

## POST /register: Register a new user

The screenshot shows a REST client interface in VS Code. The request is a POST to `http://localhost:5000/register`. The response is a 201 Created status with a JSON body containing user registration details.

**Request:**

```
POST http://localhost:5000/register
```

**Response:**

```
{
  "email": "elonMusk@gmail.com",
  "password": "$2b$10$ydeVRTFNccyHvyS40vjho0Pn5aexYMc8wxs31aYM9XwtPXcrI/uUy",
  "_id": "680e66e31bed4ac76d279b2e",
  "createdAt": "2025-04-27T17:18:27.872Z",
  "updatedAt": "2025-04-27T17:18:27.872Z",
  "__v": 0
}
```

The terminal at the bottom shows the server is running on port 5000 and has successfully connected to MongoDB. It also shows the GET and POST requests being made.

## MongoDB Compass

The screenshot shows the MongoDB Compass interface. The database is `ShoppyGlobe` and the collection is `users`. The document displayed is a user registration record.

**Document:**

```
{
  "_id": ObjectId('680e66e31bed4ac76d279b2e'),
  "email": "elonMusk@gmail.com",
  "password": "$2b$10$ydeVRTFNccyHvyS40vjho0Pn5aexYMc8wxs31aYM9XwtPXcrI/uUy",
  "createdAt": "2025-04-27T17:18:27.872+00:00",
  "updatedAt": "2025-04-27T17:18:27.872+00:00",
  "__v": 0
}
```

**POST /login:** Authenticate user and return a JWT token.

The screenshot shows the VS Code interface with a REST client tab open. The request is a POST to `http://localhost:5000/login`. The body is a JSON object: `{ "email": "elonMusk@gmail.com", "password": "Musk" }`. The response status is **200 OK** with a size of **247 Bytes** and a time of **114 ms**. The response body is a JSON object: `{ "Message": "Login success", "Token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1bWVpbCI6ImVsb25NdXNrQGdtYWlsLmNvbSI6Im1hdCI6MTc0NTc3NzAzN30.h7nxeXBJRqq7xKNoeL_egwkU6ISTpyejn2scAu-QCuo", "userId": "680e66e31bed4ac76d279b2e", "email": "elonMusk@gmail.com" }`. The terminal at the bottom shows the command prompt with the following output: `Successfully connected to MongoDB..!  
GET , http://localhost:5000/products  
POST , http://localhost:5000/register  
POST , http://localhost:5000/login`

**Invalid password**

The screenshot shows the VS Code interface with a REST client tab open. The request is a POST to `http://localhost:5000/login`. The body is a JSON object: `{ "email": "elonMusk@gmail.com", "password": "Musk" }`. The response status is **404 Not Found** with a size of **30 Bytes** and a time of **87 ms**. The response body is a JSON object: `{ "Message": "Invalid password" }`. The terminal at the bottom shows the command prompt with the following output: `GET , http://localhost:5000/products  
POST , http://localhost:5000/register  
POST , http://localhost:5000/login  
POST , http://localhost:5000/login`

## POST /addproduct: add product to the product list

The screenshot shows the VS Code REST Client interface. The request is a POST to `http://localhost:5000/addproduct`. The request body is a JSON object: `{ "title": "Essence Mascara Lash Princess", "price": 9.99, "stock": 5, "description": "The Essence Mascara Lash Princess is a popular mascara known for its volumizing and lengthening effects. Achieve dramatic lashes with this long-lasting and cruelty-free formula." }`. The response status is 201 Created, with a size of 377 Bytes and a time of 49 ms. The response body is a JSON object: `{ "title": "Essence Mascara Lash Princess", "price": 9.99, "description": "The Essence Mascara Lash Princess is a popular mascara known for its volumizing and lengthening effects. Achieve dramatic lashes with this long-lasting and cruelty-free formula.", "stock": 5, "_id": "680e7c1f568b7a62ce5218bd", "createdAt": "2025-04-27T18:49:03.110Z", "updatedAt": "2025-04-27T18:49:03.110Z", "__v": 0 }`. The terminal at the bottom shows the message: `Successfully connected to MongoDB...! POST , http://localhost:5000/addproduct`.

```
POST http://localhost:5000/addproduct

{
  "title": "Essence Mascara Lash Princess",
  "price": 9.99,
  "stock": 5,
  "description": "The Essence Mascara Lash Princess is a popular mascara known for its volumizing and lengthening effects. Achieve dramatic lashes with this long-lasting and cruelty-free formula."
}
```

Status: 201 Created Size: 377 Bytes Time: 49 ms

```
{
  "title": "Essence Mascara Lash Princess",
  "price": 9.99,
  "description": "The Essence Mascara Lash Princess is a popular mascara known for its volumizing and lengthening effects. Achieve dramatic lashes with this long-lasting and cruelty-free formula.",
  "stock": 5,
  "_id": "680e7c1f568b7a62ce5218bd",
  "createdAt": "2025-04-27T18:49:03.110Z",
  "updatedAt": "2025-04-27T18:49:03.110Z",
  "__v": 0
}
```

Successfully connected to MongoDB...!  
POST , http://localhost:5000/addproduct

## MongoDB compass

The screenshot shows the MongoDB Compass interface. The database is 'ShopyGlobe' and the collection is 'products'. The document displayed is: `{ "_id": ObjectId("680e7c1f568b7a62ce5218bd"), "title": "Essence Mascara Lash Princess", "price": 9.99, "description": "The Essence Mascara Lash Princess is a popular mascara known for its v...", "stock": 5, "createdAt": "2025-04-27T18:49:03.110+00:00", "updatedAt": "2025-04-27T18:49:03.110+00:00", "__v": 0 }`. The interface includes a sidebar with connections, a search bar, and a main area for viewing and editing documents.

Connections Edit View Collection Help

Compass

My Queries

CONNECTIONS (3)

Search connections

localhost:27017

ShopyGlobe

products

users

admin

config

org

Users

localhost:27017 > ShopyGlobe > products

Documents 0 Aggregations Schema Indexes 1 Validation

Type a query: { field: 'value' } or Generate query

Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE

25 1-1 of 1

```
{
  "_id": ObjectId("680e7c1f568b7a62ce5218bd"),
  "title": "Essence Mascara Lash Princess",
  "price": 9.99,
  "description": "The Essence Mascara Lash Princess is a popular mascara known for its v...",
  "stock": 5,
  "createdAt": "2025-04-27T18:49:03.110+00:00",
  "updatedAt": "2025-04-27T18:49:03.110+00:00",
  "__v": 0
}
```

**GET /products:** Fetch a list of products from MongoDB.

The screenshot shows a REST client interface. The URL bar contains `http://localhost:5000/products` and the method is `GET`. The `Send` button is highlighted. The `Query` tab is active, showing `Query Parameters` with a table for parameters. The `Response` tab is active, displaying a JSON array of product objects. The status bar at the bottom shows `GET , http://localhost:5000/products`.

```
GET http://localhost:5000/products

Status: 200 OK Size: 1.7 KB Time: 15 ms

Response
0  {
1    "description": "The Essence Mascara Lash Princess is a popular mascara known for its volumizing and lengthening effects. Achieve dramatic lashes with this long-lasting and cruelty-free formula.",
2    "stock": 5,
3    "createdAt": "2025-04-27T18:49:03.110Z",
4    "updatedAt": "2025-04-27T18:49:03.110Z",
5    "_v": 0
6  },
7  {
8    "_id": "680e7d36568b7a62ce5218bf",
9    "title": "Eyeshadow Palette with Mirror",
10   "price": 19.99,
11   "description": "The Eyeshadow Palette with Mirror offers a versatile range of eyeshadow shades. Perfect for creating dramatic looks or subtle enhancements."
12 }
```

## MongoDB compass

The screenshot shows the MongoDB Compass interface. The left sidebar shows the `connections` list with `localhost:27017` selected. The `ShoppYGlobe` database is expanded, showing the `products` collection. The main panel displays the `products` collection with a list of documents. The `Documents` tab is active, showing a list of documents with fields like `_id`, `title`, `price`, `description`, `stock`, `createdAt`, `updatedAt`, and `_v`.

```
localhost:27017 > ShoppYGlobe > products

Documents 0 Aggregations Schema Indexes 1 Validation

Type a query: { field: 'value' } or Generate query ⚡

[+] ADD DATA [EXPORT DATA] [UPDATE] [DELETE] 25 1-5 of 5

{
  "_id": ObjectId('680e7c1f568b7a62ce5218bd'),
  "title": "Essence Mascara Lash Princess",
  "price": 9.99,
  "description": "The Essence Mascara Lash Princess is a popular mascara known for its v...",
  "stock": 5,
  "createdAt": 2025-04-27T18:49:03.110+00:00,
  "updatedAt": 2025-04-27T18:49:03.110+00:00,
  "_v": 0
}

{
  "_id": ObjectId('680e7d36568b7a62ce5218bf'),
  "title": "Eyeshadow Palette with Mirror",
  "price": 19.99,
  "description": "The Eyeshadow Palette with Mirror offers a versatile range of eyeshado...",
  "stock": 45,
  "createdAt": 2025-04-27T18:53:42.097+00:00,
  "updatedAt": 2025-04-27T18:53:42.097+00:00,
  "_v": 0
}

{
  "_id": ObjectId('680e7d7b568b7a62ce5218c1'),
  "title": "Powder Canister",
  "price": 14.99,
  "description": "The Powder Canister is a finely milled setting powder designed to set _...",
  "stock": 59,
  "createdAt": 2025-04-27T18:54:51.993+00:00,
  "updatedAt": 2025-04-27T18:54:51.993+00:00,
  "_v": 0
}
```

**GET /products/:id** - Fetch details of a single product by its ID.

The screenshot shows a REST client interface in VS Code. The request is a GET to `http://localhost:5000/products/680e7d36568b7a62ce5218bf`. The response is a 200 OK status with 387 bytes and a time of 19 ms. The response body is a JSON object representing a product.

```
1 {
2   "_id": "680e7d36568b7a62ce5218bf",
3   "title": "Eyeshadow Palette with Mirror",
4   "price": 19.99,
5   "description": "The Eyeshadow Palette with Mirror
6     offers a versatile range of eyeshadow shades for
7     creating stunning eye looks. With a built-in
8     mirror, it's convenient for on-the-go makeup
9     application.",
10  "stock": 45,
11  "createdAt": "2025-04-27T18:53:42.097Z",
12  "updatedAt": "2025-04-27T18:53:42.097Z",
13  "__v": 0
14 }
```

**MongoDB compass**

The screenshot shows the MongoDB Compass interface. The left sidebar shows the database structure with 'products' selected. The main area displays the 'products' collection with two documents. The first document is for 'Essence Mascara Lash Princess' and the second is for 'Eyeshadow Palette with Mirror'.

```
{
  "_id": ObjectId("680e7c1f568b7a62ce5218bd"),
  "title": "Essence Mascara Lash Princess",
  "price": 9.99,
  "description": "The Essence Mascara Lash Princess is a popular mascara known for its v...",
  "stock": 5,
  "createdAt": "2025-04-27T18:49:03.118+00:00",
  "updatedAt": "2025-04-27T18:49:03.118+00:00",
  "__v": 0
}
```

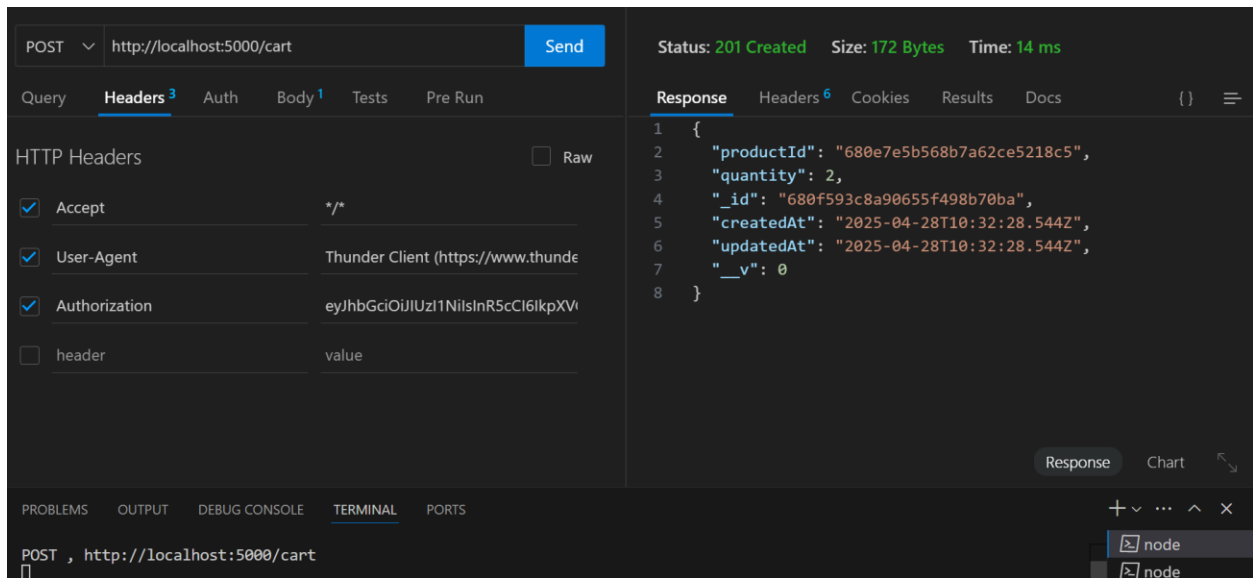
```
{
  "_id": ObjectId("680e7d36568b7a62ce5218bf"),
  "title": "Eyeshadow Palette with Mirror",
  "price": 19.99,
  "description": "The Eyeshadow Palette with Mirror offers a versatile range of eyeshado...",
  "stock": 45,
  "createdAt": "2025-04-27T18:53:42.097+00:00",
  "updatedAt": "2025-04-27T18:53:42.097+00:00",
  "__v": 0
}
```

**POST /cart:** Add a product to the shopping cart.

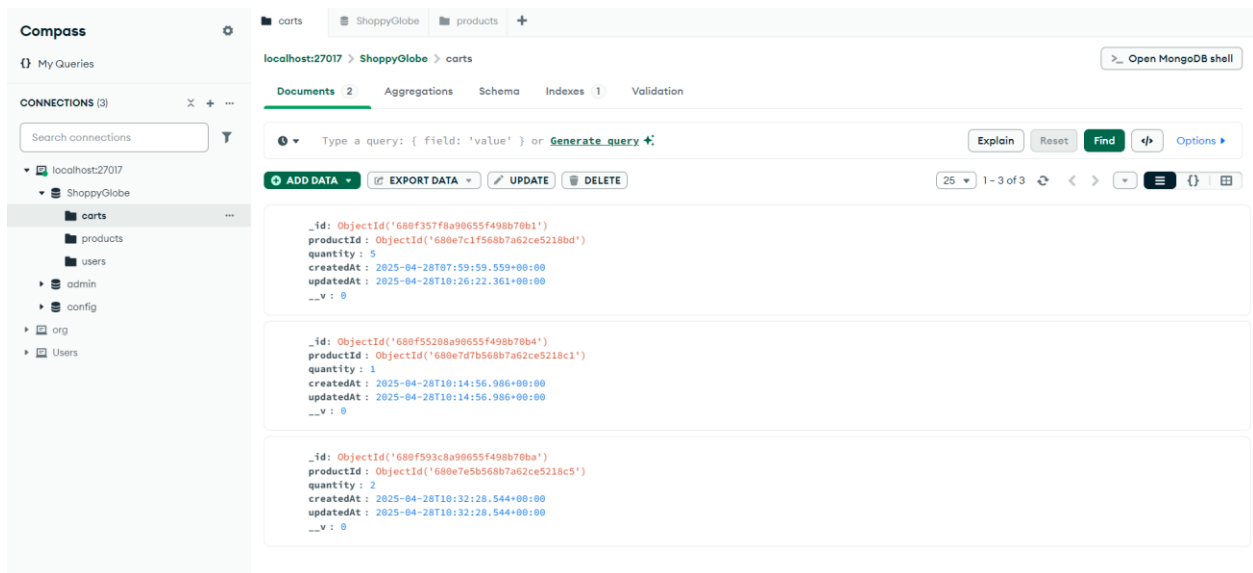
When token is not added

The screenshot shows the Thunder Client interface. The top bar includes tabs for JS files (cartRoutes.js, cartModel.js, productModel.js, cartController.js, productController.js) and a TC tab. The main panel displays a POST request to `http://localhost:5000/cart`. The **Headers** tab is active, showing standard headers like `Accept: */*` and `User-Agent: Thunder Client`. The **Response** tab shows a `404 Not Found` status, `34 Bytes` size, and `5 ms` time. The response body is a JSON object: `{ "Message": "Please Enter a Token" }`. The bottom terminal shows the command `POST , http://localhost:5000/cart`.

The screenshot shows the Thunder Client interface after a successful POST request to `http://localhost:5000/cart`. The **Body** tab is active, showing the request body in JSON format: `{ "productId": "680e7e5b568b7a62ce5218c5", "quantity": 2 }`. The **Response** tab shows a `201 Created` status, `172 Bytes` size, and `14 ms` time. The response body is a detailed JSON object: `{ "productId": "680e7e5b568b7a62ce5218c5", "quantity": 2, "_id": "680f593c8a90655f498b70ba", "createdAt": "2025-04-28T10:32:28.544Z", "updatedAt": "2025-04-28T10:32:28.544Z", "__v": 0 }`. The bottom terminal shows the command `POST , http://localhost:5000/cart`.



## MongoDB compass



**PUT /cart/:id:** Update the quantity of a product in the cart.

The screenshot shows a REST client interface with the following details:

- Method:** PUT
- URL:** `http://localhost:5000/cart/680f357f8a90655f498b70b1`
- Body:** JSON content: 

```
{
  "quantity": 5
}
```
- Status:** 200 OK
- Size:** 172 Bytes
- Time:** 28 ms
- Response:**

```
{
  "_id": "680f357f8a90655f498b70b1",
  "productId": "680e7c1f568b7a62ce5218bd",
  "quantity": 5,
  "createdAt": "2025-04-28T07:59:59.559Z",
  "updatedAt": "2025-04-28T10:26:22.361Z",
  "__v": 0
}
```
- Terminal:** Shows the command `PUT , http://localhost:5000/cart/680f357f8a90655f498b70b1` and its output.

Pass the token in header

The screenshot shows a REST client interface with the following details:

- Method:** PUT
- URL:** `http://localhost:5000/cart/680f357f8a90655f498b70b1`
- Headers:**
  - Accept:** \*/\*
  - User-Agent:** Thunder Client (https://www.thund
  - Authorization:** eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXV
- Status:** 200 OK
- Size:** 172 Bytes
- Time:** 28 ms
- Response:**

```
{
  "_id": "680f357f8a90655f498b70b1",
  "productId": "680e7c1f568b7a62ce5218bd",
  "quantity": 5,
  "createdAt": "2025-04-28T07:59:59.559Z",
  "updatedAt": "2025-04-28T10:26:22.361Z",
  "__v": 0
}
```
- Terminal:** Shows the command `PUT , http://localhost:5000/cart/680f357f8a90655f498b70b1` and its output.



## Updated quantity in MongoDB compass

The screenshot shows the MongoDB Compass interface. On the left, the 'CONNECTIONS (3)' sidebar lists 'localhost:27017' and 'ShopyGlobe'. The 'ShopyGlobe' database is expanded, showing collections: 'carts', 'products', 'users', 'admin', 'config', 'org', and 'Users'. The 'carts' collection is selected. The main area displays the 'Documents' tab for the 'carts' collection. It shows two documents with the following fields: '\_id', 'productId', 'quantity', 'createdAt', 'updatedAt', and '\_\_v'. The first document has a quantity of 2, and the second has a quantity of 1. The interface includes a search bar, a query editor, and buttons for 'ADD DATA', 'EXPORT DATA', 'UPDATE', and 'DELETE'.

Connections Edit View Collection Help

Compass

My Queries

CONNECTIONS (3)

Search connections

localhost:27017

ShopyGlobe

carts

products

users

admin

config

org

Users

localhost:27017 > ShopyGlobe > carts

Documents 2 Aggregations Schema Indexes 1 Validation

Type a query: { field: 'value' } or [Generate query](#)

Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE

25 1 - 2 of 2

```
{
  "_id": ObjectId("680f357f8a90655f490b70b1"),
  "productId": ObjectId("680e7c1f568b7a62ce5218bd"),
  "quantity": 2,
  "createdAt": 2025-04-28T07:59:59.559+00:00,
  "updatedAt": 2025-04-28T10:26:22.361+00:00,
  "__v": 0
}
```

```
{
  "_id": ObjectId("680f55208a90655f490b70b4"),
  "productId": ObjectId("680e7d7b568b7a62ce5218c1"),
  "quantity": 1,
  "createdAt": 2025-04-28T10:14:56.986+00:00,
  "updatedAt": 2025-04-28T10:14:56.986+00:00,
  "__v": 0
}
```

**DELETE /cart/:id** - Remove a product from the cart.

The screenshot shows the Thunder Client interface. The top bar displays several tabs, including 'localhost:5000/users/1' and 'localhost:5000/cart'. The main area is divided into two panels. The left panel shows the 'Headers' tab for the DELETE request to 'http://localhost:5000/cart/680f55208a90655f498b70b4'. The headers include 'Accept: \*/\*', 'User-Agent: Thunder Client (https://www.thundercli...', and 'Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9. X'. The right panel shows the 'Response' tab with a status of '200 OK', size of '44 Bytes', and time of '28 ms'. The response body is a JSON object: { 'message': 'Cart item deleted successfully' }. The bottom panel shows the terminal output: 'DELETE , http://localhost:5000/cart/680f55208a90655f498b70b4'.

MongoDB compass

The screenshot shows the MongoDB Compass interface. The left sidebar displays the 'Connections' list with 'localhost:27017' selected. The main area shows the 'carts' collection in the 'ShoppyGlobe' database. The 'Documents' tab is active, displaying two documents. The first document has fields: '\_id: ObjectId('680f357f8a90655f498b70b1')', 'productId: ObjectId('680e7c1f568b7a62ce5218bd')', 'quantity: 5', 'createdAt: 2025-04-28T07:59:59.559+00:00', 'updatedAt: 2025-04-28T10:26:22.361+00:00', and '\_\_v: 0'. The second document has fields: '\_id: ObjectId('680f593c8a90655f498b70ba')', 'productId: ObjectId('680e7e5b568b7a62ce5218c5')', 'quantity: 2', 'createdAt: 2025-04-28T10:32:28.544+00:00', 'updatedAt: 2025-04-28T10:32:28.544+00:00', and '\_\_v: 0'. The top bar shows the database name 'ShoppyGlobe' and the collection name 'carts'.