Rahul Singh

21011102081

CSE(IoT)-B

App.component.html

<div class="container">

  <div class="upper-data">

    <img src="" alt="">

    <img src="" alt="">

    <div class="weather-data">

      <div class="location"> Wellington </div>

      <div class="temperature">{{14.40 | number:'1.0-0'}}°C</div>

    </div>

  </div>

  <div class="lower-data">

      <div class="more-info-label">

        More Information

      </div>

      <div class="more-info-container">

        <div class="info-block">

          <div class="info-block-label">

            <img src="" alt="">

            <span>min</span>

          </div>

          <div class="info-block-value">

            10°C

          </div>

        </div>

        <div class="info-block">

          <div class="info-block-label">

            <img src="" alt="">

            <span>max</span>

          </div>

          <div class="info-block-value">

            20°C

          </div>

        </div>

        <div class="info-block">

          <div class="info-block-label">

            <img src="" alt="">

            <span>humidity</span>

          </div>

          <div class="info-block-value">

            77%

          </div>

        </div>

        <div class="info-block">

          <div class="info-block-label">

            <img src="" alt="">

            <span>wind</span>

          </div>

          <div class="info-block-value">

            11 km/h

          </div>

        </div>

      </div>

  </div>

</div>

Styles.css

:root {

    --blue-1: #3C4293;

    --blue-2: #4C52AD;

    --yellow-1: #FAC742;

    --white: #FFF;

    --grey-1: #EDEDED;

    --shadow-dark: rgba(0, 0, 0, 0.3);

    --shadow-light: rgba(255, 255, 255, 0.1);

}

\* {

    margin: 0;

    padding: 0;

    box-sizing: *border-box*;

}

body {

    font-family: *verdana*, *Verdana*, Geneva, *Tahoma*, *sans-serif*;

    font-size: 16px;

    width: 100%;

    height: 100vh;

    background-color: var(--blue-1);

    display: *flex*;

    justify-content: *center*;

    align-items: *center*;

}

.container {

    width: 400px;

    height: 80vh;

    background-color: var(--blue-2);

    border-radius: 20px;

    box-shadow: 10px 10px 10px var(--shadow-dark);

}

.upper-data {

    position: *relative*;

    overflow: *hidden*;

    width: 100%;

    height: 50%;

    border-top-left-radius: 20px;

    border-top-right-radius: 20px;

}

.lower-data {

    position: *relative*;

    overflow: *hidden*;

    width: 100%;

    height: 50%;

    border-bottom-left-radius: 20px;

    border-bottom-right-radius: 20px;

    padding: 1em;

    display: *flex*;

    flex-direction: *column*;

}

.upper-data img {

    position: *absolute*;

    top: 0;

    left: 0;

    width: 100%;

    height: 100%;

}

.weather-data {

    position: *relative*;

    z-index: 1;

    width: 100%;

    height: 100%;

    background-color: var(--shadow-dark);

    display: *flex*;

    flex-direction: *column*;

    align-items: *center*;

    justify-content: *center*;

}

.location {

    color: var(--white);

    text-align: *center*;

    font-size: 1.2em;

}

.temperature {

    color: var(--white);

    font-size: 4em;

    text-align: *center*;

    font-weight: 900;

}

.more-info-label {

    color: var(--white);

}

.more-info-container {

    flex: 1;

    background-color: var(--shadow-light);

    border-bottom-left-radius: 20px;

    border-bottom-right-radius: 20px;

    margin-top: 1em;

    display: *flex*;

    flex-direction: *row*;

    flex-wrap: *wrap*;

}

.info-block {

    width: 50%;

    display: *flex*;

    flex-direction: *row*;

}

.info-block-label {

    width: 50%;

    display: *flex*;

    flex-direction: *column*;

    justify-content: *center*;

    align-items: *center*;

}

.info-block-label img {

    width: 1.5em;

}

.info-block-label span {

    color: var(--white);

    font-size: 0.8em;

}

.info-block-value {

    width: 50%;

    display: *flex*;

    justify-content: *flex-start*;

    align-items: *center*;

    color: var(--white);

}

.search {

    margin-bottom: 1em;

    text-align: *center*;

}

.search input {

    background-color: var(--shadow-light);

    outline:*none*;

    border:*none*;

    border-radius: 20px;

    padding: 1em;

    color: var(--grey-1);

    font-size:0.8em;

    text-align: *center*;

}

App.component.ts

import { Component, OnInit } from '@angular/core';

import { WeatherService } from './services/weather.service';

@Component({

  selector: 'app-root',

  templateUrl: './app.component.html',

  styleUrls: ['./app.component.scss']

})

export class AppComponent implements OnInit{

  constructor(private weatherService: WeatherService){

  }

  cityName: string = 'Chennai';

  weatherData?: WeatherData;

  ngOnInit(): void {

*this*.getWeatherData(*this*.cityName);

*this*.cityName = '';

  }

  onSubmit() {

*this*.getWeatherData(*this*.cityName);

*this*.cityName = '';

  }

  private getWeatherData(cityName: string) {

*this*.weatherService.getWeatherData('Wellington')

      .subscribe({

        next: (response) => {

*this*.weatherData = response;

        console.log(response);

      }

    });

  }

}

App.module.ts

import { NgModule } from '@angular/core';

import { BrowserModule } from '@angular/platform-browser';

import { AppRoutingModule } from './app-routing.module';

import { AppComponent } from './app.component';

import { HttpClientModule } from '@angular/common/http';

import { FormsModule } from '@angular/forms';

@NgModule({

  declarations: [

    AppComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    HttpClientModule,

    FormsModule

  ],

  providers: [],

  bootstrap: [AppComponent]

})

export class AppModule { }

Weather.services.ts

import { HttpClient, HttpHeaders, HttpParams } from '@angular/common/http';

import { Injectable } from '@angular/core';

import { environment } from 'src/environments/environment.development';

import { Observable } from 'rxjs';

@Injectable({

  providedIn: 'root'

})

export class WeatherService {

  constructor(private http: HttpClient) { }

  getWeatherData(cityName: string): Observable<WeatherData>{

    return *this*.http.get(environment.weatherApiBaseUrl, {

      headers: new HttpHeaders()

      .set(environment.XRapidAPIHostHeaderName, environment.XRapidAPIHostHeaderValue)

      .set(environment.XRapidAPIKeyHeaderName, environment.XRapidAPIKeyHeaderValue),

      params: new HttpParams()

      .set('q', cityName)

      .set('units', 'metric')

      .set('mode', 'json')

    })

  }

}

Environment.ts

import { HttpClient, HttpHeaders, HttpParams } from '@angular/common/http';

import { Injectable } from '@angular/core';

import { environment } from 'src/environments/environment.development';

import { Observable } from 'rxjs';

@Injectable({

  providedIn: 'root'

})

export class WeatherService {

  constructor(private http: HttpClient) { }

  getWeatherData(cityName: string): Observable<WeatherData>{

    return *this*.http.get(environment.weatherApiBaseUrl, {

      headers: new HttpHeaders()

      .set(environment.XRapidAPIHostHeaderName, environment.XRapidAPIHostHeaderValue)

      .set(environment.XRapidAPIKeyHeaderName, environment.XRapidAPIKeyHeaderValue),

      params: new HttpParams()

      .set('q', cityName)

      .set('units', 'metric')

      .set('mode', 'json')

    })

  }

}

A screen shot of a device

Description automatically generated