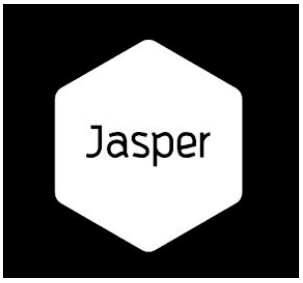
Software Testing Plan

Version: 1.0

2 April 2019



Team
Jasper

Project
Jabulani School Simulation Portal

Sponsor
Dr. Gretchen McAllister

Mentor
Ana Paula Chaves Steinmacher

Members
Karsten Nguyen
Carli Martinez
Ruben Rincon
Jasmine Mitchell

Table of Contents

Introduction

With today's ever increasing diverse population, there is now a new inquiry into how instructors can embrace the diversity of their student body — whether it is between a traditional face-to-face class, or one that is taught online. Current research suggests that diversity in a traditional classroom is a powerful asset, providing that the instructor is sensitive to individual student's backgrounds. However, it can prove difficult to deal with the diversity gap between students and teachers. To allow these teachers to engage with their students respectfully, teachers must know their students and their academic abilities individually in order to be able to respond in a culturally, socially, and linguistically appropriate manner. The best approach for teachers to obtain knowledge for handling specific diversity-related circumstances is by connecting to the experience on a personal and professional level of students with various backgrounds.

Our sponsor, Gretchen McAllister, is a Professor of Education at NAU. While teaching abroad in South America, she contemplated the idea of developing a school simulation portal that could amplify and fully encompass diversity sensitivity in an academic setting. This portal, appropriately named "Jabulani" — the siSwati (Kingdom of Swaziland) word for happiness, is to made to lay the foundation for the future of diversity training in academia. The Jabulani School Simulation Portal is a web application meant to simulate real-world classroom experiences. Its main feature is the creation and assigning of virtual classrooms populated with virtual student profiles. Learning activities based on those students are then assigned to classrooms for teachers-in-practice(TIPs) to complete. The application is mostly data driven so it must be capable of holding data for classrooms and TIP offerings, creating new classrooms, creating student profiles, creating activities and groups of activities, and editing information for most of these elements as needed. Although activities are based on and assigned to classrooms, they are externally created and thus only need uploading capabilities to the site

Our testing plans involve creating and running various tests for our web portal once it is successfully ported onto NAU's IT servers. These tests include unit tests, integration tests, and usability tests for our application. For unit testing, we'll mostly be testing the site's various features to make sure they are properly accepting or adding data. For integrations tests, since we are using Ruby on Rails we must make sure the various gems it uses are compatible with each other, as well as correctly integrating with our database and hosting services. While unit and integration tests will be done using the site, usability

tests only require our client and her coworkers to test the software personally which can be done before hosting everything onto NAU. We will check that users are not completely confused when attempting to find, access, and use any of the site's features. There will not be a need for them to access the database or hosting data on an average basis. The following sections describe all three of these test types in detail.

Unit Testing

For any software application, both Unit testing, as well as Integration testing, are very important as each of them employs a unique process to test our software application. Unit testing allows us to test individual units/components of our software that are small enough to be testable. These will help us at any point in time to ensure our software is working properly at a unit level. For instance, after any minor or major update to our database, we can run our unit tests to ensure nothing is broken by the update before committing and pushing our changes to our GitHub repository, this is sometimes called a "warning system". Similarly, we can use unit testing to help debug our code and assess at which level in our architecture our bug resides, and gain more information about it.

Since testing is built into our Rails framework now with the Rspec gem, we will utilize it to write and build unit tests for several components of our software and run them using the command line. One command can test several tests at once, which makes testing quick and easy. The Rspec gem describes itself as designed where "tests are not just scripts that verify your application code. They're also specifications (or *specs,* for short): detailed explanations of how the application is supposed to behave, expressed in plain English." We believe this will make our tests easy to understand quickly, especially for future developers on this project.

Our Plan

While our integration testing and usability testing will ensure the overall function of our main components like "building a TIP class", our unit testing will break down these central components into small pieces where we can ensure the function of each individual part of it, such as testing the creation and saving of each piece of data within a TIP class related to each table involved.

The functions of our software centrally reside in the controllers, with our models being used strictly to describe the relationships between the tables and do not yet have any validation code we can test. We currently have almost a one-to-one relationship between our controllers and our tables, where each table's controller contains the create, update,

read and delete functions for the data - these are the functions we will test in our unit testing. We will also write our unit tests with ‘unusual’ inputs to attempt to make them fail as well. Below are our main central components broken down into smaller unit tests of our function.

Functions	**Test**	**Table**
CREATE	Create a student profile “Issac” with sample strings for the remaining attributes	Student Profiles - These tests will execute commands to interact with the student profile table of this database and test to ensure these functions are working properly for a test student profile.
UPDATE	Update one of the attributes and ensure it is saved and updated	
READ	Display the values of each of the attributes	
DELETE	Delete the profile and ensure it is removed from the database	

Functions	**Test**	**Table**
CREATE	Create a virtual classroom “1st Grade” with sample strings for the remaining attributes and student profiles	Virtual Classrooms - These tests will execute commands to interact with the virtual classrooms table of this database and test to ensure these functions are working properly for a test virtual classroom with profiles inside of it.
UPDATE	Update one of the attributes and ensure it is saved and updated	
READ	Display the values of each of the attributes	
DELETE	Delete the virtual classroom and ensure it is removed from the database	

Functions	Test	Table
CREATE	Create a learning activity “Test Activity” with sample strings for the remaining attributes and a few activities inside of it	Learning Activity - These tests will execute commands to interact with the learning activities table of this database and test to ensure these functions are working properly for a test learning activity with activities inside of it.
UPDATE	Update one of the attributes and ensure it is saved and updated	
READ	Display the values of each of the attributes	
DELETE	Delete the learning activity and ensure it is removed from the database	

Functions	Test	Table
CREATE	Create a TIP Class “Test TIP Class” with sample strings for the remaining attributes and classrooms, enrollments, learning activities and profile hide/show data inside of it. Ensure it is all saved and the respective relational join tables are properly set up.	TIP Class - These tests will execute commands to interact with the TIP Class table of this database and test to ensure these functions are working properly for a TIP Class with the following inside of it: ● Virtual Classrooms ● Users through Enrollments ● Learning Activities ● Profile Hide/Show for TIPs
UPDATE	Update one of the attributes and ensure it is saved and updated Update a classroom, enrollment, learning activity and profile hide/show to now be apart of the TIP	

	class and ensure the relation is established Update a classroom, enrollment, learning activity and profile hide/show to no longer be apart of the TIP class and ensure the relation is destroyed	
READ	Display the values of each of the attributes Display the classrooms, enrollments, learning activities and profile hide/show data related to the TIP class	
DELETE	Delete the TIP class and ensure it is removed from the database, as well as all of the join tables it was related it	

Integration Testing

After we make sure that each unit is unit tested before we can start the integration testing. Integration testing is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing. The main goal of the testing performed is to expose defects in the interfaces and in the interactions between integrated components or systems Integration testing is done to test the modules/components when integrated to verify that they work as expected to test the modules which are working fine individually does not have issues when integrated. A Module, in general, is designed by an individual software developer whose understanding and programming logic may differ from other programmers. Integration Testing becomes necessary to verify the software modules work in unity. Interfaces of the software modules with the database needs to be correlating. Integration test cases differs from other test cases in the sense it focuses mainly on the interfaces & flow of data/information between the modules.

We will be taking the incremental approach, where the testing is done by joining two or more modules that are logically related. Then the other related modules are added and tested for the proper functioning. The process continues until all of the modules are joined and tested successfully. First we will be testing our end - to - end workflow/ business scenarios which takes the user through a series of webpages to complete. Then we will be Testing all links in your webpages are working correctly and make sure there are no broken links. Links to be checked will include outgoing, internal, and anchor links.

Test Case ID	Test Case Objective	Test Case Description	Expected Results
1	Sign up page	A page where the user is about to create an account	User will be able to login with the new account
2	Login page	Faculty and students will be able to login	After logging in, they will be directed to their dashboard
3	Dashboard links	Testing the dashboard links, making sure they are working and directed to the correct page for faculty and students	Making sure all the links go to the correlating correct page and the previous page
4	View/Edit Profiles	We will be seeing if the view edit profiles and the search bar works	User should be able to view all the student profiles, edit them, and search through each field
5	Add Student Profiles	User should be able to add a student profile	User will be able to add a student profile and then be redirected the view/edit student profiles, where they can search for their new student

6	View All Classrooms	User will be able to view all the classrooms	The View All Classrooms page will direct them to view all the classrooms and be able to click on the classrooms link
7	Build a classroom	User will be able to build a classroom	User will be able to a build a classroom and add a name and description. After it is made then it will be redirect to the view all classroom page.
8	Add an activity/learning activity	Users will be able to add an activity or learning activity via pdf or text	When the user creates an activity or learning activity, they should be redirected to the learning activites menu
9	Manage learning activities	User will be able to view all the learning activities and edit if needed.	User will be able to view all the learning activities and edit if chosen. After the user has edited the learning activities then it should be redirected to the manage learning activities page.
10	Update TIP Progress	User will be able to pick a student and the correlating activities and click if its complete, by default it is not complete is chosen.	After the user has altered a TIP progress, then the page should be refreshed with a message stating it was updated
11	TIPS accessing	TIPS will be able access the	TIPS will be access

	activities	pages that their respected faculty member has assigned them	the activities and it will popup in a new tab

Usability Testing

Usability Testing will allow the team to view and analyze how the end-user will interact with the web application. The purpose of this test is that it will establish a baseline of user performance and highlight any design concerns that must be addressed. It utilizes a combination of methods to determine the understandability of the user interface and experience.

Our Plan

Moderated testing is usability assessment completed by the end-user in the presence of a moderator, which is most appropriate for our system. A moderator will guide the user through the flow and take notes on observations that will be used to measure the overall usability. The subject will be encouraged to think aloud to provide the moderator with the most accurate insight. They will then attempt to complete a series of tasks that are specified in the system's user flow. The moderator will need to take note of the following:

- How long did it take for the user to complete a task?
- Is the application showing problems with functionality?
- How satisfied is the user with the web application?
- Is the user able to navigate the website without frustration?
- How many clicks does it take to reach a task?

Although online tests provide tools for analyzing the results of this information, our targeted audience is a specific demographic. Our product must be suitable for NAU students and faculty at the College of Education. Using and analyzing our custom test on the intended audience ensures that our results are as accurate and realistic as possible. Participating in the tests will be our sponsor, Gretchen McAllister, as well as her colleagues and student teachers in practice. They will range in low to moderate for their technical abilities in order to demonstrate how the average user will use the system. The tests will span across three different demonstration sessions held in the first few weeks of April in the CUPI room located in the Eastburn Education Center.

In addition to these tests, we will need to frequently test our application on our own. As specified in the Requirements Document, our system must be easy to understand from

the programmer perspective as well as being flexible for future improvements. In order to keep track of this information, we have created a document that will keep track of our personal observations when interacting with the web application. We will document information such as inadequate buttons for transitioning, flaw in user flow, or problems with functionality specified in the Requirements Document.

Conclusion

As shown above, our various testing methods should ensure a smooth quality for our web application. In summary we have our unit testing, our integration testing, and our usability testing. For unit testing we are mostly testing CRUD features on the site and making sure they are properly functional when accessed by a user. Such features include simple actions such as being able to log in, being able to edit your profile, creating classrooms or activities, assigning TIPs to offerings, and other features. Our integration testing consists of making sure that various functional modules are also capable of working(integrating) together. For example if we use the function to create a classroom and then we use a function that lists and edits all classrooms, they should be able to show the created classroom. In a similar fashion, we are checking if all functions are successfully integrating any created outputs or data with each other and not simply displaying things. Finally, usability testing involves our client and other users actually testing the front-end website portion of the product i.e. actually using the product. The usability tests will measure user metrics such as click times, difficulty or confusion on access certain site sections, or them discovering any bugs. Along with users testing the site, we will also constantly be testing our own site as we continue to add or fix any features.

Our goal is that these various methods of testing will bring to light any errors or flaws in design that can be fixed iteratively. Hopefully the tests will catch everything but since that is probably impossible our tests will preferably find as many bugs as possible. Having our client and various users test the product should provide a fresh perspective in bug catching that is harder to see as a developer that already knows how to pilot the application without errors. After these test our end result should be a more refined and usable web application for our client.