Springboard Data Scraping to DBMS- Report

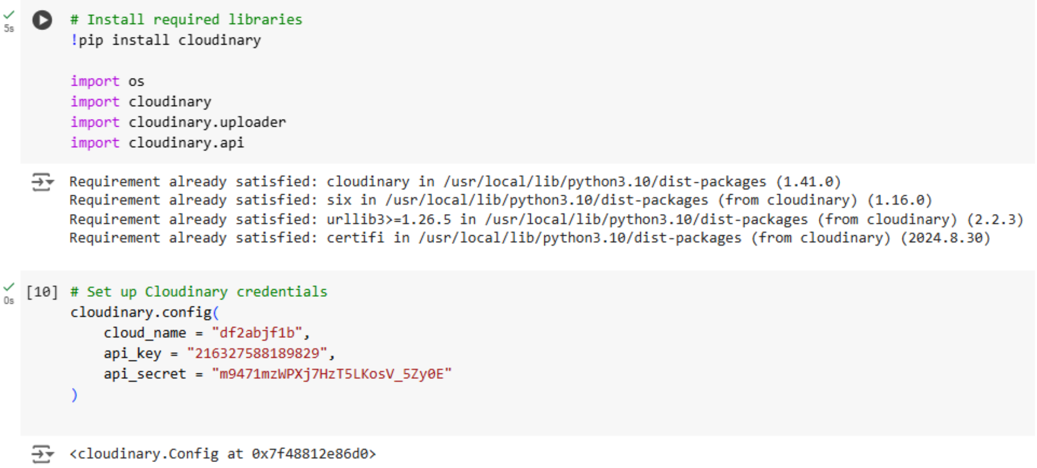
Overview:

The objective of this project is to create a Python script that can upload images to Cloudinary, a cloud-based image and video management service, and then download the uploaded images from Cloudinary.

Step-by-Step Process:

1. **Set up the Environment:**

   - Installed the required Python libraries: cloudinary, os, cloudinary.uploader, and cloudinary.api.
   - Configured the Cloudinary credentials (cloud name, API key, and API secret) in the script.



```
# Install required libraries
!pip install cloudinary

import os
import cloudinary
import cloudinary.uploader
import cloudinary.api
```

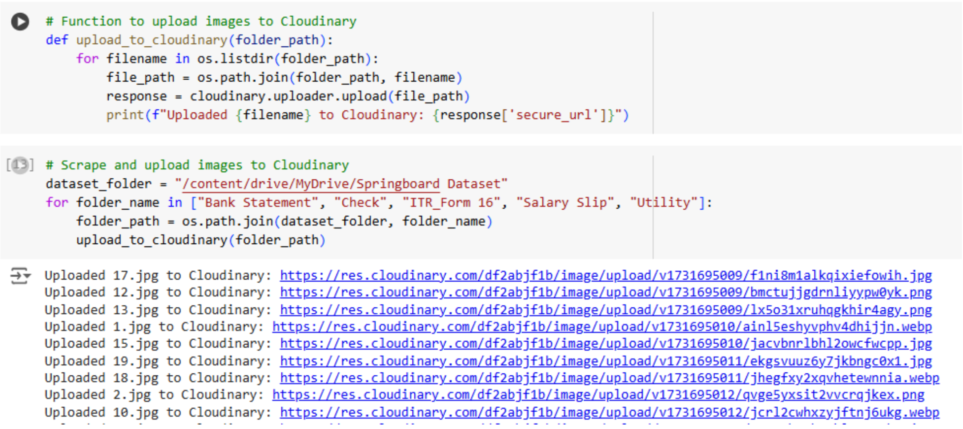
```
Requirement already satisfied: cloudinary in /usr/local/lib/python3.10/dist-packages (1.41.0)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from cloudinary) (1.16.0)
Requirement already satisfied: urllib3>=1.26.5 in /usr/local/lib/python3.10/dist-packages (from cloudinary) (2.2.3)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from cloudinary) (2024.8.30)
```

```
[10] # Set up Cloudinary credentials
cloudinary.config(
    cloud_name = "df2abjf1b",
    api_key = "216327588189829",
    api_secret = "m9471mzWPXj7HzT5LKosV_5Zy0E"
)
```

```
<cloudinary.Config at 0x7f48812e86d0>
```

2. **Implement the Upload Function:**

   - Defined a function upload_to_cloudinary() that takes a folder path as input.
   - Inside the function, used a for loop to iterate through the files in the folder.
   - For each file, constructed the full file path using os.path.join().
   - Uploaded the file to Cloudinary using cloudinary.uploader.upload() and printed the secure URL of the uploaded image.



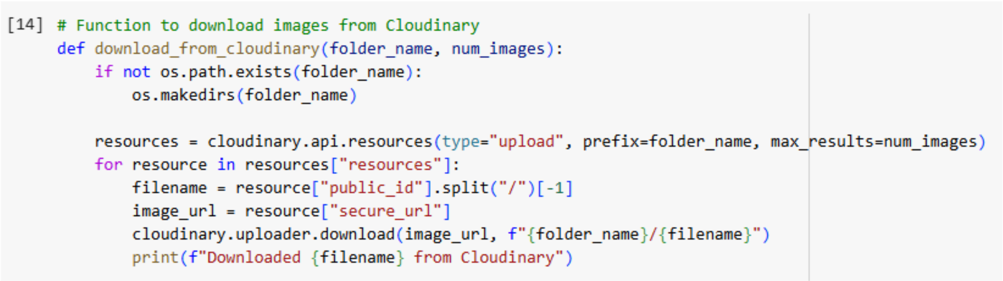
```
# Function to upload images to Cloudinary
def upload_to_cloudinary(folder_path):
    for filename in os.listdir(folder_path):
        file_path = os.path.join(folder_path, filename)
        response = cloudinary.uploader.upload(file_path)
        print(f"Uploaded {filename} to Cloudinary: {response['secure_url']}")
```

```
[13] # Scrape and upload images to Cloudinary
dataset_folder = "/content/drive/MyDrive/Springboard Dataset"
for folder_name in ["Bank Statement", "Check", "ITR_Form 16", "Salary Slip", "Utility"]:
    folder_path = os.path.join(dataset_folder, folder_name)
    upload_to_cloudinary(folder_path)
```

```
Uploaded 17.jpg to Cloudinary: https://res.cloudinary.com/df2abjf1b/image/upload/v1731695009/f1ni8m1alkqixiefowih.jpg
Uploaded 12.jpg to Cloudinary: https://res.cloudinary.com/df2abjf1b/image/upload/v1731695009/bmctujjgdrnliyypw0yk.png
Uploaded 13.jpg to Cloudinary: https://res.cloudinary.com/df2abjf1b/image/upload/v1731695009/lx5o31xruhqgkhir4agy.png
Uploaded 1.jpg to Cloudinary: https://res.cloudinary.com/df2abjf1b/image/upload/v1731695010/ainl5eshyvphv4dhijjn.webp
Uploaded 15.jpg to Cloudinary: https://res.cloudinary.com/df2abjf1b/image/upload/v1731695010/jacvbnrlbhl2owcfwcpp.jpg
Uploaded 19.jpg to Cloudinary: https://res.cloudinary.com/df2abjf1b/image/upload/v1731695011/ekgsvuuz6y7jkbngc0x1.jpg
Uploaded 18.jpg to Cloudinary: https://res.cloudinary.com/df2abjf1b/image/upload/v1731695011/jhegfxy2xqvhetewnnia.webp
Uploaded 2.jpg to Cloudinary: https://res.cloudinary.com/df2abjf1b/image/upload/v1731695012/qvge5yxsit2vvcrqjkex.png
Uploaded 10.jpg to Cloudinary: https://res.cloudinary.com/df2abjf1b/image/upload/v1731695012/jcrl2cwhxzyjftnj6ukg.webp
```

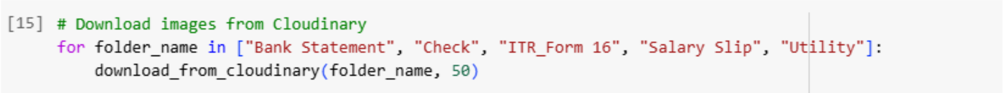
3. Implement the Download Function:

- Defined a function download_from_cloudinary() that takes a folder name and the number of images to download as input.
- Created the folder if it doesn't exist using os.makedirs().
- Used cloudinary.api.resources() to retrieve the resources (images) from Cloudinary, with a limit of the specified number of images.
- Iterated through the resources and downloaded each image using cloudinary.uploader.download().



```
[14] # Function to download images from Cloudinary
     def download_from_cloudinary(folder_name, num_images):
         if not os.path.exists(folder_name):
             os.makedirs(folder_name)

         resources = cloudinary.api.resources(type="upload", prefix=folder_name, max_results=num_images)
         for resource in resources["resources"]:
             filename = resource["public_id"].split("/")[-1]
             image_url = resource["secure_url"]
             cloudinary.uploader.download(image_url, f"{folder_name}/{filename}")
             print(f"Downloaded {filename} from Cloudinary")
```



```
[15] # Download images from Cloudinary
     for folder_name in ["Bank Statement", "Check", "ITR_Form 16", "Salary Slip", "Utility"]:
         download_from_cloudinary(folder_name, 50)
```

4. Automate the Upload and Download Process:

- Created a dataset folder variable dataset_folder to store the path to the dataset folder on Google Drive.
- Looped through the predefined folder names ("Bank Statement", "Check", "ITR Form 16", "Salary Slip", "Utility") and called the upload_to_cloudinary() function for each folder.
- Looped through the same folder names and called the download_from_cloudinary() function to download 98 images from each folder.

5. Testing and Verification:

- Ran the script and verified that the images were successfully uploaded to Cloudinary and downloaded from Cloudinary.
- Checked the downloaded images to ensure they were correctly saved in the respective folders.

6. Final Project:

Colab Notebook: Click Here to Access Colab Notebook