SDEV400 6381: Secure Programming in the Cloud

Homework 4: Lost Film Database

Mark Kardash

Professor Nathan Braun

University of Maryland Global Campus

3 March 2024

This document will contain the description, test cases, and test case screenshots for my SDEV400 Homework 4 application.

Application Description:

For my Homework 4 assignment, I created a program that acts as a database for lost films.

What is a lost film?

A lost film is a motion picture that is no longer available to view, either due to inefficient storage (Ex. Storage in a damp environment may damage tapes), deletion (ex. To use the “freed up” tape for new recordings), disasters (ex. A fire at the studio), disappearance, and other reasons. A film is only considered “lost” when no copies of it exist anywhere, including studio and TV channel archives. Most early silent films are considered permanently lost.

How did the idea come about?

The idea came from my personal fascination with lost media. I explore all sorts of lost media, whether it be songs, movies, TV shows, etc. The last two fascinate me the most, perhaps due to the amount of work and people that goes into them. It baffles me that something so many people worked so hard on can one day just disappear. Moreover, lost films are relics from various eras, containing elements of lifestyles very different from our own. I would very much like to one day find a piece of lost media, and trace it to its creators, to give them the credit they deserve. For me, it is not just about finding content, but preserving cultural heritage.

When starting this assignment, an interesting thought popped into my mind: “Imagine there is a secret organization/society out there that has access to all the different films we consider lost today?” That is when I decided to create the International Database of Lost Films, a mock archive containing the titles of several real-life lost movies, along with the year they were filmed, the director’s name, and a unique ID number.

How does it work?

The database contains a list of films called the “Film\_Catalogue”. For this assignment, the number has been limited to five. Each film is presented with a unique ID number, a title, a release year, and the name of its director. In the main Python file of the app, a user guide, along with a main menu, is displayed every time the application is launched. The user is prompted to enter a number that corresponds to a particular action in the main menu. The actions include: Creating the “Film\_Catalogue” table, creating a repository (s3 bucket) of recently found films called “found-film-repository”, placing a specially created “film” (In actuality, a “.txt” file), into the repository, looking up a film title based on its ID and release year, downloading the “film” onto my computer, deleting the “film” from the repository, and exiting the application. Some aspects will be explained in more detail below.

Additional Explanation: The “Found Film Repository”

The Found Film Repository is supposed to be a place where a user (presumably, someone from the public) can report and place a lost film they recently managed to find. In the assignment version of the app, it is just an s3 bucket, where a “.txt” file is uploaded, standing in for the “film”. The name of the file has already been pre-defined inside the function. In real life, the repository would work as follows:

1. A user accesses the Found Film Repository to place a film (in video format) they recently found in it.
2. They are prompted for its title, release year, and director.
3. The Repository accepts the film and notifies the user with a message.
4. The people in charge of the database review the film inside the repository, and decide what to do with it next (Ex. Make it known to the public, dismiss it as a hoax, ect.).
5. An additional option would be to ask the user for their contact information, such as email address, if additional clarification is needed about the film.

Additional Explanation: Looking up a Film

If the user enters “4” into the initial prompt, they will be prompted to enter the ID number and release year of the film they are looking for. In return, they will receive information about the title of the film, and who it was directed by. The program will then ask the user whether they would like to look for a film again. A response of “y” will result in the film lookup prompt appearing again, while a response of “n” will result in the program exiting the “Found Film Repository”, and returning to the main menu.

Test Cases:

A total of 10 test cases are needed to properly and completely test the functionality of my application. Their results are documented below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case #: | Input: | Expected Output: | Actual Output: | Pass/Fail: |
|  | 1 | “Table “Film\_Catalogue” created successfully”” appears. User guide and menu appear again, asking the user for another selection.  New table visible in DynamoDB. | “Table “Film\_Catalogue” created successfully”” appears. User guide and menu appear again, asking the user for another selection.  New table visible in DynamoDB. | Pass |
|  | 2 | “Bucket “found-film\_repository” has been created successfully” message appears. User guide and main menu displayed again. User asked to make another selection. Bucket visible in s3. | “Bucket “found-film\_repository” has been created successfully” message appears. User guide and main menu displayed again. User asked to make another selection. Bucket visible in s3. | Pass |
|  | 3 | “Film Metropolis.txt has been successfully added to found-film-repository" message appears. User guide and main menu displayed again. User is asked to make another selection. File visible in bucket. | “Film /home/ec2-user/environment/SDEV400Homework4MarkKardash/files/Metropolis.txt has been successfully added to found-film-repository" message appears. User guide and main menu displayed again. User is asked to make another selection. File is visible in s3. | Pass |
|  | 4 | “Welcome! You have now accessed the lost film catalogue:” message displayed. User is asked for the ID and release year of the film they are looking for, given the title and director of the film, and asked if they would like to look up another one. | “Welcome! You have now accessed the lost film catalogue:” message displayed. User is asked for the ID and release year, enters correct data, but the program does not recognize the film in question. ValidationException error thrown. User is prompted to look for a film again. | Fail |
|  | 4, 0003, 1936, y | After entering “y” as a response to search for a film again, the user gets prompted for the ID and release year. | After selecting “y” as a response to search for a film again, the user first gets prompted for the release year, receives an error message that a film with such criteria does not exist in the database, then gets prompted for the ID, and then the release year again, only to be shown the same message. | Partial pass |
|  | 4, 0002, 1912, n | After entering “n” for the prompt to search for a film again, the user gets an exit message from the film catalogue (“Thank you for using the film catalogue. You will now  be redirected to the main menu. Have a great day!”), and is redirected back to the main page, with the user guide and main menu appearing. | After entering “n” for the prompt to search for a film again, the user gets an exit message from the film catalogue (“Thank you for using the film catalogue. You will now  be redirected to the main menu. Have a great day!”), and is redirected back to the main page, with the user guide and main menu appearing. | Pass |
|  | 5 | “Metropolis.txt” file downloaded from bucket “found-film-repository" onto the user’s computer, and saved as “Metropolis-download.txt”.  Confirmation message appears. | “Metropolis.txt” file downloaded from bucket “found-film-repository" onto the user’s computer, and saved as “Metropolis-download.txt”.  Confirmation message appears. | Pass |
|  | 6 | “Metropolis.txt” file successfully deleted from the “found-film-repository". Confirmation message appears. File is no longer in bucket. User guide and main menu displayed again. User prompted for another selection. | “Metropolis.txt” file successfully deleted from the “found-film-repository". Confirmation message appears. File is no longer in bucket. User guide and main menu displayed again. User prompted for another selection. | Pass |
|  | 7 | Program exited with “Thank you for using the International Database of Lost Films! Come again!” goodbye message. | Program exited with “Thank you for using the International Database of Lost Films! Come again!” goodbye message. | Pass |
|  | 8 | “Oops! That selection was invalid.” message appears. User guide and main menu displayed again. User prompted for another selection. | “Oops! That selection was invalid.” message appears. User guide and main menu displayed again. User prompted for another selection. | Pass |



Figure 1: Successfully creating DynamoDB table.

As we can see in the above screenshot, the program confirms that we have successfully created the “Film\_Catalogue” DynamoDB table.

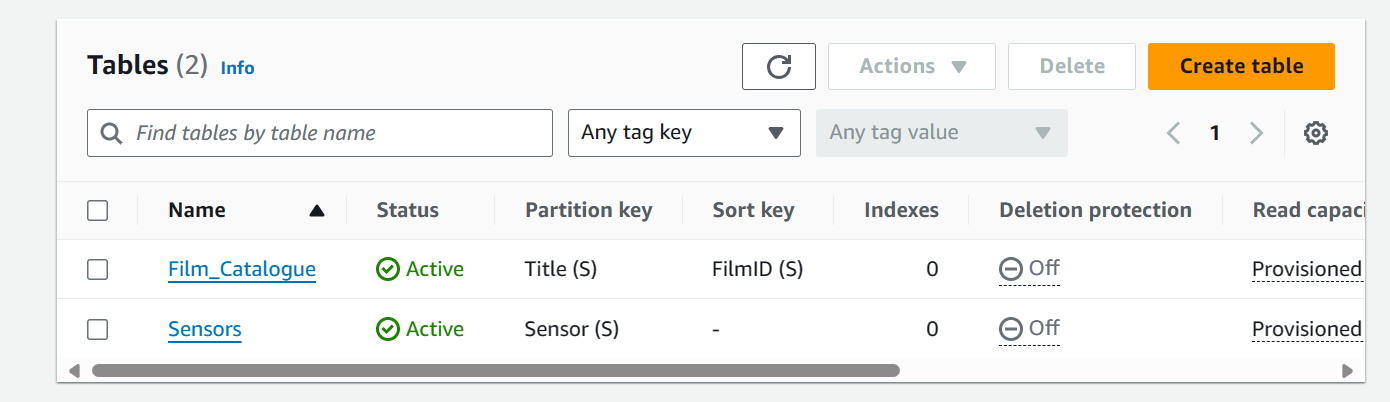


Figure 2: New DynamoDB table confirmation

The creation of the new “Film\_Catalogue” table can also be confirmed by seeing it displayed in DynamoDB.



Figure 3: s3 bucket created successfully.

Above, we can see a confirmation of having successfully created the “found-film-repository" bucket.

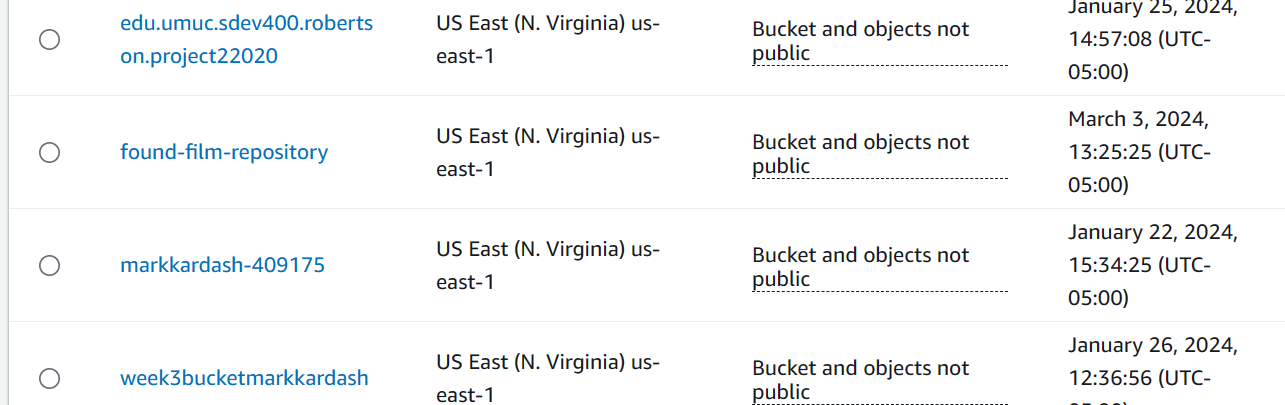


Figure 4: Bucket visible in s3

We can also confirm this by finding the bucket in s3.

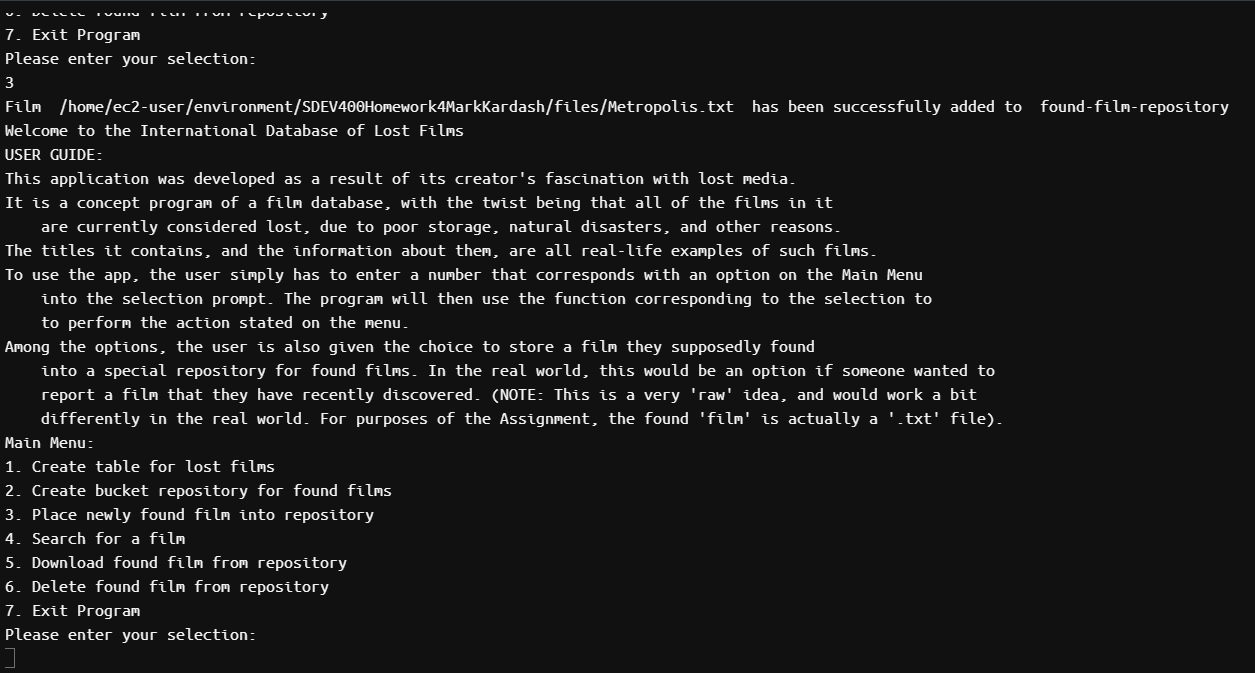


Figure 5: “Film” added to bucket

Although the confirmation message listed the full directory as opposed to just the file name, we can still say that the file was successfully added to the “found-film-repository" bucket.

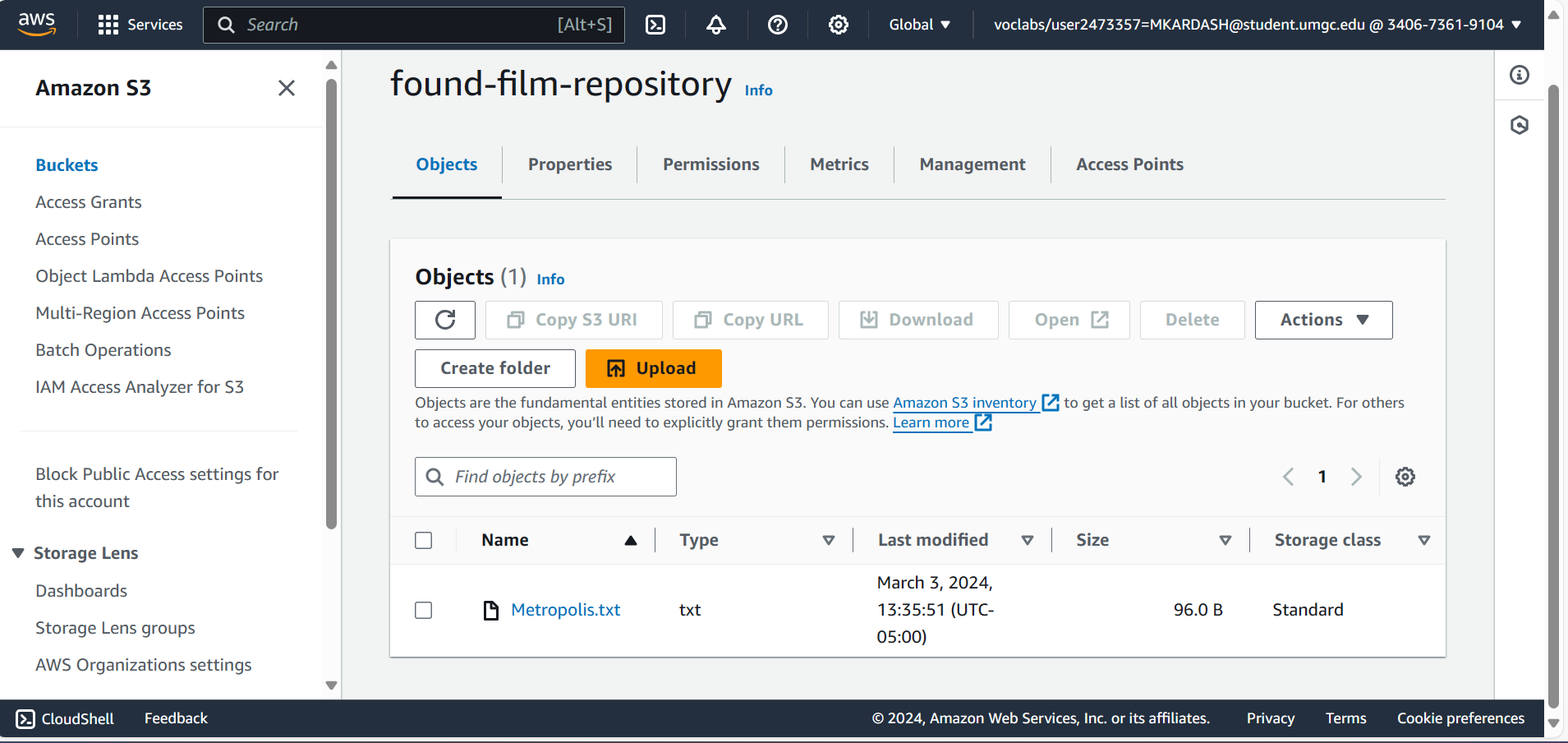


Figure 6: File visible in s3 bucket.

As a confirmation, we can see the “Metropolis.txt” file inside of the bucket.

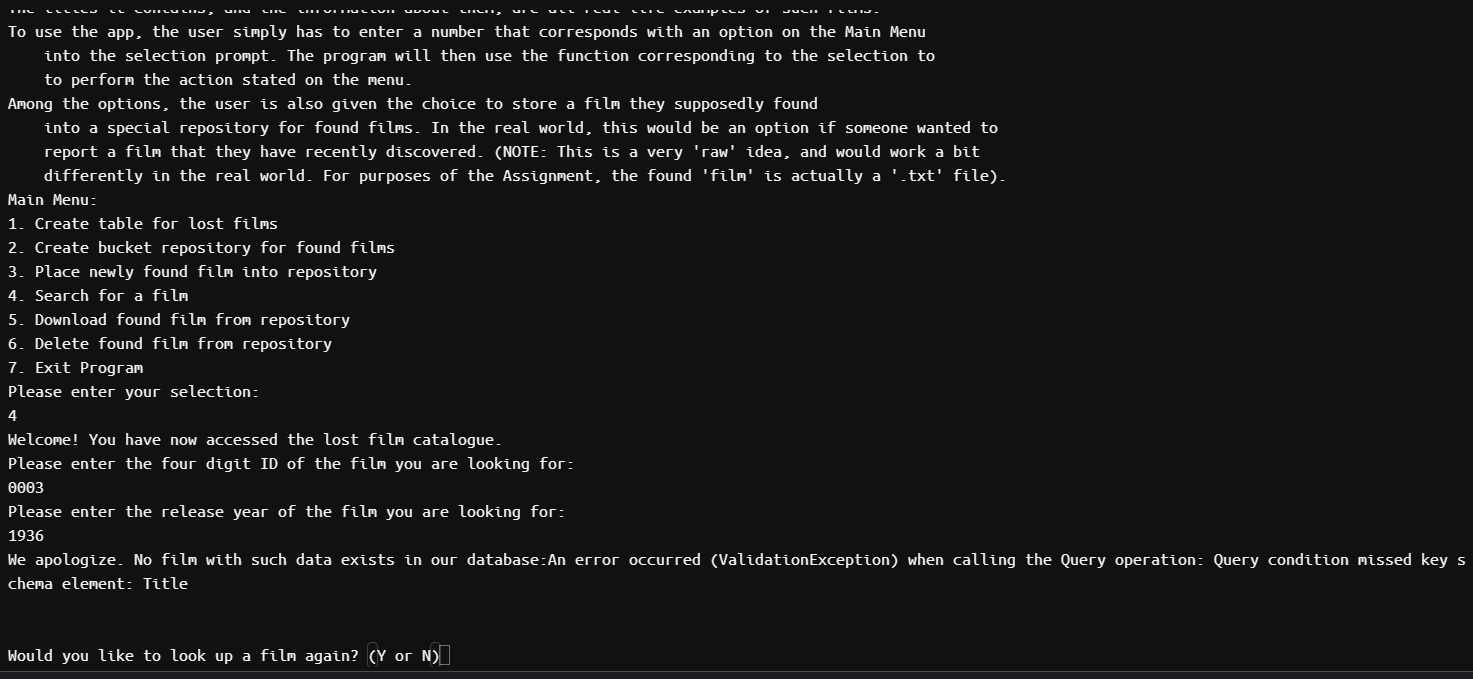


Figure 7: Program did not find film with valid data.

In this case, even when it was provided with an ID and year that were indeed included in the film catalogue, the program did not recognize the data about the sought film.

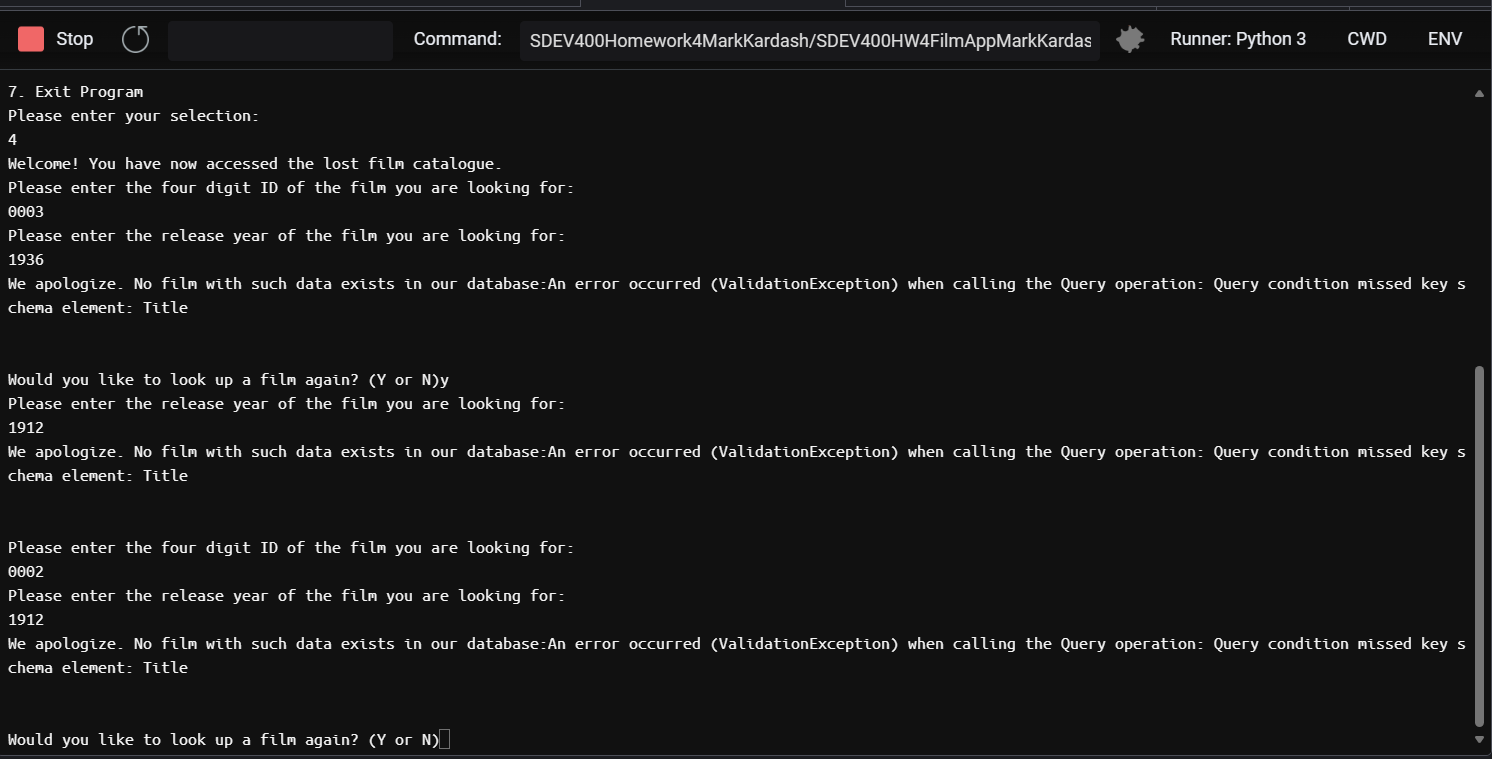


Figure 8: User prompted for the same data multiple times, only to receive error message

I only consider it a partial pass because the program did prompt the user for the expected information. However, it does so in an incorrect order, repeating one of the prompts, and throwing the error message even earlier than expected.

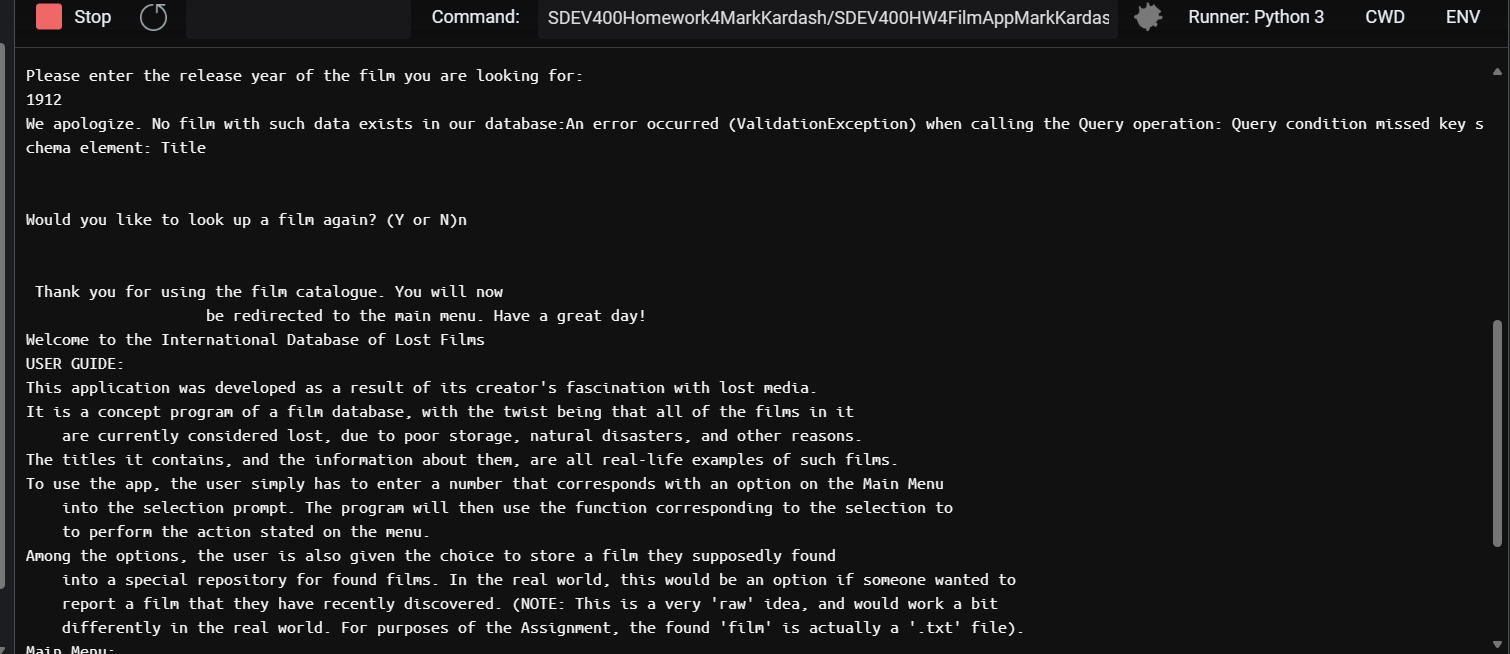


Figure 9: User exited from the film catalogue and redirected to main menu.

Entering “n” to decline another film search has the desired effect, with the program exiting the film catalogue, displaying an exit message, and redirecting the user back to the guide with the main menu.

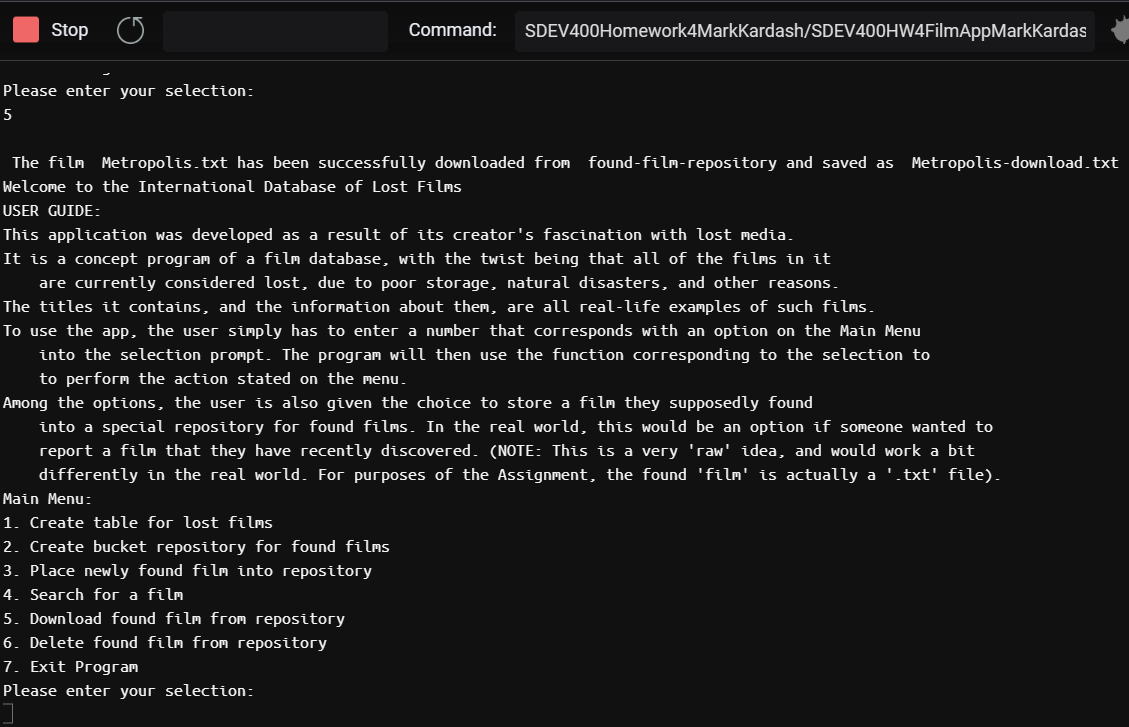


Figure 10: Confirmation message for download of “Metropolis” file.

A message has confirmed the successful download of “Metropolis.txt”, and its saving on the computer with the name of “Metropolis-download.txt”.

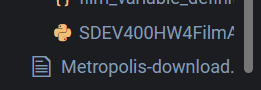


Figure 11: Downloaded file visible on user’s computer.

Seeing the downloaded file on my computer, I can now definitely know that the operation was a success.

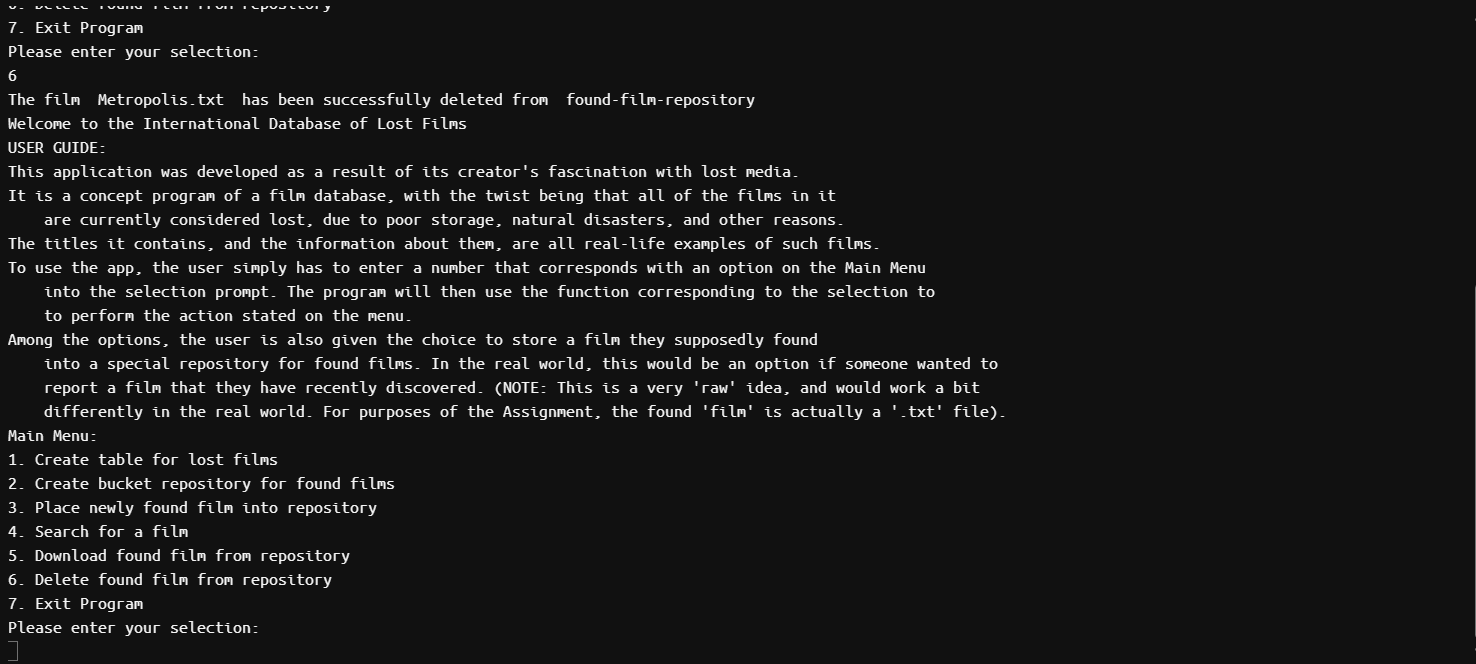


Figure 12: Confirmation of file deletion from bucket

The deletion of the file “Metropolis.txt” from the “found-film-repository" bucket is confirmed by the message above.

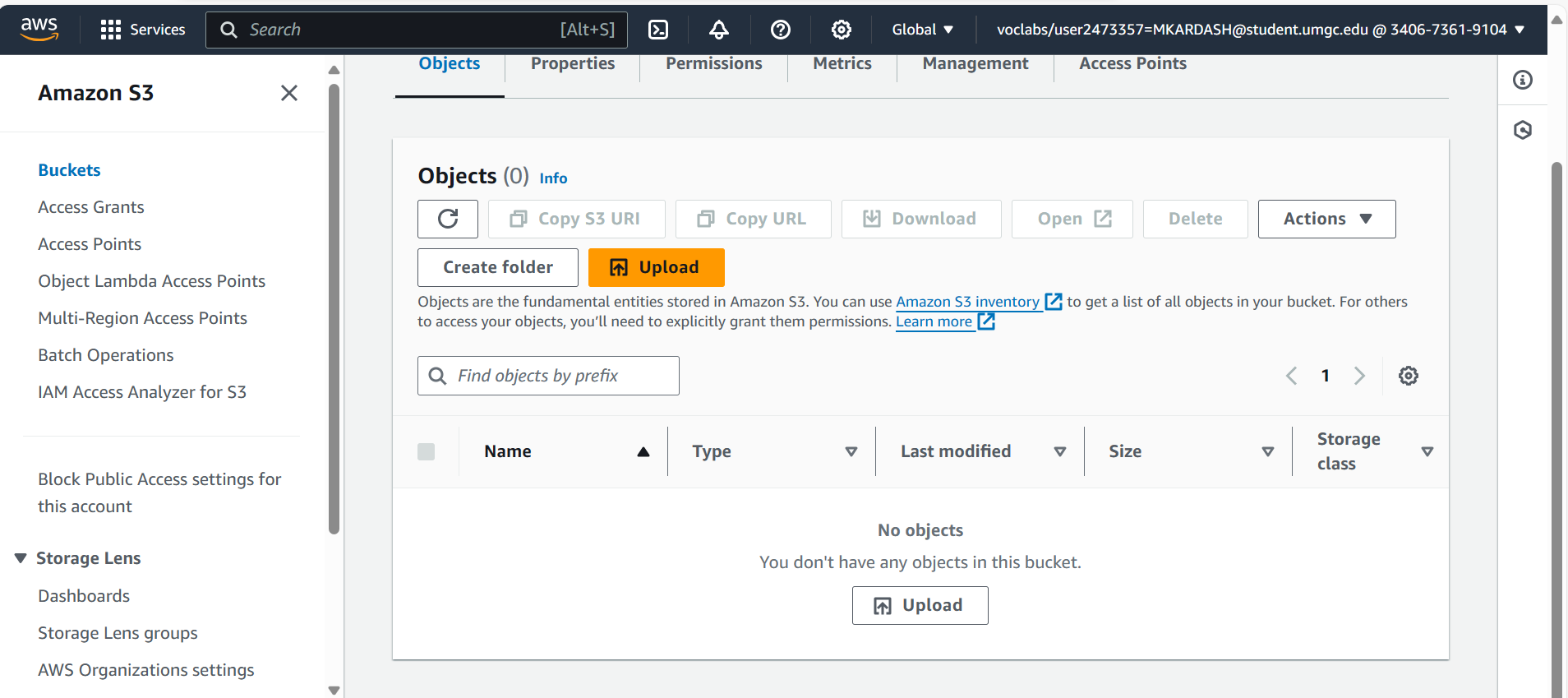


Figure 13: File is no longer in bucket.

This can also be confirmed by seeing that the file is no longer in the bucket.

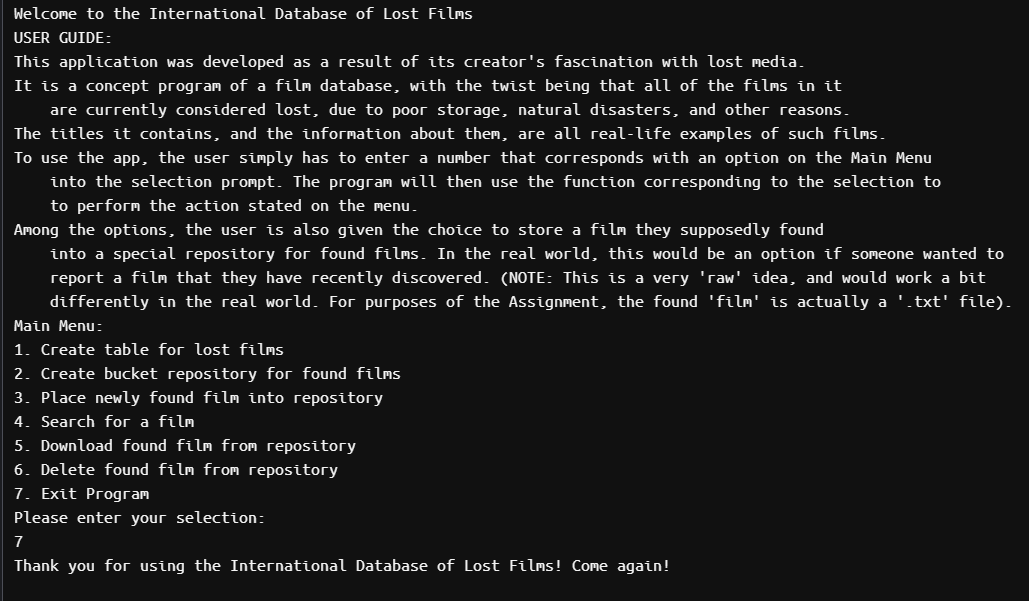


Figure 14: Exiting the program.

Upon entering “7” into the selection prompt, the program successfully exits with a goodbye message.

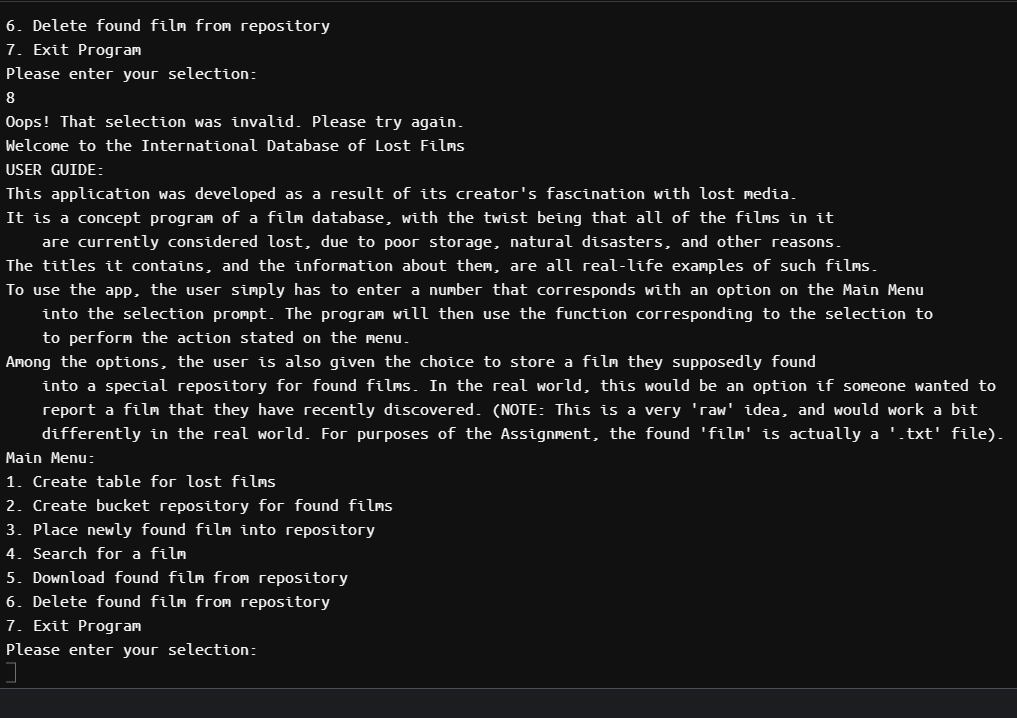


Figure 15: Invalid selection with appropriate response

Entering an invalid selection into the initial prompt results in an error message, followed by the user guide and main menu being displayed again, and the user being asked to make another selection.

Conclusion:

Although I expected this assignment to be the “final, huge, ultra-difficult project” of the course, it was, ironically, probably the easiest of all. The knowledge I had gained from previous assignments helped me put a decent application together, one that passed 8 out of 10 test cases with flying colors. The only things I still did not figure out was why the program would not recognize films from the DynamoDB table, and why it would incorrectly display prompts in a repeated film lookup. Overall, this was an interesting project that required combining some, if not all, of the things learned throughout the course.