Environment

The project is a C# MVC website build in Visual Studio 2015. If you do not have it, you can download Visual Studio Community 2015 for free from:
https://www.visualstudio.com/en-us/downloads/download-visual-studio-vs.aspx

Project

Please complete as many of the steps outlined below.

1. Standard Functionality - 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 Functionality - User Model Properties

I would like you to 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 Functionality - Actions Section

I have written the code for one of the buttons in the Actions section called "View". On the view, you will need to also display the data from the new 'DateOfBirth' property on the 'User'. You added from step 2 above.

Along with the actions section I would l like you to create the code and windows for the "Add", "Edit", "Delete" buttons.

- **Add** – Load a form that allows you to create a new user and return to the list
- **Edit** – Load a form that allows you to edit a selected user from the list
- **Delete** – Load a form 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 Functionality - Data Logging

I would like you to extend the system to capture log information regarding primary actions performed on each user in the app.

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

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 I 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 data after starting again.

You can access and edit the data within using the "ServiceFactory" and "Services" therein. You my create new Services and add methods to them as needed.