Technical Inspection

October 29, 2018

Drs. Andresen, Nielsen, and Hsu,

I have inspected and completed the steps defined in the User Manual v2.0 for Thaddeus Tuck’s MVC Reporting Website Framework. The results of my inspection can be found below. I was able to quickly get the framework installed, configured, and running the demo due to the clarity and thoroughness of the manual. I found certain steps to be slightly tedious from a user perspective, and have offered recommendations in the inspection comments. These concerns were minor, however, and did not prevent any inspection items from passing.

Sincerely,

Richard Waliser

	Item #	Inspection Item	Pass/Fail/Partial
	TI-1	Instructions enable the tester to create an ASP.NET MVC Project in Section 2.2.2 Create a New Project of the User Manual.	Pass
Comments	● Section 2.2.2 could be simplified by providing a Visual Studio project template as documented at https://docs.microsoft.com/en-us/visualstudio/ide/how-to-create-project-templates?view=vs-2017 and instead instruct the user how to install the template in Visual Studio and create a new reporting project from the template.		
	TI-2	Instructions enable the tester to setup a local NuGet repository in Section 2.2.3 Setup Local NuGet Package Repository of the User Manual.	Pass
Comments			
	TI-3	Instructions enable the tester to install the MVC Reporting Framework NuGet package in Section 2.2.4 Install the TripleT.ReportingFramework.Core.nupkg	Pass
Comments	● Section 2.2.4 could also be simplified by using the project template as suggested above. The nuget package would already be installed as part of the template. ● Recommend using semantic versioning for NuGet packages as described at https://semver.org/.		

	TI-4	Instructions enable the tester to configure the ASP.Net MVC project from TI-1 for use with the MVC Reporting Framework NuGet Package in Section 2.2.5 Configure the Project to Use the TripleT.ReportingFramework.Core.nupkg	Pass
Comments	● Section 2.2.5 could also be simplified by using the project template as suggested above. The project would already be configured as part of the template. ● On page 19, the user is instructed to “Click Site.css” and “In the… properties window…” but it’s possible that the user may not have the properties window already open. ● On page 19, the user is informed that the Site.css file ensures creation of the Content folder during deployment. A few steps, later the user is informed that this is also the purpose of the PlaceHolder style sheet which is in the same Content folder. One of these seems redundant. ● On page 20, the user is instructed to open both Web.config file in Views and the Web.config file in the project root. The user is then instructed to make a change to the Web.Config file in Views, but this instruction is sub-bulleted incorrectly under the step for the root Web.config file.		
	TI-5	Instructions enable the tester to setup a table-based report in Section 2.2.6 Creating a Table-Based Report	
Comments	● Section 2.2.6 could also be simplified by using the project template as suggested above. The demo could be included as part of the template. ● Recommend using textual code blocks for code snippets instead of images depicting code. Textual code snippets are considered an industry standard and a courtesy as demonstrated by numerous projects on https://github.com as well as many code blogs. These snippets should use a font style and size suitable for viewing code. ● Recommend combining groups of steps that instruct the user to		

	add individual using statements to a particular file into a single step with a code snippet containing all the using statements to add to that file. ● Recommend using a library such as Bogus to generate large amounts of random data for the demo instead of coding data and randomizers by hand. An example can be found at https://github.com/bchavez/Bogus#the-great-c-example. ● The DemoService class in the demo derives from a BaseService class that internally uses Activator.CreateInstance to instantiate a repository instance. Recommend passing the instance instead to the BaseService constructor so that users can have control over the dependencies used by their repositories. For example, if a repository had a constructor dependency on an EntityFramework DBContext, then the BaseService would not be able to resolve this dependency. ● The search field in the demo did not seem to work given a partial or complete value from any column.