

Tag is moved from a backlog task to a doing task

using *Microsoft*.*VisualStudio*.*TestTools*.*UnitTesting*;

using *RESTFull*.*Models*;

namespace *TESTFull*

{

[*TestClass*]

public class *TagTest*

{

[*TestMethod*]

[*DataRow*("Chicken","Caesar Salad")]

[*DataRow*("Veggie", "Caesar Salad")]

[*DataRow*("Chicken", "Fried Chicken")]

public void TestNewMenuItem(string a, string b)

{

*Tag* T = new *Tag*() { TagName = a, MenuName = b};

Assert.AreEqual(a, T.TagName);

Assert.AreEqual(b, T.MenuName);

//Required Attribute

Assert.IsNotNull(T.TagName);

}

[*TestMethod*]

public void TestMenuItemRequired()

{

*Tag* T = new *Tag*() { MenuName="SomeMenu"};

Assert.IsNotNull(T.TagName);

}

}

}

Kevin has already defined some unit tests to develop around, opening RESTFull I can implement the tag system to satisfy these tests:

namespace *RESTFull*.*Models*

{

///<summary>

/// Tags are used to link food items with a certain "food based category"

///</summary>

public class *Tag*

{

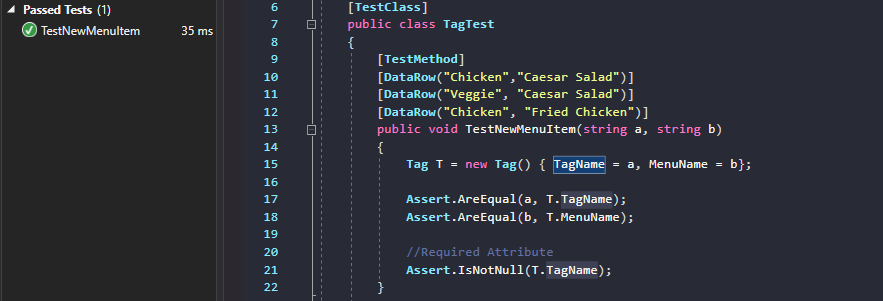
public string TagName { get; set; }

public string MenuName { get; set; }

}

}

Running the unit tests with this modified code we get this output:



The following test failed however, as the Tag name isn’t necessarily a required field

[*TestMethod*]

public void TestMenuItemRequired()

{

*Tag* T = new *Tag*() { MenuName="SomeMenu"};

Assert.IsNotNull(T.TagName);

}

Due to the structure of a Model in MVC architecture, it isn’t good to include a constructor for our model as it is supposed to be as slim as can be; to work around this, I will include “Required” attributes on the model as a means of ensuring the value won’t be null in production;

///<summary>

/// Tags are used to link food items with a certain "food based category"

///</summary>

public class *Tag*

{

[*Required*]

public string TagName { get; set; }

public string MenuName { get; set; }

}

It is now implied that the Tag will not be valid without a name, meaning we shall change the test to pass;

