

SW ENGINEERING CSC648/848

SUMMER 2019

Milestone 2 - Team 07

Erick Velez (evelez1@mail.sfsu.edu)

Kevin Paredes

Jackie Shan

Vincent Santos

Hector Aguirre

Jimmy Kwan

Date	Revision History	Comments
7/13/2019	1.0.0	Initial Draft

July 13, 2019

Table of Contents

1. Data Definitions	2
Entities.....	2
Unregistered User.....	2
Registered User.....	2
Administrator	2
Issues.....	3
2. Function Requirements.....	4
Priority 1.....	4
Priority 2.....	4
Priority 3.....	5
3. UI Mockups and Storyboards	6
4. High level Architecture, Database Organization.....	12
DB Organization	12
Media Storage	12
Search/Filter Architecture and Implementation.....	12
Automatic Categorical Priority.....	12
Sorting by Categorical Priority.....	13
Database API.....	13
New Software Tools.....	13
5. High Level UML Diagrams.....	14
Class Diagrams.....	14
Component Diagrams	15
6. Key Risks.....	16
Skill	16
Schedule	16
Technical.....	16
Teamwork.....	17
7. Project Management.....	18

1.Data Definitions

Entities

Unregistered User

The Unregistered User is the default User type for all visitors to the web application.

Data

No information of an Unregistered User will be saved.

Privileges

Unregistered Users have view-only privileges. They will be to search (using the search bar and the map) and view all Issues on the application.

Registered User

Once an Unregistered User registers, they become a Registered User.

Data

The following data will be stored for Registered Users upon registration:

- Name
- Email
- Password

A Unique User ID (UUID) will be generated for a Registered User upon registration.

Privileges

A Registered User will have read and restricted write privileges. In addition to the privileges of an Unregistered User, Registered Users will be able to generate Issues viewable by all Users. They will be able to delete self-generated Issues, but they will not be able to delete the Issues generated by other Registered Users.

Administrator

Administrators cannot be created like Registered Users. They must be manually added by the development team.

Data

Administrators will contain the same data as Registered Users.

Privileges

An Administrator will have read and write privileges. In addition to the privileges of Registered Users, Administrators will be allowed to delete Issues generated by any Registered User. They will also be able to change the status of Issues.

Issues

Issues form the majority of the application's functionality. Issues are created by Registered Users and display and contain information about environmental problems that they might have encountered.

Data

All Issues will have the following properties:

- Description: A user-supplied text description of the Issue
- Location: latitude/longitude of Issue
- Images: Optional user-supplied images of Issue. Max 3 per Issue, 8 MB each.
- Status:
 - Open
 - In-progress
 - Closed
- Categories
 - Fire
 - Oil Spill
 - Biohazard
 - Trash
 - Other

Categories will have a priority assigned to them. When viewed by an administrator, open Issues will be sorted by their category's priority.

2.Function Requirements

Priority 1

Unregistered Users

1. Unregistered Users shall be able to view any issue that has been posted
2. Unregistered Users shall be able to search using a visual map
3. Unregistered Users shall be able to access any previously completed issues
4. Unregistered Users shall be able to register using an email and password, thereby making them a Registered User

Registered Users

1. Registered Users shall have the same permissions as unregistered users (and more)
2. Registered User information shall be stored in a database
3. Registered Users shall be able to post issues
 - a. Registered Users shall be able to include a description of the issue
 - b. Registered Users shall be able to include an image of the issue
 - c. Registered Users shall be able to include the issue in a category
4. Registered Users shall be able to edit the issues they created
 - a. Registered Users shall be able to delete or add images to their posts, after they have been created

Administrators

1. Administrators shall have the same permissions as Registered Users and more
2. Administrators shall be able to login to their own dashboard
 - a. Administrators shall be able to view all issues in their dashboard organized by status (open, in-progress, or closed)
3. Administrators shall be able to update the status of each issue
 - a. Updated issues shall be moved to a different page, respective of their new status
4. Administrators shall be able to delete issues
 - a. Deleted issues shall be removed from the database
 - b. Deleted issues shall not be visible to any user through search
 - c. Deleted issues shall not be visible to any user through the map

Priority 2

Unregistered Users

1. Unregistered Users shall be able to search using specific locations
2. Unregistered Users shall be able to search using zip codes

Registered Users

1. Registered Users shall be able to view a history of their posts.

Administrators

1. Administrators shall be able to view the list of open Issues sorted by priority

Priority 3

Unregistered Users

1. Unregistered Users shall be able to view a timeline of all Issues at a specific location
 - a. The timeline shall consist of all Issues at the location sorted by date

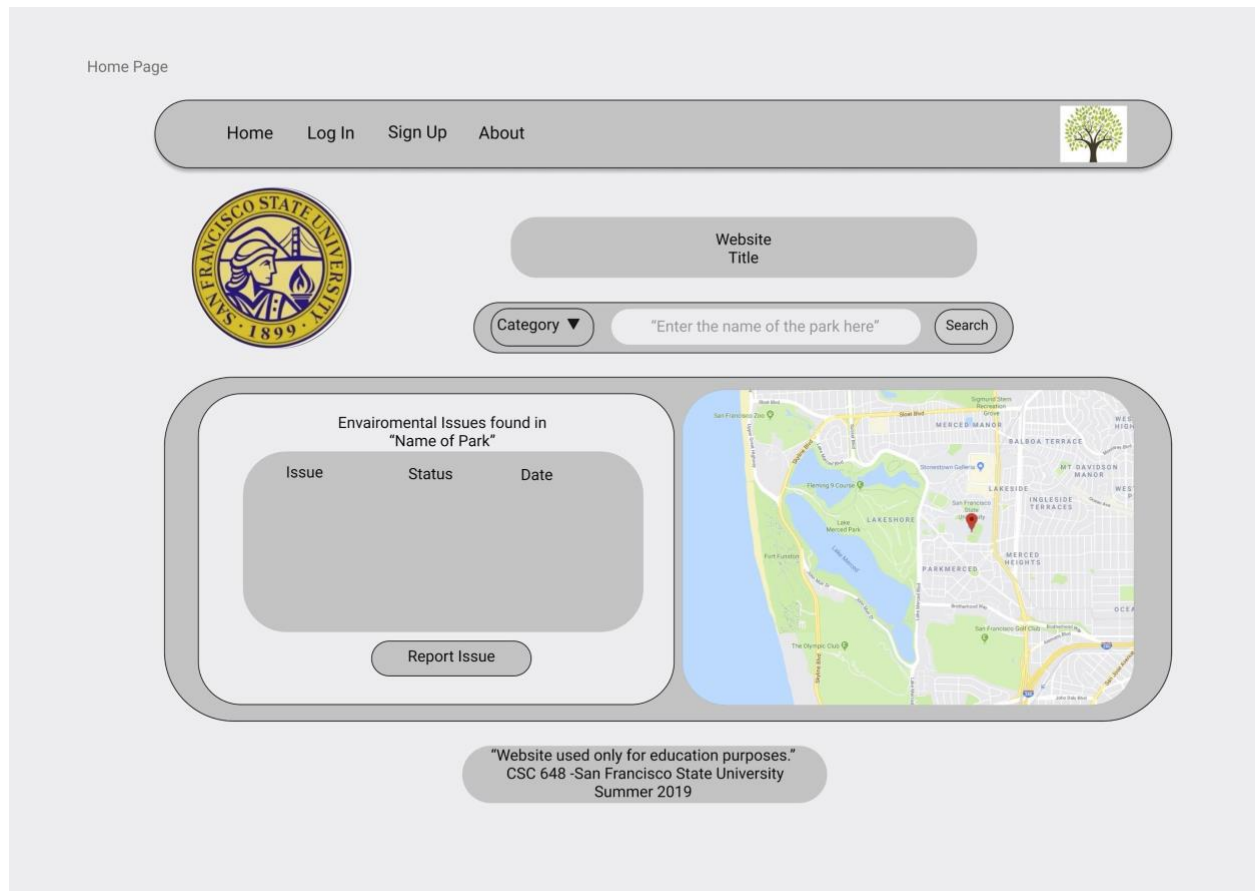
Registered Users

1. Registered Users shall be able to comment on open and in-progress Issues.

Administrators

1. Administrators shall be able to delete Registered User accounts (for punishment of abuse of Terms and Conditions of application).

3. UI Mockups and Storyboards



Log in

[Home](#)

[Log In](#)

[Sign Up](#)

[About](#)



Username

Password

[Sign up](#)

[Forgot Username/Password?](#)

Log In

"Website used only for education purposes."
CSC 648 -San Francisco State University
Summer 2019



Issues Legend



Waste
Management



Oil Spill



Wildfire

Enviromental issue

Status

Date

Category ▼



Closed ▼

6/8/19



Closed ▼

6/8/19



Closed ▼

6/8/19



Open ▼


6/8/19

Report Prompt

Category ▼

Date

Select Date



Enter report details

Attach Image

Finish

Sign up

[Home](#) [Log In](#) [Sign Up](#) [About](#)



First Name

Last Name

Email

Password

Confirm
Password

Create Account

"Website used only for education purposes."
CSC 648 -San Francisco State University
Summer 2019

Storyboards:

User:

The user will open the "**Home Page**" and insert the name of the park on the search bar then the home page will display a list of environmental issues reported on the park and display a map with a mark on the location of the map. After looking at the status of the issues reported the user will decide if the environmental issue needs to be reported. If the user decides that the issue needs to be reported then the user will click on the report issue button. This will prompt the user to the "**Log in**" page, in this page the user must successfully enter a username and password. If the user does not have an account, the user must click on the sign-up button that would prompt the user with the "**Sign up**" page. After successfully login or creating an account the user will be prompted with the "**Report Prompt**" page, the user must choose the type of issue on the category dropdown then select the date by entering numbers or choosing a specific date by using the calendar icon button. At the end the user must enter a report detail of the issue and attach an image that shows the environmental issue reported.

Admin:

The admin will open the "**Home Page**" and then click on the log in located on the top portion of the page. This will prompt the admin to the "**Log In**" page in which he must successfully enter a username and password. Then the user will be prompted to the "**Admin**" page here the admin can update status of environmental issues reported by using the dropdown status.

4. High level Architecture, Database Organization

DB Organization

Our database will contain 3 tables. These will be comprised of Users, Admins, and Issues.

- User
 - Name
 - Email
 - Password
 - Unique User ID
- Admin
 - Name
 - Email
 - Password
 - Unique Id
- Issues
 - Description
 - Location
 - Images
 - Status
 - Categories

Media Storage

We will be storing all media files as BLOBs in the database. Our application will only be accepting images from users as to keep within the constraints of the instance storage size.

Search/Filter Architecture and Implementation

We will be implementing search by using the LIKE operator in SQL. Users will be able to search for Issues by entering a keyword and we will find similar matches using this algorithm.

We will also have a drop down where users can pick issues by category such as oil spill, fire, hazardous waste, etc. This will allow specific searches if used in addition with a keyword used in the search bar.

Automatic Categorical Priority

Issues will automatically receive a priority rating determined by their category, which is specified by the User upon creation of the issue. Priority ratings for each category will be determined by

the development team, including an Other category that will be an umbrella category for obscure Issue types.

Sorting by Categorical Priority

Every Issue will have an inherent priority based off of its category. For example, an Issue under category Fire will have a higher priority than an Issue under the Trash category. When an administrator goes to view all active issues, the list will be sorted first by status (open Issues first, then in-progress) and then sorted by priority. Sorting will happen on the server-side. The list that the client receives will already be properly sorted, and thus only needs to be iterated over to display the Issues on the administrator dashboard.

Database API

Our application will include an API for querying the database that is universally known to both the backend and frontend teams. The purpose of this API is to be a link for the frontend and backend teams.

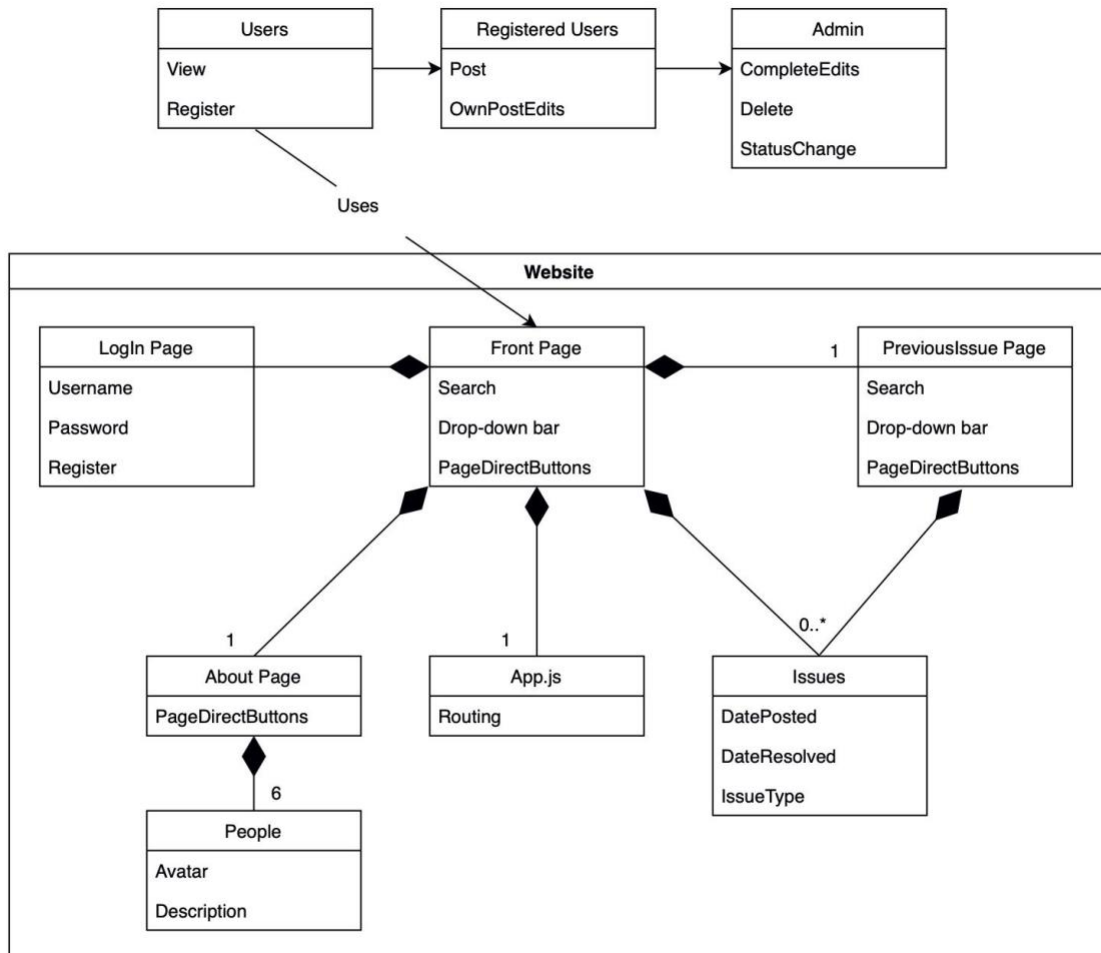
The API will provide utility functions for any request the user might make on the frontend side of the application (such as querying all Issues, querying Issues with filters, etc.). Having a concrete list of database API functions will allow the backend team to focus on certain aspects of database management instead of having to guess what request will be made by the user. Instead, a specific function will be called from the database API.

New Software Tools

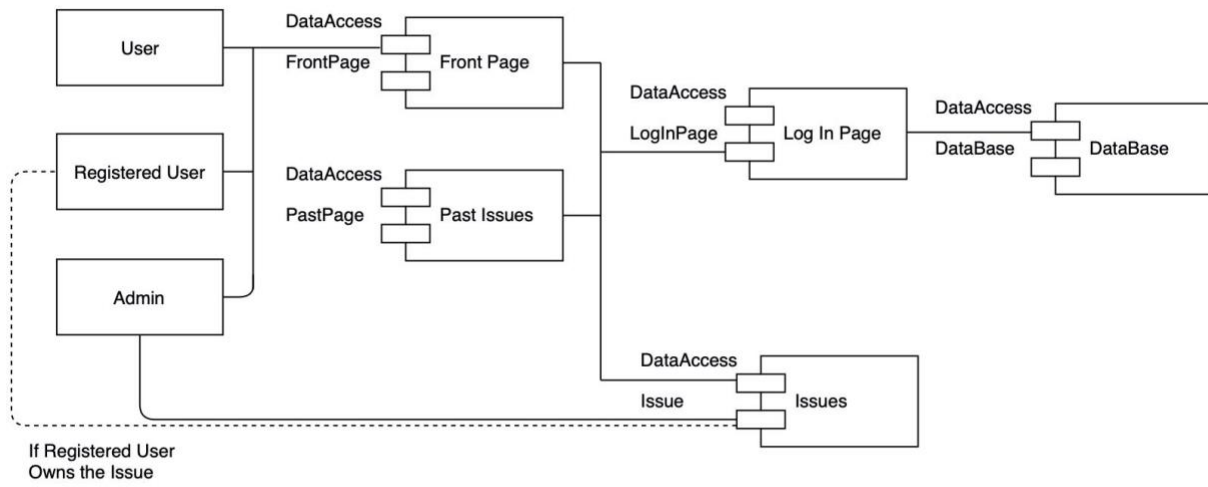
We have added the Google Maps JavaScript API and the Google Maps Places library to our software stack. We will be leveraging the Google Maps JavaScript API to display the map feature of our web application. The Google Maps Places library allows to use Google to cross reference a location's coordinates (latitude and longitude) with its name (which we might not be aware of).

5. High Level UML Diagrams

Class Diagrams



Component Diagrams



6.Key Risks

Skill

As a student development team, some of our team members might not have as much experience in developing web applications as others. According to a self-survey, no team member is an expert or very skilled in JavaScript (out of 5, the highest rated team member in JavaScript is a 3). These two factors might make it difficult in successfully delivering a product, or at the very least, hinder and slow down its development.

To counter these negative factors, those who are more familiar with certain concepts (such as databases) should explain their functionality to the rest of the team. The team should also spend individual time in studying JavaScript language implementations.

Schedule

As a student development team, the time we have to commit to the product will pose a risk to its timely completion. Our team members have obligations for other classes and even personal situations may arise.

To try to counter this as much as possible, good project management is essential. Having everyone know which tasks are assigned and monitoring their progress on the task will help prevent unknown issues from popping up, like a task being forgotten about.

Technical

The map will allow Users to view open and in-progress issues wherever they may be scrolling. One question that may arise is which Issues should be loaded and when should they be loaded.

For example, if the map can detect the User's current location, then all issues within a mile radius should be loaded and shown to the User on the map. If the User scrolls away from their current location, then other Issues around that area should be shown, but they can either be preloaded (which might require all current active Issues be loaded to be displayed on the map, or only those within a certain distance of the User's original current location) or that they be loaded only when the User changes the viewport of the map.

The first method might include a greater initial load time, but will guarantee that all Issues are ready to be shown without having to wait for new Issues to be queried from the database if the viewport of the map changes. However, it potentially requires that irrelevant Issues that will not be shown to be loaded. The second method ensures that only relevant Issues are queried, but will require more client-server communication.

Teamwork

The student development team operates somewhat remotely. We are not able to cooperate in person everyday and rely on remote work to complete our tasks, which might cause miscommunication on the requirements of our product.

The best way to remedy this is good project management, such as utilizing a system that tracks all in-progress and completed work, as well as what still needs to be done.

7. Project Management

For project management, Github's Project feature will be used throughout the development of our web application. There were two main reasons for using Github Projects over a service like Trello:

1. Keeping work centralized
2. Integration with our Github repository

Using a site like Trello would add another dependency for the student development team to have to keep track of and visit. Having our project management on Github makes less clutter in the sites we have to keep track of, as well as providing easy access to our source code.

Additionally, Github Projects allows direct reference to Github Issues, which is useful for keeping track of what bugs are currently within our source code. Being able to reference a Github Issue with an in-progress task might be useful to keep information coherent.