

PROJECT INTRODUCTION

Student Management MVC Application

This project is an ASP.NET Core MVC web application used to manage student records.

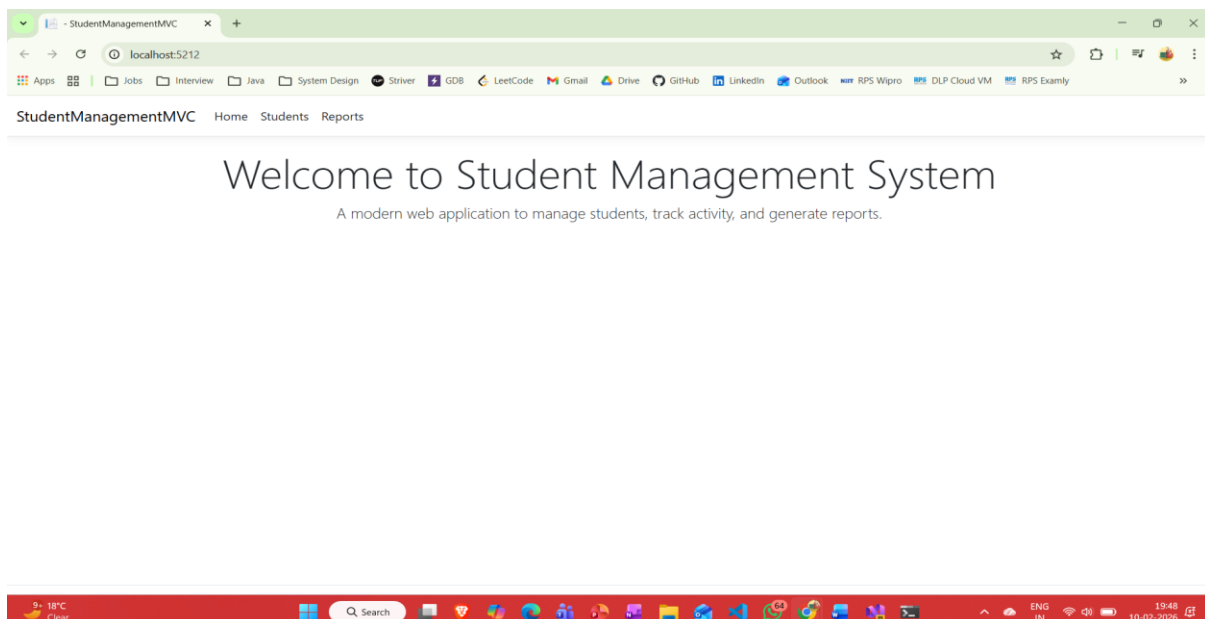
It supports viewing, adding, editing, and deleting students, and viewing reports.

Technologies Used:

- ASP.NET Core MVC
- Entity Framework Core
- ADO.NET
- SQL Server

RUN PROJECT & LOCALHOST

```
C:\Users\Aman Goswami\source\repos\StudentManagementMVC\StudentManagementMVC
dbug: Microsoft.AspNetCore.Mvc.ViewFeatures.BrowserRefreshMiddleware[0]
      Middleware loaded
dbug: Microsoft.AspNetCore.Mvc.ViewFeatures.BrowserRefreshMiddleware[0]
      Middleware loaded. Script /_framework/aspnetcore-browser-refresh.js (16525 B).
dbug: Microsoft.AspNetCore.Mvc.ViewFeatures.BrowserRefreshMiddleware[0]
      Middleware loaded. Script /_framework/blazor-hotreload.js (799 B).
dbug: Microsoft.AspNetCore.Mvc.ViewFeatures.BrowserRefreshMiddleware[0]
      Middleware loaded: DOTNET_MODIFIABLE_ASSEMBLIES=debug, __ASPNETCORE_BROWSER_TOOLS=true
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5212
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\Users\Aman Goswami\source\repos\StudentManagementMVC\StudentManagementMVC
dbug: Microsoft.AspNetCore.Mvc.ViewFeatures.BrowserRefreshMiddleware[1]
      Response markup is scheduled to include browser refresh script injection.
dbug: Microsoft.AspNetCore.Mvc.ViewFeatures.BrowserRefreshMiddleware[2]
      Response markup was updated to include browser refresh script injection.
dbug: Microsoft.AspNetCore.Mvc.ViewFeatures.BrowserRefreshMiddleware[0]
      Script injected: /_framework/aspnetcore-browser-refresh.js
```



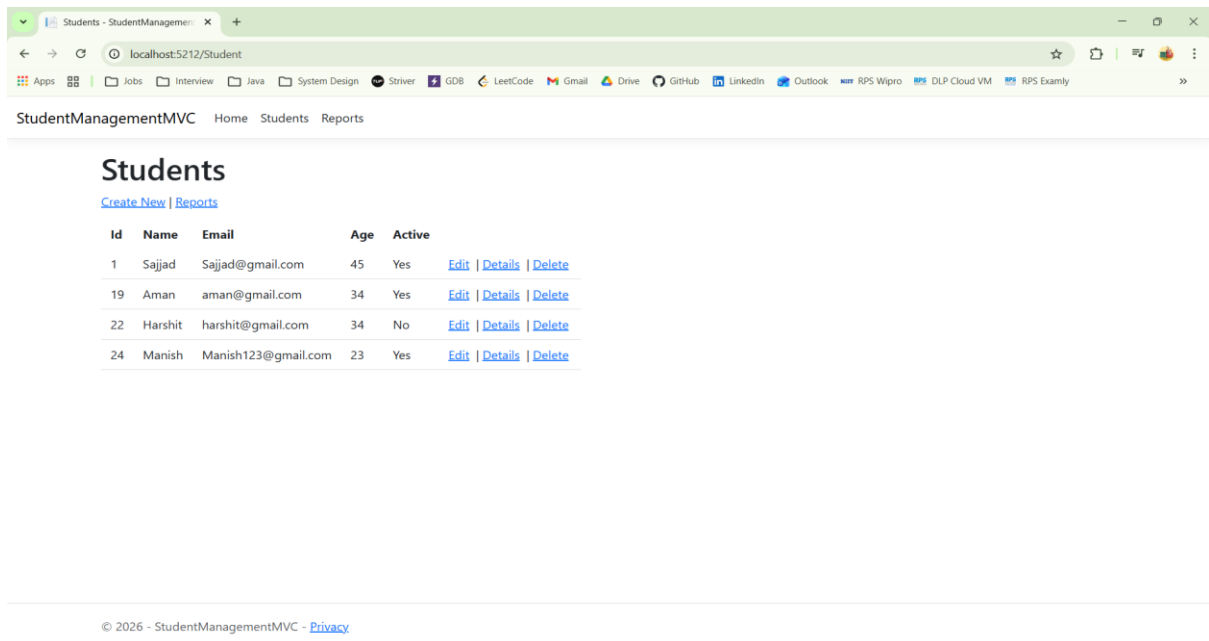
After clicking Run, the application starts from Program.cs.

The web server starts when app.Run() executes, and localhost opens automatically.

STUDENTS LIST PAGE

This page displays all students fetched from the database.

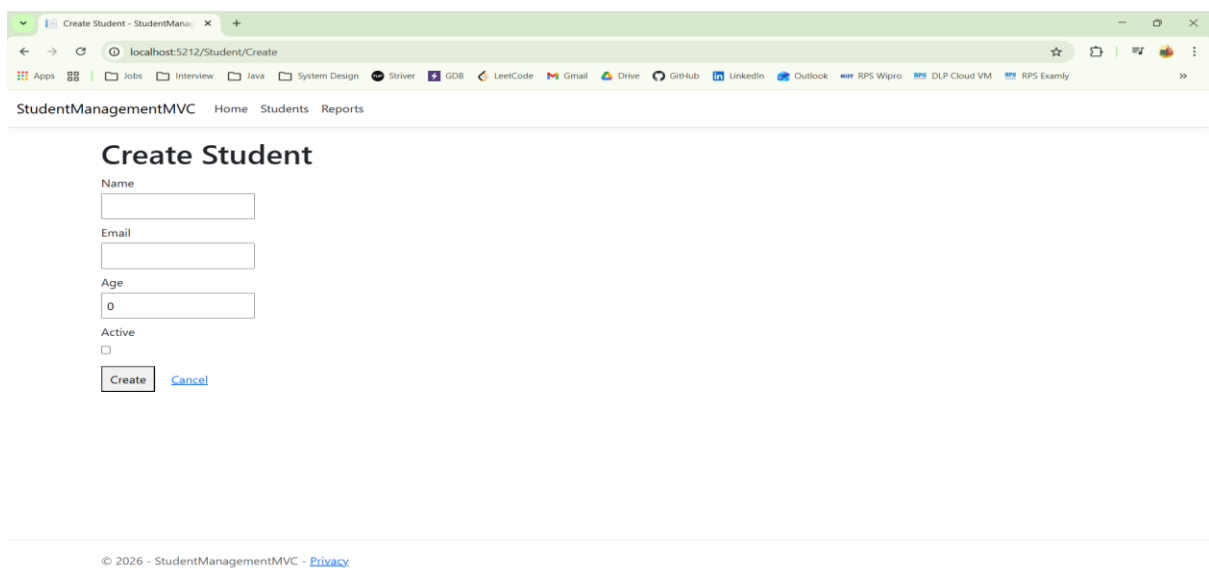
The request is handled by StudentController, and data is retrieved using StudentService and EF Core.



ADD STUDENT PAGE

This page allows adding a new student.

User input is validated, and data is saved to the database using Entity Framework Core.



ADD STUDENT SUCCESS

After submitting the form, the student is saved to the database.

A success message is displayed, and the updated list is shown.

Students - StudentManagementMVC

Home Students Reports

Students

Student added successfully.

[Create New](#) | [Reports](#)

Id	Name	Email	Age	Active	
1	Sajjad	Sajjad@gmail.com	45	Yes	Edit Details Delete
19	Aman	aman@gmail.com	34	Yes	Edit Details Delete
22	Harshit	harshit@gmail.com	34	No	Edit Details Delete
24	Manish	Manish123@gmail.com	23	Yes	Edit Details Delete
25	Asmit	asmit@gmail.com	28	Yes	Edit Details Delete

© 2026 - StudentManagementMVC - [Privacy](#)

EDIT STUDENT

This page allows editing an existing student.

The data is loaded from the database and updated using Entity Framework Core.

Students - StudentManagementMVC

Home Students Reports

Students

Student updated successfully.

[Create New](#) | [Reports](#)

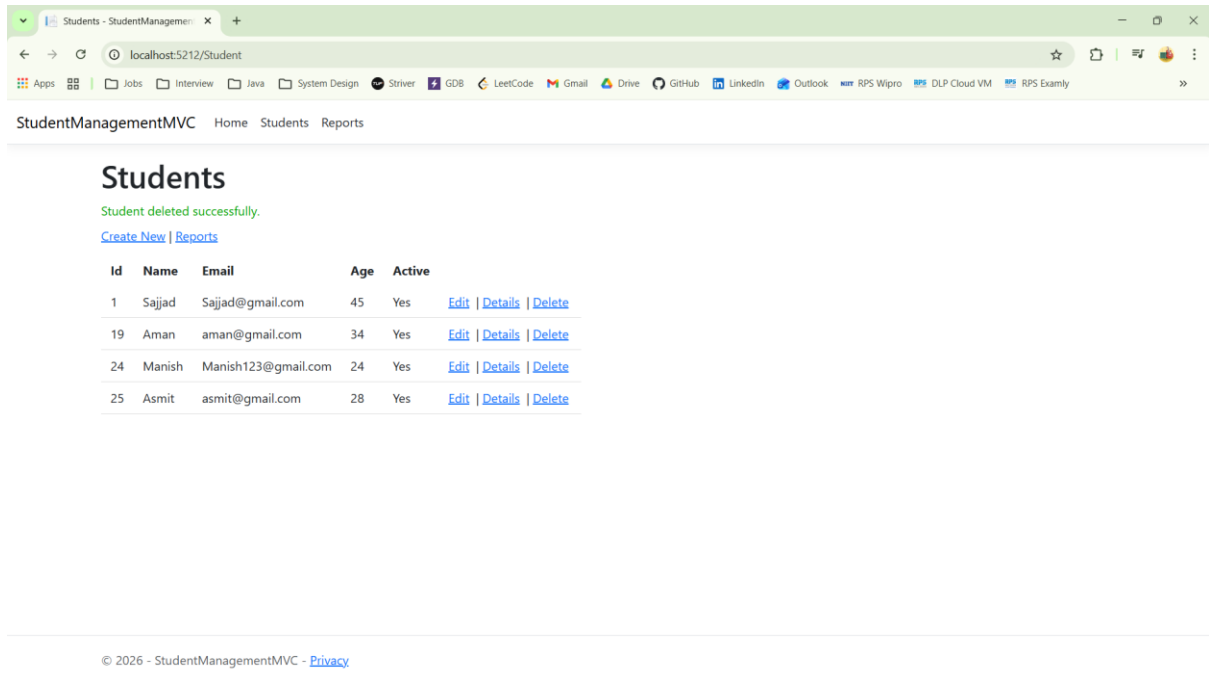
Id	Name	Email	Age	Active	
1	Sajjad	Sajjad@gmail.com	45	Yes	Edit Details Delete
19	Aman	aman@gmail.com	34	Yes	Edit Details Delete
22	Harshit	harshit@gmail.com	34	No	Edit Details Delete
24	Manish	Manish123@gmail.com	24	Yes	Edit Details Delete
25	Asmit	asmit@gmail.com	28	Yes	Edit Details Delete

© 2026 - StudentManagementMVC - [Privacy](#)

DELETE STUDENT

This page confirms student deletion.

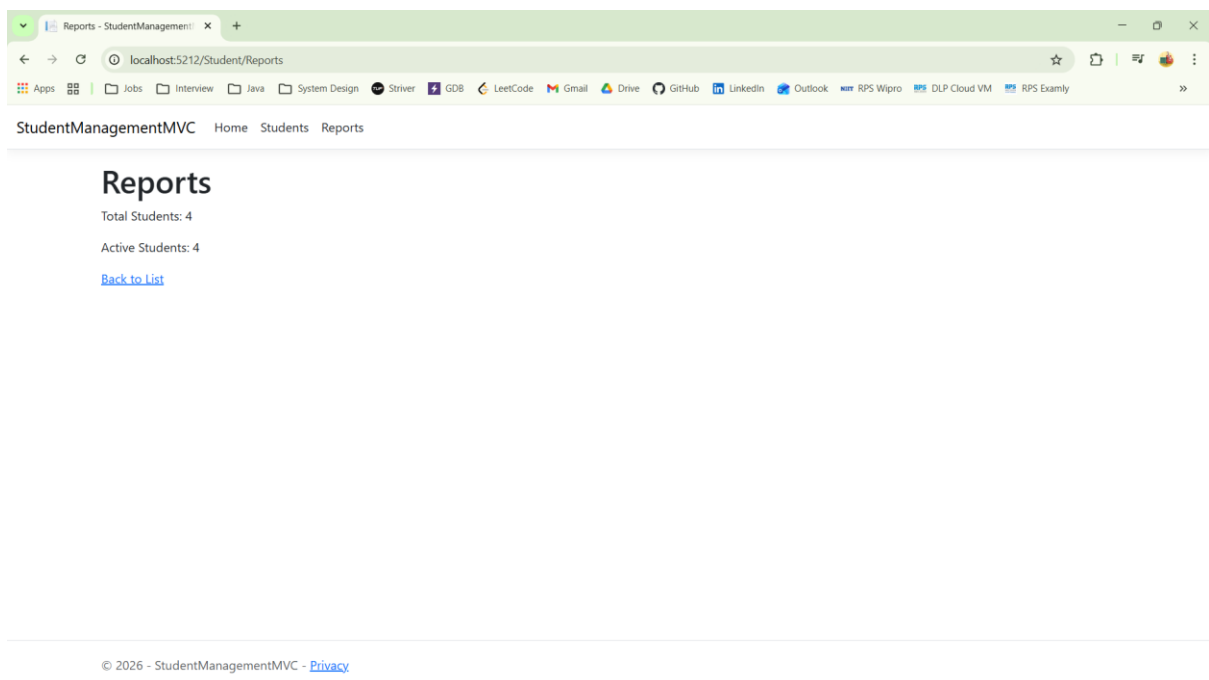
After confirmation, the record is removed from the database.



REPORTS PAGE

This page displays student reports.

ADO.NET is used here to fetch total and active students directly from the database.



DATABASE

This screenshot shows the Students table stored in SQL Server.

The database structure is created using the provided SQL script.

The screenshot displays the SQL Server Enterprise Manager interface. On the left, the Object Explorer shows the database structure for 'StudentDB'. The 'Tables' folder is expanded, showing the 'Students' table. The table's columns are listed: 'StudentId' (INT, PK, not null), 'Name' (nvarchar(100), not null), 'Email' (nvarchar(100), not null), 'Age' (int, not null), and 'IsActive' (bit, not null). The 'Columns' folder is also expanded, showing the 'StudentId' column details.

The main window shows the SQL script for creating the 'Students' table and a query to select all data from it:

```
1 CREATE TABLE Students (  
2     StudentId INT IDENTITY(1,1) PRIMARY KEY,  
3     Name NVARCHAR(100) NOT NULL,  
4     Email NVARCHAR(100) NOT NULL,  
5     Age INT NOT NULL,  
6     IsActive BIT NOT NULL  
7 );  
8 SELECT * FROM Students;
```

The 'Results' pane shows the data returned by the query:

StudentId	Name	Email	Age	IsActive
1	Sajad	Sajad@gmail.com	45	1
2	Aman	aman@gmail.com	34	1
3	Manish	Manish123@gmail.com	24	1
4	Aamit	aamit@gmail.com	28	1

The status bar at the bottom indicates that the query was executed successfully, returning 4 rows.