# Mark Riley’s Code Review of Ross Taggart

Reviewer: Mark Riley

The person being reviewed: Ross Taggart

This code review will involve reviewing the backend functionality implemented so far. Mark Riley will be reviewing the code created by Ross Taggart and will be making appropriate comments. The files reviewed are detailed below:

* MilestoneDb.cs (source/Database/MilestoneDb.cs)
* User.cs (source/Models/User.cs)
* UserController.cs (source/Controllers/UserController.cs)
* ProjectController.cs (source/Controllers/ProjectController.cs)
* LoginModel.cs (source/Models/ViewModels/LoginModel.cs)
* RegisterModel.cs (source/Models/ViewModels/RegisterModel.cs)

The tables below have been created by taking some examples from a code review checklist at;

<https://www.evoketechnologies.com/blog/code-review-checklist-perform-effective-code-reviews/>

#### MilestoneDb.cs

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | Separated interface method to create DBcontext |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐⭐⭐ | Descriptive comments, use of the ASP.NET dbcontext features |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for variables |
| Reusability | ⭐⭐⭐⭐⭐ | Used throughout the application |

#### User.cs

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | No further comments |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐⭐⭐ | Good use of commenting, makes use of Identity User framework tool |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for variables |
| Reusability | ⭐⭐⭐⭐⭐ | Used in the projects class and user controller |

#### UserController.cs

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing, all code is readable on screen |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | Separation of methods for the controller |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐ | Makes good use of framework tools but none of the code is commented |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for methods |
| Reusability | ⭐⭐⭐⭐⭐ | Used to handle all user requests |

#### ProjectController.cs

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing, all code is readable on screen |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | Separation of methods for the controller |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐ | Makes good use of framework tools but none of the code is commented again |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for methods |
| Reusability | ⭐⭐⭐⭐⭐ | Used to handle all user requests regarding projects |

#### LoginModel.cs

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing, all code is readable on screen |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | No further comments |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐⭐⭐ | Makes good use of framework tools such as the required tag, commented well |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for variables |
| Reusability | ⭐⭐⭐⭐⭐ | Used to bind values when a user logs in |

#### RegisterModel.cs

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing, all code is readable on screen |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | No further comments |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐⭐⭐ | Makes good use of framework tools such as the required tag, commented well |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for variables |
| Reusability | ⭐⭐⭐⭐⭐ | Used to bind values when a user tries to register |

**Overall Comments:**

Overall the code in this review was to a high standard, some of the classes haven't been commented where appropriate, however, functionally is fantastic.

**Post Review Actions:**

The person being reviewed has taken on board the comments given by the reviewer and will add comments to the files that did not have them pre-review. A commit will be made to the GitHub repository with the title - **Post Backend Review 1:** *Message of change*

# Ross Taggart’s Code Review of Mark Riley

Reviewer: Ross Taggart

The person being reviewed: Mark Riley

This code review will involve reviewing the backend functionality implemented so far. Ross Taggart will be reviewing the code created by Mark Riley and will be making appropriate comments. The files reviewed are detailed below:

* HomeController.cs (source/Controllers/HomeController.cs)
* MilestoneController.cs (source/Controllers/MilestoneController.cs)
* Milestone.cs (source/Models/Milestone.cs)
* Project.cs (source/Models/Project.cs)

The tables below have been created by taking some examples from a code review checklist at;

<https://www.evoketechnologies.com/blog/code-review-checklist-perform-effective-code-reviews/>

#### HomeController.cs

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | This is a small class at this current juncture and the code is as separated as possible |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐⭐⭐ | Descriptive comments, makes use of the return View() functionality of an IActionResult |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for variables |
| Reusability | ⭐⭐⭐⭐⭐ | Used to handle requests made to the landing page of the application |

#### MilestoneController.cs

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | Separation of methods for the controller |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐⭐ | Descriptive comments, makes use of the return View() functionality of an IActionResult, however, some XML summary comments start with lowercase and all of the methods do not contain comments within them |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for variables |
| Reusability | ⭐⭐⭐⭐⭐ | Used to handle requests made about a project’s milestones |

#### Milestone.cs

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | No further comments |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐⭐⭐ | Good use of XML commenting, makes use of both ASP ordinary data annotations as well as db annotations |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for variables |
| Reusability | ⭐⭐⭐⭐⭐ | Used in the projects class and milestone controller |

#### Project.cs

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | No further comments |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐⭐⭐ | Good use of XML commenting, makes use of both ASP ordinary data annotations as well as db annotations |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for variables |
| Reusability | ⭐⭐⭐⭐⭐ | Used in project and user controller |

**Overall Comments:**

Overall the code in this review was to a high standard, some of the controller methods lack internal comments, however, functionally is fantastic.

**Post Review Actions:**

The person being reviewed has taken on board the comments given by the reviewer and will add comments to the controller methods that did not have them pre-review. A commit will be made to the GitHub repository with the title - **Post Backend Review 2:** *Message of change*

# Katie King’s Code Review of Aidan Marshall

Reviewer: Katie King

The person being reviewed: Aidan Marshall

This code review will involve reviewing the front end functionality implemented so far. Katie King will be reviewing the code created by Aidan Marshall and will be making appropriate comments. The files reviewed are detailed below:

* AddProject.cshtml (source/Views/Project/AddProject.cshtml)
* EditProject.cshtml (source/Views/Project/EditProject.cshtml)
* Projects.cshtml (source/Views/Project/Projects.cshtml)
* Login.cshtml (source/Views/User/Login.cshtml)
* Register.cshtml (source/Views/User/Register.cshtml)
* \_Navbar.cshtml (source/Views/Shared/Layout/\_Navbar.cshtml)
* \_Footer.cshtml (source/Views/Shared/Layout/\_Footer.cshtml)

The tables below have been created by taking some examples from a code review checklist at;

<https://www.evoketechnologies.com/blog/code-review-checklist-perform-effective-code-reviews/>

#### AddProject.cshtml

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | Separated concerns in different divs |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐ | No comments as of yet, this should be added |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for variables |
| Reusability | ⭐⭐⭐⭐⭐ | Used throughout other pages within the application |

#### EditProject.cshtml

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | N/A |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐ | No comments as of yet, this should be added |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for variables, good use of indentation and formatting |
| Reusability | ⭐⭐⭐⭐⭐ | Used in EditMilestone.cshtml page |

#### Projects.cshtml

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing, all code is readable on screen |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | Separates each input into different divs |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐ | none of the code is commented, but there is a good use of frameworks throughout |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for methods |
| Reusability | ⭐⭐⭐⭐⭐ | This is code is adapted and used within the Milestone.cshtml Page |

#### Login.cshtml

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing, all code is readable on screen |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | Username and password separated in divs |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐ | none of the code is commented again |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for methods |
| Reusability | ⭐⭐⭐⭐⭐ | Unique to this page and cannot be used elsewhere in the application. However, can be used to create a login in another application |

#### Register.cshtml

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing, all code is readable on screen |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | All inputs separated in divs |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐⭐⭐ | Not currently commented, this will be changed. |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for variables |
| Reusability | ⭐⭐⭐⭐ | Unique to this page and cannot be used elsewhere in the application. However, can be used to create a registration in another application |

#### \_Navbar.cshtml

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad, 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing, all code is readable on screen |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | No further comments |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐⭐ | commented well, and make use of frameworks |
| Overall Readability | ⭐⭐⭐⭐⭐ | Readable variable names |
| Reusability | ⭐⭐⭐⭐⭐ | Will be able to use this within other applications. |

#### \_Footer.cshtml

|  |  |  |
| --- | --- | --- |
| Element being evaluated | Rating (1 - 5, 1 being bad, 5 being good) | Additional Comments |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing. |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | No further comments |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐⭐⭐ | commented well |
| Overall Readability | ⭐⭐⭐⭐⭐ | Readable, appropriate variable names |
| Reusability | ⭐⭐⭐⭐⭐ | Will be able to use this within other applications. |

**Overall Comments:**

Overall the code in this review was to a high standard, some of the views created haven't been commented where needed, but functionally is good.

**Post Review Actions:**

The person being reviewed has taken on board the comments given by the reviewer and will add comments to the files that did not have them pre-review. A commit will be made to the GitHub repository with the title - **Post Front End Review 1**

# Aidan Marshall’s Code Review of Katie King

Reviewer: Aidan Marshall

The person being reviewed: Katie King

This code review will involve reviewing the front-end functionality implemented so far. Aidan Marshall will be reviewing the code created by Katie King and will be making appropriate comments. The files reviewed are detailed below:

* AddMilestone.cshtml (source/Views/Milestone/AddMilestone.cshtml)
* EditMilestone.cshtml (source/Views/Milestone/EditMilestone.cshtml)
* Milestones.cshtml (source/Views/Milestone/Milestones.cshtml)
* Shared.cshtml (source/Views/Milestone/Shared.cshtml)
* \_LoggedInNavbar.cshtml (source/Views/Shared/Layout/\_LoggedInNavbar.cshtml)

The tables below have been created by taking some examples from a code review checklist at;

<https://www.evoketechnologies.com/blog/code-review-checklist-perform-effective-code-reviews/>

#### AddMilestone.cshtml

|  |  |  |
| --- | --- | --- |
| **Element being evaluated** | **Rating (1 - 5, 1 being bad 5 being good)** | **Additional Comments** |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of css naming conventions |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | Good structure with well indented divs and separation of concerns |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐⭐ | No comments as of yet, this should be added |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for variables |
| Reusability | ⭐⭐⭐⭐⭐ | This page originated from another view and using the same |

#### EditMilestone.cshtml

|  |  |  |
| --- | --- | --- |
| **Element being evaluated** | **Rating (1 - 5, 1 being bad 5 being good)** | **Additional Comments** |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | Good structure with well indented divs and separation of concerns |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐⭐ | No comments yet, this should be added |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for variables, good use of indentation and formatting |
| Reusability | ⭐⭐⭐⭐⭐ | This page originated from another view and using the same |

#### Milestones.cshtml

|  |  |  |
| --- | --- | --- |
| **Element being evaluated** | **Rating (1 - 5, 1 being bad 5 being good)** | **Additional Comments** |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing, all code is readable on screen |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | Separates each input into different divs |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐ | none of the code is commented, but there is a good use of frameworks throughout |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for methods |
| Reusability | ⭐⭐⭐⭐⭐ | This page originated from another view and using the same |

#### Shared.cshtml

|  |  |  |
| --- | --- | --- |
| **Element being evaluated** | **Rating (1 - 5, 1 being bad 5 being good)** | **Additional Comments** |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing, all code is readable on screen |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | Username and password separated in divs |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐ | none of the code is commented again |
| Overall Readability | ⭐⭐⭐⭐⭐ | Appropriate names used for methods |
| Reusability | ⭐⭐⭐⭐⭐ | This page originated from another view and using the same |

#### \_LoggedInNavbar.cshtml

|  |  |  |
| --- | --- | --- |
| **Element being evaluated** | **Rating (1 - 5, 1 being bad 5 being good)** | **Additional Comments** |
| Code Formatting (whitespace, naming conventions) | ⭐⭐⭐⭐⭐ | Cleanly written code, good use of camel casing, all code is readable on screen |
| Architecture (Separation of concerns) | ⭐⭐⭐⭐⭐ | All inputs separated in divs |
| Coding Best Practice (Comments, use of framework features) | ⭐⭐⭐⭐⭐ | Not currently commented |
| Overall Readability | ⭐⭐⭐⭐⭐ | Good simple design and layout easy to navigate |
| Reusability | ⭐⭐⭐⭐ | Good simple design and layout easy to navigate |

**Overall Comments:**

Overall the code in this review was to a high standard, some of the views created haven't been commented where needed, but functionally is of a high standard.

**Post Review Actions:**

The person being reviewed has taken on board the comments given by the reviewer and will add comments to the files that did not have them pre-review. A commit will be made to the GitHub repository with the title - **Post Front End Review 1**