# **WEEK 7 React HANDS ON**

**SUBMITTED BY :-**

NAME – AYUSH SAHOO

SUPERSET ID – 6372243

COURSE – DIGITAL NURTURE 4.0 JAVA FSE

EMAIL – [ayush.sahoo003@gmail.com](mailto:ayush.sahoo003@gmail.com)

COLLEGE – C.V. RAMAN GLOBAL UNIVERSITY

**-----------------NOTE: I HAVE ALSO DONE ADDITIONAL IMPORTANT HANDS ON -------------------**

**MANDATORY HANDS ON**

**EXERCISE 1:**

**Estimated time to complete this lab: 60 minutes.**

**Create a React Application named “cricketapp” with the following components:**

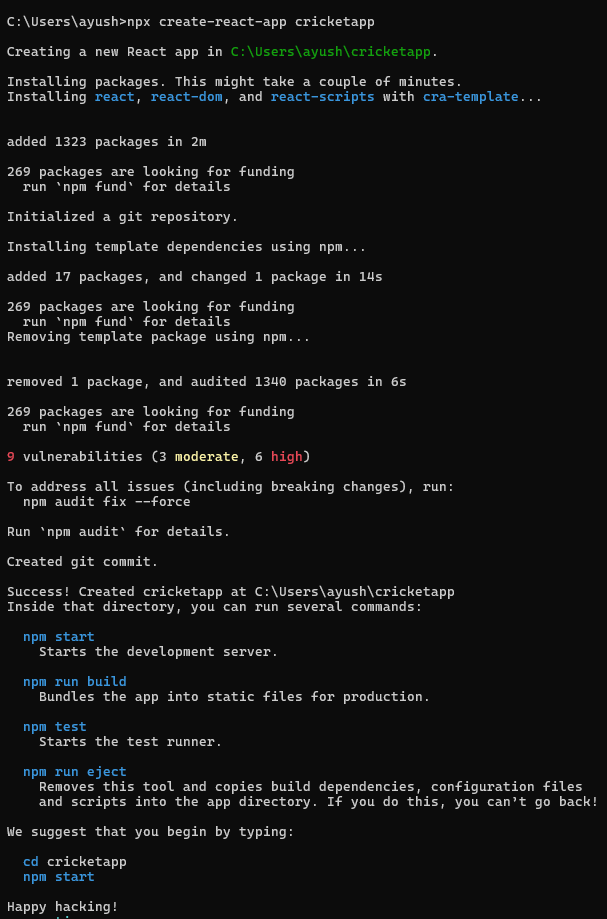
1. **ListofPlayers**

* **Declare an array with 11 players and store details of their names and scores using the map feature of ES6**
* **Filter the players with scores below 70 using arrow functions of ES6.**

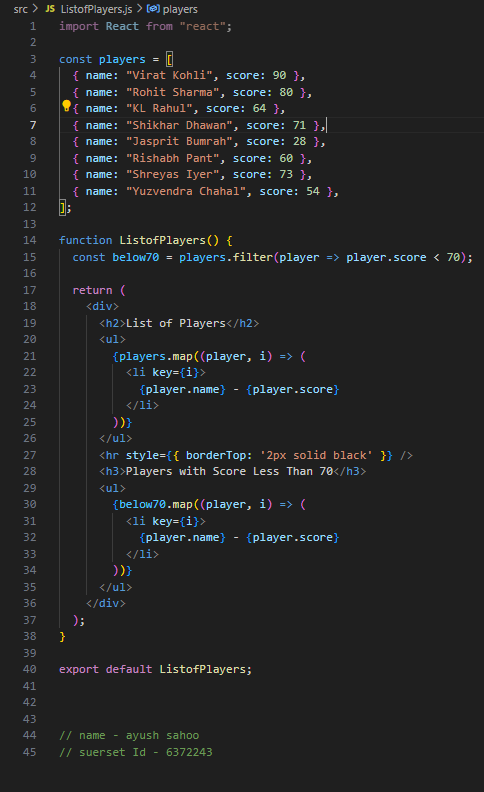
1. **IndianPlayers**
   * **Display the Odd Team Player and Even Team players using the Destructuring features of ES6**
   * **Declare two arrays T20players and RanjiTrophy players and merge the two arrays and display them using the Merge feature of ES6**

**SOLUTION:-**

Step 1: C:\Users\ayush>npx create-react-app cricketapp.



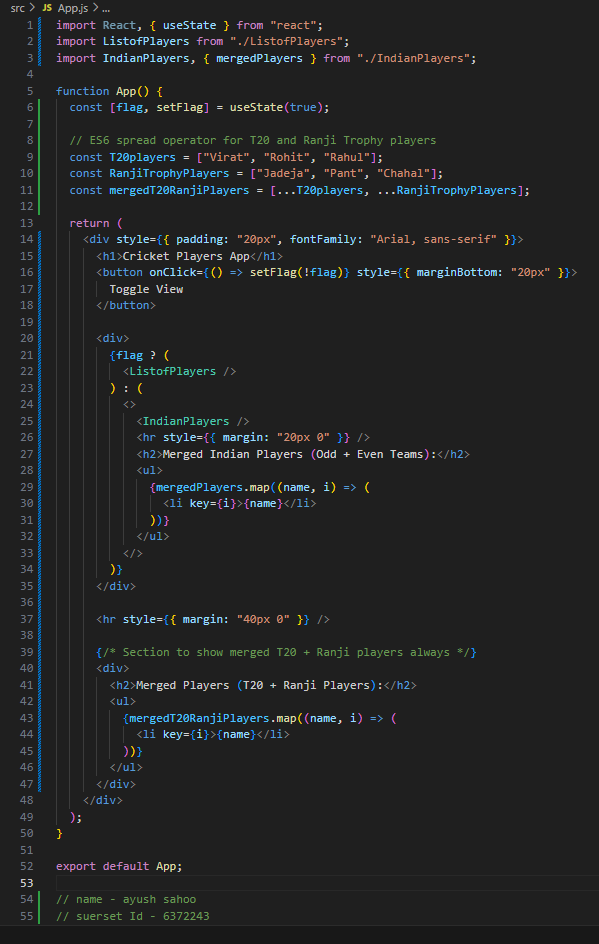
Step 2: Create “ListOfPlayers.js” in src folder.



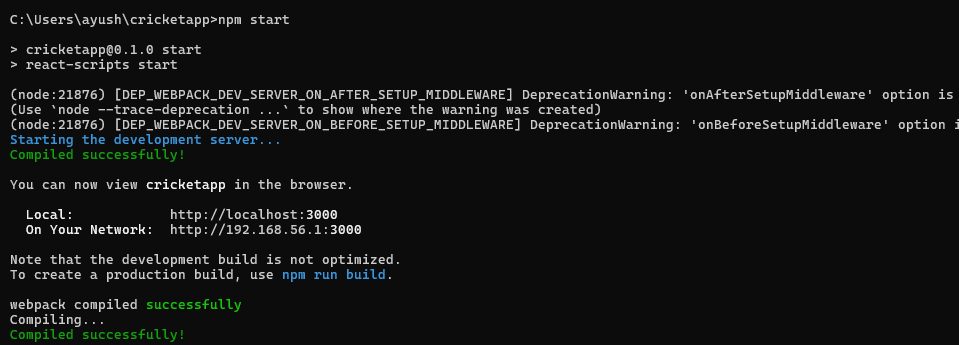
Step 3: Create “IndianPlayers.js” inside src folder.

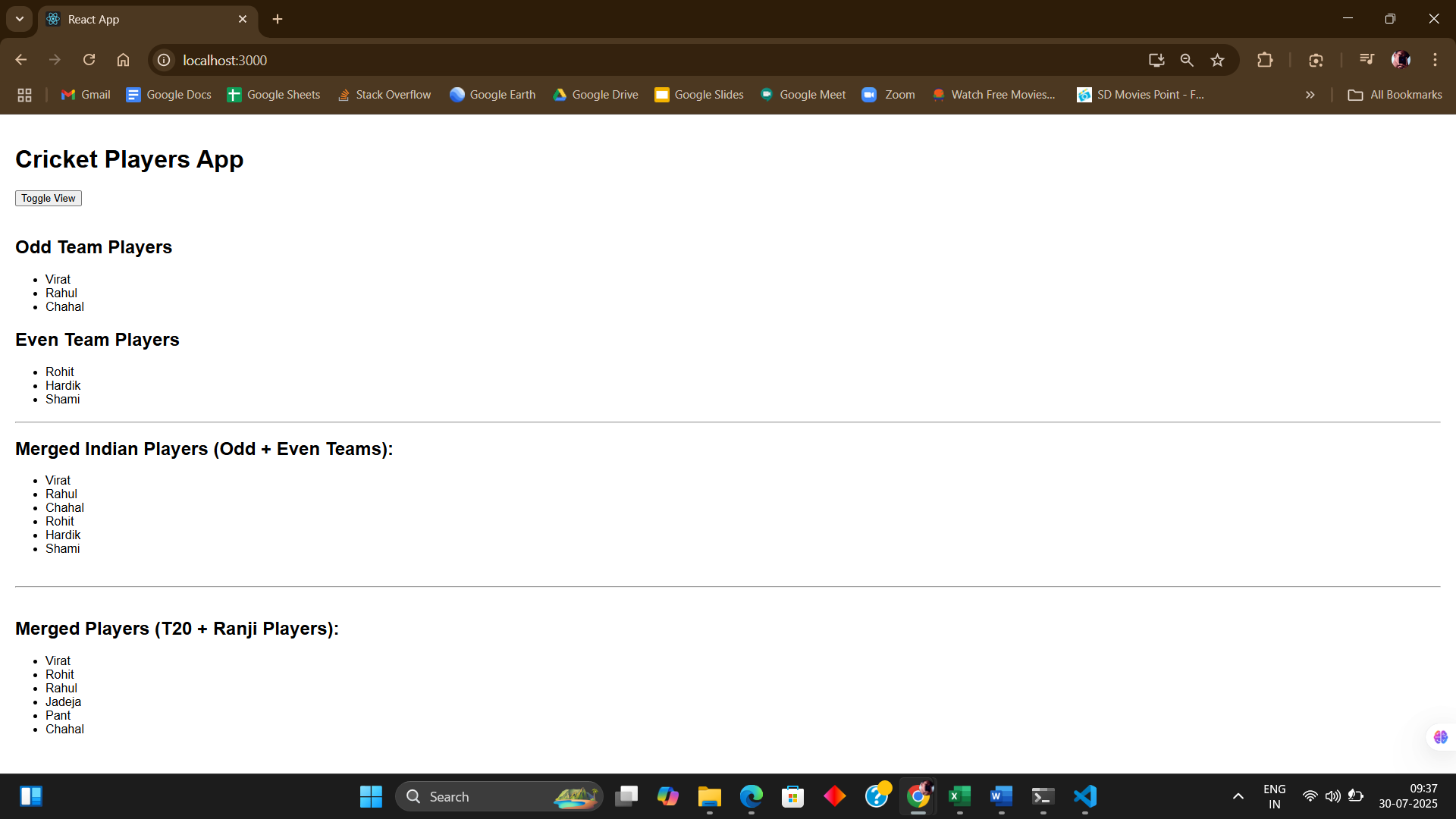


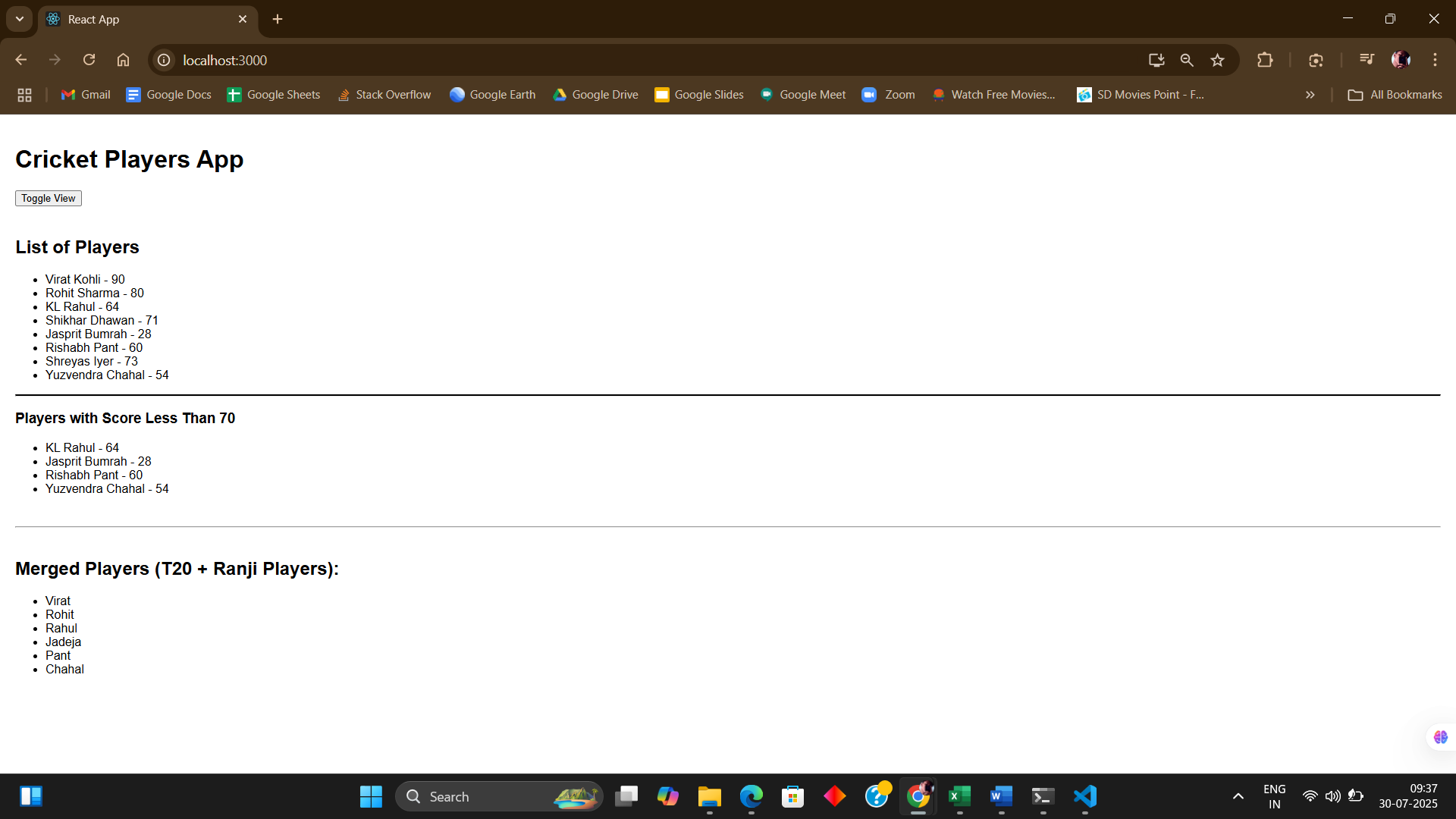
Step 4: Update “App.js”.



Step 5: C:\Users\ayush\cricketapp>npm start





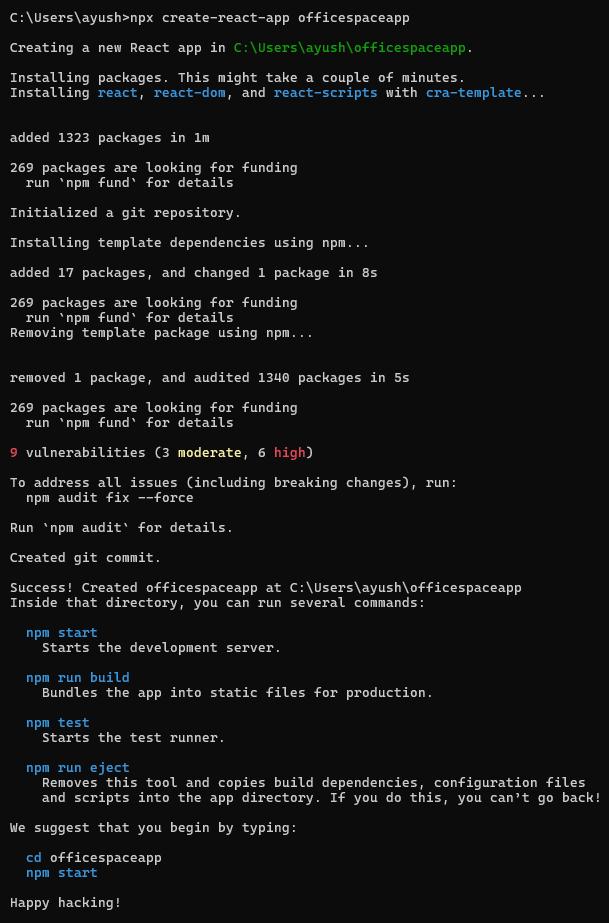


**EXERCISE 2: Create a React Application named “officespacerentalapp” which uses React JSX to create elements, attributes and renders DOM to display the page.**

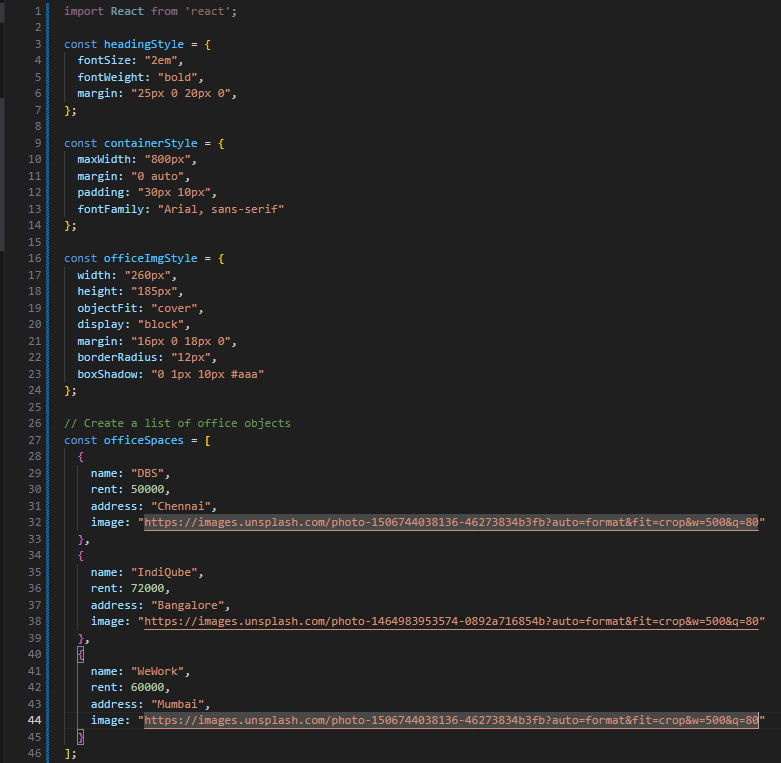
* **Create an element to display the heading of the page.**
* **Attribute to display the image of the office space**
* **Create an object of office to display the details like Name, Rent and Address.**
* **Create a list of Object and loop through the office space item to display more data.**
* **To apply Css, Display the color of the Rent in Red if it’s below 60000 and in Green if it’s above 60000.**

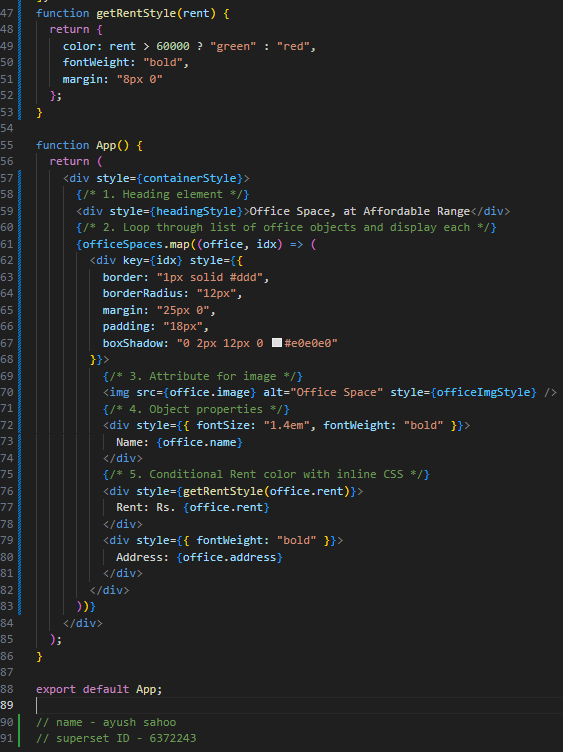
**SOLUTION:**

Step 1: C:\Users\ayush>npx create-react-app officespaceapp

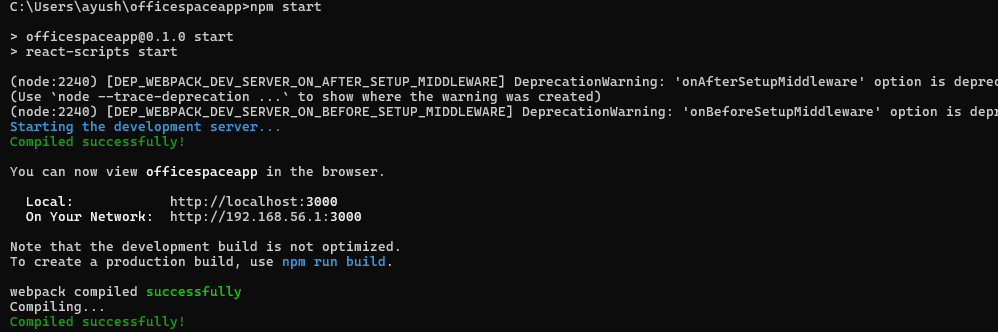


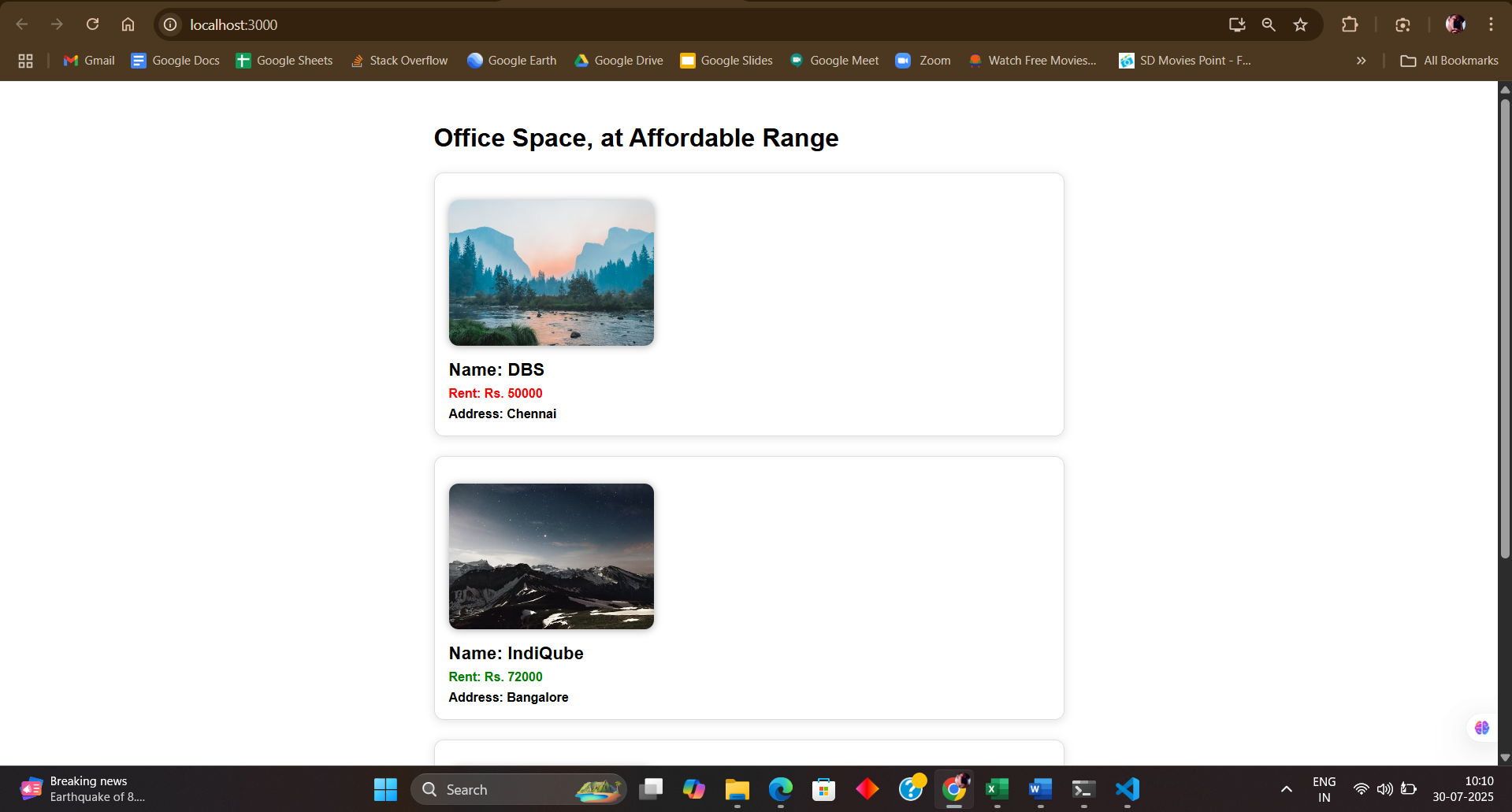
Step 2: Update “App.js”.





Step 3: C:\Users\ayush\officespaceapp>npm start



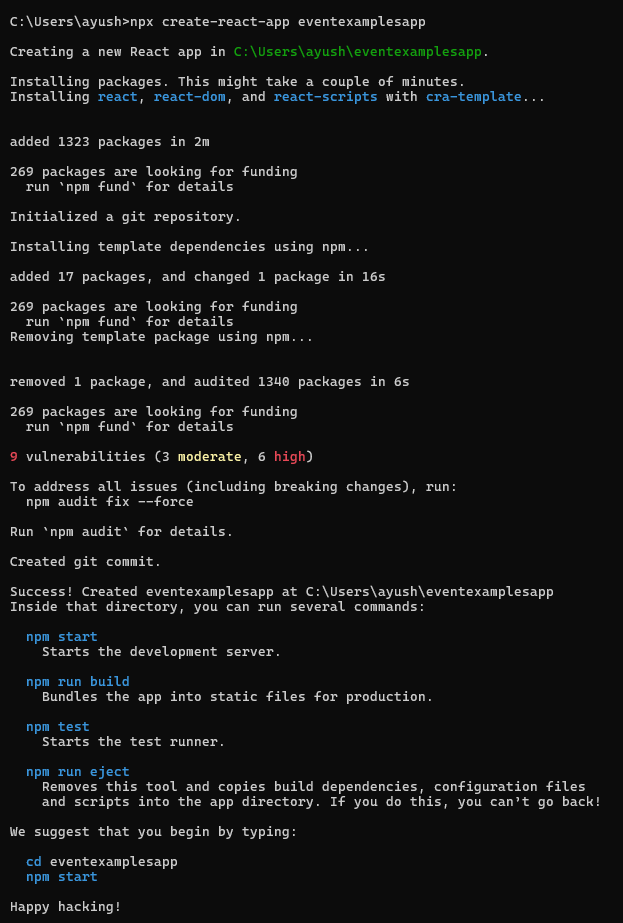


**EXERCISE 3 : Create a React Application “eventexamplesapp” to handle various events of the form elements in HTML.**

1. **Create “Increment” button to increase the value of the counter and “Decrement” button to decrease the value of the counter. The “Increase” button should invoke multiple methods.** 
   1. **To increment the value**
   2. **Say Hello followed by a static message.**
2. **Create a button “Say Welcome” which invokes the function which takes “welcome” as an argument.**
3. **Create a button which invokes synthetic event “OnPress” which display “I was clicked”**
4. **Create a “CurrencyConvertor” component which will convert the Indian Rupees to Euro when the Convert button is clicked.**
5. **Handle the Click event of the button to invoke the handleSubmit event and handle the conversion of the euro to rupees.**

**SOLUTION:**

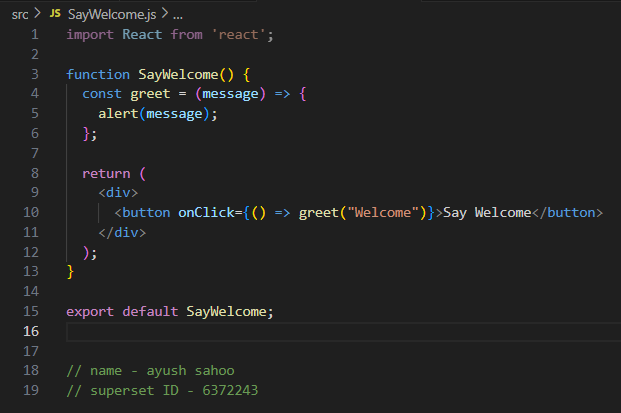
Step 1:



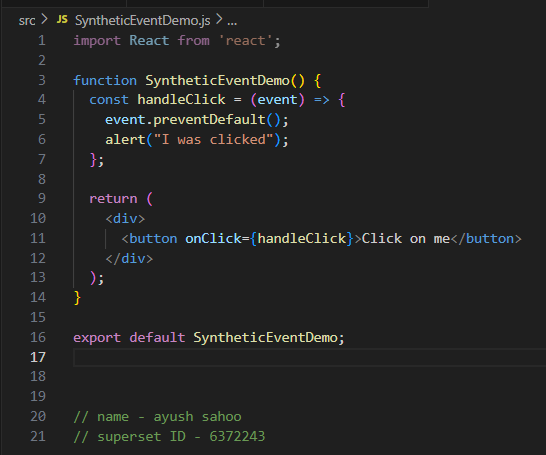
Step 2:Create “Counter.js” in src folder.



Step 3: Create “SayWelcome.js” in src folder.



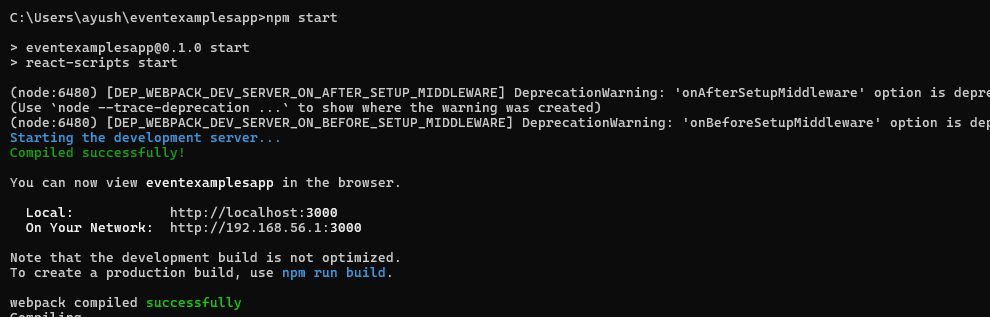
Step 4: Create “SyntheticEventDemo.js” in src folder.

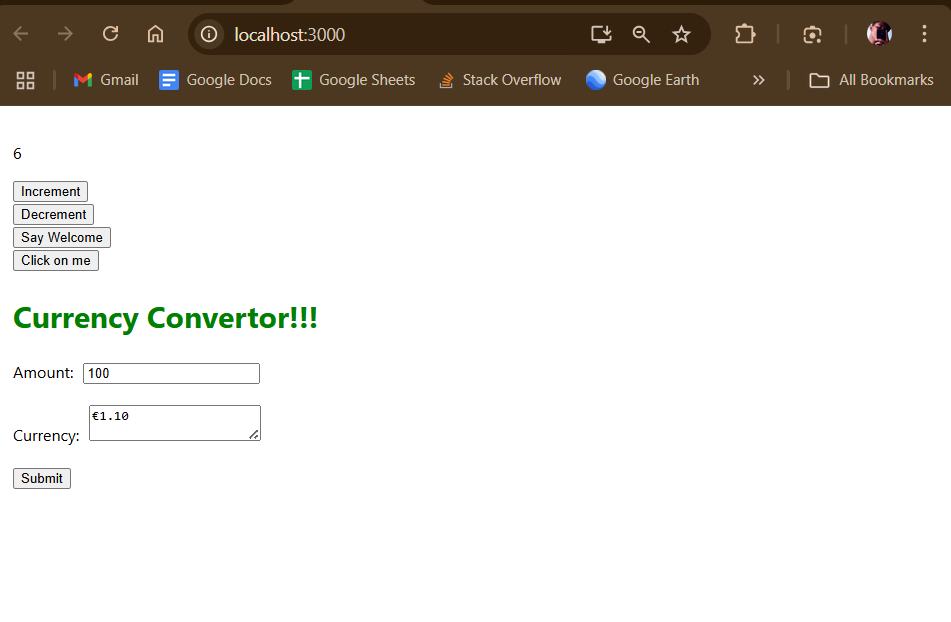


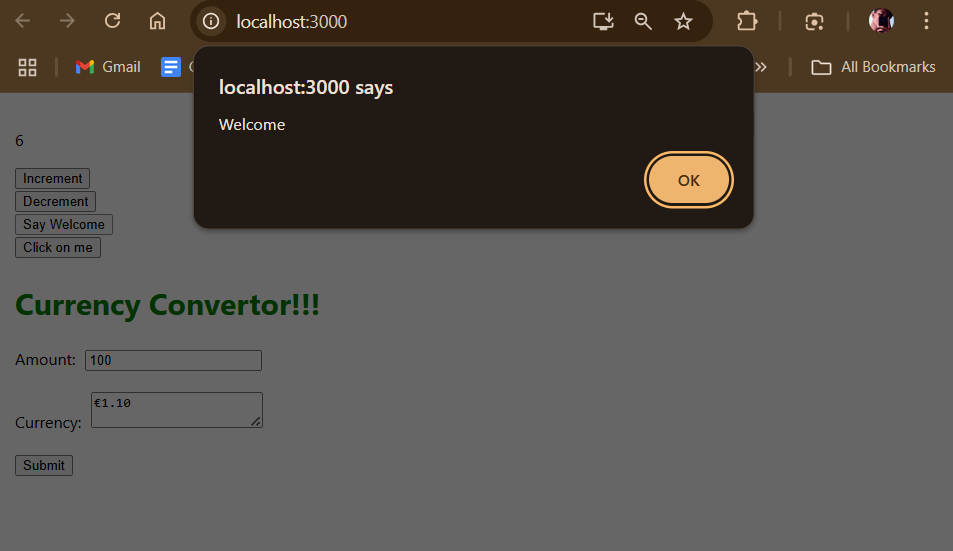
Step 5: Create “CurrencyConverter.js” in src folder.

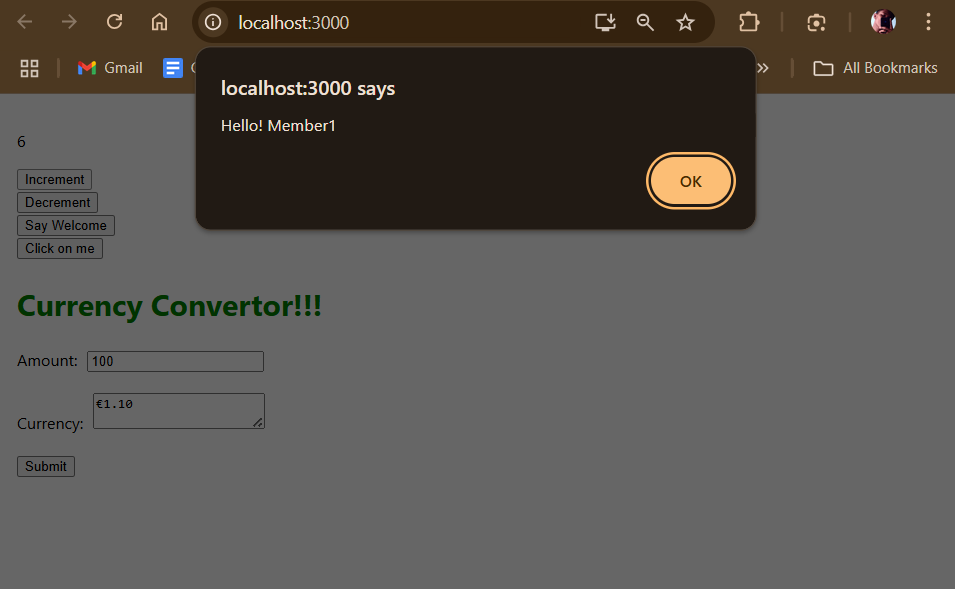


Step 6: OUTPUT









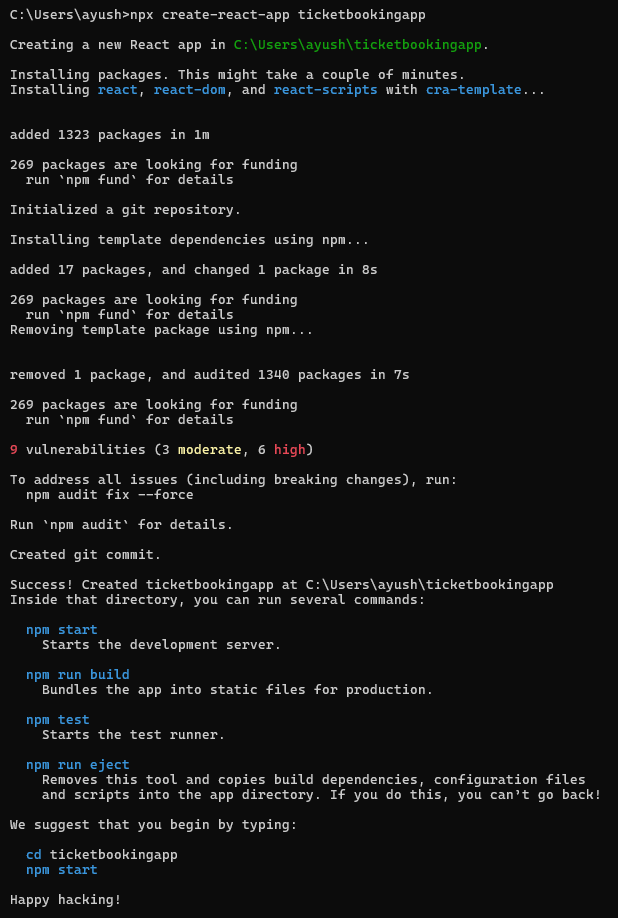
**EXERCISE 4:**

**Create a React Application named “ticketbookingapp” where the guest user can browse the page where the flight details are displayed whereas the logged in user only can book tickets.**

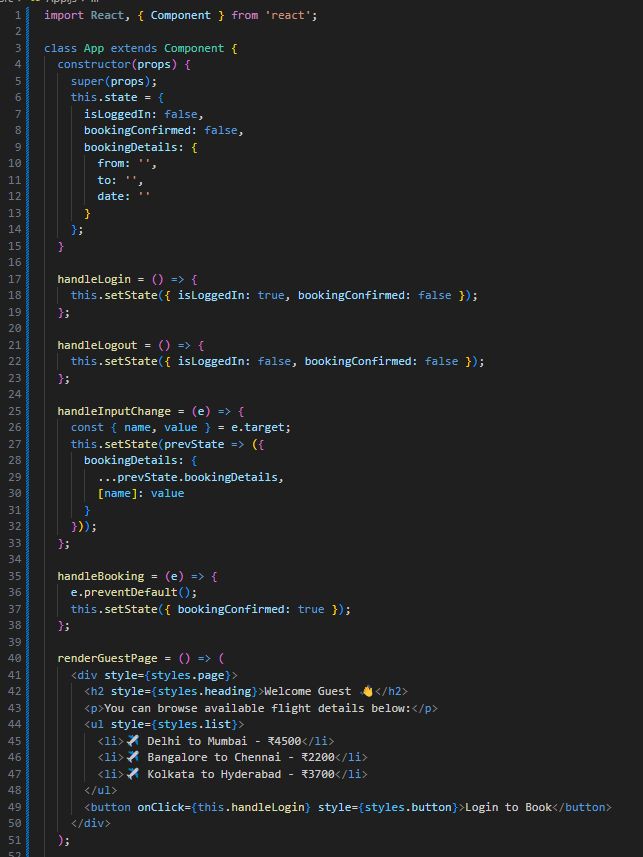
**The Login and Logout buttons should accordingly display different pages. Once the user is logged in the User page should be displayed. When the user clicks on Logout, the Guest page should be displayed.**

**SOLUTION:**

Step 1:

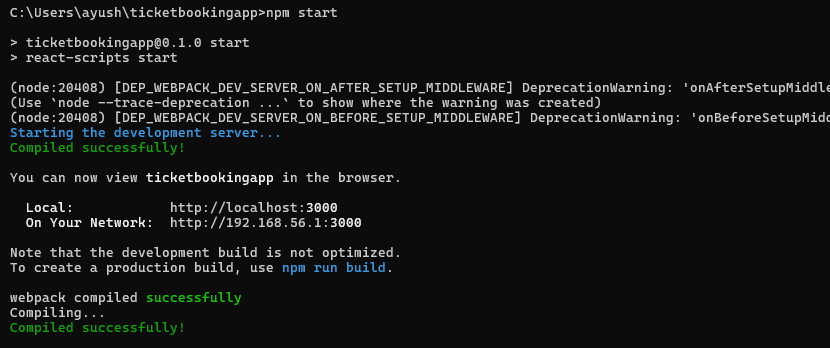


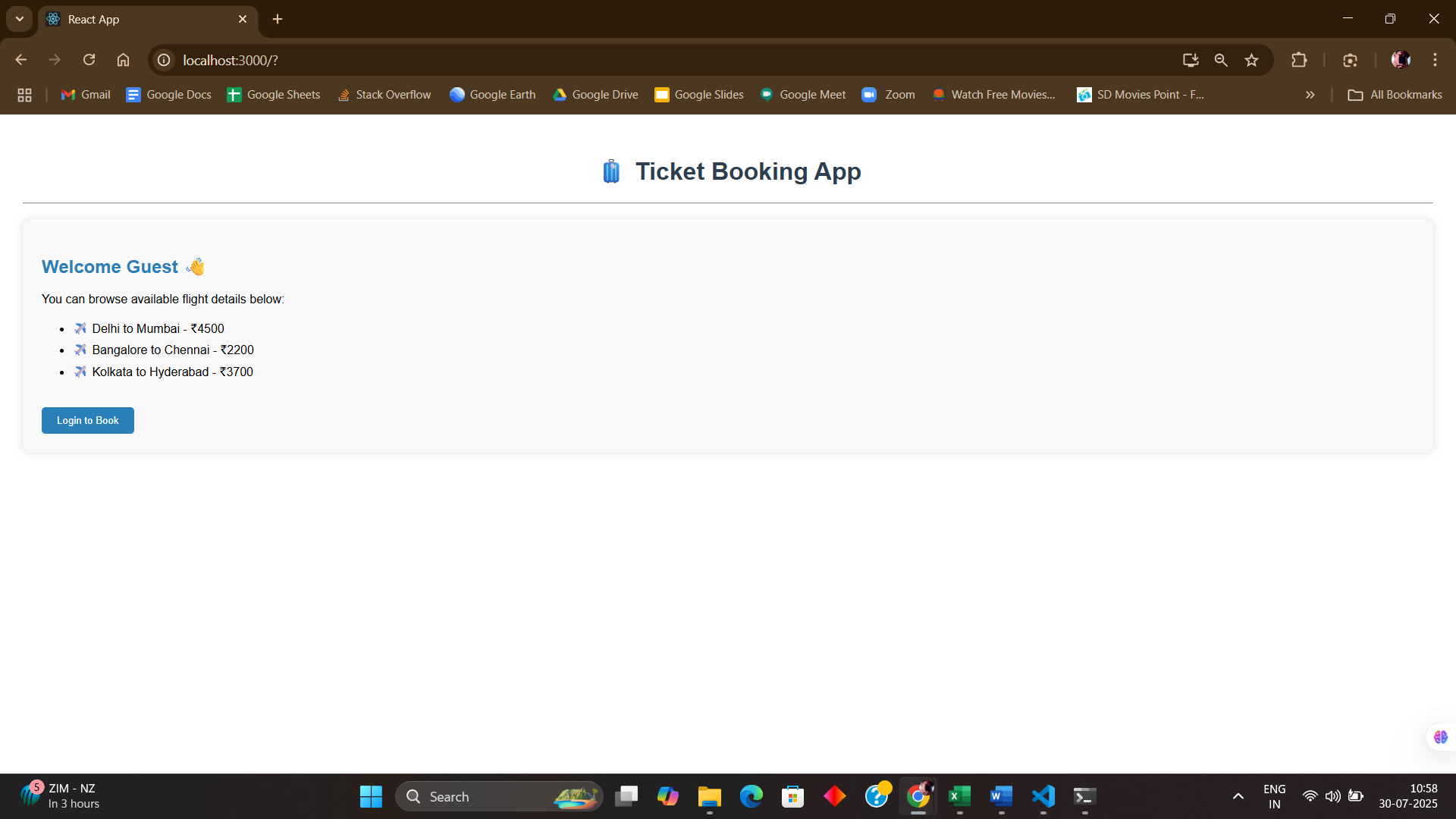
Step 2:

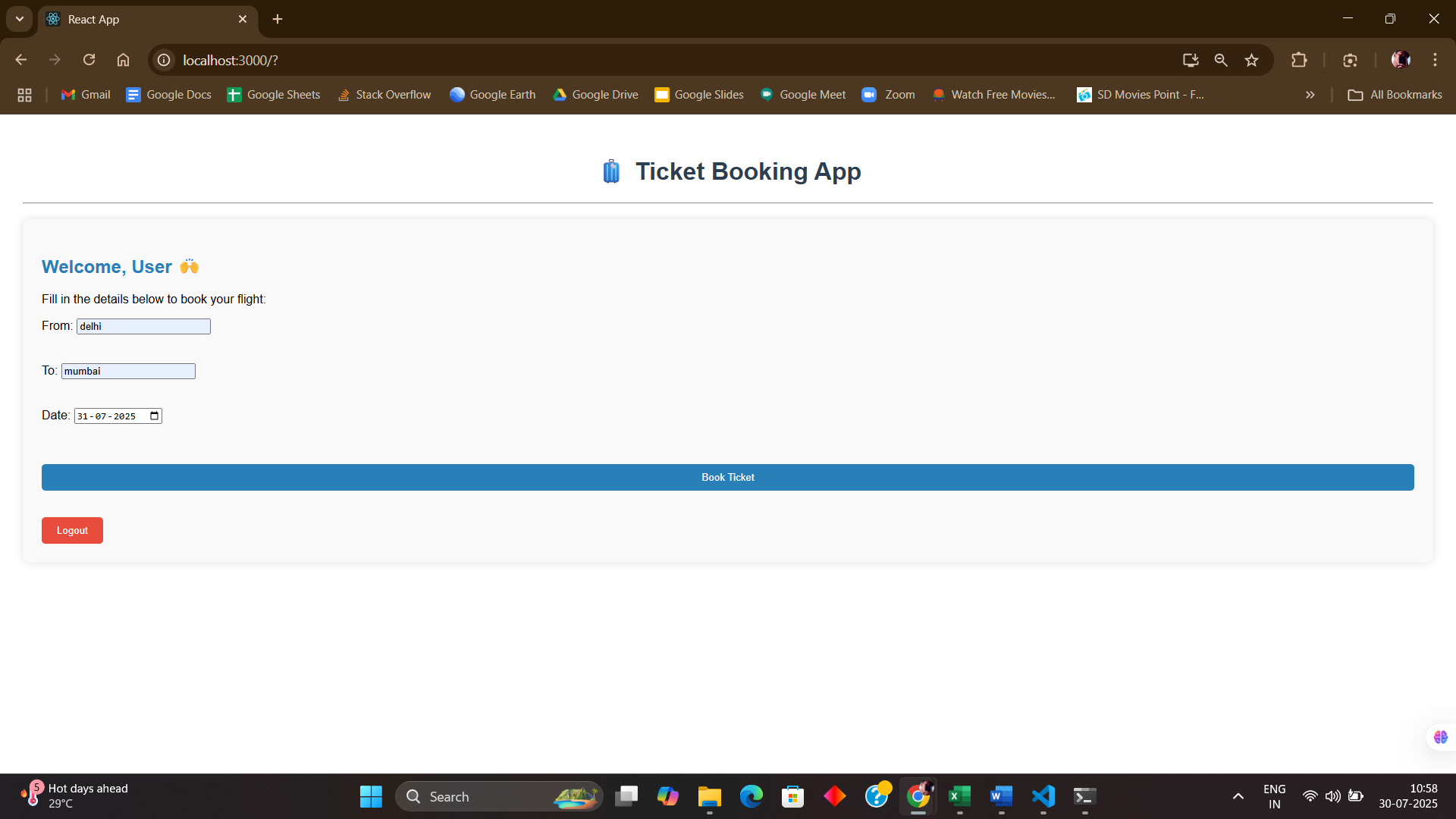
 

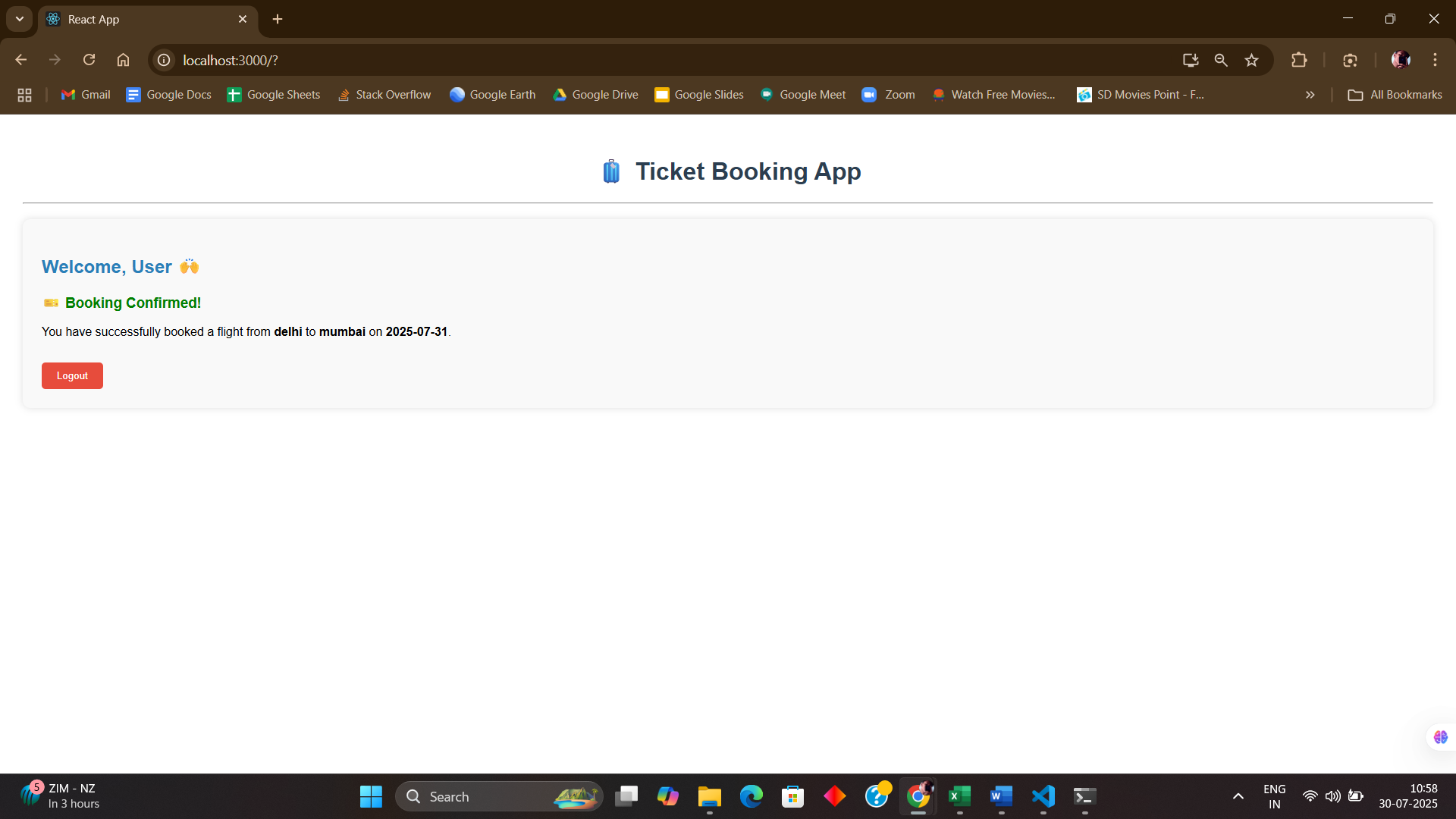


Step 3: OUTPUT









**EXERCISE 5:Create a React App named “bloggerapp” in with 3 components.**

1. **Book Details**
2. **Blog Details**
3. **Course Details**

**Implement this with as many ways possible of Conditional Rendering.**

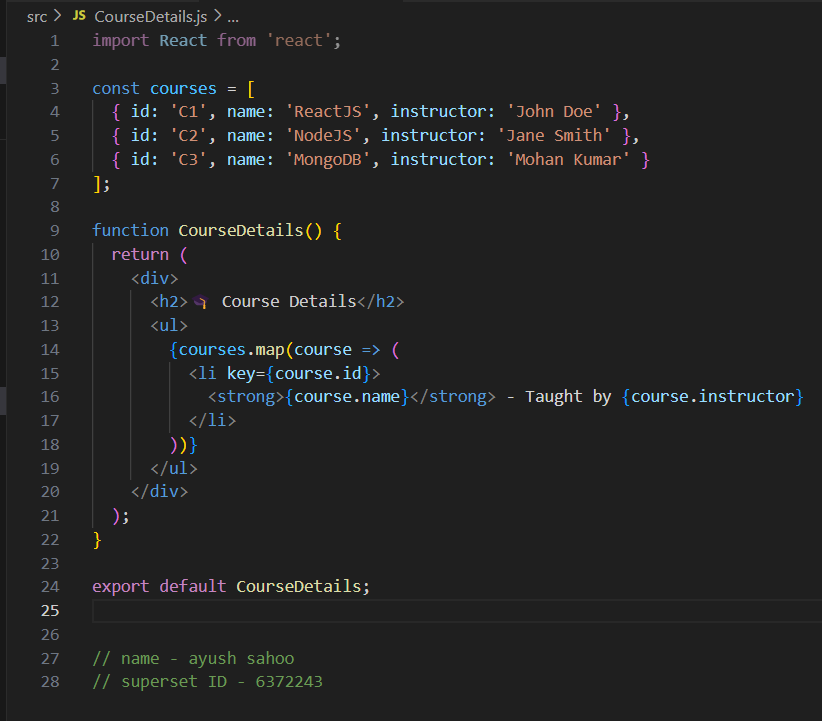
**SOLUTION:**

Step 1:

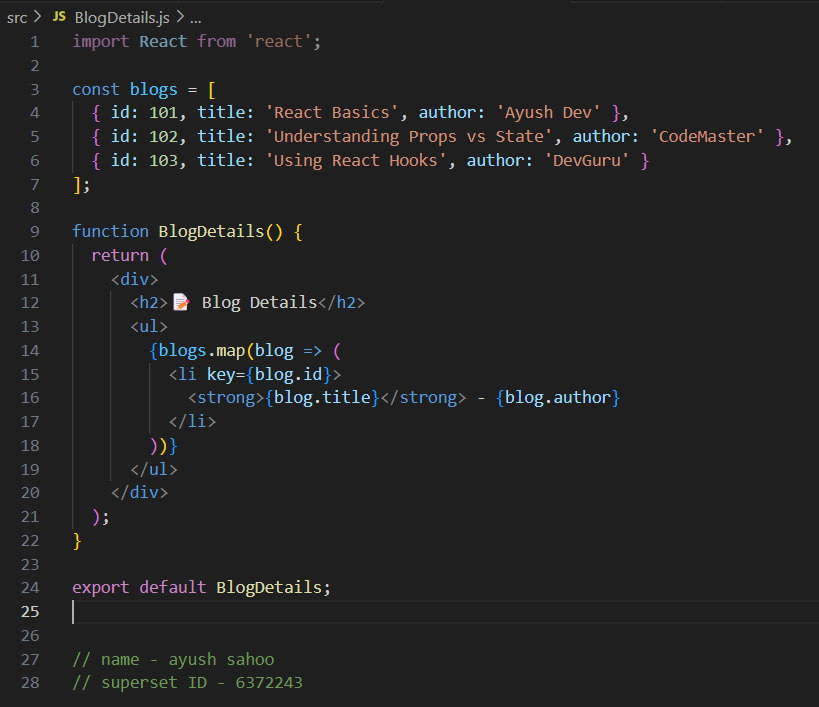
Step 2: Create “BookDetails.js” under src folder.



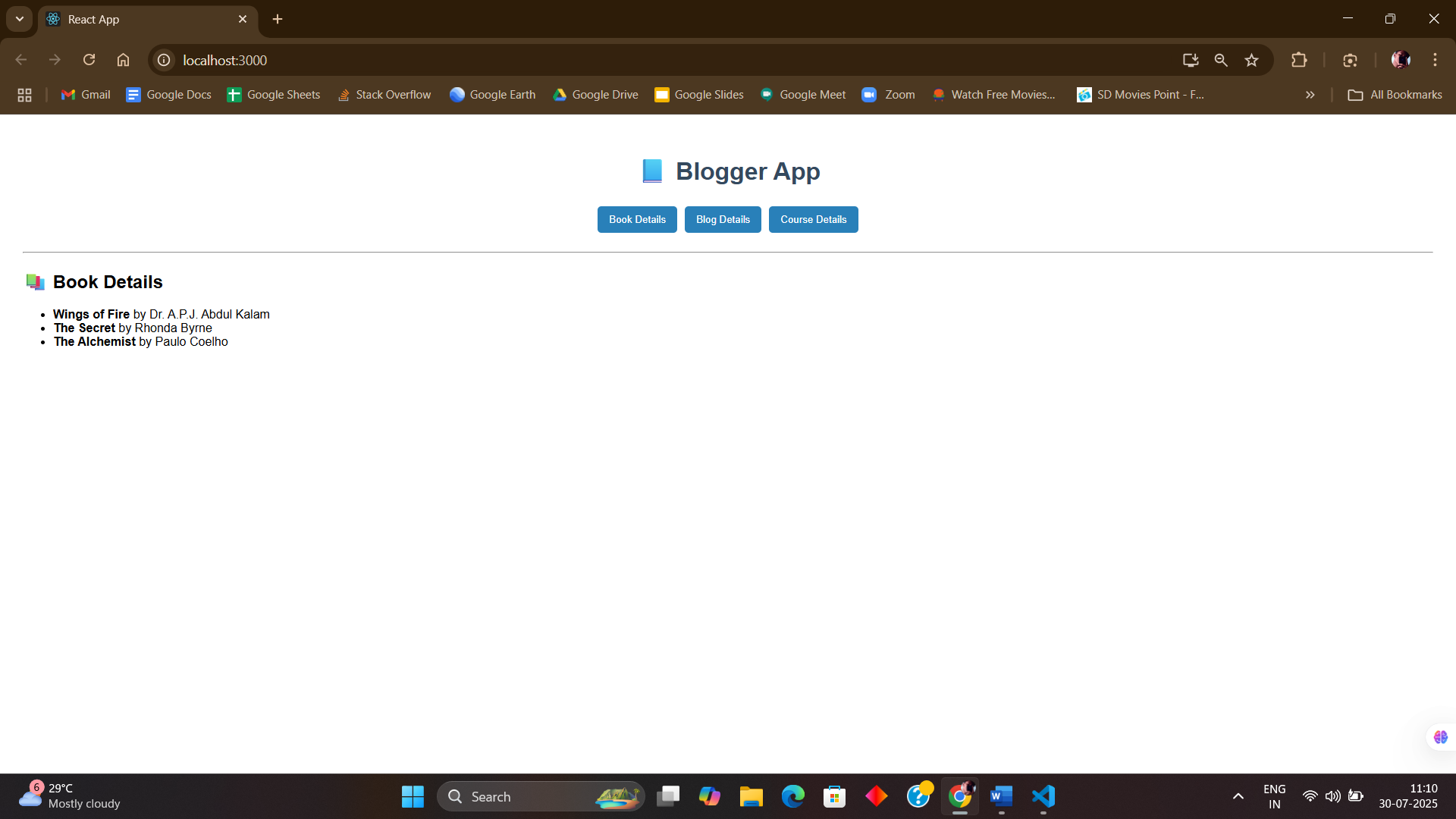
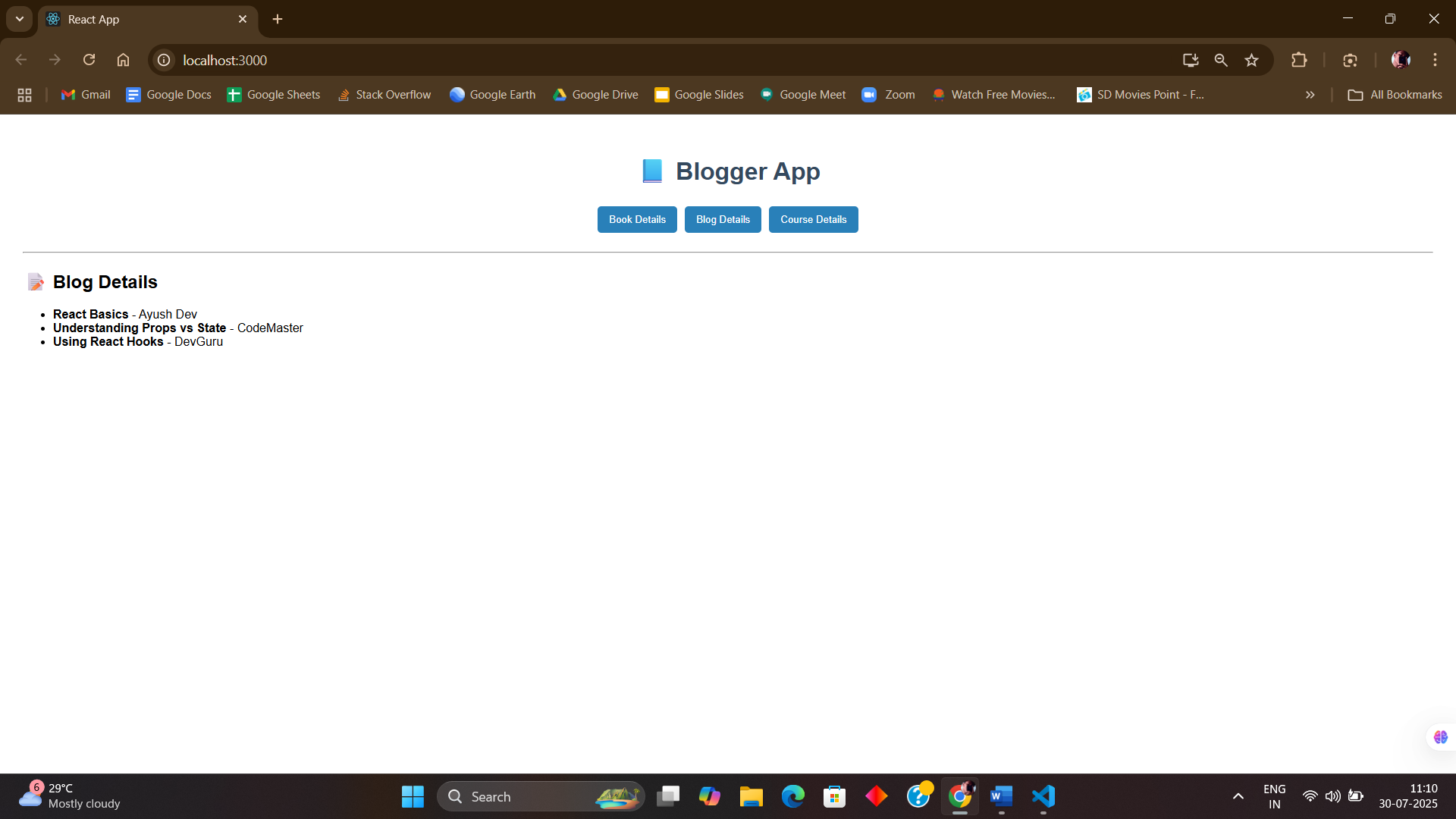
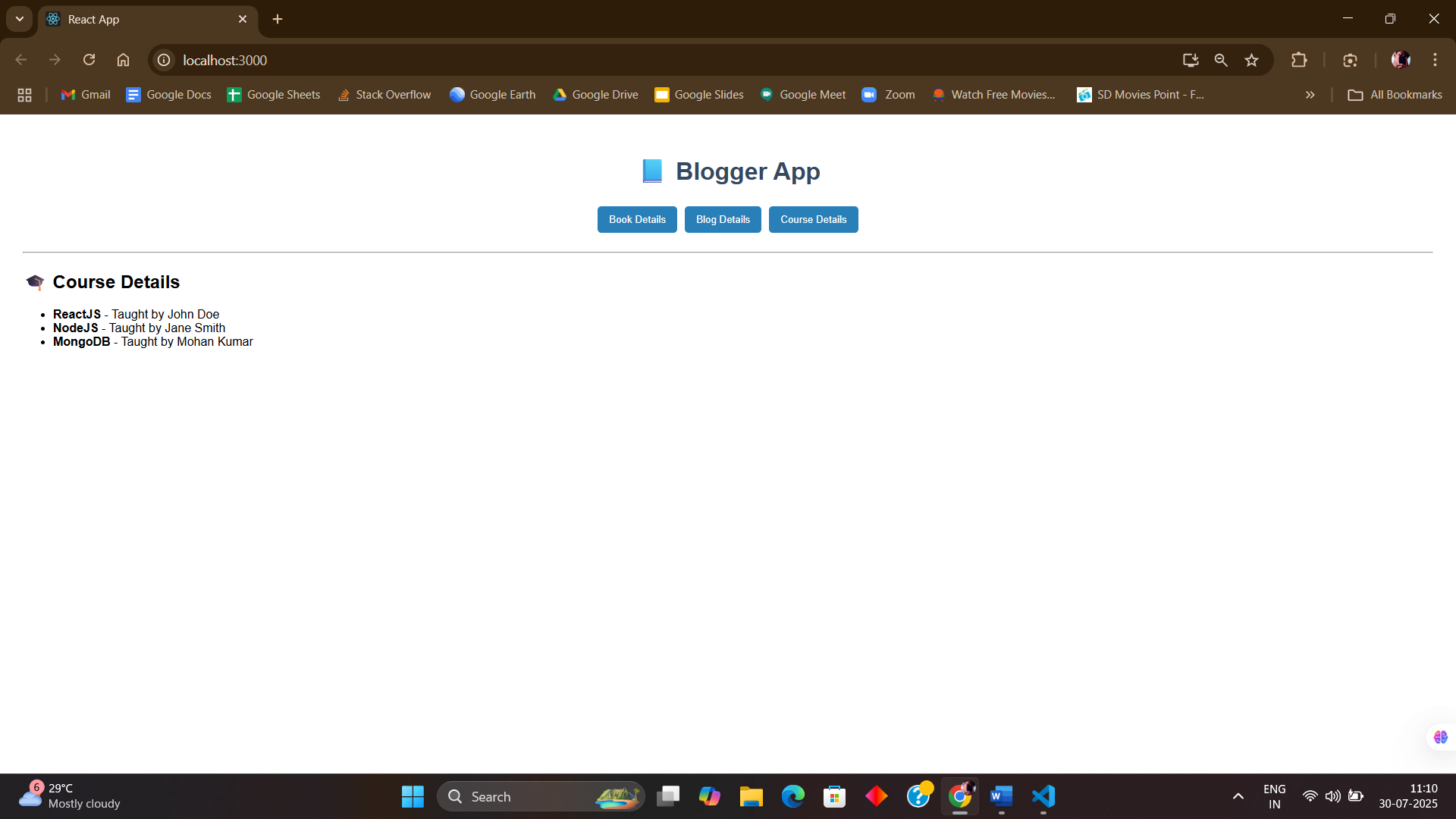
Step 3: Create “CourseDetails.js” under src folder.



Step 4: Create “BlogDetails.js” under src folder.



Step 5: OUTPUT

ADDITIONAL HANDS ON

**EXERCISE 6:**

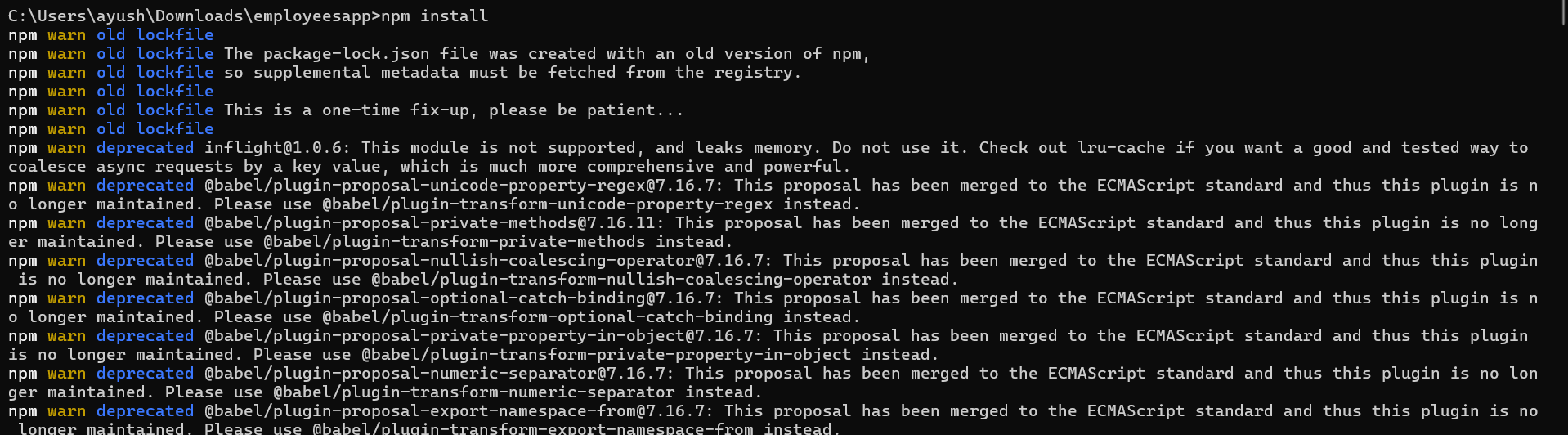
**Developers of Apps Centric Solutions have created an employee management application which supports light and dark themes for the buttons. The current solution uses the react state and props to provide the theme name to be used from App component to Employee List component and from there to Employee Card component. Quality assurance team analyzed the solutions and found the technique being used to be a substandard one. React architect suggested to use the react context API to share the theme name with nested child components instead of passing them down using props from the parent component.**

**You are assigned the task of converting the application form props only to React Context API.**

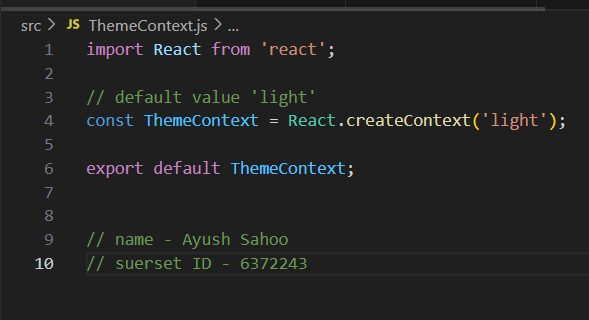
1. **Unzip the application and open it using VS Code**
2. **Go to terminal and execute *npm install* command to restore all the node modules**
3. **Run the application once to see the output. Use npm start command.**
4. **Explore the components present in App.js, EmployeesList.js and EmployeeCard.js files.**
5. **Create a new file with the name as ThemeContext.js. Define a new context in the file with the name as ThemeContext and assign it a default value of ‘light’ and export it as default form the module.**
6. **Open App component present in App.js file.** 
   1. **Import the ThemeContext in App component.**
   2. **Define the theme context provider to be the entire JSX of the App component.**
   3. **Assign the value for the theme provider from the state of the component.**
   4. **Modify the call to EmployeeList component so that theme name is no longer passed as props.**
7. **Go to EmployeeList component present in EmployeeList.js file and modify it so that theme name is not passed explicitly to its child component.**
8. **Go to EmployeeCard component inside EmployeeCard.js file**
   1. **Import the ThemeContext into the component file**
   2. **Retrieve the value of the context with the help of useContext() and store it in a variable**
   3. **Use the variable to pass the className for the buttons.**

**SOLUTION:**

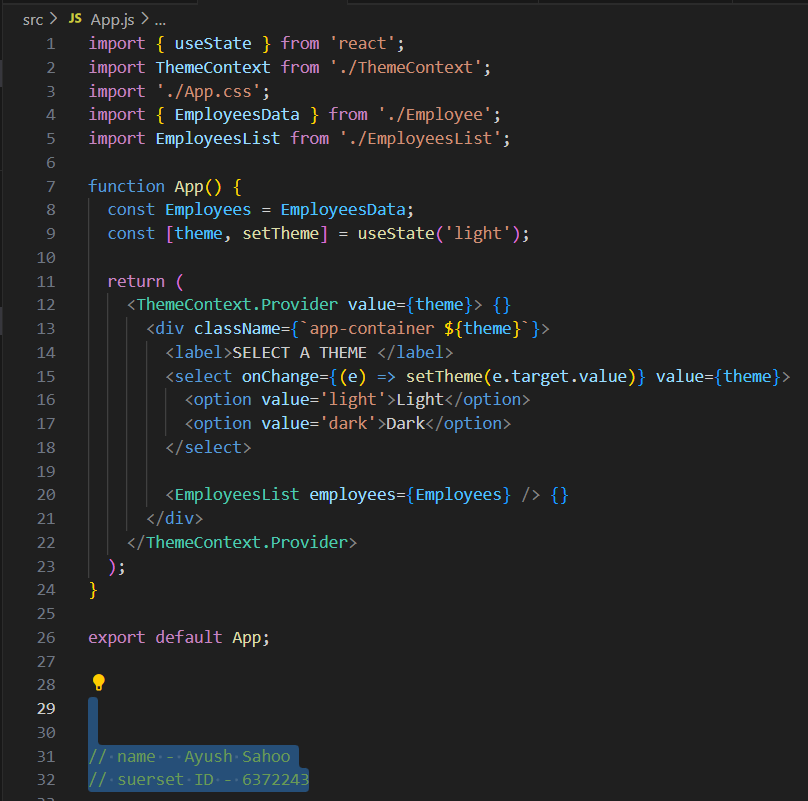
Step 1: C:\Users\ayush\Downloads\employeesapp>npm install



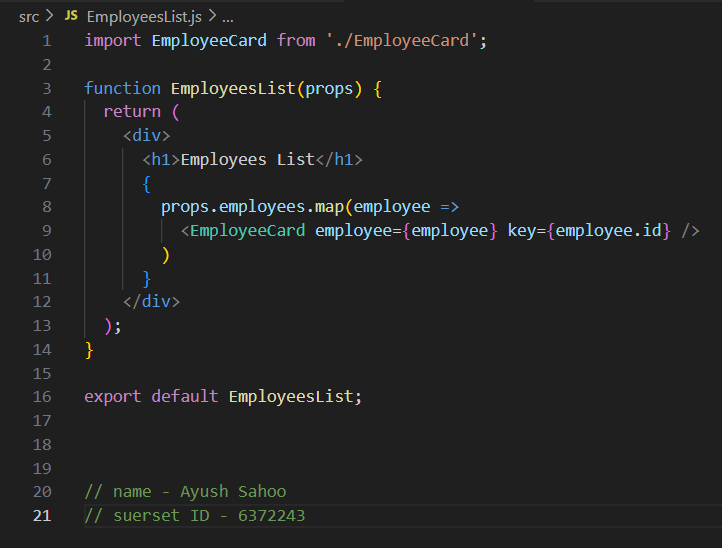
Step 2: Create “ThemeContext.js” under src folder of “employeesapp”.



Step 3: Update “App.js” under src folder.



Step 4: Update “EmployeesList.js” under src folder.



Step 5: Update “EmployeeCard.js” under src folder.



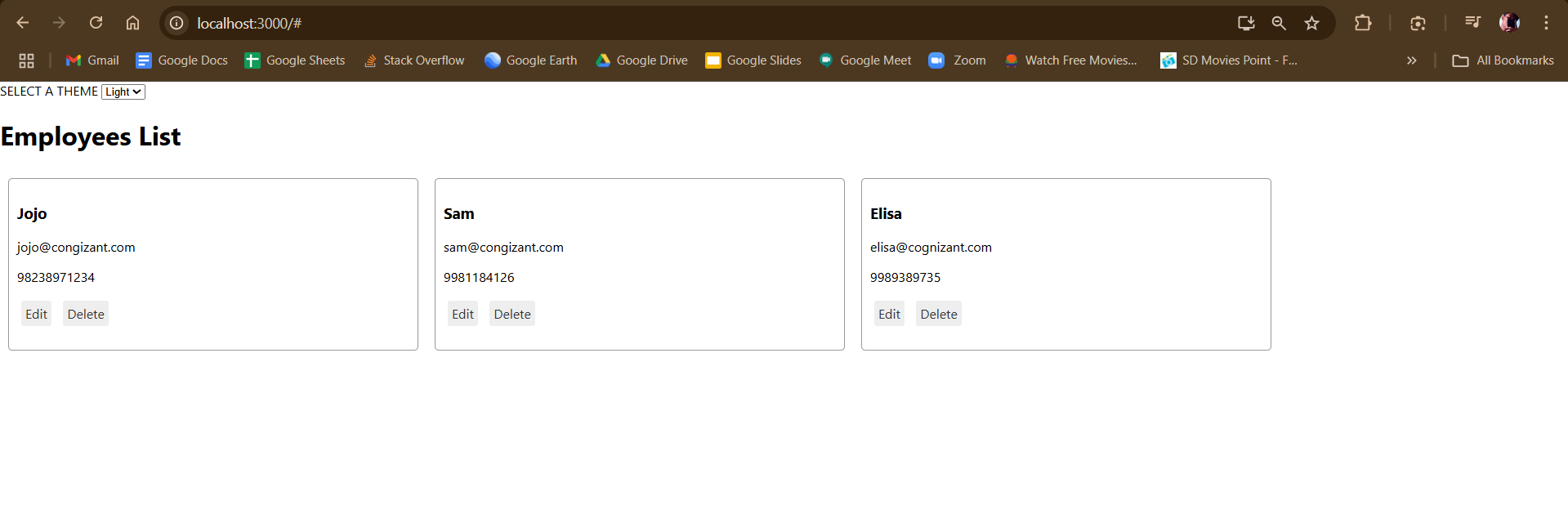
Step 6: Update “EmployeeCard.module.css” under src folder.



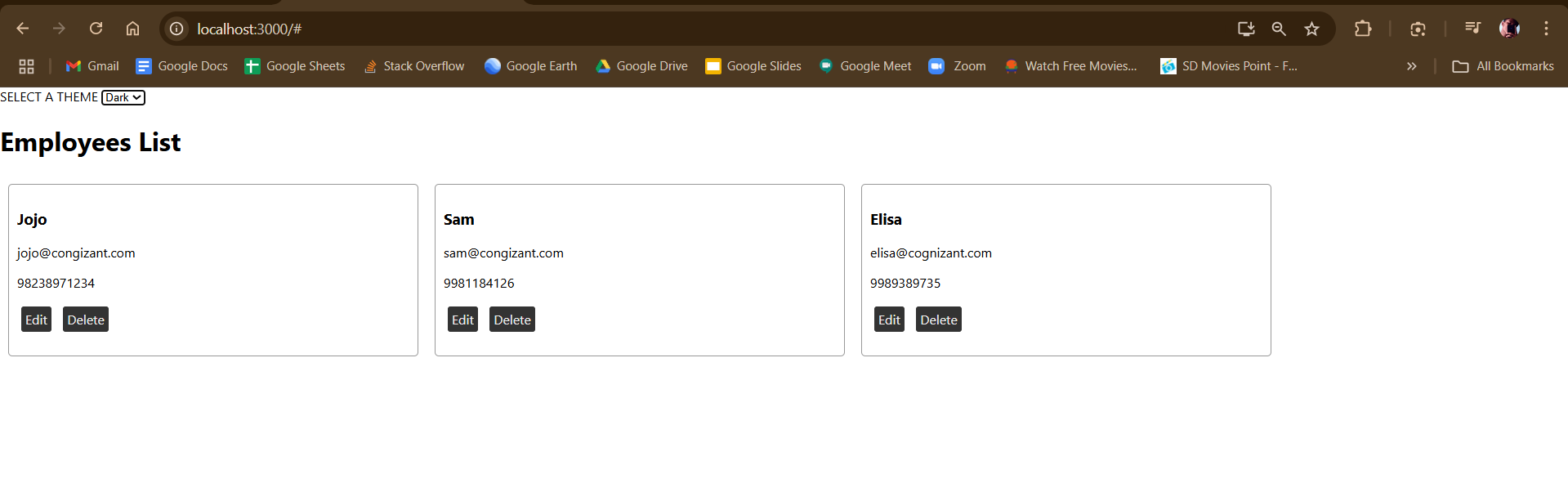
Step 7: C:\Users\ayush\Downloads/employeesapp>npm start



FOR LIGHT THEME.



FOR DARK THEME



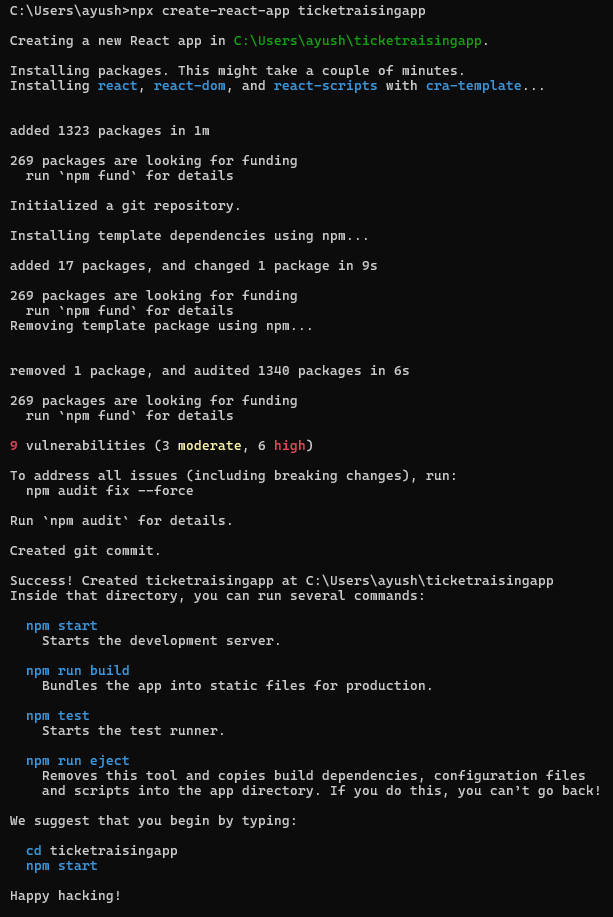
**EXERCISE 7:**

**Create a React App named “ticketraisingapp” which will help to raise a complaint and get it resolved.**

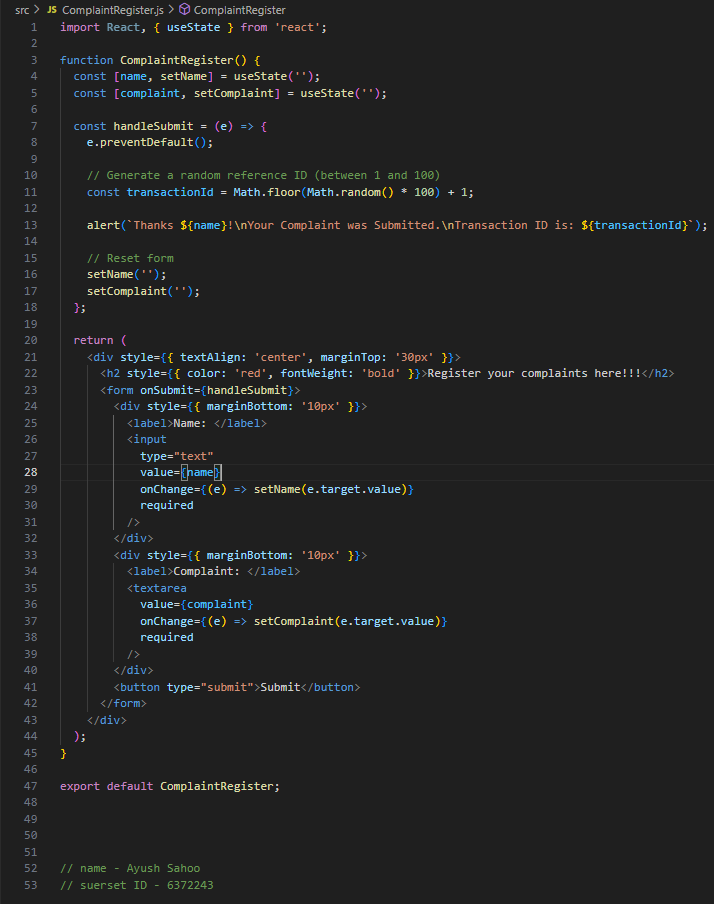
**Create a component named “ComplaintRegister” with a form containing a textbox to enter the employee name and a textarea to enter the complaint. Use “handleSubmit” event of the button to submit the complaint and generate a Reference number for further follow ups in the alert box.**

**SOLUTION:**

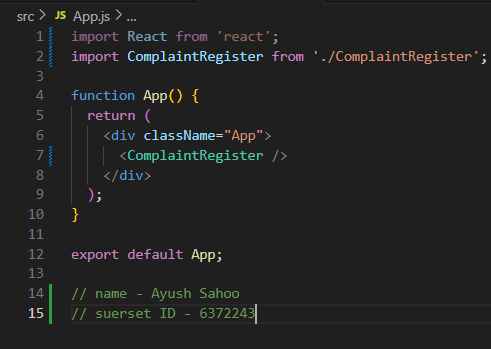
Step 1: C:\Users\ayush>npx create-react-app ticketraisingapp



Step 2: Create “ComplaintRegister.js” under src folder.

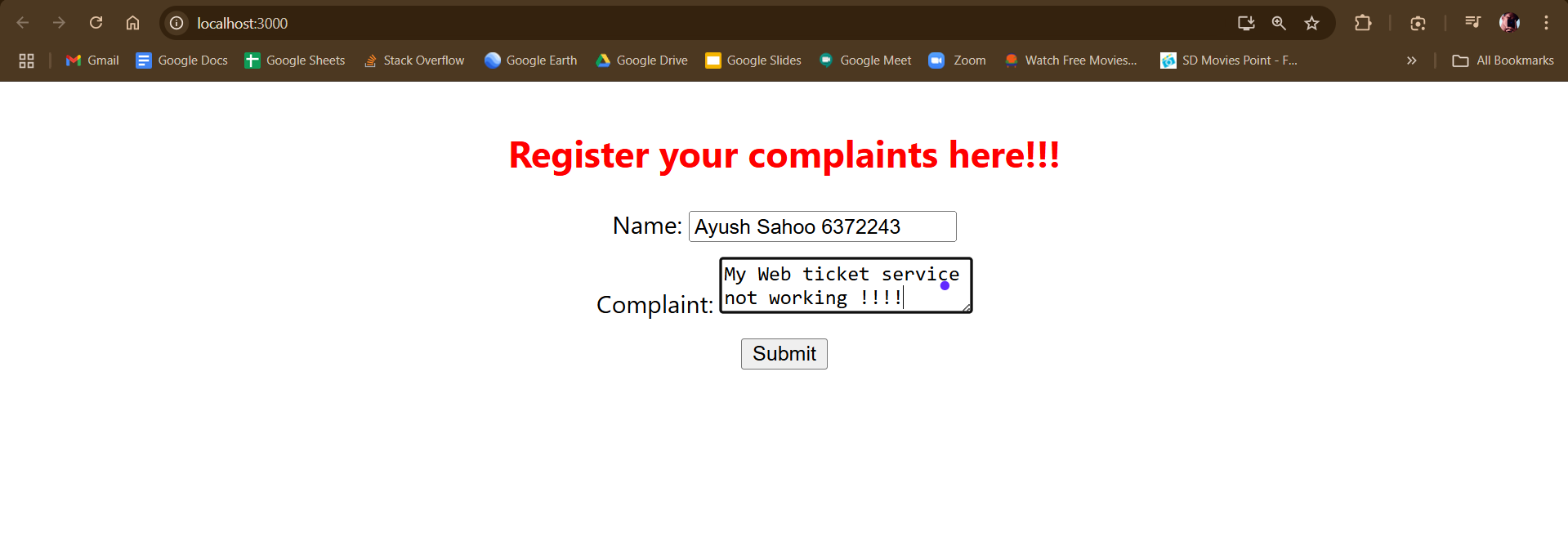


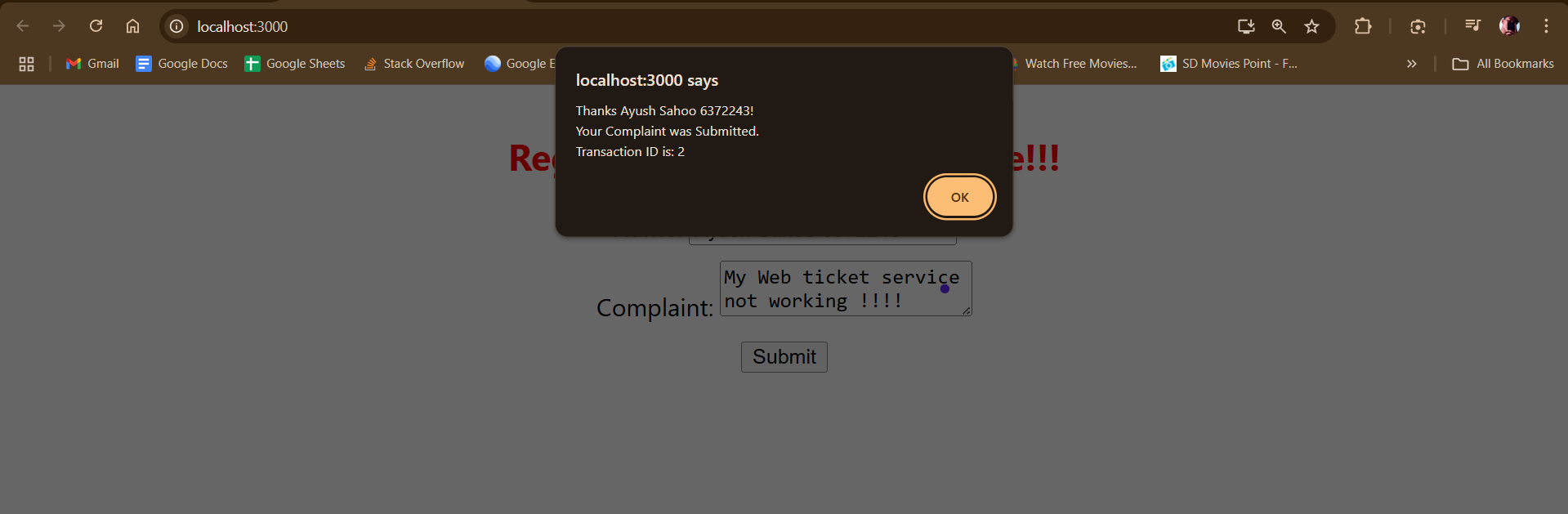
Step 3: update “App.js” under src folder.



Step 4: C:\Users\ayush\ticketraisingapp>npm start







**EXERCISE 8:**

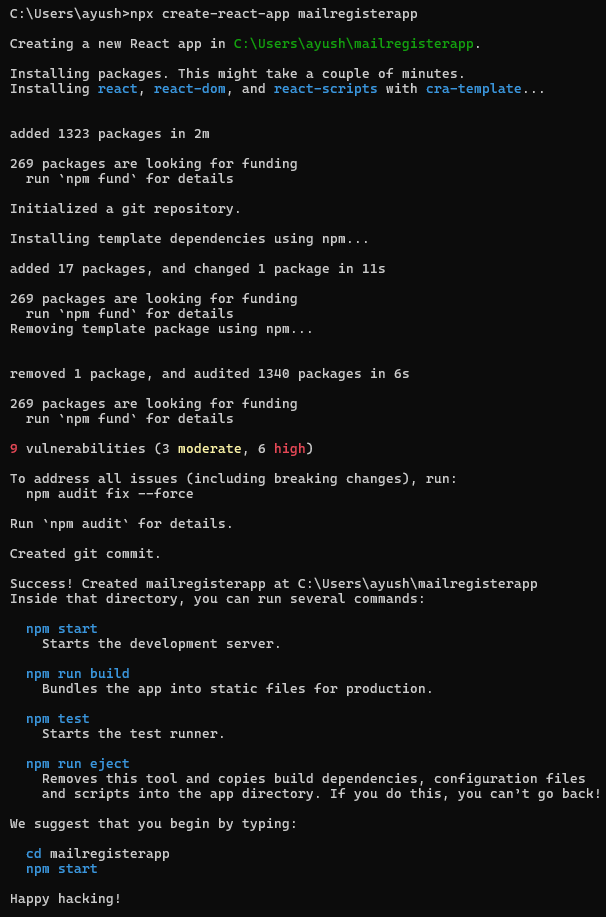
**Create a React App named “mailregisterapp” which will have a component named “register.js”. Create a form which accepts the name, email and password and validate the fields as per the following:**

1. **Name should have atleast 5 characters**
2. **Email should have @ and .**
3. **Password should have atleast 8 characters.**

**Ensure that validations are implemented through eventhandle and eventsubmit of a form.**

**SOLUTION:**

Step 1: C:\Users\ayush>npx create-react-app mailregisterapp

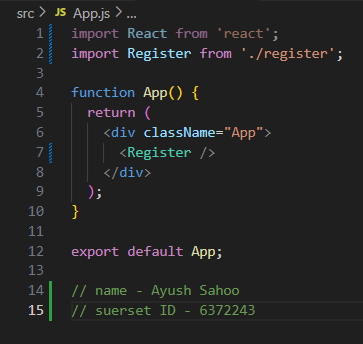


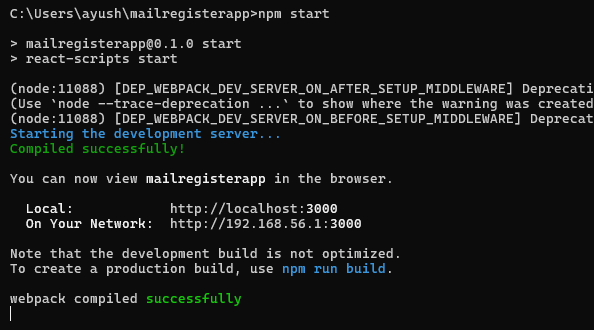
Step 2: Create a “Register.js” under src folder.





Step 3: Update “App.js” under src folder.

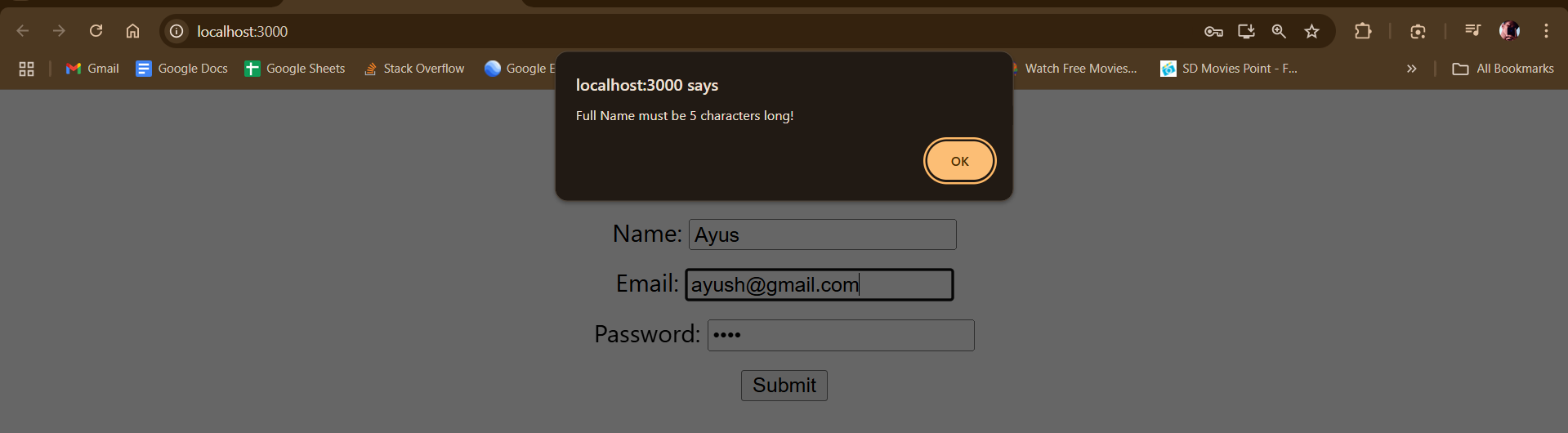


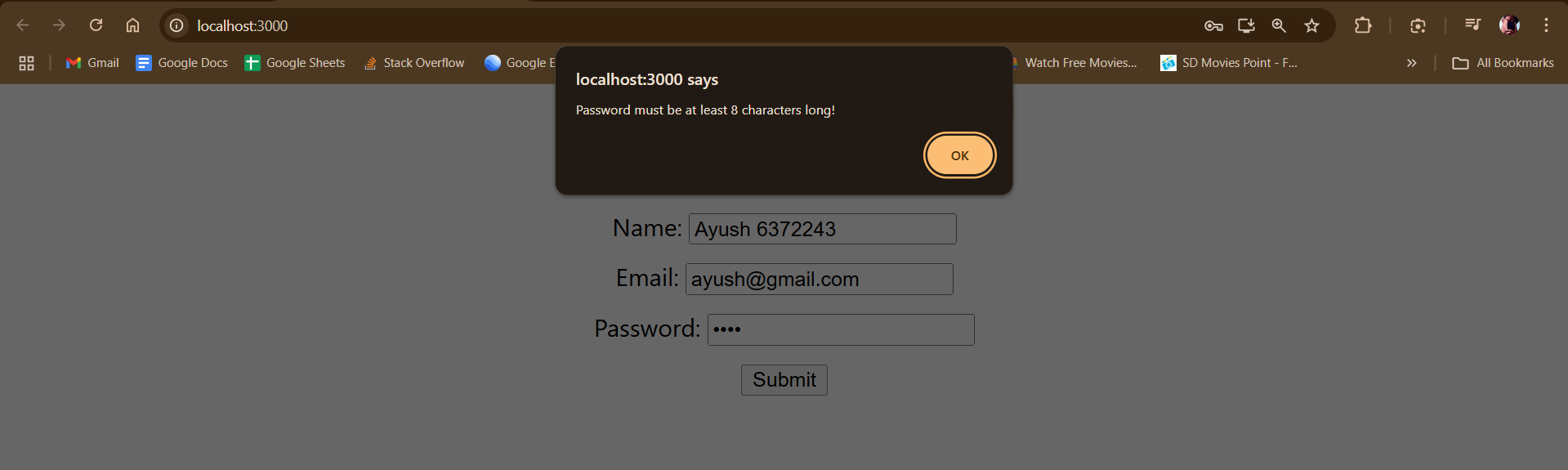
Step 4: C:\Users\ayush\mailregisterapp>npm start 

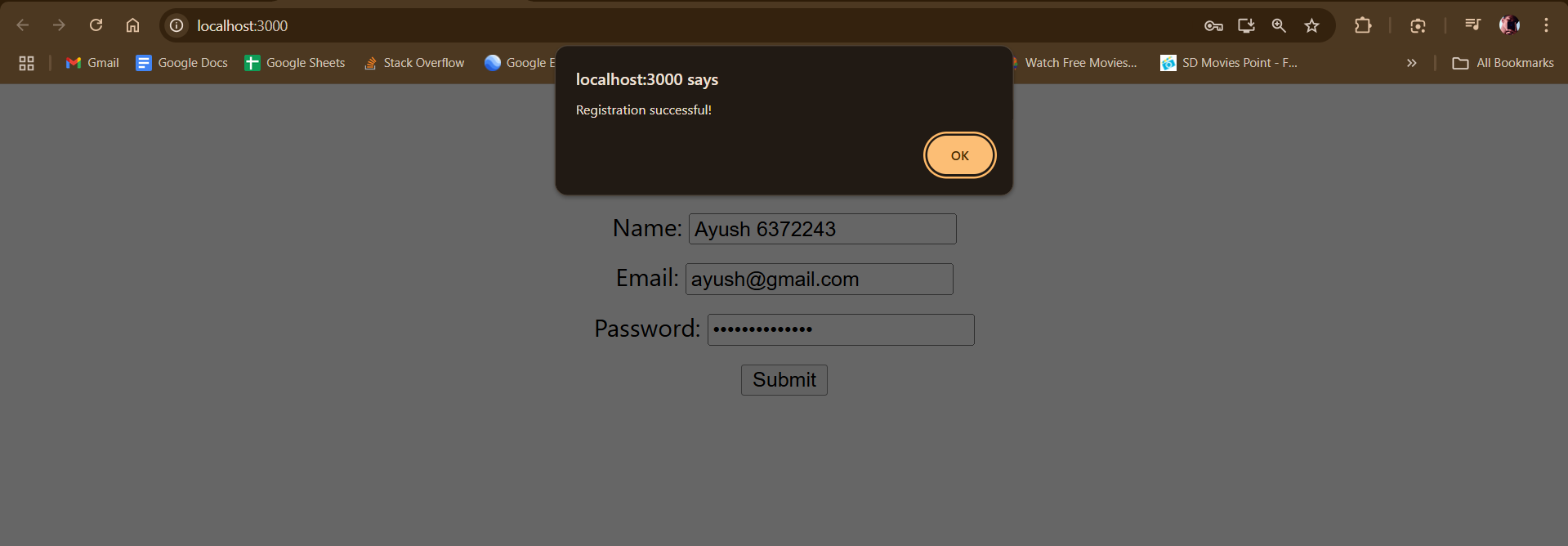
FOR EMAIL : USE OF “@” IS NECESSARY.



FOR FULL NAME : MINIMUM LENGTH SHOULD BE 5.



FOR PASSWORD : MORE THAN LENGTH OF 8 IS REQUIRED.

AFTER COMPLETING ALL THE STEPS THE REGISTRATION HAS BEEN COMPLETED.

**EXERCISE 9:**

**Create a React Application “fetchuserapp” which will retrieve the user details from** [**https://api.randomuser.me/**](https://api.randomuser.me/) **and display the title, firstname and image of a user.**

**Create a component named “Getuser” and in the asynchronous method “ComponentDidMount ()” invoke the URL using fetch method and the response can be displayed in the render method of the component.**

**SOLUTION:**

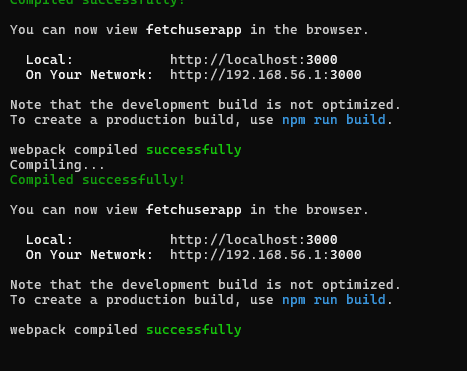
Step 1: C:\Users\ayush>npx create-react-app fetchuserapp



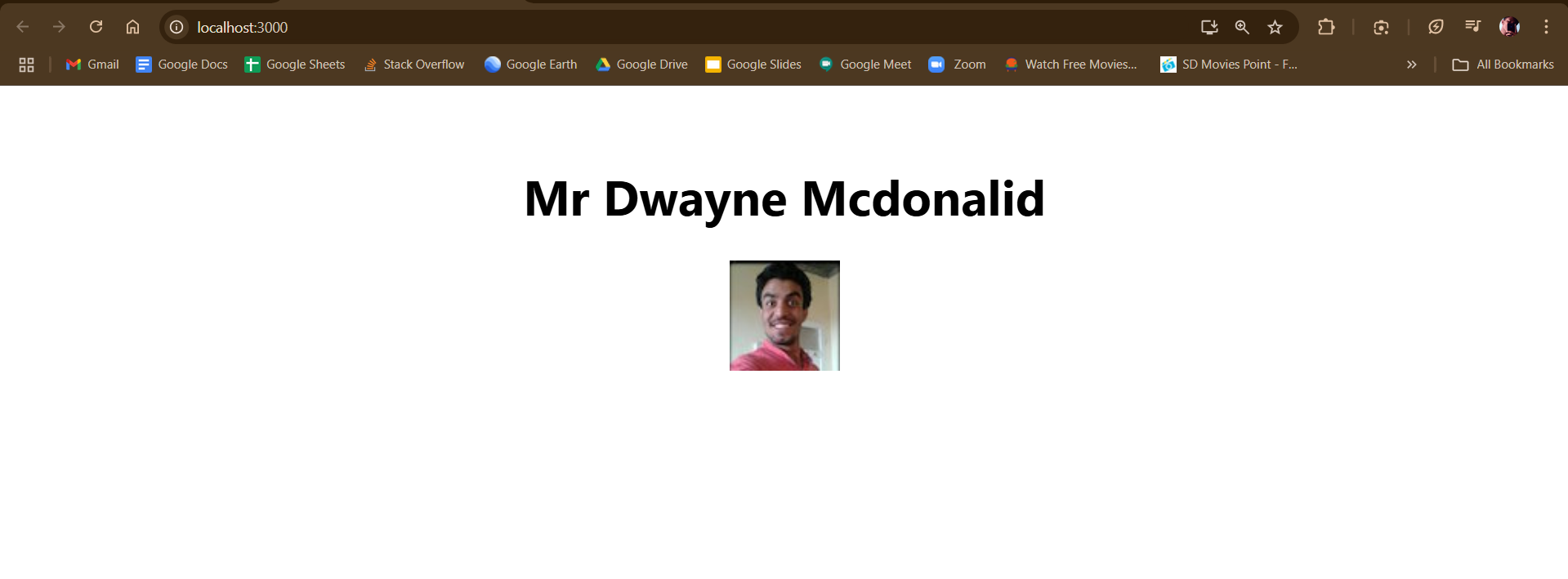
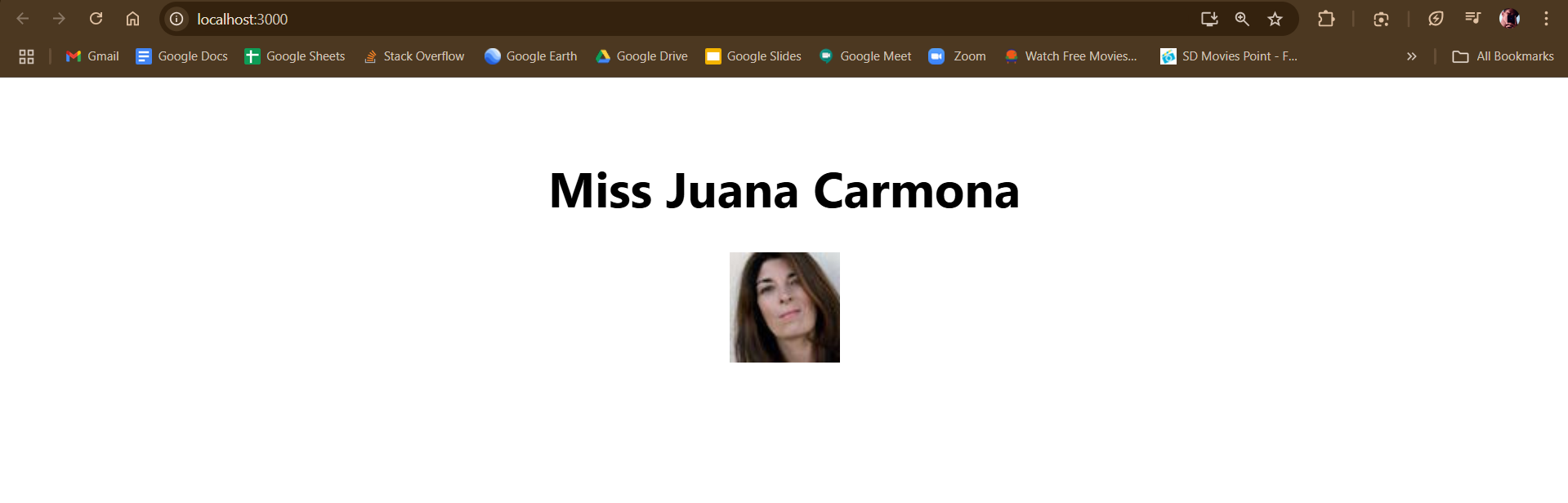
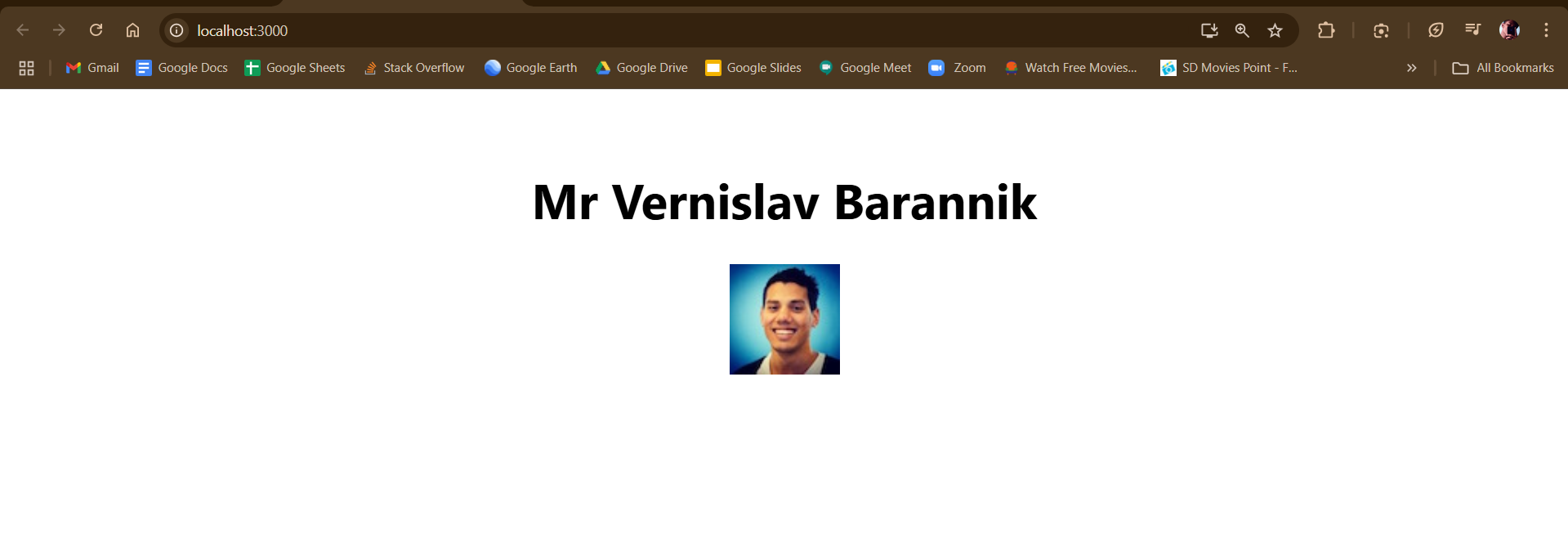
Step 2: Create a folder under src “components” and under that create a file name “Getuser.js”



Step 3: C:\Users\ayush\fetchuserapp>npm start



Now after clicking refresh each time it will show a random name and image of that person based on the API given in the question.

**x----x----x**