Automating Invoice Data Collection Using Python and Selenium

Client

Our client is a leading provider of customized, integrated, and managed communications solutions for enterprise and federal customers, in the US Telecom Industry. Combining their comprehensive offerings and project management capabilities, the client offers a portfolio of customer-centric solutions that boost enterprise productivity, reduce costs, and simplify operations.

Objective

The objective of this engagement was to analyze, optimize, and automate the existing process of invoice data collection from a diverse set of web sources.

Challenges

A significant part of the services that our client provides is dealing with large number of invoices that needed to be aggregated from diverse sources on a regular basis. This was being done manually which was very unproductive and increased the company's costs and time.

To simplify and streamline this process automation was used to eliminate those tasks that were routine, repetitive and time consuming. As the sources for data collections were not uniform a big challenge for the tech team was to analyze and effectively integrate each source into our client's existing automation solution.

Another challenge was to safely handle additional measures like security questions, Captchas, and passwords without compromising any of our clients' confidential data during automation.

Solution

In order to integrate each source, Congruent leveraged the existing customized automation framework and enhanced its capabilities to download data in diverse scenarios. The framework was merged into a centralized code to enable orchestrating all data collection jobs from a single point of control.

The framework deployed a bot-controlled browser that could be dynamically managed based on the navigation steps preset in a user created configuration file. All configurations were designed and written to be fault tolerant keeping in mind the ever-changing landscape of website technologies.

Multiple tasks or jobs could be deployed simultaneously or set to run from a queue after which the downloaded files could be stored in a data base and further cleaned and processed as part of the extraction, transformation and loading of data. A summary report was generated after each run which increased transparency and turnaround time.

Benefits

By extending the system with robust configurations, Congruent was able to automate routine and repetitive tasks. We were also able to extend the solution with additional features to cover a wider range of data sources. And by keeping the code in a centralized repository, the ability and convenience to run data collection on different sources made a significant impact.

From starting out as manual data collection, 85% of the task was automated which made it possible to scale up rapidly to process half a million accounts each month. This helped empower employees to focus on more meaningful work while at the same time cutting down on time and costs and increasing turnaround time.

The system at present successfully supports around 3,400 vendors and over100,000 accounts for which invoices are downloaded automatically each month.

Technology Used

The framework was written in Python which is a solid choice as a language to build fast, flexible systems. It was designed to be modular and easily extensible.

The Selenium library was integrated into this system as it is a robust, tried and tested solution for automated and RPA systems, with good documentation, support and a large user base.

The browsers used for the Web drivers were Chromium and Chrome, which implement high security standards and frequent updates. These features make it extremely useful while working with newer web sites.

To create a configuration file to dynamically control navigation of the bots, YAML was used as this could be easily maintained, debugged, and extended.

And as YAML is also a simple and efficient data serialization language, development time was reduced, and efficiency was increased.

Technology Used: Python | Web Chromium | Selenium Library | YAML