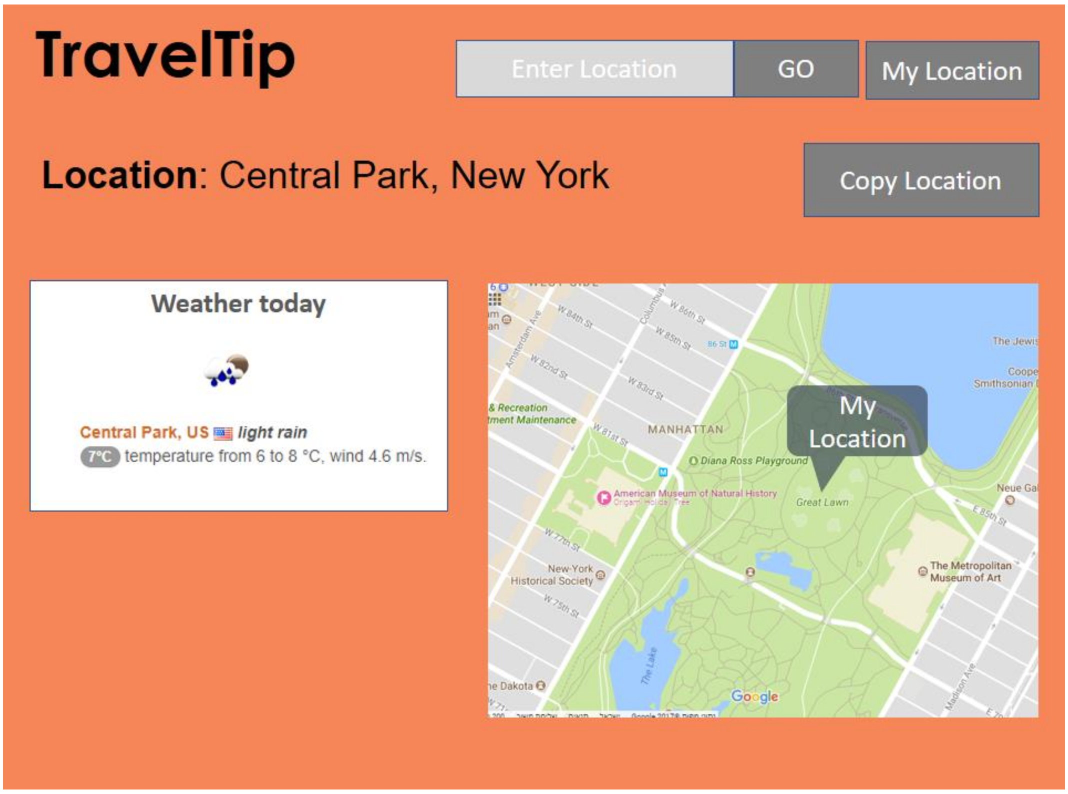
TravelTip

GeoLocation, Ajax, ES6 Practice, Git Collaboration

TravelTip is an app that helps you get information about a specific location

Collaborate with another dev on this project, get familiar with git collaboration commands and split the work on this project correctly between the two of you.

UX Concept:



Location: Central Park, New York

Weather today

Central Park, US light rain
7°C temperature from 6 to 8 °C, wind 4.6 m/s.

Requirements

1. UI shall have the following fields:
   - Map with your location marked on it
   - My Location button
   - Descriptive name of the location (Ramat Gan, Israel)
   - Weather at the location

   - Copy Location – copies (into the clipboard) a link to our app with the lat/lng as query string parameters so you can share your location with others

TIP: This app should be deployed to github-pages, so the URL should be something like: *github.io/travelTip/index.html?lat=3.14&lng=1.63*

a. When app loads it looks into the query string params and if there are lat/lng params (see here), it will display accordingly.

- Search input that uses geo-code API to get lat/lng from text

2. UI shall look nice and responsive
3. Location defined by either:
   - Location inserted
   - User location (using geolocation from device)
4. API used: open weather map, google maps, google geo code
5. Bonus: create your own designed marker on the map

Maps API guidelines

1. Maps API (add your map):
   https://developers.google.com/maps/documentation/javascript/adding-a-google-map

2. Address <> Lat/Long - google geocode api:
   https://developers.google.com/maps/documentation/geocoding/start
   Go to Google Developer Console, and enable the Geocoding API before using it

Weather API guidelines

http://openweathermap.org/

REGISTER, GET API KEY, and get started here

Sample url (API key is needed)

NOTE – take notice of the temperature units

```
`http://api.openweathermap.org/data/2.5/weather?lat=${lat}&lon=${lon}&APPID=${W_KEY}`
```

Development Workflow

1. Start together, build a simple project structure
2. Setup git and make sure the team can push and pull from the repository.
3. Design an initial page (index.html, main.js, service.js, CSS files) together (don’t spend more than 2 hours at this stage)
4. Commit, Push and Pull
5. Split and work separately and together whenever you see fit.
    a. Try to push and pull every hour to coordinate your code and practice the workflow
6. Use this opportunity to improve the code by mutual code reviews