the modules system menu<br>The user shall be able to remove the device when necessary. |
| **Post-conditions:**<br>The system shall display already saved devices/modules and various options to change modules. |

| |
|---|
| The system shall display the saved devices after removing a device. |
| **Categorization:**<br><br>**Criticality:** High<br><br>**Probability of defects:** Low<br><br>**Risk:** Medium |

| |
|---|
| *3.4.4.2.2* ***Exceptional Path: An error message is displayed*** |
| **Pre-conditions:**<br><br>The application is not connected to the internet |
| **Interactions**<br><br>An exception is thrown into the system |
| **Post-conditions:**<br><br>An error message is displayed |
| **Categorization**<br><br>**Criticality:** High<br><br>**Probability of defects:** Low<br><br>**Risk:** High |

### 3.4.4.3 Functional Requirements

**REQ-1**: The user shall be able to remove devices to the MAS android application.

**REQ-2:** System shall display all the saved devices connected to the Wi-Fi module whenever the user removes them on the MAS android application.

### 3.4.5 Acquisition of Feedback/Notification

#### 3.4.5.1 Description

This feature shall allow the system to provide feedback about the connected devices from the antenna module on the MAS android application

#### 3.4.5.2 Stimulus/Response Sequences

| *3.4.5.2.1* ***Normal Path: Successful Feedback*** |
|---|
| **Pre-conditions:**<br>The user must log in to the application successfully and switch on/off the appliance |
| **Interactions:**<br>The device gets turned on/off |
| **Post-conditions:**<br>The system shall provide positive feedback.<br>The system provides positive feedback by turning the device button green on turn on and red on turn off. |
| **Categorization:**<br>**Criticality:** High<br>**Probability of defects:** Low<br>**Risk:** Medium |

| *3.4.5.2.2* ***Exceptional Path: Feedback Error*** |
|---|
| **Pre-conditions:** |

| |
|---|
| The application is not connected to the internet |
| **Interactions**<br><br>An exception is thrown into the system |
| **Post-conditions:**<br><br>An error message is displayed |
| **Categorization**<br><br>**Criticality:** High<br><br>**Probability of defects:** Low<br><br>**Risk:** High |

#### 3.4.5.3 Functional Requirements

**REQ-1:** System shall connect to the antenna module to provide feedback to the user on android application.

**REQ-2:** System shall display device buttons with particular color according to the current activity.

## 3.4.6 Transmission of data from Wi-Fi to Antenna module

#### 3.4.6.1 Description

This feature shall allow the system to convert digital data to an analog signal. This is done to transmit data from the Wi-Fi module to the antenna module and vice versa.

#### 3.4.6.2 Stimulus/Response Sequences

| |
|---|
| *3.4.6.2.1* ***Normal Path: Successful data transmission to Antenna module*** |

| |
|---|
| **Pre-conditions:**<br><br>User must log in to the application successfully |
| **Interactions:**<br><br>The user shall turn on/off the appliance via button |
| **Post-conditions:**<br><br>System shall send data to the antenna module. |
| **Categorization:**<br><br>**Criticality:** High<br><br>**Probability of defects**: Low<br><br>**Risk:** Medium |

| |
|---|
| *3.4.6.2.2* ***Exceptional Path: An error message is displayed*** |
| **Pre-conditions:**<br><br>The application is not connected to the internet |
| **Interactions**<br><br>An exception is thrown into the system |
| **Post-conditions:**<br><br>An error message is displayed |
| **Categorization**<br><br>**Criticality:** High<br><br>**Probability of defects**: Low<br><br>**Risk:** High |

#### 3.4.6.3 Functional Requirements

**REQ-1**: Android application shall be connected to the Antenna module and Wi-Fi module.

**REQ-2**: System shall send and receive data to and from the android application over Antenna and Wi-Fi modules.

**REQ-3**: Appliance on/off

## 3.5. Other Non-functional Requirements

### 3.5.1 Availability

**NFR-1:** The system shall always be available to receive commands from the user and to turn on/off an appliance.

### 3.5.2 Reliability

**NFR-2:** The Android Application shall be up to date.

The system shall be able to work in a normal way after restarting due to an error.

### 3.5.3 Maintainability

**NFR-3:** The system shall use standard electrical components available in market which may be replaced in case of component failure.

### 3.5.4 Security

**NFR-4:** Android Application shall not allow the user to use the application or system features unless registered and signed in.

### 3.5.5 Environmental

**NFR-5:** The system shall use any component therefore reducing environmental damage.

### 3.5.6 Legal

**NFR-6:** The system will follow the customer privacy policy strictly

### 3.5.7 Usability

The graphical user interface of the MAS is to be designed with usability in such a manner that is both visually appealing and easy for the user to navigate and use the software interface.

### 3.5.8 Performance

- **Response Time**: The load time for user interface screens shall take no longer than one seconds
- **Platform**: The device application shall be compatible with Android.

### 3.5.9 Safety Requirements

The use of the MAS has no harm; nor does it have any possibility of loss or damage to the data of any external device connected to the same Wi-Fi router.

### 3.5.10 Ease of Use

The experts will need a day of training to completely understand the system.

# Chapter 4. Design and Development

## 4.1 Introduction

This chapter describes the architecture and design of MAS (Microcurve Automation System). It tracks the necessary information required to effectively define the architecture and system design to give the development team guidance on the architecture of the system to be developed. It also describes the design of a system fully enough to allow for system development to proceed with an understanding of what is to be built and how it is expected to build.

This document is intended for developers, testers, users, documentation writers, project clients, project supervisors, and project evaluators.

## 4.2 System Architecture Description

This section outlines the software and hardware architecture of the system. It also describes how the functionality and responsibilities of the system are partitioned and assigned to subsystems or components.

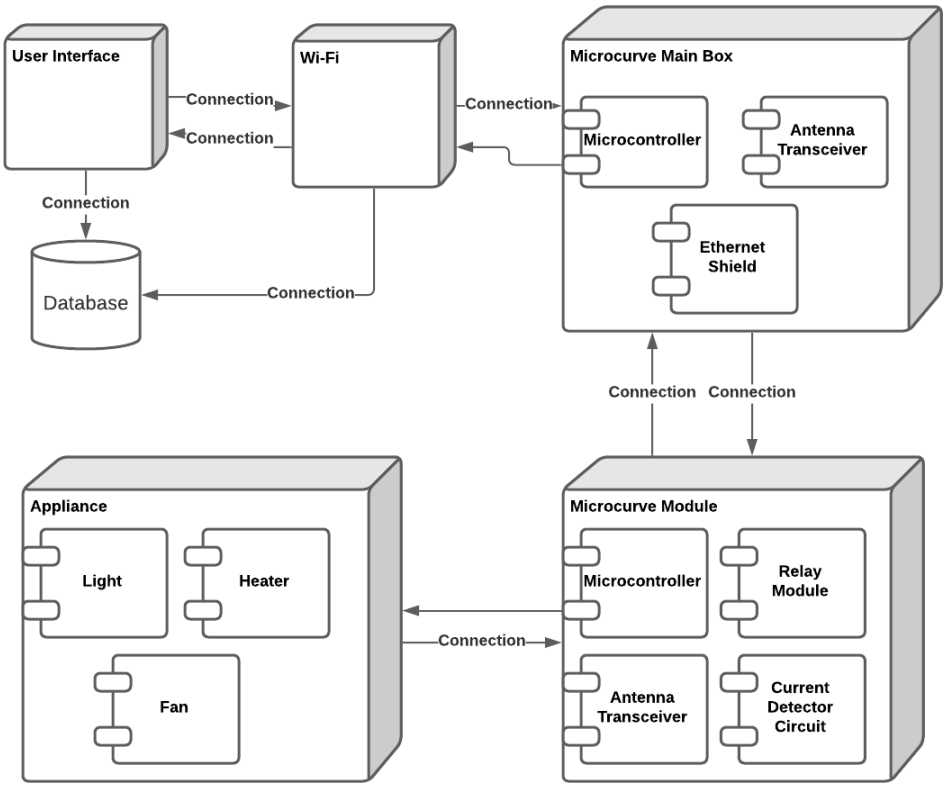


***Figure 2.1*** *Block Diagram showing main modules of Microcurve Automation System along with their components. The connection between each module is shown by arrows.*

## 4.2.1 Architecture Design

The Architectural style for MAS is layered architecture. It includes three layers, i.e., Application layer [MAS android application], Network layer [Wi-Fi module, Antenna Module], Device layer [Switches, Light Bulbs, Fans, etc.]. Each layer works independently and any changes in a particular layer will not affect the working of the rest of layers.

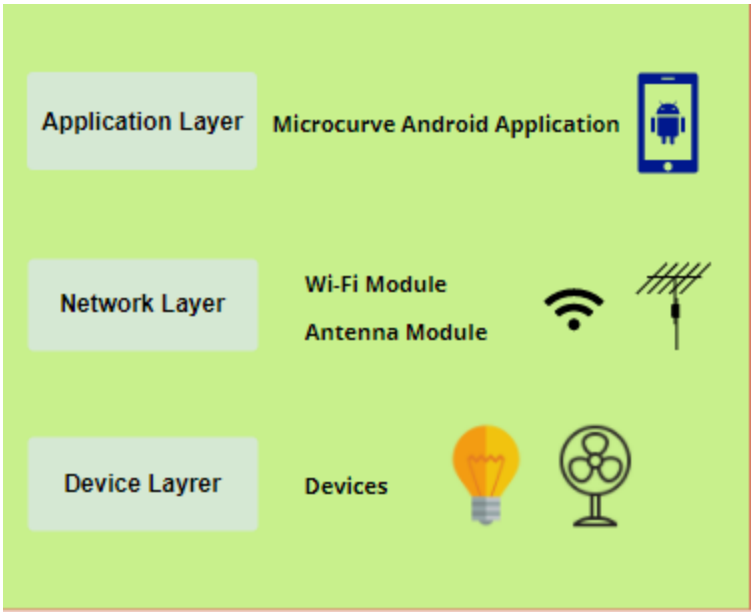


***Figure 2.2*** *Layered Architecture used in Microcurve Automation System. The first layer is the application having MAS android application, the second layer is the network layer having Microcurve module and Wi-Fi Module and the last layer is the device layer consisting of different appliances.*

#### 4.2.1.1 Model View Controller

The software part of the system i.e., the MAS android application will use an MVC architecture which will give modularity to the project files and assures that all the code gets covered in Unit Testing and thus will contribute to enhancing utility. The MVC pattern suggests splitting the code into 3 components, i.e., Model, View, and Controller. We will use the following approach for MVC:

##### 4.2.1.1.1 Model

This component stores the application data. It Is responsible for handling the communication with device and network layers. It is at the back of the software through which data of controller and view will pass through and will be modified.

##### 4.2.1.1.2 Controller

This component establishes the relationship between the View and the Model. It contains the core application logic and gets informed of the user's behavior and updates the Model as per the need.

#### 4.2.1.1.3 View

It is the user interface layer (application layer) that holds components that are visible on the screen. Moreover, it provides the visualization of the data stored in the Model and offers interaction to the user. Designing and color are part of view design. In our system, the buttons, menus, effects and all the design of application the users will see on their system will be part of View. It will collaborate with the controller.

#### 4.2.1.2 Component base Architecture

The overall design of the Microcurve Automation System will be kept component-based to enhance and accommodate changing needs. The detail of this architecture and suggested implementation is discussed in the final section of this chapter.

## 4.2.2 Overview of Modules

"Microcurve Automation System" requires several modules to work. Following is a brief overview of all these modules:

1. **Android Application**

   This subsystem provides an interface to the user for interacting with the automation system and provides the user with features such as registration, signing in, turning on/off

appliances, and adding, removing, rename any device name. This subsystem contains further components:

**a. User Interface**

This component provides all relevant user interfaces and takes care of the view part to provide all the mentioned features of the Android Application.

**b. Controller**

This component takes care of all the business logic and interaction with databases.

**c. Models**

This component takes care of all the business logic and interaction with the database

**2. Automation System**

This subsystem deals with hardware implementation and controls hardware components when the user turns on/off, adds, removes, or renames a device. This Subsystem is further divided into components.

**a. Microcurve Main box**

i. Microcontroller

ii. Antenna Transceiver

iii. Ethernet Shield

**b. Microcurve Module**

i. Microcontroller

ii. Relay Module

iii. Current Detector Circuit

iv. Antenna Transceiver

**c. Appliance**

**3. Central Database**

This component carries a central database hosted on firebase. This is used both by Android applications and automation systems.

### 4.2.2.1 Modules

This section covers detail about the Hardware involved in the project, integration, and assembly of different components in the project “Microcurve Automation System”

#### 4.2.2.1.1 Arduino UNO


#### 4.2.2.1.2 NodeMCU ESP8266

#### 4.2.2.1.3 NFR24L01 + 2.4 GHz Antenna Transceiver

#### 4.2.2.1.4 Single Channel 5V Relay Module


#### 4.2.2.1.5 Resistors

#### 4.2.2.1.6 LEDs

## 4.2.3 Structure and Relationships

This section covers the overall technical description “Microcurve automation system”. It shows the working of applications from the perspective of different viewpoints and shows relationships between different components.

#### 4.2.3.1 System Block Diagram

This diagram shows the higher-level description of the system. It shows major modules of the system and their associations and the flow of data between modules.

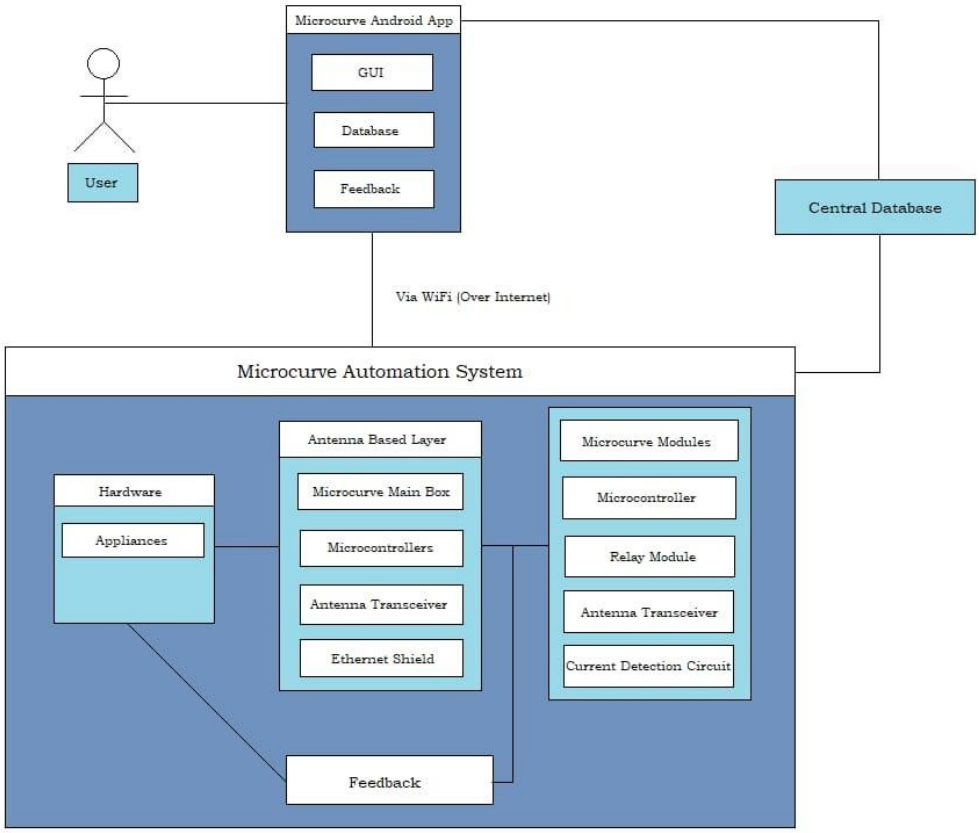


***Figure 2.3*** *System Block diagram for Microcurve Automation system. It shows the interaction between different components of the automation system with the user.*

### 4.2.3.2 User View – Use Case diagram

The following diagram shows the course of events that take place when an actor (User) interacts with the system.

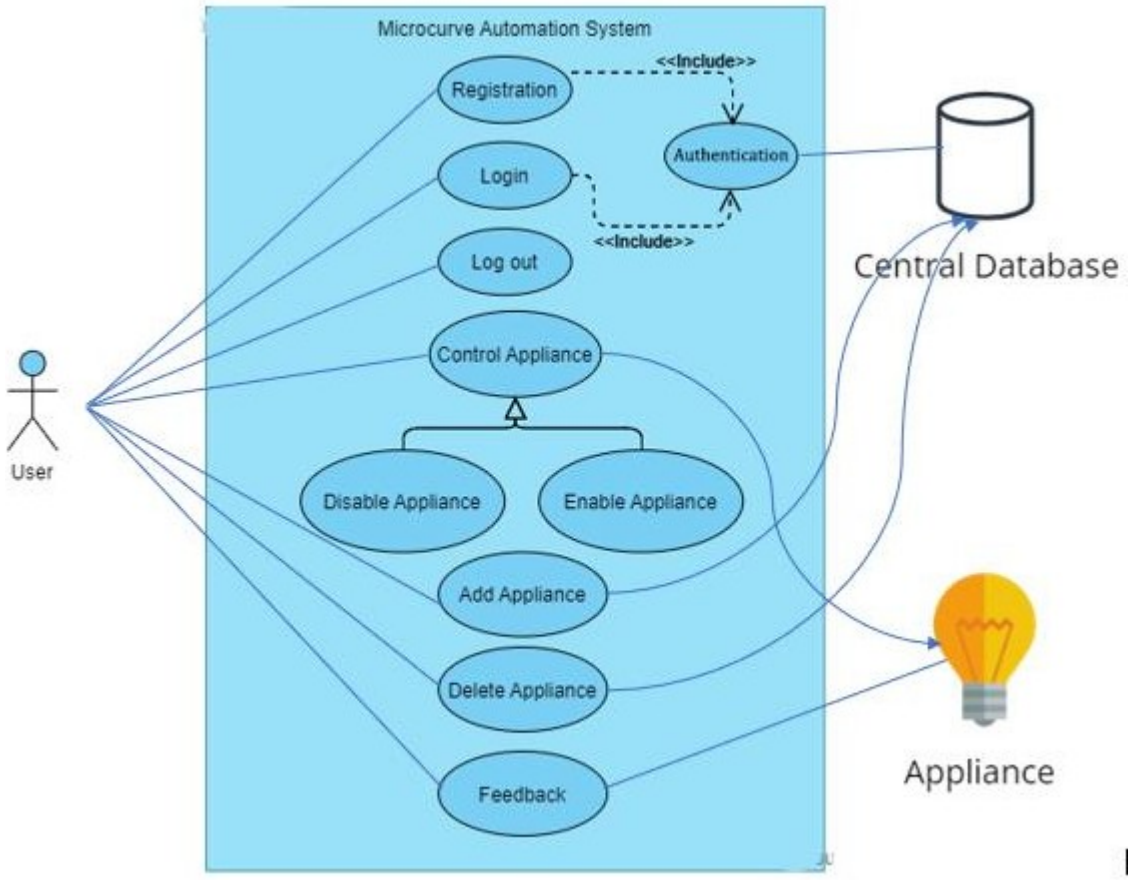


***Figure 2.4*** *Use case diagram of Microcurve automation system. Shows the interaction between the user and different components of the Android Application*.

***4.2.3.2.1.*** **Use Case 1:**

**Registration:**


**Figure 2.5** *Usecase Diagram for Registration to the application. The user is required toenter a username and password which is authenticated by the database. After Registration, the database tables are updated accordingly.*

| | | | |
|---|---|---|---|
| **Use Case ID:** | 1 | | |
| **Use Case Name:** | Registration | | |
| **Actors:** | User, Database | | |
| **Created By:** | Aiman Ali | **Last Updated By:** | Aminah Shahzad |
| **Date Created:** | 17-01-2022 | **Date Last Updated:** | 17-01-2022 |
| **Description:** | A user can sign up to MAS application by entering details on a sign-up page. Necessary details must be filled. The application database will store all the information of the user in the user table and the user's home details in the home table. | | |
| **Preconditions:** | Sign up page is displayed. | | |
| **Post conditions:** | The system redirects to the login page. | | |
| **Normal Flow (primary scenario):** | The application displays login page by default and if user wants to create new account, the sign-up button at the bottom of the login page will direct the user to the sign-up page. The user enters all the necessary details on sign up page. After submitting those details by confirm button at the end, the user waits for the verification of account | | |

| | |
|---|---|
| **Alternative Flows:** | If the user already has an account, the user can login from login interface instead. |
| **Exception flows:** | When Information is incorrectly entered or information is not complete, the application will show an error message. Reload the page and enter the data again. |

#### 4.2.3.2.2 Use case 2: Login

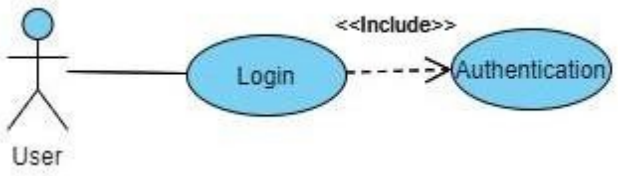


***Figure 2.6*** *Usecase Diagram for Login to the application. The user is required to enter username and password which is authenticated by database. After Login, the database tables are updated accordingly.*

| | | | |
|---|---|---|---|
| **Use Case ID:** | 2 | | |
| **Use Case Name:** | Login to Application | | |
| **Actors:** | User, Database | | |
| **Created By:** | Aiman Ali | **Last Updated By:** | Aminah Shahzad |
| **Date Created:** | 17-01-2022 | **Date Last Updated:** | 17-01-2022 |
| **Description:** | The User enters the login details i.e., a username and a password. The<br>database authenticates the login information of the user. | | |
| **Preconditions:** | Login page is displayed. | | |
| **Post conditions:** | The application displays the Dashboard interface. | | |

| | |
|---|---|
| **Normal Flow (primary scenario):** | The application displays the login page. The user enters login credentials and clicks login button. The database authenticates the login credentials and displays the dashboard interface if the user has entered correct credentials. |
| **Alternative Flows:** | When login credentials are incorrect, the user is allowed to re-enter the credentials. |
| **Exception flows:** | If the user enters an invalid username or an incorrect password, the application displays an error message and goes back to login page and asks user to re-enter correct information. |

#### 4.2.3.2.3 Usecase 3 : Logout

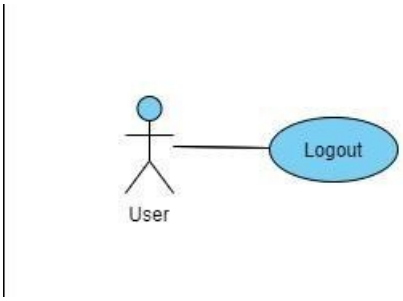


***Figure 2.7*** *Usecase diagram in which user can log out of the system. The database acts as a secondary actor. The user can log out of the system when done using the application.*

| | | | |
|---|---|---|---|
| **Use Case ID:** | 11 | | |
| **Use Case Name:** | Log out | | |
| **Actors:** | User | | |
| **Created By:** | Aiman Ali | **Last Updated By:** | Aminah Shahzad |
| **Date Created:** | 17-01-2022 | **Date Last Updated:** | 17-01-2022 |
| **Description:** | An authorized user can log out of the account. The log in session of the application gets expired. | | |

| | |
|---|---|
| **Preconditions:** | The user is authorized and logged in to the account. |
| **Post conditions:** | The log in session gets expired and the application displays log in interface. |
| **Normal Flow (primary scenario):** | Once the user logs in the account, the dashboard displays log out option. The user clicks on it and the application will display log in interface. |
| **Alternative Flows:** | The dashboard displays various other options, and the user can opt for any of them. |

#### 4.2.3.2.4 Usecase 4: Control Appliance

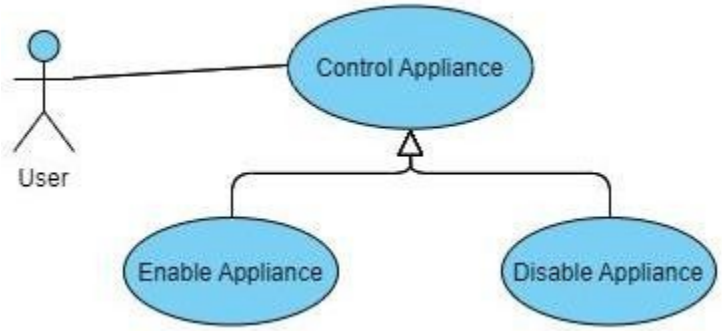


**Figure 2.8** *Usecase Diagram for Controlling Appliance. The enabled feedback is shown as a green signal on the interface while disabled feedback is shown as a red signal on the interface. Arduino controls the feedback status at the back end.*

| | | | |
|---|---|---|---|
| **Use Case ID:** | 9 | | |
| **Use Case Name:** | Enable Appliance | | |
| **Actors:** | User, Arduino, Database | | |
| **Created By:** | Aiman Ali | **Last Updated By:** | Aminah Shahzad |
| **Date Created:** | 17-01-2022 | **Date Last Updated:** | 17-01-2022 |
| **Description:** | An authorized user can turn a device on when the device is connected to Wi-Fi. | | |

| | |
|---|---|
| **Preconditions:** | The user is authenticated and logged in to the account. The particular<br>device that the user wants to turn on is also connected<br>to Wi-Fi. Dashboard is displayed. |
| **Post conditions:** | The application shows a green button with the appliance<br>name which has been turned on, on view appliance interface.<br>The device status on the<br>device table in database gets updated. |
| **Normal Flow** | Once the user's android device is connected to the Wi-Fi router. The |
| **(primary scenario):** | authorized user logs into the account, and the dashboard will display an option of Appliance menu. The user clicks on it and the application displays an interface of Appliance menu. The user can select a particular device from view appliance interface and selects enable option to turn the device on.<br>The Arduino will turn that device on. The application will display a green button with appliance name if the device successfully turns on.<br>The database will update status of the appliance in device table. |
| **Alternative Flows:** | The dashboard displays various other options, and the user can opt for any<br>of them. |
| **Exception flows:** | The user selects an appliance to turn it on, the application displays a red<br>button with device name if the appliance is malfunctioning. |

| | | | |
|---|---|---|---|
| **Use Case ID:** | 10 | | |
| **Use Case Name:** | Disable Appliance | | |
| **Actors:** | User, Arduino, Database | | |
| **Created By:** | Aiman Ali | **Last Updated By:** | Aminah Shahzad |
| **Date Created:** | 17-01-2022 | **Date Last Updated:** | 17-01-2022 |
| **Description:** | An authorized user can turn a device off when the device is connected to<br>WiFi. | | |

| | |
|---|---|
| **Preconditions:** | The user is authenticated and logged in to the account. The particular device that the user wants to turn off is also connected to WiFi.<br>Dashboard<br>is displayed. |
| **Post conditions:** | The application shows a blue button with the appliance name which has been turned off, on view appliance interface. The device status on the<br>device table in database gets updated. |
| **Normal Flow (primary scenario):** | Once the user's android device is connected to the Wi-Fi router. The authorized user logs into the account, the dashboard will display an option of Appliance menu. The user clicks on it and the application displays an interface of Appliance menu. The user can select a particular device from view appliance interface and selects disable option to turn the device off.<br>The Arduino will turn that device off. The application will display a blue button with appliance name if the device successfully turns off. The<br>database will update status of the appliance in device table. |
| **Alternative Flows:** | The dashboard displays various other options, and the user can opt for any<br>of them. |
| **Exception flows:** | The user selects an appliance to turn it off, the application displays a red<br>button with device name if the appliance is malfunctioning. |

#### 4.2.3.2.5 Usecase 5: Manage Appliance

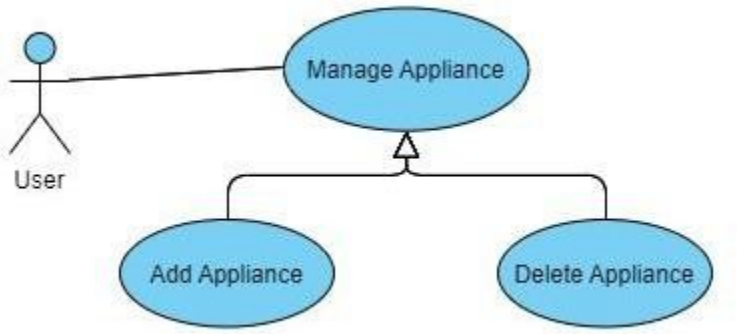


***Figure 2.9*** *Usecase for Managing Appliances. The User can view, delete, and add appliances in appliance interface. Database tables are updated once the operation(s) is performed.*

| **Use Case ID:** | 4 | | |
|---|---|---|---|
| **Use Case Name:** | Add Appliance | | |
| **Actors:** | User, Database | | |
| **Created By:** | Aiman Ali | **Last Updated By:** | Aminah Shahzad |
| **Date Created:** | 17-01-2022 | **Date Last Updated:** | 17-01-2022 |
| **Description:** | An authorized user can add appliances by going to the bottom of Appliance menu where + button is shown. The user clicks on it and the application queries user to add a code of an appliance to be added along with other details about the appliance. The database updated the device table. | | |
| **Preconditions:** | The user is authorized and logged in to the account. Dashboard is displayed. | | |
| **Post conditions:** | The application adds an appliance to the view appliance interface in respective category. The device table in database gets updated. | | |

| | |
|---|---|
| **Normal Flow (primary scenario):** | Once the user logs in the account, the dashboard will display an option of Appliance Menu. The user selects it, and the application will display<br>Appliance menu interface. The user selects + button and then the application will open a prompt and query the user to enter device code along with other appliance details. The user submits the details and then waits for the appliance to get added. |
| **Alternative Flows:** | The dashboard displays various other options, and the user can opt for any of them. |
| **Exception flows** | If the user enters wrong appliance code, then the appliance will not be added, instead the application will generate a message that the appliance is not found. |

| | | | |
|---|---|---|---|
| **Use Case ID:** | 6 | | |
| **Use Case Name:** | Delete Appliance | | |
| **Actors:** | User, Database | | |
| **Created By:** | Aiman Ali | **Last Updated By:** | Aminah Shahzad |
| **Date Created:** | 17-01-2022 | **Date Last Updated:** | 17-01-2022 |

| | |
|---|---|
| **Description:** | An authorized user can delete connected appliances by going to Appliance menu and then selecting a particular appliance from view appliance interface which will display an option to delete that appliance and then<br>save it. That appliance in the device table on the database will be removed. |
| **Preconditions:** | The user is authorized and logged in to the account. Dashboard is displayed. |
| **Post conditions:** | The application removes the particular appliance from view appliance interface when the user saves changes after deleting an appliance.<br>The device table in the database removes the instance for that appliance. |
| **Normal Flow (primary scenario):** | Once the user logs in the account, the dashboard will display an option of Appliance Menu. The user selects it, and the application will display Appliance menu interface. The user selects view appliance by clicking on the type of appliance, which will display an interface with all the appliances from that category connected to MAS. The user selects a particular device which will display an option to delete that appliance. The user selects delete option and then save changes, which will remove the appliance from view appliance interface. |
| **Alternative Flows:** | The dashboard displays various other options, and the user can opt for any of them. |
| **Exception flows:** | If there is no appliance connected to MAS, the view appliance interface will display empty screen. |

#### 4.2.3.2.6 Usecase 6: Feedback

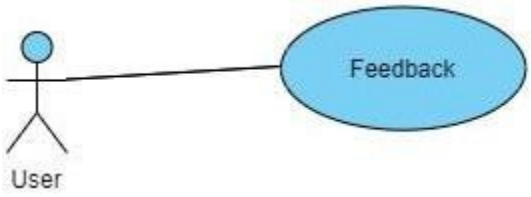


***Figure 2.10*** *Usecase Diagram for Controlling Feedback Status. The enabled feedback is shown as green signal on the interface while disabled feedback is shown as red signal on the interface.*

| | | | |
|---|---|---|---|
| **Use Case ID:** | 6 | | |
| **Use Case Name:** | Feedback | | |
| **Actors:** | User, Appliance | | |
| **Created By:** | Aiman Ali | **Last Updated By:** | Aminah Shahzad |
| **Date Created:** | 17-01-2022 | **Date Last Updated:** | 17-01-2022 |
| **Description:** | When the user turns on/ appliance feedback is given back.<br>The button is red when the user turns off the button and green when the button is turned on. | | |
| **Preconditions:** | The user is authorized and logged in to the account. | | |

| | |
|---|---|
| **Postconditions:** | The appliance turns on/off. |

| | |
|---|---|
| **Normal Flow (primary scenario):** | Once the user logs in to the account, the dashboard will display an option of Appliance Menu. The user selects it, and the application will display the Appliance menu interface. The user selects view appliance by clicking on the type of appliance, which will display an interface with all the appliances from that category connected to MAS. The user can turn on/off the appliance. |
| **Alternative Flows:** | The dashboard displays various other options, and the user can opt for any of them. |
| **Exception flows:** | If there is no appliance connected to MAS, the view appliance interface will display empty screen. |

#### 4.2.3.3 Implementation View – Class diagram

The implementation View of the system can be seen in form of class diagrams for both subsystems as under:

##### 4.2.3.3.1 Android Application's Class diagram

In the diagram below, classes for android applications are shown and their relationships with each other are depicted. The diagram has been made keeping in view MVC architecture.

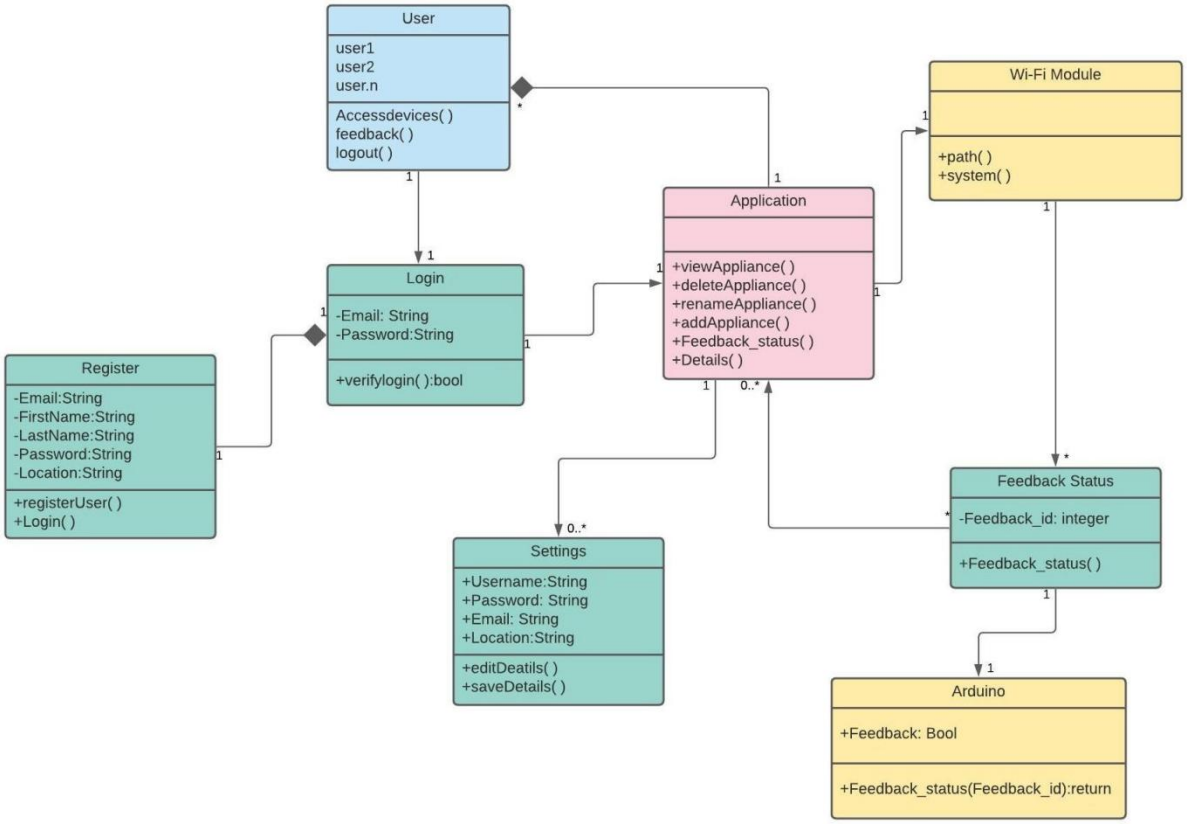


***Figure 2.11*** *Class Diagram of Microcurve Automation System showing main classes of MAS along with their methods and variables. Relationships among each class is also shown in the diagram.*

### 4.2.3.4 Dynamic View – Activity Diagram

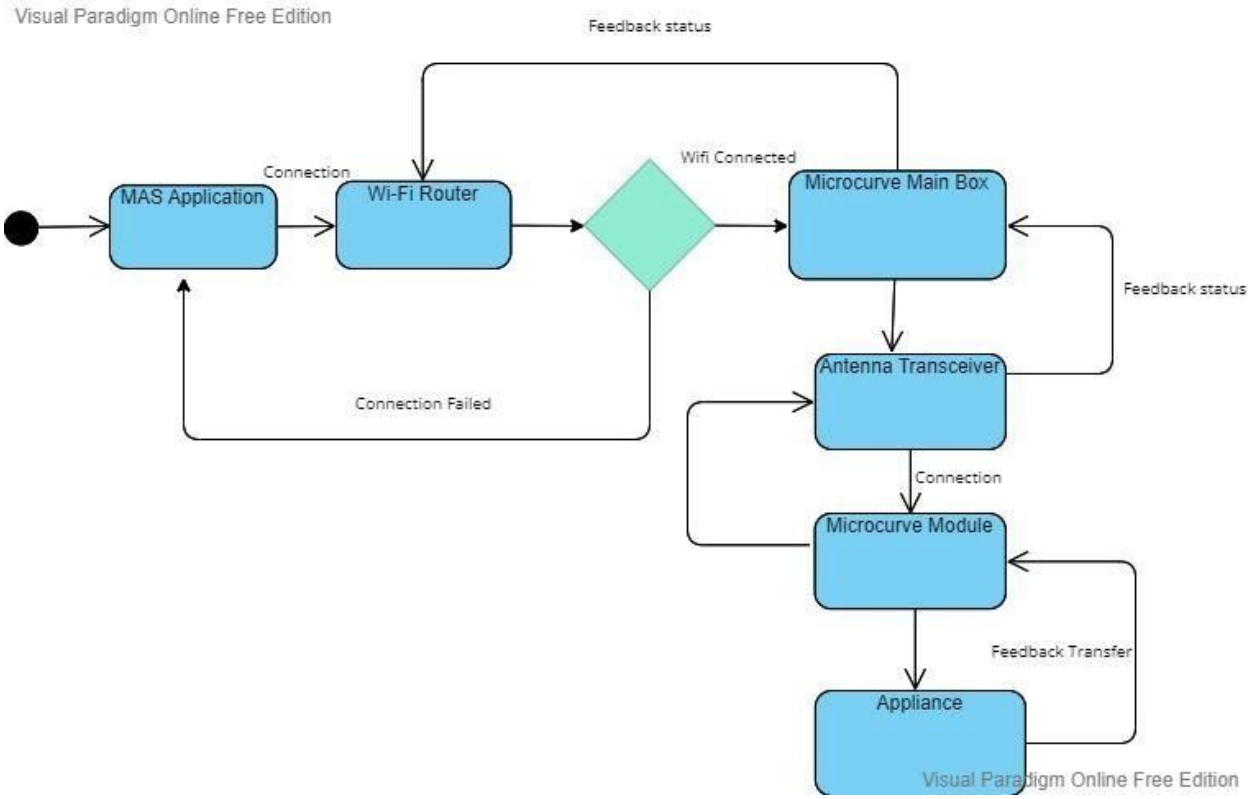


***Figure 2.12*** *Activity Diagram for hardware and software activities of MAS*

### 4.2.3.5 Sequence Diagram

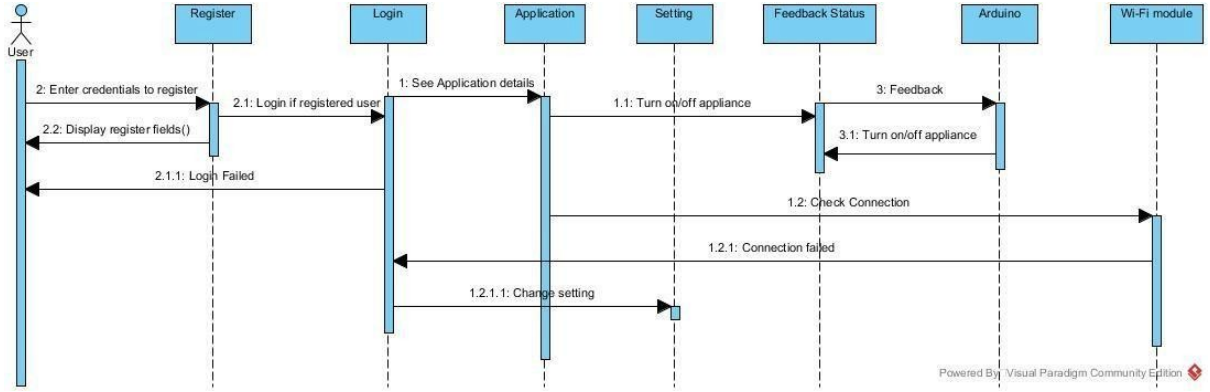


***Figure 2.13*** *Sequence Diagram of Microcurve Automation System showing the sequence of hardware objects interacting with MAS android application.*

## 4.2.4. User Interfaces

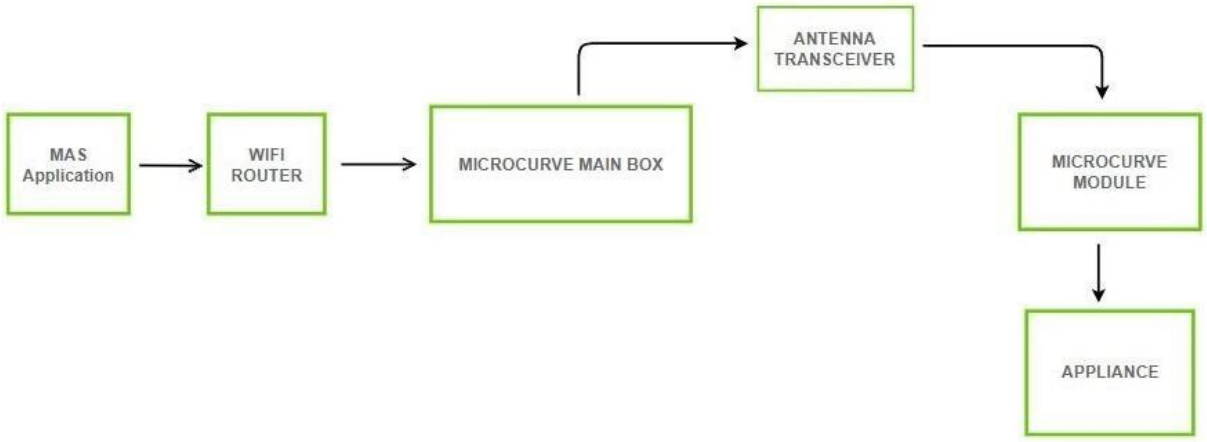


***Figure 2.14*** *Diagram showing different physical components of MAS. It includes an Android application, WIFI module, microcontrollers (each for microcurve box and module), antenna transceivers, and a current detector circuit that detects current and then provides feedback*

# Chapter 5. Project Test and Evaluation

## 5.1 Introduction

This chapter provides the test documentation for project 'Microcurve Automation system', version 1.0 that will facilitate the technical tasks of testing including the detailed test cases for black-box testing. Each test case specifies who will be performing the test, the preconditions required to execute each test case, the specific item to be tested, the input, expected output or results, and procedural steps where applicable. By providing detailed test information, we hope to reduce the probability of overlooking items and improve test coverage. Testers will be able to use each test case provided in this document to move forward and begin testing.

## 5.2 Test Items

Based on the Requirement Specifications Document for the project, "Microcurve Automation System" the following are the major modules/functionalities that are to be considered during the testing phase.

**Mobile Application**

- User Registration
- User Login
- Turn on/off Appliance
- Addition of a new device
- Remove device

**Automation System**

- Transmission of data from WiFi to Antenna Module

## 5.3 Features to be tested

For the two subsystems, Mobile Application & Automation system, the following features are to be tested in Phase-01 for the Testing of the subject project:

1. Mobile Application
2. Automation System

### 5.3.1 Reference

Test items have been stated by, “Requirement Specification Document” for the subject project.

## 5.4 Detailed Test Strategy

The project “Microcurve Automation System” is a computationally intensive system which is why systems modules are being developed independently and then these modules shall be integrated. The major focus has been laid on the development of Modules/Components independently to accommodate future changes without disturbing the already developed system. The overall strategy comprises Unit Testing using White box and Black box testing. Integration testing is performed to successfully integrate the system.

### 5.4.1 Unit Testing

Unit Testing is done at the source or code level for language-specific programming errors such as bad syntax, logic errors, or to test functions or code

modules. The unit test cases shall be designed to test the validity of the program's correctness.

#### 5.4.1.1 White Box Testing

In white-box testing, the UI is bypassed. Inputs and outputs are tested directly at the code level in functions and the results are compared according to requirements. This form of testing ignores the function of the program under test and will focus only on its code and the structure of that code. The test cases that have been generated shall cause each condition to be executed at least once. To ensure this happens, we are applying Basis (alternative) Path Testing. Because the functionality of the program is relatively simple, this method will be feasible to apply

#### 5.4.1.2 Black box testing

Black box testing typically involves running through every possible input to verify that it results in the right outputs using the software as an end-user would.

### 5.4.2 Integration Testing

Integration testing is the part where we will test all previously tested modules in a way that they are functioning normally when they are combined. For complete and accurate functioning of the system as a whole, all modules shall be integrated in specific order keeping a view of their dependency on other features to account for desired results of the system. Integration of the system takes place in two major steps:

1. All modules related to the Android Application shall be integrated and tested.
2. All modules related to the automation system shall be integrated and finally, we shall proceed to system testing.

### 5.4.3 System Testing

In the end, system testing will ensure that all the modules are working, separately and together combined. Then only the outcome of the program will decide the correctness of the whole system

## 5.5 Item Pass/Fail Criteria

Details of the test cases are specified in the section Test Deliverables.

Following are the principles outlined below, a test item would be judged as pass or fail.

- Preconditions are met
- Inputs are carried out as specified
- The result works as what specified in output => Pass
- The system doesn't work or is not the same as output specification => Fail

## 5.6 Suspension Criteria and Resumption Requirements

Testing will be suspended when a defect is introduced/found that cannot allow any further testing. Testing will be resumed after defect removal. This also includes any hardware constraint/failure.

## 5.7. Test Deliverables

Following are the test cases:

Authentication Failed/Registration Failed User enters an email address in an invalid format. Corresponding Output: Error Message Displayed Actual output Confirmed.

| Test Case | Authentication |
|---|---|
| Test Case Number | 1 |

| | |
|---|---|
| Description | This feature enables the user to get registered with the application and provides access to the application's functionality. |
| Testing technique used | Black Box Testing |
| Preconditions | Users should be connected to the internet<br>User should have a Unique Identification Code (provided at the time of purchase of product) |
| Input | User shall input Name, Email, Password, and UID. |
| Steps | Input Email Address & Password will be checked for validity.<br>UID Code will be authenticated using Firebase All input field data will be sent to Firebase and stored.<br>The user shall click the 'Register' button .<br>Credential Token will be saved on the local device for fast access in the future. |
| Expected output | The user is authenticated successfully & is displayed a message.<br>The user is taken to Home Screen |
| Alternative Path | Cause: Authentication Failed/Registration Failed User enters an email address in an invalid format. Corresponding Output: Error Message Displayed |

| | |
|---|---|
| Actual output | Confirmed |

| | |
|---|---|
| Test Case | Authorization |
| Test Case Number | 2 |
| Description | This feature enables the user to log in to the application and provide access to the application's functionality. |
| Testing technique used | Black Box Testing |
| Preconditions | Users should be connected to the internet<br>User should have credentials (Email & Password) |
| Input | User shall input email & password<br>The user should click the login button |
| Steps | Input Email & Password will be checked for validity<br>Input Email address & password will be checked for authenticity from Firebase |
| Expected output | The user is authenticated successfully & is displayed a message.<br>The user is taken to Home Screen |

| | |
|---|---|
| Alternative Path | Cause: Authentication Failed<br>The user enters their email address and password<br>Corresponding Output: Error Message Displayed |
| Actual output | Confirmed |

| | |
|---|---|
| Test Case | Addition/Removing new modules |
| Test Case Number | 3 |
| Description | This feature enables the user to add/remove new devices into the Android Application |
| Testing technique used | Black Box Testing |
| Preconditions | Users should be logged into the system successfully. |
| Input | User shall input a new device<br>User shall remove device from the existing modules<br>User shall rename device from existing modules |
| Steps | Device name will be saved in Firebase database. |
| Expected output | The name is added in database successfully & is displayed a message. |

| | |
|---|---|
| | The user is taken to Dashboard screen. |
| Alternative Path | Cause: Device name not added<br>The user enters a new device name. Corresponding Output: Error Message Displayed |
| Actual output | Confirmed |

| | |
|---|---|
| Test Case | Acquisition of Feedback/Notification |
| Test Case Number | 4 |
| Description | This feature enables the user to get a feedback upon switching on/off an appliance |
| Testing technique used | Black Box Testing |
| Preconditions | Users should be logged into the system successfully. |
| Input | User shall turn on/off an appliance |
| Steps | User shall turn on/off an appliance by clicking the button in Android Application |
| Expected output | The system shall provide positive feedback.<br>System provides positive feedback by turning the device button green when turned on an red when turned off. |

| | |
|---|---|
| Alternative Path | Cause: No positive feedback on turning on/off device<br>Restart Application.<br>Check connectivity<br><br>Corresponding Output: No device turned on/off |
| Actual output | Confirmed |

| | |
|---|---|
| Test Case | Transmission of data from Wi-Fi to antenna module |
| Test Case Number | 5 |
| Description | This feature enables the system to share data coming from the Android application from Wi-Fi module to antenna module. |
| Testing technique used | Black Box Testing |
| Preconditions | Users should be transfer data via Android Application (turn on/off appliance). |
| Input | User shall turn on/off an appliance |

| | |
|---|---|
| Steps | User shall turn on/off an appliance by clicking the button in Android Application. This data is transferred from Android application to Antenna module via Wi-Fi module. |
| | |
| Expected output | The system shall provide positive feedback.<br>System provides positive feedback by turning the device button green when turned on an red when turned off. |
| Alternative Path | Cause: No positive feedback on turning on/off device. Data lost<br>Restart Application.<br>Check connectivity<br>Corresponding Output: No device turned on/off |
| Actual output | Confirmed |

## 5.8 Environmental Needs

### 5.8.1 Software

IDE: Arduino

Android Studio

MIT-App Inventor

Java

## 5.9 Responsibilities, Staffing and Training Needs

### 5.9.1 Responsibilities

All developers of the project are responsible for the completion of all units testing and integration testing tasks.

### 5.9.2 Staffing and Training Needs

Basic knowledge of testing strategies and techniques is needed for the testing of the project. Techniques such as Black Box testing and integration testing should be known to developers. All the developers will be testing each other's work and will be actively participating in the development and testing of the project simultaneously.

## 5.10 Risks and Contingencies

Efforts have been made to remove all and every chance of failure but there are certain unpredictable factors such as network issues, corrupt input data, or system failure that may lead to some issues. Error handling will be applied more deeply to cover all these issues but unforeseen circumstances may happen. Hardware Failure is the biggest risk of all and cannot be for accounted at this instance of project development.

### 5.10.1 Schedule Risk

The project might get behind schedule. So, to complete the project on time, we will need to increase the hours/day.

### 5.10.2 Budget Risk

The budget may not be compensated at this stage of the project. Also, any Hardware Failure may increase the budget.

# Chapter 6. Future Work

# Future Work

This system can be updated in future by using firebase real time database to access the app from anywhere in the world. This will make the core system more reliable, user friendly, and easy to use.

# Chapter 7. Implementation and Code

**IDE: Ardunio Codes**
**NodeMCU**

```
#include <ESP8266WiFi.h>
const char WiFiPassword[] = "12345678";
const char AP_NameChar[] = "MICROCURVE Access Point" ;

WiFiServer server(80);

String request = "";
int LED_Pin = D5;

void setup()
{
  Serial.begin(115200);
  pinMode(LED_Pin, OUTPUT);
  digitalWrite(LED_Pin, LOW);//Turn on LED
  delay(400);
  digitalWrite(LED_Pin, HIGH);//Turn off LED
  WiFi.disconnect();
  boolean conn = WiFi.softAP(AP_NameChar, WiFiPassword);
  server.begin();

}

void loop()
{

  // Check if a client has connected
  WiFiClient client = server.available();
  if (!client)  {
    return;
  }
```

```
// Read the first line of the request
request = client.readStringUntil('\r');

if       ( request.indexOf("LEDON") > 0 )  {
  digitalWrite(LED_Pin, LOW);
  //client.flush();
  client.println("HTTP/1.1 200 OK\r\n");
  client.println( "LEDON");
  client.flush();
  Serial.println("inside ON");
}
else if  ( request.indexOf("LEDOFF") > 0 ) {
  digitalWrite(LED_Pin, HIGH);
  //client.flush();
  client.println("HTTP/1.1 200 OK\r\n");
  client.println( "LEDOFF");
  client.flush();
  Serial.println("inside OFF");
}
delay(5);

}
```

Figure 3.1 NodeMCU code

**Transmitter side Arduino**

```
#include <SPI.h>
    #include <nRF24L01.h>
    #include <RF24.h>
    #include <AESLib.h>
    int pinToRead = 4;
    RF24 radio(7, 8); // CE, CSN
    const byte address[6] = "00001";
    void setup() {

      radio.begin();
      radio.openWritingPipe(address);
      radio.setPALevel(RF24_PA_HIGH); //can set: RF24_PA_MIN, RF24_PA_LOW, RF24_PA_HIGH, RF24_PA_MAX
      radio.setDataRate(RF24_250KBPS); //set as: F24_250KBPS, F24_1MBPS, F24_2MBPS ==>250KBPS = longest range
      radio.setChannel(80); //sets channel from 2.4 to 2.524 GHz in 1 MHz increments 2.483.5 GHz is normal legal limit
      radio.stopListening();
      pinMode(pinToRead, INPUT_PULLUP);
pinMode(3, OUTPUT);
byte key[] = {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15};
void setup() {
  Serial.begin(115200);
    }
    void loop() {
      int a = digitalRead(pinToRead);
        Serial.println("encrypted:");
   if (a == HIGH)
   {
    aes128_enc_single(key, a);
    radio.write(&a, sizeof(a));
   }
   delay(500);
   }      else
     { a = 0;
      aes128_enc_single(key, a);}
        {radio.write(&a, sizeof(a));
        delay(500);
        }
```

Figure 3.2 Code – Transmitter side Arduino

**Receiver Side code for Arduino**

```
#include <SPI.h>
    #include <nRF24L01.h>
    #include <RF24.h>
    #include <AESLib.h>
    RF24 radio(7, 8); // CE, CSN
    const byte address[6] = "00001";
    void setup() {
      Serial.begin(9600);
      radio.begin();
      radio.openReadingPipe(0, address);
      radio.setPALevel(RF24_PA_HIGH); //set as: RF24_PA_MIN, RF24_PA_LOW, RF24_PA_HIGH, RF24_PA_MAX
      radio.setDataRate(RF24_250KBPS); //set as: F24_250KBPS, F24_1MBPS, F24_2MBPS ==>250KBPS = longest range
      radio.setChannel(80); //sets channel from 2.4 to 2.524 GHz in 1 MHz increments 2.483.5 GHz is normal legal limit
      radio.startListening();
      pinMode(3, OUTPUT);
      byte key[] = {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15};
    }
    void loop() {
      int a;
      if (radio.available()) {

        radio.read(&a, key, sizeof(a));}
        aes128_dec_single(key, a);
        if (a == 0)
        {digitalWrite(3, LOW);}
        else
        {digitalWrite(3, HIGH);}
        Serial.print(a);
        delay(500);

          }
```

Figure 3.3 Code – Receiver side code for Arduino

# 8. References

[1] Available: https://create.arduino.cc/projecthub/najad/using-arduino-ide-to-program-nodemcu-33e899#:~:text=Step%201%3A%20Open%20the%20example,builtin%20LED%20should%20start%20blinking.

[2] Available: https://androidexample365.com/connect-nodemcu-to-android-application/

[3] Available: https://www.thomasnet.com/articles/automation-electronics/best-home-automation-systems/#:~:text=A%20home%20automation%20system%20combines,you're%20not%20at%20home.