Chapter 4 - React setup

*In this tutorial , we learn how to do react setup ? First thing before starting our setup , we have to organize our codes so that we can have parallel services running in the frontend . Create one folder and call it* ***backend*** *,move all the codes that we do in previous tutorial and put it in this folder* ***(backend).***

***Steps to run the server from these two different directories .***

***Step 1*** *- If we want to start server from backend folder , the command we use to switch from main directory to backend folder is -*

***cd backend***

*With the help of this command , we will switch to the folder backend .*

***Step 2*** *– After switch to backend folder we will now to run the server ,*

***npm run dev***

***Step 3 –***  *If we want to come back to main folder from backend folder ,*

***cd..***

*So now ,after organize the codes we will do setup of react application.*

***npm i create-react-app frontend***

*So after setup we will cleanup an unused code that we have .*

*First thing we will do , create new folder in src and name it as* ***“components ”.*** *in this folder we have different different components like login ,registration ,dashboard and etc .*

*There is library that is called react-bootstrap , which is the version of bootstrap . On terminal , we are going to install this library .*

*npm install --save bootstrap*

*After finishing , let’s install react-bootstarp library .*

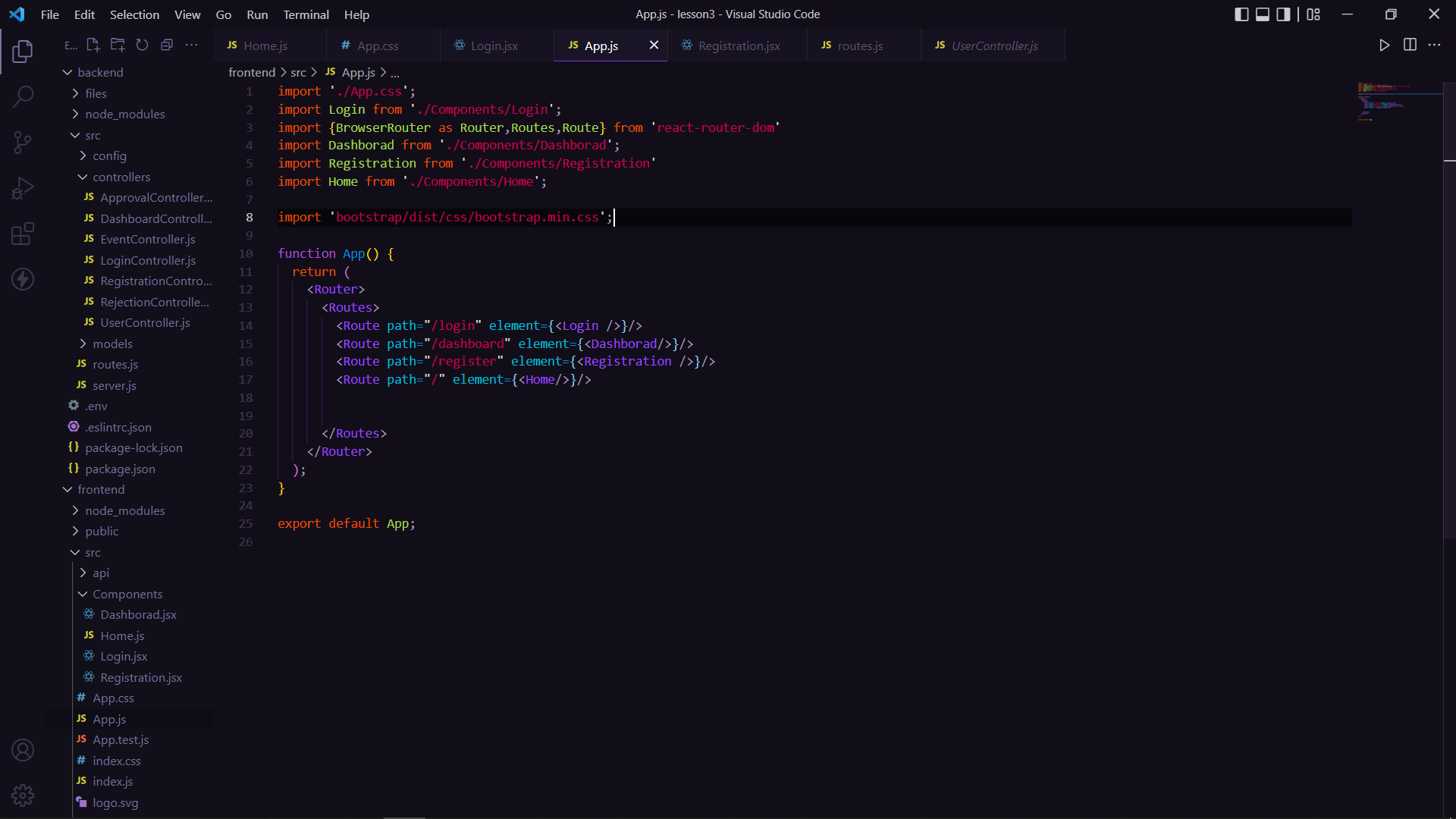
*npm install - -save react-bootstrap*

*To import bootstarp CSS in* ***src/index.js*** *file ,*

***import ‘bootstrap/dist/css/bootstrap.min.css’***

*Start the server once again and remember that you need to be inside the frontend folder .*

*After that , we will move to* ***App.js*** *file , and start writing some code*



*We are importing container from react-bootstrap .*

*React has a library they’re kind of very helpful us in navigating around the things , that is called* ***react-router-dom .***  *Stop the server and for installing*

***install i react-router-dom***

*Next thing is the configuration that allow us to use another library to fetch the data from API . Stop the server once again and write the below code*

***npm i axios***

*Axios is a Javascript library used****to make HTTP requests from node.******js or XMLHttpRequests from the browser****and it supports the Promise API that is native to JS ES6.*

*In source folder , create another folder* ***“api” .*** *Inside this folder we’re going to create an new file and will be called* ***“axios.js” .***

*import axios  from 'axios';*

*const api = axios.create({*

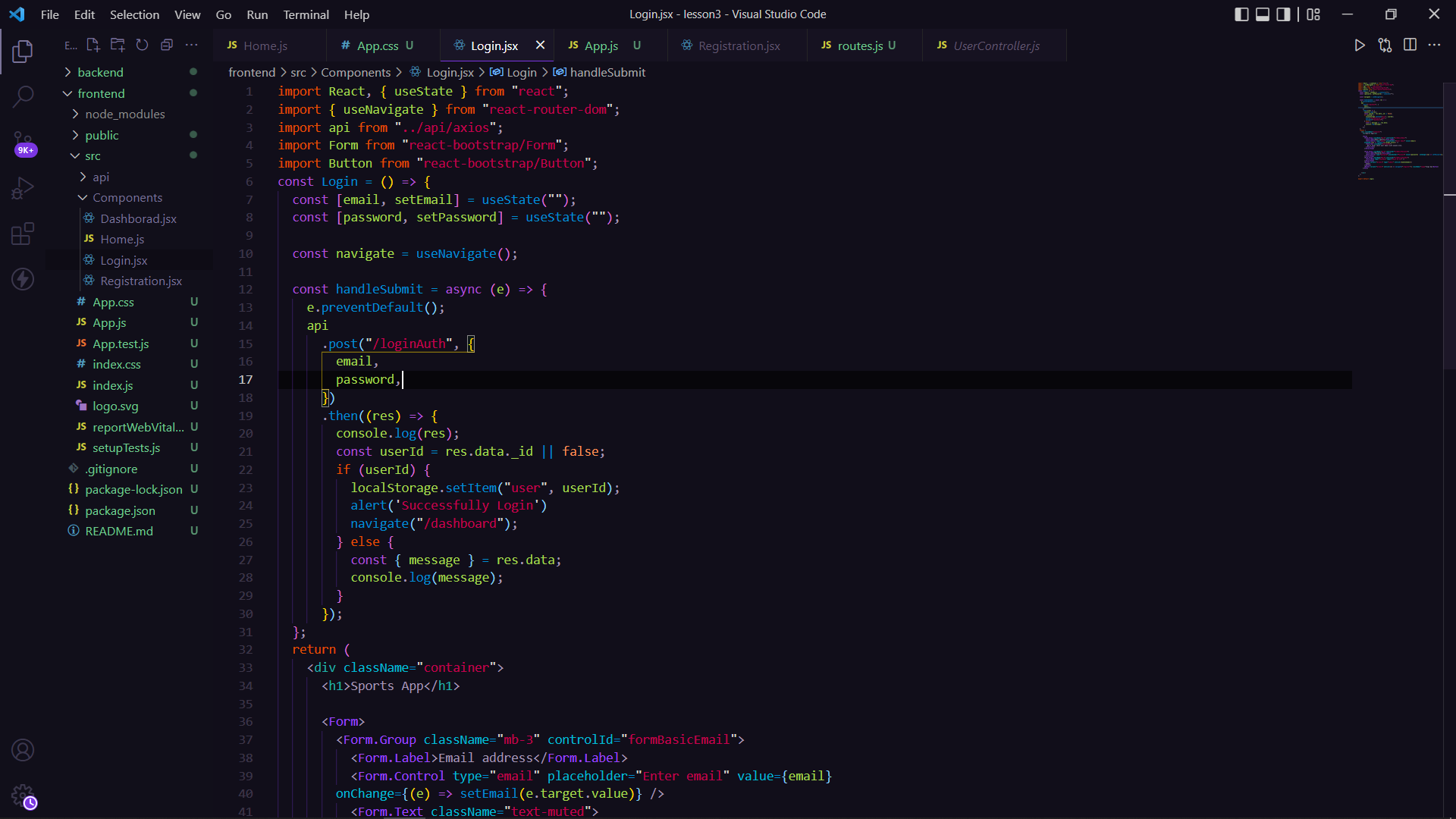
*baseURL:'http://localhost:8000'*

*})*

*export default api;*

*Using the create method , we are going to pass few configuration . Whatever URL we have at the backend we’re using here . As backend is running on Port 8000 , we are declaring in baseURL .*

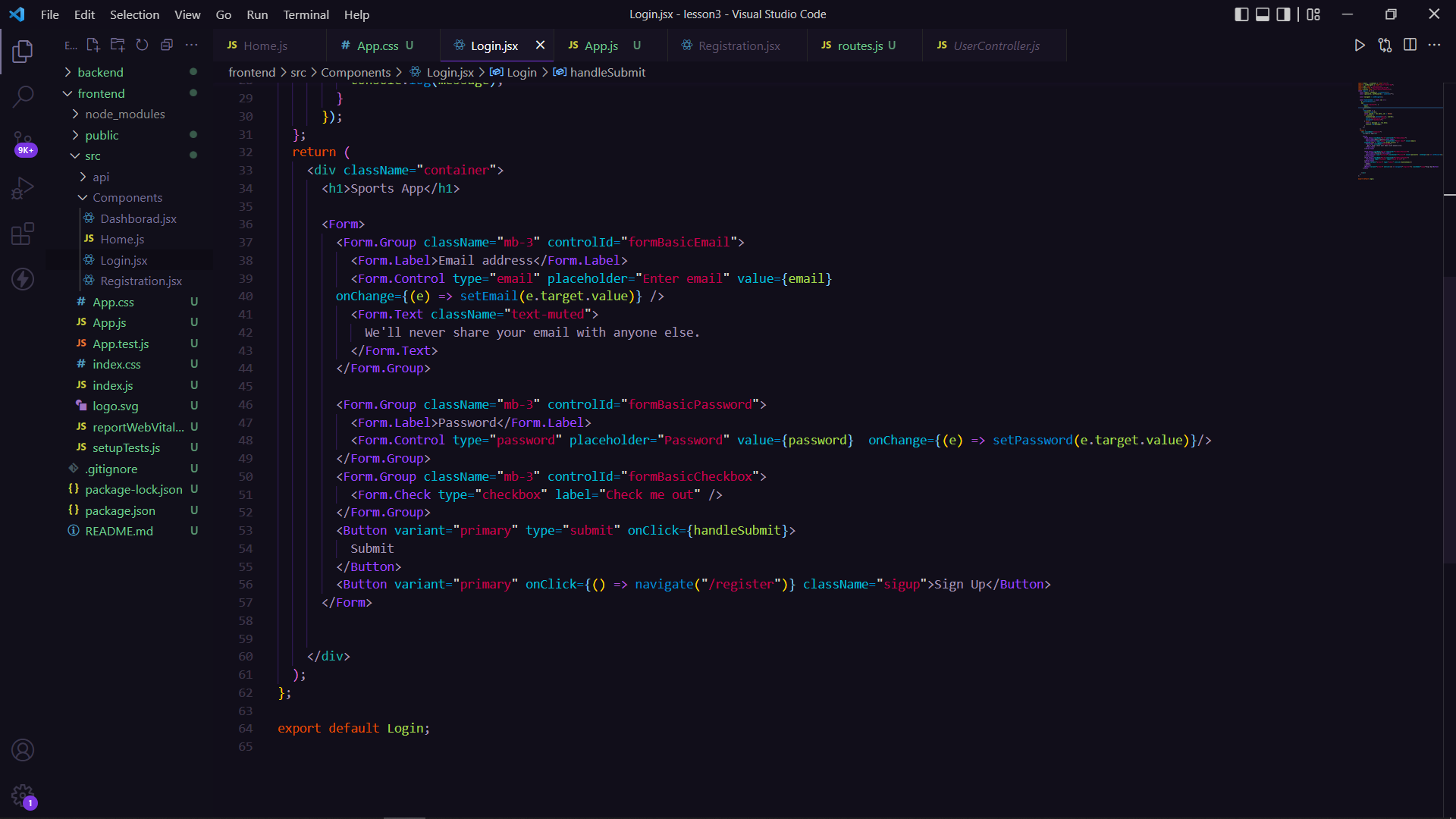
*Move in* ***Login component*** *.*



*Here , we are using* ***Hooks .*** *Hooks are special function that lets you “hook into” React features .* ***useState*** *is a hook that lets you to add react state into functional components.*

*We made one new function* ***handleSubmit*** *, which provides you some functionalities . Let’s see what functionality it offer,*

*preventDefault( ) – Whenever you click on Submit button , it will refresh the page always . So there is high chance of losing values that we type in textbox . So to prevent from all these we will use preventDefault method .*



*In input tag , we are basically pass another property that will be* ***onChange*** *. This is the function , that will take an event & it will pass the event to setEmail and inside the event we do not have to pass an entire event so for particular event we use* ***e.target.value.***

*onChange attribute - The handler function attached to this attribute will be invoked every time whenever the user types something into the input field.*

*Use API to send value to the server and check against the server .*

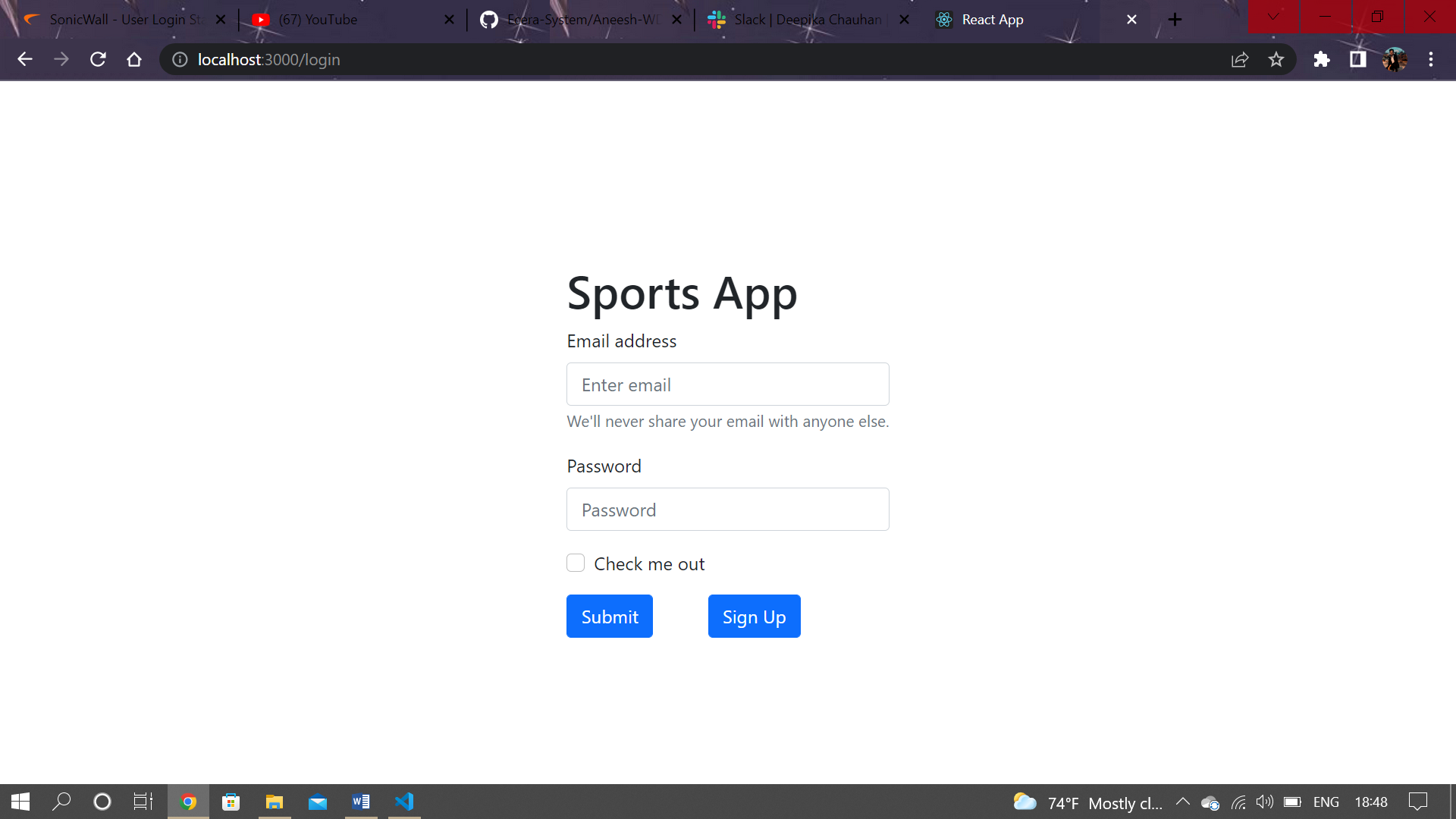
*If the response has an ID ,then we want to save it inside the browser and the by keyword* ***navigate***  *it will force you to go to another URL in this case we are redirecting to* ***dashboard page*** *. And if you are not login it will throw an error message that we will put in backend code*

*It is the history of your navigation that the router route send to us everytime whenever you go for different URL so that we can keep track where you was where you are going to .*

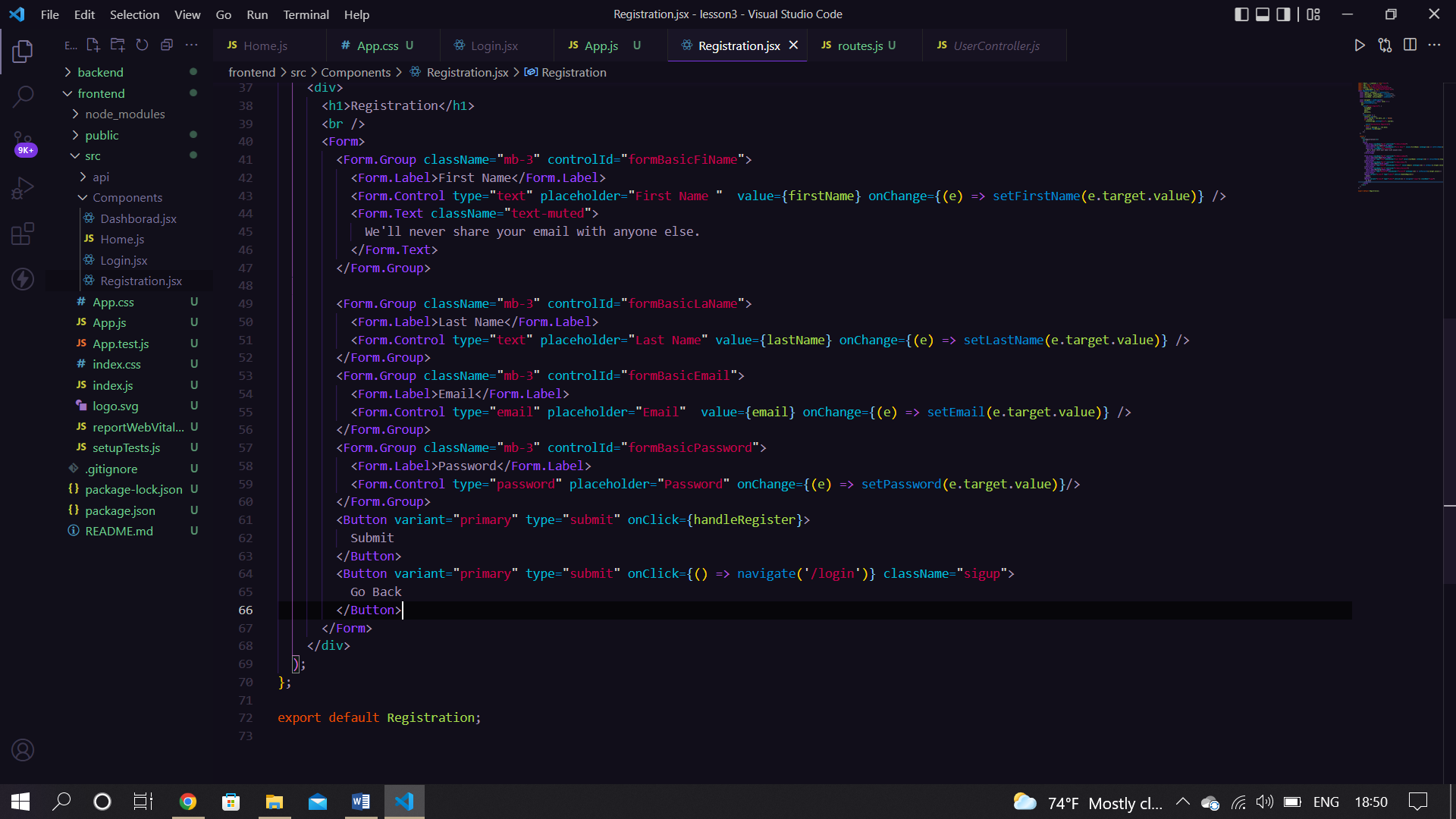
*Now ,it’s time to run code by server and test it . So , we will follow the following steps –*

***Step 1 -*** *Open Insomina or any other API that you are using prevousily . Here I am using Insomina . From getUserById request, I copy EmailID and I hope you remember the password that you write while creating new user . So go on frontend UI and paste email ID, type password and click on Submit button .*

*Step 2 - After Clicking on Submit button , you will redirect on Dashboard page . If you type wrong mailId and password it will give you error message and you can see Network (Ctrl + Shift + I ).*



*Create another component and name it* ***“Registration”*** *. We have the same code that we have in Login index file . Only few more event is add ,*



***“localhost:3000/register” .***

