ITC 6000 – DATABASE MANAGEMENT SYSTEMS

FINAL PROJECT REPORT – TREK TRIBE

AKHILA SINGARAJU

INTRODUCTION

Our goal is to develop a travel companion app that connects travelers in search of companions, whether they're solo adventurers or relocating to new cities. Users can craft detailed profiles to showcase their travel interests and destinations, then browse potential travel partners based on various criteria.

The app prioritizes secure communication through in-app calls and video chats to confirm compatibility before embarking on journeys. To ensure sustainability, we've opted for a freemium model, offering essential features for free and premium options like advanced search filters via a subscription. My motivation for this project stems from my love for travel and the aspiration to create a global community of adventurers who can exchange experiences and forge enduring memories.

BUSINESS ANALYSIS

1. Solo Adventurer – <u>Background</u>: Travel enthusiasts with a passion for exploring new destinations. They often embark on solo adventures to remote places.
<u>Why they Use the App</u>: They use the app because traveling alone can be daunting, and she believes that sharing the journey with like-minded individuals not only enhances the experience but also makes it safer and more cost-effective.
<u>How they use the App</u>: User create a detailed profile highlighting their travel preferences, interests, and the destinations they plan to visit. they actively search for potential travel companions who share the same love for adventure, carefully reviewing profiles to assess compatibility. They initiate conversations through in-app messaging and conduct video calls to get to know their potential travel partners better before making travel plans.

2. Business Traveler – <u>Background</u>: Business professional who frequently travels to different cities for work. They value efficiency and enjoys making the most of their limited free time during business trips.
<u>Why they Use the App</u>: They use the app to connect with local guides or fellow business travelers during their trips. They believe that exploring new places with others who

share their interests can make their business travel experiences more enjoyable and culturally enriching.

How they Use the App: They use advanced search filters to find users with similar schedules and interests. They relies on the app's quick meetup scheduling feature to plan activities during their free time. The ad-free premium subscription aligns with their professional needs and desire for an efficient experience.

3. New City Relocator – Background: A recent college graduate in their early 20s who is about to move to a new city for a job opportunity.

Why they Use the App: They use the app to make friends and acquaintances in the new city. They want to explore her new environment, engage in social activities, and build a network in their new location.

How they Use the App: They completes their profile, highlighting their hobbies. They occasionally opts for premium features to refine their matches but primarily use the free version to socialize and explore her new city. The app helps them connect with locals and fellow newcomers, facilitating her transition to a new place.

Rules/Logic:

Profile Creation: Users must provide essential information such as their name, age, travel destination, and interests during profile creation.

Matching Algorithm: The matching of users is based on their compatibility factors, including location, travel dates, interests, and personality traits.

Messaging: Users can only initiate conversations with other users if both parties have shown mutual interest by liking each other's profiles.

Premium Subscription: The app enforces a premium subscription model for advanced features, including unlimited messaging, advanced search filters

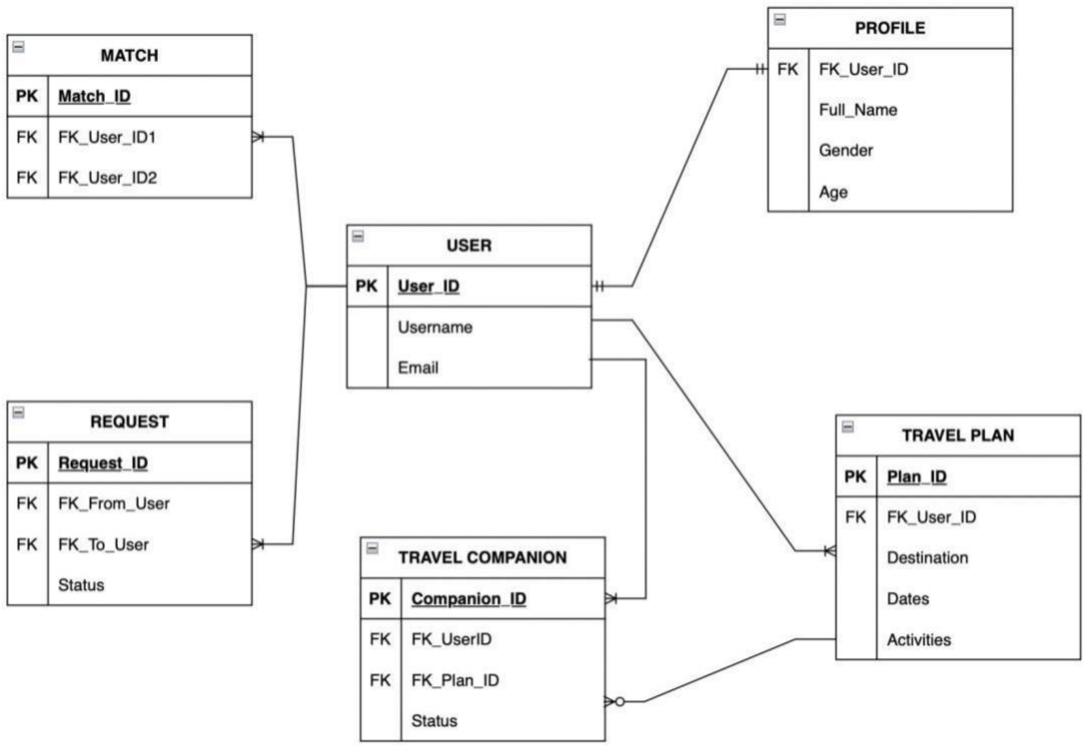
Safety Measures: To ensure safety, users can report inappropriate behavior or block other users.

Review System: Users can rate and review their travel companions after a trips.

Our project's goal is to create an application for users to find travel companions. This idea appeals to us because, as travel enthusiasts, we've experienced the challenges of finding companions when moving to a new city. Traveling alone can be daunting, and having travel companions enhances the experience, saves costs, and fosters lasting memories while promoting cultural connections.

Business Rules:

1. A User can create a Profile with their personal information.
2. A User can search for Travel Companions based on their travel plans and preferences.
3. A Travel Companion can be matched with one or more Users based on shared travel interests and destinations.
4. A Travel Plan includes details such as the destination, dates, and preferred activities.
5. A Travel Companion can send a Request to connect with another User for a specific trip.
6. A Request can have a status (e.g., pending, accepted, declined) to track its progress.
7. A match is established when both Travel Companions accept each other's requests.



USER Entity:
Attributes: user_id (Primary Key), username, email, and other user-related details.
Explanation: The USER entity represents individuals who use the application. Each user has a unique user_id as their primary key, and they can provide personal information like their username and email.

PROFILE Entity:
Attributes: user_id (Primary Key, Foreign Key), full_name, gender, age, and other personal details.
Explanation: PROFILE is associated with USER through a one-to-one relationship. It stores additional personal information about each user, such as their full name, gender, and age.

TRAVEL PLAN Entity:
Attributes: plan_id (Primary Key), user_id (Foreign Key), destination, status, and other travel-related details.
Explanation: TRAVEL PLAN represents a user's travel itinerary. Users can create multiple travel plans, and each plan is linked to a user through the user_id foreign key.

TRAVEL COMPANION Entity:
Attributes: user_id (Primary Key, Foreign Key), plan_id (Primary Key, Foreign Key), status, and other companion-related details.
Explanation: TRAVEL COMPANION connects users who share travel interests. It forms a many-to-many relationship between users and their travel plans, allowing users to connect with multiple companions for different trips.

REQUEST Entity:
Attributes: request_id (Primary Key), from_user (Foreign Key), to_user (Foreign Key), status, and match_id (Foreign Key).
Explanation: REQUEST is used for one user to send a travel companion request to another user. It has relationships with both from_user and to_user, indicating the sender and receiver of the request. When a request is accepted, it forms a match, and match_id is linked to the match entity.

MATCH Entity:
Attributes: match_id (Primary Key), FK_user_id1 (Foreign Key), FK_user_id2 (Foreign Key).
Explanation: MATCH represents a mutual acceptance of a travel partnership between users. It is associated with both users involved in the match through FK_user_id1 and FK_user_id2 foreign keys.

RELATIONSHIPS:

USER to **PROFILE** Relationship:
Each **USER** can have one corresponding **PROFILE**.
This is a one-to-one (1:1) relationship, as each user has a single profile.

USER to **TRAVEL PLAN** Relationship:
Each **USER** can create multiple **TRAVEL PLANS**, representing different trips.
This is a one-to-many (1:M) relationship, as one user can have many travel plans.

USER to **TRAVEL COMPANION** Relationship:
Each **USER** can be associated with multiple **TRAVEL COMPANIONS** based on different trips and connections.
This is a one-to-many (1:M) relationship, as one user can have multiple travel companions.

USER to **REQUEST** Relationship:
Each **USER** can send or receive multiple **REQUESTS** for travel companionship.
This is a many-to-many (M:M) relationship, as one user can have multiple requests, and each request involves two users (sender and receiver).

REQUEST to **MATCH** Relationship:
When both users involved in a request accept it, a **MATCH** is established.
This is a one-to-one (1:1) relationship, as each request results in a single match.

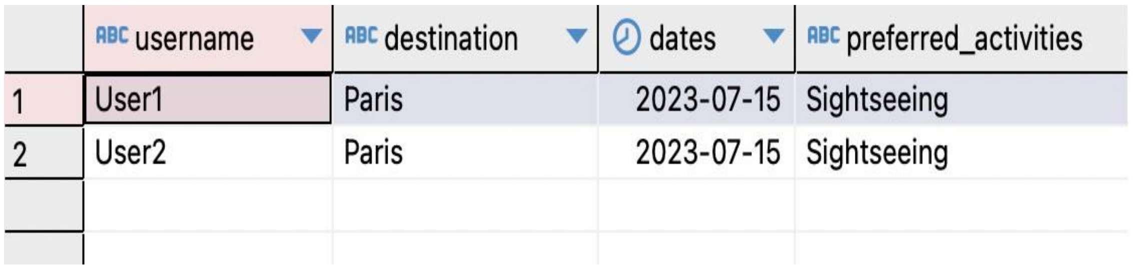
TRAVEL PLAN to **TRAVEL COMPANION** Relationship:
Each **TRAVEL PLAN** can be associated with multiple **TRAVEL COMPANIONS** who have shown interest in the same trip.
This is a one-to-many (1:M) relationship, as one travel plan can have multiple potential travel companions.

Data Implementation

Here are SQL commands representing major business rules for the "Travel Companion Finder" application:

Use Case 1: User Searches for Travel Companions

```
SELECT U.username, T.destination, T.dates, T.preferred_activities
FROM USER AS U
JOIN PROFILE AS P ON U.user_id = P.user_id
JOIN MATCH AS M ON U.user_id = M.user_id1 OR U.user_id = M.user_id2
JOIN TRAVEL_PLAN AS T ON M.user_id1 = U.user_id OR M.user_id2 = U.user_id
WHERE T.destination = 'Paris' AND T.preferred_activities LIKE '%Sightseeing%';
```

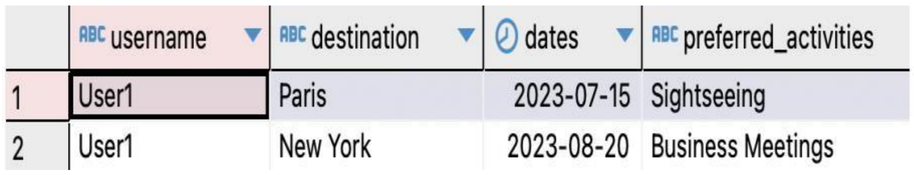


	username	destination	dates	preferred_activities
1	User1	Paris	2023-07-15	Sightseeing
2	User2	Paris	2023-07-15	Sightseeing

Description: This SQL query retrieves a list of potential travel companions based on a specific destination and preferred activities. It joins multiple tables to gather user and travel plan information.
Use Case: A user wants to find travel companions for a trip to a specific destination with shared interests.

Use Case 2: Travel Companion Match

```
SELECT U.username, T.destination, T.dates, T.preferred_activities
FROM USER AS U
JOIN MATCH AS M ON U.user_id = M.user_id1 OR U.user_id = M.user_id2
JOIN TRAVEL_PLAN AS T ON M.user_id1 = U.user_id OR M.user_id2 = U.user_id
WHERE U.user_id = 1;
```



	username	destination	dates	preferred_activities
1	User1	Paris	2023-07-15	Sightseeing
2	User1	New York	2023-08-20	Business Meetings

Description: This SQL query retrieves a list of potential travel companions based on a specific destination and preferred activities. It joins multiple tables to gather user and travel plan information.
Use Case: A user wants to find travel companions for a trip to a specific destination with shared interests.

Use Case 3: User Creates a Travel Plan

```
INSERT INTO TRAVEL_PLAN (destination, dates, preferred_activities)
VALUES ('Rome', '2023-09-15', 'Exploring historical sites');
```

Description: This SQL command inserts a new travel plan with details such as the destination, dates, and preferred activities into the TRAVEL_PLAN table.
Use Case: A user plans a new trip and creates a travel plan to share with potential travel companions.

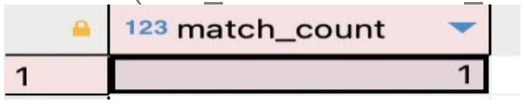
Use Case 4: Request Status Update

```
UPDATE REQUEST
SET status = 'Accepted'
WHERE request_id = 1;
```

Description: This SQL command updates the status of a request from "Pending" to "Accepted" in the REQUEST table. It's used when a request is accepted by the recipient.
Use Case: A user accepts a travel companion request, and the request status is updated.

Use Case 5: Match Establishment

```
SELECT COUNT(*) AS match_count
FROM MATCH
WHERE (user_id1 = 1 AND user_id2 = 2) OR (user_id1 = 2 AND user_id2 = 1);
```



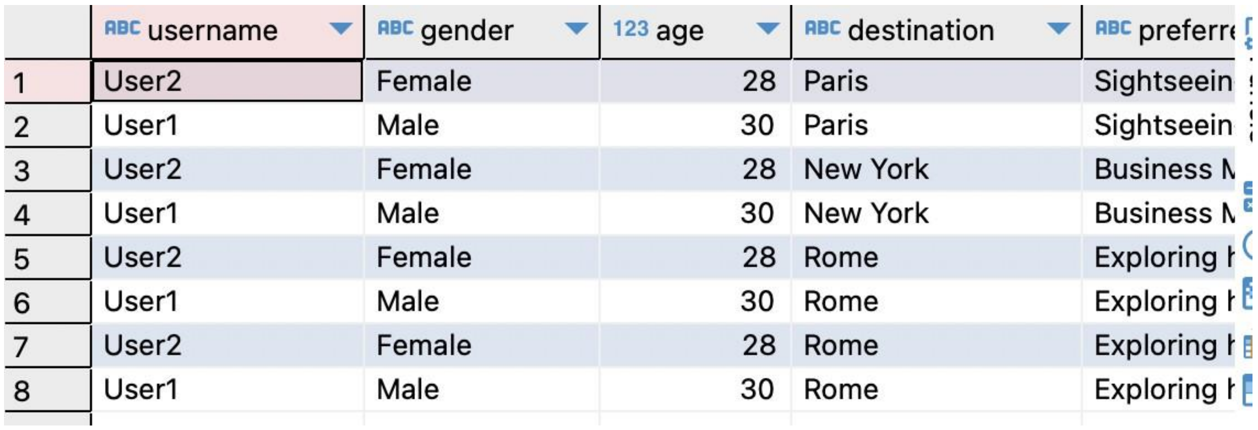
	match_count
1	1

Description: This SQL query checks if a match is established between two users by searching the MATCH table for matching user IDs.
Use Case: When both travel companions accept each other's requests, a match is established, and this query verifies the match status.

ANALYTICS, REPORTS AND METRICS

User Travel Interests:

```
SELECT U.username, P.gender, P.age, T.destination, T.preferred_activities
FROM USER AS U
JOIN PROFILE AS P ON U.user_id = P.user_id
JOIN MATCH AS M ON U.user_id = M.user_id1 OR U.user_id = M.user_id2
JOIN TRAVEL_PLAN AS T ON M.user_id1 = U.user_id OR M.user_id2 = U.user_id;
```

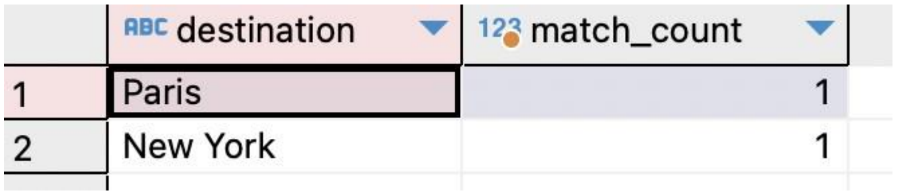


	username	gender	age	destination	preferre
1	User2	Female	28	Paris	Sightseein
2	User1	Male	30	Paris	Sightseein
3	User2	Female	28	New York	Business M
4	User1	Male	30	New York	Business M
5	User2	Female	28	Rome	Exploring H
6	User1	Male	30	Rome	Exploring H
7	User2	Female	28	Rome	Exploring H
8	User1	Male	30	Rome	Exploring H

Description: This report provides a list of users along with their travel interests, such as destination and preferred activities. It helps in understanding user preferences.

Travel Companion Matching Statistics:

```
SELECT T.destination, COUNT(*) AS match_count
FROM TRAVEL_PLAN AS T
JOIN MATCH AS M ON T.plan_id = M.user_id1 OR T.plan_id = M.user_id2
GROUP BY T.destination;
```

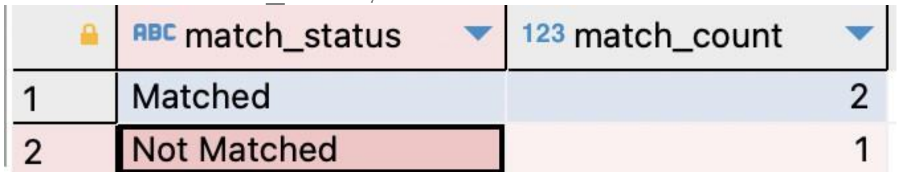


	destination	match_count
1	Paris	1
2	New York	1

Description: This report displays the number of matches for each travel destination, helping to identify popular travel spots.

Matching Success Rate

```
SELECT CASE
      WHEN R.status = 'Accepted' THEN 'Matched'
      ELSE 'Not Matched'
      END AS match_status,
      COUNT(*) AS match_count
FROM REQUEST AS R
GROUP BY match_status;
```



	match_status	match_count
1	Matched	2
2	Not Matched	1

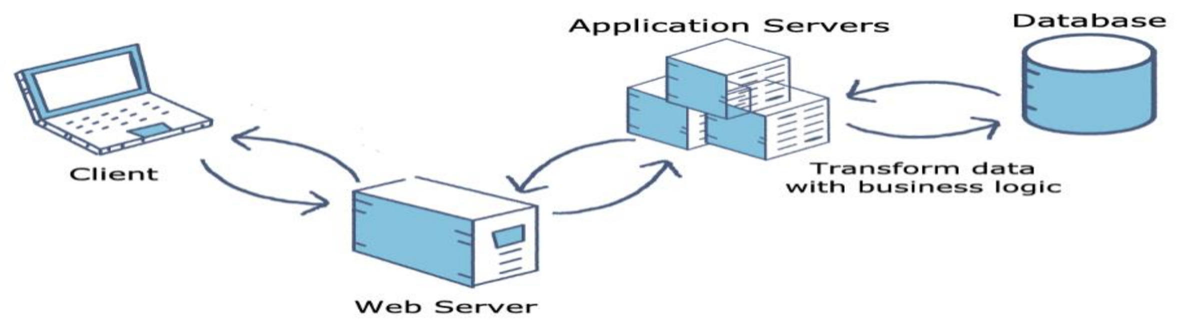
Description: This report presents the success rate of travel companion matches by categorizing them as "Matched" or "Not Matched" based on the status of the request.

Project Architecture Requirements

For the **"Trek Tribe"** project, our architecture requirements involve a client-server model, cloud hosting, and a structured approach to data storage.

1. Client-Server Architecture:
Our solution follows a client-server architecture, where clients (end-users) interact with a central server to access, store, and manage data. The client, which can be a web or mobile application, communicates with the server via APIs and protocols.

2. Hosting Model:
We have opted for cloud hosting for our application and database. Specifically, we are considering using a cloud platform like Amazon Web Services (AWS), Azure, or Google Cloud Platform (GCP). Cloud hosting offers several advantages:
Scalability: The cloud environment allows us to scale resources as needed to accommodate growing user numbers and data volumes.
Availability: Cloud platforms provide high availability, ensuring the app is accessible to users around the clock.
Redundancy: Data redundancy and backups are handled efficiently, reducing the risk of data loss.
Maintenance: Cloud hosting simplifies maintenance tasks, including software updates and security patches.



3. Storage Requirements:
Estimating the storage naeeds for our project is subject to factors like user base and data volume. The primary data entities in our database include user profiles, travel plans, requests, and matches. For a moderate-sized project, we anticipate needing several gigabytes of storage, depending on the number of users and their data. As the user base grows, we can scale storage resources accordingly.

Security Concerns:
- User Privacy: User profiles contain personal information, including names, gender, and age. Ensuring that this data is stored securely is essential to protect user privacy.

- Authentication: The system should implement strong user authentication to prevent unauthorized access to user accounts.

- Data Encryption: Travel plans may include travel dates, which could potentially reveal when users are away from home. Implementing data encryption can safeguard this information.

- Access Control: Restricting access to the database to authorized personnel and ensuring that only necessary data is exposed to each user role can prevent unauthorized data leakage.

- Request Status Handling: The handling of request statuses is crucial to prevent information leaks, especially if a user's request is declined.

Project Wrap-up and Future Considerations:

As a result of this project, I've gained valuable insights into the significance of database design, data modeling, and the optimization of queries. The key takeaways from this experience are as follows:

- Data Modeling: It has become evident that meticulous deliberation of data entities, their interrelationships, and adherence to established business rules is fundamental for creating a well-structured database.

- Query Optimization: I've learned that the formulation of well-organized SQL queries is imperative to ensure efficient data retrieval and modification processes.

- Security Awareness: Recognizing the various security considerations and privacy implications associated with handling user data is of paramount importance.

- Cloud-Based Architecture: I now understand that cloud-based solutions offer the advantages of scalability and reliability, making them applicable to a diverse array of applications.

In terms of future endeavors, I plan to further enhance my skills in database design, stay informed about evolving security best practices, and delve into advanced techniques for data analytics and reporting. This will enable me to provide even more valuable insights to stakeholders in future projects.

REFERENCES

1. How to draw Entity Relationship Diagrams (ERDs). (2020). *Gliffy by Perforce*. https://www.gliffy.com/blog/how-to-draw-an-entity-relationship-diagram
2. Kumar, A. (2023, September 1). Why User Persona is Necessary For Your App? - Appventurez. *Appventurez*. https://www.appventurez.com/blog/mobile-app-user-persona
3. *SQL Tutorial*. (n.d.). https://www.w3schools.com/sql/default.asp
4. *What is Client-Server Networking? Definition, Advantages, and Disadvantages - zenarmor.com*. (2023, October 4). https://www.zenarmor.com/docs/network-basics/what-is-client-server-network
5. *DBMS Architecture - Detailed explanation*. (2023, October 17). InterviewBit. https://www.interviewbit.com/blog/dbms-architecture/
6. Gulati, V. (2022, January 13). DBMS Architecture - Scaler Topics. *Scaler Topics*. https://www.scaler.com/topics/dbms/dbms-architecture/
7. *Major database security threats & How you can Prevent them*. (n.d.). Tripwire. https://www.tripwire.com/state-of-security/major-database-security-threats-prevent