Share it! Meet your new commute

Relational Database Management System

Instructor: Dr. Kutsal Dogan

Group 2 Members:

1. **Shweta Singh** (*sxs172332@utdallas.edu*)

2. **Adit Kansara** (*ask172030@utdallas.edu*)

3. **Duy Vu** (*dhv170030@utdallas.edu*)

4. **Anurag Kanchibhotla Subhramanya** (*axk174830@utdallas.edu*)

Table of Contents

I. COMPANY BACKGROUND

ShareIt is an American company established in 2013 and headquartered in Dallas, Texas. It allows users to register for sharing the rides at very short notice. According to ThinkProgress, 80% of the Americans drive alone to the work daily which leads to traffic, pollution and lots of wastage of gas and money (Garafalo). ShareIt provides a platform to such users to share their ride with another person who is traveling on the same route. Apart from intra-city rides, additionally it also provides a feature of sharing inter-city/inter-state rides in case of long trips.

II. CURRENT DATABASE ISSUES

1. Time consuming and probability of human error is more.
2. Involved tedious calculations for profit and other related entries.
3. Need for a dedicated employee for sending trip details to customers and owners.
4. Large amount of data maintenance is needed.
5. Record keeping was using Excel Spreadsheets which lead to incorrect and inconsistent information that increases customer wait time

III. PROPOSED SOLUTION

As the company grew, there was a need to maintain a large amount of data for which we proposed to build a simple database which would make the life of employees easier. Rather than using excel spreadsheets, migrating to this database will help in storing a large amount of data as well as makes the calculation of price and miles traveled easier. It is easy to operate, and employees can generate the reports of monthly transactions and profit. The calculations will be accurate and error-free which will make the processing faster. The layout and the details about the database if explained further in this document.

IV. SCOPE OF DATABASE

Currently, ShareIt uses excel spreadsheets to store the data and calculation which results in a lot of errors and customer dissatisfaction due to long wait times. The new database will remove the tedious calculation and data maintenance process. There are total 8 tables in this database whose description is as follows:

1. Customer table:

This table contains all the personal information of all the customers, whether he is a car owner, employee or a customer. Everyone is identified by unique id, customerID. It contains details like the customer first name, last name, email id, address, contact number and their personal preferences.

2. Owner:

This table is used to store additional information about the car owner such as SSN, license number. Each record is identified by OwnerID. This table has foreign key OwnerID that links to table Customer.

3. Owner_Ride_Offer:

This table will store the details of the ride created by the ride owners. It includes the details about the car, source location and destination location, ride start and end time, and the cost per head details. Each record will have an unique RideID. This table has foreign key OwnerID that links to table Owner, and license_plate that links to license_plate of table Car.

4. Booking:

This table stores booking information such as driver and passenger IDs, pick up and drop off locations. Each record is identified by bookingID. This table has foreign key RideID that links to table Owner_Ride_Offer, and Rider_ID that links to table Customer.

5. Car:

The ride owner needs to add details for at least one car before creating a ride. This table will store the details of cars registered by the owners and car features. The attributes of this table are license plate, make, model, manufacturer year, and car options such as hybrid, sport, 4-wheel drive, stereo, abs. Each record is identified by license plate. This table has foreign key Owner_ID that links to table Owner.

6. Review:

After completing a successful ride, the owner/customer can give their feedback on their ride offered or shared, give the owner star rating so the company can keep track of the behavior of registered car owners. Each record is identified by ReviewID. This table has foreign key transaction_ID that links to table Transaction.

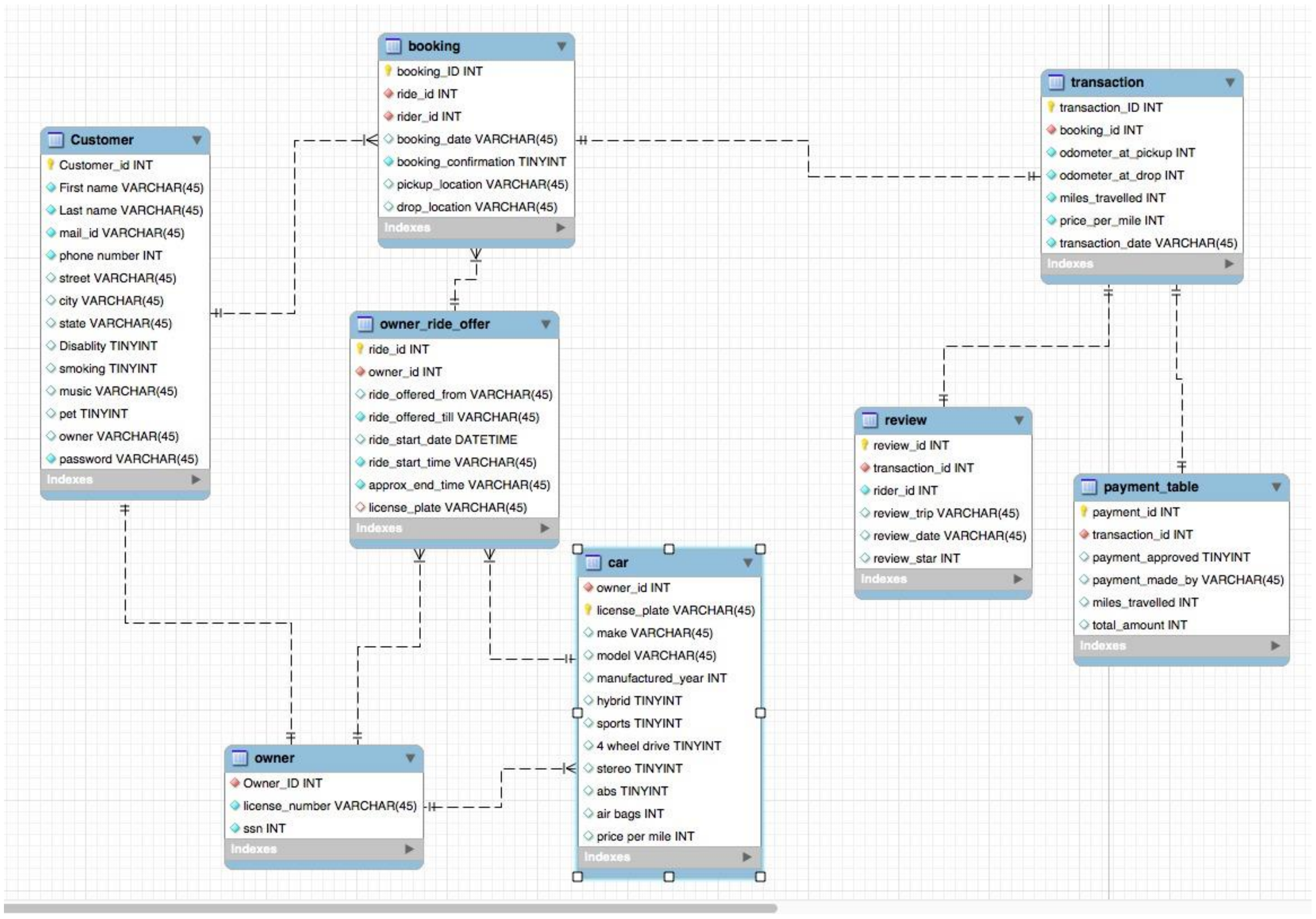
7. Payment_table:

This table is used for generating invoices, total price for the customer ride and payment method. Each record is identified by PaymentID . This table has foreign key transaction_ID that links to table Transaction.

8. Transaction:

This stores the information of the customer start-point, end-point and miles travelled. Each record is identified by TransactionID. This table has foreign key booking_ID that links to table Booking.

V. E-R DIAGRAM


booking
booking_ID INT
ride_id INT
rider_id INT
booking_date VARCHAR(45)
booking_confirmation TINYINT
pickup_location VARCHAR(45)
drop_location VARCHAR(45)
Indexes
transaction
transaction_ID INT
booking_id INT
odometer_at_pickup INT
odometer_at_drop INT
miles_travelled INT
price_per_mile INT
transaction_date VARCHAR(45)
Indexes
Customer
Customer_id INT
First name VARCHAR(45)
Last name VARCHAR(45)
mail_id VARCHAR(45)
phone number INT
street VARCHAR(45)
city VARCHAR(45)
state VARCHAR(45)
Disablity TINYINT
smoking TINYINT
music VARCHAR(45)
pet TINYINT
owner VARCHAR(45)
password VARCHAR(45)
Indexes
owner_ride_offer
ride_id INT
owner_id INT
ride_offered_from VARCHAR(45)
ride_offered_till VARCHAR(45)
ride_start_date DATETIME
ride_start_time VARCHAR(45)
approx_end_time VARCHAR(45)
license_plate VARCHAR(45)
Indexes
review
review_id INT
transaction_id INT
rider_id INT
review_trip VARCHAR(45)
review_date VARCHAR(45)
review_star INT
Indexes
payment_table
payment_id INT
transaction_id INT
payment_approved TINYINT
payment_made_by VARCHAR(45)
miles_travelled INT
total_amount INT
Indexes
car
owner_id INT
license_plate VARCHAR(45)
make VARCHAR(45)
model VARCHAR(45)
manufactured_year INT
hybrid TINYINT
sports TINYINT
4 wheel drive TINYINT
stereo TINYINT
abs TINYINT
air bags INT
price per mile INT
Indexes
owner
Owner_ID INT
license_number VARCHAR(45)
ssn INT
Indexes

VI. RELATIONAL DATABASE SCHEMA

Table: Customer

```
CREATE TABLE `Customer` (
  `Customer_id` INT NOT NULL,
  `First name` VARCHAR(45) NOT NULL,
  `Last name` VARCHAR(45) NOT NULL,
  `mail_id` VARCHAR(20) NOT NULL,
  `phone number` INT NOT NULL,
  `street` VARCHAR(45) NULL,
  `city` VARCHAR(15) NULL,
  `state` VARCHAR(15) NULL,
  `Disablity` TINYINT NULL,
  `smoking` TINYINT NULL,
  `music` VARCHAR(45) NULL,
  `pet` TINYINT NULL,
  `owner` VARCHAR(5) NULL,
  `password` VARCHAR(15) NOT NULL,
  PRIMARY KEY (`Customer_id`)
 ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

Table: Owner

```
CREATE TABLE `owner` (
  `Owner_ID` INT NOT NULL,
  `license_number` VARCHAR(45) NOT NULL,
  `ssn` INT NOT NULL,
  INDEX `owner_id_ot_idx` (`Owner_ID` ASC),
  CONSTRAINT `owner_id_ot`
    FOREIGN KEY (`Owner_ID`)
    REFERENCES `mydb`.`Customer` (`Customer_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
```

Table: car

```
CREATE TABLE `car` (
  `owner_id` INT NOT NULL,
  `license_plate` VARCHAR(45) NOT NULL,
  `make` VARCHAR(10) NULL,
  `model` VARCHAR(10) NULL,
  `manufactured_year` INT NULL,
  `hybrid` TINYINT NULL,
  `sports` TINYINT NULL,
  `4 wheel drive` TINYINT NULL,
  `stereo` TINYINT NULL,
  `abs` TINYINT NULL,
  `air bags` INT NULL,
  `price per mile` INT NULL,
  PRIMARY KEY (`license_plate`),
  INDEX `car_owner_id_ct_idx` (`owner_id` ASC),
  CONSTRAINT `car_owner_id_ct`
    FOREIGN KEY (`owner_id`)
    REFERENCES `mydb`.`owner` (`Owner_ID`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
```

Table: owner_ride_offer

```
CREATE TABLE `owner_ride_offer` (
 `ride_id` INT NOT NULL,
 `owner_id` INT NOT NULL,
 `ride_offered_from` VARCHAR(45) NULL,
 `ride_offered_till` VARCHAR(45) NOT NULL,
 `ride_start_date` DATETIME NULL,
 `ride_start_time` VARCHAR(10) NOT NULL,
 `approx_end_time` VARCHAR(10) NOT NULL,
 `license_plate` VARCHAR(10) NULL,
 PRIMARY KEY (`ride_id`),
 INDEX `owner_id_orft_idx` (`owner_id` ASC),
 INDEX `license_plate_orft_idx` (`license_plate` ASC),
 CONSTRAINT `owner_id_orft`
  FOREIGN KEY (`owner_id`)
  REFERENCES `mydb`.`owner` (`Owner_ID`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION,
 CONSTRAINT `license_plate_orft`
  FOREIGN KEY (`license_plate`)
  REFERENCES `mydb`.`car` (`license_plate`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION)
```

Table: booking

```
CREATE TABLE `booking` (
  `booking_ID` INT NOT NULL,
  `ride_id` INT NOT NULL,
  `rider_id` INT NOT NULL,
  `booking_date` VARCHAR(10) NULL,
  `booking_confirmation` TINYINT NOT NULL,
  `pickup_location` VARCHAR(15) NULL,
  `drop_location` VARCHAR(15) NULL,
  PRIMARY KEY (`booking_ID`),
  INDEX `ride_id_bt_idx` (`ride_id` ASC),
  INDEX `rider_id_btt_idx` (`rider_id` ASC),
  CONSTRAINT `ride_id_bt`
    FOREIGN KEY (`ride_id`)
    REFERENCES `mydb`.`owner_ride_offer` (`ride_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `rider_id_btt`
    FOREIGN KEY (`rider_id`)
    REFERENCES `mydb`.`Customer` (`Customer_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

Table: transaction

```
CREATE TABLE `transaction` (
 `transaction_ID` INT NOT NULL,
 `booking_id` INT NOT NULL,
 `odometer_at_pickup` INT NOT NULL,
 `odometer_at_drop` INT NOT NULL,
 `miles_travelled` INT NOT NULL,
 `price_per_mile` INT NOT NULL,
 `transaction_date` VARCHAR(10) NOT NULL,
 PRIMARY KEY (`transaction_ID`),
 INDEX `booking_id_tt_idx` (`booking_id` ASC),
 CONSTRAINT `booking_id_tt`
  FOREIGN KEY (`booking_id`)
  REFERENCES `mydb`.`booking` (`booking_ID`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION)
```

Table: review

```
CREATE TABLE `review` (
 `review_id` INT NOT NULL,
 `transaction_id` INT NOT NULL,
 `rider_id` INT NOT NULL,
 `review_trip` VARCHAR(300) NULL,
 `review_date` VARCHAR(10) NULL,
 `review_star` INT NULL,
 PRIMARY KEY (`review_id`),
 INDEX `transaction_id_rt_idx` (`transaction_id` ASC),
 CONSTRAINT `transaction_id_rt`
  FOREIGN KEY (`transaction_id`)
  REFERENCES `mydb`.`transaction` (`transaction_ID`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION)
```

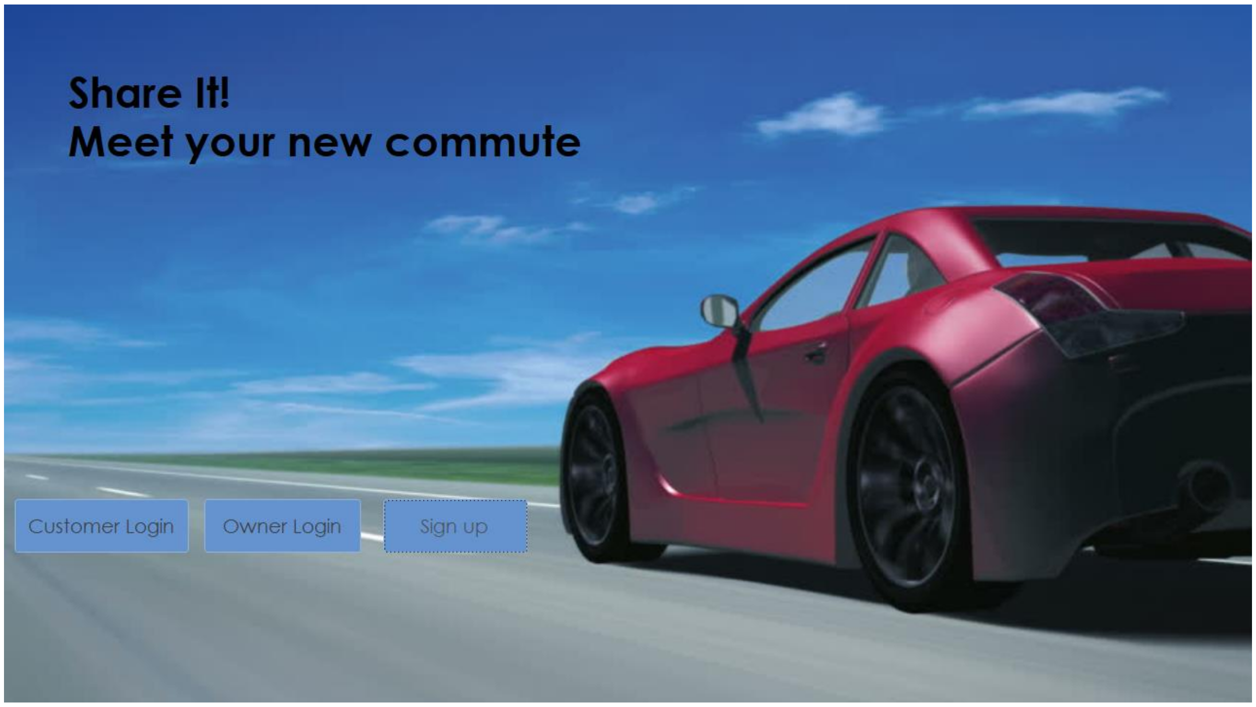
Table: payment_table

```
CREATE TABLE `payment_table` (
  `payment_id` INT NOT NULL,
  `transaction_id` INT NOT NULL,
  `payment_approved` TINYINT NULL,
  `payment_made_by` VARCHAR(20) NULL,
  `miles_travelled` INT NULL,
  `total_amount` INT NULL,
  PRIMARY KEY (`payment_id`),
  INDEX `transaction_id_pt_idx` (`transaction_id` ASC),
  CONSTRAINT `transaction_id_pt`
    FOREIGN KEY (`transaction_id`)
    REFERENCES `mydb`.`transaction` (`transaction_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

VI. MENU AND DATA INPUT SCREENS

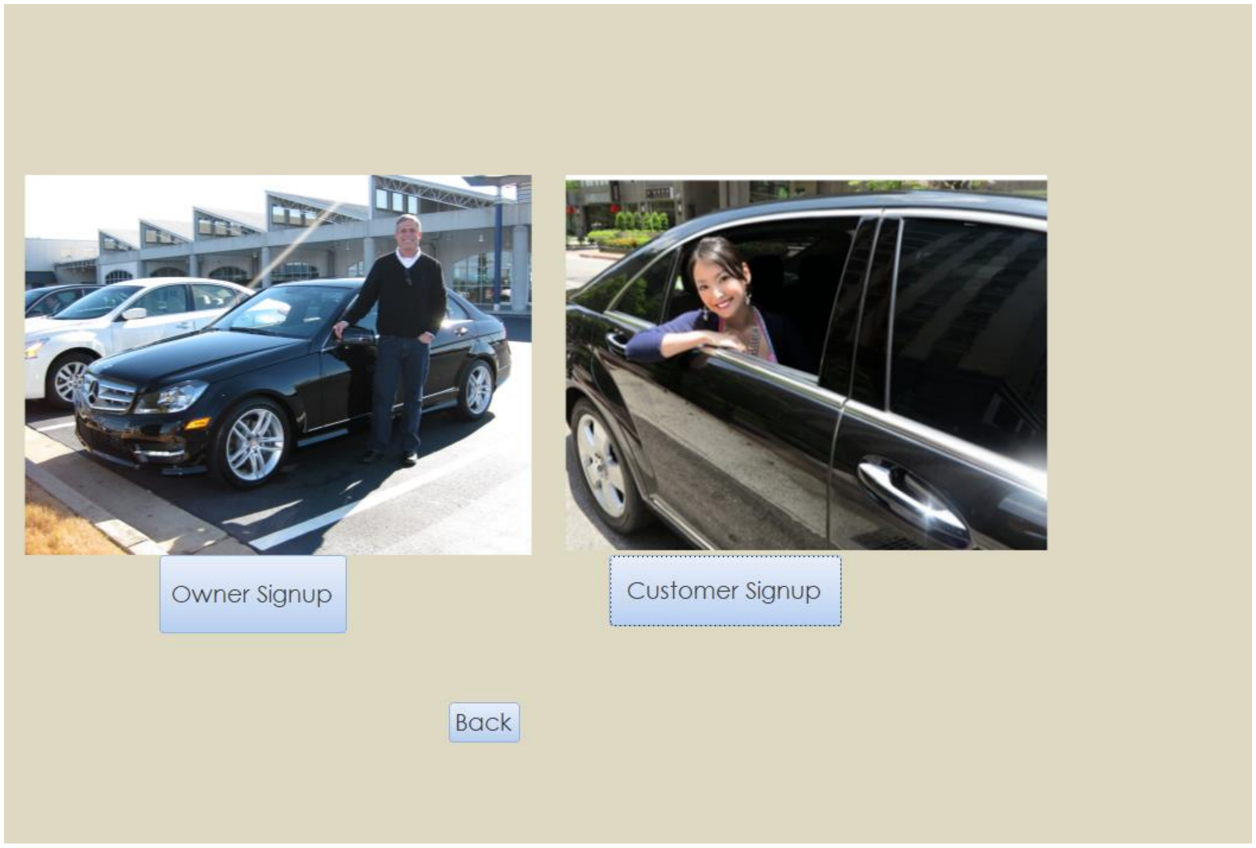
1. Dashboard

The Dashboard is the main menu of the ShareIt which allows the users and customers to register themselves. The signup button allows the user to register and after successful registration the owner login and customer login buttons allows them to enter the website.

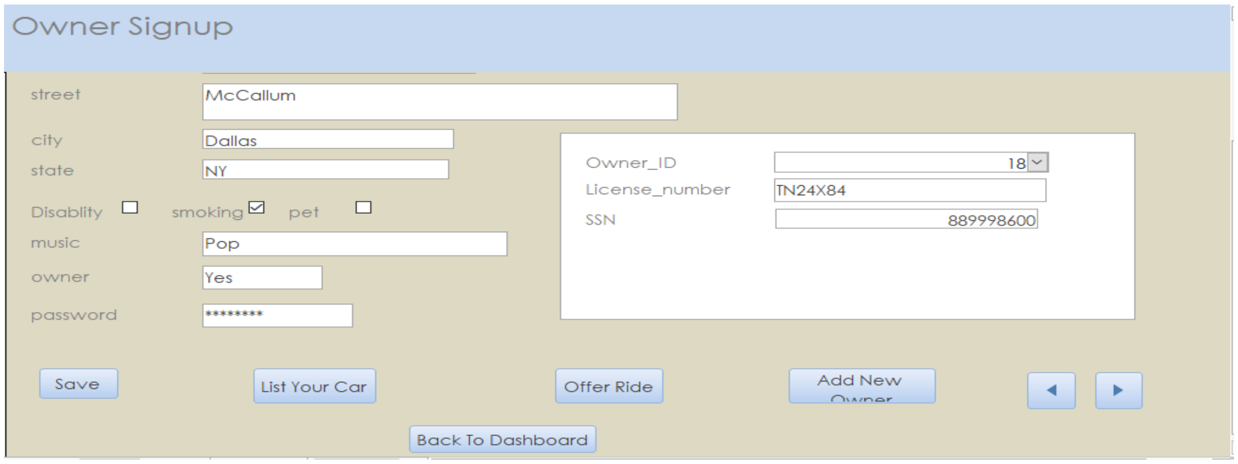


2. Signup

The signup page has separate signups for the customer and owner. Customer and owner requires user to fill data such as first name, last name, email, contact, address and personal preferences.



For owner, besides basic information, he/she needs to register the car and the SSN for company to keep track of.

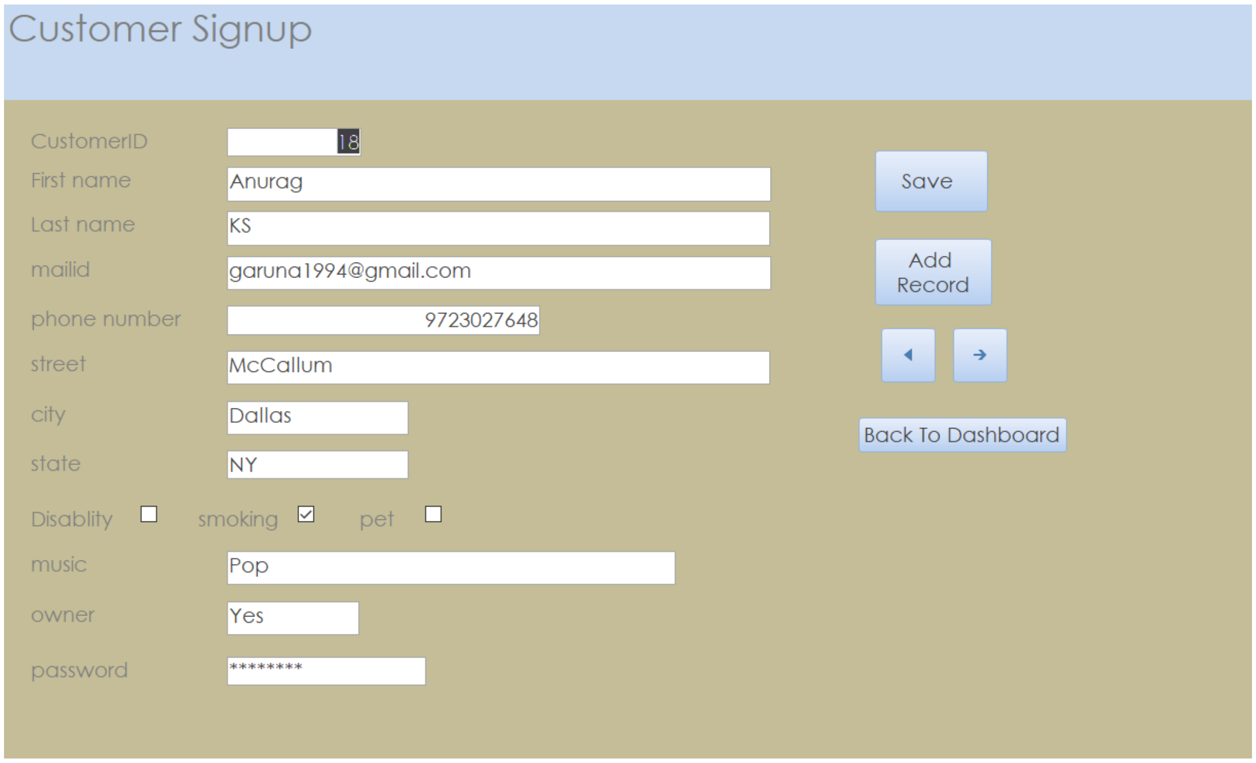


Owner Signup

street	McCallum
city	Dallas
state	NY
Disablity	☐
smoking	☑
pet	☐
music	Pop
owner	Yes
password	********

Owner_ID	18
License_number	TN24X84
SSN	889998600

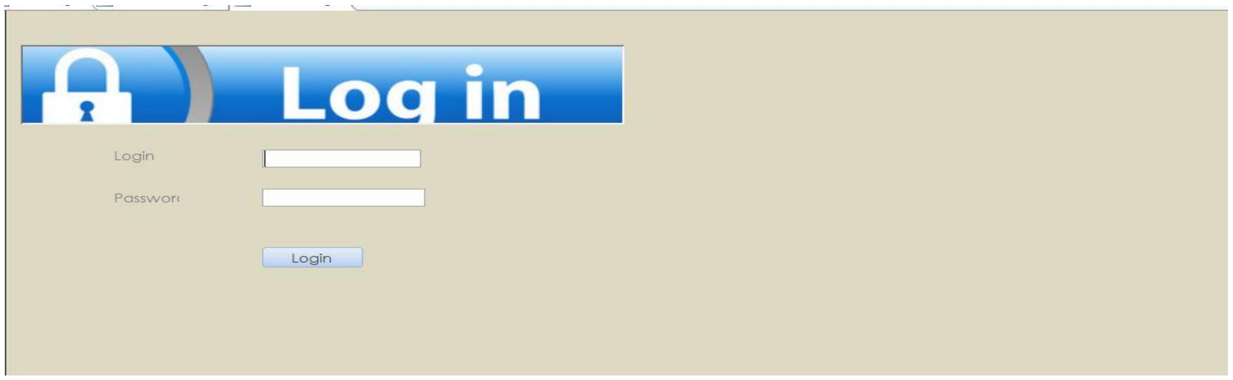
For Customer, he/she can specify the preference such as music, smoking or pet to help match with a suitable car owner


Customer Signup

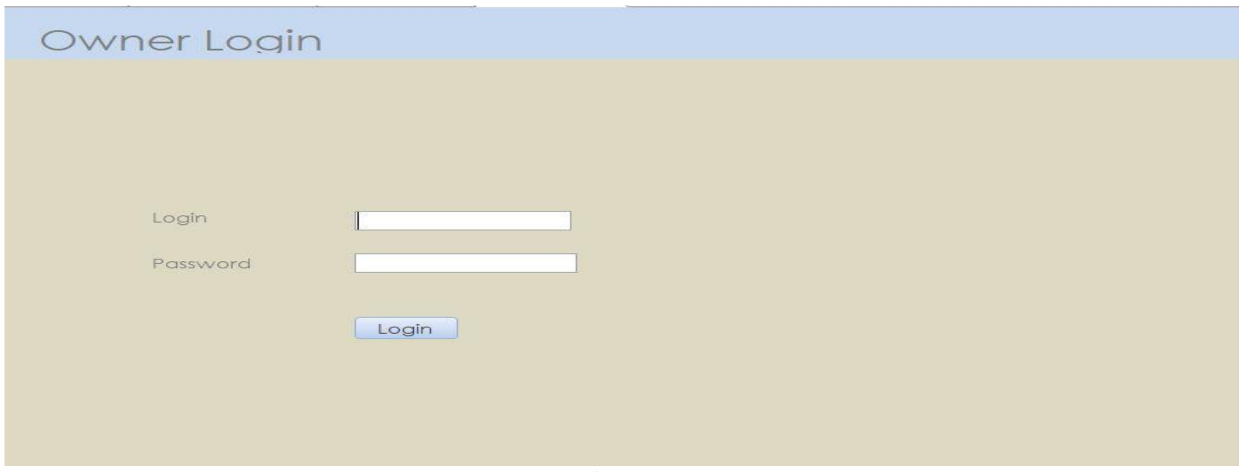
CustomerID	18
First name	Anurag
Last name	KS
mailid	garuna1994@gmail.com
phone number	9723027648
street	McCallum
city	Dallas
state	NY
Disablity	☐
smoking	☑
pet	☐
music	Pop
owner	Yes
password	********

3. Login

Like Sign Up, the login page has separate signups for the customer and owner. If the username or password does not match the system denies the permission and gives message “incorrect login or password”.



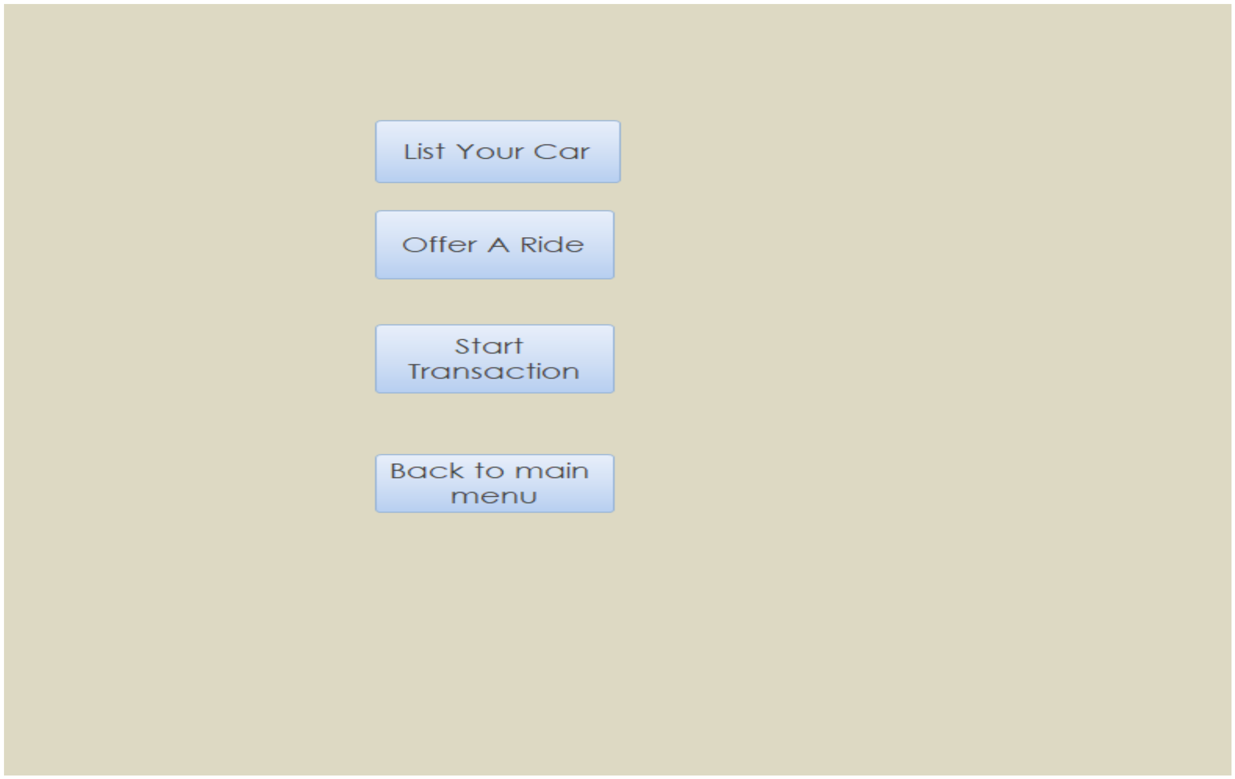
Customer



Owner

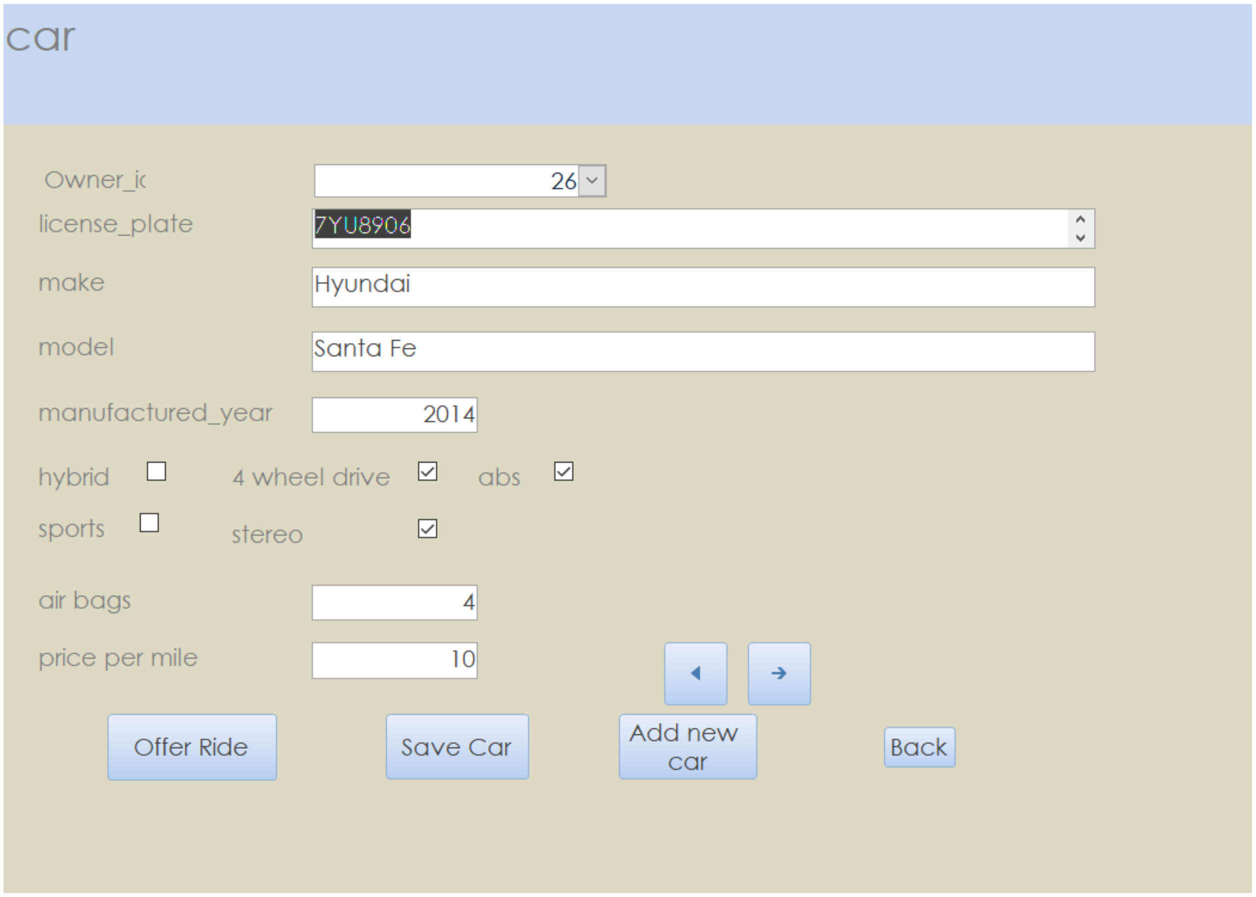
4. After owner login

It provides you the option to list out the registered cars, offer a new ride and start a transaction.



a) List your car

This form saves all the details about the car such as, manufacturing date, model number, license plate number etc. It gives options to save the car or add a new car

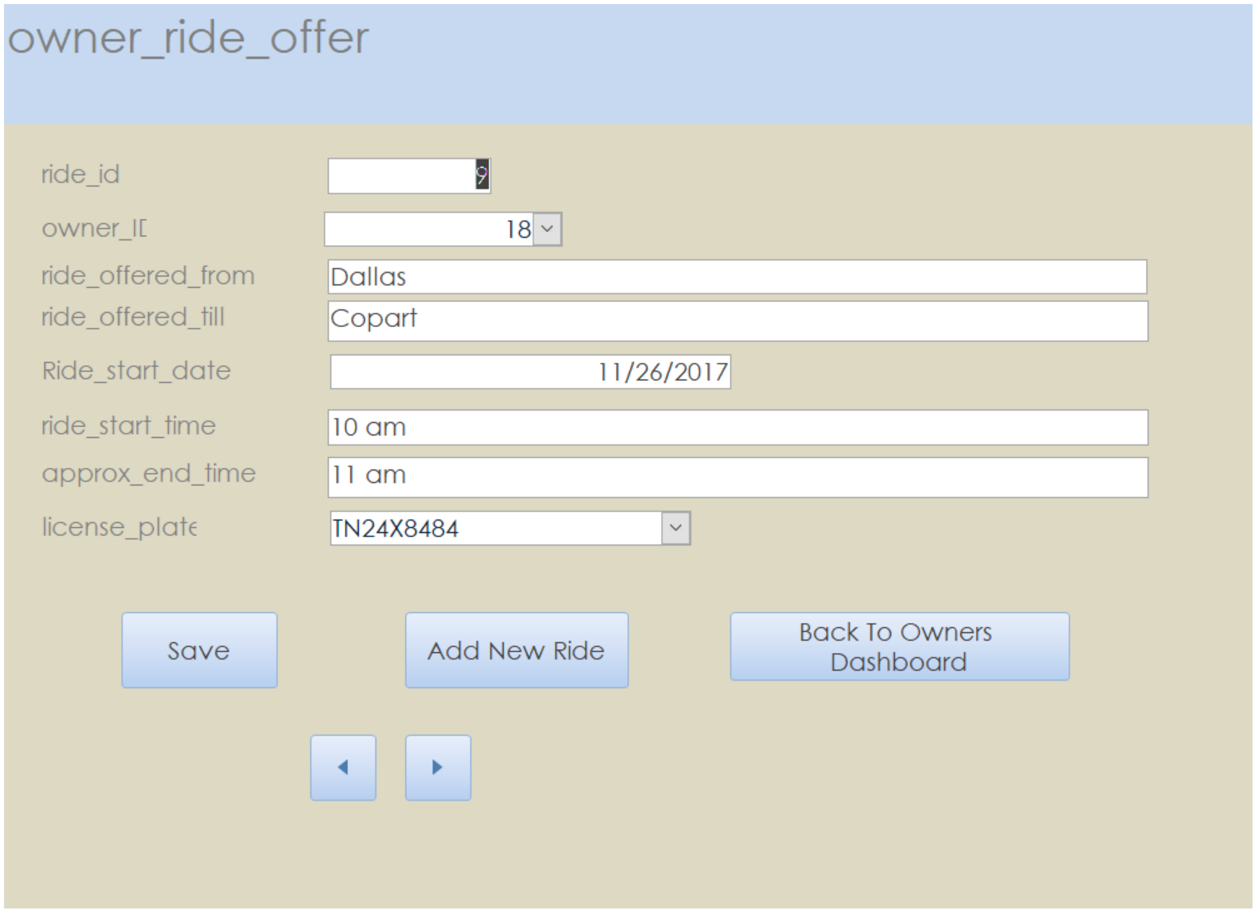


car

Owner_ic	26
license_plate	7YU8906
make	Hyundai
model	Santa Fe
manufactured_year	2014
hybrid	☐
4 wheel drive	☑
abs	☑
sports	☐
stereo	☑
air bags	4
price per mile	10

b) Offer a ride

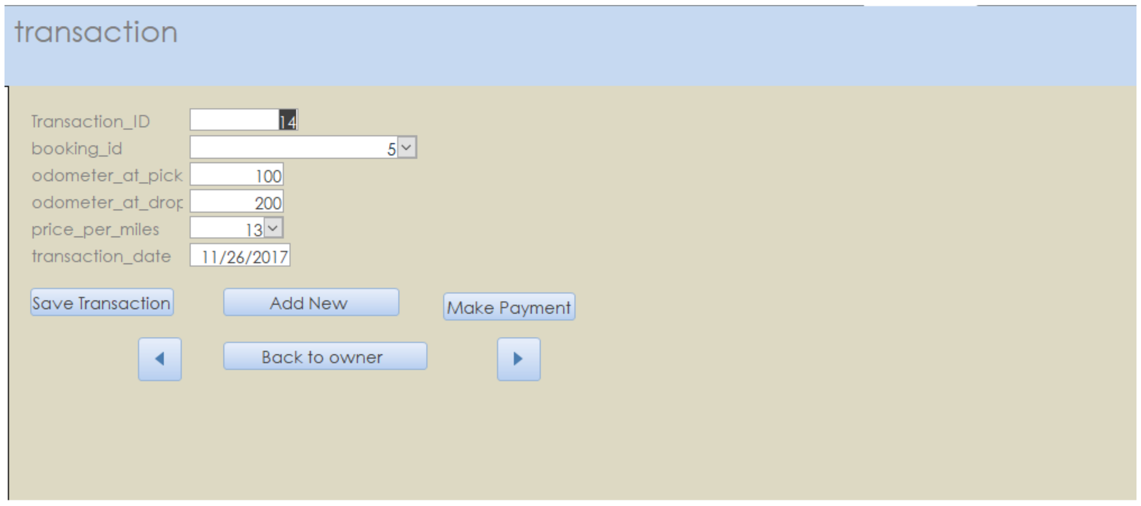
Offer a ride form offers owner to enter a new ride. It asks for the details of ride like the source, destination, ride start date, ride end date and the license plate of the car.


owner_ride_offer

ride_id	9
owner_IĽ	18
ride_offered_from	Dallas
ride_offered_till	Copart
Ride_start_date	11/26/2017
ride_start_time	10 am
approx_end_time	11 am
license_plate	TN24X8484

c) Start Transaction

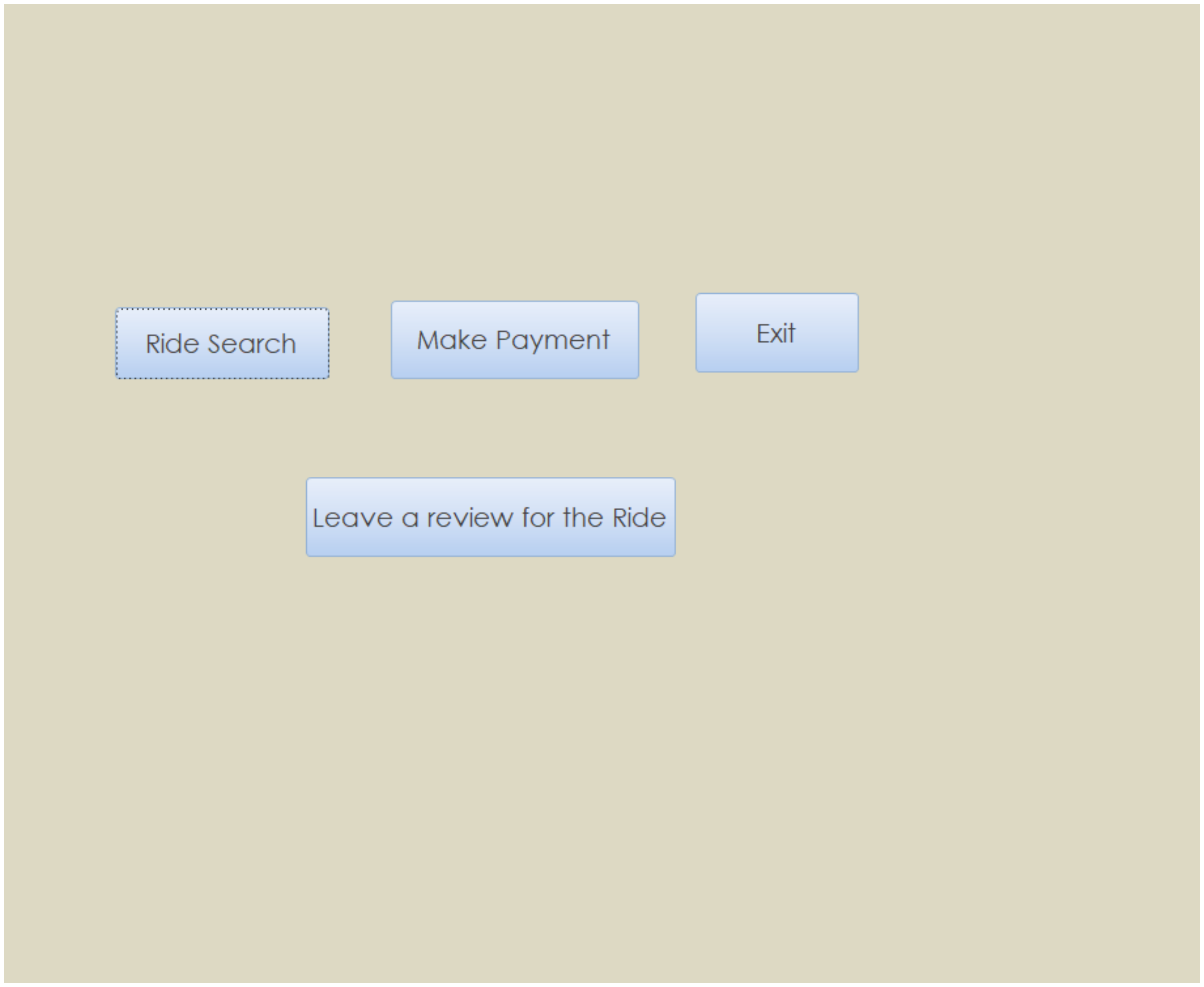
If the owner accepts a ride, he/she will need to record the ride in the Transaction form. The Transaction ID is auto-generated by the system. The owner will need to select the bookingID, record the odometer at pick and drop locations. Owner can also choose the price per mile in here. The transaction date is auto-generated but owner can manually change it. Once everything is entered correctly, owner clicks on Save Transaction or Add New to automatically save it and create new transaction


transaction

Transaction_ID	14
booking_id	5
odometer_at_pick	100
odometer_at_drop	200
price_per_miles	13
transaction_date	11/26/2017

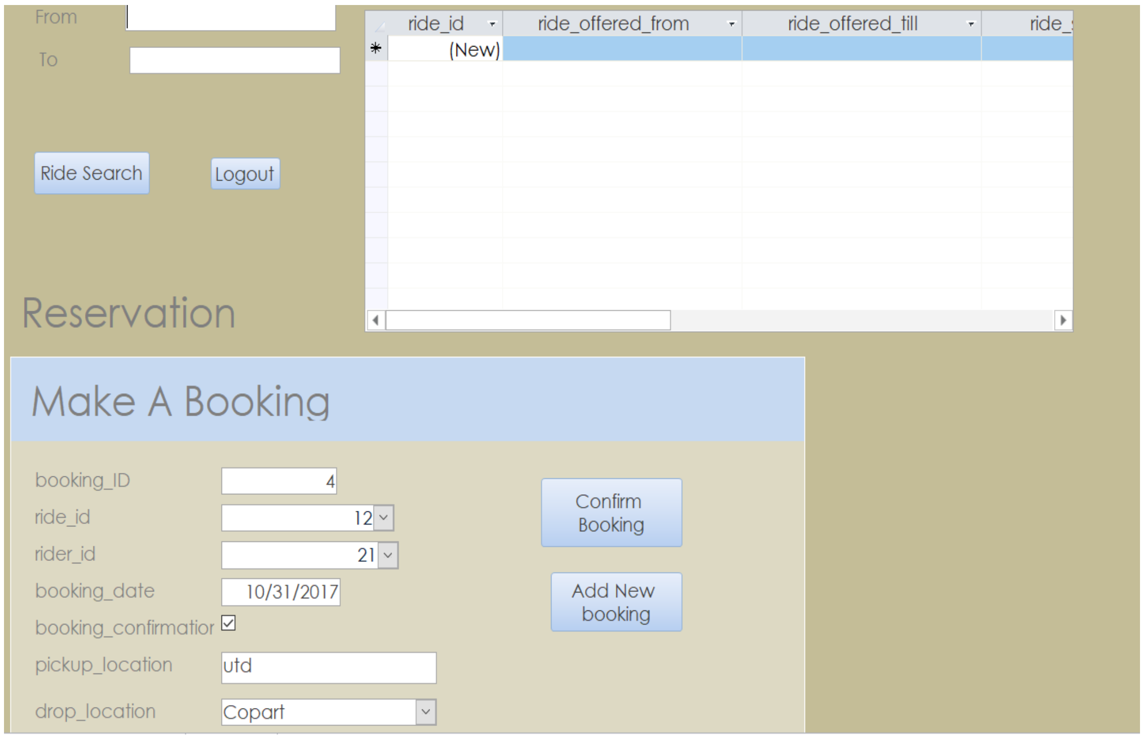
5. After Customer login

Once a customer is successfully logged in, he is redirected to a form where he gets four menu options: Ride Search, Make Payment, Leave a Review for a ride and Exit.



a). Ride Search:

Ride Search is a form which is used by customer to search any rides going to the customer's desired destination. It lists all the rides available after executing the search query and allows customer to book and confirm booking.



From

To

Ride Search | Logout

ride_id	ride_offered_from	ride_offered_till	ride_
(New)			

Reservation

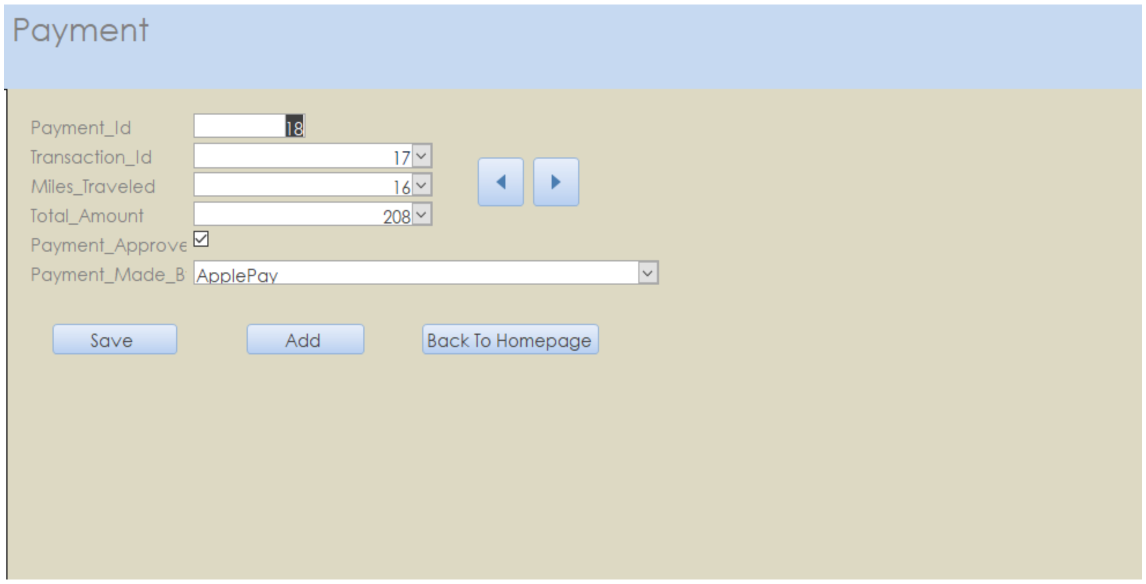
Make A Booking

booking_ID	4
ride_id	12
rider_id	21
booking_date	10/31/2017
booking_confirmatior	☑
pickup_location	utd
drop_location	Copart

Confirm Booking | Add New booking

b) Make Payment

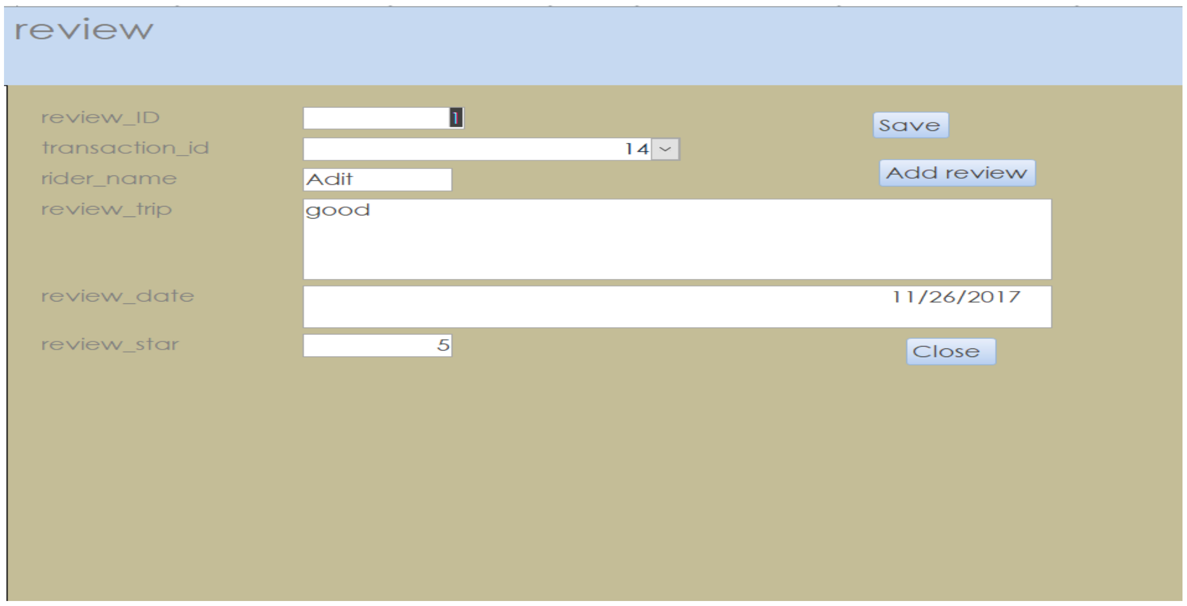
After completing a ride make payment form allows the customers to make the payment for their taken ride.


Payment

Payment_Id	18
Transaction_Id	17
Miles_Traveled	16
Total_Amount	208
Payment_Approve	✓
Payment_Made_B	ApplePay

c) Leave a review

This form is make for customers to enter a review after every successful ride and give the rating of the owners.


review

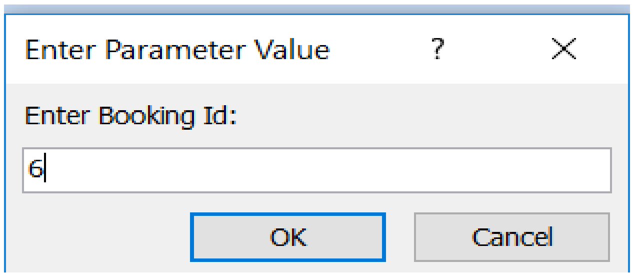
review_ID	1
transaction_id	14
rider_name	Adit
review_trip	good
review_date	11/26/2017
review_star	5

VII. SAMPLE REPORTS

1. Receipt

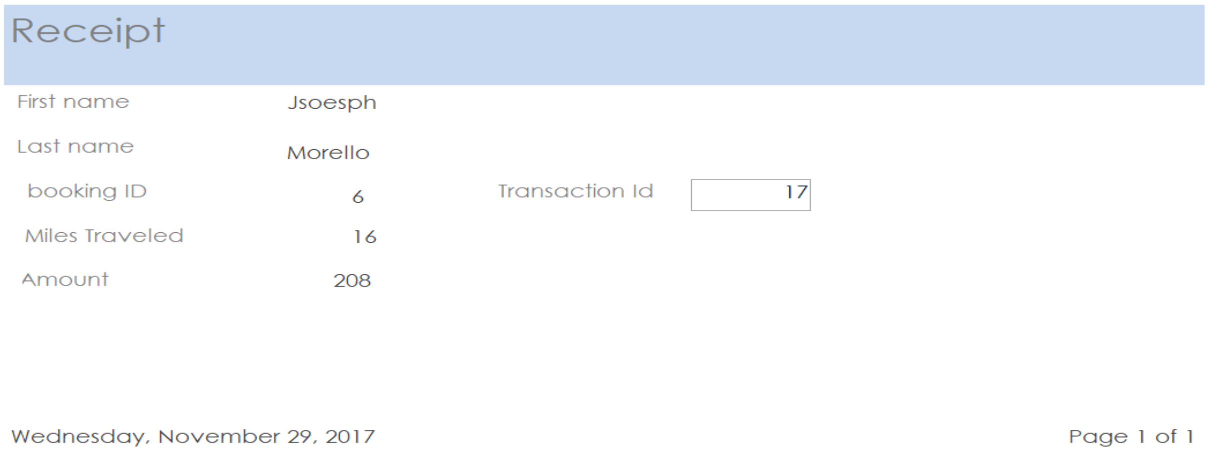
This report prompts customer to enter his booking Id and generates a receipt for him containing his booking details and total payment. For example: in the below screen the receipt is generated to booking Id 5. Customer can keep track of their past travel with us

```
SELECT booking.booking_ID, Customer.[First name], Customer.[Last name], Payment_Table.Miles_Traveled, Payment_Table.Total_Amount, Payment_Table.Transaction_Id
FROM (Customer INNER JOIN (booking INNER JOIN [transaction] ON booking.booking_ID = transaction.booking_id) ON Customer.CustomerID = booking.rider_id) INNER JOIN Payment_Table ON transaction.Transaction_ID = Payment_Table.Transaction_Id
WHERE (((booking.booking_ID)=[Enter Booking Id:]));
```


Enter Parameter Value

Enter Booking Id:

6


Receipt

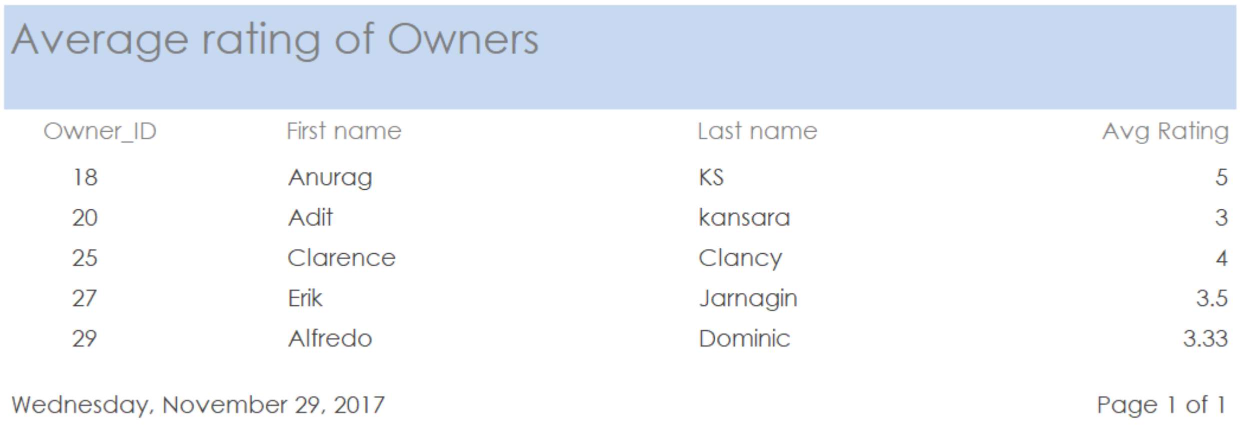
First name	Jsoesph		
Last name	Morello		
booking ID	6	Transaction Id	17
Miles Traveled	16		
Amount	208		

Wednesday, November 29, 2017 Page 1 of 1

2. Owner Review

Owner review report gives the average rating of all the owners based on the rating given by customers who took the rides offered by those owners. We can keep track of the rating of car owners and have appropriate action if any car owner has very low rating

```
SELECT owner.Owner_ID, Customer.[First name], Customer.[Last name], Round(Avg(review.review_star),2) AS Expr1
FROM ((Customer INNER JOIN owner ON Customer.CustomerID = owner.Owner_ID) INNER JOIN (owner_ride_offer INNER JOIN (booking INNER JOIN [transaction] ON booking.booking_ID = transaction.booking_id) ON owner_ride_offer.ride_id = booking.ride_id) ON owner.Owner_ID = owner_ride_offer.owner_ID) INNER JOIN review ON transaction.Transaction_ID = review.transaction_id
GROUP BY owner.Owner_ID, Customer.[First name], Customer.[Last name];
```



Average rating of Owners

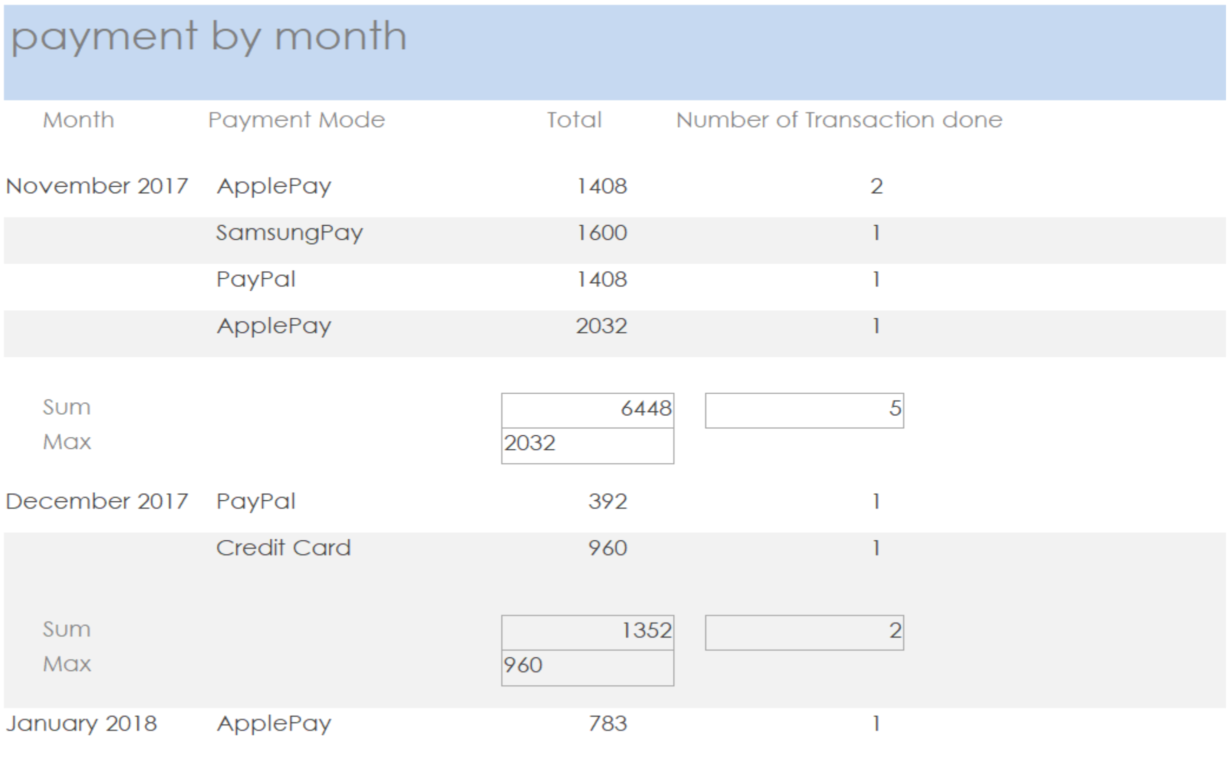
Owner_ID	First name	Last name	Avg Rating
18	Anurag	KS	5
20	Adit	kansara	3
25	Clarence	Clancy	4
27	Erik	Jarnagin	3.5
29	Alfredo	Dominic	3.33

Wednesday, November 29, 2017 Page 1 of 1

3. Monthly Payment Report

Payment by Months list all the payments done by different types of payment methods for each month. It also displays the total amount of transaction done in the month for each payment method. We can see which payment type is used the most than difference in numbers of transactions in each month

```
SELECT Payment_Table.Payment_Made_By, Sum(Payment_Table.Total_Amount) AS SumOfTotal_Amount, Count(transaction.Transaction_ID) AS CountOfTransaction_ID, transaction.transaction_date
FROM [transaction] INNER JOIN Payment_Table ON transaction.Transaction_ID = Payment_Table.Transaction_Id
GROUP BY Payment_Table.Payment_Made_By, transaction.transaction_date;
```


payment by month

Month	Payment Mode	Total	Number of Transaction done
November 2017	ApplePay	1408	2
	SamsungPay	1600	1
	PayPal	1408	1
	ApplePay	2032	1
Sum		6448	5
Max		2032	
December 2017	PayPal	392	1
	Credit Card	960	1
Sum		1352	2
Max		960	
January 2018	ApplePay	783	1

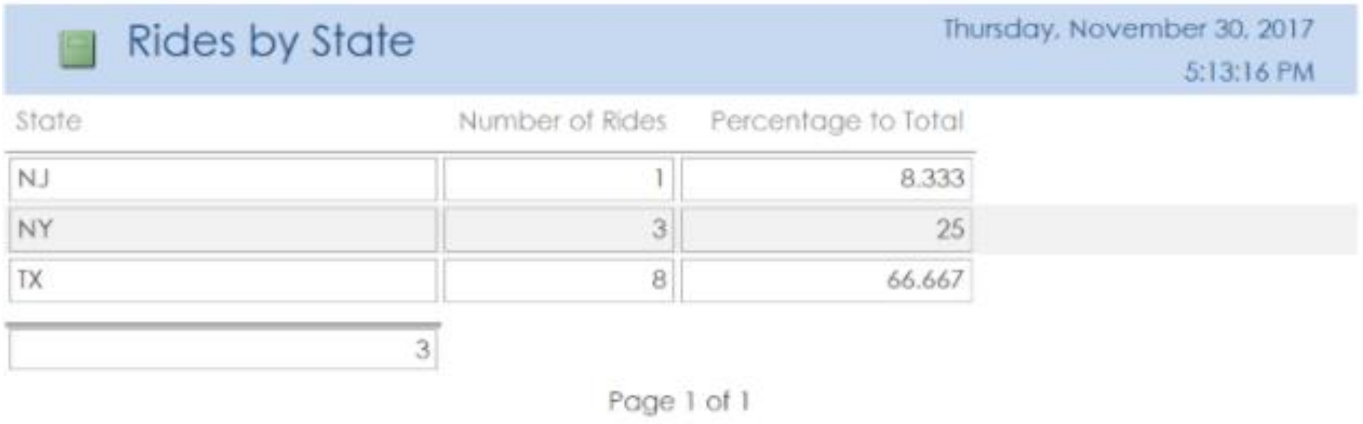
4. Ride by State

This report shows Number of Rides by States and the percentage of rides to total rides. It helps us have an overview of our performance in each state so we can adjust our business strategy.

SELECT DISTINCTROW Customer.state, Count(*) AS [Number of Rides], ROUND(([Number of Rides]/V2.Total)*100,3) AS ["Percentage to Total"]

FROM (SELECT SUM(V1.[Number of Rides]) AS Total FROM (SELECT DISTINCTROW Customer.state, Count(*) AS [Number of Rides] FROM owner_ride_offer INNER JOIN (Customer INNER JOIN booking ON Customer.[CustomerID] = booking.[rider_id]) ON owner_ride_offer.[ride_id] = booking.[ride_id] GROUP BY Customer.state) AS V1) AS V2, owner_ride_offer INNER JOIN (Customer INNER JOIN booking ON Customer.[CustomerID] = booking.[rider_id]) ON owner_ride_offer.[ride_id] = booking.[ride_id]

GROUP BY Customer.state, V2.TOTAL;



Rides by State

Thursday, November 30, 2017
5:13:16 PM

State	Number of Rides	Percentage to Total
NJ	1	8.333
NY	3	25
TX	8	66.667
3		

Page 1 of 1

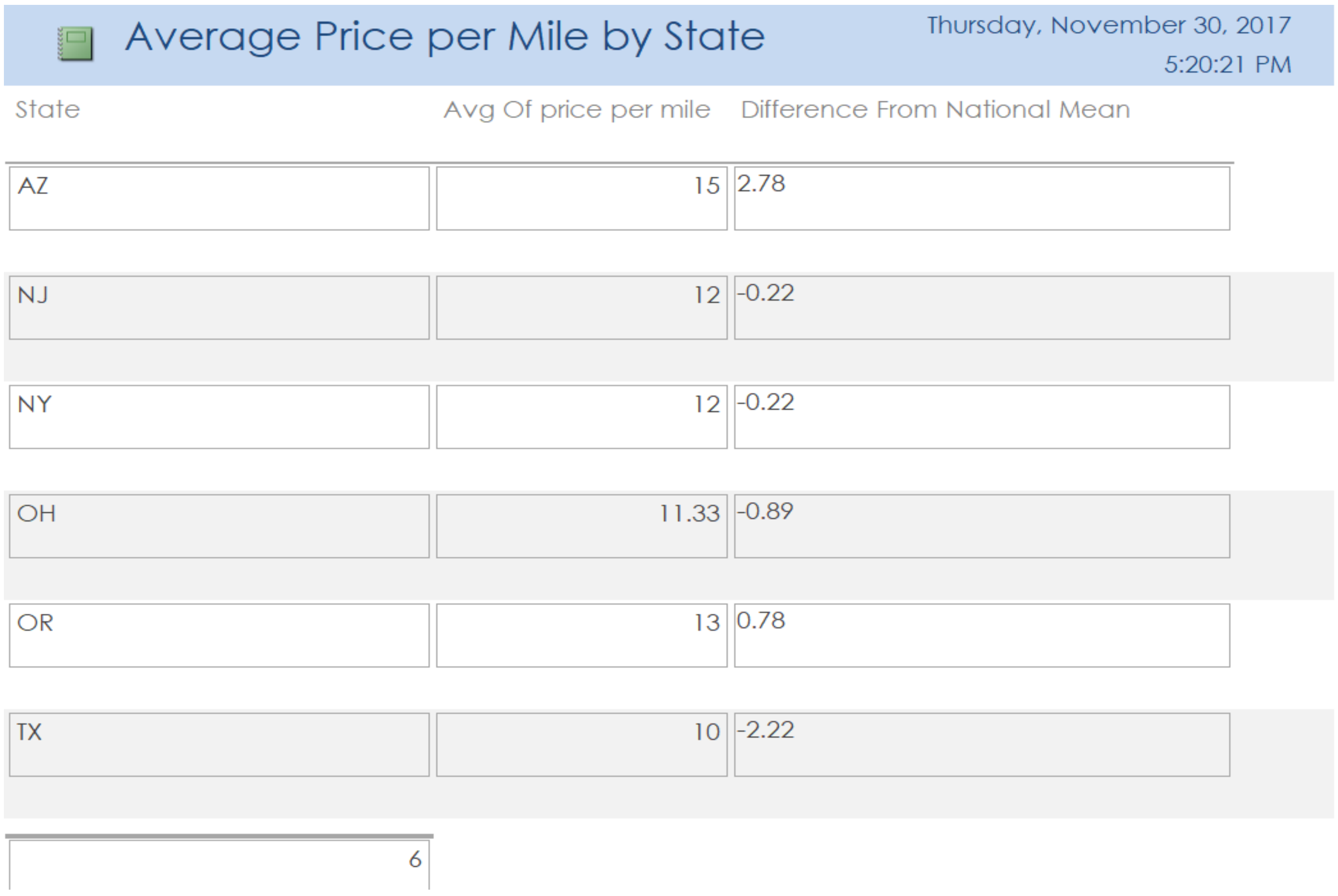
5. Average Price per Mile by State

This report displays average price by state and its difference from the national average. It helps us understand the cost in each state. The query is below:

```
SELECT V2.Customer.state, V2.[Avg Of price per mile], Round(V2.[Avg Of price per mile]-Avg(V1.[Avg Of price per mile]),2) AS Expr1

FROM (SELECT DISTINCTROW Customer.state, Round(Avg(car.[price per mile]),2) AS [Avg Of price per mile] FROM (SELECT avg(car.[price per mile]) AS [avg diff] FROM car)  AS t1, (Customer INNER JOIN owner ON Customer.[CustomerID] = owner.[Owner_ID]) INNER JOIN car ON owner.[Owner_ID] = car.[Owner_id] GROUP BY Customer.state)  AS V1, (SELECT DISTINCTROW Customer.state, Round(Avg(car.[price per mile]),2) AS [Avg Of price per mile] FROM (SELECT avg(car.[price per mile]) AS [avg diff] FROM car)  AS t1, (Customer INNER JOIN owner ON Customer.[CustomerID] = owner.[Owner_ID]) INNER JOIN car ON owner.[Owner_ID] = car.[Owner_id] GROUP BY Customer.state)  AS V2

GROUP BY V2.Customer.state, V2.[Avg Of price per mile];
```



Average Price per Mile by State

Thursday, November 30, 2017
5:20:21 PM

State	Avg Of price per mile	Difference From National Mean
AZ	15	2.78
NJ	12	-0.22
NY	12	-0.22
OH	11.33	-0.89
OR	13	0.78
TX	10	-2.22
6		

VIII. CONTRIBUTION

A. Duy Vu

- Finalized Word Report
- Developed Business Reports
- Wrote Scope of Database and Sample Reports
- Wrote part of Menu and Data Input Screens

B. Adit Kansara

- Proposed the idea of the project
- Formulated Tables, Forms and Reports
- Formatted and aligned forms and reports
- Finalized the final report
- Relational Database Schema
- Worked on Data Entry

C. Anurag Kanchibotha

- Developed Forms, Tables and Reports
- Formatted and aligned forms and reports
- Finalized the final report
- Relational Database Schema
- Prepared the ERD
- Worked on data entry

D. Shweta Singh

- Developed Reports
- Prepared the initial draft of the final report
- Relational Database Schema
- Prepared the initial ERD
- Formatted and aligned Forms