Ex.No : 9	**Develop a Weather Application using React**

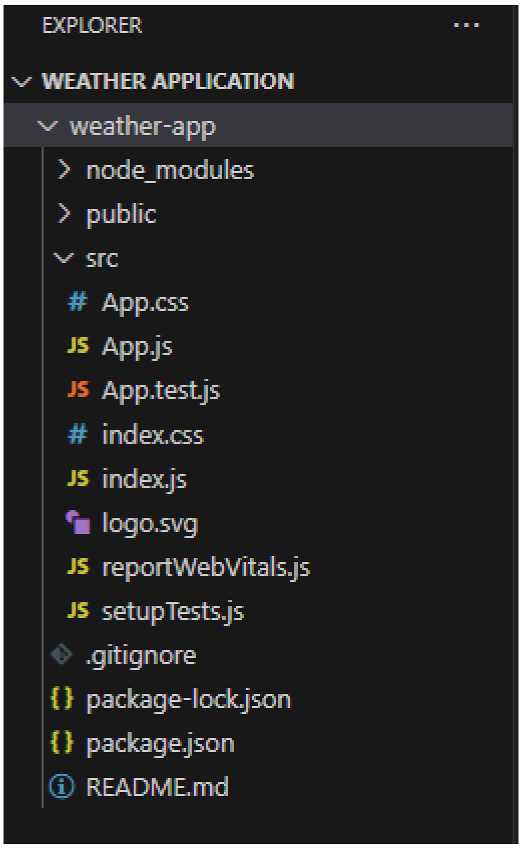
Aim:

The primary aim of this project is to develop a weather application using React that allows users to search for current weather conditions by city name. The application will utilize the OpenWeatherMap API to fetch and display weather data, including temperature, weather description, and wind speed, while also handling loading states and errors gracefully.

Algorithm:

1. **Initialize the Project:** Create a React app and install required packages (Axios, FontAwesome).
2. **Set Up State:** Use useState to manage input, weather (containing loading, data, error).
3. **Date Formatting:** Create toDateFunction to return the current date formatted as "Day Date Month".
4. **Search Function:** Define search to handle Enter key press, triggering API call to OpenWeatherMap.
5. **API Call:** Use Axios to fetch weather data with city name, unit, and API key as parameters.
6. **Handle Response**: On success, update weather state with data; on error, set error to true.
7. **Render UI:** Display title, input field, loading spinner, error message, and weather details conditionally.
8. **Style Application:** Use CSS to enhance the visual appearance of the app.
9. **Test Application:** Run the app and validate functionality with different city names.
10. **Refine and Improve:** Add features, improve error handling, and optimize UI based on user feedback.

Project Structure:



```
EXPLORER ...
WEATHER APPLICATION
  weather-app
    node_modules
    public
    src
      App.css
      App.js
      App.test.js
      index.css
      index.js
      logo.svg
      reportWebVitals.js
      setupTests.js
    .gitignore
    package-lock.json
    package.json
    README.md
```

Source Code:

App.js

```
import { Oval } from 'react-loader-spinner';
import React, { useState } from 'react';
import axios from 'axios';
import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
import { faFrown } from '@fortawesome/free-solid-svg-icons';
import './App.css';

function GfGWeatherApp() {
    const [input, setInput] = useState('');
    const [weather, setWeather] = useState({
        loading: false,
        data: {},
        error: false,
    });

    const toDateFunction = () => {
        const months = [
            'January',
            'February',
            'March',
            'April',
            'May',
            'June',
            'July',
            'August',
            'September',
            'October',
            'November',
            'December',
        ];
        const WeekDays = [
            'Sunday',
            'Monday',
            'Tuesday',
            'Wednesday',
            'Thursday',
            'Friday',
            'Saturday',
        ];
```

```
        const currentDate = new Date();
        const date = `${WeekDays[currentDate.getDay()]} ${currentDate.getDate()}
${months[currentDate.getMonth()]
          }`;
        return date;
    };

    const search = async (event) => {
        if (event.key === 'Enter') {
            event.preventDefault();
            setInput('');
            setWeather({ ...weather, loading: true });
            const url = 'https://api.openweathermap.org/data/2.5/weather';
            const api_key = 'f00c38e0279b7bc85480c3fe775d518c';
            await axios
                .get(url, {
                    params: {
                        q: input,
                        units: 'metric',
                        appid: api_key,
                    },
                })
                .then((res) => {
                    console.log('res', res);
                    setWeather({ data: res.data, loading: false, error: false });
                })
                .catch((error) => {
                    setWeather({ ...weather, data: {}, error: true });
            setInput('');
            console.log('error', error);
        });
    }
};

return (
    <div className="App">
        <h1 className="app-name">
            Weather App
        </h1>
        <div className="search-bar">
            <input
                type="text"
                className="city-search"
                placeholder="Enter City Name.."
                name="query"
                value={input}
                onChange={(event) => setInput(event.target.value)}
                onKeyPress={search}
            />
        </div>
        {weather.loading && (
            <>
                <br />
                <br />
                <Oval type="Oval" color="black" height={100} width={100} />
            </>
```

```
          )}
          {weather.error && (
            <>
              <br />
              <br />
              <span className="error-message">
                <FontAwesomeIcon icon={faFrown} />
                <span style={{ fontSize: '20px' }}>City not found</span>
              </span>
            </>
          )}
          {weather && weather.data && weather.data.main && (
            <div>
              <div className="city-name">
                <h2>
                  {weather.data.name}, <span>{weather.data.sys.country}</span>
                </h2>
              </div>
              <div className="date">
                <span>{toDateFunction()}</span>
              </div>
              <div className="icon-temp">
                <img
                  className=""
                  src={`https://openweathermap.org/img/wn/${weather.data.weather[0
].icon}@2x.png`}
                  alt={weather.data.weather[0].description}
                />
```

```
                    {Math.round(weather.data.main.temp)}
                    <sup className="deg">°C</sup>
                </div>
                <div className="des-wind">
                    <p>{weather.data.weather[0].description.toUpperCase()}</p>
                    <p>Wind Speed: {weather.data.wind.speed}m/s</p>
                </div>
            </div>
        )}
    </div>
  );
}

export default GfGWeatherApp;
```

App.css

```
* {
  font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Oxygen,
    Ubuntu, Cantarell, 'Open Sans', 'Helvetica Neue', sans-serif;
}

html {
  background-color: #f7f7f7;
}
.app-name {
  font-size: 2.3rem;
  color:chocolate;
  margin-bottom: 16px;
```

```css
}
.App {
   display: flex;
   flex-direction: column;
   justify-content: center;
   align-items: center;
   width: 600px;
   min-height: 440px;
   background-color: rgb(255, 255, 255);
   text-align: center;
   margin: 128px auto;
   border-radius: 10px;
   padding-bottom: 32px;
   box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
}
.city-search {
   width: 100%;
   max-width: 400px;
   box-sizing: border-box;
   border: 2px solid rgb(204, 204, 204);
   outline: none;
   border-radius: 20px;
   font-size: 16px;
   background-color: #e5eef0;
   background-position: 10px 12px;
   background-repeat: no-repeat;
   padding: 12px 40px 12px 40px;
   -webkit-transition: width 0.4s ease-in-out;
```

```
  transition: width 0.4s ease-in-out;
  color: #333;
}
.city-search:focus {
  width: 100%;
}
.city-name {
  font-size: 1.5rem;
  color: #444;
  margin-bottom: 8px;
}
.date {
  font-size: 1.25em;
  font-weight: 500;
  color: #777;
}
.icon-temp {
  font-size: 3rem;
  font-weight: 700;
  color: #1e2432;
  text-align: center;
}
.deg {
  font-size: 1.3rem;
  vertical-align: super;
}
.des-wind {
  font-weight: 500;
```

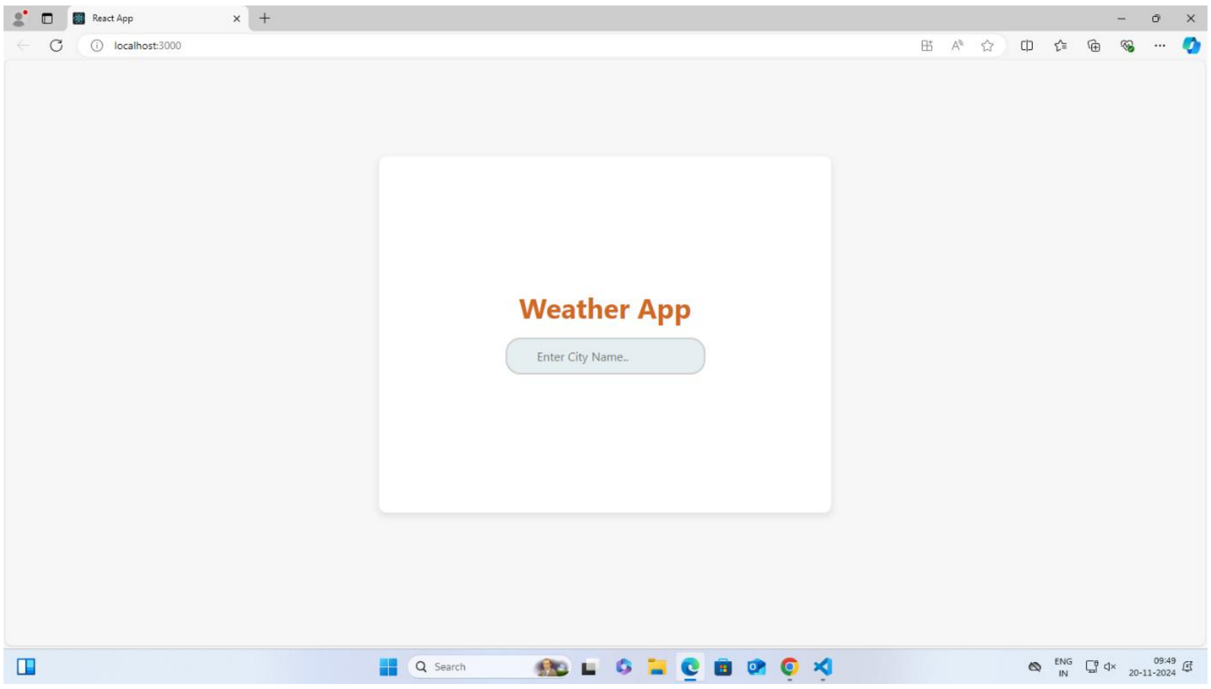
```css
    color: #666;
}
.error-message {
    display: block;
    text-align: center;
    color: #d32f2f;
    font-size: 24px;
    margin-top: auto;
}
.Loader {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100%;
}
.Loader>div {
    margin: 0 auto;
}
.weather-icon {
    display: flex;
    justify-content: center;
    align-items: center;
}
.weather-icon img {
    width: 100px;
    height: 100px;
    border-radius: 50%;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2); }
```

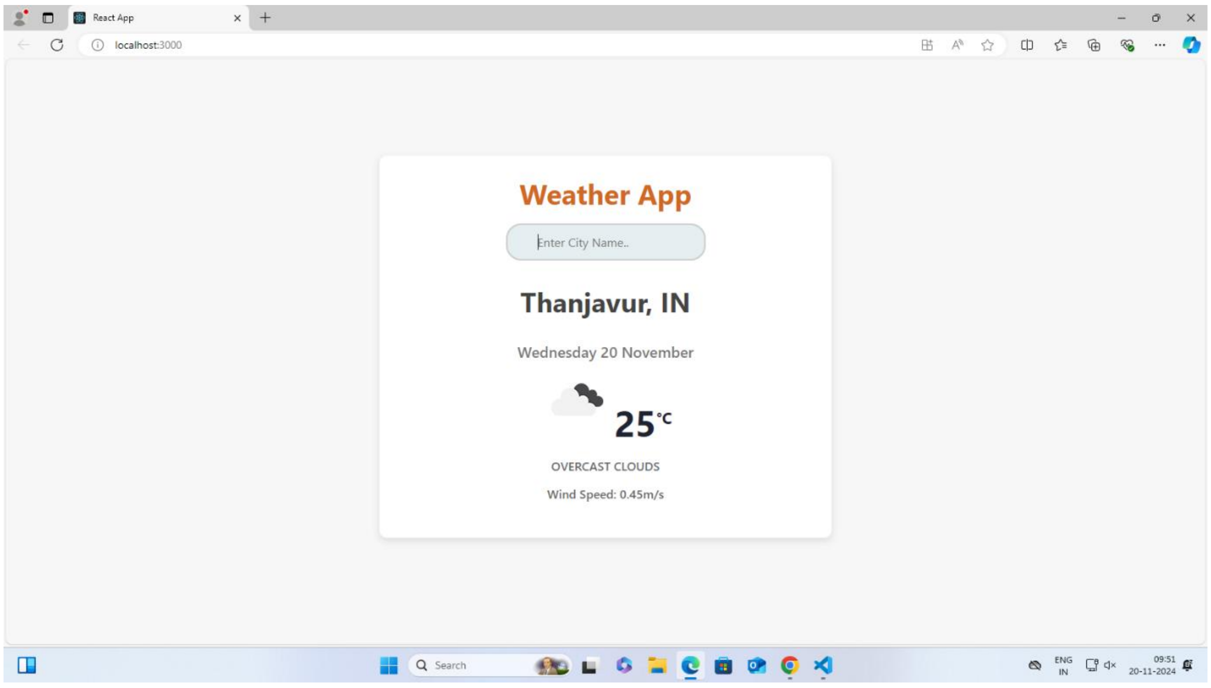
index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
reportWebVitals();
```

Output ScreenShots:




React App

localhost:3000

Weather App

Enter City Name..

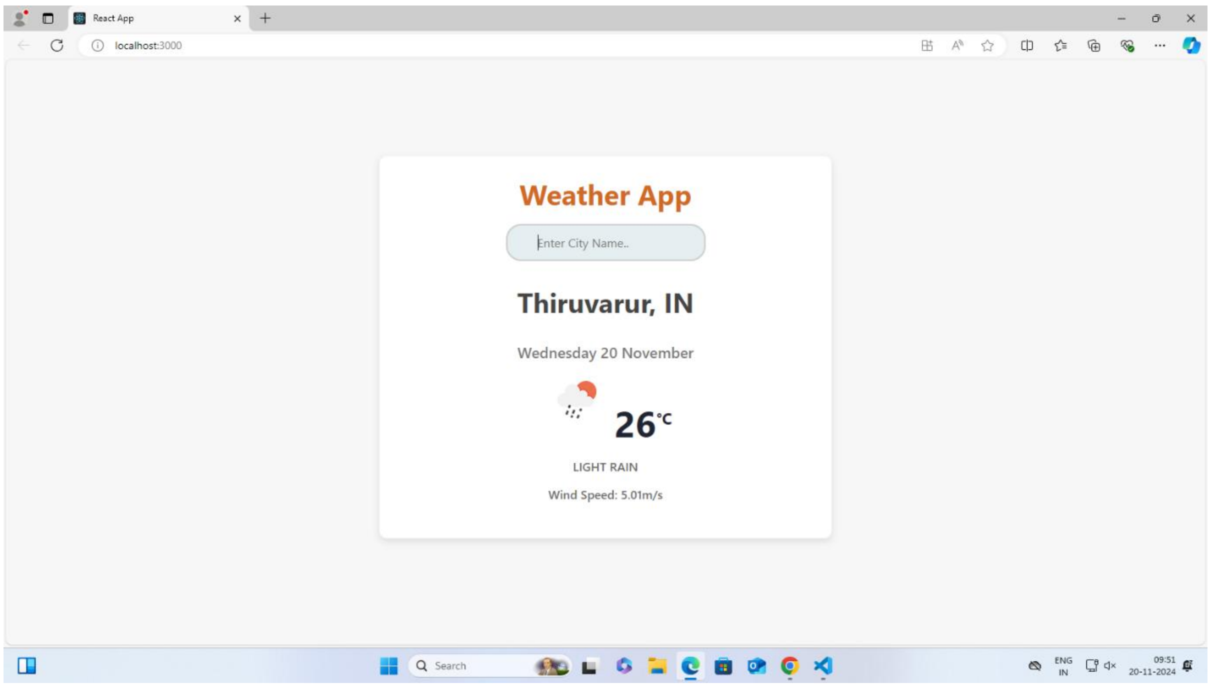
Thanjavur, IN

Wednesday 20 November

25°C

OVERCAST CLOUDS

Wind Speed: 0.45m/s


React App

localhost:3000

Weather App

Enter City Name..

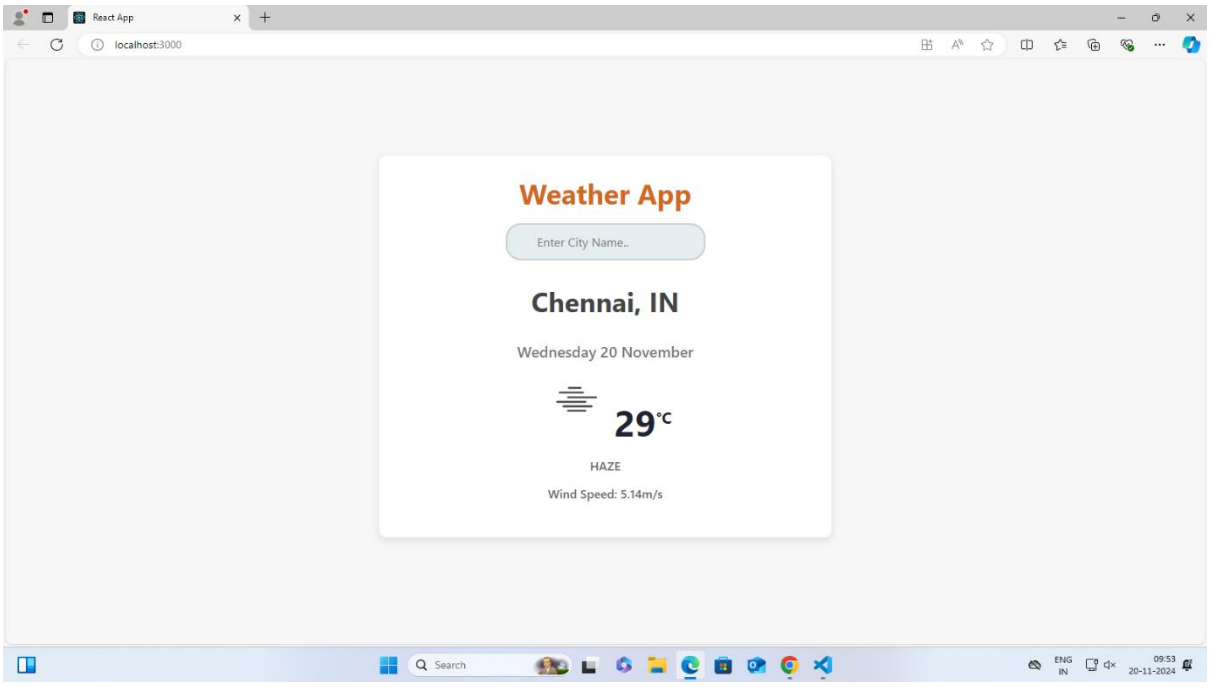
Thiruvarur, IN

Wednesday 20 November

26°C

LIGHT RAIN

Wind Speed: 5.01m/s



Result:

Thus the develop a weather application using React was implemented and executed successfully.