Challenge No.4 - Specification

Restaurant Application

Description
For this challenge you will have to create a restaurant application with React and Zustand.

There is a starter folder, with a **db.json** file with all the data which you must run on a separate terminal using the **npm run server**.

You will have to use everything that you learned to make things work like in the video.

Implement a component-based design with the following components:

- **Navbar** - Contains the header that is visible on every page.
  - The logo should **Link** to the Homepage
  - The heart icon should **Link** to the Favorites page

- **Footer** - Contains the footer including a paragraph and is visible on every page

- **RestaurantCard** - An individual restaurant card item that should be a **Link** to the restaurant details page.

- **AllRestaurants -** A container that lists all the **RestaurantCard** components based on the data from **db.json.**

- **RestaurantDetail** - The detail page for each restaurant, containing data based on the data from **db.json**.
  - This page also contains a **reviews** section. The reviews should be fully functional. Some of the restaurants you will notice contain reviews already and some of them don't. The **rating** for each restaurant is based on the review stars and shown in each of the **RestaurantCard** components as well as on the **detail pages**.
  - You can add reviews with the review **form** and each review should be added with an **ajax - PUT** request back in the **db.json** file. You can make a **PUT** request to the following endpoint: http://localhost:5001/restaurants/*restaurant-id*
  - Once you add a **review,** the rating of the restaurant gets updated based on the new star's average score.
- **PopularRestaurants** - A container that lists the **10** restaurants with the most reviews using the **RestaurantCard** component. (*you can also **sort** them here based on the rating - that's a **plus***).

- **SurpriseRestaurant** - Surprise section, which will contain a **Link** to a random restaurant **detail page** each time you click on it.

- **Cuisines** - A container that lists buttons based on all restaurant types from the **db.json**. Clicking on the button should Link to the cuisine **details page**.

- **CuisineDetail** - The detail page for each cuisine that filters all the restaurants based on the restaurant type from the **db.json** and uses the **RestaurantCard** component to show them on screen.

- **Favorites** - A container that lists all the **RestaurantCard** components that are marked as favorite - use a state **management library** to get this to work.

Make sure you use context to make your life easier, because you will need the restaurant's data in every component. Use state management for the favorites since that part should work globally and work even on reloading of the page.

Also make sure you use **Typescript** with **.tsx** file extensions and React component types for each of the components, methods and wherever needed.

Level 1: Beginner - 1 point

creating all the components from above and linking them correctly in the **App.tsx** file so they render fine on the screen. (you can connect every component in there, even though it is not like it on the screenshots).

Level 2: Intermediate - 3 points

Linking everything with the React router while making it look like it is in the video.

Level 3: Advanced - 5 points

All of the above. Plus, make all the functionality based on the video (filtering, add/remove favorites, sorting, reviews with AJAX, use context, state mgmt).

Deadline

2 weeks after its presentation, at 23:59 (end of the day).