



TED UNIVERSITY

Computer Engineering Department

**CMPE 491 Senior Project – Project Analysis Report -
AgroAutomaTED**

by

Korhan Deniz Akın

Ahmet Can Öztürk

Kaan BUDAK

26/11/2023

● Table of Contents

| | |
|---|----|
| Table of Contents | 2 |
| 1. Introduction | 3 |
| 2. Current system | 4 |
| 3. Proposed system | 5 |
| 3.1 Overview | 5 |
| 3.2 Functional Requirements | 5 |
| 3.3 Non-functional Requirements | 8 |
| 3.4 Pseudo requirements | 11 |
| 3.5 System models | 12 |
| 3.5.1 Scenarios | 12 |
| 3.5.2 Use case model | 14 |
| 3.5.3 Object and class models | 15 |
| 3.5.4 Dynamic models | 16 |
| 3.5.5 User interface - navigational paths and screen mock-ups | 17 |
| 4. Glossary | 21 |
| 5. References | 21 |

1. Introduction

The main purpose of the project is to find solutions for better use of water in a mobile app and a physical device that will reduce insufficient use of irrigation water and inefficient soil moisture management issues. We decided to create an Arduino device that has several features such as a soil moisture level sensor, a water level sensor for the water tank, and a canal to establish efficient irrigation of the soil. This device shall provide the necessary pieces of information about the soil and send them to the mobile app. The mobile app will send a notification to the user, for example, when the water tank level decreases and reaches its limit. This app shall also provide some automation for better irrigation with the help of artificial intelligence depending on plant, soil, and environmental conditions. With this approach, we aim to contribute to the overarching goal of zero hunger as outlined in the United Nations' Sustainable Development Goals (SDGs).

This application has been designed for various target audiences for the development and use of smart irrigation system applications:

1. Developers:

Detailed specifications, architecture, and coding guidelines are provided for developers involved in the implementation of the Smart Irrigation System application. Particular attention will be paid to sections on sensor integration wireless communication and application functions.

2. Project Managers:

Project managers will find information regarding project scope, timelines, and resource allocation. The project overview, requirements, and timeline sections are critical for project planning and management.

3. Users:

Users should start with the sections that will guide them in using the application. These steps include creating an account, logging in, and controlling the smart irrigation system. Particularly relevant are the sections on monitoring water levels, receiving notifications, and understanding soil conditions.

4. Testers:

Testers will find test cases, scenarios, and expected results outlined in the testing section. This includes functionality testing for user interactions, sensor accuracy, and system responsiveness aiming to achieve more accurate results.

The purpose of the product is to obtain better irrigation in terms of using the water efficiently to promote sustainable agriculture. With the scope of this project, we intend to contribute to the “Zero Hunger” and “Water and Sanitation” goals of “United Nations Sustainable Development the 17 Goals“. Especially “Target 6.4 of Clean Water and Sanitation” which mentions the importance of water-use efficiency and for the Zero Hunger goal we decided to follow “Target 2.4” which mentions agricultural productivity and production.

Additionally, this product shall help the daily users who want to irrigate their plants at their house.

2. Current system

Sri Satyasai Vatsavayi et al. (2021) proposed an irrigation system for their Bachelor Thesis in June 2021 called “ Home Irrigation System Using Internet Control”. They used an Arduino UNO I/O shield, Arduino UNO Board, humidity and temperature sensors, PIR sensor (detection of the motion), soil moisture sensor, pump for providing water to the soil, and a power supply. For the software implementation, they used Arduino IDE. "WiFi NINA" library to connect to the wifi to send data to the web browser. With these implementations, they calculate the information from the soil with sensors, provide irrigation if necessary, detect the motion, and send these data to Arduino IDE’s “Serial Monitor” which is an output console for the Arduino UNO board and to the web server. Even though this project is developed solidly in terms of hardware and might be a good source of information for our further development in our senior project, it does not provide a satisfying user interface to control the device manually or advantageously process that information. Additionally, they limited themselves to a small scope that this system is only used for daily users who are willing to irrigate their plants at home. Meaningly, all the hardware requires serial communication with the PC all the time which is not desirable for us.

An example of another existing system is the "Arduino Based Automatic Irrigation System: Monitoring and SMS Controlling" system by Yasin et.al. from the 4th Scientific International Conference – Najaf – IRAQ (4th SICN-2019). The system was developed using GSM technology and uses Arduino mega 2560. The purpose of the system is to both monitor and manage the user's irrigation system via phone messages. The working principle of the system measures soil moisture through sensors and sends an irrigation request to the user via SMS message if the soil is dry. To irrigate the soil, the farmer starts the process by sending a message to the system again. Thus, the Arduino platform allows receiving/sending SMS to/from the mobile phones of farms/homeowners based on the land needed for water or the instructions sent by the user. However, one of the shortcomings of this system is that the user can only be informed about the water level, soil moisture, etc. at a basic level. For instance, if the water in the system is low or has run out, the user is only sent the message that the water is low, but in our system, there will be details such as how much water level remains and how much irrigation can be done with this amount of water. We aim to send detailed data to the user on our site, enabling him to perform more comprehensive operations.

None of these projects are using the collected and processed information adequately. They only receive from the sensors and start the irrigation. For us this is insufficient. Therefore, we plan to build an AI model to increase the performance of the system related to the environment and soil conditions. We believe this model adds significant value to our project compared with other developments.

3. Proposed system

3.1 Overview

The AgroAutomated project aims to address the challenges associated with agricultural irrigation by leveraging a comprehensive system that integrates a mobile application and a purpose-built Arduino device equipped with advanced sensors. At the core of our project is our commitment to optimizing water usage and rectifying issues related to inefficient soil moisture management. To achieve this, we will set up an Arduino device featuring key components such as a soil moisture level sensor, a water level sensor for precise monitoring of the water tank, and a canal for ensuring the effective irrigation of soil.

From the user's perspective, the first step is to download the application and then create an account. The necessary information about the field and plants should be conveyed to the system through the mobile app. After completing this stage, communication is established with the user to reach an agreement and create a shared plan for effective implementation. Once the setup is in place, the user can receive necessary notifications from the application, such as a decrease in the water tank level, system malfunctions, and soil moisture status. The user becomes easily accessible for information about their field, and the drip irrigation system is operational.

Finally, our purposeful approach strives not only to address the inefficient use of irrigation water but also to actively contribute to the overarching goal of achieving zero hunger, aligning with the United Nations' Sustainable Development Goals (SDGs). By seamlessly integrating technology and sustainable agriculture practices, AgroAutomated represents a groundbreaking solution in the pursuit of a more resource-efficient and sustainable agricultural future.

3.2 Functional Requirements

3.2.1 Definition:

- The project aims to create a mobile application and a physical device to find solutions for better water use. This device is a system managed by an Arduino device with features such as a soil moisture level sensor, a water tank water level sensor, and a small irrigation engine for efficient irrigation. The mobile application sends information about the soil and water status to the user. The user can perform all operations automatically or can perform these operations one by one by giving commands through the mobile application. It also provides automation for better irrigation based on plant, soil, and environmental conditions using artificial intelligence.

3.2.2 General Requirements:

3.2.2.1 Hardware:

Arduino device, soil moisture level sensor, water tank, water level sensor, small irrigation motor, Grove LCD screen for control operations, distance detection sensor.

3.2.2.2 Software:

Mobile application, database management, artificial intelligence model.

3.2.3 Functionalities:

- The user registers to the system by creating an account.
- When a user record is created, a new record is added to the database.
- User login to the application.
- The user defines the system to the application depending on the features he chooses.
- There will be a sensor in the water tank that shows the water level.
- The device will use sensors to show soil quality and moisture.
- The user will receive notifications from the mobile application according to the current situation.
- The device will irrigate by taking into account the moisture of the soil using a water tank and a small irrigation engine (This process can also be done manually by the user).
- The device sends information via the internet via a wireless connection.
- It will inform the user about the soil condition and water level through the system.
- In terms of parameters collected from the hardware device, an artificial intelligence model will produce predictions available in the mobile application. These predictions will provide the user with the most appropriate information about the general situation. This feature can be turned on and off by the user if desired.

3.2.4 User Scenarios and Workflows:

The user logs in via the mobile application. After logging in, he sees notifications about the products and water tank or system. The user receives notifications when the water tank level drops or there is any problem with the plants. The user instantly monitors the soil moisture and quality status via the application. With user permission, automatic irrigation takes place depending on the soil moisture level and also according to optimal recommendations with the support of artificial intelligence.

3.2.5 Performance Requirements:

- Once the user logs in, this process should be completed in less than 3 seconds.
- The water tank level sensor should update every hour and the soil sensors should update every 10 seconds.
- Notifications sent to the user must be delivered within 5 seconds at most after the event occurs.

3.2.6 Limitations:

a. Browser Support:

The system supports modern browsers such as Google Chrome, and Mozilla Firefox.

Browsers that are outdated or have security vulnerabilities, such as Internet Explorer, should not be supported.

b. Operating System Support:

The mobile application supports iOS and Android operating systems.

No special effort is made to guarantee proper operation on older operating systems.

3.2.7 Security Requirements:

a. Password Security:

User passwords must be at least 8 characters long and contain uppercase letters, lowercase letters, numbers, and special characters.

b. Data Transmission:

Critical information, especially user passwords, should not be transmitted in clear text.

c. Session Security:

The login process should not fail more than 5 times. When this limit is exceeded, the user is expected to enter the security code to verify that he is human. If someone wants to enter the system without permission, the user will be asked for a verification code due to incorrect entry. If entered incorrectly, the system will be locked for a certain period.

Sessions should be terminated automatically if inactive for a certain period.

3.3 **Non-functional Requirements**

To define non-functional requirements in a comprehensive way both hardware and software must be addressed. Therefore, it is decided to separate their sections and evaluate

them in that manner so, that the next two sections cover non-functional requirements of hardware and then the software by inspired Altexsoft Blog(Altexsoft, 2023).

3.3.1 HARDWARE

3.3.1.1 Usability:

Using hardware and software products is fairly easy but the setup of the hardware can be confusing in the first place. Firstly put the soil moisture sensor in the soil, leaving other sensors uncovered to not block them, fill the water tank, and put the water pump in a convenient spot where the soil is to be irrigated. After that provide the WIFI name and the password to the device to make it available for usage. Then log in to the application to get the most out of the product.

3.3.1.2 Efficiency:

3.3.1.2.1 Space:

Arduino UNO Board uses ATmega328P microcontroller which consists of non-volatile and volatile memory segments. Non-volatile storage stands for the memory segments that can hold the information even if the power is removed (Department of Atmospheric Sciences, 2023). It consists of 32KB of Program ROM (also called flash memory since it runs fast) which is a storage component in the compiled code is being stored and EEPROM (Electrically Erasable Programmable ROM) is only 1KB of storage and 2KB of internal SRAM (Elliot, 2014). Some limitations are worth mentioning about these storage components.

First of all, even though EEPROM is an important component to store permanent data such as encryption key strings writing and erasing operations consists of lots of time which is 1.8 ms for a write or 3.4 ms for an erase and additionally, such EEPROMs have limited number of erase and write cycles (Elliot, 2014). This is specifically 100.000 write and erase cycles. After this limit of a cycle is reached the EEPROM becomes unreliable so, if the usage of this memory component is needed then it should not be frequent.

Secondly, ATmega328P includes 32 KB of the flash program memory space. Because of this limited program memory space, the compiled code that is to be used should be smaller than any Java program that consists of lots of large data types (Williams, 2014). The program memory space is so tiny that even using too many integers might lead to some capacity issues. Also, we have a limited amount of RAM as well. The ATmega328P chip has 2KB of RAM.

3.3.1.2.2 Performance:

Even though there are some limitations in terms of memory space of the ATmega328P microcontroller, it has a sufficient CPU. It cannot be compared with the modern PC but enough to perform most of our desired functionality with up to 20MHz CPU speed (Atmel, 2016).

In terms of memory components, flash memory is not the fastest component among storage devices it provides a sufficient amount of speed since it consists of compiled code. On the other hand, EEPROM might be the bottleneck of these storage components since it requires, 1.8 ms for a write or 3.4 ms for an erase operation (Williams, 2014). However, this should not be an observable issue.

3.3.1.3 Security

There shall be tons of data transferring during serial communication with sensors, sending data over the internet, and user authentications. The microcontroller chip which is embedded in the Arduino UNO Board consists of a tiny storage that even using overmuch integer types might be inefficient. Therefore, providing the security of the transferred data might be compelling in terms of storage limitations. On the other hand, securing user-related information is a concern that none of the information is leaked to 3rd parties.

3.3.1.4 Scalability

The hardware system shall collect data from the soil and the environment with various types of sensors. These sensors serve as the backbone of the hardware system since most of its desired functionality relies on them. Therefore, it shall manage all of them concurrently.

The hardware system shall send all the related data via a WIFI connection to the server. This has to be achieved optimally since the environmental conditions tend to change very often. For example, weather conditions, soil moisture, and atmosphere pressure are not some data that arise from whole numbers; instead, they are a bunch of floats with too many decimals. Therefore, sending this information every time they change is not a feasible approach so, it sends the data every 10 minutes to the server or when the user requests them.

In terms of battery, the system shall be energy-conservative and does not require to be recharged very often. It shall last up to 1 week.

3.3.1.5 Reliability

In terms of serial communication (communication with sensors and the microcontroller), the system might face errors. These errors are not related to the implementation but to the communication protocols themselves. For instance, I2C is a serial communication protocol that enables data transfer between the microcontroller and sensors within a short distance (Mazidi et al., 2011).

This data transfer might not be reliable if it exists in a noisy environment. On the other hand, the system shall continuously try to receive data if an error occurs.

Another thing to mention is that it shall provide useful messages to the user about the condition of the whole system. For example, if the battery is decreased and must be recharged or the water tank must be refilled the hardware system sends a message to the user.

3.3.1.6 Maintainability

As the Arduino device is not complicated and required connections can be made concerning provided documents any person can restore the hardware system in case of system failure.

3.3.1.7 Availability

The hardware system is available to the user %100 of the time if the battery power is enough.

3.3.1.8 Compatibility

As Arduino does not consist of any OS it does not require users to fulfill OS versions. However, the user must have the application provided by us to use the device.

3.3.2 SOFTWARE

3.3.2.1 Usability:

With the simple interface of the application, users can navigate between different tabs smoothly. This app is not complicated for any mobile phone user. There is a bar at the bottom that can be used to obtain different functionalities of the system.

3.3.2.2 Efficiency:

Users shall receive information from the hardware devices in a few seconds if it is desired. Communication protocols are really fast. Also, the machine learning model produces output in seconds.

3.3.2.3 Security

Such protocols like HTTPS are used to protect the user information. Also, all of the data on the server shall be encrypted.

3.3.2.4 Scalability

The application and the database are scalable to store and respond to transferred data every 10 minutes or when the user requests. Meaningfully database server is capable of storing data that comes every 10 minutes or when the user enters the application.

3.3.2.5 Reliability

The mobile application shall fulfill its purpose %98 of the time when there are no such errors in terms of hardware level.

3.3.2.6 Maintainability

In case of a failure of the application or the database system, it cannot take longer than a few hours to restore.

3.3.2.7 Availability

Similar to the hardware system, the application is up and running %100 percent of the time. Even if there are no problems with the hardware system, such as a lack of battery power, the system is available all the time to the user.

3.3.2.8 Compatibility

Flutter is a cross-platform development environment, devices that use IOS, Android operating systems, or web browsers can run the application. Concerning the “Flutter Documentation” a Flutter app can run on IOS version 11 and later, on Android devices with version 19 or higher, and from popular web browsers Chrome version 96 or higher, Firefox 99 or higher, Safari 14 or higher and Edge 96 or higher (Flutter Official Website).

3.4 **Pseudo requirements**

3.4.1 Implementation Pseudo Requirements

- The dart language will be used to develop a Flutter project for cross-platform development.
- The project implementation will follow the MVCS (Model View Controller Service) architecture for enhanced system organization and functionality.
- The C language will be used on Arduino.
- The mobile app will be available on both iOS and Android.
- During the application development, the utilization of packages from pub.dev will align with the specific requirements of our Flutter project.
- GitHub will be used to facilitate collaborative development and version control for projects.
- The progress of our project can be followed on our website.

3.4.2 Economic Pseudo Requirements

- Essential components for the Arduino device, including sensors (humidity, temperature, water, etc.) will be purchased.

3.4.3 Sustainability Pseudo Requirements

- Protective measures for the Arduino system will be implemented, encompassing the use of suitable enclosures, surge protectors, and temperature control mechanisms.
- To enhance the sustainability of the Arduino device, efforts will be directed towards implementing improvements in meeting its energy and electricity requirements more efficiently.

3.4.4 Ethical Pseudo Requirements

- All applications of the project will adhere to the Code of Ethics established by the National Society of Professional Engineers (NSPE, 2019).
- The project will give priority to environmental responsibility, aiming to minimize its ecological footprint and adhere to sustainable practices.

3.5 System models

3.5.1 Scenarios

Scenario 1: Initial Setup and Configuration

Objective: The user sets up the Arduino device and mobile app for the first time.

User Actions:

Assembles the Arduino device with the soil moisture sensor, water level sensor, and irrigation canal.

Installs the mobile app on their smartphone.

System Responses:

The Arduino device initializes and calibrates the sensors.

The mobile app prompts the user to enter initial settings, such as plant type and environmental conditions.

Scenario 2: Real-time Monitoring

Objective: The system continuously monitors soil and water conditions in real-time.

User Actions:

Opens the mobile app to check the status.

System Responses:

The mobile app displays real-time data on soil moisture levels, water tank levels, and other relevant information.

If the water tank level is low, the app sends a notification to the user.

Scenario 3: Automated Irrigation

Objective: The system utilizes artificial intelligence for automated irrigation based on sensor data.

User Actions:

Enables the automated irrigation feature in the mobile app.

System Responses:

The system collects and analyzes data from soil moisture and environmental sensors.
Based on the AI algorithm, the system determines the optimal irrigation schedule and duration.
The irrigation canal is activated as per the calculated schedule.

Scenario 4: Low Water Tank Alert

Objective: Notifies the user when the water tank level is critically low.

User Actions:

Continues with daily activities.

System Responses:

The water level sensor detects a critically low water level.
The mobile app sends an immediate notification to the user, advising them to refill the water tank.

Scenario 5: Manual Override

Objective: Allows the user to manually control the irrigation process.

User Actions:

Accesses the mobile app and selects manual control.

System Responses:

The user can manually activate or deactivate the irrigation canal through the mobile app.
Manual adjustments do not interfere with the automated irrigation schedule.

Scenario 6: System Calibration

Objective: Ensures accurate sensor readings for optimal performance.

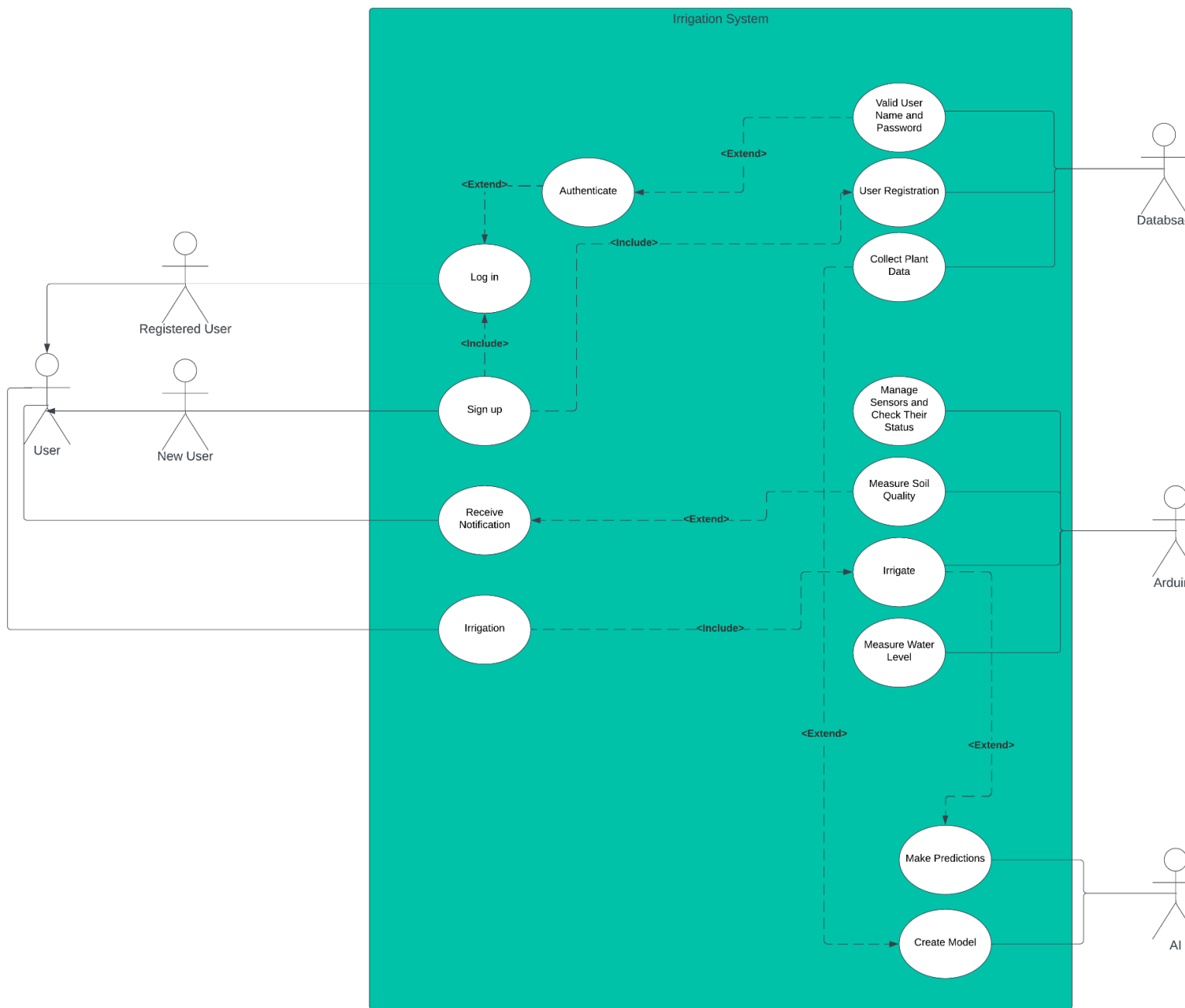
User Actions:

Notices inconsistent readings or system behavior.

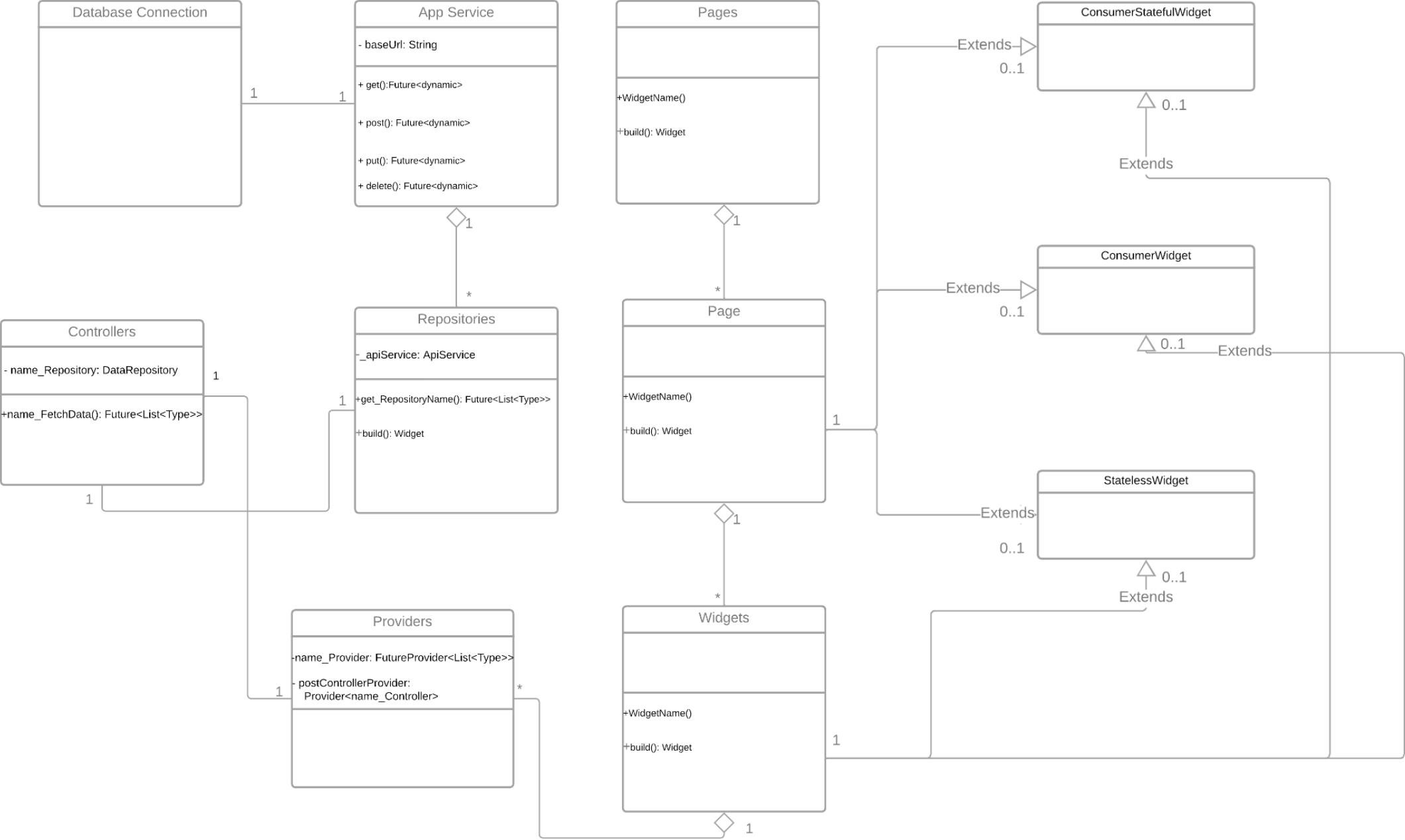
System Responses:

The mobile app guides the user through a calibration process for the soil moisture and water level sensors.
The system adjusts sensor parameters for improved accuracy.

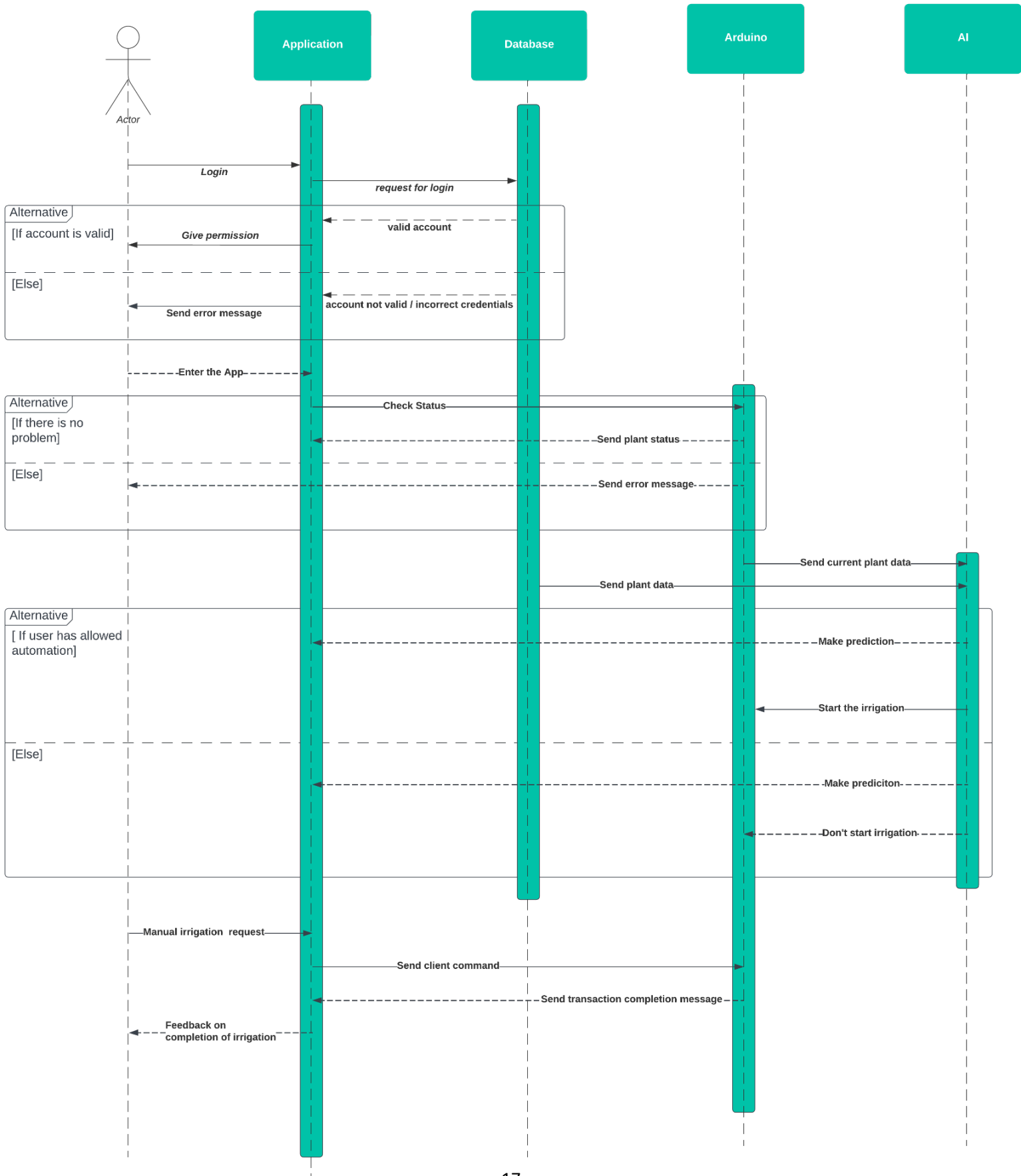
3.5.2 Use case model



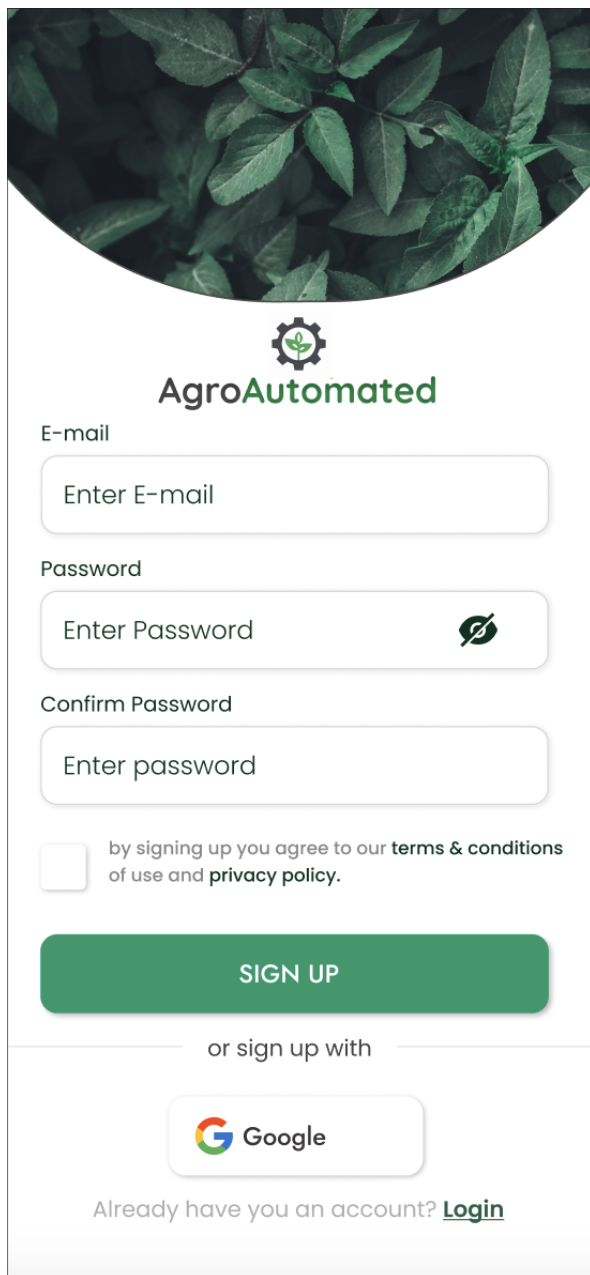
3.5.3 Object and class models



3.5.4 Dynamic models



3.5.5 User interface - navigational paths and screen mock-ups



The mock-up shows a sign-up screen for the AgroAutomated app. At the top is a circular image of green leaves. Below it is the AgroAutomated logo, which consists of a gear icon with a plant inside, followed by the text "AgroAutomated". The form includes fields for "E-mail" and "Password", each with a placeholder text "Enter E-mail" and "Enter Password" respectively. There is a "Confirm Password" field with a placeholder "Enter password". A checkbox is present with the text "by signing up you agree to our terms & conditions of use and privacy policy." Below the checkbox is a green "SIGN UP" button. At the bottom, there is a link "or sign up with" followed by a Google logo and the text "Google". At the very bottom, there is a link "Already have you an account? Login".

E-mail

Enter E-mail

Password

Enter Password


Confirm Password

Enter password

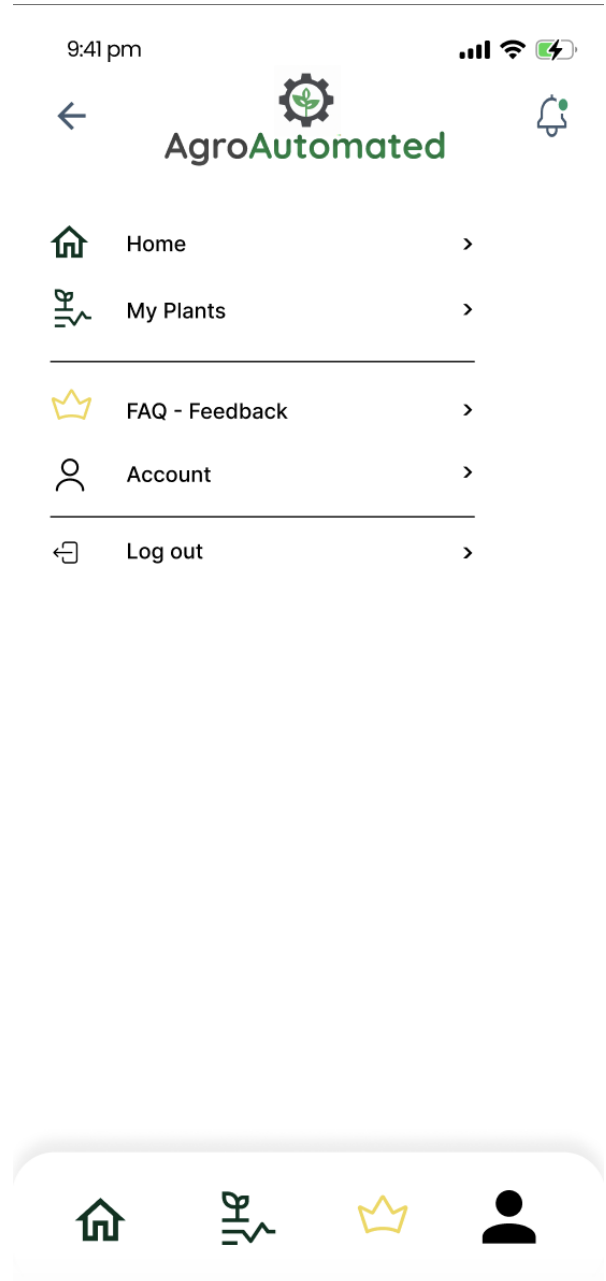
☐ by signing up you agree to our [terms & conditions](#) of use and [privacy policy](#).

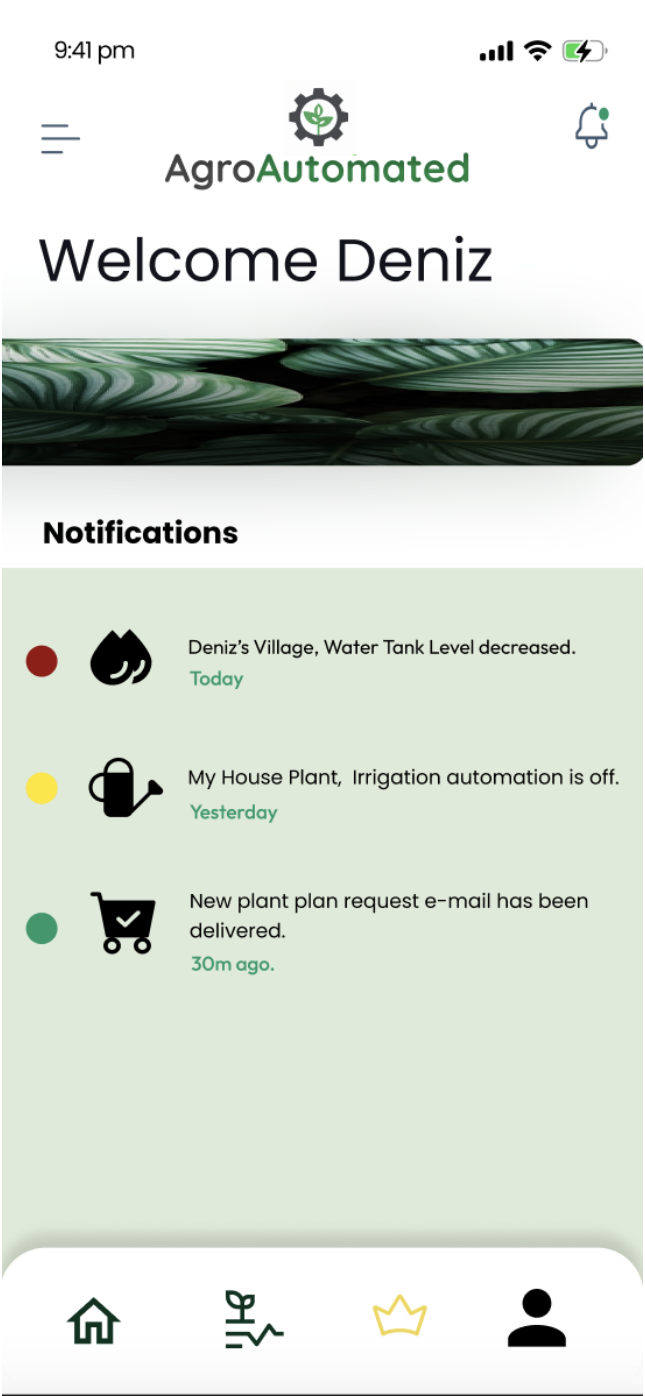
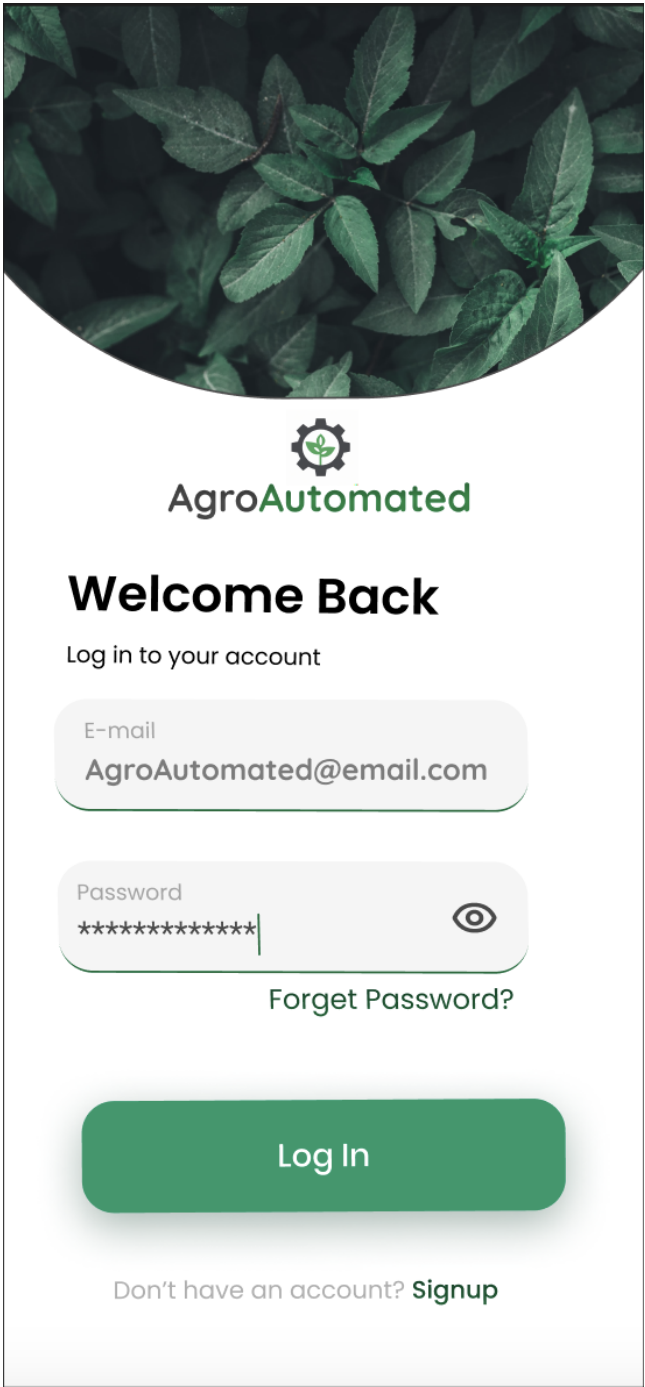
SIGN UP

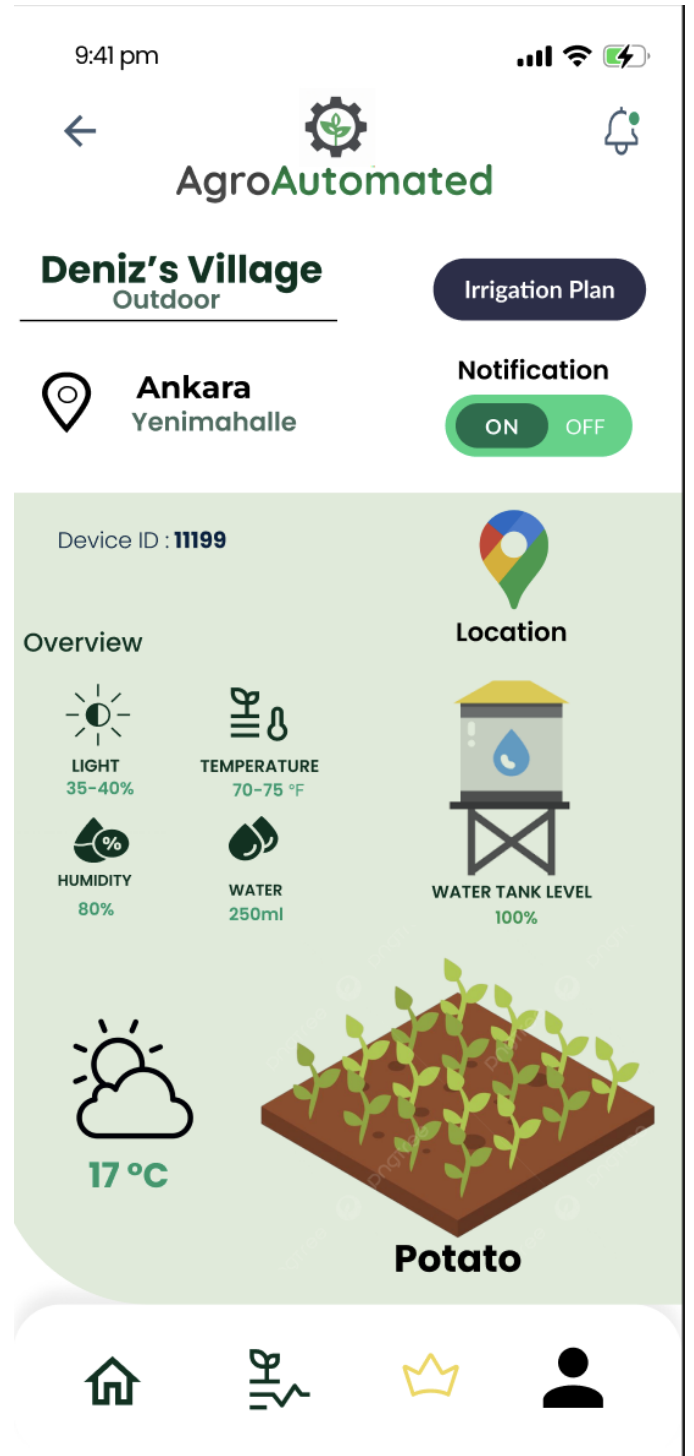
or sign up with

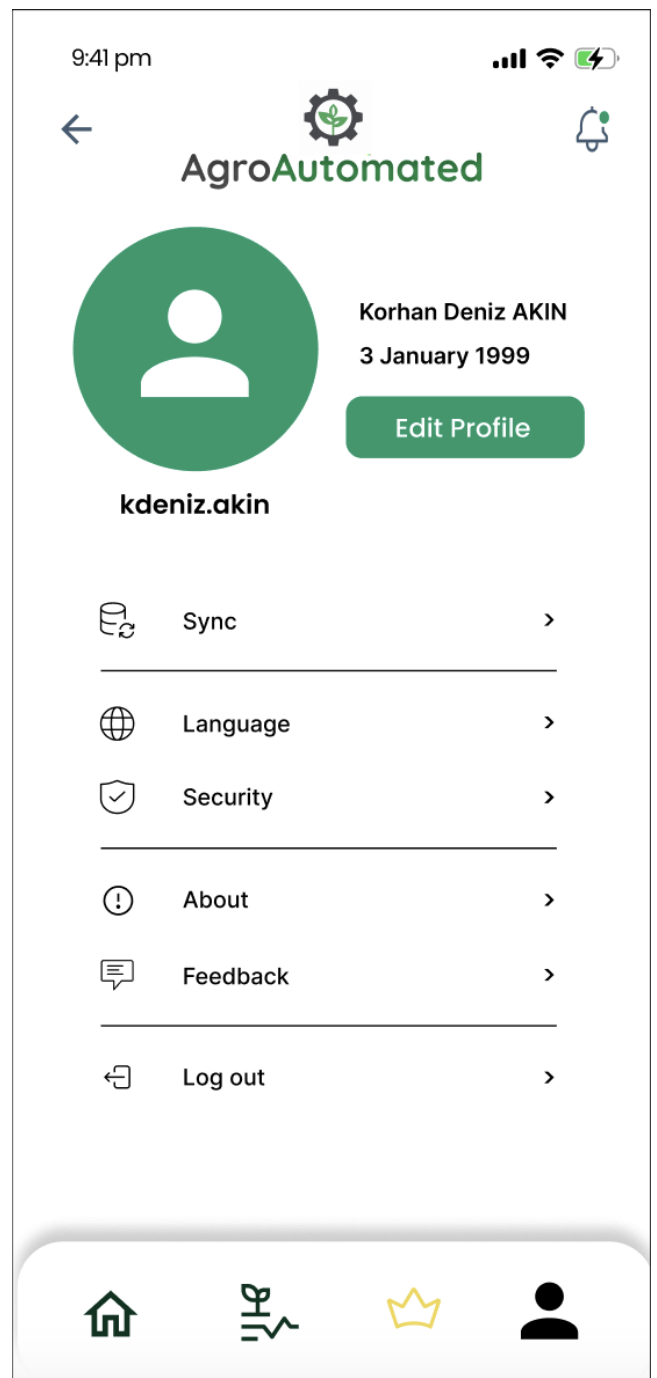
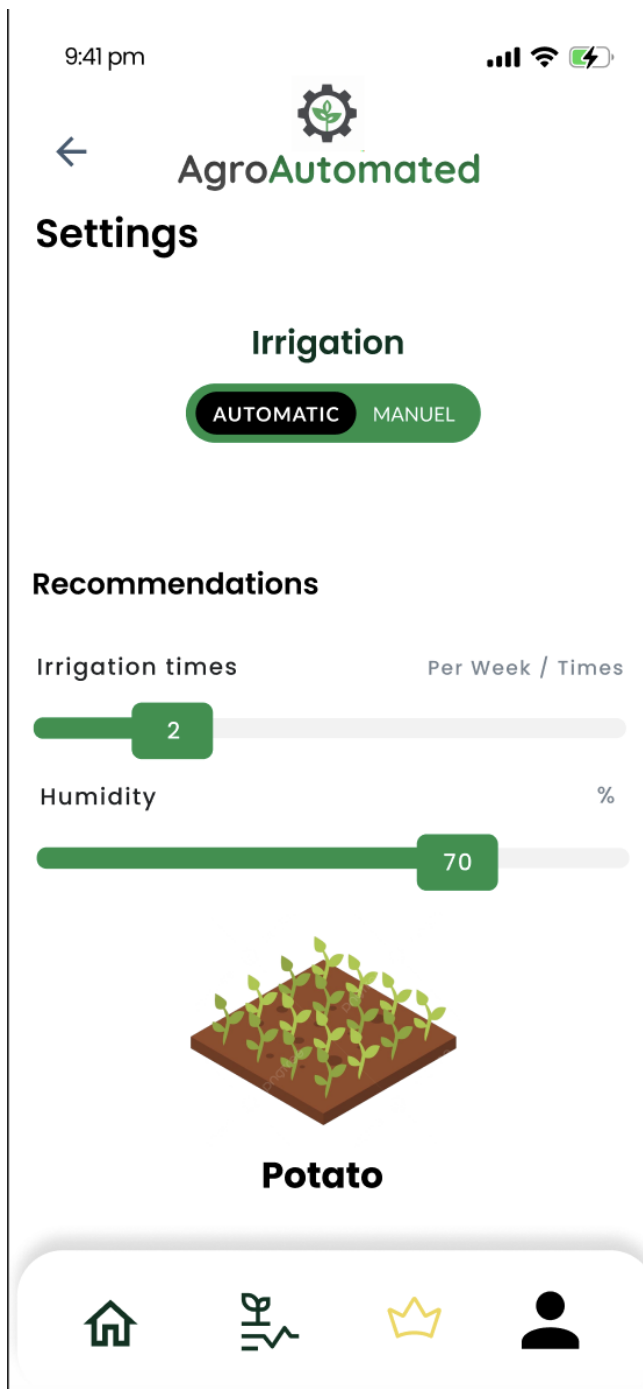
 Google

Already have you an account? [Login](#)









4. Glossary

- 4.1 KB: Kilobytes.
- 4.2 SRAM: A data memory for temporary data and calculations.
- 4.3 EEPROM: Stands for Electrically Erasable and Programmable ROM which is a storage component that has slow read and writing operations.
- 4.4 Serial Communication: Data transfer protocols between devices or components.
- 4.5 Microcontroller: A chip that is embedded into the Arduino Board. It consists of CPU, RAM, ROM, and other components.

5. References

- Department of Economic and Social Affairs, United Nations. (). Sustainable Development, End hunger, achieve food security and improved nutrition and promote sustainable agriculture. https://sdgs.un.org/goals/goal2#targets_and_indicators
- Singh, Medha. (2023). GreenGenie App Prototype. <https://www.figma.com/community/file/1218071238417011625/greengenie-app-prototype>,
- Vatsavayi, S., Girajala, M., Medavarapu, V., et al. (2021). Home Irrigation System Using Internet Control. <https://www.diva-portal.org/smash/get/diva2:1616588/FULLTEXT02>
- Yasin, H., Zeebaree, S., Zebari, I. (2019). Arduino Based Automatic Irrigation System: Monitoring and SMS Controlling. 4th Scientific International Conference – Najaf – IRAQ (4th SICN-2019). https://www.researchgate.net/publication/339646248_Arduino_Based_Automatic_Irrigation_System_Monitoring_and_SMS_Controlling
- Altexsoft. (2023). Non-functional Requirements: Examples, Types, How to Approach. <https://www.altexsoft.com/blog/non-functional-requirements/>
- School of Ocean and Earth Science and Technology at the University of Hawai'i at Mānoa, Department of Atmospheric Sciences. (2023). What are non-volatile memories and solid-state drives? [https://www.soest.hawaii.edu/atmo/index.php/knowledgebase/what-are-non-volatile-memories-and-solid-state-drives/#:~:text=Non%2Dvolatile%20memory%20\(NVM\),in%20order%20to%20retain%20data](https://www.soest.hawaii.edu/atmo/index.php/knowledgebase/what-are-non-volatile-memories-and-solid-state-drives/#:~:text=Non%2Dvolatile%20memory%20(NVM),in%20order%20to%20retain%20data)
- Atmel. (2016). Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet. https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf
- Williams, Elliot. (2014). Make AVR Programming. Make: Community.
- Mazidi, M. A., Naimi, S., & Naimi, S. (2011). The AVR Microcontroller and Embedded Systems Using Assembly and C (2nd ed.). Pearson.
- Flutter Official Website. Supported deployment platforms. <https://docs.flutter.dev/reference/supported-platforms>
- National Society of Professional Engineers. (2019). Code of Ethics for Engineers. <https://www.nspe.org/sites/default/files/resources/pdfs/Ethics/CodeofEthics/NSPECodeofEthicsforEngineers.pdf>