CS-373 IDB Technical Report: Group 1

Motivation

The motivation behind the project is to allow users to find jobs based on geographical location and/or price and simplify the housing search for those looking to move, especially if they have a budget or preferred location.

User stories

User Story: Search for cities by job

"One feature that would be very useful would be the ability to filter/search cities by a specific job title(s). This way, a user could find all the cities they could work in and pick one to live in based on information on each city. Right now, you can search for jobs which contain information on what city they are located in, but it's extremely likely that multiple jobs will be located in one city, so it isn't very useful for finding what cities you could potentially work/live in."

This is a useful idea! It should be doable because we can get location data for each job posting and we can search for job titles in the API. We will implement this in Phase 3.

User Story: View City Job Demographics

"I'm a recent graduate who is looking to move somewhere new, but don't know what kind of job I want. Currently I can search for individual jobs in a city, but ideally I want to be able to view the top jobs in each city. Having the ability to view the top X jobs/industries in a city by quantity/pay/rating would allow me to get a better sense of where I want to live."

This idea seems to be challenging but worthwhile. We should be able to get information on the number of jobs within a certain industry in a city, as well as information about pay within each job and industry, which we can then either display on the city page or on the jobs page. We will have this implemented by Phase 3, once we have implemented sorting and searching.

User Story: City budget search

"Hi, I already have a remote job and I need to know what cities fit my budget. Could there be some sort of price search which will return the closest cities and apartments as well as similar ones? Maybe a range or average cost?"

This idea seems redundant with the ability to sort cities by budget score, then sort apartments by price within that city, which will already be implemented in Phase 3.

User Story: Implement job search by apartment

"*I'm a very materialistic person, so I've decided to search for my dream apartment first. I'll need to come up with the money somehow, so I'll need to get a nearby job. Implement the reverse of your proposal: given an apartment, show me all the possible jobs in the city that I'd have to get in order to financially support my dream of living in this apartment.*"

This idea results in the list of jobs being sent back to the user being the same as that of the list of highest paying jobs, up until the point at which the income is not enough for the apartment. However, we will try to implement a reverse search in Phase 3.

Filter jobs by several categories

"I'm a recent college graduate looking for a job and I would like to be able to quickly find the best jobs that I am suited for. Specifically, being able to filter by experience requirements to find something entry level would be a huge save on time and on top of that filtering by location and salary would be great as well."

This is a great idea! Using multiple filters would absolutely be useful when searching for jobs that fit a specific skill set and situation. This will be implemented in Phase 3, when sorting and searching are implemented.

RESTful API

https://documenter.getpostman.com/view/23607622/2s83tFHBCK

Models

City - This model provides all around information about living in a given city, including: rating, budget score, safety score, ability to get around on foot, bike, or public transit, images, and various tags on what the city is known for.

Apartment - This model provides information about apartments in a given city, including: location, rating, number of reviews, phone number, and images.

Job - Provides information about any job given a job title or industry, including: company, expected salary range, location, job description.

Tools

React: JavaScript library for building user interfaces, powers the front-end
React-Bootstrap: CSS framework built on React, simplifies integration of the frameworks
React Router: library for routing in React, enable navigation between pages
Material UI: provides React components that implement Google's Material Design standard
NameCheap: domain name registrar, allows purchasing and managing domain names
AWS Amplify: platform for deploying and hosting web applications, supports continuous integration and continuous deployment
Discord: online communication platform
Visual Studio Code: code editor
Zoom: online communication platform
GitLab: git software that allows for managing version control and continuous integration/ continuous deployment
Postman: API platform for documenting, building, and using APIs
Axios: HTTP client library, used to access APIs from React application
Node Package Manager (NPM): manages installation of node packages

Data Sources

RoadGoat: get information about cities
Google Maps: get maps showing where apartments are located and travel time
Yelp: get information about apartments
Adzuna: get information about jobs
GitLab: version control and issue tracking, gets information about commits and issues

Hosting

First, we registered the domain *geojobs.me* on NameCheap. For hosting, we used a tool called AWS Amplify, which is part of Amazon Web Services. We had to set up SSL/TLS with NameCheap and Amplify. This allowed us to connect different branches of our GitLab repository to different deployments, which we could then associate with different CNAME records on NameCheap. We set dev.geojobs.me to point to the development branch and geojobs.me (and www.geojobs.me) to point to the main branch.

Challenges

It was hard for us to get started with React and Bootstrap, so there was a learning curve before we could really do much on the front-end of the site. We had difficulty finding apartment APIs that gave us the information we wanted: for example, we tried Zillow, but it wasn't quite working for what we wanted to do. At some point, the AWS Amplify project was not redeploying when our main branch was updated, so we had to manually redeploy instead of it happening automatically; identifying this issue took a lot of time though, as we were not familiar with the intricacies of AWS Amplify.