Software Testing Plan

03/26/21

Team Name - Efficient Tester

Project Sponsor: Dr. Tara Furstenau

Team Mentor: Tomos Prys-Jones

Team Members

Tyler Conger

Miguel Villareal

Xiaobai Li

Bailey Mauss

Yihao Lu

Table of Contents

1.Introduction

In the United States, laboratories routinely conduct many tests for infectious diseases. At present there is the added pressure of testing for COVID-19. Today, we are testing more samples at present than this time last year prior to the pandemic. While current testing effectively detects positive cases of COVID-19, it can be wasteful with regards to reagents, laboratory consumables, and time. With the ongoing COVID-19 outbreak and the reopening of public places, people are returning to work and school and coming into contact with more people. This increased contact could drastically increase the number of people who need testing. It has revealed a huge challenge to our ability to detect pathogens. Traditional testing is effective at detecting pathogens, but if the prevalence of infection is low in a population, many reagents and consumables are wasted, as the majority of tests will be used on healthy individuals. In these situations, it can be more effective to group samples into a single test tube and look to see whether any of the individuals are infected. If the pooled test is negative, then the researcher has saved n – 1 tests (where n is the number of individual tests that were grouped together). If the pooled sample is positive, this group is subdivided into smaller pools and the process is repeated. This method was developed by Robert Dorfman, and is used by our sponsor Tara Furstenau and her research group who help laboratory testers to improve efficiency.

For rapid testing, our clients have developed their own protocols for group testing on a 96- or 384-well plate. The positive samples can be quickly detected in two or three steps. For a 96-well plate, we first create six pools with 16 test samples in each pool. After passing the first test, we select the pool with a positive result, divide each pool into, on average, two pools and conduct the second group test. In each step the number of samples that are pooled together decreases, with individual samples being tested in the final step. An example of this three-step process would be: a) dividing the 96 well plate into six pools of 16 test samples, b) If a pool comes back positive after the first test, the pool is divided into two pools of 8 samples, c) Finally, if the second test comes back positive, the individual samples for each positive pool are tested individually. While the example above uses pools of 16 samples per test tube, followed by 4 and then 1, the optimal pool size may vary depending on the expected infection rate. After the second population test, samples from each positive pool are tested one by one. While this group testing method is known and effective at detecting infected individuals, we believe that improvements could be made to the clarity and timing of the process with a workflow assistant

application. The app would help technicians decide how to subdivide the 96-well plate, keep track of positive tests and assist in recording results.

During software testing, developers try to find defects in the software and fix it. That ensures the whole software works properly. During the process of defect fixing, the quality of software will improve rapidly. When we finish the final test and no more defects are found, it proves that our software is worth delivering.

In order for the whole process of the application to be correct, we will test every single unit functionality. In this part, the main pieces we intend to test are the login system, well plate interaction, protocol creation, and user settings.

Our integration testing focuses on the interactions within the software, as the results of each component interaction determine whether the software will work properly. The proper functioning of the interactive function ensures that the software can provide the user with a picture of the well plate with the correct information.

For usability testing, we need to test our product with users to make sure it's easy to understand and use. Since our customer background is a lab team, we will test for different user types. Users will have no experience with a new application. We want to provide potential users with as little information as possible to ensure that they can use the application with little explanation. Based on the test, we can ensure our sponsors get everything they need when we deliver the product.

Our test plan helps us find problems and bugs in the software, so we have time to fix them before delivery. This is important for developing a fully functional and deliverable software.

2. Unit Testing

What is Unit Testing?

Unit testing is a type of software testing where individual units or parts of software are tested. This is done in order to validate that each unit of the software code performs as expected. A unit might be an individual function, method, procedure, module, or object.

Unit testing will allow us to safely change our code in the future, knowing that the various functions are still working as expected. If a function is changed and suddenly returns a value that we did not expect, there may be further significant consequences in the code. In the long run, this also makes code maintenance easier, because we will know exactly which changes immediately break which features, rather than guessing and hoping that we can discover which changes caused a function to return an incorrect result. This is the goal of unit testing. We want to make sure that each individual code segment can work individually to ensure that they can work properly when assembled together. If a piece of code does not work, the problem will be isolated as a single segment or function, which is easy to fix.

Units to be tested:

- Register procedure
- Login procedure
- Forgot password procedure
    - Email input
- Protocol creation
- Well plate interaction
    - Adding user notes
    - Saving experiment status
- Settings update

Unit Testing Plan

Register procedure:

Description: The user needs to enter a valid email address and password to create an account. The length of the password must be 6 characters or more. If the length does not meet the requirements, the account registration will fail. The user also needs to ensure that the password is the same as the confirmed password, otherwise the account registration will fail.

Example input:

-Email:

Valid: 123ab@nau.edu

Invalid: 123ab

-Password:

Valid: 123456

Invalid:123

Expected result:

-The user is brought to the landing page to login If the registration is successful.

-The information filled in by the user during registration will be stored in the database so that the user can log in.

Login procedure:

Description: The login system should only allow those that have a stored account in the database to enter. If an invalid account or password is entered, the user will be prompted that the account or password is incorrect. Valid user sign-in should bring the user to the homepage.

Example input:

-Email:

Valid: 123ab@nau.edu

Invalid: 123ab

-Password:

Valid: 123456

Invalid:123

Expected result:

-User is rejected if login input is not stored in database or password for email entered is incorrect.

-User is brought to the homepage if email and password are correct.

Forget password procedure:

Description: The user resets the password by entering a valid email address. If the user enters an invalid email address (not in the database), the system will not send an email.

Example input:

-Email:

Valid: 123ab@nau.edu

Invalid: 123ab

Expected result:

-The user will receive an email from the system to reset the password.

-The user can successfully reset the password through the link in the email.

-The system will prompt that the email address is invalid, If the email address entered by the user is invalid (the email address is not in the correct format or the email address is not in the database)

Protocol creation:

Description: The user should be able to create a new protocol by entering the number of samples, well plate type, protocol name, and expected positivity rate. This protocol should then be saved in the database and the protocol should appear on the page. The user should not be able to submit a protocol with any invalid inputs.

Example input:

-Samples:

Valid: 96

Invalid: Any negative number

-Well Size:

Valid: 96

Invalid: Any input not 96 or 384

-Name:

Valid: COVID-19 Test

Invalid: No invalid inputs expected.

-Positivity %:

Valid: 3.00

Invalid: Any non-percentage number

Expected result:

The protocol will be saved to the database and should immediately become usable in an experiment. The list of protocols will update to reflect the new protocol.

Well plate interaction:

Description: The user is able to interact with a well plate image to reflect the actions taken in the actual experiment, such as taking samples, combining them into a test tube, and indicating which test tubes returned positive. The user should be able to record notes at any stage of the experiment. The user should also be able to save the experiment at its current state at any time.

Example notes input:

Note: “Well B3 is cloudy.”

Expected result:

When the user submits a note, the note should be saved along with the user who recorded it, and the state of the experiment at the time the note was written. This note should be saved within the experiment and able to be viewed later.

Experiment saving:

At any point in the experiment, the experiment should be able to be saved in its current state (including wells that have already been pipetted, tubes that are completed, notes taken, positive results, etc.) and can be returned to later. Should the user encounter network issues while attempting to save, the page should wait until network access is restored to save and close the experiment to prevent data loss.

Setting update:

Description: Users can view personal information (first name, last name, uuid, email) and modify their email address on the settings page.

Example input:

-Email:

Valid: 123ab@nau.edu

Invalid: 123ab

Expected result:

- The setting page will display the personal information of valid users.
- The user can modify the email address, invalid email addresses will be rejected.

- The new email address will be stored in the database.

3.Integration Testing

Our integration testing will include a few major linkages that must be tested. First, the interaction between the Ionic front end and the database through the Django API. Secondly, the connection between the Ionic front end and the Django front end that creates the well plate image display for the user. And thirdly, the interaction between the Django well plate image and the database back end. Each of these component connections represents an important linkage between major pieces of the application, so it is important to verify that none of these are faulty in any way. A faulty connection to the database may allow the user to add information that they should not be able to add. To avoid this, it is important to sanitize user inputs, meaning ensuring all data coming in is allowable and valid. The connections will also be important to test, as if a faulty connection exists, it means that potentially a user could be redirected to a page they should not be allowed to view, or could be disconnected from the application.

First, the sanitization of items being added to the database should be tested through some simple cross-site scripting (XSS) attacks that would see if MySQL code can be injected into the database. This type of thing can occur during data exchanges between the separate modules and it is important that these take place correctly. Thus it is important to verify that all data is exchanged between pisces within our project properly and safely.

3.1 Features that will be tested in our Ionic Application

These features tested will be the all of the main features of the application's frontend that has interaction with any piece of the backend database, and anywhere that data is transferred between the two pieces of the project.

- Login
    - Users should have the ability to login to the application and should be granted the standard permissions level.
- Logout
    - Users should have the option to logout of an account at any time
- Register
    - Users have the ability to register an account using a unique email address. New accounts will be stored in the database.
- Forgot Password

    - A user with an existing account will be able to enter their email address and use the forgot password functionality that will check if they exist in the database and send them the appropriate link to redeem a new password.
- Create a Lab Group
    - Users will be able to create their own lab group, which will be saved in the database.
- Add to Lab Group
    - Lab Group creators will be able to add other users on the platform to their lab group via an invite system. This will connect to the database first by checking if the user exists and then adding them to the appropriate group.
- Create a Protocol
    - This will add a new protocol to the database, it must contain the appropriate information that the user has entered.
- Create an Experiment
    - This will add a new experiment to the database, it must contain the appropriate information that the user has entered.

3.2 Features that will be tested in Django Application

These features that must be tested in the Django application represent the interaction between the Django application that will hold the well-plate interaction and the database where data must be stored. It will also be the linkage between the main Ionic application where the Django piece is opened and closed.

- Open the Django Application page
    - On the main Ionic page the user will select an experiment and in doing so will transition from the Ionic application to the Django application, this transition must be seamless to account for user experience.
- Closing the Django Application page
    - As the user selects save and exit the current experiment it is important that the page closes and the user is redirected back to the ionic page they originally began on.
- Saving an Experiment
    - Because this Django application is separate from the database to Ionic linkage the requests to save data must be handled effectively by AJAX.

4.Usability Testing

What is it?

Usability testing focuses on the interactions that are made between the end user and the system itself. The goals of usability testing are to ensure that end users can effectively access the functionality provided through the software system with little to no errors on the user or software sides. It works by placing one or multiple end users with your software system and having them go through the application as it would be used on an everyday basis to see what kinds of problems future end users may face, or to point out any slight improvements that could be made to the system before full deployment. This can be done in a variety of ways and, depending on the specific system, may be more useful than other forms of testing.

Plan For Testing

Before we get into our testing regime we want to discuss why we are choosing to do usability testing in the fashion that we are. The first reason is user background. As a team we understand that there are going to be a variety of different people with different positions using our system and we want to ensure that it will be viable no matter the position of the end user. Our client runs a lab with a number of lab users that range from graduate students to undergraduate students with no experience with any type of lab settings before. Our next reason is the novelty of our product to the end users, we know that our application is going to be a new software that neither graduate or undergraduate students have experience with, but we want there to be no issues during the transition since many of the current lab users are doing everything manually with their lab notebooks. These are just some of the points that we dissected when discussing how we wanted to go about our user testing and the manner in which we have each end user participate.

Testing Regime

For our usability testing regime itself we decided to split it into two separate days, one at the beginning of the week and another at the end of the week. We decided to go with this route because we believe that whatever data or observations we gather from the first testing run can be used to make improvements during the rest of the week and then have a final run through

with another end user after those changes have been made. The first usability testing session will be held with two undergraduate lab users from our clients lab who have experience working in a lab setting and dealing with the data our product will be using. Our second session will be with a technical user who has no experience in a lab setting but has experience on the software side so that they can point out things that we may have missed in the design of our system. Now, for our actual testing regime itself and what we plan on having the end users do during the session itself:

- Users will be introduced to our application and given a brief description of what it is our application is intended to be used for.
- We will have the user start on the landing page where the options for register or login are listed and will ask them to create an account.
    - Once an account has been created with valid credentials we will have the user login to the application.
    - User will be taken to the main page of application at which point we will ask them to logout and “forget their password”
    - This will entail an email being sent to the user where they will fill out a form to change their password and redirect back to the login page to enter the application once again.
- The user will enter back into the main page where we will ask that they create a lab group through the application.
    - After the lab group has successfully been made we will ask that they create a new protocol which will be used by their group.
    - We will then ask the user to create an experiment based off of the protocol created.
- They will be taken to the experiments page where the user will be asked to create an experiment in a similar fashion in which the lab group and protocol were made.
    - We will then ask to head to the well plate interaction page where the user can record data on the created experiment.
- The user will be taken to the well plate interaction page
    - Lab users will be asked to “pipette” the “tubes” in the given well plate so that they can move on to the following step.

- They will also be asked to take down notes throughout the process so that the data can be stored with the current experiment.
- We will ask the user to save the current state of the experiment and leave the current session through the middle of the experiment. After they have left we will ask that they revisit the interaction page to see the saved state of the experiment and continue through it.
- Once completed the user will be asked to save the experiment and logout of the system after returning to the experiments page.

That will be the entirety of one session with our end users. As stated, we plan on having two sessions so that after the first we can make adjustments where needed to improve end user usability. We will have the two undergraduate end users work through the first session together because as a team we feel that it will be better to get more out of the users and to make it a more comfortable experience. By doing so we hope that we can analyze how an everyday lab user might go about using the application and what kinds of changes we can make to have it optimize their time while using our system. The last end user, which is the technical user, will go last because we feel that after certain adjustments have been made concluding our first session will allow the technical user to go into more detail on what our system may be missing or could be added to better suit a user who has no experience in a lab setting in the first place. After this is completed we will take what we gathered from the technical user and implement where we see fit to better suit our application. From here we believe our product will be fully ready for deployment and for use with our client.

5.Conclusion

In conclusion, we believe that we will get a fully functional, usable and bug-free software after user testing. In order to produce an error-free and easy-to-use product for our customers, we will perform several forms of testing: unit testing, integration testing, and usability testing. Running these tests on our software will ensure that our product works the way it should. We will improve the software based on user feedback.

For unit tests, we will test the major functions of the major components of the software. It is important to ensure that errors or spurious data do not enter our workbench, as well as to ensure that each part creates and processes data in the correct way. For integration testing, it focuses on detecting the interactions between each component in the software. To make sure all the components work properly with each other. Our usability tests will focus on our software that will be directly interacted with by users. If every part of our testing goes smoothly, then we can conclude that our product is reliable and ready to use. Otherwise, timely feedback on software bugs will allow us to fix them before the deadline. We can know the satisfaction rate of our products. On this basis, we can make the necessary changes to improve our product as the deadline approaches. By collecting and analyzing all test data, we can understand the satisfaction rate of our products. Based on the data we collect from each test, our products can also meet the needs of our end users and customers. As we move forward, we may encounter some minor obstacles, but we are confident that we will reach its full potential before delivering to customers.