Two Unit Tests

- FishGroupStatus - CollectOxygenStatus(): Double - **Darren**

FishGroupStatus
Double: OxygenStatus
Double CollectOxygenStatus()

Testing For Correctness

Condition: Oxygen Status has not been collected

Condition: Oxygen Status is outdated and and needs to be updated

Testing For Error

Condition: Function should not return a negative value

Condition: The lowest possible value returned should be zero

- ChangeOxygenLevel - RaiseLevel() : Boolean - **Alex**

Event <<Interface>>
ChangeOxygenLevel
Boolean: RaiseLevel()

Testing For Correctness

Condition: If the Event monitors the Oxygen level is too high then it should change it lower

Condition: If the Event monitors the Oxygen level is too low then it should change it higher

Testing For Error

Condition: In situation the Monitor should change the Oxygen level but do not

Condition: In situation the Monitor should not change the Oxygen level but do anyway

The Event should monitor constantly the Oxygen level of the tank whether it meets the requirement of presets. If any changes happen, it will change the level accordingly to either lower or higher. After changes, the event will keep monitoring the post-change whether it did correctly or not. All will be compared to the database and anything changes happen will report to EventMonitor then it could send Push notification to the app for the user to monitor.

Two Functional Tests

- App - Login(String : Username, String Password) -> +checkUsername() : boolean, +checkPassword() : boolean - **Jose**

App - Login(String : Username, String Password)
+checkUsername() : boolean
+checkPassword() : boolean
changePH() : boolean
changeSalinity() : boolean

Testing For Correctness
Condition: The App class should allow users to sign into their account
Condition: Personalized settings and information is preserved from last sign in
Condition: User should be able to log out

Testing For Error
Condition: Signing in should direct the user to a home page
Condition: Info matches last sign in
Condition: Logging out should direct user to the login screen

In this system test, our objective is to verify the login feature's functionality within the fish tank's app. The first step is to confirm that a user can successfully create an account. This entails storing the user's information in a database that we can access from the app when the user logs in.

Next, the App class accesses the Account class to find if the user's input matches with the stored usernames and passwords. If a match is found, the user should be granted access to the app's features. If no match is found, then an error message should appear. In the event a match is found, we will proceed to adjust various settings within the app and verify that they are maintained after the user logs out and logs back in.

To accomplish this, we must first confirm that users can log out of their account in the first place and then log back in. Upon logging out, the user should be redirected to the login screen, where the login process can be repeated. Once the user logs in, their previous settings should be preserved.

- Fish - +CollectFishStats() : Double -> Collects the status of all the fish and creates an average score for them - **Alex**

Fish
Double: CollectFishStats()
Double: +SalinityReq()
Double: WaterTempReq()
Double: OxygenLevel()
Double: PHReq()

Testing For Correctness
Condition: In case it need to take all the data but actually calculated with missing data
Condition: In case it takes all the data but calculate failed and output wrong calculation

Testing For Error
Condition: It need to take data to calculate for information but could not calculate properly
Condition: It need to take data to calculate but the system failed to deliver data

The CollectFishStats function will call results from SalinityReq, WaterTempReq, OxygenLevel, PHReq in the same class of fish, which are collected from class FishGroupStatus and call all the

functions respectively. Then, it will calculate an average number from all the data to send it to class DisplayStats which is also sent to the database.

Two System Tests

- DisplayStats - +DisplayGraph() : Void - **Jose**

DisplayStats
DisplayGraph(): void

Testing For Correctness

Condition: Running and logging every event on a graph in the app

Testing For Error

Condition: Ensuring every single event logged in the graph is accurate

Class : Function - Name	Testing For:
DisplayGraph() : void	Graph is up to date and aligns with current stats
TankState: +CollectTankStats : Double	Current stats are accurate and align with what is occurring inside the tank
App : void	Accurate graph is displayed in the app

To conduct a thorough system test for this function, our first step is to verify whether the app can successfully display the graph in our app. We can accomplish this by testing the DisplayGraph() function within the DisplayStats class, which retrieves data from the hardware sensor database. If the graph fails to appear in the app after it is called from the App class, we will need to revise our approach and retest until it functions correctly.

If the graph appears as expected, our next priority is to verify its accuracy and alignment with the actual tank state. We can compare the graph's values against external resources, such as a trusted thermostat, to confirm their validity. By comparing these values, we can establish whether the graph is reliable and make any necessary adjustments.

Assuming the graph is accurate, we can periodically verify the app's ability to update properly. If the values remain consistent, we can confidently conclude that the system is functioning correctly. Otherwise, further revision is required.

- EventMonitor - +logEvent(Event eventName): void - **Darren**

EventMonitor
Double: OxygenStatus
void logEvent(Event eventName)

<u>Testing For Correctness</u>
Condition: Running and logging an event that results in the successful change of the water temperature levels
Condition: Running and logging an event that results in the unsuccessful change of PH levels due to missing supplies

<u>Testing For Error</u>
Condition: Ensuring every single event is logged.

Class : Function - Name	Testing For:
EventMonitor : logEvent(Event eventName)	Event is identified, created, run, and logged
ChangeWaterTemp: +RaiseLevel() : boolean	WaterTemperature is raised successfully;

Class : Function - Name	Testing For:
TankState : CollectWaterTemp() : Double	WaterTemperature from tank is collected

FishGroupStats: CollectWaterTempStatus() : Double	Water Temperature requirements from fish is collected
TankProfileNeedsAttention : getDescription	The profile for what a healthy tank looks like is collected