**Weather Forecast Application**

**Task 1 - Display Cities in a table:**

* ~~Show all cities in a table format with~~**~~infinite scroll~~**~~(use this api to get the city data:~~[~~https://public.opendatasoft.com/explore/dataset/geonames-all-cities-with-a-population-1000/api/?disjunctive.cou\_name\_en&sort=name~~](https://www.google.com/url?q=https://public.opendatasoft.com/explore/dataset/geonames-all-cities-with-a-population-1000/api/?disjunctive.cou_name_en%26sort%3Dname&sa=D&source=editors&ust=1712591121289265&usg=AOvVaw0_9M6cR0CMx8u_AwM0KpHK)~~).~~
* ~~The table should have columns city name, country, timezone etc.~~
* ~~Implement search as you type for this table. Utilize an autocomplete feature to suggest possible locations as users type.~~
* ~~Implement filter and sorting for each column~~
* ~~Clicking on the city name, should take to to the weather page for this city~~
* ~~Right clicking on the city name and opening in the new tab should also open the weather page for the city but in the new tab.~~

**Task 2- Weather page:**

* ~~You can reach this page by clicking on the city name in the table section as mentioned in the previous section.~~
* ~~Use~~[~~https://openweathermap.org~~](https://www.google.com/url?q=https://openweathermap.org&sa=D&source=editors&ust=1712910765903976&usg=AOvVaw0_rY_7UlcpEAmB4nfmiqLx)~~free api to display weather on this page~~
* ~~Display whatever information you get from the free api including but not limited to Current weather (temperature, weather description, humidity, wind speed, and atmospheric pressure), forecast (temperature highs and lows, weather descriptions, and precipitation chances) etc.~~
* Optional: You can get innovative and do stuff like display location on map, show options for changing unit etc
* Once weather data is loaded on this page, display basics like day high/low, for the city, on the cities table page as well.

01d – clear sky / day

01n – clear sky / night

02d – clouds/ day (few clouds) (clouds - 23)

02n- clouds / night

04d- clouds / day (overcast clouds) (clouds – 100, 85, 90, 86, 95, 98)

04n- clouds/ night (overcast clouds) (clouds- 86)

04d- clouds / day (broken clouds) (clouds-66, 75, 76, 82)

04n- clouds / day (broken clouds) (clouds-81)

03d- clouds (scattered clouds) (clouds- 30, 37, 28)

03n- clouds (scattered clouds) (clouds- 45)

10d – Rain / day (light rain) (clouds- 100)

10n – Rain / day (light rain) (clouds- 100)

13d – snow / day (clouds-100)

13n – snow / night (clouds-100)

https://api.openweathermap.org/data/2.5/forecast?lat={lan}&lon={lon}&appid=e9935208d8fd3bc8c7746fee02a95b2d

https://api.openweathermap.org/data/2.5/forecast?lat=29.2225&lon=79.5286&appid=e9935208d8fd3bc8c7746fee02a95b2d

**Other requirements:**

* Styling:
* Use appropriate color pallets & styled components to design layouts and different pages.
* Implement dynamic backgrounds based on the current weather conditions.
* Use appropriate images or animations to represent different weather conditions (e.g., sunny, rainy, cloudy).
* Responsive Design:
* Ensure that the application is responsive and works well on different screen sizes.
* Implement media queries or responsive design techniques.
* Error Handling:
* Handle errors gracefully in case of failed API requests or invalid search queries.
* Display error messages to users when necessary.
* State Management:
* Utilize React state management to display data and avoid re-fetching.
* Manage weather data state centrally and pass down necessary data as props.
* Type Safety:
* Utilize TypeScript for type safety throughout the application.
* Define interfaces or types for weather objects and API responses.

**Optional Features:**

* Utilize the preferred stack, such as **Next.js** for the frontend framework, **MST (MobX State Tree)** for state management, and **Tailwind CSS** for styling.
* Implement geolocation to automatically detect and display the weather for the user's current location.
* Maintain history of viewed locations' weather.
* Provide options to switch between different units of measurement (e.g., Celsius/Fahrenheit, metric/imperial).
* Add a feature to save favourite locations for quick access.

**Deployment:**

* Deploy your project with any hosting provider.
* Choose a hosting provider that best fits your needs:
* Netlify
* Vercel
* GitHub Pages
* AWS Amplify
* Firebase Hosting

**Submission Guidelines:**

* Create a GitHub repository for your project.
* Document any setup instructions or prerequisites in a README.md file.
* Ensure that your code is well-documented and follows best practices.
* Test your application thoroughly to ensure it works as expected.
* Deploy your application on a hosting server.
* Submit the GitHub repository link for evaluation. [https://forms.gle/yPeaPWegqv2Tab829](https://www.google.com/url?q=https://forms.gle/yPeaPWegqv2Tab829&sa=D&source=editors&ust=1713026325287515&usg=AOvVaw1r-qfZpWRp6npgDz0NKQqo)

**Note:**

* You are encouraged to add any extra features or enhancements to showcase your skills and creativity.
* Consider user experience and accessibility while designing the application.
* Make sure to handle edge cases and error scenarios gracefully.

**APIs:**

* To get locations use open public API. [https://public.opendatasoft.com/explore/dataset/geonames-all-cities-with-a-population-1000/table/?disjunctive.cou\_name\_en&sort=name](https://www.google.com/url?q=https://public.opendatasoft.com/explore/dataset/geonames-all-cities-with-a-population-1000/table/?disjunctive.cou_name_en%26sort%3Dname&sa=D&source=editors&ust=1713026325288406&usg=AOvVaw2_m6SOmRyt4uyIYSXzbObv)
* Use a free account to generate an API key to access OpenWeatherMap current and 5-day forecast weather APIs. ([https://openweathermap.org](https://www.google.com/url?q=https://openweathermap.org&sa=D&source=editors&ust=1713026325288704&usg=AOvVaw2w6YOvCP0n_tTMisdqmqlX))