**HTML**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Virtual Farmer's Market</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css" rel="stylesheet">

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

<link rel="stylesheet" href="styles1.css">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const products = {

Fruits: [

{ name: "Apples", price: 100, image: "image2.jpg" },

{ name: "Bananas", price: 50, image: "image2.jpg" },

{ name: "Oranges", price: 80, image: "image2.jpg" },

{ name: "Grapes", price: 120, image: "image2.jpg" },

{ name: "Mangoes", price: 150, image: "image2.jpg" },

{ name: "Pineapples", price: 70, image: "image2.jpg" },

],

Vegetables: [

{ name: "Carrots", price: 40, image: "image3.jpg" },

{ name: "Potatoes", price: 30, image: "image3.jpg" },

{ name: "Tomatoes", price: 35, image: "image3.jpg" },

{ name: "Onions", price: 25, image: "image3.jpg" },

{ name: "Broccoli", price: 60, image: "image3.jpg" },

{ name: "Spinach", price: 20, image: "image3.jpg" },

],

Dairy: [

{ name: "Milk", price: 50, image: "image4.jpg" },

{ name: "Cheese", price: 200, image: "image4.jpg" },

{ name: "Yogurt", price: 40, image: "image4.jpg" },

{ name: "Butter", price: 150, image: "image4.jpg" },

{ name: "Paneer", price: 250, image: "image4.jpg" },

{ name: "Cream", price: 100, image: "image4.jpg" },

],

};

let cart = [];

function addToCart(item, quantity) {

const existingItem = cart.find(cartItem => cartItem.name === item.name);

if (existingItem) {

existingItem.quantity += quantity;

} else {

cart.push({ ...item, quantity });

}

updateCartPreview();

}

function updateCartPreview() {

const cartModalBody = document.getElementById("cart-modal-body");

const totalPriceElement = document.getElementById("total-price");

cartModalBody.innerHTML = "";

let totalPrice = 0;

cart.forEach(item => {

const itemTotal = item.price \* item.quantity;

totalPrice += itemTotal;

cartModalBody.innerHTML += `

<div class="d-flex justify-content-between mb-2">

<span>${item.name} (${item.quantity} kg)</span>

<span>₹${itemTotal}</span>

</div>

`;

});

totalPriceElement.innerText = `₹${totalPrice}`;

}

function updateCatalog(categories) {

const catalogContainer = document.getElementById("product-container");

catalogContainer.innerHTML = ""; // Clear existing products

categories.forEach(category => {

if (products[category]) {

products[category].forEach((item) => {

const productCard = `

<div class="col-md-4">

<div class="card">

<img src="${item.image}" class="card-img-top" alt="${item.name}">

<div class="card-body">

<h5 class="card-title">${item.name}</h5>

<p class="card-text">₹${item.price}/kg</p>

<label for="quantity-${item.name}">Quantity (kg):</label>

<input type="number" id="quantity-${item.name}" class="form-control mb-3" min="1" value="1">

<button class="btn btn-success" onclick="addToCart(products['${category}'].find(p => p.name === '${item.name}'), parseInt(document.getElementById('quantity-${item.name}').value))">Add to Cart</button>

</div>

</div>

</div>

`;

catalogContainer.innerHTML += productCard;

});

}

});

}

function showMainCategories() {

const catalogContainer = document.getElementById("product-container");

catalogContainer.innerHTML = `

<div class="col-md-4">

<div class="card" onclick="updateCatalog(['Fruits'])">

<img src="image2.jpg" class="card-img-top" alt="Fruits" height="200" width="300">

<div class="card-body">

<h5 class="card-title">Fruits</h5>

<p class="card-text">Fresh and organic fruits sourced directly from farmers.</p>

</div>

</div>

</div>

<div class="col-md-4">

<div class="card" onclick="updateCatalog(['Vegetables'])">

<img src="image3.jpg" class="card-img-top" alt="Vegetables" height="200" width="300">

<div class="card-body">

<h5 class="card-title">Vegetables</h5>

<p class="card-text">Locally grown vegetables for your daily needs.</p>

</div>

</div>

</div>

<div class="col-md-4">

<div class="card" onclick="updateCatalog(['Dairy'])">

<img src="image4.jpg" class="card-img-top" alt="Dairy" height="200" width="300">

<div class="card-body">

<h5 class="card-title">Dairy</h5>

<p class="card-text">Organic milk, cheese, and other dairy products.</p>

</div>

</div>

</div>

`;

}

document.addEventListener("DOMContentLoaded", () => {

showMainCategories();

document.querySelector("#clear-all").addEventListener("click", () => {

document.querySelectorAll(".filter-checkbox").forEach((checkbox) => (checkbox.checked = false));

showMainCategories();

});

document.querySelectorAll(".filter-checkbox").forEach((checkbox) => {

checkbox.addEventListener("change", () => {

const selectedCategories = Array.from(document.querySelectorAll(".filter-checkbox:checked")).map(cb => cb.value);

if (selectedCategories.length > 0) {

updateCatalog(selectedCategories);

} else {

showMainCategories();

}

});

});

});

</script>

</head>

<body>

<nav class="navbar navbar-expand-lg navbar-dark bg-success fixed-top">

<div class="container">

<a class="navbar-brand" href="#">Virtual Farmer's Market</a>

<button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarNav">

<ul class="navbar-nav ms-auto">

<li class="nav-item"><a class="nav-link" href="#about">About</a></li>

<li class="nav-item"><a class="nav-link" href="#catalog">Catalog</a></li>

<li class="nav-item"><a class="nav-link" href="#features">Features</a></li>

<li class="nav-item"><a class="nav-link" href="#reviews">Reviews</a></li>

<li class="nav-item"><a class="nav-link" href="#contact">Contact</a></li>

<li class="nav-item"><button class="btn btn-light me-2" id="signin-btn" data-bs-toggle="modal" data-bs-target="#signinModal">Sign In</button></li>

<li class="nav-item"><button class="btn btn-light" data-bs-toggle="modal" data-bs-target="#cartModal">Cart</button></li>

</ul>

</div>

</div>

<div class="search-bar-container">

<form action="/search" method="GET">

<input type="text" name="q" placeholder="Search for products..." class="search-input" />

<button type="submit" class="search-button">Search</button>

</form>

</div>

</nav>

<section class="hero py-5 text-center bg-success">

<h1>Welcome to the Virtual Farmer's Market</h1>

<p>Connecting local farmers with consumers for fresh and organic produce.</p>

<a href="#about" class="btn btn-success btn-lg">Learn More</a>

</section>

<section id="about" class="py-5">

<div class="container text-center">

<h2>About Us</h2>

<p style="text-align:justify;font-size:20px;">The Virtual Farmer's Market aims to bridge the gap between local farmers and urban consumers, offering a sustainable and transparent platform for fresh produce. We believe in empowering farmers by providing them with direct access to consumers, ensuring fair pricing and reducing food wastage.</p>

<p style="text-align:justify;font-size:20px;">Our platform is designed to support local communities by promoting healthy eating habits and environmental sustainability. By purchasing directly from farmers, consumers not only get the freshest produce but also contribute to a greener planet by minimizing transportation emissions.</p>

</div>

</section>

<section id="catalog" class="py-5 bg-light">

<div class="container">

<h2 class="text-center mb-4">Catalog</h2>

<div class="row">

<!-- Sidebar for Filters -->

<div class="col-md-3">

<div class="bg-white p-3 border rounded">

<h5>Filters</h5>

<hr>

<h6>Categories</h6>

<ul class="list-unstyled">

<li><input type="checkbox" class="filter-checkbox" value="Fruits"> Fruits</li>

<li><input type="checkbox" class="filter-checkbox" value="Vegetables"> Vegetables</li>

<li><input type="checkbox" class="filter-checkbox" value="Dairy"> Dairy</li>

</ul>

<button id="clear-all" class="btn btn-warning mt-3">Clear All</button>

</div>

</div>

<!-- Product Cards -->

<div class="col-md-9">

<div class="row" id="product-container">

</div>

</div>

</div>

</div>

</section>

<section id="features" class="py-5">

<div class="container">

<h2 class="text-center mb-4">Our Features</h2>

<div class="row">

<div class="col-md-4">

<div class="card">

<div class="card-body">

<h5 class="card-title">Farmer Profiles</h5>

<p class="card-text">Learn about the farmers who grow your produce and their sustainable practices.</p>

</div>

</div>

</div>

<div class="col-md-4">

<div class="card">

<div class="card-body">

<h5 class="card-title">Subscription Options</h5>

<p class="card-text">Subscribe to weekly or bi-weekly fresh produce deliveries tailored to your preferences.</p>

</div>

</div>

</div>

<div class="col-md-4">

<div class="card">

<div class="card-body">

<h5 class="card-title">Sustainability Metrics</h5>

<p class="card-text">Track the environmental impact and benefits of your purchases.</p>

</div>

</div>

</div>

</div>

</div>

</section>

<section id="reviews" class="py-5 bg-light">

<div class="container">

<h2 class="text-center mb-4">Customer Reviews</h2>

<div class="row">

<div class="col-md-4">

<blockquote class="blockquote">

<p>"The freshest produce I've ever had! Highly recommend this service."</p>

<footer class="blockquote-footer">Aryan Paode </footer>

</blockquote>

</div>

<div class="col-md-4">

<blockquote class="blockquote">

<p>"Excellent customer service and top-notch quality!"</p>

<footer class="blockquote-footer">Tanisha Verekar</footer>

</blockquote>

</div>

<div class="col-md-4">

<blockquote class="blockquote">

<p>"Great variety and prompt delivery."</p>

<footer class="blockquote-footer">Chaitanya & Sanyukta</footer>

</blockquote>

</div>

</div>

</div>

</section>

<div class="modal fade" id="signinModal" tabindex="-1" aria-labelledby="signinModalLabel" aria-hidden="true">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="signinModalLabel">Sign In</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

</div>

<div class="modal-body">

<form id="signin-form">

<div class="mb-3">

<label for="signin-email" class="form-label">Email</label>

<input type="email" class="form-control" id="signin-email" name="email" required>

<small id="signin-email-error" class="text-danger"></small>

</div>

<div class="mb-3">

<label for="signin-password" class="form-label">Password</label>

<input type="password" class="form-control" id="signin-password" name="password" required minlength="6">

<small id="signin-password-error" class="text-danger"></small>

</div>

<button type="submit" class="btn btn-success">Sign In</button>

</form>

<hr>

<p class="text-center">Don't have an account? <a href="#" data-bs-toggle="modal" data-bs-target="#registerModal">Register here</a></p>

</div>

</div>

</div>

</div>

<div class="modal fade" id="registerModal" tabindex="-1" aria-labelledby="registerModalLabel" aria-hidden="true">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="registerModalLabel">Register</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

</div>

<div class="modal-body">

<form id="register-form">

<div class="mb-3">

<label for="register-name" class="form-label">Name</label>

<input type="text" class="form-control" id="register-name" name="name" required>

<small id="register-name-error" class="text-danger"></small>

</div>

<div class="mb-3">

<label for="register-email" class="form-label">Email</label>

<input type="email" class="form-control" id="register-email" name="email" required>

<small id="register-email-error" class="text-danger"></small>

</div>

<div class="mb-3">

<label for="register-mobile" class="form-label">Mobile Number</label>

<input type="text" class="form-control" id="register-mobile" name="mobile" required pattern="\d{10}" title="Enter a valid 10-digit mobile number">

<small id="register-mobile-error" class="text-danger"></small>

</div>

<div class="mb-3">

<label for="register-address" class="form-label">Address</label>

<textarea class="form-control" id="register-address" name="address" required></textarea>

<small id="register-address-error" class="text-danger"></small>

</div>

<div class="mb-3">

<label for="register-pincode" class="form-label">PIN Code</label>

<input type="text" class="form-control" id="register-pincode" name="pincode" required pattern="\d{6}" title="Enter a valid 6-digit PIN code">

<small id="register-pincode-error" class="text-danger"></small>

</div>

<button type="submit" class="btn btn-primary">Register</button>

</form>

</div>

</div>

</div>

</div>

<div class="modal fade" id="cartModal" tabindex="-1" aria-labelledby="cartModalLabel" aria-hidden="true">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="cartModalLabel">Your Cart</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

</div>

<div class="modal-body" id="cart-modal-body">

<!-- Cart items will be dynamically added here -->

</div>

<div class="modal-footer">

<h5>Total: <span id="total-price">₹0</span></h5>

<button type="button" class="btn btn-primary">Checkout</button>

</div>

</div>

</div>

</div>

<footer class="py-3 bg-success text-white text-center">

<div class="container">

<p>&copy; 2025 Virtual Farmer's Market. All Rights Reserved.</p>

</div>

</footer>

</body>

</html>

**Script**

document.addEventListener("DOMContentLoaded", () => {

fetchProducts(); // Fetch products once on page load

// Handle Clear All button

document.getElementById("clear-all").addEventListener("click", () => {

document.querySelectorAll(".filter-checkbox").forEach((checkbox) => (checkbox.checked = false));

fetchProducts(); // Fetch all products when filters are cleared

});

// Handle filter checkboxes

document.querySelectorAll(".filter-checkbox").forEach((checkbox) => {

checkbox.addEventListener("change", () => {

const selectedCategories = Array.from(document.querySelectorAll(".filter-checkbox:checked"))

.map((cb) => cb.value);

fetchProducts(selectedCategories); // Fetch products based on selected filters

});

});

});

// Function to fetch products (with optional filtering)

function fetchProducts(selectedCategories = []) {

let url = 'http://127.0.0.1:5001/products';

// Add query parameters for filtering

if (selectedCategories.length > 0) {

url += `?categories=${selectedCategories.join(',')}`;

}

fetch(url)

.then(response => response.json())

.then(data => {

console.log("Fetched products:", data); // Debug: Log fetched data

populateCatalog(data); // Populate the catalog

})

.catch(error => console.error("Error fetching products:", error));

}

// Function to dynamically display products in the catalog

function populateCatalog(products) {

const catalogContainer = document.getElementById('product-container');

catalogContainer.innerHTML = ''; // Clear existing content

if (!products.length) {

catalogContainer.innerHTML = '<p>No products found.</p>';

return;

}

products.forEach(product => {

const productCard = `

<div class="col-md-4">

<div class="card">

<img src="http://127.0.0.1:5001/${product.image\_path}" class="card-img-top" alt="${product.name}">

<div class="card-body">

<h5 class="card-title">${product.name}</h5>

<p class="card-text">Price/Kg: ₹${product.price}</p>

<p class="card-text">Stock: ${product.stock}</p>

<div class="input-group mb-3">

<input type="number" class="form-control" placeholder="Quantity" min="1" max="${product.stock}" id="quantity-${product.id}">

<button class="btn btn-primary" onclick="addToCart('${product.id}', '${product.name}', ${product.price})">Add to Cart</button>

</div>

</div>

</div>

</div>

`;

catalogContainer.innerHTML += productCard;

});

}

function updateCartModal() {

const cartModalBody = document.getElementById('cart-modal-body');

const totalPriceElement = document.getElementById('total-price');

cartModalBody.innerHTML = ''; // Clear existing cart items

let total = 0;

if (cart.length === 0) {

cartModalBody.innerHTML = '<p>Your cart is empty.</p>';

totalPriceElement.textContent = '₹0';

return;

}

cart.forEach(item => {

total += item.total;

const cartItem = `

<div class="cart-item">

<p>${item.name} (₹${item.price} per kg) - ${item.quantity} kg</p>

<p>Total: ₹${item.total}</p>

</div>

`;

cartModalBody.innerHTML += cartItem;

});

totalPriceElement.textContent = `₹${total}`;

}

let cart = [];

function addToCart(productId, productName, productPrice) {

const quantityInput = document.getElementById(`quantity-${productId}`);

const quantity = parseInt(quantityInput.value);

if (!quantity || quantity < 1) {

alert("Please enter a valid quantity.");

return;

}

// Check if the product already exists in the cart

const existingProduct = cart.find(item => item.id === productId);

if (existingProduct) {

existingProduct.quantity += quantity;

existingProduct.total = existingProduct.quantity \* existingProduct.price;

} else {

const cartItem = {

id: productId,

name: productName,

price: productPrice,

quantity: quantity,

total: productPrice \* quantity

};

cart.push(cartItem);

}

alert(`Added ${quantity} ${productName}(s) to the cart.`);

updateCartModal(); // Update the cart modal dynamically

}

function refreshCaptcha() {

const captchaImage = document.getElementById('captcha-image');

captchaImage.src = '/captcha-image?' + new Date().getTime(); // Prevent caching

}

document.getElementById('signin-form').addEventListener('submit', async function (e) {

e.preventDefault();

const captchaInput = document.getElementById('captcha').value;

const response = await fetch('/validate-captcha', {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify({ captcha: captchaInput })

});

const result = await response.json();

if (result.success) {

alert('CAPTCHA verified! Proceeding...');

// Submit the form or proceed with further validation

} else {

alert(result.message);

refreshCaptcha(); // Refresh CAPTCHA if validation fails

}

});

document.getElementById('signin-form').addEventListener('submit', async function (e) {

e.preventDefault();

const email = document.getElementById('signin-email').value;

const password = document.getElementById('signin-password').value;

const response = await

fetch('/login', {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify({ email, password })

});

const result = await response.json();

if (result.success) {

alert('Login successful!');

// Redirect or show dashboard

} else {

alert(result.message);

}

});

document.getElementById('register-form').addEventListener('submit', async function (e) {

e.preventDefault();

const name = document.getElementById('register-name').value;

const email = document.getElementById('register-email').value;

const password = document.getElementById('register-password').value;

const confirmPassword = document.getElementById('register-confirm-password').value;

const mobile\_no = document.getElementById('register-mobile').value;

const address = document.getElementById('register-address').value;

const pincode = document.getElementById('register-pincode').value;

if (password !== confirmPassword) {

console.error("Passwords do not match!");

document.getElementById('register-confirm-password-error').textContent = "Passwords do not match!";

return;

}

console.log("Sending registration data:", { name, email, password, mobile\_no, address, pincode });

const response = await fetch('/register', {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify({ name, email, password, mobile\_no, address, pincode }),

});

console.log("Response status:", response.status);

const text = await response.text();

console.log("Response body:", text);

try {

const result = JSON.parse(text);

if (response.ok) {

alert(result.message);

} else {

alert("Error: " + result.message);

}

} catch (err) {

console.error("Failed to parse response as JSON:", err, text);

alert("An unexpected error occurred.");

}

});

**app**

from flask import Flask, jsonify, send\_from\_directory, request, session, g

from flask\_cors import CORS

from flask import send\_file

import bcrypt

import os

import sqlite3

from PIL import Image, ImageDraw, ImageFont

import random

import string

import os

app = Flask(\_\_name\_\_)

app.secret\_key = 'your\_secret\_key' # Required for session management

CORS(app)

@app.route('/show-tables')

def show\_tables():

conn = get\_db\_connection()

cursor = conn.execute("SELECT name FROM sqlite\_master WHERE type='table';")

tables = [row[0] for row in cursor.fetchall()]

conn.close()

return jsonify({"tables": tables})

# Function to connect to the database

def get\_db\_connection():

db\_path = r'C:\Users\Pratap\Documents\Sem-6 folder\FOE\Lab project\vfm\_database.db'

print("Connecting to database at:", db\_path)

conn = sqlite3.connect(db\_path)

conn.row\_factory = sqlite3.Row # Allows access to results as dictionaries

return conn

# Home route

@app.route('/')

def home():

return send\_from\_directory('static', 'index.html') # Serve the main HTML file

# Route to fetch products with optional filtering

@app.route('/products')

def fetch\_products():

categories = request.args.get('categories') # Get the query parameter

conn = get\_db\_connection()

if categories:

categories = categories.split(',') # Convert to list

query = "SELECT \* FROM products WHERE category IN ({})".format(

','.join('?' for \_ in categories)

)

cursor = conn.execute(query, categories)

else:

cursor = conn.execute('SELECT \* FROM products') # Fetch all products if no filters are applied

products = [dict(row) for row in cursor.fetchall()]

conn.close()

return jsonify(products)

# Route to add to cart

@app.route('/add-to-cart', methods=['POST'])

def add\_to\_cart():

data = request.get\_json()

print("Cart Data:", data)

return jsonify({"message": "Item added to cart successfully", "cart": data})

# CAPTCHA generation route

@app.route('/captcha-image')

def generate\_captcha():

captcha\_text = ''.join(random.choices(string.ascii\_uppercase + string.digits, k=6))

session['captcha\_text'] = captcha\_text

# Create CAPTCHA image

image = Image.new('RGB', (150, 50), (255, 255, 255))

draw = ImageDraw.Draw(image)

font\_path = os.path.join(os.path.dirname(\_\_file\_\_), 'arial.ttf')

font = ImageFont.truetype(font\_path, 30)

draw.text((10, 5), captcha\_text, fill=(0, 0, 0), font=font)

# Save the image to a stream and return it

from io import BytesIO

img\_io = BytesIO()

image.save(img\_io, 'PNG')

img\_io.seek(0)

return send\_file(img\_io, mimetype='image/png')

# CAPTCHA validation route

@app.route('/validate-captcha', methods=['POST'])

def validate\_captcha():

user\_captcha = request.json.get('captcha', '')

if user\_captcha == session.get('captcha\_text', ''):

return jsonify({"success": True, "message": "CAPTCHA verified"})

else:

return jsonify({"success": False, "message": "Incorrect CAPTCHA"}), 400

# Route to serve images

@app.route('/images/<path:filename>')

def serve\_image(filename):

return send\_from\_directory('static/images', filename)

# Health check route

@app.route('/health')

def health\_check():

return jsonify({"status": "running"})

# User registration

@app.route('/register', methods=['POST'])

def register\_user():

try:

data = request.get\_json()

if not data:

return jsonify({"success": False, "message": "No data provided"}), 400

name = data.get('name')

email = data.get('email')

password = data.get('password')

mobile\_no = data.get('mobile\_no')

address = data.get('address')

pincode = data.get('pincode')

if not name or not email or not password or not mobile\_no or not address or not pincode:

return jsonify({"success": False, "message": "All fields are required"}), 400

# Add user to the users table

conn = get\_db\_connection()

conn.execute('INSERT INTO users (name, email, password, mobile\_no, address, pincode) VALUES (?, ?, ?, ?, ?, ?)',

(name, email, hashed\_password, mobile\_no, address, pincode))

conn.commit()

conn.close()

return jsonify({"success": True, "message": "User registered successfully"})

except sqlite3.IntegrityError:

return jsonify({"success": False, "message": "Email already exists"}), 400

except Exception as e:

print(f"Error: {e}")

return jsonify({"success": False, "message": "Internal Server Error"}), 500

# User login

@app.route('/login', methods=['POST'])

def login\_user():

data = request.get\_json()

email = data['email']

password = data['password']

conn = get\_db\_connection()

user = conn.execute('SELECT \* FROM users WHERE email = ?', (email,)).fetchone()

conn.close()

if user:

# Directly compare the plain text passwords

if password == user['password']:

session['user\_id'] = user['id']

return jsonify({"success": True, "message": "Login successful"})

else:

return jsonify({"success": False, "message": "Invalid email or password"}), 401

else:

return jsonify({"success": False, "message": "Invalid email or password"}), 401

# Load the logged-in user

@app.before\_request

def load\_logged\_in\_user():

user\_id = session.get('user\_id')

if user\_id is None:

g.user = None

else:

conn = get\_db\_connection()

g.user = conn.execute('SELECT \* FROM users WHERE id = ?', (user\_id,)).fetchone()

conn.close()

# Protected route example

@app.route('/protected')

def protected\_route():

if g.user is None:

return jsonify({"success": False, "message": "Unauthorized"}), 403

return jsonify({"success": True, "message": "Welcome, {}".format(g.user['name'])})

# Main entry point

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True, port=5001) # Flask runs on port 5001

@app.errorhandler(500)

def handle\_internal\_server\_error(e):

return jsonify({"success": False, "message": "Internal Server Error"}), 500

app.run(debug=True, port=5001)