**Creating a project:**

npx create-react-app my-app

cd my-app

npm start

**Creating frontend:**

App starts from App.js, jo there and remove everything in return.

Create components, import them and use them in App.js -> That’s all about the basic frontend

**Jumping to backend:**

Install mongoose, express, nodemon

Mongoose is a Node. js-based Object Data Modeling (ODM) library for MongoDB. It is akin to an Object Relational Mapper (ORM) such as SQLAlchemy for traditional SQL databases. The problem that Mongoose aims to solve is allowing developers to enforce a specific schema at the application layer.

Express is a node js web application framework that provides broad features for building web and mobile applications. It is used to build a single page, multipage, and hybrid web application. It's a layer built on the top of the Node js that helps manage servers and routes.

**Enough with theory, let’s start with Backend:**

* Create a folder backend

In that folder create two different folders:

* + Routes
  + Models

Models is where we will be defining schema.

MongoDB is a schema-less database but we will have to define schema for some purposes. We can use mongoose to do this.

Import mongoose and store it in an object.

Extract Schema from that object and create an object of Schema class and assign values/fields to that object of Schema.

Once done this, make a mongoose mode with name of the model and Schema object attached to it.

Export the object so that others files can call and use the schema but using required. This Schema will be called by the mongoose model name that you provide.

**Now go to the routes folder that you created:**

This is where you will be using express js. You will define how you want to handle request here. Requests to backend from front end.

You will just send specific requests from front end, depending upon which request is sent, you’ll decide which data to provide from the backend to the frontend. You can perform encryption, data validation and other things here. This is what express.js means, a framework built over node.js

Start with creating an router instance from express.Router()

* Note that Router is a function unlike Schema which was a class. A **router instance** in Express is a **mini Express application** that you can use to define routes, middleware, and route handlers independently from the main Express application. This allows you to modularize and organize your route logic by grouping related routes into separate files or modules.
* Import Schema that you created
* Import json web token:

This is a popular library for creating and verifying JSON Web Tokens (JWTs). JWTs are a compact, URL-safe means of representing claims to be transferred between two parties. The claims in a JWT are encoded as a JSON object that is used as the payload of a JSON Web Signature (JWS) structure or as the plaintext of a JSON Web Encryption (JWE) structure.

* Import body and validationResult functions

Now simply start writing the request

* Name
* Checks
* Finally the function-> what will it do, what data to fetch and what to do with the fetched data

**Fetching data:**

Again you have to create a post request to fetch the data and display it.

// To avoid reloading page -> avoid href in link

// Solution: react router DOM -> Link to=”/…”

// don’t forget to install bootstrap, even if we don’t directly use it, we are using some elements (like carousel) that might require some js files that we don’t have, and hence require bootstrap

Coming back to fetching data: (vdo 9)

We need to define some global variables to fetch data!!

Define that in db.js as fetched\_data = db.collections(“name”);

Once the data is fetched, just use it wherever required, actually use it in requests, as in return the data when a specific endpoint is hit.

Implement search functionality using useState -> yet to be studied