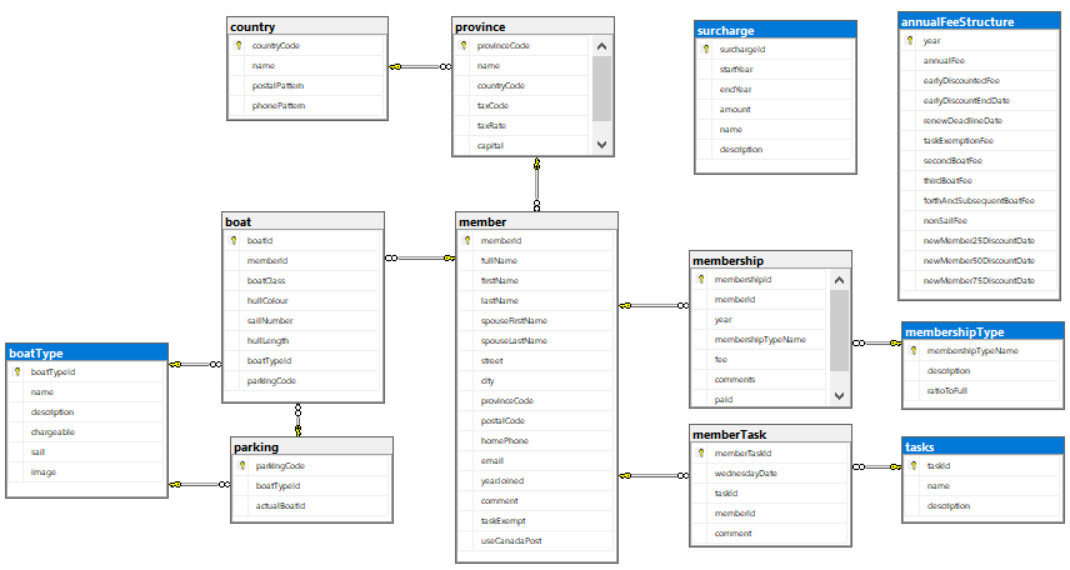
## Ms Web Tech Assignment 1 – MVC Basics

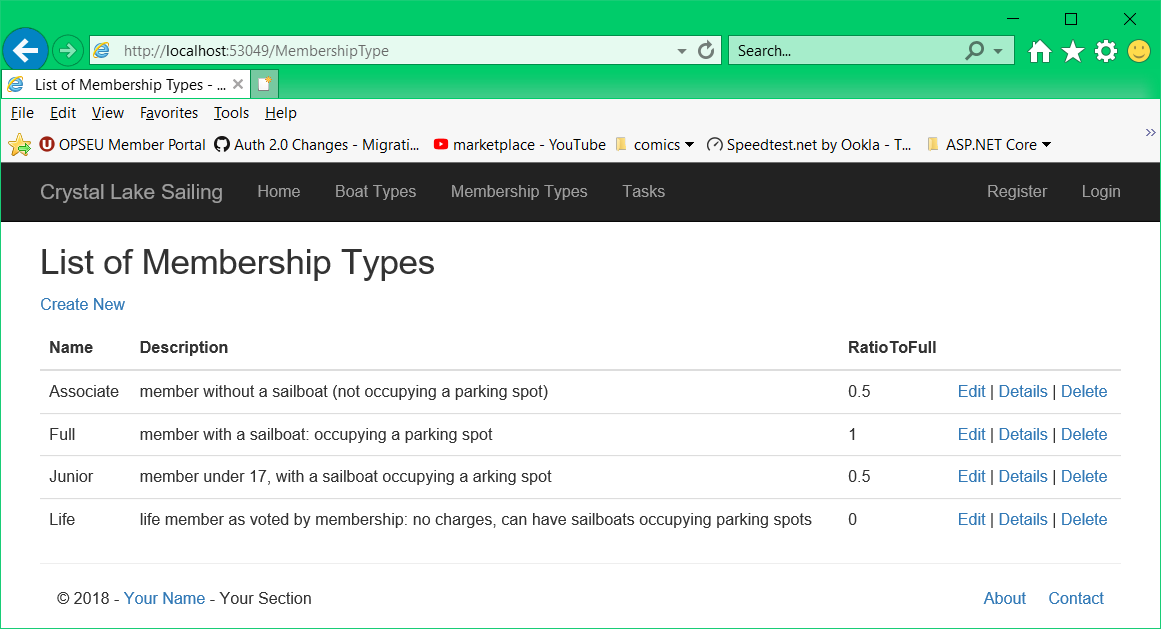
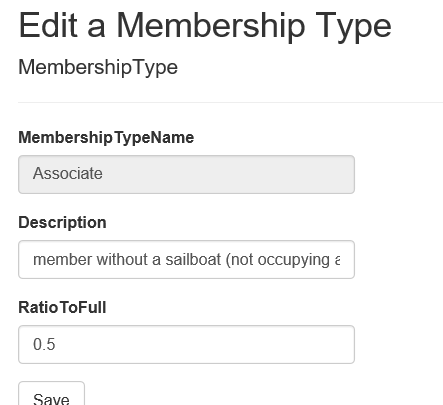
In this assignment, you are working on a site for a sailing club. You need to create the database from a script, generate the models and create views to display and maintain the ***boatType****,* ***tasks***and***membershipType*** tables. In this assignment and all subsequent assignments, whenever you see “XX”, replace it with your initials.



## XXSail project

1. Create an ASP.NET Core project based on .NET Core 2.1 and using the MVC Core technique. Call it ***XXSail*** (where XX are your initials).
   1. Remember to change the authentication to ***Individual User Accounts*** … otherwise, Entity Framework support isn’t preloaded for database access, and is a pain to add later.
2. Load the database to SQL, then generate the models & context for Entity Framework:
   1. Download the ***Sail.sql*** database script from the course site
   2. Use SQL Server Management Studio (SSMS) to execute the script and build the database.
   3. Generate the Entity Framework classes for the database into the project’s *Models* folder. Name the context ***SailContext*** (which is the default for a database named “Sail”).
   4. Add the connection string to ***appsettings.json***, add a service to ***Startup.cs*** to support dependency injection for the context, and ensure there’s a constructor in the context to load the connection string via a DbContextOptions collection.
3. Modify the default layout’s footer:
   1. Put your name and section on the left side of the footer. Your name should be a hyperlink causing the user’s e-mail client to create an email directed to your college e-mail account.
   2. Move the menu’s hyperlinks for “About” and “Contact” to the right side of the footer, retaining their horizontal alignment.

## XXMembershipTypeController



1. Generate a controller called **XXMembershipTypeController** along with the required Views to enable full CRUD maintenance on the *membershipType* table. Add a link to it on the menu.
   1. Normally, keys are not displayed on any page. The key to this table is manually entered (or doesn’t end in “ID”), so Razor provides it as an input on the Create view.
      1. Show the *MembershipTypeName* on all views for this controller, but only allow the user to modify it on the Create view
   2. Ensure all views & hyperlinks function correctly, particularly Edit.
2. Change the browser window title and the heading on the Index view to “List of Membership Types” instead of “Index” … less ambiguity for the user this way.
   1. Change the browser titles and headings on the Details, Create, Edit and Delete views as well, to reference the table being modified (“Edit Membership Type”, etc.).

## XXBoatTypeController

1. Generate a controller called **XXBoatTypeController** along with the required Views to enable full CRUD maintenance on the *boatType* table. Add a link for this controller to the menu.
   1. The description can be up to 255 bytes long, so …
      1. Replace the <input> field for *Description* on the Edit and Create views with a <textarea> displaying 2 rows by default, as shown.
      2. Text areas do not have *tag helpers* (no attributes starting with **asp-**), so you’ll have to use ***@Model.Description*** between the area’s start and end tags to display the *Description* field from the *Model* that was passed to the view.
      3. Ensure the textarea has a name … otherwise it won’t post back to the controller
   2. Again, modify the browser title and page headings on the views to mention the table being displayed or modified.

## XXTasksController

1. Generate a controller called **XXTasksController** along with the required Views to enable full CRUD maintenance on the *tasks* table (the name is plural because “Task” is a C# keyword). Add a link for this controller to the menu.
   1. Again, modify the browser title and page headings on the views to mention the table being displayed or modified.

## All Three Controllers

1. Ensure the controllers, their menu entries, and their actions all work.
2. Comments are required to describe what the program does and what each of its methods do. Generated controllers have 9 methods (actions). At 1% each, that’s a 10% penalty for each controller. Don’t bother commenting Views and Models … they usually break when you do.

## Hand In

1. Zip and upload your project folder to the [D2L Drop-Box](https://www.econestoga.ca) for this assignment.
2. Print and hand in [the marking sheet](http://econestoga.ca) (Content 🡪 assignments) with your name, section & your instructor’s name on it.