



National University
of computer and emerging sciences

Project Report

CS2005 Database Systems

Project Title

Rent A Roo

(An Online Property Rental Marketplace)

Group Members:

Syed Ahmed Mahmood (20K-0153)

Muhammad Qasim Fuzail (20K-0157)

Ali Shah Naushad (20K-1078)

Introduction

Rent A Roo is a mobile app that allows hosts to rent out their properties to earn money and at the same time allows guests, travelling away from home, to immerse themselves in a new city by staying at exciting houses rented out by the local residents for a cheaper price instead of staying at overpriced and mundane hotels.

Features

A user can't use the app unless they're registered and logged in, the app takes the user's details during registration such as email, password, name, phone number, about me etc. The user is free to edit their profile whenever they want and they can delete their profile too (only if a guest hasn't already paid for a stay at one of their listings). After logging in the user can change their profile picture from the default image, the user can then play 2 roles at the same time in the app, a guest and a host.

They can be a host and create listings with all of their property's details such as an eye catching title, description, images, address and all the amenities(iron, washing machine, TV, WiFi, smoking allowed etc) they offer at their place as well as the no. of bedrooms and bathrooms, how many people can the place accommodate, the minimum and maximum nights a guest can stay and if the property is shared with other people or will the guest have the whole place to themselves and most importantly the nightly price to stay at this listing. The host is free to edit this listing whenever they want except after they delete the listing(it will be saved in the Database for transaction history purposes), the host can also pay to promote the listing to show up on the top of any relevant search results.

The guest can search for listings and filter and sort their results too, they can also view popular listings(the top 3 most viewed listings on the app right now), they can press a listing to go to its listing page and add it to their favourites to view it later, or they can ask questions about the listing(which the host can later answer) and view previously asked and answered questions or rating and reviews about

the listing. The guest can also press a link on the listing page which redirects them to *Google Maps* where the listing's location will be displayed.

The guest can reserve the listing and will be asked to choose a valid checkin and checkout date and they will be shown their Due Amount if the reservation is successfully sent to the host, the guest can then go to the guest's reservations tab to view all their reservations and they have options to view the listing page the reservation was made to as well as check their reservation status(which may be Accepted/Rejected/Pending).

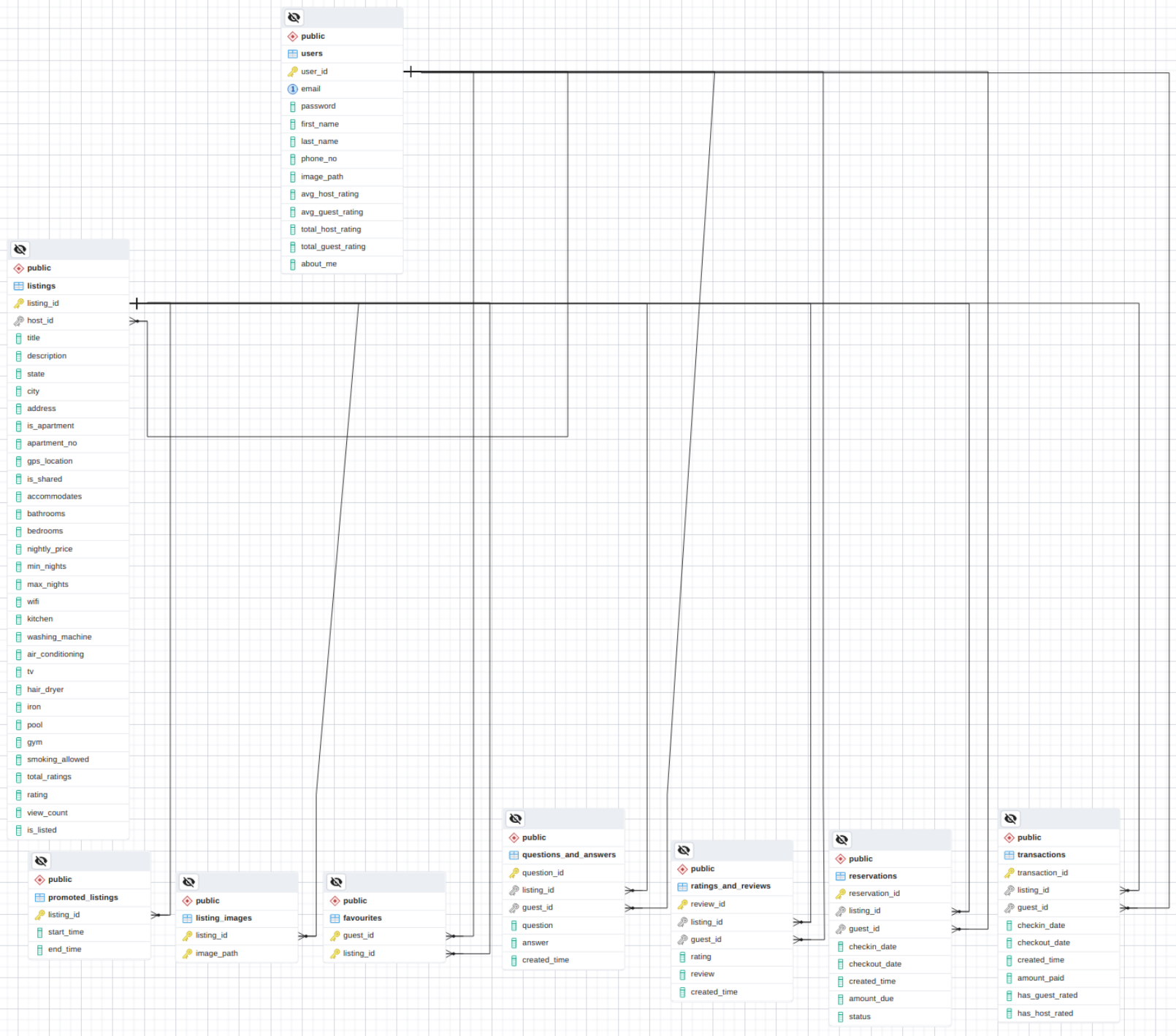
The host can go to the host's reservations tab to check any reservations they've recieved and go to the listing page related to the reservation or go to the guest's page that sent the reservation and then accept or reject the reservation, the reservation will automatically be rejected in 24hours otherwise. If the host accepts it and the guest completes their payment for the confirmed reservation, then the transaction can be viewed in the host's and guest's transactions tab for the host and guest respectively. Here they can press the view listing button to view the listing related to the transaction, after the checkout date arrives the host can rate the guest and similarly the guest can rate the host as well as rate and review the listing.

Every user has a guest rating and a host rating and they can change the app language whenever they want too

Technology

- FastAPI (A Python Web Framework)
- Flutter (SDK for android app development)
- PostgreSQL (A Relational DataBase Management System)

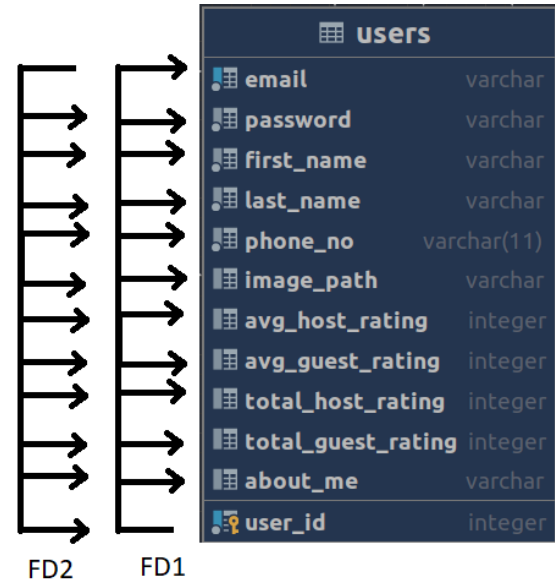
ER Diagram



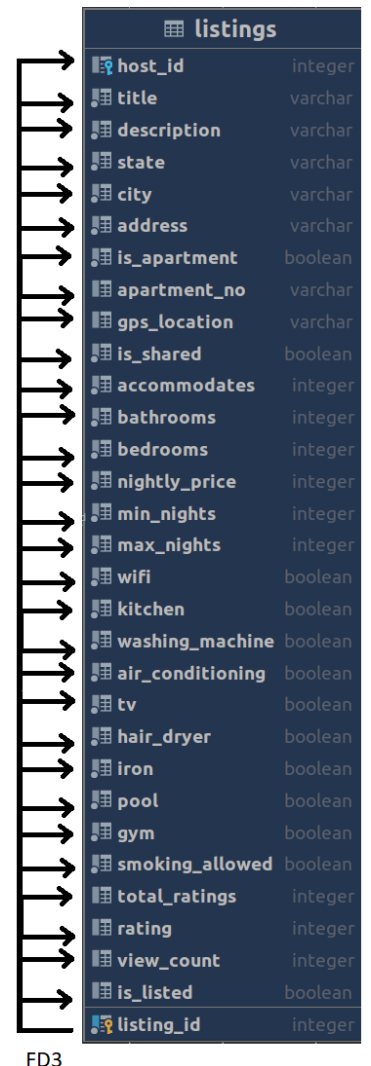
BCNF Normalisation

- All the tables below will already be in 1NF since multivalued and composite attributes aren't allowed in a modern RDBMS

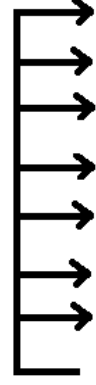
- There are 2 candidate keys in **users**, {user_id} and {email}. The Primary Key(which is not relevant to normalisation anyways) is {user_id}.
- **Note:** Multiple users can have the same phone number, eg: a married couple using the same number for both of their accounts.
- **users** has no partial dependencies so it is in 2NF, it has no transitive dependencies so it is in 3NF and finally we can check all the functional dependencies of **users** and tell that all the determinants are candidate keys.



- There is 1 candidate key in **listings**, {listing_id} which is also the Primary Key.
- **listings** has no partial dependencies so it is in 2NF, it has no transitive dependencies so it is in 3NF and finally we can check all the functional dependencies of **listings** and tell that all the determinants are candidate keys.




- There is 1 candidate key in **reservations**, {reservation_id} which is also the Primary Key.
- **reservations** has no partial dependencies so it is in 2NF, it has no transitive dependencies so it is in 3NF and finally we can check all the functional dependencies of **reservations** and tell that all the determinants are candidate keys.



reservations		
listing_id		integer
guest_id		integer
checkin_date		timestamp with time zone
checkout_date		timestamp with time zone
created_time		timestamp with time zone
amount_due		integer
status		varchar
reservation_id		integer

FD4

- There is 1 candidate key in **transactions**, {transaction_id} which is also the Primary Key.
- **transactions** has no partial dependencies so it is in 2NF, it has no transitive dependencies so it is in 3NF and finally we can check all the functional dependencies of **transactions** and tell that all the determinants are candidate keys.



transactions		
listing_id		integer
guest_id		integer
checkin_date		timestamp with time zone
checkout_date		timestamp with time zone
created_time		timestamp with time zone
amount_paid		integer
has_guest_rated		boolean
has_host_rated		boolean
transaction_id		integer

FD5

- There is 1 candidate key in **questions_and_answers**, {question_id} which is also the Primary Key.
- **questions_and_answers** has no partial dependencies so it is in 2NF, it has no transitive dependencies so it is in 3NF and finally we can check all the functional dependencies of **questions_and_answers** and tell that all the determinants are candidate keys.



questions_and_answers		
listing_id		integer
guest_id		integer
question		varchar
answer		varchar
created_time		timestamp with time zone
question_id		integer

FD6

- There is 1 candidate key in **ratings_and_reviews**, {review_id} which is also the Primary Key.
- **ratings_and_reviews** has no partial dependencies so it is in 2NF, it has no transitive dependencies so it is in 3NF and finally we can check all the functional dependencies of **ratings_and_reviews** and tell that all the determinants are candidate keys.



ratings_and_reviews	
listing_id	integer
guest_id	integer
rating	integer
review	varchar
created_time	timestamp with time zone
review_id	integer

- There is 1 candidate key in **promoted_listings**, {listing_id} which is also the Primary Key.
- **promoted_listings** has no partial dependencies so it is in 2NF, it has no transitive dependencies so it is in 3NF and finally we can check all the functional dependencies of **promoted_listings** and tell that all the determinants are candidate keys.



promoted_listings	
start_time	timestamp with time zone
end_time	timestamp with time zone
listing_id	integer

- There is 1 candidate key in **favourites**, {guest_id, listing_id} which is also the Primary Key.
- There is 1 candidate key in **listing_images**, {listing_id, image_path} which is also the Primary Key.
- **NOTE: In both favourites and listing_images, no valid FDs hold(apart from the trivial dependencies), thus both of these relations are in 2NF, 3NF and BCNF.**

favourites	
guest_id	integer
listing_id	integer

listing_images	
listing_id	integer
image_path	varchar

Source: <https://stackoverflow.com/a/59008880>