# SW Engineering CSC648/848
# Section 2 Spring 2018

## WWW Site for Reporting and Managing Environmental Issues
## "Parks Go Green"

Team 13
Damico Shields (dshields@mail.sfsu.edu)
Leo Wang
Jaimes Subroto
Justice Chase
Kimyou By
Andrew Hutzel

## Milestone 2
March 21, 2018

| Version | Date | Description |
|---|---|---|
| 1.0 | 3/21/18 | Initial document submission |
| | | |
| | | |

1. **Data Definitions**

1. **Unregistered user**: Can browse and read reports. Can submit reports if an email/phone number is provided. Stored as email and phone number.
2. **Registered user**: Can browse, read and submit reports. Must log in with a username and password that they create (and is stored). Can access a list of reports they submitted. Can post images of the park.
3. **Administrator**: Must log in with a username and password. Must have been given access by the institution using *Parks Go Green*. Can browse and read reports. Can remove reports that contain inappropriate content.
4. **City Manager**: Must log in with a username and password. Must have been marked as a city manager by the institution using *Parks Go Green*. Can browse, read and submit reports. Can change the status of the report and post images of the park.
5. **User type**: denotes the type of user (1 for admin, 2 for city manager, 3 for registered, 4 for unregistered).
6. **Park**: Name, Street address, Google Maps Data.
7. **Report**: Submission date, category, park concerned, description of issue, status, date of each change in status, image, ID of report author (viewable by administrator).
8. **Status**: Possible report statuses: Submitted, In Progress, Resolved.
9. **Category**: Possible type of environmental issue: Oil spill, Garbage, Medical waste, Bathroom.
10. **Registration record**: Registration information on users: **email** address, **phone number**, **username**, **password**.
11. **Image**: A photograph of a park. Submitted by **registered users** or **city managers**.
12. **Activity logs**: A record of activity by **users**.
13. **Captcha**: Verification tool to filter invalid posts.
14. **Terms of Service**: Agreement to not abuse the service, required to register an account.

## 2. **Functional Requirements**

Priority 1:

1. **Unregistered** and **registered users** shall be able to post **reports** on environmental issues at local **parks** to *Parks Go Green*, either by registering or by providing an **email** and **phone number**.
2. **Unregistered** and **registered users** shall be able to read **reports** on environmental issues at local parks posted by other **Unregistered** and **registered users**.
3. **Unregistered** and **registered users** shall be able to find **reports** related to **parks** they are interested in researching.
4. **City managers** shall be able to read **reports** and to change their **status** (from Submitted to In Progress or Resolved) to indicate progress made on fixing the reported issue.
5. **Administrators** shall be able to review **reports** and **images** before they become visible and delete those they deem to contain inappropriate content.

Priority 2:

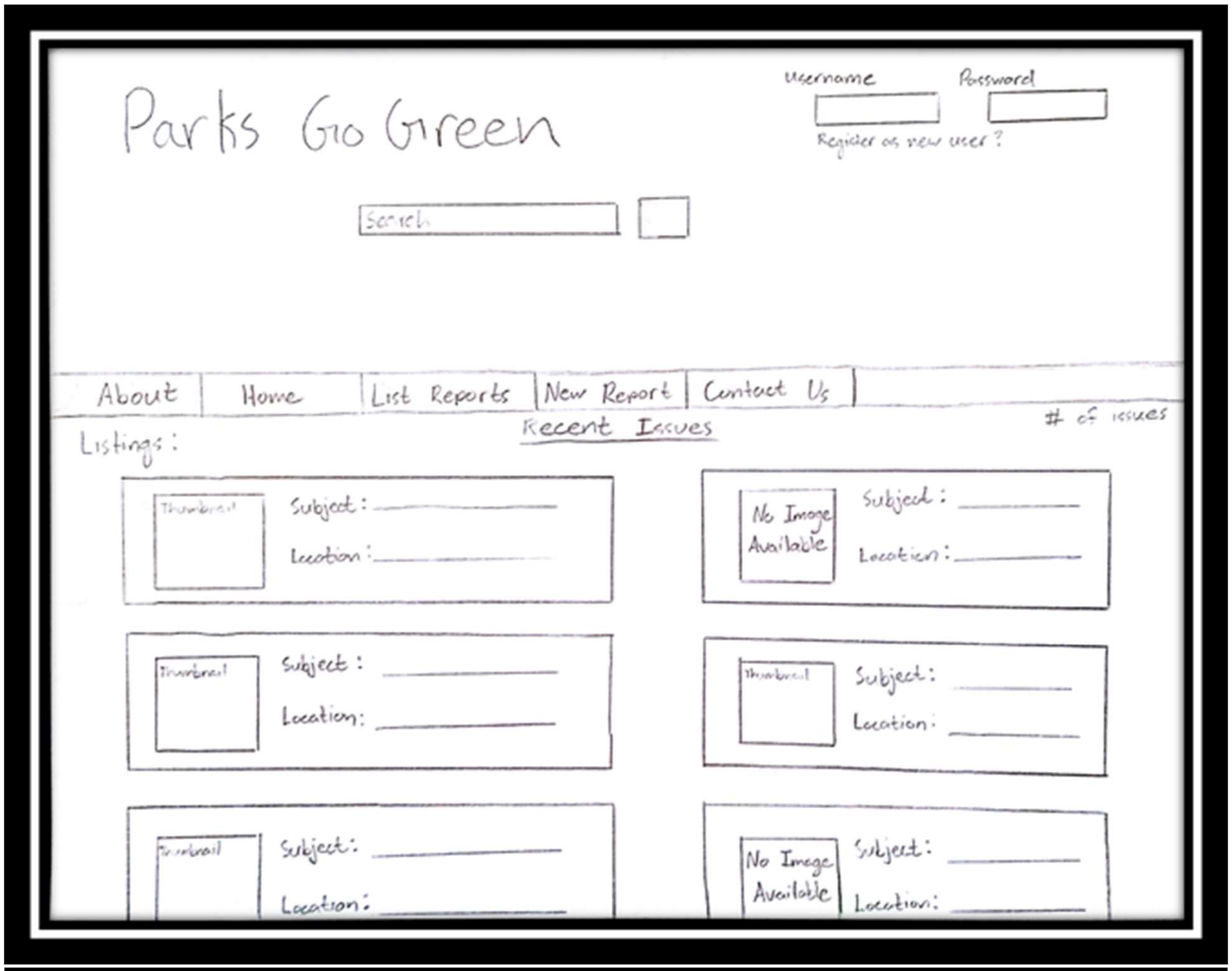
6. **Unregistered users** shall be able to register an account with a **username** and a **password** and by agreeing to the **Terms of Service**.
7. **Registered users** shall be able to upload **images** and attach them to **reports** they submit.
8. **City managers** shall be able to upload **images** of **parks**.
9. **Administrators** shall be able to review **activity logs**.
10. **Administrators** shall not be able to modify **reports**.

Priority 3:

11. **Unregistered users**, **registered users**, and **city managers** shall be able to view the **park** location in a Google Maps display.

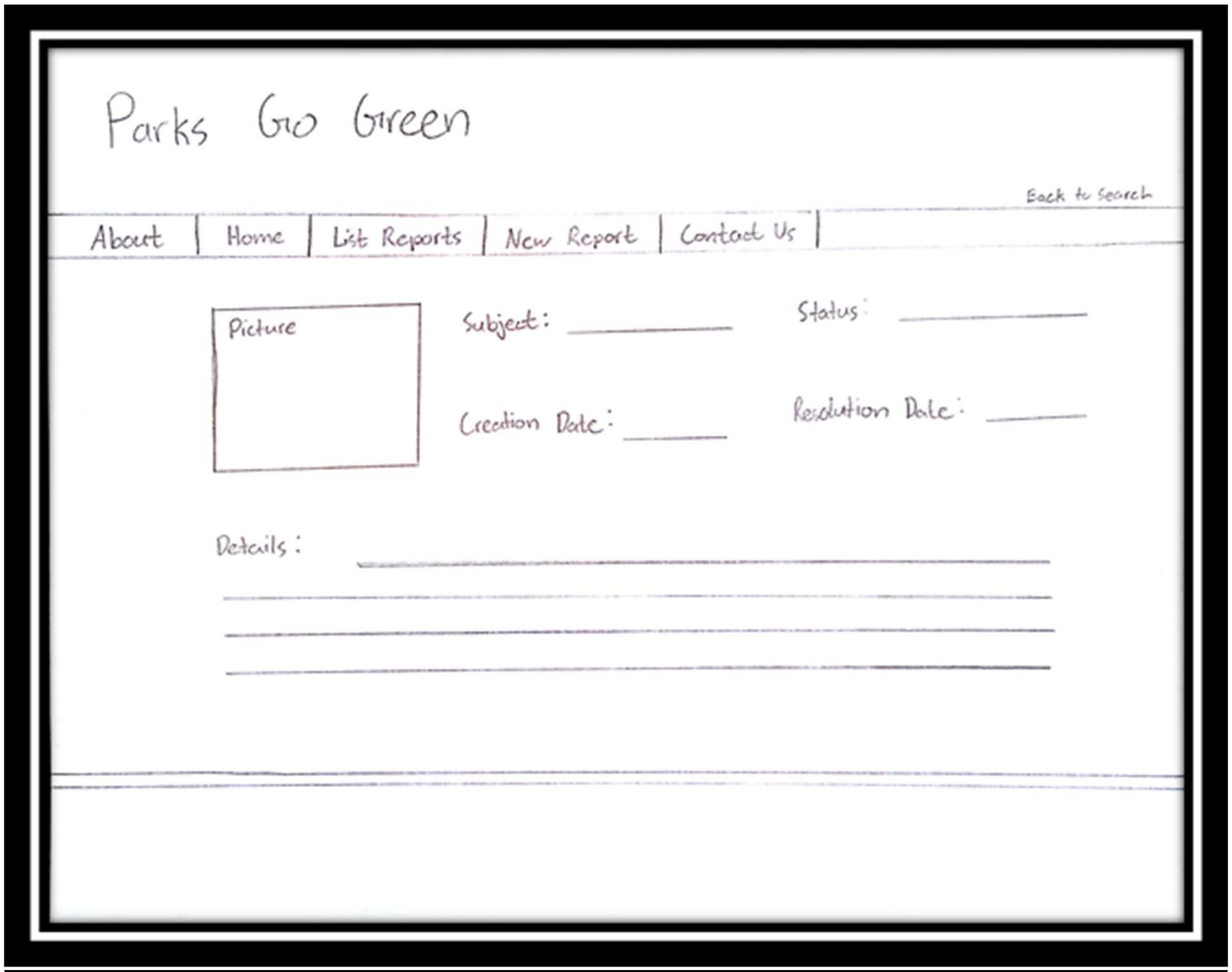
3. **UI Mockups and Storyboards**

**Main Page**



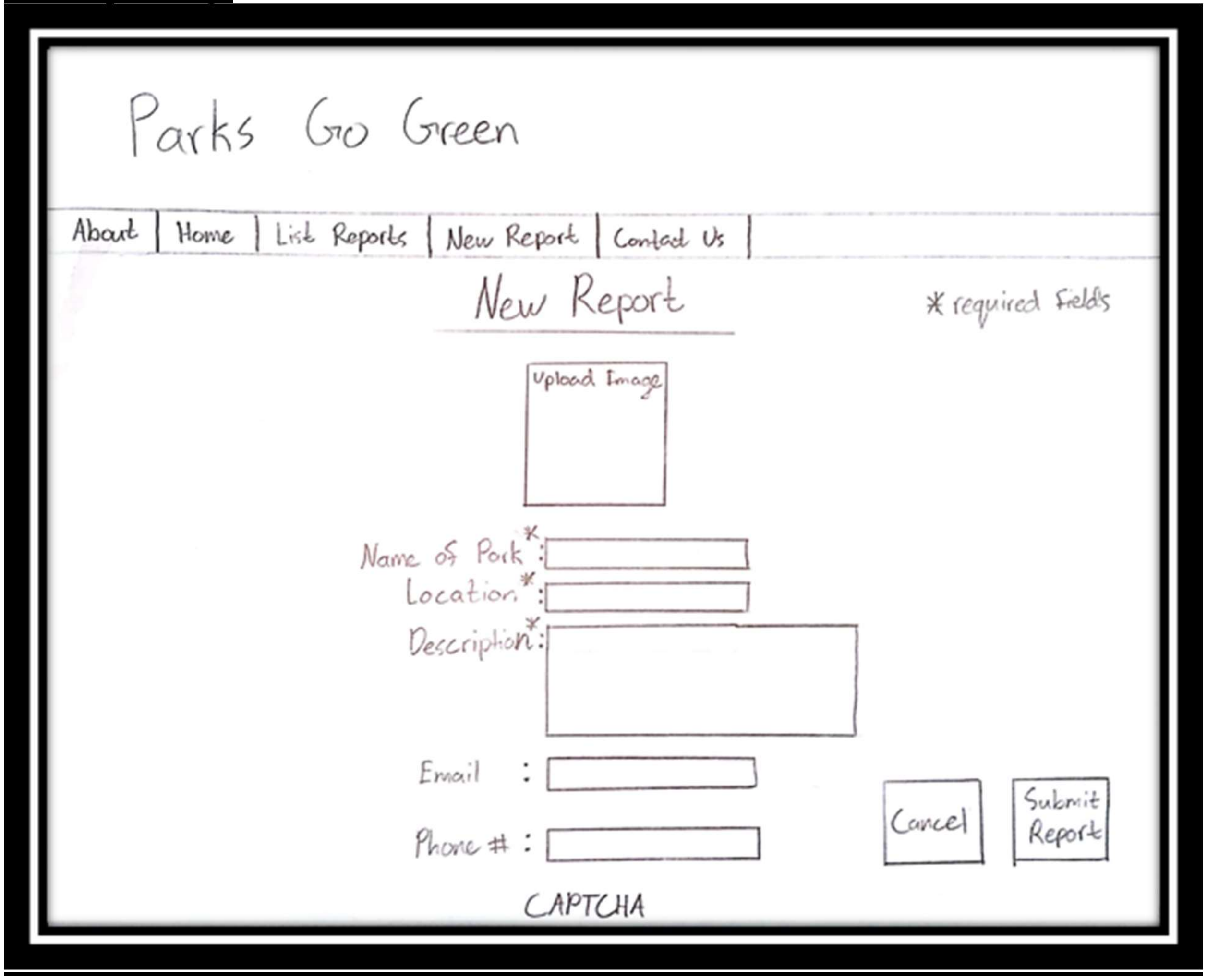
Users of all sorts land here on the main page. A search bar features prominently, with a login panel in the upper right. Below, a set of links hovers above a list of recent reports. Lisa, an **unregistered user**, clicks one of these recent reports, which leads her to an instance of the next page.

**<u>Issue Page</u>**



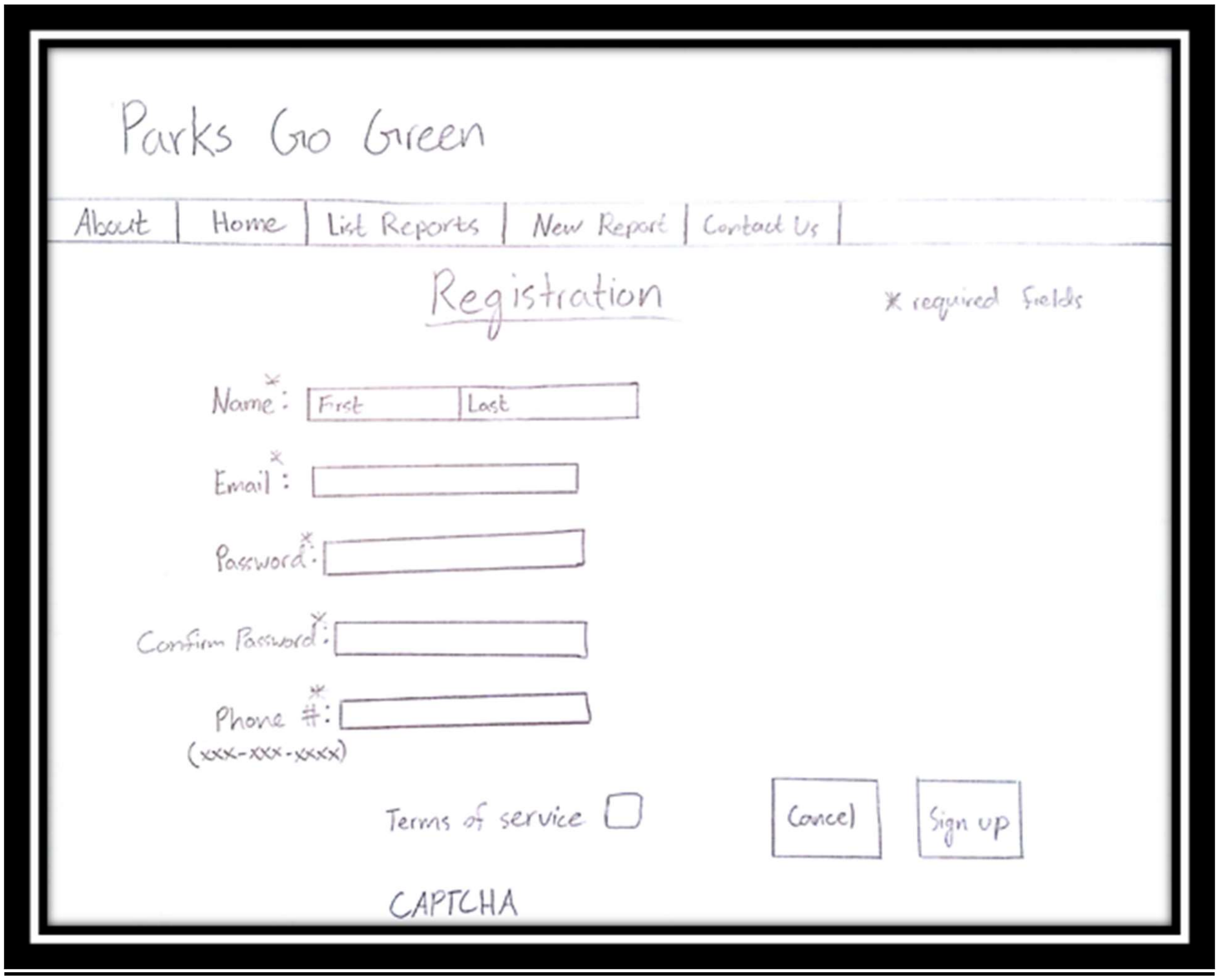
Lisa, an **unregistered user**, clicked a recent **report** to check if it was the issue she noticed at her local **park**. Here she sees all the relevant information about the park, as well as the links available from the previous page. She decides that this issue is different from the one she witnessed, so she clicks the New Report button.

**New Report Page**



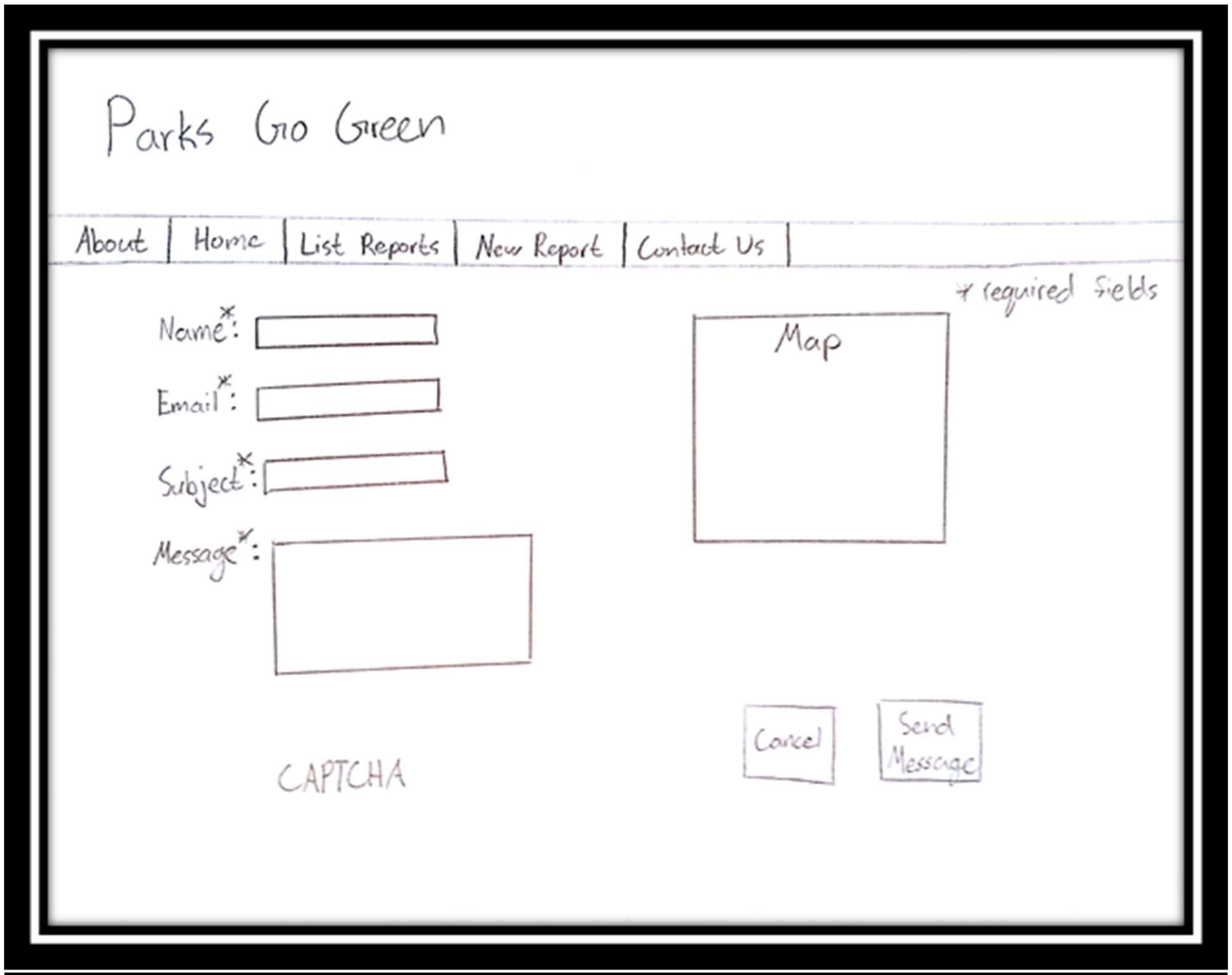
Having clicked New Report, Lisa is brought to this page with a form. As an **unregistered user**, Lisa will not be able to upload an **image**. She enters the relevant information about the **park**, supplies her personal information, solves a **captcha**, then hits the submit button.

**<u>Registration Page</u>**



Lisa has decided she wants to be able to upload **images**, so she decides to register. Here she enters her relevant information, then agrees to the **terms of service** and solves a **captcha**, then hits the sign-up button.

**Contact Us Page**



Some time later, Lisa has moved to another city and has been hired as the **city manager** for **parks**. She remembers *Parks Go Green* and wants to implement the service in her new city. She clicks the contact us page to send us a message.

## 4. **High Level Architecture, Database Organization**

Model: Database with several tables: **Categories**, **Statuses**, **Users**, **User types**, **Parks**, **Reports** for each category.

1. **Categories** will be the types of environmental issues (Oil spill, garbage, bathroom, medical waste)
2. **Statuses** will be the status of the report (Submitted, In Progress, Resolved).
3. **Users** will the list of users, with the **phone number** and **email** of the **unregistered users**, that plus the **username** and **password** of the **registered users**, **admins** and **city managers**, and the **user type**.
4. **User types** will simply have the type of user (1 **admin**, 2 **city managers**, 3 **registered users**, 4 **unregistered users**).
5. **Parks**: The name of the park, the text of the address, the Google maps data of this park.
6. **Reports**: Will have a **category**, **submission date**, the **park**, the **user**, the **description**, the **status** and the file path to an **image**.

View: To find **reports**, a search bar can be filled out, and the search button will then send the text to the controller. The controller will return the results, which will be displayed on a new page. They can be clicked, which will display the information on an **issue page**.

To register, a registration form can be filled out, which is then sent to the controller. The user is then brought back to the page they started the registration process from, now with new options available if the registration was complete.

To submit a report, the user is shown the **New Report Page**, where they can fill out the forms then submit the information.

Controller: After a user presses the search button, a function will query the database using the string entered in the search bar. Any results will be returned as a list to be displayed on the results page.

After a user submits registration data, the controller will create a new **user** with the information supplied.

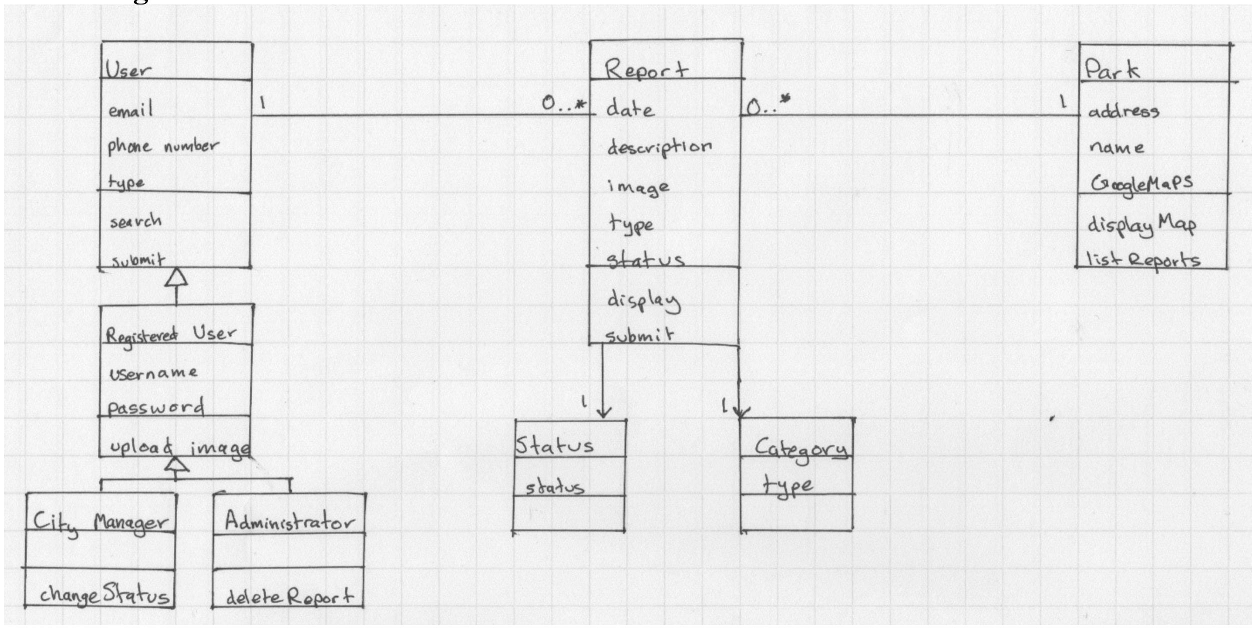
Once a report is submitted, the controller will create a new report by reading the submitted data and retrieving the relevant types from existing tables to fill in a new **report** entry. The **image** will be stored in the file system and the path will be entered in the entry.

5. **Content for vertical prototype**
This is an extremely rudimentary single table with none of the relationships described earlier, only a few text fields, as we had a very tough time with this milestone.

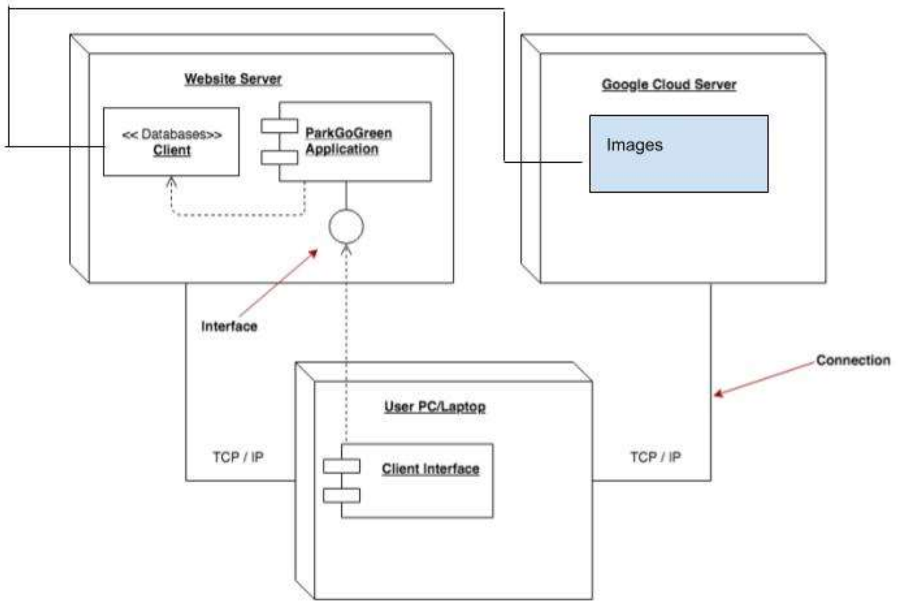
John McLaren Park | 100 John F Shelley Dr. San Francisco, CA 94134 | fine
Mission Dolores Park | Dolores St & 19th Street San Francisco, CA 94114 | bear on da loose
Glen Canyon Park | Elk St and Chenery Street, San Francisco, CA 94127 | ufo sighting
Heron's Head Park | Cargo Way & Jennings Street, San Francisco, Ca 94124 | amazon web services

## 6. **High Level UML Diagrams**

### Class Diagram



**Component and deployment diagrams**



## 7. Key Risks

**Lack of skill**: Everyone in the group is completely inexperienced with developing and deploying web applications more complex than the various HTML edits made in the CSC412 lab. While some group members have some Python experience, it is only in writing basic applications.

So far, our only solution has been to try and find guides for each step of the way, but we feel like we are somewhat stonewalled by the fact that every guide is specific to the software stack the guide's author had in mind.

**Scheduling conflicts**: Most group members are in more than one of the same classes, which doesn't make it hard to meet, but it does mean that if one of us has work to do for another class, almost all of us share that responsibility, so we cannot have some people working while others take care of other responsibilities compared to a group with more varied classes.

We will strive to parse time between these classes, but while this course is highly structured in terms of deadlines, the others are far more chaotic.

**Teamwork risks**: There is a disparity between the back end and the front end. The back end has been more independent in pursuing the milestone goals, while the front end has required more direct instruction. We had hoped it was otherwise, as we have 2 backend and 3 frontend members plus the leader, who was hoping to fill in more in the backend.

We will have to focus on concrete task assignment for the frontend team to ensure that work is spread out more evenly.