Laravel Api Documentation

Overview:

This API provides functionality for managing products and orders in an e-commerce system. It allows both admin and customer roles to interact with the system based on their permissions.

Base Url:

http://127.0.0.1:8000/api

Authentication

Token-Based Authentication: All endpoints, except for registration and login, require a Bearer token for access. This token is provided upon successful login and must be included in the Authorization header of each request.

1. Register a New User

- Endpoint: /register
- Method: POST
- Description: Registers a new customer user.
- Request Body:
  - name: String (required) - The name of the user.
  - email: String (required) - The email of the user (must be unique).
  - password: String (required) - The password for the user account.
- Response:

  Success(201):

  ```
  {
    "status": true,
    "message": "User Registered Successfully",
    "token": "API TOKEN"
  }
  ```

Error (400)

```
{
  "status": false,
  "message": "Validation error",
  "errors": {
    "email": ["The email has already been taken."]
  }
}
```

2. User Login

- Endpoint: /login
- Method: POST
- Description: Logs in a user and returns an access token.
- Request Body:

  email: String (required) - User's email.

  password: String (required) - User's password.

- Response:

  Success

```
{
  "status": true,
  "message": "User Login Successful",
  "token": "API TOKEN"
}
```

  Error

```
{
  "status": false,
  "message": "Invalid Credentials"
}
```

3. Logout

- Endpoint: /logout
- Method: GET
- Description: Logs out the authenticated user.
- Headers:
    - Authorization: Bearer {token}
- Response:

  Success (200)

  {

  "status": true,

  "message": "User Logout Successful"

  }

User Profile Management

- Get User Profile
- Endpoint: /profile
- Method: GET
- Description: Retrieves the profile information of the authenticated user.
- Headers:
    - Authorization: Bearer {token}
- Response:

  Success (200):

  {

  "status": true,

  "message": "Profile Information",

  "data": {

  "id": 1,

  "name": "User Name",

  "email": "user@example.com"

}
}

Product Management

1. List All Products

- Endpoint: /products
- Method: GET
- Description: Fetches a list of all products.
- Headers:
- Authorization: Bearer {token} (Admin only)
- Response:

Success (200):[
{
"id": 1,
"name": "Product Name",
"quantity": 10,
"description": "Product Description",
"image": "http://127.0.0.1:8000/storage/uploads/images/product.png"
}
]

2. Add a New Product

- Endpoint: /products
- Method: POST
- Description: Adds a new product to the inventory (Admin only).
- Headers:
    - Authorization: Bearer {token}
- Request Body (form-data):

name: String (required) - Name of the product.
quantity: Integer (required) - Available quantity.
description: String (optional) - Description of the product.
image: File (optional) - Product image file.

- Response:

  Success (201):


```
{
  "message": "Product created successfully",
  "product": {
    "id": 1,
    "name": "New Product",
    "quantity": 10,
    "description": "New Description",
    "image": "http://127.0.0.1:8000/storage/uploads/images/product.png"
  }
}
```


3. View a Single Product

Endpoint: /products/{id}

Method: GET

Description: Retrieves a specific product by its ID.

Headers:

Authorization: Bearer {token} (Admin only)

Response:

Success (200):


```
{
  "id": 1,
  "name": "Product Name",
  "quantity": 10,
  "description": "Product Description",
  "image": "http://127.0.0.1:8000/storage/uploads/images/product.png"
}
```


4. Update a Product

Endpoint: /products/{id}

Method: PUT

Description: Updates a product's information (Admin only).

Headers:

Authorization: Bearer {token}

Request Body (form-data):

name: String (optional) - Updated product name.

quantity: Integer (optional) - Updated quantity.

description: String (optional) - Updated description.

image: File (optional) - New product image file.

Response:

Success (200):


```
 {
  "message": "Product updated successfully",
  "product": {
    "id": 1,
    "name": "Updated Product",
    "quantity": 15,
    "description": "Updated Description",
    "image": "http://127.0.0.1:8000/storage/uploads/images/product.png"
  }
}
```


5. Delete a Product

Endpoint: /products/{id}

Method: DELETE

Description: Deletes a specific product (Admin only).

Headers:

Authorization: Bearer {token}

Response:

Success (200):


```
{
  "message": "Product deleted successfully"
}
```

Order Management

1. Place a New Order

Endpoint: /orders

Method: POST

Description: Places a new order for a customer.

Headers:

Authorization: Bearer {token}

Request Body (JSON):

product_id: Integer (required) - ID of the product to order.

quantity: Integer (required) - Quantity of the product to order.

Response:

Success (201):


```
{
  "message": "Order placed successfully",
  "order": {
    "id": 1,
    "product_id": 1,
    "quantity": 2,
    "user_id": 1,
    "delivery_status": "pending"
  }
}
```


Error (400): {
  "message": "Insufficient product quantity"
}

2. Get User Orders

Endpoint: /orders

Method: GET

Description: Fetches all orders placed by the authenticated user.

Headers:

Authorization: Bearer {token}

Response:

Success (200):


```
[
  {
    "id": 1,
    "product_id": 1,
    "quantity": 2,
    "user_id": 1,
    "delivery_status": "pending"
  }
]
```


3. Get All Orders (Admin Only)

Endpoint: /admin/orders

Method: GET

Description: Retrieves all orders in the system (Admin only).

Headers:

Authorization: Bearer {token}

Response:

Success (200):


```
[
  {
    "id": 1,
    "product_id": 1,
    "quantity": 2,
    "user_id": 1,
    "delivery_status": "pending"
  }
]
```


4. Mark an Order as Placed (Admin Only)

Endpoint: /admin/orders/{id}/mark-as-placed

Method: PUT

Description: Marks an order as placed (Admin only).

Headers:

Authorization: Bearer {token}

Response:

Success (200): {

"message": "Order marked as placed"

}

5. Update Order Status (Admin Only)

Endpoint: /admin/orders/{id}/update-status

Method: PUT

Description: Updates the status of an order (Admin only).

Headers:

Authorization: Bearer {token}

Request Body (JSON):

status: String (required) - Status can be dispatched or delivered.

Response:

Success (200):

{

"message": "Order status updated"

}