**Project Title: Web Scraping Script for Organizational Details**

**Objective:**  
The aim of this project is to develop a Python-based web scraping script that extracts organizational information from the *culture360.asef.org* website's "Organisations Directory". This directory contains valuable details about various organizations, such as their name, country, website, and associated disciplines. The script automates the extraction of this information and saves it into an Excel file for further use.

**Process and Approach:**

1. **Tools and Libraries Used:**
   * **Selenium**
   * **pandas**
   * **webdriver\_manager**
2. **How the Script Works:**
   * **Starting Point:** The script begins by accessing the main URL of the *culture360.asef.org* website's organization directory.
   * **GDPR Consent:** The script waits for and clicks the GDPR consent button if it appears, ensuring the process runs smoothly without interruption.
   * **Pagination Handling:** The script extracts the total number of pages available for scraping and iterates over these pages to gather information from each one.
   * **Data Extraction:** On each page, the script extracts details about each organization, such as the name, website, country, and disciplines. It collects this information by targeting specific HTML elements using XPath selectors.
   * **Saving Data:** The scraped data is stored in a pandas DataFrame and saved to an Excel file, which allows easy inspection and further analysis of the data.
3. **Data Fields Collected:**
   * **Name**: The name of the organization.
   * **Website**: The organization's official website (if available).
   * **Country**: The country in which the organization is based.
   * **Disciplines**: A list of fields or categories the organization is associated with (e.g., Design, Arts, etc.).
   * **Link**: The URL of the organization’s detailed page on the website.

**Output and Results:**

* The final output is an Excel file named data8\_selenium\_9\_entries.xlsx, which contains the following columns:
  + Name: Organization name.
  + Website: Organization's website URL.
  + Country: The country where the organization is located.
  + Disciplines: Comma-separated list of disciplines associated with the organization.
  + Link: The webpage link to the organization's details.

**Challenges Faced:**

* **Handling Dynamic Content:** The website’s structure involves dynamic content that loads based on user interactions, so using Selenium was necessary to properly load and extract information from these elements.
* **GDPR Consent Handling:** Some pages required interaction with a consent button, which was handled using explicit waits in Selenium to ensure that the script continued without errors.
* **Pagination Handling:** Determining how many pages needed to be scraped and ensuring the script navigated correctly through them was crucial for gathering comprehensive data.

**How to Run the Project:**  
To run this project on your local machine, follow these steps:

1. **Prerequisites:**
   * **Python 3.6+** installed on your system.
   * **Chrome Browser** installed on your system.
   * Ensure that **Google Chrome** is updated to the latest version for compatibility with the **ChromeDriver**.
2. **Install Required Libraries:** Before running the script, you need to install the following libraries:
   * **Selenium**: To interact with the browser and automate the scraping process.
   * **pandas**: To handle the scraped data and save it into an Excel file.
   * **webdriver\_manager**: To manage the ChromeDriver installation and compatibility.

Install the libraries using pip:

pip install selenium pandas webdriver\_manager

1. **Run the Script:**
   * Download the script and place it in your project folder.
   * Ensure that your system has access to the necessary browser driver (ChromeDriver) for Selenium. The webdriver\_manager library handles this automatically.
   * Execute the script by running the following command in your terminal or command prompt:
2. python script\_name.py
   * The script will navigate through the website, extract data, and save it into an Excel file (data8\_selenium\_9\_entries.xlsx).
3. **View the Output:**  
   After the script finishes executing, open the generated Excel file to view the scraped data, including the organization's name, website, country, disciplines, and a link to each organization's page.

**Conclusion:**  
This project allowed me to learn how to use web scraping techniques to automate the collection of structured data from websites. By utilizing Python, Selenium, and pandas, I was able to build a practical tool for gathering and storing data from a directory, which could be used for research, analysis, or further processing.