E-Commerce Application – Product Requirement & Execution Plan

1. Purpose of This Document

This document defines the **Minimum Viable Product (MVP)** scope for the E-Commerce application. It is written to:

- Clearly communicate system behavior and expectations
- Guide implementation (human or coding agent)
- Allow recruiters to understand architecture decisions and feature prioritization

This MVP intentionally avoids non-essential features to ensure fast delivery and high signal-to-noise ratio.

---

2. User Roles & Access Control

2.1 Roles Overview

The system supports the following roles:

Role	Description
user	Regular customer using the platform
admin	Full access to manage products and orders
visitor	Read-only demo admin account for recruiters

2.2 Visitor Role (Recruiter Demo Access)

Objective:
Allow recruiters to access the Admin Dashboard without registration or setup.

Behavior:

- A predefined `visitor` account exists in the database
- Visitor credentials (email + password) are displayed publicly on the website
- Visitor has **admin-level read/write access**, except for destructive operations (optional)

Constraints:

- Visitor account password is **hardcoded and seeded**
- Visitor account cannot be deleted
- Visitor actions may be logged (optional)

UI Requirement:

- On the Login page, show: > Demo Admin Access
  > Email: admin > Password: pass@123

---

3. Database Schema (Given & Accepted)

3.1 User Model

```
model user {
  id         Int      @id @unique @default(autoincrement())
  user_name  String
  email      String   @unique
  password   String
  role       Roles    @default(user)
  created_at DateTime @default(now())
}
```

3.2 Product Model

```
model Product {
  id            Int        @id @unique @default(autoincrement())
  createdAt     DateTime   @default(now())
  updatedAt     DateTime   @updatedAt
  name          String
  description   String?
  price         Int
  discountPrice Int?
  rating        Float      @default(5.0)
  images        String[]
  categories    Category[] @relation("ProductCategories")
}
```

3.3 Category Model

```
model Category {
  id       Int       @id @default(autoincrement())
  name     String    @unique
  products Product[] @relation("ProductCategories")
}
```

---

4. Feature Breakdown

4.1 Authentication & Authorization

4.1.1 User Registration Description:

Allow users to create an account.

Requirements:

- Email must be unique
- Password must be hashed
- Default role = user

4.1.2 Login Description:
Authenticate users and issue JWT tokens.

Requirements:

- Email + password authentication
- Access token + refresh token
- Role embedded in token

4.1.3 Role-Based Access Control Description:
Restrict endpoints based on role.

Rules:

- **user**: public + personal data
- **admin**: full access
- **visitor**: admin access (demo)

4.2 Product Management

4.2.1 Product Listing (Public) Description:
Display all available products.

Requirements:

- Paginated response
- Include images, price, rating
- Exclude inactive products (future-ready)

4.2.2 Product Details Description:
View a single product.

Requirements:

- Full product information
- Categories included

4.2.3 Product CRUD (Admin / Visitor) Create Product

- Name, price, images required
- Categories optional

Update Product

- Partial updates allowed

Delete Product

- Hard delete (MVP)
- Soft delete can be added later

4.3 Category Management

4.3.1 Category Creation (Admin) Description:
Create and assign categories to products.

Requirements:

- Category name must be unique
- Many-to-many relation supported

4.3.2 Category Listing (Public) Description:
Display categories for filtering (future use).

4.4 Cart System

4.4.1 Cart Creation Description:
Each user has one persistent cart.

Behavior:

- Auto-created on first add-to-cart
- Stored in database

4.4.2 Add Item to Cart Requirements:

- Product must exist
- Quantity 1

4.4.3 Update Cart Item

- Change quantity
- Remove if quantity = 0

4.4.4 View Cart

- List all items
- Calculate subtotal

4.5 Order Management

4.5.1 Place Order Description:
Convert cart into an order.

Behavior:

- Snapshot product price at order time

- Empty cart after order creation
- Order status = PENDING

4.5.2 User Orders Users can view their order history.

4.5.3 Admin Order View Admin / Visitor can view all orders.

- Sorted by newest first

---

5. Non-Goals (Explicitly Out of Scope)

The following features are intentionally excluded from MVP but intent to implement those later:

- Payments (Stripe, SSLCommerz, etc.)
- Reviews & comments
- Wishlist
- Coupons & discounts engine
- Email notifications
- AI / recommendation engine
- Inventory reservation logic

---

6. Security & Engineering Standards

Mandatory:

- Zod validation on all inputs
- Centralized error handling
- Prisma transactions for order creation
- Password hashing (bcrypt or argon2)
- Environment-based configuration

---

7. Deployment Expectations

- Dockerized frontend & backend
- PostgreSQL database
- Seed script for:
  - Admin account
  - Visitor demo account
  - Sample products & categories

---

8. Recruiter Notes

This project prioritizes:

- Clean architecture
- Clear scope control
- Real-world backend patterns
- Production-ready decisions

The MVP reflects how real teams ship software, not tutorial-style overengineering.
