Youtube Content Downloader Using Django

Overview

In the past, a popular method for downloading YouTube videos involved simply adding "ss" to the URL of the video you wanted to download. This method is no longer reliable, and there is a need for a more robust solution.

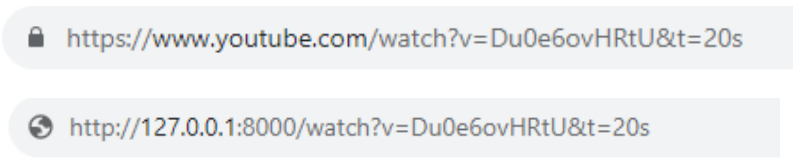
This project aims to provide a user-friendly and efficient way to download YouTube content using the Django framework. It offers two primary features:

1. Downloading YouTube Content by URL: Users can paste the URL of a YouTube video into a search bar, and the application will automatically identify the available download options.
2. Downloading YouTube Content by URL Prefix: By adding a specific prefix to the YouTube URL, users can directly access the download page without having to enter the entire URL. This feature is particularly useful for frequently downloaded videos.

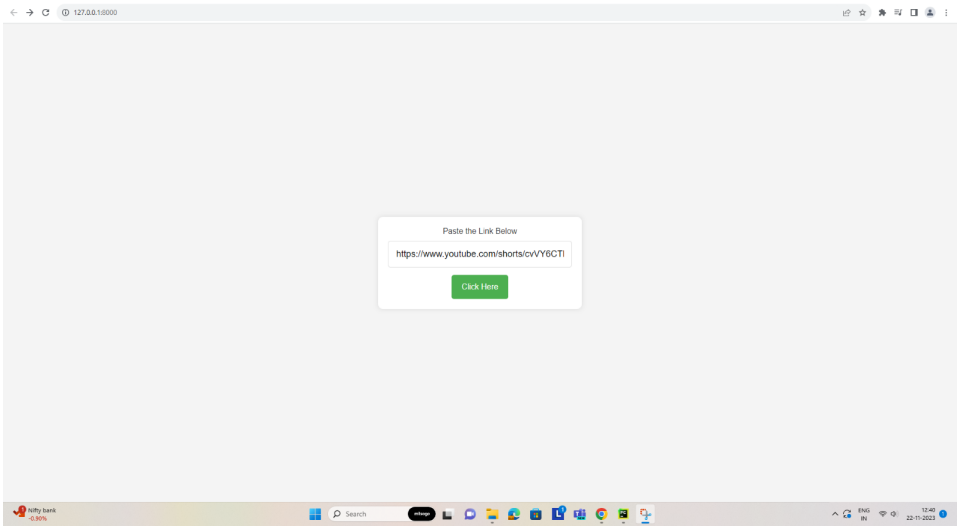
Description

1. Downloading Media by URL Prefix

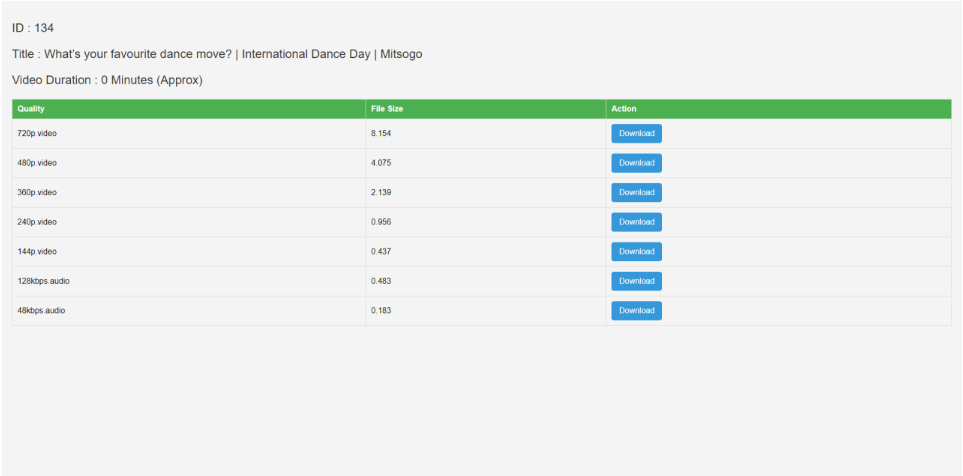
The main objective of this project is to implement the URL prefix feature. To achieve this, a dedicated domain containing the keyword "youtube" would be ideal. However, as this is not readily available, the project utilizes a URL replacement technique, substituting "www.youtube.com" with "127.0.0.1:8000" to redirect users to the project's download page.



2. Downloading Media by YouTube URL Input



Users can copy and paste the YouTube video URL into a search box



The application will process the URL and display the available download options.

For instance, using the example of Mitsogo's Favorite dance move short video, the application will present the user with various quality options (720p, 1080p, audio) for downloading. Upon selecting the desired quality, the application will initiate the download process.

Technical Description

Libraries :

Pytube is the cornerstone of this project, providing the functionality to download YouTube content in different streams (qualities) and retrieve video information.

HTML Templates :

Two HTML templates are employed:

1. Home Page: This page serves as the starting point, prompting the user to enter a YouTube URL.
2. Select Stream: After processing the user-provided URL, this page displays the available download options based on the video's available streams.

Database :

The default Django database (SQLite) is utilized to store information about downloaded videos. The database schema includes attributes such as IP address, URL, title, type, quality, and stream. Upon receiving a URL from the user, the application automatically creates a new object for the user, including the following:

- IPaddress: Stores the client's IP address (to be implemented in the future).
- URL: Stores the YouTube URL provided by the user.
- Stream: Stores the selected quality for downloading.
- Type: Determines whether the downloaded content is an audio or video file.
- Quality: Indicates the resolution of the downloaded video.
- Title: Stores the title of the downloaded video.

Workflow :

1. The user accesses the project's home page (Home.html).
2. The user enters a YouTube URL and submits it.
3. The Home function in views.py retrieves video information using Pytube and displays it on the Select Stream page (Select Stream.html).

4. The Select Stream page presents the user with available download options and video information.
5. The user selects the desired quality and initiates the download.
6. The Download function in views.py stores the downloaded file (audio or video) on the local server .
7. The file is then transferred to the user's browser using an HTTP file response, enabling the download to the client's system.