Movie Database – Master Document

Team Orange – Movie database project

Benjamin Wearn, Chris Chi, Jaron Rose

2021

Table of Contents

[Sprint One – Ben W 2](#_Toc89535030)

[Software development testing plan 2](#_Toc89535031)

[Analysis report 5](#_Toc89535032)

[Acme Entertainment Pty Ltd development requirements 5](#_Toc89535033)

[Purpose: 5](#_Toc89535034)

[Problem Statement: 5](#_Toc89535035)

[Scope and Objective: 5](#_Toc89535036)

[Target System: 5](#_Toc89535037)

[Project Requirements: 5](#_Toc89535038)

[CITE Business Rules for Software Development 5](#_Toc89535039)

[User Interface Development Guidelines: 5](#_Toc89535040)

[User Control and Freedom: 5](#_Toc89535041)

[Consistency and Standards: 5](#_Toc89535042)

[Error Prevention: 5](#_Toc89535043)

[Aesthetics: 5](#_Toc89535044)

[CITE Development Methodology: 6](#_Toc89535045)

[CITE Managed Services Quality Assurance: 6](#_Toc89535046)

[Quality Planning: 6](#_Toc89535047)

[Quality Assurance: 7](#_Toc89535048)

[Quality Control: 7](#_Toc89535049)

[Quality Assurance Department: 7](#_Toc89535050)

[Quality Assurance in Development Lifecycle: 7](#_Toc89535051)

[Multi-platform report 8](#_Toc89535052)

[Introduction 8](#_Toc89535053)

[Adaptive 8](#_Toc89535054)

[Responsive 8](#_Toc89535055)

[Side by side comparison 8](#_Toc89535056)

[Decision: Adaptive or Responsive 8](#_Toc89535057)

[Source Control 9](#_Toc89535058)

[Project management 9](#_Toc89535059)

[Code Validation – Sprint One 11](#_Toc89535060)

[Testing Table 13](#_Toc89535061)

[Sprint Two – Chris C 18](#_Toc89535062)

[Software Review Plan 18](#_Toc89535063)

[Client Requirements: 18](#_Toc89535064)

[Quality Assurance: 18](#_Toc89535065)

[User Interface Plan: 18](#_Toc89535066)

[Performance Report 19](#_Toc89535067)

[Code Optimisation: 19](#_Toc89535068)

[Performance Tools: 19](#_Toc89535069)

[Source Control – Sprint Two 21](#_Toc89535070)

[Project Management – Sprint Two 22](#_Toc89535071)

[Code Validation – Sprint Two 23](#_Toc89535072)

[Testing Table 28](#_Toc89535073)

[Sprint Three – Jaron R 32](#_Toc89535074)

[Optimisation Report: 32](#_Toc89535075)

[Installation: 32](#_Toc89535076)

[Optimiser: 32](#_Toc89535077)

[Source Control – Sprint Three 35](#_Toc89535078)

[Project Management – Sprint Three 36](#_Toc89535079)

[Code Validation – Sprint Three: 37](#_Toc89535081)

[Testing Table: 46](#_Toc89535082)

# Sprint One – Ben W

## Software development testing plan

Testing for this project will take on a variety of forms, a subset of these will be undertaken throughout development, however all of them will be able to be performed on project completion to determine the overall state of the completed project. Some of the aspects of testing to be undertaken are: -

**Functionality testing** – this kind of testing determines whether or not the website actually achieves what it sets out to do, questions to be asked are such like, do the all the links function as expected? Do all the pages show the required content? This type of testing will also include Forms testing to ensure all the required forms function as expected with required inputs or inputs as required and to ensure that no erroneous data can be entered into forms as much as possible. Included also is navigation testing to ensure that all navigation tools function as expected and help the user around in a clear and user-friendly manner.

**Validation testing** – this kind of testing is to determine whether all the different types of code are free of errors and will function as expected on various browsers, this can be achieved through validate of all HTML, CSS and PHP code, being run through a code validation testing environment.

**Portability testing** – this kind of testing determines how the website looks and/or runs on various browsers and across various platforms (such as different operating systems), it is not always possible to test every browser on every platform, however, ensuring that the website functions as expected across as many environments as possible will minimise any protentional impacts upon potential users.

**Printing compatibility** – this kind of testing in an age of green initiatives could be considered a waste of time as people are attempting to minimise the amount of printing they are doing, however, ensuring your pages print correctly and look nice when printed, will minimise environmental impacts if users are to actually print your page. This means you could be saving excess paper from being wasted as the layout of your page prints poorly.

**Performance testing** – this kind of testing is important in an age where information is served at such high speed. Ensuring your platform operates as quickly as possible will minimise the amount of traffic your platform may lose due to slow loading pages, which prompt potential users to click away and find somewhere else to be served the content they were hoping to find on your page.

**Content testing** – it Is important to test your content for a couple of reasons, firstly, it is important to include key information, without overwhelming users with information, clearly labelled headings, navigation and sections, to help them find the data they are looking for without having to read too much, as this will deter some users from browsing your page. It is important to clearly help users as the primary method of web content browsing is typically using ‘scanning’ techniques commonly referred to as the ‘F Pattern’, if your content doesn’t help those users find the content they are looking for, they may get frustrated, give up and try another platform.

**Security testing** – security testing is important to carry out to ensure the safety of your users and your databases, content, etc. by ensuring that none of the systems or users can be targeted with any known flaws will mean a good reputation and encourage people to use your platform.

**User friendliness testing** – when designing a web platform, it is very important to keep the end-user in mind, these are the people who you are developing the platform for. If the platform is not easy to use, navigate and browse your content, they will simply find somewhere else to get that content.

**Design testing** – design is something to keep in mind throughout development, the overall appearance, layout and content of your platform will either encourage or discourage users from using your platform, and with a large amount of content available on the web, you want to stand out from the competition and have users, using your platform over others. The things to keep in mind have been somewhat stated already in previous sections, however, ensuring that your content looks nice, is laid out with appropriate headings and navigation to help your users around and doesn’t feel clunky or slow, will help to ensure the continued success of your platform.

## Analysis report

### Acme Entertainment Pty Ltd development requirements

### Purpose:

The purpose of the movie rental database development, is for Acme Entertainment Pty Ltd. In which they require an update to the current prototype of their movie database. This database is to be reviewed and updated so it can be used across all the major digital platforms.

### Problem Statement:

Currently the system prototype does not allow multi-platform access, which limits users to be most compatible on a computer system, as other platforms are not optimized and tested for, which may result in other platforms having issues when accessing the system on a different platform. Although Acme would like the system to be updated in order to improve such accessibility to allow other platforms to be optimized.

### Scope and Objective:

The scope of the project is to update the current system that Acme has provided in order to allow multi-platform access from at least three other platforms.

### Target System:

The target system is to be changed to allow users to access the database on any system through integrating multi-platform solutions.

### Project Requirements:

* Develop the application with PHP
* generate SQL scripts to store the movie records in an appropriate database table
* Create a service to access the movies database stored in the server
* Multi-platform access

### CITE Business Rules for Software Development

### User Interface Development Guidelines:

### User Control and Freedom:

Allow users the control and freedom with optimal options and digital space such as backward steps, undoing and redoing previous actions.

### Consistency and Standards:

Ensure that graphic elements are maintained across similar platforms.

### Error Prevention:

Implement error catching so that potential errors are kept to a minimum.

### Aesthetics:

Remove unnecessary clutter and information that may limit user’s attentional resources.

### CITE Development Methodology:

CITE uses an agile method framework, which maintains a customer focus. This allows teams to deliver the highest priority features first for clients. Communicate accurately and unambiguously across imposed barriers such as, distance, time, culture, language. Sprint reviews ensure that the delivered features are reviewed and that feedback is shared across the team. Continuous integration allows for continual improvements on development and for problem resolutions as issues arise. CITE uses a three-sprint model:

A picture containing chart

Description automatically generated

Each sprint goes through six phases which are, planning, design, build, test, review and launch.

Planning allows the team to have clear direction, goals and understanding of requirements before, during and at the ending of development.

Design will be continuously improved on at each interval as after each interval, the design will be updated to better fit client needs and requirements if they are not met previously.

Building the project will allow teams the opportunity to have visual representation and/or to develop the actual system to be implemented. At each interval the build may be changed as errors or issues may arise.

Testing will begin once the initial build is completed to determine whether the system works as intended.

The team will then review the project before going into the next sprint interval where the six phases are then revisited and improved upon the previous until the project is finished.

### CITE Managed Services Quality Assurance:

CITE implements a Quality Management System (QMS) that ensures quality of software throughout every development. The tasks and objectives of this QMS are:

Elaboration and implementation of procedures and regulations for software development based on industry standards and practices. Product lifecycle monitoring which ensures compliance with all processes and guidelines at CITE. Product quality verification and validation to ensure needs and expectations of all clients are met. Establishment of effective collaboration within all project teams.

### Quality Planning:

CITE creates quality plans that govern the applicable set of standards, regulations, procedures, guidelines and during development of each project.

### Quality Assurance:

CITE has established processes that evaluate project performance and aim to assure that quality standards are being followed and that software comply with all requirements.

### Quality Control:

Performance trends are measured to identify defective code, verify deliverables are of high quality and are working correctly and as intended.

### Quality Assurance Department:

CITE has an independent subdivision for quality assurance. This subdivision is responsible for:

* Full-cycle testing
* Document and Code review
* Defect tracking
* Configuration Management
* Process Monitoring
* Risk Management

### Quality Assurance in Development Lifecycle:

CITE performs quality assurance (QA) throughout the entire software development lifecycle. This QA lifecycle includes 4 phases.

#### Initiation and Planning:

Project specification analysis, test plan elaboration and team assignment.

#### First Review:

Initial testing of first development deliverables, refining test plan and test items.

#### Iteration Audits:

Ongoing testing of intermediate iterations builds.

#### Final Verification and Validation:

Final product testing to ensure best possible quality and readiness for deployment.

#### Test Types:

CITE utilizes these test types on all projects to ensure quality assurance:

* Functional and Regression Testing
* GUI and Usability Testing
* Compatibility Testing
* Performance Testing
* Installation / Configuration Testing
* System/ Integration Testing
* Security Testing
* Internationalisation / Localisation Testing
* User Acceptance Testing

## Multi-platform report

### Introduction

This report has the purpose of determining which web development technology, adaptive or responsive, to be used for Team Orange’s project. To deliver a little background information of the web program, it is a website that’s connected to a database which has the purpose of user input searching for movie records as well as displaying top 10 searched movies. In brief, this report will provide the necessary information and point out which will be used.

### Adaptive

Adaptive web design practice requires the developers to create multiple versions of webpages that better fit on every different devices, which means the developers need to create one .css file for each device.

### Responsive

Responsive web design practice delivers content to the users by auto-adopting the screen sizes of the devices, and provide the best user experience by minimizing the resizing, panning and scrolling with the use of fluid grid layout.

### Side by side comparison

|  |  |
| --- | --- |
| Responsive | Adaptive |
| Same layout for all screen size | Different layouts according to the device and OS |
| One .css file fits all | Multiple version .css files for each different size |
| Hard to build as making one design for multiple devices could be complex | Easier to build as design layouts according to device |
| Works on any screen size | Work on Single screen size for each .css |
| Loads faster, only load a single layout | Loads slower, load entire possible layouts |

### Decision: Adaptive or Responsive

Due to numbers of reasons, **Responsive web design practice** will be used for the project.

1. **Responsive** requires to only create one .css file for all devices which **reduces the amount of budget** required comparing to **adaptive**, as it requires developers to create multiple .css files which **increases the amount of time and money required**.
2. Responsive method allows the web page to load quickly, increasing client satisfaction, whereas adaptive method loads slower.
3. Responsive practice fluidly fits all screen sizes whereas Adaptive practice aims to fit only one size.

## Source Control

A screenshot of a computer

Description automatically generated with medium confidence

## Project management

Project Start:

A computer screen capture

Description automatically generated with medium confidence

Project underway

A screenshot of a computer

Description automatically generated with medium confidence

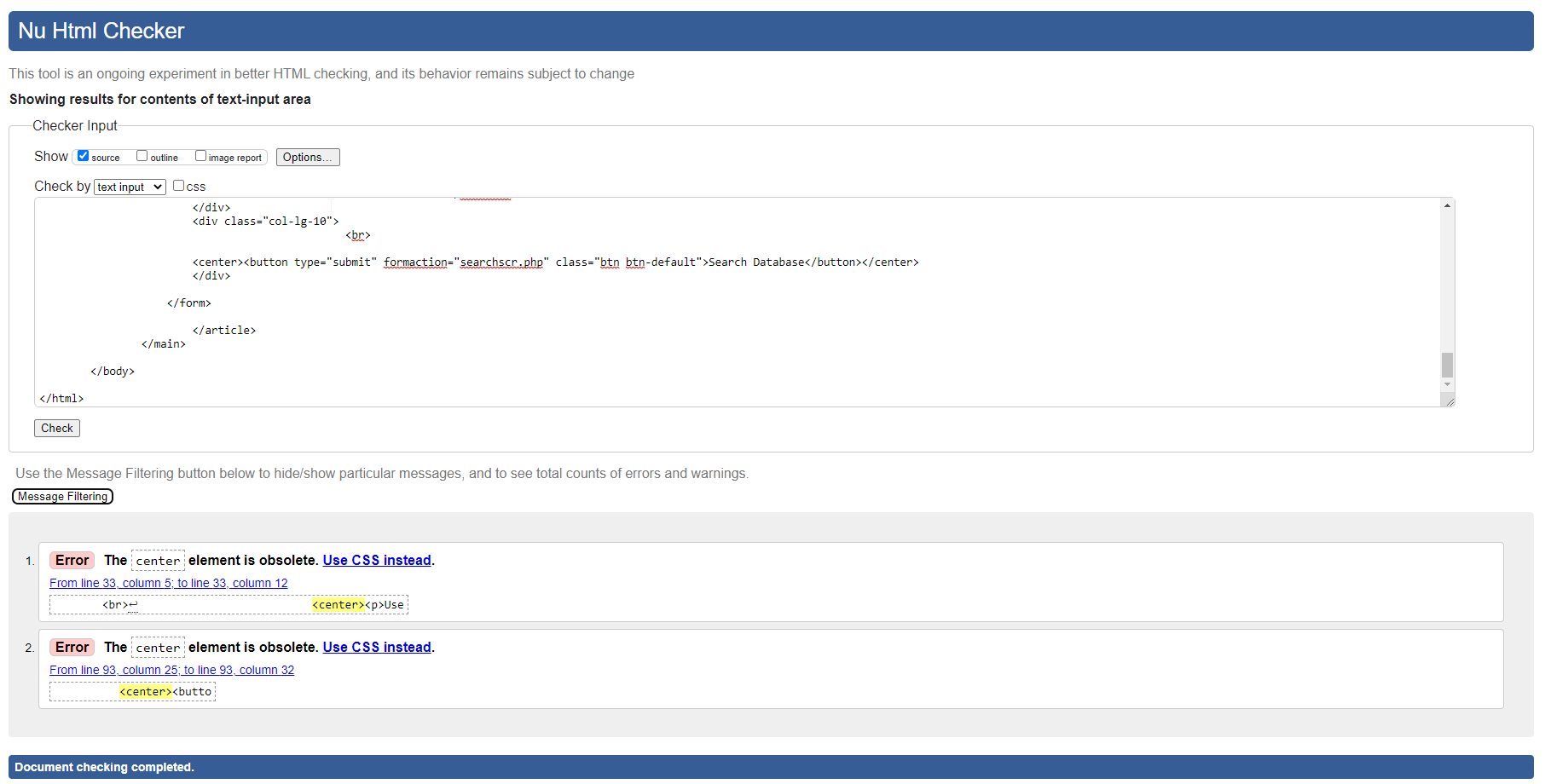
Project completion

A screenshot of a computer

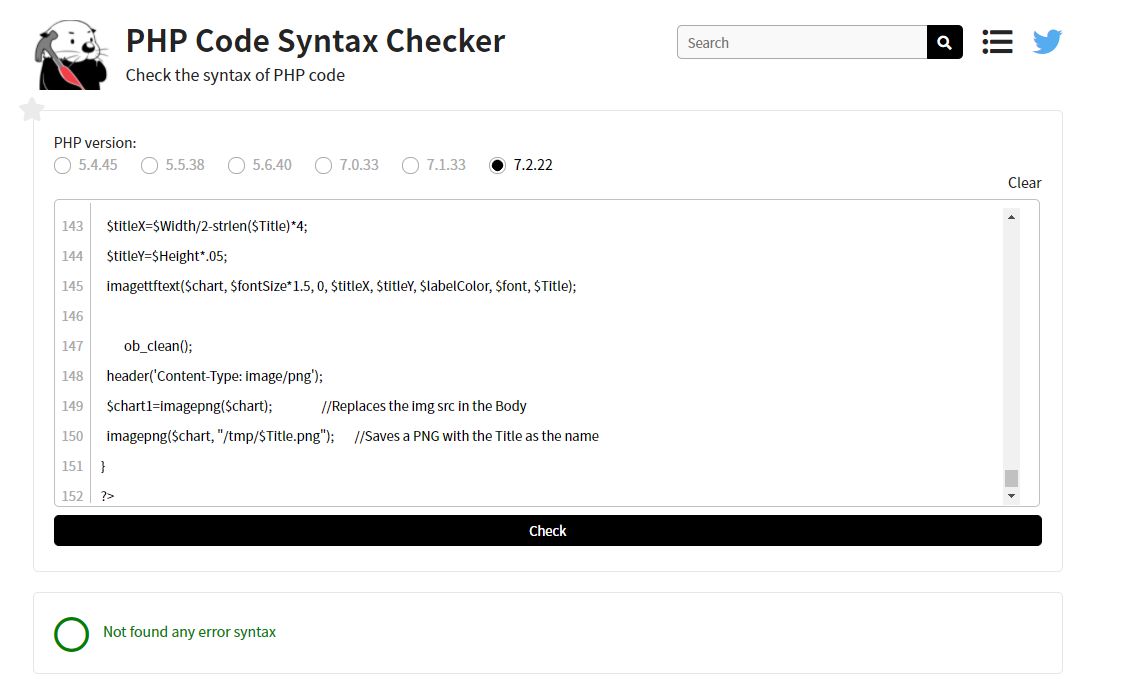
Description automatically generated

## Code Validation – Sprint One

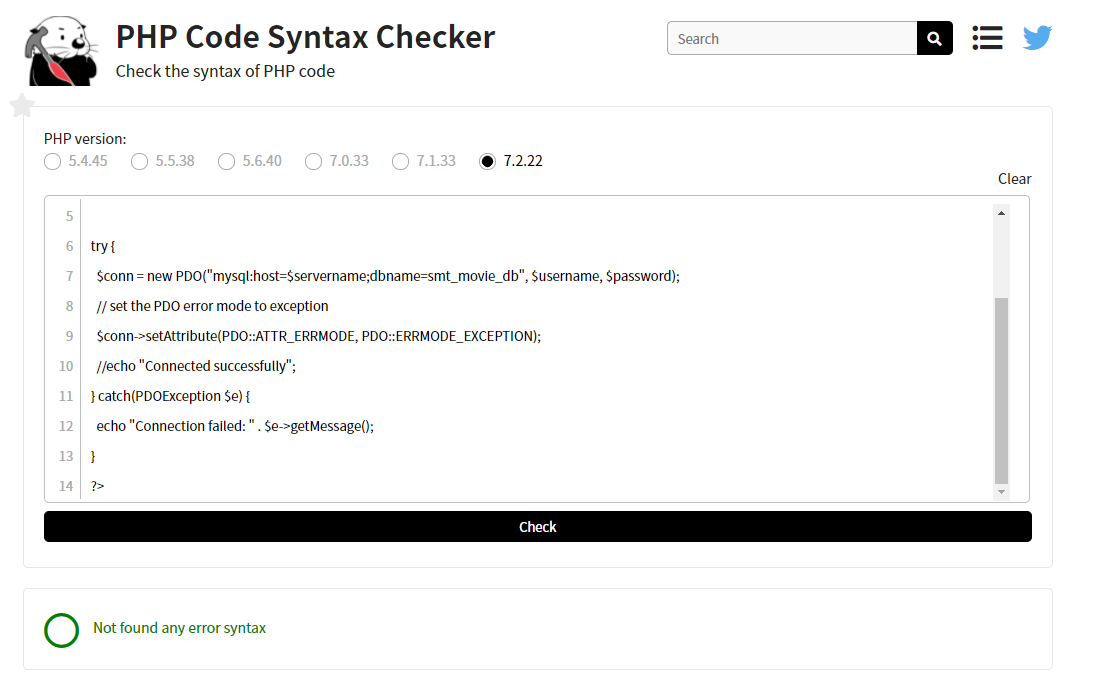
index.php



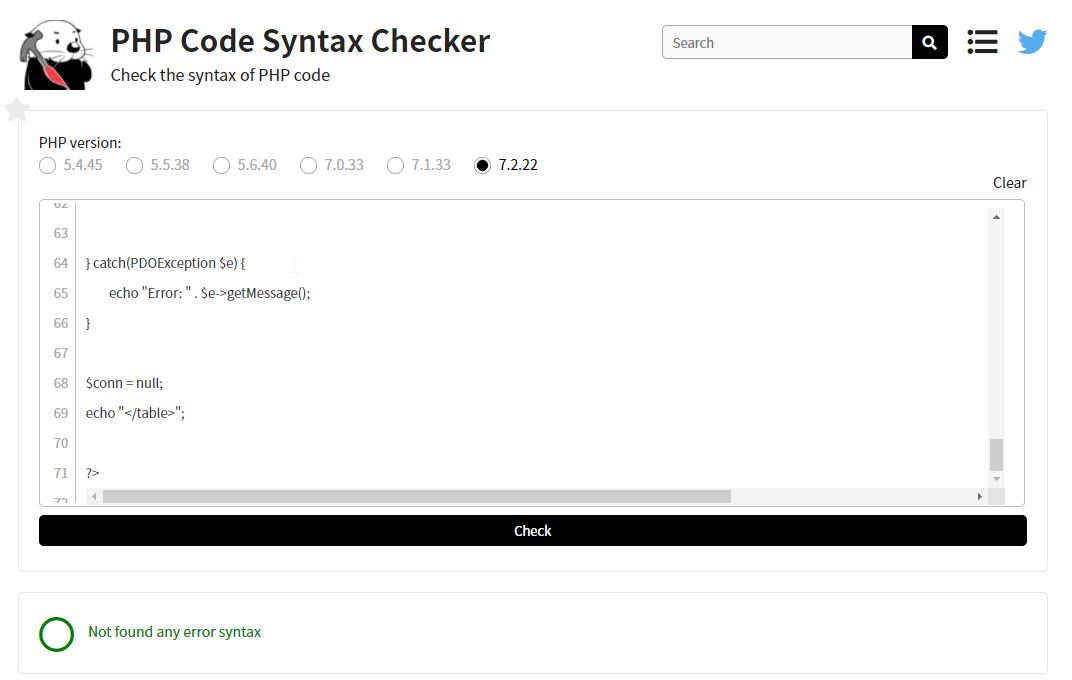
bargraph.php



connect.php

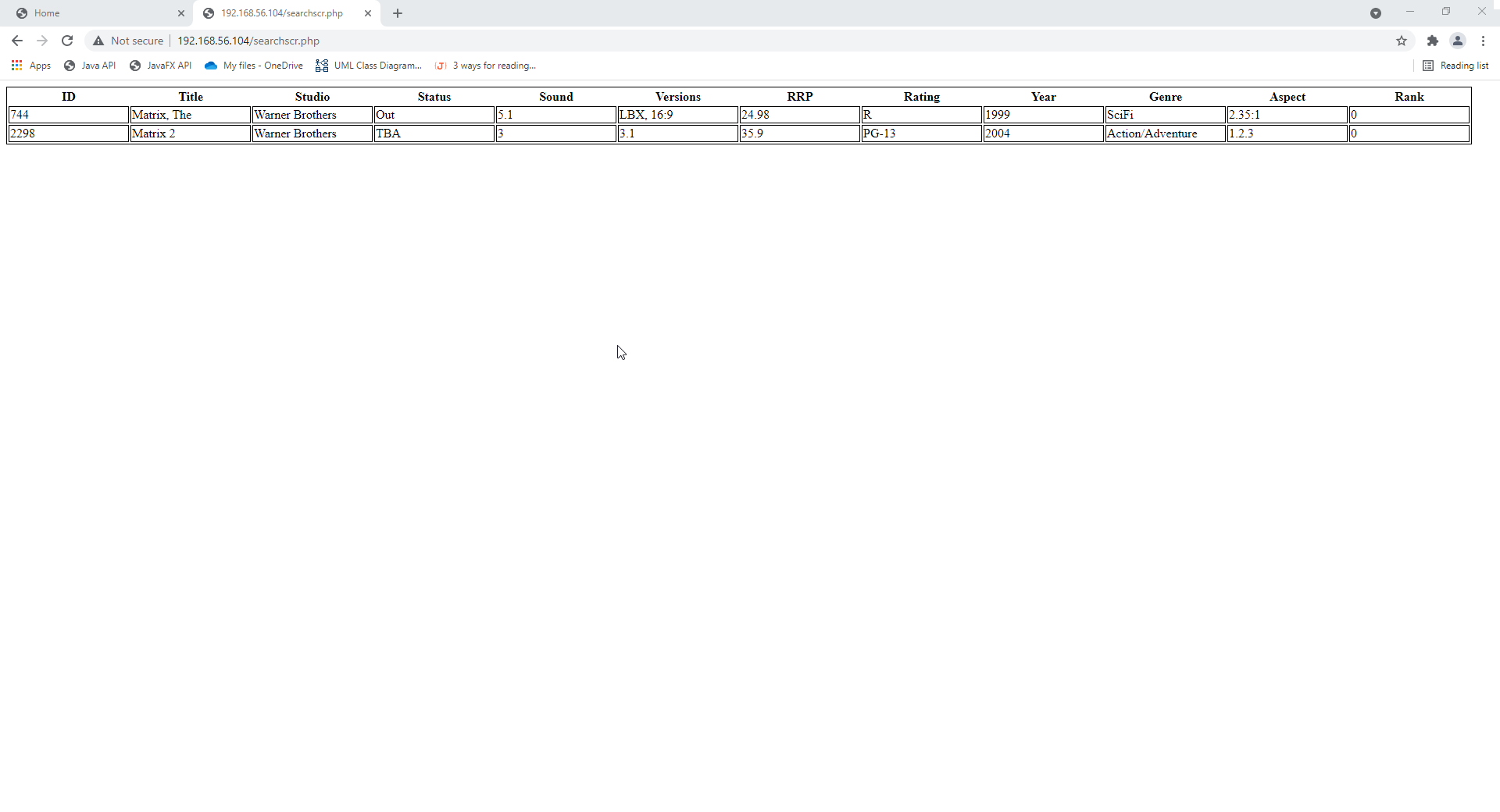


searchscr.php

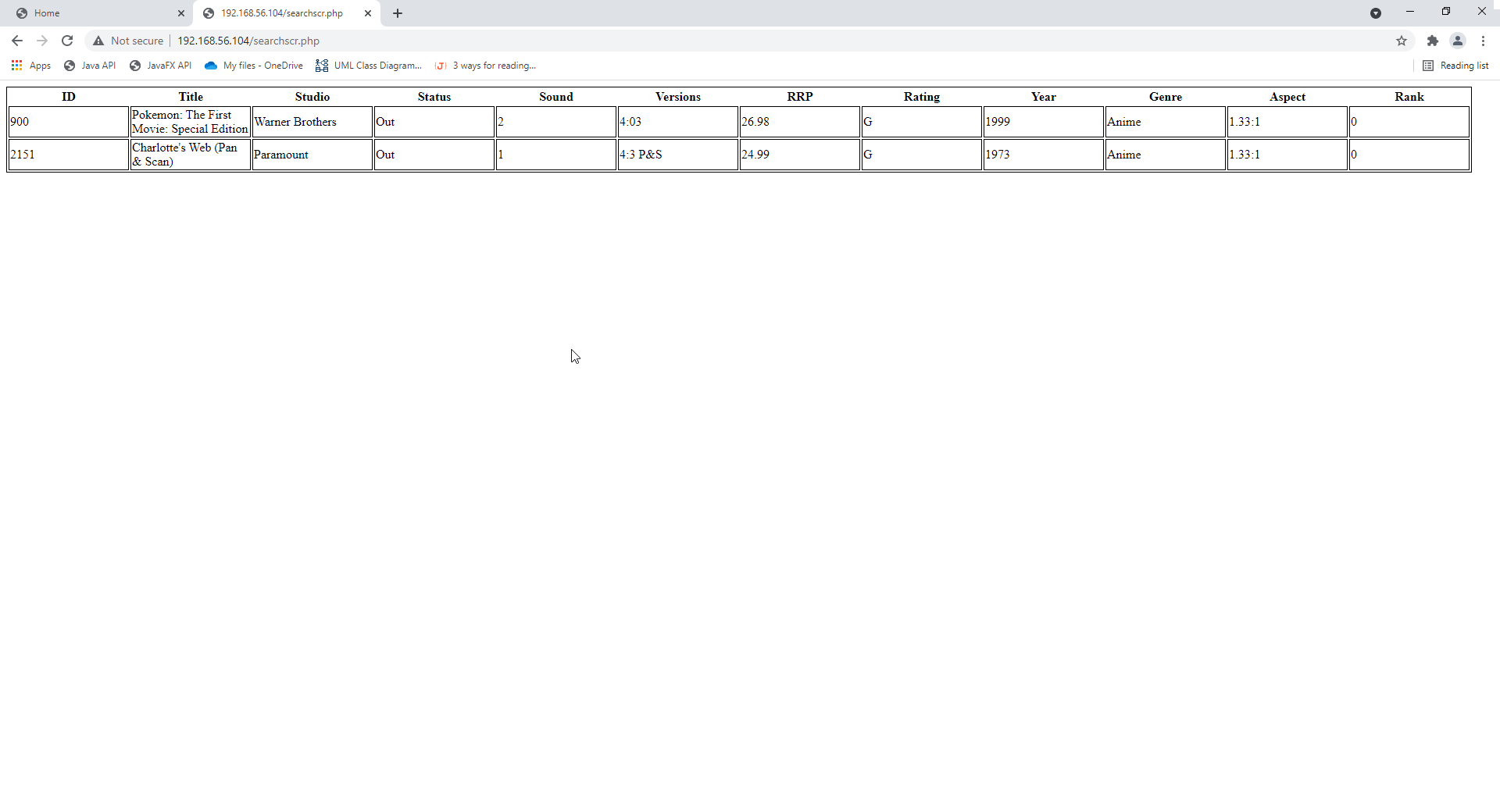


## Testing Table

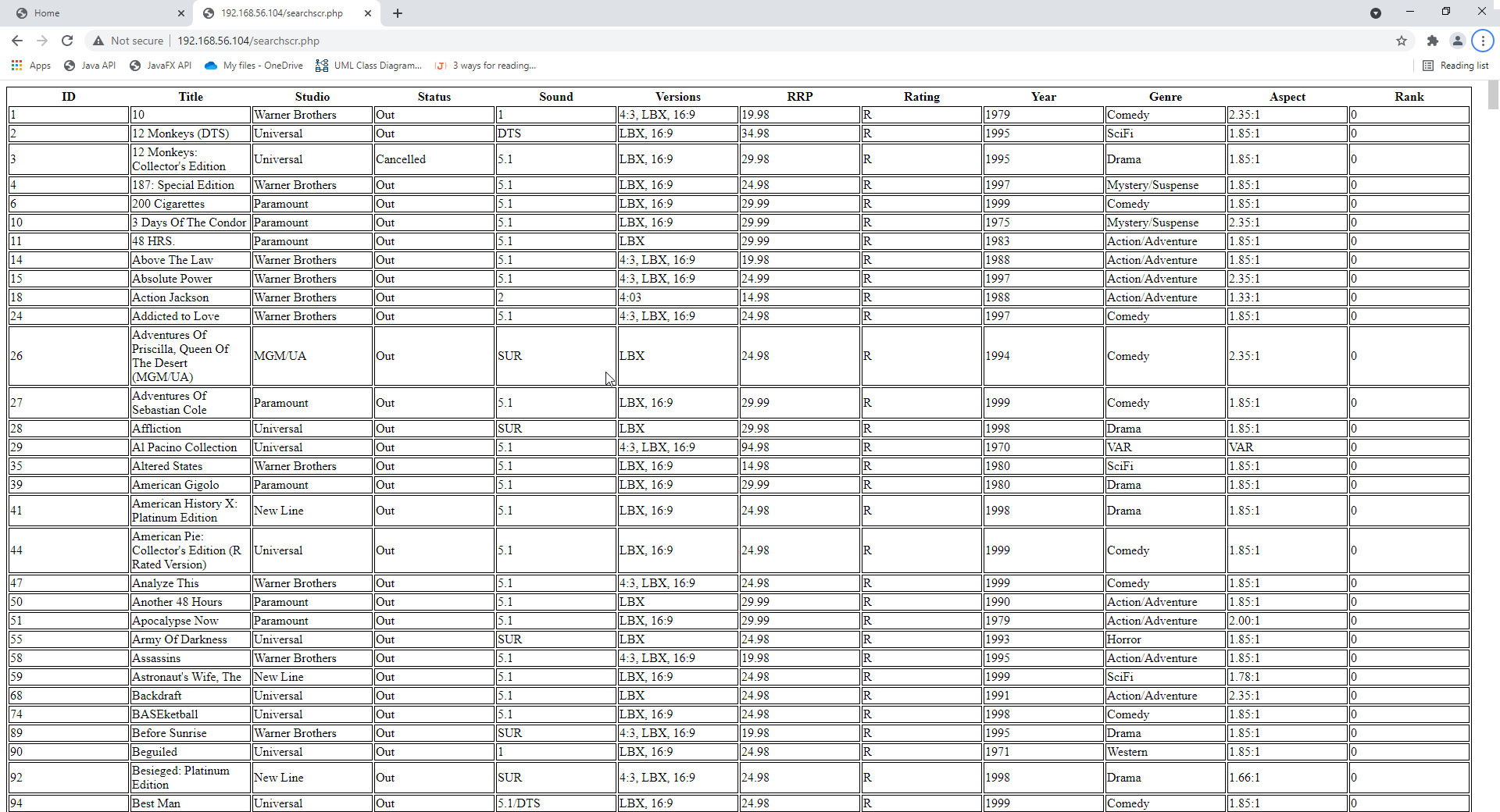
|  |  |  |  |
| --- | --- | --- | --- |
| **Test** | **Expected Output** | **Actual** | **Reference** |
| Search singular movie name | Searches and displays movies by title | As expected | Refer Figure 1 |
| Search genre | Searches and displays all movies by specific genre | As expected | Refer Figure 2 |
| Search rating | Searches and displays all movies by specific rating | As expected | Refer Figure 3 |
| Search year | Searches and displays all movies by specific year | As expected | Refer Figure 4 |
| Search genre and rating | Searches and displays all movies with specific genre and rating | As expected | Refer Figure 5 |
| Search genre and year | Searches and displays all movies with specific genre and year | As expected | Refer Figure 6 |
| Search genre, year and rating | Searches and displays all movies with specific genre, year and rating | As expected | Refer Figure 7 |
| Search year and rating | Searches and displays all movies with specific year and rating | As expected | Refer Figure 8 |
| Top 10 link | Shows bar graph containing top 10 movies | As expected | Refer Figure 9 |
| Resize page | Conforms to browser size | As expected | Refer Figure 10 |
| Run on desktop platform | Displays nicely and correct form factor | As expected | Refer Figure 11 |
| Run on tablet from factor | Displays nicely and correct form factor | As expected | Refer Figure 12 |
| Run on phone form factor | Displays nicely and correct form factor | As expected | Refer Figure 13 |



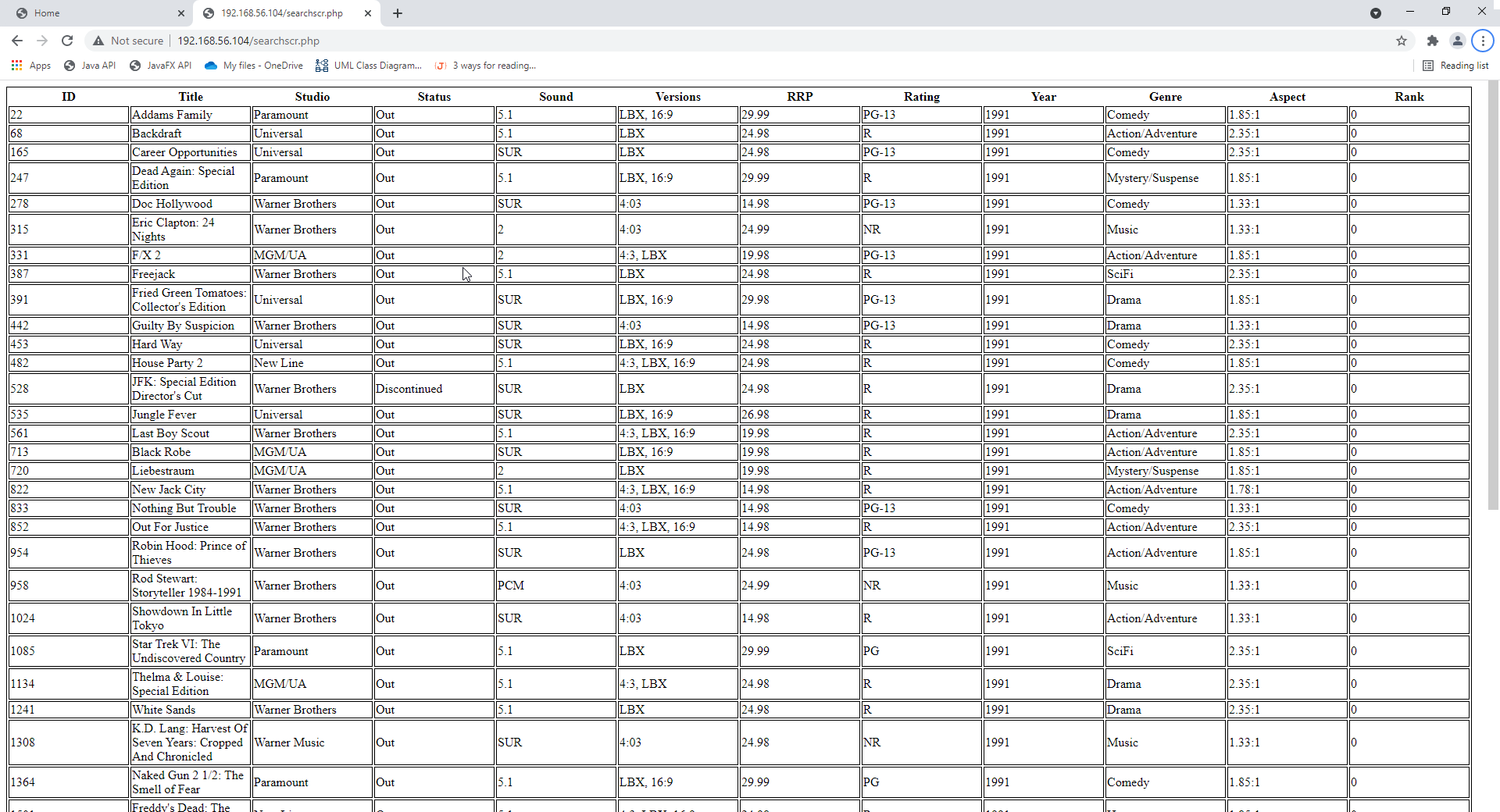
Figure



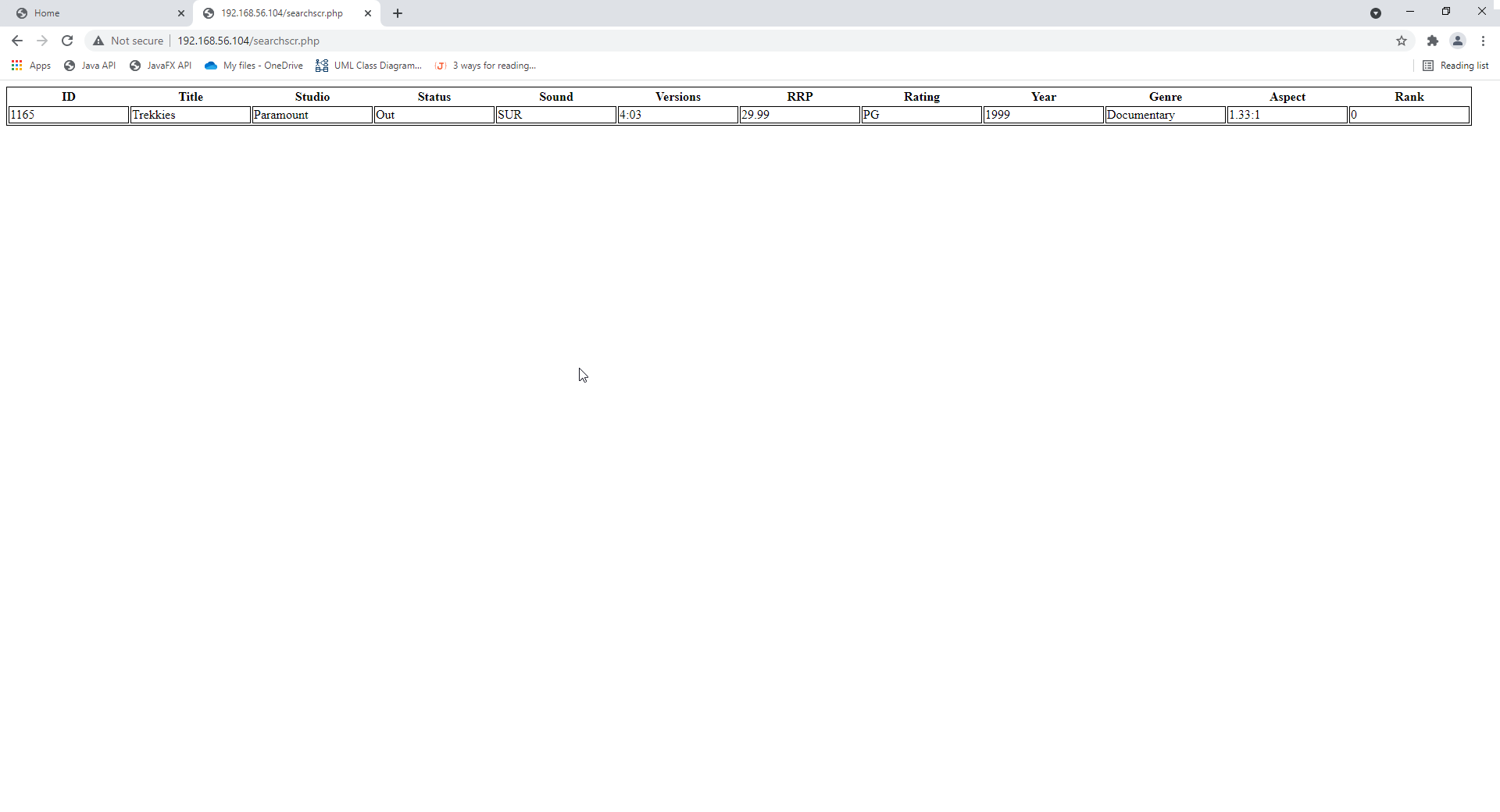
Figure



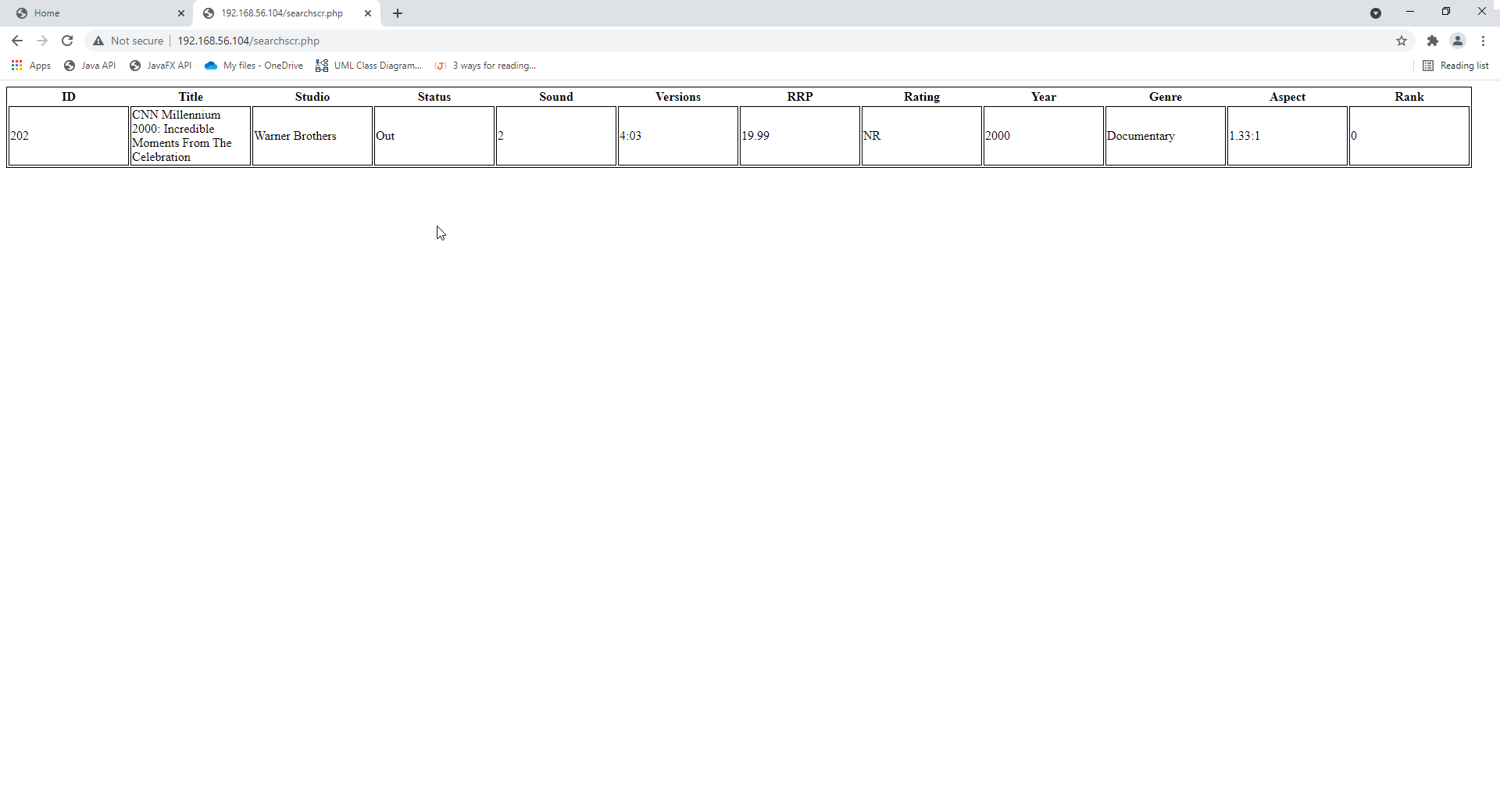
Figure



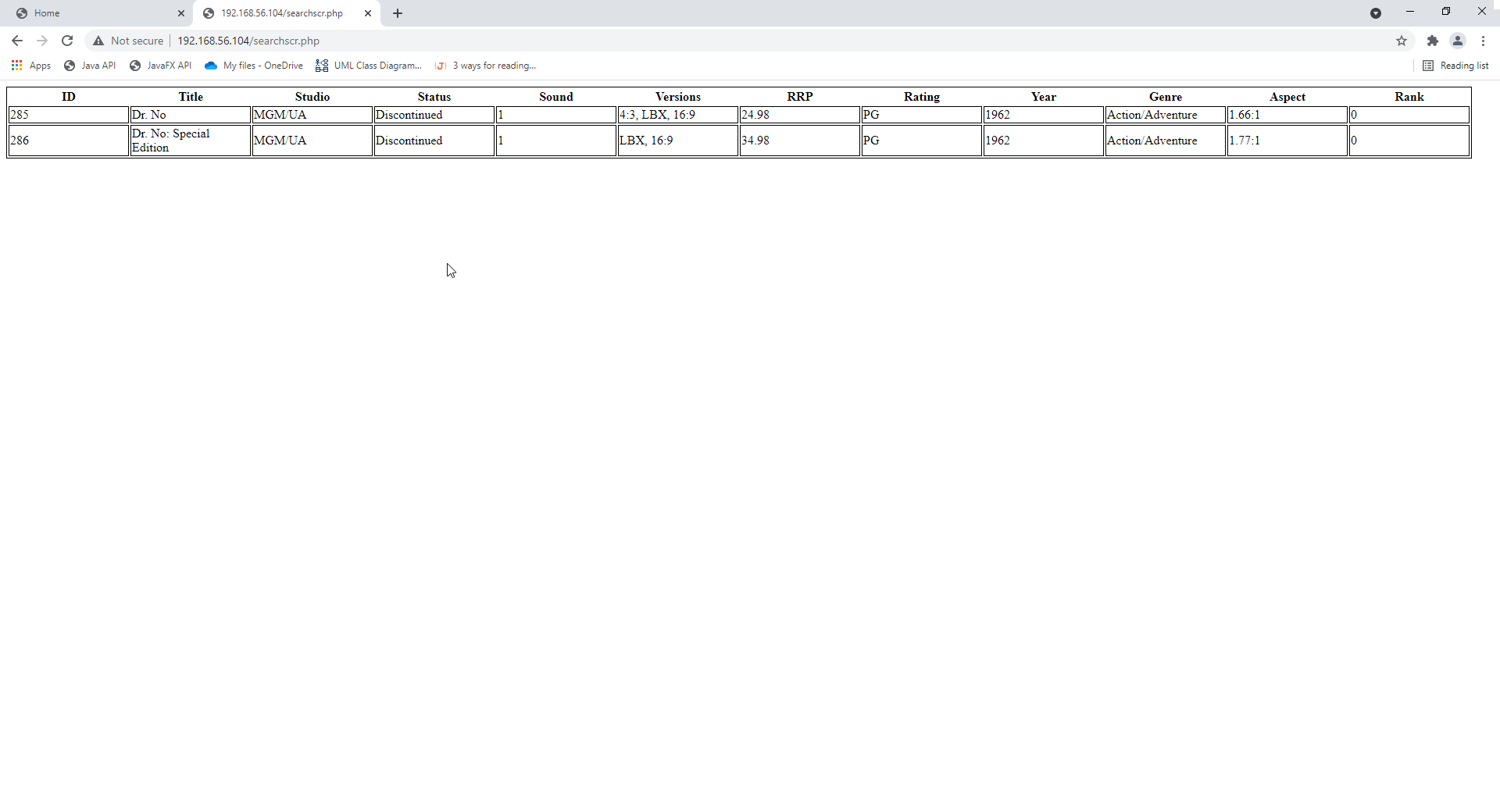
Figure



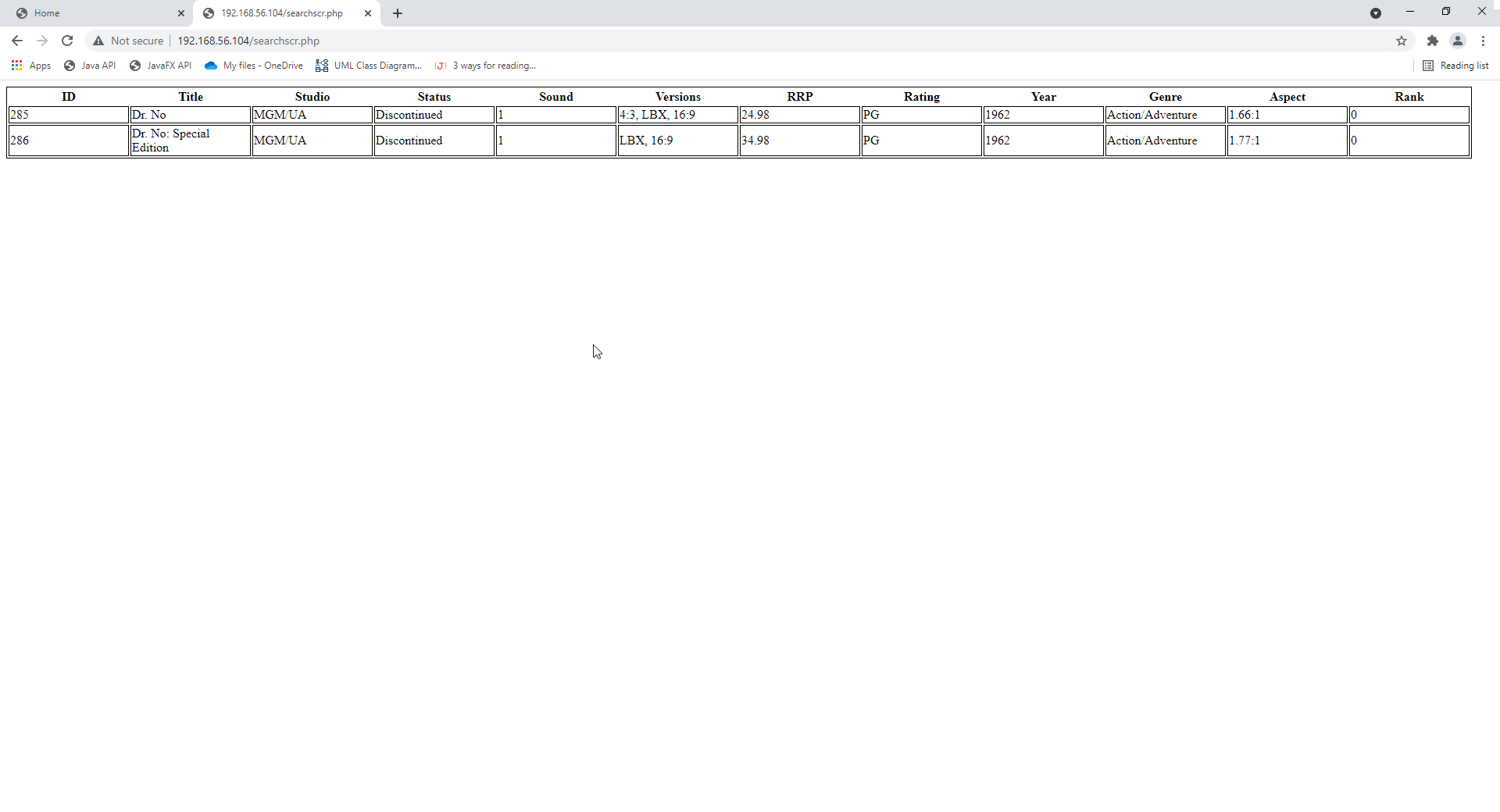
Figure



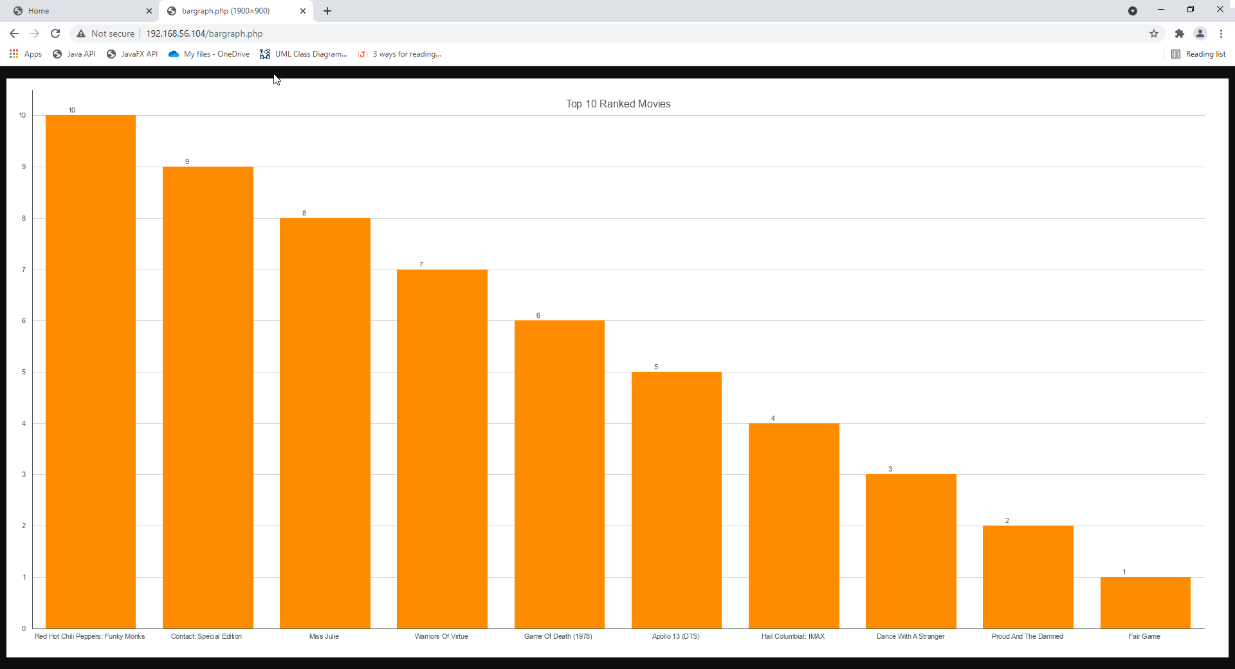
Figure



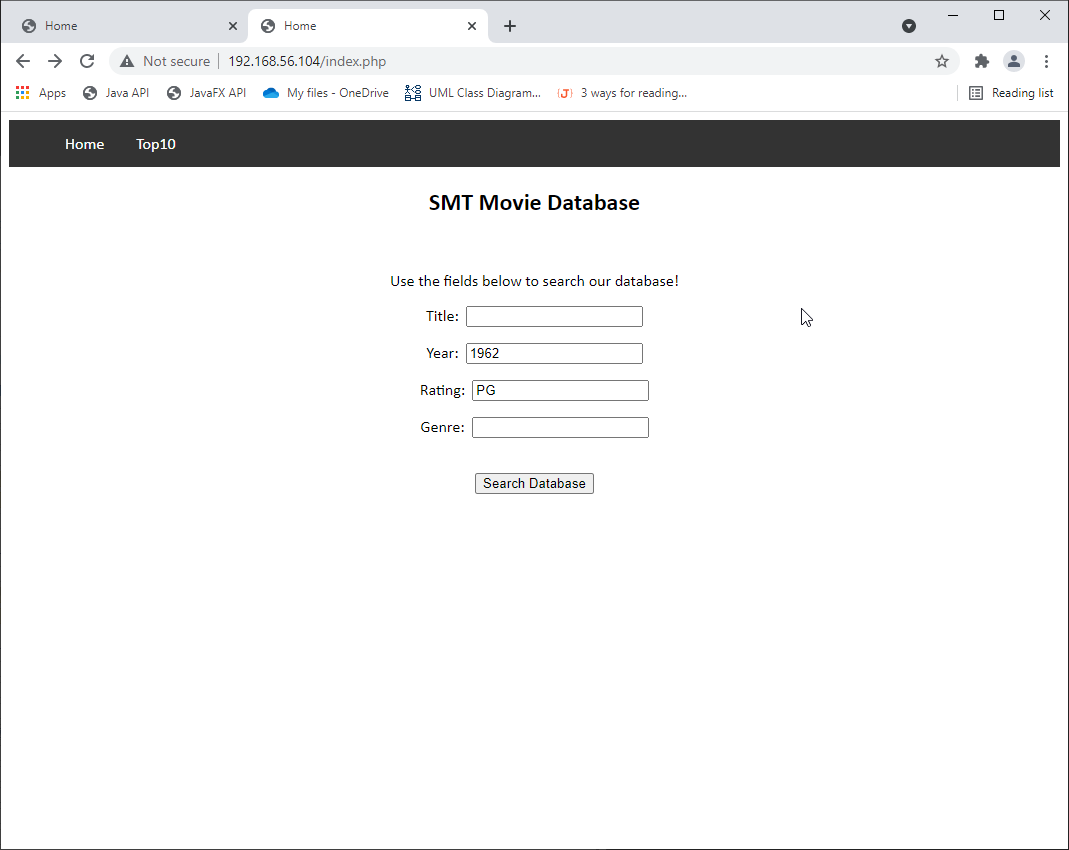
Figure



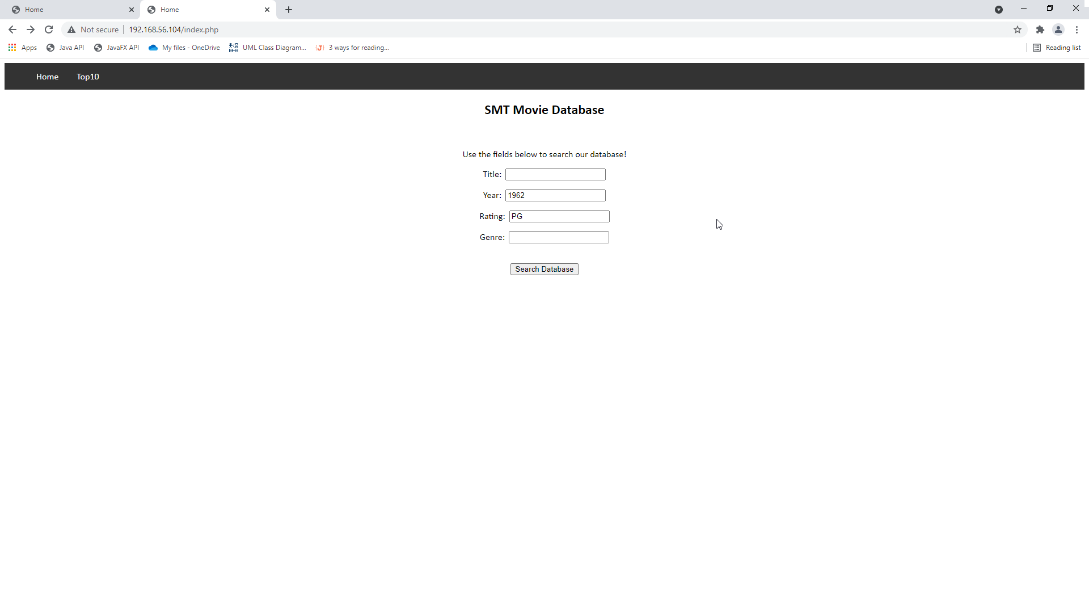
Figure



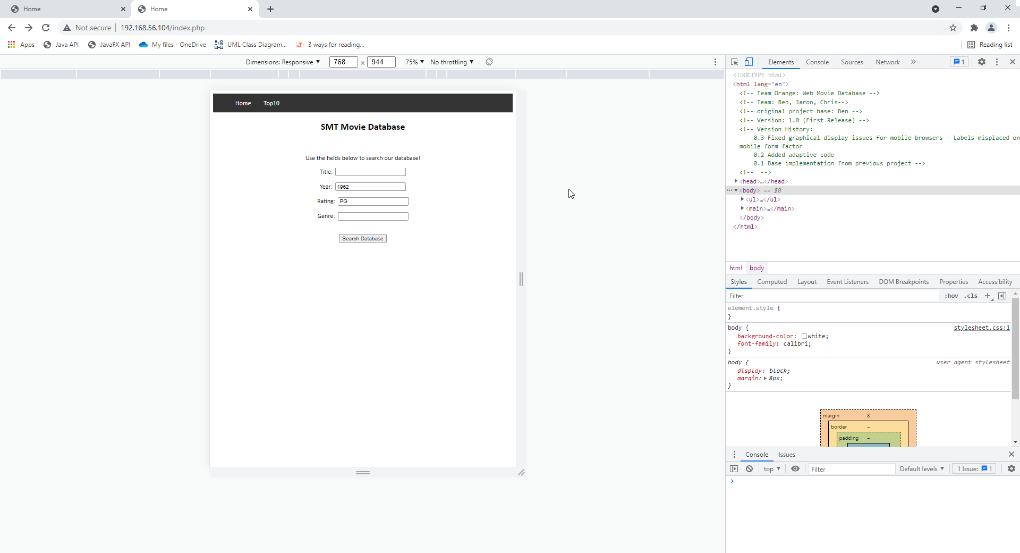
Figure



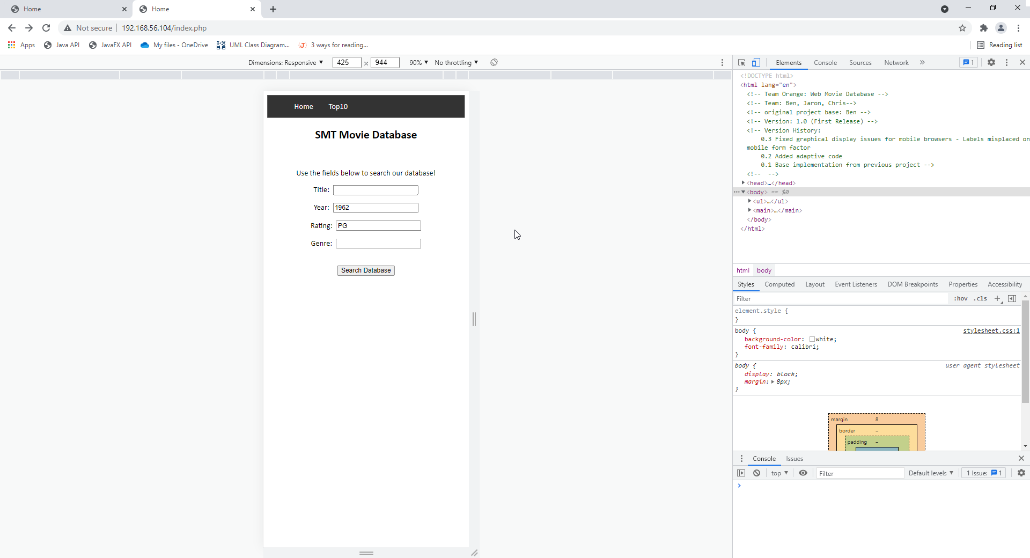
Figure



Figure



Figure



Figure

***End of Sprint One***

# Sprint Two – Chris C

## Software Review Plan

### Client Requirements:

* Free communications service - who sign up with their email
* Monthly newsletter
* Newsflash notification
* Select one or both communications options during the sign-up process
* Signup will not require a password
* Option to remove their subscription
* Removal process should send an email to the administrator requesting the removal of a members details from the database.
* Create a membership page
* Add new records to a membership database
* Capture the subscriber’s full name and email address.
* Ensure inputs are fully validated to filter out incorrect or erroneous information.
* Fully examine all possible user inputs
* Create a suitable User Interface for the administrator so they can enter an email address for removal.
* Include an option to view all members.

### Quality Assurance:

Throughout development of the movie database application, our team has created a Software Testing Plan to guide the testing of the application. This includes a testing report which will fully examine all user inputs and ensure user inputs are fully validated before the application is released. Each sprint will be using this guideline to fully test the application before moving on to the next sprint phase where testing will be done again.

### User Interface Plan:

Our team has thoroughly discussed the user interface plan for sprint two. The updates to the User Interface will be:

1. Home page: This page will have information about the database or allow the user to access other areas of the website.
2. Sign up page: This will allow users to sign up to the movie database using their name and email, with an option for email notifications.
3. Membership page: This will show the records of the user logged in or for an admin, all the users and allow adding of new records
4. Unsubscribe page: This page will allow the user to search their email and flag the database record in order for the record to be ready for removal.
5. Admin page: Will show all the flagged records of users wanting to unsubscribe, and will allow the admin to be able to drop the records from the database.

## Performance Report

### Code Optimisation:

Code optimisation is any code modification that can improve code quality and efficiency. A program may be optimised to become smaller in size, consume less memory, executes faster, or performs fewer input/output operations.

One appropriate code optimiser that could be beneficial to the project is DataDog. This code optimiser is a code optimiser for PHP through profiling. This software is used to detect the most resource-consuming methods or classes in PHP code.

This software allows users to have complete visibility into PHP performance, spend less time troubleshooting and, optimise legacy and cloud applications as it can pinpoint CPU, memory and latency bottlenecks, analyse code methods, classes and threads, allows visibility into code levels performance in development, staging and production environments and, identify performance issues such as blocked threads or memory leaks with an automatic analysis system.

Timeline

Description automatically generated

### Performance Tools:

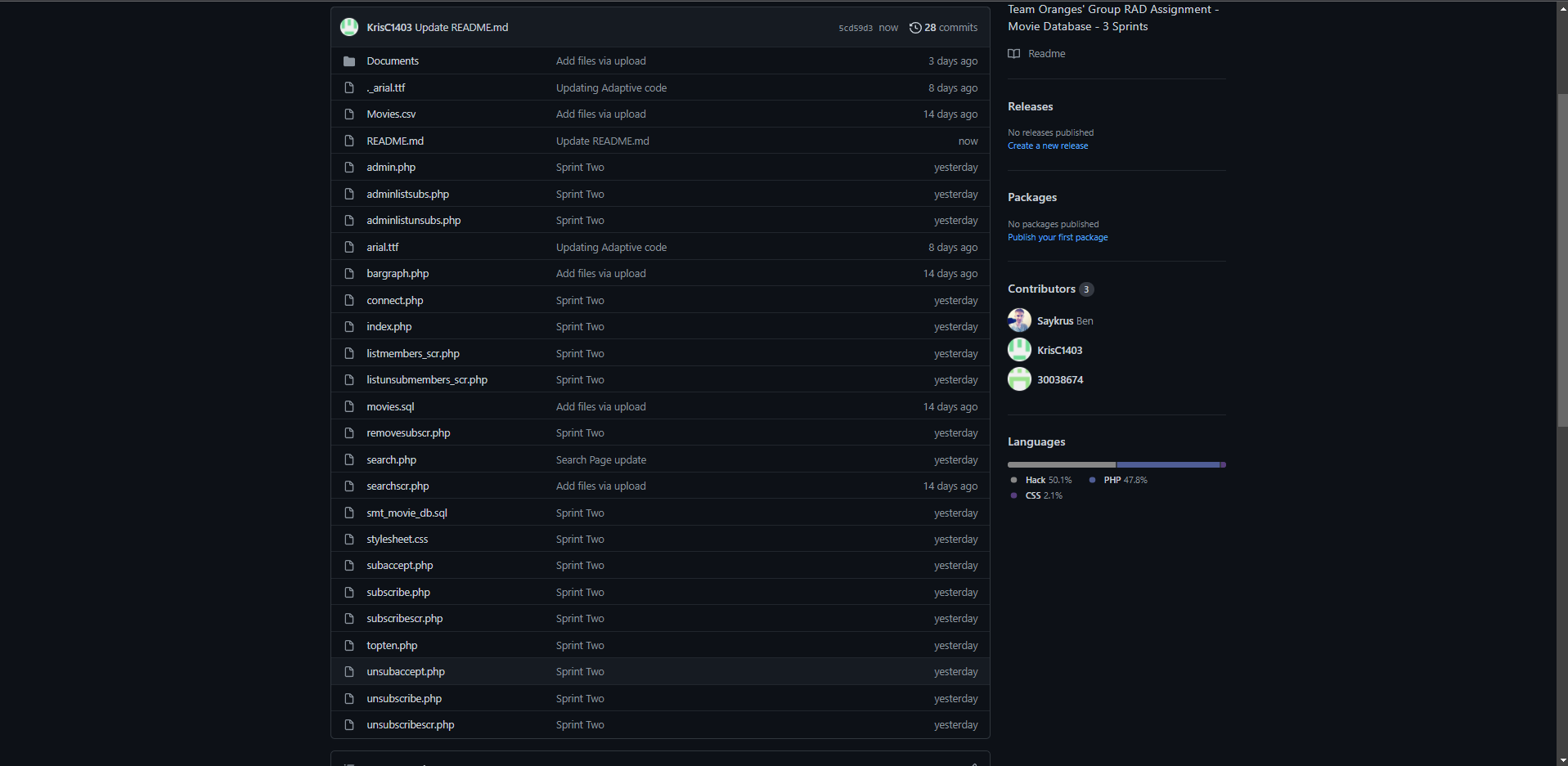
Performance tools support development in tuning the applications performance. They measure performance data during execution and provide analysis of the data to detect performance bottle necks and map optimisation opportunities. These tools uncover inefficient areas in code that take up critical resources which result in higher costs and inefficient performance.

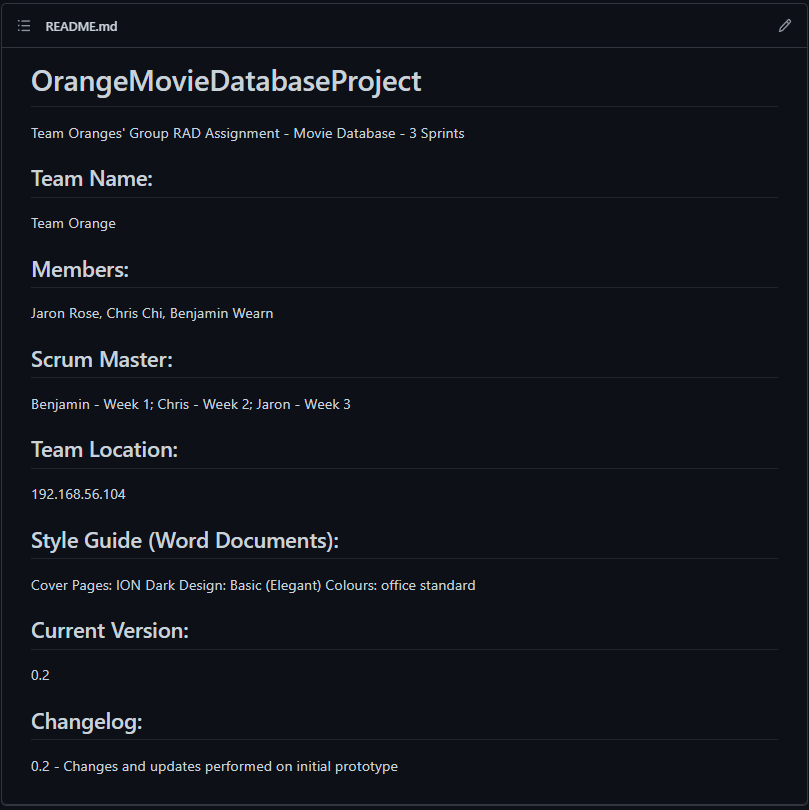
There are many performance tools that could be used throughout the project, these being gProfiler, SonarQube, and another software from DataDog. For this project, the choice of gProfiler was made, this is because it is an open-source profiler for any environment and it is free and purposed for collaboration. gProfiler continuously analyses code performance across the entire environment to optimise the most resource-consuming areas, improve performance and reduce costs and takes on an always-on approach which continuously samples CPU performance.

Graphical user interface, text, application

Description automatically generated

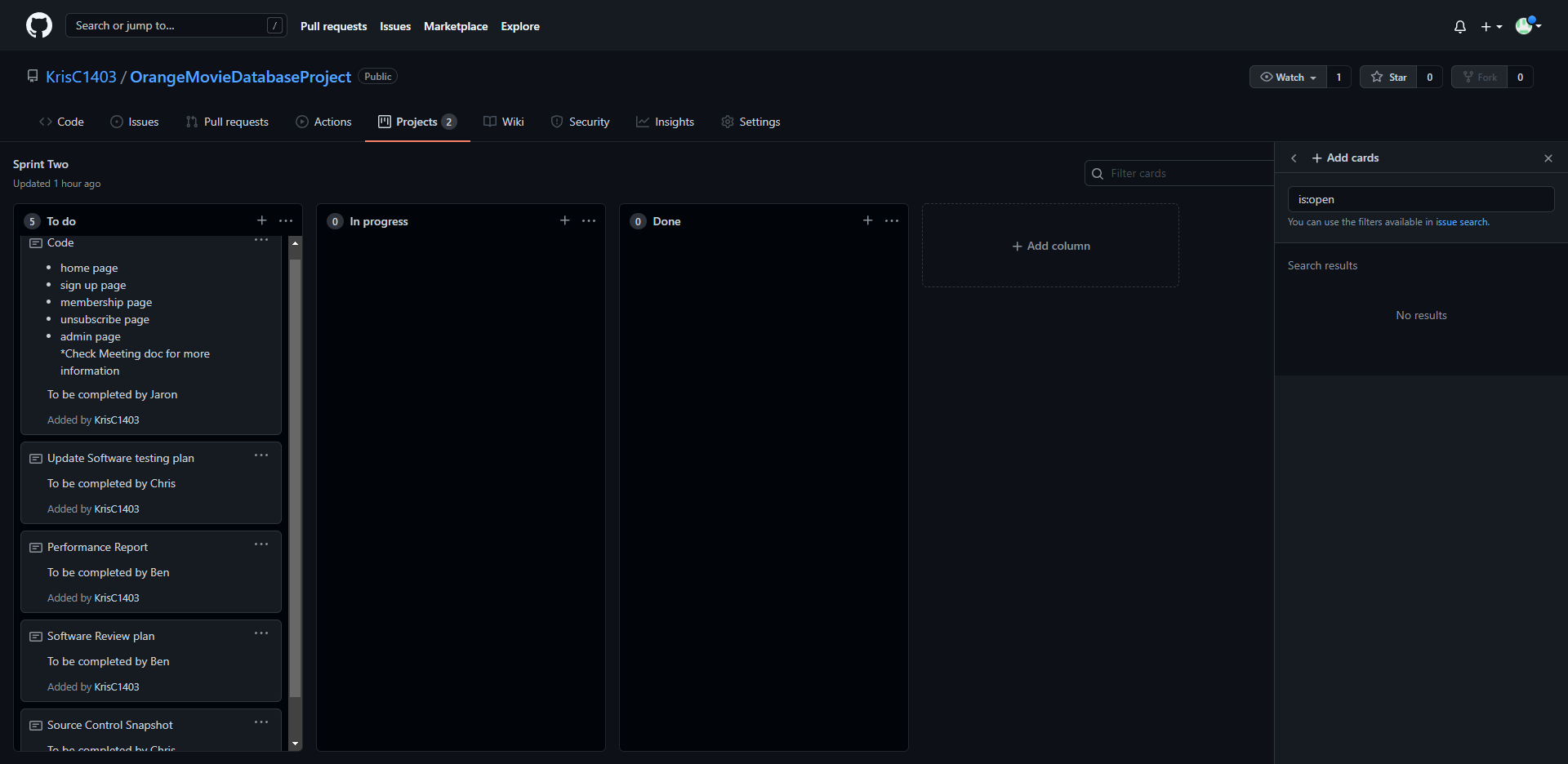
## Source Control – Sprint Two





## Project Management – Sprint Two

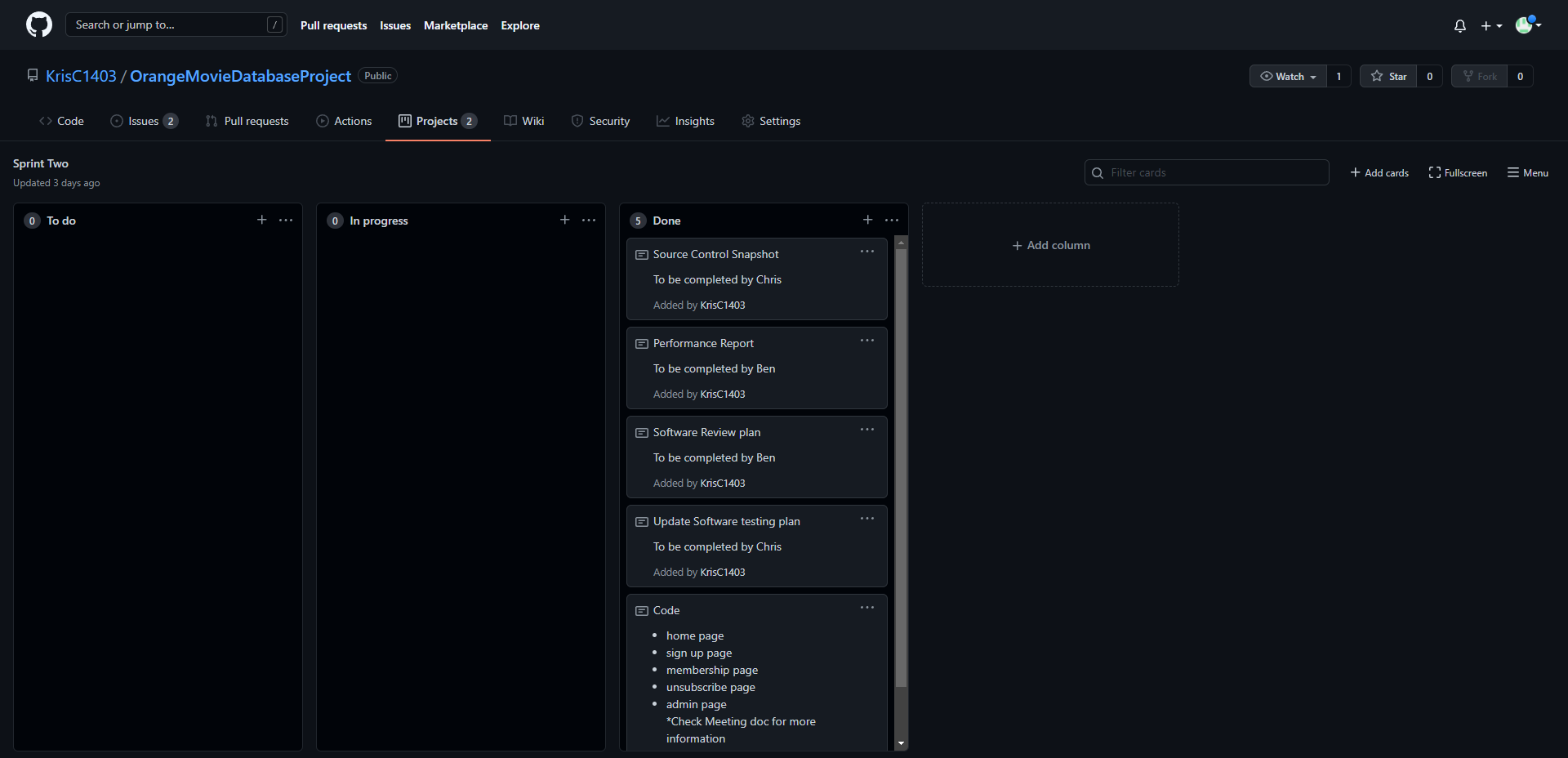
Project Start:



Project Underway:

A screenshot of a computer

Description automatically generated with medium confidence

Project Completion:

## Code Validation – Sprint Two

Index.php

Graphical user interface, text, application, email, Teams

Description automatically generated

Subscribe.php

Graphical user interface, text, application, email

Description automatically generated

subaccept.php

Graphical user interface, text, application, email

Description automatically generated

subscribescr.php

Graphical user interface, text, application

Description automatically generated

unsubscribe.php

Graphical user interface, text, application

Description automatically generated

unsubaccept.php

Graphical user interface, text, application, email

Description automatically generated

unsubscribescr.php

Graphical user interface, text, application

Description automatically generated

Admin.php

Graphical user interface, text, application, email

Description automatically generated

adminlistsubs.php  
Graphical user interface, text, application

Description automatically generated

adminlistunsubs.php

Graphical user interface, text, application, email

Description automatically generated

listmembers\_scr.php

Graphical user interface, text, application

Description automatically generated

listunsubmembers\_scr.php

Graphical user interface, text, application, email

Description automatically generated

## Testing Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Test** | **Expected Output** | **Actual** | **Reference** |
| Home Page | Show introduction | Show introduction | Refer Figure 14 |
| Sign up Page | Record name and email | Record name and email | Refer Figure 15 - 16 |
| Unsubscribe Page | Flag record for unsub | Flag record for unsub | Refer Figure 17 - 18 |
| Admin Page | Show flagged record and drop record | Show flagged record and drop record | Refer Figure 19 - 20 |

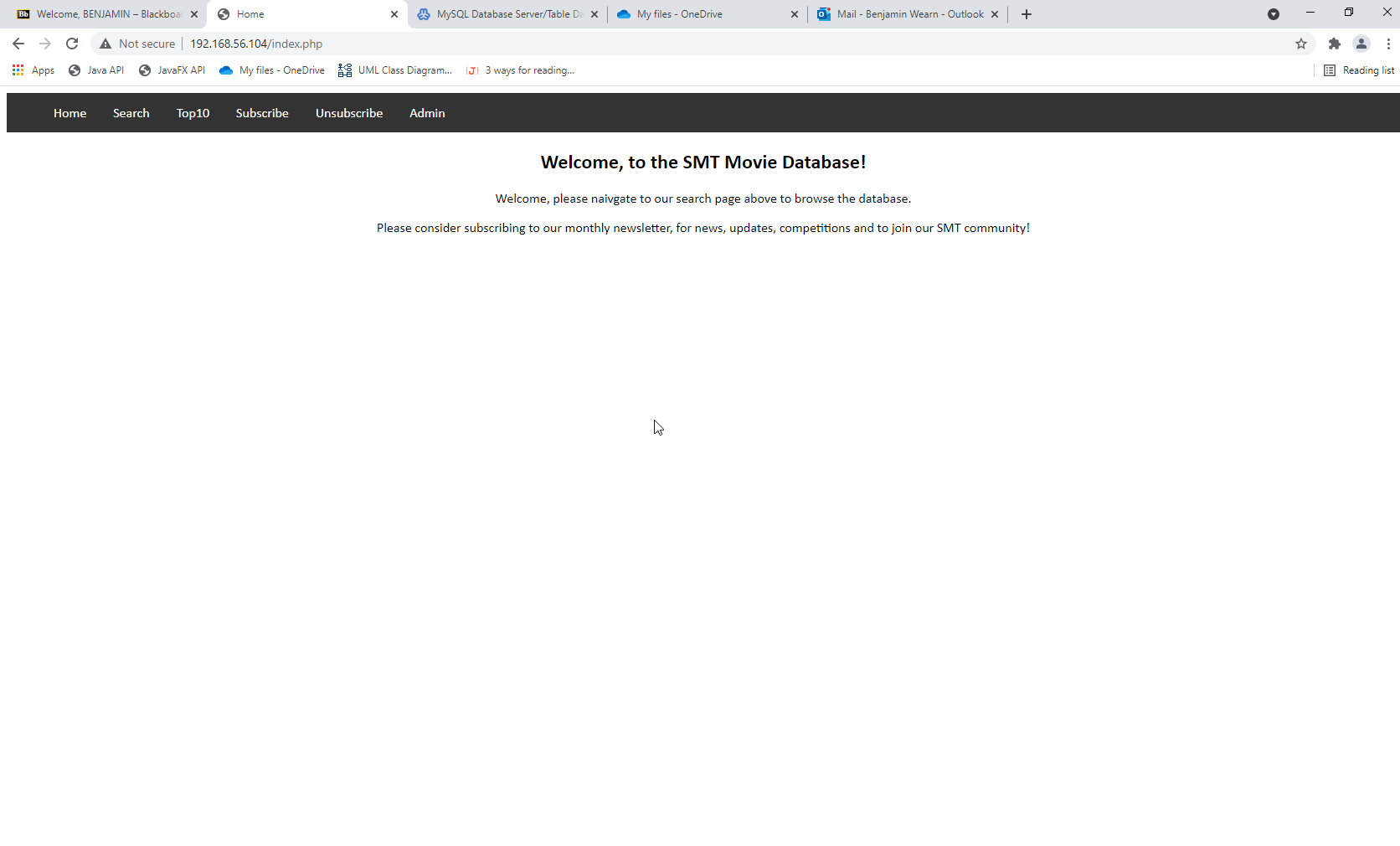


Figure - Home Page

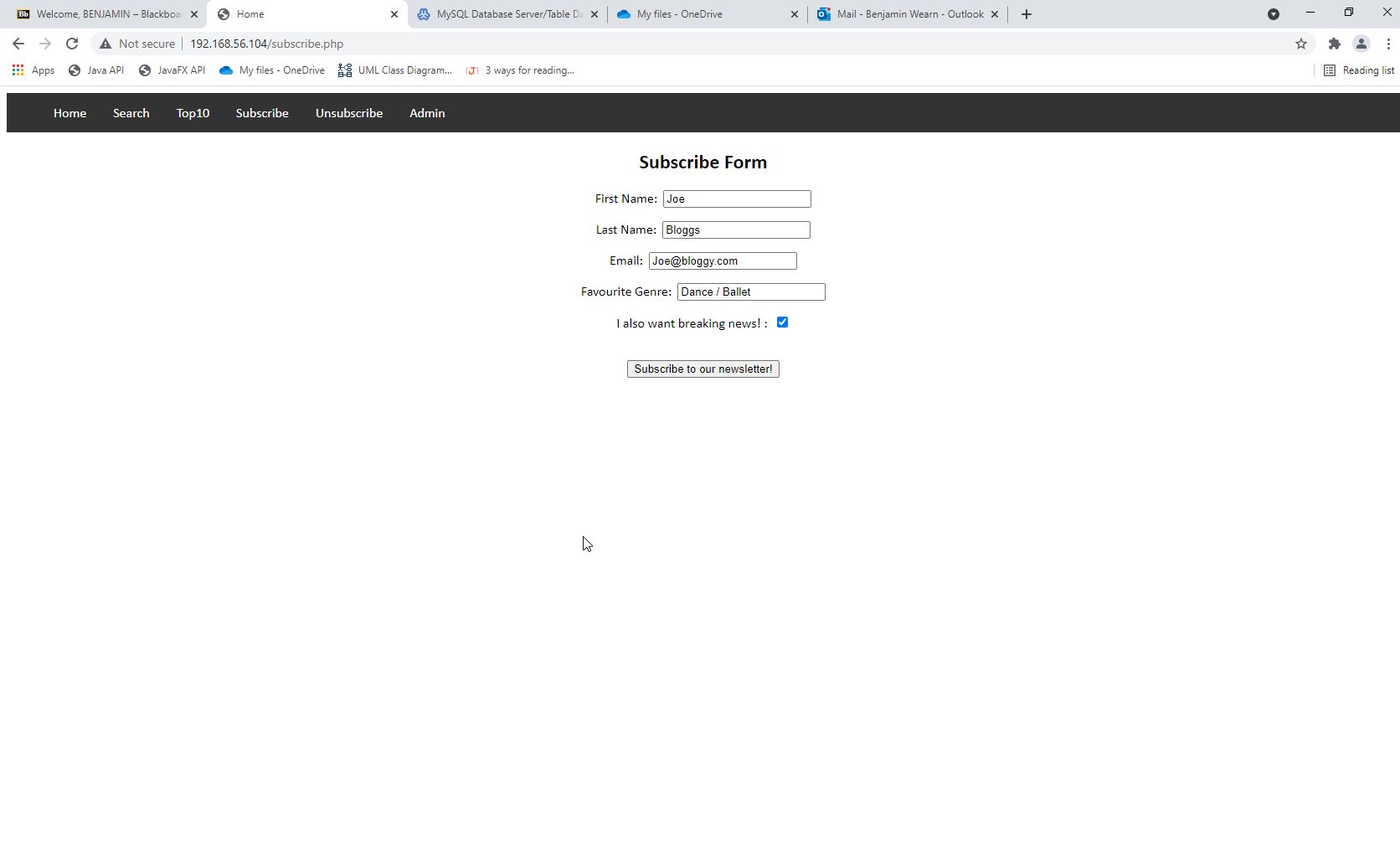


Figure - Sub Page

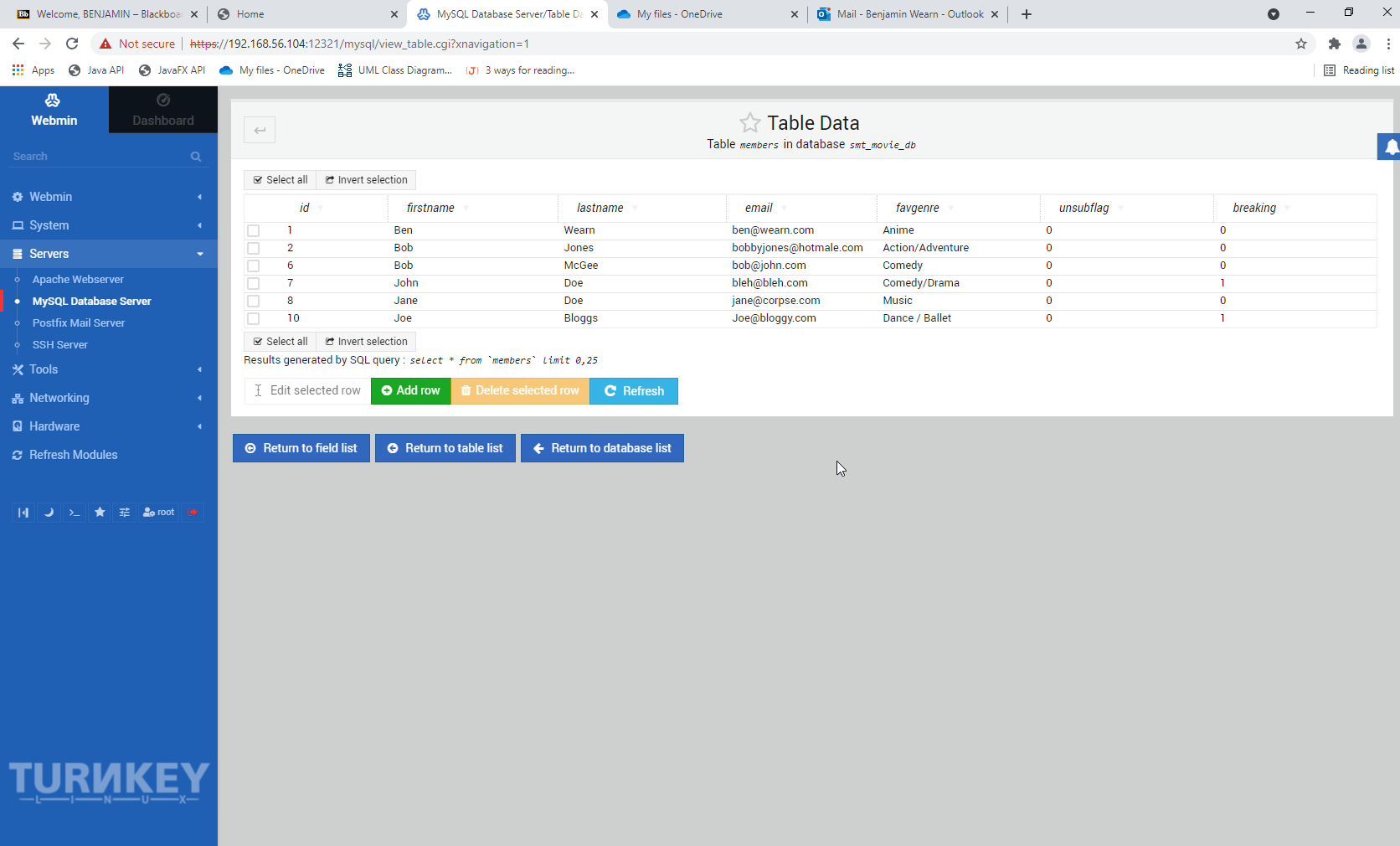


Figure - Sub Prove

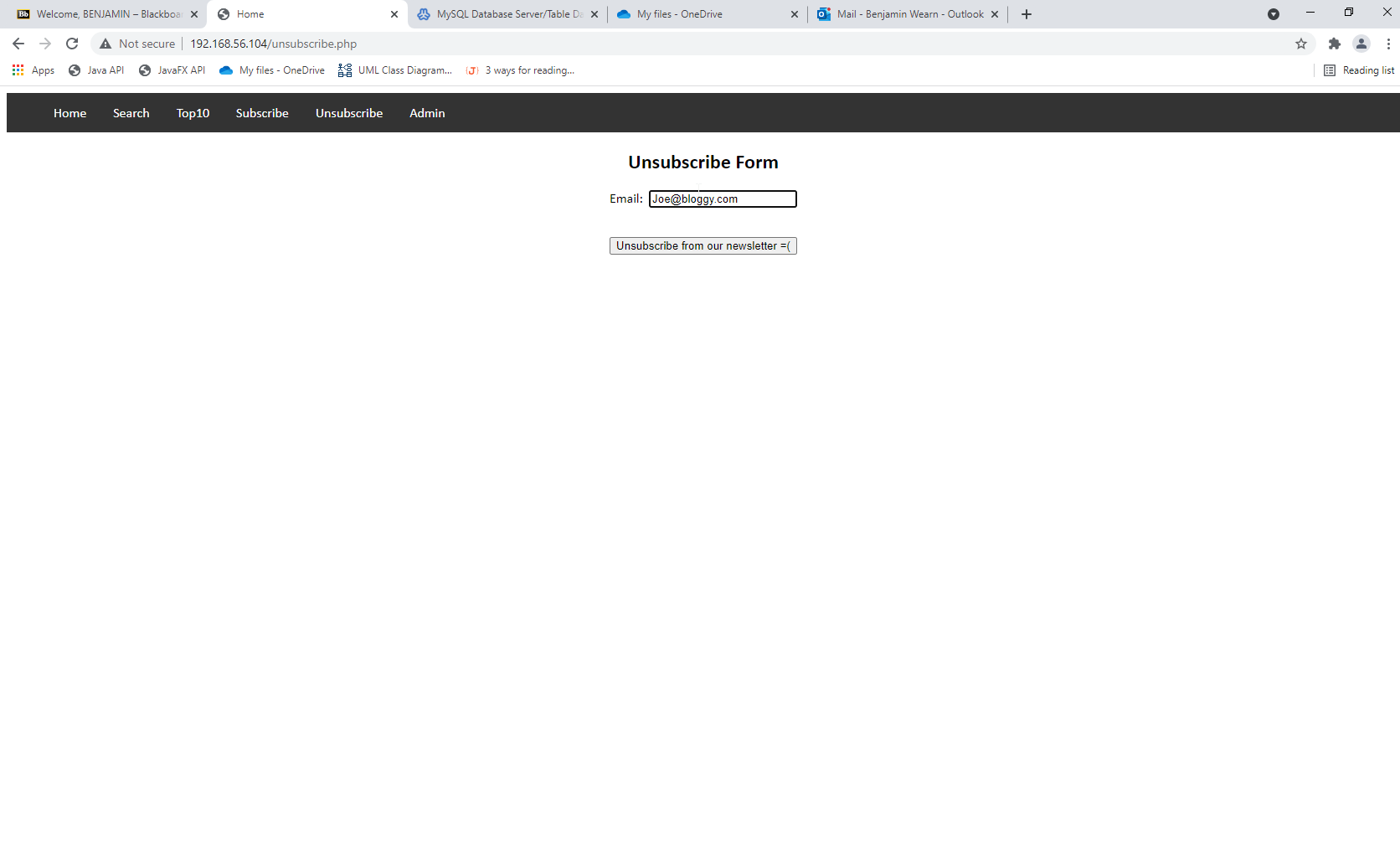


Figure - Unsub Page

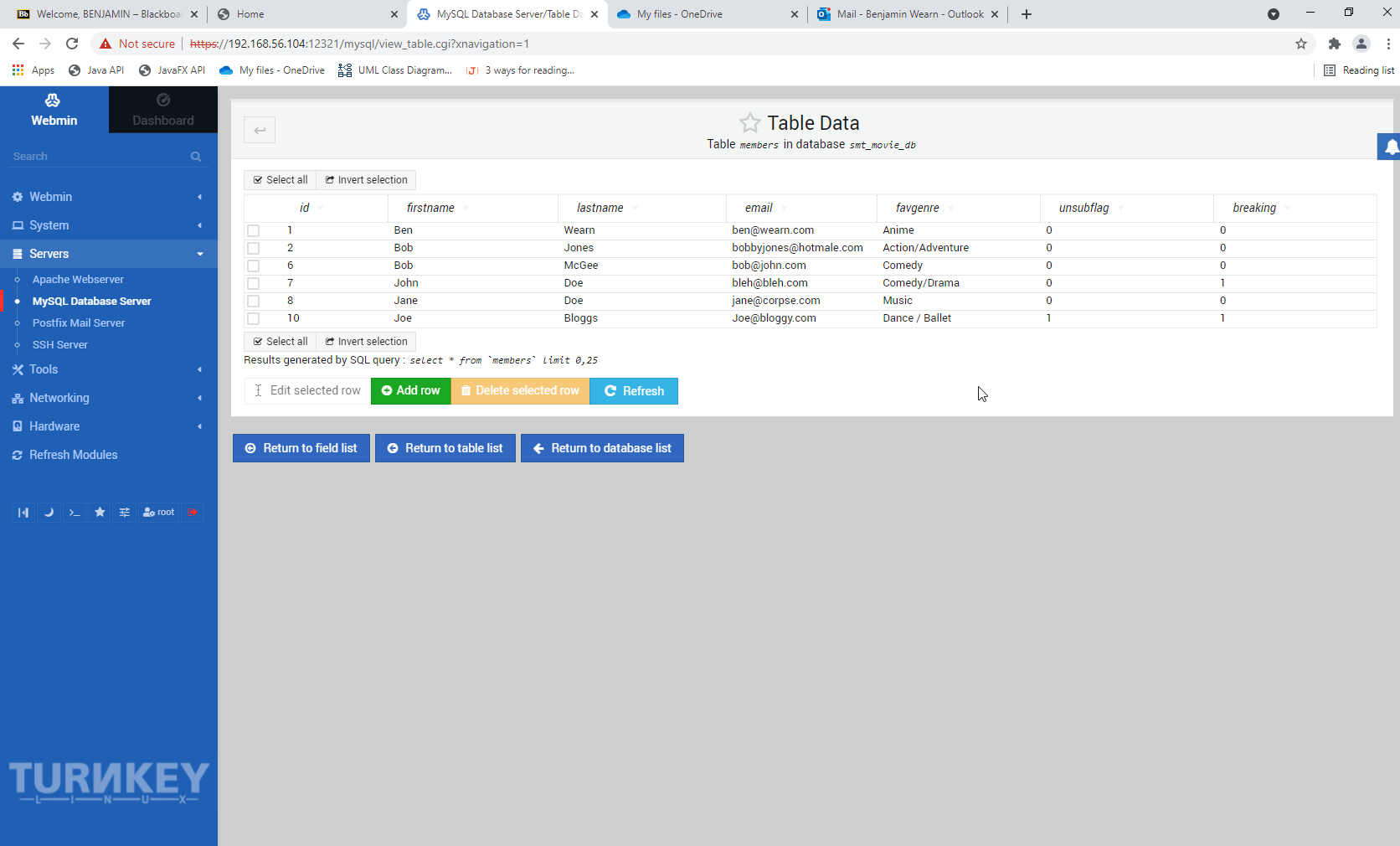


Figure - Flagged Prove

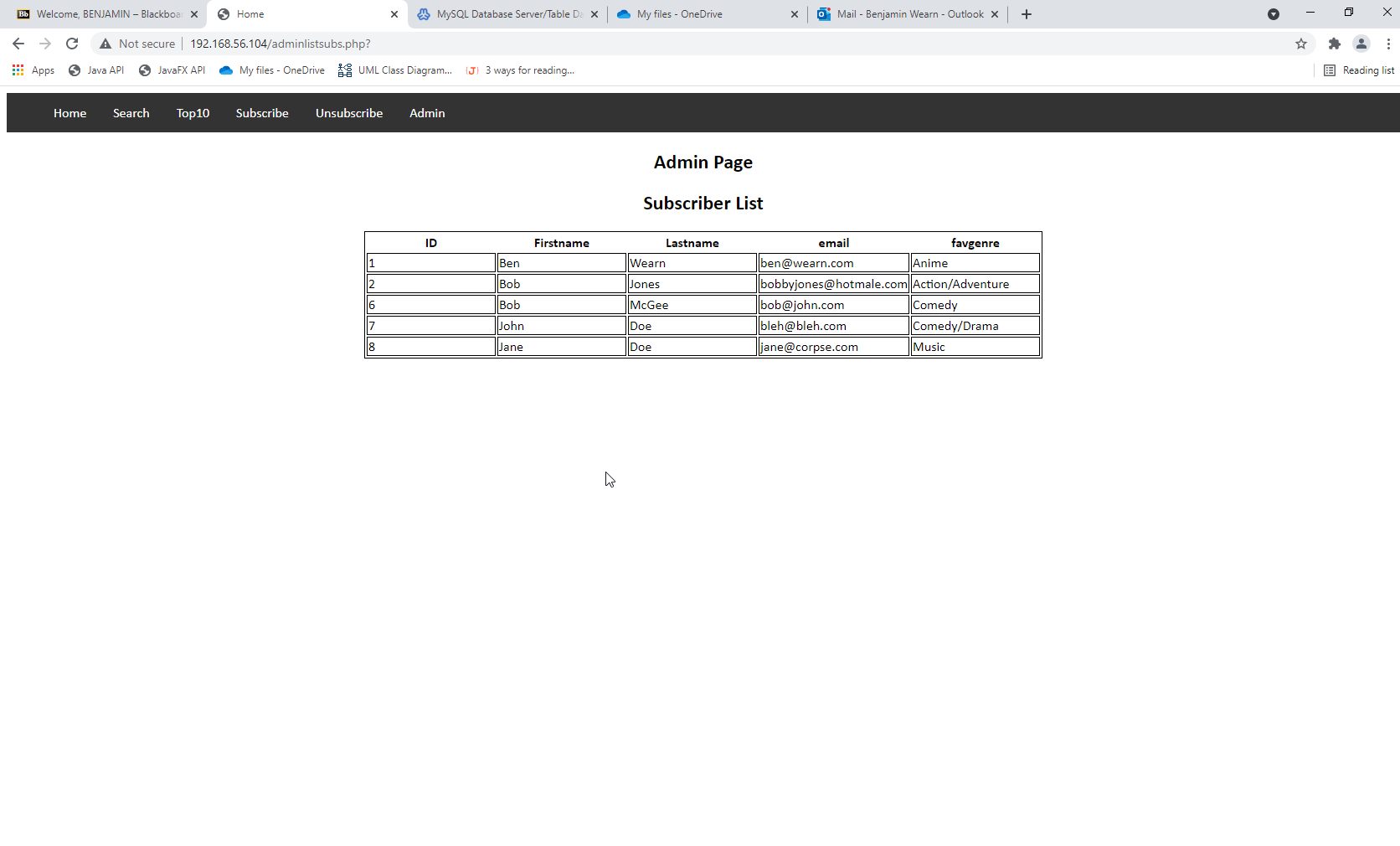


Figure - Admin sub Page

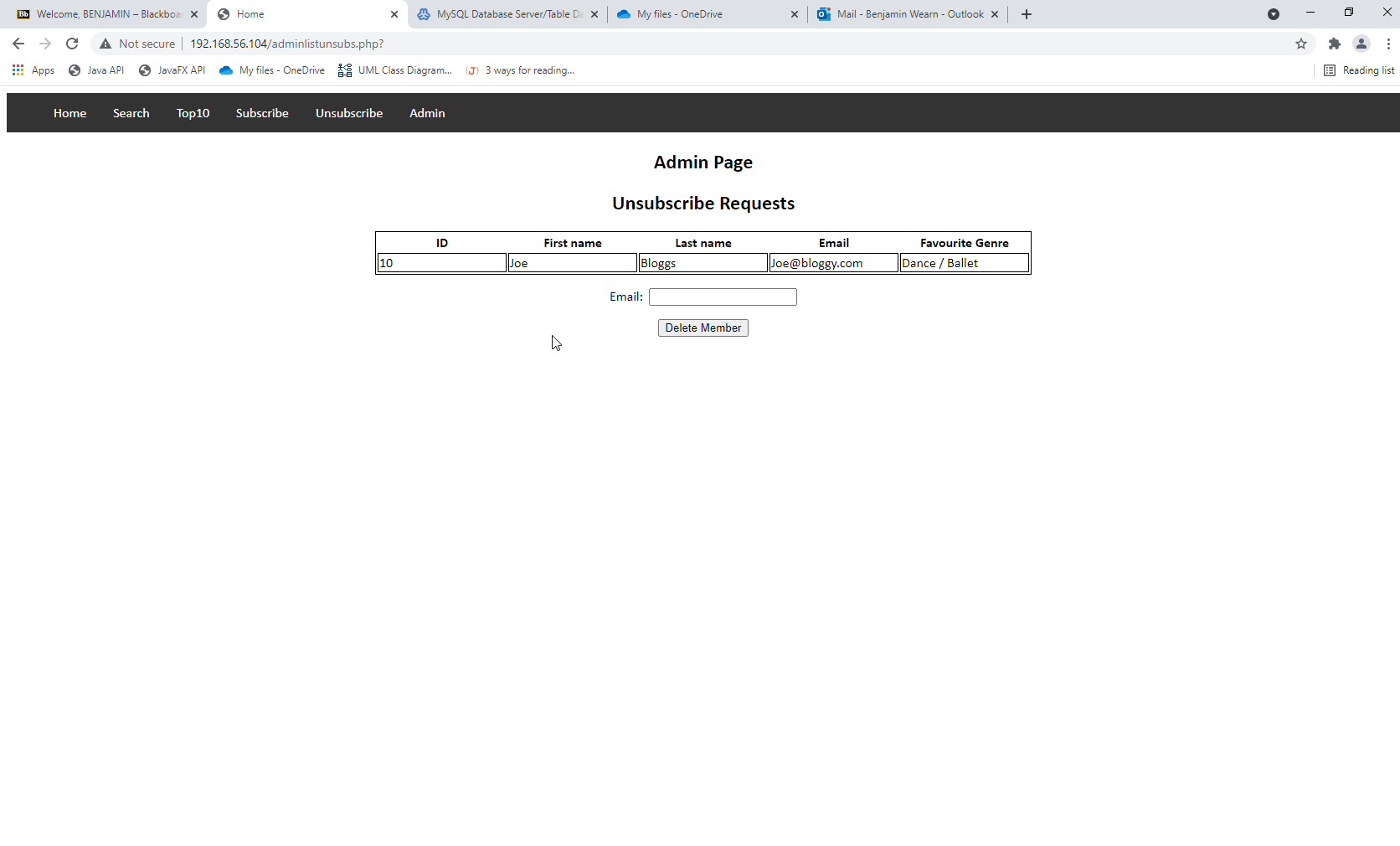


Figure - Admin Unsub Page

***End of Sprint Two***

# Sprint Three – Jaron R

## Optimisation Report:

### Installation:

To begin integration of the DataDog optimisation tool in the GitHub project repository, firstly a webhook must be added through settings.

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer screen

Description automatically generated with medium confidence

### Optimiser:

On the web browser for DataDog, there is a selection of options to choose from that allow the monitoring.

Graphical user interface

Description automatically generated

Through the metrics tab, users can set up what data is to be displayed. In example provided the graph chosen shows the system CPU usage.

Graphical user interface, application

Description automatically generated

This shows the load upon the CPU when running the project website. Although in testing, the website was not the only application running. However even with side applications running, the data shows that only 10% of the CPU was being used, thus showing that the CPU is not under strain when using the website.

Graphical user interface, chart, application, line chart

Description automatically generated

Multiple graphs may be used simultaneously, allowing for capture of more information, with labels on what information is being gathered.

Graphical user interface, application, Teams

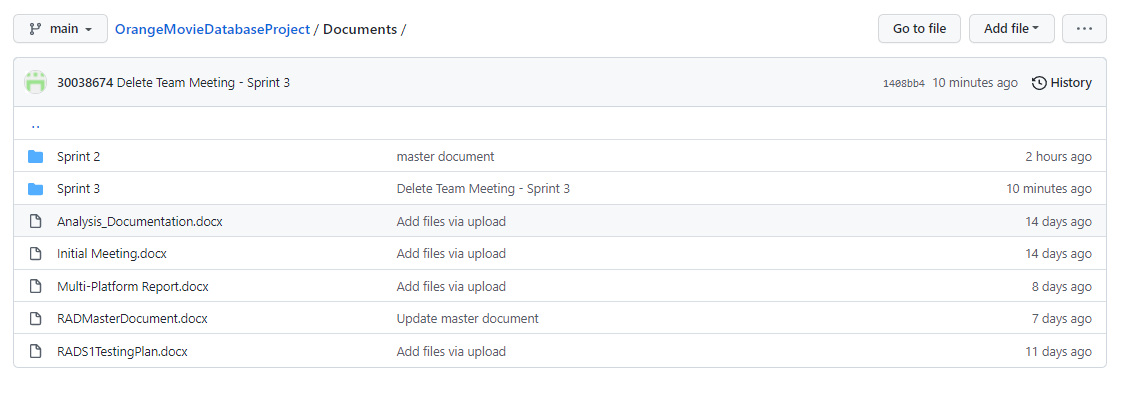
Description automatically generated

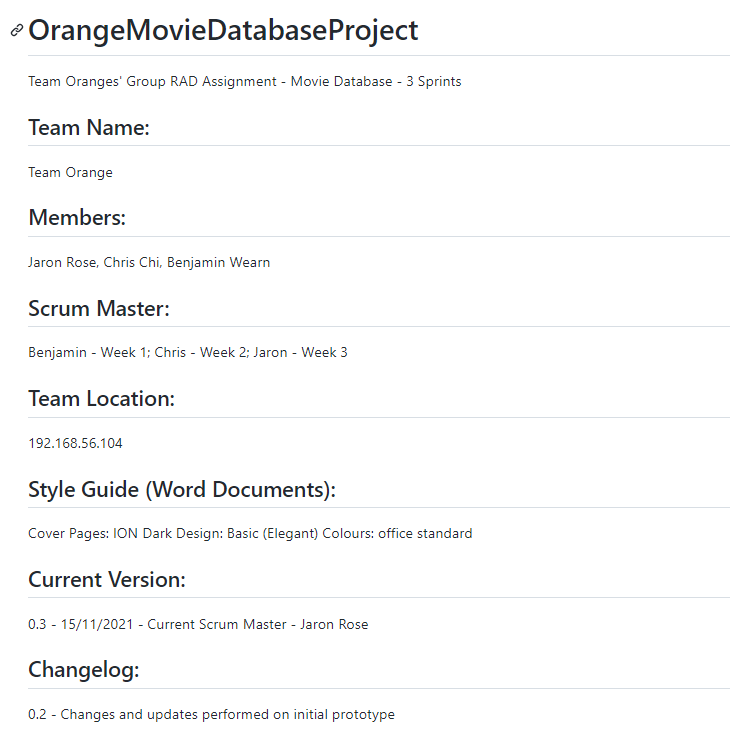
The integrations tab allows for integration of other software into the DataDog tracker in order to track information from other software.

Graphical user interface, text, application

Description automatically generated

## Source Control – Sprint Three



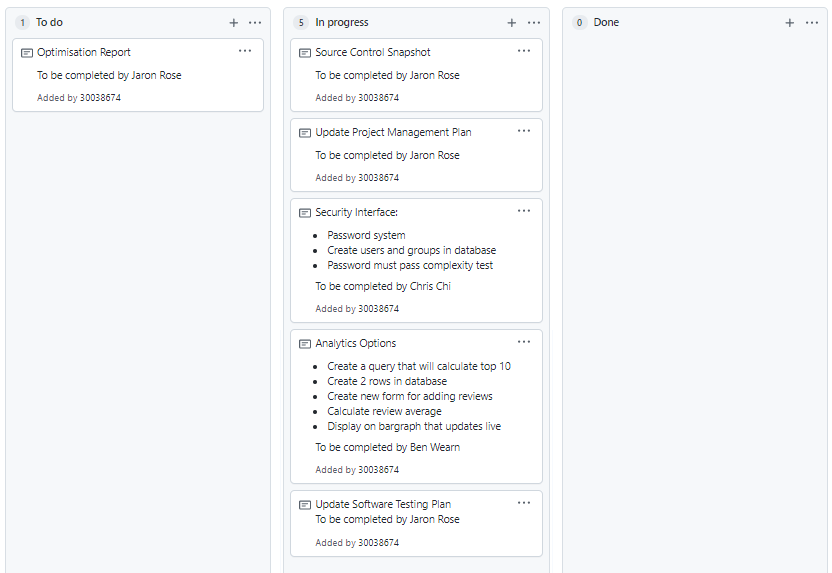


## Project Management – Sprint Three

**Project Start:**

## 

**Project Underway:**



**Project Completion:**

Graphical user interface, text, application

Description automatically generated

## Code Validation – Sprint Three:

**Addratingscr.php**

Graphical user interface, text, application, email

Description automatically generated

**Admin.php**

Graphical user interface, text, application, email

Description automatically generated

**Adminlistsubs.php**

Graphical user interface, text, application

Description automatically generated

**Adminlistunsubs.php**

Graphical user interface, text, application, email

Description automatically generated

**Bargraph.php**

Graphical user interface, text, application, email

Description automatically generated

**Index.php**

Graphical user interface, text, application, email

Description automatically generated

**Search.php**

Graphical user interface, text, application, email

Description automatically generated

**Searchscr.php**

Graphical user interface, text, application

Description automatically generated

**Subaccept.php**

Graphical user interface, text, application, email

Description automatically generated

**Subscribe.php**

Graphical user interface, text, application, email, Teams

Description automatically generated

**Topten.php**

Graphical user interface, text, application, email

Description automatically generated

**Toptenscr.php**

Graphical user interface, text, application

Description automatically generated

**Unsubaccept.php**

Graphical user interface, text, application, email

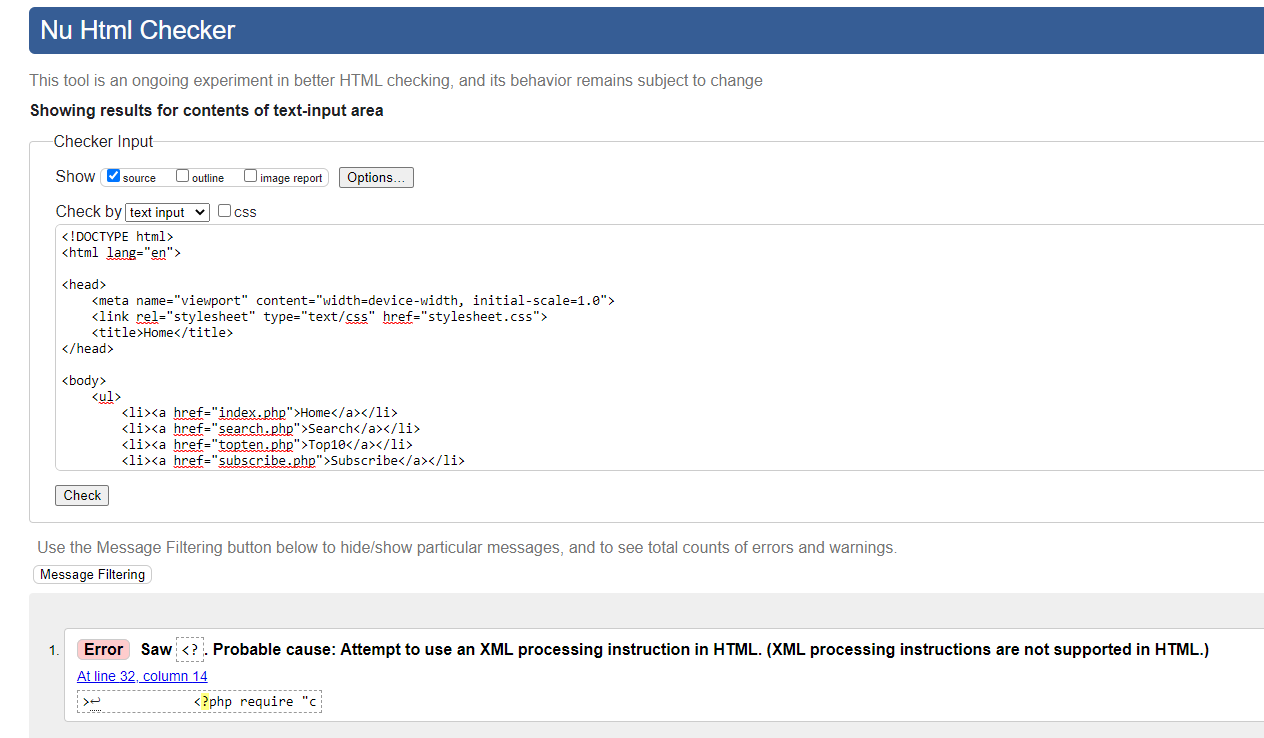
Description automatically generated

**Unsubscribescr.php**

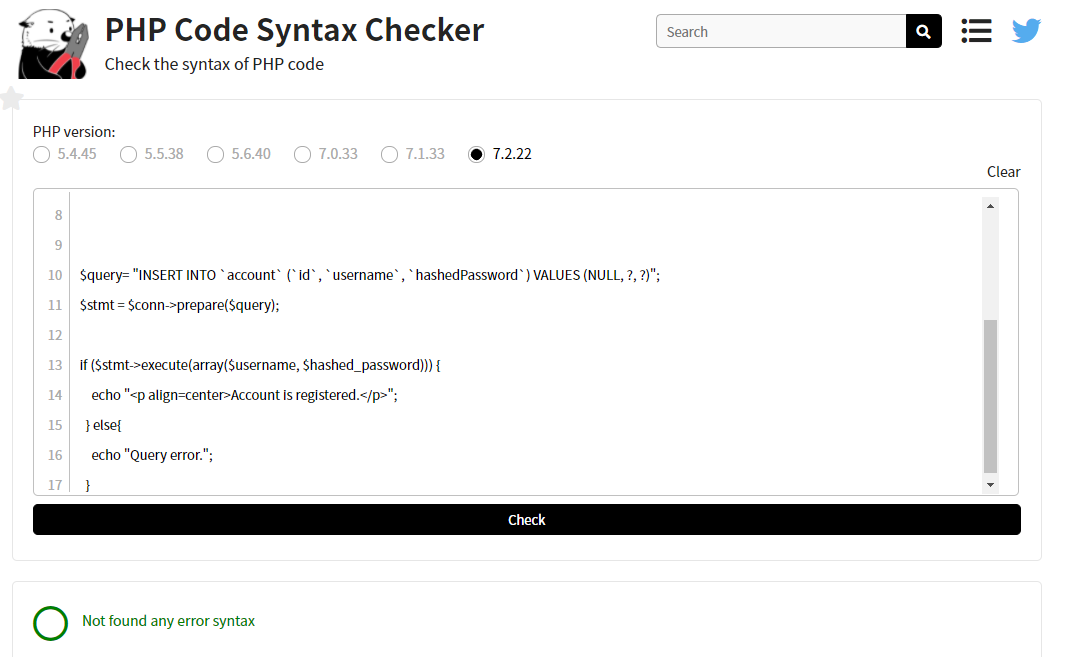
Graphical user interface, text, application

Description automatically generated

**createLogin.php**



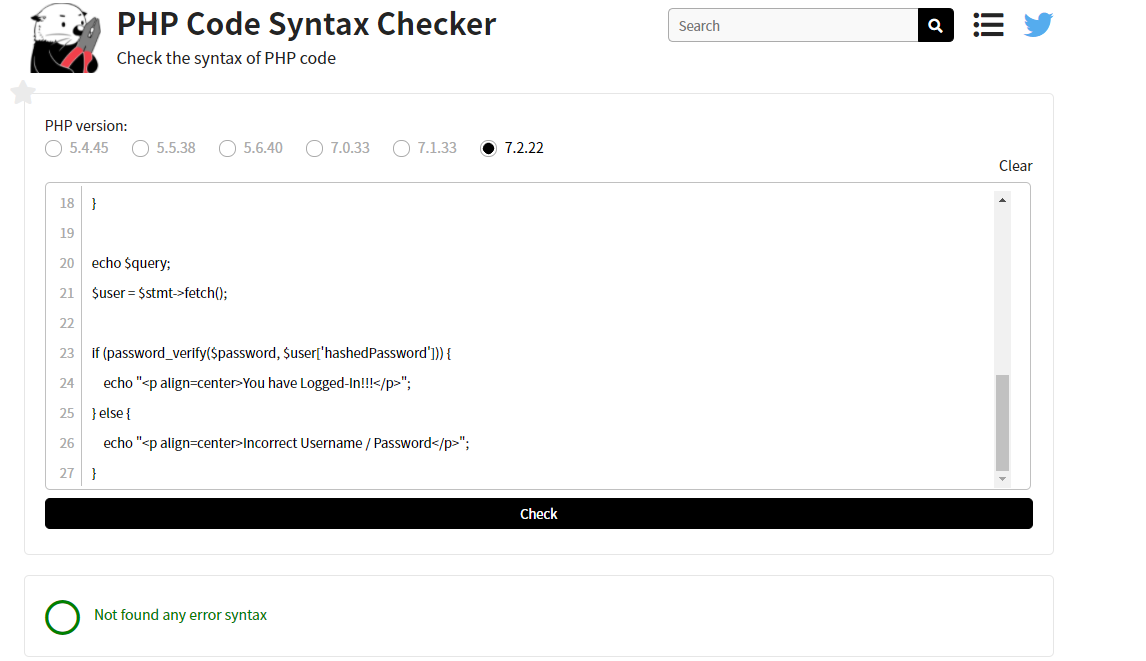
**createLogin\_scr.php**



**userLogin.php**



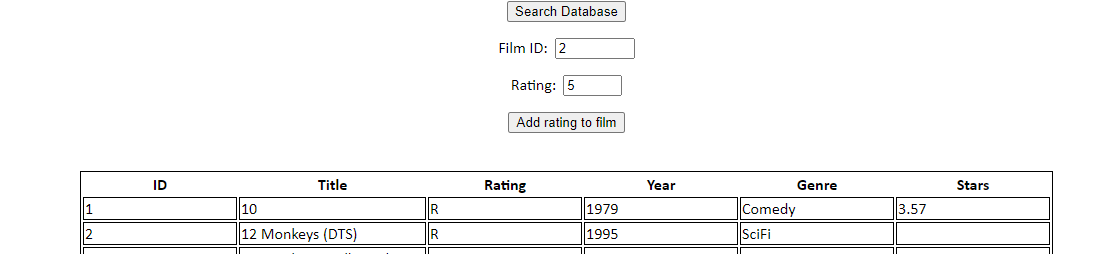
**userLogin\_scr.php**



## Testing Table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Test** | **Expected Output** | **Actual** | **Reference** |
| Add review to movie | Review is added in database | As Expected | Ref21 |
| Top 10 movies listed | Table is shown along with rating | As Expected | Ref22 |
| Top 10 updates on review | Once a review is added, table is updated | As Expected | Ref23 |
| Incorrect login | Error message | As Expected | Ref24 |
| Login | Login success | As Expected | Ref25 |
| Create user login | Successful user creation | As Expected | Ref26 |
| Login with new user | Login success | As Expected | Ref27 |

**BEFORE AND AFTER:**



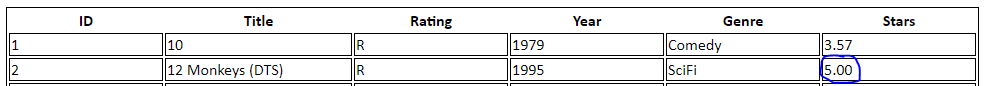


Figure 21

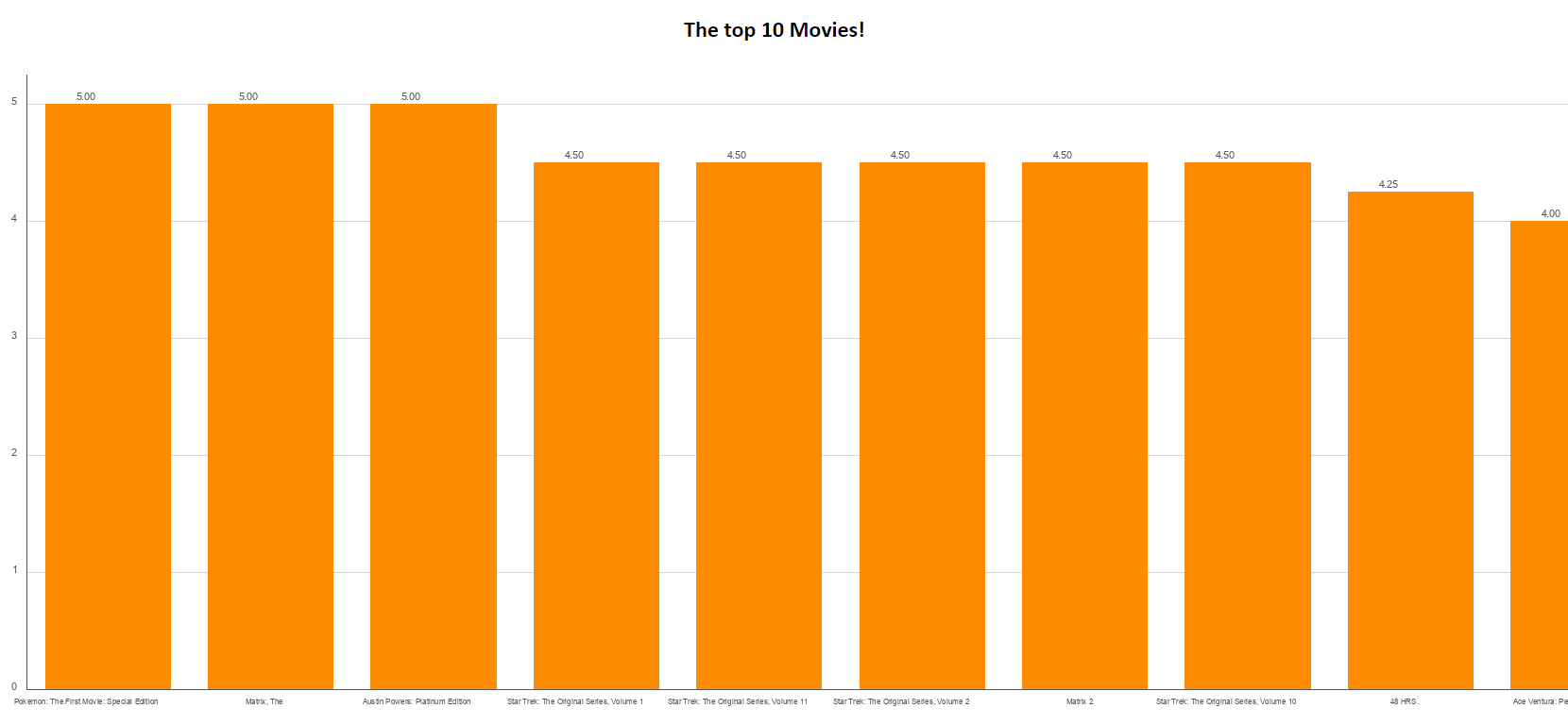


Figure 22

**BEFORE AND AFTER:**

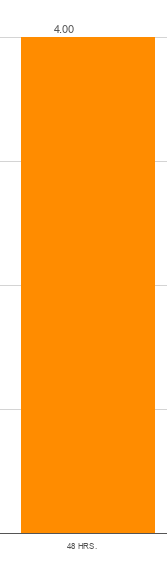


Figure 23

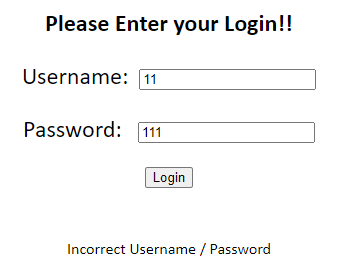


Figure 24

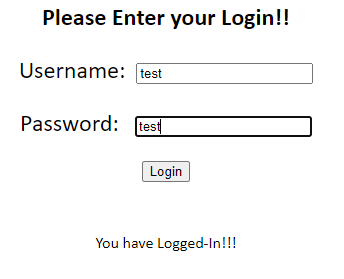


Figure 25

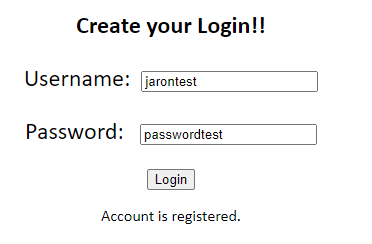


Figure 26

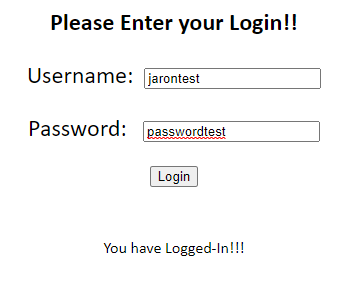


Figure 27

# Project Hand Over:

## Software Review Plan:

### Quality Control:

#### Ensuring Quality of Software:

Throughout the three sprints of the project, the way our team ensured the quality of the software was through using test plans. This test plan outlined the types of testing available to use and the type of testing that was to be used to test the software. The testing that was used throughout development was the use of Test Tables. The test tables were set for the team to use as a guideline while testing the software, allowing the team to validate all inputs and find any occurrences that were out of what was expected. FOR EXAMPLE:

Table

Description automatically generated

This is used to ensure what test was conducted, the expected outcome of the test and what happened during the test. Through doing this type of testing, the team was able to find whether the software ran as intended and if anything was needed for change.

#### Processes Used to Control Quality of Development Process:

Throughout each sprint, there were clear instructions on what needed to be done for the software to develop, this was done through a sprint document which outlined the tasks, and client requirements that were updated each week. The team discussed the updated client requirements and tasks each week, and split up the work accordingly to successfully finish each sprint. This allowed the team to understand what work was needed to be done and work more efficiently as a team. FOR EXAMPLE:

Graphical user interface, text, application

Description automatically generated

#### Code Testing:

Throughout development, each update of the software went through a code validator, whether it was HTML or PHP, the code validator showed the team whether there were inefficiencies in the code or errors in the code. FOR EXAMPLE:

Graphical user interface, text, application, email

Description automatically generated

This shows the team whether the code is working and if changes are needed.

### Future Modifications:

The software being handed over will be able to be modified and/or refined later, as all code is commented and there are code documentations showing version control, and changes that were previously made. Development plans have been saved showing all the client requirements and what was needed to be completed throughout development

### Requirements Mapping:

All requirements set by the client have been met.   
The user requirements are listed below:

#### Sprint One:

* Source Control Snapshots
* Project Management Plan Creation
* Software Development Testing Plan – Incorporated QA Standards of CITE
* Analysis Report
* Multi-Platform Report (Adaptive Vs Responsive)

#### Software Requirements:

* Multi-Platform support

#### Sprint Two:

* Source Control Snapshots
* Software Review Plan
* Performance Report
* Update Project Management Plan
* Update Software Testing Plan

##### Software Requirements:

* Add information to home page
* Sign up page (Name and Email)
* Membership page (add new records)
* Unsubscribe page (Search email, flag record for unsub)
* Admin page (show flagged records and drop records)

#### Sprint Three:

* Source Control Snapshot
* Optimisation Report
* Update Project Management Plan
* Update Software Testing Plan

##### Software Requirements:

* Implement Security Interface (Password system, for acme personnel and admin)
* Analytics Options (Calculate top 10 most popular movies)

1. Update bar graph live
2. Allow user reviews
3. Calculate top 10 using user reviews

#### Hand Over:

* Source Control Snapshot
* Software Review Report
* Update Project Management Plan
* Update Software Testing Plan

##### Software Requirements:

* WCAG Inclusion

## Source Control – Project Hand Over

Text, application

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

## Project Management – Project Hand Over

**Project Start:**

Graphical user interface, text

Description automatically generated

**Project Underway:**

Graphical user interface, text, application

Description automatically generated

**Project Completed:**

Graphical user interface, text, application

Description automatically generated

## Code Validation – Project Hand Over (All Updated Scripts)

**Index.php**

Graphical user interface, text, application, email

Description automatically generated

**Search.php**

Graphical user interface, text, application, email

Description automatically generated

**Subscribe.php**

Graphical user interface, text, application, email

Description automatically generated

**Topten.php**

Graphical user interface, text, application, email

Description automatically generated

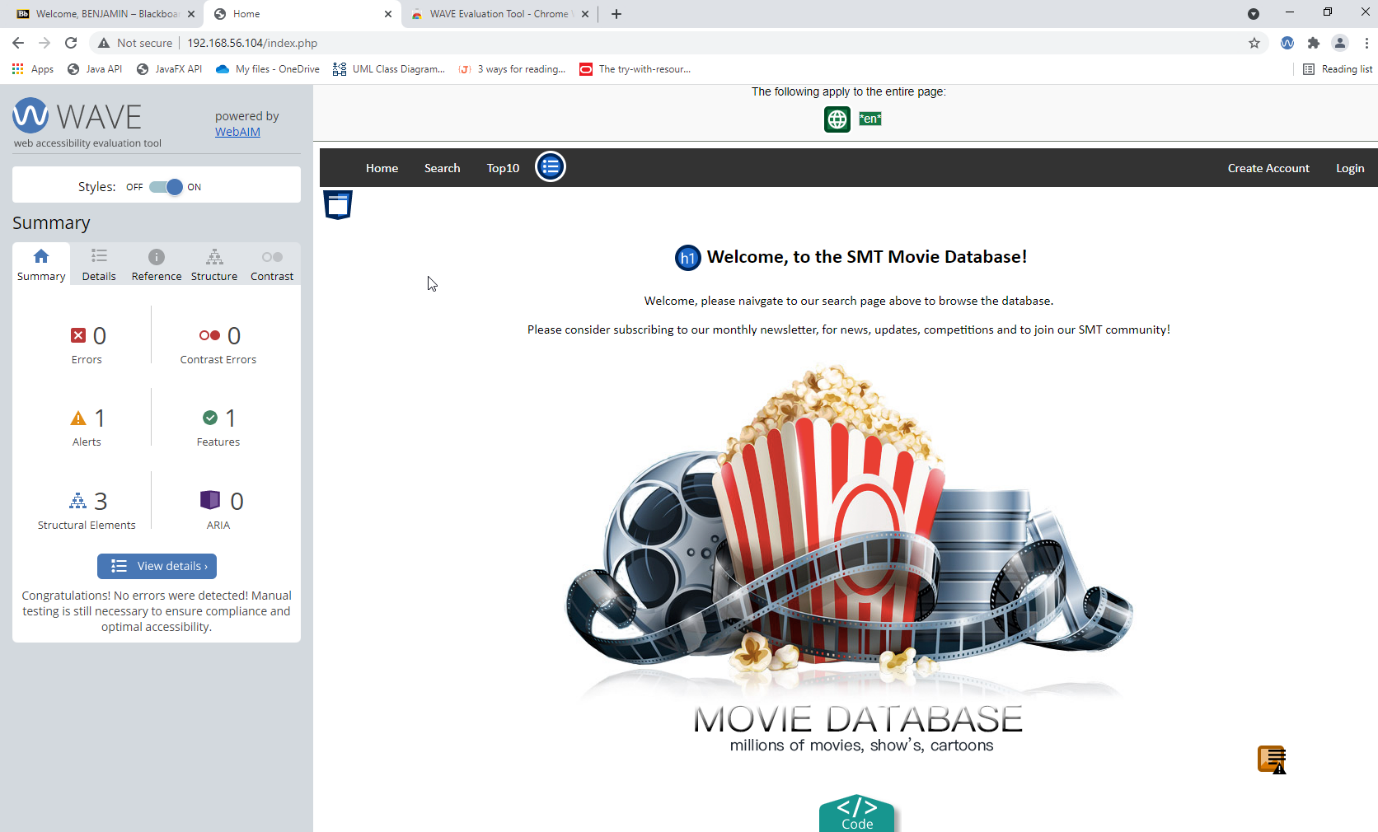
**Unsubscribe.php**

Graphical user interface, text, application, email

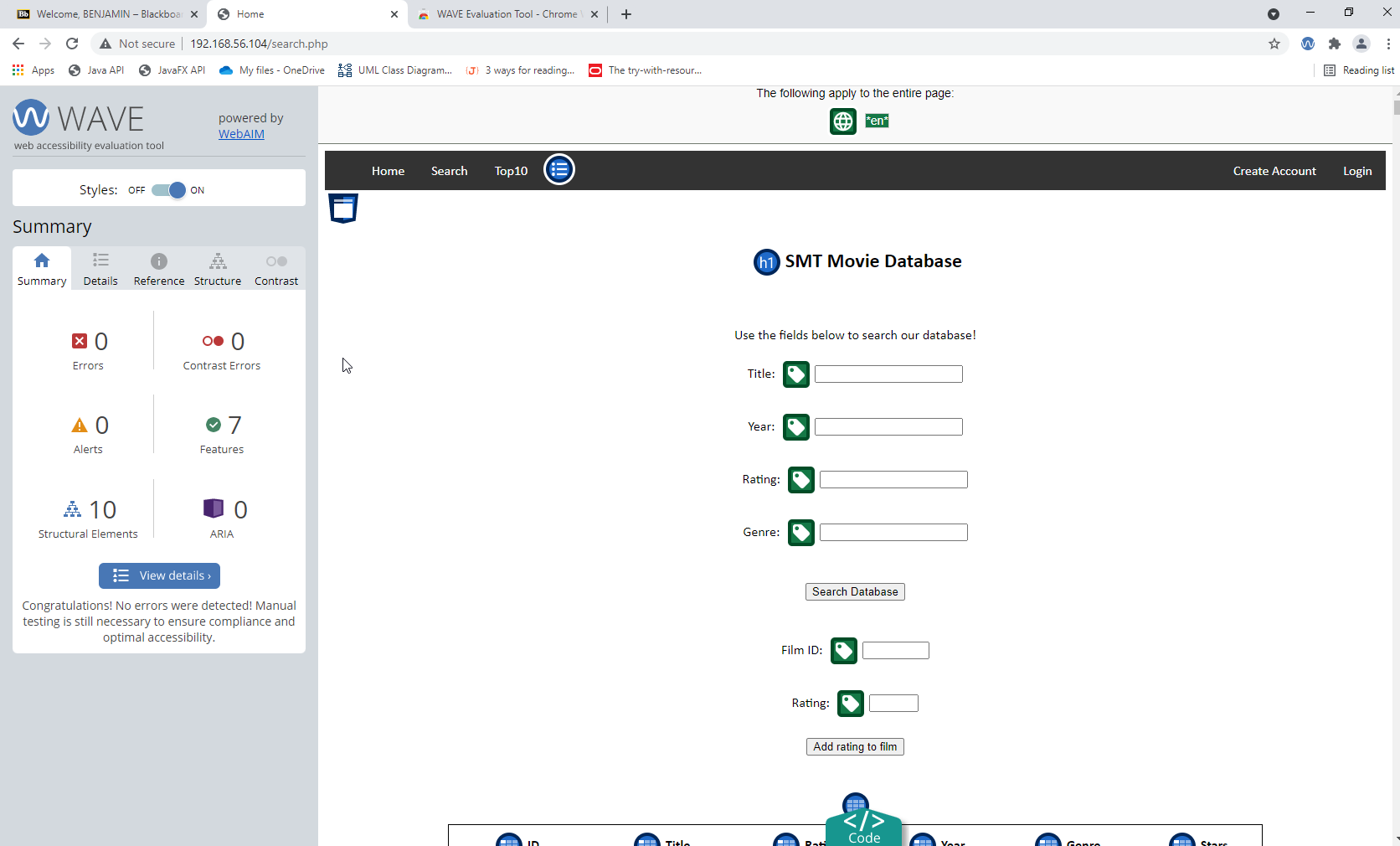
Description automatically generated

## Testing Table WCAG – Project Hand Over

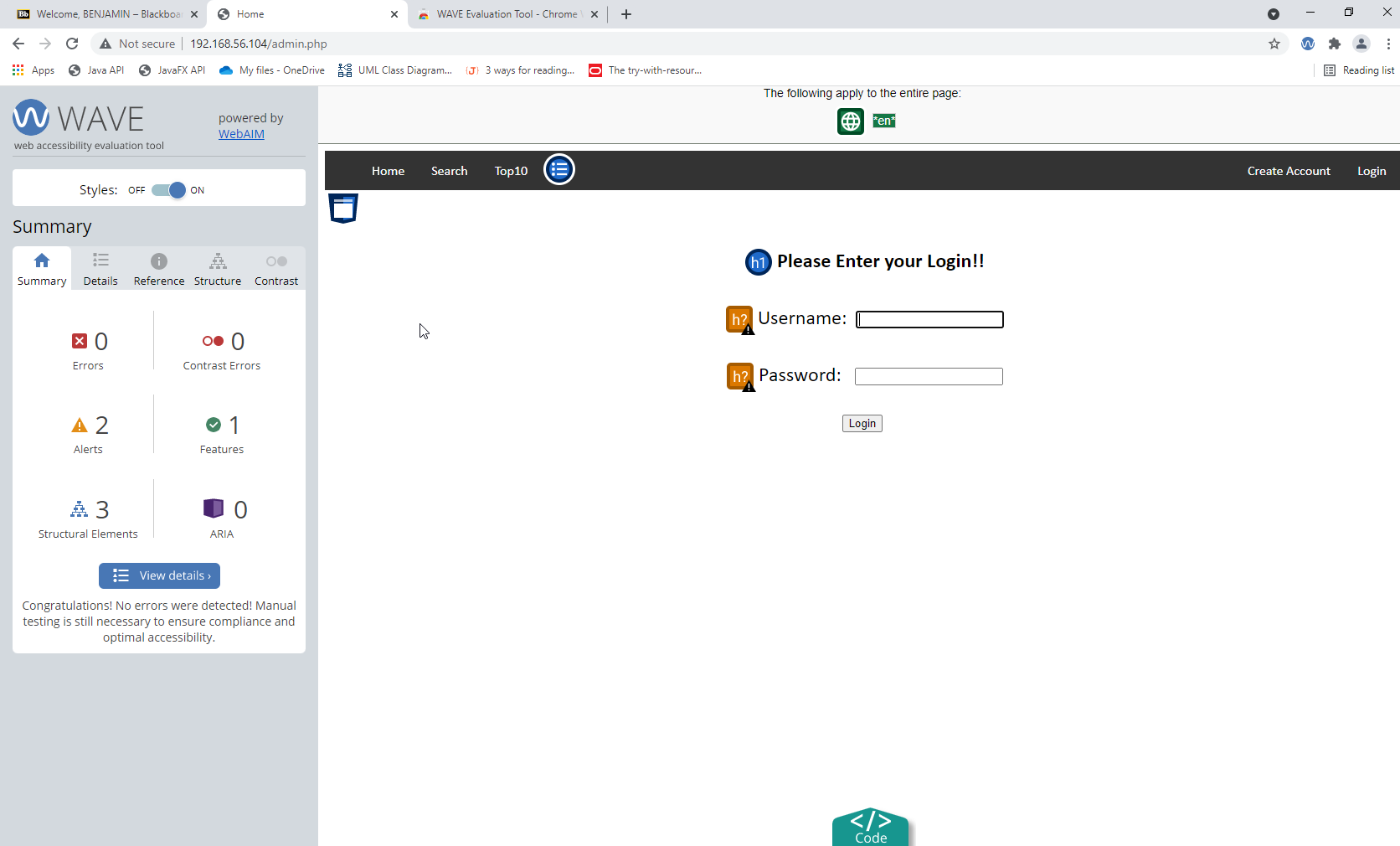
|  |  |  |  |
| --- | --- | --- | --- |
| **Test** | **Expected Output** | **Actual** | **Reference** |
| Home page WCAG | WCAG Accessible | As Expected | Figure 28 |
| Search page WCAG | WCAG Accessible | As Expected | Figure 29 |
| Login page WCAG | WCAG Accessible | As Expected | Figure 30 |
| Top10 page WCAG | WCAG Accessible | As Expected | Figure 31 |
| CreateAccount page WCAG | WCAG Accessible | As Expected | Figure 32 |



Figure



Figure



Figure



Figure



Figure

***End of Assessment***