Time	~ 1 day
Learning Goals	• Understand the various ways of implementing search • Implement a filtered search using scopes • Use gem pg_search for full-text search • Learn how to push a new feature into an app that is already in production

Hurrah! Your app is already on the cloud! Let's now imagine that there are people using your app and have given you some valuable feedback on how to improve it. From user feedback, you decided to enhance the user experience in terms of searching for relevant information. Just like what AirBnB does. The website is simple, beautiful and gets right to the point of asking you where are you going. Just like how Google took over as the dominant search engine, both AirBnB and Google were amazing sites powered by search at its core. Let's try to do something similar!

Objectives

We would like to emulate AirBnB's search function by implementing the following features:

1. Implement search filters using

- Price Range
- City
- Amenities
- Number of bedrooms
- Number of bathrooms

Feel free to refer back to AirBnB's way of doing the filters. Take note of which ones are using a text search field, which ones are using a drop-down menu, which ones are using checkboxes and radio boxes, etc. To get started, check out this article on Search and Filter Rails Models. You may also want to some research on keywords such as "scopes" and "dynamic search scopes". Please also use form_for for the search form.

ps: please do not use any gems here for this exercise. We would like you to learn how to use scopes to perform searching, filtering and sorting.

2. A full-text search

Using "Listings" as the model to search from, think of which fields that you would like to include in this search bar. Do you want to conduct a search over all the fields or only a selected few? Why or why not?

To get started, check out the documentation of the gem pg_search. For the interested, here's a nice article on using Postgres for full-text search without using a gem: http://rachbelaid.com/postgres-full-text-search-is-good-enough/.

Don't forget to checkout a new branch for this feature! Once you can perform a search, congrats! You're done with this assignment. You can proceed to commit the changes, raise pull request and merge back to master branch. Don't forget to push it to Heroku too since you have already deployed your app!

Extra Reading: Many Ways of Searching

Now, for many new apps we build, we typically use Postgres Full-Text Search to begin with. This is often a good place to start before you get fancy. It's usually a good idea to not have to deal with extra infrastructure for the cost, time and pain of managing more things until you really have to. However, the tricky bit to this way of performing a search is speed and flexibility. And like many developers, you will naturally want to play with new technology.

Another way to perform search is to use a dedicated search engine (no we don't mean Google). These are typically run on your own server, you pass data you want to be searchable to the search engine and in future, you can perform a search against it. Think of it like you needed a background job app like Sidekiq, now you need a dedicated search engine. These search engines' key advantage is speed.

Two popular search engines are Solr and Elastic Search. They are both open-source software, just like Ruby-on-Rails, that you can use. Here are some further readings:

- Elastic Search
- Searchkick
- Heroku SearchBox
- Getting Started with ElasticSearch on Rails

Another way is to use a hosted solution. These are usually expensive solutions but if you rather not deal with managing anymore extra infrastructure, this is a good way. Algolia is a popular one.

When deciding on which search engine to use, here are some things to think about:

1. Think about their individual capabilities? What are their pros and cons?
2. Consider thinking about the complexity of deploying in production (for example Heroku).
3. Consider thinking about the setup complexity.
4. How to test the search engine after deployment?

Once you consider enough rationality, you can then make a decision on which technology to use.