Rapid Application Development

Project

AT2

|  |  |
| --- | --- |
| Team B |  |
| Zara Duncanson | P229768 |
| David Perry | 30010019 |
| Alan Pedersen | P225139 |

Date: 28/06/2020

Version Control

|  |  |  |
| --- | --- | --- |
| Version Number | Description | Date |
| 0.1 | Analysis and setup | 27/05/2020 |
| 1.0 | Update for sprint 1 | 03/06/2020 |
| 2.0 | Update for sprint 2 | 10/06/2020 |
| 3.0 | Update for sprint 3 | 17/06/2020 |
| 4.0 | Update for handover | 07/07/2020 |
|  |  |  |

Table of Contents

[Table of Figures iv](#_Toc44343992)

[Table of Tables v](#_Toc44343993)

[Analysis and Setup 1](#_Toc44343994)

[Meeting Agenda and Minutes 1](#_Toc44343995)

[Source Control 5](#_Toc44343996)

[Project Management 5](#_Toc44343997)

[Software Development Test Plan 6](#_Toc44343998)

[Introduction 6](#_Toc44343999)

[Test Methodology 6](#_Toc44344000)

[Test Deliverables 7](#_Toc44344001)

[Resource and Environment Needs 7](#_Toc44344002)

[Sprint Two Update 8](#_Toc44344003)

[Analysis Documentation 9](#_Toc44344004)

[CITE Business Rules for Software Development 9](#_Toc44344005)

[CITE Coding Standards 10](#_Toc44344006)

[CITE Quality Assurance 10](#_Toc44344007)

[ACME Entertainment Pty Ltd Requirements 11](#_Toc44344008)

[Sprint 1 12](#_Toc44344009)

[Meeting Agenda and Minutes 12](#_Toc44344010)

[Sprint 1 project management plan 14](#_Toc44344011)

[Multi-Platform Report 15](#_Toc44344012)

[Adaptive 15](#_Toc44344013)

[Responsive 15](#_Toc44344014)

[Analysis 16](#_Toc44344015)

[Choice 16](#_Toc44344016)

[Development and Testing 17](#_Toc44344017)

[Code Review 17](#_Toc44344018)

[Functional Testing 17](#_Toc44344019)

[Adaptive Web Design 23](#_Toc44344020)

[Sprint 2 28](#_Toc44344021)

[Meeting Agenda and Minutes 28](#_Toc44344022)

[Sprint 2 Project Plan 31](#_Toc44344023)

[Software Review Plan 32](#_Toc44344024)

[Introduction 32](#_Toc44344025)

[Plan Purpose 32](#_Toc44344026)

[Review Schedule 32](#_Toc44344027)

[Progress Review 32](#_Toc44344028)

[Final Review 32](#_Toc44344029)

[Performance Report 33](#_Toc44344030)

[Performance Tools 33](#_Toc44344031)

[Code Optimizers 34](#_Toc44344032)

[Source Control Snapshot 35](#_Toc44344033)

[Sprint 3 36](#_Toc44344034)

[Meeting Agenda and Minutes 36](#_Toc44344035)

[Source Control Snapshot 40](#_Toc44344036)

[Project Plan 41](#_Toc44344037)

[Administration login and user groups 42](#_Toc44344038)

[Optimization Report 44](#_Toc44344039)

[SearchMovies.php 44](#_Toc44344040)

[SignUp.php 46](#_Toc44344041)

[Top10.php 46](#_Toc44344042)

[UnsubscribeUsers.php 48](#_Toc44344043)

[Movie rating system design and testing 49](#_Toc44344044)

[Sprint 3 Test Plan 52](#_Toc44344045)

[Sprint 3 Test Table 53](#_Toc44344046)

[Handover 57](#_Toc44344047)

[Source control snapshot 57](#_Toc44344048)

[Handover Project Plan 58](#_Toc44344049)

[Software Review Report 59](#_Toc44344050)

[Quality Assurance Development and Methodology 59](#_Toc44344051)

[Testing Methodology 59](#_Toc44344052)

[Future Support 60](#_Toc44344053)

[User Requirements Mapping 60](#_Toc44344054)

[Handover Test Plan 61](#_Toc44344055)

[Introduction 61](#_Toc44344056)

[Scope 61](#_Toc44344057)

[Quality Objective 61](#_Toc44344058)

[Test Methodology 61](#_Toc44344059)

[Review Results 62](#_Toc44344060)

[Look and Feel 62](#_Toc44344061)

[Code Duplication 62](#_Toc44344062)

[WCAG Review 62](#_Toc44344063)

[List of Page Sources 65](#_Toc44344064)

[References 68](#_Toc44344065)

# Table of Figures

[Figure 1: GitHub repository snapshot 5](#_Toc44344072)

[Figure 2: Example of validation for Sprint 2. 8](#_Toc44344073)

[Figure 3: sprint 1 Gantt chart 14](#_Toc44344074)

[Figure 4: sprint 1 project plan 14](#_Toc44344075)

[Figure 5: The difference between Adaptive and Responsive web design (Merlin, 2018). 15](#_Toc44344076)

[Figure 6: the search page 18](#_Toc44344077)

[Figure 7: searching by title 18](#_Toc44344078)

[Figure 8: searching by title and year from 19](#_Toc44344079)

[Figure 9: searching by title and year to 19](#_Toc44344080)

[Figure 10: searching by title, year from and year to 20](#_Toc44344081)

[Figure 11: searching by Title and date ranges 20](#_Toc44344082)

[Figure 12: including Genre values in the search 21](#_Toc44344083)

[Figure 13: searching using all constraints 21](#_Toc44344084)

[Figure 14: The Top 10 page 22](#_Toc44344085)

[Figure 15: search screen arrangement 23](#_Toc44344086)

[Figure 16: 2 x 2 pattern search screen 24](#_Toc44344087)

[Figure 17: search screen arranged in a column 25](#_Toc44344088)

[Figure 18: top 10 chart small version 26](#_Toc44344089)

[Figure 19: top 10 chart medium version 26](#_Toc44344090)

[Figure 20: top 10 chart large version 27](#_Toc44344091)

[Figure 21: Sprint 2 Gantt chart. 31](#_Toc44344092)

[Figure 22: Sprint 2 project Plan 31](#_Toc44344093)

[Figure 23: Using Google's inspect element to display a waterfall performance view 33](#_Toc44344094)

[Figure 24: Using Google's inspect element lets you take a closer look at duration of each function 33](#_Toc44344095)

[Figure 25: Branch view of Source control. 35](#_Toc44344096)

[Figure 26: Source control Sprint Two. 35](#_Toc44344097)

[Figure 27: sprint 3 source control snapshot 40](#_Toc44344098)

[Figure 28: sprint 3 project plan 41](#_Toc44344099)

[Figure 29: Admin user has access to all functions related to the movie database 42](#_Toc44344100)

[Figure 30: The user group "Acme Staff" contains users, restricts the view of tabs not required by each user 42](#_Toc44344101)

[Figure 31: View of the limited tabs available to a staff user 43](#_Toc44344102)

[Figure 32: Staff users have limited controls of queries such as deleting data from the tables 43](#_Toc44344103)

[Figure 33: Loading the Search Movies page 44](#_Toc44344104)

[Figure 34: Loading all movies from database 45](#_Toc44344105)

[Figure 35: Searching for a combination of rating and genre 45](#_Toc44344106)

[Figure 36: Loading the Search Movies page in mobile format 45](#_Toc44344107)

[Figure 37: Loading the Sign-Up page 46](#_Toc44344108)

[Figure 38: Using the form to sign up a user 46](#_Toc44344109)

[Figure 39: Loading the Top Ten page 47](#_Toc44344110)

[Figure 40: Loading the Unsubscribe Users page 48](#_Toc44344111)

[Figure 41: Unsubscribing a user from mail list 48](#_Toc44344112)

[Figure 42: rating table SQL 49](#_Toc44344113)

[Figure 43: SQL to initialise the ratings table 49](#_Toc44344114)

[Figure 44: rating update view SQL 50](#_Toc44344115)

[Figure 45: example rating query 50](#_Toc44344116)

[Figure 46: ratings applied to table 50](#_Toc44344117)

[Figure 47: setting the ratings to random values 50](#_Toc44344118)

[Figure 48: the ratings table with random values 50](#_Toc44344119)

[Figure 49: creating the top 10 rated movie view 50](#_Toc44344120)

[Figure 50: top 10 rated movie view 51](#_Toc44344121)

[Figure 51: When clicking on a movie from the table, the rating box appears 53](#_Toc44344122)

[Figure 52: When hovering over the stars, they illuminate to gold to give the user better experience 54](#_Toc44344123)

[Figure 53: Snapshot of the top 5 rated movies before a rating is submitted 54](#_Toc44344124)

[Figure 54: After a 5-star rating is added to the top movie, the count and average rating increase 54](#_Toc44344125)

[Figure 55: Page returns to search result after close button pressed in rating box 55](#_Toc44344126)

[Figure 56: The admin username and password are hard coded into the connection script 55](#_Toc44344127)

[Figure 57: The chart on the right has updated the 10th movie in the list when the data has changed without the browser being refreshed 55](#_Toc44344128)

[Figure 58: After running some searches on 12 Monkeys, the chart on the right has updated to display the new data without refreshing the browser 56](#_Toc44344129)

[Figure 59: handover source control snapshot 57](#_Toc44344130)

[Figure 60: Gantt chart showing sprint 3 and handover phases 58](#_Toc44344131)

[Figure 61: task list showing sprint 3 and handover phases 58](#_Toc44344132)

[Figure 62: A Use Case diagram represents how different users interact with the Movie database System. 60](#_Toc44344133)

[Figure 63: input element example 62](#_Toc44344134)

[Figure 64: example of a chart presentation element 63](#_Toc44344135)

[Figure 65: example chart with activation element 63](#_Toc44344136)

[Figure 66: text equivalent data for chart 64](#_Toc44344137)

# Table of Tables

[Table 1: sprint 2 testing documents 8](#_Toc44248251)

[Table 2: ACME search requirements 11](#_Toc44248252)

[Table 3: adaptive vs responsive pros & cons 16](#_Toc44248253)

[Table 4: application file list 17](#_Toc44248254)

[Table 5: top 10 chart sizes 25](#_Toc44248255)

[Table 6: movie rating table definition 49](#_Toc44248256)

[Table 7: sprint 3 test plan 52](#_Toc44248257)

[Table 8: sprint 3 test table 53](#_Toc44248258)

[Table 9: top level and supporting files 64](#_Toc44248259)

# Analysis and Setup

## Meeting Agenda and Minutes

AGENDA

RAD Setup Meeting

25/02/2020

9:00 – 10:00

Meeting called by Alan Pedersen

|  |  |
| --- | --- |
| Attendees: | Zara Duncanson  David Perry  Alan Pedersen |
| Please read: |  |
| Please bring: |  |

|  |  |  |
| --- | --- | --- |
| 9:00 – 10:00 | Introduction  Select source control  Create repository  Adaptive vs Responsive  Report  Select project management software  Project plan  Design application  Break points  Layouts  Code application  Testing plan  Presentation  Demo | Thornlie TAFE Library |

#### Additional Instruction:

|  |  |
| --- | --- |
| Team Meeting | 25/02/2020  9:00  Thornlie TAFE Library |

|  |  |  |  |
| --- | --- | --- | --- |
| Meeting called by: | Alan Pedersen | Type of meeting: | Setup Meeting |
| Facilitator: |  | Note taker: | Alan Pedersen |
| Timekeeper: |  |  |  |

|  |  |
| --- | --- |
| Attendees: | Zara Duncanson (ZD)  David Perry (DP)  Alan Pedersen (AP) |
| Please read: |  |
| Please bring: |  |

Minutes

|  |  |  |  |
| --- | --- | --- | --- |
| Agenda item: | Selection of a source and version control system | Presenter: | All |

Discussion:

GitHub was unanimously selected for source and version control. It was decided that the owner of the database adopted for the project would set up a GitHub repository and inform the other team members of its location.

Conclusions:

GitHub to be used, repository to be set up.

| Action items | Person responsible | Deadline |
| --- | --- | --- |
| * Create GitHub repository | AP | 27/05/20 |
| * Email location to team members | AP | 27/05/20 |

|  |  |  |  |
| --- | --- | --- | --- |
| Agenda item: | Selection of base version for project | Presenter: | All |

Discussion:

The three versions of the movie database were reviewed. The combined project will be drawn from the three versions. The database and search screen from APs version, the top 10 chart page from ZDs version.

Conclusions:

The files are to be loaded into GitHub and merged.

| Action items | Person responsible | Deadline |
| --- | --- | --- |
| * Load files into GitHub | AP & ZD | 27/05/20 |
| * Merge the systems | AP | 28/05/20 |
| * Review merged system | All | 29/05/20 |

|  |  |  |  |
| --- | --- | --- | --- |
| Agenda item: | Project plan | Presenter: |  |

Discussion:

Project Libre was selected as the project management software package. DP to develop a project management plan for sprint 1, project plan to be loaded into GitHub.

Conclusions:

Project Libre to be used

| Action items | Person responsible | Deadline |
| --- | --- | --- |
| * Develop project plan | DP | 29/05/20 |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Agenda item: | Multi-Platform Report | Presenter: |  |

Discussion:

ZD was tasked with developing the Multi-Platform Report. Report to be loaded into GitHub for review.

Conclusions:

| Action items | Person responsible | Deadline |
| --- | --- | --- |
| * Develop report | ZD | 02/06/20 |
|  |  |  |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Agenda item: | CITE QA standards | Presenter: |  |

Discussion:

It was noted that the CITE web site is no longer active at [www.cite.com.au](http://www.cite.com.au). Query to Stewart on sourcing standards and clarification of what is required for the Analysis Report.

Conclusions:

| Action items | Person responsible | Deadline |
| --- | --- | --- |
| * Source standards | AP | 28/05/20 |
| * Clarify report requirements | AP | 28/05/20 |
|  |  |  |

Other Information

Observers:

Resources:

Special notes:

AP to source laptop computer for future meetings

## Source Control

GitHub was selected to manage source and version control. A repository was created with the following address:

<https://github.com/AlanPedersen/RAD_project_movie_database>

A snapshot of the initial repository is presented below:

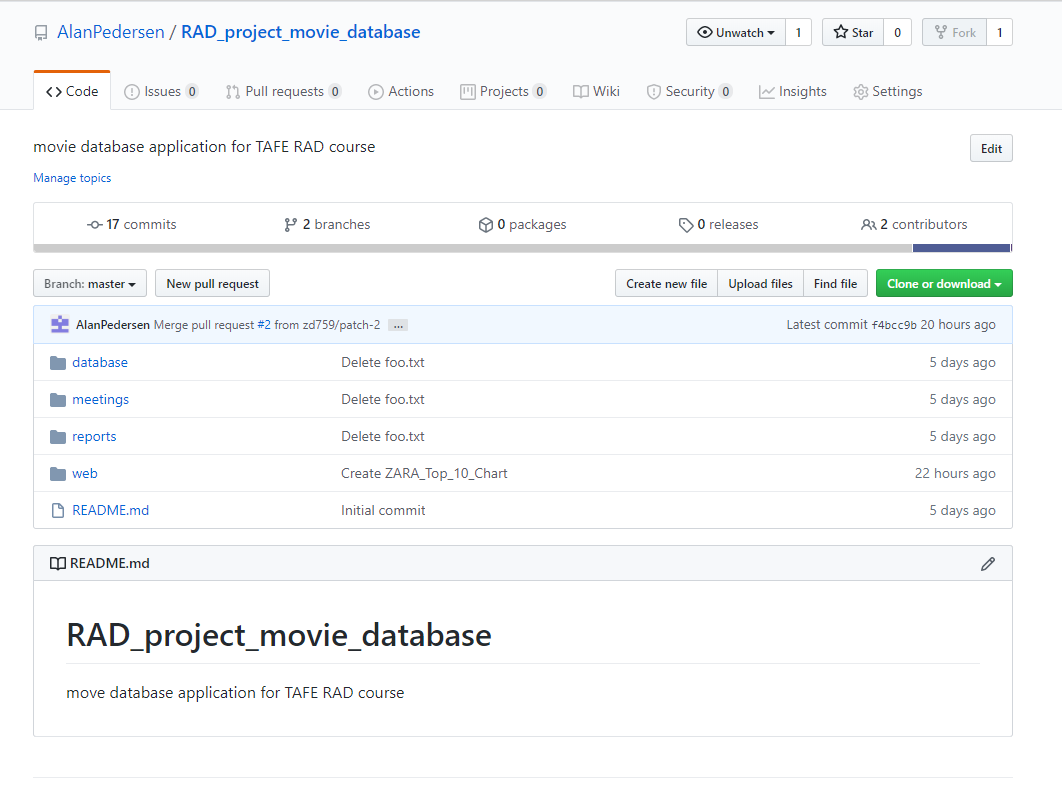


Figure 1: GitHub repository snapshot

## Project Management

Project Libre was adopted for project management.

## Software Development Test Plan

### Introduction

This Test Plan has been designed to ensure that the software meets the specified business requirements as well as the customers’ demands and expectations.

#### Scope

The scope of the testing undertaken will focus on ensuring that the project conforms to CITE published business rules and meets quality assurance standards.

At each stage of the project the delivered features will be checked against the client’s requirements to ensure that all requirements have been met.

All code developed will be checked to confirm that it complies with CITE published standards for code development.

The database of movie data will be accepted “as is” and will not be subject to any verification. While due consideration will be given to the efficiency of the application the specific performance on ACME systems will not be evaluated.

#### Quality Objective

The objective of the test plan is to ensure that the delivered application meets all the specified requirements and performs robustly.

There are two main objectives for the testing plan:

1. Compliance to specifications
   1. Compliance with CITE standards
   2. Compliance with client requirements
2. Correct functioning of the code base
   1. Does the code perform the required operation?
   2. Does the code run robustly?

#### Roles and Responsibilities

Because the application will be developed by a small team responsibility will be rotated across the members. The team member with the most relevant expertise will be allocated to each task. The other team members will be responsible for peer reviewing the product. It is expected that all team members will have significant input into the project.

### Test Methodology

#### Overview

The approach taken to development will be Rapid Application Development (RAD) using the Agile methodology. CITE have extensive experience with RAD and Agile having used them extensively over many years. CITEs approach has been demonstrated to develop high quality software faster, leading to improved business agility and a greater capacity to handle the pressures of competition.

#### Test Levels

Testing will be undertaken on two levels. There will be an ongoing process of verification to ensure that the product development complies both with CITE standards and the requirements of the customer.

* Code reviews will be undertaken on a regular basis to check on standards compliance.
* Comparisons will be made against the client’s requirements to ensure that all requirements are met.

The second level of testing will be the validation of each deliverable. This level focuses on testing that the application functions correctly.

* Does the code produce the correct output?
* Does the code function as required?

#### Bug Triage

When detected bugs and non-conforming code will be classified according to their impact on the project. Bugs that stop the execution of the code base or result in incorrect results will be accorded the highest priority. Other issues will be scheduled and addresses as appropriate.

#### Test Completeness

Testing will be considered complete when it can be demonstrated

* Testing has been completed for each functional requirement
* Testing has been completed for each non-functional requirement
* Identified issues have either been addressed of scheduled for action
* Testing documentation has been completed

### Test Deliverables

At the end of each RAD sprint the following reports will be created:

* completed testing tables
* code verification report

list of any outstanding issues.

### Resource and Environment Needs

During testing the database will be hosted using MySQL.

Apache will be used as the web server.

All reporting and development to be undertaken using standard Microsoft tools.

The PHP Code Sniffer tool will be used to validate all PHP scripts against the PEAR standard.

### Sprint Two Update

Following the same methodology of Sprint One’s Test plan, Sprint two testing will undergo the same amount of scrutiny. Testing will be performed on both new interfaces created;

* the Sign-Up page for new users to subscribe to free communications or unsubscribe if they are already subscribed
* the administrator portal with password access for an administrator user to unsubscribe users when requests are made by email
* the three associated PHP scripts which run these two interfaces

Testing will be completed on the two new interfaces separately and recorded in different documents as follows:

#### Please Refer To

|  |  |  |
| --- | --- | --- |
| **Document Name and Version** | **Description** | **Location** |
| *sprint02.user.subscribe.phpmail.design.testing.docx*  *Version 1* | *Testing and results for the Sign-Up page for new users to subscribe and the two associated scripts* | *Within the same folder this document is located.* |
| *sprint02.unsub.design.testing.docx* | *Testing and results for the administrator interface and the associated script* | *Within the same folder this document is located.* |

Table 1: sprint 2 testing documents

All of the new sprint’s web pages, interfaces and scripts have also been validated using PHP Code Sniffer against the PEAR standards.

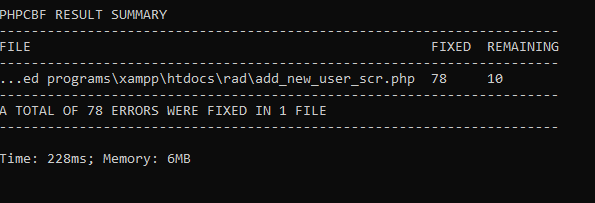


Figure 2: Example of validation for Sprint 2.

## Analysis Documentation

Since 2009 CITE has provided services to students and education institutes in Perth, Western Australia. Standing for Centre of Information Technology Excellence, CITE is not just a name, it is a promise for superior service to our clients.

A summary of our successful approach to managing software development is presented in the following sections. A more comprehensive discussion of our approach can be found on our web site www.citems.com.au.

### CITE Business Rules for Software Development

Over many years of operation CITE have developed an innovative and robust approach to software development. Our approach was developed to ensure we exceed our customers’ expectations for both quality and cost. CITE operate a program of continuous improvement, staying ahead of the pack by introducing leading industry practices to meet our customers current and future needs.

The key benefits delivered to our customers by our approach include:

#### Strong Project and Process Management

Our project managers act as a focal point for all your project-related needs. Project managers are involved in the entire project lifecycle to plan, organize, control and deploy key milestones, deliverables according to your goals. Plus all CITE Managed Services projects are supervised by our CTO, who is responsible for processes audit and enforcement, as well as for measurement of existing processes against established metrics and process improvements analysis and implementation.

#### In-Depth Requirements Analysis

Here at CITE Managed Services we pay special attention to the discovery phase to ensure that the final product fully meets your expectations. CITE Managed Services Business Analysts define easier ways to attain what you need, while Project Managers identify the most suitable development methodology and cooperation model, as well as assemble the most effective project team, and the tech experts work on technical feasibility and select the most efficient technologies.

#### Thorough Quality

We have an independent QA department responsible for providing quality assurance throughout the entire Software Development Lifecycle of every project. Depending on the size of the latter, we allocate a testing team and define a QA timeframe, which allows us to optimize involvement and project costs. Though independent by virtue of their duties, our QA experts work in close cooperation with our development team.

#### Knowledge to Build On

Since our inception in 1999, we have been building up a knowledge base by working on hundreds of development projects. We have absorbed the best approaches to achieve maximum performance and the highest quality when delivering projects, avoiding redundant complexity that affects both budget and timing.

#### DevOps and Continuous Delivery

We are embracing DevOps to ensure the synergy between development and operations by paying close attention to key pain-points in your DevOps needs and automating end-to-end delivery processes while ensuring scalability and security of your infrastructure.

#### Transparent Communication

Transparency enables both you and us to see at a glance, whether we are on track to meet the goals or are blocked on one or more directions, and eliminate these blockers. That is why right at the project launch we agree with you on a communication plan tailored to your needs to work on different levels – from core team to management stakeholders

### CITE Coding Standards

Coding standards are collections of coding rules, guidelines, and best practices. A coding standards document tells developers how they must write their code. Instead of each developer coding in their own preferred style, they will write all code to the standards outlined in the document. This makes sure that a large project is coded in a consistent style.

The coding standards will reflect the context of the Language and the Clients requirements. In the absence of these criteria CITE developers will default to the industry standard (C# – Microsoft, Java – Oracle)

All systems and projects will be covered by the following ISO Standard(s)

ISO/IEC/IEEE 12207:2017 Systems and software engineering — Software life cycle processes

### CITE Quality Assurance

CITE have implemented comprehensive quality management systems (QMS) across all their operations. Our focus is on delivering the highest quality product across the entire lifetime of the project.

CITE take QMS seriously with processes addressing:

#### Quality Planning

CITE Managed Services puts together quality plans that govern the applicable set of standards, regulations, procedures, guidelines, and tools during the development lifecycle in each project.

#### Quality Assurance

We have established processes that evaluate project performance and aim to assure that quality standards are being followed and that the deliverables comply with customer requirements.

#### Quality Control

We measure performance trends to identify defective pieces of code, verify that deliverables are of high quality and that they are complete and correct.

#### CITE boasts an independent QA department

CITE Managed Services QA department is an independent structural subdivision. Our QA team consists of skilled QA engineers who get involved in projects on a dedicated or an on-demand basis. The allocation of QA engineers depends on project size and complexity: they can be easily reallocated from one project to another upon necessity. Such flexibility allows us to optimize efforts and, thus, the overall project budget.

CITE Managed Services performs quality assurance throughout the entire software development lifecycle with QA team members being involved at all stages. A Lead QA specialist is assigned at the commencement of each project and is involved into initial business analysis and requirements specification. Such a simultaneous interaction of our development and QA teams provides for a better understanding of the project scope and the client’s business objectives.

### ACME Entertainment Pty Ltd Requirements

ACME Entertainment (ACME) require the development of a movie database application.

ACME have informed CITE that a prototype of the required product exists, and that the final application should be developed from this.

The prototype application is web based, offering the ability to search the database for matching records. The application also offers a chart detailing the 10 most frequently searched movies.

The prototype application allows the user to search using one of more of the criteria detailed in the following table, if multiple criteria are specified, they must all be met by the results:

|  |  |
| --- | --- |
| Criteria | Search Type |
| Title | The application should allow users to search by movie title. All movie titles that include the user search term should be returned. |
| Year | The user should be able to search by year or by a range of years. |
| Genre | The user should be able to search by one or more genre codes |
| Rating | The user should be able to search for movies matching the selected rating code or codes. |

Table 2: ACME search requirements

ACME have commissioned CITE update the application so that it can be used across all major digital platforms. They have become aware that may potential users access the internet only via mobile devices and require that their application support this type of access.

ACME have commissioned CITE to review contemporary approaches to multi-platform web design and recommend the best approach to adopt for their application.

Once an approach has been adopted the application should be reworked to create the multi-platform version.

# Sprint 1

## Meeting Agenda and Minutes

AGENDA

RAD Setup Meeting

03/06/2020

9:00 – 10:00

Meeting called by Alan Pedersen

|  |  |
| --- | --- |
| Attendees: | Zara Duncanson  David Perry  Alan Pedersen |
| Please read: |  |
| Please bring: |  |

|  |  |  |
| --- | --- | --- |
| 9:00 – 10:00 | Introduction  Demonstrate and review updated web site  Feedback on site | Thornlie TAFE Library |

Additional Instruction:

|  |  |
| --- | --- |
| Team Meeting | 03/06/2020  9:00  Thornlie TAFE Library |

|  |  |  |  |
| --- | --- | --- | --- |
| Meeting called by: | Alan Pedersen | Type of meeting: | Sprint 1 review Meeting |
| Facilitator: |  | Note taker: | Alan Pedersen |
| Timekeeper: |  |  |  |

|  |  |
| --- | --- |
| Attendees: | Zara Duncanson (ZD)  David Perry (DP)  Alan Pedersen (AP) |
| Please read: |  |
| Please bring: |  |

Minutes

|  |  |  |  |
| --- | --- | --- | --- |
| Agenda item: | Demonstrate and review web site | Presenter: | AP |

Discussion:

The adaptive web site design was demonstrated to the team.

The approach adopted was explained and the operation of the web site discussed.

Conclusions:

The web site design was accepted.

| Action items | Person responsible | Deadline |
| --- | --- | --- |
| * Book time for presentation | AP | 04/06/20 |
|  |  |  |

Other Information

Observers:

Resources:

Special notes:

## Sprint 1 project management plan

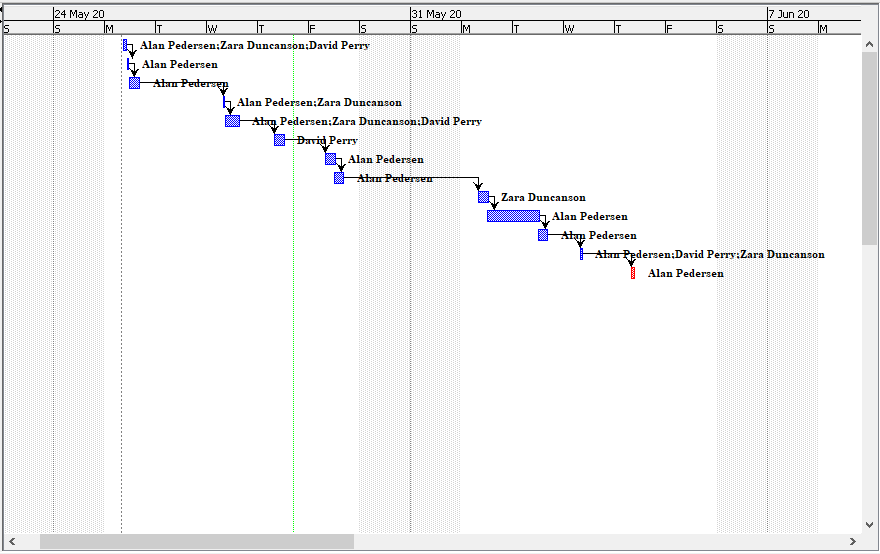


Figure 3: sprint 1 Gantt chart

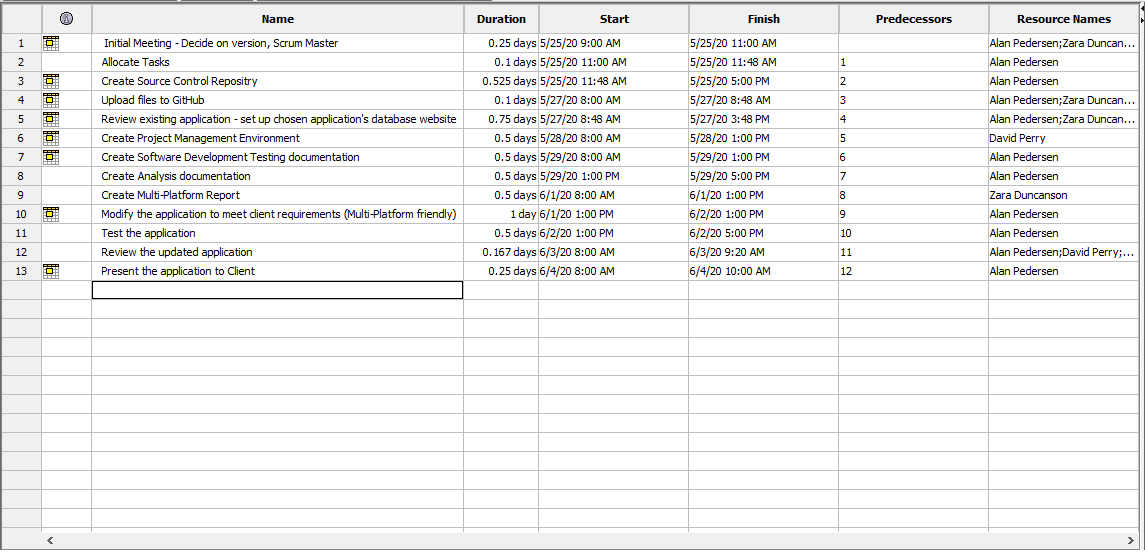


Figure 4: sprint 1 project plan

## Multi-Platform Report

Users hate websites which are hard to operate. If websites aren’t user friendly the user will leave, and resources spent on the creation of the website will be wasted. Today users are increasingly accessing web pages on smartphones, tablets and even televisions; with an estimated 79% of global internet use accessed by mobile devices in 2018 (Merlin, 2018). If a website isn’t easy to view on these devices, it will drive visitor traffic down. Therefore, it is vital to website and business success to respond to this by creating an adaptive or responsive website to maintain good traffic. Both web designs have the same goal; to build mobile-friendly websites and serve all screen sizes better.

### Adaptive

Adaptive web design demands the creation of several distinct layouts of a web page for different screen sizes, depending on what the user that accesses the site is using. There is usually a separate layout for mobile phones, tablets, and desktop computers waiting on standby until accessed, at which point the site detects the device size and responds with the matching layout. The most common layout sizes generally created are; 320, 480, 760, 960, 1200 and 1600. It may appear that more work is needed to create this response to mobile web design, yet responsive design can be more complex if not suitably used it can create display and performance issues.

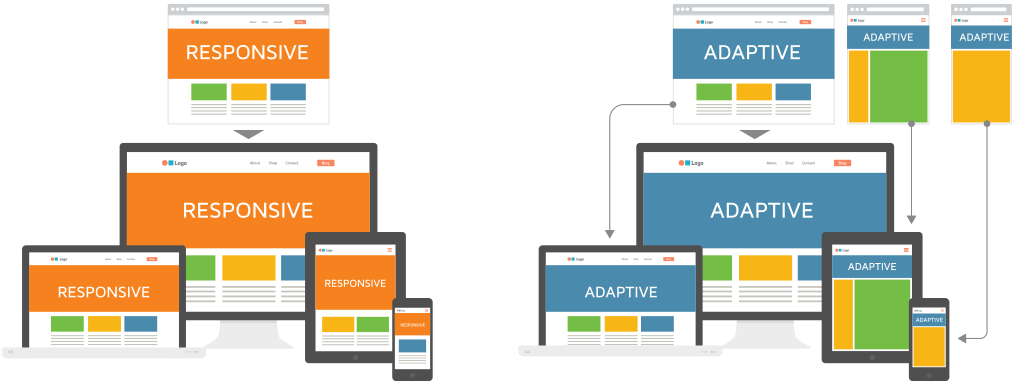


Figure 5: The difference between Adaptive and Responsive web design (Merlin, 2018).

### Responsive

Alternatively, responsive web design contains a blend of flexible grids, layouts, images, and an intelligent use of CSS media queries. It aims provides users with an optimal viewing experience, an easy to read navigation with a minimum of resizing, panning, and scrolling across a range on different devices. It is usually considered harder to adapt responsive design to websites than adaptive design, especially for beginners to web design, but this is becoming easier due to themes created by content management systems such as Wordpress. Responsive does not offer as much control as adaptative design but can be much more efficient to build and maintain for experienced designers and developers due to their fluidity.

### Analysis

The following table aims to outline and compare each practice;

|  |  |  |
| --- | --- | --- |
|  | **ADAPTIVE** | **RESPONSIVE** |
| **Summary** | Different layouts for each device or OS | Same layout for all screen sizes |
| Advantages | * Easy to build and implement * Allows for more control for design * Helps developers build the best possible user experience for their website as more controllable * Useful for retrofitting existing sites * Can have better loading speed if responsive has not been optimised | * More flexible as can work on any screen size * Fast, loads quickly as only needs to load single layout * Can be managed by a single designer, cutting down maintenance costs * More ‘search engine optimised’ than adaptive * Usually used when creating new websites as it will generally keep project budgets lower |
| **Disadvantages** | * Less flexible * May take longer to load the page as the page finds correct layout * If the project and websites become large, it may need more designers to handle the multiple layouts of each as it will become complex, increasing maintenance costs * As screens resize, the elements ‘snap’ leading to a less streamlined look | * Not as easy to build for beginners * Can be confusing to design with all layouts in mind * May require more lines of code (but may be comparable to Adaptive’s need for several different versions) * If not properly implemented the pages can suffer from site speed |

Table 3: adaptive vs responsive pros & cons

In summary, there aren’t any shortcuts when choosing an answer to web design for smaller devices. Both methods should be carefully implemented with the target audience and user experience in mind; it is not as simple as checking that all the elements fit within a page, but that visual hierarchy is maintained. Overall, it is usually best to use responsive for new projects and adaptive to implement with existing projects.

### Choice

For our project we have chosen to use both adaptive and responsive methods. This is because a responsive method worked better for the main search page, using 4 ‘blocks’ which display differently depending on the users screen size. The chart implemented was first switched to be a horizontal bar chart for better UX, and an adaptive method was implemented using an imbedded CSS with 3 elements; these are switched on and off depending on which is required by the particular user.

## Development and Testing

### Code Review

The following files are used by the application

|  |  |
| --- | --- |
| File Name | Description |
| bootstrap.min.css | Bootstrap css formatting code file |
| connect.pdo.php | Open a connection to the database |
| demo.css | Default css formatting |
| index.php | Default web application entry page |
| movie\_list\_genre.php | Read the genre codes from the database and create the selection list for the search form. |
| movie\_list\_rating.php | Read the rating codes from the database and create the selection list for the search form. |
| movie\_list\_scr.php | Create the SQL query from the form data to use when selecting movie records from the database. Run the query and format the results into a HTML table for display |
| movie\_top\_10\_google\_data\_scr.php | Get the list of top 10 searched for movies from the database. Format data so it is suitable for google charts |
| movie\_year\_limits.php | Get the minimum and maximum year values from the database |
| SearchMovies.php | Display the form to search for movies and display the results of the search. |
| Top10.php | Display a chart of the top 10 most searched for movies |

Table 4: application file list

Each file was reviewed manually and with the php code sniffer application.

### Functional Testing

The following screen dump shows the query page when first loaded, the Title field is preselected in the Sort By list:

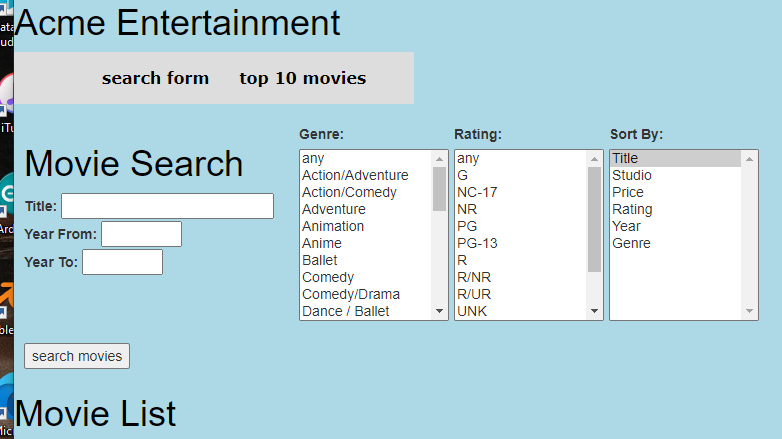


Figure 6: the search page

A search was conducted using the term “sing” in the title box, 18 records were returned all the titles containing “sing” within them:

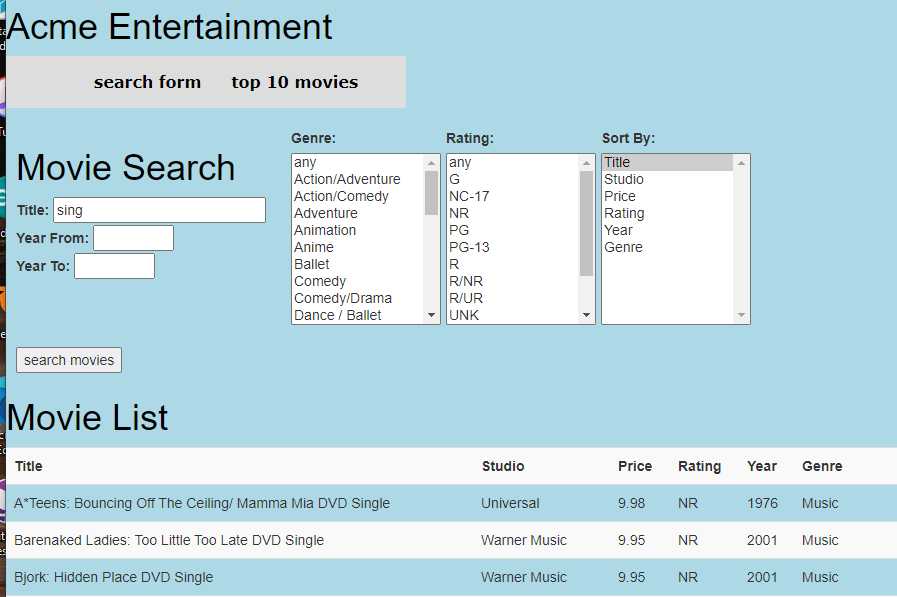


Figure 7: searching by title

A Year From value of 1970 was set and the results ordered by Year, only movies with a Year greater than 1970 are included in the report:

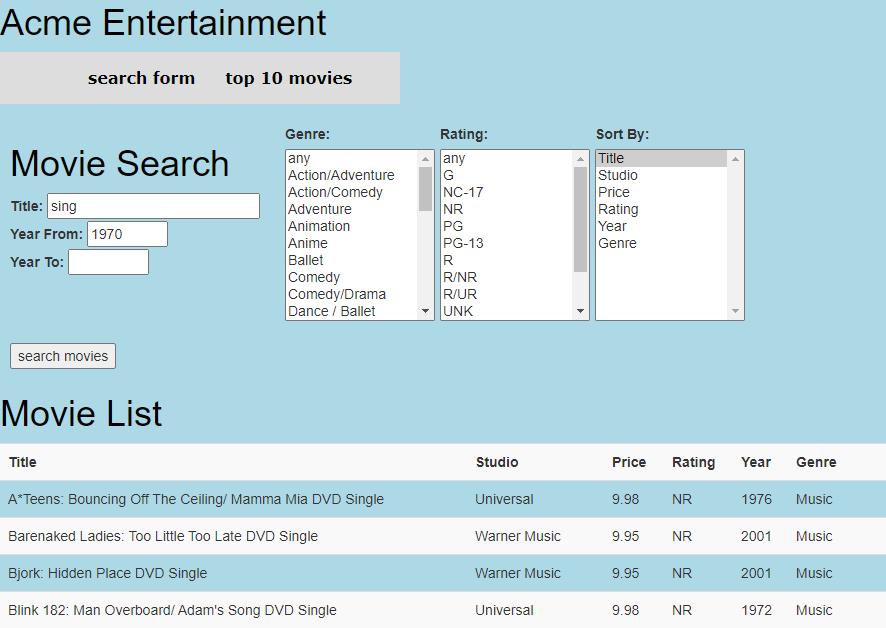


Figure 8: searching by title and year from

A Year To value of 1990 was set and the results ordered by Year, only movies with a Year less than 1990 are included in the report:

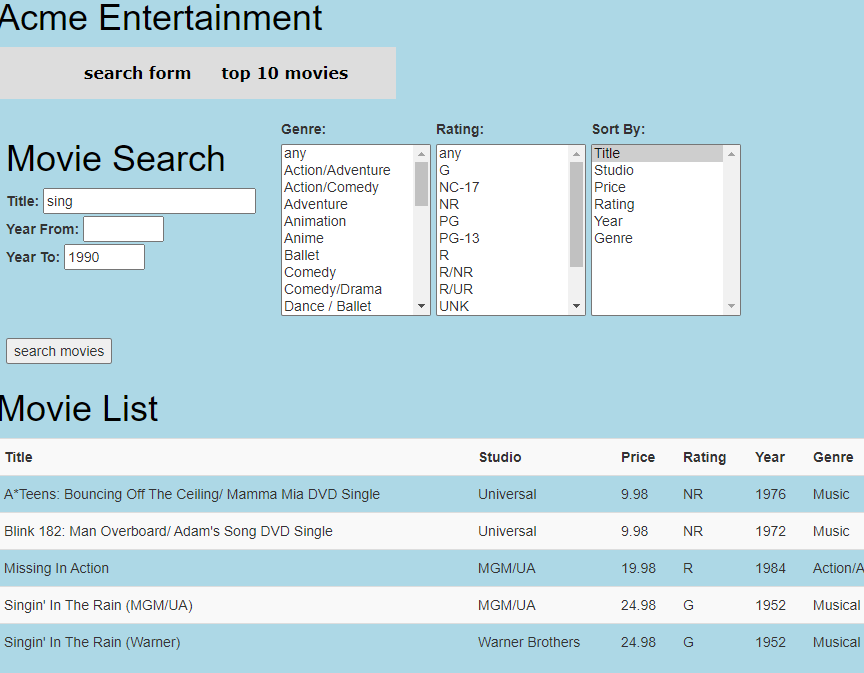


Figure 9: searching by title and year to

A Year From value of 1970 and a Year To value of 1990 was set and the results ordered by Year, only movies with a Year between 1970 and 1990 are included in the report:

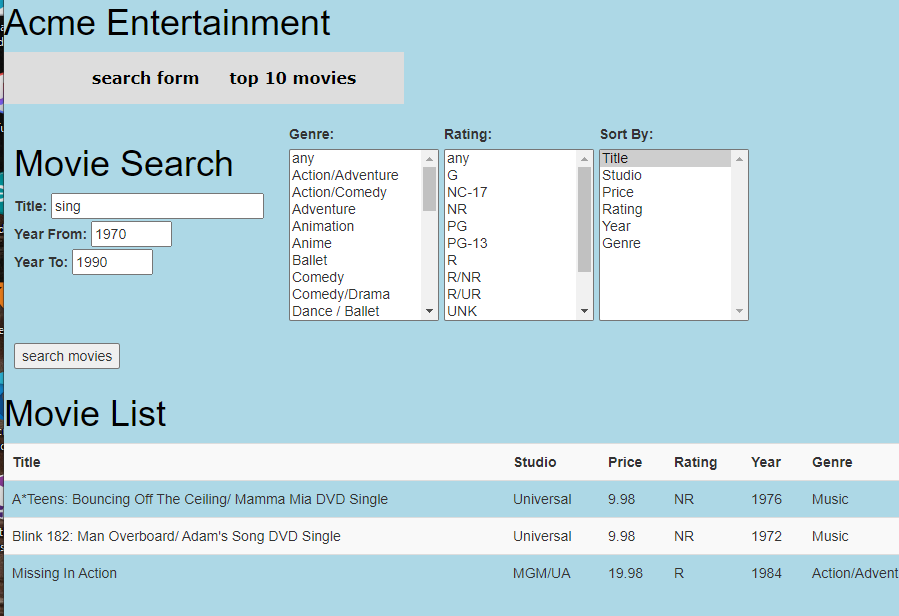


Figure 10: searching by title, year from and year to

The year values were swapped to find movies outside of the selected range:

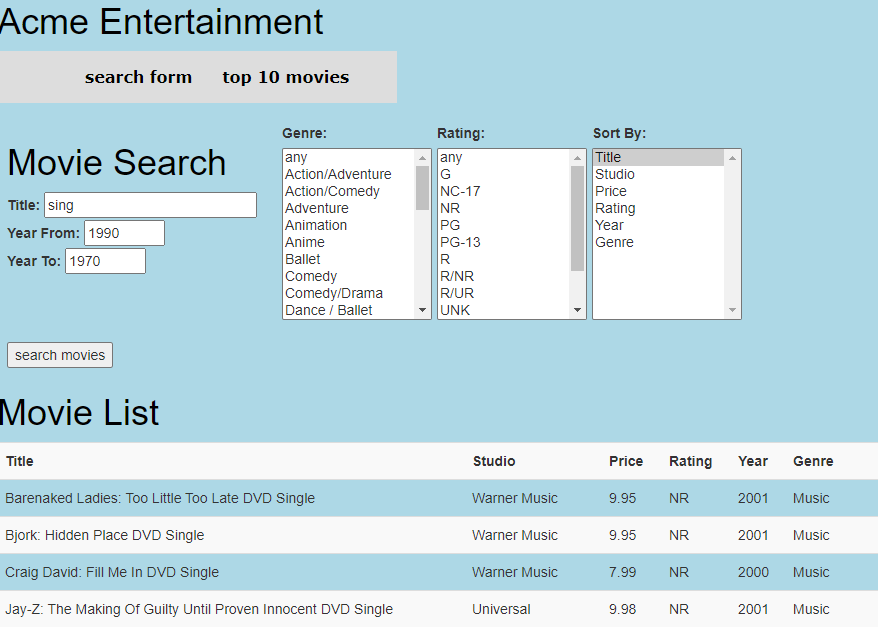


Figure 11: searching by Title and date ranges

The Music, Musical and Comedy Genres were added to the search criteria:

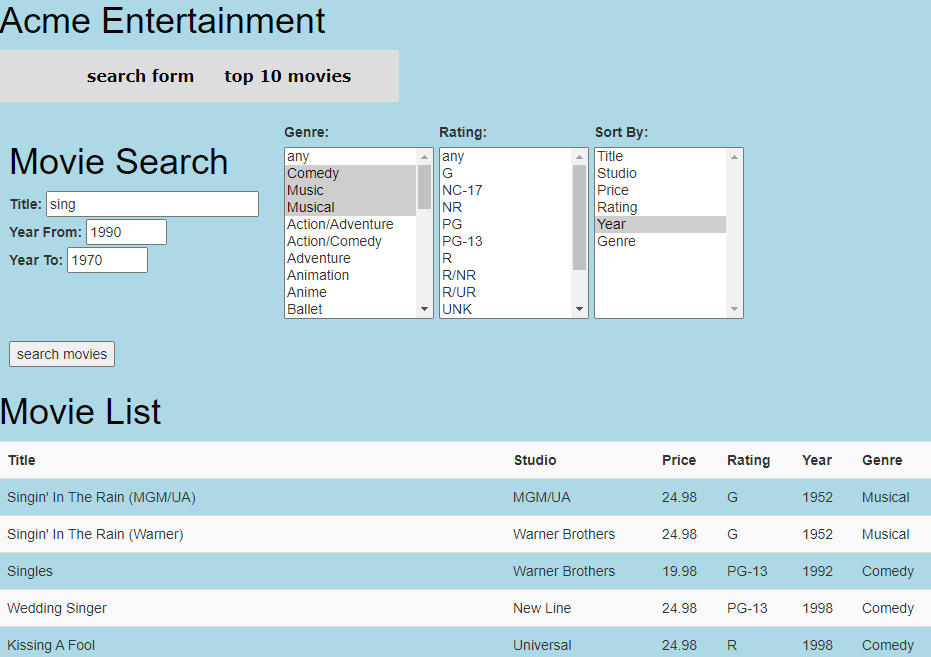


Figure 12: including Genre values in the search

The rating was constrained to G, PG and PG-13 and sorted by Rating:

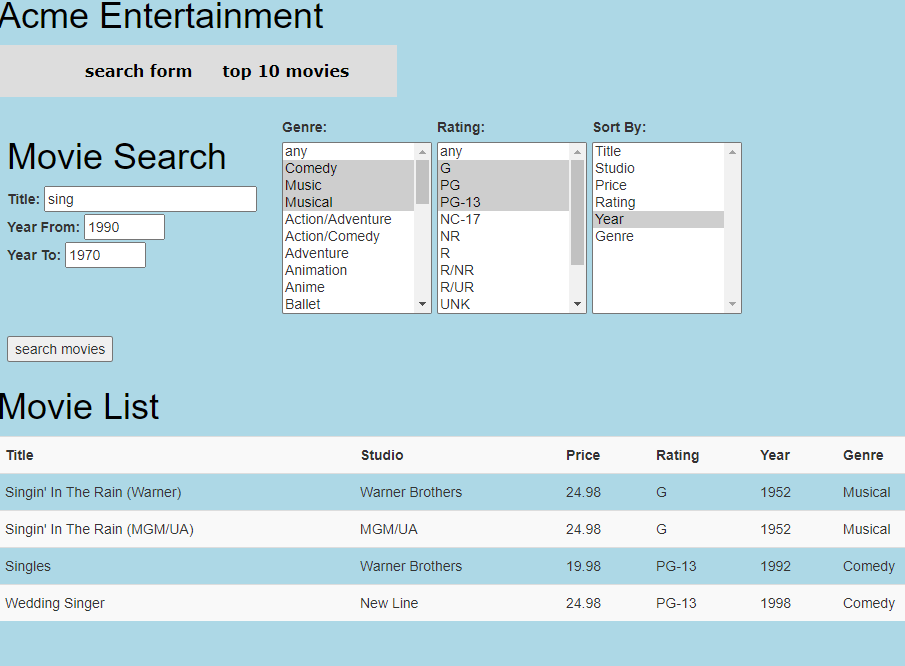


Figure 13: searching using all constraints

All the search elements were tested individually and in combination, the results have not been included for brevity.

A screen capture of the Top 10 page is presented below:

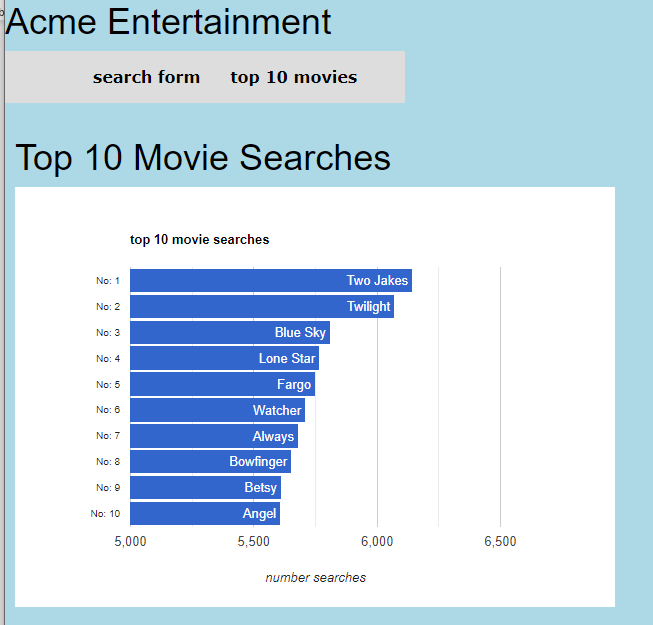


Figure 14: The Top 10 page

### Adaptive Web Design

A different approach was adopted for the two web pages.

The elements of the search page can be divided up into a series of logical blocks. The title and year search inputs, the three selection lists and the report area. These blocks can be rearranged independently to make the best use of the available screen space. The figure below shows the different blocks:

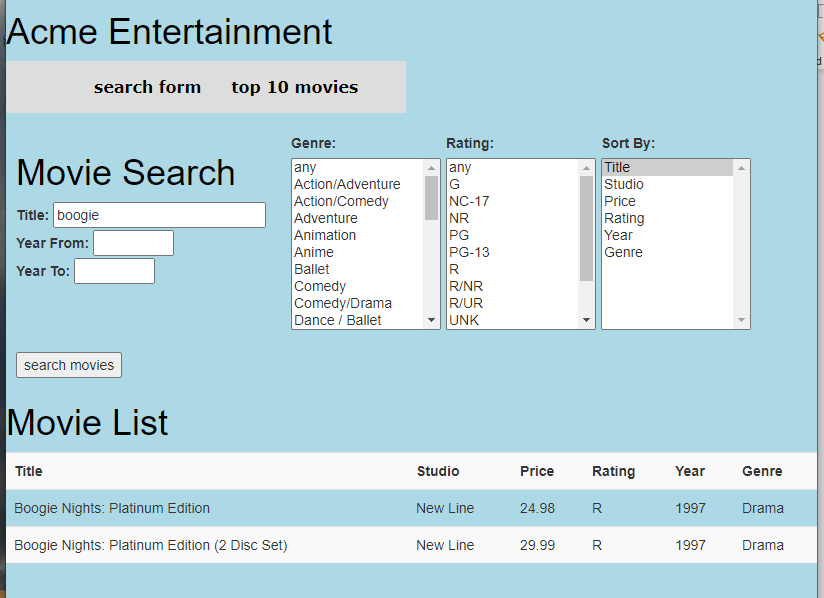


Figure 15: search screen arrangement

Three screen widths were targeted, less than 600 pixels, between 600 and 800 pixels and greater than 800 pixels.

For a screen width of greater than 800 pixels the search controls are arranged on a single line as shown in Figure 15 above.

For a screen width between 600 and 800 pixels the search controls are arranged into a 2 x 2 pattern, as shown in Figure 16 below:

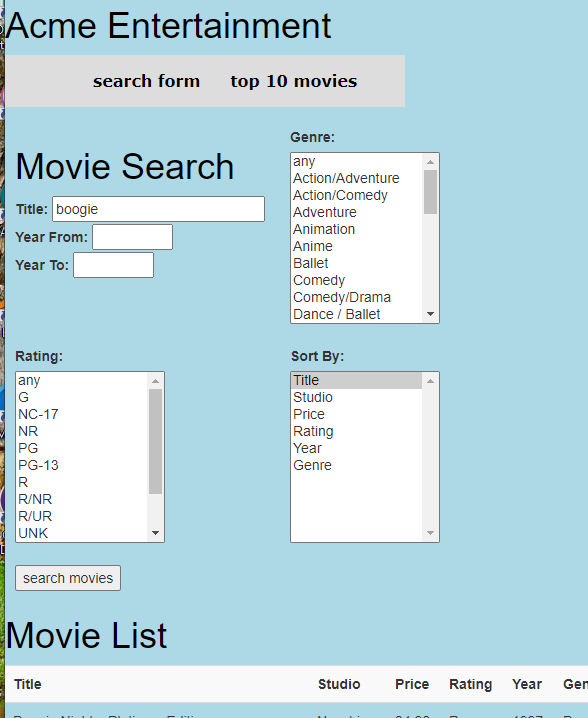


Figure 16: 2 x 2 pattern search screen

For a screen width of less than 600 pixels the search controls are arranged into a single column, as shown in Figure 17 below:

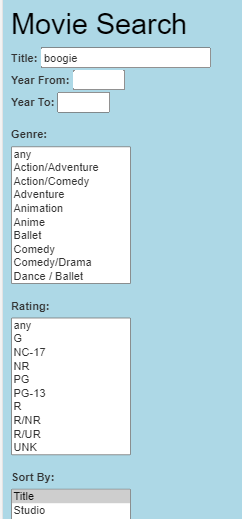


Figure 17: search screen arranged in a column

The top 10 search page displays a chart of the ten most searched for movies. The page creates three version of the chart each targeted at a range of screen widths. The versions are summarised in the following table:

|  |  |  |
| --- | --- | --- |
| Chart Size | Chart Dimensions | Screen Widths |
| Small | 325px x 220px | < 400px |
| Medium | 400px x 280px | >= 400px < 600px |
| Large | 600px x 420px | >= 600px |

Table 5: top 10 chart sizes

A screen dump of the small chart is shown below:

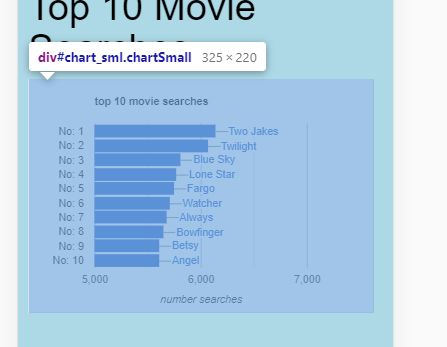


Figure 18: top 10 chart small version

A screen dump of the medium chart is shown below:

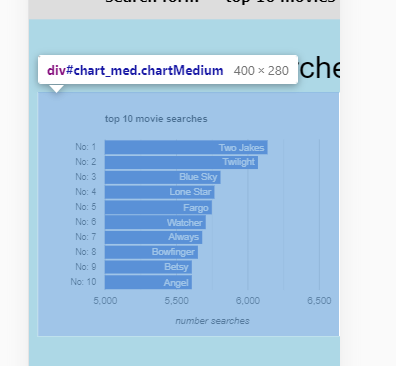


Figure 19: top 10 chart medium version

A screen dump of the large chart is shown below:

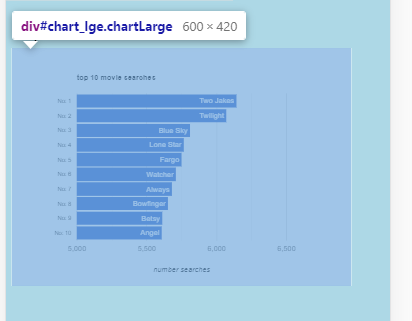


Figure 20: top 10 chart large version

# Sprint 2

## Meeting Agenda and Minutes

AGENDA

RAD Sprint Two Meeting

5/6/20

14:00 – 15:00

Meeting called by Zara Duncanson

|  |  |
| --- | --- |
| Attendees: | Zara Duncanson  David Perry  Alan Pederson |
| Please read: |  |
| Please bring: |  |

|  |  |  |
| --- | --- | --- |
| 14:00 – 15:00 | Introduction  Review Sprint One  Discuss GitHub  Share ideas for website functions and layout  Allocate tasks | Thornlie TAFE  Library |

Additional Instruction:

|  |  |
| --- | --- |
| Team Meeting | 05/06/20  14:00  Thornlie TAFE Library |

|  |  |  |  |
| --- | --- | --- | --- |
| Meeting called by: | Zara Duncanson | Type of meeting: | Sprint Two Meeting |
| Facilitator: |  | Note taker: | David Perry |
| Timekeeper: |  |  |  |

|  |  |
| --- | --- |
| Attendees: | Zara Duncanson (ZD)  David Perry (DP)  Alan Pederson (AP) |
| Please read: |  |
| Please bring: |  |

Minutes

|  |  |  |  |
| --- | --- | --- | --- |
| Agenda item: | Software Review Plan to be discussed | Presenter: | All |

Discussion:

A debrief of Sprint One tasks and the presentation was carried out. The team had a technical discussion regarding how the new pages would be laid out and function. The team agreed clarification was required regarding email function, admin login and software review plan.

Conclusions:

Clarification required

| Action items | Person responsible | Deadline |
| --- | --- | --- |
| * Email Stewart with questions | ZD | 7/6/20 |

|  |  |  |  |
| --- | --- | --- | --- |
| Agenda item: | GitHub access discussion | Presenter: | All |

Discussion:

The uploading of files to GitHub was discussed, some research was carried out to allow ZD and DP to collaborate with the team GitHub.

Conclusions:

DP and ZD to gain more access to GitHub

| Action items | Person responsible | Deadline |
| --- | --- | --- |
| * AP to invite ZD and DP to collaborate | AP | 5/6/20 |

|  |  |  |  |
| --- | --- | --- | --- |
| Agenda item: | Allocate Tasks for Sprint Two | Presenter: | ZD |

Discussion:

The tasks were discussed and allocated to give members different a different job than Sprint One.

Conclusions:

Members allocated new tasks

| Action items | Person responsible | Deadline |
| --- | --- | --- |
| * Develop Project Plan | AP | 6/6/20 |
| * Develop Report | DP | 8/6/20 |
| * Write code for web page requirements | ZD & AP | 10/6/20 |

Other Information

Observers:

Resources:

Special notes:

## Sprint 2 Project Plan

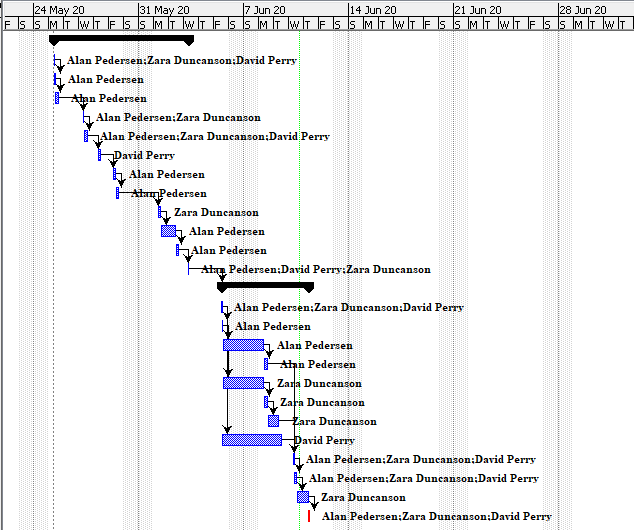


Figure 21: Sprint 2 Gantt chart.

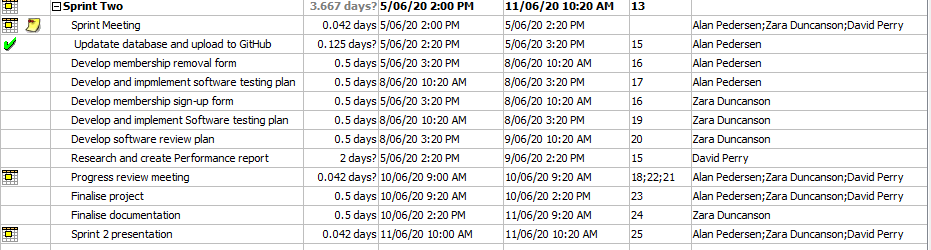


Figure 22: Sprint 2 project Plan

## Software Review Plan

### Introduction

The product under review is a web-based application for a Movie Database. The application presents a series of web forms and reports for user access as well as offering restricted access admin functionality.

The database will be hosted in MySQL, the web functionality will be provided by Apache using PHP server-side scripting.

### Plan Purpose

The purpose of this plan is to ensure that the solution created mees all CITE business rules and quality assurance standards. The solution will also be tested for compliance with the specifications specified by ACME Entertainment.

### Review Schedule

The Application is to be developed over four cycles. The first three cycles will focus on developing the application with the final cycle focused on addressing any outstanding issues before handover. A progress review will be undertaken at the end of each development cycle focusing on code systems developed during that cycle. A review of the entire application will be undertaken before final handover.

### Progress Review

At the end of each development cycle the following checks will be undertaken:

* Code Review: A review of the code developed by another team member checking for compliance with CITE standards
* Code Demonstration: The operation of the code will be demonstrated for the other team members
* Code Testing: The operation of the code will be validated by testing each function. Each function should be tested with both valid and invalid entries. The results of the testing should be recorded along with any issues identified
* Feedback: Other team members to provide feedback on the quality and operation of the code
* Product Demonstration: The product will be demonstrated to the client at the end of each cycle. All aspects of the code are to be demonstrated including valid and invalid operation. Feedback received from the client is to be incorporated into the next cycle

### Final Review

The final review will revisit all the code developed across the cycles. Focus is to be given to areas where subsequent development may have impacted on the code base.

The final review will compile all the user requirements and feedback received over the three cycles and document how these have been fulfilled.

A final presentation is to be given to the Client to demonstrate compliance with all requirements.

## Performance Report

### Performance Tools

The performance of a website is a critical factor; it should not be overlooked in the development process. A site could look good, have all the functionality required, but have excessive load and performance times which would be detrimental to the user’s experience. A website performing poorly could be all it takes for the user to become frustrated and look for a better option. If the purpose of the site is to sell a product, this could lead to missing sales, and reduce site traffic.

Like the graphics of a website, less can be more when keeping the site at optimal performance. But the best way to ensure functionality is not lost due to performance, it is to use tools to perform testing on the speed and determine what functions perform poorly. Some tools for testing include Pingdom, GTmetrix and Google Chrome inspect element. These tools provide a simple waterfall report of the tasks and their speed.

Using Googles Chrome’s built in tools would be the most appropriate for this project as other tools need a proper URL to be input, rather than using the localhost link. The tools are easily accessible by right clicking on the page and navigating to the network and performance tabs which offer some good insight. The duration of a function is recording in milliseconds and useful charts display the data.

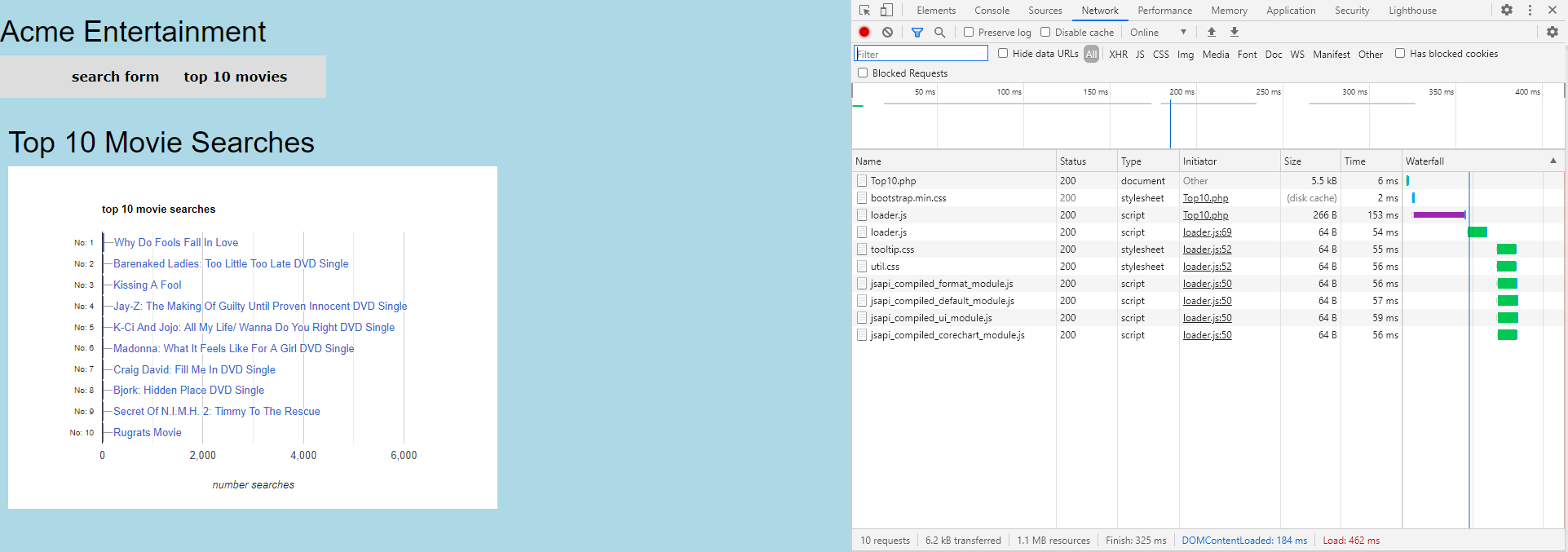


Figure 23: Using Google's inspect element to display a waterfall performance view

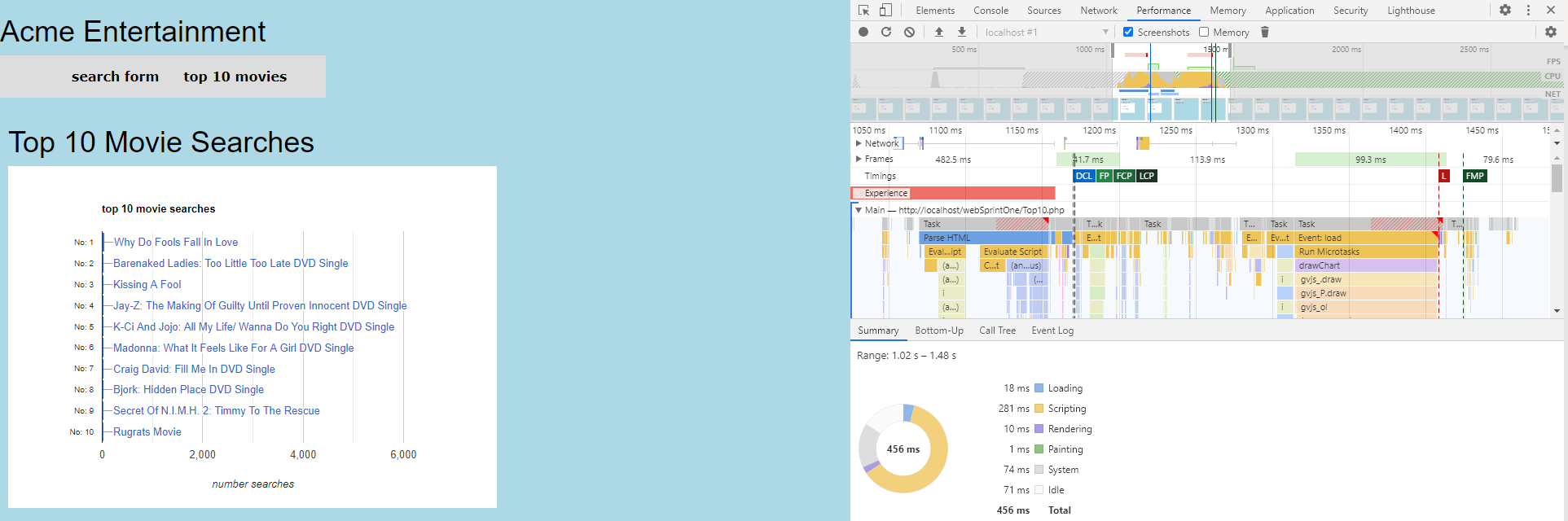


Figure 24: Using Google's inspect element lets you take a closer look at duration of each function

The tools mentioned which capture the performance are good, but there are tools to offer a more detailed insight and can suggest how to improve the performance too. Google Page Speed Insights shows a score out of 100 based on several factors that are known to cause slow load times. It also lists the suggestions of making the page faster with the estimated saving time per task. Google insights is available an extension of the Chrome browser so this tool could be used in the development of the project.

### Code Optimizers

Code optimizers check the code for common flaws such as unused variables, unnecessary object creation, incorrect white space, and lines of code with too many characters. The optimizer tools also allow code standards to be followed which can be useful for large development teams. Having different developers, with different backgrounds can result in inconsistent code which is harder to read and maintain. Running a code optimizer can save a lot of time which would be wasted by the developers trying to follow a style every step along the way. The following tools are available for PHP.

* PHP CS Fixer – Automatically fixes coding standards. Can follow defined coding standards or your teams’ style through configuration.
* PHPCD – A copy/paste detector which scans for duplicated code. The idea is that when you copy and paste code, there is the possibility for bugs to be introduced when logic is updated at one place but not another.
* PHP Mess Detector - Lets you examine code with a combination of rules. Typically used for code size rules and locating overly complicated code and bug sources.
* PHP Code Sniffer – Works by sniffing the code with a preconfigured standard or custom set of rules. Identifies missing comments, incorrect white space. Automatically fixes the errors with the code beautifier following standards such as PEAR.

The PHP Code Sniffer tool could be a good optimizer on the project. Since there are multiple developers writing code, the scripts can be run in the sniffer and a standard can be checked against. It runs in the command window and is an easy tool to work with. You can customize the standards if too many unwanted errors are being produced. The errors are customizable and can be changed to warnings or be removed if they are irrelevant.

## Source Control Snapshot

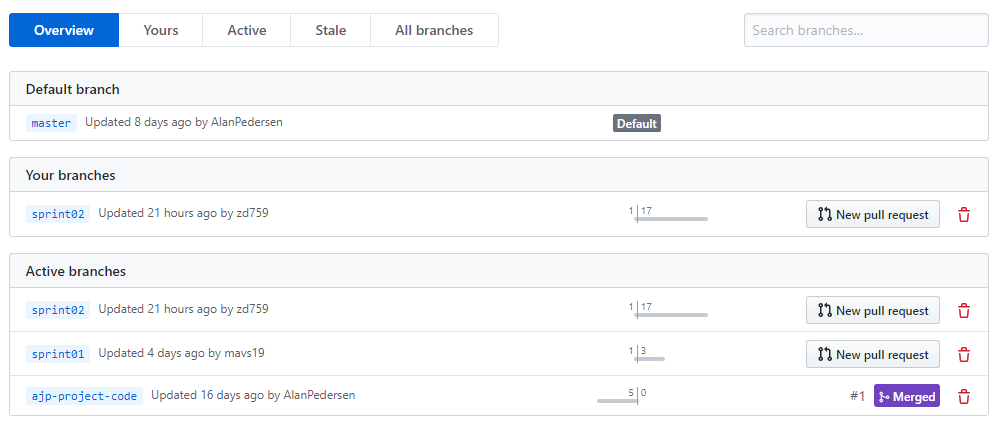


Figure 25: Branch view of Source control.

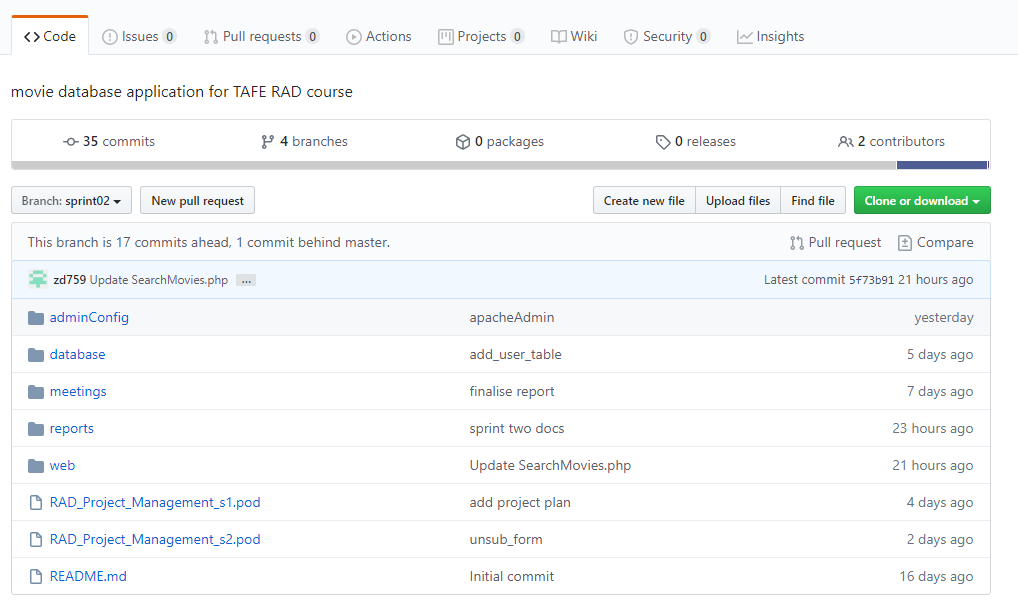


Figure 26: Source control Sprint Two.

# Sprint 3

## Meeting Agenda and Minutes

AGENDA

RAD Sprint Two Agenda

11/06/20

14:00 – 15:00

Meeting called by David Perry

|  |  |
| --- | --- |
| *Attendees:* | Zara Duncanson  David Perry  Alan Pederson |
| Please read: |  |
| Please bring: | Notebook, pen, Sprint 3 requirements resource |

|  |  |  |
| --- | --- | --- |
| 14:00 – 14:10 | Introduction  Review Sprint 2 and presentation issues | Thornlie TAFE  Library |
| 14:10 – 14:20 | Sprint 3 Review Requirements  Item 1 – Users and groups | Thornlie TAFE  Library |
| 14:20 – 14:30 | Sprint 3 Review Requirements  Item 2 – Adding ratings and new chart | Thornlie TAFE  Library |
| 14:30 – 14:40 | Sprint 3 Review Requirements  Item 3 – Auto-refresh function for chart, is socket API required | Thornlie TAFE  Library |
| 14:40 – 14:50 | Sprint 3 Review Requirements  Item 4 – Implementing performance tools | Thornlie TAFE  Library |
| 14:50 – 15:00 | Conclusion  **Summary of tasks and responsibilities** | Thornlie TAFE  Library |

Additional Instruction:

|  |  |
| --- | --- |
| Team Meeting | Date: 11/06/2020  Time: 14:00  Location: Thornlie TAFE Library |

|  |  |  |  |
| --- | --- | --- | --- |
| Meeting called by: | David Perry | Type of meeting: | Sprint Three Meeting |
| Facilitator: |  | Note taker: | Zara Duncanson |
| Timekeeper: |  |  |  |

|  |  |
| --- | --- |
| Attendees: | Zara Duncanson (ZD)  David Perry (DP)  Alan Pederson (AP) |
| Please read: | Sprint Three requirements resource |
| Please bring: | Notebook, pen, Sprint Three requirements resource. |

Minutes

|  |  |  |  |
| --- | --- | --- | --- |
| Agenda item: | Review of Sprint 2 and presentation issues | Presenter: | All |

Discussion:

ZD did present the criteria under the difficult circumstances. Everything went well and the client was satisfied with the progress.

Conclusions:

A debrief of Sprint Two tasks and the presentation was carried out.

| Action items | Person responsible | Deadline |
| --- | --- | --- |
| * None | ZD | N/A |

|  |  |  |  |
| --- | --- | --- | --- |
| Agenda item: | Sprint 3 Review Requirements  Item 1 – Users and groups | Presenter: | All |

Discussion:

The need for users and group in the database SQL requirement was discussed, the team determined they would need an administrator user that would not affect the root user which is used by scripts to perform queries. Also read only (regular) users will be required.

Conclusions:

Concluded DP will handle this task.

| Action items | Person responsible | Deadline |
| --- | --- | --- |
| * Implement user groups in SQL database | DP | 17/06/20 |

|  |  |  |  |
| --- | --- | --- | --- |
| Agenda item: | Sprint 3 Review Requirements  Item 2 – Adding ratings and new chart | Presenter: | All |

Discussion:

Ratings need implementation. The team discussed how this would work both as an interface and as a database table view to store data. Users would not have to log in to rate. Ratings would need to be averaged, etc.

Conclusions:

AP to carry out investigating and implementing this task.

| Action items | Person responsible | Deadline |
| --- | --- | --- |
| * Implement way to rate movie titles, top 10 chart | AP | 17/06/20 |

|  |  |  |  |
| --- | --- | --- | --- |
| Agenda item: | Sprint 3 Review Requirements  Item 3 – Auto-refresh function for chart, is socket API required | Presenter: | All |

Discussion:

The refresh function required could be implemented a few different ways and requires further research. Team members discussed the possible potential of using socket API.

Conclusions:

All the team will research ways to best implement this feature.

| Action items | Person responsible | Deadline |
| --- | --- | --- |
| * Implement way to auto refresh page when database updated | AP, DP, ZD | 17/06/20 |

|  |  |  |  |
| --- | --- | --- | --- |
| Agenda item: | Sprint 3 Review Requirements  Item 4 – Implementing performance tools | Presenter: | All |

Discussion:

Conclusions:

DP to carry out investigating and implementing this task.

| Action items | Person responsible | Deadline |
| --- | --- | --- |
| * Implement the previously investigated performance tools | DP | 17/06/20 |

|  |  |  |  |
| --- | --- | --- | --- |
| Agenda item: | Sprint 3 Allocate Tasks  Other items and conclusion | Presenter: | All |

Discussion:

DP as sprint leader assigned tasks for Project Plan and minutes & agenda writing to ZD. A general conclusion discussion followed with any clarification questions and a summary of tasks.

Conclusions:

ZP to write project plan for sprint 3 and type meeting minutes.

| Action items | Person responsible | Deadline |
| --- | --- | --- |
| * All team members to perform assigned tasks | ZD | 17/06/20 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

Other Information:

Special Notes:

## Source Control Snapshot

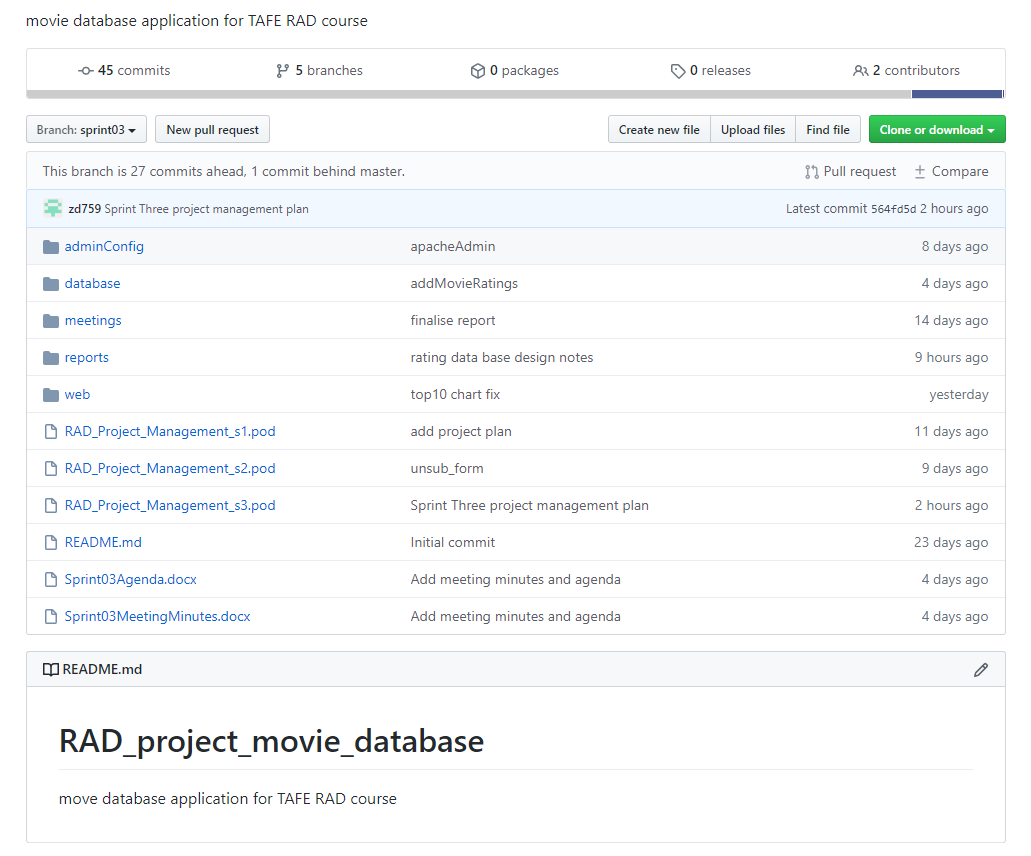


Figure 27: sprint 3 source control snapshot

## Project Plan

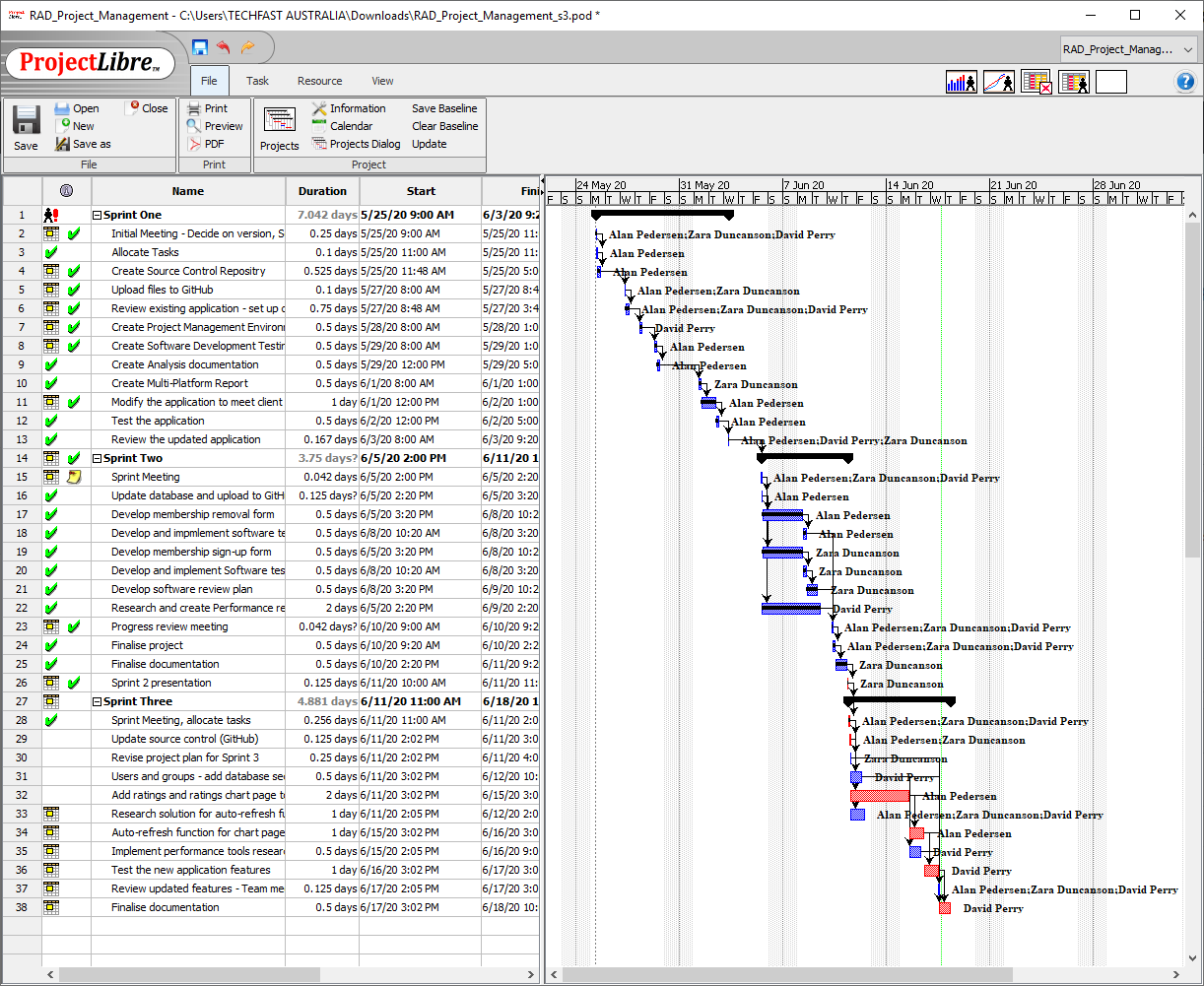


Figure 28: sprint 3 project plan

## Administration login and user groups

The database has been configured to prompt a user for a login to access. An administration account has been made with the name admin which has all the privileges allowed for the movie database. This includes all options relating to modifying the database.

An additional two users were created which are staff members with restricted access. This is based on the scenario that a staff member can have access to add movies to the database, which can be modified if needed. But they are not allowed to carry out certain tasks such as deleting records.

A user group has been made for acme staff were the two staff users are added. The privileges in the group determine what is displayed if these users are accessing the database. Although other restrictions are already in place for each user, members of the group only have a select number of tabs and options visible.

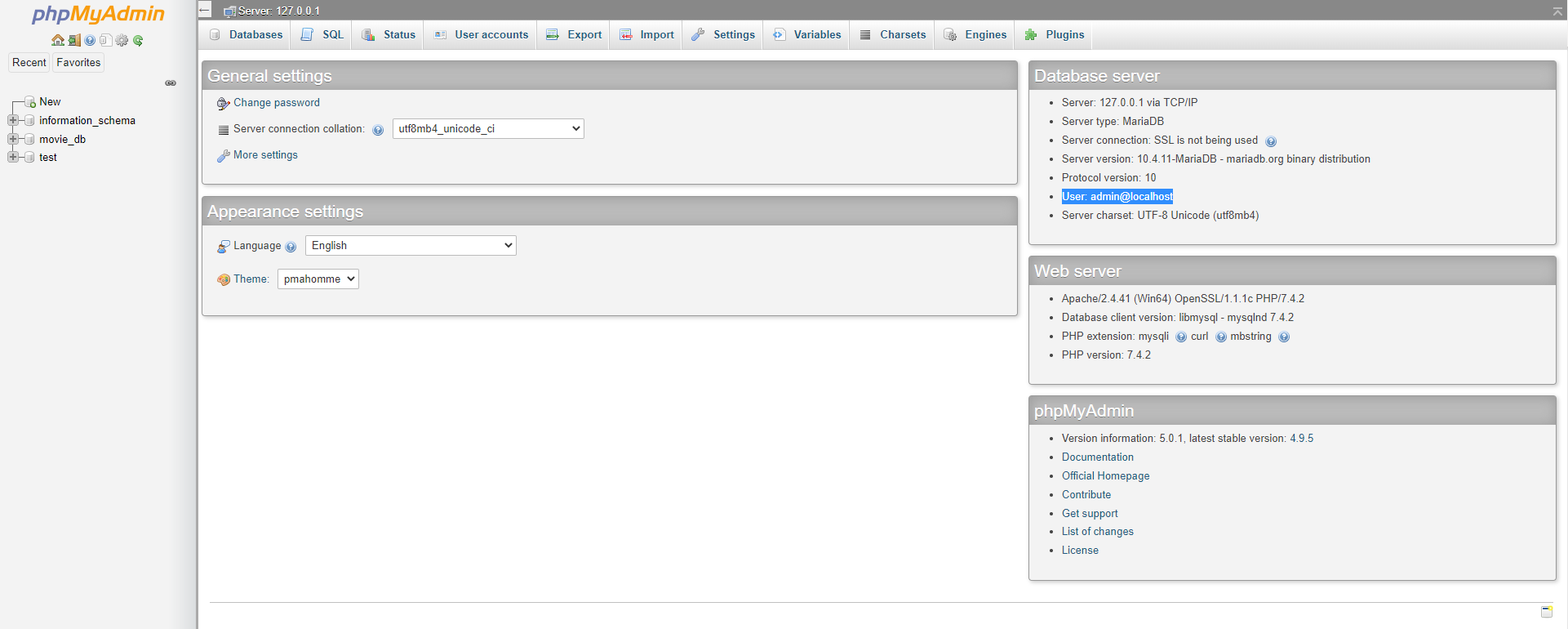


Figure 29: Admin user has access to all functions related to the movie database

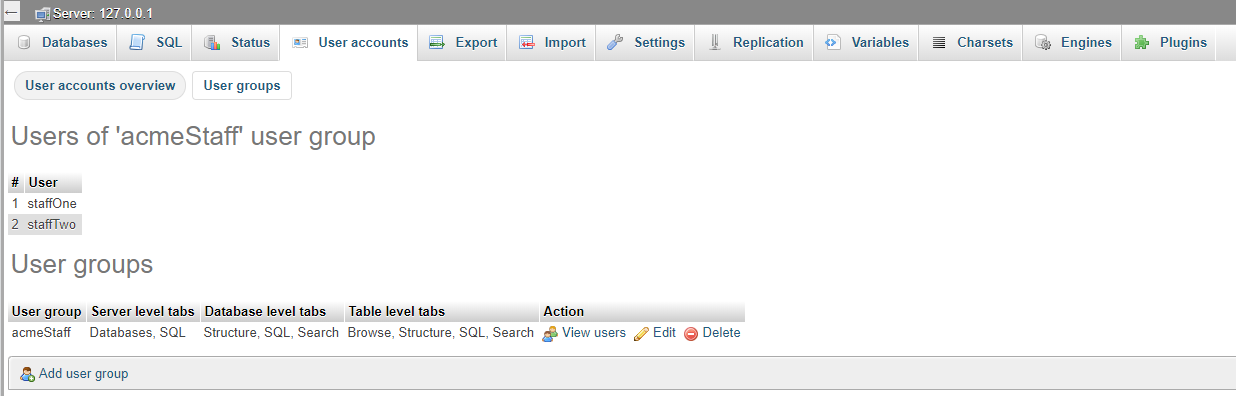


Figure 30: The user group "Acme Staff" contains users, restricts the view of tabs not required by each user

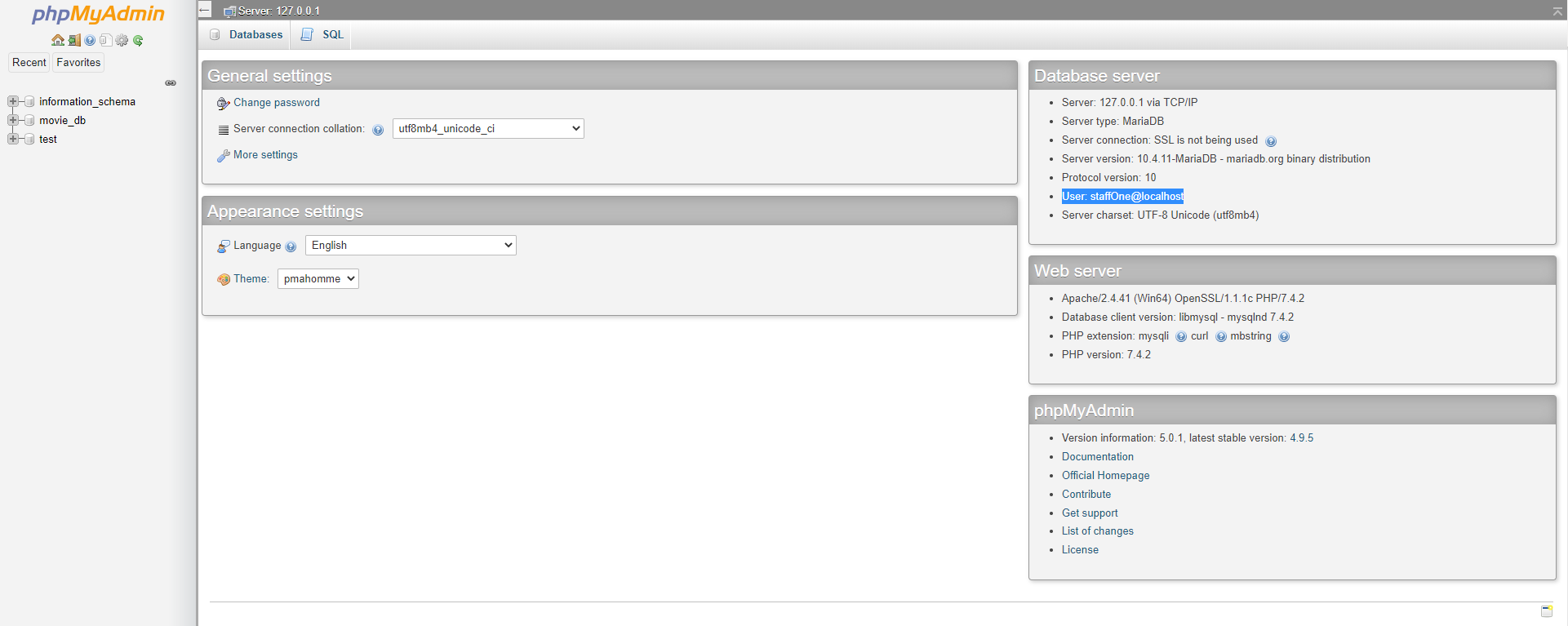


Figure 31: View of the limited tabs available to a staff user

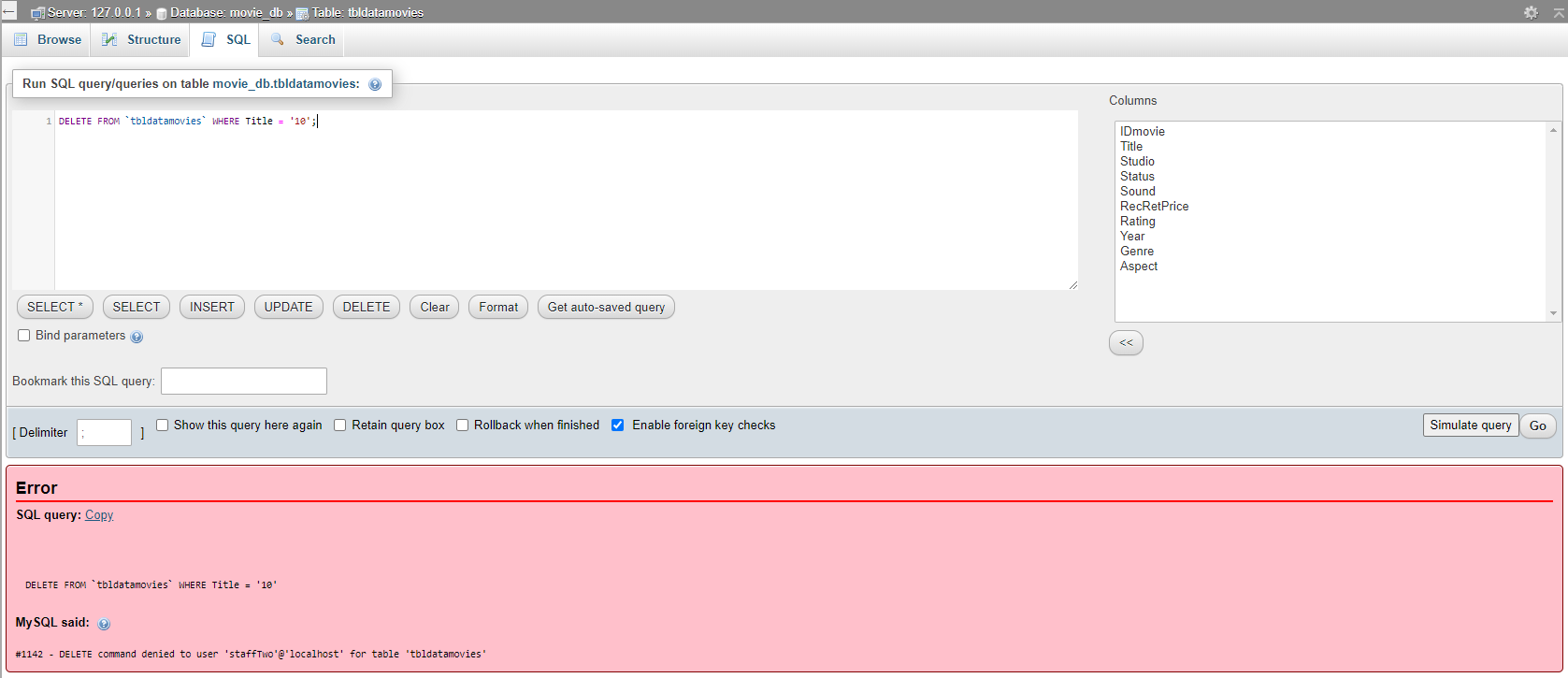


Figure 32: Staff users have limited controls of queries such as deleting data from the tables

## Optimization Report

A series of tests have been carried out on all the pages to check the speed of the initial load, and the performance of running queries which retrieves data from the database. Google development tools have been used as these are a great tool to use in the development as they offer performance checks on pages loaded with local host.

Overall, the pages all returned excellent results with no recommendation to make any modifications. Some slight modifications can be made to the top 10 page which shows the charts with the CSS but are not necessary. The pages were all tested on a desktop option, mobile option and with a slower internet speed scenario which all returned similar results to the screenshots mentioned in the report. The default tests were carried out on the desktop option.

### SearchMovies.php

The search movies page contains a form to input search data for the title and years of movies. There are also three boxes displaying the available genre, rating, and sort by options to be used with the search. To load the page, scripts will run to retrieve the relevant data from the respective database tables and populate the display.

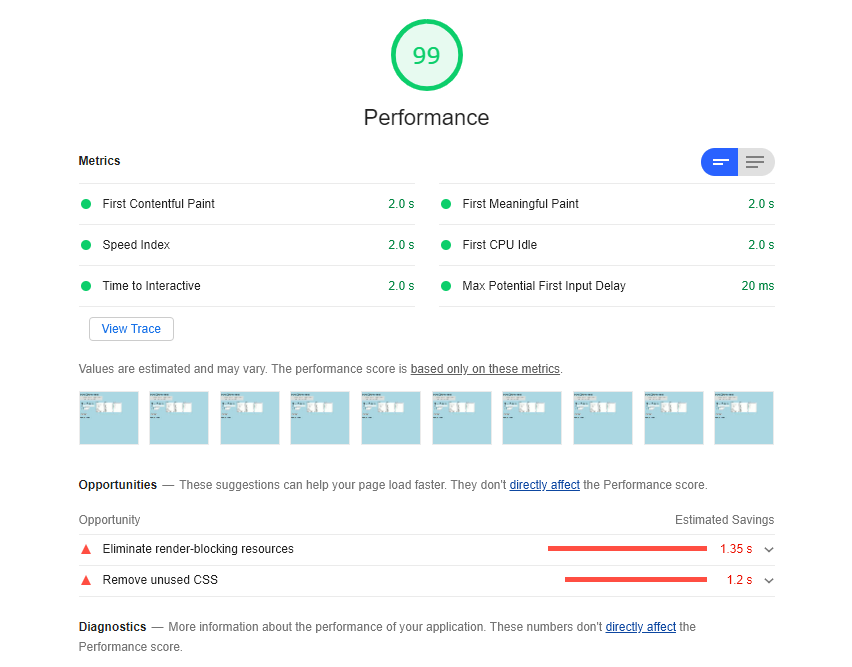
The performance of this page returns 99 which is an excellent result when it is loaded. A test to search for all the movies was carried out which retrieves and displays all 2298 movies. This was completed in roughly half a second which is a great result. Searching a query with different criteria such as selecting the genre and rating returns a fast result of around 100 milliseconds, though these times will be determined by how many rows are displayed depending on the search.

Figure 33: Loading the Search Movies page

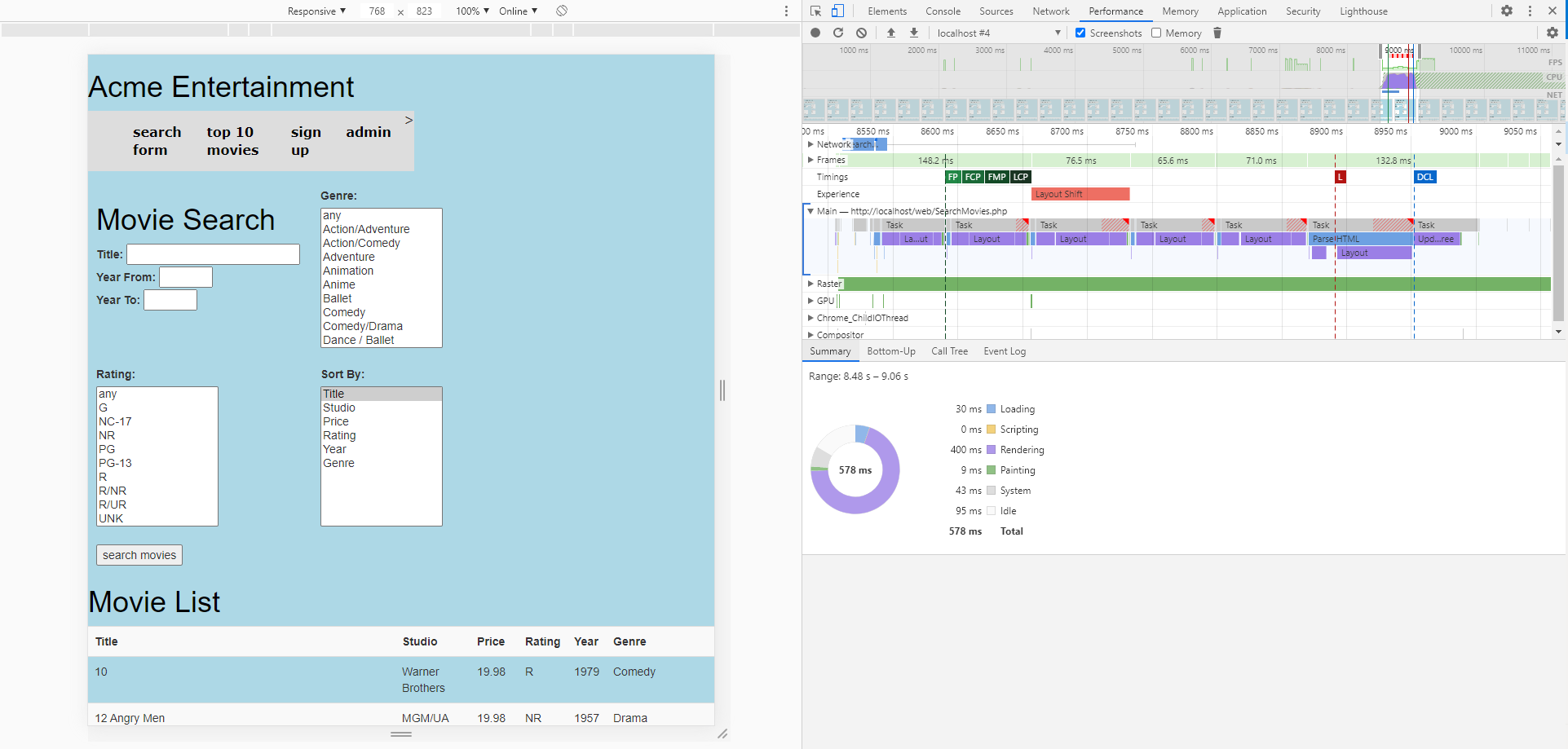


Figure 34: Loading all movies from database

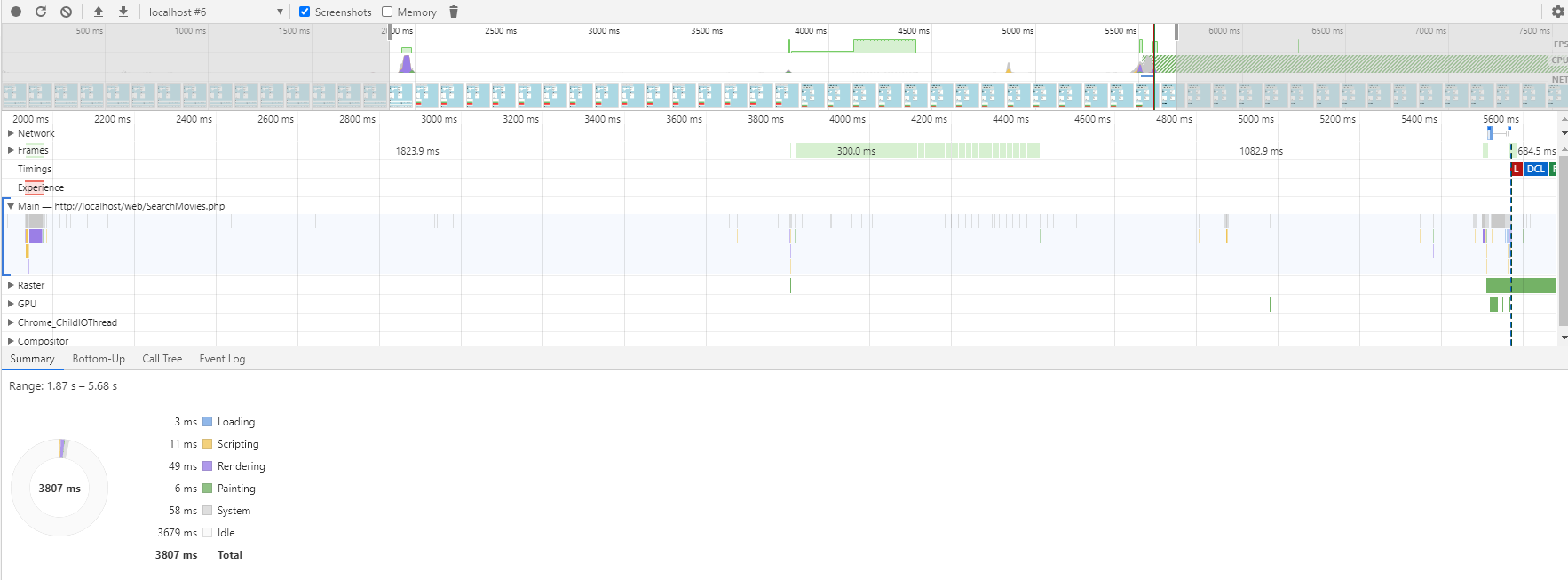


Figure 35: Searching for a combination of rating and genre

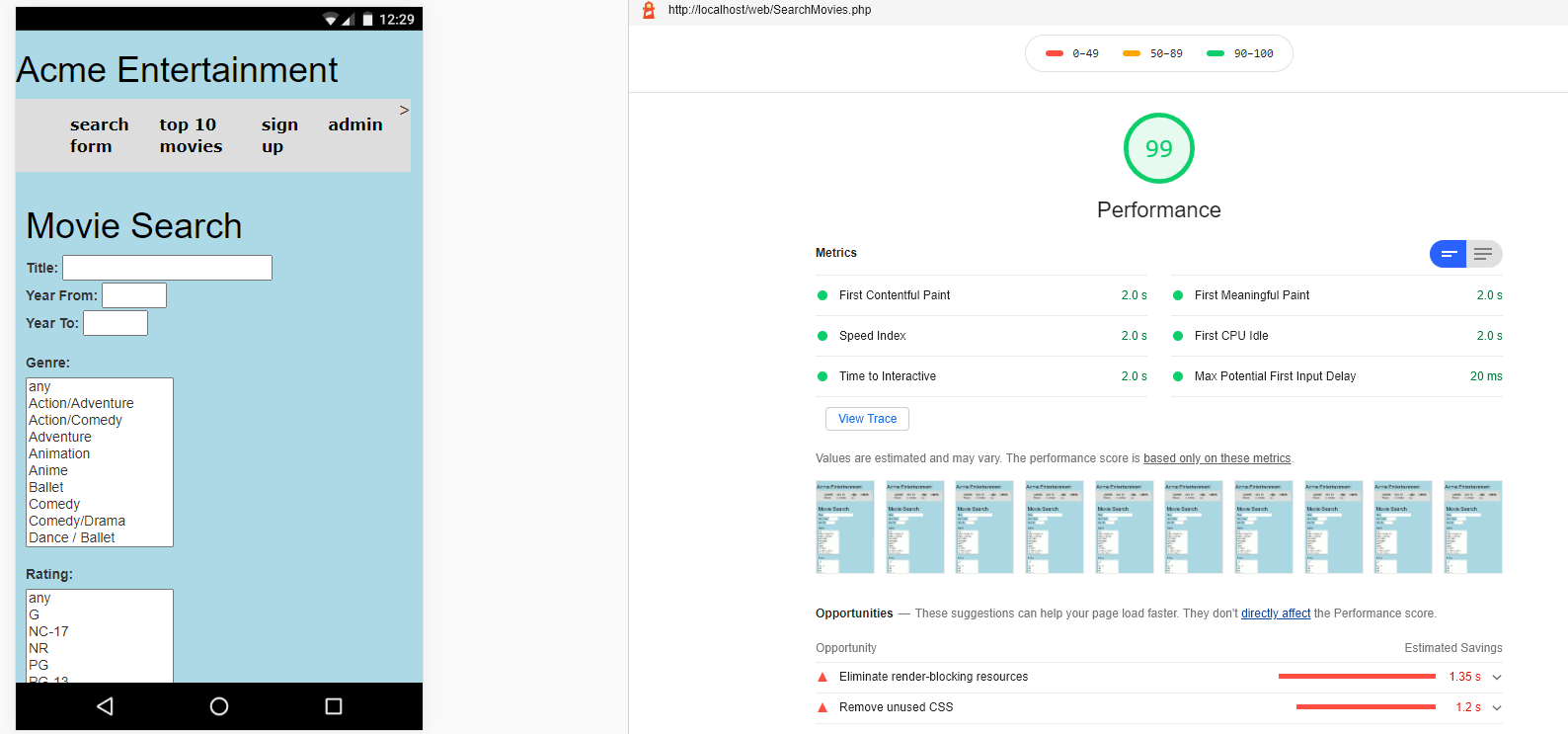


Figure 36: Loading the Search Movies page in mobile format

### SignUp.php

The sign-up page also returned a performance score of 99. This page has a form for users to sign up and opt out of a newsletter and newsflash. To reach a score of 100 the CSS would need to be modified. A test snapshot of signing up a user was carried out which adds the data to the database. This took under 50 milliseconds, so it is an almost instant result requiring no modification.

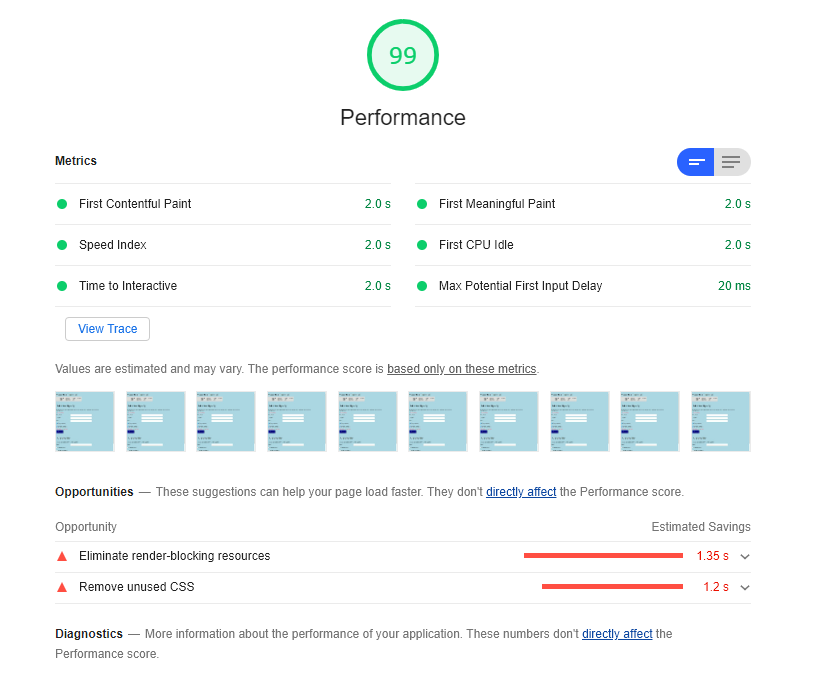


Figure 37: Loading the Sign-Up page

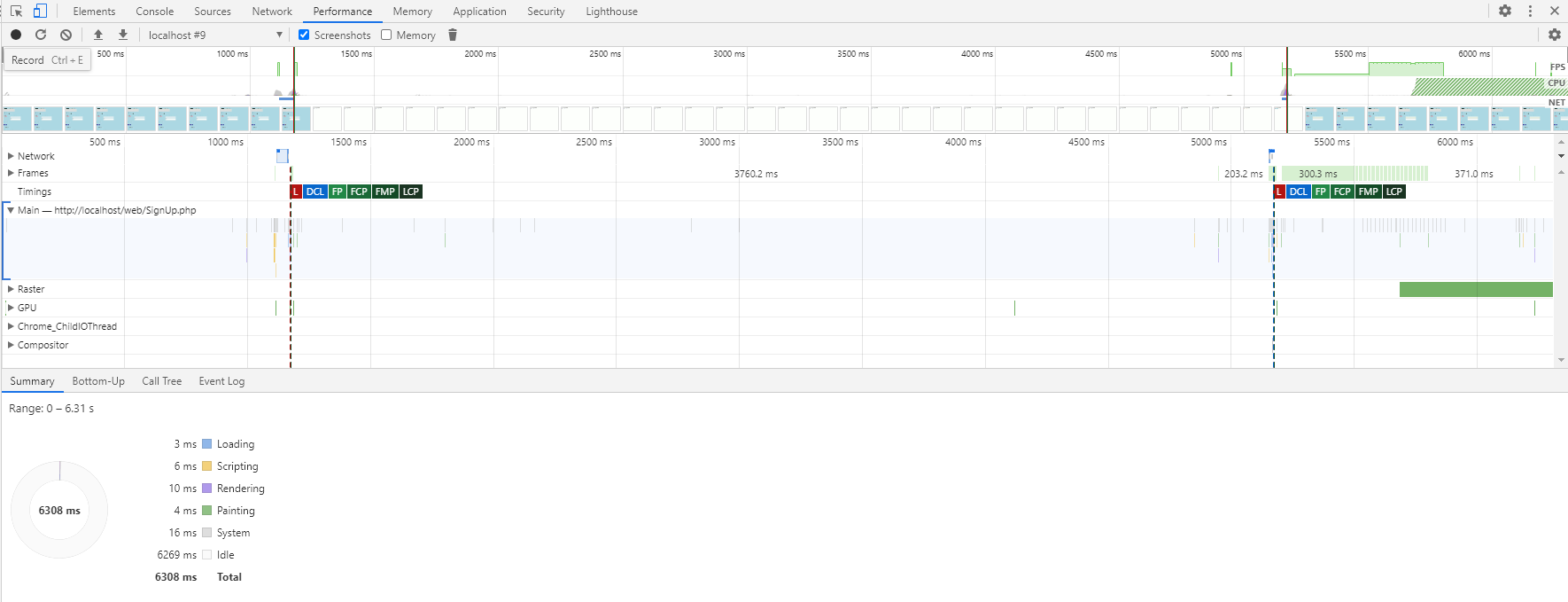


Figure 38: Using the form to sign up a user

### Top10.php

The top ten page displayed the slowest response of all the pages, but still returns a result of 94. It does need to retrieve data which populates a google chart which need to be drawn. The google charts slow the process slightly so it is expected to perform slightly slower than the other pages. Modifications with the CSS will change the result, but it is an acceptable performance.

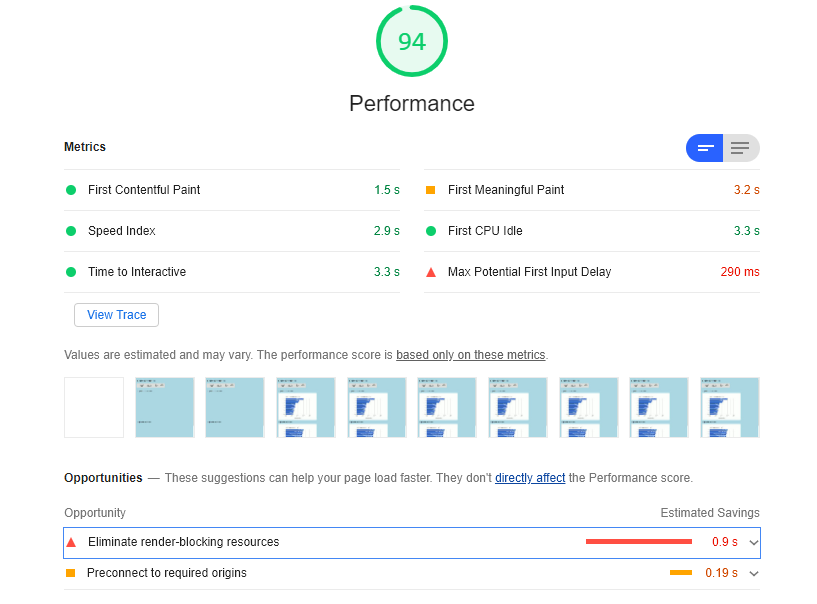


Figure 39: Loading the Top Ten page

### UnsubscribeUsers.php

The unsubscribe users page has a performance result of 100. It retrieves users’ details from the database and populates a display box. A form is printed which allows the admin to search for a user for the purpose of unsubscribing them from the mail list. The test to record the speed of unsubscribing a user returned a fast speed of under 20 milliseconds. No change to the page is required after the unsubscribe button is pressed which can explain the faster speed. The list does repopulate with the changes made to the database.

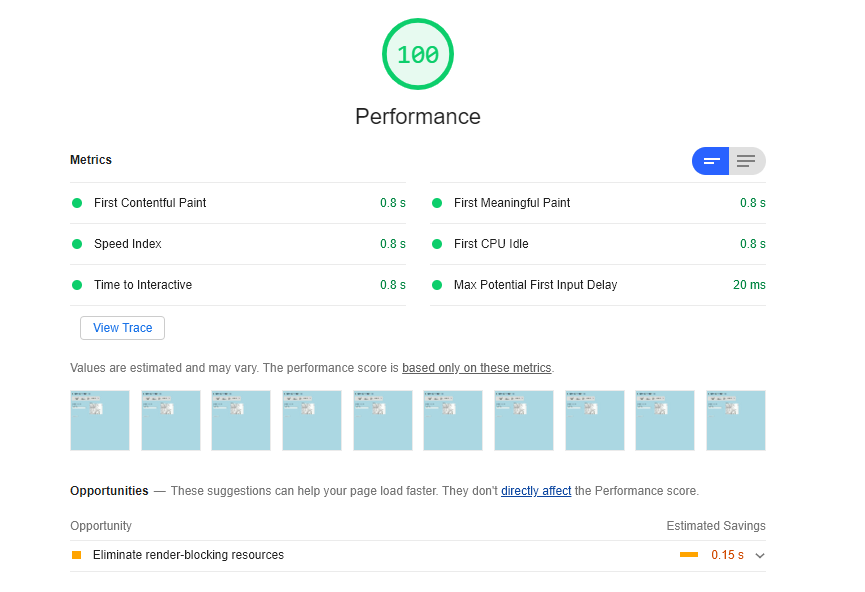


Figure 40: Loading the Unsubscribe Users page

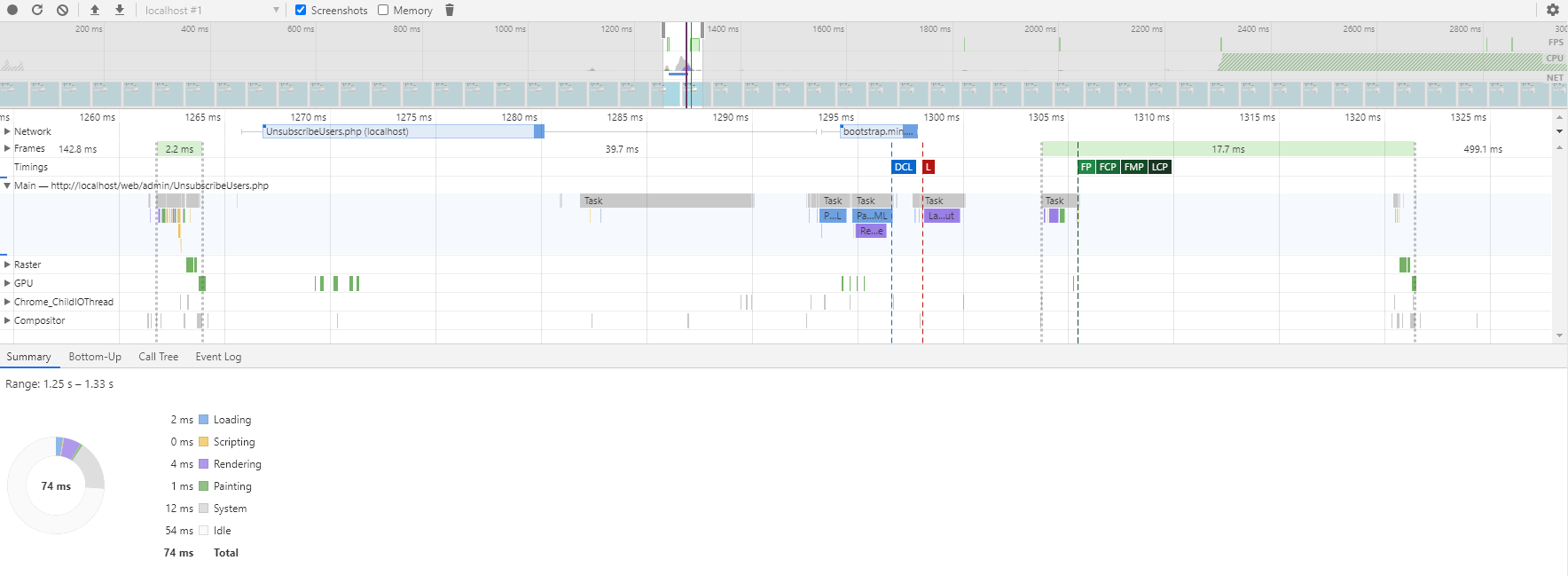


Figure 41: Unsubscribing a user from mail list

## Movie rating system design and testing

ACME Entertainment have requested that a chart be added to the application displaying the top ten movies as rated by users. Users should be able to assign a rating to each movie. Ratings will range from one star to five stars. The individual counts will be used to calculate an average rating.

To facilitate this functionality an additional table will be required to store the user ratings. The table should store the number of times each movie has been assigned each rating value along with an overall average rating.

The following table definition could be used to store the data:

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Field Type | Default | Comment |
| IDrating | Int |  | ID number used to link to tblDataMovies via the IDmovie column |
| 1starCount | Int | 0 | Number of 1-star ratings |
| 2starCount | Int | 0 | Number of 2-star ratings |
| 3starCount | Int | 0 | Number of 3-star ratings |
| 4starCount | Int | 0 | Number of 4-star ratings |
| 5starCount | Int | 0 | Number of 5-star ratings |
| movieRating | Float |  | Calculated field the average rating is calculated using the following formula:  ((1starCount) + (2starCount\*2) + (3starCount\*3) + (4starCount\*4) + (5starCount\*5)) / (1starCount + 2starCount + 3starCount + 4starCount + 5starCount) |

Table 6: movie rating table definition

The following SQL command can be used to create the table:

CREATE TABLE tbldatamovierating (

`IDrating` INT NOT NULL,

`1starCount` INT NOT NULL DEFAULT '0',

`2starCount` INT NOT NULL DEFAULT '0',

`3starCount` INT NOT NULL DEFAULT '0',

`4starCount` INT NOT NULL DEFAULT '0',

`5starCount` INT NOT NULL DEFAULT '0',

`movieRating` FLOAT AS ((1starCount + (2starCount\*2)

+ (3starCount\*3) + (4starCount\*4) + (5starCount\*5))

/ (1starCount + 2starCount + 3starCount + 4starCount + 5starCount))

);

Figure 42: rating table SQL

An SQL query can be used to initialise the records in the ratings table setting all values to 0:

INSERT INTO tbldatamovierating

(IDrating, `1starCount`, `2starCount`, `3starCount`, `4starCount`, `5starCount`)

SELECT IDmovie, 0 AS `1starCount`, 0 AS `2starCount`, 0 AS `3starCount`,

0 AS `4starCount`, 0 AS `5starCount`

FROM `tbldatamovies`

Figure 43: SQL to initialise the ratings table

It is anticipated that users will assign ratings to movies by selecting the title value. The assigned rating will then be added to the count of the assigned rating value. For example, if a user assigns a five star rating the number of five star ratings will be incremented by one.

To facilitate this interaction a view should be created linking the movie title to the ratings table. This will allow the ratings table to be updated using the movie title as the control.

The following SQL can be used to create the view:

CREATE VIEW vwTitleRating AS

SELECT Title, 1starCount, 2starCount, 3starCount, 4starCount, 5starCount, movieRating

FROM `tbldatamovies`

LEFT OUTER JOIN tbldatamovierating ON IDmovie = IDrating;

Figure 44: rating update view SQL

The ratings can be updated using a simple query such as the following to assign a 5-star rating to the film “Clockwork Orange”:

UPDATE `vwtitlerating`

SET `5starCount` = `5starCount` + 1

WHERE Title = "Clockwork Orange"

Figure 45: example rating query

The updated table is presented below showing “Clockwork Orange” with a five-star rating based on one evaluation:

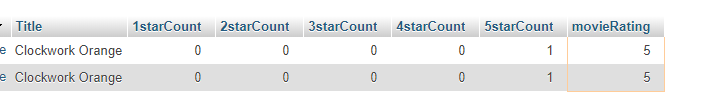


Figure 46: ratings applied to table

For testing an evaluation, the ratings table should be initialised with values. This can be completed by assigning random values to the ratings counts. The following SQL statement demonstrates the process setting a random value between 0 and 2000 for each rating count:

UPDATE `tbldatamovierating`

SET `1starCount`= FLOOR( RAND() \* 2000),

`2starCount`= FLOOR( RAND() \* 2000),

`3starCount`= FLOOR( RAND() \* 2000),

`4starCount`= FLOOR( RAND() \* 2000),

`5starCount`= FLOOR( RAND() \* 2000)

Figure 47: setting the ratings to random values

The following screen dump shows the records for “Clockwork Orange” after several runs of the random values script:

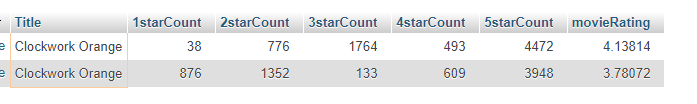


Figure 48: the ratings table with random values

A view should be created to select the top 10 rated movies for display in the chart. The view should select the Movie name and its rating selectin only the 10 highest rating movies. The following SQL can be used to create the view:

CREATE VIEW vwGetTop10Ratings AS (

SELECT Title, movieRating

FROM `vwtitlerating`

ORDER BY movieRating DESC

LIMIT 10);

Figure 49: creating the top 10 rated movie view

The records required to create the top 10 rating chart can now be retrieved through the view.

A snapshot of the output from the view is included below:

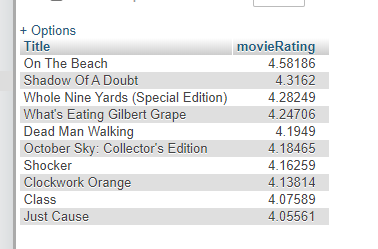


Figure 50: top 10 rated movie view

## Sprint 3 Test Plan

The testing for sprint three will follow the same standards as the previous sprints. The new requirements will have a test table to display results of all possible inputs. These results will be found in the Sprint Three Test report.

* The rating system will need to be checked to ensure the users rating correctly adds data the database.
* The chart must display the top ten movies with a correct average historic rating
* The chart must automatically refresh to display any change in data
* The website should function while logged in with an admin user and password
* Pages and functions will be tested for performance

In addition to the test table, the performance of the pages will be tested for the Optimization Report. This will analyse how quickly the page loads with a score out of 100. Also, the functions will be tested such as a search query from the movie database to record the time of execution.

|  |  |  |
| --- | --- | --- |
| **Document Name** | **Description** | **Location** |
| *Sprint03TestTable.docx* | *A series of tests carried out in a table relating the rating system.* | *Within the same folder this document is located.* |
| *Sprint03AdminLoginandUserGroups.docx* | *Some evidence and description relating to the admin and other user password access and user groups* | *Within the same folder this document is located.* |
| *Sprint03OptimizationReport.docx* | *The optimization report which carries out tests relating to loading speeds* | *Within the same folder this document is located.* |

Table 7: sprint 3 test plan

## Sprint 3 Test Table

|  |  |  |
| --- | --- | --- |
| Test | Expected Outcome | Observed Outcome |
| Clicking on a movie from the table. | Rating box displays offering the user to click from one to five stars. | As expected,  Ref: Figure 1 |
| Hover the mousse over the stars | Stars change to gold and a pop up stating how many stars will be selected | As expected,  Ref: Figure 2. |
| Submit a rating on a movie | The column in the database of the star selected should increase its count by one. The average should update | As expected,  Ref: Figure 3 & 4. |
| Press the close button on the rating box | The rating box disappears, returning the page to the previous search result | As expected,  Ref: Figure 5. |
| Connect the web application to the database with admin password | The site should function correctly with all features available | As expected,  Ref: Figure 6. |
| The top ten rating chart auto refreshes | After the average ratings are changed on a movie, the data updates in the chart automatically | As expected,  Ref: Figure 7. |
| The top ten chart for searches auto refreshes | After carrying out a search of a movie in the top ten, the chart automatically updates | As expected,  Ref: Figure 8. |

Table 8: sprint 3 test table

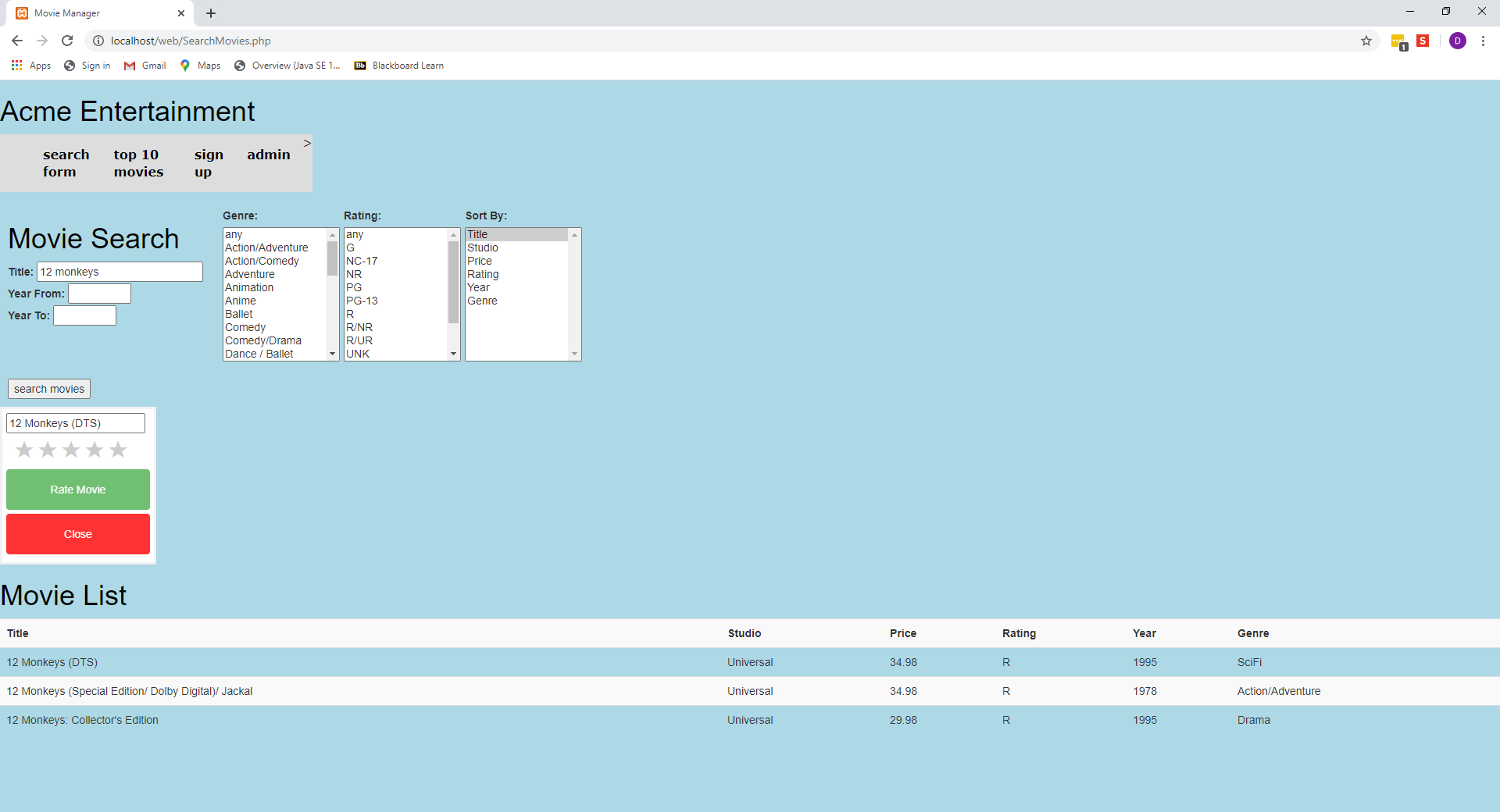


Figure 51: When clicking on a movie from the table, the rating box appears

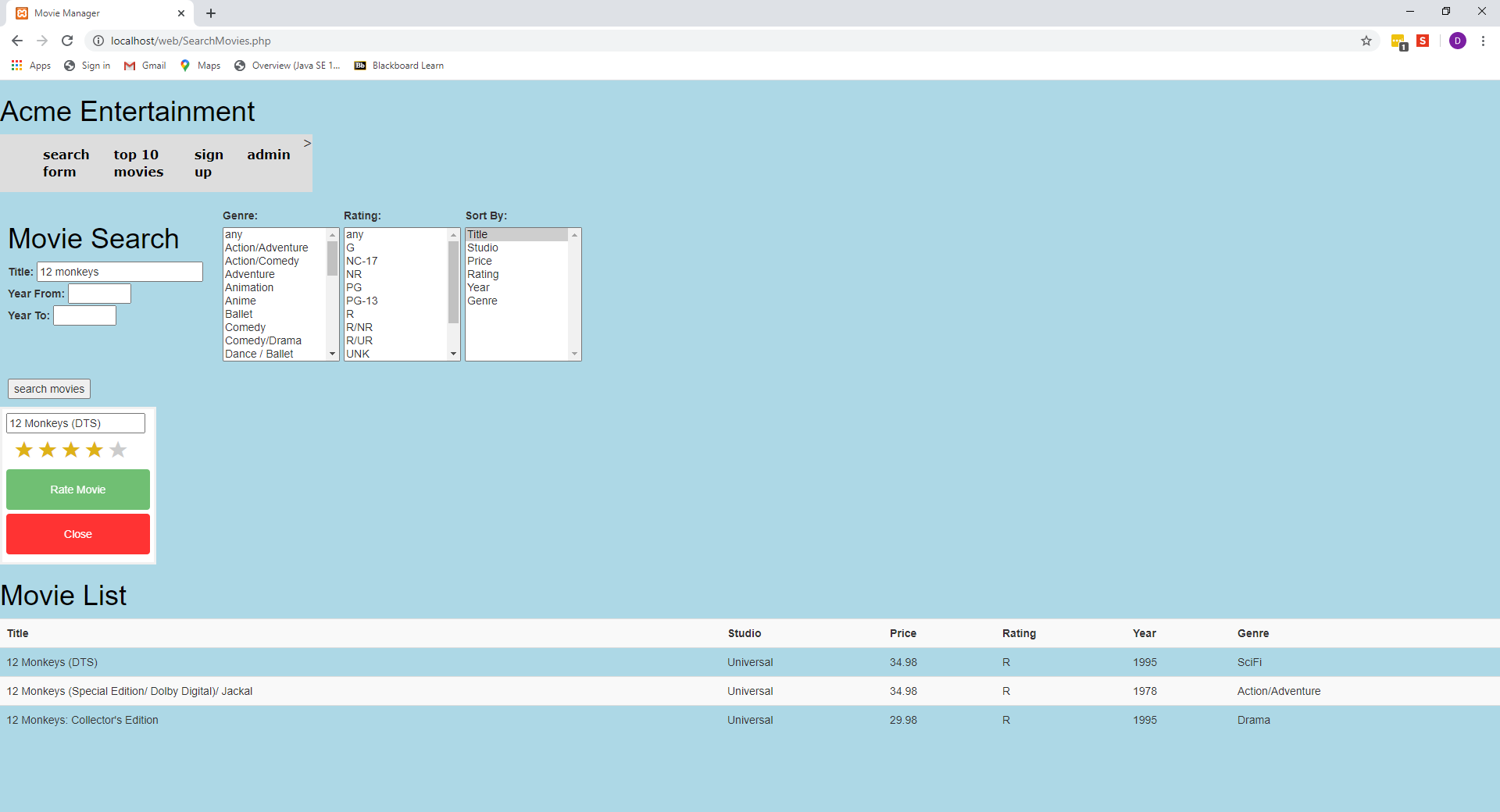


Figure 52: When hovering over the stars, they illuminate to gold to give the user better experience

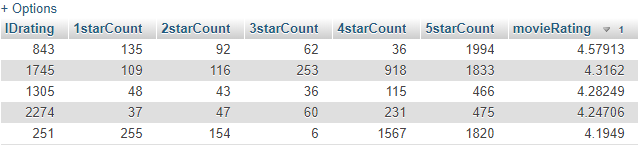


Figure 53: Snapshot of the top 5 rated movies before a rating is submitted

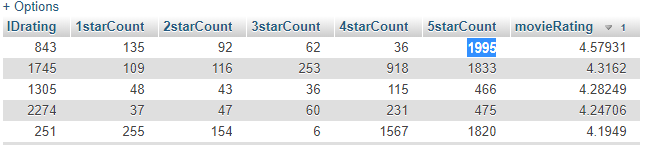


Figure 54: After a 5-star rating is added to the top movie, the count and average rating increase

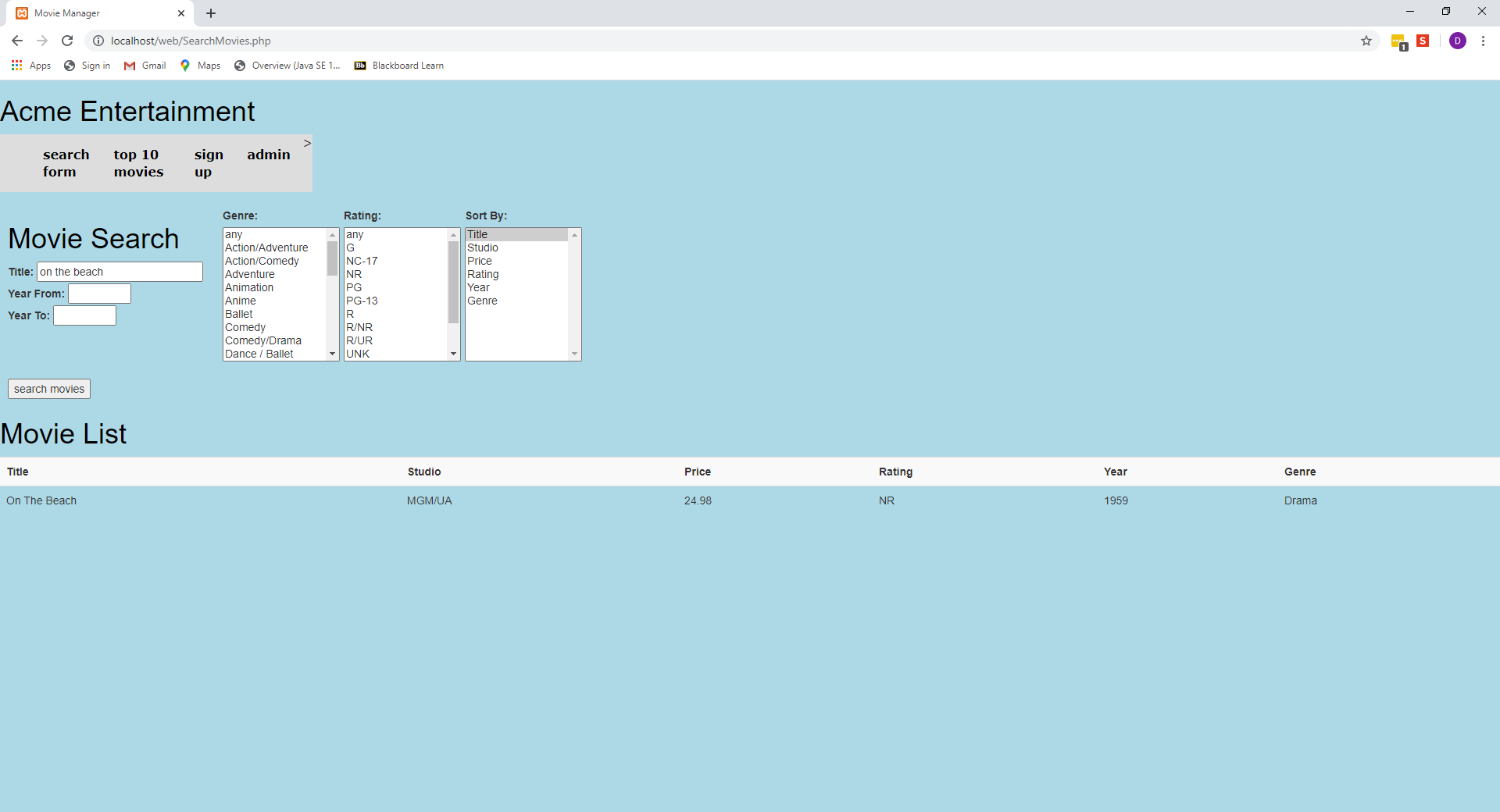


Figure 55: Page returns to search result after close button pressed in rating box

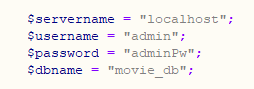


Figure 56: The admin username and password are hard coded into the connection script

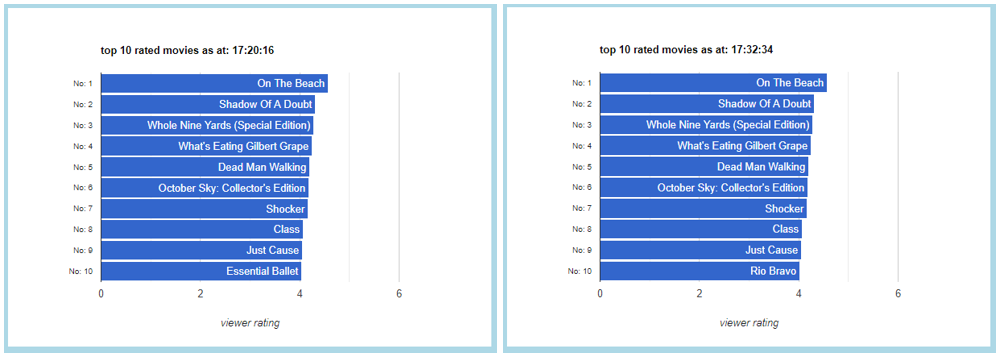


Figure 57: The chart on the right has updated the 10th movie in the list when the data has changed without the browser being refreshed

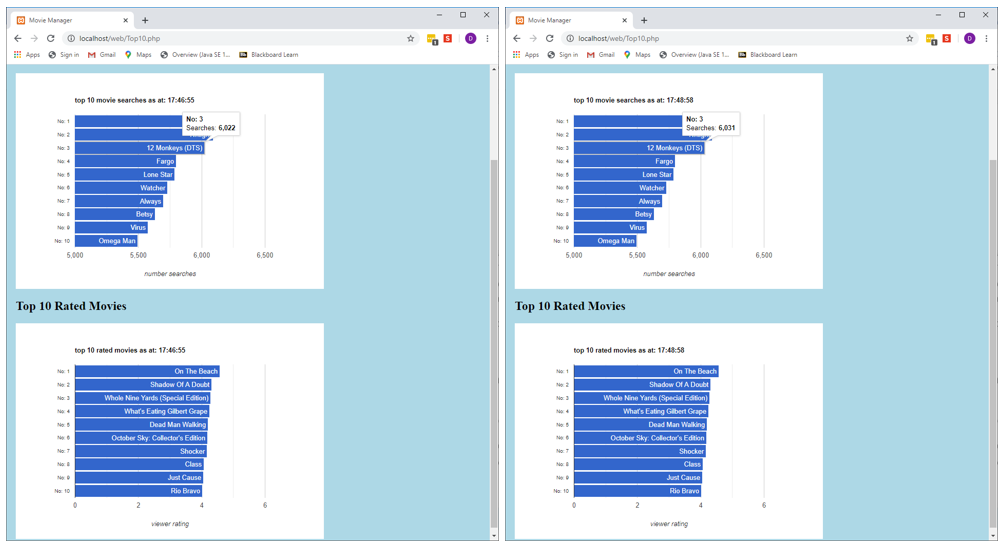


Figure 58: After running some searches on 12 Monkeys, the chart on the right has updated to display the new data without refreshing the browser

# Handover

## Source control snapshot

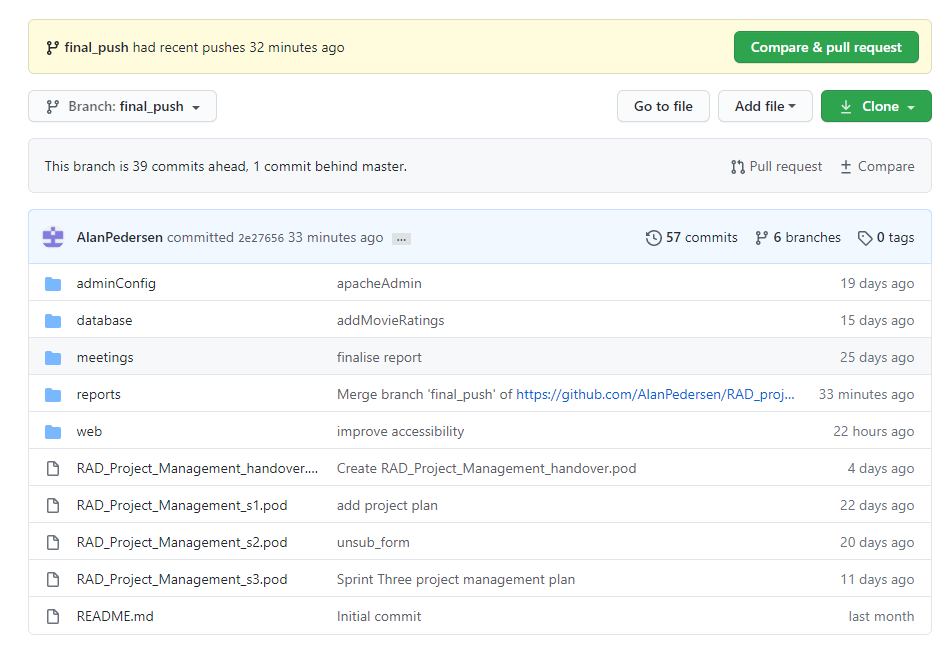


Figure 59: handover source control snapshot

## Handover Project Plan

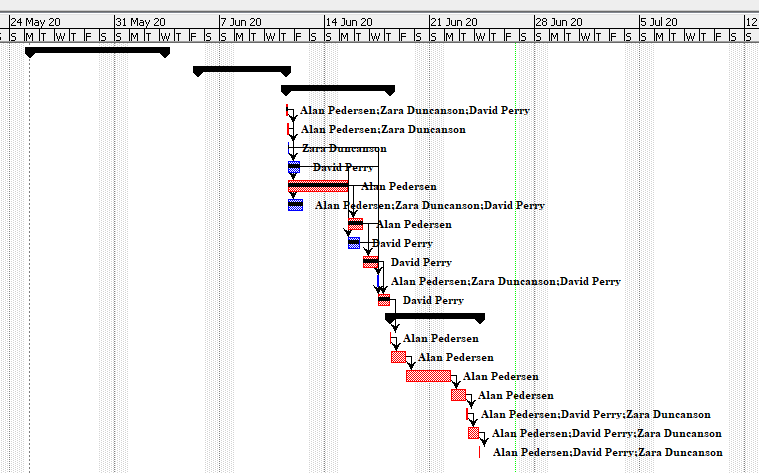


Figure 60: Gantt chart showing sprint 3 and handover phases

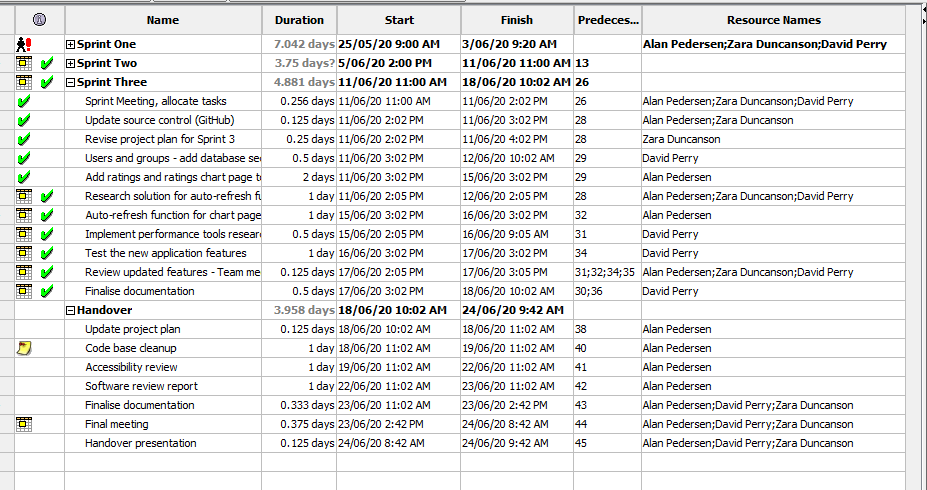


Figure 61: task list showing sprint 3 and handover phases

## Software Review Report

### Quality Assurance Development and Methodology

Throughout the development process this software has maintained quality through many different processes and approaches, from multiple team members. During Sprint One, the team used the CITE Managed Services Quality Assurance standards as a starting point for good quality assurance practice. This was summarised in the ‘Analysis Report’; building on the previously created prototype and employing these quality assurance practices as standard across each Sprint of the project development. In addition to this, testing was carried out comprehensively on all aspects and features of the application as documented in the ‘Test Plan’ and subsequent sprint testing documentation, test tables and screen captures, ensuring good UX, UI and backend functionality. This testing was also implemented with the new features required for each Sprint such as; adaptive/responsive, multi-platform, performance, user groups and administrator access, optimisation, the rating system and google charts display.

During Sprint Two, quality assurance was ensured by the team conducting a software review plan, which detailed the measures undertaken to provide excellent quality assurance including code review, team demonstrations, code testing, client feedback, and product demonstration to the client over the three sprint cycles.

Sprint three further confirmed high QA by providing an optimisation report detailing the improvement of the product using code review and optimisation tools. In addition, the PHP code sniffer tool has been used to validate all php scripts to the PEAR standard. By providing all these measures with the final product in addition to the supporting documentation, quality assurance is shown to have been maintained throughout the delivery of the movie database application.

### Testing Methodology

Each testing process uses the same blueprint to carry out testing. This process is detailed in the ‘Test Plan’ and can be summarised as follows. Testing is carried out on two levels;

* to comply with the CITE standards and to also ensure the clients requirements and requests are met with satisfaction
* testing that the application functions as required and as intended, securely and without errors

These two levels are met by carrying out regular verification on both frontend and backend code, the database, and on user experience using use cases. When bugs are detected they will be classified in order of their impact on the applications function. Bugs which negatively impact the user experience or execution of code function will be prioritised.

This blueprint is followed by the tester creating a test document consisting of a test table which carries out an example of every possible user input, action or function. This ensures complete test coverage and each use case is to be documented with screen capture evidence. Testing is considered complete when it has been carried out on each function and non-functional requirement, any identified issues (by either team member or client) have been addressed and resolved, and testing documentation has been completed in full.

### Future Support

Due to the highly documented nature of the delivery of this project, future development would be very straightforward. Any user, or any person with expertise could read through the provided documentation and development plans to ascertain an understanding of the nature of how things work. Each script, webpage, and CSS files have comment blocks which explain the functions carried out in each. The final handover project folder also includes a document called ‘RADprojectFileMap’ which describes how each page of the application runs in terms of all the different components i.e. CSS, scripting, PHP etc.

Furthermore, the source control for this project could be altered to allow access, or a version could be supplied to the client for support and backed up in case of data loss/corruption. This allows for future modifications to regress to previous sprints if there was an unforeseen change in the future that required this, or a bug that cannot be fixed.

### User Requirements Mapping

The following diagram shows how different types of users would interact with the application. Each user should be able to perform intended functions with relative ease and get the intended results, with an interface that is easy to navigate, understand and even intuitive of their needs. The requirements for staff and administrators are different; as these users provide support and facilitate the backend operations for the end users to find results by maintaining an up-to-date database, security, and subscription services. The final product delivered by the team satisfies all of these requirements while also being highly accessible, easy to understand and relevant.

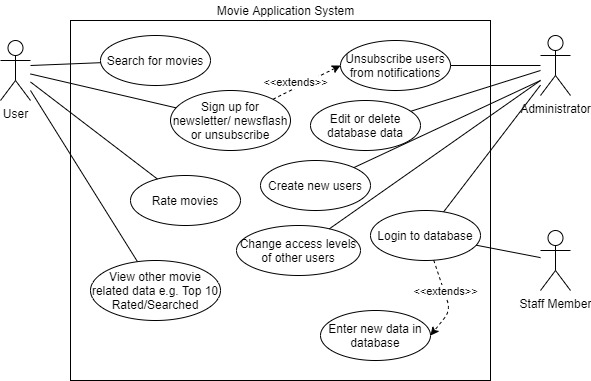


Figure 62: A Use Case diagram represents how different users interact with the Movie database System.

## Handover Test Plan

### Introduction

The final phase of the test plan will focus on reviewing the code base with a focus its long-term maintainability. The review will also assess the application against WCAG guidelines.

### Scope

The review will be restricted to a “desktop” examination of the “look and feel” of the application and an inspection of the code base.

### Quality Objective

The objective of the test plan is to ensure that the delivered application meets all the specified requirements and performs robustly.

There are two main objectives for the testing plan:

1. Compliance to specifications
   1. Compliance with CITE standards
   2. Compliance with client requirements
   3. Compliance with WCAG guidelines
2. Correct functioning of the code base
   1. Does the code perform the required operation?
   2. Does the code run robustly?

### Test Methodology

The first test will be to inspect each page within the application for a consistent “look and feel”. Each page will be inspected for consistency of font face and sizes and for consistent operation.

The code base will be inspected for duplicated elements. Where code is found to be duplicated across pages one common resource will be created and shared between the pages.

The WCAG review will focus on the semantic structure of the code. The review will examine the code to ensure it is structured to maximise the effectiveness of enhancement software.

## Review Results

### Look and Feel

Each page was examined, and the formatting and layout adjusted as required so that the appearance of the pages is consistent.

### Code Duplication

The navigation menu was moved from the individual pages and placed in a separate file that is accessed by each page. This will allow the menu to be updated by adjusting one file simplifying the maintenance requirements.

The CSS code that controls the overall presentation of the application was moved into a shared file used by all pages in the application.

The appearance of the navigation menu is controlled by CSS code, this was also moved to a common file.

The star rating system is accessed from both the search page and the details page. The common formatting elements were moved into a shared file of CSS code.

### WCAG Review

The WCAG review examined the semantic structure of the code as well as the operation of the code. Where shortcomings were encountered, they were immediately addressed.

Each form input element was checked to ensure that it had a label that was directly associated with it as well as a descriptive “title” attribute.

An example of an input element from the search form is included below:

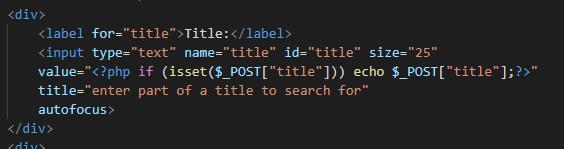


Figure 63: input element example

The operation of each for was checked for correct keyboard operation. This requires that the user is able to use the tab key to move between the input elements and that the elements are presented in the correct order. The user should be able to select from lists using the space bar and can select multiple entries where applicable using shift-arrow keys. Buttons should be operable using the Enter key.

All tables within the application were checked to ensure that the headings were correctly defined. This included using TH elements and defining the scope of the heading.

The WCAG guidelines specifies that non-text content should have equivalent text alternatives. Two of the pages within the application use charts to present information. These charts are generated dynamically and thus to not have accessible image elements within the code. Two approaches were adopted to address this.

Firstly, a dummy image element was added to the code for each chart. These elements will only appear if the scripting to generate the charts fails or is blocked. Each image element has a text alt attribute describing its contents. An aria label is also included in the HTML element that will contain the chart.

An example is included below from the top 10 page:

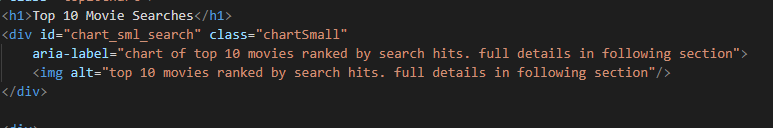


Figure 64: example of a chart presentation element

Secondly, an optional text version of the information for each chart is included. The text version is generated dynamically from the same data used to create the charts. The text is presented as a pop-up activated by a button following each chart.

An example of a chart with the activation button is included below, the activation button text turns red and is underlined when hovered over and is highlighted when tabbed to:

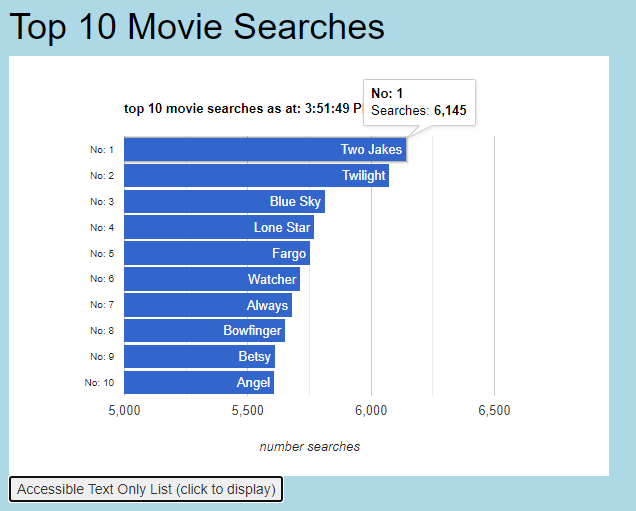


Figure 65: example chart with activation element

The text is displayed when the button is activated:

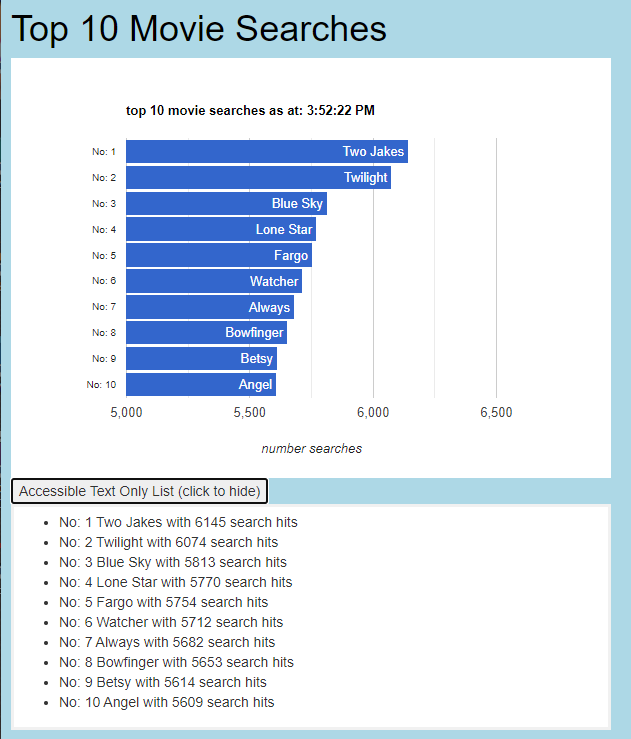


Figure 66: text equivalent data for chart

The text on the button changes to indicate its action.

## List of Page Sources

The following table details the top-level page file names and their relationship to the supporting code base.

| Top Level Files | Linked files |
| --- | --- |
| SearchMovies.php | * CSS   + base.css   + menu\_nav.css   + ratings\_formatting.css   + Bootstrap   + Embedded CSS * Scripting   + Embedded scripting * POST   + SearchMovies.php * PHP   + menu\_nav\_scr.php   + movie\_year\_limits\_scr.php     - connect.pdo.php   + movie\_list\_genre\_scr.php     - connect.pdo.php   + movie\_list\_rating\_scr.php     - connect.pdo.php   + movie\_list\_scr.php     - connect.pdo.php |
| Top10.php | * CSS   + base.css   + menu\_nav.css   + Bootstrap   + Embedded CSS * Scripting   + <https://www.gstatic.com/charts/loader.js>   + ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js   + Embedded scripting * POST * PHP   + menu\_nav\_scr.php   + movie\_top\_10\_google\_csv\_scr.php     - connect.pdo.php   + movie\_top\_rating\_google\_csv\_scr.php     - connect.pdo.php   + updateChart.php     - connect.pdo.php |
| SignUp.php | * CSS   + base.css   + menu\_nav.css   + Bootstrap   + Embedded CSS * Scripting * POST   + menu\_nav\_scr.php   + add\_new\_user\_scr.php     - connect.pdo.php   + user\_unsubscribe\_request\_scr.php     - connect.pdo.php * PHP |
| UnsubscribeUsers.php | * CSS   + ../base.css   + ../menu\_nav.css   + ../Bootstrap   + Embedded CSS * Scripting   + Embedded scripting * POST   + UnsubscribeUsers.php     - connect.pdo.php * PHP   + menu\_nav\_scr.php   + movie\_list\_usersEmails\_scr.php     - connect.pdo.php |
| MovieDetails.php | * CSS   + base.css   + menu\_nav.css   + ratings\_formatting.css   + Bootstrap   + Embedded CSS * Scripting   + <https://www.gstatic.com/charts/loader.js>   + Embedded scripting * POST   + AddMovieRating\_scr.php     - connect.pdo.php * PHP   + menu\_nav\_scr.php   + movie\_1\_rating\_google\_data\_scr.php     - connect.pdo.php   + movie\_1\_details\_google\_data\_scr.php     - connect.pdo.php   + movie\_1\_rating\_list\_data\_scr.php     - connect.pdo.php   + AddMovieRating\_scr.php     - connect.pdo.php |

Table 9: top level and supporting files

# References

Merlin, G. (2018, November 15). *Graphics Studio*. Retrieved from Responsive Vs Adaptive Website Designing in 2019: https://www.graphicsmerlin.com/responsive-vs-adaptive-web-design-2019/