

Screen-Shots of ShoppyGlobe_backend

1. Server and MongoDB connection ↓

The screenshot shows the VS Code interface with the terminal tab active. The command `npm start` is run, and the output shows the server starting on port 8000. It also indicates that Mongoose is connected successfully.

```
PS C:\Users\dell\OneDrive\Desktop\ShoppyGlobe_backend> npm start
> shoppyglobe_backend@1.0.0 start
> nodemon server.js

[nodemon] 3.1.11
[nodemon] to restart at any time, enter 'rs'
[nodemon] watching path(s): *
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting 'node server.js'
[dotenv@17.2.3] Injecting env (3) from .env -- tip: 📁 encrypt with Dotenvx: https://dotenvx.com
[dotenv@17.2.3] Injecting env (0) from .env -- tip: 🚫 add access controls to secrets: https://dotenvx.com/ops
[dotenv@17.2.3] Injecting env (0) from .env -- tip: 🌐 enable debug logging with { debug: true }
Server is running on port 8000
Mongoose is Connected Successfully
```

2. Register ↓

The screenshot shows a POST request to the endpoint `http://localhost:8000/api/auth/register`. The response status is 201 Created, and the response body contains a JSON object indicating a new user was created successfully.

```
{
  "message": "User created successfully",
  "newUser": {
    "username": "user",
    "email": "user@com",
    "password": "$2b$10$QuWPXbarn4f8BQv7K0eXmV139/r1.JiE0Y1hhbn6Ypdv6t4.a",
    "_id": "692c686a381205fc5feb0136",
    "createdAt": "2025-11-30T15:53:14.632Z",
    "updatedAt": "2025-11-30T15:53:14.632Z",
    "__v": 0
  }
}
```

3. User with same name or email

The screenshot shows a Postman request to `http://localhost:8000/api/auth/register`. The request body contains:

```
{ "username": "user123", "email": "user@.com", "password": "user@123" }
```

The response status is **500 Internal Server Error**, size **159 Bytes**, and time **325 ms**. The response body is:

```
{
  "error": "Internal Server Error",
  "message": "E11000: duplicate key error collection: ShoppyGlobe_backend.users index: email_1 dup key: { email: 'user@.com' }"
}
```

The terminal below shows the command `npm start` running in the background.

The screenshot shows a Postman request to `http://localhost:8000/api/auth/register`. The request body contains:

```
{ "username": "user", "email": "user123@.com", "password": "user@123" }
```

The response status is **400 Bad Request**, size **35 Bytes**, and time **111 ms**. The response body is:

```
{
  "error": "Username already exists"
}
```

The terminal below shows the command `npm start` running in the background.

4. Register MongoDB

The screenshot shows the MongoDB Cloud Data Explorer interface. The left sidebar has a 'Data Explorer' tab selected, showing 'Cluster' (ShoppyGlobe_backend), 'PROJECT' (ShoppyGlobe_backend), and 'CLUSTER' (Cluster0). The main area displays the 'Data' section for 'Cluster0'. It shows 'DATABASES: 1' and 'COLLECTIONS: 3'. A search bar for 'Namespaces' is present. The 'ShoppyGlobe_backend' collection is expanded, showing documents for 'carts', 'products', and 'users'. The 'users' document is selected, displaying its fields and values. The top right shows 'VERSION 8.0.16' and 'REGION AWS Mumbai (ap-south-1)'. A 'PREVIEW' button, a toggle switch for 'New Data Explorer', and buttons for 'VISUALIZE YOUR DATA' and 'REFRESH' are also visible.

ShoppyGlobe_backend.users

STORAGE SIZE: 34KB LOGICAL DATA SIZE: 378B TOTAL DOCUMENTS: 2 INDEXES TOTAL SIZE: 72KB

Find Indexes Schema Anti-Patterns Aggregation Search Indexes

Generate queries from natural language in Compass ↗

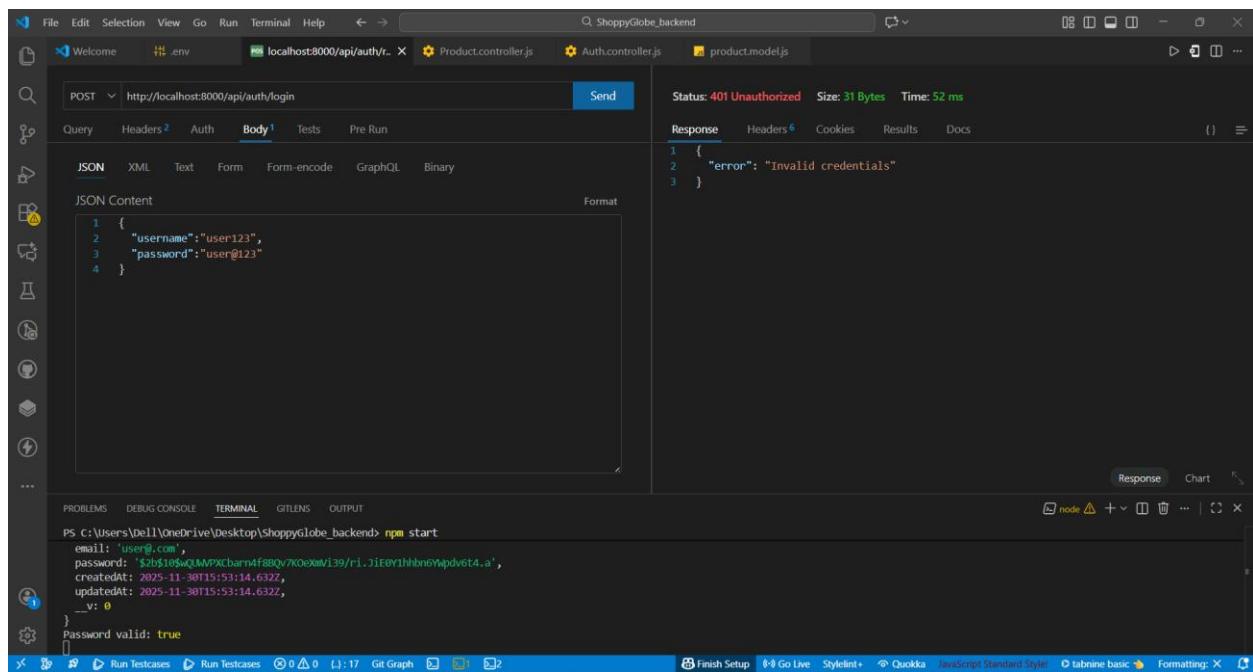
Filter ↗ Type a query: { field: 'value' } Reset Apply Options ↗

`password : "52b610$1RKXcVSUmsAeW6TKwv_iuASGFcOoMIs0rZu6IQex1BPkVkpdnC1"
createdAt : 2025-11-30T15:51:26.094+00:00
updatedAt : 2025-11-30T15:51:26.094+00:00
__v : 0`

`_id: ObjectId('692c680a381205fc5feb0136')
username : "user"
email : "user@com"
password : "52b610$QWNPYKbarn4f8B0v7K0eXmVi39/r1.JIE0Y1hhbn6Wpdv6t4.a"
createdAt : 2025-11-30T15:53:14.632+00:00
updatedAt : 2025-11-30T15:53:14.632+00:00
__v : 0`

5. Login

6. Wrong name or password



```
POST http://localhost:8000/api/auth/login
{
  "username": "user123",
  "password": "user@123"
}

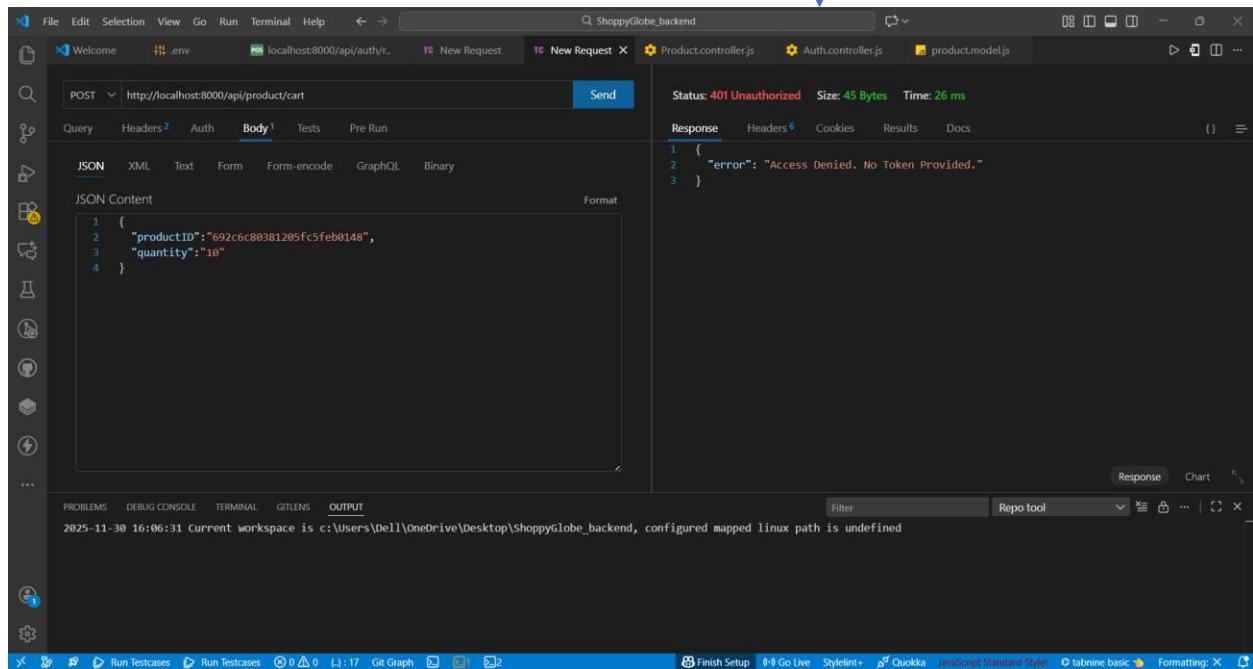
Status: 401 Unauthorized Size: 31 Bytes Time: 52 ms
{
  "error": "Invalid credentials"
}
```

PROBLEMS DEBUG CONSOLE TERMINAL GITLENS OUTPUT

```
PS C:\Users\dell\OneDrive\Desktop\ShoppyGlobe_backend> npm start
email: 'user@gmail.com',
password: '$2b$10$QqMMPXcbarnF8bQy7kOexwV139/r1..J1EBY1hhbn6WpdvSt4.a',
createdAt: 2025-11-30T15:53:14.632Z,
updatedAt: 2025-11-30T15:53:14.632Z,
__v: 0
}
Password valid: true
```

Run Testcases Run Testcases Filter Repo tool

7. Cart before logging



```
POST http://localhost:8000/api/product/cart
{
  "productID": "692c6c80381205fc5feb0148",
  "quantity": "10"
}

Status: 401 Unauthorized Size: 45 Bytes Time: 26 ms
{
  "error": "Access Denied. No Token Provided."
}
```

PROBLEMS DEBUG CONSOLE TERMINAL GITLENS OUTPUT

```
2025-11-30 16:06:31 Current workspace is c:\Users\OneDrive\Desktop\ShoppyGlobe_backend, configured mapped linux path is undefined
```

Run Testcases Run Testcases Filter Repo tool

8. After Valid Token/ logging ↓

The screenshot shows the Postman application interface. A POST request is being made to `http://localhost:8000/api/product/cart`. The request body contains the following JSON:

```
1 {
2   "productID": "692c6c80381205fc5feb0148",
3   "name": "Laptop",
4   "quantity": "10",
5   "price": "400000"
6 }
```

The response status is `201 Created`, the size is `249 Bytes`, and the time taken is `168 ms`. The response body is:

```
1 {
2   "message": "Product added to cart",
3   "data": {
4     "productID": "692c6c80381205fc5feb0148",
5     "quantity": 10,
6     "name": "Laptop",
7     "price": "400000",
8     "_id": "692c6dccc381205fc5feb014f",
9     "createdAt": "2025-11-30T16:16:12.864Z",
10    "updatedAt": "2025-11-30T16:16:12.864Z",
11    "__v": 0
12  }
13 }
```

At the bottom, it says `2025-11-30 16:06:31 Current workspace is c:\Users\OneDrive\Desktop\ShoppyGlobe_backend, configured mapped linux path is undefined`.

9. Cart in MongoDB ↓

The screenshot shows the MongoDB Compass interface connected to a cluster named `Cluster0`. The `ShoppyGlobe_backend` database is selected, and the `carts` collection is shown. The collection has 1 document.

The document details are:

```
_id: ObjectId('692c6dccc381205fc5feb014f')
productID: ObjectId('692c6c80381205fc5feb0148')
quantity: 10
name: "Laptop"
price: "400000"
createdAt: 2025-11-30T16:16:12.864+00:00
updatedAt: 2025-11-30T16:16:12.864+00:00
__v: 0
```

10. Cart Update

The screenshot shows a browser-based API testing interface. In the top navigation bar, there are tabs for 'File', 'Edit', 'Selection', 'View', 'Go', 'Run', 'Terminal', and 'Help'. Below the tabs, there are several tabs for different API endpoints: 'localhost:8000/api/auth/r...', 'GET localhost:8000/api/pr...', 'localhost:8000/api/auth/r...', 'localhost:8000/api/produ...', 'localhost:8000/api/produ...', 'Product.controller.js', and 'Auth.controller.js'. The main area is a request builder with the following details:

- Method: PUT
- URL: <http://localhost:8000/api/product/cart/692c6dcc381205fc5feb014f>
- Body tab is selected.
- JSON Content:

```
1 {
2   "quantity": "5"
3 }
```
- Response tab shows the following JSON response:

```
1 {
2   "message": "Cart updated successfully",
3   "data": {
4     "_id": "692c6dcc381205fc5feb014f",
5     "productId": "692c6cc80381205fc5feb0148",
6     "quantity": 5,
7     "name": "Laptop",
8     "price": "400000",
9     "createdAt": "2025-11-30T16:16:12.864Z",
10    "updatedAt": "2025-11-30T17:17:23.897Z",
11    "__v": 0
12  }
13 }
```
- Status: 200 OK
- Size: 252 Bytes
- Time: 82 ms

At the bottom of the interface, there are sections for 'PROBLEMS', 'DEBUG CONSOLE', 'TERMINAL', 'GITLENS', and 'OUTPUT'. The DEBUG CONSOLE section contains log messages from November 30, 2025, indicating current workspace paths. The bottom navigation bar includes links for 'Run Testcases', 'Finish Setup', 'Go Live', 'Stylelint', 'Quokka', 'Copy', 'Clipboard', 'Formatting', and 'tabnine basic'.

11. Cart Update in MongoDB

The screenshot shows the MongoDB Compass interface. At the top, the URL is 'cloud.mongodb.com/v2/692b46bb7595835ba341a34b#/metrics/replicaSet/692b4740ecf4435b7efd49ed/explorer/ShoppGlobe_backend/carts/fi...'. The interface has a sidebar with 'Cluster', 'Data Explorer', 'Real Time', 'Cluster Metrics', 'Query Insights', 'Performance Advisor', 'Online Archive', 'Command Line Tools', and 'Infrastructure as Code'. Under 'Data Explorer', it shows 'Cluster0' with 'DATABASES: 1' and 'COLLECTIONS: 3'. The 'ShoppGlobe_backend' collection is expanded, showing 'carts', 'products', and 'users'. The 'carts' section displays the following document:

```
_id: ObjectId('692c6dcc381205fc5feb014f')
productId: ObjectId('692c6cc80381205fc5feb0148')
quantity: 5
name: "Laptop"
price: "400000"
createdAt: 2025-11-30T16:16:12.864+00:00
updatedAt: 2025-11-30T17:17:23.897+00:00
__v: 0
```

Below the document, there are tabs for 'Find', 'Indexes', 'Schema Anti-Patterns', 'Aggregation', and 'Search Indexes'. A 'PREVIEW' button is visible at the top right. The interface also includes a 'New Data Explorer' toggle, 'VISUALIZE YOUR DATA' button, and 'REFRESH' button. The top right corner shows 'VERSION: 8.0.16' and 'REGION: AWS Mumbai (ap-south-1)'. The bottom right corner has a 'COMMENT' icon.

12. Cart Delete

The screenshot shows the Postman application interface. In the top left, there's a navigation bar with File, Edit, Selection, View, Go, Run, Terminal, Help. Below it is a search bar with 'ShoppGlobe_backend'. The main area has tabs for 'Welcome', '.env', and 'localhost:8000/api/auth/r...'. A 'New Request' tab is open, showing a 'DELETE' method at 'http://localhost:8000/api/product/cart/692c6dce381205f5feb014f'. The 'Body' tab is selected, showing JSON content: '1'. The 'Response' tab shows a successful 200 OK response with a size of 44 bytes and a time of 65 ms. The response body is:

```
1 {  
2   "message": "Cart item deleted successfully"  
3 }
```

. At the bottom, there's a terminal window showing log messages related to workspace paths.

13. Cart Delete in MongoDB

The screenshot shows the MongoDB Compass interface. The left sidebar has sections for Cluster, Organization (Ayush's Org - 2025-11-01), Project (ShoppyGlobe_backend), and Cluster (Cluster0). Under Data Explorer, the 'Data' section is selected, showing 'Cluster0'. It lists 'DATABASES: 1' and 'COLLECTIONS: 3'. A 'ShoppyGlobe_backend' database is expanded, showing 'carts' as the active collection. The 'carts' collection details are shown: STORAGE SIZE: 36KB, LOGICAL DATA SIZE: 0B, TOTAL DOCUMENTS: 0, INDEXES TOTAL SIZE: 36KB. Below this are buttons for Find, Indexes, Schema Anti-Patterns, Aggregation, and Search Indexes. A 'Type a query: { field: 'value' }' input field is present. The right side of the interface shows 'QUERY RESULTS: 0'.

14. MongoDB CRUB Operation (Add Product) ↓

The screenshot shows a browser-based API testing interface. The URL is `http://localhost:8000/api/addproduct`. The response status is **201 Created**, size is **242 Bytes**, and time is **62 ms**. The response body is a JSON object:

```
1 {
2   "message": "Product added successfully",
3   "data": {
4     "name": "Ipad",
5     "price": "50000",
6     "description": "Apple",
7     "stockquantity": "160"
8   }
9 }
```

The interface includes tabs for Headers, Body, Tests, and Pre Run. The Body tab shows the JSON content of the request. The Response tab shows the JSON response. The bottom pane displays terminal logs and various developer tools.

15. Add Product in MongoDB ↓

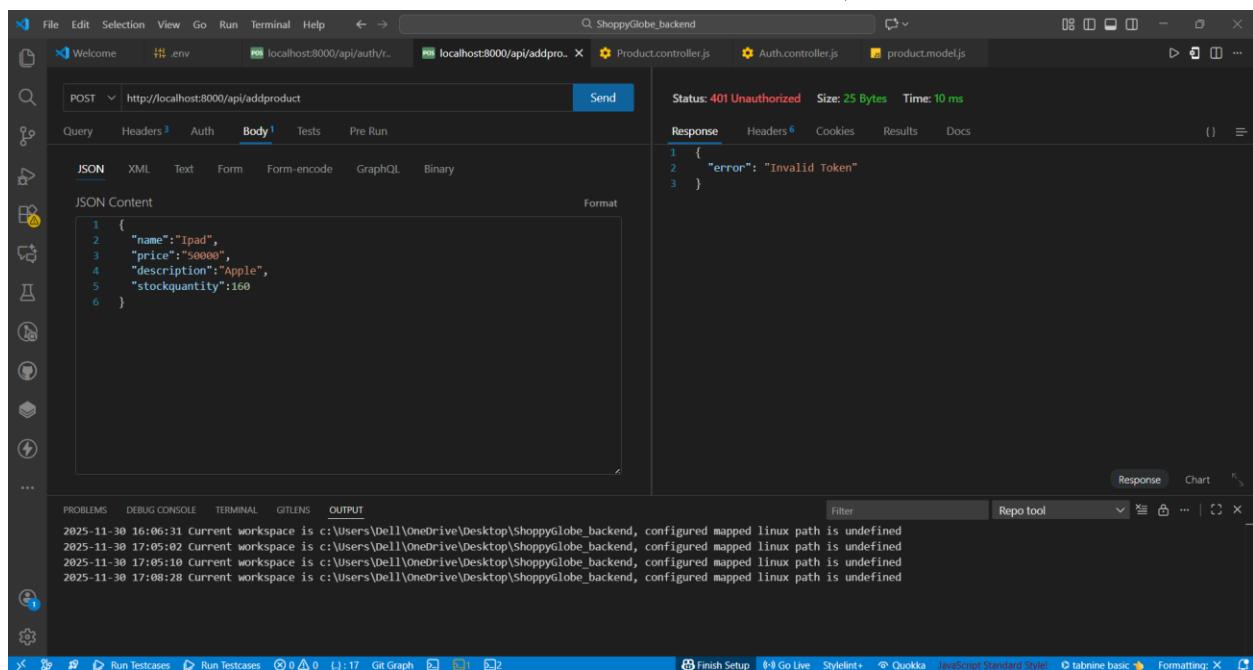
The screenshot shows the MongoDB Cloud Data Explorer interface. The cluster is **Ayush's Org - 2025-11-30**, project is **ShoppyGlobe_backend**, and cluster is **Cluster0**. The version is **8.0.16** and the region is **AWS Mumbai (ap-south-1)**.

The left sidebar shows the **Data Explorer** section, which includes **Real Time**, **Cluster Metrics**, **Query Insights**, **Performance Advisor**, **Online Archive**, **Command Line Tools**, and **Infrastructure as Code**. There is also a **Search & Vector Search** shortcut.

The main panel shows the **ShoppyGlobe_backend.products** collection. It lists one document:

```
_id: ObjectId('692c8232381205fc5feb0171')
name: "Ipad"
price: "50000"
description: "Apple"
stockquantity: "160"
createdAt: 2025-11-30T17:43:14.469+00:00
updatedAt: 2025-11-30T17:43:14.469+00:00
__v: 0
```

16. Add With Invalid Token ↓



POST <http://localhost:8000/api/addproduct> Send

Status: 401 Unauthorized Size: 25 Bytes Time: 10 ms

Response Headers Cookies Results Docs

```
1 {
2   "error": "Invalid Token"
3 }
```

Query Headers³ Auth Body¹ Tests Pre Run

JSON XML Text Form Form-encode GraphQL Binary

JSON Content Format

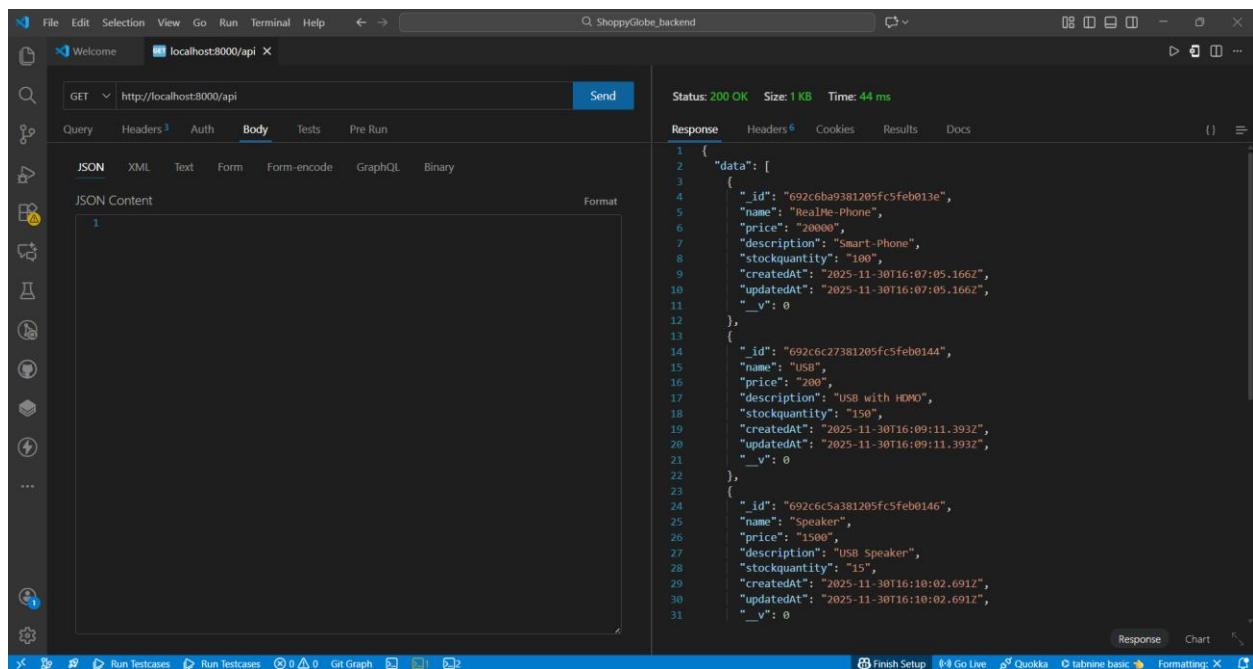
```
1 {
2   "name": "Ipad",
3   "price": "50000",
4   "description": "Apple",
5   "stockquantity": 160
6 }
```

PROBLEMS DEBUG CONSOLE TERMINAL GITLENS OUTPUT Filter Repo tool

2025-11-30 16:06:31 Current workspace is c:\Users\DELL\OneDrive\Desktop\ShoppyGlobe_backend, configured mapped linux path is undefined
2025-11-30 17:05:02 Current workspace is c:\Users\DELL\OneDrive\Desktop\ShoppyGlobe_backend, configured mapped linux path is undefined
2025-11-30 17:05:10 Current workspace is c:\Users\DELL\OneDrive\Desktop\ShoppyGlobe_backend, configured mapped linux path is undefined
2025-11-30 17:08:28 Current workspace is c:\Users\DELL\OneDrive\Desktop\ShoppyGlobe_backend, configured mapped linux path is undefined

Run Testcases Run Testcases Git Graph Filter Go Live Stylelint Quokka JavaScript Standard Style tabnine basic Formatting

17. Fetch Product From database ↓



GET <http://localhost:8000/api> Send

Status: 200 OK Size: 1 KB Time: 44 ms

Response Headers⁶ Cookies Results Docs

```
1 {
2   "data": [
3     {
4       "_id": "692c6ba9381205fc5feb013e",
5       "name": "realme-Phone",
6       "price": "20000",
7       "description": "Smart-Phone",
8       "stockquantity": "100",
9       "createdAt": "2025-11-30T16:07:05.166Z",
10      "updatedAt": "2025-11-30T16:07:05.166Z",
11      "__v": 0
12    },
13    {
14      "_id": "692c6c27381205fc5feb0144",
15      "name": "USB",
16      "price": "200",
17      "description": "USB with HDMO",
18      "stockquantity": "150",
19      "createdAt": "2025-11-30T16:09:11.393Z",
20      "updatedAt": "2025-11-30T16:09:11.393Z",
21      "__v": 0
22    },
23    {
24      "_id": "692c6c5a381205fc5feb0146",
25      "name": "speaker",
26      "price": "1500",
27      "description": "USB Speaker",
28      "stockquantity": "15",
29      "createdAt": "2025-11-30T16:10:02.691Z",
30      "updatedAt": "2025-11-30T16:10:02.691Z",
31      "__v": 0
32  ]}
```

Query Headers³ Auth Body¹ Tests Pre Run

JSON XML Text Form Form-encode GraphQL Binary

JSON Content Format

PROBLEMS DEBUG CONSOLE TERMINAL GITLENS OUTPUT Filter Repo tool

Run Testcases Run Testcases Git Graph Filter Go Live Stylelint Quokka JavaScript Standard Style tabnine basic Formatting

18. Fetch product by id

The screenshot shows the Postman interface with a successful API call. The URL is `http://localhost:8000/api/product/692c6c80381205fc5feb0148`. The response status is 200 OK, size is 213 Bytes, and time is 39 ms. The response body is a JSON object:

```
1 {
2   "data": {
3     "_id": "692c6c80381205fc5feb0148",
4     "name": "Laptop",
5     "price": "40000",
6     "description": "Intel core i5",
7     "stockquantity": "200",
8     "createdAt": "2025-11-30T16:10:40.785Z",
9     "updatedAt": "2025-11-30T16:10:40.785Z",
10    "__v": 0
11  }
12 }
```

The bottom of the screen shows the terminal output with several identical log entries:

```
2025-11-30 16:06:31 Current workspace is c:\Users\DELL\OneDrive\Desktop\ShoppyGlobe_backend, configured mapped linux path is undefined
2025-11-30 17:05:02 Current workspace is c:\Users\DELL\OneDrive\Desktop\ShoppyGlobe_backend, configured mapped linux path is undefined
2025-11-30 17:05:10 Current workspace is c:\Users\DELL\OneDrive\Desktop\ShoppyGlobe_backend, configured mapped linux path is undefined
2025-11-30 17:08:28 Current workspace is c:\Users\DELL\OneDrive\Desktop\ShoppyGlobe_backend, configured mapped linux path is undefined
```

19. Update Product

The screenshot shows the Postman interface with a successful API call. The URL is `http://localhost:8000/api/updateproduct/692c6c80381205fc5feb0148`. The response status is 200 OK, size is 259 Bytes, and time is 84 ms. The response body is a JSON object:

```
1 {
2   "message": "Product updated successfully",
3   "data": {
4     "_id": "692c6c80381205fc5feb0148",
5     "name": "Laptop-DELL",
6     "price": "40000",
7     "description": "Intel core i7",
8     "stockquantity": "350"
9   }
10 }
```

The bottom of the screen shows the terminal output with several identical log entries:

```
2025-11-30 16:06:31 Current workspace is c:\Users\DELL\OneDrive\Desktop\ShoppyGlobe_backend, configured mapped linux path is undefined
2025-11-30 17:05:02 Current workspace is c:\Users\DELL\OneDrive\Desktop\ShoppyGlobe_backend, configured mapped linux path is undefined
2025-11-30 17:05:10 Current workspace is c:\Users\DELL\OneDrive\Desktop\ShoppyGlobe_backend, configured mapped linux path is undefined
2025-11-30 17:08:28 Current workspace is c:\Users\DELL\OneDrive\Desktop\ShoppyGlobe_backend, configured mapped linux path is undefined
```

20. Before Update in MongoDB

The screenshot shows the MongoDB Cloud Data Explorer interface. On the left, the sidebar has sections like Overview, Data Explorer (which is selected and highlighted in green), Real Time, Cluster Metrics, Query Insights, Performance Advisor, Online Archive, Command Line Tools, and Infrastructure as Code. Under SHORTCUTS, there's a link to Search & Vector Search. The main area shows a database named 'ShoppyGlobe_backend' containing three collections: carts, products (selected and highlighted in green), and users. The products collection details are displayed on the right, showing a storage size of 34KB, logical data size of 1.04KB, 7 total documents, and 36KB of indexes. A document preview is shown:

```
_id: ObjectId('692c6c80381205fc5feb0148')
name : "Laptop"
price : "40000"
description: "Intel core i5"
stockQuantity : "200"
createdAt : 2025-11-30T16:10:40.785+00:00
updatedAt : 2025-11-30T16:10:40.785+00:00
__v : 0
```

21. After Update in MongoDB

The screenshot shows the same MongoDB Cloud Data Explorer interface after an update. The products collection details are now different, reflecting the changes made:

```
_id: ObjectId('692c6c80381205fc5feb0148')
name : "Laptop-DELL"
price : "40000"
description: "Intel core i7"
stockQuantity : "350"
createdAt : 2025-11-30T16:10:40.785+00:00
updatedAt : 2025-11-30T17:51:53.357+00:00
__v : 0
```

22. Delete Product from Database

The screenshot shows the Postman application interface. A DELETE request is made to the URL `http://localhost:8000/api/deleteproduct/692c6c0f381205fc5feb0142`. The response status is 200 OK, size is 249 Bytes, and time is 55 ms. The response body contains a JSON object with a message and data fields.

```
1 {
2   "message": "Product deleted successfully",
3   "data": {
4     "_id": "692c6c0f381205fc5feb0142",
5     "name": "Watch",
6     "price": "2000",
7     "description": "Smart-watch",
8     "stockquantity": "10",
9     "createdat": "2025-11-30T16:08:47.244Z",
10    "updatedat": "2025-11-30T16:08:47.244Z",
11    "__v": 0
12  }
13 }
```

23. Delete from MongoDB

The screenshot shows the MongoDB Compass interface connected to a cluster named `ShoppyGlobe_backend` in the project `ShoppyGlobe_backend`. The `products` collection is selected. The interface displays database and collection statistics, search bars, and a query builder for generating natural language queries.

