Environment

The project is a C# MVC website build in Visual Studio.
If you do not have Visual Studio, you can download Community Edition for free from:
https://visualstudio.microsoft.com/downloads

Project

Please complete as many of the steps outlined below.

The project is split into 3 levels of difficulty

- **Standard** – Functionality that is common when working as a web developer
- **Advanced** – slightly more technical tasks and problem solving
- **Expert** – tasks with a higher level of problem solving and architecture needed

1. Standard – Filters Section

I have written the code for one of the buttons in the Filters section called "Show All". I would like you to create the code for the "Active Only" and "Non Active" to filter the list view further.

- **Active Only** – This will show only users where their 'IsActive' property is set to 'True'
- **Non Active** – This will show only users where their 'IsActive' property is set to 'False

2. Standard – User Model Properties

Add a new property to the 'User' class in the system called 'DateOfBirth' which is to be used and displayed on relevant sections of the app.

3. Standard - Actions Section

Create the code and UI flows for the following actions

- **Add** – A screen that allows you to create a new user and return to the list
- **View** - A screen that displays the information about a user
- **Edit** – A screen that allows you to edit a selected user from the list
- **Delete** – A screen that allows you to delete a selected user from the list

Please ensure there is sufficient validation of the data during all processes and displayed back to the end user on the UI.

4. Advanced – Data Logging

Extend the system to capture log information regarding primary actions performed on each user in the app.

On the "View" screen like to see a log of all actions that have been performed against that user.

I would also like to see a log list view that can be searched by keywords like user full name, email, log type, etc. that leads to a log view screen containing all the data about that log entry.

5. Expert – ORM

Swop out the data layer's rudimentary implementation of an Object Relational Mapper (ORM) and implement an industry standard ORM.

An example of ORM's are

- Entity Framework
- Dapper
- NHibernate

Hint: look at replacing the *DataContext* and the internals of the methods of the *DataAccess*.

Developer Notes

Please feel free to change or refactor any code that has been supplied within the solution and think about clean maintainable code and architecture when extending the project.

There is no actual database in this project, instead we have created a mock storage class that holds the data. After closing the program, the changes to the data will not be saved as the data is only persisted in memory while the app is running and will reset to original state after starting again.