**Saint Martin’s University**

Knights of Columbus Website Update

Documentation and Notes as a General Guide

# CSC482: Senior Project II

**Dr. Harold Nelson**

Spring Semester 2023

Brandon Essary

Sunya Chay

William Flores

**Table of Contents**

**Contact Information………………………………………………………………………………..2**

**Website Code………………………………………………………………………………………...2**

**How to Link Views/cshtml Files in ASP.NET……………………………………………….....2**

**How to Link Static Files…………………………………………………………………………...2**

**How to Create a View/cshtml File in ASP.NET……………………………………………….2**

**Basic HTML Starting Guide……………………………………………………………………...2**

**How the Folders are Setup………………………………………………………………………..4**

**Navigation Bar……………………………………………………………………………………....4**

**Dropdown Menu…………………………………………………………………………………...4**

**Calendar Database…………………………………………………………………………………5**

**Database……………………………………………………………………………………………..5**

**Filtered Tables……………………………………………………………………………………...10**

**BootStrap Implementation……………………………………………………………………....13**

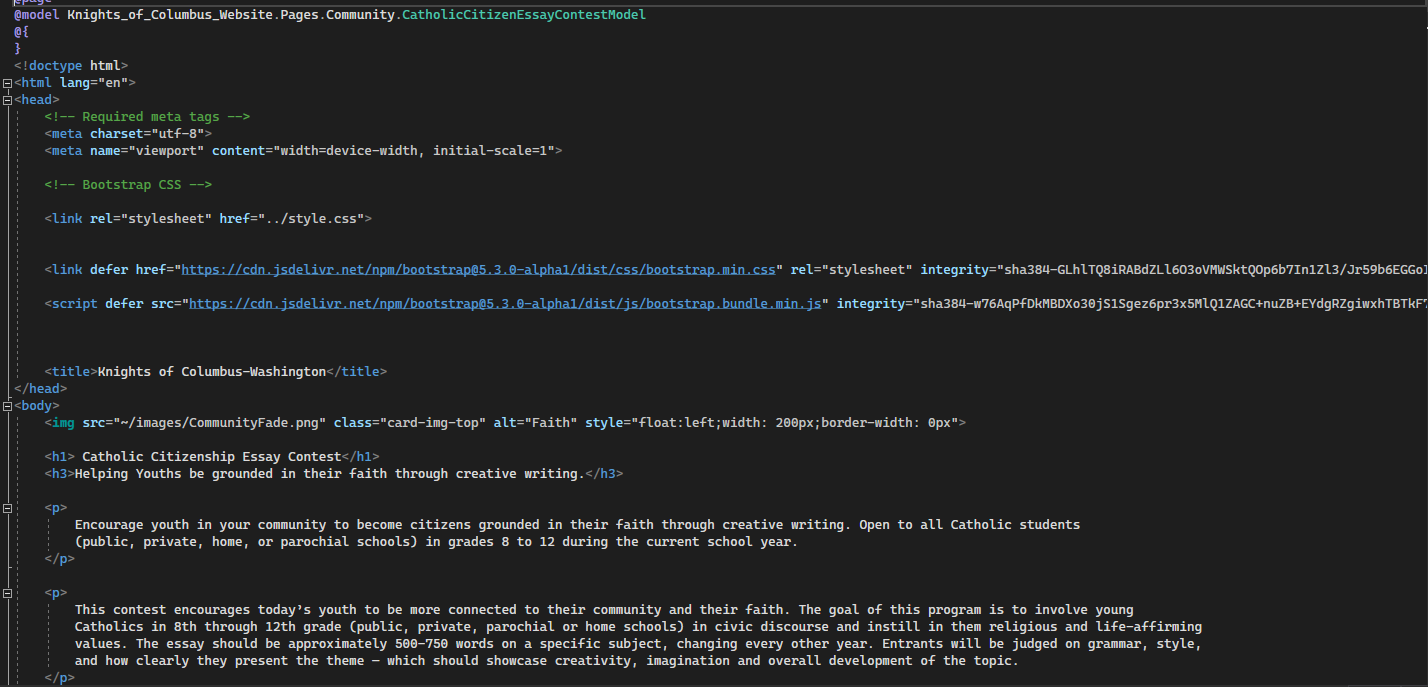
**Contact Information;**

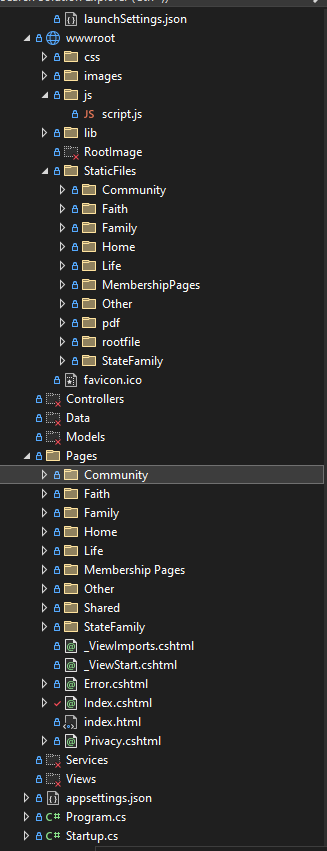
* **Brandon Essary**
  + Email: essaryb00@gmail.com
* **SC Chay**
  + Email: sunyachay@gmail.com
  + Discord: sunyx.chxy
* **William Flores**
  + Email: william.flores@stmartin.edu, william.mflores@yahoo.com

**Website Code:**

The current website is sitting in an ASP.NET MVC. The code comprises multiple pages, all organized in their respective folders. Almost all of the pages are linked to the main page or are linked to the “\_Layout” file.

* **How to link pages together in ASP.NET:**
  + For views/ cshtml files:
    - ~/main folder/ folder/example page.cshtml
      * Example: <a href “~Pages/Faith/RSVP”>RSVP\*</a>
    - **OR** ~/folder/example page.cshtml
      * Example: <a href="~/Faith/RSVP">RSVP\*</a> This one is preferred
  + **For HTML/static files:**
    - ~/mainfolder/folder/examplepage.html
      * Example: <a class="dropdown-item" href="~/StaticFiles/Home/Six Months Ltr 202301\_20230128.pdf">Next Six Months</a>
  + All static files and images are located in the wwwroot.
* **How to create a view/cshtml file in ASP.NET:**
  + Select the folder in the “Pages” folder that you want to add the cshtml file.
  + Right-click the folder and hover over “Add”
  + Select “Razor Page”
  + Click on “Add”
  + Rename the file to the name that you want and select “Add”
* **Basic HTML starting guide:**
  + <!DOCTYPE HTML> Starts the HTML code
  + <head></head> Where to put all of the bootstrap codes and the name of the page.
  + <body> </body> Where the main part of the code goes in.
  + <img src=””/> Linking an image
  + <h1> </h1> Header
  + <p> </p> Text
  + <link href=””/> Linking an css file, link, or another page.
  + EXAMPLE PSUEDOCODE:
    - <!DOCTYPE HTML>
    - <head>
    - Bootstrap links and style sheet go here!
    - Example page title
    - </head>
    - <body>
      * <h1> Example Header</h1>
      * <p>Example text</p>
      * <script src=””>JavaScript and Bootstrap links go here!</script>
    - </body>
    - </HTML>

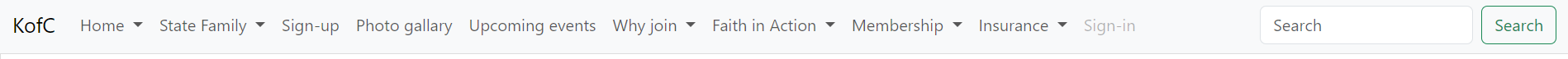


* **How the folders are set up**
  + The folders are set up by each of the four main groupings: Faith, Community, Family, and Life. Other folders contain other important documentation and cshtml files. For any files that are a pdf, those are located in the wwwroot under “StaticFiles,” “pdf.” From there, you can find all of the pdf and HTML versions of the cshtml files. Images are also stored in the wwwroot as well. The files that mainly live in the wwwroot are the static files, images, stylesheets, and the codes for Bootstrap5.

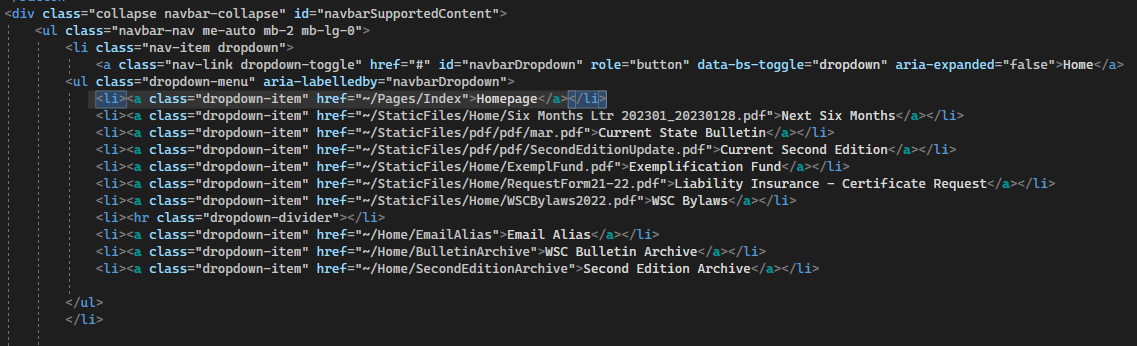
**Navigation Bar:**

The current version of Bootstrap, bootstrap5 does not support multi-level dropdown menus, to get around this problem we created CSS classes that would enable a multi-level dropdown to work. Unfortunately however moving into asp.net the navbar is unable to read the CSS file and so the multi-level dropdown remains inoperational for now. The navigation bar code lives in the “\_Layout.cshtml” file under the “Pages/Shared” folders. Some things that do need to be worked on is being able to link back to the Homepage using the KofC button in the navigation bar, and being able to use a second dropdown menu from the first one. An example of where a second set of navigation items appear is in the “Faith in Action” section in the navigation bar.

Navigation Bar

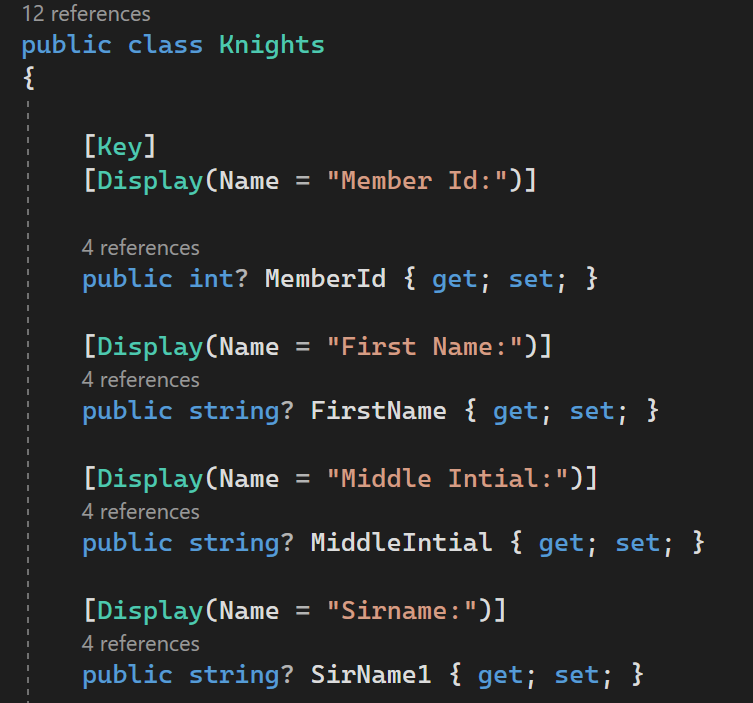
****

**Dropdown menu:**

The dropdown menu in the navigation bar uses the <li> or list function to create a dropdown menu. To put pages into the deopdown menu, you would do: “ <li><a class="dropdown-item" href="~/Pages/Index">Homepage</a></li>.” The <li> command is always in the <ul> command.

**Calendar Database:**

The calendar was coded using javascript. It was a requirement of the customer for the calendar to be able to track Knight of Columbus meetings at the state, district and local level. At this time, the best use of the calendar will be to inform site visitors of upcoming meetings for only one of the aforementioned areas.Current calendar information is stored locally. For the future group, they will need to change the coding in the javascript to send data to the database and expand the functionality of the calendar to cover the customers needs. The current code for the calendar database is stored in “\_Layout.cshtml” file under “Shared” in the “Pages” folder.

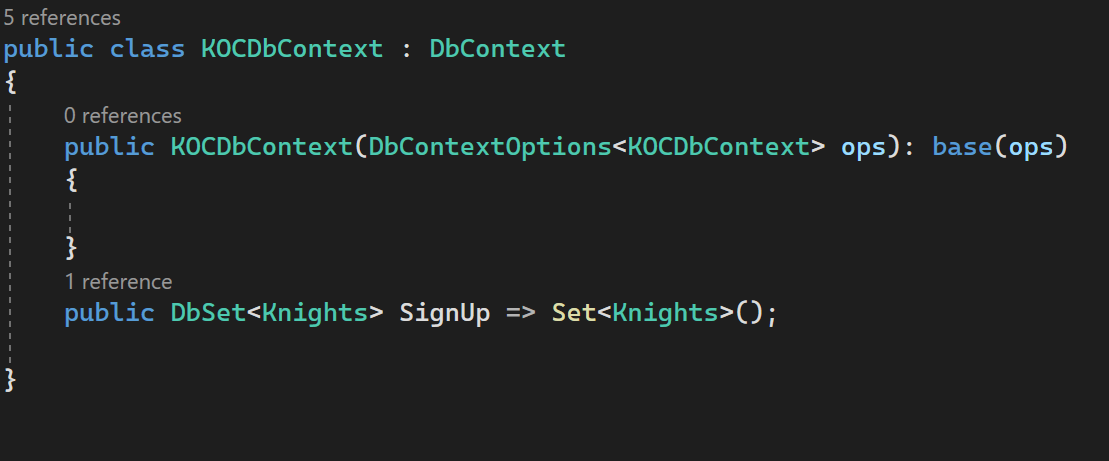


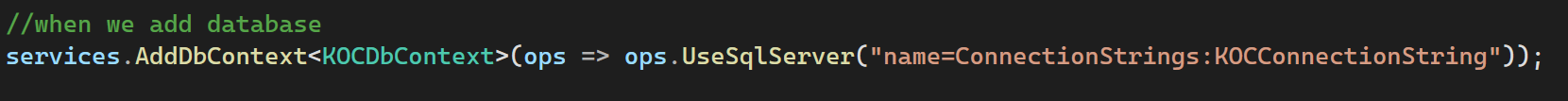
**Database:**

For this project the group decided to use Microsoft SQL Server Express. There are other types of databases that a future group can use with ASP.net. The reason for this choice was for ease of use for encryption.

For this database starts in the Models folder with “Knights.cs”. The “Knights.cs” model is the template for the table “SignUp” in the database and for the “SignUp.cshtlm” page for the website. The column names for the “SignUp” table are linked to the “Knights.cs” The database column names are “public string? EmailAddress { get; set; }”and the “SignUp.cshtlm” are the display name.

Next we created “KOCDbContext.cs” found in the Data folder.



After the “DbContext” was created in the Data folder we added the following service. It is here that a future group can change the database from SQL server to another server of their choosing. However, any table that was manually created, i.e. the current Members table, will need to be recreated in that database. 

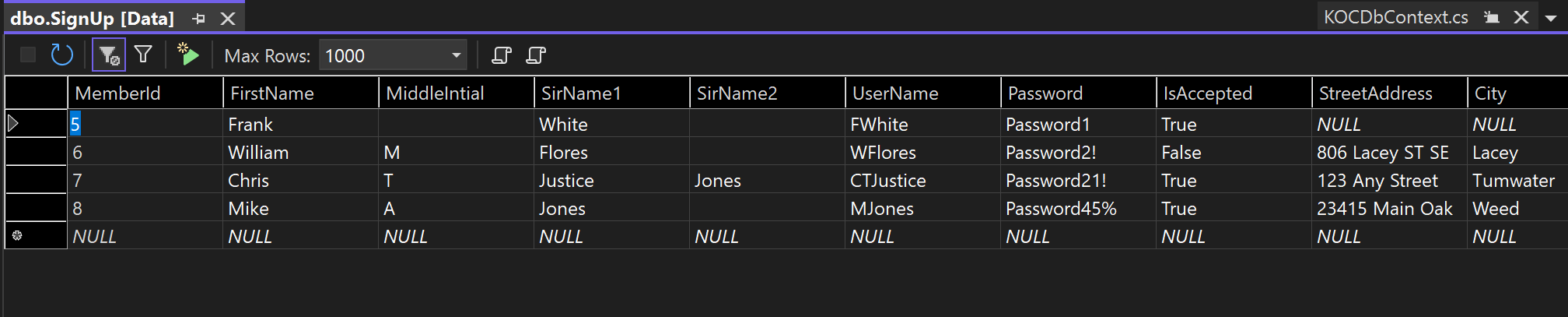
After this code is successfully entered Visual Studio will prompt the user to download SQL Server Express in “Connected Services”.

In the “Secrets.json(local)” file under “Connected Services” the connection string will read :

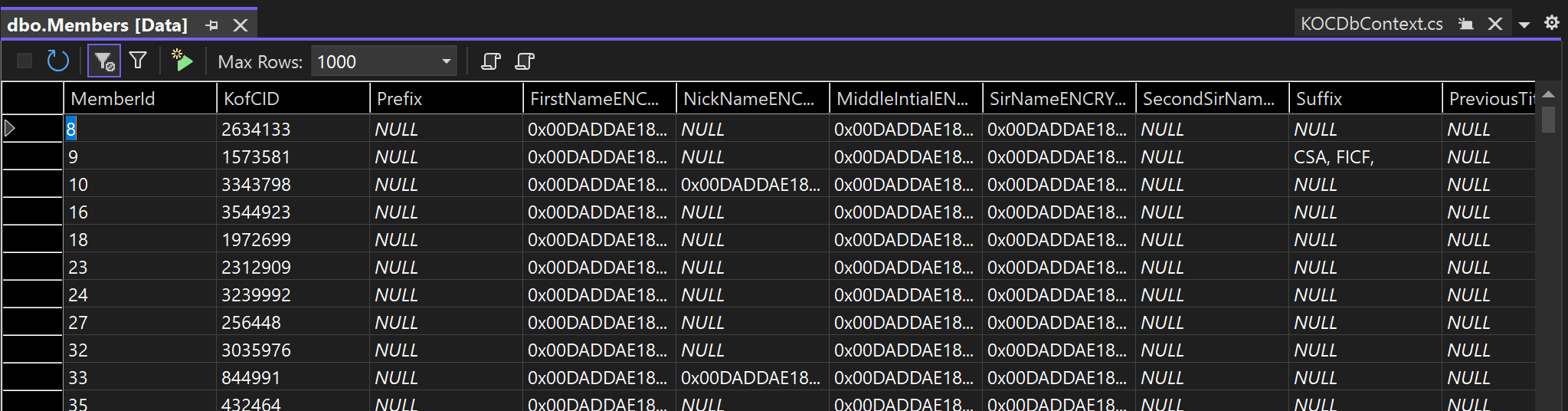
“"ConnectionStrings:KOCConnectionString": "Server=(localdb)\\mssqllocaldb;Database=aspnet-53bc9b9d-9d6a-45d4-8429-2a2761773502;Trusted\_Connection=True;MultipleActiveResultSets=true"”.

If future users want to use SQL Server on their local machines or to another online machine, this is where the connection string will need to be changed.

This is how the current “SignUp” and “Members” tables appear in the database.



The current “SignUp” table is for testing purposes and will need to be cleared before turning the project over to the customer or before the project goes live.



This is the Members table and is filled with encrypted member data. The current list has one thousand entries and should be the final format, ready for the entire members list.

The first step into incorporating the member list is to make a working copy of the excel file and delete unwanted rows. These rows are, for example, “FirstNameUpdated”. This would be the best time to add the encrypted rows to the working copy excel workbook.



After all rows are created to the customers liking, save the excel workbook as a CSV file.

The team created the Members table taking the information from the CSV file to create the columns of the table. To accomplish this the members table was manually inserted into the current database using the new query function with the following code.

create table dbo.Members

(

MemberId int,

KofCID int null,

Prefix varchar (10) null,

FirstName varchar (50) not null,

FirstNameENCRYPTED varbinary(max) null,

NickName varchar (50) null,

NickNameENCRYPTED varbinary(max) null,

MiddleIntial varchar (10) not null,

MiddleIntialENCRYPTED varbinary(max) null,

SirName varchar (50) not null,

SirNameENCRYPTED varbinary (max) null,

SecondSirName varchar (50) null,

SecondSirNameENCRYPTED varbinary (max) null,

Suffix varchar (25) null,

PreviousTitles varchar (255) null,

AddInfo1 varchar (100) null,

AddInfo1ENCRYPTED varbinary (max) null,

Address varchar (255) null,

AddressENCCRYPTED varbinary (max) null,

City varchar (50) null,

State varchar (2) null,

PostalCode varchar (10) null,

Phone varchar (14) null,

PhoneENCRYPTED varbinary (max) null,

WifesName varchar (50) null,

WifesNameENCRYPTED varbinary (max) null,

AddInfo2 varchar (100) null,

AddInfo2ENCRYPTED varbinary (max) null,

FaxNumber varchar (14) null,

FaxNumberENCRYPTED varbinary (max) null,

Council int not null,

Assembly int null,

Circle int null,

Email varchar (50) null,

EmailENCRYPTED varbinary (max) null,

Deceased bit,

CellPhone varchar (14) null,

CellPhoneENCRYPTED varbinary (max) null,

LastUpdated datetime null,

Bulletin bit,

UserID varchar (50) null,

UserIDENCRYPTED varbinary (max) null,

Data varchar (50) null,

DataENCRYPTED varbinary (max) null,

DataChanged bit,

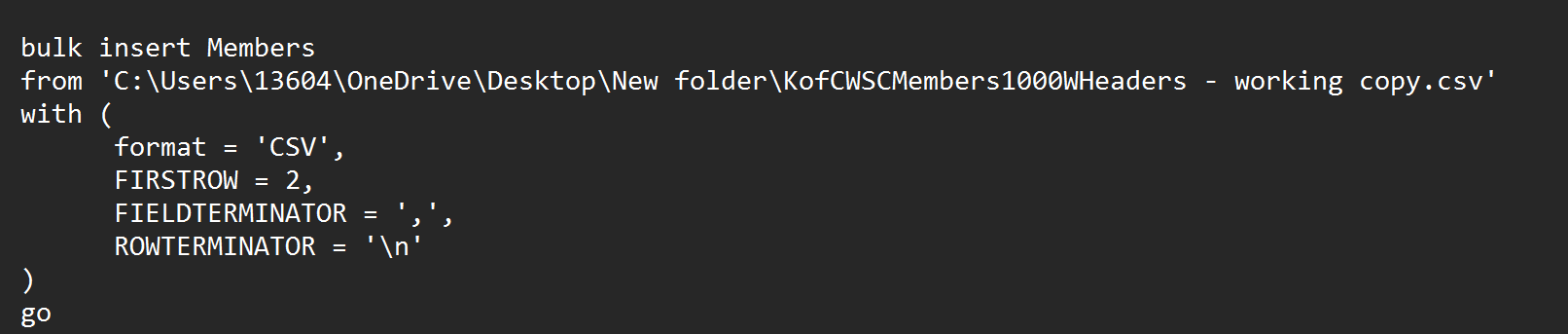
LastLoggedIn datetime null,

CanEditAdmUI bit,

DoNotEmail bit

)

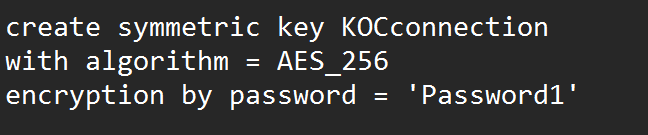
Due to working with SQL Server Express there is no import wizard to add the working copy CSV file. To get the CSV file into SQL Server Express the user will have to enter the following code into a new query.



A main goal for this project was to encrypt the members list. This would require the team to identify the personal identifiable information and encrypt it for safe storage. It was a possible finding of the team during our research phase that the current database was not encrypted and that needed to be remedied.

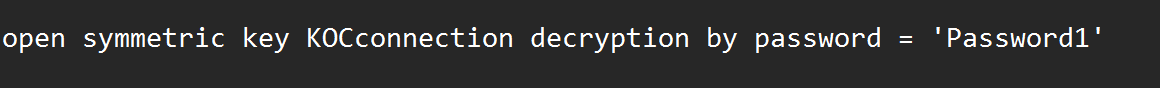
We were able to encrypt the data in the following step.

* Create the symmetric key.

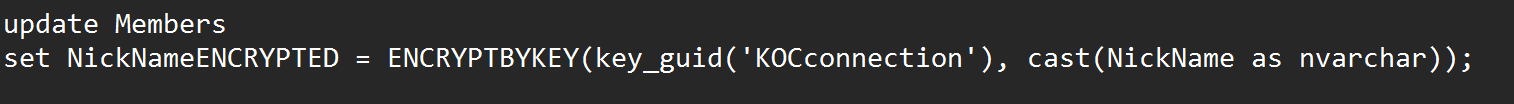


Note: The symmetric key and the password will need to be changed before the site goes live.

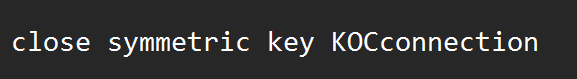
* Open the symmetric key.



* Encrypt the column.

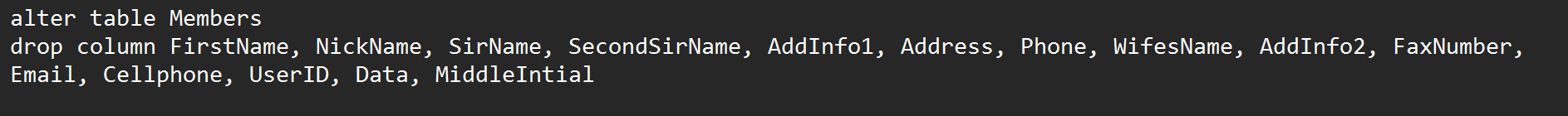


* Close the symmetric key.



Note: Ensure the symmetric key is closed.

* Drop non-encrypted columns.



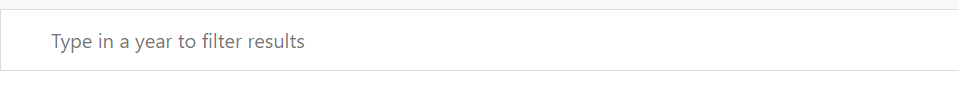
Note: The most important step.

The group did not get in contact with the original webmaster until April 28th, 2023. The purpose of that meeting was to get insight into the current database and to possibly get a sample of the current database. At the conclusion of that meeting we were able to receive approximately a thousand member list and see what would be required to incorporate the entire list for the Knights of Columbus website. With the members list inserted into our database late into the project we realize now that the “SignUp” table and page will need to be updated to match the current members list. We constructed the current “SignUp” table and page with our best educated guess. We were also able to confirm the running database is not encrypted as technology over the last twenty years has changed.

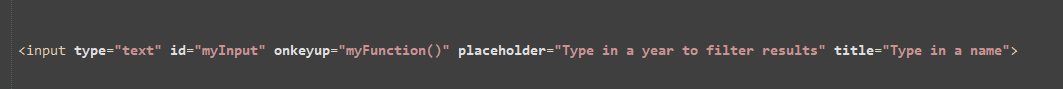
**Filter Tables:**

One goal with the project was for the ability to filter through documentation in the site. Currently we have the Knight’s monthly bulletin filterable. This was accomplished by first moving all of the links into a table. This table is then filterable through the code demonstrated below.

The filter allows you to type in a year into the below text box and have only that years bulletins appear



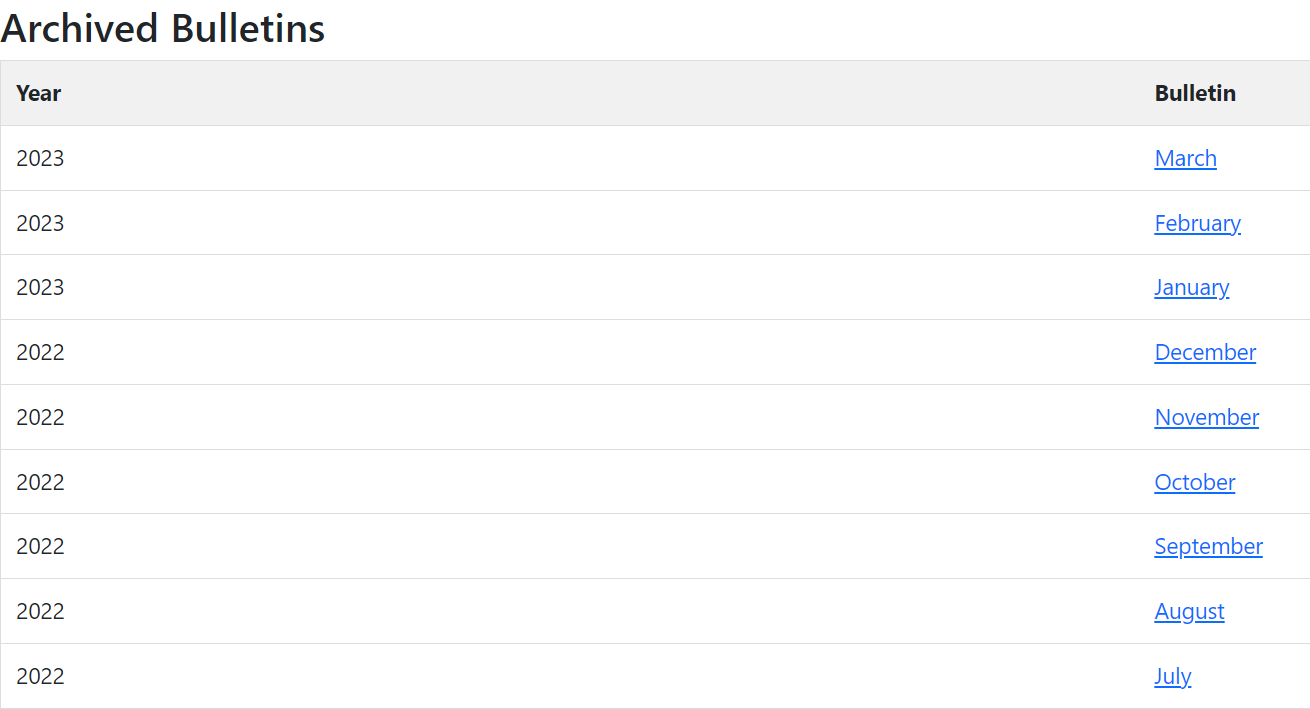
The code that makes up the text input can be seen below.



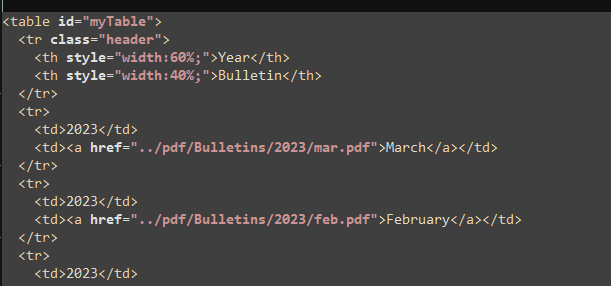
This text box was created using the html input object and is a “text” type input.

The placeholder attribute creates the text that appears in the box before typing in anything.

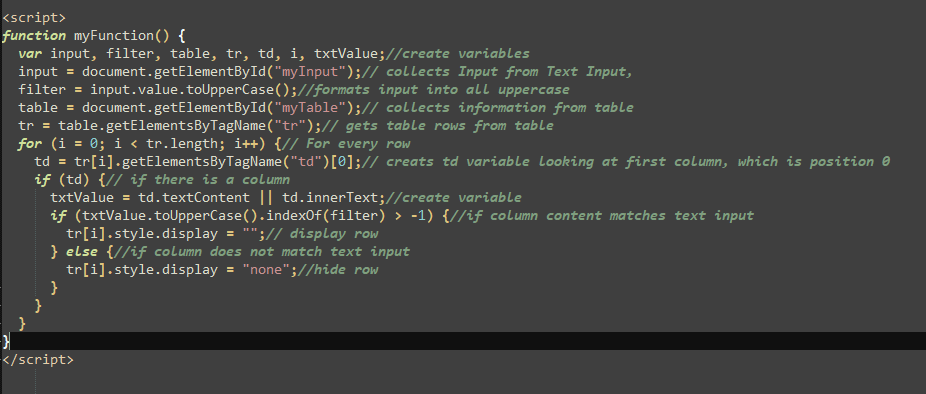
The table appears as below with the years in the first column and the bulletins in the second column



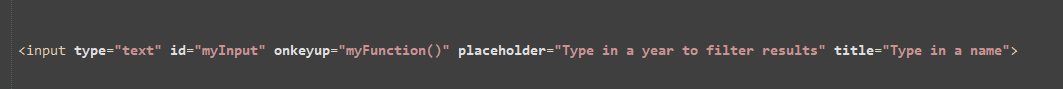
The table is created with the code below



The filtering is done by this function, this function reads the text input and table by referencing their id’s set by the id= parameter being “myInput” and “myTable” respectively. By reading both it sees if the text in the first column contains the text entered into the text input and if the row does not contain what is entered in the text input that row is hidden from view.

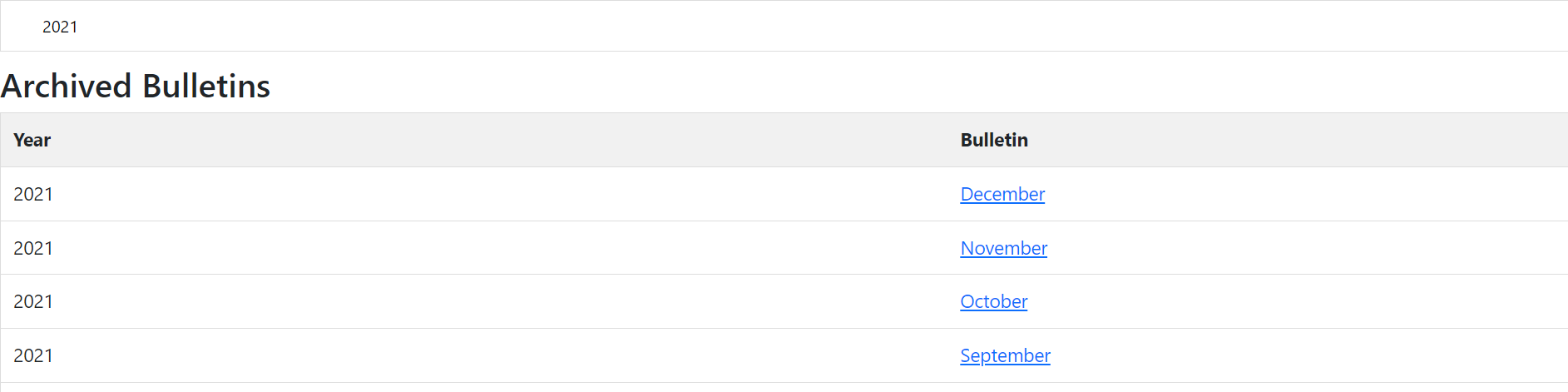


Looking back at the text input we can see the onkeyup parameter, this means that when a key is released when you are in the text field it will call the function

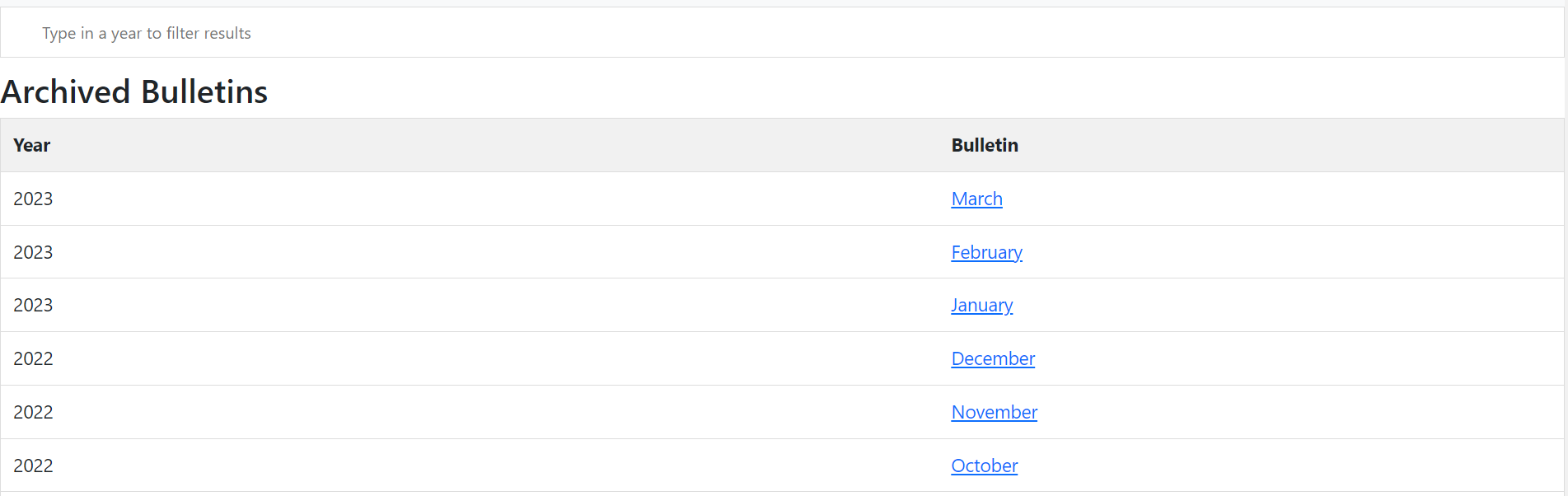


An example of the table being filtered

With 2021 typed in



Without a filter applied



**Bootstrap Implementation**

Bootstrap is used to beautify a website. For our project Bootstrap can be seen in numerous places to create a website that is both functional and up to date. In our project Bootstrap is used in the navigation bar, the cards that hold Faith, Community, Family, Life and the carousel. Furthermore Bootstrap allows the user to add to their project without having to hardcode the navigation bar or the carousel. To use Bootstrap for this program we linked stylesheets in the head of “\_Layout.cshtml”.

