Once your Json server is ready then let’s integrate our api with Angular Application.

Create one app: ng new crudapp

Move: cd crudapp

Open the same in VS code

Run: ng serve

First create component: ng g c list

Create service: ng g s student

Let’s create Student Model: right click on app and create new file student.ts

export class Student{

    id:number;

    name:string;

    email:string;

    country:string;

}

To remove the initialization error edit tsconfig.json file

{

  "compileOnSave": false,

  "compilerOptions": {

    "strictPropertyInitialization": false,

    "baseUrl": "./",

    "outDir": "./dist/out-tsc",

    "forceConsistentCasingInFileNames": true,

Save the same.

Include HttpClientModule in your app.module.ts

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

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

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

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

import { ListComponent } from './list/list.component';

@NgModule({

  declarations: [

    AppComponent,

    ListComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    HttpClientModule

  ],

Let’s edit service for all methods

Student.service.ts

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

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

import { Observable } from 'rxjs';

import { Student } from './student';

@Injectable({

  providedIn: 'root'

})

export class StudentService {

  baseUrl:string= "http://localhost:3000/students";

  constructor(private http: HttpClient) { }

  getAllStudents():Observable<Student[]>{

    return this.http.get<Student[]>(this.baseUrl);

  }

  getById(id:number):Observable<Student>{

    return this.http.get<Student>(this.baseUrl+"/"+id)

  }

  addStudent(obj:Student){

    return this.http.post(this.baseUrl,obj);

  }

  updateStudent(obj:Student,id:number){

    return this.http.put(this.baseUrl+"/"+id,obj)

  }

  deleteStudent(id:number){

    return this.http.delete(this.baseUrl+"/"+id)

  }

}

Once you are ready with service let’s inject service in our component to use.

Open list.component.ts

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

import { Student } from '../student';

import { StudentService } from '../student.service';

@Component({

  selector: 'app-list',

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

  styleUrls: ['./list.component.css']

})

export class ListComponent {

  students:Student[]

  constructor(private service: StudentService){}

  ngOnInit(){

    this.service.getAllStudents().subscribe(data=>this.students=data);

  }

}

Let’s Display this data on html page- list.component.html

<div>

    <table class="table table-bordered table-striped">

        <thead>

            <tr>

                <th>Id</th><th>Name</th><th>Email</th><th>Country</th>

            </tr>

        </thead>

        <tbody>

            <tr  \*ngFor="let s of students">

                <td>{{s.id}}</td>

                <td>{{s.name}}</td>

                <td>{{s.email}}</td>

                <td>{{s.country}}</td>

            </tr>

        </tbody>

    </table>

</div>

Let’s create Routing to display List on Html Page

App-routing.module.ts

const routes: Routes = [

  {path:'list',component:ListComponent}

];

Edit app.component.html fil for creating routerLinks

<div class="container">

  <h1>Welcome to the Simplilearn</h1>

  <ul class="nav bg-warning">

    <li class="nav-item">

      <a routerLink="list" class="nav-link">Student List</a>

    </li>

  </ul>

  <router-outlet></router-outlet>

</div>

Save All files and check you can see the data in your fronend.

Update the list html file by adding view button

<<div>

    <table class="table table-bordered table-striped">

        <thead>

            <tr>

                <th>Id</th><th>Name</th><th>Email</th><th>Country</th>

                <th>Operations</th>

            </tr>

        </thead>

        <tbody>

            <tr  \*ngFor="let s of students">

                <td>{{s.id}}</td>

                <td>{{s.name}}</td>

                <td>{{s.email}}</td>

                <td>{{s.country}}</td>

                <td><a routerLink='/details/{{s.id}}'>

                    <button class="btn btn-success">view</button>

                </a></td>

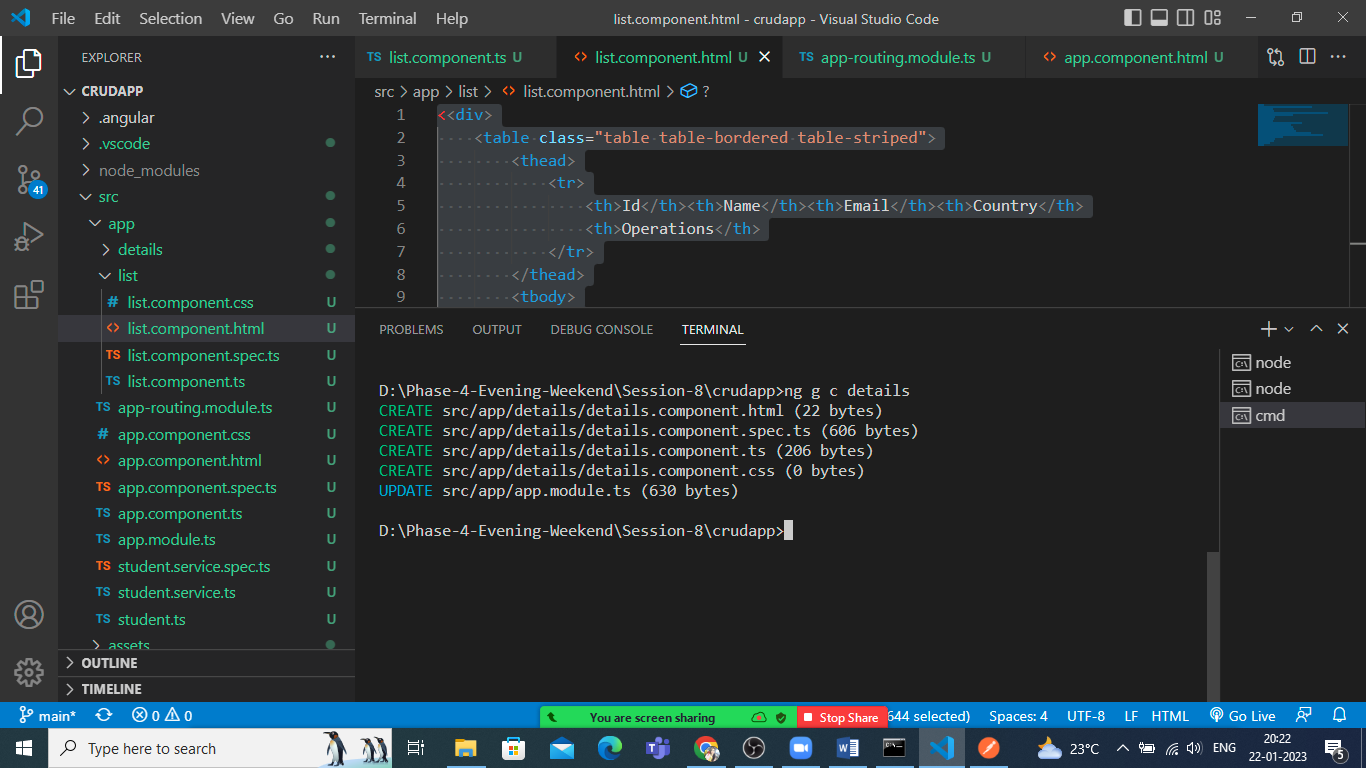
            </tr>

        </tbody>

    </table>

</div>

Now create Details component and its routing.



Update routing file

const routes: Routes = [

  {path:'list',component:ListComponent},

  {path:'details/:id',component:DetailsComponent}

];

Save the files and now let’s edit details.component.ts file

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

import { ActivatedRoute } from '@angular/router';

import { Student } from '../student';

import { StudentService } from '../student.service';

@Component({

  selector: 'app-details',

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

  styleUrls: ['./details.component.css']

})

export class DetailsComponent {

  obj:Student;

  constructor(private service:StudentService, private route:ActivatedRoute){}

  ngOnInit(){

    const id= Number(this.route.snapshot.paramMap.get('id'));

    this.service.getById(id).subscribe(data=> this.obj=data)

  }

}

Let’s display that student object in details.component.html

<div>

    <h1>Student details Page</h1>

    <h3>Welcome {{obj.name}}</h3>

    <p>

        Followings are your details:

    </p>

    <ul class="list-group">

        <li class="list-group-item">Id:{{obj.id}}</li>

        <li class="list-group-item">Email:{{obj.email}}</li>

        <li class="list-group-item">Country:{{obj.country}}</li>

    </ul>

</div>

Let’s implement delete functionality.

Open list.component.html

<td><a routerLink='/details/{{s.id}}'>

                    <button class="btn btn-success">view</button>

                </a> &nbsp; &nbsp;

                <button class="btn btn-danger" (click)="ondelete(s.id)">delete</button>

            </td>

Create ondelete method in list.component.ts file

ondelete(id:number){

    this.service.deleteStudent(id).subscribe(x=>console.log(x));

    this.students= this.students.filter((x)=> x.id !== id)

  }

Save all and check you will be able to delete your data as well.

Let’s add new Data.

Create new compo: ng g c addstu

Open app module and import FormsModule

imports: [

    BrowserModule,

    AppRoutingModule,

    HttpClientModule,

    FormsModule

  ],

Edit addstu.com.ts

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

import { Student } from '../student';

import { StudentService } from '../student.service';

@Component({

  selector: 'app-addstu',

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

  styleUrls: ['./addstu.component.css']

})

export class AddstuComponent {

  student:Student= new Student();

  constructor(private service:StudentService){}

  onSubmit(){

      this.service.addStudent(this.student).subscribe(x=>console.log(x));

      alert("Data Submitted Successfully")

  }

}

Create form in addstu.com.html file

<div class="row">

    <div class="col-md-6 offset-3">

        <form (ngSubmit)="onSubmit()">

            <div class="form-group">

                <label>Id</label>

                <input type="number" class="form-control"

                name="id" [(ngModel)]="student.id" placeholder="Id" required>

            </div>

            <div class="form-group">

                <label>Name</label>

                <input type="text" class="form-control"

                name="name" [(ngModel)]="student.name" placeholder="Name" required>

            </div>

            <div class="form-group">

                <label>Email</label>

                <input type="email" class="form-control"

                name="email" [(ngModel)]="student.email" placeholder="Email" required>

            </div>

            <div class="form-group">

                <label>Country</label>

                <input type="text" class="form-control"

                name="country" [(ngModel)]="student.country" placeholder="Country" required>

            </div>

            <button class="btn btn-primary">Register</button>

        </form>

    </div>

Add this newly created com in routing

const routes: Routes = [

  {path:'list',component:ListComponent},

  {path:'details/:id',component:DetailsComponent},

  {path:'add',component:AddstuComponent}

];

Add link in app.comp.html

<div class="container">

  <h1>Welcome to the Simplilearn</h1>

  <ul class="nav bg-warning">

    <li class="nav-item">

      <a routerLink="list" class="nav-link">Student List</a>

    </li>

    <li class="nav-item">

      <a routerLink="add" class="nav-link">Student Registration</a>

    </li>

  </ul>

  <router-outlet></router-outlet>

</div>

Save and check the data insertion.

After submission if you want to redirect user to list page then edit addstu.com.ts file as below

  constructor(private service:StudentService, private router:Router){}

  onSubmit(){

      this.service.addStudent(this.student).subscribe(x=>console.log(x));

      alert("Data Submitted Successfully")

      this.router.navigate(['list'])

  }