Institute Of Computer Engineering Technology

iCET Certified Master

Web Development
Final Project

Objectives

- Develop visually appealing and responsive website design that meets the need of the target audience and the project requirements.
- Demonstrate knowledge and understanding of HTML, CSS and JavaScript concepts and principles in the project's implementation.

Course Work Requirements

- You should use your knowledge on HTML, CSS and JavaScript to implement this project.
- Refer this course work guideline document to understand the project requirments.
- The website must be responsive and display properly on a range of devices, including desktop, tablets and smart phones.
- The website should be well-designed and visually appealing, incoparating appropriate use of colors, typorgraphy, layout and imagery.
- The website should be optimized for performance, including the use of optimized images , multimedia files and follow the best practises of web optimization.

Project Submisson

- You are required to create a Git repository on Github at the beginning of the project. All updates and changes made to the project should be committed and pushed to the repository regularly. The repository should include all necessary files and assets required to run the website, including HTML, CSS, JavaScript, images, and any other multimedia files.
- Your github link should be submitted to below LMSon or before the deadline. Please read instructions under the topic "Time Frame and Submission Process" in this document to get better understanding about the submission process.

Introduction

As part of the Web Development module's final project, you are required to develop a web application that displays real-time weather details through an API. The guidelines and requirements for this project are outlined in this document, so please read it carefully.

The web application you will be developing is a weather details display application for users. You should use HTML, CSS, and JavaScript as the primary technologies, and you can utilize any JavaScript libraries to enhance the functionality and aesthetics of the application.

The weather details display application is a publicly accessible web application that displays weather details for different areas. The application should provide predicted weather data for up to three days and show past weather data for the last seven days. Users should be able to view weather details either by their location or by searching for specific areas or countries. More information about the project requirements can be found in the following topics.

Project Flow

Step 01 : Project Analysis and Planning

1. Review project requirements and understand key objectives.
2. Analyze functional and non-functional requirements.
3. Identify dependencies and plan the project timeline accordingly.
4. Prioritize requirements based on importance.
5. Assess the feasibility of implementing requirements within given constraints.
6. Evaluate different technical solutions using HTML, CSS, and JavaScript.
7. Create a requirement specification document outlining finalized requirements, priorities, dependencies, and technical solutions.

Step 02 : Design and Prototyping

1. Gather design requirements.
2. Conduct research for inspiration.
3. Create wireframes to outline basic structure and layout.
4. Design mockups to represent the visual look and feel.
5. Consider user experience (UX) aspects for a smooth user journey.
6. Develop interactive prototypes.
7. Finalize design assets.

Step 03 : Software Development

1. Set up the development environment.
2. Break down the project into smaller tasks.
3. Write HTML structure based on the design.
4. Apply CSS styling to HTML elements.
5. Add interactivity and functionality using JavaScript.
6. Test and debug code regularly.
7. Follow coding best practices and standards.
8. Implement responsive design for various devices.
9. Optimize website performance.

Step 04 : Deployment

1. Choose a suitable hosting platform based on requirements.
2. Set up necessary infrastructure for production environment.
3. Prepare code for deployment, ensuring organization and optimization.
4. Upload code and assets to the production environment.
5. Configure domain and DNS settings for accessibility.
6. Test deployment thoroughly.
7. Implement security measures.
8. Monitor and optimize performance.
9. Launch the web application for public access.

By following this project flow, you can effectively analyze, plan, design, develop, and deploy your web application, ensuring a systematic and organized approach to the project.

Deployment

You should deploy your web application to github pages. "Github pages" free web hosting service that host your application directly from the repository. After commiting your changes to the repository you can enable github pages from settings.

You can find more about github pages from : https://pages.github.com/

Project Requirements

- Current weather display: Show the current weather conditions, including temperature, humidity, wind speed, and weather description.
- Forecast display: Provide a forecast for upcoming days, including predicted temperatures, weather conditions, and chance of precipitation.
- Location-based weather: Allow users to view weather information for their current location or search for weather in specific cities, regions, or countries.
- Historical weather data: Provide access to past weather data for a specified period, allowing users to view weather trends and historical records.
- Interactive maps: Include interactive maps to visualize weather patterns, satellite imagery, or radar information.
- Alerts and notifications: Display severe weather alerts or notifications for users based on their location or subscribed areas.
- Multiple units and formats: Support different units of measurement for temperature, wind speed, and rainfall, and provide options to switch between metric and imperial systems.
- Responsive design: Ensure the web application is responsive and optimized for various devices, including desktops, tablets, and mobile phones.
- User customization: Allow users to customize the display preferences, such as choosing between light and dark themes or selecting preferred units of measurement.
- API integration: Integrate with a weather data API to fetch real-time and forecast weather data. You should use "Weather API" service for this. You can find the documentation link and other needed informations regarding API under the topic of "Weather API" in this document.
- Performance optimization: Optimize the performance of the web application, minimizing load times and optimizing data retrieval from the API.
- User-friendly interface: Design an intuitive and user-friendly interface, making it easy for users to navigate, search for weather information, and interact with the application.
- Cross-browser compatibility: Ensure the web application works correctly on major web browsers, including Chrome, Firefox, Safari, and Edge.
- Security: Implement secure protocols and practices to protect user data and ensure secure communication with the weather API.

Weather API

Link for the documentation : https://www.weatherapi.com/docs/

To display real-time weather data in your web application, you can integrate the "Weather API" service that provides access to their free API. Follow these steps to access the API and retrieve real-time data for your application:

- Register and obtain API credentials: Sign up for the Weather API service and obtain your API credentials, including an API key or any other required authentication details.
- Read the API documentation: Familiarize yourself with the Weather API's documentation to understand its endpoints, parameters, and response formats. This will help you make the appropriate API calls and parse the data returned by the API.
- Set up API calls: Configure your web application to make HTTP requests to the Weather API's endpoints using the appropriate HTTP methods (GET, POST, etc.) and including the required parameters (e.g., location, units of measurement). - (Consider that Weather API gives only 1 Million free calls per month)
- Handle API responses: Receive the API responses in your application and handle the data returned by the API. This may involve parsing the JSON or XML response and extracting the relevant weather information, such as temperature, humidity, wind speed, and weather conditions.
- Update the user interface: Use the retrieved weather data to update the user interface of your web application dynamically. Display the real-time weather information in an organized and visually appealing manner, ensuring a seamless user experience.
- Implement error handling: Account for potential errors or exceptions that may occur during the API request or response process. Implement appropriate error handling mechanisms to handle network failures, API rate limits, and any other potential issues.
- Test and refine: Thoroughly test the integration of the Weather API within your web application to ensure the accurate retrieval and display of real-time weather data. Refine and optimize the integration as necessary to enhance performance and reliability.

By following these steps and referring to the Weather API's full documentation, you can successfully access and integrate real-time weather data into your web application, providing users with up-to-date weather information

Recommended References

- MDN Web Documentation : https://developer.mozilla.org/en-US/
- Fetch API : https://developer.mozilla.org/enUS/docs/Web/API/Fetch_API/Using_Fetch
- Bootstrap : https://getbootstrap.com/docs/5.3/getting-started/introduction/
- Tailwind CSS : https://tailwindcss.com/docs/installation
- jQuery Doc : https://api.jquery.com/
- Inspirations for UI Designs : https://dribbble.com/
- Weather API Doc : https://www.weatherapi.com/docs/
- Icons8 : https://icons8.com/icons/
- Flat UI Colors :