Testing the API of grocery stores by using different API requests such as: GET, POST, PATCH, and DELETE.

This test has done 8 different api calls:

1.Test name: Checking API valid

Purpose of the test: Check the API is valid and running on the cloud.

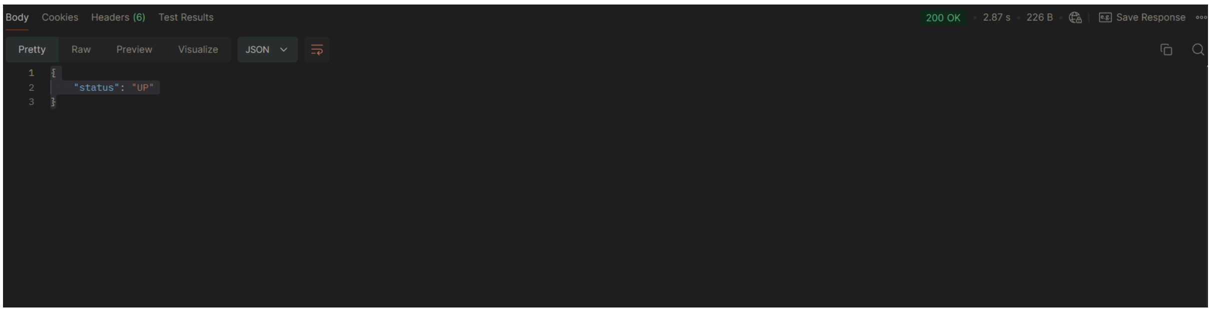
Request

METHOD: GET

URL:https://simple-grocery-store-api.glitch.me/status

Response data: {   "status": "UP"}

screenshot of the result:



```
{
    "status": "UP"
}
```

type of response: 200 OK

2.Test name: Get all grocery products

Purpose of the test: get all the grocery products

Request

METHOD: GET

URL:https://simple-grocery-store-api.glitch.me/products

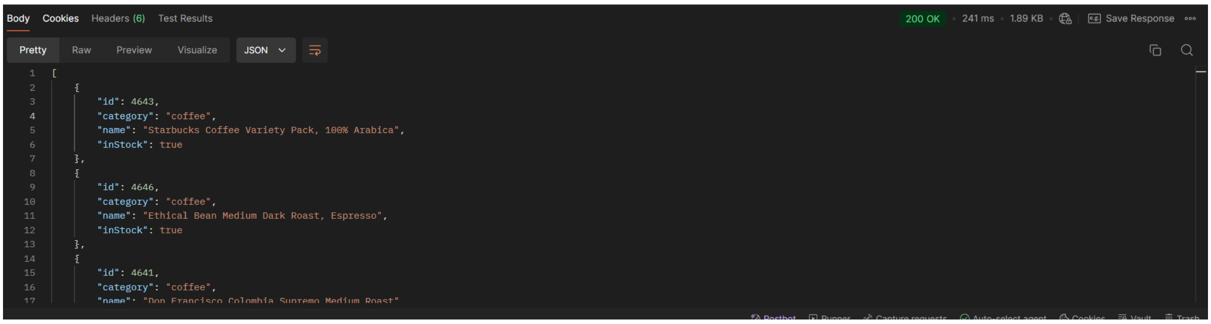
Response data:[

```
{
    "id": 4643,
    "category": "coffee",
    "name": "Starbucks Coffee Variety Pack, 100% Arabica",
    "inStock": true
},
{
    "id": 4646,
    "category": "coffee",
    "name": "Ethical Bean Medium Dark Roast, Espresso",
    "inStock": true
},
{
    "id": 4641,
    "category": "coffee",
    "name": "Don Francisco Colombia Supremo Medium Roast",
    "inStock": true
```

```json
  },
  {
    "id": 1225,
    "category": "fresh-produce",
    "name": "1/2 in. Brushless Hammer Drill",
    "inStock": true
  },
  {
    "id": 3674,
    "category": "fresh-produce",
    "name": "20V Max Cordless Drill Combo Kit",
    "inStock": true
  },
  {
    "id": 2585,
    "category": "fresh-produce",
    "name": "Green Cabbage Organic",
    "inStock": true
  },
  {
    "id": 5851,
    "category": "fresh-produce",
    "name": "Cucumber Organic",
    "inStock": false
    },
    {
        "id": 8739,
        "category": "fresh-produce",
        "name": "Fresh Spinach Organic",
        "inStock": true
    },
    {
        "id": 2177,
        "category": "meat-seafood",
        "name": "Cosco Three Step Steel Platform",
        "inStock": true
    },
    {
        "id": 1709,
        "category": "meat-seafood",
        "name": "Beef Choice Angus Ribeye Steak",
        "inStock": true
    },
    {
        "id": 1709,
        "category": "meat-seafood",
        "name": "Angus Steak Burgers",
        "inStock": true
```

```
  },
  {
    "id": 7395,
    "category": "meat-seafood",
    "name": "Boneless Skinless Chicken Breasts",
    "inStock": true
  },
  {
    "id": 8554,
    "category": "candy",
    "name": "Kinder Joy Eggs",
    "inStock": true
  },
  {
    "id": 6483,
    "category": "candy",
    "name": "Cadbury Milk Chocolate",
    "inStock": true
  },
  {
    "id": 5774,
    "category": "candy",
    "name": "HERSHEY'S, Milk Chocolate Almond",
    "inStock": true
  },
  {
    "id": 8753,
    "category": "dairy",
    "name": "Reduced Fat Milk",
    "inStock": true
  },
  {
    "id": 9482,
    "category": "dairy",
    "name": "Whole Milk",
    "inStock": true
  },
  {
    "id": 5477,
    "category": "dairy",
    "name": "Cream Cheese",
    "inStock": true
  },
  {
    "id": 5478,
    "category": "dairy",
    "name": "Low Fat Vanilla Yogurt",
    "inStock": true
  },

  {

    "id": 4875,

    "category": "bread-bakery",

    "name": "2800 Watt Inverter Generator",

    "inStock": true

  }

]
```

screenshot of the result:



```
[
    {
        "id": 4643,
        "category": "coffee",
        "name": "Starbucks Coffee Variety Pack, 100% Arabica",
        "inStock": true
    },
    {
        "id": 4646,
        "category": "coffee",
        "name": "Ethical Bean Medium Dark Roast, Espresso",
        "inStock": true
    },
    {
        "id": 4641,
        "category": "coffee",
        "name": "Don Francisco Colombia Supremo Medium Roast"
```

type of response: 200 OK

3.Test name: Get coffee product

Purpose of the test: Get the coffee product by the id

Request

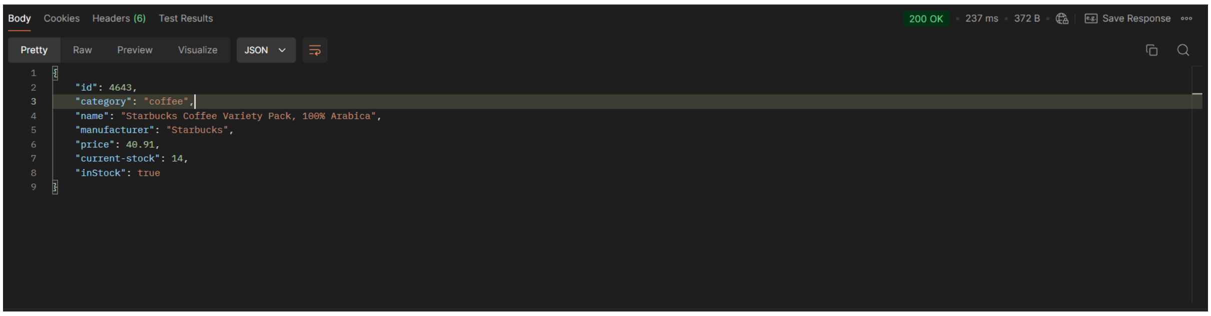
METHOD: GET

URL:https://simple-grocery-store-api.glitch.me/products/4643

Response data: {

```
  "id": 4643,

  "category": "coffee",

  "name": "Starbucks Coffee Variety Pack, 100% Arabica",

  "manufacturer": "Starbucks",

  "price": 40.91,

  "current-stock": 14,

  "inStock": true

}
```

screenshot of the result:



```
{
    "id": 4643,
    "category": "coffee",
    "name": "Starbucks Coffee Variety Pack, 100% Arabica",
    "manufacturer": "Starbucks",
    "price": 40.91,
    "current-stock": 14,
    "inStock": true
}
```

type of response: 200 OK

4.Test name: Create dynamic url cart

Purpose of the test:Create dynamic url cart

Request

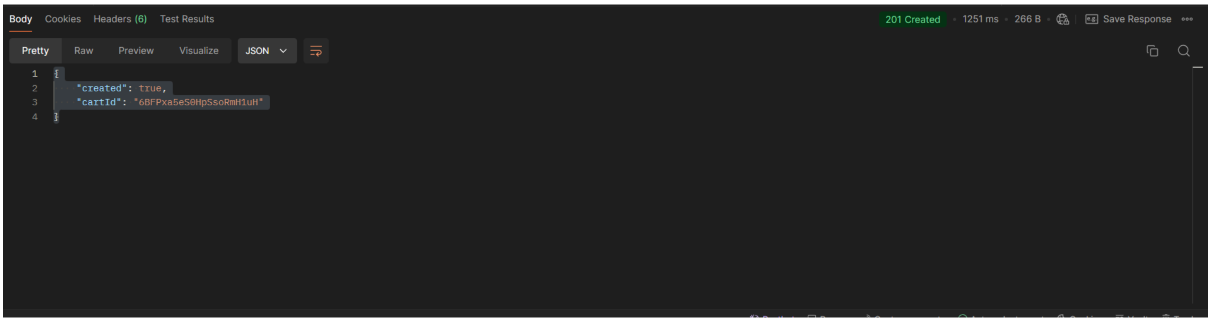
METHOD: POST

URL:https://simple-grocery-store-api.glitch.me/carts

Response data: {

```
    "created": true,

    "cartId": "6BFPxa5eS0HpSsoRmH1uH"

}
```

screenshot of the result:



```
{
    "created": true,
    "cartId": "6BFPxa5eS0HpSsoRmH1uH"
}
```

type of response: 201 CREATED

5.Test name: Add coffee to that same run cart url

Purpose of the test: Add coffee to that same run cart url

Request

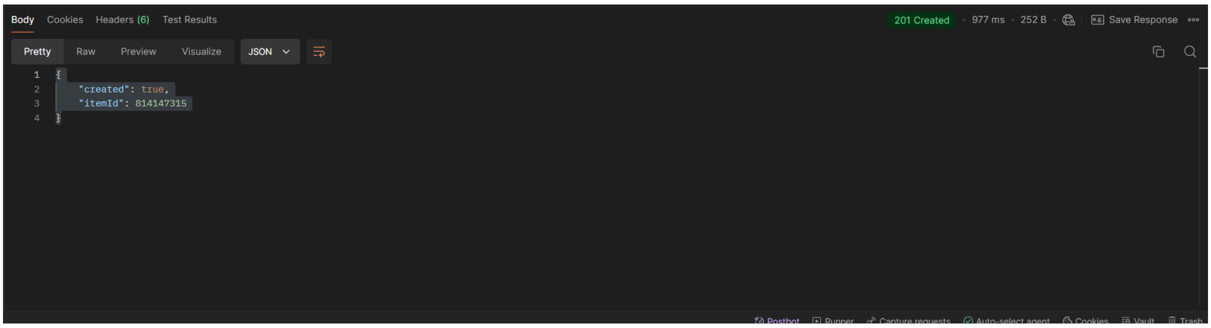
METHOD: POST

URL:https://simple-grocery-store-api.glitch.me/carts/6BFPxa5eS0HpSsoRmH1uH/items

Response data:{

```
  "created": true,

  "itemId": 814147315

}
```

screenshot of the result:



201 Created · 977 ms · 252 B

```
{
    "created": true,
    "itemId": 814147315
}
```

type of response: 201 CREATED

6.Test name: Modify the coffee quantity in the dynamic cart

Purpose of the test: Add coffee to that same run cart url

Request

METHOD: PATCH

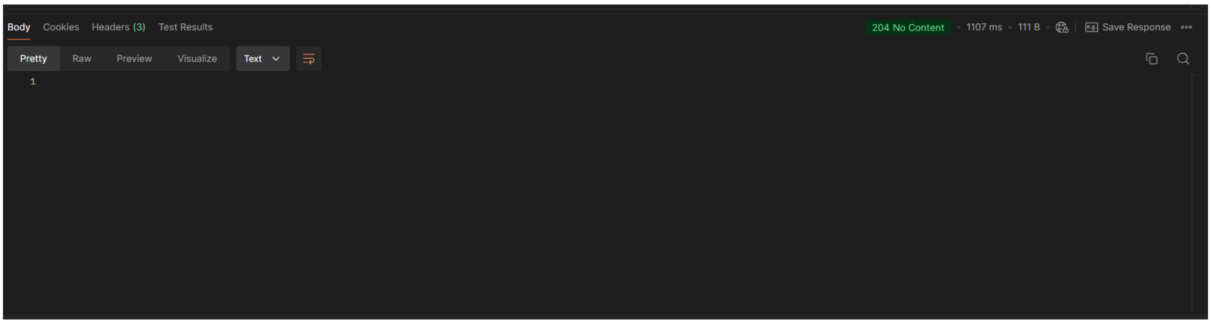
URL:https://simple-grocery-store-api.glitch.me/carts/6BFPxa5eS0HpSsoRmH1uH//items/146454867

body: {

```
    "cartId":"kyQ3dBQHiVx_8wBcU5u1n",

   "itemId": 146454867,

   "quantity": 3

}
```

Response data: empty response with 204 No Content

screenshot of the result:



type of response: 204 No Content

7.Test name: Removing the coffee from the cart

Purpose of the test: Removing the coffee from the cart

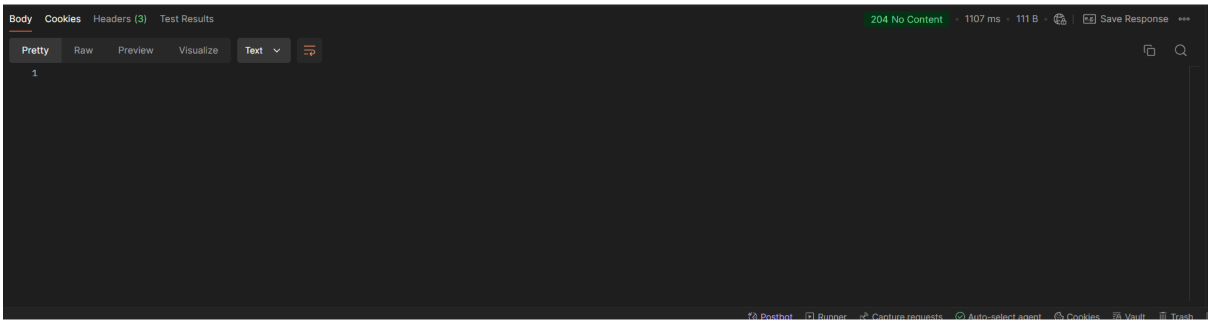
Request

METHOD: DELETE

URL:https://simple-grocery-store-api.glitch.me/carts/6BFPxa5eS0HpSsoRmH1uH//items/146454867

Response data: empty response with 204 No Content

screenshot of the result:



type of response: 204 No Content

8.Test name: Returns all the orders created by the API client

Purpose of the test: Returns all the orders created by the API client

Request

METHOD: GET

URL:https://simple-grocery-store-api.glitch.me/orders

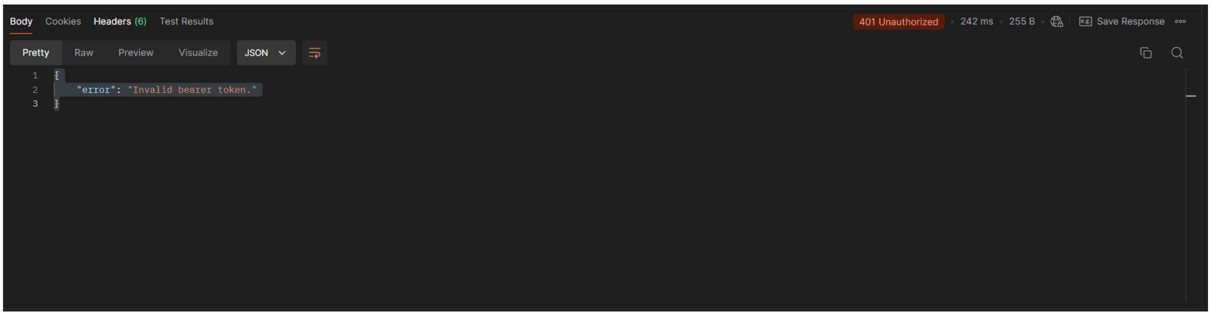
HEADERS:

Authorization: “6BFPxa5eS0HpSsoRmH1uH”

Response data:

```
{

    "error": "Invalid bearer token."

}
```

screenshot of the result:



```
{
    "error": "Invalid bearer token."
}
```

type of response:401 Unauthorized