|  |
| --- |
| Team A |
| Master Document |
| Rapid Application Development Project |

|  |
| --- |
| Aashiyan Singh, Samuel Lee, and Joshua Macaulay  17/06/2020 |

Table of Contents

[**Sprint Three** 1](#_Toc43320699)

[Introduction: 1](#_Toc43320700)

[Meeting Agenda – Team A 2](#_Toc43320701)

[Sprint Three 2](#_Toc43320702)

[Event Heading 2](#_Toc43320703)

[Additional Information: 2](#_Toc43320704)

[Source Control: 3](#_Toc43320705)

[Project Management Plan: 4](#_Toc43320706)

[Software Testing Report: 5](#_Toc43320707)

[Optimisation Report 6](#_Toc43320708)

[Introduction 6](#_Toc43320709)

[Optimising our Website: 6](#_Toc43320710)

[Database: 6](#_Toc43320711)

[Mail: 6](#_Toc43320712)

[Site Traffic: 6](#_Toc43320713)

[Code Optimisation: 7](#_Toc43320714)

[Software Testing Plan: Sprint One: 8](#_Toc43320715)

[Introduction 8](#_Toc43320716)

[Scope 8](#_Toc43320717)

[In Scope 8](#_Toc43320718)

[Out of Scope 8](#_Toc43320719)

[Quality Objective 8](#_Toc43320720)

[Test Methodology 9](#_Toc43320721)

[Overview 9](#_Toc43320722)

[Test Levels 9](#_Toc43320723)

[Test Completeness 9](#_Toc43320724)

[Test Deliverables 10](#_Toc43320725)

[Resource & Environment Needs 11](#_Toc43320726)

[Test Environment and Tools 11](#_Toc43320727)

[Glossary: 12](#_Toc43320728)

[Test Results: 13](#_Toc43320729)

[System Test Components: 13](#_Toc43320730)

[1. Movie Ratings 13](#_Toc43320731)

[2. Analytics: 13](#_Toc43320732)

[3. Staff Log-in 14](#_Toc43320733)

[Acceptance Test: 15](#_Toc43320734)

[Marking Guidelines Verification: 15](#_Toc43320735)

[References 16](#_Toc43320736)

# **Sprint Three**

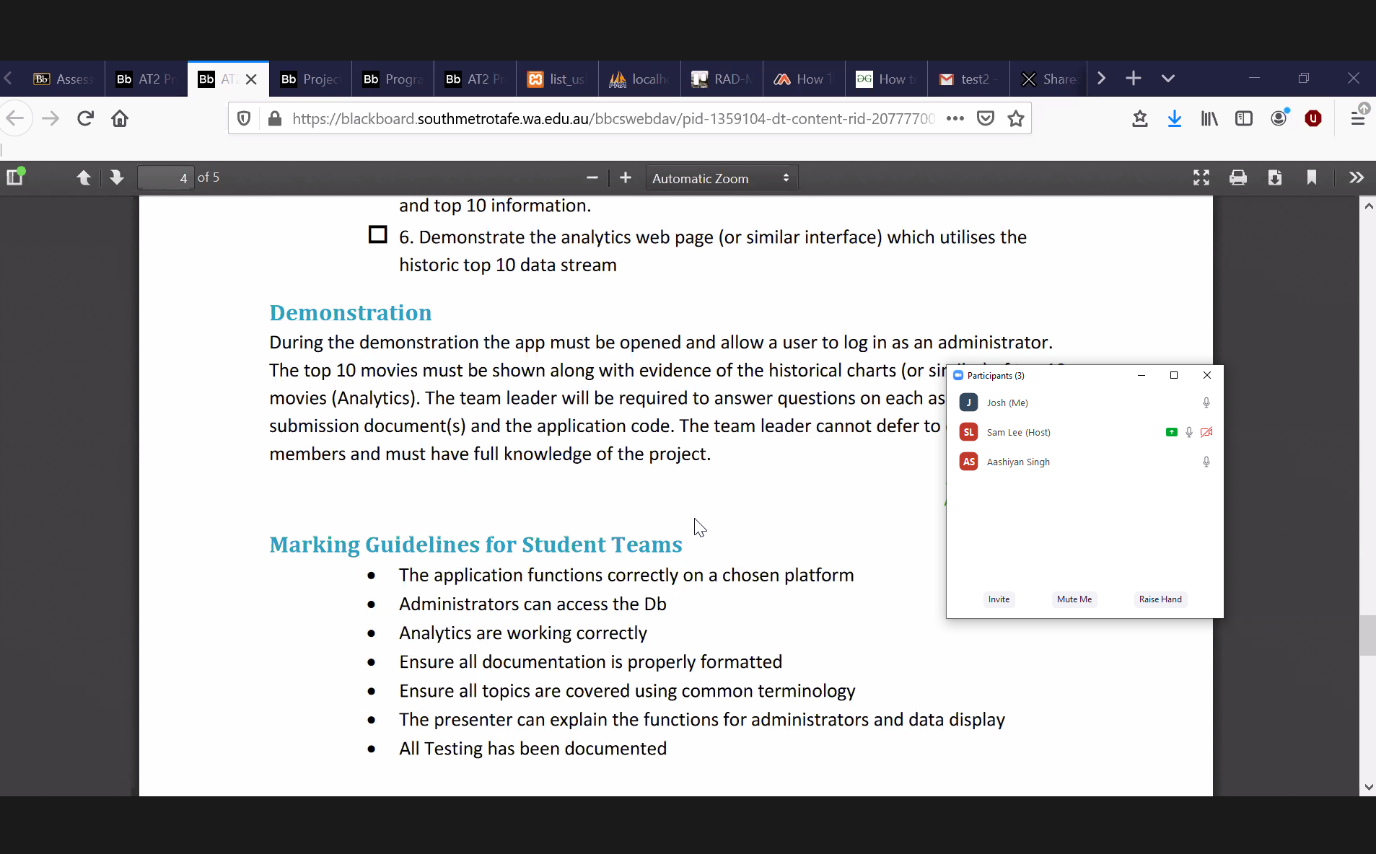
**Version 1.3:  
Last Edited 17/06/2020**

## 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:



## Meeting Agenda – Team A

### Sprint Three

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.

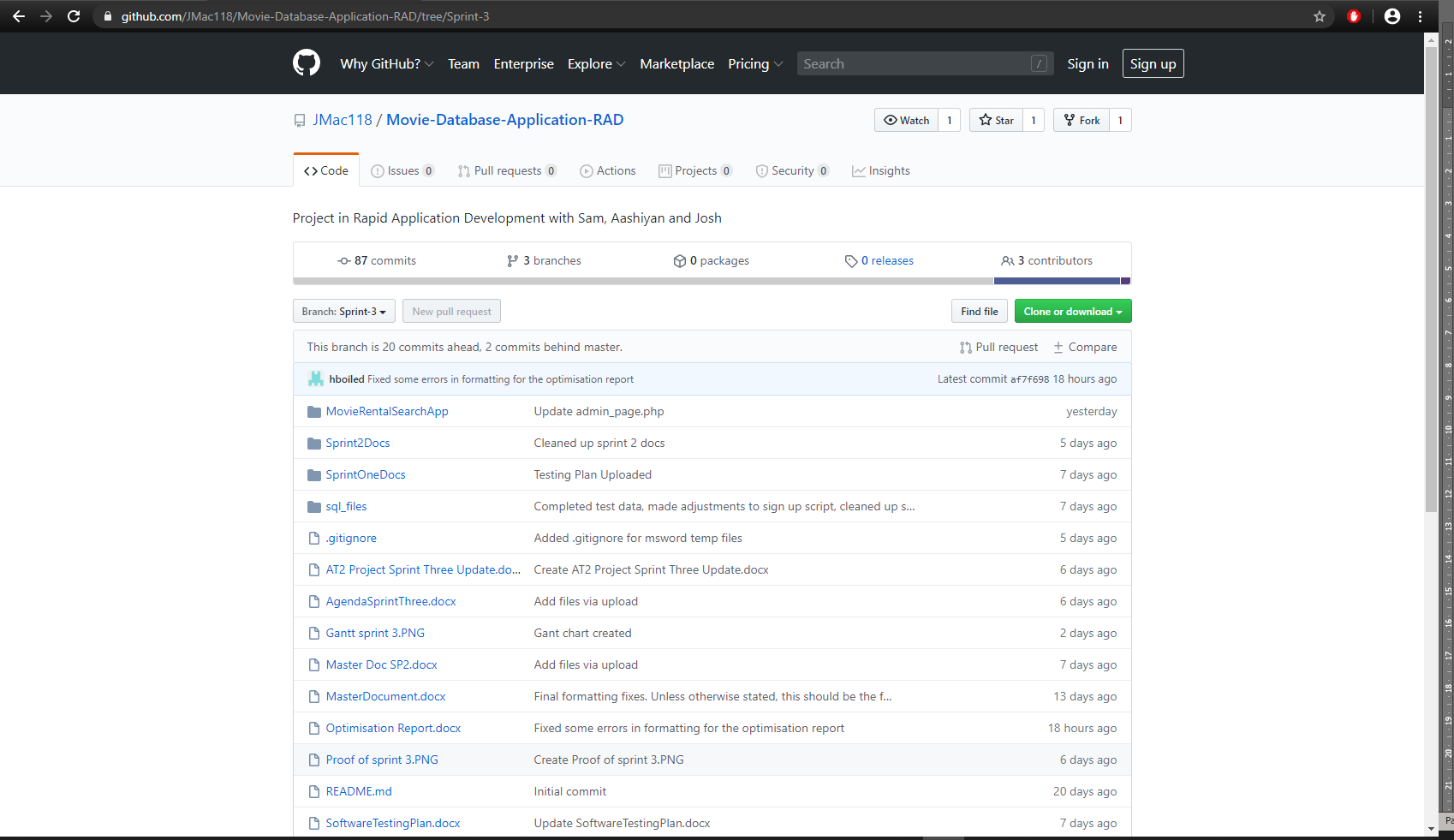
15:05 – Meeting evidence was taken, and the meeting ended.

## 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:



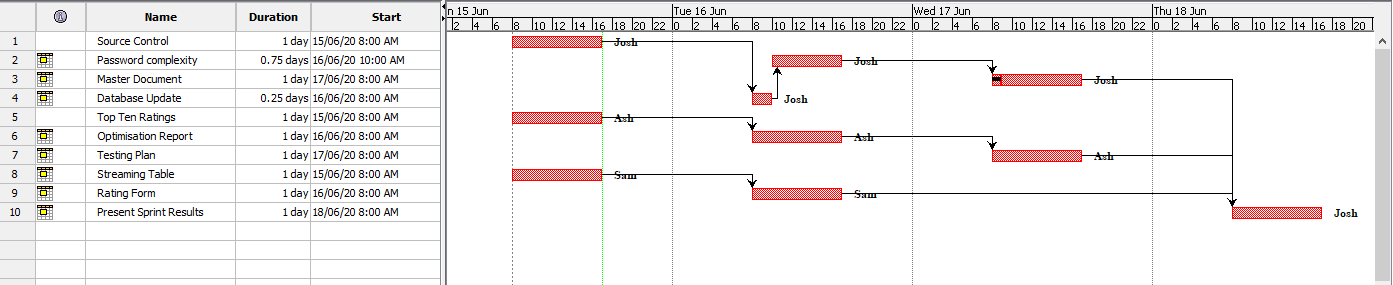
It can be found here:

<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:**

## Introduction

## 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

## Glossary:

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

## 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.

# References

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

W3Schools. (2020, 06 04). *CSS Tutorial*. Retrieved from w3schools.com: https://www.w3schools.com/css