HTTP5101 Cumulative Project Part 2

By: Christine Bittle

This cumulative project will have you build a fully operational Create, Read, Update, and Delete web application. Part 2 of this Cumulative project is to build a minimum viable product (MVP) ‘Add’ as well as ‘Delete’ functionality on the provided teachers mysql table using WebAPI and the MVC architecture pattern. This will include:

Type	Description	File
-	A class which connects to your MySQL database	/Models/SchoolDbContext.cs
Controller	A WebAPI Controller which allows you to access information about teachers, add teachers, and remove teachers from the database.	/Controllers/TeacherDataController.cs
Controller	A Controller which allows you to route to dynamic pages	/Controllers/TeacherController.cs
Model	A Model which allows you to represent information about a teacher	/Models/Teacher.cs
View	A View which uses server rendering to display teachers from the MySQL Database	/Views/Teacher/List.cshtml
View	A View which uses server rendering to display a teacher from the MySQL Database	/Views/Teacher/Show.cshtml
View	A View which allows a to input information about a new teacher	/Views/Teacher/New.cshtml

Earn Initiative marks by improving upon the MVP. Some suggestions:

- Use C# Server Side Validation to ensure that there is no missing information when a teacher is added (such as a teacher name)
- Use JavaScript and Client Side Validation to ensure that the user doesn't accidentally submit a form with missing information (such as a teacher name)
- Create documentation describing the server-rendered cycles which happen when a teacher is added or removed from the system (see server_rendered_pages.png in Module 7)
- Maintain referential integrity by making sure that any courses in the classes MySQL table are no longer pointing to a teacher which no longer exists.
- Use JavaScript and AJAX to send an XHR request for adding a teacher (See C# For Web Development Pt10 in Module 8)
- Use JavaScript and AJAX to send an XHR request for removing a teacher (See C# For Web Development Pt10 in Module 8)

5101 Assignment Rubric

Refer to the Code Quality handout in class content and assignment feedback for more details on Qualitative, Quantitative and Semantic code quality dimensions.

	Level 1 (0-25%)	Level 2 (25-50%)	Level 3 (50-75%)	Level 4 (75-100%)
Quantitative	Multiple (4+) Quantitative issues. The project needs significant improvement to meet professional development standards.	Several (2+) Quantitative issues.There are several areas of improvement needed to meet professional development standards.	One Quantitative issue. A few fixes can bring this to professional quality standards.	Zero Quantitative issues. Work is at a professional level. Work is complete, maintainable, scalable, robust, efficient, extensible, and reusable.
Qualitative	Multiple (4+) Qualitative issues. The project needs significant improvement to meet professional development standards.	Several (2+) Qualitative issues. There are several areas of improvement needed to meet professional development standards.	One Qualitative issue. A few fixes can bring this to professional quality standards.	Zero qualitative issues. Work is at a professional level. Work is concise, readable, well-documented, tested, and includes evidence of debugging.
Semantic	Multiple (4+) Semantic issues. The project needs significant improvement to meet professional development standards.	Several (2+) Semantic issues. There are several areas of improvement needed to meet professional development standards.	One Semantic issue. A few fixes can bring this to professional quality standards.	Zero semantic issues. Work is at a professional level. The work achieved is considered and aligned with the context of the project.
Initiative	The content of the work meets the bare minimum requirements.	The content of the work meets the requirements, and there is an attempt to try something new. Not working code is commented out.	The content of the work exceeds the expectations of the assignment. Not working code is commented out.	The content of the work exceeds the expectations of the assignment, and the code runs adequately.