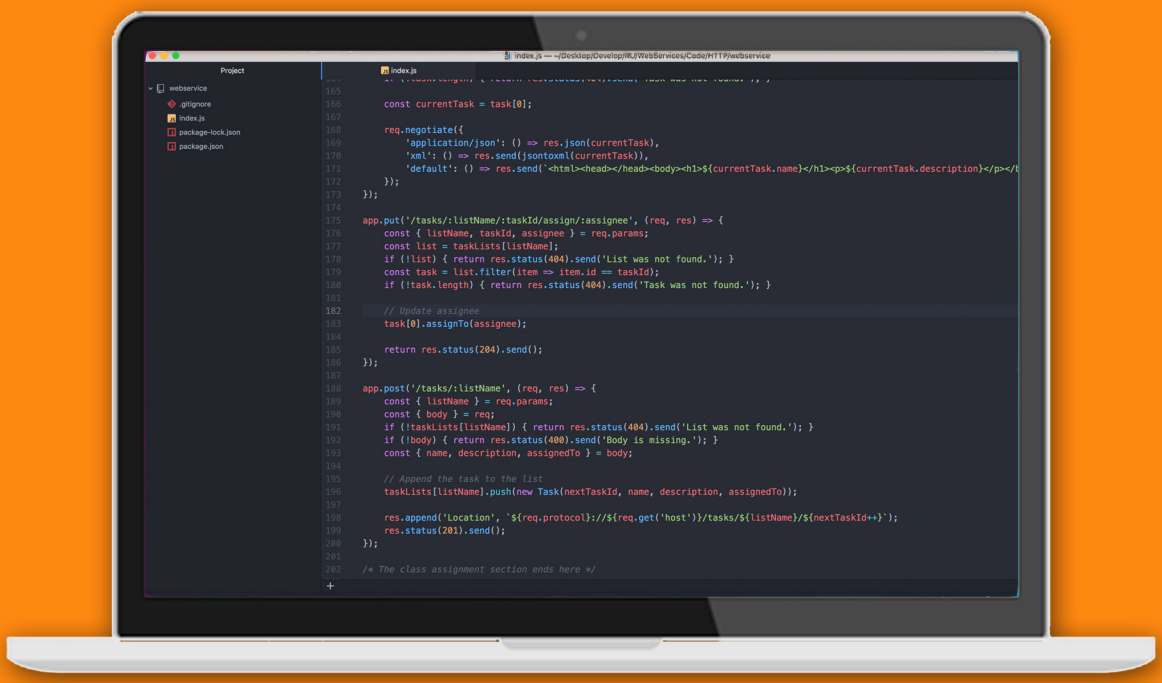

Clean that code!

Class Assignment - Clean that code!

A new company called **Clean that code!** wants to create a application that is based on a community of software developers that wants to write high cohesive, loosely coupled, readable and effective code. They have an already established codebase, but the codebase doesn't have a single unit test setup. This is where you come in! Your job is to provide 100% code coverage for the codebase.

Template

The template can be downloaded in Canvas (*template.zip*). The template includes the following:

- Six projects
  - *CleanThatCode.Community.Common*
    - **StringHelpers.cs**
  - *CleanThatCode.Community.Models*
    - **CommentDto, PostDto**
    - **Comment, Post**
  - *CleanThatCode.Community.Repositories*
    - **ICleanThatCodeDbContext / CleanThatCodeDbContext**
    - **ICommentRepository / CommentRepository**
    - **IPostRepository / PostRepository**
  - *CleanThatCode.Community.Services*
    - **IPostService / PostService**
    - **ICommentService / CommentService**
  - *CleanThatCode.Community.Tests*
    - **StringHelpersTests**
    - **Mocks/**
      - **FakeData**
  - *CleanThatCode.Community.WebApi*
    - **PostController**

Assignment description

Below is a description on what should be implemented in this assignment:

- **(30%)** Within **CleanThatCode.Community.Common** resides a file called **StringHelpers.cs** which contains a static class called **StringHelpers** which has three extension methods called **ToDotSeparatedString(), CapitalizeAllWords()** and **ReverseWords()**. Your job is to implement these methods. When they have been implemented correctly you can run **dotnet test** in the **CleanThatCode.Community.Tests** to check whether the tests for these methods run correctly. There are nine unit tests setup within the **StringHelpersTests** class in the test project.
  The unit tests within **StringHelpersTests.cs** should NOT be changed under any circumstances!

- **(15%)** Create a new test class called **CommentRepositoryTests** within the test project which should be used to unit test the **CommentRepository**. The **ICleanThatCodeDbContext** which is accepted as the first parameter to **CommentRepository** constructor should be mocked by creating your own mocked version called **CleanThatCodeDbContextMock** within the **Mocks/** folder in the test project. This mocked class should make use of the data provided in **FakeData.cs** which resides also within the **Mocks/** folder. Pass the mocked version into the **CommentRepository** constructor within the test file.

- **(5%)** Create a unit test called **GetAllCommentsByPostId_GivenWrongPostId_ShouldReturnNoComments** which should test the GetAllCommentsByPostId method within the **CommentRepository** and

pass in an invalid post id and assert the result to be of length 0.

- **(5%)** Create a unit test called **GetAllCommentsByPostId_GivenValidPostId_ShouldReturnTwoComments** which should test the GetAllCommentsByPostId method within the **CommentRepository** and pass in a valid post id and assert the result to be of length 2. (*Look at the test data, to determine what post id is valid.*)

- **(30%)** Create a new test class called **PostRepositoryTests** within the test project which should be used to unit test the **PostRepository**. The **ICleanThatCodeDbContext** which is accepted as the first parameter to **PostRepository** constructor should be mocked by using **Moq** (*a package found within NuGet and is already setup in the template*). This mocked class should make use of data provided by **NBuilder** (*a package found within NuGet and is already setup in the template*). Pass the mocked version into the **PostRepository** constructor within the test file.

  The NBuilder list should be of length 3, the first two items should have a title which contains the word “Grayskull” and the author should be “He-Man” and the last item should have a title which contains the word “Hack the planet!” and the author should be “Richard Stallman”. Other properties should not be set.

- **(5%)** Create a unit test called **GetAllPosts_NoFilter_ShouldContainAListOfThree** which should test the GetAllPosts method within the **PostRepository** and pass in no filter and check if the list is of length 3.

- **(5%)** Create a unit test called **GetAllPosts_FilteredByTitle_ShouldContainAListOfTwo** which should test the GetAllPosts method within the **PostRepository** and pass in a filter for the title containing the word “Grayskull” and check if the list is of length 2.

- **(5%)** Create a unit test called **GetAllPosts_FilteredByAuthor_ShouldContainAListOfOne** which should test the GetAllPosts method within the **PostRepository** and pass in a filter for the author containing the word “Stallman” and check if the list is of length 1.

Submission

A single compressed file (***.zip, *.rar**) should be submitted to **Canvas**. If you are working in groups, please remember to state the name of all the group members (*excluding the one submitting*).