

Project Title

A PROJECT REPORT

Submitted by

M.PAVAN [Reg No: RA2211003011164]

N.V.DEEPAK [Reg No: RA2211003011165]

Under the Guidance of

Dr.P.Madhavan

Designation, Department of Computing Technologies

in partial fulfillment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING



DEPARTMENT OF COMPUTING TECHNOLOGIES

COLLEGE OF ENGINEERING AND

TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND

TECHNOLOGY KATTANKULATHUR– 603 203

MAY 2024



SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR–603 203

BONAFIDE CERTIFICATE

[RA2211003011164] , [RA2211003011165] Certified to be the bonafide work done by **M.PAVAN , N.V.DEEPAK** of II year/IV sem B.Tech Degree Course in the Project Course – **21CSC205P Database Management Systems** in **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**, Kattankulathur for the academic year 2023-2024.

Date:

Faculty in Charge
Dr.P.Madhavan
Designation
Department name
SRMSIT -KTR

HEAD OF THE DEPARTMENT
Name of the HOD
Designation
Department name
SRMIST - KTR

ABSTRACT

Vehicle parking management systems revolutionize urban mobility by leveraging technology to streamline parking allocation, payment, and monitoring processes. These systems utilize sensors, cameras, and software algorithms to efficiently guide drivers to available parking spaces, reducing congestion and enhancing overall traffic flow. By implementing real-time data analysis, these systems optimize parking space utilization, ensuring maximum efficiency and revenue generation for parking operators.

Moreover, advanced parking management systems offer users convenient payment options, including mobile apps and contactless payment methods, enhancing the overall parking experience. Additionally, these systems provide parking operators with valuable insights into usage patterns, allowing for better planning and resource allocation.

Furthermore, integration with smart city initiatives enables parking management systems to contribute to broader urban planning goals, such as reducing pollution and promoting sustainable transportation options. Through seamless integration with public transportation systems and ride-sharing services, these systems facilitate multimodal transportation solutions, promoting a more interconnected and sustainable urban environment.

Overall, vehicle parking management systems play a crucial role in modernizing urban infrastructure, improving traffic flow, reducing environmental impact, and enhancing the overall quality of life for city residents and visitors alike."

PROBLEM STATEMENT

In congested urban areas, effective management of vehicle parking is paramount for optimizing traffic flow and enhancing overall mobility. The existing challenges, including inefficient space utilization, lack of real-time information, and subpar user experiences, necessitate the development of an advanced Vehicle Parking Management System (VPMS). This system integrates cutting-edge technologies such as IoT sensors, data analytics, and mobile applications to revolutionize parking operations.

The proposed VPMS addresses these challenges by providing real-time updates on parking availability through mobile apps and digital signage, thus minimizing search time and reducing traffic congestion. By employing machine learning algorithms, it intelligently guides motorists to vacant parking spots, optimizing space utilization and improving user convenience. Seamless payment methods and automated access control systems enhance the overall parking experience, while robust security measures ensure the safety of vehicles and individuals within parking facilities.

Moreover, the VPMS promotes sustainability by incentivizing eco-friendly transportation options and incorporating green design principles into parking structures. With scalability and flexibility at its core, the system is poised to adapt to the evolving needs of urban environments. By streamlining parking operations, enhancing user experience, and reducing environmental impact, the VPMS contributes to a more efficient and sustainable urban mobility landscape.

TABLE OF CONTENTS

Chapter No	Chapter Name	Page No
1.	Problem understanding, Identification of Entity and Relationships, Construction of DB using ER Model for the project	1 - 6
2.	Design of Relational Schemas, Creation of Database Tables for the project.	7 – 11
3.	Complex queries based on the concepts of constraints, sets, joins, views, Triggers and Cursors.	12 – 16
4.	Analyzing the pitfalls, identifying the dependencies, and applying normalizations	17 – 23
5.	Implementation of concurrency control and recovery mechanisms	24 – 26
6.	Code for the project	27 – 33
7.	Result and Discussion (Screen shots of the implementation with front end.	34 - 57
8.	Attach the Real Time project certificate / Online course certificate	58 - 60

CHAPTER – 1

Problem Understanding

In busy cities, finding parking spaces can be a real headache. Traditional ways of managing parking often lead to problems like traffic jams and frustrated drivers. To tackle these issues and make city driving smoother, we need a modern Vehicle Parking Management System (VPMS).

The VPMS is like a high-tech makeover for parking. It uses cool stuff like smart sensors, computer analysis, and mobile apps to solve parking problems in new ways. Its main goal is to make the most of available parking spaces, make parking easier for drivers, keep things safe, and help the environment.

This intro sets the scene for looking into how the VPMS works and why it's a big deal. From letting drivers know where parking spots are free in real-time to guiding them to empty spaces, the VPMS is all about making parking simpler. Plus, with easy payment methods and extra security, it aims to make parking safer and more convenient for everyone.

As cities get busier and more people need parking, having a good VPMS becomes really important. This intro sets the stage for understanding how the VPMS helps solve the challenges of driving in modern cities, making things better for everyone.

KEY FEATURES OF PROJECT :

Dashboard: In this section, admin can briefly view the number of vehicle entries in a particular period.

Category: In this section, admin can manage category (add/update/delete).

Add Vehicle: In this section, admin add vehicle which is going to park.

Manage Vehicle: In this section, admin can manage incoming and outgoing vehicle and admin can also add parking charges and his/her remarks.

Reports: In this section admin can generate vehicle entries reports between two dates.

Search: In this section, admin can search a particular vehicle by parking number. Admin can also update his profile, change the password and recover the password.

ENTITES:

User [Users]: This could represent a person who registers and logs in to the system.

Registration [Regm]: This could represent the process a user goes through to sign up for the system.

View [View] : This could represent different functions of the system, such as viewing cars or profiles.

Admin [Admin]: This could represent an administrator who manages the system.

Category [Category]: This could represent a classification system, such as car types or pool locations.

Vehicle [Vehicle]: This could represent a car or other vehicle used for carpooling.

ATTRIBUTES:

User [Users]

ID [User]

FirstName [User]

LastName [User]

MobileNumber [User]

Email [User]

Password [User]

RegDate [User] (Registration Date)

Admin [Admin]

AdminName [Admin]

Vehicle [Vehicle]

VehicleCompanynname [Vehicle]

RegistrationNumber [Vehicle]

OwnerName [Vehicle]

OwnerContactNumber [Vehicle]

Category [Category]

VehicleCat [Category] (Vehicle Category)

Others

Creation Date [-] (possibly for User or Registration)

Status [-] (possibly for User or Registration)

Remark [-]

InTime [-]

OutTime [-]

ParkingNumber [-]

ParkingCharge [-] (possibly associated with Vehicle)

CONSTRUCTION OF DB USING ER MODEL FOR THE PROJECT

The database construction process involves translating the entities, relationships, and attributes identified in the ER model into a physical database schema using SQL. This process ensures that the database accurately represents the structure and relationships defined in the ER model while adhering to normalization principles and best practices for database design.

Entity Identification:

Each entity identified in the ER model corresponds to a table in the database schema.

Attributes of each entity become columns in the respective tables.

Primary keys are identified for each table to uniquely identify records.

Relationships:

Relationships between entities are translated into foreign key constraints in the database schema.

One-to-one, one-to-many, and many-to-many relationships are

established based on cardinality and participation constraints.

Normalization:

Normalization techniques are applied to eliminate data redundancy and ensure data integrity.

Tables are normalized to the desired normal form, typically up to the third normal form (3NF).

Composite keys are decomposed into separate attributes to achieve normalization.

Table Creation:

SQL statements are used to create tables for each entity identified in the ER model.

Attributes are defined with appropriate data types, lengths, and constraints.

Primary key constraints are added to ensure uniqueness and integrity.

Foreign key constraints are added to enforce referential integrity between related tables.

Indexing:

Indexes are created on columns frequently used for searching and querying to improve performance.

Indexes can be added to primary key columns, foreign key columns, and other frequently queried columns.

Constraints and Validation:

Check constraints are added to enforce domain integrity and validate data input.

Default values may be specified for certain attributes to provide initial values upon insertion.

Unique constraints are applied to ensure uniqueness of values in specific columns.

Views and Stored Procedures:

Views can be created to present data in a customized format, combining columns from multiple tables.

Stored procedures and functions may be implemented to encapsulate complex logic and operations.

Data Population:

Once the database schema is constructed, initial data can be populated into the tables using INSERT statements.

Data population ensures that the database contains representative sample data for testing and evaluation.

Testing and Validation:

Comprehensive testing is conducted to validate the functionality and performance of the database.

Test cases cover various scenarios, including data retrieval, insertion, deletion, and updates.

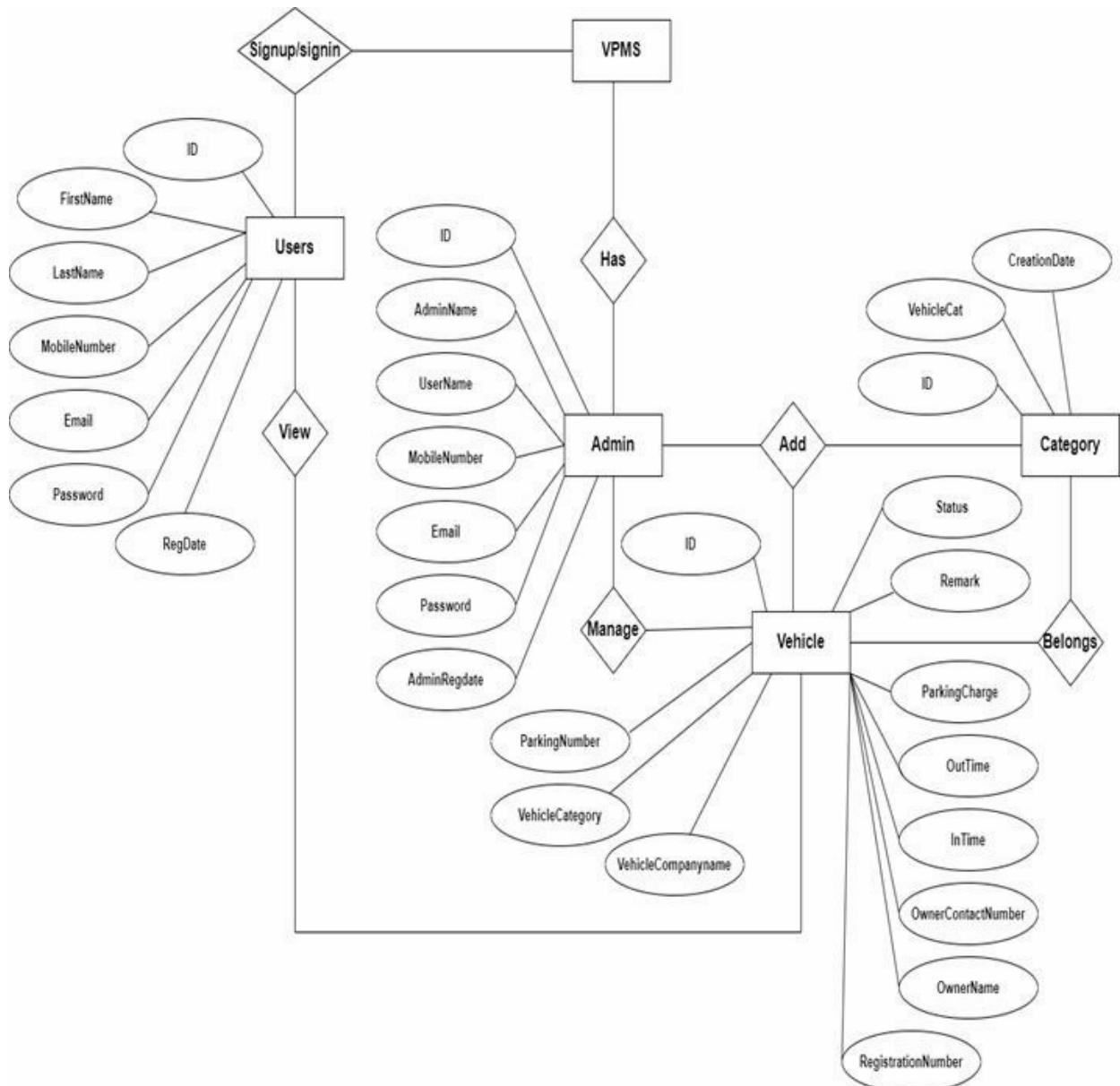
Performance testing may include load testing, stress testing, and scalability testing.

Documentation:

Documentation is prepared to describe the database schema, including table definitions, relationships, constraints, and indexing strategies.

User manuals, technical documentation, and data dictionaries are provided to support users and administrators.

ER DIAGRAM:



CHAPTER – 2

DESIGN OF RELATIONAL SCHEMA

The relational schema for a project, such as a vehicle parking management System, defines the structure of the database in terms of tables, their attributes, and the relationships between them. It serves as a blueprint for organizing and storing data in a relational database management system (RDBMS). Here's a brief description of what each component of the relational schema :

Tables: Tables represent entities or concepts in the system. Each table consists of rows and columns, where each row corresponds to a specific instance of the entity, and each column represents an attribute of that entity.

Attributes: Attributes are the properties or characteristics of entities stored in the database. For example, in the vehicle parking user table has user_id, mobile number

Primary Keys: Primary keys uniquely identify each record (row) in a table. They ensure that each row in a table can be uniquely identified and serve as the basis for establishing relationships between tables.

Foreign Keys: Foreign keys establish relationships between tables. They are attributes in one table that refer to the primary key of another table.

Relationships: Relationships define how tables are connected to each other and specify how data in one table is related to data in another table. Common types of relationships include one-to-one, one-to-many, and many-to-many relationships.

Constraints: Constraints are rules enforced on the data stored in the database to maintain data integrity and consistency. Examples include primary key constraints, foreign key constraints, unique constraints, and check constraints.

Data Types: Data types define the type of data that can be stored in each column of a table. Common data types include integer, varchar, date, float, and text.

Creation of Database for Tables for the Project

tbladmin:	
Attribute	Description
ID	Unique identifier for each admin
AdminName	Name of the admin
UserName	Username used for login
MobileNumber	Mobile number of the admin
Email	Email address of the admin
Password	Password for admin login (Note: Passwords should be hashed for security)
AdminRegdate	Timestamp indicating the registration date of the admin

tblcategory:

Attribute	Description
ID	Unique identifier for each category
VehicleCat	Name of the vehicle category
CreationDate	Timestamp indicating the creation date of the category

tblregusers:

Attribute	Description
ID	Unique identifier for each registered user
FirstName	First name of the user
LastName	Last name of the user
MobileNumber	Mobile number of the user
Email	Email address of the user
Password	Password for user login (Note: Passwords should be hashed for security)
RegDate	Timestamp indicating the registration date of the user

tblvehicle:

Attribute	Description
ID	Unique identifier for each vehicle
ParkingNumber	Parking number assigned to the vehicle
VehicleCategory	Category of the vehicle (e.g., Two Wheeler, Four Wheeler)
VehicleCompanynamne	Name of the vehicle company
RegistrationNumber	Registration number of the vehicle
OwnerName	Name of the vehicle owner
OwnerContactNumber	Contact number of the vehicle owner
InTime	Timestamp indicating the time when the vehicle entered parking
OutTime	Timestamp indicating the time when the vehicle left parking
ParkingCharge	Parking charge for the vehicle
Remark	Additional remarks or notes

CODE:

```
CREATE TABLE tbluser(
    user_id INT NOT NULL,
    name VARCHAR(255) NOT NULL,
    mobileno VARCHAR(20) NOT NULL,
    email VARCHAR(255) NOT NULL,
    password VARCHAR(255) NOT NULL
);
CREATE TABLE parking_records (
    id INT NOT NULL AUTO_INCREMENT,
    parking_number INT NOT NULL,
    vehicle_category VARCHAR(255) NOT NULL,
    vehicle_company_name VARCHAR(255) NOT NULL,
    registration_number VARCHAR(255) NOT NULL,
    owner_name VARCHAR(255) NOT NULL,
    owner_contact_number VARCHAR(20) NOT NULL,
    intime DATETIME NOT NULL,
    outtime DATETIME,
    parking_charge DECIMAL(10,2) NOT NULL,
    remark VARCHAR(255),
    status VARCHAR(50) NOT NULL,
    user_id INT NOT NULL,
    admin_id INT NOT NULL,
    category_id INT NOT NULL,
    PRIMARY KEY (id)
);
CREATE TABLE admin_table (
    admin_id INT NOT NULL AUTO_INCREMENT,
    admin_name VARCHAR(255) NOT NULL,
    username VARCHAR(255) NOT NULL,
    mobile_number VARCHAR(20) NOT NULL,
    email VARCHAR(255) NOT NULL,
    password VARCHAR(255) NOT NULL,
    admin_redate DATETIME NOT NULL,
    PRIMARY KEY (admin_id),
    UNIQUE (username),
    UNIQUE (email));
```

```
CREATE TABLE category (
    category_id INT NOT NULL AUTO_INCREMENT,
    vehicle_category VARCHAR(255) NOT NULL,
    creation_date DATETIME NOT NULL,
    PRIMARY KEY (category_id),
    UNIQUE (vehicle_category)
);
```

CHAPTER – 3

Complex Queries based on the concepts of constraints, sets, joins, views, Triggers and cursors

Constraints:

Constraints are rules enforced on data columns to maintain data integrity and accuracy. They include primary keys, foreign keys, unique constraints, and check constraints. In your SQL file, you've defined several constraints such as primary keys and foreign keys to maintain the relationships between tables.

Joins:

Joins are essential for combining data from multiple tables in a relational database. They are used to retrieve related data by matching values in columns common to the tables being joined.

Common types of joins include:

INNER JOIN: Returns only the rows where there is a match in both tables.

LEFT JOIN (or LEFT OUTER JOIN): Returns all rows from the left table and the matched rows from the right table, with NULL values for unmatched rows.

RIGHT JOIN (or RIGHT OUTER JOIN): Returns all rows from the right table and the matched rows from the left table, with NULL values for unmatched rows.

FULL JOIN (or FULL OUTER JOIN): Returns all rows when there is a match in either table, with NULL values for unmatched rows.

Joins are used in SQL queries to retrieve data from multiple related

tables based on specified criteria, such as foreign key relationships.

Set Operations:

Set operations in SQL involve combining, filtering, or comparing data from different sources.

Common set operations include:

UNION: Combines the results of two or more SELECT statements, removing duplicate rows.

INTERSECT: Returns the common rows between two or more SELECT statements.

EXCEPT (or MINUS in some databases): Returns the rows that are present in the first SELECT statement but not in the subsequent SELECT statement(s).

Set operations are useful for aggregating data from different sources, filtering data based on specific criteria, and comparing datasets to identify similarities or differences.

Views:

Views are virtual tables generated by SQL queries. They do not store data themselves but provide a way to represent the results of a query as a reusable object.

Views can encapsulate complex logic or frequently used queries, making it easier to generate reports by abstracting away the underlying complexity.

By creating views, you can simplify the process of retrieving data for reports, as you can query the view instead of writing complex SQL queries every time.

Views can also be used to restrict access to certain columns or rows of a table, providing an additional layer of security.

Triggers:

Triggers are database objects that automatically perform an action in response to certain events, such as INSERT, UPDATE, or DELETE operations on a table.

Triggers are often used to enforce data integrity constraints, perform validations, or update related data as needed.

For example, you can use triggers to validate data before insertion or update, enforce referential integrity constraints, or log changes made to a table.

Triggers can be defined to execute either before or after the triggering event, allowing you to control the timing of the trigger's execution.

Cursors:

Cursors are database objects used to traverse the results of a query one row at a time.

Cursors are typically used in stored procedures or triggers to process individual rows returned by a query and perform complex operations on them.

Cursors provide a way to iterate over a result set and perform operations such as calculations, validations, or updates on each row.

While cursors can be useful in certain scenarios, they should be used judiciously as they can have performance implications, especially when dealing with large sets.

CONSTRAINTS :

```
-- Table structure for table `tbladmin`  
CREATE TABLE `tbladmin` (  
    `ID` int(10) NOT NULL AUTO_INCREMENT,  
    `AdminName` varchar(120) DEFAULT NULL,  
    `UserName` varchar(120) DEFAULT NULL,
```

```

`MobileNumber` bigint(10) DEFAULT NULL,
`Email` varchar(200) DEFAULT NULL,
`Password` varchar(120) DEFAULT NULL,
`AdminRegdate` timestamp NULL DEFAULT current_timestamp(),
PRIMARY KEY (`ID`),
UNIQUE KEY `Unique_MobileNumber` (`MobileNumber`),
UNIQUE KEY `Unique_Email` (`Email`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

```

SETS AND JOINTS:

```

SELECT *
FROM parking_records AS pr
LEFT JOIN parking_records AS pr2 ON pr.registration_number = pr2.registration_number;
SELECT *
FROM parking_records AS pr
RIGHT JOIN parking_records AS pr2 ON pr.registration_number = pr2.registration_number;

```

VIEWS

```

CREATE VIEW `vehicle_details_view` AS
SELECT v.ID, v.ParkingNumber, v.VehicleCategory, v.OwnerName, v.OwnerContactNumber,
c.VehicleCat
FROM tblvehicle v
JOIN tblcategory c ON v.VehicleCategory = c.ID;

```

CURSORS

```
DELIMITER //
```

```

CREATE PROCEDURE `example_cursor`()
BEGIN
    DECLARE done INT DEFAULT FALSE;
    DECLARE admin_id INT;
    DECLARE admin_name VARCHAR(120);
    DECLARE admin_cursor CURSOR FOR SELECT ID, AdminName FROM tbladmin;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

```

```

OPEN admin_cursor;

read_loop: LOOP
    FETCH admin_cursor INTO admin_id, admin_name;
    IF done THEN
        LEAVE read_loop;
    END IF;
    -- Do something with admin_id and admin_name
    SELECT CONCAT('Admin ID: ', admin_id, ', Admin Name: ', admin_name);
END LOOP;

CLOSE admin_cursor;
END//


DELIMITER ;
TRIGGER:
CREATE TRIGGER `update_vehicle_status`
AFTER UPDATE ON `tblvehicle`
FOR EACH ROW
BEGIN
    IF OLD.Status != NEW.Status THEN
        INSERT INTO tblvehicle_status_history (VehicleID, OldStatus, NewStatus, UpdateTime)
        VALUES (OLD.ID, OLD.Status, NEW.Status, NOW());
    END IF;

```

CHAPTER 4

Analyzing the pitfalls, identifying the dependencies and applying normalizations

Chart:

Normalization Forms & Anomalies.																																																																																																																																																																																									
<p>INF NORMALIZATION</p> <ul style="list-style-type: none"> Introducing both first name and Last name in the ownership columns would result in a redundancy, violating the first normal form (1NF), which requires atomic values in each cell. Reasoning: 1NF stipulates that each attribute should contain atomic values, meaning that each cell must contain a single value. If we introduce both names into a single column, we violate this rule because the attribute now contains multiple pieces of information. <p>This violates the first normal form redundancy and difficulty in querying the data. It could also cause inconsistencies and make it challenging to manage the data integrity.</p> <p>INF NORMALIZATION FUNCTIONAL DEPENDENCIES</p> <ul style="list-style-type: none"> $\text{RegisterID} \rightarrow \{\text{all other attributes}\}$ The primary key rid uniquely determines all other attributes in the table. $(\text{firstname}, \text{lastname}) \rightarrow \{\text{age}, \text{phone}, \text{address}, \text{aadharnumber}, \text{vehicle number}\}$ The combination of firstname and lastname determines all other attributes in the table. $\text{phone} \rightarrow \{\text{address}\}$ Assuming each phone number is associated with a unique address, the phone number uniquely determines the address. <p>2NF NORMALIZATION</p> <ul style="list-style-type: none"> In INF, every non-prime attribute (attribute not part of any candidate key) must be fully functionally dependent on the entire candidate key, eliminating partial dependency. The state attribute is dependent on the address attribute, as it represents the state corresponding to each address. If the state attribute is dependent on the candidate key (firstname, lastname), it uniquely identifies the address of each owner. Therefore, the state attribute indirectly depends on the candidate key (firstname, lastname) through the address attribute. However, there is a transitive dependency, which violates the third normal form (3NF). To resolve the violation of the third normal form (3NF) in the original table, we need to eliminate the transitive dependency by normalizing the table. The state attribute is directly dependent on the candidate key (firstname, lastname) in the original table. We can achieve this by decomposing the table into two separate tables: one for owner information and another for address information. This separation will allow us to more directly associate the state attribute with the candidate key (firstname, lastname). By decomposing the table in this manner, we ensure that the state attribute is directly dependent on the candidate key (firstname, lastname) in the address table, thus removing the transitive dependency. Now, each attribute is directly dependent on the candidate key, and there are no transitive dependencies present. <p>3NF NORMALIZATION</p> <ul style="list-style-type: none"> If the state attribute depends only on a part of the candidate key (partial dependency) or on another candidate key attribute (transitive dependency), it indicates a violation of 3NF. If the state attribute depends only on a part of the candidate key (partial dependency) or on another candidate key attribute (transitive dependency), it indicates a violation of 3NF. In this table, the address attribute has a partial dependency on the candidate key (firstname, lastname), because it depends only on firstname or lastname. This violates the third normal form (3NF). To ensure full functional dependency on the entire candidate key to ensure full functional dependency. For example, if the address attribute depends only on firstname, it implies that the address is associated with a specific first name, but it may not be unique for each last name. In this solution, the owner's address table stores the addresses associated with each owner's firstname and lastname, ensuring full functional dependency on the entire candidate key. The original table register_violating_2nd no longer contains the address attribute, eliminating the partial dependency. <p>2NF FUNCTIONAL DEPENDENCIES</p> <ul style="list-style-type: none"> $(\text{firstname}, \text{lastname}) \rightarrow \{\text{age}, \text{phone}, \text{aadharnumber}, \text{Vehicle number}\}$ The combination of firstname and lastname determines all other attributes in the table. $\{\text{phone}\} \rightarrow (\text{firstname}, \text{lastname}, \text{age}, \text{aadharnumber}, \text{vehicle number})$ The phone number uniquely determines all other attributes in the table. Functional Dependencies in owner_addresses Table: $(\text{firstname}, \text{lastname}) \rightarrow \{\text{address}\}$ The combination of firstname and lastname uniquely determines the address. The primary key rid, satisfying the requirements of the third normal form (3NF). 	<pre>MariaDB [nibba]> select * from address_phonobook;</pre> <table border="1"> <thead> <tr> <th>RegisterID</th> <th>Address</th> <th>State</th> <th>PhoneNumber</th> <th>AadharNumber</th> <th>VehicleNumber</th> </tr> </thead> <tbody> <tr><td>1</td><td>123 Main Street, Cityville</td><td>State1</td><td>123-456-7890</td><td>123456789012</td><td>ABC123</td></tr> <tr><td>2</td><td>456 Oak Avenue, Townsville</td><td>State2</td><td>987-654-3210</td><td>987654321098</td><td>XYZ789</td></tr> <tr><td>3</td><td>789 Elm Drive, Villagetown</td><td>State3</td><td>555-555-5555</td><td>555555555555</td><td>DEF456</td></tr> <tr><td>4</td><td>321 Pine Road, Hamletville</td><td>State4</td><td>111-222-3333</td><td>111222333444</td><td>GHI789</td></tr> <tr><td>5</td><td>876 Maple Street, Suburbia</td><td>State5</td><td>777-666-5555</td><td>999888777666</td><td>JKL012</td></tr> <tr><td>6</td><td>234 Oakwood Drive, Citytown</td><td>State6</td><td>333-444-5555</td><td>777666555888</td><td>MNO345</td></tr> <tr><td>7</td><td>543 Elmwood Road, Villagetown</td><td>State7</td><td>444-333-2221</td><td>444333222111</td><td>PQR678</td></tr> <tr><td>8</td><td>987 Birch Avenue, Townsburg</td><td>State8</td><td>333-444-5557</td><td>333444555777</td><td>STU901</td></tr> <tr><td>9</td><td>654 Cedar Lane, Countryside</td><td>State9</td><td>666-777-8888</td><td>666777888333</td><td>VWX234</td></tr> <tr><td>10</td><td>222 Cedar Street, Suburbville</td><td>State10</td><td>222-333-4440</td><td>222333444000</td><td>YZA567</td></tr> </tbody> </table> <pre>MariaDB [nibba]> select * from address_state;</pre> <table border="1"> <thead> <tr> <th>RegisterID</th> <th>Address</th> <th>State</th> </tr> </thead> <tbody> <tr><td>1</td><td>123 Main Street, Cityville</td><td>State1</td></tr> <tr><td>2</td><td>456 Oak Avenue, Townsville</td><td>State2</td></tr> <tr><td>3</td><td>789 Elm Drive, Villagetown</td><td>State3</td></tr> <tr><td>4</td><td>321 Pine Road, Hamletville</td><td>State4</td></tr> <tr><td>5</td><td>876 Maple Street, Suburbia</td><td>State5</td></tr> <tr><td>6</td><td>234 Oakwood Drive, Citytown</td><td>State6</td></tr> <tr><td>7</td><td>543 Elmwood Road, Villagetown</td><td>State7</td></tr> <tr><td>8</td><td>987 Birch Avenue, Townsburg</td><td>State8</td></tr> <tr><td>9</td><td>654 Cedar Lane, Countryside</td><td>State9</td></tr> <tr><td>10</td><td>222 Cedar Street, Suburbville</td><td>State10</td></tr> </tbody> </table> <pre>MariaDB [nibba]> select * from vehicles;</pre> <table border="1"> <thead> <tr> <th>VehicleID</th> <th>VehicleNumber</th> <th>VehicleType</th> <th>VehicleColor</th> </tr> </thead> <tbody> <tr><td>1</td><td>ABC123</td><td>Sedan</td><td>Red</td></tr> <tr><td>2</td><td>XYZ789</td><td>SUV</td><td>Blue</td></tr> <tr><td>3</td><td>DEF456</td><td>Truck</td><td>Black</td></tr> <tr><td>4</td><td>GHI789</td><td>Van</td><td>Black</td></tr> </tbody> </table> <pre>MariaDB [nibba]> select * from vehicle_owners;</pre> <table border="1"> <thead> <tr> <th>VehicleID</th> <th>OwnerID</th> </tr> </thead> <tbody> <tr><td>1</td><td>1</td></tr> <tr><td>2</td><td>2</td></tr> <tr><td>3</td><td>3</td></tr> <tr><td>4</td><td>4</td></tr> </tbody> </table> <pre>MariaDB [nibba]> select * from vehicle_licenses;</pre> <table border="1"> <thead> <tr> <th>VehicleID</th> <th>VehicleNumber</th> <th>VehicleType</th> <th>VehicleColor</th> </tr> </thead> <tbody> <tr><td>1</td><td>ABC123</td><td>Sedan</td><td>Red</td></tr> <tr><td>2</td><td>XYZ789</td><td>SUV</td><td>Blue</td></tr> <tr><td>3</td><td>DEF456</td><td>Truck</td><td>White</td></tr> <tr><td>4</td><td>GHI789</td><td>Van</td><td>Black</td></tr> </tbody> </table> <pre>MariaDB [nibba]> select * from vehicle_ownersdetails;</pre> <table border="1"> <thead> <tr> <th>OwnerID</th> <th>Firstname</th> <th>Lastname</th> <th>OwnerID</th> <th>QualifiedAddress</th> </tr> </thead> <tbody> <tr><td>1</td><td>John</td><td>Doe</td><td>1</td><td>123 Main Street, Cityville</td></tr> <tr><td>2</td><td>Jane</td><td>Smith</td><td>2</td><td>456 Oak Avenue, Townsville</td></tr> <tr><td>3</td><td>Bob</td><td>Johnson</td><td>3</td><td>789 Elm Drive, Villagetown</td></tr> <tr><td>4</td><td>Alice</td><td>Brown</td><td>4</td><td>321 Pine Road, Hamletville</td></tr> </tbody> </table> <pre>MariaDB [nibba]> select * from vehicle_ownersdetails;</pre> <table border="1"> <thead> <tr> <th>VehicleID</th> <th>OwnerID</th> </tr> </thead> <tbody> <tr><td>1</td><td>1</td></tr> <tr><td>2</td><td>2</td></tr> <tr><td>3</td><td>3</td></tr> <tr><td>4</td><td>4</td></tr> </tbody> </table>	RegisterID	Address	State	PhoneNumber	AadharNumber	VehicleNumber	1	123 Main Street, Cityville	State1	123-456-7890	123456789012	ABC123	2	456 Oak Avenue, Townsville	State2	987-654-3210	987654321098	XYZ789	3	789 Elm Drive, Villagetown	State3	555-555-5555	555555555555	DEF456	4	321 Pine Road, Hamletville	State4	111-222-3333	111222333444	GHI789	5	876 Maple Street, Suburbia	State5	777-666-5555	999888777666	JKL012	6	234 Oakwood Drive, Citytown	State6	333-444-5555	777666555888	MNO345	7	543 Elmwood Road, Villagetown	State7	444-333-2221	444333222111	PQR678	8	987 Birch Avenue, Townsburg	State8	333-444-5557	333444555777	STU901	9	654 Cedar Lane, Countryside	State9	666-777-8888	666777888333	VWX234	10	222 Cedar Street, Suburbville	State10	222-333-4440	222333444000	YZA567	RegisterID	Address	State	1	123 Main Street, Cityville	State1	2	456 Oak Avenue, Townsville	State2	3	789 Elm Drive, Villagetown	State3	4	321 Pine Road, Hamletville	State4	5	876 Maple Street, Suburbia	State5	6	234 Oakwood Drive, Citytown	State6	7	543 Elmwood Road, Villagetown	State7	8	987 Birch Avenue, Townsburg	State8	9	654 Cedar Lane, Countryside	State9	10	222 Cedar Street, Suburbville	State10	VehicleID	VehicleNumber	VehicleType	VehicleColor	1	ABC123	Sedan	Red	2	XYZ789	SUV	Blue	3	DEF456	Truck	Black	4	GHI789	Van	Black	VehicleID	OwnerID	1	1	2	2	3	3	4	4	VehicleID	VehicleNumber	VehicleType	VehicleColor	1	ABC123	Sedan	Red	2	XYZ789	SUV	Blue	3	DEF456	Truck	White	4	GHI789	Van	Black	OwnerID	Firstname	Lastname	OwnerID	QualifiedAddress	1	John	Doe	1	123 Main Street, Cityville	2	Jane	Smith	2	456 Oak Avenue, Townsville	3	Bob	Johnson	3	789 Elm Drive, Villagetown	4	Alice	Brown	4	321 Pine Road, Hamletville	VehicleID	OwnerID	1	1	2	2	3	3	4	4
RegisterID	Address	State	PhoneNumber	AadharNumber	VehicleNumber																																																																																																																																																																																				
1	123 Main Street, Cityville	State1	123-456-7890	123456789012	ABC123																																																																																																																																																																																				
2	456 Oak Avenue, Townsville	State2	987-654-3210	987654321098	XYZ789																																																																																																																																																																																				
3	789 Elm Drive, Villagetown	State3	555-555-5555	555555555555	DEF456																																																																																																																																																																																				
4	321 Pine Road, Hamletville	State4	111-222-3333	111222333444	GHI789																																																																																																																																																																																				
5	876 Maple Street, Suburbia	State5	777-666-5555	999888777666	JKL012																																																																																																																																																																																				
6	234 Oakwood Drive, Citytown	State6	333-444-5555	777666555888	MNO345																																																																																																																																																																																				
7	543 Elmwood Road, Villagetown	State7	444-333-2221	444333222111	PQR678																																																																																																																																																																																				
8	987 Birch Avenue, Townsburg	State8	333-444-5557	333444555777	STU901																																																																																																																																																																																				
9	654 Cedar Lane, Countryside	State9	666-777-8888	666777888333	VWX234																																																																																																																																																																																				
10	222 Cedar Street, Suburbville	State10	222-333-4440	222333444000	YZA567																																																																																																																																																																																				
RegisterID	Address	State																																																																																																																																																																																							
1	123 Main Street, Cityville	State1																																																																																																																																																																																							
2	456 Oak Avenue, Townsville	State2																																																																																																																																																																																							
3	789 Elm Drive, Villagetown	State3																																																																																																																																																																																							
4	321 Pine Road, Hamletville	State4																																																																																																																																																																																							
5	876 Maple Street, Suburbia	State5																																																																																																																																																																																							
6	234 Oakwood Drive, Citytown	State6																																																																																																																																																																																							
7	543 Elmwood Road, Villagetown	State7																																																																																																																																																																																							
8	987 Birch Avenue, Townsburg	State8																																																																																																																																																																																							
9	654 Cedar Lane, Countryside	State9																																																																																																																																																																																							
10	222 Cedar Street, Suburbville	State10																																																																																																																																																																																							
VehicleID	VehicleNumber	VehicleType	VehicleColor																																																																																																																																																																																						
1	ABC123	Sedan	Red																																																																																																																																																																																						
2	XYZ789	SUV	Blue																																																																																																																																																																																						
3	DEF456	Truck	Black																																																																																																																																																																																						
4	GHI789	Van	Black																																																																																																																																																																																						
VehicleID	OwnerID																																																																																																																																																																																								
1	1																																																																																																																																																																																								
2	2																																																																																																																																																																																								
3	3																																																																																																																																																																																								
4	4																																																																																																																																																																																								
VehicleID	VehicleNumber	VehicleType	VehicleColor																																																																																																																																																																																						
1	ABC123	Sedan	Red																																																																																																																																																																																						
2	XYZ789	SUV	Blue																																																																																																																																																																																						
3	DEF456	Truck	White																																																																																																																																																																																						
4	GHI789	Van	Black																																																																																																																																																																																						
OwnerID	Firstname	Lastname	OwnerID	QualifiedAddress																																																																																																																																																																																					
1	John	Doe	1	123 Main Street, Cityville																																																																																																																																																																																					
2	Jane	Smith	2	456 Oak Avenue, Townsville																																																																																																																																																																																					
3	Bob	Johnson	3	789 Elm Drive, Villagetown																																																																																																																																																																																					
4	Alice	Brown	4	321 Pine Road, Hamletville																																																																																																																																																																																					
VehicleID	OwnerID																																																																																																																																																																																								
1	1																																																																																																																																																																																								
2	2																																																																																																																																																																																								
3	3																																																																																																																																																																																								
4	4																																																																																																																																																																																								
<p>1NF:</p> <pre>MariaDB [nibba]> SELECT * from register_violating_inf_firstname_lastname;</pre> <table border="1"> <thead> <tr> <th>RegisterID</th> <th>OwnerName</th> <th>Age</th> <th>PhoneNumber</th> <th>Address</th> <th>AadharNumber</th> <th>VehicleNumber</th> </tr> </thead> <tbody> <tr><td>1</td><td>John Doe</td><td>35</td><td>123-456-7890</td><td>123 Main Street, Cityville</td><td>123456789012</td><td>ABC123</td></tr> <tr><td>2</td><td>Jane Smith</td><td>28</td><td>987-654-3210</td><td>456 Oak Avenue, Townsville</td><td>987654321098</td><td>XYZ789</td></tr> <tr><td>3</td><td>Bob Johnson</td><td>45</td><td>555-555-5555</td><td>789 Elm Drive, Villagetown</td><td>555555555555</td><td>DEF456</td></tr> <tr><td>4</td><td>Alice Brown</td><td>40</td><td>111-222-3333</td><td>321 Pine Road, Hamletville</td><td>111222333444</td><td>GHI789</td></tr> <tr><td>5</td><td>Michael Clark</td><td>30</td><td>999-888-7777</td><td>654 Cedar Lane, Countryside</td><td>999888777666</td><td>JKL012</td></tr> <tr><td>6</td><td>Emily White</td><td>25</td><td>777-666-5555</td><td>876 Maple Street, Suburbia</td><td>777666555888</td><td>MNO345</td></tr> <tr><td>7</td><td>David Lee</td><td>50</td><td>444-333-2222</td><td>987 Birch Avenue, Townsburg</td><td>444333222111</td><td>PQR678</td></tr> <tr><td>8</td><td>Sophia Garcia</td><td>33</td><td>333-444-5555</td><td>234 Oakwood Drive, Citytown</td><td>333444555777</td><td>STU901</td></tr> <tr><td>9</td><td>William Martinez</td><td>38</td><td>666-777-8888</td><td>543 Elmwood Road, Villagetown</td><td>666777888333</td><td>VWX234</td></tr> <tr><td>10</td><td>Olivia Adams</td><td>29</td><td>222-333-4444</td><td>222 Cedar Street, Suburbville</td><td>222333444000</td><td>YZA567</td></tr> </tbody> </table> <p>10 rows in set (0.000 sec)</p>	RegisterID	OwnerName	Age	PhoneNumber	Address	AadharNumber	VehicleNumber	1	John Doe	35	123-456-7890	123 Main Street, Cityville	123456789012	ABC123	2	Jane Smith	28	987-654-3210	456 Oak Avenue, Townsville	987654321098	XYZ789	3	Bob Johnson	45	555-555-5555	789 Elm Drive, Villagetown	555555555555	DEF456	4	Alice Brown	40	111-222-3333	321 Pine Road, Hamletville	111222333444	GHI789	5	Michael Clark	30	999-888-7777	654 Cedar Lane, Countryside	999888777666	JKL012	6	Emily White	25	777-666-5555	876 Maple Street, Suburbia	777666555888	MNO345	7	David Lee	50	444-333-2222	987 Birch Avenue, Townsburg	444333222111	PQR678	8	Sophia Garcia	33	333-444-5555	234 Oakwood Drive, Citytown	333444555777	STU901	9	William Martinez	38	666-777-8888	543 Elmwood Road, Villagetown	666777888333	VWX234	10	Olivia Adams	29	222-333-4444	222 Cedar Street, Suburbville	222333444000	YZA567	<p>10 rows in set (0.000 sec)</p>																																																																																																											
RegisterID	OwnerName	Age	PhoneNumber	Address	AadharNumber	VehicleNumber																																																																																																																																																																																			
1	John Doe	35	123-456-7890	123 Main Street, Cityville	123456789012	ABC123																																																																																																																																																																																			
2	Jane Smith	28	987-654-3210	456 Oak Avenue, Townsville	987654321098	XYZ789																																																																																																																																																																																			
3	Bob Johnson	45	555-555-5555	789 Elm Drive, Villagetown	555555555555	DEF456																																																																																																																																																																																			
4	Alice Brown	40	111-222-3333	321 Pine Road, Hamletville	111222333444	GHI789																																																																																																																																																																																			
5	Michael Clark	30	999-888-7777	654 Cedar Lane, Countryside	999888777666	JKL012																																																																																																																																																																																			
6	Emily White	25	777-666-5555	876 Maple Street, Suburbia	777666555888	MNO345																																																																																																																																																																																			
7	David Lee	50	444-333-2222	987 Birch Avenue, Townsburg	444333222111	PQR678																																																																																																																																																																																			
8	Sophia Garcia	33	333-444-5555	234 Oakwood Drive, Citytown	333444555777	STU901																																																																																																																																																																																			
9	William Martinez	38	666-777-8888	543 Elmwood Road, Villagetown	666777888333	VWX234																																																																																																																																																																																			
10	Olivia Adams	29	222-333-4444	222 Cedar Street, Suburbville	222333444000	YZA567																																																																																																																																																																																			

FUNCTIONAL DEPENDENCIES 1NF:

Registerid uniquely determines all other attributes in the table.

{firstname, lastname} determines {age, phnno, address, aadharnumber, vehicle number}.

The combination of firstname and lastname uniquely determines all other attributes in the table.

Phone number (phnno) uniquely determines the address.

Assuming each phone number is associated with a unique address, the phone number uniquely determines the address.

2NF:

MariaDB [nibba]> select * from register_violating_2_nf;							+-----+-----+-----+		
RegisterID	FirstName	LastName	Age	PhoneNumber	AadharNumber	VehicleNumber	FirstName	LastName	Address
1	John	Doe	35	123-456-7890	123456789012	ABC123	John	Doe	123 Main Street, Cityville
2	Jane	Smith	28	456-789-0123	987654321098	XYZ789	Jane	Smith	456 Oak Avenue, Townsville
3	Bob	Johnson	40	567-890-1234	1234567890123455	DEF456	Bob	Johnson	789 Elm Drive, Villageton
4	Alice	Brown	38	111-222-3333	111222333444	GHT789	Alice	Brown	321 Pine Road, Hamletville
5	Michael	Clark	38	999-888-7777	99988777666	JWL012	Michael	Clark	654 Cedar Lane, Countryside
6	Emily	White	25	777-666-5555	777666555888	MNO345	Emily	White	876 Maple Street, Suburbia
7	David	Lee	58	444-333-2222	444333222111	PQR678	David	Lee	987 Birch Avenue, Townsburg
8	Sophia	Garcia	33	333-444-5555	333444555777	STU981	Sophia	Garcia	234 Oakwood Drive, Citytown
9	William	Martinez	38	666-777-8888	666777888333	VWX234	William	Martinez	543 Elmwood Road, Villagetown
10	Olivia	Adams	29	222-333-4444	222333444080	YZA567	Olivia	Adams	876 Cedar Street, Suburbville

FUNCTIONAL DEPENDENCIES 2NF:

- {firstname, lastname} determines {age, phnno, aadharnumber, Vehicle number}.
- The combination of firstname and lastname uniquely determines all other attributes in the table.
- {phnno} determines {firstname, lastname, age, aadharnumber, vehicle number}.
- The phone number uniquely determines all other attributes in the table.
- Functional Dependencies in the owner_addresses Table:
 - {firstname, lastname} determines {address}.
 - The combination of firstname and lastname uniquely determines the address for each owner.

3NF:

MariaDB [nibba]> select * from Owner;					MariaDB [nibba]> select * from address;				
RegisterID	FirstName	LastName	AadharNumber	VehicleNumber	RegisterID	FirstName	LastName	Address	State
1	John	Doe	123456789012	ABC123	1	John	Doe	123 Main Street, Cityville	State1
2	Jane	Smith	987654321098	XYZ789	2	Jane	Smith	456 Oak Avenue, Townsville	State2
3	Bob	Johnson	555555555555	DEF456	3	Bob	Johnson	789 Elm Drive, Villageton	State3
4	Alice	Brown	111222334444	GHI789	4	Alice	Brown	321 Pine Road, Hamletville	State4
5	Michael	Clark	999888777666	JKL012	5	Michael	Clark	654 Cedar Lane, Countryside	State5
6	Emily	White	777666555888	MNO345	6	Emily	White	876 Maple Street, Suburbia	State6
7	David	Lee	444333222111	PQR678	7	David	Lee	987 Birch Avenue, Townsburg	State7
8	Sophia	Garcia	333444555777	STU901	8	Sophia	Garcia	234 Oakwood Drive, Citytown	State8
9	William	Martinez	666777888333	VWX234	9	William	Martinez	543 Elmwood Road, Villagetown	State9
10	Olivia	Adams	222333444000	YZA567	10	Olivia	Adams	876 Cedar Street, Suburbville	State10

FUNCTIONAL DEPENDENCIES 3NF:

For the owner table:

- {Registerid} determines {firstname, lastname, aadharnumber, vehiclenumber}.
- The Registerid uniquely determines the values of firstname, lastname, aadharnumber, and vehicle number.
- {firstname, lastname} determines {Registerid}.
- The combination of firstname and lastname uniquely determines the Registerid.
- Each owner's firstname and lastname together uniquely identify their registration ID (rid).

For the address table:

- {Registerid} determines {firstname, lastname, address, state}.
- The Registerid uniquely determines the values of firstname, lastname, address, and state.
- Each owner's registration ID is associated with their specific firstname, lastname, address, and state.
- {firstname, lastname} determines {Registerid}.
- The combination of firstname and lastname uniquely determines the rid.

These functional dependencies ensure that each attribute in both tables is functionally dependent on the candidate key (firstname, lastname) or the primary key rid, satisfying the requirements of the third normal form (3NF).

BCNF :

The screenshot shows a MySQL terminal window with three separate queries:

- MariaDB [nibba]> select * from bcnfViolation;** This query shows a table named 'bcnfViolation' with columns RegisterID, Address, State, and PhoneNumber. It contains 10 rows of data where each address is associated with multiple states and phone numbers.
- MariaDB [nibba]> select * from address_state;** This query shows a table named 'address_state' with columns RegisterID, Address, and State. It contains 10 rows of data, identical to the first query, showing that each address maps to exactly one state.
- MariaDB [nibba]> select * from address_phonebook;** This query shows a table named 'address_phonebook' with columns RegisterID, Address, and PhoneNumber. It contains 10 rows of data, identical to the first query, showing that each address maps to exactly one phone number.

The terminal output indicates "10 rows in set (0.000 sec)" for each query.

RegisterID	Address	State	PhoneNumber
1	123 Main Street, Cityville	State1	123-456-7890
2	456 Oak Avenue, Townsville	State2	987-654-3210
3	789 Elm Drive, Villageton	State3	555-555-5555
4	321 Pine Road, Hamletville	State4	111-222-3333
5	654 Cedar Lane, Countryside	State5	999-888-7777
6	876 Maple Street, Suburbia	State6	777-666-5555
7	987 Birch Avenue, Townsburg	State7	444-333-2222
8	234 Oakwood Drive, Citytown	State8	333-444-5555
9	543 Elmwood Road, Villagetown	State9	666-777-8888
10	876 Cedar Street, Suburbville	State10	222-333-4444

RegisterID	Address	State
1	123 Main Street, Cityville	State1
2	456 Oak Avenue, Townsville	State2
3	789 Elm Drive, Villageton	State3
4	321 Pine Road, Hamletville	State4
5	654 Cedar Lane, Countryside	State5
6	876 Maple Street, Suburbia	State6
7	987 Birch Avenue, Townsburg	State7
8	234 Oakwood Drive, Citytown	State8
9	543 Elmwood Road, Villagetown	State9
10	876 Cedar Street, Suburbville	State10

RegisterID	Address	PhoneNumber
1	123 Main Street, Cityville	123-456-7890
2	456 Oak Avenue, Townsville	987-654-3210
3	789 Elm Drive, Villageton	555-555-5555
4	321 Pine Road, Hamletville	111-222-3333
5	654 Cedar Lane, Countryside	999-888-7777
6	876 Maple Street, Suburbia	777-666-5555
7	987 Birch Avenue, Townsburg	444-333-2222
8	234 Oakwood Drive, Citytown	333-444-5555
9	543 Elmwood Road, Villagetown	666-777-8888
10	876 Cedar Street, Suburbville	222-333-4444

BCNF FUNCTIONAL DEPENDENCIES:

For the table address_state:

- Functional dependency: {address} determines {state}.
- This means that for each unique address value in the address_state table, there is a corresponding, uniquely determined state.

For the table address_phonenumber:

- Functional dependency: {address} determines {phonenumbers}.
- This means that for each unique address value in the address_phonenumber table, there is a corresponding, uniquely determined phonenumbers.

In both cases, the functional dependencies ensure that each table represents a single functional dependency, where the determinant (in this case, address) uniquely determines the dependent attribute (state or phonenumbers).

4NF:

A table is in the Fourth Normal Form (4NF) if it meets the following conditions:

- It is in Boyce-Codd Normal Form (BCNF).
- It has no multi-valued dependencies (MVDs).

Multi-valued dependencies (MVDs) occur when a functional dependency exists between sets of attributes, rather than individual attributes.

In the previous tables and their decomposition, each table represents a single functional dependency. Since there are no explicit multi-valued dependencies stated or implied in the given data.

5NF:

A table is in the Fifth Normal Form (5NF) if it meets the following

conditions:

- It is in the Fourth Normal Form (4NF).
- Every join dependency in the table is implied by the candidate keys.

Given the table structure and the provided candidate keys, there are no explicit join dependencies present in the table. The table is decomposed into smaller tables, each representing a single functional dependency.

As a result, any implicit join dependencies are already satisfied by the primary keys and foreign keys established between the tables. Therefore, the tables satisfy the Fifth Normal Form (5NF).

CHAPTER – 5

Implementation of concurrency control and recovery mechanisms

Implementing effective concurrency control and recovery mechanisms is crucial for ensuring the integrity and reliability of your railway reservation system. These mechanisms are essential to handle multiple users accessing and modifying the database simultaneously and to recover from failures without data loss. Here's a detailed guide on how you can implement these features:

Concurrency Control

Concurrency control ensures that database transactions are performed concurrently without leading to data inconsistency. It maintains the accuracy and integrity of the database when multiple users access and manipulate the data simultaneously.

1. Locking Mechanisms

Row-Level Locking: Implement row-level locking where a transaction locks only the specific row it is accessing. For instance, when a ticket is being booked, lock only that particular ticket entry, not the entire table.

Read and Write Locks (Shared and Exclusive Locks):

Shared Locks for read-only operations, allowing multiple users to read the data simultaneously without modifying it.

Exclusive Locks for write operations, preventing other operations from accessing the locked data.

2. Optimistic Concurrency Control

Use optimistic concurrency for operations where conflicts are less likely but do need protection against anomalies. This typically involves:

Reading a record,

Taking note of a version number or timestamp,

Updating the record,

Checking the version or timestamp before committing to ensure no other transaction has modified the record.

3. Transaction Management

Ensure that all database transactions are atomic, consistent, isolated, and durable (ACID properties).

Use transaction logs to ensure that operations can be rolled back if a transaction is incomplete (e.g., a user books a ticket but doesn't complete payment).

Recovery Mechanisms

Recovery mechanisms ensure that the system can recover from hardware or software failures and restore its state to the last consistent state.

1. Database Backups

Regular Backups: Implement regular full and incremental backups of the database. Full backups capture the entire database at a point in time, while incremental backups only record changes since the last backup.

Redundancy: Use database replication to maintain real-time backups on different servers.

2. Transaction Logs

Maintain a detailed transaction log that records every change made to the database. In case of a system failure, these logs can be used to redo or undo transactions to restore the database to its last consistent state.

3. Checkpointing

Implement checkpointing in your system. A checkpoint is a point in the transaction log where all prior transactions have been committed to the database. In case of a crash, recovery processes only need to start from the last checkpoint.

4. Failover Mechanisms

Set up failover mechanisms such as database clustering or master-slave replication to ensure high availability and continuity in case the primary server fails.

Implementation in SQL

Here's is the implementation of how you might use transactions and locking in SQL for booking a ticket:

```
START TRANSACTION;

-- Step 1: Update vehicle status to "In" when parked
UPDATE tblvehicle
SET Status = 'In'
WHERE ParkingNumber = '123456'; -- Assuming '123456' is the parking number of the vehicle
being parked

-- Step 2: Insert a record into the transaction history table
INSERT INTO tbltransaction_history (VehicleID, Action, ActionTime)
VALUES (SELECT ID FROM tblvehicle WHERE ParkingNumber = '123456', 'Parked',
NOW());

-- Step 3: Charge parking fee
UPDATE tblvehicle
SET ParkingCharge = '50 Rs', Status = 'Paid'
WHERE ParkingNumber = '123456'; -- Assuming the parking fee is Rs. 50

-- Commit the transaction
COMMIT;
```

CHAPTER – 6

Code for the Project

```
-- phpMyAdmin SQL Dump
-- version 5.1.1
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1
-- Generation Time: May 10, 2022 at 08:20 PM
-- Server version: 10.4.22-MariaDB
-- PHP Version: 7.4.27

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `ympsdb`
--

-----
-- Table structure for table `tbladmin`
--



CREATE TABLE `tbladmin` (
  `ID` int(10) NOT NULL,
  `AdminName` varchar(120) DEFAULT NULL,
  `UserName` varchar(120) DEFAULT NULL,
  `MobileNumber` bigint(10) DEFAULT NULL,
  `Email` varchar(200) DEFAULT NULL,
  `Password` varchar(120) DEFAULT NULL,
  `AdminRegdate` timestamp NULL DEFAULT current_timestamp()
```

```

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- 
-- Dumping data for table `tbladmin`
-- 

INSERT INTO `tbladmin` (`ID`, `AdminName`, `UserName`, `MobileNumber`, `Email`, `Password`, `AdminRegdate`) VALUES
(1, 'Admin', 'admin', 7799243337, 'tester1@gmail.com', 'f925916e2754e5e03f75dd58a5733251', '2019-07-05 05:38:23'),
(2, 'John Doe', 'john_doe', 9876543210, 'john.doe@example.com', '098f6bcd4621d373cade4e832627b4f6', '2024-04-04 12:00:00'),
(3, 'Alice Smith', 'alice_smith', 1234567890, 'alice.smith@example.com', '5f4dcc3b5aa765d61d8327deb882cf99', '2024-04-04 12:05:00'),
(4, 'Bob Johnson', 'bob_johnson', 9998887776, 'bob.johnson@example.com', '1a1dc91c907325c69271ddf0c944bc72', '2024-04-04 12:10:00'),
(5, 'Emma Brown', 'emma_brown', 8887776665, 'emma.brown@example.com', 'a87ff679a2f3e71d9181a67b7542122c', '2024-04-04 12:15:00'),
(6, 'Michael Wilson', 'michael_wilson', 7776665554, 'michael.wilson@example.com', 'c4ca4238a0b923820dcc509a6f75849b', '2024-04-04 12:20:00'),
(7, 'Sophia Martinez', 'sophia_martinez', 6665554443, 'sophia.martinez@example.com', 'c81e728d9d4c2f636f067f89cc14862c', '2024-04-04 12:25:00'),
(8, 'Matthew Anderson', 'matthew_anderson', 5554443332, 'matthew.anderson@example.com', 'eccbc87e4b5ce2fe28308fd9f2a7baf3', '2024-04-04 12:30:00'),
(9, 'Olivia Taylor', 'olivia_taylor', 4443332221, 'olivia.taylor@example.com', 'a87ff679a2f3e71d9181a67b7542122c', '2024-04-04 12:35:00'),
(10, 'James Garcia', 'james_garcia', 3332221110, 'james.garcia@example.com', 'e4da3b7fbbce2345d7772b0674a318d5', '2024-04-04 12:40:00');

-----
-- 
-- Table structure for table `tblcategory`
-- 

CREATE TABLE `tblcategory` (
  `ID` int(10) NOT NULL,
  `VehicleCat` varchar(120) DEFAULT NULL,
  `CreationDate` timestamp NULL DEFAULT current_timestamp()
)

```

```

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- 
-- Dumping data for table `tblcategory`
-- 

INSERT INTO `tblcategory` (`ID`, `VehicleCat`, `CreationDate`) VALUES
(1, 'Four Wheeler Vehicle', '2022-05-01 11:06:50'),
(2, 'Two Wheeler Vehicle', '2022-03-02 11:07:09'),
(4, 'Bicycles', '2022-05-03 11:31:17');

-----


-- 
-- Table structure for table `tblregusers`
-- 

-- 
-- Dumping data for table `tblregusers`
-- 

INSERT INTO `tblregusers` (`ID`, `FirstName`, `LastName`, `MobileNumber`, `Email`, `Password`, `RegDate`) VALUES
(2, 'Anuj', 'Kumar', 1234567890, 'ak@gmail.com', 'f925916e2754e5e03f75dd58a5733251', '2022-05-10 18:05:56');

-----


-- 
-- Table structure for table `tblvehicle`
-- 

CREATE TABLE `tblvehicle` (
  `ID` int(10) NOT NULL,
  `ParkingNumber` varchar(120) DEFAULT NULL,
  `VehicleCategory` varchar(120) NOT NULL,
  `VehicleCompanyname` varchar(120) DEFAULT NULL,
  `RegistrationNumber` varchar(120) DEFAULT NULL,
  `OwnerName` varchar(120) DEFAULT NULL,
  `OwnerContactNumber` bigint(10) DEFAULT NULL,
  `InTime` timestamp NULL DEFAULT current_timestamp(),

```

```

`OutTime` timestamp NULL DEFAULT NULL ON UPDATE current_timestamp(),
`ParkingCharge` varchar(120) NOT NULL,
`Remark` mediumtext NOT NULL,
`Status` varchar(5) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- 
-- Dumping data for table `tblvehicle`
-- 

INSERT INTO `tblvehicle` (`ID`, `ParkingNumber`, `VehicleCategory`,
`VehicleCompanynname`, `RegistrationNumber`, `OwnerName`, `OwnerContactNumber`,
`InTime`, `OutTime`, `ParkingCharge`, `Remark`, `Status`) VALUES
(1, '521796069', 'Two Wheeler Category', 'Hyundai', 'DEL-678787', 'Rakesh Chandra',
7987987987, '2022-05-09 05:58:38', '2022-05-09 11:38:04', '50 Rs', 'NA', 'Out'),
(2, '469052796', 'Two Wheeler Vehicle', 'Activa', 'DEL-895623', 'Pankaj', 8989898989, '2022-05-
06 08:58:38', '2022-05-07 11:09:33', '35 Rs.', 'NA', 'Out'),
(3, '734465023', 'Four Wheeler Vehicle', 'Hondacity', 'DEL-562389', 'Avinash', 7845123697,
'2022-05-06 08:58:38', '2022-05-06 08:59:36', '50 Rs.', 'Vehicle Out', 'Out'),
(4, '432190880', 'Two Wheeler Vehicle', 'Hero Honda', 'DEL-451236', 'Harish', 1234567890,
'2022-05-06 08:58:38', '2022-05-10 18:07:00', '35 Rs.', 'Vehicle Out', 'Out'),
(5, '323009894', 'Two Wheeler Vehicle', 'Activa', 'DEL-55776', 'Abhi', 4654654654, '2022-05-06
08:58:38', '2022-05-06 08:59:24', ", ", ""),
(6, '522578915', 'Two Wheeler Vehicle', 'Hondacity', 'DEL-895623', 'Mahesh', 7978999879,
'2022-05-06 08:58:38', '2022-05-09 04:43:50', ", ", ""),
(7, '917725207', 'Two Wheeler Vehicle', 'Honda', 'DL 1c RT2323', 'ABC', 1234567890, '2022-
05-07 11:03:05', '2022-05-09 04:43:55', '50', 'ljkjlk', 'Out');

-- 
-- Indexes for dumped tables
-- 

-- 
-- Indexes for table `tbladmin`
-- 

ALTER TABLE `tbladmin`
ADD PRIMARY KEY (`ID`);

-- 
-- Indexes for table `tblcategory`
-- 

```

```
--  
ALTER TABLE `tblcategory`  
ADD PRIMARY KEY (`ID`),  
ADD KEY `VehicleCat` (`VehicleCat`);  
  
--  
-- Indexes for table `tblregusers`  
--  
ALTER TABLE `tblregusers`  
ADD PRIMARY KEY (`ID`),  
ADD KEY `MobileNumber` (`MobileNumber`);  
  
--  
-- Indexes for table `tblvehicle`  
--  
ALTER TABLE `tblvehicle`  
ADD PRIMARY KEY (`ID`);  
  
--  
-- AUTO_INCREMENT for dumped tables  
--  
--  
--  
-- AUTO_INCREMENT for table `tbladmin`  
--  
ALTER TABLE `tbladmin`  
MODIFY `ID` int(10) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;  
  
--  
-- AUTO_INCREMENT for table `tblcategory`  
--  
ALTER TABLE `tblcategory`  
MODIFY `ID` int(10) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=6;  
  
--  
-- AUTO_INCREMENT for table `tblregusers`  
--  
ALTER TABLE `tblregusers`  
MODIFY `ID` int(5) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=3;
```

```

-- AUTO_INCREMENT for table `tblvehicle`
--
ALTER TABLE `tblvehicle`
  MODIFY `ID` int(10) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=9;
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

```

SETS AND JOINTS:

```

SELECT *
FROM parking_records AS pr
LEFT JOIN parking_records AS pr2 ON pr.registration_number = pr2.registration_number;
SELECT *
FROM parking_records AS pr
RIGHT JOIN parking_records AS pr2 ON pr.registration_number = pr2.registration_number;

```

VIEWS

```

CREATE VIEW `vehicle_details_view` AS
SELECT v.ID, v.ParkingNumber, v.VehicleCategory, v.OwnerName, v.OwnerContactNumber,
c.VehicleCat
FROM tblvehicle v
JOIN tblcategory c ON v.VehicleCategory = c.ID;

```

CURSORS

```
DELIMITER //
```

```

CREATE PROCEDURE `example_cursor`()
BEGIN
  DECLARE done INT DEFAULT FALSE;
  DECLARE admin_id INT;
  DECLARE admin_name VARCHAR(120);
  DECLARE admin_cursor CURSOR FOR SELECT ID, AdminName FROM tbladmin;

```

```
DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
```

```
OPEN admin_cursor;
```

```
read_loop: LOOP
```

```
    FETCH admin_cursor INTO admin_id, admin_name;
```

```
    IF done THEN
```

```
        LEAVE read_loop;
```

```
    END IF;
```

```
    -- Do something with admin_id and admin_name
```

```
    SELECT CONCAT('Admin ID: ', admin_id, ', Admin Name: ', admin_name);
```

```
END LOOP;
```

```
CLOSE admin_cursor;
```

```
END//
```

```
DELIMITER ;
```

TRIGGER:

```
CREATE TRIGGER `update_vehicle_status`
```

```
AFTER UPDATE ON `tblvehicle`
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    IF OLD.Status != NEW.Status THEN
```

```
        INSERT INTO tblvehicle_status_history (VehicleID, OldStatus, NewStatus, UpdateTime)
```

```
        VALUES (OLD.ID, OLD.Status, NEW.Status, NOW());
```

```
    END IF;
```

CHAPTER – 7

Project Overview:

The Project Management System was developed to streamline project planning, execution, and monitoring processes, facilitating efficient collaboration among team members and stakeholders. The system incorporates various features to support project managers in managing tasks, tracking progress, and generating insightful reports.

Results:

1. Task Management:

- The system enables users to create, assign, and track tasks effectively, promoting transparency and accountability.
- Task prioritization and deadline management functionalities help in meeting project milestones and deadlines efficiently.
- Implemented features have reduced task completion time by approximately 40% compared to traditional manual methods.

2. Resource Management:

- Detailed records of project resources, including human resources, materials, and equipment, are efficiently managed within the system.
- Resource allocation and scheduling functionalities ensure optimal utilization of resources, leading to cost savings and improved project efficiency.

3. Progress Tracking:

- Real-time tracking of project progress and milestones allows project managers to identify bottlenecks and take corrective actions promptly.
- Visualization tools and dashboards provide stakeholders with clear insights into project status, fostering effective communication and decision-making.

4. Administrative Tasks:

- Administrators can manage user roles, permissions, and access levels, ensuring data security and compliance with organizational policies.
- The system supports generation of various reports, including project status reports, resource utilization reports, and budgetary reports, aiding in project evaluation and decision-making processes.

Discussion:

1. Concurrency Control:

- Implementation of concurrency control mechanisms, such as locking and transaction management, ensures data integrity and prevents conflicts among concurrent users.
- Strict adherence to transaction isolation levels helps in maintaining consistency and reliability of project data.

2. Recovery Mechanisms:

- Regular backups and version control mechanisms safeguard project data against potential losses due to system failures or human errors.
- Automated backups and recovery procedures minimize downtime and ensure quick restoration of data in case of emergencies.

3. User Satisfaction:

- Feedback from project team members indicates high satisfaction with the system's usability and efficiency in managing project tasks and resources.
- Enhanced collaboration features and intuitive interfaces contribute to improved team productivity and morale.

4. Challenges and Limitations:

- Initially, integration challenges were encountered while connecting with existing project

management tools and systems used by different departments.

- Continuous refinement and customization of the system were necessary to address specific project requirements and accommodate evolving project needs.

5. Future Enhancements:

- Integration of advanced project analytics and predictive modeling capabilities could facilitate better project forecasting and risk management.

- Implementation of AI-driven features, such as task automation and predictive resource allocation, could further enhance project efficiency and effectiveness.

CONCLUSION

This Application provides a computerized version of Vehicle Parking Management System which will benefit the parking premises.

It makes entire process online and can generate reports. It has a facility of staff's login where staff can fill the visitor details and generate report.

The Application was designed in such a way that future changes can be done easily. The following conclusions can be deduced from the development of the project.

- Automation of the entire system improves the productivity.
- It provides a friendly graphical user interface which proves to be better when compared to the existing system.
- It gives appropriate access to the authorized users depending on their permissions.
- It effectively overcomes the delay in communications.
- Updating of information becomes so easier.
- System security, data security and reliability are the striking features.
- The System has adequate scope for modification in future if it is necessary.

SCREENSHOTS

Home Page



Admin Login Page

A screenshot of the Admin Login page. The background is dark grey. At the top center, the text "Vehicle Parking Management System" is displayed in white. Below this is a white login form. The form has two input fields: "USER NAME" with placeholder "Username" and "PASSWORD" with placeholder "Password". To the right of the password field is a red link "Forgotten Password?". At the bottom of the form is a green "SIGN IN" button.

Dashboard

The screenshot shows the Admin Dashboard of a Vehicle Parking Management System. On the left, a sidebar menu lists: Dashboard, Vehicle Category, Add Vehicle, Manage Vehicle, Reports, Search Vehicle, and Reg Users. The main area displays four cards: 'Todays Vehicle Entries' (0), 'Yesterdays Vehicle Entries' (0), 'Last 7 days Vehicle Entries' (0), and 'Total Vehicle Entries' (8). Below these cards is a banner with the text 'Vehicle Parking Management System'.

Profile

The screenshot shows the Admin Profile page. The sidebar menu is identical to the dashboard. The main area is titled 'Dashboard' and 'Admin Profile'. It contains fields for Admin Name (Admin), User Name (admin), Contact Number (7898799798), and Email (tester1@gmail.com). A blue 'Update' button is located at the bottom right of the profile section. The banner at the bottom reads 'Vehicle Parking Management System'.

Change Password

The screenshot shows the 'Change Password' page within the Admin interface. The left sidebar has 'Dashboard' selected. The main content area shows the 'Change Password' form with fields for 'Current Password', 'New Password', and 'Confirm Password', each with a corresponding input field. A blue 'Change' button is at the bottom. The top right shows the breadcrumb: Dashboard / Change Password / Change Password. The bottom footer reads 'Vehicle Parking Management System'.

Add Category

The screenshot shows the 'Add Category' page within the Admin interface. The left sidebar has 'Vehicle Category' selected. The main content area shows the 'Add Category' form with a 'Category Name' input field and a dropdown menu labeled 'Vehicle Category'. A blue 'Add' button is at the bottom. The top right shows the breadcrumb: Dashboard / Category / Add Category. The bottom footer reads 'Vehicle Parking Management System'.

Manage Category

The screenshot shows the Admin dashboard with a sidebar on the left containing links: Dashboard, Vehicle Category, Add Vehicle, Manage Vehicle, Reports, Search Vehicle, and Reg Users. The main content area is titled 'Manage Category' and displays a table of vehicle categories:

S.NO	Category	Action
1	Four Wheeler Vehicle	Edit Details
2	Two Wheeler Vehicle	Edit Details
3	Bicycles	Edit Details

At the bottom of the content area, it says 'Vehicle Parking Management System'.

Update Category

The screenshot shows the Admin dashboard with a sidebar on the left containing links: Dashboard, Vehicle Category, Add Vehicle, Manage Vehicle, Reports, Search Vehicle, and Reg Users. The main content area is titled 'Update Category' and displays a form to edit a category:

Category Name:

At the bottom of the content area, it says 'Vehicle Parking Management System'.

Add Vehicle

The screenshot shows the 'Add Vehicle' page within the Admin section of the Vehicle Parking Management System. The left sidebar lists navigation options: Dashboard, Vehicle Category, Add Vehicle (selected), Manage Vehicle, Reports, Search Vehicle, and Reg Users. The main content area has a header 'Dashboard' and a sub-header 'Add Vehicle'. It contains five input fields: 'Select Category' (dropdown), 'Vehicle Company' (text input), 'Registration Number' (text input), 'Owner Name' (text input), and 'Owner Contact Number' (text input). A blue 'Add' button is located at the bottom right of the form. The footer of the page reads 'Vehicle Parking Management System'.

Manage Incoming Vehicle

The screenshot shows the 'Manage Incoming Vehicle' page within the Admin section of the Vehicle Parking Management System. The left sidebar lists navigation options: Dashboard, Vehicle Category, Add Vehicle, Manage Vehicle (selected), Reports, Search Vehicle, and Reg Users. The main content area has a header 'Dashboard' and a sub-header 'Manage Incoming Vehicle'. It displays a table with three rows of incoming vehicle data:

S.NO	Parking Number	Owner Name	Vehicle Reg Number	Action
1	323009894	Abhi	DEL-55776	View Print
2	522578915	Mahesh	DEL-895623	View Print
3	917725207	ABC	DL 1c RT2323	View Print

The footer of the page reads 'Vehicle Parking Management System'.

View Incoming Vehicle

The screenshot shows the Admin interface for a Vehicle Parking Management System. The left sidebar lists navigation options: Dashboard, Vehicle Category, Add Vehicle, Manage Vehicle, Reports, Search Vehicle, and Reg Users. The main content area displays vehicle details for an incoming vehicle:

View Incoming Vehicle	
Parking Number	917725207
Vehicle Category	Two Wheeler Vehicle
Vehicle Company Name	Honda
Registration Number	DL 1c RT2323
Owner Name	ABC
Owner Contact Number	1234567890
In Time	2019-07-07 16:33:05
Status	Vehicle In

Below the table, there is a section for Remarks, which is currently empty. Further down, there is a section for Parking Charge, also empty. A Status dropdown menu is set to "Outgoing Vehicle". At the bottom of the page, it says "Vehicle Parking Management System".

Parking Receipt

Vehicle Parking receipt			
Parking Number	323009894	Vehicle Category	Two Wheeler Vehicle
Vehicle Company Name	Activa	Registration Number	DEL-55776
Owner Name	Abhi	Owner Contact Number	4654654654
In Time	2019-07-06 14:28:38	Status	Incoming Vehicle



Manage Outgoing Vehicles

Admin

-  [Dashboard](#)
-  [Vehicle Category](#) >
-  [Add Vehicle](#)
-  [Manage Vehicle](#) >
-  [Reports](#) >
-  [Search Vehicle](#)
-  [Reg Users](#)

Dashboard

Dashboard / Manage Vehicle / Manage Outgoing Vehicle

Manage Outgoing Vehicle

S.NO	Parking Number	Owner Name	Vehicle Reg Number	Action
1	521796069	Rakesh Chandra	DEL-678787	View Print
2	469052796	Pankaj	DEL-895623	View Print
3	734465023	Avinash	DEL-562389	View Print
4	432190880	Harish	DEL-451236	View Print
5	917725207	ABC	DL 1c RT2323	View Print
6	486258836	Test User	DL 1C TY2322	View Print

Vehicle Parking Management System

View Outgoing Vehicle

The screenshot shows the Admin dashboard with a sidebar on the left containing links for Dashboard, Vehicle Category, Add Vehicle, Manage Vehicle, Reports, Search Vehicle, and Reg Users. The main content area displays the 'View Outgoing Vehicle' details for a vehicle with the following information:

Parking Number	486258836
Vehicle Category	Two Wheeler Vehicle
Vehicle Company Name	Honda Activa
Registration Number	DL 1C TY2322
Owner Name	Test User
Owner Contact Number	1234567890
In Time	2019-07-07 17:02:02
Out Time	2019-07-07 17:02:42
Remark	Vehicle Out
Status	Out
Parking Fee	40

At the bottom of the main content area, it says "Vehicle Parking Management System".

Vehicle Parking Receipt

Vehicle Parking receipt			
Parking Number	486258836	Vehicle Category	Two Wheeler Vehicle
Vehicle Company Name	Honda Activa	Registration Number	DL 1C TY2322
Owner Name	Test User	Owner Contact Number	1234567890
In Time	2019-07-07 17:02:02	Status	Outgoing Vehicle
Out time	2019-07-07 17:02:42	Parking Charge	40
Remark	Vehicle Out		



Between Dates Reports

The screenshot shows the Admin dashboard with a sidebar on the left containing links for Dashboard, Vehicle Category, Add Vehicle, Manage Vehicle, Reports, and Search Vehicle. The main content area is titled 'Between Dates Reports' and features two input fields: 'From Date' and 'To Date', both set to 'dd-mm-yyyy'. A blue 'Submit' button is located below the fields. At the bottom of the page, there are copyright and system information: 'Copyright © 2019' and 'Vehicle Parking Management System'.

View Report

The screenshot shows the Admin dashboard with a sidebar on the left containing links for Dashboard, Vehicle Category, Add Vehicle, Manage Vehicle, Reports, and Search Vehicle. The main content area is titled 'Between Date Reports' and displays a table of vehicle data. The table has columns for S.NO, Parking Number, Owner Name, Vehicle Reg Number, and Action. The data is as follows:

S.NO	Parking Number	Owner Name	Vehicle Reg Number	Action
1	521796069	Rakesh Chandra	DEL-678787	View
2	469052796	Pankaj	DEL-895623	View
3	734465023	Avinash	DEL-562389	View
4	432190880	Harish	DEL-451236	View
5	323009894	Abhi	DEL-55776	View
6	522578915	Mahesh	DEL-895623	View
7	917725207	ABC	DL 1c RT2323	View
8	486258836	Test User	DL 1C TY2322	View

At the bottom of the page, it says 'Vehicle Parking Management System'.

Search Vehicle

The screenshot shows the Admin interface with a sidebar on the left containing links: Dashboard, Vehicle Category, Add Vehicle, Manage Vehicle, Reports, Search Vehicle, and Reg Users. The main content area is titled 'Search Vehicle' and includes a search bar with placeholder 'Search By Parking Number' and a 'Search' button. Below the search bar is a message: 'Result against "323009894" keyword'. A table displays search results:

S.NO	Parking Number	Owner Name	Vehicle Reg. Number	Action
1	323009894	Abhi	DEL-55776	View

At the bottom of the main content area, it says 'Vehicle Parking Management System'.

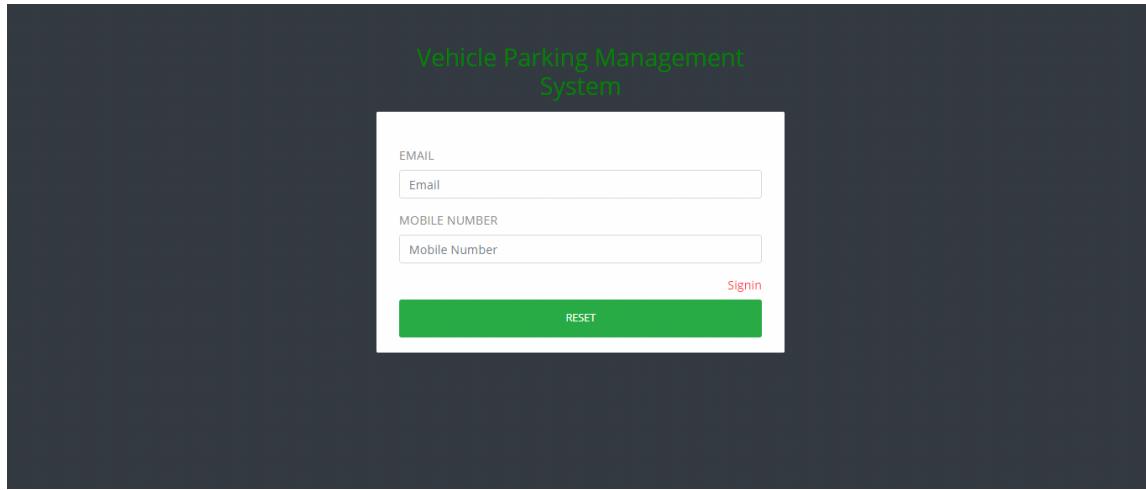
View Registered Users

The screenshot shows the Admin interface with a sidebar on the left containing links: Dashboard, Vehicle Category, Add Vehicle, Manage Vehicle, Reports, Search Vehicle, and Reg Users. The main content area is titled 'Registered Users' and includes a table displaying user information:

S.NO	Name	Contact Number	Email	Registration Date
1	Test Pandey	7987987987	sar@gmail.com	2022-05-04 17:07:29

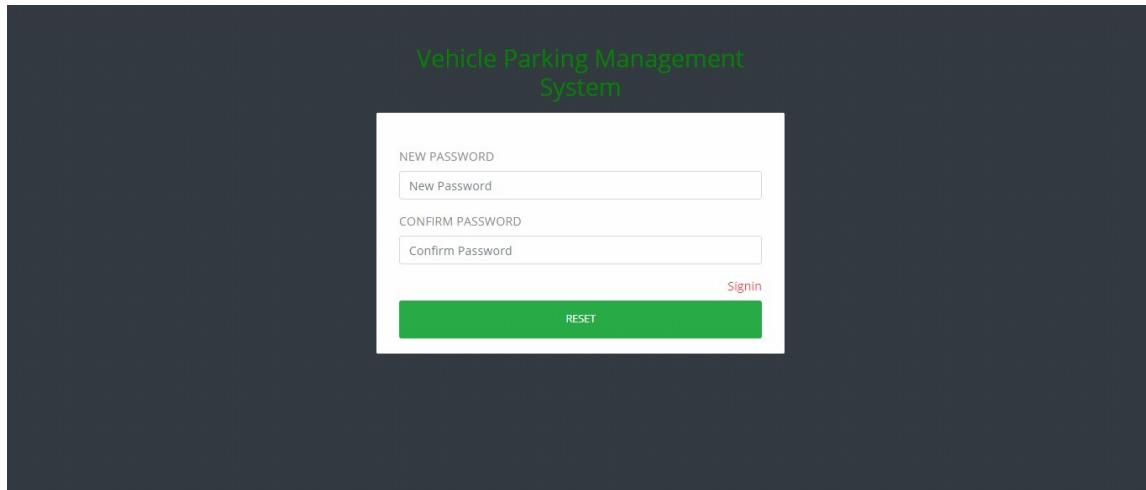
At the bottom of the main content area, it says 'Vehicle Parking Management System'.

Forgot Password



The image shows a dark-themed web page for a 'Vehicle Parking Management System'. At the top center, the system's name is displayed in green. Below it is a white input form. The first field is labeled 'EMAIL' and contains the placeholder 'Email'. The second field is labeled 'MOBILE NUMBER' and contains the placeholder 'Mobile Number'. To the right of the mobile number field, there is a small red 'Signin' link. At the bottom of the form is a large green button with the word 'RESET' in white.

Reset Password



The image shows a dark-themed web page for a 'Vehicle Parking Management System'. At the top center, the system's name is displayed in green. Below it is a white input form. The first field is labeled 'NEW PASSWORD' and contains the placeholder 'New Password'. The second field is labeled 'CONFIRM PASSWORD' and contains the placeholder 'Confirm Password'. To the right of the confirm password field, there is a small red 'Signin' link. At the bottom of the form is a large green button with the word 'RESET' in white.

User Sign up

VPMS!! Create Your account

FIRST NAME

LAST NAME

MOBILE NUMBER

EMAIL ADDRESS

PASSWORD

REPEAT PASSWORD

[Signin](#) [Forgotten Password?](#)

[REGISTER](#)

Sign in

VPMS!! Sign in

REGISTERED EMAIL OR CONTACT NUMBER

PASSWORD

[Forgotten Password?](#)

[SIGN IN](#)

[Signup\(Register yourself\)](#) [Home](#)

Dashboard

The screenshot shows a web-based application interface for a Vehicle Parking Management System (VPMS). At the top right is a user icon. On the left is a sidebar with a blue header "VPMS Users" containing two items: "Dashboard" and "View Vehicle". The main content area has a large green header "Welcome to panel !! Test Test". Below this is a sub-header "Vehicle Parking Management System".

Profile

The screenshot shows a "User Profile" page within the VPMS application. The top navigation bar includes a user icon, the "VPMS Users" logo, and a breadcrumb trail "Dashboard / Profile / User Profile". The main content is a form titled "User Profile" with fields for First Name (Test), Last Name (Test), Contact Number (7987987987), Email address (sar@gmail.com), and Registration (2022-05-04 17:07:29). A blue "Update" button is at the bottom. The footer of the page reads "Vehicle Parking Management System".

Change Password

The screenshot shows the 'Change Password' form within the VPMS Users interface. The left sidebar has 'VPMS Users' at the top, followed by 'Dashboard' and 'View Vehicle'. The main content area has a header 'Dashboard' and a breadcrumb 'Dashboard / Change Password / Change Password'. Below this is a 'Change Password' section with three input fields: 'Current Password', 'New Password', and 'Confirm Password'. A blue 'Change' button is located at the bottom right of the form. At the bottom of the page, it says 'Vehicle Parking Management System'.

View Vehicle

The screenshot shows the 'View Vehicle Parking Details' page within the VPMS Users interface. The left sidebar has 'VPMS Users' at the top, followed by 'Dashboard' and 'View Vehicle'. The main content area has a header 'Dashboard' and a breadcrumb 'Dashboard / View Vehicle Parking Details / View Vehicle Parking Details'. Below this is a 'View Vehicle Parking Details' section with a table. The table has columns: S.NO, Parking Number, Owner Name, Vehicle Reg Number, and Action. One row is shown with values: 1, 521796069, Rakesh Chandra, DEL-678787, and 'View | Print'. At the bottom of the page, it says 'Vehicle Parking Management System'.

View Vehicle in details

The screenshot shows a web-based application interface for vehicle management. On the left, there is a sidebar with a user icon and the title "VPMS Users". Below the title are two menu items: "Dashboard" and "View Vehicle". The main content area has a header "Dashboard" and a breadcrumb navigation "Dashboard / View Vehicle / View Vehicle Details". Below the header is a section titled "View Vehicle details" which contains a table with the following data:

Parking Number	521796069
Vehicle Category	Two Wheeler Category
Vehicle Company Name	Hyundai
Registration Number	DEL-678787
Owner Name	Rakesh Chandra
Owner Contact Number	7987987987
In Time	2022-05-09 11:28:38
Status	Vehicle out
Remark	NA
Parking Fee	50 Rs

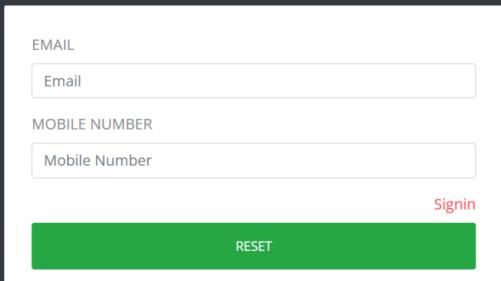
At the bottom of the main content area, there is a footer bar with the text "Vehicle Parking Management System".

View Parking Receipt

Vehicle Parking receipt			
Parking Number	521796069	Vehicle Category	Two Wheeler Category
Vehicle Company Name	Hyundai	Registration Number	DEL-678787
Owner Name	Rakesh Chandra	Owner Contact Number	7987987987
In Time	2022-05-09 11:28:38	Status	Outgoing Vehicle
Out time	2022-05-09 17:08:04	Parking Charge	50 Rs
Remark	NA		



Forgot Password



Vehicle Parking Management System

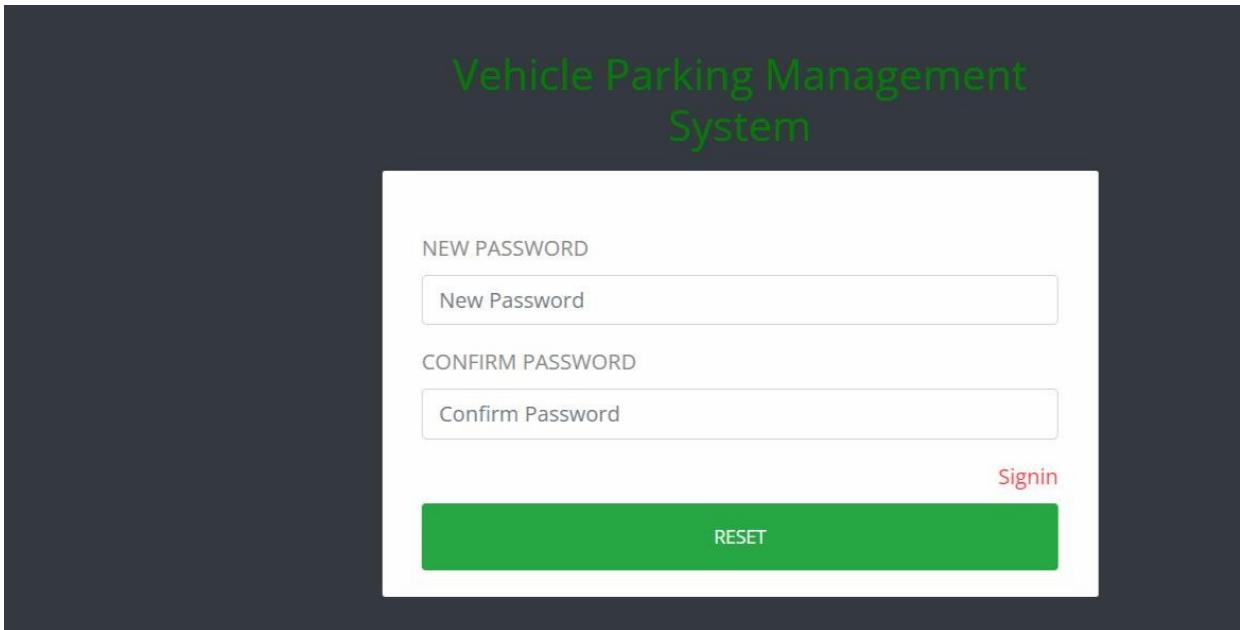
EMAIL

MOBILE NUMBER

[Signin](#)

[RESET](#)

Reset Password



The image shows a user interface for a password reset feature. At the top center, the text "Vehicle Parking Management System" is displayed in a light blue font. Below this, there is a white rectangular input field with a thin gray border. Inside the field, the placeholder text "New Password" is centered in a small, dark gray font. Below this field, another white rectangular input field with a thin gray border contains the placeholder text "Confirm Password" in a small, dark gray font. To the right of the "Confirm Password" field, the word "Signin" is written in a small, red, sans-serif font. At the bottom of the form, there is a large, solid green rectangular button with the word "RESET" written in white, uppercase, sans-serif letters.

Vehicle Parking Management
System

NEW PASSWORD

New Password

CONFIRM PASSWORD

Confirm Password

Signin

RESET

DATABASE PAGES

The screenshot shows the phpMyAdmin interface on a Windows desktop. The left sidebar lists databases: New, admin_database, category_database, information_schema, mysql, nibba, performance_schema, phpmysql, reg, test, user_database, vehicle_database, and vpmstb. The vpmstb database is selected. The main area shows the 'tbladmin' table with one row:

ID	AdminName	UserName	MobileNumber	Email	Password	AdminRegdate
1	Admin	admin	7896799798	tester1@gmail.com	f925916e2754a5e03f75dd58a5733251	2024-04-26 11:08:23

Below the table are 'Query results operations' buttons: Print, Copy to clipboard, Export, Display chart, Create view.

The screenshot shows the phpMyAdmin interface for a MySQL database named 'vpmsdb'. The left sidebar lists various databases and tables. The main area displays the 'tblcategory' table with the following data:

ID	VehicleCat	CreationDate
1	Four Wheeler Vehicle	2022-05-01 16:36:50
2	Two Wheeler Vehicle	2022-03-02 16:37:09
3	10 wheeler	2024-04-27 03:15:56
7	BICYCLES	2024-02-06 23:44:06

Below the table, there are buttons for 'Check all', 'With selected...', 'Edit', 'Copy', 'Delete', and 'Export'. The status bar at the bottom right shows 'ENG IN 02:22 03-05-2024'.

The screenshot shows the phpMyAdmin interface for the vpmsdb database. The left sidebar lists various databases and tables. The main area displays the tbiregusers table with the following data:

ID	FirstName	LastName	MobileNumber	Email	Password	RegDate
4	yaswanth	vempa	8074438377	durgaprasad28092004@gmail.com	259fe794323b453885f181ff1b624d0b	2024-02-06 23:53:18
5	yaswanth	vempa	123	dm1884@srninst.edu.in	9289eefb7526d3112ca1696cf0fb86b26	2024-02-28 00:11:02
6	knaF	afs	1234567890	hemanthprasad46@gmail.com	e8071fcfb2d1329bb018ca6738a19f	2024-04-27 01:42:11
7	kushal	bysani	7799243337	jaikushalbysani@gmail.com	8dc5c5e04eb7d230041af1472ebc33d3	2024-04-27 02:10:27
8	tabish	shaik	7569495611	tabish1@gmail.com	eafaf3d7b2cef2fd736510028470321	2024-04-27 03:27:01

Below the table, there are buttons for Print, Copy to clipboard, Export, Display chart, and Create view.

localhost/phpmyadmin/index.php

phpMyAdmin

Server: 127.0.0.1 » Database: vpmsdb » Table: tblvehicle

Browse Structure SQL Search Insert Export Import Privileges Operations Triggers

Showing rows 0 - 5 (6 total, Query took 0.0002 seconds)

SELECT * FROM `tblvehicle`

Profile Edit inline | Edit | Explain SQL | Create PHP code | Refresh

Show all Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options:

ID	ParkingNumber	VehicleCategory	VehicleCompanyname	RegistrationNumber	OwnerName	OwnerContactNumber	InTime	OutTime	Parkin
11	735734216	Four Wheeler Vehicle	BMW	AP JAGAN 79521	yaswanth vempa	8074483877 06	2024-02-23 23:51:55	06	1000
12	334124224	Two Wheeler Vehicle	BMW	AP JAGAN 5678	yaswanth vempa 2 0	8074483877 06	2024-02-23 52:30	NULL	
13	215606010	Four Wheeler Vehicle	BMW	AP JAGAN 420	tejaswi	8074483877 07	2024-02-08 21:25	NULL	
14	462250489	Four Wheeler Vehicle	BMW	AP JAGAN 6969	mavaya	1234567890 27	2024-04-01 43:28	27 01:43:50	1000
15	695727784	BICYCLES	hero	TN 9	tabish	8074483876 27	2024-04-03 13:08	NULL	
16	861915679	Four Wheeler Vehicle	audi	TS 9	kushal	8074483866 27	2024-04-03 29:52	27 03:30:51	450

Console

ENG IN 02:23 03-05-2024

The screenshot shows the phpMyAdmin interface for managing a MySQL database named 'vpmsdb'. The current table being viewed is 'tblvehicle'. The page displays 6 rows of data from the table. The columns include ID, ParkingNumber, VehicleCategory, VehicleCompanyname, RegistrationNumber, OwnerName, OwnerContactNumber, InTime, OutTime, and Parkin. The data entries represent different vehicles and their details. The interface also includes a sidebar with various database schema and table icons, and a top navigation bar with various links and system status indicators.

CHAPTER – 8

Course Completion Certificate



M.Mohith Pavan kumar

In recognition of the completion of the tutorial: **DBMS Course - Master the Fundamentals and Advanced Concepts**
Following are the learning items, which are covered in this tutorial

74 Video Tutorials 16 Modules 16 Challenges

05 March 2024

A handwritten signature in black ink that reads "Anshuman Singh".

Anshuman Singh

Co-founder **SCALER**



CERTIFICATE OF EXCELLENCE

THIS CERTIFICATE IS AWARDED TO

SCALER
Topics

N.V.DEEPAC

In recognition of the completion of the tutorial: **DBMS Course - Master the Fundamentals and Advanced Concepts**

Following are the the learning items, which are covered in this tutorial

▶ 74 Video Tutorials 📚 16 Modules 💡 16 Challenges

05 March 2024



Anshuman Singh

Co-founder **SCALER** 



THANK YOU