Geographical Recon Energy Grub Guide

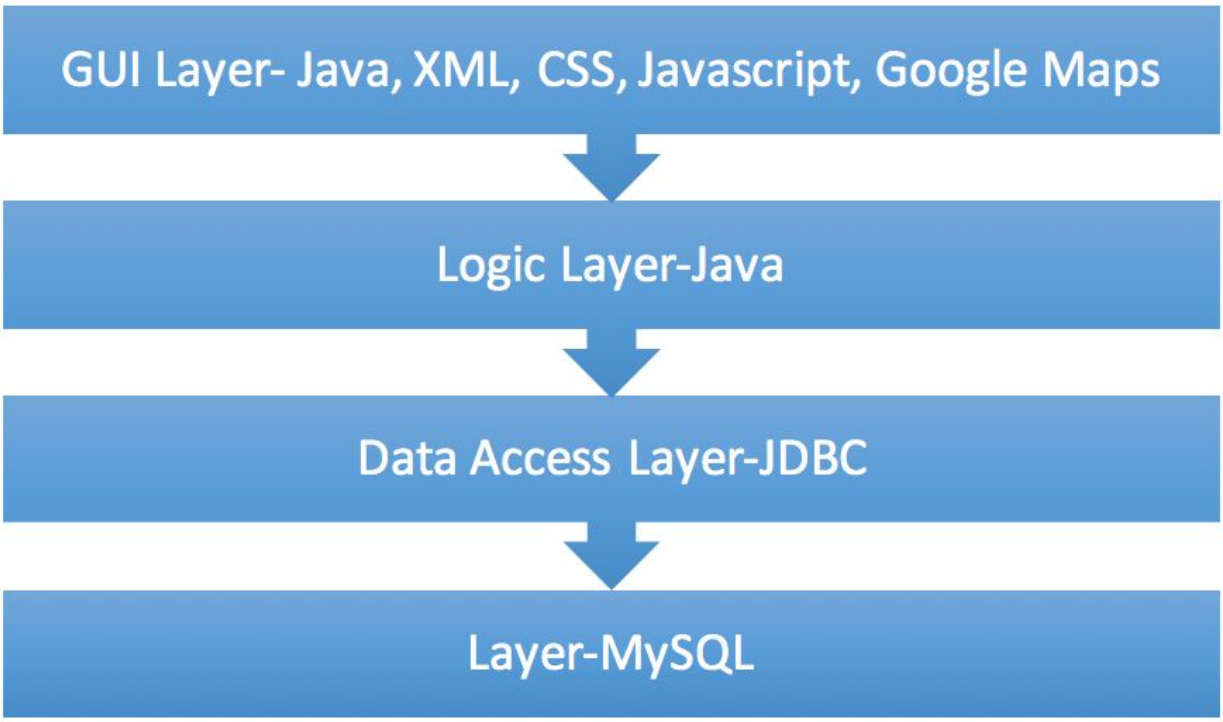
i

Design Document

Group #1 - Nick Esser, Jessica May, Oliver O'Brien, Derek Vogel
CSCE 361 - Software Engineering

1. Introduction

The purpose of this document is to establish the high level architecture and relations of our application, Geographical Recon Energy Grub Guide. This document will contain relation diagrams to represent the architecture of our application. The intended audience of this document are those familiar with software development and system architecture who are interested in developing and maintaining our application.



2. Architecture

2.1. Introduction

The high level architecture of our application will be a layered model. The highest level of our application will be the GUI layer. Because this is an android application, our GUI will largely be composed of XML/HTML, CSS, and Javascript. The next level, the logical layer, will utilize Java to provide necessary functions for our application. Our data access layer will use JDBC (Java Database Connectivity) to access our database. Finally, we will use our CSCE MySQL accounts for all database functionality and storage of information.

2.2. Modules

2.2.1. Database Layer

The database of our application will store all vital information our application will need to function. This will include user accounts and

associated information, all stored pins and associated information, and comments. Our application will use MySQL for all database functionality to define objects and relationships.

2.2.2. Data Access Layer

This layer is responsible for connecting the logical layer of our application to the database layer. Our application will utilize Java Database Connectivity (JDBC) to accomplish this. JDBC will convert any Java objects we need stored or retrieved from the MySQL database.

2.2.3. Logic Layer

This layer is responsible for all necessary functionality for our application. Because this is an android application, we will be using Java to control the logic of the system. This layer will provide all information to the GUI layer, and will reflect changes to the database/system as needed. The logic layer will also use the data access layer to store information in the database.

2.2.4. GUI Layer

This is the layer in which the user will be interacting with the application and will be responsible for viewing the campus map as implemented through the Google Maps API. In this the user will be able to drop pins, view previously dropped pins, login to the system, and traverse the map.

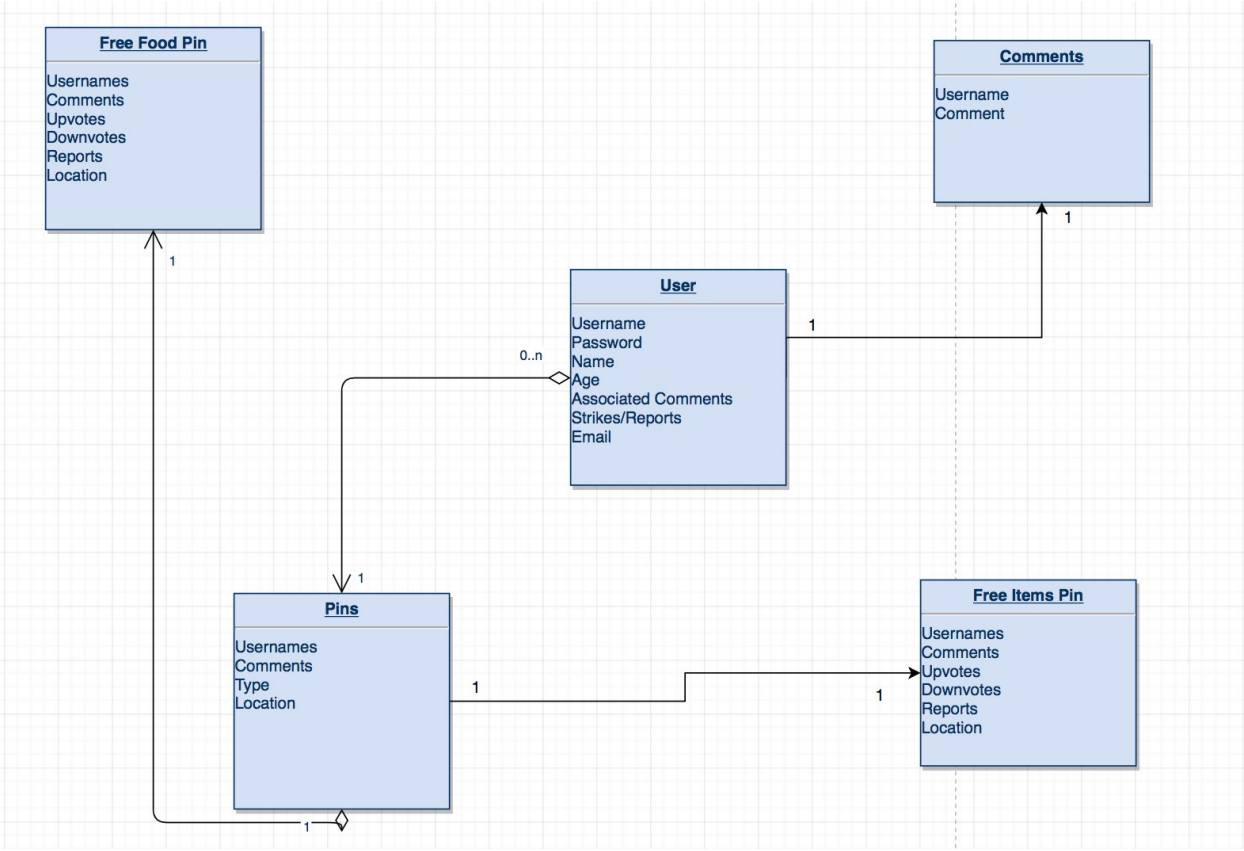
This will be accomplished using a combination of Java, JavaScript, CML, CSS, all while incorporating the Google Maps API for the base map functionality. In addition to the previous functionality mentioned, the user will also be able to report other users, vote on whether a pin is still valid, among other functionality.

3. Class Diagrams

3.1. Data Table Classes

This application will utilize MySQL for all data storage. This information includes user accounts (including user information like email, name, age, username, etc.), pins (including information about the pin such as location, title, description, etc.), and comments.

3.1.1. Schema



3.1.2. Schema Information

3.1.2.1. User

This will hold all the user's information, i.e., their username (and email), their password, their name, their age, comments associated with their account, and strikes/reports against them.

3.1.2.2. Pins

This will hold all pins that are dropped on the map, store their type (so that it can be passed to the correct subcategory), the username associated with the account that created the pin, the location where they were dropped, and comments left about the event.

3.1.2.3. Free Food Pins

Along with the information stored by generic pins, free food pins will also store a description of the free food.

3.1.2.4. Free Items Pins

Stored in this section will be information about the free item pins. This information will be the usernames who dropped

the pin, the comments left about the pins, the upvotes and downvotes about the event, any reports about the event, and the location at which it takes place.

3.1.2.5. Comments

Comments will store comment text and the username of the author.

3.2. Class Information

Classes will be implemented in Java using the data access layer and will be used to control functionality of the application. The above schema will be implemented in our logic layer to create Java classes, which will then be used by the GUI layer to represent data for the user to interact with.

3.3. GUI Layer

The GUI layer will consist of four pages: a login screen, a create account page, the main map page, and the drop and select pin screen. When a user first opens G.R.E.G.G., they will be brought to the login screen. Here the user can input their login informations which is verified through the database to make sure it matches, or an error is projected. If it is valid the app opens up the map based on the location of the user. If no user location is detected, it defaults to City Campus.

In addition, this first screen can access a create account screen if the user is using the application for the first time. Here the user inputs a username, password, “@huskers.unl.edu” email, their name, and email. Once this process is completed the user can login with the supplied information which then brings them to the main screen.

In the main page, it starts out by loading in the Google Maps API and the user is brought on the map to where their location is set to. Here they can also traverse the map viewing pins that were already set and verified by other users. In viewing these pins, the user can up vote a pin to verify it, down vote pins that are expired, and report the user for offensive material.

Lastly, the user can determine a spot in which they would like to drop a pin. Another screen is brought up in which the user can select the type of pin they would like to drop and a description. Once the pin is dropped it will only be viewed by the user who dropped it for a time. For the pin to become public, at least two other users have to drop a pin in the same area which brings up a suggestion, which is the pin of the original user. If two external users accept the

suggestion then the pin will become public and will be subject to upvoting and downvoting.