Homeloop Web Dev. Test

Instructions

You are given one week to complete this exercise.

*We do **not** expect you to spend all your evenings for the entire week on this exercise, we just want allow you enough flexibility to fit your schedule.*

If for any reason you could not complete the exercise on time, no problem. Send us what you managed to achieve, along with a short explanation of what you would have liked to have added if you had more time; we'll discuss further during the interview.

Also, do not hesitate to ask us if you have any questions! Good luck.

A tool for Visit Analysts

Scenario

In Homeloop's operational team, we have *Visit Analysts* who visit properties we are considering buying. To make the visit as efficient and useful as possible, our analysts come prepared with a pre-analysis of the property. The data collected during the visit then allows *Valuation Analysts* to prepare the offer we will make to homeowners.

Past transactions in the neighborhood are an important Part of the "context" needed by our Visit Analysts. Fortunately, we have a database that stores all the previous real-estate transactions across France. Visit Analysts need to be able to exploit this database easily and anywhere.

Your objective is to develop a webapp, functional on both desktop and mobile devices, that would allow them to do so.

Material

The homepage of the app should prompt the Visit Analyst its email (for usage tracking), and on click of the submit button geolocate them, save the query to a model, then show them the neighbouring transactions (within a radius of 1km).

To do so you will need:

- To create a Ruby on Rails project, with appropriate models as described below
- To choose and use a SQL technology
- To find a way to do geolocation

The webapp should contain two pages:

- **Home Page:** a form with an “email” field, and a “submit” button; clicking on it should bring you to the results page and update the latitude, longitude, and results count fields
- **Results Page:** a table listing the transactions around you with a link back to the homepage

The models should be the following:

- **Query (starting empty)**
  - Email
  - Latitude
  - Longitude
  - Result count
- **Transaction (pre-populated with data before the first user accesses the app)**
  - Latitude
  - Longitude
  - Price
  - # of rooms
  - Square meters
  - Date

Some example transaction data are provided in *transactions_examples.txt,* feel free to complete it with more data for testing purposes.

Screen designs are provided in the *screens* folder. **We do not expect you to do pixel-perfect front-end integration,** it is more here as a global design guideline. Also, the integration of the Google Maps on the results page is optional**.**

What we are testing

This exercise is purposefully left quite open, and we expect you to make your own design decisions (not only for the frontend, but also for the architecture and choice of technologies/libraries for the backend).

The aim is to reproduce a real work situation. **Therefore we want you to be as efficient and elegant as possible when doing this project.** This means that you can - and should - use resources found on the web or even your past projects as much as possible.

Basically, we expect you of course to build a functional tool, but also to find the right balance between time spent vs. crafting quality code.