Name : Jenil Vaghasiya

Enroll. No. : 92310133012

Implementation and Technical Documentation

Project Title : Ignite Perfume Web Application

1. Introduction

The Ignite Perfume Web Application is a working online store made to give people an easy and dependable way to buy perfumes online. The way it's built makes sure the code is good, everything works, and all parts work together smoothly, just like the people in charge wanted. The project is set up in separate parts, so it can grow and be fixed easily.

2. Code Quality

- **Coding Standards**

  JavaScript (ES6) was used for both frontend and backend, following **JSDoc-style documentation**.

  Code is **readable, well-commented, and modular**, with a clear separation between UI, business logic, and database access.

- **Best Practices Implemented**

  Separation of Concerns:

  Frontend handles only presentation and user interactions.

  Backend manages APIs, authentication, and database communication.

Database stores structured data with indexes for fast queries.

Error Handling & Validation:

Input validation using middleware (e.g., ensuring strong passwords during signup).

Error handling with proper HTTP status codes (e.g., 400 Bad Request, 401 Unauthorized).

Version Control:

GitHub was used with a **clear commit history**. Each feature and bug fix is recorded.

3. System Functionality

The implemented system fulfills all key functional requirements:

1. **User Authentication**
   - Users can register, log in, and manage their profile securely.
2. **Product Catalog**
   - Displays all perfumes with details (name, description, price, category).
   - Search and filter options for easy navigation.
3. **Shopping Cart & Checkout**
   - Users can add/remove items from the cart.
   - Secure checkout with order confirmation stored in MongoDB.
4. **Admin Panel**
   - Admin can add, update, or delete perfumes.
   - Admin can view orders and manage inventory.
5. **Deployment**
   - The application is live and accessible at **igniteperfume.vercel.app**.

4. Integration Across Components

The system is designed with **three integrated layers**:

- **Frontend (React.js)**
  - Handles user interface, dynamic pages, and state management.
  - Connects to backend APIs via Axios/Fetch.
- **Backend (Node.js + Express.js)**
  - Provides REST APIs for authentication, products, and orders.
  - Middleware ensures security and error handling.
- **Database (MongoDB Atlas)**
  - Stores user accounts, perfume catalog, cart items, and order details.
  - Indexes applied for faster search queries.

Integration Workflow Example (Checkout Process):

1. User selects perfumes in React frontend.
2. Data is sent to backend via secure API request.
3. Backend validates and stores the order in MongoDB Atlas.
4. Response is sent back to frontend with success confirmation.

5. Code Structure and Organization

```
ignite-perfume/
├── frontend/              # React.js Frontend
│   ├── src/
│   │   ├── components/    # Reusable UI components
│   │   ├── pages/         # Page views (Home, Products, Cart, Admin)
│   │   ├── services/      # API calls (Axios)
│   │   └── App.js
│
├── backend/               # Node.js Backend
```

```
│   ├── routes/             # API route definitions
│   ├── controllers/        # Business logic
│   ├── models/             # Mongoose schemas
│   ├── middlewares/        # Validation & error handling
│   └── server.js
│
├── database/
│   └── mongoConnection.js   # MongoDB Atlas connection
│
├── package.json            # Dependencies
├── README.md               # Setup & instructions
```

6. Implementation Details

- **Frontend (React.js)**
    1. React Router for navigation.
    2. Context API for state management.
    3. Responsive design using CSS/Tailwind.
- **Backend (Node.js + Express)**
    1. REST APIs with authentication using JWT.
    2. Secure password hashing using bcrypt.
    3. Error-handling middleware for consistent API responses.
- **Database (MongoDB Atlas)**
    1. Collections: users, products, orders.
    2. Schema validation with Mongoose.

8. Instructions for Running the System

1. **Clone the repository**

   git clone https://github.com/your-repo/ignite-perfume.git
   cd ignite-perfume

2. **Setup Backend**

   cd backend
   npm install
   npm start

3. **Setup Frontend**

   cd frontend
   npm install
   npm start

4. **Environment Variables**
   - Create .env file with:
   - MONGO_URI=your-mongodb-uri
   - JWT_SECRET=your-secret
5. **Access the Application**
   - Localhost: http://localhost:3000
   - Live: igniteperfume.vercel.app