Software Development

Presented By:
Khaled Gamal

ASP.NET Core

MVC

Section 7

CRUD – Operations – By URL

Insert New Student In Table By URL

C# (StudentsController.cs)

```
public IActionResult InsertByURL(string fname, string lname)
{
Student student = new Student { EnrollmentDate = DateTime.Now, FirstMidName = fname,
LastName = lname };
_context.Students.Add(student);
_context.SaveChanges();
return RedirectToAction("Index");
}
```

- Type this URL in the browser search box:

https://localhost:####/students/insertbyurl?fname=software&lname=development

Insert New Student In Table By URL

ID: 7
FirstMidName: Laura
LastName:Norman
EnrollmentDate:2003-09-01 12:00:00 AM
Delete | Edit | Details

ID: 8
FirstMidName: Nino
LastName:Olivetto
EnrollmentDate:2005-09-01 12:00:00 AM
Delete | Edit | Details

ID: 9
FirstMidName: software
LastName:development
EnrollmentDate:2024-04-21 10:17:02 AM
Delete | Edit | Details

Search for one Student In Table By URL

C# (StudentsController.cs)

```
public IActionResult SearchByURL(int id)
{
return Content(_context.Students.FirstOrDefault(s => s.ID == id).ToString());
}
```

C# (Student.cs)

```
public override string ToString()
{
return $" ID = {ID} \n LastName = {LastName} " +
$"\n FirstMidName = {FirstMidName} \n EnrollmentDate = {EnrollmentDate}";
}
```

- Type this URL in the browser search box:

https://localhost:####/Students/searchbyurl/2

Search for one Student In Table By URL

```
ID = 2
LastName = Alonso
FirstMidName = Meredith
EnrollmentDate = 2002-09-01 12:00:00 AM
```

Delete one Student from table By URL

C# (StudentsController.cs)

```
public IActionResult DeleteByURL(int id)
{
Student st = _context.Students.FirstOrDefault(S => S.ID == id);
if (st != null)
{
_context.Students.Remove(st);
_context.SaveChanges();
}
return RedirectToAction("Index");
}
```

- Type this URL in the browser search box:

https://localhost:####/Students/deletebyurl/9

Delete one Student from table By URL

EnrollmentDate:2003-09-01 12:00:00 AM

Delete | Edit | Details

ID: 8

FirstMidName: Nino

LastName:Olivetto

EnrollmentDate:2005-09-01 12:00:00 AM

Delete | Edit | Details

Update one Student in table By URL

C# (StudentsController.cs)

```
public IActionResult UpdateByURL(int id, string fname, string lname)
{
Student st = _context.Students.FirstOrDefault(S => S.ID == id);
if (st != null)
{
st.FirstMidName = fname;
st.LastName = lname;
_context.Students.Update(st);
_context.SaveChanges();
}
return RedirectToAction("Index");
}
```

- Type this URL in the browser search box:

https://localhost:####/Students/updatebyurl?id=2&fname=Khaled&lname=Gamal

Update one Student in table By URL

ID: 2

FirstMidName: Meredith

LastName:Alonso

EnrollmentDate:2002-09-01 12:00:00 AM

ID: 2

FirstMidName: Khaled

LastName:Gamal

EnrollmentDate:2002-09-01 12:00:00 AM

Delete | Edit | Details

CRUD – Operations – By Views

Create (insert) – Operation

Insert New Student

- Lets add a button in the Students index page to add new student and some style for the page (Highlighted):

CSHTML (Index.cshtml)

```
@model IEnumerable<Student>
@{
ViewData["Title"] = "Students";
}
<div class="m-2">
<a class="btn btn-primary" asp-action="Insert" asp-controller="Students">Add New Student</a>
</div>
<div class="overflow-auto" style="height:570px">
@foreach (var s in Model)
{
<div class="card bg-warning m-2">
<div class="card-body m-2">
<h2>@Html.DisplayNameFor(s => s.ID): @s.ID</h2>
<h2>@Html.DisplayNameFor(s => s.FirstMidName): @s.FirstMidName</h2>
<h2>@Html.DisplayNameFor(s => s.LastName): @s.LastName</h2>
<h2>@Html.DisplayNameFor(s => s.EnrollmentDate): @s.EnrollmentDate</h2>
<a class="btn btn-danger" asp-action="Delete" asp-route-id="@s.ID">Delete</a> |
<a class="btn btn-success" asp-action="Edit" asp-route-id="@s.ID">Edit</a> |
<a class="btn btn-info" asp-action="Details" asp-route-id="@s.ID">Details</a>
</div>
</div>
}
</div>
```

Insert New Student

- Lets create two actions for inserting operation one for GET and the other for POST in the Students Controller.
- Create a view for insert operation that contains a form that takes the values of a new student and post them to the server.

Insert New Student

C# (StudentsController.cs)

```
[HttpGet]
public IActionResult Insert()
{
    return View();
}

[HttpPost]
public IActionResult Insert(Student student)
{
    ModelState.Remove("Enrollments");
    if (ModelState.IsValid)
    {
        _context.Students.Add(student);
        _context.SaveChanges();
        return RedirectToAction("Index");
    }
    return View(student);
}
```

Insert New Student

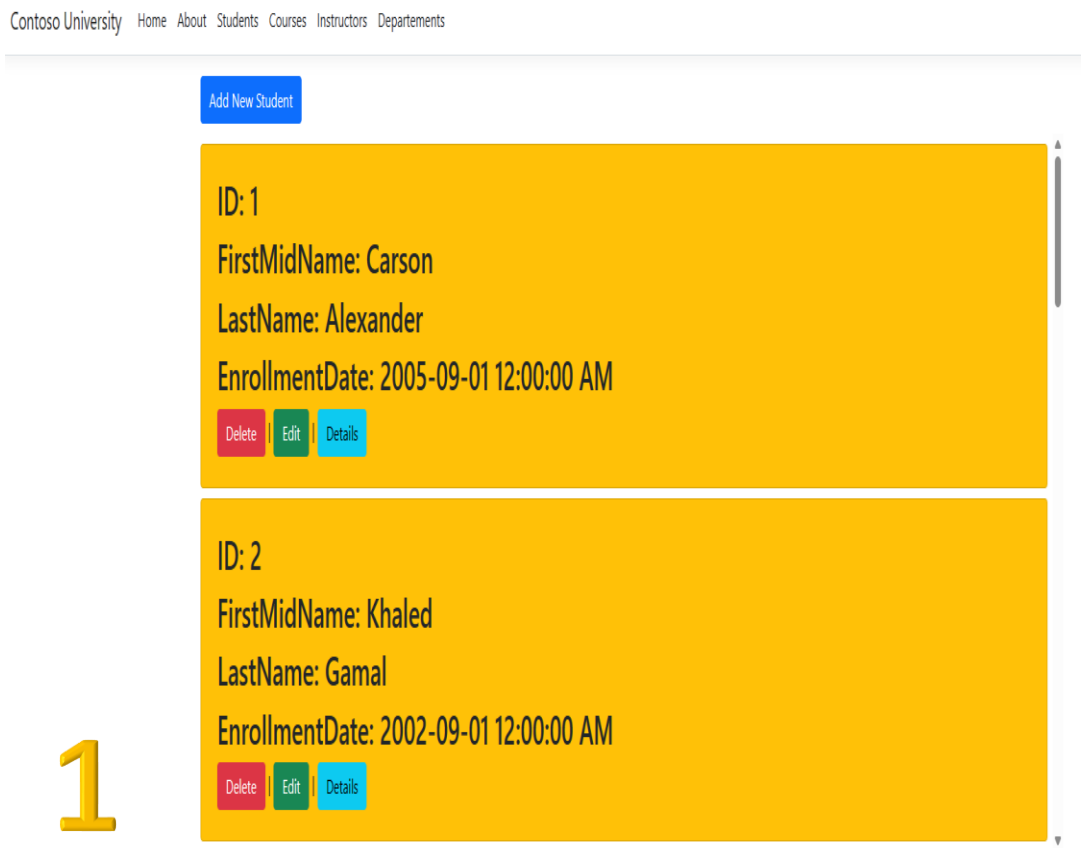
CSHTML (Insert.cshtml)

```
@model Student;
@{
    ViewData["Title"] = "Add New Student";
}
<h1>Add New Student</h1>
<form asp-controller="Students" asp-action="Insert" method="post" class="bg-light">
    <div class="form-group m-2">
        <label asp-for="FirstMidName"></label>
        <input asp-for="FirstMidName" class="form-control" />
        <span asp-validation-for="FirstMidName" class="text-danger"></span>
    </div>

    <div class="form-group m-2">
        <label asp-for="LastName"></label>
        <input asp-for="LastName" class="form-control" />
        <span asp-validation-for="LastName" class="text-danger"></span>
    </div>

    <div class="form-group m-2">
        <label asp-for="EnrollmentDate"></label>
        <input asp-for="EnrollmentDate" class="form-control" />
        <span asp-validation-for="EnrollmentDate" class="text-danger"></span>
    </div>

    <button type="submit" class="btn btn-primary m-2">Add</button>
</form>
```

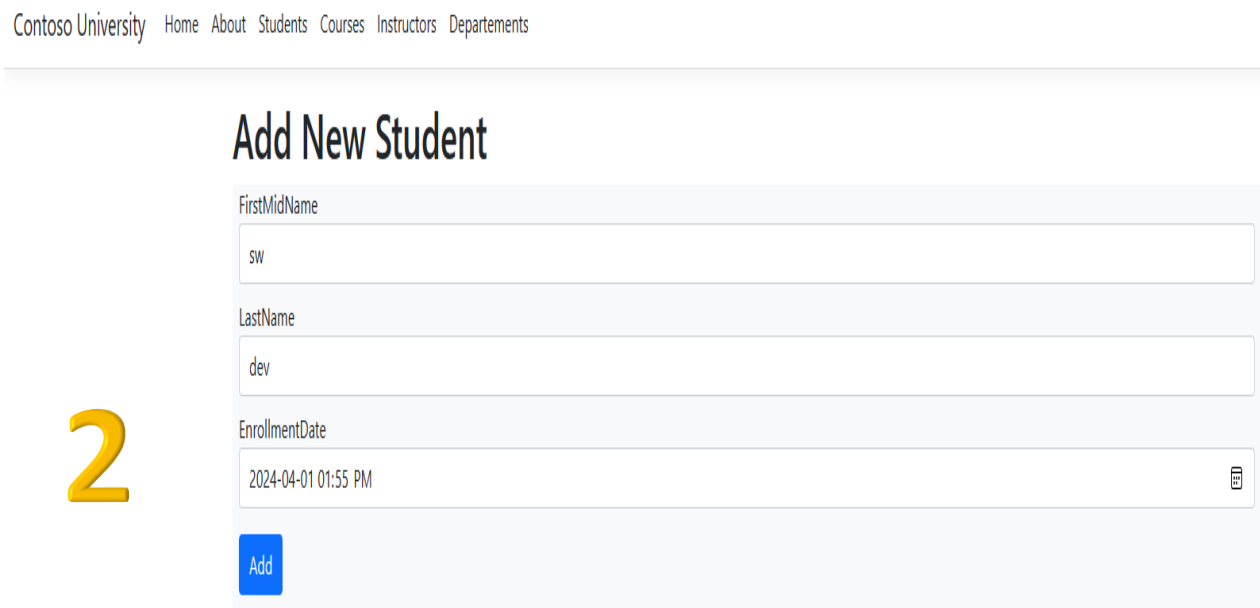
Insert New Student

- In this example, the <form> tag is used to create an HTML form that will submit the data to the “**Insert**" action method in the “**Students**" controller.
- The asp-controller and asp-action attributes are used to specify the controller and action method that will handle the form submission.
- The asp-for attribute is used to bind the HTML input elements to the corresponding properties of the Student model.
- The asp-validation-for attribute is used to display any validation errors for the input fields.
- Finally, an HTML submit button is created with the text " **Add** " and the class "btn btn-primary".
- When the user clicks this button, the data in the form will be submitted to the " **Insert** " action method in the " **Students** " controller for processing.

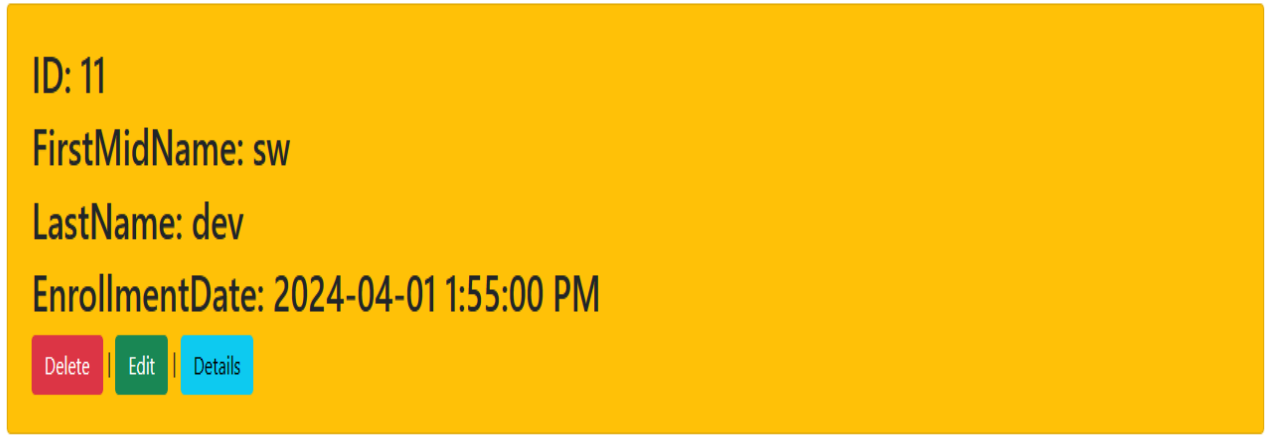
Insert New Student



1



2



3

Read (Details) – Operation

Read (Details) – Operation for each student

- In the index page of students we create three buttons for every student (Delete – Edit - Details) and every button has an action to process it and they send the ID of that student to that action.
- So lets create an *action* called *Details* that takes one integer parameter called *id* that will render a view that shows the details of the student with that ID.

C# (StudentsController.cs)

```
[HttpGet]
public IActionResult Details(int id)
{
    Student student = _context.Students.Where(S => S.ID == id).
        Include(s => s.Enrollments).
        ThenInclude(c => c.Course).
        AsNoTracking().
        FirstOrDefault();
    return View(student);
}
```

Read (Details) – Operation for each student

- The Include and ThenInclude methods cause the context to load the Student.Enrollments navigation property, and within each enrollment the Enrollment.Course navigation property.
- The AsNoTracking method improves performance in scenarios where the entities returned won't be updated in the current context's lifetime.

C# (StudentsController.cs)

```
[HttpGet]
public IActionResult Details(int id)
{
    Student student = _context.Students.Where(S => S.ID == id).
        Include(s => s.Enrollments).
        ThenInclude(c => c.Course).
        AsNoTracking().
        FirstOrDefault();
    return View(student);
}
```

Read (Details) – Operation for each student

- Then create the Details View:

CSHTML (Details.cshtml)

1

```
@model Student;
@{
    ViewData["Title"] = "Student's Deatils";
}
<h1>Student's Deatils</h1>
<div class="m-2 bg-light">
<p><strong>ID:</strong> @Model.ID</p>
<p><strong>First Mid Name:</strong> @Model.FirstMidName</p>
<p><strong>Last Name:</strong> @Model.LastName</p>
<p><strong>Date of Enrollment:</strong>
@Model.EnrollmentDate.ToShortDateString()</p>
<p><strong>@Html.DisplayNameFor(model => model.Enrollments):</strong></p>
</div>
<div class="m-2">
    <table class="table table-bordered table-striped">
        <thead>
            <tr>
                <th>
                    EnrollmentID
                </th>
                <th>
                    Course Title
                </th>
                <th>
                    Grade
                </th>
            </tr>
        </thead>
```

CSHTML (Details.cshtml)

2

```
<tbody>
            @foreach(var e in Model.Enrollments)
            {
                <tr>
                    <td>
                        @Html.DisplayFor(modelItem => e.EnrollmentID)
                    </td>
                    <td>
                        @Html.DisplayFor(modelItem => e.Course.Title)
                    </td>
                    <td>
                        @Html.DisplayFor(modelItem => e.Grade)
                    </td>
                </tr>
            }
        </tbody>
    </table>
</div>
<div class="m-2">
    <a class="btn btn-primary" asp-action="Index" asp-controller="Students">Back to list</a>
</div
```

Read (Details) – Operation for each student

Student's Deatils

ID: 1

First Mid Name: Carson

Last Name: Alexander

Date of Enrollment: 2005-09-01

Enrollments:

EnrollmentID	Course Title	Grade
1	Chemistry	A
2	Microeconomics	C
3	Macroeconomics	B

Back to list

Update (Edit) – Operation

Update (Edit) – Operation for each student

- So lets create two *actions* called *Edit* first one handle GET verb and takes one integer parameter called *id* that will render a view that show a form with the values of the student with that ID within its controls to be edited by submitting that form.
- And the other action for handling POST verb or form submission.

C# (StudentsController.cs)

```
[HttpGet]
public IActionResult Edit(int id)
{
    Student student = _context.Students.Find(id);
    return View(student);
}

[HttpPost]
public IActionResult Edit(Student student)
{
    ModelState.Remove("Enrollments");
    if (ModelState.IsValid)
    {
        _context.Students.Update(student);
        _context.SaveChanges();
        return RedirectToAction("Index");
    }
    return View(student);
}
```

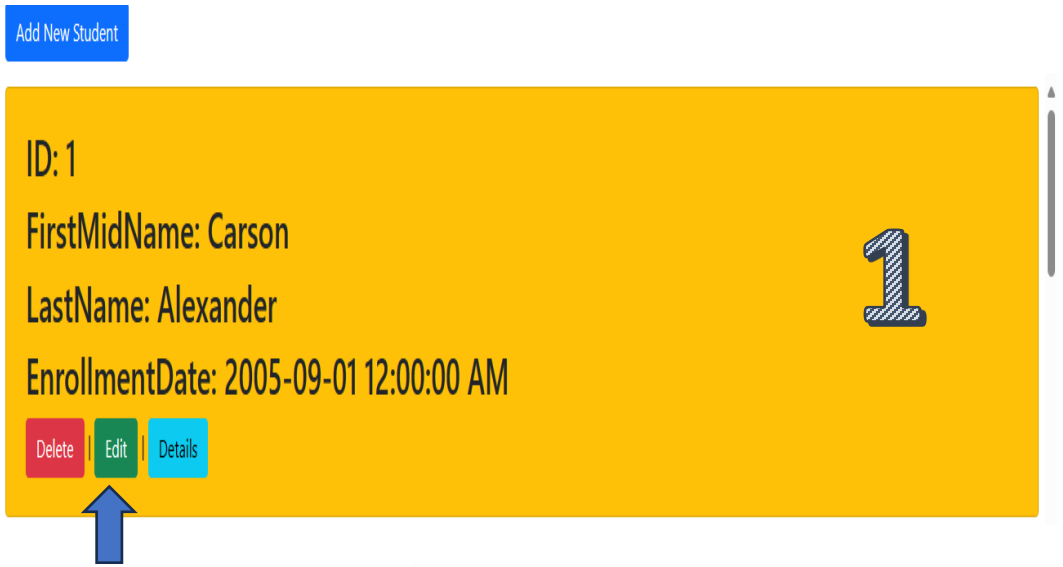
Update (Edit) – Operation for each student

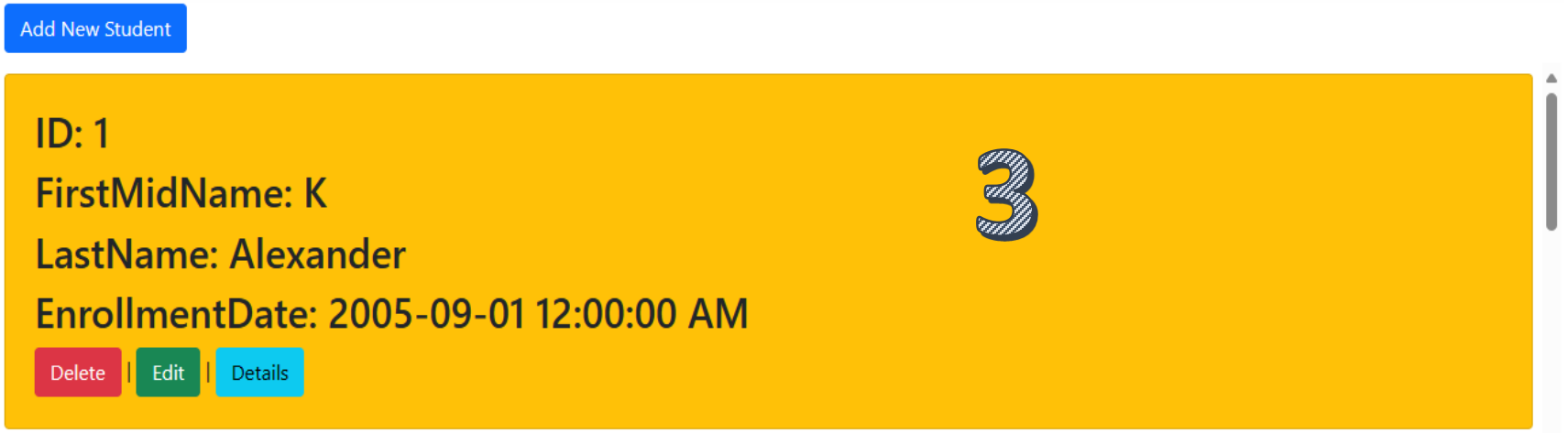
- Then create the Edit View which is similar to Insert View.

CSHTML (Edit.cshtml)

```
@model Student;
@{
    ViewData["Title"] = "Edit Student";
}
<h1>Edit Student Information</h1>
<div class="m-2">
    <a class="btn btn-primary" asp-action="Index" asp-controller="Students">Back to list</a>
</div>
<form asp-controller="Students" asp-action="Edit" method="post" class="bg-light">
    <div class="form-group m-2">
        <label asp-for="FirstMidName"></label>
        <input asp-for="FirstMidName" class="form-control" />
        <span asp-validation-for="FirstMidName" class="text-danger"></span>
    </div>
    <div class="form-group m-2">
        <label asp-for="LastName"></label>
        <input asp-for="LastName" class="form-control" />
        <span asp-validation-for="LastName" class="text-danger"></span>
    </div>
    <div class="form-group m-2">
        <label asp-for="EnrollmentDate"></label>
        <input asp-for="EnrollmentDate" class="form-control" />
        <span asp-validation-for="EnrollmentDate" class="text-danger"></span>
    </div>
    <button type="submit" class="btn btn-primary m-2">Save</button>
</form>
```

Update (Edit) – Operation for each student





Delete – Operation

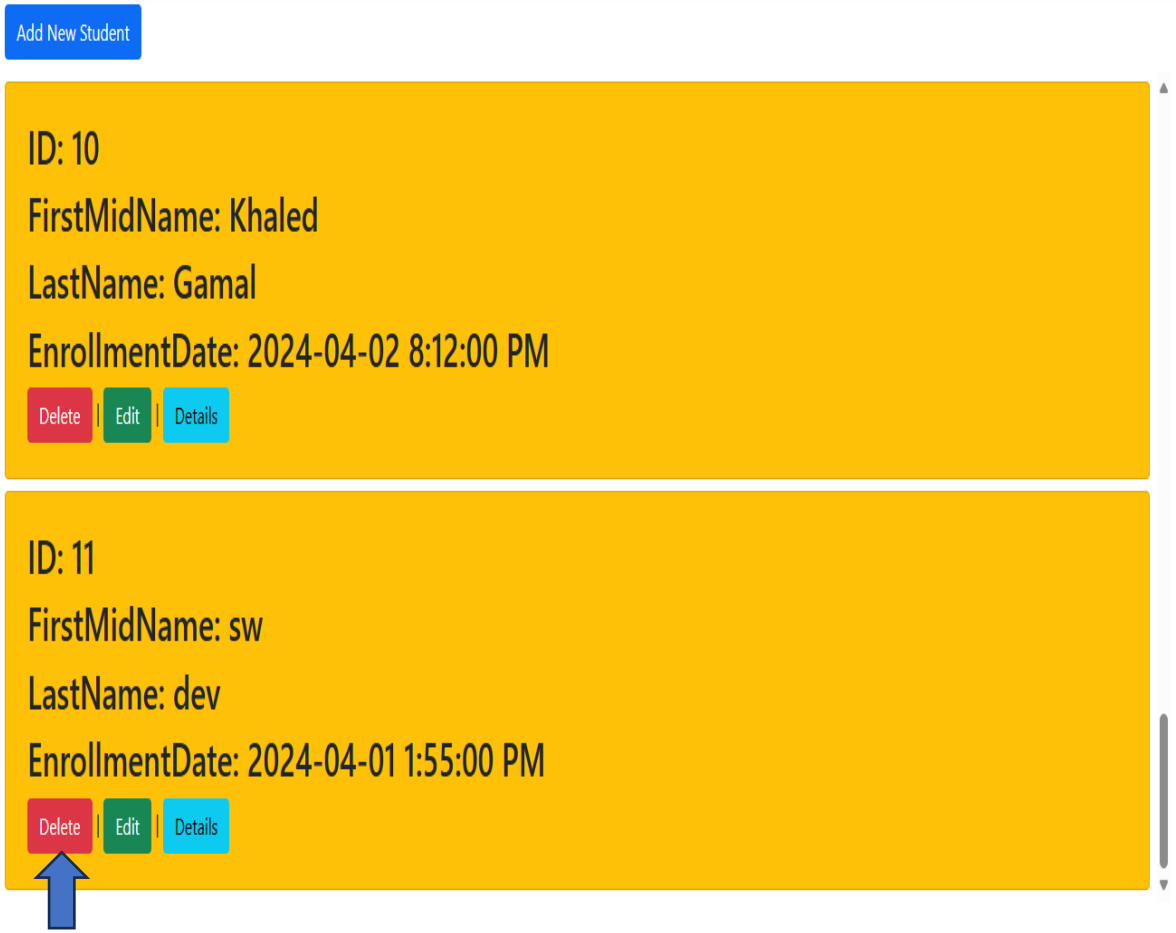
Delete – Operation for each student

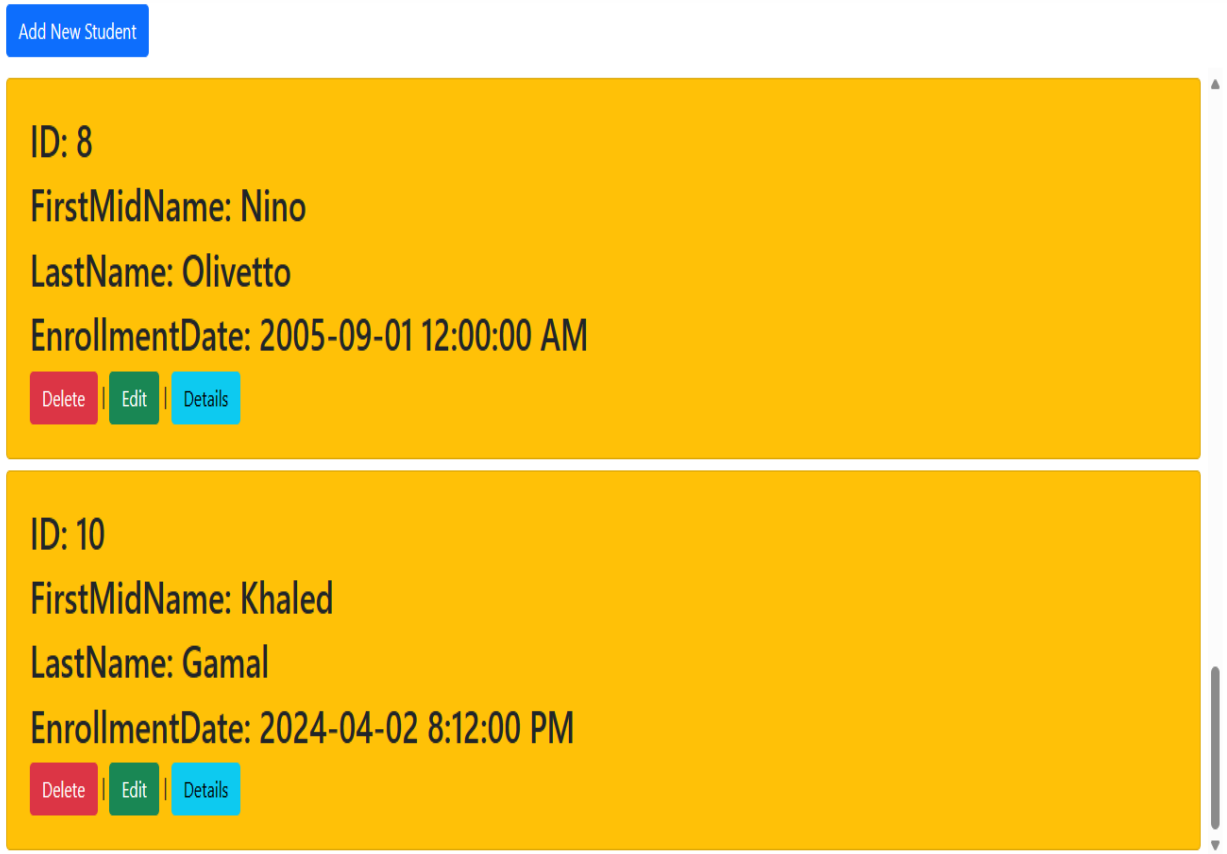
- So lets create an *action* called *Delete* takes one integer parameter called *id* that will remove the student with that ID.

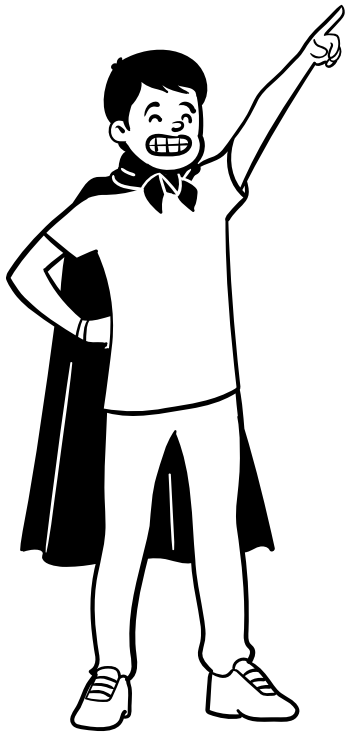
C# (StudentsController.cs)

```
public IActionResult Delete(int id)
{
    Student st = _context.Students.Find(id);
    if (st != null)
    {
        _context.Students.Remove(st);
        _context.SaveChanges();
    }
    return RedirectToAction("Index");
}
```

Delete – Operation for each student







Any Questions?


THANK
YOU!