Documentation backend for social media

1. For backend
    a. Create folder
    b. After that, change to same directory
    c. npm install -g @sanity/cli
    d. if any conflict use npm install –legacy-peer-deps -g @sanity/cli
    e. then, sanity init
    f. data configuration -> y
    g. chose template
    h. change in git ignore of outer folder (change /node moduler to node module)

2. Create the necessary schema for social media
    a. Create comment, pin, postedBy, save, user component and import in index.js
    b. Write the necessary schema in all the component
    c. Example

```
export default {
    name: ‘user’,
    title: ‘User’,
    type: ’document’,
    fields: [
        {
            name: ‘userName’,
            title: ‘UserName’,
            type: ‘string’,
        },
    ].
}
```

Documentation frontend for social media

1) **Installing the packages**
   a) Install the tailwind css and include the code in index.css.
   b) Import in index.js

      i) Import creatRoot from ‘react-dom/client’
      ii) Import {GoogleOAuthProvider} from ‘@react-oauth/google’
      iii) Import BrowserRouter as Router
      iv) Use this like

```
<GoogleOAuthProvider clientId= {process.env.xyz}>
        <Router>
        <App/>
        </Router>
</ GoogleOAuthProvider >
```

2) Create .env file outside of scr folder and write necessary code

3) **Create the client.js in src**
   a) Write code to connect with backend sanity (refer the social hub)

4) **In App.js**
   a) Import the login and home
   b) Create routes for login and home components
   c) (at the end of the project) fetch user data from local storage and protect the routes.

5) Create the utils folder (utility -> code where it can be used everywhere)
   a) Create fetchUserLS.js -> fetch the user login data from local storage
   b) Create sanitydataFetching.js -> fetch the data from sanity(refer the code)

6) **Create the container folder**

a) **Create Home.js ()**
   i) Import all the components (sidebar, userProfile), HomeImage, other necessary file (refer the code)
   ii) Tailwind design for smaller device then other device sizes
   iii) Detail explanation provided in code base.

b) **Create Homeimage.js()**
   i) Import search, pinDetail, navbar, feed createpin from component
   ii) Create routes for each of them

7) **In components**
   a) **In login**

      i) All the code are written
      ii) After google login-> decode using jwt_decode and save the data in localStorage
      iii) Localstorage data is used globally in all other components such as, app.js, home.js, …
      iv) Create fields for backend like sanity to populate these data and post it into the database.
      v) All the other explanation available in the code

   b) **In sidebar**
      i) Userdata is pass through Home.js, it is used various part to populate the data, and to navigate user to other routes such as user-profile in user's image
      ii) Import categories from sanitydatafetching
      iii) Map through the name and routes user to other link like categories/category.name
      iv) All these route are mentioned in HomeImage

   c) **In HomeImage -> Feed**
      i) Use useParams to the id of the category while clicking the category

ii) If there is categoryId then fetch the pins and posts using searchQuery else fetch data using fetchquery
iii) Import the MasonryLayout and pass the data to child

d) <u>In MasonryLayout</u>

i) Map through the data and pass the data to Pin component with necessary data.

e) <u>In Pin.js</u>

i) Data is pass as props from Masonary Layout
ii) Create the element over image like, save, link, download, etc…
iii) Refer code

f) <u>In PinDetail.js</u>

i) Write necessary jsx, details are provided in same file

g) <u>In Navbar.js</u>

i) Write jsx
ii) Populate the props pass down from HomeImage

h) <u>In Search.js</u>

i) The populated state from navbar is passdown from homeimage
ii) Fetch the data from searchQuery and Pass the pins to Masonary Layout

i) In navbar <- search logo, search in another component,
j) In navbar <- createpin logo, route in homeimage
k) In navbar <- userprofile logo linked through routes in homeimage