API Documentation: GG Shopping Cart API

Welcome to the documentation for the GG Shopping Cart API. This document provides an overview of the API endpoints, request and response formats, and how to use the API to manage shopping cart activities.

Table of Contents

1. Introduction

The GG Shopping Cart API allows users to perform shopping cart management activities, such as adding products, updating cart items, fetching product details, and more. The API is secured secure and unique JWT token generated upon user authentication.

2. User signup and Authentication

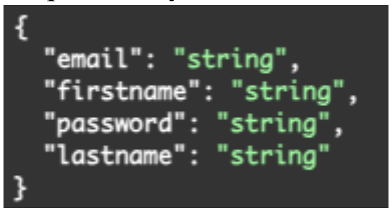
Users can log in and be authenticated. Upon a successful authentication a JWT Token is generated and returned to the user. This token must be passed in all the subsequent requests made by the user to get access to the API resources.

To obtain the Authorization token:

- Use the `/login` endpoint to obtain an access token.
- Include the received token in the `Authorization` header as `Bearer <access_token>` for subsequent requests.

a. Register

- Endpoint: `POST /api/users/register`
- Description: Register a new user.
- Request Body:

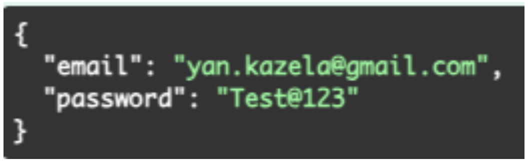


```
{
  "email": "string",
  "firstname": "string",
  "password": "string",
  "lastname": "string"
}
```

- Response: Returns user registration status.

b. Login

- Endpoint: `POST /api/users/register`
- Description: Register a new user.
- Request Body:



```
{
  "email": "yan.kazela@gmail.com",
  "password": "Test@123"
}
```

- Response: User data and jwt Token.

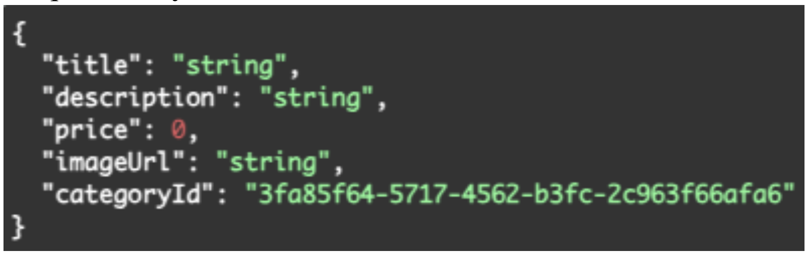
c. Logout

- Endpoint: `POST /api/users/register`
- Description: Logins the user.
- Request Body: null
- Response: Logout status

3. Products

a. Post Product

- Endpoint: `POST /api/products`
- Description: Adds a new product.
- Request Body:



```
{
  "title": "string",
  "description": "string",
  "price": 0,
  "imageUrl": "string",
  "categoryId": "3fa85f64-5717-4562-b3fc-2c963f66afa6"
}
```

- Response: Created product with Id property

b. Get Products

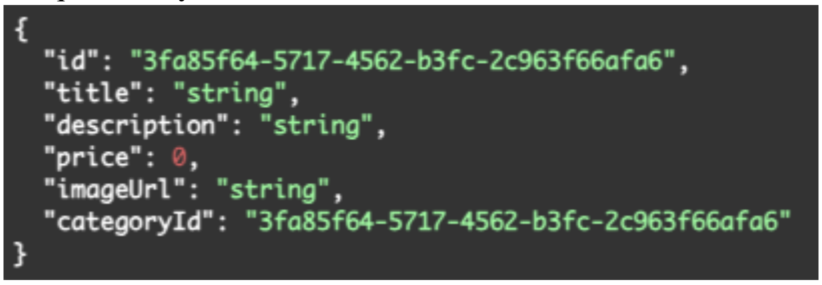
- Endpoint: **GET** `/api/products.`
- Description: Gets the list of products with pagination.
- Query params: pageNumber and pageSize
- Response: Returns paginated list of products.

c. Get Product

- Endpoint: **GET** `/api/products/{id}`
- Description: Returns a single product.
- Response: Returns a single product.

d. Put Product

- Endpoint: `PUT /api/products`
- Description: Updates an existing product.
- Request Body:



```
{
  "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "title": "string",
  "description": "string",
  "price": 0,
  "imageUrl": "string",
  "categoryId": "3fa85f64-5717-4562-b3fc-2c963f66afa6"
}
```

- Response: Updated product

e. **Delete Product**

- Endpoint: **GET** `/api/products/{id}`
- Description: Delete an existing product.
- Response: Returns response status.

f. Upload Product Image

- Endpoint: **POST** `/api/products/uploads`
- Description: uploads an image.
- Request body: this endpoint accepts a file binary.
- Response: Returns the url to attach to a product.

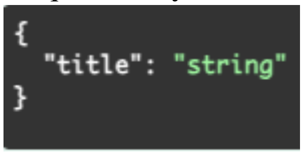
g. Get Product Image

- Endpoint: **GET** `/api/products/images/{imageName}`
- Description: Retrieves an uploaded image.
- Response: Product Image

4. Categories

a. **Post Category**

- Endpoint: `POST /api/categories`
- Description: Adds a new category.
- Request Body:



```
{
  "title": "string"
}
```

- Response: Created category with Id property

b. **Get Categories**

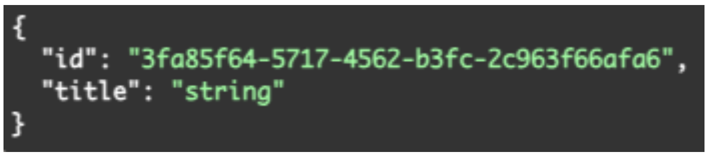
- Endpoint: **GET** `/api/categories`
- Description: Gets the list of categories.
- Response: Returns list of categories.

c. **Get Category**

- Endpoint: **GET** `/api/categories/{id}`
- Description: Returns a single category.
- Response: Returns a single category.

d. **Put Category**

- Endpoint: `PUT /api/categories`
- Description: Updates an existing category.
- Request Body:



```json
{
  "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "title": "string"
}
```

- Response: Updated category

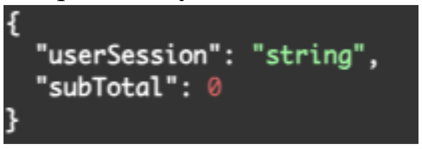
e. Delete Category

- Endpoint: **GET** `/api/category/{id}`
- Description: Deletes an existing category.
- Response: Returns response status.

5. Carts and Line Items

a. Post Cart

- Endpoint: `POST /api/carts/cart`
- Description: Adds a new cart using the client app session.
- Request Body:



```json
{
  "userSession": "string",
  "subTotal": 0
}
```

- Response: Created product with Id property

b. Get Cart

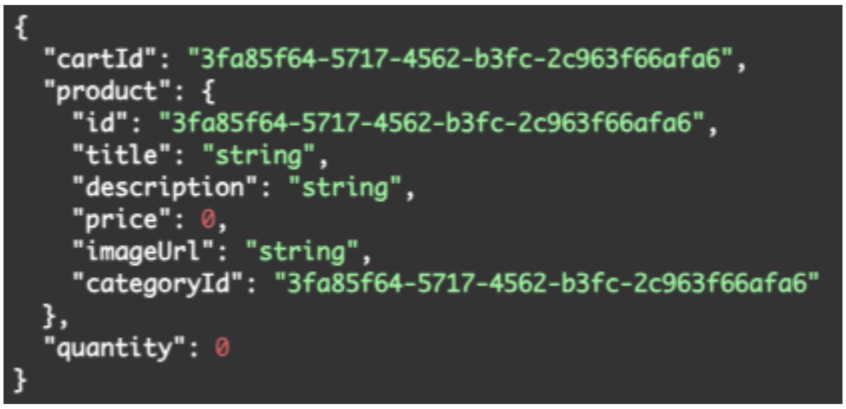
- Endpoint: **GET** `/api/carts`
- Description: Returns a single cart associated to the usersession sent in the request headers.
- Response: Returns a single cart.

c. Delete Cart

- Endpoint: **GET** `/api/carts`
- Description: Deletes a single cart associated to the usersession sent in the.
- Response: Returns response status.

d. Post Line Item

- Endpoint: `POST /api/carts/lineitem`
- Description: Adds a new item to a cart.
- Request Body:

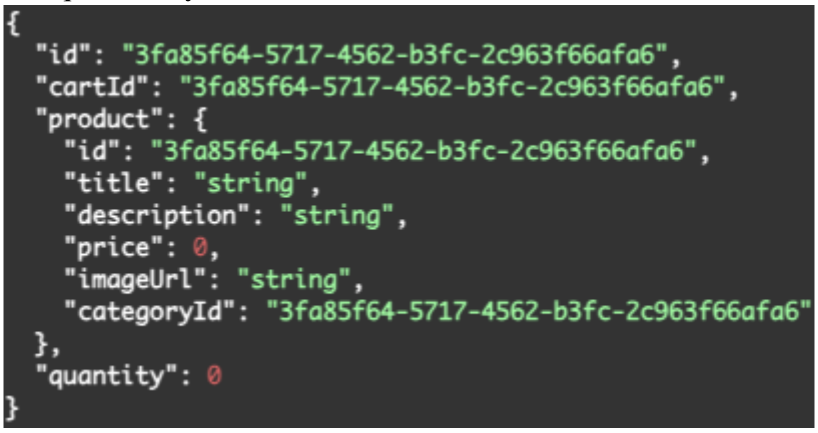


```json
{
  "cartId": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "product": {
    "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
    "title": "string",
    "description": "string",
    "price": 0,
    "imageUrl": "string",
    "categoryId": "3fa85f64-5717-4562-b3fc-2c963f66afa6"
  },
  "quantity": 0
}
```

- Response: Updated cart with the new item

e. Put Line Item

- Endpoint: **PUT** `/api/carts/lineitem`
- Description: Updates an existing item
- Request Body:



```json
{
  "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "cartId": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "product": {
    "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
    "title": "string",
    "description": "string",
    "price": 0,
    "imageUrl": "string",
    "categoryId": "3fa85f64-5717-4562-b3fc-2c963f66afa6"
  },
  "quantity": 0
}
```

- Response: Returns the updated line item.

f. Delete Line Item

- Endpoint: **GET** `/api/carts/lineitem/{id}`
- Description: Deletes an item from a cart.
- Response: Returns response status.

6. Error Handling

In case of errors, the API will return appropriate HTTP status codes along with error messages. The response format will include:

- `status`: HTTP status code
- `message`: Error message