

Name : G.Siva Rama Krishna

Roll No : 23BCS10153

Documentation for Building a Spring Boot Application for a Cart Service

Implemented methods:

- 1) Get all carts
- 2) Get a Cart by ID
- 3) Get a User's Carts by ID
- 4) Get carts in a date range
- 5) Add a new cart
- 6) Update a cart
- 7) Delete a cart

Setup

Open The project in IntelliJ Ultimate and run the project

<http://localhost:8080>

You will be greeted with "Welcome"

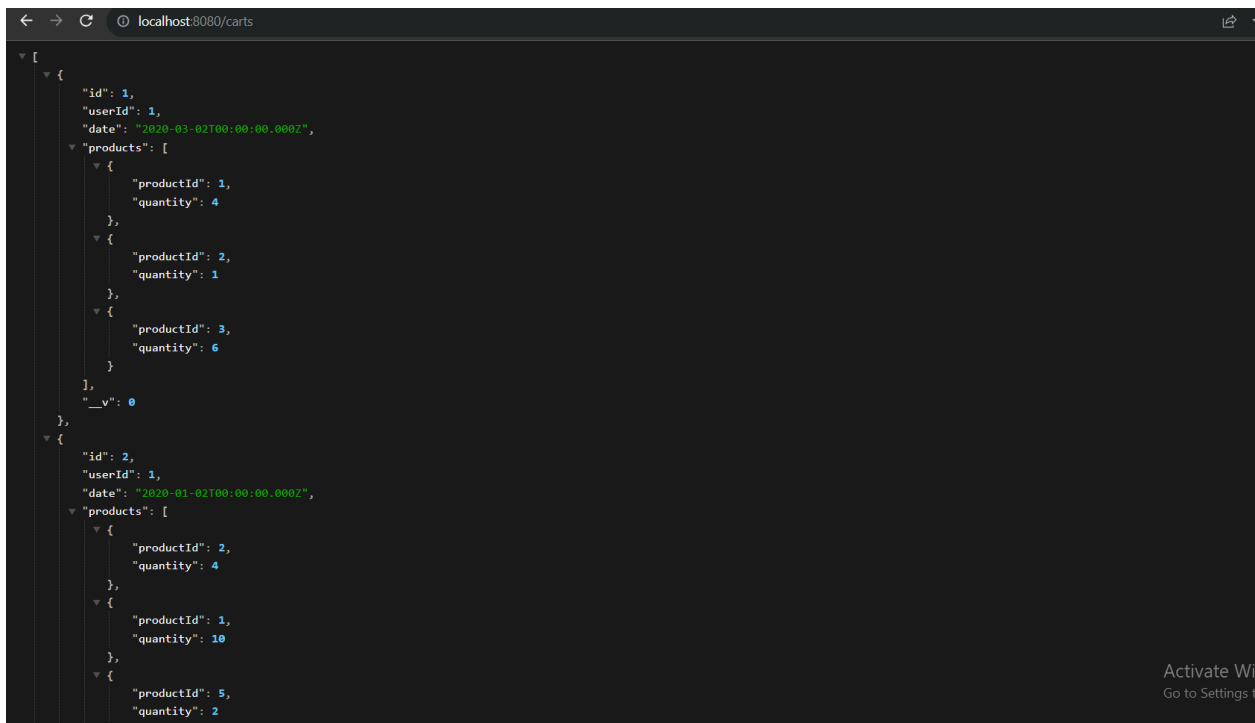
You are recommended to install JSON formatter extension for better understanding of the output

<https://web.postman.co/workspace>

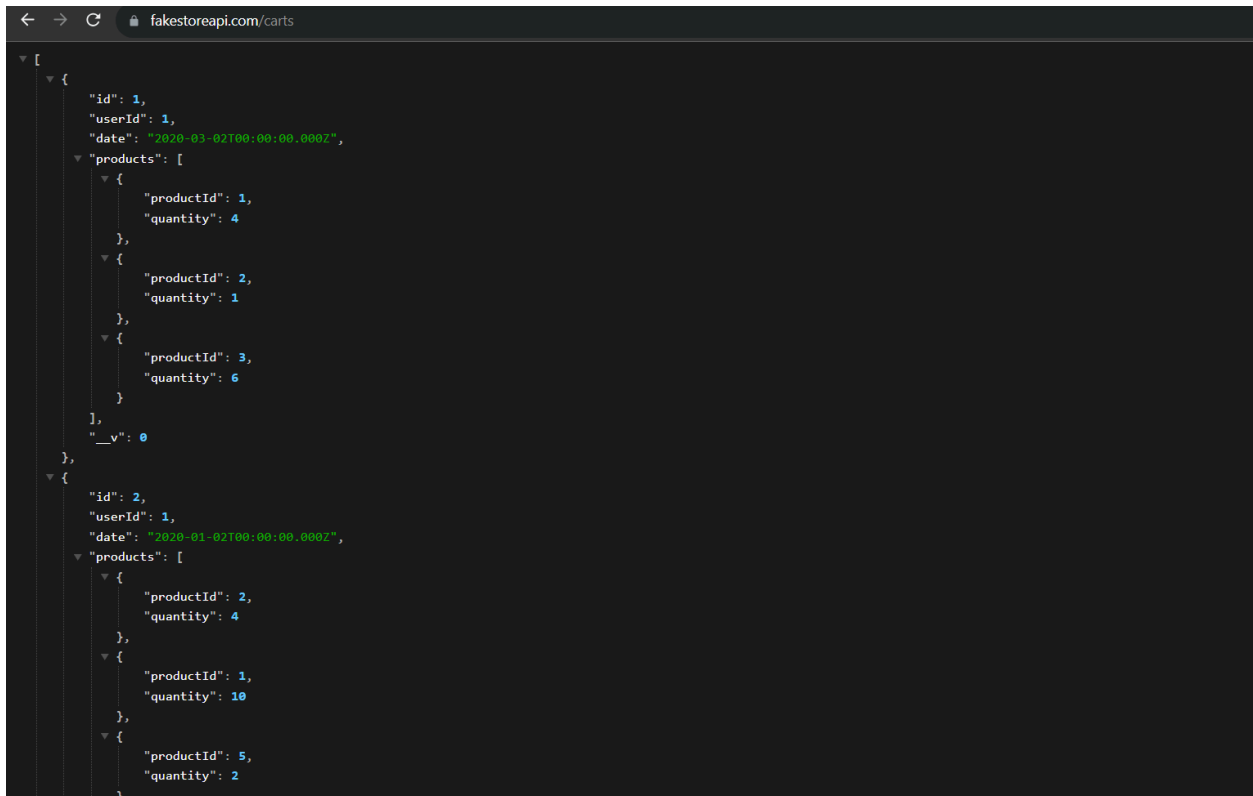
Open this to test function of adding, updating and deleting a cart.

1) Get all Carts

For getting all the carts We are mapping it to “/carts” if we find “/carts” in the url , we will Call the method of ProductService class “getCarts()” And this will use RestTemplate’s method “getForObject()” passing the url “<https://fakestoreapi.com/carts>” and extract the objects from there and returns to our localhost. You can verify this by opening <http://localhost:8080/carts> .



```
[{"id": 1, "userId": 1, "date": "2020-03-02T00:00:00.000Z", "products": [{"productId": 1, "quantity": 4}, {"productId": 2, "quantity": 1}, {"productId": 3, "quantity": 6}], "__v": 0}, {"id": 2, "userId": 1, "date": "2020-01-02T00:00:00.000Z", "products": [{"productId": 2, "quantity": 4}, {"productId": 1, "quantity": 10}, {"productId": 5, "quantity": 2}], "__v": 0}]
```



```
[{"id": 1, "userId": 1, "date": "2020-03-02T00:00:00.000Z", "products": [{"productId": 1, "quantity": 4}, {"productId": 2, "quantity": 1}, {"productId": 3, "quantity": 6}], "_v": 0}, {"id": 2, "userId": 1, "date": "2020-01-02T00:00:00.000Z", "products": [{"productId": 2, "quantity": 4}, {"productId": 1, "quantity": 10}, {"productId": 5, "quantity": 2}]}
```

2) Get a Cart by ID

For getting a cart of an id , We are mapping it to “/carts/{id}” if we find “/carts/{id}” in the url (Where {id} is a number we are expecting), we will Call the method of ProductService class “getCartById()” And this will use RestTemplate’s method “getForObject()” passing the url “<https://fakestoreapi.com/carts/{id}>” and extract the objects from there and returns to our localhost. You can verify this by opening <http://localhost:8080/carts/5> . And compare it with <https://fakestoreapi.com/carts/5>
Feel free to use any number other than 5

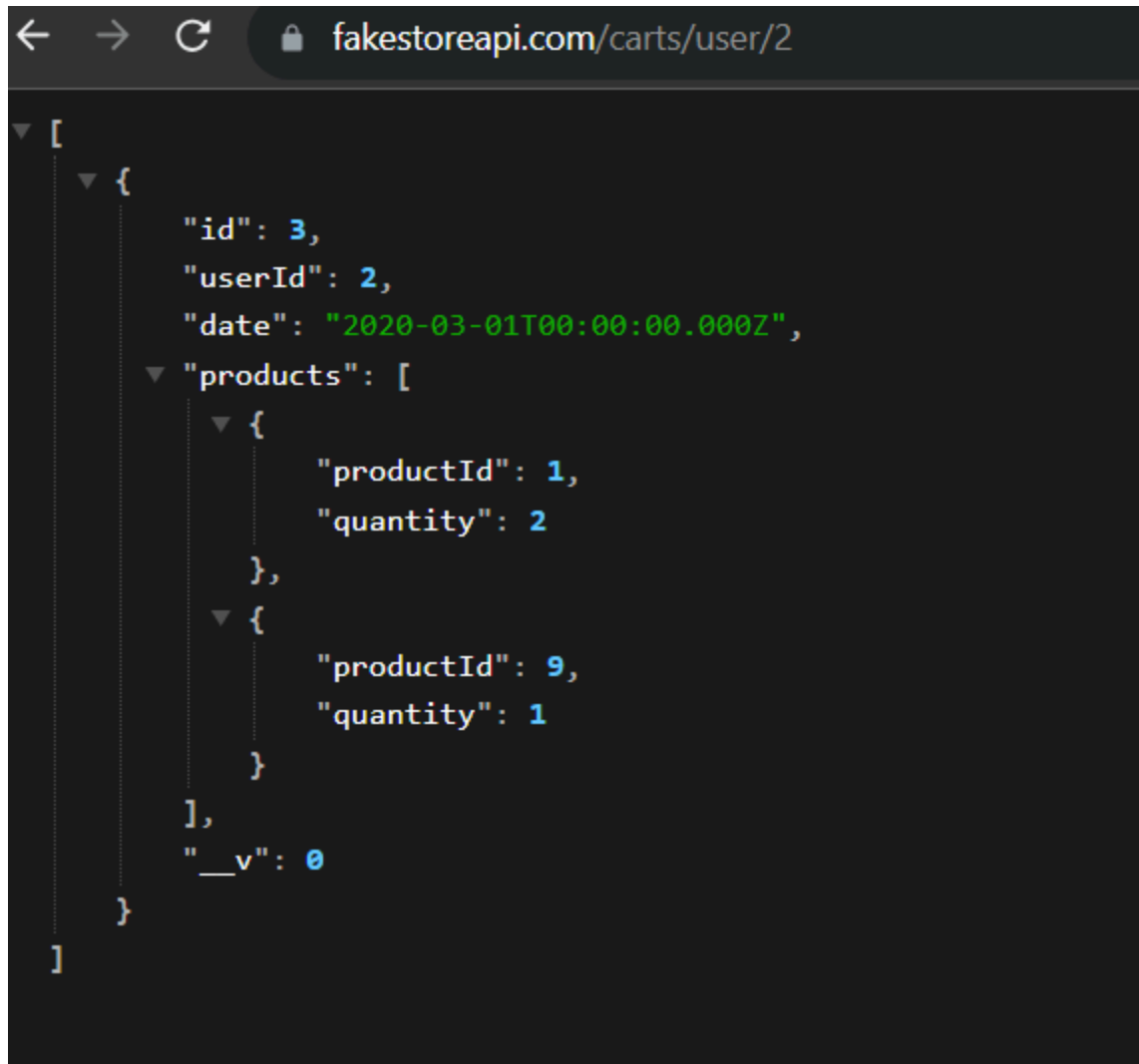
```
localhost:8080/carts/5
{
  "id": 5,
  "userId": 3,
  "date": "2020-03-01T00:00:00.000Z",
  "products": [
    {
      "productId": 7,
      "quantity": 1
    },
    {
      "productId": 8,
      "quantity": 1
    }
  ],
  "__v": 0
}
```

```
fakestoreapi.com/carts/5
{
  "id": 5,
  "userId": 3,
  "date": "2020-03-01T00:00:00.000Z",
  "products": [
    {
      "productId": 7,
      "quantity": 1
    },
    {
      "productId": 8,
      "quantity": 1
    }
  ],
  "__v": 0
}
```

3) Get a User's cart By ID

For getting all the carts We are mapping it to `"/carts/user/{id}"` if we find `"/carts/user/{id}"` in the url (Where `{id}` is a number we are expecting), we will Call the method of ProductService class `"getUserById()"` And this will use RestTemplate's method `"getForObject()"` passing the url `"https://fakestoreapi.com/carts/user/{id}"` and extract the objects from there and returns to our localhost. You can verify this by opening `http://localhost:8080/carts/user/2` And compare it with `https://fakestoreapi.com/carts/user/2`
Feel free to use any number other than 2

```
[
  {
    "id": 3,
    "userId": 2,
    "date": "2020-03-01T00:00:00.000Z",
    "products": [
      {
        "productId": 1,
        "quantity": 2
      },
      {
        "productId": 9,
        "quantity": 1
      }
    ],
    "__v": 0
  }
]
```



4) Get Carts in a data Range

We have implemented 3 functions for this to work. We will request for the start date and end date using “@RequestParam” annotation and store them in their individual variables. If both start date and end date are not null we will call the function `getByStartAndEndDates()` Which will take both dates as input and calls <https://fakestoreapi.com/carts?startdate={startdate}&enddate={enddate}>

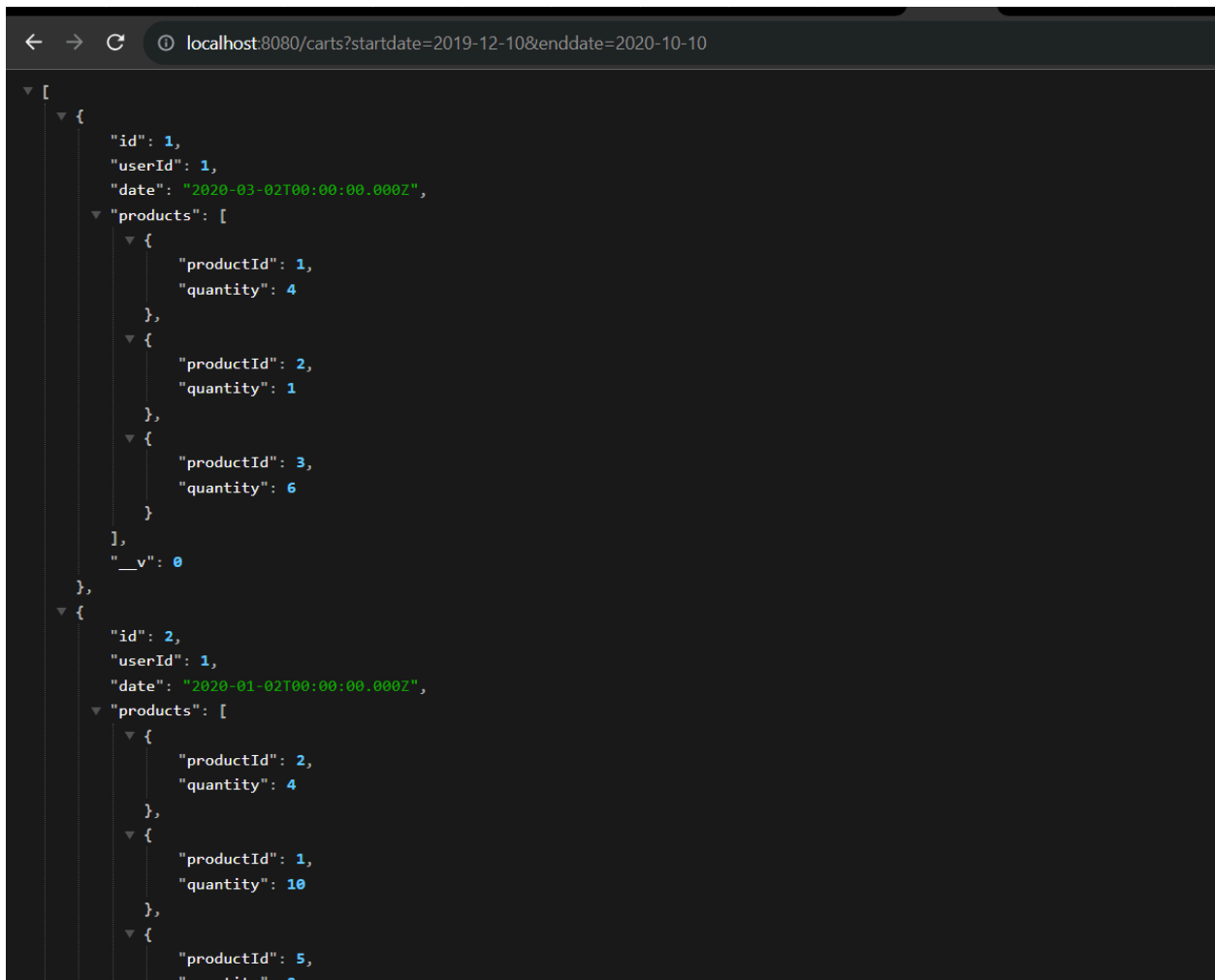
And takes it's carts and returns it.

And if either is null we would call the function and pass the parameter which is not null to get the required carts

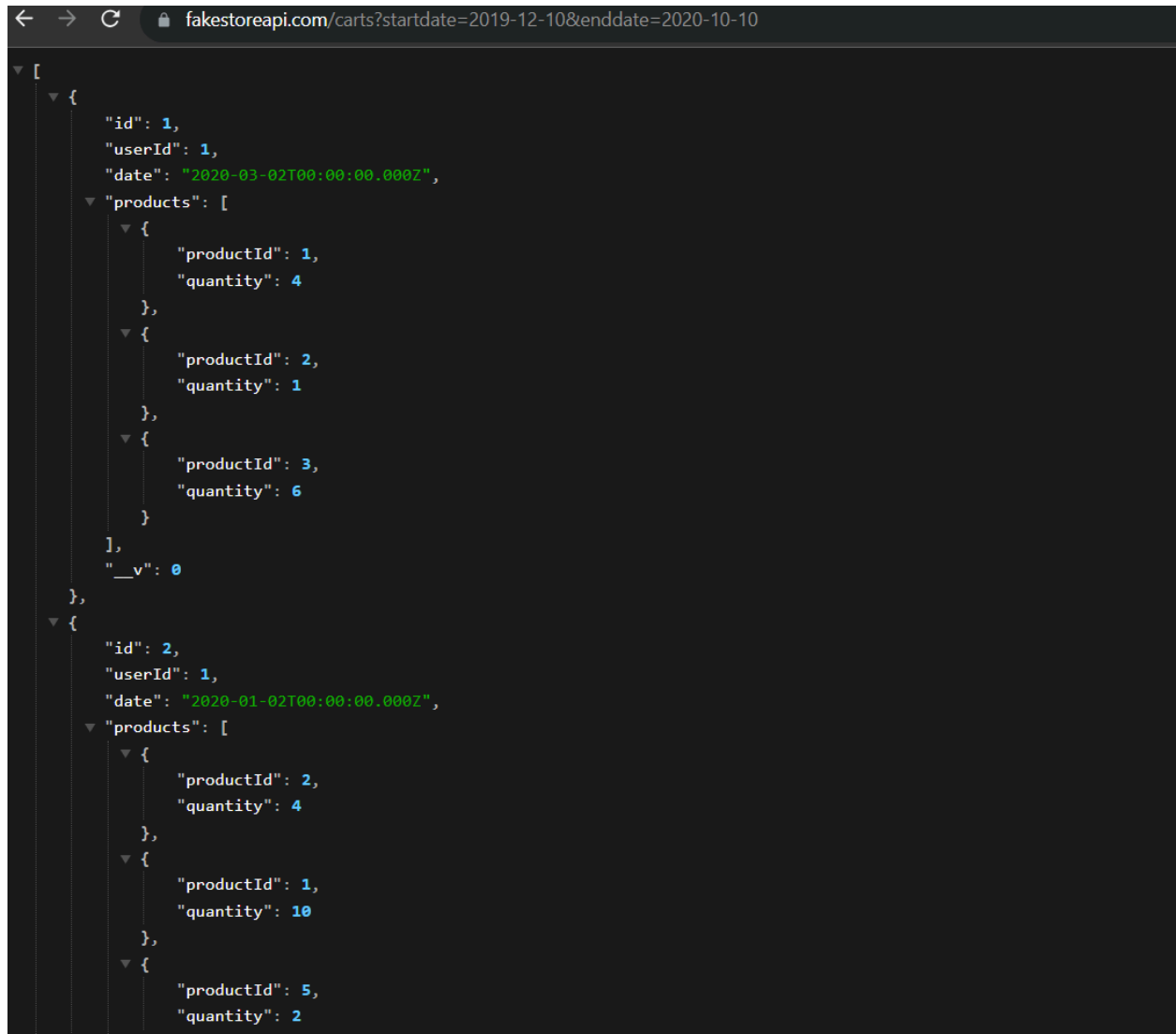
You can test this out by

<https://fakestoreapi.com/carts?startdate=2019-12-10&enddate=2020-10-10> and

<http://localhost:8080/carts?startdate=2019-12-10&enddate=2020-10-10>



```
[{"id": 1, "userId": 1, "date": "2020-03-02T00:00:00.000Z", "products": [{"productId": 1, "quantity": 4}, {"productId": 2, "quantity": 1}, {"productId": 3, "quantity": 6}], "__v": 0}, {"id": 2, "userId": 1, "date": "2020-01-02T00:00:00.000Z", "products": [{"productId": 2, "quantity": 4}, {"productId": 1, "quantity": 10}, {"productId": 5, "quantity": 2}]}
```



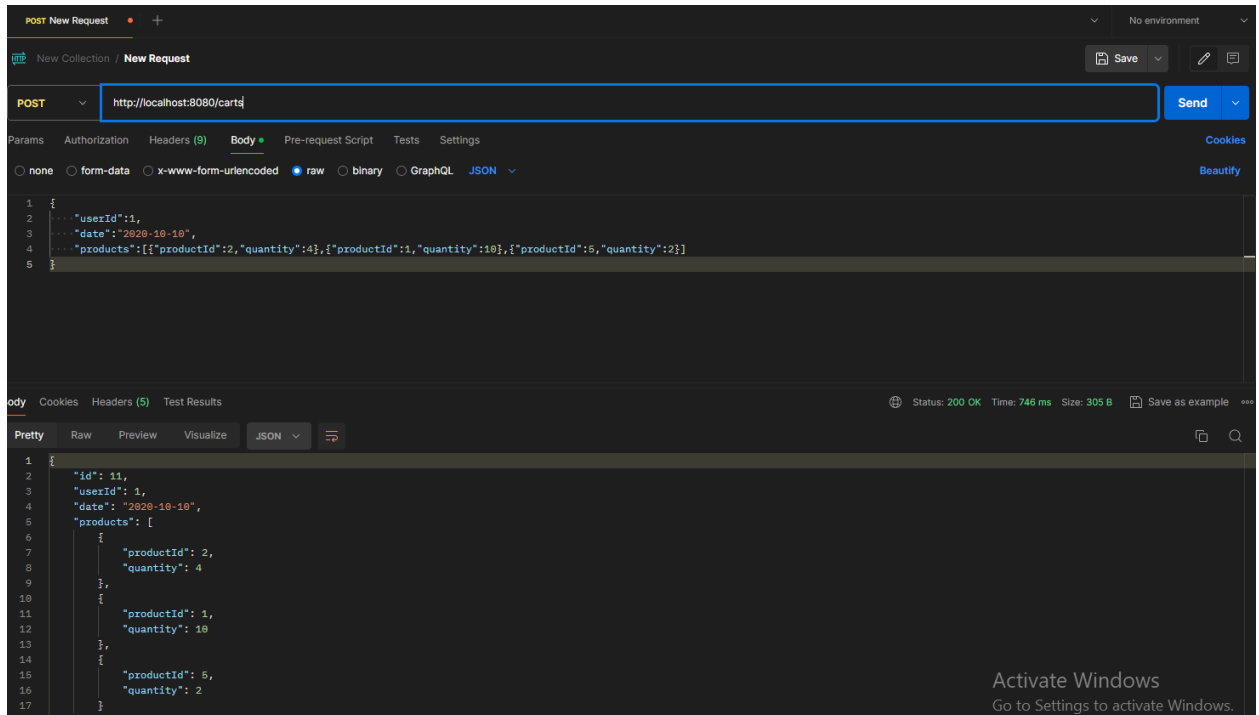
5) Add a new Cart

We will map it to “/carts” and Use annotation “@RequestBody” to send the cart to add to carts. And using RequestTemplate postForObject() method We will add the new cart to the list.

In postman we will select the Post method to add a new cart
Give the link <http://localhost:8080/carts>

Open the body and go to raw

Enter the details of the cart you want to add

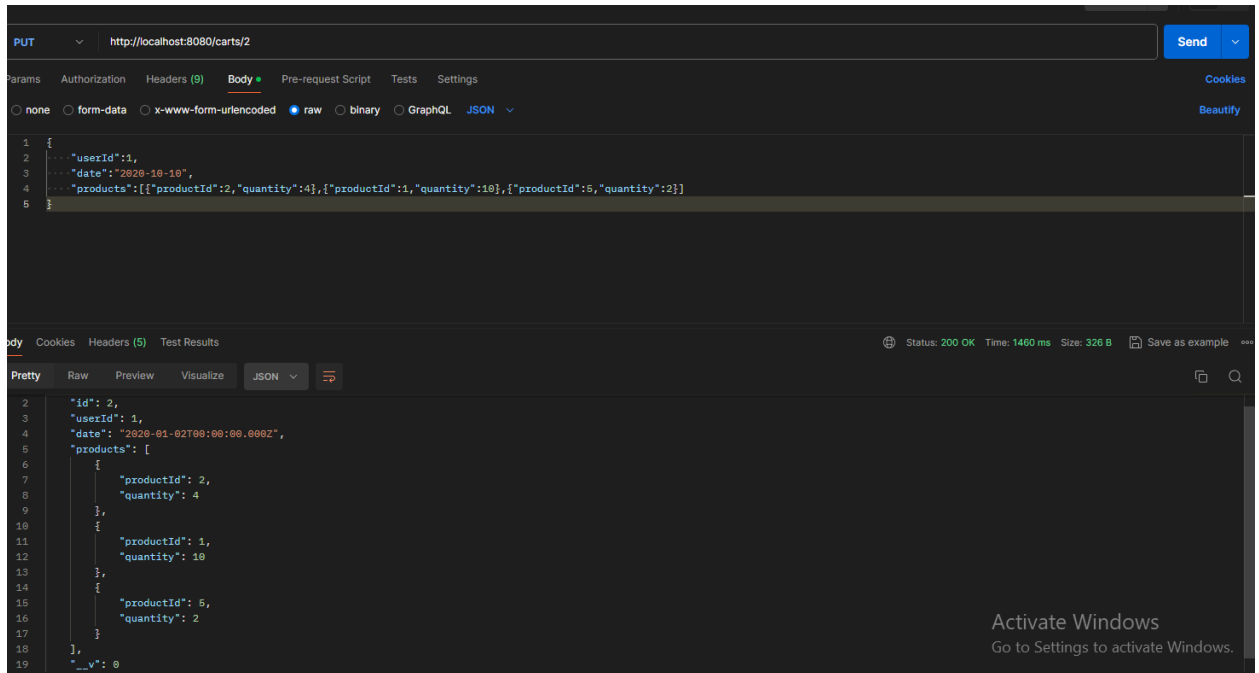


6) Edit a Cart

We will map it to `"/carts/{id}"` and Use annotation `"@RequestBody"` to send the cart to edit. And using `RequestTemplate put()` method We will edit cart.

In postman we will select the Put method to edit a cart
Give the link <http://localhost:8080/carts/{id}>

Open the body and go to raw
Enter the details of the cart you want to edit



Here we successfully edited the 2nd cart

7) Delete a Cart

We will map it to `"/carts/{id}"` and. And using RequestTemplate delete() method We will delete cart.

In postman we will select the Delete method to edit a cart
Give the link <http://localhost:8080/carts/{id}>

