Software Testing Plan

Version 1.0
19 March 2021

Team DigiFolio
Dr. Andy Wang
Fabio Santos

Burbank, Logan(lead)
Braudaway, Jackson
Chen, Kailin
Marschel, Parker
Yang, Zhenyu

Table of Contents

1. Introduction

Throughout the implementation process this last semester, our team here at DigiFolio have been building an application for our client, Dr. Wang. The software is primarily focused on data collection and visualization. The primary purpose and reason behind this application is to help automate the collection of career data from students, and visualize statistics regarding that collected data. This is in order to help Dr. Wang see how the student body is doing related to their career success.

In order to ensure we can deliver a sufficiently working application, we will need to design and implement a plan for testing the application, which is what we will be focusing on. Our plan will consist of testing in three different areas:

- Unit Testing
- Integration Testing
- Usability Testing

Looking at the unit testing, our main goal with this section is to set up a plan to test each individual module's functions. As our application has separated many of the modules into the different pages that can be accessed, this comes down to testing each of our web pages to ensure that their functions work correctly.

Moving on to the integration testing, our plan here is to focus on how our modules interact with one another. This will focus on more specific functionality of our system, rather than a general sense of every piece on each page. Ultimately, specifics of this part revolve around when modules are using other modules' APIs in order to function.

Looking at the final section, for usability testing, our plan is less code based and more of a plan of action. This section will delve into the details of how we intend to test our application when it is being used by an end user. This testing will allow us to see and understand how users interact with our current user interface, and we can make notes and adjustments to allow for better usability overall.

Ultimately, we have decided to go with the plan mentioned, involving the three different sections, due to the data-central nature of the application. Overall, the functionality most users will get out of the site will involve simple inputs and outputs, and this is where we hope to focus most of the testing through implementing our plan, as this is the place where we are most likely to have errors show up.

With our software, the dangers of not testing are not anything drastic, but it could put the database into a state where it is unusable, and that could render the site useless.

However, as our site is mainly a data tool, the possible outcome of that risk is minimal to user safety and data safety, as most of the data is also publicly available.

2. Unit Testing

Unit testing is done to ensure that the components and modules of the application are working properly and returning the desired functionality. Making sure that each component functions correctly on its own is essential when it comes to combining each unit together in order to have an application that works as needed. By performing specific unit tests to different functions and operational procedures, any issues can be resolved before the application moves forward, thus providing a cleaner system. For our application we plan on testing the individual functions and procedures belonging to each of the different main modules of our application to make sure they work as they are intended to. By doing this we will be able gain a clear understanding on the behaviour of the individual units that make up our application and fix issues that arise ranging from inappropriate handling of different types of input to problems that could crash the application. For this project the individual modules that will be tested are the sign-in module, user search module, user profile module, dashboard module, and our database. Each of these modules must work as they should by themselves before they can work together to form the overall functionality of the application as a whole.

2.1 User Searching

In our application this component is meant to allow a user to look up others who are pursuing degrees in the same or related fields. Entering the name or email of a person should provide a list of those who are registered users and allow the individual making the search to view their profile. The person making this search should not have privileges to change or modify that user's profile but rather only view what is contained on it. To test this unit we first plan on providing the search form with random inputs including characters such as ‘?’, ‘/’, ‘..’, etc. and different phrases. Also, because this module needs to access the database we need to provide different inputs that may allow a user to access parts of the database that they should not be able to. More specifically, this would be done by manipulating the SQL queries that parse through this database with different inputs such as “admin’--”, or “‘OR 1=1’--” which if not handled correctly could give the user admin privileges to the system or trick the query into retrieving all information because ‘1=1’ results to true and the query would no longer treat the rest of itself after because ‘--’ is a comment indicator. It is little things like these that could breach the confidentiality, integrity, and even the availability of our system.

2.2 User Profile

The user profile module of our application is meant to provide the user with a detailed representation of their goals and milestones. Here the user should be able to add/modify things on their profile to keep it updated and store personal information as well as information that will be viewable to other people when they use the search module. Our plan for testing this unit will include setting visibility permissions and making sure others who search for that user cannot see the elements set to private and only can see those set as public. In addition we will be implementing testing to make sure invalid input(s) does not get added as a milestone to a user's profile. We will also be testing to make sure that every milestone an individual has on their profile is displayed as well. In order to do this we will be providing bad random input such as words composed of a random sequence of special characters, i.e. "?//[{@&", etc. We will also create users with several milestones to ensure that parsing through a user's milestones collects each one. We also will need to test and make sure if a user has 2 of the same milestones that they are not repeated on their profile. Overall, our unit testing for the users profile will consist of creating different test users, where some of these users have multiple milestones, and where some try to input invalid data to add to their profile. By doing this we will be able to guarantee that a users page cannot have data that does not make sense and also that whatever information they want to display is displayed to others and sensitive information is only viewable by that user.

2.3 User Login/Register

Within the user interface module, we must test that users can register and log in. As a user comes to the home screen, they will be prompted to Log In using the top button. Once they are here, they will have the option of continuing with logging in, or registering. We will need to ensure that when the user is logging in, the email field will check to be an email. If something else is passed in, we have to reject the attempt at signing in before it ever involves our database lookup. The password, however, can be any string, so that will not be checked for any format. The other primary testing for the login, at least, is to ensure it finds an account that exists, and this can be tested by passing in known emails and unknown emails and seeing the known ones give different results.

From the login page, there will be an option to register. This page is similar looking to the login page, except that it is much more important for testing. While the login page is basic in its testable pieces, the registration page must test the information that is put in to ensure our database remains consistent. This includes checking that an email is put in, which can be tested by sending in strings lacking the ending for an email string. The next step is to ensure that the password field is being matched to a repeat field for it, as we will not be displaying the password on the screen. This allows the user to ensure they are not mistyping the password, and lets us confirm that is actually what they want the password to be. Considering the rest of the options are selections rather than raw string inputs, the rest of the fields do not need to be tested for any kind of format. If the email and password are not tested, though, our database could end up with all kinds of inconsistencies.

2.4 User Dashboard

Student users, once logged in, will have a home screen (dashboard displayed) which will contain a navigation bar. The bar should have options such as “Dashboard” to return the user back to the home page, “Portfolio” that will take them to their own profile page, and a search bar to look up other students' public profiles. Going back to Dashboard, once the users click the button ’Dashboard’, the dashboard will allow the student to view their own goals and progression toward them as well as being able to report an issue to administrators. Furthermore, he is able to search other alumni profiles as well as report a data problem to administrators through the buttons on the page. Except the student users, faculty will be the most different as these users will have the highest level of privileges on the system. Upon login, a faculty user will also have the same navigation bar. However, when searching alumni or students the faculty member will be allowed to make changes to their profiles including changing a graduating students privileges to that of an alumni, or edit any other parts of a profile necessary. The dashboard for faculty members will also provide them with the option to view student profile progress or overall department progress. If the user goes to their profile page they will be able to logout or modify their profile (username, password, etc.).

2.5 LinkedIn Data Handler

The LinkedIn data handler module will be a core module to the system, as it is what is providing the data for the rest of the system to use. This data will typically be scraped from student LinkedIn profiles, and when involving web scraping, there is a chance at running into errors. For this reason, the LinkedIn data handler must be tested at each of its basic functions to ensure everything remains consistent. This ultimately means that first, the URL that is passed in as the user page must be checked to ensure that it is a user profile, and not some other site. From there, it must pull the career experience data from LinkedIn and ensure that the data is actually present before continuing. Otherwise, the system may try processing null data due to an error in the scraping, whether it be due to a bad URL given or a lack of experiences. Finally, the system must test that the data that has been pulled can be classified into our database categories, Company and Milestone Type. Once all of this has been checked, the system will then be allowed to insert the data into our database. This will help keep inconsistent scraping from causing errors and inconsistencies, as well as ensure the LinkedIn module is working properly.

With all of the pieces of our unit testing laid out, we feel that we have a good plan for handling each part of our program. These tests will only ensure that the individual modules are working properly on their own, however. In order to ensure the modules work well with one another, we move on to talk about integration testing.

3. Integration Testing

Integration testing is connecting all the components together to make one system that works together. Once the components are connected together, tests can be made to check if they are acting together as expected. The main focus of the integration testing is to check the communications between the user's inputs from the frontend, backend and database. These tests will be able to show the interactions between each module as well as show our expected results from each test.

3.1 Account Login

Users on this website will be able to create their own account in the system. Once an account is created the user will be able to login with an email address and password that will be kept in the database. This part of the system is able to keep track of who is logged in as well as all the information the user enters while logged in. Overall, the registration of a user is the starting point into the rest of the system and connections between the rest of the modules.

Integration Test	Description	Expected Result
User Registration	A user can register an account	- A user will be able to enter all information needed to create an account - A user's information is then added and entered into the database and will be able to login with Email and Password
User Login	A user can login to their account that that	- A user can login with their Email and Password

	registered	- A session will be created where the browser keeps track of who is logged in - A user then has their own profile page they can add personal information to
User Logout	A user can logout of their profile	- A user can logout of their profile they logged into and will have to sign in again to access their account - The session created in the browser for the user will end after sign out

3.2 Dashboard

Both registered and non registered users are able to access the dashboard module of the website. Anyone is able to choose the fields that are incorporated into creating the graphs. These graphs will display information from the database that is requested by a user. Expected results are that the information from the database will be displayed correctly in the graph requested.

Integration Test	**Description**	**Expected Result**
Pie Chart Display	A user can request a Pie Chart	- The information requested will be taken from the Database and displayed as a Pie Chart
Bar Graph Display	A user can request a Bar Graph	- The information requested will be taken from the Database and displayed as a Bar Graph
Pyramid Chart Display	A user can request a Pyramid Chart	- The information requested will be taken from the Database and displayed as a Pyramid Chart

3.3 Profiles

Inside each user’s profile a user will be able to add, edit and delete a milestone. A milestone is entered by adding what type of job it is as well as the name of the company. The milestone will then be put into the database after the user has entered all the information and hit submit. The expectation of our system is to correctly input each individual user’s milestones. Another expectation is also to correctly display each user’s milestones on their profile page. The communication between the modules and the database are extremely important and are expected to work smoothly throughout the system.

Integration Test	**Description**	**Expected Result**
Student User Adds a Milestone to their Profile Page	A student can add a new milestone to their experiences in their profile page	- New milestone is added to the user’s profile page - The correct milestone is retrieved from the database and is viewable by the user
Student User Edits a Milestone in their Profile Page	A student can edit an existing milestone in their experiences in their profile page	- Existing milestone is updated in the database with the new information the user has entered in each field - New information is then updated on their profile page for the user to see
Student User Deletes a Milestone in their Profile Page	A student can delete an existing milestone in their experiences in their profile page	- Existing milestone is deleted in the database - The deleted milestone is no longer visible on the user page because it no longer exists in the database
Student User Changes the Visibility of their Profile Page	A student can change the visibility of their own profile from PUBLIC to PRIVATE and vise versa	- The visibility is changed in the database after a button is clicked - The current visibility of

		the user’s profile will be seen from their page

3.4 Search

The search module is able to search the entire database for a keyword that displays any public profiles and display their pages in a table. The expected response is that private profiles are not visible and that only public profiles are displayed. A user is then able to view any public information that a user has allowed to be public on their own page. Overall, the search module has to communicate with the database to correctly display information that has been allowed to be viewed publicly.

Integration Test	**Description**	**Expected Result**
User Search for Profiles	A user is able to search for other profiles	- A list of PUBLIC profiles will appear in the table that match the searched criteria - All PRIVATE profiles will not be shown to Student Users

3.5 LinkedIn

The LinkedIn module is able to search a user’s LinkedIn profile and populate our database with some data from the LinkedIn profile. Once this has been done, the milestones need to be checked for consistency of data to ensure our database module can handle inserting the data.

Integration Test	**Description**	**Expected Result**
Insert Pulled Data into Database	A piece of data can be formatted correctly to be inserted in the database	- A new milestone will be seen on the user profile when scraped - The milestone will fit into one of the four type categories

4. Usability Testing

Usability testing is bringing end-users into the testing process. Instead of testing our website with code as we did in the previous documents, we will test our website by having people use our website. The goal of usability testing is to get feedback on the user interface of our website and better understand if our website is intuitive. With this feedback, we can update our website to make it more user-friendly. Usability testing can also expose bugs in our website we did not find in the unit integration testing, so we can also expose those and get them fixed. For our website, this is a significant testing phase, since the user of our website may not have a plentiful technology experience, we need our website to be intuitive to create an easy user experience for every user.

For usability testing with our website, we plan to have two different testing phases, and ideally, with new participants added for each phase. Our first test phase will be a conversation with individual participants selected by team members. The participants may range from roommates to friends or family members. In our second part, we will work with our client, Dr. Wang, to collect participants from the College of Engineering, Informatics, and Applied Sciences. Our criteria for our participants is as follows:

- Never seen the website: The best feedback we get from the user interface and overall user experience will come from participants who have not yet seen our website. In this way, they do not know how to interact with the features of the website. We want to be able to get feedback from new participants on our website, so we can create a user-friendly experience from the first opening of the website.
- Access to a computer with audio capabilities: To test our website, participants in these sessions will need to be able to access the website, therefore, they will need a computer to do so. In addition, due to the public safety guidelines of COVID-19, we cannot do sessions in person. Therefore, our participants will need to be able to access a computer that has a built-in or external microphone that they can use. This will allow video conferencing with screen sharing capabilities.

For each session, we will provide participants with a user manual, which is attached in the appendix of this document, and they will follow different scenarios and tasks. The tasks will be presented vaguely, just a general description of what they should do, so as not to affect any of their decisions. We will ask the participants to read the manual carefully while explaining their decision. We will have two participants work together in a meeting because it will make them feel more comfortable talking about their options and help us understand that if both participants cannot understand how to complete a task, we designed Where is the main flaw.

To conduct these sessions, we will use the video conferencing software Zoom. With Zoom, we can allow participants to share their computer screens with us so we can see how they interact and tell us why they interact in this way.

Now that we have covered the overall plan for usability testing, we can go into specifics for our website.

Website Testing

In order for the website test to provide valuable feedback, ideally, we will provide a different set of participants for the administrator portal and the student portal. Participants will get a link to our web application. The user manuals that participants receive will be site-specific and have special instructions for the administrator portal and student portal.

After finding the participants who participated in the admin portal test, we will provide these participants with a link to our website. We will also send them any necessary information or attachments required for the test. Participants will then follow the relevant instructions in the user manual regarding administrator tasks.

Participants of the student portal test will also receive a link to the website. They will then follow the student-specific tasks outlined in the user manual.

The results of our usability testing plan will complement unit testing and integration testing, thus providing us with valuable insights on application quality. Usability testing will conclude our software testing process and allow us to make final changes to the website before handing it over to the customer.

5. Conclusion

Overall, our plan can be summarized into the three main sections mentioned above: unit testing, integration testing, and usability testing. With the implementation of each of these sections that has been described, we expect the application to be fully tested in every user function to some degree. Through our unit testing, we will be able to test each basic function to ensure it is working as intended at the core functions of the application. Then our integration testing will take care of checking that our system is working as one system by using consistent data and methods between our modules. Finally, our usability testing will shift the focus to how the user actually interacts with the application, allowing us to ensure a primarily good experience when using our application.

With each section focusing on a different area of interaction with the website, the plan appears to cover any area that may be in need of testing. Through all of these different forms of testing, we are confident that our plan we have laid out will ensure that our application continues to run smoothly and efficiently.