Student API – CRUD Operations with ASP.NET Web API

Agenda

- Angular Application having a Home Page and a Navigation Bar
- Install bootstrap
- Implementing Routes and link to CRUD based Links
- Components
  - **studentList**
  - **addStudent**
  - **studentDetails**
  - **deleteStudent**
- Student List Component
- Add New Student Component
- Student Details Component
- Delete a Student Component
- ASP.NET Core Web API
  - Create a student table in SQL Server
  - Create Web API using ASP.NET Core and entity framework
- Test Web API
- Final Application Execution

Angular Application having a Home Page and a Navigation Bar

- In my D drive I created a folder named NGLearn. And now go to terminal/ command prompt and use NG CLI command to create angular application.
- Create a basic Angular Application → studentCrudDemo

> **> ng new studentCrudDemo**

```
npm                          ×   +   ∨

Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.

D:\NGLearn>ng new studentCrudDemo
? Which stylesheet format would you like to use? (Use arrow keys)
> CSS              [ https://developer.mozilla.org/docs/Web/CSS        ]
  Sass (SCSS)      [ https://sass-lang.com/documentation/syntax#scss   ]
  Sass (Indented)  [ https://sass-lang.com/documentation/syntax#the-indented-syntax ]
  Less             [ http://lesscss.org                                ]
```

```
npm                          ×   +   ∨

Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.

D:\NGLearn>ng new studentCrudDemo
? Which stylesheet format would you like to use? CSS      [ https://developer.mozilla.org/docs/Web/CSS
                 ]
? Do you want to enable Server-Side Rendering (SSR) and Static Site Generation (SSG/Prerendering)? (y/N) N
```

```
C:\WINDOWS\system32\cmd.    ×   +   ∨

D:\NGLearn>ng new studentCrudDemo
? Which stylesheet format would you like to use? CSS      [ https://developer.mozilla.org/docs/Web/CSS
                 ]
? Do you want to enable Server-Side Rendering (SSR) and Static Site Generation (SSG/Prerendering)? no
CREATE studentCrudDemo/angular.json (2724 bytes)
CREATE studentCrudDemo/package.json (1087 bytes)
CREATE studentCrudDemo/README.md (1103 bytes)
CREATE studentCrudDemo/tsconfig.json (1045 bytes)
CREATE studentCrudDemo/.editorconfig (290 bytes)
CREATE studentCrudDemo/.gitignore (629 bytes)
CREATE studentCrudDemo/tsconfig.app.json (439 bytes)
CREATE studentCrudDemo/tsconfig.spec.json (449 bytes)
CREATE studentCrudDemo/.vscode/extensions.json (134 bytes)
CREATE studentCrudDemo/.vscode/launch.json (490 bytes)
CREATE studentCrudDemo/.vscode/tasks.json (980 bytes)
CREATE studentCrudDemo/src/main.ts (256 bytes)
CREATE studentCrudDemo/src/index.html (314 bytes)
CREATE studentCrudDemo/src/styles.css (81 bytes)
CREATE studentCrudDemo/src/app/app.component.html (20239 bytes)
CREATE studentCrudDemo/src/app/app.component.spec.ts (972 bytes)
CREATE studentCrudDemo/src/app/app.component.ts (324 bytes)
CREATE studentCrudDemo/src/app/app.component.css (0 bytes)
CREATE studentCrudDemo/src/app/app.config.ts (318 bytes)
CREATE studentCrudDemo/src/app/app.routes.ts (80 bytes)
CREATE studentCrudDemo/public/favicon.ico (15086 bytes)
√ Packages installed successfully.
    Successfully initialized git.

D:\NGLearn>
```

- Exit the CMD and open folder in Visual Studio Code
- Open a terminal window
- Install bootstrap

```
npm i bootstrap --save
```

- Add to angular.json file

```
"styles": [

          "node_modules/bootstrap/dist/css/bootstrap.css",

          "src/styles.css"

        ],
"scripts": ["node_modules/bootstrap/dist/js/bootstrap.js"]
```

- I want that an entire application background should be having a same background color, so added a style in D:\NGLearn\studentCrudDemo\src\styles.css

```css
body {
  background-color: hsl(210, 100%, 85%);
}
```

- Add pages folder and create following components in pages folder
    - PS D:\NGLearn\studentCrudDemo\src\app\pages> **ng g c studentList**
    - PS D:\NGLearn\studentCrudDemo\src\app\pages> **ng g c addStudent**
    - PS D:\NGLearn\studentCrudDemo\src\app\pages> **ng g c studentDetails**
    - PS D:\NGLearn\studentCrudDemo\src\app\pages> **ng g c deleteStudent**
- Add routing to application
- In app.component.ts

```ts
import { Component } from '@angular/core';
import { RouterLink, RouterOutlet } from '@angular/router';

@Component({
  selector: 'app-root',
  standalone: true,
  imports: [RouterOutlet, RouterLink],
  templateUrl: './app.component.html',
  styleUrl: './app.component.css'
})
export class AppComponent {
  title = 'routedemo';
}
```

- Go to app.routes.ts

```ts
import { Routes } from '@angular/router';
import { StudentListComponent } from './pages/student-list/student-list.component';
import { AddStudentComponent } from './pages/add-student/add-student.component';
import { StudentDetailsComponent } from './pages/student-details/student-details.component';
import { DeleteStudentComponent } from './pages/delete-student/delete-student.component';

export const routes: Routes = [
    {
        path: 'student-list',
        component: StudentListComponent
    },
    {
        path: 'add-student',
        component: AddStudentComponent
    },
    {
        path: 'student-details',
        component: StudentDetailsComponent
    }, {
        path: 'delete-student',
        component: DeleteStudentComponent
    },
];
```

- Add a bootstrap sample navbar

```html
<nav class="navbar navbar-expand-sm navbar-dark bg-dark">
  <div class="container-fluid">
    <a class="navbar-brand" href="javascript:void(0)">Students</a>
    <button
      class="navbar-toggler"
      type="button"
```

```html
        data-bs-toggle="collapse"
        data-bs-target="#mynavbar"
    >
        <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="mynavbar">
        <ul class="navbar-nav me-auto">
          <li class="nav-item">
            <a class="nav-link" routerLink="student-list">List</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" routerLink="add-student">AddNew</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" routerLink="student-details">Details</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" routerLink="delete-student">Delete</a>
          </li>
        </ul>
        <form class="d-flex">
          <input class="form-control me-2" type="text" placeholder="Search" />
          <button class="btn btn-primary" type="button">Search</button>
        </form>
      </div>
  </div>
</nav>
<router-outlet />
```
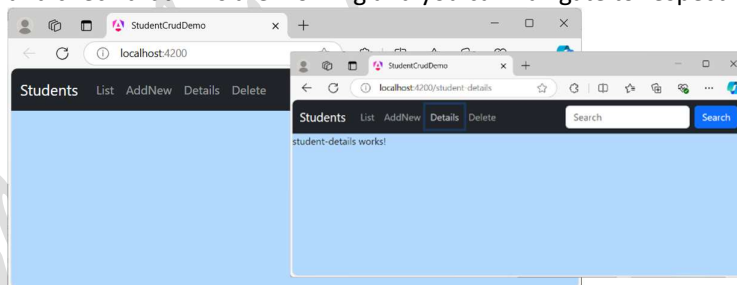
- Now run application and check that links are working and you can navigate to respective contents

Student List Component – Get all Student Records from API

- Here we need to get data from student service.
- Add service folder and add student service in it.

ng g s student

- We need to use HttpClient so in app.config.ts add provider

```typescript
import { ApplicationConfig, provideZoneChangeDetection } from '@angular/core';
import { provideRouter } from '@angular/router';

import { routes } from './app.routes';
import { provideHttpClient } from '@angular/common/http';

export const appConfig: ApplicationConfig = {
  providers: [provideHttpClient(), provideZoneChangeDetection({
eventCoalescing: true }), provideRouter(routes)]
};
```
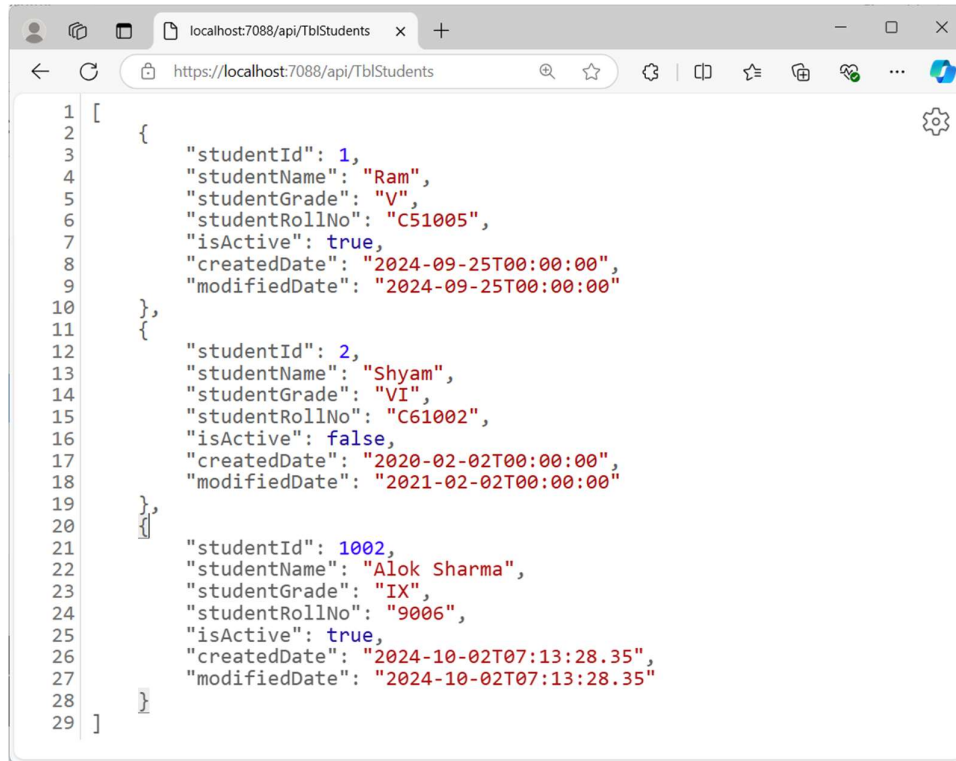
- Now create a student service to get student data from API. This API we have generated using ASP.NET Core Web API with SQL Server and Entity Framework. Documentation is available separately check StudentAPI git folder for the same.
- API url is https://localhost:7088/api/TblStudents which return JSON as


```typescript
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';

@Injectable({
  providedIn: 'root'
})
export class StudentService {

  constructor(private http: HttpClient) { }

  getStudents() {
    return this.http.get("https://localhost:7088/api/TblStudents");
  }
}
```

- Consume data obtained from API
- Go to student-list-component.ts and modify:

```typescript
import { Component, inject, OnInit } from '@angular/core';
import { StudentService } from '../../service/student.service';
@Component({
  selector: 'app-student-list',
  standalone: true,
  imports: [],
  templateUrl: './student-list.component.html',
```
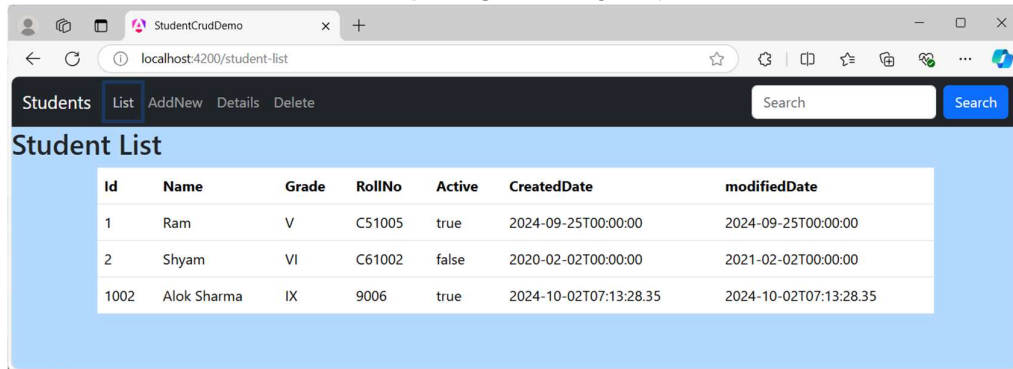
```
  styleUrl: './student-list.component.css'
})
export class StudentListComponent implements OnInit {
  studentService = inject(StudentService)
  studentList: any[] = [];
  ngOnInit(): void {
    this.loadStudents();
  }
  loadStudents() {
    this.studentService.getStudents().subscribe((res: any) => {
      this.studentList = res;
    })
  }
}
```

- Now the response obtained from API is collected into studentList array.
- Display studentList array data in HTML file of student-list component.
- Go to student-list.component.html and design as:

```html
<h2>Student List</h2>
<div class="container">
  <div class="row">
    <div class="col-12">
      <table class="table table-borderd">
        <thead>
          <tr>
            <th>Id</th>
            <th>Name</th>
            <th>Grade</th>
            <th>RollNo</th>
            <th>Active</th>
            <th>CreatedDate</th>
            <th>modifiedDate</th>
          </tr>
        </thead>
        <tbody>
          @for (item of studentList; track $index) {
          <tr>
            <td>{{ item.studentId }}</td>
            <td>{{ item.studentName }}</td>
            <td>{{ item.studentGrade }}</td>
            <td>{{ item.studentRollNo }}</td>
            <td>{{ item.isActive }}</td>
            <td>{{ item.createdDate }}</td>
            <td>{{ item.modifiedDate }}</td>
          </tr>
          }
        </tbody>
      </table>
    </div>
  </div>
</div>
```
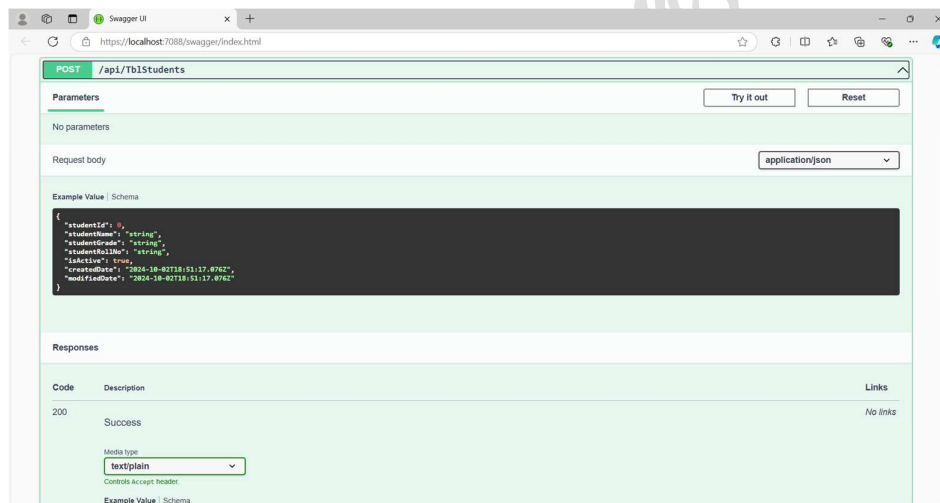
- Test it in browser:

Add-Student Component – Add New Student Records using API

- To add a new student record, we need to use a data entry form and need to post student object data to Web API. Api's post method make arrangement to store data in SQL Server database table. (Your API should be CORS enabled.)
- As in my API following is the POST section

- Here if you try is out then you are allowed to add data by defining data items in JSON object

```
{
    "studentId": 0,
    "studentName": "string",
    "studentGrade": "string",
    "studentRollNo": "string",
    "isActive": true,
    "createdDate": "2024-10-02T18:51:17.076Z",
    "modifiedDate": "2024-10-02T18:51:17.076Z"
}
```

- Same we will be using in our Angular component to send object with data in add-student component.
- First create a Template Form to Add New Student

- While creating form we will not send StudentId as it is defined a primary key as well identity column. Autogenerated ID will be assigned to newly added record.
- In HTML below I commented the div for studentId as we are not submitting it.
- Go to add-student.component.html and design form as:

```html
<h2>Add New Student</h2>
<div class="form-container">
  <div class="form-header text-center">
    <h2>Student Form</h2>
  </div>
  <form id="studentForm">
    <!-- <div class="form-group">
      <label for="studentId">Student ID:</label>
      <input type="number" class="form-control" id="studentId" name="studentId"
        Required />
    </div> -->

    <div class="form-group">
      <label for="studentName">Student Name:</label>
      <input
        type="text"
        [(ngModel)]="studentObj.studentName"
        class="form-control"
        id="studentName"
        name="studentName"
        required
      />
    </div>

    <div class="form-group">
      <label for="studentGrade">Student Grade:</label>
      <input
        type="text"
        [(ngModel)]="studentObj.studentGrade"
        class="form-control"
        id="studentGrade"
        name="studentGrade"
        required
      />
    </div>

    <div class="form-group">
      <label for="studentRollNo">Student Roll No:</label>
      <input
        type="text"
        [(ngModel)]="studentObj.studentRollNo"
        class="form-control"
        id="studentRollNo"
        name="studentRollNo"
        required
      />
    </div>

    <div class="form-group form-check">
      <input
        type="checkbox"
        [(ngModel)]="studentObj.isActive"
        class="form-check-input"
```

```html
        id="isActive"
        name="isActive"
        checked
      />
      <label class="form-check-label" for="isActive">Is Active</label>
    </div>

    <div class="form-group">
      <label for="createdDate">Created Date:</label>
      <input
        type="date"
        [(ngModel)]="studentObj.createdDate"
        class="form-control"
        id="createdDate"
        name="createdDate"
        required
      />
    </div>

    <div class="form-group">
      <label for="modifiedDate">Modified Date:</label>
      <input
        type="date"
        [(ngModel)]="studentObj.modifiedDate"
        class="form-control"
        id="modifiedDate"
        name="modifiedDate"
      />
    </div>

    <button
      type="submit"
      class="btn btn-primary btn-block"
      (click)="onSubmit()"
    >
      Submit
    </button>
  </form>
</div>
```
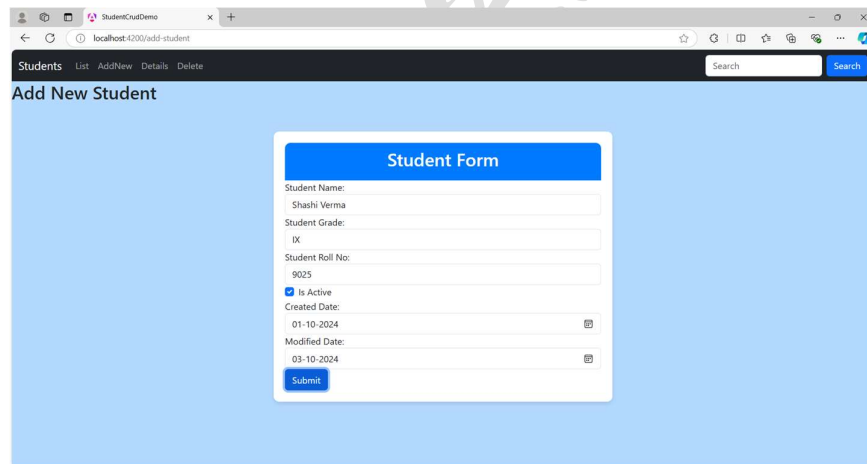
- Go to add-student.component.ts and add the object in class
- Inject HttpClient
- Create onSubmit()

```typescript
import { HttpClient } from '@angular/common/http';
import { Component, inject } from '@angular/core';
import { FormsModule } from '@angular/forms';

@Component({
  selector: 'app-add-student',
  standalone: true,
  imports: [FormsModule],
  templateUrl: './add-student.component.html',
  styleUrl: './add-student.component.css'
})
export class AddStudentComponent {
```
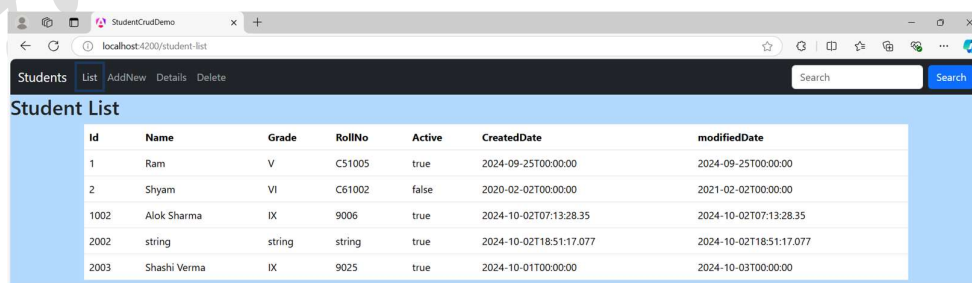
```
studentObj: any = {
  "studentId": 0,
  "studentName": "",
  "studentGrade": "",
  "studentRollNo": "",
  "isActive": true,
  "createdDate": "",
  "modifiedDate": ""
}
http = inject(HttpClient);
onSubmit() {
  debugger;
  this.http.post("https://localhost:7088/api/TblStudents",
this.studentObj).subscribe((res: any) => {
    debugger;
    if (res.studentId >= 0)
      alert("Student Record Created!");
    else {
      alert("Some Problem in Student Creation")
    }
  })
}
}
```

Run the app and go to AddNew link and fill record and submit as:

- Check by clicking student-list

<div align="center">

❀✠End of Chapter✠❀

</div>