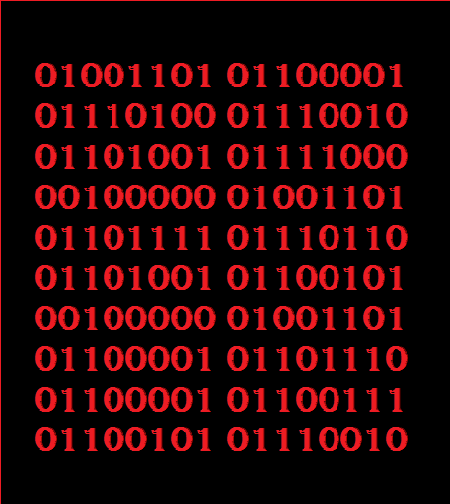
Senior Project Proposal

Matrix Movie Manager

Matrix



Submitted By: Ryan Williams

Submitted To: Calvin Caldwell

Email: [ryan.williams@oit.edu](mailto:ryan.williams@oit.edu)

Date: 5/16/2016

Version: 1.0

# Legal Notice

Matrix makes no warranties of representations of any kind concerning the accuracy or suitability of the information in this document for any purpose. All information is provided “AS IS” and with specific disclaimer of any warranties of merchantability, fitness for purpose, title and/or non-infringement. In no event shall Matrix, its employees or agents be liable for any direct, indirect or limitation only applies to the extent permitted by law and is without prejudice to any express provisions , to the contrary, in any express provisions to the contrary, in any written license or subscription agreement form Matrix in respect of the document or references.

# Copyright Notice

NOTICE: All information contained herein is, and remains the property of Matrix and its suppliers, if any. The intellectual and technical concepts contained herein are proprietary Matrix and its suppliers and may be covered by U.S. and Foreign Patents, patents in process, and are protected by trade secret or copyright law. Dissemination of this information or reproduction of this material is strictly forbidden unless prior written permission is obtained from Matrix.

# Revision History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Author | Company | Version | Date | Filename | Comments |
| Ryan Williams | Matrix | 0.1 | 5/12/2016 | Senior Project Proposal | Created the document |
| Ryan Williams | Matrix | 1.0 | 5/15/2016 | Senior Project Proposal | Initial Draft for 5/16/2016 |
| Ryan Williams | Matrix | 1.1 | 5/31/2016 | Senior Project Proposal | Final Draft for 6/3/2016 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# Signatory Page

This document is accepted by:

Signature (Calvin Caldwell) Date

This document is submitted by:

Signature (Ryan Williams) Date

# Table of Contents

Contents

[1. Legal Notice 2](#_Toc452573636)

[2. Copyright Notice 2](#_Toc452573637)

[3. Revision History 3](#_Toc452573638)

[4. Signatory Page 4](#_Toc452573639)

[5. Table of Contents 5](#_Toc452573640)

[6. Introduction 6](#_Toc452573641)

[6.1) Purpose 6](#_Toc452573642)

[6.2) Scope 6](#_Toc452573643)

[6.3) Intended Audience 7](#_Toc452573644)

[7. Project Management 7](#_Toc452573645)

[7.1) Change Management Procedure 7](#_Toc452573646)

[7.1.1) CAT Team 7](#_Toc452573647)

[7.1.2) Medium 7](#_Toc452573648)

[7.1.3) Protocol 8](#_Toc452573649)

[7.1.4) Time Lines 8](#_Toc452573650)

[7.1.5) Impact analysis 8](#_Toc452573651)

[7.1.6) Archive 8](#_Toc452573652)

[7.2) Software Delivery, Installation and Acceptance Criteria 8](#_Toc452573653)

[7.3) Documentation and On-Line Help 8](#_Toc452573654)

[7.4) Project Risks 9](#_Toc452573655)

[7.5) Customer Responsibilities 9](#_Toc452573656)

[7.6) Status Reposting 9](#_Toc452573657)

[7.6.1) Work Completed This Week 9](#_Toc452573658)

[7.6.2) Work Too Be Completed Next Week 9](#_Toc452573659)

[7.6.3) Issues 9](#_Toc452573660)

[8. System General Description 10](#_Toc452573661)

[8.1) Problem Statement 10](#_Toc452573662)

[8.2) Perspective 11](#_Toc452573663)

[8.2.1) History 11](#_Toc452573664)

[8.3) Major Subsystems 11](#_Toc452573665)

[8.4) Relation of System to Existing System(s) 11](#_Toc452573666)

[8.5) Hardware Platform Description 11](#_Toc452573667)

[8.6) Software Platform Description 12](#_Toc452573668)

[9. Product Requirements 12](#_Toc452573669)

[9.1) Functional 12](#_Toc452573670)

[9.2) Third Party Libraries 13](#_Toc452573671)

[9.3) Performance 13](#_Toc452573672)

[9.4) Reliability 14](#_Toc452573673)

[9.5) Data Dependency 14](#_Toc452573674)

[9.6) Security/Safety 14](#_Toc452573675)

[9.7) Constraints 14](#_Toc452573676)

[10. User Profiles 14](#_Toc452573677)

[11. Glossary 15](#_Toc452573678)

[12. Appendices 15](#_Toc452573679)

[**Change Form** 15](#_Toc452573680)

# 

# Introduction

# 6.1) Purpose

The purpose of this document is to provide details on the Matrix movie management system as well as to give management outlines for the course of this project. It will provide specifications for all levels of the software as well as recommended hardware specifications to run this software. It will also provide a guideline for management and progress reporting documents.

# 6.2) Scope

This document will detail how the project will be managed at both the design team and oversight levels, a description of the project, and the necessary requirements for this project to be considered completed.

# 6.3) Intended Audience

The intended audience for this document is Calvin Caldwell. The document is being written as a requirement for Calvin Caldwell’s CST 334 Senior Project Planning class in spring of 2016 at Oregon Institute of Technology. This document is written in such a way that a reader with a basic understanding of computer will understand the majority of it.

# Project Management

This section serves to detail how the project will be managed over the course of development, from beginning to release.

# 7.1) Change Management Procedure

This section details who can submit changes to the project and what will be required of them in order for the development team to process their request.

# 7.1.1) CAT Team

The CAT team consist of the senior project instructor and me (Ryan Williams). The team will evaluate the impact that a change would have on the production of the system. Submitted changes can be accepted or rejected by the team with a brief statement to explain why.

# 7.1.2) Medium

All changes will be submitted via a typed document. These documents MUST be in .doc or .docx format. The document should contain a detailed description of the changes to be made in the project. This will be submitted to the development team by attaching the change request document to an email and sending it to ryan.williams@oit.edu. Email Subject will as such: Change Request - <Keywords> and the body of the email will contain a brief description of the change request. Emails that do not meet these guidelines will not be considered.

# 7.1.3) Protocol

All change requests will initiate a meeting with Calvin Caldwell. This meeting will be a discussion to determine if the submitted change request fits within the scope of this project and whether to implement the change or not.

# 7.1.4) Time Lines

A reasonable response to any change request will be provided to the requester within 3 business days of the submission.

# 7.1.5) Impact analysis

Any request change should not extend the development time of the project beyond the allotted 10 week period. Changes that extend the development time beyond that period will be denied.

# 7.1.6) Archive

Email correspondence will be stored on the mail server as well as a minimum of two local backups. Official change request documentation will be kept on an off side cloud storage server.

# 7.2) Software Delivery, Installation and Acceptance Criteria

Software delivery will be distributed to all parties for evaluation. The delivery will include an installation package, written documentation of system requirements and a read me that will contain some basic instructions on how to use the program. This program will be publically available on GitHub upon completion. Acceptance criteria will be based on the completeness of the below requirements and acceptable operation of the software provided

# 7.3) Documentation and On-Line Help

Software documentation will be available in three forms: paper documentation, program help, and digital documentation in the form of a read me file that is provided with the installation. This documentation will describe the system requirements, installation process, and operation of the software.

# 7.4) Project Risks

The project risks include successful operation and manipulation on a majority of the common movie file types. Including: mp4, mkv, and avi. Another risk with this project is the conservation of data used as the program will be modifying the Meta data of the movie files.

# 7.5) Customer Responsibilities

The customer is responsible for providing the files that will be managed by the software. The customer will also be responsible for evaluating the performance of the software and providing feedback in a timely manner.

# 7.6) Status Reposting

Weekly reports will be submitted to Calvin Caldwell. These reports will cover these main sections: work completed this week, work to be completed next week, and setbacks and issues.

# 7.6.1) Work Completed This Week

This section will cover all the work completed in the last 7 days. These reports will be submitted on a weekly basis.

# 7.6.2) Work To Be Completed Next Week

This section will detail the work that is expected to be completed within the next week of work on the project.

# 7.6.3) Issues

Any issues that may have occurred in the previous weeks tasks shall be listed in this section of the report.

# System General Description

This section of the proposal will provide a description of the project in full detail.

# 8.1) Problem Statement

The main idea of this project is to have a way of graphically managing a movie file library that gets the useful information (ie. Actors, runtime, and cover art) from an external source and presents those in a viewable and more useable way than file explorer. There are many features that I will include in this project so that it meets my needs as a movie collector. This will be a list of features that I want to have to make the app functional.

**Information gathering from IMDB-** What I mean by this is that given the title of a movie, I will get that movies information from IMBD and populate it into a movie object that will be used in the library. The data that will be gathered from IMDB is, actors, rating, runtime, cover art, summary, and its date of release.

**Displaying movies visually-** My idea for this is in a grid format of movie objects that will display the cover art and movie name below it. This library will be fully searchable by name, actors, runtime, personal and critic rating, recently added, and recently viewed. When a movie object is selected from the list, a sidebar will appear to display information about the movie, including: cover art, actors, run time, average rating, personal rating, and plot summary. This menu will also include the play button that I will talk about a little later.

**“Do I have this” button-** This is a button that personally would save me enormous amounts of time when trying to find movies. The idea is that you click the button and provide the name of the movie that you are looking for. The app will then search your local library for the movie and tell you if you have it or not. It will then bring the movie up in the selected sidebar discussed earlier and allow you to play it. No more will I wait to download a movie that I already have in possibly better qualities.

**Direct play button-** This is a button that appears in the selected movie side bar. This button does exactly what it sounds like. But depending on the file format there are different players that better display the formats. For example, VLC media player is great for newer movie files including mp4 and mkv, but windows media player is better at handling older files that where converted from VHS to a digital format.

**“Suggest a movie” button-** This is an idea that I thought would be interesting to try and implement. Basically the app would keep track of your previous movies that you have viewed. Then when this button is clicked it will do a search of IMDB based on your currently most watched genre in the past 5-10 movies and provide a list of options that you may like or want to check out.

# 8.2) Perspective

This section serves to document some perspective on the development team and the project software.

# 8.2.1) History

This is an issue that I personally have been dealing with the last 3 years. Personally I have a digital collection of nearly 300 movies but lack an easy way of accessing them beyond using windows explorer to select a title. Even when selecting a title, in the case of movie remakes or copies, I have no way of knowing if the title I’m selecting is really the one I want. With this problem in mind, I’m making a local movie management system that will be able to more accurately display important information (Cover art, genre, actors, and summary) about my movies and allow for easy access to them when wanted.

# 8.3) Major Subsystems

The front end of this software will be an easy to navigate windows presentation forms application that will be a populated catalog of the locally stored content of the system. This content currently missing information (cover art, summary, actors, genre, critic rating) will be pulled from a api known as OMDb (description listed below in 9.2). This content will then be stored in the Meta data values of the movie.

# 8.4) Relation of System to Existing System(s)

This program as stated earlier will be working with a few different systems.

* VLC- VLC is a video playback program that will be linked to the catalog to provide seamless playback of selections without leaving the main application.
* Windows Media player- Windows media player is a media playback application that is provided with each installation of windows. It is the primary media viewer for the platform and arguably the best for supporting older video and audio display types. I will also be using this in the program for displaying content when needed.
* OMDb api- This is a large scale movie database that pulls information from IMDB (Internet Movie Database), one of the most popular movie information and review sites in the world. I will be using this system to populate the Meta data of each of the movies stored locally on the system

# 8.5) Hardware Platform Description

Users will need a system with at least an Intel Pentium 4 processor, 350 MB of hard-disk space, and 1 GB of ram.

# 8.6) Software Platform Description

It is recommended that this program run on windows 7 or above. The program will primarily developed using C# on a windows 10 platform.

# Product Requirements

# Functional

The area of functional requirements describes operations the system is intended to perform. These operations are captured as short statements; each statement intended to describe a single behavior. These statements are grouped based on similarity. The groups are called functional areas and encompass functional modules to be treated as individual efforts in development, testing and deployment.

Functional Requirements are prioritized as follows:

* H-A : High Priority, Architecture – These requirements are mandatory for architectural integrity of the system technical operation.
* H : High Priority – These requirements are part of system basic operation. Without these requirements, the system cannot be considered operational.
* M : Mid-Level Priority – These requirements are necessary for a final delivered system. The system will function without these operations; however, it may not be useful from and end user perspective.
* L : Low Level Priority – These requirements are items that would be nice to have implemented but do not add to necessary functions for end system implementation.

These are that functional requirements relating to the Matrix movie manager

1. There will be a catalog style interface
   1. Movies will displayed in a grid
      1. Movies in grid will be displayed by Cover Art
      2. Grid will be scrollable
   2. Movies will be selectable
      1. Selecting a Movie will bring up a side display
         1. Side display will show Information about a movie
            1. Title
            2. Actors
            3. Summary
            4. Runtime
         2. There will be a options area
            1. Options area will allow for open in local folder location
         3. There will be a play button in the side display
            1. Play button will launch the appropriate video player for the format
            2. After Video playback is completed or ended the User will be prompted for rating the movie
   3. Movies will be searchable
      1. By Title
      2. By Actor
      3. By Genre
      4. Runtime
      5. Personal and Critic Rating
      6. Recently added
      7. Recently viewed
2. There will be a suggest a Movie feature
   1. Will suggest a movie based on the 5 previously viewed movies
3. There will be a previously viewed catalog
   1. Will catalog all of the movies watched by the user through the program
   2. Data will be used for suggested movie feature
4. User will be able to search for information about new or unowned titles using the OMDb api
   1. Search will be done through Internet Movie Database
      1. Will be able to search by
         1. Title
         2. Actor
         3. Genre
         4. Rating
5. User will be able to quick search through local catalog to find out if a movie is owned
   1. If the movie is not owned, the program will ask the user if they would like to view the information page about the searched movie
6. User will be able to define locations of local collection folders
   1. User will be able to select folders from separate drives or file locations to view in the catalog

# Third Party Libraries

1. OMDb api - System will retrieve information (including Cover art, actors, critic rating, summary, genre, release date) from OMDb

## Performance

These statements are related to minimum and maximum criteria for system speed and metrics related to system efficiency and operation.

1. System should never have data collection of load times more than 5 seconds
2. System should allow for any file sizes to be used
3. System should allow for standard video formats to be used
4. The system should have little to no delay when starting to watch a selected movie

# Reliability

The reliability of this software will be about 98% of time. Issues that the user may see may come from conflicts or changes made to the API called database. This will affect the user in the content they may see because of changes that are not in the programs control to regulate.

# Data Dependency

Users’ local content will be stored in folders on their storage drive(s). This data should be accessed at the minimum speed of the hardware of the system (~ 5mb/second).

Movie information (genre, actors, rating, cover art, summary, release date), will be pulled from the OMDb API. This data will be stored as Meta data on the local content files. This data should be transferred to the application, based on the network speed of the machine that is running it. Minimum of 500 KB per second.

# Security/Safety

All information that is gathered through the API will be obtained through encrypted packets.

# Constraints

Any movies that are obtained by the users of this program are assumed to obtained in a legal way. Through purchase or rental services. Matrix does not take responsibility for any content that may not have been obtained by other means. Programing of the system will meet the standards of the C# programing language.

# User Profiles

Users of this software will fit one or all of the following descriptions:

* Interested in movies
* Interested in managing personal digital movie libraries
* Interested in learning about movies

# Glossary

# Appendices

# **Change Form**

Name (Print):

Response E-Mail:

Request of Change:

Purpose of request of change:

Additional Comments: