

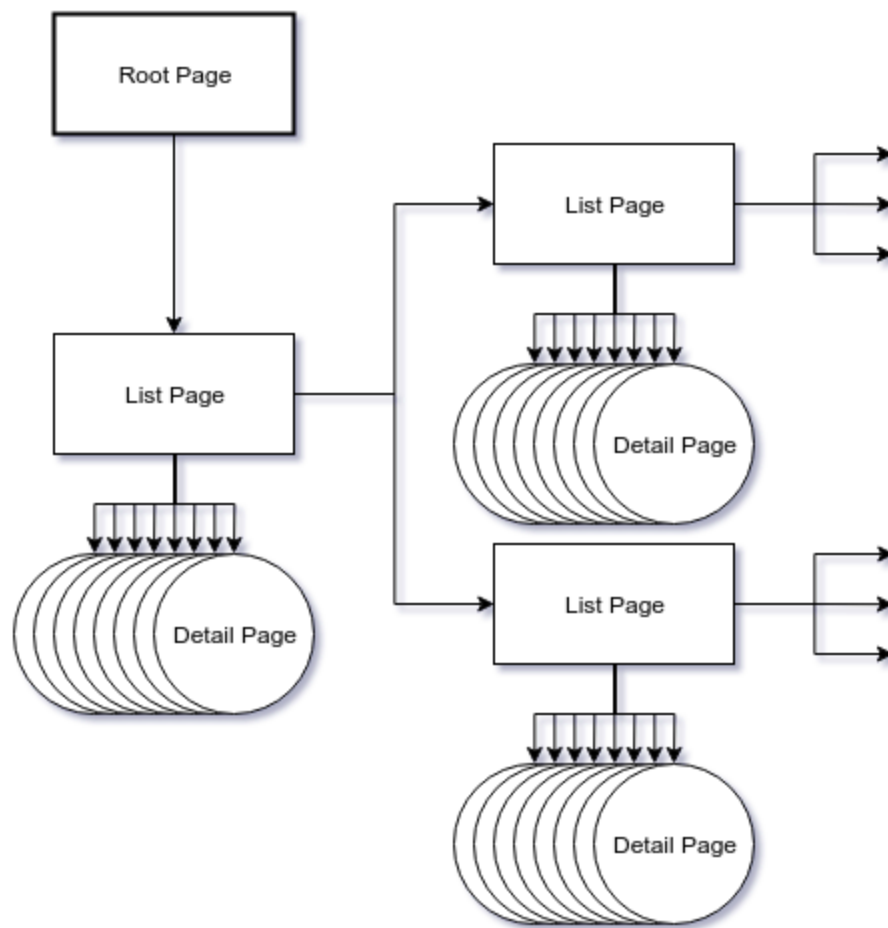
# BI-VWM | VII-1 Deep Web

## The objective of the thesis

The main objective of this thesis is to develop proof of concept implementation of web scraper and an API through which the scraped data can be accessed. The focus of the scraping are online bazaars and/or auctions and the ultimate goal is to build a technology which allows end user to search for a stolen item that is now being sold somewhere online. In order to achieve this goal the scraper needs to be (1) ready to scale and (2) programmed so that it can be easily extended to scrape through multiple different websites (Resources).

## Basic terms

To make it easy to add new Resources let's first look into what all online bazaars or auction houses have in common, based on this a general IResource interface can then be put together. Nearly all of these websites structure their content into categories and subcategories based on a category the user will receive a set of Items. This set can then sometimes be filtered and narrowed down further. Either way at the end the website presents a first page of the result set - let's call this page the **Root Page**. This page has a unique URL (determined by the category and/or the filters) and contains (1) a list of the Items that match the query (each links to it's **Detail Page**), (2) links to one or more pages that display other items that also match the query. Let's call each page that lists multiple items (including the Root Page) a **List Page**.



### *Basic Terms*

From the diagram above we can see that each Root Page defines a set of Items. This set will be a subset of all the items available on that Resource. Thus each Resource (e.g. sbazar.cz or cyklobazar.cz) can be divided into multiple Root Pages.

Let's store the Root Pages in a database table `root_pages`. Root Page has its `url` - that's where the scraper will start its work. Each page is also attached to a `resource` that way we'll know which pages to scrape.

## Implementation - Scraper

The goal of this project is to be able to scrape many different bazaar websites. To do this efficiently let's use docker containers and have one container for each Resource. All the scraper containers will run the same image and will pick the right

implementation of the general IResource interface (adapter pattern) - this way each container will be able to scrape and parse pages of given resource.

Based on the simple analysis and description of the basic terms this is the interface that each resource needs to implement:

```
interface IResource
{

    public function getName(): string;

    public function getBaseUrl(): string;

    // Processing List Page
    public function getDetailUrls(string $listPageBody): array
;

    public function getNextListsUrls(string $listPageBody): array;

    // Processing Detail Page
    public function parseDetailPage(string $pageBody): Item;

}
```

The scraper then calls these methods in this life cycle

1. Fetch Root Pages from database (`root_pages`) and push them into a List Pages queue
2. Start processing a queue of requests
3. When a List Page response is received use the adapter to parse it
  - a. `getDetailUrls` → push them into a Detail Pages queue
  - b. `getNextListsUrls` → push them to List Pages queue

4. When a Detail Page response is received use the adapter to parse it
  - a. `parseDetailPage` → Item → use storage to persist the item
5. End when there are no remaining requests in the queues

The scraper runs in PHP\* and is built on top of the [ReactPHP](#) framework. Used libraries include

- [symfony/Console](#) - to crate a cli interface for the scraper (`scrape` )
- [nette/DI](#) - dependency injection container
- [clue/reactphp-buzz](#) - “simple, async PSR-7 HTTP client for concurrently processing any number of HTTP requests”
- [symfony/dom-crawler](#)
- And few others - see `composer.json` for more details.

In summary, there is now one image (`scraper`) which can run in multiple containers (each container can run any of the implemented Resources). Next to these containers we have a MySQL database container. To persist the data even when the whole stack gets shut down there will also be one named volume which will be used by the database.

```
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/k.lucie/detail/75037002-detske-kolo-16-head
sbazar.dev | ✓ Item dětské kolo 16 Head saved
sbazar.dev | o Processing LIST: https://www.sbazar.cz/628-kola/cela-cr/cena-neomezena/nejnovejsi/2
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/euroalukola/detail/75058162-ghost
sbazar.dev | ✓ Item Ghost saved
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/jiri.brada2/detail/75021882-cyklo-trenazer
sbazar.dev | ✓ Item cyklo trenážer saved
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/darekg/detail/66573242-brasna-na-bok-nosice
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/jan.marsa/detail/75021047-alu-trekove-celoodpruzene-kolo-28-hlinikovy-ram-22
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/darekg/detail/66492327-zadni-nosic-sport-arsenal
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/darekg/detail/66484372-podsedlova-brasna-na-kolo
sbazar.dev | ✓ Item ALU trekové celoodpružené kolo 28 hliníkový rám 22 saved
sbazar.dev | ✓ Item Zadní nosič Sport Arsenal saved
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/darekg/detail/66485002-podsedlova-brasna-na-kolo
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/f.festa/detail/75025432-divci-kolo-dema
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/bazarkolal/detail/68010227-horske-kolo-merida-matts-sport-300-395
sbazar.dev | ✓ Item Divčí kolo Dema saved
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/darekg/detail/66484582-podsedlova-brasna-na-kolo
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/darekg/detail/66484512-ramova-brasna-na-kolo
sbazar.dev | ✓ Item Horské kolo Merida MATTS SPORT 300 [395] saved
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/darekg/detail/66595312-rychloupinak-pod-sedlo-novy
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/darekg/detail/66595437-hlavove-slozeni-ahead-nove
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/michal.dudek/detail/75057187-riditka-ritchey-comp-720mm
sbazar.dev | ✓ Item Riditka RITCHEY COMP 720mm saved
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/bazarkolal/detail/68007492-karbonove-horske-kolo-voice-xc-1293
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/michal.dudek/detail/75056727-kliky-shimano-deore-fc-m615
sbazar.dev | ✓ Item Karbonové horské kolo Voice XC [1293] saved
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/Sloupenska/detail/75056382-specialized
sbazar.dev | ✓ Item KLIKY SHIMANO DEORE FC-M615 saved
sbazar.dev | ✓ Item Specialized saved
```

*Running one scraper container (sbazar.dev). Scraping and saving bike items from sbazar.cz.*

```

cyklobazar.dev | o Processing LIST: https://www.cyklobazar.cz/retro-historicka-kola?vp-page=3
cyklobazar.dev | o Item Nabídka: Casovkářský speciál Favorit saved
cyklobazar.dev | o Item Nabídka: Favorit Passat saved
cyklobazar.dev | o Item Nabídka: Renovované kolo favorit, původní díly, velikost 54 saved
cyklobazar.dev | o Item Nabídka: Silnička saved
cyklobazar.dev | o Processing DETAIL: https://www.cyklobazar.cz/inzerat/334027/favorit-cronos
cyklobazar.dev | o Item Nabídka: Norco Search Gravel bike - nové kolo saved
cyklobazar.dev | o Item Nabídka: Favorit Cronos saved
cyklobazar.dev | o Processing DETAIL: https://www.cyklobazar.cz/inzerat/339074/s-works-crux-2018-vel-56cm-cca-178-187cm-skvely-stav
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/darekg/detail/66492327-zadni-nosic-sport-arsenal
cyklobazar.dev | o Item Nabídka: S-Works Crux 2018, vel 56cm (cca 178-187cm) Skvělý stav saved
sbazar.dev | o Item Zadní nosič Sport Arsenal saved
cyklobazar.dev | o Processing DETAIL: https://www.cyklobazar.cz/inzerat/334027/favorit-cronos
cyklobazar.dev | o Item Nabídka: Favorit Cronos saved
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/jiri.brada2/detail/75021882-cyklo-trenazer
cyklobazar.dev | o Processing DETAIL: https://www.cyklobazar.cz/inzerat/333780/specialized-sirrus-elite-carbon
cyklobazar.dev | o Processing DETAIL: https://www.cyklobazar.cz/inzerat/333748/merida-silex-7000-vel-m-177cm-2018
sbazar.dev | o Item cyklo trenážer saved
cyklobazar.dev | o Item Nabídka: Specialized Sirrus Elite Carbon saved
cyklobazar.dev | o Item Nabídka: MERIDA SILEX 7000, vel. M (177cm), 2018 saved
cyklobazar.dev | o Processing DETAIL: https://www.cyklobazar.cz/inzerat/333699/kellys-soot-50-2018
cyklobazar.dev | o Processing DETAIL: https://www.cyklobazar.cz/inzerat/333389/favorit-fl-super-special
cyklobazar.dev | o Item Nabídka: Favorit FL super special saved
cyklobazar.dev | o Processing DETAIL: https://www.cyklobazar.cz/inzerat/333406/merida-cyclocross-5v-vel-48-cm
sbazar.dev | o Processing DETAIL: https://www.sbazar.cz/f.festa/detail/75025432-divci-kolo-dema
cyklobazar.dev | o Item Nabídka: Merida Cyclocross 5V vel. 48 cm saved
cyklobazar.dev | o Processing DETAIL: https://www.cyklobazar.cz/inzerat/333577/kona-jake-the-snake-gravel-cyclocross

```

*Running two scraper containers. Scraping bikes from sbazar.cz and cyklobazar.cz.*

\* Yes, PHP supports concurrency trying this out was one of the motivations driving this project.

## Implementation - API

To build an API there are two more services running in this stack. PHP-FPM and NGINX web server are used for this purpose. There is be a simple [Nette](#) application which uses [NextrasORM](#) and [apitte](#) to map the data and expose a REST API.

As part of this project there only one endpoint (items) was implemented, however adding more endpoints in the future will be really simple.

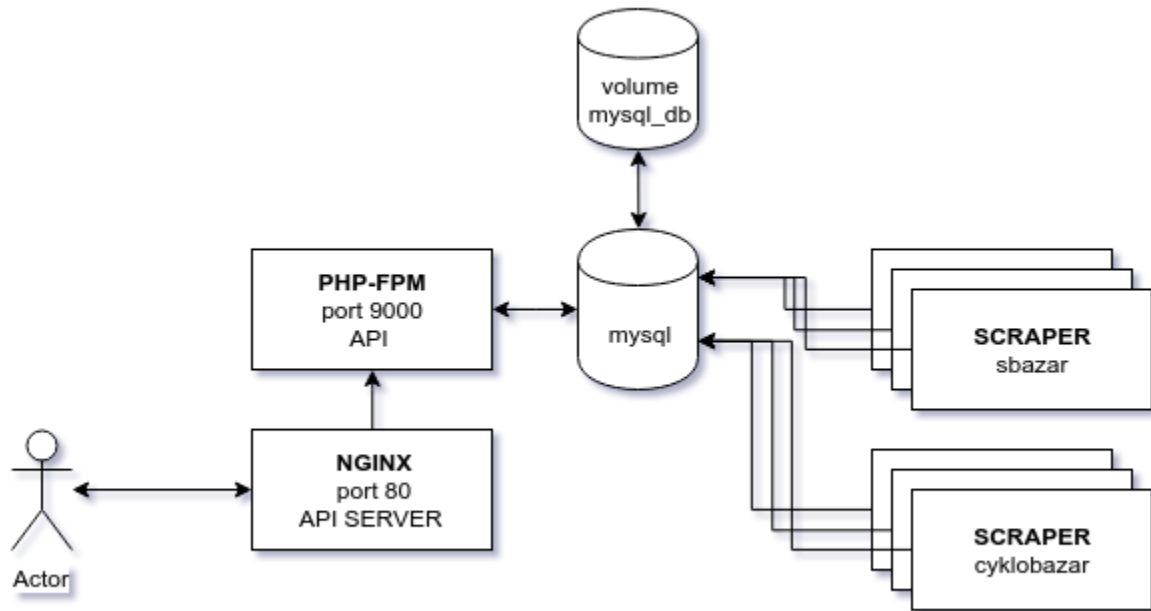
```

{
  "id": 1,
  "detailUrl": "https://www.sbazar.cz/ola.svadova/detail/75065527-kolo",
  "title": "kolo",
  "description": "prodán kolo"
},
{
  "id": 2,
  "detailUrl": "https://www.sbazar.cz/ola.svadova/detail/75065477-kolo",
  "title": "kolo",
  "description": "prodán staré kolo, k opravě"
},
{
  "id": 3,
  "detailUrl": "https://www.sbazar.cz/nacan89/detail/75017457-focus-velikost-n-ran-40-cm-velikost-kol-26-alu-",
  "title": "FOCUS velikost M rám 40 cm velikost kol 26 ALU-",
  "description": "Pěkné kolo Německé značky FOCUS velikost M rám 40 cm velikost kol 26 ALU-RAM přehazování schinano ,odpružená přední vidlice .Kolo je opravdu moc pěkné možno zaslat i na dobírku .Koukněte na mou nabídku."
},
{
  "id": 4,
  "detailUrl": "https://www.sbazar.cz/zdoxansky/detail/69831647-silnicni-kolo-giant-mozno-i-s-pedaly-a-tretram",
  "title": "Silniční kolo Giant, možno i s pedály a tretram",
  "description": "Prodám silniční Giant OCR 35,5cm. Gazetyn Shinano Sora. Rafky Mavic CXP21, Sedlo Selle Italia, košíky lahvi Zefal. Perfektní stav. Velikost: XL / 21-23" / 180-200 cm. Cena 6.500,- Kč nebo s pedály Shinano PD-1 cyklistickými tretram Shinano RTS1 SPD ve. 44 45 za celkovou cenu 7.000,- Kč. Ve všední dny volat pouze po 15. hodině."
},
{
  "id": 5,
  "detailUrl": "https://www.sbazar.cz/blkeriv/detail/45230278-predstavec-easton-delka-110mm-uhel-5-a-5-stupnu",
  "title": "Představec Easton délka 110mm, úhel 5 a 5 stupnu",
  "description": "nové/nepoučené. Ideální osobní odběr Brno - Královo Pole, Jundrov. Inzerát platí do smazání. K dispozici 1 kus."
},
{
  "id": 6,
  "detailUrl": "https://www.sbazar.cz/blkeriv/detail/26137947-nove-riditka-easton-mtb-660mm-318mm-359g",
  "title": "NOVÉ řídítka Easton MTB 660mm, 31.8mm, 359g",
  "description": "Úhel 9 stupnu, hmotnost 359 gramů - váženo. Monkey Bar Originální/aktuální foto. Inzerát je platný do smazání. Ideální osobní předání v Brně - Královo Pole, Jundrov."
},
{
  "id": 7,
  "detailUrl": "https://www.sbazar.cz/Veruna.Krausova/detail/74785947-panske-celoodpruzene-kolo",
  "title": "Pánské celoodpružené kolo",
  "description": "Prodám pánské celo odpružené kolo ve skvělém stavu - viz foto. Kolo bylo dělané na zakázku. Velikost kol 26". V případě dotazů piště."
},
{
  "id": 8,
  "detailUrl": "https://www.sbazar.cz/karelvoel59/detail/75044837-horske-kolo-29-merida-big-nine-xt-edition",
  "title": "Horské kolo 29" Merida Big Nine XT edition",
  "description": "Nabízím k prodeji tohle velmi lehké a výborně vybavené horské kolo Merida Big Nine XT edition. Původní cena kola 36.990 Kč, prodám za 17.900 Kč. Velikost kol 29". Velikost rámu 21" - pro výšku postav cca 171-190 cm. Základ kola tvoří velmi lehký rám typu Big 9 TFS-single ALU 6066 TFS (Techno Forming System), díky kterému je reálná celková hmotnost kola vč. pedálů jen cca 12,5 kg. Kolo disponuje vzduchovou vidlicí Rock Shox 30 TK29 se zdvihem 100 mm, regulací rychlosti tlumení a dálkovým uzamykáním z řídítek. Hydraulické brzdy s pákami Shinano Deore M506, trženy Shinano M447, kotouče 180 mm vpředu a 160 mm vzadu. Přehazovačka Shinano XT Shadow-uzamykatelný ramínkem. Přesmykač Shinano XT triple. Řazení Shinano XT 40/30/22 zubů. Převody 10x3 30 rychlostí. 10-ti rychlostí kazeta Shinano CS-M50-10 11 až 36 zubů. Nabíje kol Shinano M4050. Na kol"
}

```

*Example of the api/v1/items response*

## Summary of the stack



*Bazaar Scraper stack*

## Performance and throttling

The REST API is not interesting from this perspective, let's focus on the Scrapping. ReactPHP abstracts away the exact mechanisms of how concurrency is achieved using its EventLoop\Factory by default the [StreamSelectLoop](#) which makes use of the `stream_select()` method is used. This works out of the box since PHP 5.3.

Quoting from the documentation of ReactPHP:

*Under the hood, it does a simple `select` system call. This system call is limited to the maximum file descriptor number of `FD_SETSIZE` (platform dependent, commonly 1024) and scales with  $O(m)$  ( $m$  being the maximum file descriptor number passed). This means that you may run into issues when handling thousands of streams concurrently and you may want to look into using one of the alternative event loop implementations listed below in this case.*

<https://reactphp.org/event-loop/#streamselectloop>

EventLoop: EventLoop Component - ReactPHP •

This is the implementation used in this project and other alternatives have not been tested. However since the stack is built using docker it would be fairly simple to add the extra php extensions needed.

The main challenge of this implementation is that even though PHP supports concurrency it still runs in one thread and thus - deep down - everything is blocking. Docker containers to the rescue! As described in the previous section we can run multiple containers with the same Resource adapter. Each of the containers is independent and all of these containers can run on one host. This way asynchronicity can be achieved.

Next possible limiting factor could be the database if we're running many containers we would be keeping a lot of open connections to the database and we'd be querying the data very often the performance could decrease. However MySQL database is very well optimised to handle this.

Other factors that limit the performance of the scraper include: the internet connection of the server running the scraper, the response time of the requested website, the speed at which the page can be parsed.

Moving onto the side of the website that is being scraped. Some websites as part of their DDoS protections etc will block IP that send too many requests (also let's be the good guys and don't abuse the target websites - there are developers on the other side). To address this issue the scraper implementation uses (in memory) queue which limits the number of concurrent requests can be sent in the same time. This can be changed in the configuration (`config.neon`) of the scraper.

## Where do we go from here?

In this section let's discuss the shortcomings of this implementation and how it can be improved so that it is not just a proof of concept but a technology that can be used in the real world.

## Multiple scraping modes

As described in the first section of this document, the scraper now goes into the detail page of every item. This was the most information can be retrieved, however most of the time the List Pages already contain a lot of information about each item (title, short description, price, small photo). For some use cases this could be good enough. It would probably be useful to have two different scraping modes:

1. Brief - only collect the information from the List Pages, don't query the Detail Pages at all. This will be significantly faster.
2. Thorough - query each detail page and parse the complete information from there (currently implemented)

## Extend the data collected

As a proof of concept the current implementation is collecting only few parameters for each item, it would be fairly easy to extend this. The Item entity could also potentially be extended by some JSON field where all other information could then be stored. All this could then be pushed into something like elastic search for better performance.

## Download images and use them for searching

This could be the killer feature of this technology. Collect the URLs of the images, have another service that will be responsible only for downloading and processing of these images. The end user would then be able to search for items based on an image. When applied to the use case of searching for stolen items online this could be really beneficial in reducing the number of items that the end user needs to go through.

## In conclusion

The goal of build a proof of concept technology has been achieved. There is more work to be done to actually turn this into something that help people find their stolen bikes, but this is just the first step.

## Resources

- <https://sergeyzhuk.me/2018/02/12/fast-webscraping-with-reactphp/>
- <https://reactphp.org/event-loop>



- <https://github.com/apitte/core/tree/master/.docs>

Antonín Vlček

vlcekan1@fit.cvut.cz

 @tonyvlcek