

Triveous

BACKEND ASSIGNMENT: "E-commerce API"

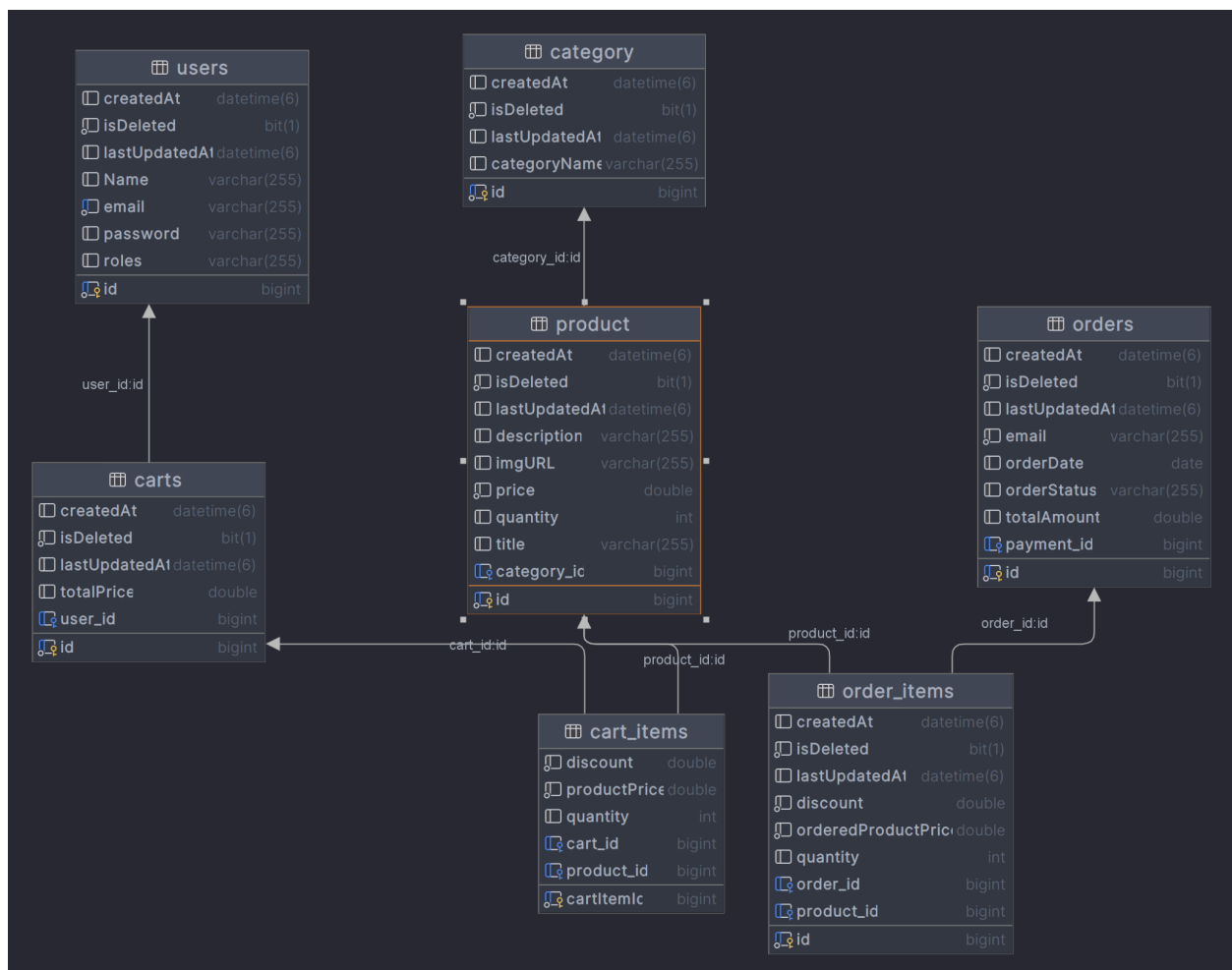
We have different services in E-commerce Backend API.

Services:

User Service
Cart Service
Order Service
Category Service
Product Service

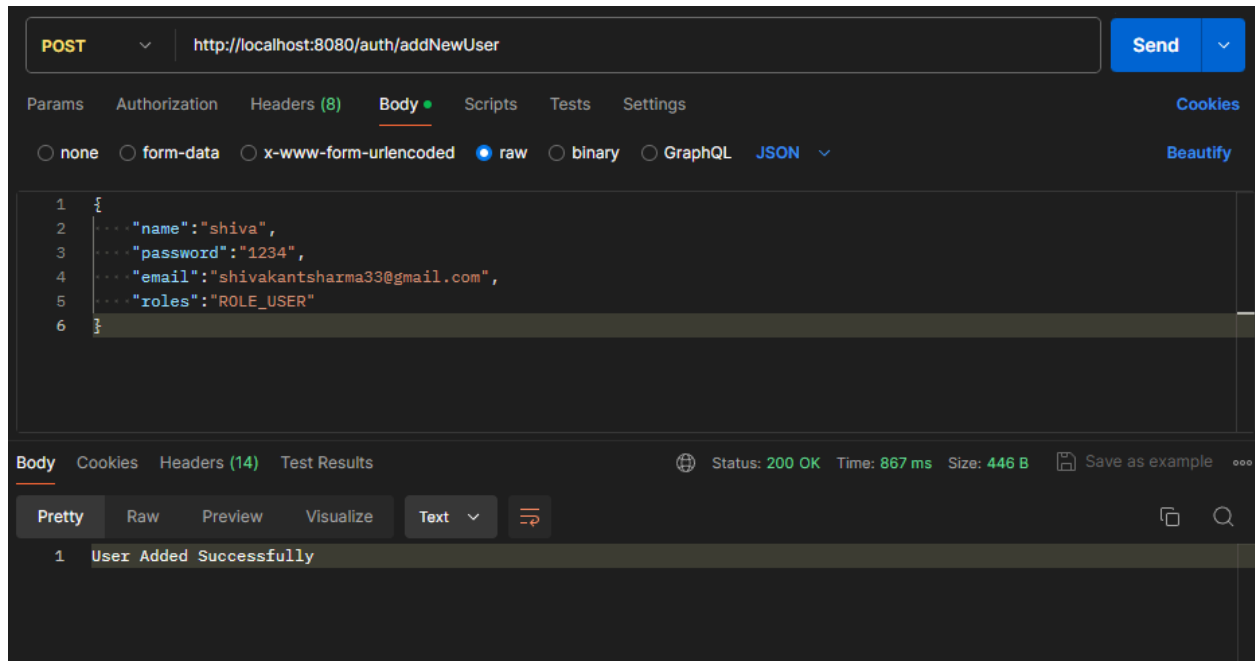
User Service contains user signup, user login. We have two types of users-Admin and Normal user

Below is the schema Diagram.



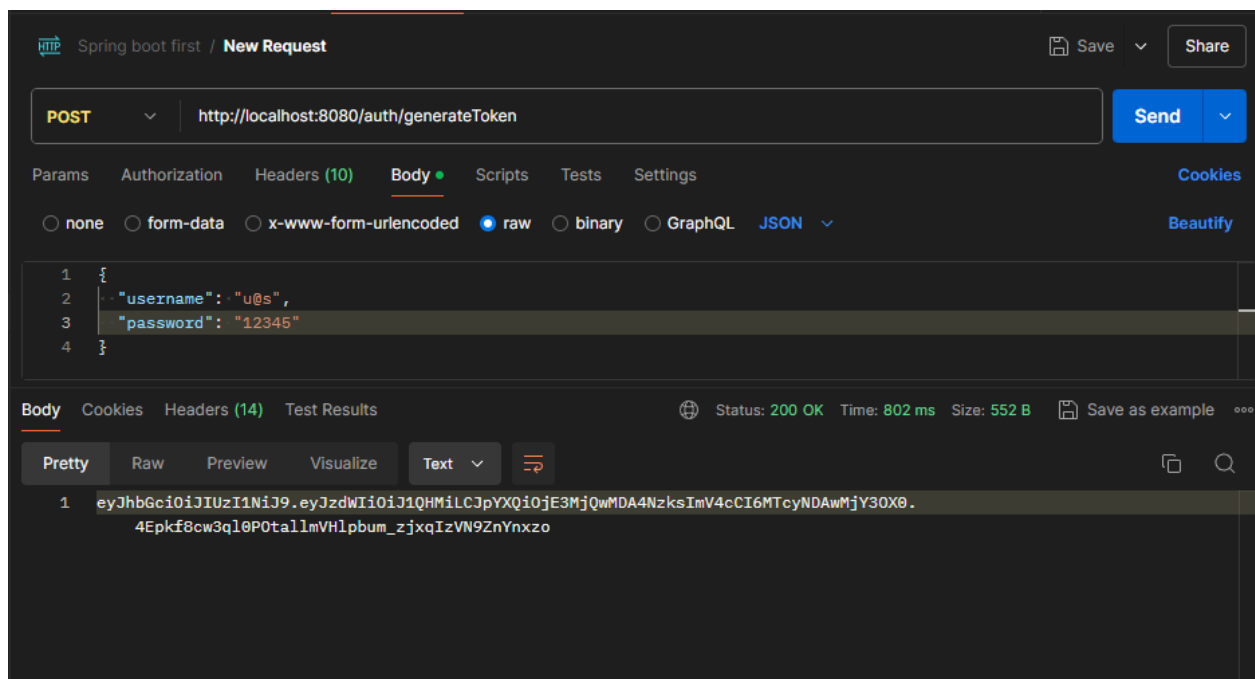
User Service

User Signup: User Sign up will take place in two roles ADMIN and USER

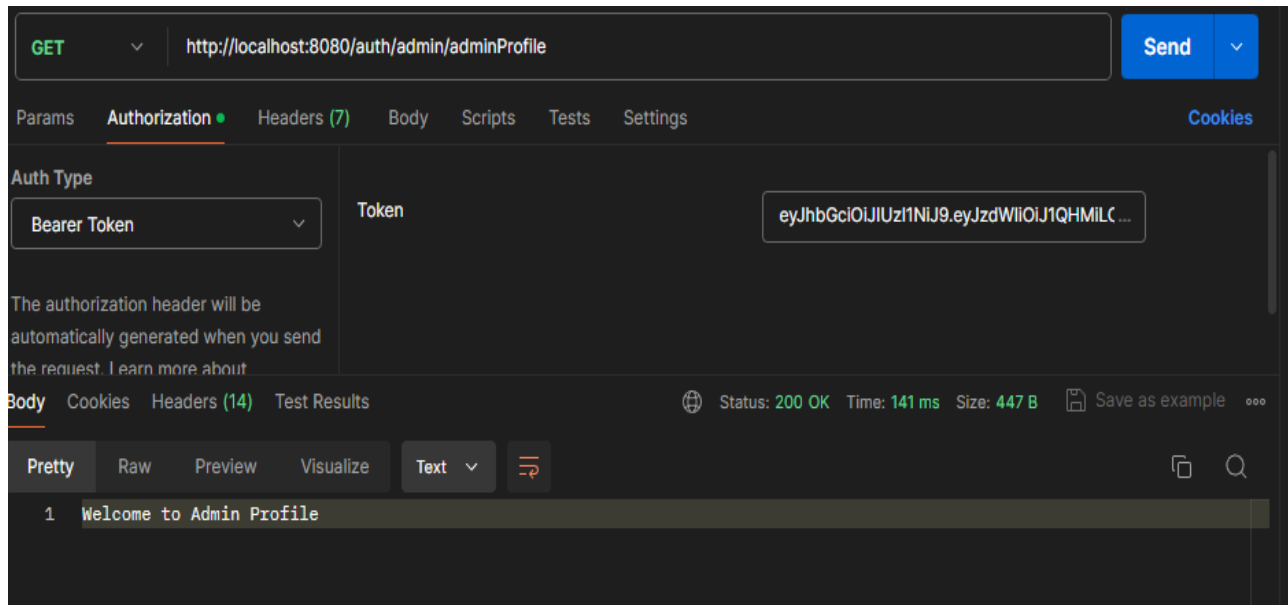


User login generates JWT token which requires for making any other request in API. Admin can access every path and user can only access public path.

Here I have generated admin token for making changes in Project.

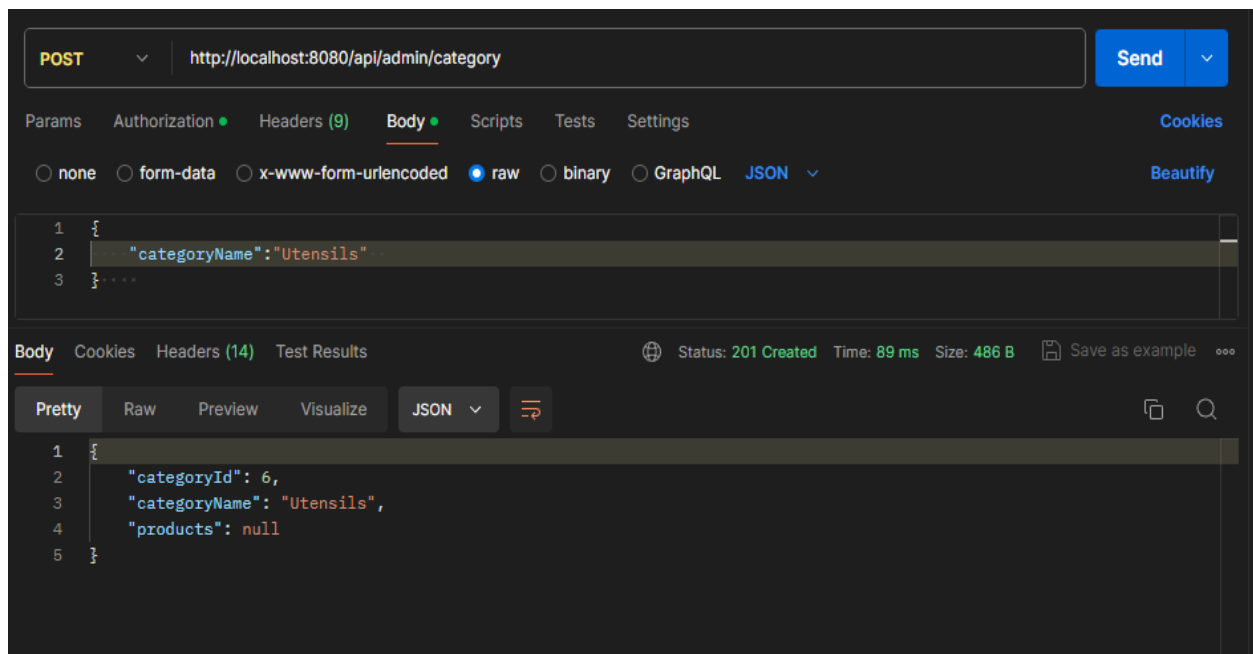


User validation by Token

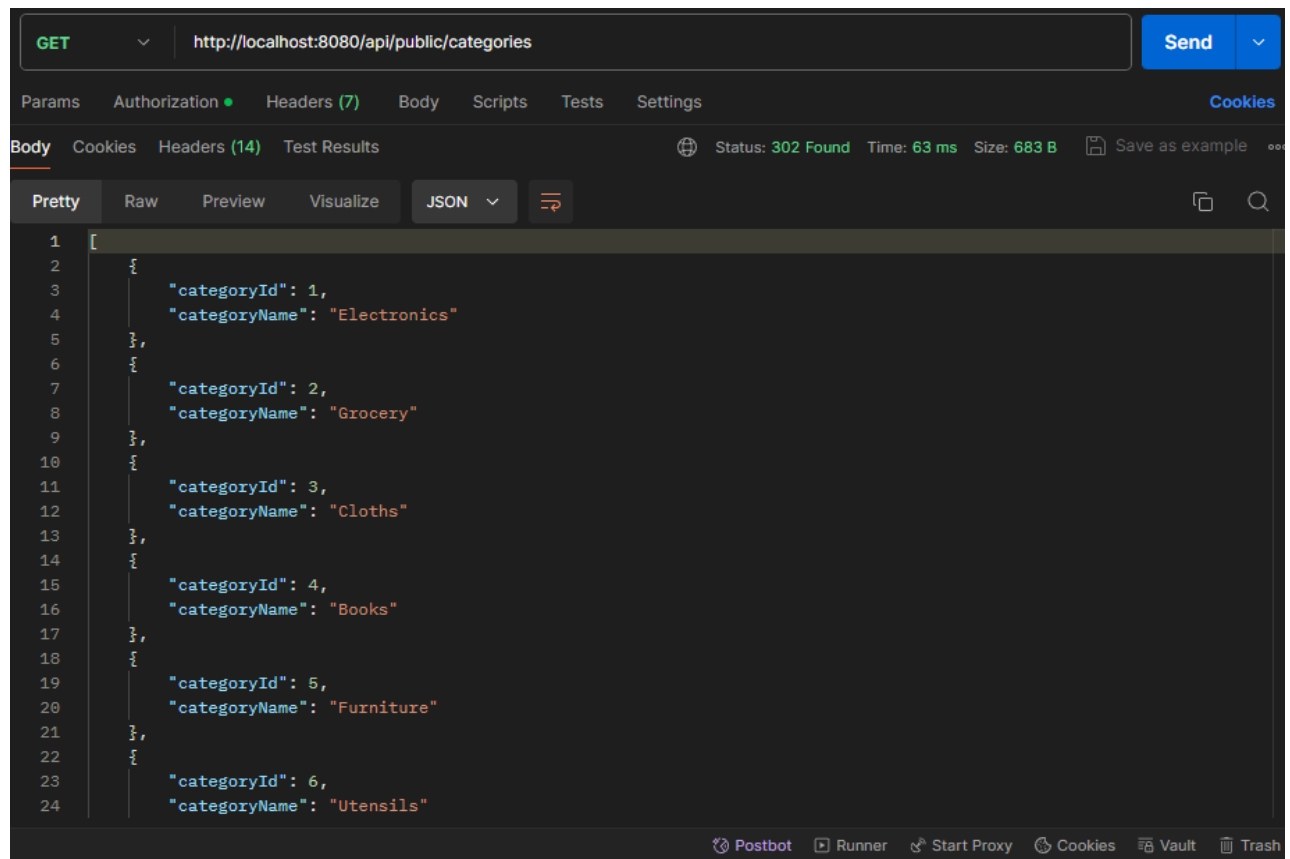


Category Service

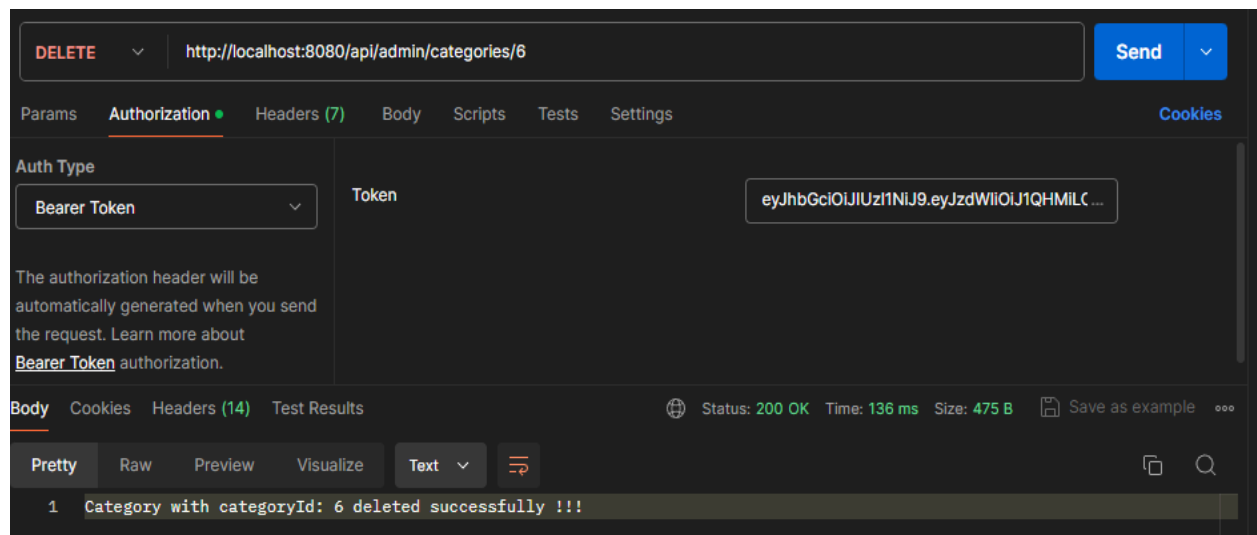
Add Category- Only admin can add a new category



Get all Categories-admin and general user both can see all categories

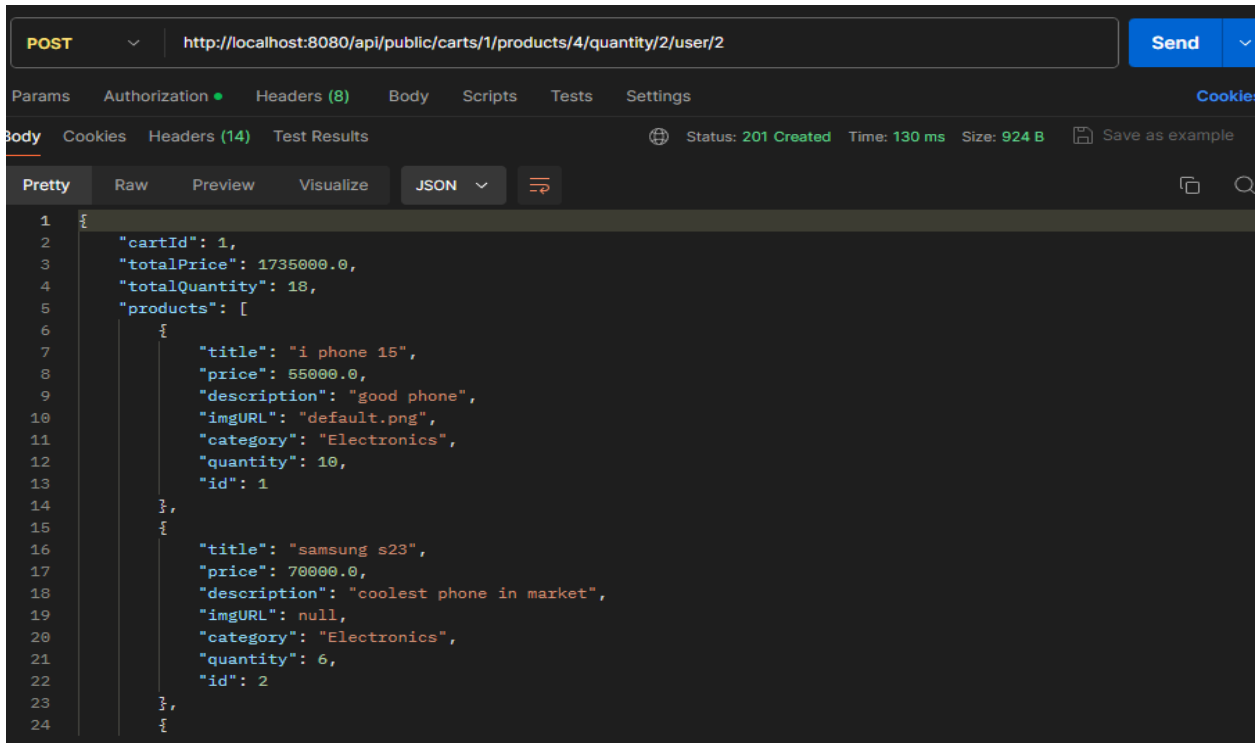


Delete category- only admin can delete a category.



Cart Services

Add Product to cart-It will create cart, if cart does not exist and add product into it . If Cart already exists then add product into it.

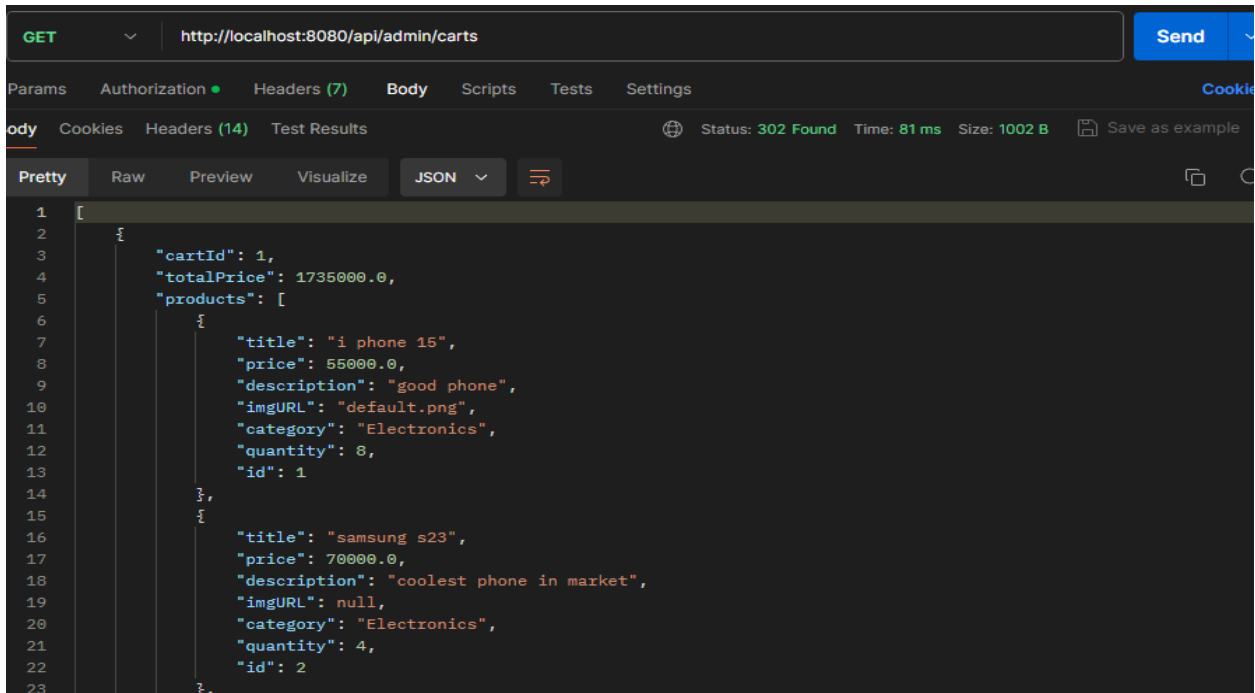


```
POST http://localhost:8080/api/public/carts/1/products/4/quantity/2/user/2

Status: 201 Created Time: 130 ms Size: 924 B

{
  "cartId": 1,
  "totalPrice": 1735000.0,
  "totalQuantity": 18,
  "products": [
    {
      "title": "i phone 15",
      "price": 55000.0,
      "description": "good phone",
      "imgURL": "default.png",
      "category": "Electronics",
      "quantity": 10,
      "id": 1
    },
    {
      "title": "samsung s23",
      "price": 70000.0,
      "description": "coolest phone in market",
      "imgURL": null,
      "category": "Electronics",
      "quantity": 6,
      "id": 2
    }
  ]
}
```

Get all the carts- Only admin can get all the carts

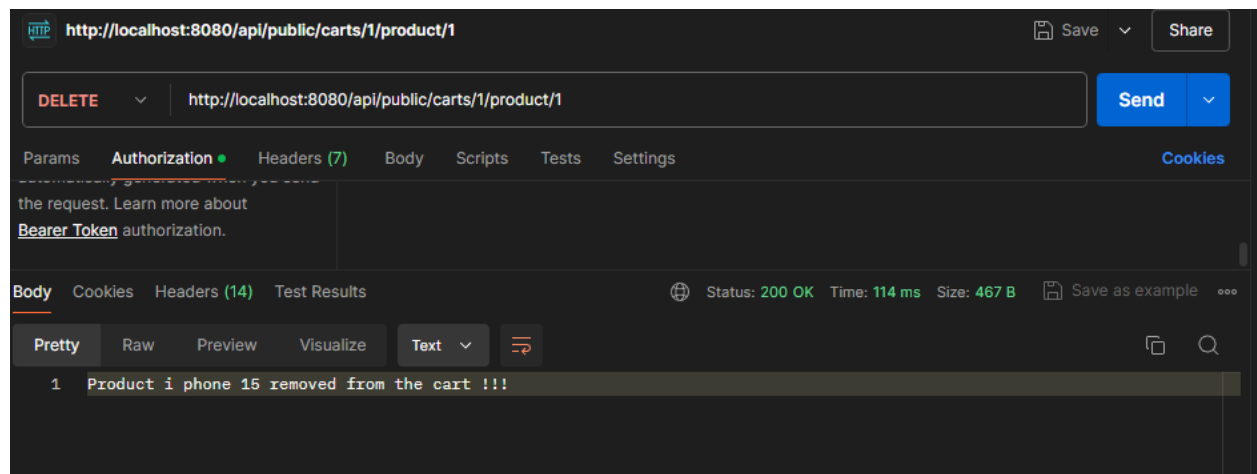


```
GET http://localhost:8080/api/admin/carts

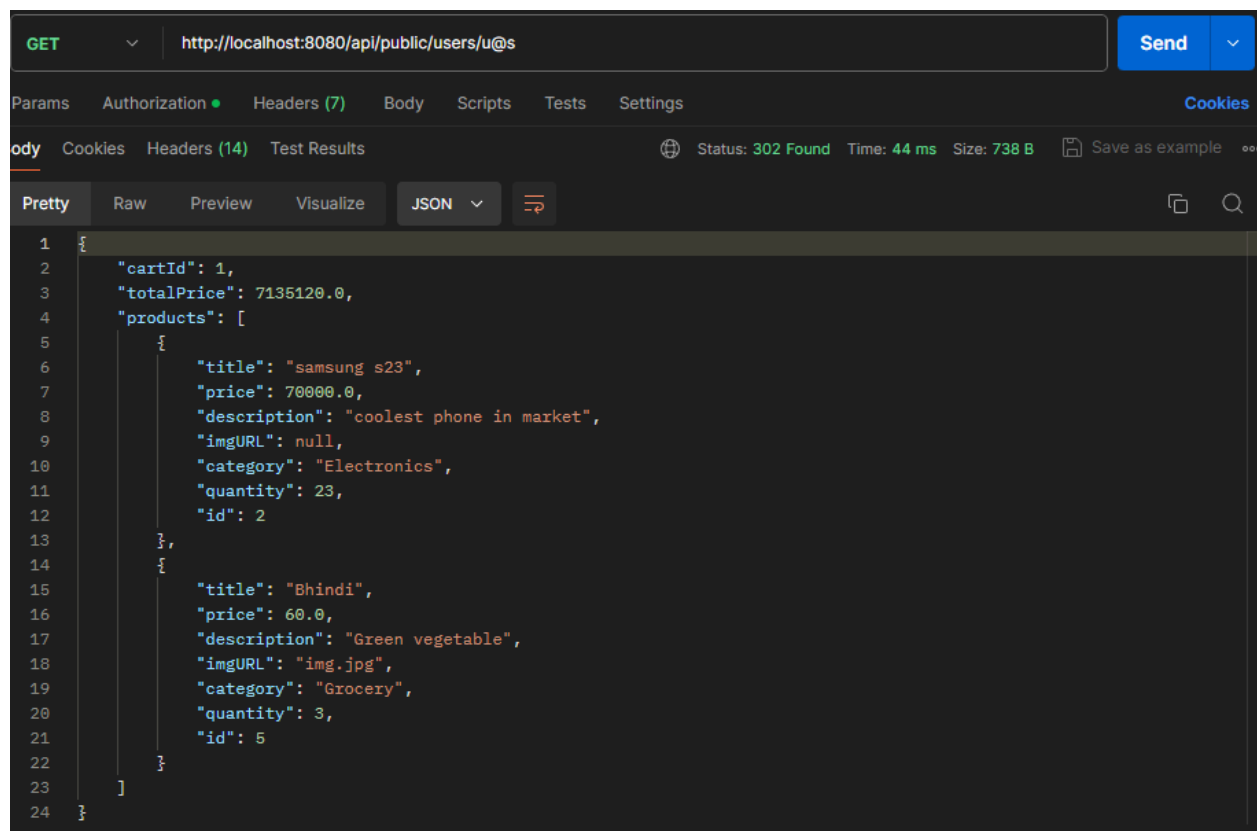
Status: 302 Found Time: 81 ms Size: 1002 B

[
  {
    "cartId": 1,
    "totalPrice": 1735000.0,
    "products": [
      {
        "title": "i phone 15",
        "price": 55000.0,
        "description": "good phone",
        "imgURL": "default.png",
        "category": "Electronics",
        "quantity": 8,
        "id": 1
      },
      {
        "title": "samsung s23",
        "price": 70000.0,
        "description": "coolest phone in market",
        "imgURL": null,
        "category": "Electronics",
        "quantity": 4,
        "id": 2
      }
    ]
  }
]
```

Delete Product from cart



View cart by user id - one user has only one cart related to it, but can have multiple order. Once cart converted into order then new cart can be created again.



Product Services

Add Product - only admin can add a product to database.

The screenshot shows a REST client interface with a POST request to `http://localhost:8080/api/admin/categories/5/product`. The request body is a JSON object with the following fields: `title` ("Movable chair"), `imgURL` ("chair.jpg"), `description` ("Chair which can help you relax full day"), `quantity` (12), `price` (15000), and `category` ("Furniture"). The response status is 201 Created, with a time of 44 ms and a size of 590 B. The response body is a JSON object with the same fields as the request, plus an `id` field with the value 7.

```
POST http://localhost:8080/api/admin/categories/5/product

{
  "title": "Movable chair",
  "imgURL": "chair.jpg",
  "description": "Chair which can help you relax full day",
  "quantity": 12,
  "price": 15000,
  "category": "Furniture"
}
```

Body: Cookies Headers (14) Test Results Status: 201 Created Time: 44 ms Size: 590 B Save as example

```
Pretty Raw Preview Visualize JSON
{
  "title": "Movable chair",
  "price": 15000.0,
  "description": "Chair which can help you relax full day",
  "imgURL": "chair.jpg",
  "category": "Furniture",
  "quantity": 12,
  "id": 7
}
```

Get all products: User and admin both can see all products.

The screenshot shows a REST client interface with a GET request to `http://localhost:8080/api/public/products`. The response status is 302 Found, with a time of 63 ms and a size of 1.13 KB. The response body is a JSON array of three product objects. The first object has `category` "Electronics", `quantity` 20, and `id` 4. The second object has `title` "Bhindi", `price` 60.0, `description` "Green vegetable", `imgURL` "img.jpg", `category` "Grocery", `quantity` 3, and `id` 5. The third object has `title` "Movable chair", `price` 15000.0, `description` "Chair which can help you relax full day", `imgURL` "chair.jpg", `category` "Furniture", `quantity` 12, and `id` 7.

```
GET http://localhost:8080/api/public/products

[
  {
    "category": "Electronics",
    "quantity": 20,
    "id": 4
  },
  {
    "title": "Bhindi",
    "price": 60.0,
    "description": "Green vegetable",
    "imgURL": "img.jpg",
    "category": "Grocery",
    "quantity": 3,
    "id": 5
  },
  {
    "title": "Movable chair",
    "price": 15000.0,
    "description": "Chair which can help you relax full day",
    "imgURL": "chair.jpg",
    "category": "Furniture",
    "quantity": 12,
    "id": 7
  }
]
```

Body: Cookies Headers (14) Test Results Status: 302 Found Time: 63 ms Size: 1.13 KB Save as example

Get single product: by id

GET <http://localhost:8080/api/public/products/5> Send

Params Authorization Headers (7) Body Scripts Tests Settings

Auth Type: Bearer Token Token: eyJhbGciOiJIUzI1NiJ9.eyJzdWIIOiJ1QHMILC...

Body Cookies Headers (14) Test Results Status: 302 Found Time: 29 ms Size: 549 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "title": "Bhindi",
3   "price": 60.0,
4   "description": "Green vegetable",
5   "imgURL": "img.jpg",
6   "category": "Grocery",
7   "quantity": 3,
8   "id": 5
9 }
```

Delete product: by Id

DELETE <http://localhost:8080/api/admin/products/5> Send

Params Authorization Headers (7) Body Scripts Tests Settings Cookies

Auth Type: Bearer Token Token: eyJhbGciOiJIUzI1NiJ9.eyJzdWIIOiJ1QHMILC...

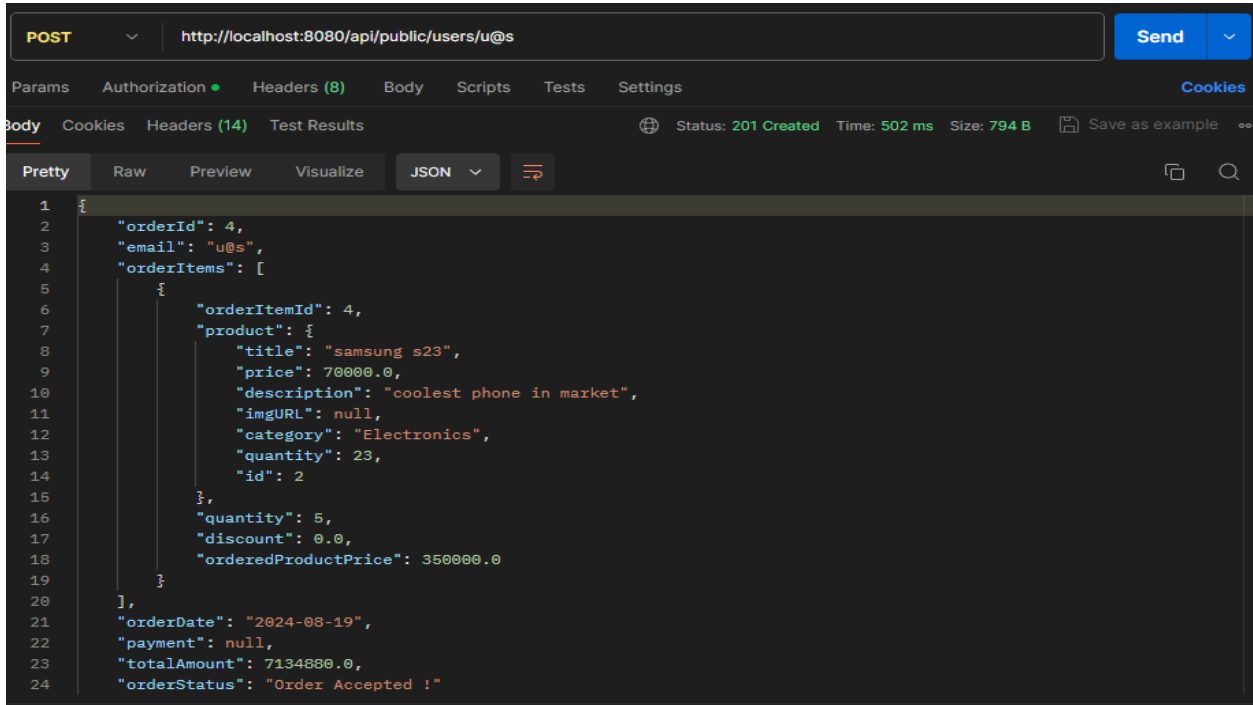
Body Cookies Headers (14) Test Results Status: 200 OK Time: 201 ms Size: 473 B Save as example

Pretty Raw Preview Visualize Text

```
1 Product with productId: 5 deleted successfully !!!
```

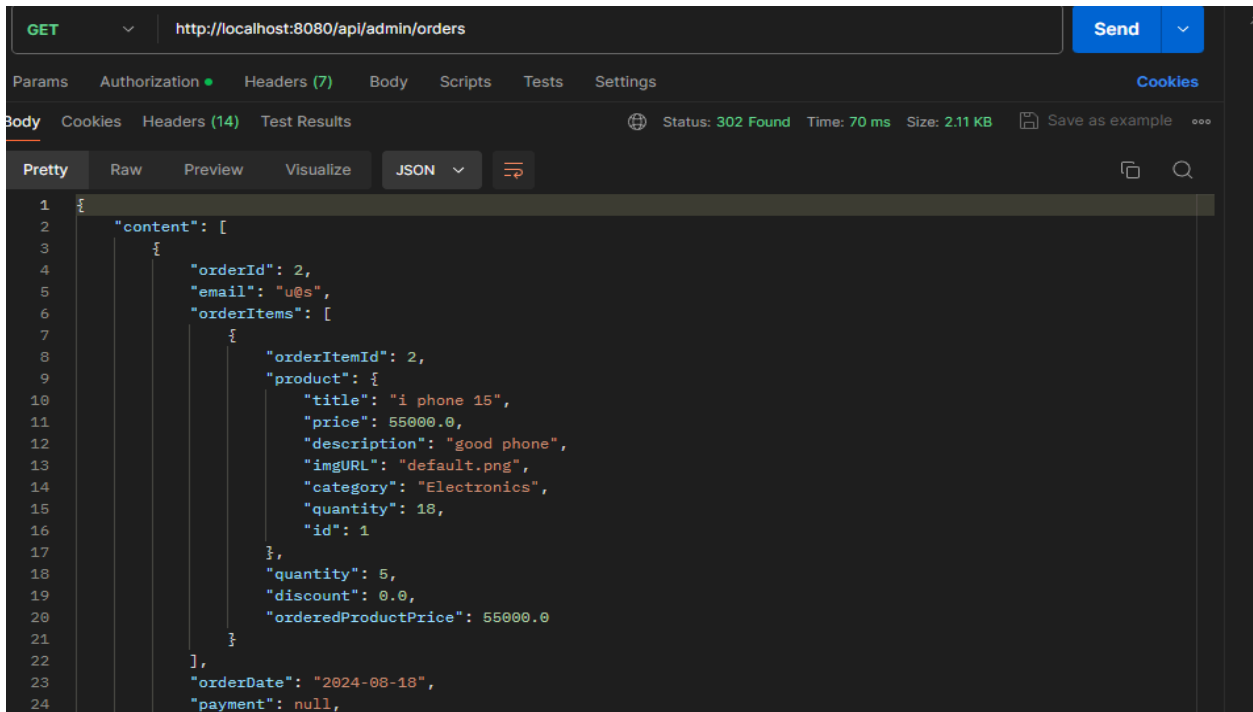

Order Services

Create a order by user id : User can create a order from the cart, once order is created , cart item gets deleted and cart becomes empty.We have created separate table for that cartitem , where the items gets deleted.



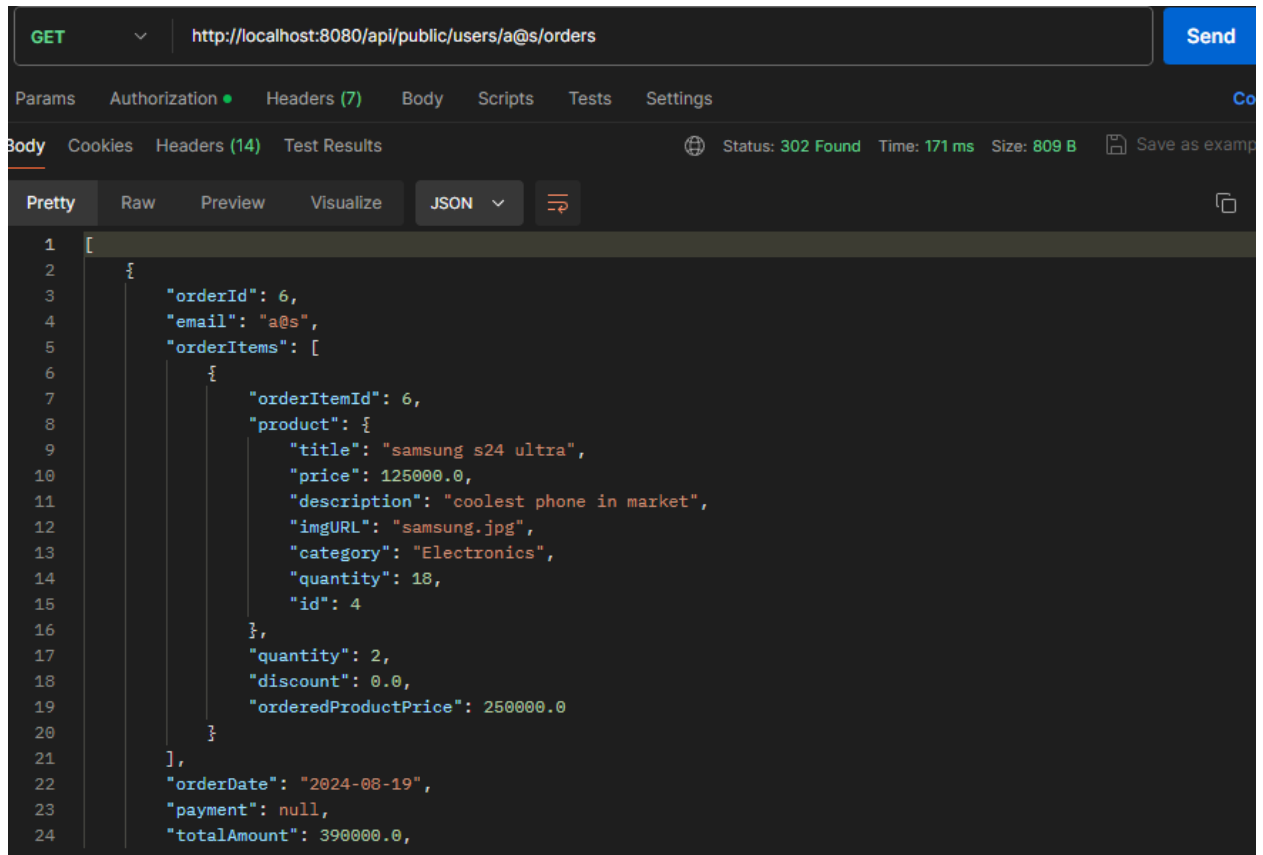
```
POST http://localhost:8080/api/public/users/u@s
Status: 201 Created Time: 502 ms Size: 794 B
JSON
{
  "orderId": 4,
  "email": "u@s",
  "orderItems": [
    {
      "orderItemId": 4,
      "product": {
        "title": "samsung s23",
        "price": 70000.0,
        "description": "coolest phone in market",
        "imgURL": null,
        "category": "Electronics",
        "quantity": 23,
        "id": 2
      },
      "quantity": 5,
      "discount": 0.0,
      "orderedProductPrice": 350000.0
    }
  ],
  "orderDate": "2024-08-19",
  "payment": null,
  "totalAmount": 7134880.0,
  "orderStatus": "Order Accepted !"
}
```

Get all the Generated Orders: we can get all the orders generated by all the users, only admin can see it.



```
GET http://localhost:8080/api/admin/orders
Status: 302 Found Time: 70 ms Size: 2.11 KB
JSON
{
  "content": [
    {
      "orderId": 2,
      "email": "u@s",
      "orderItems": [
        {
          "orderItemId": 2,
          "product": {
            "title": "i phone 15",
            "price": 55000.0,
            "description": "good phone",
            "imgURL": "default.png",
            "category": "Electronics",
            "quantity": 10,
            "id": 1
          },
          "quantity": 5,
          "discount": 0.0,
          "orderedProductPrice": 55000.0
        }
      ],
      "orderDate": "2024-08-18",
      "payment": null
    }
  ]
}
```

Get the orders by user id : All the users can see their order by their user id



The screenshot shows a REST client interface with a GET request to `http://localhost:8080/api/public/users/a@s/orders`. The response is a JSON array containing one object for user 'a@s'. The object includes an 'orderId' of 6, an 'email' of 'a@s', and an 'orderItems' array with one item. The item has an 'orderId' of 6, a 'product' object with details like 'samsung s24 ultra' and a price of 125000.0, and a 'quantity' of 2. The total amount for the order is 390000.0.

```
1 [
2   {
3     "orderId": 6,
4     "email": "a@s",
5     "orderItems": [
6       {
7         "orderId": 6,
8         "product": {
9           "title": "samsung s24 ultra",
10          "price": 125000.0,
11          "description": "coolest phone in market",
12          "imgURL": "samsung.jpg",
13          "category": "Electronics",
14          "quantity": 18,
15          "id": 4
16        },
17        "quantity": 2,
18        "discount": 0.0,
19        "orderedProductPrice": 250000.0
20      }
21    ],
22    "orderDate": "2024-08-19",
23    "payment": null,
24    "totalAmount": 390000.0,
```