SW Engineering CSC 648/848
Section 02

www.ASOPF.com

A Song of Plague & Fire -
WWW Site for Public Safety (COVID & Wildfier) for California
Team 05

Ufkun Erdin (Team lead) uerdin@mail.sfsu.edu
Alexander De Charry (Back-end lead, Github master)
Siddhi Rote (Front-end lead)
Nicholas Handy (Back-end lead)
Huan Nguyen
Erik Loza
Mohammad Razavi

Milestone 2

9/26/2020

Functional Requirements

Priority 1

Both users & admins, when registering, will have the same registration form with the following fields:

- Required
    - Name
    - Phone number
    - Email address
    - Password
- Optional
    - Zip code

Website should implement some sort of back-end analytics software (ie: Google web stats) to allow us to monitor user activity on the site to better update and change the site to serve our users

Users

Able to see the following stats:

- COVID Cases per 100k in CA
- COVID Deaths per 100k in CA
- COVID Cases & Deaths nationwide
- Number of fires in county with respect to evacuation level

Able to use the following functionalities:

- Search for county-specific data (Wildfire & COVID)
- Register for alerts
    - Should be able to toggle which specific stats they get alerts for
- Be able to log in/register on website as users

Admin

Able to see & edit the following stats:

- COVID Cases per 100k
- COVID Deaths per 100k
- COVID Cases & Deaths nationwide
- Number of fires in county with respect to evacuation level

Able to use the following functionalities:

- Search for county-specific data (Wildfire & COVID)
- Access “admin page” to modify data

- Have a “data submission” page where they can submit various fields of data to the site for updates
- Register for alerts
    - Should be able to toggle which specific stats they get alerts for
- Be able to log in/register on website as admin
    - Admin request automatically sent when registering as CA Health/Safety officer
    - Notified by email when admin account is set up

Priority 2

Whenever a new user accesses the site, we should use their IP address to try and determine their county. If the IP is not in California, then display state-wide stats

The website should clearly articulate, within a footnote or a separate page, the requirements for reporting data to our site. This page should change slightly depending on if it is an admin user accessing or regular user.
If an admin user is accessing this page, then we should provide detailed instructions for how they can navigate the website to report.
If a regular user, then the page should serve more as a citation to relieve concerns of accuracy from the user.

Website should be able to send & receive content to and from users (ie: picture files, graphs, video files, etc)

Users

Able to see the following stats:

- COVID Homeless impact in CA

Admin

Input & view data on school reopenings

Priority 3

For both users & admins, we’d like the idea of having a mobile app synced with our web app that can be downloaded by the users.

Users

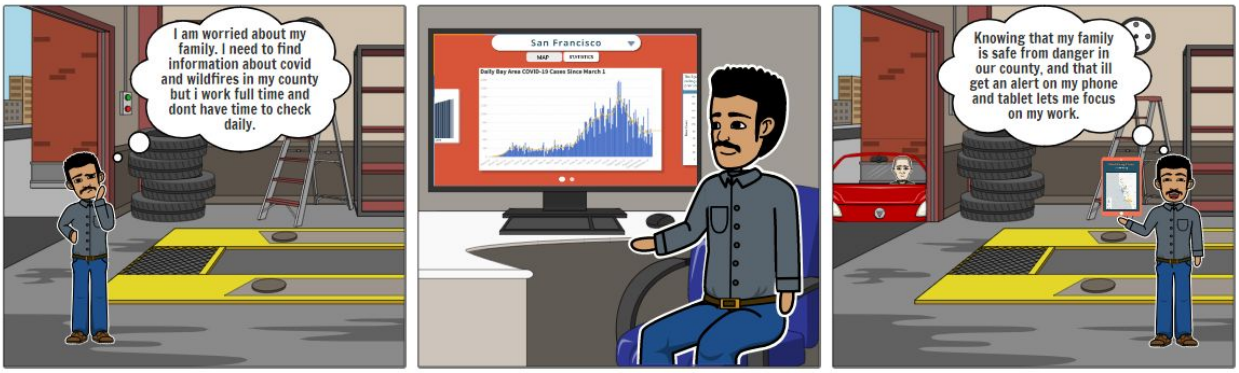
View stats on false-positives and false-negatives (for efficacy of tests)

Admin

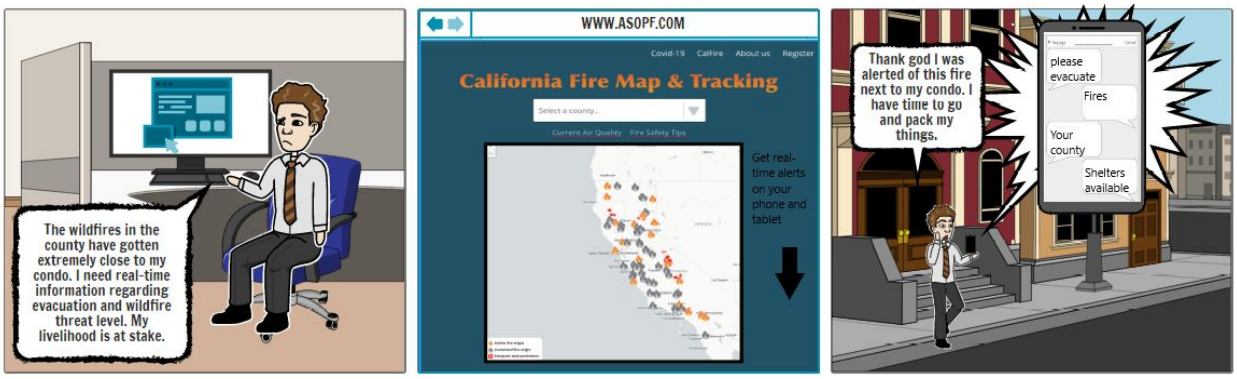
View & edit stats on false-positives and false-negatives (for efficacy of tests)

UI Mockups & Storyboards

Use case #1

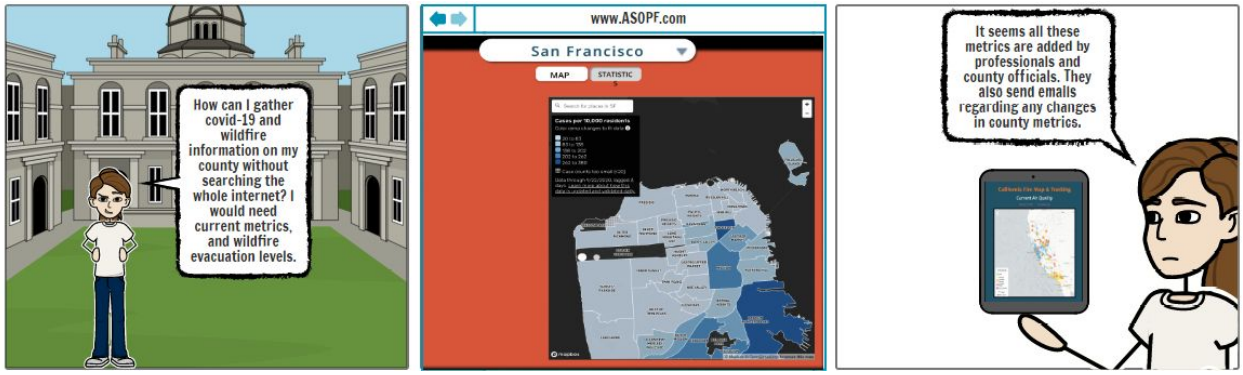


Use case #2



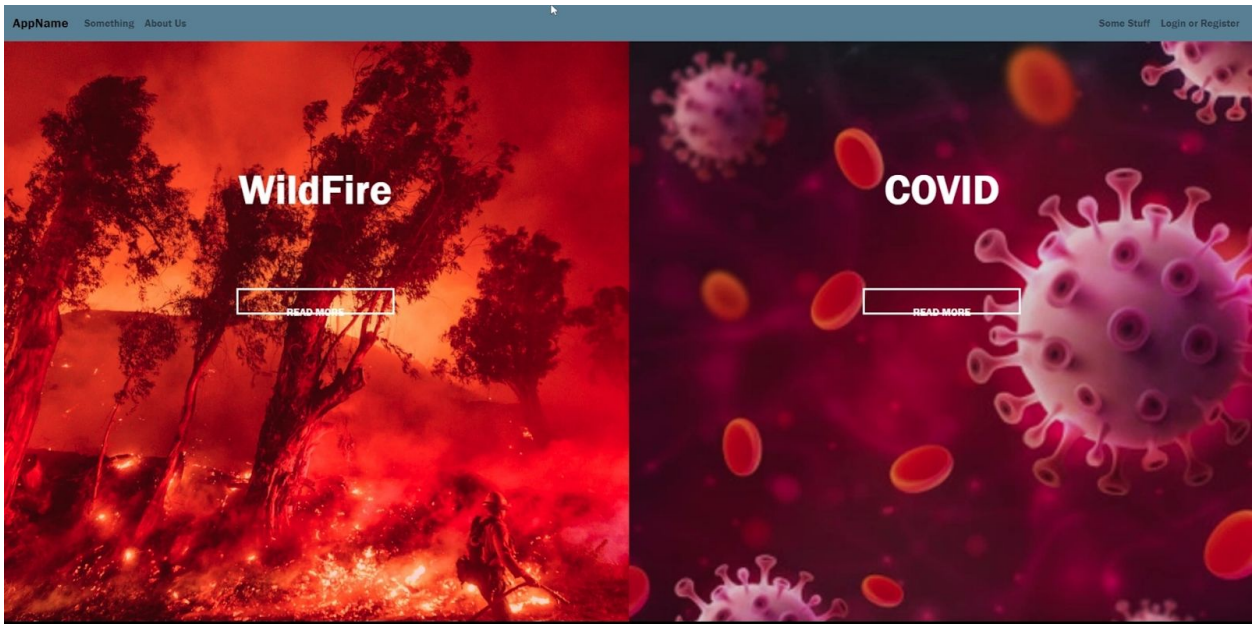
Use case #3

Use case #4

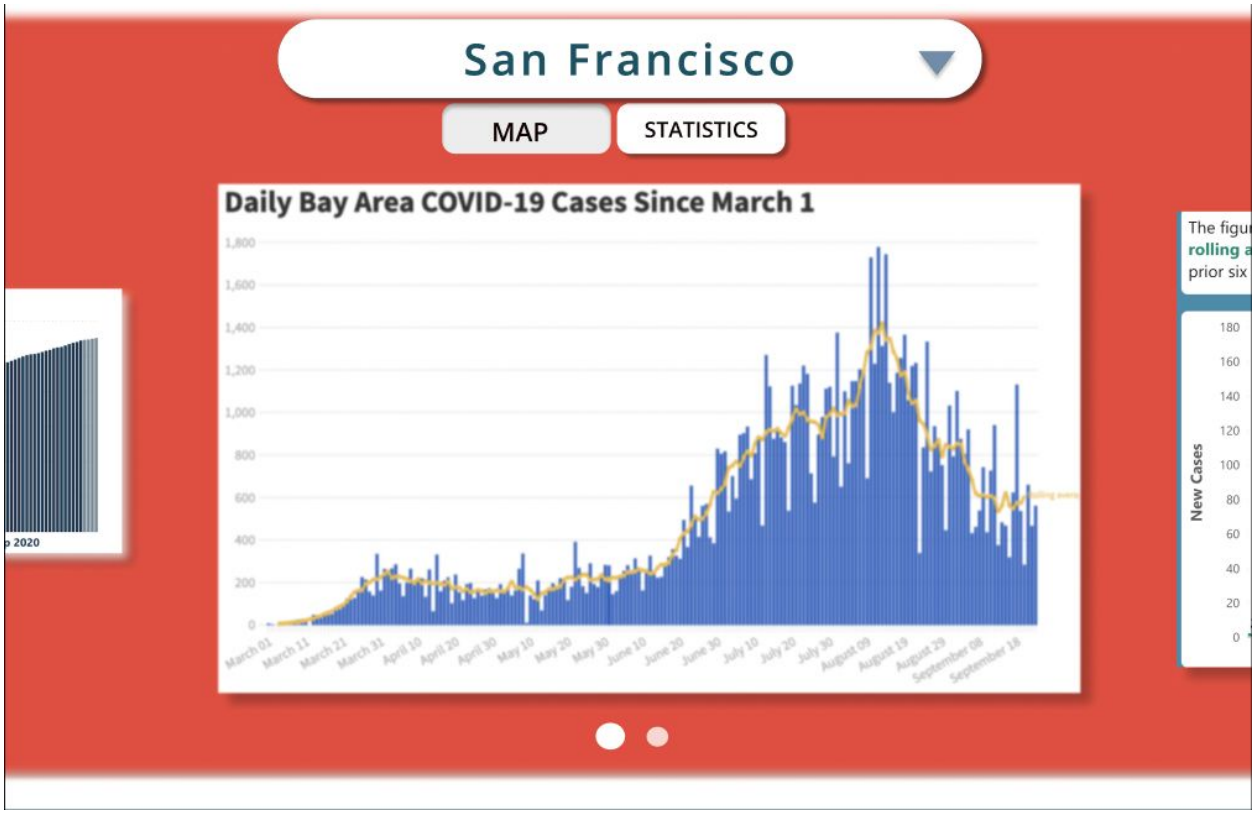


Use case #5

This mockup is of our landing page. We want to have a sidebar with an RSS feed from other sources (county twitter/facebook pages that send out updates that users can follow). The images are just placeholders for now, we are still searching for the right images.



This is another version of our front page. Similar concept to the last mockup, but this one will have a link to access wildfire or covid pages specifically. The RSS feed may be within the specific pages on this one, but (not included in the screenshot) the front pages will have a quick look of stats for each covid & wildfire.



One of the main draws for our website over our competitors is that we want to offer a wide array of views for the data. In addition to tables showing raw numbers, we want plenty of charts, graphs, and maps to visually illustrate the stats, so that from just a quick look a user can have a good idea of the status.

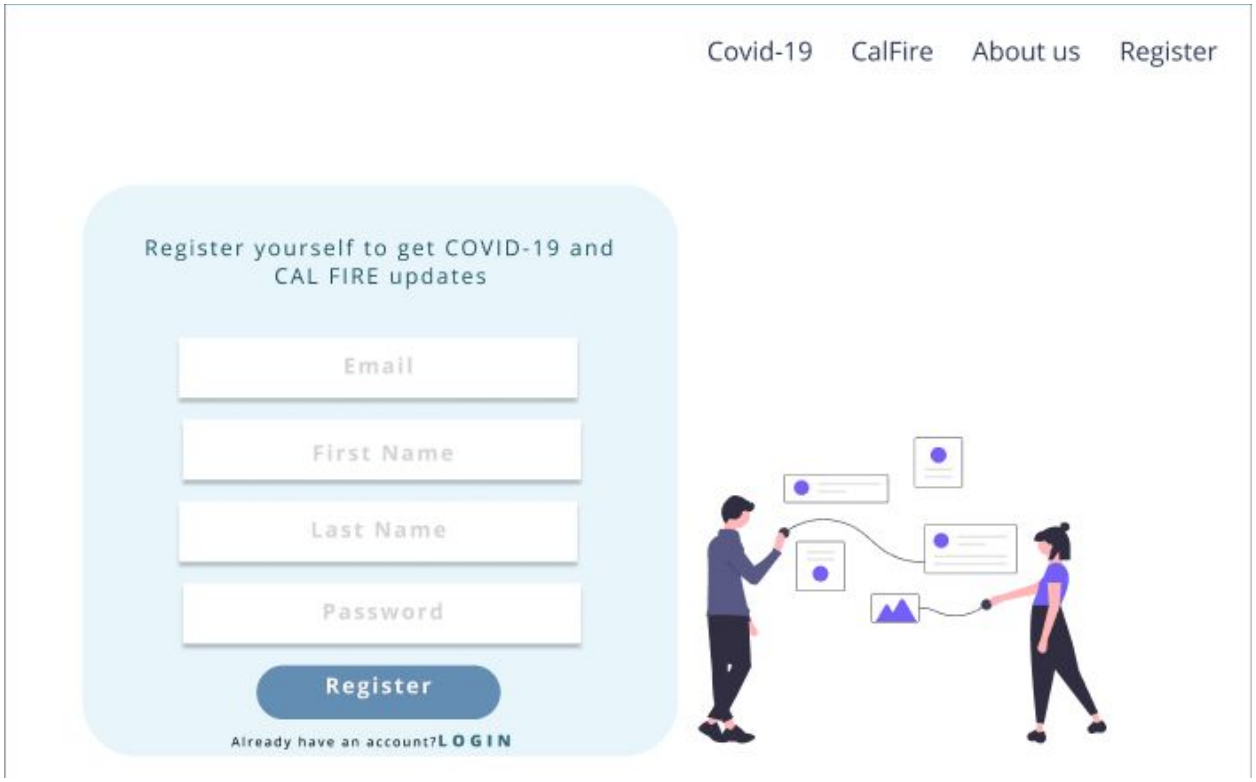

San Francisco
MAP
STATISTICS
Search for places in SF
Cases per 10,000 residents
Color ramp changes to fit data
20 to 83
83 to 138
138 to 202
202 to 262
262 to 380
Case counts too small (<20)
Data through 9/22/2020, lagged 3 days. Learn more about how this data is updated and validated daily.
TREASURE ISLAND
NORTH BEACH
MARINA
RUSSIAN HILL
CHINATOWN
PRESIDIO
PACIFIC HEIGHTS
NOB HILL
PRESIDIO HEIGHTS
JAPANTOWN
TENDERLOIN
GOLDEN GATE PARK
OUTER RICHMOND
INNER RICHMOND
LONE MOUNTAIN/ USF
SOUTH OF MARKET
HAYES VALLEY
MISSION BAY
GOLDEN GATE PARK
HAIGHT ASHBURY
CASTRO/UPPER MARKET
MISSION
POTRERO HILL
INNER SUNSET
TWIN PEAKS
SUNSET/ PARKSIDE
NOE VALLEY
GLEN PARK
BERNAL HEIGHTS
WEST OF TWIN PEAKS
HUNTERS POINT
LAKE SHORE
OCEAN VIEW/ MERCED/ INGLESIDE
OUTER MISSION
EXCELSIOR
VISITACION
mapbox
© Mapbox © OpenStreetMap Improve this map

Covid-19 CalFire About us Register

California Fire Map & Tracking

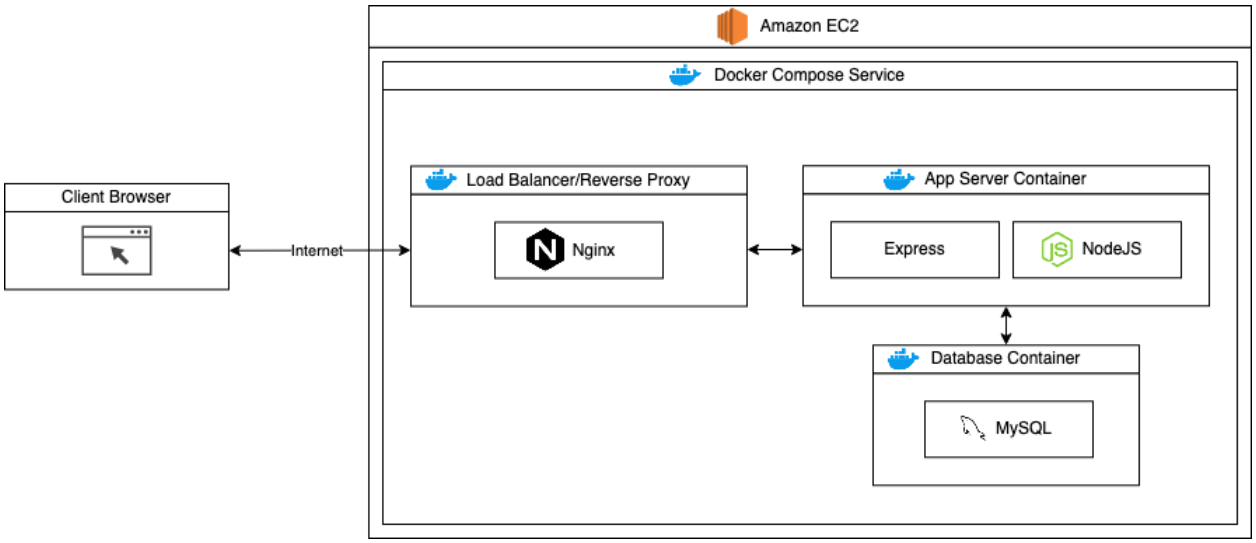
Select a county..

Current Air Quality Fire Safety Tips



A mockup of our login page. Once a user's account is registered, they will be able to opt in to notifications through their account page. We also want to allow the users to customize their alerts/notifications. Users can select *specific data points* that they wish to receive updates on (ie: covid new cases, covid deaths, wildfire new cases, etc). This way, their regular report will be customized to exactly what they want it to be.

High-Level Architecture & Organization



Tools and Systems

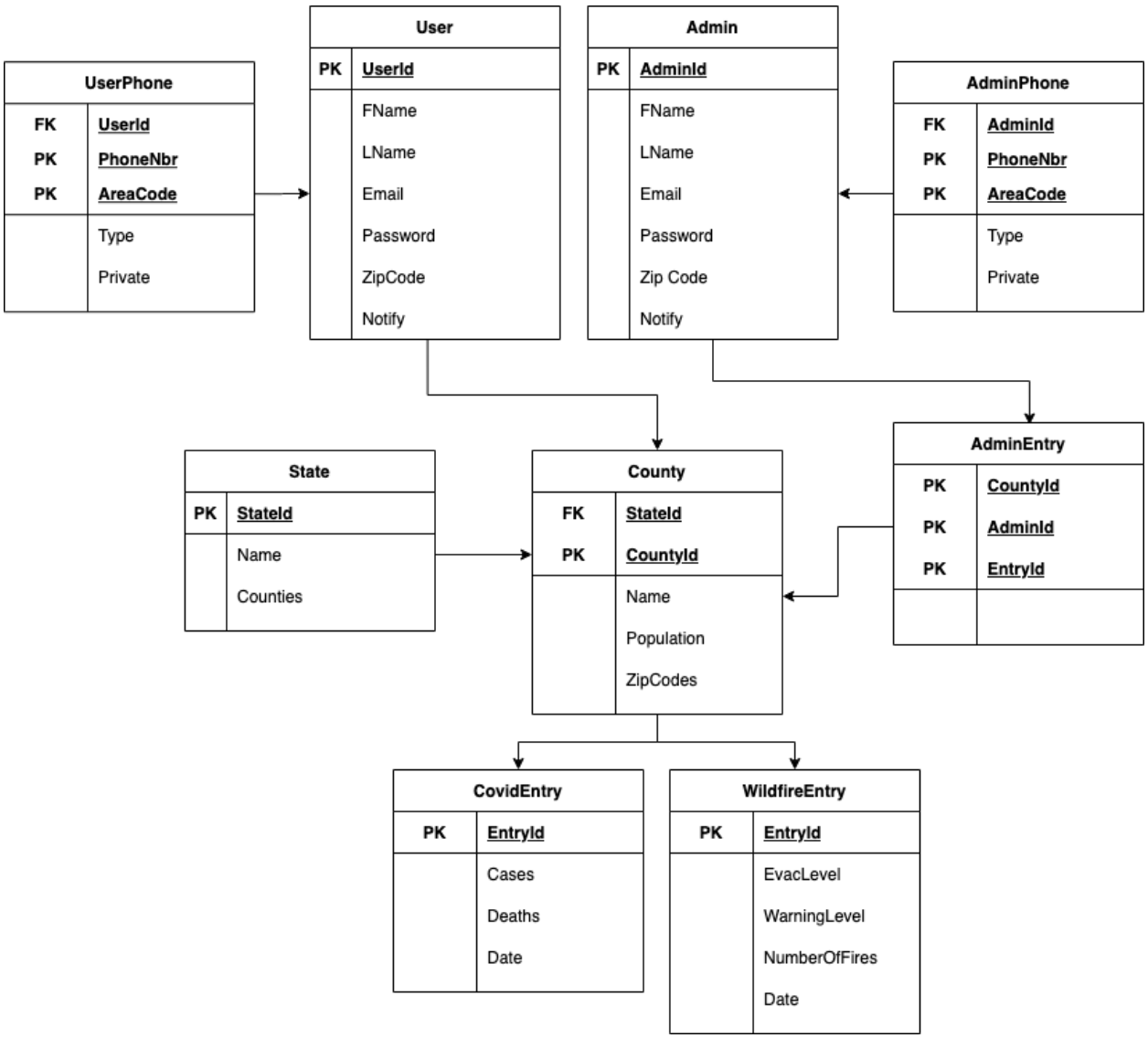
The system uses a customized stack that can be compared closest to a MEAN stack in structure. The main difference is the use of EJS as a templating engine instead of AngularJS and MySQL in place of MongoDB. The stack includes MySQL, ExpressJS, EJS, NodeJS. An additional component to this stack is the Nginx load balancer that sits in front of the NodeJS application servers.

- MySQL Database
  - A multi-threaded, multi-user, sequential Database Management System (DBMS).
- ExpressJS
  - Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.
- EJS (Embedded Javascript)
  - EJS is a simple templating language that lets you generate HTML markup with plain JavaScript.
- NodeJS
  - An asynchronous event-driven JavaScript runtime, built on Chrome's V8 javascript engine.
- Nginx
  - An open source software for web serving, reverse proxying, caching, load balancing, media streaming, and etc.
- Docker
  - Tool used to containerize environments to keep consistency across environments.

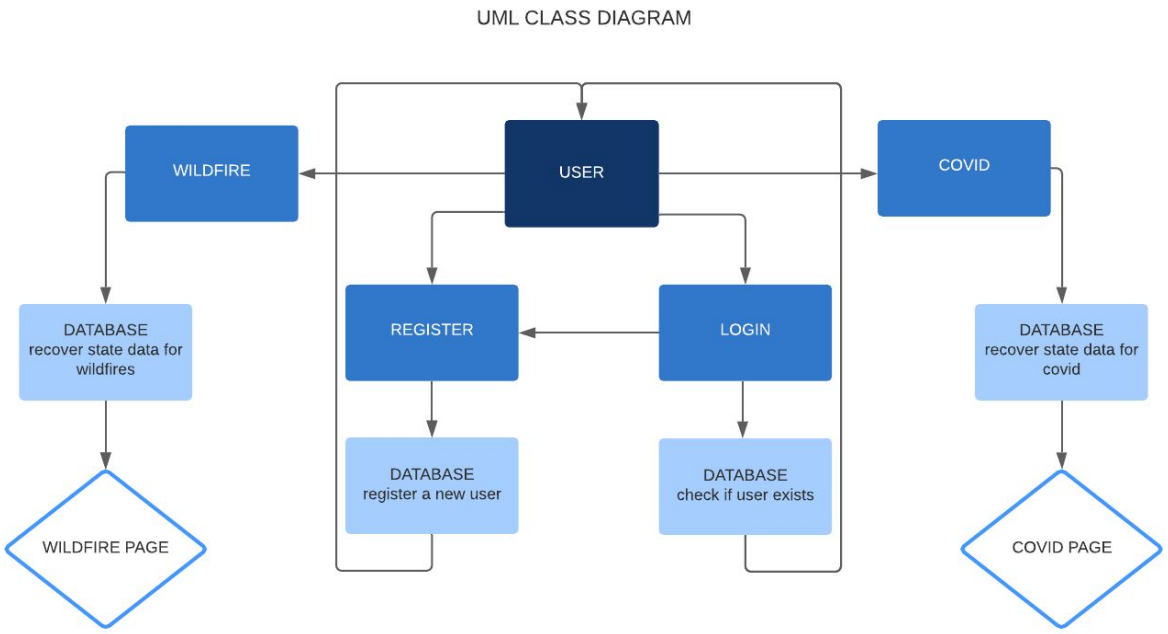
Software Components

- Server Hardware
    - AWS EC2
    - OS: Ubuntu LTS 18.04
    - Type: t.2 Micro 8gb
- Docker
    - Docker-Compose
    - Containers
        - Nginx: 1.19.2
        - MySQL: 8.0.21
        - NodeJS App
    - Containers are network connected internally
- Database
    - MySQL: 8.0.21

Database Schema (ERD)



High-Level UML Diagrams

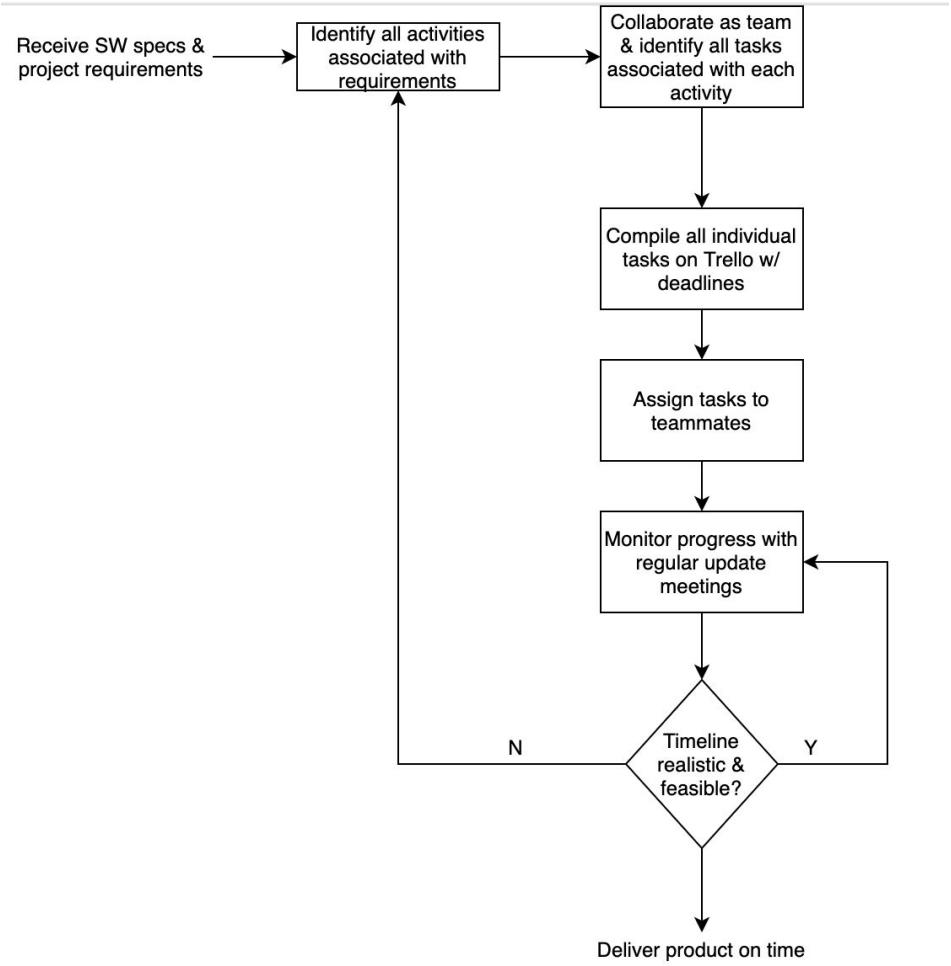


Actual Key Risks

RISK TYPE	POSSIBLE RISKS	RESOLUTIONS
Skills	1. Teammate has difficulty with using software stack	1. Document the step-by-step as well as the big picture of how each element works - provide links to online tutorials
Schedule	1. Global team in various time zones	1. Leverage organizational tools such as Trello to ensure regular progress & synchronization of tasks
Technical	1. Teammate has difficulty with configuring local DB	1. Have one or more meetings dedicated to training & monitoring learning progress of everyone on these tools
Teamwork	1. Teammate overreaching for tasks 2. Teammate slacking on tasks	1. Review tasks & delegate to teammates to ensure smooth flow & not burning anyone out 2. Make sure tasks & duties are clearly articulated to each team member & deadlines stressed
Legal/Content	1. Using non-free-use assets 2. Using implementations of others	1. Ensure all images are pulled from free-use sites and citing sources for all assets used 2. Any time code is pulled from another source, cite it in our code

Project Management

Project Scheduling Process



As described in the flowchart, we begin with a macro view of what needs to get done, and work as a team to identify each micro-task associated with achieving the macro goal. Through subdividing larger tasks to smaller ones, we manage to create a Trello board of each subtask and assign each task, with a deadline, to team members and communicate regularly on progress and interruptions/difficulties.

Management of Two Teams

We've divided our team up into two sub-teams (front & back end), which are as follows:

- Front-End Team
    - Siddhi Rote (Lead)
    - Erik Loza
    - Mohammad Pavi
    - Huan Nguyen
- Back-End Team
    - Nicholas Handy (Lead)
    - Alexander De Charry
    - Ufkun Erdin (Project Lead)

Tasks on the Trello board(s) are tagged as either front end or back end, so that team members can easily identify which tasks on the board they should work on. Additionally, tasks are prioritized from Priority 1 - Priority 3. This will give team members a better idea of what tasks to work on first.

The team has two meetings every week: one after class on Tuesday and one on Saturday afternoon. These meetings serve as a status update for each teams to share their progress and spot check. If any member of the team needs help or is unclear, this is where most of the work in that field is done. Additionally, we have a group message on Discord where every member is regularly active & participating in discussions.