Test Plan: G.R.E.G.G.

By Group 1: Oliver O’Brien, Jessica May, Derek Vogel, Nicolas Esser

1. Introduction

The purpose of this document is to provide a framework for the testing of the application G.R.E.G.G that will be used in Phase 4. This will provide the basis for testing procedure on the application while also explaining each of the steps. The end goal of this document is to provide the user and reader an explanation of our methodology in testing.

To clarify, the main way we will be testing this application is by downloading to an Android device and testing every possible button click. This will consist of testing all the building options combined with the two category options. We will also test creating an account and logging in from the main screen.

2. Testing Requirements

2.1. Application Startup

2.1.1. Launch the application.

2.2. Login Page

2.2.1. Attempt login with incorrect password.

2.2.2. Attempt login with incorrect email.

2.2.3. Attempt login with blank input fields.

2.2.4. Attempt to navigate to registration page.

2.2.5. Attempt to login with correct login information.

2.3. Registration Page

2.3.1. Attempt to create a password with username and password.

2.3.2. Attempt to create a account with an already existing username.

2.4. View Map

2.4.1. Check that the map loads on the application screen.

2.4.2. Check that map is navigable by the user, i.e., they can move the map to look at different areas.

2.5. Place Food Pin

2.5.1. Attempt to place pins at all 80 buildings, with descriptions

2.6. Place Free Stuff Pin

2.6.1. Attempt to place a free stuff pin at all 80 buildings, with descriptions

2.7. View Map with Pins

2.7.1. Insure that pins show up after pin creation.

2.7.2. Insure pins still show after logout and login.

2.7.3. Insure pins from other users show up.

2.8. Database Testing

2.8.1. Users are created accurately and stored correctly.

2.8.2. Pins are created accurately and stored correctly.

2.8.3. Buildings are all present in application.

3. Test Cases

3.1. Application Startup

3.1.1. Launch the application.

3.1.1.1. Purpose

3.1.1.1.1. To test that the application loads correctly when a user opens the application.

3.1.1.2. Setup

3.1.1.2.1. Launch the application from the device's homescreen.

3.1.1.3. Inputs and Expected Outputs

3.1.1.3.1.

Input Field	**Input**	**Expected Output**
Touching device.	Clicking the application.	The application opens to the login screen.

3.1.1.4. Requirements Covered: RS 3.2.1.1.1

3.2. Login Page

3.2.1. Attempt login with incorrect password.

3.2.1.1. Purpose

3.2.1.1.1. Testing login with incorrect password

3.2.1.2. Setup

3.2.1.2.1. Launch application.

3.2.1.3. Inputs and Outputs Expected

Input Field	**Input Value**	**Information Accepted**	**Expected Output**
Username	email@email.com	True	-
Password	wrong_password	False	Login Failed

3.2.1.4. Requirements Covered: RS 3.2.1.1.1

3.2.2. Attempt login with incorrect email.

3.2.2.1. Purpose

3.2.2.1.1. Test if the application will prevent the user from logging with with an email that is not in the database.

3.2.2.2. Setup

3.2.2.2.1. From the login screen on which the application opens, enter in a email that is known to not exist in the database.

3.2.2.3. Inputs and Expected Outputs

3.2.2.3.1.

Input Field	**Input**	**Information Accepted**	**Expected Output**
Username/email	herbie@huskers.unl.edu	False	Login Failed
Password	asdfgh	False	Login Failed

3.2.2.4. Requirements Covered: RS 3.2.1.1.1

3.2.3. Attempt login with blank input fields.

3.2.3.1. Purpose

3.2.3.1.1. Test if the application will allows users to progress past the login screen with blank input fields.

3.2.3.2. Setup

3.2.3.2.1. Launch application.

3.2.3.3. Inputs and Expected Outputs

Input Fields	**Input Value**	**Information Accepted**	**Expected Output**
Username	-	false	Login Failed
Password	-	false	Login Failed

3.2.4. Attempt to navigate to registration page.

3.2.4.1. Purpose

3.2.4.1.1. Test the applications ability to navigate away from the login screen to a registration page.

3.2.4.2. Setup

3.2.4.2.1. Launch the application and click the registration button when the login page loads.

3.2.4.3. Input and Expected Output

Input Fields	**Input**	**Information Accepted**	**Expected Output**
Button	Press registration button	True	Navigated to registration page

3.2.4.4. Requirements Covered: RS 3.2.1.1.1

3.2.5. Attempt to login with correct login information.

3.2.5.1. Purpose

3.2.5.1.1. Validate that the login process works correctly and allows users to enter the application when they submit the correct information.

3.2.5.2. Setup

3.2.5.2.1. Launch the application and enter a correct username and password.

3.2.5.3. Inputs and Expected Outputs

Input Field	Input	Information Accepted	Output
Username	email@email.com	True	Map screen loaded
Password	12345	True	Map screen loaded

3.2.5.4. Requirements Covered: RS 3.2.1.1.1

3.3. Registration Page

3.3.1. Attempt to create a password with username and password.

3.3.1.1. Purpose

3.3.1.1.1. To test the applications ability to create a new user.

3.3.1.2. Setup

3.3.1.2.1. Launch the application, when login screen loads, click on registration button. Then enter in a email and password and create an account.

3.3.1.3. Inputs and Expected Outputs

Input Field	Input	Input Accepted	Output
Username/email	test@email.com	True	Account created
Password	123456	True	Account created

3.3.1.4. Requirements Covered: 3.2.1.1.1

3.3.2. Attempt to create a account with an already existing username.

3.3.2.1. Purpose

3.3.2.1.1. Test that the application rejects a user who tries to register with a username already in use.

3.3.2.2. Setup

3.3.2.2.1. When on the registration screen, enter in a known username (i.e. create an account with one username and then go back to the registration page and try to log in with it).

3.3.2.3. Inputs and Expected Outputs

Input Field	**Input**	**Input Accepted**	**Output**
Email	email@fake.com	True	Account Created
Password	12345	True	Account Created
Email	email@fake.com	False	Username Already in Use

3.3.2.4. Requirements Covered: RS 3.2.1.1.1

3.4. View Map

3.4.1. Check that the map loads on the application screen.

3.4.1.1. Purpose

3.4.1.1.1. Test that once a user is logged in, the map of UNL loads correctly.

3.4.1.2. Setup

3.4.1.2.1. Log into an account and see if the map screen loads after.

3.4.1.3. Inputs and Expected Outputs

Input Field	**Input**	**Input Accepted**	**Output**
Username	email@email.com	True	Map loads
Password	12345	True	Map loads

3.4.1.4. Requirements Covered: RS 3.2.1.1.2

3.4.2. Check that map is navigable by the user, i.e., they can move the map to look at different areas.

3.4.2.1. Purpose

3.4.2.1.1. To make sure the map works correctly on the application and doesn't only show the Union.

3.4.2.2. Setup

3.4.2.2.1. After map has loaded, try to move the perspective around campus.

3.4.2.3. Inputs and Expected Outputs

Input Field	**Input**	**Input Accepted**	**Output**
Touch	Use finger to move map.	True	Map moves.

3.4.2.4. Requirements Covered: 3.2.1.1.2

3.5. Place Food Pin

3.5.1. Attempt to place pins at all 80 buildings, with a description

3.5.1.1. Purpose

3.5.1.1.1. To test that the user can create a Free Food pin associated with a building on the map, with a description

3.5.1.2. Setup

3.5.1.2.1. Navigate to the drop pin menu and traverse through until a pin is dropped

3.5.1.3. Inputs and Expected Values

Input Value	**Expected Output**
Food pin with a description attached to a building	The pin is added to the database with the description
Food pin without a descriptions that's attached to a building	The pin is added to the database with the description field left empty

3.5.1.4. Requirements Covered: RS 2.5.1

3.6. Place Free Stuff Pin

3.6.1. Attempt to place a free stuff pin at all 80 buildings, with a description

3.6.1.1. Purpose

3.6.1.1.1. To test that the user can create a Free Stuff pin associated with a building on the map, with a description

3.6.1.2. Setup

3.6.1.2.1. Navigate to the drop pin menu and traverse through until a pin is dropped

3.6.1.3. Inputs and Expected Outputs

Input Value	**Expected Output**
Free Stuff pin with a description attached to a building	The pin is added to the database with the description
Free Stuff pin without a descriptions that’s attached to a building	The pin is added to the database with the description field left empty

3.6.1.4. Requirements Covered: RS 2.6.1

3.7. View Map with Pins

3.7.1. Insure that pins show up after pin creation.

3.7.1.1. Purpose

3.7.1.1.1. To test that the created pins show up on the map

3.7.1.2. Setup

3.7.1.2.1. Launch the map and navigate to the location of recently dropped pins to verify they exist

3.7.1.3. Inputs and Expected Values

Input Field	**Input Value**	**Expected output**
Free Food Pin	Pin at a building	The pin shows up at the location on the map
Free Stuff Pin	Pin at a building	The pin shows up at the location on the map
No Pin	None	No pin will show up

3.7.1.4. Requirements Covered: RS 2.7.1

3.7.2. Insure pins still show after logout and login.

3.7.2.1. Purpose

3.7.2.1.1. To test that previous pins placed still appear on the map

3.7.2.2. Setup

3.7.2.2.1. Open the map and navigate to the pin dropped to test requirement 3.7.1

3.7.2.3. Inputs and Expected Values

Input Value	**Expected output**
Free Food Pin at a building	The pin shows up at the location on the map
Free Stuff Pin at a building	The pin shows up at the location on the map
None	No pin will show up

3.7.2.4. Requirements Covered: RS 2.7.2

3.7.3. Insure pins from other users show up.

3.7.3.1. Purpose

3.7.3.1.1. To show that other users can view the previous pins, even when logged in on a different account and ran on a different device

3.7.3.2. Setup

3.7.3.2.1. Open the map and navigate to pins placed down by another user as described in 3.7.1

3.7.3.3. Inputs and Expected Values

Input	**Expected Output**
Free Food Pin at a building	The pin shows up at the location on the map
Free Stuff Pin at a building	The pin shows up at the location on the map
None	No pin will show up

3.7.3.4. Requirements covered: RS 2.7.3

3.8. Database Testing

3.8.1. Users are created accurately and stored correctly.

3.8.1.1. Purpose

3.8.1.1.1. To show that each user is saved correctly in the database

3.8.1.2. Setup

3.8.1.2.1. This is done each time a new account is created

3.8.1.3. Inputs and Expected Outputs

Input Value	**Expected Output**
No users are created	This database table is null
One user is created	The user table will have one entry
Multiple users are created	The user table will have multiple entries

3.8.1.4. Requirements Covered: RS 2.8.1

3.8.2. Pins are created accurately and stored correctly.

3.8.2.1. Purpose

3.8.2.1.1. To ensure that the pins being created are stored in the appropriate columns.

3.8.2.2. Setup

3.8.2.2.1. This is done by creating pins in the application and then checking the Pins table in the database

3.8.2.3. Inputs and Expected Outputs

Input Value	**Expected Results**
No pins are dropped in the application	The pins table is empty
One pin is dropped in the application	The pins table contains one entry, covering fields of type, building, user, and description.
Multiple pins are dropped by one user in the application	The pins table contains multiple entries with user being the same on all of them, along with type, building, and description
Multiple pins are dropped by multiple users in the application	The pins table contains multiple entries with covering fields user, type, building, and description.

3.8.2.4. Requirements Covered: RS 2.8.2

3.8.3. Buildings are all present in application.

3.8.3.1. Purpose

3.8.3.1.1. To ensure that all buildings on all of UNL's three campuses are in the database and can be used in the application.

3.8.3.2. Setup

3.8.3.2.1. Look at map and database and make sure that all buildings on the map have their names in the database.

3.8.3.3. Inputs and Expected Outputs

3.8.3.4. Disclaimer: this table does not show all of the inputs as we have 80 buildings to check. It does give an example of how we will go about checking them and further information will be provided in the testing results.

Input Field	**Input**	**Input Accepted**	**Output**
Building Name	Avery	True	Avery is located in database.
Building Name	Union	True	Union is located in database.
Building Name	Old Father	True	Old Father is located in database.

3.8.3.5. Requirements Covered: RS 3.2.1.3.1