Forecast.component.ts

import { ***Component***, OnInit } from '@angular/core';  
import { FormBuilder, FormGroup } from '@angular/forms';  
import { ApixuService } from '../apixu.service';  
  
@Component({  
 selector: 'app-forecast',  
 templateUrl: './forecast.component.html',  
 styleUrls: ['./forecast.component.css']  
})  
export class ForecastComponent implements OnInit {  
 public weatherSearchForm: FormGroup;  
 public weatherData: any;  
 public forecastData: any;  
  
 constructor(private formBuilder: FormBuilder, private apixuService: ApixuService) {}  
  
  
 ngOnInit() {  
 this.weatherSearchForm = this.formBuilder.group({  
 location: ['']  
 });  
 }  
 sendToAPIXU(formValues) {  
 // console.log(formValues.location);  
 this.apixuService  
 .getWeather(formValues.location)  
 .subscribe(data => {  
 this.weatherData = data;  
 ***console***.log(this.weatherData);  
 });  
 this.apixuService  
 .getForecast(formValues.location)  
 .subscribe(data => {  
 this.forecastData = data;  
 ***console***.log(this.forecastData);  
 });  
  
 }  
}

forecast.component.html

import { ***Component***, OnInit } from '@angular/core';  
import { FormBuilder, FormGroup } from '@angular/forms';  
import { ApixuService } from '../apixu.service';  
  
@Component({  
 selector: 'app-forecast',  
 templateUrl: './forecast.component.html',  
 styleUrls: ['./forecast.component.css']  
})  
export class ForecastComponent implements OnInit {  
 public weatherSearchForm: FormGroup;  
 public weatherData: any;  
 public forecastData: any;  
  
 constructor(private formBuilder: FormBuilder, private apixuService: ApixuService) {}  
  
  
 ngOnInit() {  
 this.weatherSearchForm = this.formBuilder.group({  
 location: ['']  
 });  
 }  
 sendToAPIXU(formValues) {  
 // console.log(formValues.location);  
 this.apixuService  
 .getWeather(formValues.location)  
 .subscribe(data => {  
 this.weatherData = data;  
 ***console***.log(this.weatherData);  
 });  
 this.apixuService  
 .getForecast(formValues.location)  
 .subscribe(data => {  
 this.forecastData = data;  
 ***console***.log(this.forecastData);  
 });  
  
 }  
}

apixu.service.ts

import { ***Injectable*** } from '@angular/core';  
import { HttpClient } from '@angular/common/http';  
  
@Injectable({  
 providedIn: 'root'  
})  
export class ApixuService {  
  
 constructor(private http: HttpClient) {  
 }  
errorMessageText = 'The city name you have entered is incorrect. Please enter a valid city name.';  
 getWeather(location) {  
 return this.http.get(  
 'https://api.apixu.com/v1/current.json?key=aca39b995d1241e78b300948193108&q=' + location  
  
 );  
 }  
  
 getForecast(location) {  
 ***console***.log(location);  
 return this.http.get(  
 'https://api.apixu.com/v1/forecast.json?key=aca39b995d1241e78b300948193108&q=' + location  
 );  
 }  
}

app.component.html

<router-outlet></router-outlet>

Apixu.service.spec.ts

import { ***TestBed*** } from '@angular/core/testing';  
  
import { ApixuService } from './apixu.service';  
  
describe('ApixuService', () => {  
 beforeEach(() => ***TestBed***.configureTestingModule({}));  
  
 it('should be created', () => {  
 const service: ApixuService = ***TestBed***.get(ApixuService);  
 expect(service).toBeTruthy();  
 });  
});

app.component.ts

import { ***Component*** } from '@angular/core';  
  
@Component({  
 selector: 'app-root',  
 templateUrl: './app.component.html',  
 styleUrls: ['./app.component.css']  
})  
export class AppComponent {  
 title = 'weatherapp';  
}

app.module.ts

import { BrowserModule } from '@angular/platform-browser';  
import { ***NgModule*** } from '@angular/core';  
import {RouterModule} from '@angular/router';  
import {***allAppRoutes***} from './routes';  
import { AppRoutingModule } from './app-routing.module';  
import { AppComponent } from './app.component';  
import { ForecastComponent } from './forecast/forecast.component';  
import { ReactiveFormsModule } from '@angular/forms';  
import { HttpClientModule } from '@angular/common/http';  
import { ApixuService } from './apixu.service';  
@NgModule({  
 declarations: [  
 AppComponent,  
 ForecastComponent  
 ],  
 imports: [  
 BrowserModule,  
 AppRoutingModule,  
 RouterModule.*forRoot*(***allAppRoutes***),  
 ReactiveFormsModule,  
 HttpClientModule  
 ],  
 providers: [ApixuService],  
 bootstrap: [AppComponent]  
})  
export class AppModule { }

app-routing.module.ts

import { ***NgModule*** } from '@angular/core';  
import { Routes, RouterModule } from '@angular/router';  
  
  
const routes: Routes = [];  
  
@NgModule({  
 imports: [RouterModule.*forRoot*(routes)],  
 exports: [RouterModule]  
})  
export class AppRoutingModule { }

routes.ts

import { Routes } from '@angular/router';  
  
import { ForecastComponent } from './forecast/forecast.component';  
  
export const ***allAppRoutes***: Routes = [  
 { path: '', component: ForecastComponent }  
];

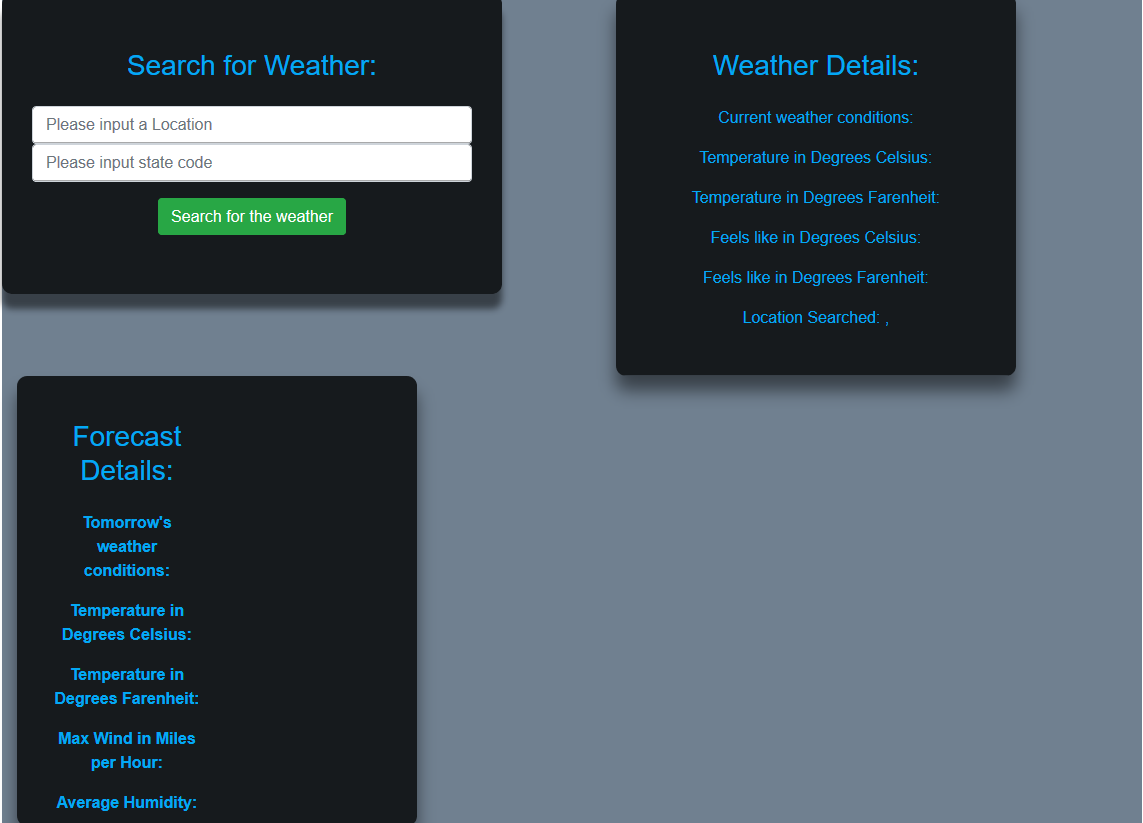
angular.json

{  
 "$schema": "./node\_modules/@angular/cli/lib/config/schema.json",  
 "version": 1,  
 "newProjectRoot": "projects",  
 "projects": {  
 "weatherapp": {  
 "projectType": "application",  
 "schematics": {},  
 "root": "",  
 "sourceRoot": "src",  
 "prefix": "app",  
 "architect": {  
 "build": {  
 "builder": "@angular-devkit/build-angular:browser",  
 "options": {  
 "outputPath": "dist/weatherapp",  
 "index": "src/index.html",  
 "main": "src/main.ts",  
 "polyfills": "src/polyfills.ts",  
 "tsConfig": "tsconfig.app.json",  
 "aot": false,  
 "assets": [  
 "src/favicon.ico",  
 "src/assets"  
 ],  
 "styles": [  
 "node\_modules/bootstrap/dist/css/bootstrap.css",  
 "src/styles.css"  
 ],  
 "scripts": [  
 "node\_modules/jquery/dist/jquery.slim.js",  
 "node\_modules/popper.js/dist/umd/popper.js",  
 "node\_modules/bootstrap/dist/js/bootstrap.js"  
 ]  
 },  
 "configurations": {  
 "production": {  
 "fileReplacements": [  
 {  
 "replace": "src/environments/environment.ts",  
 "with": "src/environments/environment.prod.ts"  
 }  
 ],  
 "optimization": true,  
 "outputHashing": "all",  
 "sourceMap": false,  
 "extractCss": true,  
 "namedChunks": false,  
 "aot": true,  
 "extractLicenses": true,  
 "vendorChunk": false,  
 "buildOptimizer": true,  
 "budgets": [  
 {  
 "type": "initial",  
 "maximumWarning": "2mb",  
 "maximumError": "5mb"  
 },  
 {  
 "type": "anyComponentStyle",  
 "maximumWarning": "6kb",  
 "maximumError": "10kb"  
 }  
 ]  
 }  
 }  
 },  
 "serve": {  
 "builder": "@angular-devkit/build-angular:dev-server",  
 "options": {  
 "browserTarget": "weatherapp:build"  
 },  
 "configurations": {  
 "production": {  
 "browserTarget": "weatherapp:build:production"  
 }  
 }  
 },  
 "extract-i18n": {  
 "builder": "@angular-devkit/build-angular:extract-i18n",  
 "options": {  
 "browserTarget": "weatherapp:build"  
 }  
 },  
 "test": {  
 "builder": "@angular-devkit/build-angular:karma",  
 "options": {  
 "main": "src/test.ts",  
 "polyfills": "src/polyfills.ts",  
 "tsConfig": "tsconfig.spec.json",  
 "karmaConfig": "karma.conf.js",  
 "assets": [  
 "src/favicon.ico",  
 "src/assets"  
 ],  
 "styles": [  
 "src/styles.css"  
 ],  
 "scripts": []  
 }  
 },  
 "lint": {  
 "builder": "@angular-devkit/build-angular:tslint",  
 "options": {  
 "tsConfig": [  
 "tsconfig.app.json",  
 "tsconfig.spec.json",  
 "e2e/tsconfig.json"  
 ],  
 "exclude": [  
 "\*\*/node\_modules/\*\*"  
 ]  
 }  
 },  
 "e2e": {  
 "builder": "@angular-devkit/build-angular:protractor",  
 "options": {  
 "protractorConfig": "e2e/protractor.conf.js",  
 "devServerTarget": "weatherapp:serve"  
 },  
 "configurations": {  
 "production": {  
 "devServerTarget": "weatherapp:serve:production"  
 }  
 }  
 }  
 }  
 }},  
 "defaultProject": "weatherapp"  
}

Forecast.

div class="container">  
 <div class="row">  
 <div class="col-md-62">  
 <h3 class="text-center my-4">Search for Weather:</h3>  
 <form [formGroup]="weatherSearchForm" (ngSubmit)="sendToAPIXU(weatherSearchForm.value)" >  
 <div class="form-group">  
 <input  
 class="form-control"  
 type="text"  
 id="weatherLocation"  
 aria-describedby="weatherLocation"  
 placeholder="Please input a Location"  
 formControlName="location" />  
 <input type="text" id="code" placeholder="please enter state code">  
  
 </div>  
 <div class="text-center">  
 <button type="submit" class="btn btn-success btn-md" onClick="f()">  
 Search for the weather  
 </button>  
 </div>  
 </form>  
</div>  
<div class="col-md-61" id="wd">  
 <h3 class="text-center my-4">Weather Details:</h3>  
 <p class="text-center">Current weather conditions: {{this.weatherData?.current.condition.text}}</p>  
 <p class="text-center">Temperature in Degrees Celsius: {{this.weatherData?.current.temp\_c}}</p>  
 <p class="text-center">Temperature in Degrees Farenheit: {{this.weatherData?.current.temp\_f}}</p>  
 <p class="text-center">Feels like in Degrees Celsius: {{this.weatherData?.current.feelslike\_c}}</p>  
 <p class="text-center">Feels like in Degrees Farenheit: {{this.weatherData?.current.feelslike\_f}}</p>  
 <p class="text-center">Location Searched: {{this.weatherData?.location.name}},  
 {{this.weatherData?.location.country}}</p>  
 </div>  
</div>  
 <div class="rows" id="fd">  
 <div class="col-md-6">  
 <h3 class="text-center my-4">Forecast Details:</h3>  
 <b> <p class="text-center">Tomorrow's weather conditions: {{this.forecastData?.forecast.forecastday[0].day.condition.text}}</p>  
 <p class="text-center">Temperature in Degrees Celsius: {{this.forecastData?.forecast.forecastday[0].day.avgtemp\_c}}</p>  
 <p class="text-center">Temperature in Degrees Farenheit: {{this.forecastData?.forecast.forecastday[0].day.avgtemp\_f}}</p>  
 <p class="text-center">Max Wind in Miles per Hour: {{this.forecastData?.forecast.forecastday[0].day.maxwind\_mph}}</p>  
 <p class="text-center">Average Humidity: {{this.forecastData?.forecast.forecastday[0].day.avghumidity}}</p>  
 </b> </div>  
 </div>  
</div>  
<style>  
 .rows  
 {  
 top: 50%;  
 left: 50%;  
 width: 500px;  
 height: 500px;  
 padding: 40px;  
 background: rgba(0,0,0,.8);  
 box-sizing: border-box;  
 box-shadow: 0 15px 25px rgba(0,0,0,.5);  
 border-radius: 10px;  
 }  
 .container  
 {  
 background-color: slategrey;  
  
 }  
 .col-md-6  
 {  
 font-family: sans-serif;  
 font-size:16px;  
 color: #03a9f4;  
 }  
 .col-md-61  
 {  
 position: relative;  
 top: 50%;  
 left: 10%;  
 width: 400px;  
 padding: 30px;  
 background: rgba(0,0,0,.8);  
 box-sizing: border-box;  
 box-shadow: 0 15px 25px rgba(0,0,0,.5);  
 border-radius: 10px;  
 font-family: sans-serif;  
 font-size:16px;  
 color: #03a9f4;  
 }  
 .col-md-62  
 {  
 top: 50%;  
 left: 10%;  
 width: 500px;  
 height: 300px;  
 padding: 30px;  
 background: rgba(0,0,0,.8);  
 box-sizing: border-box;  
 box-shadow: 0 15px 10px rgba(0,0,0,.5);  
 border-radius: 10px;  
 font-family: sans-serif;  
 font-size:16px;  
 color: #03a9f4;  
 }  
</style>

* This is how the screen look like. We have to enter city and state and click on search for the weather.



* After clicking search for the weather, the following weather details and the next day’s forecast details would be displayed.

