**Overall Test Plan**

Our approach to the testing is to test each large component to make sure that the flow of information is present. By this, we are testing to see if information can be passed to and from each component of the project. We will create test data that can be passed through each of the large components of the project to see if consistency will be there. There are three large components that will be tested, the Pi Server, the Dive Master registration and QR code generation, and the User Dive Logs with the ability to scan the QR code from the Dive Master.

**Test Case Descriptions**

|  |  |
| --- | --- |
| PS1.1 | **AWS Server Test 1** |
| PS1.2 | This test will ensure that the server can be reached by the app. |
| PS1.3 | This test will get a SQL query from the app and that will be stored into the database that is set up. |
| PS1.4 | Inputs: A made up query that has information that the app requires. |
| PS1.5 | Outputs: If using MySQL functions, we can see and retrieve any information that has been passed to the database. |
| PS1.6 | Abnormal |
| PS1.7 | Whitebox |
| PS1.8 | Functional |
| PS1.9 | Unit Test |
| PS1.10 | Results: This was tested by making a bare bone app that focused on backend. If the app could connect and send the information, it would give a notification that the data was sent. The result of this was a success. |

|  |  |
| --- | --- |
| PS1.1 | **AWS Server Test 2** |
| PS2.2 | This test will ensure that items are being queried to the SQL portion correctly. |
| PS2.3 | While conducting test 1, we will have to also see if the information is correct that is being passed or retrieved when using an SQL command server side. A new query will be made server side passing information to the SQL server if needed. |
| PS2.4 | Input: Information from server-side SQL command or from app. |
| PS2.5 | Output: Data that matches what was put in and for the correct columns. |
| PS2.6 | Whitebox |
| PS2.7 | Normal |
| PS2.8 | Functional |
| PS2.8 | Unit Test |
| PS2.10 | Results: This test was also a success as the make-shift app sent correct data. |

|  |  |
| --- | --- |
| PS3.1 | **AWS Server Test 3** |
| PS3.2 | This test will ensure that the items in the SQL server are retrievable. |
| PS3.3 | This tests that there are no errors when trying to send information back to the app such as errors that appear in the server log. |
| PS3.4 | Inputs: Request from the app for information. |
| PS3.5 | Output: The requested information (First name, Last name, Dive Number, and Company) are sent back to the app. |
| PS3.6 | Normal |
| PS3.7 | Blackbox |
| PS3.8 | Functional |
| PS3.9 | Unit |
| PS3.10 | Results: This test was somewhat successful as building the UI of the make-shift app resulted in nothing being displayed yet meeting the functional logic of, "If receive info, go to next UI page" |

|  |  |
| --- | --- |
| DM1.1 | **Dive Master Test 1** |
| DM1.2 | This test will test to make sure that the UI is not messed up. |
| DM1.3 | We will go through Dive Master information and fill it out ensuring that the parts can be accessed. |
| DM1.4 | Input: Navigation filling the entries. |
| DM1.5 | Output: No errors |
| DM1.6 | Normal |
| DM1.7 | Whitebox |
| DM1.8 | Performance |
| DM1.9 | Integration |
| DM1.10 | Results: This test was successful as the UI was able to send information to the MySQL data base. |

|  |  |
| --- | --- |
| DM2.1 | **Dive Master Test 2** |
| DM2.2 | This test will ensure that it can send the information to the server to be queried. |
| DM2.3 | We will send a series of test queries to the main SQL server and make sure that no errors are given to our app upon making the push. |
| DM2.4 | Input: Entry information from the app. |
| DM2.5 | Output: Entries show up in the SQL table on the. |
| DM2.6 | Normal |
| DM2.7 | Whitebox |
| DM2.8 | Functional |
| DM2.9 | Unit Testing |
| DM1.10 | Results: This test was successful as the data sent was the same as what was stored in the MySQL database |

|  |  |
| --- | --- |
| DM3.1 | **Dive Master Test 3** |
| DM3.2 | This test will ensure that the dive master can retrieve and create a barcode from the information from the server. |
| DM3.3 | We will attempt to retrieve the information from one of the test data and make sure that a QR Code can be generated and displayed given another QR Scanner. |
| DM3.4 | Input: Information from the server. |
| DM3.5 | Output: A QR Code containing the information retrieved from the server. |
| DM3.6 | Normal |
| DM3.7 | Blackbox |
| DM3.8 | Functional |
| DM3.9 | Unit |
| DM3.10 | Results: This was not able to be fully implemented to the ability that we wanted it to be. |

|  |  |
| --- | --- |
| DM4.1 | **Dive Master Test 4** |
| DM4.2 | This test will ensure that the information from the QR code matches the queries of the Pi Server and what was put into the Dive Master information. |
| DM4.3 | We will make sure the Dive Master information on the QR code matches what we would be able to pull up if we pulled the data on the server. |
| DM4.4 | Input: QR Code |
| DM4.5 | Output: The information from the Dive master in readable form. |
| DM4.6 | Normal |
| DM4.7 | Blackbox |
| DM4.8 | Functional |
| DM4.9 | Unit |
| DM4.10 | Results: We were not able to achieve this in the environment we had nor were we able to implement this test or functionality. |

|  |  |
| --- | --- |
| UL1.1 | **User Log Test 1** |
| UL1.2 | This will test that information on the Dive Logs can be added to the queries along with no information for where QR Code is going to scan for and fill in later. |
| UL1.3 | We will fill the information out on the Dive Log and ensure that something can indeed be save successfully and error free. |
| UL1.4 | Input: Information in the entries |
| UL1.5 | Output: A savable data entry to the SQL database with the signature areas as NULL or blank. |
| UL1.6 | Normal |
| UL1.7 | Whitebox |
| UL1.8 | Functional |
| UL1.9 | Unit |
| UL1.10 | Results: We were able to save the information to a SQLite database that is save to the local Android device. |

|  |  |
| --- | --- |
| UL2.1 | **User Log Test 2** |
| UL2.2 | This test will ensure that the information can be retrieved from the queries on the phone and in readable form. |
| UL2.3 | We will call the information by the press of a button that can get the information from the database and display it in readable form. |
| UL2.4 | Input: Call for the Log entry |
| UL2.5 | Output: The information from the Log entry |
| UL2.6 | Normal |
| UL2.7 | Blackbox |
| UL2.8 | Functional |
| UL2.9 | Unit |
| UL2.10 | Results: Information was received from the database. |

|  |  |
| --- | --- |
| UL3.1 | **User Log Test 3** |
| UL3.2 | This test will ensure that the log can be edited later for any reason. |
| UL3.3 | We will test the capabilities of editing the entries by opening and changing some of the entries except for those who require information from a QR Code Scan |
| UL3.4 | Input: Edit button to entry and information to change |
| UL3.5 | Output: Updated entry in the SQL database on the phone. |
| UL3.6 | Normal |
| UL3.7 | Blackbox |
| UL3.8 | Functional |
| UL3.9 | Unit |
| UL3.10 | Results: We were able to make the dive logs editable for later modifications. |

|  |  |
| --- | --- |
| UL4.1 | **User Log Test 4** |
| UL4.2 | This test will ensure that we can scan a QR Code from a Dive Master and get the required information from it for the Dive Master signature. |
| UL4.3 | We will be testing the ability to scan a QR code and have that information appear in the entry fields to be save later. |
| UL4.4 | Input: QR Code |
| UL4.5 | Output: Information into the fields for the Dive Master signature |
| UL4.6 | Normal |
| UL4.7 | Blackbox |
| UL4.8 | Functional |
| UL4.9 | Unit |
| UL4.10 | Results: We were not able to test this as the environment we had was not capable of this. The functionality was there but since we were using an Android emulator, the camera function was more difficult to use. |

|  |  |
| --- | --- |
| FT1.1 | **Full Test 1** |
| FT1.2 | This test will be used to see if the application can work from start to finish. |
| FT1.3 | We will utilize two phones and have the app installed; one will be used as the Dive Master while the other as a normal User. We will see if everything works as it should as the expected results for our project. |
| FT1.4 | Input: Dive Master Information on one phone and User Logs in the other phone. |
| FT1.5 | Output: Dive Master QR Code that when scanned will fill in the information on the Users dive information. |
| FT1.6 | Normal |
| FT1.7 | Blackbox |
| FT1.8 | Function |
| FT1.9 | Integration |
| FT1.10 | Results: We got a working prototype in the Android Studio Emulator that we could construct a demo for and have the demo recorded and ready for the Expo. |

**Test Case Matrix**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Normal/ Abnormal** | **Blackbox/ Whitebox** | **Functional/ Performance** | **Unit/ Integration** |
| **PS1** | Normal | Whitebox | Functional | Unit |
| **PS2** | Normal | Whitebox | Functional | Unit |
| **PS3** | Normal | Blackbox | Functional | Unit |
| **DM1** | Normal | Blackbox | Performance | Integration |
| **DM2** | Normal | Whitebox | Functional | Unit |
| **DM3** | Normal | Blackbox | Functional | Unit |
| **DM4** | Normal | Blackbox | Functional | Unit |
| **UL1** | Normal | Whitebox | Functional | Unit |
| **UL2** | Normal | Blackbox | Functional | Unit |
| **UL3** | Normal | Blackbox | Functional | Unit |
| **UL4** | Normal | Blackbox | Functional | Unit |
| **FT1** | Normal | Blackbox | Performance | Integration |