**Task - 1**

**Weather Watch**

Weather Watch lets you fetch and view current weather for various cities. Since this is an academic exercise, the list of cities and weather conditions is provided as a static dataset using a simulated API.

**Key Skills**

The intent of this exercise is to give you a thorough workout on the following React features:

1. Class Components
2. Function Components
3. Side Effects - Querying an API using Fetch API
4. State management
5. Using componentDidMount() & componentDidUpdate() lifecycle methods
6. Using debounce (from Lodash) to limit API calls
7. Using Refs
8. Conditional rendering
9. Type-checking using PropTypes

User Stories

* The app opens with the home screen as shown below. The app defaults to Bengaluru as the location:

Diagram

Description automatically generated

* As the user starts to type in the name of a location, available options should show up as shown in the image below. Clicking on a location button loads and displays weather for that location. You can also change the temperature units from Celsius to Fahrenheit by using the units drop down menu

Diagram

Description automatically generated

**API Endpoints**

We’re using an in-memory database and a simulated API that you can query right away using the fetch API. This API features a limited database of records from the OpenWeatherMap service. Data returned is static and does not reflect actual weather conditions.  
  
The following endpoints can be used readily:  
  
**GET      https://api.weatherserver.com/weather/cities/<keyword\_to\_search>** : This endpoint allows you to search for available cities in the database. It returns an Array with the following shape:

[{

  id: Number,

  name: String

}]

The id here is a unique location code that is to be used for querying the weather using the /weather/current endpoint as described below.

**Note:** Since this is a limited & static database, available locations include London, Chicago, New York, New Delhi, Tokyo, Toronto, New Jersey, Colombo, Kolkata, Chennai, Bengaluru, Chandigarh, Brisbane, Sydney, Queenstown, Vancouver, Paris, Rome & Los Angeles.

**GET      https://api.weatherserver.com/weather/current/<cityId>/<units> :** This endpoint accepts the unique identifier of the location (cityId) as well as temperature unit keywords (C for Celcius or F for Fahrenheit) and it returns back an object with the following shape:

{

  location: String,

  icon: String,

  conditions: String,

  temp: Number,

  temp\_max: Number,

  temp\_min: Number,

  feels\_like: Number,

  wind\_speed: Number,

  wind\_direction: Number,

  pressure: Number,

  humidity: Number

}

**Component Tree**

Graphical user interface, diagram

Description automatically generated

**Important Points**

**General Notes**

* Implement prop type validators across all components where props have been implemented.
* Create all components in the /src/components
* Stylesheet for the assignment is provided to you and comes imported into the index.js file.
* Use the design specification documents for all components to understand how the UIs are to be built and the exact CSS selectors to use so that components render correctly.
* Please ensure all naming conventions exactly as and where mentioned for successful completion and grading of the assignment.

**Notes for Component: <App />**

The App Component represents the main component that composes together other components in the application and also implements state management.

* This should be a Class component
* Implement componentDidMount() and componentDidUpdate() lifecycle methods
* Use the Fetch API for querying the API
* Implement a loading indicator using a state variable flag. For rendering a UI, simply create a <div className=”is-loading” />
* Intercept network errors when using the Fetch API and display a UI. To render the UI, simply create a <div className=”error-panel” />
* List of functions to implement:
  1. searchLocations(keyword): This function should query /weather/cities/<keyword> and should implement the debounce function from Lodash which is pre-installed. Just import the function using import debounce from "lodash.debounce";
  2. getWeather() : This function is invoked whenever the app loads for the first time or whenever the search feature is used to select a new location. This would also be invoked whenever the SetUnits component is used to select a different temperature unit.