|  |
| --- |
| Master Document |
| Movie Rental Application |
| RAD Project |

|  |
| --- |
| Aashiyandeep Singh, Samuel Lee, Joshua Macaulay  [Date] |

Contents

[**Sprint Two** 4](#_Toc42761003)

[Introduction: 4](#_Toc42761004)

[Meeting Agenda 5](#_Toc42761005)

[Additional Information 5](#_Toc42761006)

[Source Control: 6](#_Toc42761007)

[Project Management Plan: 7](#_Toc42761008)

[Performance Report 8](#_Toc42761009)

[Code Formatting: 8](#_Toc42761010)

[Profiler: 8](#_Toc42761011)

[Conclusion: 9](#_Toc42761012)

[Software Review Plan 10](#_Toc42761013)

[Functionality 10](#_Toc42761014)

[Ease of use 10](#_Toc42761015)

[Design 10](#_Toc42761016)

[Meeting the Criteria 10](#_Toc42761017)

[Software Testing Plan 11](#_Toc42761018)

[Introduction 11](#_Toc42761019)

[Scope 11](#_Toc42761020)

[In Scope 11](#_Toc42761021)

[Out of Scope 11](#_Toc42761022)

[Quality Objective 11](#_Toc42761023)

[Roles and Responsibilities 11](#_Toc42761024)

[Test Methodology 12](#_Toc42761025)

[Overview 12](#_Toc42761026)

[Test Levels 12](#_Toc42761027)

[Test Completeness 12](#_Toc42761028)

[Test Deliverables 13](#_Toc42761029)

[Resource & Environment Needs 14](#_Toc42761030)

[Test Environment and Tools 14](#_Toc42761031)

[Terms/Acronyms 15](#_Toc42761032)

[Test Results 16](#_Toc42761033)

[Functionality Test: 16](#_Toc42761034)

[Sign-up Page 16](#_Toc42761035)

[Admin Log-in 17](#_Toc42761036)

[Display All Users 17](#_Toc42761037)

[Delete Subscription from Database 18](#_Toc42761038)

[Sending an Alert 18](#_Toc42761039)

[Sending a Newsletter 18](#_Toc42761040)

[Logging out 19](#_Toc42761041)

[Accessing admin pages with no log-in 19](#_Toc42761042)

[Bibliography 20](#_Toc42761043)

# **Sprint Two**

Version 1.2

## Introduction:

Team A – Members for first sprint:  
  
The roles for this sprint have been determined as follows.  
Aashiyan

* Sign up page
* Select one or both methods of news (newsletter, newsflash) and validate.
* Update master document
* Project management (Gantt chart)

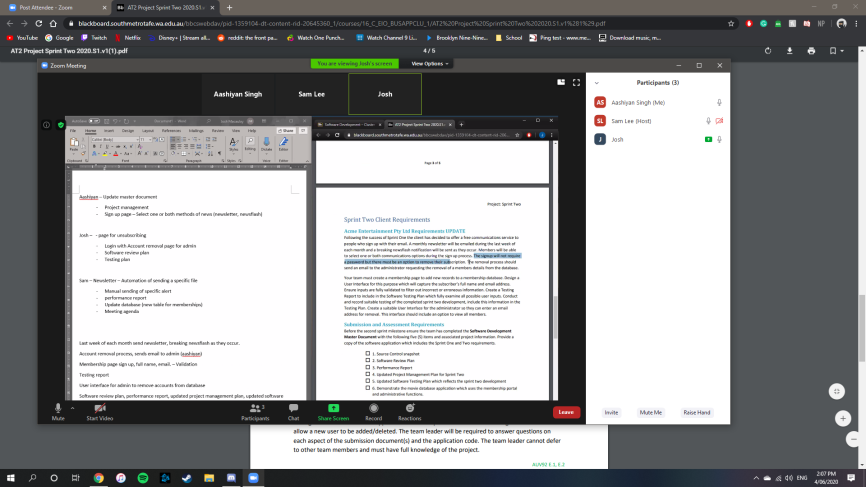
Josh

* Page for unsubscribing
* Login with Account removal page for admin (list all members)
* Software review plan
* Testing plan (test inputs into sign up for validation)

Sam

* Newsletter – Automation of sending a specific file
* Manual sending of specific alert
* performance report
* Update database (new table for memberships)
* Meeting agenda

This was decided at the meeting we held on the 4th June 2020. The meeting agenda is below which explains what we discussed in the meeting.

Evidence of the meeting:  


## Meeting Agenda

**Sprint Two**

04/06/2020

13:30

Meeting called by: Aashiyan

Attendees: Aashiyan, Sam, Josh

Please read: RAD Project Sprint 2 Requirements

Please bring: Supplies

Platform: Zoom

|  |  |  |
| --- | --- | --- |
| Time | Event Handling | Location |
| 13:30 – 14:30 | Scrum Meeting | Zoom - Online |

### Additional Information

Discussion Topics:

13:35 – Ash took responsibility for the project management plan and master document.

13:45 – Sam was tasked with handling the meeting agenda

13:50 – We discussed who would write the reports. Josh was assigned the Software Development Plan and to update the Software Testing Plan. Sam would complete the Performance report.

14:05 – Josh would remain responsible for the source control management as the repository was hosted under his account.

14: 10 – It was decided that we split the programming tasks into 3 and work on each individually but with some collaboration to ensure components worked together properly.   
- Ash is doing the sign-up page and validation.  
- Josh is handling unsubscribing and admin log in.  
- Sam is responsible for the newsletter/alerts and creation of the table.

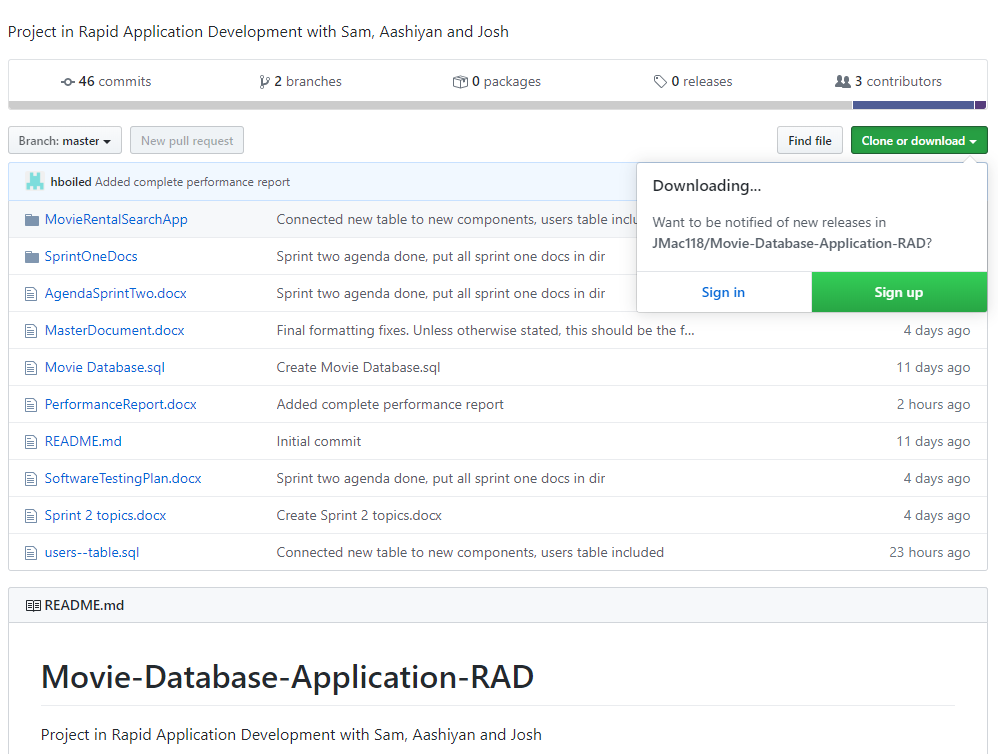
14:20 – We decided to test each implementation individually, with the Scrum Master giving a final review and validation test.

14:25 – Final points made and closing of meeting.

## Source Control:

We have chosen as our source control tool, Git. To host our repository, we are using GitHub.

Josh remained responsible for the source control management as the repository was hosted under his account.



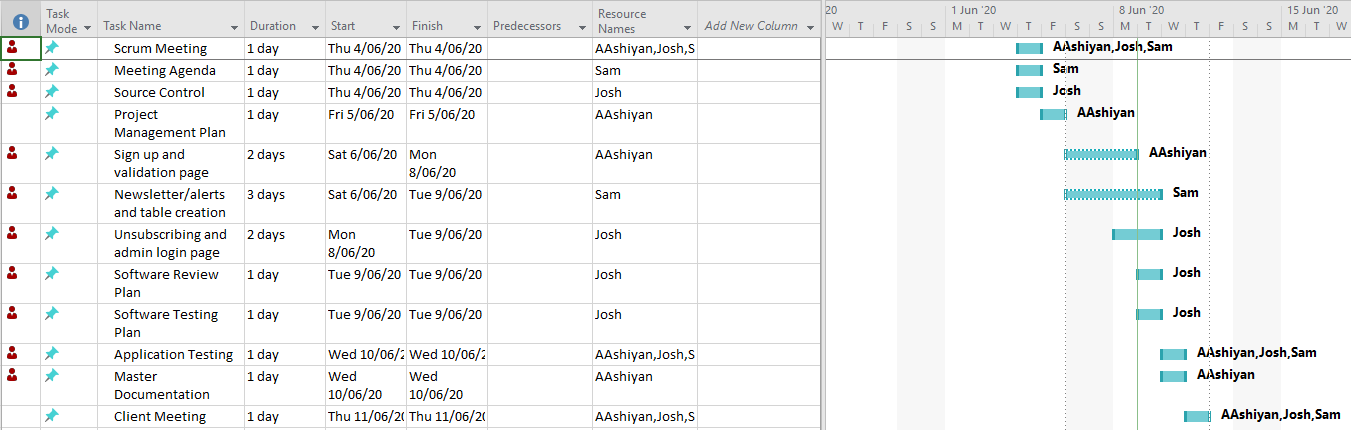
It can be found here:

<https://github.com/JMac118/Movie-Database-Application-RAD>

## Project Management Plan:

**Ash was responsible for planning out the project management plan. Our working progress mirrors that of what was set out in the following Gantt chart and it was further updated when the requirements were completed. The overlap in tasks reflects a period of collaboration.**

**Start Date: 4/06/2020  
Estimated Finish Date: 10/06/2020  
Actual Finish Date: 10/06/2020**



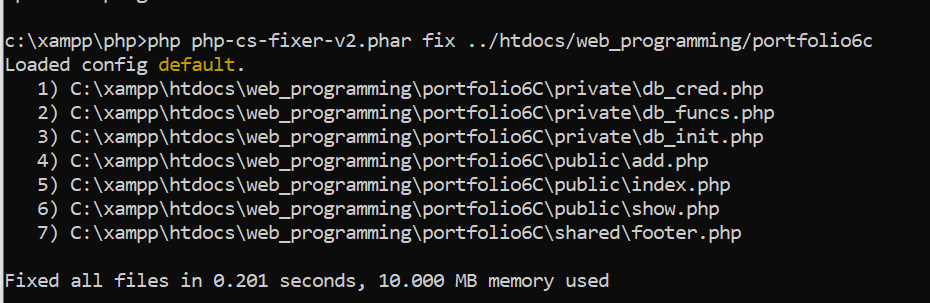
## Performance Report

### Code Formatting:

The PHP Coding Standards Fixer is a good choice for the project for the enforcement of coding standards. This tool is simple to install and use and allows the development team to find all code formatting errors and apply the necessary fixes to them, depending on what coding standard has been selected.

To install, it can be simply downloaded from <https://cs.symfony.com> and extracted to the desired directory. To use, run the command in the terminal as follows:  
- php php-cs-fixer.phar fix “/path/to/dir”   
Where the part in quotation marks is the directory or individual file where the tool will be run.

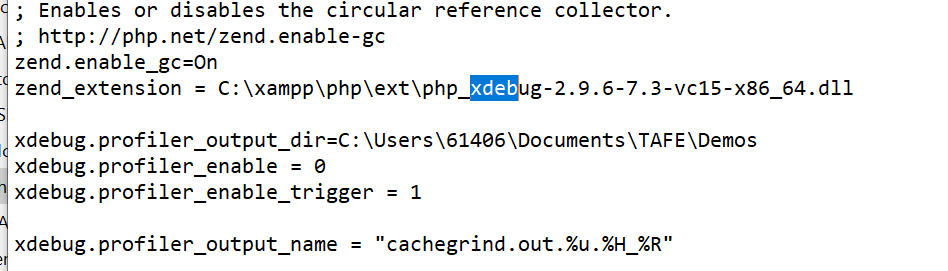
Example of how it is used and the result:



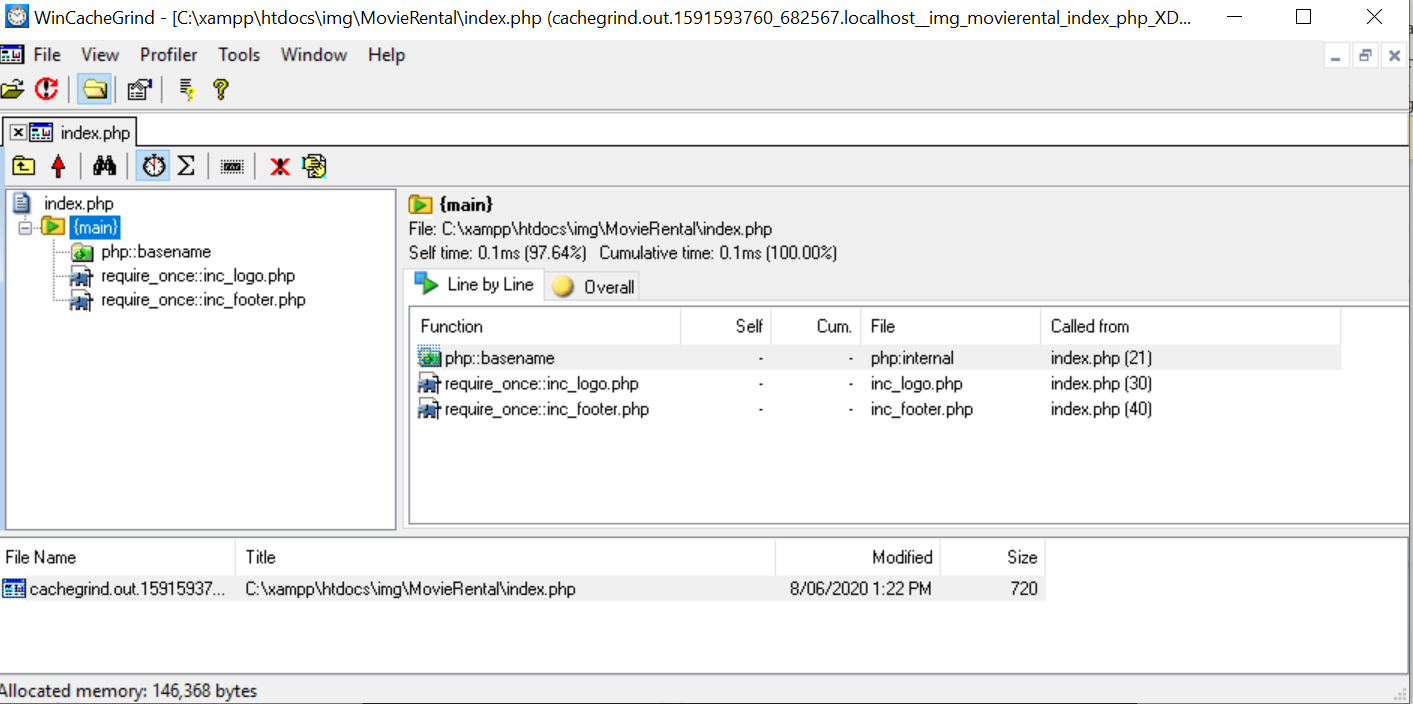
For our project, we will be using the default coding standards because they suitably reformat code to a standard accepted by our team.

### Profiler:

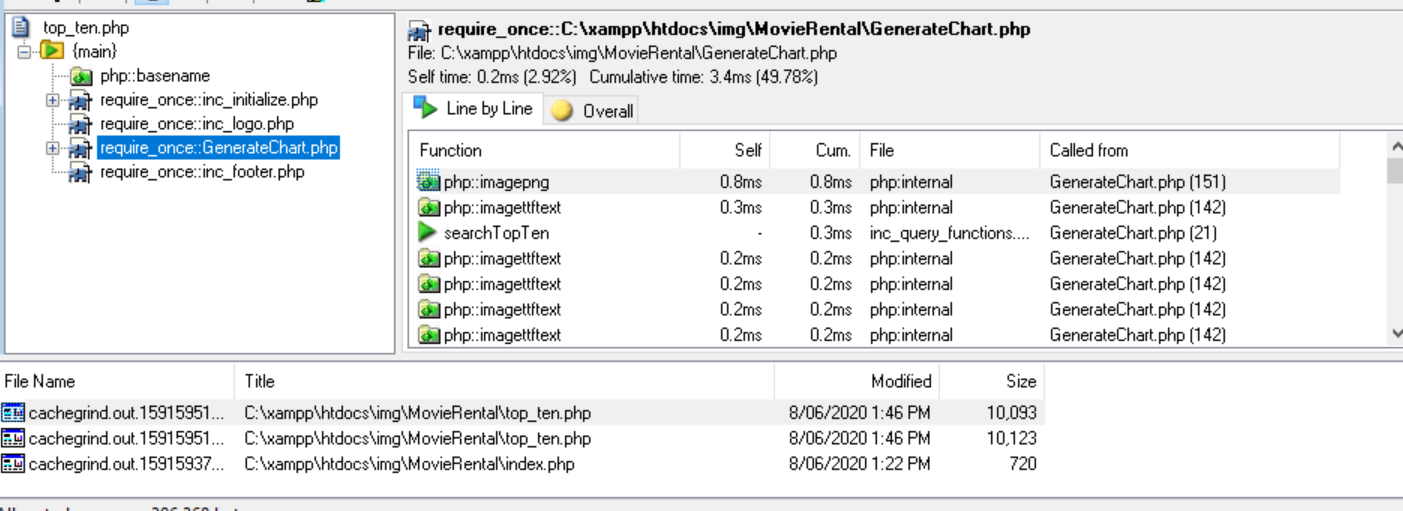
Profiling tools are used to measure the performance of a program by analysing the resources allocated to it, typically per line of code. It is a performance analysis tool which can be used to evaluate a program’s performance and potentially find the root cause of excessive resource usage.

For our project, we used Xdebug, which is an extension PHP debugger which also includes a software profiler. Its installation varies depending on the operating system, but for the machines our team used, which were all Windows, it was required to download the Windows binary corresponding to the installed version of PHP, and configuring the php.ini file to enable use of Xdebug’s profiler like so:  


Running the profiler is dependent on the configuration set in the php.ini file. We set it to trigger on pages where the extension “?XDEBUG\_PROFILE” was appended to the end of the URL. The outcome of the profiling session is stored in a specified directory. This file cannot be read conventionally and requires the use of a 3rd party tool, such as WinCacheGrind for Windows machines.



The analysis indicates that the server is running as expected and there are no performance issues.



Here when the loading of the graph was profiled, we noticed no issues with performance, and everything was running as expected.

### Conclusion:

The tools outlined in this report are useful and can be used to clean up code and detect performance issues. In our project, the code optimiser helped format the code so that it looks neat, presentable and organised. The profiler indicated to us that there were presently no performance issues.

## Software Review Plan

### Functionality

* Search movies by title and genre.
* Search movies by genre, rating, and year.
* Display Top 10 searched films.
* Sign-up page.
* Admin log-in page.
* Admin user display.
* Admin user deletion.
* Admin newsflash send.
* Automated Newsletter sends.
* Responsive Pages

### Ease of use

* Can navigate entire website in 3 clicks.
* Easy menu traversal.

### Design

* Cool colour palette.
* Drop down menu with highlighted options.
* Options in menu are simple and make sense.

### Meeting the Criteria

This website is good to use as it meets all the necessary criteria outlined above. By structuring the website in its current iteration, the navigation of the website is clear, straightforward, and easy. It provides a pleasant experience for its users with its simple application of its required functionality where every task that can be performed, does so with fast results.

With the website’s functionality, it performs all expected tasks and displays the outputs in a nice overview, such as the movie list output in tables. The searches are quite fast as well as the graph generator. The log in page has adequate security, which could be improved upon but functionally completes its task. Behind the log in page, the admin page allows the simple administrative tasks and data access that would be required by an admin.

Website ease-of-use is important to consider when designing a webpage. As such, this site has a simple menu design that allows its users to swap from page to page at any point in its use. It takes only one click to swap from a search to the sign-up page for example.

The design of the website was initially for a function first website, but the colour palette was changed from its original to one that is easier to view and looks more appealing. The drop-down menu highlights the selected option to make it clearer and have been worded simply for an easy understanding of each options purpose.

Regarding the entire website, the design, functionality, and implementation of its pages provides a software system that is good to use.

# Software Testing Plan

## Introduction

### Scope

### In Scope

* Basic website functionality to render site through webhost and connect to DB.
* CITE Quality Assuances practices.
* Application responsiveness to different devices.
* Application responsiveness to window resizing.
* Runtime bugs and issues fixed.
* Newsletter subscription sign-up.
* Administration log-in.
* Subscription deletion possible by admin.
* Monthly Newsletter and Alerts sent to subscribers.

### Out of Scope

* All components and issues not included in the marking guide.

## Quality Objective

**Objectives:**

* Ensure the Application Under Test (AUT) conforms to both functional and non-functional requirements
* Ensure the AUT meets the quality standards defined by the client
* Bugs and issues are identified and fixed before deployment

## Roles and Responsibilities

Detail description of the Roles and responsibilities of different team members like

* Sam Lee – Software Developer, Tester.
* Aashiyan Singh – Project Management, QA Analyst, Software Developer.
* Joshua Macaulay – Test manager, Software Developer, Source Control Manager.

# Test Methodology

## Overview

The software development methodology we have chosen for this project is Rapid Application Development. The main strengths of this methodology is the speed at which the product can be developed, combined with its flexibility in adapting to changing requirements. Being an agile-like methodology, makes it suitable to handle projects where the client can request new requirements at any time during the development process.

The quicker the application is developed, the quicker we can move on and test existing components.

## Test Levels

For the second sprint, we will be testing the new functionality of our application and how it well performs to the general user. To achieve this, the testing methods employed will be the following:

System Testing: A form of black box testing where the tester will go through the application as a whole and test the various functionalities presented in the application. The tester does not know the implementation, to simulate an end user trying the application for the first time. For this project, a test table will be used to record the actual result of each test case.

Acceptance Testing: The formal validation test, where the criteria set out by the client is compared against the application. In RAD, the goal is to make sure the client is satisfied with the product. In this case, when the project satisfies the marking criteria, it will have passed the Acceptance phase.

## Test Completeness

* System Testing – Responsiveness with windows resizing and rendering on different devices is achieved and verified.
* Acceptance Testing – Tests are validated against marking criteria.
* All documentation is completed and verified, then put into the Master Document.

# Test Deliverables

|  |
| --- |
| * Test Plan * System Testing Table * Validation Test Against Marking Criteria * Client Sign Off |

The following will be delivered as part of the completed testing phase, and added to the master document.

# Resource & Environment Needs

## Test Environment and Tools

Test System Specifications:

Processor: Intel® Core™ i5-8250U @ 1.60 GHz, 1.80 GHz  
RAM: 8.00 GB

System Type: 64-bit Operating System, x64-based processor

Required Software:

1. Windows 7 and above
2. Office 2013 and above
3. Google Chrome or Mozilla Firefox
4. Text editor – Visual Studio Code recommended
5. XAMPP – For hosting web server and database

# Terms/Acronyms

|  |  |
| --- | --- |
| **TERM/ACRONYM** | **DEFINITION** |
| API | Application Program Interface |
| AUT | Application Under Test |
| RAD | Rapid Application Development |

# Test Results

## Functionality Test:

- Ensure all functional requirements have been met.

- Website should be easy to navigate, display well, and be relatively fast

- All components related to user subscription service and admin tools should be tested thoroughly.

## Sign-up Page

|  |  |  |  |
| --- | --- | --- | --- |
| Subject | Expected | Actual | Reference |
| 1.1 Sign-up page, Full Name, Email, Newsletter, and Newsflash | Added user subscription to database with newsletter and newsflash = 1 | Meets expectations. | TestEvidence/1.1 TestEvidence/1.1.1 |
| 1.2 Sign-up page, empty full name with other details complete. | Prompt user to enter full name | Meets expectations. | TestEvidence/1.2 |
| 1.3 Sign-up page, empty email with other details complete. | Prompt user to enter email | Meets expectations. | TestEvidence/1.3 |
| 1.4 Sign-up page, email entered but incorrect format, other details complete. | Prompt user to enter an email with example. | Meets expectations. | TestEvidence/1.4 |
| 1.5 Sign-up page, Newsletter not ticked but other details complete | Added user subscription to database with newsletter = 0 but newsflash = 1 | Meets expectations. | TestEvidence/1.5 TestEvidence/1.5.1 |
| 1.6 Sign-up page, Newsflash not ticked but other details complete | Added user subscription to database with newsletter = 1 but newsflash = 0 | Meets expectations. | TestEvidence/1.6 TestEvidence/1.6.1 |

## Admin Log-in

|  |  |  |  |
| --- | --- | --- | --- |
| Subject | Expected | Actual | Reference |
| 2.1 Log-in, correct username and password | Access granted to admin page. | As expected. | 2.1 |
| 2.2 Log-in, empty username field, correct password. | Prompt user to enter username. | As expected. | 2.2 |
| 2.3 Log-in, correct username, empty password field | Prompt user to enter password. | As expected. | 2.3 |
| 2.4 Log-in, incorrect username, correct password. | Warn user password/username combination are incorrect | As expected. | 2.4.1, 2.4.2 |
| 2.5 Log-in, correct username, incorrect password | Warn user password/username combination are incorrect | As expected. | 2.5.1, 2.5.2 |
| 2.6 Log-in, incorrect username, incorrect password | Warn user password/username combination are incorrect | As expected. | 2.6.1, 2.6.2 |

## Display All Users

|  |  |  |  |
| --- | --- | --- | --- |
| Subject | Expected | Actual | Reference |
| 3.1 Display All Users, button selected | Page loaded with all users displayed in a table. | As expected, displays the user data. | 3.1.1, 3.1.2, 3.1.3 |

## Delete Subscription from Database

|  |  |  |  |
| --- | --- | --- | --- |
| Subject | Expected | Actual | Reference |
| 4.1 Delete subscription, user email entered, delete selected | Delete all associated data in the database of that user, alert admin on-screen. | As expected. | 4.1.1, 4.1.2 |
| 4.2 Delete subscription, empty email field, delete selected | Prompt admin to enter an email address. | As expected. | 4.2 |
| 4.3 Delete subscription, non-existent email entered, delete selected. | Alert admin that no such email exists. | As expected. | 4.3 |
| 4.4 Delete subscription, incorrect email format entered, delete selected | Prompt admin to enter an email address in correct format. | As expected. | 4.4 |

## Sending an Alert

|  |  |  |  |
| --- | --- | --- | --- |
| Subject | Expected | Actual | Reference |
| 5.1 Sending an Alert, subject entered, message entered, submit selected | Alert email sent out to all newsflash subscribers. | Meets expectations. | TestEvidence/5.1  TestEvidence/5.1.1 |
| 5.2 Sending an Alert, empty subject field, message entered, submit selected | Prompt admin to enter subject. | Meets expectations. | TestEvidence/5.2 |
| 5.3 Sending an Alert, subject entered, empty message field, submit selected | Prompt admin to enter message. | Meets expectations. | TestEvidence/5.3 |

## Sending a Newsletter

|  |  |  |  |
| --- | --- | --- | --- |
| Subject | Expected | Actual | Reference |
| 6.1 Sending a Newsletter, automated requirements reached. | Newsletter email sent out to all newsletter subscribers. | Meets expectations. NOTE: task scheduler was adjusted to run 1 minute later instead of monthly, for testing purposes. | TestEvidence/6.1  TestEvidence/6.1.1 |

## Logging out

|  |  |  |  |
| --- | --- | --- | --- |
| Subject | Expected | Actual | Reference |
| 7.1 Logging out, button selected | Session terminated, redirects user to home page. | Log out button sends user to home page, if reloading admin page requires another login | 7.1.1, 7.1.2, 7.1.3 |

## Accessing admin pages with no log-in

|  |  |  |  |
| --- | --- | --- | --- |
| Subject | Expected | Actual | Reference |
| 8.1 Accessing admin page, no log-in | Sent to home page. | Meets expectation. Note: Developer tools was used to record redirect from admin page to index. | TestEvidence/8.1 |
| 8.2 Accessing Display all users page, no log-in | Sent to home page. | Meets expectation. Note: Developer tools was used to record redirect from list users to index. | TestEvidence/8.2 |

# Bibliography

ProfessionalQA. (2020, April 19). Retrieved from https://www.professionalqa.com/software-review