1. INTRODUCTION

We have chosen to produce a Car Rental system. In our system, Customer can rent a car based on make and a model. Our system provides customer to have different pick-up and drop off locations and will impose late fee if the rental car is returned beyond the return date and time. The Customers can purchase car rental insurance which is optional and can use up to one discount coupon to their final bill. Customers who have membership will be by default given a 10% discount in their final bill. We will see detailed description below.

1.1 Objective of the project

- To ease the car renting for masses.
- To create a web based system to book, rent car for a person, family or an entire organization.

1.2 Features of the project

- Car rental agency should have collection of cars.
- Each car should belong to a particular Car Category and each car will belong to a particular location.
- Customer, based on his location and car category preferences, rents a car.
- Based on his location and car category preferences, list of cars available to rent will be shown along with available date and time (from and to).
- Customer will select a car from the suggestions and should be able to reserve it for rent.
- When a customer reserves a car, he/she should be able to optionally purchase a Car Insurance Plan and should be able to apply at most one discount code.
- If a customer is also a member of the car rental agency and has a membership ID then he/she will be given a default 10% discount in additional to the discount code applied. Therefore the total discount percentage will be 10 plus the discount percentage given by the discount code applied.
- Billing is generated when a car is returned.
- Customer can return the car before the due date, on the due date or he/she can return it late also.

- If a customer returns a car after the due date, additional late fee is calculated and added to the bill.
- A default 8.25% tax is applied on the amount which also includes the late fee and this tax is added to obtain the total amount to which the discount will be applied and a final amount is obtained.
- Once the car is returned it becomes available for the booking.
- A booking can be cancelled until 5 days before the actual pick up.
- Company may have several discount plans like weekend discount, corporate discount etc.
- Car price will be calculated based on the selected make and model.

2. SYSTEM DESIGN

2.1 ER DIAGRAM – HIGH LEVEL DATA MODELING

Entity-Relationship (ER) model is a high-level data model that is useful in developing a conceptual design for a database. Creation of an ER diagram, which is one of the first steps in designing a database, helps the designer(s) to understand and to specify the desired components of the database and the relationships among those components.

It is a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of data within databases or information systems.

An ER model is a diagram containing entities or "items", relationships among them, and attributes of the entities and the relationships.

2.1.1 Entities

An entity is a real-world item or concept that exists on its own. The set of all possible values for an entity is the entity type.

- **Customer:** Customer will be the one who is using car rental system for reserving a car. He can be a member of the system or a non-member of the system. Member of the system will have membership id. Customer entity will store details like customer driving license number, email, address, name, and phone number.
- **Car:** Car entity will have list of cars available in the system. Each car will be associated with a car category and car will have attributes like make, model, mileage and registration number. Car will also have separate flag to check the availability of the car.
- **Car Category:** Every car has a car category. Price is calculated based on the car category. Car category will have attributes like no of person, no of luggage's, name, and cost per day and late fee per hour.
- **Location:** Location entity here denotes the pickup and drop off location of the car. Customer can pick up the car from the particular location and can have same or different drop off location. Location will have attributes like Location id, name and address.
- **Booking:** Each car reservation will be monitored in the entity called booking. Booking will have attributes like booking id, from date and time of booking and due return date

and time and actual return date and time of the booking, and booking status. This booking amount might also include rental insurance and discount code.

- **Billing:** When a customer returns a car, a bill will be generated on the particular booking. Billing have attributes like Bill ID, bill date, bill status, total late fee, tax amount, and total amount.
- **Discount_Customer:** can apply discount code while the bill is generated. Each discount code has different discount percentage. Discount will have attributes like discount code, name, expiry date and discount percentage.
- **Car_Rental_Insurance:** Customer may already have car rental insurance or can buy one while booking the car. Car rental insurance will have attributes like insurance code, coverage type, name and cost per day.

2.1.2 Relations

A relationship type is a set of associations among entity types.

- **Car to Car Category**: Every car is associated with a car category. Once customer selects a car, the cost per day is obtained from the car category that the selected car belongs to. The relation name is 'Belongs to'.
- **Car to Location:** Customer will be picking up or dropping the car in a particular location. Customer can pick up or drop-off the car at the particular location. So, cars will be present at a location. The relation name is 'Current location'.
- **Booking to Billing:** Once customer returns a car bill will be generated for each booking. There can be case like booking is cancelled in that case no bill will be associated with the booking. The relation name is 'Gives'.
- **Booking to Discount:** Customer may apply a discount code when he/she books a car. This discount will be applied to the total amount after tax and late fee while the bill is generated. Based on the discount code total amount will be reduced by some percentage. The relation name is 'Has'.
- **Booking to Car Rental Insurance:** Customer can select rental insurance while booking a car so that rental insurance will cover damages based on the coverage type. The relation name is 'Includes'.

- **Booking to Location:** Customer can pick a car for rent from a particular location. The relation name is ‘Pick up location’.

- **Booking to Location:** Customer can drop off rental car in a particular location. The relation name is ‘Drop off location’.

- **Customer to Car to Booking:** Customer will select car for rent. So the customer will be related to the both car and the booking. The relation between these 3 entities is a ternary relation and the relation name is ‘Rents’.

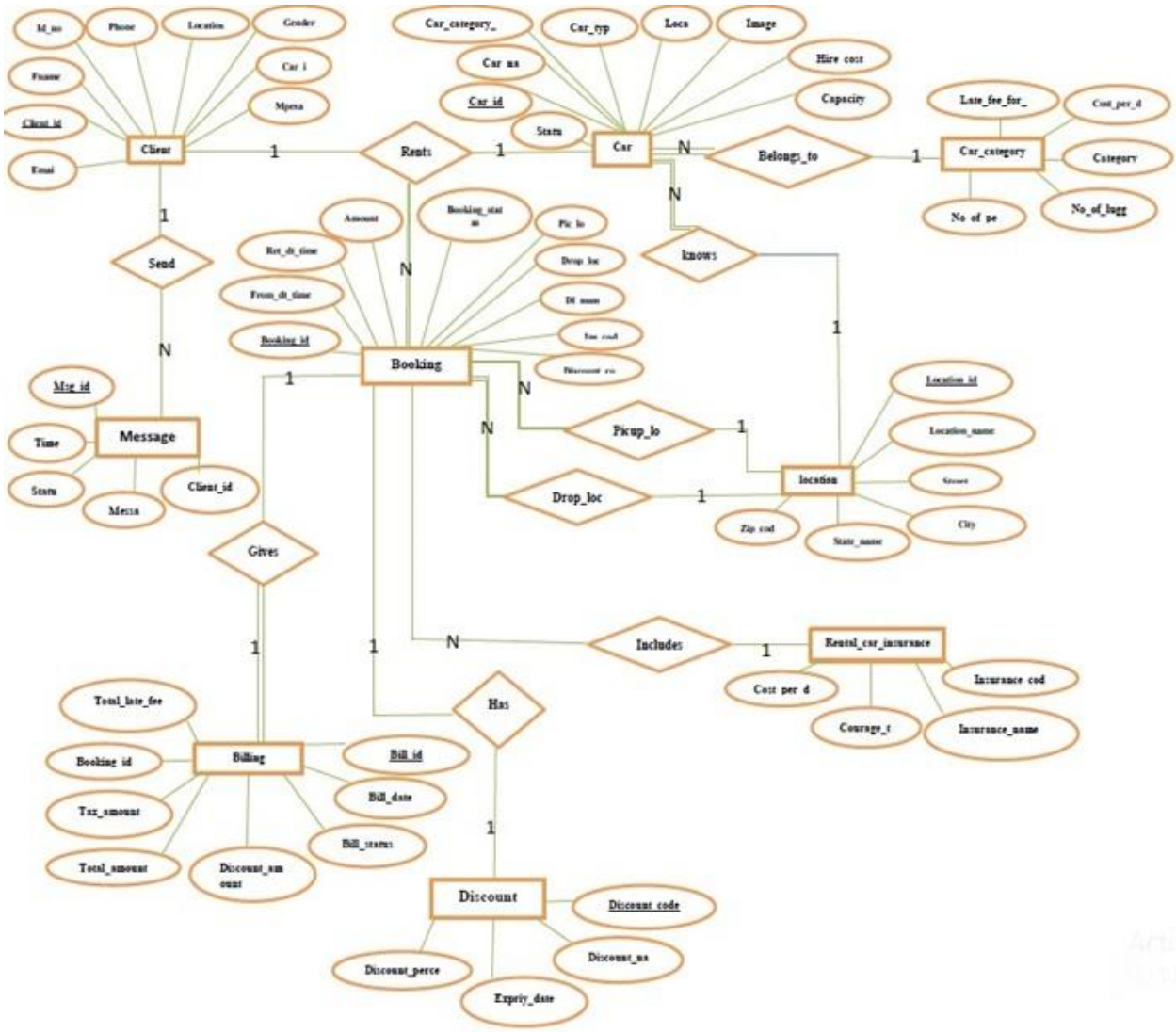


Figure 2.1: ER Diagram

2.2 SCHEMA DIAGRAM – CONCEPTUAL DATA MODELING

A **conceptual schema** or conceptual data model is a map of concepts their relationships used for databases. This describes the semantics of an organization and represents a series of assertions about its nature. Specifically, it describes the things of significance to an organization (entity classes), about which it is inclined to collect information, and characteristics of (attributes) and associations between pairs of those things of significance (relationships).For convenience, we have chosen to represent our final relational schema in 2NF normalization. We have de-normalized from 3NF to 2NF.

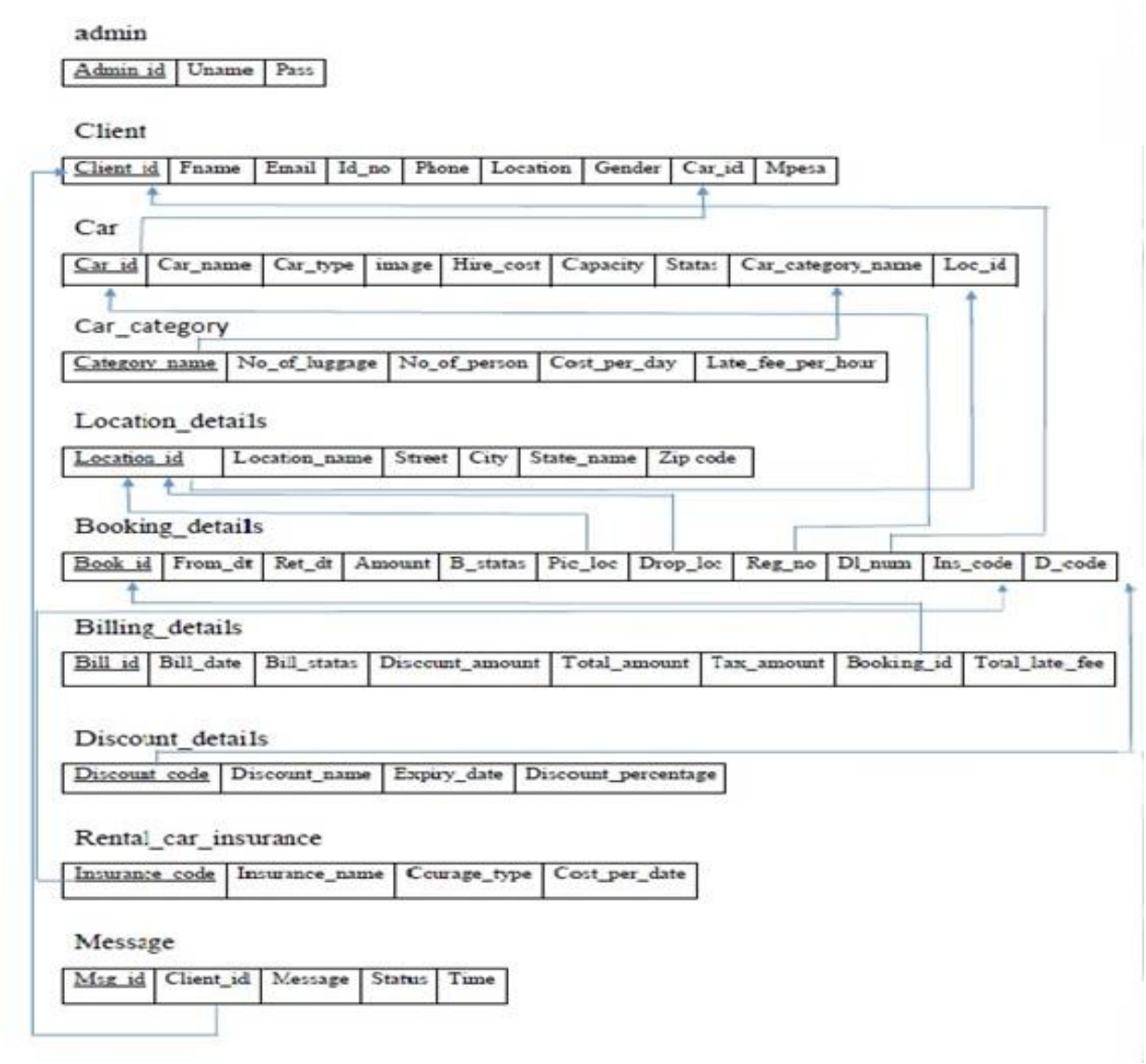


Figure 2.2: Schema Diagram.

2.3 State Diagram

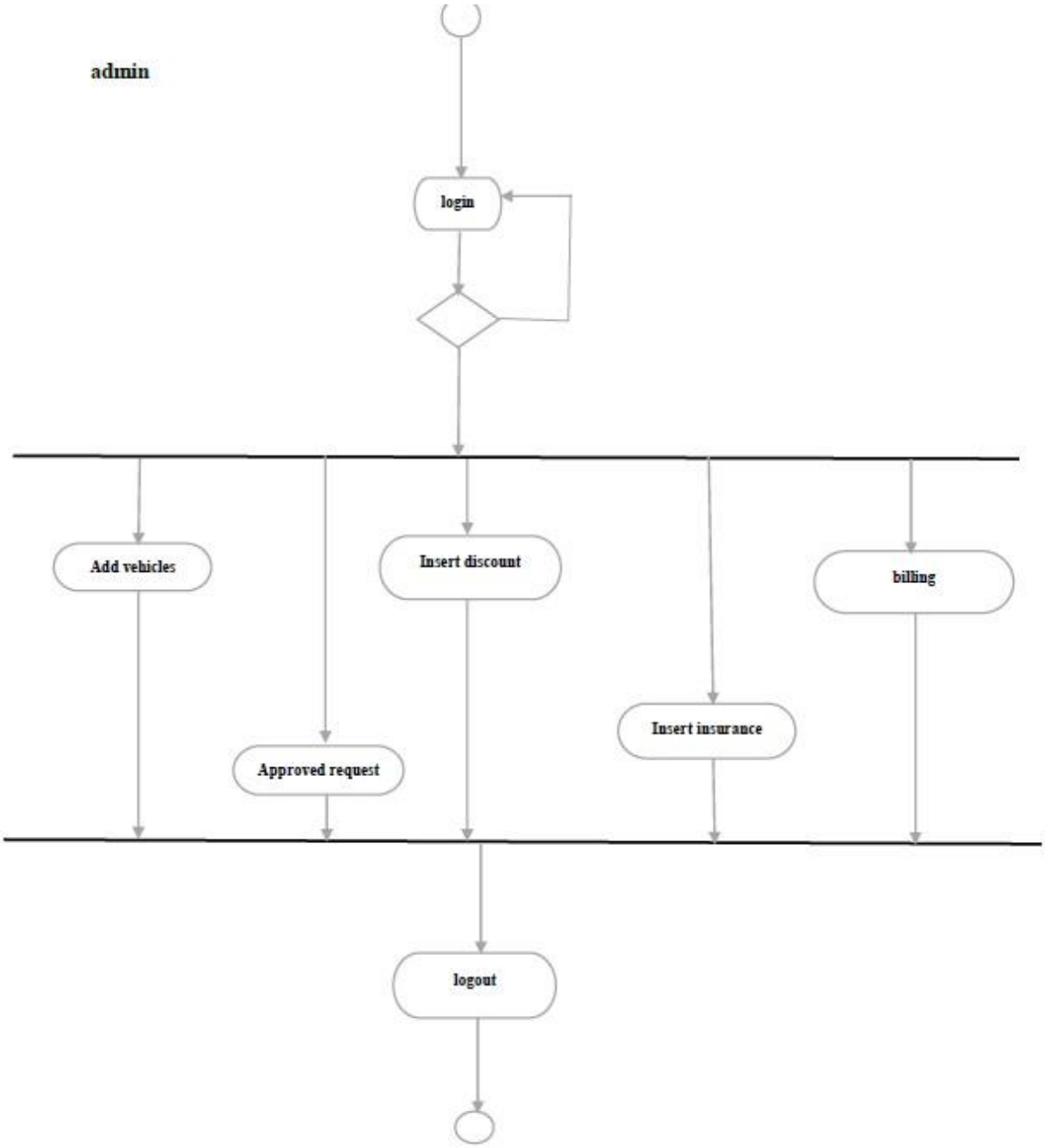


Figure 2.3: State Diagram of the User

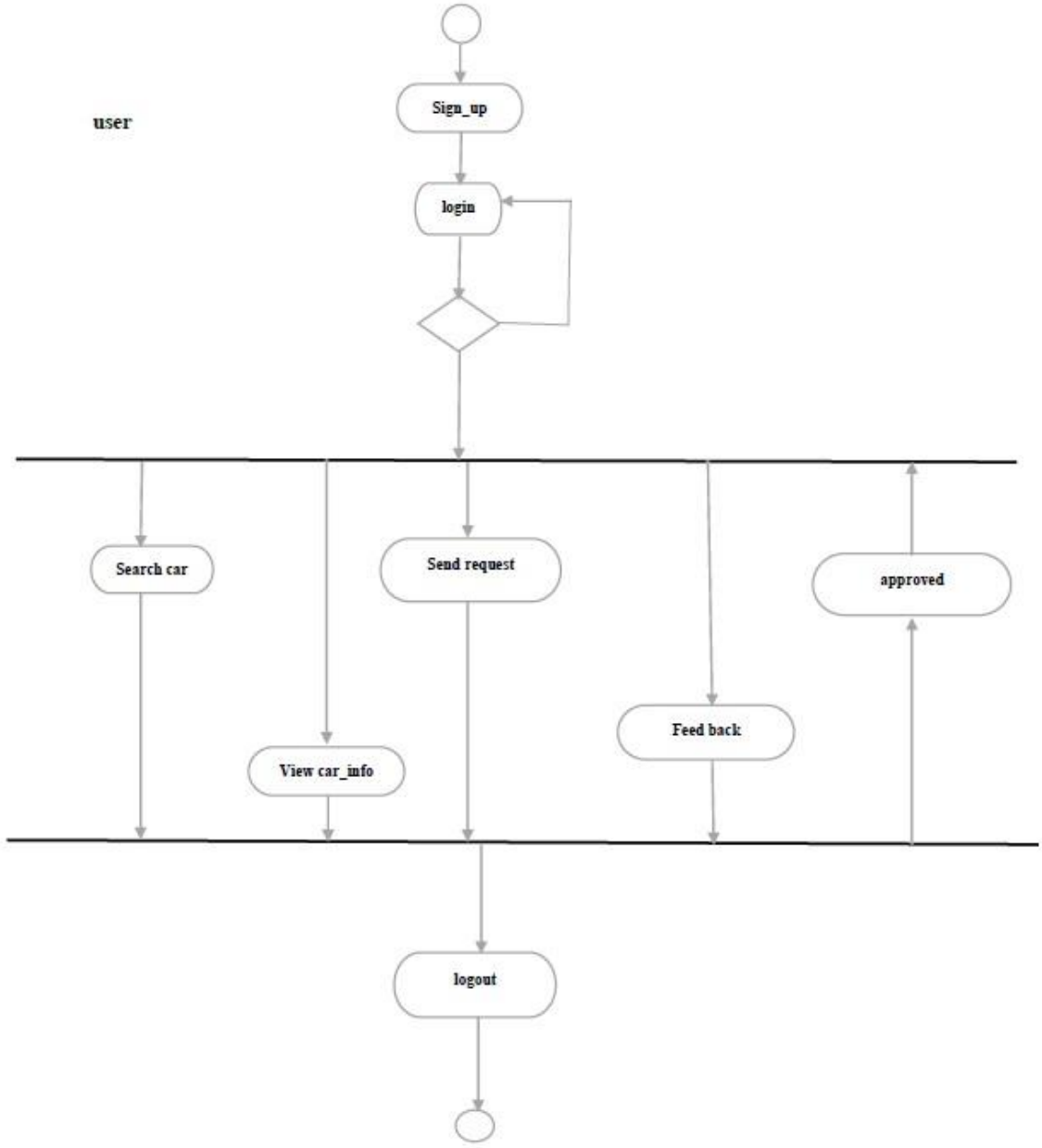


Figure 2.4: State Diagram of the Admin

3. NORMALIZATION UP TO 3NF

All the relations mentioned in the relational schema are in **1NF** because the multi valued attribute phone of all the employees has been divided with another relation with employee_id as primary key. Since every relation has only one primary key, all the other attributes are dependent on that key so there is no partial dependency so it is in **2NF.**Also relation is said to be in 3NF if X->Y, X should be a super key or Y should be a prime attribute. As all the relations contain unique primary key so that key itself is the minimal super key hence we can declare that all the relations are in **3NF.**

- **Customer_Details Relation:**
  - DL_number → Fname, Mname, Lname, Phone_number, Email_id, Street, City, State, Zip code, Membership_id, Membership_type.
  - Zipcode → State,City
- **Car Relation:**
  - Registration_number → Model, Make, Model_year, Car_category_name, Loc_id, Mileage, Availability_flag.
  - Model → Make
- **Car_Category Relation:**
  - Category_name → No_of_luggage, No_of_person, Cost_per_day, Late_fee_per_hour.
- **Location _Details Relation:**
  - Location_id → Name,Street,City,State,Zip code.
  - Zip code → State,City.
- **Booking_Details Relation:**
  - Booking_id → From_dt_time, Ret_dt_time, Amount, Booking_status, Pickup_loc, Drop_loc, Reg_num, DL_num, Ins_code, Act_ret_dt_time, Discount_code.
  - Bill_id → Bill_date, Bill_status, Discount_amt, Total_amt, Tax_amt, Booking_id, Total_late_fee.
- **Discount_Details Relation:**
  - Discount_code → Discount_name,Expiry_date,Discount_percentage.
  - Discount_name → Discount_code ,Expiry_date,Discount_percentage.

- Rental_Car_Insurance Relation:
  - Insurance_code → Insurance_name,Coverage_type,Cost_per_day.
  - Insurance_name → Insurance_code ,Coverage_type,Cost_per_day .

3.1 Functional dependencies that violated normalization rules:

The Following transitive dependencies exist in the relational schema.

- **Customer_Details Relation**
  - DL_number → Zipcode.
  - Zipcode → State, City.
- **Car Relation**
  - Registration_number → Model_name.
  - Model_name → Make.
- **Location_Details Relation**
  - Location_id → Zipcode.
  - Zipcode → State, City.

4. SYSTEM IMPLEMENTATION

4.1 INTRODUCTION TO SQL

SQL (**Structured Query Language)**, is a language used to request data from a database, to add, update, or remove data within a database, or to manipulate the metadata of the database. SQL is an ANSI (American National Standards Institute) standard.

SQL is a declarative language in which the expected result or operation is given without the specific details about how to accomplish the task. The steps required to execute SQL statements are handled transparently by the SQL database. Sometimes SQL is characterized as non-procedural because procedural languages generally require the details of the operations to be specified, such as opening and closing tables, loading and searching indexes, or flushing buffers and writing data to files systems. Therefore, SQL is considered to be designed at a higher conceptual level of operation than procedural languages because the lower level logical and physical operations aren't specified and are determined by the SQL engine or server process that executes it.

Instructions are given in the form of statements, consisting of a specific SQL statement and additional parameters and operands that apply to that statement. SQL statements and their modifiers are based upon official SQL standards and certain extensions to that each database provider implements. Commonly used statements are grouped into the following categories:

Data Query Language (DQL)

1. SELECT - Used to retrieve certain records from one or more tables.

Data Manipulation Language (DML)

1. INSERT - Used to create a record.
2. UPDATE - Used to change certain records.
3. DELETE - Used to delete certain records.

Data Definition Language (DDL)

1. CREATE – used to create a new table, a view of a table, or other object in database.
2. ALTER – used to modify an existing database object, such as a table.
3. DROP – used to delete an entire table, a view of a table, or other database object.

Data Control Language (DCL)

1. GRANT - Used to give a privilege to someone.
2. REVOKE - Used to take back privileges granted to someone.

4.2 RELATIONAL ALGEBRAIC QUERIES:

The fundamental operations of the relational algebra are simple operations involving one or two relations as their operands. A result of an operation may be further used as an operand in another operation . The combined effects of a sequence of operations determine the final result
When building a query the task is to find the operation sequence that will produce a correct result. Since the operations of the relational algebra are quite simple, many intermediate results might have to be produced before the final result is reached. The intermediate results are used as operands in the operations that produce new intermediate results.

Some of the relational algebraic expressions that are used in our project are listed below.

1. For each car_category which more than two cars.retrieve the car_category,car_name and number of cars on that category.

   π(car_category, car_name) σ(car_category_name = category_name and car_category>=2) car,car_category;

2. Show the resulting hire_cost of every of any clients purchace benz cars hire_cost is given 10% rise.

   π(client_name) σ(category_name=benz and car_id=car_id) car_category,client;

3. For each car retrieve the car_id,car_name and number of cars on that locations.
   $\pi_{(car\_id,car\_name)}\ \sigma_{(loc\_id=location\_id)}$ car,location_details;

4. Retrieve bill_statas total_amount,tax_amount and booking_id who booked in 2010.
   $\pi_{(bill_status,total_amount,tax_amount,booking_id)}\ \sigma_{(booking_id=booking_idandrent_dt_time=2010)}$car,location_details;

5. Find the sum of discount total_amount of discount_details the maximum and minimum total.
   $\mathcal{F}_{\mathbf{min}(total\_amount)\ \mathbf{max}(total\_amount)}$ discount_details;

6. Retrieve the total numbers of cars in maruthi suzuki car_category.
   $\mathcal{F}_{\mathbf{count}(car\_name)}\ \sigma_{(car\_category=maruthi\ suzuki\ and\ Car\_category\_name=category\_name)}$ car,car_category;

7. Retrieve the cars names which have the word Suzuki in its car_category.
   $\pi_{(car\_name)}\ \sigma_{(category=suzuki)}$ car;

8. Retrieve car_name which have the hirecost between 1000 and 2000.
   $\pi_{(car\_name)}\ \sigma_{(hire\ cost=1000\ and\ hire\_cost=2000)}$ car;

9. Retrieve the discount_name and expriy date which has the discount percentage from 0% to 5%.
   $\pi_{(discount\_name,expriy\_date)}\ \sigma_{(discount\_percentage=0\ and\ discount\_percentage=5)}$ discount_details

10. Retrieve bill_id,booking_id,total_amount who not approved bill_statas having less than Rs 10000.
    $\pi_{(bill\_id,booking\_id,total\_amount)}\ \sigma_{(bill\_status=pendingandtotal\_amount>=1000)}$
    billing_details,booking_details;

11. Retrieve the name of cars who have not booked.
    $\pi_{(car\_name)}\ \sigma_{(car\_id!=reg\_no)}$ car,booking_details;

4.3 QUERIES DESIGNED USING SQL COMMANDS:

```
Create table client (
Client_id varchar(25) not null,
Fname varchar(25) not null,
Email varchar(25) not null,
Id_no int(11) not null
Phone int(11),
Location varchar(25) not null,
Gender varchar(25) not null,
Car_id int(11) not null,
Status varchar(25) not null,
Mpesa varchar(25),
Primary_key(client_id)
);

Create table car (
Car_id varchar(25) not null,
Car_name varchar(25) not null,
Car_type varchar(25) not null,
Image text not null,
Hire_cost int(11) not null,
Capacity int(25) not null,
Status varchar(25) not null,
Car_category_name varchar(25) not null,
Location_id char(4),
registration_number char(7),
Primary key(car_id),
Foreign key (car_category_name) references car_category(car_category_name),
Foreign key(loc_id) references location_details(location_id)
);
```

```
Create table car_category (
Category_name varchar(25) not null,
No_of_luggege int(11) not null,
No_of_person int(11) not null,
Cost_per_day decimal(5,2) not null,
Constraint categorypk primary key(category_name)
);

Create table location_details (
Location_id char(4),
Location_name varchar(50),
Street varchar(30),
City varchar(20),
State_name varchar(20),
Zip_code int(5)
);

Create table booking_details (
Booking_id char(5) not null,
From_dt_time timestamp not null,
Ret_dt_time timestamp,
Amount decimal(10,2) not null,
Booking_status char(1) not null,
Picup_loc char(4),
Drop_loc char(4),
Reg_num char(7) not null,
Dl_num char(7) not null,
Ins_code char(4) default null,
Act_ret_dt_time timestamp,
Discount_code char(4) default null,
Constraint bookingpk primary key(booking_id),
Constraint bookingfk3 foreign key(reg_num) references car(registration_number));
```

```
Create table billing_details (
Bill_id varchar(6) not null,
Bill_date date not null,
Bill_status char(1) not null,
Discount_amount decimal(10,2) not null,
Total_amount decimal(10,2) not null,
Tax_amount decimal(10,2) not null,
Booking_id char(5) not null,
Total_late_fee decimal(10,2) not null,
Constraint billingpk primary key(bill_id),
Constraint billingfk1 foreign key(booking_id) references booking_details(booking_id));
Create table discount_details (
Discount_code char(4) not null,
Discount_name varchar(25) not null,
Expriy_date date not null,
Discount_percentage decimal(4,2) not null
Constraint discountpk primary key(discount_code),
Constraint discountsk unique(discount_name)
);

Create table rental_car_insurance (
Insurance_code char(4) not null,
Insurance_name varchar(50),
Courage_type varchar(20) not null,
Cost_per_day decimal(4,2) not null,
Constraint insurancepk primary key(insurance_code),
Constraint insurancesk unique(insurance_name));

Create table message (
Msg_id int(11) not null,
Client_id int(11) not null,
Message varchar(25) not null,
```

```
Status varchar(25) not null,
Time time,
Primary key(msg_id),
Foreign key(client_id) references client(client_id)
);
```

```
INSERT INTO `client` (`client_id`, `fname`, `email`, `id_no`, `phone`, `location`, `gender`, `car_id`, `status`, `mpesa`, `FROM_DT_TIME`, `RET_DT_TIME`) VALUES ('5', 'Ram', 'ram@gmail.com', '12345', '9632456981', 'mysuru', 'male', '220', 'available', '120', CURRENT_TIMESTAMP);
```

```
INSERT INTO `cars` (`car_id`, `car_name`, `car_type`, `image`, `hire_cost`, `capacity`, `status`, `CAR_CATEGORY_NAME`, `LOC_ID`) VALUES ('123', 'santro', 'hyundai', 'image1.jpg', '12000', '5', 'available', 'MINI VAN', 'L103')
```

```
INSERT INTO CAR_CATEGORY VALUES('ECONOMY',2,5,30,0.9);
INSERT INTO CAR_CATEGORY VALUES('COMPACT',3,5,32,0.96);
INSERT INTO CAR_CATEGORY VALUES('MID SIZE',3,5,35,1.05);
INSERT INTO CAR_CATEGORY VALUES('STANDARD',3,5,38,1.14);
INSERT INTO CAR_CATEGORY VALUES('FULL SIZE',4,5,40,1.2);
INSERT INTO CAR_CATEGORY VALUES('LUXURY CAR',5,5,75,2.25);
```

```
INSERT INTO CARS VALUES ('ABX1234', 'CIVIC', 'HONDA' 2014, 10000, 'ECONOMY', 'L101', 'A);
INSERT INTO CAR VALUES('SDF4567','FIESTA','FORD', 2015, 15000, 'ECONOMY', 'L102','N');
```

```
INSERT INTO DISCOUNT_DETAILS VALUES ('D109','WEEKLY RENTALS', '2020-11-09',25);
INSERT INTO DISCOUNT_DETAILS VALUES ('D972','ONE WAY SPECIAL', '2016-12-15',20);
```

INSERT INTO BOOKING_DETAILS VALUES('B1001','2018-11-11 08:00:00','2018-11-15 17:00:00', 7500,'Approved','L101','L101','103','7','I204',NULL,'D297');
INSERT INTO BOOKING_DETAILS VALUES('B1003','2018-11-09 13:56:33','2018-11-18 00:00:00', 8000,'Approved','L103','L103','101','10','INS1',NULL,'D109');
INSERT INTO `billing_details` (`BILL_DATE`, `BILL_STATUS`, `DISCOUNT_AMOUNT`, `TOTAL_AMOUNT`, `TAX_AMOUNT`, `BOOKING_ID`, `TOTAL_LATE_FEE`) VALUES (CURRENT_TIMESTAMP, 'cleared', '10', '10', '10', 'B1008', '20');

1. For each car_category which more than two cars.retrieve the car_category,car_name and number of cars on that category.

   Select car_category,car_name,count(*) no_cars
   From car_category join car on
   Car_category_name = category_name
   Group by car_category
   order by car_name
   Having count(*) > 2;

2. Show the resulting hire_cost of every of any clients purchace benz cars hire_cost is given 10% rise.

   Select client_name ,1.1 * hire_cost inscost
   From car,car_category,client
   Where car_category=”benz” and client car_id=car car_id.;

3. Set the discount percentage as 25% for holiday special.

   Update discount details
   Set discount percentage =25%
   Where discount name=”holiday special”;

4. For each car retrieve the car_id,car_name and number of cars on that locations.

   Select car_id,car_name ,count(*)
   From car c,location_details l
   Where c.loc_id = l.location_id

Group by car_id;

5. Retrieve bill_statas total_amount,tax_amount and booking_id who booked in 2011.
   Select bill_statas,total_amount,tax_amounr,booking_id
   From billing_details,booking_details
   Where booking_id=booking_id and from_dt_time like “2010%”
   group by total_amount;

6. Retrieve the name of cars who have not booked.
   Select car_name
   from car
   where not exist(select * from
   booking_details
   where car_id=regno);

7. Retrieve the name of cars who have booked
   Select car_name
   from car
   where exit(select * from
   booking_details
   where car_id=regno);

8. Find the sum of discount details percentage of discount_details the maximum and minimum discount.
   Select min(total_amount)minimum, max(total_amount)maximum
   From billing_details;

9. Retrieve the total numbers of cars in maruthi suzuki car_category.
   Select count(*) total_car
   From car,car_category
   Where car_category=”maruthi suzuki” and car category_name=category_name;

10. Retrieve the names and whos names being with T

    Select fname

    From client

    Where fname like “T%”;

11. Retrieve the cars names which have the word Suzuki in its car_category.

    Select car_name

    From car

    Where category like “%suzuki%”;

12. Retrieve car_name which have the hirecost between 1000 and 2000.

    Select car_name

    From car

    Where (hire_cost between 1000 and 2000);

13. Retrieve the discount_name and expriy date which has the discount percentage from 0% to 5%.

    Select discount_name,expriy_date

    From discount_details

    Where (discount percentage between 0% and 5%);

14. Create view cars1 as

    Select location_name,street,city

    From location_details,car

    Where loc_id=loction_id

    And model in(Suzuki,benz);

15. Create view clients as

    select client_id,fname,car_name

    from client,car

    where car_id=car_id and car_id in(abc123,bfx345,cde654);

16. Retrieve bill_id,booking_id,total_amount who not approved bill_statas having less than Rs 10000.

```
Select bill_id,booking_id,total_amount
From billing_details,booking_details
Where bill_status=”pending”
Group by bill_id
Having total_amount<10000;
```

17. Stored Procedure 1: CALCULATE_LATE_FEE_AND_TAX

Given the return date and time while booking, actual return date and time, registration number of the car and booking amount, this procedure calculates the total late fee using late fee per hour, return date and time and the actual return date and time of the rental car. Once the total late fee is obtained, it is added to the amount and the total tax is calculated.

```
CREATE OR REPLACE PROCEDURE CALCULATE_LATE_FEE_AND_TAX (
actualReturnDateTime IN BOOKING_DETAILS.ACT_RET_DT_TIME%TYPE,
ReturnDateTime IN BOOKING_DETAILS.RET_DT_TIME%TYPE, regNum IN
BOOKING_DETAILS.REG_NUM%TYPE,  amount IN
BOOKING_DETAILS.AMOUNT%TYPE, totalLateFee OUT
BILLING_DETAILS.TOTAL_AMOUNT%TYPE, totalTax OUT
BILLING_DETAILS.TAX_AMOUNT%TYPE ) AS
lateFeePerHour CAR_CATEGORY.LATE_FEE_PER_HOUR%TYPE; hourDifference
DECIMAL(10,2);
 BEGIN
SELECT LATE_FEE_PER_HOUR INTO lateFeePerHour    FROM CAR_CATEGORY CC
INNER JOIN CAR C ON CC.CATEGORY_NAME =    C.CAR_CATEGORY_NAME
WHERE C.REGISTRATION_NUMBER = regNum;
 IF actualReturnDateTime > ReturnDateTime THEN      hourDifference := (TO_DATE
(TO_CHAR (actualReturnDateTime,    'dd/mm/yyyy  hh24:mi:ss'), 'dd/mm/yyyy  hh24:mi:ss')
```

```
- TO_DATE (TO_CHAR (ReturnDateTime, 'dd/mm/yyyy  hh24:mi:ss')  ,'dd/mm/yyyy
hh24:mi:ss'))*(24);
totalLateFee := hourDifference  *  lateFeePerHour;
 ELSE
  totalLateFee := 0;
 END IF;
totalTax := (amount + totalLateFee)  *  0.0825;
 END;
```

18. Stored Procedure 2: CALCULATE_DISCOUNT_AMOUNT

Given the driving license number, total amount and the discount code, this procedure calculates the discount amount based on the discount code used by the customer while booking the rental car. Additional 10% discount is given to the customers who have membership ID. The discount amount is calculated on the total amount obtained after adding the late fee and the tax amount.

```
CREATE OR REPLACE PROCEDURE CALCULATE_DISCOUNT_AMOUNT
 (dlNum IN CUSTOMER_DETAILS.DL_NUMBER%TYPE,
 amount IN BILLING_DETAILS.TOTAL_AMOUNT%TYPE,
discountCode IN DISCOUNT_DETAILS.DISCOUNT_CODE%TYPE,
 discountAmt OUT BILLING_DETAILS.DISCOUNT_AMOUNT%TYPE) AS
 --local declarations
 memberType CUSTOMER_DETAILS.MEMBERSHIP_TYPE%TYPE;
discountPercentage DISCOUNT_DETAILS.DISCOUNT_PERCENTAGE%TYPE;
 BEGIN
 SELECT MEMBERSHIP_TYPE INTO memberType FROM CUSTOMER_DETAILS
 WHERE DL_NUMBER = dlNum;
   IF NVL(discountCode,'NULL') <> 'NULL' THEN
  SELECT DISCOUNT_PERCENTAGE INTO discountPercentage
  FROM DISCOUNT_DETAILS WHERE DISCOUNT_CODE = discountCode;
  IF memberType = 'M' THEN
  discountAmt := amount * ((discountPercentage+10)/100);
```

```
 ELSE
 discountAmt := amount * (discountPercentage/100);
 END IF;
ELSE
IF  memberType = 'M' THEN
discountAmt := amount * 0.1;
ELSE
discountAmt := 0;
END IF;
END IF;
END;
```

19. Trigger 1: GENERATE_BILLING

This trigger inserts a tuple into the Billing_Details table when the actual return date is updated and booking status is updated to ‘R’ in Booking_Details table. It generates Bill when the rental car is returned. This is triggered whenever a row is updated in Booking_Details table.

```
CREATE OR REPLACE TRIGGER GENERATE_BILLING
 AFTER UPDATE ON BOOKING_DETAILS
FOR EACH ROW
 WHEN ( (TO_CHAR(NEW.ACT_RET_DT_TIME),'NULL') <> 'NULL' AND
NEW.BOOKING_STATUS ='R')
lastBillId BILLING_DETAILS.BILL_ID%TYPE;
 newBillId BILLING_DETAILS.BILL_ID%TYPE;
 discountAmt BILLING_DETAILS.DISCOUNT_AMOUNT%TYPE;
totalLateFee BILLING_DETAILS.TOTAL_LATE_FEE%TYPE;
 totalTax BILLING_DETAILS.TAX_AMOUNT%TYPE;
totalAmountBeforeDiscount BILLING_DETAILS.TOTAL_AMOUNT%TYPE;
 finalAmount BILLING_DETAILS.TOTAL_AMOUNT%TYPE;
BEGIN
SELECT BILL_ID INTO lastBillId FROM ( SELECT BILL_ID, ROWNUM AS
```

```
RN FROM  BILLING_DETAILS)
WHERE RN= (SELECT MAX(ROWNUM) FROM BILLING_DETAILS);
newBillId := 'BL' || TO_CHAR(TO_NUMBER(SUBSTR(lastBillId,3))+1);
CALCULATE_LATE_FEE_AND_TAX(:NEW.ACT_RET_DT_TIME,
:NEW.RET_DT_TIME,    :NEW.REG_NUM,:NEW.AMOUNT, totalLateFee, totalTax);
 totalAmountBeforeDiscount := :NEW.AMOUNT + totalLateFee + totalTax;
CALCULATE_DISCOUNT_AMOUNT(:NEW.DL_NUM, totalAmountBeforeDiscount,
:NEW.DISCOUNT_CODE, discountAmt);
 finalAmount := totalAmountBeforeDiscount - discountAmt;
INSERT INTO BILLING_DETAILS
(BILL_ID,BILL_DATE,BILL_STATUS,DISCOUNT_AMOUNT,
TOTAL_AMOUNT,TAX_AMOUNT,BOOKING_ID,TOTAL_LATE_FEE)
VALUES (newBillId,to_date(SYSDATE,'YYYY-MM-DD'),'P',
discountAmt,finalAmount,totalTax,:NEW.BOOKING_ID,totalLateFee);
END;
```

20. Trigger 2: UPDATE_CAR_DETAILS

This trigger updates the availability flag, mileage and location of the car in the car table when the actual return date is updated or when a booking is cancelled. This is triggered whenever a row is updated in Booking_Details table.

```
--This trigger updates the availability flag, mileage and location in the car table
 -- when the car is returned.

CREATE OR REPLACE TRIGGER UPDATE_CAR_DETAILS
 AFTER UPDATE ON BOOKING_DETAILS
FOR EACH ROW
WHEN ((TO_CHAR(NEW.ACT_RET_DT_TIME),'NULL') <> 'NULL' OR
NEW.BOOKING_STATUS ='C')
 DECLARE
 BEGIN
```

```
 IF :NEW.BOOKING_STATUS ='C' THEN
UPDATE CAR SET AVAILABILITY_FLAG = 'A' , LOC_ID = :NEW.PICKUP_LOC
WHERE        REGISTRATION_NUMBER = :NEW.REG_NUM;
ELSE
 IF NVL(TO_CHAR(:NEW.ACT_RET_DT_TIME),'NULL') <> 'NULL' THEN
UPDATE CAR SET AVAILABILITY_FLAG = 'A' , LOC_ID = :NEW.DROP_LOC,
MILEAGE = MILEAGE+GET_MILEAGE WHERE REGISTRATION_NUMBER =
:NEW.REG_NUM;
END IF;
 END IF;
END;
```

5. SYSTEM TESTING AND RESULT ANALYSIS

Testing is the process in which the system is run on manually created input so that the system is correctly working as desired or not. During systems testing, the system is used experimentally to ensure that software does not fail. In other words, we can say that it will run according to its specifications and in the way users expect. Special test data are input for processing, and the results examined.

A limited number of users may be allowed to use the system so that analyst can see whether they try to use it in unforeseen ways. It is desirable to discover any surprises before the organizations implements the system and depends on it.

Testing of a system is generally done in two phases – one is **Unit Testing** which is done for each module independently on its completion and the other one is **System Testing** which is done at the end of a project.

VALIDATION CRITERIA

The validation criteria in this project are as follows…

In portal system also, the user inputs are validated before storing them, and then further for displaying etc. The main validates that are done in portal **System** are as follows-

All the screens have similar look and feel. They all have the almost same color combinations in its background. This provides a better user interface to the users.

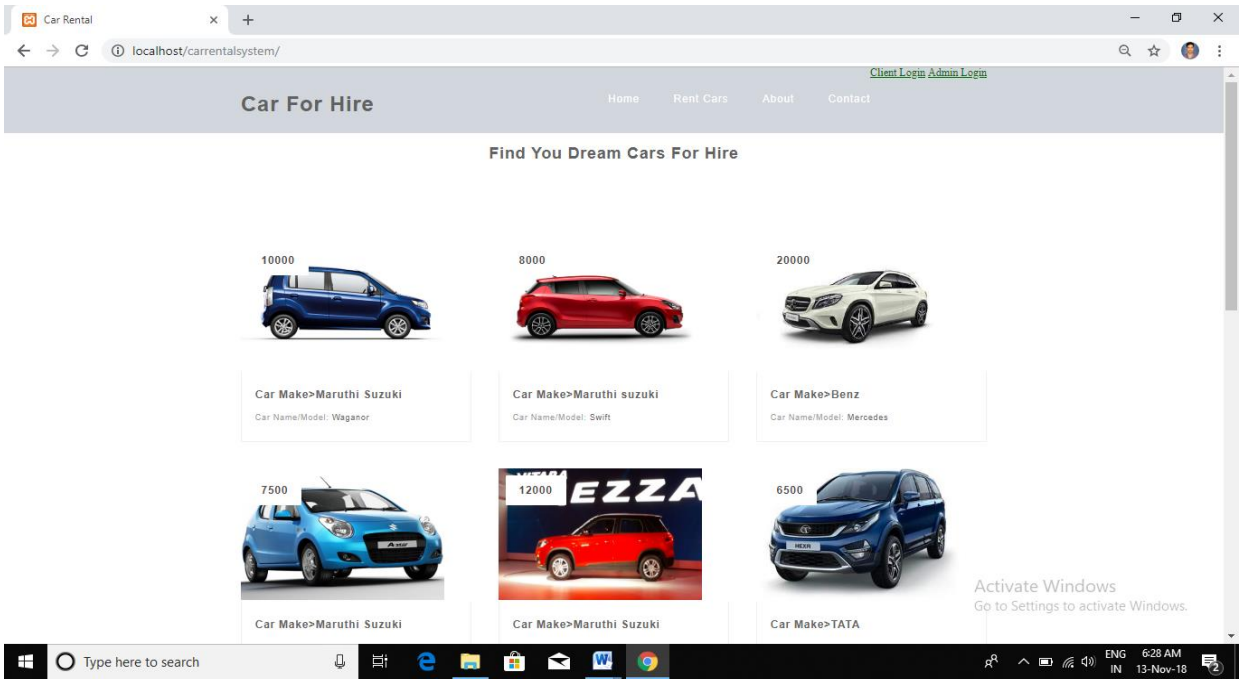
1) The primary key value cannot be duplicated.
2) All the entries in any combo box have been sorted in alphabetical order.
   This helps a user while selecting value from the combo box.

IMPORTANCE OF TESTING

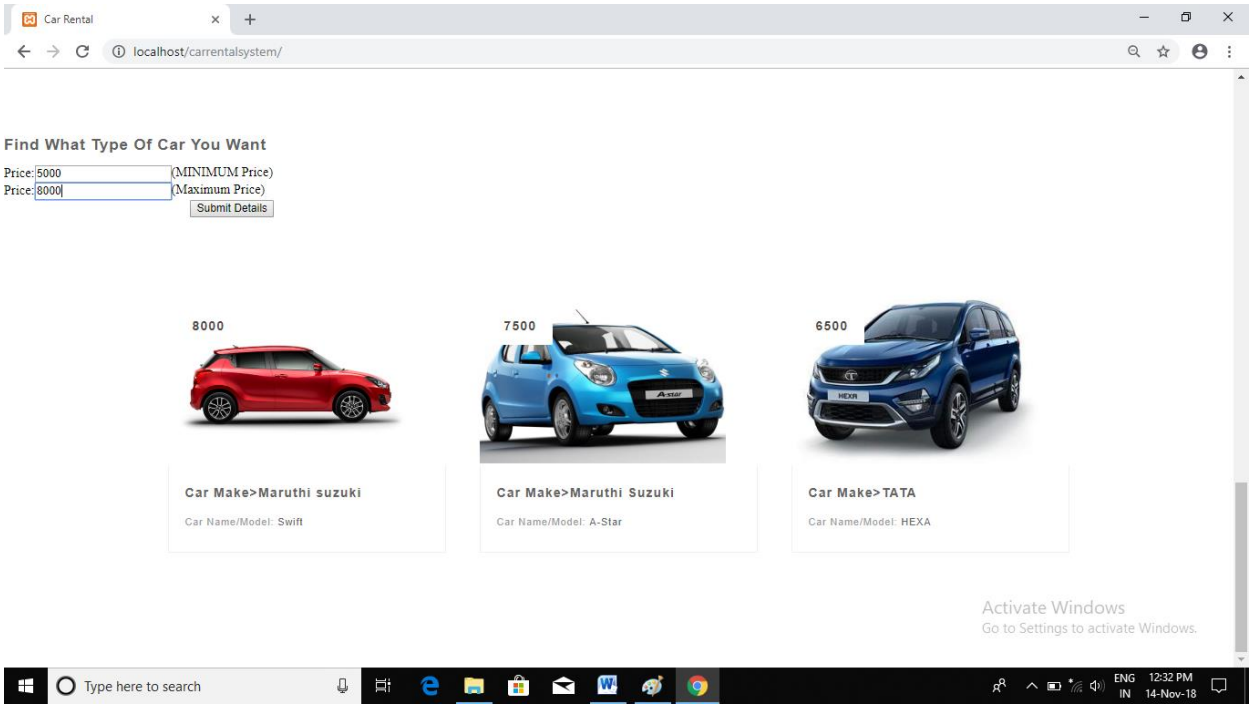
During systems testing, the system is used experimentally to ensure that the software does not fail. In other words, we can we can say that it will run according to its specifications and in the way users expect. Special test data are input for processing, the results examined.

The importance of system testing is that the system is expected to run according to member's requirement delivering it to the customer. The system is tested on the basis of specification so that it does not fail on user site.

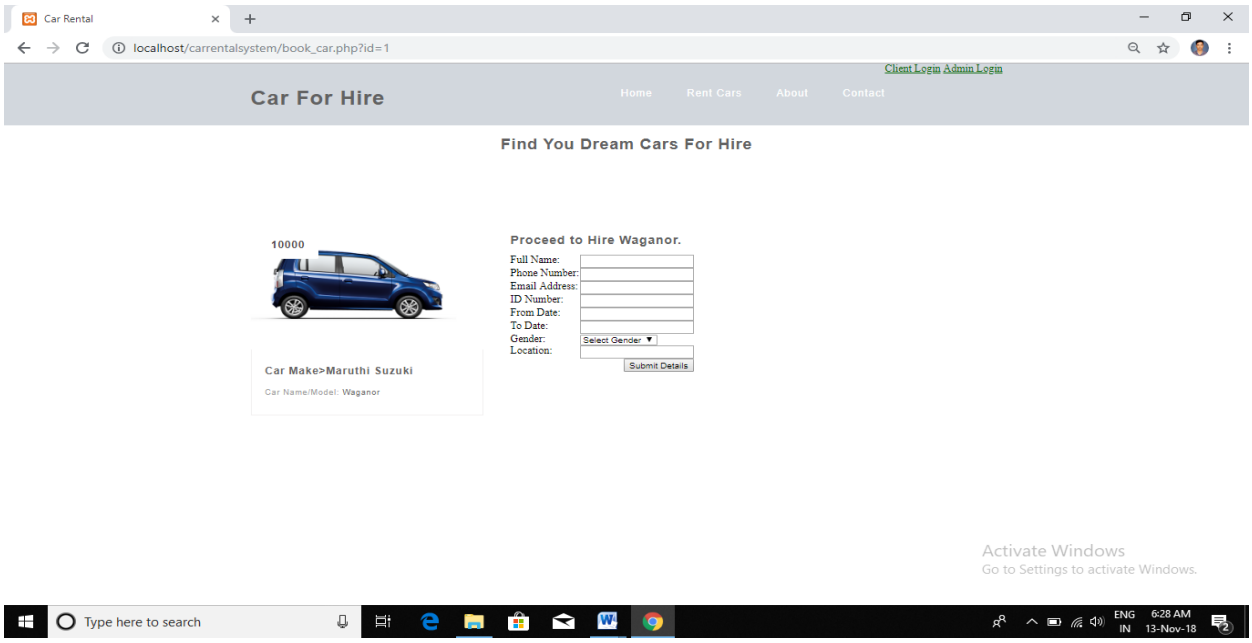
The system is checked with respect to several aspects and the screenshots of those test results are attached below:



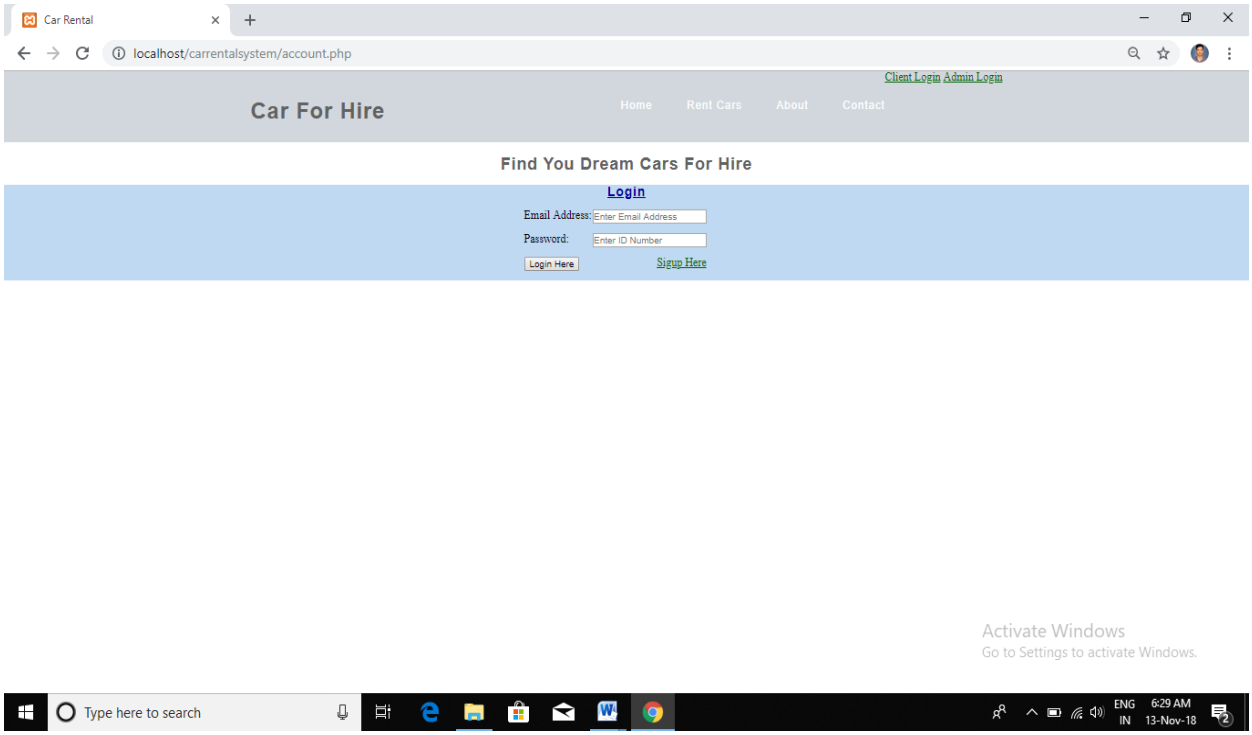
This is the first page of our simple and yet effective application. It has a list of options of different cars with its information and hire cost of the car. And also an option for user and admin to login. When the car image is clicked it goes to next page.



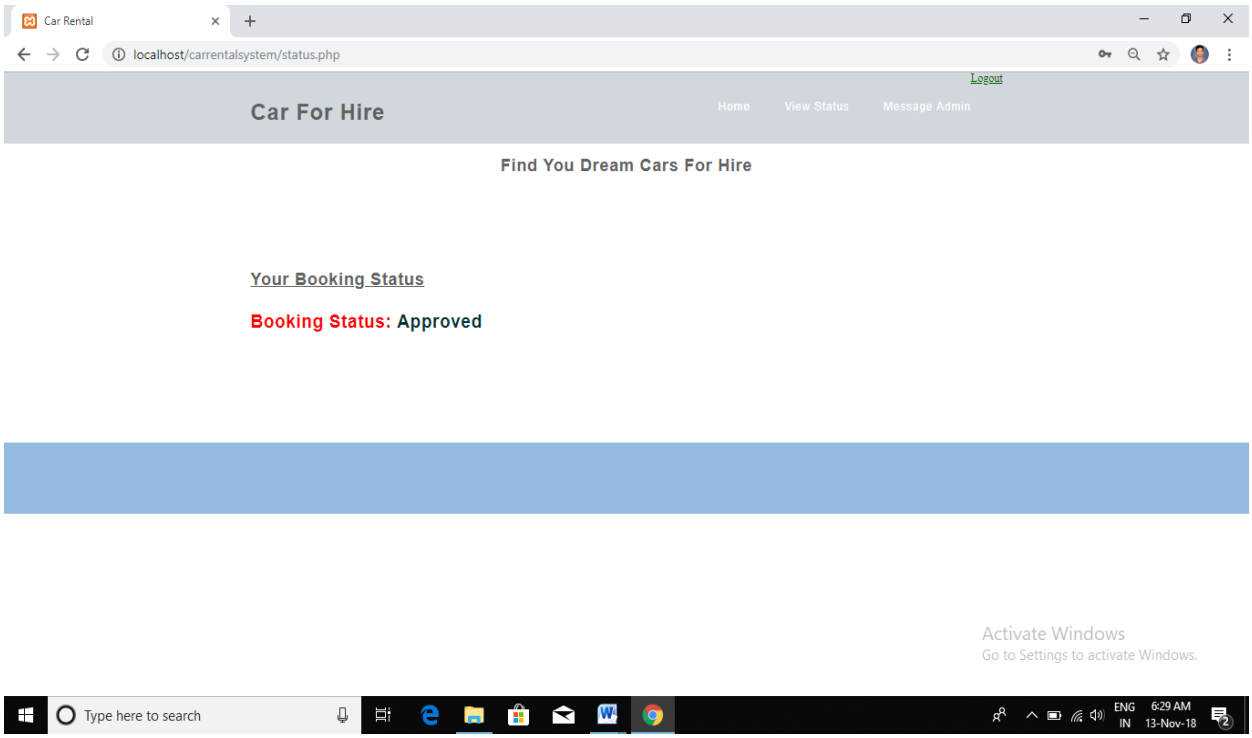
This is also first page in this search button is available to find the car based on the your range.



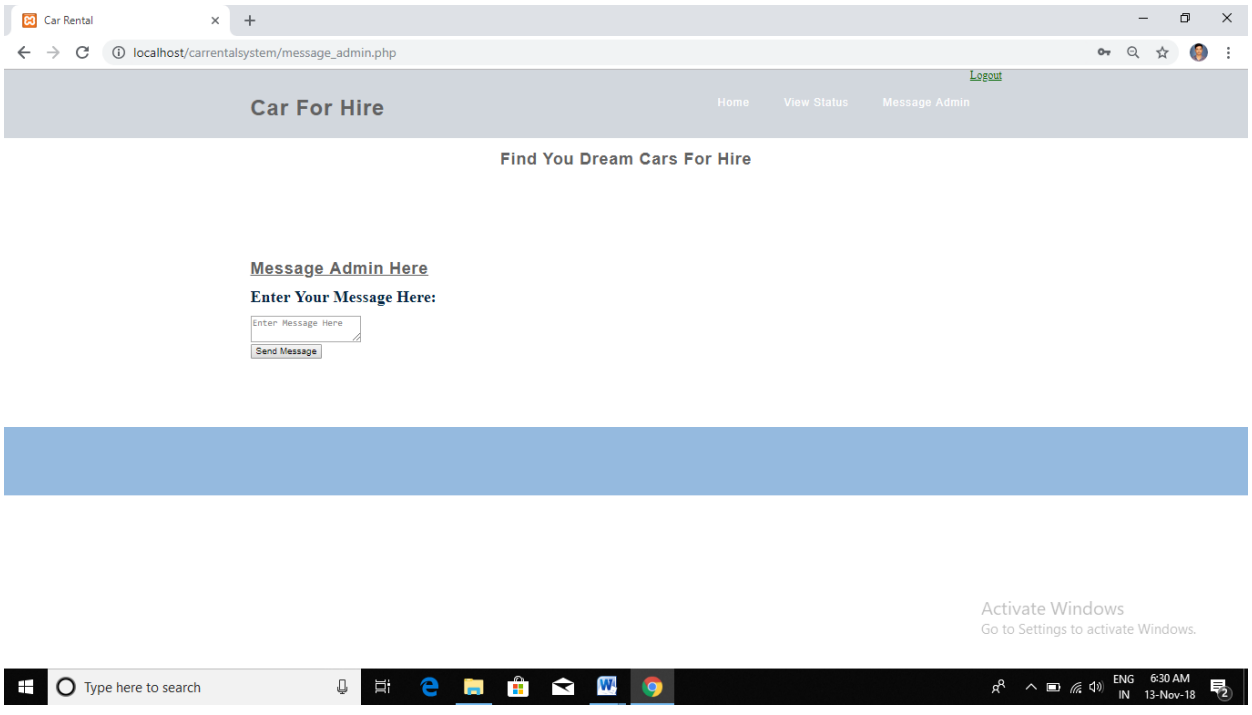
In this page user can fill the information about himself.



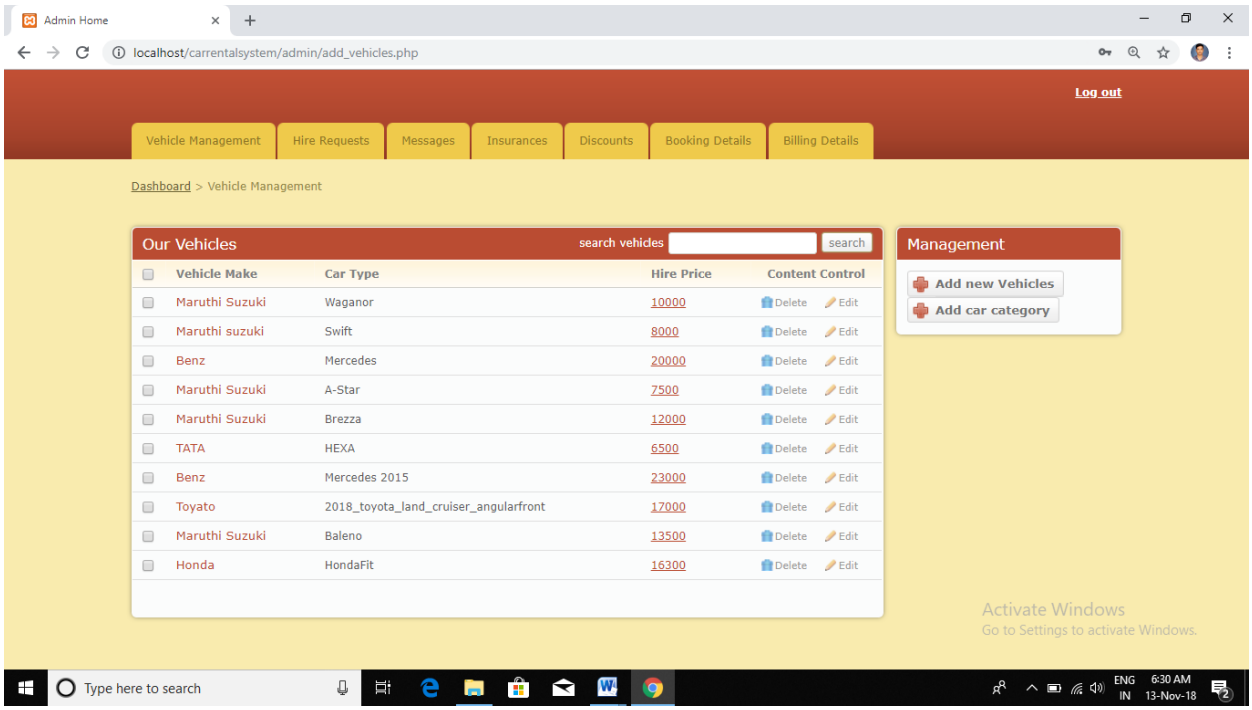
This figure depicts the user login page and sign up button.



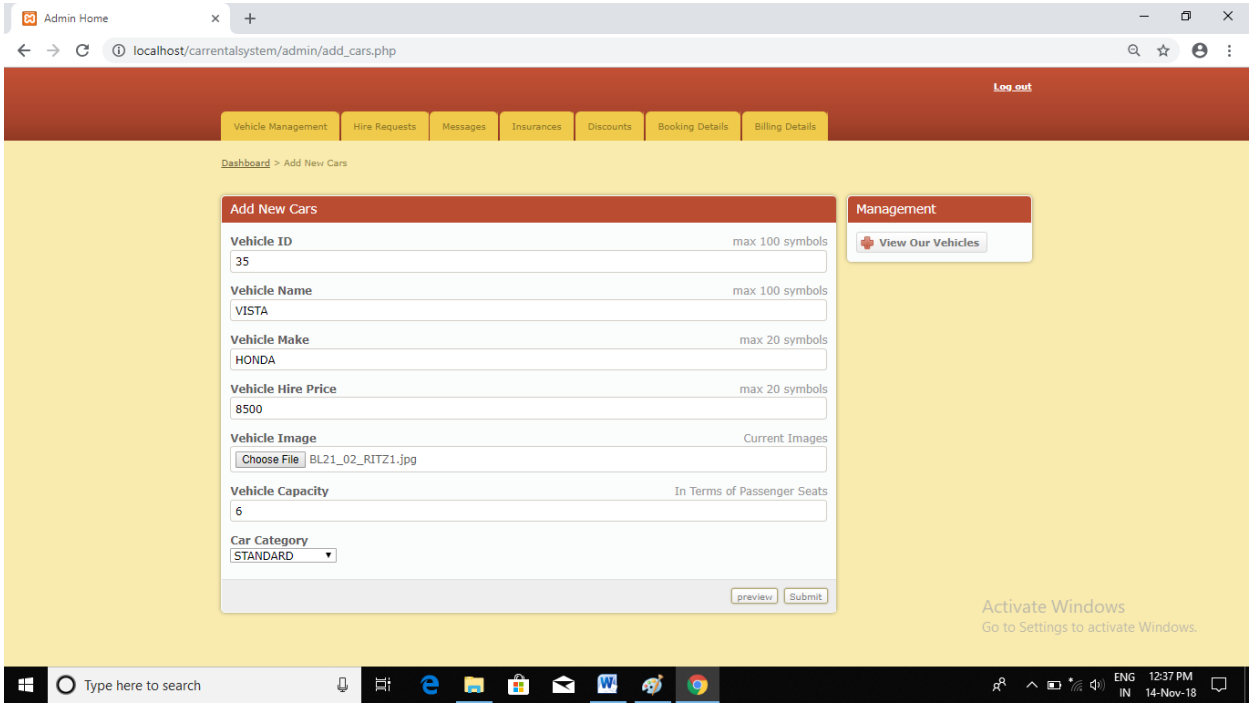
This figure depicts status of the hire request.



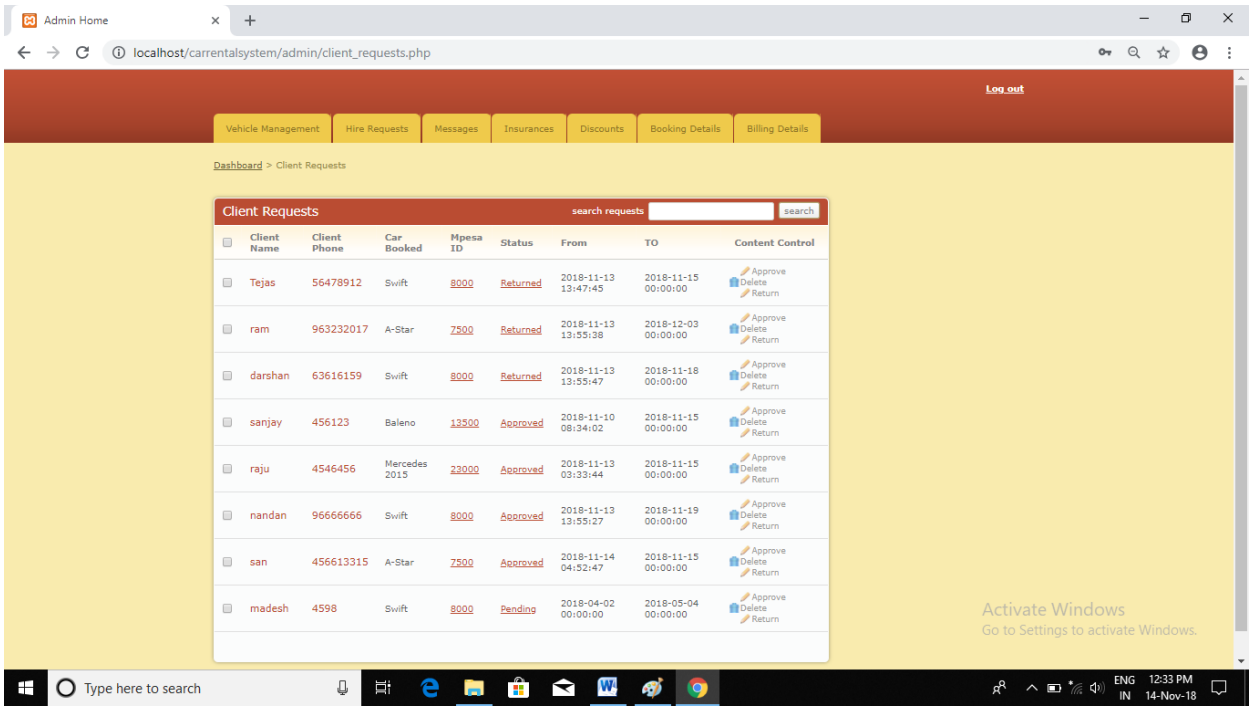
In this Figure the user can able to send the messages or feedback to the admin.



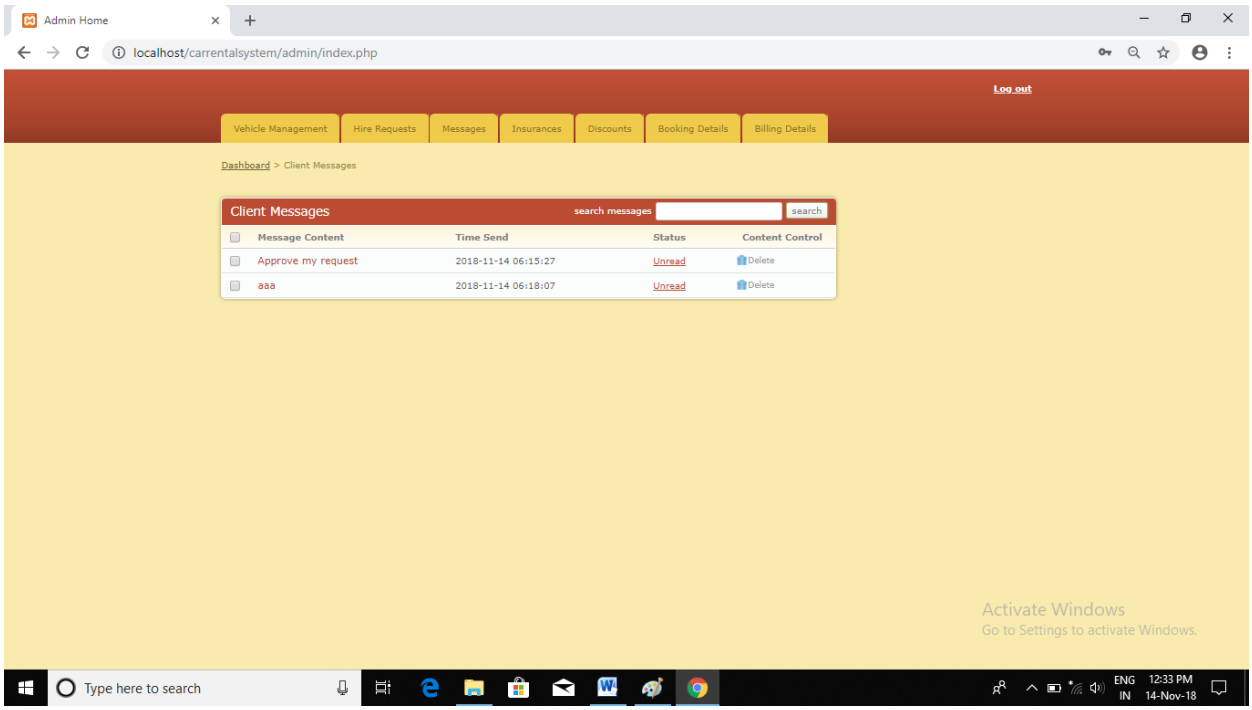
This is admin side web page, this table displays the available car information in this company.



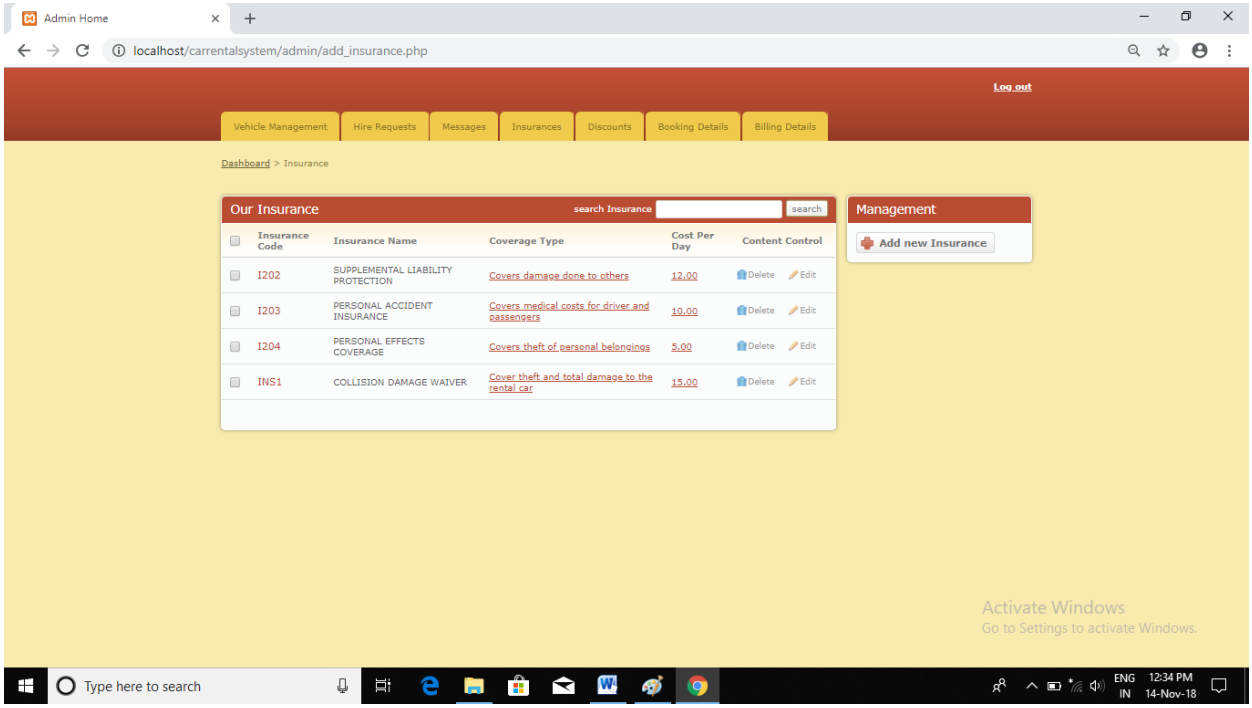
This is admin view, in this table we add new car information with its image and click submit button. The car is inserted to the car database table and it also shown in user view.



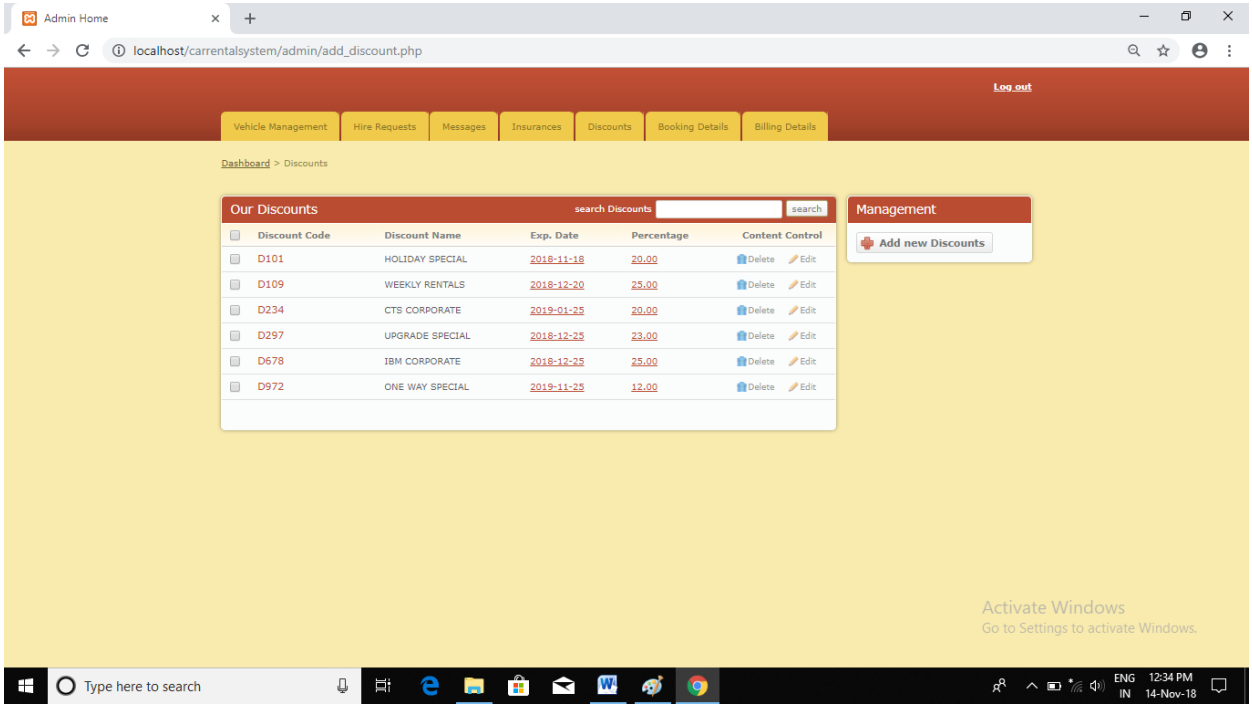
This table displays the hire requests and its status, in this page admin can able to approve the requests.



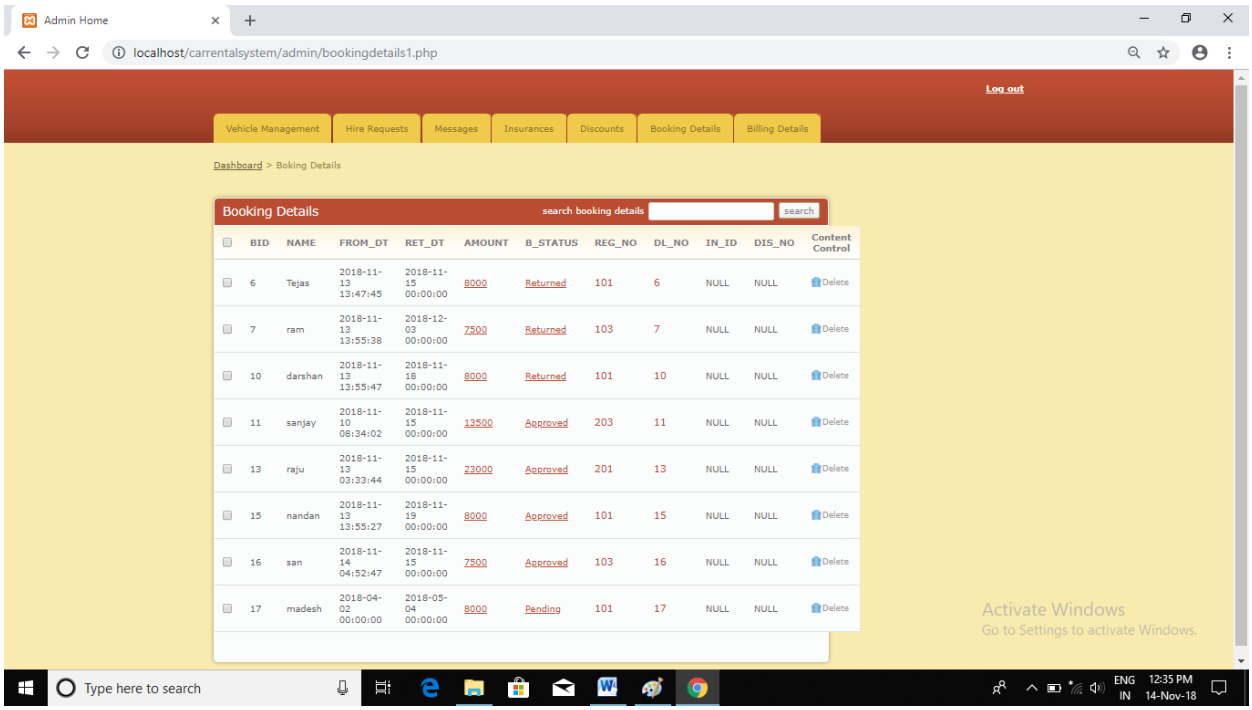
This page is admin view, in this page the client sended messages or feedback are available in this page.



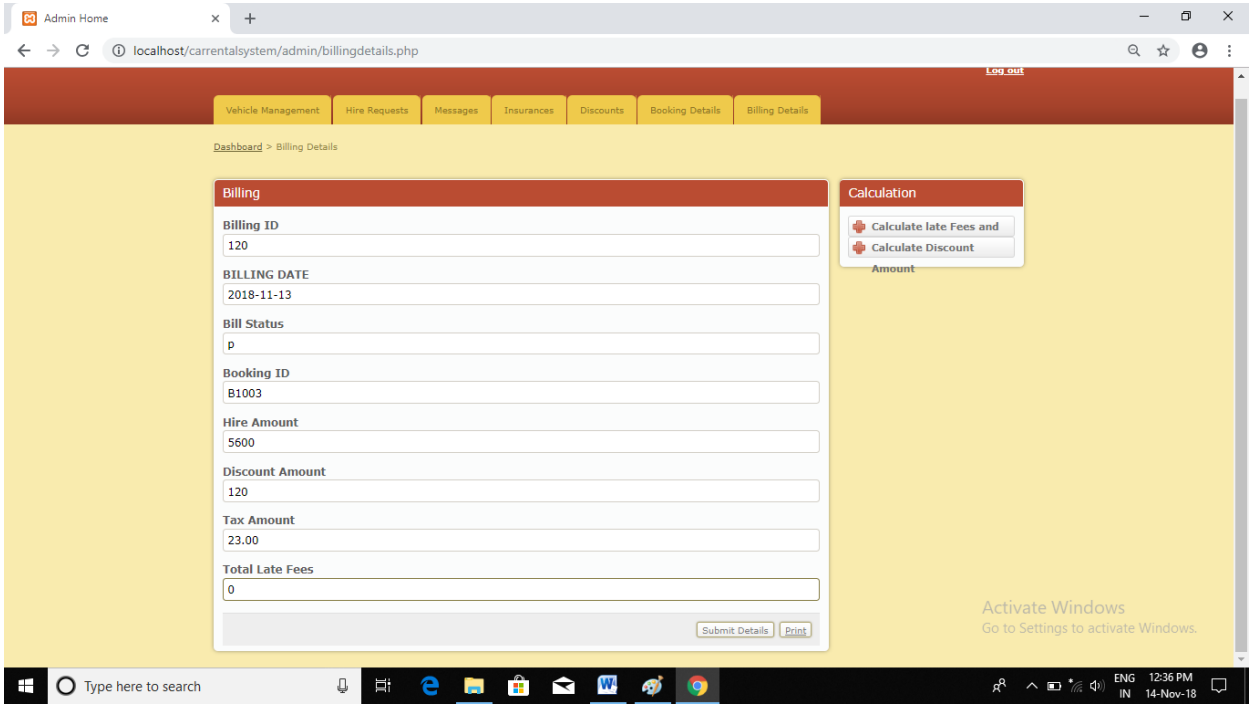
This is insurance table, it shows available insurances. Then we want to add new insurance by using ‘Add new insurance’ button.



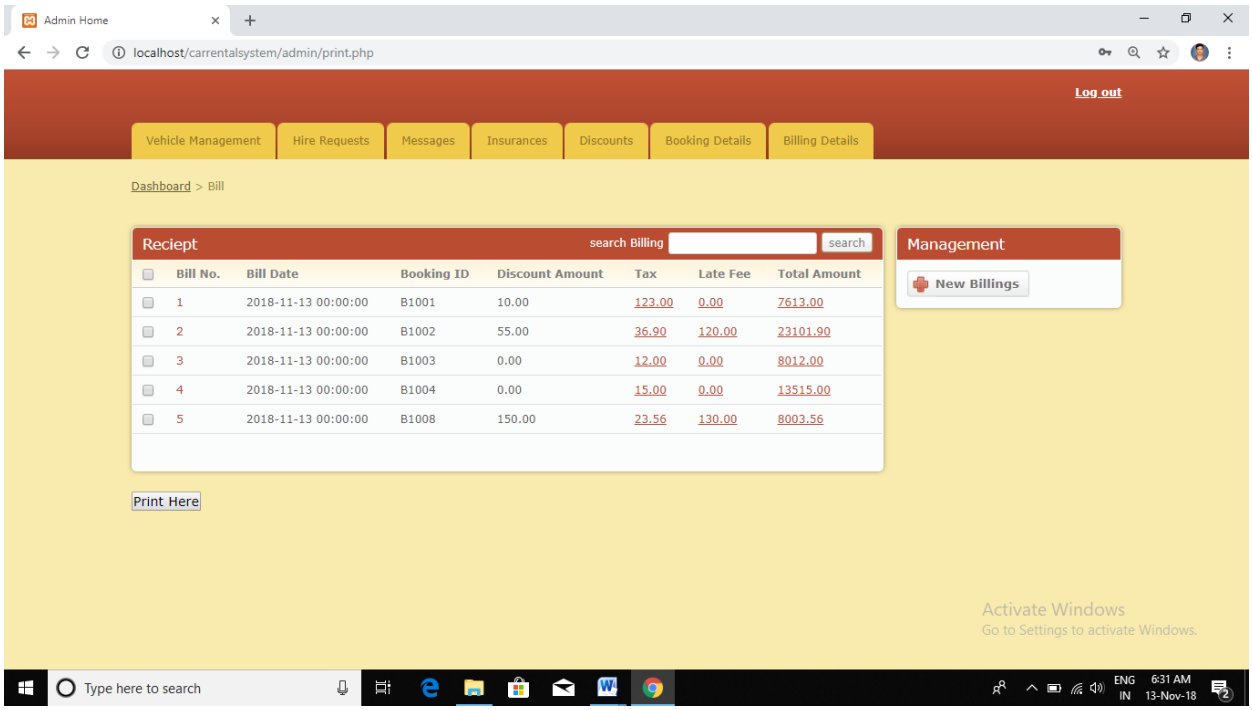
This is discount table, it shows available discounts. Then we want to add new discount by using ‘Add new discount’ button.



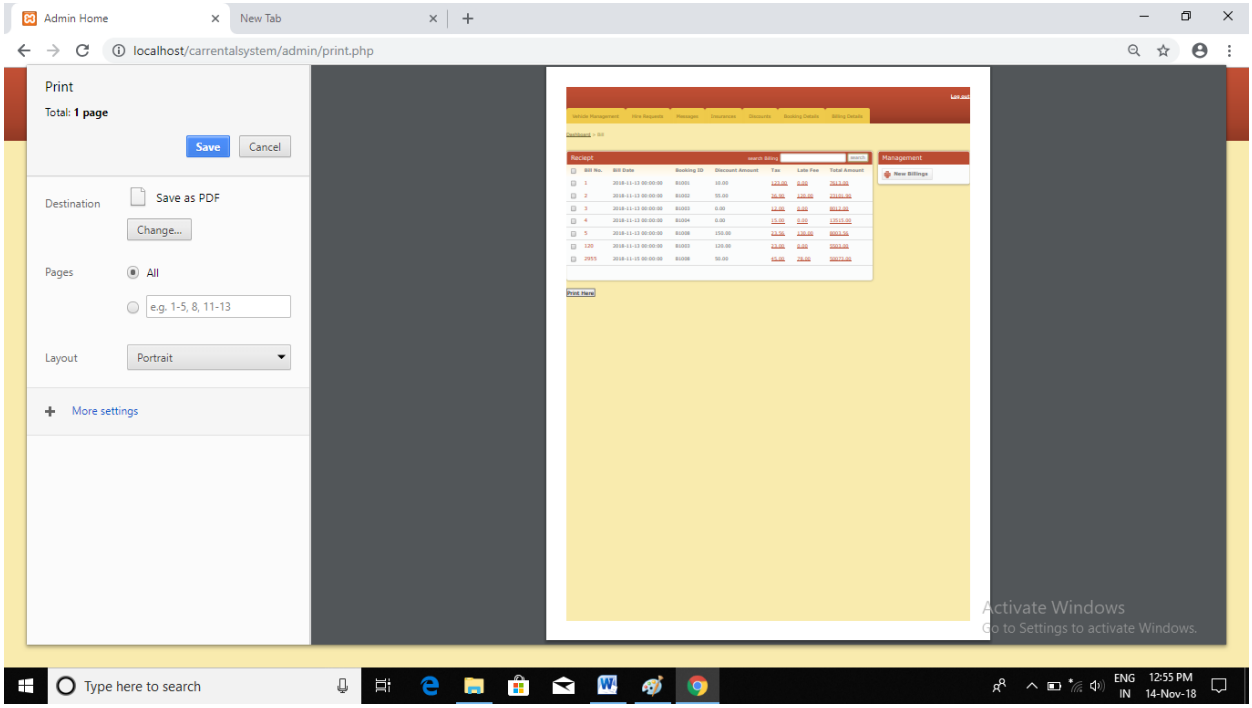
This table displays the booking details and its status.



In this page, we generate billing and to calculate the late fees and discount amount by clicking the ‘calculate late fees’ and ‘calculate discount’ amount.



This table displays the billing details and we can also print the billings using ‘print here’ button. It shown in the below picture.



6. CONCLUSION

Car rental business emerged with a new goodies compared to the past experience where every activity concerning car rental business is limited to a physical location only. Even though the physical location has not been totally eradicated; the nature of functions and how these functions are achieved has been reshaped by the power of internet. Nowadays, customers can reserve cars online, rent car online, and have the car brought to their door step once the customers is registered member or go to the office to pick the car.

The web based car rental system has offered an advantage to both customers as well as Car Rental Company to efficiently and effectively manage the business and satisfy customers need at the click of button.

The projects minimizes the use of books and is easily accessible by just clicking on the button. The project can be maintained for life long purpose as the data can be edited, deleted and added if required. If basically aims at low cost. most durable, secure and reliable use experience. The project is done as efficient as possible.

7. REFERENCES

- Database Management System - Ramez Elmasri , Shamkant B.Navathe
- https://devzone.zend.com/6/php-101-php-for-the-absolute
- http://php.net/manual/en/tutorial.php
- https://stackify.com/learn-php-tutorials/
- https://www.w3schools.com/php/php_mysql_intro.asp
- Learning PHP, MySQL, JavaScript, and CSS: A Step-by-Step Guide to Creating Dynamic Websites – by Robin Nixon
- "Database System Concepts" by Abraham Silberschatz and S Sudarshan