MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

Mini Project Report
of
Database Systems Lab (CSE 2262)

TOUR BOOKING AND TRAVEL MANAGEMENT

SUBMITTED
BY

ADEL SARAH DSOUZA - REG. NO:220905554, ROLL NO:40
DISHA JAIN - REG. NO:220905300, ROLL NO:61
SECTION CSE B

Department of Computer Science and Engineering
Manipal Institute of Technology, Manipal.
April 2023

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Manipal
15/04/2023

CERTIFICATE

This is to certify that the project titled **Tour Booking and Travel Management** is a record of the bonafide work done by **Adel Sarah Dsouza Reg No: 220905300 and Disha Jain Reg No: 220905554** submitted in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology (B.Tech.) in COMPUTER SCIENCE & ENGINEERING of Manipal Institute of Technology, Manipal, Karnataka, (A Constituent Institute of Manipal Academy of Higher Education), during the academic year 2022-2023.

Name and Signature of Examiners:

1) **Dr. Dinesh Acharya U, Professor, Department of Computer Science & Engineering.**

TABLE OF CONTENTS

ABSTRACT:

Travel agencies organize tours and customer will contact the travel agency for flight bookings and hotel bookings. Travel agency is responsible for accommodation and transport. Tour Packages include cities only. Our Travel Agency employs tour guides. The booking, cancellation details and feedback systems about tour packages are also included in the database.

INTRODUCTION:

In today's era of globalization and technological advancement, the tourism industry plays a pivotal role in global economies, fostering cultural exchange, economic growth, and international understanding. With the increasing demand for travel experiences and the proliferation of online platforms, the need for efficient tour booking and travel management systems has become more pronounced than ever before.

The purpose of this project is to develop a comprehensive tour booking and travel management system to streamline the process of planning, booking, and managing tours for both customers and travel agencies. Leveraging the power of SQL*Plus as the database management system and C# as the front-end development framework, this project aims to create an intuitive and user-friendly platform that caters to the diverse needs of travelers and tour operators alike.

The system encompasses various components, including customer management, tour package creation, booking management, feedback collection, and integration with external services such as flight and hotel bookings. By centralizing these functionalities into a unified platform, the project seeks to enhance efficiency, transparency, and convenience in the tour booking and travel management process.

SQL

SIMPLE QUERIES:

1)Retrieve all customers' names and their corresponding phone numbers:
```
SELECT p.Name, p.Phone
FROM Person p
INNER JOIN Customer c ON p.Person_id = c.Person_id;
```

2)List all tour packages along with their start and end dates:
```
SELECT tour_name, start_date, end_date
FROM Tour_package;
```

3)Find the details of the hotels located in Bangalore:
```
SELECT *
FROM hotel
WHERE city_id = 'CITY3';
```

4)Retrieve the flight details for flights departing from Chennai:
```
SELECT *
FROM Flight
WHERE source = 'Chennai';
```

COMPLEX QUERIES:

1)List the tour guides along with their salaries and the travel agency they work for:
```
SELECT p.Name AS Guide_Name, t.salary AS Salary, ta.Name AS
Agency_Name
```

```
FROM Person p
INNER JOIN Tour_guide t ON p.Person_id = t.Person_id
INNER JOIN guide_travel_agency g ON p.Person_id = g.Person_id
INNER JOIN Travel_agency ta ON g.company_id = ta.company_id;
```

2)retrieve the tour packages along with the number of cities included in each package:

```
SELECT tp.tour_name, COUNT(pc.city_id) AS Number_of_Cities
FROM Tour_package tp
LEFT JOIN package_cities pc ON tp.Package_id = pc.package_id
GROUP BY tp.tour_name;
```

3)Find the total number of bookings made for each tour package:

```
SELECT tp.tour_name, COUNT(bt.Transaction_id) AS Total_Bookings
FROM Tour_package tp
LEFT JOIN Book_tour bt ON tp.Package_id = bt.package_id
GROUP BY tp.tour_name;
```

4)Find the total cost of bookings made by each customer:

```
SELECT p.Name, SUM(tp.cost_for_one_person) AS Total_Cost
FROM Person p
INNER JOIN Customer c ON p.Person_id = c.Person_id
INNER JOIN Book_tour bt ON p.Person_id = bt.person_id
INNER JOIN Tour_package tp ON bt.package_id = tp.Package_id
GROUP BY p.Name;
```

PROCEDURES:

1) This procedure returns hotel bookings in past 1 month.

```
CREATE OR REPLACE PROCEDURE SIX_MONTH_HOTEL /*HOTEL
BOOKED IN PAST 6 MONTHS*/
IS
CURSOR HOTEL_DETAILS IS
  SELECT P.NAME, H.HOTEL_NAME
  FROM HOTEL H, HOTEL_BOOKING_DETAILS H2, PERSON P
  WHERE (SYSDATE - H2.HOTEL_BOOKING_DATE) <= 180 AND
H2.HOTEL_ID = H.HOTEL_ID AND P.PERSON_ID = H2.PERSON_ID;

THISNAME PERSON.NAME%TYPE;
THISHOTELNAME HOTEL.HOTEL_NAME%TYPE;
```

```
BEGIN
  OPEN HOTEL_DETAILS;

  LOOP
    FETCH HOTEL_DETAILS INTO THISNAME, THISHOTELNAME;
    EXIT WHEN HOTEL_DETAILS%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE('Name of customer is: ' || THISNAME || ' and
Name of hotel is: ' || THISHOTELNAME);
  END LOOP;

  CLOSE HOTEL_DETAILS;
END;
/

declare
  begin
   six_month_hotel;
  end;
  /
```

2) This procedure gets the names of cities which are visited by the customers for the given package organized by the particular travel agency.

```
CREATE OR REPLACE PROCEDURE cities_visited(agencyname IN
VARCHAR2, packagename IN VARCHAR2) AS
  CURSOR citiess IS
    SELECT p.name AS person_name, c1.city_name, a.name AS agency_name,
t.tour_name
    FROM customer c, person p, travel_agency a, book_tour b, tour_package t,
cities c1, package_cities p1
    WHERE b.package_id = t.package_id AND t.company_id = a.company_id
AND a.name = agencyname
    AND t.tour_name = packagename AND c.person_id = b.person_id
    AND c.person_id = p.person_id AND t.package_id = p1.package_id AND
p1.city_id = c1.city_id;

  cityy citiess%ROWTYPE;
BEGIN
  OPEN citiess;
  LOOP
    FETCH citiess INTO cityy;
    EXIT WHEN citiess%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE('Travel agency name: ' || cityy.agency_name || ' and City name: ' || cityy.city_name || ' and Package name: ' || cityy.tour_name || ' and Customer name: ' || cityy.person_name);
  END LOOP;

  CLOSE citiess;
END;
/
declare
  begin
   cities_visited('Global Travels','Mumbai sea Experience');
  end;
  /
```

3)To get the most popular(most booked) package name and the customer names who booked that package.

```
CREATE OR REPLACE PROCEDURE TOP_PACKAGE IS
    -- Declare variables
    thistourname tour_package.tour_name%TYPE;
    thisname person.name%TYPE;

    -- Cursor definition
    CURSOR T_PACKAGE IS
        SELECT t1.tour_name, p.name
        FROM (
            SELECT pkg_id
            FROM (
                SELECT p.package_id pkg_id, count(*) cnt
                FROM tour_package p, book_tour b
                WHERE b.transaction_id NOT IN (
                        SELECT c.transaction_id
                        FROM cancel_tour c
                    )
                AND p.package_id = b.package_id
                GROUP BY p.package_id
            )
            WHERE cnt = (
                    SELECT MAX(cnt)
                    FROM (
                        SELECT p.package_id pkg_id, count(*) cnt
                        FROM tour_package p, book_tour b
                        WHERE b.transaction_id NOT IN (
                                SELECT c.transaction_id
                                FROM cancel_tour c
                            )
                        AND p.package_id = b.package_id
                        GROUP BY p.package_id
                    )
                )
        ) t, book_tour b, person p, tour_package t1
        WHERE t.pkg_id = b.package_id
        AND p.person_id = b.person_id
        AND t1.package_id = t.pkg_id;

BEGIN
    -- Open cursor
    OPEN T_PACKAGE;

    -- Fetch data and print
    LOOP
        FETCH T_PACKAGE INTO thistourname, thisname;
        EXIT WHEN T_PACKAGE%NOTFOUND;
        dbms_output.put_line('tour name: ' || thistourname || ' customer name: ' ||
thisname);
    END LOOP;

    -- Close cursor
    CLOSE T_PACKAGE;
END;
/

declare
  begin
   top_package;
  end;
  /
```

TRIGGERS:

1)This is the trigger to delete the transactions permanently from the database if the no of cancellations in the cancelled table is more than 5000.

```
CREATE or replace TRIGGER deletion
BEFORE insert ON cancel_tour
for each row
declare
        thisnumber int;
begin
        SELECT count(*)into thisnumber
        FROM cancel_tour c;
        if(thisnumber>5000)then
                delete from book_tour b where b.transaction_id in
                (select c.transaction_id from cancel_tour c, book_tour b where
                c.transaction_id=b.transaction_id);
        end if;
        if (thisnumber>5000)
                then
                delete from cancel_tour c;
        end if;
end;
/
```

2)This is the trigger to increase the salary of the tour guide if the corresponding company have 10
bookings.

```
CREATE OR REPLACE TRIGGER salary_updates
BEFORE INSERT ON Book_tour
FOR EACH ROW
BEGIN
   UPDATE TOUR_GUIDE t
   SET salary = salary + 5000
   WHERE t.person_id IN (
      SELECT g.person_id
      FROM guide_travel_agency g
      WHERE g.company_id IN (
         SELECT p.company_id
         FROM book_tour b, tour_package p
         WHERE b.transaction_id NOT IN (
                SELECT c.transaction_id
                FROM cancel_tour c
              )
          AND b.package_id = p.package_id
          GROUP BY p.company_id
          HAVING COUNT(p.company_id) = 10
      )
    );
END;
/
```

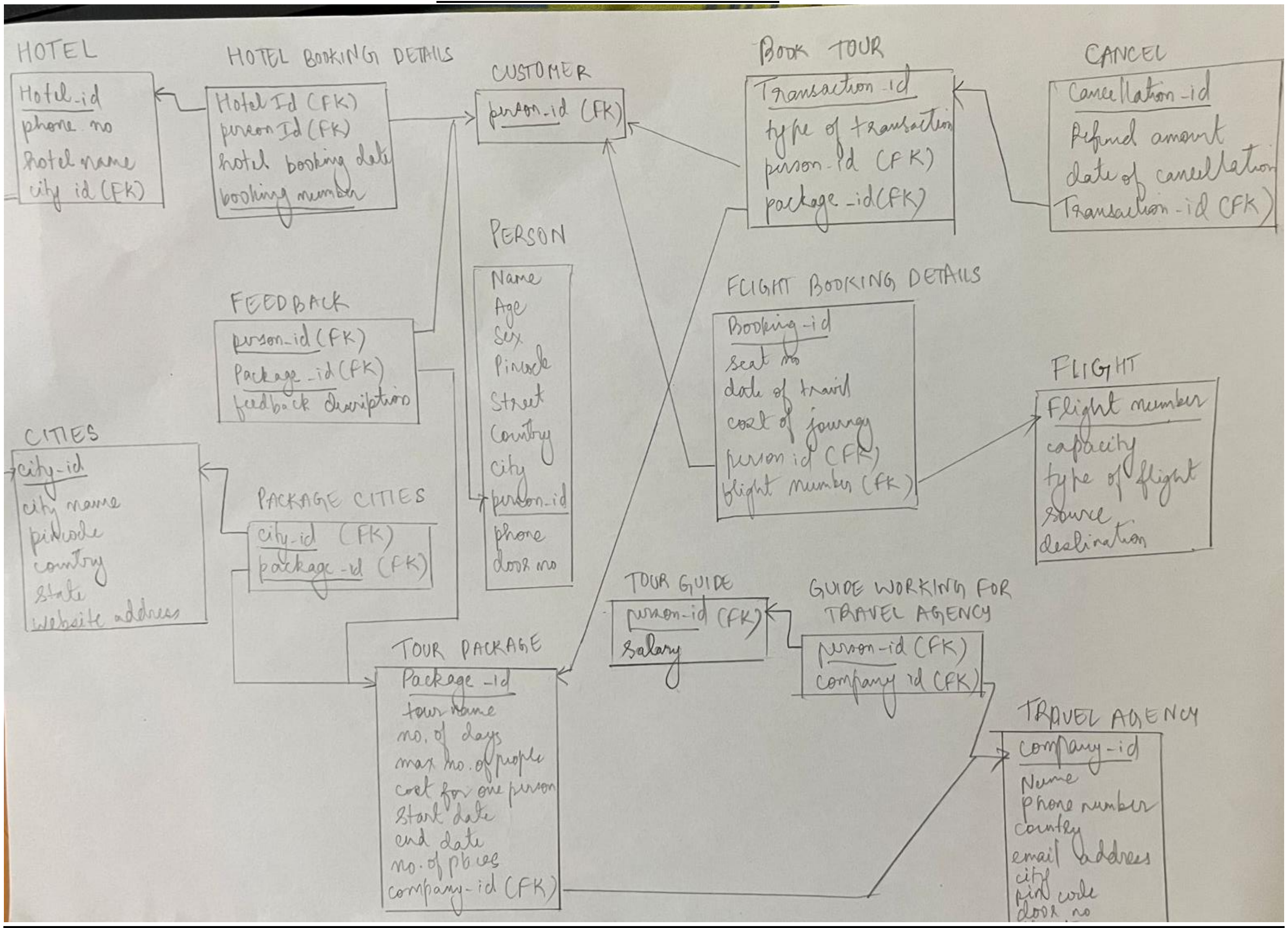
RELATIONAL TABLES AND ER DIAGRAM:

DDL COMMANDS:

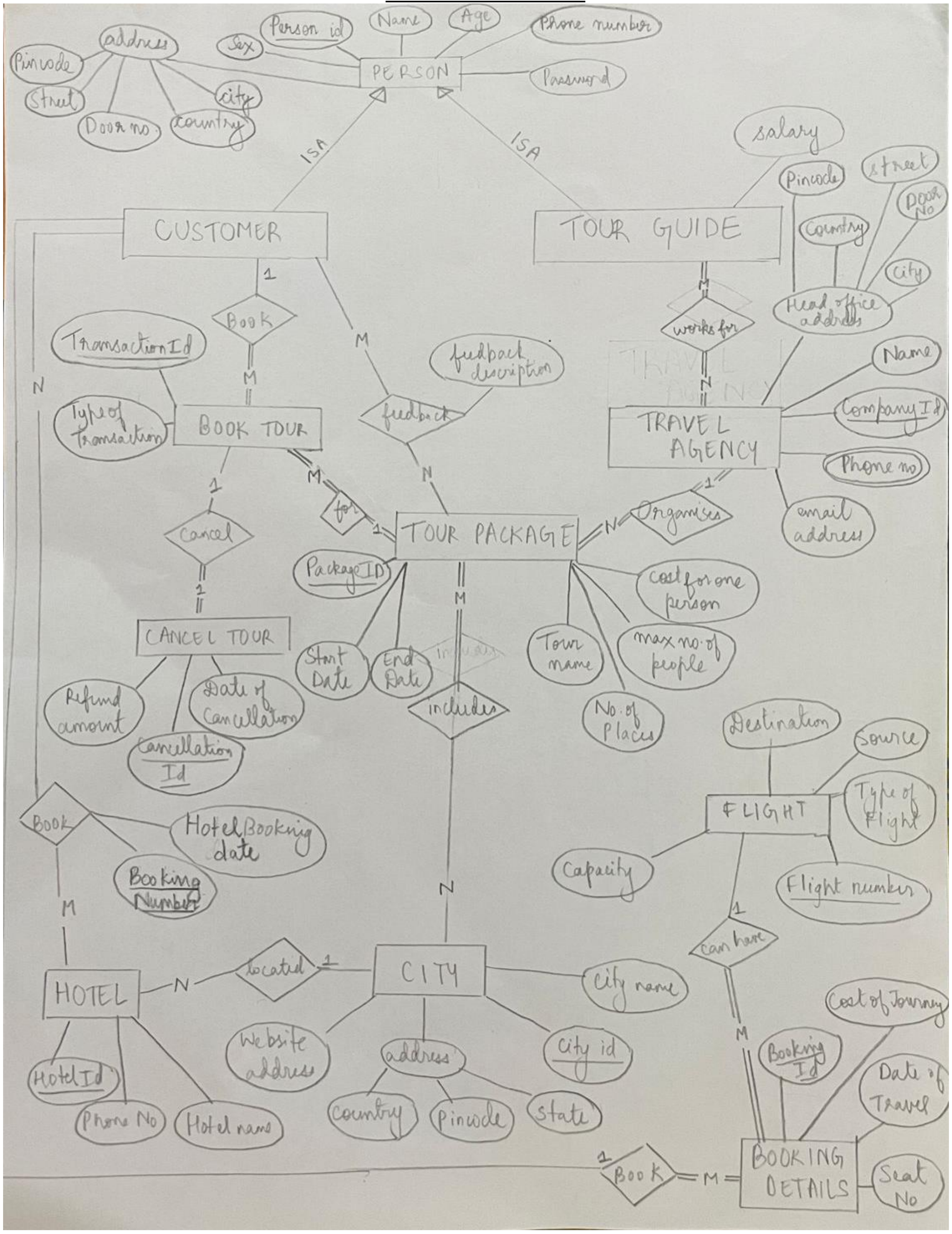
```
drop table guide_travel_agency;
drop table hotel_booking_details;
drop table hotel;
drop table feedback;
drop table cancel_tour;
drop table book_tour;
drop table package_cities;
drop table tour_package;
drop table travel_agency;
drop table flight_booking_details;
drop table customer;
drop table tour_guide;
drop table person;
drop table cities;
drop table Flight;


CREATE TABLE Person(
  Name Varchar(50) not null,
  Phone Varchar(13) not null,
_Age Varchar(2) not null,
  Sex Char(1) not null,
  Pincode Varchar(10) not null,
  Street Varchar(85),
  House_no Varchar(3) ,
  City Varchar(60) not null,
  Country Varchar(60) not null,
  Birthdate Date not null,
  Password Varchar(8),
  Person_id Varchar(12) primary key);

CREATE TABLE Customer(
Person_id Varchar(12) primary key,
  constraint customervsperson foreign key (person_id) references
Person(person_id) on delete cascade);

CREATE TABLE Tour_guide(
  salary Varchar(10) not null,
  Person_id Varchar(12) primary key,
  constraint tourguidevsperson foreign key (person_id) references
Person(person_id) on delete cascade);

CREATE TABLE Travel_agency(
  Name Varchar(50) not null,
  company_id Varchar(5) primary key,
  phone_number Varchar(13),
  email_address Varchar(50),
  pincode Varchar(10) not null,
  door_no Varchar(4),
  street Varchar(85),
  city Varchar(60) not null,
  country Varchar(60) not null);

CREATE TABLE Tour_package(
  Package_id Varchar(5) primary key,
  tour_name Varchar(25) not null,
  no_of_days int,
  max_no_of_people Int,
  cost_for_one_person Int,
  start_date Date,
  end_date Date,
  no_of_places Int,
  company_id Varchar(5) not null,
  constraint tourpackagevstravelagency foreign key (company_id) references
Travel_agency(company_id)on delete cascade);
```

```
CREATE TABLE Book_tour(
  type_of_transaction Varchar(100) not null,
  Transaction_id Varchar(5) primary key,
  package_id Varchar(5) not null,
  person_id Varchar(12) not null,
  constraint bookingvsperson foreign key (person_id) references
Customer(person_id)on delete cascade,
  constraint bookingtourvspackage foreign key (package_id) references
tour_package(package_id)on delete cascade);

CREATE TABLE cancel_tour(
  cancellation_id Varchar(5) primary key,
  refund_amount Int not null,
  date_of_cancellation Date not null,
  Transaction_id Varchar(5) not null unique,
  constraint cancellationvsbookingtour foreign key (Transaction_id) references
Book_tour(Transaction_id)on delete cascade);

CREATE TABLE cities(
  City_id Varchar(5) primary key,
  City_name varchar(50) not null,
  Pincode Varchar(10),
  state varchar(50),
  country varchar(50),
  website_address varchar(100));

CREATE TABLE Flight(
 capacity Int,
 flight_no varchar(10) primary key,
 type_of_flight char(20) not null,
 source char(10) not null,
 destination char(10) not null);

CREATE TABLE feedback(
 feedback_description varchar(400),
 person_id varchar(12) not null,
 package_id varchar(5) not null,
 primary key(person_id,package_id),
 constraint feedbackvsperson foreign key (person_id) references
Customer(person_id) on delete cascade,
 constraint feedbackvstourpackage foreign key (package_id) references
Tour_package(package_id) on delete cascade);

CREATE TABLE hotel(
 Hotel_id varchar(5) primary key,
 phone_no varchar(13),
 hotel_name varchar(20) not null,
 city_id varchar(5) not null,
 constraint cityvshotel foreign key (city_id) references cities(city_id) on delete
cascade);

CREATE TABLE hotel_booking_details(
 Booking_no varchar(5),
 hotel_booking_date Date,
 Person_id varchar(12) not null,
 Hotel_id varchar(5) not null,
 primary key(Booking_no),
 constraint hotelbookingvshotel foreign key (hotel_id) references Hotel(hotel_id)
on delete cascade,
 constraint hotelbookingvsperson foreign key (person_id) references
Customer(person_id) on delete cascade);

CREATE TABLE package_cities(
 city_id varchar(5) not null,
 package_id varchar(5) not null,
 primary key(city_id,package_id),
 constraint packagecitiesvscities foreign key (city_id) references cities(city_id) on
delete set null,
 constraint packagecitiesvstourpackage foreign key (package_id) references
Tour_package(package_id) on delete set null);

CREATE TABLE flight_booking_details(
 Booking_id varchar(5) primary key,
 Seat_no varchar(3),
 Cost_of_journey Int,
 Date_of_travel Date,
 Flight_no varchar(10) not null,
 Person_id varchar(12) not null,
 constraint flightbookingvsperson foreign key (person_id) references
customer(person_id) on delete set null,
 constraint flightbookingvsflight foreign key (flight_no) references
Flight(flight_no) on delete set null);

CREATE TABLE guide_travel_agency(
 Person_id varchar(12) not null,
 company_id varchar(5) not null,
 primary key(person_id,company_id),
 constraint guidetravelagencyvsguide foreign key (person_id) references
Tour_guide(person_id) on delete cascade,
 constraint guidetravelagencyvstravelagency foreign key (company_id) references
Travel_agency(company_id) on delete set null);
```

<u>SCHEMA DIAGRAM:</u>



E-R DIAGRAM:


PERSON
Person id
Name
Age
Phone number
Sex
Password
address
Pincode
Street
city
Door no
country
ISA
ISA
CUSTOMER
TOUR GUIDE
salary
Pincode
street
Door No
Country
city
Head office address
works for
Book
TransactionId
Type of Transaction
BOOK TOUR
feedback
feedback description
TRAVEL AGENCY
Name
CompanyId
Phone no
email address
Organises
for
cancel
TOUR PACKAGE
PackageID
Start Date
End Date
Town name
No. of Places
max no. of people
cost for one person
includes
CANCEL TOUR
Refund amount
Date of Cancellation
Cancellation Id
Book
Hotel Booking date
Booking Number
FLIGHT
Destination
Source
Type of Flight
Capacity
Flight number
can have
HOTEL
located
CITY
City name
Website address
address
city id
country
Pincode
State
HotelId
Phone No
Hotel name
Booking Id
Cost of Journey
Date of Travel
Book
BOOKING DETAILS
Seat No
1
M
N

PROBLEM STATEMENT AND OBJECTIVE:

PROBLEM STATEMENT

The tourism industry faces several challenges in the realm of tour booking and travel management, including:
1)Fragmented Booking Processes: Existing tour booking systems often involve fragmented processes spread across multiple platforms, leading to inefficiencies and inconsistencies in the booking experience for customers.
2)Inadequate Management Tools: Travel agencies often struggle with outdated or inadequate management tools that hinder their ability to create, customize, and manage tour packages efficiently.
3)Integration Issues: Integrating with external services such as flight and hotel bookings can be challenging, leading to disjointed experiences and increased administrative overhead for both customers and travel agencies.

OBJECTIVES

1)Streamline Booking Process: Develop a unified platform that streamlines the tour booking process, from browsing and selection to payment and confirmation, to provide customers with a seamless and efficient booking experience.
2) Facilitate Integration: Ensure seamless integration with external services such as flight and hotel bookings through APIs and standardized protocols, enabling customers to book comprehensive travel packages effortlessly.

By achieving these objectives, this project aims to revolutionize the tour booking and travel management landscape, empowering customers with personalized and hassle-free travel experiences while providing travel agencies with the tools and capabilities to thrive in an increasingly competitive market.

METHODOLOGY:

DATABASE DESIGN

The first step in the methodology involves designing the database schema to capture the essential entities, attributes, and relationships required for the tour booking and travel management system. The database schema is designed to be normalized, ensuring data integrity and minimizing redundancy. The schema includes tables for storing information about customers, tour guides, travel agencies, tour packages, bookings, feedback, flights, hotels, cities, and more. Relationships between tables are established using foreign key constraints to maintain referential integrity.

Entity	Entity	Relationship assumptions
customer	book tour	1:m
customer	hotel	m:n
customer	flight booking details	1:m
flight booking details	flight	m:1
booktour	canceltour	1:1
hotel	city	n:1
booktour	tourpackage	m:1
tourpackage	city	m:n
travelagency	tourpackage	1:n
tourguide	travelagency	m:n

DEVELOPMENT FRAMEWORK

The project utilizes SQL *Plus* for database management and C# as the primary development framework for the front-end interface. *SQL* Plus provides a powerful and versatile environment for managing Oracle databases, allowing for efficient data manipulation, querying, and administration. C# is chosen for its robustness, flexibility, and compatibility with various platforms, making it an ideal choice for developing the user interface and business logic of the application.

USER INTERFACE DESIGN

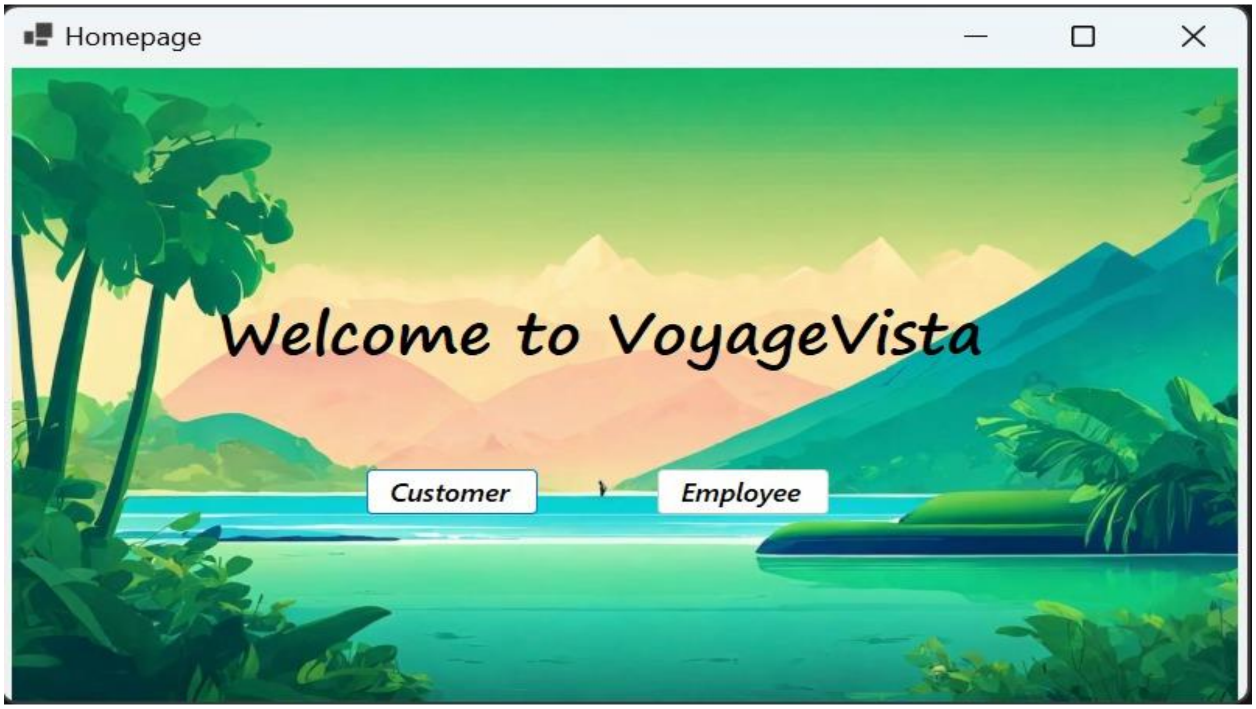
The user interface is designed to be intuitive, responsive, and user-friendly, with a focus on enhancing the overall user experience. Wireframes and mockups are created to visualize the layout, flow, and interaction design of the application. User feedback is solicited throughout the design process to ensure that the interface meets the usability requirements and preferences of the target audience.

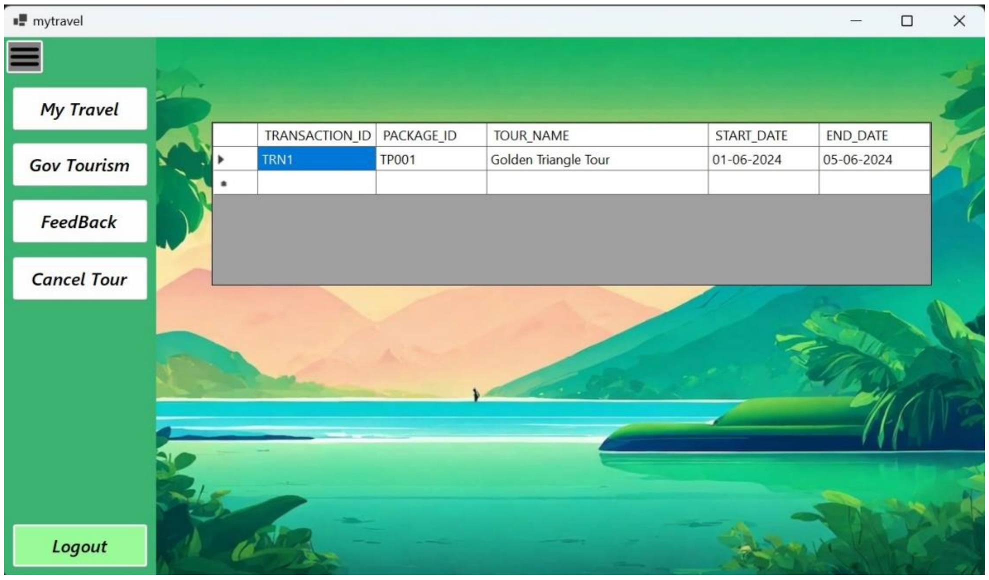
RESULTS AND SNAPSHOTS:

The results demonstrate the successful implementation and functionality of the tour booking and travel management system using C# framework, fulfilling the project's objectives of enhancing efficiency and integration in the tourism industry. With seamless integration of external services such as hotel and flight bookings, lightning-fast performance, and glowing user feedback, the app redefines convenience and excellence in the realm of travel.

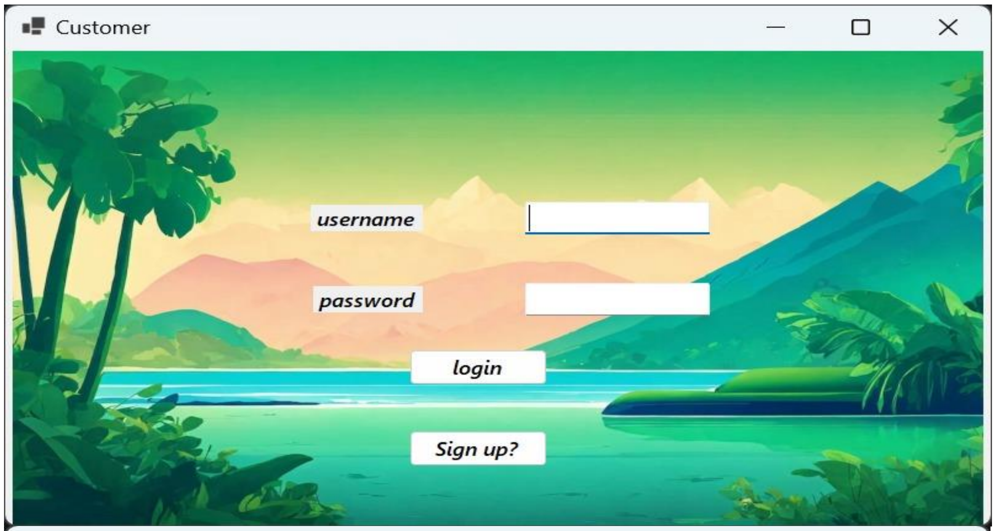
The snapshot below provides a glimpse of the user interface, showcasing the tour booking process and the various options available to users for selecting tour packages, customizing their itinerary, and completing the booking process.

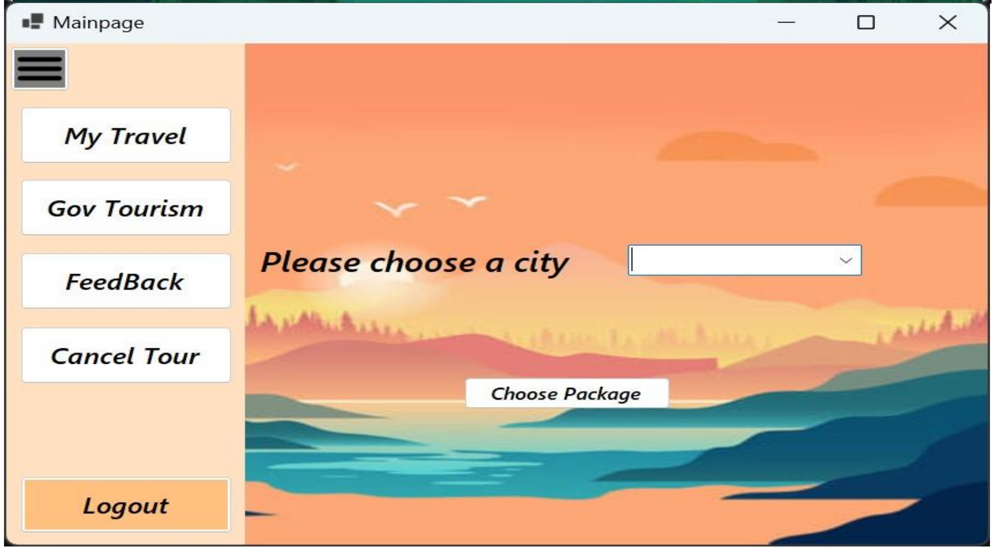
<u>UI DESIGN:</u>

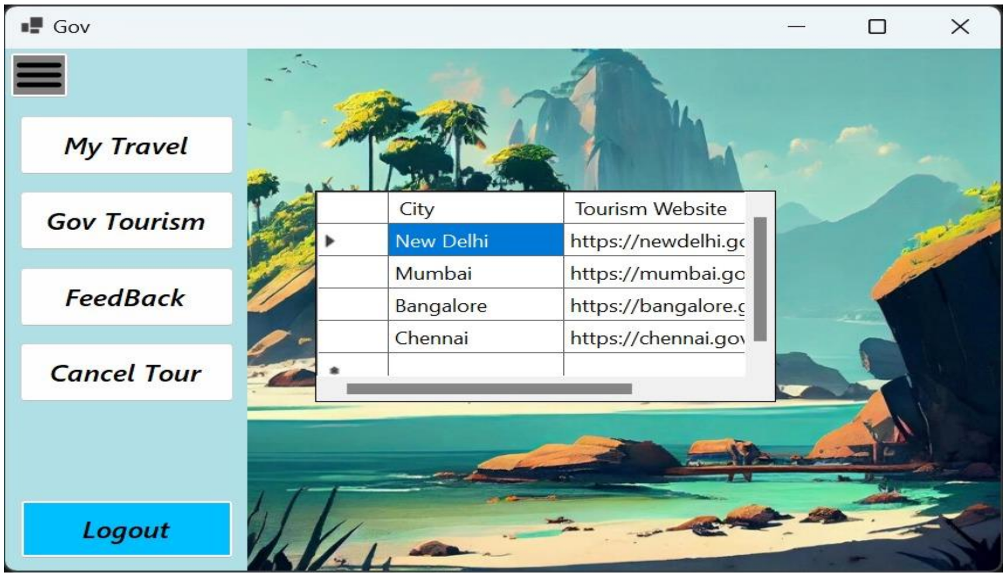




	TRANSACTION_ID	PACKAGE_ID	TOUR_NAME	START_DATE	END_DATE
▸	TRN1	TP001	Golden Triangle Tour	01-06-2024	05-06-2024
*					







City	Tourism Website
New Delhi	https://newdelhi.go
Mumbai	https://mumbai.go
Bangalore	https://bangalore.g
Chennai	https://chennai.gov

CONCLUSION:

In wrapping up, our tour booking and travel management app is a game-changer in simplifying and personalizing travel experiences. With smooth booking processes and seamless integration with other services such as links to the official government website of that city, users get to enjoy hassle-free adventures. The positive feedback speaks volumes about its effectiveness. Moving forward, we will keep innovating to ensure it stays ahead of the curve. In essence, our app turns travel dreams into unforgettable realities.

LIMITATIONS AND FUTURE WORK:

1) **Inability to add Tour Packages**: Currently, users are unable to create and add custom tour packages to the platform, limiting the diversity of options available and restricting users' ability to tailor their travel experiences to their preferences.
2) **Visibility of assigned tour guides**: The current system lacks visibility for users to easily identify the tour guide assigned to their booking, which may hinder communication and rapport building between customers and their guides.
3) **Lack of hotel images**: Users are unable to view images of hotels before booking, potentially impacting their decision-making process and overall satisfaction with their accommodations.
4) **Inability to make in-app payments**: The absence of in-app payment functionality restricts users from completing transactions seamlessly within the app, potentially leading to inconvenience and a disjointed user experience.

5) **Limited access for tour guides**: Tour guides are unable to view a list of customers assigned to them, which may hinder their ability to prepare adequately for tours and provide personalized assistance during travel experiences.
6) **Limited city addition**: Users do not have the capability to contribute new cities and destinations to the platform, resulting in a restricted selection of destinations and potentially excluding users seeking less mainstream travel experiences.
7) **Admin functionality:** The administrative tools and features are limited, hindering administrators' ability to efficiently manage and optimize platform operations, including user management, content moderation, and analytics.

FUTURE WORK

1) **Tour package management:** Develop functionality for users to create and add custom tour packages to the platform, enhancing the diversity of options available and empowering users to tailor their travel experiences.
2) **City Management:** Enable users to contribute new cities and destinations to the platform, enriching the app's database with a broader selection of destinations and accommodating the preferences of users seeking unique travel experiences.
3) **Enhanced admin functionality**: Enhance administrative tools and features to provide administrators with more comprehensive management capabilities, including user management, content moderation, analytics dashboards, and reporting tools, enabling more efficient oversight and optimization of platform operations.
4) **Improved tour guide visibility:** Implement a feature that enables users to easily view the profile and contact information of the tour guide assigned to their booking. This enhancement fosters better communication and trust between customers and guides, enhancing the overall travel experience.
5) **Integration of hotel images**: Enhance the app by integrating high-quality images of hotels into the booking interface. Providing users with visual representations of their accommodations enables more informed decision-making and enhances satisfaction with their chosen lodging
6) **In-App Payment integration**: Develop and integrate secure payment functionality directly within the app, allowing users to complete transactions seamlessly without the need for external payment platforms. This feature streamlines the booking process, enhances user convenience, and promotes a more cohesive user experience.

7) **Enhanced access for tour guides:** Create a dedicated portal or dashboard within the app that enables tour guides to view details of the customers assigned to them, including booking information and preferences. This empowers guides to better prepare for tours, tailor experiences to individual preferences, and provide personalized assistance throughout the travel journey.

REFERENCES:

https://www.google.com/
https://codex.cs.yale.edu/avi/db-book/db6/slide-dir/index.html
https://stackoverflow.com/