A Mini Project ReportOn
“YouTube Video Downloader”

Submitted in partial fulfillment of the requirement of the University of Mumbai forthe Degree of **Bachelor of Engineering**
(Computer Engineering)

By
Krishna Gupta
Rishikesh Sonawane

Under the guidance of
Prof. Akshata Laddha

Department of Computer Engineering
Suman Educational Trust’s
Dilkap Research Institute of Engineering & Management Studies
Mamdapur, Post: Neral, Taluka: Karjat,

Dist: Raigad-410101

University of Mumbai
Academic Year 2021-2022

Dilkap Research Institute of Engineering & Management Studies

Department of Computer Engineering
Academic Year 2020-2021

CERTIFICATE

This is to certify that
Mr.Krishna Gupta
Mr.Rishikesh Sonawane
Sem.VIII, BE Computer,

has satisfactorily completed the requirements ofthe Mini Project entitled

"YouTube Video Downloader"

As prescribed by the University of Mumbai Under the guidance of

Prof. Akshata Laddha

Guide
(Prof. Akshata Laddha)

HOD
(Prof. Indira Joshi)

Internal Examiner
Prof.

External Examiner
Prof.

Acknowledgement

We would like to express our gratitude and appreciation to all those who gave us the possibility to complete this project and this report. I and my team thank our head of department Prof. Indira Joshi for giving us the accessory environment to acquire knowledge and skill. A special thanks to Prof. Akshata Laddha, whose help, stimulating suggestions and encouragement, helped us to coordinate our project especially in writing this report.

We would also like to acknowledge with much appreciation the crucial role of the staff of Computer Laboratory, who gave the permission to use all required machinery and the necessary material to complete the report special thanks goes to our friends, who gave suggestions about the source code.

Table of Contents

Abstract

"Cloud" is a collective term for a large number of developments and possibilities. It is not an invention, but more of a "practical innovation", combining several earlier inventions into something new and compelling. Much like the iPod is comprised of several existing concepts and technologies (the Walkman, MP3 compression and a portable hard disk), cloud computing merges several already available technologies: high bandwidth networks, virtualization, Web 2.0 interactivity, time sharing, and browser interfaces. Cloud Computing is a popular phrase that is shorthand for applications that were developed to be rich Internet applications that run on the Internet (or "Cloud"). Cloud computing enables tasks to be assigned to a combination of software and services over a network. This network of servers is the cloud. Cloud computing can help businesses transform their existing server infrastructures into dynamic environments, expanding and reducing server capacity depending on their requirements. A cloud computing platform dynamically provisions, configures, reconfigures, and deprovisions servers as needed. Servers in the cloud can be physical machines or virtual machines. Advanced clouds typically include other computing resources such as storage area networks (SANs), network equipment, firewall and other security devices.

Chapter 1: Introduction

Anyone who uses the Internet, probably watches YouTube, not only for entertainment but also for educational purposes. Due to low bandwidth in many places, people can't stream videos properly without any buffering.

To resolve, we could choose a better Internet Service Provider with higher bandwidth. But many of us wouldn't agree, so we have tons of ways to download youtube videos into our local storage.

Even if you have high bandwidth, there are many reasons that you would want a video from YouTube in your local storage. Some of the non-copyrighted free content is great for the creators to download and create new content from using the video or audio.

So, if you are a content creator, definitely you would want to know how to download YouTube videos. Moreover, there are many movies and songs which you would like to download and play later.

YouTube has an option to download, which is only available on the YouTube Android app. For browsers, there are several plugins and many applications for all major operating systems. However, officially the YouTube website doesn't provide you with an option to download.

So, we can use third-party and open-source projects to download YouTube videos. Here, we will discuss some of the best ways to download a YouTube video into your local directory. YouTube might block certain services and also blocks some videos to download. If you hit a roadblock like that, kindly try using other ways.

PROBLEM STATEMENT

The YouTube downloader project is a python project. The object of this project is to

download any type of video in a fast and easy way from YouTube in your device. In this python project, user has to copy the YouTube video URL that they want to download and simply paste that URL in the ‘paste link here’ section and click on the download button, it will start downloading the video. When video downloading finishes, it shows a message ‘downloaded’ popup on the window below the download button.

Chapter 2: Literature Survey

YouTube is becoming a major resource for sharing and consuming video content. It is gaining immense popularity and support from viewer community due to its comprehensive repository of videos. Also, it supports diversity by having different facets such as modals, languages, domains and cultures. For a YouTube (Lanyu Shang, 2019) content developer or a YouTuber with various notable channels, (Lanyu Shang, 2019) this is a profession with a lot of monetary potential. The younger generations are recently shifting to YouTube (Lanyu Shang, 2019) and other OTT platforms, away from the traditional television. A YouTube (Lanyu Shang, 2019) video often consists of a title, thumbnail, video content along with other non-video features. Despite it being unethical, content developers deliberately manipulate the heading and the thumbnail so as to attract more audience and baiting them into viewing their content. There are quite a few instances when the content of the video mismatches with the heading of the video or the thumbnail of the video. This is known as a Clickbait (Lanyu Shang, 2019)Video. Our aim is to classify a video as to whether it is a Clickbait (Lanyu Shang, 2019) or not. This is critically important as a majority of people spend their time on YouTube (Lanyu Shang, 2019) and not getting what they search for is a waste of their precious time. We use sentiment analysis on viewer comments to identify a video as click bait or not

DATASET

We are only working with YouTube (Lanyu Shang, 2019) data that consists of viewer comments. The data is collected with the help of YouTube (Lanyu Shang, 2019) API v3. We created a Google Developer account and generated a key to extract all the details of a video in the form of a JSON file. This dataset contains all the details of the trending YouTube videos along with its likes, dislikes, comments, tags and views for each video for a particular year, which comprises a top-level comment and replies, if any exist, to that comment

Chapter 3: Resources

1.Asgiref

ASGI is a standard for Python asynchronous web apps and servers to communicate with each other, and positioned as an asynchronous successor to WSGI. You can read more at https://asgi.readthedocs.io/en/latest/

This package includes ASGI base libraries, such as:

- Sync-to-async and async-to-sync function wrappers, asgiref.sync
- Server base classes, asgiref.server
- A WSGI-to-ASGI adapter, in asgiref.wsgi

2.Certifi

Certifi provides Mozilla's carefully curated collection of Root Certificates for validating the trustworthiness of SSL certificates while verifying the identity of TLS hosts. It has been extracted from the Requests project.

Installation

- certifi is available on PyPI. Simply install it with pip:
  $ pip install certifi
- Usage
- To reference the installed certificate authority (CA) bundle, you can use the built-in function:
  $ python -m certifi
  /usr/local/lib/python3.7/site-packages/certifi/cacert.pem

3.Dj-Database-Url

- This simple Django utility allows you to utilize the 12factor inspired

DATABASE_URL environment variable to configure your Django application.

- The dj_database_url.config method returns a Django database connection dictionary, populated with all the data specified in your URL. There is also a *conn_max_age* argument to easily enable Django's connection pool.
- If you'd rather not use an environment variable, you can pass a URL in directly instead to dj_database_url.parse.
- **Supported Databases**

  Support currently exists for PostgreSQL, PostGIS, MySQL, MySQL (GIS), Oracle, Oracle (GIS), and SQLite.
- **Installation**

  Installation is simple:

  $ pip install dj-database-url

4.Django

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Thanks for checking it out.

All documentation is in the "docs" directory and online at https://docs.djangoproject.com/en/stable/.

5.Gunicorn

Gunicorn 'Green Unicorn' is a Python WSGI HTTP Server for UNIX. It's a pre-fork worker model ported from Ruby's Unicorn project. The Gunicorn server is broadly compatible with various web frameworks, simply implemented, light on server resource usage, and fairly speedy.

- **Installation**

  Gunicorn requires **Python 3.x >= 3.5**.
- Install from PyPI:

  $ pip install gunicorn

7.Pafy

Features

- Retreive metadata such as viewcount, duration, rating, author, thumbnail, keywords
- Download video or audio at requested resolution / bitrate / format / filesize
- Command line tool (ytdl) for downloading directly from the command line
- Retrieve the URL to stream the video in a player such as vlc or mplayer
- Works with age-restricted videos and non-embeddable videos
- Small, standalone, single importable module file (pafy.py)
- Select highest quality stream for download or streaming
- Download video only (no audio) in m4v or webm format
- Download audio only (no video) in ogg or m4a format
- Retreive playlists and playlist metadata
- Works with Python 2.6+ and 3.3+
- Optionally depends on youtube-dl (recommended; more stable)

8. Pytz

- pytz brings the Olson tz database into Python. This library allows accurate and cross platform timezone calculations using Python 2.4 or higher. It also solves the issue of ambiguous times at the end of daylight saving time, which you can read more about in the Python Library Reference (datetime.tzinfo).
- Almost all of the Olson timezones are supported.
- This library differs from the documented Python API for tzinfo implementations; if you want to create local wallclock times you need to use the localize() method documented in this document. In addition, if you perform date arithmetic on local times that cross DST boundaries, the result may be in an incorrect timezone (ie. subtract 1 minute from 2002-10-27 1:00 EST and you get 2002-10-27 0:59 EST instead of the correct 2002-10-27 1:59 EDT). A normalize() method is provided to correct this. Unfortunately these issues cannot be resolved without modifying the Python datetime implementation (see PEP-431).
- **Installation**
    - This package can either be installed using pip or from a tarball using the standard Python distutils.
    - If you are installing using pip, you don't need to download anything as the latest version will be downloaded for you from PyPI:
    - pip install pytz

9.Sqlparse

- sqlparse is a non-validating SQL parser for Python. It provides support for parsing, splitting and formatting SQL statements.
- The module is compatible with Python 3.5+ and released under the terms of the New BSD license.
- Visit the project page at https://github.com/andialbrecht/sqlparse for further

information about this project.

$ pip install sqlparse

10.Whitenoise

Radically simplified static file serving for Python web apps

- With a couple of lines of config WhiteNoise allows your web app to serve its own static files, making it a self-contained unit that can be deployed anywhere without relying on nginx, Amazon S3 or any other external service. (Especially useful on Heroku, OpenShift and other PaaS providers.)
- It's designed to work nicely with a CDN for high-traffic sites so you don't have to sacrifice performance to benefit from simplicity.
- WhiteNoise works with any WSGI-compatible app but has some special auto-configuration features for Django.
- WhiteNoise takes care of best-practices for you, for instance:
    - Serving compressed content (gzip and Brotli formats, handling Accept-Encoding and Vary headers correctly)
    - Setting far-future cache headers on content which won't change

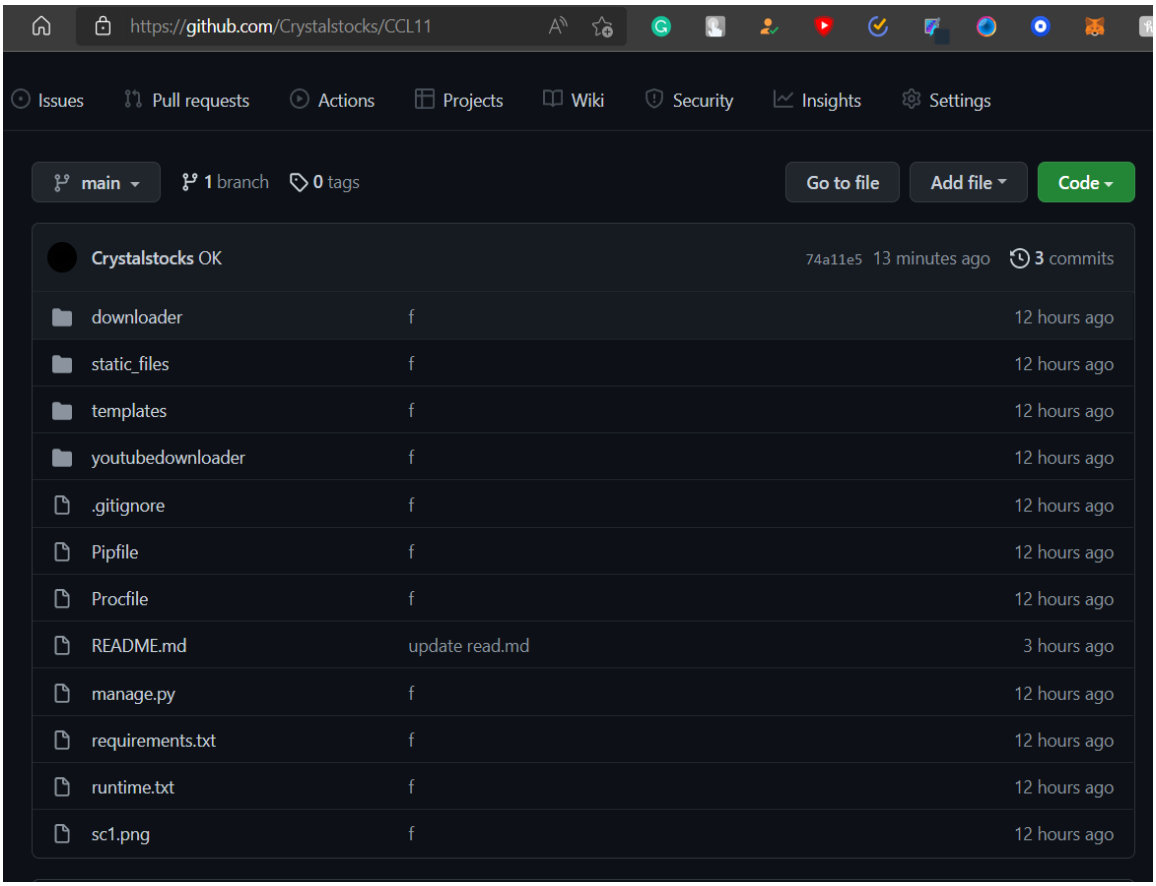
11.Youtube-dl

- Command-line program to download videos from YouTube.com and other video sites

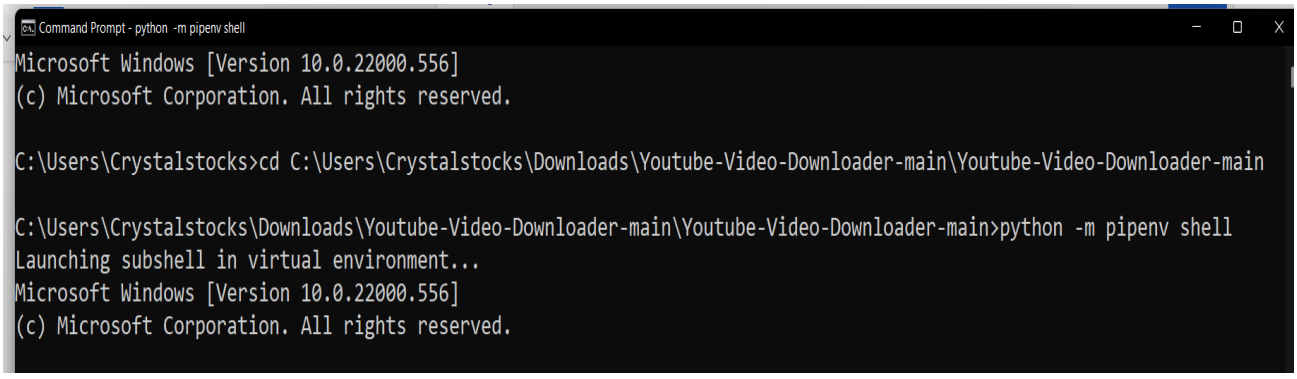
Chapter 4: Project Implementation

Setup/Installation

1. **Download** and **extract** the source code zip file from Github.
2. Source Code : Crystalstocks/CCL11 (github.com)

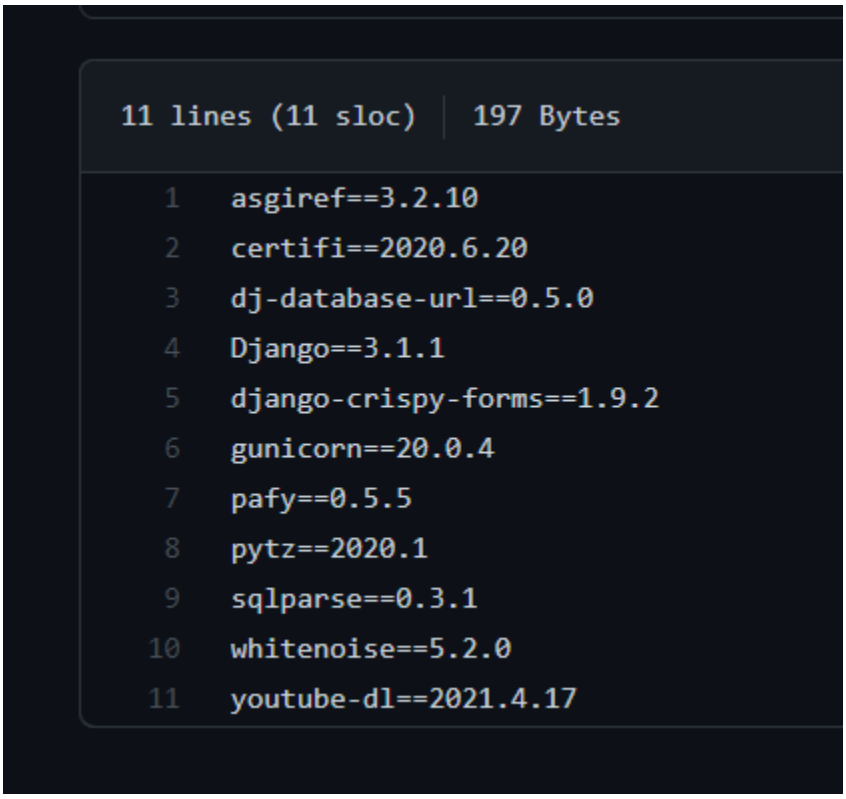


3. **Open** your **Terminal/Command Prompt** window. *(Make sure to add "python" and "pip" in your environment variables)*



4. **Change** the **working directory** to the extracted source code folder.
5. **Run** the following **commands**:
    - pip install Django
    - pip install -r requirements.txt

```
(Youtube-Video-Downloader-main-mabSZswC) C:\Users\Crystalstocks\Downloads\Youtube-Video-Downloader-main\Youtube-Video-Download
er-main>pip install -r requirements.txt
Requirement already satisfied: asgiref==3.2.10 in c:\users\crystalstocks\.virtualenvs\youtube-video-downloader-main-mabszswc\l
ib\site-packages (from -r requirements.txt (line 1)) (3.2.10)
Requirement already satisfied: certifi==2020.6.20 in c:\users\crystalstocks\.virtualenvs\youtube-video-downloader-main-mabszsw
c\lib\site-packages (from -r requirements.txt (line 2)) (2020.6.20)
Requirement already satisfied: dj-database-url==0.5.0 in c:\users\crystalstocks\.virtualenvs\youtube-video-downloader-main-mab
szswc\lib\site-packages (from -r requirements.txt (line 3)) (0.5.0)
Requirement already satisfied: Django==3.1.1 in c:\users\crystalstocks\.virtualenvs\youtube-video-downloader-main-mabszswc\lib
\site-packages (from -r requirements.txt (line 4)) (3.1.1)
Requirement already satisfied: django-crispy-forms==1.9.2 in c:\users\crystalstocks\.virtualenvs\youtube-video-downloader-main
-mabszswc\lib\site-packages (from -r requirements.txt (line 5)) (1.9.2)
Requirement already satisfied: gunicorn==20.0.4 in c:\users\crystalstocks\.virtualenvs\youtube-video-downloader-main-mabszswc\
lib\site-packages (from -r requirements.txt (line 6)) (20.0.4)
Requirement already satisfied: pafy==0.5.5 in c:\users\crystalstocks\.virtualenvs\youtube-video-downloader-main-mabszswc\lib\s
ite-packages (from -r requirements.txt (line 7)) (0.5.5)
Requirement already satisfied: pytz==2020.1 in c:\users\crystalstocks\.virtualenvs\youtube-video-downloader-main-mabszswc\lib\
site-packages (from -r requirements.txt (line 8)) (2020.1)
Requirement already satisfied: sqlparse==0.3.1 in c:\users\crystalstocks\.virtualenvs\youtube-video-downloader-main-mabszswc\l
ib\site-packages (from -r requirements.txt (line 9)) (0.3.1)
Requirement already satisfied: whitenoise==5.2.0 in c:\users\crystalstocks\.virtualenvs\youtube-video-downloader-main-mabszswc
\lib\site-packages (from -r requirements.txt (line 10)) (5.2.0)
Requirement already satisfied: youtube-dl==2021.4.17 in c:\users\crystalstocks\.virtualenvs\youtube-video-downloader-main-mabs
zswc\lib\site-packages (from -r requirements.txt (line 11)) (2021.4.17)
Requirement already satisfied: setuptools>=3.0 in c:\users\crystalstocks\.virtualenvs\youtube-video-downloader-main-mabszswc\l
ib\site-packages (from gunicorn==20.0.4->-r requirements.txt (line 6)) (60.9.3)
```



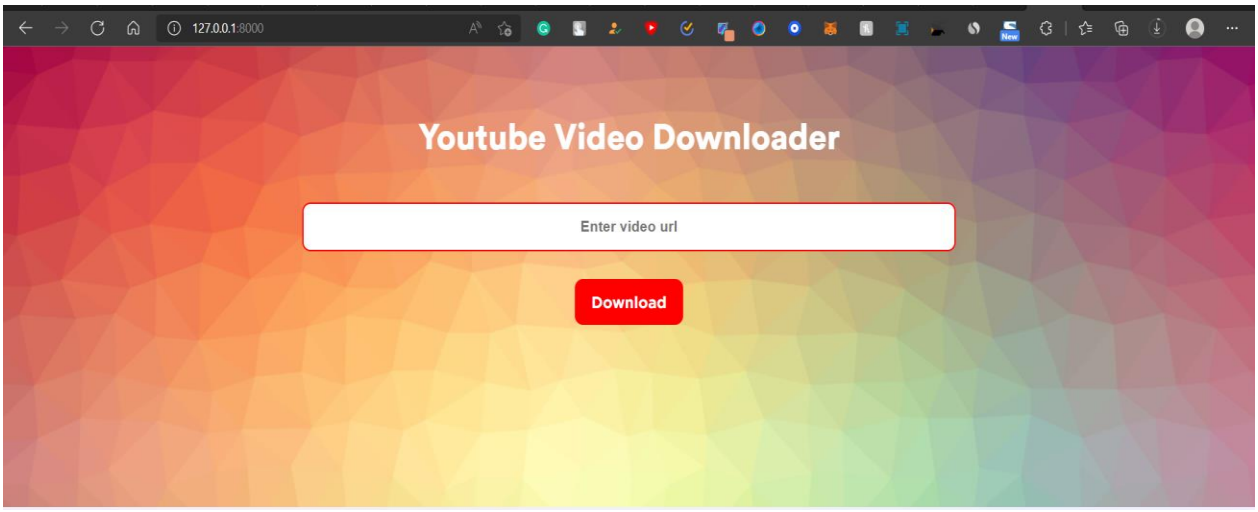
- python manage.py migrate
- python manage.py runserver

```
(Youtube-Video-Downloader-main-mabSZswC) C:\Users\Crystalstocks\Downloads\Youtube-Video-Downloader-main\Youtube-Video-Download
er-main>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
March 21, 2022 - 14:13:16
Django version 3.1.1, using settings 'youtubedownloader.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

6. **Open** a **web browser** http://127.0.0.1:8000/

- http://127.0.0.1:8000/ Localhost View:



7. Push project source code to github .

- If you don't have Git installed, see this article on how to set it up.
- Open up a Windows command prompt.
- Change into the directory where your source code is located in the command prompt.
- First, create a new repository in this directory git init. This will say *"Initialized empty git repository in ....git"* (... is the path).
- Now you need to tell Git about your files by adding them to your repository. Do this with git add filename. If you want to add all your files, you can do git add .
- Now that you have added your files and made your changes, you need to *commit* your changes so Git can track them. Type git commit -m "adding files". -m lets you add the *commit message* in line.
- So far, the above steps is what you would do even if you were not using GitHub. They are the normal steps to start a Git repository. Remember that Git is distributed (decentralized), meaning you don't need to have a "central server" (or even a network connection), to use Git.
- Now you want to push the changes to your Git repository hosted with GitHub. You do this by telling Git to *add a remote location*, and you do that with this command:

- git remote add origin https://github.com/yourusername/your-repo-name.git
- *Note: your-repo-name should be created in GitHub before you do a git remote add origin ...
- Once you have done that, Git now knows about your remote repository. You can then tell it to push (which is "upload") your committed files:
- git push -u origin master

8. Host Django Project on Heroku

- Requirements.txt file: Create requirements.txt file in the same directory as your manage.py. Run the following command in the console with the virtual environment activated:

  (myvenv) $ pip install dj-database-url gunicorn whitenoise

  (myvenv) $ pip freeze > requirements.txt

- Check your requirements.txt. It will be updated with the packages currently installed in your project.
- Procfile: Create a file named Procfile in the same directory as manage.py. you will see the Heroku logo as Procfile's icon. Add the following line to it:

  web: gunicorn <project_name>.wsgi --log-file -

- Here project name will be the name of the folder in which your settings.py is present. Procfile explicitly declares what command should be executed to start your app.
- Runtime.txt file: Create runtime.txt file in the same directory as your manage.py. Add the python version you want to use for your web app:

  python-3.7.1

- Settings.py: Modify your settings.py as per the instructions below:
  - Set debug as False.

    DEBUG = False

  - Modify allowed hosts.

ALLOWED_HOSTS = ['127.0.0.1', 'anhsirk4.herokuapp.com']

- To disable Django's static file handling and allow WhiteNoise to take over add 'nostatic' to the top of your 'INSTALLED_APPS' list.

```
INSTALLED_APPS = [
   'whitenoise.runserver_nostatic',
   'django.contrib.staticfiles',
   # ...
]
```

- Add WhiteNoise to the MIDDLEWARE list. The WhiteNoise middleware should be placed directly after the Django SecurityMiddleware (if you are using it) and before all other middleware:

```
MIDDLEWARE = [
 'django.middleware.security.SecurityMiddleware',
 'whitenoise.middleware.WhiteNoiseMiddleware',
 # ...
]
```

- Update your database settings.

```
import dj_database_url
DATABASES = {
   'default': {
      'ENGINE': 'django.db.backends.postgresql_psycopg2',
      'NAME': '<database_name>',
      'USER': '<user_name>',
      'PASSWORD': '<password>',
      'HOST': 'localhost',
      'PORT': '',
```

}

}

db_from_env = dj_database_url.config(conn_max_age=500)

DATABASES['default'].update(db_from_env)

- To serve files directly from their original locations (usually in STATICFILES_DIRS or app static subdirectories) without needing to be collected into STATIC_ROOT by the collectstatic command; set WHITENOISE_USE_FINDERS to True.

  WHITENOISE_USE_FINDERS = True

- WhiteNoise comes with a storage backend that automatically takes care of compressing your files and creating unique names for each version so they can safely be cached forever. To use it, just add this to your settings.py:

  STATICFILES_STORAGE =

  'whitenoise.storage.CompressedManifestStaticFilesStorage'

- Final modified Contents of settings.py:
- **import dj_database_url**

  ...

  DEBUG = False

  ALLOWED_HOSTS = ['127.0.0.1', '.herokuapp.com']

  INSTALLED_APPS = [

  'whitenoise.runserver_nostatic',

  #...

  ]

  MIDDLEWARE = [

  'django.middleware.security.SecurityMiddleware',

  'whitenoise.middleware.WhiteNoiseMiddleware',

```
        #...
    ]
    ...
    DATABASES = {
       'default': {
          'ENGINE': 'django.db.backends.postgresql_psycopg2',
          'NAME': '<database_name>',
          'USER': '<username>',
          'PASSWORD': '<password>',
          'HOST': 'localhost',
          'PORT': '',
       }
    }
    WHITENOISE_USE_FINDERS = True
    ...
    db_from_env = dj_database_url.config(conn_max_age=500)
    DATABASES['default'].update(db_from_env)
    STATICFILES_STORAGE =
    'whitenoise.storage.CompressedManifestStaticFilesStorage'
```

- Heroku account
    - Install your Heroku toolbelt which you can find here: https://toolbelt.heroku.com/
    - Authenticate your Heroku account either running the below command in cmd or gitbash

      $heroku login
    - Here the directory of the project(resume) to be deployed is active
    - Sometimes the cmd or git bash may freeze at certain commands. Just

use CTRL+C to come out of it.

- Commit any changes on git before deploying.

  $ git status

  $ git add -A.

  $ git commit -m "additional files and changes for Heroku"

- Pick your application name which will be displayed on the domain name– [your app's name]. herokuapp.com and create the application using below command:

  $ heroku create <your_app's_name>

- **Debugging:** If collectstatic failed during a build, a traceback was provided that will be helpful in diagnosing the problem. If you need additional information about the environment collectstatic was run in, use the DEBUG_COLLECTSTATIC configuration.

  $ heroku config:set DEBUG_COLLECTSTATIC=1

- **Disabling Collectstatic**: Sometimes, you may not want Heroku to run collectstatic on your behalf. You can disable the collectstatic build step with the DISABLE_COLLECTSTATIC configuration:

  $heroku config:set DISABLE_COLLECTSTATIC=1

- Finally, do a simple git push to deploy our application:

  $ git push heroku master

- When we deployed to Heroku, we created a new database and it's empty. We need to run the migrate and createsuperuser commands.

  $ heroku run python manage.py migrate

- 9. To open your site run:

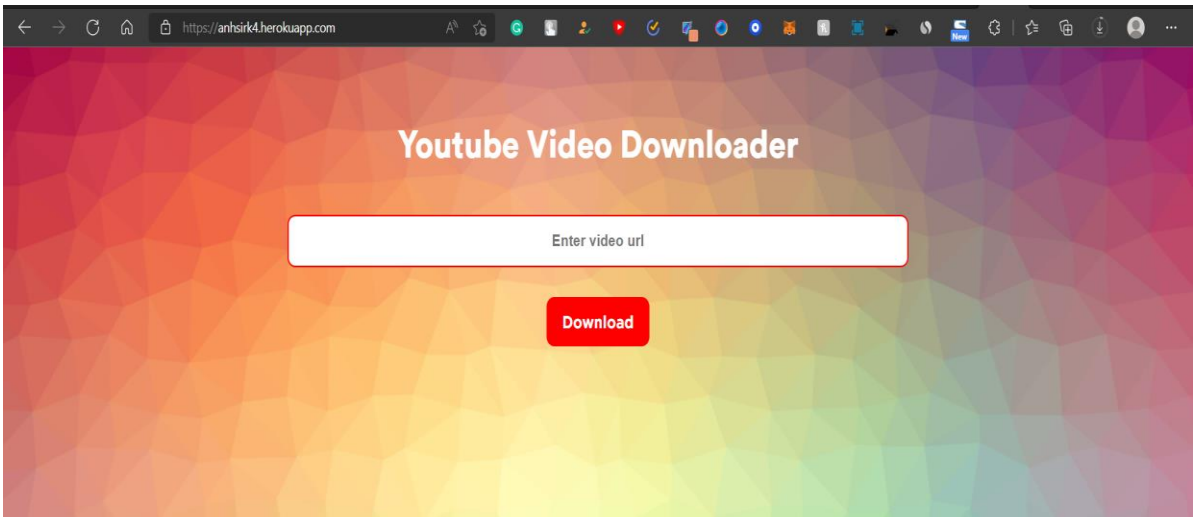
  $ heroku open

- Resolving Errors

- In case you see application error on your website run:

  $heroku logs --tail

9. Open a **web browser** [(anhsirk4.herokuapp.com)](anhsirk4.herokuapp.com)



Chapter 5: Conclusion

YouTube, not only for entertainment but also for educational purposes. Due to low bandwidth in many places, people can't stream videos properly without any buffering. To resolve, we could choose a better Internet Service Provider with higher bandwidth. But many of us wouldn't agree, so we have tons of ways to download youtube videos into our local storage. Even if you have high bandwidth, there are many reasons that you would want a video from YouTube in your local storage. Some of the non-copyrighted free content is great for the creators to download and create new content from using the video or audio. So, if you are a content creator, definitely you would want to know how to download YouTube videos. Moreover, there are many movies and songs which you would like to download and play later. YouTube has an option to download, which is only available on the YouTube Android app. For browsers, there are several plugins and many applications for all major operating systems. However, officially the YouTube website doesn't provide you with an option to download The YouTube downloader project is a python project. The object of this project is to download any type of video in a fast and easy way from YouTube in your device. In this python project, user has to copy the YouTube video URL that they want to download and simply paste that URL in the ‘paste link here’ section and click on the download button, it will start downloading the video. When video downloading finishes, it shows a message ‘downloaded’ popup on the window below the download button.

Chapter 6: References

1. Gunderloy, Jorden BPB Publications (2000) - “Mastering SQL Server”

2. Luke Welling and Laura Thomson (5th Edition) - “PHP and MySQL Web Development”

3. Roger S.Pressmen, T. Mc. GH. – Software Engineering (Theoretical Approach)

4. Thereon Willis wrox publications (2000) - “Beginning SQLServer”

5. Anjali.V et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 7 (2) , 2016, 760-763

6. www.youtube.com

7. www.wikipedia.com

8.www.google.com