

## **Project Title**

E-Commerce Web Application

## **Objective**

To design and develop a full-stack e-commerce website where:

- Users can sign up, login, view products, add items to cart, and checkout
- Admin can add, update, and delete products
- Backend APIs communicate with frontend using JSON and JWT authentication

## **Tech Stack**

<b>Area</b>	<b>Technology</b>
Frontend	ReactJS
Backend	Node.js
Database	MongoDB
State/Requests	Axios
Authentication	JWT
Version Control	Git & GitHub

## Database Schema

### Users

Field	Type
name	String
email	String (unique)
password	Hashed
role	user/admin
cart	Array of product + qty

### Products

Field	Type
name	String
price	Number
description	String
stock	Number

## Orders

Field	Type
user	ObjectId
items	product + qty + price
total	Number
status	placed/shipped

## Frontend UI

Page	Purpose
Home	Landing screen
Signup	Create user account
Login	Authentication
User Dashboard	Browse products
Cart Page	View and checkout
Admin Dashboard	CRUD Products

## Backend & API Integration

### Key Routes

Endpoint	Function
POST /auth/signup	User signup
POST /auth/login	JWT login
GET /products	Fetch products
POST /products	Admin add product
PUT /products/:id	Update product
DELETE /products/:id	Delete product
POST /cart/add-to-cart	Add to cart
POST /orders	Checkout order

# API Testing Using Postman

## 1. Signup request success JSON

The screenshot shows the Postman application interface. On the left, the sidebar displays collections, environments, flows, and history. A collection named "E-commerce Assignment" is selected, and a specific request titled "GET new user signup" is highlighted with a red border.

In the main workspace, a POST request is defined to "http://localhost:5000/api/auth/signup". The request body is set to raw JSON:

```
1 {
2   "name": "Apoorva2",
3   "email": "apoorva2@example.com",
4   "password": "apoorva212"
5 }
```

The response status is 200 OK, with a response time of 130 ms and a response size of 556 B. The response body is displayed as JSON:

```
1 {
2   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
3   eyJpZC16IjY5MGMyNjExMzQ5YjQzYmQxM1MDExNyIsImhdCI6MTc2MjQwMzg1NywiZXhwIjoxNzYzMDA4NjU3fQ.
4   xn_1xYuLjCvxINavKzFsdWtK9TeeGHNCzkptMxh2k",
5   "user": {
6     "id": "690c2611349b43bd19c50117",
7     "name": "Apoorva2",
8     "email": "apoorva2@example.com",
9     "role": "user"
10   }
11 }
```

At the bottom, there are buttons for Runner, Start Proxy, Cookies, Vault, and Trash.

## 2. Login returning JWT

### User login

The screenshot shows the Postman interface for a "User login" request. The left sidebar lists collections like "App Authorization", "CRUD", "E-commerce", and "E-commerce Assignment". Under "E-commerce Assignment", there are three items: "GET new user signup", "GET user login", and "GET admin login". The main workspace shows a POST request to "http://localhost:5000/api/auth/login". The "Body" tab is selected, showing raw JSON input:

```
1 {  
2   "email": "apoorna2@example.com",  
3   "password": "apoorna212"  
4 }
```

The response status is "200 OK" with a response time of 140 ms and a size of 556 B. The response body is displayed in JSON format:

```
1 {  
2   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
3   eyJpZC16IjY3MGMyNjExMzQ5YjQ2YmQxOwMlMDExNylsImhcdI6MTc2MjQwNDayMywizXhwIjoxNzYMDA40D1zf0.  
4   XBCMTwtwH6SX-pyZKhrTBa_Nkpe_XWmy9YLipvS1AI",  
5   "user": {  
6     "id": "698c2611349b43bd19c50117",  
7     "name": "Aporna2",  
8     "email": "apoorna2@example.com",  
9     "role": "user"  
10  }  
11 }
```

### Admin Login

The screenshot shows the Postman interface for an "Admin login" request. The left sidebar lists collections like "App Authorization", "CRUD", "E-commerce", and "E-commerce Assignment". Under "E-commerce Assignment", there are three items: "GET new user signup", "GET user login", and "GET admin login". The main workspace shows a POST request to "http://localhost:5000/api/auth/login". The "Body" tab is selected, showing raw JSON input:

```
1 {  
2   "email": "admin@example.com",  
3   "password": "admin123"  
4 }
```

The response status is "200 OK" with a response time of 142 ms and a size of 551 B. The response body is displayed in JSON format:

```
1 {  
2   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
3   eyJpZC16IjY3MGItZTg4YjRNDA5M2180GE5MjM0ZStiImhcdI6MTc2MjQwNDay50SwizXhwIjoxNzYMDA40TK5fQ.  
4   QWRatW1LKKNM_Ik3i63tB09v9Q0BmD2kobSAN1muX4",  
5   "user": {  
6     "id": "690b0e8b8dad099b48a9234e",  
7     "name": "Admin",  
8     "email": "admin@example.com",  
9     "role": "admin"  
10  }  
11 }
```

### 3. CRUD Operations on Products (Admin)

#### Create a new product

The screenshot shows the Postman interface with a collection named "E-commerce Assignment". A POST request is made to `http://localhost:5000/api/products`. The request body contains the following JSON:

```
1 {
2     "name": "Bedsheets",
3     "price": 400,
4     "description": "Comfy sheets for your beds.",
5     "stock": 5
6 }
```

The response status is **201 Created** with a response time of 12 ms and a size of 531 B. The response body is:

```
1 {
2     "message": "Product added successfully",
3     "product": {
4         "name": "Bedsheets",
5         "price": 400,
6         "description": "Comfy sheets for your beds.",
7         "stock": 5,
8         "_id": "690c2870349b43bd19c5011b",
9         "createdAt": "2025-11-06T04:47:44.495Z",
10        "updatedAt": "2025-11-06T04:47:44.495Z",
11        "__v": 0
12    }
13 }
```

#### Update an existing product

The screenshot shows the Postman interface with a collection named "E-commerce Assignment". A PUT request is made to `http://localhost:5000/api/products/690c2870349b43bd19c5011b`. The request body contains the following JSON:

```
1 {
2     "price": 120
3 }
```

The response status is **200 OK** with a response time of 22 ms and a size of 528 B. The response body is:

```
1 {
2     "message": "Product updated successfully",
3     "product": {
4         "name": "Bedsheets",
5         "price": 120,
6         "description": "Comfy sheets for your beds.",
7         "stock": 5,
8         "_id": "690c2870349b43bd19c5011b",
9         "createdAt": "2025-11-06T04:47:44.495Z",
10        "updatedAt": "2025-11-06T04:57:16.618Z",
11        "__v": 0
12    }
13 }
```

## Fetch products

The screenshot shows the Postman interface with a collection named "E-commerce Assignment". A GET request is made to `http://localhost:5000/api/products`. The response status is 200 OK, and the response body is a JSON array of products:

```
[{"_id": "69cc2876349b43bd19c5011b", "name": "Bedsheets", "price": 120, "description": "Comfy sheets for your beds.", "stock": 5, "createdAt": "2025-11-06T04:47:44.495Z", "updatedAt": "2025-11-06T04:57:16.618Z", "__v": 0}, {"_id": "690b4fff2c105cbe3297dd47", "name": "Headphones", "price": 200, "description": "High-quality headphones for music and calls.", "stock": 10, "createdAt": "2025-11-06T04:47:44.495Z", "updatedAt": "2025-11-06T09:31:26.477Z", "__v": 0}]
```

## 4. Add to cart (User)

The screenshot shows the Postman interface with a collection named "E-commerce Assignment". A POST request is made to `http://localhost:5000/api/cart/add-to-cart` with the following raw JSON body:

```
{"productId": "69cc2876349b43bd19c5011b"}
```

The response status is 200 OK, and the response body is a JSON object:

```
{"msg": "Added to cart", "product": {"_id": "69cc2876349b43bd19c5011b", "name": "Bedsheets", "price": 120, "description": "Comfy sheets for your beds.", "stock": 4, "createdAt": "2025-11-06T04:47:44.495Z", "updatedAt": "2025-11-06T09:31:26.477Z", "__v": 0}}
```



Github Repo: [E-Commerce Website](#)