Report for Coursework
(Object-Oriented Programming)

Ho Cong Khanh
UG Scholar, Computing Department, FPT Greenwich University, HCMC, Vietnam
(Email: khanhhcgcs220658@fpt.edu.vn)

Table of Contents

I. Introduction

This is the report of the project I made based on the VideoPlayer COMP 1752 python file given, this coursework follows what I have learned from the course which includes Classes attached for each GUI that operates a specific function. In this coursework, I have reached all the steps and the innovation I developed here is a function to add new videos to the list of videos, a search function, and extract then add new data into a JSON file. I was inspired by some of the digital platforms that have the basic function of adding new videos to the list. Moreover, the Exception method is added to handle the error that appears when the system operates. In brief, all stages of the coursework are tackled and innovated to design a better system with more functionalities for the customer to modify for their use.

II. Design and Development

1. create_video_list.py function

Firstly, for this function, I will provide a picture of the GUIs to demonstrate where the button and entry text for users to type in the input they want and interact with the system:

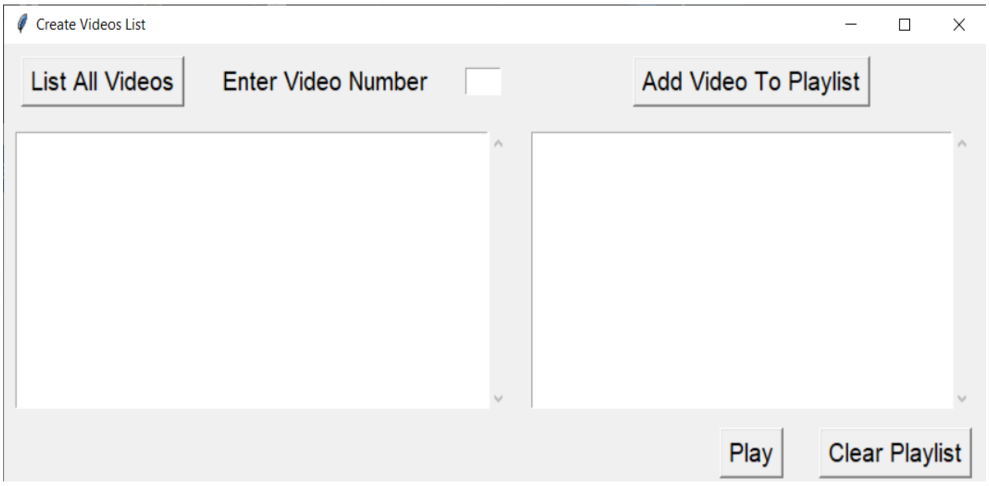
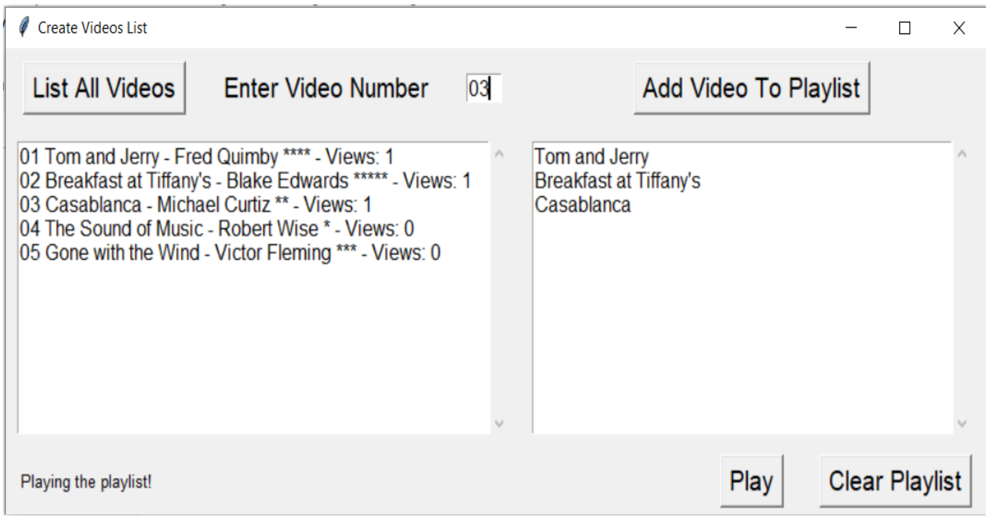


Fig. 1. Create Video List GUI

As you can see, this GUI is similar to the check_videos.py because when I read the requirements, this function requires the user to enter the video number it will check for the validation of whether the number is invalid or valid. If the number is validated the video will be added to the playlist.

For further development, I created a "Clear Playlist" button to wipe out the playlist for the user and added new videos to the playlist, this button is set at the right bottom corner next to the "Play" button. Moreover, for the "Play" button when the users press this, each video added to the playlist will increase the play count of each video added.



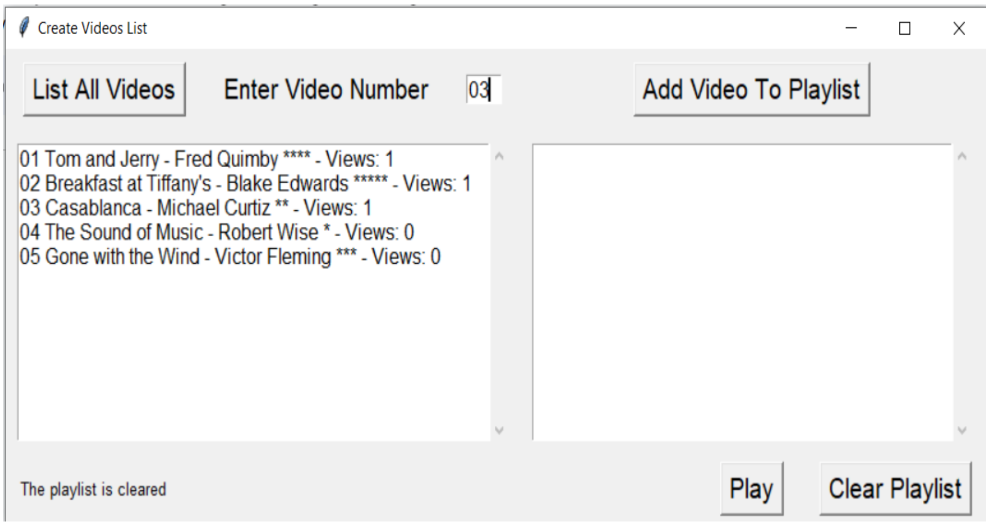
01 Tom and Jerry - Fred Quimby **** - Views: 1
02 Breakfast at Tiffany's - Blake Edwards ***** - Views: 1
03 Casablanca - Michael Curtiz ** - Views: 1
04 The Sound of Music - Robert Wise * - Views: 0
05 Gone with the Wind - Victor Fleming *** - Views: 0

Tom and Jerry
Breakfast at Tiffany's
Casablanca

Playing the playlist!

Fig. 1. 2. Create Video List Operate 1

And here is the screenshot for the “Clear Playlist” button:

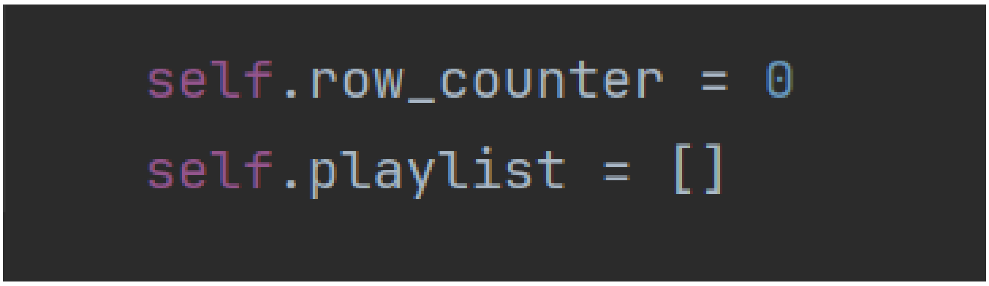


01 Tom and Jerry - Fred Quimby **** - Views: 1
02 Breakfast at Tiffany's - Blake Edwards ***** - Views: 1
03 Casablanca - Michael Curtiz ** - Views: 1
04 The Sound of Music - Robert Wise * - Views: 0
05 Gone with the Wind - Victor Fleming *** - Views: 0

The playlist is cleared

Fig. 1. 3. Create Video List Operate 2

In addition to how I developed the function of this requirement, I will give a screenshot of each function of each button and an explanation, respectively.



```
self.row_counter = 0
self.playlist = []
```

Fig. 2. New Parameters

For these two functions, I created two parameters, row_counter is for formatting rows when adding various videos, and playlist has created a list to store videos.

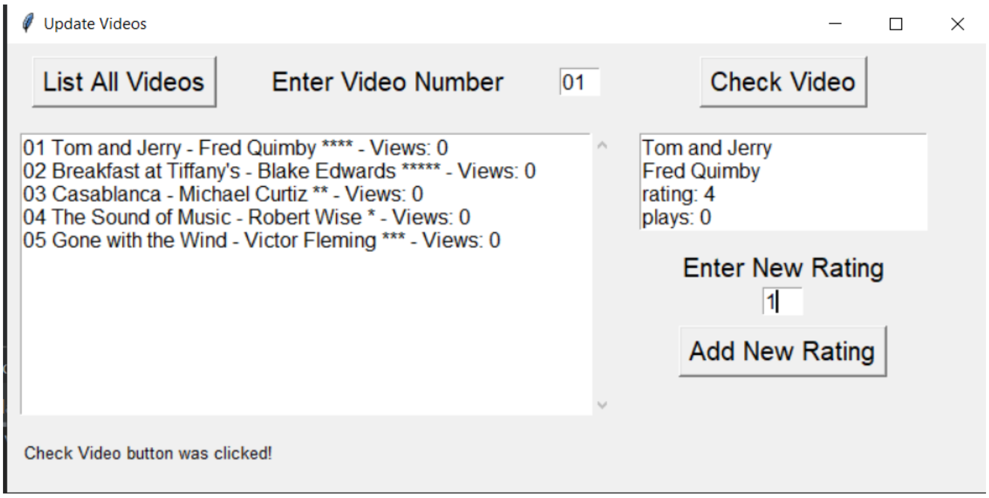
For the “add_video_clicked” function, the key and name are for storing the input, subsequently, conditions are used for defining the logic for the function to add new videos to the list using the built-in function “append()” and “insert()” as I used in the GUI of the playlist is a ScrolledText.

Secondly, I will demonstrate how the “play_playlist_clicked” function works, I called the key again to get the input and still used the conditions to create the logic, and then I called the function “increment_play_count” in the video_library.py.

In brief, this is the first function which is required by the coursework. This code is based on what I learned and provided code, so it is not difficult but needs time to finish. The next section will be the second required function.

2. update_video.py

In the case of this request, for the GUI I have some innovations but still based on check_videos.py. All the basic GUIs of the “check_videos.py” of the main are the same and I added two new elements to the right section. Which is an Entry widget for entering a new rating and a Button widget to add the new rating, the other functions still work the same as the check_video.py.



Enter Video Number: 01

```
01 Tom and Jerry - Fred Quimby **** - Views: 0
02 Breakfast at Tiffany's - Blake Edwards ***** - Views: 0
03 Casablanca - Michael Curtiz ** - Views: 0
04 The Sound of Music - Robert Wise * - Views: 0
05 Gone with the Wind - Victor Fleming *** - Views: 0
```

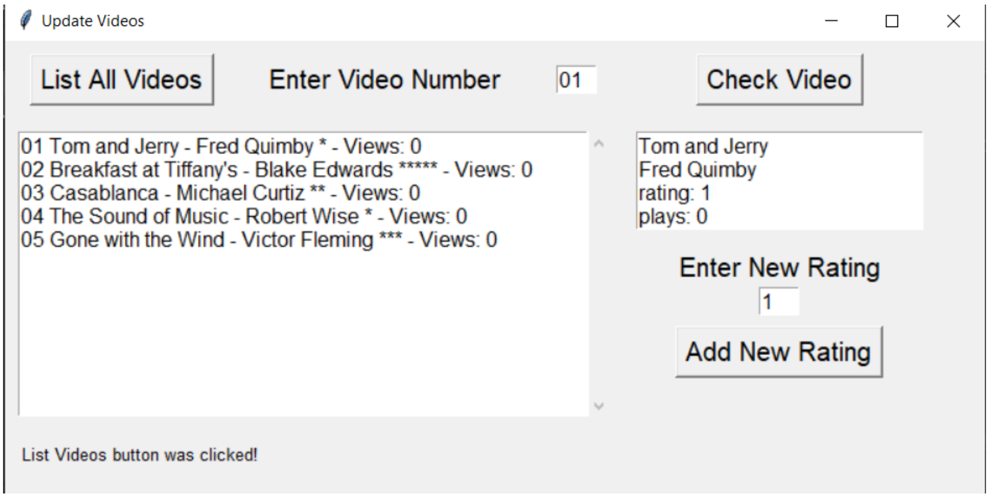
```
Tom and Jerry
Fred Quimby
rating: 4
plays: 0
```

Enter New Rating: 1

Check Video button was clicked!

Fig. 3. Update Video GUI

This is the screenshot when I listed all videos, then checked for video 01 and entered a new rating which was 1 but I didn't click the "Add New Rating button". After clicking the button the rating will then be changed associated with the stars of video 01.



Enter Video Number: 01

```
01 Tom and Jerry - Fred Quimby * - Views: 0
02 Breakfast at Tiffany's - Blake Edwards ***** - Views: 0
03 Casablanca - Michael Curtiz ** - Views: 0
04 The Sound of Music - Robert Wise * - Views: 0
05 Gone with the Wind - Victor Fleming *** - Views: 0
```

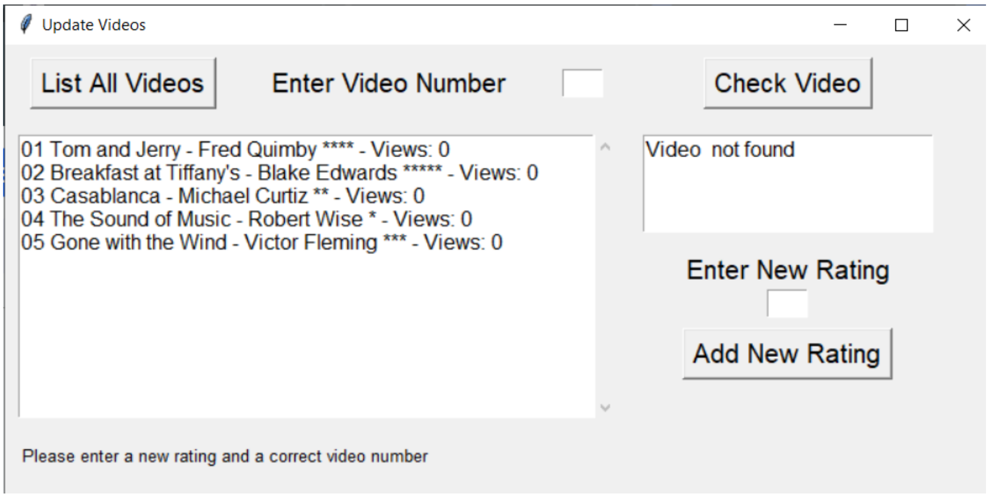
```
Tom and Jerry
Fred Quimby
rating: 1
plays: 0
```

Enter New Rating: 1

List Videos button was clicked!

Fig. 3. 1. Update Video Operate 1

If the the user does not enter the video number and the new rating there will be an announcement to notify the problem.



```
01 Tom and Jerry - Fred Quimby **** - Views: 0
02 Breakfast at Tiffany's - Blake Edwards ***** - Views: 0
03 Casablanca - Michael Curtiz ** - Views: 0
04 The Sound of Music - Robert Wise * - Views: 0
05 Gone with the Wind - Victor Fleming *** - Views: 0
```

```
Video  not found
```

```
Please enter a new rating and a correct video number
```

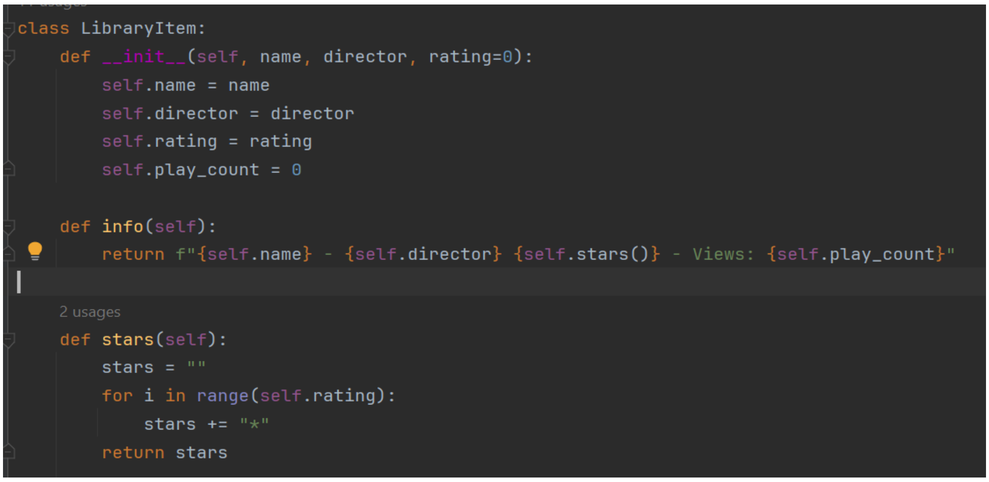
Fig. 3. 1. Update Video Operate 2

For a deeper understanding, I will provide the screenshots of the functions. These two screenshots are for the logic of the requirement. As you can see, the variables for storing the inputs are the same as my previous function, I also commented on how each line of code works for clarification.

3. Innovations

3.1. Views

I added "self.play_count" as views to show how many times the videos are watched.



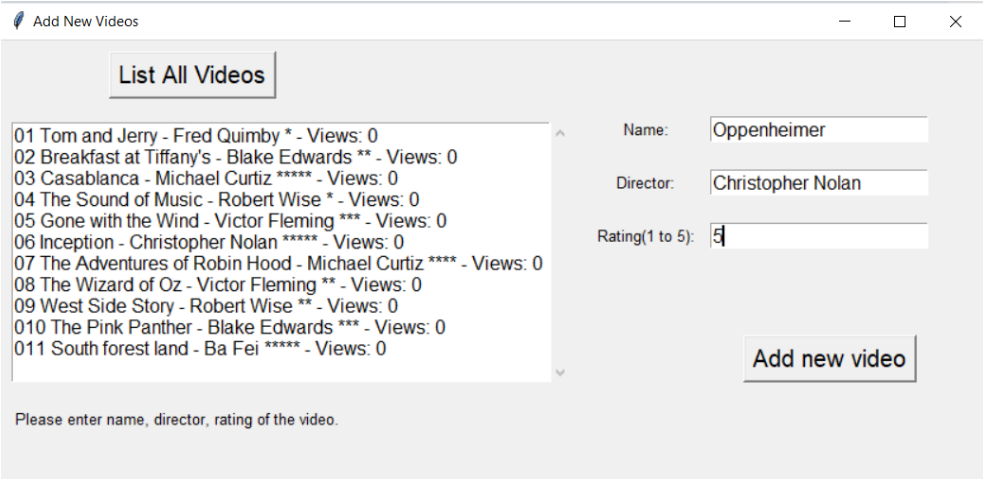
```python
class LibraryItem:
    def __init__(self, name, director, rating=0):
        self.name = name
        self.director = director
        self.rating = rating
        self.play_count = 0

    def info(self):
        return f"{self.name} - {self.director} {self.stars()} - Views: {self.play_count}"

    2 usages
    def stars(self):
        stars = ""
        for i in range(self.rating):
            stars += "*"
        return stars
```

Fig. 4. Library Item Innovation

3.2. add_new_videos.py

For new functionality, I created a basic one to provide a more usable system: adding a new video to the playlist. For the GUIs, I still base on the check_video.py to synchronise the whole system. For a further view of the GUIs, this is the screenshot of this function.

This function requires users to enter the name, director and rating to add a new video.



```
01 Tom and Jerry - Fred Quimby * - Views: 0
02 Breakfast at Tiffany's - Blake Edwards ** - Views: 0
03 Casablanca - Michael Curtiz ***** - Views: 0
04 The Sound of Music - Robert Wise * - Views: 0
05 Gone with the Wind - Victor Fleming *** - Views: 0
06 Inception - Christopher Nolan ***** - Views: 0
07 The Adventures of Robin Hood - Michael Curtiz **** - Views: 0
08 The Wizard of Oz - Victor Fleming ** - Views: 0
09 West Side Story - Robert Wise ** - Views: 0
010 The Pink Panther - Blake Edwards *** - Views: 0
011 South forest land - Ba Fei ***** - Views: 0
```

Name: Oppenheimer

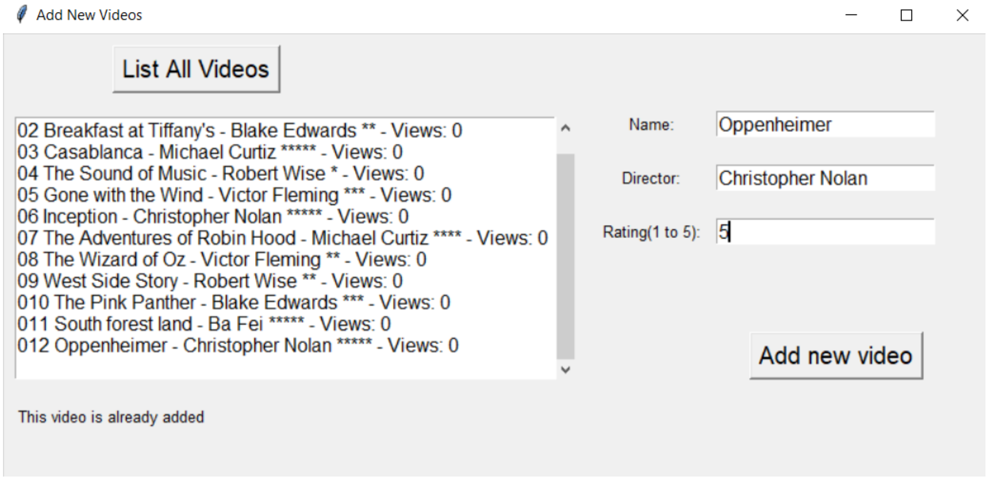
Director: Christopher Nolan

Rating(1 to 5): 5

Please enter name, director, rating of the video.

Fig. 5. Add New Video GUI

If one of the obligated elements is not entered there will be a notification for users. In addition, if the video with the same detail is already added, there will be a label announced for this error.



```
02 Breakfast at Tiffany's - Blake Edwards ** - Views: 0
03 Casablanca - Michael Curtiz ***** - Views: 0
04 The Sound of Music - Robert Wise * - Views: 0
05 Gone with the Wind - Victor Fleming *** - Views: 0
06 Inception - Christopher Nolan ***** - Views: 0
07 The Adventures of Robin Hood - Michael Curtiz **** - Views: 0
08 The Wizard of Oz - Victor Fleming ** - Views: 0
09 West Side Story - Robert Wise ** - Views: 0
010 The Pink Panther - Blake Edwards *** - Views: 0
011 South forest land - Ba Fei ***** - Views: 0
012 Oppenheimer - Christopher Nolan ***** - Views: 0
```

Name: Oppenheimer

Director: Christopher Nolan

Rating(1 to 5): 5

This video is already added

Fig. 5. Add New Video GUI

This code fully commented on how each line works. For the highlighted one, the line of code of how I handle the duplicate error and add new videos. I have created a function in video_library.py, named get_new_video. This function will take the input when it was called and check if the video is added, then stored in the library.

3.3. JSON file and Save

In this part I created a database using JSON and two functions to read and store new data back in the database, that's why among the above code there will be a "global library" called to store new data. These are the two functions.

They will import data from the JSON file provided in the attached file and then save new ratings and play counts, new videos from functions of stage two.

After all the modifications, I created a function called Save for users to save their new videos or ratings into the JSON file.

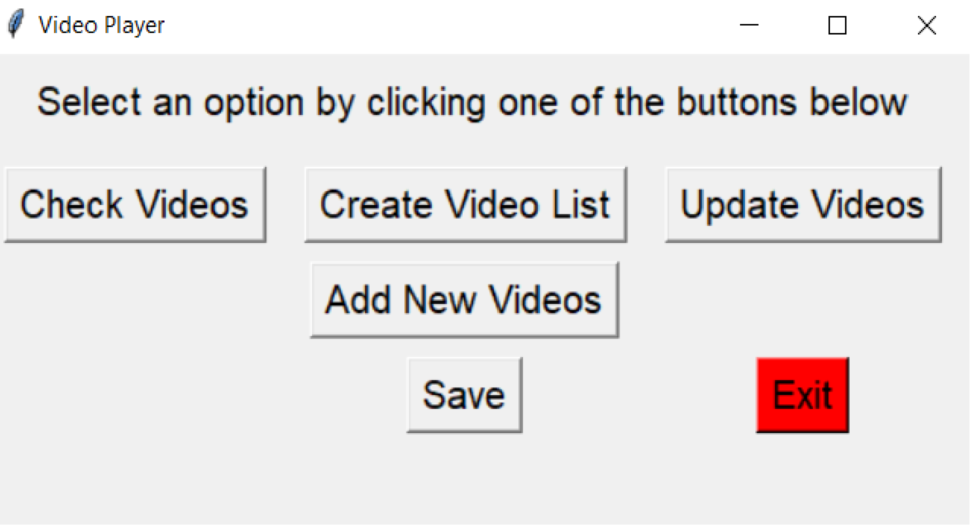
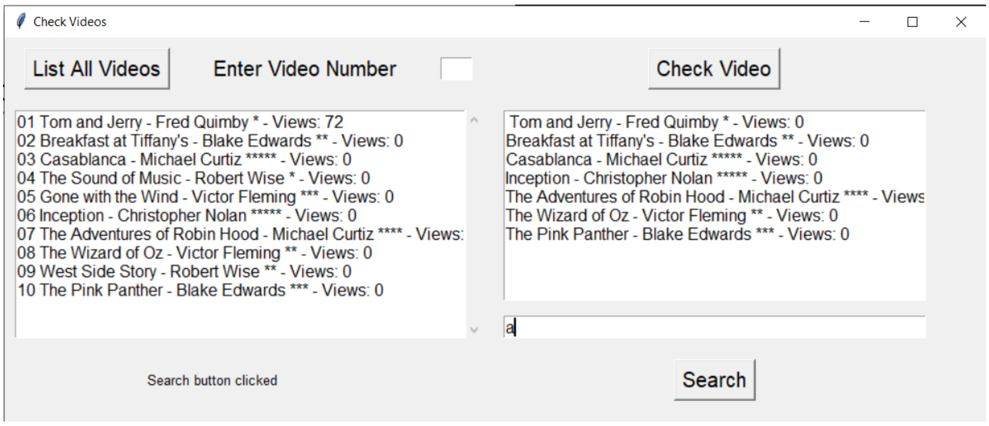


Fig. 5. Add New Video GUI

3.4. Search function

In this section, I created a method which is a search bar for users to enter a letter, a word or even a name and director or movie, and then it will display the video that matches the searched word.



```
01 Tom and Jerry - Fred Quimby * - Views: 72
02 Breakfast at Tiffany's - Blake Edwards ** - Views: 0
03 Casablanca - Michael Curtiz ***** - Views: 0
04 The Sound of Music - Robert Wise * - Views: 0
05 Gone with the Wind - Victor Fleming *** - Views: 0
06 Inception - Christopher Nolan ***** - Views: 0
07 The Adventures of Robin Hood - Michael Curtiz **** - Views:
08 The Wizard of Oz - Victor Fleming ** - Views: 0
09 West Side Story - Robert Wise ** - Views: 0
10 The Pink Panther - Blake Edwards *** - Views: 0
```

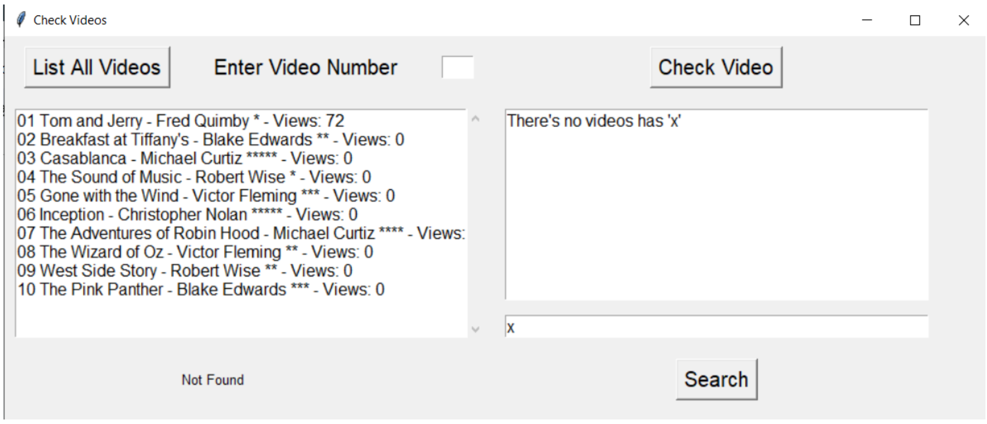
```
 Tom and Jerry - Fred Quimby * - Views: 0
Breakfast at Tiffany's - Blake Edwards ** - Views: 0
Casablanca - Michael Curtiz ***** - Views: 0
Inception - Christopher Nolan ***** - Views: 0
The Adventures of Robin Hood - Michael Curtiz **** - Views
The Wizard of Oz - Victor Fleming ** - Views: 0
The Pink Panther - Blake Edwards *** - Views: 0
```

a

Search button clicked

Fig. 6. Search Function Operate 1

In contrast, if the string is unable to be found, there will be a label to announce it.



```
01 Tom and Jerry - Fred Quimby * - Views: 72
02 Breakfast at Tiffany's - Blake Edwards ** - Views: 0
03 Casablanca - Michael Curtiz ***** - Views: 0
04 The Sound of Music - Robert Wise * - Views: 0
05 Gone with the Wind - Victor Fleming *** - Views: 0
06 Inception - Christopher Nolan ***** - Views: 0
07 The Adventures of Robin Hood - Michael Curtiz **** - Views:
08 The Wizard of Oz - Victor Fleming ** - Views: 0
09 West Side Story - Robert Wise ** - Views: 0
10 The Pink Panther - Blake Edwards *** - Views: 0
```

```
There's no videos has 'x'
```

x

Not Found

Fig. 6. 1. Search Function Operate 2

III. Testing and Faults

For the test table, I created three for testing the GUIs of each function, I used a small range of tests. Due to the fact that there is a limited range of functions in my system. The test table gives a deeper view of how each input is handled by my system. For faults, that appeared on my system, there are several but I fixed all the bugs, so I will choose two out of them to demonstrate what I have encountered during creating the system.

Firstly, when pressing the "Play" button from a playlist of up to 2 videos it will only increase the latest one. The reason for the failure is that of logic it put on the function did not take all playlists but only took the last one to increase the play count. Then I found a solution and tackled the problem. Secondly, when creating the search function, there is a problem which is the search string can not be taken when the input is uppercase. However, the solution is solved using the built-in function "lower()".

IV. Conclusion, Further Development and Reflection

In brief, I have created a fully functional system which can serve as a basic text-only video platform similar to YouTube—every innovation, inspired by various kinds of digital platforms in recent years. However, due to my lack of coding skills and experience, I cannot develop a system that can be worked more efficiently and visually.

If I have three more months, I can develop a video storage and poster to provide more detailed information for users to choose a video. Moreover, I can create different playlists for users to add and save to a library and then users can choose the playlist with videos they want to watch. Briefly, I think I can make it more visual and usable for users.

Answering question (a), I believe that I learned the way to handle a system and also the way to create and develop a coding project, with tests, reports and source code. For the most difficult parts, I consider it must be when creating a test, specifically a test table. Because there are lots of cases that can happen when a function is operating and I can not catch all the cases and put them in the test, so that error can occur, then customer feedback can be a feasible solution to acquire bugs and fix them. The easiest section, I think is stages 1 and 2 of the requirements, this part requires only comments on how the function works and GUIs, for me this is easy because GUIs demand the knowledge of grid() to align all the interfaces.

Question (b), as answered above, gives me a small insight into how to develop a project for code. Moreover, this project helped me to understand object-oriented programming and logic in coding. Additionally, utilisation of the built-in function in Python, outside databases such as CSV, and JSON files. This is a work experience for the future and I can use this coursework to test new functions and add some new things to try. In the long term, this can help me increase my knowledge of how to implement code, and databases together. Lastly, I can use this coursework to test new ideas and improve my skills.

V. Appendix

1. The commented version

Is in the attached code.

2. Test Table

2.1. Unit testing for LibraryItem

For the library_item.py, this Class has three functions and I created a LibraryItem_test.py to test this Class. Using the pytest which I have learned to build this, this is the screenshots of the file. This unit test is simple but still covers to test how the library_item.py works.

2.2. Testing for GUI of create_video_list.py

GUI	Sample Input	Sample Action	Expected output	Actual output as expected
Create Video List	"01" in input_txt	"Add Video To Playlist" button pressed	Video 01 added to the playlist	yes

Create Video List	Video 01 in the playlist	"Play" button pressed	Video 01's views increase by 1	yes
Create Video List	nothing in input_txt	"Add Video To Playlist" button pressed	"Invalid video number" in the label field	yes
Create Video List	Video 01 in the playlist	"Clear Playlist" button pressed	"The playlist is cleared" in in the label field and the playlist cleared.	yes
Create Video List	nothing in input_txt	"Play" button pressed	"the playlist is empty" in the label field.	yes

2.3. Testing for GUI of update_video.py

GUI	Sample Input	Sample Action	Expected output	Actual output as expected
Update Videos	"one" in input_txt	"Add New Rating" button pressed	"Please enter a number from the list of videos not a string!" in the label field	yes
Update Videos	"0" in new_rating	"Add New Rating" button pressed	"Invalid rating! Please enter a number from 1 to 5." in the label field	yes
Update Videos	nothing in input_txt and new_rating	"Add Video To Playlist" button pressed	"Invalid video number" in the label field	yes
Update Videos	Video 01 in input_txt and new rating is 1	"Add New Rating" button pressed	"New rating added" in in the label field and video 01's rating is 1.	yes
Update Videos	Video 01 in input_txt and new rating is 1	"List All Videos" button is pressed	Video 01's star is *.	yes

2.4. Testing for GUI of add_new_video.py

I have some clarifications about this test, I considered the video's name and director can be numbered or alphanumeric so the GUI will take the details as both.

GUI	Sample Input	Sample Action	Expected output	Actual output as expected
Add New Videos	"South forest land" in name "Ba Fei" in director "5" in rating	"Add New Video" button pressed	"Video South forest land added successfully!" in the label field	yes
Add New Videos	"South forest land" in name "Ba Fei" in director "5" in rating	"Add New Video" button pressed	"This video is already added" in the label field	yes
Add New Videos	nothing in input	"Add New Video" button pressed	"Please enter name, director, rating of the video." in the label field	yes
Add New Videos	"0" in rating	"Add New Video" button pressed	"New rating added" in in the label field and video 01's rating is 1.	yes
Add New Videos	" " in name	"Add New Video" button pressed	"Please enter name of the video." in the label field	yes
Add New Videos	" " in director	"Add New Video" button pressed	"Please enter director of the video." in the label field	yes
Add New Videos	" " in rating	"Add New Video" button pressed	"Please enter rating from 1 to 5 of the video." in the label field	yes

3. Coursework Google Drive Link

https://drive.google.com/drive/folders/1vTFz2mXrMXGTjPU-UkNEKh5iAEzZ26a2?usp=sharing

This is the link direct to my file that contains the coursework. Thank you!