Test Plan

For MVC Reporting Website Framework

(MVC RWF)

Version 1.0

Submitted in partial fulfillment of the Masters of Software Engineering Degree.

Thaddeus Thomas Tuck

CIS 895 – MSE Project

Department of Computing and Information Sciences

Kansas State University

Committee Members

Dr. Daniel Andresen

Dr. Mitchell Neilsen

Dr. William Hsu

Change Log

Version #	Changed By	Release Date	Change Description
1.0	Thaddeus Tuck	5/5/2018	Initial Release

Table of Contents

1. Test Plan Identifier

TripleT.ReportingFramework.Core.Tests

2. Introduction

This document provides the methodology for how the MVC Reporting Website Framework will be tested. Visual Studio allows the creation of unit testing projects which can be used to unit test aspects of a Visual Studio project. Selenium is a framework/API that allows for the incorporation of integration tests of the website to verify that framework properly renders the webpages generated by the MVC Reporting Website Framework.

2.1 References

[1] Tuck, T. Vision Plan 2.0 Retrieved 05/05/2018, from Thaddeus Tuck's Project Page Web Site: http://thaddeustuckmastersproject.azurewebsites.net

[2] Naga Sowjanya Karumuri, Test Plan v2.1 Retrieved 05/05/2018, from Sowjanya's Project Page Web Site: http://people.cis.ksu.edu/~sowji/100jiMSE/

[3] Hill, K. Test Plan 1.0 Retrieved 05/05/2018, from Kyle Hill's Project Page Web Site: http://mse.cs.ksu.edu/hill/

3. Test Items

The following items will be tested:

- Model Related Items
- Structure Related Items
- View Related Items
- Extensibility Related Items
- Performance Related Items
- Project Initialization Related Items
- NuGet Package Implementation Related Items
- Table-based Report Related Items

4. Unit Tested Features

All features listed below will be tested through unit and integration testing through Visual Studio and Selenium and can be found in [1].

4.1 Model Related Items

- **MRI 1 –** The framework shall have a model that represents a cell (td) in a HTML table.
- **MRI 2 –** The framework shall have a model that represents a row (tr) in a HTML table.
- **MRI 3 –** The framework shall have a model that represents a column (th) in a HTML table.
- **MRI 4 –** The framework shall have a model that represents a table in HTML.
- **MRI 5 –** The framework shall have a base model that can indicate the occurrence of an error.
- **MRI 6 –** The framework models shall have properties and functions to facilitate advanced filtering.

4.2 Structure Related Items

- **SRI 1 –** The framework shall have a base repository class to facilitate standard functions to assist with model binding and processing for framework supported views.
- **SRI 2 –** The framework shall have a base service class to facilitate processing of requests from the browser and providing a central connection for necessary repositories and calculation processes.
- **SRI 3 –** The framework shall have a base controller class that will provide functions, extensions, and properties to aid in processing view models into views.

4.3 View Related Items

- **VRI 1 –** The framework shall provide a base layout that provides sections for loading CSS, JavaScript, and other HTML components.
- **VRI 2 –** The framework shall provide a specific layout for the framework supported DataTable report.
- **VRI 3 –** The framework shall provide a view that contains the HTML generated from the framework's table model and its subsequent component models.
- **VRI 4 –** The framework shall provide partials containing necessary CSS, JavaScript, and HTML to facilitate advanced filtering, column selection, and column ordering for the DataTable view.

4.4 Extensibility Related Items

- **ERI 1 –** The framework shall have a service provider which will allow the end developer to create, extend, and/or override components of the MVC Reporting Framework.
- **ERI 2 –** The framework shall have extension classes that can be used to customize standard options for whether features of the framework are enabled or disabled by default and at run time.
- **ERI 3 –** The end developer shall have the ability to use multiple ASP.NET MVC projects that reference the MVC Reporting Framework NuGet package to create a fully extensible website.

4.5 Performance Related Items

- **PRI 1 –** The framework shall be able to use post loading to render and be able to search and filter at least eighty thousand rows with twenty columns in less than thirty seconds from the beginning of DataTable initialization on the framework developer's machine in the Google Chrome Browser.
    - OS Name: Microsoft Windows 10 Pro
    - Version: Version 10.0.17134 Build 17134
    - System Manufacturer: MSI
    - System Model: MS-7885
    - System Type: x64-based PC
    - Processor: Intel(R) Core(TM) i7-5930K CPU @ 3.50GHz, 3501 Mhz, 6 Core(s), 12 Logical Processor(s)
    - Installed Physical Memory (RAM): 32.0 GB

5. Human Tested Features

All features listed below will be tested by human developers as forms of system integration and user acceptance testing.

5.1 Project Initialization Related Items

- The tester should be able to use provided instructions to create an ASP.NET MVC project.
- The tester should be able to use provided instructions to prepare the created ASP.NET MVC project for use with the MVC Reporting Framework NuGet Package.

5.2 NuGet Package Implementation Related Items

- The tester should be able to use the provided instructions to setup a local NuGet repository.
- The tester should be able to use the provided instructions to install the MVC Reporting Framework NuGet Package.

5.3 Table-based Report Related Items

- The tester should be able to use the provided instructions to setup a table-based report.
- The tester should be able to use framework extensions and models to setup filtering without writing JavaScript or HTML.
- **ERI 3 –** The tester should be able to use the provided instructions to create a second ASP.NET MVC project that references the MVC Reporting Framework NuGet package to create a second table-based report that is accessible and usable by the primary ASP.NET MVC website.

6. Approach

Testing will be done by the MVC Reporting Framework developer in the form of unit and integration testing through Visual Studio and by independent human testers that will be provided detail instructions for user acceptance testing and system integration testing.

7. Item Pass/Fail Criteria

7.1 Unit Tested Features

A requirement will be determined to have passed if all unit tests related to that requirement's component(s) are validated and considered passed by Visual Studio's unit testing framework.

7.2 Human Tested Features

A requirement will be determined to have passed if the human tester is able to understand and complete the instructions to achieve the desired result from the instructions.

8. Suspension Criteria and Resumption Requirements

8.1 Suspension Criteria

Testing will be halted if a test case fails during testing. The reason for the failure and a suggested solution will be logged with the test case.

8.2 Resumption Criteria

The failed test cases will be rerun from the beginning of the test after logging the failure, its cause, and a possible solution to the problem in the test log. Other independent test cases will continue to be executed to fix issues with independent project areas.

9. Test Deliverables

A separate Test Log document will document all test cases and record if each test case passed and when/if it failed.