 ****

**CAPSTONE PROJECT REPORT**

**PROJECT TITLE**

Web and mobile development: Design and develop an e-commerce platform with personalized product recommendations

**REPORT SUBMITTED BY:**

R.Lavanya(192321022)

**COURSE CODE**: CSA1087

**COURSE NAME:** Software Engineering

**ABSTRACT:**

This project focuses on developing a web and mobile e-commerce platform with personalized product recommendations to enhance user experience. Key subtopics include:

**1. Responsive Platform Design**

* Create a mobile- and web-friendly interface for seamless usability.

**2. Personalized Product Recommendations**

* Implement machine learning to suggest products based on user behavior.

**3. Core E-commerce Features**

* Include secure payment options, a shopping cart, and order management.

This platform will enhance user engagement, improve merchant efficiency, and support scalable business operations.

**INTRODUCTION:**

E-commerce platforms play a crucial role in meeting consumer demands for convenience and personalization. This project aims to design a responsive web and mobile platform with personalized product recommendations powered by machine learning. It will analyze user behavior to provide tailored suggestions, enhancing engagement and boosting sales. The platform will include secure payment systems, real-time inventory management, and vendor tools. With scalable architecture, it ensures reliability and a seamless shopping experience for all users.

**Background:**

The rise of digital commerce has made responsive web and mobile platforms essential for businesses. Personalization, driven by machine learning, enhances user satisfaction by providing tailored product recommendations. This project leverages these advancements to create a modern, user-focused e-commerce solution.

**Objectives:**

The main objective is to design and develop a responsive e-commerce platform with personalized product recommendations to enhance user experience and drive sales.

1. To design and develop a responsive e-commerce platform optimized for both web and mobile devices.
2. To implement a personalized product recommendation system using machine learning algorithms.
3. To integrate secure payment gateways, real-time inventory management, and user-friendly order management features.
4. To ensure scalability and security of the platform, handling increasing user traffic and data protection needs.

**Methodology:**

The development process will begin with requirement analysis to define user needs and platform specifications. The design phase will focus on creating responsive UI/UX for both web and mobile platforms. Machine learning algorithms will be implemented to deliver personalized product recommendations. After development, the platform will be tested for functionality, security, and performance, followed by deployment and continuous maintenance.

**Requirements Gathering**

1. **Functional Requirements**: User registration, personalized product recommendations, secure payment integration, and real-time inventory management.
2. **Non-Functional Requirements**: Responsive design, scalability, security, and fast load times.
3. **Technical Requirements**: Integration of machine learning algorithms and external payment systems.
4. **User Experience**: Intuitive navigation and a seamless, personalized shopping experience.

**System Design**

1. **Architecture**: Client-server model with separate front-end (responsive web and mobile) and back-end (RESTful API).
2. **Database**: Relational (MySQL/PostgreSQL) for user and product data; NoSQL (MongoDB) for recommendation and activity data.
3. **Recommendation Engine**: Machine learning algorithms for personalized product suggestions based on user behavior.
4. **Security and Scalability**: Secure payment integration, HTTPS, and cloud deployment for scalability and data protection.

**Code:**

**Frontend Code (HTML/CSS)**

**index.html:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Personalized E-Commerce</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<header>

<h1>Welcome to Our E-Commerce Platform</h1>

</header>

<section id="products">

<h2>Recommended Products</h2>

<div id="product-list">

<!-- Dynamically populated -->

</div>

</section>

<script src="app.js"></script>

</body>

</html>

**Style.css:**

body {

font-family: Arial, sans-serif;

}

header {

background-color: #333;

color: white;

text-align: center;

padding: 1em;

}

#products {

padding: 20px;

}

.product-item {

border: 1px solid #ddd;

padding: 10px;

margin: 10px;

width: 200px;

display: inline-block;

}

.product-item img {

width: 100%;

}

**Backend Code (Node.js/Express)**

**app.js (Backend Server)**

const express = require('express');

const path = require('path');

const app = express();

const port = 3000;

const recommendedProducts = [

{ id: 1, name: 'Smartphone', price: 500, image: 'smartphone.jpg' },

{ id: 2, name: 'Laptop', price: 1200, image: 'laptop.jpg' },

{ id: 3, name: 'Headphones', price: 150, image: 'headphones.jpg' },

];

app.use(express.static(path.join(\_\_dirname, 'public')));

app.get('/', (req, res) => {

res.sendFile(path.join(\_\_dirname, 'public', 'index.html'));

});

app.get('/api/recommendations', (req, res) => {

res.json(recommendedProducts);

});

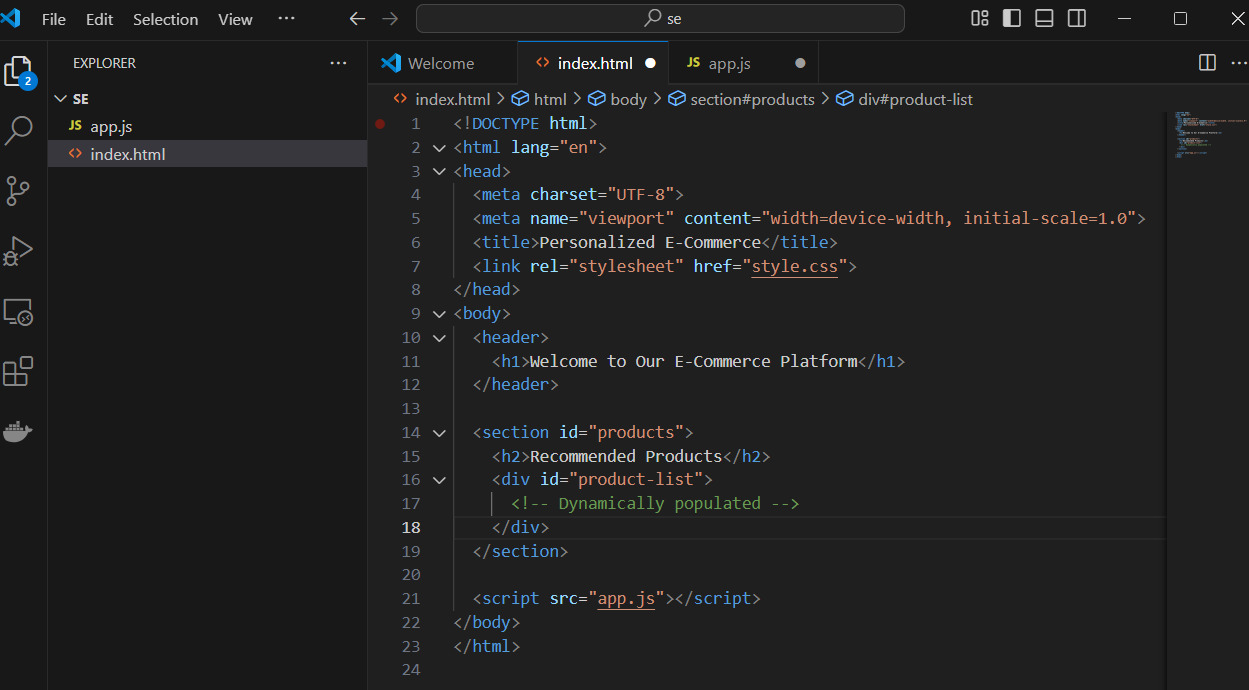
app.listen(port, () => {

console.log(`Server running at http://localhost:${port}`);

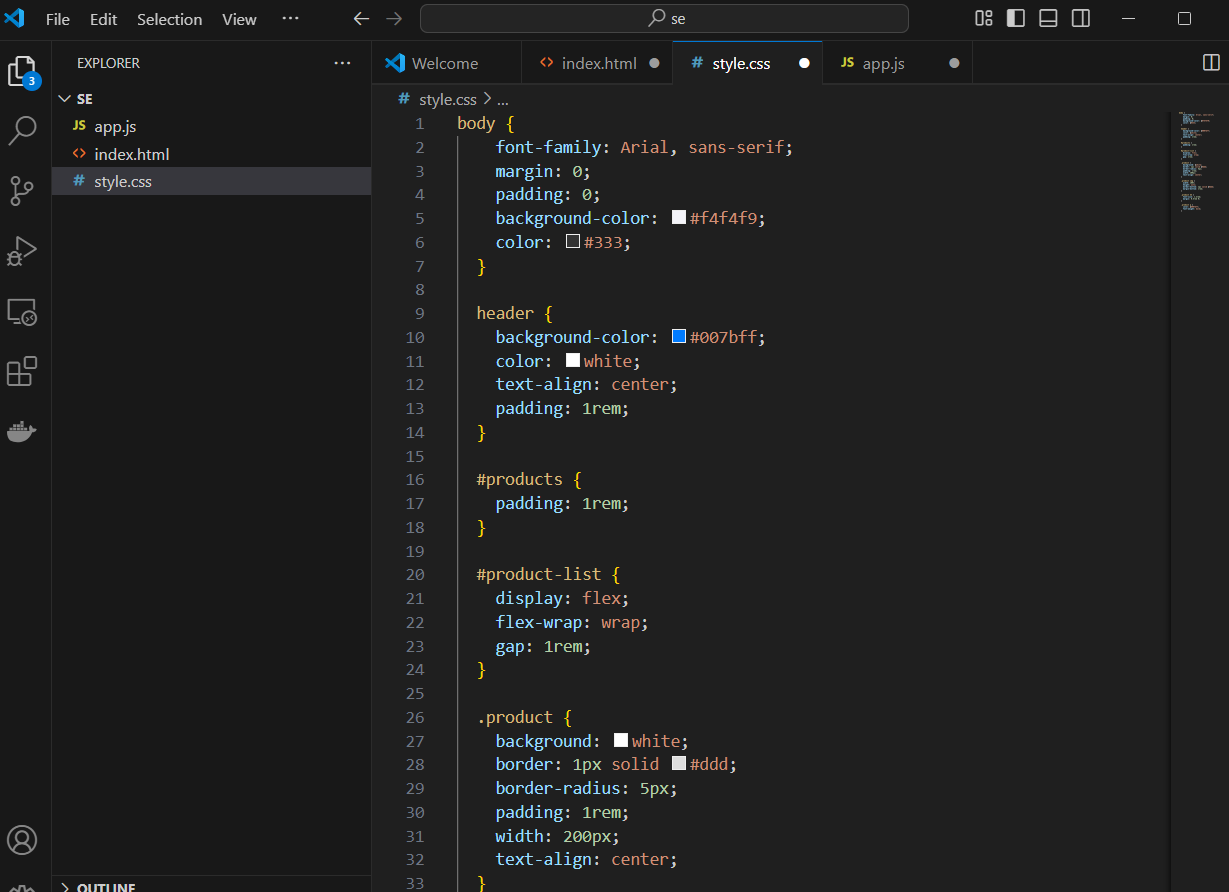
});

**Frontend Code (HTML/CSS)**

**index.html:**

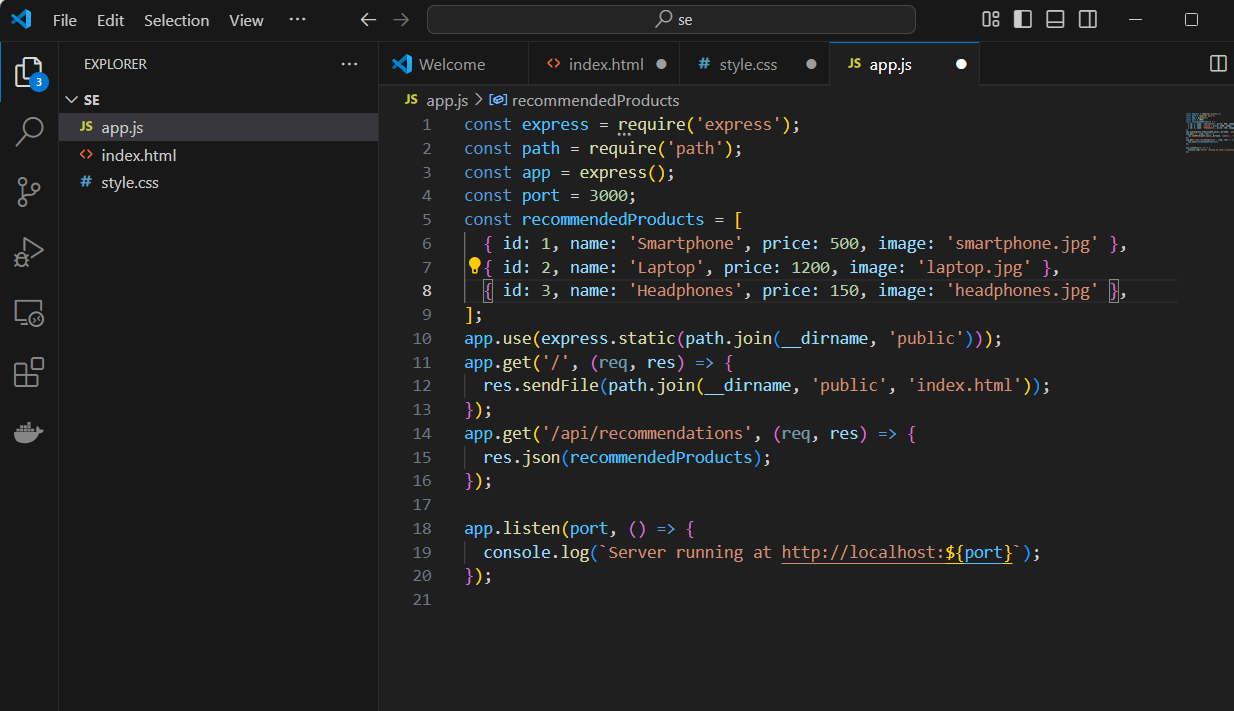
****

**Style.css**

****

**Backend Code (Node.js/Express)**

**app.js (Backend Server)**

****

**Development**

**Project Initialization:**

**Objective:** Set up project repositories, version control, and basic structure for web and mobile platforms.

* **Task:** Create a Git repository for version control using GitHub or GitLab.
* **Task:** Set up the basic folder structure:
  + /backend for the API and server code.
  + /frontend for the web app (React).
  + /mobile for the mobile app (React Native).
* **Task:** Choose tech stack (e.g., **Node.js** for the backend, **React** for the frontend, **React Native** for mobile).

.

**Backend Development (API & Database)**

**Objective:** Build the server-side infrastructure to manage products, user accounts, and handle recommendations.

**Backend (Node.js/Express):**

* **Set Up Express Server**:
  + Initialize Node.js project with npm init.
  + Install necessary dependencies like **express**, **mongoose** (for MongoDB), and **jsonwebtoken** (for authentication).
* **API Endpoints**:
  + **GET /api/products**: Fetch a list of products for display.
  + **POST /api/users/login**: User login API.
  + **POST /api/users/register**: User registration API.
  + **GET /api/recommendations**: Get personalized product recommendations based on user preferences.

**Database (MongoDB):**

* **Set Up MongoDB**:
  + Use **MongoDB** to store user data (account info, browsing history, etc.) and product details.
  + Store **user behaviors** (browsing patterns, previous purchases) for generating personalized recommendations.

Code:

const express = require('express');

const mongoose = require('mongoose');

const app = express();

mongoose.connect('mongodb://localhost:27017/ecommerceDB', {

useNewUrlParser: true,

useUnifiedTopology: true

});

const Product = mongoose.model('Product', {

name: String,

price: Number,

category: String,

image: String

});

app.get('/api/products', async (req, res) => {

const products = await Product.find();

res.json(products);

});

// Example product recommendation endpoint

app.get('/api/recommendations', (req, res) => {

const recommendedProducts = [

{ name: 'Product 1', price: 20, image: 'product1.jpg' },

{ name: 'Product 2', price: 30, image: 'product2.jpg' }

];

res.json(recommendedProducts);

});

app.listen(3000, () => console.log('Server running on port 3000'));

**Frontend Development (Web)**

Objective: Develop the web-based user interface to display products and manage user interactions.

Web App (React):

Set Up React Project:

Use create-react-app to initialize the project.

Install necessary libraries for routing (react-router-dom) and state management (redux or context API).

Components:

Product List: Display products dynamically fetched from the backend API.

Product Recommendations: Show personalized product recommendations based on the backend response.

User Authentication: Implement login/signup forms, session management using JWT.

Integration with Backend:

Fetch product data from the backend API (GET /api/products).

Fetch personalized recommendations (GET /api/recommendations).

Display them in a responsive and interactive UI.

Code:

import React, { useEffect, useState } from 'react';

function App() {

const [products, setProducts] = useState([]);

const [recommendations, setRecommendations] = useState([]);

useEffect(() => {

// Fetch products

fetch('/api/products')

.then(res => res.json())

.then(data => setProducts(data));

// Fetch personalized recommendations

fetch('/api/recommendations')

.then(res => res.json())

.then(data => setRecommendations(data));

}, []);

return (

<div>

<h1>Welcome to Our E-Commerce Platform</h1>

<h2>Recommended Products</h2>

<div id="product-list">

{recommendations.map(product => (

<div key={product.name}>

<img src={product.image} alt={product.name} />

<h3>{product.name}</h3>

<p>{product.price}$</p>

</div>

))}

</div>

</div>

);

}

**Features**

**1. User Interface (UI):**

* Minimalist design focusing on ease of navigation and interactivity.
* Dark mode and accessibility support.

**2. Recommendation System:**

* Personalized suggestions on the home page based on browsing history and purchases.
* "Customers who bought this also bought" section on product pages.
* Real-time updates to recommendations as users interact with the site.

**3. Admin Dashboard:**

* Manage products, categories, and inventory.
* View analytics on user interactions and sales trends**.**

**4. Payment Integration:**

* Popular options like Stripe, PayPal, or Razorpay.
* Support for multiple currencies and local payment methods.

**Development Phases (Sprints)**

**Sprint 1**:

* **Planning and Requirements:** Define the platform’s features, user stories, and technology stack. Design the overall system architecture.

**Sprint 2:**

* **Backend Development:** Develop the server-side functionality using Node.js, connect to MongoDB, and create API endpoints to handle product data and recommendations.

**Sprint 3:**

* **Frontend Development:** Design the user interface (UI) and integrate it with the backend. Implement the functionality for displaying products and recommendations.

**Sprint 4:**

* **Personalization Implementation**: Implement a recommendation engine to personalize products based on user behavior and preferences.

**Sprint 5:**

* **Testing and Deployment:** Perform testing (unit testing, integration testing), fix bugs, and deploy the platform to a production environment.

**Sprint 6:**

* **Maintenance and Updates**: Monitor the platform’s performance, gather user feedback, and implement further improvements and features.

**Mobile Development (React Native)**

Objective: Develop a mobile app that replicates the e-commerce functionality on mobile devices.

Mobile App (React Native):

* Set Up React Native Project:
  + Initialize the React Native project using npx react-native init.
  + Install libraries for navigation (react-navigation) and state management.
* Components:
  + Product List: Fetch and display products from the backend API.
  + Personalized Recommendations: Show recommended products based on user behavior.
  + User Authentication: Implement login/signup and session management

**Testing and Debugging**

Objective: Ensure that the platform works as expected and is free from bugs.

* Unit Testing: Write tests for API endpoints (backend) and UI components (frontend and mobile).
* Integration Testing: Test the interaction between frontend and backend.
* UI/UX Testing: Test the user experience for smooth navigation and interface consistency.

**Deployment**

* Objective: Deploy the platform to production.
* Backend Deployment: Deploy the Express backend on a platform like Heroku or AWS.
* Frontend Deployment: Deploy the React app on platforms like Netlify or Vercel.
* Mobile App Deployment: Publish the mobile app on Google Play and Apple App Store..

**Maintenance:**

* Regular Updates: Keep backend and frontend dependencies up-to-date.
* Database Management: Monitor and optimize database performance.
* Bug Fixes: Address user-reported issues promptly.
* Enhancements: Add new features and improve recommendations over time.
* User Support: Provide assistance through FAQs and customer support.

**Advantages & Applications:**

**Advantages**

 **Enhanced User Experience**: Personalization improves satisfaction.

 **Increased Conversion Rates**: Tailored recommendations boost purchases.

 **Better Retention**: Personalized experiences encourage repeat customers.

 **Data-Driven Insights**: Track and leverage user behavior for marketing.

 **Competitive Edge**: Personalization helps differentiate from competitors..

**Applications**

 Retail E-Commerce: Personalized suggestions based on browsing and buying.

 Fashion: Recommend clothes and accessories based on preferences.

 Electronics: Suggest compatible gadgets or accessories.

 Food Delivery: Recommend meals based on past orders or preferences.

 Health & Beauty: Personalized skincare or wellness product suggestions.

 Travel: Recommend destinations or packages based on past trips.

 Subscription Services: Personalized content recommendations (e.g., Netflix, Spotify).

 B2B E-Commerce: Recommend tools/services based on business needs.

**Conclusion:**

Designing and developing an e-commerce platform with personalized product recommendations offers significant advantages for both businesses and consumers. By integrating personalized recommendations, businesses can enhance user experience, increase conversion rates, and foster customer loyalty. The platform's maintenance is crucial to ensure continuous performance, security, and user satisfaction. The ability to gather and analyze user data enables the delivery of tailored recommendations, improving the overall shopping experience. With a wide range of applications across industries such as retail, fashion, electronics, and more, this type of platform has the potential to transform online shopping into a more engaging and efficient process, benefiting both consumers and businesses alike.