eRestaurant rating app

Potential names:	(H)ate it	(R)ate it
	Ate or hate	Ate n hate

Ideas

- Letterboxd, but for restaurants
- Tinder like swiping features, to like

A more personalised way to rate and manage the restaurants you and your friends have gone to

Pages

Home

Hot new restaurant

Diary

Explore

User

Search

The following tags are supported in open street maps:
Sub tags could be found with the cruise tag, this is a good idea for a future update.

bar, biergarten, cafe, fast_food, food_court, ice_cream, pub, restaurant

Casual dining
Fine dining
Food trucks
Fast food
Café
Buffet
Ghost restaurant
Casual
Pop-up restaurants
Pub
Pizzeria
Bistro
Family Style
Bakery
Family restaurant
Pop-ups
Cafe
Cafeteria
Deli
Contemporary casual
Fast food or quick service
Food truck or stand
Restaurants
Buffet style

Key Technologies

Frontend	Angular
middleware	
Backend	Postgresql Nginx Express Docker

Git for version control: https://github.com/Nicholas-Parise/Restaurant-Rating

Server Considerations

Oracle offers an **always free tier** that is pretty good for what you get. I already have a VM running that hosts my portfolio, which does mean that I will have less storage in this project. The max storage available is 200gb and the minimum storage size is 47 gb.

Oracle Arm Compute Instance

	CPU	Ram	Boot Storage	Block storage
Database	2 core	12 gb	47 gb	58 gb
resWebVM	2 core	12 gb	47 gb	0 gb

Resizing boot storage seems like it is a paid option but I can make a block storage or any size so I will choose a block storage of the remaining size I have which is 59gb (200 - 47*3). To be safe it will be 58GB.

The VM's will share a subnet to ensure a fast connection, the oracle website says that very fast speeds can be obtained with this method.

The unknown is if I want to expose the database server to the open internet, my original idea was to use this database server as both the database and a CDN to hold the images so I could have a request such as: img.'nameOfWebsite'.com/img/'specificImage'. This seems like the most sensible option.

Open street maps

Getting data from open street maps was a huge pain and not very intuitive at all. The docs are sub par and figuring out the simplest things was very difficult. However now that the grunt work is done it should be a lot easier to add data to the database.

Data download:

data is freely available at many sources

Complete download can be found here:	https://planet.openstreetmap.org/
by country at this website	https://download.openstreetmap.fr/extracts/

Processing data:

The data is in the form of a .osm.pbf file which is a highly compressed file format that needs special tools to extract required data. These extracts are very data dense and in this application most can be ignored and not used.

There are two tools open street maps seems to recommend the most:

Osmfilter: a cli tool to filter objects and output them in a .osm file format for geo analysis.
osm2pgsql: a cli tool that can filter objects and output data into a psql database.

For my application I went with **osm2pgsql** as I already needed the data in a psql database and this saved me the multiple steps.

Osm2pgsql

This is an open source project with extensive but very hard to implement documentation, there is also not much online on how to use this tool but thankfully they have example projects on the main site.

This article helped: https://learnosm.org/en/osm-data/osm2pgsql/
These too: https://osm2pgsql.org/examples/poi-db/
https://osm2pgsql.org/examples/road-length/

The following steps is how I set up the environment, this was complicated as I was performing these tasks on windows.

1. Installed psql on my device and included the PostGIS extensions. Added an account with my device name
2. Downloaded osm2pgsql

3. Include psql in the system path
4. Wrote my lua script to extract data
5. Ran osm2psql

This is the command I ran when I performed my analysis:
osm2pgsql --password -d pois -O flex -S pois.lua canada-latest.osm.pbf

This command asks for the database password and runs the lua script

Lua script

Tags is a json data type, and the values can be extracted by the following code:

```
a.country = object.tags["addr:country"]
a.brand = object.tags.brand
```

The data is then added to the database row by row. This works perfectly except for latitude and longitude. I could not find a way to extract a latitude and longitude from the data in lua, so instead it can be done with the following query. I'm sure there is a way to do this all inside the lua script on creation but nothing I tried worked.

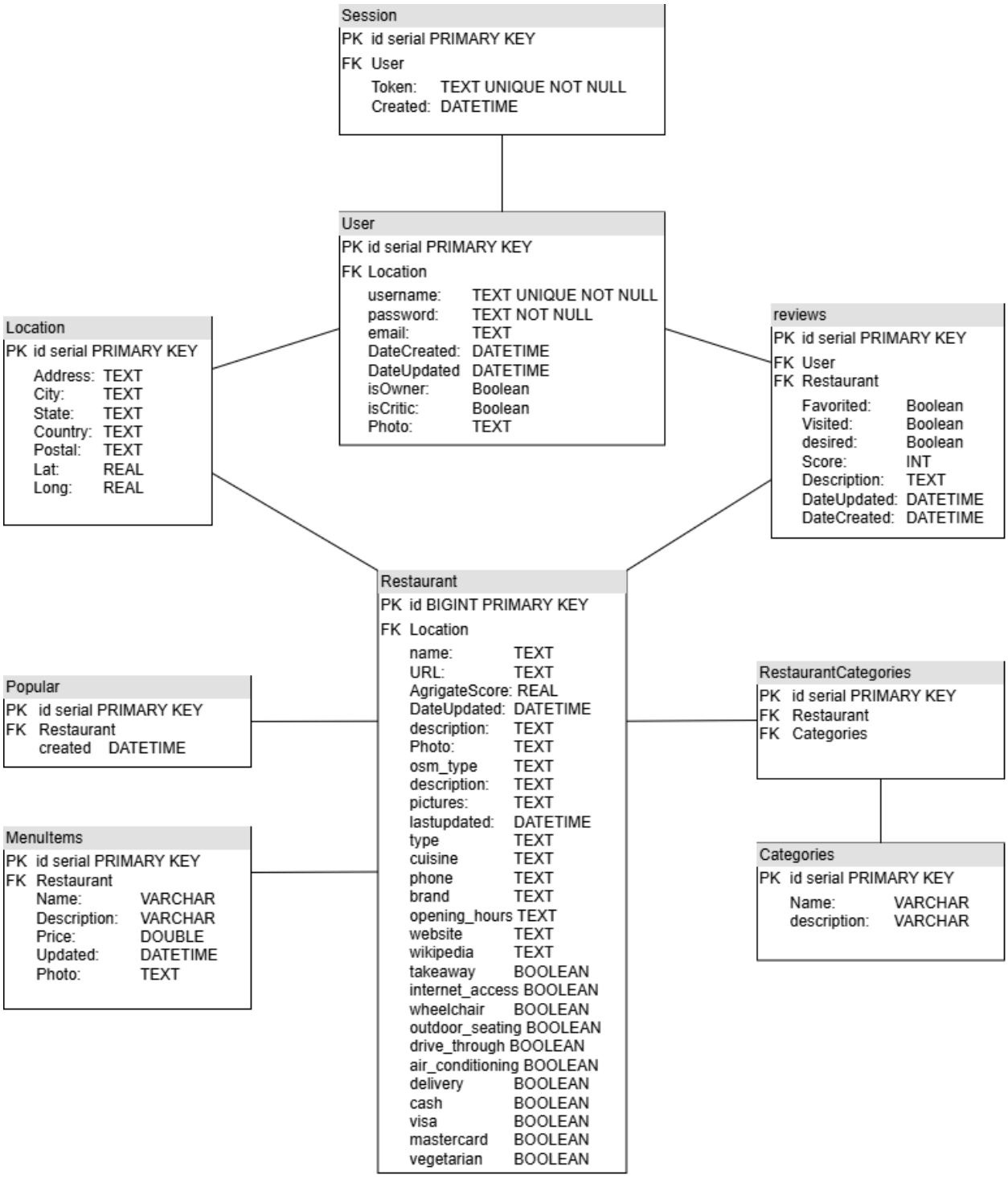
```
BEGIN;
UPDATE pois
SET
   lon = ST_X(ST_Transform(geom, 4326)),
   lat = ST_Y(ST_Transform(geom, 4326));
COMMIT;
```

Notes - size and scale

The canada dataset has **~½ billion nodes** and is around **~5gb in size**, after extracting all the restaurant data the **final size is 70 mb** and only 70,000 rows or a reduction of 98.6%. If this trend continues then the full earth file that is **78 GB** should only be around **1.1 GB**.

According to statistics Canada as of 2021 there are **97,000 restaurants** in Canada and this data set **only captured 70,000** which is an **error of almost 30%** off the true number, since open street maps is community driven something like this is to be expected. The hope is that users can submit missing restaurants and amend false entries in the dataset. If this trend continues then the accuracy of the entire world dataset is very questionable.

Database Schema



SQL create query

<table>
<tr>
<td rowspan="2">CREATE TABLE restaurants (
id BIGINT PRIMARY KEY,
location_id INTEGER
REFERENCES locations (id),
name TEXT,
description TEXT,
pictures TEXT,
lastupdated TIMESTAMP,
type TEXT,
cuisine TEXT,
phone TEXT,
brand TEXT,
opening_hours TEXT,
website TEXT,
wikipedia TEXT,
takeaway BOOLEAN,
internet_access BOOLEAN,
wheelchair BOOLEAN,
outdoor_seating BOOLEAN,
drive_through BOOLEAN,
air_conditioning BOOLEAN,
delivery BOOLEAN,
cash BOOLEAN,
visa BOOLEAN,
mastercard BOOLEAN,
vegetarian BOOLEAN);</td>
<td>CREATE TABLE locations (
id SERIAL PRIMARY KEY,
addr TEXT,
city TEXT,
province TEXT,
country TEXT,
postalcode TEXT,
lat REAL,
lon REAL);</td>
<td>CREATE TABLE menuItems(
id SERIAL PRIMARY KEY,
restaurant_id BIGINT
REFERENCES restaurants (id),
name TEXT,
description TEXT,
lastupdated TIMESTAMP,
price real,
pictures TEXT);</td>
</tr>
<tr>
<td>CREATE TABLE reviews(
id SERIAL PRIMARY KEY,
restaurant_id BIGINT
REFERENCES
restaurants(id),
user_id BIGINT
REFERENCES users(id),
favorited: Boolean,
visited Boolean,
desired Boolean,
score INT,
description TEXT,
dateUpdated TIMESTAMP,
dateCreated TIMESTAMP);</td>
<td>CREATE TABLE users(
id SERIAL PRIMARY KEY,
username TEXT UNIQUE NOT
NULL,
password TEXT NOT NULL,
email TEXT,
datecreated TIMESTAMP,
dateupdated TIMESTAMP,
isadmin BOOL,
iscritic BOOL,
isowner BOOL);</td>
</tr>
<tr>
<td colspan="2">CREATE TABLE restaurant_cats(
id SERIAL PRIMARY KEY,
restaurant_id BIGINT REFERENCES restaurants (id),
category_id INTEGER
REFERENCES restaurants (id),
created TIMESTAMP);</td>
<td>CREATE TABLE sessions(
id SERIAL PRIMARY KEY,
user_id integer REFERENCES
users(id),
token TEXT UNIQUE,
created TIMESTAMP);</td>
</tr>
<tr>
<td>CREATE TABLE popular(
id SERIAL PRIMARY KEY,
restaurant_id BIGINT
REFERENCES restaurants (id) ,
created TIMESTAMP);</td>
<td>CREATE TABLE categories(
id SERIAL PRIMARY KEY,
name TEXT,
description TEXT,
created TIMESTAMP);</td>
<td></td>
</tr>
</table>

Appendices