Zillow Real Estate Scraper - Phoenix, AZ

Project Overview

This project demonstrates large-scale web scraping from Zillow to extract real estate listings in Phoenix, Arizona. It utilizes Scrapy with Playwright integration to handle dynamic, JavaScript-rendered pages.

Key Features

- Scrapes 20 pages of listings from Zillow (approximately 800+ properties)
- Extracts structured data including address, price, beds, baths, area, geolocation, lot size, and property type
- Supports automated pagination
- Uses Playwright to render JavaScript content
- Cleaned and processed output saved in `zillow_listings.csv`

Technologies Used

- Python 3.x
- Scrapy
- Scrapy-Playwright
- Playwright (for browser automation)
- Pandas (for data cleaning and analysis)

Setup Instructions

1. Create a Python Virtual Environment

python -m venv venv

Activate it:

- On Windows:

venv\Scripts\activate

2. Upgrade pip

pip install --upgrade pip

3. Install Required Packages

pip install scrapy scrapy-playwright

4. Install Playwright Browsers

playwright install

How the Spider Works

Spider Configuration

```
class ZilspiderSpider(scrapy.Spider):
    name = "zillow_playwright"
    allowed_domains = ["zillow.com"]
```

Defines the spider's name and domain restrictions.

Request Pages

```
def start_requests(self):
    for page in range(1, 21):
        url = f"https://www.zillow.com/phoenix-az/{page}_p/"
        yield scrapy.Request(
            url=url,
            callback=self.parse,
            meta={"playwright": True}
        )
```

Loops through 20 pages and triggers JavaScript rendering with Playwright.

Parse Listings from JSON

```
def parse(self, response):
    script = response.xpath("//script[@id='__NEXT_DATA__']/text()").get()
    data = json.loads(script)
    listings = data.get("props", {}).get("pageProps", {}).get("searchPageState", {})\
                .get("cat1", {}).get("searchResults", {}).get("listResults", [])
```

Extracts listing data from embedded JSON script tag.

Yield Extracted Fields

```
for home in listings:
    yield {
        "address": home.get("address"),
        "price": home.get("price"),
        "beds": home.get("beds"),
        "baths": home.get("baths"),
        "area": home.get("area"),
        "statusText": home.get("statusText"),
        "latLong": home.get("latLong"),
        "livingArea": home.get("livingArea"),
        "homeType": home.get("homeType"),
        "yearBuilt": home.get("yearBuilt"),
        "lotAreaUnit": home.get("lotAreaUnit"),
        "lotAreaValue": home.get("lotAreaValue"),
        "zipcode": home.get("zipcode"),
        "city": home.get("city"),
        "state": home.get("state"),
        "country": home.get("country"),
        "url": f"https://www.zillow.com{home.get('detailUrl')}"
    }
```

Run the Spider

```
scrapy crawl zillow_playwright -o zillow_listings.csv
```

Data Cleaning

After scraping, the data is cleaned using Pandas:

- Fill missing `beds`, `baths`, `area` with 0
- Remove listings with no address or coordinates
- Save to cleaned CSV or display with `print(df.head())`

Deliverables

- `zillow_listings.csv` - Raw scraped data
- Cleaned Pandas DataFrame (viewed or exported)
- GitHub Repository: [insert your link here]
- Video Walkthrough: [insert your YouTube video link here] (Note: Apologies for any quality issues in the video)

Contact

For questions, please reach out via email or GitHub.

Author

Bhavagna Shreya Bandaru Master's in Information Technology, Arizona State University
Email: bbandar5@asu.edu