Smart Aquaponics System

Ubaid Ur Rehman (BSE183056)

Hammad Waseem (BSE183011)

Huraira Arshad (BSE183154)

Supervised By

Mr. Samir Obaid

Spring-2022

BS Software Engineering

Department of Software Engineering

Capital University of Science & Technology, Islamabad

Submission Form for Final-Year

PROJECT REPORT

Version	1

NUMBER OF MEMBERS	3

TITLE	Smart Aquaponics System

SUPERVISOR NAME	Mr. Samir Obaid

MEMBER NAME	REG. NO.	EMAIL ADDRESS
Hammad Waseem	BSE183011	hammadwaseem12@gmail.com
Ubaid Ur Rehman	BSE183056	ubaidur404786@gmail.com
Huraira Arshad	BSE183154	hurairakyani123@gmail.com

MEMBERS' SIGNATURES

_____

_____

_____

Supervisor's Signature

APPROVAL CERTIFICATE

This project, entitled as “Menu Drive (Smart AquaPonics System)” has been approved for the award of

Bachelors of Science in Software Engineering

Committee Signatures:

Supervisor: _____

(Mr. Samir Obaid)

Project Coordinator: _____

(Mr. Ibrar Arshad)

Head of Department: _____

(Dr. Nadeem Anjum)

DECLARATION

We declare that "No portion of the work referred to, in this project has been submitted in support of an application for another degree or qualification of this or any other university/institute or other institution of learning". It is further declared that this undergraduate project, neither as a whole nor as a part thereof has been copied out from any sources, wherever references have been provided.

MEMBERS' SIGNATURES

ACKNOWLEDGEMENTS

We are thankful to Allah Almighty for this blessing that we are now Alhamdulillah Graduating from an honorable institute of Pakistan and last but not the least to our supervisor without his guidance we are unable to complete our project on time. We are thankful to him for being so cooperative and helpful to us. We acknowledge his efforts towards this project and we will always remain thankful to him for his teaching and supervision to us.

Executive Summary

The system provide dashboard on mobile or tablet to user for visualized and analyzed system data on it. Only authorized user can access this system. User can easily control any type of hardware (sensors, controllers, cameras, fans or air conditioner, light) with the help of dashboard. Our services also provide notification system in which system will notify if any hardware discard or damage or if image cannot properly capture (lens issue), when user add in database, payment notification or plant disease detail enter in database system will notify the user for attractive involvement.

Table of Contents

Chapter 1 1
Introduction 1
1.1. Project Introduction 1
1.2. Existing Examples / Solutions 1
1.3. Business Scope 7
1.4. Useful Tools and Technologies 7
1.5. Project Work Break Down 8
1.6. Project Time Line 9
Chapter 2 10
Requirement Specification and Analysis 10
2.1. Functional Requirements 10
2.2. Non-Functional Requirements 12
2.3. Selected Functional Requirements 13
2.4. System Use Case Modeling 15
2.5. System Use Case Description 17
2.6. System Sequence diagram 47
2.7. Domain Model 63
Chapter 3 64
System Design 64
3.1. Layer Definition 64
3.1.1. Presentation Layer 64
3.1.2. Business Logic Layer: 64
3.1.3. Database Layer: 64
3.2. Software Architecture 64
3.3. Class Diagram 69
3.4. Sequence Diagram 70
3.5. Entity Relationship Diagram 76
3.6. Database Schema 77
3.7. User Interface 78
Chapter 4 92
Software Development 92

4.1. Coding Standards 92
4.2. Development Environment 92
4.3. Software Description 92
Chapter 5 101
Software Testing 101
5.2.1. Test Case 1 Register User: 101
5.2.2. Test Case 2 Login user: 103
5.2.3. Test Case 3 Login without valid Data: 104
5.2.4. Test Case 4 Login without entering password: 105
5.2.5. Test Case 5 View Area: 106
5.2.6. Test Case 6 Add Area: 107
5.2.7. Test Case 7 View Sensor: 108
5.2.8. Test Case 8 Add Sensor: 109
5.2.9. Test Case 9 Update Area: 110
5.2.10. Test Case 10 Delete Area: 111
5.2.11. Test Case 11 Update Sensor: 112
5.2.12. Test Case 12 Delete Sensor: 113
5.2.13. Test Case 13 View Device: 114
5.2.14. Test Case 14 Add Device: 115
5.2.15. Test Case 15 Update Device: 116
5.2.16. Test Case 16 Delete Device: 117
5.2.17. Test Case 17 View Service Plain: 118
5.2.18. Test Case 18 Add Service Plain: 119
5.2.19. Test Case 19 Update Service Plain: 120
5.2.20. Test Case 20 Delete Service Plain: 121
5.2.21. Test Case 21 Control Area: 122
5.2.22. Test Case 22 Get Notification: 123
5.2.23. Test Case 23 Smart Automation: 124
Chapter 6 125
Software Deployment 125
6.1. Deployment Process Description 125
References 131

List of Figures

Figure 1. 1 Image processing activity diagram 5
Figure 1. 2 Project Breakdown 8
Figure 1. 3 Project Timeline 9
Figure 2. 1 use case Diagram 15
Figure 2. 2 use case Diagram 2 16
Figure 2. 3 SSD (Sign up) 47
Figure 2. 4 SSD (Log in) 48
Figure 2. 5 SSD (View telemetry data) 49
Figure 2. 6 SSD (Define rule schema) 50
Figure 2. 7 SSD (Add roles and responsibilities) 51
Figure 2. 8 SSD (Manage authorization) 52
Figure 2. 9 SSD (Add sensor) 53
Figure 2. 10 SSD (Schedule task) 54
Figure 2. 11 SSD (Manage devices) 55
Figure 2. 12 SSD (Define area) 56
Figure 2. 13 SSD (Add device) 57
Figure 2. 14 SSD (subscribe Plan) 58
Figure 2. 15 SSD (Add Area) 59
Figure 2. 16 SSD (delete Area) 60
Figure 2. 17 SSD (update area) 61
Figure 2. 18 SSD (view Area) 62
Figure 2. 19 Domain Model 63
Figure 3. 1 Architecture view 65
Figure 3. 2 Image processing activity diagram 66
Figure 3. 3 Image normalization 67
Figure 3. 4 Output image 68
Figure 3. 5 Class Diagram 69
Figure 3. 7 View Area (SD) 70
Figure 3. 8 Update Area (SD) 71
Figure 3. 9 Delete Area (SD) 72
Figure 3. 10 Add Area (SD) 73
Figure 3. 11 Subscribe plan (SD) 74
Figure 3. 12 Add Client (SD) 75
Figure 3. 13 ERD Diagram 76
Figure 3. 14 Database Schema diagram 77
Figure 3. 15 User Dashboard GUI 78
Figure 3. 16 Sign Up GUI 78
Figure 3. 16 View Area GUI 79
Figure 3. 17 Log in interface GUI 79
Figure 3. 18 Add Area Management GUI 80
Figure 3. 19 Delete Area GUI 80
Figure 3. 20 View Sensor Management GUI 81
Figure 3. 21 Add Sensor GUI 81

Figure 3. 22 delete Sensor Management GUI 82
Figure 3. 23 update Sensor GUI 82
Figure 3. 24 add device Management GUI 83
Figure 3. 25 delete device GUI 83
Figure 3. 26 view device Management GUI 84
Figure 3. 27 update device GUI 84
Figure 3. 28 add service plain Management GUI 85
Figure 3. 29 delete service plain GUI 85
Figure 3. 30 view service plain Management GUI 86
Figure 3. 31 update service plain GUI 86
Figure 3. 30 monitor area Management GUI 87
Figure 3. 31 control Area GUI 87
Figure 3. 34 control device Management GUI 88
Figure 3. 35 get notification GUI 88
Figure 4. 1 Post User Data Code 93
Figure 4. 2 Login User Get Code 94
Figure 4. 3 Add Area Code 95
Figure 4. 4 Add Area Code 96
Figure 5. 1 Register user test1 102
Figure 5. 2 Login test2 103
Figure 5. 3 User Login test3 104
Figure 5. 4 Login test4 105
Figure 5. 5 Test 5 View Area 106
Figure 5. 6 Test Add Area 107
Figure 5. 7 Test 7 View Sensor 108
Figure 5. 8 Test 8 Add Sensor 109
Figure 5. 9 Test 9 Update Area 110
Figure 5. 10 Test 10 Delete Area 111
Figure 5. 11 Test 11 Update Sensor 112
Figure 5. 12 Test 12 Delete Sensor 113
Figure 5. 13 Test 7 View Device 114
Figure 5. 14 Test 14 Add Device 115
Figure 5. 15 Test 15 Update Device 116
Figure 5. 16 Test 16 Delete Sensor 117
Figure 5. 17 Test 17 View Service Plain 118
Figure 5. 18 Test 18 Add Service Plain 119
Figure 5. 19 Test 19 Update Service Plain 120
Figure 5. 20 Test 20 Delete Service Plain 121
Figure 5. 21 Test 20 Control Area 122
Figure 5. 22 Test 22 Get Notification 123
Figure 5. 23 Test 23 Set Automation 124
Figure 6. 1 control panel 125
Figure 6. 2 control panel 2 126
Figure 6. 3 control panel 3 126

Figure 6. 4 control panel 4 127
Figure 6. 5 control panel 5 127
Figure 6. 6 control panel 6 128
Figure 6. 7 control panel 7 128
Figure 6. 8 control panel 8 129

List of Tables

Table 1. 1 Comparison IOT TOOLS 4
Table 2. 1 function requirement 10
Table 2. 2 non function requirement 12
Table 2. 3 Selected Set of Requirements for Current Iteration 13
Table 2. 4 use case description sign up 17
Table 2. 5 use case description login user 18
Table 2. 6 use case add area 19
Table 2. 7 use case view area 20
Table 2. 8 use case update area 21
Table 2. 9 use case delete area 22
Table 2. 10 use case description view telemetry data 23
Table 2. 11 use case description define rule schema 24
Table 2. 12 use case description add rule and responsibilities 25
Table 2. 13 use case description manage authorization 26
Table 2. 14 use case description view sensor 27
Table 2. 15 use case description add sensor 28
Table 2. 16. use case description delete sensor 29
Table 2. 17 Use case update sensor 30
Table 2. 18 use case schedule task 31
Table 2. 19 use case control devices 32
Table 2. 20 Use case add devices 33
Table 2. 21 Use case description add service plans 34
Table 2. 22 Use case description delete service plain 35
Table 2. 23 case description update service 36
Table 2. 24 Use case description view service 37
Table 2. 25 Use case description subscribe service 38
Table 2. 26 Use case description enable smart monitor 39
Table 2. 27 Use case description get notification 40
Table 2. 28 Use case description enable telemetry log 41
Table 2. 29 Use case description add crop information 42
Table 2. 30 use case description view device 43
Table 2. 31 Use case description update device 44
Table 2. 32 Use case description delete device 45
Table 2. 30 Use case description add configure telemetry data 46
Table 3. 1 Layers Definition 64
Table 5. 1 Test 1 Register user 101
Table 5. 2 Test 2 Login User 103
Table 5. 3 Test 3 Login without entering Email 104
Table 5. 4 Test 4 Login without entering password 105
Table 5. 5 Test 5 View Area 106
Table 5. 6 Test 6 Add Area 107
Table 5. 7 Test 7 View Sensor 108
Table 5. 8 Test 8 Add Sensor 109

Table 5. 9 Test 9 Update Area 110
Table 5. 10 Test 10 Delete Area 111
Table 5. 11 Test 11 Update Sensor 112
Table 5. 12 Test 12 delete Sensor 113
Table 5. 7 Test 13 View Device 114
Table 5. 8 Test 14 Add Device 115
Table 5. 15 Test 15 Update Device 116
Table 5. 16 Test 16 Delete Device 117
Table 5. 17 Test 17 View Service Plain 118
Table 5. 18 Test 18 Add Service Plain 119
Table 5. 19 Test 19 Update Service Plain 120
Table 5. 20 Test 20 Delete Service Plain 121
Table 5. 21 Test 21 Control Area 122
Table 5. 22 Test 22 Get Notification 123

Chapter 1
Introduction

A system of aquaculture in which the waste produced by farmed fish or other aquatic creatures supplies the nutrients for plants grown hydroponically, which in turn purify the water. AquaPonics system will provide main benefit that you grow own food any time at home or any plant growth organization. Our system cannot depend on climate change, food fertilizer and also no need of any chemical pesticides.

1.1. Project Introduction

Our project is basically to design a middleware of IoT that serves as an interface between components of the IoT, making communication possible among elements that would not otherwise be capable. It connects different already existing programs that were not originally designed to be connected. The main feature of this project to apply image processing with the help of IoT devices which connect hardware to software with advanced technology. This system will provide advanced image processing on plant leaf for detect the diseases with the help of hardware, software's services. Our product will entertain user to connect different types of IoT devices without any advance knowledge.

1.2. Existing Examples / Solutions

There are plenty of existing software that are acting as middleware like Ubidots, Main-Flux, Thing Speak, Things Board, Thing speak, Google IoT but none of these are domain specific. So, the main difference in our software is it is domain specific in agriculture.

In previous existing system the main focus of software is on control environment however system make for plant or fishing growth individually or both for as AquaPonics control environment system in which user can control all hardware through using system. Similarly, many more application exit which perform and produce profit through control environment such as control sensors of temperature. Sensors of light control, motors and many more other sensors who play important role to control the environment. AquaPonics with small scale are appropriate for domestic household. The main purpose of these units is food production for subsistence and domestic use, as many units can have various types of vegetables and herbs growing at once. In the past five years, AquaPonics groups, societies and forums have developed considerably and served to disseminate advice and lessons learned on these small-scale units. You must have an initial startup for this business is high because the profits could not meet the demands of the initial investment plan. Although many academic institutes in the United States of America, Europe and Asia have constructed large units, most have been for academic research rather than food production, and are not intended or designed to compete with other producers in the private sector. There are several successful farms throughout the world. One group of experts in Hawaii (United States of America) has created a fully-fledged commercial system. Another largescale and commercially successful AquaPonics operation is located in Newburgh, New York (United States of America), and reaps profits through multiple revenue streams from diverse fish and vegetable species and a successful marketing strategy to local restaurants, grocery, and health food and farmer's markets [1].

In our AquaPonics updated system we provide services for remotely or at an organization which hardware camera can send image and system will process the image for produce the final result on admin dashboard after that admin will able to decide his decision. Our system provides image processing for leaf deceases and provide accurate solution for increase the production of food. User can subscribe and get fully functional features or any type of notification from this system and also user can control all hardware from user dashboard [2].

- In existing tool **Ubitots** provide only general data visualized on web application without focus on any specific data just like sensors, motors, camera required specific system for visualization in proper format our system specially build for these types of hardware in which user can manage any type of hardware or generate report on them.

Features:

- Data capture.
- data visualization.
- built-in dashboard.
- rules management (or event management).
- graph built

- **Main-flux** is another open-source platform is specially for highly secure open-source user can send or transfer data at any time but our system is mainly focus on only admin access their system and we provide RBAC in which only authorized user can access our system these features can compare with existing tool.

Features:

- Connectivity, Message Routing
- Device Management. & Device Registry.
- Data Management, Data Aggregation, Storage Database
- Application Enablement
- UI-Data Visualization
- Integration Capabilities
- Security

- **Thing Speak** is IoT Cloud platform where you can send sensor data to the cloud. You can also analyze and visualize your data with MATLAB or other software, including making your own applications. Thing Speak includes a Web Service (REST API) that lets you collect and store sensor data in the cloud and develop Internet of Things applications. It works with Arduino, Raspberry Pi and MATLAB.

Features:

- Easily configure devices to send data to Thing-Speak using popular IoT protocols
- Visualize your sensor data in real-time.
- Aggregate data on-demand from third-party sources.
- Use the power of MATLAB to make sense of your IoT data.
- Run your IoT analytics automatically based on schedules or events.

- **Think-Board** is also an open-source IoT tool platform in which data collection, processing, visualization, and device management involved our system also provide these type of services with limited subscription level. Supporting the most popular communication protocols, Things-Board is standard, all-rounder open source lot platform. It will help you collect and store sensor data, define algorithms to process a big amount of data, and set up rules on how the platform should react. You can set triggers for actions to happen when a platform gets certain readings and alarms to get off if necessary. The Things Board also has a pretty nice means to create dashboards and the rules engine can be used by people with little to no programming experience [3].

Features:

- Telemetry Data Collection. Collect and store telemetry data in reliable way, surviving network and hardware failures.
- Multi-tenancy. Support multi-tenant installations out-of-the-box.
- Data Visualization.
- Horizontal scalability.
- IoT Rule Engine.
- Fault-tolerance.
- Device Management.
- Security.

Table 1. 1 Comparison IOT TOOLS

NO	Features	Ubiots	Mainfulx	Think-Speak	Think-Board	Smart AquaPonics
1	Security	✗	✓	✗	✓	✓
2	Dashboard and visualization	✓	✓	✓	✓	✓
3	RBAC	✓	✗	✗	✗	✓
4	Notification	✗	✗	✗	✗	✓
5	Control Hardware	✗	✗	✓	✓	✓
6	Manage hardware devices	✗	✗	✓	✓	✓
7	Image Processing	✗	✗	✗	✗	✓
8	Mobile Interface	✗	✗	✗	✗	✓
9	Database Storage	✓	✓	✓	✓	✓
10	Analyzed Data	✓	✓	✓	✓	✓

Smart AquaPonics Features:

- **Image processing flow chart**

Image processing to figure out plant diseases. For image processing first we need hardware camera device that use for capture plant image then send to web server that is integrate with hardware device. In server side the image processing algorithms apply on plant image in which noise of plant removed by Gaussian filter after this pre-processing apply on filter image in this case Smoothing Filter, RGB to Gray Color Conversion apply on resulted image. After smoothing filter apply the image segmentation in which K-means clustering algorithm use for segmentation after that features extraction using histogram equalization apply for separation of image classification that identify the plant have diseases or it healthy and also provide spray solution to user.

- **Image processing activity diagram**

Image processing to figure out plant diseases. For image processing first we need hardware camera device that use for capture plant image then send to web server that is integrate with hardware device. In server side the image processing algorithms apply on plant image in which noise of plant removed by Gaussian filter after this pre-processing apply on filter image in this case Smoothing Filter, RGB to Gray Color Conversion apply on resulted image. After smoothing filter apply the image segmentation in which K-means clustering algorithm use for segmentation after that features extraction using histogram equalization apply for separation of image classification that identify the plant have diseases or it healthy and also provide spray solution to user.

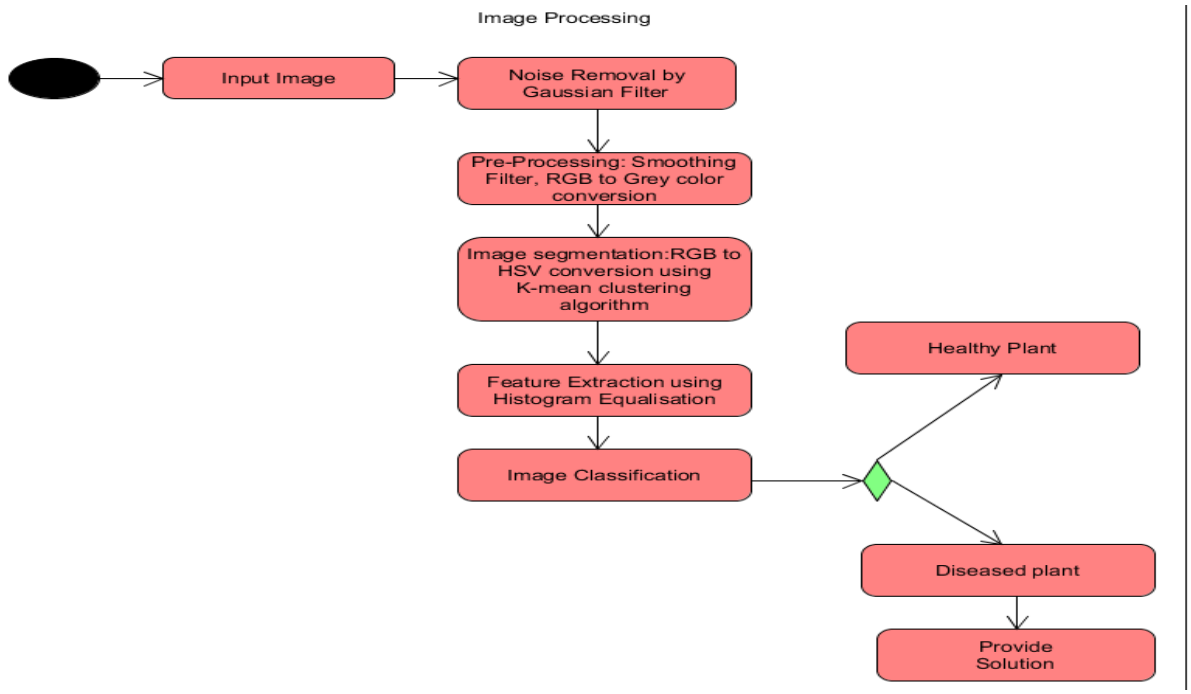

Figure 1. 1 Image processing activity diagram

- **Mobile (Android and IOS) client:**

In this Smart AquaPonics System client interact with Mobile device in which user will be able to use sign in or sign up facility after register account user will be able to add any hardware device (light sensors, temperature sensors, fans, cameras, motors etc.) which relate with this system user also able to delete or update any type of device. User can visualize and analyzed data using Mobile dashboard and also user can generate weekly or monthly report with the help of this system. Any type of alert of damage from device will be notify on mobile device and this system also provide to user to control environment by manually or automate by system.

- **User Subscription:**

Before the use of services user must be a part of account its means he/she purchase membership for using this system features. The main and core image processing services facilitate the user in which services provide only the solution of plant disease or control full environment automatically then no need of any user or admin to control manually. In user subscription the other features in which notification, provide database, security, manage devices, visualization and analyzed data, view record of plant growth from day first to end of growth day system will maintain all record, RBAC, Mobile user friendly interface provide to client.

Dataset:

In Dataset we collect all type of diseases image in which system apply KNN algorithm on train set and generate the solution however it’s may healthy plant or defected plant with their solution.

1.3. Business Scope

This software will be going to be successful in market because it can also be used for general purpose IoT projects and also for specifically project related to agriculture. This project provides services to worldwide companies, If any company or user take picture through camera and send plant image to the system will provide image processing services for controlled agriculture environment then he must need to deployed or purchase our services, with the help of this system notification services, different subscription level facility and rule engine services will provide to subscriber, the system control the whole environment by manually or automate by company user. In daily life plant need for their growth fertilizer, water, proper sunlight, well maintain temperature and also need specific time if farmer want their plant grow naturally but our system not only handle the plant growth it also deals with fishing growth with controlled automated system its save time and cost of person.

Features

- Identify Image (Image processing to detect plant disease)
- Services give only Subscribe users or pay our services
- View Dashboard through mobile view
- User shall be able to control IOT devices through mobile device.
- Add or remove hardware device from mobile.
- System shall notify user for subscribe system (Any problem like sensor is not working)
- Role base access control only authorized people access.

1.4. Useful Tools and Technologies

- C#, Python (backend).
- Java, Flutter, XML (frontend).
- MQTT, HTML (protocols for communication of IoT devices).
- Azure (Deployment).

1.5. Project Work Break Down

The major module of proposed system is that have identified now are gathering of right requirements, analysis and specification of requirement different algorithm that has used to obtain an optimized arrangement at the time of implementation, designing of front-end, linking result with the front- end and testing.

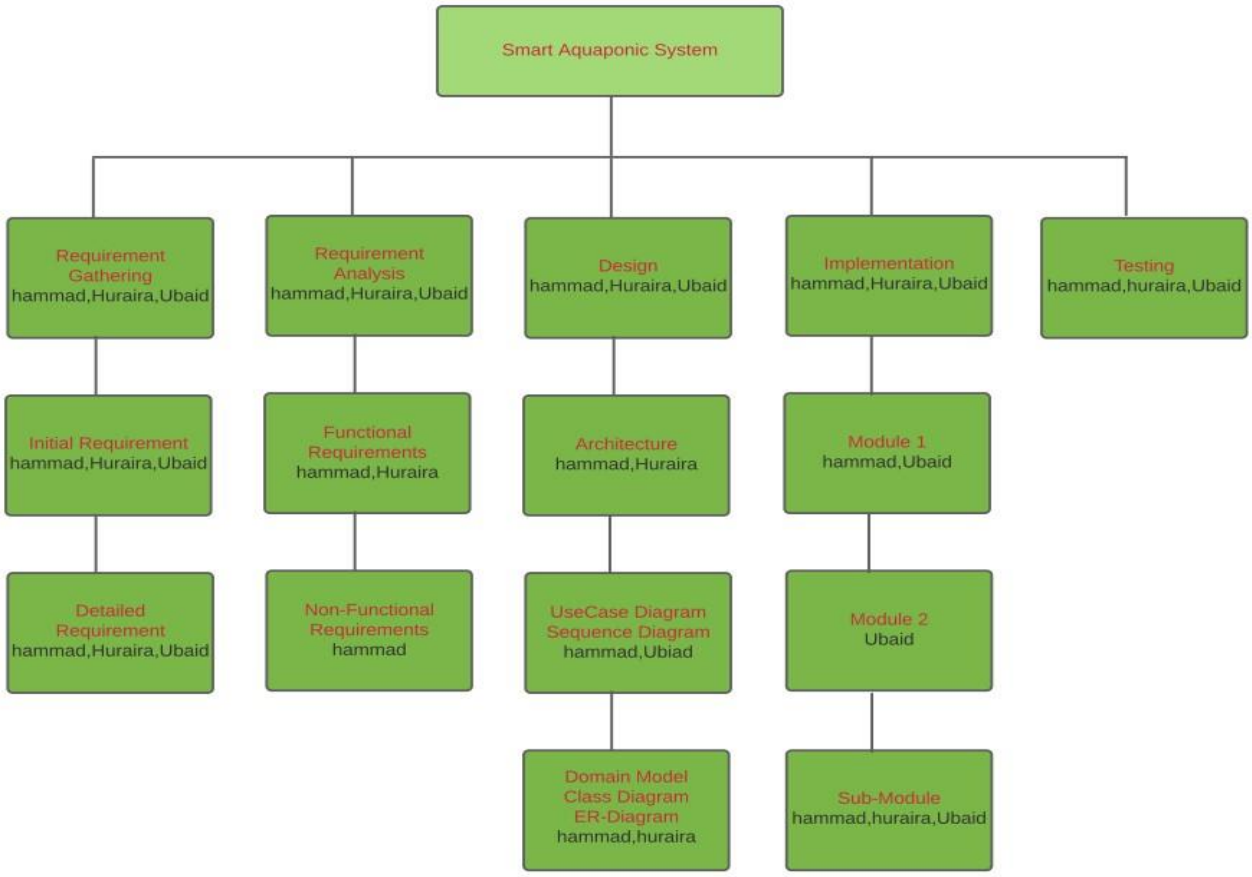

Figure 1. 2 Project Breakdown

1.6. Project Time Line

The Time Line of our project in which we have planned the management of all the major tasks of our project. We have divided our project into five major tasks initially, i.e. requirements elicitation, requirement specification, project design, implementation (coding), and testing.

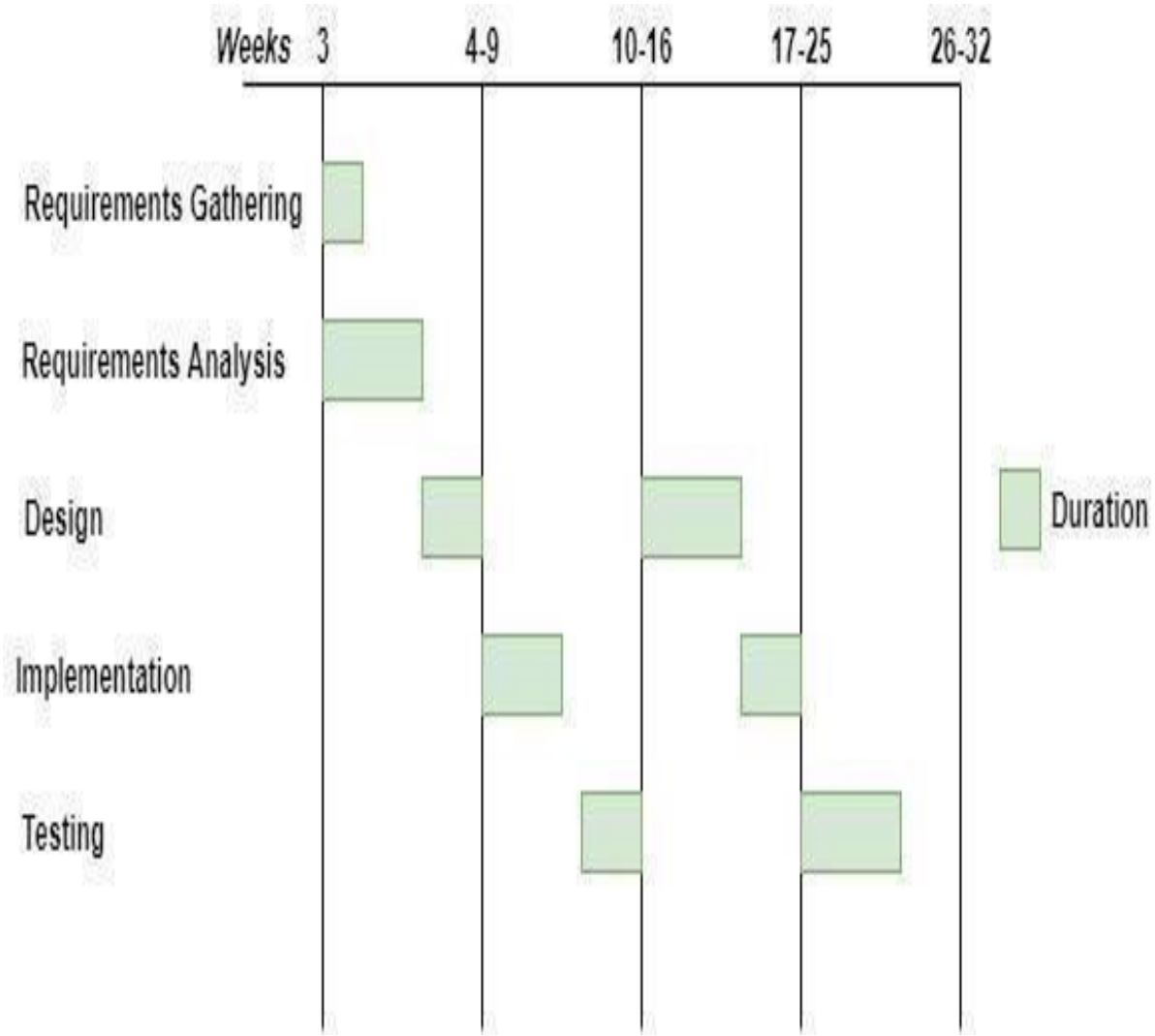

Figure 1. 3 Project Timeline

Chapter 2
Requirement Specification and Analysis

Requirement's analysis is a process of deciding client assumptions for a new or modified product. These highlights, called requirements, should be quantifiable, applicable and nitty gritty. In software engineering, such requirements are frequently called functional specifications. In Chapter 2 we will enlist the functional and non-functional requirements and model functional requirements in the form of use case model.

2.1. Functional Requirements

A functional requirement defines a function of a system. Functional requirements are the specific functionalities that define the behavior of the system and what a system is supposed to accomplish.

Table 2. 1 function requirement

S.No.	Functional Requirement	Status
1.	User should be able to sign up to the system through mobile client.	Completed
2.	User should be able to login to the system through mobile client.	Completed
3.	User should be able to add new Aquaponics Area.	Completed
4.	User should be able to view Aquaponics Area.	Completed
5.	User should be able to update AquaPonics Area.	Completed
6.	User should be able to delete AquaPonics Area.	Completed
7.	User should be able to add AquaPonics Sensor.	Completed
8.	User should be able to view AquaPonics Sensor.	Completed
9.	User should be able to update AquaPonics Sensor.	Completed
10.	User should be able to add AquaPonics Device.	Completed
11.	User should be able to view AquaPonics Device.	Completed
12.	User should be able to update AquaPonics Device.	Completed
13.	User should be able to add AquaPonics Service Plain.	Completed
14.	User should be able to view AquaPonics Service Plain.	Completed
15.	User should be able to update AquaPonics Service Plain.	Completed
16.	User should be able to delete AquaPonics Service Plain.	Completed
17.	User shall be able to control an AquaPonics area with the help of a dashboard. There a user shall be able to change status of devices.	Completed

18.	User should be able to view telemetry data.	Completed
19.	User should be able to get free subscription	Completed
20.	User shall be able to view plants atmosphere by providing plants sensors and getting atmosphere status.	Completed
21.	User should be able to choose automaticity as a service to enable plants growing without human intervention (by the assistance of system).	Completed
22.	User should be able to receive status notification by using mobile and web interface.	Completed

2.2. Non-Functional Requirements

Non-functional requirements include URPS which stands for Usability, Reliability, Performance and Supportability. These requirements specify criteria that can be used to judge the operation of a system, rather than specific behaviors of the system.

Table 2. 2 non function requirement

S.No.	Non-Functional Requirement	Category
1.	Only authorized users shall be able to access the system.	Security
2.	System shall be having usability factor with respect to learnability. For example, there shall be helping frames in order to help a new user to operate the system.	Usability
3.	System shall be having usability factor with respect to interactivity. For example, a system shall notify a user about changes if a user subscribes for notification.	Usability

2.3. Selected Functional Requirements

In our project we're following iterative development model for majority of our functional requirements. Whereas, an incremental will be followed for some of the functional requirements. The table 2.3 describes all the requirements that will be develop iteratively. Whereas, the manage authorization requirement from table 2.1 will be develop in second development cycle.

Table 2. 3 Selected Set of Requirements for Current Iteration

S.No.	Functional Requirement	Status
1.	User should be able to sign up to the system through mobile client.	Completed
2.	User should be able to login to the system through mobile client.	Completed
3.	User should be able to add new AquaPonics Area.	Completed
4.	User should be able to view AquaPonics Area.	Completed
5.	User should be able to update AquaPonics Area.	Completed
6.	User should be able to delete AquaPonics Area.	Completed
7.	User should be able to add AquaPonics Sensor.	Completed
8.	User should be able to view AquaPonics Sensor.	Completed
9.	User should be able to update AquaPonics Sensor.	Completed
10.	User should be able to add AquaPonics Device.	Completed
11.	User should be able to view AquaPonics Device.	Completed
12.	User should be able to update AquaPonics Device.	Completed
13.	User should be able to add AquaPonics Service Plain.	Completed
14.	User should be able to view AquaPonics Service Plain.	Completed

15.	User should be able to update AquaPonics Service Plain.	Completed
16.	User should be able to delete AquaPonics Service Plain.	Completed
17.	User shall be able to control an AquaPonics area with the help of a dashboard. There a user shall be able to change status of devices.	Completed
18.	User should be able to view telemetry data.	Completed
19.	User should be able to get free subscription	Completed
20.	User shall be able to view plants atmosphere by providing plants sensors and getting atmosphere status.	Completed
21.	User should be able to choose automaticity as a service to enable plants growing without human intervention (by the assistance of system).	Completed
22.	User should be able to receive status notification by using mobile and web interface.	Completed

2.4. System Use Case Modeling

A use case is a list of actions or steps of an event, defining the interactions between an actor and a system, to achieve a desired goal. The actor can be a human or other external system. User's use cases are shown in the following the figure.

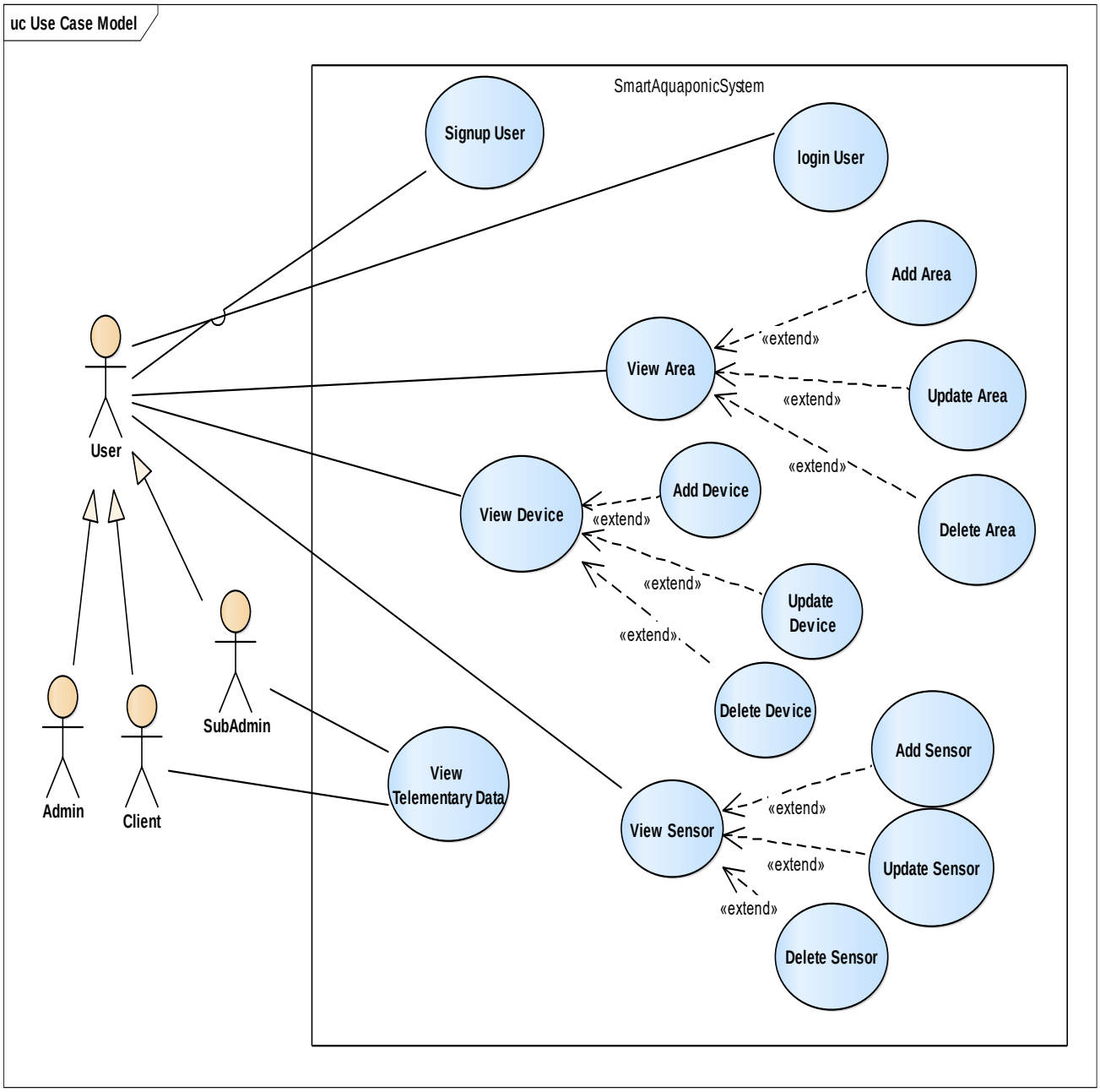

Figure 2. 1 use case Diagram

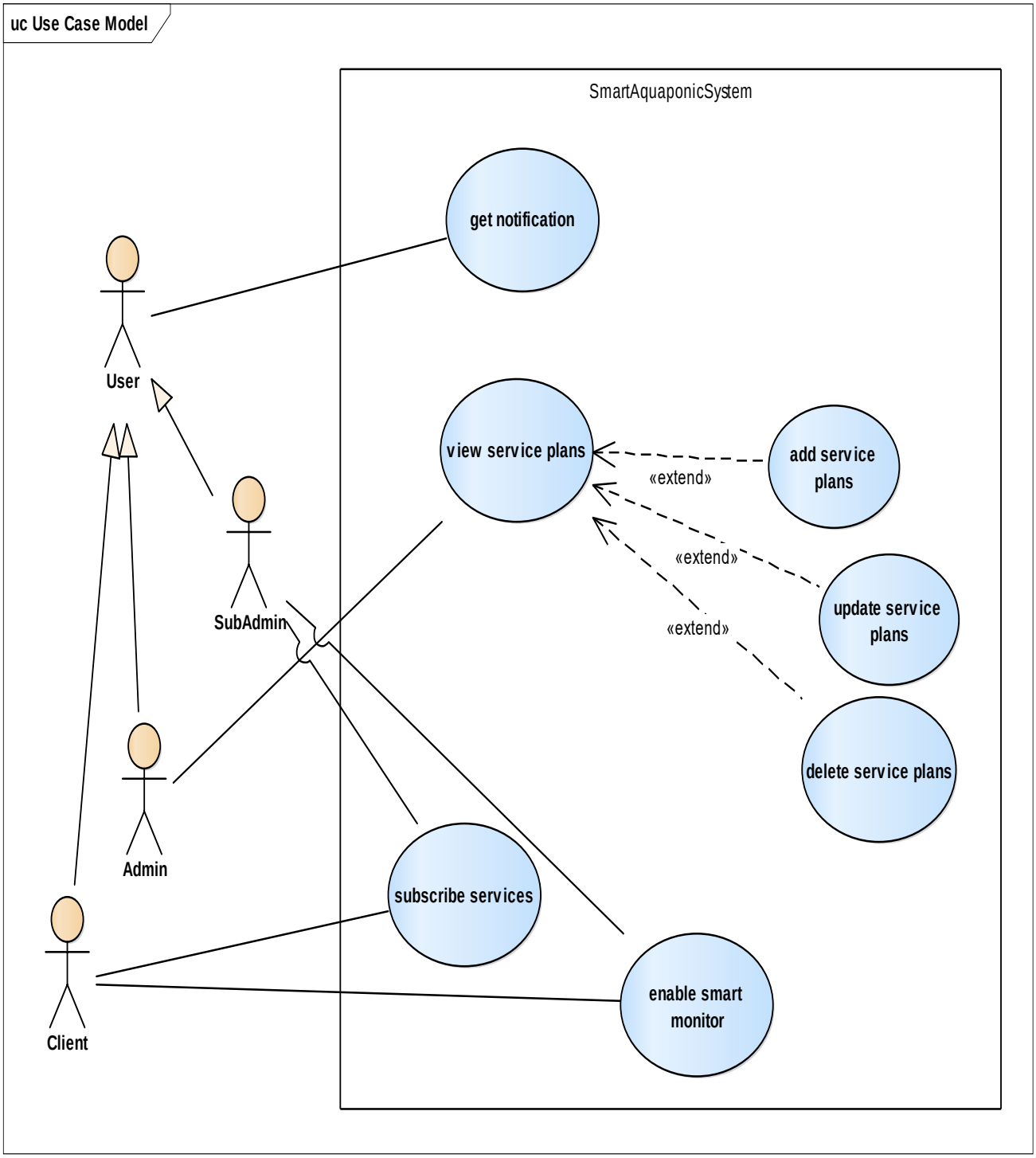

Figure 2. 2 use case Diagram 2

2.5. System Use Case Description

UC1: The users can sign-up into the system by providing username, email and password for signing in into the system.

Table 2. 4 use case description sign up

Use Case ID:	UC1		
Use Case Name:	Sign up User		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Client, Admin		
Description:	The users can sign-up into the system by providing username, email and password for signing in into the system.		
Trigger:	User click on sign-up Button.		
Preconditions:	User must be a new user.		
Post conditions:	User will be signed-up to the system.		
Normal Flow:	**User**	**System**	
	1. User requests to sign-up to the system.	2. System displays sign-up form on user interface.	
	3. User enters appropriate name, username, password , re password and role then press continue.	4. System shows a message of successfully registered if provided data is valid.	
Alternative Flow:	3a. If user enters email / username and password in a wrong format (inappropriate) 1. System prompts user about wrong email or password format. 3b. What happens if user enters email that already exists. 1. System prompts user about that email already exists. 3c. What happens if user not agrees with terms and conditions. 1. System prompts user about agreeing with terms and conditions.		

UC2: The user can sign-in into the system by providing username and password.

Table 2. 5 use case description login user

Use Case ID:	UC2		
Use Case Name:	Login User		
Created By:	Ubaid	**Last Updated By:**	Hammad
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Client, Admin		
Description:	The user can sign-in into the system by providing username and password.		
Trigger:	User gets registered with the system.		
Preconditions:	User should be a registered.		
Post conditions:	User will be signed in to the system.		
Normal Flow:	**User**	**System**	
	1. User requests to login into the system.	2. System displays login form on user interface.	
	3. User enters correct username and password and role.	4. System verifies username and password.	
		5. System displays dashboard of the system on user interface.	
Alternative Flow:	3a. If user enters username and password in a wrong format (inappropriate) 1. System prompts user about wrong username or password format. 3b. What happens if user enters wrong username or password. 1. System prompts user about that username or password entered is incorrect.		

UC3: User will define area for the gadgets to view the telemetry data of the sensors.

Table 2. 6 use case add area

Use Case ID:	UC3		
Use Case Name:	Add Area		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Client, Admin		
Description:	User will define area for the gadgets to view the telemetry data of the sensors.		
Trigger:	User will open dashboard.		
Preconditions:	User should be signed in to the system.		
Post conditions:	Area will be added to the system successfully.		
Normal Flow:	**User**	**System**	
	1. User clicks add button on dashboard.	2. System displays define area form on dashboard.	
	3. User enters title and description of the area and clicks on save button.	4. System displays area added successfully.	
Alternative Flow:	3a. if user enters title of the area that is already defined in the system. 1. System notifies user that the area with similar tittle exists in the system.		

UC4: User will define area for the gadgets to view the telemetry data of the sensors.

Table 2. 7 use case view area

Use Case ID:	UC4		
Use Case Name:	View Area		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Client, Admin		
Description:	User will be able to view area for the gadgets to view the telemetry data of the Areas.		
Trigger:	User will open dashboard.		
Preconditions:	User should be signed in to the system.		
Post conditions:	Area will be display in list form to the user successfully.		
Normal Flow:	**User**	**System**	
	1. User clicks on Area Card on home dashboard.	2. System displays list of areas on display page.	
Alternative Flow:	1a. if the network of server is shutdown during loading of data. 1. System notifies user that the area data cannot load due to internal service issue.		

UC5: User will update area for the gadgets to update the telemetry data of the areas.

Table 2. 8 use case update area

Use Case ID:	UC5		
Use Case Name:	Update Area		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Client, Admin		
Description:	User will update area for the gadgets to view the telemetry data of the sensors.		
Trigger:	User will open dashboard.		
Preconditions:	User should be signed in to the system.		
Post conditions:	Area will be updated to the system successfully.		
Normal Flow:	**User**	**System**	
	1. User clicks on Area Card on home main dashboard.	2. System displays CRUD area dashboard.	
	3. User click update icon from list of particular area.	4. System display all detail in form of particular selected area.	
	5. User enters title and description of the area and clicks on update button.	6. System displays area updated successfully.	
Alternative Flow:	5a. if user enters title of the area that is already defined in the system. 5b. System notifies user that the area with similar tittle exists in the system.		

UC6: User will delete area for the gadgets to delete the telemetry data of the areas.

Table 2. 9 use case delete area

Use Case ID:	UC6		
Use Case Name:	Delete Area		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Client, Admin		
Description:	User will delete area for the gadgets from view the telemetry data of the areas.		
Trigger:	User will open dashboard.		
Preconditions:	User should be signed in to the system.		
Post conditions:	Area will be delete in to the system successfully.		
Normal Flow:	**User**	**System**	
	1. User clicks on Area Card on main dashboard.	2. System displays CRUD area dashboard.	
	3. User click delete icon for particular area.	4. System display all detail in form of area.	
	5. User clicks on delete button.	6. System displays area delete successfully.	
Alternative Flow:	1a. if the network of server is shutdown during loading of data. 1. System notifies user that the area data cannot load due to internal service issue.		

UC7: The user can view readings of sensors.

Table 2. 10 use case description view telemetry data

Use Case ID:	UC7			
Use Case Name:	View Telemetry Data			
Created By:	Ubaid	Last Updated By:		Ubaid
Date Created:	14/12/2021	**Last Revision Date:**		14/12/2021
Actors:	Client, Admin			
Description:	The user can view readings of sensors.			
Trigger:	User clicks on view telemetry data.			
Preconditions:	User should signed-in into the system.			
Post conditions:	User will view selected sensor data.			
Normal Flow:	**User**		**System**	
	1. User requests to view telemetry data by click telemetry card.		2. System displays retrieved data of sensor on user interface	
Alternative Flow:	1a. If user requests telemetry data but there is no data stored in database of respective sensor. 1. System prompts user about that there is no data found.			

UC8: User will define rule schema for the device working.

Table 2. 11 use case description define rule schema

Use Case ID:	UC8		
Use Case Name:	Define Rule Schema		
Created By:	Ubaid	**Last Updated By:**	Huraira
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Client, Admin		
Description:	User will define rule schema for the device working.		
Trigger:	Add rule schema for the device.		
Preconditions:	Device should be connected to the system.		
Post conditions:	Device should work according to defined rules.		
Normal Flow:	**User**	**System**	
	1. User requests to define rule and schema	2. System displays rule and schema on user interface.	
	3. User enters rule scheme for the device by using sensor id and device id.	4. System displays successfully added message on user interface.	
Alternative Flow:	3a. if user enters inappropriate rule scheme 1. System prompts user about wrong data entered. 3b. if user enter wrong sensor id 1. System prompts user about wrong sensor id.		

UC9: User will define rules and responsibilities for the registered users.

Table 2. 12 use case description add rule and responsibilities

Use Case ID:	UC9		
Use Case Name:	Add Rules and Responsibilities		
Created By:	Ubaid	**Last Updated By:**	Huraira
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Admin		
Description:	User will define rules and responsibilities for the registered users.		
Trigger:	User will click on Add roles Button.		
Preconditions:	Users should be registered on the system.		
Post conditions:	Users should work according to their defined roles and responsibilities.		
Normal Flow:	**User**	**System**	
	1. Admin requests to view users list.	2. System displays registered users.	
	3. Admin add roles of users by user id.	4. System shows role added successfully on the user interface.	
	5. Admin add responsibilities of user by user id.	6. System shows responsibilities added successfully on the user interface.	

UC10: User will manage assigned roles and responsibilities.

Table 2. 13 use case description manage authorization

Use Case ID:	UC10		
Use Case Name:	Manage Authorization		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Admin		
Description:	User will manage assigned roles and responsibilities.		
Trigger:	User will click on Manage Roles button.		
Preconditions:	Roles and responsibilities should already assign to the users.		
Post conditions:	Admin will give successfully authorization to other users.		
Normal Flow:	**User**	**System**	
	1. Admin requests to view user by id.	2. System displays user on user interface.	
	3. Admin updates the roles of users.	7. System show a message, role updated successfully on the user interface.	
	4. Admin updates the responsibilities of users.	8. System shows a message, responsibilities updated successfully on the user interface.	
Alternative Flow:	1a. If user requests to view list but there is no data found 1. System prompts user about that there is no list found. 3a. if user not already assigned any role 1. System prompts admin that user is not assigned with any role or responsibility.		

UC11: User will view sensors in the system.

Table 2. 14 use case description view sensor

Use Case ID:	UC11		
Use Case Name:	View Sensor		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Client, Admin		
Description:	User will view sensor in the system.		
Trigger:	User will click on Sensor Card button.		
Preconditions:	Users should login into the system.		
Post conditions:	Sensors display successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to display sensor in the system	2. System displays list of sensor record in array list form successfully.	
Alternative Flow:	1. System cannot load sensor record due to network issue.		

UC12: User will add sensor in the system.

Table 2. 15 use case description add sensor

Use Case ID:	UC12		
Use Case Name:	Add Sensor		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Client, Admin		
Description:	User will add sensor in the system.		
Trigger:	User will click on add sensor button.		
Preconditions:	Users should login into the system.		
Post conditions:	Sensor added successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to add new sensor in the system	2. System displays adds new sensor form on user interface.	
	3. User add new sensor by adding description, protocol, identifier, status its area id and click add button.	4. System shows sensor added successfully message on user interface.	
Alternative Flow:	3a. What happens if user enters telemetry data in wrong format. 1. System prompts user that you entered wrong format of telemetry data. 3b. if the sensor already exists in the system. 1. System notifies user that sensor already exist in the system.		

UC13: User will delete sensor in the system.

Table 2. 16. use case description delete sensor

Use Case ID:	UC13		
Use Case Name:	Delete Sensor		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Client, Admin		
Description:	User will delete sensor in the system.		
Trigger:	User will click on delete sensor button.		
Preconditions:	Users should login into the system.		
Post conditions:	Sensor deleted successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to delete sensor in the system	2. System displays delete sensor form on user interface.	
	3. User delete sensor by its id and press delete button.	4. System shows sensor deleted successfully message on user interface.	
Alternative Flow:	3a. What happens if user enters telemetry data in wrong format. 1. System prompts user that you entered wrong format of telemetry data. 3b. if the sensor is not existing in the system. 1. System notifies user that sensor not exist in the system.		

UC14: User will update sensor in the system

Table 2. 17 Use case update sensor

Use Case ID:	UC14		
Use Case Name:	Update Sensor		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Client, Admin		
Description:	User will update sensor in the system.		
Trigger:	User will click on update sensor button.		
Preconditions:	Users should login into the system.		
Post conditions:	Sensor updated successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to update sensor in the system	2. System displays update sensor form on user interface.	
	3. User update sensor by adding description, protocol, identifier, status and its area id and press update button.	4. System shows sensor updated successfully message on user interface.	
Alternative Flow:	3a. What happens if user enters telemetry data in wrong format. 1. System prompts user that you entered wrong format of telemetry data. 3b. if the sensor already exists in the system. 1. System notifies user that sensor already exist in the system.		

UC15: User will schedule task by defining date and time.

Table 2. 18 use case schedule task

Use Case ID:	UC15		
Use Case Name:	Schedule Task		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Client, Admin		
Description:	User will schedule task by defining date and time.		
Trigger:	User will click on Schedule task Button.		
Preconditions:	Task were defined in the system.		
Post conditions:	System will perform task according to the schedule set by user.		
Normal Flow:	**User**	**System**	
	1. User requests to view task list.	2. System displays task list on user interface.	
	3. User schedules the tasks date and time by using task id.	4. System displays schedule set successfully on user interface.	
Alternative Flow:	1a. if there is no task in the system. 1. System notifies user that there is no task defined in the system. 3a. if user enters previous date. 1. System prompts user that this date is pasted. Please enter a new date.		

UC16: User will manage devices attached to the system.

Table 2. 19 use case control devices

Use Case ID:	UC16		
Use Case Name:	Control Area		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Client, Admin		
Description:	User will manage devices attached to the system		
Trigger:	User will on Control Device Button.		
Preconditions:	Devices should be added in the system.		
Post conditions:	Device status will be updated successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to view device by a specific device id.	2. System displays device information on user interface.	
	3. User changes the status of a device.	4. System displays status updated successfully.	
Alternative Flow:	1a. if no device is attach to the system with respective device id. 1. System notifies user about no device found.		

UC17: User will add new device in the system.

Table 2. 20 Use case add devices

Use Case ID:	UC17		
Use Case Name:	Add Device		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Client, Admin		
Description:	User will add new device in the system.		
Trigger:	User will click on add new device button.		
Preconditions:	Users should login into the system.		
Post conditions:	Device added successfully in the system.		
Normal Flow:	**User**	**System**	
	1. User requests to add new device in the system.	2. System displays add new device form on user interface.	
	3. User add new device by adding description, protocol, identifier and status.	4. System shows device added successfully message on user interface.	
Alternative Flow:	3a. What happens if user enters status in wrong format. 1. System prompts user that you entered wrong format of status of the device (inappropriate). 3b. if the device already exists in the system. 1. System notifies user that device already exist in the system.		
Special Requirements:	The system shall help user to understand the format for status of the device to be entered (Usability with respect to learnability).		

UC18: User will add service plan in the system.

Table 2. 21 Use case description add service plans

Use Case ID:	UC18		
Use Case Name:	Add service plans		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Admin		
Description:	User will add service plan in the system.		
Trigger:	User will click on add service plans button.		
Preconditions:	Users should login into the system.		
Post conditions:	service plans added successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to add new service plans in the system	2. System displays adds new service plans form on user interface.	
	3. User add new service plans by adding id, description, and type data.	4. System shows service plans added successfully message on user interface.	
Alternative Flow:	3a. What happens if user enters telemetry data in wrong format. 1. System prompts user that you entered wrong format of telemetry data. 3b. if the sensor already exists in the system. 1. System notifies user that sensor already exist in the system.		

UC19: User will delete service plans in the system.

Table 2. 22 Use case description delete service plain

Use Case ID:	UC19		
Use Case Name:	Delete service plans		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Admin		
Description:	User will delete service plans in the system.		
Trigger:	User will click on delete service plans button.		
Preconditions:	Users should login into the system.		
Post conditions:	service plans deleted successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to delete service plans in the system	2. System displays delete service plans form on user interface.	
	3. User delete service plans by adding id.	4. System shows service plans deleted successfully message on user interface.	
Alternative Flow:	3a. What happens if user enters telemetry data in wrong format. 1. System prompts user that you entered wrong format of telemetry data. 3b. if the sensor is not existing in the system. 1. System notifies user that sensor not exist in the system.		

UC20: User will update service plans in the system.

Table 2. 23 case description update service

Use Case ID:	UC20		
Use Case Name:	Update service plans		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Admin		
Description:	User will update service plans in the system.		
Trigger:	User will click on update service plans button.		
Preconditions:	Users should login into the system.		
Post conditions:	service plans updated successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to update service plans in the system	2. System displays update service plans form on user interface.	
	3. User update service plans by adding id, description and type data.	4. System shows service plans updated successfully message on user interface.	
Alternative Flow:	3a. What happens if user enters telemetry data in wrong format. 1. System prompts user that you entered wrong format of telemetry data. 3b. if the service already exists in the system. 1. System notifies user that sensor already exist in the system.		

UC21: User will view service plans in the system.

Table 2. 24 Use case description view service

Use Case ID:	UC21		
Use Case Name:	view service plans		
Created By:	Ubaid	**Last Updated By:**	Hammad
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Admin		
Description:	User will view service plans in the system.		
Trigger:	User will click on view service plans button.		
Preconditions:	Users should login into the system.		
Post conditions:	service plans view successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to view service plans in the system	2. System displays view service plans list on user interface.	
Alternative Flow:	3a. What happens if user enters telemetry data in wrong format. 1. System prompts user that you entered wrong format of telemetry data. 3b. if the service cannot exists in the system. 1. System notifies user that cannot exist in the system.		

UC22: User will subscribe services plans in the system.

Table 2. 25 Use case description subscribe service

Use Case ID:	UC22		
Use Case Name:	Subscribe services		
Created By:	Ubaid	**Last Updated By:**	Hammad
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	User		
Description:	User will subscribe services plans in the system.		
Trigger:	User will click on subscribe service plans button.		
Preconditions:	Users should login into the system.		
Post conditions:	service plans subscribe successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to subscribe service plans in the system	2. System displays service plans list on user interface.	
	3. User select appropriate service plans	4. System shows payment form	
	5. User enter card no , CVV, expr date and press pay	6. System shows payment process successfully done	
Alternative Flow:	3a. What happens if user enters card data in wrong format. 1. System prompts user that you entered wrong format of telemetry data. 3b. if the plain already exists in the system. 1. System notifies user that plain already exist in the system.		

UC23: User will enable smart monitor in the system.

Table 2. 26 Use case description enable smart monitor

Use Case ID:	UC23		
Use Case Name:	Enable smart monitor		
Created By:	Ubaid	**Last Updated By:**	Hammad
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	User		
Description:	User will enable smart monitor in the system.		
Trigger:	User will click on enable smart monitor plans button.		
Preconditions:	Users should login into the system.		
Post conditions:	enable smart monitor subscribe successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to enable smart monitor in the system	2. System displays enable smart monitor successfully done.	
Alternative Flow:	1a. if the enable smart monitor already exists in the system. 1. System notifies user that enable smart monitor already exist in the system.		

UC24: User will get notification in the system.

Table 2. 27 Use case description get notification

Use Case ID:	UC24		
Use Case Name:	Get Notification		
Created By:	Ubaid	**Last Updated By:**	Hammad
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	User		
Description:	User will get notification in the system.		
Trigger:	User will click on get notification plans button.		
Preconditions:	Users should login into the system.		
Post conditions:	get notification viewed successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to get notification in the system	2. System show notification list successfully done.	
Alternative Flow:	1a. if the get notification already get in the system. 1. System notifies user that enable notification already exist in the system.		

UC25: User will enable telemetry log in the system.

Table 2. 28 Use case description enable telemetry log

Use Case ID:	UC25		
Use Case Name:	Enable telemetry log		
Created By:	Ubaid	**Last Updated By:**	Hammad
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Admin		
Description:	User will enable telemetry log in the system.		
Trigger:	User will click on enable telemetry log plans button.		
Preconditions:	Users should login into the system.		
Post conditions:	enable telemetry log successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to enable telemetry log in the system	2. System displays enable telemetry log successfully on user interface.	
Alternative Flow:	1a. if the enable telemetry log already exists in the system. 1. System notifies user that enable telemetry log already exist in the system.		

UC26: User will add crop information in the system.

Table 2. 29 Use case description add crop information

Use Case ID:	UC26		
Use Case Name:	Add crop information		
Created By:	Ubaid	**Last Updated By:**	Hammad
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Admin		
Description:	User will add crop information in the system.		
Trigger:	User will click on add crop information log plans button.		
Preconditions:	Users should login into the system.		
Post conditions:	add crop information log successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to add crop information in the system	2. System displays add crop information form on user interface.	
	3. User enter id, description and press done	4. System shows displays add crop information successfully done	
Alternative Flow:	3a. What happens if user enters to add crop information data in wrong format. 1. System prompts user that you entered wrong format of telemetry data. 3b. if the to add crop information already exists in the system. 1. System notifies user that to add crop information already exist in the system.		

UC27: User will view device in the system.

Table 2. 30 use case description view device

Use Case ID:	UC27		
Use Case Name:	View Device		
Created By:	Ubaid	**Last Updated By:**	Ubaid
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Client, Admin		
Description:	User will view device in the system.		
Trigger:	User will click on device Card button.		
Preconditions:	Users should login into the system.		
Post conditions:	Device display successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to display device in the system	2. System displays list of device record in array list form successfully.	
Alternative Flow:	2. System cannot load device record due to network issue.		

UC28: User will be able to update device .

Table 2. 31 Use case description update device

Use Case ID:	UC28		
Use Case Name:	Update Device		
Created By:	Ubaid	**Last Updated By:**	Hammad
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	User		
Description:	User will be able to update device.		
Trigger:	User will click update device button.		
Preconditions:	Users should login into the system.		
Post conditions:	Update device data successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to update device data in the system	2. System displays list of devices to user.	
	3. User enter id, description ,name, status and press done	4. System shows displays update device successfully done	
Alternative Flow:	3a. What happens if user enters to configure telemetry data in wrong format. 1. System prompts user that you entered wrong format of telemetry data. 3b. if the to add configure telemetry data information already exists in the system. 1. System notifies user that to configure telemetry data information already exist in the system.		

UC29: User will be able to delete device .

Table 2. 32 Use case description delete device

Use Case ID:	UC29		
Use Case Name:	delete Device		
Created By:	Ubaid	**Last Updated By:**	Hammad
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	User		
Description:	User will be able to delete device.		
Trigger:	User will click delete device button.		
Preconditions:	Users should login into the system.		
Post conditions:	delete device data successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to delete device data in the system	2. System displays list of devices to user.	
	3. User enter id and press done	4. System shows displays delete device successfully done	

UC30: User will configure telemetry data in the system.

Table 2. 33 Use case description add configure telemetry data

Use Case ID:	UC30		
Use Case Name:	Configure telemetry data		
Created By:	Ubaid	**Last Updated By:**	Hammad
Date Created:	14/12/2021	**Last Revision Date:**	14/12/2021
Actors:	Admin		
Description:	User will configure telemetry data in the system.		
Trigger:	User will click on configure telemetry data log plans button.		
Preconditions:	Users should login into the system.		
Post conditions:	configure telemetry data successfully.		
Normal Flow:	**User**	**System**	
	1. User requests to configure telemetry data in the system	2. System displays configure telemetry data form on user interface.	
	3. User enter id, description ,time interval and press done	4. System shows displays configure telemetry data successfully done	
Alternative Flow:	3a. What happens if user enters to configure telemetry data in wrong format. 1. System prompts user that you entered wrong format of telemetry data. 3b. if the to add configure telemetry data information already exists in the system. 1. System notifies user that to configure telemetry data information already exist in the system.		

2.6. System Sequence diagram

This is login system sequence diagram in which we request to signup then system display signup form to user and then user fill all credentials and submit then system responds to us that you are successfully signup for the system.

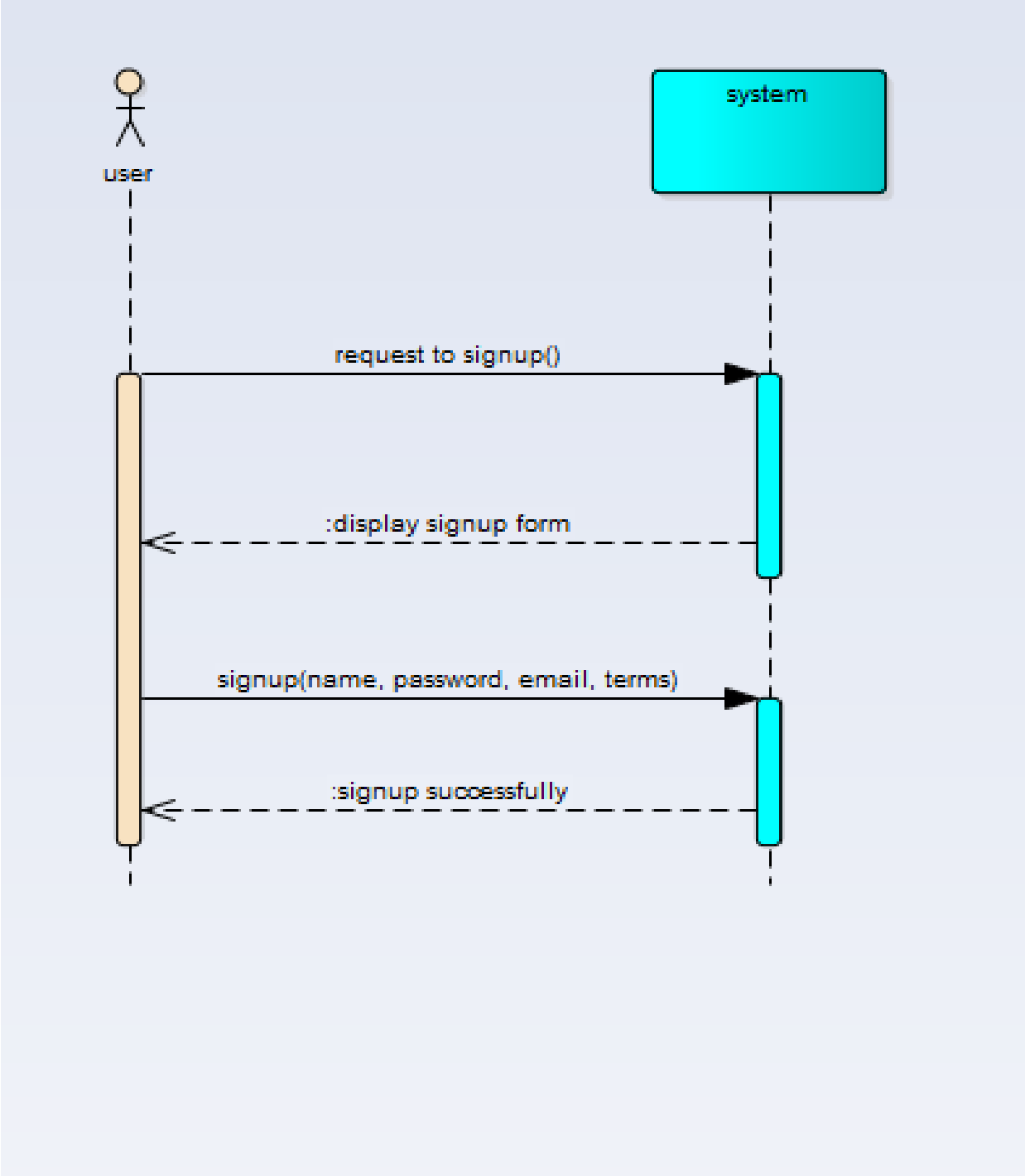

Figure 2. 3 SSD (Sign up)

This is log in system sequence diagram in which we request to login to the system then system display login form to user and then user fill all the credentials and submit then system responds that you are successfully logged in to the system.

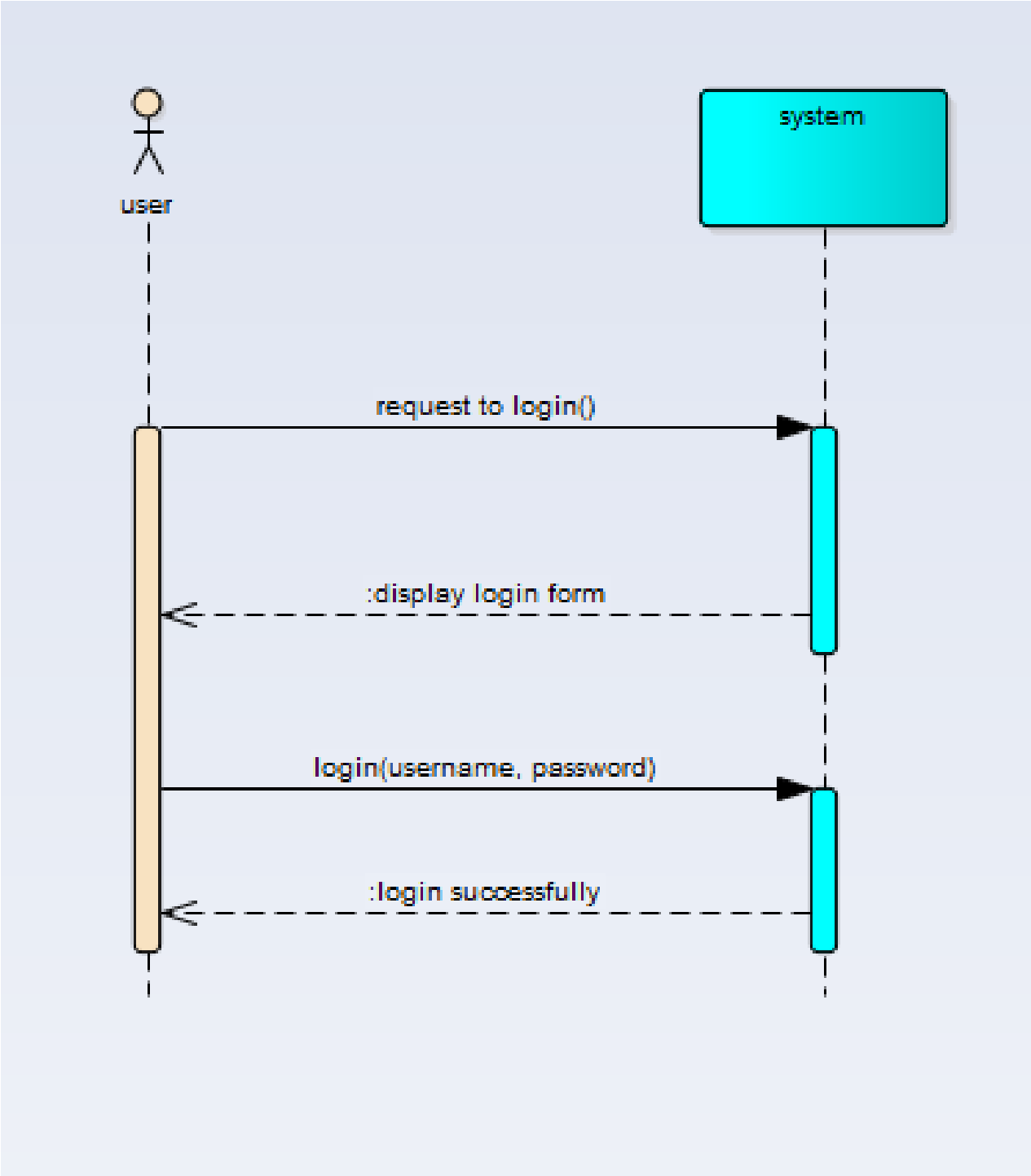

Figure 2. 4 SSD (Log in)

This is View telemetry data system sequence diagram in which we request to view telemetry data then system retrieve sensor data from database and then system displayed all telemetry data to user.

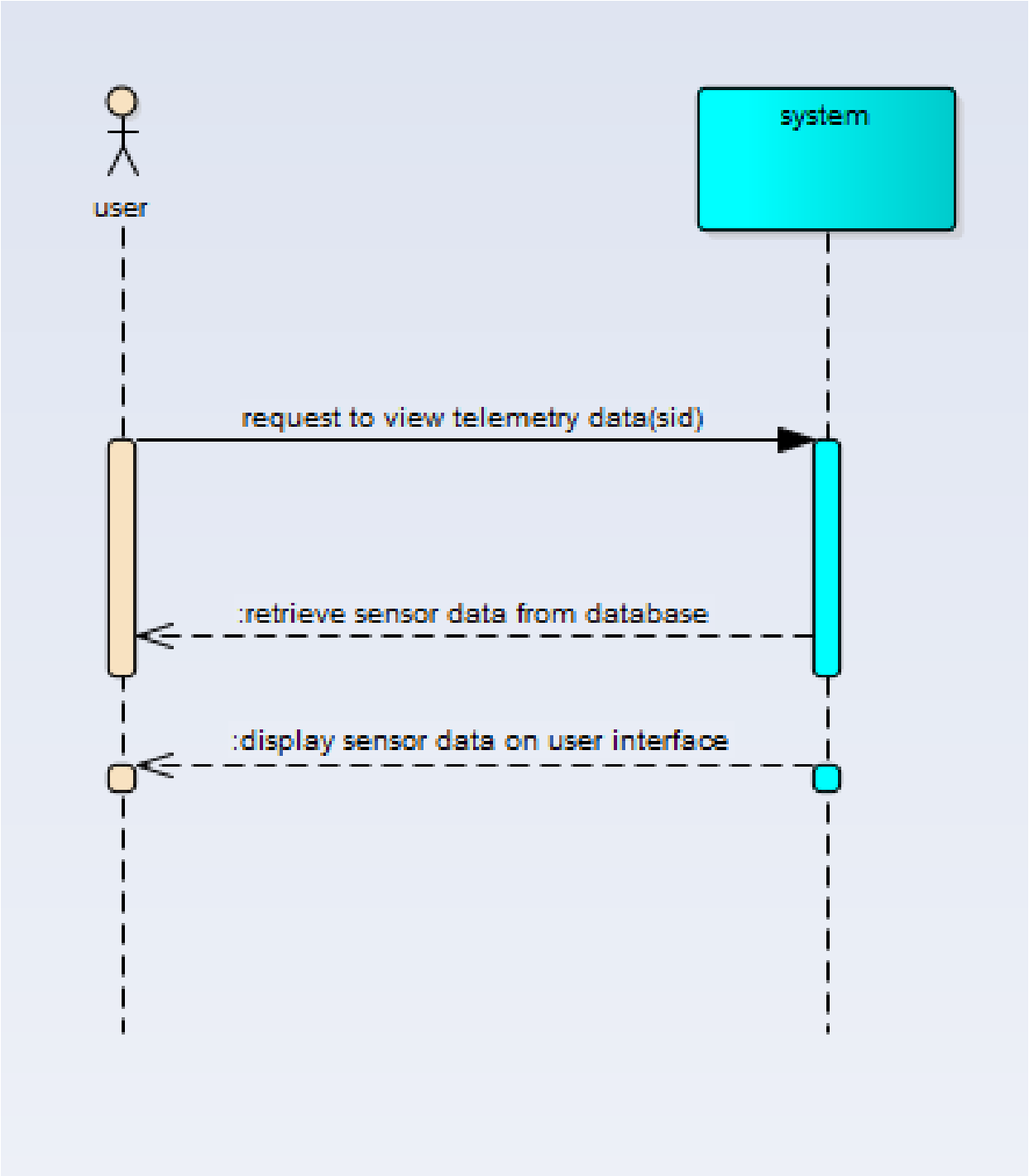

Figure 2. 5 SSD (View telemetry data)

This is define rule schema system sequence diagram in which we request to define rules schema to the system then system display rules schema form to user and then user define all the rules and submit then system responds that you are successfully add rule schema to the system.

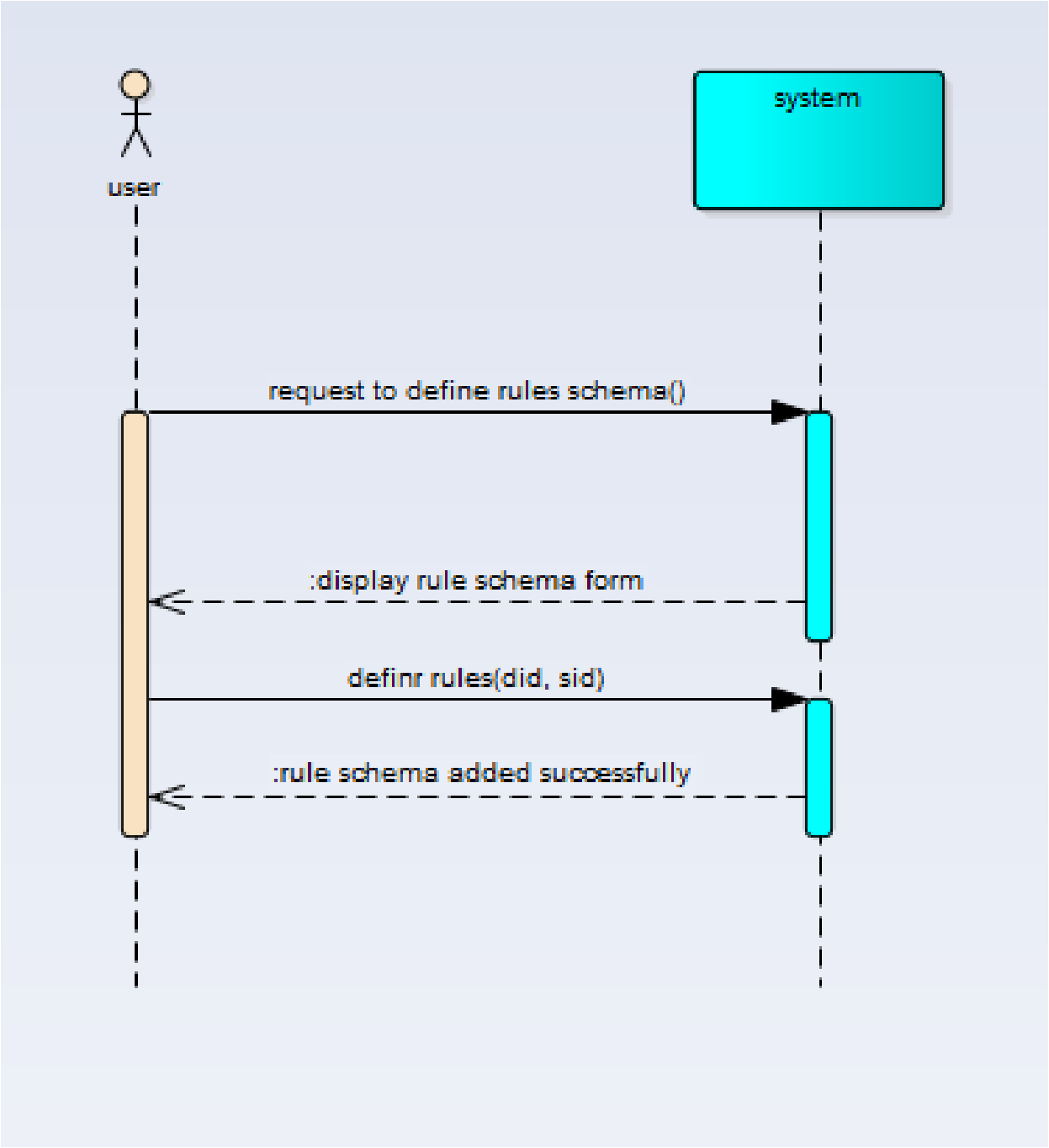

Figure 2. 6 SSD (Define rule schema)

This is add roles and responsibilities system sequence diagram in which we request to view user list then system display list of all users and then user also add roles and submit then system responds that you are successfully add roles in to the system.

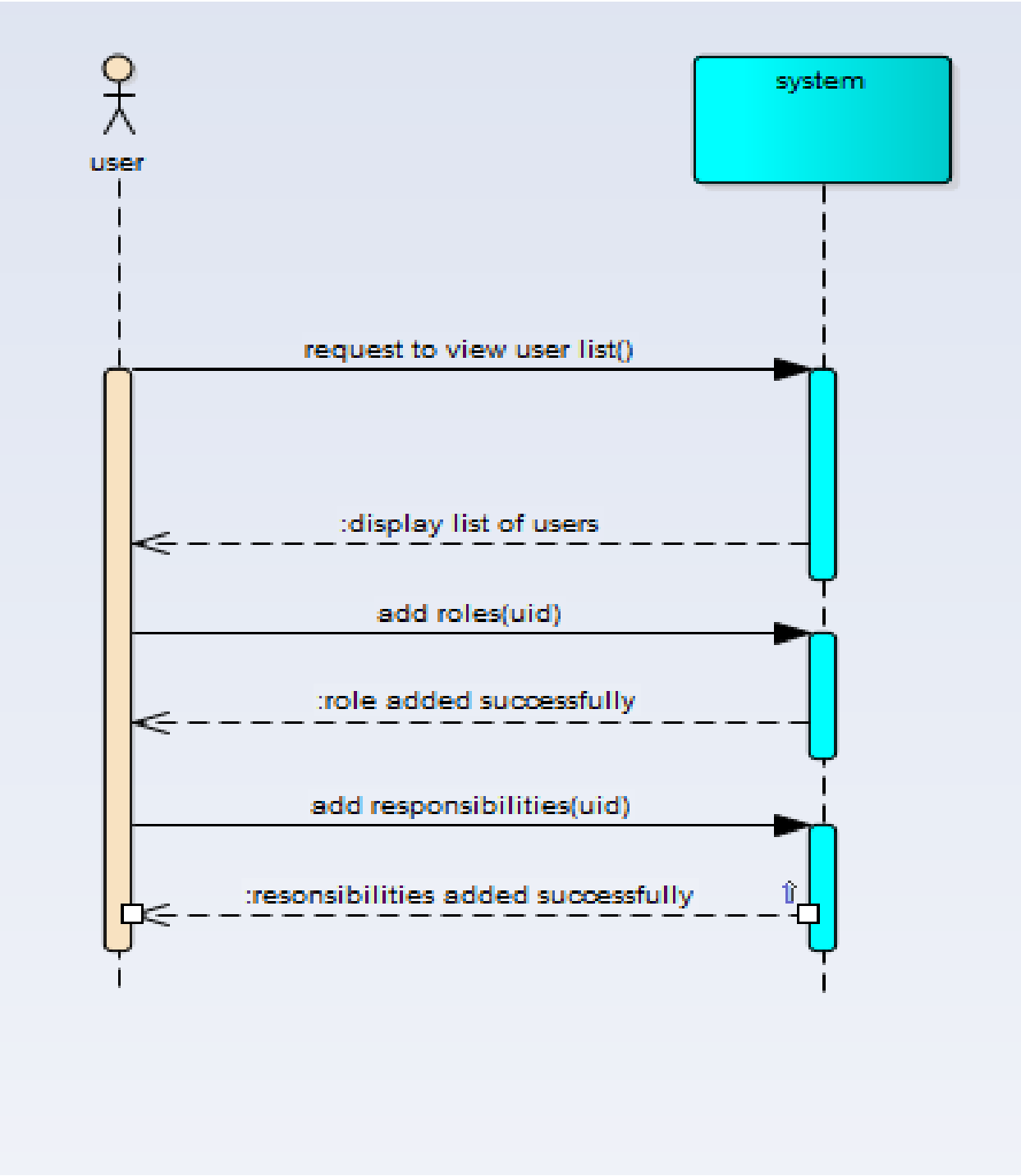

Figure 2. 7 SSD (Add roles and responsibilities)

This is managing authorization system sequence diagram in which we request to view user then system display user profiles and then user update roles and responsibilities and submit then system responds that you are successfully update the role and responsibilities to the system.

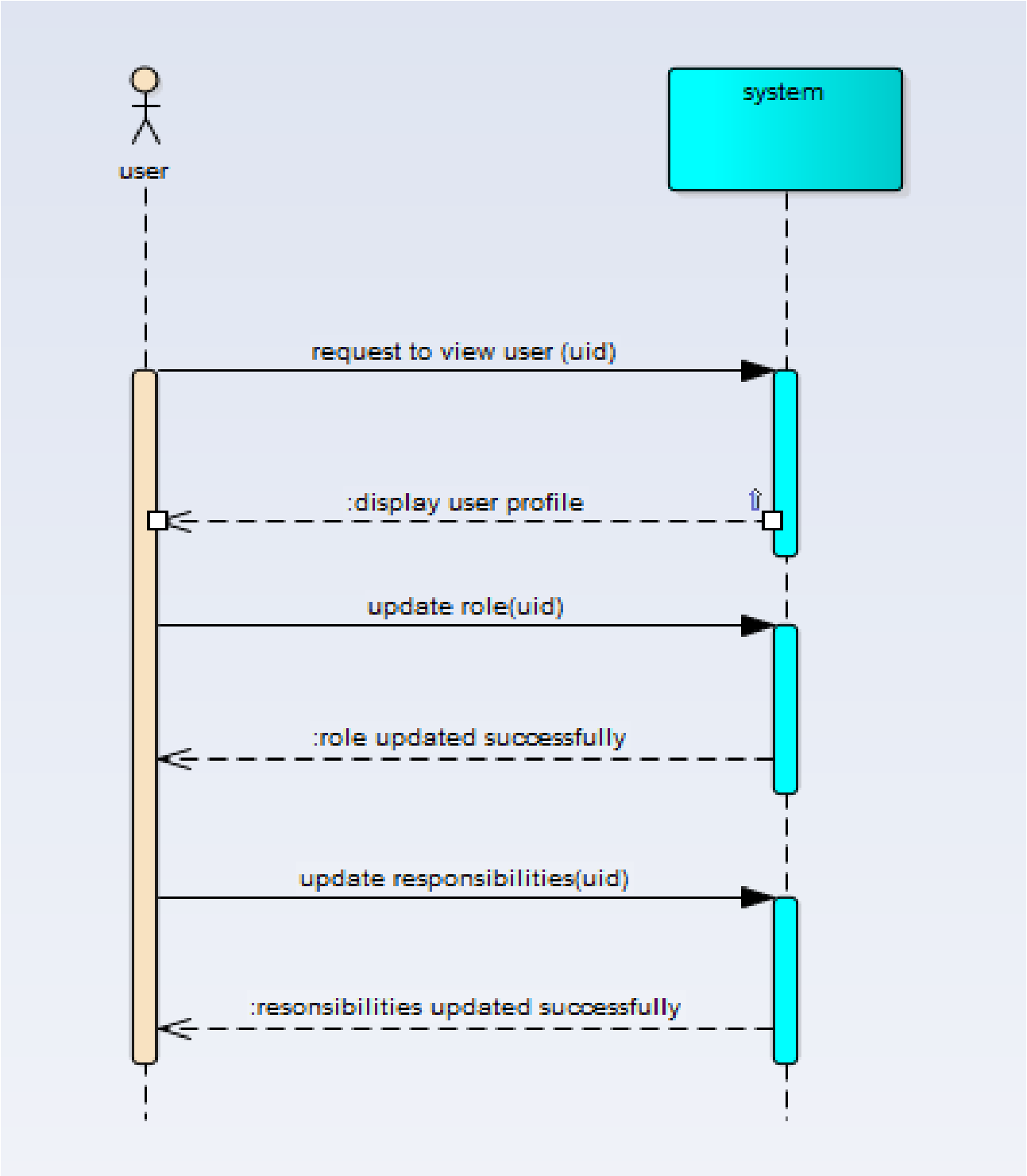

Figure 2. 8 SSD (Manage authorization)

This is Add sensor system sequence diagram in which we request to add new sensor to the system then system display new sensor form to user and then user fill all the credentials and submit then system responds that you are successfully add sensor into the system.

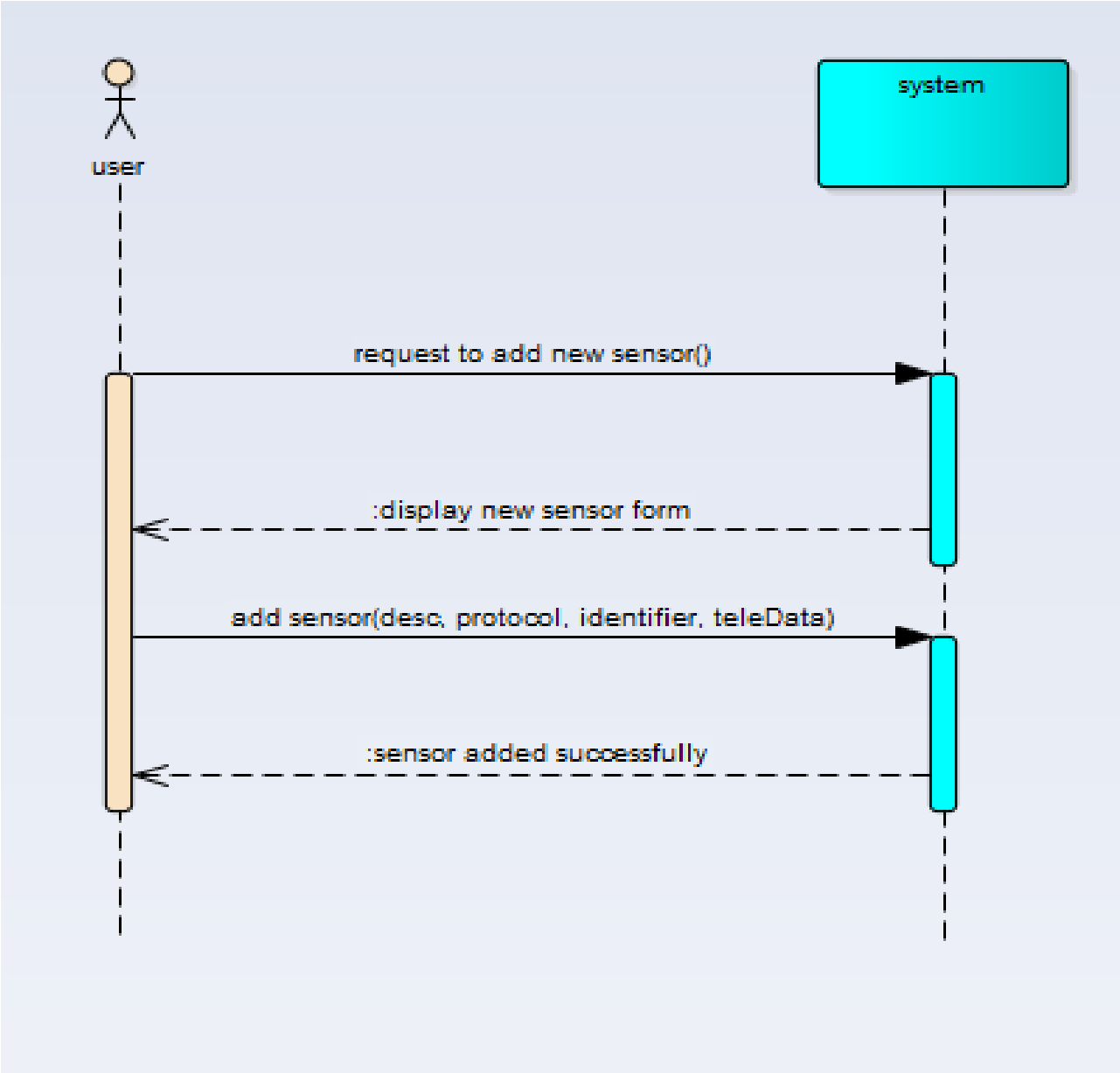

Figure 2. 9 SSD (Add sensor)

This is schedule task system sequence diagram in which we request to view task list to the system then system display list of tasks to user and then user schedule tasks and submit then system responds that you are successfully scheduled the tasks.

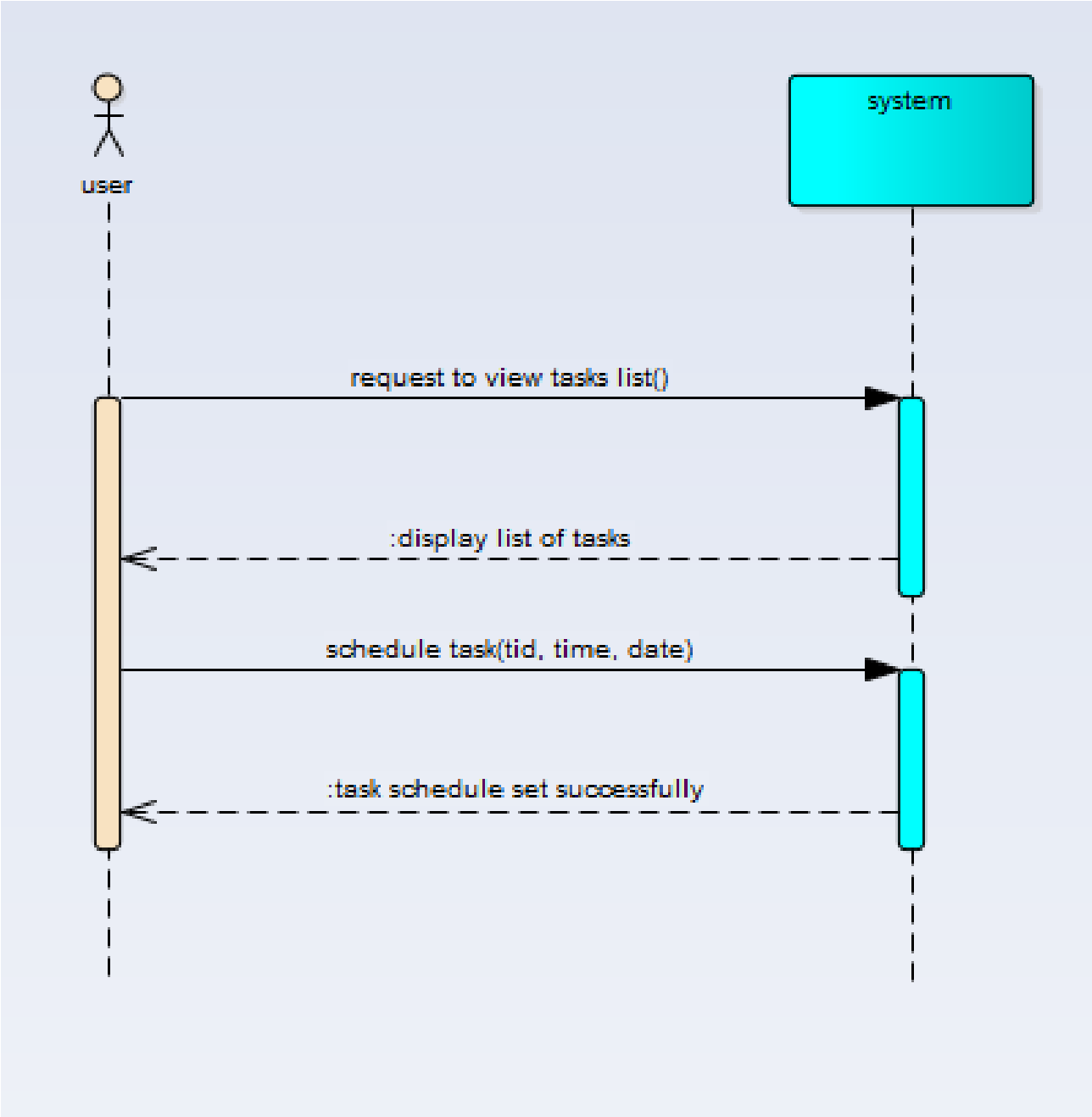

Figure 2. 10 SSD (Schedule task)

This is managing devices system sequence diagram in which we request to view device to the system then system display device information to user and then user can change status of the device and submit then system responds that you are successfully update the status of the system.

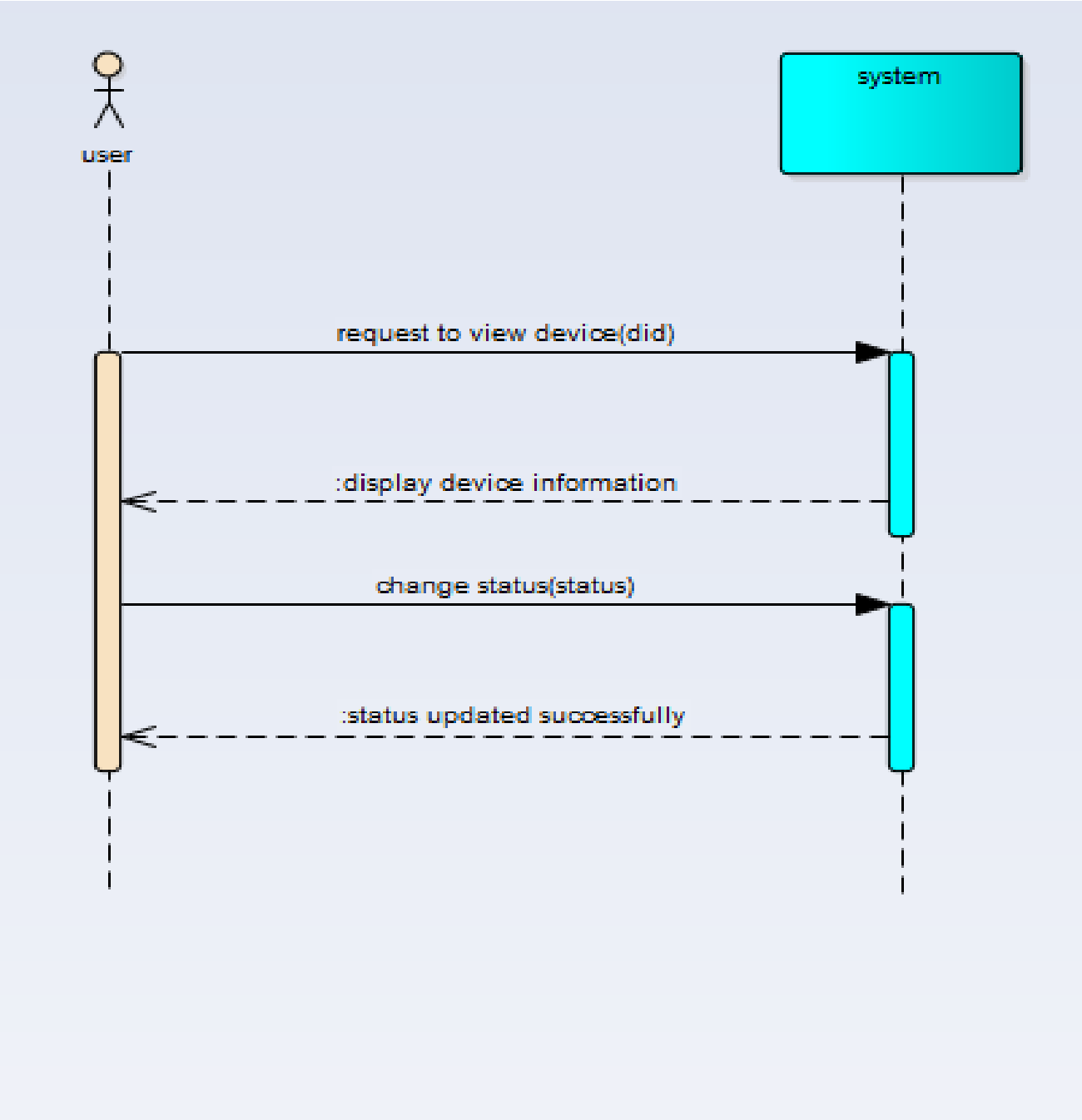

Figure 2. 11 SSD (Manage devices)

This is define area system sequence diagram in which we click on the dashboard then system display defines area form to user and then user add area and submit then system responds that you are successfully define the area into the system.

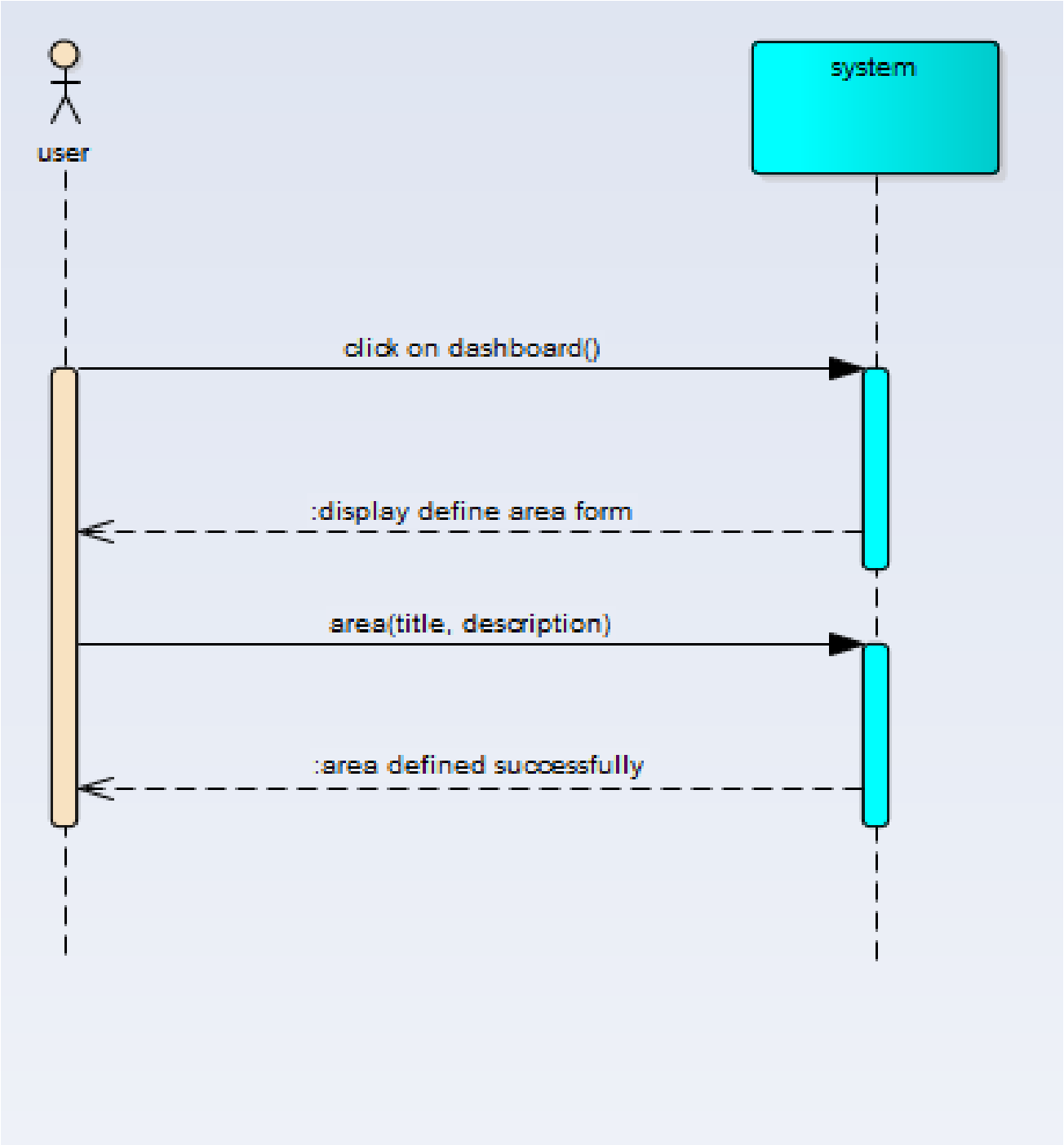

Figure 2. 12 SSD (Define area)

This is Add device system sequence diagram in which we request to add new device in the system then system display new device form to user and then user fill all the credentials and submit then system responds that you are successfully add new device into the system.

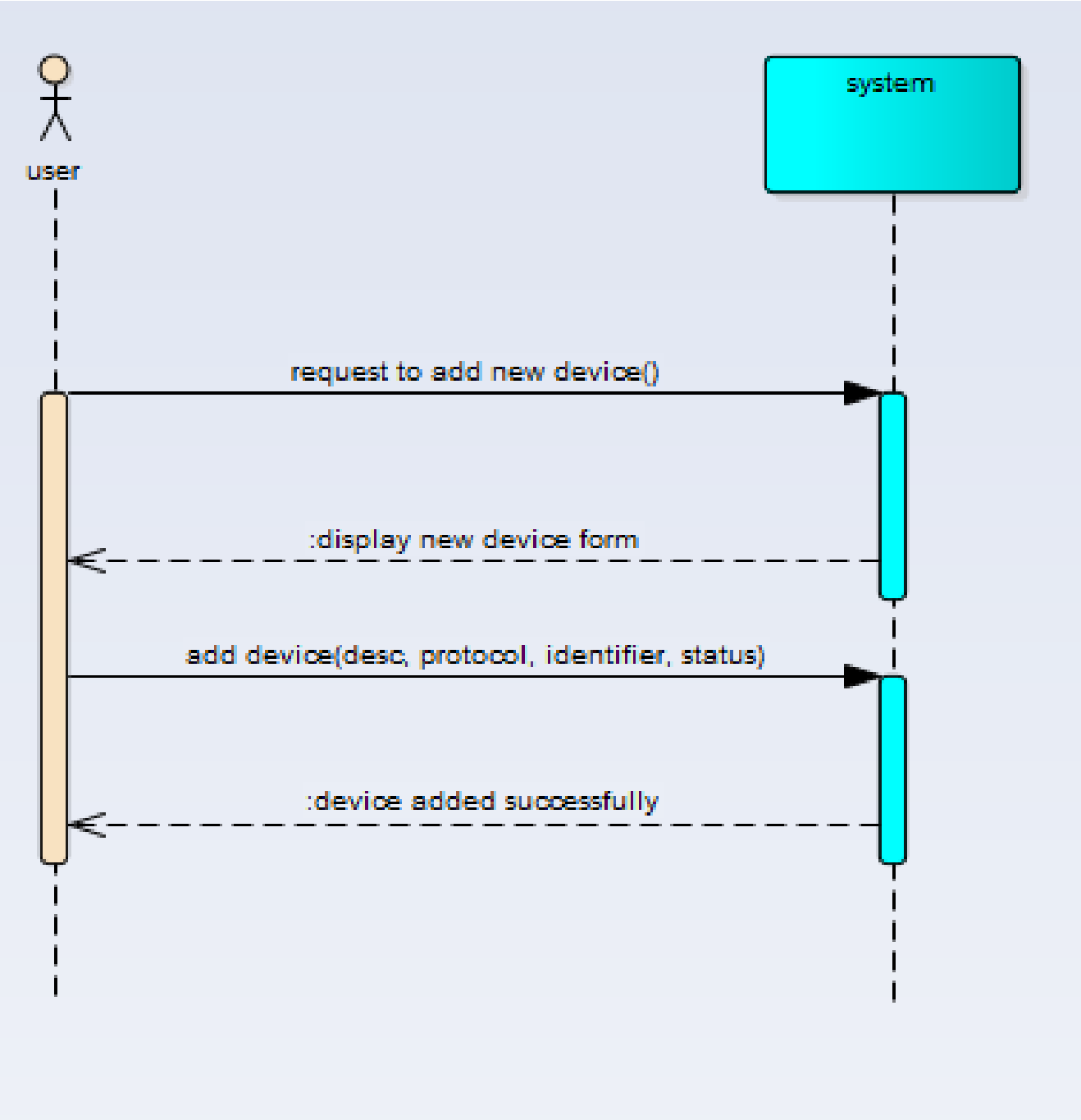

Figure 2. 13 SSD (Add device)

This is subscribing plan system sequence diagram in which we click on subscription then system shows us a form in that form user fill credentials and then user choose plan criteria and then system display payment method form to user and then user fill payment method form and submit then system responds that you are successfully subscribe the plan.

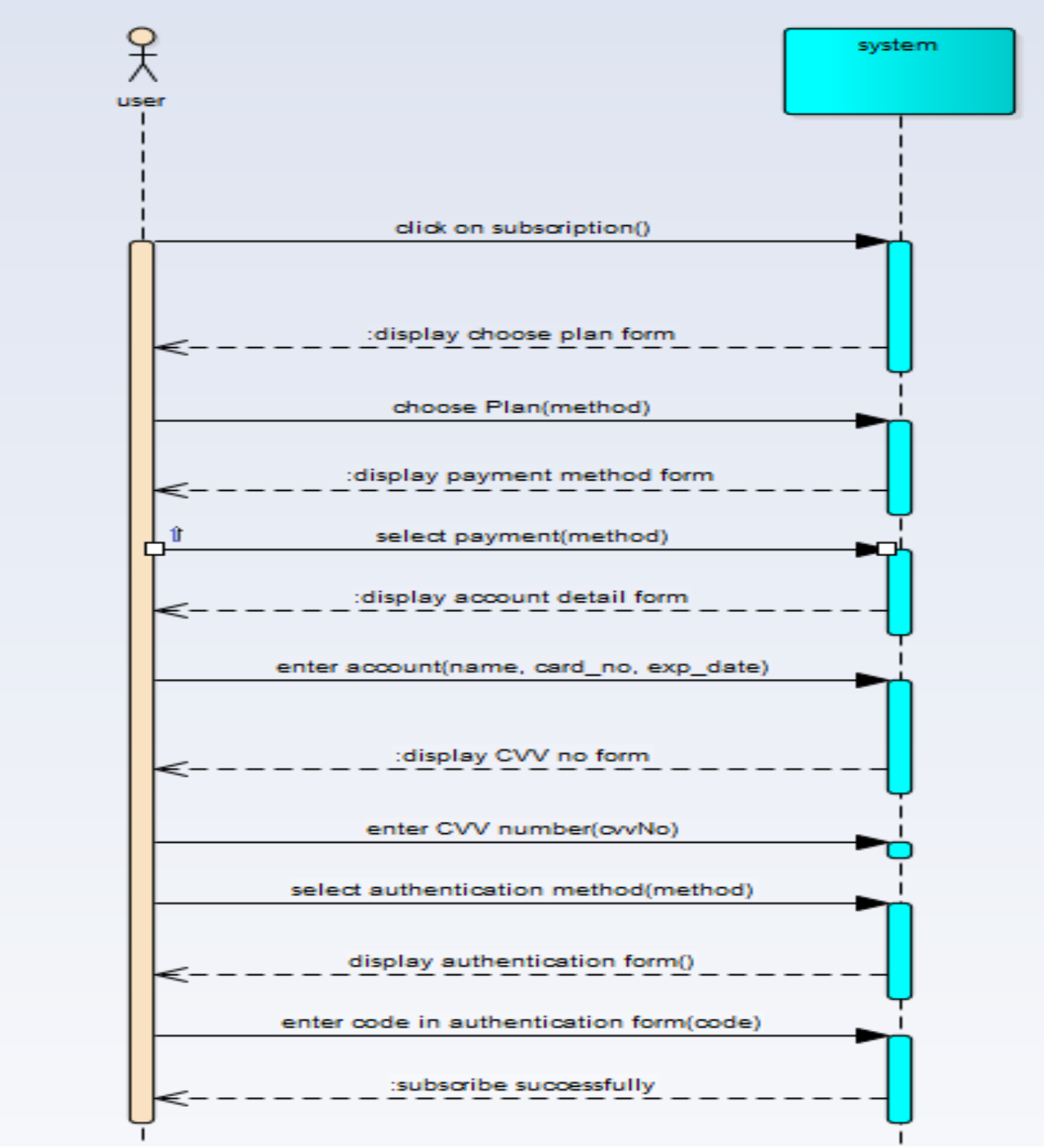

Figure 2. 14 SSD (subscribe Plan)

This is Add Area system sequence diagram in which we click on add area button then system display add Area form to user and then user fill all the credentials and submit then system responds that you are successfully add area into the system.

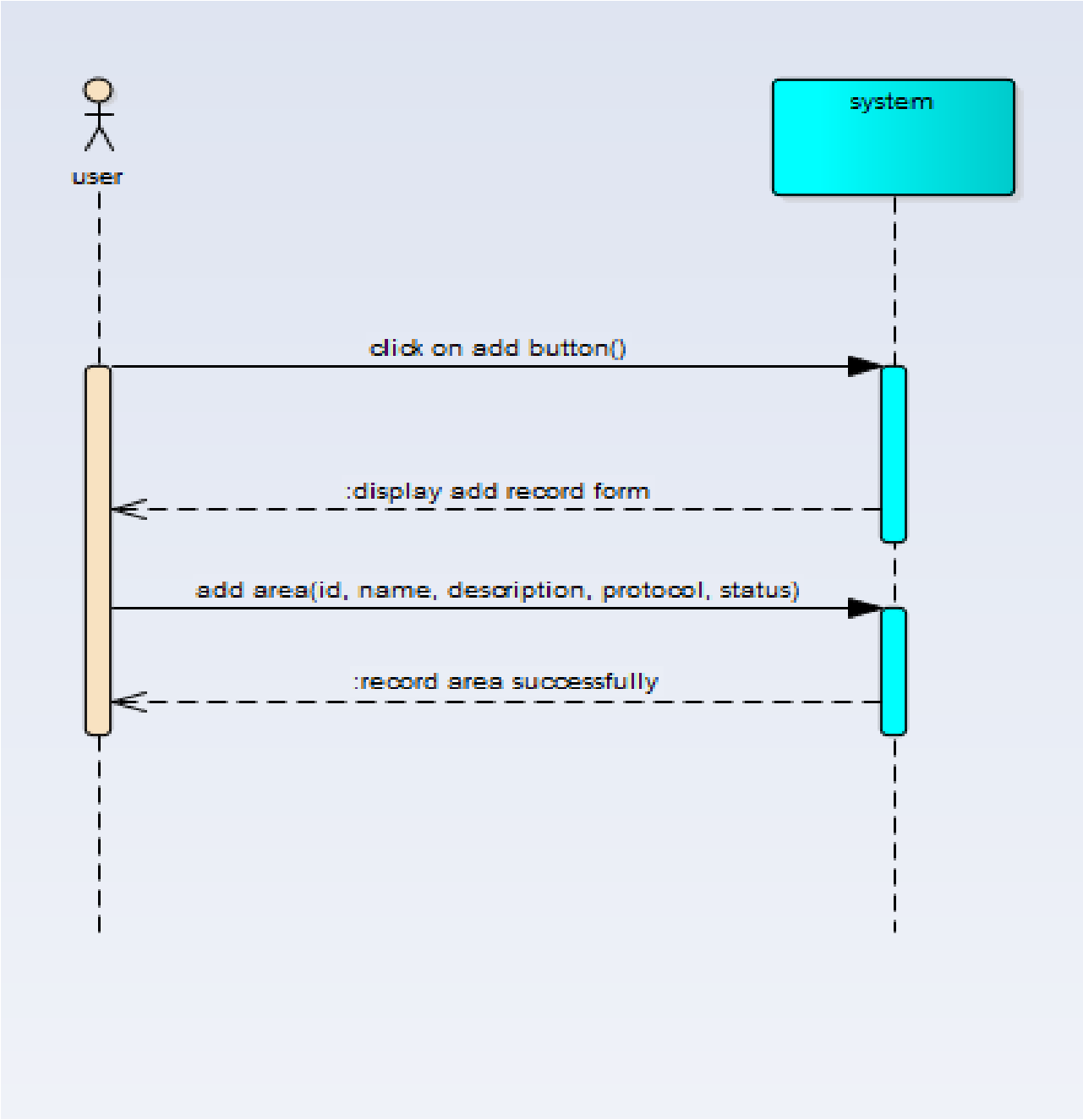

Figure 2. 15 SSD (Add Area)

This is delete Area system sequence diagram in which we click on delete area button then system display delete Area form to user and then user fill all the credentials and submit then system responds that you are successfully delete area into the system.

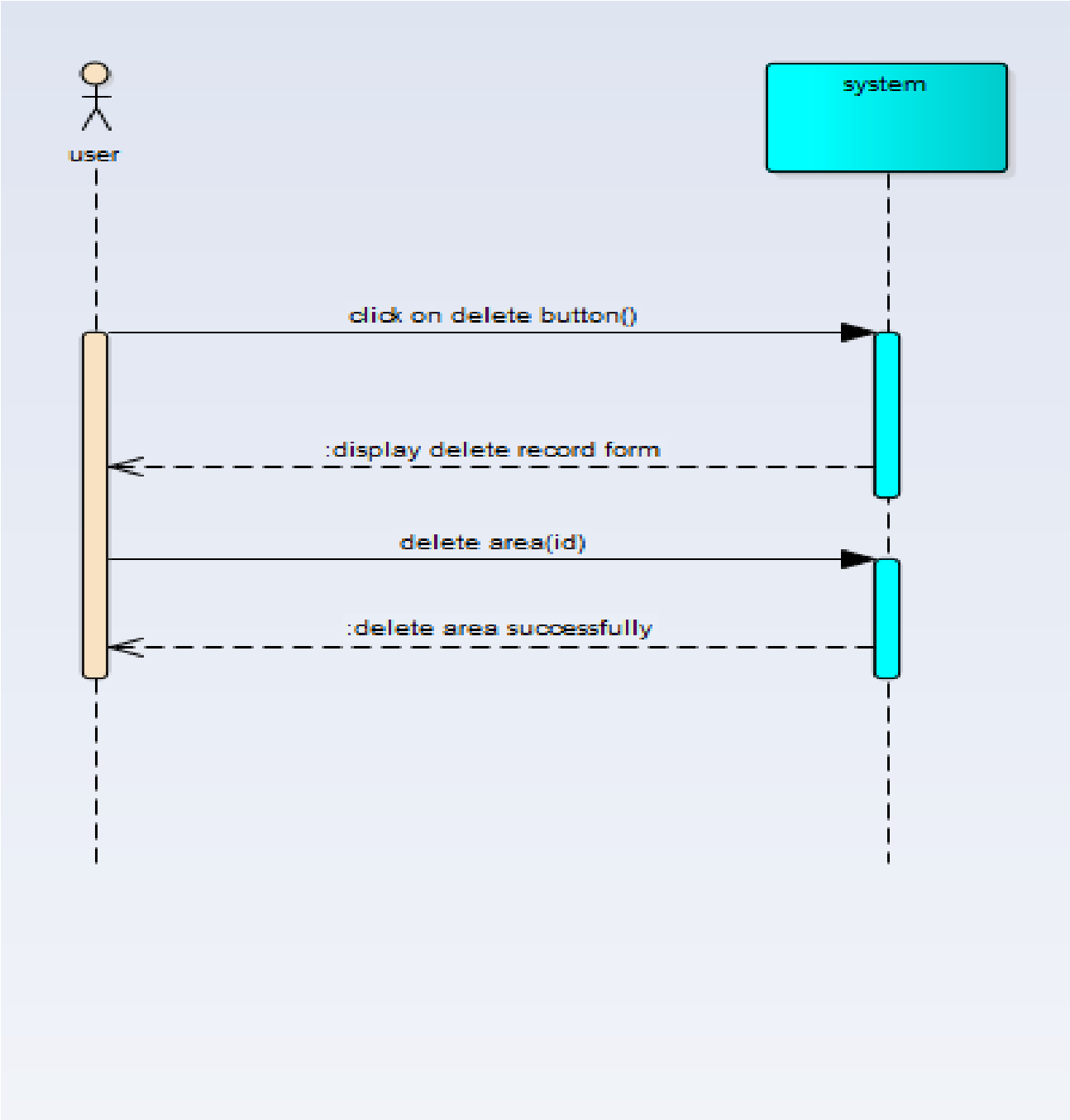

Figure 2. 16 SSD (delete Area)

This is update Area system sequence diagram in which we click on update area button then system display update Area form to user and then user fill all the credentials and submit then system responds that you are successfully update area.

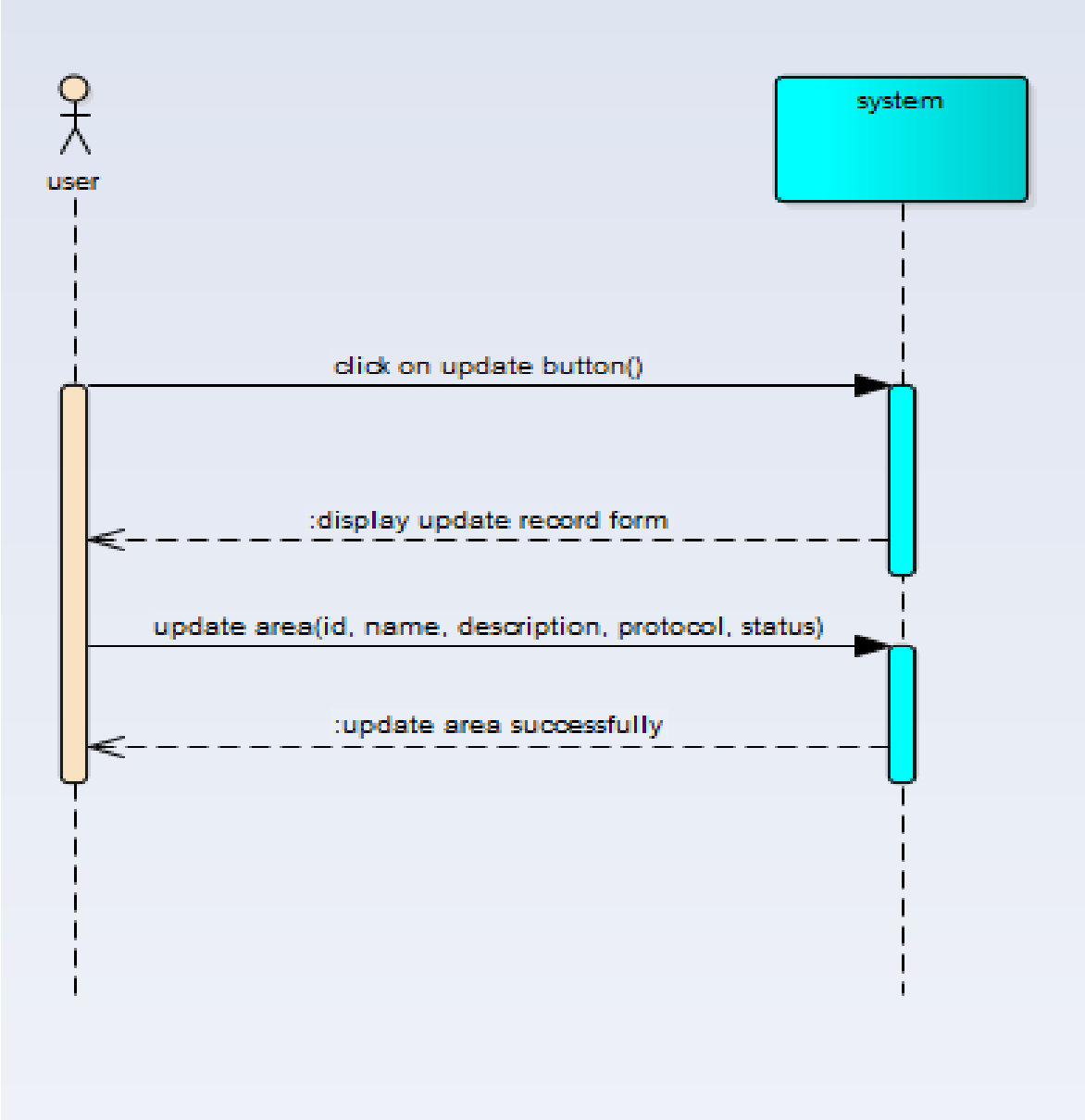

Figure 2. 17 SSD (update area)

This is view Area system sequence diagram in which we click on area button then system display list of Areas and crud operations options to user then system responds that you are successfully viewed area list of the system.

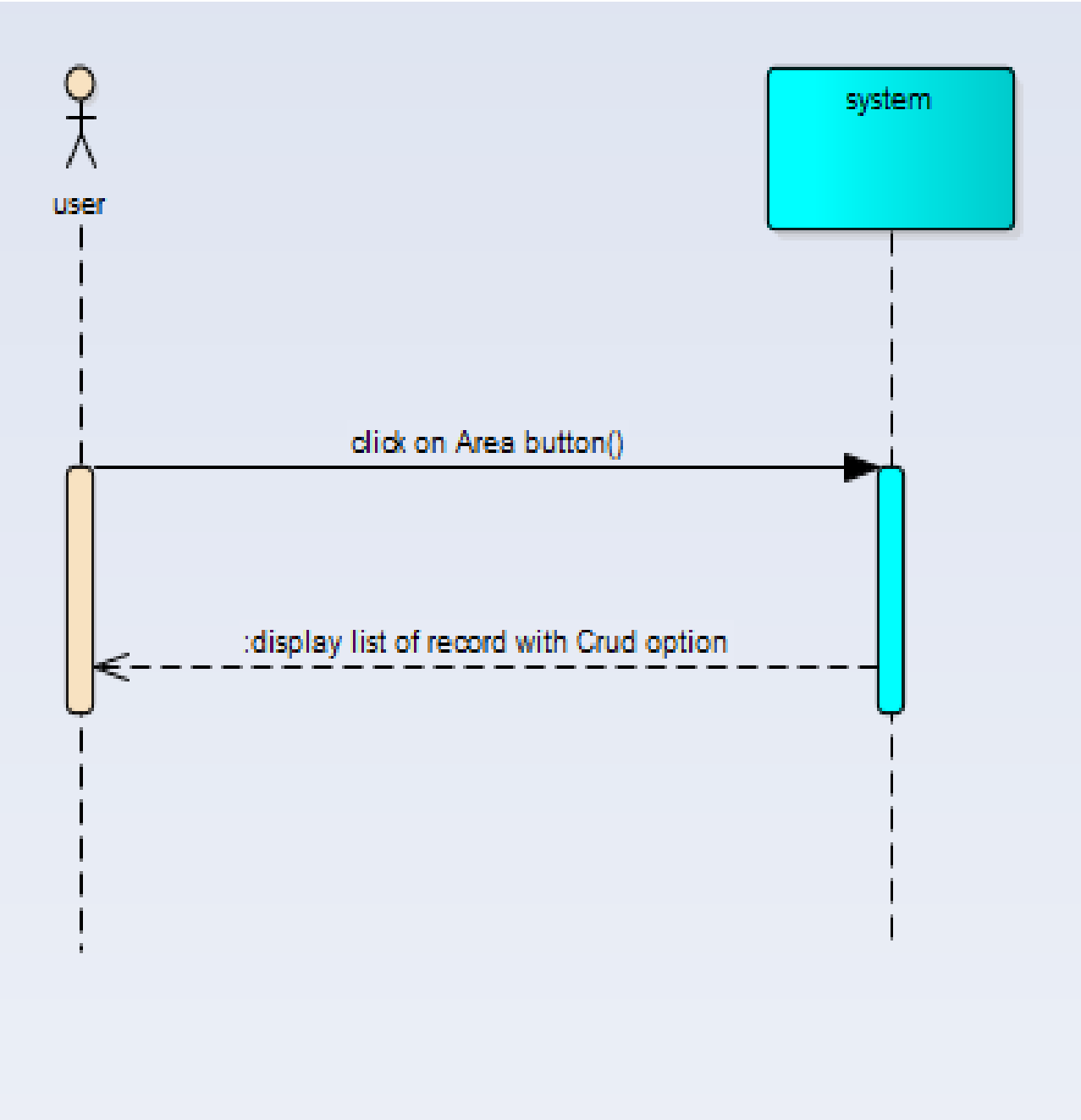

Figure 2. 18 SSD (view Area)

2.7. Domain Model

The basic concepts of the domain are Area, User, Status, claim, Device, Sensors and Telemetry Data, Role, Plan, Card tables

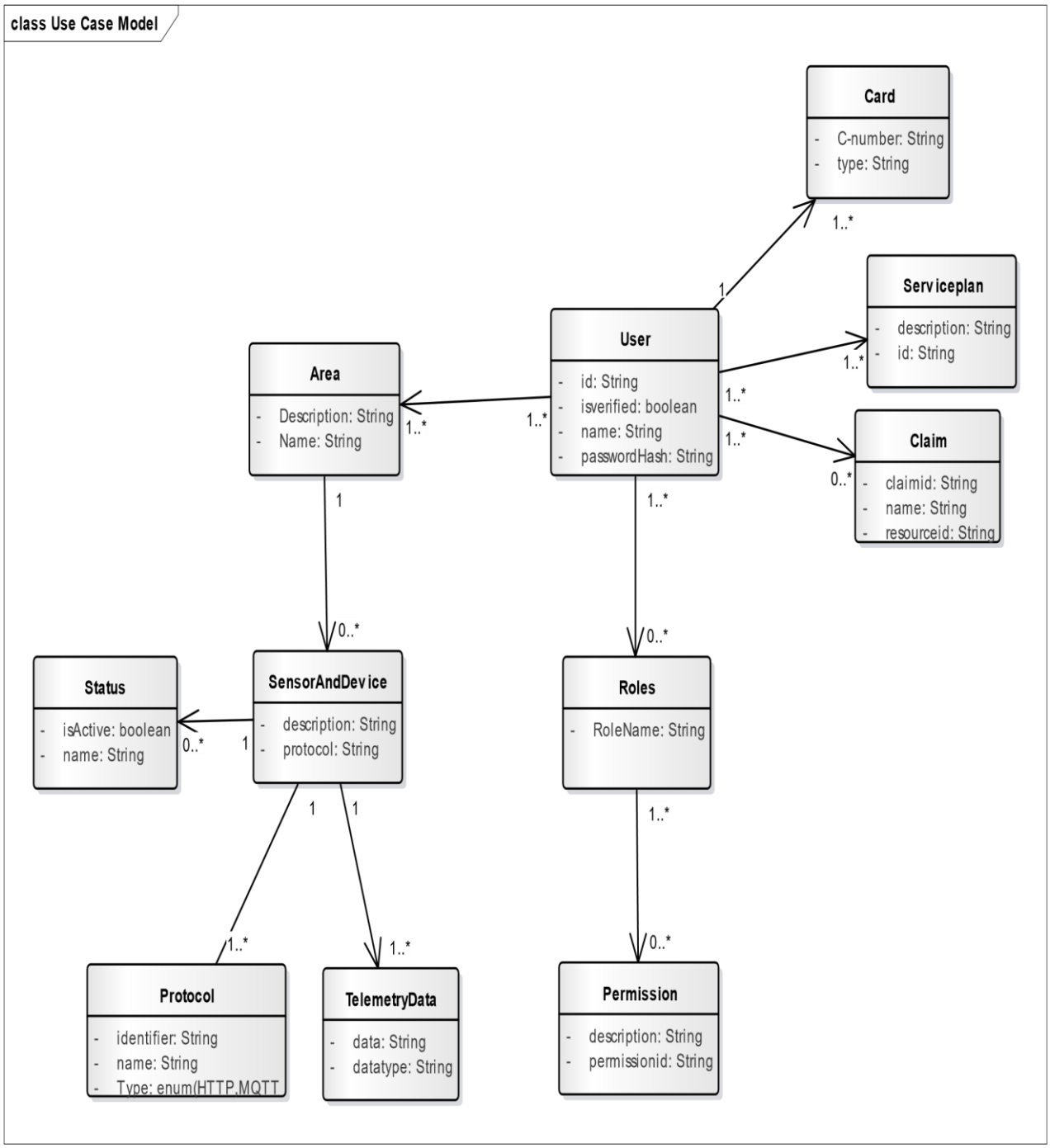

Figure 2. 19 Domain Model

Chapter 3
System Design

The purpose of this chapter is to provide necessity information needed for the development phase. Without an sufficient design, that delivers required function and quality attributes, the project will fail. Nonetheless, communicating architecture to its stakeholders is as important as creating it in the first place.

3.1. Layer Definition

Table 3. 1 Layers Definition

Layers	Description
Presentation Layer	This layer used for the interaction between the user and the system through a graphical user interface.
Business Logic Layer	This layer contains the business logic of the intended system to be built that includes all the constraints and functions.
Database Layer	This layer contains the database of the intended application to be built.

3.1.1. Presentation Layer

This layer occupies the top level and displays information related to services available on a mobile application. This tier communicates with other tiers by sending results to the server and other tiers in the network.

3.1.2. Business Logic Layer:

Application Layer also called the middle tier, logic tier, business logic or logic tier, this tier is pulled from the presentation tier. It controls application functionality by performing detailed processing.

3.1.3. Database Layer:

Database layer includes database servers where information is stored and retrieved. Data in this tier is kept independent of application servers or business logic.

3.2. Software Architecture

Software architecture is described as the organization or structure of a system, where the system represents a collection of components that accomplish a specific function or set of functions. We have mobile client and AquaPonic system (Sensors, fan, fishes, devices) they linked with Web server where we have rules defined for our system and also have our trained model for image processing who predict that plant has disease or not and then we have database where all data are stored. Below is the architecture diagram of the system.

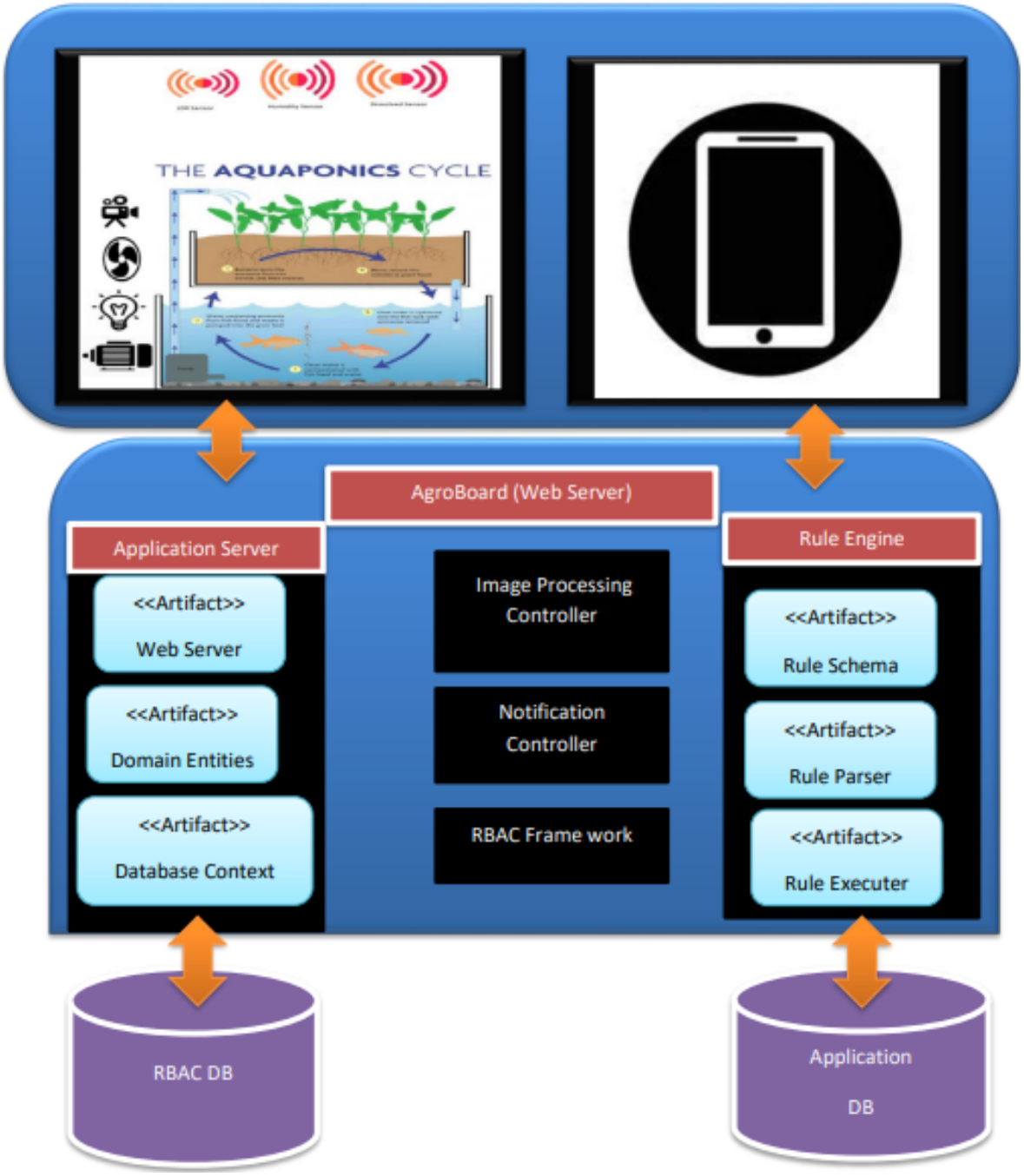

Figure 3. 1 Architecture view

Image processing flow chart:

This is image processing architecture in which firstly we take input image from camera or take recent pictures from database then we resize the image because all the images don't have same dimensions then we normalize this image. In normalization we change its pixel size and its intensity and we set range between 0 to 255.then we perform pre-processing, in pre-processing we remove noise from the image, and also we perform filtering and segmentation then we take this image to our model then they predict that this image has disease or not.

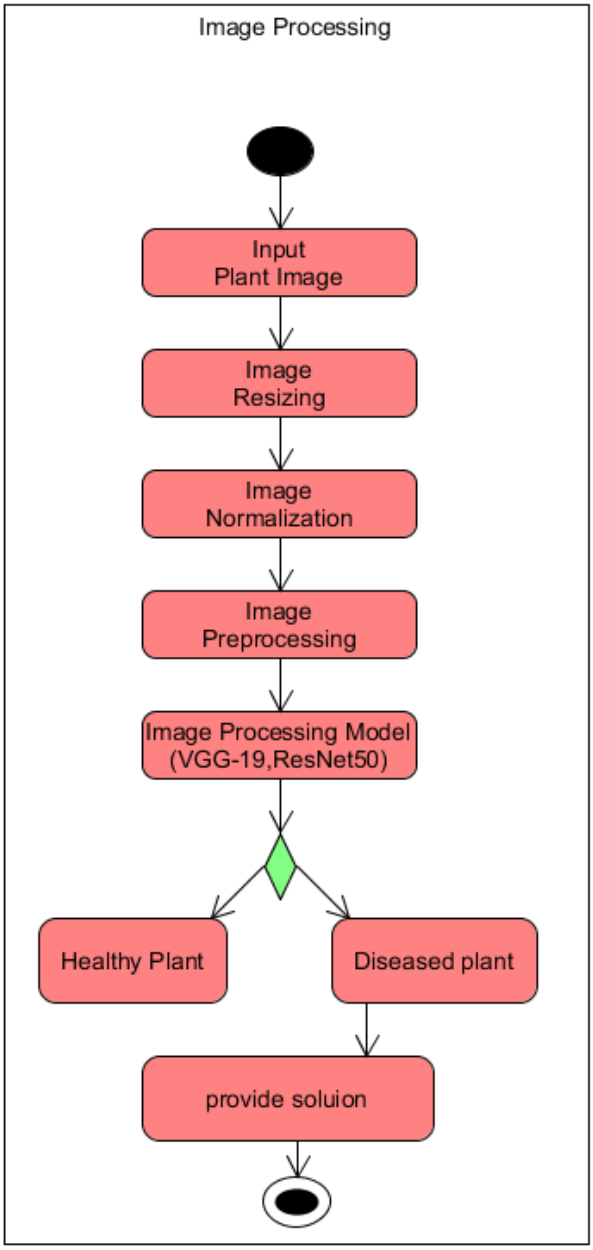

Figure 3. 2 Image processing activity diagram

Input Image:

In image processing firstly we take input image from camera and from recent images. Then we perform resizing. **Image Resizing:**

Image resizing is necessary because we need to increase or decrease the total number of pixels, whereas remapping can occur when you are correcting for lens distortion or rotating an image.

Image Normalization:

In image processing, normalization is a process that changes the range of pixel intensity values. Its normal purpose is to convert an input image into a range of pixel values that are more familiar or normal to the senses (0 to 255).

Figure 3. 3 Image normalization

Preprocessing:

Pre-processing is basically operations name with images at the lowest level of abstraction both input and output are intensity images. These plant images are of the same kind as the original data captured by the sensor, with an intensity image usually represented by a matrix of image function values (brightness's). The aim of pre-processing is an improvement of the image data that suppresses unwilling distortions or enhances some image features important for further processing, although geometric transformations of images (e.g. rotation, scaling, translation) are classified.

Image Processing models (VGG-19 & ResNet50):

VGG-19 is a convolutional neural network that is trained on more than a million images from the database. The network is 19 layers deep and can classify images into 1000 object categories, such as a rose, fruit. As a result, the network has learned and predict the value.

ResNet-50 is a convolutional neural network that is 50 layers deep. You can load a pretrained version of the network trained on more than a million images from the database. The pre-trained network can classify images into 1000 object categories, such as rose, fruit.

Output Image or Label:

After applying machine learning models that model predicts the label or image who have disease or not.

Figure 3. 4 Output image

3.3. Class Diagram

The class diagram describes the attributes and operations of a class and the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages:

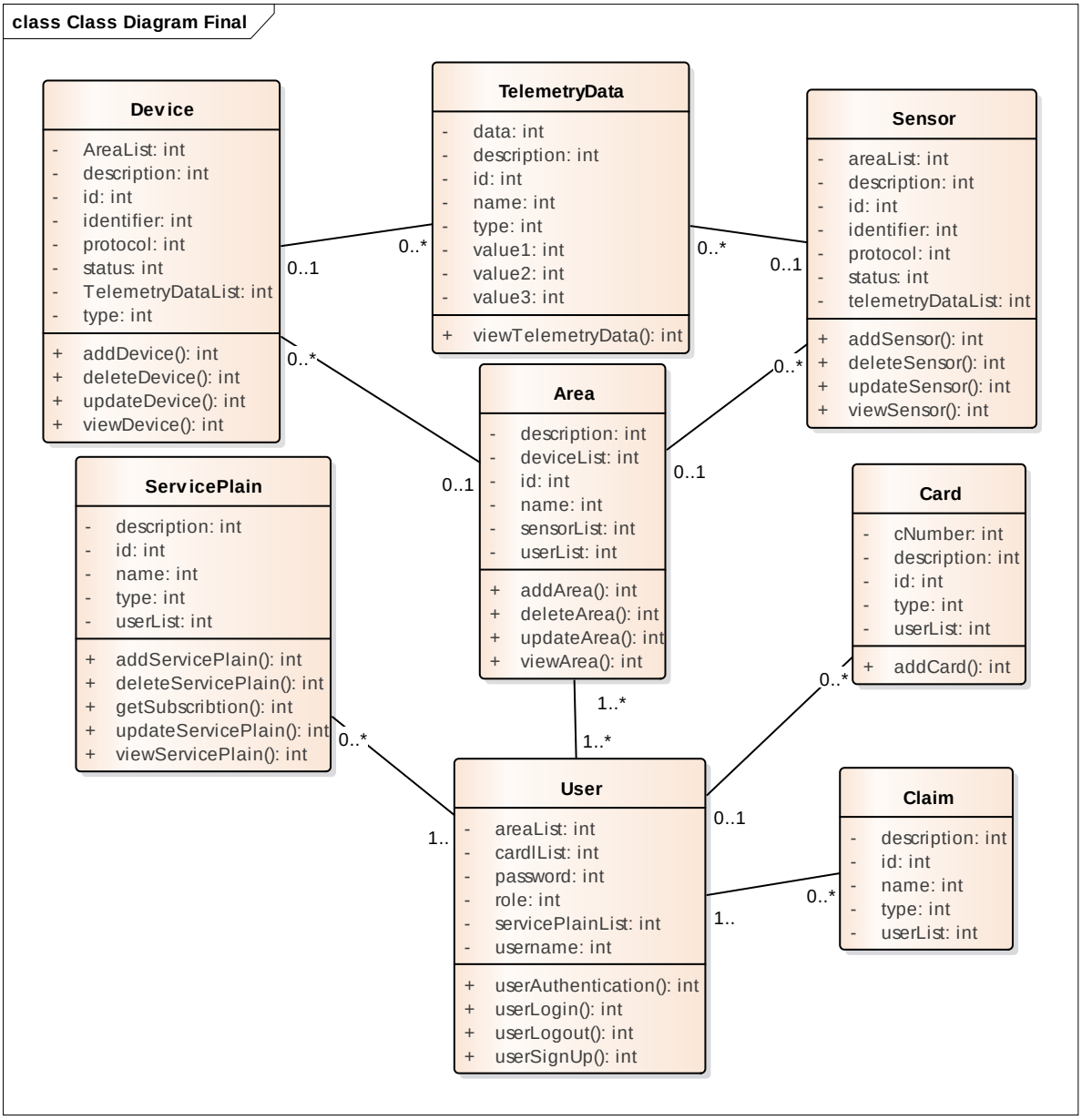

Figure 3. 5 Class Diagram

3.4. Sequence Diagram

Sequence Diagram model the flow of logic within your system in a visual manner enabling you both to document and validate your logic, and are commonly used for both analysis and design purposes.

In this case user click add area button on home page after that recycle View create get record from Context DB class

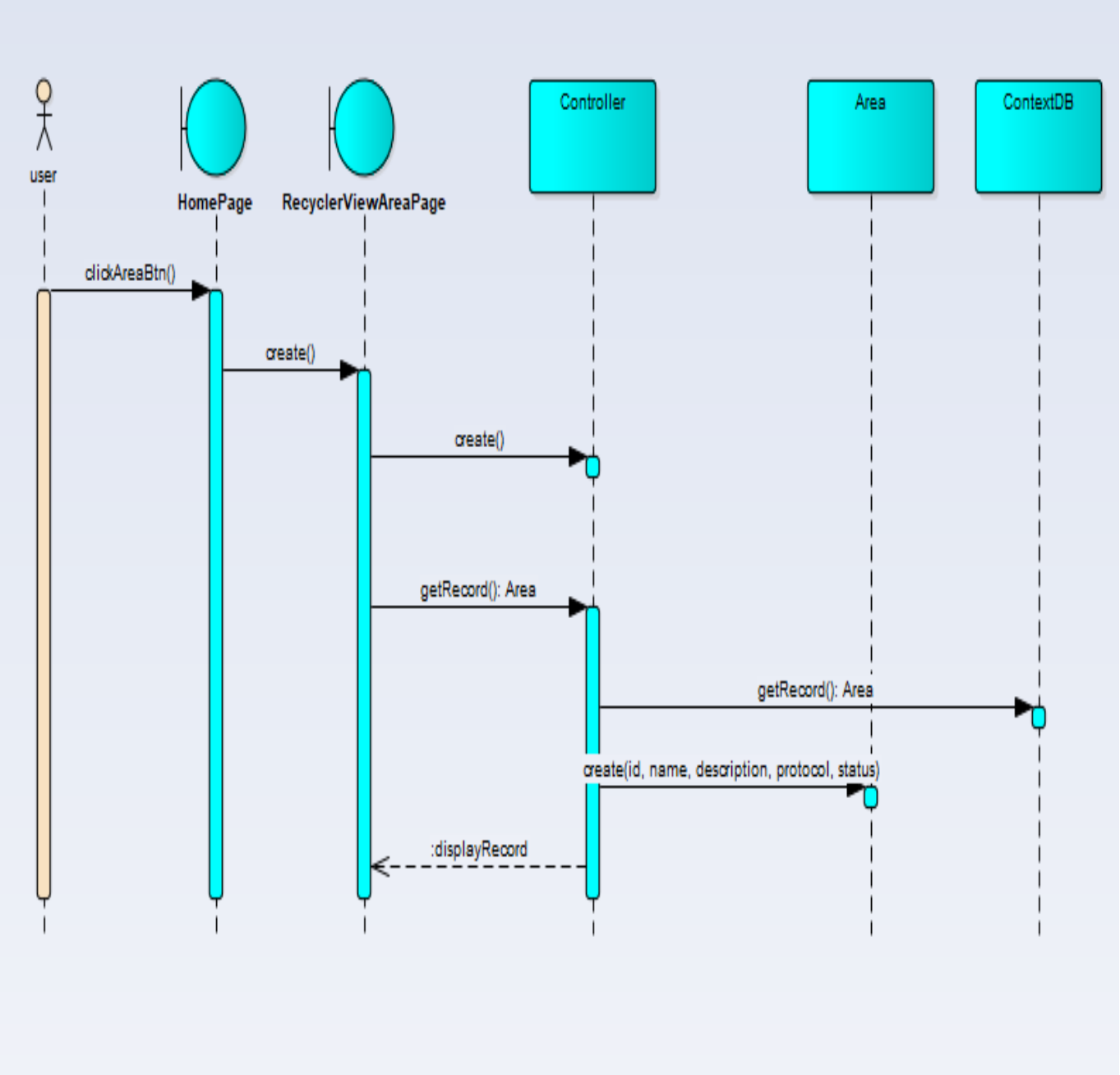

Figure 3. 6 View Area (SD)

In this case user click on update area button then system will display update for to user. In which user enter information and click on update record button then system save all data on Context DB

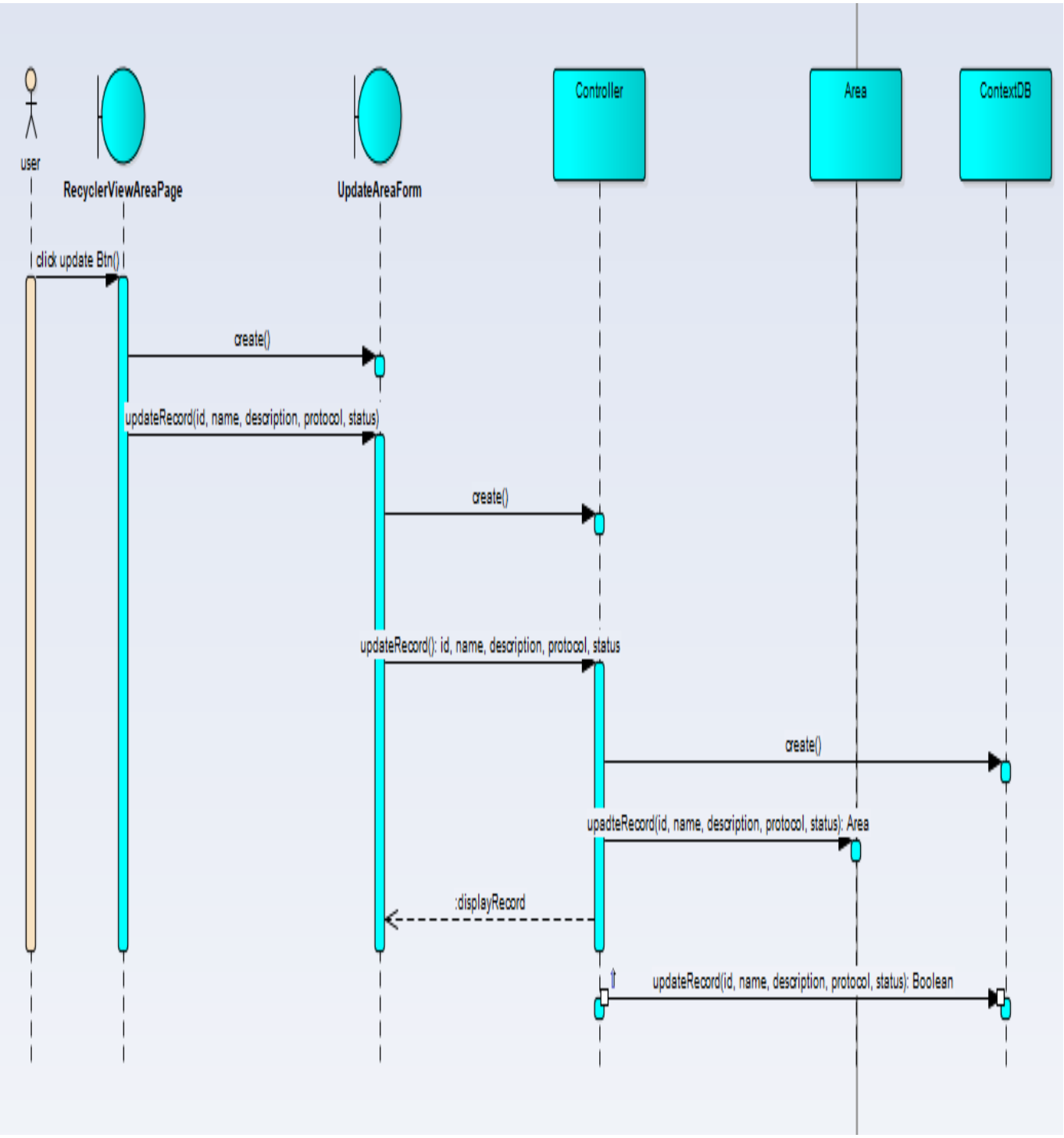

Figure 3. 7 Update Area (SD)

In this case user click on update area button then system display deletes area form to user. In which user enter id and click on delete record button then system will delete record from system and then system show that data will be deleted from Context DB.

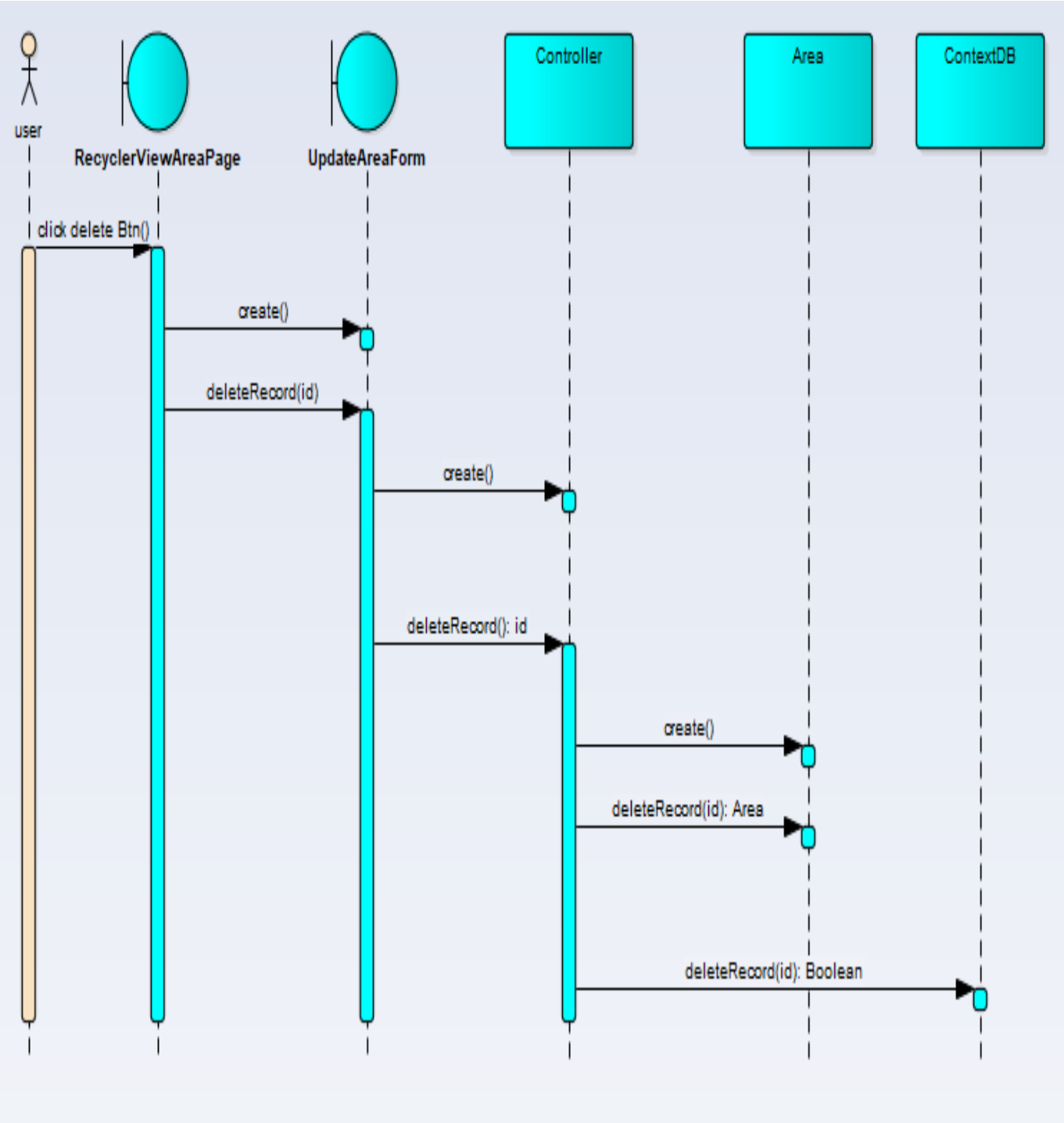

Figure 3. 8 Delete Area (SD)

In this case user click on add area button then system will display add area form in which user enter information and click on add record button then system save all data on Context DB.

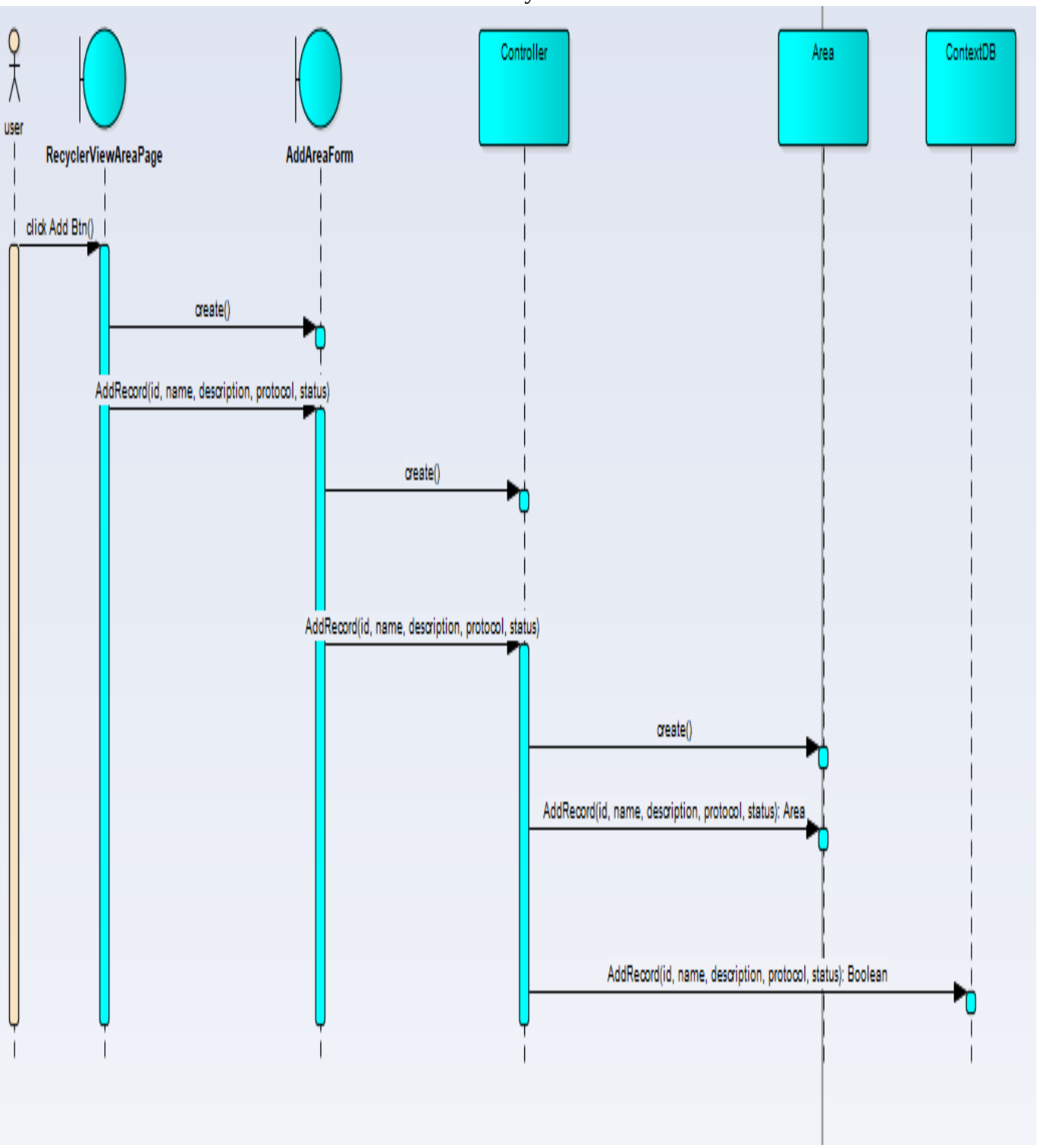

Figure 3. 9 Add Area (SD)

In this scenario user click on subscription button and then system display subscription form in which user select plan and click on process then user select payment method and enter account detail and then enter cvv number after verify code user is successfully verified by system.

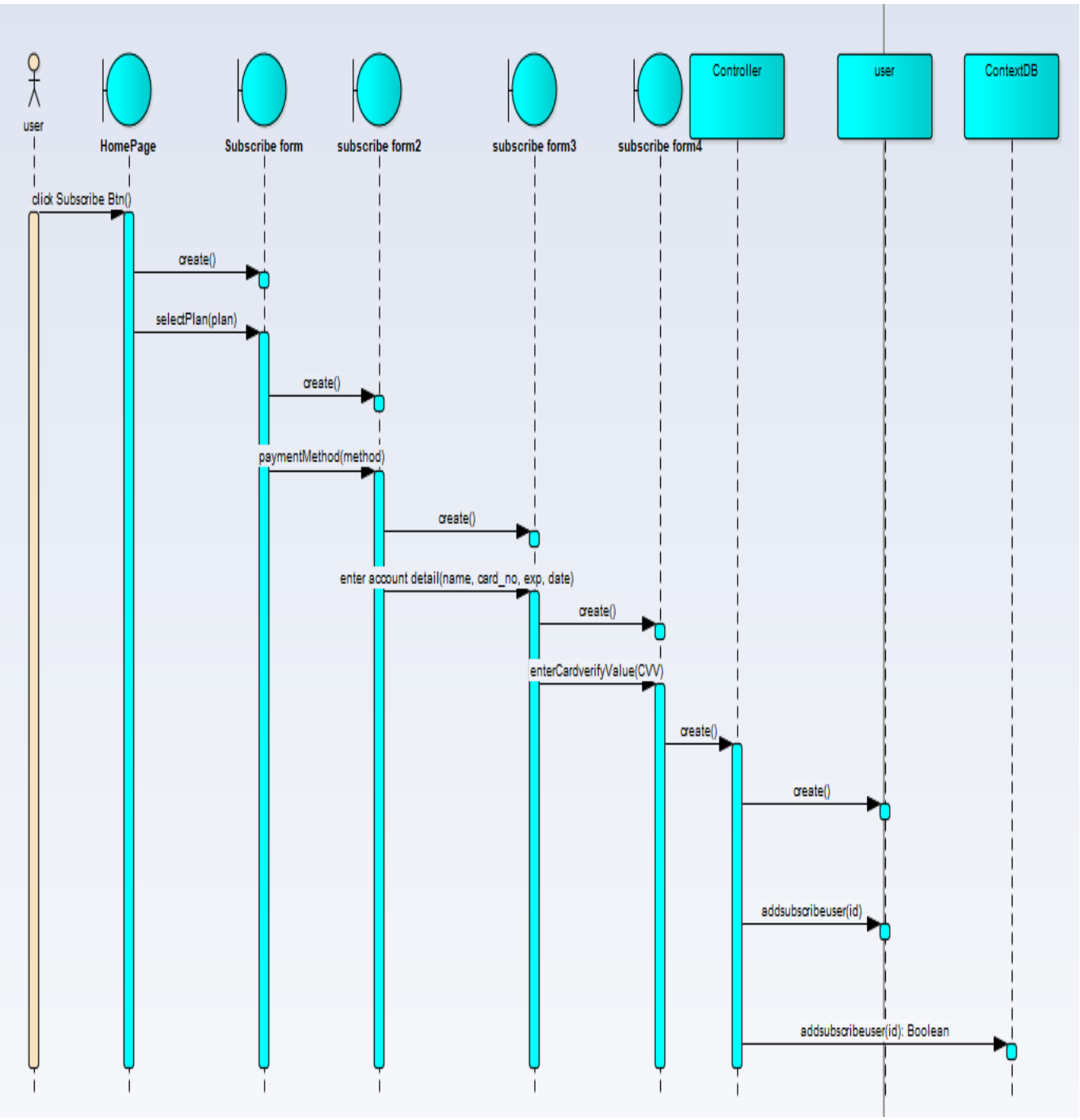

Figure 3. 10 Subscribe plan (SD)

In this case user click on add Client button then system will display add form in which user enter information and click on add record then system save all data on Context DB.

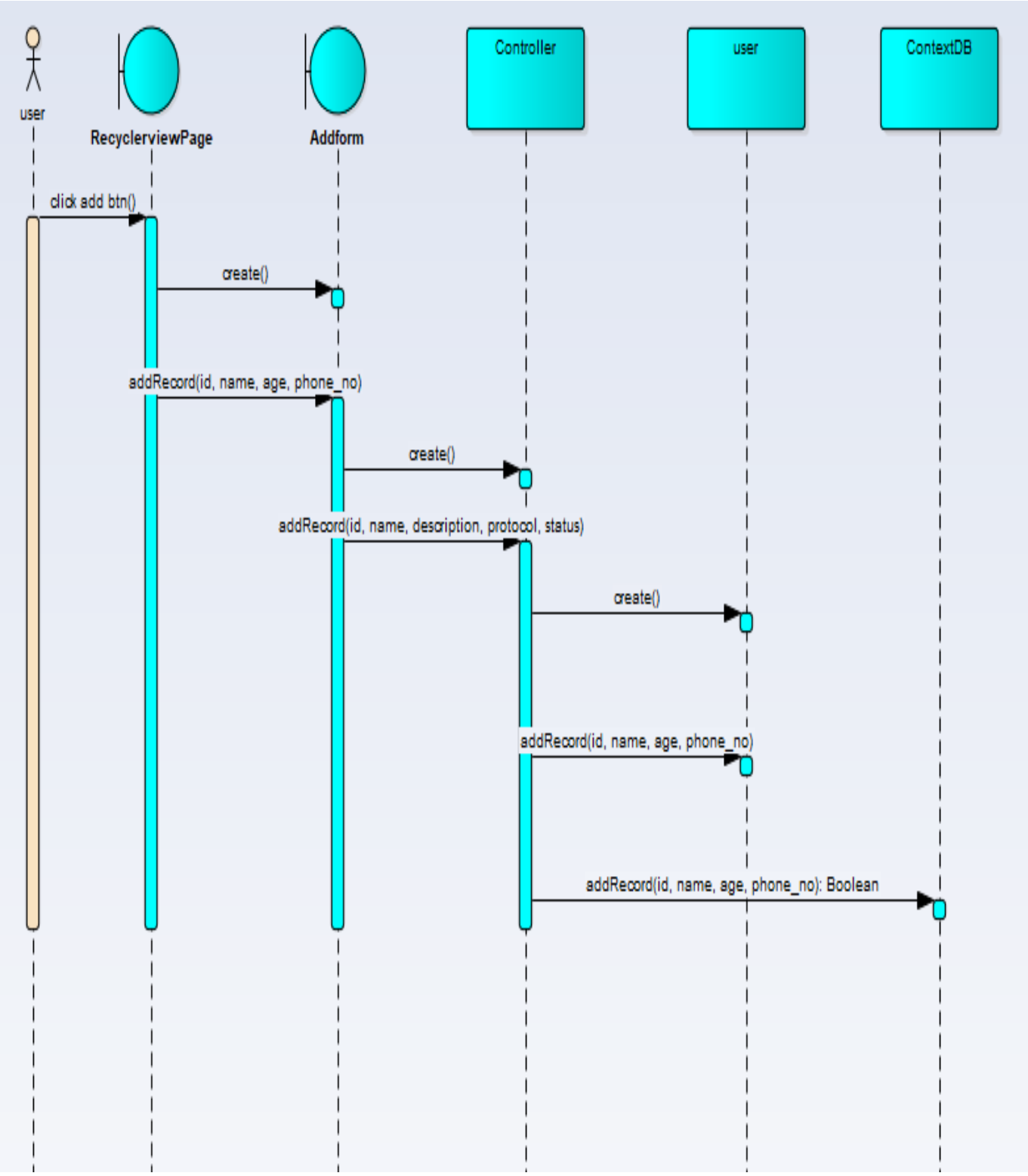

Figure 3. 11 Add Client (SD)

3.5. Entity Relationship Diagram

The entity-relationship model (or ER model) is a way of graphically representing the logical relationships of entities (or objects) in order to create a database. An entity– relationship model (ER model) describes inter-related things of interest in a specific domain of knowledge. An ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between instances of those entity types.

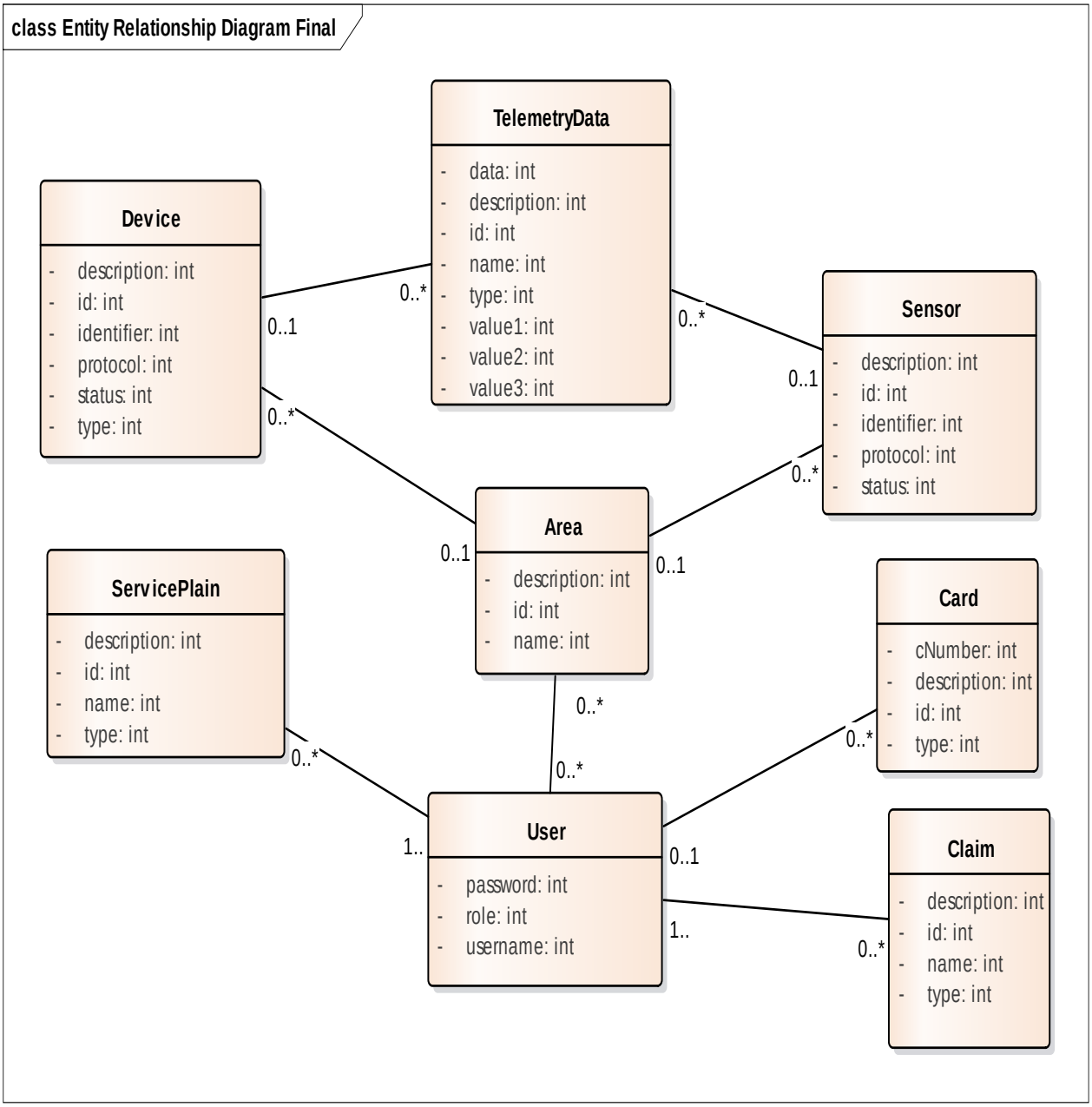

Figure 3. 12 ERD Diagram

3.6. Database Schema

A database schema represents the logical configuration of all or part of a relational database. It can exist both as a visual representation and as a set of rules known as integrity constraints that govern a database. These rules are expressed in a data definition language, such as SQL.

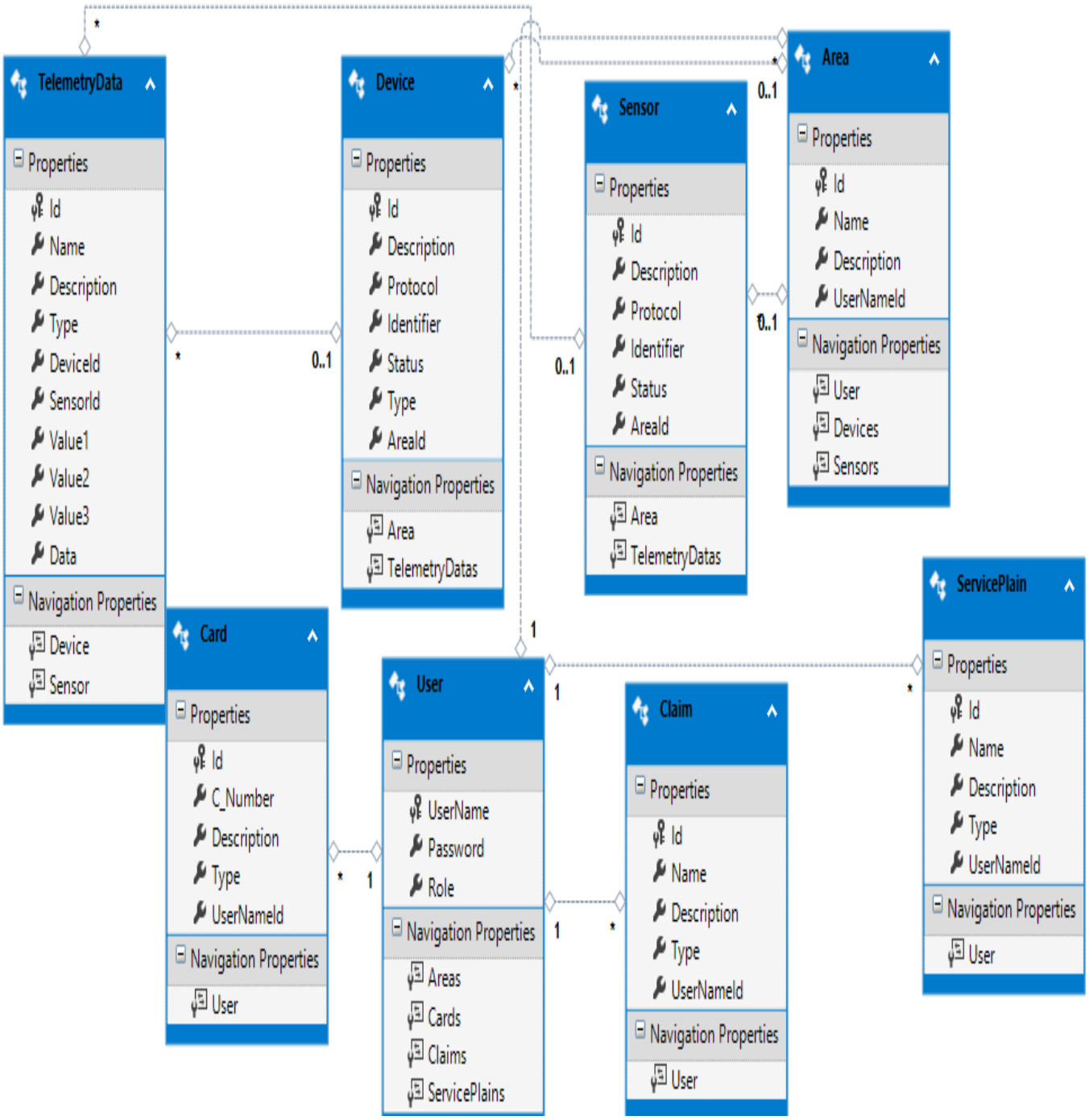

Figure 3. 13 Database Schema diagram

3.7. User Interface

Home Page

This User interface is our Dashboard in which we have news regarding Area, Devices, and Sensors etc. We have eight management options. Area, Device, Sensor, Claims, Rule engine, Data, Subscription etc.

Create Account

This is our Signup page in this page we enter credentials to sign up for this system.

Figure 3. 14 User Dashboard GUI

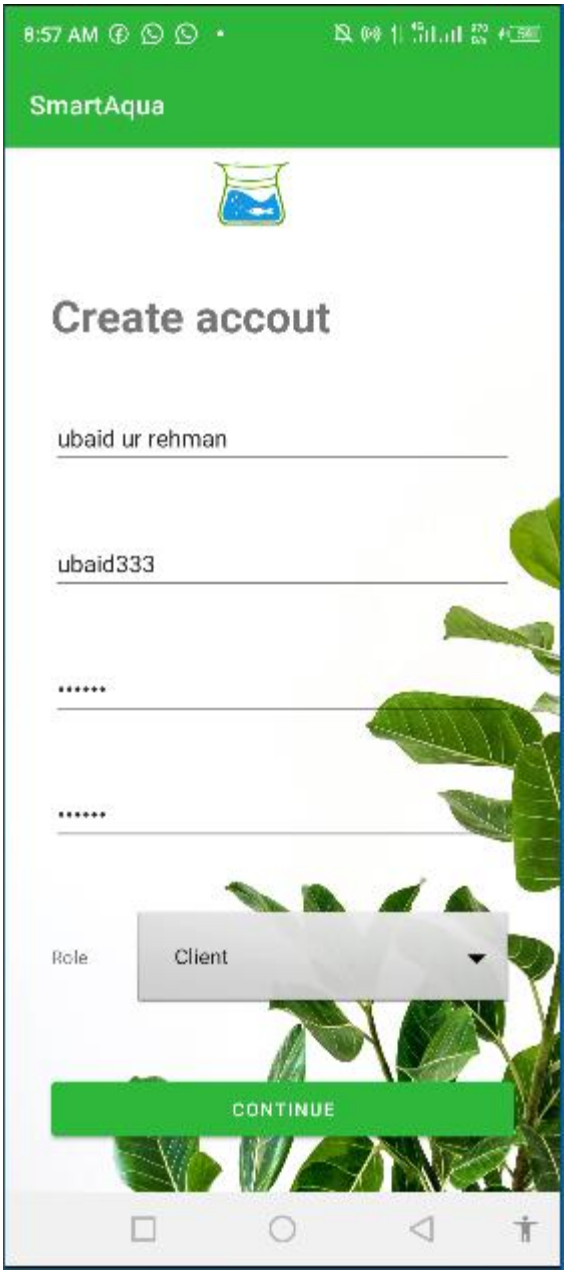

Figure 3. 15 Sign Up GUI

View Area

In Home Page when user click Area card then display a dashboard we have multiple option in which one have Area. In area management we can perform (CRUD) add, remove, update the area into the system.

Login User

After Signup and verification, user enter username and password (credentials) then we successfully enter into the system.

Figure 3. 16 View Area GUI

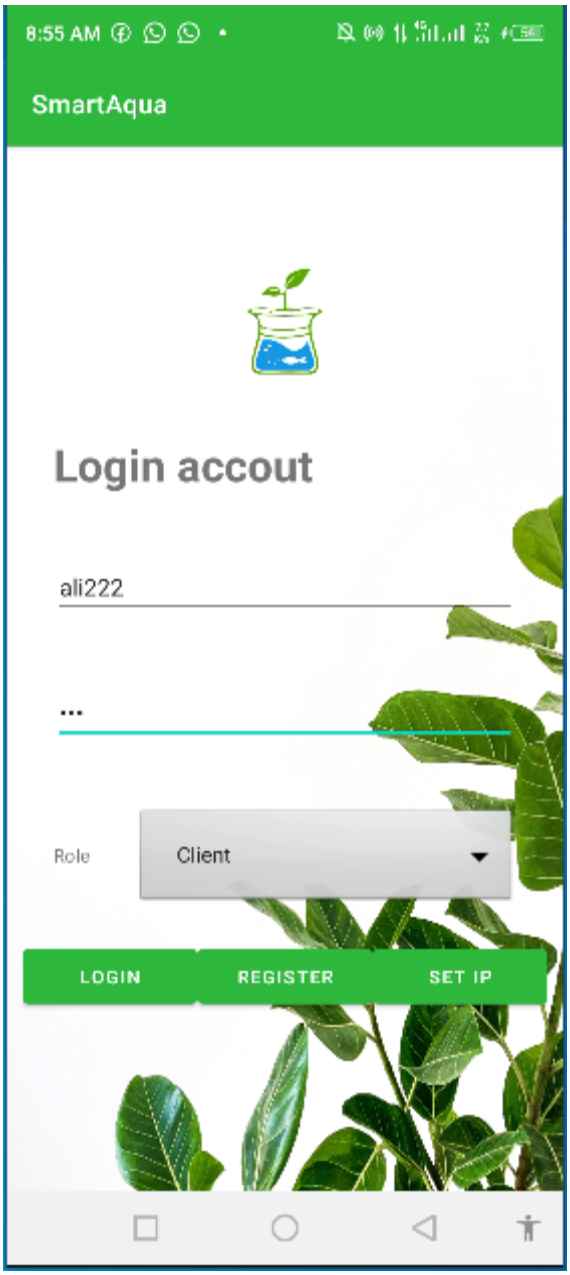

Figure 3. 17 Log in interface GUI

:

Add Area

In Area dashboard when user click on Add icon the add record form will open for record data.

Delete Area

When user click on delete icon of selected area then all detail displays in form and user press delete button.

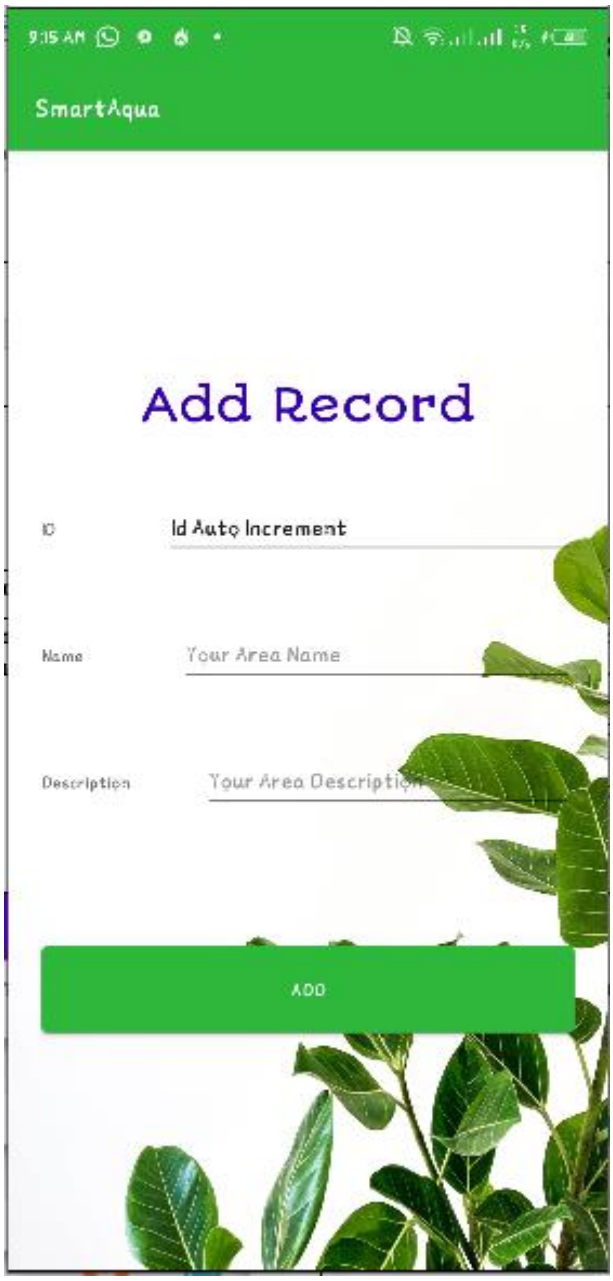

Figure 3. 18 Add Area Management GUI

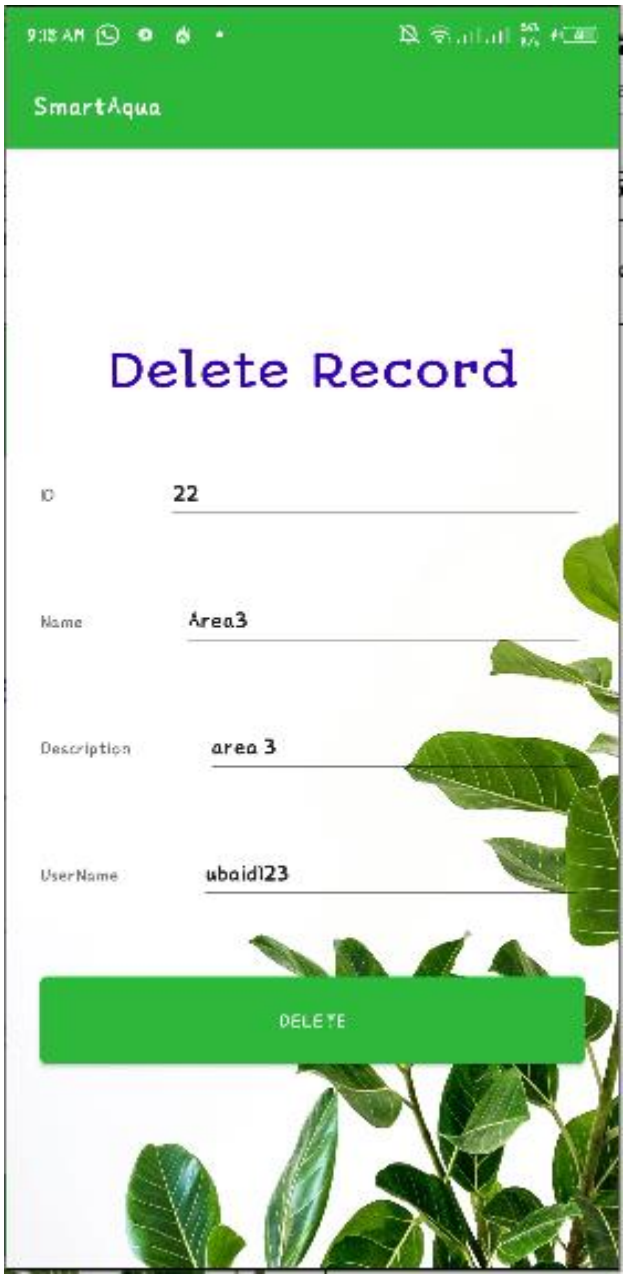

Figure 3. 19 Delete Area GUI

View Sensor

In Home Page when user click Sensor card then display a dashboard we have multiple option in which one have Sensor. In sensor management we can perform (CRUD) add, remove, update the sensor into the system.

Add Sensor

When user click on add icon in sensor dashboard then new form will open for record sensor.

Figure 3. 20 View Sensor Management GUI

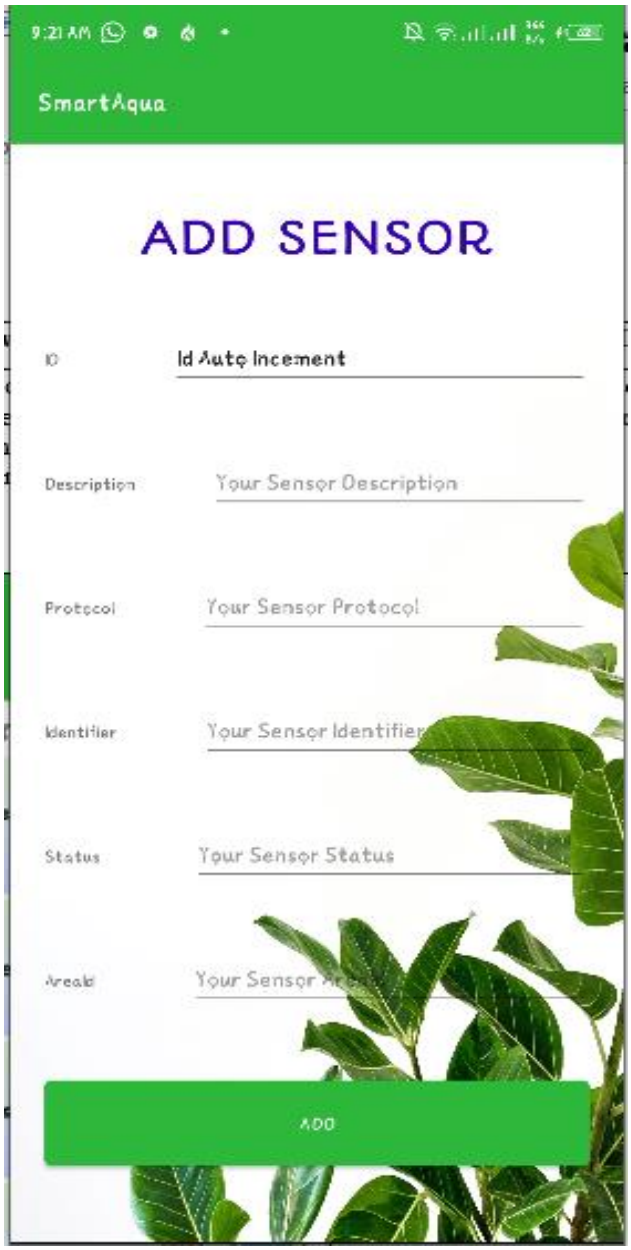

Figure 3. 21 Add Sensor GUI

Delete Sensor

When user click on delete icon of selected sensor then all detail displays in form and user press delete sensor.

Update Sensor

When user click on update icon of selected sensor then all detail displays in form and user press update sensor.

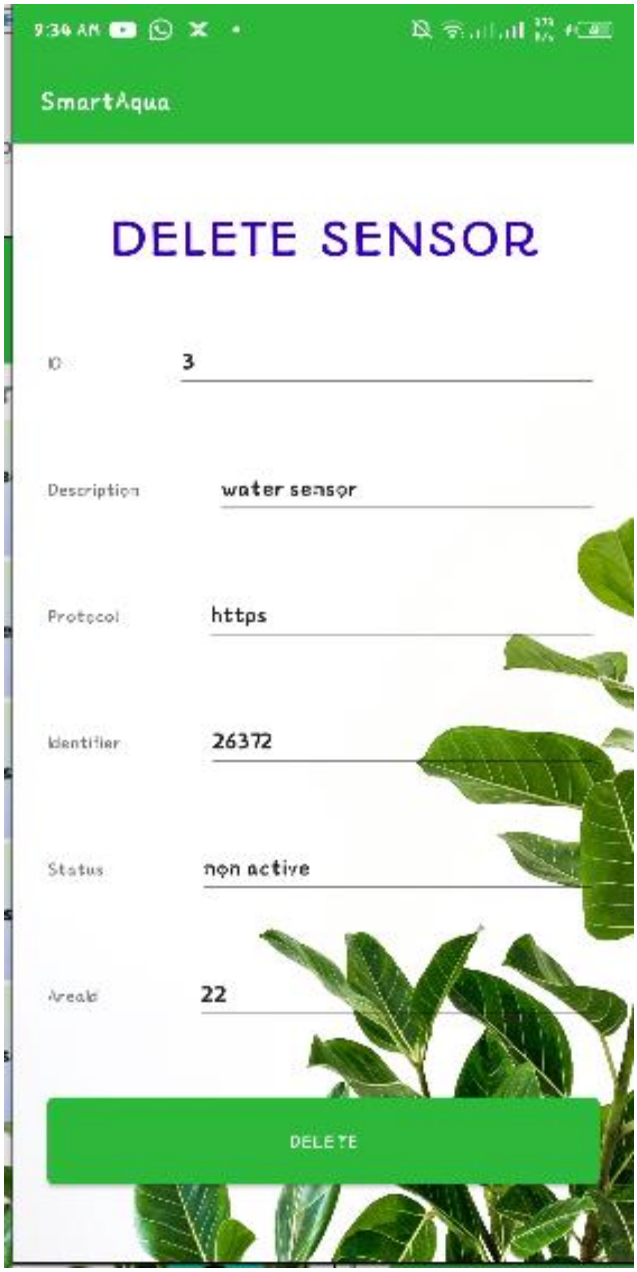

Figure 3. 22 delete Sensor Management GUI

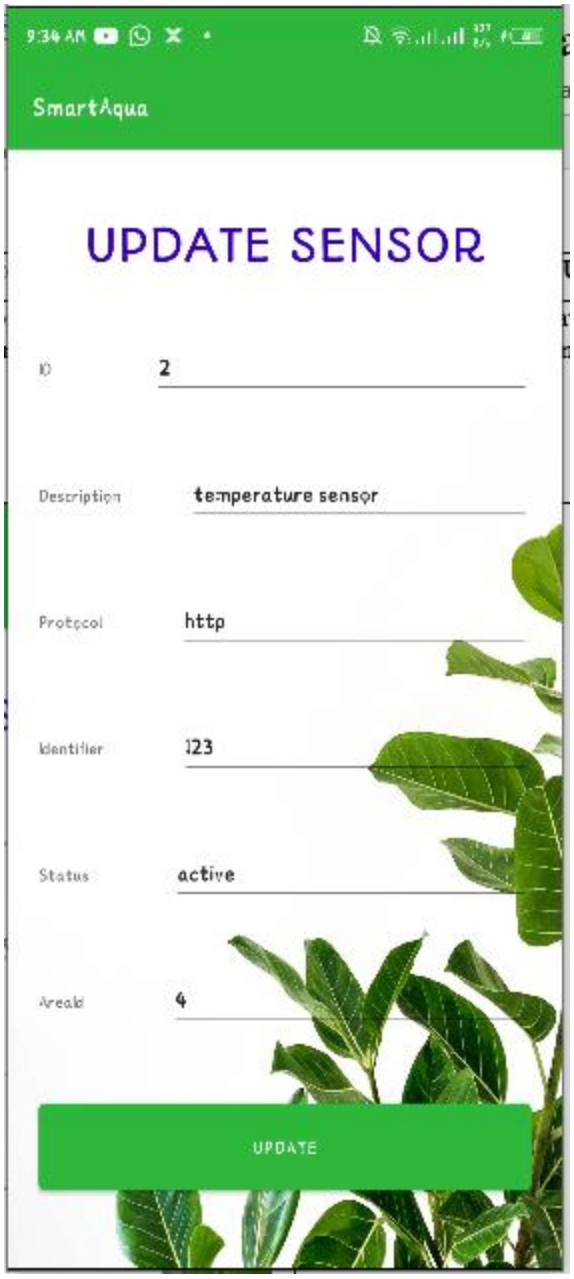

Figure 3. 23 update Sensor GUI

Add Device

In device dashboard when user click on Add icon the add record form will open for record data.

Delete Device

When user click on delete icon of selected area then all detail displays in form and press delete button for successfully deleted device.

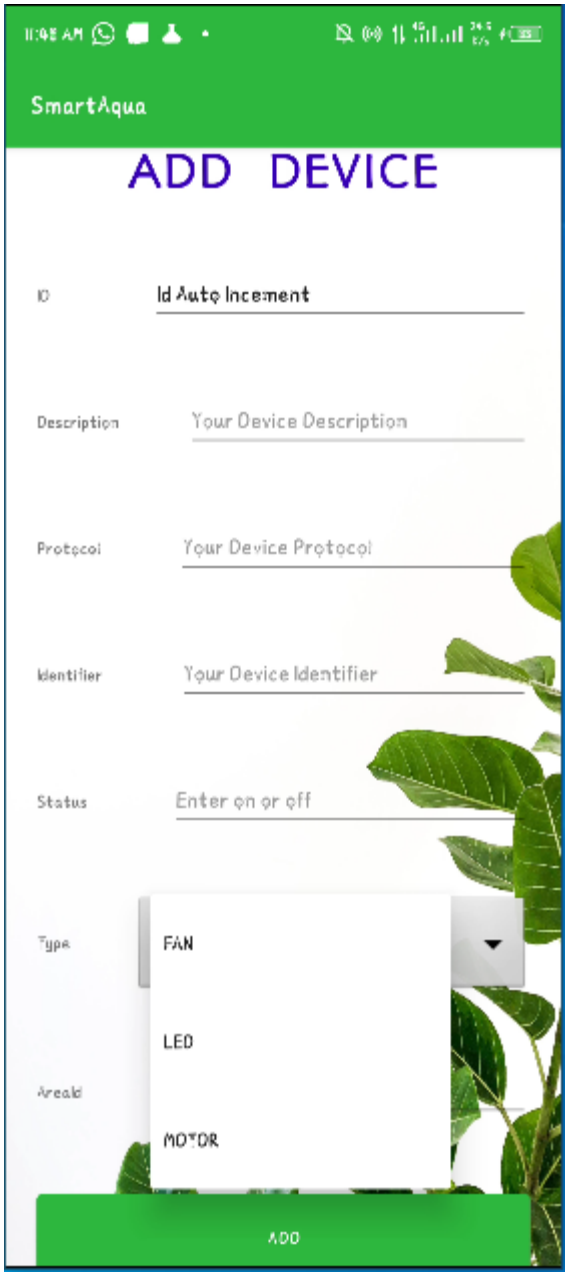

Figure 3. 24 add device Management GUI

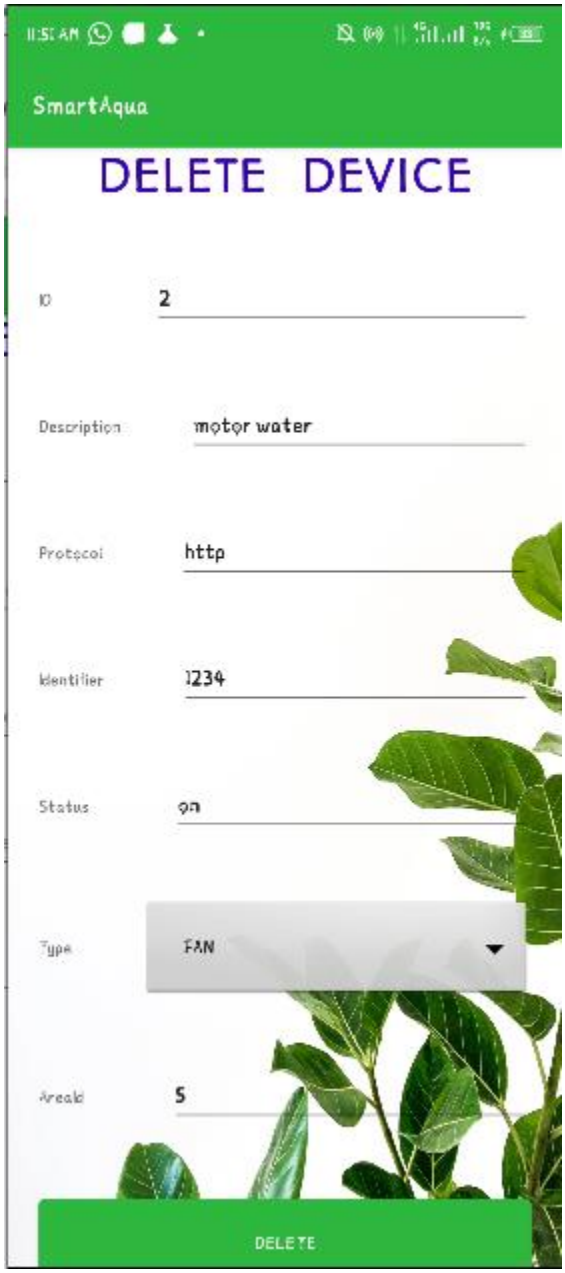

Figure 3. 25 delete device GUI

View Device

In device dashboard when user click on view icon the list of devices displays on mobile interface.

Update Device

When user click on update icon of selected device then all detail displays in form and press update after modification.

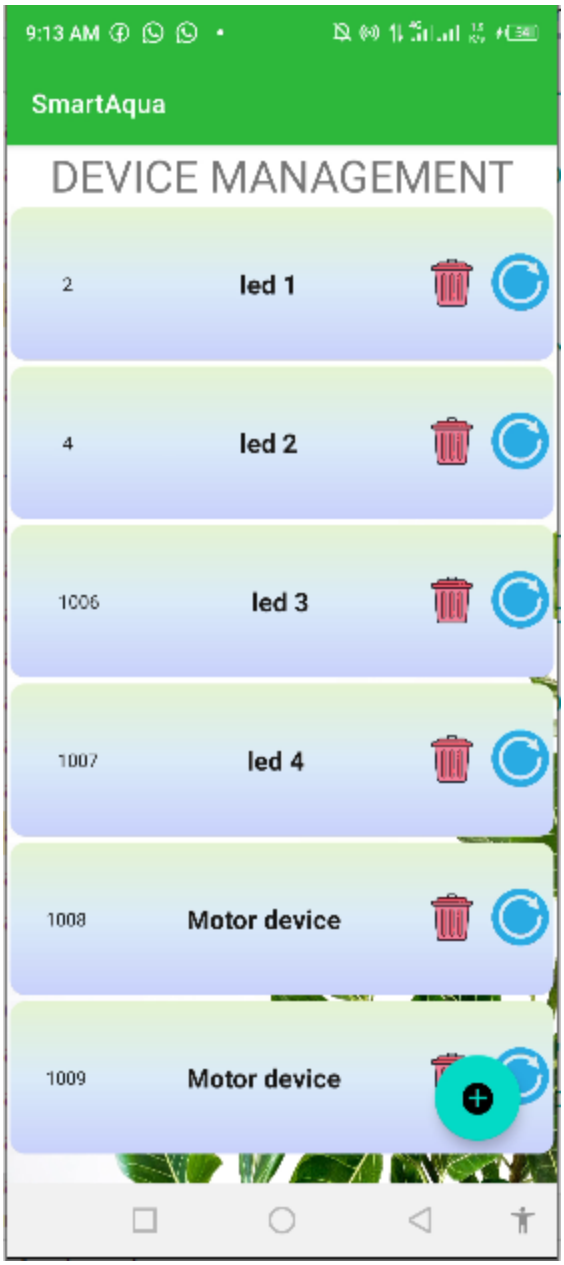

Figure 3. 26 view device Management GUI

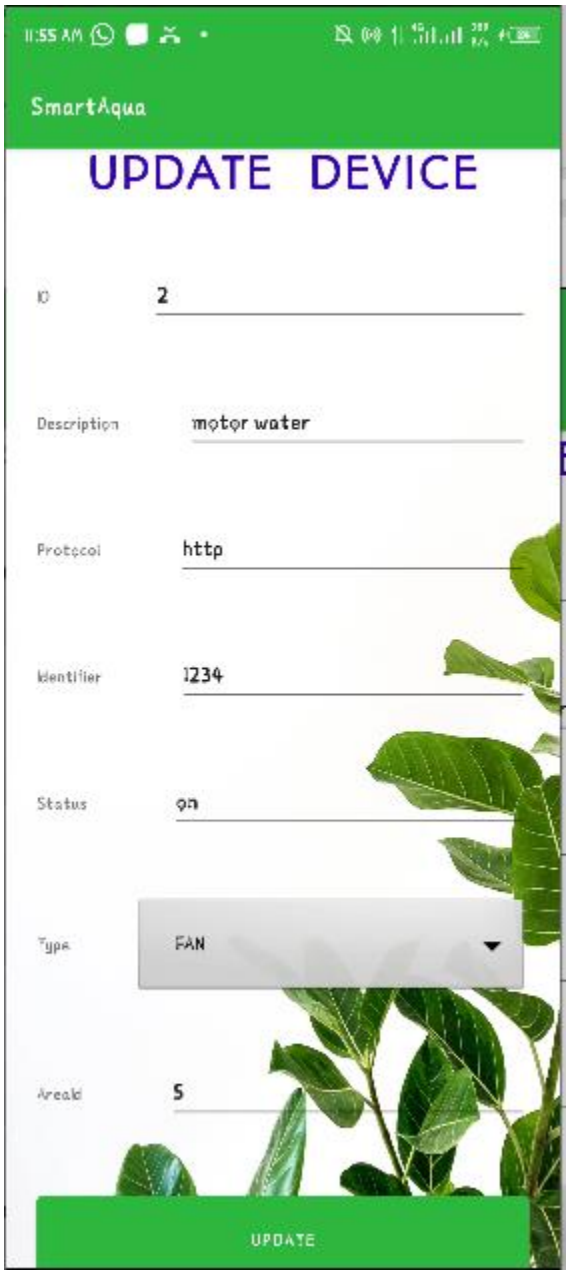

Figure 3. 27 update device GUI

Add Service Plain

In service plain dashboard when user click on Add icon the add record form will open for record data.

Delete Service Plain

When user click on service plain icon of selected service plain then all detail displays in form and user press delete for delete record.

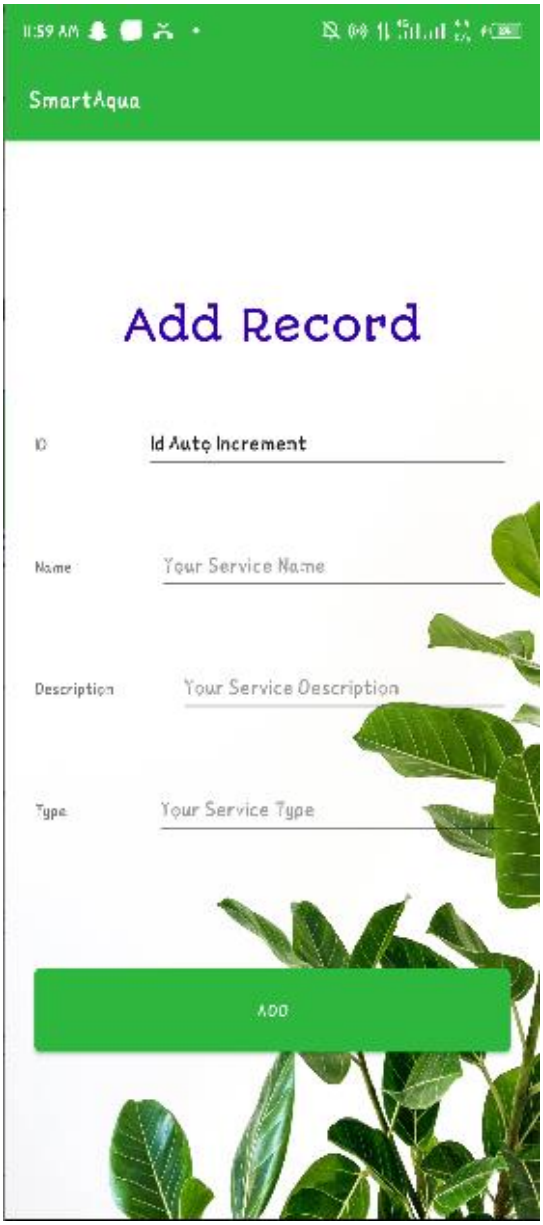

Figure 3. 28 add service plain Management GUI

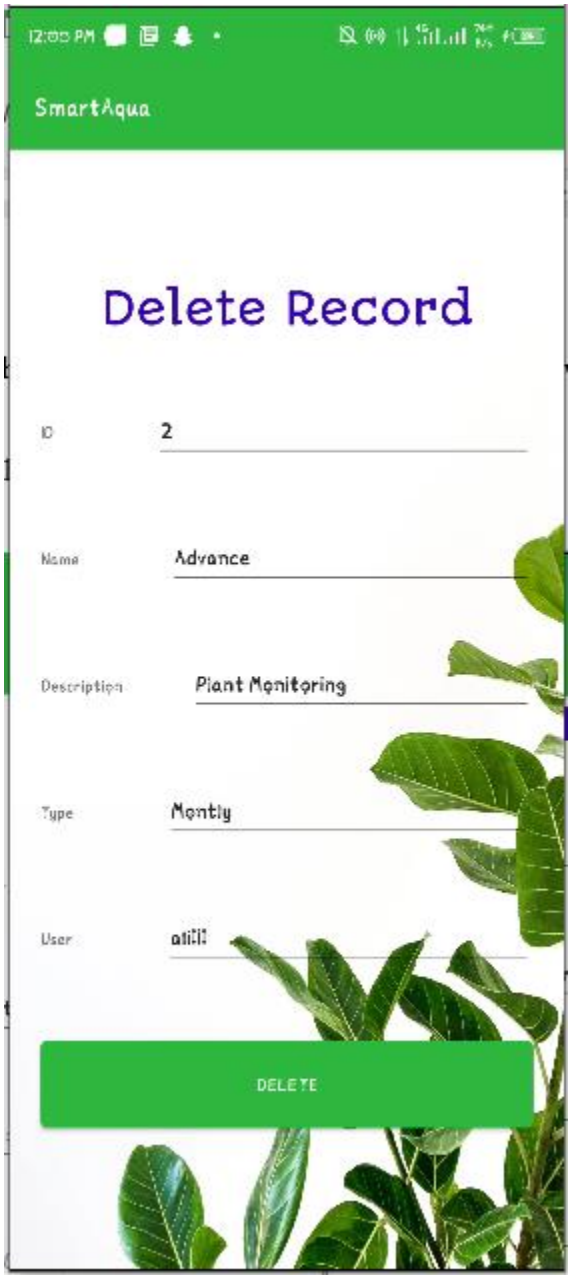

Figure 3. 29 delete service plain GUI

View Service Plain

In service plain dashboard when user click on view icon the list of service plains displays of mobile interface.

Update Service Plain

When user click on update icon of selected service plain then all detail display in form and user press update button after modification.

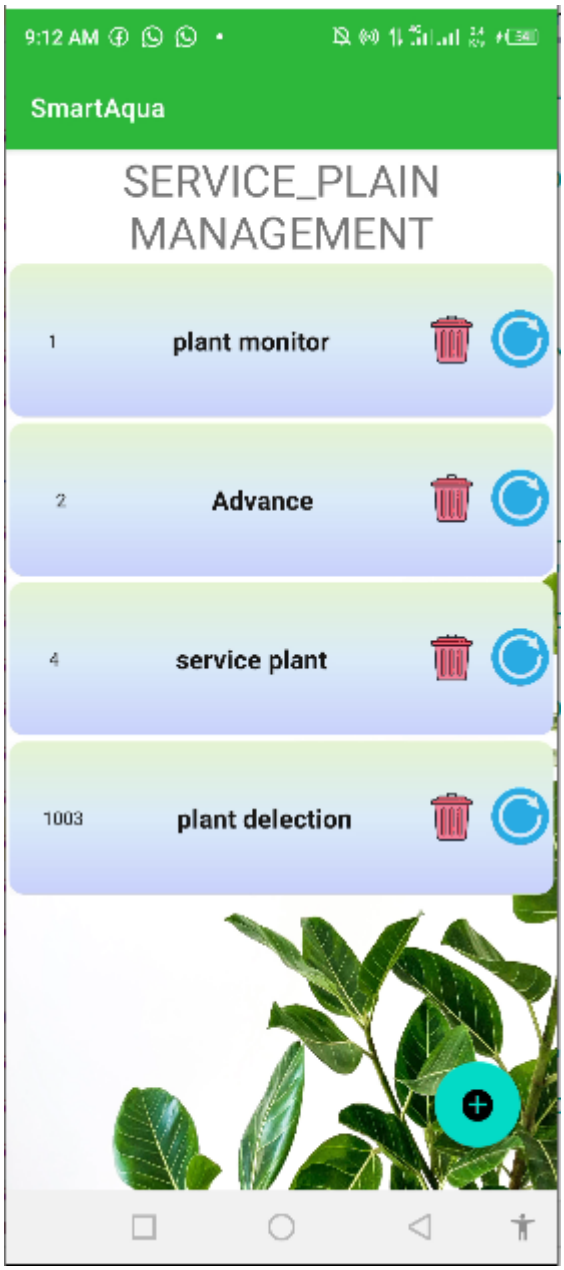

Figure 3. 30 view service plain Management GUI

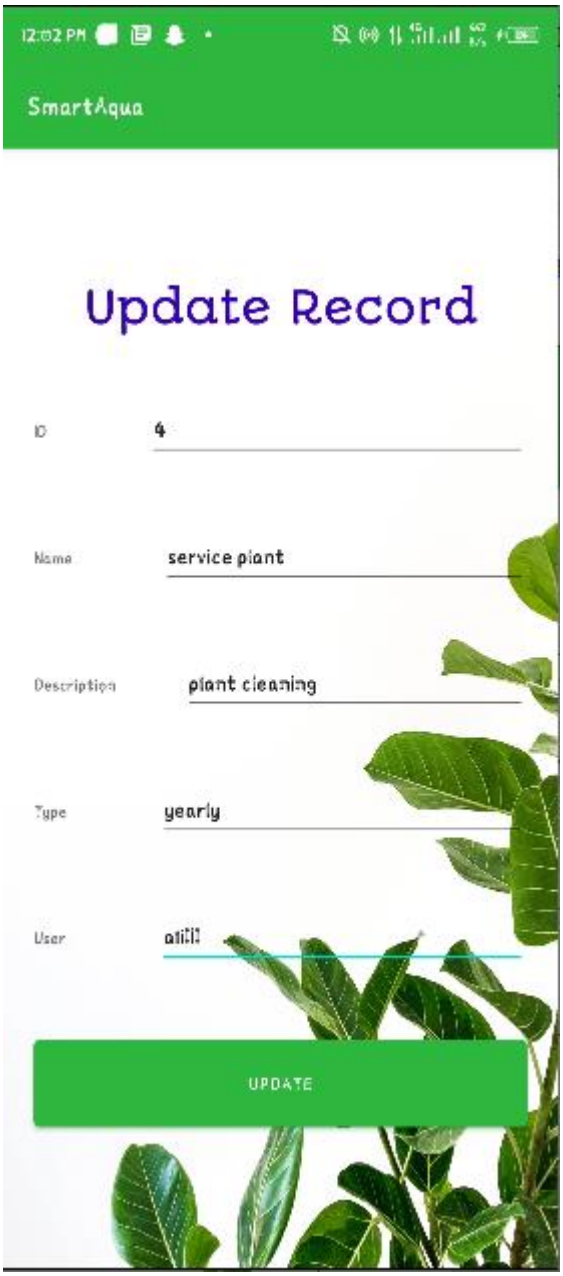

Figure 3. 31 update service plain GUI

Control Area

In mobile interface the user control and monitor the all devices and sensor which are place in particular area by changing their status (on or off the LED) etc.

Figure 3. 32 monitor area Management GUI

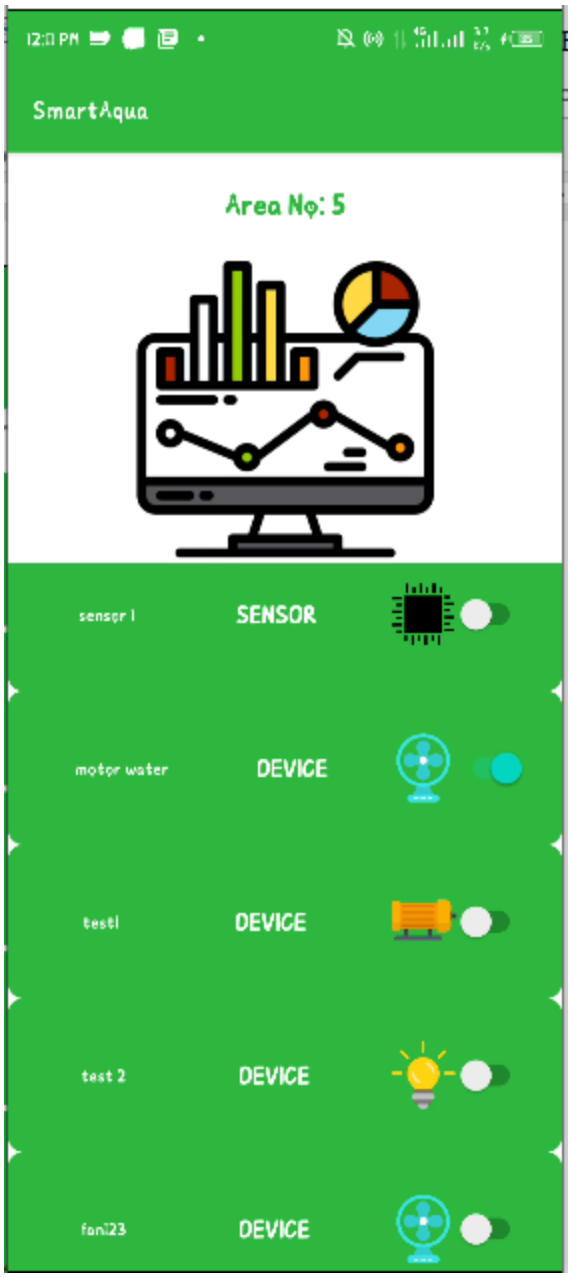

Figure 3. 33 control Area GUI

Get Notification

In mobile interface the user get notification after update status of device.

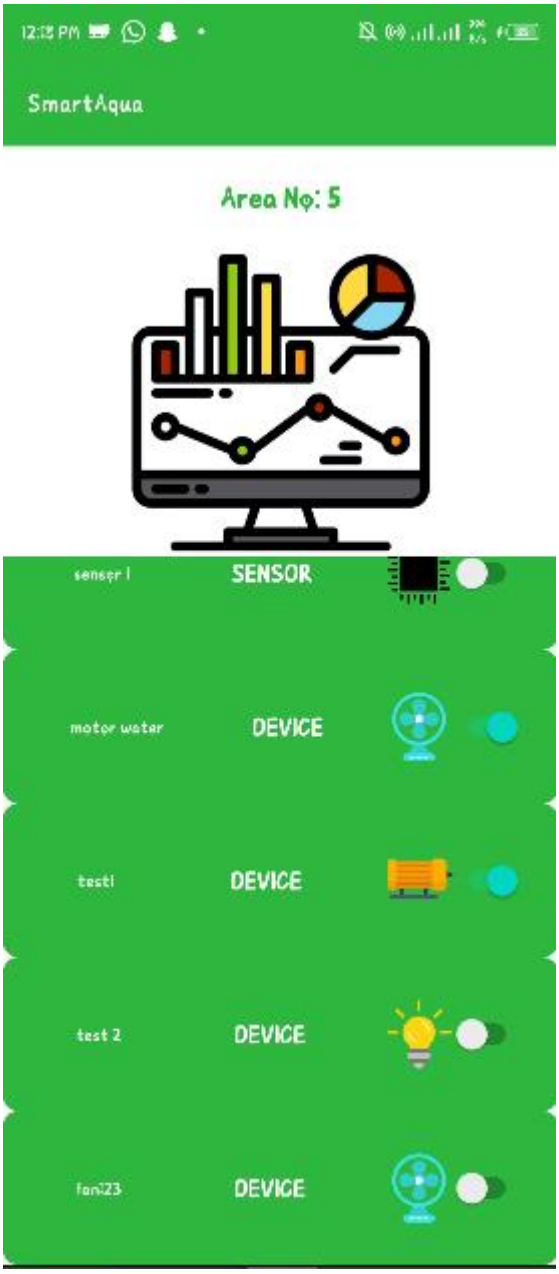

Figure 3. 34 control device Management GUI

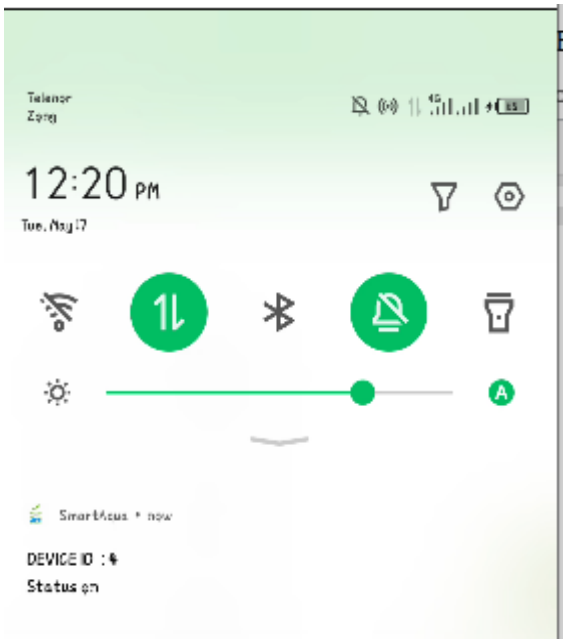

Figure 3. 35 get notification GUI

View Telemetry Data

In mobile interface the user view telemetry data of sensor.

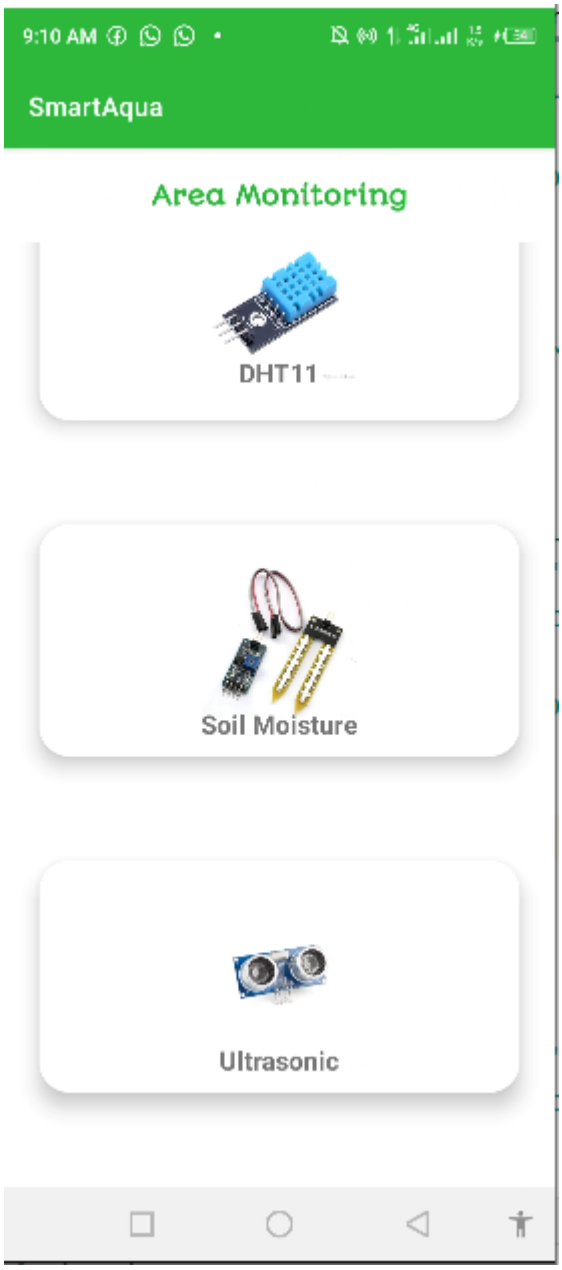

Figure 3. 36 View Telemetry Sensors GUI

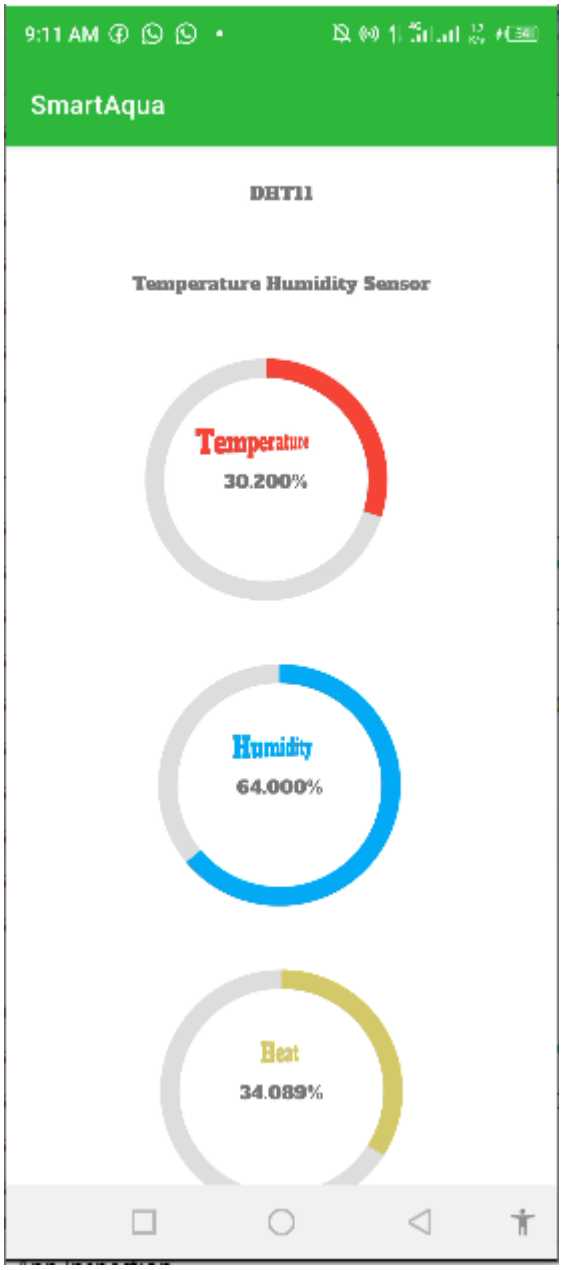

Figure 3. 37 get progress GUI

View Popup Interface

In mobile interface the user view popup automates system and popup for IP server interface.

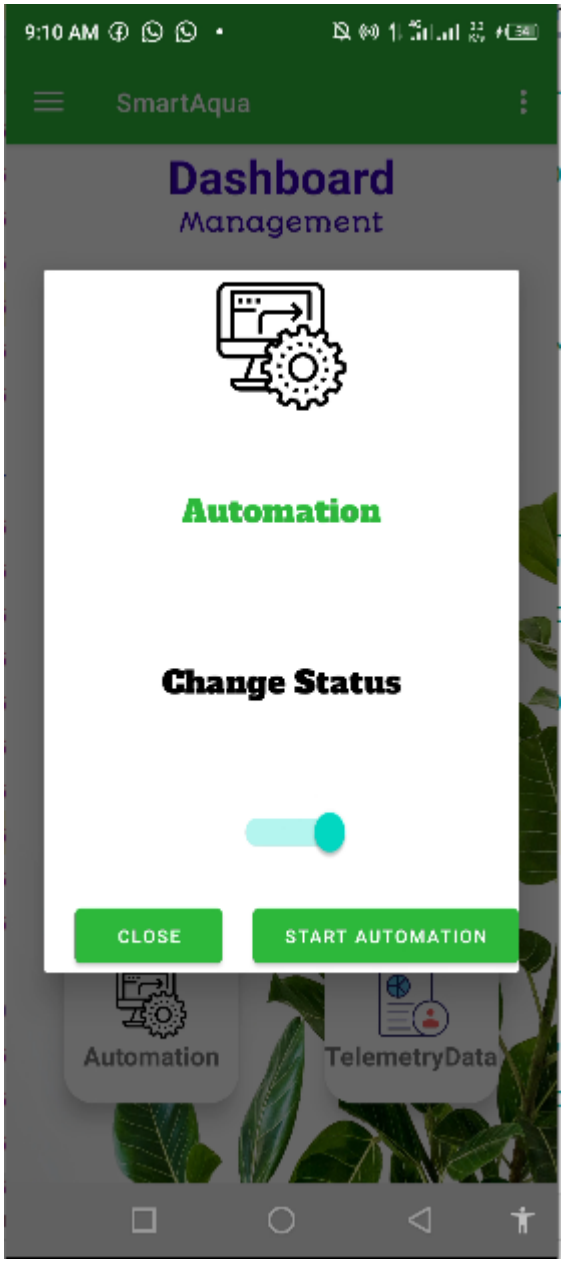

Figure 3. 38 View Automation GUI

Figure 3. 39 add server IP GUI

View Service Plain

In mobile interface the user view service plain.

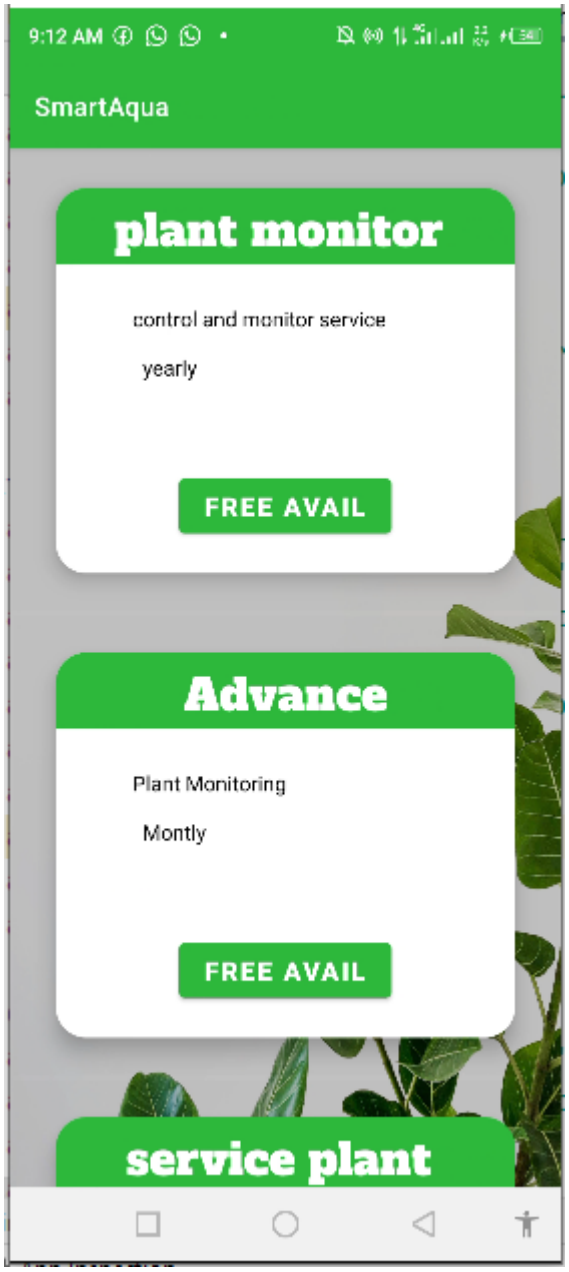

Figure 3. 40 View Service Plain GUI

Figure 3. 41 Service Plain Management GUI

Chapter 4

Software Development

4.1. Coding Standards

We used the camel notation naming standards during the development of our system. For Example, **to create the variable for admin name the syntax should be adminName.** Similarly, for the **method / functions the syntax is getAdimName ().** We mentioned the proper comments in code to make it understandable.

4.2. Development Environment

We implement our system in the following programming languages.

a. Java
b. ASP**.** NET Web API Framework
c. Visual Studio Database
d. XML

The tools we used for the development purpose are following

a. Android Studio
b. Visual Studio 2017
c. Internet Information Services.

Reason for the Selection of these Technologies.

We selected the Java for the front-end development because it is the latest technology and facilitate us with the single page mobile application which will reduce the server response time and try to get only required data from the server and did not whole page of the web on every request

We used Web API Framework latest version launched by the Microsoft for the backend. We create Web API's in this framework and put our logic there. This frame is tending in market and reliable and competitive solution for the backend in market.

4.3. Software Description

The Module of the Smart Aqua is following which should be implemented in this iteration.

Register User.

In register major challenge we faced during the implementation of this module to connect front end with back end using web API.

In this description the user register himself by unique username if user already exit then API return 409 error which indicate the duplication of username.

Snippets of the Code Post Data for register

Code to Post Data on the Web API to store in a Database so that admin can manage it.

We use doInBackground for post data using web API

```
        try {
            JsonObject postData = new JsonObject();
            postData.addProperty( property: "UserName", userName);
            postData.addProperty( property: "Password", pass);
Log.i( tag: "key1", msg: userName+" "+pass);
            URL url = new URL(UrlAttributeHelper.myURLUser);
            urlConnection = (HttpURLConnection) url.openConnection();
            urlConnection.setRequestProperty("Content-Type", "application/json");
            urlConnection.setRequestMethod("POST");
            urlConnection.setDoOutput(true);
            urlConnection.setDoInput(true);
            urlConnection.setChunkedStreamingMode(0);
            OutputStream out = new BufferedOutputStream(urlConnection.getOutputStream());
            BufferedWriter writer = new BufferedWriter(new OutputStreamWriter(
                    out, charsetName: "UTF-8"));
            writer.write(String.valueOf(postData));
            writer.flush();

            code  = urlConnection.getResponseCode();
```

Figure 4. 1 Post User Data Code

Login User.

The main problem we faced during the implementation of this module is keep the API URL same when user connect multiple networks. So if user correctly enter username and its password then system give permission to access home page of this module.

Snippets of the Code GET Data for Login.

```
@Override
protected String doInBackground(String... args) {
    // Like StringBuffer, use to generate String from output stream of data.
    StringBuilder result = new StringBuilder();
    try {
        // Original
        //URL ip
        URL url = new URL(UrlAttributeHelper.myURLUser);
        urlConnection = (HttpURLConnection) url.openConnection();
        InputStream in = new BufferedInputStream(urlConnection.getInputStream());
        // To store inout stream's result temp
        BufferedReader reader = new BufferedReader(new InputStreamReader(in));
        //Combine results into single String
        String line;
        while ((line = reader.readLine()) != null) {
            result.append(line);
        }
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        urlConnection.disconnect();
    }

    return result.toString();
}
```

Project update recommended
Android Gradle Plugin can be upgraded.

Figure 4. 2 Login User Get Code

View Area

The main challenge we faced during the implementation of this module is keep the API URL same when user connect multiple networks. But in this case we modify manually URL API and get all Area record for display in Android Application.

Snippets of the Code GET Area for View Area.

The same as above post method but the changes apply on doInBackground Method

//In doInBackground only change of API

URL url = new URL(UrlAttributeHelper.*myURLArea*);

Add Area

The main important part of module to add area by auto incremental id which other detail such as its name, description with their username id which get form Share preference Library .

Snippets of the Code Post Area for Add Area.

```
try {
    JsonObject postData = new JsonObject();
    postData.addProperty( property: "Name", name);
    postData.addProperty( property: "Description", description);
    //  get share preferences user name
    SharePreferencesDAOClass sharePreferencesDAOClass = new SharePreferencesDAOClass(context);
    postData.addProperty( property: "UserNameId", sharePreferencesDAOClass.getUser());
Log.i( tag: "key", msg: "detail "+name+" "+description+" "+sharePreferencesDAOClass.getUser());
    URL url = new URL(UrlAttributeHelper.myURLArea);
    urlConnection = (HttpURLConnection) url.openConnection();
    urlConnection.setRequestProperty("Content-Type", "application/json");
    urlConnection.setRequestMethod("POST");
    urlConnection.setDoOutput(true);
    urlConnection.setDoInput(true);
    urlConnection.setChunkedStreamingMode(0);
    OutputStream out = new BufferedOutputStream(urlConnection.getOutputStream());
    BufferedWriter writer = new BufferedWriter(new OutputStreamWriter(
            out,  charsetName: "UTF-8"));
    writer.write(postData.toString());
    writer.flush();
```

Figure 4. 3 Add Area Code

View Sensor.

The main challenge we faced during the implementation of this module is keep the API URL same when user connect multiple networks. But in this case we modify manually URL API and get all sensor record for display in Android Application.

Snippets of the Code GET Sensor for View Sensor.

The same as above post method but the changes apply on doInBackground Method

```
//In doInBackground only change of API
URL url = new URL(UrlAttributeHelper.myURLArea);
```

Add Sensor.

The main important part of module to add area by auto incremental id which other detail such as its name, description with their username id which get form Share preference Library .

Snippets of the Code Post Sensor for Add Sensor.

```
// POST
try {
    JsonObject postData = new JsonObject();
    postData.addProperty( property: "Description", description);
    postData.addProperty( property: "Protocol", protocol);
    postData.addProperty( property: "Identifier", identifier);
    postData.addProperty( property: "Status", status);
    postData.addProperty( property: "AreaId",areaId );
    //  get share preferences user name
    URL url = new URL(UrlAttributeHelper.myURLDeviceAndSensor);
    urlConnection = (HttpURLConnection) url.openConnection();
    urlConnection.setRequestProperty("Content-Type", "application/json");
    urlConnection.setRequestMethod("POST");
    urlConnection.setDoOutput(true);
    urlConnection.setDoInput(true);
    urlConnection.setChunkedStreamingMode(0);
    OutputStream out = new BufferedOutputStream(urlConnection.getOutputStream());
    BufferedWriter writer = new BufferedWriter(new OutputStreamWriter(
            out, charsetName: "UTF-8"));
    writer.write(postData.toString());
    writer.flush();

    int code = urlConnection.getResponseCode();
    if (code != 201) {
        throw new IOException("Invalid response from server: " + code);
```

Figure 4. 4 Add Area Code

Update Sensor.

The sensor where the PUT method call of HttpConnection class call.

```
try {
    JsonObject postData = new JsonObject();
    postData.addProperty( property: "Id", id);
    postData.addProperty( property: "Description", description);
    postData.addProperty( property: "Protocol", protocol);
    postData.addProperty( property: "Identifier", identifier);
    postData.addProperty( property: "Status", status);
    postData.addProperty( property: "AreaId",areaId );
    URL url = new URL( spec: sharePreferencesDAOClass.getIpAddressSensor() +"/"+id);
    urlConnection = (HttpURLConnection) url.openConnection();
    urlConnection.setRequestProperty("Content-Type", "application/json");
    urlConnection.setRequestMethod("PUT");
    urlConnection.setDoOutput(true);
    urlConnection.setDoInput(true);
    urlConnection.setChunkedStreamingMode(0);
    OutputStream out = new BufferedOutputStream(urlConnection.getOutputStream());
    BufferedWriter writer = new BufferedWriter(new OutputStreamWriter(
            out,  charsetName: "UTF-8"));
    writer.write(postData.toString());
    writer.flush();
    int code = urlConnection.getResponseCode();
    if (code != 201) {
        throw new IOException("Invalid response from server: " + code);
```

Figure 4. 5 Update Sensor

Delete Sensor.

The sensor where the Delete method call of HttpConnection class call.

```
// Delete
try {
    JsonObject postData = new JsonObject();
    URL url = new URL( spec: sharePreferencesDAOClass.getIpAddressSensor() +"/"+id);
    urlConnection = (HttpURLConnection) url.openConnection();
    urlConnection.setRequestProperty("Content-Type", "application/json");
    urlConnection.setRequestMethod("DELETE");
    urlConnection.setDoOutput(true);
    urlConnection.setDoInput(true);
    urlConnection.setChunkedStreamingMode(0);
    urlConnection.connect();
    int code = urlConnection.getResponseCode();
    if (code != 201) {
        throw new IOException("Invalid response from server: " + code);
    } else {
        // when successfully add data
  intent = new Intent(context.getApplicationContext(), SensorAndDeviceManagementRecyclerView2.class);
intent.putExtra(IntentAttributeHelper.MANAGE_TYPE_INTENT_KEY,IntentAttributeHelper.MANAGE_AREA_INTENT_VALUE);
        intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
        context.startActivity(intent);
        ((RegisterPage1) context).finish();
    }
    BufferedReader rd = new BufferedReader(new InputStreamReader(
            urlConnection.getInputStream()));
    String line, mydata2 = " ";
```

Figure 4. 6 Delete Sensor

Update Arduino Sensors

The Arduino where DHT11, Soil Moisture and Ultrasonic Sensor Update their value on server by PUT method.

TEST5 | Arduino 1.8.19

File Edit Sketch Tools Help

TEST5

```
void PutSoilMoisureSensor(int soil){
 // Prepare JSON document
DynamicJsonDocument doc(2048);
doc["Id"] = 1;
doc["Name"] = "Soil Moisture";
  doc["Description"] = "Read Soil Moisture value ";
  doc["Type"] = "Sensor";
//  doc["DeviceId"] = "";
  doc["SensorId"] = 3;
  doc["Value1"] = String(soil);
//  doc["Value2"] = String(humidity,3);
//  doc["Value3"] = String(heat,3);
  doc["Data"] = "Soil Moisture";

// Serialize JSON document
String json;
serializeJson(doc, json );

WiFiClient client;  // or WiFiClientSecure for HTTPS
HTTPClient http;
 String contentType = "application/x-www-form-urlencoded";
```

Figure 4. 7 Arduino Update Sensor

Water Pump Automation

The Arduino where Soil Moisture value automate the water pump by continuously taking value from sensor.

TEST5_RELAY_DEVICES | Arduino 1.8.19

File Edit Sketch Tools Help

TEST5_RELAY_DEVICES §

```
  Serial.print("Id :");
Serial.println(Id);
  const char * Status=doc["Value1"];
  Serial.print("Status Automation : ");
Serial.println(Status);
soilMoistureString=String(Status);
soilMoistureInt=soilMoistureString.toInt();

if(checkAutomation=="true"){
  Serial.println("Automation Apply");
   if(soilMoistureInt<50)
        {
          Serial.println("soilMoisture < 50 Pump OFF ");
            digitalWrite(waterPumpMotor, LOW);
        }
       else
       {
         Serial.println("soilMoisture > 50 Pump ON ");
           digitalWrite(waterPumpMotor,HIGH);
       }
```

Figure 4. 8 Water Pump Automation

Chapter 5
Software Testing

5.1. Testing Environment

We are manually testing the interface by inputting several valid and invalid inputs rather than using a testing IDE or framework.

5.2. Test Cases:

5.2.1. Test Case 1 Register User:

Table 5. 1 Test 1 Register user

Test Case: User register to the system

Test Description : Check User Register with valid Data	**Test Case no :** 1
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User register to the system successfully	**Test ID:** 01
Version: 1	**Test Type:** System testing
Scenario: User go to register page. User enter name User enter username User enter password User enter re-password User select role User press register button.	
Test Data: Name: ubaid ur Rehman Username: ubaid12345 Password: 12345 Re-Password: 12345 Role: Client	
Expected Result: User should Register into an application.	
Actual Result: User register to the system successfully.	
Status: Passed	

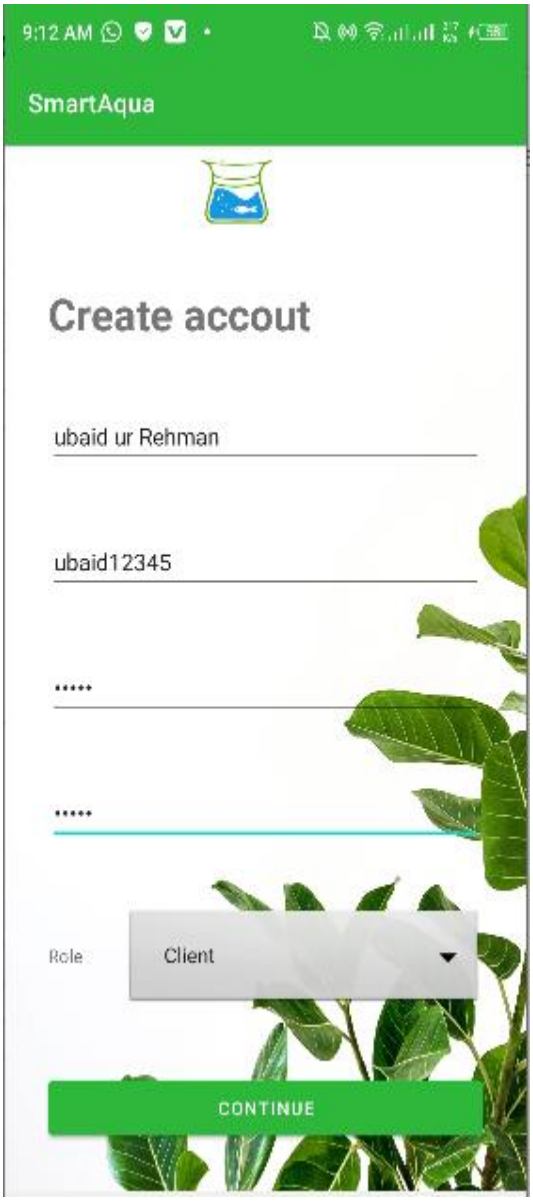

Figure 5. 1 Register user test1

5.2.2. Test Case 2 Login user:

Table 5. 2 Test 2 Login User

Test Case: Login user

Test Description: Check User Login with valid Data	**Test Case no :** 2
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can login to the system Successfully	**Test ID:** 02
Version: 1	**Test Type:** System testing
Scenario: User go to login page User enter email and password User enter Role User press login button.	
Test Data: Username: ubaid12345 Password: 12345 Role: Client	
Expected Result: User should Login into an application.	
Actual Result: User Login to the system successfully.	
Status: Passed	

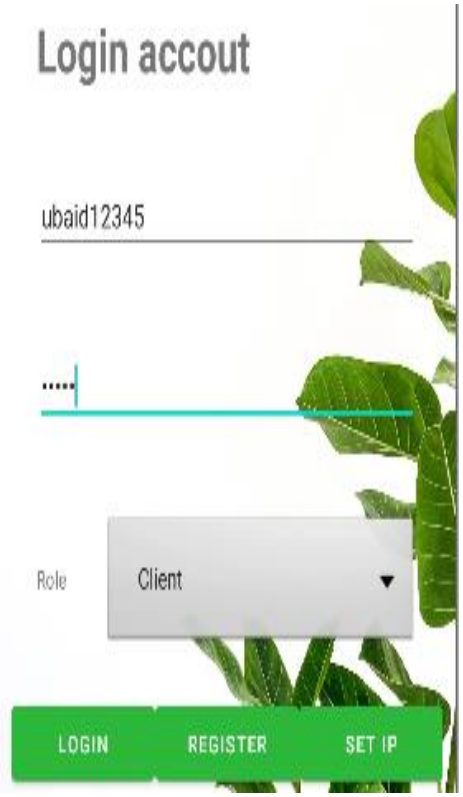

Figure 5. 2 Login test2

5.2.3. Test Case 3 Login without valid Data:

Table 5. 3 Test 3 Login without entering Email

Test Case: login to the system without entering Email

Test Description: Check User Login without valid Data	**Test Case no :** 3
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User cannot login to the system successfully	**Test ID:** 03
Version: 1	**Test Type:** System testing
Scenario: User go to login page. User enter password without entering email User enter role User press login button.	
Test Data: Username: Password: 12345 Role: Client	
Expected Result: User should cannot login to the system successfully	
Actual Result: User cannot login to the system successfully	
Status: Passed	

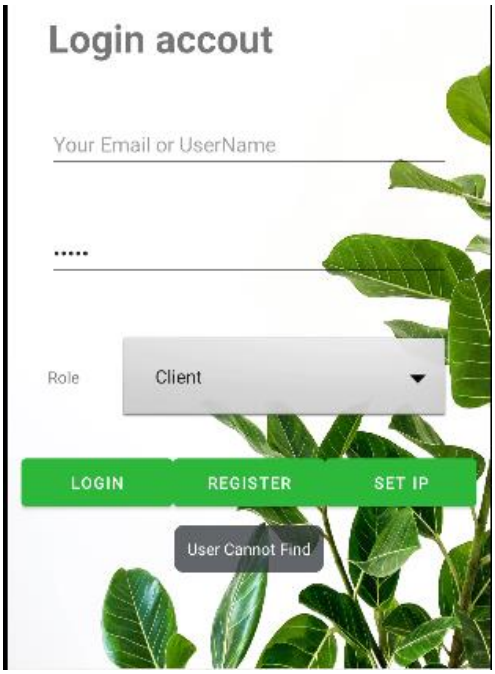

Figure 5. 3 User Login test3

5.2.4. Test Case 4 Login without entering password:

Table 5. 4 Test 4 Login without entering password

Test Case: login to the system without entering password

Test Description: login to the system without entering password	**Test Case no :** 4
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User cannot login to the system successfully.	**Test ID:** 04
Version: 1	**Test Type:** System testing
Scenario: User go to login page. User enter email without entering password User press login button.	
Test Data: Username: ubaid123 Password:	
Expected Result: User should cannot login to the system successfully.	
Actual Result: User cannot login to the system successfully	
Status: Passed	

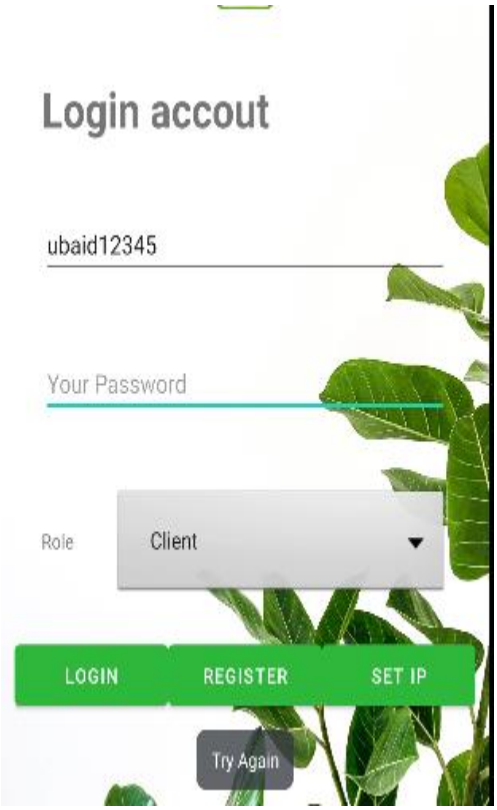

Figure 5. 4 Login test4

5.2.5. Test Case 5 View Area:

Table 5. 5 Test 5 View Area

Test Case: View Area

Test Description: view area by selecting button area card	**Test Case no :** 5
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can view list of area in list form.	**Test ID:** 05
Version: 1	**Test Type:** System testing
Scenario: User go to Home page. User select area button. User view all list of areas.	
Test Data: Input: Press area button	
Expected Result: User should be able to view list of area successfully.	
Actual Result: User view list of area successfully	
Status: Passed	

Figure 5. 5 Test 5 View Area

5.2.6. Test Case 6 Add Area:

Table 5. 6 Test 6 Add Area

Test Case: Add Area

Test Description: Check User add area in system.	**Test Case no :** 6
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can add area into the system successfully.	**Test ID:** 06
Version: 1	**Test Type:** System testing
Scenario: User go to Area Home page. User select add button. User enter area name and description of area. User press add button. User add area successfully.	
Test Data: Name: Plant Area Description: Winter plant are growth here with in 3 month.	
Expected Result: User should be add area into the system successfully.	
Actual Result: User add area into the system successfully.	
Status: Passed	

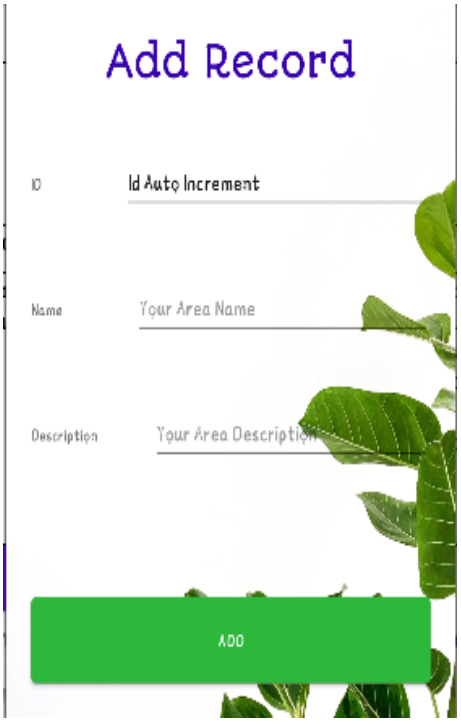

Figure 5. 6 Test Add Area

5.2.7. Test Case 7 View Sensor:

Table 5. 7 Test 7 View Sensor

Test Case: View Sensor

Test Description: view sensor by selecting button Sensor card	**Test Case no :** 7
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can view list of Sensor in list form.	**Test ID:** 07
Version: 1	**Test Type:** System testing
Scenario: User go to Home page. User select Sensor button. User view all list of Sensors.	
Test Data: Input: Press sensor button	
Expected Result: User should be able to view list of Sensor successfully.	
Actual Result: User view list of Sensor successfully	
Status: Passed	

Figure 5. 7 Test 7 View Sensor

5.2.8. Test Case 8 Add Sensor:

Table 5. 8 Test 8 Add Sensor

Test Case: Add Sensor

Test Description: Add Sensor by selecting button of Add.	**Test Case no :** 8
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can add Sensor into the system successfully.	**Test ID:** 08
Version: 1	**Test Type:** System testing
Scenario: User go to Sensor Home page. User select add button. User enter area description, protocol, identifier, area id of Sensor. User press add button. User add Sensor successfully.	
Test Data: Name: Moisture Sensor Protocol: https Identifier:1234 Status: off AreaId:5	
Expected Result: User should add Sensor into the system successfully.	
Actual Result: User add Sensor into the system successfully.	
Status: Passed	

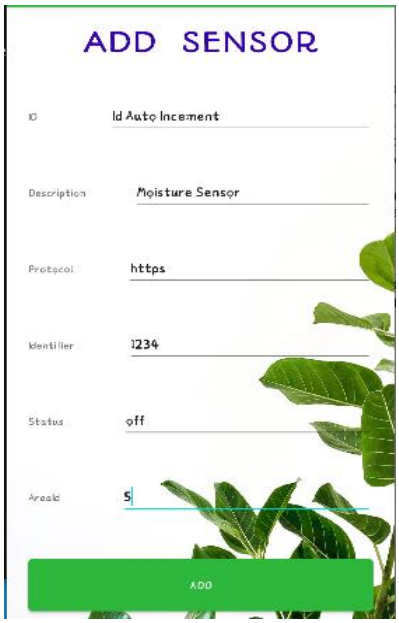

Figure 5. 8 Test 8 Add Sensor

5.2.9. Test Case 9 Update Area:

Table 5. 9 Test 9 Update Area

Test Case: Update Area

Test Description: Update Area by selecting button of Update.	**Test Case no :** 9
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can update area into the system successfully.	**Test ID:** 09
Version: 1	**Test Type:** System testing
Scenario: User go to Area Home page. User select selected area update icon. User update area by its name. User press update button. User update area successfully.	
Test Data: Name: Fishing Area	
Expected Result: User should update area into the system successfully.	
Actual Result: User update area into the system successfully.	
Status: Passed	

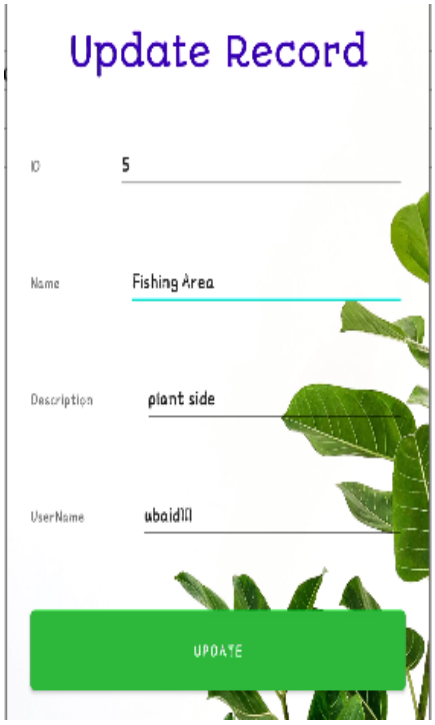

Figure 5. 9 Test 9 Update Area

5.2.10. Test Case 10 Delete Area:

Table 5. 10 Test 10 Delete Area

Test Case: Delete Area

Test Description: delete Area by selecting button of delete.	**Test Case no :** 10
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can delete area into the system successfully.	**Test ID:** 010
Version: 1	**Test Type:** System testing
Scenario: User go to Area Home page. User click selected area delete icon. User press delete button.	
Test Data: input: delete button	
Expected Result: User should delete area into the system successfully.	
Actual Result: User delete area into the system successfully.	
Status: Passed	

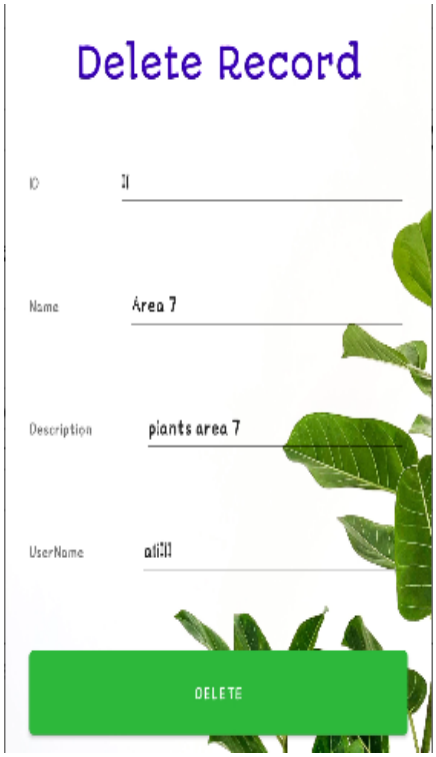

Figure 5. 10 Test 10 Delete Area

5.2.11. Test Case 11 Update Sensor:

Table 5. 11 Test 11 Update Sensor

Test Case: Update Sensor

Test Description: Update Sensor by selecting button of Update.	**Test Case no :** 11
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can update sensor into the system successfully.	**Test ID:** 011
Version: 1	**Test Type:** System testing
Scenario: User go to sensor Home page. User select selected sensor update icon. User update sensor by its status. User press update button. User update sensor successfully.	
Test Data: Status: off	
Expected Result: User should update sensor into the system successfully.	
Actual Result: User update sensor into the system successfully.	
Status: Passed	

Figure 5. 11 Test 11 Update Sensor

5.2.12. Test Case 12 Delete Sensor:

Table 5. 12 Test 12 delete Sensor

Test Case: Delete Sensor

Test Description: delete Sensor by selecting button of delete.	**Test Case no :** 12
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can delete sensor into the system successfully.	**Test ID:** 012
Version: 1	**Test Type:** System testing
Scenario: User go to sensor Home page. User click selected sensor delete icon. User press delete button.	
Test Data: input: delete button	
Expected Result: User should delete sensor into the system successfully.	
Actual Result: User delete sensor into the system successfully.	
Status: Passed	

Figure 5. 12 Test 12 Delete Sensor

5.2.13. Test Case 13 View Device:

Table 5. 13 Test 13 View Device

Test Case: View Device

Test Description: view device by selecting button Sensor card	**Test Case no :** 13
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can view list of device in list form.	**Test ID:** 013
Version: 1	**Test Type:** System testing
Scenario: User go to Home page. User select device button. User view all list of devices.	
Test Data: Input: Press device button	
Expected Result: User should be able to view list of device successfully.	
Actual Result: User view list of device successfully	
Status: Passed	

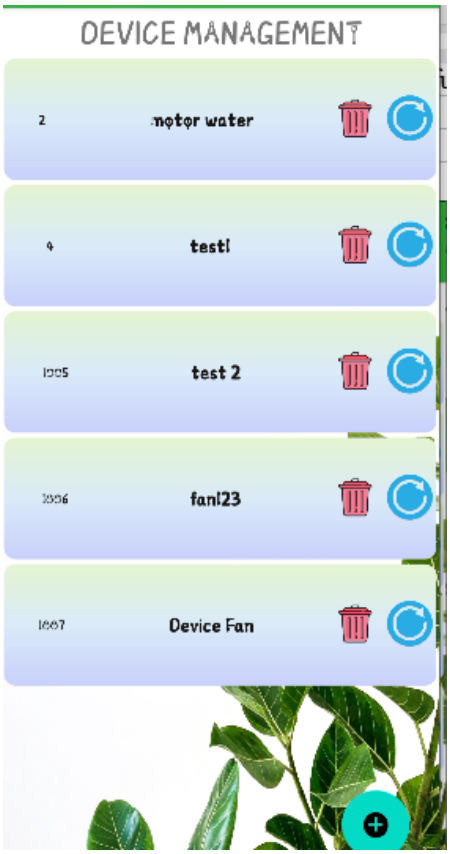

Figure 5. 13 Test 7 View Device

5.2.14. Test Case 14 Add Device:

Table 5. 14 Test 14 Add Device

Test Case: Add Device

Test Description: Add Device by selecting button of Add.	**Test Case no :** 14
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can add Device into the system successfully.	**Test ID:** 014
Version: 1	**Test Type:** System testing
Scenario: User go to Device Home page. User select add button. User enter Device description, protocol, identifier, area id of Sensor. User press add button. User add Device successfully.	
Test Data: Description: Motor device. Protocol: https Identifier:1234 Status: off Type: MOTOR AreaId:5	
Expected Result: User should add Device into the system successfully.	
Actual Result: User add Device into the system successfully.	
Status: Passed	

Figure 5. 14 Test 14 Add Device

5.2.15. Test Case 15 Update Device:

Table 5. 15 Test 15 Update Device

Test Case: Update Device

Test Description: Update Device by selecting button of Update.	**Test Case no :** 15
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can update Device into the system successfully.	**Test ID:** 015
Version: 1	**Test Type:** System testing
Scenario: User go to Device Home page. User select selected Device update icon. User update Device by its status. User press update button. User update Device successfully.	
Test Data: Status: on	
Expected Result: User should update Device into the system successfully.	
Actual Result: User update Device into the system successfully.	
Status: Passed	

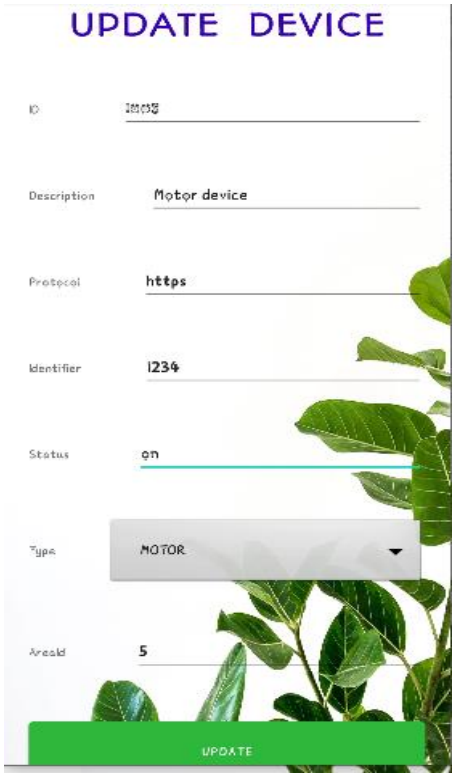

Figure 5. 15 Test 15 Update Device

5.2.16. Test Case 16 Delete Device:

Table 5. 16 Test 16 Delete Device

Test Case: Delete Device

Test Description: delete Device by selecting button of delete.	**Test Case no :** 16
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can delete Device into the system successfully.	**Test ID:** 016
Version: 1	**Test Type:** System testing
Scenario: User go to Device Home page. User click selected Device delete icon. User press delete button.	
Test Data: input: delete button	
Expected Result: User should delete Device into the system successfully.	
Actual Result: User delete Device into the system successfully.	
Status: Passed	

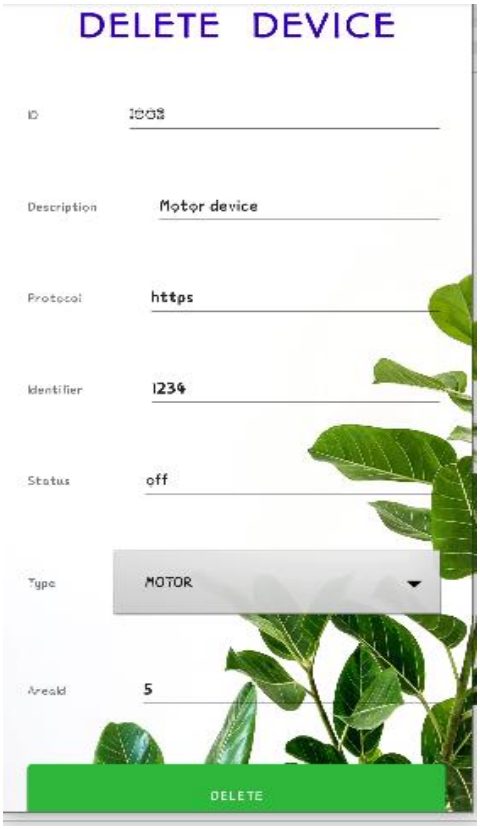

Figure 5. 16 Test 16 Delete Sensor

5.2.17. Test Case 17 View Service Plain:

Table 5. 17 Test 17 View Service Plain

Test Case: View Service Plain

Test Description: view Service Plain by selecting button Sensor card	**Test Case no :** 17
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can view list of Service Plain in list form.	**Test ID:** 017
Version: 1	**Test Type:** System testing
Scenario: User go to Home page. User select Service Plain button. User view all list of Service Plain.	
Test Data: Input: Press Service Plain button	
Expected Result: User should be able to view list of Service Plain successfully.	
Actual Result: User view list of Service Plain successfully	
Status: Passed	

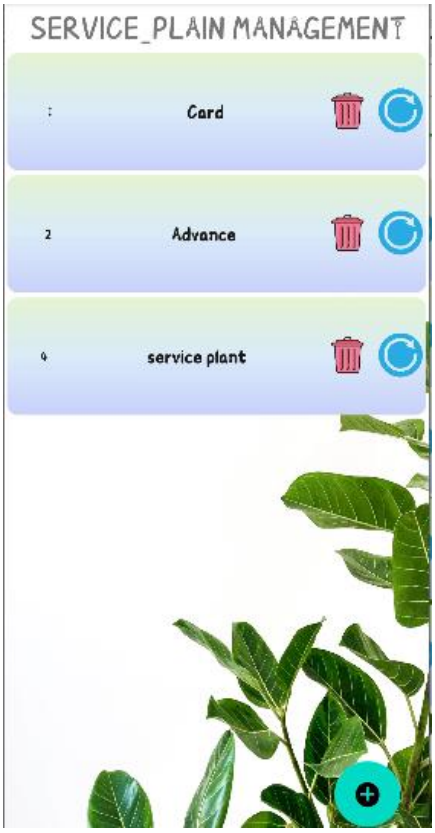

Figure 5. 17 Test 17 View Service Plain

5.2.18. Test Case 18 Add Service Plain:

Table 5. 18 Test 18 Add Service Plain

Test Case: Add Service Plain

Test Description: Add Service Plain by selecting button of Add.	**Test Case no :** 18
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can add Service Plain into the system successfully.	**Test ID:** 018
Version: 1	**Test Type:** System testing
Scenario: User go to Service Plain Home page. User select add button. User enter Service Plain name, description, type. User press add button. User add Service Plain successfully.	
Test Data: Name: image detection Description: check image disease by image processing. Type: Monthly	
Expected Result: User should add Service Plain into the system successfully.	
Actual Result: User add Service Plain into the system successfully.	
Status: Passed	

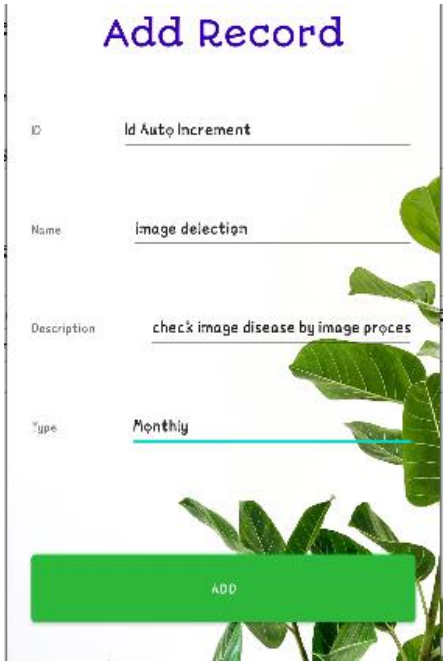

Figure 5. 18 Test 18 Add Service Plain

5.2.19. Test Case 19 Update Service Plain:

Table 5. 19 Test 19 Update Service Plain

Test Case: Update Service Plain

Test Description: Update Service Plain by selecting button of Update.	**Test Case no :** 19
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can update Service Plain into the system successfully.	**Test ID:** 019
Version: 1	**Test Type:** System testing
Scenario: User go to Service Plain Home page. User select selected Service Plain update icon. User update Service Plain by its type. User press update button. User update Service Plain successfully.	
Test Data: type: yearly	
Expected Result: User should update Service Plain into the system successfully.	
Actual Result: User update Service Plain into the system successfully.	
Status: Passed	

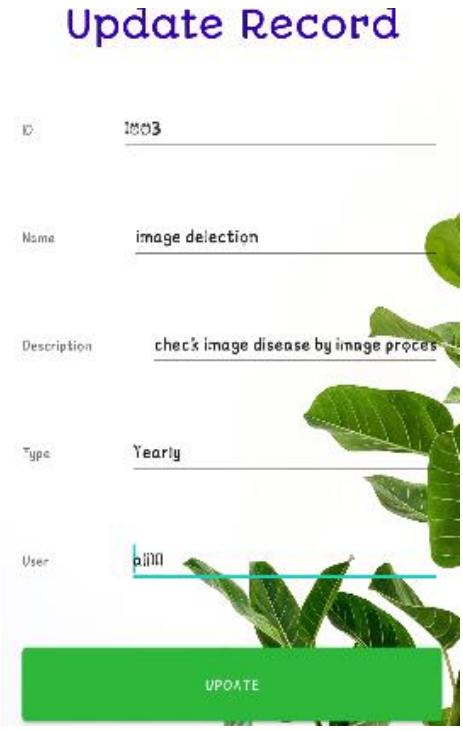

Figure 5. 19 Test 19 Update Service Plain

5.2.20. Test Case 20 Delete Service Plain:

Table 5. 20 Test 20 Delete Service Plain

Test Case: Delete Service Plain

Test Description: delete Service Plain by selecting button of delete.	**Test Case no :** 20
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can delete Service Plain into the system successfully.	**Test ID:** 020
Version: 1	**Test Type:** System testing
Scenario: User go to Service Plain Home page. User click selected Service Plain delete icon. User press delete button.	
Test Data: input: delete button	
Expected Result: User should delete Service Plain into the system successfully.	
Actual Result: User delete Service Plain into the system successfully.	
Status: Passed	

Figure 5. 20 Test 20 Delete Service Plain

5.2.21. Test Case 21 Control Area:

Table 5. 21 Test 21 Control Area

Test Case: Control Area

Test Description: Control area by user to change the status of devices or sensor.	**Test Case no :** 21
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can control the area.	**Test ID:** 021
Version: 1	**Test Type:** System testing
Scenario: User go to Control Area Home page. User select area in list. User change the status of selected sensor or device. User press update button	
Test Data: Input 1: select area 5 Input 2: select motor device for change off into on. Input 3: update button	
Expected Result: User should change area status into the system successfully.	
Actual Result: User change area status into the system successfully.	
Status: Passed	

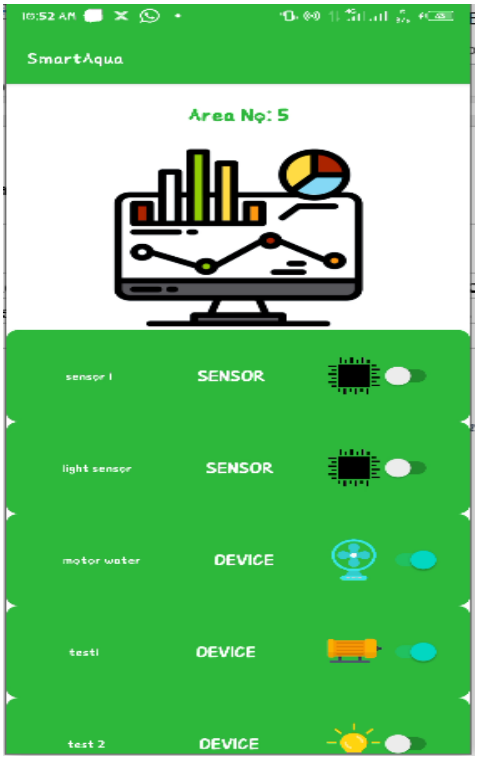

Figure 5. 21 Test 20 Control Area

5.2.22. Test Case 22 Get Notification:

Table 5. 22 Test 22 Get Notification

Test Case: Get Notification

Test Description: User Should be get notification after status of device are change.	**Test Case no :** 22
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can get the status notification.	**Test ID:** 022
Version: 1	**Test Type:** System testing
Scenario: User go to control area button. User select area in list. User change the status of selected sensor or device. User press update button	
Test Data: Input 1: select area 5 Input 2: select motor device for change off into on.	
Expected Result: User should get notification of change status into the system successfully.	
Actual Result: User get notification of change status into the system successfully.	
Status: Passed	

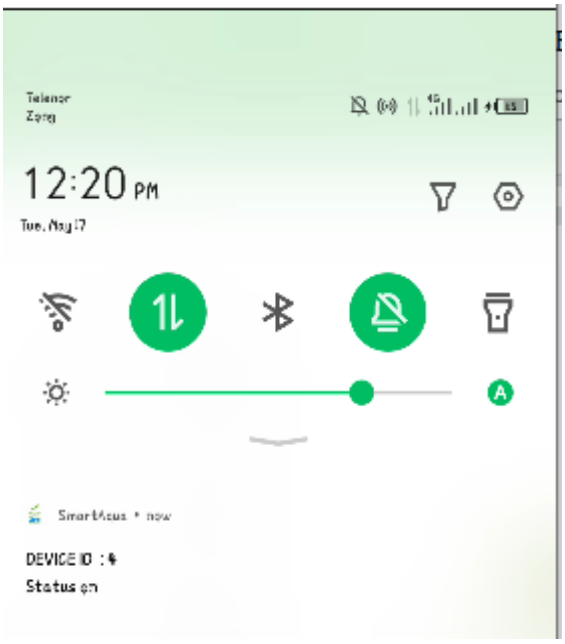

Figure 5. 22 Test 22 Get Notification

5.2.23. Test Case 23 Smart Automation:

Table 5. 23 Test 23 Smart Automation

Test Case: Smart Automation

Test Description: User Should be set automate system using popup.	**Test Case no :** 23
Date: 01 Aug 2022	
System: Smart AquaPonic	
Objective: User can change smart automation.	**Test ID:** 023
Version: 1	**Test Type:** System testing
Scenario: User go to home page. User select automation card. User change the automation status using popup form. User press set automate button.	
Test Data: Input 1: press automation card. Input 2: press switch automation. Input 3: press set automation button.	
Expected Result: User should set status of automation into the system successfully.	
Actual Result: User set automation change status into the system successfully.	
Status: Passed	

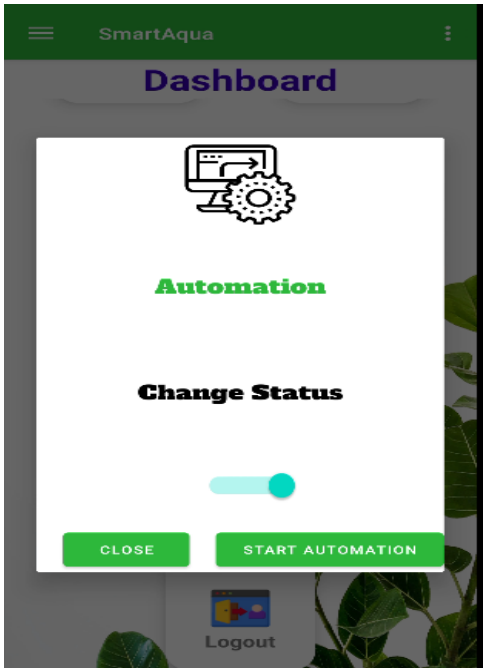

Figure 5. 23 Test 23 Set Automation

Chapter 6
Software Deployment

6.1. Deployment Process Description

In this section we provide some information about how to install mobile app or deploy ASP . NET Web Framework system.

ASP.NET Web Deploy IIS Server

Step 1: Press Window button and write control panel

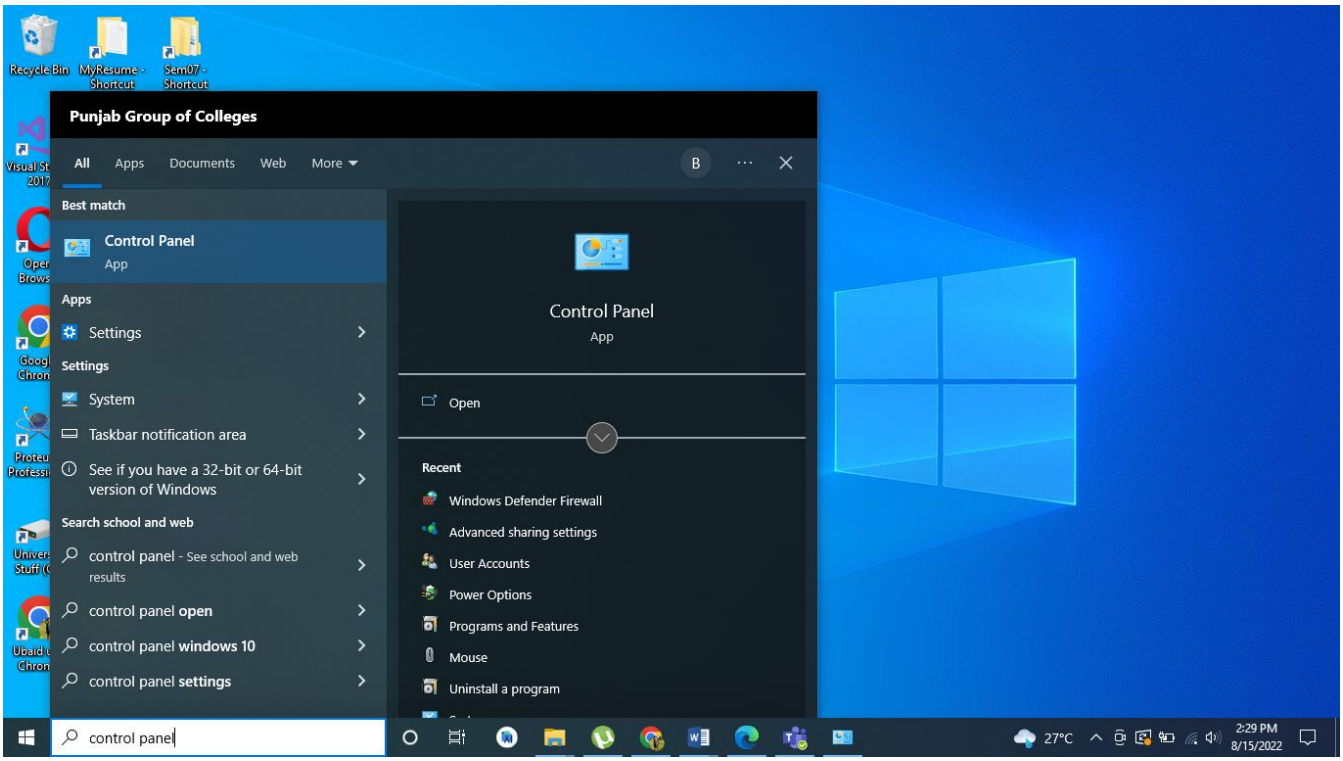

Figure 6. 1 control panel

Step 2: Open Control Panel and click window defender firewall

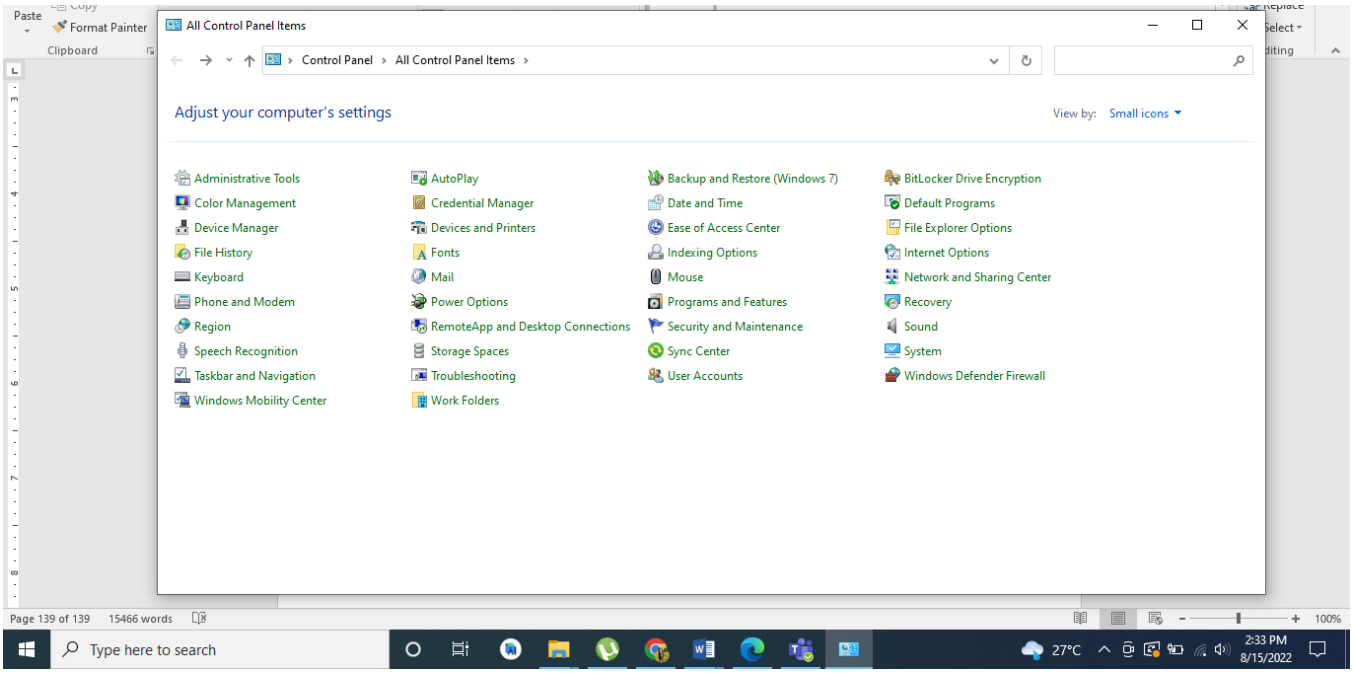

Figure 6. 2 control panel 2

Step 3: Click window defender firewall and turn window defender off if it is on.

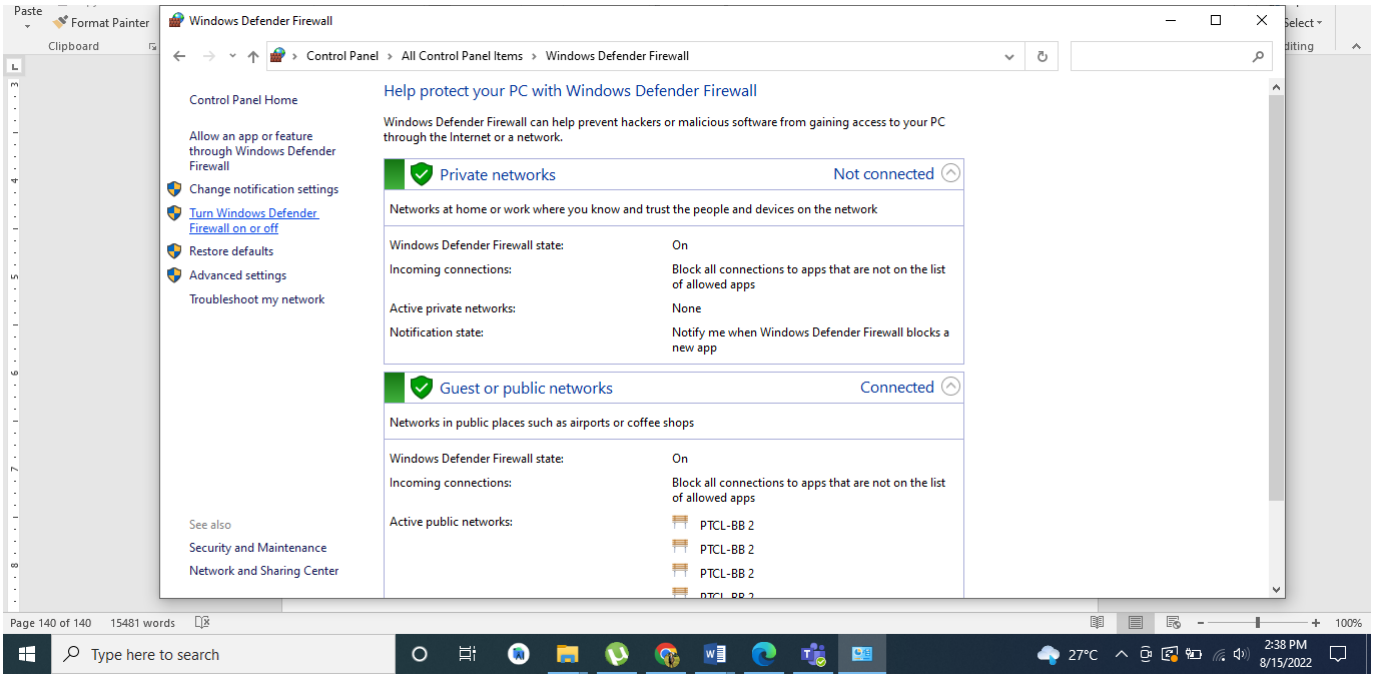

Figure 6. 3 control panel 3

Step 4: Click on turn of window defender and press ok

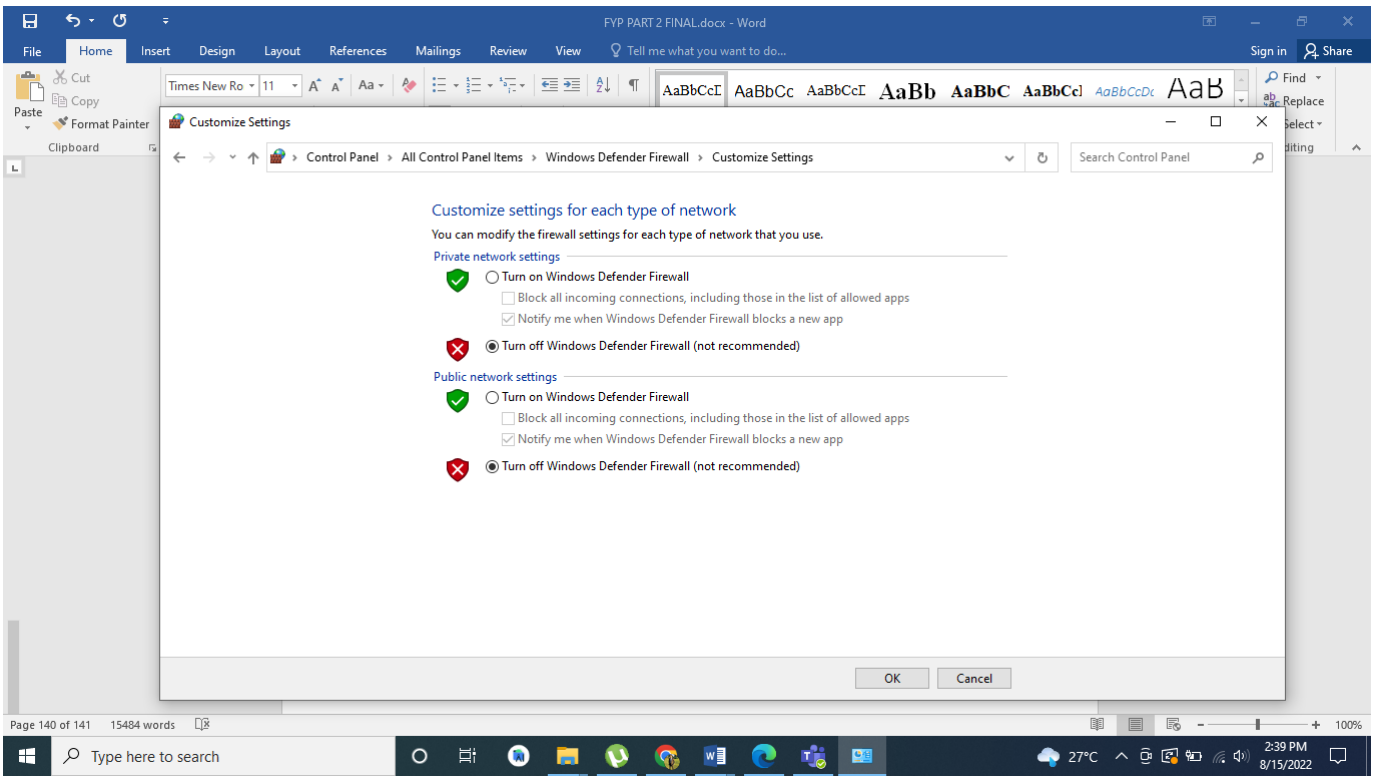

Figure 6. 4 control panel 4

Step 5: Now search and learn on google that how to Enabling IIS and required IIS components on Windows 10 then move on step 6

Step 6: Press window and enter internet information services (IIS) Manager for open manager dashboard.

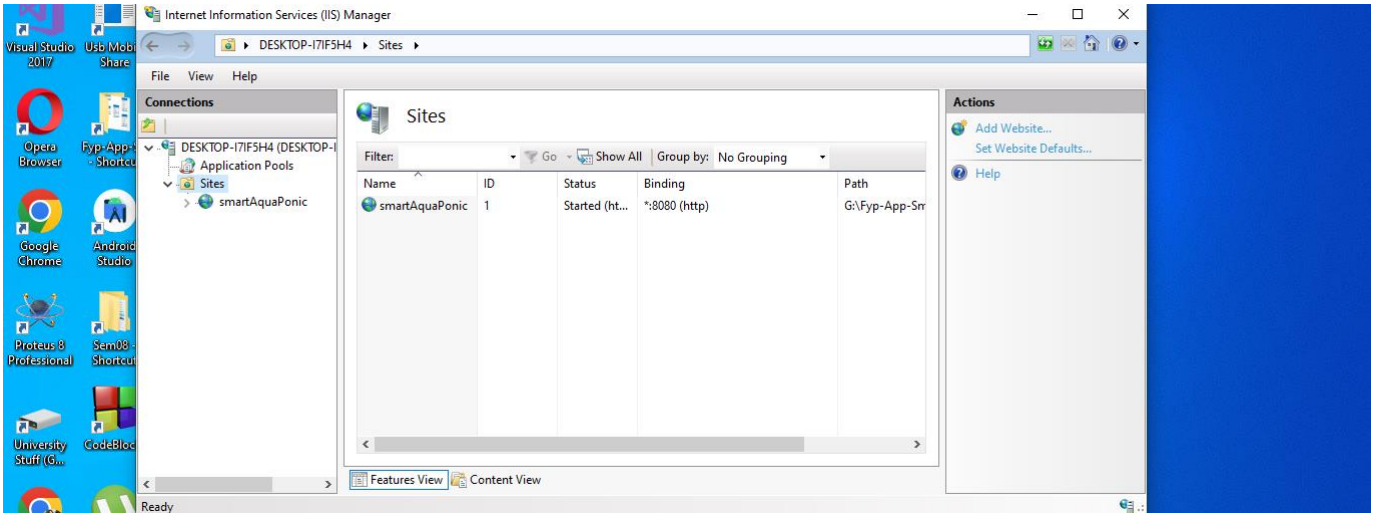

Figure 6. 5 control panel 5

Step 7: Right click on sites and press add website.

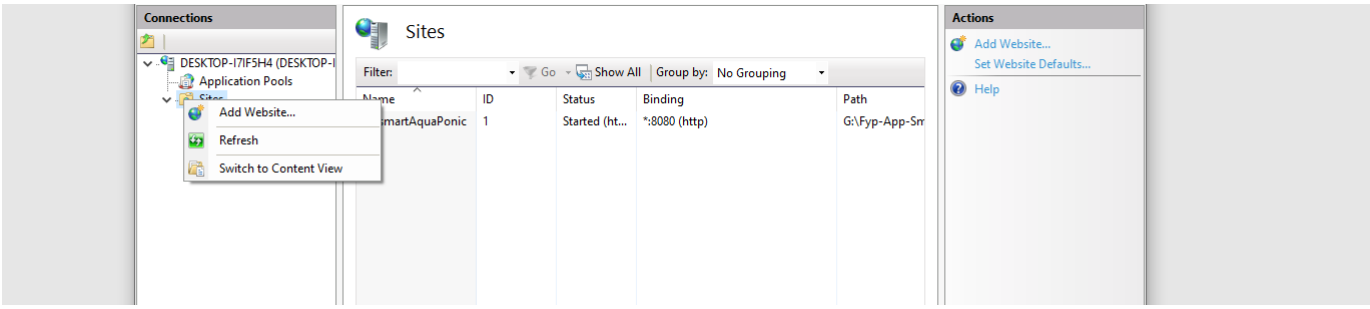

Figure 6. 6 control panel 6

Step 8: Now enter site name, path where exist and enter port number then press ok.

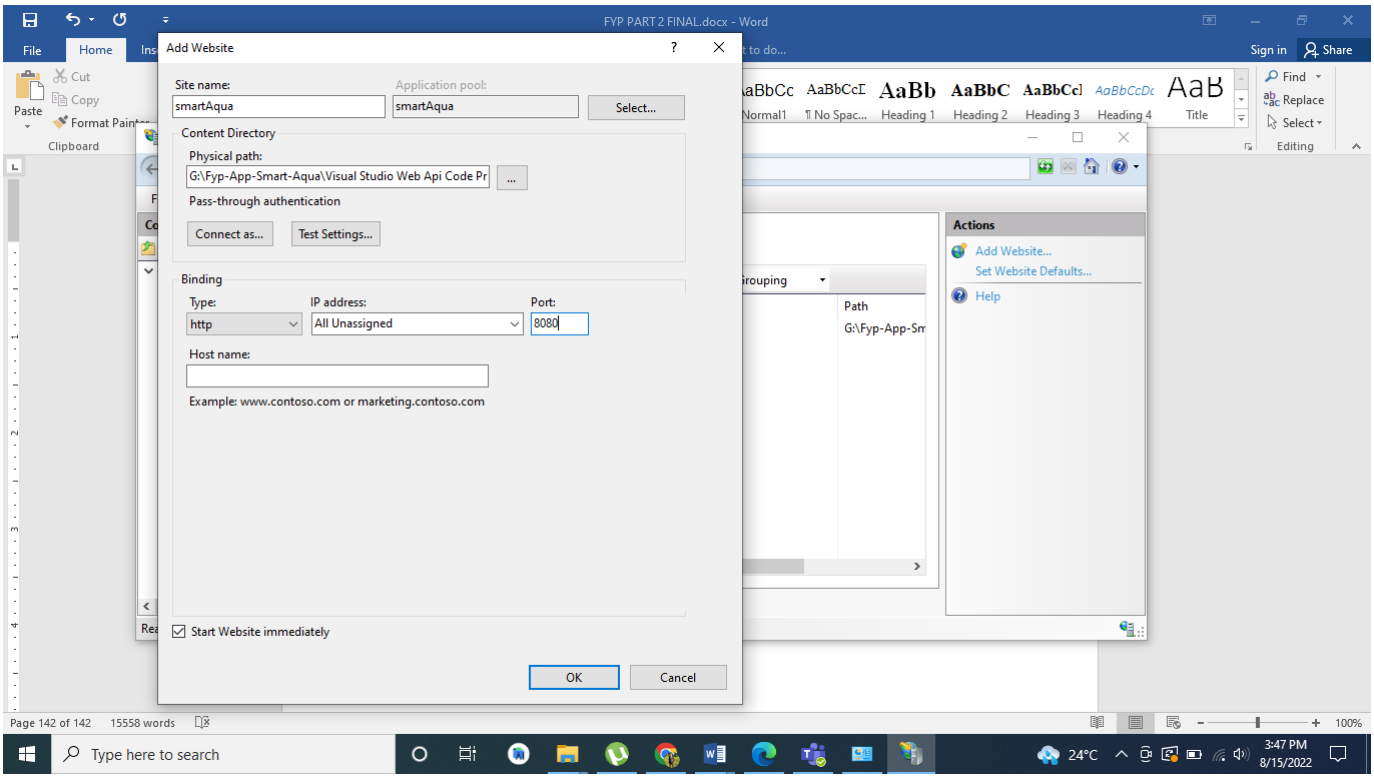

Figure 6. 7 control panel 7

Step 9: Now one last step go to application tool then right click of your site name click on advanced setting after this then change identity into a local system for proper work.

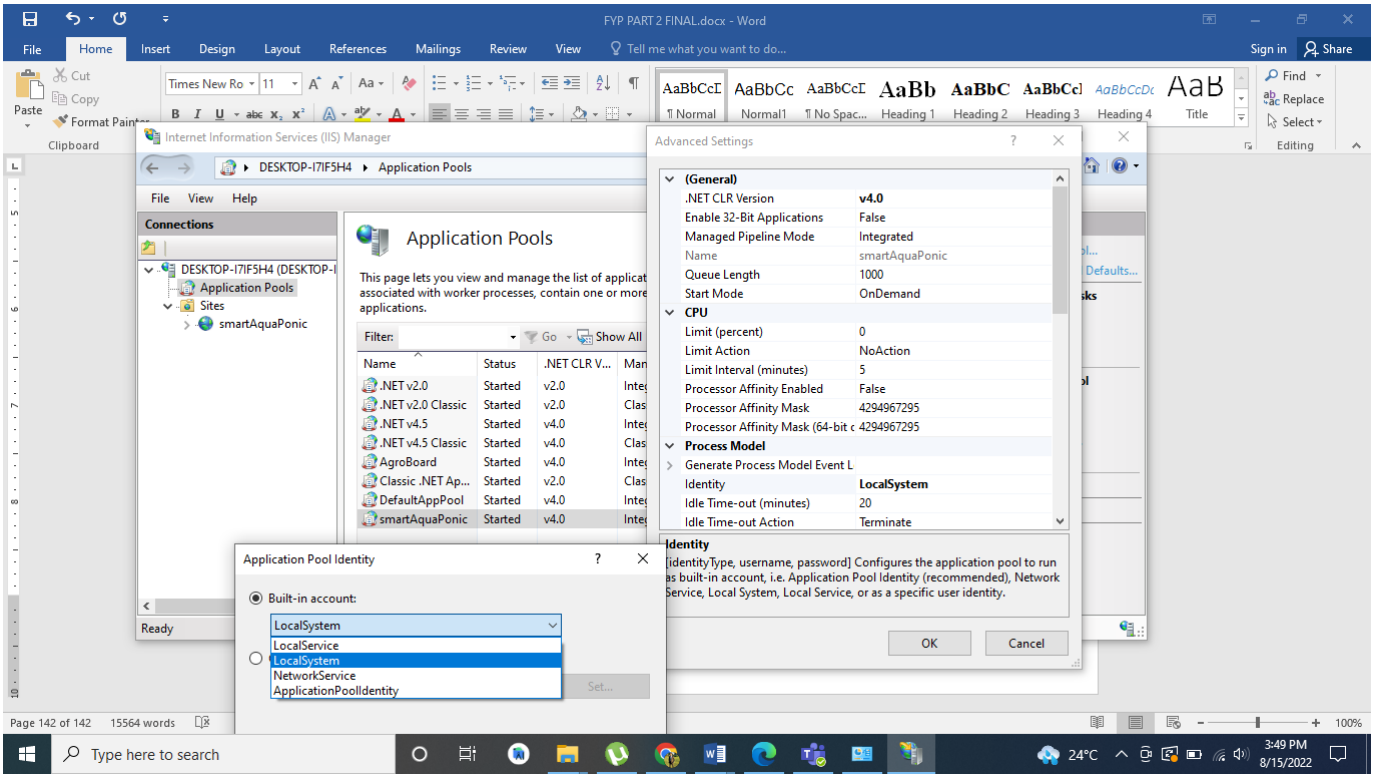

Figure 6. 8 control panel 8

After the completion of these step now simply install android app from google play store and use these services.

REPORT APPROVAL CERTIFICATE

The report of the project, “Title” has been approved based on the following evaluation guideline.

Project Evaluation Guidelines

Artifacts Guidelines	
Analysis and Design artifacts are syntactically correct (use-case model, SSDs, domain model, class diagram, SDs, ERDs, Flow charts, Activity Diagram, DFDs)	
Consistency and traceability have been maintained among different artifacts	
General Guidelines	
Formatting (font style, indentation) is according to the FYP template and consistent throughout the document	
Captions are added to all the figures and tables. Figure captions must be placed below each figure, and table captions must be provided above the table	
Each figure or table is followed by some text describing what it represents	

Name & Signature (Examiner 1)	Name & Signature (Examiner 2)	Name & Signature (Examiner3)

Name & Signature
(Supervisor)

References

1. G. Morabito L. Atzori (Eds.) D. Giusto, A. Iera. "The Internet of Things." *Springer*, 2010.

2. S. Marusica M. Palaniswamia J. Gubbia, R. Buyyab. Internet of things (iot). "A vision, architectural elements, and future directions." *Future Generation Computer*, 2013.

3. Sauser, J. Boardman and B. *System of systems*. 2006.