**BASIC Web Development Projects**

**Task 1: Product Landing Page**

**Skills Required**: CSS, Image editing

1. **Layout with Columns**:
   * Use CSS Flexbox or Grid to create a responsive column layout.
   * Example: Divide the page into a header, main content, and footer. The main content can have a two-column layout with product details and an image.
2. **Image Editing**:
   * Use tools like Photoshop, GIMP, or online editors like Pixlr to crop and optimize images for the web.
   * Ensure images are web-optimized (e.g., use JPEG for photos, PNG for graphics with transparency).
3. **Design Templates**:
   * Use free templates from sources like Bootstrap, or design your own using CSS.
   * Ensure the design is clean and user-friendly.

Example HTML structure:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Product Landing Page</title>

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

</head>

<body>

<header>

<h1>Product Name</h1>

<nav>

<ul>

<li><a href="#features">Features</a></li>

<li><a href="#pricing">Pricing</a></li>

<li><a href="#contact">Contact</a></li>

</ul>

</nav>

</header>

<main>

<section class="product-info">

<div class="column">

<h2>Product Features</h2>

<p>Describe the features of your product here.</p>

</div>

<div class="column">

<img src="product-image.jpg" alt="Product Image">

</div>

</section>

</main>

<footer>

<p>&copy; 2024 Your Company</p>

</footer>

</body>

</html>

Example CSS (styles.css):

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

box-sizing: border-box;

}

header {

background-color: #333;

color: white;

padding: 1rem;

text-align: center;

}

nav ul {

list-style: none;

padding: 0;

}

nav li {

display: inline;

margin: 0 1rem;

}

nav a {

color: white;

text-decoration: none;

}

.product-info {

display: flex;

justify-content: space-around;

padding: 2rem;

}

.column {

flex: 1;

margin: 1rem;

}

footer {

background-color: #333;

color: white;

text-align: center;

padding: 1rem;

position: fixed;

bottom: 0;

width: 100%;

}

**Task 2: Basic Portfolio Website**

**Skills Required**: Responsiveness, UX design, HTML, CSS, Icons

1. **Structure**:
   * Create sections for about me, projects, experience, and contact information.
   * Use semantic HTML tags (e.g., <header>, <section>, <footer>).
2. **Responsiveness**:
   * Use CSS media queries to ensure the site looks good on all devices (mobile, tablet, desktop).
   * Example: @media (max-width: 600px) { ... } for mobile styles.
3. **UX Design**:
   * Keep the design simple and intuitive.
   * Use consistent colors, fonts, and spacing.
   * Ensure easy navigation.
4. **Icons**:
   * Use icon libraries like Font Awesome for social media links and other icons.
   * Example: <i class="fab fa-github"></i> for a GitHub icon.

Example HTML structure:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>My Portfolio</title>

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

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css">

</head>

<body>

<header>

<h1>My Portfolio</h1>

<nav>

<ul>

<li><a href="#about">About Me</a></li>

<li><a href="#projects">Projects</a></li>

<li><a href="#contact">Contact</a></li>

</ul>

</nav>

</header>

<main>

<section id="about">

<h2>About Me</h2>

<p>Brief introduction about yourself.</p>

</section>

<section id="projects">

<h2>Projects</h2>

<p>Showcase your projects here.</p>

</section>

<section id="contact">

<h2>Contact</h2>

<p>Email: youremail@example.com</p>

<p>GitHub: <a href="https://github.com/yourusername"><i class="fab fa-github"></i></a></p>

</section>

</main>

<footer>

<p>&copy; 2024 Your Name</p>

</footer>

</body>

</html>

Example CSS (styles.css):

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

box-sizing: border-box;

}

header {

background-color: #333;

color: white;

padding: 1rem;

text-align: center;

}

nav ul {

list-style: none;

padding: 0;

}

nav li {

display: inline;

margin: 0 1rem;

}

nav a {

color: white;

text-decoration: none;

}

main {

padding: 2rem;

}

section {

margin: 2rem 0;

}

footer {

background-color: #333;

color: white;

text-align: center;

padding: 1rem;

position: fixed;

bottom: 0;

width: 100%;

}

@media (max-width: 600px) {

nav li {

display: block;

margin: 0.5rem 0;

}

}

Task 3: Weather Forecast Website

**Skills Required**: JavaScript, Node.js, ReactJS

1. **API Integration**:
   * Use a weather API like OpenWeatherMap to fetch data.
   * Example: fetch('https://api.openweathermap.org/data/2.5/weather?q=London&appid=YOUR\_API\_KEY')
2. **User Input**:
   * Create a form for the user to enter their location.
   * Example: <input type="text" id="location" placeholder="Enter your city">
3. **Display Data**:
   * Use React components to display the weather data.
   * Example: Create a WeatherCard component to show temperature, conditions, and forecast.
4. **Location Detection**:
   * Use the Geolocation API to automatically detect the user's location.
   * Example: navigator.geolocation.getCurrentPosition(success, error)

Example React structure:

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

function WeatherApp() {

const [location, setLocation] = useState('');

const [weatherData, setWeatherData] = useState(null);

const fetchWeather = async () => {

const response = await fetch(`https://api.openweathermap.org/data/2.5/weather?q=${location}&appid=YOUR\_API\_KEY`);

const data = await response.json();

setWeatherData(data);

};

const handleLocationChange = (e) => {

setLocation(e.target.value);

};

const handleFetchWeather = () => {

fetchWeather();

};

useEffect(() => {

navigator.geolocation.getCurrentPosition((position) => {

const { latitude, longitude } = position.coords;

// Fetch weather for the detected location

});

}, []);

return (

<div>

<h1>Weather Forecast</h1>

<input type="text" value={location} onChange={handleLocationChange} placeholder="Enter your city" />

<button onClick={handleFetchWeather}>Get Weather</button>

{weatherData && (

<div>

<h2>{weatherData.name}</h2>

<p>Temperature: {weatherData.main.temp}</p>

<p>Conditions: {weatherData.weather[0].description}</p>

</div>

)}

</div>

);

}

export default WeatherApp;