Geographical Recon Energy Grub Guide

Software Requirements Specification

By Group #1: Oliver O'Brien, Nicolas Esser, Derek Vogel, and Jessica May

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to provide explanation on the system G.R.E.G.G (Geographic Recon Energy Grub Guide) and its features.

1.2 Scope

G.R.E.G.G. will be an Android application that allows University of Nebraska-Lincoln (UNL) students and faculty to find and share the locations of free items and food on campus. Users will be able to mark locations on a map of campus and include a category and description which will be associated with the mark. Marks will not be visible to other users until three or more individuals have put a mark of the same category in the same location. Once a mark is visible, users will be able to vote on whether the mark is correct. Marks with a low enough score will be hidden. Android SDK will be used for the development of this app.

1.3 Definitions, acronyms, and abbreviations

G.R.E.G.G: Geographical Recon Energy Grub Guide
Pin: A marker dropped by a user on this map with an associated description and category.
API: Application Program Interface, a set system of programs used for developing applications.

1.4 Overview

The rest of this document will provide an overall descriptions of the system we are building. This will include the general purpose of the system, use cases, system specifications, and system requirements. The appendix will include descriptions of the different phases of development we envision and what we expect the application to look like.

2. Overall description

2.1 Product perspective

2.1.1 System Interfaces

Besides the database and the pin dropping interface, the only other interface our application will have to use is the Google Maps API. Besides this, it should be able to run on any Android phone.

2.1.2 User Interfaces

As is common with mobile applications, users will use touchscreen to navigate G.R.E.G.G and to drop and vote on pins.

2.1.3 Hardware Interfaces

As long as the device has an Android operating system, G.R.E.G.G. will be able to be downloaded and used.

2.1.4 Software Interfaces

This application will require Google Play Store to be downloaded.

2.1.5 Communications Interfaces

The application will need to be able to communicate with our database and the internet in order to update the map interface. Users will also need to use a keyboard interface to create posts.

2.1.6 Memory

G.R.E.G.G will have a relatively simple graphical interface, and will unlikely need much memory to function. Any phone that is capable of running Google Maps will likely be able to run the application as well.

2.1.7 Operations

There are only a handful of operations a user will make. Any user will be able to create a new pin on the map, report a pin as no longer being useful, and commenting on other users created pins. These will all be simple tasks with minimal interface.

2.1.8 Site Adaptation Requirements

There are no adaptation requirements for the system.

2.2 Product functions

This app's purpose is to make free items and food on the University of Nebraska-Lincoln easier to find.

2.3 User characteristics

The intended users for G.R.E.G.G. are students and faculty of the University of Nebraska-Lincoln. They will need to have an Android device. There is no specific age requirement, however the user will need to have a "unl.edu" email account to create an account on the application. This is to increase the amount of accuracy in pin dropping.

2.4 Constraints

Since this will be a mobile application designed for the Android operating system, we may be constrained by older versions of the OS that may not run the application as efficiently as newer versions. This will also be optimized for phone sized screens, so tablets running the application may not display correctly.

2.5 Assumptions and dependencies

We have four assumptions. The first two assumptions are that G.R.E.G.G. must abide by is that the base interface is connected to the Google Maps API and that the mobile device must be running Android. It is also assumed that the device can connect to the internet and communicate with the database and other devices. We are also assuming that users will have a 'huskers.unl.edu' email account in order to sign up.

3. Specific requirements

3.1 External Interface Requirements

The app will not require any external interfaces.

3.2 Functions

3.2.1 Phase 1

3.2.1.1 Start the Application

3.2.1.1.1   Upon starting the application, users will be presented with the map of campus centered on their location.

3.2.1.1.1   Users will be able to select a tab that allows them to switch between Downtown and East Campus maps.

3.2.1.2 Basic Pin Dropping

3.2.1.2.1   To drop a pin in this phase, the user must hold their finger over the location they wish to drop the pin for 3 (three) seconds.

3.2.1.2.2   Once that is completed a menu is brought up in which the user can select the category of pin and a description of it. Once the user taps confirm it is immediately placed on the map.

3.2.1.2.3   When a pin is dropped, it is added to a database.

3.2.1.3 Pin Display on Map

3.2.1.3.1   Pin locations and categories will be read from a database.

3.2.1.3.1   Pin icons will be placed at the locations read from the database. Icons will depend on category.

3.2.2 Phase 2

3.2.2.1 Start

3.2.2.1.1   If the user does not already have an account or is not logged in, it will begin on a login in screen that allows them to make an account or sign in to a previously made one.

3.2.2.1.2   If the user has already created an account and logged in, the application will open up to the home screen of their location on UNL's City/East campus.

3.2.2.2 Advanced Pin Dropping

3.2.2.2.1   In this stage, we will improve our pin dropping method. Users will need to be validated by at least two other users for their pin to show up (i.e. three users drop a pin in the same place).

3.2.2.2.2   The pins will now show a section to upvote or downvote a pin. Upvotes will ensure that the pin stays in the location. While downvoting will move to delete the pin from the location. Upvoting will be used to agree there is an event in the location, downvoting will be used to say that event is no longer taking place. To completely remove a pin, three downvotes will be needed.

3.2.2.2.3   After clicking on a pin, there will be an option to report the user who dropped it. Reporting will be used to report a user who entered a pin location for an event that is not actually taking place. Our validation process should stop this process from happening, however multiple users could post unreal events and validate each other. Reporting the event, will report all three users.

3.2.2.2.4   Reporting comments will be used for deleting inappropriate comments. The comment will be removed after one report, to ensure that discriminatory things are not being posted.

3.2.2.3 Notifications

3.2.2.3.1   Once the validation process described in section 3.2.2.2.3 is completed, a push notification, assuming it is allowed in the user's settings, is sent to the user. In this notification, it states the type of pin that was dropped and the location (which campus it appears on).

3.2.2.3.2   If the user taps on the notification, the app is opened and the map is set to where that pin was dropped, giving the user its location.

3.2.3 Improving the Interface

3.2.3.1   In this phase, the application will be aesthetically improved to make the user experience more enjoyable.

3.3 Performance Requirements

We expect our application to run as well as other map applications. This means very little stuttering, crashing, or lag.

3.4 Logical Database Requirements

Our database will need to be able store multiple types of information. In phase 1, we are planning on the application to be able to store pin locations placed by any user, information about the pins (categories, title, etc.) and replicate the pin for other users on their map. In phase 2, our database will be able to store user profiles/information, comments made by users, and contains logic that allows pins to be controlled.

3.5 Design Constraints

The main design constraints for G.R.E.G.G is that it will only be available on Android devices. This eliminates iOS users from being able to access the application.

3.6 Software System Attributes

3.6.1 Reliability

This application should run on an Android device without lag, freezing, or crashing.

3.6.2 Availability

This application should be available at all times. If the application were to crash or freeze, it should be able to close, be reopened, and work as normal.

3.6.3 Security

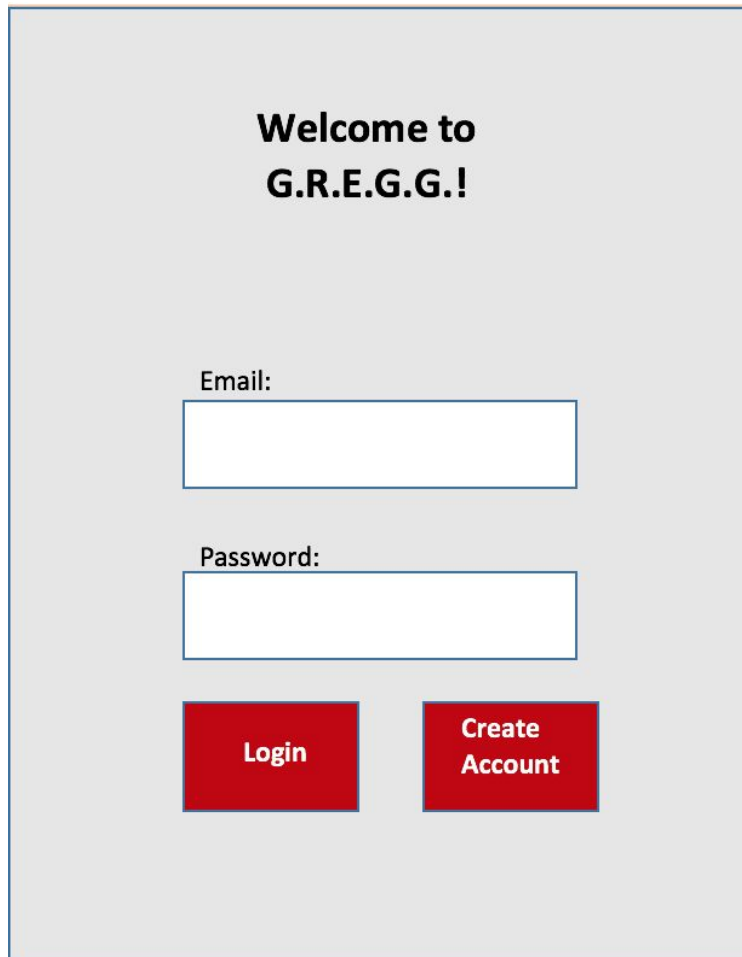The security of this application will be the email and password of the user.

3.6.4 Maintainability

Once the key features, like the map and the pins, are created, the application will be easily maintained. Since the map is the main focus of the application new layers and features can be easily added on top of that.

3.6.5 Portability

This application will be installed on Android mobile devices and can be installed through the Google App Store.

4. Appendix

4.1 Login Screen

**Welcome to
G.R.E.G.G.!**

Email:

Password:

Login

**Create
Account**
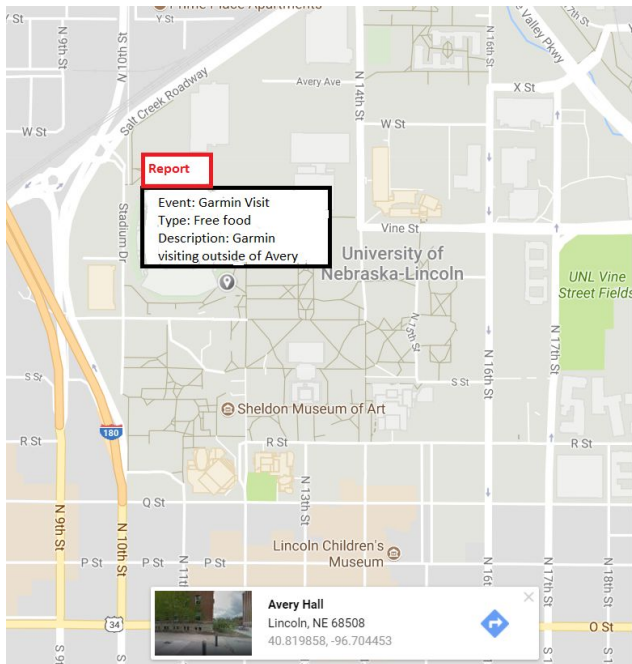
4.2 Create Account

Create Account

Email:

Password:

Confirm Password:

Create

4.3 Map Screen

4.4 Side Menu

Drop Pin
Settings
Account

Food
Free Stuff