ASP.NET (.Net Framework) MVC – Empty Project

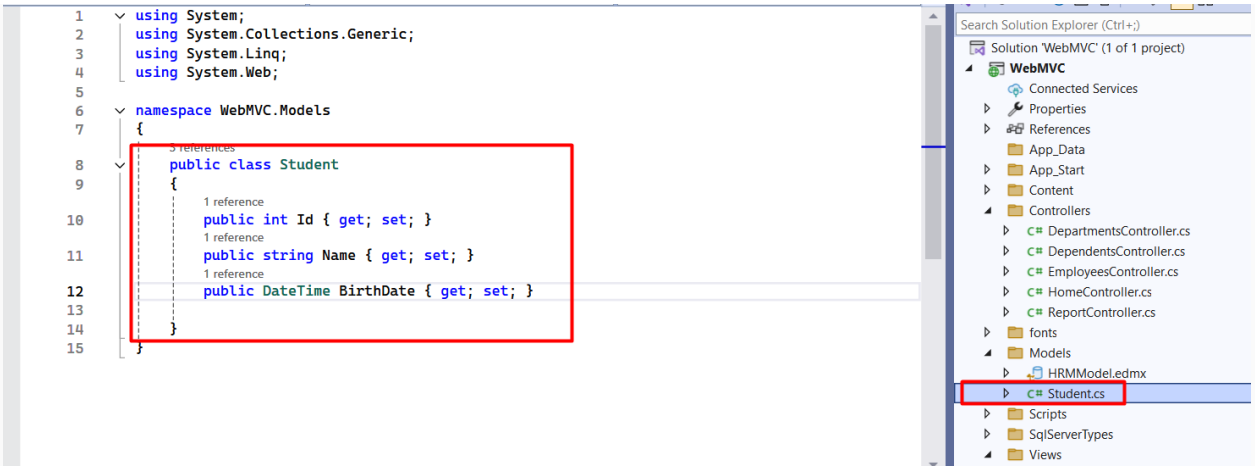
This project aims to create a web application using ASP.NET MVC (.NET Framework) that allows users to filter student data based on birth dates and display the results in a report. The application is connected to an SQL Server database to store and retrieve student information. By following this project, you will learn how to build a form, handle form submissions, query a database, and display data in a structured format using MVC views.

Step 1: Set Up Your Project

1. **Create a new ASP.NET MVC project** in Visual Studio.
    - Open Visual Studio.
    - Select **File > New > Project**.
    - Choose **ASP.NET Web Application (.NET Framework)**.
    - Select **MVC** template and click **Create**.

Step 2: Create the Model

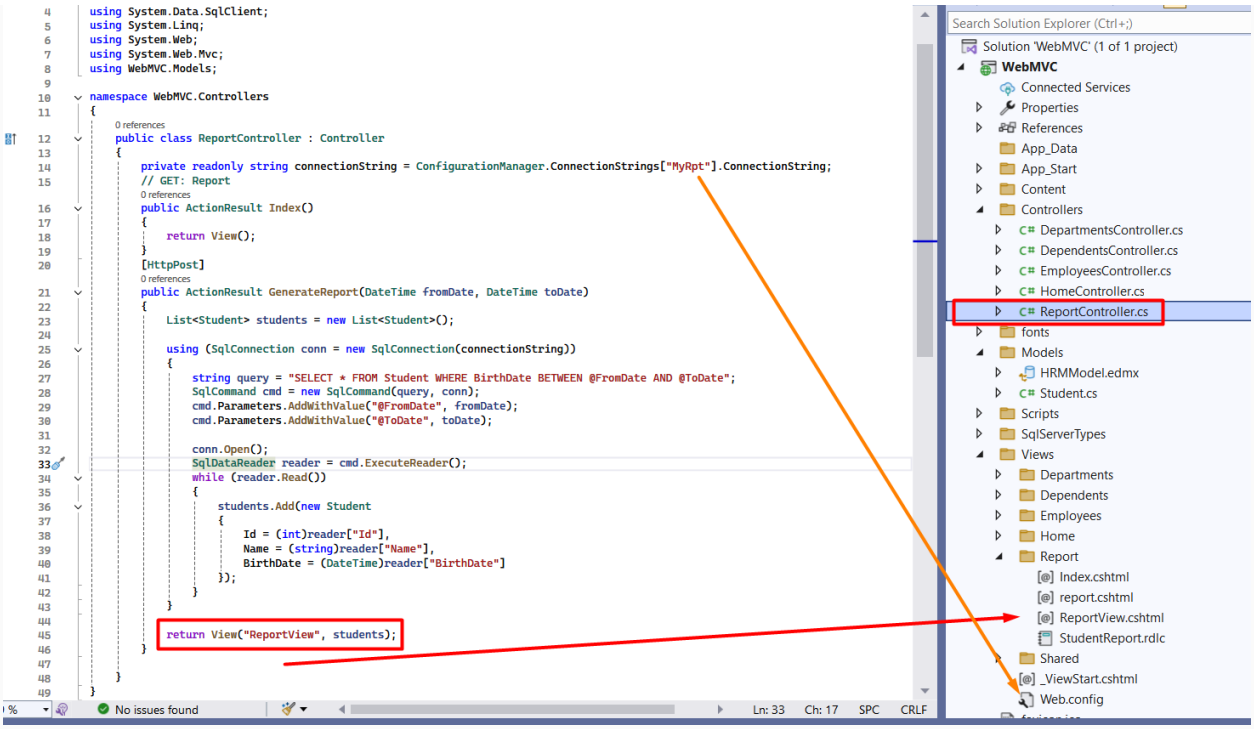
1. **Add a new class** for the student model.
    - Right-click on the **Models** folder and select **Add > Class**.
    - Name it `Student.cs`.




```
public class Student
{
    public int Id { get; set; }
    public string Name { get; set; }
    public DateTime BirthDate { get; set; }
}
```

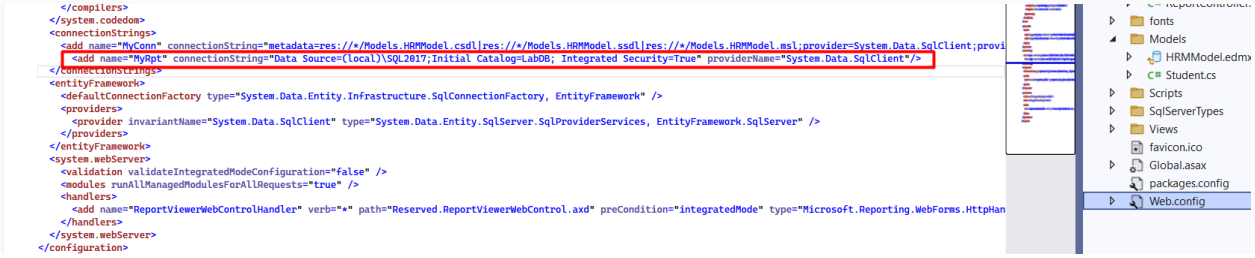
Step 3: Create the Controller

1. **Add a new controller** to handle the form submission and report generation.
    - Right-click on the **Controllers** folder and select **Add > Controller**.
    - Choose **MVC Controller - Empty** and name it `ReportController`.



MyRpt in Web.Config to <ConnectionStrings> <ConnectionStrings>

<add name="MyRpt" connectionString="Data Source=(local)\SQLEXPRESS; Initial Catalog=LabDB; Integrated Security=True" providerName="System.Data.SqlClient"/>




```
public class ReportController : Controller
{
    private readonly string connectionString =
ConfigurationManager.ConnectionStrings["MyRpt"].ConnectionString;

    public ActionResult Index()
    {
        return View();
    }

    [HttpPost]
    public ActionResult GenerateReport(DateTime fromDate, DateTime toDate)
    {
        List<Student> students = new List<Student>();

        using (SqlConnection conn = new SqlConnection(connectionString))
        {
```

```
            string query = "SELECT * FROM Student WHERE BirthDate BETWEEN
@FromDate AND @ToDate";
            SqlCommand cmd = new SqlCommand(query, conn);
            cmd.Parameters.AddWithValue("@FromDate", fromDate);
            cmd.Parameters.AddWithValue("@ToDate", toDate);

            conn.Open();
            SqlDataReader reader = cmd.ExecuteReader();
            while (reader.Read())
            {
                students.Add(new Student
                {
                    Id = (int)reader["Id"],
                    Name = (string)reader["Name"],
                    BirthDate = (DateTime)reader["BirthDate"]
                });
            }
        }

        return View("ReportView", students);
    }
}
```

Step 4: Create the Views

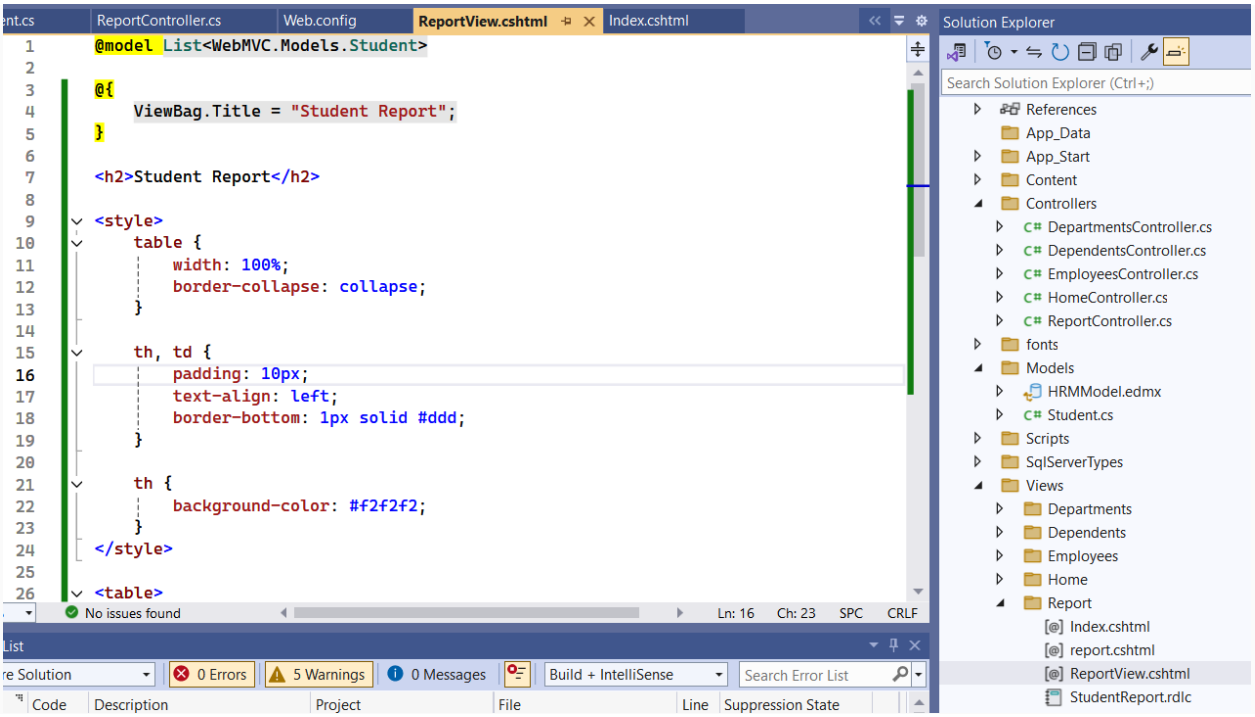
1. **Create the Index view** to display the form.
   - Right-click on the **Views\Report** folder and select **Add > View**.
   - Name it **`Index.cshtml.`**

```
@{
    ViewBag.Title = "Student Report";
}

<h2>Enter Date Range</h2>

<form method="post" action="/Report/GenerateReport">
    <div>
        <label for="fromDate">From Date:</label>
        <input type="date" id="fromDate" name="fromDate" required />
    </div>
    <div>
        <label for="toDate">To Date:</label>
        <input type="date" id="toDate" name="toDate" required />
    </div>
    <button type="submit">Submit</button>
</form>
```

2. **Create the Report view** to display the filtered student information.
   - Right-click on the **Views\Report** folder and select **Add > View**.
   - Name it **`ReportView.cshtml.`**




```
@model List<YourProjectName.Models.Student>

@{
    ViewBag.Title = "Student Report";
}

<h2>Student Report</h2>

<style>
    table {
        width: 100%;
        border-collapse: collapse;
    }
    th, td {
        padding: 10px;
        text-align: left;
        border-bottom: 1px solid #ddd;
    }
    th {
        background-color: #f2f2f2;
    }
</style>

<table>
    <thead>
        <tr>
            <th>ID</th>
            <th>Name</th>
            <th>Birth Date</th>
        </tr>
    </thead>
    <tbody>
        @foreach (var student in Model)
```

```
        {
            <tr>
                <td>@student.Id</td>
                <td>@student.Name</td>
                <td>@student.BirthDate.ToShortDateString()</td>
            </tr>
        }
    </tbody>
</table>
```

Step 5: Create the `Student` Table in SQL Server

1. **Create the `Student` table** in your SQL Server database.

```
CREATE TABLE Student (
    Id INT PRIMARY KEY IDENTITY(1,1),
    Name NVARCHAR(100),
    BirthDate DATE
);
```

2. **Insert sample records** into the `Student` table.

```
INSERT INTO Student (Name, BirthDate) VALUES
('John Doe', '2000-01-15'),
('Jane Smith', '1998-05-23'),
('Alice Johnson', '1999-03-12'),
('Bob Brown', '2001-07-19'),
('Charlie Davis', '1997-11-30'),
('Diana Evans', '2002-02-25'),
('Ethan Foster', '1996-09-10'),
('Fiona Green', '2000-12-05'),
('George Harris', '1998-04-18'),
('Hannah King', '1999-06-22');
```

Step 6: Update the Connection String

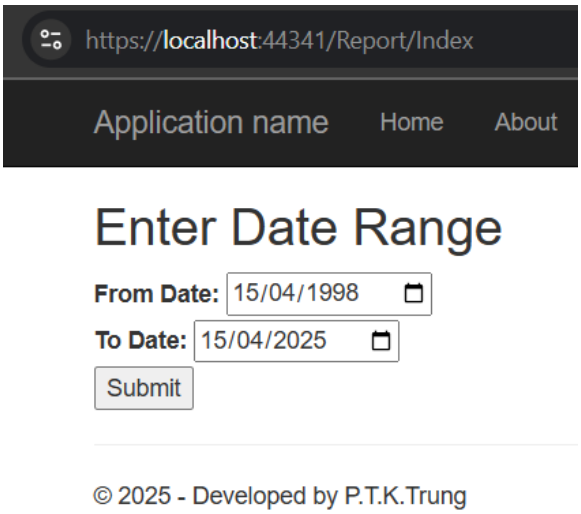
1. **Update the connection string** in `Web.config`.

```
<connectionStrings>
    <add name="DefaultConnection" connectionString="Data
Source=YOUR_SERVER_NAME; Initial Catalog=YOUR_DATABASE_NAME;Integrated
Security=True" providerName="System.Data.SqlClient" />
</connectionStrings>
```

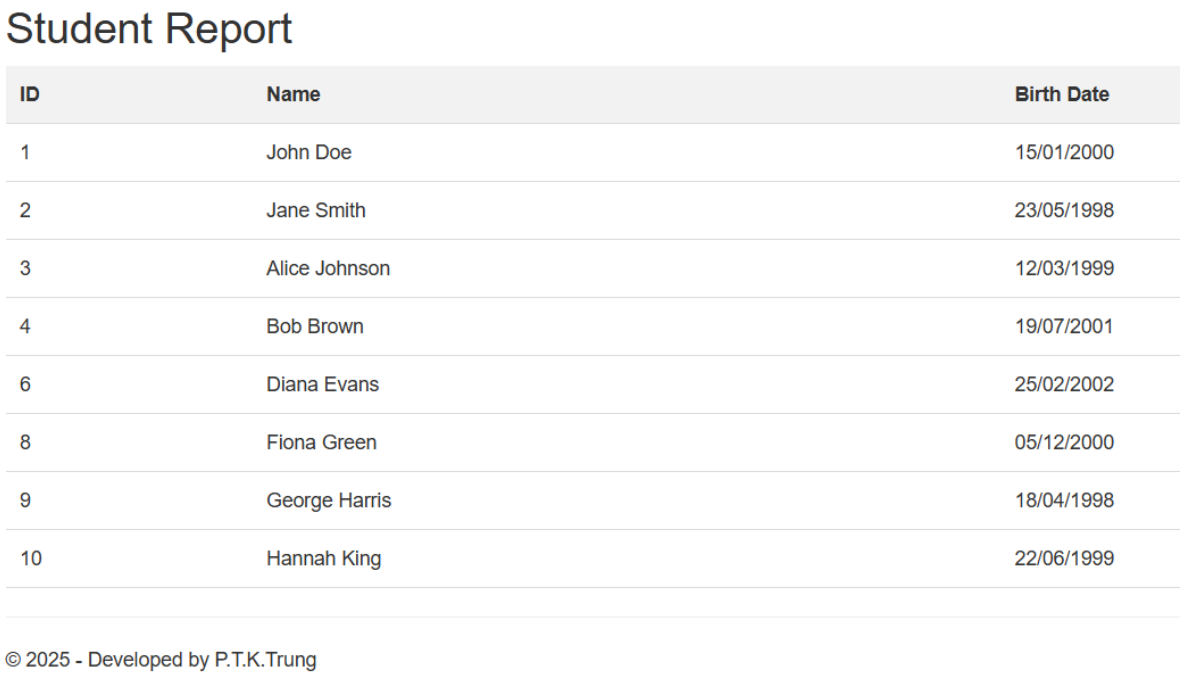
Same with Step 3 add name="MyRpt" or MyConn

Step 7: Run Your Application

1. **Build and run** your application.
    - Press **F5** to start debugging.
2. **Navigate** to the URL where your application is hosted (https://localhost:xxxx/Report/Index)



Click Submit then it will show Student Report



Lessons Learned

1. **Understanding MVC Architecture**: This project reinforces the principles of Model-View-Controller architecture, helping you understand how to separate concerns and manage data flow in a web application.
2. **Database Integration**: You will gain practical experience in connecting an ASP.NET MVC application to a SQL Server database, including creating tables and executing queries.
3. **Form Handling**: Learn how to create and handle forms in MVC, including validating user input and processing form submissions.
4. **Data Presentation**: Discover techniques for presenting data in a structured and visually appealing manner using HTML and CSS within MVC views.
5. **Debugging and Troubleshooting**: Enhance your problem-solving skills by debugging and resolving common issues that arise during development

Enjoy ☺

Email: tg_phamthaikytrung@tdtu.edu.vn