5 Testing

Testing is an extremely important component of most projects, whether it involves a circuit, a process, power system, or software.

The testing plan should connect the requirements and the design to the adopted test strategy and instruments. In this overarching introduction, given an overview of the testing strategy and your team's overall testing philosophy. Emphasize any unique challenges to testing for your system/design.

In the sections below, describe specific methods for testing. You may include additional types of testing, if applicable to your design. If a particular type of testing is not applicable to your project, you must justify why you are not including it.

When writing your testing planning consider a few guidelines:

- **Is our testing plan unique to our project? (It should be)**
- **Are you testing related to all requirements? For requirements you're not testing (e.g., cost related requirements) can you justify their exclusion?**
- **Is your testing plan comprehensive?**
- **When should you be testing? (In most cases, it's early and often, not at the end of the project)**

5.1 Unit Testing

What units are being tested? How? Tools?

In our testing, we will be testing a few metrics. One of the major metrics in which we will be testing is the response time of the dog to the PTSD signal sent to the dog device. This can be measured by timing the delay with a stopwatch from when the signal is applied to when the dog shows a physical reaction.The forced signal will be applied via phone or watch which will be connected to the dog device. Once this has been proven, we then plan on testing the dogs reaction time to go from the forced signal to going to assist the veteran. This once again can be tested by forcing a signal via phone or watch to the dog device and beginning measurement via a stop watch from the time the signal is sent to the time the dog goes to run its routine with the veteran. Both these tests will be measurements of time (in seconds most likely) which will measure the time it takes our device to function as its intended when interfacing with the human and dog. When it comes to the dog device battery life, we plan on testing the lifetime of this device by using a voltmeter to measure the nominal voltage of the battery over the span of a given time interval. The reason for such a test is to make sure that the lifetime of our device is sustainable over a reasonable period of time that this device will be utilized. We will use time functions and battery voltage sampling with different sample rates of refresh checking the heart rate of the watch device to ensure we dial in the proper battery life specified by the client. We will run manual tests on hardware to test vibration for physical feel to ensure vibration motor is properly soldered to device.

For the software, we plan on implementing multiple dart unit test to test various use cases and functionality. To test a use case, our test will simulate a user and ensure that the flow of use case can be accomplished, for example, if a user wants to navigate to the home page from the dog's setting page, we will provide our test case with the setting page and have it simulate tapping the necessary buttons that would navigate the user to the home page. We will also use mock objects to test various functionality, such as sending and receiving and storing data between all three devices involved in our system.

5.2 Interface Testing

What are the interfaces in your design? Discuss how the composition of two or more units (interfaces) are being tested. Tools?

Our interface testing will be testing how the devices interact with each other. The tests for the user fitness device have already been tested through their respective manufacturers. Our app will use mock objects to test (in flutter/dart this done with Mockito) to test edge cases (the data is smaller or greater than expected data), and test various workloads (large/small amounts of data coming in or sending out).

When the hardware device is being tested, we will verify the physical impulse of vibration motors. We will test bluetooth connection by pinging back and forth over the bluetooth connection, as well as executing all functions that are supported over our bluetooth protocol to ensure all sensors are functioning properly. We will also send tests to the battery to send it under load by engaging all vibration motors to draw current and measure how hard the voltage sags under load. If the battery doesn't pass the test, it will indicate if it needs to be replaced. We will also test the optimal sampling of the heart rate to not kill the battery of the watch too quickly. This will be done using time functions to save time differences and testing different sampling rates and the value of the battery voltage before and after these sample rate tests.

5.3 Integration Testing

What are the critical integration paths in your design? Justification for criticality may come from your requirements. How will they be tested? Tools?

Our plan for this is to emulate the app within android studios (if possible run on an actual device), and attempt to connect to our prototyped hardware. We will provide both the devices with various debugging tools, to ensure that we can see the connection is established, and data can be sent and received between the two devices.

The most critical point of integration is on the bluetooth protocol. To begin, we will write mock response code and testing on each device to test when coding that nothing breaks before integration. During integration we will make tests of every function in our API that will test communication between the two devices fully to validate it is up to spec.

5.4 System Testing

Describe system level testing strategy. What set of unit tests, interface tests, and integration tests suffice for system level testing? This should be closely tied to the requirements. Tools?

All functions will have mock responses for each function testing to ensure that code doesn't break when changes are made when not connected to a hardware device. There will be a similar test protocol for the hardware device that will run through all functions provided by the hardware API with mock instructions and responses from the watch. There will be a final system level protocol that will test everything while connected with real commands and responses for full system testing coverage.

5.5 Regression Testing

How are you ensuring that any new additions do not break the old functionality? What implemented critical features do you need to ensure do not break? Is it driven by requirements? Tools?

Our unit tests will be responsible for ensuring working functionality does not break as we continue to add new additions. We are aiming for a code coverage of 80% (this final number will be provided by client), a strong code coverage will ensure that if a faulty change is made, a unit test will fail indicating that the change has broken something. With all functions being tested, we can make changes and test them before pushing any code to the repository. This will keep our software functioning as expected over multiple iterations of refactoring and adding new functions.

5.6 Acceptance Testing

How will you demonstrate that the design requirements, both functional and non-functional are being met? How would you involve your client in the acceptance testing?
We will be noting our requirements being met by physically/electronically keeping a testing document on us with written requirements from both our team and client noting our progress, success, and details of said requirements. We could also potentially record or take pictures of us testing the requirements to enhance our proof even more. We plan to involve our client in this acceptance testing by notifying them when they have been met at the next meeting via sharing our documentation of said testing and also verbally explaining it to them. Given that we will require service dogs for our training, we also will be involving them physically by having them with us when we test our device on the dogs to make sure we are handling them properly.

5.7 Security Testing (if applicable)

We have a closed loop system containing non-sensitive data, so this isn't a high concern. Functionality is of highest importance.

5.8 Results

What are the results of your testing? How do they ensure compliance with the requirements? Include figures and tables to explain your testing process better. A summary narrative concluding that your design is as intended is useful.

The result of our testing is a prebuilt code base to test all usage cases of our system at the press of a button. This can be ran at any time when coding, providing instant feedback to errors, reducing problems from faulty code being unchecked. The requirements will be programmed into the testing to ensure if anything breaks and a requirement is not being fulfilled, it will be caught and handled before becoming an issue. Our testing will ensure that the design of the arduino based vibration system, and the apple watch based sensing mechanism, will be able to catch cases of PTSD attacks whenever the dog isn't focused or aware. This will dramatically reduce the number of flare ups, and the severity of their impact when they do occur. It will provide our users with a better mindset everyday knowing they have an extra layer of protection from their disease when going throughout their day. This will profoundly improve the mental health of our users and dramatically improve the quality of life of our users by reducing all of the negative impacts of PTSD more effectively than any other available solution on the market.

Test first approach:

Determine function to add -> Write test of function output -> Write code to implement function ->

<- Repeat <- Refactor <- Run Test Code <-