Milestone-Assessment-5(reattempt)

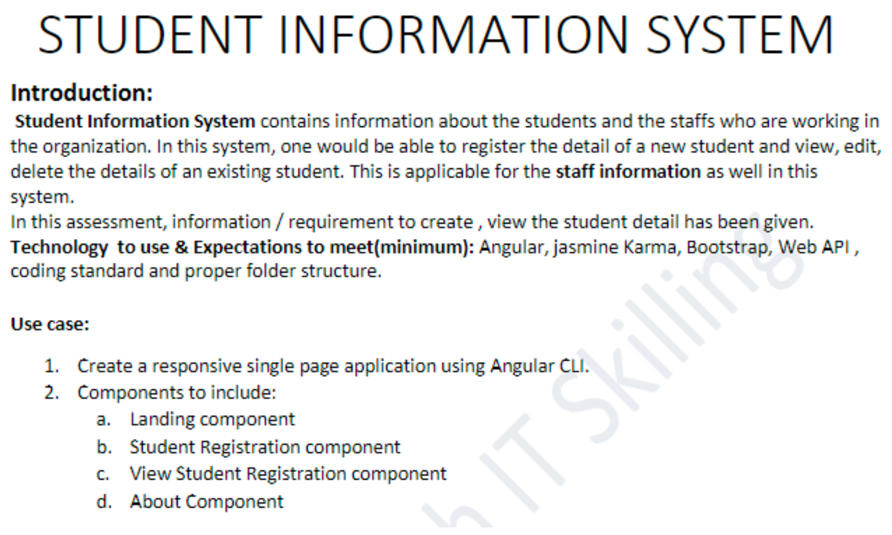
Webapi with Angular

Name: B Karthik

MID: M1082972

Source code:
https://github.com/BKarthik1/BKarthik1.git

Question

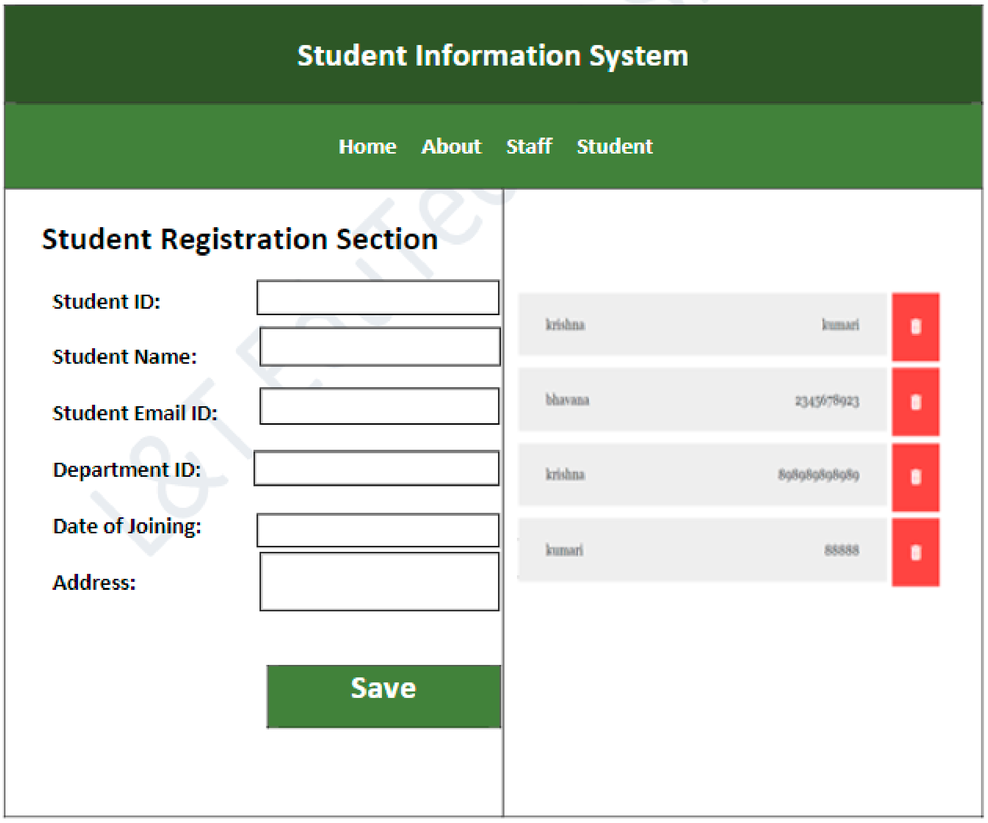


STUDENT INFORMATION SYSTEM

Introduction:
Student Information System contains information about the students and the staffs who are working in the organization. In this system, one would be able to register the detail of a new student and view, edit, delete the details of an existing student. This is applicable for the **staff information** as well in this system.
In this assessment, information / requirement to create , view the student detail has been given.
Technology to use & Expectations to meet(minimum): Angular, jasmine Karma, Bootstrap, Web API , coding standard and proper folder structure.

Use case:

1. Create a responsive single page application using Angular CLI.
2. Components to include:
   a. Landing component
   b. Student Registration component
   c. View Student Registration component
   d. About Component



Student Information System

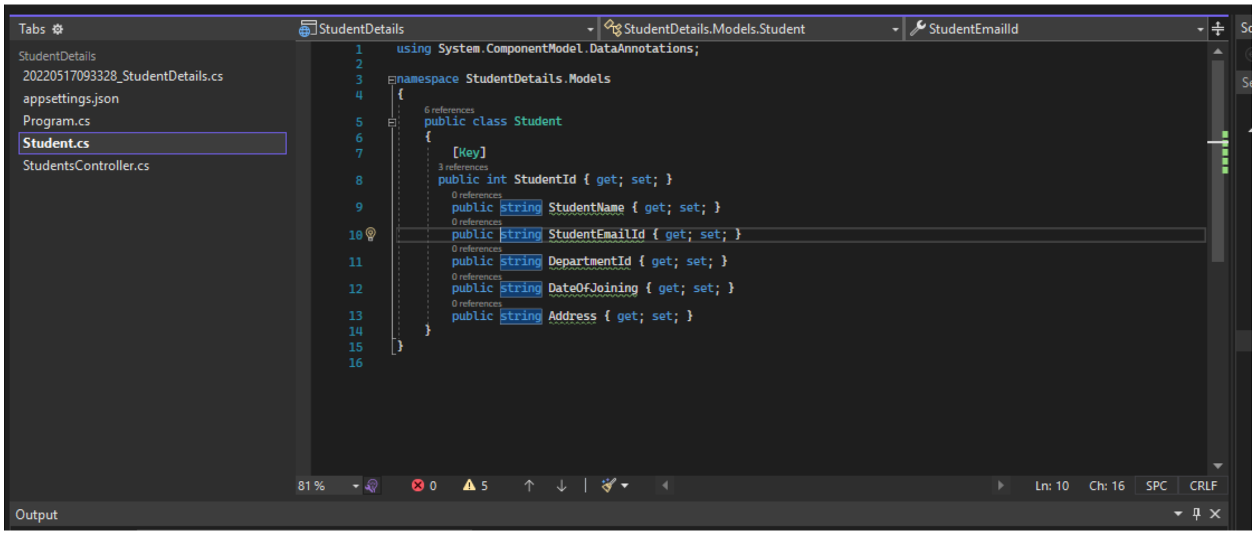
Student Registration Section

Student ID:	
Student Name:	
Student Email ID:	
Department ID:	
Date of Joining:	
Address:	

krishna	kumari
bhavana	2345678923
krishna	8989898989
kumari	88888

First we have create Web Api Part

First model

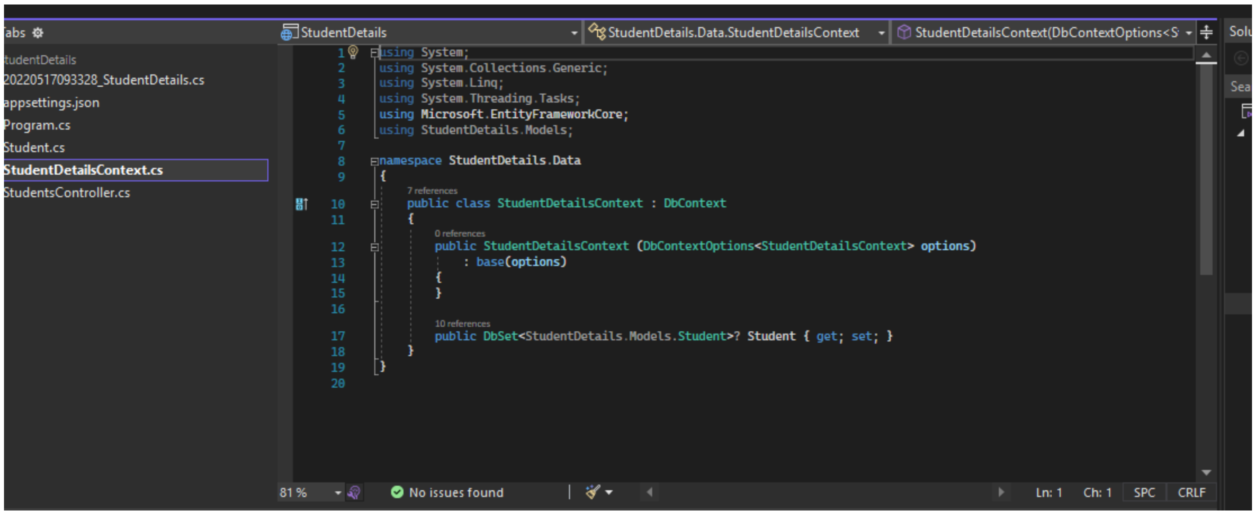


```
using System.ComponentModel.DataAnnotations;

namespace StudentDetails.Models
{
    public class Student
    {
        [Key]
        public int StudentId { get; set; }
          public string StudentName { get; set; }
          public string StudentEmailId { get; set; }
          public string DepartmentId { get; set; }
          public string DateOfJoining { get; set; }
          public string Address { get; set; }
    }
}
```

Model Part code

```
using System.ComponentModel.DataAnnotations;

namespace StudentDetails.Models
{
    public class Student
    {
        [Key]
      public int StudentId { get; set; }
        public string StudentName { get; set; }
        public string StudentEmailId { get; set; }
        public string DepartmentId { get; set; }
        public string DateOfJoining { get; set; }
        public string Address { get; set; }
    }
}
```

Next Create Data folder



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.EntityFrameworkCore;
using StudentDetails.Models;

namespace StudentDetails.Data
{
    7 references
    public class StudentDetailsContext : DbContext
    {
        0 references
        public StudentDetailsContext (DbContextOptions<StudentDetailsContext> options)
            : base(options)
        {
        }

        10 references
        public DbSet<StudentDetails.Models.Student>? Student { get; set; }
    }
}
```

Code

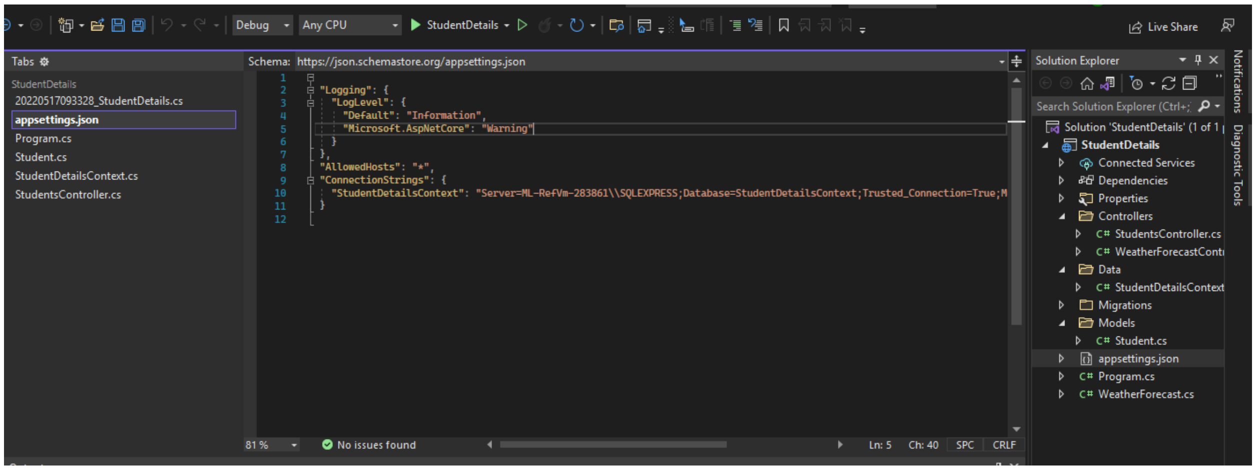
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.EntityFrameworkCore;
using StudentDetails.Models;

namespace StudentDetails.Data
{
    public class StudentDetailsContext : DbContext
    {
        public StudentDetailsContext (DbContextOptions<StudentDetailsContext>
options)
            : base(options)
        {
        }

        public DbSet<StudentDetails.Models.Student>? Student { get; set; }
    }
}
```

Next we have to create Controller

 Code

```
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using StudentDetails.Data;
using StudentDetails.Models;

namespace StudentDetails.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class StudentsController : ControllerBase
    {
        private readonly StudentDetailsContext _context;

        public StudentsController(StudentDetailsContext context)
        {
            _context = context;
        }

        // GET: api/Students
        [HttpGet]
        public async Task<ActionResult<IEnumerable<Student>>> GetStudent()
        {
          if (_context.Student == null)
          {
              return NotFound();
          }
            return await _context.Student.ToListAsync();
        }

        // GET: api/Students/5
        [HttpGet("{id}")]
        public async Task<ActionResult<Student>> GetStudent(int id)
        {
          if (_context.Student == null)
          {
              return NotFound();
          }
            var student = await _context.Student.FindAsync(id);

            if (student == null)
            {
                return NotFound();
            }

            return student;
        }

        // PUT: api/Students/5
        // To protect from overposting attacks, see
https://go.microsoft.com/fwlink/?linkid=2123754
        [HttpPut("{id}")]
        public async Task<IActionResult> PutStudent(int id, Student student)
        {
            if (id != student.StudentId)
            {
                return BadRequest();
            }

            _context.Entry(student).State = EntityState.Modified;

            try
            {
                await _context.SaveChangesAsync();
            }
            catch (DbUpdateConcurrencyException)
            {
                if (!StudentExists(id))
                {
                    return NotFound();
                }
                else
                {
                    throw;
                }
            }

            return NoContent();
        }

        // POST: api/Students
        // To protect from overposting attacks, see
https://go.microsoft.com/fwlink/?linkid=2123754
        [HttpPost]
        public async Task<ActionResult<Student>> PostStudent(Student student)
        {
          if (_context.Student == null)
          {
              return Problem("Entity set 'StudentDetailsContext.Student'  is
null.");
          }
            _context.Student.Add(student);
            await _context.SaveChangesAsync();

            return CreatedAtAction("GetStudent", new { id = student.StudentId },
student);
        }

        // DELETE: api/Students/5
        [HttpDelete("{id}")]
        public async Task<IActionResult> DeleteStudent(int id)
        {
            if (_context.Student == null)
            {
                return NotFound();
            }
            var student = await _context.Student.FindAsync(id);
            if (student == null)
            {
                return NotFound();
            }

            _context.Student.Remove(student);
            await _context.SaveChangesAsync();

            return NoContent();
        }

        private bool StudentExists(int id)
        {
            return (_context.Student?.Any(e => e.StudentId ==
id)).GetValueOrDefault();
        }
    }
}
```

Next Connection to database

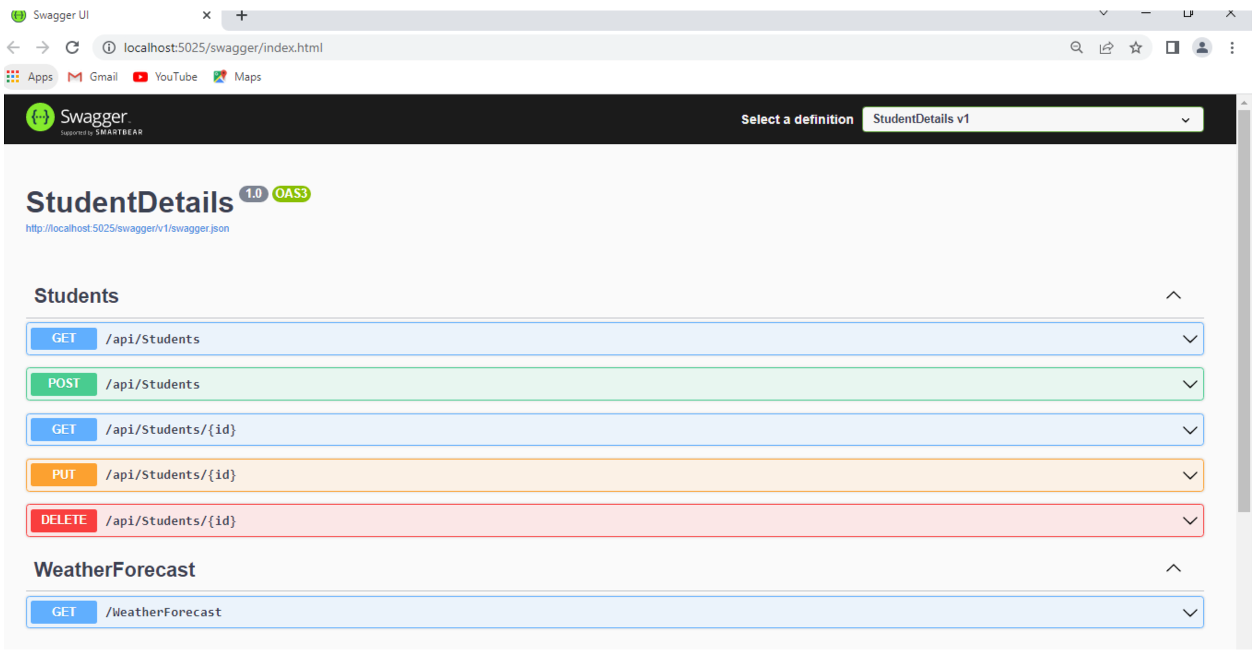


```

"Logging": {
  "LogLevel": {
    "Default": "Information",
    "Microsoft.AspNetCore": "Warning"
  }
},
"AllowedHosts": "*",
"ConnectionStrings": {
  "StudentDetailsContext": "Server=ML-RefVm-283861\\SQLEXPRESS;Database=StudentDetailsContext;Trusted_Connection=True;M
}

```

Next add-migration

Update Migration

After doing this we get



StudentDetails 1.0 OAS3

http://localhost:5025/swagger/v1/swagger.json

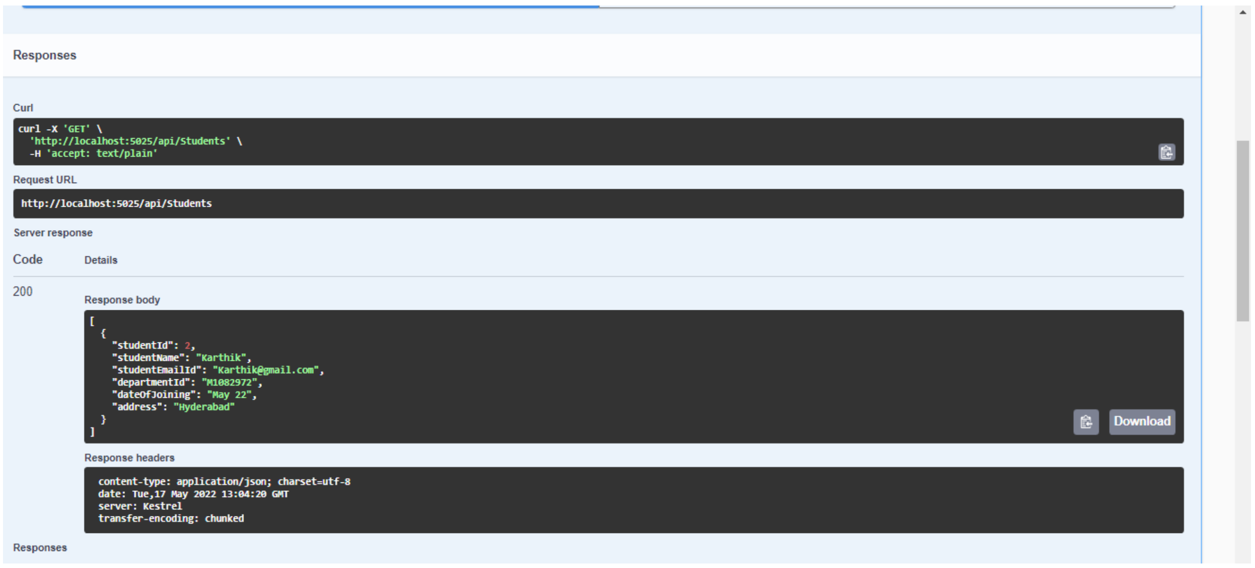
Students

```
GET     /api/Students
POST    /api/Students
GET     /api/Students/{id}
PUT     /api/Students/{id}
DELETE  /api/Students/{id}
```

WeatherForecast

```
GET     /WeatherForecast
```

It running



Responses

Curl

```
curl -X 'GET' \
  'http://localhost:5025/api/Students' \
  -H 'accept: text/plain'
```

Request URL

```
http://localhost:5025/api/Students
```

Server response

Code Details

200

Response body

```
[
  {
    "studentId": 2,
    "studentName": "Karthik",
    "studentEmailId": "Karthik@gmail.com",
    "departmentId": "M1082972",
    "dateOfJoining": "May 22",
    "address": "Hyderabad"
  }
]
```

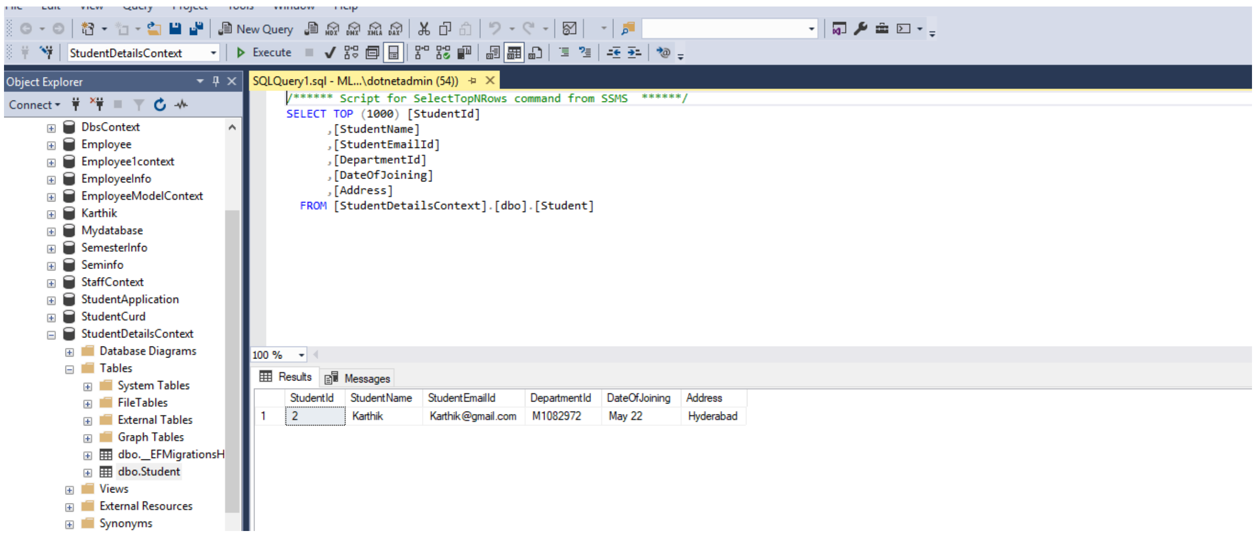
Response headers

```
content-type: application/json; charset=utf-8
date: Tue,17 May 2022 13:04:20 GMT
server: Kestrel
transfer-encoding: chunked
```

Responses

All get, post , get , put , Delete all are running

Let check in Database



```sql
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP (1000) [StudentId]
      ,[StudentName]
      ,[StudentEmailId]
      ,[DepartmentId]
      ,[DateOfJoining]
      ,[Address]
  FROM [StudentDetailsContext].[dbo].[Student]
```

	StudentId	StudentName	StudentEmailId	DepartmentId	DateOfJoining	Address
1	2	Karthik	Karthik@gmail.com	M1082972	May 22	Hyderabad

See the data was inserted

So we done with Api

Next we have do Angular

Let open cmd

And give commod

ng new Student

next

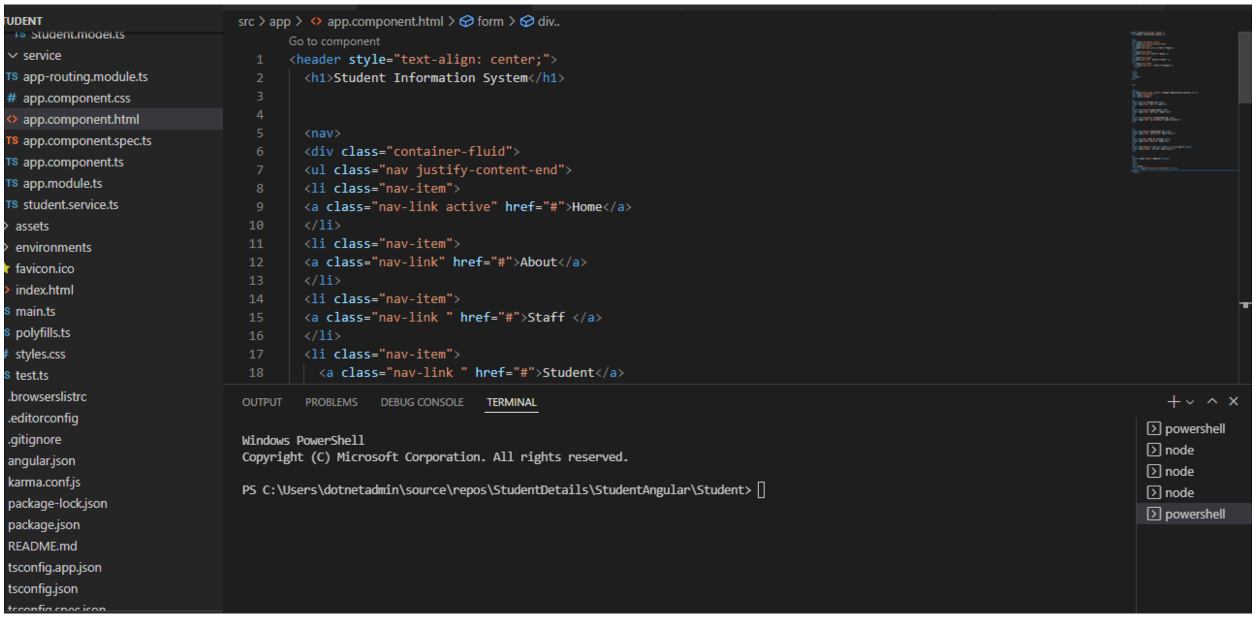
cd student

code .

then it will open in Vs

After installs npm bootstrap jquery popper.js --save

Let Create Html and css



```
src > app > app.component.html > form > div..
Go to component
<header style="text-align: center;">
  <h1>Student Information System</h1>


  <nav>
  <div class="container-fluid">
  <ul class="nav justify-content-end">
  <li class="nav-item">
  <a class="nav-link active" href="#">Home</a>
  </li>
  <li class="nav-item">
  <a class="nav-link" href="#">About</a>
  </li>
  <li class="nav-item">
  <a class="nav-link " href="#">Staff </a>
  </li>
  <li class="nav-item">
    <a class="nav-link " href="#">Student</a>
```

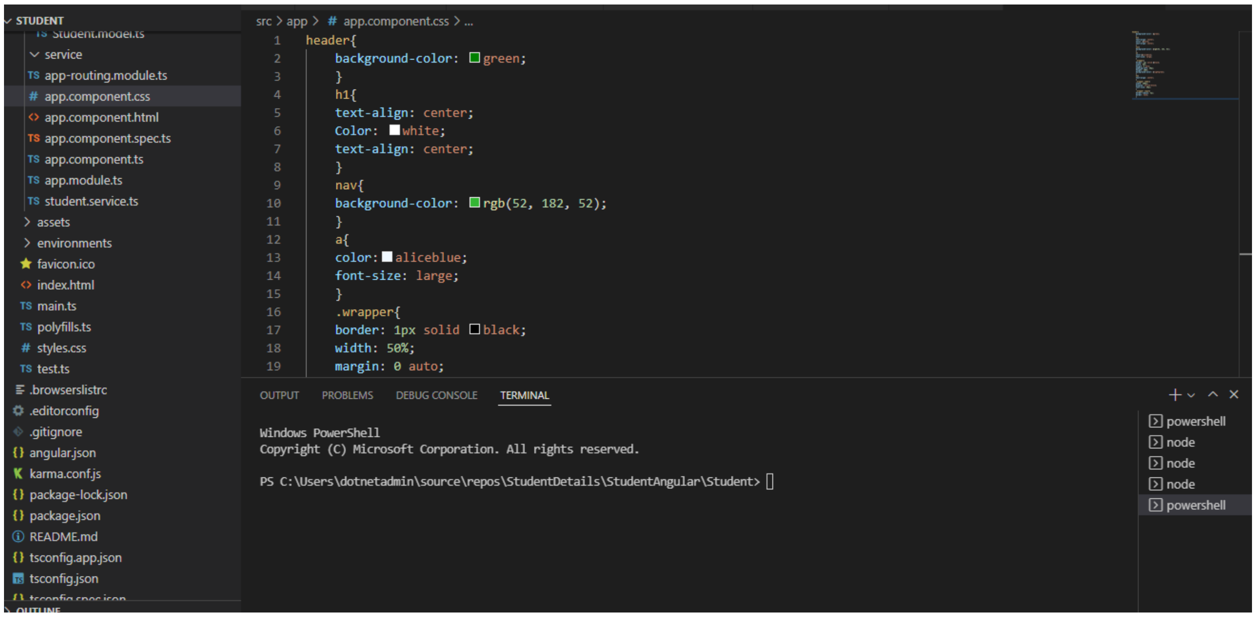
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\dotnetadmin\source\repos\StudentDetails\StudentAngular\Student>
```

Html Code

```
<header style="text-align: center;">
  <h1>Student Information System</h1>


  <nav>
  <div class="container-fluid">
  <ul class="nav justify-content-end">
  <li class="nav-item">
  <a class="nav-link active" href="#">Home</a>
  </li>
  <li class="nav-item">
  <a class="nav-link" href="#">About</a>
  </li>
  <li class="nav-item">
  <a class="nav-link " href="#">Staff </a>
  </li>
  <li class="nav-item">
    <a class="nav-link " href="#">Student</a>
    </li>

  </ul>
  </div>
  </nav>
  </header>
  <br>
```

```html
<br>


<form >
<h3 style="text-align: center;">Student Registration Section</h3><br>
<div class="container">
<div class="wrapper">

<p>
<label for="id">Student ID</label>
<input type="text" id="id" name="id">
</p>
<p>
<label for="name">Student Name</label>
<input type="text" id="name" name="name">
</p>
<p>
<label for="StudEmail">StudentEmailId</label>
<input type="text" id="StudEmail" name="StudEmail">
</p>



<p>
<label for="Deptid">Department ID</label>
<input type="text" id="Deptid" name="Deptid">
</p>
<p>
<label for="doj">Date Of Joining</label>
<input type="date" id="doj" name="doj">
</p>
<p>
<label for="Address"><Address></Address><br>(in years)</label>
<input type="number" id="Add" name="Address">
</p>


<p>
<button type="submit">Register</button>
</p>
</div>
</div>
<div class=" ">
  <a><i  class="fa-solid fa-trash-can"></i></a>
</div>
</form>
```

Next css



```
header{
    background-color: green;
    }
    h1{
    text-align: center;
    Color: white;
    text-align: center;
    }
    nav{
    background-color: rgb(52, 182, 52);
    }
    a{
    color:aliceblue;
    font-size: large;
    }
    .wrapper{
    border: 1px solid black;
    width: 50%;
    margin: 0 auto;
```

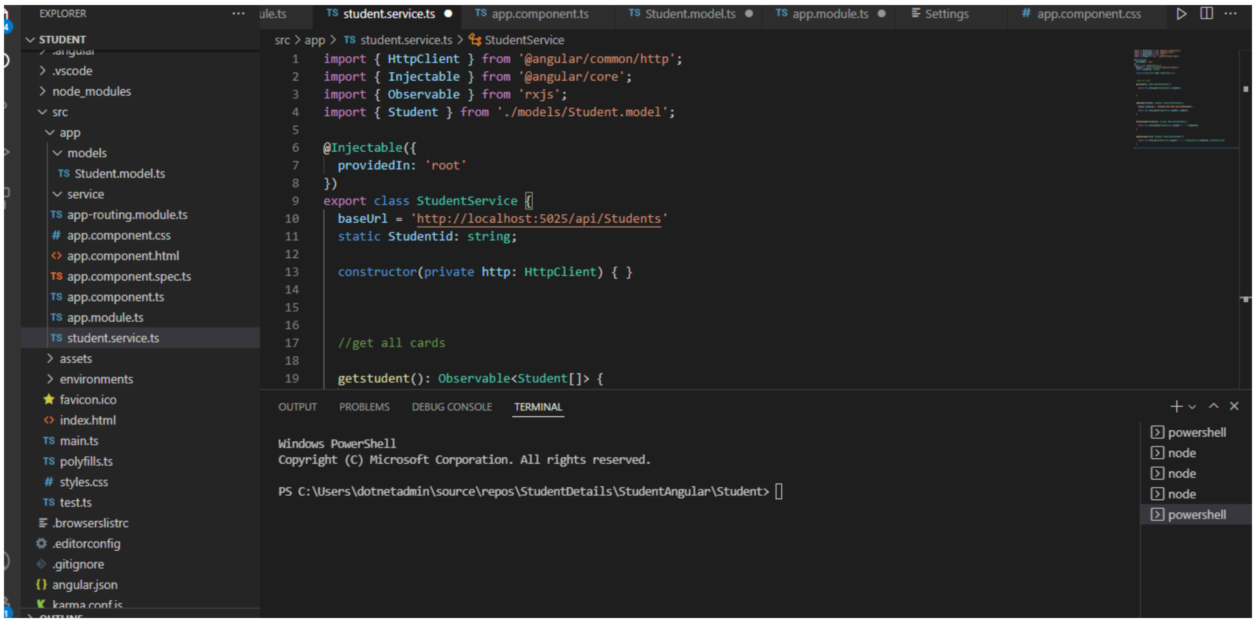
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\dotnetadmin\source\repos\StudentDetails\StudentAngular\Student>
```

Css code

```
header{
    background-color: green;
    }
    h1{
    text-align: center;
    Color: white;
    text-align: center;
    }
    nav{
    background-color: rgb(52, 182, 52);
    }
    a{
    color:aliceblue;
    font-size: large;
    }
    .wrapper{
    border: 1px solid black;
    width: 50%;
    margin: 0 auto;
    padding-left: 20px;
    height: 100%;
    background-color: lightgreen;
    }
    h2{
    text-align: center;
    }
    .wrapper label{
    width: 200px;
    display: inline-block;
    font-size: 20px;
    }
    .wrapper input{
    border-radius: 5px;
    border: none;
    }
```

Next creating Service and modules

Service.ts



```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs';
import { Student } from './models/Student.model';

@Injectable({
  providedIn: 'root'
})
export class StudentService {
  baseUrl = 'http://localhost:5025/api/Students'
  static Studentid: string;

  constructor(private http: HttpClient) { }



  //get all cards

  getstudent(): Observable<Student[]> {
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\dotnetadmin\source\repos\StudentDetails\StudentAngular\Student>
```

Code

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs';
import { Student } from './models/Student.model';

@Injectable({
  providedIn: 'root'
})
export class StudentService {
  baseUrl = 'http://localhost:5025/api/Students'
  static Studentid: string;

  constructor(private http: HttpClient) { }



  //get all cards

  getstudent(): Observable<Student[]> {

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



  }



  addStudent(Student: Student): Observable<Student> {

    Student.StudentId = '00000000-0000-0000-0000-000000000000';

    return this.http.post<Student>(this.baseUrl, Student);

  }



  deleteStudent(StudentId: string): Observable<Student> {

    return this.http.delete<Student>(this.baseUrl + '/' + StudentId);

  }



  updateStudent(card: Student): Observable<Student> {
    return this.http.put<Student>(this.baseUrl + '/' + StudentService.Studentid,
StudentService);

  }

}
```

Appcomponent.ts code

```
import { Component, OnInit } from '@angular/core';
import { Student } from './models/Student.model';
import { StudentService } from './student.service';

@Component({

   selector: 'app-root',

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

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

  })

  export class AppComponent {

   title = 'Student';

   Student: Student[] = [];

   student: Student = {

    StudentId: '',
    StudentName: ' ',
    StudentEmailId: ' ',
    DepartmentId: ' ',
    DateOfJoining: ' ',
    Address: ' '


   }

   constructor(private StudentService: StudentService) {

   }
```

```
 ngOnInit(): void {

 this.getstudent();

 }

getstudent(){

 this.StudentService.getstudent()

.subscribe(

response => {

 this.Student = response;

 }

);

}

 onSubmit () {

 if (this.student.StudentId === '') {

this.StudentService.addStudent(this.student)

.subscribe(

 response => {

 this.getstudent();



 }

);

}else {

 this.updateCard(this.student);
 }

 }

 deleteCard(id: string) {

  this.StudentService.deleteStudent(id)

 .subscribe(


  response => {


  this.getstudent();

 }

  );

 }

 populateForm(Student: Student) {

this.student = Student;

 }

 updateCard(Student: Student) {

 this.StudentService.updateStudent(Student)

  .subscribe(

 response => {

 this.getstudent();

 }

 );
```

```
  }

  }
```

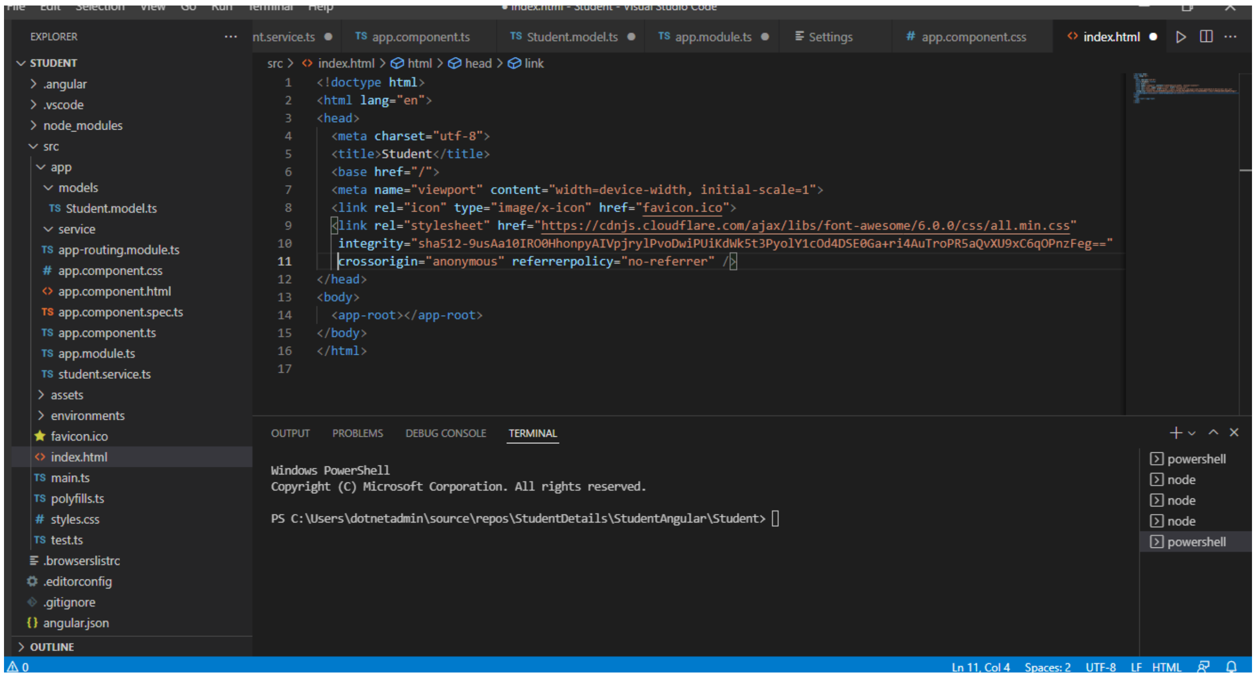
Appmodule.ts

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import {HttpClientModule} from '@angular/common/http';
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { FormsModule } from '@angular/forms';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    HttpClientModule,
    FormsModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Student.model.ts

```
export interface Student{
    StudentId: String;
    StudentName: String;
    StudentEmailId: String;
    DepartmentId: String;
    DateOfJoining: String;
    Address: String;
}
```

Index.Html

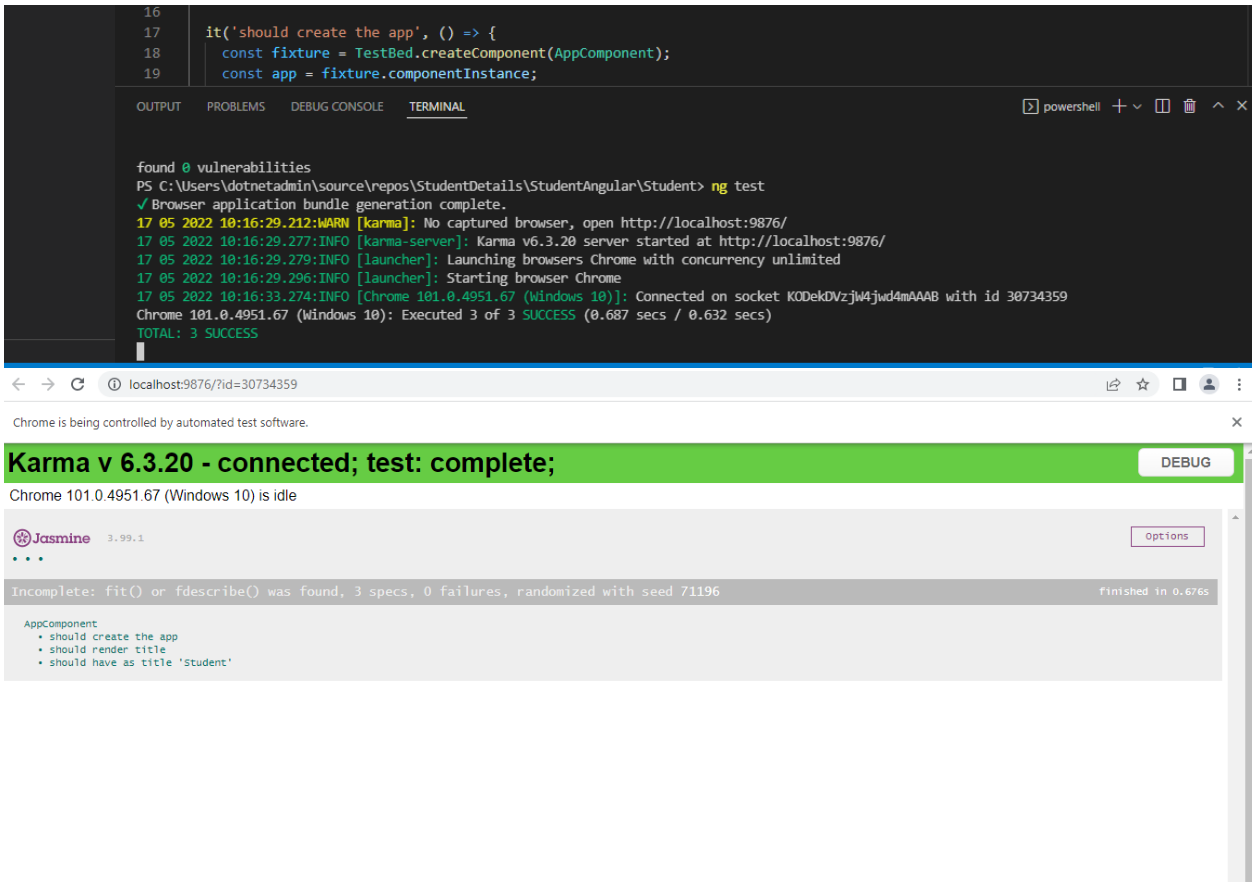


```html
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Student</title>
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css"
   integrity="sha512-9usAa10IRO0HhonpyAIVpjrylPvoDwiPUiKdWk5t3PyolY1cOd4DSE0Ga+ri4AuTroPR5aQvXU9xC6qOPnzFeg=="
   crossorigin="anonymous" referrerpolicy="no-referrer" />
</head>
<body>
  <app-root></app-root>
</body>
</html>
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\dotnetadmin\source\repos\StudentDetails\StudentAngular\Student>
```

Next doing

ng test



```
it('should create the app', () => {
  const fixture = TestBed.createComponent(AppComponent);
  const app = fixture.componentInstance;
```

```
found 0 vulnerabilities
PS C:\Users\dotnetadmin\source\repos\StudentDetails\StudentAngular\Student> ng test
✓ Browser application bundle generation complete.
17 05 2022 10:16:29.212:WARN [karma]: No captured browser, open http://localhost:9876/
17 05 2022 10:16:29.277:INFO [karma-server]: Karma v6.3.20 server started at http://localhost:9876/
17 05 2022 10:16:29.279:INFO [launcher]: Launching browsers Chrome with concurrency unlimited
17 05 2022 10:16:29.296:INFO [launcher]: Starting browser Chrome
17 05 2022 10:16:33.274:INFO [Chrome 101.0.4951.67 (Windows 10)]: Connected on socket KODekDVzjW4jwd4mAAAB with id 30734359
Chrome 101.0.4951.67 (Windows 10): Executed 3 of 3 SUCCESS (0.687 secs / 0.632 secs)
TOTAL: 3 SUCCESS
```

localhost:9876/?id=30734359
Chrome is being controlled by automated test software.

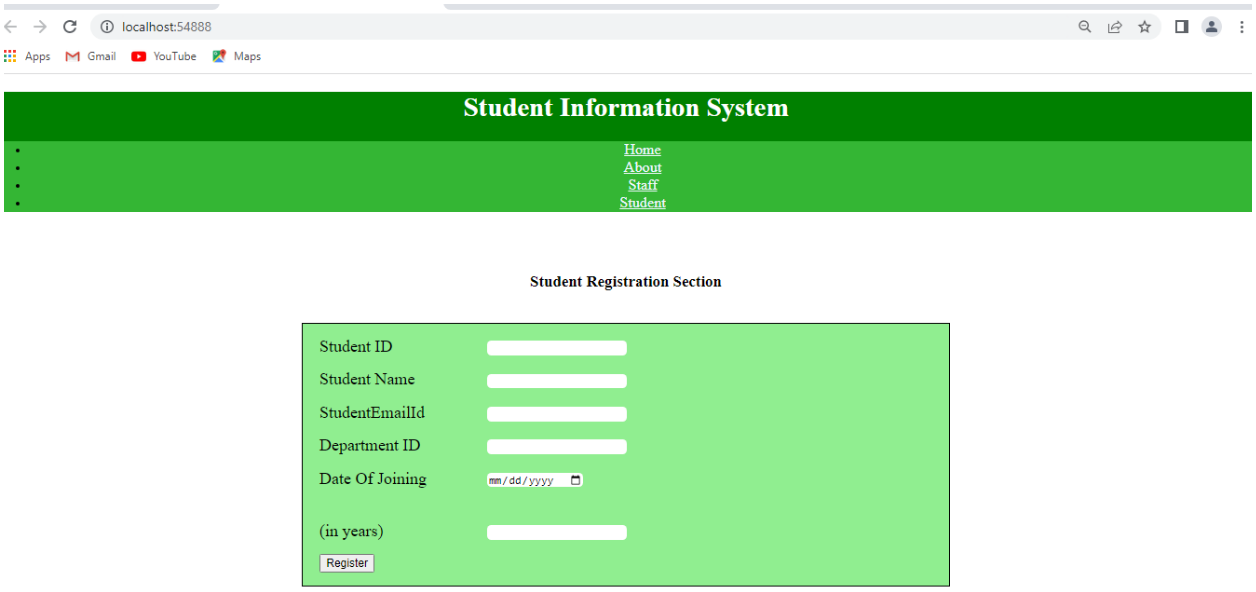
Karma v 6.3.20 - connected; test: complete;

Chrome 101.0.4951.67 (Windows 10) is idle

```
Incomplete: fit() or fdescribe() was found, 3 specs, 0 failures, randomized with seed 71196    finished in 0.676s

AppComponent
  • should create the app
  • should render title
  • should have as title 'Student'
```

Output

Next run the code ng serve

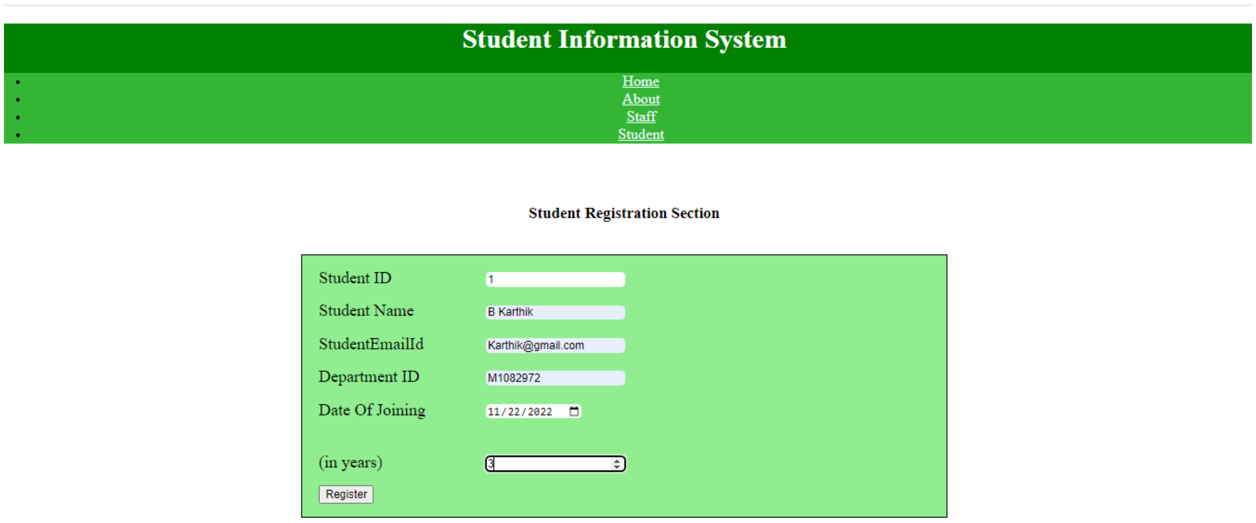


Student Information System

- Home
- About
- Staff
- Student

Student Registration Section

Student ID	
Student Name	
StudentEmailId	
Department ID	
Date Of Joining	mm/dd/yyyy
(in years)	



Student Information System

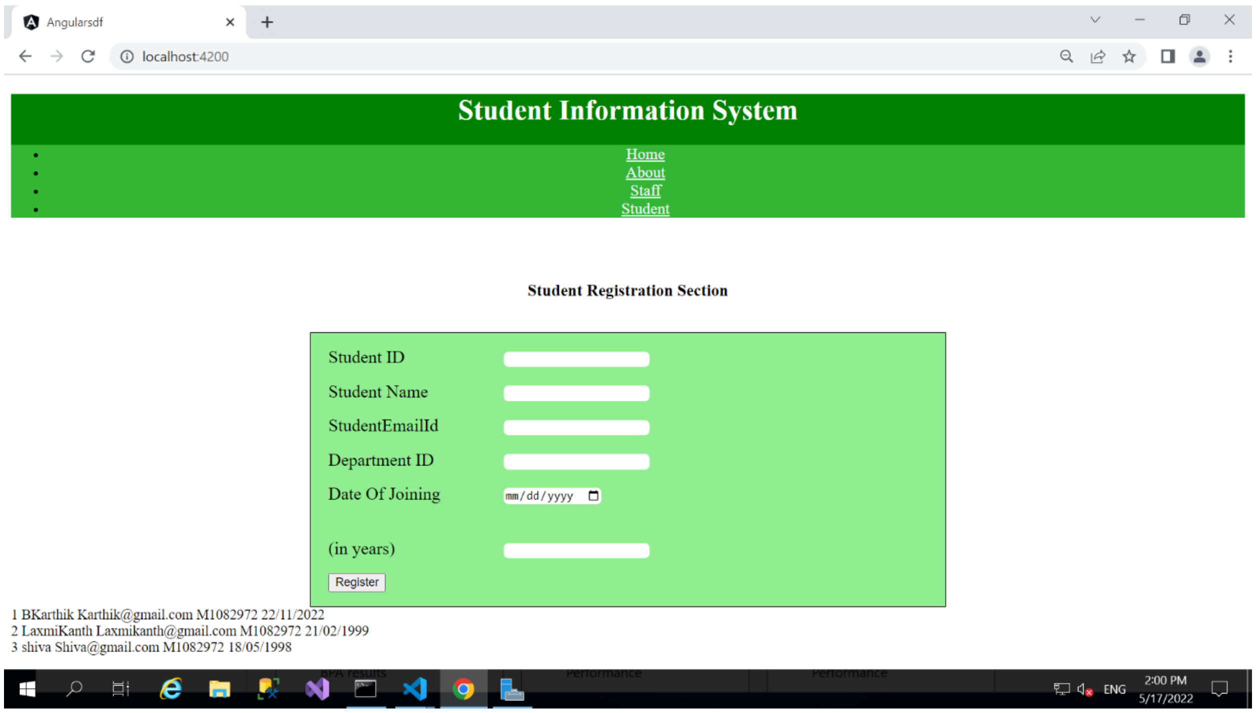
- Home
- About
- Staff
- Student

Student Registration Section

Student ID	1
Student Name	B Karthik
StudentEmailId	Karthik@gmail.com
Department ID	M1082972
Date Of Joining	11/22/2022
(in years)	3

After Clicking the register button

We get



Student Information System

- Home
- About
- Staff
- Student

Student Registration Section

Student ID	
Student Name	
StudentEmailId	
Department ID	
Date Of Joining	mm/dd/yyyy
(in years)	

```
1 BKarthik Karthik@gmail.com M1082972 22/11/2022
2 LaxmiKanth Laxmikanth@gmail.com M1082972 21/02/1999
3 shiva Shiva@gmail.com M1082972 18/05/1998
```

Getting register details