## INT422 Assignment 10

JavaScript and Ajax. Web services. Bulk load data.

Read/skim **all** of this document before you begin work.

## Due date

Section A, B & C: not required

*If you wish to submit the lab before the due date and time, you can do that.*

## Objective(s)

There are several objectives for this Assignment 10:

* Continue using security principles
* Working with JavaScript in a web app
* Working with Ajax to consume content and/or web service resources
* Bulk load data (initial or ongoing) into the data model of the web app
* Post to a public web server

## Introduction and overview

You can choose the problem domain and therefore the data that you’ll use for this final assignment of INT422.

Too broad? Here’s some guidance:

Select a problem domain that has *easy-to-obtain data*. There are many areas of human activity that can be used. Maybe we can suggest, in the entertainment domain, **sports** of some kind. Statistics are easy to obtain, as well as data about players, teams, and so on. Publicly available on the web, with data that’s easy to copy, paste, and use. Non-text media types (e.g. photos, video) are also easily available. For example:

* NHL
* MLB
* NFL
* NBA
* MLS
* NASCAR
* Olympic-themed sports
* eSports
* EPL (English Premier League, football)
* IPL (Indian Premier League, cricket)
* etc.

That’s not the only area of human activity that would work, however. Other forms of entertainment – for example screen-displayed content (i.e. traditionally known as movies and television, but now much broader, to include cable/satellite, online, streaming, on-demand, and so on) could work.

Almost anything could work. Anything human activity that’s competitive or measured/tracked, and generates data. Keep it clean and family-friendly however.

**What we’re looking for, in general**

For this assignment, you must create a data model that has about three (3) entities. No less than three, and no more than five. Don’t make the data model too complicated. In your data model, at least two of the entities must be associated (probably a one-to-many).

During the first week of the assignment’s lifetime, you will define and build the basic structure of the app. This will include design model classes, a *small amount* of initial data (programmatically or by using an HTML Form), view model classes, manager methods for “get all” and “get one”, and controllers for each entity that will work with “get all” and “get one”.

The first-week work must also include at least something that uses JavaScript or Ajax. Think again of their value propositions, and come up with something that would work.

Finally, the home page for your first-week work will briefly describe what your project is about.

Then, after you learn about bulk data loading, you can add that feature to the web app. You’ll use that knowledge to capture, prepare, and upload *larger amounts* of data to your web app, to complete its data model coverage. For example, during the first week, you may decide to programmatically add data for one “team” and two or three “players”. During the second week, you can bulk load all teams, and all players.

## Specifications overview and work plan

Here’s a brief list of specifications that you must implement:

* Follows best practices
* Implements the recommended system design guidance
* Customized appearance, with appropriate menu items
* Customized security claims configuration
* Publish the app to the web

During the Friday-or-Monday class/session, your professor will help you *get started* and *make progress* on this assignment. There will be a small *in-class grading* task  done during the class/session.

## Create the project, based on the project template

Create a new web app, named Assignment10.

It MUST use the “Web app project **v4″** project template. Before using it, you must install it into your Visual Studio configuration.

After creating the web app, customize the home page. Change the large “**Learn more >>**” button to **“Assignment 10 on Azure**” and set the button link to the URL of the assignment 10 on Azure.

Build/compile the app, to refresh the packages. Do NOT run the app yet.

If you wish, temporarily disable error-handling for HTTP errors 500 and higher:

* Open the Global.asax.cs source code file.
* In the Application\_EndRequest() method, comment out the “if (code >= 500)…” statement.

Warning: Your teachers believe that the best way to work through this assignment is to do itincrementally. Get one thing working, before moving on to the next. Test each part.

## Customize the app’s appearance

You will customize the appearance all of your web apps and assignments. Never submit an assignment that has the generic auto-generated text content. Make the time to customize the web app’s appearance.

For this assignment, you can defer this customization work until later. Come back to it at any time, and complete it before you submit your work.

Follow the guidance from [Assignment 1](https://petermcintyre.com/bti420/graded-work/assign1/) to customize the app’s appearance.

**Important:**

The web app’s “home” page MUST include a brief description of the app’s purpose and features.

Each student will be implementing their own custom idea, so you MUST tell the view (and your professor!) about the web app.

## Improve the role claims configuration

As you have done a couple of times now, configure your own role claim names in the app.

## Design your data model, data classes, and make some initial data

At this point, you can design and code your data model. Obviously, you will need design model classes, DbSet<TEntity> properties, view model classes, and AutoMapper maps.

Make sure that there is easily-available (and/or easily-creatable) data for the entities in your data model. Remember from above, a first-week task is to create a small amount of data, and a second-week task is to bulk load more data into your model. Make sure that the data model design closely matches the source(s) of data.

## Add more data using a bulk data-loading process

Recently, you learned how to load data into your web app’s data store, from external sources that could include CSV, Excel workbooks, and web services.

At this point, implement a data-loading process, that will load more data.

How much data?

In total, your app should have hundreds or thousands of objects in its data store. If your  data model includes two associated entities in a one-to-many association, it’s possible that the root end would have tens or hundreds of objects, and the to-many end would have hundreds or thousands. That’s the scale we’re looking for.

## JavaScript and/or Ajax in your web app

Recently, you learned how to use JavaScript and Ajax with ASP.NET MVC web apps.

At this point, implement some JavaScript and/or Ajax. The functionality should be either *documented* or *discoverable*. Make sure the user (and the professor who marks your work) knows about it and can benefit from it.

## Publish to Azure

Follow the guidance in this document to deploy/publish your web app to Azure:

[Web apps – deploy to Microsoft Azure](https://scs.senecac.on.ca/~wei.song/int422/assignments/INT422-Assignment7.html" \t "_blank)

Please note that you must NOT create a new database server. Instead, create a new SQL Database, inside the existing database server that was created for your previous assignment.

Suggested names, assuming that your Microsoft Account name uses the recommended format “pmcintyr-wa2016”:

Web app: wsong18-wa2017a10

SQL database: Assign10Store (or A10Store)

## Testing your work

While designing and coding your web app, use the Visual Studio debugger to test your algorithms, and inspect the data that you are working with.

In a browser, test your work, by doing tasks that fulfill the use cases in the specifications.

## Reminder about academic honesty

You must comply with the College’s academic honesty policy. Although you may interact and collaborate with others, *you must submit your own work*.

## Submitting your work

Here’s how to submit your work, before the due date and time:

1. Locate the folder that holds your solution files. In Solution Explorer, right-click the “Solution” item, and choose “Open Folder in File Explorer”. It has three  (or more) items: a Visual Studio Solution file, a folder that has your project’s source code, and a “packages” folder. Go UP one level.

2. Make a copy of the folder. This is the version that you will be uploading.

3. Remove the “packages” folder from the copied folder; also, remove the “bin” and “obj” folders.

4. Compress/zip the copied folder. The zip file SHOULD be about 2MB or less in size. If it isn’t, you haven’t followed the instructions properly.

5. Login to My.Seneca/Blackboard. Open the Web Programming on Windows course area. Click the “Assignments” link on the left-side navigator. Follow the link for this lab. Submit/upload your zip file. The page will accept three submissions, so if you upload, then decide to fix something and upload again, you can do so.