CS631-103

DATA MANAGEMENT SYSTEM DESIGN

PROJECT REPORT

PROJECT: RENT A CAR

GROUP NO-06

TEAM MEMBERS:-

MOHAMMED ZAHOORUDDIN ZOHAIB-ZM254

SRI HARSHA MALYA C-SC2545

TABLE OF CONTENTS

INTRODUCTION

A web application is a schedule of applications that is compiled by a remote server and generated via the internet through a browser interface. The reservation and rental of vehicles is managed through the car rental system, a computerized system. In order to manage their fleets of automobiles and trucks and offer their clients rental services, car rental firms often use it. A computerized database with all of the company's vehicles, client information, rental agreements, and other data often makes up the system. The system is made to offer effective fleet management and control, as well as a simple and convenient rental booking and payment interface for consumers. Additionally, the system often has features like automatic billing, fleet tracking, and online reservation and payment.

PART 1

SYSTEM DESIGN

This section of the application shows that the scheme's precise design has been finished. The general method for this assignment was created using the program. The construction of the procedure model approach depends on this procedure's particular task being designed. To finish the web application CAR RENTAL SYSTEM operation, PHP, HTML, CSS, and SQL were used.

The argument is correctly given to protect the link between the database's front back and finishing date in this program, which is composed of manifestations. The application of crud is related to the area of computer science that encourages the creation and application of centralized storage. To create, access, and edit this application, the crud application must be implemented.

NORMALISATION TABLE

Billing_amt

```
CREATE TABLE BILLING_DETAILS
( BILL_ID CHAR(6) NOT NULL,
 BILL_DATE DATE NOT NULL,
 BILL_STATUS CHAR(1) NOT NULL,
 DISCOUNT_AMOUNT NUMBER(10,2) NOT NULL,
 TOTAL_AMOUNT NUMBER(10,2) NOT NULL,
 CONSTRAINT BILLINGPK
 PRIMARY KEY (BILL_ID)
);
INSERT INTO "BILLING_DETAILS" values ('abc','040401','s','10','99');
INSERT INTO "BILLING_DETAILS" values ('DEF','040601','F','16','199');
INSERT INTO "BILLING_DETAILS" values ('GHI','051001','F','25','149');
INSERT INTO "BILLING_DETAILS" values ('IJK','111101','s','30','129');
INSERT INTO "BILLING_DETAILS" values ('LMN','061201','s','45','1799');
```

Functional Dependancy:

select * from billing_amt;

bill_id -> bill_date, total_amount, discount, bill_status

Above relation is not in 3NF as there is transitive dependency.

So the above table is decomposed into:

billing_amount (bill_id, bill_date, total_amount, bill_status)

total_amount -> discount

bill_id_total_amount (discount)

Above relations is in 3NF because there is no transitive dependency, i.e x -> y such that x is super key.

Customer

```
CREATE TABLE CUSTOMER
( LICENSE_NO CHAR(8) NOT NULL,
 NAME VARCHAR(25) NOT NULL,
 ADDRESS VARCHAR(30) NOT NULL,
 CONSTRAINT CUSTOMERPK
 PRIMARY KEY (LICENSE_NO)
```

);

INSERT INTO customer VALUES (“john','dales ave’,’000111’);

INSERT INTO customer VALUES(‘smith’,'logan ave’,’111222’);

INSERT INTO customer VALUES(‘mama’,'sip ave’,’222333');

INSERT INTO customer VALUES(‘leroy,’broadwav’,'333444');

INSERT INTO customer VALUES(‘mark’,’newark’,’444555’);

SELECT * FROM customer

Functional Dependancy:

license_no -> name, address

Above relations is in 3NF because there is no transitive dependency, i.e x -> y such that x is super key

---

Rental_loc

```
Create TABLE RENTAL_LOC(
   Rental_loc_id int,
   Primary Key(Rental_loc_id),
   Phone int,
   Street_Name Varchar(20),
   State Varchar(20),
   Zipcode int
   );
INSERT ALL
  INTO RENTAL_LOC VALUES (1,123,'KINGS AVENUE','NJ',07029)
  INTO RENTAL_LOC VALUES (2,456,'CROSS AVENUE','NJ',07029)
  INTO RENTAL_LOC VALUES (3,789,'CLEVELAND AVENUE','NJ',07029)
  INTO RENTAL_LOC VALUES (4,147, 'HARRIS AVENUE','NJ',07029)
  INTO RENTAL_LOC VALUES (5,258,'REYNOLDS AVENUE','NJ',07029)
SELECT * FROM RENTAL_LOC;
```

Funtional dependency:

rental_location_id -> phone, state, street_name, zipcode

Above relation is not in 3NF as there is transitive dependency.

rental_location_id -> phone, state

state -> street_name, zipcode

So the above table is decomposed into:

rental_location (rental_location_id, phone, state)

rental_location_state (street_name, zipcode)

Reservation

```
CREATE TABLE reservation(
 reservation_id VARCHAR(100) not null primary key,
from_date_time VARCHAR(100),
return_date VARCHAR(256),
license_no int references customer(id)
  );
INSERT INTO reservation VALUES('123456','12:00 05nov','20:48 10nov');
INSERT INTO reservation VALUES('456789','12:30 05jan','20:58 15jan');
INSERT INTO reservation VALUES('789012','12:15 05dec','20:22 20dec');
INSERT INTO reservation VALUES('345678','12:18 15nov','20:17 28nov');
INSERT INTO reservation VALUES('000000','12:28 20nov','20:02 18dec');
SELECT * FROM reservation;
```

Funtional Dependency:

reservation_id -> from_date_time, return_date_time, license_no

Above relations is in 3NF because there is no transitive dependency, i.e x -> y such that x is super key.

Car

```
CREATE TABLE car(
 VIN VARCHAR(100) not null primary key,
company VARCHAR(100),
car_year int,
  rental_location_id int references rental_loction(id)

);
INSERT INTO car VALUES('MIJ926','FORD','2014',15678);
INSERT INTO car VALUES('ZMU455','HONDA','2016',15789);
INSERT INTO car VALUES('MZU789','HYNDAI','2018',15786);
INSERT INTO car VALUES('MSUZ221','MERCEDES','2020',15999);
INSERT INTO car VALUES('ZA114','BMW','2022',15001);
```

SELECT * FROM car;

Funitonal Dependency:

VIN -> company, car_year, rental_location_id

Above relations is in 3NF because there is no transitive dependency, i.e x -> y such that x is super key.

Car_rent

CREATE TABLE car_rent(

weekly int,

daily int

);

INSERT INTO car_rent VALUES('2500','1000');

INSERT INTO car_rent VALUES('3500','1500');

INSERT INTO car_rent VALUES('4500','2000');

INSERT INTO car_rent VALUES('5500','2500');

INSERT INTO car_rent VALUES('6500','3000');

SELECT * FROM car_rent;

Model

CREATE TABLE model(

model_no varchar[50] primary key,

model_name varchar[50]

);

INSERT INTO model VALUES('A2250','awd');

INSERT INTO model VALUES('C5555','fwd');

INSERT INTO model VALUES('Z555','4*4');

INSERT INTO model VALUES('H888','fwd');

INSERT INTO model VALUES('U7893','awd');

SELECT * FROM model;

Funtional Dependency:

model -> model_name

Above relations is in 3NF because there is no transitive dependency, i.e x -> y such that x is super key.

CRUD APPLICATION

The four fundamental operations of persistent storage are CRUD operations, which stand for Create, Read, Update, and Delete. These procedures would be used to manage the cars and users hiring them in the context of a car rental system.

Create:

The system can add new automobiles and users with this function. Either manually entering the data into the system or importing it from another source could be used to do this.

Read:

Using this operation, the system is able to search for and recover previously stored data. This can entail looking for a certain vehicle or discovering which users have used the system to rent vehicles.

Update:

Using this action, the system is able to change data that has already been stored there. This can involve changing the make and model of an automobile or a user's address or other contact information.

Delete:

This operation enables the system to erase previously stored data. This can entail removing a vehicle from the database or removing a user who is no longer renting vehicles.

CREATE

Users can add a document to the database using the begin function. The create operation is another name for the insert method. Recognize that the word specific is a column and that a memorandum is a row. It uses the "create" method to add new data to the database in this particular subset. The "Database" mysql application server is constructed using the "Insert" option choices. The PHP system refers to this application as trying to create. The management is in charge of updating the database with new properties. This program's user must add new rows by manually entering each piece of data.

READ

The read operation and the tracking function are analogous. This enables users to read the importance of the vast material in the table as well as to explore and recover it. Reading is a built-in feature that is used to download and check up a variety of various record keeping. By looking for distinct entries, implementing sentences, or attempting to meet other needs of a particular application, the information has been represented. By using keyword phrases or customizing their selection criteria, users may be able to fund the desired record.

UPDATE

The update procedure can be used to update the tracking records in the database. As a result, the record has been completely altered, and users must start updating the data for each province. As a result, both the conventional database records and all of the specific importance must be altered. The verify that it performs is used to update database records. To fully edit a record, users could be necessary to make changes to the data in a number of fields. The list may have been created, for example, by a company that keeps track of the details of menu items. . As a result, in addition to determining the features of the new car, the database item that already exists is also used, along with all possible values for the attributes.

DELETE

A delete operation is a procedure used to remove data or records from a car rental system. This can involve erasing client data, rental data, vehicle data, payment data, or any other information pertaining to the car rental system. Delete operations are frequently employed when data is outdated or inaccurate. For instance, if a consumer cancels their rental, the system would erase their information and all related data.

ENTITY RELATIONSHIP DIAGRAM

An entity relationship diagram (ERD) for an automobile rental system shows the relationships between the system's entities (people, places, and things). Customers, rental automobiles, rental locations, and payment options are some of the system's entities. Rental agreements connect consumers to rental cars and rental locations, rental payments connect customers to payment options, and automobile maintenance form the relationships between the parties (which link rental cars to rental locations). The ERD can be used to examine how various entities interact as well as assist in designing a database for the automobile rental system.

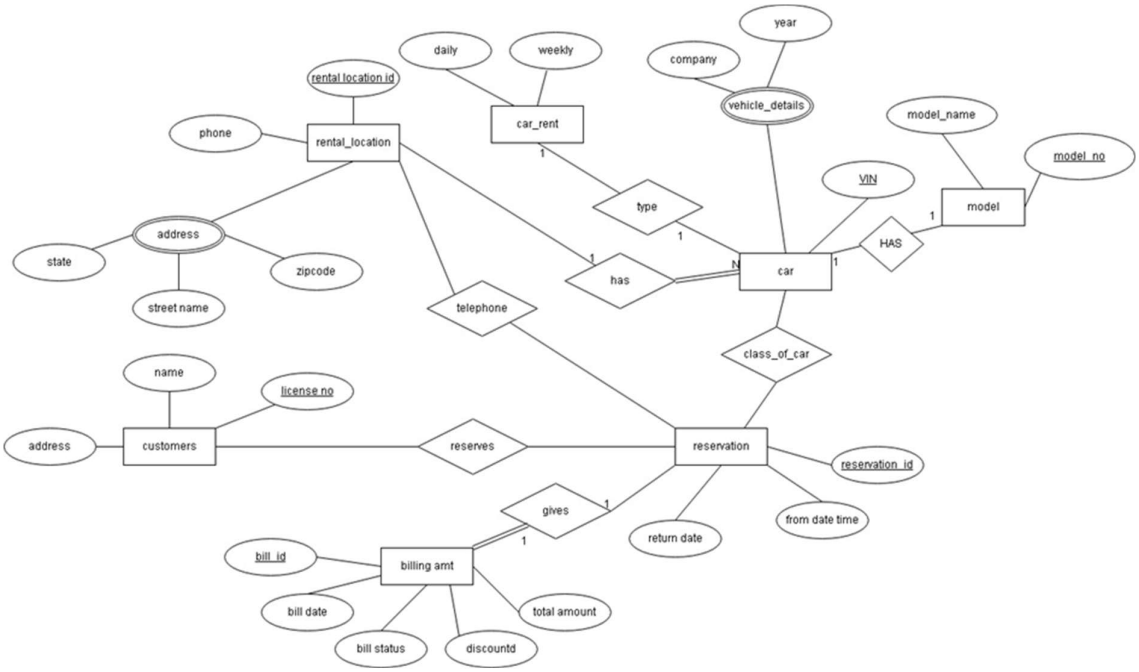

FIG 1: ENTITY RELATIONSHIP DIAGRAM

When constructing a database of information, an entity relationship diagram is crucial. ER diagrams that pick the data can be used to amass commodities and their individual properties.

To finish the full software, a number of procedures and task kinds were used. Designing the system's unique structure is the initial step in this implementation. The complete system in this project is built using the programs. A system design process could be developed on its own without the specialized tasks of system design development. The application for the website was built entirely from the ground up using PHP, HTML, CSS, and SQL. The project's database was designed on the "Xampp" server.

RELATION SCHEMA DIAGRAM

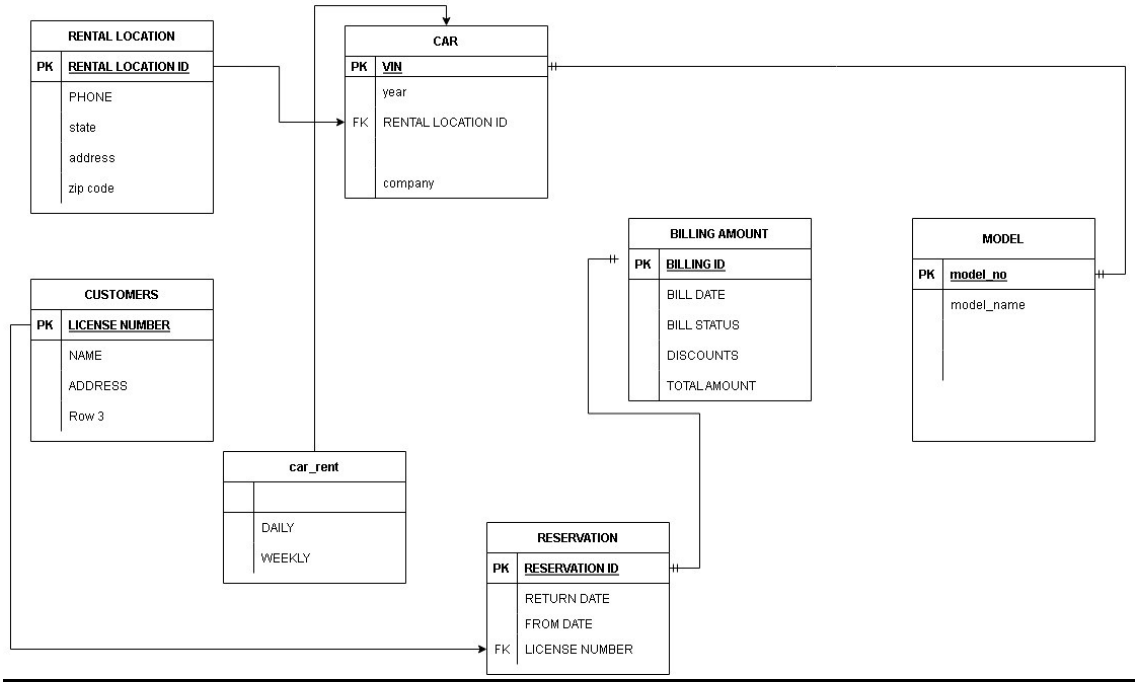

FIG2. RELATION SCHEMA DIAGRAM

A relational schema is also a data model representing a database structure logically. A relational schema uses tables to show a relationship between two or more entities.
Each table in a relational schema is referred to as a relation. Rows in the table are called tuples, whereas the table columns are attributes. Tuples can be seen as the instances of an entity. A table can have many instances.
A table can have multiple columns or attributes. Each attribute also has a domain that specifies or limits the value an attribute can take.

There are mainly two types of keys:

Primary key: A primary key is an identification attribute of each instance within a table. For the same reason, it can not have null or duplicate values.

Foreign key: A foreign key links two tables in a relational model. It refers to a field in a table that is the primary key of another table. It can be a single attribute or a set of attributes.

PART-2

SYSTEM DEVELOPMENT

USER INTERFACE(UI)

FIG 3: HOME PAGE

A car rental system's home page might provide details on the various vehicles available for rent, the many places where they can be rented, the cost of doing so, and any special offers or discounts. It might also have connections to pages with more in-depth details about the various automobiles, instructions for making a reservation, and a page with frequently asked questions. Links to the company's social media pages, customer evaluations, and ratings could all be found on the home page. The contact details for the car rental company should also be on the home page.

ACCOUNT LOGIN

FIG 4: LOGGING IN THE ACCOUNT

The credentials that clients must enter in order to access a car rental system are known as user login details. A username and password are typically required to log in to a car rental system. While the password is typically a combination of letters, numbers, and/or special characters that the client has chosen, the username is typically an email address that the customer has provided to the automobile rental business. Customers can access the car rental service and make, change, and cancel reservations for rentals by entering the proper user login information.

VERIFICATION OF LOGIN CREDENTIALS

The login information is compared with the database stored in phpmyadmin and cross-referenced back to the login page, which then redirects to the home page and displays a welcome dialog box.

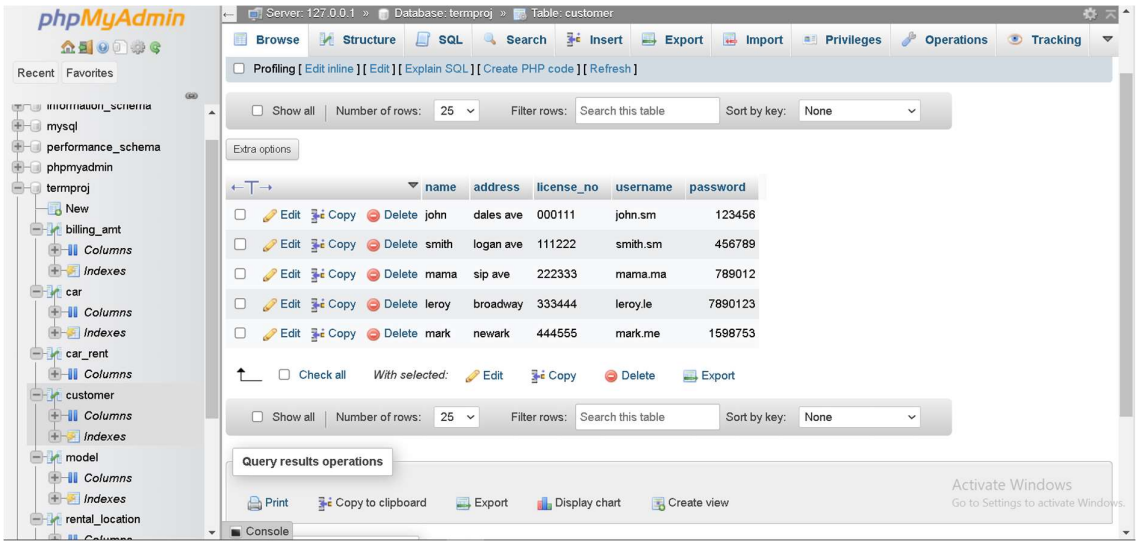

FIG5: DATABASE

FIG6: VERIFYING CREDENTIALS AND LETTING THE USER IN

RESERVING A CAB

As you can see, there are many different cars to choose from, and they are all available. At the bottom, you may choose which car you want explicitly, for how many days, from when to, when, and where your pickup will be.

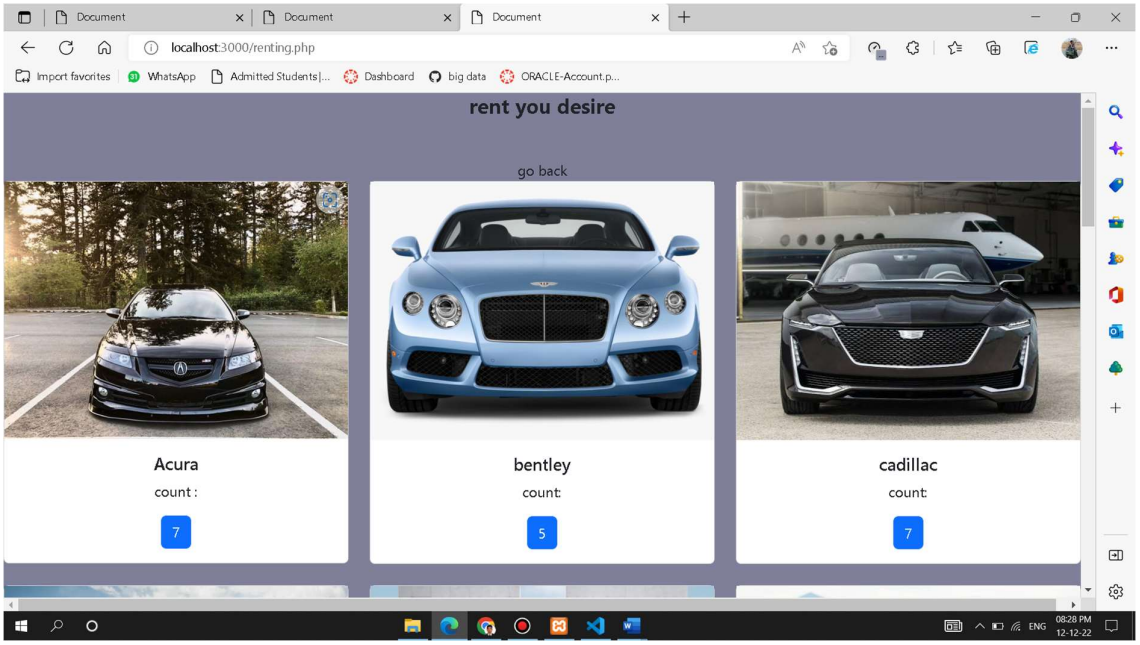

FIG7: RESERVING CAB 1

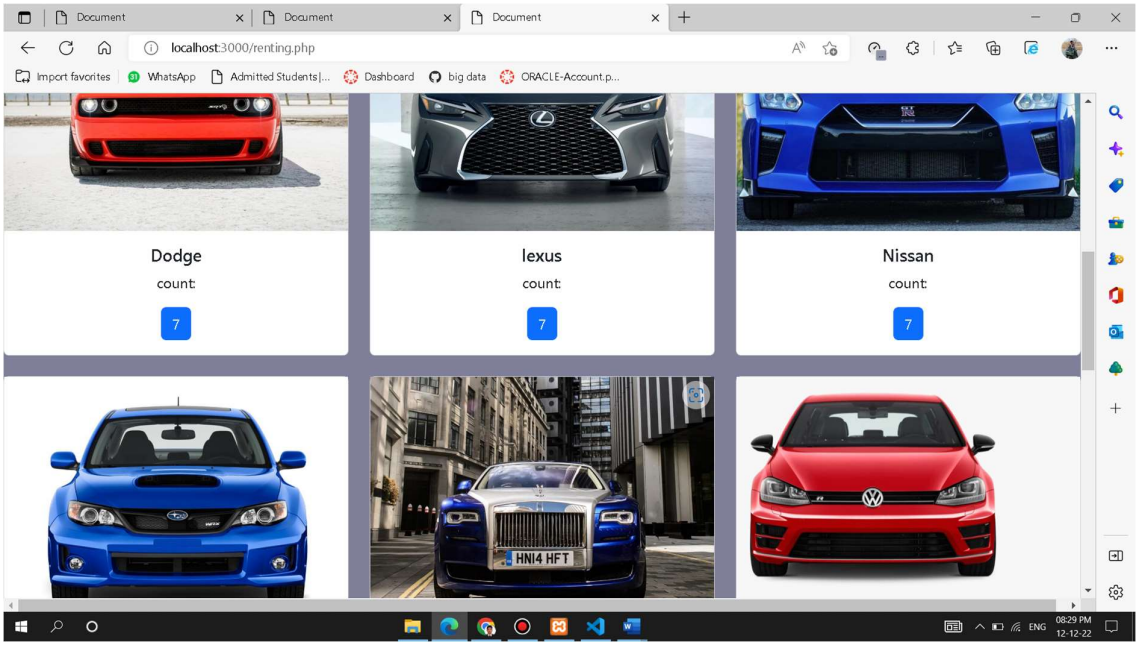

FIG8: RESERVING CAB 2

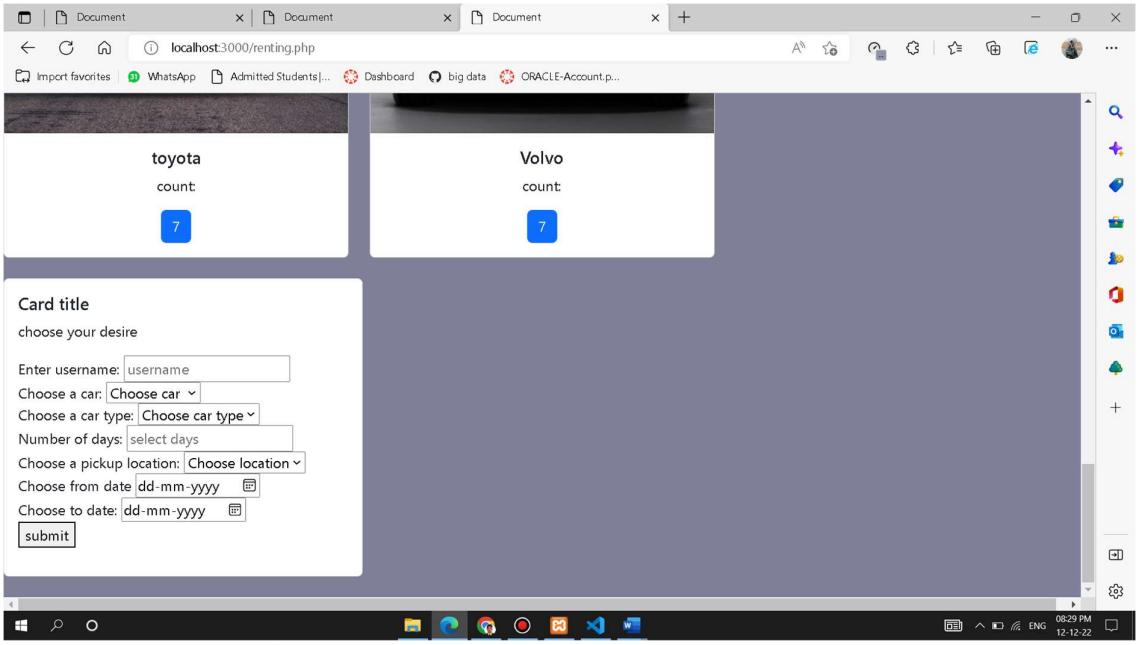

FIG9: RESERVING CAB 3

QUERY1

The GROUP BY clause is a clause in the SELECT statement. It allows you to create groups of rows that have the same value when using some functions (such as SUM, COUNT, MAX, MIN, and AVG).

Grouping is needed when working with aggregate functions.

The SQL GROUP BY clause allows you to specify the columns to group by.

The HAVING clause allows you to filter records after the GROUP BY is applied.

The group by clause's implementation is shown in Figure below. As you can see, the print command informs you of the query that is being run.

With a condition to group by and a condition that total amount is larger than 1, the query extracts bill date, bill id, total amount, and bill status from the billing amount table

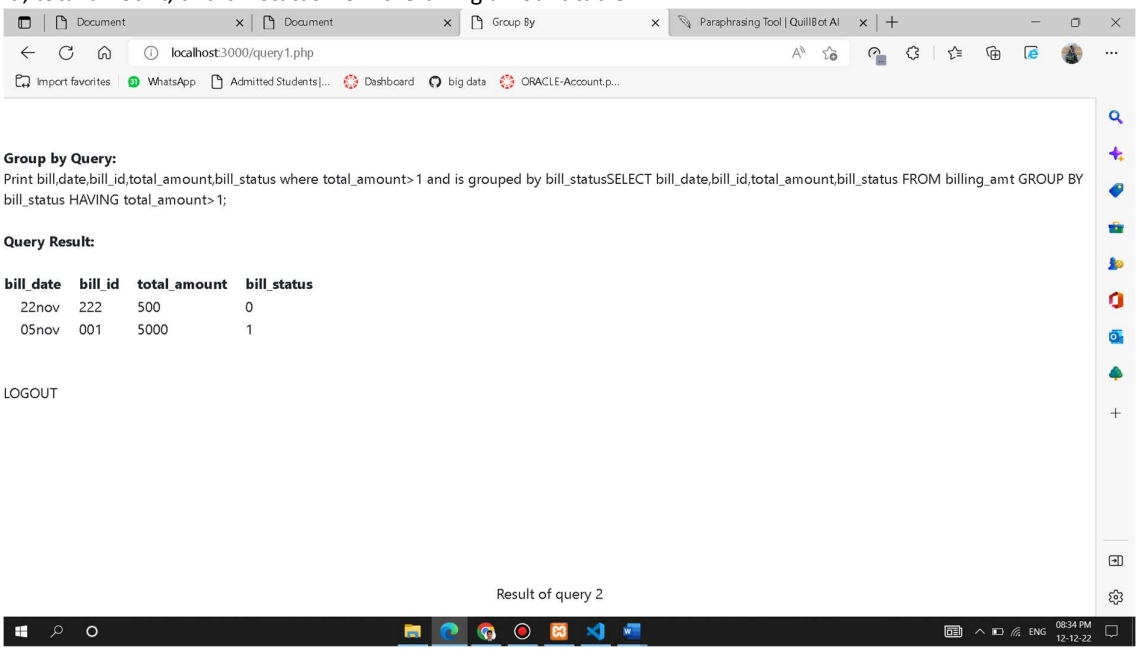

FIG10: RESULT OF QUERY1

QUERY2

The SQL IN condition (sometimes called the IN operator) allows you to easily test if an expression matches any value in a list of values. It is used to help reduce the need for multiple OR conditions in a SELECT, INSERT, UPDATE, or DELETE statement.

The IN operator is used to specify the list of values or sub query in the WHERE clause. A sub-query or list of values must be specified in the parenthesis

The use of the in clause is demonstrated in fig below.

As you can see, the print command informs you of the query that is being run.

The query retrieves the following data from the billing amount table: bill date, bill id, total amount, and bill status. The condition is to choose the bill ids that are more than the average of the total amount.

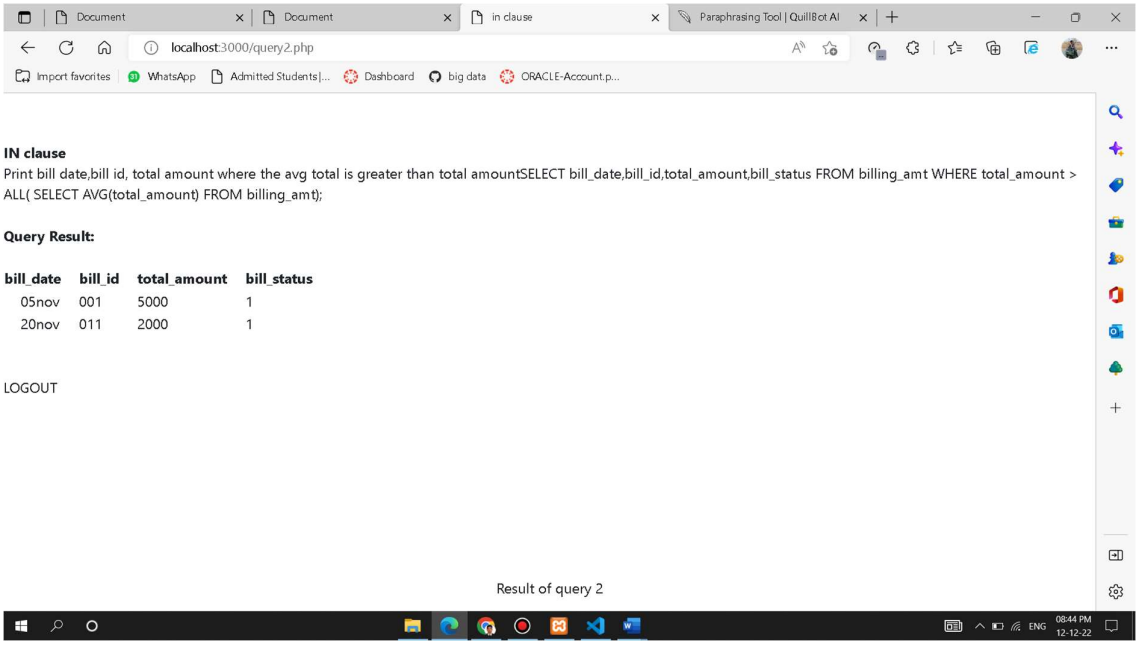

FIG11: RESULT OF QUERY2

CONCLUSION

A automobile rental system enables clients to rent cars for a predetermined amount of time. A customer interface, a fleet management system, and a payment system are typically the system's three key parts. Customers may look for, reserve, and manage their rental automobiles via the user interface. The fleet management system gives the rental company control over the upkeep and availability of their fleet. Finally, a secure payment system enables users to pay for rentals.

Businesses employ automobile rental systems to let clients rent cars for brief periods of time, typically from a few hours to a few weeks. The system typically consists of the fleet of vehicles owned by the rental company, a counter where clients can make reservations and pick up their vehicles, an online booking system, and an accounting system to keep track of rental payments and other associated costs. Customers can make bookings in person or online, and before they can pick up the automobile, they must have a valid driver's license and credit card details. The rental provider will process all necessary fees and charges after the vehicle is returned.