Fish Tank Monitoring System

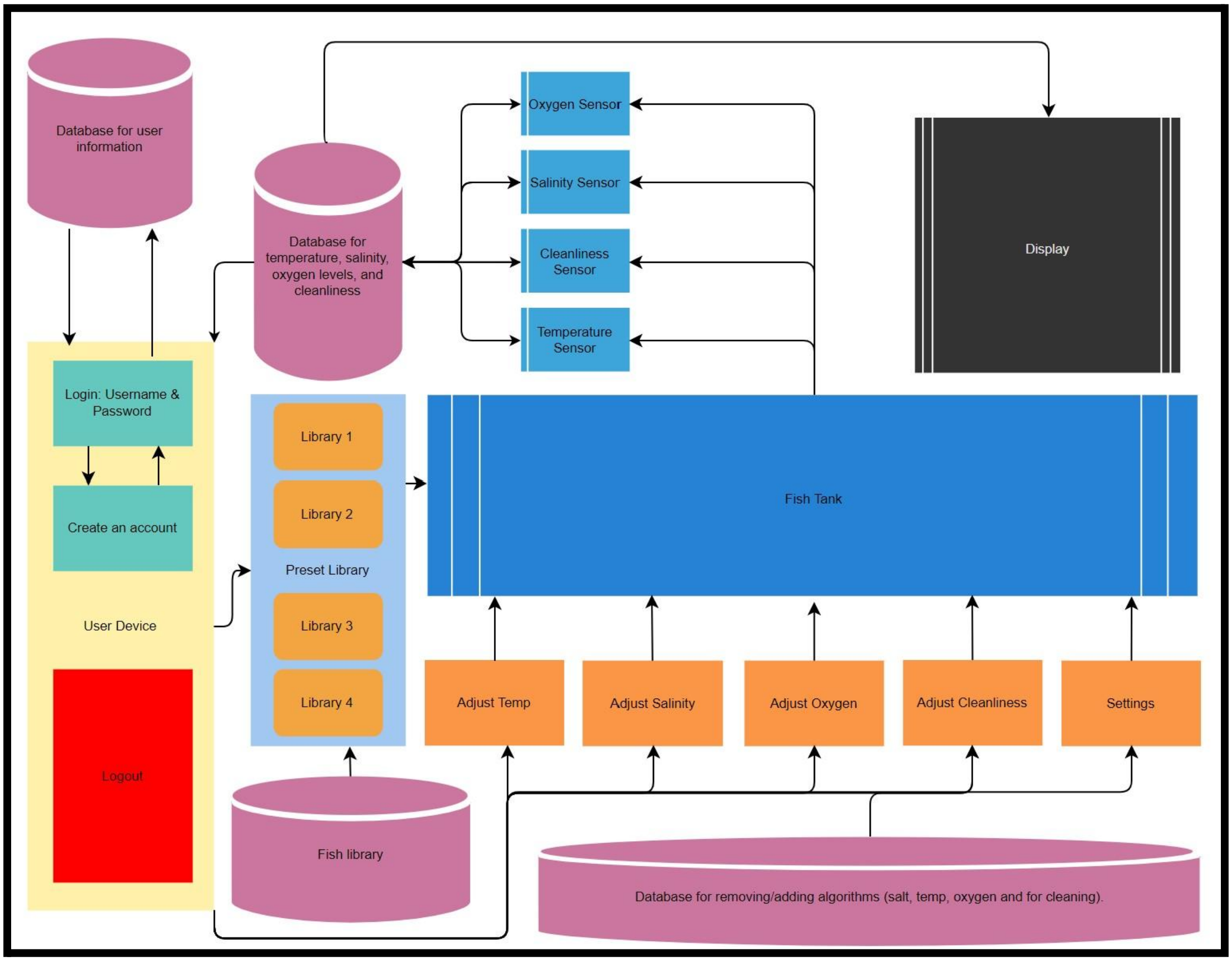
Software Design Specification Document

Development Team:	Role:	Contact:
Jose Garcia	Hardware Engineer	jgarciagomez7895@sdsu.edu
Alex Vo	Software Engineer	avo3681@sdsu.edu
Darren Lee	Security Engineer	dlee0083@sdsu.edu

System Overview

- The idea of the fishtank we are creating is to provide the user with an environment where they are able to access the necessary information to create a thriving aquarium. They will have access to information about a variety of fish as well as what fish can live in similar environments. The user will also be able to adjust certain living conditions for the fish such as temperature, oxygen levels, salinity, and cleanliness. The tank itself will have an automatic chemical balancing feature and will cater to the needs of the users and fish. The user will be able to interact with the fish tank system through the use of a mobile app that interfaces with the tank management system and hardware sensors. The app will also update users on the status of their tank via push notifications.

Software Architecture Overview

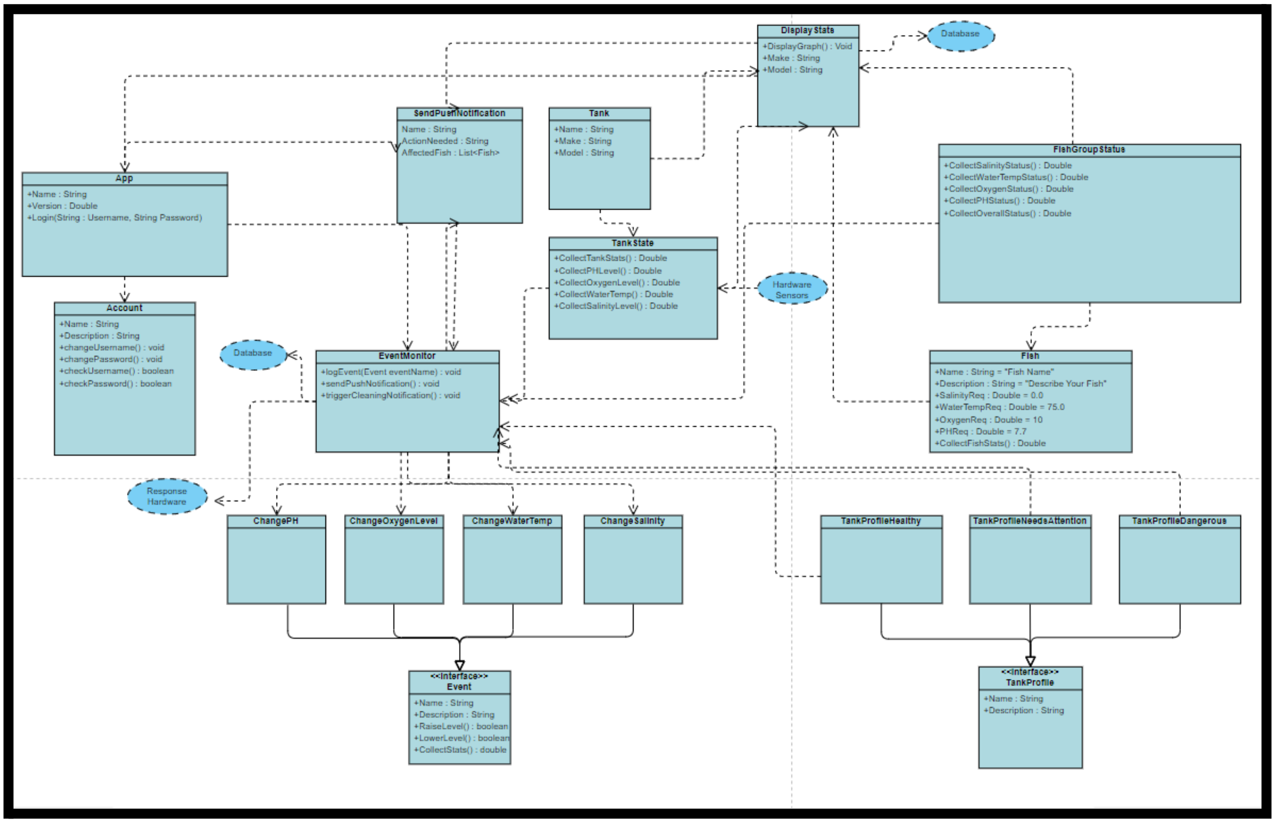


Pictured: Architectural Overview of the Fish Tank Monitoring System

Description of the software architecture diagram

Our fish tank is equipped with sensors that gather information about its salinity, temperature, oxygen levels, and cleanliness. This information is then stored in a database, allowing you to access historical data on these conditions throughout the day. Moreover, the current conditions are displayed on the exterior of the fish tank as well as in the accompanying app. The app has a login feature that stores user information, such as their username, email, phone number, and password, in a separate database. Once logged in, users can access fish libraries to learn more about the optimal living conditions for different fish species, including which ones can coexist. This account also stores the data and history of the user's fish tank. The fish library database contains information on the living conditions of various fish species. The app allows users to manually adjust the temperature, salinity, oxygen, and cleanliness levels of the fish tank through their phone. These commands are stored in a separate database, and users can adjust the intensity and quantities of each element in the settings.

UML Class Diagram


Display Stats
+DisplayGraph() : Void
+Make : String
+Model : String
Database
SendPushNotification
Name : String
ActionNeeded : String
AffectedFish : List<Fish>
Tank
+Name : String
+Make : String
+Model : String
App
+Name : String
+Version : Double
+Login(String : Username, String Password)
FishGroupStatus
+CollectSalinityStatus() : Double
+CollectWaterTempStatus() : Double
+CollectOxygenStatus() : Double
+CollectPHStatus() : Double
+CollectOverallStatus() : Double
TankState
+CollectTankStats() : Double
+CollectPHLevel() : Double
+CollectOxygenLevel() : Double
+CollectWaterTemp() : Double
+CollectSalinityLevel() : Double
Hardware Sensors
Account
+Name : String
+Description : String
+changeUsername() : void
+changePassword() : void
+checkUsername() : boolean
+checkPassword() : boolean
Database
EventMonitor
+logEvent(Event eventName) : void
+sendPushNotification() : void
+triggerCleaningNotification() : void
Fish
+Name : String = "Fish Name"
+Description : String = "Describe Your Fish"
+SalinityReq : Double = 0.0
+WaterTempReq : Double = 75.0
+OxygenReq : Double = 10
+PHReq : Double = 7.7
+CollectFishStats() : Double
Response Hardware
ChangePH
ChangeOxygenLevel
ChangeWaterTemp
ChangeSalinity
TankProfileHealthy
TankProfileNeedsAttention
TankProfileDangerous
<<Interface>>
Event
+Name : String
+Description : String
+RaiseLevel() : boolean
+LowerLevel() : boolean
+CollectStats() : double
<<Interface>>
TankProfile
+Name : String
+Description : String

Pictured: Unified Modeling Language (UML) graph of the Fish Tank Monitoring System

Description of classes

EventMonitor	The event monitor responds to incoming events and creates events depending on the data from the: - TankState - TankProfile - FishGroupStatus
<<interface>> Event	Event is an interface which is implemented as new variables are added to the tank.
ChangeSalinity	ChangeSalinity implements the Event interface and changes the salt levels within the tank.
ChangeWaterTemp	ChangeWaterTemp implements the Event interface and changes the water temperature levels within the tank.

ChangeOxygenLevel	ChangeOxygenLevels implements the Event interface and changes the oxygen level inside the tank.
ChangePH	ChangePHLevels implements the Event interface and changes the acidity level inside the tank.
CleanTank	Sends notification to the user to clean the tank, cannot be done chemically.
TankState	Tank State represents the current state of the tank based off of the data from the hardware sensors.
FishGroupStatus	FishGroupStatus takes the total status of all the fish based on the biological needs of the fish.
Fish	This class represents a fish and represents the biological needs and description of a particular fish.
<<interface>> Tank Profile	Tank profile is an interface that is implemented by different types of tank profiles.
Tank Profile - Healthy	This class represents the conditions for a healthy tank.
Tank Profile - Needs Attention	This class represents the conditions for an unhealthy tank that requires intervention or assistance.
Tank Profile - Dangerous	This tank represents the conditions for an unhealthy tank in which the inhabitants inside are at extreme risk.
Tank	The tank class represents the physical attributes along with the make and model of the Tank.
Account	The account class represents a user and includes a username and password. Outside classes can use the checkUsername() and the checkPassword() functions for authentication. These functions return a boolean to validate if the user authenticated successfully.
DisplayStats	The display stats is used by the application to display the statistics of the fish, as well as different chemical levels in the tank.
App	The app class is the main class to authenticate the user, in order to access all the other programs.
SendPushNotification	The SendPushNotification class is used to send notifications to the user's phone. It also sends the fish along with it to let them know what fish are affected by the tank’s current state.

Attributes and Operations

EventMonitor	+logEvent(Event eventName) - Logs an event attempt in the database +sendPushNotification() - Sends a push notification to the owner’s phone +triggerCleaningNotification() - Creates a notification to clean the fish tank
<<interface>> Event	+Name +Description +RaiseLevel - Returns true if level is raised, false otherwise +LowerLevel - Returns true if level is raised, false otherwise
ChangeSalinity	Implements Event Interface
ChangeWaterTemp	Implements Event Interface
ChangeOxygenLevel	Implements Event Interface
ChangePH	Implements Event Interface
TankState	+CollectTankStats
FishGroupStatus	+CollectSalinityStatus - Check to see whether the salinity of the tank matches overall fish needs. +CollectWaterTempStatus - Check to see whether the water temperature of the tank matches overall fish needs. +CollectOxygenStatus - Check to see whether the oxygen levels of the tank matches overall fish needs. +CollectPHStatus - Check to see whether the PH level of the tank matches overall fish needs. +CollectOverallStatus - Collects all statuses and creates an average for the statuses of all the fish.
Fish	+Name +Description +SalinityReq +WaterTempReq

	+OxygenReq +PHReq +CollectFishStats()
<<interface>> Tank Profile	+Name +Description
Tank Profile - Healthy	+Implements TankProfile Interface
Tank Profile - Needs Attention	+Implements TankProfile Interface
Tank Profile - Dangerous	+Implements TankProfile Interface
Tank	+Name +Description

Account	+Name
	+Description
	+Username()
	+Password()
	+ChangeUsername()
	+ChangePassword()

DisplayStats	+DisplayGraph() : Returns void as it displays the graph of the tank data. +Make +Model
App	+Name : String +Version : Double +Login(String : Username, String Password)
SendPushNotification	Name : String ActionNeeded : String AffectedFish : List<Fish>

Development Plan & Timeline

Name	Role
Jose Garcia	Responsible for the hardware integration with the software system. Will develop APIs to connect with the other backend

	software used by Alex. Will be responsible for developing unit and integration tests in order to ensure each hardware module operates as intended.
Alex Vo	Responsible for software integration of the system and ensuring proper control logic for the tank monitoring system. Alex will be responsible for creating the overall end to end testing suite along with ensuring the proper alerts and notifications are sent between each class.
Darren Lee	Responsible for setting up the security and authentication of the system. His responsibilities will include verifying the software integrity of the system such as ensuring that hardware does not fail. In addition, he will ensure that all users are authenticated successfully. Because this system can be accessed remotely through an app we would like to prevent our clients' fish tank system from being accessed.

2023 Development Timeline (9 Months)

March - Jose Garcia

- Investigate the market to determine the best commercial off-the-shelf hardware options
- Experiment with different hardware to identify the best reliability.
- Begin mapping out code that will follow the UML diagram

May - Jose Garcia

- Develop software notification system, and start development on android and iOS platforms.
- Create methods to add/remove dirt and gunk in the tank, salinity, and oxygen as well as increase/decrease temperature.
- Create and test databases to retrieve and store fish data

July - Alex Vo

- Meet with the client to determine to demonstrate a working model of the tank
- Make any changes that the client might want (materials, settings, user interface)
- Create user information database
- Find a discover any possible bugs in the prototype
- Begin testing

August - Darren Lee

- Refine the product, start creating unit tests and identify any bugs
- Finalize materials for fish tank
- Ensure clients needs were met

October - Alex Vo

- Discuss with the client about any final adjustments
- Deliver final product to the client for public use

December - Darren Lee

- Follow up with the client, and resolve any issues/bugs post-release after a couple of months of being out.
- Create a maintenance system
- Test before providing any updates