a RDBMS PROJECT

PROJECT REPORT

Bachelor of Technology (B.Tech)
in
Computer and Communication Engineering

Relational Database Management System
CS1432 | IV SEMESTER (Core Course)

Faculty : **Mr. Satyabrata Roy**

DOMESTIC

AIRLINES MANAGEMENT SYSTEM

RDBMS PROJECT

SUBMISSION BY

Mayank Jhanwar
RegNo. : 189303179
Email : mayank.189303179@muj.manipal.edu
/MayankJhanwar

Namit Varshney
RegNo. : 189302111
Email : namit.189302111@muj.manipal.edu
/NamitVarshney

INSPIRED BY LIFE

MANIPAL UNIVERSITY
JAIPUR

Established under the Manipal University Jaipur Act (No. 21 of 2011)

JUNE **2020**

INDEX

DOMESTIC
AIRLINES MANAGEMENT SYSTEM
RDBMS PROJECT

MANIPAL UNIVERSITY
JAIPUR

Established under the Manipal University Jaipur Act (No. 21 of 2011)

CERTIFICATE

This is to certify that the project entitled
"**Domestic Airlines Management System**"

is a bonafide work carried out as part of the course
Relational Database and Management System

under my guidance by

Mayank Jhanwar
Reg. No: 189303179

Namit Varshney
Reg. No: 189302111

student of **B.Tech - IV Semester** at the
Department of Computer & Communication Engineering , Manipal University Jaipur,

during the academic **Semester IV** , in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer & Communication Engineering**, at
Manipal University Jaipur

Month of Submission
JUNE 2020

Signature of Instructor
Mr. Satyabrata Roy

DOMESTIC

AIRLINES MANAGEMENT SYSTEM

RDBMS PROJECT

In this DBMS project, we are focusing on ticket booking function, this function will take the user inputs like their requirements. It also accepts user identity info such as Name, age, phone-no. It can provide us the total details about the flights and passengers

SOFTWARE REQUIRENMENTS

Specifying the hardware and software requirements is imperative for any management system. It goes without saying that any compatibility issues that arise due to the hardware and software being incompatible can have serious consequences for the stability of the system, especially in a high-stakes environment like a healthcare location. Therefore, to ensure robustness of the system, it is necessary to follow the required hardware and software specifications.

- **Backend :**
  SQL Workbench
- **Frontend** :
  HTML - CSS - PHP
- **Server**
  WAMP Server

Objectives

This project aims to automate the Airlines management system. This project is developed mainly to administrate flight bookings. The purpose of this project, titled 'Domestic Airlines Management System' is to computerize the front-office management of the Airline institution using software that is fast, simple, user-friendly, and cost-effective. It deals with the collection of passengers' information, booking date, destination, etc. Traditionally, it was done manually. The main function of the system is to register and store passenger and flight details, and retrieve those details as and when required, and to manipulate these details meaningfully.

The Need of Proposed System

The first step in the development life cycle of a system is the identification of a need to change or improve an existing system. An initial investigation on the existing system was carried out. The earlier systems used for the management of such airport locations were completely manual. Many problems were identified during the initial study of this existing system. To develop this software, we studied the problems faced at airports. Based on the information collected, we have created a database management system to make the current process more efficient and convenient.

SUBMISSION BY

Mayank Jhanwar
RegNo. : 189303179
Email : mayank.189303179@muj.manipal.edu

Namit Varshney
RegNo. : 189302111
Email : namit.189302111@muj.manipal.edu

Faculty
Mr. Satyabrata Roy

- ## DATABASE DESIGN FOR DOMESTIC AIRLINES MANAGEMENT SYSTEM

• ENTITY LIST

1. Passengers
2. Contact_Info
3. State
4. Transactions
5. Flight_Schedules
6. Airfare
7. Aircrafts
8. Branches
9. Employees
10. Route

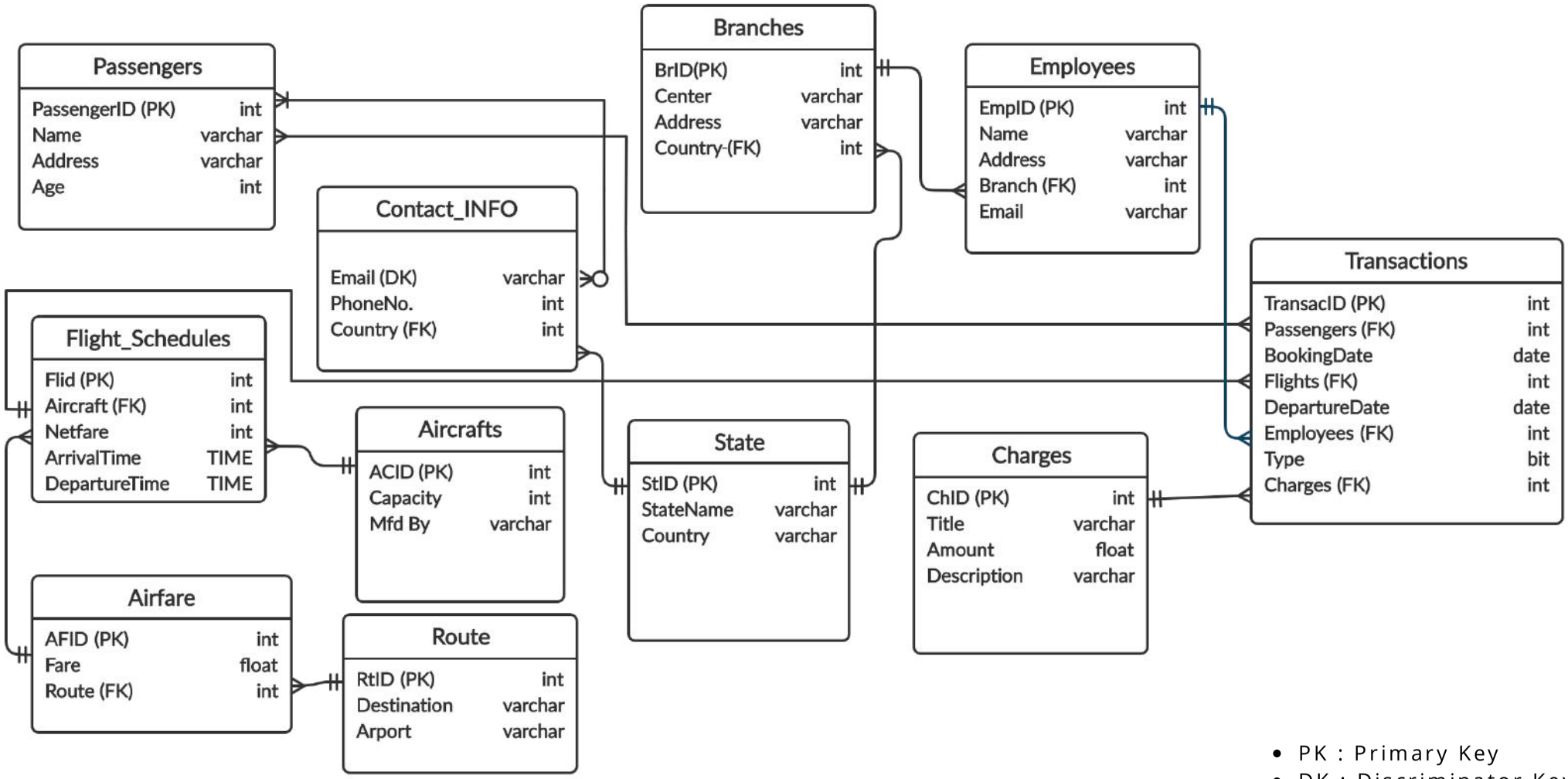
• ENTITIES WITH RELEVANT ATTRIBUTES

1. **Passengers**
   PassengerID, Name, Address, Age
2. **Contact_Info**
   Email, PhoneNo., State
3. **State**
   StID, StateName, Country
4. **Transactions**
   TransacID, Passengers, BookingGate, Flights, DepartureDate, Employees, Type, Charges
5. **Flight_Schedules**
   FLID, Aircraft, Netfare, ArrivalTime, DepartureTime
6. **Airfare**
   AFID, Fare, Route
7. **Aircrafts**
   ACID, Capacity, MfdBy
8. **Branches**
   BrID, Center, Address, State
9. **Employee**
   TransacID, Passengers, BookingDate, Flights, DepartureDate, Employees, Type, Charges
10. **Route**
    RtID, Destination, Airport

• ENTITY CLASSIFICATION WITH RESPECTIVE PRIMARY & FOREIGN KEY:

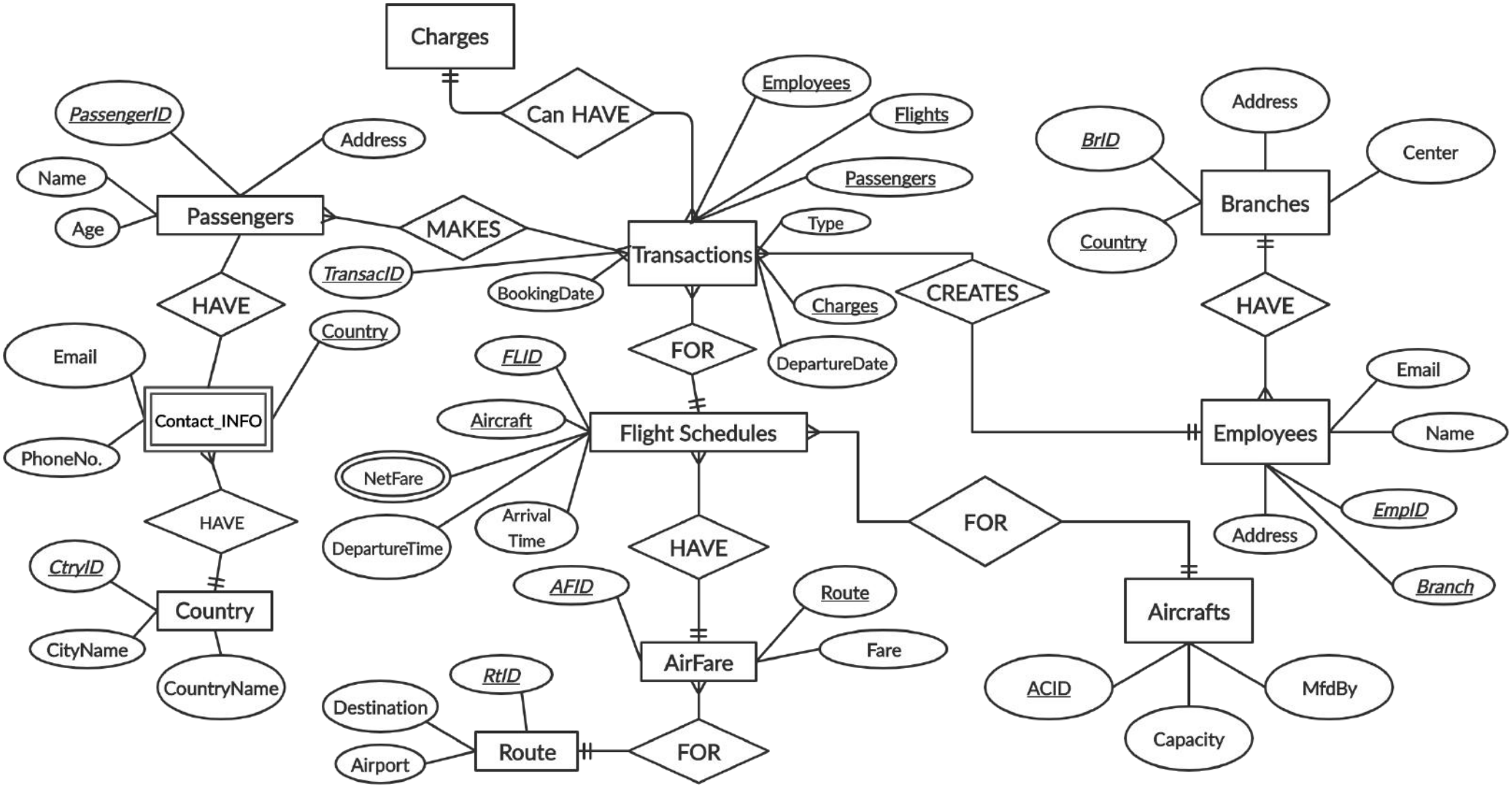
	PRIMARY KEY	FOREIGN KEY
Passengers	PassengerID	-----
Contact_INFO	-----	State
State	StID	----
Transactions	TransacID	Passengers, Flights, Employees, Charges
Flight_Schedules	Flid	Aircraft , Netfare
Airfare	AfID	Route
Aircrafts	AcID	-------
Branches	BranchID	State
Employee	EmpID	Branch
Route	RtID	-------

- **RELATIONAL MODEL**



- PK : Primary Key
- DK : Discriminator Key
- FK : Foreign Key

• ER (ENTITY RELATIONSHIP) DIAGRAM



• CARDINALITY

Representation

- One to Many Relationship
- Many to One Relationship
- One to One Relationship

• **AirCrafts & Flight_Schedule**	One to Many Relationship
• **Route & AirFare**	One to Many Relationship
• **AirFare & Flight_Schedule**	One to Many Relationship
• **Transactions & Charges**	Many to One Relationship
• **State & Branches**	One to Many Relationship
• **Contact_INFO & State**	Many to One Relationship
• **Passengers & Contact_INFO**	One to One Relationship
• **Passengers & Transactions**	Many to Many Relationship
• **Branches & Employee**	One to Many Relationship
• **Employees & Transactions**	One to Many Relationship
• **Transactions & Flight_Schedule**	Many to One Relationship

• WEAK ENTITY

Contact_INFO

- **Discriminator Key** : Email
- **Prmary Key** : PassengerID + Email
- **Strong Entity** : Passengers

• MULTIVALUE ATTRIBUTE

Netfare

1. **Flight Fare**
2. **Meals Fare**
3. **Luggage Fare**

• DATA DEFINITION LANGUAGE IMPLEMENTATIONS:

Creating Database

```
create database DAMS;
use DAMS;
```

Aircrafts

```
CREATE TABLE AirCrafts
(
   AcID INT Primary Key,
  Capacity INT NOT NULL,
  MfdBy Varchar(128) NOT NULL
);
```

Route

```
CREATE TABLE Route
(
  RtID INT,
  Airport  Varchar(32) NOT NULL,
  Destination  Varchar(32) NOT NULL,
  PRIMARY KEY (RtID)
);
```

Airfare

```
CREATE TABLE AirFare
(
  AfID INT,
  Route INT,
  Fare float,
  PRIMARY KEY (AfID),
  CONSTRAINT fk_Route FOREIGN KEY (Route) REFERENCES Route(RtID)
);
```

Charges

```
CREATE TABLE Charges
(
  ChID INT PRIMARY KEY,
  Title Varchar(32)  NOT NULL,
  Amount INT NOT NULL,
  Description  Varchar (400)
);
```

- **DATA DEFINITION LANGUAGE IMPLEMENTATIONS:**

Flight_Schedule

```
CREATE TABLE Flight_Schedule
(
  FlID INT,
  FlightDate DATETIME,
  DepartureTime DATETIME,
  ArrivalTime DATETIME,
  AirCraft INT,
  NetFare INT,
  PRIMARY KEY (FlID),
  CONSTRAINT fk_AirCraft FOREIGN KEY (AirCraft) REFERENCES AirCrafts(AcID),
  CONSTRAINT fk_NetFare FOREIGN KEY (NetFare) REFERENCES AirFare(AfID)
);
```

State

```
CREATE TABLE State
(
 StID INT,
 StateName Varchar(32),
 Country Varchar(32),
 PRIMARY KEY (StID)
);
```

Transactions

```
CREATE TABLE Transactions
 (
   TransacID INT PRIMARY KEY,
   BookingDate DATETIME,
   DepartureDate DATETIME,
   Passengers INT,
   Flights INT,
   Type BIT,
   Employees INT,
   Charges INT,

   CONSTRAINT fk_Passengers FOREIGN KEY (Passengers) REFERENCES Passengers(PassengerID),
   CONSTRAINT fk_Flights FOREIGN KEY (Flights) REFERENCES Flight_Schedule(FlID),
   CONSTRAINT fk_Employee FOREIGN KEY (Employees) REFERENCES Employee(EmpID),
   CONSTRAINT fk_Charges FOREIGN KEY (Charges) REFERENCES Charges(ChID)
);
```

- **DATA DEFINITION LANGUAGE IMPLEMENTATIONS:**

Employees

```
CREATE TABLE Employee
 (
   EmpID INT PRIMARY KEY,
   Name Varchar (32) NOT NULL,
   Address Varchar(32) NOT NULL,
   Branch INT NOT NULL,
   Email Varchar(16) NOT NULL,

   CONSTRAINT fk_Branch FOREIGN KEY (Branch) REFERENCES Branches(BrID)
);
```

Branches

```
CREATE TABLE Branches
(
  BrID INT PRIMARY KEY,
  Center Varchar(16) NOT NULL,
  Address Varchar(32) NOT NULL,
  State INT,

  CONSTRAINT fk_StateOfEmployee FOREIGN KEY (State) REFERENCES State(StID)
);
```

Contact_Details

```
CREATE TABLE Contact_Details
(
  Email Varchar(16) NOT NULL,
  PhoneNo DOUBLE NOT NULL,
  State INT NOT NULL,

  CONSTRAINT fk_State FOREIGN KEY (State) REFERENCES State(StID)
);
```

Passengers

```
CREATE TABLE Passengers
 (
   PassengerID INT PRIMARY KEY,
   Name Varchar(32) NOT NULL,
   Address Varchar(64) NOT NULL,
   Age INT NOT NULL

);
```

- **DATA DEFINITION LANGUAGE IMPLEMENTATIONS:**

Employees

```
CREATE TABLE Employee
 (
   EmpID INT PRIMARY KEY,
   Name Varchar (32) NOT NULL,
   Address Varchar(32) NOT NULL,
   Branch INT NOT NULL,
   Email Varchar(16) NOT NULL,

   CONSTRAINT fk_Branch FOREIGN KEY (Branch) REFERENCES Branches(BrID)
);
```

Branches

```
CREATE TABLE Branches
(
  BrID INT PRIMARY KEY,
  Center Varchar(16) NOT NULL,
  Address Varchar(32) NOT NULL,
  State INT,

  CONSTRAINT fk_StateOfEmployee FOREIGN KEY (State) REFERENCES State(StID)
);
```

Contact_Details

```
CREATE TABLE Contact_Details
(
  Email Varchar(16) NOT NULL,
  PhoneNo DOUBLE NOT NULL,
  State INT NOT NULL,

  CONSTRAINT fk_State FOREIGN KEY (State) REFERENCES State(StID)
);
```

Passengers

```
CREATE TABLE Passengers
 (
   PassengerID INT PRIMARY KEY,
   Name Varchar(32) NOT NULL,
   Address Varchar(64) NOT NULL,
   Age INT NOT NULL

);
```

- **SCHEMAS**

Aircrafts

Result Grid | Filter Rows: | Export: | Wr

Field	Type	Null	Key	Default	Extra
AcID	int(11)	NO	PRI	NULL	
Capacity	int(11)	NO		NULL	
MfdBy	varchar(128)	NO		NULL	

Route

Result Grid | Filter Rows: | Export: | Wrap Cell C

Field	Type	Null	Key	Default	Extra
RtID	int(11)	NO	PRI	0	
Airport	varchar(32)	NO		NULL	
Destination	varchar(32)	NO		NULL	

Airfare

Result Grid | Filter Rows: | Export:

Field	Type	Null	Key	Default	Extra
AfID	int(11)	NO	PRI	0	
Route	int(11)	YES	MUL	NULL	
Fare	float	YES		NULL	

Charges

Result Grid | Filter Rows: | Export: | Wrap Ce

Field	Type	Null	Key	Default	Extra
ChID	int(11)	NO	PRI	NULL	
Title	varchar(32)	NO		NULL	
Amount	int(11)	NO		NULL	
Description	varchar(400)	YES		NULL	

Employees

Result Grid | Filter Rows: | Export: | W

Field	Type	Null	Key	Default	Extra
EmpID	int(11)	NO	PRI	NULL	
Name	varchar(32)	NO		NULL	
Address	varchar(32)	NO		NULL	
Branch	int(11)	NO	MUL	NULL	
Email	varchar(100)	YES		NULL	

Branches

Result Grid | Filter Rows: | Export: | V

Field	Type	Null	Key	Default	Extra
BrID	int(11)	NO	PRI	NULL	
Center	varchar(16)	NO		NULL	
Address	varchar(32)	NO		NULL	
State	int(11)	YES	MUL	NULL	

Contact_Details

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
Email	varchar(100)	YES		NULL	
PhoneNo	double	NO		NULL	
State	int(11)	NO	MUL	NULL	

- **SCHEMAS**

PASSENGERS

Result Grid | Filter Rows: | Export: | Wrap Cell C

Field	Type	Null	Key	Default	Extra
PassengerID	int(11)	NO	PRI	NULL	
Name	varchar(32)	NO		NULL	
Address	varchar(64)	NO		NULL	
Age	int(11)	NO		NULL	

FLIGHT_SCHEDULE

Result Grid | Filter Rows: | Export: | Wr

Field	Type	Null	Key	Default	Extra
FlID	int(11)	NO	PRI	0	
FlightDate	datetime	YES		NULL	
DepartureTime	datetime	YES		NULL	
ArrivalTime	datetime	YES		NULL	
AirCraft	int(11)	YES	MUL	NULL	
NetFare	int(11)	YES	MUL	NULL	

STATE

Result Grid | Filter Rows: | Export: | Wra

Field	Type	Null	Key	Default	Extra
StID	int(11)	NO	PRI	0	
StateName	varchar(32)	YES		NULL	
Country	varchar(32)	YES		NULL	

TRANSACTIONS

Result Grid | Filter Rows: | Export: | Wrap C

Field	Type	Null	Key	Default	Extra
TransacID	int(11)	NO	PRI	NULL	
BookingDate	datetime	YES		NULL	
DepartureDate	datetime	YES		NULL	
Passengers	int(11)	YES	MUL	NULL	
Flights	int(11)	YES	MUL	NULL	
Type	bit(1)	YES		NULL	
Employees	int(11)	YES	MUL	NULL	
Charges	int(11)	YES	MUL	NULL	

- **DATA MANIPULATION LANGUAGE IMPLEMENTATIONS INSERTION OF VALUES**

Inserting Values into AIRCRAFTS

```
INSERT into AirCrafts
 ( AcID, Capacity, MfdBy)
 VALUES
 (001, 75, "Alenia Aeronotica"),
 (002, 100, "Airbus Corporate"),
 (003, 500, "Boeing Business"),
 (004, 100, "Alenia Aeronotica"),
 (005, 75, "Boeing Business"),
 (006, 85, "Airbus Corporate"),
 (007, 95, "Airbus Corporate"),
 (008, 100, "Gulfstream Aerospace"),
 (009, 100, "Gulfstream Aerospace"),
 (010, 500, "Gulfstream Aerospace"),
 (011, 85, "Gulfstream Aerospace"),
 (012, 500, "Alenia Aeronotica"),
 (013, 95, "Alenia Aeronotica"),
 (014, 100, "Airbus Corporate"),
 (015, 200, "Alenia Aeronotica"),
 (016, 250, "Alenia Aeronotica"),
 (017, 250, "Alenia Aeronotica"),
 (018, 300, "Textron Aviation"),
 (019, 300, "Textron Aviation"),
 (020, 300, "Textron Aviation");
);
```

OUTPUT

Result Grid | Filter Rows:

AcID	Capacity	MfdBy
1	75	Alenia Aeronotica
2	100	Airbus Corporate
3	500	Boeing Business
4	100	Alenia Aeronotica
5	75	Boeing Business
6	85	Airbus Corporate
7	95	Airbus Corporate
8	100	Gulfstream Aerospace
9	100	Gulfstream Aerospace
10	500	Gulfstream Aerospace
11	85	Gulfstream Aerospace
12	500	Alenia Aeronotica
13	95	Alenia Aeronotica
14	100	Airbus Corporate
15	200	Alenia Aeronotica
16	250	Alenia Aeronotica
17	250	Alenia Aeronotica
18	300	Textron Aviation
19	300	Textron Aviation
20	300	Textron Aviation
NULL	NULL	NULL

- **DATA MANIPULATION LANGUAGE IMPLEMENTATIONS INSERTION OF VALUES**

Inserting Values into ROUTE

INSERT INTO **Route**
Values
(1, "Kathmandu", "Pokhara"),
(2, "Pokhara" , "Kathmandu"),
(3, "Delhi" , "Mumbai"),
(4, "Mumbai" , "Delhi"),
(5, "Delhi" , "Surat"),
(6, "Surat" , "Delhi"),
(7, "Jaipur" , "Surat"),
(8, "Surat", "Jaipur"),
(9, "Jaipur" , "Indore"),
(10, "Indore", "Jaipur"),
(11, "Mumbai", "Chennai"),
(12, "Mumbai", "Hyderabad"),
(13, "Hyderabad", "Mumbai"),
(14, "Chennai", "Mumbai"),
(15, "Chennai", "Hyderabad"),
(16, "Hyderabad", "Chennai"),
(17, "Delhi" , "Chennai"),
(18, "Bangalore" , "Delhi"),
(19, "Delhi", "Bangalore"),
(20, "Delhi", "Jammu");

OUTPUT

Result Grid Filter Rows:

RtID	Airport	Destination
1	Kathmandu	Pokhara
2	Pokhara	Kathmandu
3	Delhi	Mumbai
4	Mumbai	Delhi
5	Delhi	Surat
6	Surat	Delhi
7	Jaipur	Surat
8	Surat	Jaipur
9	Jaipur	Indore
10	Indore	Jaipur
11	Mumbai	Chennai
12	Mumbai	Hyderabad
13	Hyderabad	Mumbai
14	Chennai	Mumbai
15	Chennai	Hyderabad
16	Hyderabad	Chennai
17	Delhi	Chennai
18	Bangalore	Delhi
19	Delhi	Bangalore
20	Delhi	Jammu
NULL	NULL	NULL

- **DATA MANIPULATION LANGUAGE IMPLEMENTATIONS INSERTION OF VALUES**

Inserting Values into AIRFARE

```
INSERT INTO AirFare
(AflD, Route, Fare)
VALUES
(1001, 1, 1250.00),
(1002, 2, 1250.00),
(1003, 3, 2000.00),
(1004, 4, 2000.00),
(1005, 5, 1500.00),
(1006, 6, 1500.00),
(1007, 7, 1000.00),
(1008, 8, 1000.00),
(1009, 9, 1650.00),
(1010, 10, 1650.00),
(1011, 11, 2100.00),
(1012, 12, 2400.00),
(1013, 13, 2400.00),
(1014, 14, 2100.00),
(1015, 15, 3600.00),
(1016, 16, 3600.00),
(1017, 17, 4000.00),
(1018, 18, 5000.00),
(1019, 19, 5000.00),
(1020, 20, 3600.00);
```

OUTPUT

Result Grid | Filter Rows:

AflD	Route	Fare
1001	1	1250
1002	2	1250
1003	3	2000
1004	4	2000
1005	5	1500
1006	6	1500
1007	7	1000
1008	8	1000
1009	9	1650
1010	10	1650
1011	11	2100
1012	12	2400
1013	13	2400
1014	14	2100
1015	15	3600
1016	16	3600
1017	17	4000
1018	18	5000
1019	19	5000
1020	20	3600
NULL	NULL	NULL

- **DATA MANIPULATION LANGUAGE IMPLEMENTATIONS INSERTION OF VALUES**

Inserting Values into CHARGES

INSERT INTO **Charges**
(ChID, Title, Amount, Description)
VALUES
(1,'Urgent Cancellation', 33.33, '33.3% will be charged for cancellation for booking within 11 hrs from flight time'),
(2,'Cancellation', 44.33, '44.3% will be charged for cancellation for booking within 11 hrs from flight time');

OUTPUT

Result Grid | Filter Rows: | Edit: | Export/Import:

ChID	Title	Amount	Description
1	Urgent Cancellation	33	33.3% will be charged for cancellation for booki...
2	Cancellation	44	44.3% will be charged for cancellation for booki...
NULL	NULL	NULL	NULL

Inserting Values into STATE

INSERT INTO **State**
(StID, StateName, Country)
VALUES
(1, 'Rajasthan', 'India'),
(2, 'Maharashtra', 'India'),
(3, 'Andhra Pradesh', 'India'),
(4, 'Karnataka', 'India'),
(5, 'Kashmir', 'India'),
(6, 'Kerala', 'India'),
(7, 'Madhya Pradesh', 'India'),
(8, 'Tamil Nadu', 'India'),
(9, 'Sikkim', 'India'),
(10, 'Gujarat', 'India'),
(11, 'Delhi', 'India');

OUTPUT

Result Grid | Filter Rows:

StID	StateName	Country
1	Rajasthan	India
2	Maharashtra	India
3	Andhra Pradesh	India
4	Karnataka	India
5	Kashmir	India
6	Kerala	India
7	Madhya Pradesh	India
8	Tamil Nadu	India
9	Sikkim	India
10	Gujarat	India
11	Delhi	India
NULL	NULL	NULL

- **DATA MANIPULATION LANGUAGE IMPLEMENTATIONS**
  INSERTION OF VALUES

Inserting Values into FLIGHT_Schedule

INSERT INTO **Flight_Schedule**
(FlID, FlightDate, DepartureTime, ArrivalTime, AirCraft, Netfare)
VALUES
(1, '2020-01-23', '2020-01-23 23:20:00', '2020-01-24 1:20:00', 1, 1001),
(2, '2020-02-28', '2020-01-28 23:25:00', '2020-01-29 2:30:00', 3, 1003);

OUTPUT

Result Grid | Filter Rows: | Edit: | Export/Import:

FlID	FlightDate	DepartureTime	ArrivalTime	AirCraft	NetFare
1	2020-01-23 00:00:00	2020-01-23 23:20:00	2020-01-24 01:20:00	1	1001
2	2020-02-28 00:00:00	2020-01-28 23:25:00	2020-01-29 02:30:00	3	1003
NULL	NULL	NULL	NULL	NULL	NULL

Inserting Values into PASSENGERS

insert into **Passengers**
(PassengerID,Name,Address,Age)
values
(101,'Mayank Maheshwari','Manipal University Jaipur,Dehmi Kalan,Rajasthan',20),
(102,'Ankit Grover','Valentina,Hiranandani Estate,Thane,Maharashtra',19),
(103,'Rajesh Sharma','Kota, Rajasthan',56),
(104,'Brajesh Maheshwari','Kota, Rajasthan',49),
(105,'Govind Maheshwari','Kota, Rajasthan',34),
(106,'Brijwesh Bannerjee','Thiruvananthanpuram,Kerala',24),
(107,'Sridhar Maheshwari','Telangana,Andhra Pradesh',19),
(108,'Saikat Dutta','Nagpur,Maharashtra',21),
(109,'Shobhit Joshi','Srinagar,Kashmir',24),
(110,'Rishabh Gajral','Gangtok,Sikkim',23);

OUTPUT

Result Grid | Filter Rows: | Edit: | Export/Import:

PassengerID	Name	Address	Age
101	Mayank Maheshwari	Manipal University Jaipur,Dehmi Kalan,Rajasthan	20
102	Ankit Grover	Valentina,Hiranandani Estate,Thane,Maharashtra	19
103	Rajesh Sharma	Kota, Rajasthan	56
104	Brajesh Maheshwari	Kota, Rajasthan	49
105	Govind Maheshwari	Kota, Rajasthan	34
106	Brijwesh Bannerjee	Thiruvananthanpuram,Kerala	24
107	Sridhar Maheshwari	Telangana,Andhra Pradesh	19
108	Saikat Dutta	Nagpur,Maharashtra	21
109	Shobhit Joshi	Srinagar,Kashmir	24
110	Rishabh Gajral	Gangtok,Sikkim	23
NULL	NULL	NULL	NULL

- **DATA MANIPULATION LANGUAGE IMPLEMENTATIONS INSERTION OF VALUES**

Inserting Values into TRANSACTIONS

insert into **Transactions**
(TransacID,BookingDate,DepartureDate,Passengers,Flights,Type,Employees,Charges)
values
(111,'2020-06-15','2020-01-15 23:59:59.99',101,1,1,1,NULL),
(112,'2020-06-14','2020-01-14 22:59:59.99',102,2,1,2,NULL),
(113,'2020-06-13','2020-01-13 21:59:59.99',103,2,1,3,NULL),
(114,'2020-06-12','2020-01-12 20:59:59.99',104,1,1,4,NULL),
(115,'2020-06-11','2020-01-11 19:59:59.99',105,1,1,5,NULL);

OUTPUT

Result Grid | Filter Rows: | Export: | Wrap C

Field	Type	Null	Key	Default	Extra
TransacID	int(11)	NO	PRI	NULL	
BookingDate	datetime	YES		NULL	
DepartureDate	datetime	YES		NULL	
Passengers	int(11)	YES	MUL	NULL	
Flights	int(11)	YES	MUL	NULL	
Type	bit(1)	YES		NULL	
Employees	int(11)	YES	MUL	NULL	
Charges	int(11)	YES	MUL	NULL	

- **DATA MANIPULATION LANGUAGE IMPLEMENTATIONS INSERTION OF VALUES**

Inserting Values into BRANCHES

INSERT INTO **Branches**
(BrID, Center, Address, State)
VALUES
(00101, 'Mumbai', 'Borivalli', 2),
(00102, 'Jaipur', 'Mansarovar', 1),
(00103, 'Indore', 'Sanchi', '7'),
(00104, 'Hyderabad', 'Maheshwaram', '3'),
(00105, 'Chennai', 'Kapaswaram', '8'),
(00106, 'Jammu', 'Jammu Cantonment', '5'),
(00107, 'Bangalore', 'Devanahalli', '4'),
(00108, 'Surat', 'Dumas', '10'),
(00109, 'Delhi', 'Noida', '11');

OUTPUT

Result Grid | Filter Rows: | Edit:

BrID	Center	Address	State
101	Mumbai	Borivalli	2
102	Jaipur	Mansarovar	1
103	Indore	Sanchi	7
104	Hyderabad	Maheshwaram	3
105	Chennai	Kapaswaram	8
106	Jammu	Jammu Cantonment	5
107	Bangalore	Devanahalli	4
108	Surat	Dumas	10
109	Delhi	Noida	11
NULL	NULL	NULL	NULL

Inserting Values into EMPLOYEE

INSERT INTO **Employee**
(EmpID, Name, Address, Branch, Email)
VALUES
(1, 'Diwan Adhikari', 'Bagbazaar - 11, KTM', 00101 ,'the.one@yahoo.com'),
(2, 'Mayank Jindal', 'Mansarovar-East', 00102 , 'jindal@gmail.com'),
(3, 'Namit Varshney', 'Sanchi-West', 00103 , 'namit@outlook.com'),
(4, 'Ankit Grover', 'Kapaswaram-East', 00105, 'grover@outlook.com'),
(5, 'Aaarushi Goyal', 'Maheshwaram-North', 00104, 'goyal@outlook.com'),
(6, 'Sushant', 'Jammu-WEST', 00106, 'sushant@gmail.com'),
(7, 'Riya Kumar', 'Bangalore-east', 00107, 'rkumar@gmail.com'),
(8, 'Sushmita', 'Parvat Patia', 00108, 'sushmita@outlook.com'),
(9, 'Devvrat Singh', 'Krishna Nagar', 00109, 'Dsingh@gmail.com');

- **DATA MANIPULATION LANGUAGE IMPLEMENTATIONS INSERTION OF VALUES**

OUTPUT (EMPLOYEE)

Result Grid | Filter Rows: | Edit: | Export/Import:

EmpID	Name	Address	Branch	Email
1	Diwan Adhikari	Bagbazaar - 11, KTM	101	the.one@yahoo.com
2	Mayank Jindal	Mansarovar-East	102	jindal@gmail.com
3	Nam[illegible]	[illegible]nchi-West	103	namit@outlook.com
4	Ankit Grover	Kapaswaram-East	105	grover@outlook.com
5	Aaarushi Goyal	Maheshwaram-North	104	goyal@outlook.com
6	Sushant	Jammu-WEST	106	sushant@gmail.com
7	Riya Kumar	Bangalore-east	107	rkumar@gmail.com
8	Sushmita	Parvat Patia	108	sushmita@outlook.com
9	Devvrat Singh	Krishna Nagar	109	Dsingh@gmail.com
NULL	NULL	NULL	NULL	NULL

Inserting Values into CONTACT_DETAILS

insert into **Contact_Details**
(Email,PhoneNo,State)
values
('may@gmail.com',9892567625,1),
('agr@gmail.com',9892567626,2),
('raj@gmail.com',9892567727,1),
('braj@gmail.com',9892568626,1),
('govind@gmail.com',9892567645,1),
('bjee@gmail.com',9892565641,1),
('sridh@gmail.com',9894667645,3),
('saik@gmail.com',9892577645,2),
('joshi@gmail.com',9893567645,5),
('gaj@gmail.com',9792567745,9);

OUTPUT (CONTACT_DETAILS)

Result Grid | Filter Rows:

Email	PhoneNo	State
may@gmail.com	9892567625	1
agr@gmail.com	9892567626	2
raj@gmail.com	9892567727	1
braj@gmail.com	9892568626	1
govind@gmail.com	9892567645	1
bjee@gmail.com	9892565641	1
sridh@gmail.com	9894667645	3
saik@gmail.com	9892577645	2
joshi@gmail.com	9893567645	5
gaj@gmail.com	9792567745	9

Making Project Functional

Frontend - Backend Linking

This section discusses in detail, about how we made our Project Functional with Graphical User Interface :

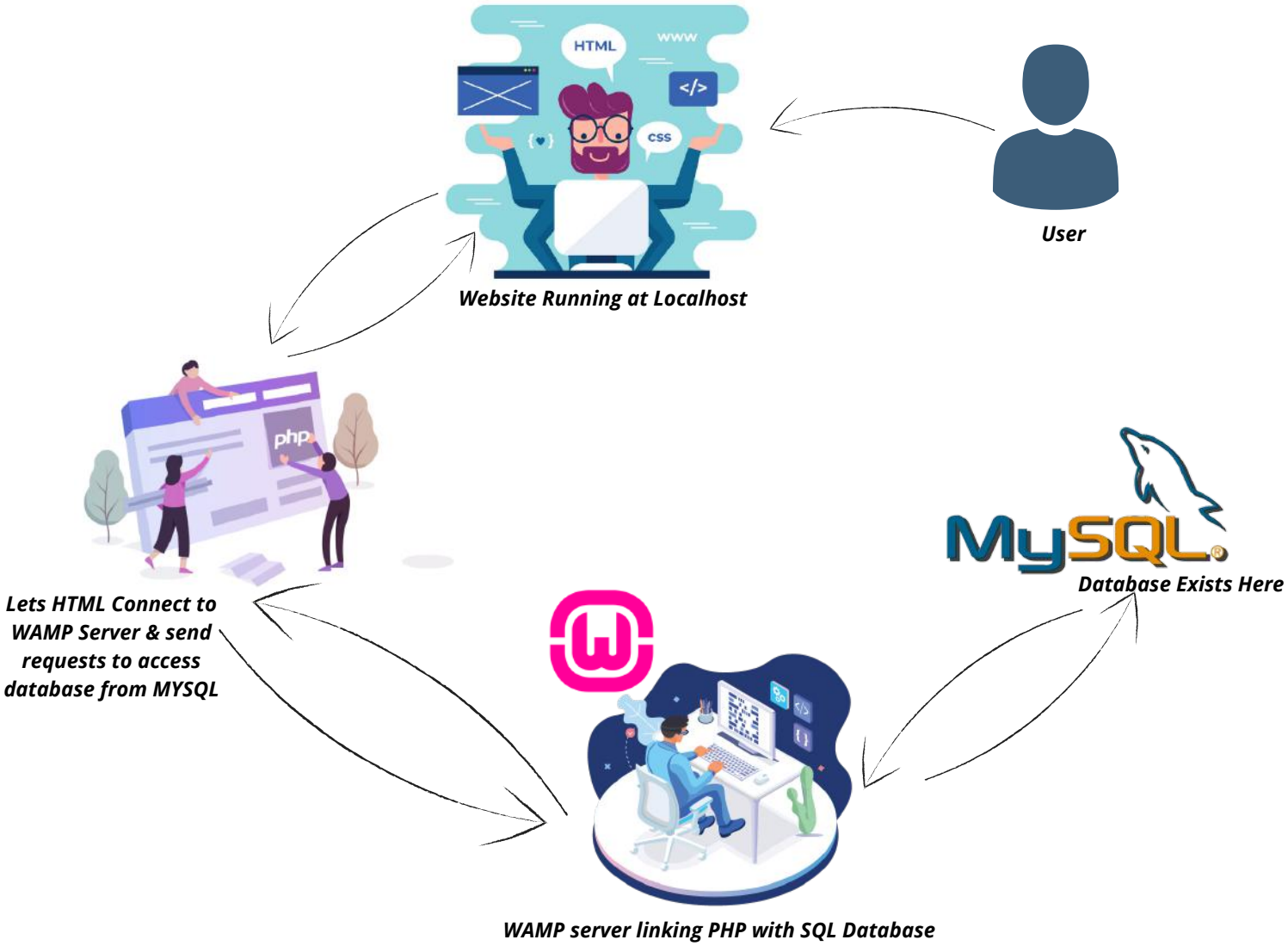
1. **Front-end:**
   The project's GUI has been made through web pages written in HTML and design elegantly with CSS for it to be attractive

2. **Back-end:**
   PHP *is being used along with HTML and CSS as a medium to make a connection with the database.*

3. **Database:**
   All the database used in this project is being created on the ***MYSQL Workbench*** *by writing SQL commands.*

4. **Server***:*
   PHP, SQL Workbench Database are all connected with HTML-CSS GUI by establishing a server on the localhost by using ***WAMP Server****, which helps us in running the project successfully.*

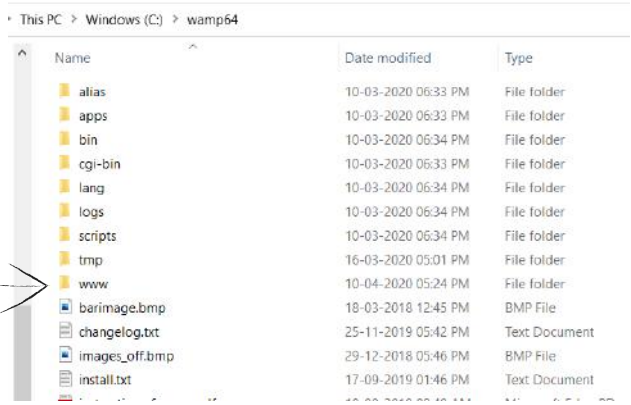
The linkage can be better understood by referring the diagram shown below:

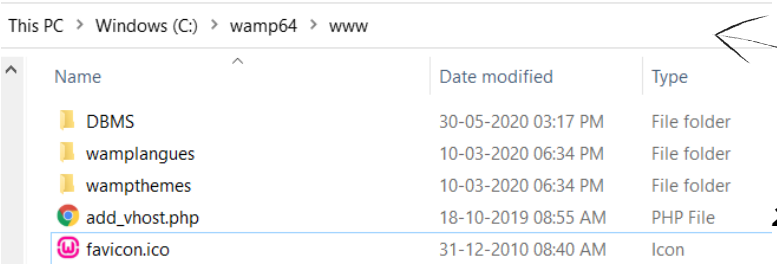


Running WAMP server

1. ***Adding Project Files in the working directory of the WAMP Server -***

www *is the working diectory of the wamp server. All the project files need to be stored here*

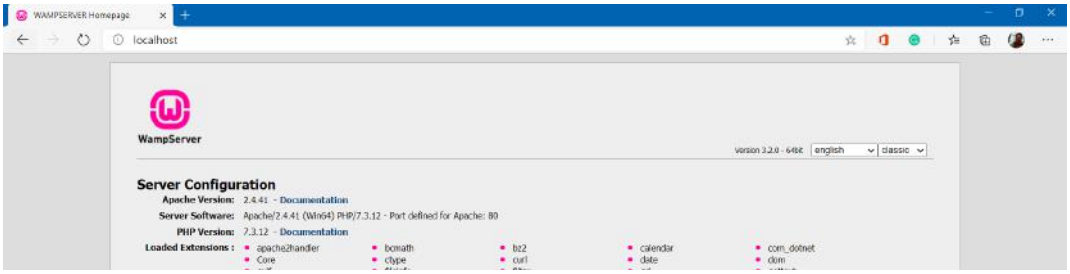




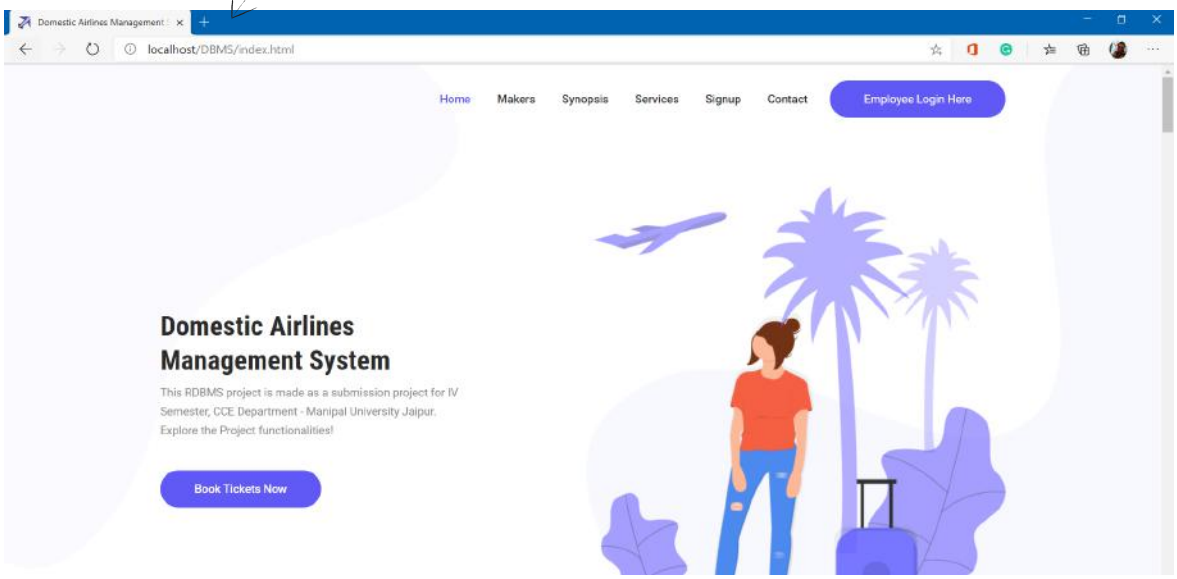
2. Storing DBMS Website folder in www directory.

3. running localhost server

We can see the localhost now running WAMP Server



4. opening project files on WAMP SERVER by typing localhost/DBMS/index.html



Project Functionalities

The project SQL Queries are written according to the range of functionalities, we are providing to our users (Both Employees and Customer) with a constraint on them following their usage rights.

The functionalities, which we are providing range from **simple displaying of records** **to** **booking tickets** **to making a** **constraint on users by making a login form.** (*As a result, it divides the services accordingly different for both customers and Employees)***.**

Here in this section, we will represent all the queries which are used in making the GUI successfully working, with the help of SQL Workbench. (*Later in this report, we will explain the GUI Functionality with the queries written and explained here)*

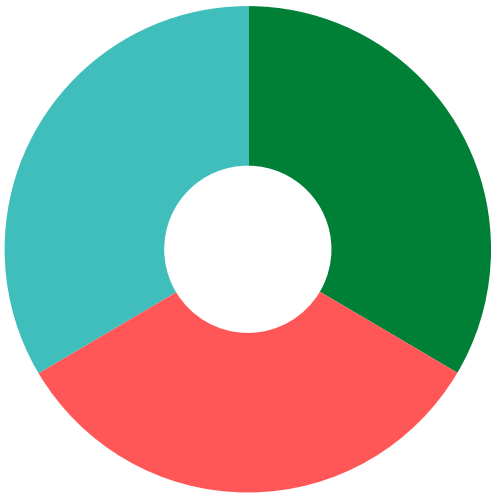
List of Project Functionalities & their segmentation

We are thereby, *segmenting* the usage of services on the basis of **users are who are required to login** *(Customers & Employees)* and **the services which can be viewed without login.**

With Login (Employee)

Perform CRUD queries

1. Add New Data
2. Delete Existing Data
3. See Employees Record



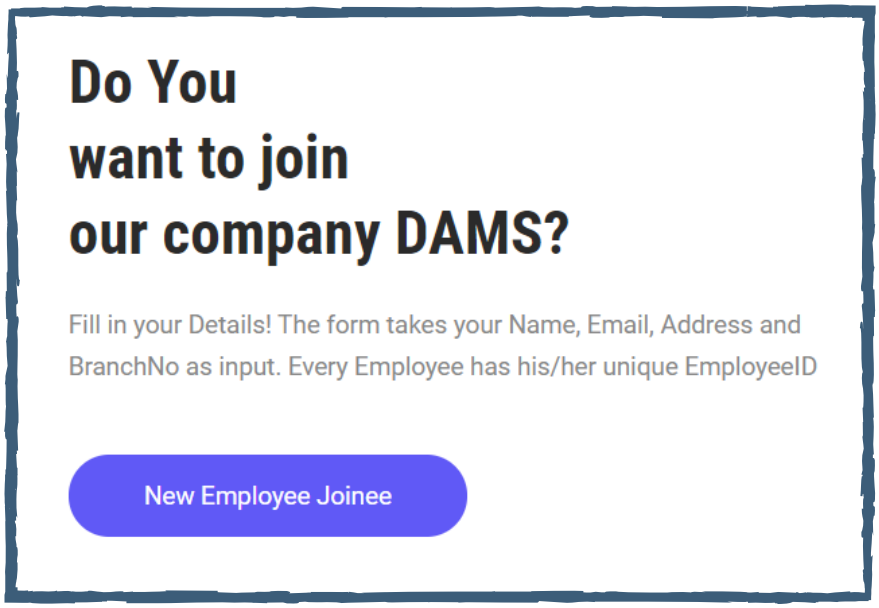
With Login (Customer)

1. Booking Flight Tickets
2. Search Flights

Without Login

1. **Access Services Section** : See database of Airfares , Aircraft, States we serve, Running flight schedules, Currently running routes, & Employees record
2. **New Customer Sign Up**
3. **New Employee Joinee**

Without LOGIN Section Queries:



Snapshot from GUI representation

New Employee Sign UP

The user is asked to enter the following Information :

1. *EmployeeID*
2. *Full Name*
3. *Address*
4. *Branch Code (as displayed)*
5. *Email ID*

Inserting Values into Employee

INSERT INTO **Employee**
(EmpID, Name, Address, Branch, Email)
VALUES
(10, 'Srijan Gautam', 'D- Block Geeta Colony', 00109 ,'srijan@yahoo.com'),

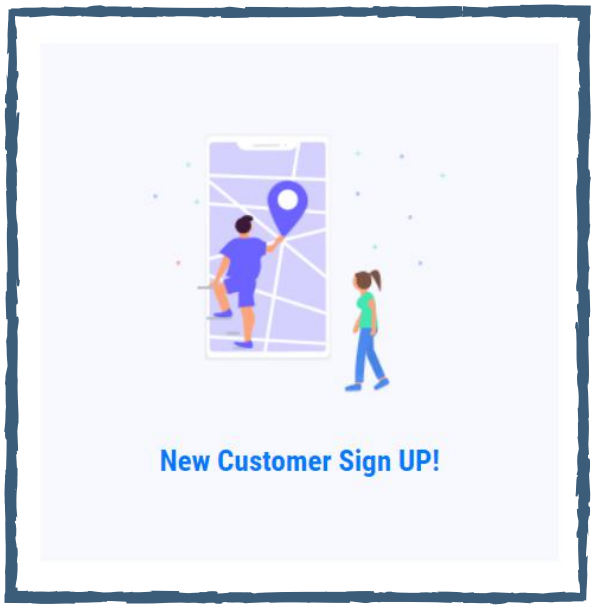
New Employee Added into the Database

EmpID	Name	Address	Branch	Email
1	Diwan Adhikari	Bagbazaar - 11, KTM	101	the.one@yahoo.com
2	Mayank Jindal	Mansarovar-East	102	jindal@gmail.com
3	Namit Varshney	Sanchi-West	103	namit@outlook.com
4	Ankit Grover	Kapaswaram-East	105	grover@outlook.com
5	Aaarushi Goyal	Maheshwaram-North	104	goyal@outlook.com
6	Sushant	Jammu-WEST	106	sushant@gmail.com
7	Riya Kumar	Bangalore-east	107	rkumar@gmail.com
8	Sushmita	Parvat Patia	108	sushmita@outlook.com
9	Devvrat Singh	Krishna Nagar	109	Dsingh@gmail.com
10	Srijan Gautam	D- Block Geeta Colony	109	srijan@yahoo.com
NULL	NULL	NULL	NULL	NULL

Pic : Snapshot from SQL Workbench

Added Successfully

Without LOGIN Section Queries:



Snapshot from GUI representation

New Customer Sign UP

The user is asked to enter the following Information :

1. *PassengerID*
2. *Full Name*
3. *Address*
4. *Age*

Every new Customer needs to signup as having a PassengerID is compulsary for booking flight ticktes.

Inserting Values into Passengers

Insert into **Passengers**
(PassengerID,Name,Address,Age)
values
(110,'Viraj Pandit','A-603 Galaxy Heights Bandra Mumbai',24);

New Customer Added into the Database

Result Grid | Filter Rows: | Edit: | Export/Import:

PassengerID	Name	Address	Age
101	Mayank Maheshwari	Manipal University Jaipur,Dehmi Kalan,Rajasthan	20
102	Ankit Grover	Valentina,Hiranandani Estate,Thane,Maharashtra	19
103	Rajesh Sharma	Kota, Rajasthan	56
104	Brajesh Maheshwari	Kota, Rajasthan	49
105	Govind Maheshwari	Kota, Rajasthan	34
106	Brijwesh Bannerjee	Thiruvananthanpuram,Kerala	24
107	Sridhar Maheshwari	Telangana,Andhra Pradesh	19
108	Saikat Dutta	Nagpur,Maharashtra	21
109	Shobhit Joshi	Srinagar,Kashmir	24
110	Rishabh Gajral	Gangtok,Sikkim	23
111	Viraj Pandit	A-603 Galaxy Heights Bandra Mumbai	24
NULL	NULL	NULL	NULL

Pic : Snapshot from SQL Workbench

Added Successfully

Without LOGIN Section Queries:

Snapshot from GUI representation

displaying Aircrafts Record

Aircrafts database contain the following Information :

1. ***AircraftID*** *(unique for every aircraft)*
2. ***Capacity*** *(no. of passengers an aircraft can hold)*
3. ***Manufactured By*** *(name of the manufacturing company of the Aircraft)*

Displaying Values from Aircrafts table

select EmpID, Name, Address, Branch, Email from **Employee**;

displaying records

AcID	Capacity	MfdBy
1	75	Alenia Aeronotica
2	100	Airbus Corporate
3	500	Boeing Business
4	100	Alenia Aeronotica
5	75	Boeing Business
6	85	Airbus Corporate
7	95	Airbus Corporate
8	100	Gulfstream Aerospace
9	100	Gulfstream Aerospace
10	500	Gulfstream Aerospace
11	85	Gulfstream Aerospace
12	500	Alenia Aeronotica
13	95	Alenia Aeronotica
14	100	Airbus Corporate
15	200	Alenia Aeronotica
16	250	Alenia Aeronotica
17	250	Alenia Aeronotica
18	300	Textron Aviation
19	300	Textron Aviation
20	300	Textron Aviation
NULL	NULL	NULL

displayed records Successfully

Pic : Snapshot from SQL Workbench

Without LOGIN Section Queries:

Snapshot from GUI representation

displaying States serving Record

States database contain the following Information :

1. ***StID*** *(unique for every state)*
2. ***StateName*** *(name of the state)*
3. ***Country*** *(country name - by default INDIA)*

Displaying Values from State table

select StID, StateName, Country from **State**;

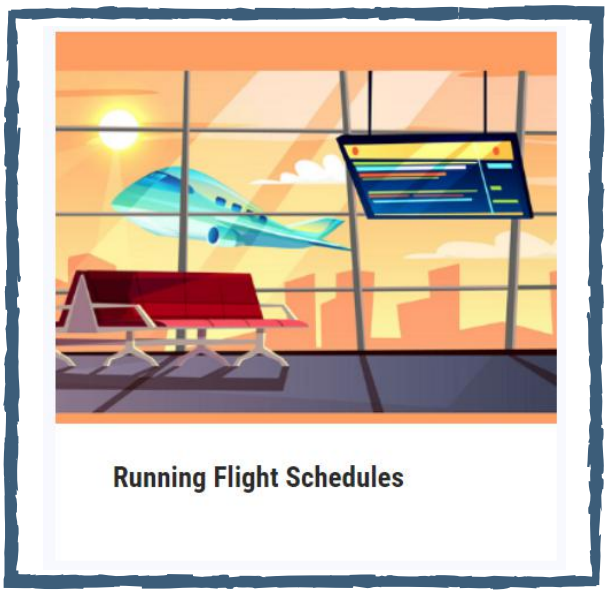
displaying records

StID	StateName	Country
1	Rajasthan	India
2	Maharashtra	India
3	Andhra Pradesh	India
4	Karnataka	India
5	Kashmir	India
6	Kerala	India
7	Madhya Pradesh	India
8	Tamil Nadu	India
9	Sikkim	India
10	Gujarat	India
11	Delhi	India
NULL	NULL	NULL

displayed records Successfully

Pic : Snapshot from SQL Workbench

Without LOGIN Section Queries:



Snapshot from GUI representation

displaying flight schedules Record

flight schedules database contain the following Information :

1. ***FlID*** *(unique for every flight schedule)*
2. ***FlightDate*** *(Date of travelling)*
3. ***DepartureTime*** *(Departuring time from the Origin)*
4. ***ArrivalTime*** *(Arriving time at the destination)*
5. ***AirCraft*** *(AircraftID of the aircraft for the schedule)*
6. ***Netfare*** *(NetfareID for the current route)*

Displaying Values from Flight_Schedule table

select FlID, FlightDate, DepartureTime, ArrivalTime, AirCraft, Netfare from **Flight_schedule**;

displaying records

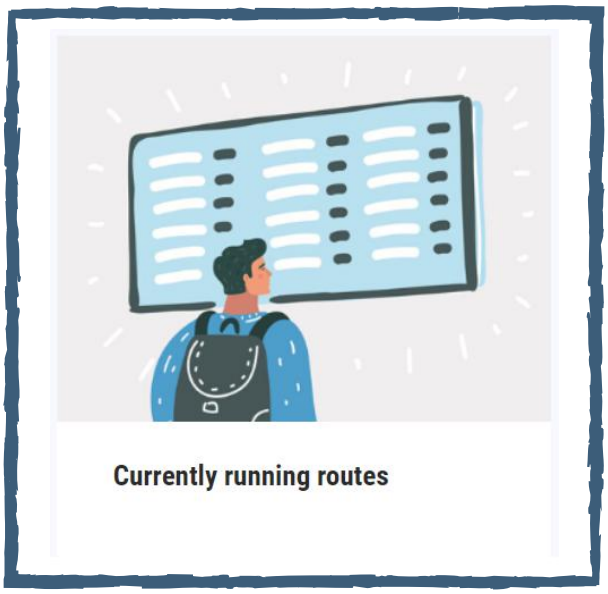
Result Grid | Filter Rows: | Edit: | Export/Import:

FlID	FlightDate	DepartureTime	ArrivalTime	AirCraft	Netfare
1	2020-01-23 00:00:00	2020-01-23 23:20:00	2020-01-24 01:20:00	1	1001
2	2020-02-28 00:00:00	2020-01-28 23:25:00	2020-01-29 02:30:00	3	1003
NULL	NULL	NULL	NULL	NULL	NULL

Pic : Snapshot from SQL Workbench

displayed records Successfully

Without LOGIN Section Queries:



Snapshot from GUI representation

displaying currently running routes Record

Routes database contain the following Information :

1. ***RtID*** *(unique for every route)*
2. ***Destination*** *(name of the destination city)*
3. ***Airport*** *(name of the origin city)*

Displaying Values from Route table

select RtID, destination, airport from **Route**;

displaying records

Result Grid | Filter Rows:

RtID	destination	airport
1	Pokhara	Kathmandu
2	Kathmandu	Pokhara
3	Mumbai	Delhi
4	Delhi	Mumbai
5	Surat	Delhi
6	Delhi	Surat
7	Surat	Jaipur
8	Jaipur	Surat
9	Indore	Jaipur
10	Jaipur	Indore
11	Chennai	Mumbai
12	Hyderabad	Mumbai
13	Mumbai	Hyderabad
14	Mumbai	Chennai
15	Hyderabad	Chennai
16	Chennai	Hyderabad
17	Chennai	Delhi
18	Delhi	Bangalore
19	Bangalore	Delhi
20	Jammu	Delhi
NULL	NULL	NULL

displayed records Successfully

Pic : Snapshot from SQL Workbench

Without LOGIN Section Queries:

Snapshot from GUI representation

displaying airfares Record

States database contain the following Information :

1. ***AfID*** *(unique for every Airfare)*
2. ***Route*** *(unique RouteID from the route database)*
3. ***Fare*** *(charges for that route)*

Displaying Values from Airfare table

select AfID, Route, Fare from **Airfare**;

displaying records

Result Grid | Filter R

AfID	Route	Fare
1001	1	1250
1002	2	1250
1003	3	2000
1004	4	2000
1005	5	1500
1006	6	1500
1007	7	1000
1008	8	1000
1009	9	1650
1010	10	1650
1011	11	2100
1012	12	2400
1013	13	2400
1014	14	2100
1015	15	3600
1016	16	3600
1017	17	4000
1018	18	5000
1019	19	5000
1020	20	3600
NULL	NULL	NULL

displayed records Successfully

Pic : Snapshot from SQL Workbench

Without LOGIN Section Queries:

Snapshot from GUI representation

displaying Employees Record

States database contain the following Information :

1. ***EmpID*** *(unique for every employee)*
2. ***Name*** *(name of the employee)*
3. ***Address*** *(address of the employee)*
4. ***Branch*** *(BranchID of the state working)*
5. ***Email*** *(email of the employee)*

Displaying Values from Employee table

select EmpID, Name, Address, Branch, Email from **Employee**;

displaying records

Result Grid | Filter Rows: | Edit: | Export/Impor

EmpID	Name	Address	Branch	Email
1	Diwan Adhikari	Bagbazaar - 11, KTM	101	the.one@yahoo.com
2	Mayank Jindal	Mansarovar-East	102	jindal@gmail.com
3	Namit Varshney	Sanchi-West	103	namit@outlook.com
4	Ankit Grover	Kapaswaram-East	105	grover@outlook.com
5	Aaarushi Goyal	Maheshwaram-North	104	goyal@outlook.com
6	Sushant	Jammu-WEST	106	sushant@gmail.com
7	Riya Kumar	Bangalore-east	107	rkumar@gmail.com
8	Sushmita	Parvat Patia	108	sushmita@outlook.com
9	Devvrat Singh	Krishna Nagar	109	Dsingh@gmail.com
10	Srijan Gautam	D- Block Geeta Colony	109	srijan@yahoo.com
NULL	NULL	NULL	NULL	NULL

Pic : Snapshot from SQL Workbench

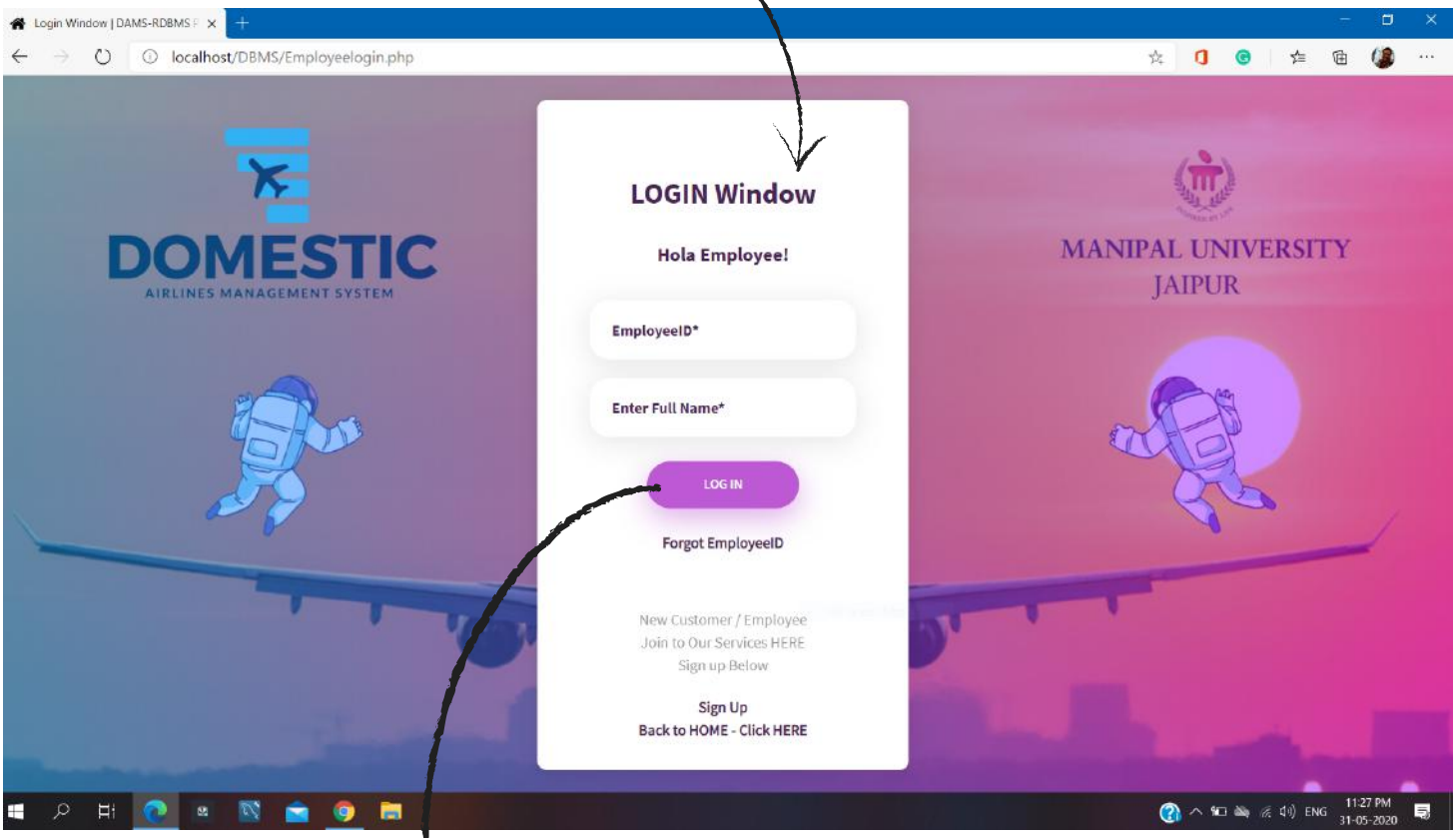
displayed records Successfully

With LOGIN Section Queries: Employee

*For **gaining access into the Employee section** & services, **An Employee needs to login into the system with his/her login credentials*** as represented in the snapshot of the actual GUI Interface designed for Employee of Domestic Airlines Management System.

The Employee Login window asks for the following credentials :

1. **EmployeeID** *(unique for every Employee)*
2. **Full Name** *(as entered while signing up for DAMS)*



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

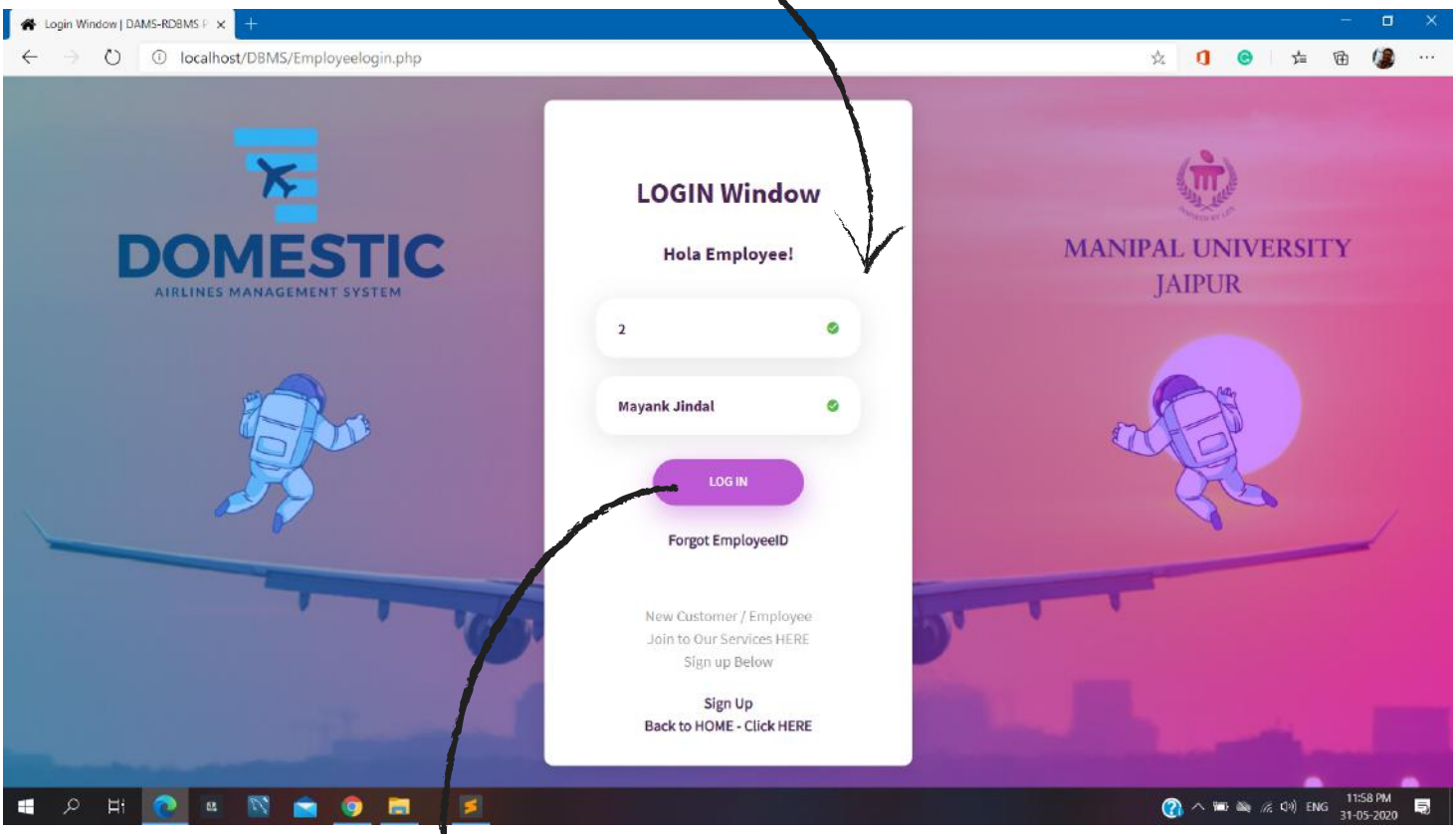
After an employee enters the details - clicking on the LOGIN button runs the following SQL query in the database: (The SQL queries are written as such to accept the values from the user input)

$sql = select * from Employee where EmpID='".$userid."' AND Name='".$username."' limit 1;

We will now see the actual working of the Employee login page, and then proceeding to their roles which can only be accessed by them.

With LOGIN Section Queries: Employee

*Now, as I have created the database - **Mayank Jindal with EmployeeID = 2 is an employee in the DAMS**. Let us enter these details in the form as shown below:*



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

*After the details **EmployeeID = 2 and Name = Mayank Jindal** are entered, and then the employee click on the login button -> The following sql query runs in the sql:*

$sql = select * from Employee where EmpID= '2' AND Name= 'Mayank Jindal' limit 1;

Result Grid | Filter Rows: | Edit:

EmpID	Name	Address	Branch	Email
2	Mayank Jindal	Mansarovar-East	102	jindal@gmail.com
NULL	NULL	NULL	NULL	NULL

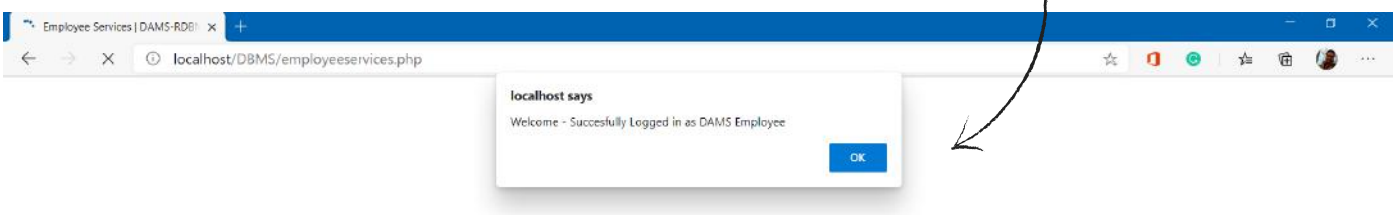
Pic : Snapshot from SQL Workbench

The query runs in the background, and finds that EmployeeID = 2 and Name= Mayank Jindal exists, so it ***validates the employee and outputs the dialogue box as shown in the next page.***

- SQL QUERIES ACCORDING TO GUI FUNCTIONALITIES

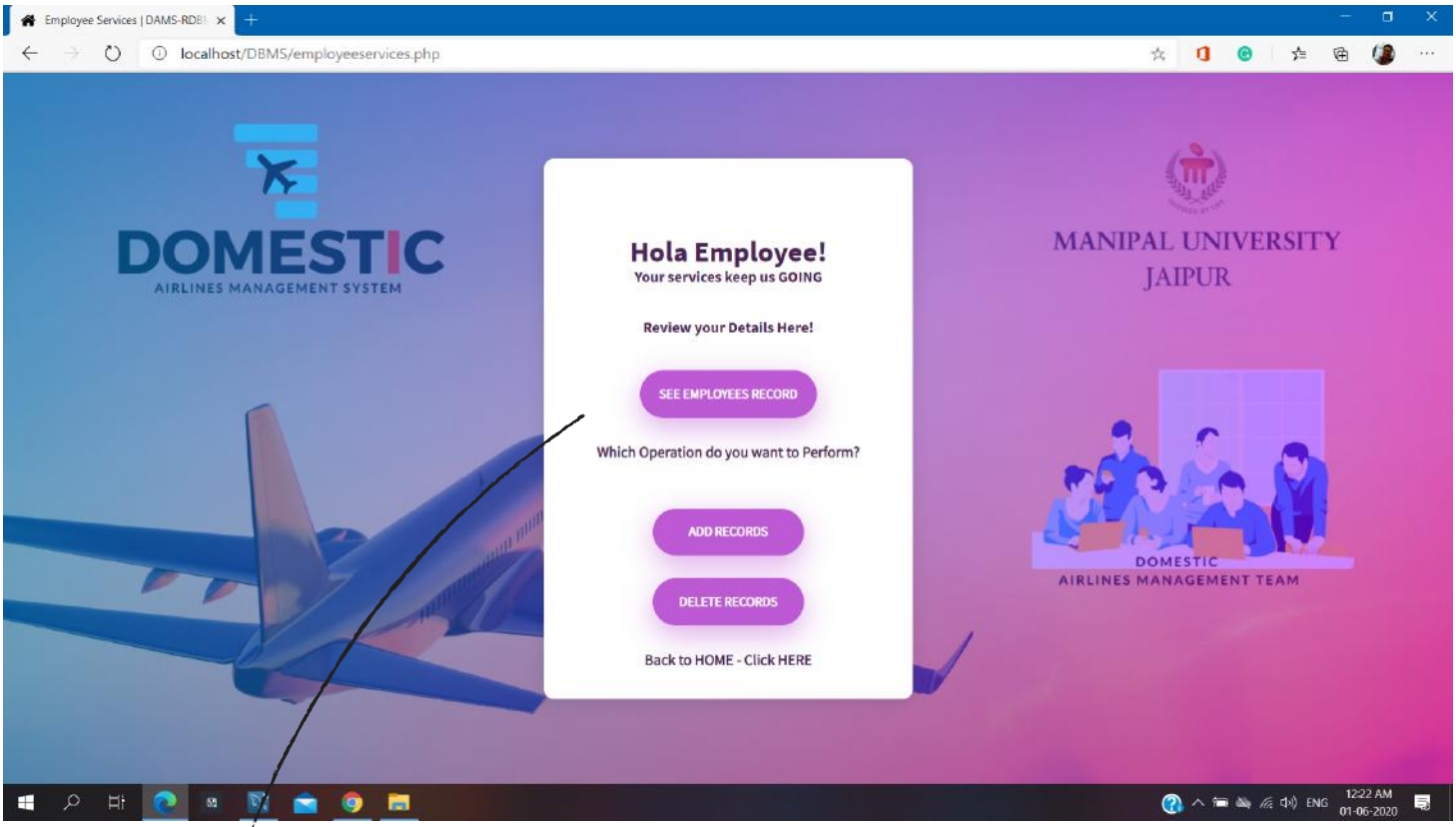
With LOGIN Section Queries: Employee

The Dialogue box validates that, the employee has logged in with the correct credentials.



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

After Clicking on OK. The employee gets his tool box and access kit - ***The Employee have access to see the Employees Records and do the operations like Adding a new record and deleting and existing record.***



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

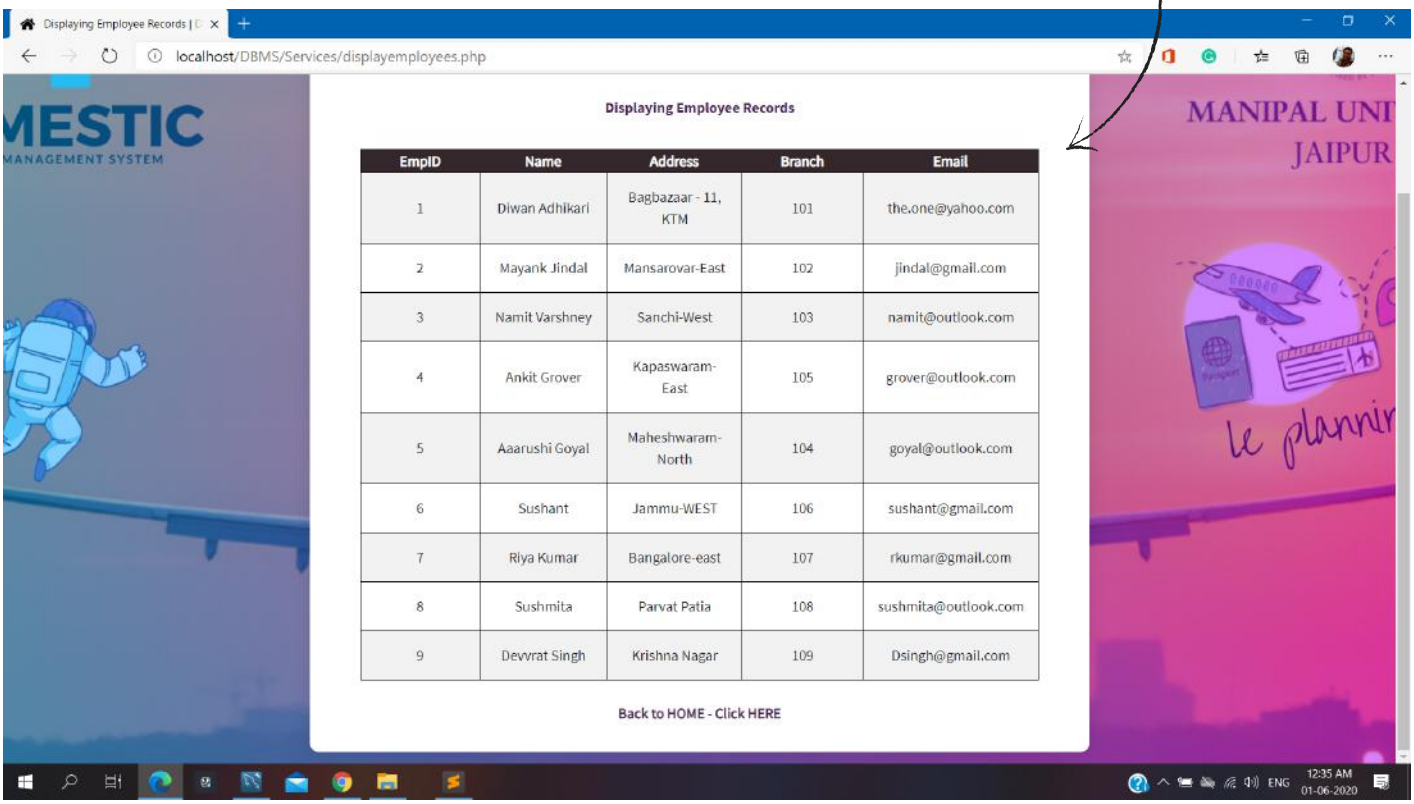
Employee Tool Box

1. *See All Employee Records*
2. *Add New Records (to the database)*
3. *Delete Existing Records from the database.*

- **SQL QUERIES ACCORDING TO GUI FUNCTIONALITIES**

With LOGIN Section Queries: Employee

*After an employee clicks on the **See Employee Records** button, the following output is displayed.*



EmpID	Name	Address	Branch	Email
1	Diwan Adhikari	Bagbazaar - 11, KTM	101	the.one@yahoo.com
2	Mayank Jindal	Mansarovar-East	102	jindal@gmail.com
3	Namit Varshney	Sanchi-West	103	namit@outlook.com
4	Ankit Grover	Kapaswaram-East	105	grover@outlook.com
5	Aaarushi Goyal	Maheshwaram-North	104	goyal@outlook.com
6	Sushant	Jammu-WEST	106	sushant@gmail.com
7	Riya Kumar	Bangalore-east	107	rkumar@gmail.com
8	Sushmita	Parvat Patia	108	sushmita@outlook.com
9	Devvrat Singh	Krishna Nagar	109	Dsingh@gmail.com

*Pic : Snapshot from actual **GUI representation** running at **localhost server***

BACKEND *: After the button See Employees Record is clicked, -> The following sql query runs in the sql:*

select EmpID, Name, Address, Branch, Email from Employee;

We can surely observe, that both the outputs are showing the results - which gives us a sense of accuracy

Result Grid | Filter Rows: | Edit: | Export/Impo

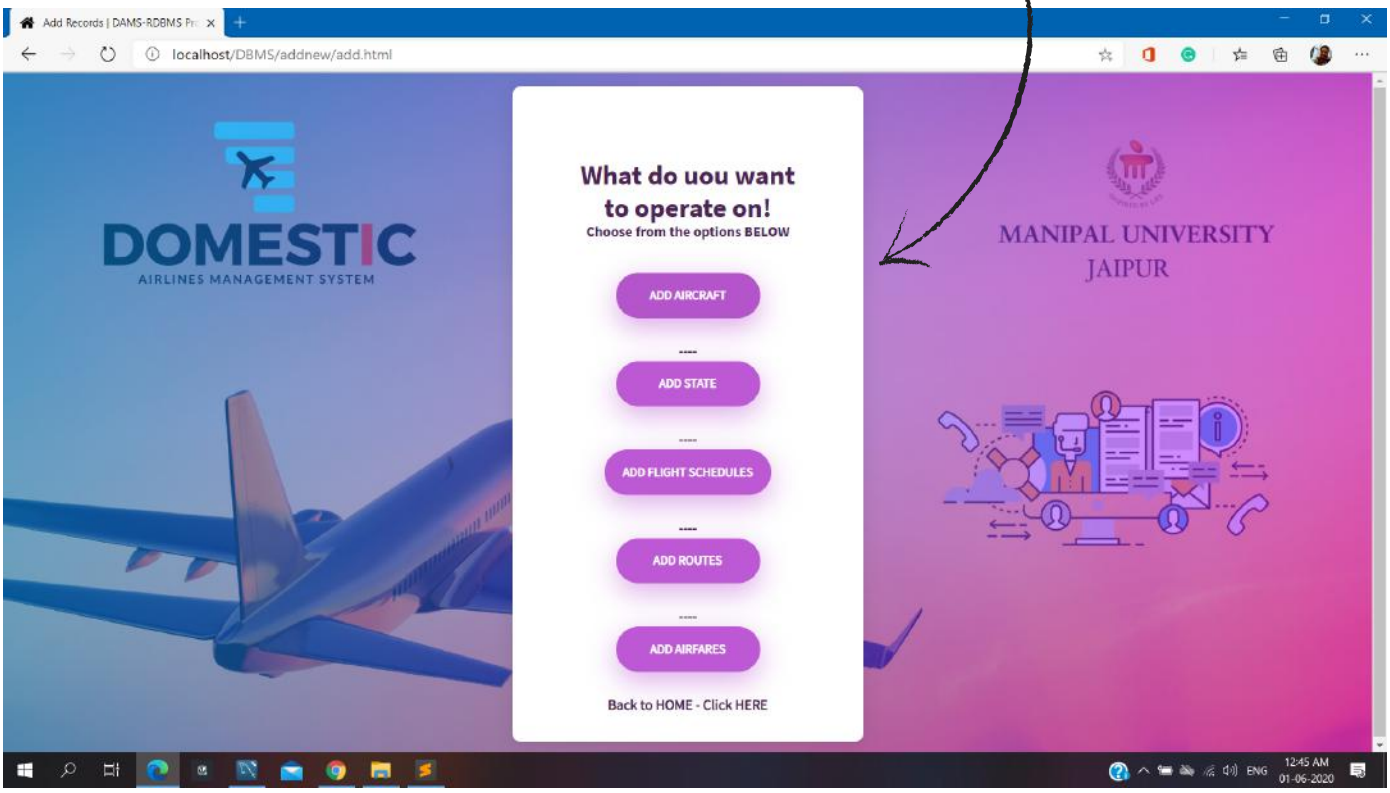
EmpID	Name	Address	Branch	Email
1	Diwan Adhikari	Bagbazaar - 11, KTM	101	the.one@yahoo.com
2	Mayank Jindal	Mansarovar-East	102	jindal@gmail.com
3	Namit Varshney	Sanchi-West	103	namit@outlook.com
4	Ankit Grover	Kapaswaram-East	105	grover@outlook.com
5	Aaarushi Goyal	Maheshwaram-North	104	goyal@outlook.com
6	Sushant	Jammu-WEST	106	sushant@gmail.com
7	Riya Kumar	Bangalore-east	107	rkumar@gmail.com
8	Sushmita	Parvat Patia	108	sushmita@outlook.com
9	Devvrat Singh	Krishna Nagar	109	Dsingh@gmail.com
NULL	NULL	NULL	NULL	NULL

Pic : Snapshot from SQL Workbench

- SQL QUERIES ACCORDING TO GUI FUNCTIONALITIES

With LOGIN Section Queries: Employee

*Proceeding towards **employee services**, After an employee clicks on the **Add Records** button, the following page is displayed showing different options :*



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

The employee can add new records to the following tables present in the database :

Aircrafts :
If the company adds a new aircrafts, it is the duty of employee to add it to the database.

States :
expanding horizons of service, the employee can introduce a new state with a unique state id.

Flight Schedules :
maintaining the flight schedules is the sole duty of an employee.

Routes :
Maintaining records of the origin and destination

Airfares *:*
Defining and adding airfares for the new routes, is the sole duty of an employee

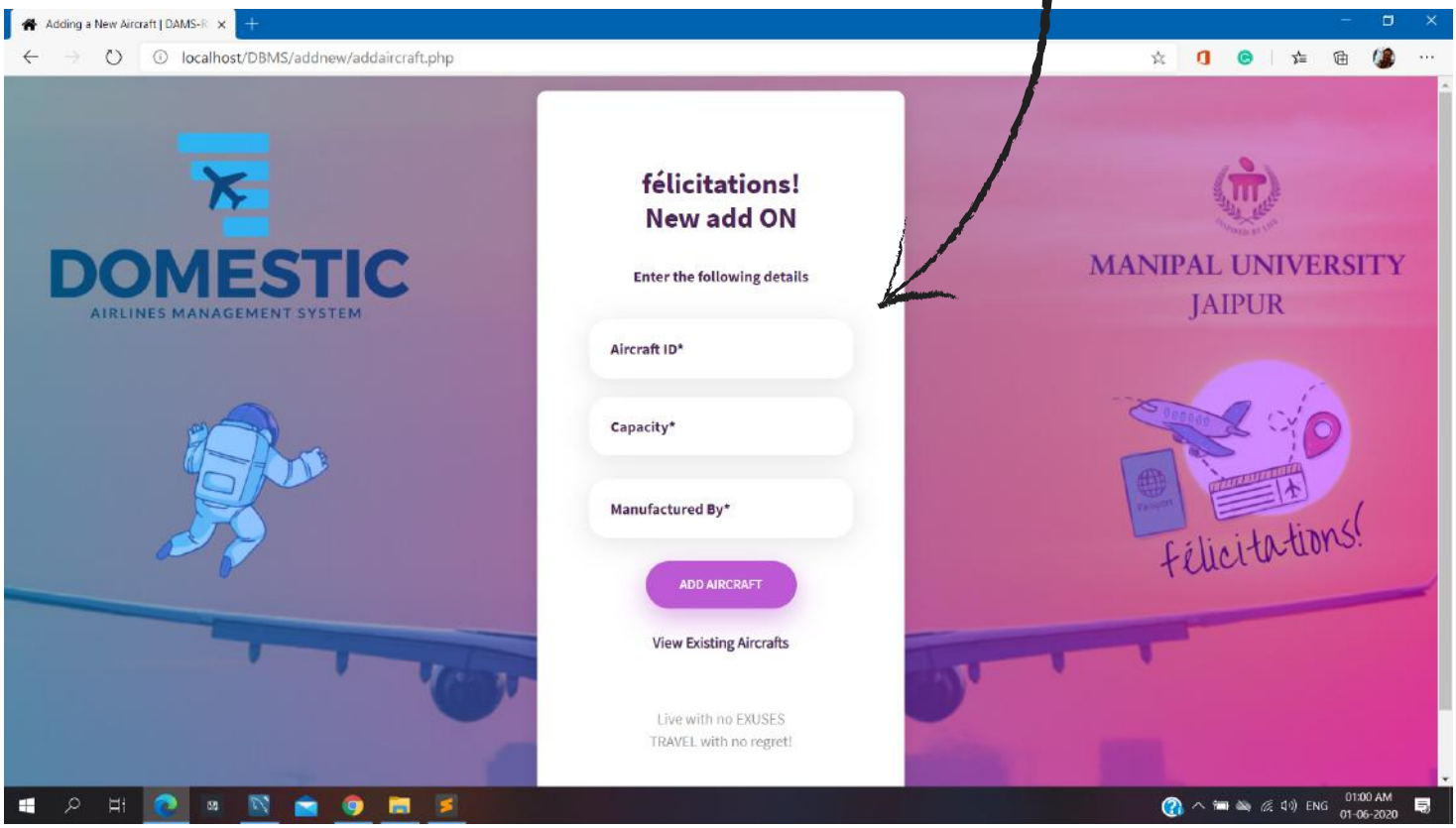
*Let us assume, that we want to add a new aircraft to the database. So the employee click on **Add Aircraft** button.*

We will see the working of the Add Aircraft section in the next page. (All other actions work the same way)

With LOGIN Section Queries: Employee

*Proceeding towards **employee services**, After an employee clicks on the **Add Aircrafts** button, the following page is displayed asking us to enter the following details:*

1. ***AircraftID** (unique for every aircraft)*
2. ***Capacity** (no. of passengers an aircraft can hold)*
3. ***Manufactured By** (name of the manufacturing company of the Aircraft)*



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

*After an employee enters the **Aircraft details** - clicking on the **Add Aircraft button** runs the following SQL query in the database: (The SQL queries are written as such to accept the values from the user input php form)*

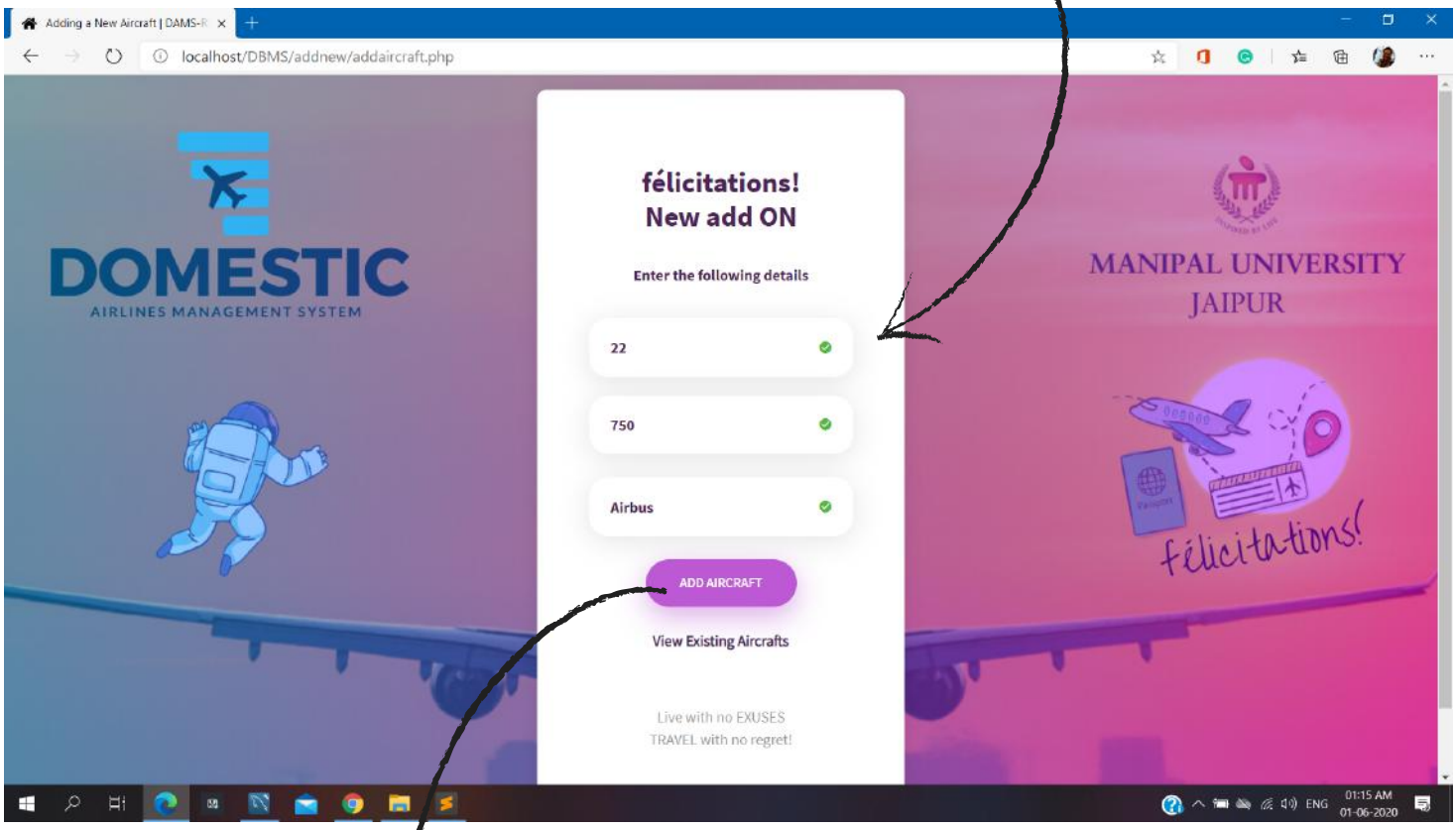
```
$query = mysqli_query($connection, "insert into Aircrafts(AcID, Capacity, MfdBy) values ($acid, $capacity, '$mfdby')");
```

We will now see the actual working of the Add Aircraft page, and then proceeding to see the updated database of the aircrafts.

With LOGIN Section Queries: Employee

Now, Let us enter these details in the form as shown below:

1. ***AircraftID*** *(unique for every aircraft)* = ***22***
2. ***Capacity*** *(no. of passengers an aircraft can hold)* = ***750***
3. ***Manufactured By*** *(name of the manufacturing company of the Aircraft)* = ***Airbus***



Pic : Snapshot from actual ***GUI representation*** *running at* ***localhost server***

BACKEND

After an employee enters the ***Aircraft details*** *- clicking on the* ***Add Aircraft button*** *runs the following SQL query in the database:*

INSERT into AirCrafts (AcID, Capacity, MfdBy) VALUES (22, 750, "Airbus");

New Aircraft Added Successfully:

(if the addition of new Aircraft is successful, the page displays the following message as shown on the next page.)

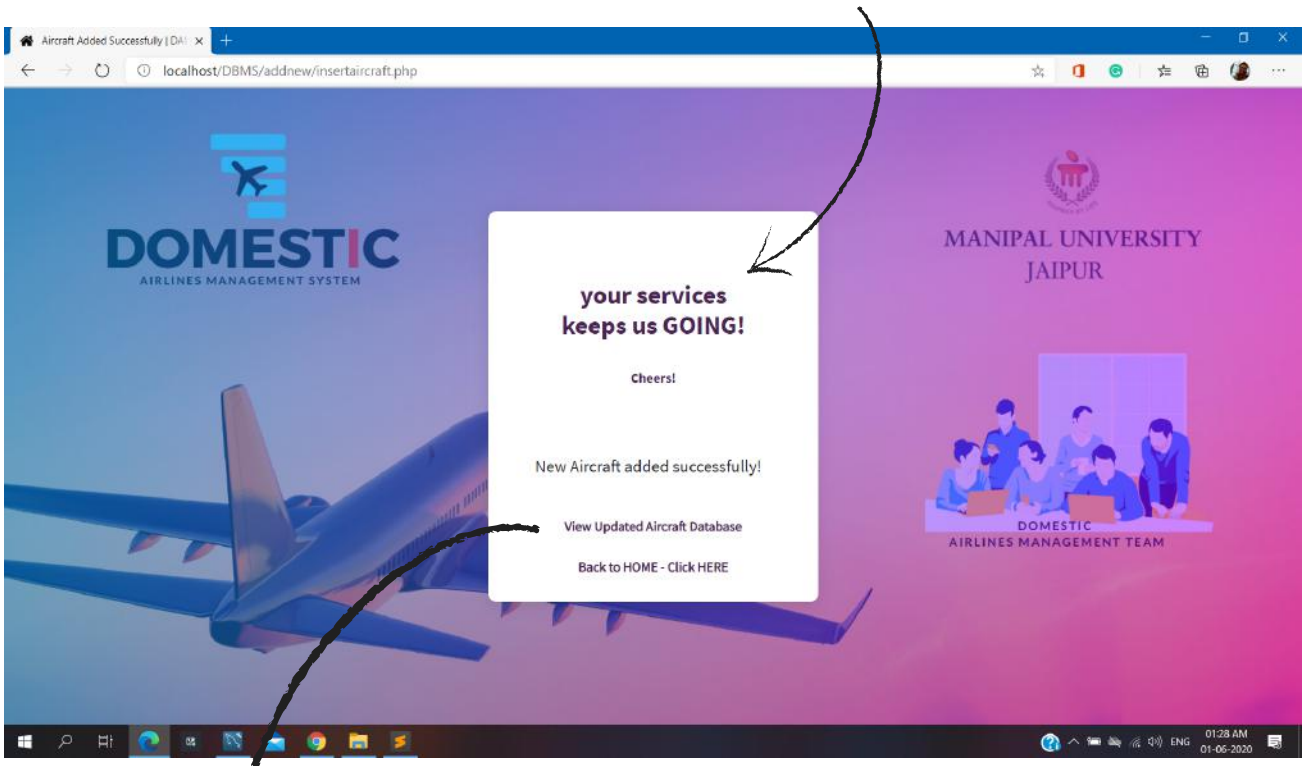
12	500	Alenia Aeronotica
13	95	Alenia Aeronotica
14	100	Airbus Corporate
15	200	Alenia Aeronotica
16	250	Alenia Aeronotica
17	250	Alenia Aeronotica
18	300	Textron Aviation
19	300	Textron Aviation
20	300	Textron Aviation
21	500	Textron Aviation
22	750	Airbus
NULL	NULL	NULL

Pic : Snapshot from SQL Workbench

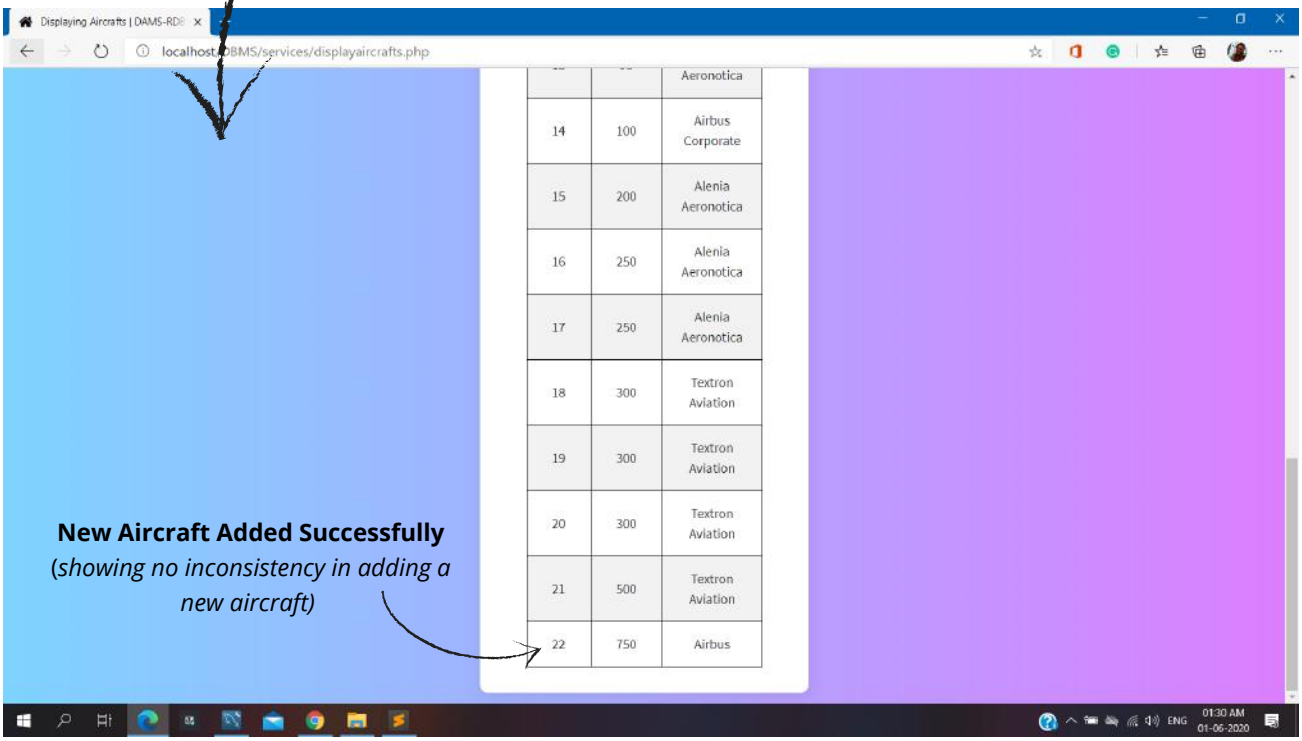
- SQL QUERIES ACCORDING TO GUI FUNCTIONALITIES

With LOGIN Section Queries: Employee

New Aircraft Added Successfully:
if the addition of new Aircraft is successful, the page displays the following message as shown below :



*Pic : Snapshot from actual **GUI representation** running at **localhost server***



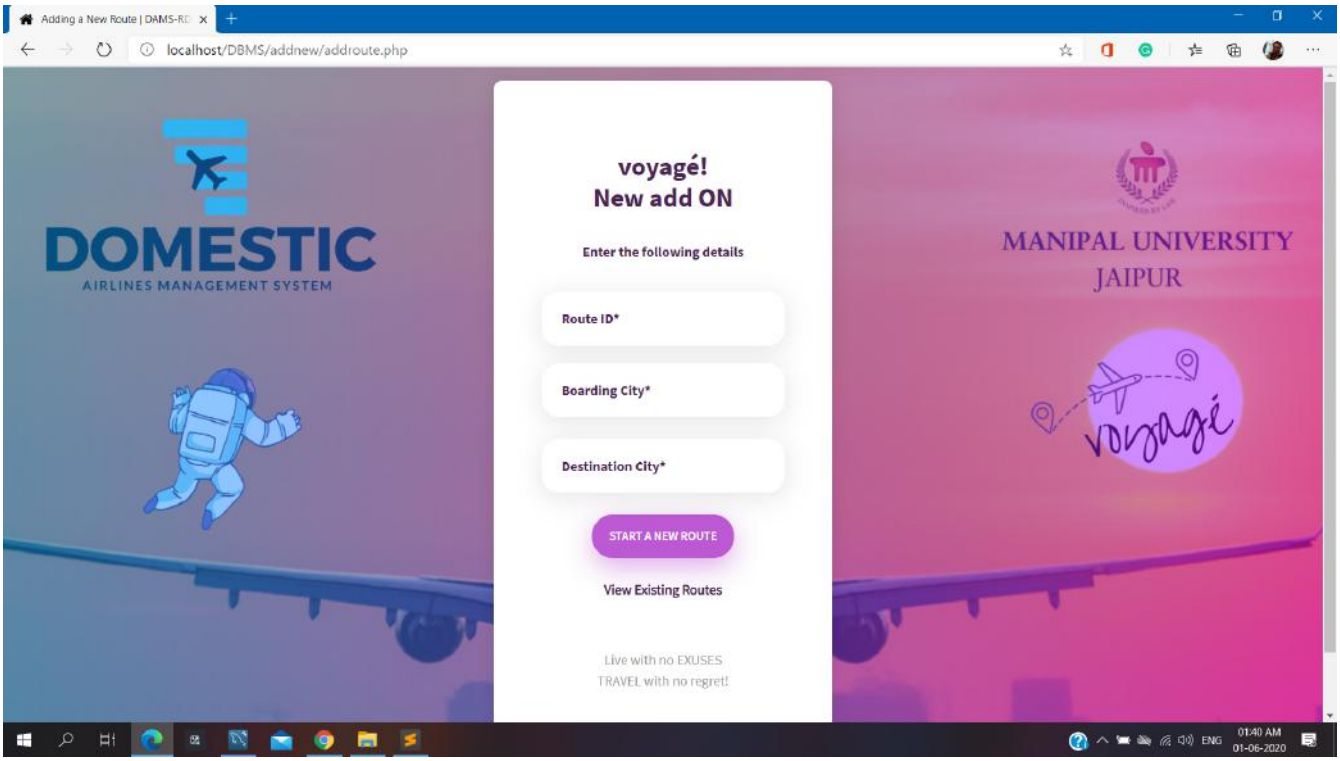
*Pic : Snapshot from actual **GUI representation** running at **localhost server***

In the same way , Addition of new route, flight_schedule takes place that is why I am only showing their forms in the next page

- SQL QUERIES ACCORDING TO GUI FUNCTIONALITIES

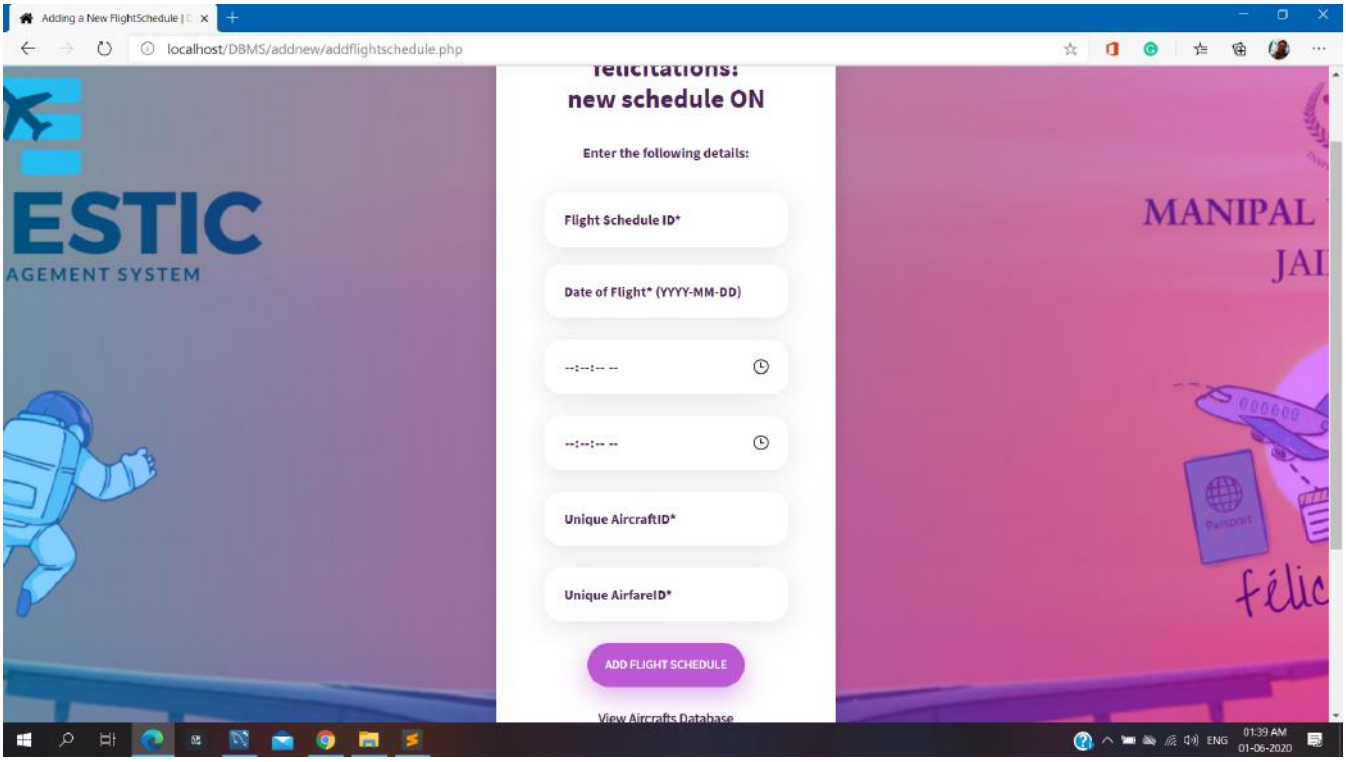
With LOGIN Section Queries: Employee

New Route Addition form:



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

New Flight Schedule Addition form:

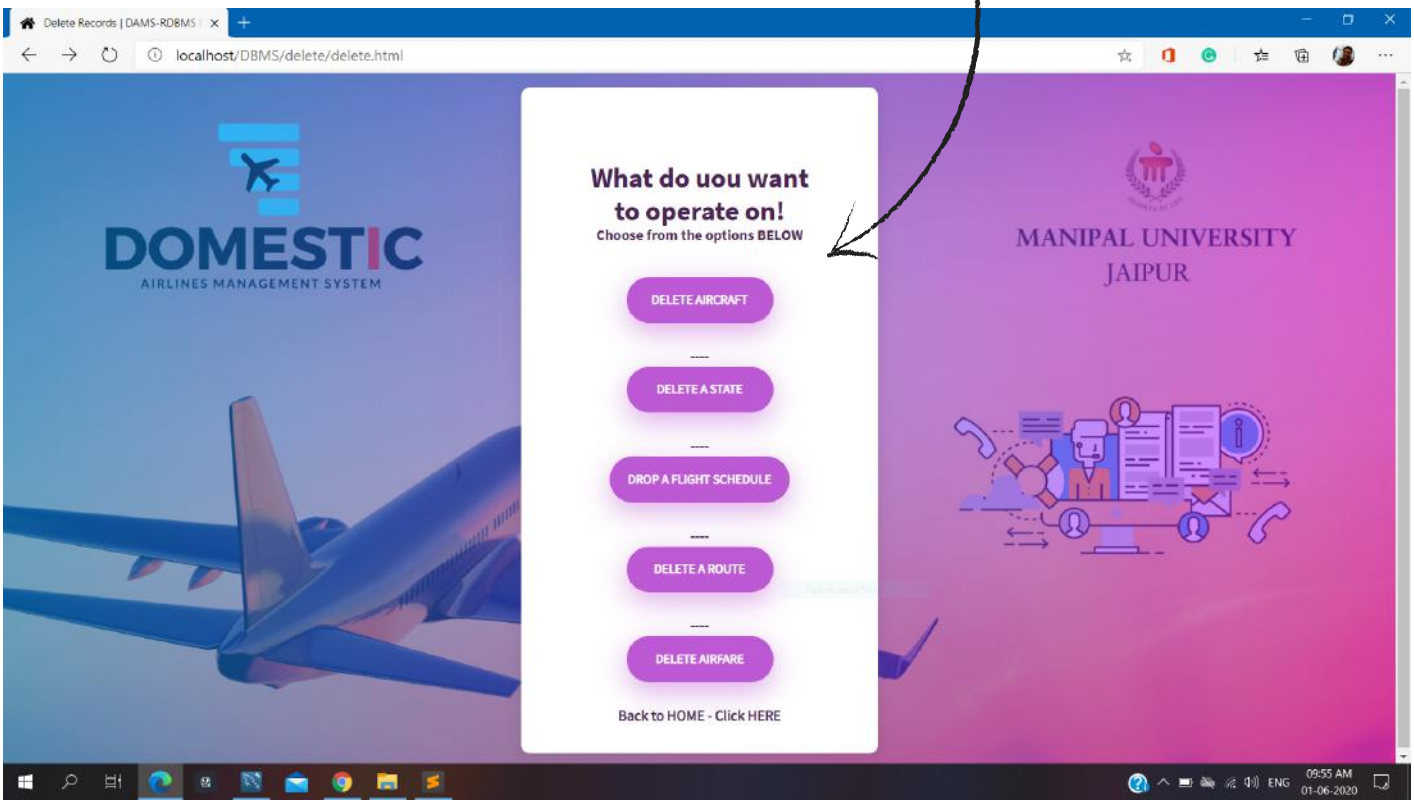


*Pic : Snapshot from actual **GUI representation** running at **localhost server***

- SQL QUERIES ACCORDING TO GUI FUNCTIONALITIES

With LOGIN Section Queries: Employee

*Proceeding towards **employee services**, After an employee clicks on the **Delete Records** button, the following page is displayed showing different options :*



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

The employee can delete records of the following tables present in the database :

Aircrafts :
If the company discards an aircraft, it is the duty of employee to add it to the database.

States :
for any known issue, the employee can abandon a state with a unique state id.

Flight Schedules :
maintaining the flight schedules is the sole duty of an employee.

Routes :
Maintaining records of the origin and destination

Airfares *:*
Defining and deleting airfares for the abandoned routes, is the sole duty of an employee

*Let us assume, that we want to delete an aircraft from the database. So the employee click on **Delete Aircraft** button.*

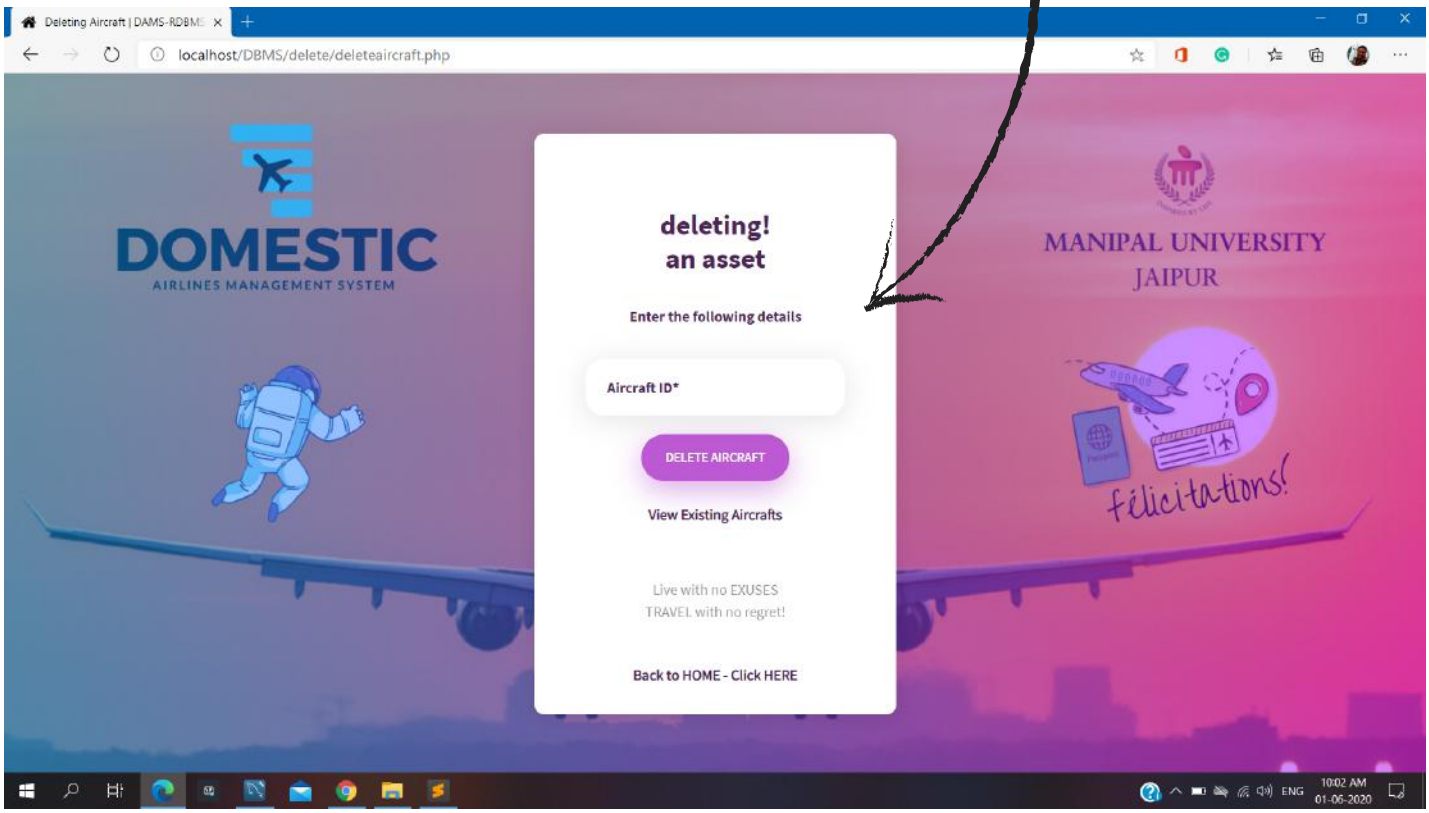
We will see the working of the Delete Aircraft section in the next page. (All other actions work the same way)

With LOGIN Section Queries: Employee

*Proceeding forward, After an employee clicks on the **Delete Aircrafts** button, the following page is displayed asking us to enter the following details:*

1. ***AircraftID** (unique for every aircraft)*

*The employee can also refer to the existing aircarfts database, by clicking on the "**View Existing Aircrafts**" button, in order to prevent deletion of aircraft with accuracy.*



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

*After an employee enters the **Aircraft ID**- clicking on the **Delete Aircraft button** runs the following SQL query in the database: (The SQL queries are written as such to accept the values from the user input php form)*

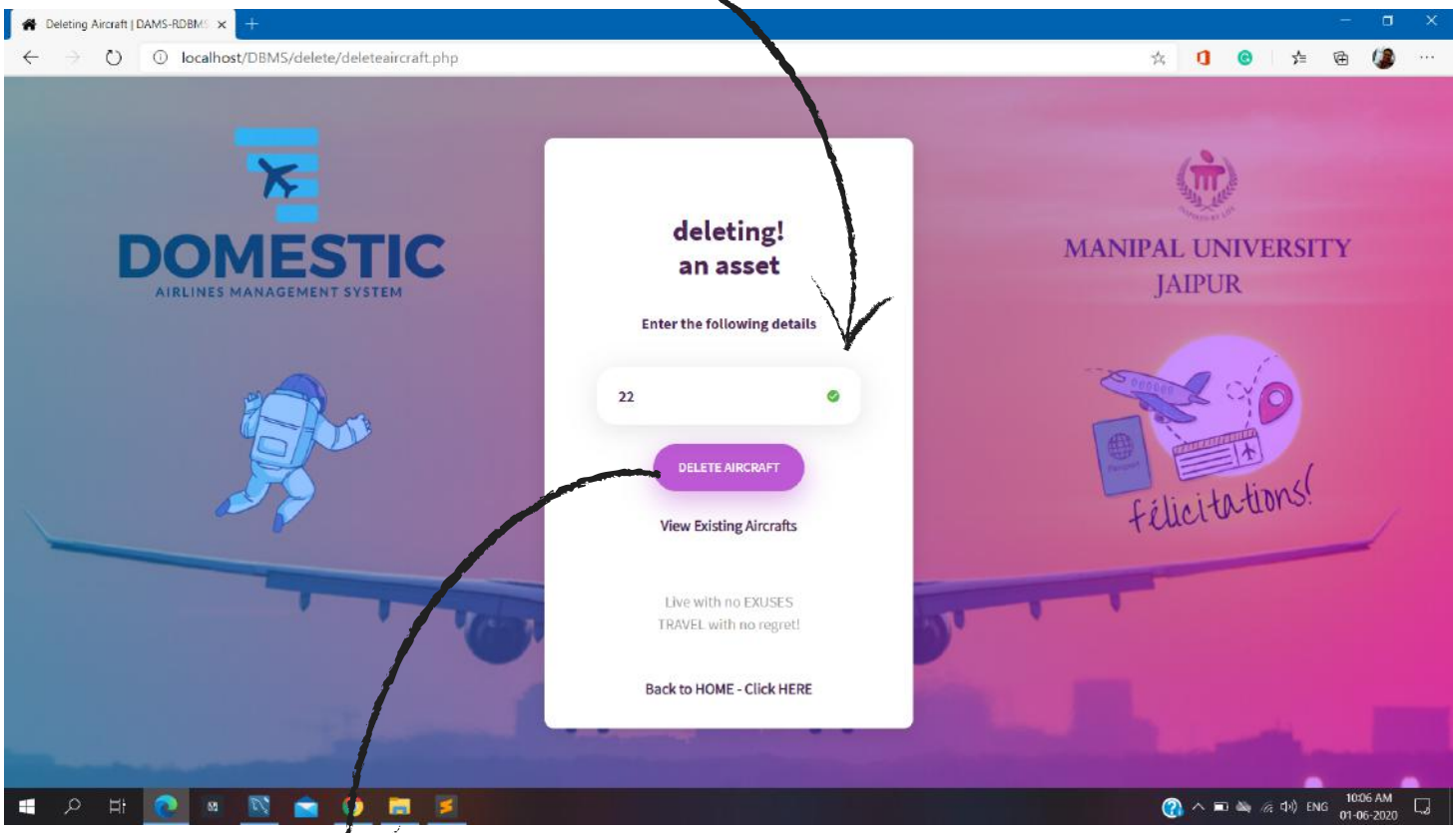
```
$query = mysqli_query($connection, "DELETE FROM Aircrafts WHERE acid='$acid'");
```

We will now see the actual working of the Delete Aircraft page, and then will be proceeding to see the updated database of the aircrafts.

With LOGIN Section Queries: Employee

Now, Let us enter these details in the form as shown below:

1. ***AircraftID*** *(unique for every aircraft)* = ***22***



Pic : Snapshot from actual ***GUI representation*** *running at* ***localhost server***

BACKEND

After an employee enters the ***Aircraft ID****- clicking on the* ***Delete Aircraft button*** *runs the following SQL query in the database:*

DELETE FROM Aircrafts WHERE AcID = 22;

12	500	Alenia Aeronotica
13	95	Alenia Aeronotica
14	100	Airbus Corporate
15	200	Alenia Aeronotica
16	250	Alenia Aeronotica
17	250	Alenia Aeronotica
18	300	Textron Aviation
19	300	Textron Aviation
20	300	Textron Aviation
21	500	Textron Aviation
22	750	Airbus
NULL	NULL	NULL

Pic : Snapshot from SQL Workbench

Deletion in Process

10	500	Gulfstream Aerospace
11	85	Gulfstream Aerospace
12	500	Alenia Aeronotica
13	95	Alenia Aeronotica
14	100	Airbus Corporate
15	200	Alenia Aeronotica
16	250	Alenia Aeronotica
17	250	Alenia Aeronotica
18	300	Textron Aviation
19	300	Textron Aviation
20	300	Textron Aviation
21	500	Textron Aviation
NULL	NULL	NULL

Pic : Snapshot from SQL Workbench

Aircraft deleted Successfully:

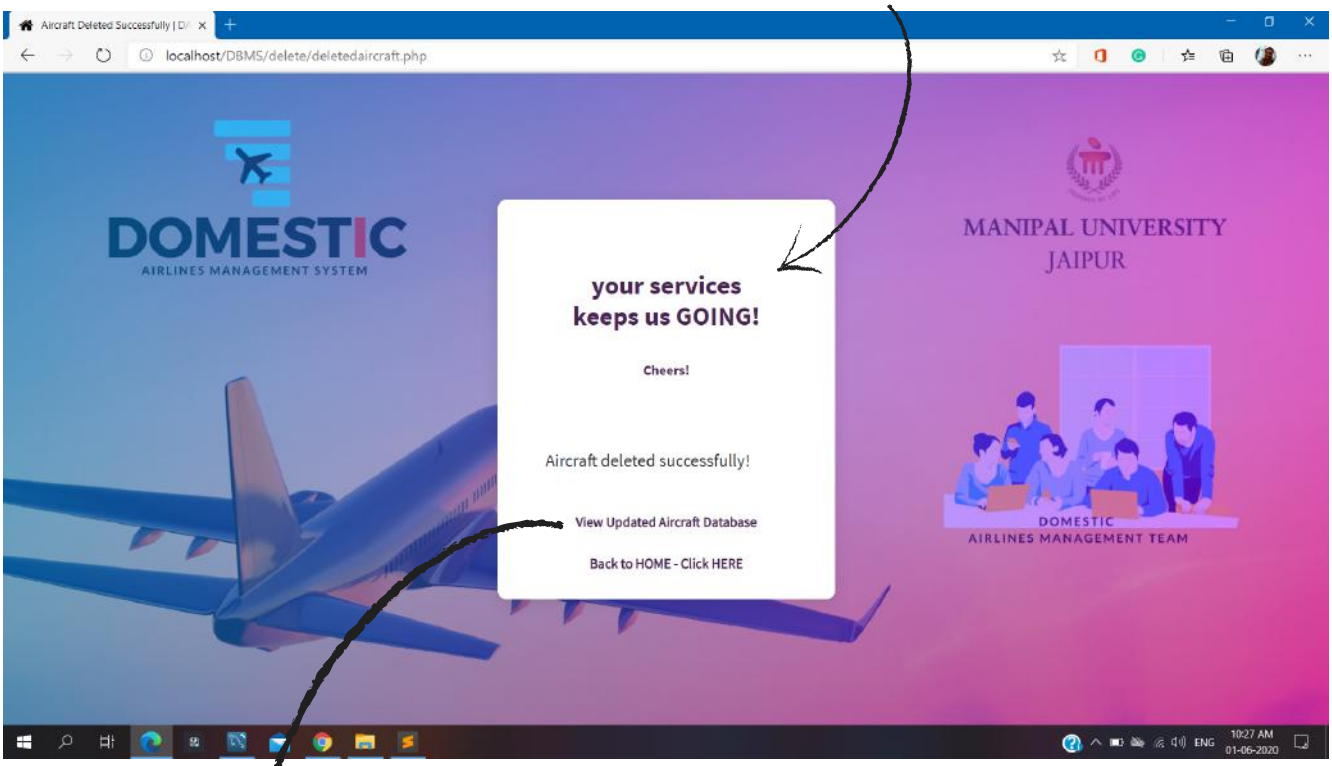
(if the deletion of Aircraft is successful, the page displays the following message as shown on the next page.)

- SQL QUERIES ACCORDING TO GUI FUNCTIONALITIES

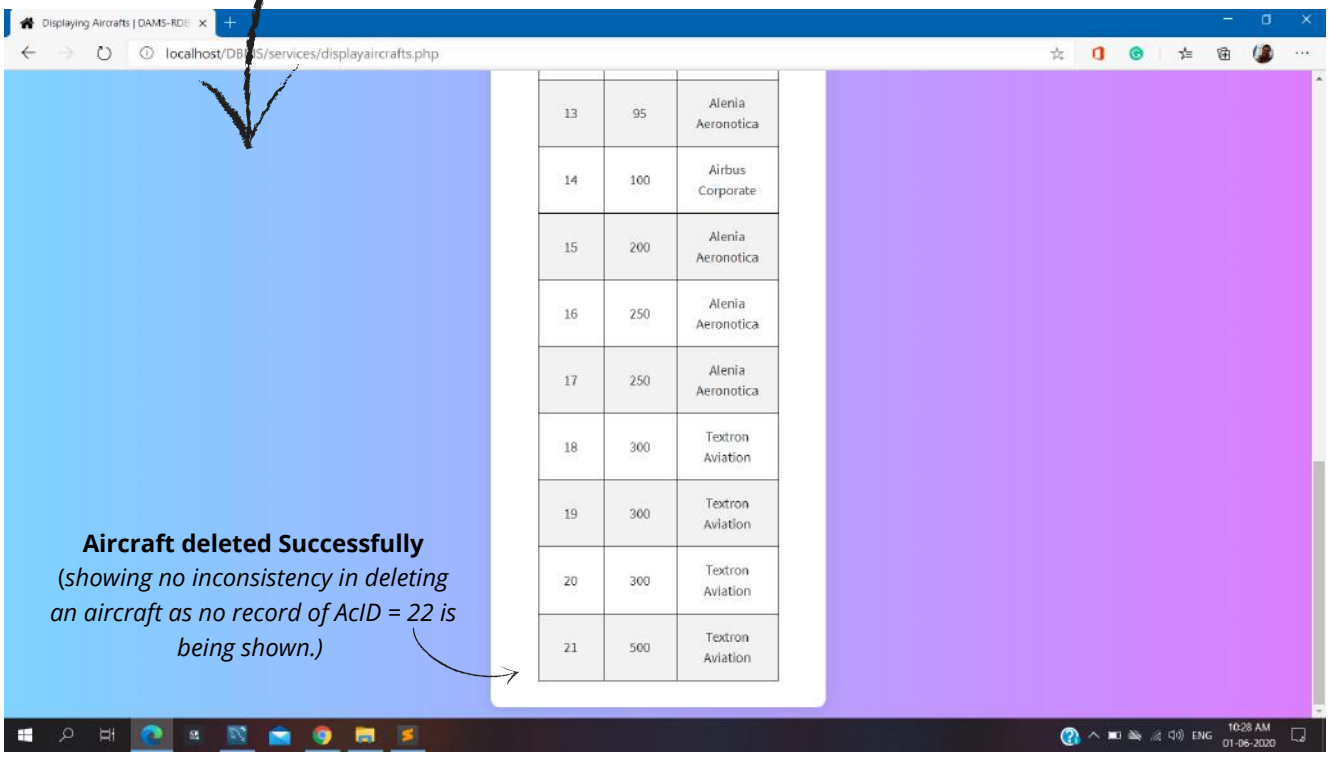
With LOGIN Section Queries: Employee

Aircraft deleted Successfully:

if the deletion of Aircraft is successful, the page displays the following message as shown below :



*Pic : Snapshot from actual **GUI representation** running at **localhost server***



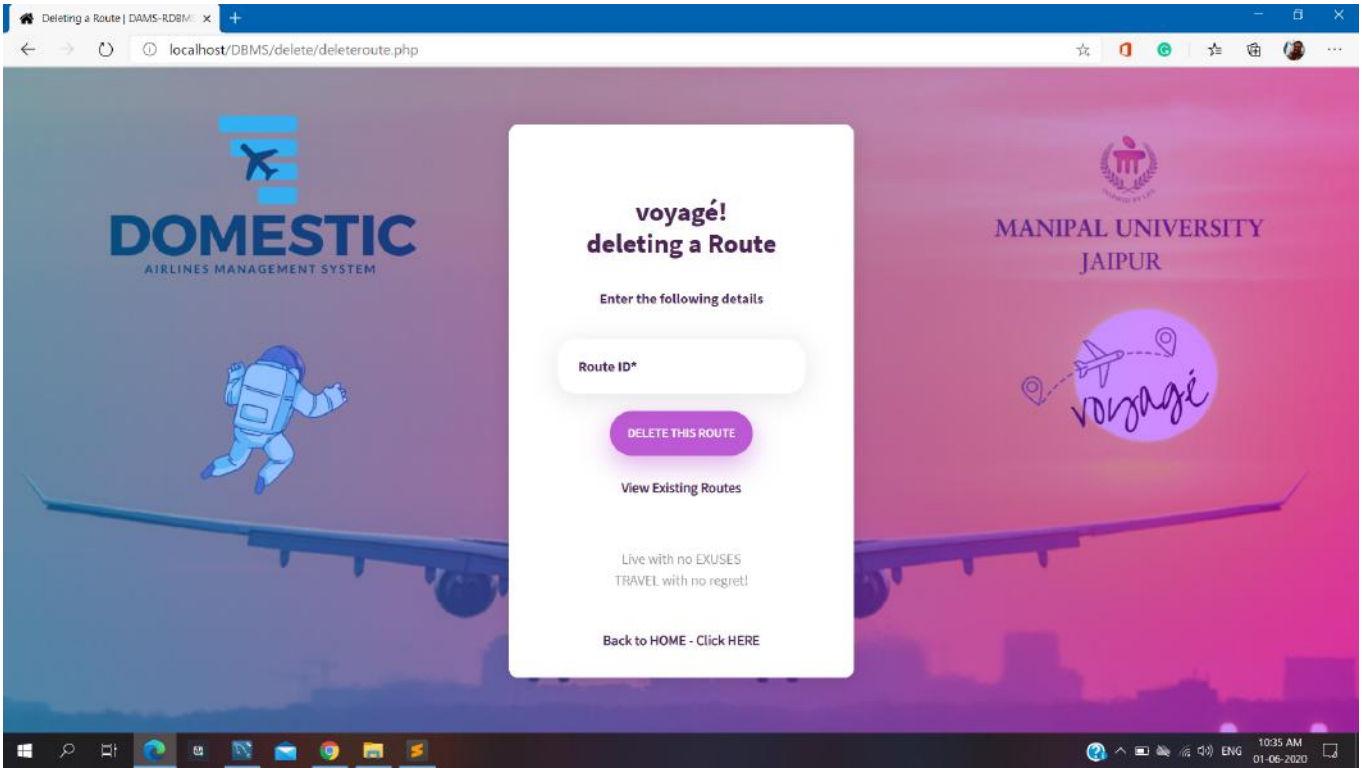
*Pic : Snapshot from actual **GUI representation** running at **localhost server***

In the same way , deletion of a route, flight_schedule takes place that is why, I am only showing their forms in the next page

- SQL QUERIES ACCORDING TO GUI FUNCTIONALITIES

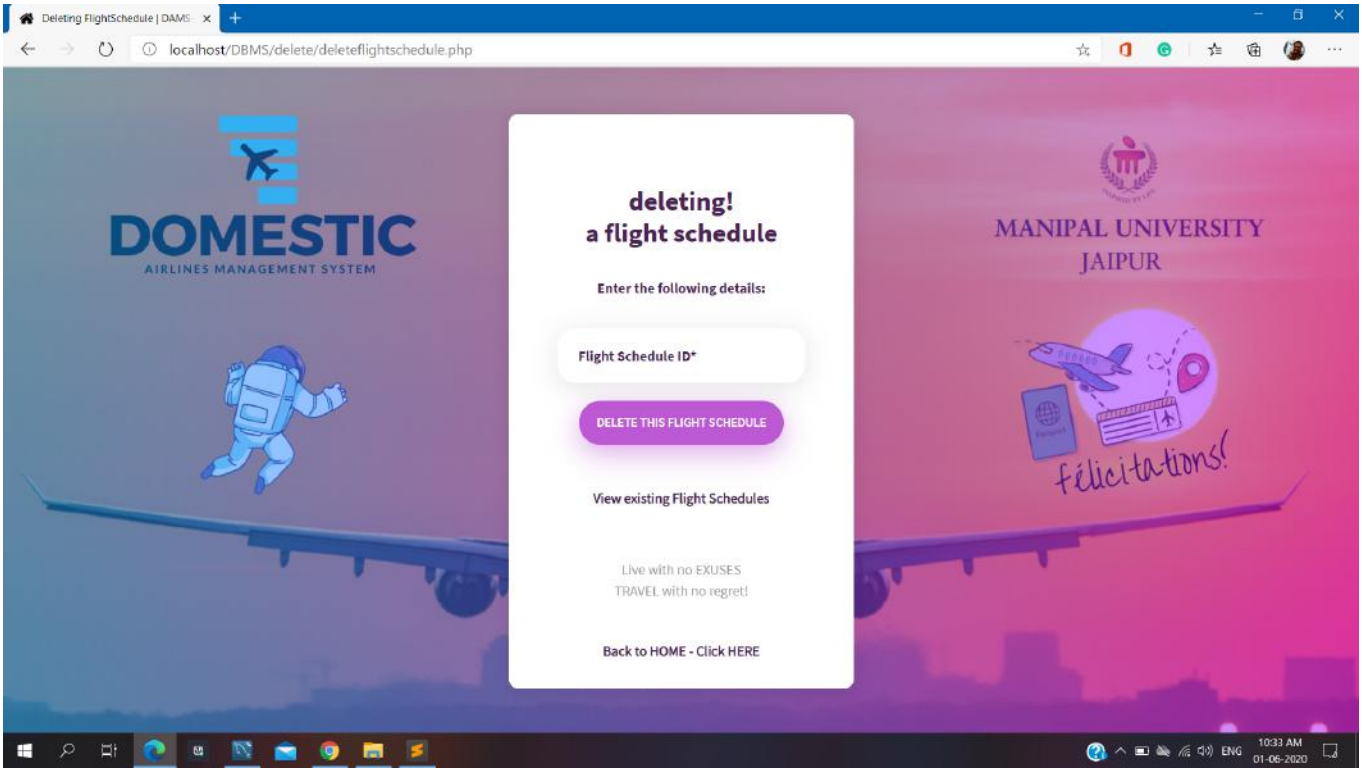
With LOGIN Section Queries: Employee

Deleting route form:



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

Deleting Flight Schedule form:



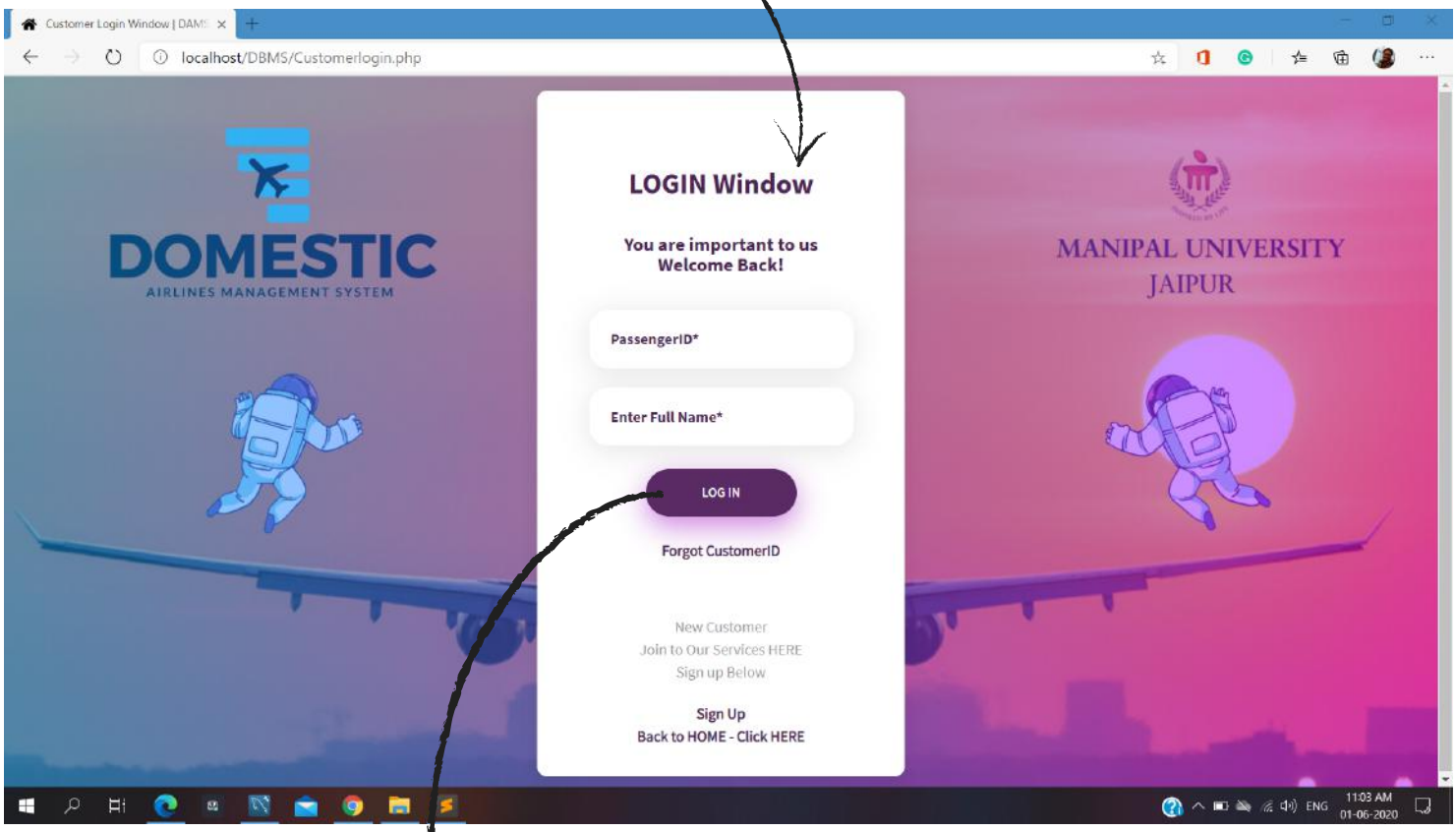
*Pic : Snapshot from actual **GUI representation** running at **localhost server***

With LOGIN Section Queries: CUSTOMER

*For **gaining access into the Customer section** & services, **A customer needs to login into the system with his/her login credentials*** as represented in the snapshot of the actual GUI Interface designed for Employee of Domestic Airlines Management System.

The Employee Login window asks for the following credentials :

1. **PassengerID** *(unique for every Customer)*
2. **Full Name** *(as entered while signing up for DAMS)*



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

After a customer enters the details - clicking on the LOGIN button runs the following SQL query in the database: (The SQL queries are written as such to accept the values from the user input)

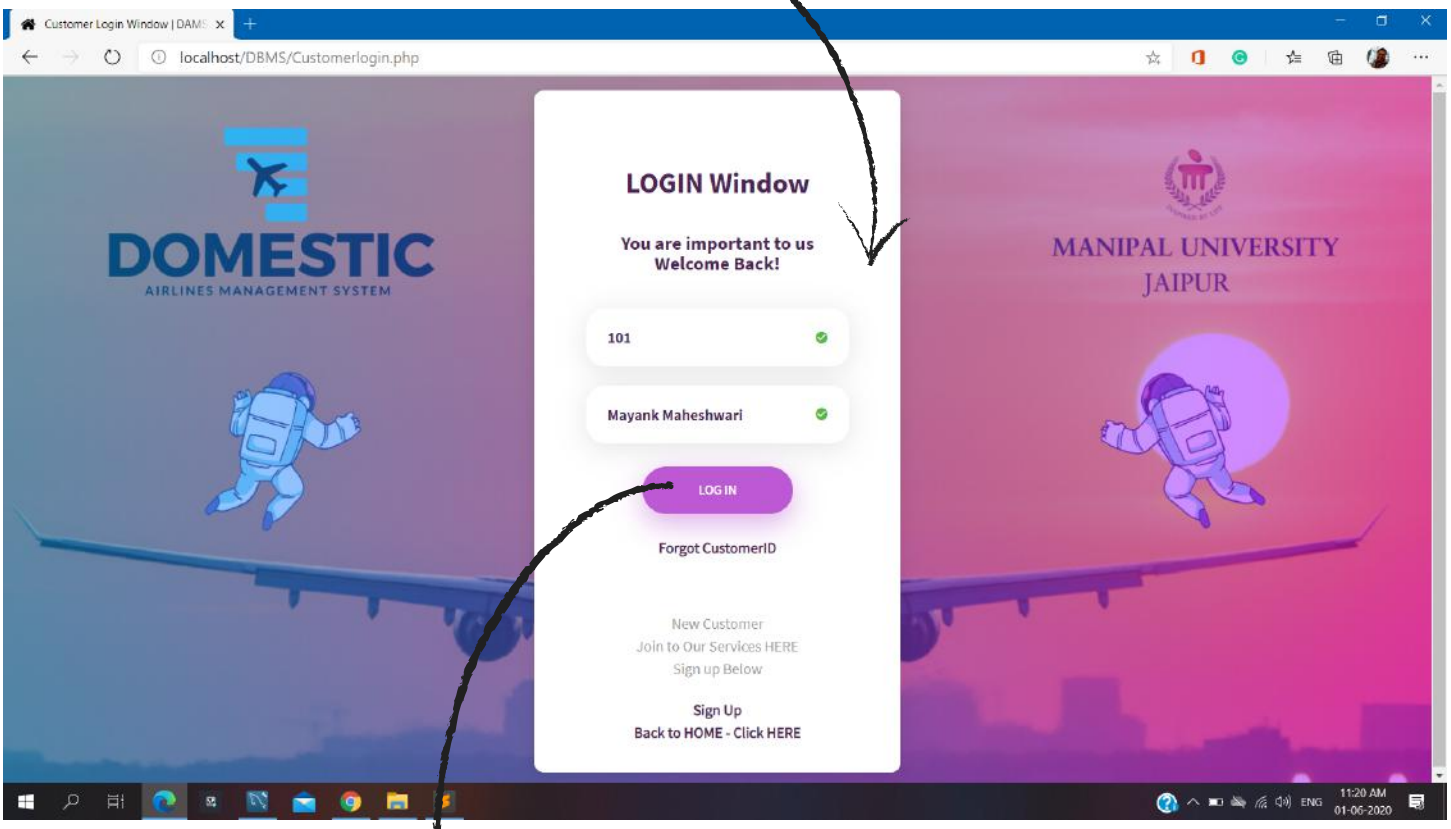
$sql="select * from Passengers where PassengerID='".$userid."'AND Name='".$username."' limit 1";

We will now see the actual working of the Customer login page, and then proceeding to the flight booking system. ***Customer Login is necessary for booking flight tickets.***

- SQL QUERIES ACCORDING TO GUI FUNCTIONALITIES

With LOGIN Section Queries: CUSTOMER

*Now, as I have created the database - **Mayank Maheshwari with EmployeeID = 101 is a registerd customer in the DAMS**. Let us enter these details in the form as shown below:*



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

*After the details **PassengerID= 101 and Name = Mayank Maheshwari** are entered, and then the customer click on the login button -> The following sql query runs in the sql:*

select * from Passengers where PassengerID= '101' AND Name= 'Mayank Maheshwari' limit 1;

Result Grid | Filter Rows: | Edit: | Export/Import:

PassengerID	Name	Address	Age
101	Mayank Maheshwari	Manipal University Jaipur,Dehmi Kalan,Rajasthan	20
NULL	NULL	NULL	NULL

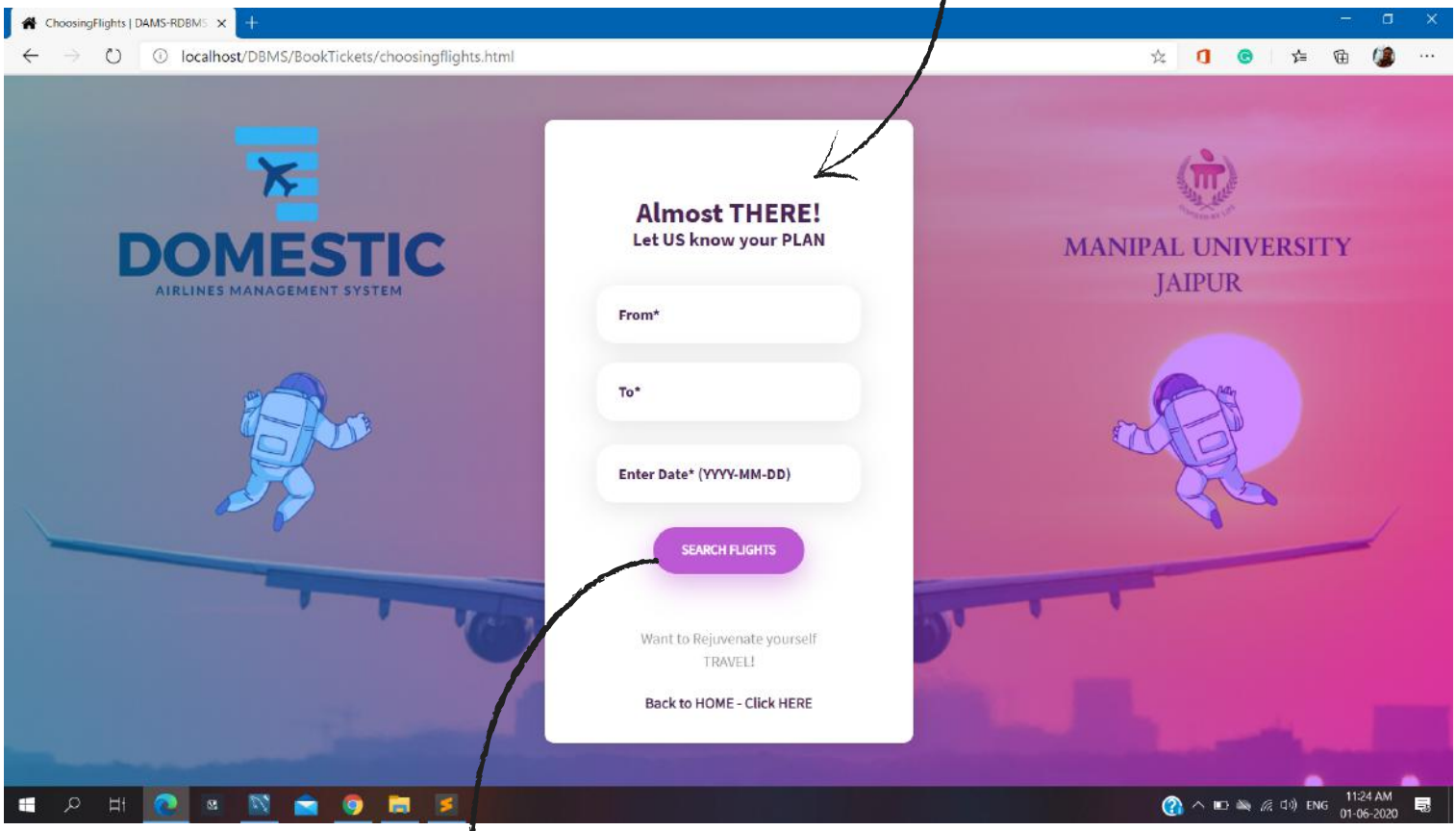
Pic : Snapshot from SQL Workbench

The query runs in the background, and finds that PassengerID= 101 and Name= Mayank Maheshwari exists, so it ***validates the customer and lands him into the flight booking system as shown in the next page.***

- SQL QUERIES ACCORDING TO GUI FUNCTIONALITIES

With LOGIN Section Queries: CUSTOMER

The sql query validates that, the customer has logged in with the correct credentials & opens the following page -



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

*The customer needs to enter the following details : **From** : Boarding City **To** : Destination City **Date :** Travelling Date*

*After a customer enters the details - clicking on the **Search Flights** button runs the following SQL query in the database: (The SQL queries are written as such to accept the values from the user input)*

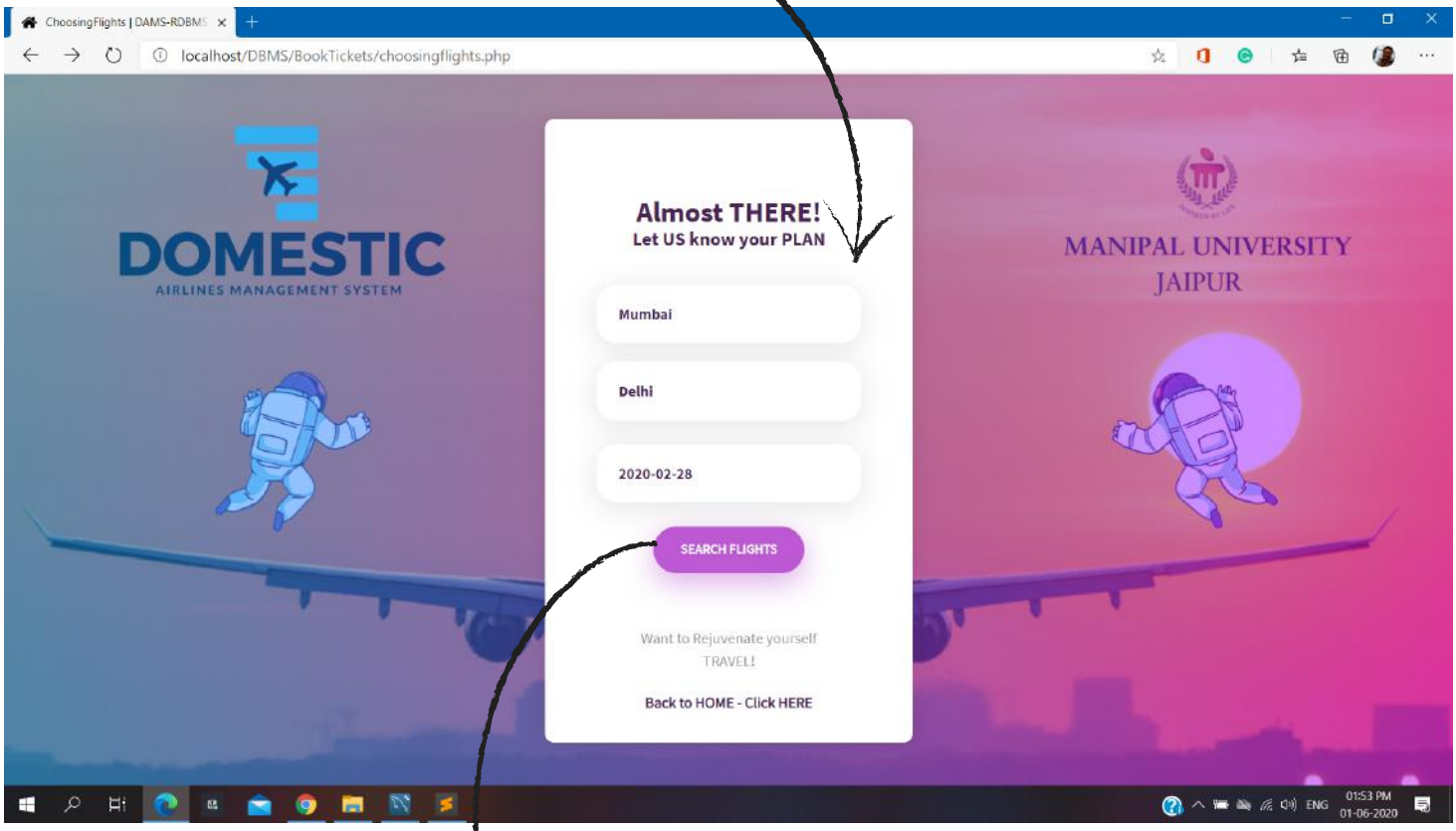
```
$stmt = $conn->prepare("select FlID, destination, airport, Fare, FlightDate, ArrivalTime, DepartureTime
from Route, Airfare, Flight_schedule where Route = RtID AND destination = '$to' AND airport = '$from'
AND flightdate = '$date';");
```

We will now see the actual working of the Customer Booking page, and then proceeding to the flight booking system. ***Note : Customer Login is necessary for booking flight tickets.***

- SQL QUERIES ACCORDING TO GUI FUNCTIONALITIES

With LOGIN Section Queries: CUSTOMER

*Now, as I have created the database - **Mumbai to Delhi on 2020-02-28 is a running flight schedule in the DAMS**. Let us enter these details in the form as shown below:*



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

*After the details **From = Mumbai , To = Delhi and Date = 2020-02-28** are entered, and then the customer click on the search flights button -> The following sql query runs in the sql:*

```
select FlID, destination, airport, Fare, FlightDate, ArrivalTime, DepartureTime from Route, Airfare, Flight_schedule
where Route = RtID AND destination = 'Delhi' AND airport = 'Mumbai' AND flightdate = '2020-02-28';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

FlID	destination	airport	Fare	FlightDate	ArrivalTime	DepartureTime
2	Delhi	Mumbai	2000	2020-02-28	02:30:00	23:25:00

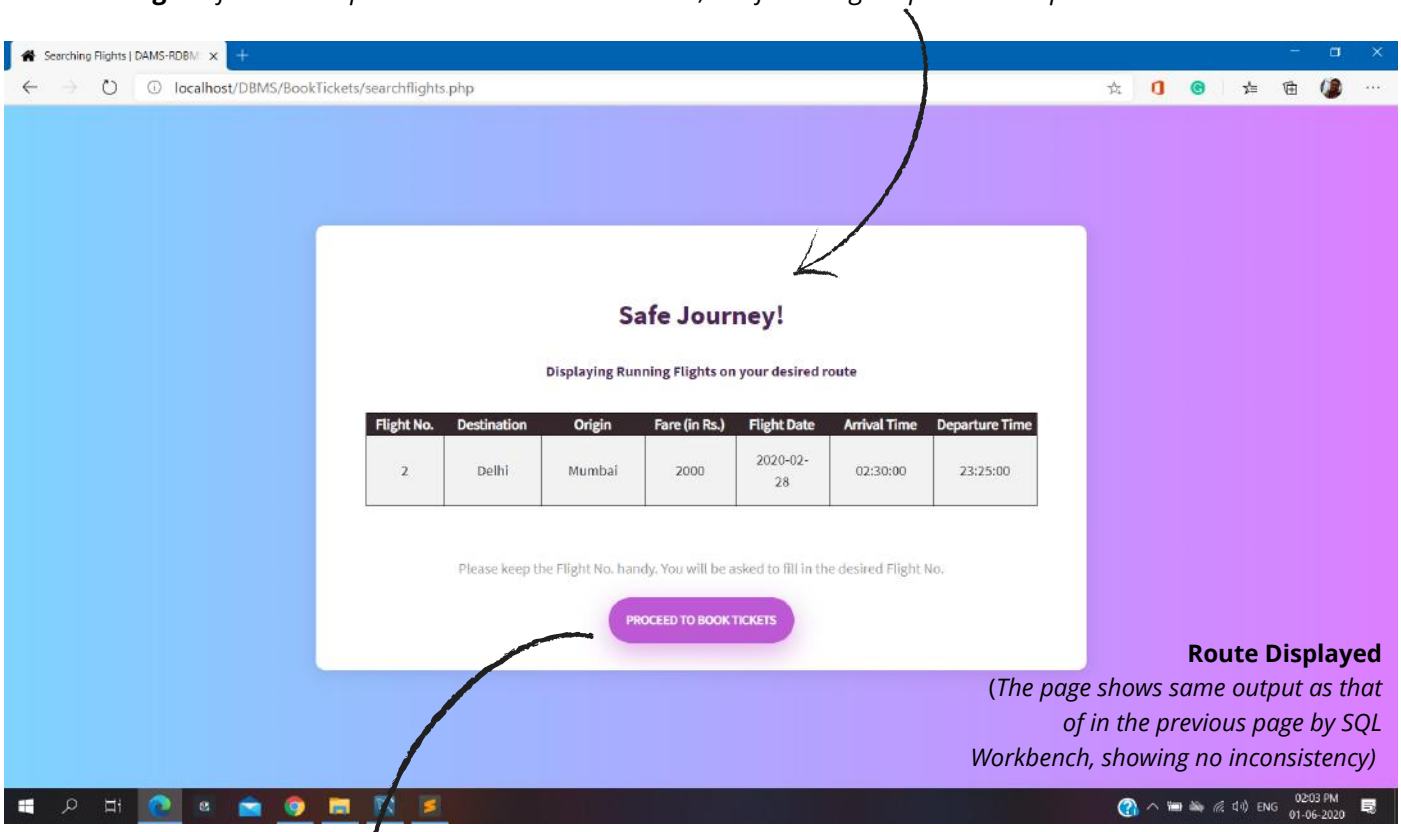
Pic : Snapshot from SQL Workbench

The query runs in the background, and finds that From = Mumbai , To = Delhi & Date = 2020-02-28 exists a route, so it ***shows the customer the details of the route on the next page.***

- SQL QUERIES ACCORDING TO GUI FUNCTIONALITIES

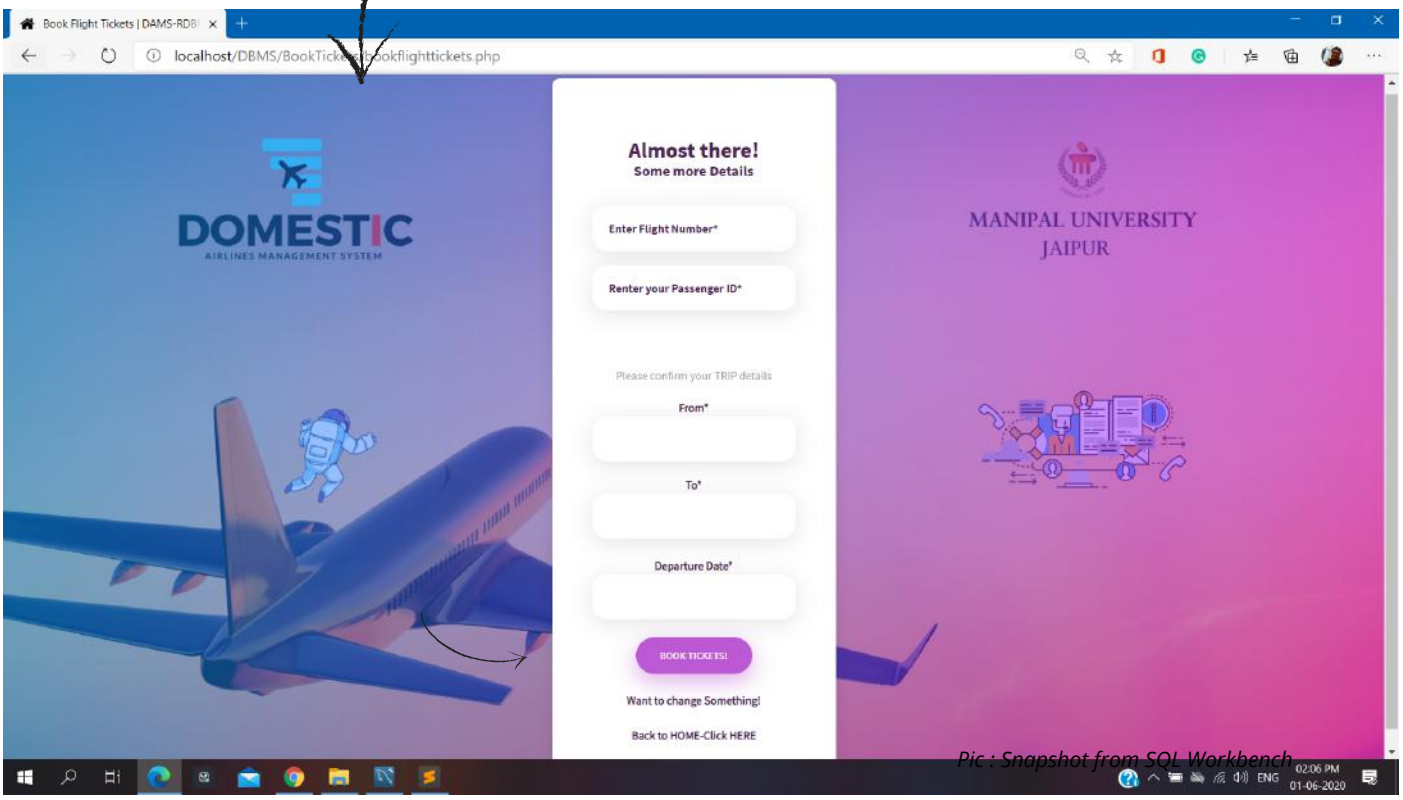
With LOGIN Section Queries: CUSTOMER

Route Found Page : *if a route is present on the entered details, the following output shows up*



Route Displayed
(The page shows same output as that of in the previous page by SQL Workbench, showing no inconsistency)

*Pic : Snapshot from actual **GUI representation** running at **localhost server***



Pic : Snapshot from SQL Workbench

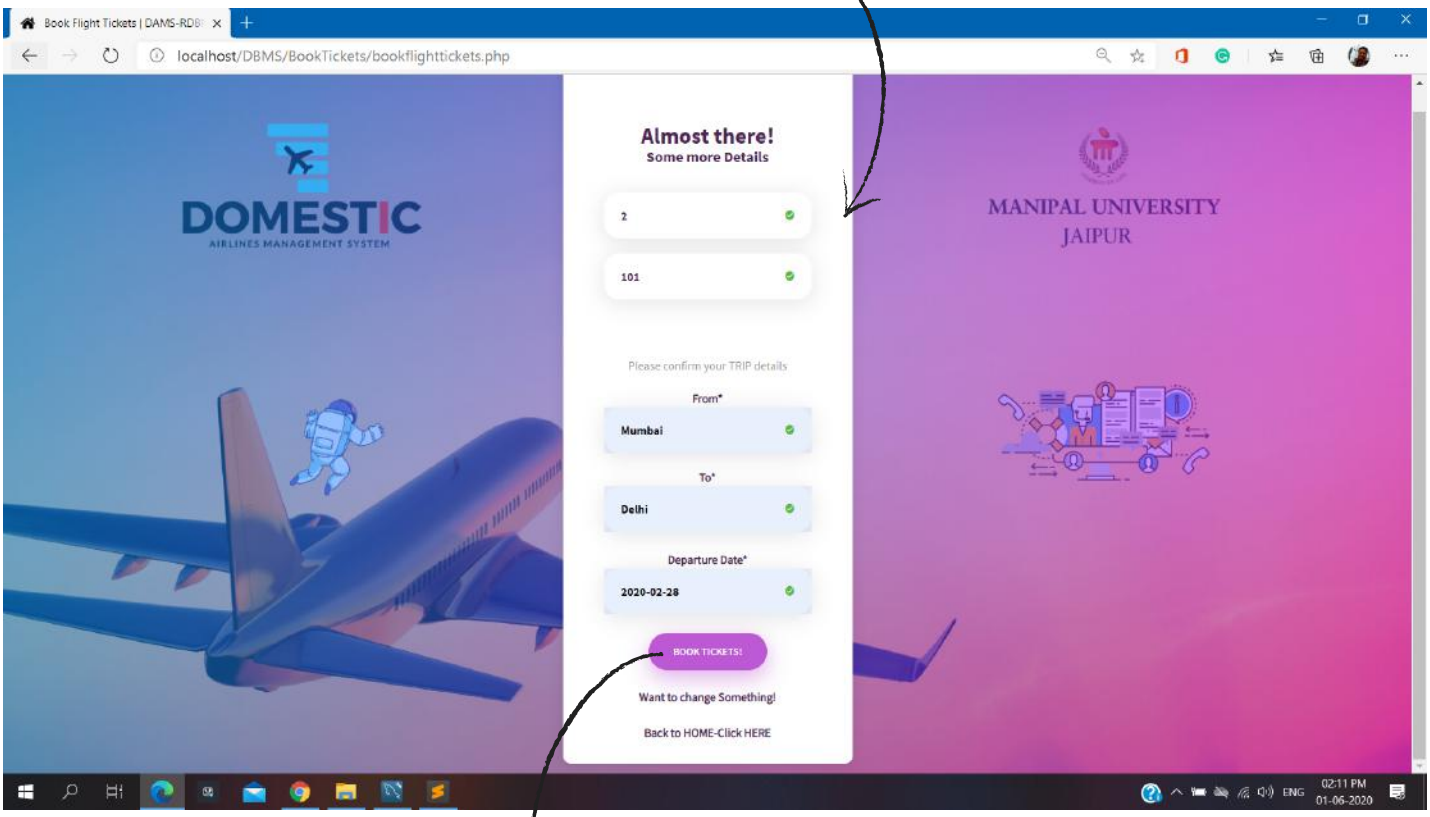
*Pic : Snapshot from actual **GUI representation** running at **localhost server***

The page demands, some additional details from the customer

- SQL QUERIES ACCORDING TO GUI FUNCTIONALITIES

With LOGIN Section Queries: CUSTOMER

The Customer enters the following details in the form as shown below :



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

*After the details **FlightID = 2, PassengerID = 101 , From = Mumbai , To = Delhi and Date = 2020-02-28** are entered, and then the customer click on the Book Ticktes button -> The following sql query runs in the sql:*

select Name, Age, FlID, destination, airport, Fare, FlightDate, ArrivalTime, DepartureTime from Route, Airfare, Flight_schedule, Passengers where Route = RtID AND destination = 'Mumbai' AND airport = 'Delhi' AND flightdate = '2020-01-23' AND PassengerID = '101';

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

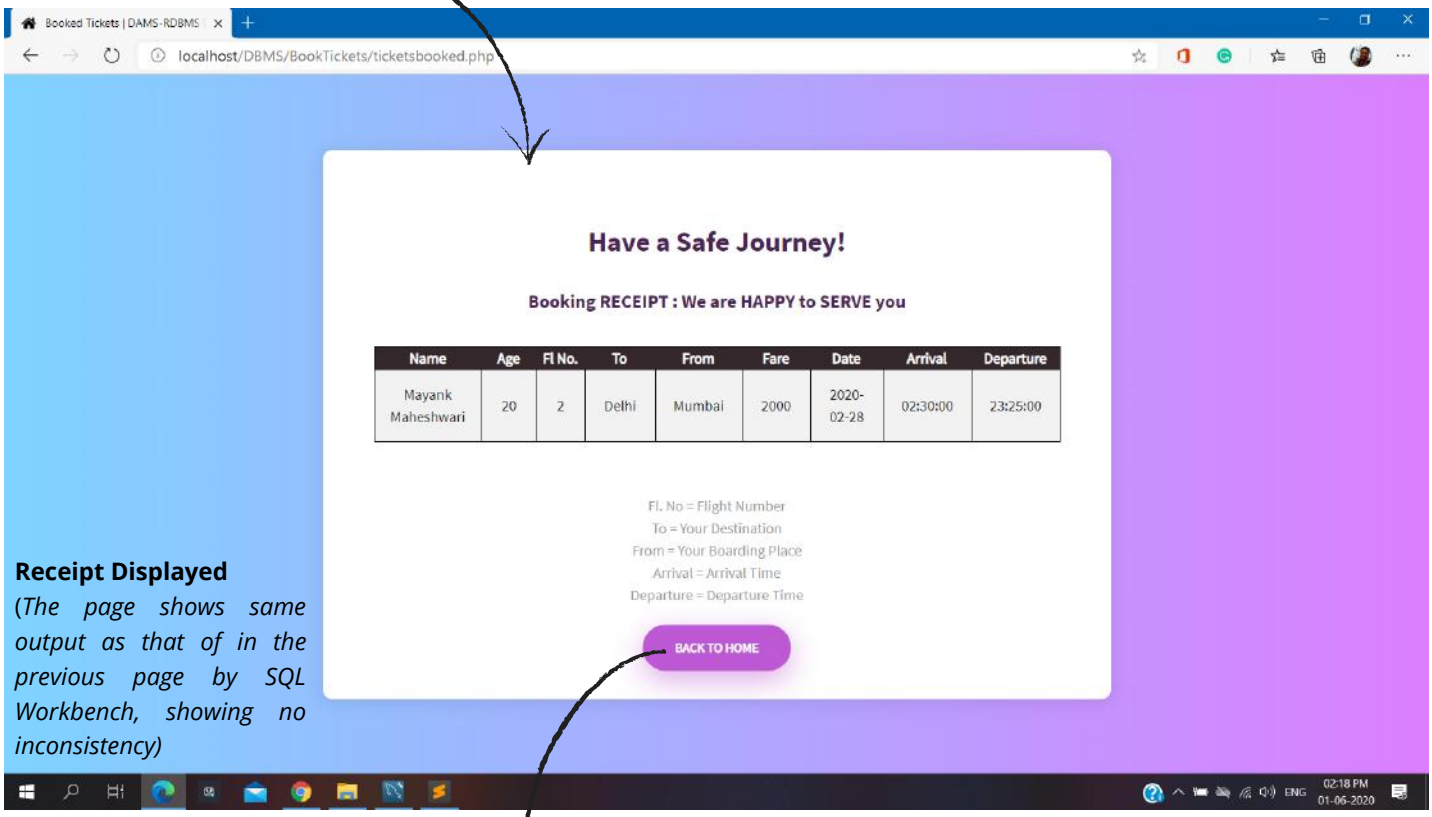
Name	Age	FlID	destination	airport	Fare	FlightDate	ArrivalTime	DepartureTime
Mayank Maheshwari	20	1	Mumbai	Delhi	2000	2020-01-23	01:20:00	23:20:00

Pic : Snapshot from SQL Workbench

The query runs in the background, and finds that From = Mumbai , To = Delhi & Date = 2020-02-28 exists a route and validates that PassengerID = 101 exists so it ***books the customer ticket and the details of the ticket are printed on the next page.***

With LOGIN Section Queries: CUSTOMER

Tickets Booked Successfully :

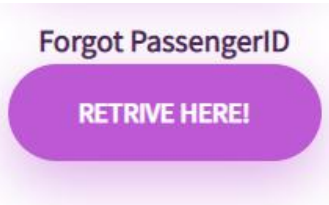


Receipt Displayed
(The page shows same output as that of in the previous page by SQL Workbench, showing no inconsistency)

*Pic : Snapshot from actual **GUI representation** running at **localhost server***

TICKETS BOOKED SUCCESSFULLY

Additional Features



*If a customer forgets his/her passenger ID, it can be retrieved by accessing the forgot PassengerID section. The customer needs to enter the following details : **FULL Name & Email ID entered at the time of SIGNUP.***

*After a customer enters the details - clicking on the **Retrieve** button runs the following SQL query in the database: (The SQL queries are written as such to accept the values from the user input)*

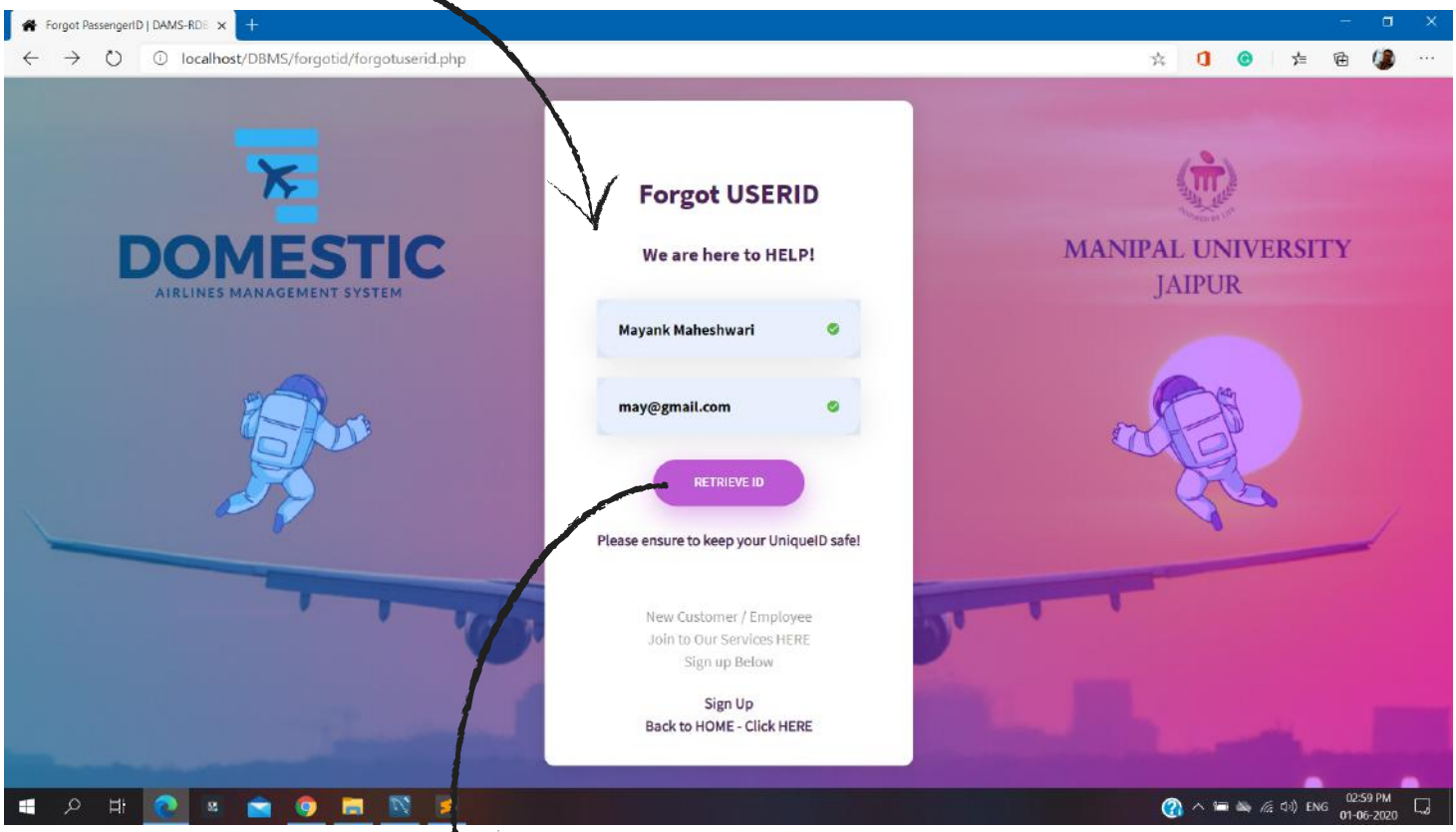
```
$stmt = $conn->prepare("select PassengerID, Name, Email from Passengers, Contact_details
where Name = '$Username' AND Email = '$emailID' ;");
```

let us see the working of this section in the next page ->

- SQL QUERIES ACCORDING TO GUI FUNCTIONALITIES

With LOGIN Section Queries: CUSTOMER

Forgot USERID form



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

*After a customer enters the details - clicking on the **Retrieve** button runs the following SQL query in the database:*

select PassengerID, Name, Email from Passengers, Contact_details where Name = 'Mayank Maheshwari' AND Email = 'may@gmail.com' ;

Result Grid | Filter Rows: | Expo

	PassengerID	Name	Email
▶	101	Mayank Maheshwari	may@gmail.com

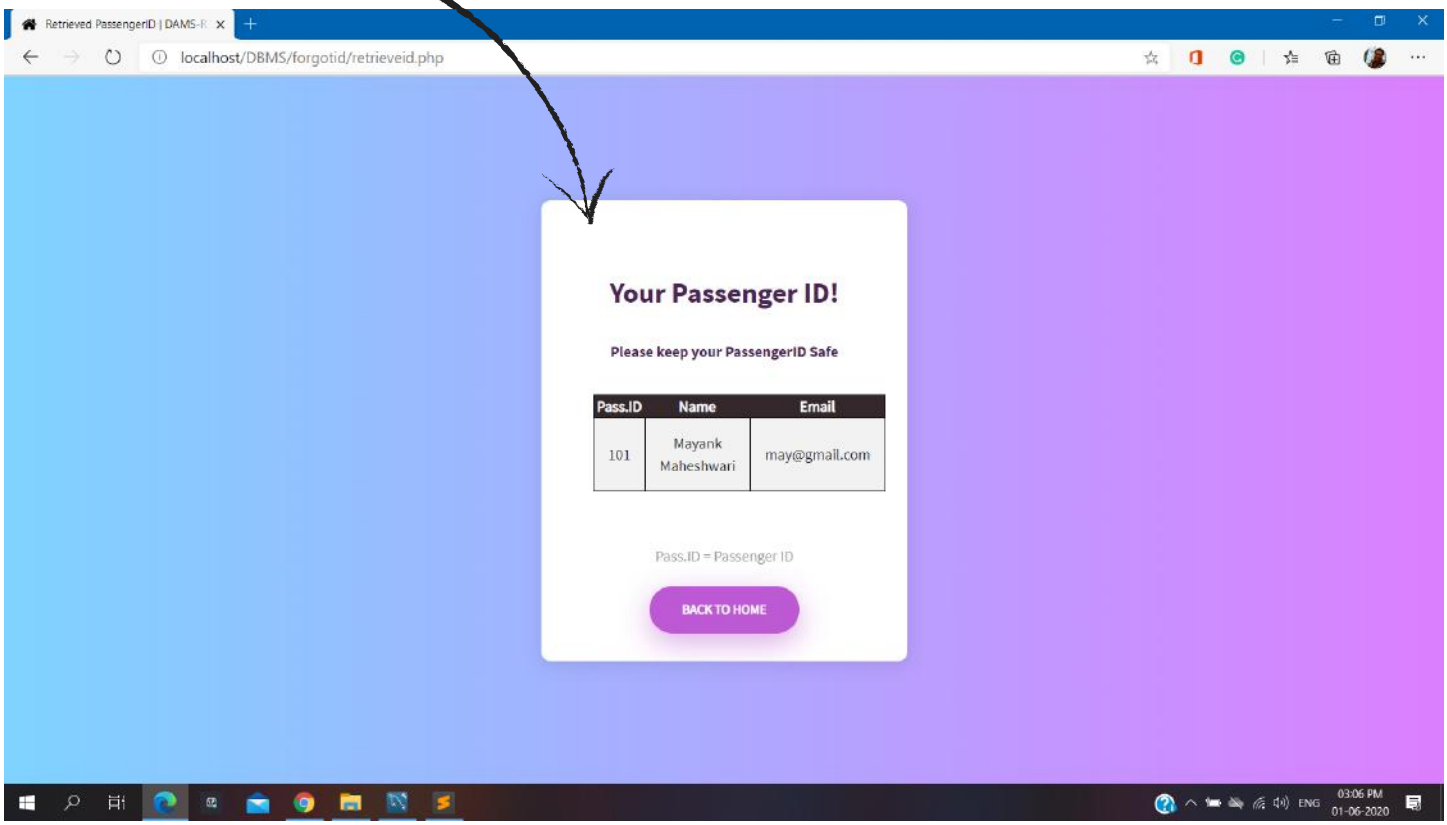
Pic : Snapshot from SQL Workbench

The query runs in the background, and finds that Name = Mayank Maheshwari , EmailID = may@gmail.com exists a Customer and validates the input.

The customer lands on to a page, which shows the details of the PassengerID, Name and EmailID ->

With LOGIN Section Queries: CUSTOMER

Retrieved USERID



*Pic : Snapshot from actual **GUI representation** running at **localhost server***

PassengerID Displayed

(The page shows same output as that of in the previous page by SQL Workbench, showing no inconsistency)

future SCOPE & Improvements for the DAMS PROJECT

This section discusses in detail, about how the project can be improved upon in the future :

1. ***Ticket Booking for more than one Passenger:*** *Right now, the project is being designed only for booking one ticket at a time. By making some changes in the ER Diagram - tickets of more than one passenger can be booked at the same time.*
2. ***Modification Access for Employees*** *: Our Employees only have the right to add or delete a record to the database. In the future time - Modification access can also be given to the employees.*
3. ***No Money paying link*** *: The pages can be designed for the users in the future, for payment options with more security.*
4. ***Checking availability of tickets*** *: The users are not able to check the no. of seats available in a flight before booking. It can be done accordingly by making changes in the ER diagram.*
5. ***Ticket Booking can be done only by Customers :*** *for now, the ticket booking rights are only given to the user. The same access rights should be given to the employees too.*
6. *A lot more services can be designed for the users in future with required improvements.*

• BIBLIOGRAPHY

Abraham Silberschatz , Henry F. Korth, S.Sudarshan
"Database System Concepts"

Johannes Gehrke and Raghu Ramakrishnan
"Database Management Systems"

Creatly - ER Diagram and Relational Database Creator
https://creately.com/diagram-type/template/gstx57ng1/er-diagram

MANIPAL UNIVERSITY
JAIPUR

Established under the Manipal University Jaipur Act (No. 21 of 2011)

DECLARATION

I hereby declare that the project entitled

Domestic Airlines Management System

submitted as part of the partial course requirements for the course

Relational Database And Management System

for the award of the degree of

Bachelor of Technology in Computer & Communication Engineering

at Manipal University Jaipur

during the IV Semester- 2020 , has been carried out by me. I declare that the project has not formed the basis for the award of any degree, associate ship, fellowship or any other similar titles elsewhere.

Further, I declare that I will not share, re-submit or publish the code, idea, framework and/or any publication that may arise out of this work for academic or profit purposes without obtaining the prior written consent of the Course Faculty Mentor and Course Instructor.

Mayank Jhanwar
RegNo. : 189303179
Email : mayank.189303179@muj.manipal.edu

Namit Varshney
RegNo. : 189302111
Email : namit.189302111@muj.manipal.edu

Month of Submission

JUNE 2020