Milestone 2

Spill

Spring 2018 SW Engineering CSC648/848

Section 01 Team 06

3/18/2018

Peter Mutch (peter2mutch@gmail.com)
Satjit Bola
Alaric Gonzales
Lorraine Goveas
Albert Fernandez Saucedo
Harpreet Singh

History Table:

Data Definitions

Unregistered User: A person who utilizes the website that is not registered with the database. This user can only access the website as “read-only” and can browse and search the website for posts. This type of user has the least amount of privileges.

Registered User: A person who utilizes the website that is registered with the database. This user is granted special privileges such as posting a new post, editing a previous post submitted by the same user, or deleting a previously created post, as well as all features available to unregistered users.

Admin: A registered user with special permissions geared toward keeping the website running as intended and removing inappropriate content. Site admins are also responsible for tracking and removing/merging multiple posts about the same post.

Post: A user report of an issue they noticed. This post would include a location, time the issue was noticed, a category from a fixed menu that best defines the issue, the reporter’s name, a description of the issue with the option of adding pictures, and status of the issue which would be changing with time.

Status: This refers to the state in which a post is currently. A post can be pending verification, a work in progress, or resolved.

Verified: In this state, the post has been approved by an as an accurate representation of a real-world issue.
In Progress: In this state, the post has been accepted by an admin and is displayed on the main website.
Planned: In this state, the issue is planned for cleanup or resolution but is not being actively treated.
Resolved: In this state, the issue has been resolved by the appropriate authorities/facilities. The post is changed from in progress to resolved by an admin or the user who originally submitted it.
Locked: In this state, only admins can make changes to the post. The locked state is used for any posts that have been previously verified but now has complications.

Agency: An organization or a business in charge of resolving and setting the status of the issues posted on the website.

Comment: A short response left by a registered user on a post that had information relevant and pertaining to the post.

User Profile: Picture of the user, name, and email. Can be edited by the user in the user panel.

User Panel: See the user's posts, up-votes, and edit profile.

Admin Panel: The resource used by administrators to help mediate content submitted by users on the website.

User Record: A record in the database that stores a registered user's name and e-mail address.

Functional Requirements

Priority 1

Unregistered User

-shall be able to register for an account
-shall be able to search and browse for incidents by location

Registered User

-shall be able to register for an account
-shall be able to search and browse for incidents by location
-shall be able to login to their account
-shall be able to make a post of an environmental problem and upload a photo
-shall be able to view their previous posts
-shall be able to search and browse for incidents by location

Environmental Agency User

-shall be able to view posts for the city they are working for
-shall be able to comment on posts
-shall be able to change the status of posts
-shall have a dashboard to view users and view the modified status of posts

Website Admin

-shall be able to view all posts
-shall be able to remove posts
-shall be able to suspend user accounts
-shall not be able to edit posts

Priority 2

Registered User

-shall be able to mark their post as resolved
-shall be able to comment on other posts

Unregistered User
Environmental Agency
Website Admin

UI Mockups and Storyboards

High-Level Architecture, Database Organization

1. High-level Architecture

The architecture of our web application will be using GCE (Google Compute Engine) and nginx as the server. We will be using MongoDB as the database connected to this server. We will be using node.js to manage the application's server along with express.js and mongoose. The front end will be written in react.js. Spill will be optimized for Google Chrome and Mozilla Firefox browsers, the most recent version and one version previous. The source code for Spill will be hosted on a GitHub repository and developed independently, collaboratively and simultaneously by all Team 06 developers.

We will be organizing our application in MVC system architecture format. The Model will be our MongoDB database which shall contain information about users and reported posts. The View will be the UI which will display data from the database be created using react.js.The controller will perform actions based on input from the user and changes to the model.

2. DB Organization

We will be using MongoDB for our database. Our Database will be organized into two different schemas: Users and Posts.

postRecords
recordId
userId
title
Category
description
status
type
address
city
state

zipcode
location_Lat
location_Lng
locationName
image_Src
image_Height
image_Width
image_Alt

registeredUsers
userId
name
email
password
phoneNumber

3. Media Storage

For media storage, we have decided to store our files in a file system and store the path to each file in our database.

4. Search/filter Architecture

For our search algorithm, we will be using RegExp in our get API method which will allow the user to search for a location from the postRecords schema.

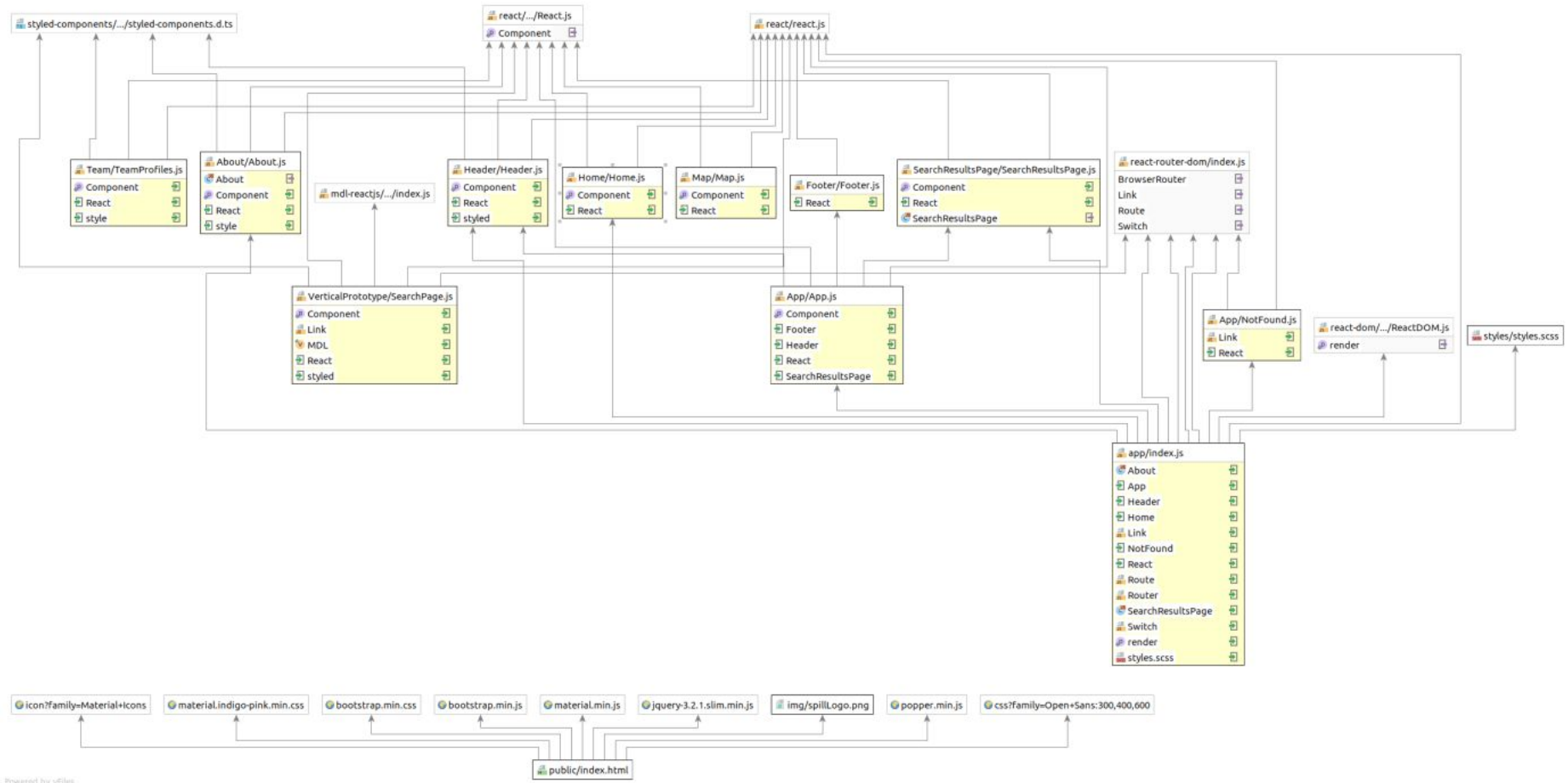
5. APIs

We have created a postRecord API which can get records from the database, search for records and create new records.

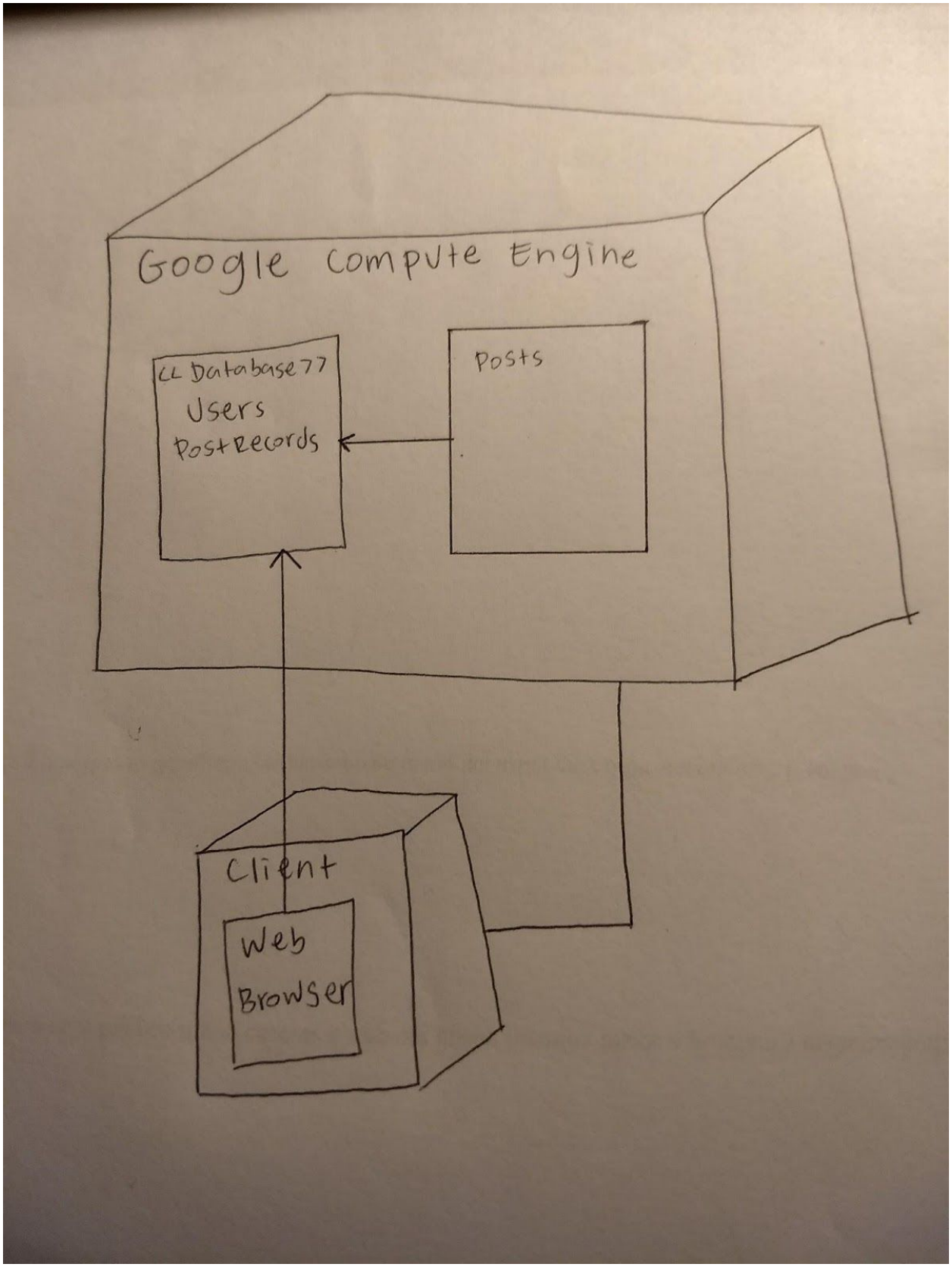
Content for Vertical Prototype

High Level UML Diagrams

UML Class Diagram


styled-components/.../styled-components.d.ts
react/.../React.js
Component
react/react.js
Team/TeamProfiles.js
Component
React
style
About/About.js
About
Component
React
style
mdl-reactjs/.../index.js
Header/Header.js
Component
React
styled
Home/Home.js
Component
React
Map/Map.js
Component
React
Footer/Footer.js
React
SearchResultsPage/SearchResultsPage.js
Component
React
SearchResultsPage
react-router-dom/index.js
BrowserRouter
Link
Route
Switch
VerticalPrototype/SearchPage.js
Component
Link
MDL
React
styled
App/App.js
Component
Footer
Header
React
SearchResultsPage
App/NotFound.js
Link
React
react-dom/.../ReactDOM.js
render
styles/styles.scss
app/index.js
About
App
Header
Home
Link
NotFound
React
Route
Router
SearchResultsPage
Switch
render
styles.scss
icon?family=Material+Icons
material.indigo-pink.min.css
bootstrap.min.css
bootstrap.min.js
material.min.js
jquery-3.2.1.slim.min.js
img/spillLogo.png
popper.min.js
css?family=Open+Sans:300,400,600
public/index.html
Powered by yFiles

UML Deployment Diagram


Google Compute Engine
Database
Users
PostRecords
Posts
Client
Web
Browser

Key Risks

Type	Risk	Solution
skills	Each member has a different set of skills. Most of the team is new to javascript.	We have been going through javascript tutorials.
schedule	Everyone has different class and work schedules.	We discussed everyone's schedules and found a day and time where everyone on the team could meet outside of class.
technical		
teamwork		
legal/content		