**Weather App Documentation**

The Main objective of our Weather Forecast App is to provide users with accurate, real-time weather information and forecasts to help them plan their daily activities and stay safe during severe weather conditions.

The main functionalities and features of this app includes:

**Current weather Condition:**

Location: Displays the specific location (e.g., Arayat, PH).

Temperature: Shows the current temperature (e.g., 31°C).

Weather Description: Provides a brief description of the weather (e.g., Few Clouds).

Date and Time: Indicates the date and time of the weather report.

**Additional Weather Details:**

Humidity: Displays the current humidity level.

Wind Speed: Shows the wind speed

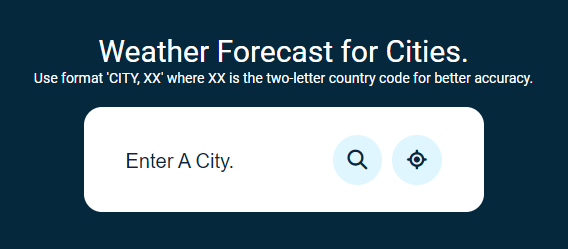
**Location Based Reporting:**

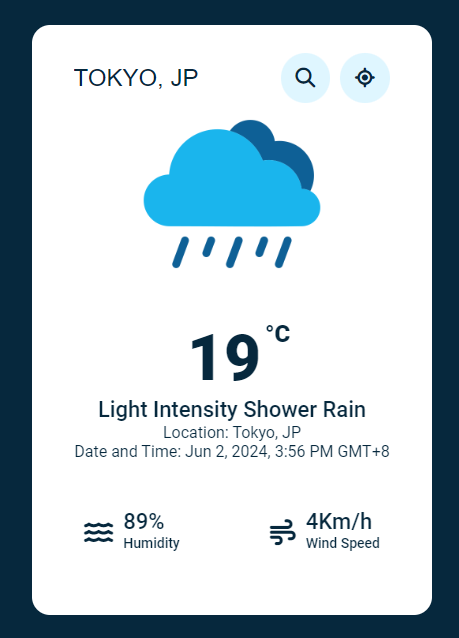
Automatically sets the location to the current location of the user.

It also has search functionality to allow users to search weather conditions in different cities.

Also shows an image in the background of the City.

**When you first open the App**



**Sample Weather**

**CODE STRUCTURE**

**Program Structure for Weather Forecast Web Application**

1. Introduction

- Purpose :

"Our weather app website provides accurate and up-to-date weather forecasts to help users plan their day with confidence. Whether you're checking the forecast for work, travel, or leisure, our platform delivers reliable information tailored to your location. With intuitive design and comprehensive data, we strive to empower users to make informed decisions, whatever the weather may bring."

2. Project Overview

- Project Name : Weather Forecast Web Application

- Description : A web application that provides weather forecasts for various cities using WebSocket and external APIs.

3. System Requirements

- Software Requirements :

- Web Browser (Chrome, Firefox, etc.)

- Node.js

- Hardware Requirements :

- Basic computer or server to host the application

4. Technology Stack

- Frontend HTML, CSS, JavaScript

- Backend Node.js, WebSocket

- APIs OpenWeather API, Pexels API

- Libraries : Font Awesome for icons

5. Directory Structure

/weather-forecast-app

│

├── /images # Contains image assets

│ └── 404.png # Image for not found error

│ └── clear.png # Image for clear weather

│ └── rain.png # Image for rain weather

│ └── snow.png # Image for snow weather

│ └── cloud.png # Image for cloudy weather

│ └── mist.png # Image for misty weather

│

├── /public

│ └── style.css # CSS file for styling

│ └── script.js # JavaScript file for frontend logic

│

├── /server

│ └── server.js # Node.js server with WebSocket setup

│

├── index.html # Main HTML file

│

└── package.json # NPM package file

6. Code Explanation

6.1. HTML (`index.html`)

- Purpose : The main structure of the web application, including input fields, buttons, and placeholders for weather information.

- Key Elements :

- `search-box`: Input and buttons for searching by city or using current location.

- `weather-box`: Displays the weather information.

- `weather-details`: Additional weather details like humidity and wind speed.

- `not-found`: Error message display for invalid searches.

6.2. CSS (`style.css`)

- Purpose : Styling for the web application, including layout, colors, fonts, and responsiveness.

- Key Classes :

- `.container`: Main container for the content.

- `.search-box`: Styles for the search input and buttons.

- `.weather-box`: Styles for the weather display area.

- `.weather-details`: Styles for additional weather information.

- `.not-found`: Styles for the error display.

6.3. JavaScript (`script.js`)

- Purpose : Handles the frontend logic, including WebSocket connections and DOM manipulation.

- Key Functions :

- `displayWeather(data)`: Updates the DOM with weather information.

- Event listeners for search button and location button.

6.4. Node.js Server (`server.js`)

- Purpose : Backend server setup with WebSocket to communicate with the frontend and fetch weather data from the OpenWeather API.

- Key Functions :

- WebSocket connection handling

- Fetching weather data based on city or geolocation

7. Installation and Setup

- Steps :

1. Clone the repository.

2. Navigate to the project directory.

3. Install dependencies: `npm install`.

4. Start the server: `node server/server.js`.

5. Open `index.html` in a web browser.

8. Usage Instructions

- How to Use :

- Enter a city name and click the search icon.

- Click the location icon to use the current location for weather data.

- View the weather information displayed in the application.

9. API References

- OpenWeather API : Provides weather data.

- Pexels API : Provides background images based on the city and country.

10. Troubleshooting

- Common Issues :

- API key errors

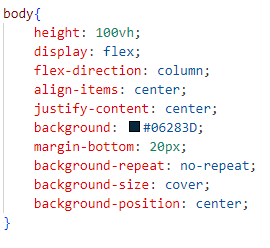
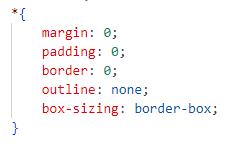
- WebSocket connection issues

- Ensure valid API keys are used.

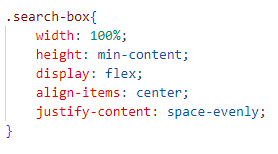
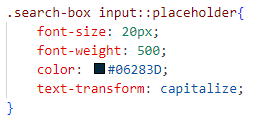
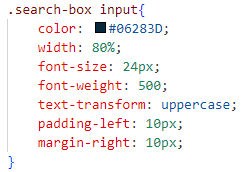
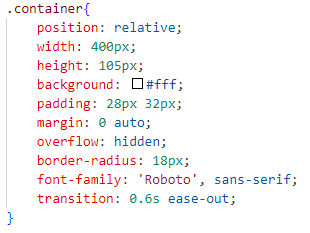
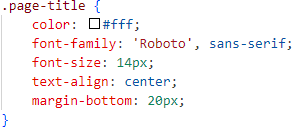
- Check WebSocket server status.

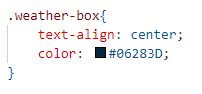
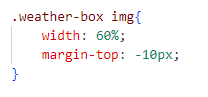
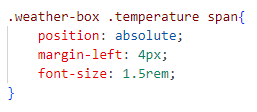
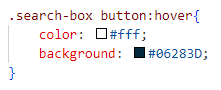
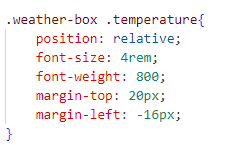
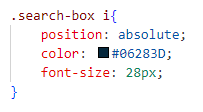
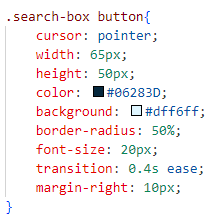
**Html Explanation:**

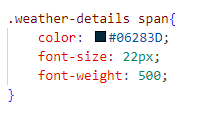
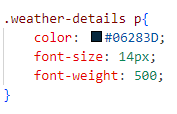
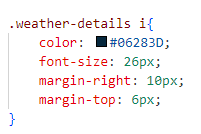
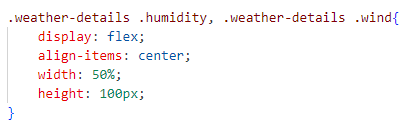
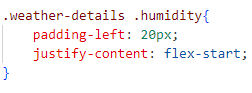
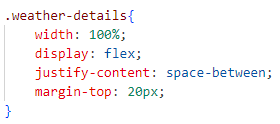
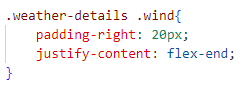
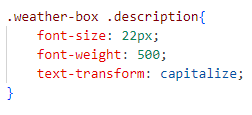


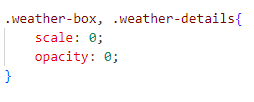
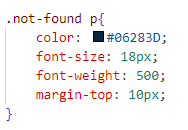
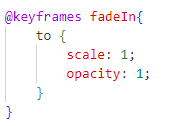
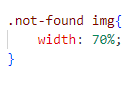
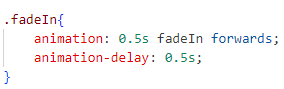
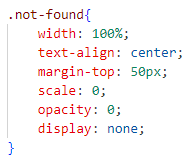


**CSS Explanation:**



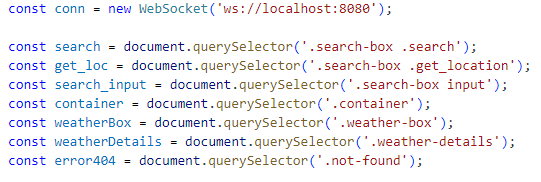


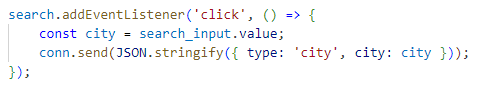
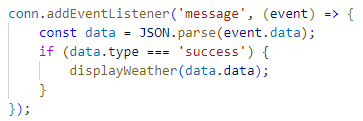
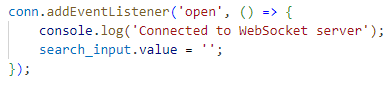
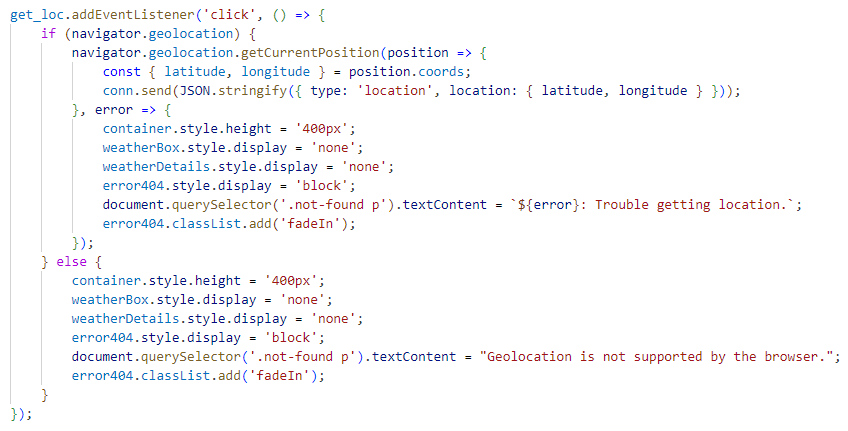


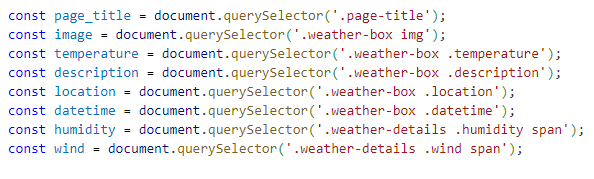


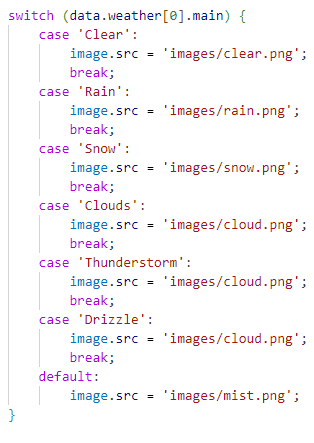




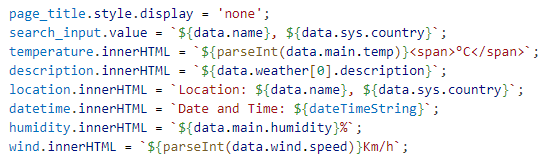
**Javascript Explanation:**



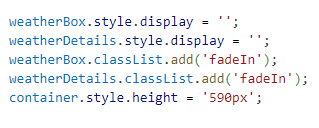














**Websocket Explanation:**