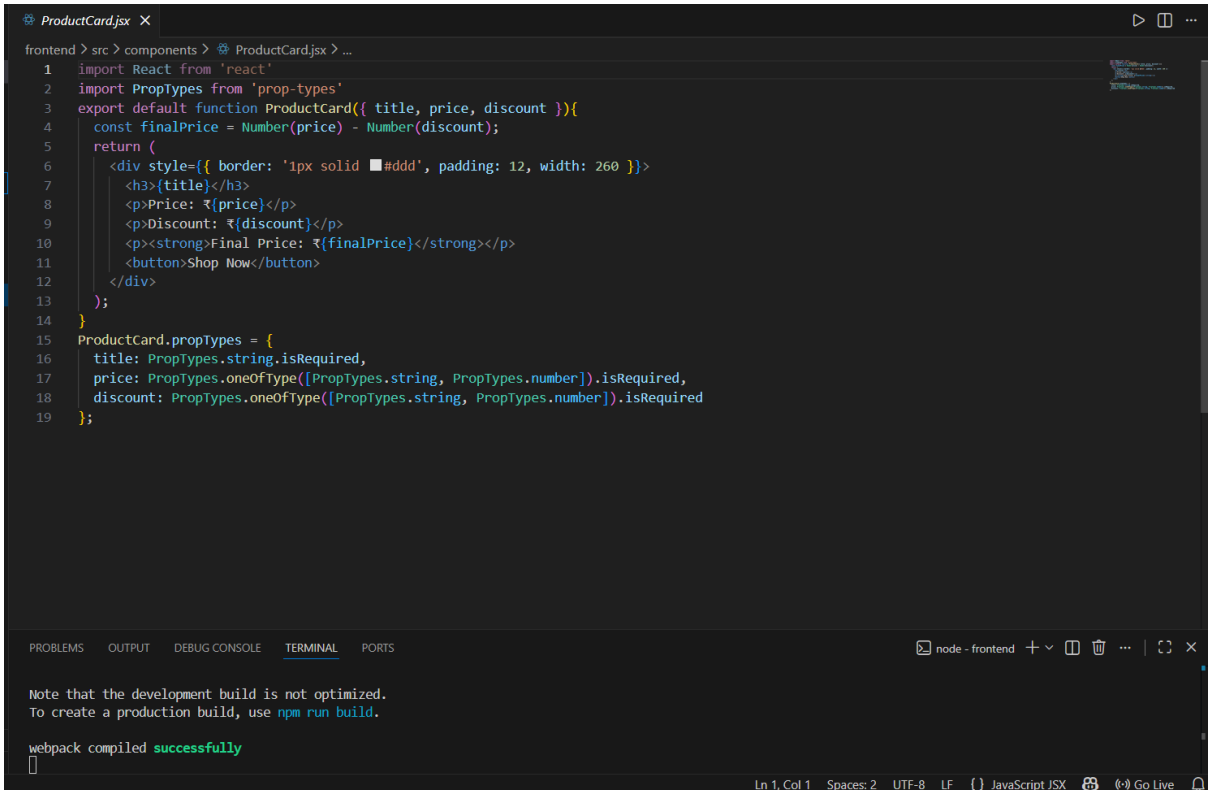
Q-1 React Basics (JSX, Components, Props)

Product Card



ShopNow

Home

Sneakers

Price: ₹120

Discount: ₹20

Final Price: ₹100

Shop Now

Q-2 React State + Controlled and Uncontrolled Components

LoginForm.jsx

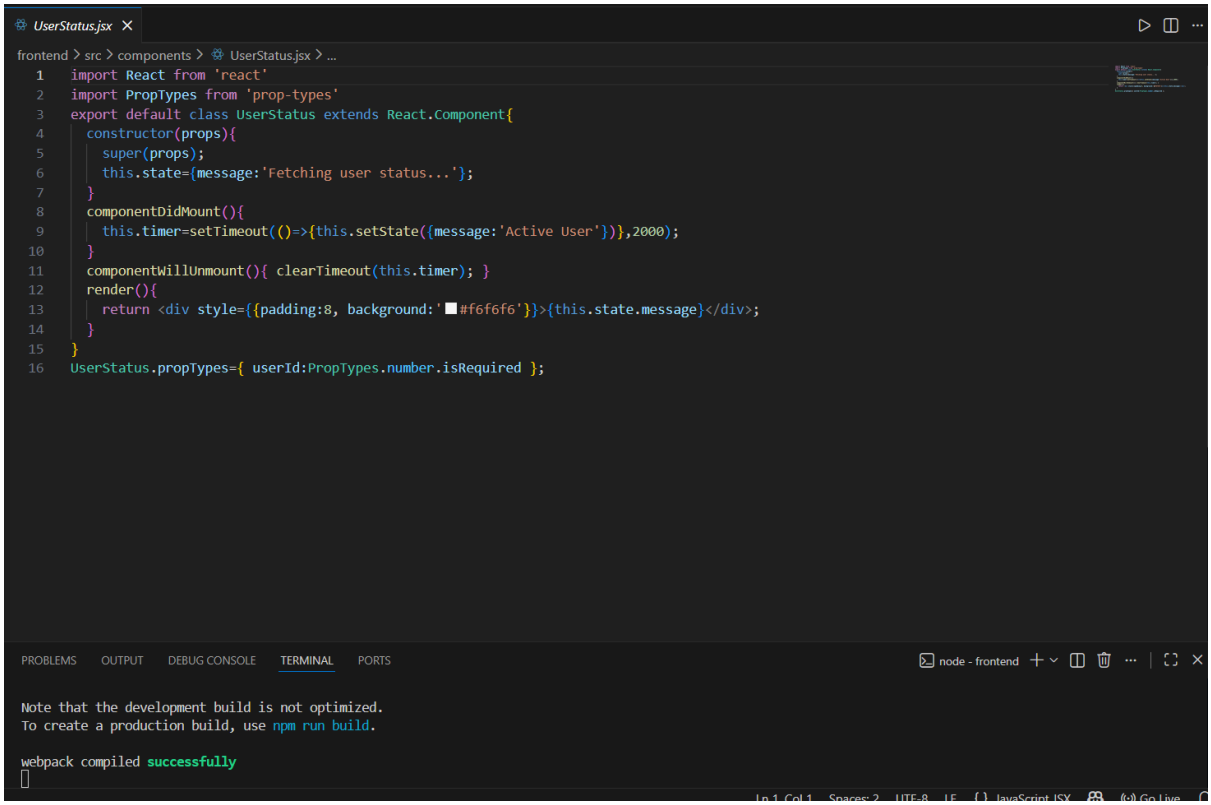
frontend > src > components > LoginForm.jsx > LoginForm

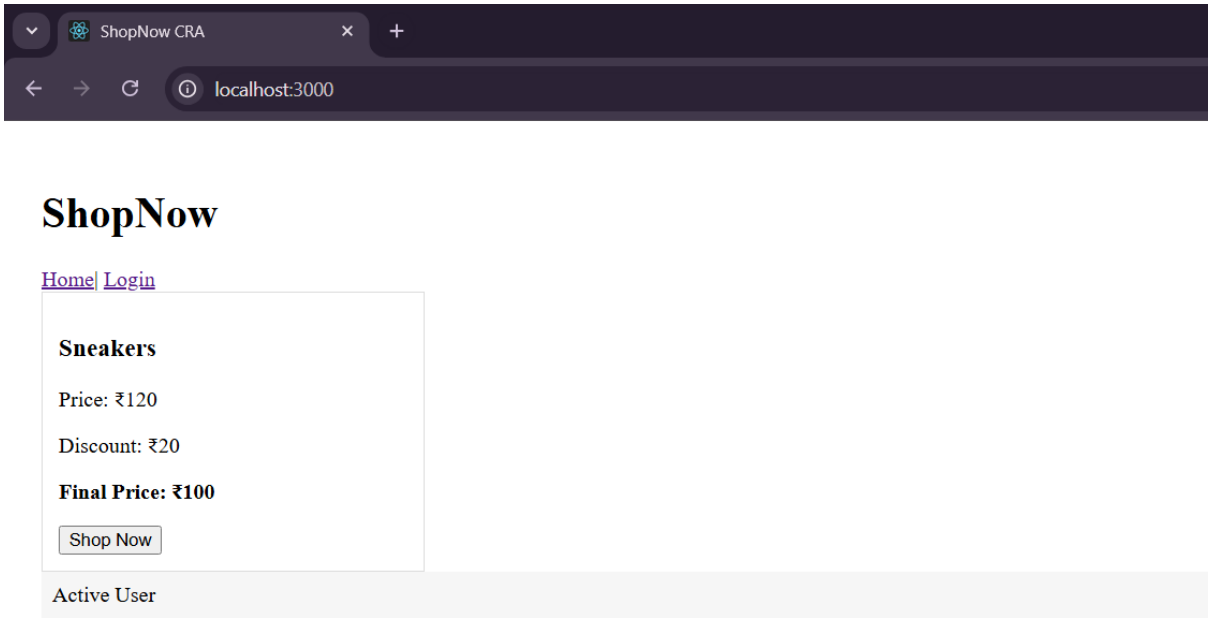
```jsx
import React, { useState, useRef } from 'react'
export default function LoginForm(){
  const [username, setUsername] = useState('');
  const passwordRef = useRef();
  function handleSubmit(e){
    e.preventDefault();
    console.log('username:', username);
    console.log('password:', passwordRef.current.value);
    alert('Logged (check console)');
  }
  return (
    <form onSubmit={handleSubmit} style={{ maxWidth: 360 }}>
      <div>
        <label>Username</label>
        <input value={username} onChange={e => setUsername(e.target.value)} />
      </div>
      <div>
        <label>Password</label>
        <input type="password" ref={passwordRef} />
      </div>
      <button type="submit">Submit</button>
    </form>
  );
}
```

ShopNow

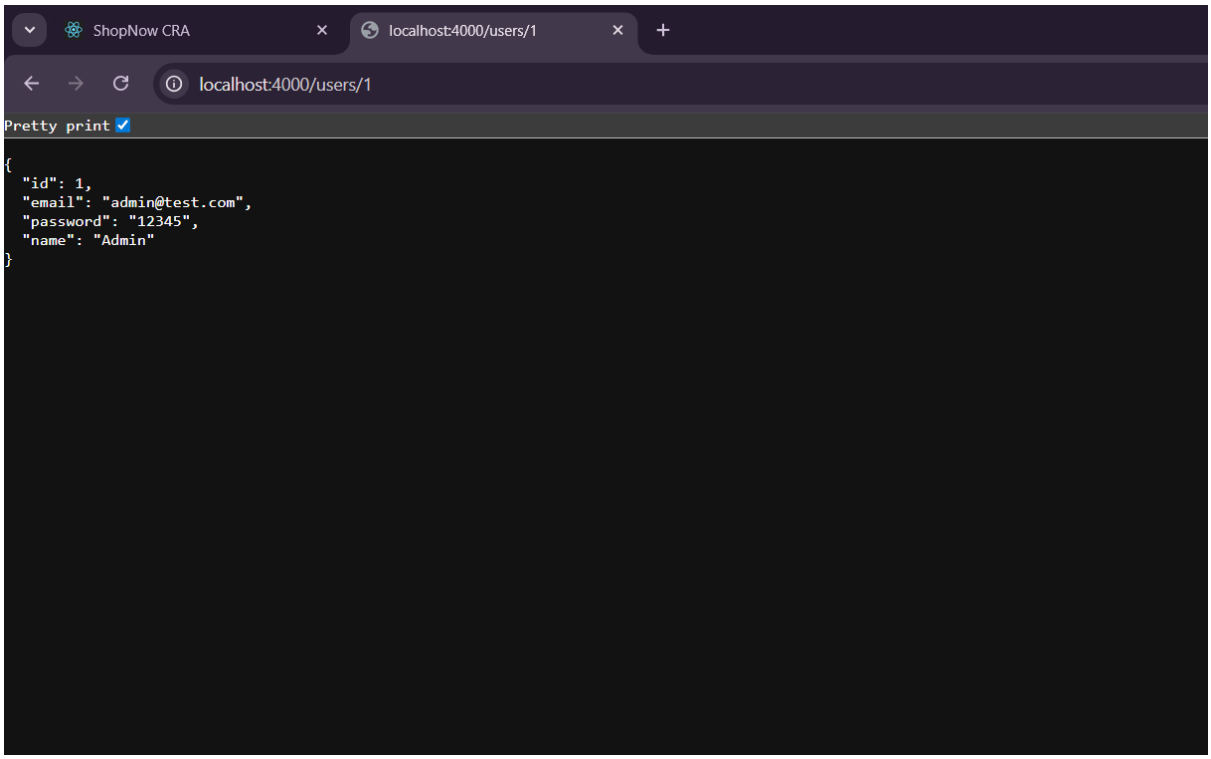
Home| Login
Username
Password
Submit

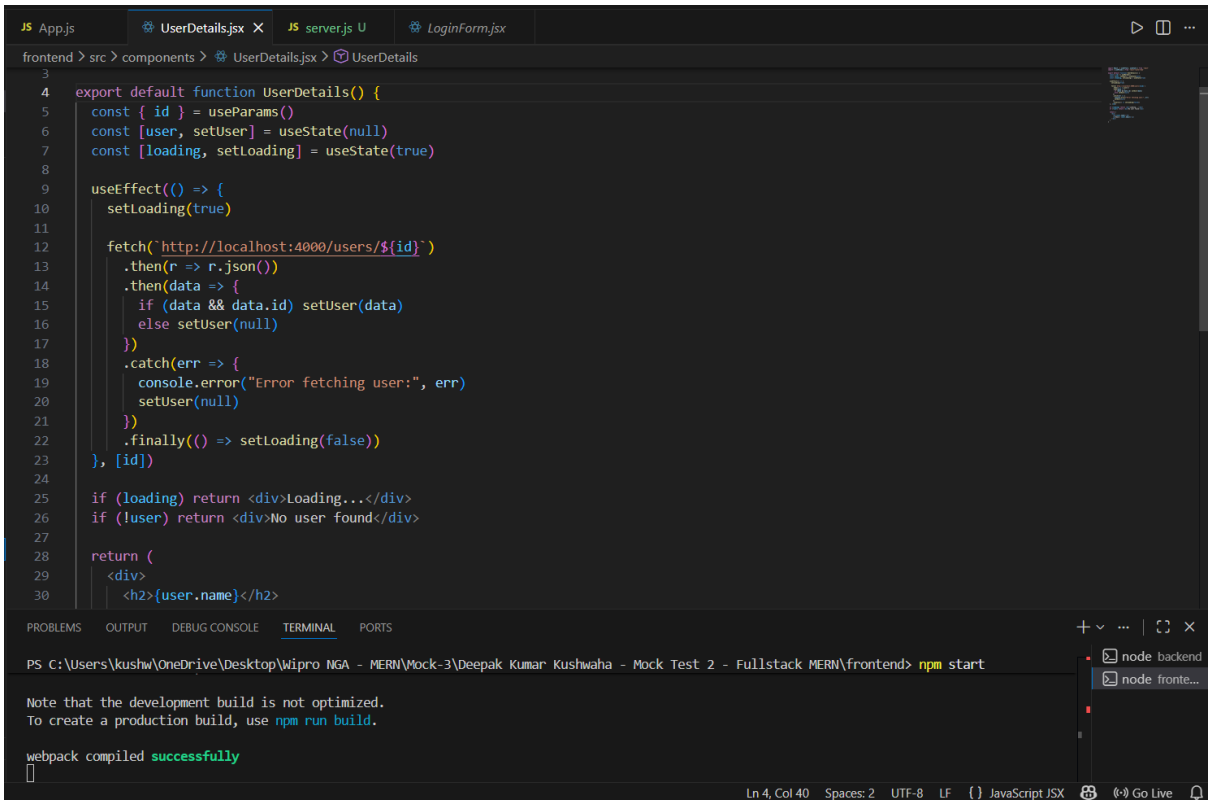
Q-3 React Class Component, Lifecycle, PropTypes, Styling





Q-4 React Router + API Integration





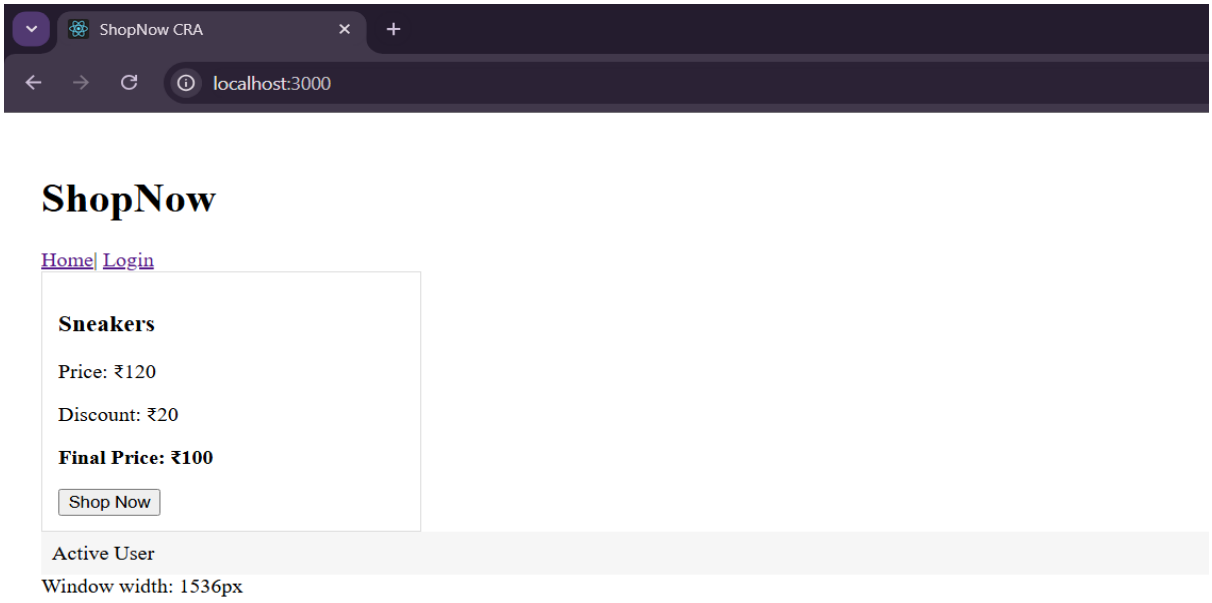
ShopNow

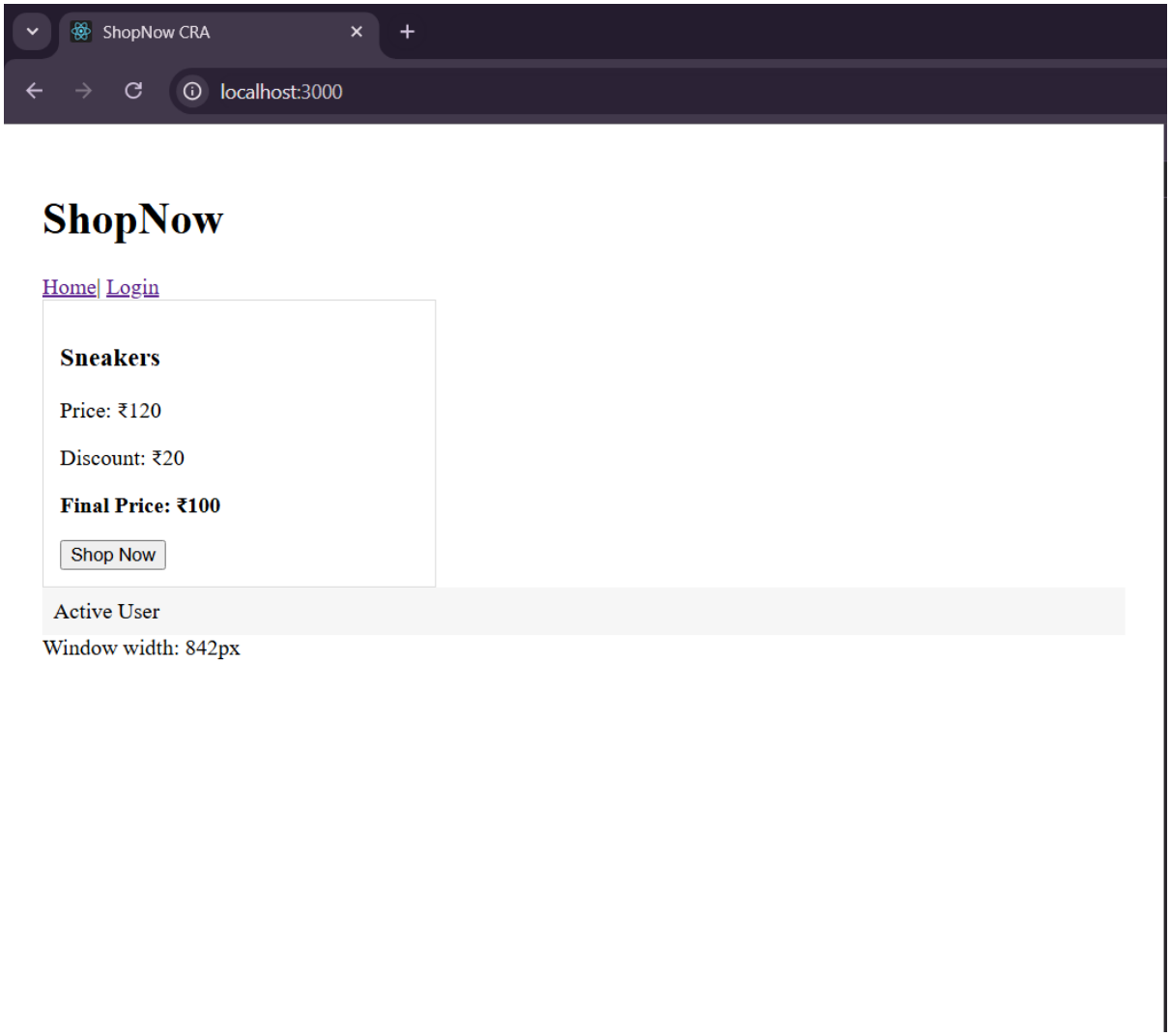
Home| Login

Admin

Email: admin@test.com

Q-5 Reusability Using HOC or Render Props




ShopNow CRA
localhost:3000
ShopNow
Home| Login
Sneakers
Price: ₹120
Discount: ₹20
Final Price: ₹100
Shop Now
Active User
Window width: 842px

Q-6 Formik + Yup Validation

App.js | FormikLogin.jsx | UserDetails.jsx | server.js U

frontend > src > components > FormikLogin.jsx > ...

```jsx
import React from 'react'
import { Formik, Field, Form, ErrorMessage } from 'formik'
import * as Yup from 'yup'
const schema=Yup.object().shape({
  email:Yup.string().email().required(),
  password:Yup.string().min(6).required()
});
export default function FormikLogin(){
  return (
    <Formik initialValues={{email:'',password:''}} validationSchema={schema}
      onSubmit={v=>{console.log(v);alert('Submitted')}}>
      <Form>
        <div><label>Email</label><Field name="email" /><ErrorMessage name="email" /></div>
        <div><label>Password</label><Field name="password" type="password" /><ErrorMessage name="password" /></div>
        <button type="submit">Login</button>
      </Form>
    </Formik>
  );
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\kushw\OneDrive\Desktop\Wipro NGA - MERN\Mock-3\Deepak Kumar Kushwaha - Mock Test 2 - Fullstack MERN\frontend> npm start

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

node backe
node fronte

ShopNow

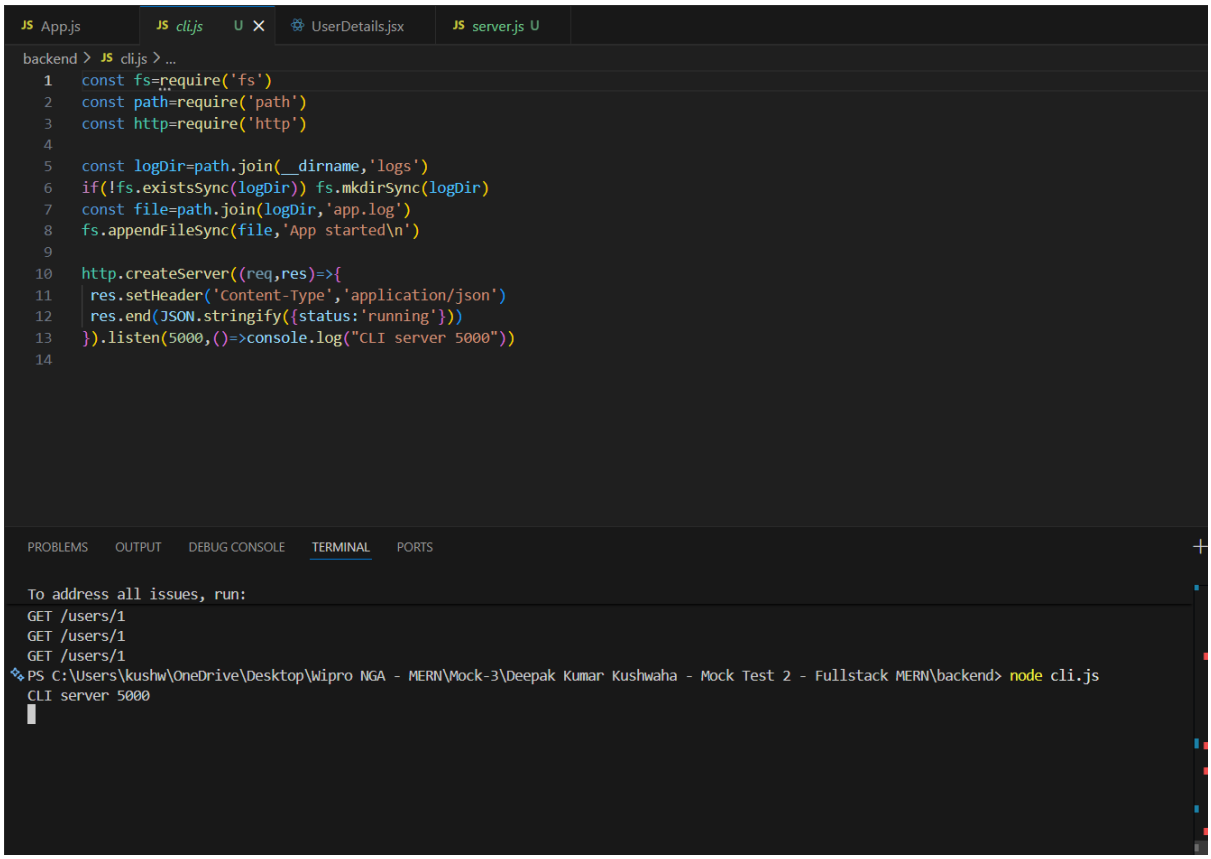
Home| Login| Formik

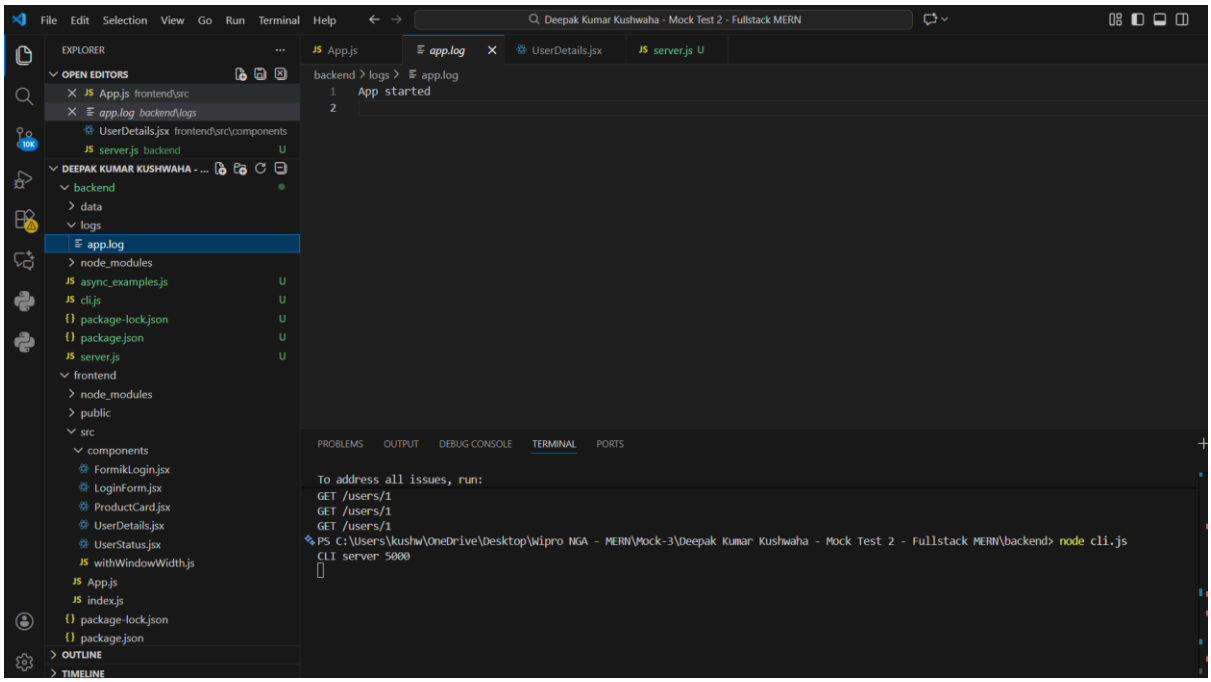
Email abx email must be a valid email

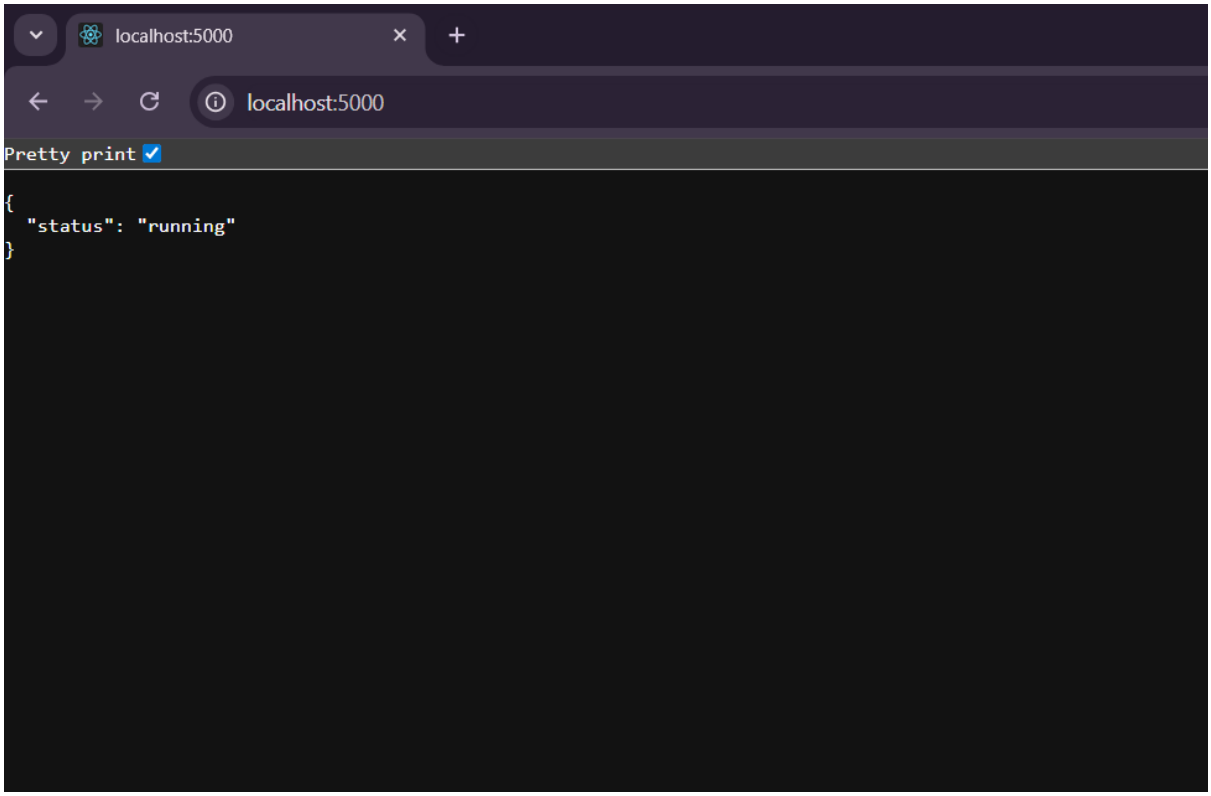
Password password is a required field

Login

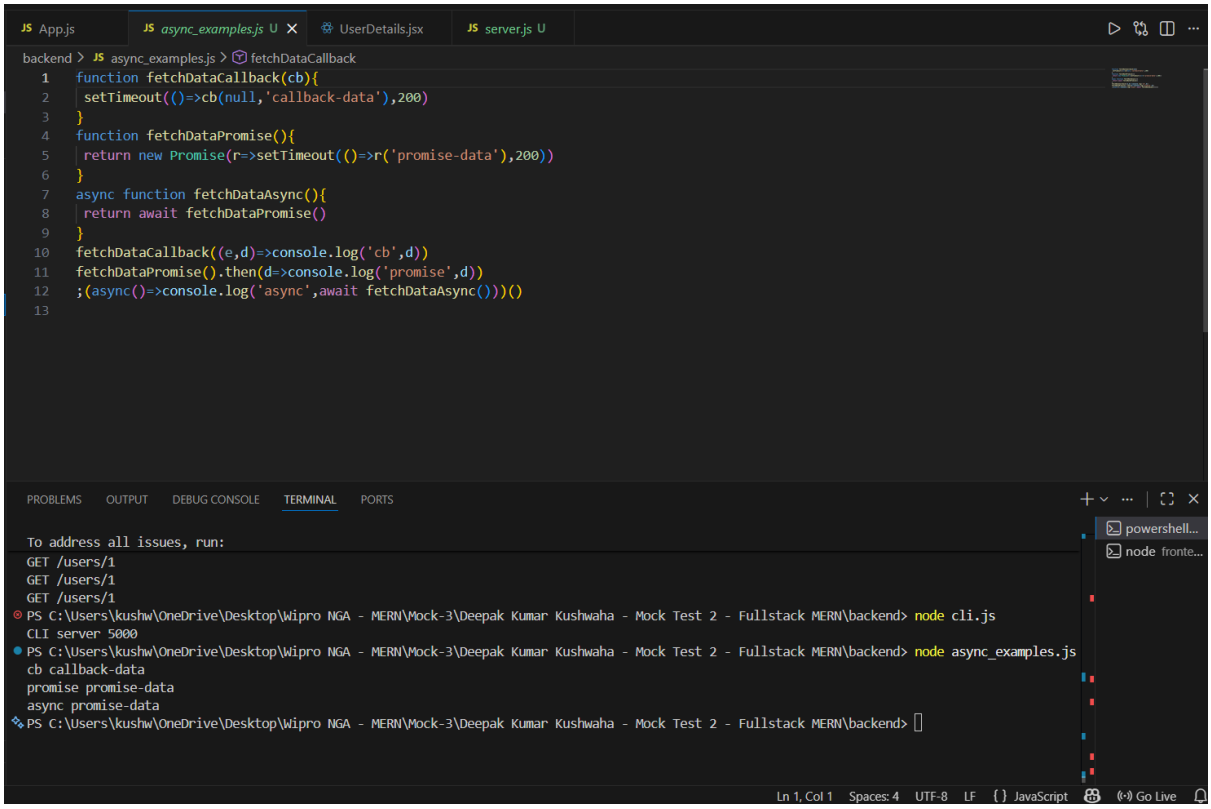
Q-7 Node.js Core Modules






localhost:5000
localhost:5000
Pretty print
{
"status": "running"
}

Q-8 Asynchronous JavaScript (Callbacks → Promise → Async/Await)



Q-9 Express Routing + Middleware + Validation

App.js | products.json U | UserDetails.jsx | server.js U

backend > server.js > app.use() callback

```
6
7   const app = express()
8   app.use(cors());
9   app.use(bodyParser.json())
10
11  app.use((req,res,next)=>{console.log(req.method, req.url); next();})
12
13  let products = [ { id:1, name:'Sneakers', price:120}, {id:2,name:'Boots',price:150}]
14
15  app.get('/products',(req,res)=>res.json(products))
16
17  app.post('/products',[
18   body('name').isString().notEmpty(),
19   body('price').isNumeric()
20  ],(req,res)=>{
21   const errors = validationResult(req)
22   if(!errors.isEmpty()) return res.status(400).json({errors:errors.array()})
23   const p={id:products.length+1, ...req.body}
24   products.push(p)
25   res.status(201).json(p)
26  })
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\kushw\OneDrive\Desktop\Wipro NGA - MERN\Mock-3\Deepak Kumar Kushwaha - Mock Test 2 - Fullstack MERN\backend> node cli.js
CLI server 5000
PS C:\Users\kushw\OneDrive\Desktop\Wipro NGA - MERN\Mock-3\Deepak Kumar Kushwaha - Mock Test 2 - Fullstack MERN\backend> node async_examples.js
cb callback-data
promise promise-data
async promise-data
PS C:\Users\kushw\OneDrive\Desktop\Wipro NGA - MERN\Mock-3\Deepak Kumar Kushwaha - Mock Test 2 - Fullstack MERN\backend> no
                                                                                                                        node server.js
Backend running 4000
PS C:\Users\kushw\OneDrive\Desktop\Wipro NGA - MERN\Mock-3\Deepak Kumar Kushwaha - Mock Test 2 - Fullstack MERN\backend> node server.js
Backend running 4000
GET /
GET /products
```

http://localhost:4000/products Save

GET http://localhost:4000/products Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

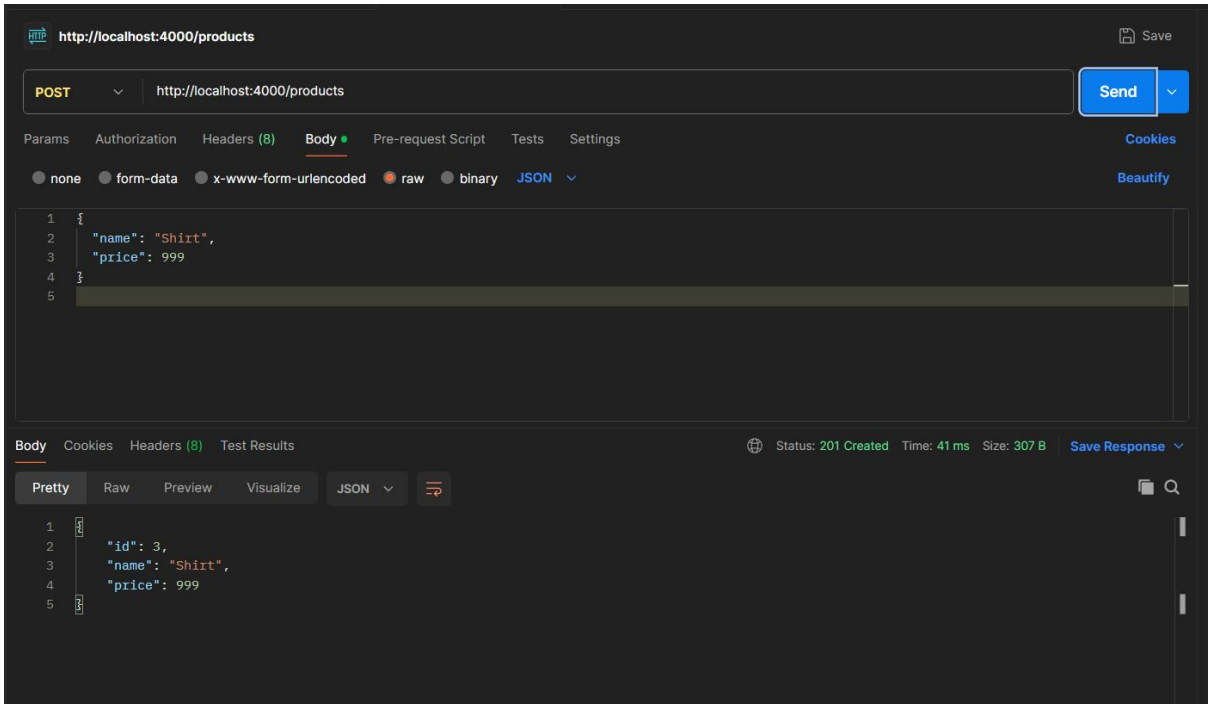
Query Params

Key	Value	Bulk Edit
Key	Value	

Body Cookies Headers (8) Test Results Status: 200 OK Time: 20 ms Size: 343 B Save Response

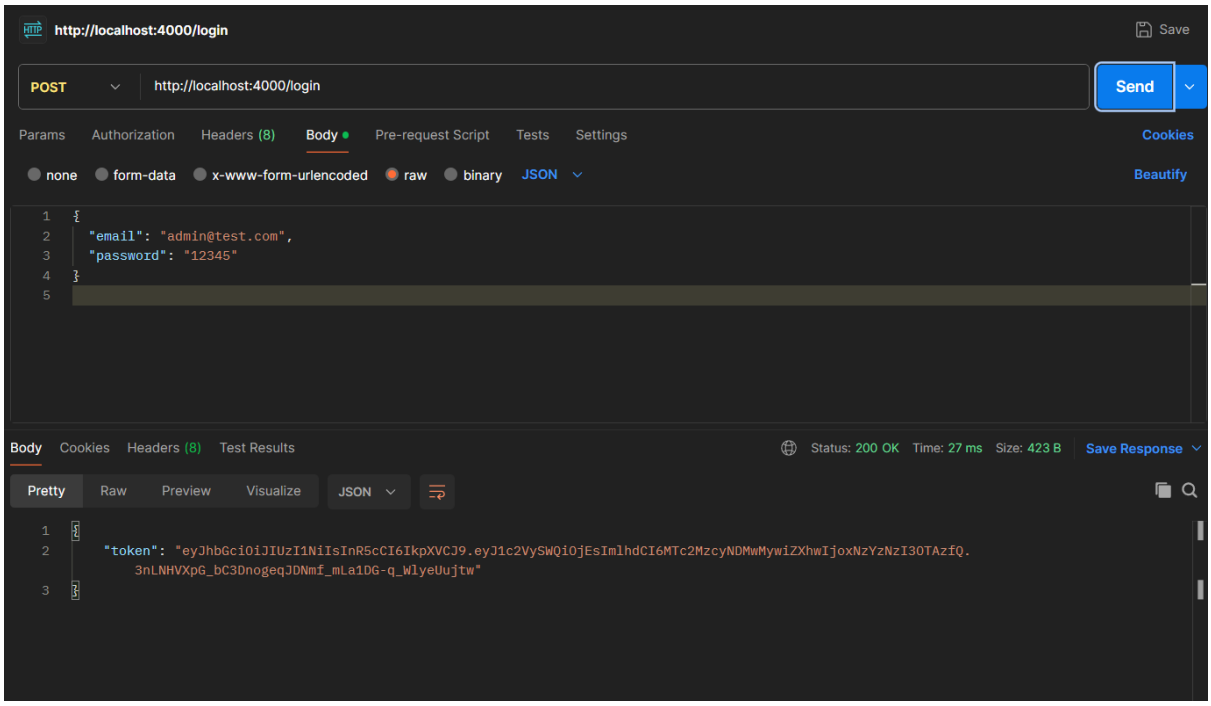
Pretty Raw Preview Visualize JSON

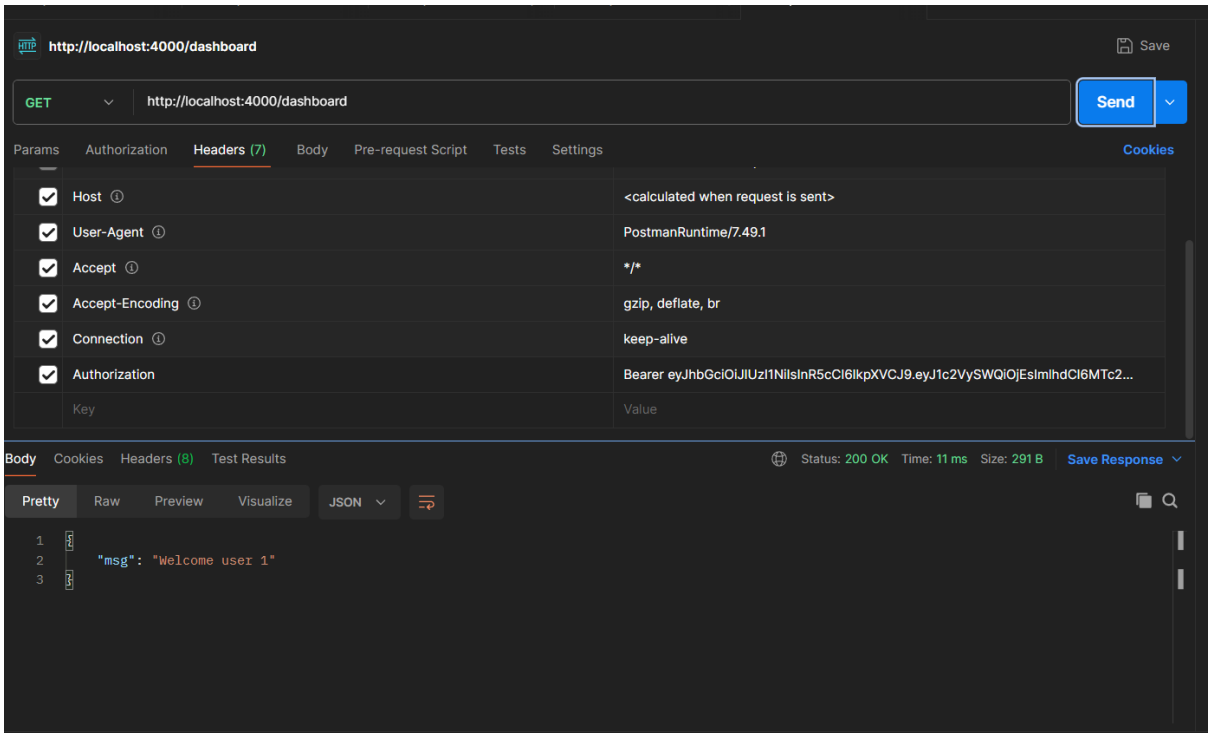

```
[
    {
        "id": 1,
        "name": "Sneakers",
        "price": 120
    },
    {
        "id": 2,
        "name": "Boots",
        "price": 150
    }
]
```

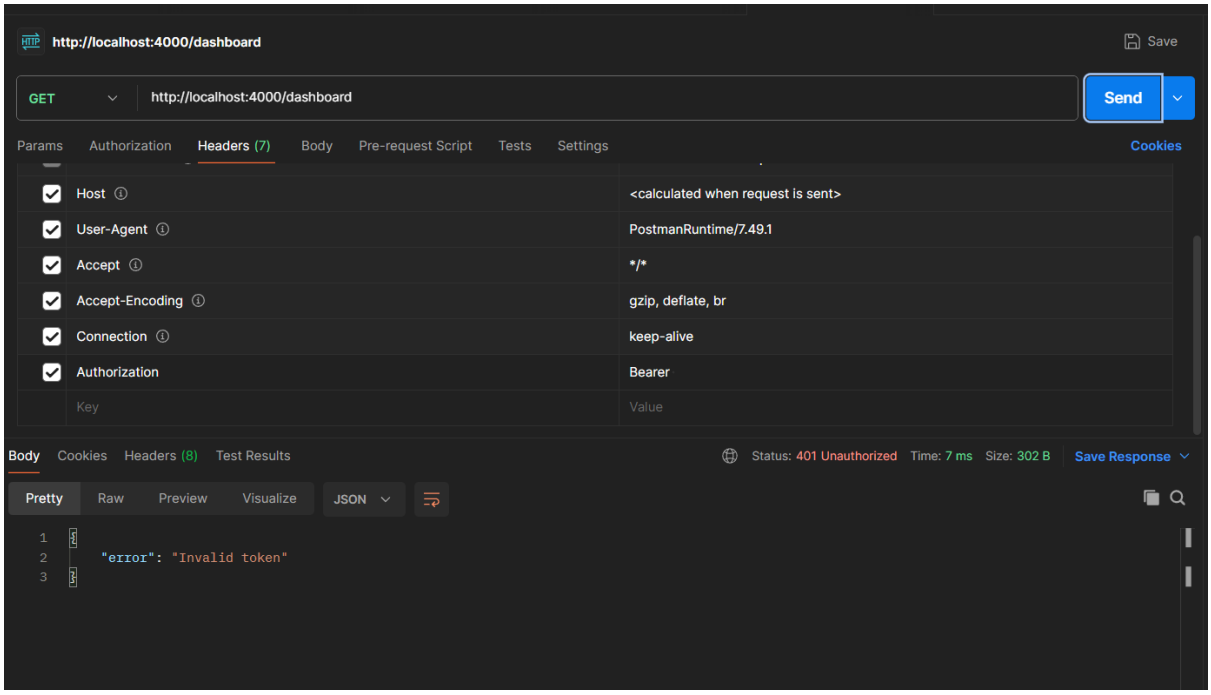



```
Backend running 4000
PS C:\Users\kushw\OneDrive\Desktop\Wipro NGA - MERN\Mock-3\Deepak Kumar Kushwaha - Mock Test 2 - Fullstack MERN\backend> node server.js
Backend running 4000
GET /
GET /products
GET /products
POST /products
```

Q-10 REST API + JWT Authentication






http://localhost:4000/dashboard
Save
GET
http://localhost:4000/dashboard
Send
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Settings
Cookies
Host
<calculated when request is sent>
User-Agent
PostmanRuntime/7.49.1
Accept
/
Accept-Encoding
gzip, deflate, br
Connection
keep-alive
Authorization
Bearer
Key
Value
Body
Cookies
Headers (8)
Test Results
Status: 401 Unauthorized
Time: 7 ms
Size: 302 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
1 {
2 "error": "Invalid token"
3 }