zyte

ZYTEPAPER

Enterprise web scraping: A guide to scraping at a scale

How to build a scalable web scraping infrastructure for your business or project.

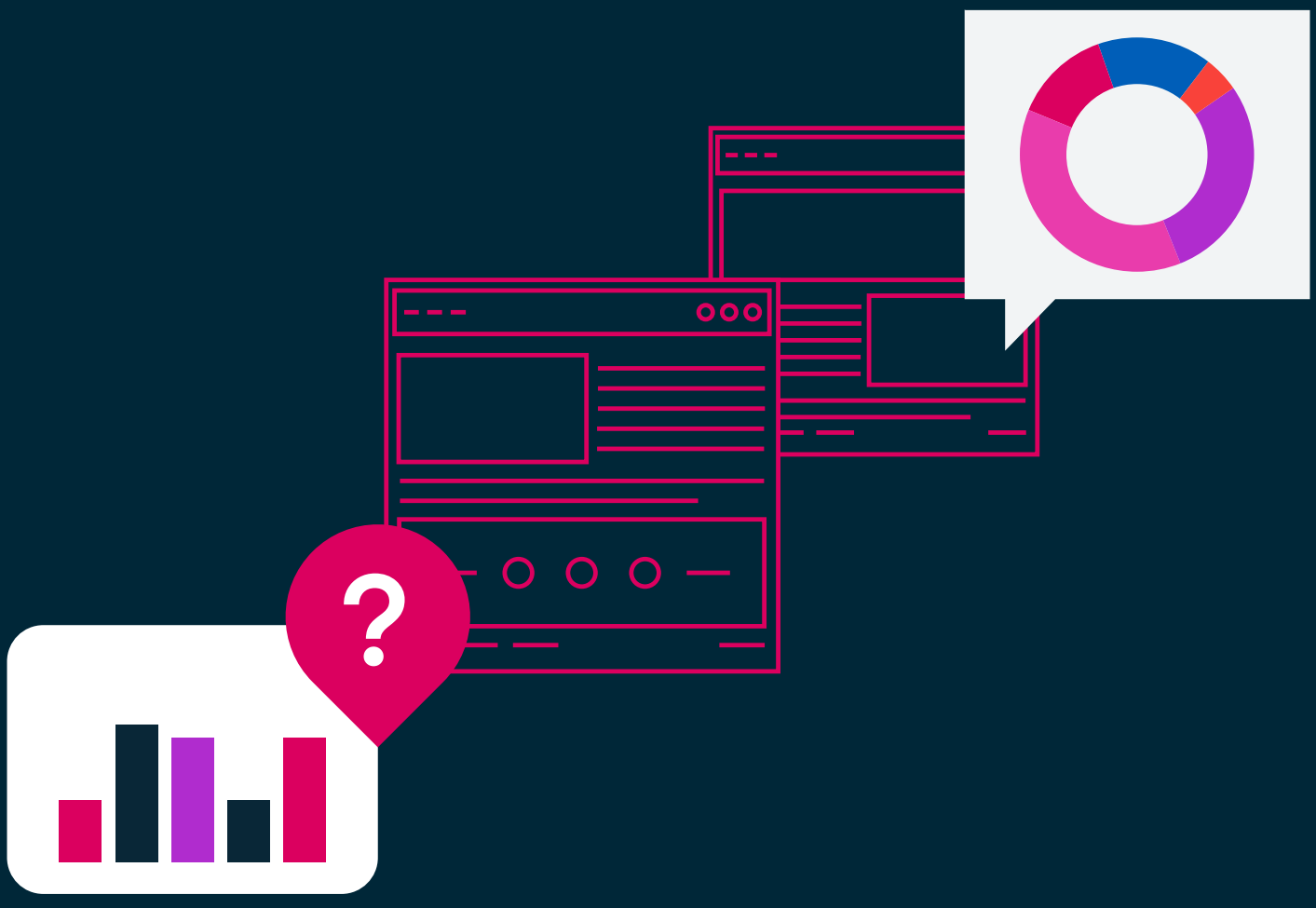
Introduction

Web scraping can look deceptively easy when your starting out. There are numerous open-source libraries/ frameworks, visual scraping tools and data extraction tools that make it very easy to scrape data from a website. However, when you want to scrape websites at scale, things start to get very tricky, very fast.

That is why it is critical that you build a web scraping infrastructure that is able to overcome the two biggest challenges of scaping the web at scale: **speed** and **data quality.**

As time is usually a limiting constraint, scraping at scale requires your crawlers to scrape the web at very high speeds without compromising data quality. This need for speed makes scraping large volumes of product data very challenging.

In this guide, we're going to discuss the **5 foundations** you need to put in place if you want to scrape the web at scale.



Foundation #1

Scalable architecture

The first building block of any large scale web scraping project is to development of a scalable architecture. Without an architecture that enables you to asynchronously scrape data from the web, you're unlikely to get enough throughput with your spiders.

The first step to do this is to:

Separate discovery spiders from extraction spiders

When scraping at a smaller scale, you can often get away with having the same spider discover and scrape records. However, when scraping at scale you really need to separate your discovery spiders from your extraction spiders.

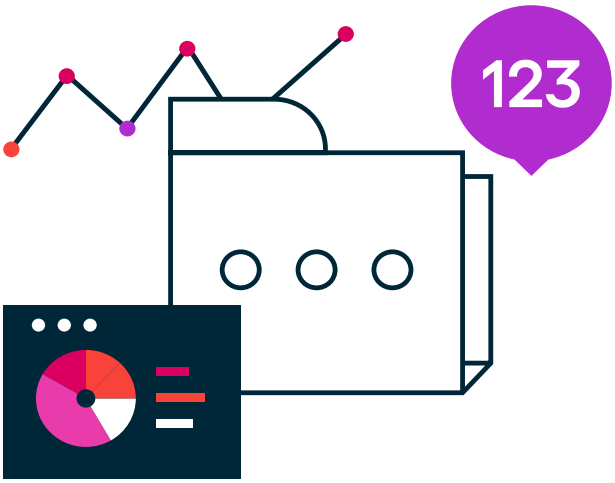
In the case of e-commerce, this would be developing one spider, the product discovery spider, to discover and store the URLs of products in the target category, and another spider to scrape the target data from the product pages.

Using this approach allows you to split the two core processes of the web scraping, crawling and scraping, and enables the allocation of more resources to one process over the other, and helping you avoid bottlenecks. Which brings us onto the next point.violating copyright.

Allocate more resources to your extraction spiders

Typically, in most web scraping projects, there will be some form of index page that contains links to numerous other pages that need to be scraped. In the case of e-commerce, these pages are typically category "shelf" pages that contain links to numerous product pages. For blog articles, there is alway a blog feed that contains links to each of the individual blog posts.

As each product category "shelf" can contain anywhere from 10 to 100 products and extracting product data is more resource heavy than extracting a product URL, discovery spiders typically run faster than product extraction spiders. When this is the case, you need to have multiple extraction spiders for every discovery spider. A good rule of thumb is to create a separate extraction spider for each ~100,000 page bucket.



Foundation #2

High performance configuration

After developing a scaleable architecture during the planning stages of your web scraping project, the next fundamental foundation you need to develop when scraping at scale is configuring your hardware and spiders for high performance.

Oftentimes, when developing enterprise scale web scraping projects, speed is the most important concern. In a lot of applications, enterprise scale spiders need to have finished their full scrape in a defined period of time. In the case of e-commerce, where companies are using web data to adjust their pricing, their spiders need to have scraped their competitors entire catalogue of products within a couple hours so that they can adjust.

In the case of travel comparison websites, they need to make sure their data is always up to date otherwise they mightn't be showing the correct prices.

This need for speed poses big challenges when developing an enterprise level web scraping infrastructure. Your web scraping team will need to find ways to squeeze every last ounce of speed out of your hardware and make sure that it isn't wasting fractions of a second on unnecessary processes.

To do this enterprise, web scraping teams need to develop a deep understanding of the web scraping framework they are using (for example Scrapy), build a robust proxy management infrastructure, and configure the hardware they are using so can get the optimal performance from their spiders.

The most important consideration for building a high throughput web scraping infrastructure **is spider design** and **crawling efficiency.**

Foundation #3

Spider design & crawl efficiency

When scraping at scale, you always need to be focused on crawling efficiency and robustness. Your goal should always be to solely extract the exact data you need in as few requests and as reliably as possible. Any additional requests or data extraction slow the pace at which you can crawl a website.

However, when scraping at scale, not only do you have to navigate potentially hundreds of websites with sloppy code, you will also have to deal with constantly evolving websites. A good rule of thumb is to expect your target website to make changes that will break your spider (drop in data extraction coverage or quality) every 2-3 months.

That mightn't sound like too big a deal, but when you are scraping at scale, those incidents really add up. For example, one of Zyte's larger e-commerce projects has ~4,000 spiders targeting about 1,000 e-commerce websites, meaning they can experience 20-30 spiders failing per day.

Unfortunately, there is no simple solution to these challenges. Most of the time, it is just a matter of committing more resources to your web scraping projects. To use the previous web scraping project as an example again, that project has a team of full-time 18 crawl engineers and 3 dedicated QA engineers to ensure the client always has reliable data feed.

However, with experience you'll begin to develop more robust and higher performance spiders that allow you to shave hours off scrapes and ensure that you're not constantly troubleshooting broken spiders. Here are some best practices to keep in mind:

- Instead of having multiple spiders for all the possible layouts a target website might use, it is best practice to have only one product extraction spider that can deal with all the possible rules and schemes used by different page layouts. The more configurable your spiders are the better.

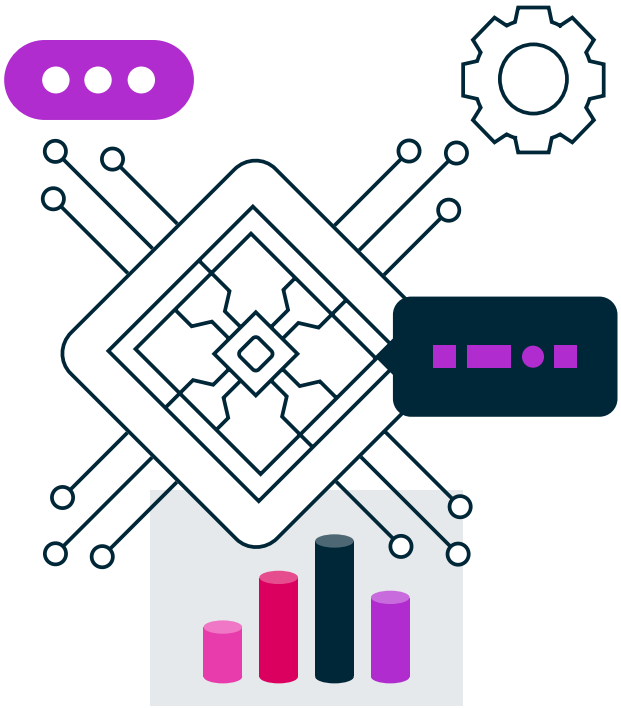
- Only use a headless browser, such as Splash or Puppeteer, to render javascript as a last resort. Rendering javascript with a headless browser whilst crawling is very resource intensive and severely impacts the speeds at which you can crawl.

- f you can get the data you need from a single page without requesting each individual item page then always confine your scraping to the index/category page. An example of this is scraping product data, if you can get the data you need from the shelf page (e.x. Product names, price, ratings, etc.) without requesting each individual product page, then don't request the product pages.

- Don't request or extract images unless you really have to.

Although these practices will make your spiders more complex (some of our spiders are thousands of lines long), it will ensure that your spiders are easier to maintain.

As most companies need to extract product data on a daily basis, waiting a couple days for your engineering team to fix any broken spiders isn't an option. When these situations arise, Zyte uses a machine learning based data extraction tool that we've developed as a fallback until the spider has been repaired. This ML-based extraction tool automatically identifies the target fields on the target website (product name, price, currency, image, SKU, etc.) and returns the desired result.



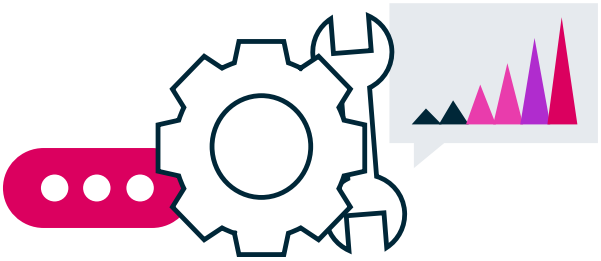
Foundation #4

Scalable proxy infrastructure

Alongside building scalable and robust spiders, you also need to build a scalable proxy management infrastructure.
Having a robust proxy management system is critical if you want to be able to reliably scrape the web at scale and target location-specific data. Without a healthy and well-managed proxy pool, your team will quickly find itself spending most of it's time trying to manage proxies and will be unable to adequately scrape at scale.

When scraping at scale, you will need a sizeable list ofproxies and will need to implement the necessary IP rotation, request throttling, session management and blacklisting logic to prevent your proxies from getting blocked.

Unless you have or are willing to commit a sizeable team to manage your proxies you should outsource this part of the scraping process. Your best option is to use a single endpoint proxy solution that takes care of all complexities of managing proxies.



Crawlera, the smart proxy network developed by Zyte, is one such single endpoint proxy solution that allows you to focus on the data, not proxies.

Unfortunately, just using a proxy service won't be enough to ensure you reliably scrape larger websites. More and more websites are using sophisticated anti-bot countermeasures that monitor your crawlers behaviour to detect that it isn't a real human visitor.

Not only do these anti-bot countermeasures make scraping e-commerce sites more difficult, overcoming them can significantly dent your crawlers performance if done incorrectly.

As a result, to ensure you can achieve the necessary daily throughput from your spiders you'll often need to design your spider to counteract anti-bot countermeasures without using a headless browser such as Splash or Puppeteer. These browsers render any javascript on the page but as they are very heavy on resources they drastically slow the speed at which you can scrape a website. Making them practically unusable when scraping at scale, other than a edge case where you have pursued every other available option.

Foundation #5

Automated data QA

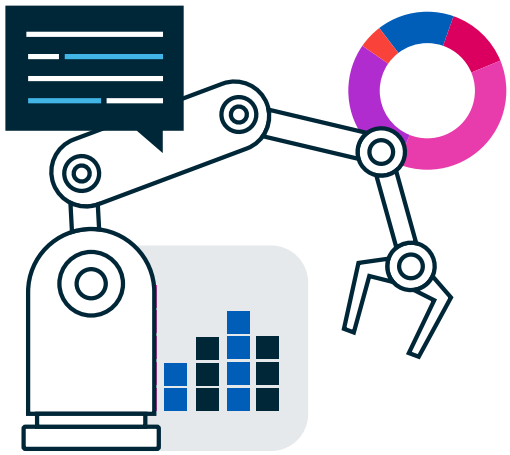
The key component of any large-scale web scraping project is having a system for automated data quality assurance.

Data quality assurance is often one of the most overlooked aspects of web scraping. Everyone is so focused on building spiders and managing proxies that they rarely think about QA until they are running into serious problems.

At its core, a web scraping project is only as good as the data it can produce. Even if you have the fanciest web scraping infrastructure on the planet, unless you have a robust system to ensure you are getting a reliable stream of highly qualified data your web scraping project will often be discontinued.

The key to data quality assurance for large-scale web scraping projects is making it as automated as possible. If you are scraping millions of records per day, it is impossible to manually validate the quality of your data.

At Zyte we recommend that you apply a similar four-layer QA process, as the QA process we apply to all the projects we undertake with clients.



Layer 1 – Pipelines

Pipelines are rule-based Scrapy constructs designed to cleanse and validate the data as it is being scraped.

Layer 2 – Spidermon

Spidermon is a spider monitoring framework we've developed to monitor and validate the data as it is being scraped.

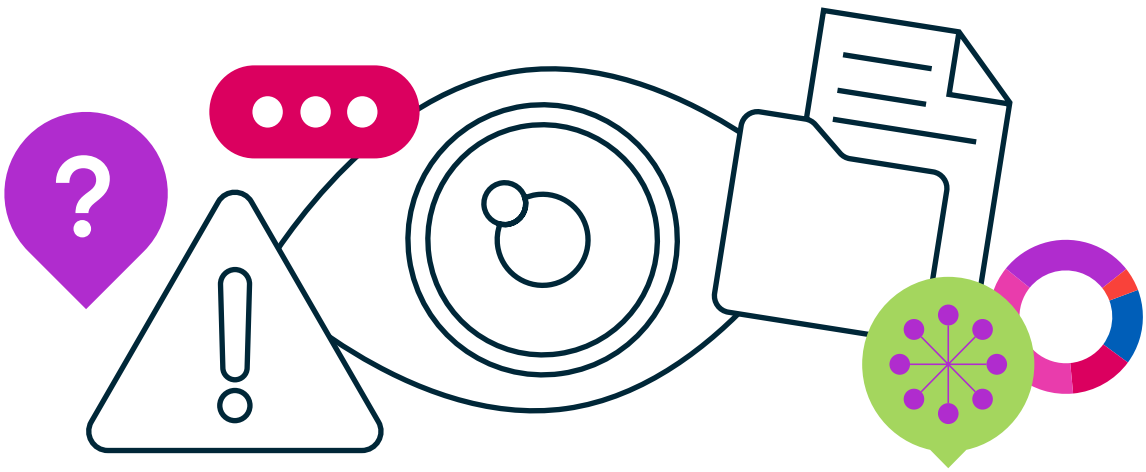
Layer 3 – Manually-Executed Automated QA

The third component of Scrapinghub's QA process are the Python-based automated tests our dedicated QA team develops and executes. During this stage, datasets are analysed to identify any potential sources of data corruption. If any issues are found, these are then manually inspected by the QA engineer.

Layer 4 – Manual/Visual QA

The final step is to manually investigate any issues flagged by the automated QA process and additionally manually spot check sample sets of data to validate that the automated QA steps haven't missed any data issues.

Only after passing through all four of these layers is the dataset then delivered to the client.



Enterprise web scraping: Build in-house or outsource?

This is the most frequent question we hear companies debating internally when it comes to enterprise web scraping.

Oftentimes, most of these companies already have developed some form of internal web scraping infrastructure, however, now it is either too much of a burden to handle or they want to scale it up but don't have the internal resources to do so.

In some cases, an in-house solution might make the most sense, depending on your current resources and needs. However, the question companies need to be asking themselves is: **how will these resources requirements and business needs evolve over time?**

Your goal isn't to just have a solution that meets your needs for today, but one that can evolve and meet all future needs as your requirements and goals change. For a lot of companies, this means outsourcing their web scraping to a dedicated web scraping firm is the best option. It allows them to have a world class web scraping infrastructure without the hassle of maintaining it, and if their business needs change their web scraping partner can quickly upgrade the infrastructure to meet those new business requirements.

For those of you who are interested in scraping the web at scale, but are wrestling with the decision of whether or not you should build up a web scraping team in-house or outsource it to a web scraping firm then be sure to set up a free consultation with our Solution Architect team, who will help you choose the best solution for your needs.

At Zyte we turn websites into data with industry leading technology and services.

Our solutions include:

- **Data Extraction Service**
  Let our web scraping experts build and manage the bespoke data extraction solution for your business needs.
- **Smart Proxy Manager (formerly Crawlera)**
  Forget about proxy lists. We manage hundreds of thousands of proxies, so you don't have to.
- **Automatic Extraction powered by AI**
  Instantly access accurate web data through our user-friendly interface or various Extraction APIs and save time getting the data you need.
- **Data extraction platform**
  Access developer tools, data extraction APIs and documentation, built and maintained by our world-leading team of over 100 extraction experts.

zyte

It's yours. The web data you need.

Access clean, valuable data with web scraping services that drive your business forward.

Talk to us

www.zyte.com

Cuil Greine House, Ballincollig Commercial Park Link Road, Ballincollig, Co. Cork, Ireland