CAR BOOKING SYSTEM

-Sagar Lotlikar (2114040)

-Sahil Kholkar (2114041)

-Sarthak Mhalsekar (2114045)

CHAPTER 1

DATABASE DESIGN METHODOLOGY

1.1 CANDIDATE SYSTEM DESCRIPTION

Project that involves the interaction between the employee and customer with respect to renting a car. The system will allow the customer to rent a car.

So our entites include the Employee(acting as admin), Customer, Car, Booking, Payment.

Employee: The Employee entity represents individuals associated with the car Booking service. Each employee is uniquely identified by an employee ID (emp_id) and has a corresponding password (emp_password). Employees play a crucial role in managing various aspects of the car Booking system, such as managing bookings, and ensuring the overall efficiency of the service.

Customer: The Customer entity encapsulates information about individuals who utilize the car Booking service. Customers are identified by a unique customer ID (cust_id) and provide details such as their first name (fname), last name (lname), email address (email), username (username) for login, password (password), and address (address). This entity enables the system to maintain personalized customer profiles and manage their interactions with the car Booking service.

Booking: The Booking entity tracks the details of each car reservation made by customers. It includes a unique booking ID (book_id) and references to the rented car (car_id) and the customer making the booking (cust_id). Additional information comprises the booking date (book_date), pickup date (pickup_date), return date (return_date), and the location of the booking (book_place). This entity facilitates the efficient organization and tracking of reservations within the car Booking system.

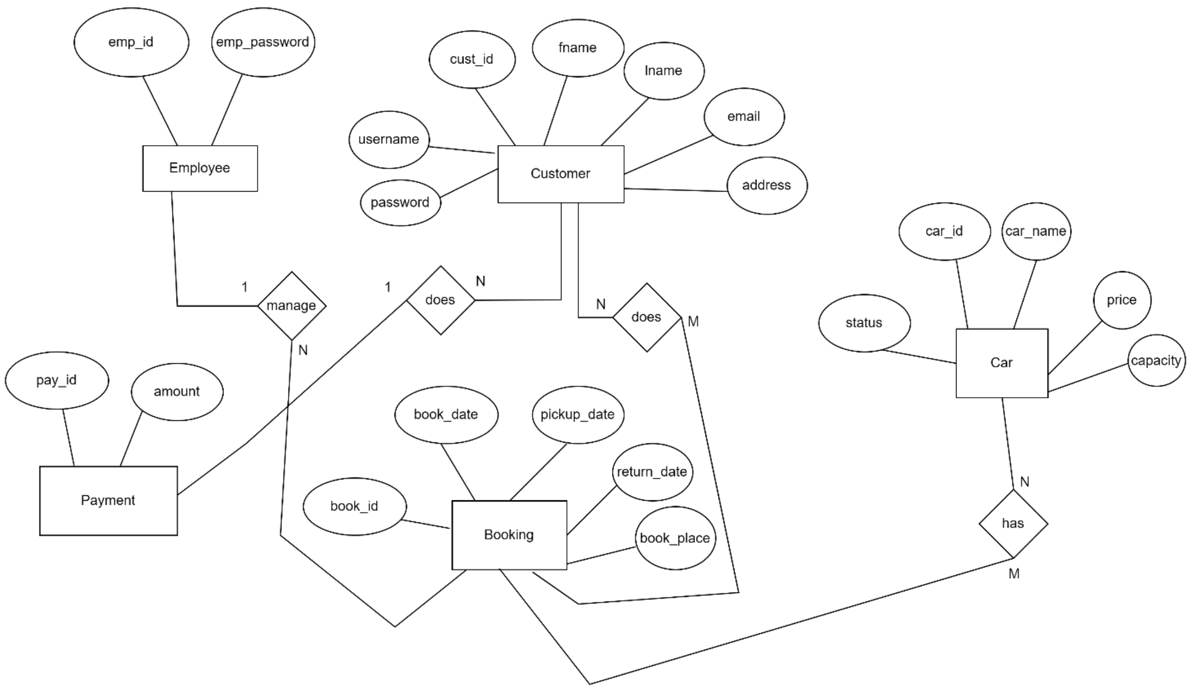
Car: The Car entity represents the fleet of vehicles available for Booking. Each car is uniquely identified by a car ID (car_id) and is associated with attributes such as the car's name or model (car_name), Booking price (price), maximum capacity (capacity), and availability status (status). This entity enables the system to manage and display information about the cars offered for rent, including their current availability.

Payment: The Payment entity records financial transactions associated with car bookings. It includes a unique payment ID (pay_id) and references the corresponding booking (book_id). The amount attribute signifies the payment made for the booking. This entity ensures accurate financial tracking, allowing the system to keep a record of payments and provide customers with transparent billing information.

1.2 ENTITY RELATIONSHIP DIAGRAM

An entity relationship diagram shows the connection between entity sets that are kept in the database. To put it another way, ER diagrams assist in describing the logical organisation of the database. Entities, attributes, and relationships are the three fundamental ideas that serve as the foundation of the ER diagram.

Rectangles are represented as entities in the ER diagram, ovals are used to indicate attributes and diamond shapes are used to represent relationships. An ER diagram and a flowchart initially have a similar appearance. designed to show how a system is divided into smaller portions and to highlight the flow of data between those parts. It is useful for analysing existing as well as proposed systems.


emp_id
emp_password
Employee
cust_id
fname
lname
email
username
Customer
password
address
car_id
car_name
price
status
Car
capacity
1
manage
1
does
N
N
does
M
N
pay_id
amount
Payment
book_date
pickup_date
return_date
book_id
Booking
book_place
N
has
M

Constraints between entities Employee and Booking

Cardinality: There is a 1:N cardinality with respect to Employee and Booking entities as 1 employee will manage many(N) booking in the system provided by us.

Participation of the entities: The employee and booking entity has total participation in the ER diagram.

Constraints between entities Customer and Booking

Cardinality: There is a N:M cardinality with respect to Customer and Booking entities as N customer will book many(N) booking in the system provided by us.

Participation of the entities: The customer and booking entity has total participation in the ER diagram.

Constraints between entities Booking and Car

Cardinality: There is a N:M cardinality with respect to Booking and Car entities as N booking will have many(N) car in the system provided by us.

Participation of the entities: The booking and car entity has total participation in the ER diagram.

Constraints between entities Payment and Customer

Cardinality: There is a 1:N cardinality with respect to Customer and Payment entities as N customer will make 1 booking in the system provided by us.

Participation of the entities: The payment and customer entity has total participation in the ER diagram.

1.3 CONVERSION OF THE ER DIAGRAM TO RELATIONAL MAPPING VIA ALGORITHMS

Step 1: Mapping of Regular Entities

- For each regular (strong) entity type E in the ER schema, create a relation R that includes all the simple attributes of E.
- Choose one of the key attributes of E as the primary key for R.
- If the chosen key of E is composite, the set of simple attributes that form it will together form the primary key of R.

Step 2: Mapping of Weak Entity Types

- For each weak entity type W in the ER schema with owner entity type E, create a relation R & include all simple attributes (or simple components of composite attributes) of W as attributes of R.
- Also, include as foreign key attributes of R the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s).
- The primary key of R is the combination of the primary key(s) of the owner(s) and the partial key of the weak entity type W,

Step 3: Mapping of Binary 1:1 Relation Types

- For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R.
- There are three possible approaches:
- Foreign Key approach: Choose one of the relations-say S-and include a foreign key in S the primary key of T. It is better to choose an entity type with total participation in R in the role of S.
- Merged relation option: An alternate mapping of a 1:1 relationship type is possible by merging the two entity types and the relationship into a single relation. This may be appropriate when both participations are total.
- Cross-reference or relationship relation option: The third alternative is to set up a third relation R for the purpose of cross-referencing the primary keys of the two relations S and T representing the entity types.

Step 4: Mapping of Binary 1:N Relationship Types.

- For each regular binary 1:N relationship type R, identify the relation S that represents the participating entity type at the N-side of the relationship type.
- Include as a foreign key in S the primary key of the relation T that represents the other entity type participating in R.
- Include any simple attributes of the 1:N relation type as attributes of S.

Step 5: Mapping of Binary M:N Relationship Types.

- For each regular binary M:N relationship type R, create a new relation S to represent R.
- Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types; their combination will form the primary key of S.
- Also include any simple attributes of the M:N relationship type (or simple components of composite attributes) as attributes of S.

Step 6: Mapping of Multivalued attributes.

- For each multivalued attribute A, create a new relation R.
- This relation R will include an attribute corresponding to A, plus the primary key attribute K-as a foreign key in R-of the relation that represents the entity type of relationship type that has A as an attribute.
- The primary key of R is the combination of A and K. If the multivalued attribute is composite, we include its simple components.

employee

emp_id	emp_password

customer

cust_id	fname	lname	email	password	username	address

car

car_id	car_name	price	capacity	status

booking

book_id	car_id	cust_id	book_date	pickup_date	return_date	book_place

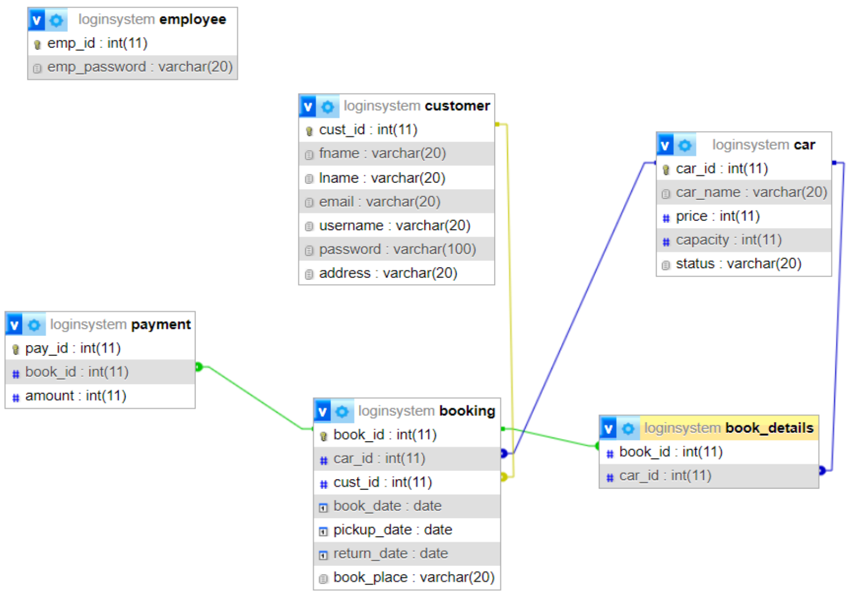
book_details

book_id	car_id

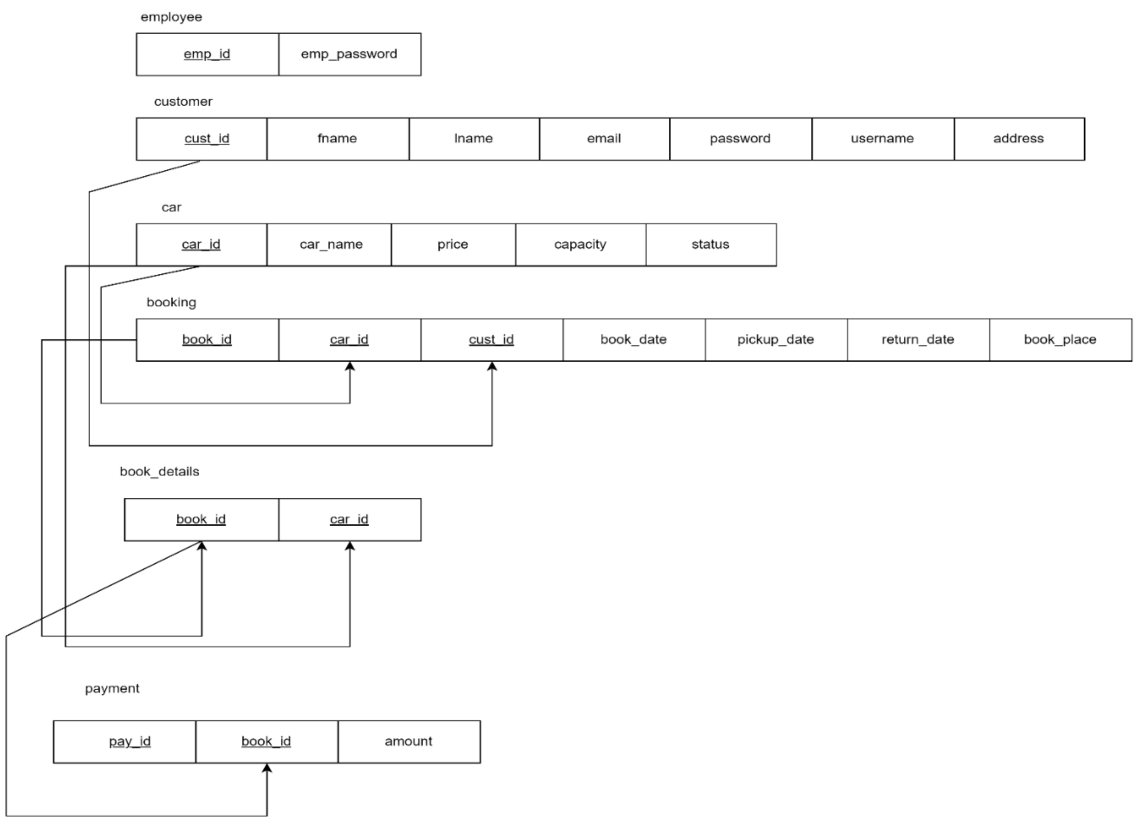
payment

pay_id	book_id	amount

1.3 RELATIONAL SCHEMA FOR THE MANAGEMENT SYSTEM ALONG WITH KEY CONSTRAINTS



RESPECTIVE SCHEMA



1.3 NORMALISATION CHART OF THE DATABASE

1st Normal Form

The employee table is in 1NF because it has no repeating groups and each column has a single value for each row. Customer, car, booking and payment is also in 1NF reason is same

employee

emp_id	emp_password

customer

cust_id	fname	lname	email	password	username	address

car

car_id	car_name	price	capacity	status

booking

book_id	car_id	cust_id	book_date	pickup_date	return_date	book_place

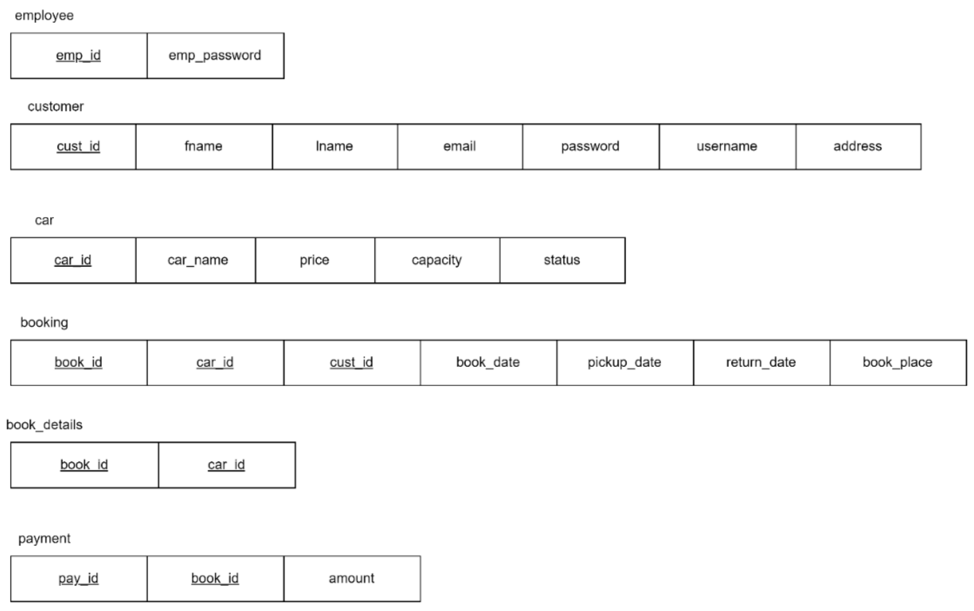
payment

pay_id	book_id	amount

2nd Normal Form

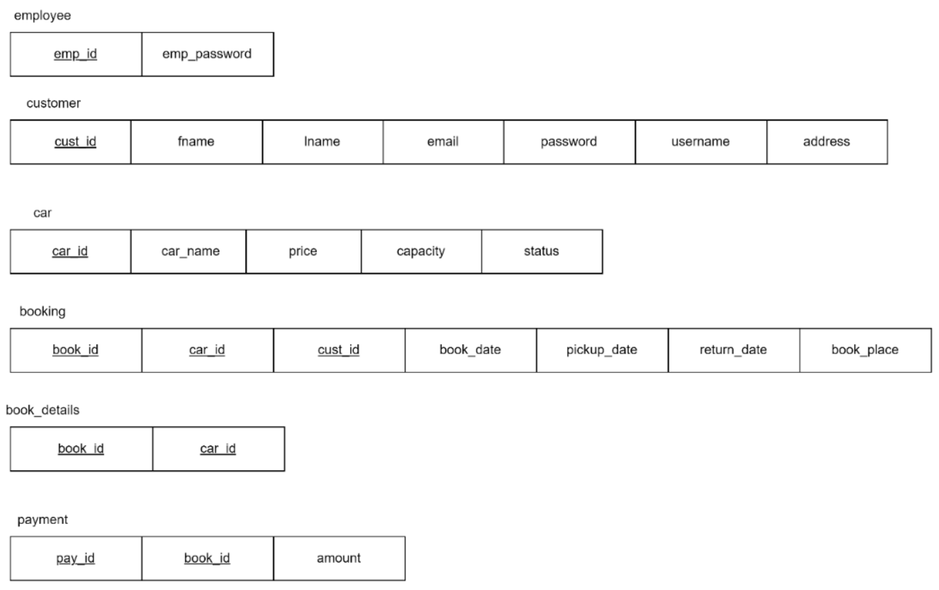
The Booking table is in 1NF because it has no repeating groups, but it is not in 2NF because the TotalAmount column depends on the CarlD and the ReturnDate columns, which are not part of the primary key. To make the Booking table in 2NF, we need to remove the TotalAmount column and create a new table called BookingDetail with the following columns:

BookingID (foreign key), CarlD (foreign key), ReturnDate, and TotalAmount. The BookingDetail table is in 2NF because every non-key column depends on the whole primary key, which is the combination of BookingID and CarlD. The Payment table is in 2NF because every non-key column depends on the primary key which is PaymentID



3rd Normal Form

The Car table is in 3NF because every non-key column depends only on the primary key, which is CarlD. The Customer table is also in 3NF for the same reason. The Booking table is in 3NF because every non- key column depends only on the primary key, which is BookinglD. The BookingDetail table is in 3NF because every non-key column depends only on the primary key, which is the combination of BookingID and CarlD. The Payment table is in 3NF because every non-key column depends only on the primary key, which is redundant or PaymentID. The car Booking system database model is in 3NF. This means that the database has no transitive dependencies, and it ensures data consistency and integrity.



1.5.1 **DATABASE**

```
CREATE TABLE `available_cars_view` (
`car_id` int(11)
,`car_name` varchar(20)
,`price` int(11)
,`capacity` int(11)
);

CREATE TABLE `booking` (
  `book_id` int(11) NOT NULL,
  `car_id` int(11) DEFAULT NULL,
  `cust_id` int(11) DEFAULT NULL,
  `book_date` date DEFAULT NULL,
  `pickup_date` date DEFAULT NULL,
  `return_date` date DEFAULT NULL,
  `book_place` varchar(20) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

INSERT INTO `booking` (`book_id`, `car_id`, `cust_id`, `book_date`, `pickup_date`,
`return_date`, `book_place`) VALUES
(23, 5, 9, '2023-11-22', '2023-11-23', '2023-11-30', 'Mapusa');

CREATE TABLE `booking_details_view` (
`book_id` int(11)
,`book_date` date
,`pickup_date` date
,`return_date` date
,`book_place` varchar(20)
,`car_name` varchar(20)
,`price` int(11)
,`customer_fname` varchar(20)
,`customer_lname` varchar(20)
,`customer_email` varchar(20)
);

CREATE TABLE `book_details` (
  `book_id` int(11) DEFAULT NULL,
  `car_id` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

CREATE TABLE `car` (
  `car_id` int(11) NOT NULL,
  `car_name` varchar(20) DEFAULT NULL,
  `price` int(11) DEFAULT NULL,
  `capacity` int(11) DEFAULT NULL,
  `status` varchar(20) NOT NULL DEFAULT 'Available'
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

```
INSERT INTO `car` (`car_id`, `car_name`, `price`, `capacity`, `status`) VALUES

(1, 'mahindra thar', 7000, 4, 'Available'),

(2, 'Tata Nexon', 6000, 4, 'Available'),

(3, 'Mahindra XUV700', 4000, 4, 'Available'),

(4, 'Mahindra Scorpio N', 7000, 4, 'Available'),

(5, 'Tata Harrier', 7000, 4, 'Not Available'),

(6, 'Skoda Kushaq', 7000, 4, 'Available');



CREATE TABLE `customer` (

  `cust_id` int(11) NOT NULL,

  `fname` varchar(20) DEFAULT NULL,

  `lname` varchar(20) DEFAULT NULL,

  `email` varchar(20) DEFAULT NULL,

  `username` varchar(20) DEFAULT NULL,

  `password` varchar(100) DEFAULT NULL,

  `address` varchar(20) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;


INSERT INTO `customer` (`cust_id`, `fname`, `lname`, `email`, `username`, `password`,
`address`) VALUES

(9, 'Sagar', 'Lotlikar', 's@gmail.com', 'sagar', '202cb962ac59075b964b07152d234b70',
'Cuncolim');


CREATE TABLE `customer_return_days_view` (
`cust_id` int(11)
,`fname` varchar(20)
,`lname` varchar(20)
,`book_id` int(11)
,`return_date` date
,`days_until_return` int(7)
);

CREATE TABLE `employee` (
  `emp_id` int(11) NOT NULL,
  `emp_password` varchar(20) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

INSERT INTO `employee` (`emp_id`, `emp_password`) VALUES
(1, '1');

CREATE TABLE `payment` (
  `pay_id` int(11) NOT NULL,
  `book_id` int(11) DEFAULT NULL,
  `amount` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

INSERT INTO `payment` (`pay_id`, `book_id`, `amount`) VALUES
(8, 23, 49000);
```

```
CREATE TABLE `payment_summary_view` (

`pay_id` int(11)

,`book_id` int(11)

,`payment_amount` int(11)

,`customer_fname` varchar(20)

,`customer_lname` varchar(20)

,`customer_email` varchar(20)

);
```

VIEWS USED

```
CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY
DEFINER VIEW `available_cars_view`  AS SELECT `car`.`car_id` AS `car_id`,
`car`.`car_name` AS `car_name`, `car`.`price` AS `price`, `car`.`capacity` AS `capacity`
FROM `car` WHERE `car`.`status` = 'Available' ;


CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY
DEFINER VIEW `booking_details_view`  AS SELECT `b`.`book_id` AS `book_id`,
`b`.`book_date` AS `book_date`, `b`.`pickup_date` AS `pickup_date`, `b`.`return_date` AS
`return_date`, `b`.`book_place` AS `book_place`, `c`.`car_name` AS `car_name`, `c`.`price`
AS `price`, `cu`.`fname` AS `customer_fname`, `cu`.`lname` AS `customer_lname`,
`cu`.`email` AS `customer_email` FROM ((`booking` `b` join `car` `c` on(`b`.`car_id` =
`c`.`car_id`)) join `customer` `cu` on(`b`.`cust_id` = `cu`.`cust_id`)) ;


CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY
DEFINER VIEW `customer_return_days_view`  AS SELECT `cu`.`cust_id` AS `cust_id`,
`cu`.`fname` AS `fname`, `cu`.`lname` AS `lname`, `b`.`book_id` AS `book_id`,
`b`.`return_date` AS `return_date`, to_days(`b`.`return_date`) - to_days(curdate()) AS
```

```
`days_until_return` FROM (`customer` `cu` join `booking` `b` on(`cu`.`cust_id` =
`b`.`cust_id`)) WHERE `b`.`return_date` >= curdate() ;


CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY
DEFINER VIEW `payment_summary_view`  AS SELECT `p`.`pay_id` AS `pay_id`,
`p`.`book_id` AS `book_id`, `p`.`amount` AS `payment_amount`, `cu`.`fname` AS
`customer_fname`, `cu`.`lname` AS `customer_lname`, `cu`.`email` AS `customer_email`
FROM ((`payment` `p` join `booking` `b` on(`p`.`book_id` = `b`.`book_id`)) join `customer`
`cu` on(`b`.`cust_id` = `cu`.`cust_id`)) ;


ALTER TABLE `booking`
  ADD PRIMARY KEY (`book_id`),
  ADD KEY `car_id` (`car_id`),
  ADD KEY `cust_id` (`cust_id`);


ALTER TABLE `book_details`
  ADD KEY `book_id` (`book_id`),
  ADD KEY `car_id` (`car_id`);
ALTER TABLE `car`
  ADD PRIMARY KEY (`car_id`);


ALTER TABLE `customer`
  ADD PRIMARY KEY (`cust_id`);


ALTER TABLE `employee`
  ADD PRIMARY KEY (`emp_id`);
```

```
ALTER TABLE `payment`
  ADD PRIMARY KEY (`pay_id`),
  ADD KEY `book_id` (`book_id`);

ALTER TABLE `booking`
  MODIFY `book_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=24;

ALTER TABLE `car`
  MODIFY `car_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=7;

ALTER TABLE `customer`
  MODIFY `cust_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=10;

ALTER TABLE `employee`
  MODIFY `emp_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;

ALTER TABLE `payment`
  MODIFY `pay_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=9;

ALTER TABLE `booking`
  ADD CONSTRAINT `booking_ibfk_1` FOREIGN KEY (`car_id`) REFERENCES `car`
(`car_id`),
  ADD CONSTRAINT `booking_ibfk_2` FOREIGN KEY (`cust_id`) REFERENCES
`customer` (`cust_id`);
```

```
ALTER TABLE `book_details`
  ADD CONSTRAINT `book_details_ibfk_1` FOREIGN KEY (`book_id`) REFERENCES
`booking` (`book_id`),
  ADD CONSTRAINT `book_details_ibfk_2` FOREIGN KEY (`car_id`) REFERENCES
`car` (`car_id`);
ALTER TABLE `payment`
  ADD CONSTRAINT `payment_ibfk_1` FOREIGN KEY (`book_id`) REFERENCES
`booking` (`book_id`);
```

TRIGGERS USED

```
CREATE TABLE booking_trigger (
    trigger_id INT AUTO_INCREMENT PRIMARY KEY,
    event_type VARCHAR(10),
    book_id INT,
    car_id INT,
    cust_id INT,
    event_timestamp TIMESTAMP
);

CREATE TABLE payment_trigger (
    trigger_id INT AUTO_INCREMENT PRIMARY KEY,
    event_type VARCHAR(10),
    pay_id INT,
    book_id INT,
    amount INT,
    event_timestamp TIMESTAMP
);

DELIMITER //
CREATE TRIGGER after_booking_insert
AFTER INSERT ON booking FOR EACH ROW
BEGIN
    INSERT INTO booking_trigger (event_type, book_id, car_id, cust_id, event_timestamp)
    VALUES ('INSERT', NEW.book_id, NEW.car_id, NEW.cust_id, NOW());
END;
//
DELIMITER ;


DELIMITER //
CREATE TRIGGER after_payment_insert
AFTER INSERT ON payment FOR EACH ROW
BEGIN
    INSERT INTO payment_trigger (event_type, pay_id, book_id, amount, event_timestamp)
    VALUES ('INSERT', NEW.pay_id, NEW.book_id, NEW.amount, NOW());
END;
//
DELIMITER ;
```

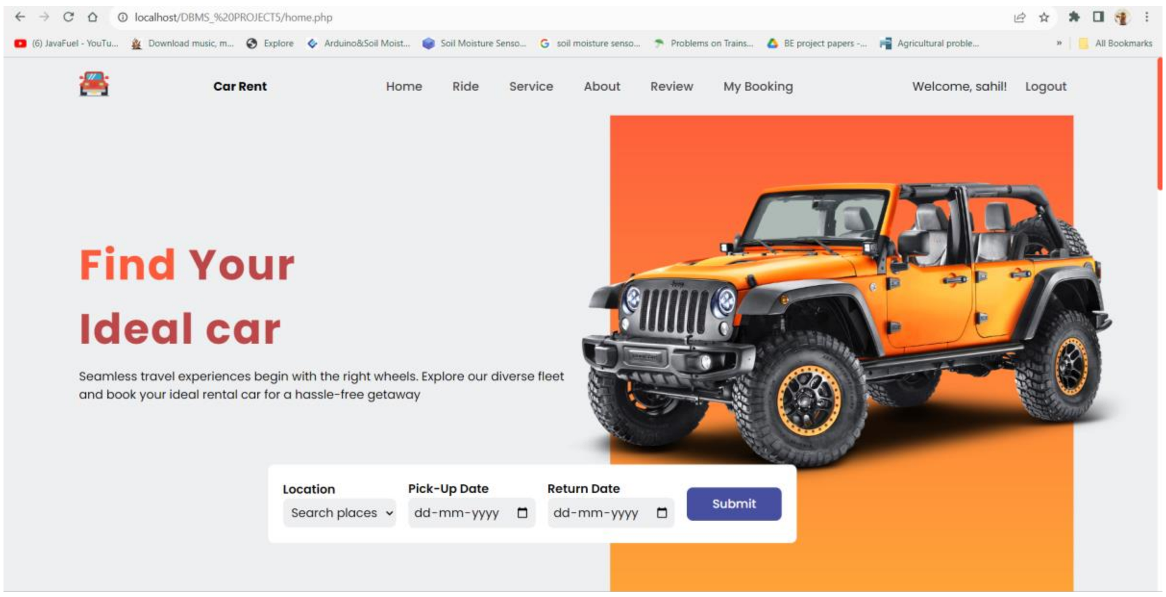
<u>JOINS USED</u>

```
$sqlCarEarnings = "SELECT

        car.car_id,

        car.car_name,

        COUNT(booking.book_id) AS total_bookings,

        SUM(payment.amount) AS total_amount

    FROM

        car

    LEFT JOIN booking ON car.car_id = booking.car_id

    LEFT JOIN payment ON booking.book_id = payment.book_id

    GROUP BY

        car.car_id, car.car_name

    ORDER BY

        total_amount desc;

";


$sql = "SELECT b.*, c.fname, c.Lname, p.amount

              FROM booking b

              JOIN customer c ON b.cust_id = c.cust_id

              LEFT JOIN payment p ON b.book_id = p.book_id";
```

CHAPTER 2

IMPLEMENTATION

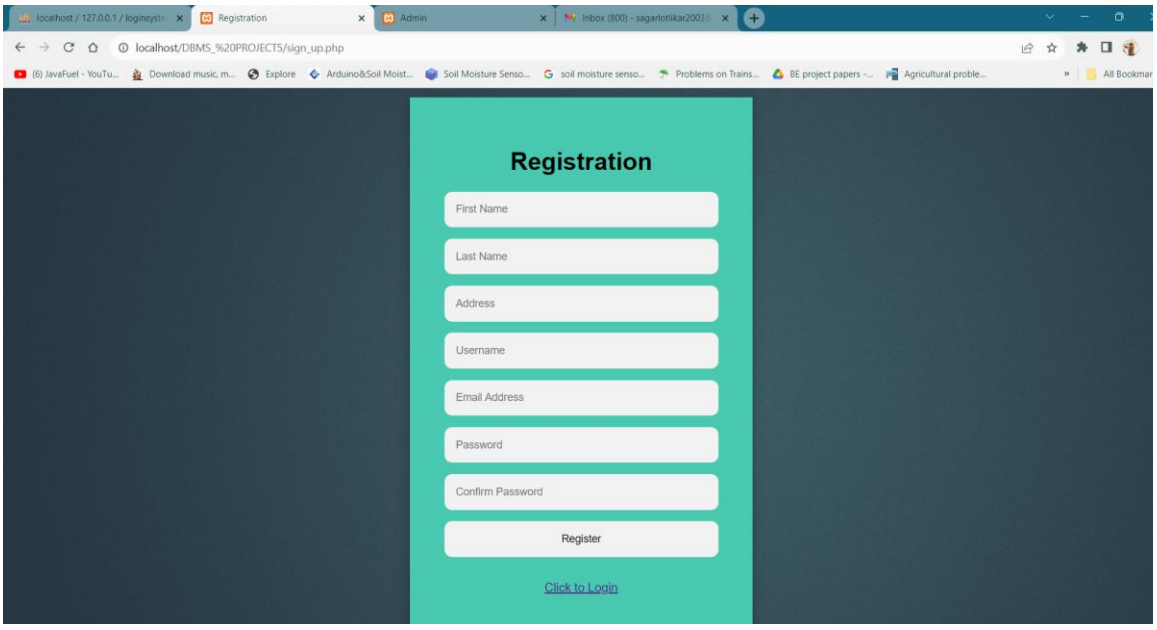
5.1 HOME PAGE



Seamless travel experiences begin with the right wheels. Explore our diverse fleet and book your ideal rental car for a hassle-free getaway

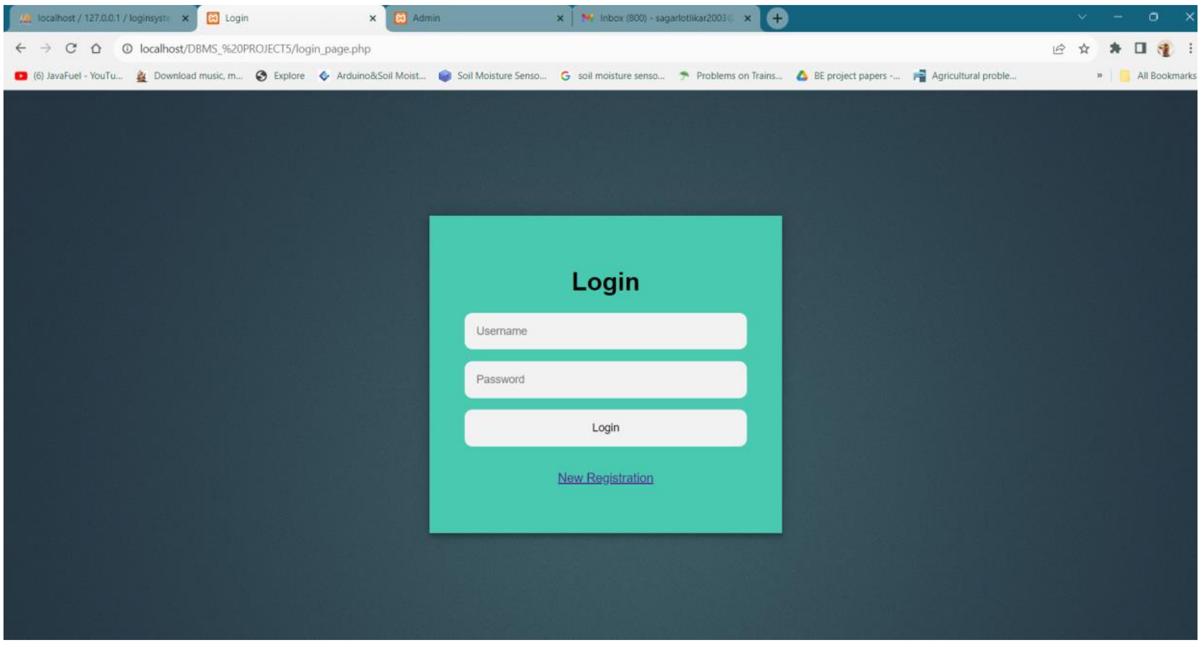
The home page of our project serves as the central hub for our car Booking management system, offering a user-friendly interface that caters to both customers and administrators. The design is clean and intuitive, ensuring a seamless user experience.

5.2 SIGN IN PAGE



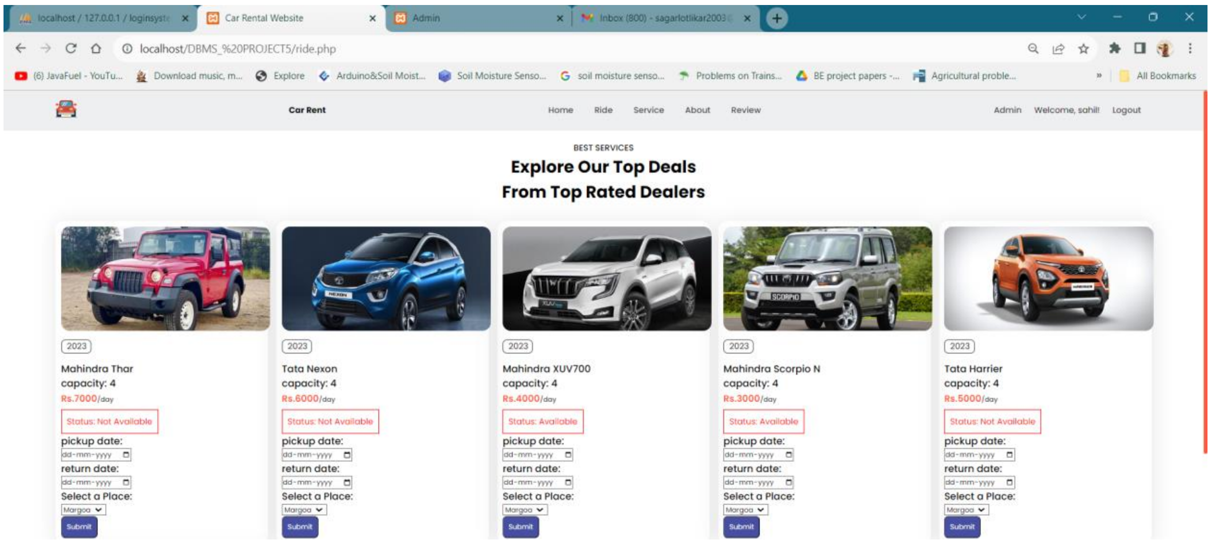
In the sign in page if you are new user you can register to login

5.3 LOGIN PAGE



After registration you can login here or if you are an existing user

5.4 RIDE PAGE



2023
Mahindra Thar
capacity: 4
Rs.7000/day
Status: Not Available
pickup date: dd-mm-yyyy
return date: dd-mm-yyyy
Select a Place: Margoa
Submit

2023
Tata Nexon
capacity: 4
Rs.6000/day
Status: Not Available
pickup date: dd-mm-yyyy
return date: dd-mm-yyyy
Select a Place: Margoa
Submit

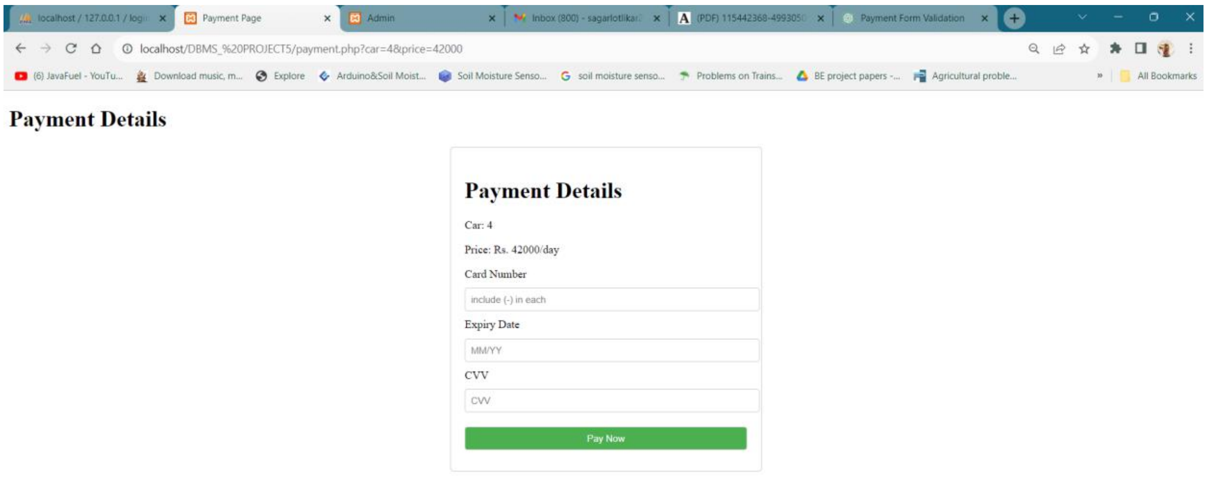
2023
Mahindra XUV700
capacity: 4
Rs.4000/day
Status: Available
pickup date: dd-mm-yyyy
return date: dd-mm-yyyy
Select a Place: Margoa
Submit

2023
Mahindra Scorpio N
capacity: 4
Rs.3000/day
Status: Available
pickup date: dd-mm-yyyy
return date: dd-mm-yyyy
Select a Place: Margoa
Submit

2023
Tata Harrier
capacity: 4
Rs.5000/day
Status: Not Available
pickup date: dd-mm-yyyy
return date: dd-mm-yyyy
Select a Place: Margoa
Submit

Here the customers will be able to book the rented cars as per customers preferred date.

5.5 PAYMENT PAGE


localhost/DBMS_%20PROJECT5/payment.php?car=4&price=42000

Payment Details

Payment Details

Car: 4

Price: Rs. 42000/day

Card Number

include (-) in each

Expiry Date

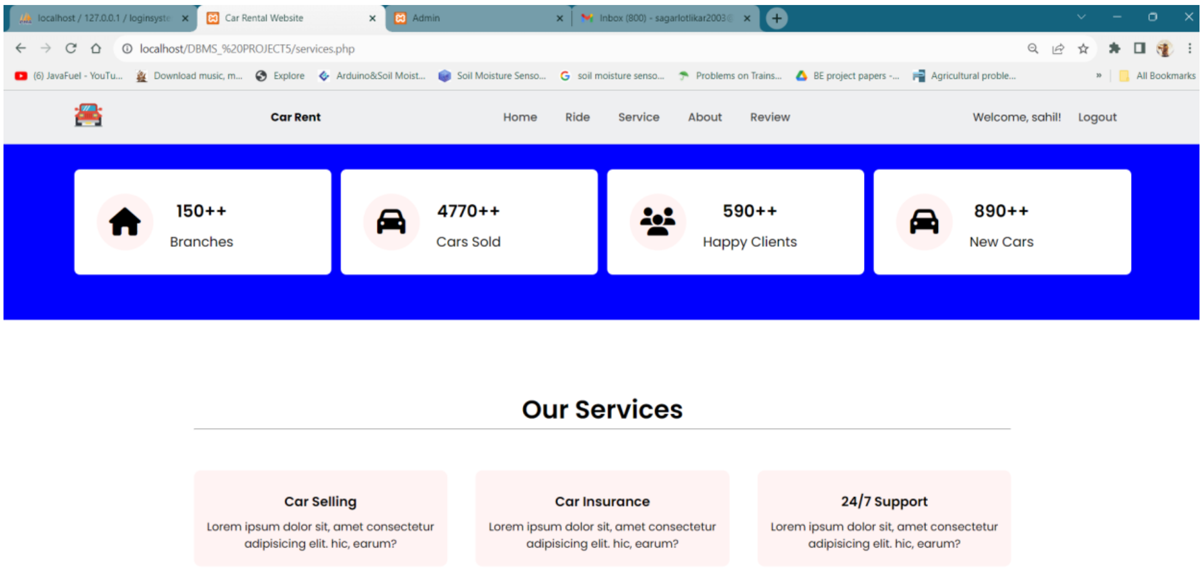
MM/YY

CVV

CVV

After booking a car, customer will be proceeding to the payment page

5.6 SERVICE PAGE


localhost/DBMS_%20PROJECT5/services.php

150++ Branches

4770++ Cars Sold

590++ Happy Clients

890++ New Cars

Our Services

Car Selling

Lorem ipsum dolor sit, amet consectetur adipisicing elit. hic, earum?

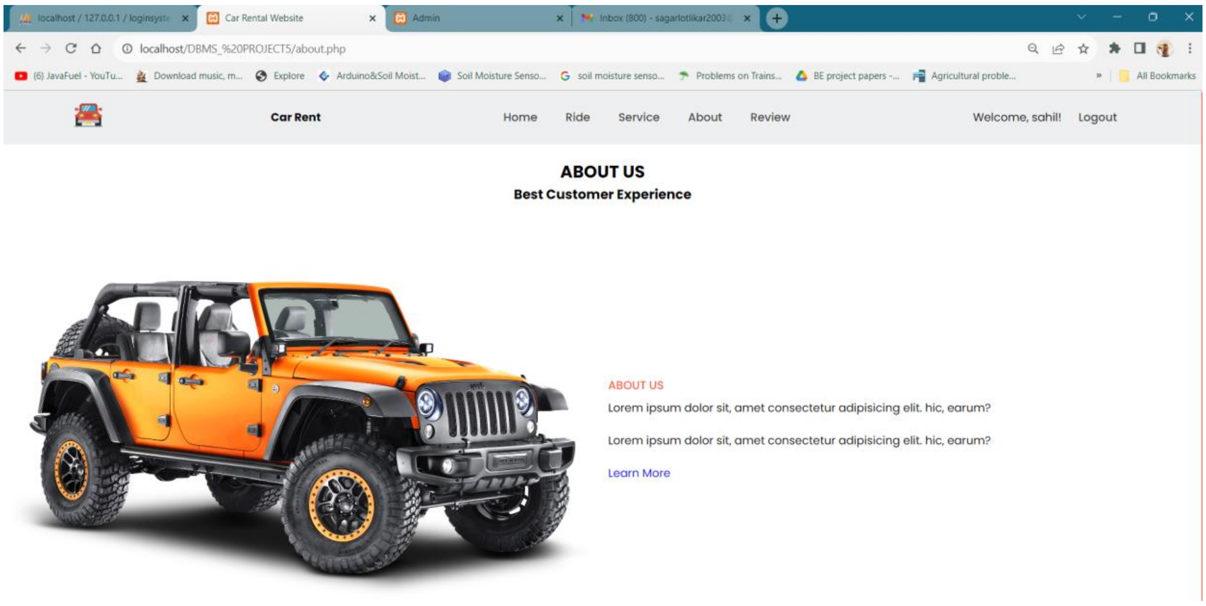
Car Insurance

Lorem ipsum dolor sit, amet consectetur adipisicing elit. hic, earum?

24/7 Support

Lorem ipsum dolor sit, amet consectetur adipisicing elit. hic, earum?

5.6 ABOUT US PAGE



ABOUT US

Best Customer Experience

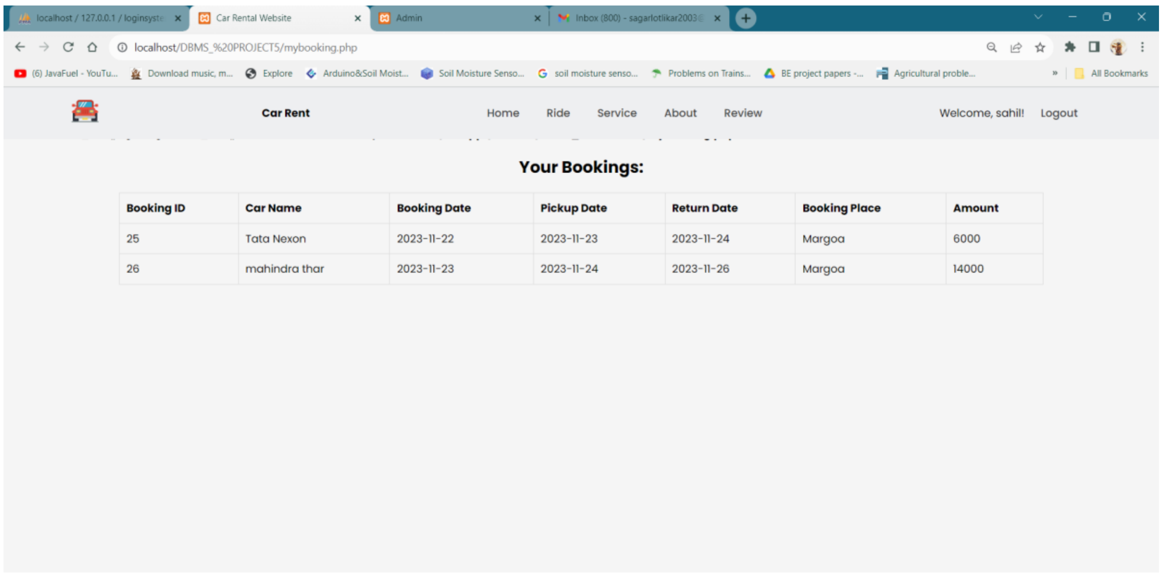
ABOUT US

Lorem ipsum dolor sit, amet consectetur adipisicing elit. hic, earum?

Lorem ipsum dolor sit, amet consectetur adipisicing elit. hic, earum?

Learn More

5.6 MY BOOKING

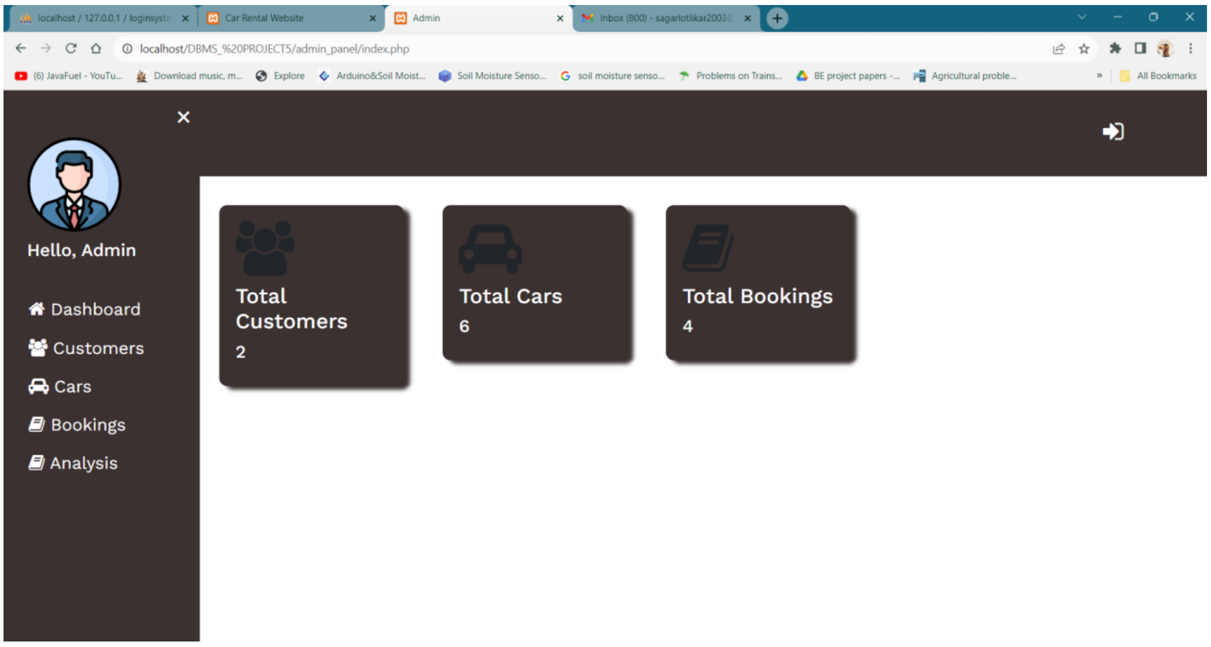


Your Bookings:

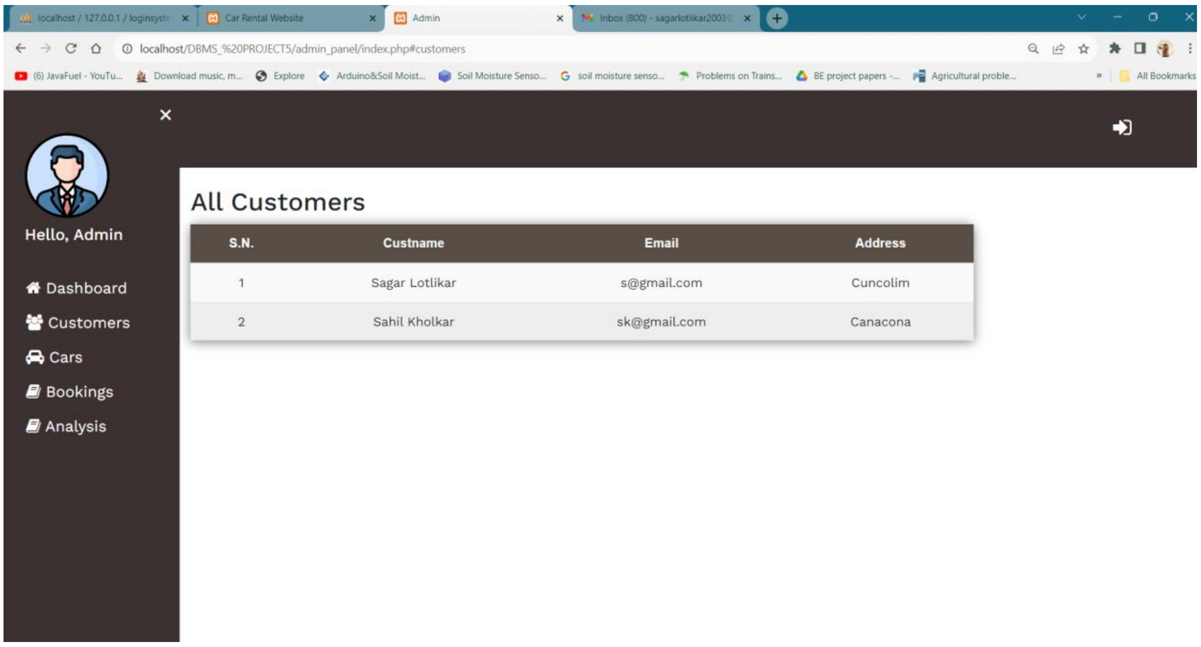
Booking ID	Car Name	Booking Date	Pickup Date	Return Date	Booking Place	Amount
25	Tata Nexon	2023-11-22	2023-11-23	2023-11-24	Margoa	6000
26	mahindra thar	2023-11-23	2023-11-24	2023-11-26	Margoa	14000

Here the customer will able to see all the bookings which he had made

5.2 ADMIN'S PAGE



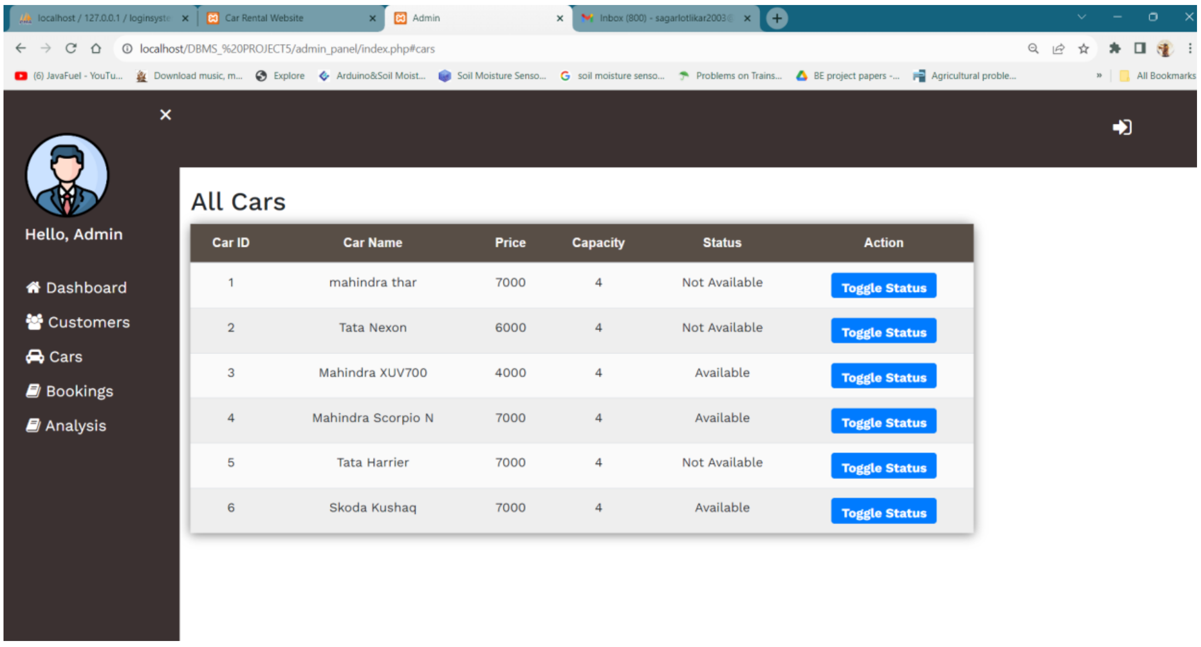
The home page of admin project serves as the all details of customers for our car Booking management system, offering a user-friendly interface that admin can handle all customers details The design is clean and intuitive, ensuring a seamless user experience.



All Customers

S.N.	Custname	Email	Address
1	Sagar Lotlikar	s@gmail.com	Cuncolim
2	Sahil Kholkar	sk@gmail.com	Canacona

Here admin can view all the registered customers

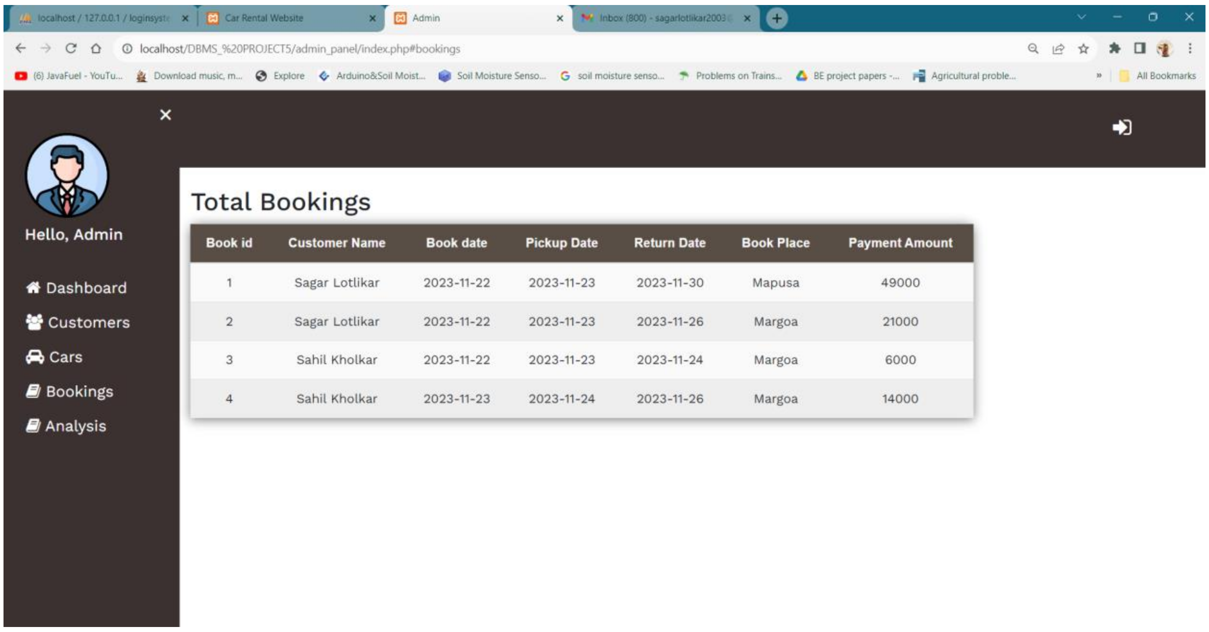

Hello, Admin

Dashboard
Customers
Cars
Bookings
Analysis

All Cars

Car ID	Car Name	Price	Capacity	Status	Action
1	mahindra thar	7000	4	Not Available	Toggle Status
2	Tata Nexon	6000	4	Not Available	Toggle Status
3	Mahindra XUV700	4000	4	Available	Toggle Status
4	Mahindra Scorpio N	7000	4	Available	Toggle Status
5	Tata Harrier	7000	4	Not Available	Toggle Status
6	Skoda Kushaq	7000	4	Available	Toggle Status

Here the admin can make changes to status of the cars

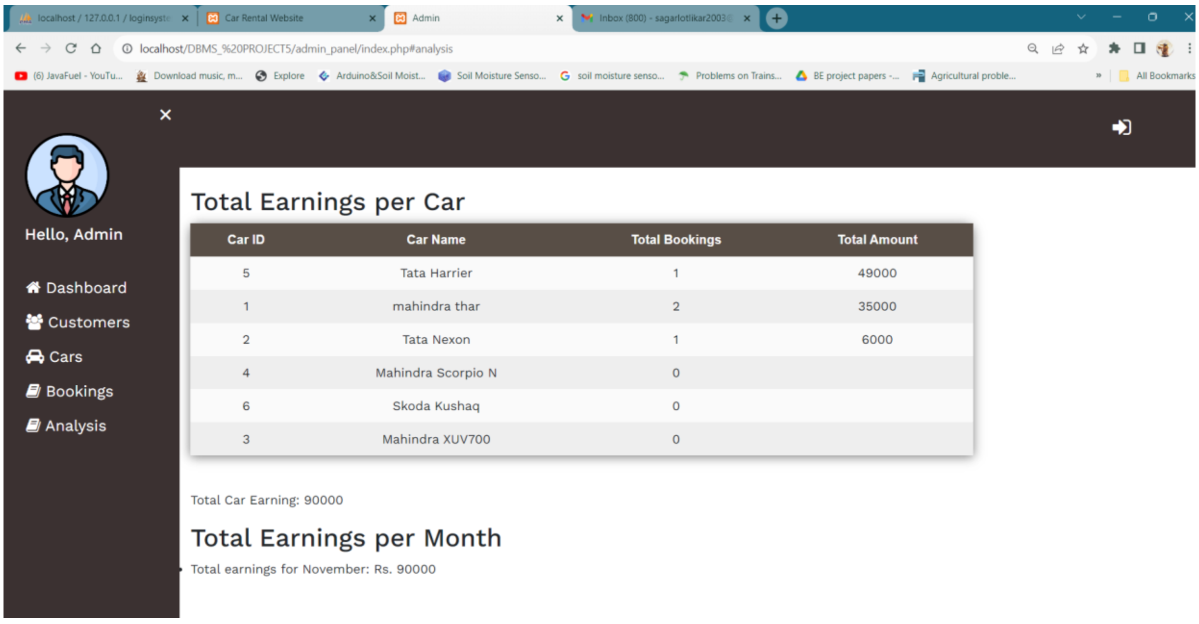

Hello, Admin

Dashboard
Customers
Cars
Bookings
Analysis

Total Bookings

Book id	Customer Name	Book date	Pickup Date	Return Date	Book Place	Payment Amount
1	Sagar Lotlikar	2023-11-22	2023-11-23	2023-11-30	Mapusa	49000
2	Sagar Lotlikar	2023-11-22	2023-11-23	2023-11-26	Margoa	21000
3	Sahil Kholkar	2023-11-22	2023-11-23	2023-11-24	Margoa	6000
4	Sahil Kholkar	2023-11-23	2023-11-24	2023-11-26	Margoa	14000

Here the admin can view all customer booking details



Hello, Admin

Dashboard

Customers

Cars

Bookings

Analysis

Total Earnings per Car

Car ID	Car Name	Total Bookings	Total Amount
5	Tata Harrier	1	49000
1	mahindra thar	2	35000
2	Tata Nexon	1	6000
4	Mahindra Scorpio N	0	
6	Skoda Kushaq	0	
3	Mahindra XUV700	0	

Total Car Earning: 90000

Total Earnings per Month

- Total earnings for November: Rs. 90000

Here admin can analyze which car was booked most of the times and the total amount also the profit per month