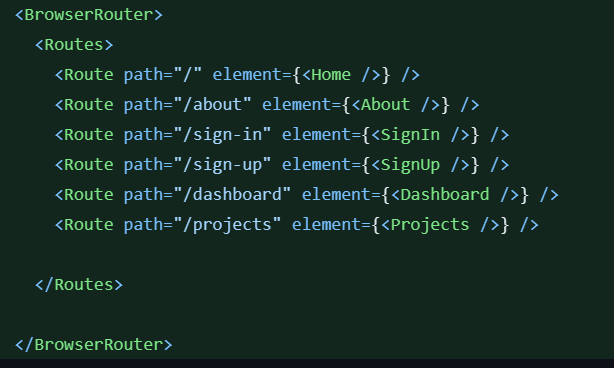
1. Create pages and routes

* Installed react-router-dom for creating routes
* Syntax for routing (imported browser router route and routes from react router



1. Creating header section

* Installed flowbite easy to use
* You need to check how to install flowbite by using documentation its related with tailwind
* For header section we use  which is coming from flowbite inside this tag we can use tailwind classes
* Since we want when we click the logo we are redirected to home page we need to wrap the logo inside the Link tag
* Link vs anchor
* Link tag prevents a full page reload when navigating between different views in a single-page application.
* The HTML <a> element is used for navigation in traditional web applications, causing a full page reload when the user clicks the link.
* Next we want a search box so a form is used inside form text area is required which is coming from flowbite
* We want another three pages so hamburger menu we use  and also  this is not self closing we wrap this to all the pages such as about , project etc
* 
* In the photo link Is used like that along with the definition of path , as div is used because anchor tag inside anchor tag
* useLocation retrieves the current location using the useLocation hook and then extracts the pathname property from the location object. This path variable will contain the current URL path of the React Router application.
* For example, if the current URL "https://example.com/products", then path will be set to "/products". This can be useful for dynamic rendering or conditional behavior based on the current route or path in your React application. Active keyword is used here to give effect css for that particular page

1. Create and run the server

* Installed nodemon and express
* We need to define type as module inside package.json
* `import express from 'express';`: This line imports the Express.js framework using ES6's `import` statement. In older versions of Node.js, you would typically use `const express = require('express');`, but with ES6 modules, you can use the `import` syntax.
* `const app = express();`: This line creates an instance of the Express application. The `app` variable is an instance of the Express application, and it is used to configure routes and settings for your web server. `app.listen This part of the code makes the server listen on port 3000. The `listen` method takes two arguments: the port number (3000 in this case) and a callback function that will be executed once the server is successfully running. In the callback function, a simple log message is printed to the console indicating that the server is running.

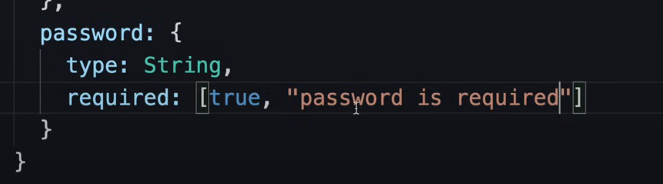
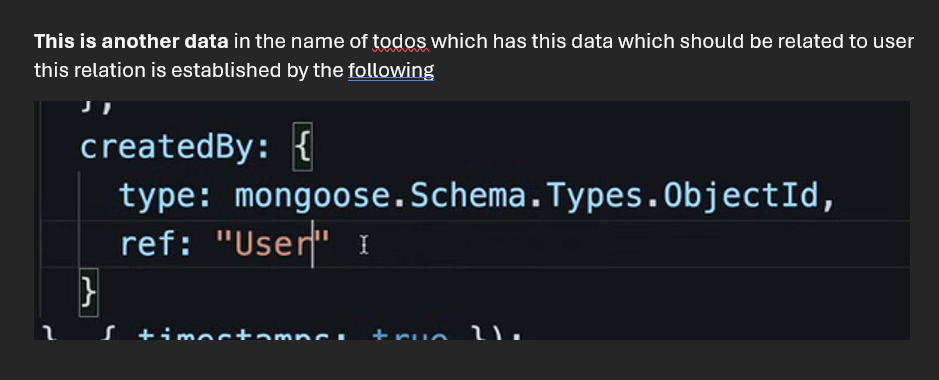
1. Connecting database- Mongo DB

* Nothing here it’s a generic code nothing else

1. Creating user model

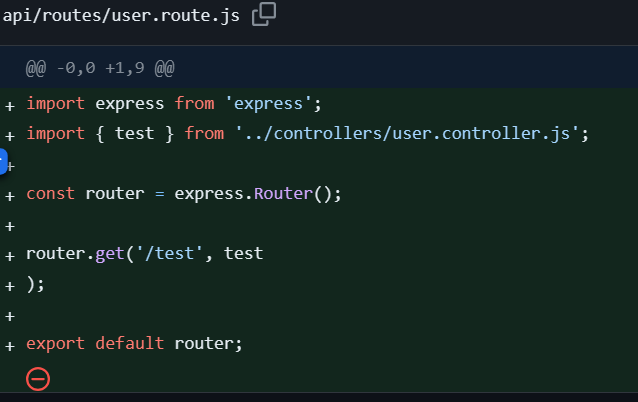
A screen shot of a computer program

Description automatically generated

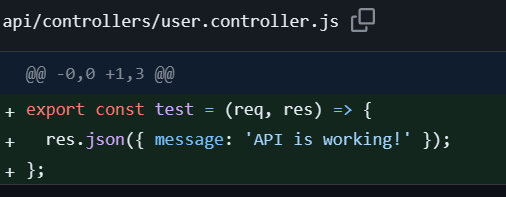
* We can write custom message also if the required field is not give like this 
* We use ref key word 

1. Creating a test api route for testing

* For any api we need two things route and controller
* Here api means handling authentication,sign in sign up etc
* Firstly we need to create a route in that given path like this



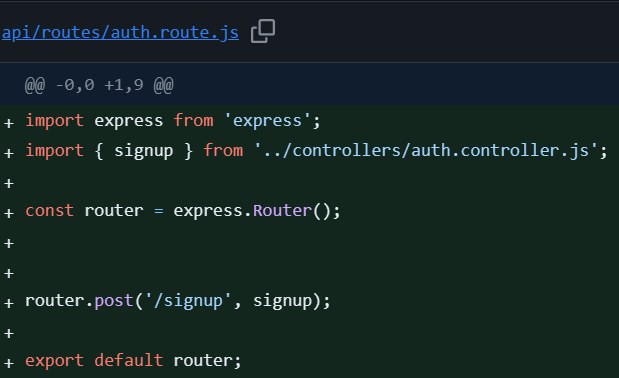
* Then we need to create a controller like this



* Then lastly we need to write this in index file by importing userRoutes



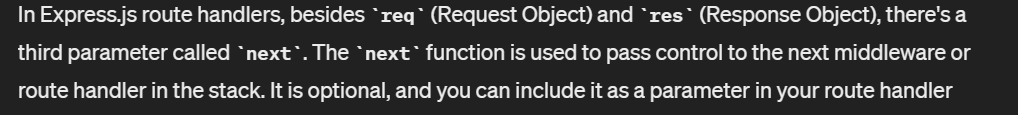
1. Creating sign-up api route

*  similar as told above
* Now we’ll make signup function
* While creating all these functions keep in mind that we get 4 types of parameters which are



* A black background with white text

  Description automatically generated
* A black background with white text

  Description automatically generated
* 
* A black background with white text

  Description automatically generated
* A black background with white text

  Description automatically generated