Description:

MyBnB is a simple reproduction of the AirBnB system that allows users to rent properties to stay at for certain amounts of time as well as list properties they themselves own for others to rent. It is implemented in Java embedded SQL using MySQL and consists of a simple text-based UI that allows a user to create an account and use that account to add/edit/book a listing. The program also allows users (any user at this point) to view reports based on data in the database, from number of listings per country to what are the most popular words in a listing's comments.

One conceptual encountered was how to maintain information provided in a segmented fashion, as to reduce the amount of redundancy in the system, while also ensuring that when we need certain information, we don't have to go through the entire system. The issue didn't have a quick and easy solution, and the end result was that the database evolved throughout the project, from its original design.

This is not strange as, while most requirements for a project might be provided at the beginning, a perfect solution for each situation cannot be developed until there is enough implementation to see what else might be required and what changed need to be made.

One large change that was made from the beginning to the end was the implementation of views in the sql database, stemming from requirements for the reports and when searching through listings.

Another minor change was in storing a listing's amenities, where an ENUM of possible amenities was first used, but was later changed to a relation between the listing and a table of amenities with an amenity id they could be corelated to. This reduced string comparisons to int comparisons and allows amenities to be added later by database insertion; in the program, these amenities are later retrieved as options for the end user when they add a listing, all of which increased the flexibility of the system.

Assumptions:

Some assumptions made are as follows:

- Users can both rent and host properties, as long as it is not their own property.
- Card numbers are not unique, which means many users can have a certain card number.
- If a user deletes their account, all associated information, from property listings to credit information, to comments are all removed from the database as well, not only for security reasons but also to maintain the database. The bookings however, remain until the listing is deleted.
- To continue the above, when a user deletes a listing, the same thing happens, where their bookings and listing comments are deleted. This is because that information is no longer useful to any user.
- Assume that unit can be NULL for an address, as this column does not always apply.
- Consequently, assume latitude and longitude are not unique as there can be multiple listings on a single coordinate, eg, an apartment complex. Beyond that, depending on the decimal specificity of the coordinates, we may have many addresses in the same location.
- Assume there is no need to check the credit card information beyond logging it in the right type and format.
- Assume users can only make comments on listings or people they have a booking in common with.
- Assume listings can have identical availability details (other than the id), for example, if a user wishes to rent out a house but on a 'shared' basis with multiple renters. In that case there may be multiple listings and availabilities that are identical.
- Assume when a booking is make, the availability changes to remove that booking time, and any subsequent changes to the availability doesn't affect the bookings.
- Assume hosts and renters can cancel their bookings at any time, before, after and during their stay, in which case, the listing owner will have to manually add the newly freed availability to the database, as we cannot assume the user wishes to have the listing available again, especially if they host cancelled a booking themselves.

ER Diagram:

Files - `ER Diagram I.jpeg`
Files - `ER Diagram II.jpeg`

Relation Schema:

The relation schema and the keys.

Users (*sin*, name, email, password, dob, occupation, phoneNum)

--(sin is user_id in other tables)

CreditInfo (*user_id*, *card_num*, card_type, exp_date)

--(multiple users can have the same card, assuming verification/security is not implemented)

Listings (*listing_id*, user_id, listing_type, num_bedrooms, num_beds, num_bathrooms, title, description)

Address (latitude, longitude, *listing_id*, street, unit, city, state, country, zipCode)

Amenities (*amenity_id*, amenity)

--(holds all amenities)

ListingAmenities (*listing_id*, *amenity_id*)

Availability (*availability_id*, listing_use, start_date, end_date, price, available, listing_id)

Bookings (*booking_id*, start_date, end_date, status, listing_id, user_id, card_num, cost)

ListingComments (comment, date, renter_id, listing_id, *booking_id*, rating)

UserComments (comment, date, *commenter_id*, commentee_id, *booking_id*, rating)

--(renter commenting on host and vice versa)

DLL Statements:

File - database.sql

Source Code:

Folder - src/*