

CST8256 Web Programming Language I

Lab 5

Objective

1. Use Entity Framework Core to access database.

Due Date

See Brightspace for the due date. To earn 7 points, you are required:

1. Complete the lab as required.
2. Zip your visual studio project folder and submit the zipped file before the due date.
3. Demonstrate your lab work during the lab session after the due date.

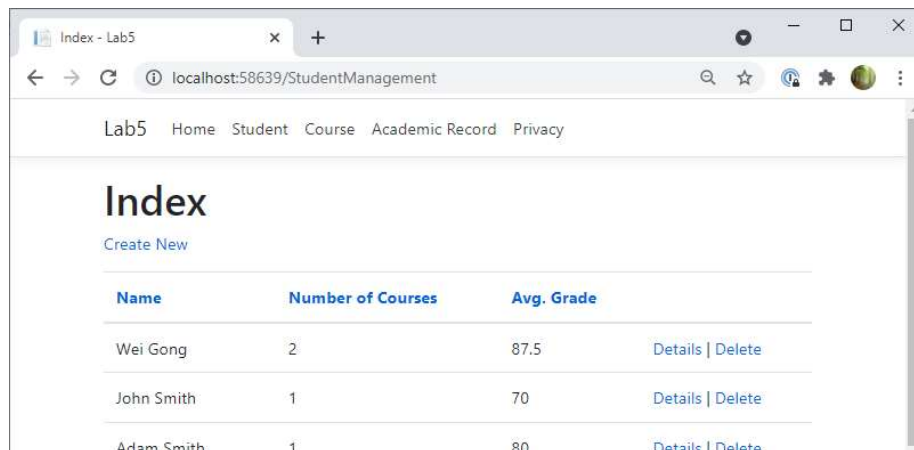
Requirements

This lab continues lab 4 to modify/complete the Visual Studio/Entity Framework generated data access code to provide the desired functions. You can continue work on your lab 4 or make a copy of lab 4 and rename the copy to lab 5.

1. Razor pages for managing students

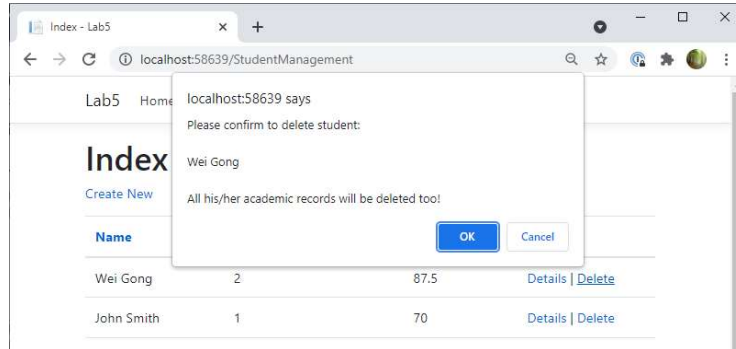
- **StudentManagement/Index**

Usually, the index page for each DB entity should provide some high-level summary about the entity in concern. Modify the generated **StudentManagement/Index** page such that it also shows the number of courses each student took and the average grade the student achieved. The column heads should be clickable for sorting the students by the clicked column, respectively. Also remove the Edit links, there is nothing to edit in our case.



Name	Number of Courses	Avg. Grade	
Wei Gong	2	87.5	Details Delete
John Smith	1	70	Details Delete
Adam Smith	1	80	Details Delete

The generated **Delete** link leads the user to the **Delete** page to get the confirmation from the user. To avoid unnecessary roundtrips to the server-side, the confirmation prompts should happen at the client-side. Change the behavior of the **Delete** link such that when the user clicks the link, a client-side prompt dialog box is shown to get the confirmation from the user, as shown below:

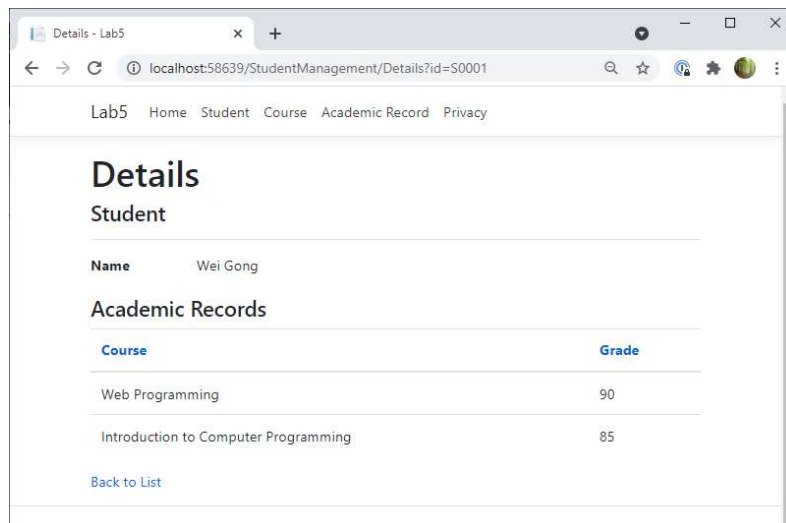


If the user confirms the deletion, the student and all his/her academic records will be deleted (**Note:** you must delete the associated academic records first, and then delete the student. Otherwise, a DB referential integrity error will happen).

Delete **StudentManagement/Delete.cshtml** and its code behind. This page is no longer needed.

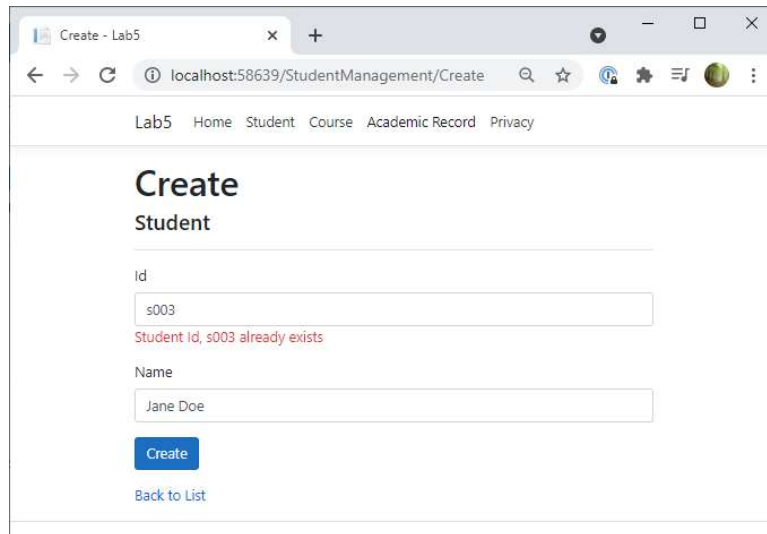
- **StudentManagement/Detail**

Besides the entity's immediate properties, usually more details of the related entities should be shown on this page. Modify **StudentManagement/Detail** page to show the student's academic records as well as the name of the student, as follows. Again the column heads should be clickable to sort the academic records accordingly.



- **StudentManagement/Create**

When saving a new entity into DB, the generated code does not check for the DB's primary key constrain. Add the validation to the page's model class (StudentManagement/Create.cshtml.cs file) such that an error message is displayed if the entered student id already exists in the database Student table as shown below:



The screenshot shows a web browser window with the address bar displaying 'localhost:58639/StudentManagement/Create'. The page has a navigation bar with links: Lab5, Home, Student, Course, Academic Record, and Privacy. The main content area is titled 'Create Student'. It contains two input fields: 'Id' with the value 's003' and 'Name' with the value 'Jane Doe'. Below the 'Id' field, a red error message reads 'Student Id, s003 already exists'. At the bottom of the form, there is a blue 'Create' button and a blue link labeled 'Back to List'.

2. Razor pages for managing courses

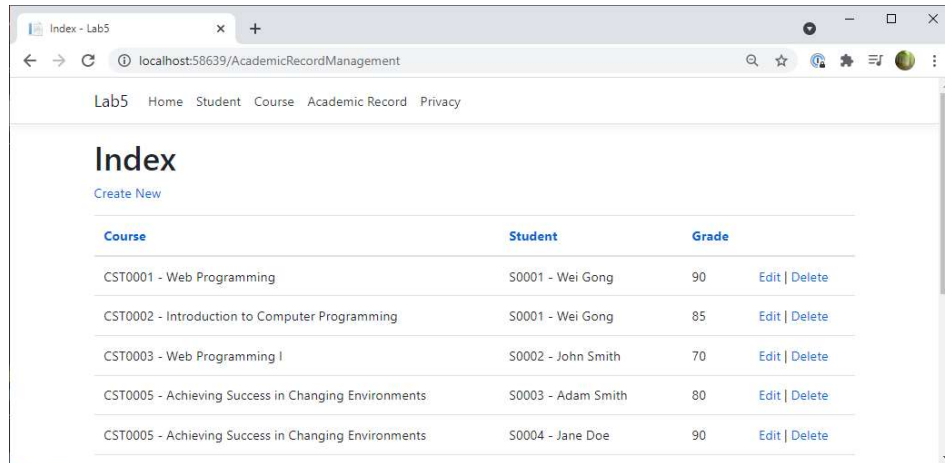
Review the generated Razor pages for managing the course entity. You can modify these pages in the same way as you did with Razor pages for managing the student entity. However, we will not modify any of the pages in this lab.

3. Razor pages for managing Academic Records

Although rudimentary, the generated Razor pages for CRUD accessing to **Strong** entities (a.k.a **Parent** entity, entities which can exist on its own, such as Student and Course) are fully functional. On the other hand, the generated Razor pages for CRUD accessing to **Associative** entities (entities for relating other entities, such as AcademicRecord) are mostly not functional, because the code-gen does not know which properties of the related entities should be displayed/used to complete the desired functionality.

- **AcademicRecordManagement/Index**

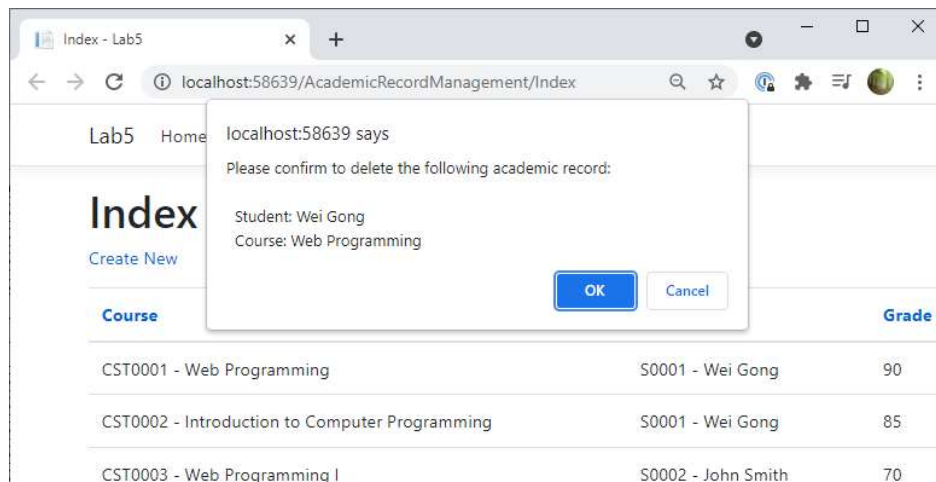
Instead of just showing course code and student id, which are meaningless to the user, modify the page to include the course title and student name for each academic record, as below:



Course	Student	Grade	
CST0001 - Web Programming	S0001 - Wei Gong	90	Edit Delete
CST0002 - Introduction to Computer Programming	S0001 - Wei Gong	85	Edit Delete
CST0003 - Web Programming I	S0002 - John Smith	70	Edit Delete
CST0005 - Achieving Success in Changing Environments	S0003 - Adam Smith	80	Edit Delete
CST0005 - Achieving Success in Changing Environments	S0004 - Jane Doe	90	Edit Delete

The column heads of the academic records table should be clickable. Clicking the Course column head will sort the academic records by course title alphabetically in descending order, Student column head by student name alphabetically in descending order, and Grade column head by grade numerically in descending order.

When the user clicks the **Delete** link, a prompt dialog box will be shown to get the confirmation from the user before deleting the associated academic record as:

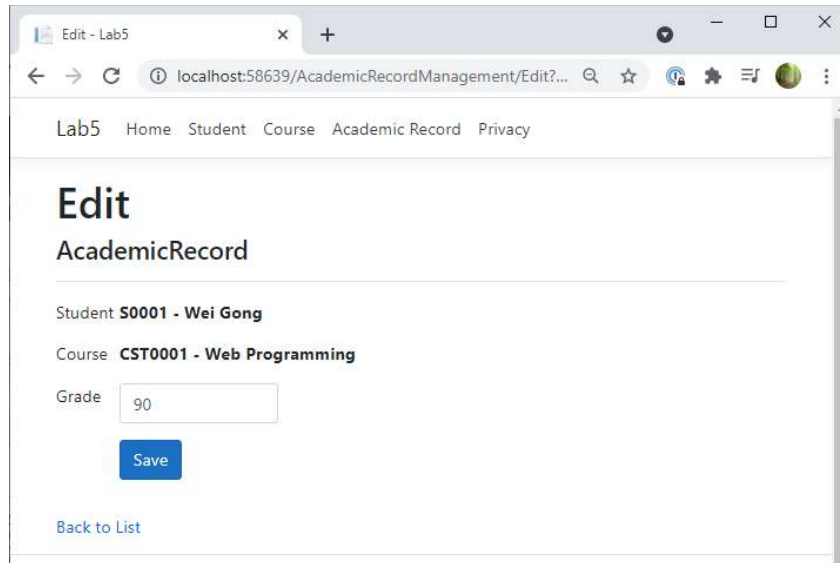


- **AcademicRecordManagement/Details**

Remove this page and Delete links in the Index page. Since there are no more details to show, the Details page is not need.

- **AcademicRecordManagement/Edit**

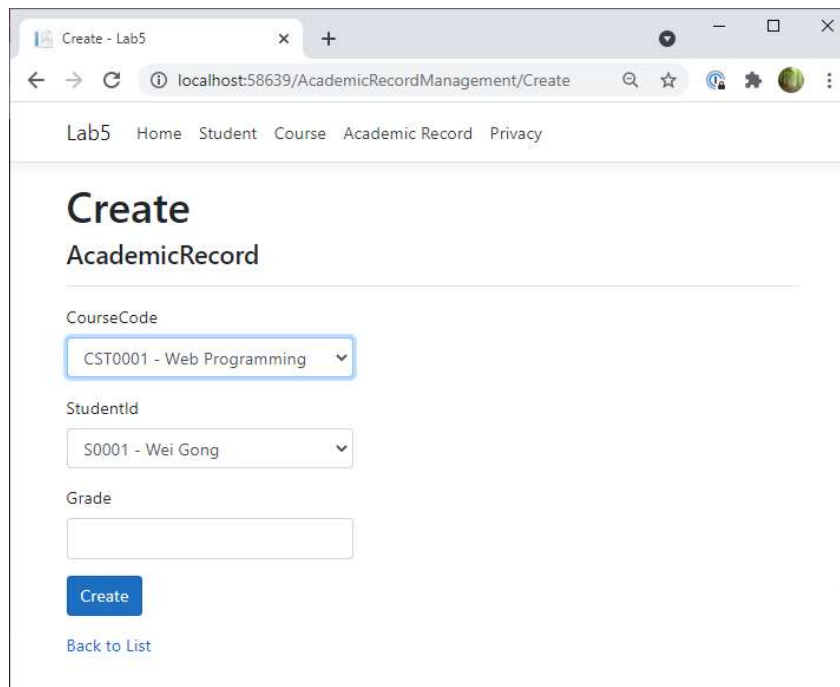
Add the student id, student name, course code and course title to the page so that the user knows which academic record he/she is updating.



The screenshot shows a web browser window with the address bar displaying 'localhost:58639/AcademicRecordManagement/Edit?...'. The page has a navigation bar with links: Lab5, Home, Student, Course, Academic Record, and Privacy. The main heading is 'Edit AcademicRecord'. Below this, the form displays 'Student S0001 - Wei Gong' and 'Course CST0001 - Web Programming'. There is a 'Grade' input field containing the value '90'. A blue 'Save' button is positioned below the grade field. At the bottom left, there is a blue link labeled 'Back to List'.

- **AcademicRecordManagement/Create**

The generated student dropdown list and the course dropdown list on this page only show student id and course code, respectively. They are not very useful to the user. Modify the page such that the student id and the student's name are displayed in the student dropdown list, and the course code and the course title are displayed in the course dropdown list as shown below:



The screenshot shows a web browser window with the address bar displaying 'localhost:58639/AcademicRecordManagement/Create'. The page has a navigation bar with links: Lab5, Home, Student, Course, Academic Record, and Privacy. The main heading is 'Create AcademicRecord'. Below this, the form displays two dropdown menus: 'CourseCode' with the selected value 'CST0001 - Web Programming' and 'StudentId' with the selected value 'S0001 - Wei Gong'. There is an empty 'Grade' input field. A blue 'Create' button is positioned below the grade field. At the bottom left, there is a blue link labeled 'Back to List'.

If an academic record for a given course and a given student already exist in the AcademicRecord table, an error message should be displayed to the user as:

Lab5 Home Student Course Academic Record Privacy

Create AcademicRecord

CourseCode
CST0001 - Web Programming

StudentId
S0001 - Wei Gong

Grade
78

The specified academic record already exist!

Create

[Back to List](#)