Full Stack Development with MERN

Project Documentation format

1. Introduction

- **Project Title:** SHOPEZ - E-Commerce Application
- **Team Members:** (Team id: NM2024TMID00164)

Belshia N	(413021104701)	- Team Leader
Divya R	(413021104007)	
Naresh Kumar A	(413021104021)	
Jason Asfan A	(413021104011)	

2. Project Overview

Purpose:
Shopez is a dynamic and scalable e-commerce platform designed to simplify online shopping for customers while providing robust administrative control for product management. The application aims to emulate a real-world e-commerce system, delivering a seamless user experience for both shoppers and administrators. One of the primary objectives of this project is to gain hands-on experience with MongoDB, exploring its potential as a database solution in modern web applications.

Features:

- **Admin Login and Management:**
  The admin interface enables authorized users to log in and manage the product catalog efficiently. Admins can add, update, or delete products, ensuring that the inventory remains up-to-date and relevant. This feature provides the flexibility to introduce new products, remove discontinued items, and modify existing entries.
- **Customer Login and Shopping Experience:**
  Customers can create an account or log in to access personalized features. The customer interface allows users to:
  - Browse the product catalog with detailed descriptions and images.
  - Add desired items to their shopping cart.
  - Proceed to checkout with secure payment options.
  - View order history and track their purchases in real time.
- **Responsive Design:**
  Shopez is built to be fully responsive, ensuring an optimal user experience across a range of devices, including desktops, tablets, and smartphones**.**
- **Shopping Cart and Checkout:**
  Customers can manage their shopping cart, adjusting quantities or removing items

before proceeding to checkout. The checkout process is streamlined to minimize friction and enhance user satisfaction.

- **Order Management:**
  The system tracks customer orders, allowing users to view and manage their purchase history. This includes order details, statuses, and estimated delivery timelines**.**
- **Secure Authentication and Authorization:**
  Shopez implements robust authentication mechanisms using JSON Web Tokens (JWT). This ensures that only authorized users can access the respective admin and customer functionalities.
- Database Integration with MongoDB:
  At the core of the application is MongoDB, a NoSQL database that provides flexible schema design to accommodate the dynamic nature of e-commerce data. The database stores critical information, including user details, product inventory, and transaction records. This integration ensures efficient data retrieval and scalability as the application grows.

Key Use Cases:

1. **Admin Use Case:**
   - Log in using admin credentials.
   - Add new products with details such as name, price, description, and images.
   - Update product information or remove items no longer available.
   - Monitor the inventory to ensure product availability.
2. **Customer Use Case:**
   - Register and log in to access personalized shopping features.
   - Browse the product catalog, add items to the cart, and proceed to checkout.
   - Review and manage order history, ensuring transparency and convenience.

Goals of the Project:

- To create a user-friendly e-commerce solution for both administrators and customers.
- To explore and utilize MongoDB as the backend database, focusing on efficient storage and retrieval of structured and unstructured data.
- To simulate real-world scenarios, preparing developers for challenges in full-stack application development.

With Shopez, the aim is to provide a feature-rich platform that not only meets user expectations but also serves as a learning milestone in understanding and applying MongoDB in modern web applications.

3. Architecture

The architecture of Shopez is designed to ensure scalability, maintainability, and seamless integration of the frontend, backend, and database. The application follows a modern full-stack development approach, leveraging the power of React.js, Node.js, Express.js, and MongoDB. Here’s a detailed breakdown of the architecture:

Frontend Architecture

The frontend is developed using React.js, a JavaScript library known for its efficiency in building interactive user interfaces.

- Component-Based Design: The UI is divided into reusable components such as Header, Product List, Product Detail, Cart, and Checkout. This approach promotes code reusability and easier debugging.
- State Management: React’s useState and useContext hooks manage state across components, ensuring a dynamic and responsive user experience.
- Routing: React Router is used to enable seamless navigation across different pages like Home, Product Details, Cart, Checkout, and Admin Dashboard.
- Responsiveness: The app is designed to be fully responsive, ensuring usability on various devices, including desktops, tablets, and smartphones.

Key functionalities include:

- Displaying product catalogs.
- Managing cart operations like adding, updating, and removing items.
- Handling user authentication and routing based on roles (admin or customer).

Backend Architecture

The backend is built using Node.js and Express.js, providing a robust environment to handle API requests and business logic.

- RESTful APIs: The backend exposes a set of RESTful APIs for CRUD operations on products, user management, order processing, and authentication.
- Middleware: Express.js middleware is used for handling tasks like request parsing, error handling, and user authentication.
- Role-Based Access Control: Middleware differentiates between admin and customer roles, ensuring that sensitive actions like adding or deleting products are restricted to admins.
- Security: The backend uses JWT (JSON Web Tokens) to manage secure authentication and session handling.

Key API endpoints include:

- Authentication APIs: /login, /register
- Product Management APIs: /products, /products/:id
- Order Management APIs: /orders, /orders/:id

Database Architecture

The database is powered by MongoDB, a NoSQL database known for its flexibility and scalability.

- Database Schema: MongoDB uses a flexible schema, allowing dynamic storage of data. The major collections include:
    - Users: Stores customer and admin details with fields for roles, authentication tokens, and profile information.
    - Products: Contains details like product name, description, price, category, stock availability, and images.
    - Orders: Maintains order details, including customer information, product IDs, quantities, order status, and timestamps.
- Data Relationships: Although MongoDB is schema-less, relationships between data are handled effectively using embedded documents and references. For example, order documents reference product and user collections.

Server Configuration and Connections

- Server Deployment: The backend server is hosted on Node.js, with Express.js managing routing and middleware functionalities.
- Database Connection: MongoDB is connected to the backend server via the MongoDB Node.js driver. Connection pooling is enabled to handle multiple concurrent requests efficiently.
- Environment Variables: Sensitive data like API keys, MongoDB connection URIs, and JWT secret keys are stored in environment variables for enhanced security.

Key Benefits of the Architecture

- Scalability: The use of MongoDB and Node.js ensures the app can handle growing data and user demands efficiently.
- Flexibility: MongoDB's schema-less nature allows quick modifications and additions to the data model.
- Efficiency: The asynchronous, non-blocking architecture of Node.js enables the app to handle multiple concurrent requests smoothly.
- Modularity: The separation of concerns between the frontend, backend, and database ensures that updates or modifications to one layer do not disrupt the others.

4. Setup Instructions

This section provides step-by-step instructions to set up the Shopez e-commerce application locally. Please ensure you have all the necessary prerequisites installed before proceeding with the setup.

Prerequisites

Before you begin, ensure you have the following software installed on your machine:

- Node.js (version 14.x or higher): For running the backend and frontend servers.
  - You can check if Node.js is installed by running node -v in the terminal.
  - If not installed, download from Node.js official website.
- MongoDB (version 4.x or higher): For storing and managing the application's data.

  You can either run MongoDB locally or use a cloud service like MongoDB Atlas for remote database management.

  - To check if MongoDB is installed, run mongod --version in the terminal.
- Git (optional, but recommended): For cloning the repository.
  - Verify if Git is installed by running git --version.
  - Install Git from Git official website.

Installation Steps

Follow these steps to set up the project locally:

1. **Clone the Repository**
   - Start by cloning the Shopez repository to your local machine using Git

   ```
   git clone https://github.com/your-username/shopez.git
   ```

2. **Navigate to the Project Directory**
   - Once cloned, navigate into the project directory:

   ```
   cd shopez
   ```

3. **Install Backend Dependencies**
   - Navigate to the server directory (backend folder) and install the required dependencies using npm or yarn:

   ```
   cd server

   npm install

   # or

   yarn install
   ```

   - This will install dependencies like Express, Mongoose, JWT, etc., required for the backend.
4. **Set Up Environment Variables**

- Create a .env file in the server directory to store sensitive information such as MongoDB connection URI and JWT secret key. Example .env file:

  ```
  env

  MONGO_URI=mongodb://localhost:27017/shopez_db

  JWT_SECRET=your_jwt_secret_key

  PORT=5000
  ```

5. **Install Frontend Dependencies**
   - Navigate to the client directory (frontend folder) and install the frontend dependencies:

     ```
     Copy code

     cd ../client

     npm install

     # or

     yarn install
     ```

6. **Start the Application**
   - Now you are ready to run the frontend and backend locally.
   - First, start the backend server:

```
cd server

npm start

# or

yarn start
```

   - This will start the backend server on http://localhost:5000.
   - In a separate terminal window, start the frontend server:

```
cd client

npm start

# or

yarn start
```

    - This will start the frontend React app on http://localhost:3000.
    - Both servers should now be running. You can visit http://localhost:3000 in your browser to access the Shopez e-commerce application.
7. **Database Configuration**
    - If you're running MongoDB locally, ensure the MongoDB service is running. If you're using MongoDB Atlas or any other cloud service, make sure your .env file contains the correct connection string for your remote database.
8. **Verify Setup**
    - To verify that everything is set up correctly, open your browser and go to:
        - http://localhost:3000: To access the Shopez frontend.
        - http://localhost:5000: You can use tools like Postman to test **backend API endpoints if required (e.g., /products, /orders, etc.).**

5. Folder Structure

Explanation of Client Folder and Files:

- **public/:** Contains static assets like the index.html file, which serves as the entry point to the application, as well as icons and manifest files.
- **src/:** All the source code for the frontend is here. It is further organized into subdirectories:
    - **components/:** Reusable components that are used throughout the app. For example, ProductCard.js displays individual products, and CartItem.js is used to show items in the shopping cart.
    - **context**/: Implements the React Context API for global state management. This is where the application’s global state (authentication, cart, and product data) is handled.
    - **pages/:** Contains components for different pages or views in the application, such as the homepage, product detail page, and checkout page.
    - **services/:** Contains API services for interacting with the backend, including functions for user authentication, product management, and order handling.
  - **utils/:** Contains utility functions used throughout the app, like formatting currency values or protecting routes for authenticated users.
  - **styles/:** Contains global CSS files or styled-components for styling the frontend.

Explanation of Server Folder and Files:

- **config/**: Contains configuration files like db.js, where MongoDB connection logic resides, and config.js for storing environment variables (e.g., JWT secret).
- **controllers/**: Functions responsible for handling incoming HTTP requests and applying business logic. Controllers handle different operations like user authentication, product management, and order processing.
- **models/**: Mongoose models that define the structure of MongoDB collections. For example, User.js defines the user schema, Product.js defines the product schema, and Order.js defines the order schema.

- **routes/**: Contains Express routes that map API endpoints to specific controller functions. For example, authRoutes.js defines routes like /login and /register.
- **middleware/**: Includes middleware functions for handling common tasks such as authentication (e.g., authMiddleware.js ensures the user is authenticated before accessing certain routes) and error handling (e.g., errorMiddleware.js catches and returns errors).
- **utils/**: Helper functions like generateToken.js, which is responsible for generating JWT tokens used in authentication.
- **server.js**: The entry point for the backend server where Express is configured and routes are connected.

6. Running the Application

- Provide commands to start the frontend and backend servers locally.
  - **Frontend:** `npm start` in the client directory.
  - **Backend:** `npm start` in the server directory.

7. API Documentation

The Shopez backend exposes several RESTful APIs that facilitate communication between the frontend and the database. These APIs are structured to support key features like authentication, product management, and order processing. Below is a comprehensive list of the available endpoints along with their descriptions, methods, parameters, and example responses.

1. Authentication APIs

POST /api/auth/register

Registers a new user (admin or customer). The request body should contain the user's details for creating an account.

Request Body:

```
{
  "name": "John Doe",
  "email": "john.doe@example.com",
  "password": "password123",
  "role": "customer" // 'admin' or 'customer'
}
```

Response:

- **200 OK**: Successfully registered.

```
{
  "message": "User registered successfully!"
}
```

- **400 Bad Request**: Invalid or missing fields.

```
{
  "message": "Missing required fields."
}
```

POST /api/auth/login

Logs in an existing user and returns a JWT token for subsequent requests.

Request Body:

```
{
  "email": "john.doe@example.com",
  "password": "password123"
}
```

Response:

- **200 OK**: Successfully logged in.

```
{
  "token": "jwt.token.string",
  "user": {
    "id": "userId",
    "name": "John Doe",
    "email": "john.doe@example.com",

    "role": "customer"

  }

}
```

- **401 Unauthorized**: Incorrect credentials.

```
{

  "message": "Invalid credentials."

}
```

8. Authentication

Authentication is a crucial aspect of the Shopez e-commerce application, as it ensures secure access to both customer and admin features. All authentication-related actions are handled through JWT (JSON Web Tokens), which provide a stateless, secure method of user authentication and authorization.

User Registration API

POST /api/auth/register

Registers a new user (either customer or admin). This API allows the creation of a new account by sending the user's details in the request body. The role parameter determines whether the user is an admin or a customer.

Request Body:{

```
          "name": "John Doe",

          "email": "john.doe@example.com",

  "password": "password123",

            "role": "customer"  // 'admin' or 'customer'

          }
```

- **name**: The name of the user.
- **email**: The user’s email address (must be unique).
- **password**: The user’s password (must be encrypted before storing).
- **role**: The role of the user. Can be either admin or customer.

Response:

- **200 OK**: Successfully registered the user.

```
{
  "message": "User registered successfully!"
}
```

- **400 Bad Request**: Invalid or missing fields, or if the email is already taken.

```
{
  "message": "Missing required fields or email already exists."
}
```

9. User Interface

The **User Interface (UI)** of the Shopez application is designed to provide a seamless and intuitive shopping experience for customers and efficient management tools for admins. The interface has been developed with modern web technologies and is fully responsive, ensuring that users can interact with the platform on a variety of devices, from desktops to smartphones. Below is an overview of the main features and design elements.

1. Landing Page (Home)

The **Home Page** is the first point of contact for users. It showcases a clean, user-friendly design with easy navigation and access to various parts of the application. The homepage is designed to highlight products, promotions, and special offers.

- **Category Filters**: Allows users to browse products by category (e.g., electronics, clothing, accessories).
- **Search Bar**: A search feature to help users quickly find products by name or category.

2. Product Listing Page

The **Product Listing Page** allows customers to browse available products, filter by category, and view detailed product information.

3. Product Detail Page

On the **Product Detail Page**, users can see more information about a specific product. This page provides a detailed view to help customers make purchasing decisions.

Key Features:

- **Product Images**: Multiple images of the product, with a zoom-in option for better details.
- **Product Information**: Includes product name, description, price, stock availability, and specifications.
- **Add to Cart**: A button that allows customers to add the product to their shopping cart.
- **Reviews and Ratings**: A section where users can view customer reviews and ratings.
- **Quantity Selection**: A dropdown or input box to specify the number of items to add to the cart.

4. Shopping Cart

The **Shopping Cart** page shows all the items that a customer has added to their cart. It allows users to review their selections and make changes before proceeding to checkout.

Key Features:

- **Item List**: Displays each item in the cart with its name, quantity, price, and total price.
- **Quantity Control**: Users can update the quantity of products or remove them.
- **Total Price**: Shows the total cost of all items in the cart.
- **Proceed to Checkout**: A button to start the checkout process.

5. Checkout Page

The **Checkout Page** is where customers enter their shipping and payment details to complete the purchase. It has a straightforward design to ensure a smooth transaction process.

Key Features:

- **Shipping Information**: Users provide their shipping address, including name, address, and phone number.
- **Payment Options**: Various payment methods are offered (e.g., credit card, PayPal).
- **Order Summary**: A breakdown of the items in the cart, along with total pricing and estimated shipping costs.
- **Place Order Button**: A final button to submit the order.

6. User Account Dashboard (for Logged-in Users)

The **User Account Dashboard** allows customers to manage their profile, track their orders, and update account settings.

7. Admin Dashboard (for Admin Users)

The **Admin Dashboard** is used for managing the Shopez platform. Admins can add, edit, and delete products, view user information, and manage orders.

Key Features:

- **Product Management**: Admins can add new products, update existing products, or delete discontinued ones.
- **Order Management**: Admins can view and update the status of orders (e.g., pending, shipped, delivered).
- **User Management**: Admins can view and manage customer accounts and roles.

10. Testing

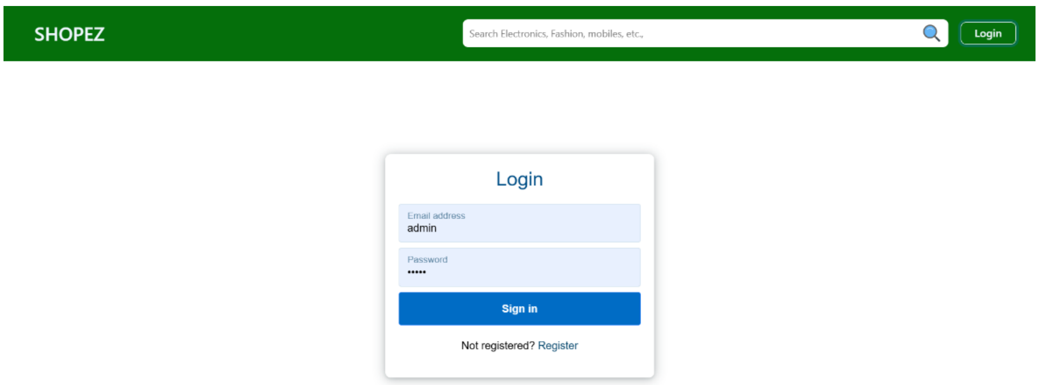
Manual testing involves testing the application’s functionality by simulating real-user actions without the use of automated testing tools. The objective is to ensure that the application behaves correctly across all features and edge cases, validating the overall performance, usability, and security of the platform.

Testing Levels:

- **Unit Testing**: Ensures that individual components function correctly (e.g., product display, cart operations).
- **Integration Testing**: Ensures that different modules of the application interact seamlessly (e.g., frontend interacting with backend APIs, database).
- **System Testing**: Verifies that the application works as a whole (e.g., end-to-end workflow from browsing products to checkout).
- **User Acceptance Testing (UAT)**: Ensures that the application meets the users' needs and expectations, focusing on the customer and admin workflows.

11. Screenshots or Demo

- **Demo video link:** https://github.com/BelshiaN/SHOPEZ.git
- **Screenshots:**



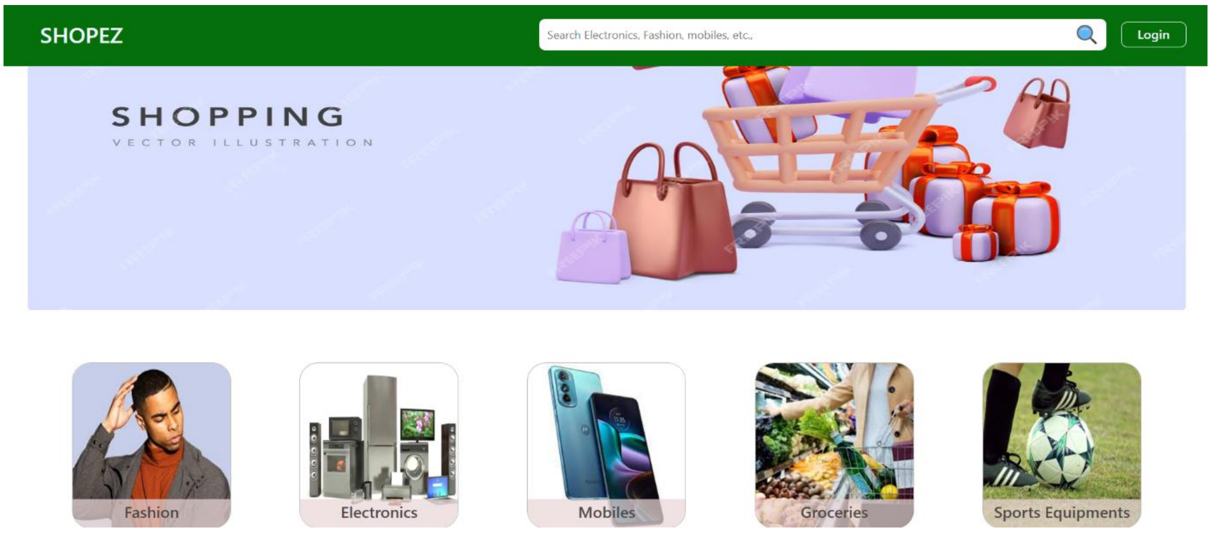
Login

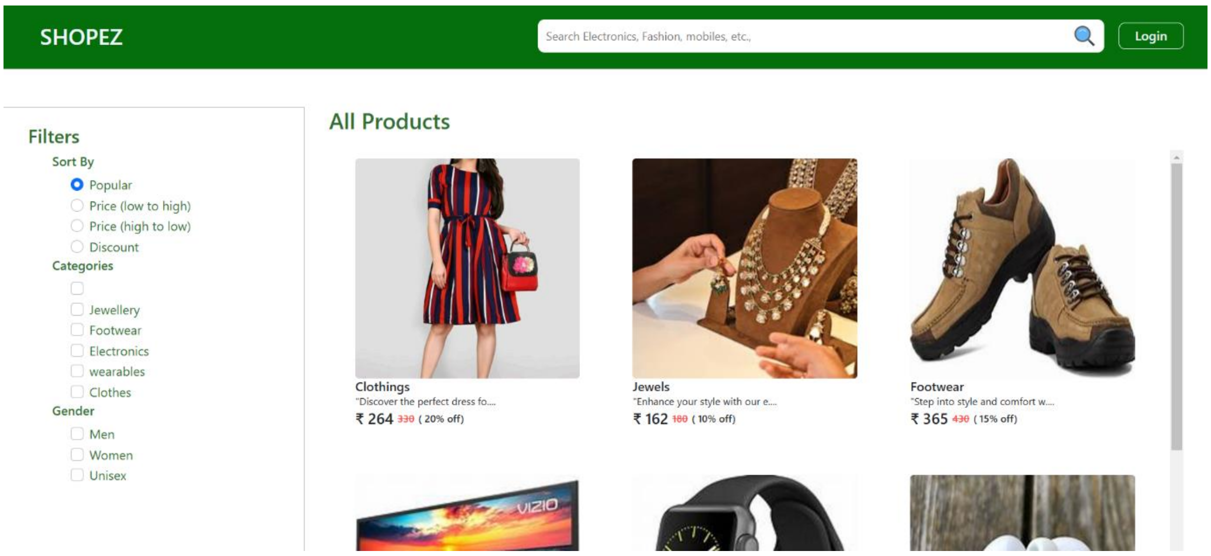
Email address
admin

Password
•••••

Sign in

Not registered? Register

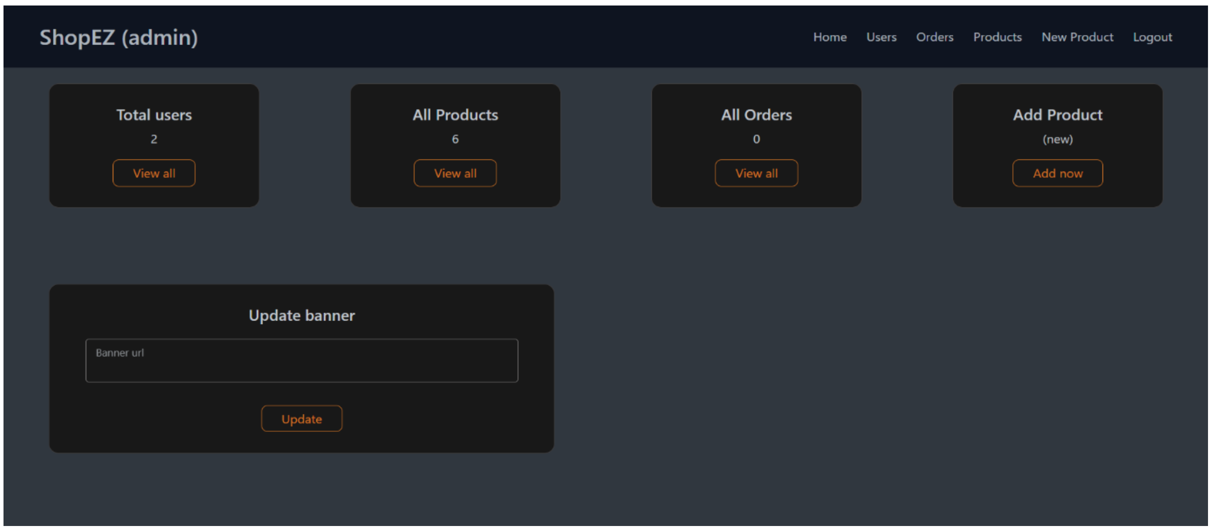




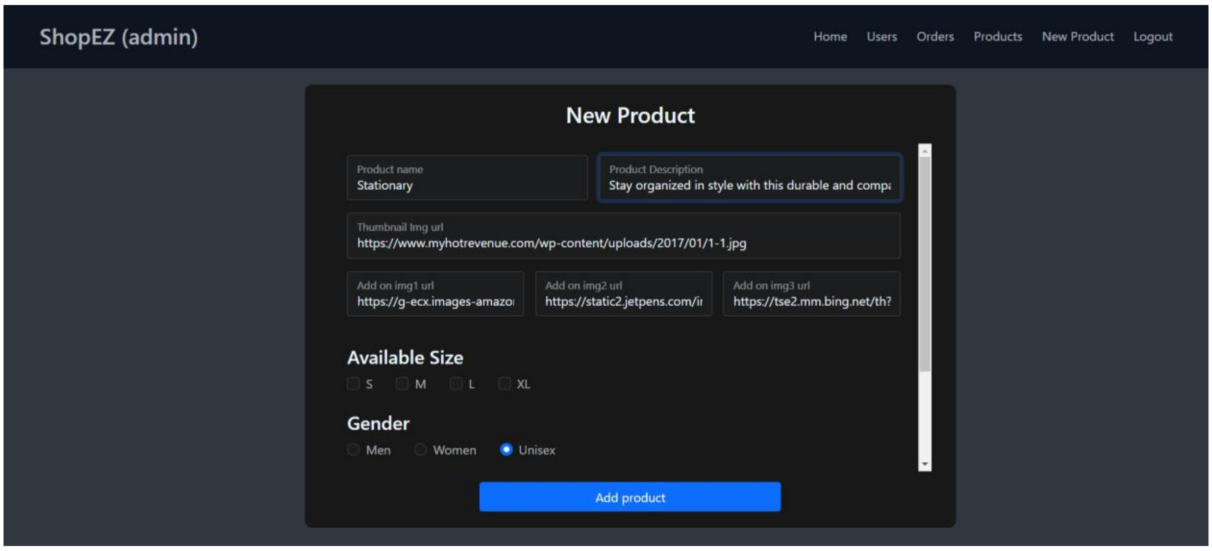
Clothings
"Discover the perfect dress fo....
₹ 264 ~~330~~ (20% off)

Jewels
"Enhance your style with our e....
₹ 162 ~~180~~ (10% off)

Footwear
"Step into style and comfort w....
₹ 365 ~~430~~ (15% off)



Total users	All Products	All Orders	Add Product
2	6	0	(new)
View all	View all	View all	Add now



New Product

Product name: Stationary

Product Description: Stay organized in style with this durable and compa

Thumbnail Img url: https://www.myhotrevenue.com/wp-content/uploads/2017/01/1-1.jpg

Add on img1 url: https://g-ecx.images-amazo

Add on img2 url: https://static2.jetpens.com/i

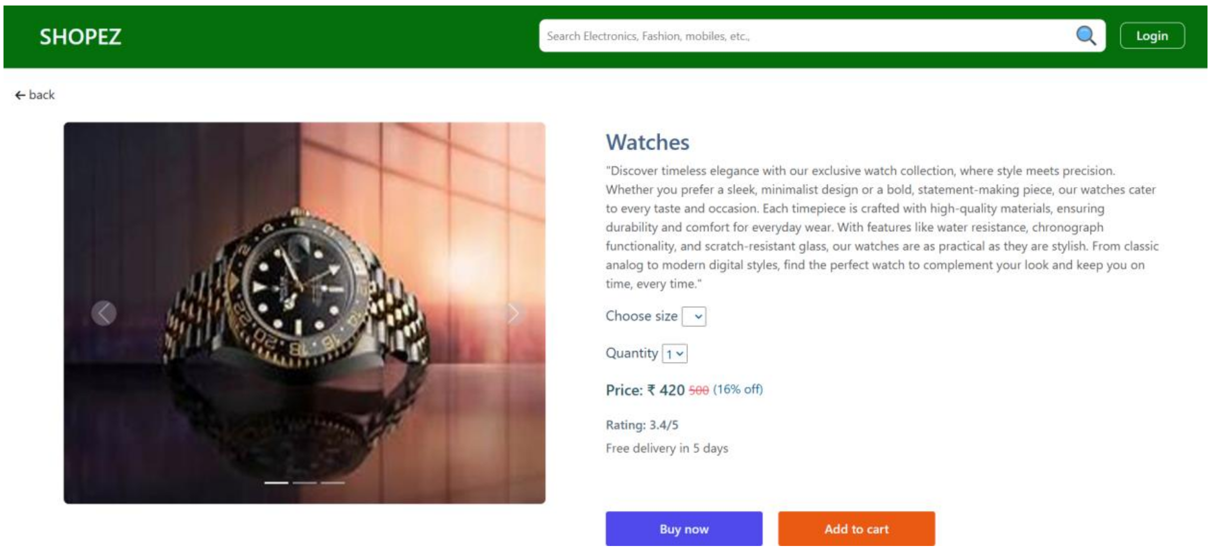
Add on img3 url: https://tse2.mm.bing.net/th?

Available Size

S M L XL

Gender

Men Women [✓ Unisex]



Watches

"Discover timeless elegance with our exclusive watch collection, where style meets precision. Whether you prefer a sleek, minimalist design or a bold, statement-making piece, our watches cater to every taste and occasion. Each timepiece is crafted with high-quality materials, ensuring durability and comfort for everyday wear. With features like water resistance, chronograph functionality, and scratch-resistant glass, our watches are as practical as they are stylish. From classic analog to modern digital styles, find the perfect watch to complement your look and keep you on time, every time."

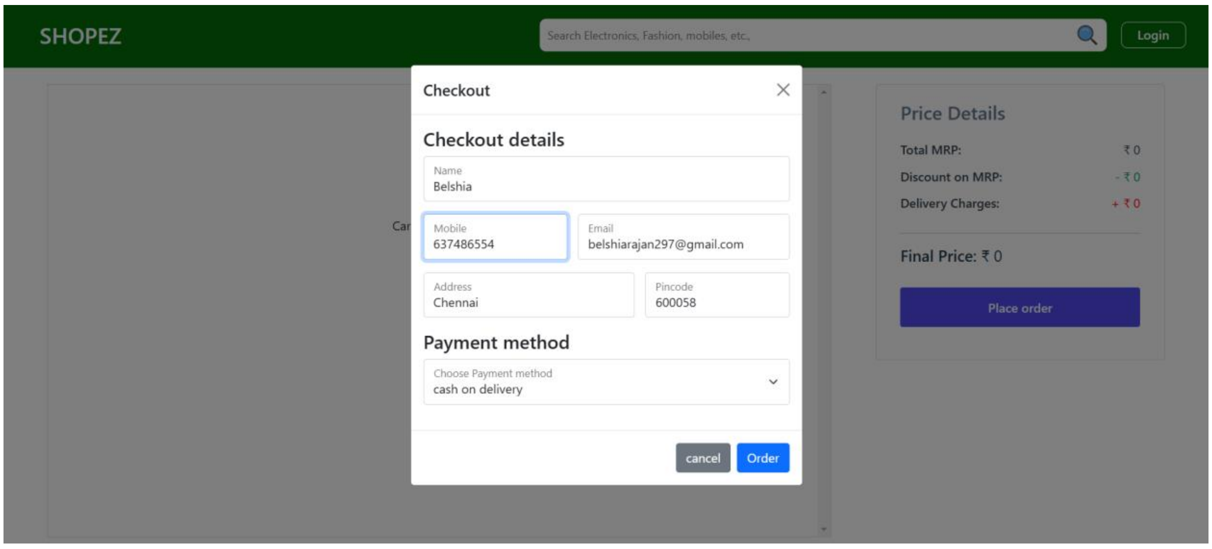
Choose size

Quantity 1

Price: ₹ 420 ~~500~~ (16% off)

Rating: 3.4/5

Free delivery in 5 days



Checkout

Checkout details

Name: Belshia

Mobile: 637486554

Email: belshiarajan297@gmail.com

Address: Chennai

Pincode: 600058

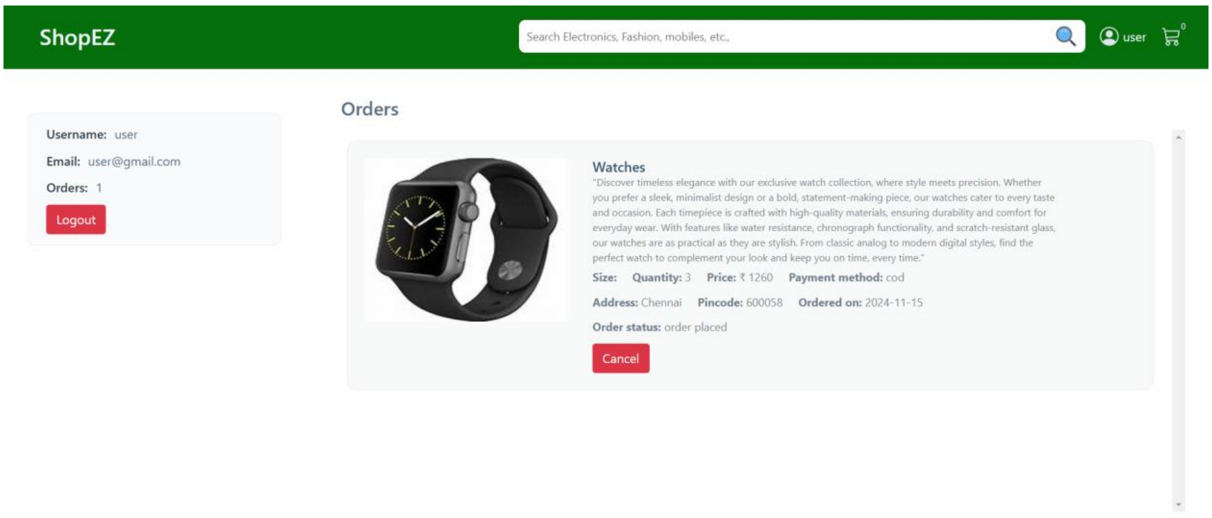
Payment method

Choose Payment method: cash on delivery

Price Details

Total MRP:	₹ 0
Discount on MRP:	- ₹ 0
Delivery Charges:	+ ₹ 0

Final Price: ₹ 0



Username: user

Email: user@gmail.com

Orders: 1

Orders

Watches

"Discover timeless elegance with our exclusive watch collection, where style meets precision. Whether you prefer a sleek, minimalist design or a bold, statement-making piece, our watches cater to every taste and occasion. Each timepiece is crafted with high-quality materials, ensuring durability and comfort for everyday wear. With features like water resistance, chronograph functionality, and scratch-resistant glass, our watches are as practical as they are stylish. From classic analog to modern digital styles, find the perfect watch to complement your look and keep you on time, every time."

Size: **Quantity:** 3 **Price:** ₹ 1260 **Payment method:** cod

Address: Chennai **Pincode:** 600058 **Ordered on:** 2024-11-15

Order status: order placed

12. Known Issues

- Admin Dashboard Performance
- Mobile Checkout Experience
- Order History Pagination
- Product Image Zoom on Mobile
- JWT Token Expiry Handling
- Browser Compatibility for Admin Panel
- Payment Gateway Integration (Test Environment)

13. Future Enhancements

In the future, Shopez aims to introduce several new features and improvements to enhance both user experience and administrative control. Key enhancements include adding multi-currency support for global customers, implementing product reviews and ratings to improve product transparency, and integrating advanced search and filter options to make browsing more efficient. Additionally, AI-driven personalized recommendations, enhanced mobile support, and a referral program for customers are on the roadmap. These features will further improve shopping convenience, increase customer engagement, and streamline administrative tasks, positioning Shopez for continued growth and user satisfaction.

Moreover, plans include integrating a subscription-based product delivery option, improving the checkout process with faster payment gateways, and adding an integrated analytics dashboard for admins to track performance and make data-driven decisions.