

Team Buff Buddies

Recitation 112 - Team #2

Members: Owen Carlson, Aileen Ma, Briana Griffin, Alex Mazur

Jupiter - Milestone 4

Updated Feature List 3/29/2020 (Feature Priority is Descending) -

- **Weather Comparison Lists (Unchanged - In Progress):**

A user will be allowed to compare different locations at the same time, side-by-side. Each location is represented by a weather card. If the amount of locations entered exceeds the size of the screen then they will overflow into a grid pattern. If a user is logged in then they will be allowed to save any list that they have made.

- **Weather Cards (Revised - In Progress):**

Each weather card contains the current weather at a specific location. A weather card is a member of a list of cards that will be displayed on the screen at the same time. Weather cards also contain estimated travel time from user specified location.

- **Trip Time Predictor (Revised - In Progress):**

Uses Google Maps API to predict travel time to a location. This data is displayed on an individual weather card.

- **Node Server (Unchanged - In Progress):**

This is the backbone to the entire website. The backend server will make all API calls, handle user login, and connect to the database.

- **Dynamic Route Viewer (Unchanged - To Do):**

Beneath the list of weather card(s) there will be a map pulled from Google Maps API that displays a route to the location. The user will be able to change which route is loaded and displayed on the map. If possible, we might be able to overlay multiple routes on the map at the same time.

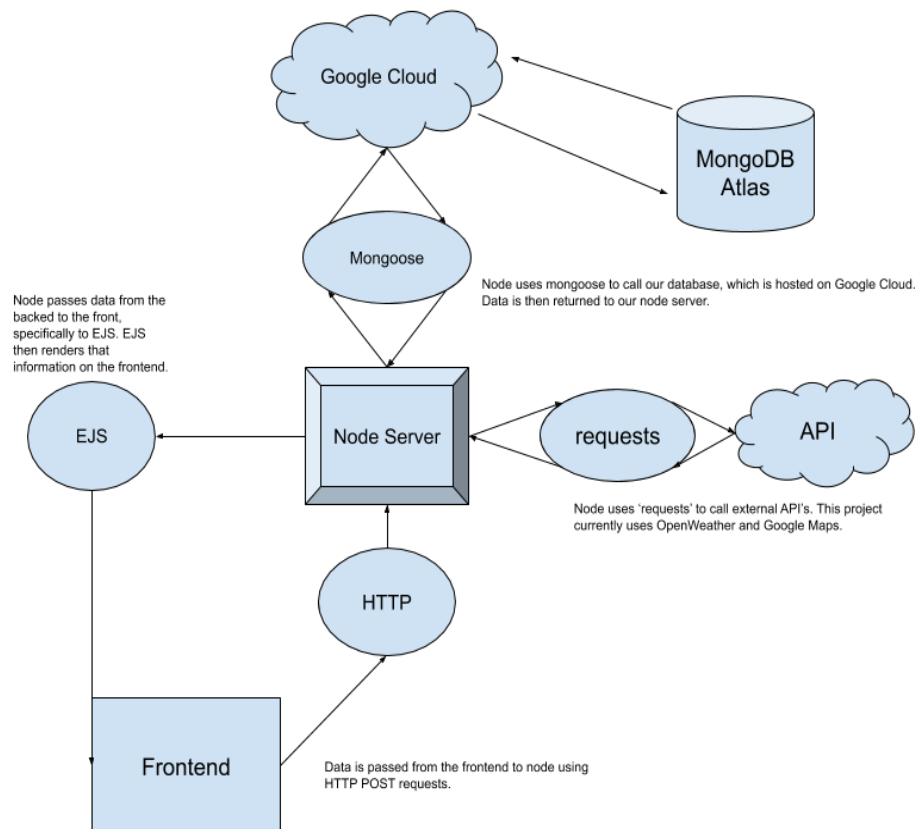
- **User Accounts (Unchanged - Finished):**

Each user will have the ability to login to the site with an email and password. Logging in will allow a user to save location data, location comparison lists, etc.

- **Database (Revised - Finished):**

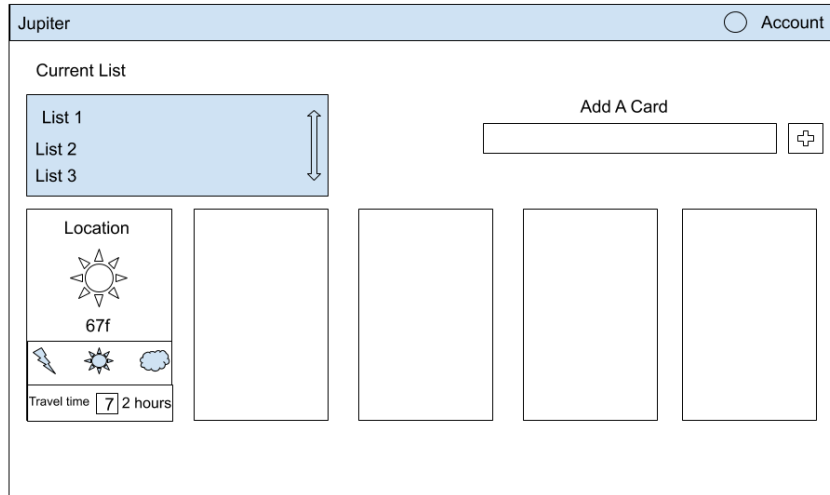
Uses MongoDB to store user account information.

Architecture Diagram:

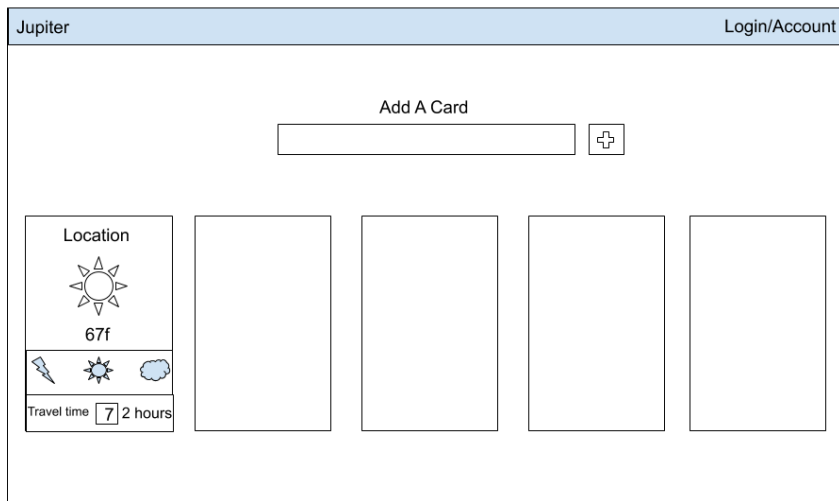


Wireframes:

This wireframe shows a logged in user-

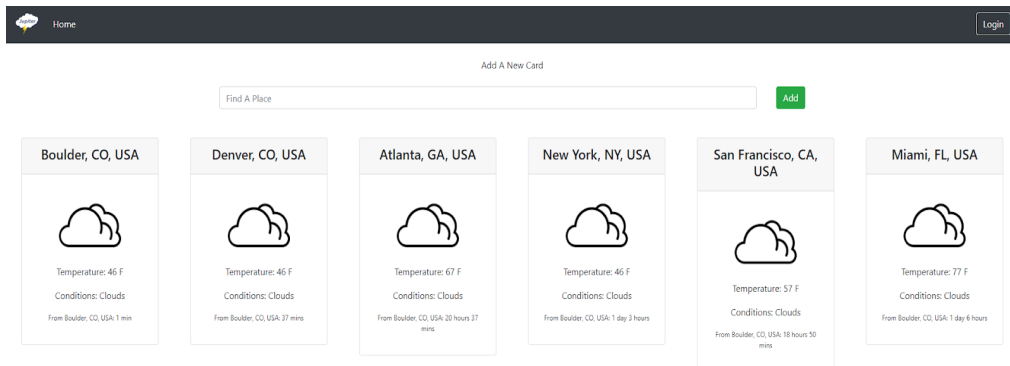


This wireframe shows the standard logged out front page-

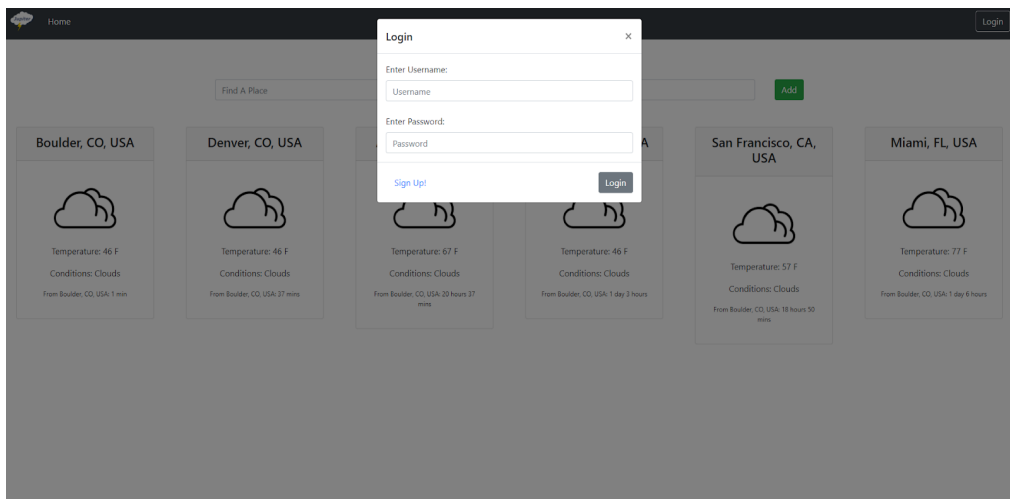


Current Look:


Standard Logged Out Home Page after adding some locations-



The Login Menu, leads to a Signup Page as well-



A Logged In User Page (admin)-


 Home

adminLogout

Add A New Card

Add

Boulder, CO, USA




Temperature: 46 F

Conditions: Clouds

From Boulder, CO, USA: 1 min

Denver, CO, USA




Temperature: 46 F

Conditions: Clouds

From Boulder, CO, USA: 37 mins

Atlanta, GA, USA




Temperature: 67 F

Conditions: Clouds

From Boulder, CO, USA: 20 hours 37 mins

New York, NY, USA




Temperature: 46 F

Conditions: Clouds

From Boulder, CO, USA: 1 day 3 hours

San Francisco, CA, USA




Temperature: 57 F

Conditions: Clouds

From Boulder, CO, USA: 18 hours 50 mins

Miami, FL, USA




Temperature: 77 F

Conditions: Clouds

From Boulder, CO, USA: 1 day 6 hours

The User Registration Page-

 Home

Login

Username

Password

Repeat Password

Submit

Web Services:

Jupiter currently uses two API's, called using the 'request' node module.

1. OpenWeather

- The data being sent is a user entered location
- The data being received is a large JSON object that contains current weather at that location

2. Google Maps

- The data being sent is two user selected locations, an origin and a destination.
- The data being received is a JSON object that contains an estimated time of travel.

Database Design:

Jupiter uses MongoDB atlas, hosted on the Google Cloud Platform. We access the database using the 'mongoose' node module. We are only storing user account information and a user list, the schema for which is as follows:

```
let userSchema = new mongoose.Schema({
  username: {
    type: String,
    required: true
  },
  password: {
    type: String,
    required: true
  },
  lists: Array
});
```

This can be thought of as a JSON object, with fields 'username', 'password', and 'lists'.