Clothing Website Documentation

(Rakeshkumar Shah)

1. Project Overview

This project is an e-commerce platform for clothing, offering a seamless shopping experience. It features four main sections: Men's Wear, Kids' Wear, School Uniforms, and Under Garments. Built with Next.js for the frontend and Firebase for backend services, the platform ensures a modern and efficient shopping experience.

2. Technology Stack

- **Frontend:** Next.js
- **Backend:** Firebase (Authentication & Firestore)
- **Styling:** Tailwind CSS
- **API Testing:** Postman
- **Development Environment:** VS Code

3. Project's Basic Folder Structure

/Ecommerce-project

|-- frontend

  |-- components

  |-- pages

  |-- styles

|-- backend (if any backend logic exists)

|-- public

|-- .env.local (Environment Variables)

|-- package.json

4. Core Features

4.1 User Authentication

- Signup/Login using Firebase Authentication.
- Logout functionality.

- Forgot Password Option (via Email / SMS).
- Password reset option.

4.2 Product Listings

- Categories: **Men's Wear, Kids' Wear, School Uniforms, Under Garments**.
- Pagination and lazy loading for performance optimization.

4.3 Product Details

- Individual product pages with images, descriptions, Category, Quantity, Size, and pricing.
- Size Options
- Add to Cart button.
- Favourite button

4.4 Cart System

- Add, update, and remove items.
- Increase / Decrease Quantity of added items. (Update Quantity)
- Total price calculation of whole cart's products. (Refer Flipkart)

4.5 Checkout Process (Not Decided Yet)

- User address and payment details collection.
- Order confirmation page.

4.6 Order Management

- Order tracking, history, and status updates.

4.7 User Profile

- Account details and saved addresses.
- Show "Orders" button.
- Show "Favourite" button.
- Show "Add to cart" Button.

4.8 Search & Filters

- Filter products by category, price, and popularity.
- User can filter products with multiple tags.

4.9 Admin Panel (if included)

- Manage product inventory.
- Track customer orders.

5. API Documentation

5.1 Authentication API

- **POST /api/auth/signup** – Register a user.
- **POST /api/auth/login** – Authenticate a user.

5.2 Products API

- **GET /api/products** – Fetch all products.
- **GET /api/products/****:id** – Fetch a single product.

5.3 Cart API

- **POST /api/cart** – Add item to cart.
- **GET /api/cart** – Retrieve cart items.

5.4 Order API

- **POST /api/order** – Place an order.
- **GET /api/orders** – Retrieve order history.

5.4 Cart API

- **POST /api/** favourite – Add item to favourite.
- **GET /api/**favourite – Retrieve favourite items.

6. Database Structure (Firebase Firestore)

Collections:

- **users:** Stores user details and order history.
- **products:** Stores product data (name, category, price, images, etc.).
- **orders:** Stores placed orders and their statuses.

- **cart:** Stores user cart items.

7. Deployment Guide

7.1 Local Setup

1. Clone the repository.
2. Install dependencies: npm install.
3. Configure Firebase credentials in .env.local.
4. Run the development server: npm run dev.

7.2 Hosting

- **Frontend:** ….
- **Backend:** Use Firebase for Firestore and authentication.

8. Security Measures

- Firebase authentication for user access control.
- Firestore security rules to prevent unauthorized access.
- Input validation and protection against XSS and SQL injection.

9. Performance Optimization

- Lazy loading for product images.
- Caching strategies using Next.js.
- Minified and optimized Tailwind CSS.

10. Troubleshooting & FAQs

Q1: Firebase Authentication is not working?

Solution: Check if Firebase API keys are correctly configured in .env.local.

Q2: Images are not loading properly?

Solution: Verify the correct public storage path and ensure Next.js Image Optimization is enabled.

NOTE* - This Documentation is not fully complete, may update in feature.