COS 457
Database Systems
Sophia Priola and Ashley Pike

MAINE PAD FINDER

Data Scraping Documentation – Apartments.com

Overview

This tool programmatically collects rental listing data (street address, city, state, zipcode, unit, rent, square footage, bedrooms, bathrooms, and availability) from Apartments.com search results to support analysis and prototyping for the MainePad Finder application.

How To Run

Requirements:

- Python 3
- Selenium
- Firefox browser
- GeckoDriver

Run the script:

```
py apartment_finder.py
```

At runtime the script prompts for:

- A starting Apartments.com search URL.
- An output CSV file path.

How It Works

1. **Initialize**
   Opens Firefox via Selenium, reads the starting Apartments.com search URL from user input, and initializes `listingData` with the header:

   ["*Street*","*City*","*State*","*Zipcode*","*Unit*","*Rent*","*SqFt*","*Bedrooms*","*Bathrooms*","*Availa*

2. **Collect Listing URLs**
   On each search results page, finds all `a.property-link[href]` elements, adds unique URLs to `urls`, then clicks the `.next` button to move through pages until no further page is available. The initial driver is then closed.

3. **Scrape Each Listing**
   For every URL in `urls`, opens the listing in a new Firefox driver, extracts and cleans the address, and splits it into: Street, City, State, Zipcode. Then determines whether the page is:

   - a multi-unit property or
   - a single-unit / individual listing.

4. **Multi-Unit Listings**
   For listings with multiple units
   - Locates the active tab section and pricing
   - For each unit row, extracts: unit number, rent (numeric), square footage (numeric), number of bedrooms, number of bathrooms, and availability.
   - Appends one row per unit to `listingData`.
5. **Single-Unit Listings**
   For listings identified as single-unit / whole-property:
   - Treats the entire property as one unit (**Unit** field left blank unless included in address).
   - **Rent:** Finds the first element containing a dollar sign, removes "$" and commas, takes the first value if a range is shown, and converts it to an integer.
   - **SqFt:** Finds text containing "sqft" / "Sq Ft" and uses the first numeric value as square footage.
   - **Bedrooms:** Finds text containing "Bed" or "Studio". "Studio" is mapped to 0; otherwise the first numeric value is used.
   - **Bathrooms:** Finds text containing "Bath" and uses the first numeric value.
   - **Available:** Finds text containing "Available" / "Available Now"; sets this field to 1 if it indicates immediate availability, otherwise 0.
   - Appends a single row:

     $$[Street, City, State, Zipcode, Unit, Rent, SqFt, Bedrooms, Bathrooms, Available].$$

Data Format

- **Street** – Street address.
- **City** – City of the listing.
- **State** – State (assumed Maine for this project context).
- **Zipcode** – Postal code.
- **Unit** – Unit identifier (Possibly blank for whole-property listings).
- **Rent** – Monthly rent (integer, first value if a range is shown).
- **SqFt** – Square footage (integer, when available).
- **Bedrooms** – Number of bedrooms (0 for studio).
- **Bathrooms** – Number of bathrooms.
- **Available** – 1 if available now / marked as available, else 0.

Assumptions and Limitations

This scraper assumes that Apartments.com HTML structure matches the selectors used: rent values are present and parseable, bed/bath information includes "Bed"/"Bath", and availability text is clearly labeled.

This implementation is built for a course project prototype. It is not optimized for high-volume or commercial use.