2021

OneStop Media Library conceptual design

DATABASE SYSTEMS PROJECT

AUTHORS

Names	ID	E-mail	GitHub
Adonay Dereje	ATR/7743/10	donniedereje1@gmail.com	adonaydereje
Eyob Eshetu	ATR/0164/10	eyobgeletu@tutanota.com	eyob-geletu
Fekadesilassie Tsegaye	ATR/5519/10	beshoo.f1990@gmail.com	beshoo-07
Fraol Bereket	ATR/8289/10	fraolbereket@gmail.com	fraolb
Henok Hailu	ATR/4973/10	henokhailu96@gmail.com	henokhailu96

VERSION HISTORY

Version Number	Date of Revision	Reason for Revision
1.00	Apr. 17, 2021	-

Table of Contents

INTRODUCTION

PURPOSE OF DOCUMENT

The document will provide the requirements for the OneStop application. It will list all the necessary database entities and relationships, software dependencies as well as product features for the software. This document is intended for future reference by the developers and reviewers.

SCOPE OF THE PROJECT

OneStop is a desktop application that organizes video, music and game media into one library. The user can track list of played media and owned media. This software is useful for people with large media libraries, people looking for better user experience, or people who would want all their media in one location.

OVERVIEW

This document is prepared based on the guidelines set in the IEEE STD 830-1198 Recommended Standard Template. The structure of the document is shaped around this standard with some modifications for the sake of clarity.

The document begins with an overall description of the software, which contains a description of the required functional features, user characteristics, program dependencies and program constraints. Then it will describe the specific features of the software, which includes the external applications the software will use, the software's functions and the database requirements. Finally, definitions and references will be listed.

The document will be supported with mock interface, ER Diagrams, tables and other visual aids for better readability and simplicity.

OVERALL DESCRIPTION

This section will provide an overall description of the requirements and specifications. It will be further discussed in the specific requirements later in the documentation.

PRODUCT PERSPECTIVE

OneStop is specifically intended to run on a Windows 10 operating system with an x64 based processor. The software will require an internet connection to fetch the required data from its API integrations. OneStop makes use of the following API integrations to function: AniDB for fetching information about anime files, IGDB for game files, Musicbrainz for music files and TMDb for movies and TV series.

PRODUCT FUNCTIONS

OneStop looks for media in a user-defined directory, searches for its information online using its naming scheme, creates a local database based on the information, displays the media in menus and submenus, plays it using external players or optionally internal players and finally logs the user's library data.

Fetching Media Information

The application will get the name of the file from a user-defined directory based on a specific naming scheme. This file name will be searched in an online database pertaining to the media. If the file name exists, information about the media will be fetched.

Creating a Local Database

From the fetched information a local database will be created for each media. This will contain the metadata provided by the online database systems in relation to the user file.

Displaying Media

Based on the media information, the application will sort through the database and display each media in its respective menu and submenu.

Playing Media

The application will play the user's files using external players that will be specified later in the documentation. An optional internal player will be added as a bonus feature later in the project.

Logging User Data

The user's played media, owned media and playtime will be logged in a local database. This will be media specific.

An online profile feature will be considered in the future.

USER CHARACTERISTICS

The software is designed for users familiar with the Windows 10 environment and desktop applications. It requires a basic understanding of the various hardware components associated with modern day computers and their usage in desktop applications.

The user is required to comply with a proper naming scheme for the media files for accurate information to be provided. OneStop will use the directory folder structure and file name to search for the media metadata online and therefore requires an identifiable information.

The user must create a folder for each media: Anime, Games, Movies, Music and Series. Each type of media will have its own file naming scheme and folder structure. Anime files must have a [title] + [season_no] + [episode_no] structure. Series has a similar naming scheme. Music should have an [artist] / [album_title] / [track _no] + [track_title] scheme. For movie files [title] + [year] must be used.

CONSTRAINTS

The project will strictly be for personal use and will therefore not require any premium licensing. It will be under the Creative Commons license when uploaded on Github.

The hardware must be a Windows 10 machine with an x64 processor as mentioned above. This is the hardware requirement mentioned above.

OneStop will be developed using C# and .NET framework and Microsoft SQL Server. Other languages will be added when deemed necessary.

The application will have to interface with its API integrations for proper function. For further information see the External Interface section.

ASSUMPTIONS AND DEPENDENCIES

The project is designed under the assumption that it will be released as a Windows 10 desktop application. The languages it will use will be:

- C# for the entire program
- .NET Framework for the GUI
- JSON for fetching some API information
- Microsoft SQL Server for the database

SPECIFIC REQUIREMENTS

EXTERNAL INTERFACE

OneStop makes use of the following external interface: AniDB, IGDB MusicBrainz and TMDb for API integration and Media Player Classic for an external player.

API

AniDB

AniDB is a non-profit database of anime information that is freely open to the public. As well as providing general anime information, with an account, users can organise their anime collection, keep track of what they've seen and other extra features.

IGDB

IGDB.com is a gaming website that gathers all relevant information about games in one place, builds social and exploratory features on top of this information, gathers a community of both Gamers and people from the game industry and let them communicate with each other, and focuses on our users and let them decide on the design and features.

MusicBrainz

MusicBrainz is a community-maintained open-source encyclopedia of music information. This means that anyone can help contribute to the project by adding information about artists and their works.

In 2000, Gracenote took over the free CDDB project and commercialized it, essentially charging users for accessing the very data they themselves contributed. In response, Robert Kaye founded MusicBrainz. The project has since grown rapidly from a one-man operation to an international community of enthusiasts that appreciates both music and music metadata. Along the way, the scope of the project has expanded from its origins as a mere CDDB replacement to the true music encyclopedia MusicBrainz is today.

As an encyclopedia and as a community, MusicBrainz exists only to collect as much information about music as possible. MusicBrainz does not discriminate or prefer one "type" of music over another, and it collects information about as many different types of music as possible. Whether it is published or unpublished, popular or fringe, western or non-western, human or non-human – MusicBrainz has it all.

TMDb

The Movie Database (TMDb) is a community-built movie and TV database. Every piece of data has been added by its community dating back to 2008. TMDb's strong international focus and breadth of data is largely unmatched.

PLAYER

Media Player Classic

Media Player Classic is an extremely light-weight media player for Windows. It is written in C++, supports GPU assisted decoding, and has been translated in 42 languages.

GUI

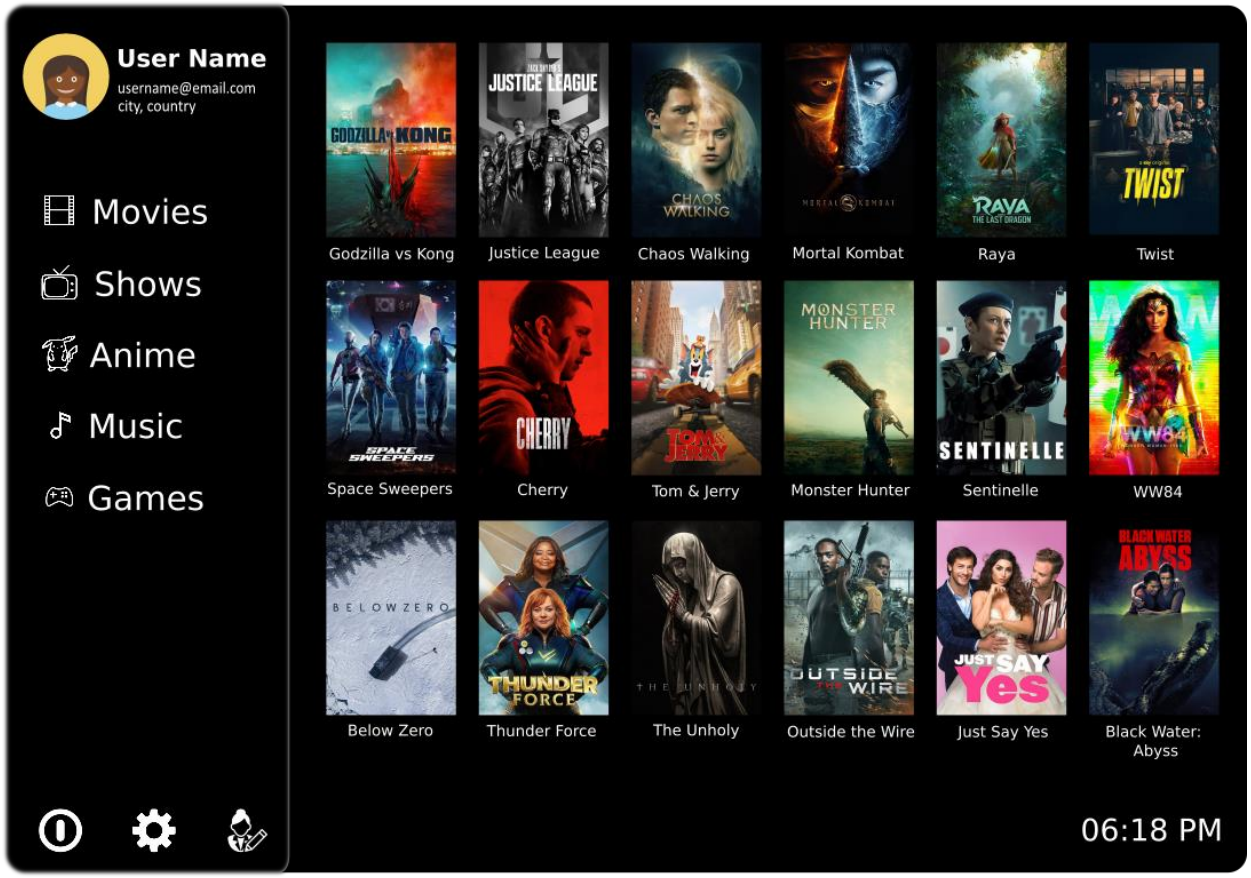
The software provides a good and attractive graphic interface for the user. The user can interact with the software with no difficulty. The elements of the GUI will be easy to understand and self-explanatory.

- The media files that will be fetched by the software are categorized based on the information they have or their type.
- It will allow the user to easily pick between movie, series, and music categories.
- The user can select and see the necessary information about the selected media file. The information will be displayed in the right section of the GUI window.
- The information may include summary about the media file. For example, if the media file is a TV show; it will include the season, the episode, date released, etc.

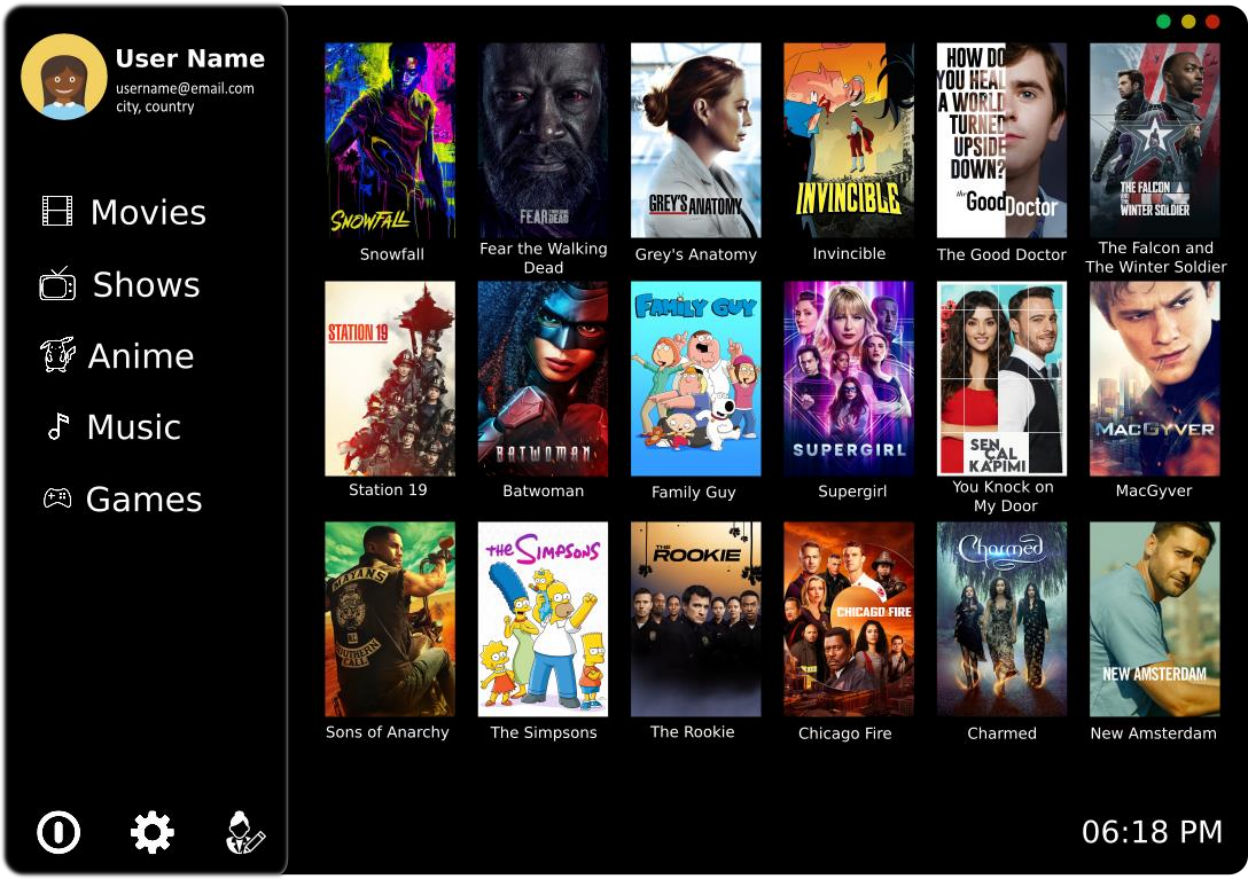
PERFORMANCE REQUIREMENTS

The database system has to be fast and there should not be major delays in retrieving information. Since the files are already in a local drive response time under load conditions should be less than 4 sec (for any request of the user). When gathering information from other APIs the response time should not exceed 10 seconds. Otherwise, it should stop and retry.

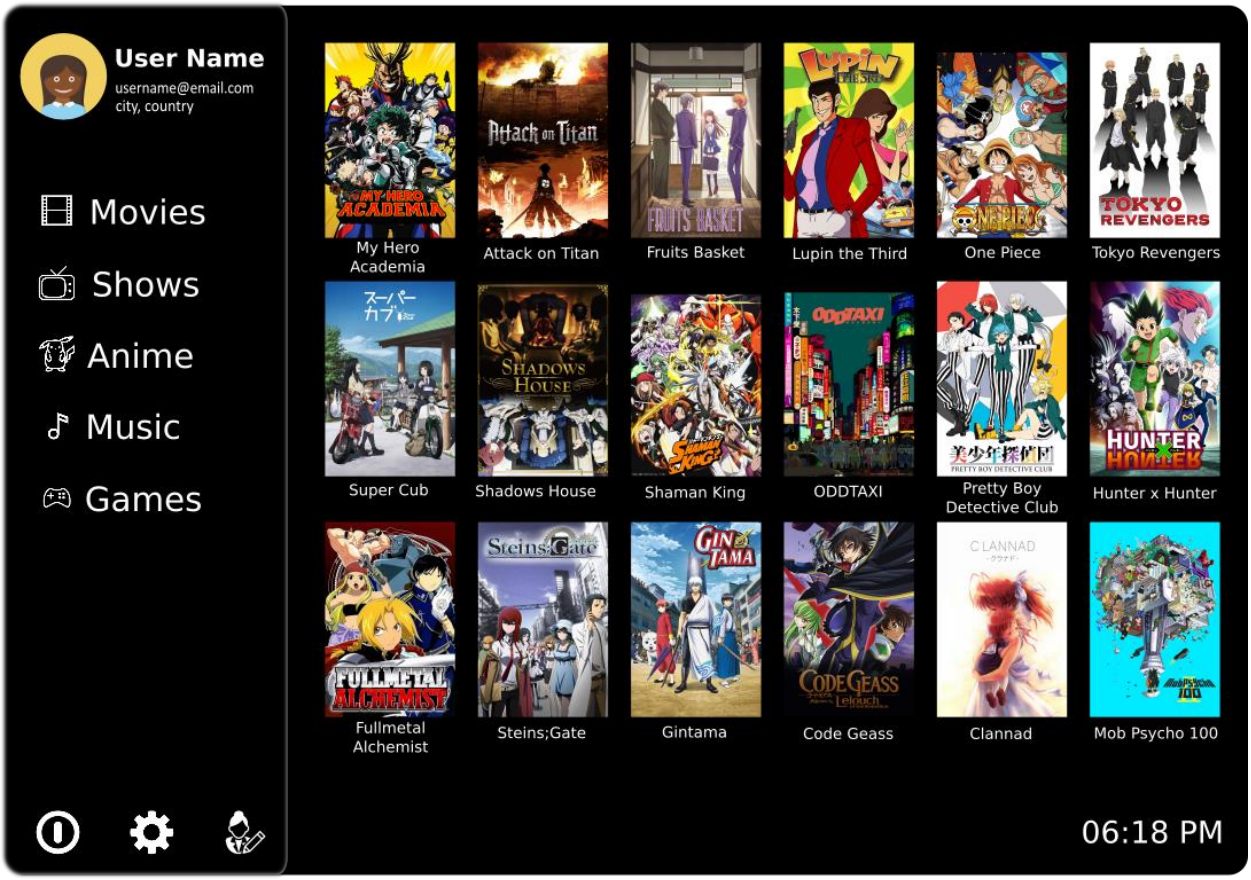
Movies Mockup Screen



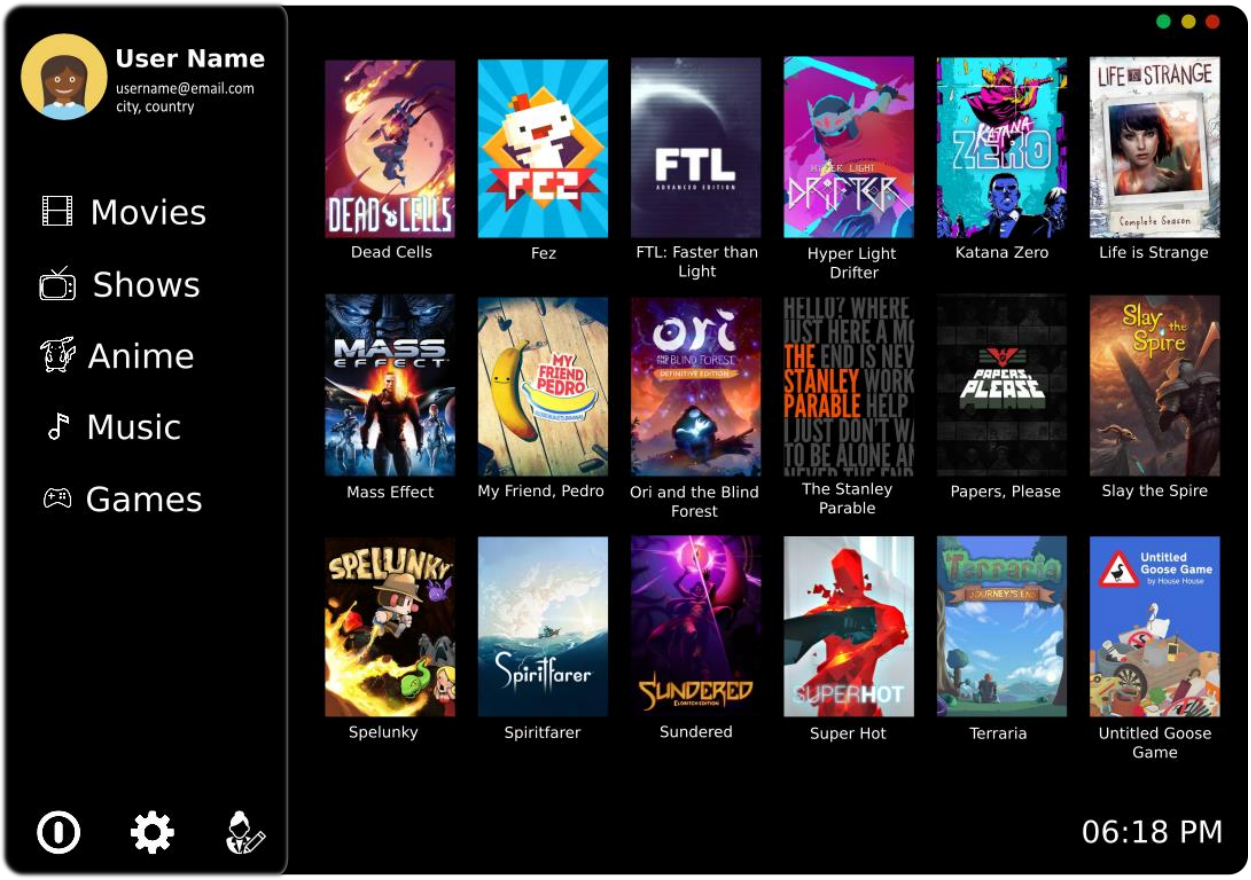
Shows Mockup Screen



Anime Mockup Screen



Games Mockup Screen

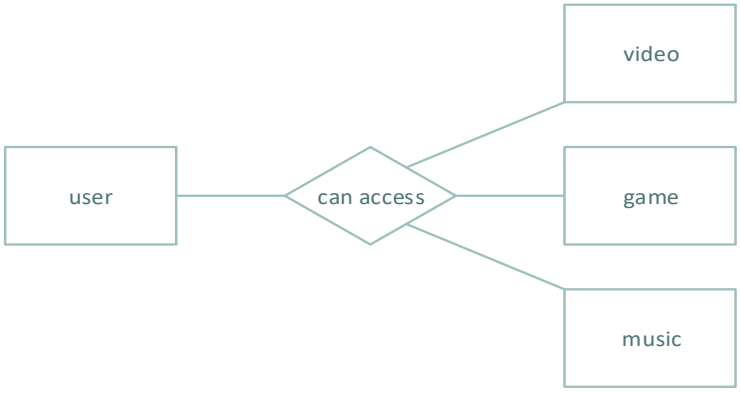


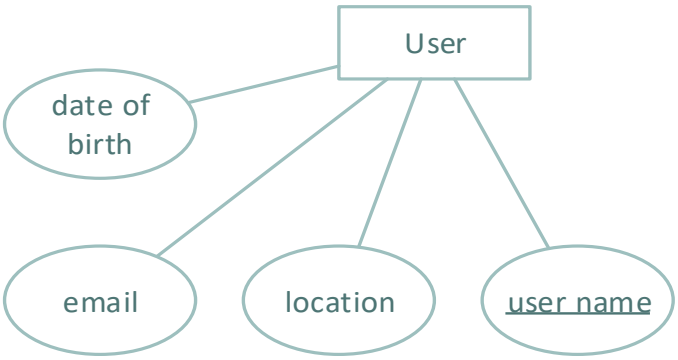
DATABASE REQUIREMENTS

This project focuses on the database management system of the media library software.

USER ENTITY

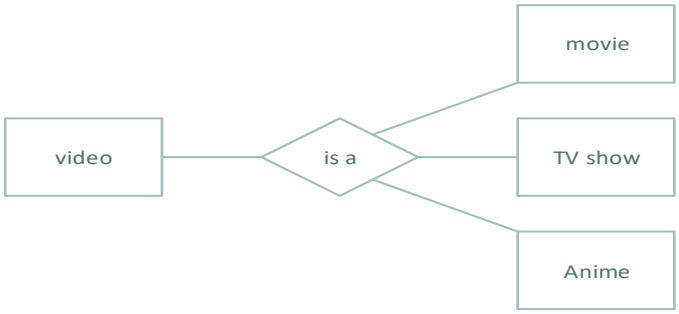
- The database will store user information such as username, date of birth, location (country) and email in user entity.
    - Username will be the primary key
    - Date of birth and location will help for suggestions
    - The email will allow the user to back-up to cloud
    - User entity will have one-to-many relationship with video, music and game.
    - Therefore, **User can access music, video or the game**



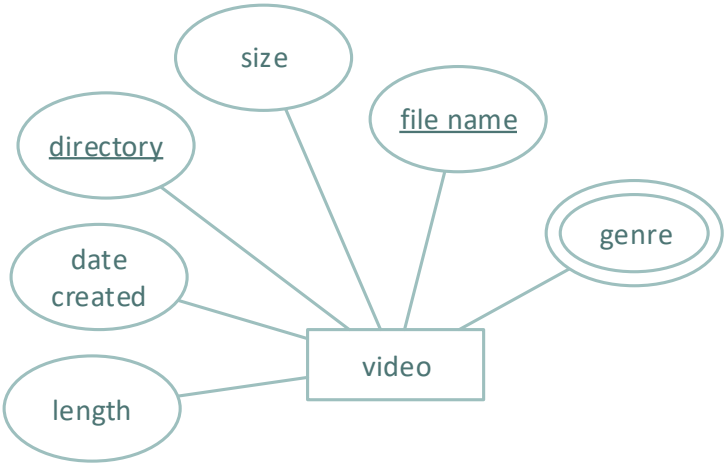


VIDEO ENTITY

- will have length, date created, size, directory, file name and genre
  - Length will tell the information how long the video will take to watch
  - Directory will describe the source of the video in the local drive
  - Genre describes the category of the video. It is a multi-valued attribute and will have values like action, comedy, horror, etc.
  - File name and directory are key attributes
  - Video will have one-to-many relationship with movie, TV show and anime.
  - Therefore, **video is a movie, TV show or Anime**

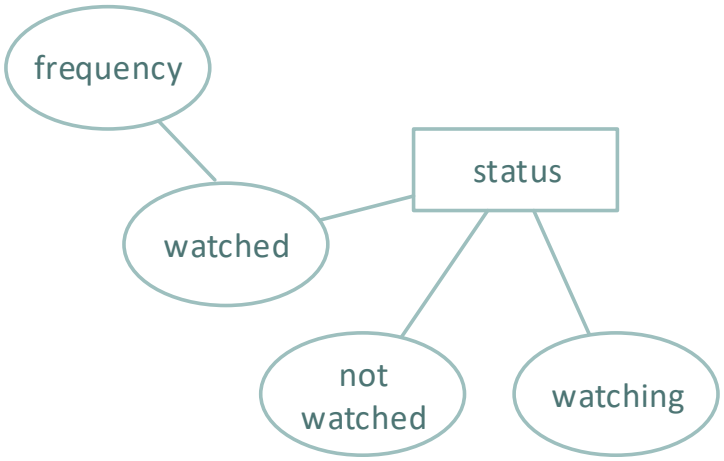


  - Video will also have one to one relationship with status.
  - Then **video has a status of status attributes**



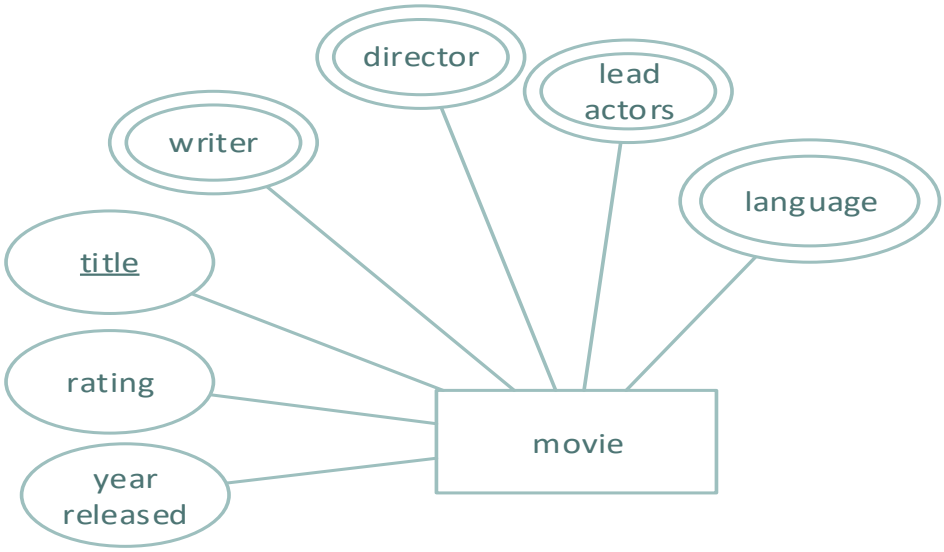
STATUS ENTITY

- will have watched, watching and not watched attributes
    - Watched will have information of already watched videos
    - Watching will be videos the user started but hasn't finished watching
    - Not watched will be videos the user hasn't opened.
    - Watching will have a composite attribute called frequency
    - Frequency will record how many times the video have been watched



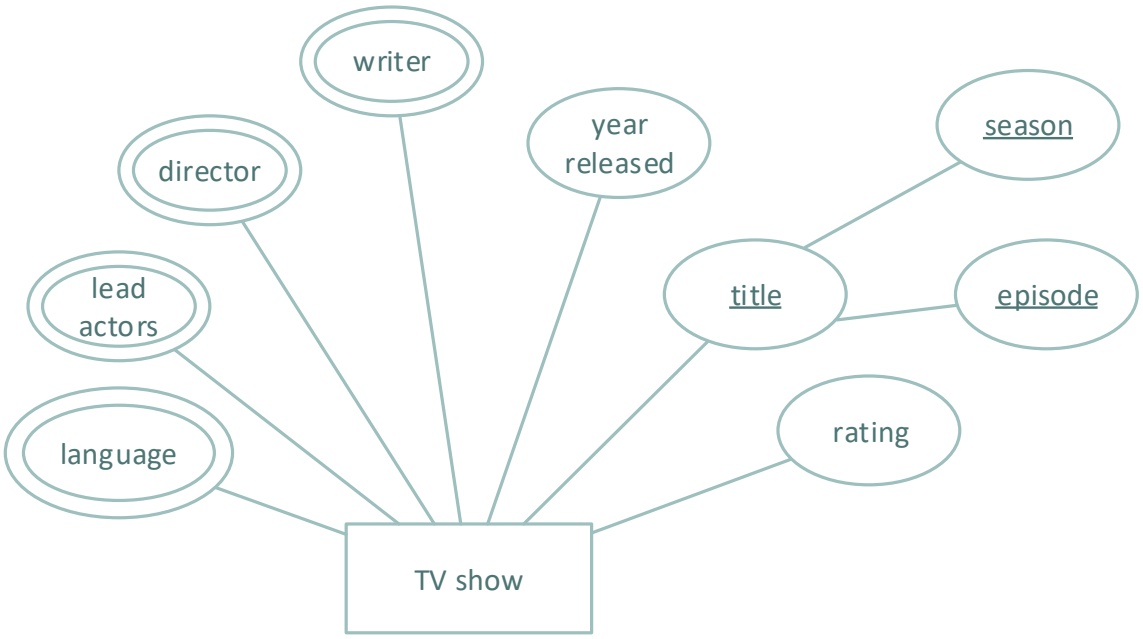
MOVIE ENTITY

- will have language, lead actors, directors, writer, title, rating and year released attributes under it
    - Language is multi-valued and will tell the languages of the movies
    - Lead actors will tell the few actors participated in the movie
    - Director and writer will tell the names of the directors and writers of the movie which can be more than one often.
    - Rating will tell the information how people find the movie interesting
    - Title will be the primary key attribute



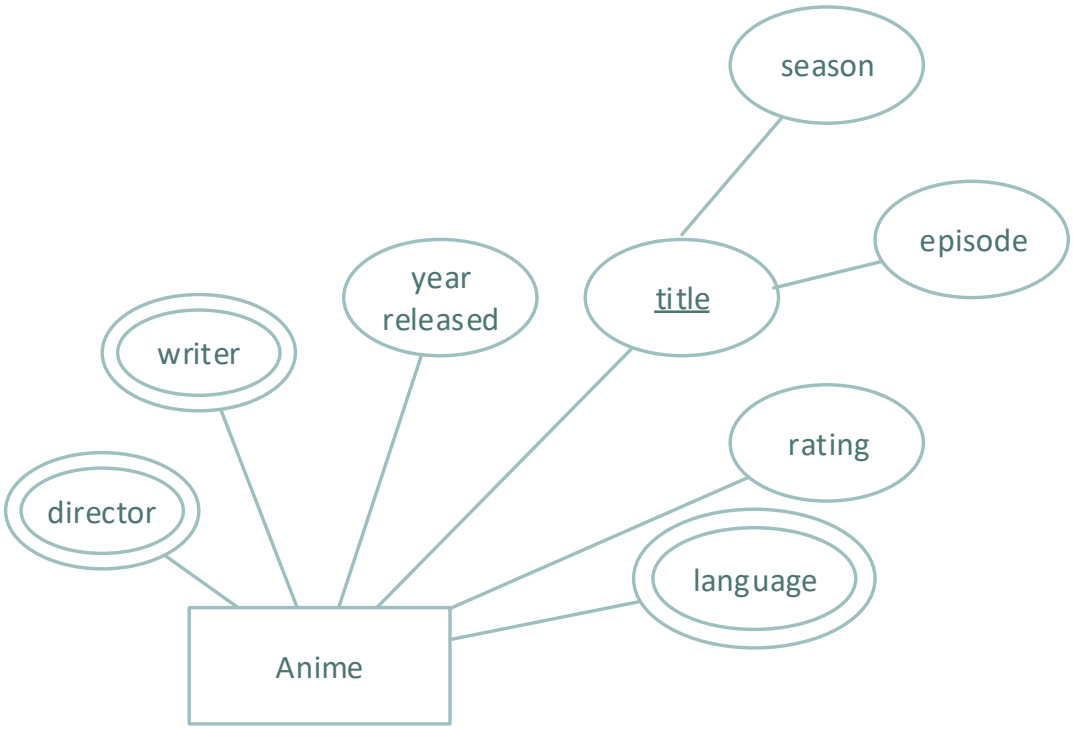
TV SHOW ENTITY

- just like movie entity it will have language, lead actors, directors, writer, title, rating and year released attributes under it
- In addition to that title will be composite and have season and episode attributes
    - Seasons and episodes will differentiate the items under the same series title
    - Title will be the primary key attribute



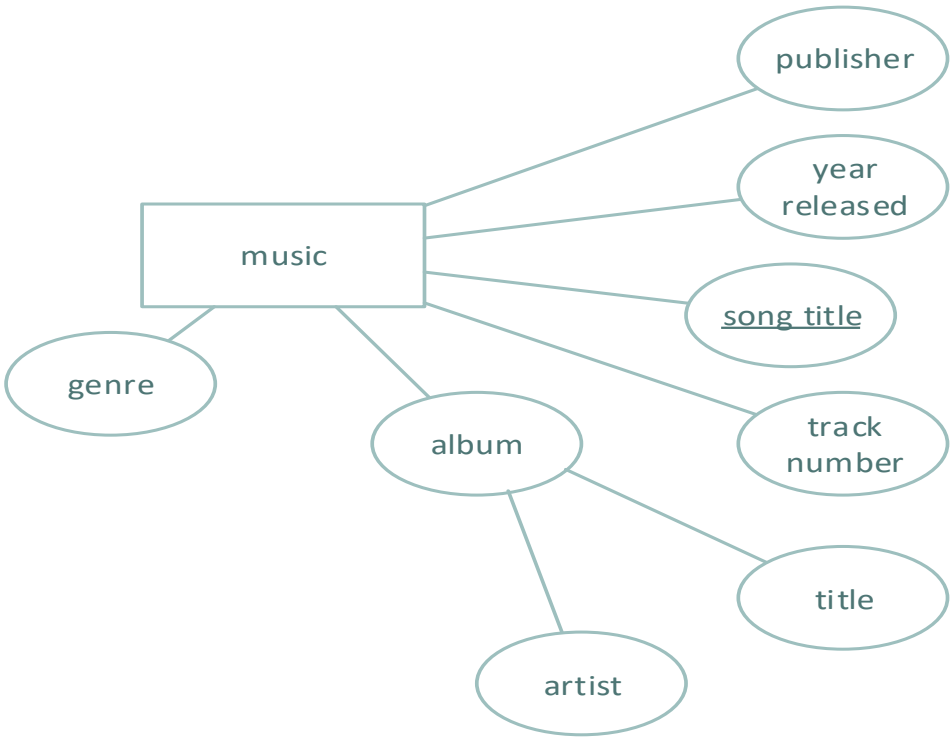
ANIME ENTITY

- will have the same attributes as TV shows except lead actors
    - Title will be the primary key attribute
    - User may be required to define the directory of the files for this entity manually



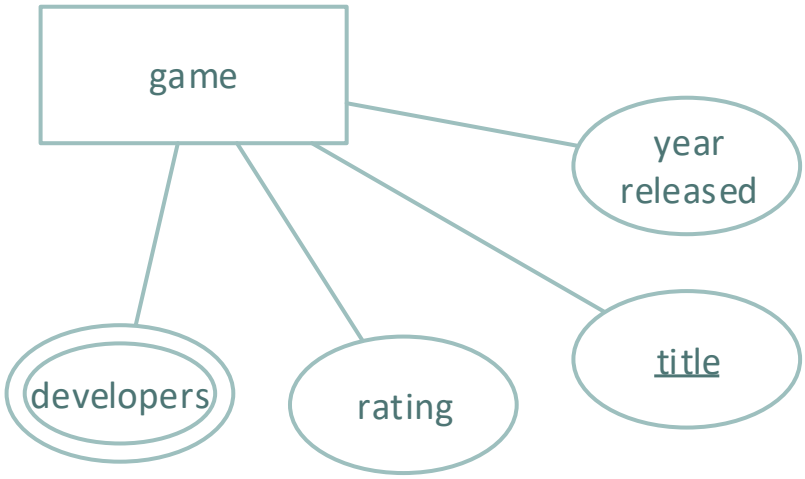
MUSIC ENTITY

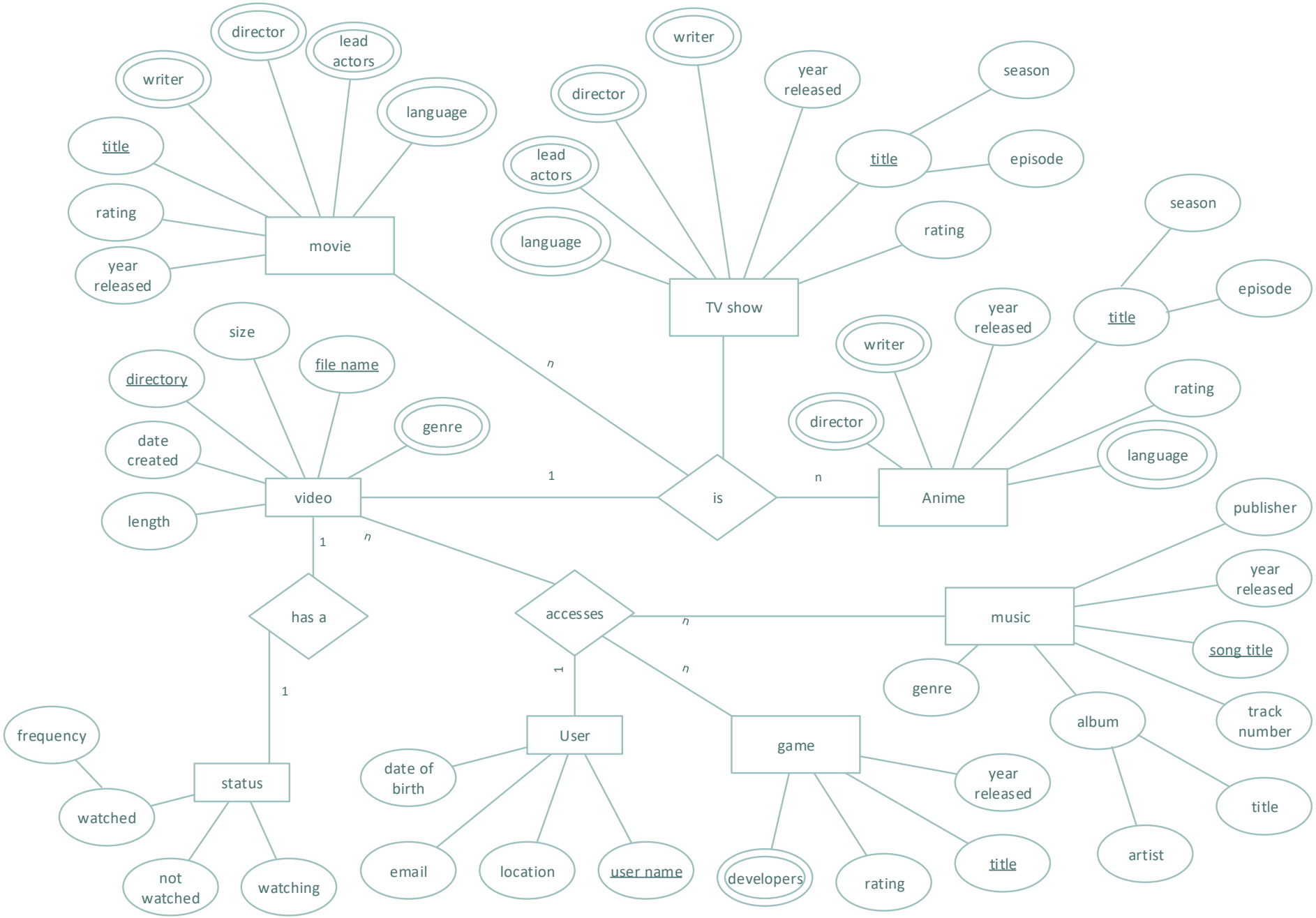
- will have song title, album, year released, track number, genre and publisher
    - Album is a composite of title and artist; it indicates the album the song is a part of
    - Song title is the primary key
    - Genre in music will have values like pop, jazz, EDM, etc.



GAME ENTITY

- will have title, year released, rating and studio
    - Title will be the primary key
    - Rating will tell how much people find the game interesting
    - developers will tell the people participated in the game development




director
lead actors
writer
language
title
rating
year released
movie
writer
director
year released
season
lead actors
title
episode
language
rating
TV show
season
episode
title
year released
writer
rating
director
language
Anime
size
file name
directory
genre
date created
length
video
1
n
is
1
n
1
n
has a
accesses
1
n
1
publisher
year released
song title
track number
album
title
artist
music
genre
frequency
watched
status
not watched
watching
User
date of birth
email
location
user name
game
year released
title
developers
rating

SOFTWARE SYSTEM ATTRIBUTES

Reliability

The software may crush at some point due to viruses, unresolved bugs, operating system failure or in case of power supply failure while using the software. Therefore, the users are recommended to back-up the information in their profile to a cloud using their email account.

Availability

It will be available 24/7, since it is a desktop application anyone who have installed the software can access it at any time.

Security

Security won't be an issue in this software. The user only needs to upload current state of the software to his cloud account like Google drive for a back-up. And this is only if the user wants to.

Maintainability

The software and the database system are maintainable. Repair and feature update might occur in the future. The frame work we will use will allow that.

Portability

The software works on any computer that is running in windows operating system.

DEFINITIONS, ACRONYMS AND ABBREVIATIONS

API: is the acronym for Application Programming Interface

DBMS: A database management system (DBMS) is a software package designed to define, manipulate, retrieve and manage data in a database

SQL: Structured Query Language, used to communicate with a database

.NET framework: a software framework developed by Microsoft that runs primarily on Microsoft Windows.

C#: used in creating applications of Microsoft at a large scale

GUI: (graphical user interface) is a system of interactive visual components for computer software

REFERENCES

Institute of Electrical and Electronics Engineers, IEEE STD 830-1998

AniDB, https://wiki.anidb.net/Main_Page

IGDB, https://www.igdb.com/about

MusicBrainz, https://musicbrainz.org/doc/About

TheMovieDB, https://www.themoviedb.org/about

Media Player Classic, https://mpc-hc.org/about/