Master Documentation

Handover

Team F

Table of Contents

[Source Control 3](#_Toc44579850)

[Software Review Report 4](#_Toc44579851)

[Quality Assurance 4](#_Toc44579852)

[Quality Control 5](#_Toc44579853)

[IDE 5](#_Toc44579854)

[Screen Reader 5](#_Toc44579855)

[Test Procedure 6](#_Toc44579856)

[Methodology 6](#_Toc44579857)

[Test method 7](#_Toc44579858)

[Testing process 7](#_Toc44579859)

[Recommendation 7](#_Toc44579860)

[New Prototype Option 7](#_Toc44579861)

[Current Prototype 7](#_Toc44579862)

[Coding Comments 8](#_Toc44579863)

[Test Plans 9](#_Toc44579864)

[Project Management 10](#_Toc44579865)

[Recommendation’s Summary 10](#_Toc44579866)

[Requirements 11](#_Toc44579867)

[Master Documentation for Each Sprint 13](#_Toc44579868)

[Project Management Plan 14](#_Toc44579869)

[Software Testing Plan 15](#_Toc44579870)

[Scope 15](#_Toc44579871)

[In Scope 15](#_Toc44579872)

[Quality Objective 15](#_Toc44579873)

[Roles and Responsibilities 15](#_Toc44579874)

[methodology 16](#_Toc44579875)

[Agile model and RAD model 16](#_Toc44579876)

[Test method 16](#_Toc44579877)

[Test Completeness 17](#_Toc44579878)

[Resource & Environment Needs 17](#_Toc44579879)

[Testing Tools 17](#_Toc44579880)

[Test Environment 17](#_Toc44579881)

[Test Deliverables 18](#_Toc44579882)

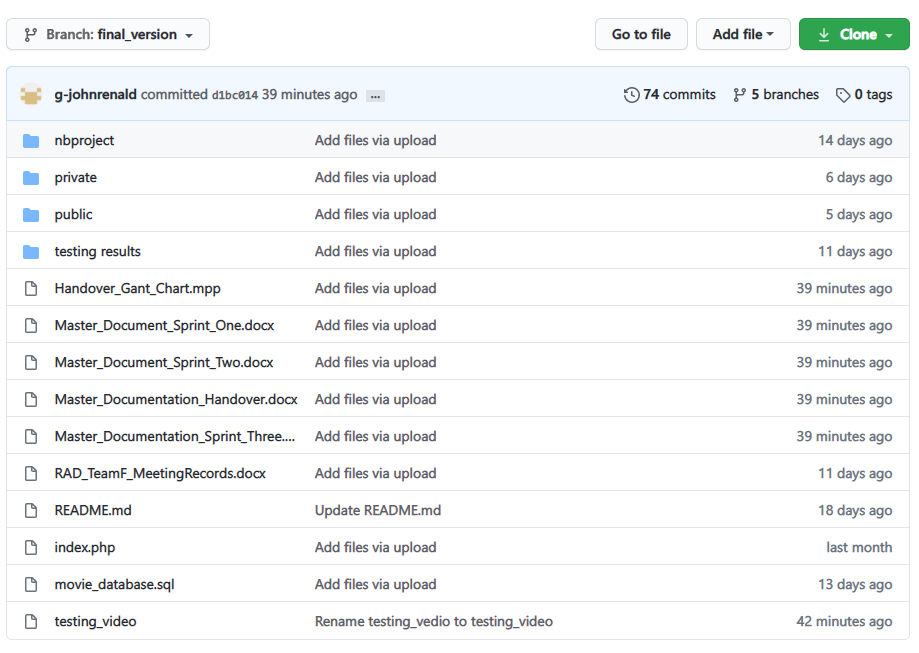
[Test case: 18](#_Toc44579883)

[Report 23](#_Toc44579884)

[Glossary of terms 24](#_Toc44579885)

# Source Control

We created a new branch for the final version of our software. We included Gantt chart of our project management plan, master documents for each sprint, and master document of handover.



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version #** | **Implemented**  **By** | **Revision**  **Date** | **Approved**  **By** | **Approval**  **Date** | **Reason** |
| 1.0 | *<Cheng Liang Chen>* | *<28/05/2020>* | *< Team F >* | *<04/06/2020>* | Implement responsive web site |
| 2.0 | *<Cheng Liang Chen>* | *<04/06/2020>* | *< Team F >* | *<11/06/2020>* | Subscribe page and send email to subscribers |
| 3.0 | *<Cheng Liang Chen, John>* | *<11/06/2020>* | *< Team F >* | *<18/06/2020>* | Movie ranking system and admin portal |
| 4.0 | *<Cheng Liang Chen>* | *<18/06/2020>* | *< Team F >* | *<25/06/2020>* | WCAG requirements |

This project will use GitHub which can help developers easily to manage software’s version to store and distribute every version. The history of application version will be recorded in GitHub repository. <https://github.com/SMT-Diploma-of-Software-Development/Movie_Theater_Database>

# Software Review Report

## Quality Assurance

Our team has maintained the quality of the product by following the RAD cycle and consistently conducting Whitebox testing at the end of phase of each iteration. White box testing is tested by who understand the structural or implementation of application. Tester understand how the application works. Usually, white box testing tests application’s internal structure or working. Testers will choose inputs to exercise path through the code and determine the expected outputs.

In addition, to ensure that our product is accessible and WCAG friendly, we strictly follow the official guideline of WCAG and conduct Whitebox testing consistently. The guideline is consisted of five major elements which are listed below:

1. **Perceivable**

* Provide text alternatives for non-text content.
* Provide captions and other alternatives for multimedia.
* Create content that can be presented in different ways, including by assistive technologies, without losing meaning.
* Make it easier for users to see and hear content.

1. **Operable**

* Make all functionality available from a keyboard.
* Give users enough time to read and use content.
* Do not use content that causes seizures or physical reactions.
* Help users navigate and find content.
* Make it easier to use inputs other than keyboard.

1. **Understandable**

* Make text readable and understandable.
* Make content appear and operate in predictable ways.
* Help users avoid and correct mistakes.

1. **Robust**

* Maximize compatibility with current and future user tools.

1. **Pointer Accessible**

* Make it easier for users to operate functionality through various inputs beyond keyboard.

We use these guidelines as a checklist to make our product accessible. We clarify each element carefully and modify as needed with explanations that justify the modifications.

As our product does not contain non-text content, we assume that we are not required to implement text alternatives which means that we already met the first checklist. We mainly focus on adjusting our User Interface and confirming if our product can be read using a Screen Reader.

## Quality Control

We utilized the processes of coding and testing by using IDE and screen reader software respectively.

### IDE

**NetBeans**

A full IDE (Integrated Development Environment) which means that it has extensive and intelligent development features, such as seamless software modularization, language extensions and easier ways to integrate libraries into a project.

### Screen Reader

**NonVisual Desktop Access**

NVDA is a free and open-source screen reader for Microsoft Windows. Screen readers enable people with blindness, visual impairments, and/or reading disabilities to use computers and browse websites by reading out loud the text and alternative text on the screen.

We use NVDA to conduct a user test to confirm if our product could be read without any errors and usable.

## Test Procedure

### Methodology

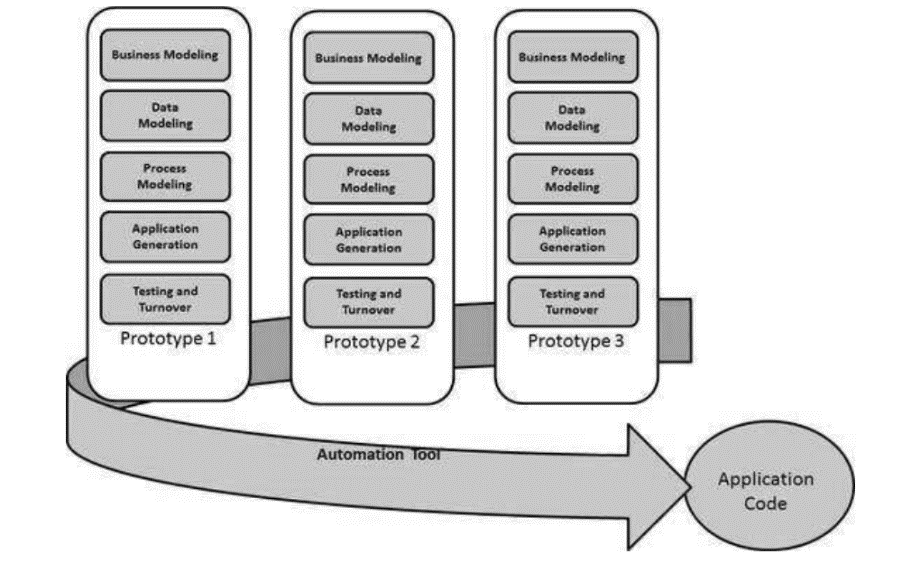
This project will adopt Agile model and RAD model, testing phase will be in the end of each iteration.

**Agile model and RAD model**

Agile model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations. Each iteration (Requirement->Design->Development->**Testing**->Evaluation) typically lasts from about one to three weeks. **At the end of the iteration, a working product is displayed to the customer and important stakeholders**. (tutorialspoint, 2019)

Hence testing phase will execute in each interation, it ensure bug can be found early.

Rapid Application Development (formerly known as Rapid Application Development, abbreviated as RAD) is a methodology for software development. RAD methodology is quite different with traditional SDLC, traditional SDLC emphasize requirement analysis before code start. Changing requirement may not so feasible. However**, RAD emphasize the iterative and incremental delivery of working models to the customer rapidly and allow customer give feedback**.



### Test method

**White box testing:**

White box testing or structural testing is tested by who understand the structural or implementation of application. Tester understand how the application works. Usually, white box testing tests application’s internal structure or working. Testers will choose inputs to exercise path through the code and determine the expected outputs.

### Testing process

In testing phase, tester need to

* Communicate with application developer and understand how the application works.
* Decide testing functional scope and non-functional scope.
* Setting the testing goal or completeness. Does all logic path are coverd and the applicatoin meet client’s requiment?
* Decide what testing tool need to be used and check the testing environment.
* Design testing table can document or record the result.

## Recommendation

### New Prototype Option

The website design and solution may be reimplemented to use website creators/hosting services such as WIX, Squarespace, this will allow the client to modify elements of their website as their business evolves with user friendly tools which require no technical or programming knowledge. Comes at a cost of subscription and dependence of the service provider but enables user friendly website design/creation tools for the client.

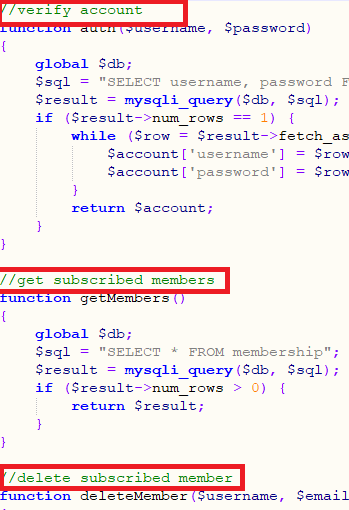
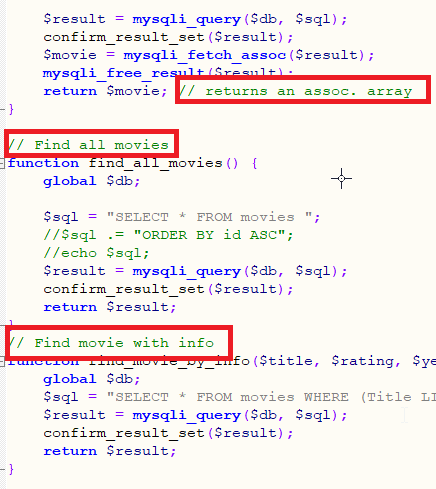
### Current Prototype

At this current moment the website requires someone with programming/technical knowledge to modify and enhance it, using Xampp for simplifying networking protocols such as HTTP and SQL it is very easy to code as most work can be done without being required to consider various technical networking components, simply place a ‘html/PHP’ page called index within the htdocs folder of XAMPP and one is ready to begin developing from the Home page, Xampp also provides a MySQL database which is fully configurable via a GUI interface page PhpMyAdmin.

### Coding Comments

Coding comments have been placed where they are deemed appropriate by the programmers involved and are adjusted for clarity during team meetings/review. It is recommended for anyone who revisits the code of the project to revise these comments as they make modification as the comments are aimed to guide anyone in pursuit of understanding the code, by providing context and clarity for the various elements which may be at play in the block.

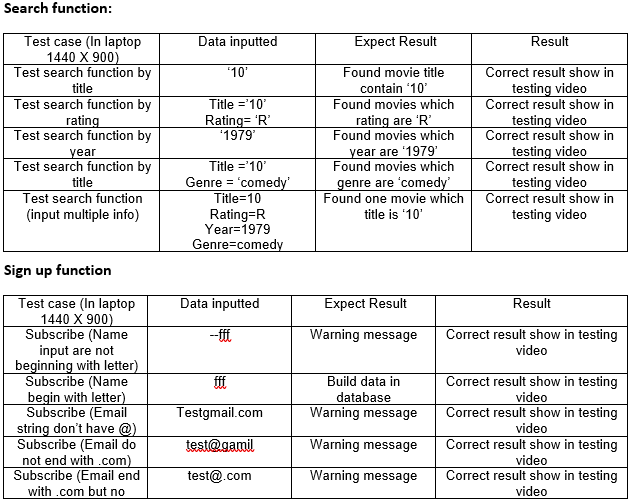
Example of coding comments clarifying processes.

### Test Plans

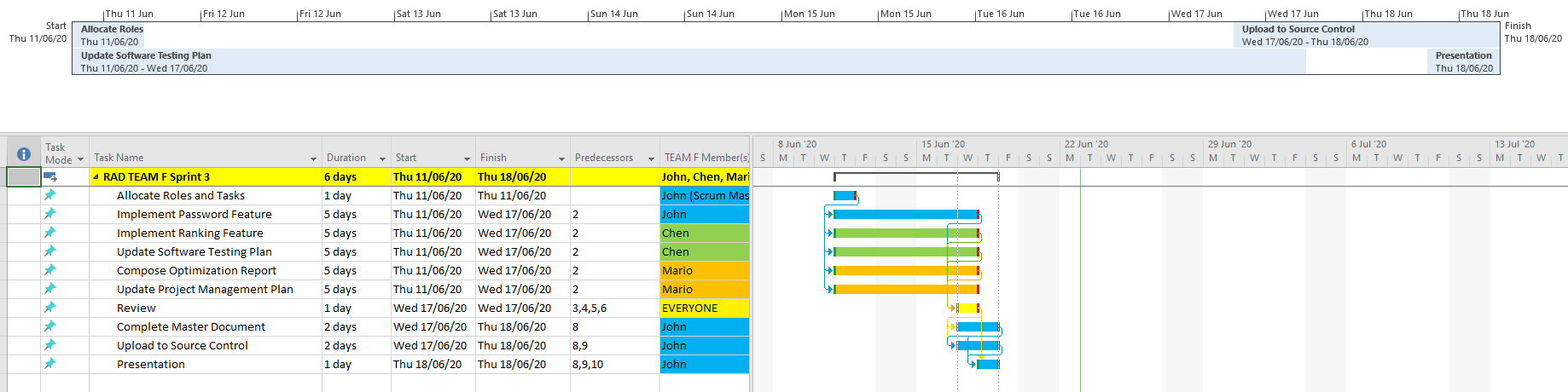
A testing plan is attached and is updated at every Sprint of development for this project, it is recommended that this practice be continued in the same synchronous manner. This testing plan aids in keeping the functionality of the project at an optimal level by revising priority elements/processes that were already implemented after modifying or including new elements in the project.

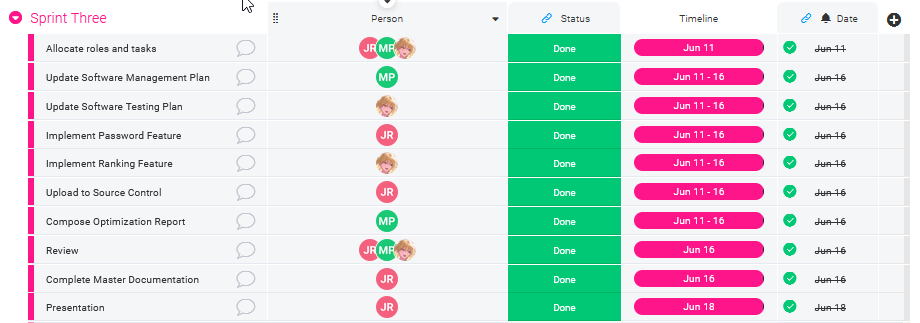
Examples of tests from our testing plan.

### Project Management

For the implementation of all sprints project management kept all team members coordinated with another and on-track with tasks. During modification and refinement, it is recommended that the team recruited for refinements do the same in order to keep track and illustrate to the client the timeline of events taking place. Tools that can be used for this process include Trello, Monday.org and Microsoft Project.

Examples of development plans, extrapolated from TEAM F’s sprint 3.  




### Recommendation’s Summary

Clear, concise and most importantly up-to-date documentation is the most important factor in keeping future endeavors of modification possible and easy. In most of my recommendations it is specified to keep current and update the comments and testing plan as these enable any other programmer who may take up this body of work to understand what is already implemented. An “out-of-the-box” solution has also been provided in any case that the client may want to manage and modify their website personally with website hosting/creation services such as WIX, Square Space or Weebly.

## Requirements

The following table illustrates the mapping of the requirements provided to TEAM F by the client throughout all development phases and the solutions that they have implemented in response.

|  |  |  |  |
| --- | --- | --- | --- |
| Dev Phase | Requirements | Solutions | Client Feedback |
| Sprint 1 | * Multi-Platform Report. * Responsive or Adaptive implementation for website. * Testing Plan. | * Multi-Platform Report recommends responsive solution. * Responsive Design implemented with HTML5 via <meta> tag. * Testing Plan Created. | Client satisfied with solutions presented and implemented by the team. |
| Sprint 2 | * Email Registration Feature. * Monthly Newsletter. * ‘newsflash’ email newsletter. * Admin User Interface for email removal. | * Google SMTP / free account – for receiving and sending client emails. * PHPMailer – used to send emails via php. * Windows Task Scheduler – used for scheduling monthly newsletters * MySQL database used for storing users. * PhpMyAdmin used for administrating users / removal. | Client representing Acme mostly satisfied, but has mentioned there is an admin UI missing and that use of PhpMyAdmin is not sufficient for satisfying requirements. |

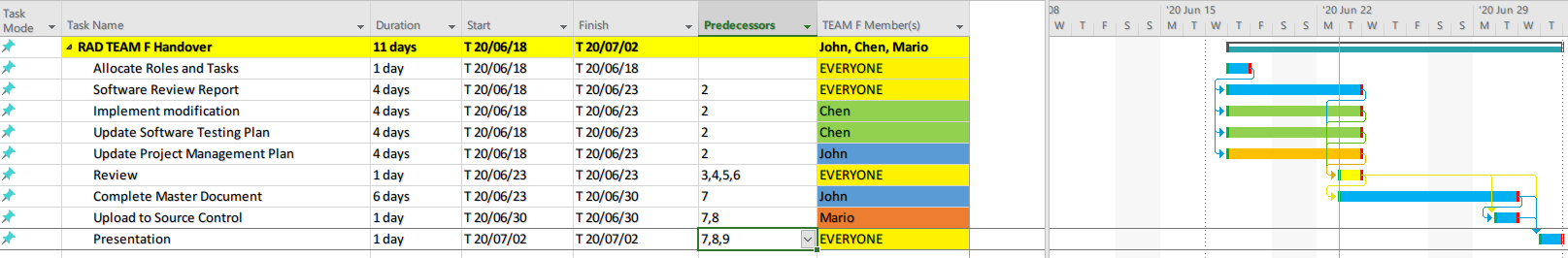
|  |  |  |  |
| --- | --- | --- | --- |
| Sprint 3 | * Security Optimization – Admins with logins. * Movie Rating system * Dynamic refresh of ratings chart. * Optimization + Report. | * Admin/Staff login implemented for User/Email removal. * Admin User interface has login separate from other staff. * All movies shown from searching have a rating feature / button out of 5 stars, calculates average of each movie and plots top 10 on the chart. * Small/simple AJAX implemented for dynamic chart refresh. * Optimized code presentation/beautify, search function, ajax implementation and database ‘root’ password. | Client satisfied with solutions presented and implemented by the team. Sprint 1 ,2 , 3 requirements have been met. |
| Hand-Over | * Final modifications/enhancements * Conformation to Web Content Accessibility Guidelines. * Software Review Report. | * Reviewed Web Content Accessibility Guidelines with the prototype/project – testing/modifing for the blind and the colour blind, as all content is visual. * Software Review Report split into team tasks and then composed after team review. * Finished Report hand in hand with project is studied, revised and modified as appropriate. | No feedback received yet. |

### Master Documentation for Each Sprint

|  |  |
| --- | --- |
| **Sprint** | **Link** |
| Sprint One | [Master\_Document\_Sprint\_One.docx](file:///W:\INTERNET%20DOWNLOADS\3dBuzzTutorials\Master_Document_Sprint_One.docx) |
| Sprint Two | [Master\_Document\_Sprint\_Two.docx](file:///W:\INTERNET%20DOWNLOADS\3dBuzzTutorials\Master_Document_Sprint_Two.docx) |
| Sprint Three | [Master\_Documentation\_Sprint\_Three.docx](file:///W:\INTERNET%20DOWNLOADS\3dBuzzTutorials\Master_Documentation_Sprint_Three.docx) |

# Project Management Plan

Following the sprint three, we continued to using Microsoft Project to maintain the consistency of the process of our project. The actual Microsoft project can be found on GitHub repository or inside the folder which is [Handover\_Gant\_Chart.mpp](file:///W:\INTERNET%20DOWNLOADS\3dBuzzTutorials\Handover_Gant_Chart.mpp).



# Software Testing Plan

## Scope

### In Scope

Defines the features, functional or non-functional requirements of the software that **will be** tested

Non-functional:

* Web site meet WCAG requirements
  + Perceivable(transfer text content into speech for blind)
  + Operable(all functions available from keyboard, help users navigate)
  + Understandable(Make content readable and help users avoid and correct mistakes.)
  + Robust(using IDE and code sniffer to decrease mistake and increase the robust of applicatoin)

### Quality Objective

* Ensure the Application Under Test conforms to functional and non-functional requirements
* Product quality verification and validation to ensure that it complies with clients’ business needs and expectations
* Bugs/issues are identified and fixed before release
* GUI provide user a good experience

### Roles and Responsibilities

* CITE Managed Services QA department: QA testing, configuration manage, risk manage
* Test Manager
* Developers and tester

## methodology

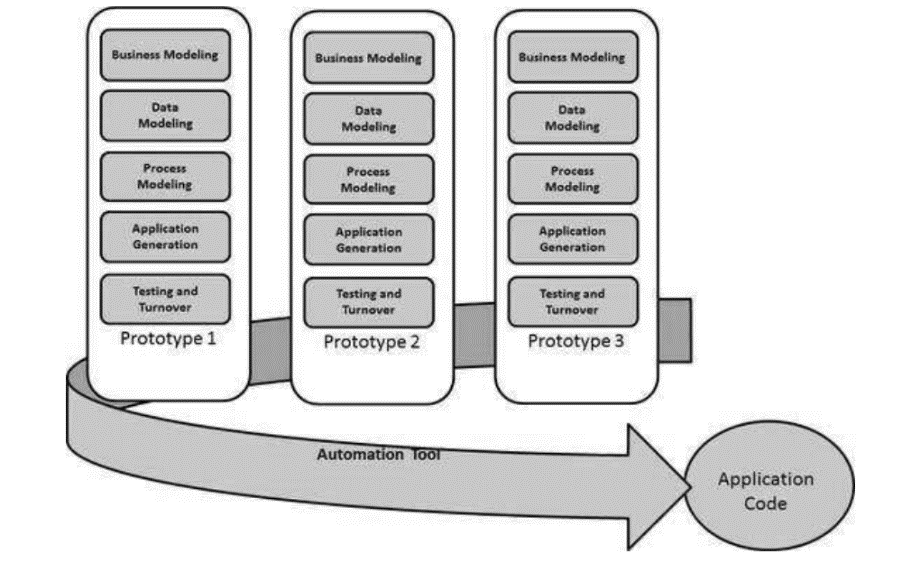
This project will adopt Agile model and RAD model, testing phase will be in the end of each iteration.

### Agile model and RAD model

Agile model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations. Each iteration (Requirement->Design->Development->**Testing**->Evaluation) typically lasts from about one to three weeks. **At the end of the iteration, a working product is displayed to the customer and important stakeholders**. (tutorialspoint, 2019)

Hence testing phase will execute in each interation, it ensure bug can be found early.

Rapid Application Development (formerly known as Rapid Application Development, abbreviated as RAD) is a methodology for software development. RAD methodology is quite different with traditional SDLC, traditional SDLC emphasize requirement analysis before code start. Changing requirement may not so feasible. However**, RAD emphasize the iterative and incremental delivery of working models to the customer rapidly and allow customer give feedback**.



### Test method

White box testing:

White box testing or structural testing is tested by who understand the structural or implementation of application. Tester understand how the application works. Usually, white box testing tests application’s internal structure or working. Testers will choose inputs to exercise path through the code and determine the expected outputs.

### Test Completeness

Criterias that deem testing complete.

* 100% test coverage (all functionalities)
* All Test cases executed manually
* Software are tested under a various of platform.
* All testing result will be documented.

## Resource & Environment Needs

### Testing Tools

* XAMPP or server environment
* Multiplatform simulation environment (Google Developer tools)
* NVDA (screen reader)
* Testing document
* Computer

### Test Environment

* OS: Windows 10 64bit
* Processor: Intel Core i7
* Memory: 16 GB RAM
* Graphics: NVIDIA GeForce GTX 1650
* Storage: 100 GB available space

## Test Deliverables

Here mentions all the Test Artifacts that will be delivered during different phases of the testing lifecycle.

### Test case:

Testing video<https://www.youtube.com/watch?v=uY8qrLqCx2Q>

**Perceivable (transfer text content into speech for blind)**

**Using NVDA screen reader, testing in PC version (depend on different browser, different screen reader different operating order, speech content will be different. The speech content sometimes is uncontrollable)**

|  |  |  |  |
| --- | --- | --- | --- |
| Test case (In laptop 1440 X 900) | Data inputted | Expect Result | Result |
| Tab to home page link | Tab to home page link | Speaker tell blind what to do | Correct result show in testing video |
| Tab to sign up page link | Tab to sign up page link | Speaker tell blind what to do | Correct result show in testing video |
| Tab to search movie page link | Tab to search movie page link | Speaker tell blind what to do | Correct result show in testing video |
| Tab to top ten-page link | Tab to top ten-page link | Speaker tell blind what to do | Correct result show in testing video |
| Tab to contact page link | Tab to contact page link | Speaker tell blind what to do | Correct result show in testing video |
| Tab to start search page link | Tab to start search page link | Speaker tell blind what to do | Correct result show in testing video |
| Tab to title text field of search movie part | Tab to title text field of search movie part | Speaker tell blind what to do | Correct result show in testing video |
| Tab to rating text field of search movie part | Tab to rating text field of search movie part | Speaker tell blind what to do | Correct result show in testing video |
| Tab to year text field of search movie part | Tab to year text field of search movie part | Speaker tell blind what to do | Correct result show in testing video |
| Tab to genre text field of search movie part | Tab to genre text field of search movie part | Speaker tell blind what to do | Correct result show in testing video |
| Tab to search button | Tab to search button | Speaker tell blind what to do | Correct result show in testing video |
| Tab to top ten list button | Tab to top ten list button | Speaker tell blind what to do | Correct result show in testing video |
| Tab to name text field of contact part | Tab to name text field of contact part | Speaker tell blind what to do | Correct result show in testing video |
| Tab to email text field of contact part | Tab to email text field of contact part | Speaker tell blind what to do | Correct result show in testing video |

|  |  |  |  |
| --- | --- | --- | --- |
| Tab to subject text field of contact part | Tab to subject text field of contact part | Speaker tell blind what to do | Correct result show in testing video |
| Tab to message text field of contact part | Tab to message text field of contact part | Speaker tell blind what to do | Correct result show in testing video |
| Tab to “to the home top” button | Tab to “to the home top” button | Speaker tell blind what to do | Correct result show in testing video |

|  |  |  |  |
| --- | --- | --- | --- |
| Test case (In laptop 1440 X 900) | Data inputted | Expect Result | Result |
| Tab Username text field of sign up page | Tab Username text field of sign up page | Speaker tell blind what to do | Correct result show in testing video |
| Tab email text field of sign up page | Tab email text field of sign up page | Speaker tell blind what to do | Correct result show in testing video |
| Tab monthly newsletter check box of sign up page | Tab monthly newsletter check box of sign up page | Speaker tell blind what to do | Correct result show in testing video |
| Tab newsflash check box of sign up page | Tab newsflash check box of sign up page | Speaker tell blind what to do | Correct result show in testing video |
| Tab subscribe button | Tab subscribe button | Speaker tell blind what to do | Correct result show in testing video |
| Tab unsubscribe button | Tab unsubscribe button | Speaker tell blind what to do | Correct result show in testing video |

|  |  |  |  |
| --- | --- | --- | --- |
| Test case (In laptop 1440 X 900) | Data inputted | Expect Result | Result |
| Tab evaluate button of search result page | Tab evaluate button of search result page | Speaker tell blind what to do | Correct result show in testing video |
| Tab select box of evaluate modal | Tab select box of evaluate modal | Speaker tell blind what to do | Correct result show in testing video |
| Tab submit button of evaluate modal | Tab submit button of evaluate modal | Speaker tell blind what to do | Correct result show in testing video |
| Tab cancel button of evaluate modal | Tab cancel button of evaluate modal | Speaker tell blind what to do | Correct result show in testing video |

**Operable (all functions available from keyboard, help users navigate)**

**Search function:**

|  |  |  |  |
| --- | --- | --- | --- |
| Test case (In laptop 1440 X 900) | Data inputted | Expect Result | Result |
| Test search function by title | ‘10’ | Found movie title contain ‘10’ | Correct result show in testing video |
| Test search function by rating | Title =’10’  Rating= ‘R’ | Found movies which rating are ‘R’ | Correct result show in testing video |
| Test search function by year | ‘1979’ | Found movies which year are ‘1979’ | Correct result show in testing video |
| Test search function by title | Title =’10’  Genre = ‘comedy’ | Found movies which genre are ‘comedy’ | Correct result show in testing video |
| Test search function (input multiple info) | Title=10  Rating=R  Year=1979  Genre=comedy | Found one movie which title is ‘10’ | Correct result show in testing video |

**Sign up function**

|  |  |  |  |
| --- | --- | --- | --- |
| Test case (In laptop 1440 X 900) | Data inputted | Expect Result | Result |
| Subscribe (Name input are not beginning with letter) | --fff | Warning message | Correct result show in testing video |
| Subscribe (Name begin with letter) | fff | Build data in database | Correct result show in testing video |
| Subscribe (Email string don’t have @) | Testgmail.com | Warning message | Correct result show in testing video |
| Subscribe (Email do not end with .com) | test@gamil | Warning message | Correct result show in testing video |
| Subscribe (Email end with .com but no character between @ and .com) | test@.com | Warning message | Correct result show in testing video |
| Subscribe (Email include hyphen) | Test-@gmail.com | Warning message | Correct result show in testing video |
| Subscribe (Email include space) | Test @gmail.com | Warning message | Correct result show in testing video |
| Subscribe (Correct format) | test@gmail.com | Build data in database | Correct result show in testing video |
| Subscribe (Correct format but didn’t choose subscribe plan) | test@gmail.com | Warning message | Correct result show in testing video |
| Unsubscribe | Correct mail which database have | Send email | Correct result show in testing video |
| Unsubscribe | Wrong mail which database don’t have | Guide to home page | Correct result show in testing video |

|  |  |  |  |
| --- | --- | --- | --- |
| Send monthly and newsflash mail | When event trigger | Send email | Correct result show in testing video |

**Rating function**

|  |  |  |  |
| --- | --- | --- | --- |
| Test case (In laptop 1440 X 900) | Data inputted | Expect Result | Result |
| Using searched table to evaluate movie | 0 star | Rating decrease | Correct result show in testing video |
| Using searched table to evaluate movie | 5 star | Rating increase | Correct result show in testing video |
| Using top ten list table to evaluate movie | 0 stars | Rating decrease | Correct result show in testing video |
| Using top ten list table to evaluate movie | 5 star | Rating increase | Correct result show in testing video |
| Update movie rate (edit database) | Title 10  Rating=5 | Top ten chart changed | Correct result show in testing video |
| Update movie rate (edit database) | Title:10  Rating=0  Title: 12 Monkeys (DTS)  CR=5 | Top ten list changed | Correct result show in testing video |
| Test top ten chart refresh function | non | Chart fade in every 5 sec(prevent seizures) | Correct result show in testing video |
| Every text field have hint | Non | Every text field have hint | Correct result show in testing video |

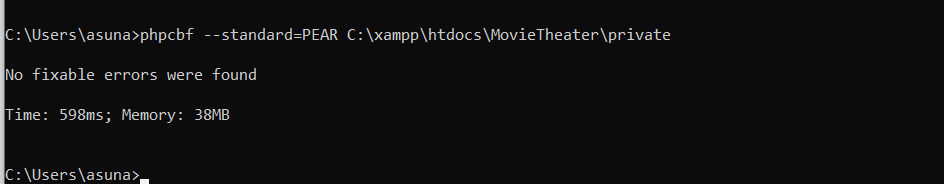
**Understandable (Make content readable and help users avoid and correct mistakes.)**

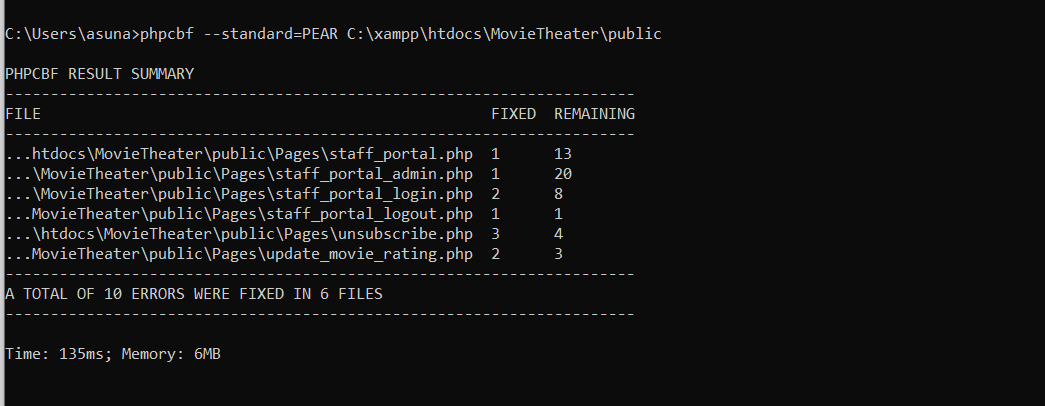
|  |  |  |  |
| --- | --- | --- | --- |
| Test case (In laptop 1440 X 900) | Data inputted | Expect Result | Result |
| Hover unsubscribe button | Hover unsubscribe button | Hint tell user the action will submit and come back to home page | Correct result show in testing video |
| Invalid input in subscribe form | Invalid input in subscribe form | Error message tell user how to do | Correct result show in testing video |

**Robust**

* Using PHP\_CodeSniffer: Using the optimization tool ‘PHP Beautify’ to the ‘Pear’ conventions/standards.

Below are screenshots of PHP Code Sniffer being applying the Pear standard to the private a public (in that order) php pages of the project.





* Using IDE(NetBeans/Eclipse): Using IDE to help us avoid significant syntax mistake and enhance the application robust

### Report

Result are as expected:

* Every element such as button, link and so on have voice hint for blind.
* Every logic path are tested, all output are as expected(under no screen reader envionment)
* Sing up and unsubscribe function working correctly.
* Ranking function working correctly.
* In the applictaoin, there are no significant bug.
* Every path had noticed users when the operating of users is wrong.
* The layout is responsive

Result are not as expected

* When the results of search movie are too many, the screen reader will not be able to afford it. Screen reader will read unexpected content and confuse user.
* Sometimes screen reader will not read invalid input warning message correctly and will not focus on html’s element as expect. However, when NVDA is closed all functions are working correctly. The consequence may be because NVDA is design for blind and it has its own algorithm to read html’s content and our team can’t change it. But our team have already added aria-label to every element for blind as possible as we can.

# Glossary of terms

|  |  |
| --- | --- |
| **Term** | **Definition** |
| WCAG | Web Content Accessibility Guidelines (WCAG) is developed through the W3C process in cooperation with individuals and organizations around the world, with a goal of providing a single shared standard for web content accessibility that meets the needs of individuals, organizations, and governments internationally. |
| IDE | An integrated development environment (IDE) is a software application that provides comprehensive facilities to computer programmers for software development. |
| RAD | Rapid Application Development (RAD) is a form of agile software development methodology that prioritizes rapid prototype releases and iterations |
| NVDA | NonVisual Desktop Access (NVDA) is a free and open source screen reader for the Microsoft Windows |

# Master Documents Tree/Links

**Folder:**

Sprint One Master Document – [folder-link](Master_Document_Sprint_One.docx)

Sprint Two Master Document – [folder-link](Master_Document_Sprint_Two.docx)

Sprint Three Master Document – [folder-link](Master_Documentation_Sprint_Three.docx)

**Github:**

Sprint One Master Document – [hyperlink](https://github.com/SMT-Diploma-of-Software-Development/Movie_Theater_Database/blob/final_version/Master_Document_Sprint_One.docx)

Sprint Two Master Document – [hyperlink](https://github.com/SMT-Diploma-of-Software-Development/Movie_Theater_Database/blob/final_version/Master_Document_Sprint_Two.docx)

Sprint Three Master Document – [hyperlink](https://github.com/SMT-Diploma-of-Software-Development/Movie_Theater_Database/blob/final_version/Master_Documentation_Sprint_Three.docx)