

# Ecommerce

**AUTHORIZATION** Bearer Token

---

Token

---

## Product

**AUTHORIZATION** Bearer Token

---

This folder is using Bearer Token from collection **Ecommerce**

---

## PATCH update a product using put



http://localhost:4000/api/v1/admin/product/659424b135144f104e885b71

This endpoint allows administrators to update a specific product using the product's ID. The HTTP PATCH request should be sent to http://localhost:4000/api/v1/admin/product/659424b135144f104e885b71.

## Request Body

The request should include a form-data body with the following parameters:

- **category** (text): Description of the product category.
- **images** (file): Images of the product.

## Response

Upon a successful update (Status: 200), the response will include:

- **success** (boolean): Indicates if the update was successful.
- **updatedProduct** (object): Contains the updated product details, including ID, name, description, price, ratings, category, stock, number of reviews, images, reviews, creation date, and version.

Example response:

```
json
```

```
{
  "success": true,
  "updatedProduct": {
    "_id": "",
    "name": ""
```

```
name": "",
"description": "",
"price": 0,
"ratings": 0,
"category": "",
"Stock": 0,
"numOfReviews": 0
```

## AUTHORIZATION Bearer Token

This request is using Bearer Token from collection **Ecommerce**

## Body formdata

**category** electronics

**images**

## GET get all product



http://localhost:4000/api/v1/products

# Get Products

This endpoint retrieves a list of products.

## Request

### HTTP Request

```
GET http://localhost:4000/api/v1/products
```

## Request Body

null

## Response

- **Status:** 200
- **success** (boolean): Indicates if the request was successful.
- **products** (array): An array of products, each containing the following details:
  - **numOfReviews** (number): The number of reviews for the product.
  - **\_id** (string): The unique identifier of the product.
  - **name** (string): The name of the product.
  - **description** (string): The description of the product.
  - **price** (number): The price of the product.
  - **ratings** (number): The ratings of the product.
  - **images** (array): An array of images for the product, each containing the following details:

- `_id` (string): The unique identifier of the image.
- `public_id` (string): The public identifier of the image.
- `url` (string): The URL of the image.
- `category` (string): The category of the product.
- `Stock` (number): The stock quantity of the product.
- `reviews` (array): An array of reviews for the product.
- `createdAt` (string): The date and time when the product was created.
- `productCount` (number): The total count of products.
- `resultPerPage` (number): The number of results per page.
- `filteredProductsCount` (number): The count of filtered products.

## AUTHORIZATION Bearer Token

This request is using Bearer Token from collection **Ecommerce**

## POST create a product



`http://localhost:4000/api/v1/product/new`

## Add New Product

This endpoint allows you to add a new product to the system.

### Request Body

- `images` (file): The image file of the product.
- `name` (text): The name of the product.
- `description` (text): The description of the product.
- `price` (text): The price of the product.
- `category` (text): The category of the product.
- `Stock` (text): The stock quantity of the product.

### Response

Upon successful creation, the response will have a status code of 201 and a JSON object with the following structure:

json

```
{
  "success": true,
  "product": {
    "name": "",
    "description": "",
    "price": 0,
    "ratings": 0,
    "category": "",
    "Stock": 0,
    "numOfReviews": 0,
```

```
"id": "",
```

## AUTHORIZATION Bearer Token

This request is using Bearer Token from collection **Ecommerce**

## Body formdata

## DELETE delete a product



`http://localhost:4000/api/v1/product/639ef0347e8af9f1ed1bc099`

This endpoint sends an HTTP DELETE request to the specified URL to delete a product with the given ID. The request does not require a request body.

The response to the request returns a status code of 500, along with a JSON object containing a "success" boolean field and a "message" string field.

## AUTHORIZATION Bearer Token

This request is using Bearer Token from collection **Ecommerce**

## GET get single product



`http://localhost:4000/api/v1/product/6589e322357b8aeb9834e06c`

# Get Product Details

This endpoint retrieves the details of a specific product based on the provided product ID.

## Request

### HTTP Request

```
GET http://localhost:4000/api/v1/product/6589e322357b8aeb9834e06c
```

### Request Body

There is no request body for this request.

## Response

### Success Response

- Status: 200
- Body:

json

```
{
  "success": true,
  "product": {
    "numOfReviews": 0,
    "_id": "",
    "name": "",
    "description": "",
    "price": 0,
    "ratings": 0,
    "images": [
      {}
    ]
  }
}
```

## AUTHORIZATION Bearer Token

---

This request is using Bearer Token from collection **Ecommerce**

---

## authentication

## AUTHORIZATION Bearer Token

---

This folder is using Bearer Token from collection **Ecommerce**

---

## POST Register a User



http://localhost:4000/api/v1/register

## Register User

This endpoint allows you to register a new user.

### HTTP Request

POST http://localhost:4000/api/v1/register

### Request Body

- `name` (text) - The name of the user.
- `email` (text) - The email address of the user.
- `password` (text) - The password for the user account.
- `avatar` (file) - The avatar image for the user.

### Response

- Status: 201
- `success` (boolean) - Indicates if the user registration was successful.
- `user` (object) - Information about the registered user, including name, email, avatar filename, URL, role,

creation date, ID, and version.

- `token` - The authentication token for the registered user.

## AUTHORIZATION Bearer Token

---

This request is using Bearer Token from collection **Ecommerce**

### Body formdata

---

<b>name</b>	mohan
<b>email</b>	mohan@gmail.com
<b>password</b>	mohanmohan
<b>avatar</b>	

## POST Login user



`http://localhost:4000/api/v1/login`

## AUTHORIZATION Bearer Token

---

Token

### Body raw (json)

---

```
json

{
  "email" : "test@gmail.com",
  "password" : "testtest"
}
```

## GET Logout User



`http://localhost:4000/api/v1/logout`

This endpoint sends an HTTP GET request to `http://localhost:4000/api/v1/logout` to log the user out.

The request does not contain a request body.

## Response

• Status: 200

- Status: 200

- Body:

```
json

{
  "success": true,
  "message": ""
}
```

**AUTHORIZATION** Bearer Token

---

Token

---

## POST Forgot password



http://localhost:4000/api/v1/password/forgot

**AUTHORIZATION** Bearer Token

---

This request is using Bearer Token from collection **Ecommerce**

**Body** raw (json)

---

```
json

{
  "email" : "madan@gmail.com"
}
```

---

## PUT Reset Password



http://localhost:4000/api/v1/password/reset/a2e866c380a24dcdf3123517c202a318b0a9d8ed

**AUTHORIZATION** Bearer Token

---

This request is using Bearer Token from collection **Ecommerce**

**Body** raw (json)

---

```
json
```

```
{  
  "password" : "madangopal",  
  
  "confirmPassword" : "madangopal"  
}
```

## GET get user details



http://localhost:4000/api/v1/user

This endpoint makes an HTTP GET request to retrieve user information from the API. The request does not require a request body. The response will have a status code of 200 and will include a JSON object with a "success" key indicating the success status, and a "user" object containing user details such as avatar, user ID, name, email, role, creation date, and version. The "avatar" object within the "user" object includes the filename and URL of the user's avatar image.

### AUTHORIZATION Bearer Token

This request is using Bearer Token from collection **Ecommerce**

## PUT update password



http://localhost:4000/api/v1/password/update

### AUTHORIZATION Bearer Token

This request is using Bearer Token from collection **Ecommerce**

### Body raw (json)

json

```
{  
  "email" : "madan@gmail.com",  
  "oldPassword" : "madan1234",  
  "newPassword" : "madangopal",  
  "newConfirmPassword" : "madangopal"  
}
```

## PATCH update user profile



http://localhost:4000/api/v1/user/update



This endpoint allows you to update user information using an HTTP PATCH request. The request should be sent to `http://localhost:4000/api/v1/user/update` .

## Request Body

The request should have a form-data body with the following parameters:

- `role` : (text) - Represents the updated role of the user.
- `avatar` : (file) - Represents the updated avatar image of the user.

## Response

Upon a successful update, the endpoint returns a status code of 200 along with the following response:

```
json
{
  "success": true,
  "user": {
    "avatar": {
      "filename": "",
      "url": ""
    },
    "_id": "",
    "name": "",
    "email": "",
    "role": ""
  }
}
```

The `success` field indicates whether the update was successful. The `user` object contains the updated user information, including the avatar filename and URL.

### AUTHORIZATION Bearer Token

Token <token>

### Body formdata

role admin

avatar

## admin

### AUTHORIZATION Bearer Token

This folder is using Bearer Token from collection [Ecommerce](#)



http://localhost:4000/api/v1/admin/user/65942c5675585f6b48717a2b

This API endpoint sends an HTTP GET request to retrieve details of a specific user with the provided ID. The request should be made to http://localhost:4000/api/v1/admin/user/65942c5675585f6b48717a2b.

The response to the last execution of this request returned a status code of 200, indicating a successful operation. The response body included a "success" key with a value of true, and a "user" object containing details such as avatar, ID, name, email, role, creation date, and version.

The "avatar" object within the "user" object includes "filename" and "url" properties, while other user details such as "\_id", "name", "email", "role", "createdAt", and "\_\_v" are also provided.

Please note that the request does not include a request body.

## AUTHORIZATION Bearer Token

Token

## GET get all users



http://localhost:4000/api/v1/admin/users

This endpoint makes an HTTP GET request to retrieve a list of admin users. The request does not include a request body. The response will have a status code of 200, and it will contain a JSON object with a "success" field indicating the success status, and an array of "users" where each user object includes details such as avatar, ID, name, email, role, creation date, and version.

## AUTHORIZATION Bearer Token

This request is using Bearer Token from collection **Ecommerce**

## PUT update user



http://localhost:4000/api/v1/admin/user/63a43415ffd464812ec5394f

## AUTHORIZATION Bearer Token

This request is using Bearer Token from collection **Ecommerce**

**Body** raw (json)

json

```
{
  "name" : "testing",
  "email" : "testing@testing.com",
  "password" : "testing123",
  "role" : "user",
  "createdAt" : "2023-01-01T00:00:00.000Z",
  "updatedAt" : "2023-01-01T00:00:00.000Z",
  "version" : 0
}
```

```
"email" : "testing@gmail.com",

"role" : "admin"
}
```

## DELETE delete user



http://localhost:4000/api/v1/admin/user/63a43415ffd464812ec5394f

### AUTHORIZATION Bearer Token

This request is using Bearer Token from collection **Ecommerce**

### Body raw (json)

```
json

{
  "name" : "testing",
  "email" : "testing@gmail.com",
  "role" : "admin"
}
```

## GET get all admin products



http://localhost:4000/api/v1/admin/products

This endpoint makes an HTTP GET request to retrieve a list of products for the admin.

The response will have a status code of 200, indicating a successful request. The response body will contain a JSON object with a "success" key set to true, and an array of "products" where each product object includes various attributes such as ID, name, description, price, ratings, category, stock, number of reviews, images, reviews, creation date, and version.

The "images" array within each product object contains details about the product images including filename, URL, and ID.

The "reviews" array within each product object is initially empty.

### AUTHORIZATION Bearer Token

Token	<token>
-------	---------

## AUTHORIZATION Bearer Token

---

This folder is using Bearer Token from collection **Ecommerce**

---

## PUT create or update review



http://localhost:4000/api/v1/reviews

## AUTHORIZATION Bearer Token

---

This request is using Bearer Token from collection **Ecommerce**

## Body raw (json)

---

json

```
{
  "rating" : 3,
  "comment" : "very nice"
}
```

## GET get all reviews



http://localhost:4000/api/v1/reviews?productId=63a2ef6460490becd4061e41

## AUTHORIZATION Bearer Token

---

This request is using Bearer Token from collection **Ecommerce**

## PARAMS

---

productId	63a2ef6460490becd4061e41
-----------	--------------------------

## DELETE delete a review



http://localhost:4000/api/v1/reviews?productId=63a2ef6460490becd4061e41

## AUTHORIZATION Bearer Token

---

This request is using Bearer Token from collection **Ecommerce**

## PARAMS

---

productId 63a2ef6460490becd4061e41

---

## Order

**AUTHORIZATION** Bearer Token

---

This folder is using Bearer Token from collection **Ecommerce**

---

### POST create new order



http://localhost:4000/api/v1/order/new

**AUTHORIZATION** Bearer Token

---

This request is using Bearer Token from collection **Ecommerce**

**Body** raw (json)

---

json

```
{
  "itemPrice" : 200,
  "taxPrice" : 36,
  "shippingPrice" : 100,
  "totalPrice" : 336,
  "orderItems" : [
    {
      "product" : "639ef490f18f5f2b2f298fc6",
      "name" : "product1",
      "price" : 1200,
      "image" : "sample image",
      "quantity" : 1
    }
  ],
  "shippingInfo" : {
    "address" : "010 Los Angeles",
    "city" : "LA",
    "state" : "California",
    "country" : "USA",
    "pinCode" : 13343,
    "phoneNo" : 1341341341
  },
  "paymentInfo" : {
    "id" : "sample payment",
    "status" : "succeeded"
  }
}
```

```
{
```

## GET get single order



http://localhost:4000/api/v1/order/63b485ed9b891f56a0a76882

**AUTHORIZATION** Bearer Token

This request is using Bearer Token from collection **Ecommerce**

## GET get all orders of a user



http://localhost:4000/api/v1/orders/me

**AUTHORIZATION** Bearer Token

This request is using Bearer Token from collection **Ecommerce**

## GET get all orders -- admin



http://localhost:4000/api/v1/admin/orders

**AUTHORIZATION** Bearer Token

This request is using Bearer Token from collection **Ecommerce**

## PUT update an order -- admin



http://localhost:4000/api/v1/admin/order/63b49082b1be59800769552e

**AUTHORIZATION** Bearer Token

This request is using Bearer Token from collection **Ecommerce**

**Body** raw (json)

json

```
{
```

```
  "orderStatus" : "Delivered"
```

```
}
```

---

## **DELETE** delete an order -- admin



http://localhost:4000/api/v1/admin/order/63b49082b1be59800769552e

### **AUTHORIZATION** Bearer Token

---

This request is using Bearer Token from collection **Ecommerce**