Assumptions: For this app, I chose my stack to be React.js v16.3, Express.js v4.3, Node.js v8.6, and PostgreSQL v10.3. PostgreSQL because it is faster for writing while MySQL is better for reading data into a database. My app will use flux pattern to re-render the app when there is an update in state from a user's input. I plan to integrate Redux to handle my app's state, and Sequelize, a JavaScript ORM, to make my codebase more clean, readable, and easier to debug. For the React components, I'll probably reuse some modules as the app scales, so I will separate these modules into different files. I'll create a index.js file in the components folder to make my components export easier to read and to avoid writing out long imports in other classes (i.e. " import Loading from './components/Loading/Loading' " can now be written as " import { Loading } from 'components' "). In addition to the index.js export file, I'll also configure webpack to use my app's folder as the root directory to shorten my imports. I will keep my components in my components folders and my API helpers in my utils folder for better organization.

Decision: Since this was a single page app, with minimum user concurrency actions and only one type of data insertion (text), I omitted using Redux and Sequelize to save development time. I also chose MySQL v5.7 over PostgreSQL because I already have MySQL server installed. Other than these changes, everything else in my assumption was implemented.

Tasks Completed: The app shall fetch HTML contents from a URL. The app will only fetch HTML if the URL is structured like this: "http//www.google.com" if not, an error would be returned. When a query has been submitted by the user, the status "Fetching URL..." would be displayed until the HTML result or error is returned to the client. If HTML is valid, it would be stored in MySQL database.

Tasks Uncompleted: I didn't write out any tests for this app (I probably wouldn't need to since it only has one functional feature). I also didn't get a chance to research how I could implement API keys in my API to authenticate users.