Tour and Travel DBMS Project

1. **Introduction**

The project is aimed at building a Tour and Travel Management System that helps users plan their trips. The system provides information about various travel destinations, booking services, and tour packages. The system is built using a Database Management System (DBMS) to handle all the backend data operations, including storing travel destinations, customer details, booking information, payment, and tour details.

Scope:

- Store customer information, their booking details, and payment records.
- Maintain data about available tours, packages, and destinations.
- Enable users to view available tours, make bookings, and pay for their tours online.
- Provide the admin with functionality to manage customers, bookings, and available tours.

2. Types of Users:-

2.1. **Admin User**:

- The admin has access to manage all aspects of the system.
- Can add, update, or delete tour packages, destinations, and customer records.
- Can view customer bookings, payments, and history.

2.2. **Customer User**:

- Customers can browse available tour packages and destinations.
- Can book a tour and make payments.
- Can view their past bookings and payment history.

3. **Objectives / Purpose:**

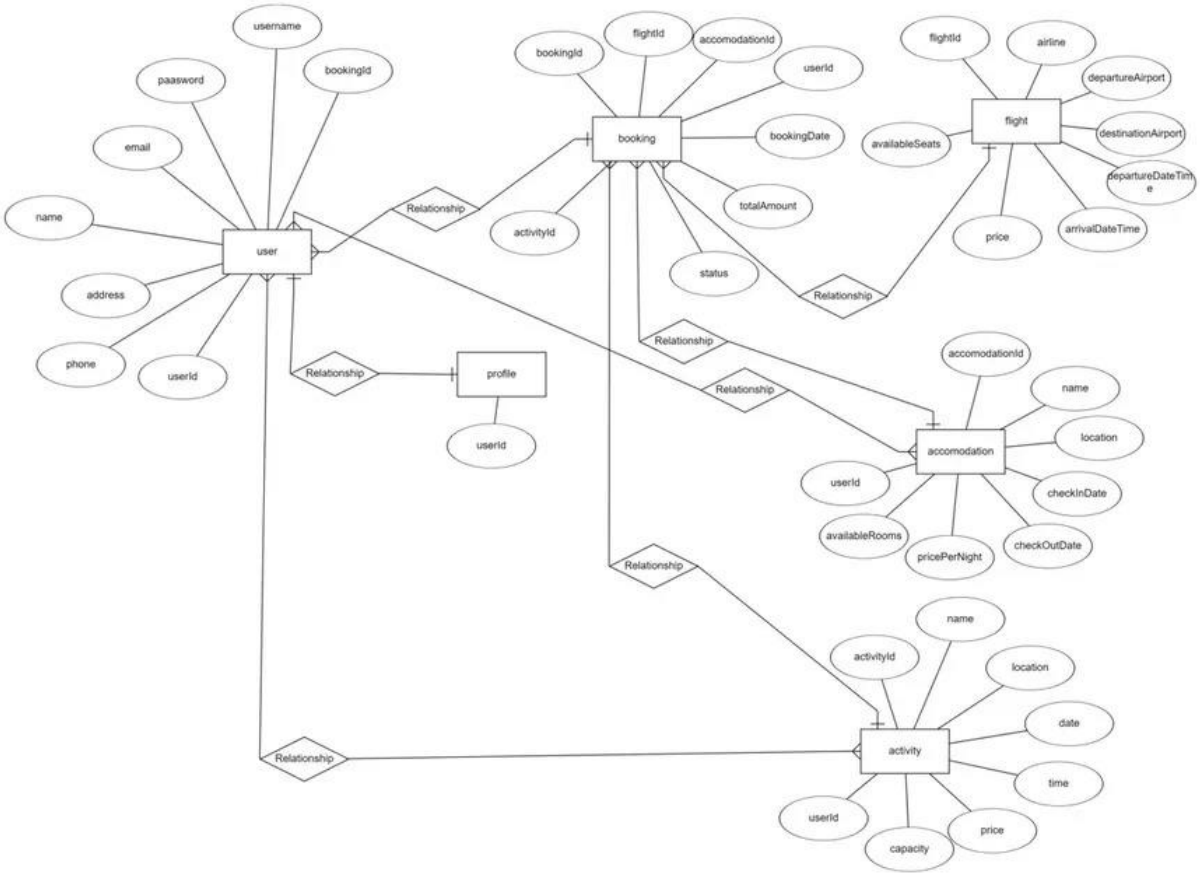
The main objectives of the Tour and Travel DBMS Project are:

- To organize and manage data related to customers, bookings, and travel packages efficiently.
- To provide convenience for both the admin and customer by allowing easy access to relevant information.
- To create a robust and scalable database to handle growing amounts of data as more customers and tours are added.
- To improve booking accuracy and help customers easily find the right tours based on their preferences.
- To facilitate secure transactions by managing payment records.

4. Designing: ER Model and Relational Diagram

4.1. ER Model

1. Entities:
2. Customer: Customer information like name, address, contact details, etc.
3. Tour: Information on various tour packages, including destinations, duration, price, etc.
4. Booking: A record of customer bookings, including the customer, tour, booking date, and status.
5. Payment: Payment information associated with a booking, such as amount, payment status, date, etc.
6. Destination: Details about travel destinations like name, country, and description.



4.2. **Relational Diagram**

Tables:

- Customer (CustomerID, Name, Address, Email, Phone, etc.)
- Tour (TourID, TourName, Description, Duration, Price, DestinationID)
- Booking (BookingID, CustomerID, TourID, BookingDate, Status)
- Payment (PaymentID, BookingID, PaymentAmount, PaymentDate, PaymentStatus)
- Destination (DestinationID, DestinationName, Country, Description)

The relational diagram should define the relationships between tables:

- Customer to Booking: One-to-many (A customer can make many bookings).

- Tour to Booking: One-to-many (A tour can be booked by many customers).
- Booking to Payment: One-to-one (Each booking corresponds to one payment).
- Destination to Tour: One-to-many (A destination can be part of many tours).

5. **Database: Tables and Full Data**

Customer Table

CustomerID	Name	Address	Email	Phone
1	John Doe	123 Elm Street, NY	john@example.com	123-456-7890
2	Alice Smith	456 Oak Street, CA	alice@example.com	987-654-3210

Tour Table

TourID	TourName	Description	Duration	Price	DestinationID
1	Paris Vacation	Visit the Eiffel Tower	7 days	2000	1
2	Bali Adventure	Beach and culture tour	5 days	1500	2

Booking Table

BookingI	CustomerID	TourID	BookingDate	Status
1	1	1	2025-04-01	Confirmed
2	2	2	2025-04-02	Pending

Payment Table

PaymentID	BookingID	PaymentAmount	PaymentDate	PaymentStatus
1	1	2000	2025-04-02	Completed
2	2	1500	2025-04-03	pending

Destination Table

DestinationID	DestinationName	Country	Description
1	Paris	France	Iconic city with historic landmarks
2	Bali	Indonesia	Famous for beaches and culture

6. **Coding: Queries**

6.1. **Creative DML Queries**

Retrieve all bookings for a specific customer:

sql

```
SELECT * FROM Booking WHERE CustomerID = 1;
```

Find available tours:

sql

```
SELECT Tour.TourName, Tour.Description, Tour.Price
FROM Tour
WHERE Tour.TourID NOT IN (SELECT TourID FROM Booking WHERE Status =
'Confirmed');
```

Calculate total payment for all completed bookings:

sql

```
SELECT SUM(PaymentAmount) AS TotalRevenue
FROM Payment
WHERE PaymentStatus = 'Completed';
```

Insert a new customer:

sql

```
INSERT INTO Customer (Name, Address, Email, Phone)
```

VALUES ('Emma Johnson', '789 Pine St, TX', 'emma@example.com', '555-1234');

Update booking status:

sql

UPDATE Booking

SET Status = 'Confirmed'

WHERE BookingID = 2;

7. **Result/Output of Code**

- After executing the above queries, the following results are expected:
- The customer bookings are displayed based on their customer ID.
- Available tours are shown that have not been booked yet.
- The total revenue from all completed payments is displayed.
- New customer data is added successfully.
- Booking status is updated correctly.

8. **Conclusion**

This Tour and Travel DBMS Project successfully integrates various database management functionalities for an online tour and travel system. The database is designed to store and manage customer information, booking details, payments, and tour packages efficiently. By using SQL queries for data retrieval, insertion, and updates, the system allows both the admin and customers to perform tasks effectively.

The project has shown how a DBMS can provide a foundation for real-world applications in travel management.

9. **References**

SQL Documentation - https://www.w3schools.com/sql/

Database Design Concepts - C. J. Date, An Introduction to Database Systems.

Travel Website Examples - https://www.expedia.com/