UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CURSO CIÊNCIA DA COMPUTAÇÃO

Alan Djon Lüdke

MARKETSCRAPER: UM SCRAPER PARA A BUSCA E EXTRAÇÃO DE INFORMAÇÕES DE PRODUTOS EM SUPERMERCADOS NA REGIÃO DE FLORIANÓPOLIS

Florianópolis
2023

Alan Djon Lüdke

MARKETSCRAPER: UM SCRAPER PARA A BUSCA E EXTRAÇÃO DE INFORMAÇÕES DE PRODUTOS EM SUPERMERCADOS NA REGIÃO DE FLORIANÓPOLIS

Trabalho de Conclusão de Curso submetido ao curso de Ciência da Computação do Centro Tecnológico da Universidade Federal de Santa Catarina como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação

Orientador: Prof. André Wust Zibetti, Dr.

Florianópolis

2023

Lüdke, Alan
MARKETSCRAPER : UM SCRAPER PARA A BUSCA E EXTRAÇÃO DE INFORMAÇÕES DE PRODUTOS EM SUPERMERCADOS NA REGIÃO DE FLORIANÓPOLIS / Alan Lüdke ; orientador, André Wust Zibetti, 2023.
56 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro Tecnológico, Graduação em Ciências da Computação, Florianópolis, 2023.

Inclui referências.

1. Ciências da Computação. 2. Comparativo de preços. 3. web scraping. I. Zibetti, André Wust. II. Universidade Federal de Santa Catarina. Graduação em Ciências da Computação. III. Título.

Alan Djon Lüdke

MARKETSCRAPER: UM SCRAPER PARA A BUSCA E EXTRAÇÃO DE INFORMAÇÕES DE PRODUTOS EM SUPERMERCADOS NA REGIÃO DE FLORIANÓPOLIS

Este trabalho de conclusão de curso é submetido ao corpo docente do Departamento de Informática e Estatística da Universidade Federal de Santa Catarina como um dos requisitos para a obtenção do título de Bacharel em Ciência da Computação.

Florianópolis, 14 de Junho de 2023.

_____

Coordenação do Curso

Banca examinadora

_____

Prof. André Wust Zibetti, Dr.
Orientador
Instituição UFSC

_____

Prof.(a) Carina Friedrich Dorneles, Dr.(a)
Avaliadora
Instituição UFSC

_____

Prof. Elder Rizzon Santos, Dr.
Avaliador
Instituição UFSC

Este trabalho é dedicado aos meus amigos, família e mentores que me ajudaram a seguir em frente quando eu não acreditava ser possível

AGRADECIMENTOS

Àqueles a quem devo tudo: pai e mãe, por todo suporte e amor ao longo de toda minha trajetória. Aos meus irmãos, que sempre serviram de inspiração e exemplo. Meu orientador, André Wust Zibetti que auxiliou durante todo o desenvolvimento do trabalho. Aos verdadeiros amigos que conheci na graduação. A todos os professores que tive ao longo da vida, em especial aos da UFSC, por todos ensinamentos, os quais carregarei comigo no meu dia a dia profissional.

RESUMO

Durante a pandemia, observou-se um expressivo aumento na demanda por alimentos e produtos essenciais, impulsionando a compra online desses itens. Diante desse cenário, as empresas se adaptaram rapidamente, disponibilizando seus produtos em lojas virtuais para atender às necessidades dos consumidores. Nesse contexto, surge a necessidade de uma ferramenta que facilite a comparação de preços entre diferentes supermercados online. Com base nessa demanda crescente, este trabalho tem como objetivo principal o desenvolvimento de um conjunto de scrapers capazes de extrair e organizar dados sobre produtos e preços de diferentes supermercados da região de Florianópolis: Angeloni, Bistek, Fort, Giassi e Imperatriz. Essa ferramenta permite a centralização dessas informações em uma única fonte, possibilitando aos consumidores comparar preços de produtos e acessar um histórico de preços. Com a implementação desses scrapers, busca-se fornecer uma solução eficiente para auxiliar os consumidores na tomada de decisões informadas, contribuindo para uma experiência de compra mais satisfatória e transparente no mercado de supermercados online.

Palavras-chave: scrapping; web scrappers; raspagem de dados; comparativos de preços;

ABSTRACT

During the pandemic, there has been a significant increase in the demand for food and essential products, leading to a surge in online purchases of these items. In response to this situation, companies quickly adapted by making their products available in virtual stores to meet consumer needs. In this context, there arises a need for a tool that facilitates price comparison among different online supermarkets. Based on this growing demand, this work aims to develop a set of scrapers capable of extracting and organizing data on products and prices from different supermarkets in the Florianópolis region: Angeloni, Bistek, Fort, Giassi, and Imperatriz. This tool enables the centralization of this information in a single source, allowing consumers to compare product prices and access a price history. By implementing these scrapers, the goal is to provide an efficient solution to assist consumers in making informed decisions, contributing to a more satisfying and transparent shopping experience in the online supermarket market.

Keywords: scraping; web scrapers; data scraping; price comparison;

LISTA DE FIGURAS

LISTA DE CÓDIGOS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

SQL	Structured Query Language
API	Application Programming Interface
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
URL	Uniform Resource Locator
XML	Extensible Markup Language
APP	Aplicativo
TCC	Trabalho de Conclusão de Curso
CSS	Cascading Style Sheets
HTTPS	HyperText Transfer Protocol Secure

SUMÁRIO

1 INTRODUÇÃO

O aumento da disponibilização de informações na internet para auxílio na tomada de decisão tem trazido inúmeras facilidades para os consumidores, entre elas a capacidade de pesquisar preços de produtos em supermercados. Essas informações são extremamente valiosas para o consumidor comum, pois permitem comparar preços e encontrar as melhores ofertas. No entanto, a navegação entre diferentes plataformas de supermercados ainda é um desafio, dificultando a comparação de preços entre diferentes estabelecimentos.

Na região de Florianópolis, assim como em outras áreas metropolitanas, a importância das informações de produtos em supermercados é amplificada pela abundância de estabelecimentos comerciais nesse setor. Segundo estudo de DELUCA (2001), o setor supermercadista na região tem um papel fundamental na economia local, contribuindo para o desenvolvimento regional e gerando empregos. No entanto, os consumidores enfrentam desafios para obter informações consistentes e atualizadas sobre os produtos oferecidos por diferentes supermercados na região.

A necessidade de informações precisas e comparáveis entre supermercados motivou o desenvolvimento de técnicas automatizadas para aquisição desses dados. Segundo análise do Departamento Intersindical de Estatística e Estudos Socieconômicos (Dieese) realizado no período de janeiro a dezembro de 2022 o encarecimento da cesta básica em Florianópolis foi de R$ 73,00 ou pouco mais de 10%. Ter a disponibilidade de informações precisas e confiáveis promove transparência no mercado de supermercados, estimulando a concorrência entre os estabelecimentos e consequentemente a preços mais competitivos.

Os crawlers ou scrapers têm sido amplamente utilizados para coletar informações de produtos em diferentes setores, incluindo supermercados. Essas ferramentas automatizadas têm a capacidade de extrair dados como preços, descrições de produtos e categorias de maneira eficiente e em grande escala.

Apesar dos avanços alcançados por trabalhos anteriores, algumas limitações permanecem. Por exemplo, conforme mencionado por Arthur, et al (2019), muitos trabalhos focam em um único supermercado, têm escopo limitado geograficamente ou então utilizam de APIs prontas para construírem as aplicações. Isso restringe a

abrangência das informações disponíveis para os consumidores e limita a tomada de decisão.

Enquanto existem trabalhos que abordam a coleta de informações de produtos em supermercados, é importante ressaltar que esta pesquisa se diferencia por considerar aspectos específicos da região de Florianópolis. Além disso, para aqueles trabalhos que abordam a região, propõe-se o aumento da gama de scrapers de supermercados na região. O trabalho também tem o potencial de gerar contribuições sociais, econômicas e tecnológicas, promovendo a transparência e a concorrência saudável no comércio varejista de alimentos da região.

Os próximos capítulos estão organizados da seguinte forma: No Capítulo 2, apresenta-se a fundamentação teórica, abordando a extração de dados na web, com destaque para o web scraping, e o framework Scrapy. No Capítulo 3, discute-se trabalhos relacionados, incluindo aplicativos e plataformas web, bem como trabalhos de conclusão de curso relevantes para o tema. O Capítulo 4 apresenta o Market-scraper, fornecendo uma visão geral do projeto, as ferramentas e bibliotecas utilizadas, a fonte de dados e as convenções adotadas para a padronização das informações. São detalhadas as etapas de extração dos dados, incluindo a análise de sitemaps, catálogos e produtos de diferentes supermercados. Descrevem-se também o tratamento e sanitização dos itens, o armazenamento dos dados e a validação da ferramenta, mostrando alguns resultados conquistadas com o uso da mesma. Por fim, o Capítulo 5 traz as considerações finais e sugestões para trabalhos futuros.

2 OBJETIVO GERAL

Construir um conjunto de scrapers que farão a aquisição de dados de produtos de supermercados online da região de Florianópolis;

2.1 OBJETIVOS ESPECÍFICOS

- Identificar potenciais supermercados e avaliar a possibilidade de extração de dados;
- Propor e implementar estratégias de extração de dados de produtos;
- Propor um pipeline de extração, conformação e armazenamento;
- Organizar os dados extraídos em um banco de dados estruturado;
- Armazenar os dados extraídos e o log de extração em uma base.

3 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são descritas as definições e os conceitos necessários para melhor compreensão do projeto.

3.1 EXTRAÇÃO DE DADOS NA WEB

A extração de dados na web refere-se ao processo de coletar informações disponíveis em páginas da web para posterior análise e uso. Com o rápido crescimento da quantidade de dados disponíveis online, a extração de dados na web tornou-se uma área de pesquisa e desenvolvimento de grande importância. A extração de dados pode ser aplicada em diversas áreas, como pesquisa acadêmica, análise de mercado, monitoramento de preços, entre outros.

Segundo Mendonça (2003), as fontes de informação são sistemas que atendem a consultas, fornecendo uma resposta adequada para cada solicitação submetida. No contexto da Internet, essas fontes possuem formatos textuais e são manipuladas pelo paradigma de requisição e resposta implementado pelo protocolo Hypertext Transfer Protocol (HTTP). Geralmente, fazem uso dos formatos semi estruturados como o HTML e o XML.

Ao realizar a extração de dados na web, é fundamental compreender a estrutura das páginas e os formatos dos dados desejados. Os dados podem ser estruturados, como tabelas e listas, ou não estruturados, como texto livre e imagens (MITCHELL, 2019).

O uso de web crawlers ou web scrappers tornou-se cada vez mais comum nos últimos anos. Essas técnica permite aos usuários extrair dados de sites e utilizá-los para diversos fins. Indivíduos e empresas podem coletar grandes quantidades de dados de várias fontes e usá-los para tomar decisões informadas. Isso levou ao aumento da concorrência entre as empresas, que se esforçam para oferecer as melhores informações aos seus clientes (Glez-Peña, D. et al., 2013).

O web scraping envolve a extração de dados específicos de uma página da web, como textos, imagens, tabelas ou qualquer outro tipo de conteúdo estruturado. É uma abordagem mais direcionada, na qual os dados são extraídos de maneira seletiva com base em critérios específicos. Geralmente, o web scraping é realizado

por meio de programas ou scripts que analisam o código HTML das páginas para identificar e extrair os elementos desejados (Calò, 2014).

A prática de extração de dados da web, conhecida como Web Scraping, consiste em obter de forma direta informações significativas de páginas da internet, por meio da utilização de bots, para posterior análise dos dados extraídos.

> [...] os web scrapers podem acessar lugares que as ferramentas de pesquisa tradicionais não conseguem. Um pesquisa no Google por “voos mais baratos para Boston” resultará em uma grande quantidade de anúncios publicitários e sites populares para busca de voos. O Google sabe apenas o que esses sites dizem em suas páginas de conteúdo, mas não os resultados exatos de várias consultas fornecidas a uma aplicação de busca de voos. No entanto, um webscraper bem desenvolvido pode colocar em um gráfico o custo de um voo para Boston ao longo do tempo para uma variedade de sites e informar qual é o melhor momento para comprar uma passagem. (Mitchell, 2019, p. 12-13)

3.2 FRAMEWORK SCRAPY

O Scrapy é um framework escrito em Python que permite a criação de crawlers e scrapers para navegar e raspar grandes conjuntos de dados a partir da web ou de outras fontes (Scrapy, 2023). Ele pode ser usado em diversas aplicações como, por exemplo, mineração de dados, processamento de informação ou histórico de arquivos (Camargo et al, 2017).

O Scrapy possui uma vasta gama de recursos, bem como provê várias maneiras de se realizar uma mesma tarefa, no entanto, dois elementos são considerados como elementos chave no desenvolvimento desta pesquisa: as spiders e os seletores. Uma spider é representada por uma classe que define o comportamento personalizado de rastreamento e análise em páginas de um site específico ou de um conjunto deles, isto é, é uma classe que especifica regras de web crawling e web scraping (Scrapy, 2023).

Já os seletores são critérios aplicados dentro das spiders para escolher segmentos de um documento XML/HTML usando expressões XPath, CSS ou expressões regulares.

Além desses elementos, sua arquitetura também é baseada em vários outros componentes que trabalham em conjunto para realizar o processo de scraping:

- **Engine:** controla o fluxo de todo o processo de scraping. É responsável por receber as requisições dos spiders, escalonar as requisições para o download, processar as respostas recebidas, enviar as páginas para os

spiders analisarem e gerenciar as filas de requisições e itens extraídos. A Engine coordena todas as partes do Scrapy e garante que o scraping seja executado de forma assíncrona e eficiente.

- **Scheduler:** é o componente responsável por gerenciar as requisições agendadas. Ele recebe as requisições dos spiders por meio da Engine e as coloca em uma fila, garantindo que elas sejam processadas de forma ordenada. Além disso, é responsável por evitar requisições duplicadas e controlar o fluxo de requisições.
- **Downloader:** responsável pelo download das páginas web. Ele recebe as requisições do Scheduler, executa as solicitações HTTP correspondentes e retorna as respostas ao Engine. O Downloader também lida com questões relacionadas a cookies, cabeçalhos HTTP e proxies, se necessário.
- **Item Pipeline:** responsável por processar os itens extraídos pelos spiders. Ela permite que você defina um fluxo de processamento para os dados extraídos, como limpeza, validação e armazenamento em um banco de dados. A Item Pipeline recebe os itens extraídos dos spiders, aplica uma série de processadores a eles e, em seguida, decide o que fazer com os itens processados, como salvá-los em um banco de dados ou descartá-los.
- **Middlewares:** ficam entre o Engine e o Downloader, permitindo a customização do processo de scraping. Eles podem manipular as requisições e respostas, adicionando cabeçalhos personalizados, manipulando cookies, redirecionando solicitações, entre outras tarefas. Os Middlewares são úteis para adicionar funcionalidades globais ao scraping ou para lidar com problemas específicos em determinados websites.

4 TRABALHOS RELACIONADOS

Neste capítulo serão apresentados os trabalhos relacionados ou similares. Na Seção 3.1 são explicados os aplicativos relacionados ao trabalho desenvolvido. Na seção 3.2, são apresentadas propostas encontradas em trabalhos de conclusão de curso.

4.1 APLICATIVOS E PLATAFORMAS WEB

4.1.1 Superpagg

O Superpagg, é destinado para mobile e funciona através de Crowdsourcing, um processo colaborativo que utiliza da inteligência coletiva para obter seus resultados. A fonte de informação única é o portal de Nota Fiscal Cidadã, que só é disponibilizada com o cadastro e aprovação do usuário utilizador. Além disso, a ferramenta está limitada a somente alguns supermercados de Belém-PA e Ananindeua-PA. Como o usuário sincroniza sua conta manualmente, ele terá somente acesso aos seus hábitos de consumo passados, não sendo possível a comparação de produtos antes da compra (Arthur, et al, 2019).

4.1.2 Meus Preços

O aplicativo mobile Meus Preços possibilita a pesquisa de preços em supermercados de São Paulo utilizando as informações de compras dos cupons fiscais do sistema da Nota Fiscal Paulista. Normalmente os estabelecimentos demoram um dia para enviar os dados dos cupons fiscais para o sistema da Nota Fiscal Paulista. Porém, de acordo com o site oficial do aplicativo, os estabelecimentos têm até 60 dias para enviar estas informações. Com isso, os preços praticados pelos estabelecimentos podem ser alterados neste intervalo. Assim como o Superpagg, também só oferece uma análise descritiva dos custos.

4.1.3 ClickSuper

O aplicativo mobile recentemente lançado em Florianópolis, se propõe a deixar milhares de produtos atualizados diariamente nas principais redes de mercados, supermercados, atacados, farmácias e petshops da sua região,

comparados por preço diretamente no aplicativo. Porém, como sua atuação é ampla, a ferramenta não provê a grande quantidade de supermercados de Florianópolis, recomendando a compra de produtos em supermercados apresentando preços de produtos de supermercados de cidades vizinhas.

Figura 1 - Interface gráfica do ClickSuper

Fonte: o Autor (2023)

4.2 TRABALHOS DE CONCLUSÃO DE CURSO

4.2.1 QFEX

A proposta do trabalho de conclusão de curso proposta por Gilnei, intitulado “qFex: um crawler para busca e extração de questionários de pesquisa em documentos HTML” consiste em um Web Crawler para extrair questionários de uma lista de sites e salvá-los de forma estruturada, a fim de criar uma base para elaboração de novos questionários (MATHIAS, G. N., 2017).

Para varrer a web, o Crawler se baseia em uma lista de *seeds* e busca questionários utilizando de várias heurísticas criadas pelo autor e que trazem uma taxa de sucesso aceitável.

A solução pode ser dividida em duas: O Crawler, que deve varrer a Web em busca de questionários que possuam certas características e então deve salvar seus

links em um banco de dados e o Extractor, que por sua vez, deve apanhar os links adquiridos na etapa anterior e extrair os dados dos questionários para um outro banco de dados.

4.2.2 EXTRAÇÃO E COMBINAÇÃO POR SIMILARIDADE: UM ESTUDO DE CASO NAS REDES DE SUPERMERCADOS EM FLORIANÓPOLIS

A proposta do trabalho de conclusão de curso de Diogo conta com o desenvolvimento de uma API que age como um Web scraper/wrapper a fim de realizar a extração de preços e produtos de supermercados que disponibilizam serviço de vendas online na região de Florianópolis. Os supermercados escolhidos foram o Bistek, Imperatriz e Fort Atacadista, (LUIZ, 2022).

Após a extração, em seu trabalho fez-se a análise do grau de similaridade de texto após uma etapa de transformação e normalização, chamada de camada de pré-processamento. Essa camada pode ser dividida em duas etapas: integração e indexação, onde os dados extraídos são comparados através de algoritmos de similaridades a fim de combinar os produtos identificados como similares e salvar estas combinações.

O trabalho proposto teve o foco na comparação por similaridade entre produtos de supermercados diferentes, portanto, além de objetivo os trabalhos se diferem nas técnicas de extração, framework utilizado, quantidade de supermercados analisados e técnicas de verificação de confiabilidade de informações ao período de análise. Além disso, o autor realizou o trabalho de aquisição dos dados com o uso de APIs. Esta utilização entrega o dado de uma forma estruturada porém é altamente dependente da disponibilidade da API.

4.2.3 SISTEMA WEB PARA COMPARAÇÃO DOS PREÇOS DE SUPERMERCADOS ONLINE

O sistema web proposto por Carvalho, aproveita da disponibilidade da API REST do VTEX para aquisição padronizada das informações. O contexto de seu trabalho é da cidade de Belo Horizonte, Minas Gerais, e tem como objetivo a

construção de um sistema utilizando Angular, Node.js, PostgreSQL para dispor de uma API e um conjunto de páginas web para comparação de preços históricos.

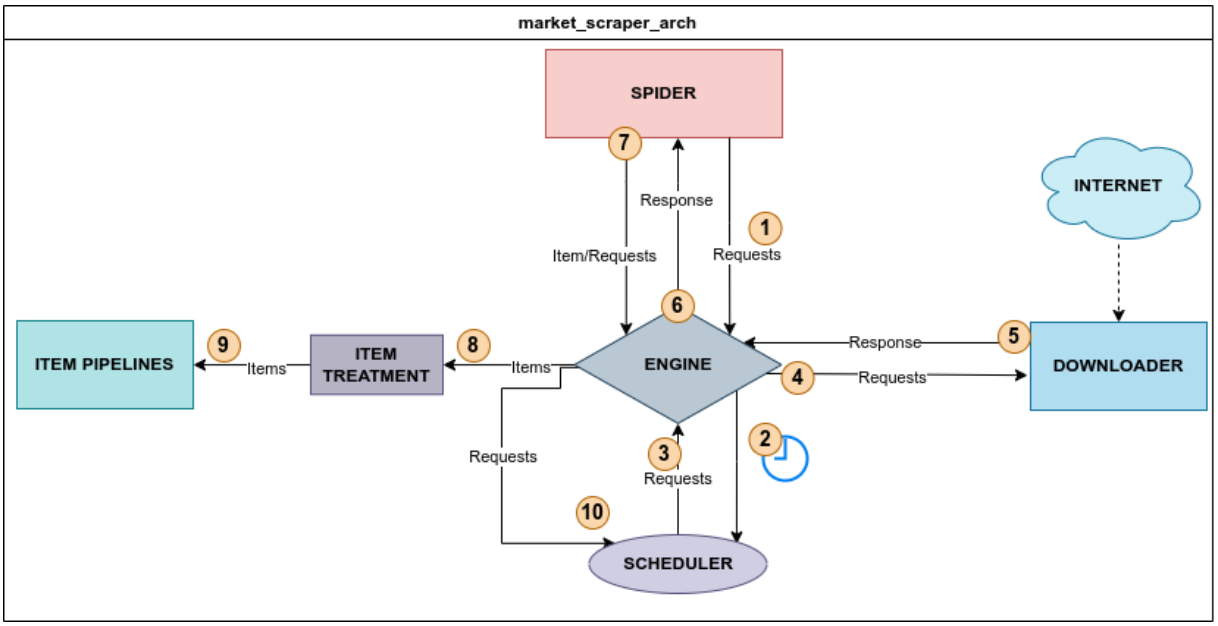
5 MARKET-SCRAPER

Este capítulo detalha o funcionamento dos componentes utilizados para extração, tratamento e sanitização inicial e armazenamento das dado utilizados.

5.1 VISÃO GERAL

Amplamente utilizado para extrair dados de websites de forma eficiente e estruturada, a arquitetura do framework Scrapy é baseada em um design modular e reutilizável que oferece uma estrutura organizada para o desenvolvimento de projetos de web scraping. Ele divide o processo de web scraping em diferentes componentes, como a lógica de extração, o processamento de itens e o armazenamento dos dados. Essa abordagem modular torna mais fácil a manutenção e a expansão do código, pois cada componente pode ser trabalhado de forma independente.

Figura 2 - Arquitetura geral do scraper



Fonte: o Autor (2023)

O fluxo de dados, baseada na arquitetura padrão do Scrapy, é controlado pelo componente ENGINE (mecanismo de execução) seguindo os seguintes passos:

1. O primeiro passo do ENGINE é obter as solicitações iniciais para iniciar o SPIDER. Nessa etapa são carregadas as informações das configurações definidas pelo componente.
2. O ENGINE começa a enfileirar as requisições no SCHEDULER, geralmente definidos dentro de cada SPIDER através do atributo padrão chamado start_urls. Após isso acontecer, o SCHEDULER fica esperando novas requisições para serem interpretadas.
3. O SCHEDULER retorna qual será a próxima requisição para o ENGINE principal.
4. O ENGINE então manda a requisição para o DOWNLOADER, para iniciar a aquisição das informações na página.
5. Uma vez que todas as informações da página foram adquiridas, o DOWLOADER envia para o ENGINE.
6. O ENGINE recebe a resposta do DOWNLOADER e envia para o SPIDER processar, passando por um componente chamado MIDDLEWARE, que é responsável por pós-processar o item/requisição, lidar com erros, etc.
7. O SPIDER processa a resposta e retorna os itens raspados e adiciona novos requests ao ENGINE.
8. O ENGINE envia os itens processados para o ITEM TREATMENT, para serem sanitizados e padronizados de acordo com a classe do item e seus atributos.
9. Depois que cada um do item ser sanitizado, ele é enviado para o ITEM PIPELINES, enviando então os requests processados para o SCHEDULER, requerendo possíveis novos requests para serem feitas.
10. O processo se repete, da etapa 3, até não haver mais requests enfileirados pelo SCHEDULER.

5.2 FERRAMENTAS E BIBLIOTECAS

Python: Os web scrapers foram desenvolvidos utilizando a linguagem python na versão 3.9.0.

Framework:

- **Scrapy**: versão 2.9.0, é um framework robusto para percorrer sites da web e extrair dados estruturados que podem ser usados para uma ampla variedade de aplicativos úteis, como mineração de dados,

processamento de informações ou arquivamento histórico. (KOUZIS-LOUKAS, 2016)

Bibliotecas:

- **xmltodict**: versão 0.13.0, é uma biblioteca python responsável por, dentre outras coisas, facilitar a interpretação de arquivos crus no formato XML e converte-los em um objeto dicionário manipulável.
- **psycopg2-binary:** versão 2.9.6, é uma extensão do psycopg2 que fornece uma implementação pré-compilada e estática do adaptador PostgreSQL para Python, permitindo a interação com bancos de dados PostgreSQL.
- **Twisted:** versão 22.10.0, é uma estrutura assíncrona de rede para Python, projetada para facilitar o desenvolvimento de aplicativos de rede robustos e de alto desempenho. Com suporte a diversos protocolos de rede e recursos avançados, como tolerância a falhas e criptografia, o Twisted oferece uma abordagem flexível e escalonável para a programação assíncrona, permitindo o desenvolvimento de servidores, clientes e outros componentes de rede eficientes.
- **azure-storage-blob:** versão 12.16.0, é usada para interagir com o serviço de armazenamento de blobs do Microsoft Azure. Ela permite que os desenvolvedores gerenciem facilmente operações como upload, download, exclusão e listagem de blobs, além de recursos avançados, como compartilhamento de acesso e gerenciamento de metadados, facilitando o armazenamento de objetos na nuvem.
- **requests:** versão 2.31.0, é usada para fazer requisições HTTP de maneira simples e eficiente. Ela fornece métodos convenientes para enviar solicitações HTTP, como GET, POST, PUT e DELETE, além de oferecer recursos avançados, como manipulação de autenticação, cookies e sessões. A biblioteca é amplamente utilizada para realizar comunicação com APIs da web, obter dados de servidores remotos e interagir com serviços baseados em HTTP.
- **pandas:** versão 1.5.1, é uma biblioteca python usada para análise e manipulação de dados de forma eficiente. Ela oferece estruturas de dados flexíveis, como DataFrames, que permitem a organização, limpeza e transformação de dados. Com recursos avançados para

agregação, filtragem e processamento de dados em grandes conjuntos de dados, o pandas é amplamente utilizado em ciência de dados e análise exploratória de dados.

5.3 FONTE DE DADOS

O projeto visa adquirir informações de produtos dos supermercados na região de Florianópolis. Para isso, fez-se um levantamento dos supermercados que possuíssem venda online de produtos, fossem comumente conhecido pela população e possuíssem páginas web funcionais. Acordou-se que os supermercados que contemplam tais requisitos são: Angeloni, Bistek, Fort Atacadista, Giassi e Imperatriz.

Cada site possui uma combinação de elementos únicos que formam a estrutura do seu corpo e também a disponibilização do seu conteúdo. O frontend dessa estrutura é composta basicamente por HTML, CSS e Javascript. (ROBBINS, 2012).

Para facilitar a diferenciação entre os diversos tipos de páginas, dividiu-se então em categorias. Para cada uma dessas categorias foi realizadas a raspagem de dados:

- **‘sitemap’**: Esse tipo de página tem o formado em XML e descreve a estrutura e organização de um site. De acordo com Ceci & Lanotte (2020), ele é projetado para ajudar os mecanismos de busca a indexarem e navegarem pelo conteúdo de um site de forma mais eficiente. O sitemap.xml fornece informações sobre as páginas do site e sua hierarquia. Isso permite que os mecanismos de busca rastreiem e indexem o conteúdo de maneira mais precisa, melhorando a visibilidade e a classificação do site nos resultados de pesquisa. Essa hierarquia dispõe alguns atributos como uma lista de endereços responsáveis por centralizar os departamentos e uma lista de endereços responsáveis por centralizar o catálogo de produtos de todo o site.
- **‘catalog’:** Esse tipo de página tem o formado em XML e proporciona um catálogo de produtos acessíveis pelo site incluindo qual o endereço de cada um deles, a frequência de atualização e a data da última modificação.
- **‘products’**: Esse tipo de página é o mais complexo dos três. Ele é responsável por centralizar a interface com o cliente comum, que navega na

página em busca de alimentar sua cesta de produtos e assim efetuar a compra. Nesse tipo de página é comum que haja uma lista de departamentos, cada um com seu tipo de produto, e também uma combinação entre paginação e filtros para facilitar a navegação. Atributos como nome do produto, preço, preço promocional, preço promocional do clube, departamento em que se encontra e url do produto podem e foram alvo desse tipo de scraping.

A construção dos scrapers foi feita com base na disponibilidade dessas informações e categorias.

Tabela 1 - Nomenclaturas adotadas para os scrapers

Nome do supermercado	**Nomenclatura do scraper - sitemap**	**Nomenclatura do scraper - catalog**	**Nomenclatura do scraper - products**
Angeloni	angeloni_sitemap_spider	angeloni_catalog_spider	angeloni_products_spider
Bistek	Não aplicável	Não aplicável	bistek_products_spider
Fort Atacadista	fort_sitemap_spider	fort_catalog_spider	fort_products_spider
Giassi	giassi_sitemap_spider	giassi_catalog_spider	giassi_products_spider
Imperatriz	imperatriz_sitemap_spider	imperatriz_catalog_spider	imperatriz_products_spider

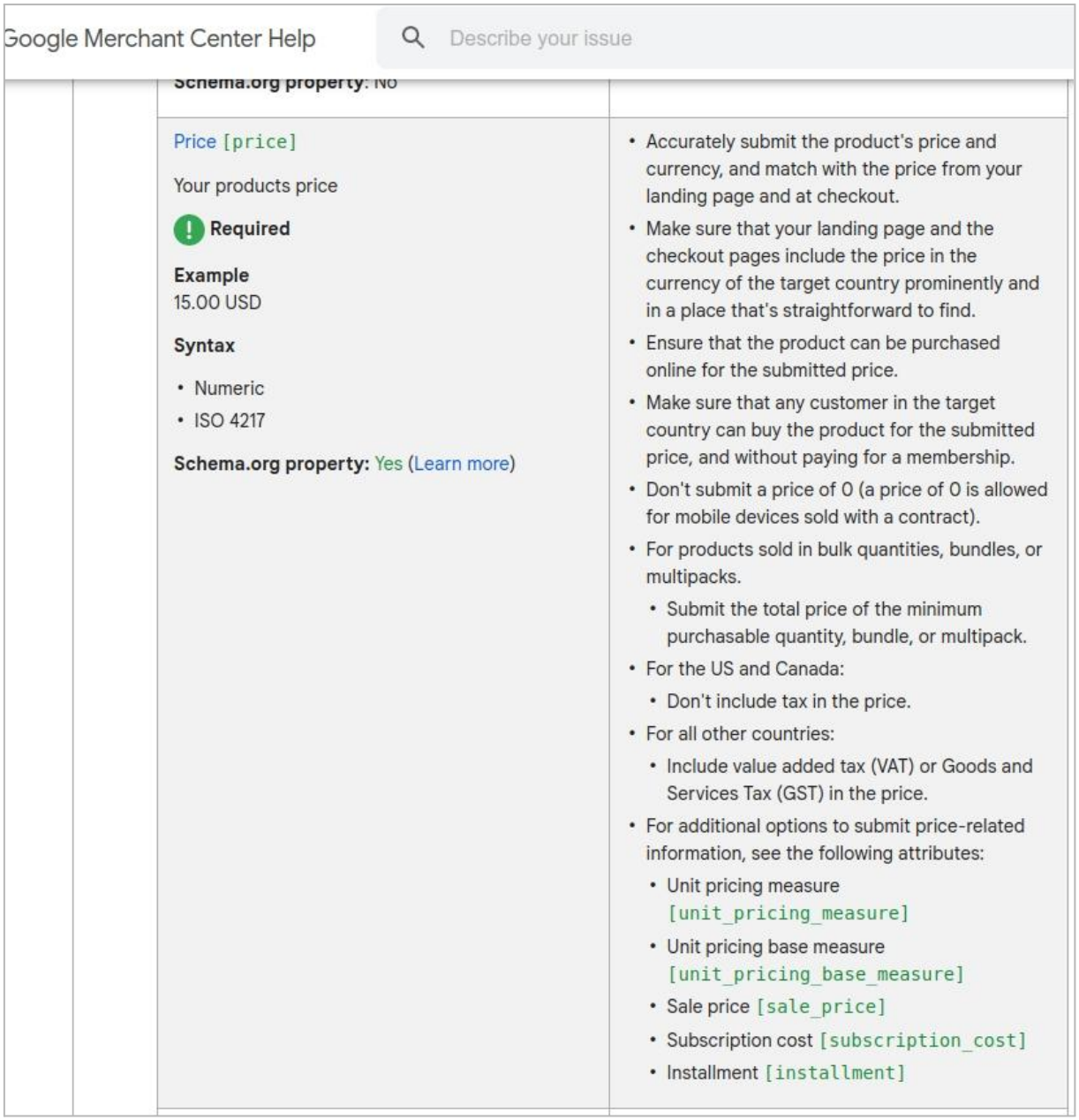
Fonte: o Autor (2023)

Como pode-se observar, fez-se um total de 13 scrapers, sendo que 4 são do tipo ‘sitemap’, 4 do tipo ‘catalog’ e 5 do tipo ‘products’.

5.4 PADRONIZAÇÃO DAS INFORMAÇÕES E CONVENÇÕES

Ao analisar as páginas alvo dos scrapers, percebe-se um conjunto de atributos que podem ser adquiridos. Estes atributos foram baseados no catálogo de especificações de produtos do Google (GOOGLE, 2022). Seu principal objetivo é a padronização de atributos para indexação mas também pode ser usado somente como convenção de nomenclatura de atributos.

Figura 3 - Exemplo de especificação de atributo (price)



Fonte: GOOGLE (2022)

Ao padronizar as informações, as empresas garantem que seus produtos correspondam às consultas de pesquisa dos usuários, resultando em anúncios e listagens gratuitas de alto desempenho. Além disso, a padronização dos dados melhora a experiência do usuário, fornecendo informações claras que facilitam a comparação e a tomada de decisões.

Esse conjunto de atributos comporão o objeto Item, do scrapy. Os Items são objetos utilizados para representar os dados extraídos de uma página web. Eles

permitem definir a estrutura dos dados que você deseja extrair. Cada Item é uma instância de uma classe que herda da classe scrapy.Item, e seus campos são definidos como atributos da classe. Por exemplo, se você estiver raspando informações de produtos em um site de comércio eletrônico, pode-se criar um Item chamado ProductItem com campos como title, price, description, etc.

5.4.1 PÁGINAS DO TIPO SITEMAP

Tabela 2 - Mapeamento de atributos para itens do tipo sitemap

Nome do atributo	Descrição do atributo	Tipo do dado
market_name	Nome do spider responsável pela aquisição da informação	String
initial_page	Endereço da página inicial do supermercado.	String
robots_page	Endereço da página responsável pelo conteúdo robots.txt do supermercado.	String
sitemap_page	Endereço da página responsável pelo conteúdo sitemap.xml do supermercado.	String
departments_page	Lista de endereços responsáveis por armazenar os endereços dos departamentos do supermercado.	List
products_pages	Lista de endereços responsáveis por armazenar os endereços dos produtos do supermercado.	List
brands_page	Lista de endereços responsáveis por armazenar os endereços das marcas que o supermercado filtrou.	String
date_extracted	Data e hora que o produto foi extraído no formato DD/MM/YYYY HH:MM:SS.	String
links	Lista de endereços encontrados no sitemap.	List

Fonte: o Autor (2023)

5.4.2 PÁGINAS DO TIPO CATALOG

Tabela 3 - Mapeamento de atributos para itens do tipo catalog

Nome do atributo	Descrição do atributo	Tipo do dado
market_name	Nome do spider responsável pela aquisição da informação	String
loc	Endereço para a página oficial do produto	String
changefreq	Assume-se que a periodicidade que o produto é atualizado.	String
lastmod	Assume-se que a última data o produto foi atualizado no formato YYYY-MM–DD.	String
date_extracted	Data e hora que o produto foi extraído no formato DD/MM/YYYY HH:MM:SS.	String

Fonte: o Autor (2023)

5.4.3 PÁGINAS DO TIPO PRODUCTS

Tabela 4 - Mapeamento de atributos para itens do tipo products

Nome do atributo	Descrição do atributo	Tipo do dado
market_name	Nome do spider responsável pela aquisição da informação	String
name	Nome do produto	String
url	Endereço da página onde o produto foi encontrado	String
product_url	Endereço para a página oficial do produto	String
price	Preço do produto	Float
sale_price	Preço promocional do produto	Float
sale_price_club	Preço promocional do produto para usuários pertencentes a um clube de vantagens	Float
department	Departamento de onde o produto foi extraído	String
category	Categoria do produto	String
images_url	Endereço da imagem de prateleira	String
date_extracted	Data e hora que o produto foi extraído no formato DD/MM/YYYY HH:MM:SS	String
unit_pricing_base_measure	Referência auxiliar para precificação.	String

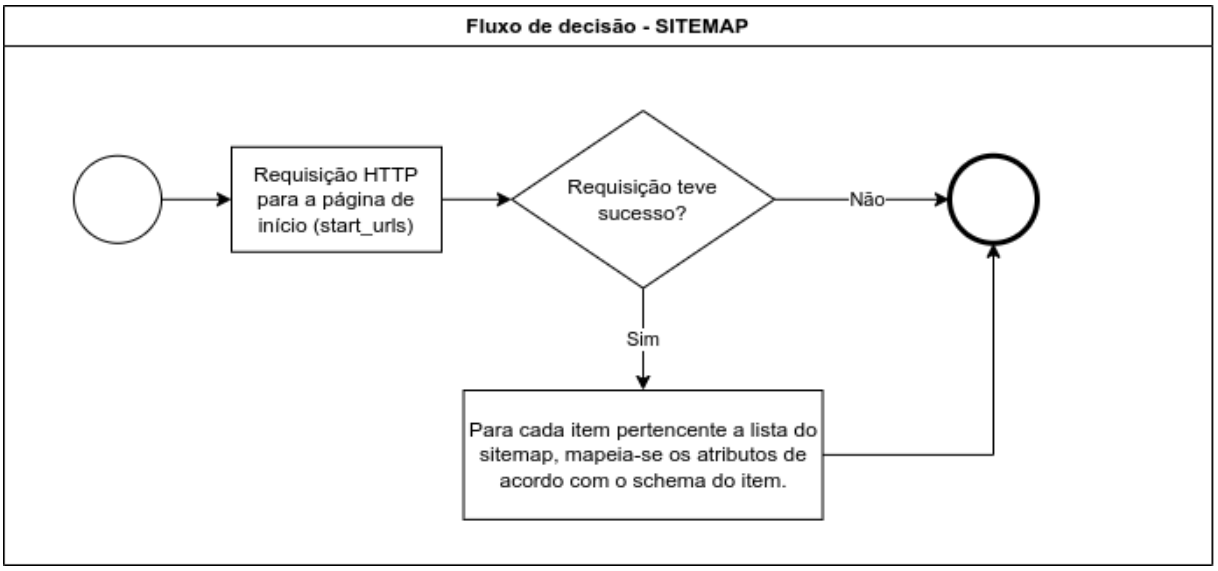
Fonte: o Autor (2023)

5.5 EXTRAÇÃO DOS DADOS

5.5.1 ANÁLISE DE SITEMAP

Os sitemaps disponíveis online e que foram analisados são os dos supermercados Angeloni, Fort Atacadista, Giassi e Imperatriz. Todos os sites do tipo sitemap seguem uma estrutura parecida, diferenciando somente na nomenclatura e disponibilização dos atributos.

Figura 4 - Fluxo de decisão para sites do tipo sitemap



Fonte: o Autor (2023)

O código dos spiders do tipo segue a lógica do fluxo de decisão:

1. Inicia-se o fluxo fazendo uma requisição HTTP para o primeiro item da lista chamada start_urls, que geralmente só tem um item e indica qual a página responsável por centralizar as informações do sitemap.
   OBS: Os sites responsáveis pelos sitemaps dos supermercados Angeloni, Fort Atacadista, Giassi e Imperatriz são, respectivamente:
   - <https://www.angeloni.com.br/super/sitemap.xml>;
   - <https://www.deliveryfort.com.br/sitemap.xml>;
   - <https://www.giassi.com.br/sitemap.xml>;
   - <https://www.supermercadosimperatriz.com.br/sitemap.xml>;

2. Se a requisição não for concluida com sucesso, finaliza-se o processo e nenhuma ingestão é feita. Se há retorno da página, cria-se uma lista de links e itera-se sobre ela.
    a. Para cada item, mapeia-se os atributos de acordo com o schema respectivo do sitemap conforme mostrado abaixo.

Código 1 - Exemplo de response de https://www.deliveryfort.com.br/sitemap.xml

```
<sitemapindex xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
   <sitemap>
      <loc>https://www.deliveryfort.com.br/sitemap/brand-1.xml</loc>
      <lastmod>2023-06-13</lastmod>
   </sitemap>
   <sitemap>
      <loc>https://www.deliveryfort.com.br/sitemap/category-1.xml</loc>
      <lastmod>2023-06-13</lastmod>
   </sitemap>
   <sitemap>
      <loc>https://www.deliveryfort.com.br/sitemap/product-1.xml</loc>
      <lastmod>2023-06-13</lastmod>
   </sitemap>
   <sitemap>
      <loc>https://www.deliveryfort.com.br/sitemap/product-2.xml</loc>
      <lastmod>2023-06-13</lastmod>
   </sitemap>
   <sitemap>
      <loc>https://www.deliveryfort.com.br/sitemap/product-3.xml</loc>
      <lastmod>2023-06-13</lastmod>
   </sitemap>
   <sitemap>
      <loc>https://www.deliveryfort.com.br/sitemap/product-4.xml</loc>
      <lastmod>2023-06-13</lastmod>
   </sitemap>
   <sitemap>
      <loc>https://www.deliveryfort.com.br/sitemap/product-5.xml</loc>
      <lastmod>2023-06-13</lastmod>
   </sitemap>
</sitemapindex>
```

Fonte: o Autor (2023)

O response adquirido via requisição HTTP vem no formato XML, e é mapeado para seguir a estrutura do item do tipo sitemap.

Código 2 - Exemplo de item extraído do spider ‘fort_sitemap_spider’

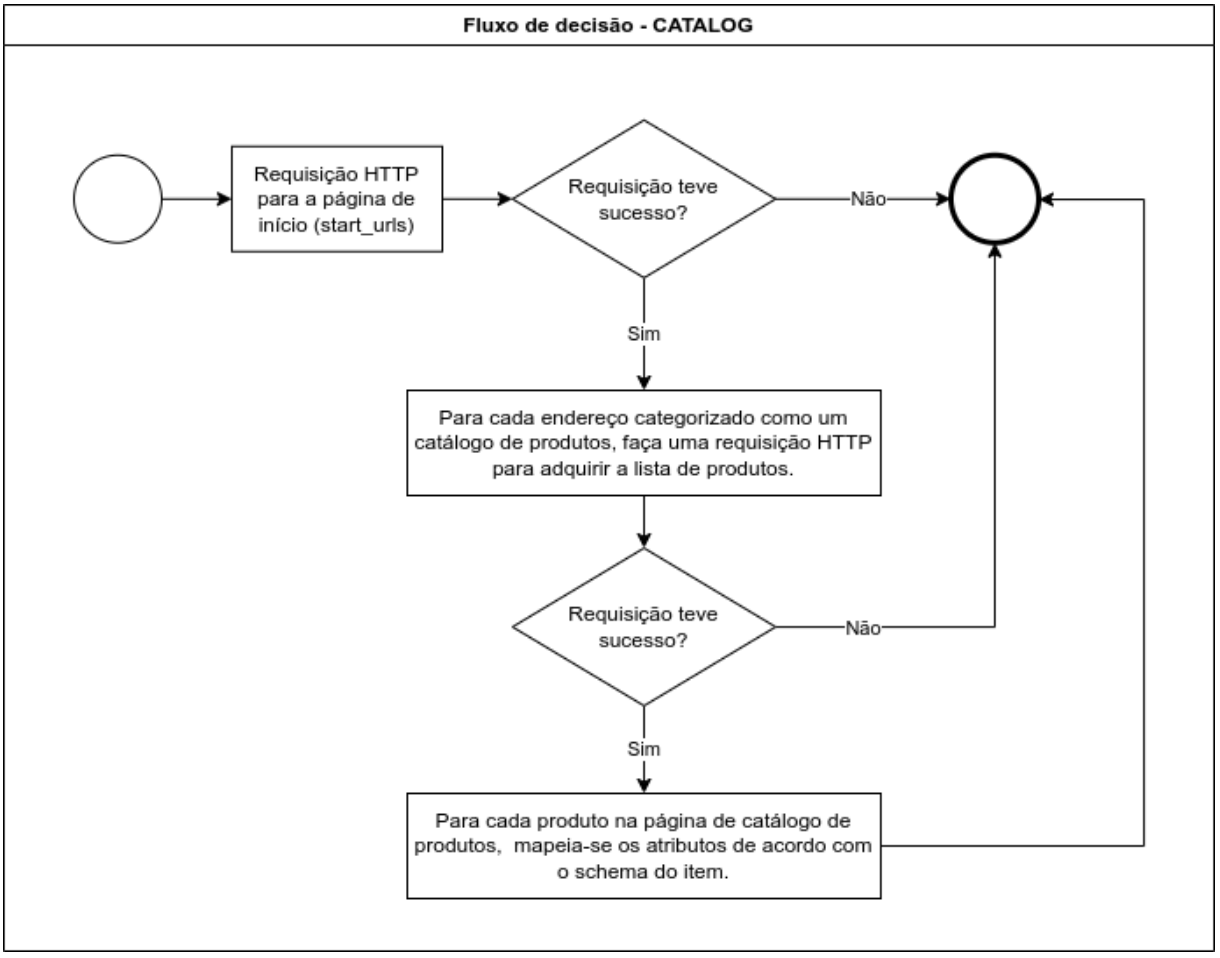

```
{
   "market_name": "fort_sitemap_spider",
   "initial_page": "https://www.deliveryfort.com.br/",
   "robots_page": "https://www.deliveryfort.com.br/robots.txt",
   "sitemap_page": "https://www.deliveryfort.com.br/sitemap.xml",
   "departments_page": [
      "https://www.deliveryfort.com.br/sitemap/category-1.xml"
   ],
   "products_pages": [
      "https://www.deliveryfort.com.br/sitemap/product-1.xml",
      "https://www.deliveryfort.com.br/sitemap/product-2.xml",
      "https://www.deliveryfort.com.br/sitemap/product-3.xml",
      "https://www.deliveryfort.com.br/sitemap/product-4.xml",
      "https://www.deliveryfort.com.br/sitemap/product-5.xml"
   ],
   "brands_page": [
      "https://www.deliveryfort.com.br/sitemap/brand-1.xml"
   ],
   "date_extracted": "13/06/2023 16:39:02",
   "links": [
      "https://www.deliveryfort.com.br/sitemap/brand-1.xml",
      "https://www.deliveryfort.com.br/sitemap/category-1.xml",
      "https://www.deliveryfort.com.br/sitemap/product-1.xml",
      "https://www.deliveryfort.com.br/sitemap/product-2.xml",
      "https://www.deliveryfort.com.br/sitemap/product-3.xml",
      "https://www.deliveryfort.com.br/sitemap/product-4.xml",
      "https://www.deliveryfort.com.br/sitemap/product-5.xml"
   ]
}
```


Fonte: o Autor (2023)

5.5.2 ANÁLISE DE CATALOG

Os catalogs disponíveis online e que foram analisados são os dos supermercados Angeloni, Fort Atacadista, Giassi e Imperatriz. Todos os sites do tipo catalog seguem uma estrutura parecida, diferenciando somente na nomenclatura e disponibilização dos atributos.

Figura 5 - Fluxo de decisão para sites do tipo catalog



Fonte: o Autor (2023)

O código dos spiders do tipo segue a lógica do fluxo de decisão:

1. Inicia-se o fluxo fazendo uma requisição HTTP para o primeiro item da lista chamada start_urls, que geralmente só tem um item e indica qual a página responsável por centralizar as informações do sitemap.
   OBS: Os sites responsáveis pelos sitemaps dos supermercados Angeloni, Fort Atacadista, Giassi e Imperatriz são, respectivamente:

- <https://www.angeloni.com.br/super/sitemap.xml>;
- <https://www.deliveryfort.com.br/sitemap.xml>;
- <https://www.giassi.com.br/sitemap.xml>;
- <https://www.supermercadosimperatriz.com.br/sitemap.xml>;

2. Se a requisição não for concluida com sucesso, finaliza-se o processo e nenhuma ingestão é feita. Se há retorno da página do sitemap, faz-se uma requisição para cada página responsável por centralizar o catálogo de produtos, também chamado como ‘products_pages’.
3. Se a requisição não for concluida com sucesso, finaliza-se o processo e nenhuma ingestão é feita. Se há retorno da página, itera-se sobre ela cada produto e mapeia-se os atributos de acordo com o schema respectivo do catalog conforme mostrado abaixo.

A resposta das páginas de catálogo são longas e verbosas, portanto mostra-se abaixo somente um pequeno pedaço desse resultado.

Código 3 - Exemplo de response de https://www.deliveryfort.com.br/sitemap/product-1.xml

```
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
<url>

<loc>https://www.deliveryfort.com.br/papel-aluminio-wyda-pratic-natural-30cmx4m/p</loc>
   <lastmod>2023-06-13</lastmod>
   <changefreq>weekly</changefreq>
</url>
<url>
<loc>https://www.deliveryfort.com.br/azeite-de-oliva-borges-extra-virgem-vidro-500ml/p</loc>
   <lastmod>2023-06-13</lastmod>
   <changefreq>weekly</changefreq>
</url>
<url>
   <loc>https://www.deliveryfort.com.br/biscoito-wafer-mybit-morango-100g/p</loc>
   <lastmod>2023-06-13</lastmod>
   <changefreq>weekly</changefreq>
</url>
...
```

Fonte: o Autor (2023)

O response adquirido via requisição HTTP vem no formato xml e é mapeado para seguir a estrutura do item do tipo catalog.

Código 4 - Exemplo de item extraído do spider ‘fort_catalog_spider’


```
{
   "market_name": "fort_catalog_spider",
   "loc": "https://www.deliveryfort.com.br/cafe-do-ponto-expresso-almofada-1kg/p",
   "changefreq": "weekly",
   "lastmod": "2023-06-13",
   "date_extracted": "13/06/2023 17:44:33"
}
```


Fonte: o Autor (2023)

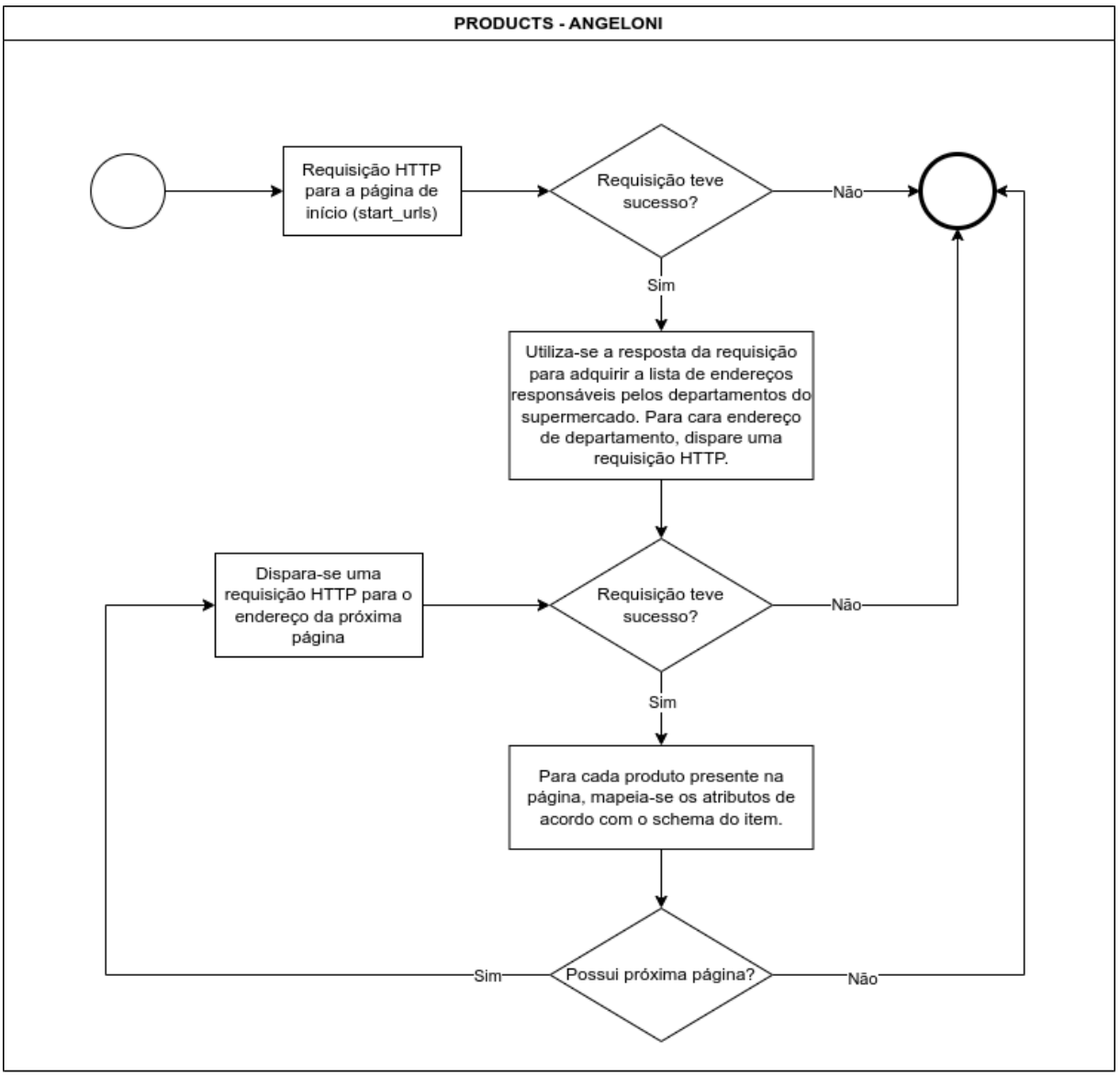
5.5.3 ANÁLISE DE PRODUCTS

Os sites de produtos disponíveis online e que foram analisados são os dos supermercados Angeloni, Fort Atacadista, Giassi e Imperatriz.

5.5.3.1 ANGELONI

O spider angeloni_products_spider responsável por raspar dados dos produtos do supermercado angeloni possui como start_url a página inicial do supermercado: <https://www.angeloni.com.br/super>.

Figura 6 - Fluxo de decisão para angeloni_products_spider



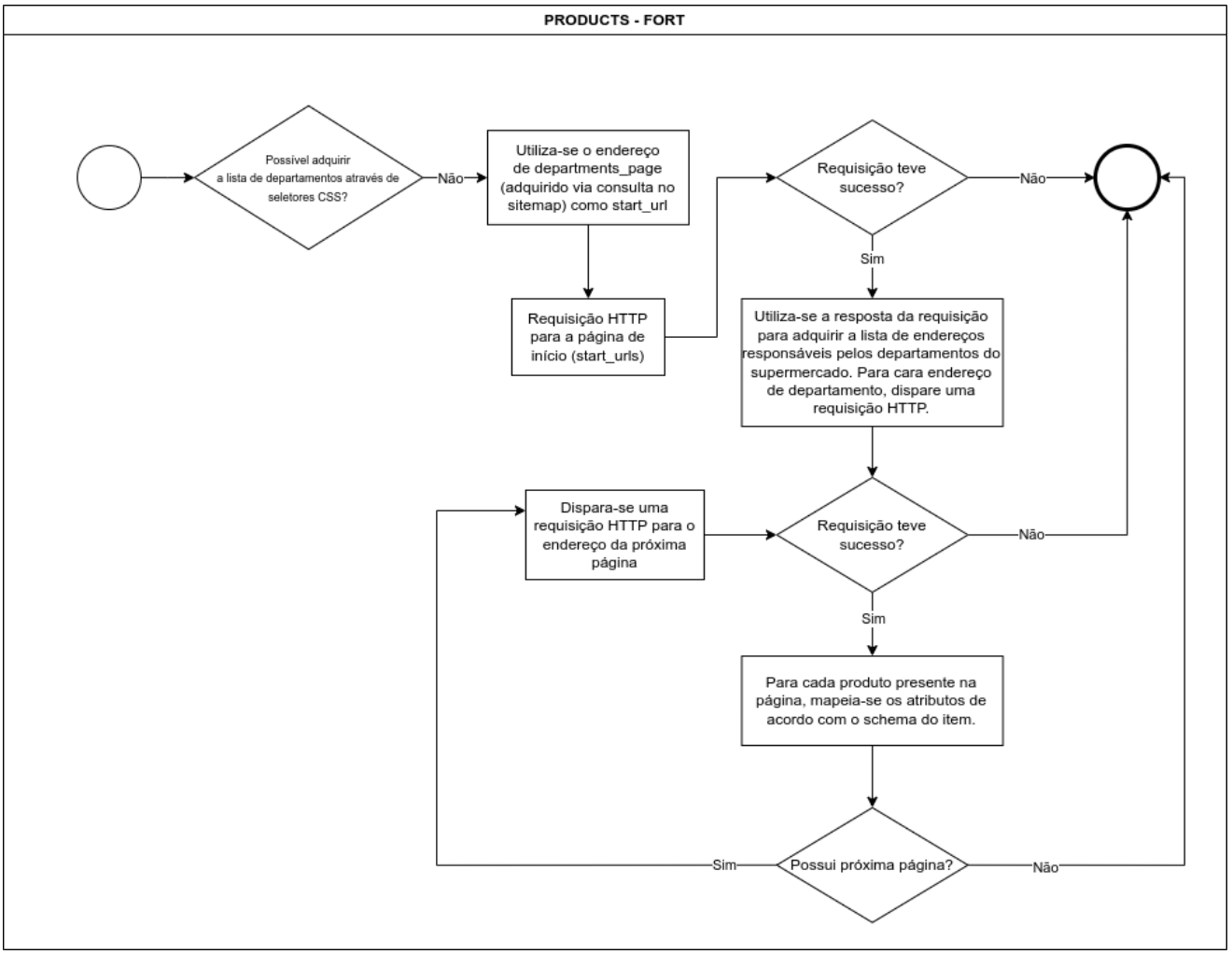
Fonte: o Autor (2023)

1. Faz-se uma requisição para o endereço presente em start_url;
2. Verifica-se o sucesso do retorno da requisição. Se teve sucesso o próximo passo é utilizar os seletores CSS para selecionar todos os endereços de departamentos e atribui-los a uma lista temporária. Se não, retorna sem output.
3. Para cada endereço presente na lista, dispara-se uma requisição HTTP.
4. Verifica-se o sucesso do retorno da requisição. Se teve sucesso, para cada produto presente na página, mapeia-se os atributos do item tipo produtos.
5. Se houver próxima página, faz-se uma requisição para o endereço da próxima página e repete-se o passo 4. Se não, finaliza.

5.5.3.2 FORT ATACADISTA E IMPERATRIZ

Os spiders fort_products_spider e imperatriz_products_spider possuem uma lógica parecida, partindo inicialmente da start_url <https://www.deliveryfort.com.br/sitemap/category-1.xml> e <https://www.supermercadosimperatriz.com.br/sitemap/category-1.xml> respectivamente.

Figura 7 - Fluxo de decisão para fort_products_spider e imperatriz_products_spider



Fonte: o Autor (2023)

1. Como não é possível adquirir a informação dos departamentos via seletor CSS sem renderização de javascript na página inicial, optou-se por utilizar como start_url o endereço que concentra os links dos departamentos, disponível no sitemap.
2. Faz-se uma requisição para o endereço presente em start_url;
3. Verifica-se o sucesso do retorno da requisição. Se teve sucesso o próximo passo é utilizar os seletores CSS para selecionar todos os endereços de departamentos e atribui-los a uma lista temporária. Se não, retorna sem output.
4. Para cada endereço presente na lista, dispara-se uma requisição HTTP.
5. Verifica-se o sucesso do retorno da requisição. Se teve sucesso, para cada

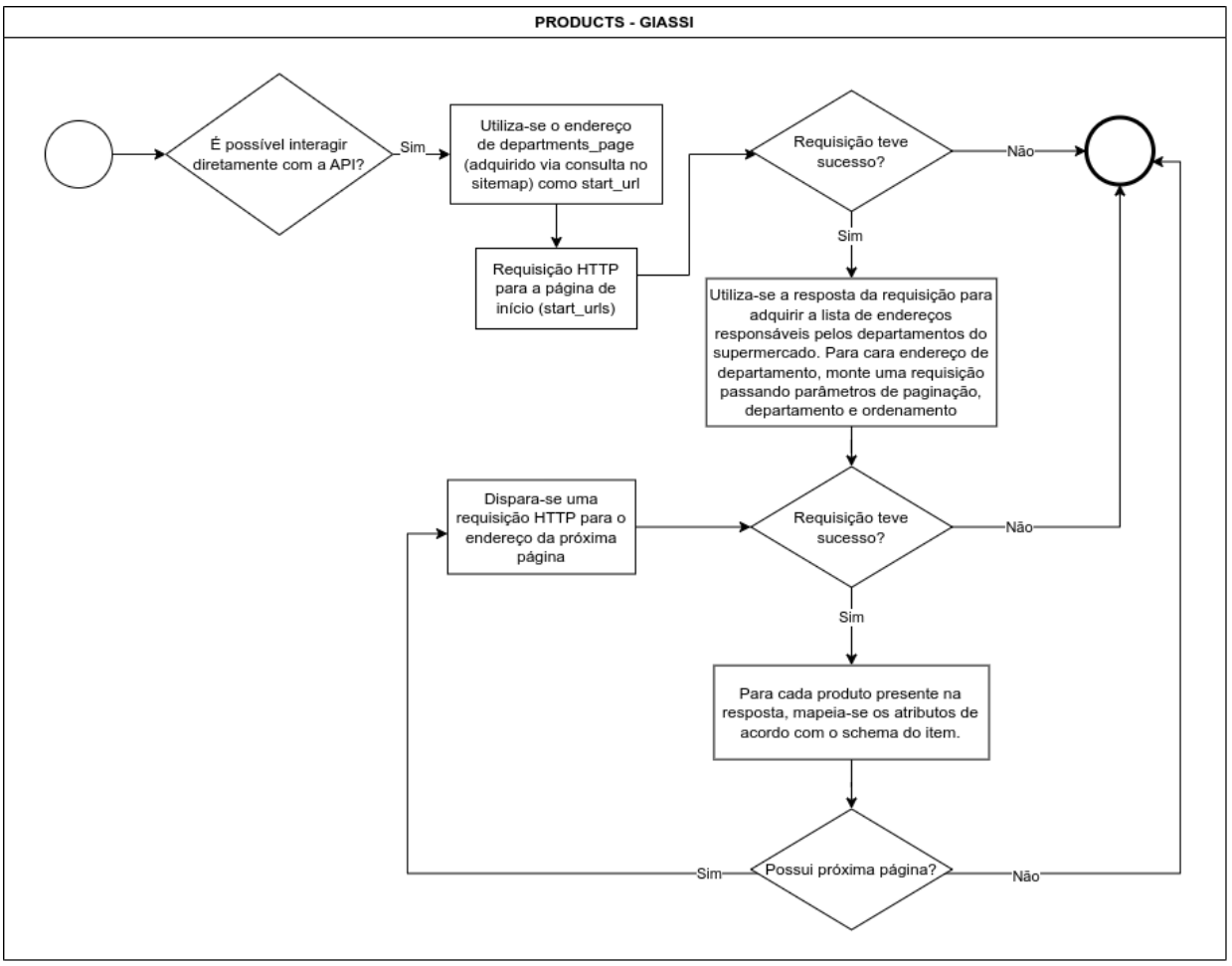
produto presente na página, mapeia-se os atributos do item tipo produtos.

6. Se houver próxima página, faz-se uma requisição para o endereço da próxima página e repete-se o passo 4. Se não, finaliza.

5.5.3.3 GIASSI

O spider giassi_products_spider responsável por raspar dados dos produtos do supermercado Giassi possui como start_url a página inicial do supermercado: <https://www.angeloni.com.br/super>.

Figura 8 - Fluxo de decisão para giassi_products_spider



Fonte: o Autor (2023)

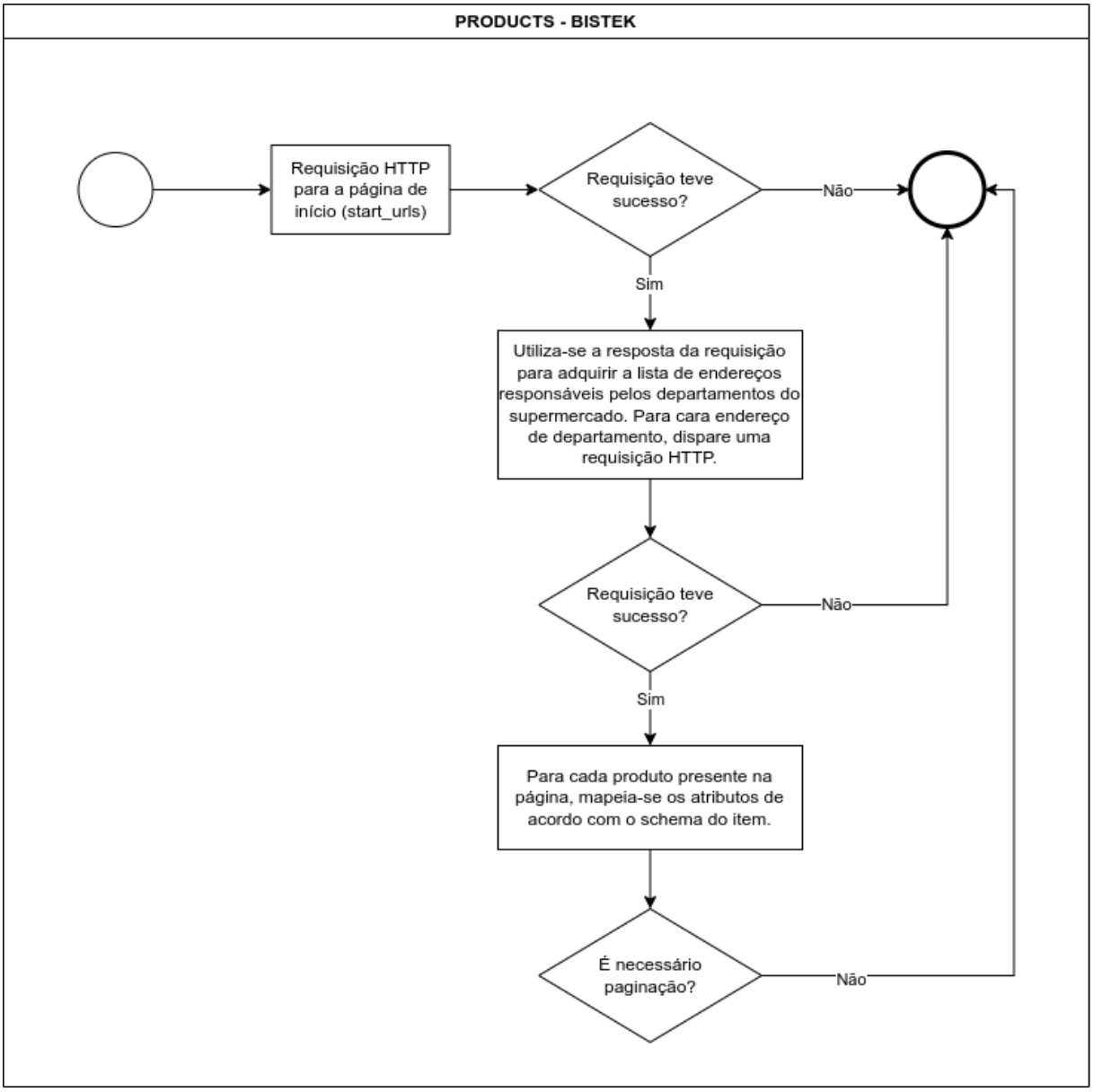
1. Verifica-se se é possível interagir diretamente com a API. Se sim, utiliza-se o endereço de departments_page como start_url;

2. Faz-se uma requisição para o endereço presente em start_url;
3. Verifica-se o sucesso do retorno da requisição. Se teve sucesso o próximo passo é utilizar os seletores CSS para selecionar todos os endereços de departamentos e atribui-los a uma lista temporária. Se não, retorna sem output.
4. Para cada endereço presente na lista, dispara-se uma requisição HTTP.
5. Verifica-se o sucesso do retorno da requisição. Se teve sucesso, para cada produto presente na página, mapeia-se os atributos do item tipo produtos.
6. Se houver próxima página, faz-se uma requisição para o endereço da próxima página e repete-se o passo 3. Se não, finaliza.

5.5.3.4 BISTEK

O spider bistek_products_spider responsável por raspar dados dos produtos do supermercado bistek possui como start_url a página inicial do supermercado: <https://www.bistek.com.br>.

Figura 9 - Fluxo de decisão para bistek_products_spider



Fonte: o Autor (2023)

1. Faz-se uma requisição para o endereço presente em start_url;
2. Verifica-se o sucesso do retorno da requisição. Se teve sucesso o próximo passo é utilizar os seletores CSS para selecionar todos os endereços de

departamentos e atribui-los a uma lista temporária. Se não, retorna sem output.

3. Para cada endereço presente na lista, dispara-se uma requisição HTTP.
4. Verifica-se o sucesso do retorno da requisição. Se teve sucesso, para cada produto presente na página, mapeia-se os atributos do item tipo produtos.
5. Como nunca haverá próxima página pois estamos utilizando o parâmetro “?product_list_limit=all”, que é responsável por retornar todos os produtos daquele departamento, finaliza-se a extração.

5.6 TRATAMENTO E SANITIZAÇÃO DO ITEM

A padronização de atributos possibilita que cada produto seja considerado um Item pelo Scrapy. Naturalmente, cada atributo do item logo após ser extraído, possui alguns ruídos no seu texto e precisa ser conformado. Desta forma, são feitas algumas transformações para limpeza desse texto tais como a substituição de caracteres indesejados.

Para isto, utilizou-se um componente fundamental no framework Scrapy chamado Item Loader. Ele desempenha um papel crucial ao permitir a definição de regras e transformações personalizadas aplicadas aos itens coletados de páginas web. O Item Loader possibilitou a estruturação e normalização dos dados extraídos, garantindo a consistência e a integridade dos mesmos. Além disso, o Item Loader oferece a flexibilidade de aplicar expressões regulares, filtragem de caracteres inválidos e conversões de formato, entre outras funcionalidades, tornando o tratamento e a sanitização de dados mais eficientes e precisos.

As principais funções das classes presentes no items.py, que implementa o Item Loader são:

- **remove_currency(value):** Esta função remove a moeda presente em um valor. Ela substitui a string "R$" por uma string vazia e, em seguida, divide o valor em uma lista utilizando "/" como delimitador e retorna o primeiro elemento da lista. Essa função é útil para extrair apenas o valor numérico de um preço, removendo a moeda.
- **remove_measure_aspects(value)**: Essa função remove aspectos de medida de um valor. Ela divide a string utilizando "/" como delimitador e retorna o primeiro elemento da lista resultante. Essa função é útil para extrair apenas a parte relevante de um valor que pode conter informações adicionais relacionadas a medidas.
- **remove_parenthesis(value)**: Essa função remove parênteses de uma string. Ela substitui os parênteses "(" e ")" por strings vazias e, em seguida, remove os espaços em branco no início e no final da string resultante. Essa função é útil para eliminar parênteses desnecessários em uma string.
- **remove_spaces(value)**: Esta função remove os espaços em branco no início e no final de uma string. Ela utiliza o método strip() para realizar essa

operação. Essa função é útil para eliminar espaços desnecessários ao redor de uma string.

- **remove_breaklines(value)**: Essa função remove quebras de linha (caracteres "\n") de uma string. Ela substitui esses caracteres por strings vazias. Essa função é útil para eliminar quebras de linha que podem estar presentes em uma string e interferir na sua formatação adequada.
- **to_float(value)**: Esta função converte uma string de valor em um valor de ponto flutuante (float). Ela utiliza a biblioteca price_parser para criar um objeto Price a partir da string de valor e, em seguida, retorna o atributo amount_float desse objeto. Essa função é útil para converter valores monetários representados como strings em números de ponto flutuante que podem ser usados em cálculos matemáticos.

Após o tratamento dos atributos do item, ele é enviado para o ITEM PIPELINES para então ser armazenado.

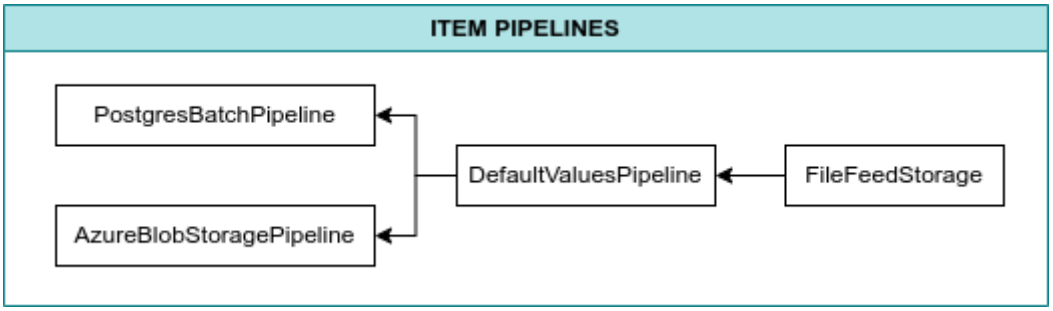
5.7 ARMAZENAMENTO

O armazenamento das informações fez-se através do componente chamado Item Pipelines, no Scrapy. É uma funcionalidade que desempenha um papel fundamental na etapa de pós-coleta, permitindo a aplicação de transformações, validações e filtragens nos itens extraídos, além de facilitar sua persistência em diferentes formatos e locais de armazenamento. Os benefícios do Item Pipelines são numerosos, uma vez que ele oferece uma abordagem modular e flexível para manipular os itens coletados, permitindo a execução de tarefas personalizadas de pré e pós-processamento.

Além disso, o uso de pipelines simplifica o código de extração, separando a lógica de raspagem da lógica de armazenamento, o que resulta em um código mais limpo e de fácil manutenção. Com a capacidade de direcionar os itens para bancos de dados, sistemas de arquivos, serviços de nuvem ou qualquer outra forma de armazenamento desejada, o Item Pipelines garante a eficiência e a confiabilidade do armazenamento de dados, tornando-o uma ferramenta indispensável para quem busca extrair e gerenciar informações de forma estruturada e automatizada.

No projeto utilizou-se de alguns pipelines para manipular e persistir os itens:

Figura 10 - Conjunto principal de Item Pipelines



Fonte: o Autor (2023)

- **FileFeedStorage:** É uma classe responsável por armazenar os dados coletados durante o processo de raspagem em arquivos locais no sistema de arquivos do computador. Definiu-se no arquivo de configurações que os dados extraídos seriam armazenados em um diretório no formato JSON e o log de extração com extensão .log.
- **DefaultValuesPipeline:** Este pipeline define valores padrão para os campos do item caso eles não estejam presentes. Ele percorre cada item e, se algum campo estiver ausente, atribui um valor nulo a ele. Isso garante que todos os campos tenham algum valor, mesmo que não tenham sido extraídos durante o processo de scraping.
- **PostgresBatchPipeline:** Este pipeline lida com a persistência dos itens em um banco de dados PostgreSQL. Ele estabelece uma conexão com o banco de dados com base nas configurações fornecidas através das variáveis de ambiente do projeto e insere os itens na tabela especificada. A diferença importante deste pipeline é que ele insere os itens em lote (batch), em vez de inserir um item de cada vez. Isso melhora a eficiência, pois reduz o número de transações de banco de dados necessárias.
- **AzureBlobStoragePipeline**: Este pipeline armazena os itens em um serviço de armazenamento do Azure Blob Storage. Ele estabelece uma conexão com a conta do Azure Blob Storage usando as configurações fornecidas através das variáveis de ambiente do projeto e envia os itens para o contêiner especificado. Ele é usado para fazer o upload dos dados coletados para o armazenamento em nuvem do Azure.

Além desses pipelines principais, também implementou-se 3 pipelines alternativos:

- **PostgresWithValidationPipeline:** Este pipeline lida com a persistência dos itens em um banco de dados PostgreSQL. Ele estabelece uma conexão com o banco de dados utilizando as configurações fornecidas no arquivo de configuração do Scrapy. Antes de inserir um item, ele verifica se o item já existe no banco de dados com base no nome, data de extração, preço e preço de venda. Se o item já existir, uma mensagem de aviso será registrada. Caso contrário, o item será inserido na tabela do banco de dados.
- **PostgresWithoutValidationPipeline:** Similar ao pipeline anterior, este pipeline também lida com a persistência dos itens em um banco de dados PostgreSQL. No entanto, ele não realiza a validação de duplicidade antes da inserção. O item é simplesmente inserido na tabela do banco de dados.
- **PostgresDelayedPipeline:** Este pipeline armazena os itens em uma lista durante o processo de raspagem e insere todos os itens no banco de dados PostgreSQL no final do processo. Ele estabelece uma conexão com o banco de dados utilizando as configurações fornecidas no arquivo de configuração do Scrapy. No final do processo de raspagem, todos os itens da lista são inseridos na tabela do banco de dados. Em comparação com o PostgresBatchPipeline, para um grande número de itens, haverá um alto volume de consultas individuais ao banco de dados, o que pode resultar em um desempenho inferior.

5.8 VALIDAÇÃO DA FERRAMENTA

A construção inicial dos scrapers se teve no dia 01 de fevereiro de 2022 onde o primeiro scraper de produtos foi projetado para adquirir as informações do site do Bistek Supermercados. Desde então, tem-se feito requisições geralmente diárias para que pudessemos utilizar da maior base de dados e estratégias de scraping.

Cada um dos spiders podem ser validados através da análise da contagem diária de produtos raspados. Pensando nisso, construiu-se a tabela abaixo que consiste de uma análise de todo o período: dia 01 de fevereiro de 2022 até o dia 14 de junho de 2023. Tem-se então qual a data mínima, e provável primeira execução de cada um dos scrapers, a data máxima e também a contagem de dias únicos extraídos. Este último é importante pois possibilita o grau de aproveitamento de cada um dos spiders, contemplando todas as dificuldades encontradas durante o caminho.

Tabela 5 - Relação de início, fim e dias extraídos por cada spider.

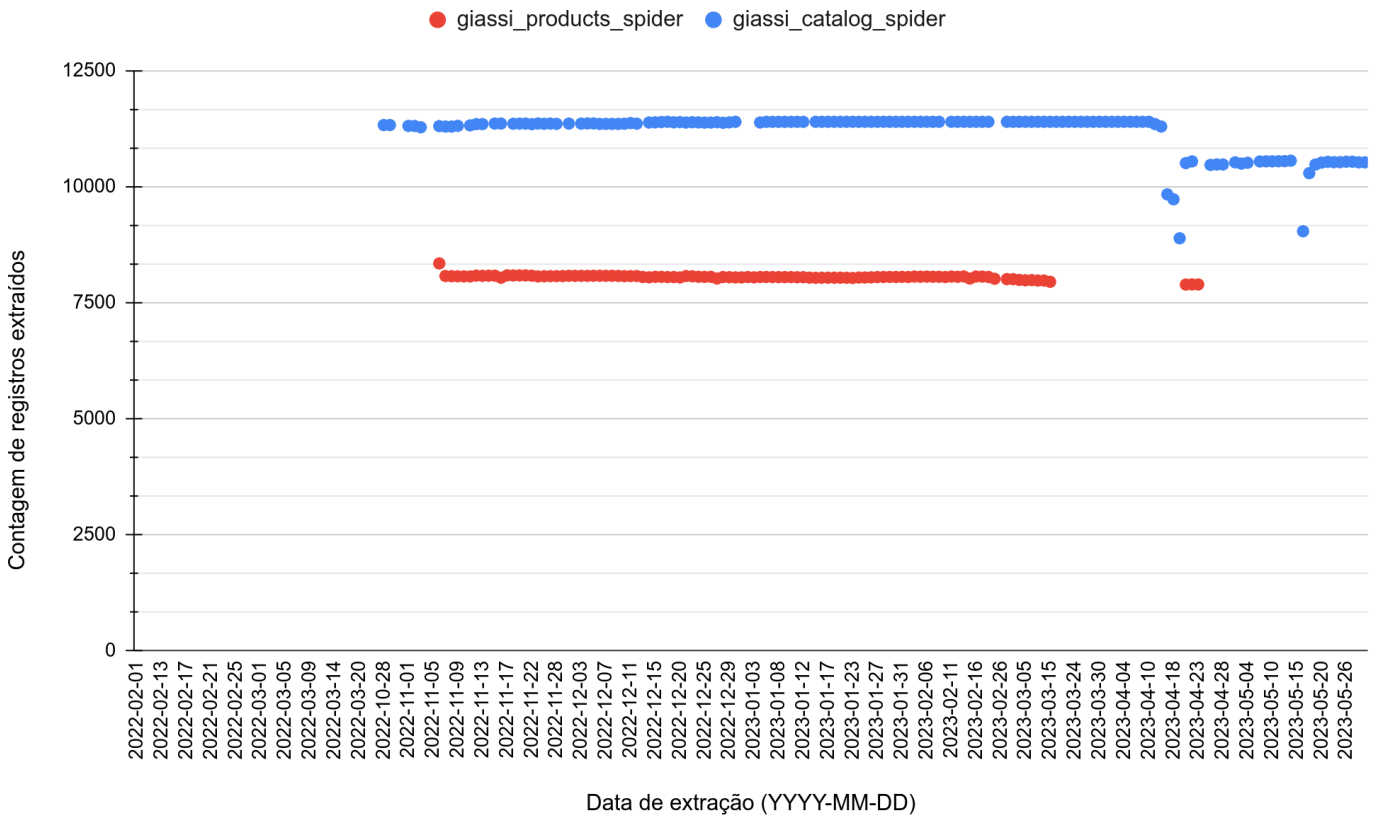
Nome do spider	**Data mínima de extração**	**Data máxima de extração**	**Contagem de dias únicos extraídos**
angeloni_catalog_spider	2022-10-28	2023-06-14	83
fort_catalog_spider	2022-10-28	2023-06-14	86
big_products_spider	2022-11-06	2023-03-15	99
giassi_products_spider	2022-11-06	2023-04-23	102
giassi_catalog_spider	2022-10-28	2023-06-14	140
imperatriz_products_spider	2023-06-13	2023-06-14	150
imperatriz_catalog_spider	2022-10-28	2023-06-14	153
fort_sitemap_spider	2022-10-28	2023-06-14	159
giassi_sitemap_spider	2022-10-28	2023-06-14	159
imperatriz_sitemap_spider	2022-10-28	2023-06-14	159
angeloni_sitemap_spider	2022-10-28	2023-06-14	160
fort_products_spider	2022-02-20	2023-06-14	182
angeloni_products_spider	2022-02-13	2023-06-14	189
bistek_products_spider	2022-02-01	2023-06-14	194

Fonte: o Autor (2023)

Percebe-se que a descoberta do endereço responsável pelo catálogo dos supermercados impactou diretamente na quantidade de dias adquiridos pois quase todos os spiders do tipo catálogo se encontram nos primeiros lugares em relação à menos quantidade de dias com extração.

Para melhor entendimento das fases de cada um dos spiders responsáveis por raspar as informações dos produtos, construiu-se um gráfico que coloca lado a lado a quantidade de produtos únicos esperados, adquiridos através do catálogo de produtos, e a quantidade de itens adquiridos através do spider do tipo products conforme pode-se verificar na Figura 11.

Figura 11 - Quantidade de registros extraídos por dia no Giassi x quantidade esperada



Fonte: o Autor (2023)

Pode-se perceber que há sempre mais itens admitidos pelo catálogo de produtos do que aqueles alcançados pelo scraping. Além disso, também percebe-se que recentemente o spider de produtos teve seu funcionamento cessado por um período, o problema foi

concertado e, após alguns dias, o problema voltou novamente e o status desse spider é inativo no momento.

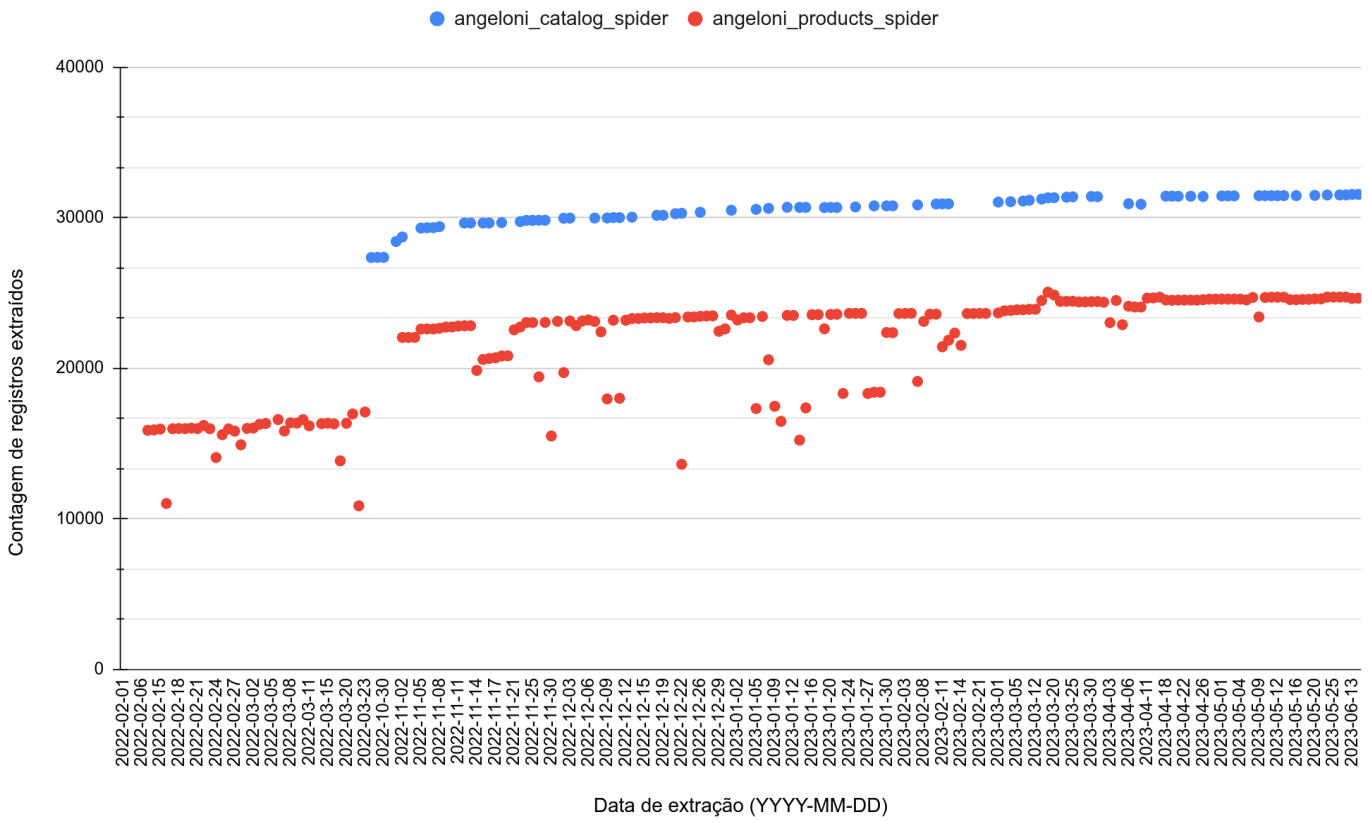
Quando analisa-se os produtos extraídos nos sites do Angeloni, mostrado na Figura 12, tem-se claramente 3 cenários:

Cenário 1: Desde o início da construção do scraper até o dia 20 de março de 2022 não havia extração do tipo catalog. Isto se deve pois, naquela época, não era sabido que ele existia e nem que era disponível para facilitar a navegação/SEO.

Cenário 2: Até o dia 05 de março de 2023 teve-se um período onde a contagem de produtos raspados não era constante.

Cenário 3: Após esta data até hoje, a aquisição está estável e confiável.

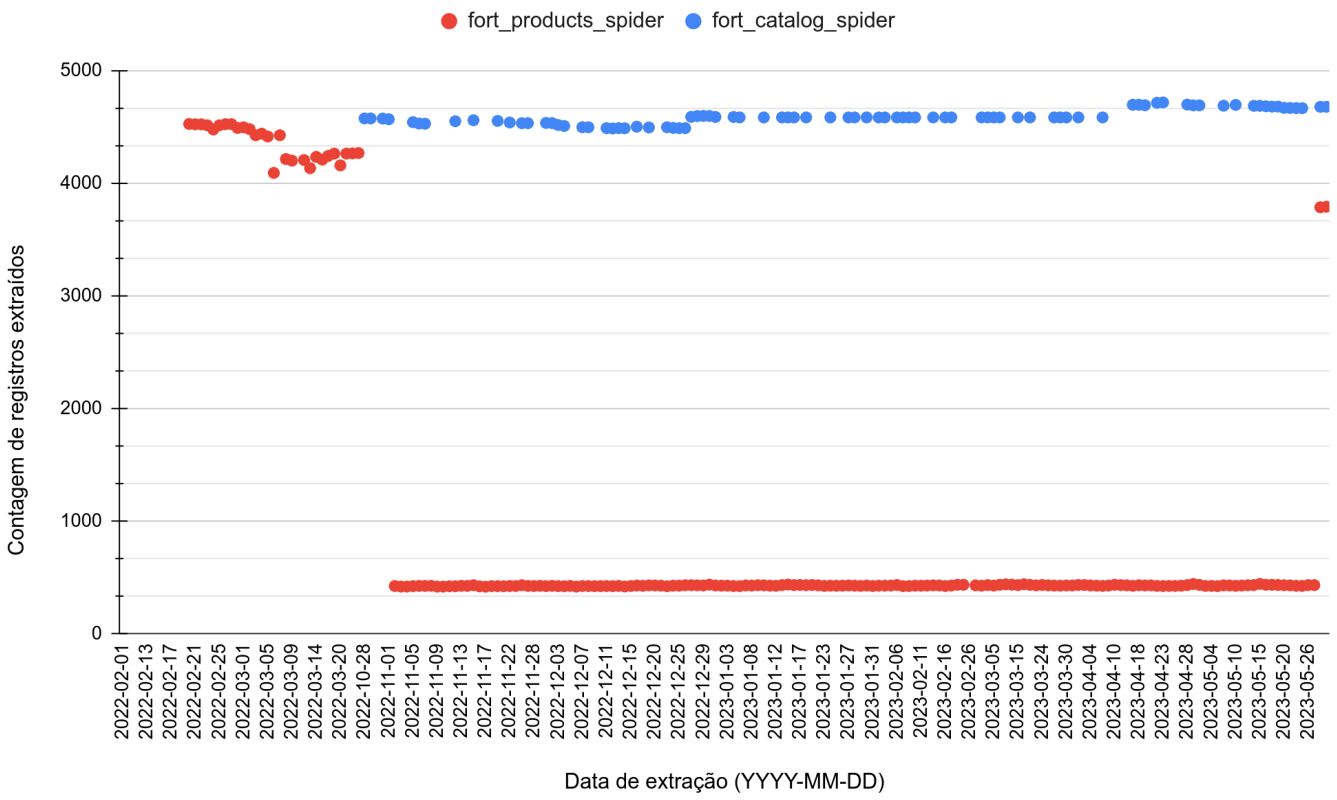
Figura 12 - Quantidade de registros extraídos por dia no Angeloni x quantidade esperada



Fonte: o Autor (2023)

Da mesma forma que o Angeloni, os scrapers do Fort também passaram por 3 fases. Antes do conhecimento do catálogo de produtos, a fase intermediária onde haviam muitos produtos repetidos sendo adquiridos e atualmente, onde o catálogo continua tendo mais produtos porém o spider está funcionando conforme esperado.

Figura 13 - Contagem de registros extraídos por dia no Fort x quantidade esperada



Fonte: o Autor (2023)

Para uma comparação histórica de um produto entre dois ou mais supermercados, ainda há a dificuldade de similaridade caso queira-se fazer a equiparação de dois produtos. Porém, isso não impede que o levantamento manual de cada nomenclatura seja efetuada. Selecionou-se então um produto comum pertencente a cesta básica: O “café 3 corações 500g tradicional”. Esse produto foi elencado recentemente por uma pesquisa do PROCON/SC como um produto que ofereceu 33.36% de variação de valores entre o mais caro e o mais barato. Sendo R$ 14.99 (KOCH S/A) e R$ 19.99 (Imperatriz). (Correios, 2023)

Cada supermercado possui uma filial padrão para utilização do site caso o usuário não digite o CEP. Para levantamento de qual o nome do produto em cada um dos supermercados, levantou-se algumas palavras chaves como '%café%', '%500%', '%cora%' e ’%tradicional%'. Com base nessas palavras, pode-se obter o produto respectivo nos 5 supermercados conforme pode-se observar na Tabela 5.

Tabela 5 - Relação de nomenclaturas encontradas pelos spiders

market_name	**name**	**product_url**
angeloni_products_spider	Café 3 Corações Tradicional 500g	https://www.angeloni.com.br/super/p/cafe-3-coracoes-tradicional-500g-2814564
angeloni_products_spider	Café 3 Corações Tradicional à Vácuo 500g	https://www.angeloni.com.br/super/p/cafe-3-coracoes-tradicional-a-vacuo-500g-4637772
big_products_spider	Café Torrado e Moído Tradicional 3 Corações Pacote 500g	https://www.big.com.br/cafe-torrado-e-moido-tradicional-3-coracoes-500g-7896005800010/p
big_products_spider	Café Tradicional a Vácuo 3 Corações 500g	https://www.big.com.br/cafe-tradicional-3-coracoes-pacote-a-vacuo-500g-7896045102440/p
bistek_products_spider	Café 3 Corações Tradicional 500g	https://www.bistek.com.br/cafe-3-coracoes-500g-tradicional.html
bistek_products_spider	Café Torrado e Moído a Vácuo Tradicional 3 Corações 500g	https://www.bistek.com.br/cafe-3-coracoes-500g-tradicional.html
fort_products_spider	Café 3 Corações Tradicional 500g	https://www.deliveryfort.com.br/cafe-3-coracoes-tradicional-500g/p
imperatriz_products_spider	CAFÉ 3 CORAÇÕES 500G A VÁCUO TRADICIONAL	https://www.supermercadosimperatriz.com.br/cafe-3-coracoes-500gr-vacuo-tradicional-48555/p

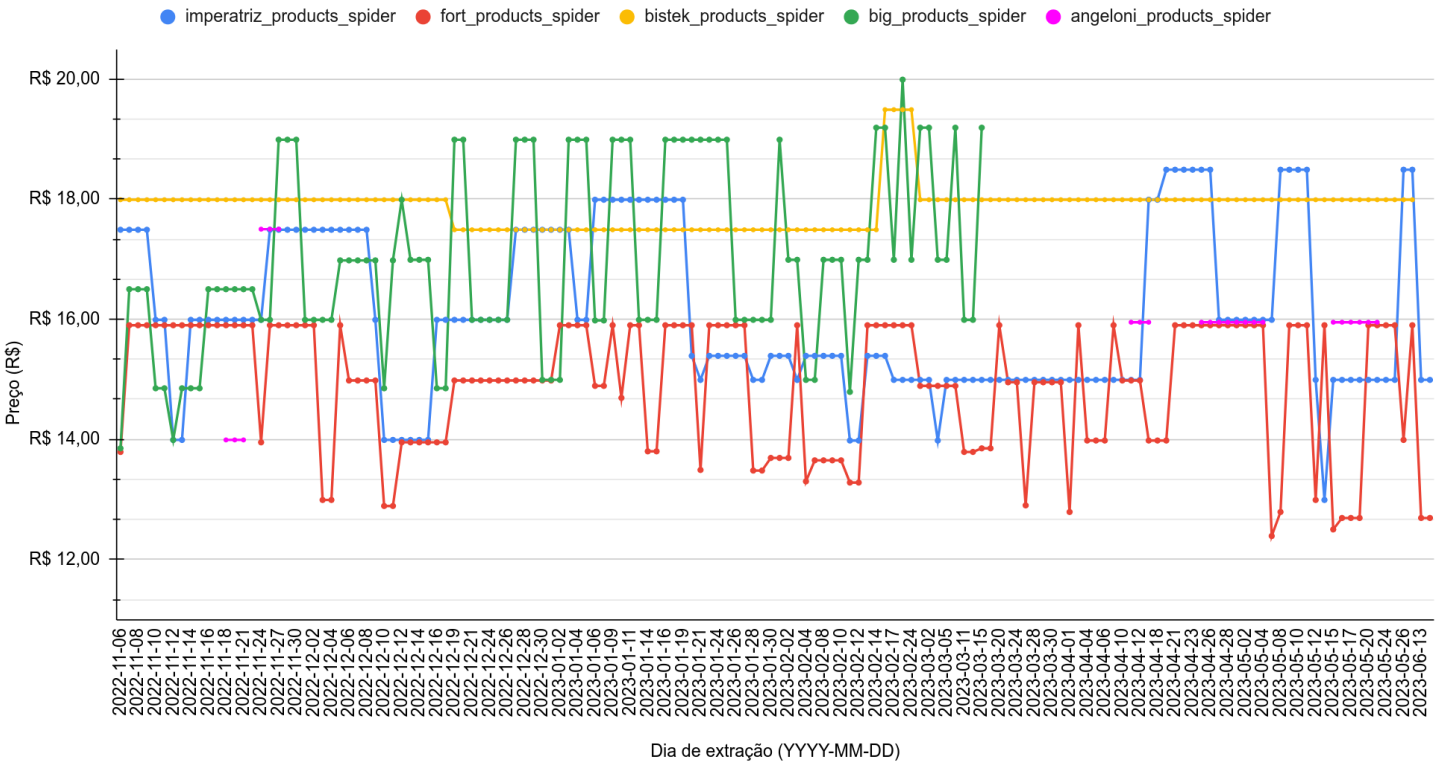
Fonte: o Autor (2023)

Pode-se observar alguns pontos interessantes sobre a Tabela 5:

- o spider bistek_products_spider possui dois produtos: "Café 3 Corações Tradicional 500g" e "Café Torrado e Moído a Vácuo Tradicional 3 Corações 500g" e os dois estão utilizando o mesmo endereço;
- Endereços do spider big_products_spider não está mais disponível.
- A complexidade da comparação de produtos é clara: há variações de ordenamento e informações extras em alguns deles, como a especificação do produto ser a Vácuo ou não.

Por fim, conformou-se todos os nomes para Café 3 Corações Tradicional 500g, excluindo aqueles que tiveram dois ou mais variações de nomes com menos registros.

Figura 14 - Análise histórica de preços do item “CAFÉ 3 CORAÇÕES 500G TRADICIONAL” para vários supermercados.



Fonte: o Autor (2023)

Assim como foi feito para o CAFÉ, uma vez que a base de informações esteja consolidada, também será possível construir trabalhos que farão a aproximação por semelhança, fazendo assim com que gráficos como esse sejam comuns.

6 CONSIDERAÇÕES FINAIS

O trabalho teve sucesso em seu objetivo: construção de um conjunto de scrapers, utilizando o framework Scrapy, para adquirir informações de produtos, como nome do produto, preço, preço promocional, preço promocional do clube, departamento em que se encontra e url do produto. Os supermercados alvos foram Angeloni, Bistek, Fort Atacadista, Giassi e Imperatriz, e foi-se necessário a especialização de todos os scrapers a fim de tratar as particularidades de aquisição de cada um. Outras tarefas complementares foram desenvolvidas, dentre elas os pipelines de tratamento e armazenamento em Postgres e na Azure Blob Storage. Além disso, utilizou-se da conteinerização de cada um dos spiders para facilitar o versionamento, empacotamento das bibliotecas e execução do script.

7 TRABALHOS FUTUROS

Como trabalhos futuros tem-se:

- Aumento da gama de supermercados como alvos da raspagem de dados de products, catalog e sitemap como o Hippo, Hiperbom, HiperSelect, Supermercado Koch, Carrefour, Prado Supermercados e Atacadão.
- Implementar testes unitários para validar que as informações adquiridas são concisas e confiáveis.
- Implementação de um monitoramento dos spiders ou, ao menos, um sistema de alertas para caso de falha de extração e ingestão.
- Utilização da hospedagem de container de cada um dos spiders dentro do ACR (Azure Container Registry) e o agendamento da execução automática através de Azure Logic Apps ou Azure Functions.
- Conformação de departamentos entre diferentes supermercados para que seja possível fazer um levantamento mais acurado da distribuição dos produtos ao longo do supermercado.

- Utilização de técnicas de NLP (processamento de linguagem natural) para assimilar produtos iguais de diferentes supermercados.
- Modelagem star-schema para simplificação das consultas, análises e normalização dos dados.
- Utilizar das informações do catálogo e validar se a periodicidade de atualização das informações dos produtos é respeitada (weekly, daily).
- Análise exploratória e estatística dos dados;
- Validar hipótese do catálogo x produtos

REFERÊNCIAS

Arthur, P. H. G. et al. **Superpagg - Uma Ferramenta Crowdsourcing para Comparação de Preços em Supermercados. Brazilian Journal of Development**. Brazilian Journal of Development. Disponível em: https://ojs.brazilianjournals.com.br/ojs/index.php/BRJD/article/view/4860/4477>. Acesso em: 14 jun. 2023.

CALÒ, Alessandro. **Extração e análise de informações jurídicas públicas**. USP, 2014. Disponível em: <https://bcc.ime.usp.br/tccs/2014/sandro/Monografia.pdf/>. Acesso em: 20 mar. 2023.

CAMARGO, C. L. C. M. et al. **Spiders de extração de dados do turismo nacional**. In: ENCONTRO DE COMPUTAÇÃO E INFORMAÇÃO DA UNIVERSIDADE LUTERANA DO BRASIL, 1., Tocantins, 2017. Anais... Tocantins: ULBRA, 2017. p. 1-7. Disponível em: <http://ulbra-to.br/encoinfo/wp-content/uploads/2020/03/Camargo_et_al_2017_Spiders_de_extra%C3%A7%C3%A3o_de_dados_do_turismo_naci_d1APl7w.pdf>. Acesso em 13 jun. 2023.

CARVALHO, M. A. B. **Sistema web para comparação dos preços de supermercados online**. Monografia. Universidade Federal de Ouro Preto, Ouro Preto. out, 2022. Disponível em: <https://monografias.ufop.br/bitstream/35400000/5477/6/MONOGRAFIA_SistemaWebComparacao.pdf>. Acesso em: 14 jun. 2023.

CECI, Michelangelo; LANOTTE, Pasqua Fabiana. **Closed sequential pattern mining for sitemap generation**. World Wide Web, v. 24, n. 1, p. 175-203, 2021. Disponível em: <http://www.mecs-press.net/ijem/ijem-v4-n3/IJEM-V4-N3-3.pdf>. Acesso em 13 jun. 2023.

DELUCA, M. A. M. **Varejo supermercadista da Grande Florianópolis: uma análise das cinco forças competitivas de Porter**. Florianópolis, 2001. Disponível em: <https://repositorio.ufsc.br/handle/123456789/82082?show=full>. Acesso em: 11 jun. 2023.

Glez-Peña, D. et al.. **Web scraping technologies in an API world**. Briefings in

Bioinformatics, v. 15, n. 5, 04 2013, p. 788–797. ISSN 1467-5463. Disponível em: <https://academic.oup.com/bib/article/15/5/788/2422275>. Acesso em: 02 jun. 2023.

GOOGLE. **Product data specification**. 2022. Google Merchant Center. Disponível em: https://support.google.com/merchants/answer/7052112. Acesso em: 12 dez. 2022.

IREK, J. **Web scraping for food price research**. British Food Journal, v. 121, p. 3350–3361, 2019. DOI: 10.1108/BFJ-02-2019-0081. Disponível em: <https://www.researchgate.net/publication/337186825_Web_scraping_for_food_price_research>. Acesso em 02 mar. 2023.

MATHIAS, G. N. **qFex: um crawler para busca e extração de questionários de pesquisa em documentos HTML**. 2017. Trabalho de Conclusão de Curso – Curso de Sistemas de Informação, Universidade Federal de Santa Catarina, UFSC, Florianópolis, 2017. Acesso em: 13 jun. 2023.

MENDONÇA, Eduardo. **Extração Resiliente de Dados RDF a partir de Fontes Dinâmicas em Linguagem de Marcação**. Disponível em: <http://www.livrosgratis.com.br/arquivos_livros/cp108124.pdf>. Acesso em: 04 jun. 2023.

MITCHELL, R. **Web Scraping com Python: Coletando mais dados da web moderna.** [S.l.]: Novatec Editora, 2019. ISBN 9788575227343.

KOUZIS-LOUKAS, D. **Learning Scrapy**. Birmingham: Packt Publishing, 2016. 270 p. Disponível em: <https://scrapy.org/>. Acesso em: 12 jun. 2023.

LUIZ, D. C. V. **EXTRAÇÃO E COMBINAÇÃO POR SIMILARIDADE: UM ESTUDO DE CASO NAS REDES DE SUPERMERCADOS EM FLORIANÓPOLIS**. 2022. Trabalho de Conclusão de Curso – Curso de Sistemas de Informação, Universidade Federal de Santa Catarina, UFSC, Florianópolis, 2022. Acesso em: 12 jun. 2023.

ROBBINS, J. N. **Learning Web Design**: a beginner's guide to html, css, javascript, and web graphics. 4. ed. Califórnia: O'Reilly Media, 2012. 624 p.

MARKETSCRAPER: UM SCRAPER PARA A BUSCA E EXTRAÇÃO DE INFORMAÇÕES DE PRODUTOS EM SUPERMERCADOS NA REGIÃO DE FLORIANÓPOLIS

Alan Djon Lüdke[1], André Wüst Zibetti[2]

Departamento de Informática e Estatística - INE
Universidade Federal de Santa Catarina - UFSC/Florianópolis

alan.ludke@grad.ufsc.br, andre.zibetti@ufsc.br

Abstract: During the pandemic, there has been a significant increase in the demand for food and essential products, leading to a surge in online purchases of these items. In response to this situation, companies quickly adapted by making their products available in virtual stores to meet consumer needs. In this context, there arises a need for a tool that facilitates price comparison among different online supermarkets. Based on this growing demand, this work aims to develop a set of scrapers capable of extracting and organizing data on products and prices from different supermarkets in the Florianópolis region: Angeloni, Bistek, Fort, Giassi, and Imperatriz. This tool enables the centralization of this information in a single source, allowing consumers to compare product prices and access a price history. By implementing these scrapers, the goal is to provide an efficient solution to assist consumers in making informed decisions, contributing to a more satisfying and transparent shopping experience in the online supermarket market.

Resumo: Durante a pandemia, observou-se um expressivo aumento na demanda por alimentos e produtos essenciais, impulsionando a compra online desses itens. Diante desse cenário, as empresas se adaptaram rapidamente, disponibilizando seus produtos em lojas virtuais para atender às necessidades dos consumidores. Nesse contexto, surge a necessidade de uma ferramenta que facilite a comparação de preços entre diferentes supermercados online. Com base nessa demanda crescente, este trabalho tem como objetivo principal o desenvolvimento de um conjunto de scrapers capazes de extrair e organizar dados sobre produtos e preços de diferentes supermercados da região de Florianópolis: Angeloni, Bistek, Fort, Giassi e Imperatriz. Essa ferramenta permite a centralização dessas informações em uma única fonte, possibilitando aos consumidores comparar preços de produtos e acessar um histórico de preços. Com a implementação desses scrapers, busca-se fornecer uma solução eficiente para auxiliar os consumidores na tomada de decisões informadas, contribuindo para uma experiência de compra mais satisfatória e transparente no mercado de supermercados online.

1. Introdução

O aumento da disponibilização de informações na internet para auxílio na tomada de decisão tem trazido inúmeras facilidades para os consumidores, entre elas a capacidade de pesquisar preços de produtos em supermercados. Essas informações são extremamente valiosas para o consumidor comum, pois permitem comparar preços e encontrar as melhores ofertas. No

entanto, a navegação entre diferentes plataformas de supermercados ainda é um desafio, dificultando a comparação de preços entre diferentes estabelecimentos.

Na região de Florianópolis, assim como em outras áreas metropolitanas, a importância das informações de produtos em supermercados é amplificada pela abundância de estabelecimentos comerciais nesse setor. Segundo estudo de DELUCA (2001), o setor supermercadista na região tem um papel fundamental na economia local, contribuindo para o desenvolvimento regional e gerando empregos. No entanto, os consumidores enfrentam desafios para obter informações consistentes e atualizadas sobre os produtos oferecidos por diferentes supermercados na região.

A necessidade de informações precisas e comparáveis entre supermercados motivou o desenvolvimento de técnicas automatizadas para aquisição desses dados. Segundo análise do Departamento Intersindical de Estatística e Estudos Socioeconômicos (Dieese) realizado no período de janeiro a dezembro de 2022 o encarecimento da cesta básica em Florianópolis foi de R$ 73,00 ou pouco mais de 10%. Ter a disponibilidade de informações precisas e confiáveis promove transparência no mercado de supermercados, estimulando a concorrência entre os estabelecimentos e consequentemente a preços mais competitivos.

Os crawlers ou scrapers têm sido amplamente utilizados para coletar informações de produtos em diferentes setores, incluindo supermercados. Essas ferramentas automatizadas têm a capacidade de extrair dados como preços, descrições de produtos e categorias de maneira eficiente e em grande escala.

Apesar dos avanços alcançados por trabalhos anteriores, algumas limitações permanecem. Por exemplo, conforme mencionado por Arthur, et al (2019), muitos trabalhos focam em um único supermercado, têm escopo limitado geograficamente ou então utilizam de APIs prontas para construírem as aplicações. Isso restringe a abrangência das informações disponíveis para os consumidores e limita a tomada de decisão.

Enquanto existem trabalhos que abordam a coleta de informações de produtos em supermercados, é importante ressaltar que esta pesquisa se diferencia por considerar aspectos específicos da região de Florianópolis. Além disso, para aqueles trabalhos que abordam a região, propõe-se o aumento da gama de scrapers de supermercados na região. O trabalho também tem o potencial de gerar contribuições sociais, econômicas e tecnológicas, promovendo a transparência e a concorrência saudável no comércio varejista de alimentos da região.

2. Fundamentação teórica

2.1. Extração de dados na web

A extração de dados na web refere-se ao processo de coletar informações disponíveis em páginas da web para posterior análise e uso. Com o rápido crescimento da quantidade de dados disponíveis online, a extração de dados na web tornou-se uma área de pesquisa e desenvolvimento de grande importância. A extração de dados pode ser aplicada em diversas áreas, como pesquisa acadêmica, análise de mercado, monitoramento de preços, entre outros.

Segundo Mendonça (2003), as fontes de informação são sistemas que atendem a consultas, fornecendo uma resposta adequada para cada solicitação submetida. No contexto da Internet, essas fontes possuem formatos textuais e são manipuladas pelo paradigma de requisição e resposta implementado pelo protocolo Hypertext Transfer Protocol (HTTP). Geralmente, fazem uso dos formatos semi estruturados como o HTML e o XML.

Ao realizar a extração de dados na web, é fundamental compreender a estrutura das páginas e os formatos dos dados desejados. Os dados podem ser estruturados, como tabelas e listas, ou não estruturados, como texto livre e imagens (MITCHELL, 2019).

O uso de web crawlers ou web scrappers tornou-se cada vez mais comum nos últimos anos. Essas técnica permite aos usuários extrair dados de sites e utilizá-los para diversos fins. Indivíduos e empresas podem coletar grandes quantidades de dados de várias fontes e usá-los para tomar decisões informadas. Isso levou ao aumento da concorrência entre as empresas, que se esforçam para oferecer as melhores informações aos seus clientes (Glez-Peña, D. et al., 2013).

O web scraping envolve a extração de dados específicos de uma página da web, como textos, imagens, tabelas ou qualquer outro tipo de conteúdo estruturado. É uma abordagem mais direcionada, na qual os dados são extraídos de maneira seletiva com base em critérios específicos. Geralmente, o web scraping é realizado por meio de programas ou scripts que analisam o código HTML das páginas para identificar e extrair os elementos desejados (Calò, 2014).

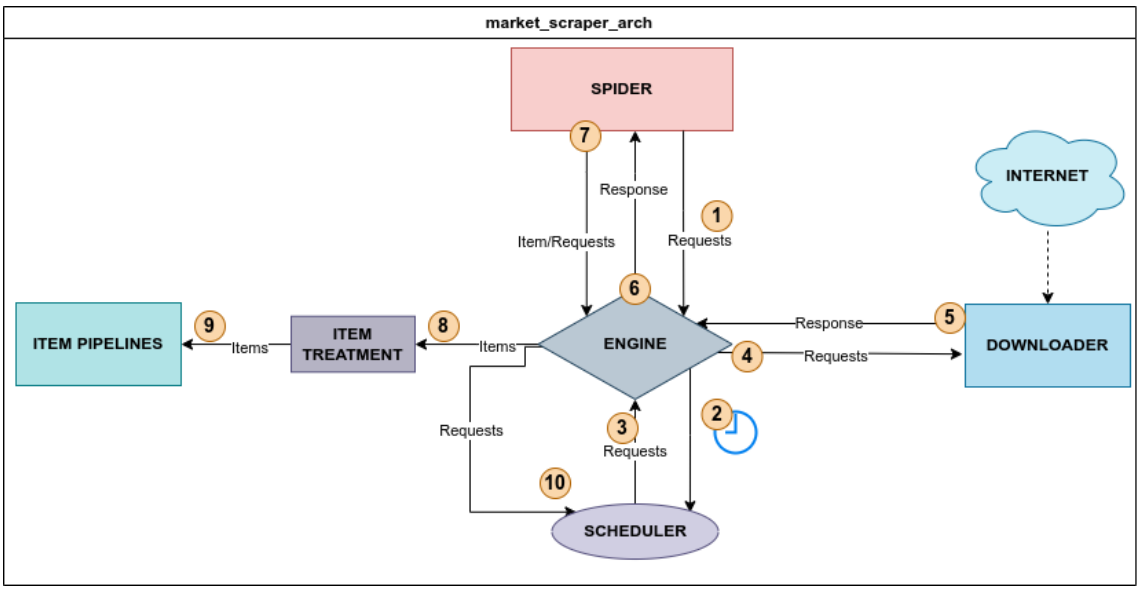
2.2. Framework Scrapy

O Scrapy é um framework escrito em Python que permite a criação de crawlers e scrapers para navegar e raspar grandes conjuntos de dados a partir da web ou de outras fontes (Scrapy, 2023). Ele pode ser usado em diversas aplicações como, por exemplo, mineração de dados, processamento de informação ou histórico de arquivos (Camargo et al, 2017).

O Scrapy possui uma vasta gama de recursos, bem como provê várias maneiras de se realizar uma mesma tarefa, no entanto, dois elementos são considerados como elementos chave no desenvolvimento desta pesquisa: as spiders e os seletores. Uma spider é representada por uma classe que define o comportamento personalizado de rastreamento e análise em páginas de um site específico ou de um conjunto deles, isto é, é uma classe que especifica regras de web crawling e web scraping (Scrapy, 2023).

Já os seletores são critérios aplicados dentro das spiders para escolher segmentos de um documento XML/HTML usando expressões XPath, CSS ou expressões regulares.

Figura 1 - Arquitetura geral do scraper



Fonte: o Autor (2023)

3. Trabalhos Relacionados

3.1. Meus Preços

O aplicativo mobile Meus Preços possibilita a pesquisa de preços em supermercados de São Paulo utilizando as informações de compras dos cupons fiscais do sistema da Nota Fiscal Paulista. Normalmente os estabelecimentos demoram um dia para enviar os dados dos cupons fiscais para o sistema da Nota Fiscal Paulista. Porém, de acordo com o site oficial do aplicativo, os estabelecimentos têm até 60 dias para enviar estas informações. Com isso, os preços praticados pelos estabelecimentos podem ser alterados neste intervalo. Assim como o Superpagg, também só oferece uma análise descritiva dos custos.

3.2. ClickSuper

O aplicativo mobile recentemente lançado em Florianópolis, se propõe a deixar milhares de produtos atualizados diariamente nas principais redes de mercados, supermercados, atacados, farmácias e petshops da sua região, comparados por preço diretamente no aplicativo. Porém, como sua atuação é ampla, a ferramenta não provê a grande quantidade de supermercados de Florianópolis, recomendando a compra de produtos em supermercados apresentando preços de produtos de supermercados de cidades vizinhas.

3.3. QFEX

A proposta do trabalho de conclusão de curso proposta por Gilnei, intitulado “qFex: um crawler para busca e extração de questionários de pesquisa em documentos HTML” consiste em um Web Crawler para extrair questionários de uma lista de sites e salvá-los de forma estruturada, a fim de criar uma base para elaboração de novos questionários (MATHIAS, G. N., 2017).

Para varrer a web, o Crawler se baseia em uma lista de seeds e busca questionários utilizando de várias heurísticas criadas pelo autor e que trazem uma taxa de sucesso aceitável.
A solução pode ser dividida em duas: O Crawler, que deve varrer a Web em busca de questionários que possuam certas características e então deve salvar seus links em um banco de dados e o Extractor, que por sua vez, deve apanhar os links adquiridos na etapa anterior e extrair os dados dos questionários para um outro banco de dados.

3.4. EXTRAÇÃO E COMBINAÇÃO POR SIMILARIDADE: UM ESTUDO DE CASO NAS REDES DE SUPERMERCADOS EM FLORIANÓPOLIS

A proposta do trabalho de conclusão de curso de Diogo conta com o desenvolvimento de uma API que age como um Web scraper/wrapper a fim de realizar a extração de preços e produtos de supermercados que disponibilizam serviço de vendas online na região de Florianópolis. Os supermercados escolhidos foram o Bistek, Imperatriz e Fort Atacadista, (LUIZ, 2022).

Após a extração, em seu trabalho fez-se a análise do grau de similaridade de texto após uma etapa de transformação e normalização, chamada de camada de pré-processamento. Essa camada pode ser dividida em duas etapas: integração e indexação, onde os dados extraídos são comparados através de algoritmos de similaridades a fim de combinar os produtos identificados como similares e salvar estas combinações.

O trabalho proposto teve o foco na comparação por similaridade entre produtos de supermercados diferentes, portanto, além de objetivo os trabalhos se diferem nas técnicas de extração, framework utilizado, quantidade de supermercados analisados e técnicas de verificação de confiabilidade de informações ao período de análise. Além disso, o autor

realizou o trabalho de aquisição dos dados com o uso de APIs. Esta utilização entrega o dado de uma forma estruturada porém é altamente dependente da disponibilidade da API.

4. Market Scraper

O projeto visa adquirir informações de produtos dos supermercados na região de Florianópolis. Para isso, fez-se um levantamento dos supermercados que possuíssem venda online de produtos, fossem comumente conhecido pela população e possuíssem páginas web funcionais. Acordou-se que os supermercados que contemplam tais requisitos são: Angeloni, Bistek, Fort Atacadista, Giassi e Imperatriz.

Cada site possui uma combinação de elementos únicos que formam a estrutura do seu corpo e também a disponibilização do seu conteúdo. O frontend dessa estrutura é composta basicamente por HTML, CSS e Javascript. (ROBBINS, 2012).

Para facilitar a diferenciação entre os diversos tipos de páginas, dividiu-se então em categorias. Para cada uma dessas categorias foi realizadas a raspagem de dados:

- **'sitemap'**: Esse tipo de página tem o formado em XML e descreve a estrutura e organização de um site. De acordo com Ceci & Lanotte (2020), ele é projetado para ajudar os mecanismos de busca a indexarem e navegarem pelo conteúdo de um site de forma mais eficiente. O sitemap.xml fornece informações sobre as páginas do site e sua hierarquia. Isso permite que os mecanismos de busca rastreiem e indexem o conteúdo de maneira mais precisa, melhorando a visibilidade e a classificação do site nos resultados de pesquisa. Essa hierarquia dispõe alguns atributos como uma lista de endereços responsáveis por centralizar os departamentos e uma lista de endereços responsáveis por centralizar o catálogo de produtos de todo o site.
- **'catalog':** Esse tipo de página tem o formado em XML e proporciona um catálogo de produtos acessíveis pelo site incluindo qual o endereço de cada um deles, a frequência de atualização e a data da última modificação.
- **'products'**: Esse tipo de página é o mais complexo dos três. Ele é responsável por centralizar a interface com o cliente comum, que navega na página em busca de alimentar sua cesta de produtos e assim efetuar a compra. Nesse tipo de página é comum que haja uma lista de departamentos, cada um com seu tipo de produto, e também uma combinação entre paginação e filtros para facilitar a navegação. Atributos como nome do produto, preço, preço promocional, preço promocional do clube, departamento em que se encontra e url do produto podem e foram alvo desse tipo de scraping.

5. Validação da ferramenta

A construção inicial dos scrapers se teve no dia 01 de fevereiro de 2022 onde o primeiro scraper de produtos foi projetado para adquirir as informações do site do Bistek Supermercados. Desde então, tem-se feito requisições geralmente diárias para que pudessemos utilizar da maior base de dados e estratégias de scraping.

Cada um dos spiders podem ser validados através da análise da contagem diária de produtos raspados. Pensando nisso, construiu-se a tabela abaixo que consiste de uma análise de todo o período: dia 01 de fevereiro de 2022 até o dia 14 de junho de 2023. Tem-se então qual a data mínima, e provável primeira execução de cada um dos scrapers, a data máxima e também a contagem de dias únicos extraídos. Este último é importante pois possibilita o grau de aproveitamento de cada um dos spiders, contemplando todas as dificuldades encontradas durante o caminho.

Tabela 1 - Relação de início, fim e dias extraídos por cada spider.

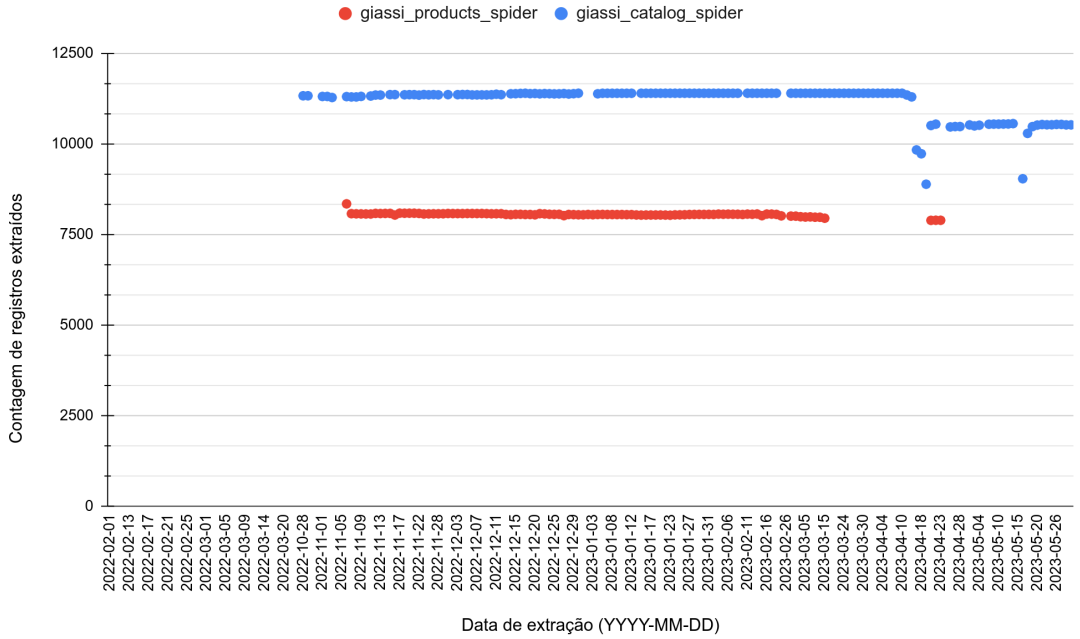
Nome do spider	Data mínima de extração	Data máxima de extração	Contagem de dias únicos extraídos
angeloni_catalog_spider	2022-10-28	2023-06-14	83
fort_catalog_spider	2022-10-28	2023-06-14	86
big_products_spider	2022-11-06	2023-03-15	99
giassi_products_spider	2022-11-06	2023-04-23	102
giassi_catalog_spider	2022-10-28	2023-06-14	140
imperatriz_products_spider	2023-06-13	2023-06-14	150
imperatriz_catalog_spider	2022-10-28	2023-06-14	153
fort_sitemap_spider	2022-10-28	2023-06-14	159
giassi_sitemap_spider	2022-10-28	2023-06-14	159
imperatriz_sitemap_spider	2022-10-28	2023-06-14	159
angeloni_sitemap_spider	2022-10-28	2023-06-14	160
fort_products_spider	2022-02-20	2023-06-14	182
angeloni_products_spider	2022-02-13	2023-06-14	189
bistek_products_spider	2022-02-01	2023-06-14	194

Fonte: o Autor (2023)

Percebe-se que a descoberta do endereço responsável pelo catálogo dos supermercados impactou diretamente na quantidade de dias adquiridos pois quase todos os spiders do tipo catálogo se encontram nos primeiros lugares em relação à menos quantidade de dias com extração.

Para melhor entendimento das fases de cada um dos spiders responsáveis por raspar as informações dos produtos, construiu-se um gráfico que coloca lado a lado a quantidade de produtos únicos esperados, adquiridos através do catálogo de produtos, e a quantidade de itens adquiridos através do spider do tipo products conforme pode-se verificar na Figura 2.

Figura 2 - Quantidade de registros extraídos por dia no Giassi x quantidade esperada



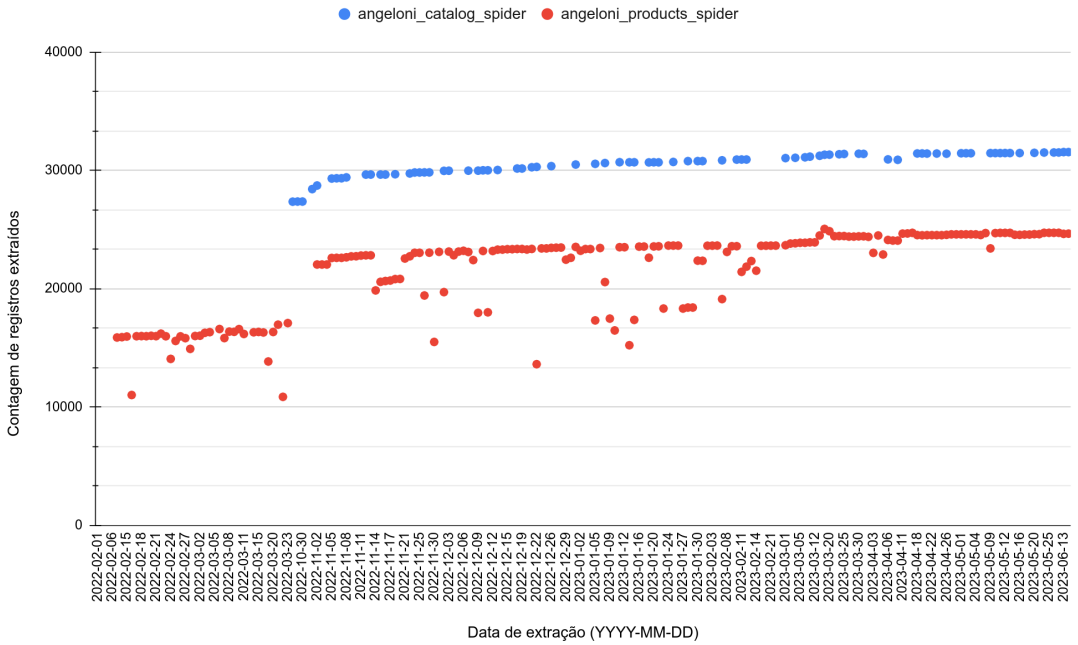
Fonte: o Autor (2023)

Pode-se perceber que há sempre mais itens admitidos pelo catálogo de produtos do que aqueles alcançados pelo scraping. Além disso, também percebe-se que recentemente o spider de produtos teve seu funcionamento cessado por um período, o problema foi concertado e, após alguns dias, o problema voltou novamente e o status desse spider é inativo no momento.

Quando analisa-se os produtos extraídos nos sites do Angeloni, mostrado na Figura 12, tem-se claramente 3 cenários:

- **Cenário 1:** Desde o início da construção do scraper até o dia 20 de março de 2022 não havia extração do tipo catalog. Isto se deve pois, naquela época, não era sabido que ele existia e nem que era disponível para facilitar a navegação/SEO.
- **Cenário 2:** Até o dia 05 de março de 2023 teve-se um período onde a contagem de produtos raspados não era constante.
- **Cenário 3:** Após esta data até hoje, a aquisição está estável e confiável.

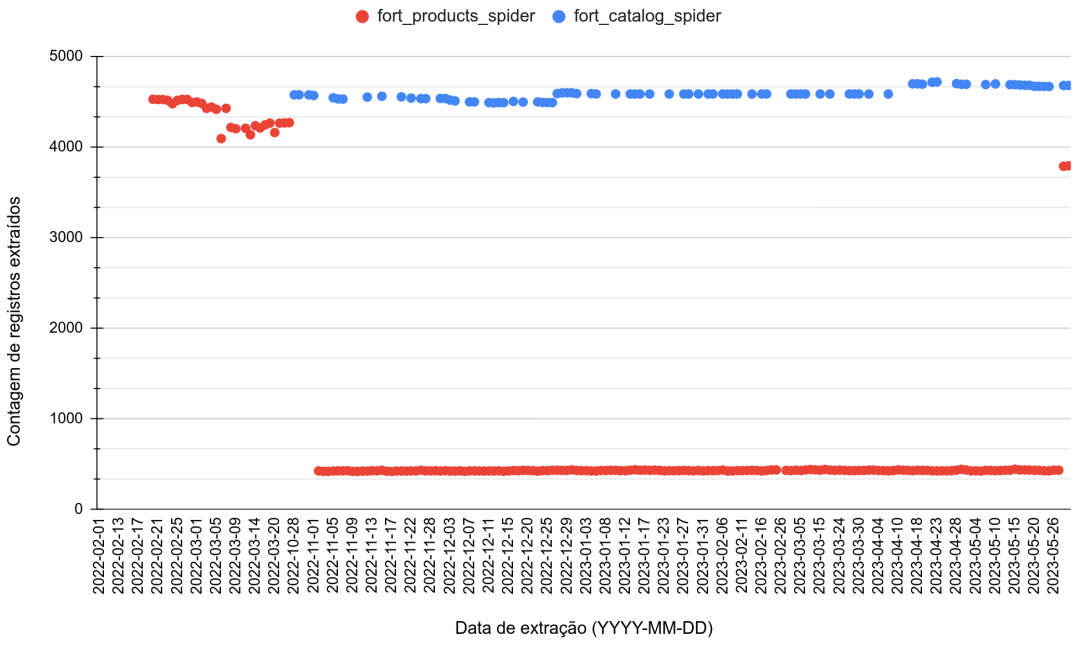
Figura 3 - Quantidade de registros extraídos por dia no Angeloni x quantidade esperada



Fonte: o Autor (2023)

Da mesma forma que o Angeloni, os scrapers do Fort também passaram por 3 fases. Antes do conhecimento do catálogo de produtos, a fase intermediária onde haviam muitos produtos repetidos sendo adquiridos e atualmente, onde o catálogo continua tendo mais produtos porém o spider está funcionando conforme esperado.

Figura 4 - Contagem de registros extraídos por dia no Fort x quantidade esperada



Fonte: o Autor (2023)

Para uma comparação histórica de um produto entre dois ou mais supermercados, ainda há a dificuldade de similaridade caso queira-se fazer a equiparação de dois produtos. Porém, isso não impede que o levantamento manual de cada nomenclatura seja efetuada. Selecionou-se então um produto comum pertencente a cesta básica: O “café 3 corações 500g tradicional”. Esse produto foi elencado recentemente por uma pesquisa do PROCON/SC como um produto que ofereceu 33.36% de variação de valores entre o mais caro e o mais barato. Sendo R$ 14.99 (KOCH S/A) e R$ 19.99 (Imperatriz). (Correios, 2023)

Cada supermercado possui uma filial padrão para utilização do site caso o usuário não digite o CEP. Para levantamento de qual o nome do produto em cada um dos supermercados, levantou-se algumas palavras chaves como '%café%', '%500%', '%cora%' e ’%tradicional%'. Com base nessas palavras, pode-se obter o produto respectivo nos 5 supermercados conforme pode-se observar na Tabela 5.

Tabela 2 - Relação de nomenclaturas encontradas pelos spiders

market_name	**name**
angeloni_products_spider	Café 3 Corações Tradicional 500g
angeloni_products_spider	Café 3 Corações Tradicional à Vácuo 500g
big_products_spider	Café Torrado e Moído Tradicional 3 Corações Pacote 500g
big_products_spider	Café Tradicional a Vácuo 3 Corações 500g
bistek_products_spider	Café 3 Corações Tradicional 500g
bistek_products_spider	Café Torrado e Moído a Vácuo Tradicional 3 Corações 500g
fort_products_spider	Café 3 Corações Tradicional 500g
imperatriz_products_spider	CAFÉ 3 CORAÇÕES 500G A VÁCUO TRADICIONAL

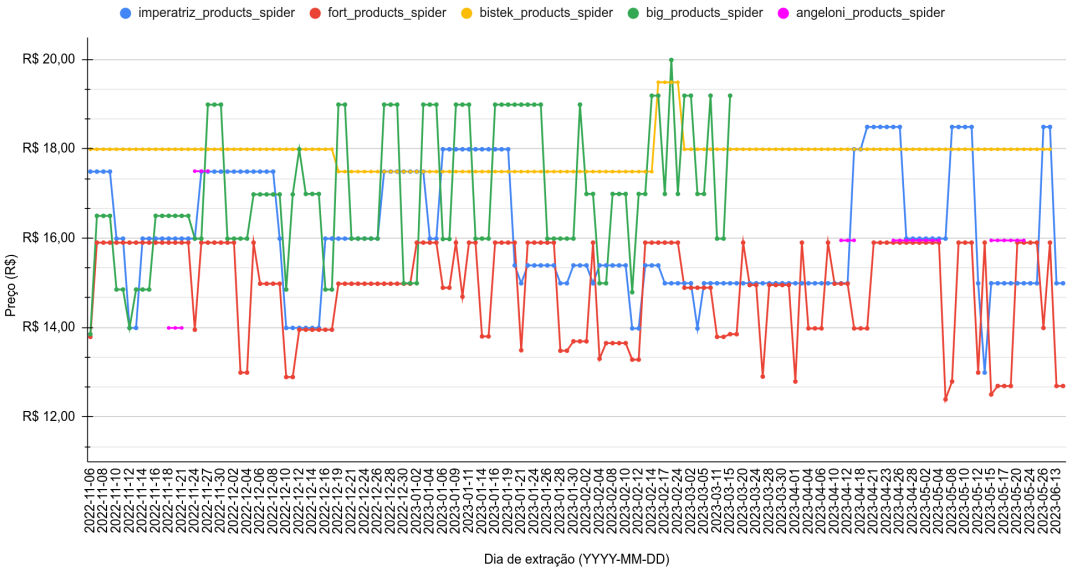
Fonte: o Autor (2023)

Pode-se observar alguns pontos interessantes sobre a Tabela 5:

- o spider bistek_products_spider possui dois produtos: "Café 3 Corações Tradicional 500g" e "Café Torrado e Moído a Vácuo Tradicional 3 Corações 500g" e os dois estão utilizando o mesmo endereço;
- Endereços do spider big_products_spider não está mais disponível.
- A complexidade da comparação de produtos é clara: há variações de ordenamento e informações extras em alguns deles, como a especificação do produto ser a Vácuo ou não.

Por fim, conformou-se todos os nomes para Café 3 Corações Tradicional 500g, excluindo aqueles que tiveram dois ou mais variações de nomes com menos registros.

Figura 5 - Análise histórica de preços do item “CAFÉ 3 CORAÇÕES 500G TRADICIONAL” para vários supermercados.



Fonte: o Autor (2023)

Assim como foi feito para o CAFÉ, uma vez que a base de informações esteja consolidada, também será possível construir trabalhos que farão a aproximação por semelhança, fazendo assim com que gráficos como esse sejam comuns.

6. Considerações finais

O trabalho teve sucesso em seu objetivo: construção de um conjunto de scrapers, utilizando o framework Scrapy, para adquirir informações de produtos, como nome do produto, preço, preço promocional, preço promocional do clube, departamento em que se encontra e url do produto. Os supermercados alvos foram Angeloni, Bistek, Fort Atacadista, Giassi e Imperatriz, e foi-se necessário a especialização de todos os scrapers a fim de tratar as particularidades de aquisição de cada um. Outras tarefas complementares foram desenvolvidas, dentre elas os pipelines de tratamento e armazenamento em Postgres e na Azure Blob Storage. Além disso, utilizou-se da conteinerização de cada um dos spiders para facilitar o versionamento, empacotamento das bibliotecas e execução do script.

7. Trabalhos futuros

Como trabalhos futuros tem-se:

- Aumento da gama de supermercados como alvos da raspagem de dados de products, catalog e sitemap como o Hippo, Hiperbom, HiperSelect, Supermercado Koch, Carrefour, Prado Supermercados e Atacadão.
- Implementar testes unitários para validar que as informações adquiridas são concisas e confiáveis.
- Implementação de um monitoramento dos spiders ou, ao menos, um sistema de alertas para caso de falha de extração e ingestão.

- Utilização da hospedagem de container de cada um dos spiders dentro do ACR (Azure Container Registry) e o agendamento da execução automática através de Azure Logic Apps ou Azure Functions.
- Conformação de departamentos entre diferentes supermercados para que seja possível fazer um levantamento mais acurado da distribuição dos produtos ao longo do supermercado.
- Utilização de técnicas de NLP (processamento de linguagem natural) para assimilar produtos iguais de diferentes supermercados.
- Modelagem star-schema para simplificação das consultas, análises e normalização dos dados.
- Utilizar das informações do catálogo e validar se a periodicidade de atualização das informações dos produtos é respeitada (weekly, daily).
- Análise exploratória e estatística dos dados;
- Validar hipótese do catálogo x produtos

8. Referências

Arthur, P. H. G. et al. **Superpagg - Uma Ferramenta Crowdsourcing para Comparação de Preços em Supermercados. Brazilian Journal of Development**. Brazilian Journal of Development. Disponível em: https://ojs.brazilianjournals.com.br/ojs/index.php/BRJD/article/view/4860/4477>. Acesso em: 14 jun. 2023.

CALÒ, Alessandro. **Extração e análise de informações jurídicas públicas**. USP, 2014. Disponível em: <https://bcc.ime.usp.br/tccs/2014/sandro/Monografia.pdf/>. Acesso em: 20 mar. 2023.

CAMARGO, C. L. C. M. et al. **Spiders de extração de dados do turismo nacional**. In: ENCONTRO DE COMPUTAÇÃO E INFORMAÇÃO DA UNIVERSIDADE LUTERANA DO BRASIL, 1., Tocantins, 2017. Anais... Tocantins: ULBRA, 2017. p. 1-7. Disponível em: <http://ulbra-to.br/encoinfo/wp-content/uploads/2020/03/Camargo_et_al_2017_Spiders_de_extra%C3%A7%C3%A3o_de_dados_do_turismo_naci_d1APl7w.pdf>. Acesso em 13 jun. 2023.

CARVALHO, M. A. B. **Sistema web para comparação dos preços de supermercados online**. Monografia. Universidade Federal de Ouro Preto, Ouro Preto. out, 2022. Disponível em: <https://monografias.ufop.br/bitstream/35400000/5477/6/MONOGRAFIA_SistemaWebComparacao.pdf>. Acesso em: 14 jun. 2023.

CECI, Michelangelo; LANOTTE, Pasqua Fabiana. **Closed sequential pattern mining for sitemap generation**. World Wide Web, v. 24, n. 1, p. 175-203, 2021. Disponível em: <http://www.mecs-press.net/ijem/ijem-v4-n3/IJEM-V4-N3-3.pdf>. Acesso em 13 jun. 2023.

DELUCA, M. A. M. **Varejo supermercadista da Grande Florianópolis: uma análise das cinco forças competitivas de Porter**. Florianópolis, 2001. Disponível em:

<https://repositorio.ufsc.br/handle/123456789/82082?show=full>. Acesso em: 11 jun. 2023.

Glez-Peña, D. et al.. **Web scraping technologies in an API world**. Briefings in Bioinformatics, v. 15, n. 5, 04 2013, p. 788–797. ISSN 1467-5463. Disponível em: <https://academic.oup.com/bib/article/15/5/788/2422275>. Acesso em: 02 jun. 2023.

GOOGLE. **Product data specification**. 2022. Google Merchant Center. Disponível em: https://support.google.com/merchants/answer/7052112. Acesso em: 12 dez. 2022.

IREK, J. **Web scraping for food price research**. British Food Journal, v. 121, p. 3350–3361, 2019. DOI: 10.1108/BFJ-02-2019-0081. Disponível em: <https://www.researchgate.net/publication/337186825_Web_scraping_for_food_price_research>. Acesso em 02 mar. 2023.

MATHIAS, G. N. **qFex: um crawler para busca e extração de questionários de pesquisa em documentos HTML**. 2017. Trabalho de Conclusão de Curso – Curso de Sistemas de Informação, Universidade Federal de Santa Catarina, UFSC, Florianópolis, 2017. Acesso em: 13 jun. 2023.

MENDONÇA, Eduardo. **Extração Resiliente de Dados RDF a partir de Fontes Dinâmicas em Linguagem de Marcação**. Disponível em: <http://www.livrosgratis.com.br/arquivos_livros/cp108124.pdf>. Acesso em: 04 jun. 2023.

MITCHELL, R. **Web Scraping com Python: Coletando mais dados da web moderna.** [S.l.]: Novatec Editora, 2019. ISBN 9788575227343.

KOUZIS-LOUKAS, D. **Learning Scrapy**. Birmingham: Packt Publishing, 2016. 270 p. Disponível em: <https://scrapy.org/>. Acesso em: 12 jun. 2023.

LUIZ, D. C. V. **EXTRAÇÃO E COMBINAÇÃO POR SIMILARIDADE: UM ESTUDO DE CASO NAS REDES DE SUPERMERCADOS EM FLORIANÓPOLIS**. 2022. Trabalho de Conclusão de Curso – Curso de Sistemas de Informação, Universidade Federal de Santa Catarina, UFSC, Florianópolis, 2022. Acesso em: 12 jun. 2023.

ROBBINS, J. N. **Learning Web Design**: a beginner's guide to html, css, javascript, and web graphics. 4. ed. Califórnia: O'Reilly Media, 2012. 624 p.