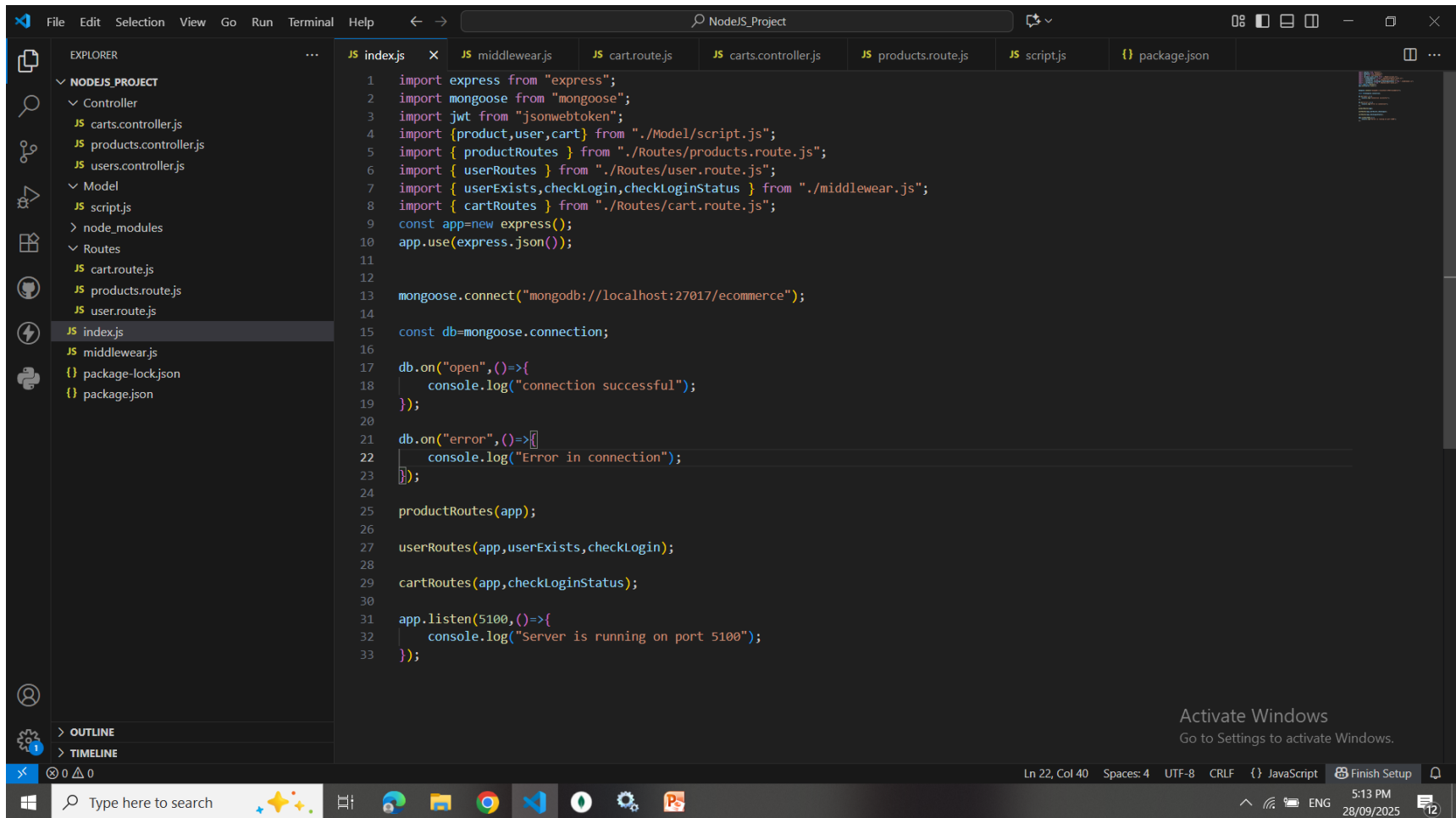


Code:



The screenshot shows the Visual Studio Code interface with a project named 'NodeJS_Project'. The Explorer sidebar on the left shows the file structure: 'NODEJS_PROJECT' containing 'Controller' (with 'carts.controller.js', 'products.controller.js', and 'users.controller.js'), 'Model' (with 'script.js'), 'Routes' (with 'cart.route.js', 'products.route.js', and 'user.route.js'), 'node_modules', 'package-lock.json', and 'package.json'. The main editor window displays the code for 'index.js'.

```
1 import express from "express";
2 import mongoose from "mongoose";
3 import jwt from "jsonwebtoken";
4 import { product,user,cart } from "../Model/script.js";
5 import { productRoutes } from "../Routes/products.route.js";
6 import { userRoutes } from "../Routes/user.route.js";
7 import { userExists,checkLogin,checkLoginStatus } from "../middleware.js";
8 import { cartRoutes } from "../Routes/cart.route.js";
9 const app=new express();
10 app.use(express.json());
11
12
13 mongoose.connect("mongodb://localhost:27017/ecommerce");
14
15 const db=mongoose.connection;
16
17 db.on("open",()=>{
18   console.log("connection successful");
19 });
20
21 db.on("error",()=>{
22   console.log("Error in connection");
23 });
24
25 productRoutes(app);
26
27 userRoutes(app,userExists,checkLogin);
28
29 cartRoutes(app,checkLoginStatus);
30
31 app.listen(5100,()=>{
32   console.log("Server is running on port 5100");
33 });
```

The status bar at the bottom indicates the current position is 'Ln 22, Col 40', with 'Spaces: 4', 'UTF-8', 'CRLF', and 'JavaScript' file type. A notification for 'Activate Windows' is visible in the bottom right corner.

Post request “/product”:

The screenshot displays the VS Code interface with the Thunder Client extension. A POST request is configured to `http://www.localhost:5100/product`. The request body is a JSON object: `{ "productId": "PI001", "title": "iPhone 16", "brand": "Apple", "stockQuantity": 100, "price": 100000 }`. The response status is 200 OK, with a size of 43 Bytes and a time of 620 ms. The response body is: `{ "message": "new product added in database" }`. The terminal shows the command `npm start` being executed, with output indicating that the server is running on port 5100.

THUNDER CLIENT

New Request

Activity Collections Env

filter activity

POST thunderclient.com/product just now

POST localhost:5100/register 22 hours ago

GET localhost:5100/register 5 days ago

POST localhost:5100/register 14 days ago

GET localhost:5100/products 16 days ago

PUT localhost:5100/users 28 days ago

POST http://www.localhost:5100/product Send

Query Headers Auth Body Tests Pre Run

JSON XML Text Form Form-encode GraphQL Binary

JSON Content Format

```
1 {
2   "productId": "PI001",
3   "title": "iPhone 16",
4   "brand": "Apple",
5   "stockQuantity": 100,
6   "price": 100000
7 }
```

Status: 200 OK Size: 43 Bytes Time: 620 ms

Response Headers Cookies Results Docs

```
1 {
2   "message": "new product added in database"
3 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS F:\visualStudioCode\NodeJS_Project> npm start

```
[nodemon] 3.1.10
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node index.js`
Server is running on port 5100
connection successful
{}

```

Activate Windows
Go to Settings to activate Windows.

main* 0 0 0

Finish Setup Go Live

Type here to search

ENG 5:28 PM 28/09/2025

Get request “/products”:

The screenshot shows the Visual Studio Code interface with a REST client request and its response.

Request:

- Method: GET
- URL: `http://localhost:5100/products`
- Body: JSON Content

Response:

Status: 200 OK Size: 386 Bytes Time: 32 ms

```
1 [
2   {
3     "_id": "68d922d102ceaf7975feffd0",
4     "productId": "PI001",
5     "title": "iPhone 16",
6     "brand": "Apple",
7     "stockQuantity": 100,
8     "price": 100000,
9     "__v": 0
10  },
11  {
12    "_id": "68d9235702ceaf7975feffd3",
13    "productId": "PI002",
14    "title": "Air Conditioner",
15    "brand": "Daikin",
16    "stockQuantity": 50,
17    "price": 40000,
18    "__v": 0
19  }
20 ]
```

Terminal:

```
PS F:\visualStudioCode\NodeJS_Project> npm start
{
  _id: new ObjectId('68d9241b02ceaf7975feffdb'),
  productId: 'PI003',
  title: 'Apples',
  stockQuantity: 1000,
  price: 15,
  __v: 0
}
```

Get request “/product/:id”:

The screenshot shows the Visual Studio Code interface with a REST client tab open. The request is a GET to `http://www.localhost:5100/product/PI002`. The response is a 200 OK status with a JSON body containing product details.

Request:

```
GET http://www.localhost:5100/product/PI002
```

Response:

```
{
  "_id": "68d9235702ceaf7975feffd3",
  "productId": "PI002",
  "title": "Air Conditioner",
  "brand": "Daikin",
  "stockQuantity": 50,
  "price": 40000,
  "__v": 0
}
```

Terminal:

```
PS F:\visualStudioCode\NodeJS_Project> npm start

{
  _id: new ObjectId('68d9241b02ceaf7975feffd3'),
  productId: 'PI003',
  title: 'Apples',
  stockQuantity: 1000,
  price: 15,
  __v: 0
}
```

Post request “/register”:

The screenshot shows the Visual Studio Code interface with a REST client request configured. The Explorer on the left shows the project structure for 'NODEJS_PROJECT', including files like 'index.js', 'middleware.js', and 'script.js'. The main editor area displays a 'New Request' tab with a POST method to 'http://www.localhost:5100/register'. The 'Body' tab is selected, showing a JSON payload. The 'Response' tab on the right shows the server's response, which is a 200 OK status with a JSON object containing user details. The bottom panel shows the terminal with the command 'npm start' executed.

Request:

```
POST http://www.localhost:5100/register
```

JSON Content:

```
1 {
2   "userId": "UI001",
3   "firstName": "Pratik",
4   "lastName": "Mehta",
5   "email": "pratik@gmail.com",
6   "password": "123456789",
7   "mobile": "1234567890",
8 }
```

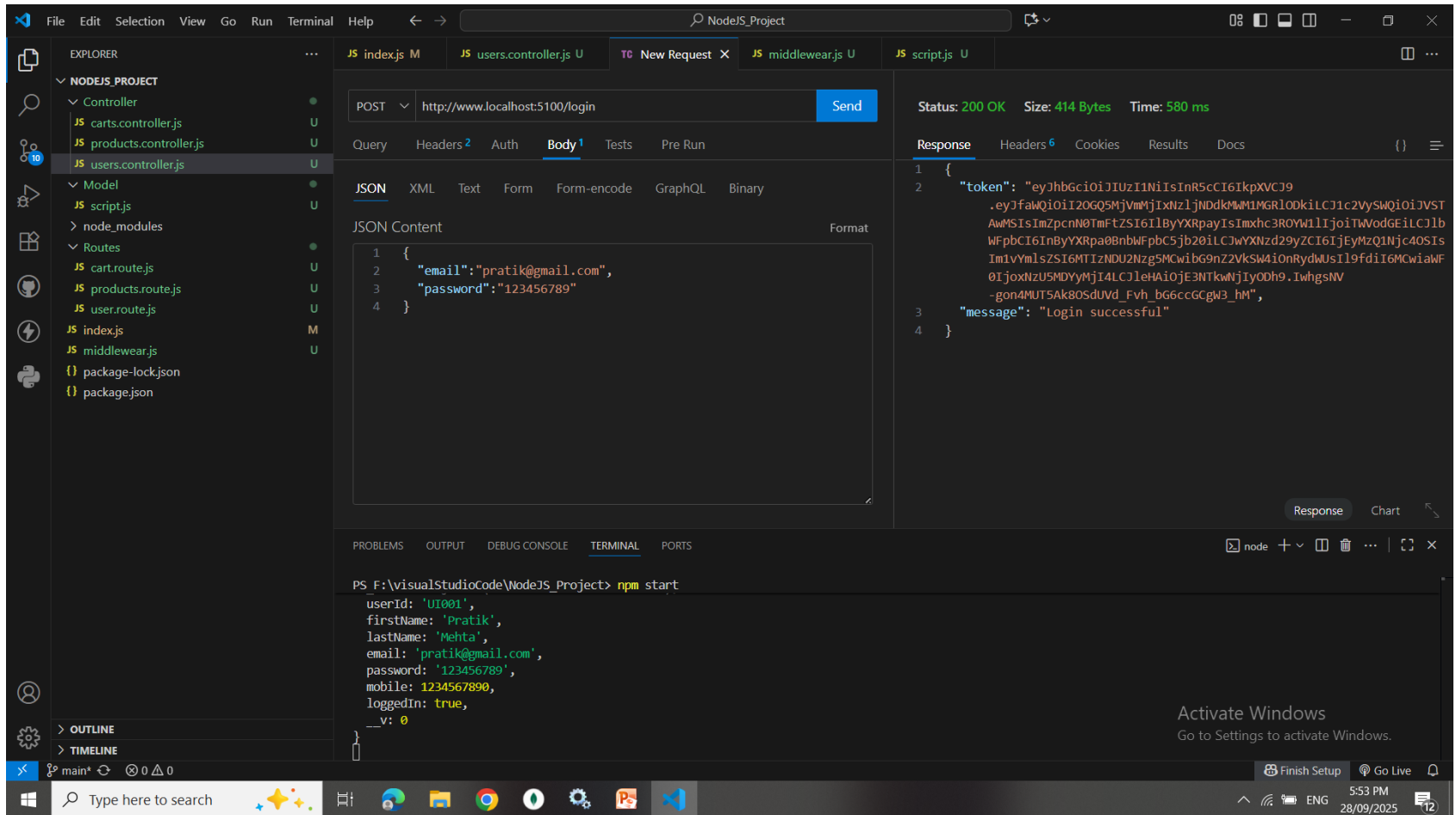
Response:

```
1 {
2   "userId": "UI001",
3   "firstName": "Pratik",
4   "lastName": "Mehta",
5   "email": "pratik@gmail.com",
6   "password": "123456789",
7   "mobile": "1234567890",
8   "loggedIn": false,
9   "_id": "68d925f22179c47d1c50de89"
10 }
```

Terminal:

```
PS F:\visualStudioCode\NodeJS_Project> npm start
```

Post request “/login”:



Post request “/cart”:

The screenshot shows the Visual Studio Code interface with a REST client request configured. The Explorer on the left shows the project structure for 'NODEJS_PROJECT', including files like 'index.js', 'middleware.js', 'cart.route.js', and 'products.controller.js'. The main editor displays a REST client request to 'http://www.localhost:5100/cart' with a POST method. The request body is a JSON object: { "userId": "UI001", "productId": "PI002" }. The response is shown as 'Status: 200 OK', 'Size: 32 Bytes', and 'Time: 37 ms'. The response body is a JSON object: { "message": "item added to cart" }. The terminal at the bottom shows the command 'npm start' and the output of the application, which includes user details and a logged-in status.

VS Code Explorer: NODEJS_PROJECT

- Controller
 - JS carts.controller.js
 - JS products.controller.js
 - JS users.controller.js
- Model
 - JS script.js
- node_modules
- Routes
 - JS cart.route.js
 - JS products.route.js
 - JS user.route.js
- JS index.js
- JS middleware.js
- package-lock.json
- package.json

REST Client Request: POST http://www.localhost:5100/cart

Send

Status: 200 OK Size: 32 Bytes Time: 37 ms

Response: { "message": "item added to cart" }

Terminal:

```
PS F:\visualStudioCode\NodeJS_Project> npm start
userId: 'UI001',
firstName: 'Pratik',
lastName: 'Mehta',
email: 'pratik@gmail.com',
password: '123456789',
mobile: 1234567890,
loggedIn: true,
_v: 0
```

Put request “/cart”:

The screenshot shows the Visual Studio Code interface with a REST client tab open. The request is a PUT to `http://www.localhost:5100/cart`. The body is a JSON object: `{ "userId": "UI001", "productId": "PI002", "stockQuantity": 1 }`. The response is a 200 OK with a JSON body: `{ "updatedProductQuantity": { "_id": "68d92b95df5ac665669e6e45", "userId": "UI001", "productId": "PI002", "quantity": 3, "__v": 0 } }`. The terminal shows the command `npm start` being executed.

VS Code Explorer shows the project structure:

- NODEJS_PROJECT
 - Controller
 - JS carts.controller.js
 - JS products.controller.js
 - JS users.controller.js
 - Model
 - JS script.js
 - node_modules
 - Routes
 - JS cart.route.js
 - JS products.route.js
 - JS user.route.js
 - JS index.js
 - JS middleware.js
 - package-lock.json
 - package.json

REST Client Request:

```
PUT http://www.localhost:5100/cart
```

Body (JSON):

```
{ "userId": "UI001", "productId": "PI002", "stockQuantity": 1 }
```

Response (200 OK):

```
{ "updatedProductQuantity": { "_id": "68d92b95df5ac665669e6e45", "userId": "UI001", "productId": "PI002", "quantity": 3, "__v": 0 } }
```

Terminal:

```
PS F:\visualStudioCode\NodeJS_Project> npm start
```

Terminal Output:

```
__v: 0
}
{
  _id: new ObjectId('68d92b95df5ac665669e6e45'),
  userId: 'UI001',
  productId: 'PI002',
  quantity: 3,
  __v: 0
}
```

Windows Taskbar: 6:10 PM, 28/09/2025

Delete request “/cart”:

The screenshot shows the Visual Studio Code interface with a REST client request and response for a DELETE request to `http://www localhost:5100/cart`. The request body is a JSON object with `userId: "UI001"` and `productId: "PI002"`. The response is a 200 OK status with a JSON body containing `"delete operation info": { "acknowledged": true, "deletedCount": 1 }`.

Request Details:

- Method: DELETE
- URL: `http://www localhost:5100/cart`
- Body (JSON):

```
{  "userId": "UI001",  "productId": "PI002"}
```

Response Details:

- Status: 200 OK
- Size: 64 Bytes
- Time: 104 ms
- Body (JSON):

```
{  "delete operation info": {    "acknowledged": true,    "deletedCount": 1  }}
```

Terminal Output:

```
PS F:\visualStudioCode\NodeJS_Project> npm start
__v: 0
{
  _id: new ObjectId('68d92b95df5ac665669e6e45'),
  userId: 'UI001',
  productId: 'PI002',
  quantity: 3,
  __v: 0
}
```

Post request “/cart”(Accessing cart after jwt expires):

