README for Online Store Web Application (Front-end & Back-end)

# Introduction

This project is an online store web application that comprises both front-end and back-end components. The online store allows users to browse products, add them to the cart, make payments, and track orders. The front-end is built with modern web technologies for an intuitive user interface, while the back-end provides a robust API and database system to manage products, users, and transactions. This document explains the architecture, technologies used, and instructions for setting up and running the project.

# Frontend

## Overview

The front-end of this web application is responsible for the user interface and user experience. It is built using modern technologies to ensure responsiveness, scalability, and performance. Users can browse through categories, view product details, and manage their shopping carts. Authentication and payment gateways are integrated to ensure a secure purchasing process.

## Technologies Used

1. \*\*HTML/CSS/JavaScript\*\*: For structuring and styling the web pages, and for handling dynamic interactions.  
2. \*\*React.js\*\*: A JavaScript library for building responsive and scalable user interfaces.  
3. \*\*Redux\*\*: State management for handling application-wide states like user authentication and cart management.  
4. \*\*Bootstrap\*\*: A CSS framework to create a mobile-first, responsive web design.  
5. \*\*Axios\*\*: For making HTTP requests to the back-end API.

## Key Features

1. \*\*Product Listing and Details\*\*: Users can browse products and view detailed descriptions and pricing.  
2. \*\*Shopping Cart\*\*: Users can add, update, or remove items from their shopping cart.  
3. \*\*User Authentication\*\*: Sign-up, login, and logout functionality for registered users.  
4. \*\*Checkout and Payment Gateway Integration\*\*: Secure payment through PayPal or Stripe API.  
5. \*\*Order Tracking\*\*: Users can track the status of their orders post-purchase.

# Backend

## Overview

The back-end is responsible for handling the business logic, database interactions, and API endpoints. It ensures smooth communication between the front-end and the server, managing tasks such as user authentication, product management, and order processing. The backend is built using Node.js and Express, with a MongoDB database.

## Technologies Used

1. \*\*Node.js\*\*: A JavaScript runtime used to build the server-side of the application.  
2. \*\*Express.js\*\*: A web framework for Node.js used to create APIs and handle HTTP requests.  
3. \*\*MongoDB\*\*: A NoSQL database for storing user data, product catalogs, orders, and more.  
4. \*\*Mongoose\*\*: An ODM (Object Data Modeling) library for MongoDB that simplifies database operations.  
5. \*\*JWT (JSON Web Tokens)\*\*: For secure authentication and session management.  
6. \*\*Bcrypt\*\*: Used to hash user passwords for secure storage.  
7. \*\*Stripe/PayPal API\*\*: Integrated payment gateways for processing transactions.

## Key Features

1. \*\*RESTful API\*\*: Exposes endpoints for managing products, users, orders, and payments.  
2. \*\*Authentication & Authorization\*\*: JWT-based authentication system for secure user access.  
3. \*\*Product Management\*\*: Admin can create, update, delete, and view products through secure endpoints.  
4. \*\*Order Management\*\*: Create and manage customer orders, and process payments through Stripe or PayPal.  
5. \*\*Database Management\*\*: MongoDB is used to store all data related to users, products, and orders.

# Project Setup

## Frontend Setup

1. Clone the repository using:  
 ```bash  
 git clone <repository-url>  
 ```  
2. Navigate to the frontend directory:  
 ```bash  
 cd frontend  
 ```  
3. Install dependencies:  
 ```bash  
 npm install  
 ```  
4. Start the development server:  
 ```bash  
 npm start  
 ```  
 The frontend should now be running on `http://localhost:3000`.

## Backend Setup

1. Navigate to the backend directory:  
 ```bash  
 cd backend  
 ```  
2. Install dependencies:  
 ```bash  
 npm install  
 ```  
3. Set up environment variables:  
 - Create a `.env` file in the backend directory.  
 - Add the following variables:  
 ```bash  
 PORT=5000  
 MONGO\_URI=<your-mongo-uri>  
 JWT\_SECRET=<your-jwt-secret>  
 STRIPE\_KEY=<your-stripe-key>  
 ```  
4. Start the backend server:  
 ```bash  
 npm start  
 ```  
 The backend API should now be running on `http://localhost:5000`.

# API Documentation

The backend exposes several RESTful API endpoints for interaction with the frontend. Some of the key endpoints include:  
1. \*\*POST /api/auth/login\*\*: Login user.  
2. \*\*POST /api/auth/register\*\*: Register new user.  
3. \*\*GET /api/products\*\*: Fetch all products.  
4. \*\*POST /api/orders\*\*: Create a new order.  
5. \*\*GET /api/orders/:id\*\*: Get details of a specific order.

# License

This project is licensed under the MIT License. See the LICENSE file for details.