# SEEL 4213 - SOFTWARE ENGINEERING

DR. ISMAHANI BINTI ISMAIL

Group assignment: Smart Fish Farm Monitoring Application


| No. | Name | Matric Number |
|---|---|---|
| 1 | BEH ZHEN XIAN | A20EE0017 |
| 2 | LOW LIE XIN | A20EE0106 |
| 3 | VINCENT YEN WEI JIE | A20EE0225 |

## Project Overview

Sustainable Development Goal (SDG) 2 aims to achieve “Zero Hunger”. It ensures that every individual has access to safe and sufficient food that meets their daily requirements and diet. It also supports sustainable agriculture and food systems that does not compromise the economic, social and environmental well-being of the people and the environment. In this project, we aim to develop an application that provides a convenient platform for fish farmers to efficiently complete their day to day tasks in a more productive manner. With that, our project addresses the goal of eliminating hunger especially in regions heavily reliant on fish as an important source of protein.

Python is an open source high-level interpreted programming and object-oriented programming language whereby code can be segmented into different objects. In Python, the Object Oriented Programming (OOP) represents the classes and the class can define more than one object. Each class can contain a lot of attributes and the methods in order to extend the functionality. In general, Python is powerful for creating a frontend (UI) and backend (database, callback) application. Not only that, we also use MySQL which is an open-source database because of the popularity and the large community support. All the sensor values (water-level, temperature and pH value) and the fish data (Fish ID, species, length and etc) will be saved into the database.

Our Project features the use of external sensors to collect scientific data in a systematic manner. An alarm is also included to improve user experience by alerting the user when sensor values exceed critical thresholds. A personal Dashboard is developed and equipped with many functionalities such as live fish location viewer, parameters shown in a Graph user interface and the ability to control physical motions of the robot from a distance. To further enhance the user experience, we also included a Blynk application which displays real-time data on the go while providing the user with the ability to control the cruise of the robot in water as well.

# Requirement and Background

Fish farming is a crucial sector for sustaining livelihoods and providing a significant source of nutrition. However, fish farmers face various challenges, such as adverse weather conditions, water quality fluctuations, and unexpected events like floods. To address these issues, a comprehensive Fish Farm Monitoring Software is proposed.

Our fish farm monitoring software aims to help the farmer to ensure the optimal conditions for fish growth by monitoring and alerting on the water parameters. This can help minimize losses caused by adverse weather conditions, such as floods and extreme temperatures. The software assists in real-time data collection, analysis, and record-keeping to help in decision making. Farmers can also use the software for relocating the floating fish cages in event of floods by controlling the motors.

The functional requirements of our software includes real-time monitoring and alerting of the water parameters, database management, real-time display, and motor control. The system will employ several sensors to monitor the water parameters including pH level, water level and temperature. The automated data collection will occur at a regular interval to provide a real-time update of the readings. The system will alert the farmer when either of the water parameters exceed the predetermined levels using the Blynk platform for immediate action. For the database management, the software will have a centralized database for storing the historical water parameter data and fish information including species, quantity, and growth data for future analysis. In the meanwhile, the system will have a user-friendly dashboard displaying real-time readings of the water parameters. Farmers can also view the list of fish information in the database. Finally, the software will integrate with motor control systems to assist the movement of floating fish cages. During the event of flood, the system shall allow the relocation process of the fish cages.

For the non-functional requirements, the software should ensure the security, reliability, and scalability of its operations. Robust security measures will be implemented to safeguard sensitive fish farm data, with access to the system restricted through user authentication. The system will be designed to ensure the reliability and accuracy of data collection and monitoring, ensuring prompt delivery of alerts to enable timely responses. To accommodate the evolving needs of fish farmers, the software architecture is built to be scalable, allowing seamless integration of new sensors and fish cages. Two key assumptions are made to support the system's functionality: stable internet connectivity is assumed for real-time data transmission and alerts, and the sensors employed for monitoring water parameters must be compatible with the software. These considerations collectively contribute to the software's effectiveness in enhancing fish farming practices while prioritizing data security, reliability, and adaptability.

## User case diagram

Below shows the user case diagram. It illustrates the user case where a user wants to log in and out of the software using preset credentials. The system then verifies the username and password by fetching data from the database. If the provided credentials do not match any username-password pair in the database, it displays an error message to the user, providing user feedback and improving user experience. Furthermore, the software features the ability to set user settings such as changing passwords, email and usernames to provide customisation to the user.

The main feature here is to provide the user with an interface to show meaningful data related to our application. The fish data collected from the sensors such as water level, water temperature and water pH parameters are shown in real-time to the users via a real-time plotted graph. The application also provides a section for the user to conveniently set the threshold limit of the above mentioned parameters. A confirmation button and message is also included to further enhance the user experience and to prevent user error when using our application. We also provide the convenience to control the cruise of the robot using a slider.

Other than that, we also included an interface to view Fish data. Fish data such as Tag ID, species, gender etc. are shown. Data is first inserted manually by the user and it is updated to the database. The vital Fish data is then retrieved and shown to the end user.


Figure 1: User case diagram

# Design procedure

***Methodology***

The application was built on top of the Python programming language with the version of 3.9 together with the Anaconda virtual environment which can simplify the development process. Furthermore, all the required libraries have to be installed using “pip” commands in command prompt such as Pyqt5, numpy, and etc in order to extend the application functionality.

Not only that, Mysql needs to be installed in our computer whereby the host is localhost and creates the root user for the database in order to store the data into the database. That credential is very important when integrating with Python. Before creating a database, the understanding of the data structure is very important because every data might have different data length and data type. In our case, we are creating 2 separate tables which are “senValues” and “fishData” because those tables are not related to each other. Each data will be assigned a primary key as a unique id for every data.

| id | tagID | species | gender | cageNo | length |
|---|---|---|---|---|---|
| 1 | 10253652 | tilapia | male | 1 | 12.3 |
| 2 | 102478963 | tilapia | male | 1 | 14.8 |
| 3 | 10445687 | tilapia | female | 1 | 15.5 |
| 4 | 1321324657 | Tilapia | Male | 1 | 15.3 |
| 5 | 1231545123 | Tilapia | Male | 1 | 15.3 |
| 6 | 32135464 | Tilapia | Male | 1 | 15.9 |
| 7 | 13213465 | Tilapia | Male | 1 | 16 |
| 8 | 13645456 | Tilapia | Male | 1 | 13 |
| 9 | 1321346565 | Tilapia | Male | 1 | 15 |
| NULL | NULL | NULL | NULL | NULL | NULL |

| id | Timestamp | WaterLevel | TempValue | phValue |
|---|---|---|---|---|
| 1 | 1705395564.86506 | 0 | 0 | 0 |
| 2 | 1705395565.86506 | 0 | 0 | 0 |
| 3 | 1705395566.86506 | 0 | 0 | 0 |
| 4 | 1705395567.86506 | 0 | 0 | 0 |
| 5 | 1705395568.86506 | 0 | 0 | 0 |
| 6 | 1705395569.86506 | 0 | 0 | 0 |
| 7 | 1705395570.86506 | 0 | 0 | 0 |
| 8 | 1705395571.86506 | 0 | 0 | 0 |
| 9 | 1705395572.86506 | 0 | 0 | 0 |
| 10 | 1705395573.86506 | 0 | 0 | 0 |
| 11 | 1705395574.86506 | 0 | 0 | 0 |
| 12 | 1705395575.86506 | 0 | 0 | 0 |
| 13 | 1705395576.86506 | 0 | 0 | 0 |
| 14 | 1705395577.86506 | 0 | 0 | 0 |

Figure 2: Fishes data (table left) and sensor values (table right)

Based on the program flow chart below, the system needs to be turned on in which is a microcontroller with the WiFi connected. Thus, the application can be controlled remotely and work as expected. Due to the different procedures for controlling the sensor data and fish data, the flow charts are also different to each other.

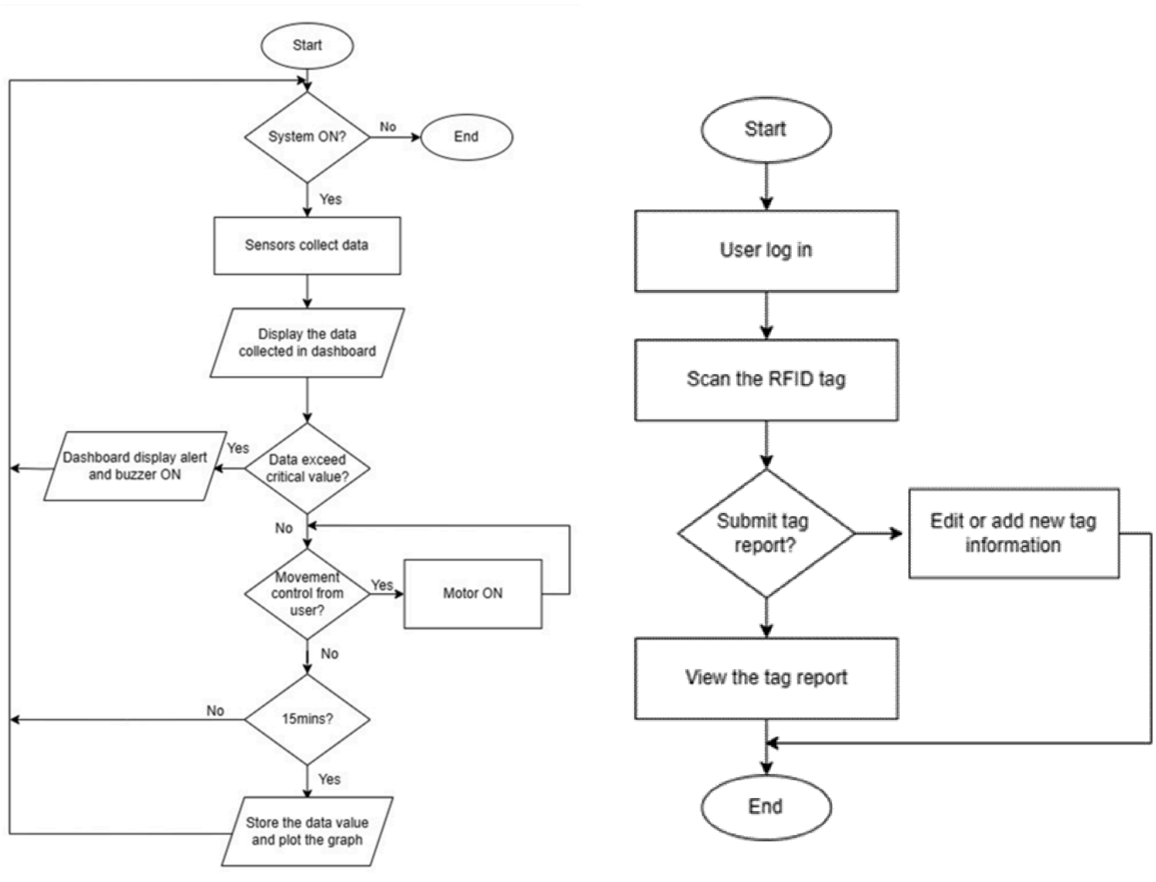


Figure 3: Program flowchart for application development

***Modular development***

The application was divided into 2 classes which are MainWindow and FishDetailsWidget. Each class represents the different page of the application whereby MainWindow is for the real-time sensor value update while the FishDetailsEWidget is for the fish data manipulation. In MainWindow, there are 9 methods which represent different functions such as initialize dataframe, setup graph graph and so on. The same flow goes to the FishDetailsEWidget which contains a lot of attributes and 5 methods specifically for that class. Both classes are not inherited to each other but use the callback function provided by the Pyqt5 package to switch between the pages.

*Interface design*

In terms of software development, the Python programming language, coupled with the Pyqt5 framework, is utilized for the application. The Human Machine Interface (HMI) design tool Qt5 designer is employed to construct the user interface (UI), and the styling of the application relies on the CSS framework. The application is comprised of two pages, namely the real-time update graph (main page) and the fish data (second page).

On the main page, five parameters that can be modified by users to meet the requirements of the fish farm, such as water-level limit, temperature limit, pH limit, motor left, and motor right, have been incorporated. The limit values are utilized to establish a maximum value for triggering alarms and notifying the user. Given the portable nature of our product, an embedded motor can be controlled using motor slider parameters on the dashboard. Furthermore, the display data in the dashboard is updated every second from the sensors.

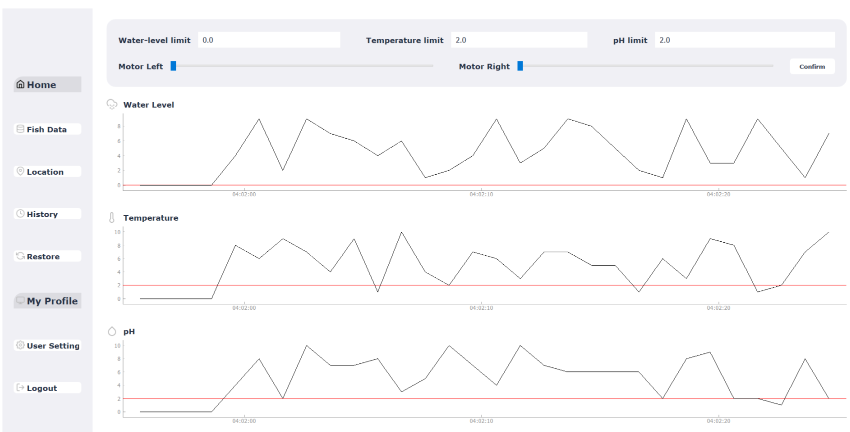


Figure 4: Application main page

Concerning the second page, details about the fishes can be observed from the dashboard, including the total number of fishes inside the cage, unique IDs of the fishes, etc. To enhance the practicality of the prototype, users can manually add, update, and delete fish data in the dashboard. Thus, all the data will be stored in the database.

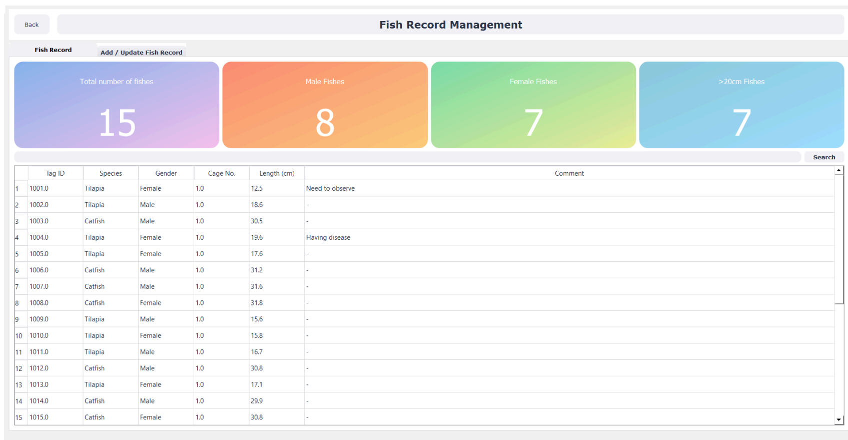


Figure 5: Application second page

***Class diagram***

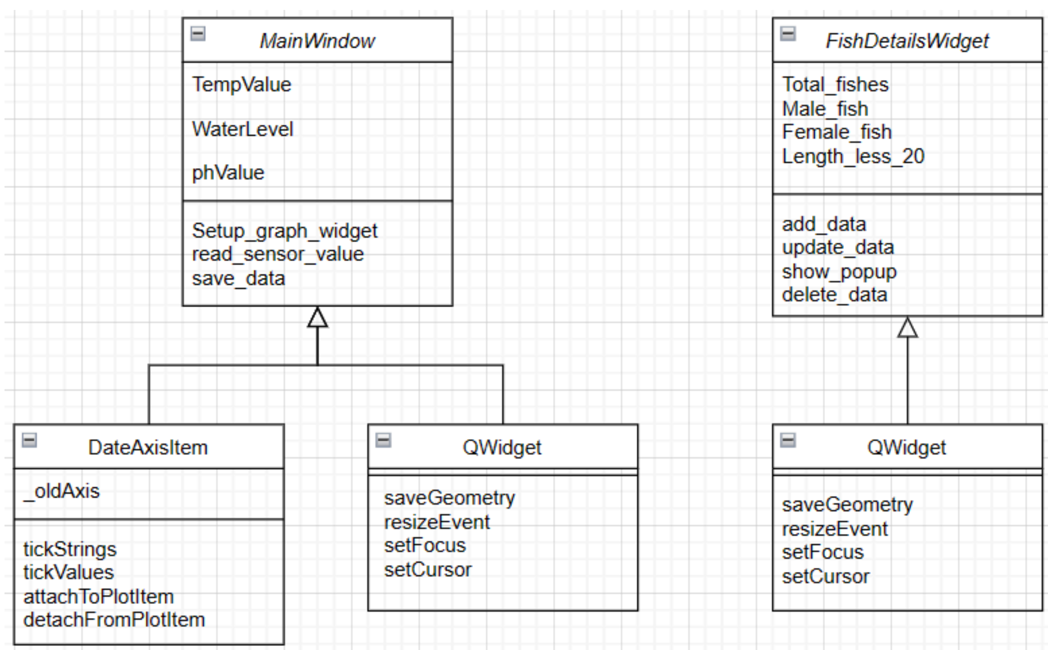


Figure 6: Application’s class diagram