CS 5200 - Database Management Systems - Final Report
Airbnb Management System

Group Name: RachelJacobJNeelakantanS
Group Members: Joann Rachel Jacob & Sriram Hariharan Neelankantan

1. Introduction

Airbnb is a widely popular platform that connects hosts who want to rent their homes or apartments to travelers who need a place to stay. With its massive user base and global reach, Airbnb has revolutionized the travel and hospitality industry. However, managing a property on Airbnb can be a complex task, involving a wide range of activities such as property listings, reservations, payments, guest communication, and more. Therefore, there is a need for a robust and efficient Airbnb management system that can help hosts streamline their property management process and provide a seamless experience to their guests. We designed a basic user-interactive application that conducts multiple CRUD operations in the backend to create a seamless flow in the application, allowing users to add, update, or remove listings, view, book, or submit reviews on a booking, and so on. This program provides a simple imitation of numerous different online hotel booking sites.

2. Project Description

In this project, we propose to design and develop a smaller version of Airbnb Management System that will provide a comprehensive set of features to hosts and guests alike. The system will be built using a relational database management system (RDBMS) and will leverage various technologies such as SQL, Python, and Streamlit to provide a simple user interface. The system will be able to handle a few aspects of property management, from creating listings to managing bookings.

For this project, we will be using the Inside Airbnb dataset, which contains detailed information about Airbnb listings in different cities around the world. This data is collected through web scraping and is updated regularly. It includes information such as location, size, price, availability, and host information. This dataset is freely available to the public and has been used by researchers and data scientists to study various aspects of the sharing economy and the impact of Airbnb on local communities.

The system will provide the basic functionalities for hosts and guests. Hosts can register, log in, view their listings, and create new listings. Guests can view listings, check availability, and make bookings. They can also post reviews on the listings. The system will use SQL to store and retrieve data and Streamlit to provide an interactive web-based interface.

The goal of this project is to provide a simple and efficient way for hosts and guests to manage their properties and bookings. By leveraging the Inside Airbnb dataset and the latest technologies, we aim to provide a seamless user experience that simplifies the management of short-term rentals.

Data Source: http://insideairbnb.com/get-the-data/ (We have chosen Boston for this project**).** We have made use of neighbourhoods and listings csv files for populating data. We have not used reviews data front the website.

3. Why does this interest us?

The Airbnb Management System is an interesting domain for several reasons. First, it caters to the booming hospitality industry, which has seen exponential growth in recent years, and it offers a unique platform for property owners to rent their space and for guests to find affordable lodging. The system allows for flexible bookings, including short-term rentals, which is appealing to both property owners and guests. Additionally, there's real-world data available which is very good to work and play around with which is extremely useful especially being in the Data Science field. Overall, the Airbnb Management System provides a convenient and reliable way for property owners to generate additional income and for travelers to find affordable lodging while on the move.

4. Technical Specifications

Database: MySQL

Languages: Python3 on the back-end

Software: MySQL Server, MySQL Workbench, VSCode.

Apps: GitHub for version control

Libraries: Streamlit for frontend, pandas for handling real-world Airbnb data, pymysql for connecting Python and MySQL, hashlib for user password hashing, BeautifulSoup for UI.

Hardware: Macbook Pro. There are no specific machine restrictions for the project, as long as the system can run the software tools listed above.

5. README

a. Download and extract the zip file named RachelJacobJNeelakantanS_project.zip. This contains 3 folders:
   i. Application Code - This contains the code for running the application.
   ii. Database Schema - This contains the sql file used for creating the schema and the sql dump file for populating the database.
   iii. Final Report.
   iv. Miscellaneous Docs - UML diagram, Logical Design, User flow diagram, Powerpoint presentation.

b. Open **Terminal** and do the following steps to set up the environment. Navigate to the project directory:

   cd RachelJacobJNeelakantanS_project/Application Code

   Create a virtual environment and activate it using the following commands.

   python3 -m venv airbnb_env
   source airbnb_env/bin/activate

c. Install the required packages. The following commands will install all the necessary Python packages needed for the applications including Streamlit, pymysql, pandas and more. All the required dependencies are included in requirements.txt.

   python3 -m pip install --upgrade pip
   pip3 install -r requirements.txt

d. Run the dump file Database_Schema/airbnb_dump.sql to create a database named 'airbnb'. Pre-population of data into tables named Countries, States, Cities, Neighbourhoods, PropertyTypes, RoomTypes, Users, Hosts and Listings to match Boston's data is done by admin. The data populated using this is provided in the database dump. So, **this need not be repeated.**

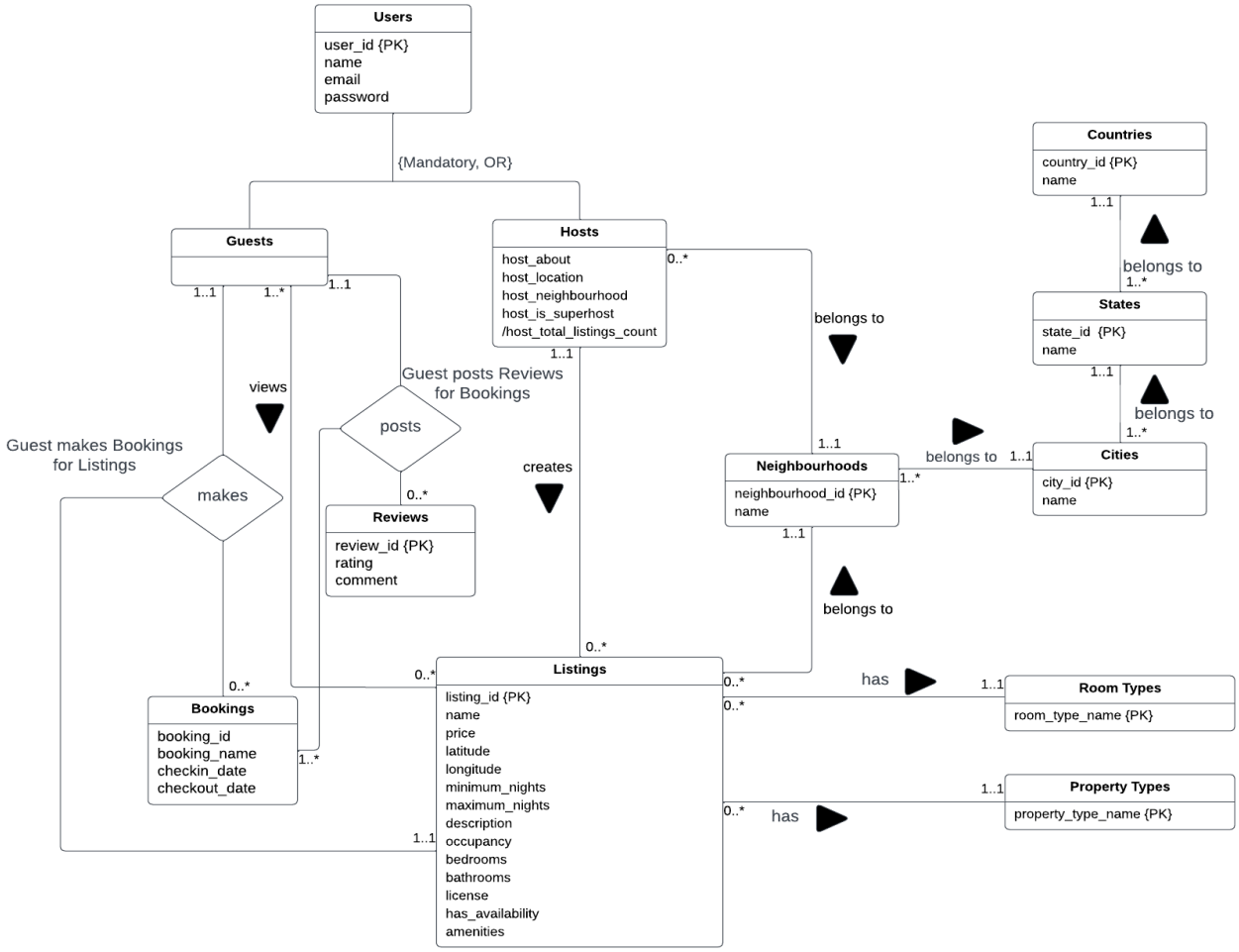
   python3 code/populate_data.py

e. To run the application, first open the file, *RachelJacobJNeelakantanS_project/Application Code/code/home.py* in IDE and change the connection parameters to reflect your DB username and password. Then execute the following command in the terminal (switch to the working directory *RachelJacobJNeelakantanS_project/Application Code* before executing the below command):

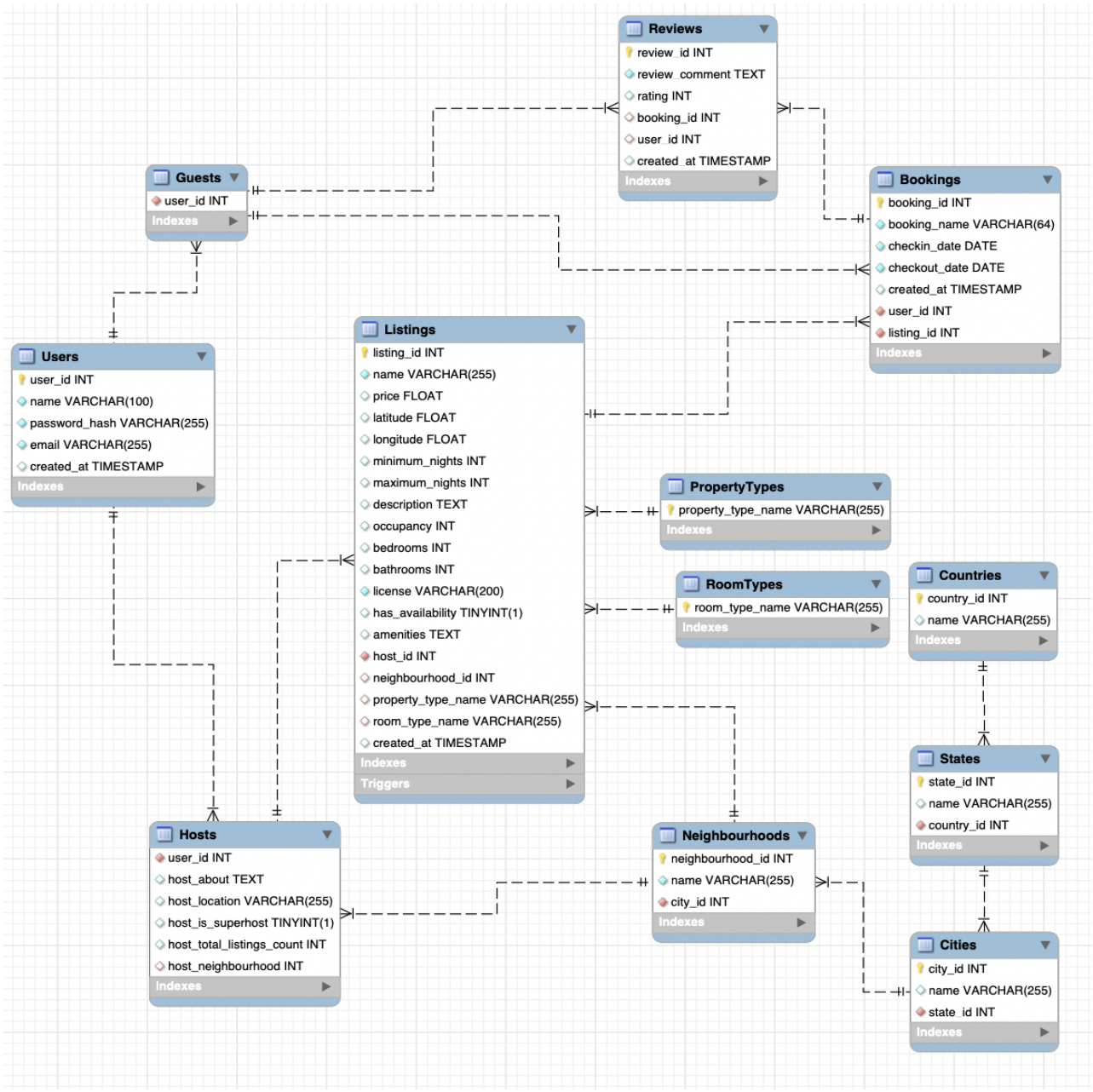
   streamlit run code/home.py

This should open up the application in the browser. There are two user roles. You can register as a host or guest and login. Based on the user role, different functionalities will be listed on the page. Below are the credentials already available in the database dump. **Host Credentials:** Pre-populated host - Username - frank4804@airbnb.com, Password - frank@123, Newly created host - Username - joann@airbnb.com, Password - joann@123. **Guest Credentials**: Newly created guest - Username - sriram@airbnb.com, Password - sriram@123

6. Conceptual Design



7. Logical Design

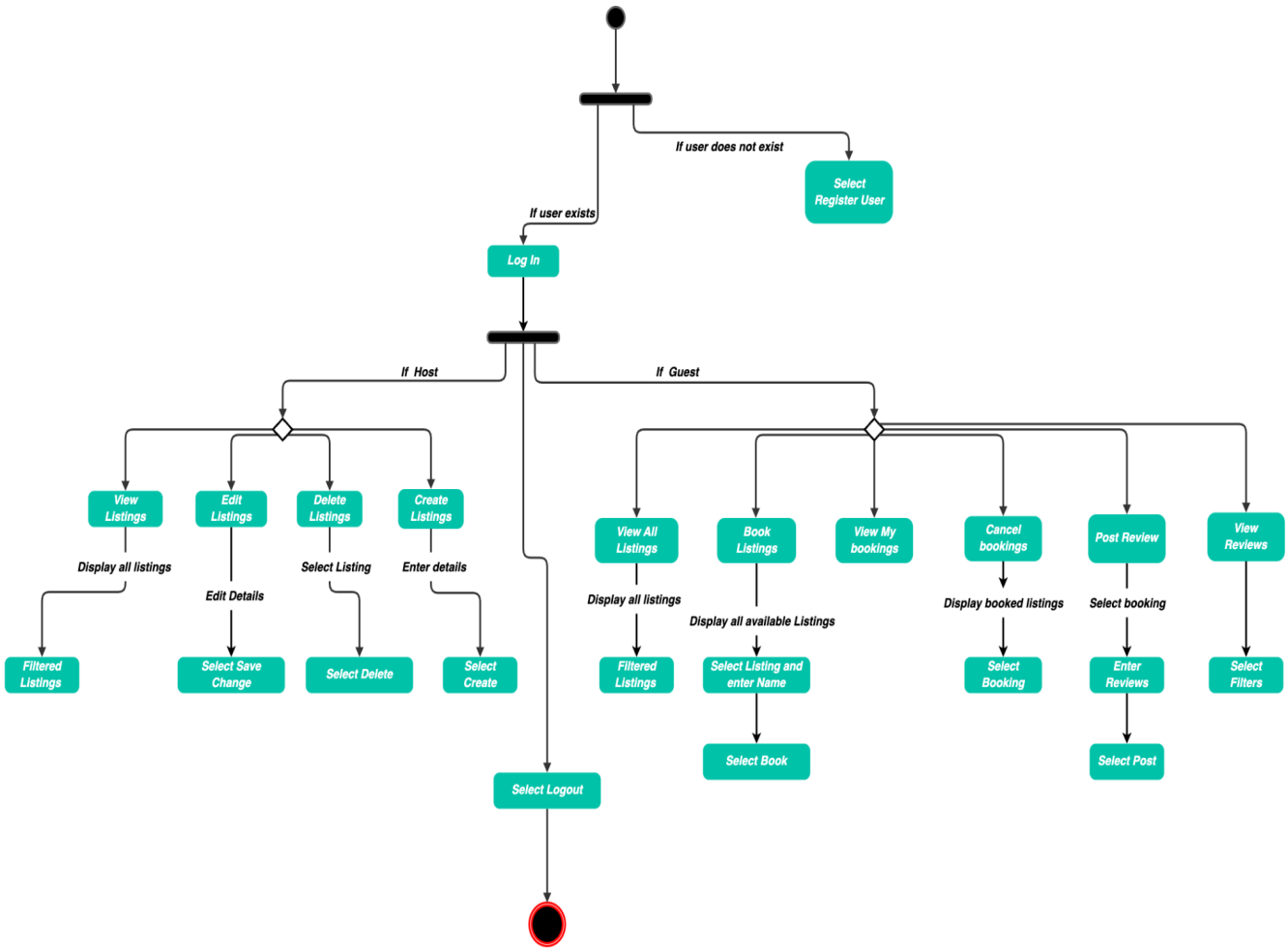


Triggers: update_host_total_listings_count and update_host_total_listings_count_on_delete on Listings table to modify total listings count of a host based on listings created and deleted.

Procedures: search_listings to get all listings based on criteria given by user

Functions: avg_listing_rating to get the average rating of a listing.

8. User Flow Diagram



Step 1: The application's welcome page consists of two options: Login & Register. Users who have already registered can login using their credentials. New users have to go through the register page, which can be accessed by selecting the register button.

Step 2: The registration page displays a form, which upon submitting , creates a record of the new user in the backend. The form varies depending upon the two user roles i.e, Guest or Host.

Step 3: After logging in to the application, depending on whether a user is a host or a guest, the available functionalities differ. The logged-in user is checked in the backend to determine if they are a user or a host.

Step 4: If a user is a Guest, they can view listings, book listings, view bookings, cancel listings, post reviews for their booking and view reviews of all the listings.

I. View Listings: Populates all the listings from the database.
II. Book Listings: Filters such as price, date availability, neighborhood,etc., are present to view listings. A guest can choose one of the listings and book, which will add the booking in the backend.
III. View Booking: Maps the user id to find the booking made by the logged in user and populates them.
IV. Cancel booking: The user can choose to cancel any of their bookings. The record of the booking is deleted from the database.
V. Post Reviews: The user can post ratings and reviews to any of their bookings.
VI. View Reviews: Reviews already posted for the listings can be viewed by choosing any listing that the user wants to see.

Step 5: Hosts can view their listings, edit listings, delete listings and create listings.

I. Create Listings: Hosts can create their own listings by providing information in a form and submitting. This will create a new listing mapped to their unique id in the backend.
II. Edit Listings: The hosts can choose to edit any of their listings. They can select from a list of their listings and modify any of the fields they would like to and save changes. This change is immediately reflected in all the tabs that contain this listing.
III. View Listings: This feature displays all the listings created by the host who is logged in.
IV. Delete Listings: Upon choosing a listing to be deleted and clicking on the delete button, that listing is deleted from the database.

Step 6: Logout: The logout button is present on all the pages of the application. The application then erases the login credential from the session state of the application and redirects to the login/register page again.

9. Lessons Learned

In terms of technical expertise, we learned how to use Python libraries like Pandas for data manipulation and MySQL for database creation and querying, as well as how to work with foreign keys and relational databases. We also gained a better knowledge of the significance of data cleaning and preprocessing before analysis, as well as enhanced our skills in dealing with data problems and exceptions in Python programming. Creating the application using Streamlit helped us get hands-on experience in Streamlit.

In terms of insights and time management, we understood the importance of thoroughly comprehending the data domain in order to draw relevant conclusions. We gained important insights into the market and its dynamics by examining the Airbnb dataset. We also learned how

to set clear goals and deadlines, create checkpoints and milestones to track project progress, and ensure timely completion.

We could have investigated alternate database management systems or data storage technologies, such as NoSQL databases or cloud storage options, as alternative approaches to the project. An alternative could have been Flask instead of Streamlit to implement APIs as well as experiment with various data visualization libraries and tools to create more aesthetically appealing and useful graphs and charts. Furthermore, we could have used machine learning methods to perform predictive modeling and analysis on the dataset.

Due to foreign key constraints, we faced problems entering data into the Listings table during the project. We fixed this by ensuring that the Users and Hosts table were appropriately populated before entering data into Listings and identifying appropriate substitute values for missing data so that adequate analysis could be performed. We had to devise some extra pre-processing methods to transform the data into a more relevant and layman-understandable manner because we were dealing with raw data from AirBnB that was somewhat pre-processed. Some of the text in the listings description column had html tags, which were transformed to intelligible language while retaining the html intended effects.

Regarding code not implemented to perfection, review functionality is not implemented in the best way possible. Guests can post multiple reviews on the same booking as of now. Also, reviews are linked with booking and if a booking is canceled, the reviews will remain but with booking id set to null and won't be able to trace back to corresponding listings. This is something that can be improved.

10. Future Work

In the future, the Airbnb management system application database can be utilized for various purposes such as:

a. **Business Analysis**: Analyzing the data to gain insights into the Airbnb market trends, pricing strategies, and customer preferences to make informed business decisions.
b. **Predictive Modeling**: Using machine learning algorithms to predict the rental prices of properties based on various factors such as location, amenities, and other variables.
c. **Recommendation System**: Developing a recommendation system that suggests properties to users based on their preferences, search history, and booking patterns.
d. **Integrating with other services**: Integrating the database with other services such as third-party booking and payment systems, social media platforms, and marketing automation tools to streamline operations and enhance user experience.

Potential areas for added functionality include:

a. **User Roles**: Adding more user roles, such as Admin, Manager, etc., could enable different views of the application depending on the user role.
b. **Booking Management**: Adding features to manage bookings, such as booking confirmation, payment, cancellation, and refund processing. We can also improve on Review functionality for guests.
c. **Property Images**: As in a traditional hotel booking website, images should be part of the listings' view. This could be done by storing image files as blobs or compressed binary files into the database , which will help us retrieve them later and populate in the listing's view page.
d. **User Account**: There is some scope of functionality improvements here. We could add features like resetting password, adding a profile image, storing and viewing payment methods.
e. **UI**: Improving the UI with more interactive and aesthetic components such as multiple tabs for different types of properties etc, would improve the look and appeal of the webpage.