**Detailed Design Document**

**Voluntold**

**Volunteer Networking Application**

Prepared by: Gianluca Bastia, Arunn Chanthirakanthan, David Hudson, Nicholas Perkins, Brian Wolf

**Table of Contents**

1. Introduction……………………………………………………………………….……3
2. Technologies…………………………………………………………………….…….5
3. Module Design
   1. Front-End………………………………………………………………………6
   2. Back-End………………………………………………….……………………7
4. Database Design……………………………………………………………………...8
5. Rest Endpoints………………………………………………………………………...9
6. Proof of Concept………………………………………………………………………10
7. Test Cases……………………………………………………………………………..11

1. **Introduction**

Voluntold was created to facilitate volunteer event discovery and involvement. Increasing event awareness will help strengthen communities, by providing more accessibility to local events and opportunities to work with others toward a common goal.

The application will utilize a back-end web service to handle database requests and requests for user authentication; the front end of the application will be built as a web app, and will make use of bootstrap for cross-platform functionality (to support both desktop and mobile web browsers). The web app was chosen as the primary launch platform because of its versatility--a web app is portable across a wide range of devices, from iOS devices to Windows desktops. The wide reach of the Internet will help our application, and volunteer events, reach as many people as possible.

The back-end will be inaccessible to the user. A REST interface will be used to connect this back-end to the user interface, or front-end, of the application. The back-end will be created and tested on a local machine.

The application works by prompting users to log in with an existing account; users without an account may utilize the create account function to, as expected, create a new account. Following successful login, the user will be directed to the main page. This page will display events within a user-specified range. Clicking on an event will allow the user to see additional event details, and will allow the user to sign up for the event as a volunteer. The top portion of the main page will contain 3 search fields--these will allow a user to search events by location, or date.

The application will also have a create event page, allowing the user to create a new event by filling in a form; the form will contain a name, location, date, and start time. A profile tab will allow the user to see events they have created and events they have signed up for. This page will also allow users to change their password.

**2. Technologies**

REST (Representational State Transfer) - REST is a back-end architectural style that is often used in Web development. REST is good for internet use, as well as mobile applications, because it lacks the need for bandwidth.

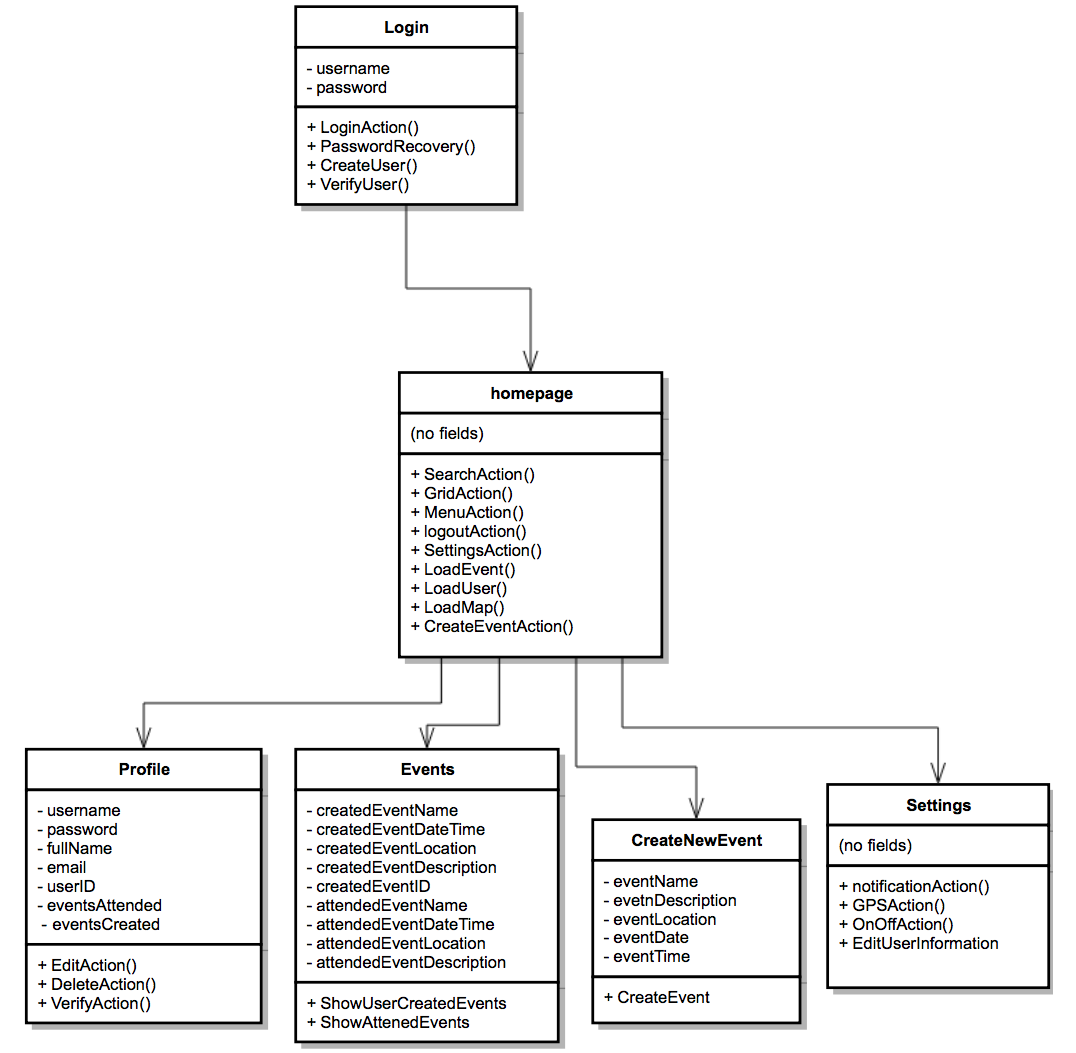
MySql - MySql is an open source relational database management system. MySql is currently owned by Oracle corporation. Most of Mysql is run via the command line. Mysql is a high level language allowing easy integration into the backend.

Google Maps API - Google Maps API is a popular mapping API that allows a programmer to utilize Google Maps functionality. It provides a wide host of services that will be fundamental to the application. Functions such as data visualization and directions will provide helpful navigation to events, as well as location services to search for events within a given a given mile radius. The primary use of the Google Maps API involves geocoding, using an address to retrieve a location’s latitude and longitude coordinates.

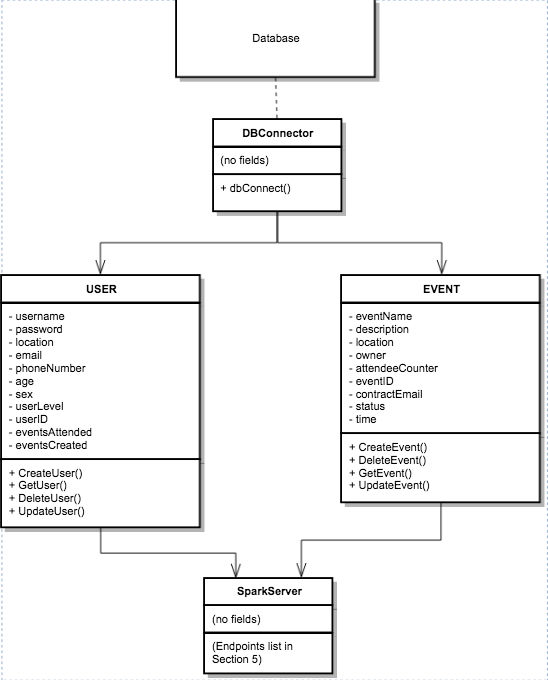
Bootstrap - Bootstrap is a very popular HTML, CSS, and JS framework for developing responsive programs on the web. Bootstrap will be instrumental in front-end web development. It will also scale the web app to fit on a wide range of devices, while only requiring a single code base.

**3. Module Design**

1. Front-end



b. Back-end

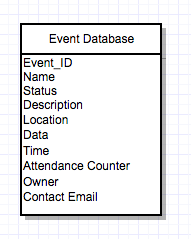


**4. Database Design**

Voluntold uses one database with multiple tables to hold different types of information. Event information is stored in an event table, while user data is stored in its own user table.

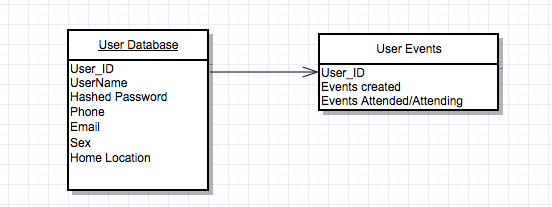
Event Table

Contains information about events that are being held, will contain all pertinent information from event name, location, owner, etc.



User Table

Will contain usernames for all users, will also contain hashed passwords for users and will be used for authentication. This database will contain information such as email, phone, and user level for all users.



**5. Rest Endpoints**

Login Page

Login

-Get- User/username/Password

Create user

-Post- User/username/Password/Email/Location/Age/Phone

Homepage

Search (Location)

-Get- Events/Search/location/

Search (Name)

-Get- Events/Search/name/

Search (Date)

-Get- Events/Search/date/

Search

-Get- Events/Search/date/location/name

Profile

Retrieve user information

-Get- User/username/location/email/eventsAttended/eventsCreated

Change user password

-Put- User/password/username/NewPassword

Create Event

Make new event

-Post- Events/name/description/location/owner/time/contactemail

Retrieve event id from database

-Get- Events/EventID/name/location/owner/EventID

-Put- User/username/eventsCreated

Event View

View event details

-Get- Events/name/description/location/owner/time/contactemail/eventID

Volunteer (Event signup)

-Put- User/username/eventsAttended

-Put- Events/EventID/attendenceCounter

**6. Proof of Concept**

To promote familiarity with the necessary technologies, we designed proof of our concept. This proof of concept will help minimize the learning curve for new tools and technologies during the development phase.

Our proof of concept demonstrates the use of spark server and mysql. We made a simple database in mysql called names that holds the first and last names of people. Using a database connector, the database was connected to our Maven project. Utilizing the tools imported from spark, we created a simple endpoint that returns all the names from the database in the JSON format. The data was parsed to JSON using Google’s GSON library.

The proof of concept can be found online at

<https://github.com/perkins109/Senior-Proj-Volunteer-/tree/master/Rest-Test>

**7. Test Plan**

Login Page

Logging In

Authenticates users based on predefined username and password combinations.

|  |  |
| --- | --- |
| Case | Expected Result |
| Both Fields Empty/Not Found | Invalid user/pass |
| Username Empty/Not Found | Invalid user/pass |
| Password Empty/Not Found | Invalid user/pass |
| Valid Info Entered (both fields) | Authentication Successful |

Creating User

Creates a new account in the database.

|  |  |
| --- | --- |
| Case | Expected Result |
| Empty data fields | Missing data field |
| Any/all fields contain invalid data | Prompts user to enter requisite info |
| Filled data fields w/ valid data | Account creation successful |

Home Page

Allows users to search for nearby events, select their profile or settings, and logout.

|  |  |
| --- | --- |
| Case | Expected Result |
| Empty search fields | Error, no results found |
| Search field filled | Displays corresponding events |
| Log out | Logs user out |

Profile

Contains user event information, attended and created

|  |  |
| --- | --- |
| Case | Expected Result |
| Click on Events Created | Displays events created by user |
| Click on Events Attended | Displays events attending |

Create Event

Creates a new event in the database with time and location information.

|  |  |
| --- | --- |
| Case | Expected Result |
| Required field(s) empty | Prompts user to enter requisite info |
| Any/all fields contain invalid data | Prompts user to enter requisite info |
| All fields filled appropriately | Creates new event in DB |