AUTOMATE GOOGLE FORM

Presented By
Mansoor ul Hassan

Submitted To
Sir Muhammad Rizwan

Introduction

This project automates the process of submitting responses to a Google Form using Python, Selenium, and Pandas. It is especially useful for handling repetitive tasks where multiple entries need to be submitted based on pre-existing data. The automation reads data from a CSV file, navigates the form fields, inputs the required information, and submits the form on behalf of the user.

Objective

The primary objective of this project is to streamline the submission process for Google Forms. The script automates data entry by reading from a CSV file, reducing manual input effort, and ensuring accuracy in the data submission process.

Features

- **Automated Form Filling**: Automatically inputs data into text fields, radio buttons, checkboxes, and dropdown menus based on the data provided in the CSV file.
- **Dynamic Element Handling**: Utilizes Selenium's WebDriverWait to handle dynamic web elements, ensuring that the script works even with loading delays.
- **Customizable Inputs**: Reads fields such as Full Name, Preferred Method of Communication, Skills, Employment Status, and Meeting Date from the CSV file.
- **Form Submission**: After filling in the details, the form is submitted automatically, and the browser waits to confirm the submission before proceeding to the next entry.

Process Overview

1. **Data Loading**: The script loads data from a CSV file (info.csv). Each row in the CSV contains details for one form submission.
2. **Browser Automation**: The script uses Selenium WebDriver to launch Chrome, navigate to the Google Form URL, and fill in the form fields.

3. **Data Entry**: Each row from the CSV is processed, with the script entering data into the corresponding fields on the form.
4. **Form Submission**: After all fields are filled, the form is submitted, and the browser waits for confirmation before moving on to the next entry.
5. **Completion**: After processing all rows in the CSV, the browser is automatically closed.

Project Flow

The script runs through a few key steps:

1. **Initialization**: The Selenium WebDriver is initialized, and the browser window is maximized for optimal interaction.
2. **Form Navigation**: The script navigates to the Google Form URL and waits for the necessary elements to load.
3. **Data Input**: For each entry, the script fills in the form fields, selects checkboxes, radio buttons, and dropdowns, and enters the date.
4. **Submission**: Once all fields are populated, the form is submitted.
5. **Looping**: This process repeats for each row in the CSV file.

Screenshot

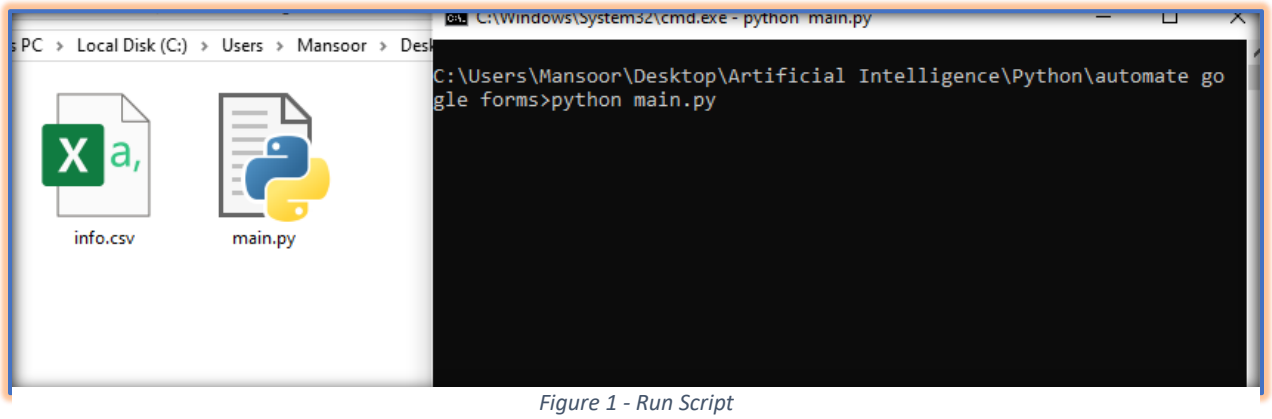

Figure 1 - Run Script

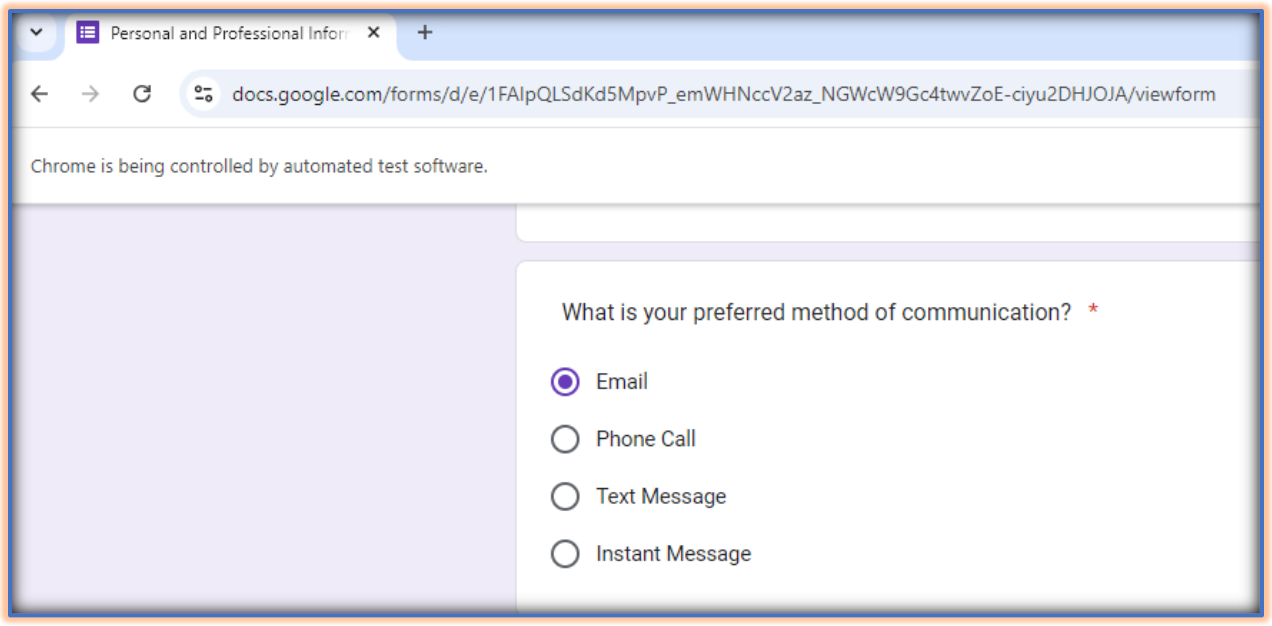

Figure 2 - Selenium Open Chrome and Start Filling Form

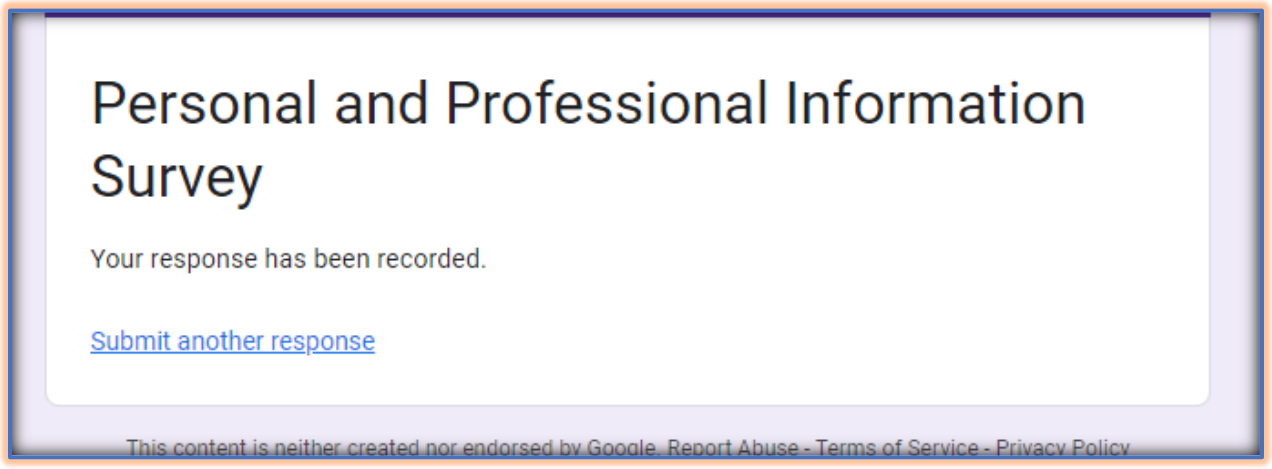

Figure 3 - Selenium Submitted Form

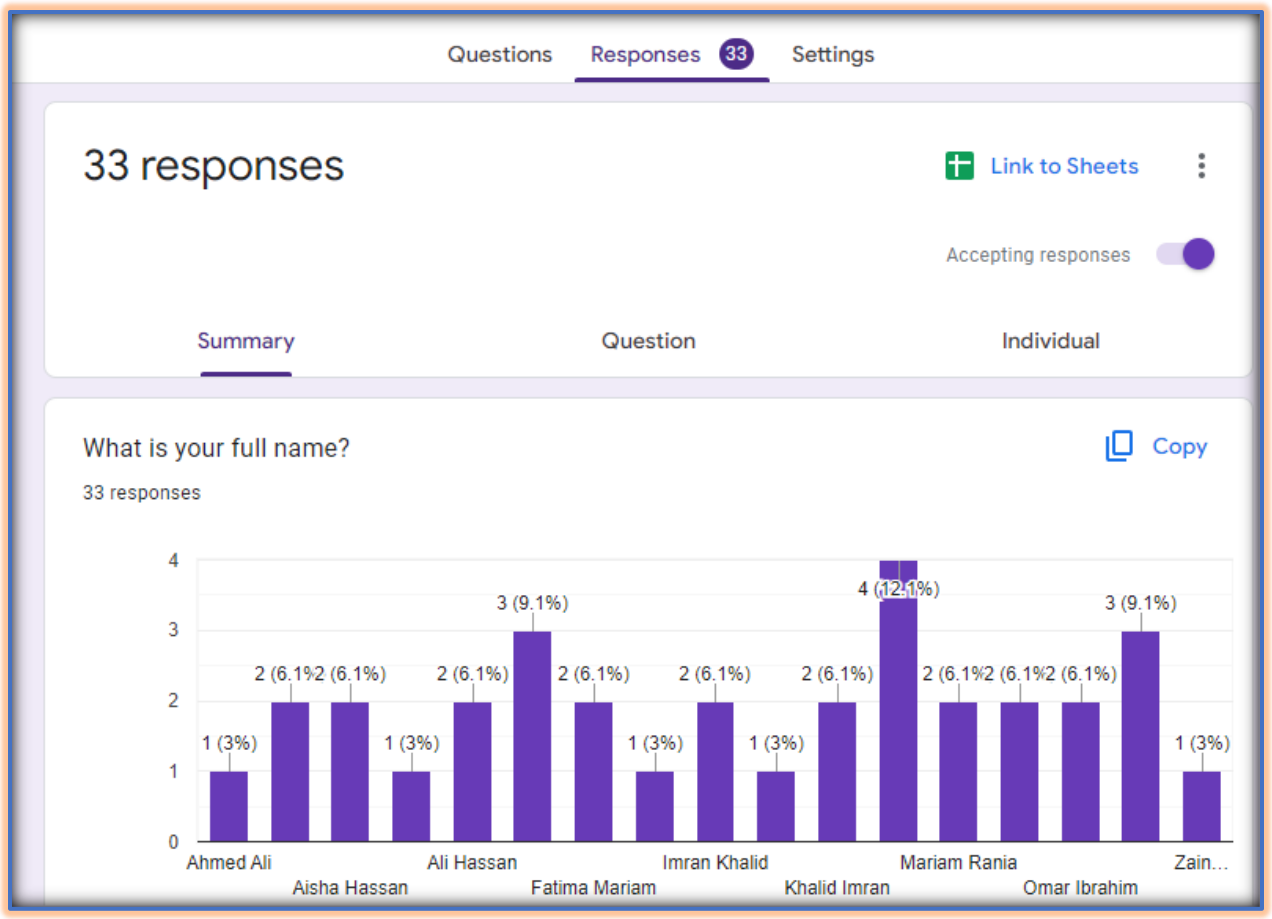

Figure 4 - Results after running complete script

These screenshot illustrates the automated form-filling process in action, showing how the script navigates through the Google Form and enters the required information from the CSV file.

Use Cases

This project is highly beneficial for:

- **Data Entry Automation**: When managing large volumes of form submissions based on pre-existing data.
- **Survey Participation**: Filling out multiple forms quickly and accurately for survey or registration purposes.
- **Batch Form Submissions**: Automating form submissions for events, registrations, or feedback gathering.

Future Enhancements

- **Error Logging**: Adding a logging mechanism to capture errors during the submission process.
- **Captcha Handling**: Integrating tools to handle CAPTCHA if encountered.

- **Multiple Form Support**: Extending the script to support different types of forms or dynamic form elements.

Repository

For further details and to access the code, please visit the GitHub repository: Google Form Auto-Filler

---

This report summarizes the Google Form Auto-Filler project. The repository linked above contains the complete code and setup instructions. Feel free to clone the repository and modify it as needed for your use case.
