Web Tech Assignment 1

Assignment 1

This assignment explores some of the different methods to incorporating dynamic content into a website. This document is for all students regardless of their study mode and the assignment **must be done individually**. This **document specifies the minimum requirements** to get a passing grade – the more effort you put in the better the outcome!

Each person is to complete the Assignment 1 that utilises the SAPOL Expiation database. There are several tasks to complete as outlined below

1. An Expiation Code View with Details View
2. A Regions View with Details View

## Download the Assignment Web Application

The web application has been pre-built complete with the Expiation Database model. You will need to download the CreateExpiationsDB script, generate the database and then update the connection string in your web application to use your copy of the Expiations database.

## Create a new Controllers with Actions and Views

You will need to create new controllers complete with new Actions and Views. Your new **Controllers will need to support the standard Index and Details Views associated with the Model View Controller design pattern.**

# Tasks

## Expiation Code List [25%]

This Index View allows users to see a list of **expiation codes**, **their descriptions**

and to search for specific codes/descriptions using words like “Exceed” and “fail”. **(Based on the descriptions)**

If the code/description starts with the search text, then it should appear 1st in the match, otherwise if it contains the search text it should appear second.

When the page first loads, the codes and descriptions should be in order of the Expiation Code.

The page provides an [auto-complete](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/datalist) search text-box that shows a list of matching expiation codes as the user enters search terms. This View must:

* Provide the [auto-complete](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/datalist) expiation code match using @razor
* The View must **initially load showing the expiation codes and descriptions in alphabetical order**
* The View must use model display properties (such as display name) where appropriate
* The View must only utilise a model/ViewModel for all data displayed and any search data. The ViewBag/ViewData object can only be used for setting the page title.

(**This means you cannot use the ViewBag/ViewData object for the auto-complete list etc.)**

* The View must make use of the appropriate ASP Tag Helpers and only use HTML helpers for displaying model property names.
* Hovering over a row in the table/card showing the expiation code + description highlights the row/card
* Clicking on the row/card opens a Detail page **in a new tab** with the correct expiation data showing

(requires JavaScript) -- Done

* + The Expiation Code selected
  + The Expiation Description

**Note**: Các yêu cầu cần có:

* Các mã code và descriptions load theo thứ tự của Expiation code (EC) which sẽ load theo thứ tự chữ cái. --OK
* Sử dụng model/ViewModel để load và tìm kiếm data. --OK
* Tìm kiếm code bằng key words như là fail hoặc exceed. --OK
* Dùng CSS để làm hover -- OK
* Click on thẻ card sẽ mở Detail Page ở tab mới. (Dùng JS)--OK
* Search theo first và second match –OK
* **DataList** – OK

For full marks, your data needs to be presented aesthetically using bootstrap classes (the more effort and research into the styles the better the mark). The query needs to return the number of expiations for each expiation code, for the current year. At the top of your table/cards you need to show the number of matching expiations to date. Given the amount of data, you need to make sure this count is efficient as possible. You may find the following links useful:

* [Efficient Querying](https://docs.microsoft.com/en-us/ef/core/performance/efficient-querying)
* [Complex Query Operators](https://docs.microsoft.com/en-us/ef/core/querying/complex-query-operators)
* [IQueriable Methods (Average, Sum etc)](https://docs.microsoft.com/en-us/dotnet/api/system.linq.queryable?view=net-5.0)
* [LinqPad](https://www.linqpad.net/)

## Expiation Codes Detail [25%]

Selecting an Expiation Code on the previous page navigates to this page which clearly shows **the Expiation Code selected**,

the **description** and **any other detail** for the selected expiation code.

In addition, the page should have a drop-down list that defaults to the current year. This can be programmed to show the current year minus 2 (ie, have the options of selecting 2020, 2021, 2022).

You need to make sure the current year is generated from **the server date**, and then the other years calculated from that (so next year it defaults to 2021, 2022, 2023).

The page should load showing a **monthly breakdown of expiations counts for the selected expiation code**, year and grouped by the Notice Status Description

Each Month name should appear only once in the table/Card and then each status description with count under that month. Clearly colour the Month rows and apply [bootstrap styles](https://getbootstrap.com/docs/5.0/content/tables/) to the notice status descriptions so the same status can be easily identified for each month (e.g. Withdrawn is always a pale red background etc).

Each month should be accompanied by a Total number of expiations matching the selected expiation code for the month (irrelevant of status). At the bottom of the page, the total number of expiations matching the selected code for the year-to-date

As with the other Views, the Expiation Code Details View:

* Must only load if an expiation code is passed from the previous Index View
* Must make use of an appropriate ViewModel and **not use any ViewBag/ViewData objects except for the page title**
* The View must use model display properties (such as display name) where appropriate
* Make use of efficient queries where possible

**Example: Expiation Code Detail** showing grouping **by Month and Notice Status Description**.

|  |  |  |
| --- | --- | --- |
| **Month** | **Notice Status Description** | **Status Count** |
| **January** |  | Total: 618 |
|  | COURTS ENFORCEMENT (pending or enforced) | 31 |
|  | COURTS RELIEF STATUS | 47 |
|  | EXPIATED | 131 |
|  | ISSUED OR CAUTIONED | 401 |
|  | WITHDRAWN | 8 |
| **February** |  | Total: 579 |
|  | COURTS ENFORCEMENT (pending or enforced) | 19 |
|  | COURTS RELIEF STATUS | 43 |
|  | EXPIATED | 111 |
|  | ISSUED OR CAUTIONED | 398 |
|  | WITHDRAWN | 8 |
|  |  | Year To Date: 1,197 |

Note: Các việc yêu cầu cần làm:

* Chọn những code từ trang trước qua trang mới. --OK
* Làm drop Down list và tải data vào (để đảm bảo năm hiện tại được tạo từ ngày máy chủ) --OK
* Each month should be accompanied by a Total number of expiations matching the selected expiation code for the month (irrelevant of status). At the bottom of the page, the total number of expiations matching the selected code for the year-to-date --OK
* Each Month name should appear only once in the table/Card and then each status description with count under that month. The page should load showing a monthly breakdown of expiations counts for the selected expiation code, year and grouped by the Notice Status Description and Month. --OK
* Clearly colour the Month rows and apply [bootstrap styles](https://getbootstrap.com/docs/5.0/content/tables/) to the notice status descriptions so the same status can be easily identified for each month (e.g. Withdrawn is always a pale red background etc). --OK

## Region Expiations List [20%]

This View shows a List of Regions (in alphabetical order) and the total number of expiations for each region with the option to filter by the selected year. As with the previous View, this page must make professional use of bootstrap and:

* Contain a drop-down list that defaults to the current year. This can be programmed to show the current year minus 2 (ie, have the options of selecting 2020, 2021, 2022). You need to make sure the current year is generated from the server date, and then the other years calculated from that (so next year it defaults to 2021, 2022, 2023) --OK
* **Clicking on a Region name** opens a new tab that navigates the user to **a Details view** that **shows the list of Expiation Codes, Descriptions** and total number of expiations of that type for the selected region and year.

## Region Expiations Detail [20%]

This is the equivalent of the Expiation Codes Detail View except you are displaying the number of expiations for each Expiation Code in the selected region and year. The page must utilise bootstrap styles and load showing:

* The Selected Region
* The Selected Year
* The total number of expiations
* A list showing each Expiation Code, Description and total number of expiations for that code
* Order the results by Expiation Code.

## HTML Validation (10%)

Your pages will need to generate valid HTML. We suggest you use the [W3C HTML validator](https://validator.w3.org) to check this! You need to use appropriate class names, bootstrap classes where possible and avoid using IDs where you can.

# Overall Requirements

For server-side queries you must use Entity Framework LINQ or Lambda expressions – no SQL statements are permitted.

Any new Model/ViewModel classes you create need to be placed in the **ViewModels** folder.

You should have minimal custom CSS styles – use [Bootstrap V5](https://getbootstrap.com/docs/5.0/getting-started/introduction/) as your starting point. Minimise any CSS you write and if included it must be done in the site.css file or create a separate [render-section](https://docs.microsoft.com/en-us/aspnet/core/mvc/views/layout?view=aspnetcore-6.0#sections).

Once you have completed your assignment, zip up your entire project (sln file, project folder with all files) and submit it to the course website.

# Presentation

After the due date, each student will participate in a viva to discuss their individual work. You must complete the viva in order to get a passing grade for this assignment.