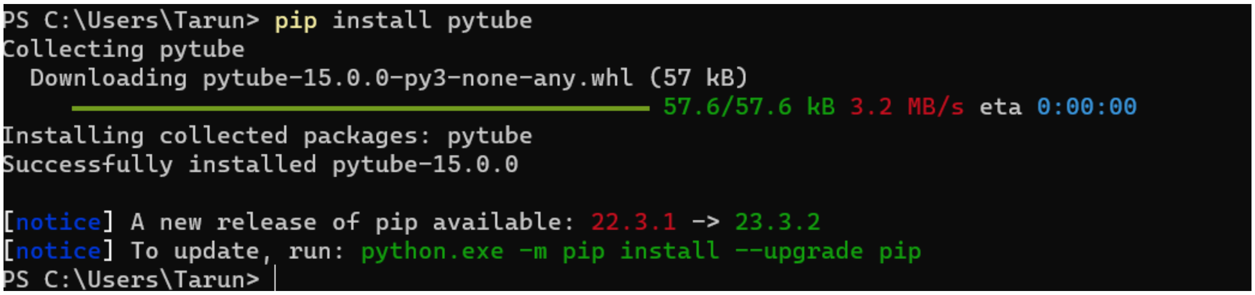
Automated YT Video Downloader

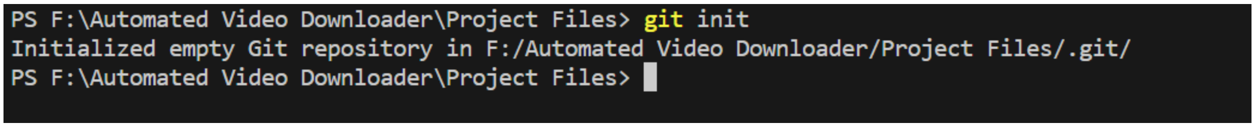
Description: This is a python automation project to download youtube videos and view it locally in offline mode.

Step 1:
Download pytube library



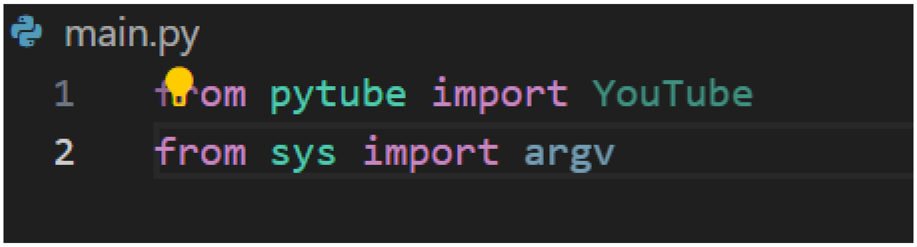
```
PS C:\Users\Tarun> pip install pytube
Collecting pytube
  Downloading pytube-15.0.0-py3-none-any.whl (57 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 57.6/57.6 kB 3.2 MB/s eta 0:00:00
Installing collected packages: pytube
Successfully installed pytube-15.0.0

[notice] A new release of pip available: 22.3.1 -> 23.3.2
[notice] To update, run: python.exe -m pip install --upgrade pip
PS C:\Users\Tarun>
```

Step 2:
Initialize a empty git repository



```
PS F:\Automated Video Downloader\Project Files> git init
Initialized empty Git repository in F:/Automated Video Downloader/Project Files/.git/
PS F:\Automated Video Downloader\Project Files>
```

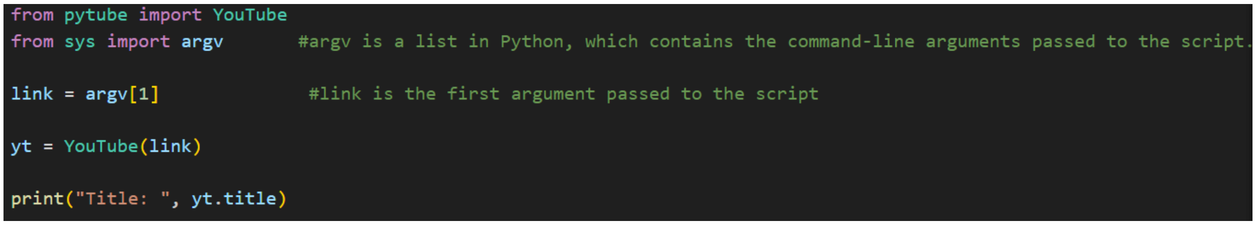
Step 3:
Import the libraries



main.py

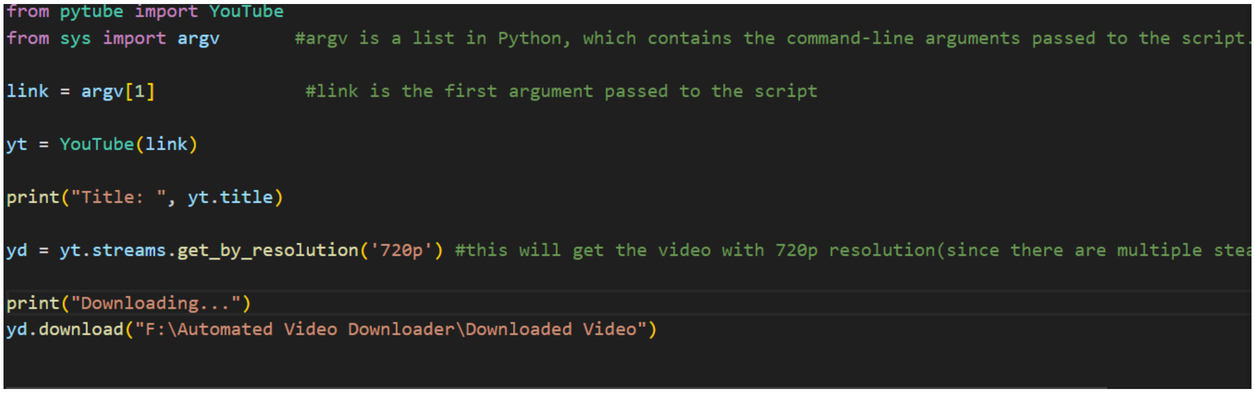
```
from pytube import YouTube
from sys import argv
```

Step 4:
Use the library commands to get the title of the video



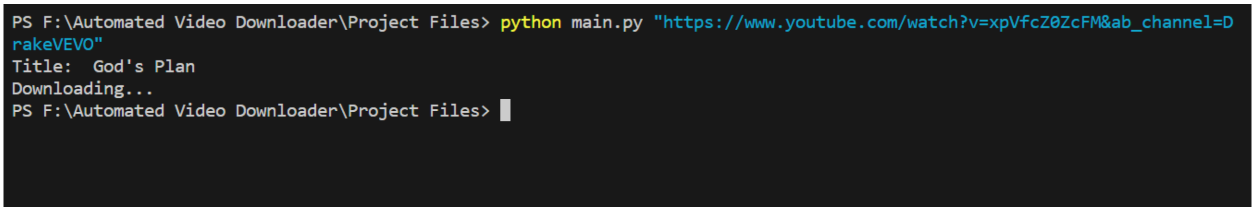
```
from pytube import YouTube
from sys import argv       #argv is a list in Python, which contains the command-line arguments passed to the script.

link = argv[1]              #link is the first argument passed to the script

yt = YouTube(link)

print("Title: ", yt.title)
```

Step 5:
Write the code for downloading the video and specify the directory where you want to store the downloaded video



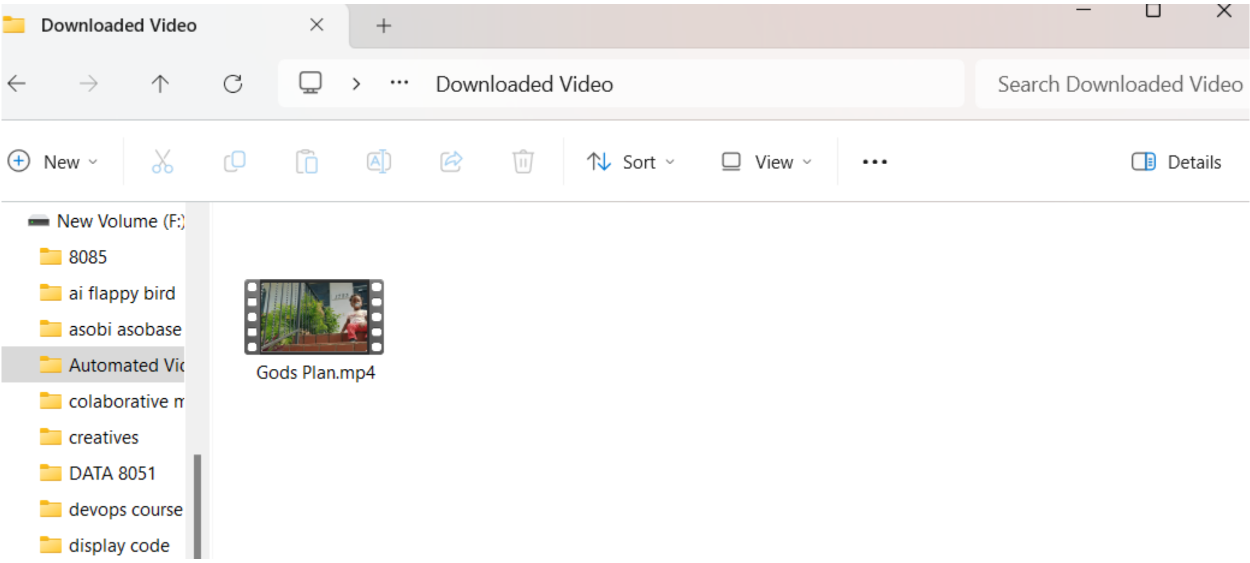
```
from pytube import YouTube
from sys import argv       #argv is a list in Python, which contains the command-line arguments passed to the script.

link = argv[1]              #link is the first argument passed to the script

yt = YouTube(link)

print("Title: ", yt.title)

yd = yt.streams.get_by_resolution('720p') #this will get the video with 720p resolution(since there are multiple stea

print("Downloading...")
yd.download("F:\Automated Video Downloader\Downloaded Video")
```

Step 6:
Run the code

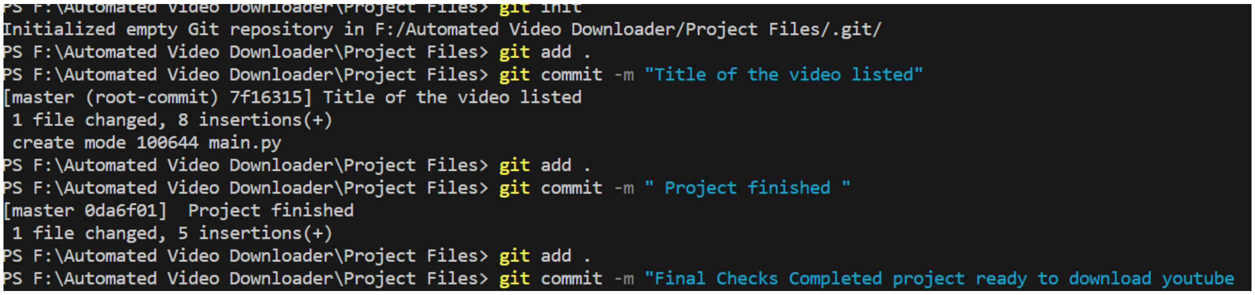


```
PS F:\Automated Video Downloader\Project Files> python main.py "https://www.youtube.com/watch?v=xpVfcZ0ZcFM&ab_channel=D
rakeVEVO"
Title:  God's Plan
Downloading...
PS F:\Automated Video Downloader\Project Files> 
```

Step 7:
Check the folder location for the downloaded video

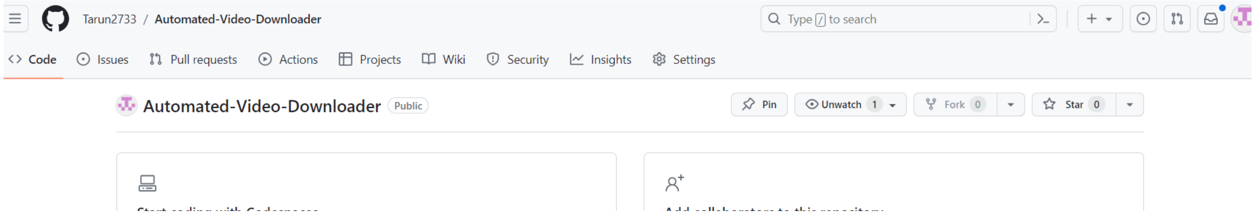


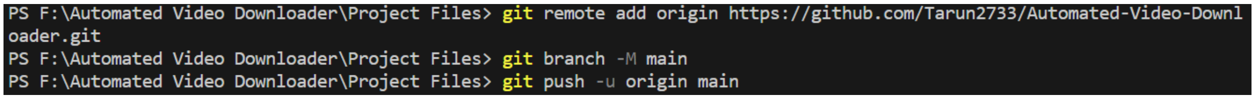
Step 8:
Keep committing to the VCS



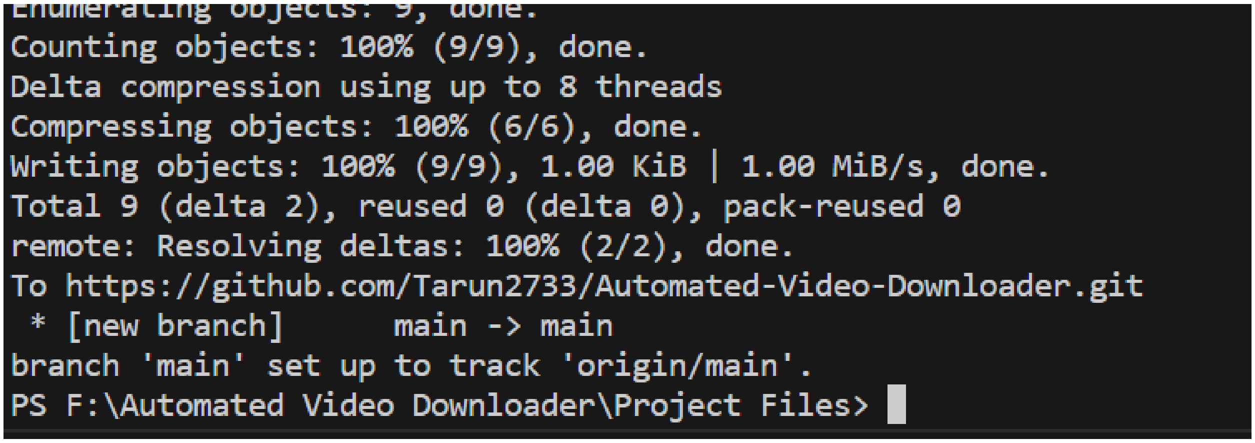
```
PS F:\Automated Video Downloader\Project Files> git init
Initialized empty Git repository in F:/Automated Video Downloader/Project Files/.git/
PS F:\Automated Video Downloader\Project Files> git add .
PS F:\Automated Video Downloader\Project Files> git commit -m "Title of the video listed"
[master (root-commit) 7f16315] Title of the video listed
 1 file changed, 8 insertions(+)
 create mode 100644 main.py
PS F:\Automated Video Downloader\Project Files> git add .
PS F:\Automated Video Downloader\Project Files> git commit -m " Project finished "
[master 0da6f01]  Project finished
 1 file changed, 5 insertions(+)
PS F:\Automated Video Downloader\Project Files> git add .
PS F:\Automated Video Downloader\Project Files> git commit -m "Final Checks Completed project ready to download youtube
```

Step 9:
Create a github repository and push the files





```
PS F:\Automated Video Downloader\Project Files> git remote add origin https://github.com/Tarun2733/Automated-Video-Downl
oader.git
PS F:\Automated Video Downloader\Project Files> git branch -M main
PS F:\Automated Video Downloader\Project Files> git push -u origin main
```



```
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 1.00 KiB | 1.00 MiB/s, done.
Total 9 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/Tarun2733/Automated-Video-Downloader.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
PS F:\Automated Video Downloader\Project Files>
```