Module 6 – RxJS and HttpClient

Demo Document 1 – Fetch data for weather service using HttpClient and observables

edureka!

Create weather service to display weather data of different cities.

In this demo, we will see how to fetch data from weather service using HttpClient and observables

Step 1 – Now open https://openweathermap.org/api and create account to get API key, once account is created obtain API key

Step 2 – Open weather.service.ts and import HttpClient , here HttpClient will be used as **built in service**. Also import Observable and rxjs/Rx.

Observables and rxjs will be used to get response.

```
module4-demo1 > src > app > services > TS weather.service.ts > WeatherService
1   import { Injectable } from '@angular/core';
2   import { HttpClient } from '@angular/common/http';
3   import 'rxjs/Rx';
4   import { Observable } from 'rxjs';
5
6   @Injectable({
7     providedIn: 'root'
8   })
9   export class WeatherService {
10
11    constructor() { }
12  }
13
```

Step 3 – Inject the service via constructor. Now inject HttpClient as shown below

```
TS weather.model.ts       TS weather.service.ts  ●

module4-demo1 > src > app > services > TS weather.service.ts > WeatherService > [∅]constructor
  1    import { Injectable } from '@angular/core';
  2    import { HttpClient } from '@angular/common/http';
  3    import 'rxjs/Rx';
  4    import { Observable } from 'rxjs';
  5
  6    @Injectable({
  7      providedIn: 'root'
  8    })
  9    export class WeatherService {
 10      baseUrl : string;
 11      appId: string;
 12      units: string;
 13      url : string;
 14      forecast: any;
 15      constructor(private http: HttpClient) {
 16        this.baseUrl = 'http://api.openweathermap.org/data/2.5/';
 17        this.appId = 'b5ea0f92c5b8deb13ccec7d557ac6a4d';
 18        this.units = 'metric';
 19      }
 20    }
 21
```

Step 4 – Now create getWeatherForecast method to get weather response from API via observables.

```
TS weather.model.ts       TS weather.service.ts  ●

module4-demo1 > src > app > services > TS weather.service.ts > WeatherService > handleError
 15      constructor(private http: HttpClient) {
 16        this.baseUrl = 'http://api.openweathermap.org/data/2.5/';
 17        this.appId = 'b5ea0f92c5b8deb13ccec7d557ac6a4d';
 18        this.units = 'metric';
 19      }
 20
 21      getWeatherForecast(cityName: string): Observable<any> {
 22
 23        this.url = this.baseUrl + 'forecast?q=' + cityName + '&appid=' + this.appId + '&units=' + this.units  ;
 24        return this.http.get(this.url
 25        // '&units=' + this.units
 26        )
 27        //return this.forecast;
 28        //.catch(this.handleError);
 29    }
 30
 31    private handleError(error: any) {
 32      // In a real world app, we might use a remote logging infrastructure
 33      let errMsg: string;
 34      errMsg = error.message ? error.message : error.toString();
 35      console.error(errMsg);
 36      return Observable.throw(errMsg);
 37    }
```

Step 5 – Call weather service in ngOnInit method as shown below

```
ngOnInit() {
    |
    this.cityinitail = 'Bangalore'
    this._weatherService.getWeatherForecast(this.cityinitail)
        .subscribe(data => { this.weatherForecastData = data ,console.log(this.weatherForecastData)} , error => this.errorMessage

        this.onResetControls();

}
```

Step 6 – Open weather.component.html and copy html code to display weather information.

Step 7 – Open app.module.ts and import weather service and HttpClientModule as shown below

```
app.module.ts ×    <> app.component.html    TS weather.service.ts

module4-demo1 > src > app > TS app.module.ts > AppModule
 1    import { BrowserModule } from '@angular/platform-browser';
 2    import { NgModule } from '@angular/core';
 3
 4    import { AppRoutingModule } from './app-routing.module';
 5    import { AppComponent } from './app.component';
 6    import { WeatherComponent } from './weather/weather.component';
 7    import { WeatherService } from './services/weather.service';
 8    import { HttpClientModule } from '@angular/common/http';
 9
10    @NgModule({
11      declarations: [
12        AppComponent,
13        WeatherComponent
14      ],
15      imports: [
16        BrowserModule,
17        HttpClientModule,
18        AppRoutingModule
19      ],
20      providers: [WeatherService],
21      bootstrap: [AppComponent]
22    })
23    export class AppModule { }
24
```

Step 8 – Open app.component.ts and copy app-weather component as shown below

```
TS app.module.ts      <> app.component.html  ×    TS weather.component.ts

module4-demo1 > src > app > <> app.component.html > ...
    1     <!--The content below is only a placeholder and can be replaced.-->
    2     <div style="text-align:center">
    3       <h1>
    4         Welcome to {{ title }}!
    5       </h1>
    6
    7     </div>
    8     <app-weather></app-weather>
    9
   10     <router-outlet></router-outlet>
   11
```

Step 9 – Open weather.component.css and copy all css classes from the demo code.

**Step 10** – Open Index.html and copy bootstrap and other references as below.

```
TS app.module.ts    <> weather.component.html    TS weather.component.ts    <> index.html  ×    # weather.component.css    <> app.component.html    TS wea

module4-demo1 > src > <> index.html > ⊘html > ⊘head > ⊘meta
    1    <!doctype html>
    2    <html lang="en">
    3    <head>
    4        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
    5
    6        <!-- jQuery library -->
    7        <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
    8
    9        <!-- Popper JS -->
   10        <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>
   11
   12        <!-- Latest compiled JavaScript -->
   13        <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>
   14        <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.6.3/css/all.css" integrity="sha384-UHRtZLI+pbxtHCWp1t77Bi1l
   15
   16        <meta charset="utf-8">
   17        <title>Module4Demo1</title>
   18        <base href="/">
   19
```
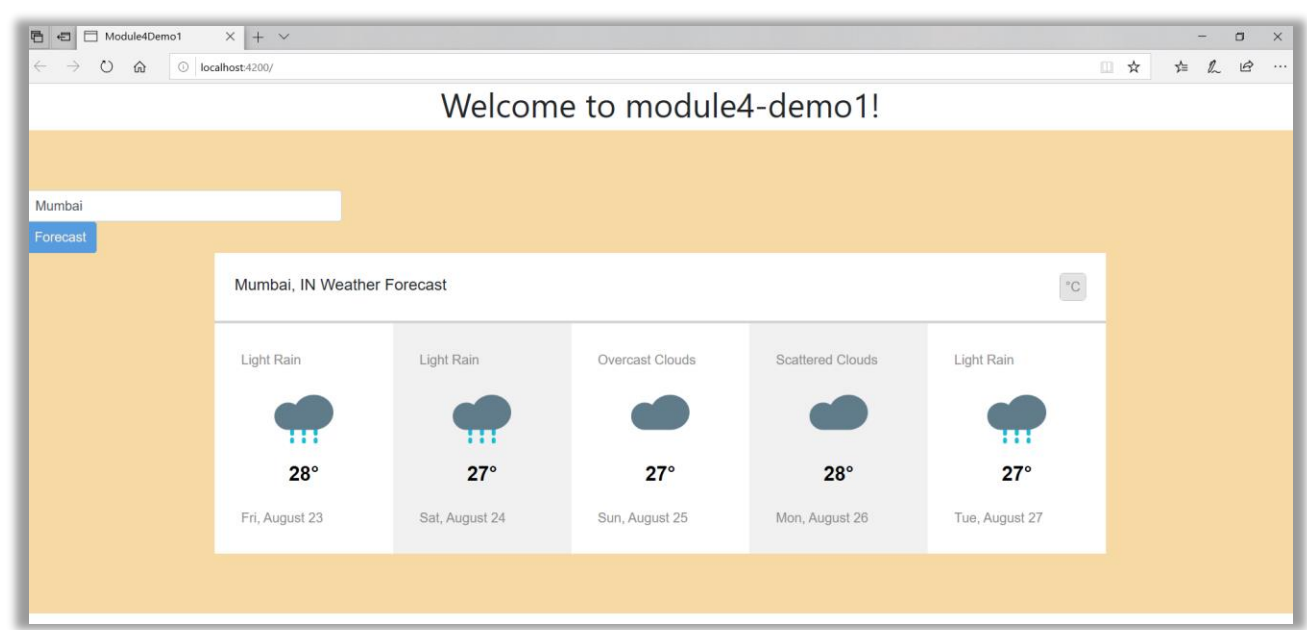
Step 11 – Copy images from images folder from demo code.
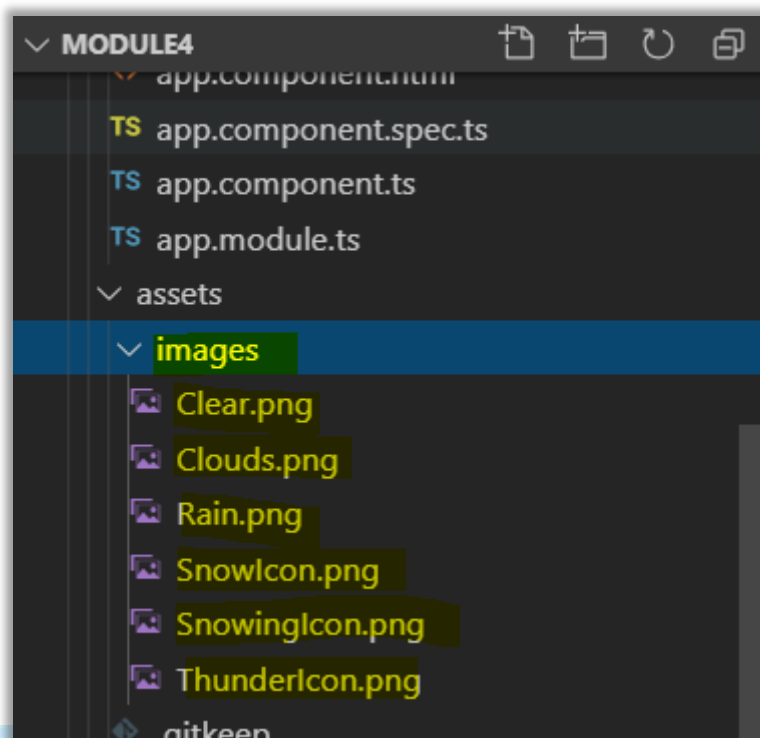
Step 12 – Run application using command 'ng serve –open'
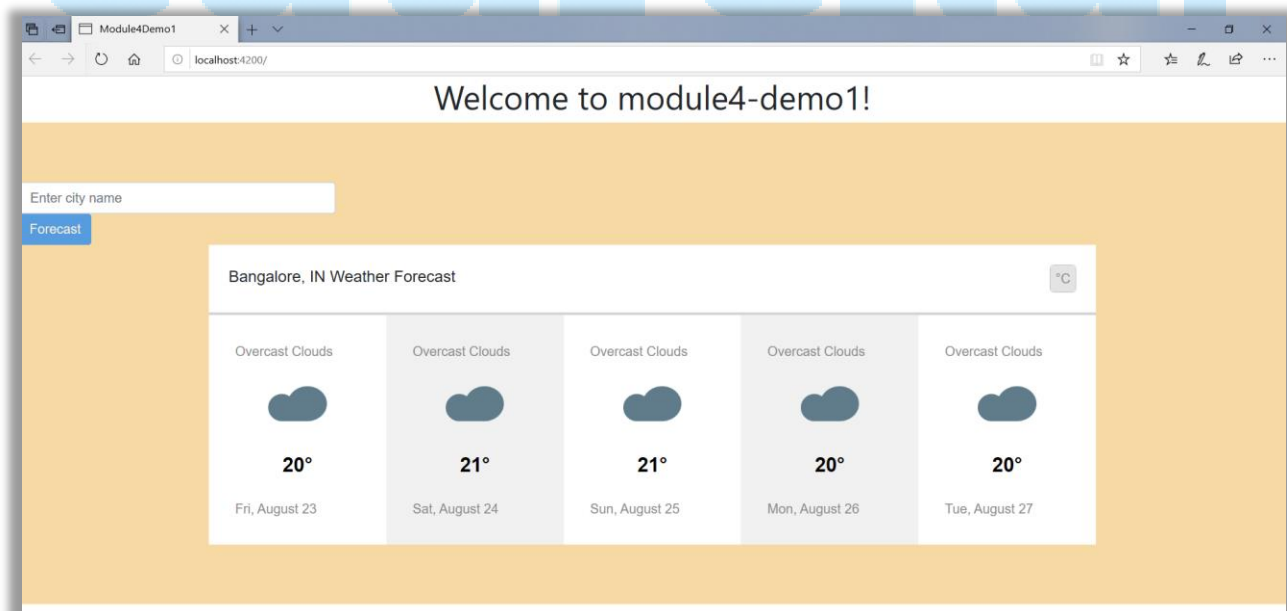
Type city name inside text box 'Enter City name' and click on Forecast button.