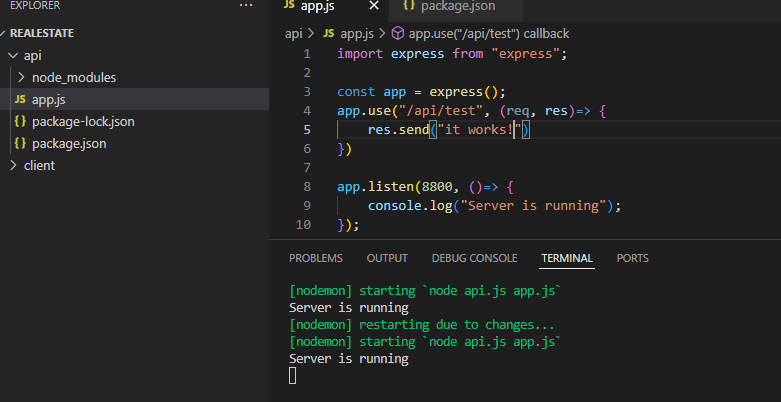
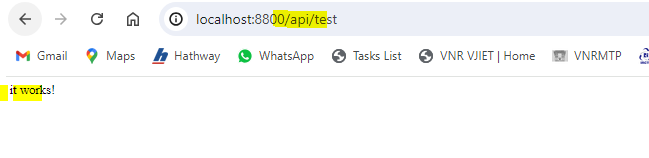
**Real Estate project**

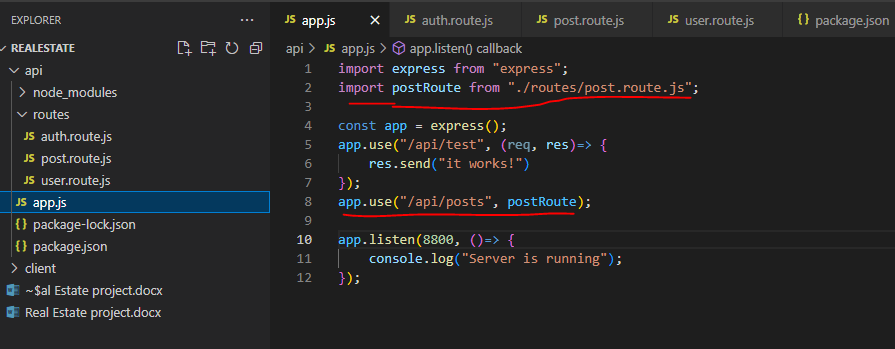
Create project folder and initiate app via express

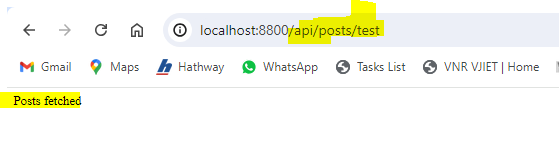


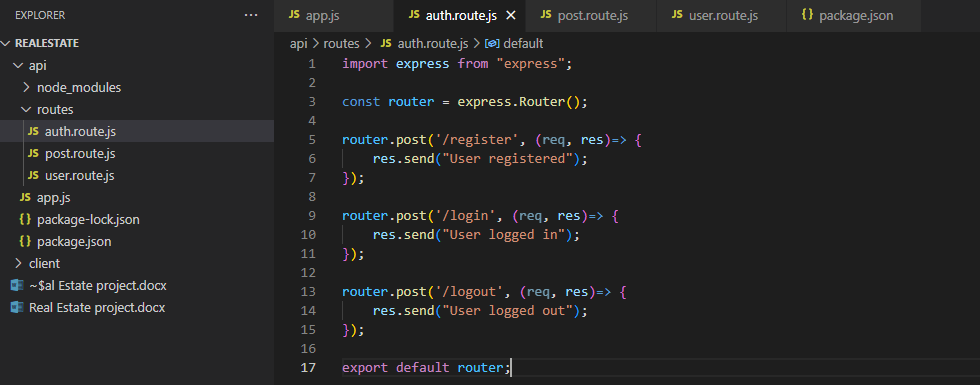


## **Test Routes**





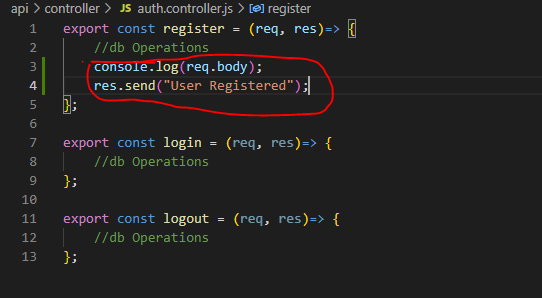


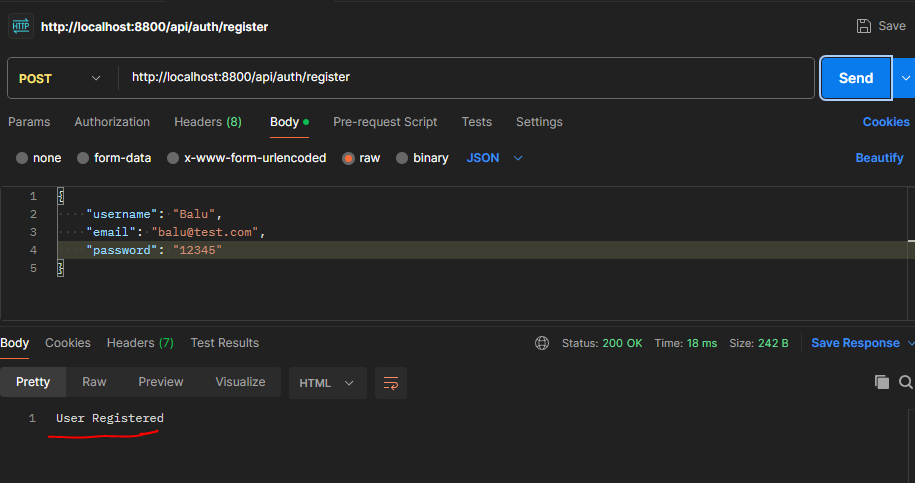


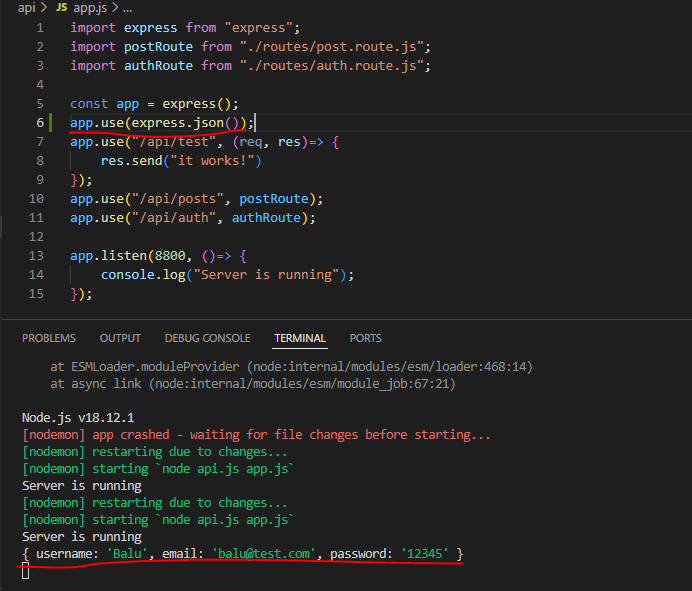




### **Register API test**

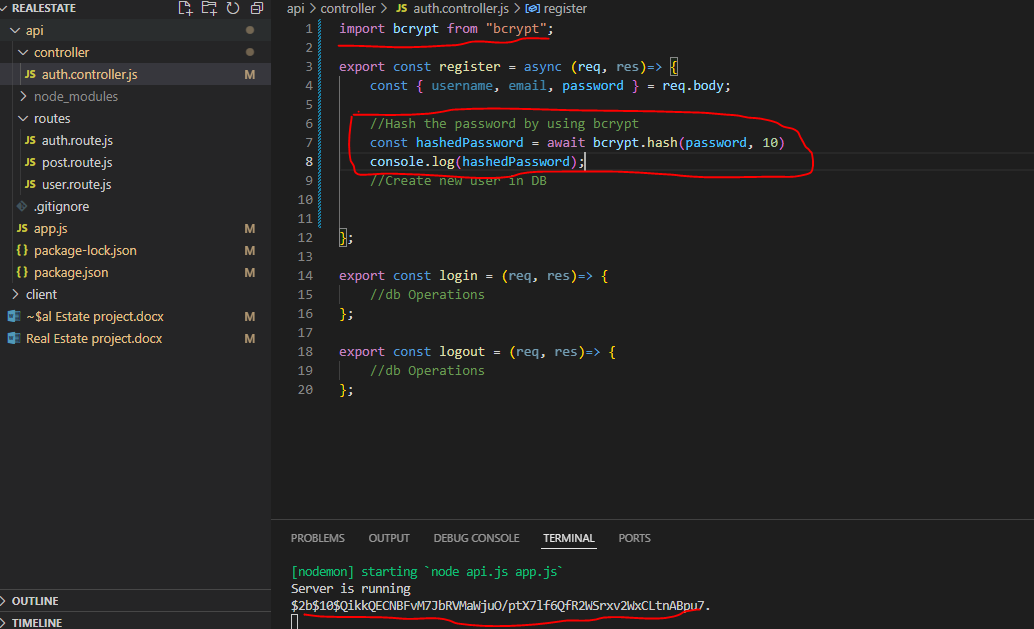






## **Bcrypt – Used for hashing passwords**

Hash the passwords using bcrypt => npm I bcrypt



## **Prisma**

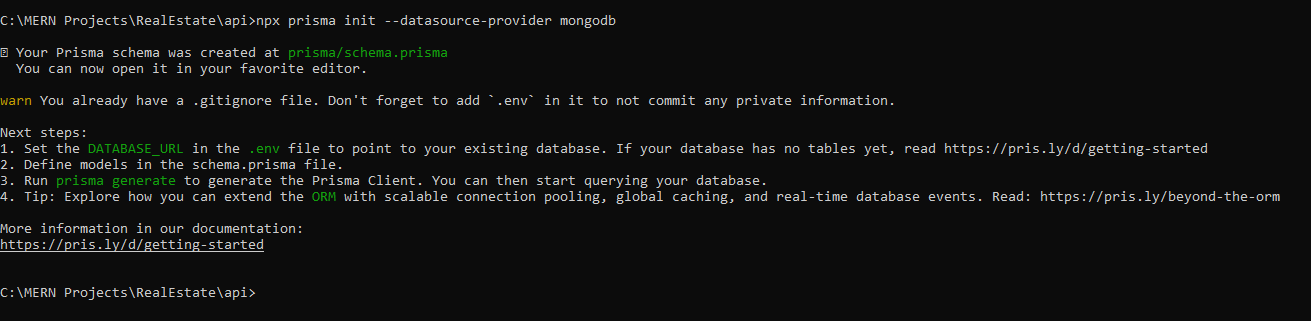
To store the user details to db we use prisma

Install => npm I prisma

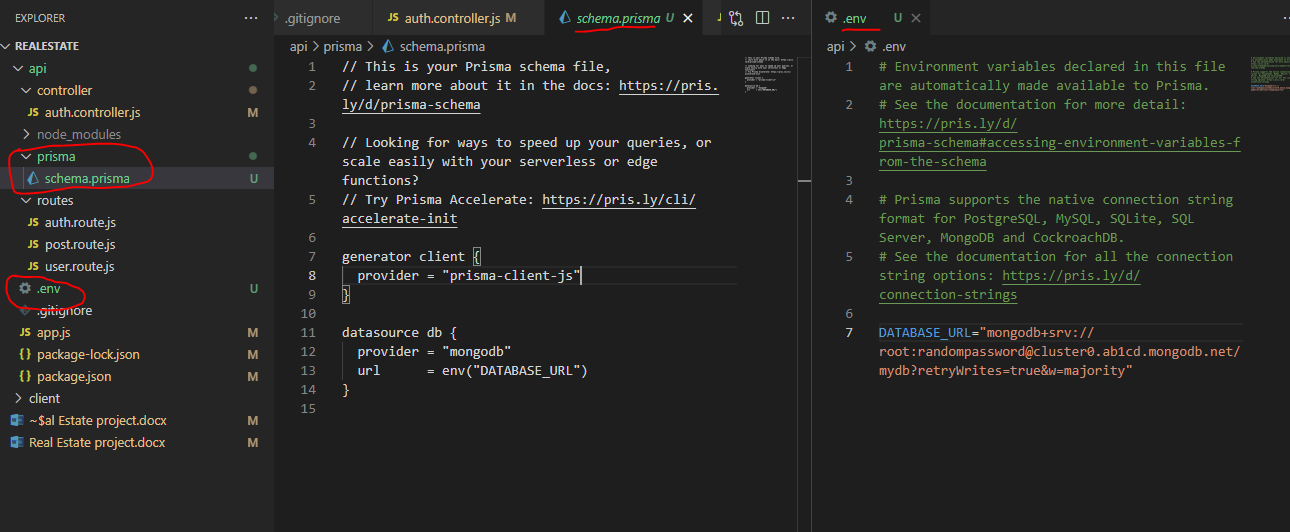
Follow <https://www.prisma.io/docs/getting-started/quickstart>

Now initialize prisma

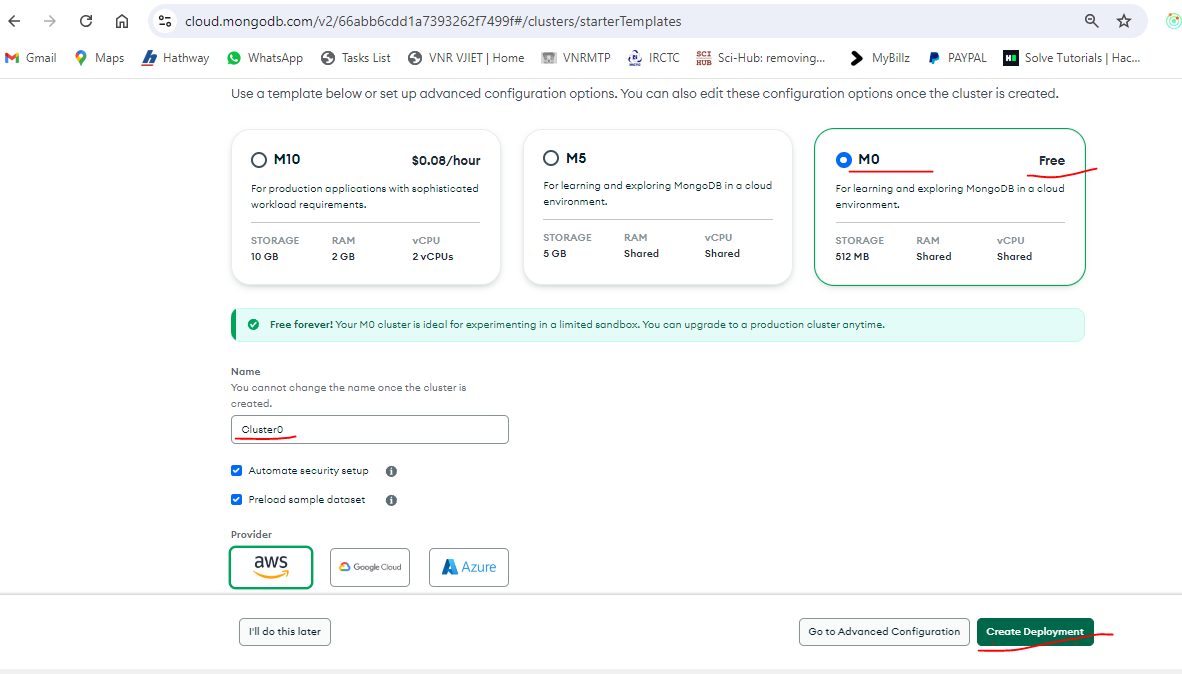
Run => npx prisma init --datasource-provider mongodb

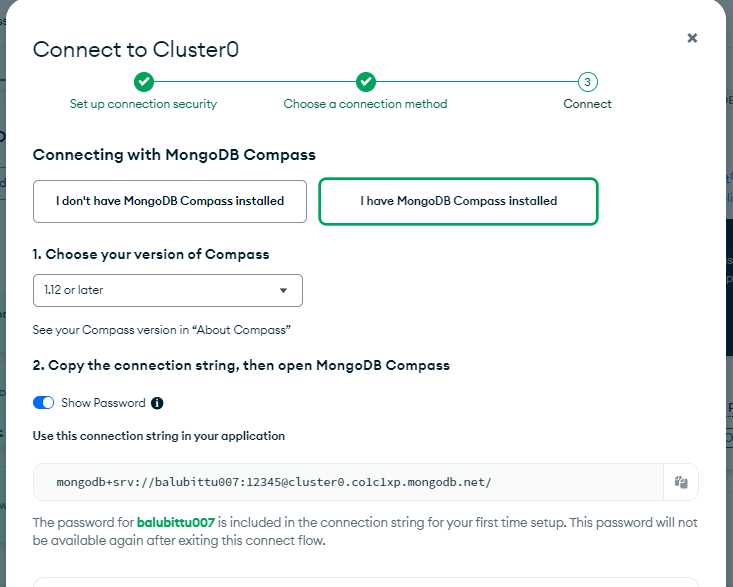


A new folder prisma is created also the env file is generated



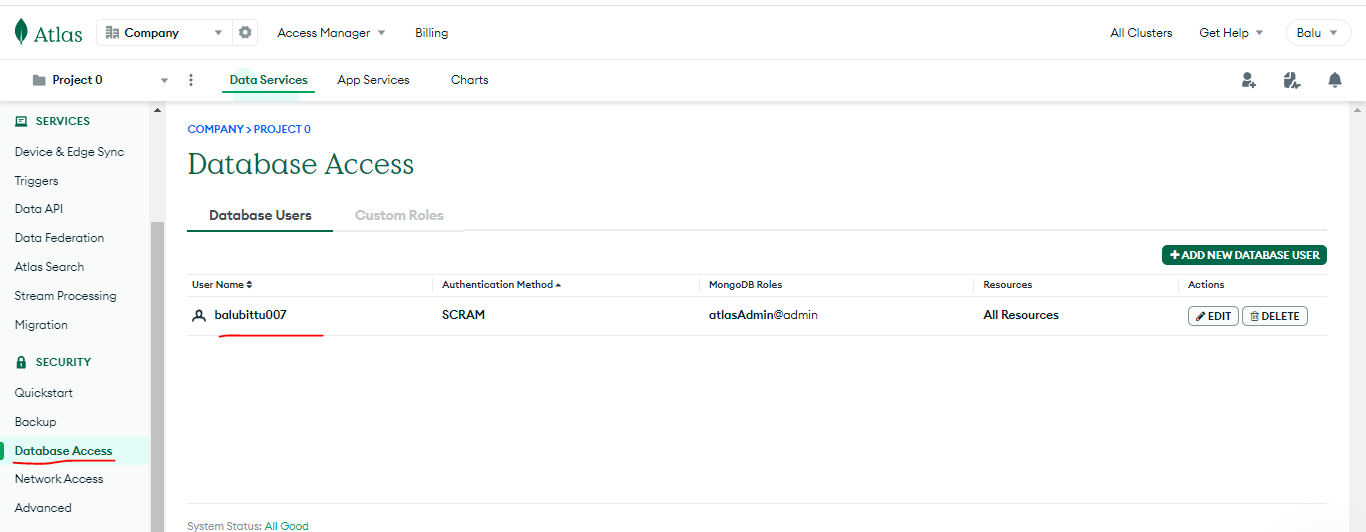
Create deployment in atlas



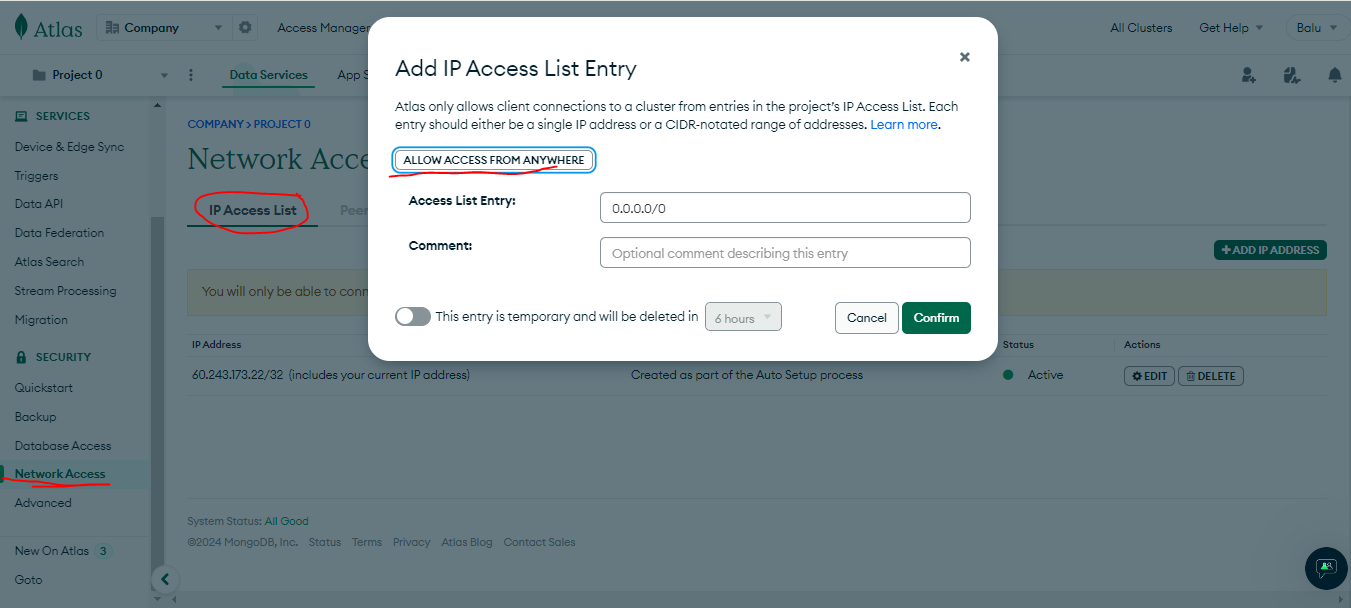


mongodb+srv://balubittu007:12345@cluster0.co1c1xp.mongodb.net/

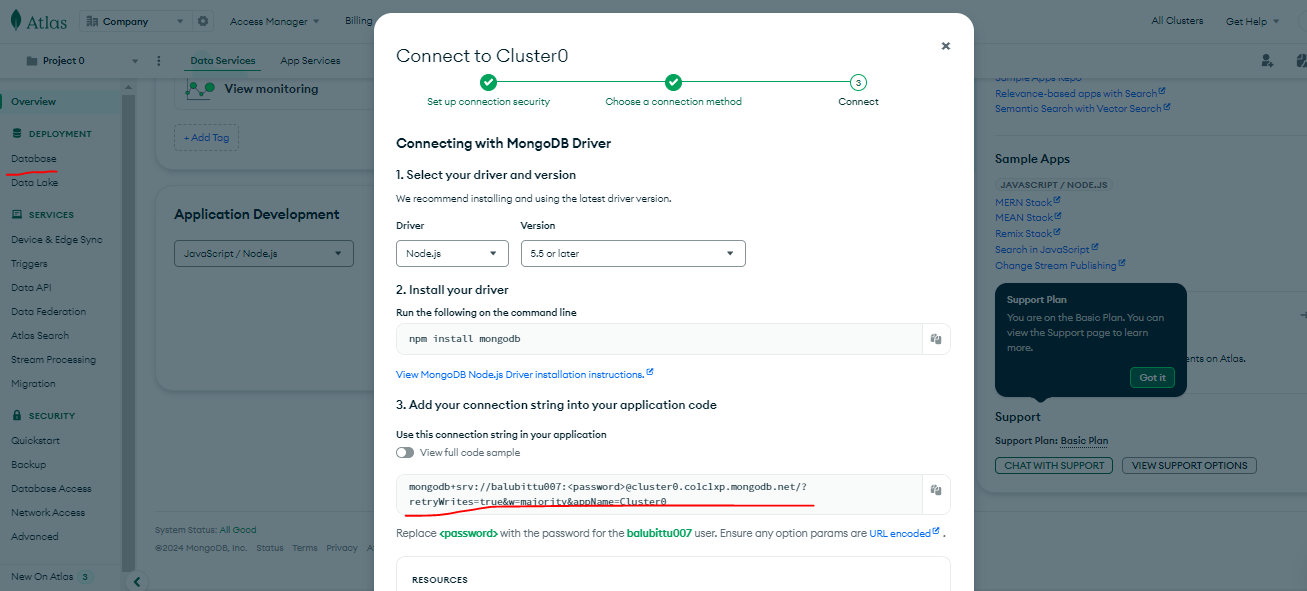
We can see the users in Database access

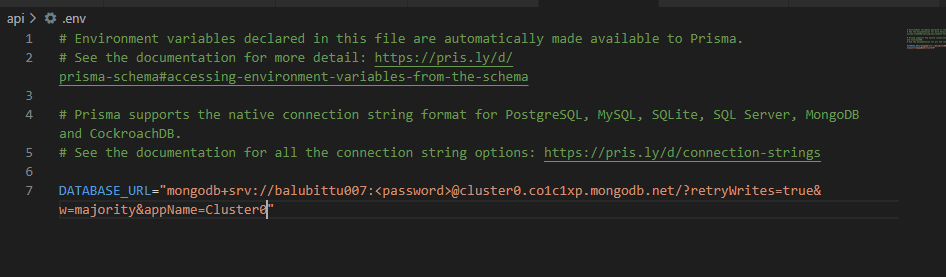


After deploying the app we need to enter server ip

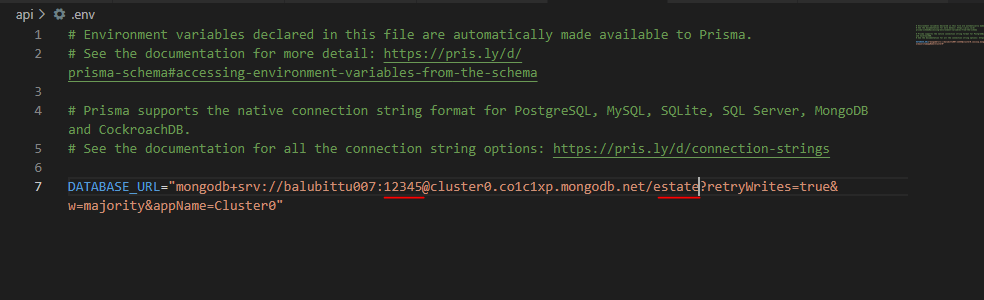


Get connection string from here



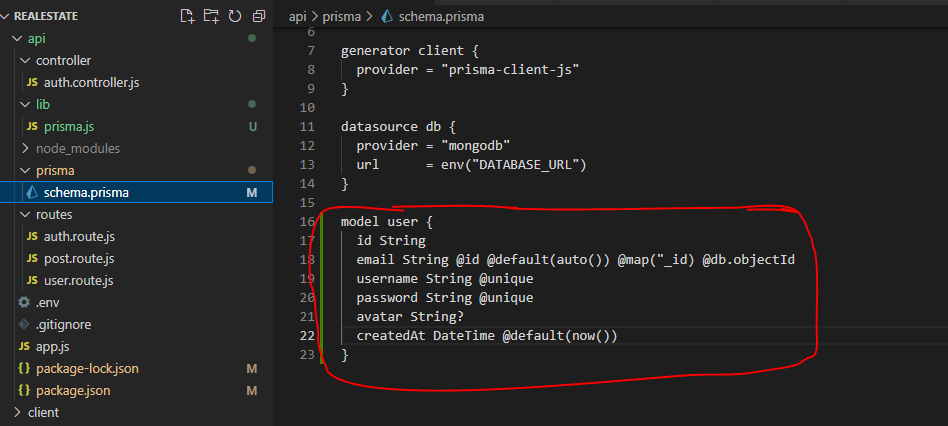


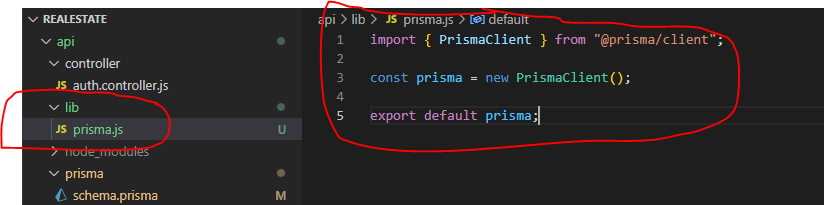
Update password and collection name in the url



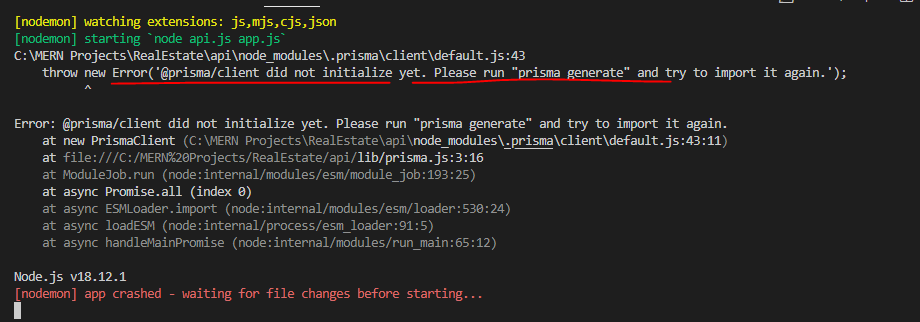
### **Creating user model:**

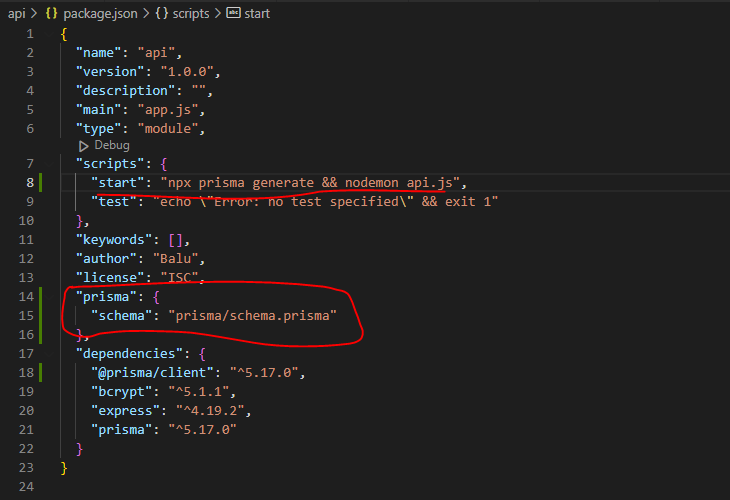
Run npm I @prisma/client





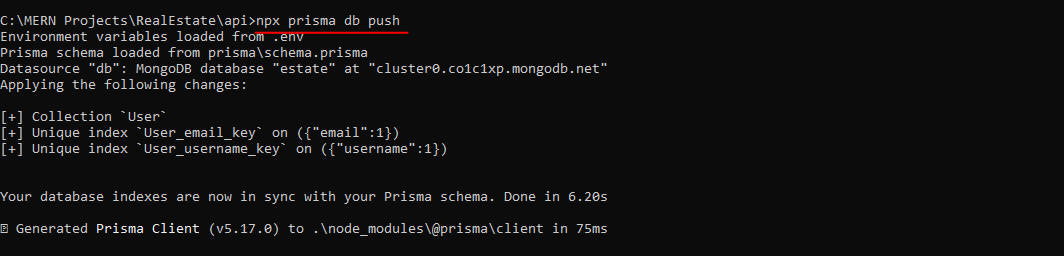
On running the server got the error below

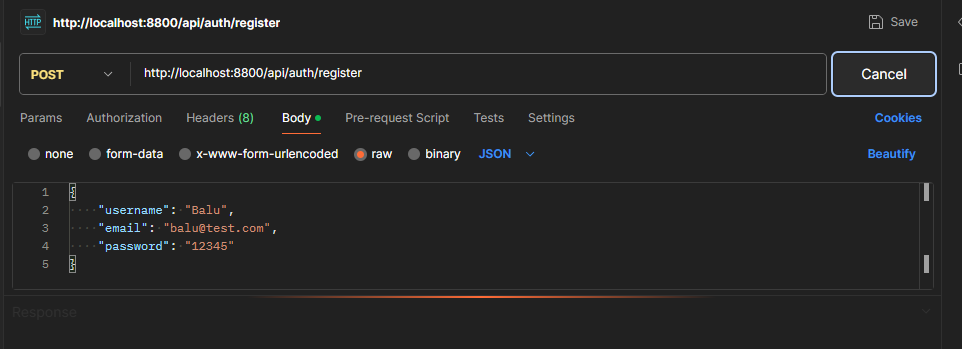


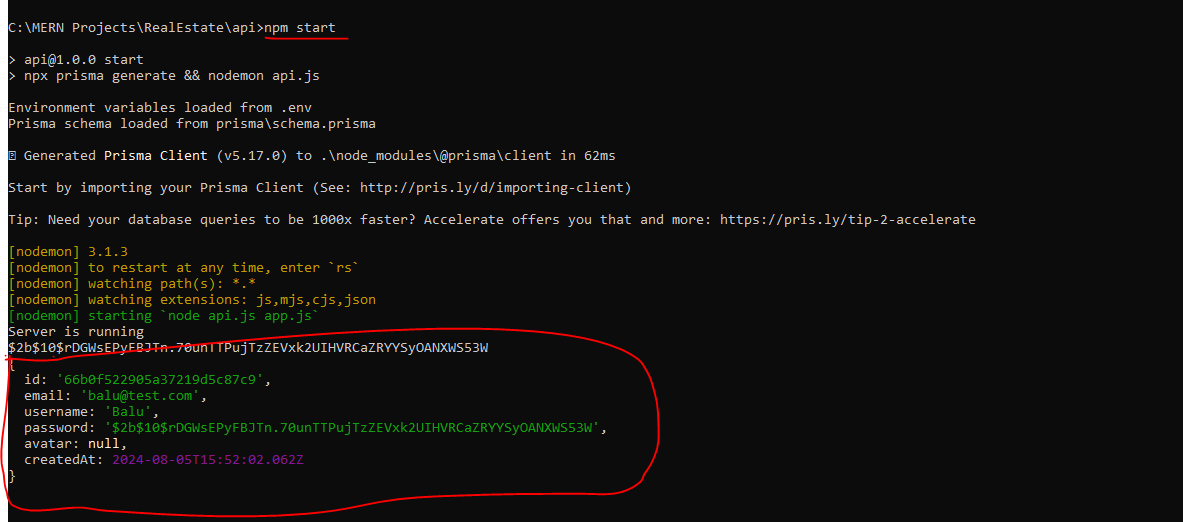




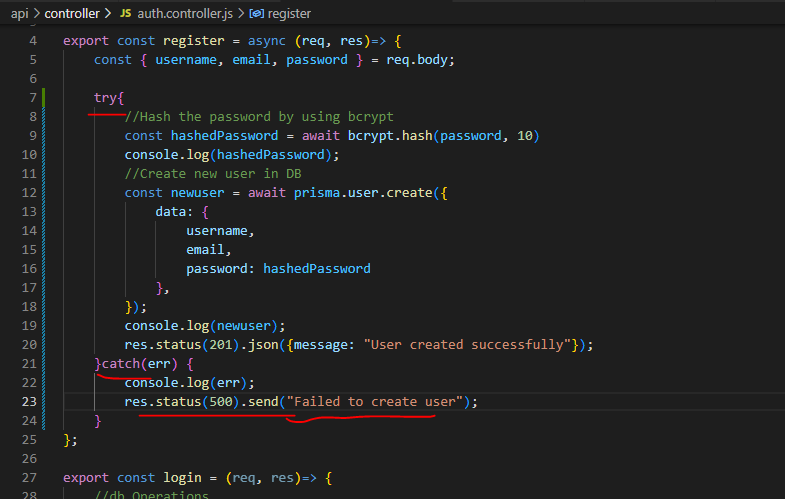
The error is caused due to not pushing changes on prisma schema to db.=> npx prisma db push



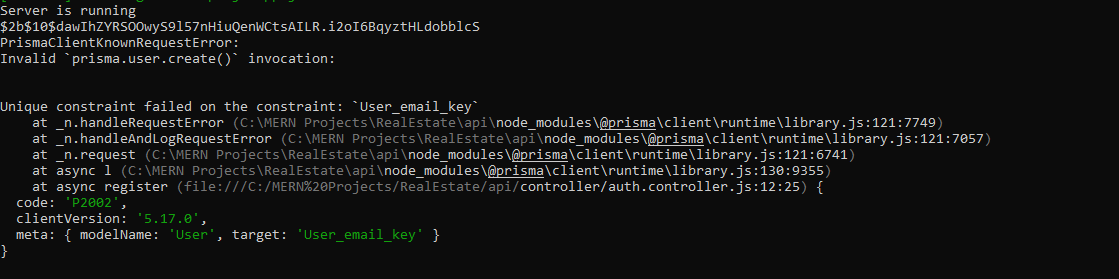
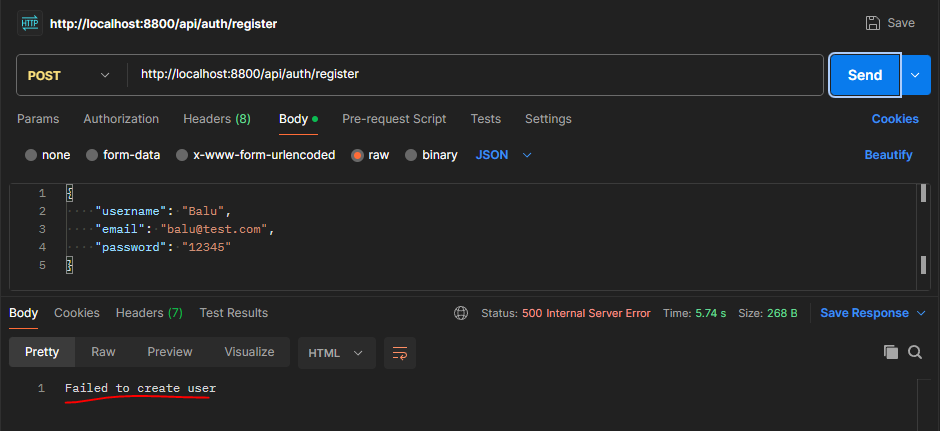




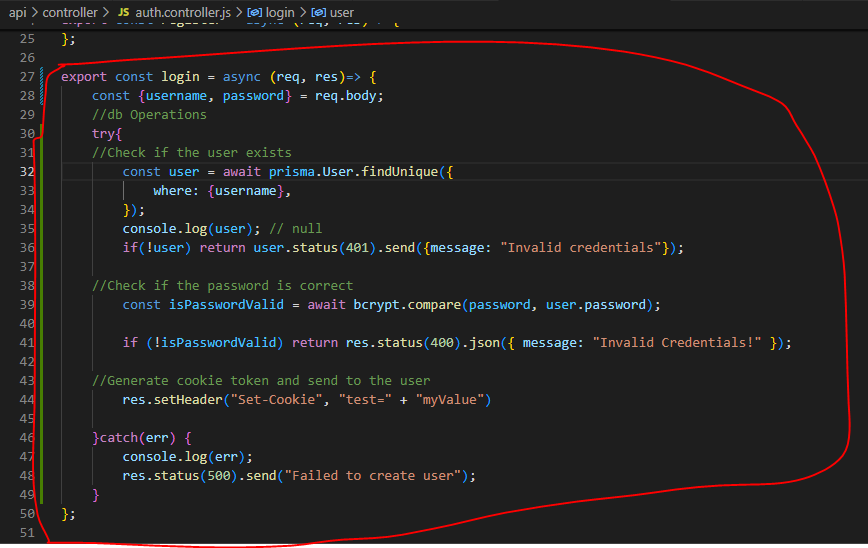
### **Register route:**

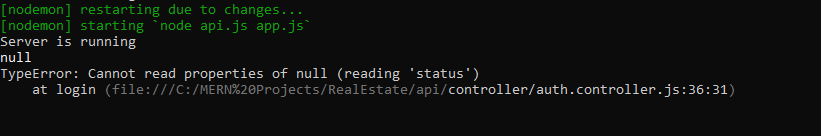


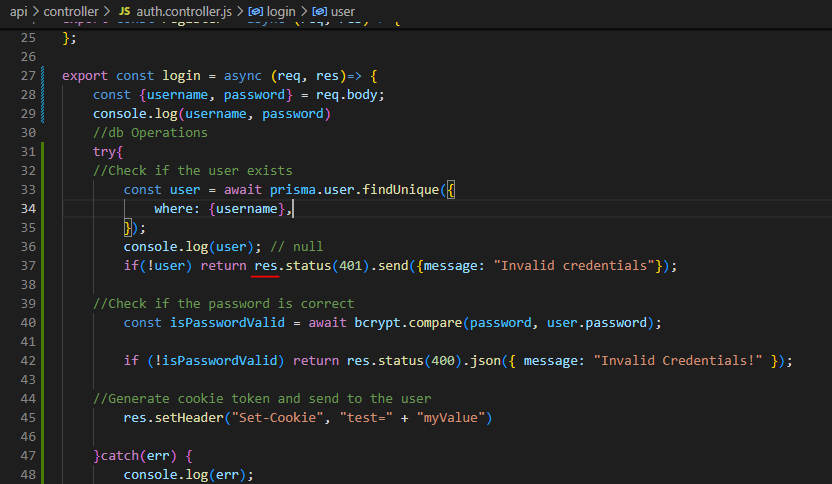
User creation failed due to non unique data sent via post request

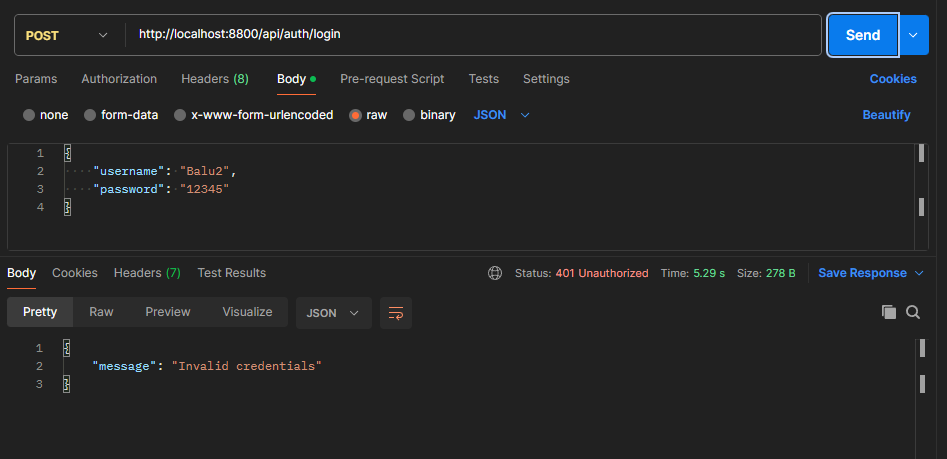
 

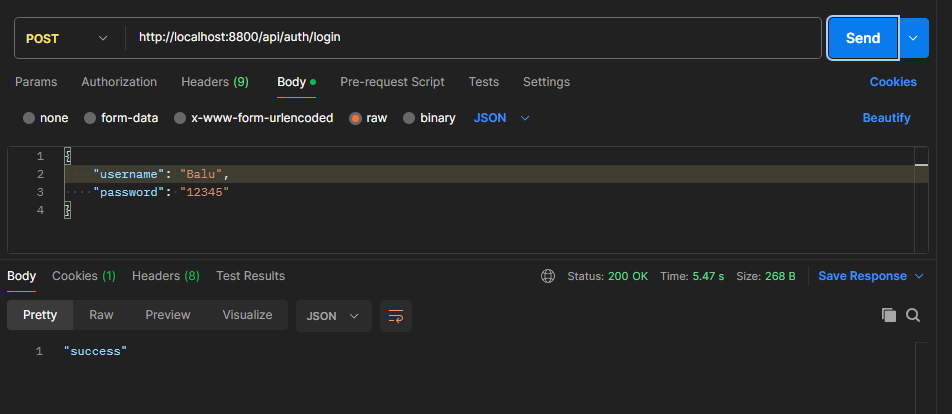
### **Login Route:**



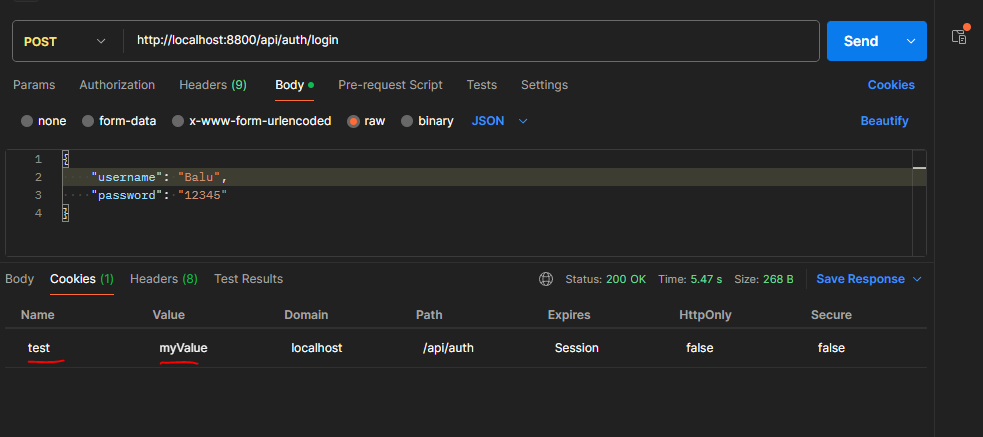


Error caused due to user instead of res



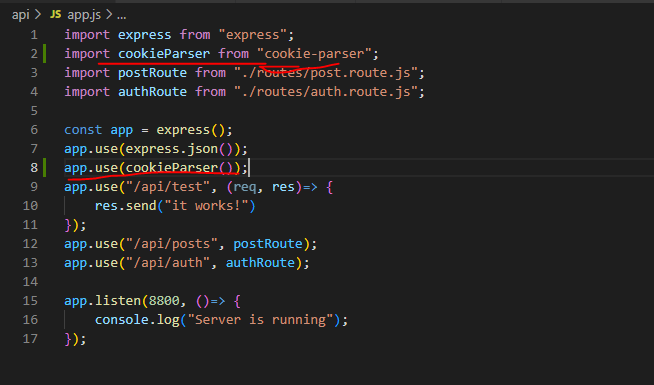


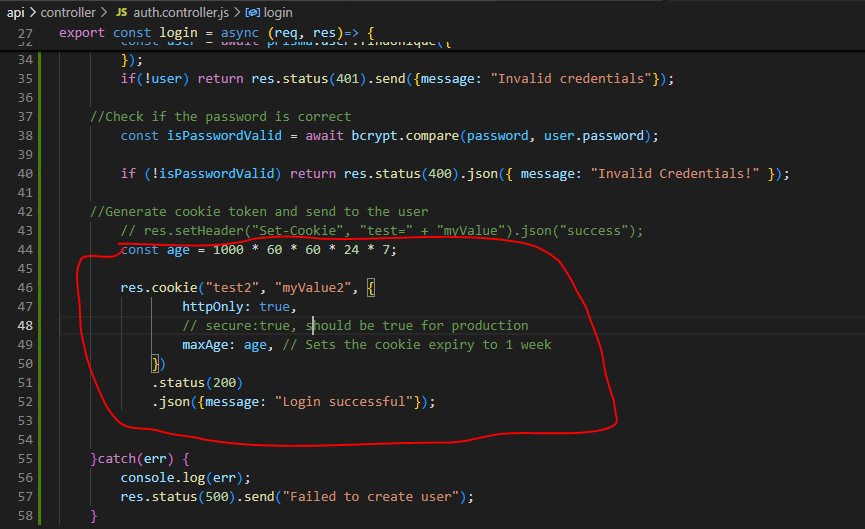
We receive the cookie as well

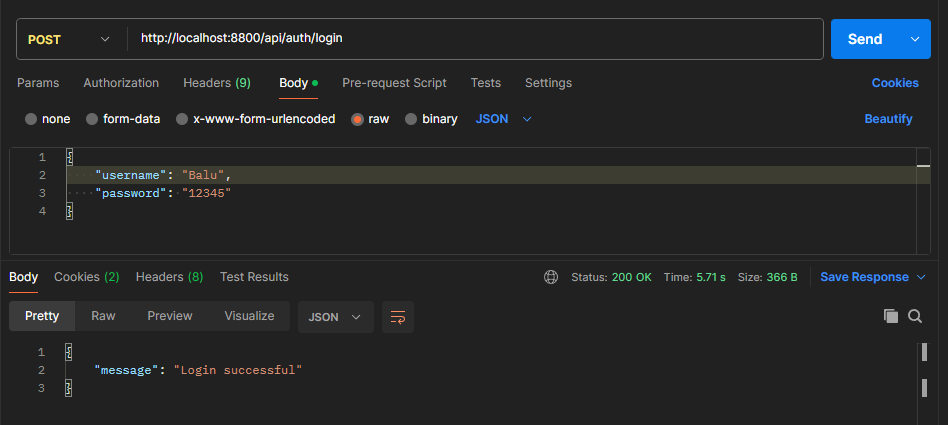


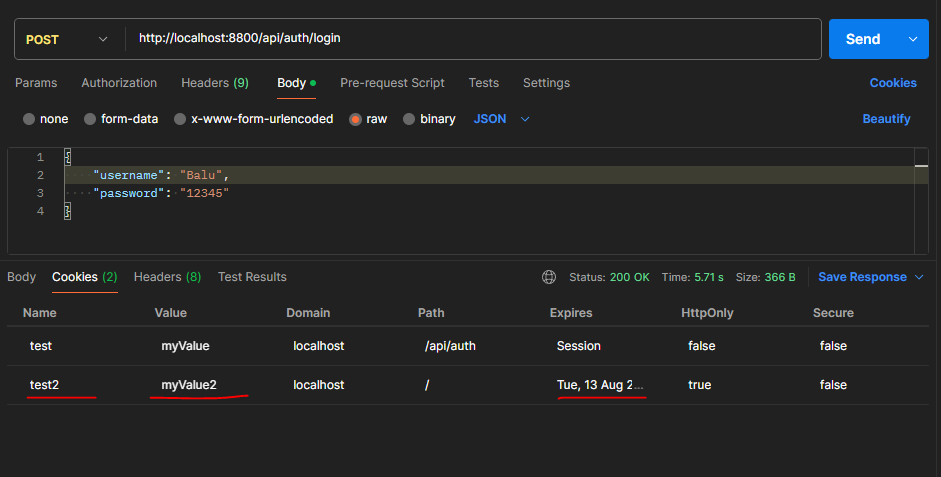
#### **Cookie Parser**

Npm I cookie-parser







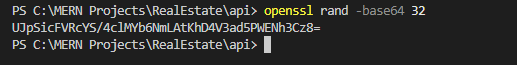


#### **JWT:**

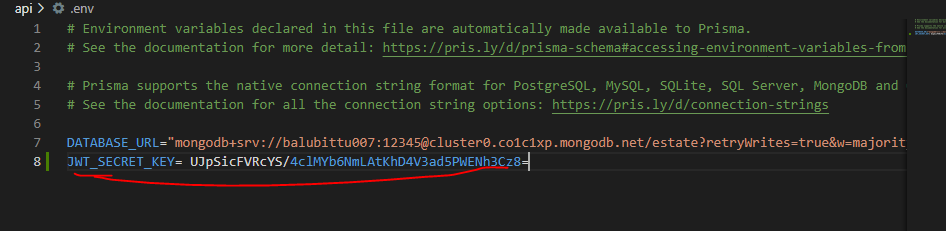
Npm I jsonwebtoken

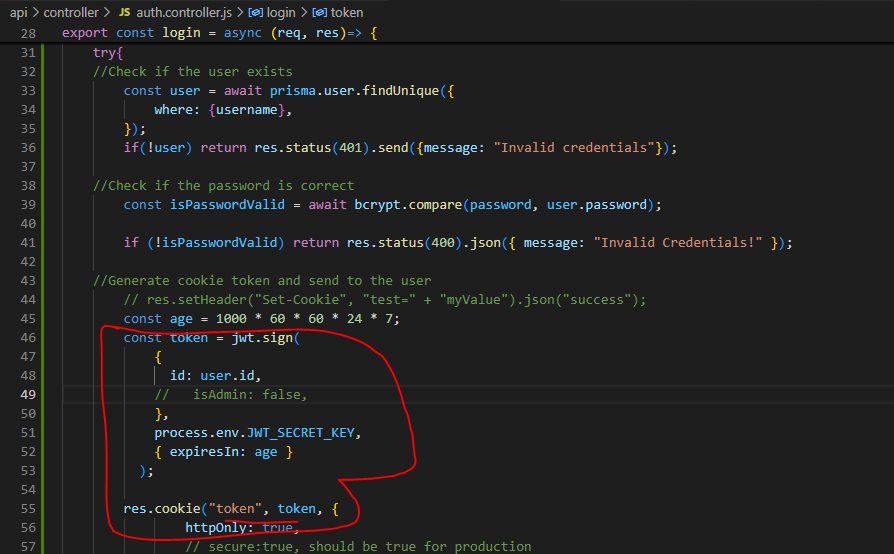
To generate random secret key run below

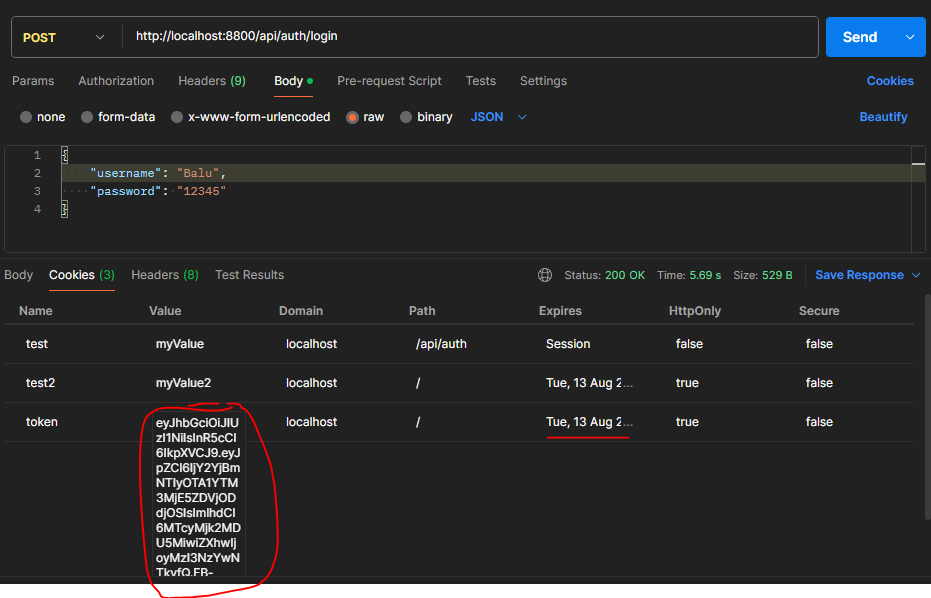
openssl rand -base64 32



Store it inside env file

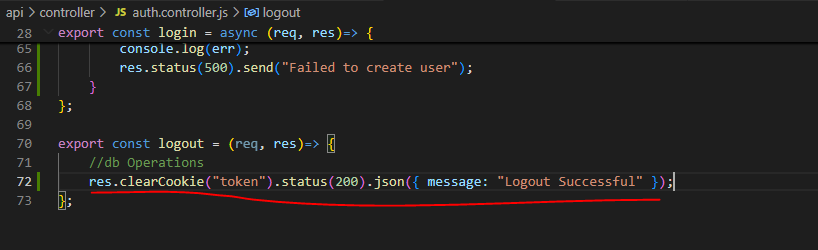


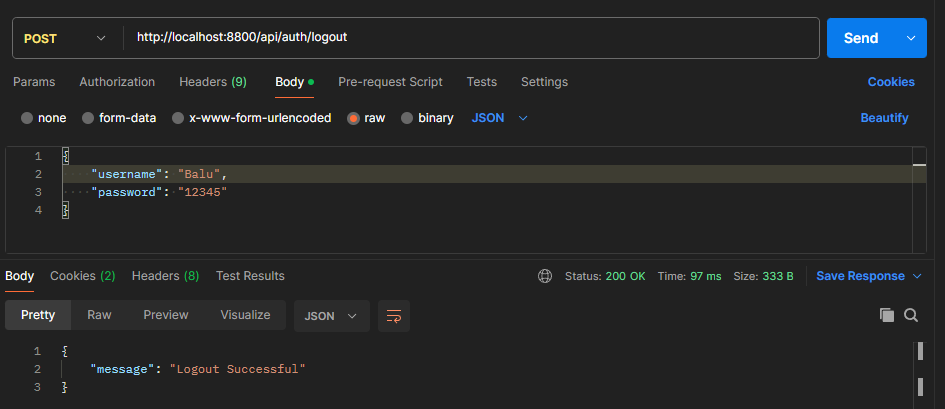




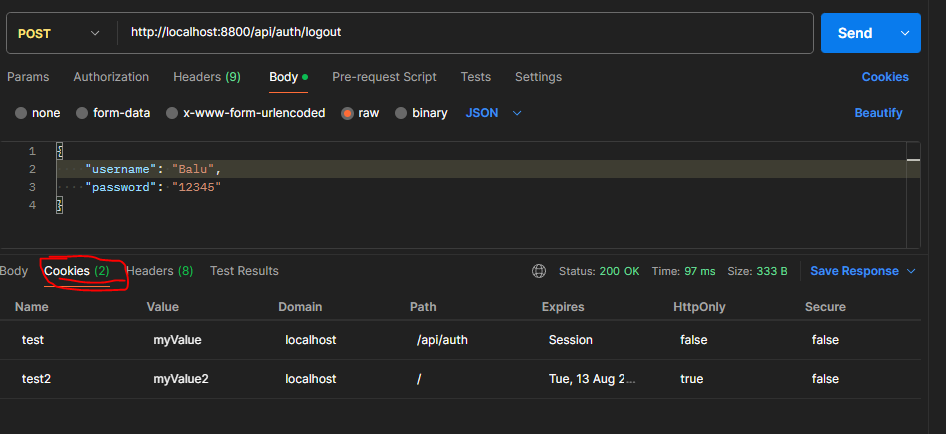
So, right now token consists of user id. On making different api calls we need to decrypt token and get the user id.

### **Logout:**

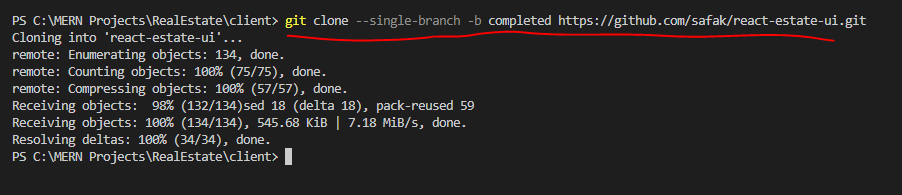


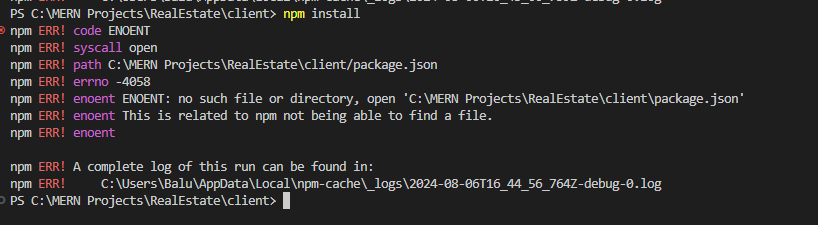


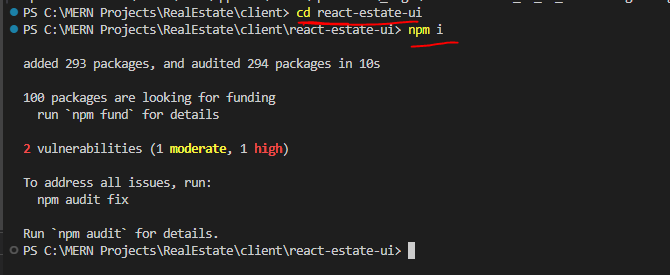
It will also clear token from cookie



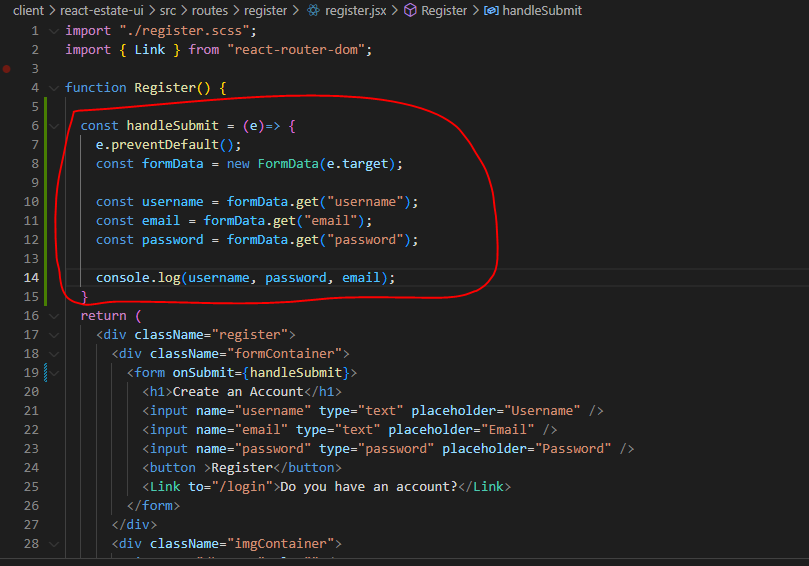
## **UI clone:**

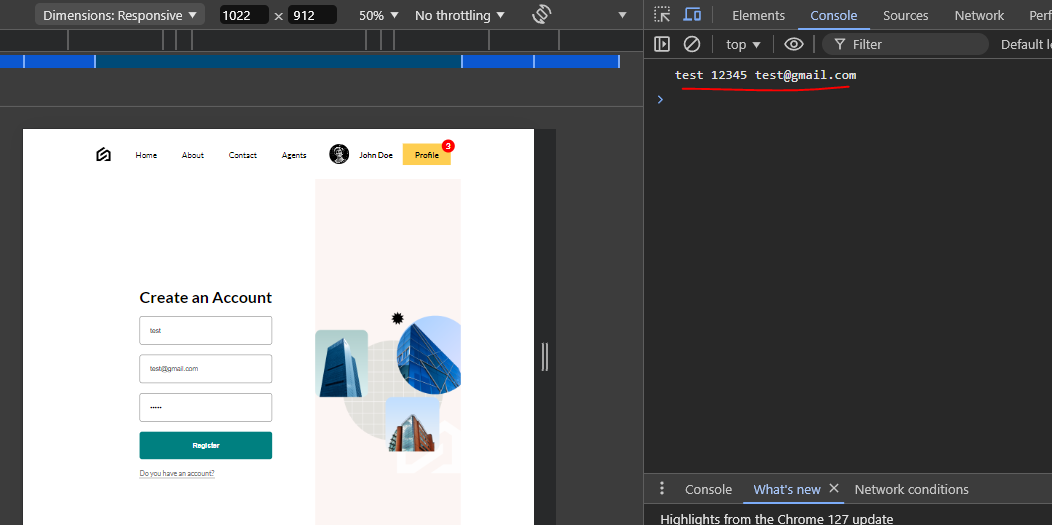




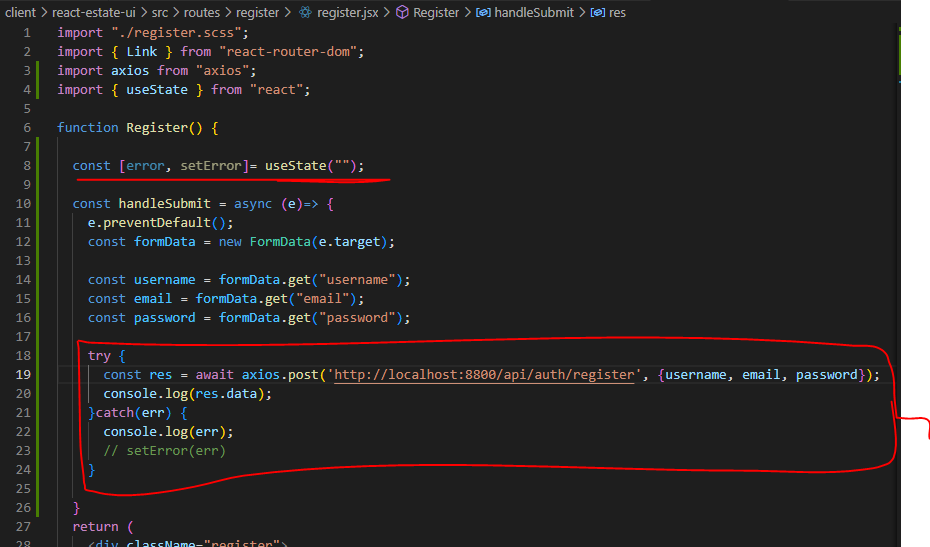


### **Integrating register with UI:**

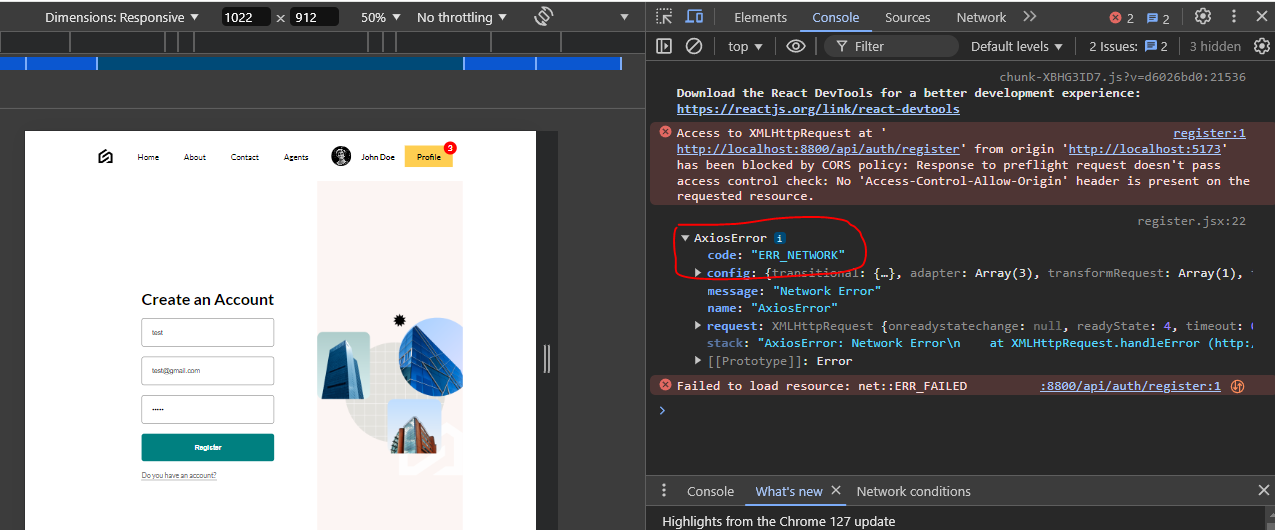




Install axios=> npm I axios

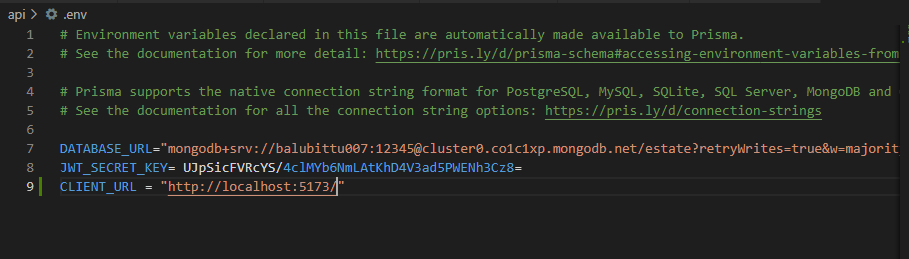


### **CORS Error:**

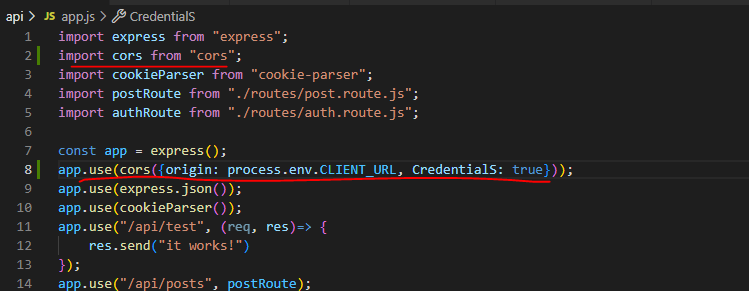


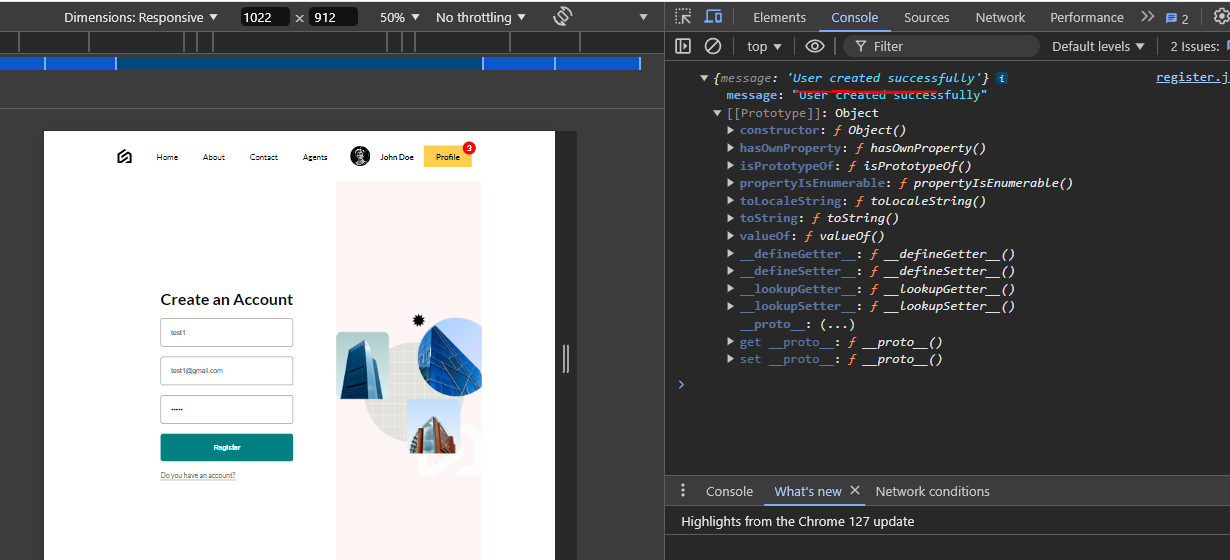
Npm I cors

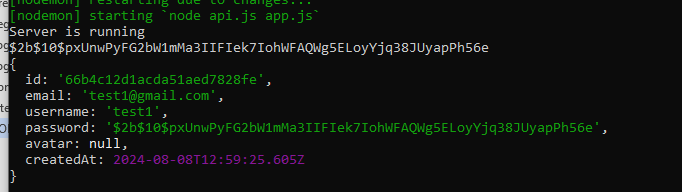
Client url should be updated after deploying the application.



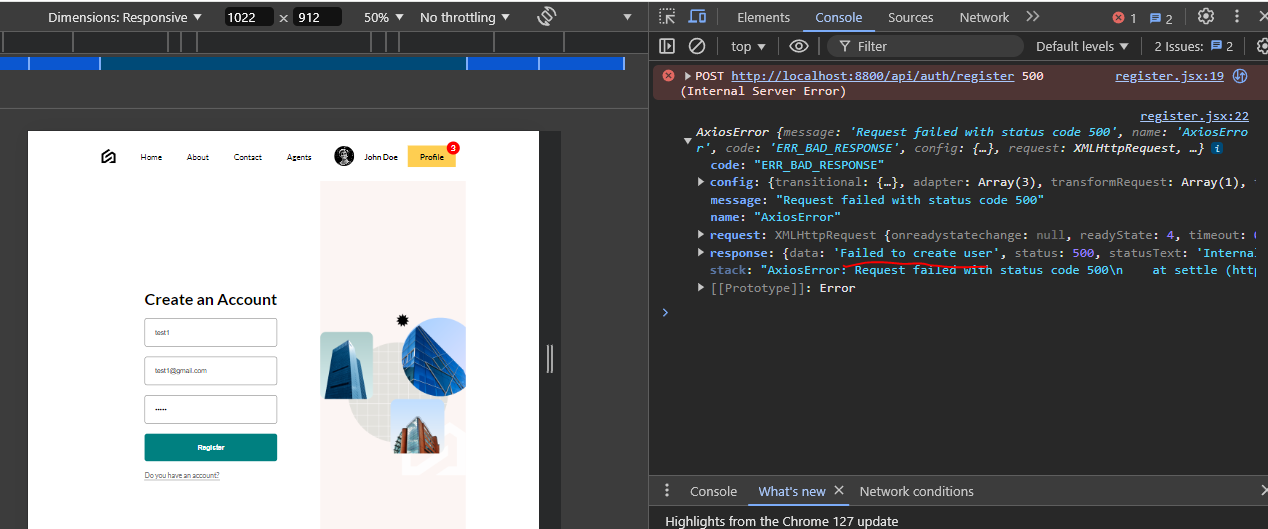
Using credentials: true will allows us to send cookies to the client.

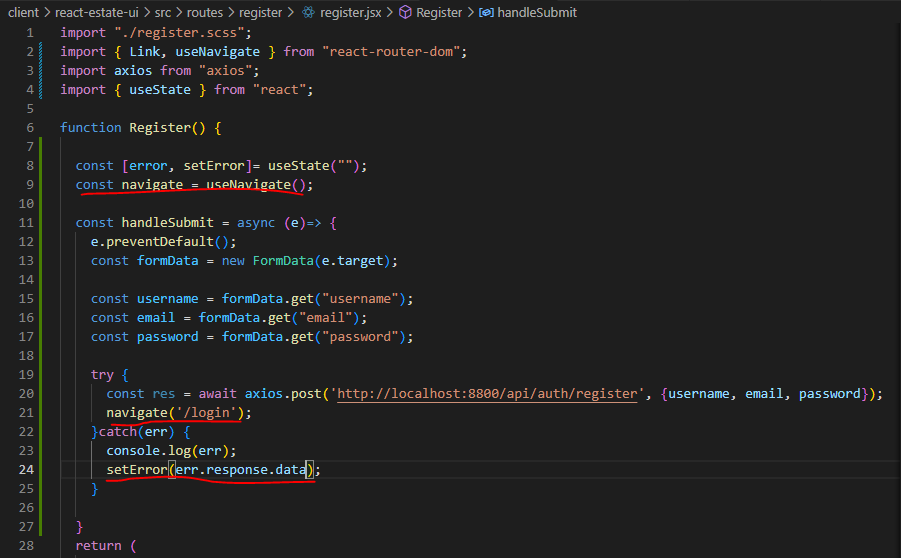


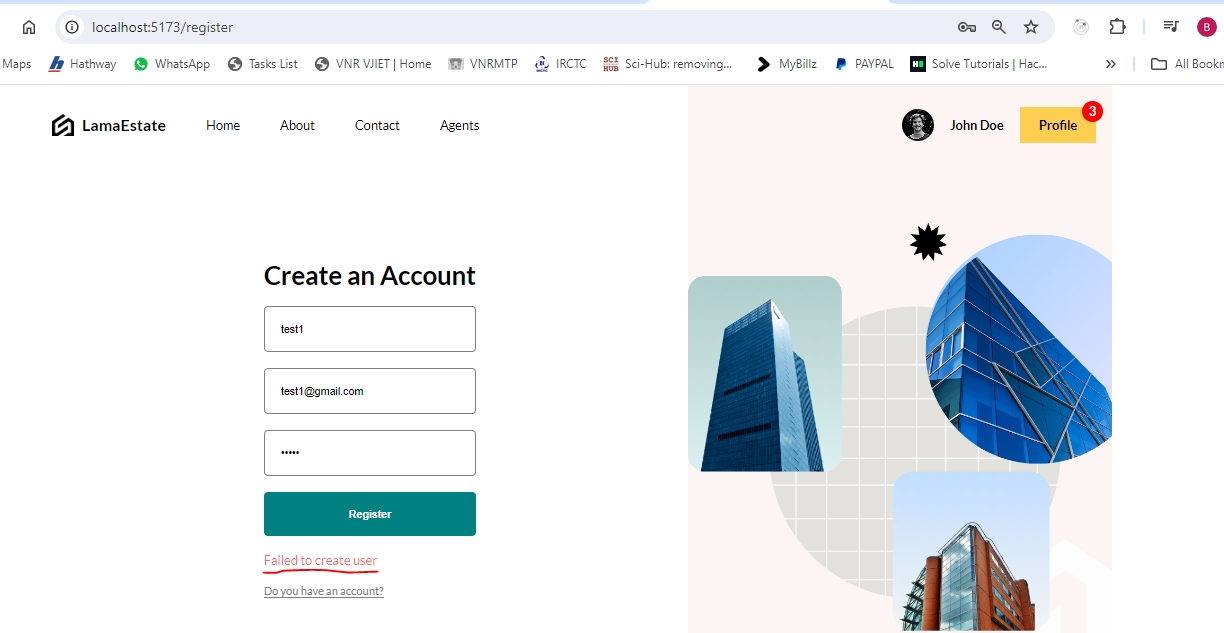




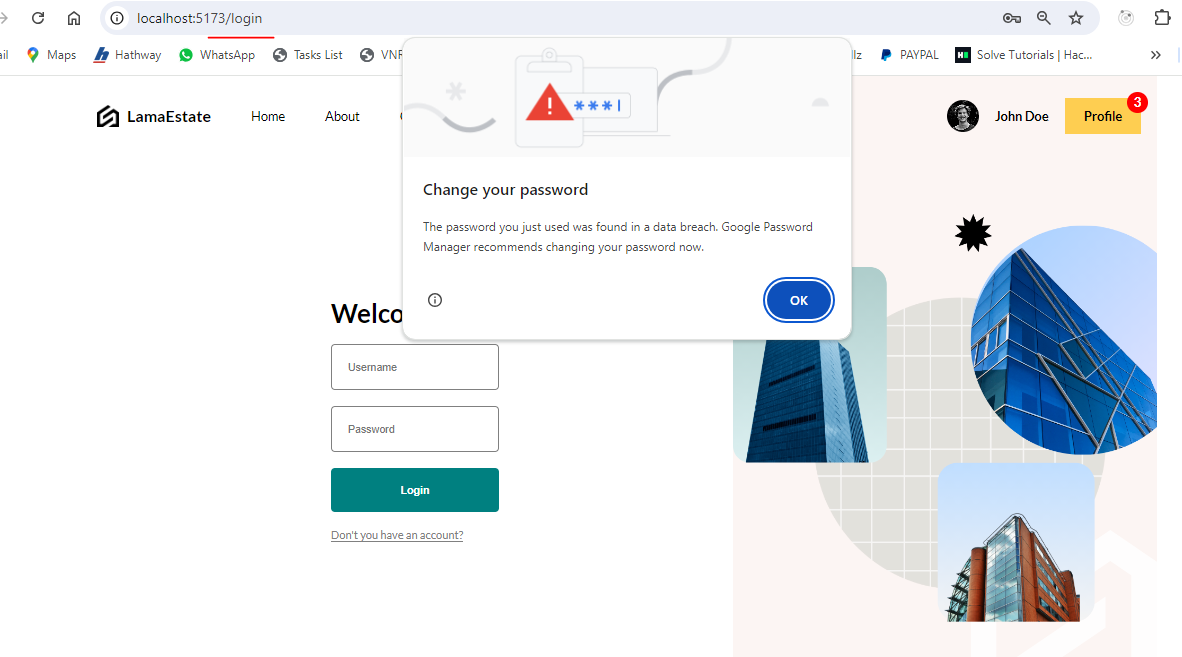
If we try to create the user again with same details. We get error





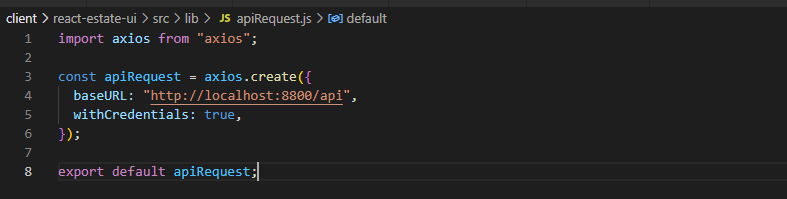


In case of unique details we get navigated to login page



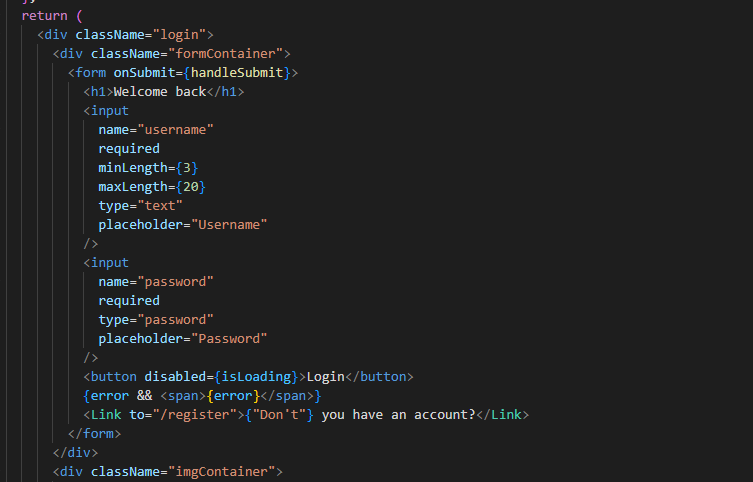
### **Storing user info in browser:**

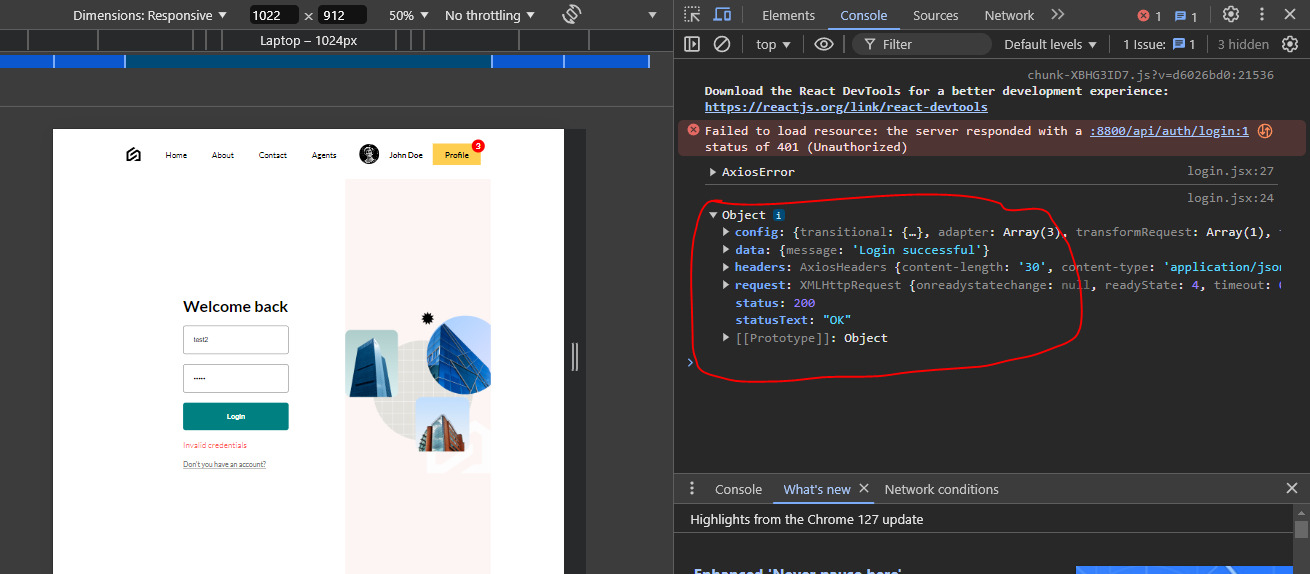
Create base url in a separate file



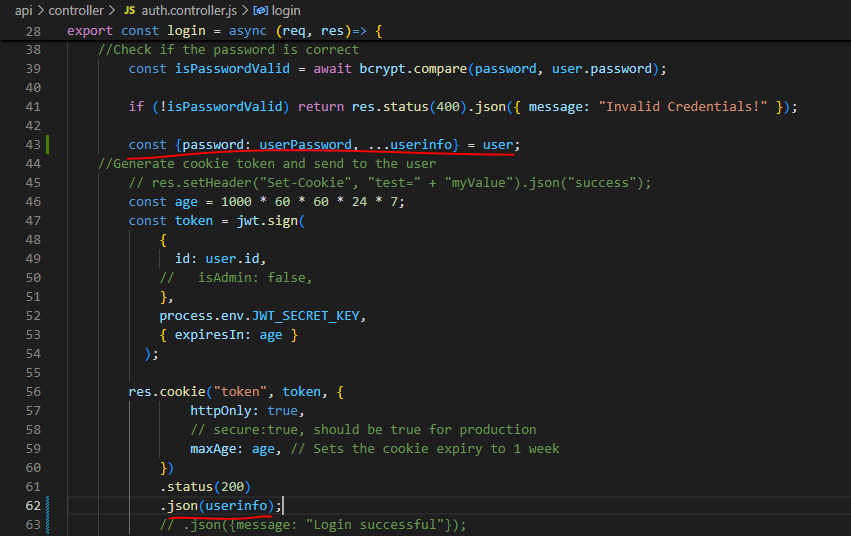


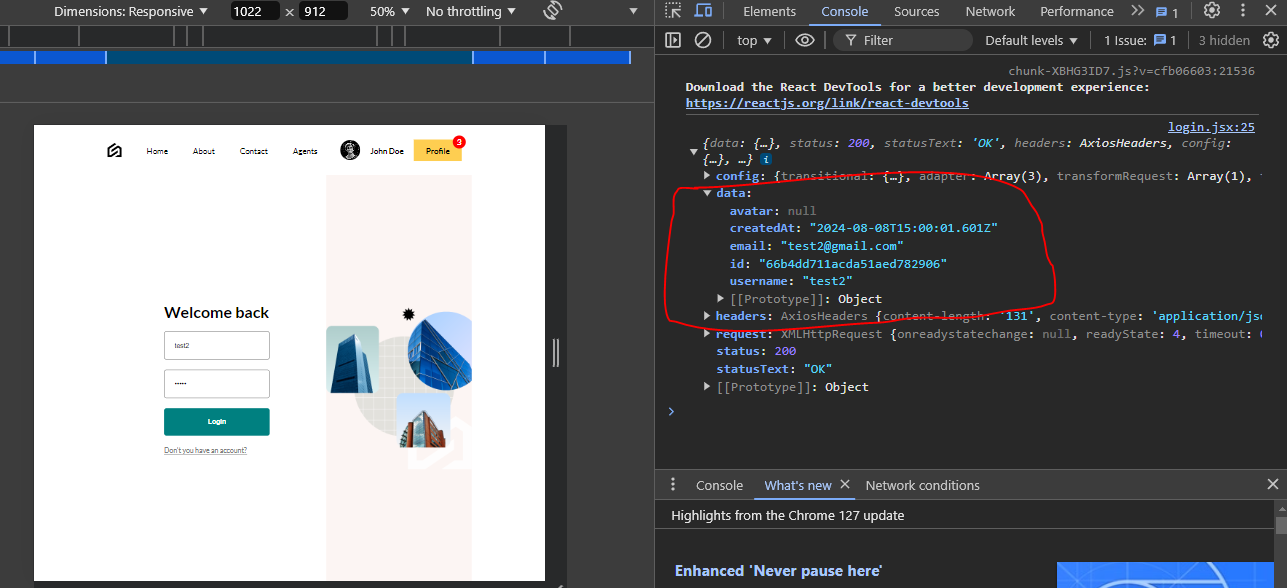




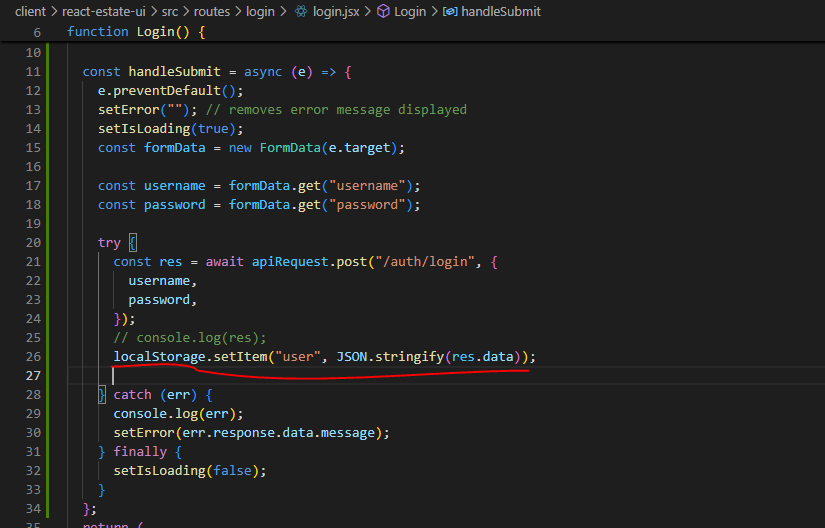


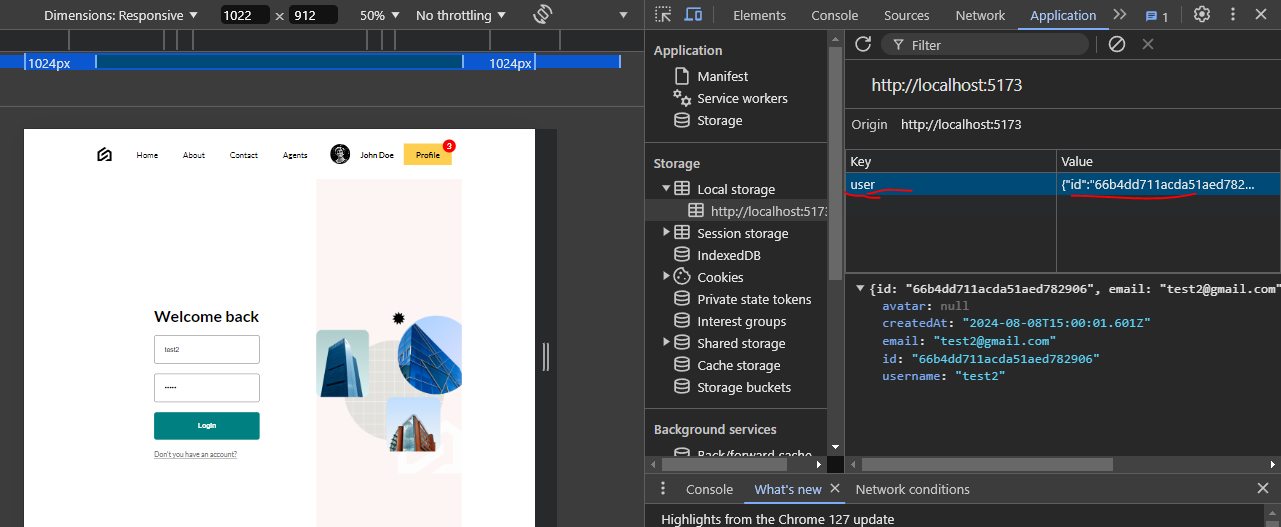
But instead of sending the message. We need to send the user info.



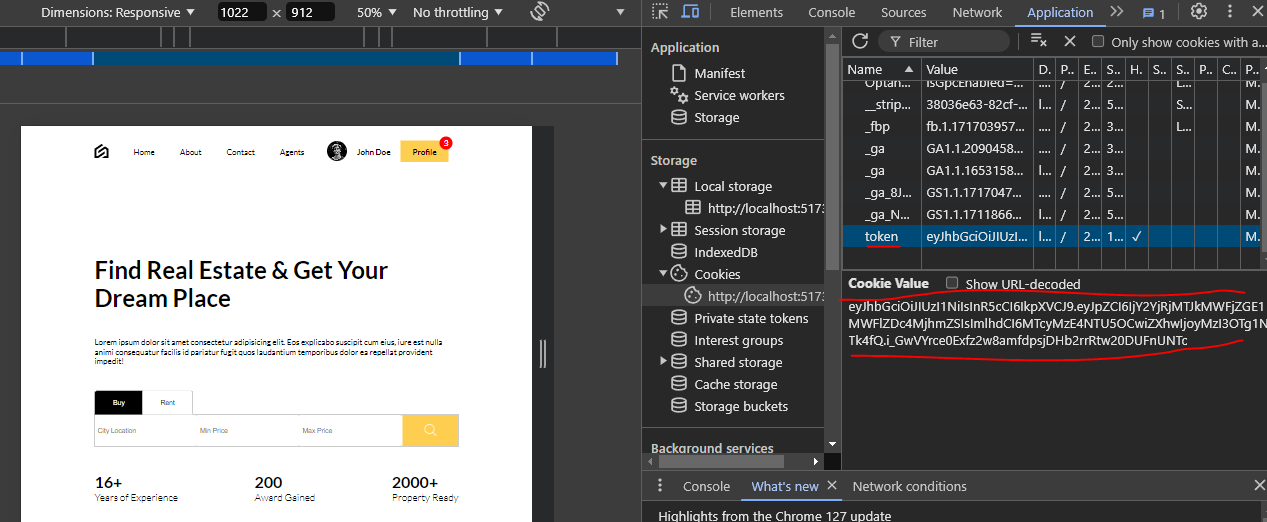


Also we save & use this user info in browser just to display the user name but to make any other api calls to create/delete posts we are gonna use cookies. This makes our application secure.

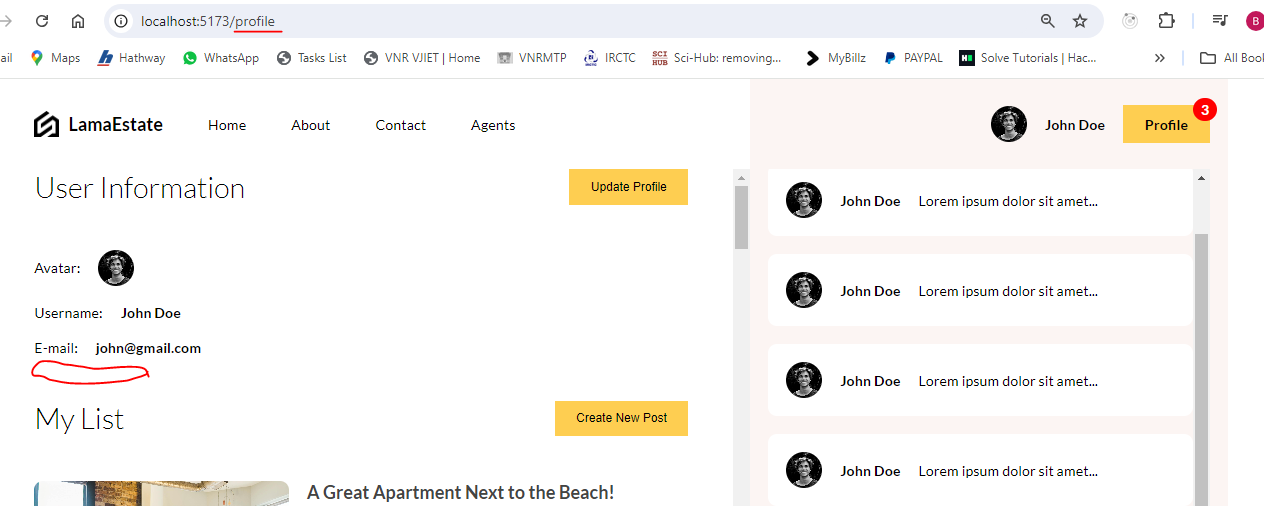


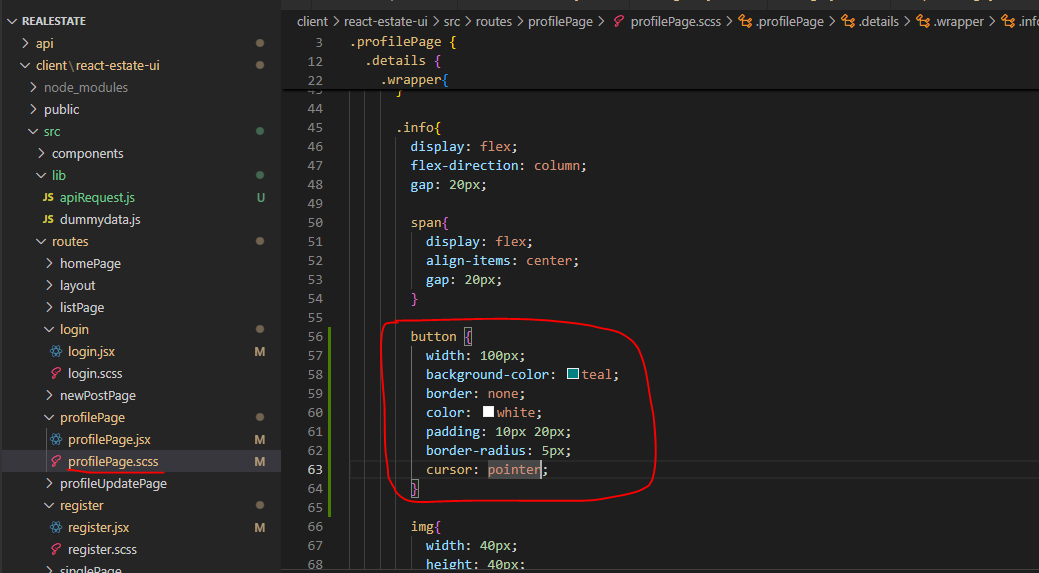


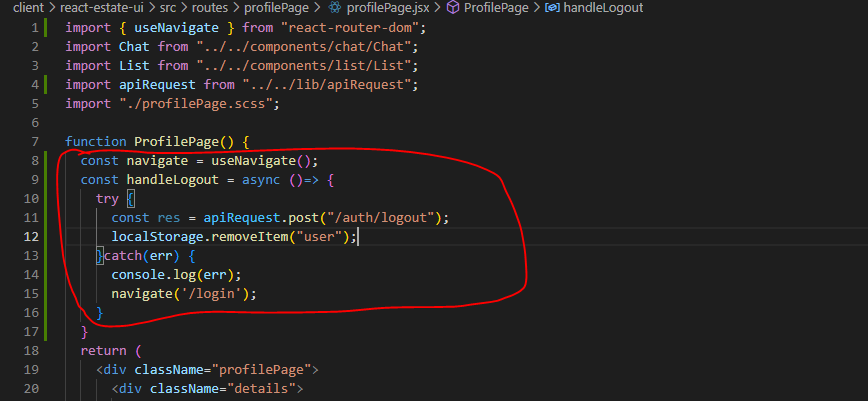
Also token gets stored in cookie



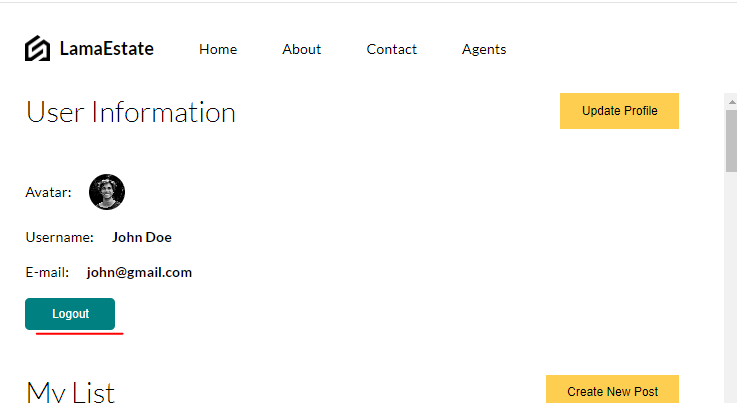
### **Creating logout button in profile page:**

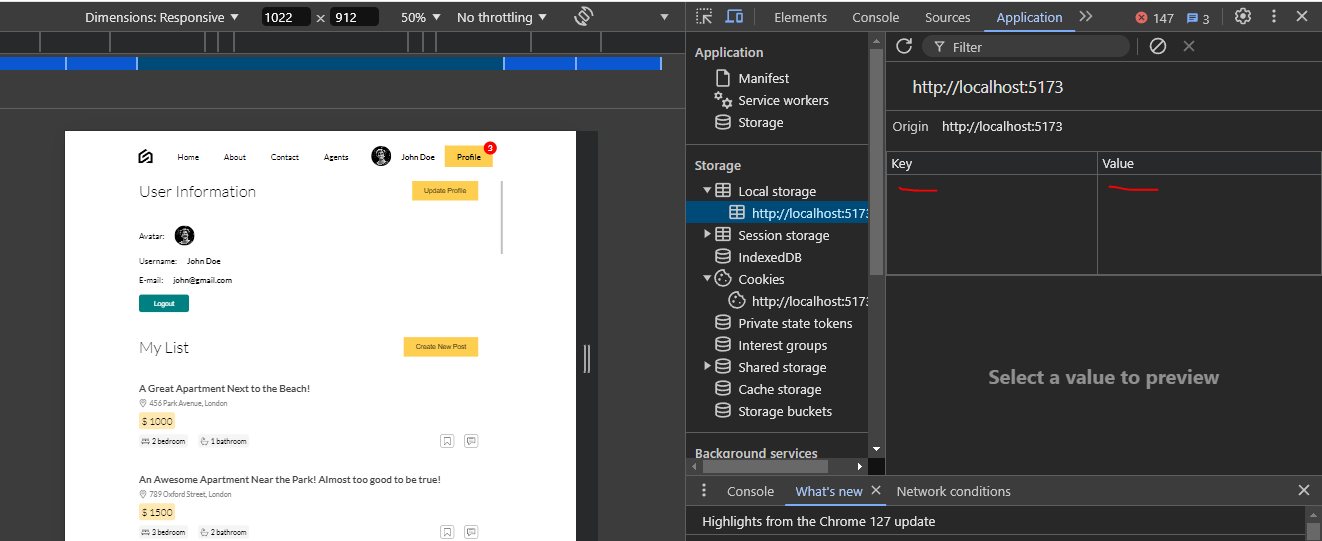


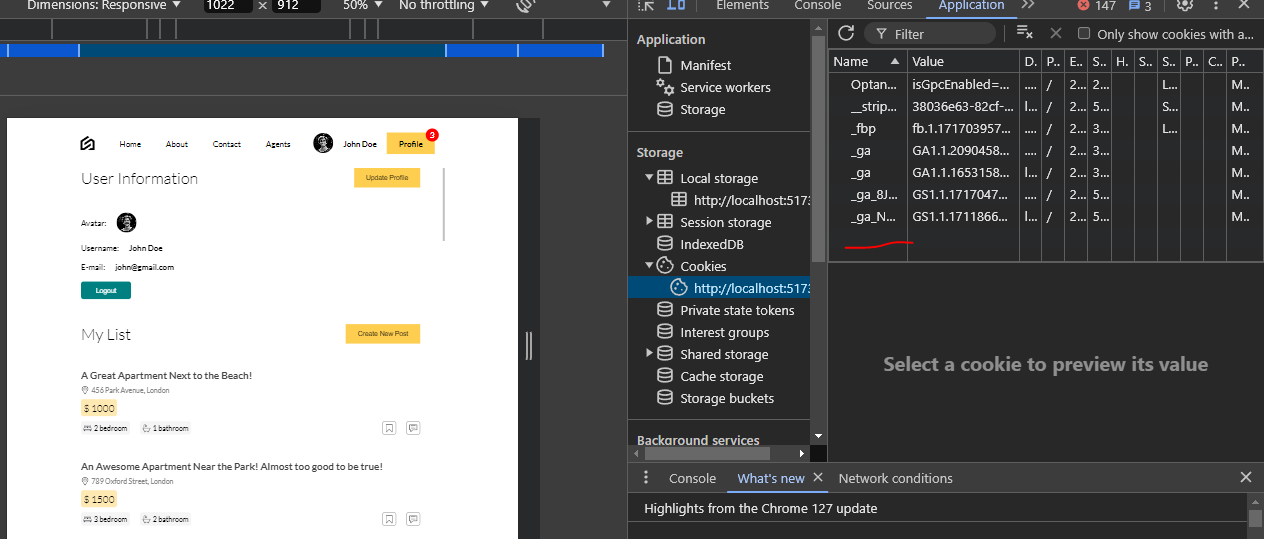






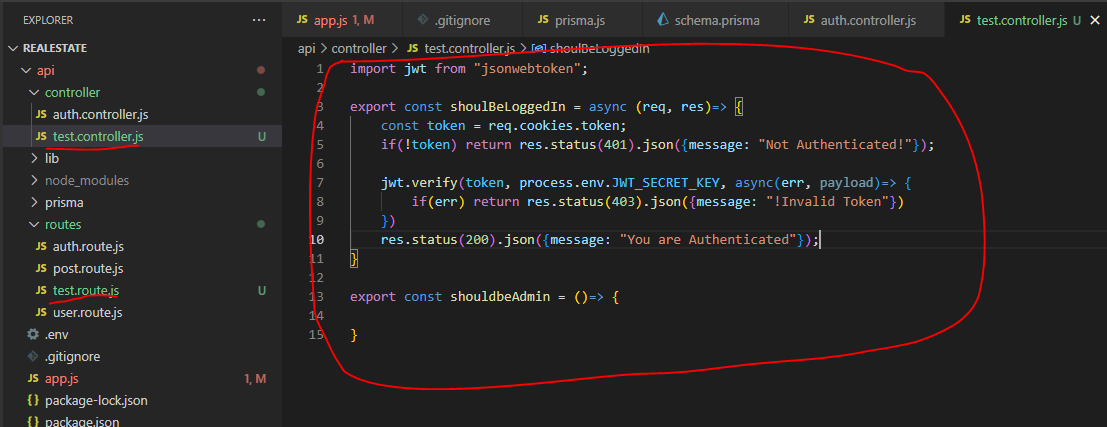


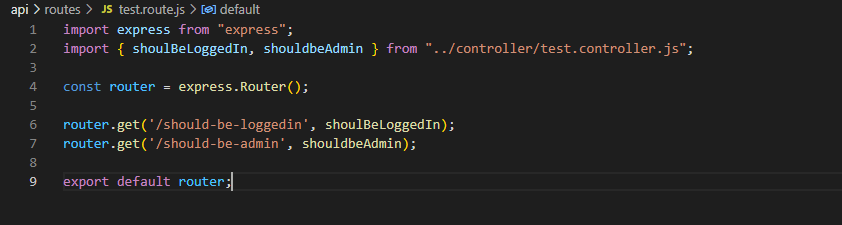
After clicking on logout button local storage is cleared also cookie got cleared



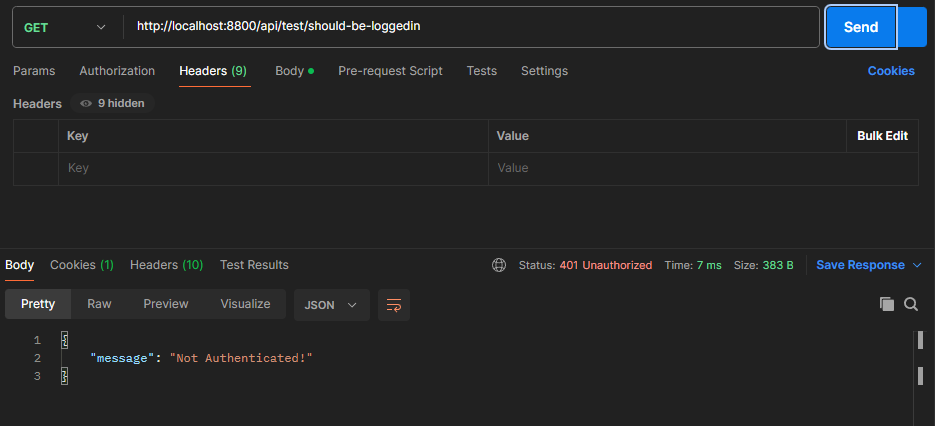
## **Authorization - Protect Routes in API:**

### **Protected test routes**

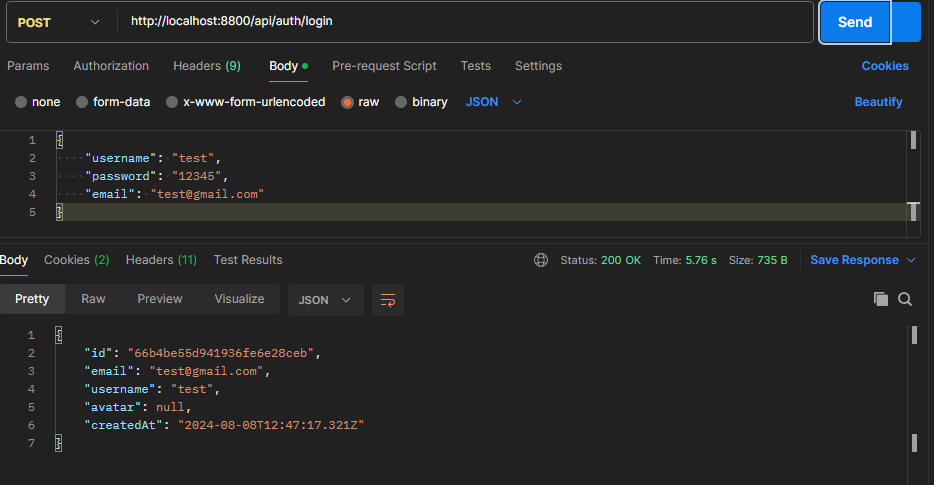


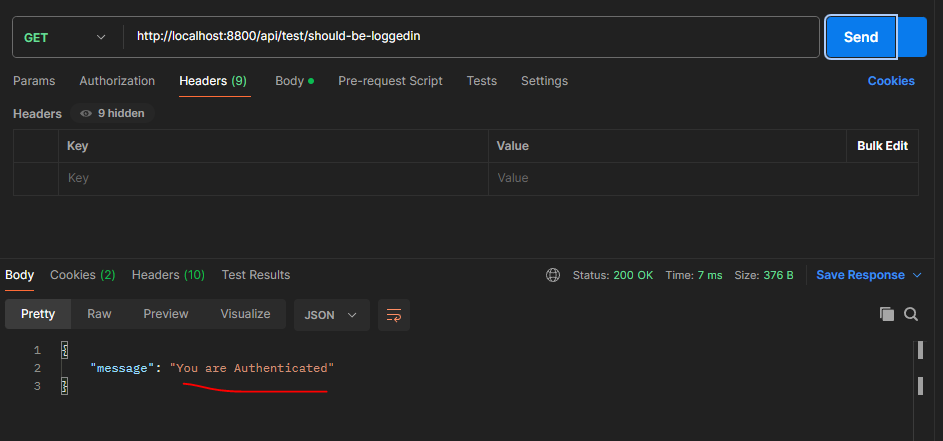


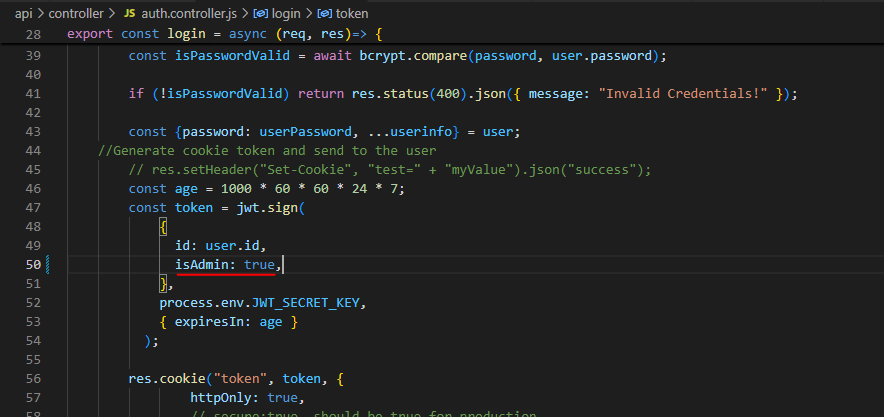
Without token we get below error

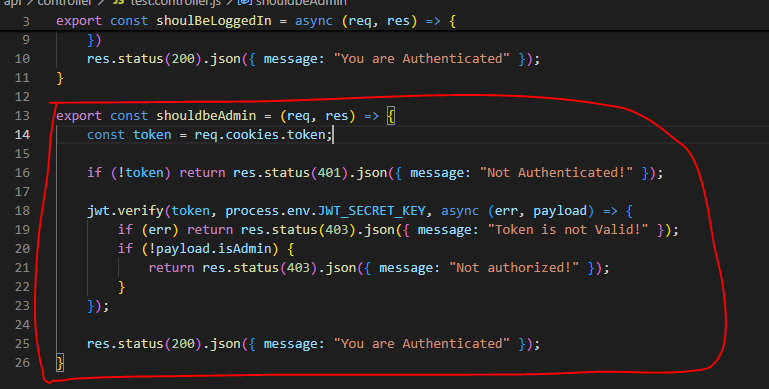


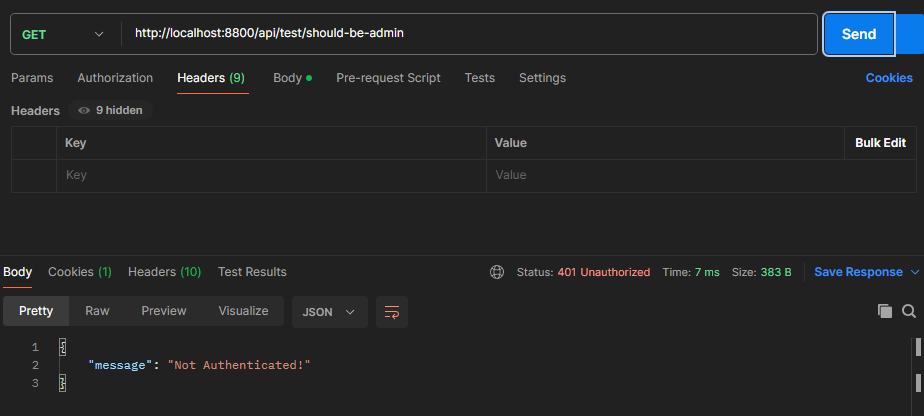
After making a login request we receive token



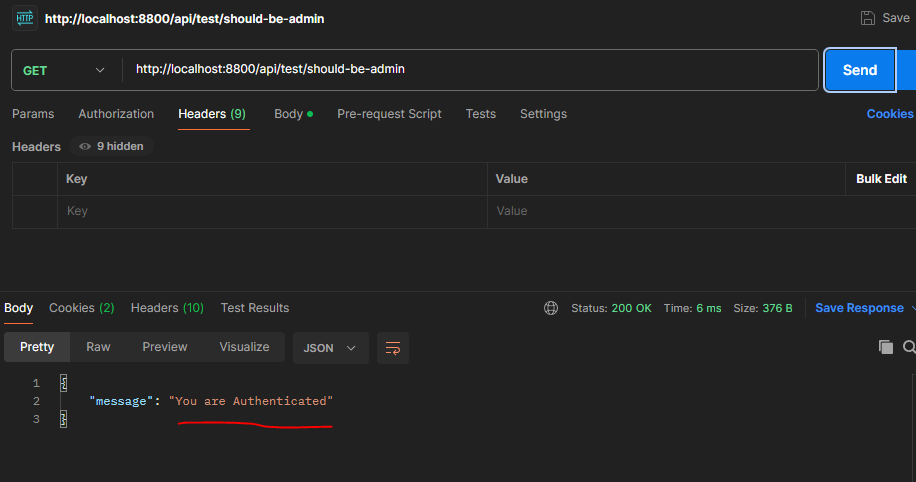






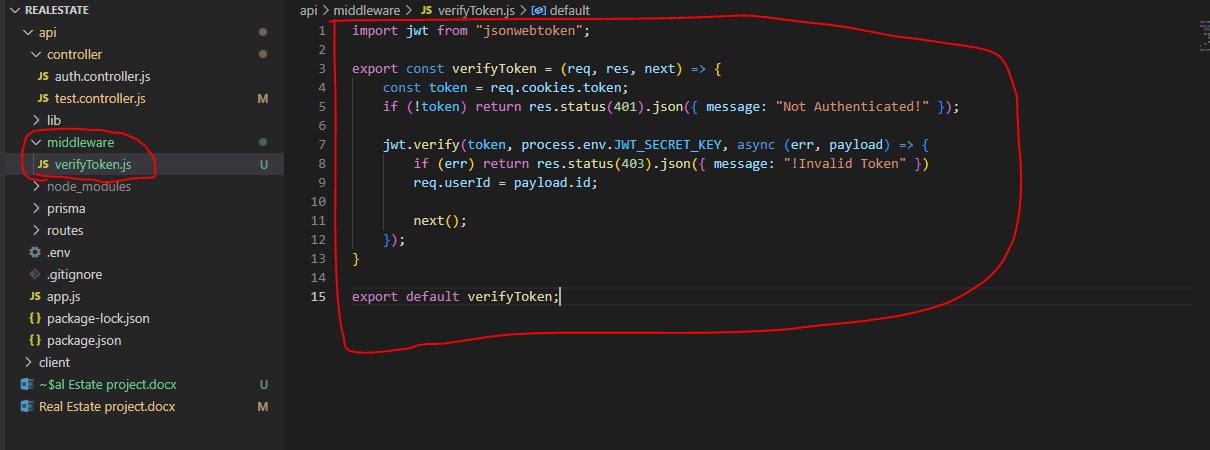


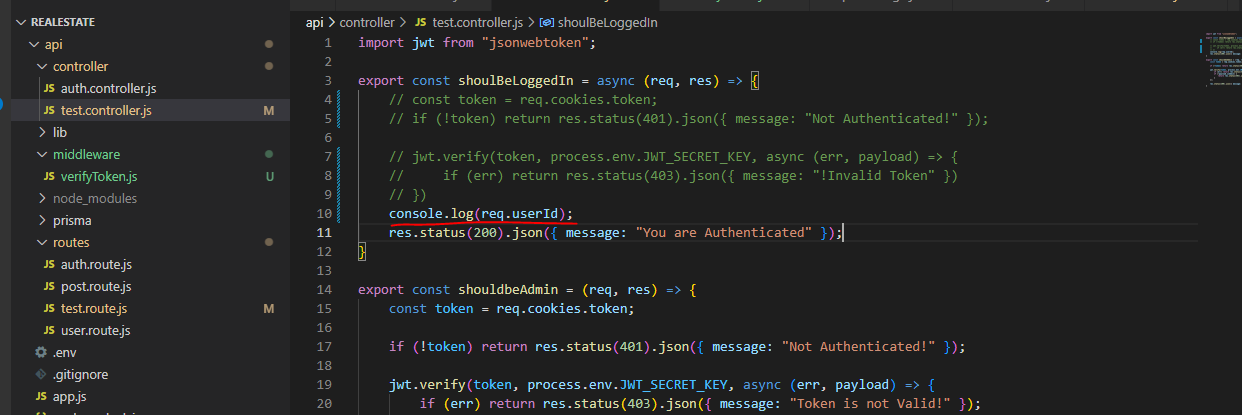
After login

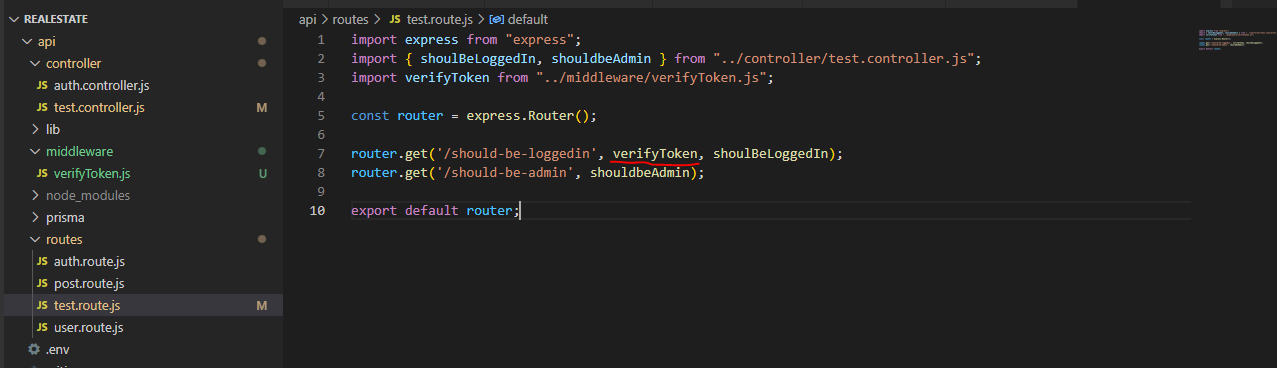


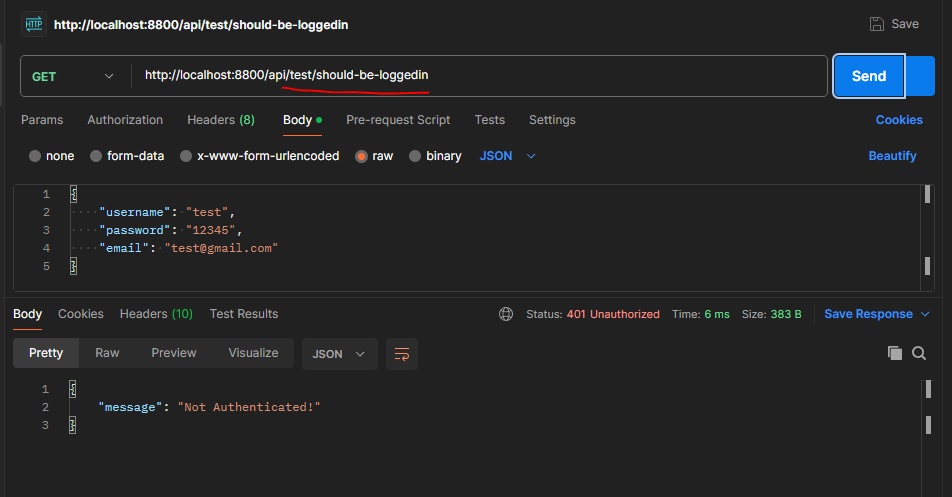
# **Express Js Middleware:**

In many requests we are gonna need this verification. We have to verify the token first for example when we want to create a new post or update our profile for the chat functionality. Basically we have to write the code again and again for authorization. To prevent this we can use express js middleware.





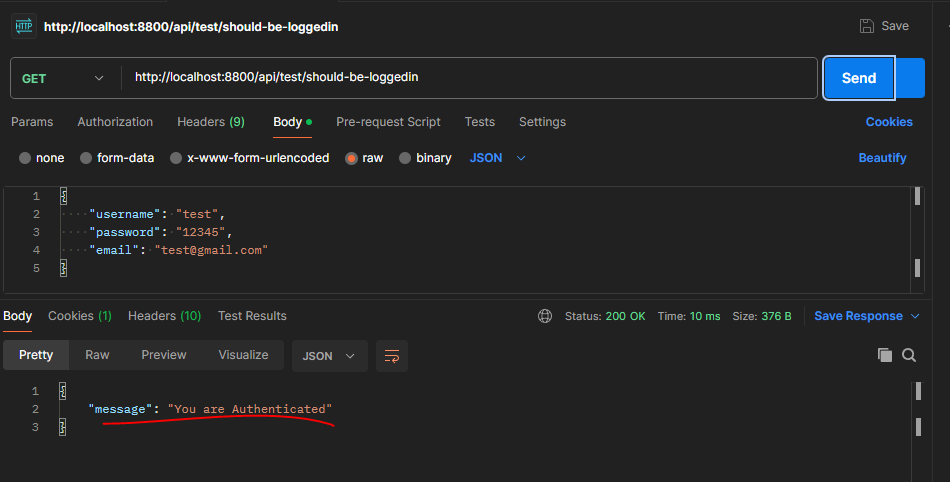




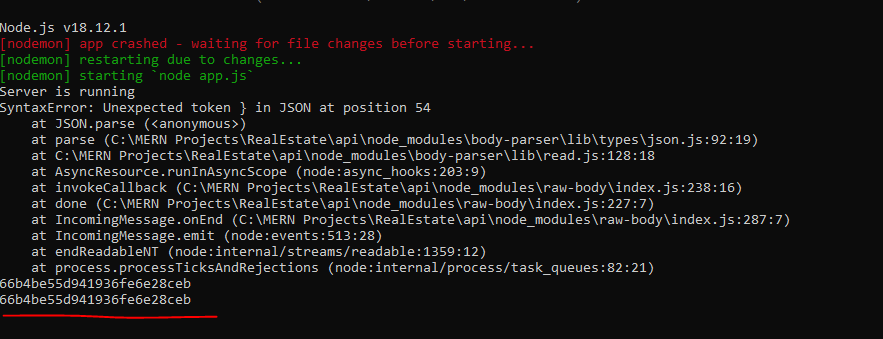
Getting logged in



Now



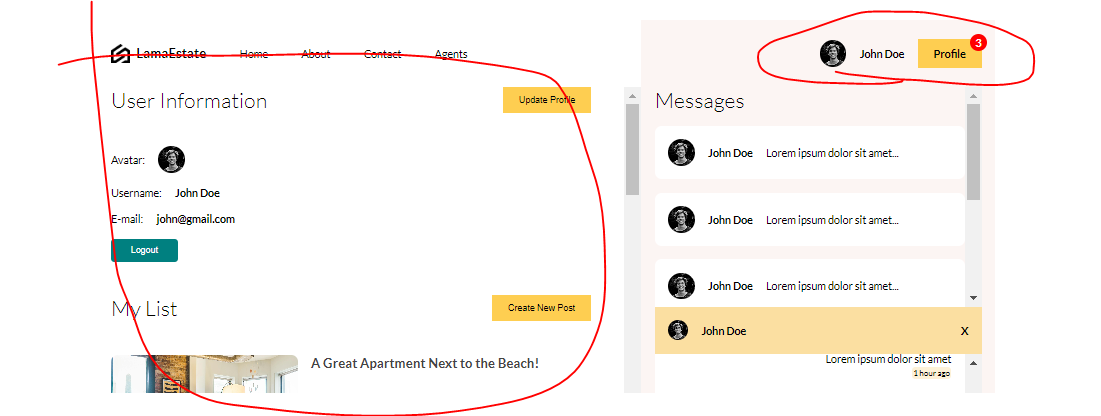
Our id got printed on logs

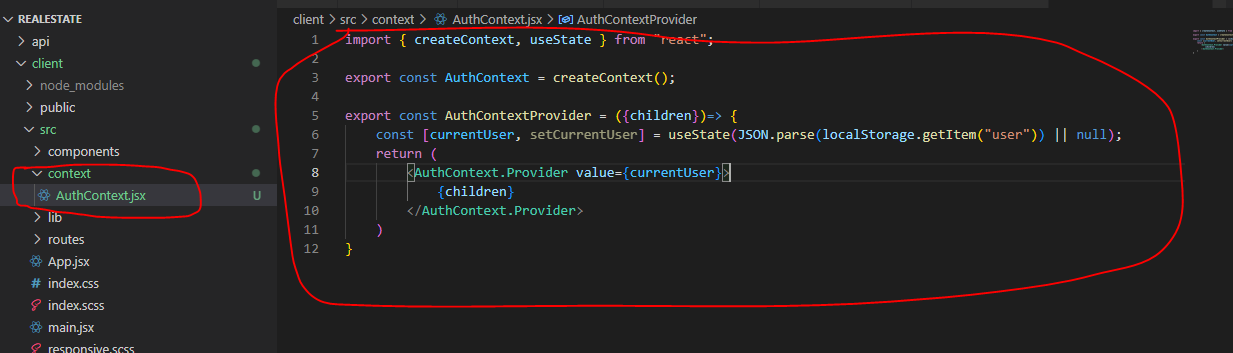


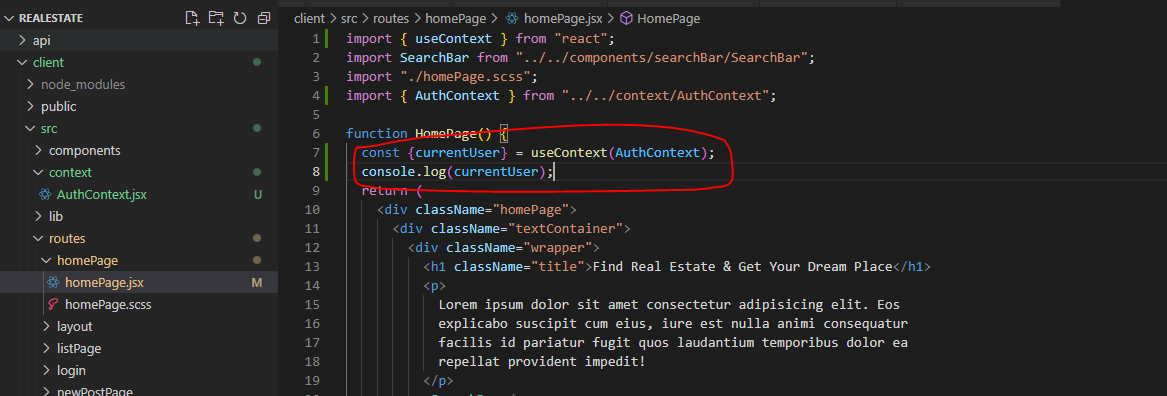
# **React Context API:**

Using this id we verify the user to delete any posts.

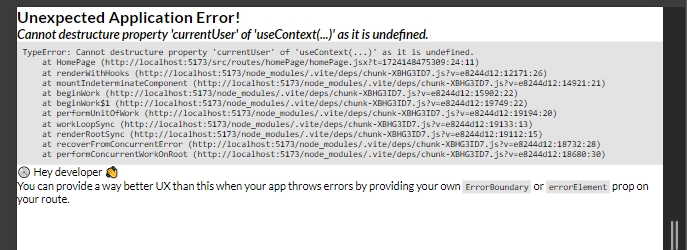
So, when we are not logged in we should not see the profile page. We need the user information every where. So we can store user info in react context api. Whenever it is needed we can call useContext hook and we can reach that data. Also we can create the function to update the profile.

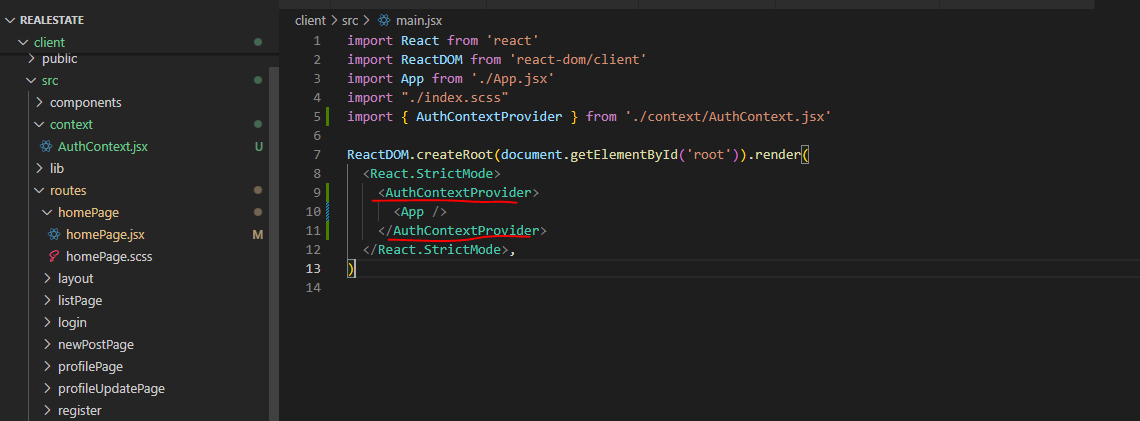




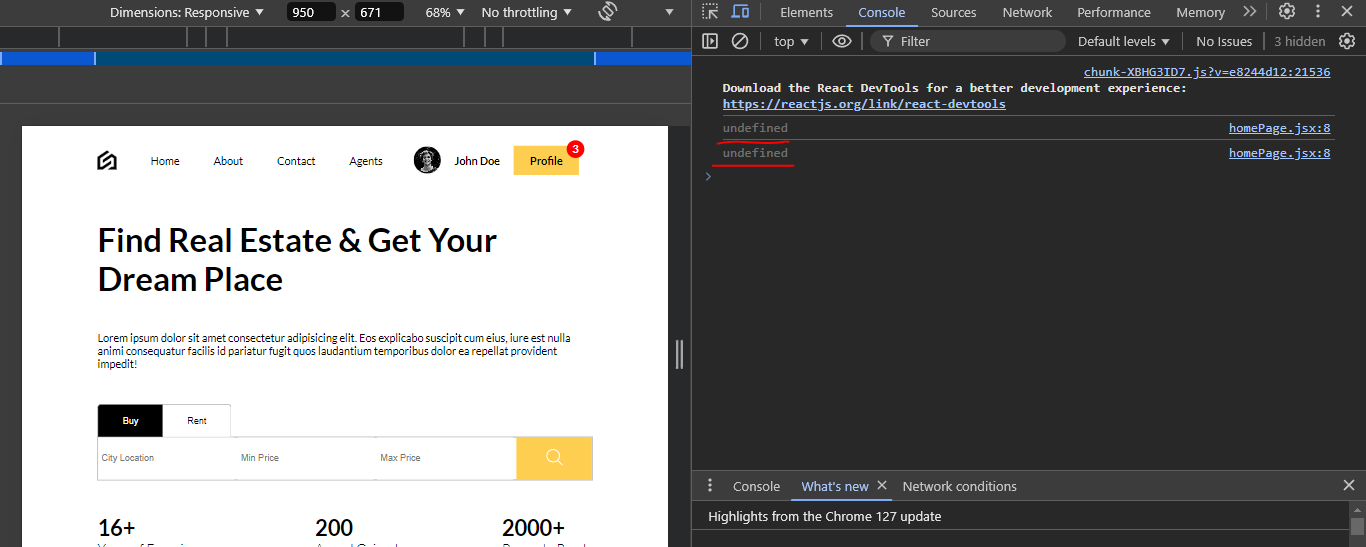


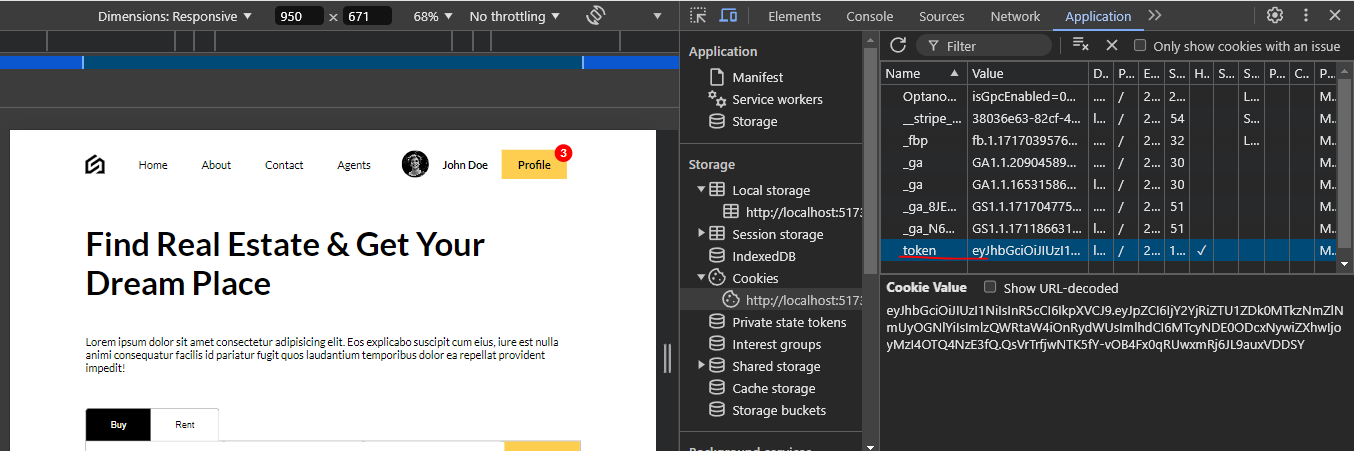
We get the below error due to we didn’t wrapped the application in provider yet.

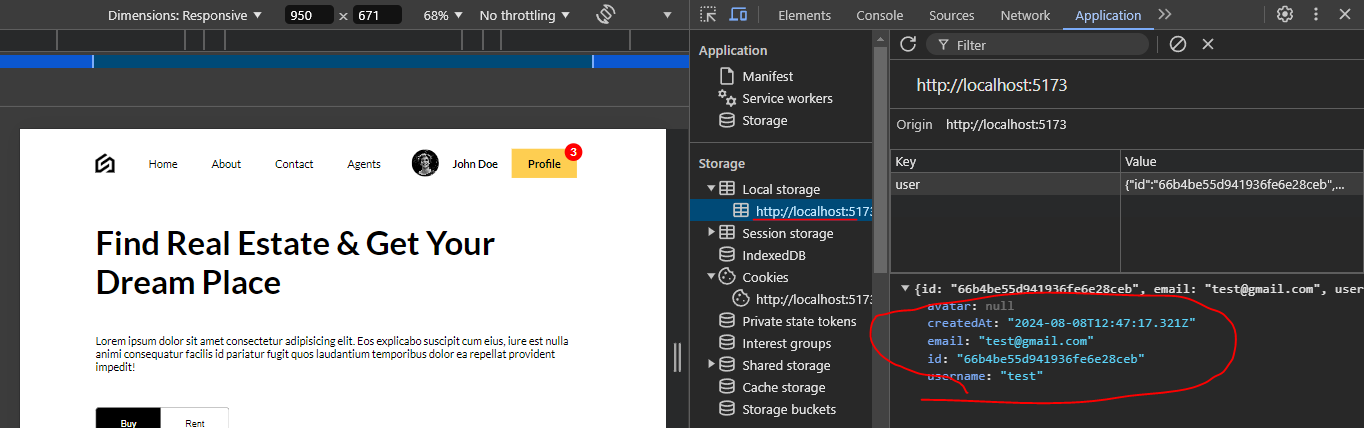


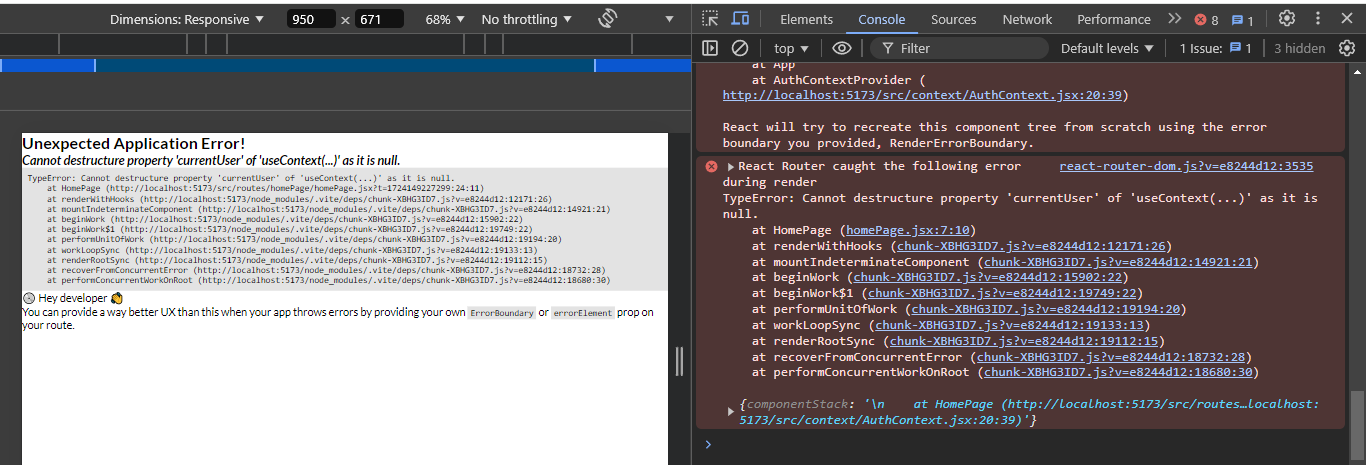


Still undefined is getting printed eventhough we got the user details stored in local storage.

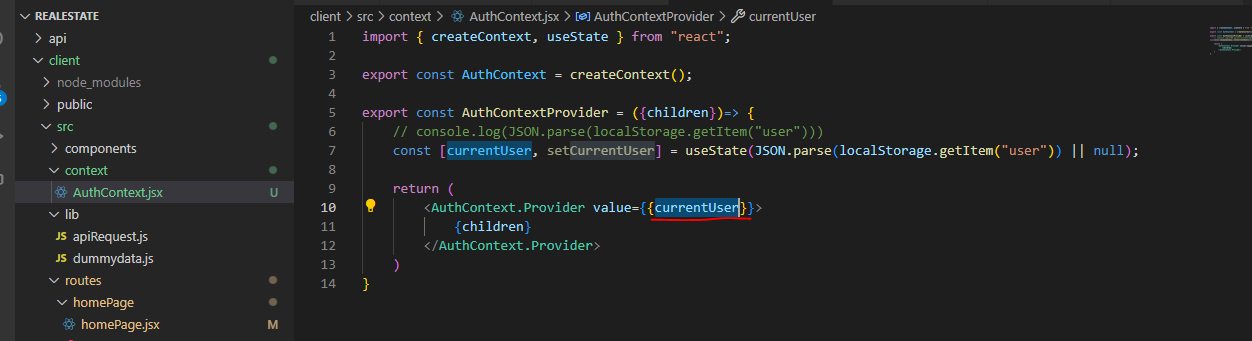


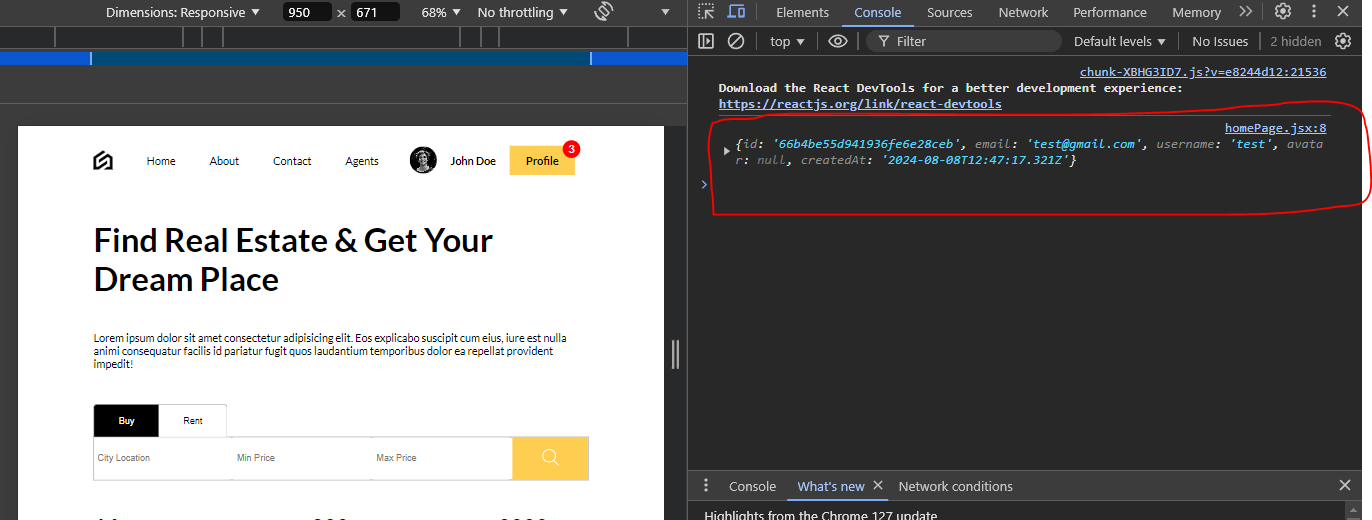




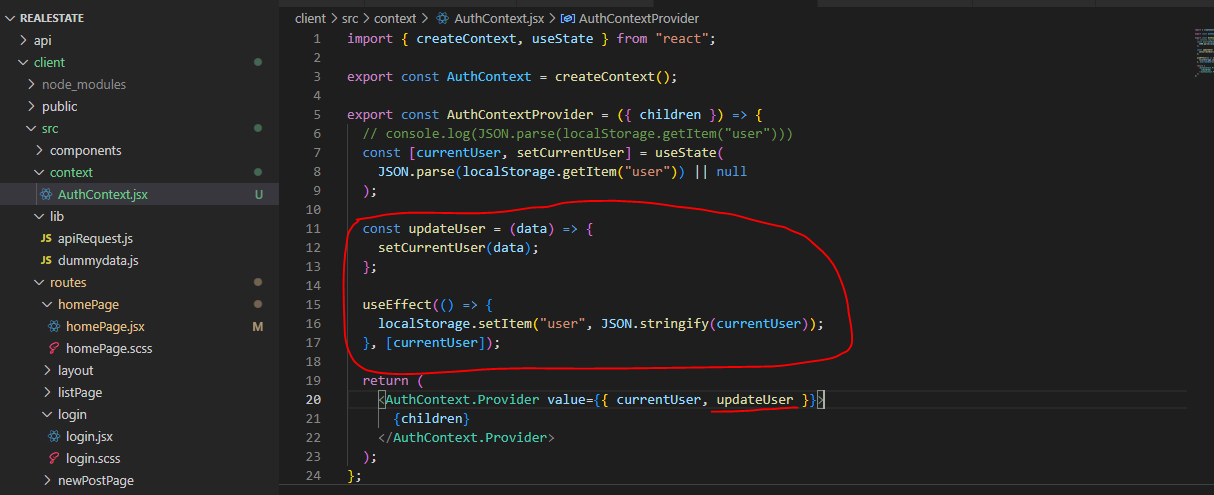


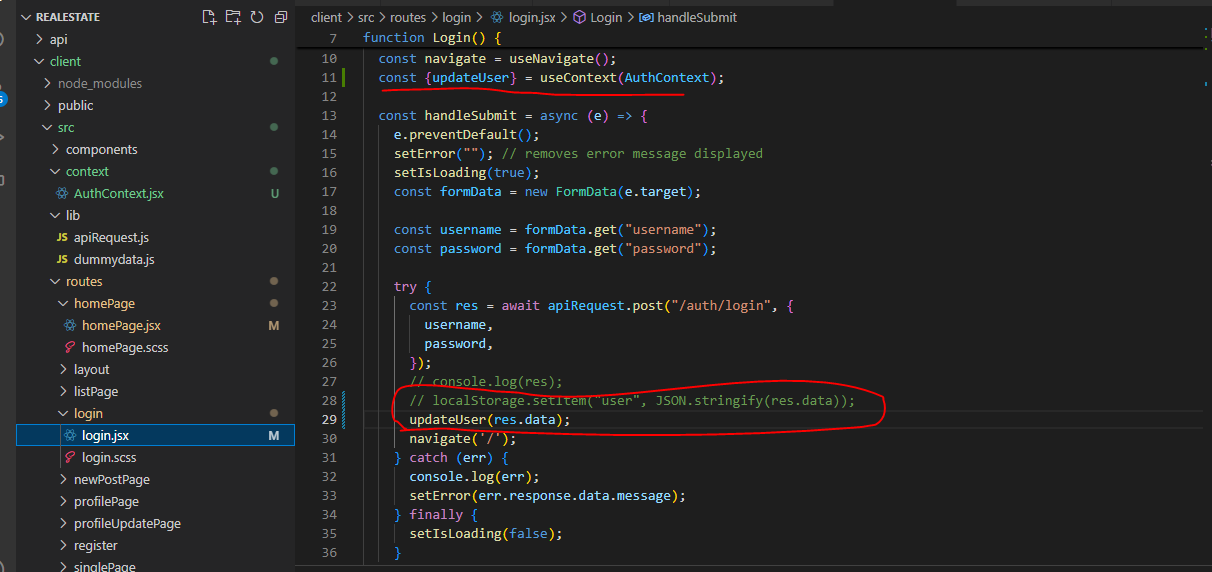
The error was occurred due to currentUser was not wrapped with curly braces



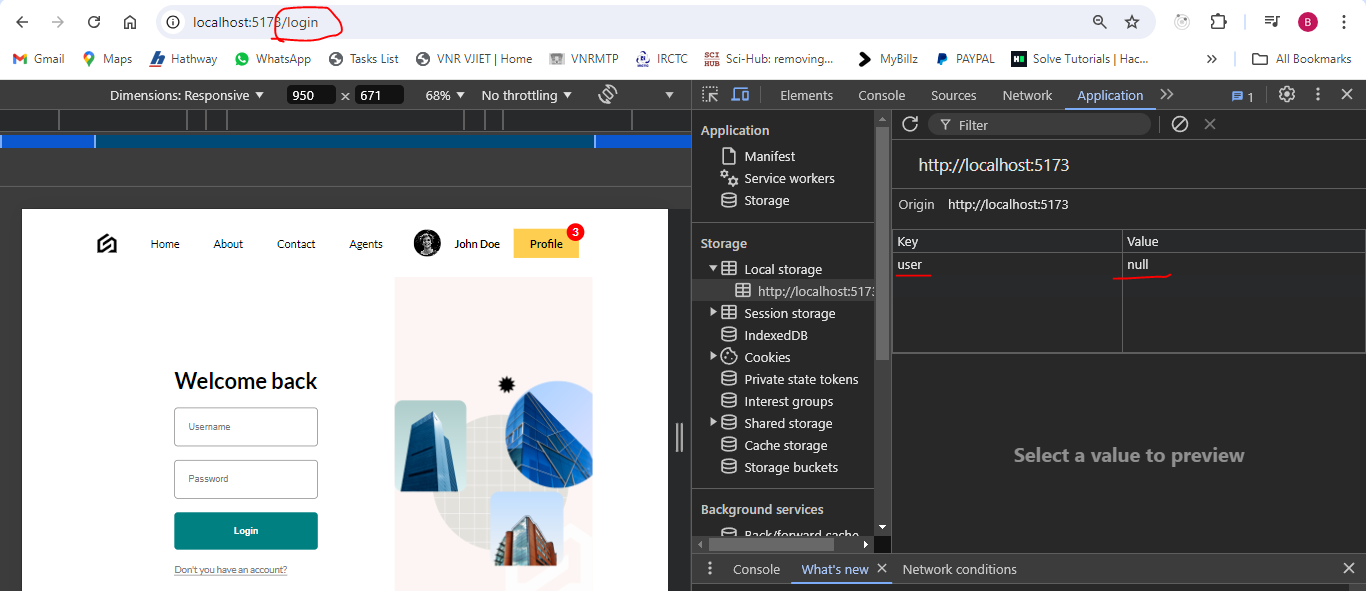


Now we got the user details but we shouldn’t refresh the page every time when we make any change. Instead of changing our local storage in the login page we can handle that in the context.

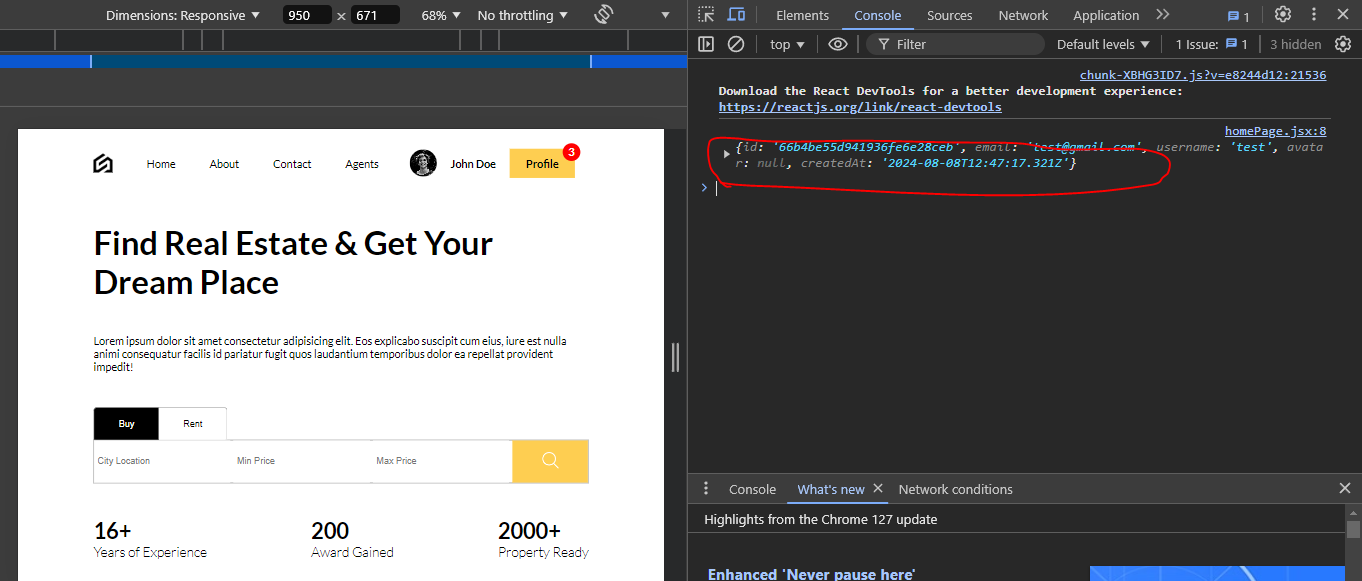




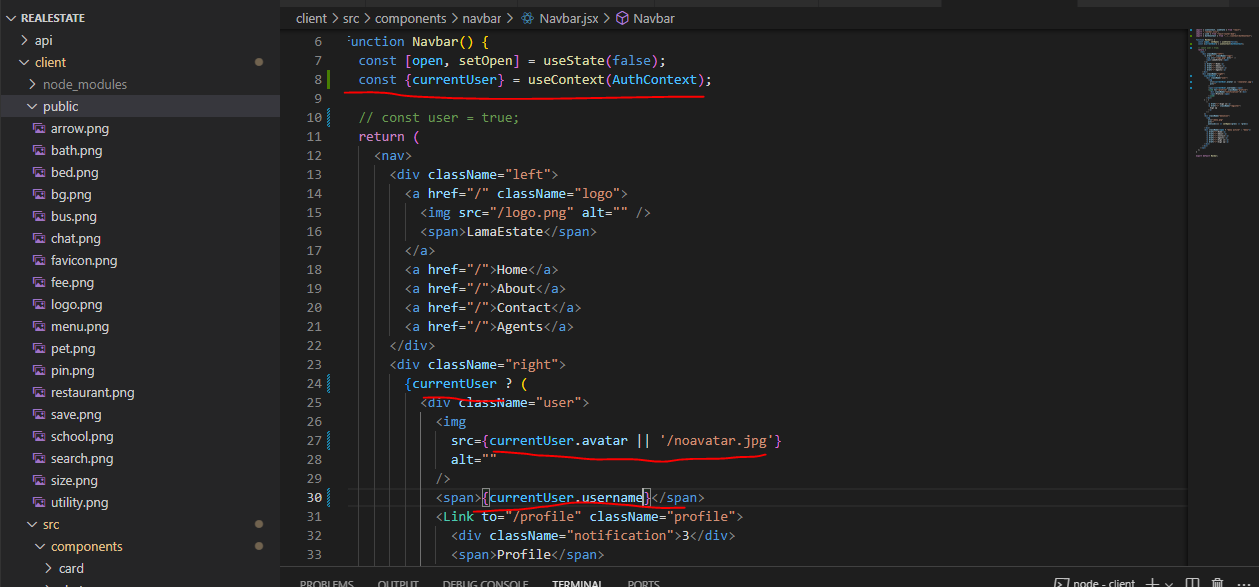
After clicking on logout and redirecting to login page

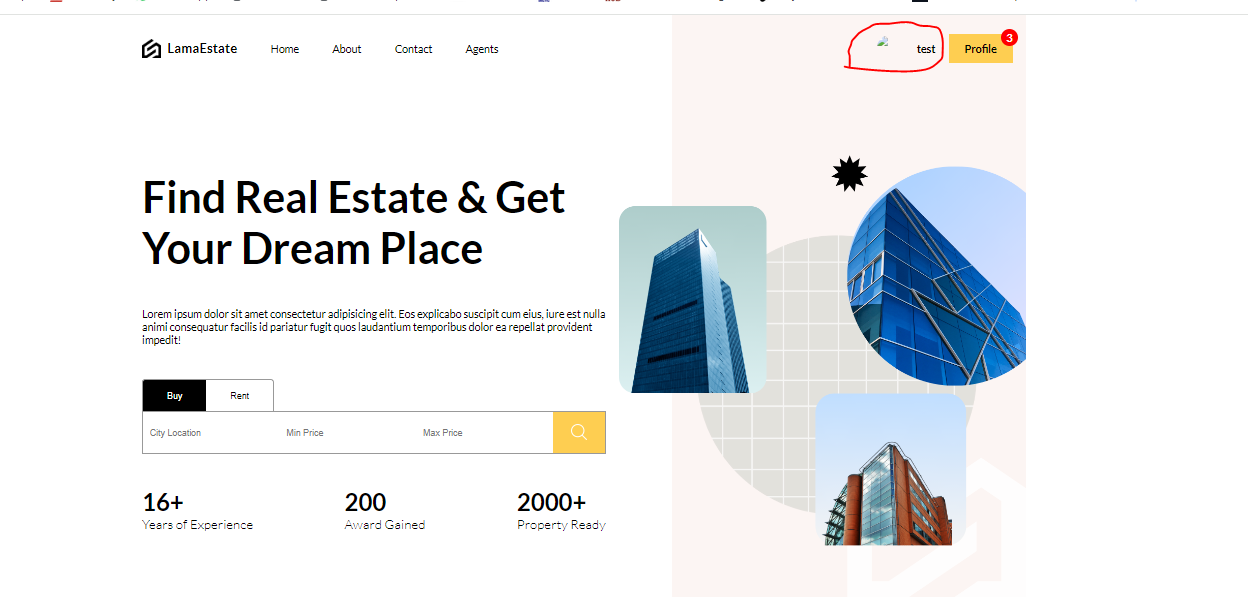


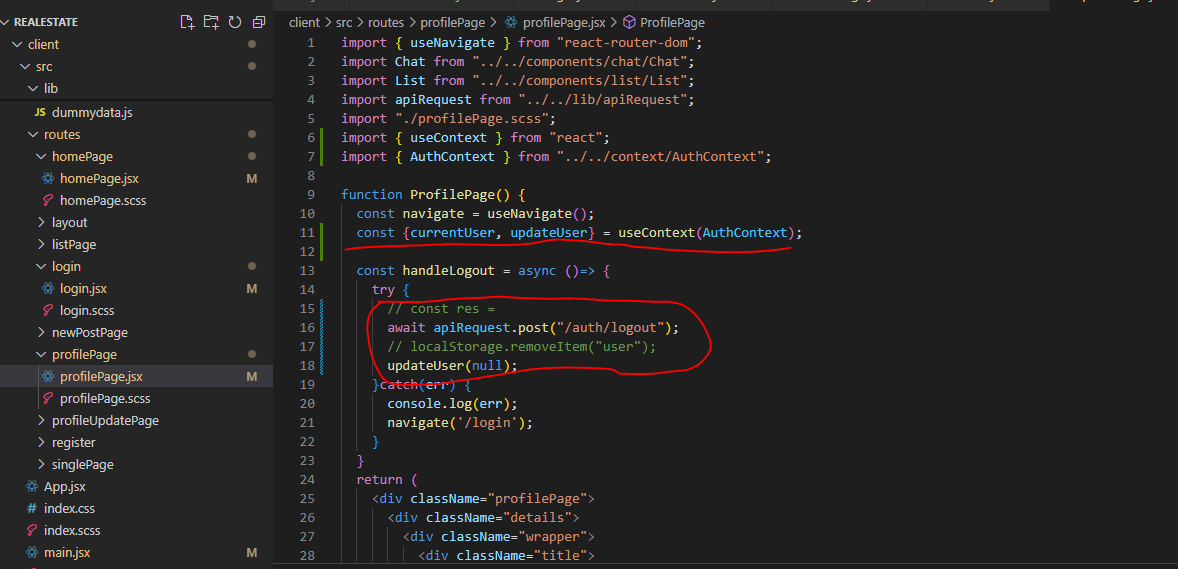
After login we can see the user without refresing the page

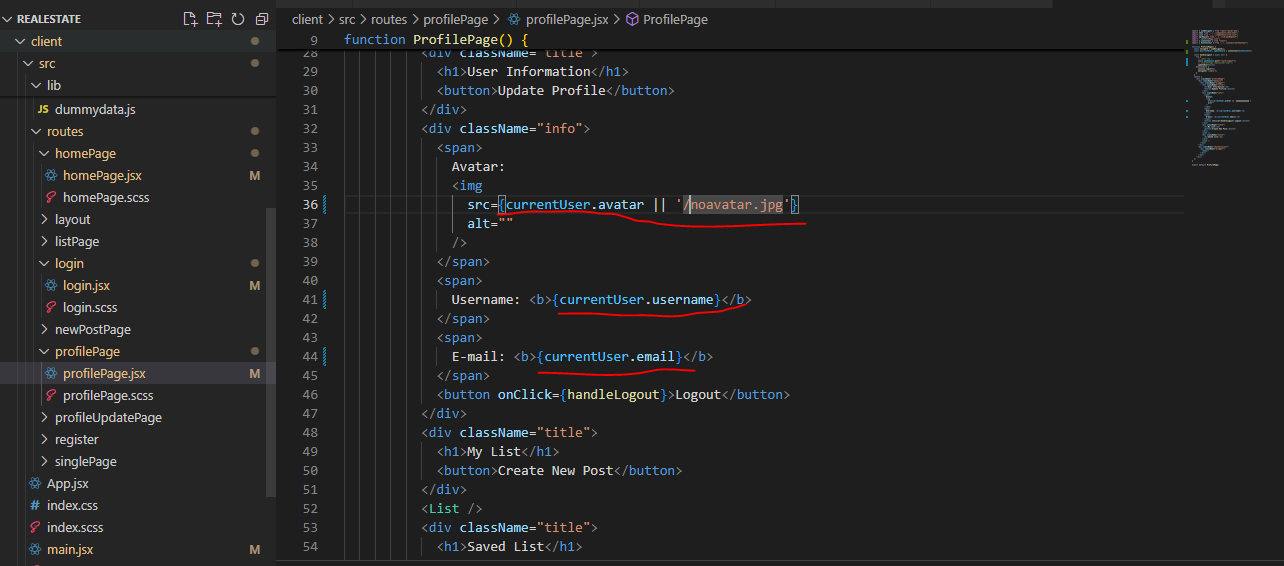


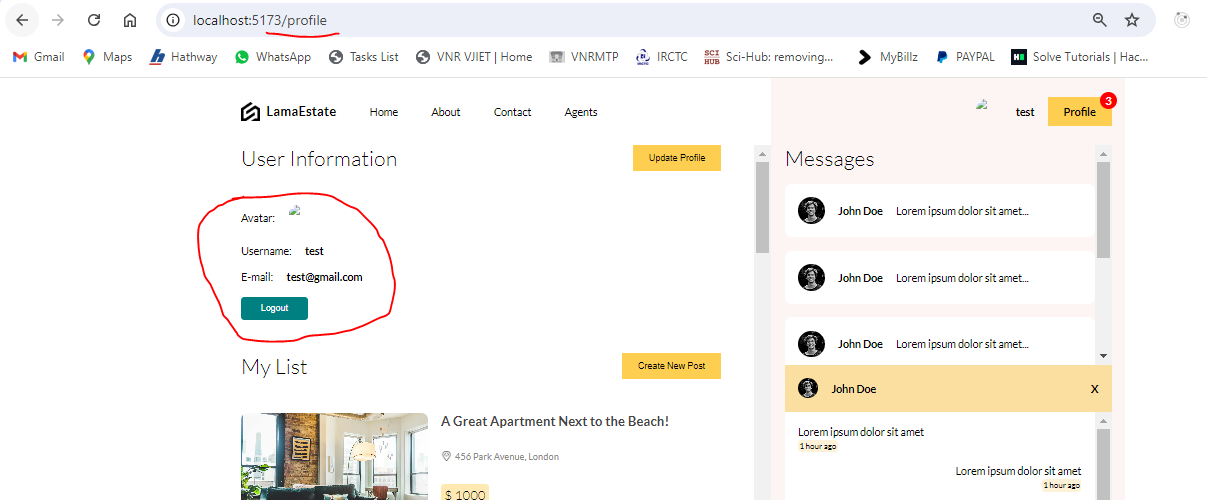
So, in the Navbar we can call usecontext and use our user information.













Now these links points to login and register page.

