Video Library Management System - Documentation

# Project Overview

The Video Library Management System is a Python-based console application that allows users to add, edit, delete, and display videos from a library. The application should allow users to store video information such as title, director, release year, genre, and duration. Users are also able to search for videos based on different criteria’s.

The application utilizes functions, collections, strings, numbers, datetime objects, and file operations. The application also ensures data persistence through file operations, saving and loading video data from a JSON file.

# Features

* Add New Videos: Users can add videos by entering details like title, director, release year, genre, and duration.
* Edit Videos: Users can modify the details of existing videos.
* Delete Videos: Users can remove videos from the library.
* Display All Videos by Name: Lists all videos stored in the library by their title.
* Display Detailed Information: Shows detailed information for a specific video selected by the user.
* Search Videos by Criteria: Allows users to search for videos based on director, release year, genre, or duration.
* Data Persistence: Video data is saved in a JSON file, this ensures that information is retained between sessions.

# How to Run the Project

## Prerequisites:

Spyder(Python 3.11) installed on your system.

System Requirements: Windows, MacOS

## Running the Application:

Run the python video\_library\_management.py code in Spyder(Python 3.11)

## Using the Application:

Follow the on-screen prompts to add, edit, delete, display, and search for videos.  
Choose the appropriate option from the menu by entering the corresponding number.

# Project Structure

video\_library\_management.py = Main application script  
README.md = Project overview and instructions

Documentation = docx

# Code Structure

## Main Functions:

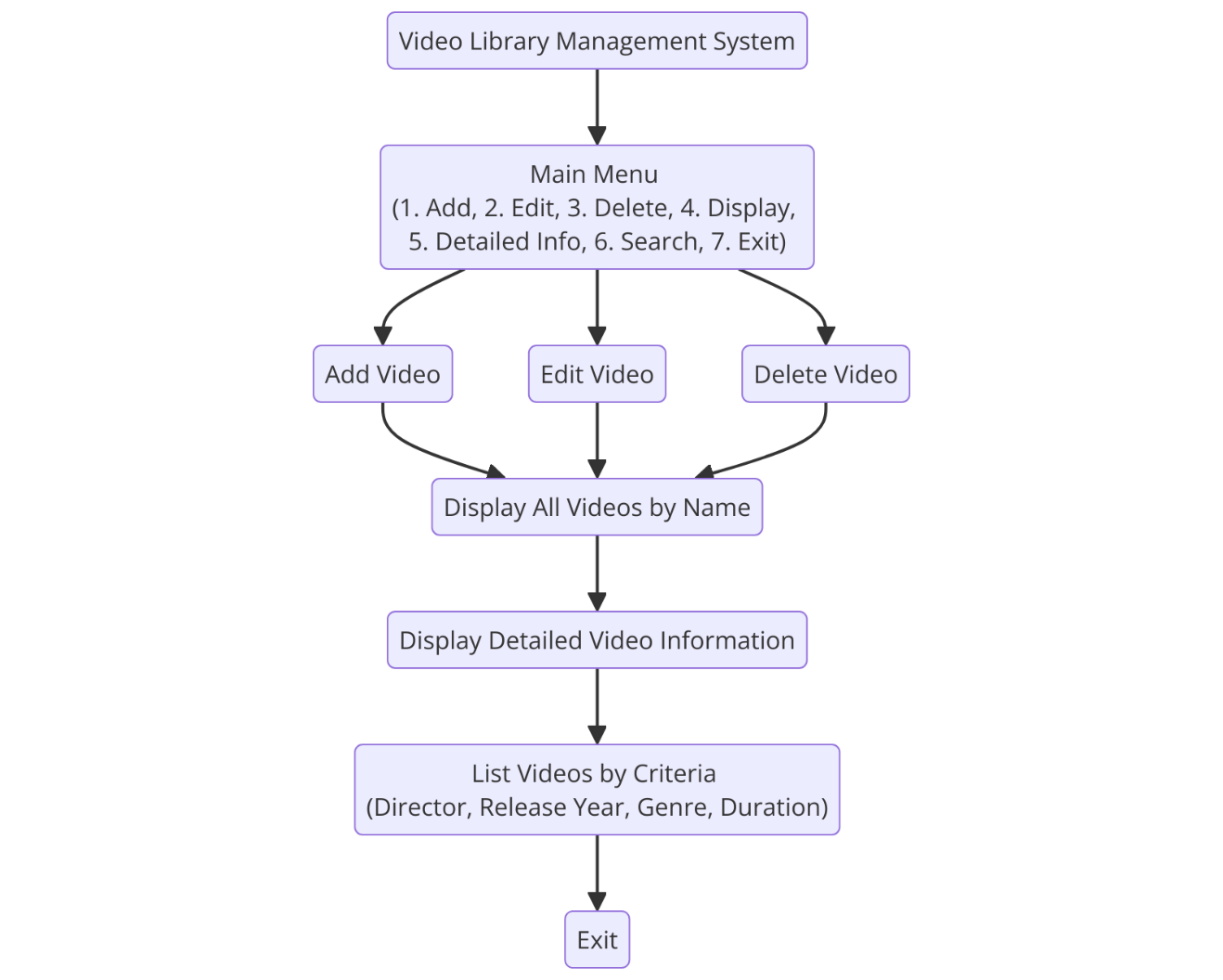
* main(): The main loop that displays the menu and handles user input.
* load\_videos(): Loads video data from the JSON file.
* save\_videos(): Saves video data to the JSON file.
* add\_video(): Prompts the user to enter video details and adds the video to the library.
* edit\_video(): Allows the user to edit details of an existing video.
* delete\_video(): Deletes a video from the library.
* display\_videos(): Displays all videos in the library by title.
* display\_detailed\_info(): Displays detailed information for a specific video.
* list\_videos\_by\_criteria(): Lists videos based on specific criteria (e.g., director, release year, genre).
* search\_videos(): Searches for videos based on user-specified criteria.

## Helper Functions:

* validate\_year(): Validates that the input for the release year is a valid year.
* validate\_duration(): Validates that the duration input is a positive integer.

# Block Diagram

Here's a basic block diagram that explains the flow of the application:



# Contributions

Ral H Ugamah

Andrew Payne

Github Links

@amdp64

@RalHillary