team058-dbquest
Ayushe Nagpal, Reva Jethwani, Achintya Sanjay, Karan Shah

AirDnB: All-in-One Platform

team058-dbquest
Ayushe Nagpal, Reva Jethwani, Achintya Sanjay, Karan Shah

Stage 1

AirDnB: AirBnB Database Project

1. **Project Summary:**

A one-stop portal for users seeking to explore or stay in NYC, offering detailed insights into AirBnB accommodations in neighborhoods based on safety, transportation options, and local amenities.

2. **Description of an application of your choice. State as clearly as possible what you want to do. What problem do you want to solve, etc.?**

We want to provide a seamless experience for those seeking to explore or stay in New York City with safety, convenience, and local amenities in mind through a web application. By integrating comprehensive data on Airbnb listings, crime statistics, Citibike and public transportation options, as well as local stores and restaurants, our platform will provide users with a one-stop portal for informed decision-making. We are addressing the challenge of users scattering to different sources to do their research by aggregating all this data.

Safety is a paramount concern for anyone staying in a new city. Our application will integrate up-to-date crime statistics by neighborhood, offering users a clear view of the safety landscape of New York City. This feature will enable users to make informed decisions about where they choose to stay, prioritizing their safety and peace of mind. To facilitate easy navigation around the city, our platform will provide comprehensive data on Citibike rental stations and public transportation options, including subway lines, bus routes, and their schedules.

3. **What would be a good creative component (technically challenging function) that can improve the functionality of your application? (What is something cool that you want to include? How are you planning to achieve it?)**

The creative component that we will be displaying on our application is a map that updates as you change how you want to filter the results. It will display a user's chosen amount of AirBnBs on a map based on what filters the user selects. For example, if the user wishes to see 25 AirBnbs in the safest neighborhoods, it will display 25 markers of different AirBnB locations that are ranked by safety.

We are planning to achieve it by using some sort of map API. Once we calculate the top AirBnBs by some metric for each filter option we choose later on, we will query a new table of these AirBnBs and graph markers on the map for each one.

4. **Usefulness. Explain as clearly as possible why your chosen application is useful. What are the basic functions of your web application? (What can users of this website do? Which simple and complex features are there?). Make sure to answer the following questions: Are there any similar websites/applications out there? If so, what are they, and how is yours different?**

The chosen application is useful because it allows potential visitors of the city to easily gauge the safety of their specific destination. New York City is vast with many boroughs, and there are no accessible applications currently available that would allow people to comprehensively understand and compare the safety of AirBnB sites. While people can cross-check crime rates with neighborhoods, it can be difficult to pinpoint a specific location (where they would actually stay), making it difficult to accurately gauge safety.

Our web application will allow users to specifically check bike routes, nearby restaurants, Citi bike locations, and crime rates by choosing their AirBnb location and applying the respective filters. Basic features include identifying AirBnBs by price and location, and more complex features would include applying different filters and putting these locations on a map. Our application ultimately addresses the need for a holistic platform that combines safety, accommodation, transportation, and local activity in one intuitive interface. Users can plan with confidence with access to up-to-date crime statistics and transportation schedules to make informed decisions.

5. **Realness. We want you to build a real application. So, make sure to locate real datasets. Describe your data sources (Where is the data from? In what format [csv, xls, txt,...], data size [cardinality and degree], what information does the data source capture?). It would be hard to satisfy stage 2 requirements with one dataset. Thus, we strongly recommend identifying at least two different data sources for your project.**

    a. AirBnBs in NYC:
        i. http://insideairbnb.com/get-the-data
        ii. Format: CSV
        iii. Cardinality: 39,720
        iv. Degree: 18
        v. AirBnB Information for NYC
    b. Crime in NYC:

   i. Data Source: City of New York Website https://data.cityofnewyork.us/Public-Safety/NYPD-Complaint-Data-Historic/qgea-i56i/data_preview
   ii. Format: CSV
   iii. Cardinality: 8359721 entries
   iv. Degree: 35
c. Restaurants in NYC:
   i. Data Source: City of New York Website https://data.cityofnewyork.us/Transportation/Open-Restaurant-Applications-Historic-/pitm-atqc/data_preview
   ii. Format: CSV
   iii. Cardinality: 14428 entries
   iv. Degree: 35
d. Citibike Locations in NYC:
   i. Data Source: https://www.kaggle.com/datasets/akkithetechie/new-york-city-bike-share-dataset
   ii. Format: CSV
   iii. Cardinality: 735502
   iv. Degree: 17
   v. CitiBike Ride History in NYC

6. **A detailed description of the functionality that your website offers. This is where you talk about what the website delivers. Talk about how a user would interact with the application (i.e., things that one could create, delete, update, or search for).**

The user can interact with the website by changing what filters they would like to use when searching for an AirBnb in New York City. They can also change how many of the top AirBnBs they would like to view when seeing the top recommendations. For example, if the user wishes to see 25 AirBnbs in the safest neighborhoods (using crime database), it will display 25 markers of different AirBnB locations that are ranked by safety.

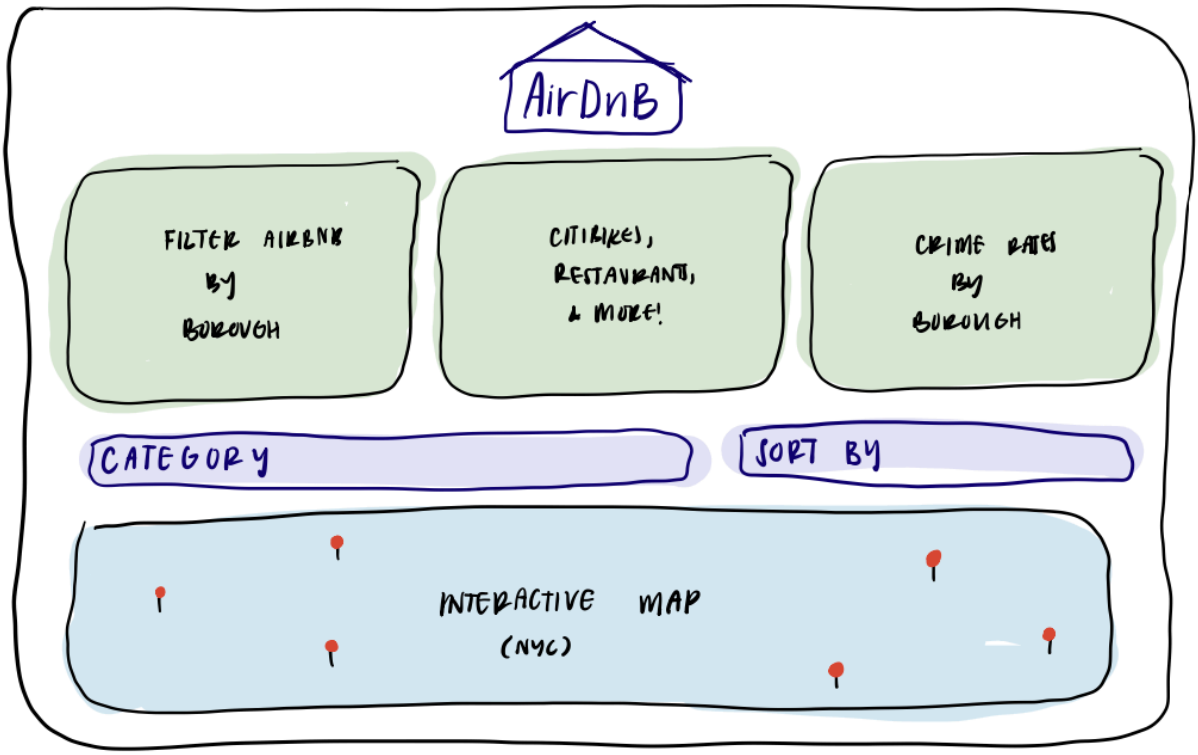
The website offers a map that will allow users to visualize the AirBnBs in relation to whatever filter they placed.

Customize Searches: Based on safety, transportation options, and proximity to amenities.
Interact with Dynamic Maps: Showcasing Airbnb listings, Citibike stations/paths, public transport options, and local businesses and update according to filters.

Data Integration:

- Airbnb Listings: Users can search and filter Airbnb options by price, location, amenities, and availability, making it easier to find the perfect stay.
- Crime Statistics: Updated crime data by neighborhood allows users to assess the safety of potential stays, ensuring peace of mind.
- Transportation Data: Detailed information on Citibike rental stations, subway lines, bus routes, and schedules helps users plan their travel within the city efficiently.
- Local Amenities: Information on nearby stores, restaurants, and attractions enables users to explore what each neighborhood has to offer.

a. A low-fidelity UI mockup: What do you imagine your final application's interface might look like? A PowerPoint slide or a pencil sketch on a piece of paper works!



b. Project Distribution:

i. Overall work will be shared throughout, but the parts below are what each person is responsible for

ii. Frontend:

1. Ayushe Nagpal
    a. User Interface Design
2. Reva Jethwani
    a. Dynamic Components

iii. Backend:
1. Achintya Sanjay
    a. Creating functions for filtering options
    b. Building a relational database management system
2. Karan Shah
    a. Setting up SQL Database, joining different tables, producing map output table

CRUD, Search, and Application Functions

Create
Functionality: Users can create new accounts/profiles.
This allows users to save their preferences, favorite listings, or previous searches for future reference. For example, they might want to save a list of favorite Airbnbs or preferred filters for quicker access.

Read
Functionality: Users can view listings, crime statistics, transportation options, and local amenities.
This is the core functionality of the application – users can read information about Airbnbs, crime statistics, transportation options, and local amenities to make informed decisions about where to stay in NYC.

Update
Functionality: Users can update their profile information or preferences.
This allows users to modify their saved preferences, such as updating their preferred filters or changing their saved favorite listings.

Delete
Functionality: Users can delete their accounts/profiles or remove saved preferences/favorite listings.
This gives users control over their data and preferences. For example, they might want to delete their account if they don't plan to use the platform or remove outdated saved preferences after traveling.

Search
Functionality: Users can search for Airbnbs based on various criteria such as price, location, amenities, safety (crime statistics), transportation options, and proximity to local amenities.
Users can search for specific types of Airbnbs that meet their criteria, such as finding accommodations in safe neighborhoods with easy access to transportation and nearby amenities.

Potential Application
Functionality: Display a map that updates based on user-selected filters, showing Airbnb listings, Citibike stations/paths, public transport options, and local businesses.
A map enhances the user experience by providing a visual representation of the search results, so users can see the spatial distribution of Airbnbs, transportation options, and local amenities on a map, making it easier to understand the geographical context of their search results.

Stage 2

Entities and Their Attributes:

User
- UserID (Primary Key)
- Username
- Email
- Preferences (e.g., safety, amenities)

AirbnbListing
- ListingID (Primary Key)
- HostID (Foreign Key, references User)
- Address
- Price
- Amenities
- SafetyRating (Derived from CrimeData)

CrimeData
- CrimeID (Primary Key)
- Location
- Date
- Type
- Severity

PublicTransportStation
- StationID (Primary Key)
- Type (e.g., Subway, Bus)
- Location
- Schedule

Subway Station
- StationID (Primary Key)
- Location
- AvailableBikes

- TotalDocks

StoreRestaurant

- BusinessID (Primary Key)
- Name
- Type (Store or Restaurant)
- Location
- Rating

Relationships:

- User to AirbnbListing: A "books" relationship can exist between User and AirbnbListing, indicating which user booked which Airbnb.
- AirbnbListing to CrimeData: A "located in" relationship can show the association between an Airbnb listing and crime data, based on geographical location.
- AirbnbListing to PublicTransportStation/CitibikeStation: "Nearby" relationships to indicate proximity between Airbnb listings and transportation options.
- AirbnbListing to StoreRestaurant: Another "nearby" relationship to represent the proximity of Airbnb listings to local businesses.

Conceptual ER Diagram Overview:

- A User entity connected to AirbnbListing through a "Books" relationship.
- AirbnbListing is centrally connected to multiple entities:
  - A "Located in" relationship with CrimeData to indicate the safety rating.
  - "Nearby" relationships with PublicTransportStation, CitibikeStation, and StoreRestaurant to show proximity to each feature.
- CrimeData, PublicTransportStation, CitibikeStation, and StoreRestaurant are standalone entities that are connected to AirbnbListing based on geographical data.

Entities and Their Assumptions:

User: Represents individuals using the application, either as Airbnb hosts or guests. Assumed to be a central entity for personalizing the application experience.

- Modeled as an entity because users have unique attributes and can have various interactions within the app (e.g., booking Airbnbs, setting preferences).
- Many-to-many relationship with AirBnB listing, as users can book multiple listings, and each listing can be booked my many users

AirbnbListing: Represents the accommodations available for booking. It's a distinct entity due to its complex attributes, including location, price, and amenities.

- A listing can have various relationships with locations, such as proximity to crime, transport, and amenities, justifying its status as a separate entity.

- One-to-many relationship with crime data, as each listen can be affected by multiple incidents of crime

CrimeData: Represents crime incidents in NYC, which is essential for assessing the safety of areas surrounding Airbnb listings.

- Modeled as an entity because each record has unique attributes (e.g., type, severity) and impacts multiple Airbnb listings differently.

PublicTransportStation and CitibikeStation: These represent available public and bike transport options. They're separated due to their differing nature and data sources but are crucial for assessing the accessibility of Airbnb listings.

- Each station has unique characteristics, such as type or availability, necessitating distinct entities.

StoreRestaurant: Represents local businesses. It's an entity because each business has unique attributes (e.g., type, rating) and contributes differently to the appeal of an area.

Relationships and Cardinality:

- User to AirbnbListing: Modeled as a many-to-many relationship because a user can book multiple listings and a listing can be booked by multiple users. This necessitates an associative entity to track bookings.
- AirbnbListing to CrimeData: A one-to-many relationship, assuming that each listing is affected by multiple crime incidents based on its location. This relationship helps calculate a safety rating for each listing.
- AirbnbListing to PublicTransportStation/CitibikeStation: A many-to-many relationship, as a listing can be near multiple stations and a station can serve multiple listings. This highlights the accessibility of the listing.
- AirbnbListing to StoreRestaurant: Also a many-to-many relationship, reflecting the proximity of multiple businesses to a listing and vice versa, affecting the listing's desirability.

Normalization:

The schema will be normalized to the Third Normal Form (3NF) because it ensures that all data is logically stored, eliminating redundancies and dependencies that aren't related to a table's key. 3NF is chosen over BCNF to balance between strict normalization and practical usability, ensuring that all attributes are dependent on the key, the whole key, and nothing but the key, while still allowing for practical application design.

Tables that have already hold 3NF status:

Restaurants Table

Primary Key: Global ID
Attributes: Restaurant Name, Latitude, Longitude

Subway Table
Primary Key: Station Name
Attributes: Latitude, Longitude

Crime in NYC Table
Primary Key: Complaint_Num
Attributes: Latitude, Longitude

Airbnb Table
Primary Key: ListingId
Attributes: URL, Name, Latitude, Longitude

Account Information Table
Primary Key: Username

- Attributes: First Name, Last Name, Email, Phone Number
- Action Needed: Ensure that each username has a unique email and phone number. If users can have multiple emails or phone numbers, consider creating a separate table for contact information with a foreign key reference to the Username.

Steps to Apply 3NF to the Graph Table:

Identify the Primary Key: Determine what the primary key of the Graph table should be. It could be the URL if it's unique for each record.

Remove Transitive Dependencies:

- Eliminate attributes that do not depend on the primary key of the Graph table. For example, Station Name, Restaurant Name, and their corresponding location data should be removed since they are related to the Subway and Restaurants tables, respectively.
- Instead, include foreign keys that reference the primary keys of the associated tables.

Create New Tables if Necessary:

- If there is a need to maintain relationships between graphs and locations (like subways or restaurants), create a new table to manage these relationships with foreign keys pointing to the Graph table and the Subway/Restaurants tables.

Update Your Queries and Application Logic:

- With the normalized structure, you'll likely need to update your SQL queries and any application logic that interacts with the database to accommodate the new table structures.

Logical Design (Relational Schema):

- User(UserID: INT [PK], Username: VARCHAR(255), Email: VARCHAR(255), Preferences: VARCHAR(255))

- AirbnbListing(ListingID: INT [PK], HostID: INT [FK to User.UserID], Address: VARCHAR(255), Price: DECIMAL, Amenities: VARCHAR(255), SafetyRating: DECIMAL)
- CrimeData(CrimeID: INT [PK], Location: VARCHAR(255), Date: DATE, Type: VARCHAR(255), Severity: INT)
- PublicTransportStation(StationID: INT [PK], Type: VARCHAR(255), Location: VARCHAR(255), Schedule: VARCHAR(255))
- CitibikeStation(StationID: INT [PK], Location: VARCHAR(255), AvailableBikes: INT, TotalDocks: INT)
- StoreRestaurant(BusinessID: INT [PK], Name: VARCHAR(255), Type: VARCHAR(255), Location: VARCHAR(255), Rating: DECIMAL)
- ListingCrime(ListingID: INT [FK to AirbnbListing.ListingID], CrimeID: INT [FK to CrimeData.CrimeID])
  To capture the many-to-many relationship between Airbnb listings and crime data.
- ListingTransport(ListingID: INT [FK to AirbnbListing.ListingID], StationID: INT, StationType: VARCHAR(255))
  To differentiate between PublicTransport and Citibike stations while capturing the many-to-many relationship.
- ListingBusiness(ListingID: INT [FK to AirbnbListing.ListingID], BusinessID: INT [FK to StoreRestaurant.BusinessID])
  To represent the many-to-many relationship between Airbnb listings and nearby restaurants.

Within stage one, we implemented some changes based on the comments given. First, we wrote out “AirDnB” explicitly as “AirBnB Database Project,” as it is a play on words. We also explicitly outlined the connection between CRUD, search and our application functions, explaining how a user might apply these functions for their own personal use.