# **Instagram Scraper Documentation**

## **1. Steps**

### **Objective**

The script efficiently scrapes Instagram user profile data and post engagements (likes and comments) by:

1. Logging into Instagram.
2. Navigating to multiple user profiles.
3. Extracting user profile details (username, profile picture, followers, and following).
4. Dynamically scrolling to load posts.
5. Hovering over posts to extract engagement details.
6. Saving the collected data in a CSV file.

### **Process Steps**

#### **1. Logging In**

* The login\_instagram function automates the login process using Selenium WebDriver.
* It navigates to the Instagram login page and waits for the username and password fields to load.
* After submitting the credentials, the script waits until the homepage is loaded, verifying successful login.

#### **2. Navigating to Profiles**

* The script navigates to each Instagram profile URL listed in profile\_urls using driver.get().

#### **3. Scrolling to Load Posts**

* The scroll\_to\_load\_posts function dynamically scrolls until:
  + The required number of posts (default **50**) is loaded.
  + No more posts are available.
* It tracks unique post URLs using a set and ensures new posts are being loaded during each scroll.

#### **4. Scraping Engagement Details**

* The scrape\_engagement\_by\_hover function hovers over post thumbnails to extract likes and comments.
* A retry mechanism (up to **3 retries**) ensures robustness.
* Extracted engagement data is converted to integers using convert\_to\_number.

#### **5. Collecting Data**

* The scrape\_instagram\_user\_info function extracts:
  + **Username**
  + **Profile picture URL**
  + **Number of posts, followers, and following**
  + **Engagement details from posts**
  + **Average engagement rate**

#### **6. Saving Data**

* The script aggregates data from all profiles into all\_user\_data and saves it in **CSV format** using save\_to\_csv().

## **2. Data Modeling**

### **Input Data**

| **Parameter** | **Description** |
| --- | --- |
| **Credentials** | Instagram username and password for login. |
| **Profile URLs** | List of Instagram profile URLs to scrape. |

### **Output Data (CSV Format)**

| **Field** | **Description** |
| --- | --- |
| **user name** | Instagram username. |
| **user image** | URL of the profile picture. |
| **bio** | User's bio text. |
| **number of posts** | Total number of posts on the profile. |
| **number of followers** | Total number of followers. |
| **number of following** | Total number of accounts the user is following. |
| **last 50 posts** | List of post URLs. |
| **average engagement** | Average engagement across posts. |

## **3. High-Level Technical Solution**

### **Technology Stack**

* **Programming Language**: Python
* **Web Automation**: Selenium WebDriver
* **Data Storage**: CSV format using the csv module

### **Key Components**

#### **1. Selenium WebDriver**

* Automates browser interactions for login, navigation, scrolling, and hovering.
* Uses **ChromeDriver** with configurations for optimized performance.

#### **2. Error Handling & Retry Logic**

* Ensures graceful handling of:
  + Element loading delays.
  + Hover interactions failures.

#### **3. Dynamic Content Handling**

* Uses **scrolling & waiting** to ensure posts load dynamically before extraction.

## **4. Running the Code**

### **Prerequisites**

1. **Install Python 3.x** on your system.
2. **Install dependencies** using:  
    pip install selenium webdriver-manager python-dotenv

OR

pip install -r ./requirements.txt

### **Steps to Execute**

1. Copy the script into a Python file (e.g., instagram\_scraper.py).
2. Update **Instagram credentials** in .env:  
    INSTAGRAM\_USERNAME=your\_username

INSTAGRAM\_PASSWORD=your\_password

1. Add Instagram **profile URLs** to the profile\_urls list.
2. Run the script:  
    python instagram\_scraper.py
3. After execution, the output will be available in:  
    instagram\_data.csv

## **5. Test Plan & Results**

### **Test Plan**

#### **Objective**

* Validate whether the script successfully:
  + Logs in to Instagram.
  + Navigates to profiles.
  + Extracts profile details & engagements.
  + Handles errors and edge cases.

#### **Test Scenarios**

| **Test Case** | **Expected Outcome** |
| --- | --- |
| **Login with valid credentials** | Successfully logs into Instagram. |
| **Login with incorrect credentials** | Displays error: "Invalid credentials provided." |
| **Navigate to multiple profiles** | Successfully loads the profile pages. |
| **Scrape public profiles** | Extracts full user details. |
| **Handle private profiles** | Skips private profiles with an appropriate message. |
| **Scroll to load posts** | Dynamically loads up to **50 posts**. |
| **Extract likes & comments** | Hovering successfully retrieves engagement details. |
| **CSV export validation** | The instagram\_data.csv file is correctly populated. |

### **Sample Test Output**

| **User Name** | **Profile Picture** | **Bio** | **Posts** | **Followers** | **Following** | **Last 50 Posts** | **Average Engagement** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Cristiano Ronaldo | Image URL | Football Legend | 4000 | 600M | 500 | [Posts URL] | 78000.12 |
| Dua Lipa | Image URL | Singer & Artist | 2500 | 100M | 200 | [Posts URL] | 5200.45 |

## **6. Conclusion**

* This script **automates** Instagram profile data extraction using Selenium.
* Handles **dynamic content** (scrolling, hover interactions).
* Provides **robust error handling** with retry logic.
* Saves structured **CSV output** for further analysis.