House Search Engine

http://housing-site-host.s3-website-us-west-2.amazonaws.com/

Table of Contents

Introduction

House hunting can be very time consuming, especially at the “hot” area, e.g. the San Francisco Bay Area. Usually, people need to look through the popular Real Estate Websites, such as Redfin or Zillow. These website contains a lot of information about the houses on the market, but sometimes users will be overwhelmed by the huge amount of information and spend too much time looking through each houses.

The House Search Engine we designed is to help users to refine their searching and obtain the necessary information about the houses that match the searching criteria.

Operation

The UI of the House Search Engine contains three portions: (1) Searching Parameter Selection (fig. 1), (2) Searching Result Display (fig. 2) and (3) New Listing Examples (fig. 3).

1. Searching Parameter Selection

In the Searching Parameter Selection section, users need to input the below information:

- Street
- City
- Zip Code
- Property Type
- Price Range
- # of Bedroom
- # of Bathroom

The input will be used to compare with the database, which is the house information crawled from Redfin. Only the houses that match all the input criteria will be provided to users. If the parameters are left empty, these parameters will not be used in the searching.

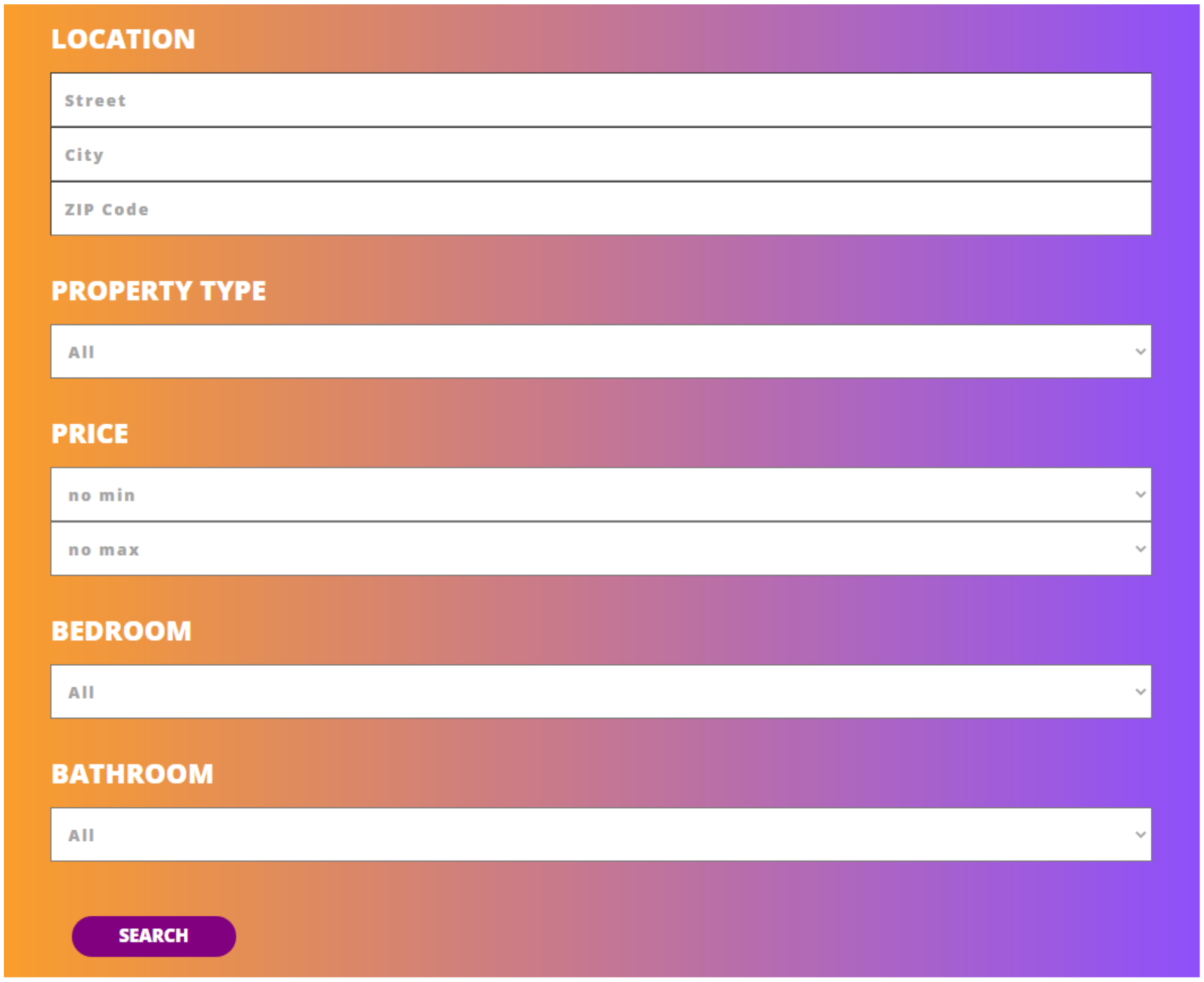
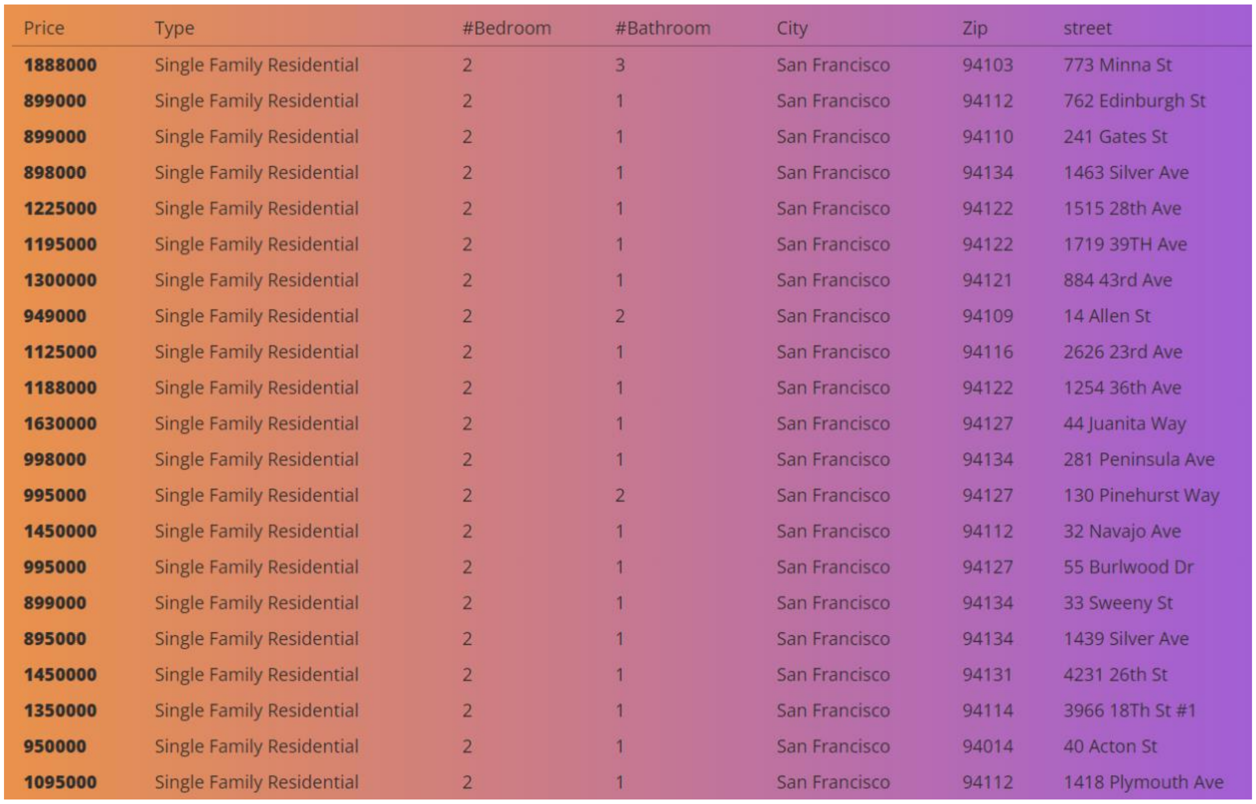


Fig. 1 Searching Parameter Selection section.

2. Searching Result Display

After users submit the search, the matched results will be displayed in the Searching Result Display section as a table. The table contains each parameter in one column.

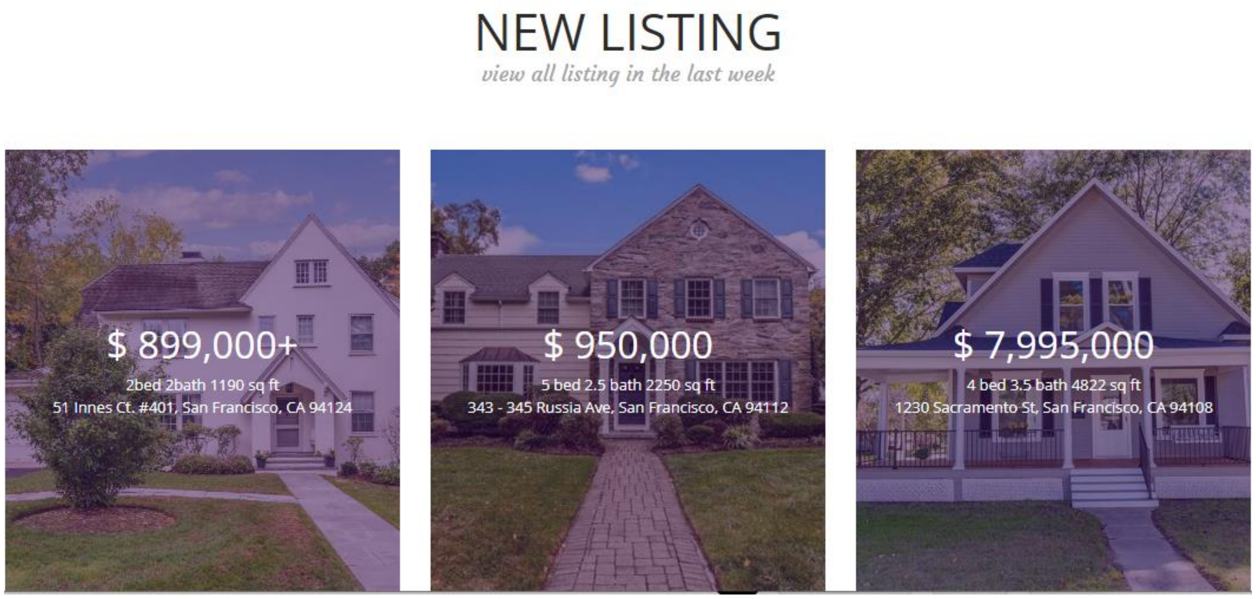


Price	Type	#Bedroom	#Bathroom	City	Zip	street
1888000	Single Family Residential	2	3	San Francisco	94103	773 Minna St
899000	Single Family Residential	2	1	San Francisco	94112	762 Edinburgh St
899000	Single Family Residential	2	1	San Francisco	94110	241 Gates St
898000	Single Family Residential	2	1	San Francisco	94134	1463 Silver Ave
1225000	Single Family Residential	2	1	San Francisco	94122	1515 28th Ave
1195000	Single Family Residential	2	1	San Francisco	94122	1719 39TH Ave
1300000	Single Family Residential	2	1	San Francisco	94121	884 43rd Ave
949000	Single Family Residential	2	2	San Francisco	94109	14 Allen St
1125000	Single Family Residential	2	1	San Francisco	94116	2626 23rd Ave
1188000	Single Family Residential	2	1	San Francisco	94122	1254 36th Ave
1630000	Single Family Residential	2	1	San Francisco	94127	44 Juanita Way
998000	Single Family Residential	2	1	San Francisco	94134	281 Peninsula Ave
995000	Single Family Residential	2	2	San Francisco	94127	130 Pinehurst Way
1450000	Single Family Residential	2	1	San Francisco	94112	32 Navajo Ave
995000	Single Family Residential	2	1	San Francisco	94127	55 Burlwood Dr
899000	Single Family Residential	2	1	San Francisco	94134	33 Sweeny St
895000	Single Family Residential	2	1	San Francisco	94134	1439 Silver Ave
1450000	Single Family Residential	2	1	San Francisco	94131	4231 26th St
1350000	Single Family Residential	2	1	San Francisco	94114	3966 18Th St #1
950000	Single Family Residential	2	1	San Francisco	94014	40 Acton St
1095000	Single Family Residential	2	1	San Francisco	94112	1418 Plymouth Ave

Fig. 2 Searching Result Display section

3. New Listing Example

The last section is the New Listing Examples, which displays 3 examples of newly listed on Redfin. Currently, this section does not have extra functionality.



NEW LISTING

view all listing in the last week

$ 899,000+
2bed 2bath 1190 sq ft
51 Innes Ct. #401, San Francisco, CA 94124

$ 950,000
5 bed 2.5 bath 2250 sq ft
343 - 345 Russia Ave, San Francisco, CA 94112

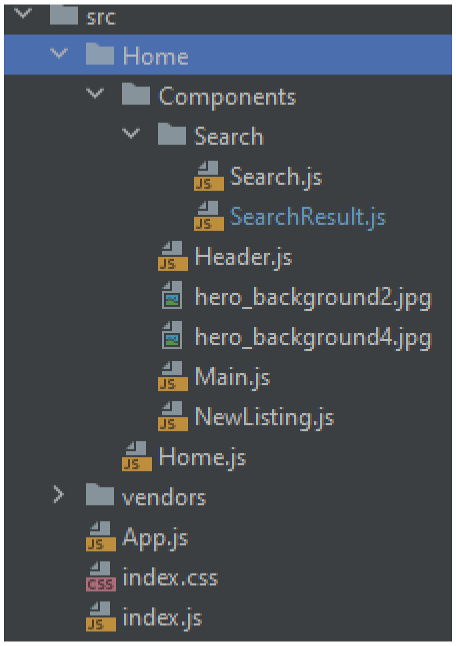
$ 7,995,000
4 bed 3.5 bath 4822 sq ft
1230 Sacramento St, San Francisco, CA 94108

Fig. 3 New Listing Examples section

Code Implementation

1. Front End – Xinyuan and Hao

The front end website is created by React Javascript. The different sections on the websites are controlled by different files: Header.js, Main.js, Search.js, SearchResult.js and Newlisting.js (Fig. 4).



```
src
  Home
    Components
      Search
        Search.js
        SearchResult.js
      Header.js
      hero_background2.jpg
      hero_background4.jpg
      Main.js
      NewListing.js
    Home.js
  vendors
  App.js
  index.css
  index.js
```

Fig. 4 Front End Code

Header.js displays the logo on the header

Main.js: displays the background image and the slogan

Search.js has three important functionalities:

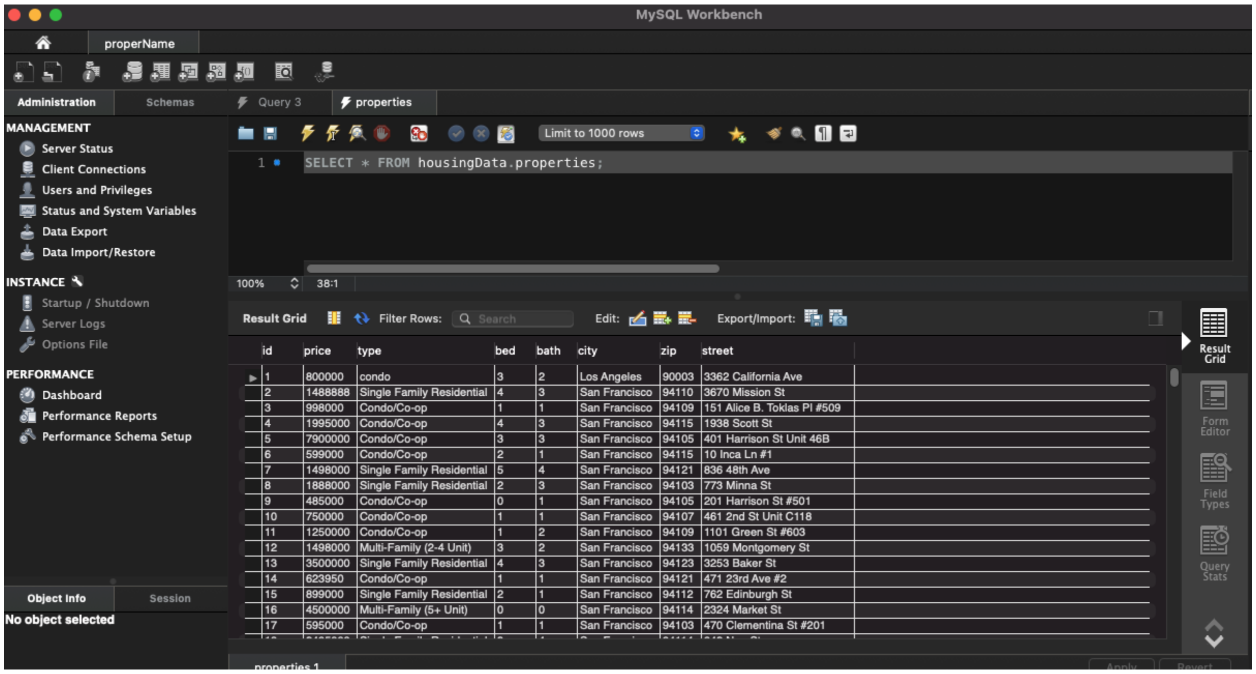
- Collects the user inputs
- Perform the comparison with database
- Pass the searching result to SearchResult.js for displaying

SearcheResult.js gets the result from Search.js and displays the result as a table

Newlisting.js displays the examples of newly listed houses

2. Back End - Hao

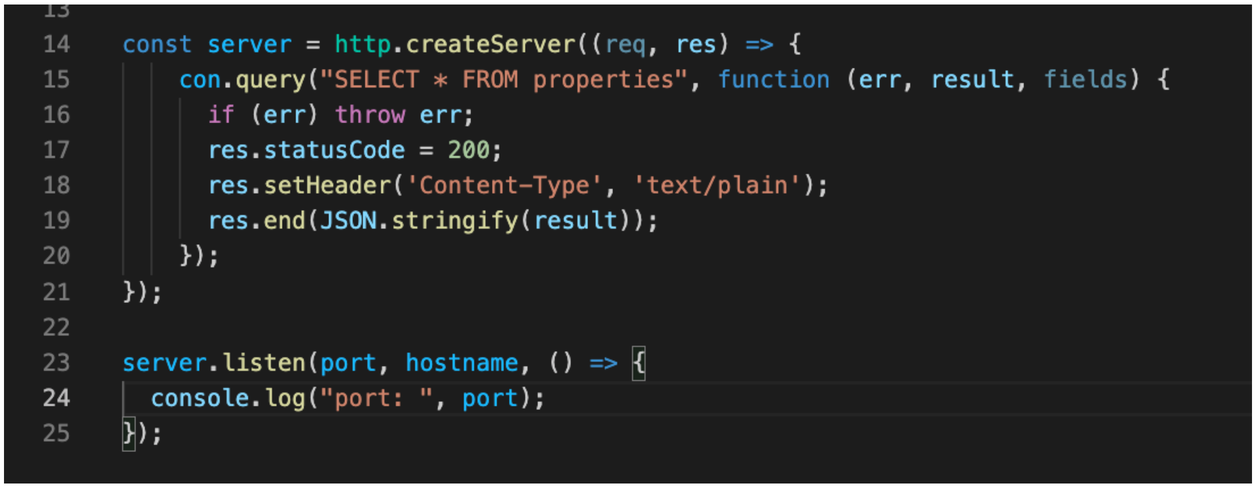
A MySQL database was set up using AWS RDS to store the data. The table has eight fields: *id, prince, type, bed, bath, city, zip, street.*



```
SELECT * FROM housingData.properties;
```

id	price	type	bed	bath	city	zip	street
1	800000	condo	3	2	Los Angeles	90003	3362 California Ave
2	1488888	Single Family Residential	4	3	San Francisco	94110	3670 Mission St
3	998000	Condo/Co-op	1	1	San Francisco	94109	151 Alice B. Toklas Pl #509
4	1995000	Condo/Co-op	4	3	San Francisco	94115	1938 Scott St
5	7900000	Condo/Co-op	3	3	San Francisco	94105	401 Harrison St Unit 46B
6	599000	Condo/Co-op	2	1	San Francisco	94115	10 Inca Ln #1
7	1498000	Single Family Residential	5	4	San Francisco	94121	836 48th Ave
8	1888000	Single Family Residential	2	3	San Francisco	94103	773 Minna St
9	485000	Condo/Co-op	0	1	San Francisco	94105	201 Harrison St #501
10	750000	Condo/Co-op	1	1	San Francisco	94107	461 2nd St Unit C118
11	1250000	Condo/Co-op	1	2	San Francisco	94109	1101 Green St #603
12	1498000	Multi-Family (2-4 Unit)	3	2	San Francisco	94133	1059 Montgomery St
13	3500000	Single Family Residential	4	3	San Francisco	94123	3253 Baker St
14	623950	Condo/Co-op	1	1	San Francisco	94121	471 23rd Ave #2
15	899000	Single Family Residential	2	1	San Francisco	94112	762 Edinburgh St
16	4500000	Multi-Family (5+ Unit)	0	0	San Francisco	94114	2324 Market St
17	595000	Condo/Co-op	1	1	San Francisco	94103	470 Clementina St #201

These data are converted using a node server to JSON format.



```
const server = http.createServer((req, res) => {
    con.query("SELECT * FROM properties", function (err, result, fields) {
      if (err) throw err;
      res.statusCode = 200;
      res.setHeader('Content-Type', 'text/plain');
      res.end(JSON.stringify(result));
    });
});

server.listen(port, hostname, () => {
  console.log("port: ", port);
});
```

Initially, we are going to use this as an API server. But we found storing the JSON file in an AWS s3 bucket and serve it is more efficient. So we put the converted data in AWS s3 buckets and served it from there.

The front end is able to fetch the JSON file directly from the s3 bucket.

3. Data collection - James

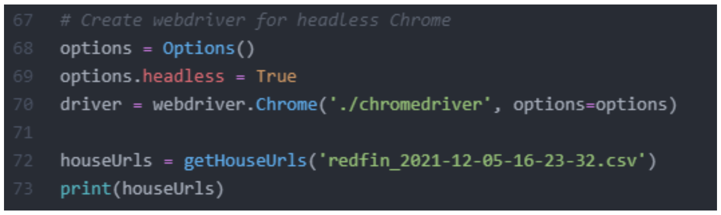
The data was collecting using a python script to scrape the data from the redfin website. The script requires a version of python that supports the BeautifulSoup package. This can be done by running a Python 3.7 environment using Anaconda.

The script then pulls a list of urls from Redfin. When searching up houses on redfin, it is possible to download a csv list of houses that contain their redfin url with the Download All button. In this case, we downloaded all of the housing data from the San Francisco area.

The scraper also needs the latest version of chromedriver, which can be downloaded from https://chromedriver.chromium.org/downloads. Once chromedriver and the list of urls has been placed in the same folder as the scraper, the script is run in a terminal with python.

The script goes to each url listed in the downloaded csv and looks at specific locations in the html of the visited page to gather numerical and text data related to the house. This data is then saved to output files.

This generates 2 files. One file is called textData.txt, which contains all the text data related to a house on redfin, along with its url. The other is called sqldatabase.csv, which contains numerical data from the house and is used to populate the database.



```
# Create webdriver for headless Chrome
options = Options()
options.headless = True
driver = webdriver.Chrome('./chromedriver', options=options)

houseUrls = getHouseUrls('redfin_2021-12-05-16-23-32.csv')
print(houseUrls)
```

Fig. 5 Downloaded file and chromedriver in the script

Future Improvement

The current version has not fulfilled all our initial design due to the limited the time. For example, the original plan is to add weight on different searching parameters to further refine the searching. The future improvements and how to implement them are proposed in the below list:

1. Add weight to each parameter:

   Users can rate each parameter (1~5), and the weight will be calculated by **rate/total rate.**

   Ex: Price rate = 3, # of bathroom rate = 2 and # of bedroom rate = 2. The weight will be Price weight = 3/(3+2+2), # of bathroom weight = 2/(3+2+2) and # of bedroom weight = 2/ (3+2+2).

2. Add more house features:
   More parameters can be added, such as house size or year built.
3. Result displayed with url:
   Current result is displayed as a table with all the parameters. The url will be added in the result, so that users can directly link to the Redfin website to check more details about the interested houses.