CS 486C - Team Badgers
Software Testing Plan

Northern Arizona University

March 26, 2021

Sponsor: Dave Hopfensperger, Hanz Yeazel, and Glenn Austin

Team Mentor: Sambashiva Kethireddy

Members:
Abdulrahman Alamoudi
Tristan Marcus
Logan Ritter
Robel Tegegne
Yuanbo Xu

Overview:

The purpose of this document is to outline our team's plan to test our product. This will ensure our project's implementation exhibits the necessary functional and non-functional characteristics. This document will serve as an outline which the team will follow throughout the testing phase of our project.

Table of Contents

1. Introduction

Since the beginning of March 2020, the world has been dealing with the SARS-Cov-2 pandemic, better known as Covid-19 or Coronavirus. The virus has affected many people's lives causing a massive change in work environments. Most people, even those who have not caught the disease themselves, have been impacted. Millions of people have lost their jobs due to the inability to attend work because their business is not considered essential and the state government is requiring the business to be closed. There are not a lot of places that are capable of migrating their employees to a virtual workspace, but even those who can are stumbling upon a new set of issues. The majority of companies that were able to operate virtually are finding that many employees are struggling with motivation, recognition, and many other things that they use to find daily in the office. Since virtual employees are now abruptly working on a laptop in their home “office”, they lack the social interaction that was constantly present in the workplace of old. In most cases productivity is down due to the lack of oversight, depression is up due to the lack of social interaction, and motivation is down due to the lack of recognition.

Our sponsor for this project is State Farm and more specifically Dave Hopfensperger and Glenn Austin who represent the Enterprise Technology Department of State Farm. In Enterprise Technology there are hundreds of teams and around 6,000 employees that work on a wide variety of things to keep the company running smoothly while also keeping State Farm up to date in this ever growing world of technology. State Farm has always been a great place to work for employees primarily because of the respect and recognition they show their employees. In the office there was constant praise and recognition for everyone who produced quality work State Farm is known for. This came in the form of verbal praise, buying your coworker a coffee or donuts, and even sometimes ribbons to be displayed on your desk. Now since the majority of Enterprise Technology is working at home virtually due to the Covid19 Pandemic, it is not currently possible to award an employee with anything more than an email, or praise on a skype call. Mr. Hopfensperger and Mr. Austin have proposed that this product can solve a lot of the social problems presented by the virtual workspace nature.

In this document we will be discussing why and how we are going to perform three types of testing. The first being our Unit testing which will go over each key component of the product and test their functionality. The next type of testing will be our integration testing where we will ensure that all our components are communicating and working together properly. Lastly we will test our usability which will ensure our product works perfectly from the user perspective. For the unit testing, the team will carry out the testing by manually testing each unit and component. They will make a list of all the functionality and go one by one making sure each component does what it's supposed to do and throws errors when needed. For integration, we will make sure that all our components are communicating properly by ensuring that our waterfall feed is displaying actions on multiple users applications and also ensuring that the correct badges and kudos are displaying where and when they need to be displayed. Lastly for our Usability testing, each member will create an account on the application and will use the application as intended. They will perform actions that are supposed to work and attempt to perform actions that should break the application. The majority of our testing will take place in the Unit testing as our application has a lot of different functionality. We only have three components to our application which is why our integration testing will not be long. Performing all these tests will serve to help make our application better and fully functional from all aspects. It will also help weed out any unforeseen errors that may occur when the product is handed over to our clients.

2. Unit Testing

Unit testing is a form of testing that focuses on each individual component of a product and only tests the functionality of that component. The purpose of unit testing is to get a detailed inspection of a small portion of your application rather than looking at the product as a whole. Most unit testing is just simple input and output verification. If a unit or specific functionality passes our unit tests this proves to the client and potential users that our product is working as intended. If a unit fails this also helps as it shows us developers where fixes and improvements need to be made.

Our goal with unit testing is to identify any components that do not work as intended and fix them as needed. Doing this will allow us to get closer to our final product while making sure each component is working properly. The components of our application that we will be unit testing are; the user login and authentication, the badge delivery system, the Kudos system, badge printing, the email signature generator, and our waterfall feed and badge library. For each of these components we will have 3 phases of our testing.

1. Front-end execution - The first phase is to perform the action on the front end and verify the results.
2. Back-end functionality - The second phase is to ensure the functionality on the backend was performed correctly, this will be done using postman.
3. Database verification - The final phase will be to verify the action performed has correctly updated the database with the new data.

For each component we will discuss the purpose of each component and provide the unit testing descriptions for each of the components functions.

2.1. User Login Authentication

The goal of our User Login Authentication system is to manage and provide a way for users to securely create and continue using our application. It also allows us a way to uniquely identify every user who signs up for our application using a token and their email address. The three functions we will test are the Register, Login, and Log out functions. Below are the unit testing descriptions for each of the functions.

Register

A. **Description:** The purpose of this function is to allow a new user to create an account on the application using their email address.

B. **Main Flow:**

   1. Go to the application Login page.
   2. Click the register button at the bottom of the modal to switch from login to register.
   3. Fill in the email field with a valid email.
   4. Fill in the password field with your new password.
   5. Click the register confirmation button at the bottom of the modal to submit your information.
   6. The user will then be directed to the application homepage.

C. **Expected Outcome:** A user will have successfully created an account with the information they have provided.

Login

A. **Description:** The purpose of this function is to allow a user who is already registered to login to the application with their account information.

B. **Main Flow:**

1. Go to the application login page.
2. Fill in the email field with a valid email.
3. Fill in the password field with your password.
4. Click the login button at the bottom of the modal to submit your information.
5. The user will then be directed to the application homepage.

C. **Expected Outcome:** A user will have logged in to the application with their pre-existing account.

Log Out

A. **Description:** The purpose of this function is to allow a user to log out of their account before closing the application.

B. **Main Flow:**

1. While logged in to the application, locate the logout button in the top right corner of the screen.
2. Click the logout button.
3. The user will then be directed to the login page and will no longer be logged in to their account.

C. **Expected Outcome:** A user will have successfully and securely logged out of their account on the application.

2.2. Badge Delivery System

The goal of our badge delivery system is to provide a way for our users to send and receive badges with other users. This is the heart of the application and will serve as the main rewards system for our users. The function we will be testing is the Send Badge function.

Send Badge

A. **Description:** The purpose of this function is to allow a user to send a badge to another user using their email address. A user may also send a personal message to the receiver with the badge.

B. **Main Flow:**

1. Click the "Send a Badge" tab on the left side of the homepage.
2. The Send a Badge Modal will pop up.
3. Fill in the recipient field with the recipients email address.
4. Fill in the message field with a reason as to why you are sending the badge.
5. Select a badge from the list on the right hand side of the modal.
6. Click the Send Badge button.
7. The badge will then be added to the recipients badge library.
8. The application will then create and post a message onto the waterfall feed for all users to see.

C. **Expected Outcome:** A badge will be sent to the recipients badge library and a waterfall feed message will be posted.

2.3. Kudos System

The goal of our Kudos System is to provide the user with a second reward type in the form of a social currency. Users will be able to send and receive Kudos from their own personal bank to other users. Users will also be able to spend their Kudos in exchange for rewards such as candy and coffee. The functions we will be testing are the Send Kudos and Spend Kudos functions.

Send Kudos

A. **Description:** The purpose of this function is to allow a user to send kudos to another user.

B. **Main Flow:**

1. Click the “Send Kudos” tab on the left side of the homepage.
2. The Send Kudos Modal will pop up.
3. Fill in the recipient field with the recipients email address.
4. Fill in the message field with a reason as to why you are sending Kudos.
5. Fill in the amount field with the amount of Kudos you wish to send.
6. Click the Send Kudos button.
7. The amount of Kudos will then be subtracted from your bank and added to the recipients bank.
8. The application will then create and post a message onto the waterfall feed for all users to see.

C. **Expected Outcome:** Kudos will be sent from one user's bank to another and a message will be posted in the waterfall feed.

Spend Kudos

A. **Description:** The purpose of this function is to allow a user to spend the kudos they have earned on an item in the reward shop.

B. **Main Flow:**

   1. Click the “Spend Kudos” tab on the left side of the homepage.
   2. The Spend Kudos Modal will pop up.
   3. Select the item(s) that you wish to purchase with your Kudos
   4. Click the confirm button at the bottom of the modal.
   5. The amount of Kudos that the item(s) cost will then be deducted from the users Kudos bank.

C. **Expected Outcome:** An Item from the shop will have been purchased by the user and a receipt will be provided for the user to print. A message will also be posted in the waterfall feed.

2.4. Badge Printing

The goal of the badge printing component is to allow the user to select a badge from their bade library to print and display in the real world. The user may also save the badge certificate on their machine. The function we will be testing is the Select Badge function.

Select Badge

A. **Description:** The purpose of this function is to allow the user to select a badge from their badge library for printing.

B. **Main Flow:**

1. Click the “Print Badge” tab on the left side of the homepage.
2. The Print Badge Modal will pop up.
3. Select a badge.
4. Click the confirm button.
5. The application will open the certificate PDF in another tab for the user to do with as they please.

C. **Expected Outcome:** The user will have access to the PDF of the selected badge.

2.5. Email Signature Generator

The goal of the email signature generator is to allow the user to create a custom email signature highlighting the badges and Kudos that they have earned. The user will have full customization of the signature. The function we will be testing is the Customize Signature function.

Customize Signature

A. **Description:** The purpose of this function is to allow the user to customize their email signature.

B. **Main Flow:**

1. Click the “Email Signature Generator” tab on the left side of the homepage.
2. The Email Signature Generator Modal will pop up.
3. Fill out the name field with the name you would like to be on the signature.
4. Fill out the information field with any extra information you would like to be in the signature. (email, phone number, etc.)
5. Select a font.

6. Select up to three badges.
7. Click the generate button.
8. The Modal will then preview the generated signature which the user can then copy to their clipboard.

C. **Expected Outcome:** A custom email signature will have been generated for the user to copy.

2.6. Waterfall Feed and User Badge Library

The goal of the waterfall feed is to display any action such as badges or Kudos earned on the homepage of the application for every user to see. This allows users to show off the accomplishments they have achieved. The goal of the User Badge Library is to display to the user the badges they have earned. The functions that we will be testing are the Display Feed, Update Feed, Display Library, and Update Library functions.

Display Feed (Initial)

A. **Description:** The purpose of this function is to load the most recent version of the waterfall feed when the application is launched.

B. **Main Flow:**

1. Log into the application.
2. The application will pull the most recent messages.
3. The application will display the most recent version of the waterfall feed.

C. **Expected Outcome:** The most recent version of the waterfall feed will be displayed when the application is first opened.

Update Feed

A. **Description:** The purpose of this function is to update the waterfall feed for all users when one user performs an updating action.

B. **Main Flow:**

1. A user performs an updating action.
2. The application Updates the feed with a new message.
3. The updated waterfall feed is displayed to the user in real time.

C. **Expected Outcome:** The most recent version of the waterfall feed will be displayed given the user was already logged in before the updating action was performed.

Display Library (Initial)

A. **Description:** The purpose of this function is to load the correct badges that the user has earned in the past on the right hand side panel(personal badge library).

B. **Main Flow:**

1. Log into the application.
2. The application will pull the badges the user has earned from the database.
3. The application will display the users badges in the personal badge library on the right panel.

C. **Expected Outcome:** All the badges the user has earned in the past will be displayed when the application is first opened.

Update Library

A. **Description:** The purpose of this function is to update the users personal badge library after they earn a new badge.

B. **Main Flow:**

   1. A user sends you a new badge.
   2. The application updates your badge library.
   3. The application displays the updated badge library on the right side panel.

C. **Expected Outcome:** The updated version containing the new badge and all the previous badges earned is displayed to the user in real time.

After all these unit tests are performed the team will have a better understanding of what is working and what still needs improvements. Next we will do our integration testing to ensure that all our components are communicating and working with one another.

3. Integration Testing

After the completion of our unit testing we will move our to our next round of testing which will be our integration testing. Integration testing is a series of tests whose purpose is to expose problems and errors that occur between two interfaces and their interactions with one another. Although each individual component may work perfectly on its own it is essential to do integration testing to ensure that the product also works perfectly together as a whole.

For our project we have three technical interfaces that are in constant communication with each other. Those interfaces are the front-end, the back-end, and the database. To test that these modules are communicating properly with one another and the data passed between them is correct we have divided our testing into three separate groups. The first group of testing will happen on the Kudos. The second group of testing will be on the badges. The last group of testing will be on the waterfall feed, this ensures that our action components are communicating with our homepage.

When performing the integration testing our team will be carrying out two checks after an action is performed to verify the transfer of data is equal on all three interfaces.

1. Front-end - The first check will be to verify the correct value/result is appearing on our applications front-end (for example if Kudos are spent, verify the Kudos bank has lost the amount of Kudos spent). If the correct value/result is showing we also know the back-end is performing properly as the majority of our functionality happens on the back-end.

2. Database - The Second check will be to ensure the database has the same value from the first check. This can be done by checking our database dashboard.

3.1. Badges

Our badges are widely used on our application and are the main form of achievement that users can earn. Every user has a badge library that is stored in our database and is a list of badge objects that is constantly being added to. We will need to ensure that these badges in the users badge library can be accessed by both the front and back ends of our system safely. When a function is performed involving a badge, this functionality happens on our back-end and the badges are displayed on our front end for the user to view. There are three different components that utilize these badges and use all three of our interfaces to do so. Those three components are receiving a badge, printing badges, and the email signature generator. Since there are multiple components that use the same badges, we must ensure that the transfer of the badge and all its data, between interfaces, is correctly passed. We can ensure this by performing our integration tests. Below we have described how our interfaces should interact and how we plan on testing the integration between each of our interfaces when a badge is received, a badge is printed, and when an email signature is generated.

Receiving a Badge

A. **Description:** This test will ensure that all our interfaces are working together correctly when a user receives a badge.

B. **Main Flow:**

   1. Create and login to a new account (this is so the users badge library is clear and it is easier to tell if receiving a badge has worked).

    2. Have another user send you a badge and tell you which badge was sent and the message that was sent with it.
    3. **VERIFY** that the badge and message sent by the other user is appearing in your badge library on the right panel.
    4. **VERIFY** in the database dashboard that the badge and message sent are the same badge and message that are showing in your badge library.

C. **Expected Outcome:** The badge and message sent by another user are consistent between interfaces.

Badge Printing

A. **Description:** This test will ensure that all our interfaces are working together correctly when a user prints a badge.

B. **Main Flow:**

1. Login to an account that has preexisting badges (but not too many as it will be hard to tell if all badges are appearing in further steps).
2. Click the "Print a Badge" tab on the left panel.
3. Once the modal pops up, **VERIFY** that all the badges available to print match those that are in your badge library.
4. Select any badge and click submit.
5. **VERIFY** that the PDF the application sent you to is the same badge as the one selected in the above step.

C. **Expected Outcome:** The badges available to print are consistent with the badges in the badge library and the badge selected matches the badge in the PDF that the application sent the user to.

Email Signature Generator

A. **Description:** This test will ensure that all our interfaces are working together correctly when a user generates an email signature.

B. **Main Flow:**

1. Login to an account that has preexisting badges (but not too many as it will be hard to tell if all badges are appearing in further steps).
2. Click the “Email Signature Generator” tab on the left panel.
3. Once the modal pops up, **VERIFY** that all the badges available to select match those that are in your badge library.
4. Fill out the rest of the information fields.
5. Select up to three badges and click submit.
6. **VERIFY** that the email signature generated contains the same badge(s) as the one(s) selected in the above step.

C. **Expected Outcome:** The badges available to include in the email signature are consistent with the badges in the badge library and the badges selected match the badges in the generated email signature that the application has created for the user.

After performing all of these integration tests we can confirm that the data for our badges are correctly being passed between our interfaces.

3.2. Kudos

Similar to our badges, the Kudos system will be widely used on our application so we must ensure that the amount of Kudos a user has is safely stored and altered as needed. We will store the amount of Kudos a user has in our database. Any alteration on a user's Kudos, such as sending Kudos to another user, receiving Kudos, or spending Kudos will happen on our back end. The visual representation of a user's Kudos will happen on the front-end. As you can see we use the same Kudos value on all three of our interfaces so it is vital that we ensure the communication when passing the Kudos data between our interfaces is working properly. Below we have described how our interfaces should interact and how we plan on testing the integration between each of our interfaces when Kudos are sent, received or spent.

Send Kudos

A. **Description:** This test will ensure that all our interfaces are working together correctly when a user sends Kudos.

B. **Main Flow:**

    1. Click the "Send Kudos" tab on the left panel.
    2. Once the modal pops up, **VERIFY** the amount of Kudos available to send matches your Kudos bank on the homepage.
    3. Fill out the recipient, message, and amount fields. Then click submit.
    4. Next, **VERIFY** the amount sent has been subtracted from your Kudos bank.
    5. Lastly, **VERIFY** the amount of Kudos showing in our database matches the Kudos bank amount on the homepage.

C. **Expected Outcome:** All values through the test are consistent between interfaces.

Receiving Kudos

A. Description: This test will ensure that all our interfaces are working together correctly when a user Receives Kudos.

B. Main Flow:

1. Have another user send you a specific amount of Kudos.
2. **VERIFY** the amount sent has been added to your Kudos bank.
3. Lastly, **VERIFY** the amount of Kudos showing in our database matches the Kudos bank amount on the homepage.

C. Expected Outcome: All values through the test are consistent between interfaces.

Spend Kudos

A. Description: This test will ensure that all our interfaces are working together correctly when a user Spends Kudos.

B. Main Flow:

1. Click the “Spend Kudos” tab on the left panel.
2. Once the modal pops up, **VERIFY** the amount of Kudos available to spend matches your Kudos bank on the homepage.
3. Select any amount of items whose total is less than your Kudos bank amount. Then click submit.
4. Next, **VERIFY** the amount spent has been subtracted from your Kudos bank.
5. Lastly, **VERIFY** the amount of Kudos showing in our database matches the Kudos bank amount on the homepage.

C. Expected Outcome: All values through the test are consistent between interfaces.

After performing all of these integration tests we can confirm that the data for our Kudos System is correctly being passed between our interfaces.

3.3. Waterfall Feed

Our application utilizes a waterfall feed on the homepage to allow users to see the accomplishments and achievements of their peers. This waterfall feed will be similar to the ones used by twitter and facebook. The waterfall feed will contain messages that show the user when another user has earned a new badge or some Kudos in the form of a formatted message. We must ensure that when a user sends a badge or Kudos that the formatted message is correctly stored and displayed to every user in the application. Similar to our badges and kudos, the messages will be stored in the database in a list that is ordered from newest to oldest and will only contain 50 of the most recent messages (any more would be unnecessary). Below we have described how our interfaces should interact and how we plan on testing the integration between each of our interfaces for the waterfall feed on start up, when a badge is sent, and when Kudos are sent.

Start up

A. **Description:** This test will ensure that all our interfaces are working together correctly on the waterfall feed on startup.

B. **Main Flow:**

   1. Login to the application.

2. **VERIFY** that the messages in the waterfall feed match the messages in the database dashboard in the correct order.

C. Expected Outcome: All messages displaying in the waterfall feed are consistent throughout the interfaces and are appearing in the correct order.

Send a Badge

A. Description: This test will ensure that all our interfaces are working together correctly on the waterfall feed when a user sends a badge.

B. Main Flow:

1. Click the “Send a Badge” tab on the left panel.
2. Send a specific user a badge with a message.
3. Next, **VERIFY** the message formatted is displaying at the top of the waterfall feed with all the correct information.
4. Next, have another user **VERIFY** the message formatted is displaying at the top of the waterfall feed with all the correct information on their homepage.
5. Lastly, **VERIFY** the message formatted displaying is consistent in the database dashboard.

C. Expected Outcome: The message formatted by the Send a Badge modal is displaying in the correct location for all users on the waterfall feed and is consistent in the database.

Send Kudos

A. **Description:** This test will ensure that all our interfaces are working together correctly on the waterfall feed when a user sends Kudos.

B. **Main Flow:**

    1. Click the "Send Kudos" tab on the left panel.
    2. Send a specific user Kudos with a message.
    3. Next, **VERIFY** the message formatted is displaying at the top of the waterfall feed with all the correct information.
    4. Next, have another user **VERIFY** the message formatted is displaying at the top of the waterfall feed with all the correct information on their homepage.
    5. Lastly, **VERIFY** the message formatted displaying is consistent in the database dashboard.

C. **Expected Outcome:** The message formatted by the Send Kudos modal is displaying in the correct location for all users on the waterfall feed and is consistent in the database.

After all the integration tests are performed the team will have a better understanding of which interfaces are working and communicating correctly between each other and what still needs improvements. Lastly we will do our usability testing to ensure that our product as a whole is easy to use and is simple to learn.

4. Usability Testing

Usability testing is different from the other types of testing previously mentioned as it does not test the code but instead assesses how easy the product as a whole is to use. This testing section is critical to our overall product as it determines the quality of it. If the product is easy to use, learn, and navigate then the application passes the usability testing. On the other hand if the product is not easy to use and users struggle with it, the product fails.

When we deliver our product to our clients we want to make sure that it is easy to learn and use so that they do not become frustrated with it. Along with our usability testing, we will be providing our clients with a user manual that will teach them how to navigate through our application. This manual will focus on each individual component of our product and will have a very detailed description on how to use them.

For our usability testing we will have two separate phases or test groups to ensure the products ease of use. The first test group will be the team as we know the product best and can figure out any major issues or errors before we allow other users to use the product. The second testing group will be our clients as they are the users who will be using the product when it is finished. The reason we want an outside group to test our product usability is due to the fact that it is difficult to assess the usability of the product as the developer because we know the ins and out of the code. Having users who have never seen the product perform testing will allow us to fully understand the product's ease of use and its shortcomings.

The plan for the first phase, team testing, of our application will go as followers. We will start by clearing the database so that our application is at a fresh start. Next we will all sign up and login. Then, for the next few days we will be using the product as if we were state farm employees. We will make a list of things that work perfectly, things that work but not as intended or are hard to use, and things that do not work at all or are extremely difficult to use. After we have finished our first round of testing we will go through our list and with the items in the “working but not as intended” group we will fix and find a way to make them easier to use. With the items that do not work at all or are difficult to use we will also fix our errors and work to enhance these items so that they are easy to use and can become a functional component of our product. After our fixes and improvements we will do one more round of testing with our team to make sure the product is working as intended and is easy to use for us.

For the second phase of testing, the client testing, we will deliver our product to our clients with the user manual that teaches them how to use the product. The clients will use the product for a day or two to make sure they use every aspect of the application. For each user who tests the product we will ask them to make two lists for us. The first list ranking each function's ease of use. The client will rank the function 1-10, 1 being impossible to use and 10 being very easy to use/near perfect. We would also ask that they provided a reason as to why they ranked the function the score they did. The second list that we will ask the clients to make is a list of things they do not enjoy or things they wish were better. This could be things from the GUI to performance based criteria such as speed. This list will also help the team understand how the product looks from the user perspective.

After the clients finish their testing they will send us their list and the team will analyze each of the lists carefully. We will fix any errors reported and enhance any component that was reported hard to use. We will also weigh the opinions of our clients on any non-functional suggestion from the second list, and make changes where we see fit.

Performing these two phases of testing will allow the team to better understand and address the usability of our product. This is one of the most important testing sections as it determines if the application is usable and likeable.

5. Conclusion

Many companies, since the beginning of March, were forced to switch to a virtual workspace due to the covid-19 pandemic. With this new workspace a new set of problems have been introduced. Those include a lack of recognition and thus their enthusiasm and incentive at work drops sharply. Although some companies have tried mechanisms to provide the missing acknowledgement and feedback, they were only partially effective. State Farm on the other hand has challenged our team to build a gamified web based application to award and incentivise their employees in the form of badges and Kudos.

In this document we went over our plan to test our product in three different ways to ensure we are delivering the best product possible to our clients. The first of those three test types is unit testing, where we discussed how we are going to test every functionality of all our components to figure out what still needs to be improved. The second test type is integration testing, where we discussed how we are going to test the communication and compatibility between each of our interfaces. Our interfaces include our front-end, back-end, and database. The third and final testing type is usability testing, where we discussed how we will test our product for its ease of use by having two rounds of testing. The first round of usability testing will be done by our team. The second round of testing will be done by our clients, the reason for this is because they are our audience for this product.

Currently, the team is working on finishing touches and bug fixes with our product. We plan to have a finished product ready for testing in the next week or two. Once we have finished our product we will immediately begin unit testing, then integration testing, and lastly usability testing. Our team will follow this document as a guide for our testing.