# Lab 03 – Python project

*Imanuel Leiserowitz*

URL to GitHub repository:

https://github.com/leunami420/Lab03_YT_SC_Downloader.git

## 1. App idea & use case diagram

The app should be a YouTube and SoundCloud downloader that allows users to download audio and video files from these platforms by entering a valid URL. The user can select the format of the downloaded file (audio or video) using an option menu, and the app will display a message indicating the status of the download process. Additionally, when the download is finished, the app will open the folder where the file was downloaded to using the default file explorer for the user's operating system.

```
              +---------------+
              | User          |
              +---------------+
                     |
       +----------------------------------+
       | Use Case: Download Media Content |
       +----------------------------------+
                     |
     +------------------------+---------------------+
     | Preconditions:                               |
     | - User has a valid URL.                      |
     +------------------------+---------------------+
                     |
     +------------------------+---------------------+
     | Basic Flow of Events:                        |
     | 1. User enters a URL.                        |
     | 2. User selects a format.                    |
     | 3. User clicks "download".                   |
     | 4. The application                           |
     |    downloads the media                       |
     |    content.                                  |
     +------------------------+---------------------+
                     |
     +------------------------+---------------------+
     | Postconditions:                              |
     | - The media content is                       |
     |   downloaded to the user's                   |
     |   computer                                   |
     +------------------------+---------------------+
```

## 2.+3.

First we import all the libraries we need.

```
from __future__ import unicode_literals
import os
import subprocess
import tkinter
import customtkinter
import youtube_dl
```

For the GUI I´ll be using customtkinter, which makes it easy to build GUIs in python. I studied the Documentation aswell as the Example Scripts to learn how to build the Interface.

```
class App(customtkinter.CTk):

    def __init__(self):
        super().__init__()
        self.selectedFormat = "Video"
        self.geometry("720x480")
        self.title("YouTube and SoundCloud Downloader")



        self.label = customtkinter.CTkLabel(master=self,
text="YouTube and SoundCloud Downloader")
        self.label.pack()

        self.url_entry = customtkinter.CTkEntry(master=self,
placeholder_text="Enter a valid URL")
        self.url_entry.pack(padx=10, pady=10)

        self.label = customtkinter.CTkLabel(self, text="Enter a
Soundcloud or Youtube URL")
        self.label.pack()

        self.optionmenu = customtkinter.CTkOptionMenu(self, val-
ues=["Audio", "Video"],
                                                      command=option-
menu_callback)
        self.optionmenu.set("Select format")
        self.optionmenu.pack(padx=10, pady=10)

        self.button_find = customtkinter.CTkButton(self,
text="Download", command=button_find_event)
        self.button_find.pack(padx=10, pady=10)
```

CTk: This widget is used as the base class for the App class, allowing for a custom appearance and additional functionality.

CTkLabel: This widget is used to display the text "YouTube and SoundCloud Downloader" at the top of the GUI.

CTkEntry: This widget is used to allow the user to input a URL for downloading content from either Soundcloud or YouTube.

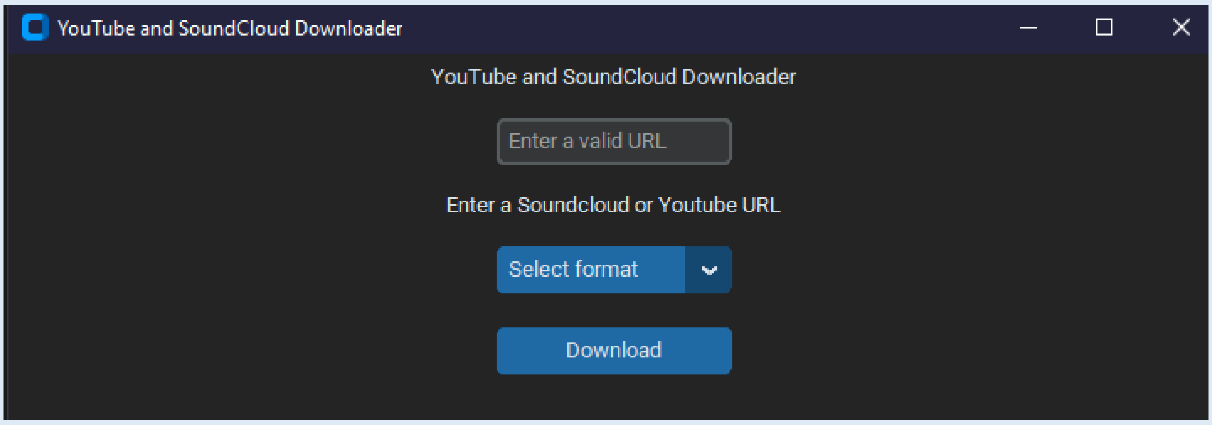
CTkOptionMenu: This widget is used to allow the user to select the download format (audio or video) from a dropdown menu.

CTkButton: This widget is used to trigger the download of the content at the specified URL. When the button is clicked, the button_find_event function is called, which downloads the content and updates the label with the status of the download.

Now I have a basic GUI that I can run from my main.py file using

```
app = gui.App()
app.mainloop()
```

.


YouTube and SoundCloud Downloader
YouTube and SoundCloud Downloader
Enter a valid URL
Enter a Soundcloud or Youtube URL
Select format
Download

I tried to use the modules pytube and SClib to download the files, but after many failed implementation-attempts, I ended up using youtube_dl, which can Download from both Soundcloud and Youtube.

**The youtube_dl module is used in the app to download content from YouTube and Soundcloud. Here's how it's used:**

- First, the appropriate ydl_opts dictionary is constructed based on the selected download format (audio or video).

  For audio:

```
if self.selectedFormat == "Audio":
    ydl_opts = {
        'format': 'bestaudio/best',
        'outtmpl': './downloads/%(title)s.%(ext)s',
        'noplaylist': True,
        'progress_hooks': [my_hook],
    }
```

For video:

```
if self.selectedFormat == "Video":
    ydl_opts = {
        'format': 'best',
        'outtmpl': './downloads/%(title)s.%(ext)s',
        'noplaylist': True,
        'progress_hooks': [my_hook],
    }
```

- The with statement is used to create a YoutubeDL object, passing the ydl_opts dictionary as an argument. The download method of the YoutubeDL object is called with a list containing the URL of the content to be downloaded.

```
with youtube_dl.YoutubeDL(ydl_opts) as ydl:
    ydl.download([entry])
```

- During the download process, the my_hook function is called periodically to update the GUI label with the current status of the download (either "Downloading..." or "Downloaded!").

```
def my_hook(d):
    if d['status'] == 'finished':
        self.label.configure(self, text="Down-
loaded!")
        open_folder_with_explorer()
    if d['status'] == 'downloading':
        self.label.configure(self, text="Download-
ing...")
```

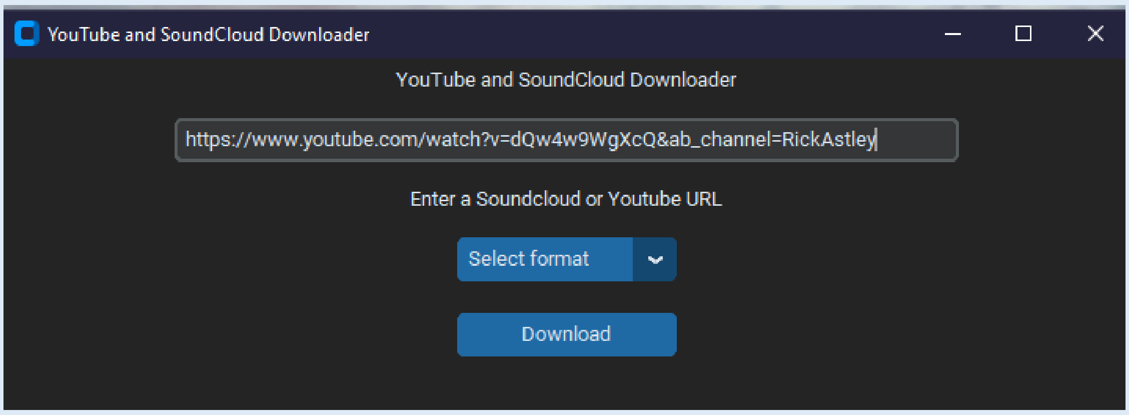
- Once the download is complete, the open_folder_with_explorer function is called to open the downloads folder where the content was saved.

```
def open_folder_with_explorer():
    currentPath = os.getcwd()
    folderName = 'downloads'
    path = os.path.join(currentPath, folderName)
    if os.name == 'nt':
        # For Windows
        subprocess.Popen(f'explorer "{path}"')
    elif os.name == 'posix':
        # For Mac
        subprocess.Popen(['open', path])
    else:
        raise OSError(f'Unsupported platform:
{os.name}')
```

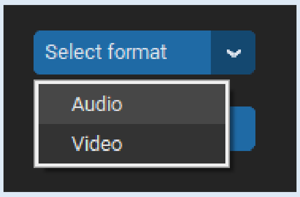
I used the following tutorials to learn how to implement the youtube_dl library into my script

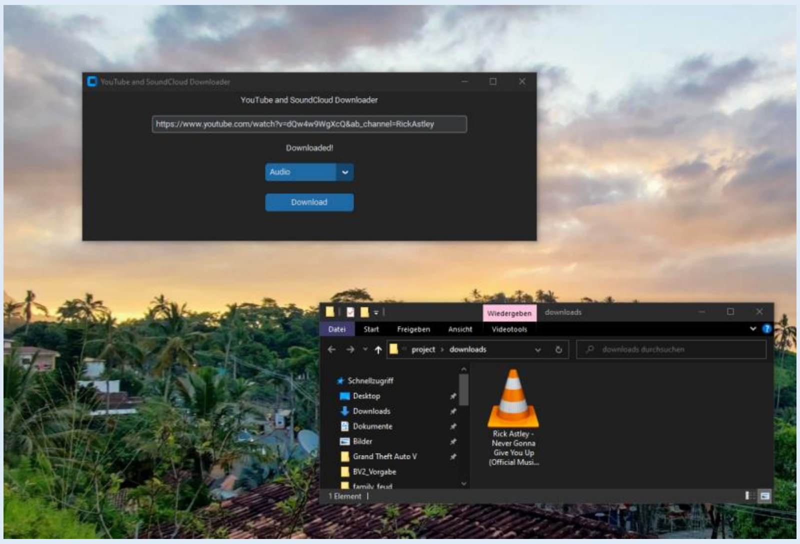
- https://www.assemblyai.com/blog/how-to-build-a-youtube-downloader-in-python/
- https://www.bogotobogo.com/VideoStreaming/YouTube/youtube-dl-embedding.php

## Using the App


YouTube and SoundCloud Downloader
YouTube and SoundCloud Downloader
Enter a Soundcloud or Youtube URL
Select format
Download

https://www.youtube.com/watch?v=dQw4w9WgXcQ&ab_channel=RickAstley





https://www.youtube.com/watch?v=dQw4w9WgXcQ&ab_channel=RickAstley

Ideas for future implementations that are possible with the youtube_dl library:

- Add support for downloading content from other popular websites like Vimeo, Dailymotion, and Facebook.
- Allow users to specify the output directory for downloaded content instead of using the default "downloads" directory in the app's working directory.
- Implement a feature to automatically convert downloaded video files to different formats (e.g. from MP4 to AVI or from MOV to WMV).
    - I tried to implement a Converter that converts the downloaded file to a desired format but I was unable to implement ffmpeg into my script.
- Implement a feature to search for and download entire playlists or channels from YouTube.
- Add support for downloading subtitles or closed captions for downloaded videos.

- Implement a feature to download only a specific portion of a video (e.g. download only the first 30 seconds of a music video).
- Use the CustomTkinter ProgressBar Widget to display the downloaded percentage.

## Reflection

I had a lot of fun building this little app, which fills a use case in my day-to-day life even though the app is not really user friendly yet, since it does not provide a lot of feedback, and doesn’t include a File Converter yet.