The App

Scenario: The government wants to help people make the transition to electric vehicles. One of the ways they're doing that is collecting owner satisfaction with their EVs, aggregating the data, and making it available to consumers to help with their choice.

The Rate My EV app does just that. Here's how it works:

- When a new owner registers their EV for the first time, the registration office creates a unique token for that vehicle and emails the owner a link to the app with that token.
- The new owner follows the link, which opens a page for them to leave their review. Once they've done that, it's in the database.
- Other users can open the app and view average ratings for all the models of EVs that have been rated.

Administering the data: Adding `/admin` to the URL takes you to the administrator client. Here's where the registration office adds tokens. They can also determine what categories the new owners will rate their EVs on. Finally, there are options for creating and manipulating test data, which isn't factored in to the ratings that users see.

Installing and running the app. The app's code is hosted on GitHub at QuanticSchool / Rate-My-EV. Instructions for installing and running the app are in `README.md`.

To: ev_owner@mail.com

From: EVpromo@your.gov

Subj: Rate your EV

Thanks for choosing a new EV! Click on this link to let others know what you think of it. If you take the time to help out, you'll be eligible for a tax credit.

Sincerely, Department of EVs

Architecture

Components: Rate My EV is a simple app, but has the three main components of most web apps:

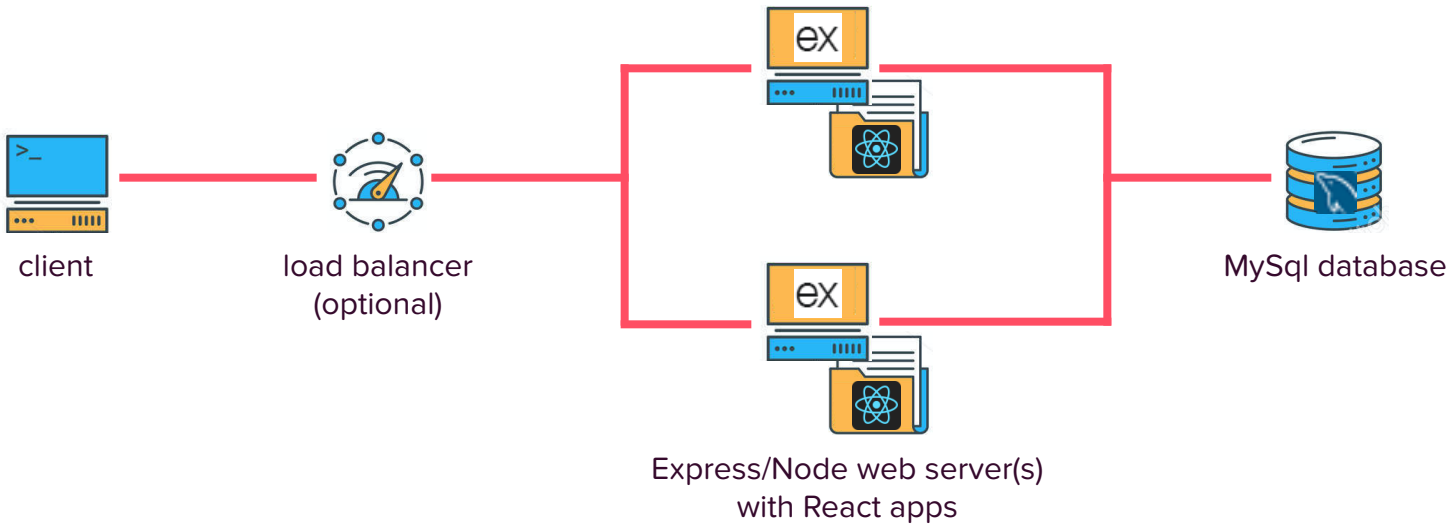
- The app's front end consists of two React apps: one for the administrator client and one for all other users. Both use the Create React App (CRA) template. The adminstrator client is in the `/admin` folder, and the user client is in the `/client` folder.
- The app's web server is implemented using Express and Node. It has two jobs. First, it serves the static files (HTML, CSS, and JavaScript) that comprise the front end. Second, it expose a REST API for the front end (and other consumers as needed) to create, retrieve, update, and delete information in the database. The API is documented in `server.js`. Note that the web server is not stateful, so it's easy to scale by adding multiple instances behind a load balancer.
- The app's database (`ev_ratings`) is hosted using MySQL. There are separate user accounts for database administration (`Quantic_Dev`) and users (`ratemyev_user`).

You can find more information about the various frameworks/ environments on their websites:

React: reactjs.org

Express: expressjs.com

MySQL: mysql.com



The icons are from the same set as the ones I used for CLOUD1 (see 1.01 folder). When we purchase those we should replace the bitmap ones on the page with the vector versions. The logos for Express, Node, and MySQL are taken from the favicon on each of their websites. They're not critical and can be deleted if we're concerned about copyright issues.