PROJECT SPECIFICATION

FEND Capstone - Travel App

Development Environment and Architecture

CRITERIA	MEETS SPECIFICATIONS
Architecture	The project should have a structure like the one shown below. All files shown must be present (Webpack may be split into multiple config files, and names may differ) and the app must successfully render a home page with clear design and functionality added when index.html is loaded in the browser. The project should not contain errors in the browser console.

```
- Root:
  - `package.json`
  - `readme.md`
  - `webpack.config.js`
  - src folder
    - server folder
      - `server.js` (name will vary)
    - client folder
      - `index.js`
      - html/views folder
        - `index.html`
      - js folder
        - `app.js` (name will vary)
      - styles folder
        - `style.scss` (name will vary - may be broke int
o partials)
```

Webpack	Webpack config should contain at least 3 scripts, express server, build and test. Additionally, dev server may be included.

CRITERIA	MEETS SPECIFICATIONS
Testing	• Check if Jest has been installed and `npm run test` script is implemented to run Jest. • There should be at least one test for the express server. • There should be at least one test for the application javascript client.
Offline capabilities	The project must have service workers installed.

HTML & CSS

CRITERIA	MEETS SPECIFICATIONS
Usability	• All features are usable across modern desktop and phone browsers. • Ensure the HTML elements, eg. texts and buttons, are proportionate and readable in small screen devices.
Styling	Styling is set up in a logical way. All interactive elements have hover states.
HTML Structure	HTML structure should be indented properly with classes and ID's that make sense.
Visual Design	The design should clearly be different from the design used in projects 3 and 4.

API and JS Integration

CRITERIA	MEETS SPECIFICATIONS
Server `src > server > server.js`	`server.js` file should be taken directly from passed project 3 with the addition of added member: value pairs and the required API keys.
index.js `src > client > index.js`	• At least one function should be imported. • At least one event listener should be imported. • (styles referenced in html/css)
app.js `src > client > js > app.js`	1. There should be URLS and API Keys for at least 3 APIs, including Geonames, Weatherbit, and Pixabay. You can feel free to use more than 3 APIs. 2. There should be a primary object with placeholder member value pairs. 3. There should be a primary function that is exported to `index.js` (index.js file should import the functions from other files).

Documentation

CRITERIA	MEETS SPECIFICATIONS
README	A `README` file is included detailing the app and all dependencies. **Other requirements:** The Readme file should have non-default text in it that is specific to this project. It doesn't have to be thorough, but should have some basic info. Bonus points if correct

CRITERIA	markdown is used. MEETS SPECIFICATIONS
Comments	Comments are present and effectively explain longer code procedure when necessary.
Code Quality	Code is formatted with consistent, logical, and easy-to-read formatting as described in the Udacity JavaScript Style Guide.

Suggestions to Make Your Project Stand Out!

At least one of these is required, but the rest are great additional ways to further customize and improve your project!

- Add end date and display length of trip.
- Pull in an image for the country from Pixabay API when the entered location brings up no results (good for obscure localities).
- Allow user to add multiple destinations on the same trip.
    - Pull in weather for additional locations.
- Allow the user to add hotel and/or flight data.
    - Multiple places to stay? Multiple flights?
- Integrate the REST Countries API to pull in data for the country being visited.
- Allow the user to remove the trip.
- Use Local Storage to save the data so that when they close, then revisit the page, their information is still there.
- Instead of just pulling a single day forecast, pull the forecast for multiple days.
- Incorporate icons into forecast.
- Allow user to Print their trip and/or export to PDF.
- Allow the user to add a todo list and/or packing list for their trip.
- Allow the user to add additional trips (this may take some heavy reworking, but is worth the challenge).
    - Automatically sort additional trips by countdown.
    - Move expired trips to bottom/have their style change so it's clear it's expired.