RAD

AT2

Blayde Dietsch, Noah Jackson and Mitchel King

2019

Table of Contents

[1 Sprint One 3](#_Toc24713723)

[1.1 Source Control 3](#_Toc24713724)

[1.2 Project management plan 4](#_Toc24713725)

[1.3 Software Development Testing Plan 4](#_Toc24713726)

[1.3.2 Test Methodology 4](#_Toc24713727)

[1.3.3 Test deliverables 5](#_Toc24713728)

[1.3.4 Resource and environment needs 5](#_Toc24713729)

[1.4 Analysis documentation 6](#_Toc24713730)

[1.4.1 CITE business rules for software development 6](#_Toc24713731)

[1.4.2 CITE managed services quality assurance 7](#_Toc24713732)

[1.4.3 Acme entertainment development requirements 7](#_Toc24713733)

[1.5 Testing 8](#_Toc24713734)

[1.5.1 Testing Procedures 8](#_Toc24713735)

[1.5.2 Integration Testing 9](#_Toc24713736)

[1.5.3 System Testing 9](#_Toc24713737)

[1.5.4 Final Testing 9](#_Toc24713738)

[2 Sprint two 10](#_Toc24713739)

[2.1 Software review plan 10](#_Toc24713740)

[2.1.1 Meeting minutes 10](#_Toc24713741)

[2.1.2 Performance report 11](#_Toc24713742)

[2.2 Source Control 12](#_Toc24713743)

[2.3 Project Management Plan 12](#_Toc24713744)

[2.4 Software Development Testing Plan 12](#_Toc24713745)

[2.4.1 Test Methodology 12](#_Toc24713746)

[2.4.2 Test Deliverables 12](#_Toc24713747)

[2.4.3 Resource and Environment needs 12](#_Toc24713748)

[2.4.4 Terms/Acronyms 12](#_Toc24713749)

[2.5 Testing 12](#_Toc24713750)

[2.5.1 Testing Procedures 12](#_Toc24713751)

[2.5.2 Integration Testing 12](#_Toc24713752)

[2.5.3 System Testing 12](#_Toc24713753)

[2.5.4 Final Testing 12](#_Toc24713754)

[2.6 Demonstration 12](#_Toc24713755)

# Sprint One

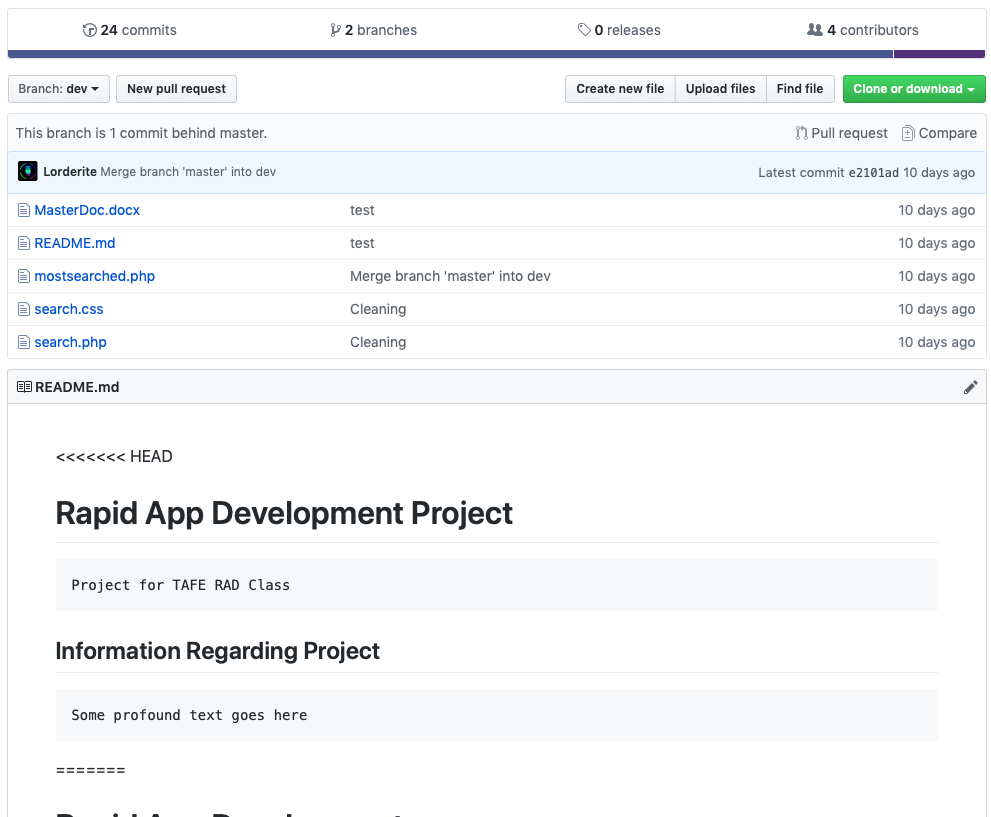
## Source Control

Source control for this project is GitHub, working with a local copy each, and merging with the central repo when changes are completed.

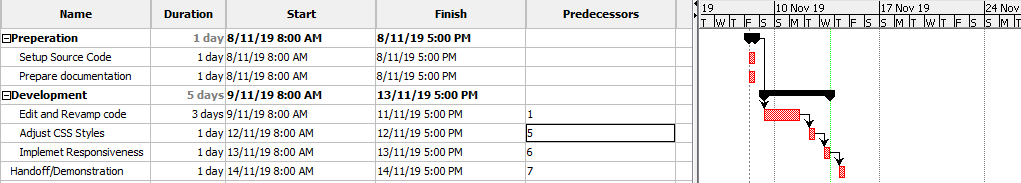
GitHub lets you see different versions of the source code and the differences between copies, issues and bugs can be created and tracked and assigned to team members.

Git Hub: <https://github.com/mkjking/RAD>

Snapshots;



## Project management plan



## Software Development Testing Plan

#### Scope

The scope of this testing plan is focused on the responsive design. The responsive design should be done before the end of the sprint so the testing plan for this sprint is to get the responsive design working.

#### In Scope

* Basic Functionality (searching, most searched)
* Responsive Design

#### Out of Scope

* Additional Features

#### Quality Objectives

This sprint requires that the responsive design fully works. Therefore, our Quality Objectives are:

* A fully functional responsive design implemented by the Second Sprint start.
* All basic functionality (searches etc.) is fully implemented and works as intended.

### Test Methodology

#### Overview

Due to the nature of responsive design, the testing will be done on multiple devices at various resolutions and browser sizes. The focus of the testing will be making sure there are no major issues with the responsive design, and that the page responds as intended.

#### Test levels

**Unit Test**

Each individual component should be tested separately from the whole project first, making sure that the code intention matches the result. This means that components such as image resizing, table resizing, font size changes and more should be tested individually first.

**Integration Test**

The tested components, i.e. image resizing, should then be integrated into the main system and tested to make sure the intended results do not change due to change in environment variables.

**System Test**

The entire system should be tested to see if the new integration has affected any other functionality, for example does the movie search still work, can you still search by rating etc.

**Acceptance Test**

After all components have been added and finalised, a final testing session should be done to make sure that the system is in acceptable state and works as intended.

#### Bug triage

*The bug triage shall use the following format:*

1. **Initial screenshot and description**

Post a screenshot of the issue occurring, and describe what the issue is, how it occurred and on what system it occurred.

1. **Confirmation of issue**

Recreate the issue and test in another environment/system to confirm the issue and whether it is system dependent.

1. **Issue address**

Address what fixes you have put in place/what changes you have made to fix the issue.

1. **Conclusion of issue**

Give a final statement on the condition of the bug and whether it was fixed or needs addressing later down the pipeline.

#### Test completeness

To finalise any level of testing, the goal requirements being solved must be achieved and/or furthered towards. This requires proper reading of the software requirements to make sure the product fits those requirements.

### Test deliverables

The first sprint has only two deliverables, to be shown at the end of the sprint to the relevant stakeholder:

* Basic functionality of the website (searching for the movies, displaying a table etc.)
* Working responsive design across the product.

These should be tested before showing the stakeholder, and then may be presented.

### Resource and environment needs

#### Testing tools

The web program is being hosted on a local server, notably XAMPP Apache server. Any development features of the browsers that the product is being tested on (Chrome, Firefox, Edge) to test the product.

#### Test environment

The product will be tested multiple popular browsers, namely Chrome and Firefox, on multiple systems.

|  |  |
| --- | --- |
| **Environment Variable Type** | **Environment Variable Used** |
| Operating System (64 Bit) | Windows 10, Mac OS |
| Web Browser | Chrome, Firefox, Edge |
| Resolutions | 1920x1080, 1366x768, 1280x720, 640x480 |

## Analysis documentation

### CITE business rules for software development

CITE has a few standards and rules it follows for development

Coding standards;

* Naming conventions
* File naming and organization
* Formatting and indentation
* Comments and documentation
* Classes, Functions Interfaces
* Pointer and reference usage
* Testing

IP and Security;

* Intellectual property protection
  + Your software assets for your business have economic value, this value depends on the intellectual property rights involved. CITE uses industry best practices, all appropriate legal and physical guidelines to protect to protect your IP.
* Information Non-disclosure
  + A non-disclosure agreement is signed with all customers to provide peace of mind with all security and confidentiality issues.
  + All employees also sign a confidentiality agreement and must acknowledge understanding of CITE security policies.
* Distributed environment
  + All our online services run on CITE’s global network, not a local repository or server.
  + All data is distributed among this shared infrastructure with CITE server locations all around the world.
* Title transfer
  + We understand you want full ownership of the IP that we may develop.
  + CITE have a system in place that all employees and contractors must sign. This system enables CITE to own all IP as soon as it is created, waiving all moral rights to the individual.
* Data destruction
  + When data or services are retired from our system, we make sure the disks follow the data destruction procedure.
  + The procedure involves wiping the disks with the IT security team’s policies. If issues arise from wiping, then the disks are physically destroyed.
  + All data is destroyed if the disks are to leave CITE property.
* Physical security
  + Swipe cards
  + Role based access
  + CCTV
  + Fire protection
  + UPS systems, backup power

### CITE managed services quality assurance

Cite has a complex quality management system which follows a strict set of tasks and procedures to ensure quality is always maintained.

QMS;

* Procedures and regulations based on industry standards and best practices.
* Monitoring throughout lifecycle of the project to ensure consistent compliance with the regulations and practices.
* Product quality and verification against client needs and expectations.
* Create a collaboration environment for the project team to communicate efficiently
* Quality planning
  + CITE services makes custom plans that inform the client of standards, regulations, procedures guidelines and tools
* Quality assurance
  + CITE have processes in place to regularly check how the project is going to make sure it is meeting customer requirements.
* Quality control
  + Performance data is taken to decide whether the code is efficient or defective, to make sure quality is always high.
* Independent QA department
  + The QA department that perform these processes and tests are an independent branch of CITE, with its own dedicated QA engineers. Having a QA structure like this allows CITE to be extremely flexible with many ongoing projects.

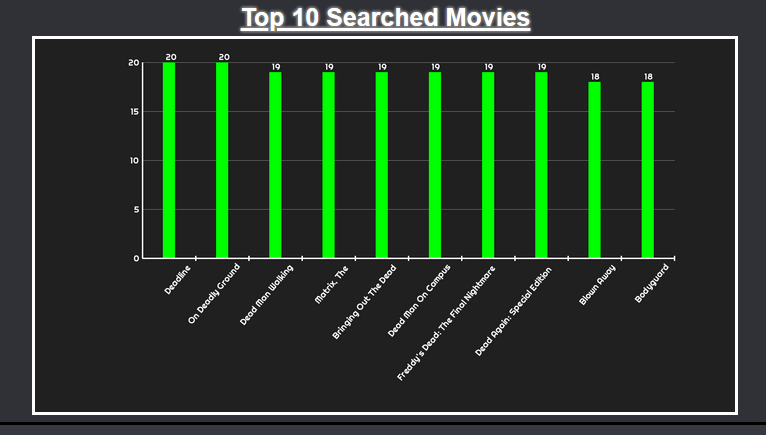
### Acme entertainment development requirements

* Deploy a movie database application
* RAD agile methodology
* Movie website must be able to search
* Movie website must be able to display top 10
* Movie website must be responsive to different screen sizes

## Testing

### Testing Procedures

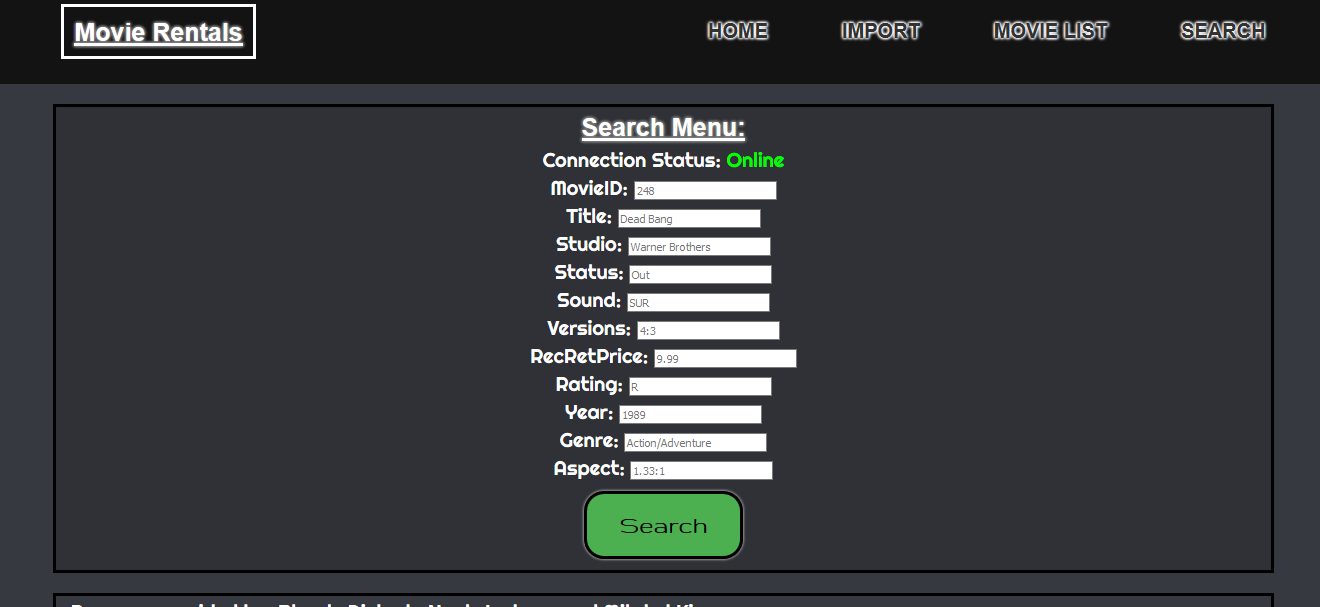
#### Top 10 Searched Function



Currently the top10searched.php function works as intended on all tested resolutions referenced in the 1.3.4.2 table.

|  |  |  |
| --- | --- | --- |
| **TESTED** | **WORKING** | **COMMENT** |
| 1920x1080 Resolution | Yes | Different browser sizes work as well as custom resolution |
| 1280x720 Resolution | Yes |  |
| 1366x768 Resolution | Yes |  |
| Graph Drawing | Yes | Font issues fixed before testing |

#### Search Function



|  |  |  |
| --- | --- | --- |
| **TESTED** | **WORKING** | **COMMENT** |
| Table Display | YES | Working in all 4 CSS sizes (1500+px, 1500px, 1000px, 500px) |
| MovieID | YES |  |
| Title | YES |  |
| Rating | YES |  |
| Year | YES |  |
| Database Connection | YES |  |

#### Import Function

|  |  |  |
| --- | --- | --- |
| **TESTED** | **WORKING** | **COMMENT** |
| Database Creation | YES |  |
| Table Fill | NO | Currently empties the database, needs addressing |

### Integration Testing

|  |  |  |
| --- | --- | --- |
| **TESTED** | **WORKING** | **COMMENT** |
| Search | YES | Still works as intended |
| Movie List | YES | Still works as intended |
| Top 10 | YES | Still works as intended |
| Import | NO | If pressed will break database connection |

### System Testing

|  |  |  |
| --- | --- | --- |
| **TESTED** | **WORKING** | **COMMENT** |
| Search | YES | Does not break other system functionality |
| Movie List | YES | Does not break other system functionality |
| Import | NO | Will break other system functionality, needs addressing |

### Final Testing

The basic functionality of the product works as intended, and the functions work and do not break the system with the exception of import.php, which needs addressing before the end of next sprint.

# Sprint two

## Software review plan

### Meeting minutes

* Opening

Meeting was held 14th November in a Tafe campus lecture room. The meeting was called by the client in order to present Sprint one process, and to discuss Sprint two requirements.

* Present

Blayde, Noah, Mitch, and the Client Mr. A. Samway

* Absent:

NA

* Approval of Agenda

Agenda is approved by the development team and the client.

* Business from the Previous Meeting

Discuss requirements for responsive HTML and CSS design, and the sites goal and description.

* New Business

New meeting was discussed to include functionality for Email subscription services.

Users must be able to subscribe to newsletters and information, must have admin for only administrators to revoke access to the subscriptions.

Users must be able to request resignation from subscription services, prompting a request email sent to the admin account. Once the admin has approved, confirmation email will be sent to the user, asking to confirm action.

* Additions to the Agenda

### Performance report

This application is developed using PHP and HTML, both are different languages serving different sides of the application.

* + Performance Tools

Profilers can be used for PHP to accurately measure resources and performance statistics. Every action PHP makes uses resources on the machine, when applications increase in size and become more complex, it’s easy for performance margins to blow out as each inefficiency wherever it may be will stack.

Popular profiling tools for PHP standard profiling are Xhprof, and Xdebug, standard profilers record stack traces and give snapshots of hardware usage, Howlong the program spends at each line of code and counts method calls etc.

Tracing profiler tools like Z-Ray, are much more lightweight and allow you to debug while you code. Setting breakpoints and watching data and how the program responds. Tracing profilers are intended to catch errors during application development rather than end product testing, unlike standard profilers, tracing profilers do not offer server usage statistics like CPU and memory.

APM(Application performance monitoring) tools offer more in depth analysis of the entire stack development. APM tools run all the time on the server and provide an interface of statistics for the user. Website owners and developers can view performance data, and customer data of all actions, links, or whatever tasks the user may use on the application.

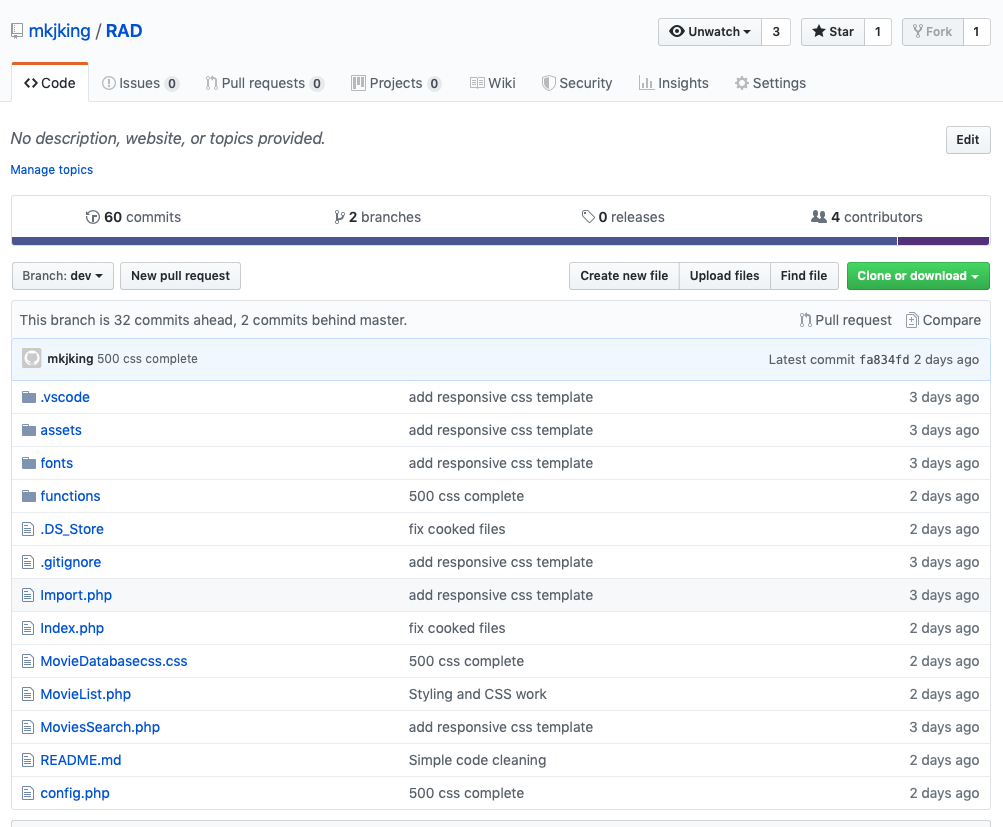
APM’s are costly and suited to larger projects when non-stop detailed data recording is needed.

* + Code optimisers

Code optimisers are simple tools that usually comply with some set of standards relative to the language. They let users insert code to apply these standards and output the modified code. Code optimisers are essential to maintain standards while also improving performance.

Code optimisers can also just tidy the structure, spacing and readability of the code. Implementing and researching good practice coding procedures to reduce loading times, like handling images correctly and only when needed etc.

## Source Control



## Project Management Plan

## Software Development Testing Plan

#### Scope

The scope of this testing plan is focused on the creation of membership and administrator portals. This plan will outline the functions and features required and follow testing for each.

#### In Scope

* Membership email subscription service
* Administrative functions

#### Out of Scope

* Additional Features

#### Quality Objectives

This sprint requires that the membership portal and administrative functions fully work. Therefore, our Quality Objectives are:

* A fully functioning membership portal with the ability to subscribe to emails, and request unsubscribe
* A fully functioning administrative portal with the ability to receive emails from members requesting to be unsubscribed
* All functions and design elements to achieve this will be fully implemented

### Test Methodology

#### Overview

Testing this plan will require multiple devices and multiple email accounts. Due to the nature of these test procedures there are lots of variations of software that will need to be considered.

#### Test levels

**Test Procedures**

Each individual component should be tested separately from the whole project first, making sure that the code intention matches the result. This means that components Administrator functions, and membership functions will be tested first.

**Integration Test**

The tested components, i.e. email generation, should then be integrated into the main system and tested to make sure the intended results do not change due to change in environment variables.

**System Test**

The entire system should be tested to see if the new integration has affected any other functionality, for example does the movie search still work, can you still search by rating etc.

**Acceptance Test**

After all components have been added and finalised, a final testing session should be done to make sure that the system is inacceptable state and works as intended.

#### Bug triage

*The bug triage shall use the following format:*

1. **Initial screenshot and description**

Post a screenshot of the issue occurring, and describe what the issue is, how it occurred and on what system it occurred.

1. **Confirmation of issue**

Recreate the issue and test in another environment/system to confirm the issue and whether it is system dependent.

1. **Issue address**

Address what fixes you have put in place/what changes you have made to fix the issue.

1. **Conclusion of issue**

Give a final statement on the condition of the bug and whether it was fixed or needs addressing later down the pipeline.

#### Test completeness

To finalise any level of testing, the goal requirements being solved must be achieved and/or furthered towards. This requires proper reading of the software requirements to make sure the product fits those requirements.

### Test Deliverables

The second sprint has only two deliverables, to be shown at the end of the sprint to the relevant stakeholder:

* Membership functionality of the website
* Administrator functionality of the website

These should be tested before showing the stakeholder, and then may be presented.

### Resource and Environment needs

#### Testing tools

The web program is being hosted on a local server, notably XAMPP Apache server. Any development features of the browsers that the product is being tested on (Chrome, Firefox, Edge) to test the product.

#### Test environment

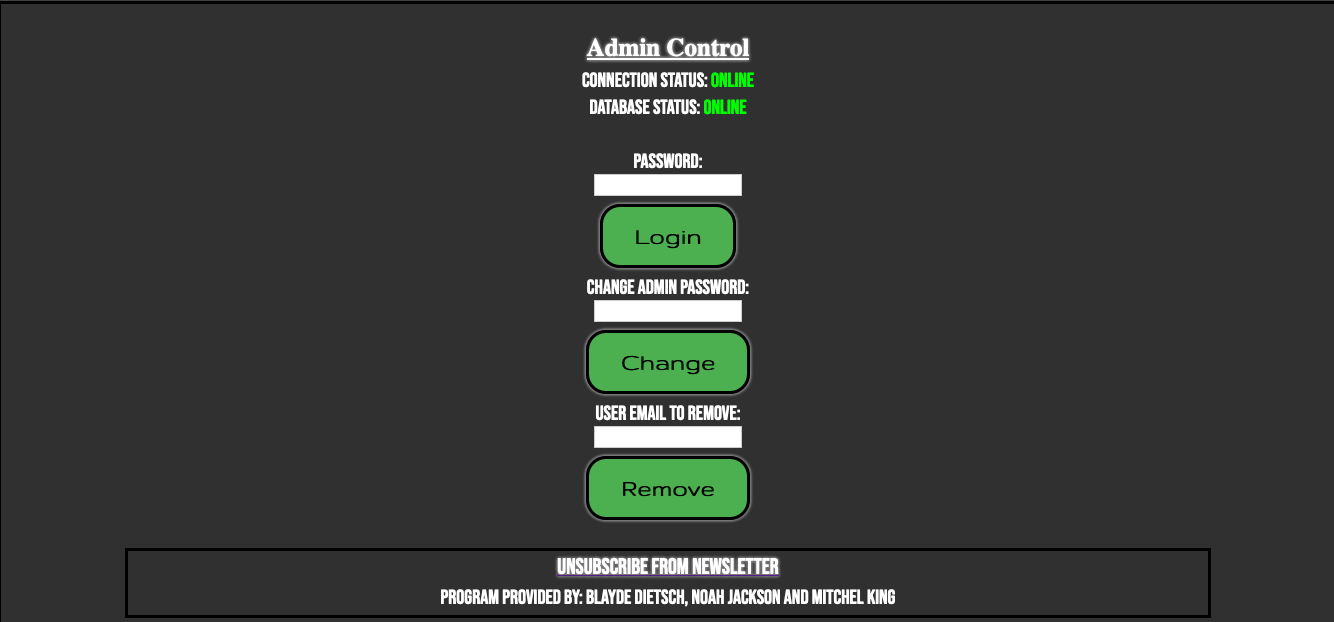
The product will be tested multiple popular email providers, namely Gmail and Outlook, on multiple systems.

|  |  |
| --- | --- |
| **Environment Variable Type** | **Environment Variable Used** |
| Operating System (64 Bit) | Windows 10, Mac OS |
| Email provider | Gmail, Outlook, iCloud |
| Hardware | Desktop, Mobile |

## Testing

### Testing Procedures

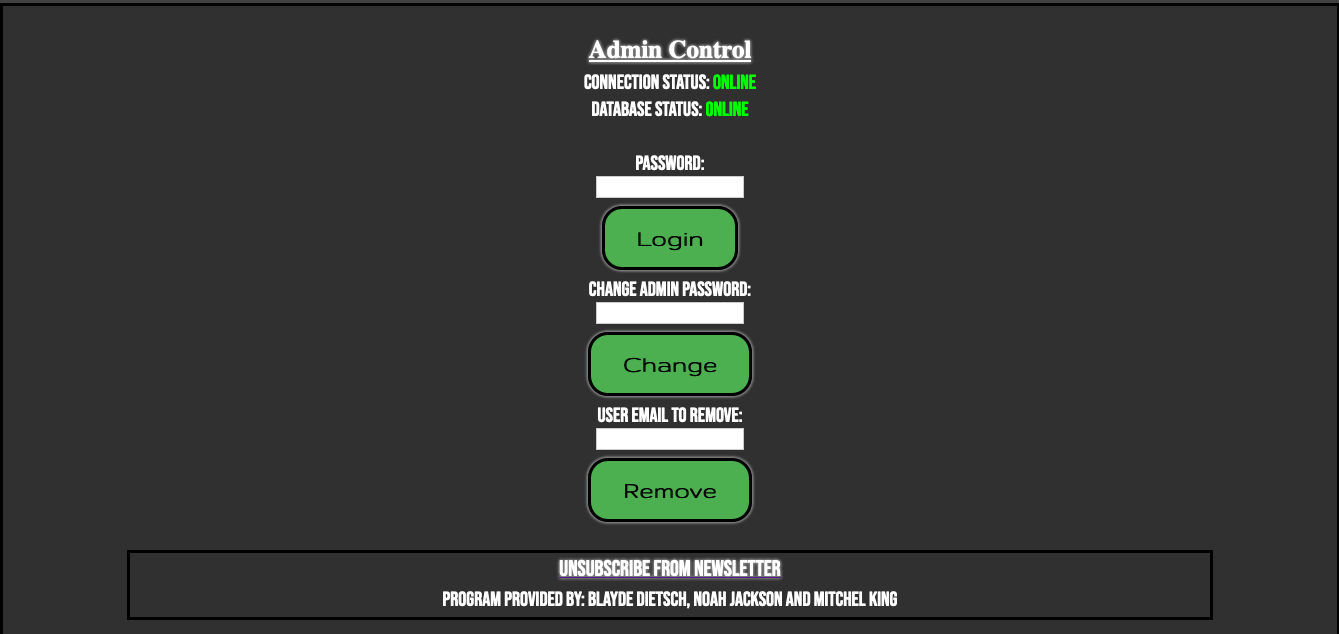
#### Admin login



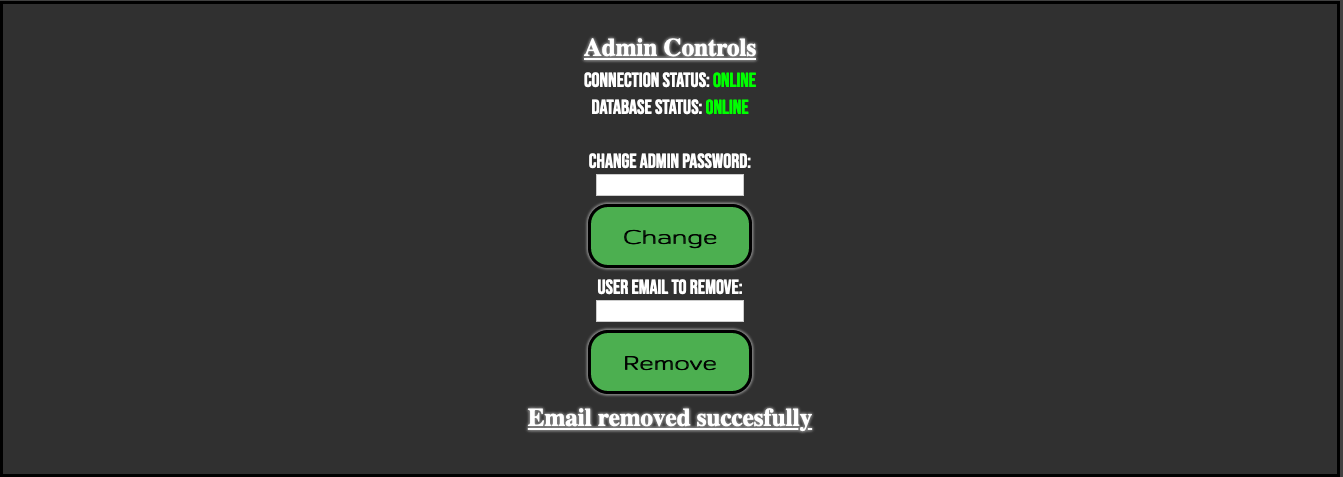
Admin Login page works as intended, one admin account exists in the database and password is checked against this on attempted login;

|  |  |  |
| --- | --- | --- |
| **TESTED** | **WORKING** | **COMMENT** |
| Random input | Yes | Fails all incorrect user input, anything can be entered without errors |
| Correct password | Yes | Logs in and displays next menu |
| Firefox | Yes |  |
| Safari | Yes |  |

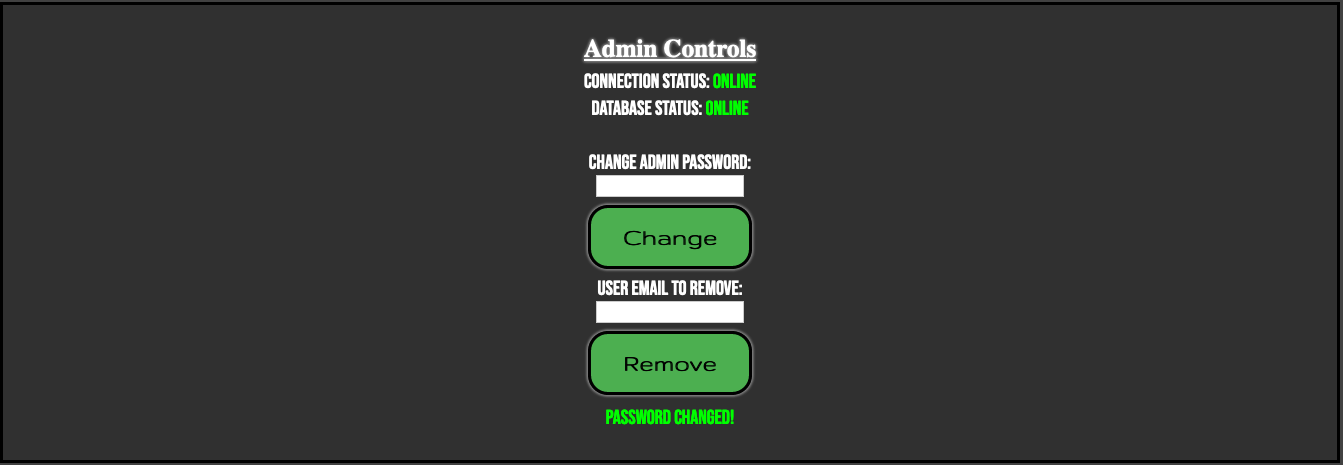
#### Admin Control



Once logged in, AdminControl.php is displayed showing various administrator options, change admin password and remove users from mailing list.



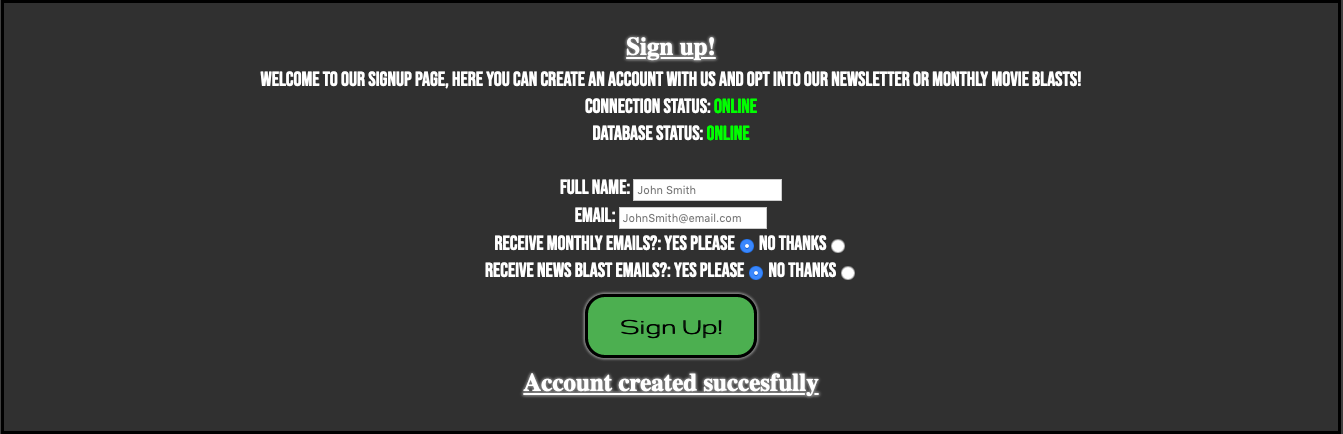
Email successfully removed from the database.



|  |  |  |
| --- | --- | --- |
| **TESTED** | **WORKING** | **COMMENT** |
| Random user input | Yes | Throws error email doesn’t exist |
| Change Admin Password | Yes | Once logged in, changes the admin password to sent string |
| Remove user from email list | Yes | If email entered exists, the user is removed from the db |
| Firefox | Yes |  |
| Safari | Yes |  |

Admin password successfully changed.

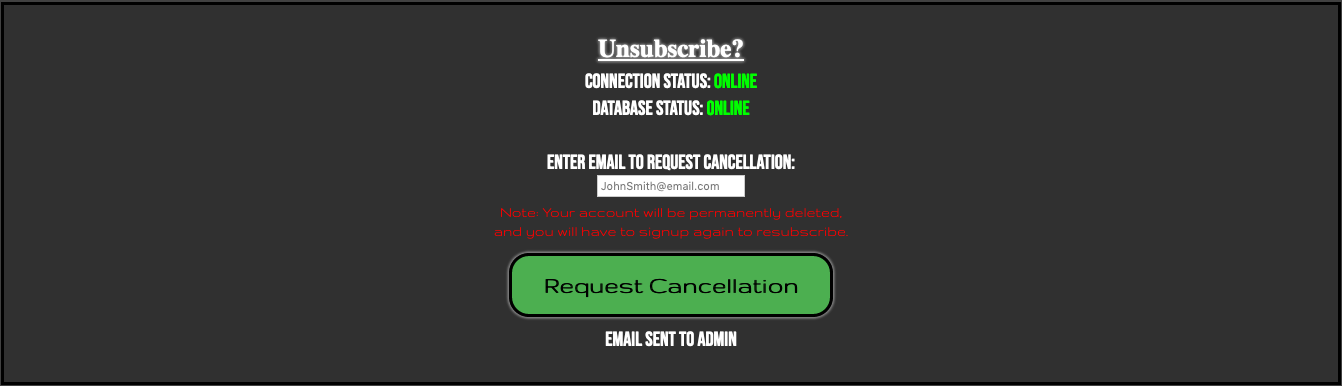
#### Signup



Sign up page for users to enter name, email, and which subscription they prefer.

|  |  |  |
| --- | --- | --- |
| **TESTED** | **WORKING** | **COMMENT** |
| Random user input | Tes | Throws error if email is not in the correct format |
| Monthly emails | Yes | Radiobox is interpreted as a 1 or 0, and sent to the db |
| Blast emails | Yes | Radiobox response is sent to the db |
| User created | Yes | User is added to the db with correct information |
| Firefox | Yes |  |
| Safari | Yes |  |

#### Unsubscribe



Unsubscribe link is available on every page for the user to request removal from the mailing list.

|  |  |  |
| --- | --- | --- |
| **TESTED** | **WORKING** | **COMMENT** |
| Incorrect input | No | Displays error if email doesn’t exist, but also shows ‘email sent to admin’ |
| Email sent to admin account | No | Emails are sent to admin regardless if user requesting exists or not |
| Firefox | Yes |  |
| Safari | Yes |  |

### Integration Testing

|  |  |  |
| --- | --- | --- |
| **TESTED** | **WORKING** | **COMMENT** |
| Admin Login | YES | Still works as intended |
| AdminControl | YES | Still works as intended |
| Signup | YES | Still works as intended |
| Unsubscribe | No | Generates emails whether user account exists or not |

### System Testing

### Final Testing

## Demonstration