Apartment Scraper

# Overview

Your program is a web scraper that scrapes information about apartments for sale from the website sreality.cz. It uses the Puppeteer library to automate a headless Chrome browser and navigate to each page of search results, extract information about each apartment and store it in memory.

The backend is built using Node.js and the Express.js framework. It provides a REST API for retrieving the scraped apartment data from the database. The API endpoint /apartments supports pagination with a default page size of 25. It returns a JSON response with the apartments data retrieved from the database.

The backend uses the PostgreSQL database to store the scraped data. It connects to the database using the pg package and the configuration details provided in the pool object. It uses connection pooling to optimize performance and handle multiple requests concurrently.

# Usage

To use the program, you need to have Node.js and Puppeteer installed. Once these dependencies are installed, you can run the program by executing the command node scraper.js in your terminal. The program will then begin scraping apartment data from the sreality.cz website.

# Configuration

The program is currently configured to scrape up to 25 pages of search results, and up to 500 apartments in total. These values can be adjusted by modifying the MAX\_PAGES and MAX\_APTS constants at the beginning of the code.

# Output

The program outputs the scraped apartment data to the console, in the format:

Name: [apartment name]

Locality: [apartment locality]

Price: [apartment price]

Images:[array of images]

At the end of the scraping process, the program will also output a message indicating how many apartments were scraped

During the scraping process, the program will periodically output progress updates to the console. These updates will include the number of apartments that have been scraped so far, as well as a random joke to keep things light-hearted.

Scraping completed. Scraped [number of apartments] apartments.

# Dockerization

Your program has been dockerized using Docker Compose to simplify deployment and setup. The docker-compose.yml file defines the services that make up your application, including the backend, frontend, Postgres database, and the scraper. Each service is built from a Dockerfile in its respective directory, and can be run using the command docker-compose up in your terminal.

Running the Scraper

To run the scraper, follow these steps:

Make sure you have Docker installed on your machine.

Clone this repository to your local machine.

Navigate to the root directory of the repository in your terminal.

Run the following command to start the scraper and the other services:

docker-compose up

This will start the scraper, as well as a PostgreSQL database for storing the scraped data, a backend server for serving the data, a frontend server for displaying the data, and a pgAdmin web interface for managing the database.

If you encounter a NAT error on Windows, you can try running the following commands in a Windows PowerShell terminal to fix it:

net stop winnat

docker-compose up

net start winnat

Then try running docker-compose up again.

Once the services are running, you can access the frontend by navigating to http://localhost:8080 in your web browser. This will display a table of the scraped data.

Note: It may take a few minutes for the scraper to complete scraping all the pages.