SYSC3010 |
Computer Systems Development Project

FLORALYFE

Plant Monitoring System Proposal

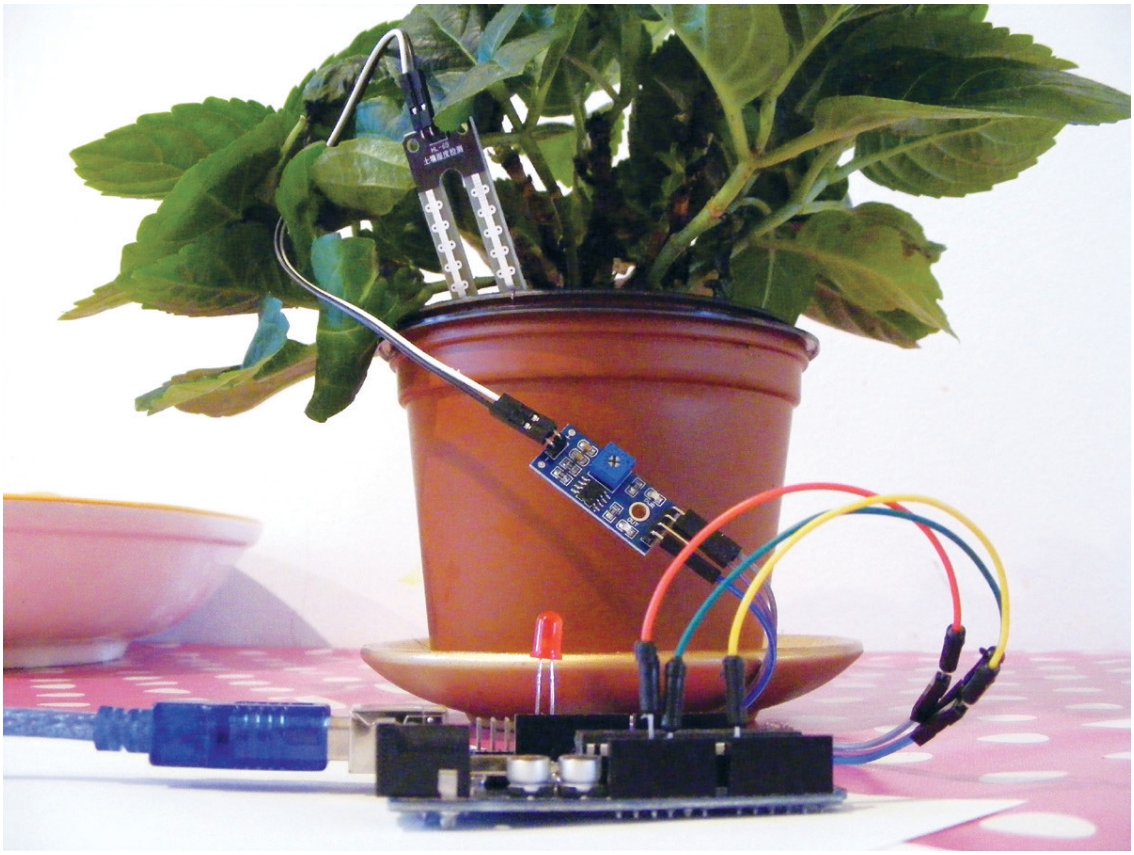


Figure 1. Mock representation of physical Floralyfe system [1].

Group L3W - G5

Abdalla Abdelhadi, 101142768
Zakariyya Almalki, 101152124
Yousef Yassin, 101143052

TA: Roger Selzler

February 13th, 2022

Table of Contents

1 Introduction

This document has been produced in response to a requirement of SYSC 3010 where students, working in teams, develop and implement a project for practical applications. Our group, Botanica (L3W G5), presents Floralyfe, a smart plant monitoring system that facilitates home-gardening. The following proposal document presents general information on the Floralyfe project whilst discussing the motivation and objectives of the system. A system overview follows, detailing the project's components, use cases, and how they come together in the final system structure. The document closes with comments on logistics concerning the project work plan, technical requirements, and required hardware.

1.1 Background

Houseplant demand has surged by 20% during the Covid-19 pandemic, with 66% of American households owning at least one houseplant [2]. The desire to become a plant parent has become significantly popular such that sales in the U.S. have increased by 50 percent to $1.7 billion as a result [3]. A possible explanation for this surge in sales is the studied reduction in stress resulting from interaction with plants [2]. Reduced stress would benefit adults enduring a stressful period, notably that imposed by the pandemic [3]. Despite the benefits, many new plant parents have also become aware of the heavy demand for plant maintenance [2]. Maintaining the health of indoor plants requires effort and time to ensure the reception of adequate levels of water and the hobby has often been described as "expensive, time-consuming and demoralizing" [4]. The increased effort has deterred many from owning plants. To solve the problem, many apps have since been released, such as *Blossom* and *PlantIn*, to help organize and assist with plant care [5]. These apps support setting up reminders to water an individual's plants, recognize diseases early, and offer personalized tips on growth optimization. However, none of these solutions have conveniently integrated with a system to perform these tasks, thus keeping individuals tied to their plants and leaving the plant hobby tedious and time-consuming.

1.2 Motivation

Studies have shown that the average houseplant owner will kill seven plants. It makes sense that the leading fears of most houseplant owners lie in ensuring their plant receives sufficient water. More than 60% of houseplant parents are frightened about not giving enough water to their plants [2]. To help ease the stress and attract new plant parents, *Floralyfe* is a remote solution that will continuously monitor and maintain the health of several plants and their environment. This will allow more individuals to experience and harvest the sense of purpose and joy offered by an influx of flora.

1.3 Project Objective

To broaden the plant parent demographic, *Floralyfe* aims to make the plant-growing hobby simpler and thus attractive. The system intends to remove the attached upkeep of plant care. *Floralyfe* will ensure stress-free and healthy plant growth, allowing individuals to spend more time enjoying their plants rather than worrying about their health.

1.4 Specific Goals

To facilitate the care of houseplants, *Floralyfe* will satisfy the following goals:

Remote Plant Vital Monitoring: The system shall allow users to register up to 2 plants and periodically measure the temperature, humidity, and soil moisture for these plants every hour. These measurements will then be visualized on a web interface.

Automatic Irrigation: The system shall water a user's plants. When moisture is low, a selected plant shall be watered until soil moisture returns to above the optimal value. Vital optima will be determined through a third-party API based on user input or automatic recognition of the plant species.

Vital Notifications: The system shall notify users of their plants' health. When a plant's vital measurements drop below the optimum, the plant's owner will be notified of the event detailing the critical vital(s). A corresponding icon will be displayed on the system's sense hat to communicate the notification, removing the need to check the user interface.

Plant Live Camera Feed: The system shall frequently send image frames of the selected plant to the frontend to create a live video feed. The frontend will allow users to alternate between selected plants, upon which the camera will rotate to face the new plant. Users shall also be able to rotate the camera yaw from the client, allowing them to observe their plants' environment.

Plant-specific Notes: The system shall allow users to track the progress of their plants. Users shall be able to create note posts for each of their plants to list important information, akin to a plant diary, to track plant growth.

2 System Design

2.1 System Overview Diagram

Floralyfe will be structured as a real-time distributed system. A detailed structural view of the system is illustrated by the deployment diagram in Figure 2. below. As conveyed by the network of clusters in the figure, the project shall consist of several deployed nodes, each with a specific task. These nodes will intercommunicate over the internet. Namely, the design consists of three *Plant Monitoring Stations* that are depicted in Figure 3. below. Each station is centered around a Raspberry Pi device running in headless mode. The stations will host a local database to store identifying information about their registered user to allow for multiplexed broadcasting. The database will also store a daily image of each registered plant to form a growth slideshow. The RPi stations additionally rely on three interconnected subsystems to achieve more complex functionality: the Vital Monitoring Subsystem, the Automatic Irrigation Subsystem, and the Camera Monitoring Subsystem.

Further detail regarding these three core subsystems, among others, is provided under *Section 2.2: Component Details*. In short, the vital monitoring system is responsible for sensor data collection, some of which is processed by the automatic irrigation system to determine when to water a plant. Along with these two systems, the camera monitoring system periodically captures images of a selected plant. All of this information — the sensor data, watering events, and camera images — are sent to a remotely deployed frontend client interface to be visually displayed to the user.

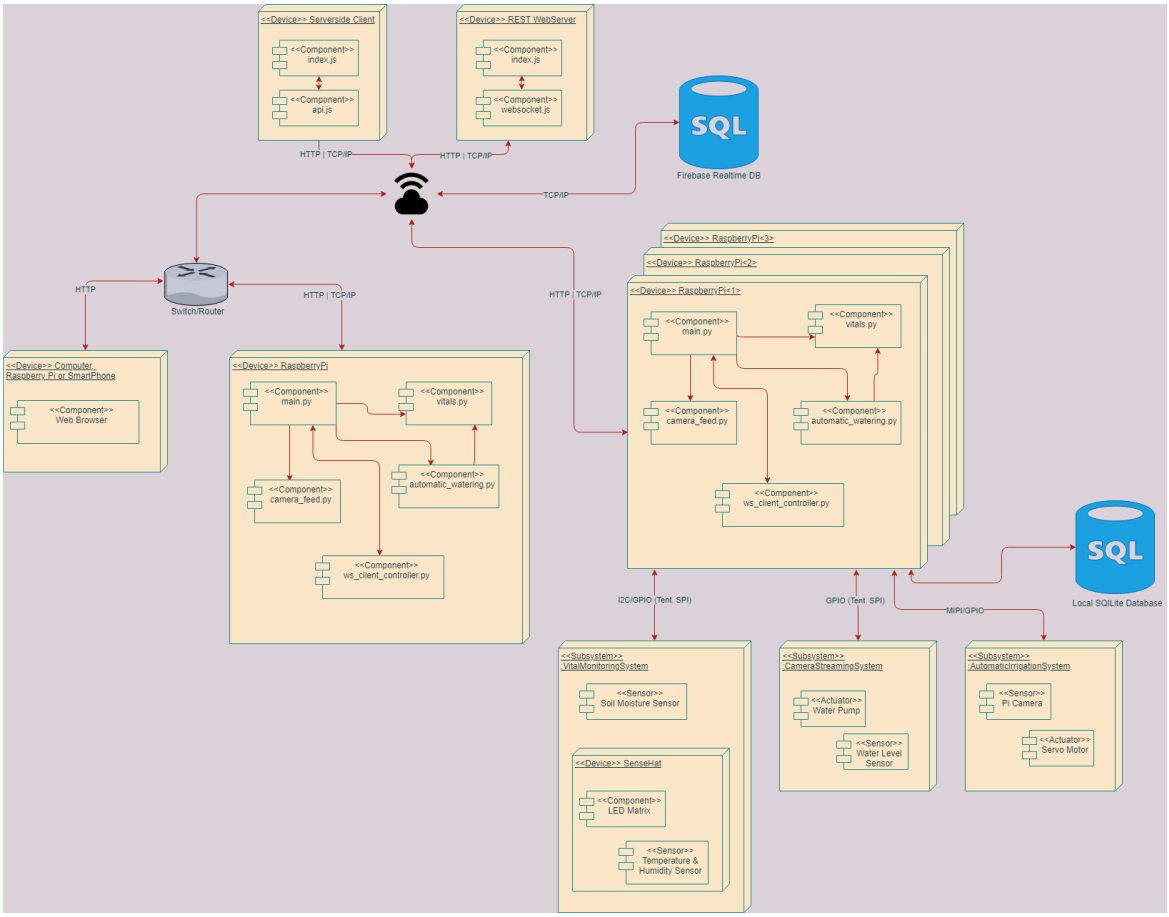


Figure 2. Deployment diagram

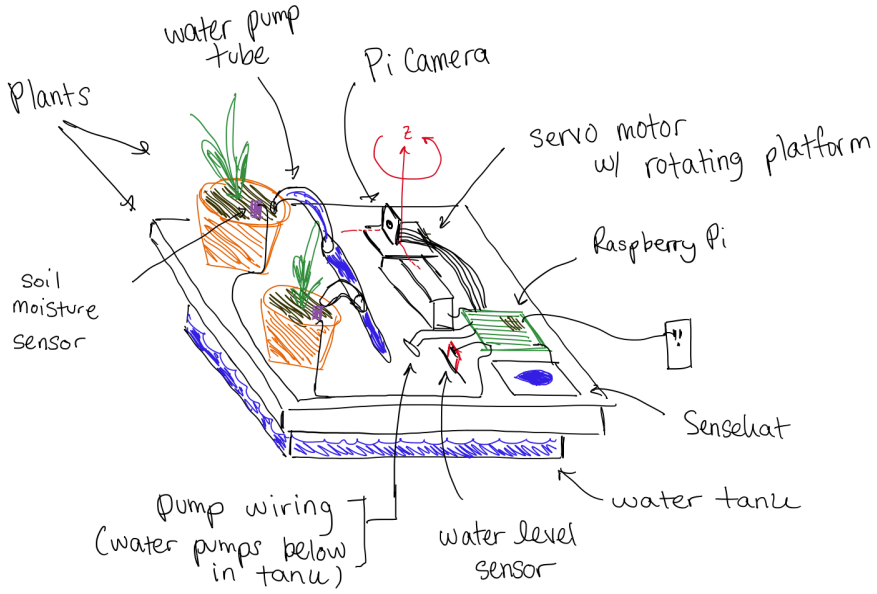


Figure 3. Structural Representation of Floralyte Plant Monitoring Station

**Wiring in Figure 3 is simplified for brevity. In reality, the wiring will go through a breadboard circuit to support organization and to route sensor data through an analog to digital conversion (ADC) circuit. Refer to 2.2 Component Details for detailed circuit diagrams.*

The frontend consists of a web-based graphical user interface that serves as a dashboard for the entire *Floralyfe* system. The interface will authenticate users such that they can be identified. The dashboard will additionally support viewing live raw and graphically displayed sensor data for a selected plant. The client receives this information from the backend server.

The communication and control center of the *Floralyfe* system depends on a backend server. Like the frontend, the backend will also be deployed on a remote server such that it can be easily reached by all system nodes. The server will consist of two primary components; a REST API and a WebSocket server. The REST API provides a consistent interface to the Firebase Realtime database which persists and transports information from the physical Raspberry Pi systems to the frontend website application. To decrease the load on the database, a WebSocket server will also be used for real-time, duplex, and multiplexed communication between all system nodes to perform remote tasks such as controlling the camera angle and manual watering.

2.1.1 Communication Protocols

The backend server will handle most of the communication by serving as a middleman that is connected to the frontend, physical Raspberry Pi systems, and database. The server functions by exposing a REST API that can be reached by all other nodes in the system to wrap database operations using HTTP POST and GET requests. The server itself will be the only node directly communicating with Firebase and will thus act as an adapter or wrapper, exposing a consistent interface to more conveniently query the store. The backend communicates with the database using a WebSocket operating on TCP/IP due to the real-time constraint [6]. Similarly, the backend server will also expose a WebSocket server that can be used for real-time, duplex, and multiplexed message passing between Raspberry Pi systems and the frontend client. This socket will also be using TCP/IP.

The primary purpose of the backend WebSocket is to send control messages from the frontend to a Raspberry Pi system and for the systems to send periodically captured images of plants back to the frontend in real-time. These images are captured using a Pi camera module that communicates with the Raspberry Pi using the MIPI camera serial interface protocol [7]. These images are saved on a second local database. Additional data acquisition is performed by the SenseHat temperature and humidity sensors. These sensors depend on the I2C serial protocol to interface with the Raspberry Pi. The remaining moisture and water tank level sensors are digital and will use GPIO to communicate with the Pi [8]. The system actuators - the camera servo motor and water pumps - use a DC power source and will thus also be interfaced using GPIO.

2.2 Component Details

The *Floralyfe* system consists of six primary components. These components include cloud and local databases serving as persistent data stores, an automatic irrigation system, a camera monitoring system, a vital (sensor) monitoring system, and a pair of remote servers hosting a backend and frontend.

2.2.1 Databases

The first component in the proposed system is the database which serves as a persistent data store. *Floralyfe* will rely on two databases; a cloud real-time Firebase database and a local SQLite database. Firebase will store five relational tables that link a user to their system device and their plants. Plants are further associated with sensor measurements, notifications, and notes. The local SQLite database on each system will store unique user identifying information and a daily photo taken of each of a user's plants to create a local growth time lapse. The schemas for both databases have been merged in Figure 4. below. Each database entry is identified by a universally unique identifier to support the maintainability of the system and facilitate associations between entries.

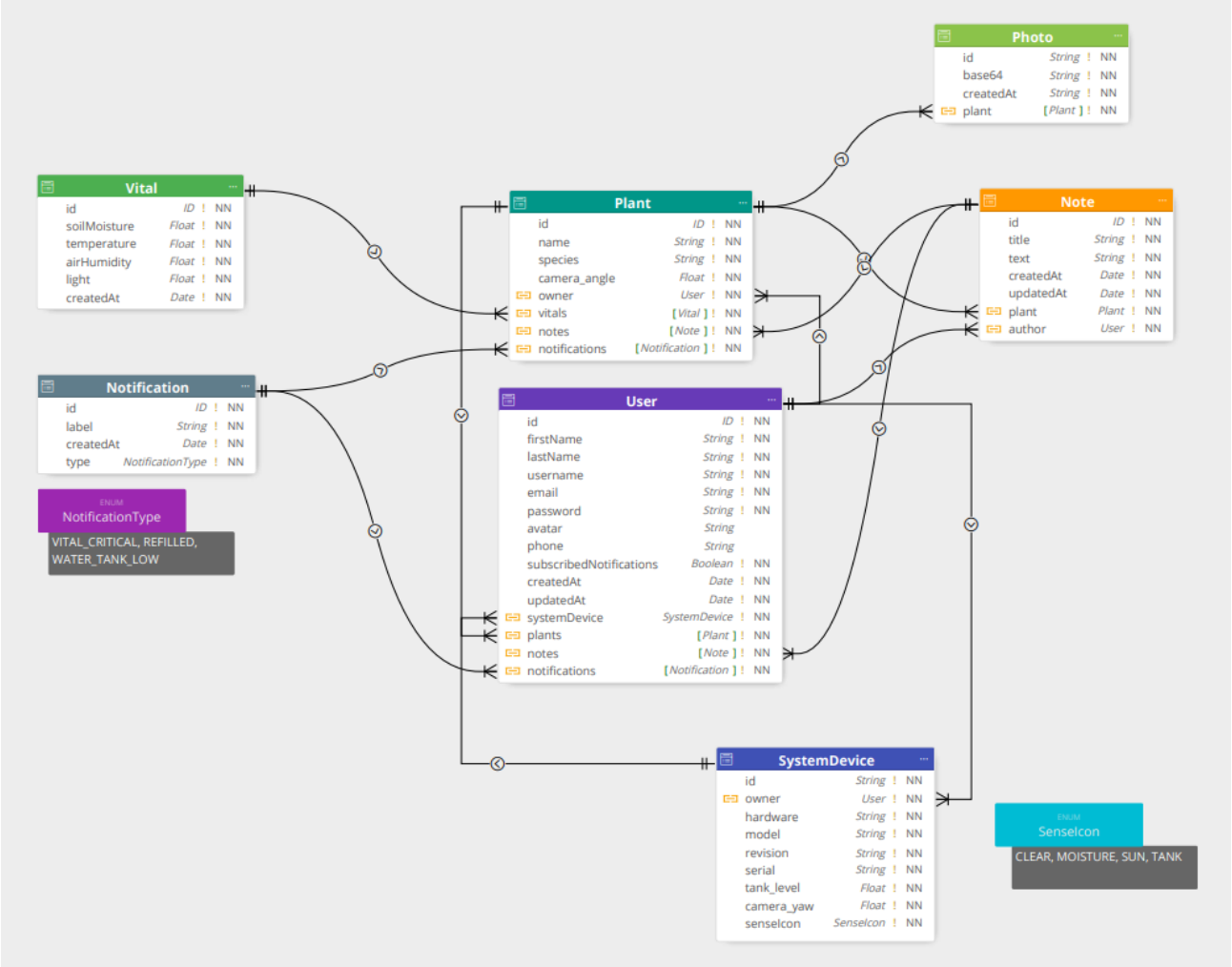


Figure 4. Database schema diagram

2.2.2 Automatic Irrigation Subsystem

The second component in the proposed system is the automatic irrigation system. This system will periodically acquire digital moisture readings from the vital monitoring subsystem, specifically a moisture sensor placed within the soil to monitor a plant's moisture. Once the soil moisture drops below the plant's ideal moisture, a GPIO pin will be activated to power a water pump to water the plant's soil provided that

sufficient water remains in the tank. Sufficient water levels are determined by a digital water level sensor connected to the Raspberry Pi in the manner illustrated by Figure 6., along with the moisture and pump connections. The subsystem consists of a closed-feedback loop, meaning the moisture levels will be checked a few minutes after watering to ensure adequate moisture has been retrieved. Otherwise, the watering process is repeated until moisture level has been retrieved. The subsystem is also listening to a WebSocket message originating from the frontend to manually override the feedback control loop and water the plant.

2.2.3 Camera Monitoring Subsystem

The third component in the proposed system is the camera monitoring subsystem. Figure 4. displays the camera monitoring subsystem with connections consisting of a Pi Camera mounted on top of a servo motor to control the camera's orientation. The system will be listening to a WebSocket message originating from the frontend to either indicate a control message to turn the camera to a certain heading or request images. When a plant is selected and viewed on the frontend, the camera monitoring subsystem will be responsible for sending a consistent but periodic image stream to simulate a live feed of the plant and its environment. Furthermore, at a specified time during the day, while sensors are polled, the system will take a picture of all the registered plants. These images are saved in the local database to incrementally build a timelapse of plant growth that can be locally retrieved and analyzed later.

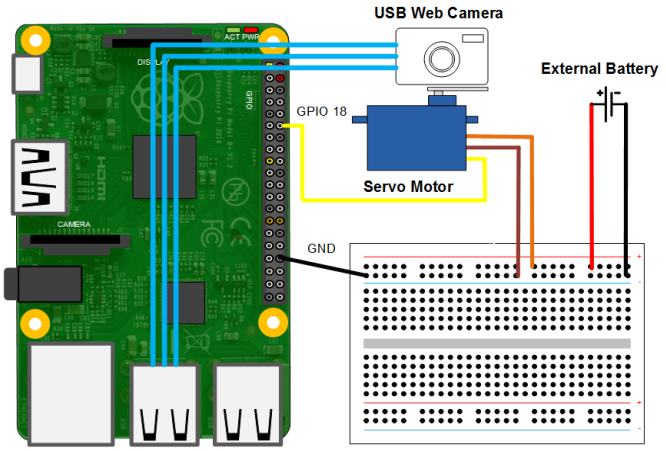


Figure 5. Camera System Hardware Schematic [7]

2.2.4 Vital Monitoring Subsystem

The fourth component in the proposed system is the vital monitoring subsystem. This component will monitor essential plant vitals and check that the watering tank is not depleted. To monitor, the subsystem will use the SenseHat to measure the temperature and humidity in the plants' environment. The system will also use digital capacitive moisture sensors to monitor the moisture level within a plant's soil around a specified threshold value. A water tank will be used as a water source to automatically water plants when the moisture level drops below an optimal threshold. Further, the system will recognize when the tank becomes depleted with a digital water sensor. Figure 6. below depicts the connections of these sensors to the Raspberry Pi system, along with the circuitry required for the pumps in the automatic irrigation system.

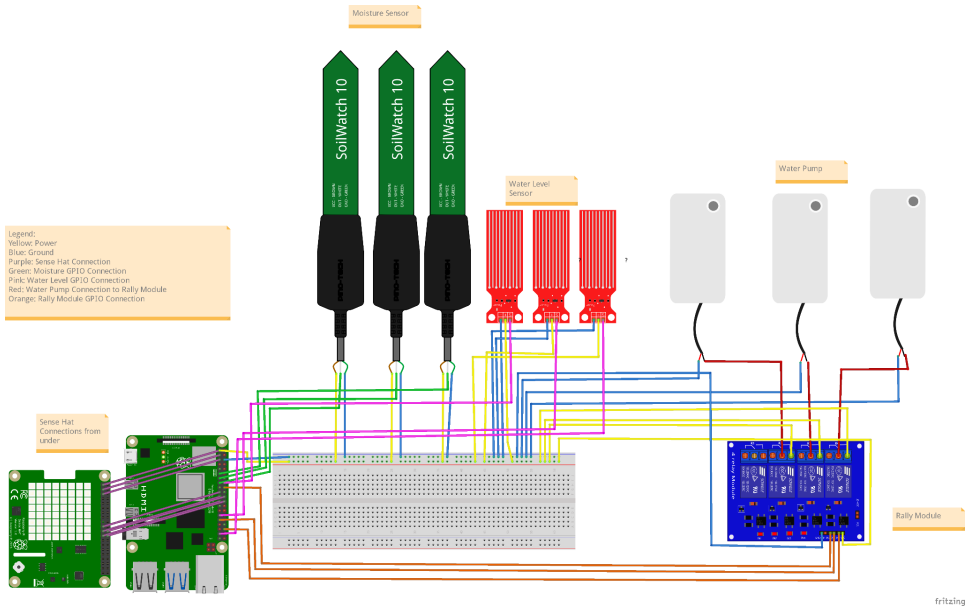


Figure 6. Vital Monitoring and Water Pumping Hardware Schematic

Plant vitals will be monitored periodically, every hour, and saved to the firebase cloud database to be displayed on the frontend website. During these periodic checks, the system will recognize if a measured value falls below a specified optimum determined by the user or an external plant API/database. In critical cases where a sensor value falls too low, the vital monitoring subsystem will request the backend server send a notification containing the critical value and plant information. The backend server will also log the event in a transaction store for future reference and notify the user of the event by email. The SenseHat will also be set with an informative icon to communicate the event without use of the frontend - for example, a water droplet will be displayed when soil moisture is low.

2.2.5 Backend REST Server

The fifth component in the proposed system is the backend REST server. The backend will be deployed on a remote node such that it is accessible by all nodes in the system and will act as both a middleman to route messages between nodes along with encapsulating the database interface with a consistent interface. The server will host a websocket on an empty port which will be accessed by both the frontend and connected Raspberry Pi devices.

The WebSocket will allow devices to listen to messages containing a specific id or topic tag such that the single connection can be shared among various clients with messages being multiplexed and demultiplexed using a specified field. The socket connection will be used to send real-time control to override requests from the frontend. These requests include making the camera face a specific plant or rotate a specified amount, setting an informative icon on the SenseHat, or watering the currently selected plant.

The backend will also feature a REST API to expose a consistent interface that acts as a wrapper to the Firebase database. The details for interpreting database queries will be abstracted away by the backend

server, allowing the React-based frontend and Python-based device clients to share a common request structure, thus enhancing maintainability and cohesion between nodes.

2.2.6 Frontend Website Graphical User Interface

The sixth and final major component in the proposed system is the frontend web-based interface. The website will appear similar to the wireframe shown in Figure 7. below. The site will consist of a login screen to authenticate users and a single page dashboard to serve all a user's plants' information to help monitor their health. This information includes an image feed of the currently selected plant, a list of notifications, and user-created notes for this plant and sensor data — both live values and historically plotted charts. The image feed will be obtained from the backend WebSocket server while the remaining information will be requested from the backend but obtained from the firebase database.

The user interface will also be bidirectional in that it will allow users to send control commands to change the state of their Raspberry Pi system. An input to change the currently selected plant will be present, thus changing the camera orientation to face the selected plant along with changing the values for the displayed information to correspond with the current selection. There will additionally be inputs to freely control the camera orientation to analyze the plants' environment, to change the informative icon displayed on the SenseHat, and to manually override the irrigation subsystem and water the selected plant on demand. Like the image reception, these input requests will be sent to the user's Raspberry Pi using the backend WebSocket.

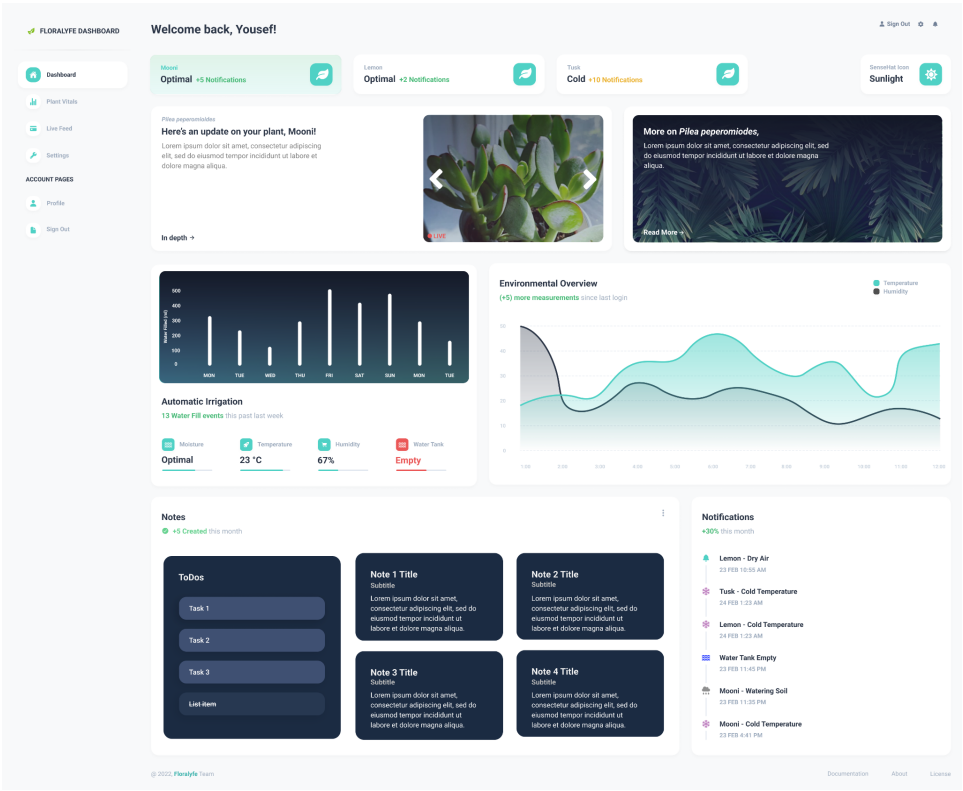


Figure 7. Floralyfe dashboard wireframe modified from Purity UI template [9]

2.3 Use Cases

To satisfy the goals proposed in *Section 1.4*, the *Floralyfe* system will satisfy six core use cases described and illustrated in the subsections below.

2.3.1 Use Case 1: RegisterSystem

The RegisterSystem use case allows users to register themselves along with a maximum of 2 plants to the system. Figure 8. below depicts a sequence diagram of the RegisterSystem use case with registration of a single plant. The registration process involves two distinct parts. First, the user must use the frontend to register and create an account. Second, they must then tie their Raspberry Pi system to their account by adding their username to the machine's local database as well. From there, the system will register the plants that are currently connected (based on connected sensors).

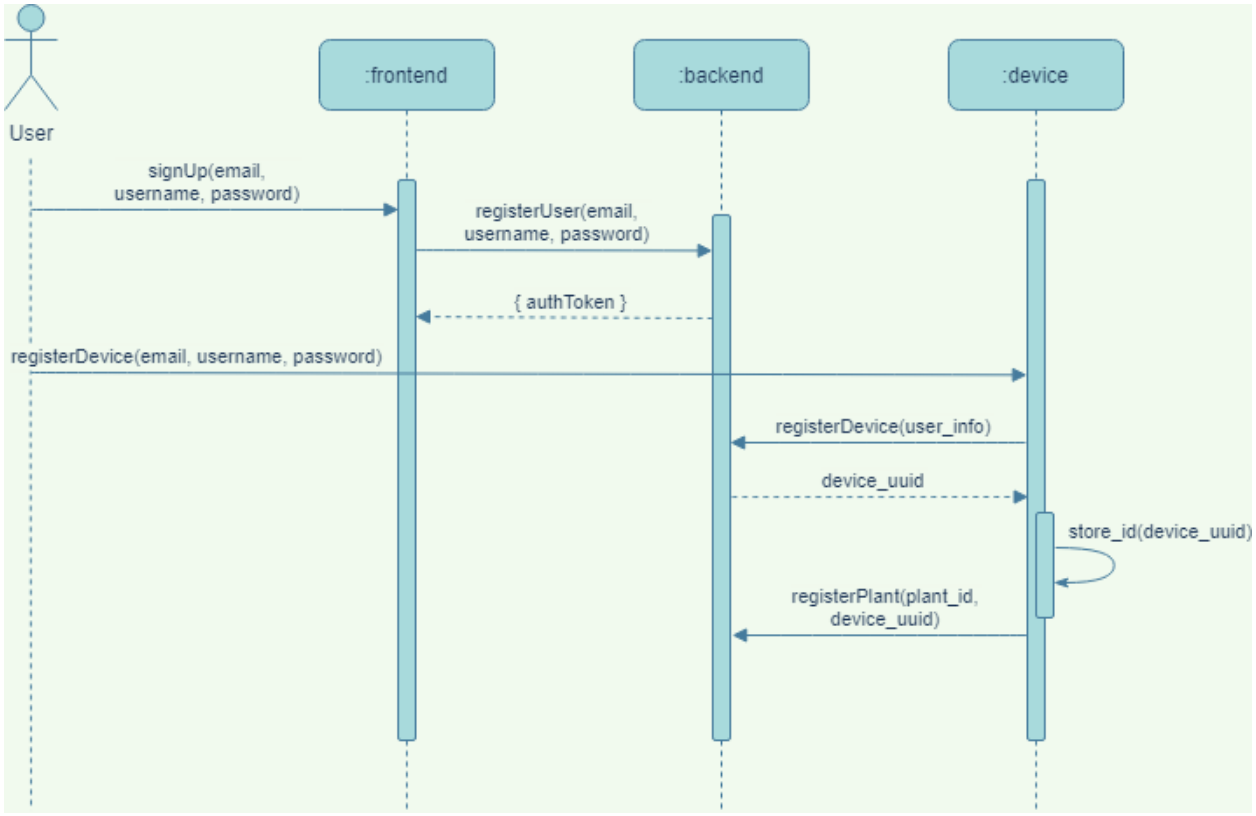


Figure 8. Sequence diagram for use case 1: RegisterSystem

2.3.2 Use Case 2: ViewPlantData

The ViewPlantData use case allows a user to select a plant through the frontend website. Figure 9. below depicts a sequence diagram of the ViewPlantData use case. Once a plant has been selected, a live feed of periodically captured image frames will be sent and displayed on the web interface. Along with the feed, periodic plant vitals will be captured through the connected sensors to send soil moisture, water tank level, temperature, and humidity data to the frontend to be visualized.

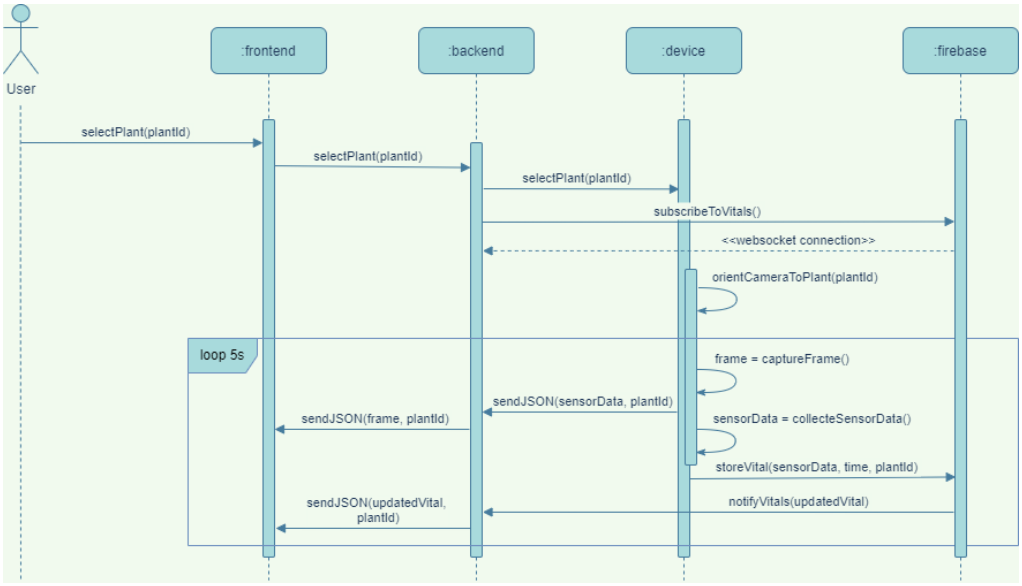


Figure 9. Sequence diagram for use case 2: ViewPlantData

2.3.3 Use Case 3: NotifyUser

The NotifyUser use case involves running a check on every measured vital and ensuring that it is above the recommended optimal value for the selected plant. Figure 10. below depicts a sequence diagram of the NotifyUser use case. The optimal values for each plant are predetermined by the user or through a third-party plant database on plant registration. If a vital measurement is below the optimal value, the NotifyUser use case will send an email to the user. The user will be able to subscribe using their email address through the web interface to have the option of being sent an email notification when any of their plant's vitals are in a critical condition.

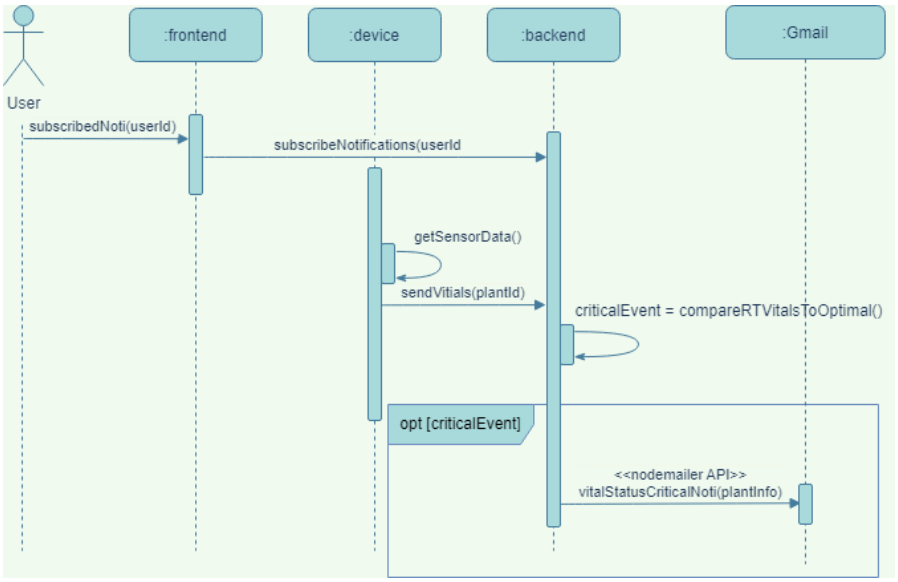


Figure 10. Sequence diagram for use case 3: NotifyUser

2.3.4 Use Case 4: AutomaticPlantWatering

The AutomaticPlantWatering use case will periodically measure the soil moisture levels of each registered plant every hour through the moisture sensor. Figure 11. below depicts a sequence diagram of the AutomaticPlantWatering use case. If the sensor gather's data suggesting that the soil is dry, then the automatic watering system willactivate and provide sufficient water to the plant until the soil moisture level reaches an ideal level.

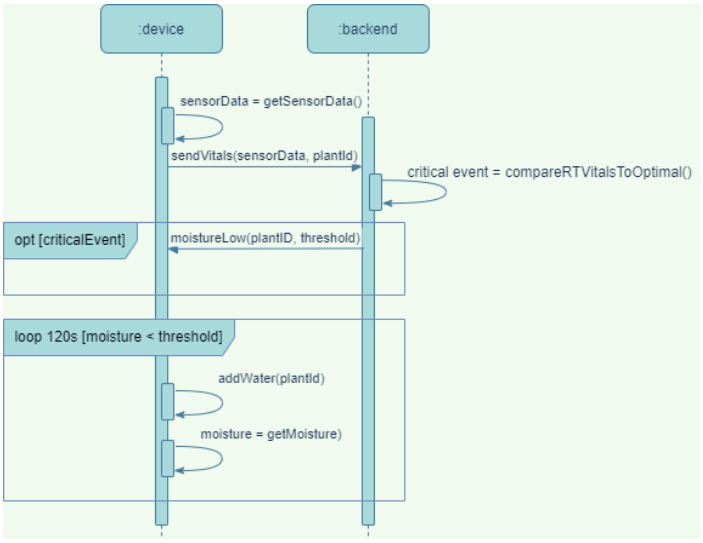


Figure 11. Sequence diagram for use case 4: AutomaticPlantWatering

2.3.5 Use Case 5: CreatePlantNote

The CreatePlantNote use case allows users to create notes tied to one of their registered plants through the web interface. Figure 12. below depicts a sequence diagram of the CreatePlantNote use case. Creating a note consists of sending a request to the backend, which in turn creates a request to create the note in the cloud firebase. Notes are similarly retrieved by the frontend, with a request to the backend server which is rerouted to firebase. These notes will be retrievable and displayed near associated plants on the frontend dashboard, allowing users to create a plant diary of sorts.

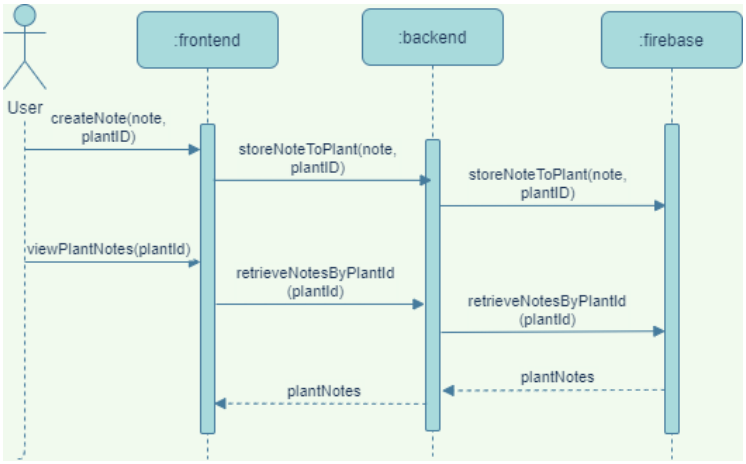


Figure 12. Sequence diagram for use case 5: CreatePlantNote

3 Work Plan

3.1 The Project Team

The *Floralyfe* team is made up of 3 group members: Abdalla Abdelhadi, Yousef Yassin, and Zakariyya Almalki. Yousef is interested in developing real-time distributed systems. He also has strengths in database architecture and full-stack development. Zakariyya enjoys everything about robotics, developing with cloud-based tools, and experimenting with embedded systems. Abdalla's strengths lie within FPGA programming; he also enjoys the creation and analysis of electrical circuits.

3.1.1 Roles and Tasks

Table 1. *Assignment of team members to major project tasks*

Major Tasks	**Primary**	**Secondary**
Sensor Data Collection	Abdalla & Zakariyya	Yousef
Achieve Multiplexed Bidirectional Communication	Abdalla & Yousef	Zakariyya
Complete backend cloud API	Yousef & Zakariyya	Abdalla
Complete frontend client interface	Yousef	Abdalla
Complete Camera Monitoring Subsystem	Zakariyya	Yousef
Complete Automatic Irrigation Subsystem	Abdalla	Zakariyya
Connect all subsystems	All Members	
Complete full unit tests	All Members	

The roles and tasks involved in creating the *Floralyfe* system are summarized in Table 1. above. The duties within the table are assigned based on each group member's strengths and interests. Everyone in the group is interested in the sensor data collection process as it will improve and help develop our circuitry and embedded hardware building skills. For the development of central processes such as communication and the backend server API, all members will work together. Working together is beneficial here since these systems are integral to *Floralyfe* and having all members understand them well is beneficial in developing the client subsystems. Yousef has experience working with frontend technologies, therefore he has been assigned the frontend client interface. Zakariyya enjoys data collection and processing so he's in charge of the camera monitoring system. Finally, Abdalla is interested in building automated feedback systems which put him in charge of the automatic irrigation system.

3.1.2 Teamwork Strategy

We will be following the Agile Software Development workflow as a teamwork strategy. Agile will help us if we need to quickly modify or change an aspect of the project as we will regularly communicate with each other and be giving status updates. Agile will also ensure that all team members remain on the same page

regarding the overall status of the project development through consistent progress updates and communication. Our team will use multiple methods of communication, such as Slack for ideas and setting up meetings. The team will also use zoom meetings to work and discuss the project during lab hours, SMS for quick reminders, and GitHub for anything code-based such as sharing the codebase, performing code reviews, and continuous integration.

3.1.3 What we will need to learn

The team has many technologies to learn as we have very limited experience with many aspects of our system, namely the integration of hardware and software in a distributed manner. The *Floralyfe* system depends on sensor data acquisition which is not trivial with a Raspberry Pi. We will need to learn how to create an analog to digital converter circuit to read data from moisture and water level sensors. Once data is able to be read, we will need to learn how to interpret this data and create control systems to automatically operate on feedback and regulate a plant's environment. On the software side, the work associated with establishing bidirectional communication through WebSockets will also be new and require some learning. Likewise, the creation and deployment of a backend server, server REST API, and frontend client interface will need to be learned as well since we do not have much experience or knowledge working experience with these components. Finally, strategies to effectively stub and unit test the system, especially hardware components, will need to be learned as hardware development is a new domain for the team.

3.2 Project Milestones

In accordance with our team's workflow, implementation of the *Floralyfe* system will incrementally take place through a series of seven milestones described in Table 2. below.

Table 2. *Major milestones in Floralyfe implementation*

Milestone Number	Milestone Name	Description	Deadline
1	Model and View Initialization	Create the frontend interface (with fake data) and database schemas with adapter wrappers (firebase and local).	February 23
2	Bidirectional and Multiplexed Communication	Achieve multiplexed bidirectional end to end communication with a websocket. Create a backend server to query firebase and integrate with the frontend. Develop a notification sending mechanism and deploy development backend and frontend servers.	February 27
3	Assemble and Interpret Hardware System	Acquire, write to and interpret sensor data. Assemble basic hardware systems. Integrate data with interface and add user login system.	March 5
4	Camera Monitoring Subsystem and	Complete camera monitoring system. Have images being sent periodically from each pi's camera to the frontend, with the ability to rotate cameras freely and to face selected plants. Store one picture of a plant, daily, to a	March 9

	Plant Registration	local database. Add plant registration system to frontend.	
5	Smart Irrigation Subsystem and Establish Vital Notifications	Complete automatic irrigation system. Add a closed feedback loop to check and restore soil moisture levels for each plant. Complete notification system to email users of critical vitals and low water tank levels. Update SenseHat display with representative icon for the plant it's facing. Integrate sensor data on frontend and control watering from client.	March 16
6	Combine Subsystems	Combine all subsystems together and ensure proper functionality. Make final additions to each subsystem, touch-up interface and add ability to create and read notes for plants.	March 20
7	Debugging	Quality assurance check. Run integration tests across the system and perform debugging to finalize the demo procedure and project as a whole.	March 31

Note: Unit tests are to be implemented incrementally throughout each milestone.

3.3 Schedule of Activities

Figure 13. below consists of a Gantt chart that displays a timeline with all the aforementioned project milestones and deliverables with their deadlines. A list of tasks leading up to each milestone is found towards the bottom half of the chart.

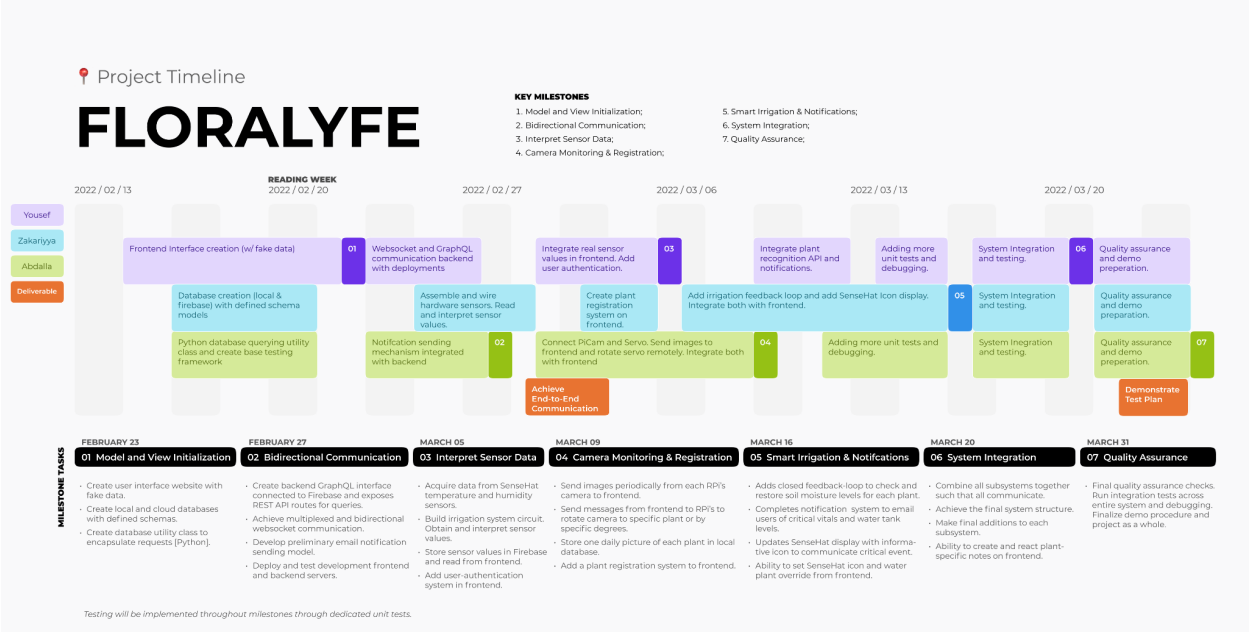


Figure 13. Project Timeline Gantt Chart modified from BIAsia Template [10]

4 Project Requirements Checklist

Computer Components

Is there a computer per student?

Yes, there are three Raspberry Pis being used with each hosting and instance of the *Floralyfe* system. There will also be a website that can be assessed externally from any other computer.

Is at least one RPi in headless mode?

Yes, all the RPis will be in headless mode. The task of each RPi is to collect data periodically and communicate with the backend/frontend. The frontend will be deployed remotely and will thus be accessible without use of the RPi devices.

Hardware Components

Is there at least one hardware device per student in the group?

Yes, there are a total of four different hardware devices used. The first one is a camera sensor that provides live streaming of the plants. The second hardware device is a servo motor that rotates the camera. There is a watering system, consisting of at least one soil moisture sensor, a water level sensor and water pump that allows the watering of plants. The SenseHat will also be used to collect temperature and humidity data.

Is there an actuator?

Yes, there are two different actuators in the system. The first consists of a servo motor to rotate the camera. The second is a DC water pump that pumps water from a water tank to the plant soil.

Is there a feedback loop? Interaction between input/output devices?

Yes, the automatic irrigation system will consist of a closed feedback loop. When soil moisture is detected to be low, the system will add water to the soil and, after two minutes, check the moisture level once again. If the moisture level is still low, this process is repeated. Otherwise, the feedback loop terminates.

Software Components

Is there a database?

Yes, there are two databases in the system. Each RPi will have a local database to store identifying info for its user, in addition to daily pictures of their registered plants. A Firebase Realtime database will be used to store user/device/plant information, plant vitals, notifications and plant notes.

Does the computer hosting the database have other responsibilities?

Yes, each RPi is hosting a local database along with running the entire *Floralyfe* system described within this report.

Does the database contain two tables (or types of information)?

Yes. As shown in Figure 4, the database schema consists of several tables including user/device information, plant information and timestamped plant sensor vitals.

Is there a periodic timing loop?
Yes, the vitals of the plants will be monitored and recorded periodically every hour. When a user selects a plant, camera frames of that plant will be sent to the frontend every 5 seconds.

Is there processing of the IoT data read?
Yes, the vitals of the plants that are collected will be displayed to the user on a website in a graphical format. This information will also be compared to the optimal living conditions of the plant to inform the user if any conditions are not ideal. These optima will be defined by the user or through an external plant recognition API.

Are notifications sent?
Yes, notifications through email will be sent to the user to inform them when their plant is recognized to be in a critical state. Also, an email will be sent whenever the water tank needs to be refilled.

Is there bidirectional GUI running on a computer?
Yes, there is a website that receives plant sensor information from the raspberry pi and displays it beside the according plant. The user is also able to select a plant and control the camera to view the plant and its surroundings. The website will send this information back to the raspberry pi to execute the command. Functionality to remotely and manually request watering of a plant will also be available.

5 Additional Hardware Required

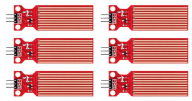
Implementation of the *Floralyfe* system will require additional hardware, depicted in two tables below. Table 3. shows the additional hardware that will be required from Carleton University while Table 4. shows the additional components that will need to be purchased.

Table 3. *Additional hardware provided by Carleton University*

Component Name	Picture	Quantity
Camera		3
Breadboard		3
Servo Motor		3

Wires		50-100

Table 4. *Additional hardware required to be purchased*

Component Name	**Picture**	**Quantity**	**Price/Unit ($)**
Water Sensor and Pump		3	30.99
Water Level Sensor		1	10.99
			Total: 110.46 (before tax and shipping)

6 References

[1] A. Williams, “How to build a plant monitor with Arduino,” *Electronics Weekly*, 12-Sep-2016. [Online]. Available: https://www.electronicsweekly.com/blogs/gadget-master/arduino/build-plant-monitor-arduino-2016-09/. [Accessed: 09-Feb-2022].

[2] “Houseplant statistics in 2022 (incl.. Covid & Millennials),” *Garden Pals*, 17-Jan-2022. [Online]. Available: https://gardenpals.com/houseplant-statistics/. [Accessed: 08-Feb-2022].

[3] B. Bharti, “The houseplant industry is thriving, thanks to millennials and their 'plant babies',” *nationalpost*, 08-May-2019. [Online]. Available: https://nationalpost.com/news/canada/the-houseplant-industry-is-thriving-thanks-to-millennials-and-their-plant-babies. [Accessed: 08-Feb-2022].

[4] E. J. J. Sullivan, “Covid lockdowns turned buying plants into the next big pandemic trend - for good reason,” *NBCNews.com*, 30-Jan-2021. [Online]. Available: https://www.nbcnews.com/think/opinion/covid-lockdowns-turned-buying-plants-next-big-pandemic-trend-good-ncna1256223. [Accessed: 08-Feb-2022].

[5] “Start your plant care journey,” *Blossom*. [Online]. Available: https://blossomplant.com/. [Accessed: 08-Feb-2022].

[6] “Surveillance camera with Raspberry Pi,” *Surveillance Camera with Raspberry Pi - MATLAB & Simulink*. [Online]. Available: https://www.mathworks.com/help/supportpkg/android/ref/surveillance-camera-with-raspberry-pi-r-hardware.html. [Accessed: 08-Feb-2022].

[7] “Firebase realtime database | firebase documentation,” *Google*, 2022. [Online]. Available: https://firebase.google.com/docs/database. [Accessed: 08-Feb-2022].

[8] “Kuman 5PCS soil moisture sensor module kit compatible with Raspberry Pi 3 pi 2 RPI 1 model B+ B compatible with Arduino Mega 2560 with 10pin female to female jump cables 20pin male to female Dupont Jump Wire Automatic Watering System KY70,” *Amazon.ca: Tools & Home Improvement*. [Online]. Available: https://www.amazon.ca/Kuman-Moisture-Compatible-Raspberry-Automatic/dp/. [Accessed: 09-Feb-2022].

[10] “Purity UI dashboard by Creative Tim & Simmmple,” *Creative Tim*. [Online]. Available: https://demos.creative-tim.com/purity-ui-dashboard/?_ga=2.165130298.429844829.1644303643-921955921.1644303643#/admin/dashboard. [Accessed: 09-Feb-2022].

[11] “Project roadmap timeline 项目规划时间线,” *Figma*. [Online]. Available: https://www.figma.com/community/file/1045731141584677708. [Accessed: 09-Feb-2022].