|  |
| --- |
| BCFollowMe Mobile Application |
| Test Case Document |
|  |

|  |
| --- |
| Dylan McGraw, Brandon Larsen, and Paul Rachner  11/3/2014 |

**Table of Contents**

Introduction………………………………………………………………………………………..2

Constraints………………………………………………………………………………………...2

Approach…………………………………………………………………………………………..3

Roles………………………………………………………………………………………………3

Schedule…………………………………………………………………………………………...3

Test Cases…………………………………………………………………………………………3

References…………………………………………………………………………………………8

**Introduction**

The purpose of testing this mobile application is to make sure it is working properly before presenting it to our clients for use. Our goal is to get all the bugs out of the program so that it will run the way it was designed to run.

**Constraints**

One constraint that we have is that the members of the Quality Control Team don’t have extensive programming experience with Android. Another constraint is that we only have two phones to work with to test the application.

**Test Items & Environment**

The bug tracking software we decided to use is called Axosoft\*. It is an easy to use website (<https://bridgewater.axosoft.com>) that allows the Quality Control Team and the App Development team to easily track the bugs that are currently in the code. The phones we are using to run and test the app are two Moto G Android smartphones (phone version: 4.4 Kit Kat; app version: 1.08).

***\*Tutorial on Axosoft:***

The Axosoft site is fairly simple to use and easy to understand once you get an understanding of how to use the site. In the Organize Panel, there are four sections; Projects, Releases, Users & Teams, and Customers. The Projects panel shows the projects that are currently being worked on. The user can add or edit their projects by clicking on the “Add” or “Edit” buttons at the top of the panel. When the user clicks the “Add” button, a box shows up that asks you to enter the name of your new project. The Release panel shows the projects you have access to. In the Users & Teams panel, users can add users and also create teams within the panel. To add or edit these users, click on the “Add” or “Edit” buttons at the top of the panel. When adding a user, you simply enter the users e-mail address into the box and Axosoft sends them an email explaining to them how to confirm that they have been added. In the Customers panel, you can add or edit customers but clicking the “Add” or “Edit” buttons at the top of the panel. When adding a customer, it will ask you to type the company/customer name and also a URL. The main panel on the page is where you can add bugs and you add them by clicking the “Add” button at the top of the panel. When you click the “Add” button, it will show you different options you can set. You add a title, assign due dates for when you want the bugs to be tested by and assign who should test it. You can also assign priority to the bugs and the severity of the bugs.

For more information, the following link goes more in depth: <http://www.youtube.com/watch?v=r8ocw0-euOo>

**Approach**

The type of testing we are using for this mobile application is manual black box testing.

**Roles**

Dylan McGraw: Manager of the team and in charge of writing up everything but the test cases for this document (Introduction, Constraints, Test Items & Environment, Approach, Roles, Schedule, and References). Dylan’s role in the testing process is to manage the team and assign code to members to test.

Paul Rachner: Team member in charge of writing the other half of the test cases for this document (Splash Screen, Create Account, Login, View Main Screen, Load Excursion, Start Recording Route, and Add Observation). Paul’s role in the testing process is to test the code that was passed down from the App development team.

Brandon Larsen: Team member in charge of writing half of the test cases for this document (View Observation, Edit Observation, Stop Recording Route, Edit Excursion, Save Excursion, Logout, and Exit). He was also responsible for finding a free bug tracking software. Brandon’s role in the testing process is to test the code that was passed down from the App development team.

**Schedule**

Wednesday, November 12- Complete Test Case document

Once the Test Case document is completed, the Quality Control team will be getting training from App Development team on how to install the application via the website.

After this document is complete, we are going to test for bugs in the code on a weekly basis or as often as needed. The App Development team will pass along their code to the Quality Control team manager. The code will then be passed down to the members of the Quality Control team to test the code for bugs.

Wednesday, December 3- have the bug free application ready to present to the clients

**Test Cases**

**Test Case ID TC001 “Splash Screen”:**

**Summary** Verify that the splash screen is displayed for a brief period of time and the Main screen is displayed after the splash screen.

**Prerequisite** User clicks on the app

**Instructions** Turn on the app making sure the app is not currently loaded.

**Test Data and Expected Results**

1. Start app with user logged in. We should see the splash screen for a moment and then it should go to the Main Screen.
2. Start app with user not logged in. We should see the splash screen for a moment and then it should go to the Login Screen.

**Test Case ID TC002 “Create Account”:**

**Summary** Verify the Create Account button works

**Prerequisite** The app is on the Login Screen

**Instructions** Click the Create Account button.

**Test Data and Expected Results**

1. Should see the Create Account Screen.

**Test Case ID TC003 “Login”:**

**Summary** Verify login credentials

**Prerequisite** App has the Login Screen loaded

**Instructions** Enter login name and password and click the Login button.

**Test Data and Expected Results**

1. Login with valid username and valid password. System should display Main Screen.
2. Login with invalid username. System should display error message and clears the fields.
3. Login with valid username and invalid password. System should display error message and clear the fields.

**Test Case ID TC004 “View Main Screen”:**

**Summary** Main Screen is loaded and GPS is displayed.

**Prerequisite** User is logged into the app

**Instructions** User chooses to edit observation and the Edit Observation Screen is displayed. User hits the Save button to save the edited fields.

**Test Data and Expected Results**

1. User logs in, and map is displayed.

**Test Case ID TC005 “Load Excursion”:**

**Summary** Loading Excursion onto app

**Prerequisite** Main screen is loaded

**Instructions** Select an available excursion

**Test Data and Expected Results**

1. The User clicks an excursion that is wanted to be viewed. If the excursion isn’t on the device it is downloaded. Excursion is displayed by being plotted on the map and observations marked as well.

**Test Case ID TC006 “Start Recording Route”:**

**Summary** Current route is recorded

**Instructions** Turn on phone’s GPS calculator

**Prerequisite** Main screen is displayed and the app isn’t recording a GPS location at that time. Device setting to calculate GPS must be on.

**Test Data and Expected Results**

1. The user’s GPS location is calculated regularly and is stored on the map.

**Test Case ID TC007 “Add Observation”:**

**Summary** Add observation on the GPS map.

**Prerequisite** Main screen is displayed and the user clicked the Observation on the map.

**Instructions** Click the Observation mark on the screen. Scroll down the box to see all the fields.

**Test Data and Expected Results**

1. User selects an Observation from the Add Observation Screen and displays it on the map.

**Test Case ID TC008 “View Observation”:**

**Summary** Verify that the user is able to click and view a previously recorded observation

**Prerequisite** The app is at the main screen with a preexisting excursion loaded

**Instructions** From the main screen, click an observation mark from the excursion on the map.

**Test Data and Expected Results**

1. Click “observation” on map, a view observation screen will appear, featuring a scrollable box that contains the title, description and GPS location of the observation along with a button labeled “Back” and possibly “Edit”.
2. Click the “Back” button, the user is returned to the main excursion screen.
3. Click on an excursion that the user created. The Edit Observation screen should appear.
4. Load an excursion that the user did not create. An edit button should not be on the view observation screen.

**Test Case ID TC009 “Edit Observation”:**

**Summary** Verify that the user is able to edit the data elements of an observation within an excursion.

**Prerequisite** The application is displaying the view observation screen for an observation that the user logged in created.

**Instructions** Click the Edit Observation button

**Test Data and Expected Result**

1. Click the Edit Observation button, a menu screen should display with editable controls.
2. Click the Observation Data pane, a menu will show the title field, populated with existing observation data that is editable when clicked.
3. Click the description field, a menu screen will be populated with existing observation data that can be edited when clicked
4. Click the GPS Location button, a menu field will be populated with existing observation data, and will be editable through a clickable edit button
5. Click Cancel, the user returns to the main screen without saving any changes.
6. Click Save, the observation should save with any new user input information, then return the user to the main screen.
7. Click the Delete button, the observation should be deleted altogether and returns the user to the main screen.

**Test Case ID TC010 “Stop recording Route”:**

**Summary** Verify that the user is able to stop the application from recording user location.

**Prerequisite** The application is at the main screen and recording user location.

**Instructions** Press the stop recording button

**Test Data and Expected Result**

1. The stop recording button is pressed and the application stops recording user location

**Test Case ID TC011 “Edit Excursion”:**

**Summary** The user is able to take a preexisting excursion and edit data elements.

**Prerequisite** The application is displaying the main screen, and a preexisting excursion is loaded.

**Instructions** From the main screen, click the “Edit excursion” option.

**Test Data and Expected Result**

1. Click Edit Excursion, a menu screen is displayed with editable preference controls.
2. Enter “Test” in the name field, “Test description” in the description field, set Route toggle to ‘walking”, set toggle to private, and the excursion should remain editable.
3. Enter “Test” in the name field, “Test description” in the description field, set Route toggle to ‘walking”, set toggle to public, and the excursion should remain editable.
4. Enter “Test” in the name field, “Test description” in the description field, set Route toggle to ‘driving”, set toggle to private, and the excursion should remain editable.
5. Enter “Test” in the name field, “Test description” in the description field, set Route toggle to ‘driving”, set toggle to public, and the excursion should remain editable.
6. Click cancel, the user should return to the main screen.
7. Click Save, the excursion should update with any new user input information.

**Test Case ID TC012 “Save Excursion”:**

**Summary** Verify that the application is able to record and save an excursion to memory.

**Prerequisite** Application is displaying main screen, and an element of data for the excursion has been changed.

**Instructions** On the main screen, edit an element of data for an excursion, click the save button.

**Test Data and Expected Result**

1. With the excursion named default, click Save. The app should invoke the user to edit the name of the excursion.
2. With the excursion named a unique name and a valid Internet connection present, click Save. The excursion will upload to the application server and a message indicating successful save and upload will appear.
3. With the excursion named a unique name and an invalid Internet connection present, click Save. The excursion will be saved, but not uploaded to the server. A message indicating successful save, but unsuccessful upload, will appear.

**Test Case ID TC013 “Logout”:**

**Summary** Verify that the user is able to log out of the application through a logout button and dialogue box.

**Prerequisite** The application is logged in.

**Instructions** With the application logged in, click the “logout” button.

**Test Data and Expected Result**

1. “Logout” dialogue box is displayed. Click “logout” option. The user is returned to the Login screen.
2. “Logout” dialogue box is displayed. Click the “cancel” option. The user is returned to the Login screen.

**Test Case ID TC014 “Exit”:**

**Summary:** Verify that the application exits after an excursion is uploaded.

**Prerequisite** Application is running, with an excursion ready to upload.

**Instructions** With app running, attempt to exit.

**Test Data and Expected Result**

1. Begin uploading excursion, we should see a notification of the upload and the application should terminate after the excursion is fully completed.
2. Without an excursion loading, the app should immediately terminate.

**References**

The Requirements Analysis Document (RAD) is available on the mobile application website. (<http://bc-followme.azurewebsites.net/documents.html>)