Project Documentation and code explanation

Hosted URL : https://todolist-c917e.web.app/

Project Description:

The task chosen was to implement a TodoList website with basic CRUD functionalities. This task was completed through a **Web App** SPA based on **React** framework.

Project Dependencies (npm):

1) Material UI : Used as preference of UI library

2) Firebase : Used to implement back-end.

3) React-router : Used to implement routing functionality and move to different pages.

4) Axios : Helper dependency to make API calls

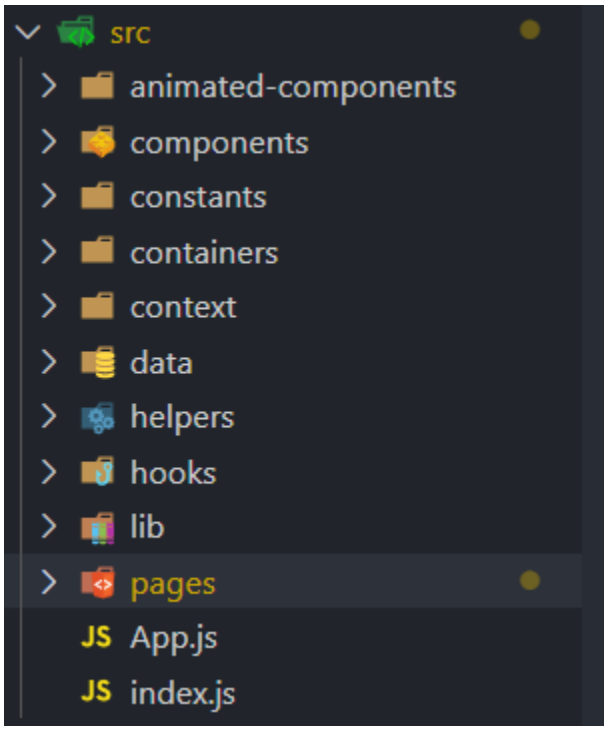
5) React-spring : Helper dependency to give transitions.

General Intuition:

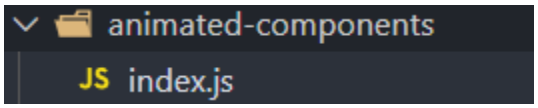
The general intuition is that when the user logins, we fetch all the lists made by that user and display it on the browse Page. Any creation, updation or deletion is

reflected in the Firebase Firestore and the component is reloaded.

Project Folder Structure:

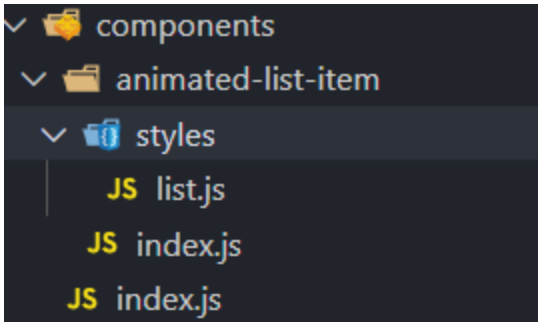


1) Animated components:



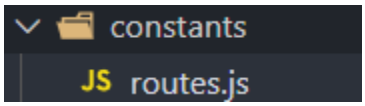
This folder is somewhat of a utility folder which imports the required components from Material Ui library and then exports them by passing them through the animated function provided by react-spring. This is done to ensure that transitions are applied to all components and not just a single component.

2) Components



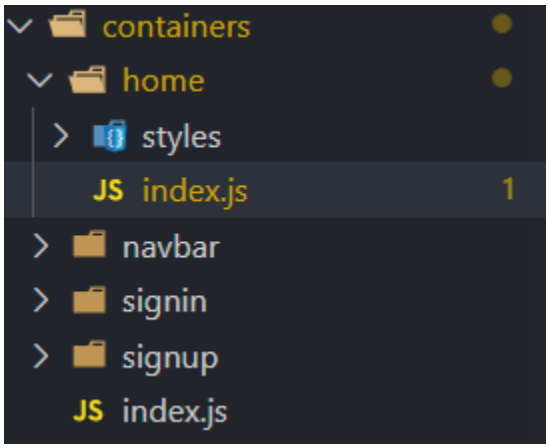
This folder contains the animated-list-item component. This component is the main crux of the project and is entirely responsible for the functioning of the TodoList page.
The styles folder contains the general styling used by the component.

3) Constants



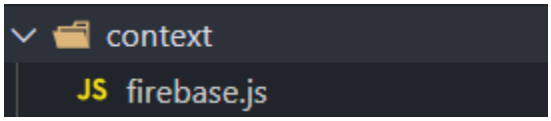
The constants folder contains a file called routes.js which exports the paths of the routes used. This is done so that there is a single source of these routes' path and need to be changed at only one place if desired.

4) Containers



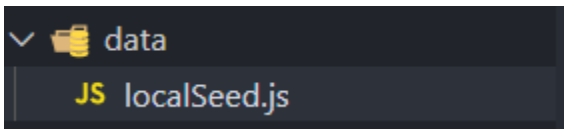
Similar to components, this folder contains very simple components which are intended to be used only once. This was originally made to be a folder where different custom components would be assembled.

5) Context



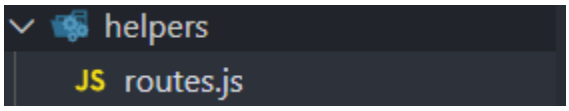
This folder contains a single file which is responsible for the creation of the firebase context. This is done so that any component at any place in the hierarchy can access firebase utilities by using the useContext( ) hook.

6) Data



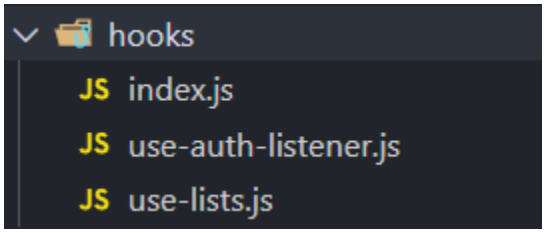
This is a testing folder which contains static seed to test if the above components worked correctly. In further updations, this folder would also contain a seed for the online database (Firebase firestore)

7) Helpers



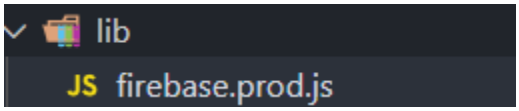
This folder contains a utility file. Essentially this defines if a route is protected or public. If a protected route is being accessed by someone who is not logged in, this ensures that the person is redirected to the signin page. It also ensures that once a person logs in, he/she gets automatically redirected to the browse page when accessing any other route.

8) Hooks



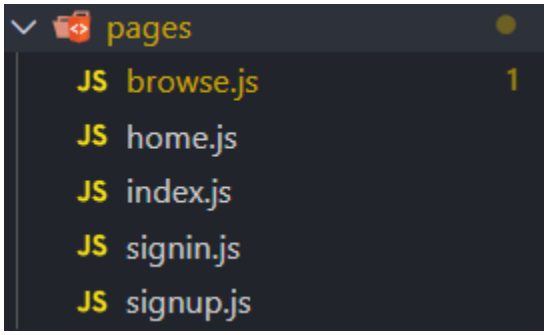
This folder contains 2 custom hooks. The use-auth-listener hook is responsible for listening to changes in the logged in/ logged out state of the user and changing the component based on that.
The use-lists hook is responsible for fetching the data(Todo Lists) associated with the current user from Firebase firestore.

9) Lib



This is used to initialize the firebase values provided at the time of project creation.

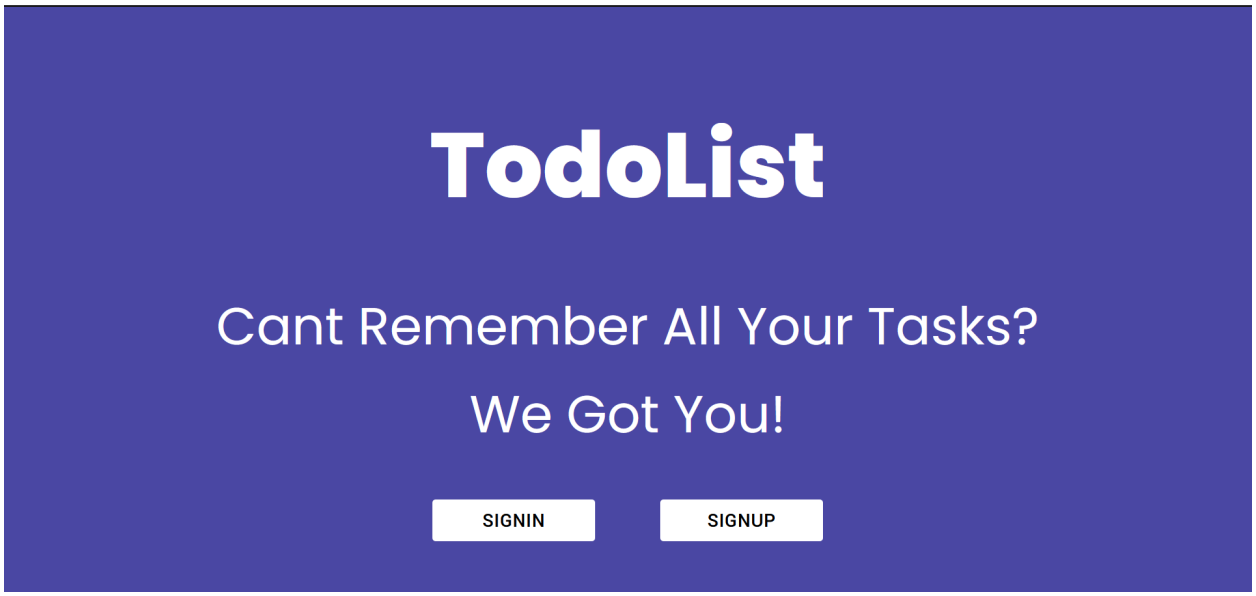
10) Pages



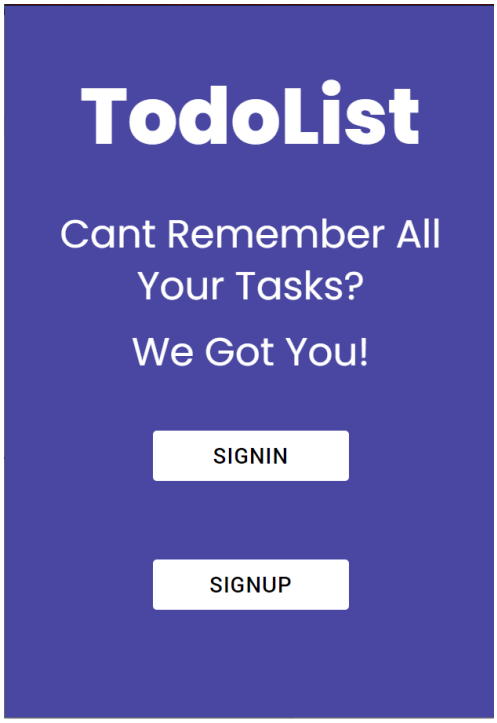
A general folder which assembles the components from the container and other component folders for any particular page.

Results :

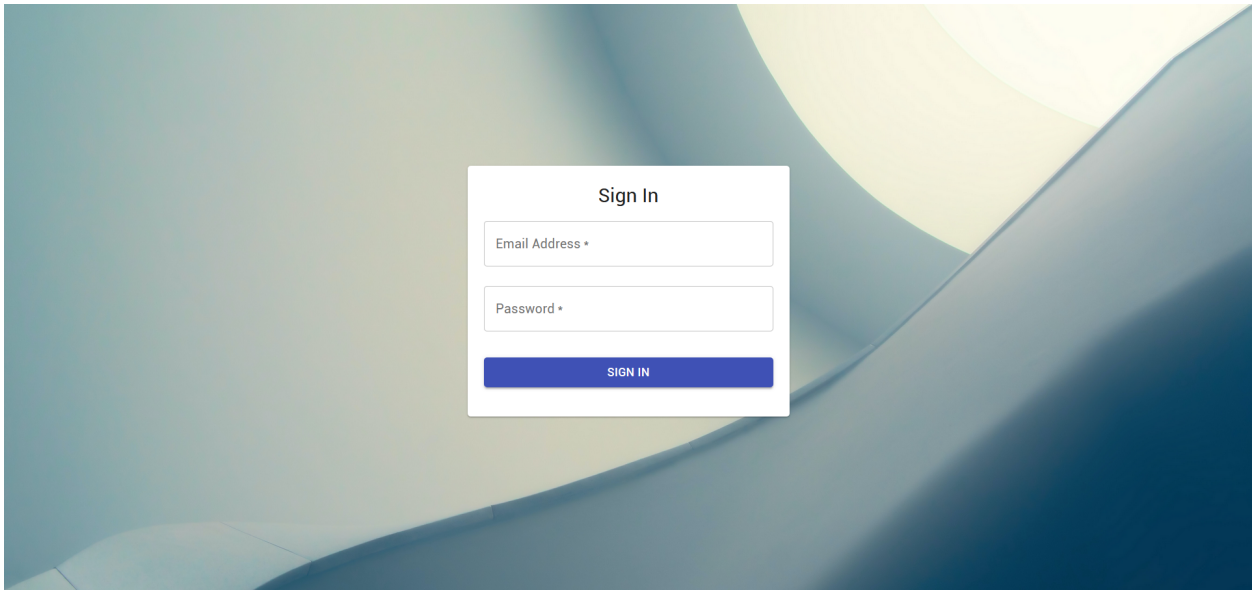
Home page



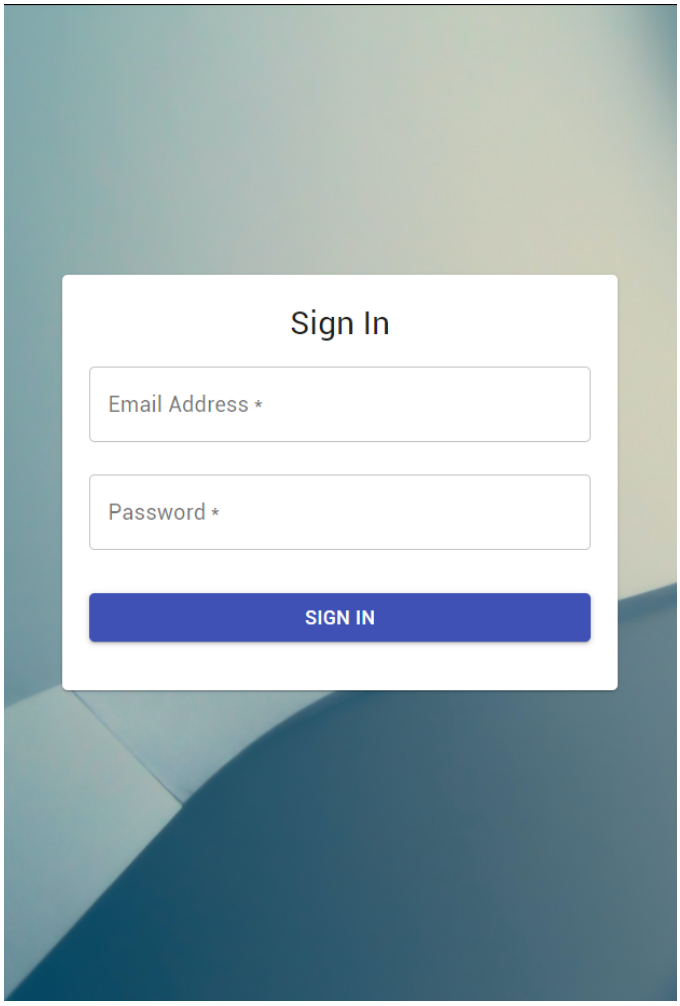
Mobile view:



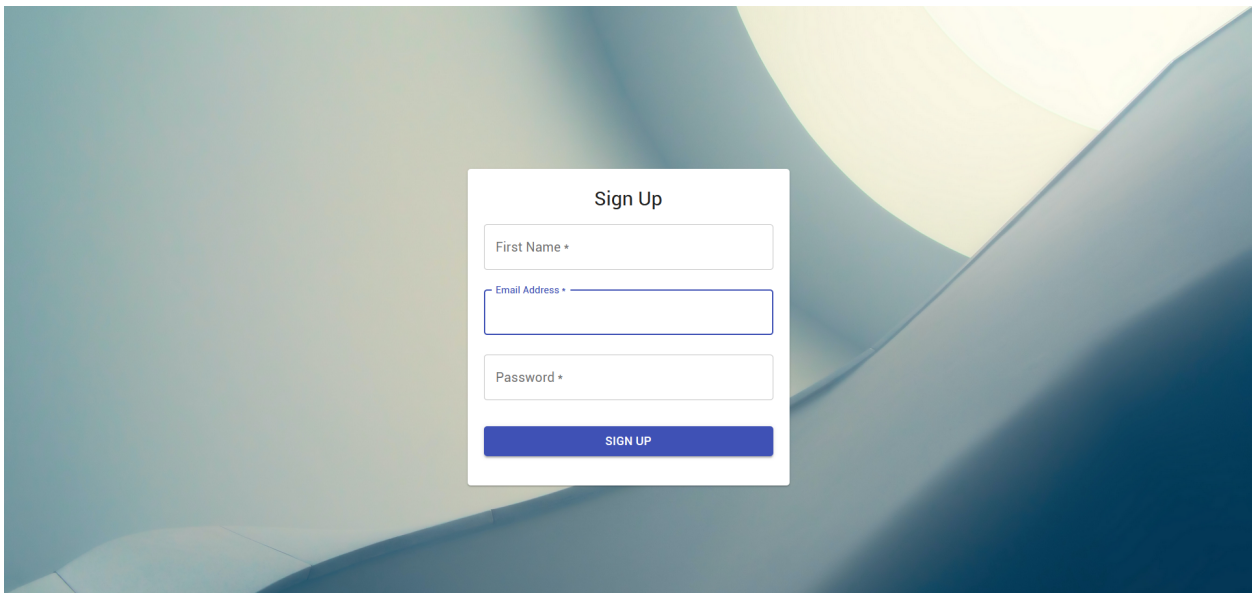
Signin page


Sign In
Email Address *
Password *
SIGN IN

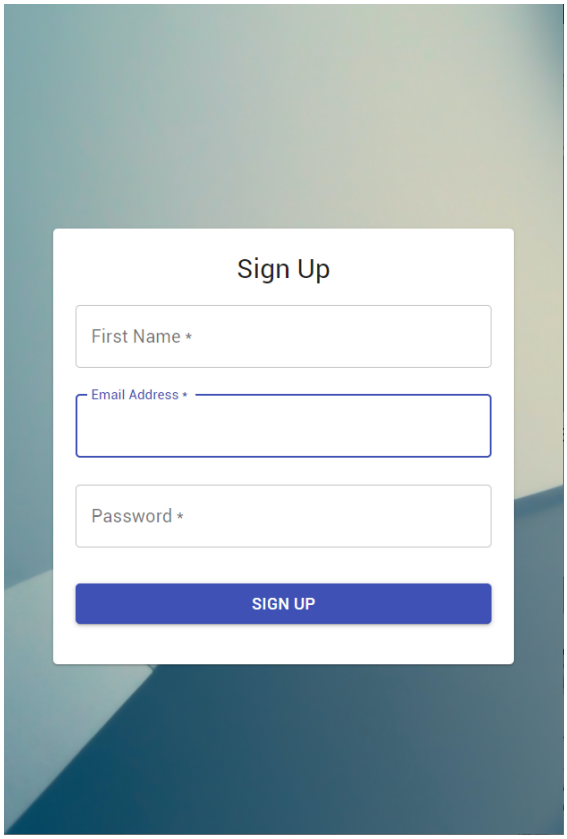
Mobile view:


Sign In
Email Address *
Password *
SIGN IN

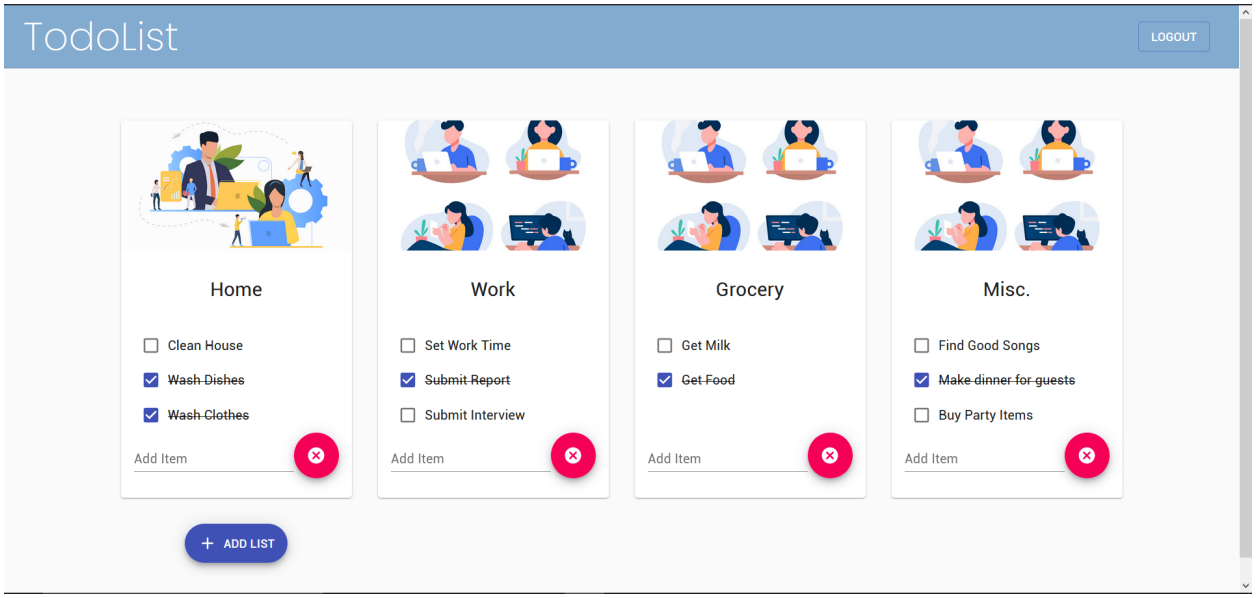
Signup page



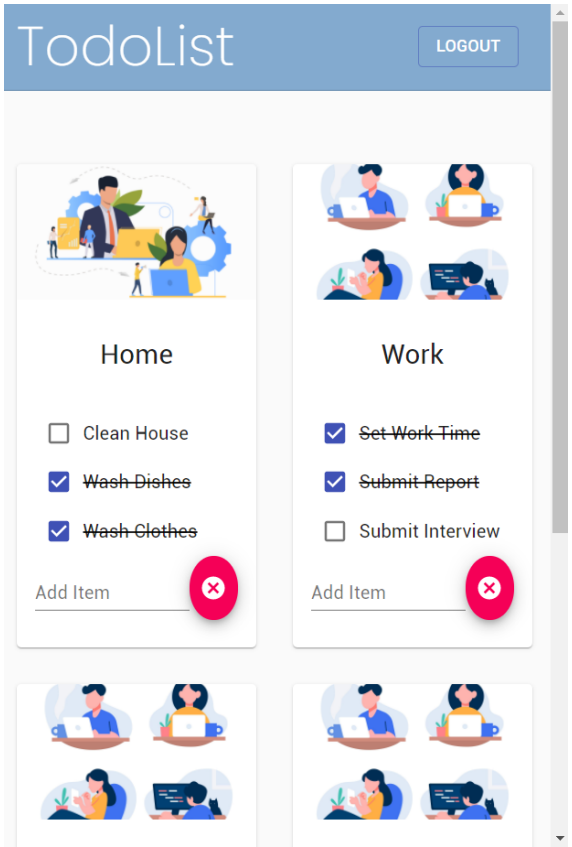
Mobile view:



Browse page



Mobile view:



Special Functionalities:

1) The Title for any list can be edited by simply clicking on it and starting typing.

2) Any new list starts with a random pokemon name(Potential conversion of project to be entirely pokemon themed.)

3) Complete Routing functionality implemented

4) Responsive design

5) Real time connection with database