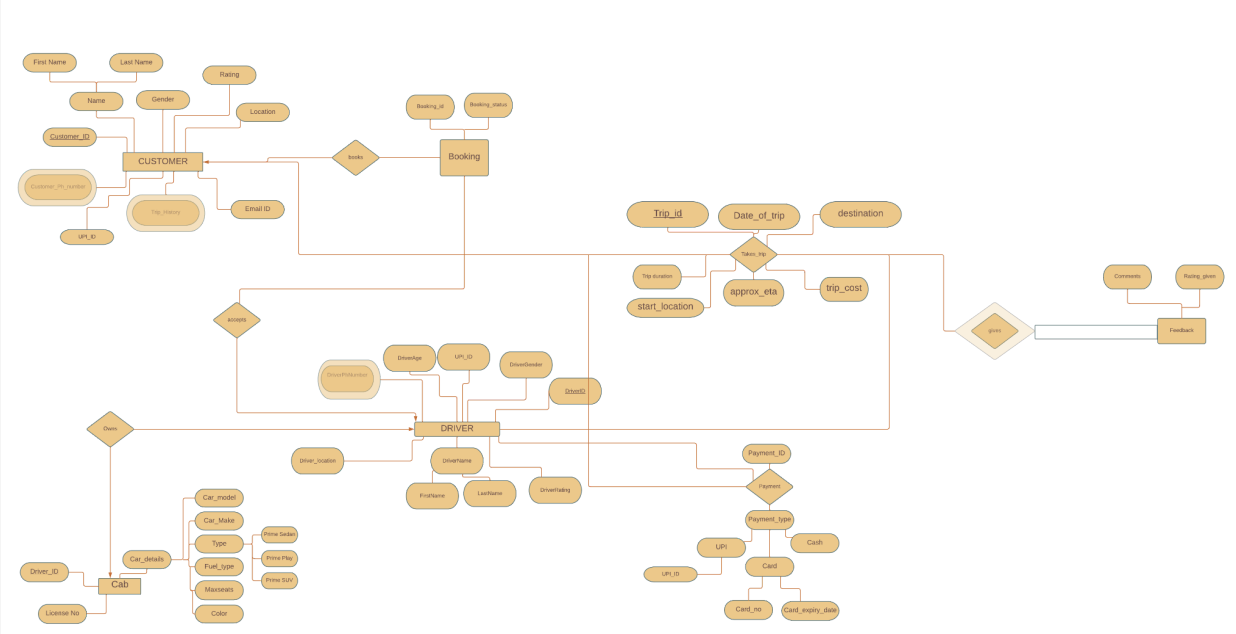
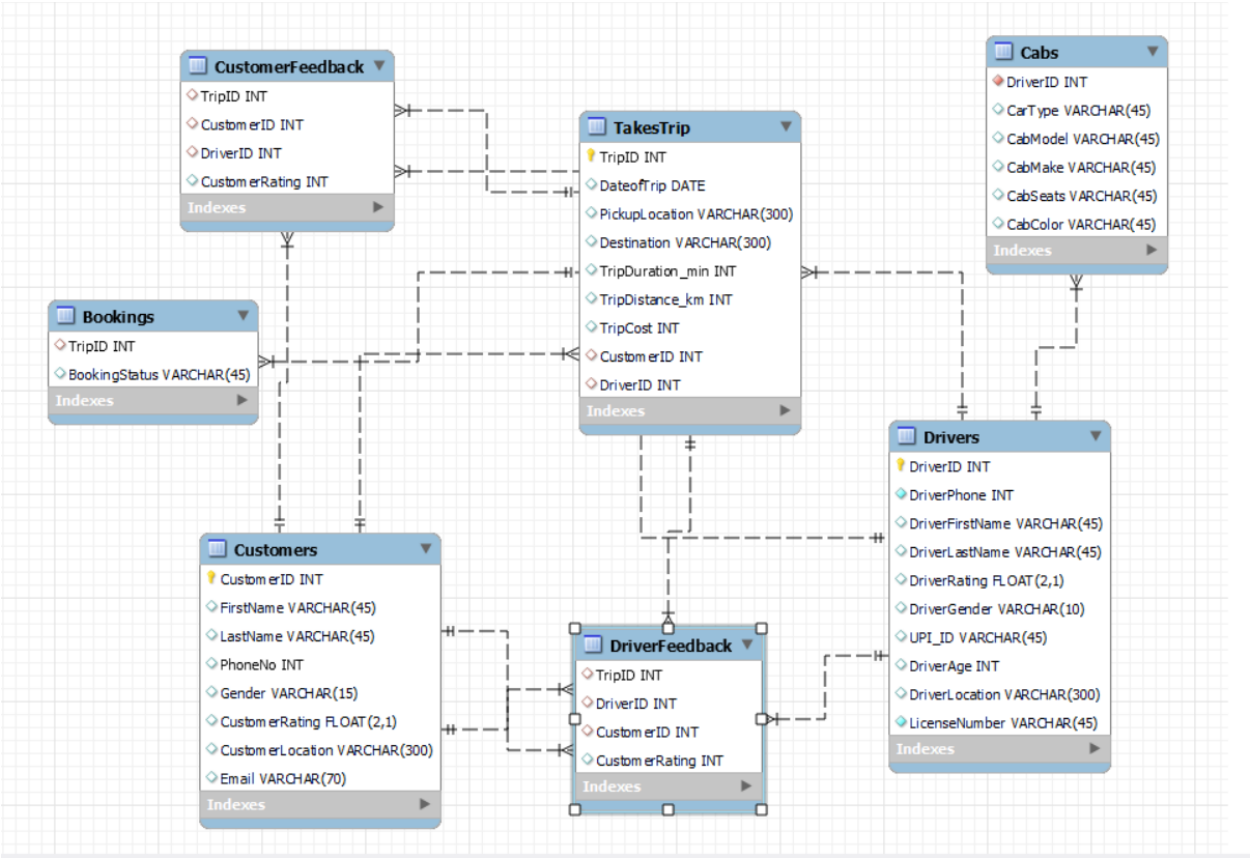
Final Project Evaluation

Sai Leela Rahul Pujari: 2020401
Saksham Lall : 2020402
Abhey Kalia : 2020420
Hardik Kumar : 2020506

ER Diagram



Relational Schema



Weak Entity: The weak entities are Driver and Customer Feedback because they can't exist without other entities.

Creating the Database:

```
CREATE DATABASE IF NOT EXISTS db;
use db;

CREATE TABLE IF NOT EXISTS DriverFeedback(
  TripID INT NULL DEFAULT NULL,
  DriverID INT NULL DEFAULT NULL,
  CustomerID INT NULL DEFAULT NULL,
  CustomerRating INT NULL DEFAULT NULL,
  INDEX TripID (TripID ASC) VISIBLE,
  INDEX DriverID (DriverID ASC) VISIBLE,
  INDEX CustomerID (CustomerID ASC) VISIBLE,
  CONSTRAINT driverfeedback_ibfk_1
    FOREIGN KEY (TripID)
```

```
    REFERENCES Cab_hailing_website.TakesTrip (TripID),
  CONSTRAINT driverfeedback_ibfk_2
    FOREIGN KEY (DriverID)
    REFERENCES Cab_hailing_website.Drivers (DriverID),
  CONSTRAINT driverfeedback_ibfk_3
    FOREIGN KEY (CustomerID)
    REFERENCES Cab_hailing_website.Customers (CustomerID),
  CONSTRAINT driverfeedback_ibfk_4
    FOREIGN KEY (CustomerID)
    REFERENCES Cab_hailing_website.Customers (CustomerID));


CREATE TABLE CustomerFeedback (
  TripID INT NULL DEFAULT NULL,
  CustomerID INT NULL DEFAULT NULL,
  DriverID INT NULL DEFAULT NULL,
  CustomerRating INT NULL DEFAULT NULL,
  INDEX DriverID (DriverID ASC) VISIBLE,
  INDEX CustomerID (CustomerID ASC) VISIBLE,
  INDEX TripID (TripID ASC) VISIBLE,
  CONSTRAINT customerfeedback_ibfk_1
    FOREIGN KEY (DriverID)
    REFERENCES Cab_hailing_website.Drivers (DriverID),
  CONSTRAINT customerfeedback_ibfk_2
    FOREIGN KEY (CustomerID)
    REFERENCES Cab_hailing_website.Customers (CustomerID),
  CONSTRAINT customerfeedback_ibfk_3
    FOREIGN KEY (TripID)
    REFERENCES Cab_hailing_website.TakesTrip (TripID));


CREATE TABLE IF NOT EXISTS Drivers (
  DriverID INT NOT NULL,
  DriverPhone INT NOT NULL,
  DriverFirstName VARCHAR(45) NULL DEFAULT NULL,
  DriverLastName VARCHAR(45) NULL DEFAULT NULL,
  DriverRating FLOAT NULL DEFAULT NULL,
  DriverGender VARCHAR(10) NULL DEFAULT NULL,
  UPI_ID VARCHAR(45) NULL DEFAULT NULL,
  DriverAge INT NULL DEFAULT NULL,
  DriverLocation VARCHAR(300) NULL DEFAULT NULL,
  LicenseNumber VARCHAR(45) NOT NULL,
  PRIMARY KEY (DriverID),
  UNIQUE INDEX DriverID_UNIQUE (DriverID ASC) VISIBLE,
```

```
  UNIQUE INDEX DriverPhone_UNIQUE (DriverPhone ASC) VISIBLE);

Select * from Drivers;

drop table Drivers;

CREATE TABLE IF NOT EXISTS Cabs (
  DriverID INT NOT NULL,
  CabType VARCHAR(45) NULL DEFAULT NULL,
  CabModel VARCHAR(45) NULL DEFAULT NULL,
  CabMake VARCHAR(45) NULL DEFAULT NULL,
  CabSeats INT NULL DEFAULT NULL,
  CabColor VARCHAR(45) NULL DEFAULT NULL,
  LicenseNumber VARCHAR(45) NOT NULL,
  FuelType VARCHAR(45) NULL DEFAULT NULL,
  PRIMARY KEY (DriverID),
  INDEX DriverID (DriverID ASC) VISIBLE,
  -- CONSTRAINT cabs_ibfk_1
  --   FOREIGN KEY (DriverID)
  --     REFERENCES Cab_hailing_website.Drivers (DriverID));
  CONSTRAINT cabs_ibfk_1 FOREIGN KEY (DriverID) REFERENCES Drivers(DriverID));

Select * from Cabs;

drop table Cabs;

CREATE TABLE IF NOT EXISTS Customers (
  CustomerID INT NOT NULL AUTO_INCREMENT,
  FirstName VARCHAR(45) NULL DEFAULT NULL,
  LastName VARCHAR(45) NULL DEFAULT NULL,
  PhoneNo INT NULL DEFAULT NULL,
  Gender VARCHAR(15) NULL DEFAULT NULL,
  CustomerRating FLOAT NULL DEFAULT NULL,
  CustomerLocation VARCHAR(300) NULL DEFAULT NULL,
  Email VARCHAR(70) NULL DEFAULT NULL,
  Address VARCHAR(70) NULL DEFAULT NULL,
  UPI_ID VARCHAR(45) NULL DEFAULT NULL,
  PRIMARY KEY (CustomerID),
  UNIQUE INDEX CustomerID_UNIQUE (CustomerID ASC) VISIBLE,
  UNIQUE INDEX PhoneNo_UNIQUE (PhoneNo ASC) VISIBLE);

Select * from Customers;

drop table Customers;
```

```
CREATE TABLE IF NOT EXISTS Trip (
  TripID INT NOT NULL,
  TripDate datetime NULL DEFAULT NULL,
  Duration INT NULL DEFAULT NULL,
  ETA TIME NULL DEFAULT NULL,
  Cost FLOAT NULL DEFAULT NULL,
  Destination VARCHAR(300) NULL DEFAULT NULL,
  StartLocation VARCHAR(300) NULL DEFAULT NULL,
  CustomerID INT NULL DEFAULT NULL,
  DriverID INT NULL DEFAULT NULL,
  INDEX TripID (TripID ASC) VISIBLE,
  PRIMARY KEY (TripID),
  CONSTRAINT bookings_ibfk_1 FOREIGN KEY (TripID) REFERENCES Trip(TripID));
```

SQL QUERIES

```
SELECT DriverID, DriverFirstName, DriverLastName, Rating FROM Drivers ORDER BY Rating DESC LIMIT 5;

SELECT Drivers.DriverID, SUM(Trips.Distance*10+50) FROM Trips join Drivers on Trips.DriverID=Drivers.DriverID where Drivers.DriverID=6;

SELECT DriverID, DriverFirstName, DriverLastName, Rating FROM Drivers WHERE Rating BETWEEN 4 AND 4.5;

SELECT TripDate, Trips.CustID, COUNT(Trips.CustID) FROM Trips, Customers WHERE Trips.CustID = Customers.CustID AND TripDate BETWEEN "2022-04-19 00:00:00" AND "2022-04-29 21:59:59";

SELECT Rating, DriverID, IF(Rating<4, "Decent Driver", "Very Good Driver") FROM Drivers;

SELECT AVG(Rating) FROM Drivers WHERE DriverLocation = 'Delhi';

-- 1. Search cab w 4 seats and is Porsche
SELECT DriverID, CabMake, CabSeats from Cabs where CabMake = "Porsche" and CabSeats = 4;

-- 2. Find name of driver in previous query
```

```
SELECT drivers.DriverID, CabMake, DriverFirstName, DriverLastName, CabSeats from Cabs, Drivers where Cabs.DriverID = Drivers.DriverID and CabMake = "Porsche" and CabSeats = 4;

-- 3. Find driver whose rating is between 4 and 4.5
SELECT DriverID, DriverFirstName, DriverLastName, DriverRating  FROM drivers WHERE DriverRating BETWEEN 4 AND 4.5;

-- 4. Find all trips made on date X
SELECT TripDate, Trip.CustomerID, COUNT(Trip.CustomerID) FROM Trip, Customers WHERE Trip.CustomerID = Customers.CustomerID AND TripDate BETWEEN "2020-04-19 00:00:00" AND "2020-04-19 21:59:59";

-- 5. check average rating of customers from specific location
SELECT AVG(CustomerRating) FROM Customers WHERE CustomerLocation = 'Oak Valley';

-- to check how many people in Oak Valley, use
SELECT * FROM customers WHERE CustomerLocation = 'Oak Valley';

-- 6. Check rating of drivers and check if there are good drivers or not in a specific location
SELECT DriverRating, DriverID,  IF(DriverRating<4, "Decent Driver", "Very Good Driver") FROM Drivers;

-- 7. find all drivers where customer location and driver location is same
SELECT * FROM Drivers JOIN Customers ON DriverLocation = 'Gina' AND CustomerLocation='Gina';

-- 8. check average rating of drivers from specific location
SELECT AVG(DriverRating), AVG(CustomerRating) FROM Drivers, Customers WHERE DriverLocation = 'Gina' AND CustomerLocation = 'Gina';

-- 9. Query to fetch last record from table
SELECT driverID, FROM drivers, customers where CustomerID select max(Rowid) from Employee;

-- 10. Check cost of a certain trip for a specific driverf
SELECT DriverID, FirstName, LastName from Drivers where DriverID IN (SELECT DriverID, Cost from Trip where DriverID = 163 or DriverID = 822
```

VIEWS AND GRANTS

This is grant for root user

REVOKE ALL PRIVILEGES ON . FROM 'root'@'localhost'; GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, INDEX, ALTER, CREATE TEMPORARY TABLES, CREATE VIEW, TRIGGER, SHOW VIEW ON . TO 'root'@'localhost' REQUIRE NONE WITH GRANT OPTION MAX_QUERIES_PER_HOUR 0 MAX_CONNECTIONS_PER_HOUR 0 MAX_UPDATES_PER_HOUR 0 MAX_USER_CONNECTIONS 0;