Table of Contents

[Introduction: 1](#_Toc44502815)

[Sprint One - Meeting Agenda 2](#_Toc44502816)

[Additional Information: 2](#_Toc44502817)

[Source Control: 3](#_Toc44502818)

[Project Management Plan: 4](#_Toc44502819)

[Software Testing Report: 5](#_Toc44502820)

[**CITE Business Analysis:** 6](#_Toc44502821)

[CITE Business Rules for software development 6](#_Toc44502822)

[QMS Tasks and Objectives 6](#_Toc44502823)

[CITE Managed Services QA: Comprehensive Approach to Quality 6](#_Toc44502824)

[Quality Planning 6](#_Toc44502825)

[Quality Assurance 6](#_Toc44502826)

[Quality Control 6](#_Toc44502827)

[Acme Entertainment Pty Ltd Development Requirements: 6](#_Toc44502828)

[**Multi-Platform Report:** 7](#_Toc44502829)

[Web Design Methods 7](#_Toc44502830)

[Adaptive Web Design 7](#_Toc44502831)

[Responsive Web Design 8](#_Toc44502832)

[Our Pick 8](#_Toc44502833)

[Software Testing Plan: Sprint One: 9](#_Toc44502834)

[Scope 9](#_Toc44502835)

[In Scope 9](#_Toc44502836)

[Out of Scope 9](#_Toc44502837)

[Quality Objective 9](#_Toc44502838)

[Objectives: 9](#_Toc44502839)

[Roles and Responsibilities 9](#_Toc44502840)

[Test Methodology 10](#_Toc44502841)

[Overview 10](#_Toc44502842)

[Test Levels 10](#_Toc44502843)

[Test Completeness 10](#_Toc44502844)

[Test Deliverables 11](#_Toc44502845)

[Resource & Environment Needs 12](#_Toc44502846)

[Test Environment and Tools 12](#_Toc44502847)

[Test Results: 13](#_Toc44502848)

[System Test Components: 13](#_Toc44502849)

[1. MotoG4 Mobile Device: 13](#_Toc44502850)

[2. iPad Device: 14](#_Toc44502851)

[3. Laptop Device: 15](#_Toc44502852)

[Acceptance Test: 16](#_Toc44502853)

[Marking Guidelines Verification: 16](#_Toc44502854)

[Introduction: 17](#_Toc44502855)

[Sprint Two Meeting Agenda 18](#_Toc44502856)

[Additional Information 18](#_Toc44502857)

[Source Control: 19](#_Toc44502858)

[Project Management Plan: 20](#_Toc44502859)

[Performance Report 21](#_Toc44502860)

[Code Formatting: 21](#_Toc44502861)

[Profiler: 21](#_Toc44502862)

[Conclusion: 22](#_Toc44502863)

[Software Review Plan 23](#_Toc44502864)

[Functionality 23](#_Toc44502865)

[Ease of use 23](#_Toc44502866)

[Design 23](#_Toc44502867)

[Meeting the Criteria 23](#_Toc44502868)

[Software Testing Plan 24](#_Toc44502869)

[Scope 24](#_Toc44502870)

[In Scope 24](#_Toc44502871)

[Out of Scope 24](#_Toc44502872)

[Quality Objectives 24](#_Toc44502873)

[Roles and Responsibilities 24](#_Toc44502874)

[Test Methodology 25](#_Toc44502875)

[Overview 25](#_Toc44502876)

[Test Levels 25](#_Toc44502877)

[Test Completeness 25](#_Toc44502878)

[Test Deliverables 26](#_Toc44502879)

[Resource & Environment Needs 27](#_Toc44502880)

[Test Environment and Tools 27](#_Toc44502881)

[Test Results 28](#_Toc44502882)

[Functionality Test: 28](#_Toc44502883)

[Sign-up Page 28](#_Toc44502884)

[Admin Log-in 29](#_Toc44502885)

[Display All Users 29](#_Toc44502886)

[Delete Subscription from Database 29](#_Toc44502887)

[Sending an Alert 30](#_Toc44502888)

[Sending a Newsletter 30](#_Toc44502889)

[Logging out 30](#_Toc44502890)

[Accessing admin pages with no log-in 30](#_Toc44502891)

[Introduction: 31](#_Toc44502892)

[Sprint Three Meeting Agenda 32](#_Toc44502893)

[Event Heading 32](#_Toc44502894)

[Additional Information: 32](#_Toc44502895)

[Source Control: 33](#_Toc44502896)

[Project Management Plan: 34](#_Toc44502897)

[Software Testing Report: 35](#_Toc44502898)

[Optimisation Report 36](#_Toc44502899)

[Introduction 36](#_Toc44502900)

[Optimising our Website: 36](#_Toc44502901)

[Database: 36](#_Toc44502902)

[Mail: 36](#_Toc44502903)

[Site Traffic: 36](#_Toc44502904)

[Code Optimisation: 37](#_Toc44502905)

[Software Testing Plan: Sprint One: 38](#_Toc44502906)

[Scope 38](#_Toc44502907)

[In Scope 38](#_Toc44502908)

[Out of Scope 38](#_Toc44502909)

[Quality Objective 38](#_Toc44502910)

[Test Methodology 39](#_Toc44502911)

[Overview 39](#_Toc44502912)

[Test Levels 39](#_Toc44502913)

[Test Completeness 39](#_Toc44502914)

[Test Deliverables 40](#_Toc44502915)

[Resource & Environment Needs 41](#_Toc44502916)

[Test Environment and Tools 41](#_Toc44502917)

[Test Results: 42](#_Toc44502918)

[System Test Components: 42](#_Toc44502919)

[1. Movie Ratings 42](#_Toc44502920)

[2. Analytics: 42](#_Toc44502921)

[3. Staff Log-in 43](#_Toc44502922)

[Acceptance Test: 44](#_Toc44502923)

[Marking Guidelines Verification: 44](#_Toc44502924)

[References 47](#_Toc44502925)

[Glossary: 48](#_Toc44502926)

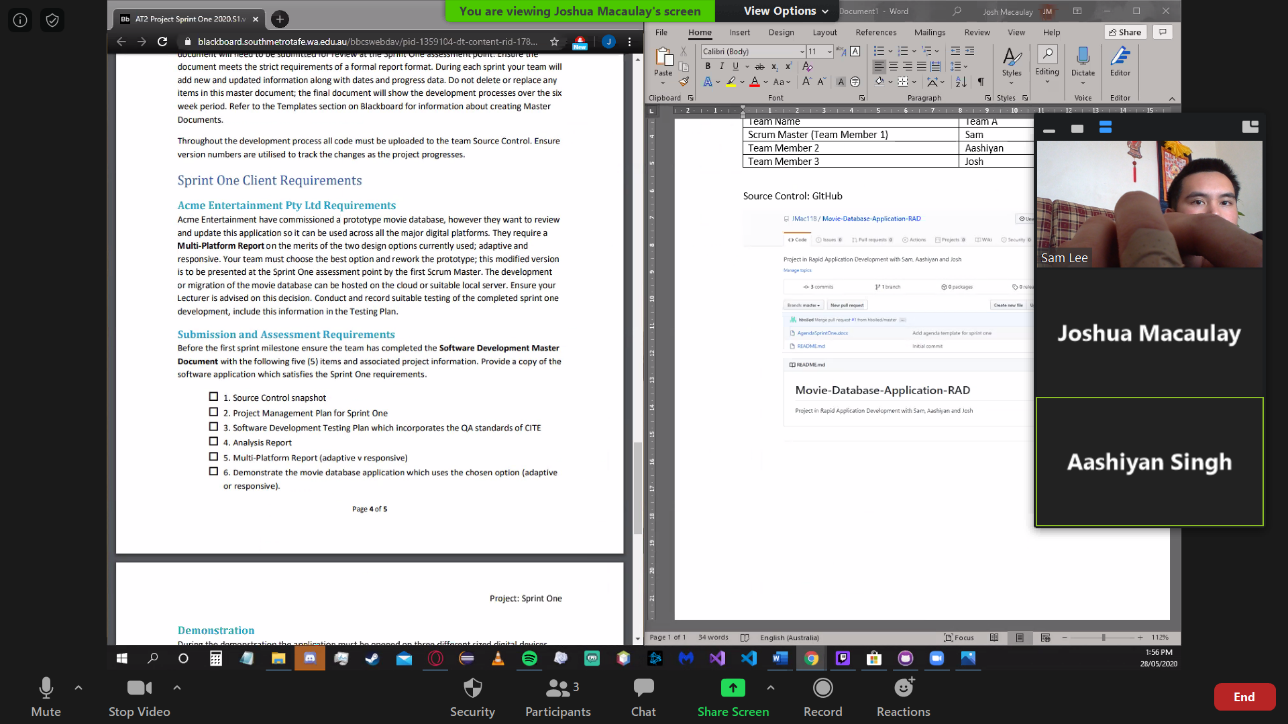
Sprint One

**Version 1.1:  
Last Edited 04/06/2020**

# Introduction:

Team A – Members for first sprint:  
  
The roles for this sprint have been determined as follows.  
- Sam Lee: Scrum Master, Testing Manager   
- Aashiyan Singh: Platform Planner, QA Analyst  
- Joshua Macaulay: Software Developer, Source Control Manager

This was decided at the meeting we held on the 28th of May, 2020.  
  
On the following page is the meeting agenda, which summarises the outcome of the meeting and details the delegation of tasks.  
  
Evidence of Meeting taking place:



# Sprint One - Meeting Agenda

28/05/2020

1:30

Meeting called by: Sam

Attendees: Sam, Aashiyan, Josh

Please read: Project Document

Please bring: Coffee  
Platform: Zoom

|  |  |  |
| --- | --- | --- |
| Time | Event Heading | Location |
| 1:30 -3:00 | Scrum Meeting | Zoom |

## Additional Information:

During these meeting we discussed which team member will be working on each aspect of the project.

At 1:35 we decided to use GitHub as our means of source control.

At 1:45 the Software Development Testing Plan was designated to Sam and since he is the scrum master for sprint one, he will also do the Project Management Plan.

Aashiyan was designated to do the Analysis Report at 2:10 and shortly afterwards was also given the Multi-Platform Report to complete.

At 2:25 it was decided that Josh would do the implementation of the Responsive update for the website.

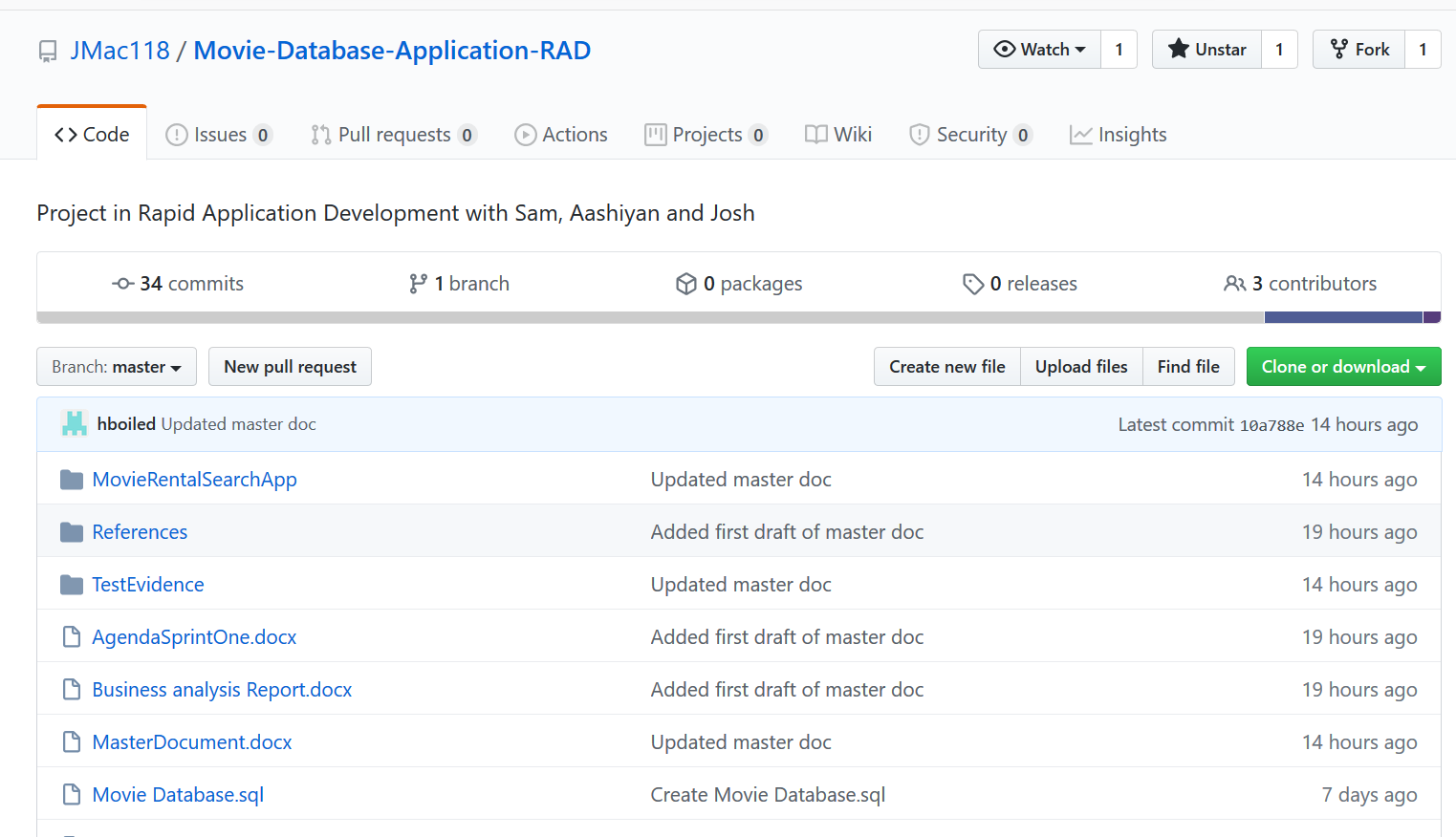
The rest of the meeting was just to ensure each team member knew what they needed to complete with a short questions and answers session.

# Source Control:

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

Joshua was responsible for the creation and maintenance of our source control repository, so the repository is hosted on his GitHub account.

Snapshot:



It can be found here:

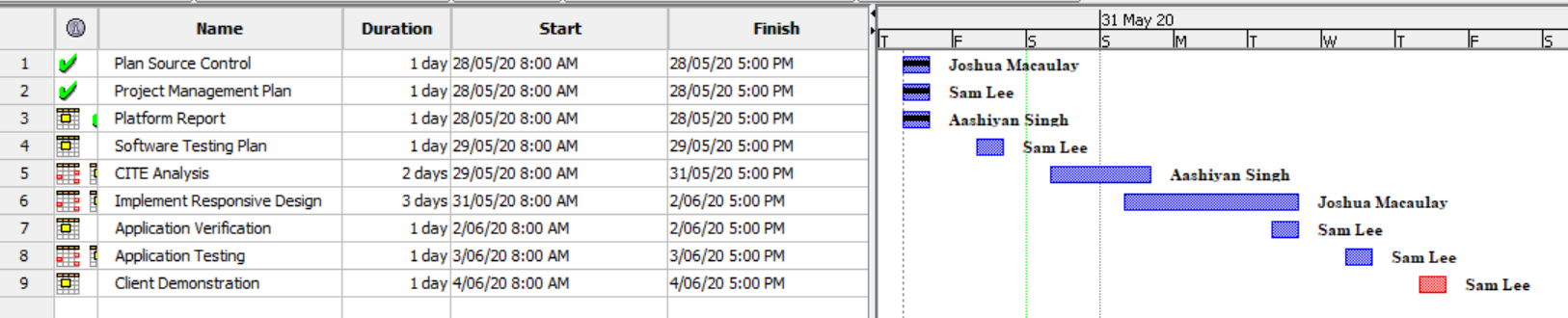
<https://github.com/JMac118/Movie-Database-Application-RAD>

# Project Management Plan:

**Sam 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.**

**While Joshua was implementing the responsive design, Sam was verifying the application against responsive design guidelines and the marking criteria, as can be seen in tasks 6 to 7.**

**Start Date: 28/05/2020  
Estimated Finish Date: 03/06/2020  
Actual Finish Date: 03/06/2020**

****

# Software Testing Report:

**The way we conduct the testing of our software will be detailed specifically in Testing Plan Documents for each sprint. The following pages will include a copy of the testing plan Sam has prepared for the first sprint.**

**Our testing and quality assurance practices will closely follow those specified by CITE Managed Services. Aashiyan has prepared a business analysis document detailing the software development, quality control and quality assurance practices of CITE to support our testing plans.**

**The criteria detailed in this testing report will be applied to all subsequent testing plans for each sprint.**

* **On the most fundamental level, all existing components of the application must be functional, which in the case of this project, are:  
  - the basic searching operations using the movie database are available and working  
  - GUI and elements are fully legible**
* **Any bugs and issues must be identified and reported so that they can be fixed during the following sprint.**
* **ACME ltd. Development Requirements are satisfied:  
  - Website has responsive or adaptive design  
  - Tests based on design choice have been run using the 3 major platforms (Desktop/Laptop, mobile, tablet) and are successful**
* **Client is satisfied with the delivered test results and application and signs off on it.**

# **CITE Business Analysis:**

## CITE Business Rules for software development

CITE Managed Services has implemented a Quality Management System (QMS) which provides and ensures the highest quality when it comes to comprising a complex set of engineering and managerial activities.

### QMS Tasks and Objectives

* To implement the procedures and regulations based on the industry standards for software deployment.
* To verify and validate the product to that it complies with clients’ business needs and expectations.
* To monitor the product lifecycle to ensure compliance with established processes and guidelines
* To establish a heathy and friendly environment between all project team members.

## CITE Managed Services QA: Comprehensive Approach to Quality

### Quality Planning

CITE Managed Services starts with planning applicable set of standards, regulations, procedures, guidelines, and tools in each project’s development lifecycle.

### Quality Assurance

To assure that quality standards are being followed and customer requirements are being met, CITE Managed Services have established processes that evaluate project performance.

### Quality Control

To identify defective pieces of code, CITE Managed Services measure performance trends to verify that product is of high quality, complete and correct.

## Acme Entertainment Pty Ltd Development Requirements:

Acme Entertainment Pty Ltd has assigned our team with the task of implementing a movie database. Our team needs to create a multi-platform report and choose one of the two design options currently used, which are adaptive and responsive. In this report, we will be explaining the two web designs and will also be comparing the two to figure out the advantages and disadvantages of the two design methods. In the end, we will choose either adaptive or responsive method to rework our prototype.

# **Multi-Platform Report:**

## Web Design Methods

### Adaptive Web Design

Adaptive web design was first introduced by a web designer named Aaron Gustafson in 2011. In adaptive approach, the sites are designed to have multiple fixed layout sizes. Depending on the amount of space available, the site detects and picks one of the prefixed layouts that fits the screen. For example, if the site is opened on a desktop browser, it will pick the layout best suited for a desktop screen, resizing the browser will not affect the layout of the web page.

Many websites use adaptive design. Some of which are Amazon, USA today, About.com and Apple. They mostly use six prefixed screen widths: 320, 480, 760, 960, 1200, 1600.

Adaptive design has the best user experience on all the devices because unlike responsive design, where the desktop design works into the smaller devices, adaptive design offers to have custom designs for desktop, mobile, tablet and any other devices that the site can be opened. Designers can design different buttons, navigation tools and other interface tools based on users’ needs for a device. Adaptive websites often outperform websites with responsive design. They are usually 2-3 times faster than the responsive ones as it gives less data to the user for it to deliver a better user experience. In adaptive design, there is also the option to create mobile only websites which is denoted with “m.” in the URL bar.

Adaptive design has some strong advantages, but it also comes with some drawbacks. Firstly, it requires a lot more work to create an adaptive design than responsive design. So many designers try to retrofit the existing websites to make them more accessible. It requires a lot of work, which means it needs a large team to maintain, which make the cost of creating adaptive design more expensive than of responsive. Adaptive design is also less flexible as if a new device is launched with a screen size different than the existing one, that could create problems. Which means the designer either must create a new layout or edit the existing ones. They require much more maintenance in the long run than responsive design.

### Responsive Web Design

Responsive web design was first introduced by a web designer and developer named Ethan Marcotte (Graham, 2015). In responsive approach, the site uses just one layout for the site and adjusts to better fit the screen of the user’s device. It uses flexible grids and layout to present the information of the website. For example, if you open a responsive website on the desktop browser and try to adjust the size of the browser, the website will try to arrange the contents to fit the browser window. For smaller devices such as phones, the website checks for the available space and fits the content to the size.

Many websites such as Dropbox, GitHub, Shopify etc use responsive web design.

Responsive design is much easier and requires much less work to implement and maintain than adaptive design. It will cut down the time and cost to maintain and update the website. Designers mainly need to create a single design for the website that can be used on all devices. It can also be more user friendly as its mostly the same on all devices and users will get the same experience on different devices. Responsive websites are also much more search engine friendly as they have the same URL that serves all devices (Merlin, 2018).

Responsive designs drawback could be that the websites might be slower on different devices. Images on the website could affect the loading times of a website. The same website that opens quickly on a desktop might take much longer on mobile device or a tablet. Putting ads on the website could also be big challenge. As the website will flow on different devices, ads might not configure properly and adjust to the screen.

## Our Pick

**We picked Responsive web design as our primary approach because it is easier to manage and implement. We also used some adaptive design choices to fix some issues with our display table.**

# **Software Testing Plan: Sprint One:**

## 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.

### 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 – Test manager, Tester, Project Management.
* Aashiyan Singh – QA Analyst, Platform Planning.
* Joshua Macaulay – 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 first sprint, we will be testing the visual responsiveness of our application and how it performs on different devices. 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 Report (For whole project) * 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

# Test Results:

Screenshots are provided in the directory “TestEvidence”. For each entry in the table, an id number will prefix its corresponding screenshot. Screenshots have been provided outside this document to make it neater and more organised.

## System Test Components:

### 1. MotoG4 Mobile Device:

|  |  |  |  |
| --- | --- | --- | --- |
| Subject | Expected | Actual | Reference |
| Home page, scaled 100%, vertical view | Responds to mobile device and renders mobile friendly view. | Meets expectation. | TestEvidence/1.1 |
| Home page, scaled 60% (to fit), vertical view | Whole screen renders to mobile friendly view and footer is fixed at bottom. | Meets expectation. | TestEvidence/1.2 |
| Home page, scaled 95% horizontal view | Whole screen renders to mobile friendly view horizontally, footer fixed at bottom. | Meets expectation. | TestEvidence/1.3 |
| Search result, scaled 100%, vertical view | Results table clearly visible with responsive design detecting mobile device and not rendering extra columns so all fits. | Meets expectation. | TestEvidence/1.4 |
| Browse results, scaled 60% (to fit) vertical view | Results table visible, footer fixed at bottom but not obscuring results. | Meets expectation. | TestEvidence/1.5 |
| Browse results, scaled 95%, horizontal view | Results table visible, wider horizontal dimension detected and adjusted column display to include more columns. | Meets expectation. | TestEvidence/1.6 |
| Top 10 graph, scaled 95%, horizontal view | Graph image is scaled to device and alignment. | Meets expectation. | TestEvidence/1.7 |
| Top 10 graph, scaled 60% (to fit), vertical view | Graph image is scaled to device and alignment. | Meets expectation. | TestEvidence/1.8 |

### 2. iPad Device:

|  |  |  |  |
| --- | --- | --- | --- |
| Subject | Expected | Actual | Reference |
| Home page, scaled 100%, vertical view | Whole screen renders to iPad friendly view and footer is fixed at bottom. | Meets expectation. | TestEvidence/2.1 |
| Home page, scaled 40% (to fit), vertical view | Responds to iPad device and renders iPad friendly view. | Meets expectation. | TestEvidence/2.2 |
| Home page, scaled 50% horizontal view | Whole screen renders to iPad friendly view horizontally, footer fixed at bottom. | Meets expectation. | TestEvidence/2.3 |
| Search result, scaled 100%, vertical view | Results table clearly visible with responsive design detecting iPad device and not rendering the extra column so all fits. | Meets expectation. | TestEvidence/2.4 |
| Browse results, scaled 40% (to fit) vertical view | Results table visible, footer fixed at bottom but not obscuring results. | Meets expectation. | TestEvidence/2.5 |
| Browse results, scaled 75%, horizontal view | Results table visible, wider horizontal dimension detected and adjusted column display to include one more column. | Meets expectation. | TestEvidence/2.6 |
| Top 10 graph, scaled 95%, horizontal view | Graph image is scaled to device and alignment. | Meets expectation. | TestEvidence/2.7 |
| Top 10 graph, scaled 60% (to fit), vertical view | Graph image is scaled to device and alignment. | Meets expectation. | TestEvidence/2.8 |

### 3. Laptop Device:

|  |  |  |  |
| --- | --- | --- | --- |
| Subject | Expected | Actual | Reference |
| Home page standard view | Displays full website, fit into single view. | Meets expectation. | TestEvidence/3.1 |
| Search results standard view | Displays all columns for search query. | Meets expectation. | TestEvidence/3.2 |
| Browse results standard view | Table of results fits view; footer does not obscure bottom results. | Meets expectation. | TestEvidence/3.3 |
| Top 10 graph standard view | Graph enlarged to reflect larger device, still viewable and scales well. | Meets expectation. | TestEvidence/3.4 |
| Top 10 graph resized window | Graph and other elements resize dynamically against changing window size. | Meets expectation. | TestEvidence/3.5 |

# Acceptance Test:

## Marking Guidelines Verification:

* Application functions correctly on multiple platforms:  
    
  This has been demonstrated in the test documentation, “Test Results”. Screenshots which serve as reference and proof of functionality are supplied in the directory “TestEvidence”. The screenshots have been labelled and with a descriptive name and are prefixed with a corresponding number identified in the test table.
* All documentation is properly formatted:  
    
  The master documented has been formatted so that each heading is on a new page. Headers, footers, table of contents and a cover page are included to assist with the organisation of the document. The master document feature was not used because of its tendency to corrupt and make the file unusable.
* Ensure all topics are adequately covered using common terminology:  
    
  The documents “Multi-Platform Report” and “Business Analysis Report” have been included in the master document. The former covers responsive versus adaptive web design and makes a choice as to which design, we have utilised. The latter covers software development, quality control and quality assurance practices relating to CITE Managed Services. These practices are integrated into our testing and quality assurance plan.
* All testing has been documented:  
    
  A test table has been constructed for the various platforms with various test cases. Each test case was carried out, an expectation was set for the outcome, and then verified against a screenshot provided in the TestEvidence directory which shows the actual outcome. This document is included in the master document.

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)

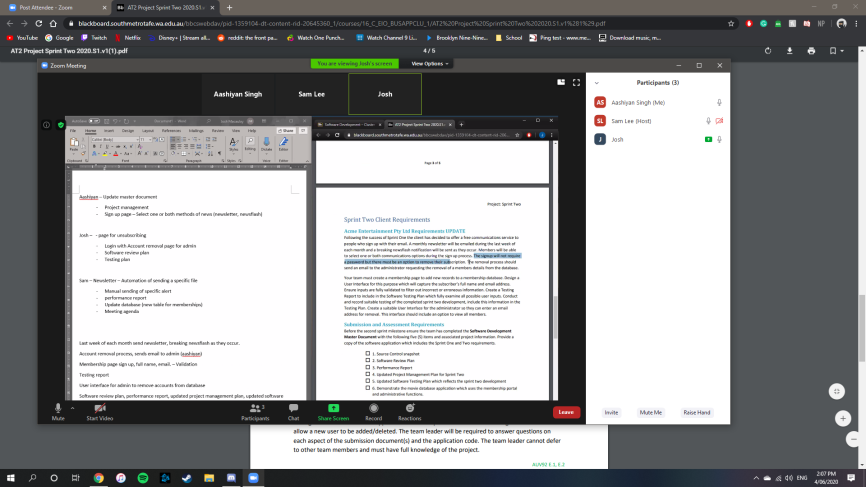
Josh

* 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:  


# Sprint Two Meeting Agenda

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.

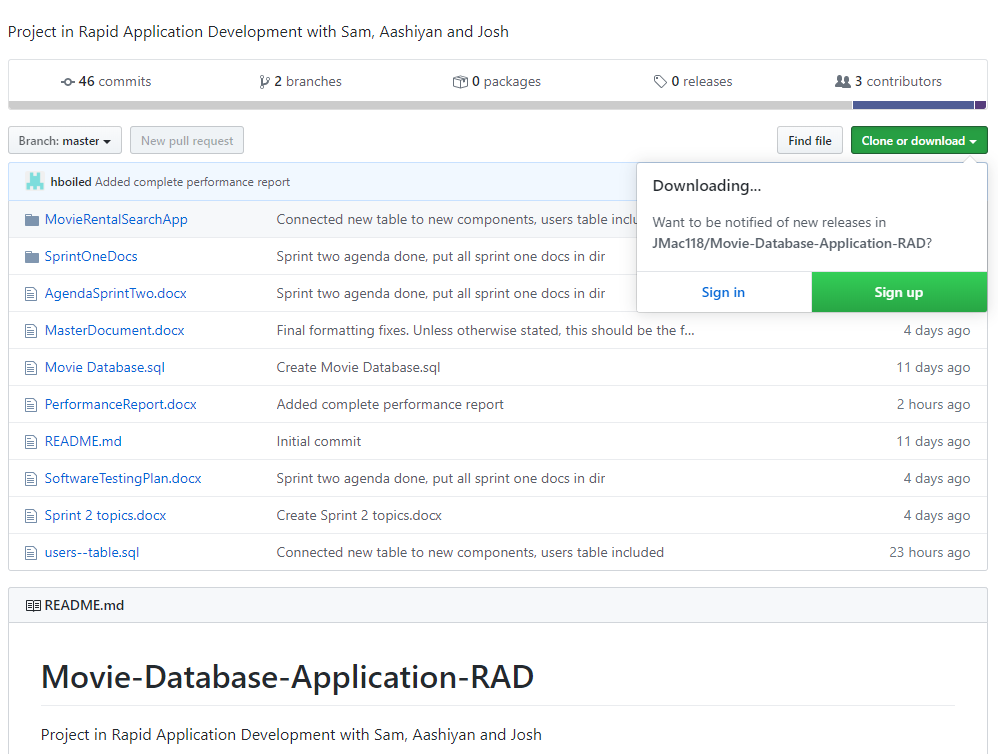
14:20 – We decided to test each implementation individually, with the Scrum Master giving a final review and validation test.

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.



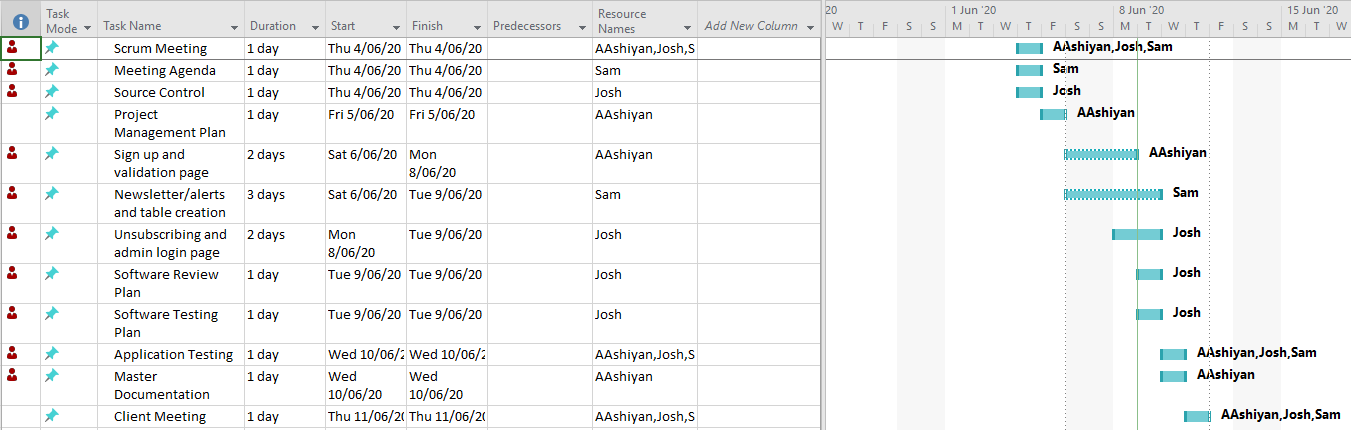
It can be found here:

<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**



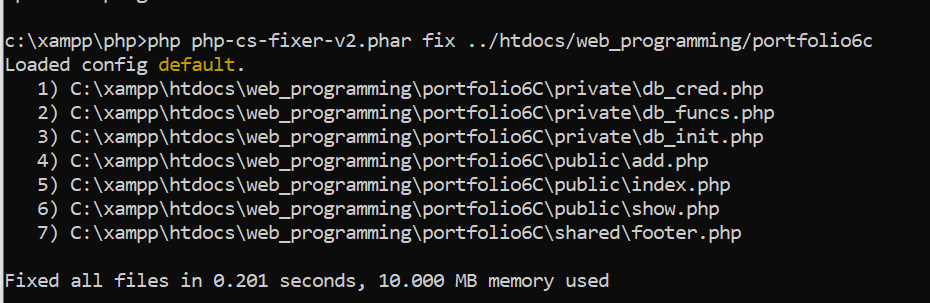
# Performance Report

## 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.

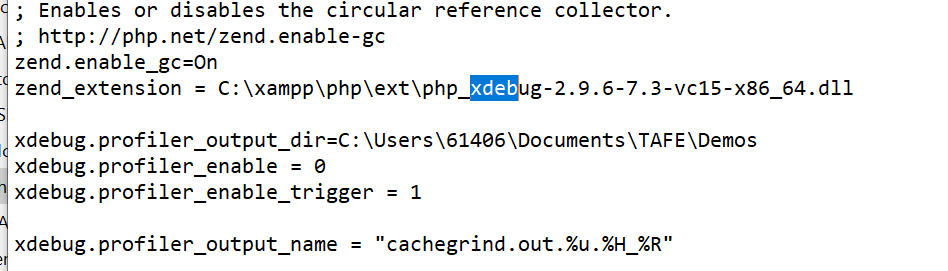
Example of how it is used and the result:



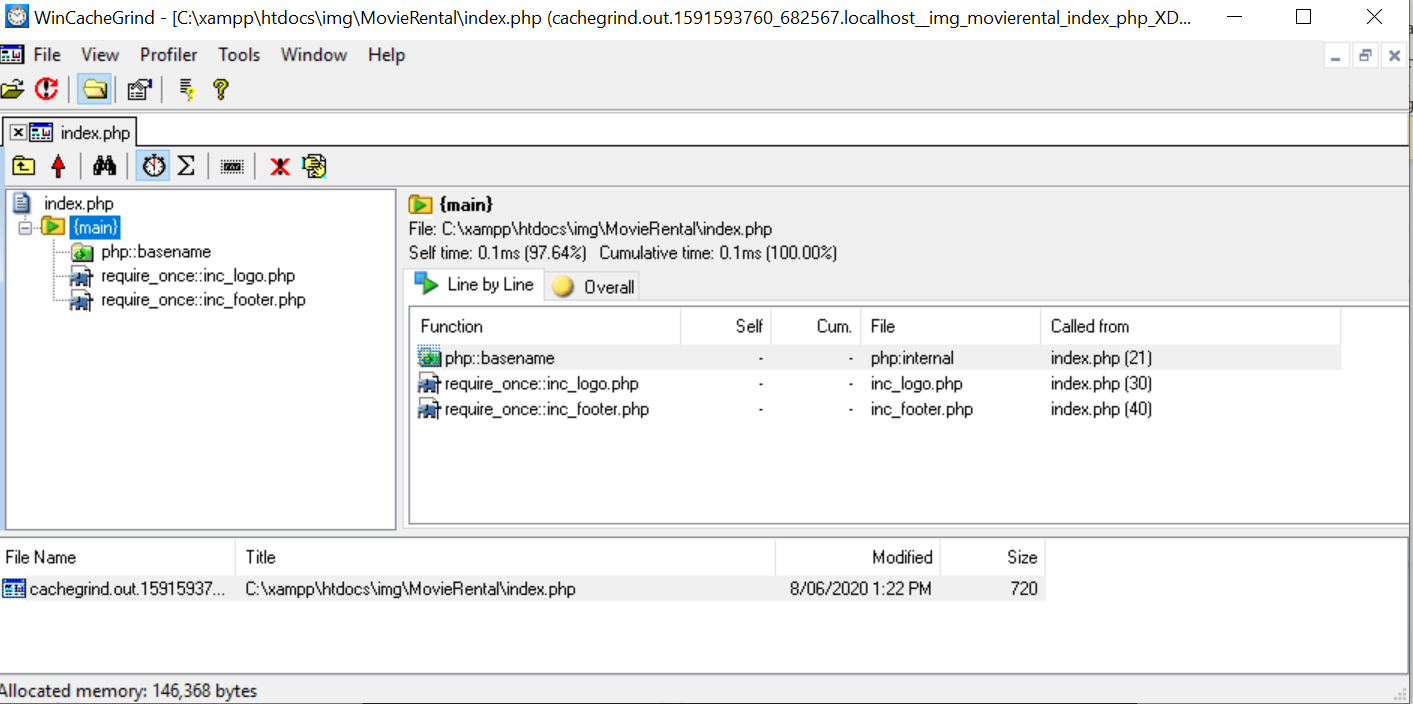
For our project, we will be using the default coding standards because they suitably reformat code to a standard accepted by our team.

## Profiler:

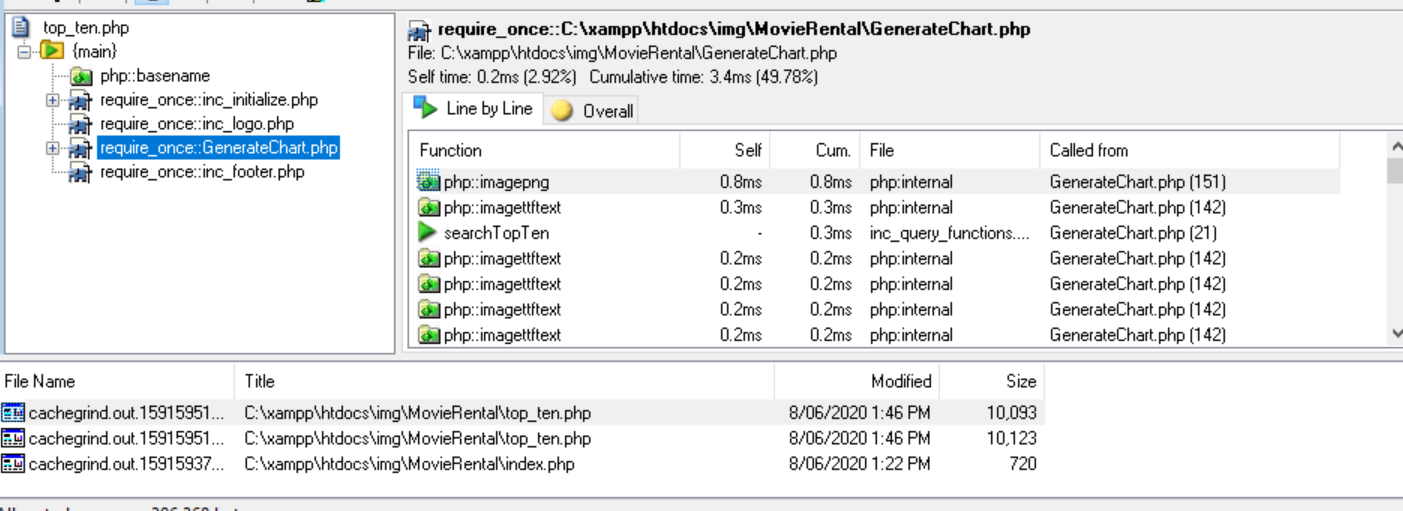
Profiling tools are used to measure the performance of a program by analysing the resources allocated to it, typically per line of code. It is a performance analysis tool which can be used to evaluate a program’s performance and potentially find the root cause of excessive resource usage.

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

## 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 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

# 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 |

Sprint Three

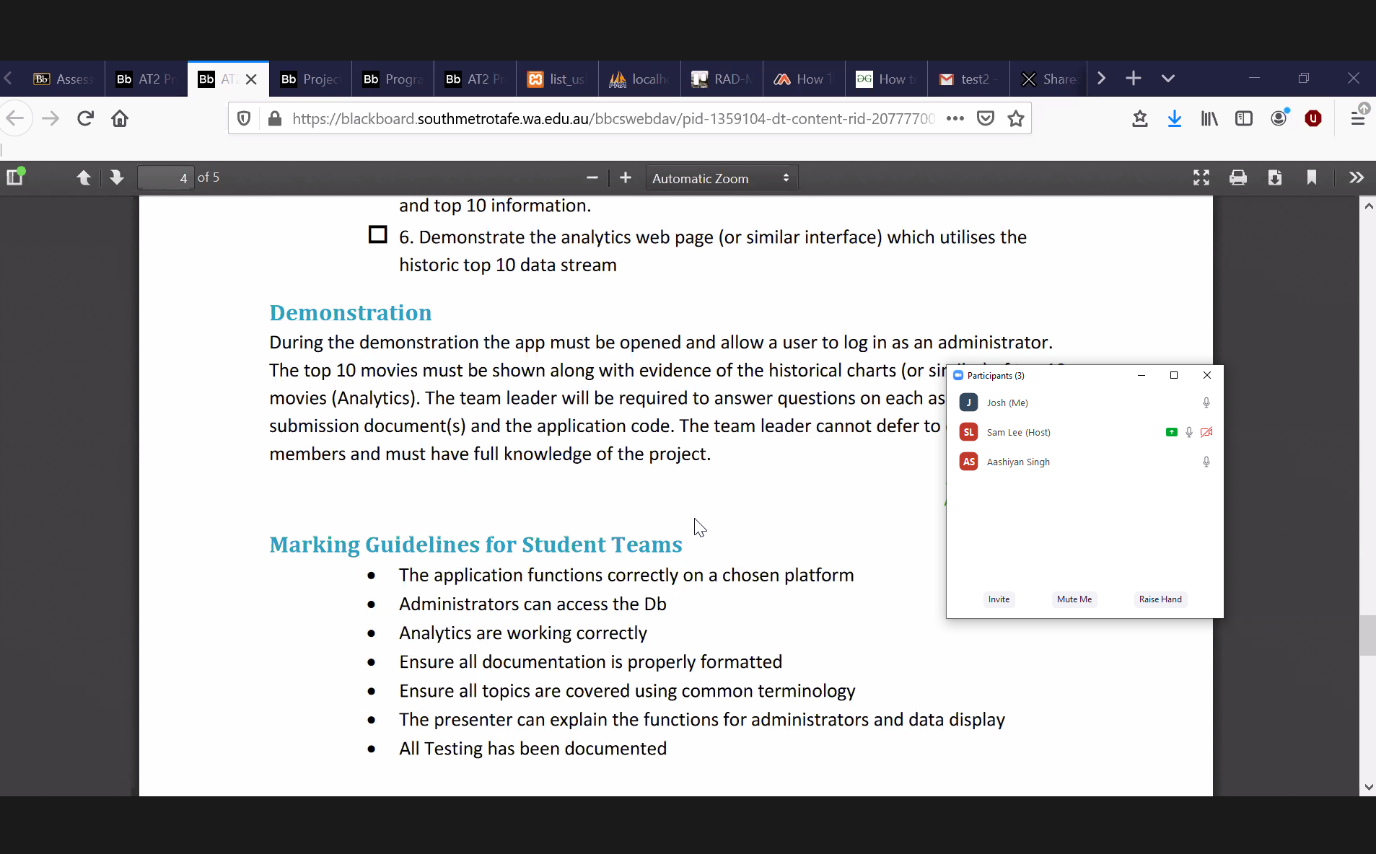
**Version 1.3:  
Last Edited 17/06/2020**

# Introduction:

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

-Joshua Macaulay: Scrum Master, Source Control Manager  
- Sam Lee: Software Developer, Analytics  
- Aashiyan Singh: Software Developer, Optimisation

This was decided at the meeting we held on the 11th of June, 2020.  
  
On the following page is the meeting agenda, which summarises the outcome of the meeting and details the delegation of tasks.  
  
Evidence of Meeting taking place:



# Sprint Three Meeting Agenda

11/06/2020

14:00

Meeting called by: Joshua

Attendees: Aashiyan, Sam, Josh

Please read: RAD Project Sprint 3 Requirements

Please bring: Supplies

Platform: Zoom

|  |  |  |
| --- | --- | --- |
| Time | Event Heading | Location |
| 14:00 – 15:05 | Scrum Meeting | Zoom - Online |

## Additional Information:

Discussion Topics:

14.00 – Josh took the responsibility of being the scrum leader and doing the master document.

14:15 – Josh stayed responsible for the source control.

14:25 – Ash was tasked with handling the meeting agenda

14:35 – The team discussed all the tasks that needed to be done.

14: 50 – After discussing all the tasks, we decided to divide up the tasks:

* Ash is doing the Top Ten ratings, Optimisation report and the Testing plan.
* Josh is handling the Master Doc, password complexity, Database update and updates to the project management
* Sam is responsible for Streaming Table and the Rating form.

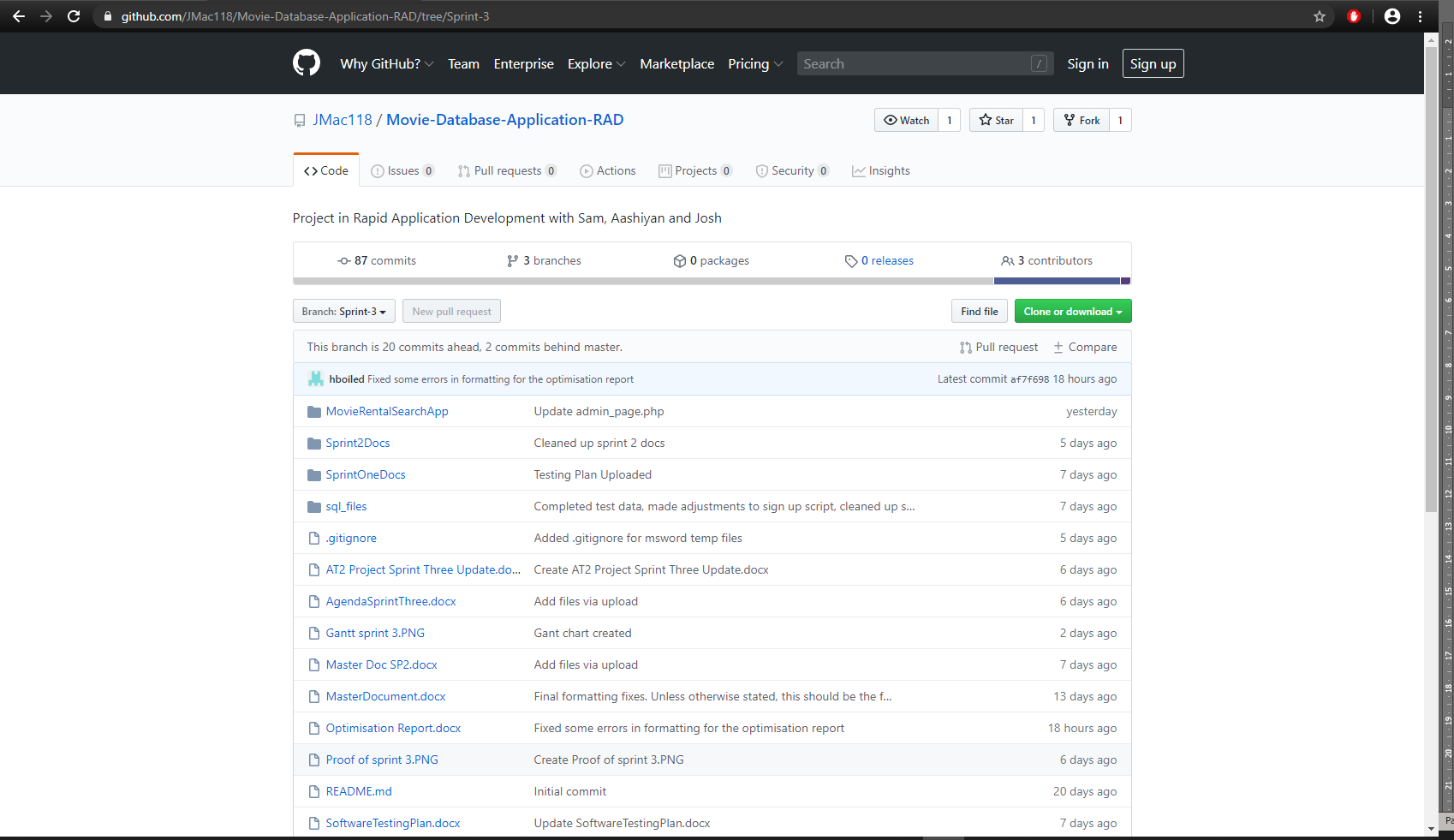
15:05 – Meeting evidence was taken, and the meeting ended.

# Source Control:

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

Joshua was responsible for the creation and maintenance of our source control repository, so the repository is hosted on his GitHub account.

Snapshot:



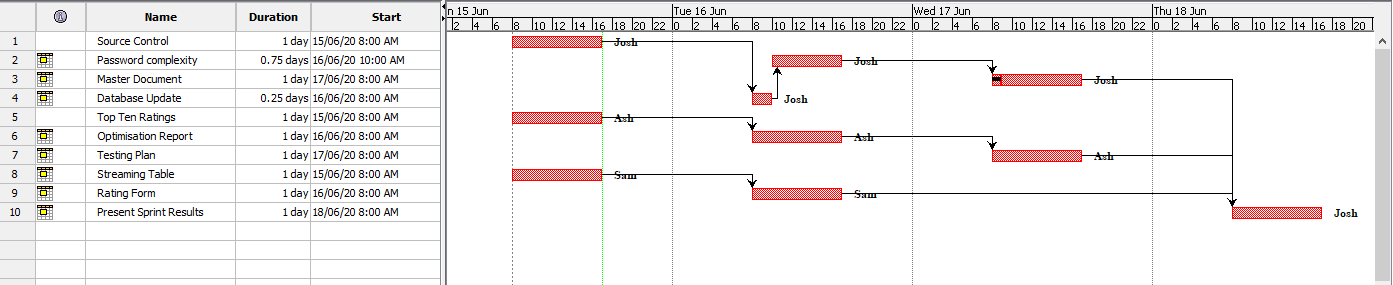
It can be found here:

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

# Project Management Plan:

On this third sprint, Josh was responsible for the project management. A Gantt chart was created by Josh to assist in ensuring our team progressed on time to meet the deadline. The chart displays periods of concurrent work as we progressed through the requirements.

**Start Date: 11/06/2020  
Estimated Finish Date: 17/06/2020  
Actual Finish Date: 17/06/2020**

****

# Software Testing Report:

**The way we conduct the testing of our software will be detailed specifically in Testing Plan Documents for each sprint. The following pages will include a copy of the testing plan Sam has prepared for the first sprint.**

**Our testing and quality assurance practices will closely follow those specified by CITE Managed Services. Aashiyan has prepared our testing plan document to ensure quality control and quality assurance practices of CITE.**

**The criteria detailed in this testing report will be applied to all subsequent testing plans for each sprint.**

* **On the most fundamental level, all existing components of the application must be functional, which in the case of this project, are:  
  - the basic searching operations using the movie database are available and working  
  - GUI and elements are fully legible**
* **Any bugs and issues must be identified and reported so that they can be fixed during the following sprint.**
* **ACME ltd. Development Requirements are satisfied:  
  - The login page has a password with minimum level complexity.  
  - Analytics are undertaken to display data of the website’s usage timeline.**

**- New rating system with input from users.**

* **Client is satisfied with the delivered test results and application and signs off on it.**

# Optimisation Report

## Introduction

Most users are used to sites that are highly optimised for performance. When this level of optimisation is not met, users tend to leave the website without viewing the rest of it. For this reason, it is very important for our website to have its performance optimised so that it provides a good user experience.

Optimisation report contains all the data about your experience, including its performance and how your variants performed against your objective. The report includes a summary at the top with key information about its performance and actionable data in a series of charts and tables.

One of the methods to ensure the quality of your code is to have the best code optimisation process and techniques. Your code may be optimized so that it utilises memory, executes more rapidly, and performs fewer input and output operations.

The first step is to identify the impediments in your application’s performance to find the root cause of the problem. By identifying the problem, you can plan and select the best possible solution and option for your application. You can then implement the solution, and afterward you can measure the results. You can use a performance tool such as prefix or retrace.

## Optimising our Website:

### Database:

We can optimise our MySQL database by using phpMyAdmin with a few simple steps. It simply has a built in optimise options for our table. To perform the optimisation, simply open the database you would like to optimise, select all the tables, and choose optimise table option and it will execute the optimisation.

### Mail:

In our project, the time it takes to send the email and the time it appears in the inbox for our users could not be more optimised without investing in premium options. There is nothing much to do when we send that email to the user. After hitting send on the alerts, it simply depends on the SMTP servers, and various other filters, to deliver it to the users as soon as they can.

We are using the free Gmail server for our SMTP server. Being a free service used by many amateurs and professionals, it is not the most efficient option available. To improve mail performance, it is recommended to switch to a premium industry option like <https://sendgrid.com/> or another provider, which further provides an API for further performance optimisation. These options are also able to manage high mail traffic.

### Site Traffic:

There are several options for managing high traffic to the website. We first must establish what our expectations are regarding how high our traffic will be. If it becomes more popular, we will have to consider upgrading to a host that can accommodate our needs better, such as using a Virtual Private Server or dedicated hosting plan. A CDN (Content Delivery Network) is also a good option. It enables clients that are geographically distant from the host to request content from these CDNs, which are placed strategically at points where most traffic comes from, to reduce the strain of requests to the host machine.

### Code Optimisation:

Using performance tools like a profiler allow the developer to see what lines of source code are using disproportionate amount of resources. Once these are identified, they can be refactored and tested against performance milestones to make the site load faster.

Regular code reviews are also important since the profiler may not always find all problems. Rewriting the code and redesigning the system can have significant performance boosts if done properly.

A website can easily be slowed down by overuse of modules and features. Deciding what is essential to the site and what features to keep, then removing the unimportant can result in performance gains. This can be achieved by switching to using lightweight components where possible, avoiding redirects, optimising image loading, and having media load incrementally on demand.

<https://www.keycdn.com/blog/high-traffic>

# **Software Testing Plan: Sprint One:**

## Scope

### In Scope

* Login page has password protection with a minimum level of complexity.
* Login for different user groups.
* Top ten ratings chart.
* Option for users to leave a rating on a movie.
* New database analytics for movie search count.
* Web page analytics.
* 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.

### 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

# 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 (Singh, 2019). 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 this sprint, we will be testing the new functionality and security of our application and how optimised the code performs. 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 – New functions perform without error and completing their criteria.
* Acceptance Testing – Tests are validated against marking criteria.
* All documentation is completed and verified, then put into the Master Document.

## Test Deliverables

|  |
| --- |
| * Test Report (For whole project) * 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™ i7-7700k @ 3.60 GHz  
RAM: 16.00 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

# Test Results:

Screenshots are provided in the directory “TestEvidence”. For each entry in the table, an id number will prefix its corresponding screenshot. Screenshots have been provided outside this document to make it neater and more organised.

## System Test Components:

### 1. Movie Ratings

|  |  |  |  |
| --- | --- | --- | --- |
| Subject | Expected | Actual | Reference |
| * 1. Rating movie with no ratings –   200 cigarettes | Changes the rating to 5 | Meets expectation. | TestEvidence/1.1 |
| * 1. Rating a movie with existing ratings –   Terminator | Adds it to the previous ratings numbers and averages out the movie rating. | Meets expectation. | TestEvidence/1.2 |
| 1.3 clicking search without entering a movie title | Prompt user to enter a movie | Meets expectation. | TestEvidence/1.3 |
| 1.4 entering random data as the movie title | It refreshes with no result | Meets expectation. | TestEvidence/1.4 |

### 2. Analytics:

|  |  |  |  |
| --- | --- | --- | --- |
| Subject | Expected | Actual | Reference |
| 2.1 Analytics selected from the drop-down menu | Graph shows the current top rated 10 movies | Meets expectation. | TestEvidence/2.1 |
| 2.2 New movie rated – 8 seconds | Updates the graph. | Meets expectation. | TestEvidence/2.2 |
| 2.3 existing movie rated – 200 cigarettes | Updates the graph. | Meets expectation. | TestEvidence/2.3 |

### 3. Staff Log-in

|  |  |  |  |
| --- | --- | --- | --- |
| Subject | Expected | Actual | Reference |
| 3.1 Log-in, correct username and password | Access granted to staff page. | As expected. | 3.1 |
| 3.2 Log-in, empty username field, correct password. | Prompt user to enter username. | As expected. | 3.2 |
| 3.3 Log-in, correct username, empty password field | Prompt user to enter password. | As expected. | 3.3 |
| 3.4 Log-in, incorrect username, correct password. | Warn user password/username combination are incorrect | As expected. | 3.4 |
| 3.5 Log-in, correct username, incorrect password | Warn user password/username combination are incorrect | As expected. | 3.5 |
| 3.6 Log-in, incorrect username, incorrect password | Warn user password/username combination are incorrect | As expected. | 3.6 |

# Acceptance Test:

## Marking Guidelines Verification:

* Application functions correctly on multiple platforms:  
    
  This has been demonstrated in the test documentation, “Test Results”. Screenshots which serve as reference and proof of functionality are supplied in the directory “TestEvidence”. The screenshots have been labelled and with a descriptive name and are prefixed with a corresponding number identified in the test table.
* All documentation is properly formatted:  
    
  The master documented has been formatted so that each heading is on a new page. Headers, footers, table of contents and a cover page are included to assist with the organisation of the document. The master document feature was not used because of its tendency to corrupt and make the file unusable.
* Ensure all topics are adequately covered using common terminology:  
    
  The documents “Multi-Platform Report” and “Business Analysis Report” have been included in the master document. The former covers responsive versus adaptive web design and makes a choice as to which design, we have utilised. The latter covers software development, quality control and quality assurance practices relating to CITE Managed Services. These practices are integrated into our testing and quality assurance plan.
* All testing has been documented:  
    
  A test table has been constructed for the various platforms with various test cases. Each test case was carried out, an expectation was set for the outcome, and then verified against a screenshot provided in the TestEvidence directory which shows the actual outcome. This document is included in the master document.

# Final Handover

## Project Management Plan

All three sprints are now complete, and the project can move onto the final stage: the compiling of documentation and the addition of some finishing touches. The last additions are Web Content Accessibility and a final round of testing.

A close up of a map

Description automatically generated

## Web Content Accessibility Guidelines

For the final phase of the applications development, the team is adding to the web pages according to the Web Content Accessibility Guidelines (WCAG). These guidelines were created to provide shared web content accessibility that meets the needs of individuals, organizations, and governments internationally (Henry, 2018). The WCAG documents outline how to make web content more accessible to people with disabilities.

The requirements our team will implement for WCAG are:

* Alt tags for images.
* Tags have titles
* Functional at 200% zoom.
* Navigate using keyboard tabs.
* Labels for explanation.

# References

Graham, G. (2015, November 17). *The Difference Between Responsive and Adaptive Design*. Retrieved from css-tricks.com: https://css-tricks.com/the-difference-between-responsive-and-adaptive-design/

Merlin, G. (2018, November 18). *Responsive Vs Adaptive Website Designing in 2019*. Retrieved from graphicsmerlin.com: https://www.graphicsmerlin.com/responsive-vs-adaptive-web-design-2019/

Singh, A. (2019, December 06). *What Is Rapid Application Development (RAD)*. Retrieved from Capterra.com: https://blog.capterra.com/what-is-rapid-application-development/

# Glossary:

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