# AffordAbode

Arman Sandher, Gunakaushik Vengalasetti, Rudy Cruz

*Department of Software Engineering, San Jose State University*

*1 Washington Square, San Jose, CA 95192 United States*

[1]arman.sandher@sjsu.edu

[3]gunakaushik.vengalasetti@sjsu.edu

rudy.cruz@sjsu.edu

***Abstract*— This paper proposes a solution to help students and renters find which rental listings in the desired areas a more suited for their needs and allow them to find listings that are more fairly priced. Key Components include the problem statement, market demand, and the separate components of the project.**

*Keywords*— **AI in data handling, Online Scraping Methods, Web Hosting, RedFin Rental Listings**

## I. Introductoin

Rental properties are becoming more and more expensive in the Bay Area as time goes on. More Rental properties are built every year, however, it is not enough to fulfill the demand. With lax restrictions put in place to limit landlords from continually increasing prices, rent for properties has continued to rise. In the Bay Area, where prices continue to rise and where economic stability has worsened, it becomes increasingly more important to ensure that people get a bang for their buck.

This becomes more and more prominent for college students. Oftentimes, on-campus housing is unavailable either due to not having enough available space, so students have to rely on finding their own place to rent by themselves, or with other students. Eventually, having to find housing while going to school becomes a challenge.

Our solution offers a remedy to this challenge. AffordAbode offers to list rentals in a user's desired city, zip code, or neighborhood with all the information that regularly comes with rental sites, with the added benefit that the listings are rated. This rating is based on multiple factors, including price, bed, bath, and area. Using this information, our AI model comes up with an expected price. Depending on the price difference, it grants the listing a score from 0 to 130, where anything under 75 is considered a bad deal, and anything over 100 is considered a good deal. Our solution also offers the ability to enter any interests, location of work or college, and transportation requests into a profile for the user to help further narrow down a search.

With all of this, AffordAbode becomes an increasingly easier way to find a rental listing that is not overpriced. Students can find what they need in the area they are looking for.

## II. Problem Statement

The issue with today's with finding rental living spaces in the Bay Area is with too much demand and not enough supply. This leads to everyone having to run around and find a rental that fits their needs. Students have the added issue that many do not have the luxury to spend a long time searching and evaluating each listing. On top of this, there may be similar listings, and without a tool to compare them, it is difficult to determine what might be a better fit for their needs.

## III. Market Demand

On discussing this with multiple students, we concluded that it is something that students would like to use. With the housing market in the Bay Area getting more expensive, the demand for a product that can evaluate a listing from an objective standpoint is becoming more prominent. Students already have a lot of pressure on them, and having a solution to conveniently lessen the burden of picking a rental listing to look into further becomes more and more valuable.

## IV. Front End

The front end is used with NextJS and React. It is designed to be simple and easy to use. The user is first introduced to the landing page, after which, they are prompted to create an account or sign in with an existing account, or use Google Sign-In. They will be led to the listings search page, where the user can search using a city, zip code, or neighborhood. Listings are currently only scraped from RedFin. All listings under $10,000 per month are displayed on screen. The 10k limit is put in place because it is not reasonable for a student, or multiple students, to be paying rent over this amount.

While on the listings page, a user can search for a specific property using the search feature on this screen and filter the listings further with filters on the sidebar. Displayed on the listings card are the first picture displayed on RedFin for this listing, the price, bedrooms, bathrooms, square footage, a link to the original listing, a price comparison of the expected price and the listed price, and a score based on the difference between these two prices. Clicking on the listing brings out a flyout menu that gives the user more information on the listing.

### A. Scraping

RedFin was our chosen website to scrape. Other websites like Zillow and Facebook did not allow scraping, have bot protection, and do not offer their API for use. Other sites like RentCast.io offered listing as well, but required a paid license for anything past the first ten listings, and did not offer any pictures. We settled on RedFin. While it does not have an API that can be used by the public, it allowed us to scrape their website with little interference. We are also open to the opportunity to use other sites to expand our search and provide a more complete and comprehensive selection of rental listings.

Because RedFin uses JavaScript, the project needed to use something compatible. The project uses Scrapy-Selenium to achieve this. The scraper is initialized by opening up the page, mimicking human actions, looking up the input the user had entered in the search page, and getting the link that comes up. The link would end up looking like:

redfin.com/city/[randomly-assigned-city-number]
/[city-name]/
OR
redfin.com/zipcode/[zipcode]

Our scraper was designed to append "/apartments-for-rent/" and "page-[page-number] to iterate over all rental listings over all pages. Unfortunately, RedFin only displays 350 listings at once, even if there are over 1000 in any particular place.

It scrapes all important information that we need and stores it in a JSON that is later read from and used for calculations.

## V. Back End

## VI. Original Contributions

### A. Arman Sandher

Designed and implemented the scraper for the project. Designed the front end. Implemented all filters and search APIs. Implemented teammate's model to display the listing scores. Implemented flyout menu, and displayed all information stored in the JSON created by the scraper. Created the presentation and the majority of the essay for this project.

### B. Gunakaushik Vengalasetti

### C. Rudy Cruz