Azati

Advanced Scraping Platform for Cellular Data Extraction

Our team developed an advanced scraping platform to help the customer receive daily phone call statistics. The solution consists of several scraping scripts that extract information from web UI with Selenium.

CUSTOMER:

If you are a business owner or a manager, you want to control the productivity of the sales department or call-center from anywhere.

Imagine a call center receiving and transmitting a large volume of inquiries. There is a considerable number of phone numbers from various providers, and each number can be used by several operators simultaneously. That is why it is quite complicated to track the quality of calls made by each employee, considering the amount of incoming or outgoing requests.

Our client manages such a call center, and it is impossible to integrate any call tracking or monitoring software into the existing infrastructure. It was necessary to collect cellular statistics in another way.

A considerable number of companies collect data to check service quality. If someone collects all the data manually, this person spends a lot of time doing routine work. There is specific software that can reduce the time spent on doing manual actions. But if you want to extract information about phone calls, the task becomes much more sophisticated.

Telecommunication providers store the list of calls and call history. Most often, it is stored in various databases as structured or semi-structured data. It is quite tricky to collect, process, systematize, and analyze this information. This is why there are several issues related to record matching and linking.

In our particular case, each provider had a dashboard where a user can see brief information about one cellphone number: bandwidth usage, call records, current balance, etc. If a company had hundreds of phone numbers, days needed for a regular person to manually collect the required information.

The customer wanted us to automate data extraction to cut down general costs, as there were several employees responsible for collecting this data.

OBJECTIVE:

The customer had already built a partially functioning MVP, but still, several employees were collecting and analyzing all the data manually. The previous vendor had successfully created five scrapers, one per each provider. The customer wanted us to finalize the application.

However, there were several issues with code quality and general performance – the solution worked incredibly slow. It did not provide the required insights by itself, as data required additional post-processing. Therefore, ongoing maintenance and monitoring are essential for scrapers to work effectively.

Since the project start, we were responsible for software improvements, feature development and bug fixing. As the previous vendor developed MVP two years ago, there was an issue with outdated technologies, and some algorithms also required significant tuning.

There were several challenges our team faced during the process.

CHALLENGE #1:

There were several issues with the existing scripts, and our customer requested a quick fix as soon as possible. One day the provider rolled out a new user interface, and scripts naturally stopped working. That's why the existing solution required timely ongoing updates.

In general, web scraping is a long and resource-intensive process, and it requires day-to-day monitoring. The improvement of the scraping algorithms should go in line with the development of authorization bypassing algorithms.

Our daily responsibility became checking out whether scripts are still able to work in the current technological environment.

CHALLENGE #2:

The same time as data collection became automated, several significant constraints came into sight. The more pages we parsed, the more time it took algorithms to process the data. With the increase of phone numbers, the number of pages to parse also began to grow. Hence it caused the problem with the lack of processing power and storage.

As this issue popped from time to time, it was easily fixed – we moved the solution to a high-performance dedicated server and set up a flexible resource orchestrator.

CHALLENGE #3:

Another issue was related to application performance.

The customer wanted us to provide the daily statistics generated by the solution. It means that the data collection and processing should be fast enough. It was a bit complicated to manage as the number of phone numbers was increasing over time.

Even today, we cannot guarantee 100% coverage as there are some issues with performance. But now our solution covers about 86% of all numbers required by the customer. Every month helps us to understand the pitfalls of the process and find out possible optimizations.

PROCESS:

As we mentioned earlier, there were five scrapers responsible for data collection: one per each provider. Later our engineers rebuilt one from scratch.

OUR TEAM PASSED THROUGH THREE CRITICAL PHASES WHILE BUILDING A WEB SCRAPER:

01. Passing authentication and authorization:

The scraper was designed to draw account data; that's why passing the authorization was critical. The customer provided us a spreadsheet with usernames (where phone number is a username) and passwords. This information helped the tool login under the guise of a regular user into operators' profile.

As login and password are required for a successful sign in, there are possible vulnerabilities related to this authentication method. If a third-party person knows the password, he can control the account. Providers worry about the safety of user data, so they built-in two-factor authentication.

More and more websites and services are willing to use a two-factor authentication. According to proponents, this approach drastically reduces the probability of online identity theft and other online fraud. The victim's password would no longer be enough to give a thief permanent access to their information.

As there was a considerable amount of cell numbers within the organization, asking everyone to provide us the SMS-code was a time-consuming and annoying process.

In order not to distract operators from work, our team built an intercepting module to pass a two-factor authentication automatically. It was working in the background, intercepting SMS from the provider and forwarding it to our email.

02. Building the scraper

When a web scraper signed in, we can collect all the information from the account. Script downloaded the HTML code into the cache and extracted the required information according to XPath.

THE SCRAPPER COLLECTED:

- The number of calls
- Calls duration
- Whether the call was accepted
- Whether the client called back to the operator
- Whether the operator called back to the customer
- How much calls were accepted by the operator
- Calls recordings
- How much traffic was spent

03. Matching the data and reports generation

Every day our business analyst receives the list of phone numbers customer wants us to check. Our engineers created an embedded database where the authorization information was matched to the corresponding phone number.

When the data collection is finished, a support specialist generates a CSV file and sends it to the customer. We do so, as web scrapers do not have any user interface, but it is a highly requested feature.

SOLUTION:

THE APPLICATION IS A GROUP OF INTERCONNECTED SCRIPTS, **THAT CAN BE DIVIDED INTO SEVERAL MODULES:**

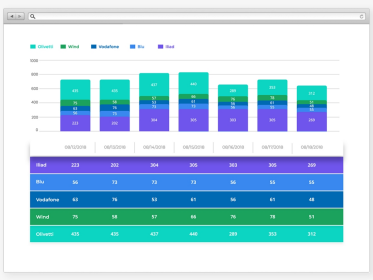
01	02	03
Authentication module to bypass two-factor authentication	**Five scrapers to execute the data extraction**	**Small script to analyze and match scraped data**

Once per day, the customer receives a report with all the required information.

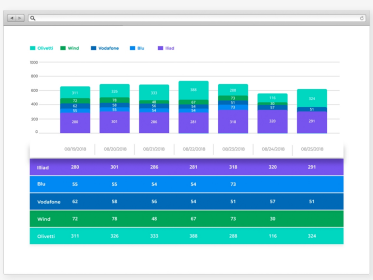
TECHNOLOGIES:

PostgreSQL · redis · Selenium · Sinatra

SCREENSHOTS:



	08/12/2018	08/13/2018	08/14/2018	08/15/2018	08/16/2018	08/17/2018	08/18/2018
Iliad	223	202	304	305	303	305	269
Blu	56	73	72	73	56	55	55
Vodafone	63	79	63	65	56	61	48
Wind	75	58	57	66	76	78	51
Olivetti	435	435	437	448	289	353	312



	08/19/2018	08/20/2018	08/21/2018	08/22/2018	08/23/2018	08/24/2018	08/25/2018
Iliad	283	307	286	281	318	320	291
Blu	55	55	54	54	73		
Vodafone	62	58	56	54	51	57	51
Wind	72	79	48	67	73	38	
Olivetti	311	326	333	308	288	116	324

RESULTS:

The project is not completed yet. Since the project has started, our team accomplished the following tasks related to feature development and ongoing updates:

01	02	03
OUR ENGINEERS REBUILT, TESTED, AND SUCCESSFULLY LAUNCHED A STABLE VERSION OF A SCRAPER	**OUR SUPPORT SPECIALIST PROVIDES DAILY STATISTICS TO THE CUSTOMER**	**WE BUILT A MODULE THAT CAN BYPASS TWO-FACTOR AUTHENTICATION AUTOMATICALLY**

THE CUSTOMER HAS **THE OPPORTUNITY TO:**

- Track the number and quality of calls made by each employee;
- Receive the call recordings;
- Track the quality of incoming calls (who is the operator and how many calls he received; how many calls were missed, the average duration of a call);
- View statistics of the incoming calls;
- View the average traffic consumption;
- How much calls were accepted by the operator

NOW:

Today our team is maintaining web scrapers, improving the stability of the system, as well as adding new features.

As the scraping process requires constant attention and day-to-day tuning, it is hardly possible for us to train someone without a specific technical background to do this work on site. This is why our engineers do everything possible to automate this process in order to help the customer cut down maintenance costs.

CONTACT US:

USA

184 South Livingston Avenue
Section 9, Suite 119
Livingston, NJ 07039

+1 201 464 6906

info@azati.com

www.azati.ai

BELARUS

K. Marx Street 31-509, Grodno,
230025, Belarus

+375 29 6845855

sales@azati.com