VISVESVARAYA TECHNOLOGICAL UNIVERSITY

BELAGAVI-590018

A DBMS Mini Project Report

on

" Placement Experience Database "

Submitted in partial fulfilment of the requirements for the V semester

B.E in Computer Science and Engineering

of Visvesvaraya Technological University, Belagavi

Submitted by:

Eshwar K	*1RN21CS056*
Dhanush M	*1RN20CS049*

Under the Guidance of:

Dr. A N Ramya Shree
Asst. Professor
Dept. of CSE

ESTD : 2001
An Institute with a Difference

Department of Computer Science and Engineering
(Accredited by NBA upto 30.06.2025)
RNS Institute of Technology
Channasandra, Dr.Vishnuvardhan Road, Bengaluru-560 098
2023-2024

RNS Institute of Technology

Channasandra, Dr.Vishnuvardhan Road, Bengaluru-560098

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

(Accredited by NBA upto 30.06.2025)

CERTIFICATE

Certified that the mini project work entitled **“Placement Experience Database”**has been successfully carried out by **”Eshwar K** bearing USN **”1RN20CS056”** and **“Dhanush M** bearing USN **”1RN20CS049”**, bonafide students of **”RNS Institute of Technology”** in partial fulfilment of the requirements for the 5th semester of **”Bachelor of Engineering in Computer Science and Engineering of Visvesvaraya Technological University”**, Belagavi, during the academic year 2023-2024. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the DBMS laboratory requirements of 5th semester BE, CSE.

Signature of the Guide	Signature of the HoD	Signature of the Principal
Dr.A N Ramya Shree	**Dr. Kiran P**	**Dr. Ramesh Babu H S**
Associate Professor	Professor and HOD	Principal
Dept. of CSE	Dept. of CSE	

External Viva:

Name of the Examiners	Signature with Date
1.	
2.	

Acknowledgement

Any achievement does not depend solely on individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. Several personalities, in their capacities, have helped us to carry out this project work. We want to take this opportunity to thank them all.

We would like to profoundly thank the **Management of RNS Institute of Technology** for providing a healthy environment for the successful completion of this project work.

We are grateful to our Director **Dr. M K Venkatesha**,and Principal **Dr. Ramesh Babu H S**, RNSIT, Bangalore, for their support towards the completion of this mini project.

We would like to thank **Dr. Kiran P**, Professor & Head, Department of Computer Science & Engineering, RNSIT, Bangalore, for his valuable suggestions and expert advice.

We deeply express our sincere gratitude to our guide **Dr. A N Ramya Shree**, Asst. Professor, Department of CSE, RNSIT, Bangalore, for her able guidance, regular source of encouragement and assistance throughout this project.

We would like to thank all the teaching and non-teaching staff of Department of Computer Science & Engineering, RNSIT, Bengaluru-98 for their constant support and encouragement.

Abstract

The Traffic Violation Tracking System (TVTS) is a comprehensive solution designed to enhance the efficiency of traffic law enforcement and violation management by automating various processes and reducing manual paperwork. The system provides a user-friendly interface for traffic police officers to manage tasks related to traffic violations seamlessly.

TVTS aims to streamline the process of issuing traffic violation invoices, maintaining accurate records of violations and payments, and generating insightful reports for better decision-making. By automating routine tasks, such as recording violations, managing fines, and tracking violators' information, TVTS enables law enforcement agencies to focus more on enforcing traffic rules effectively and promoting road safety.

Key features of TVTS include:

- User-friendly interface for traffic police officers to manage violation-related tasks efficiently.
- Automated generation and issuance of traffic violation invoices to violators.
- Centralized database for storing and retrieving accurate records of violations and payments.
- Reporting functionality to generate insightful reports on violation trends, fine collections, and other relevant metrics.
- Integration with other systems for seamless data exchange and collaboration.

By leveraging technology to streamline traffic violation management processes, TVTS empowers law enforcement agencies to enforce traffic rules more effectively, reduce administrative burdens, and ultimately contribute to safer roads for all.

Contents

Chapter 1

Introduction

1.1 DATABASE TECHNOLOGIES

The essential feature of database technology is that it provides an internal representation (model) of the external world of interest. Examples are, the representation of a particular date/time/flight/aircraft in an airline reservation or of the item code/item description/quantity on hand/reorder level/reorder quantity in a stock control system. The technology involved is concerned primarily with maintaining the internal representation consistent with external reality; this involves the results of extensive RD over the past 30 years in areas such as user requirements analysis, data modelling, process modelling, data integrity, concurrency, transactions, file organisation, indexing, rollback and recovery, persistent programming, object-orientation, logic programming, deductive database systems, active database systems... and in all these (and other) areas there remains much more to be done. The essential point is that database technology is a CORE TECHNOLOGY which has links to:

- Information management / processing
- Data analysis / statistics
- Data visualization / presentation
- Multimedia and hypermedia
- Office and document systems
- Business processes, workflow, CSCW (computer-supported cooperative work)

Relational DBMS is the modern base technology for many business applications. It offers flexibility and easy-to-use tools at the expense of ultimate performance. More recently relational systems

have started extending their facilities in directions like information retrieval, object orientation and deductive/active systems which lead to the so-called 'Extended Relational Systems'.
Information Retrieval Systems began with handling library catalogues and then extended to full free-text by utilizing inverted index technology with a lexicon or thesaurus. Modern systems utilize some KBS (knowledge-based systems) techniques to improve the retrieval. Object-Oriented DBMS started for engineering applications in which objects are complex, have versions and need to be treated as a complete entity. OODBMSs share many of the OOPL features such as identity, inheritance, late binding, overloading and overriding. OODBMSs have found favours in engineering and office systems but haven't been successful yet in traditional application areas. Deductive / Active DBMS has evolved over the last 20 years and combines logic programming technology with database technology. This allows the database itself to react to the external events and also to maintain its integrity dynamically with respect to the real world.

1.2 CHARACTERISTICS OF DATABASE APPROACH

Traditional form included organising the data in file format. DBMS was a new concept then, and all kinds of research was done to make it overcome the deficiencies in traditional style of data management. A modern DBMS has the following characteristics

- Real-world entity A modern DBMS is more realistic and uses real-world entities to design its architecture. It uses behaviour and attribute too. For example, a school database may use students as an entity and their age as an attribute.
- Relation-based tables DBMS allows entities and relations to form tables. A user can understand the architecture of a database by just looking at the table names.
- Isolation of data and application A database system is entirely different than its data. A database is an active entity, whereas data is said to be passive, on which the database works and organizes. DBMS also stores metadata, which is data about data, to ease its own process.
- Less redundancy DBMS follows the rules of normalization, which splits a relation when any of its attributes has redundancy in its values. Normalization is a mathematically rich and scientific process that will reduces the data redundancy
- Consistency Consistency is a state where every relation in a database remains consistent. There exists methods and techniques, that can detect an attempt of leaving database in an inconsistent

state. DBMS can provide greater consistency as compared to earlier forms of data storing applications like file-processing systems.

- Query Language DBMS is equipped with query language, which makes it more efficient to retrieve and manipulate data. A user can apply as many and the filtering options as required to retrieve a set of data. Traditionally it was not possible where file-processing system was used
- ACID Properties DBMS follows the concepts of Atomicity, Consistency, Isolation, and Durability (normally shortened as ACID). These concepts are applied on transactions, which manipulate data in a database. ACID properties help the database to stay healthy in multi-transactional environments and also in case of failure.
- Security Features like multiple views offer security to certain extent when users are unable to access the data of other users and departments. DBMS offers methods to impose constraints while entering data into the database and retrieving the same at a later stage. DBMS offers many different levels of security features, which enables multiple users to have different views with different features. For example, a user in the Sales department cannot see the data that belongs to the Purchase department. It can also be helpful in deciding how much data of the Sales department should be displayed to the user. Since a DBMS is not saved on the disk as traditional file systems, it is very hard for miscreants to break the code.
- Multiuser and Concurrent Access DBMS supports multi-user environment and allows them to access and manipulate data in parallel. Though there are restrictions on transactions when users attempt to handle the same data item, but users are always unaware of them.

1.3 APPLICATIONS OF DBMS

Applications of Database Management Systems :

- Telecom: There is a database to keeps track of the information regarding the calls made, network usage, customer details etc. Without the database system it is hard to maintain such huge amounts of data which gets updated every millisecond.
- Industry: Whether it is a manufacturing unit, a warehouse or a distribution centre, each one needs a database to keep the records of the ins and outs. For example, a distribution centre should keep a track of the product units that were supplied to the centre as well as the products that got delivered from the distribution centre on each day; this is where DBMS comes into picture

- Banking System: For storing information regarding a customer, keeping a track of his/her day to day credit and debit transactions, generating bank statements etc is done with through Database management systems.
- Education sector: Database systems are frequently used in schools and colleges to store and retrieve the data regarding the student , staff details, course details, exam details, payroll data, attendance details, fees details etc. There is lots of inter-related data that needs to be stored and retrieved in an efficient manner.
- Online shopping: You must be aware of the online shopping websites such as Amazon, Flip kart etc. These sites store the product information, your addresses and preferences, credit details and provide you the relevant list of products based on your query. All this involves a Database management system.

1.4 PROBLEM DESCRIPTION/STATEMENT

The Traffic Violation Tracking System (TVTS) aims to simplify traffic violation management for law enforcement agencies. It tackles the challenge of issuing violation invoices and maintaining accurate records efficiently.

Key Points:

1.Simplify Traffic Violation Management: TVTS seeks to simplify the process of managing traffic violations, focusing on automation and reducing paperwork.

2.Efficient Record Maintenance: The system aims to maintain accurate records of violations and payments, ensuring data integrity and accessibility for law enforcement officers.

3.User-Friendly Interface: TVTS provides a user-friendly interface for traffic police officers, allowing them to easily report violations, manage records, and generate reports.

4.Goals:

- Automate traffic violation management processes.
- Reduce manual paperwork and improve efficiency.
- Maintain accurate records to aid law enforcement efforts.

Overall, TVTS addresses the need for a streamlined and efficient system to manage traffic violations, ultimately contributing to improved enforcement and road safety.

Chapter 2

Requirement Analysis

2.1 Hardware Requirements

The Hardware requirements are very minimal and the program can be run on most of the machines. Processor : i5 processor Processor Speed : 1.2 GHz RAM : 1 GB Storage Space : 40 GB Monitor Resolution : 1024*768 or 1336*768 or 1280*1024

2.2 Software Requirements

1. Operating System used: Windows 10 2. Technologies used: HTML, CSS, PHP, Bootstrap 3. XAMPP Server: MySQL, PhpMyAdmin 4. IDE used: Visual Studio Code 5. Browser that supports HTML

2.3 Functional Requirements

2.3.1 Major Entities

2.3.2 End User Requirements

2.3.3 HTML

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from a local storage and render them to multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects like interactive forms can be embedded into the rendered page. It provides a way to create structured documents by denoting structural semantics for the text like headings, paragraphs, lists, links, quotes and other items. HTML elements are delimited by tags that are written within angle brackets. Tags such as img tag and input tag introduce content into the page directly. Other tags such as ¡p¿...¡/p¿ surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page. HTML can also embed programs written in a scripting language such as JavaScript which affect the behaviour and content of web pages. Inclusion of CSS defines the look and layout of content.

2.3.4 CSS

Cascading Style Sheets (CSS) is a style sheet language which is used for describing the presentation of a document written in markup language. Although most often its used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is also applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications. CSS is designed primarily to enable the separation of presentation and content, including aspects such as the layout, colours, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share the formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

2.3.5 PHP

PHP is a server-side scripting language designed primarily for web development but is also used as a general-purpose programming language. Originally created by Rasmus Lerdorf in 1994, the PHP reference implementation is now produced by The PHP Development Team. PHP originally stood for Personal Home Page, but it now stands for the recursive acronym PHP: Hypertext Pre-processor. PHP code can be embedded into HTML or HTML5 markup, or it can be used in combination with various web template systems, web content management systems and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server software combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page.PHP code can also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications. The standard PHP interpreter, powered by the Zend Engine, is a free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers, on almost every operating system and platform, free of charge. The PHP language evolved without a written formal specification or standard until 2014, leaving the canonical PHP interpreter as a de facto standard. Since 2014 work has gone into creating a formal PHP specification. HP development began in 1995 when Rasmus Lerdorf wrote several Common Gateway Interface (CGI) programs in C, which he used in order to maintain Restaurant Management System his personal homepage. He extended them to work with web forms and to communicate with databases, and called this implementation ”Personal Home Page/Forms Interpreter” or PHP/FI PHP/FI could help to build simple, dynamic web applications. To accelerate bug reporting and to improve the code, Lerdorf initially announced the release of PHP/FI as ”Personal Home Page Tools (PHP Tools) version 1.0” on the Usenet discussion group on June 8, 1995 This release already had the basic functionality that PHP has as of 2013. This included Perl-like variables, form handling, and the ability to embed HTML. The syntax resembled that of Perl but was simpler, more limited and less consistent.

2.3.6 MySQL

MySQL is a Relational Database Management System (RDBMS). MySQL server can manage many databases at the same time. In fact, many people might have different databases managed by a single MySQL server. Each database consists of a structure to hold onto the data itself. A data-base can exist without data, only a structure, be totally empty, twiddling its thumbs and waiting for data to be stored in it. Data in a database is stored in one or more tables. You must create the data-base and the tables before you can add any data to the database. First you create the empty database. Then you add empty tables to the database. Database tables are organized in rows and columns. Each row represents an entity in the database, such as a customer, a book, or a project. Each column contains an item of information about the entity, such as a customer name, a book name, or a project start date. The place where a particular row and column intersect, the individual cell of the table, is called a field. Tables in databases can be related. Often a row in one table is related to several rows in another table. For instance, you might have a database containing data about books you own. You would have a book table and an author table. One row in the author table might contain information about the author of several books in the book table. When tables are related, you include a column in one table to hold data that matches data in the column of another table. MySQL, the most popular Open Source

SQL database management system, is developed, distributed, and supported by MySQL AB. MySQL AB is a commercial company, founded by the MySQL developers. It is a second generation Open Source company that unites Open Restaurant Management System Source values and methodology with a successful business model.

- MySQL is a database management system. A database is a structured collection of data. It can be anything from a simple shopping list to a picture gallery or the vast amount of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

- MySQL is a relational database management system. A relational database stores data in separate tables rather than putting all the data in one big storeroom. This adds speed and flexibility. The SQL part of "MySQL" stands for "Structured Query Language." SQL is the most common standardized language used to access databases and is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. "SQL-92" refers to the standard released in 1992, "SQL:1999" refers tothe standard released in 1999, and "SQL:2003" refers to the current version of the standard. We use the phrase "the SQLstandard" to refer to the current version of the SQL Standard.

- MySQL software is Open Source. Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), to define what you may and may not do with the software in different situations. The MySQL Database Server is very fast, reliable, and easy to use.

MySQL Server was originally developed to handle large databases and has been successfully used in highly demanding production environments for several years. MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.

2.3.7 XAMPP Server

Xampp server installs a complete, ready-to-use development environment. Xampp server allows you to fit your needs and allows you to setup a local server with the same characteristics as your production. While setting up the server and PHP on your own, you have two choices for the method of connecting PHP to the server. For many servers, PHP has a direct module interface (also called SAPI). These servers include Apache, Microsoft Internet Information Server, Netscape and iPlanet servers. Many other servers support ISAPI, the Microsoft module interface (OmniHTTPd for example). If PHP has no module support for your web server, you can always use it as a CGI or FastCGI processor. This means you set up yourserver to use the CGI executable of PHP to process all PHP file requests on the server.

- OpenGL is case sensitive
- Line Color, Filled Faces and Fill Color not supported.
- Shadow plane is not supported.

Chapter 3

Database Design

3.1 Attributes

The database contains the following tables:

1. **police_info:**
   - Police_id
   - User_name
   - Police_name
   - Phone_no
   - Email
   - Password
2. **vehicle_details:**
   - Vechicle_number
   - Vehicle_owner
   - Vehicle_name
   - Vehicle_type
   - Vehicle_model
3. **Violations_cost:**
   - Violation_type
   - Amount

4. **challan:**

   - Challan_id
   - Challan_date
   - Challan_cost
   - Violation_type
   - Police_id
   - Vehicle_number

3.2 ER Diagram

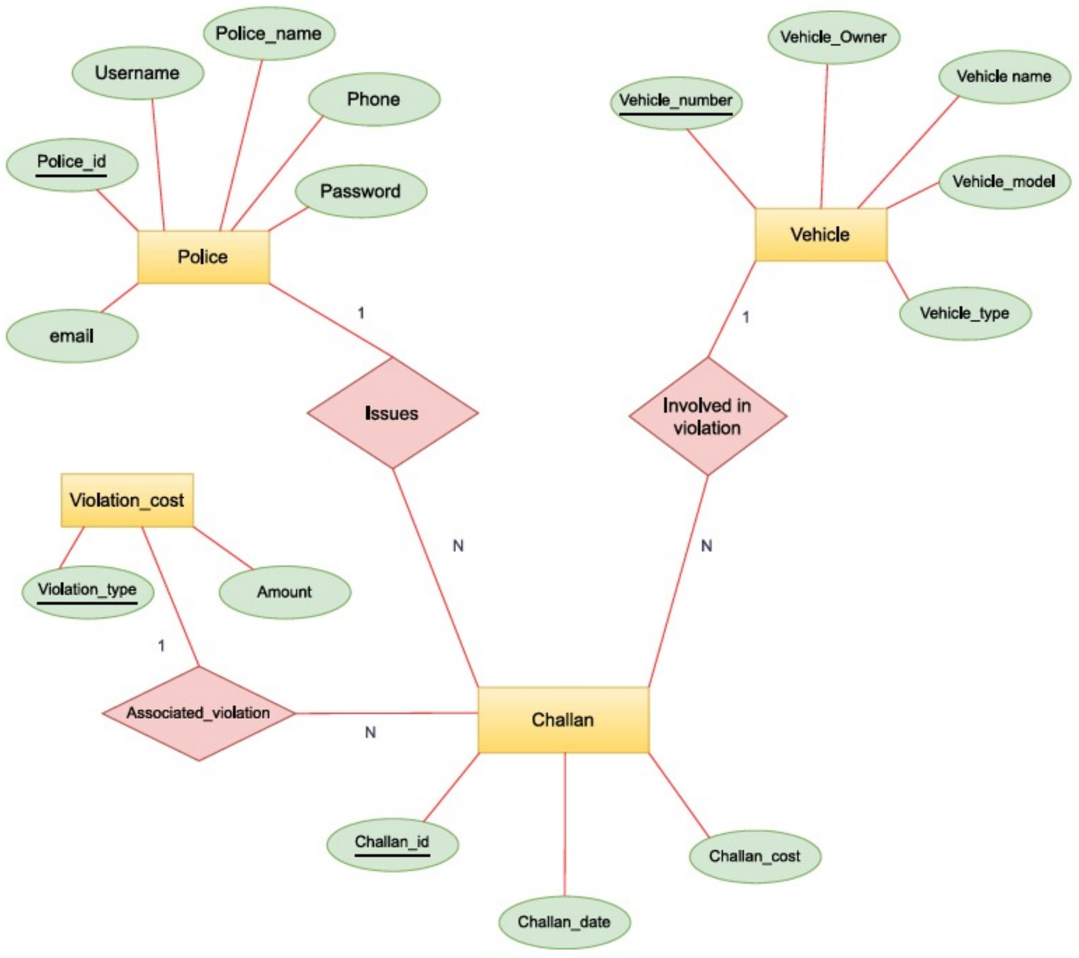


Fig.3.1. ER Diagram for Placement Management System

The Entity Relationship Diagram explains the relationship among the entities present in the database. ER models are used to model real-world objects and the relation between these real-world objects.ER Diagram is the structural format of the database.
ER Model is used to model the logical view of the system from a data perspective which consists of these symbols:

- **Rectangles**: Rectangles represent Entities in the ER Model.
- **Ellipses:** Ellipses represent Attributes in the ER Model.
- **Diamond:** Diamonds represent Relationships among Entities.
- **Lines:** Lines represent attributes to entities and entity sets with other relationship types.
- **Double Ellipse:** Double Ellipses represent Multi-Valued Attributes.
- **Double Rectangle:** Double Rectangle represents a Weak Entity.

Info about ER Diagram:

It consists of four Attributes mainly:-

- police_info
- vehicle_detail
- violations_cost
- challan

3.3 Relational Schema

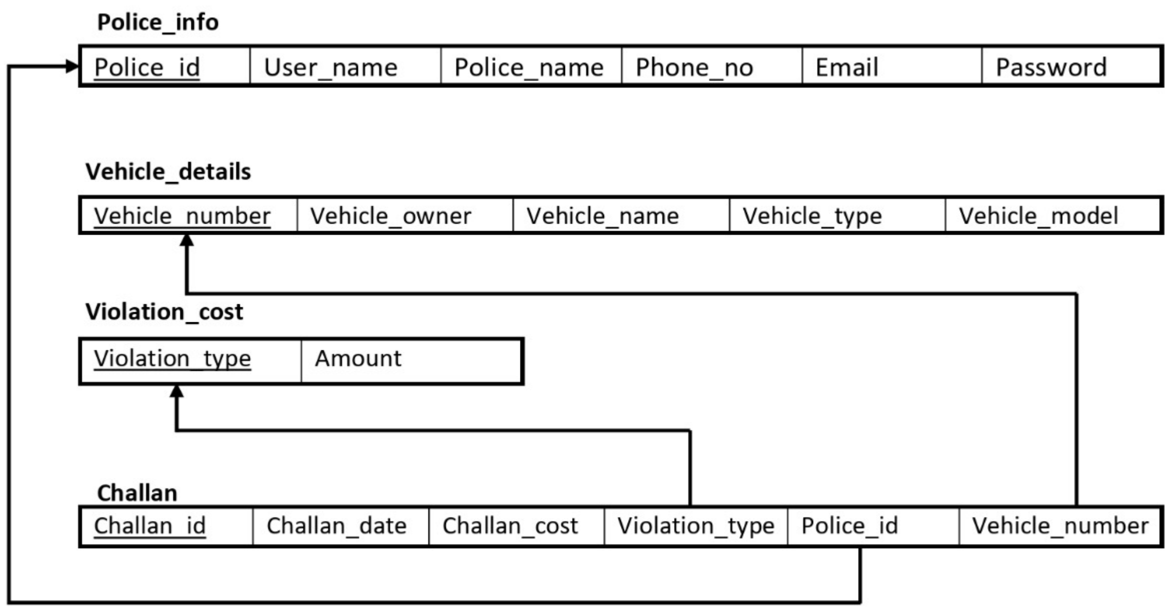


Fig.3.2. Schema Diagram for Placement Management System

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data.A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams. It's the database designers who design the schema to help programmers understand the database and make it useful.

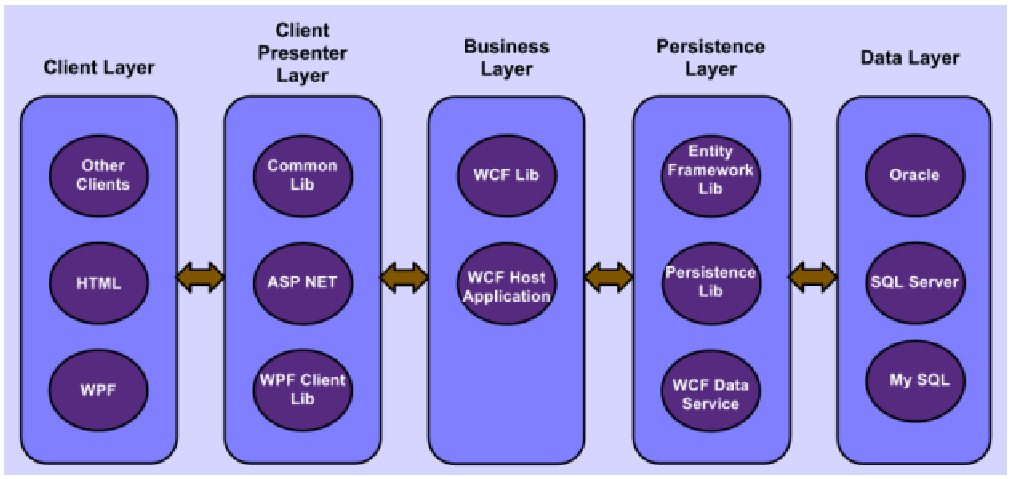
Chapter 4

Implementation

4.1 Creating Database Connection

- You can connect your deployments by either: Providing your connection string. Specifying Advanced Connection Options. Advanced connection options allow you to specify authentication, TLS/SSL, and SSH connection options.
- Here we make use of MongoDB Compass which is a GUI for MongoDB to establish connection to our server.
- Connect to the MongoDB Server 3.1 db = connect("localhost:27017/myDatabase")
- Access the database 4.1.db.myNewCollection1.insertOne(x: 1)

4.2 Architecture used(4-tier architecture)



Four Tier architecture is a client–server architecture in which presentation, application processing, and data management functions are physically separated. Four-tier application architecture provides a model by which developers can create flexible and reusable applications. By segregating an application into tiers, developers acquire the option of modifying or adding a specific layer, instead of reworking the entire application

Presentation layer

This is the topmost level of the application. The presentation tier displays information related to services such as browsing merchandise, purchasing and shopping cart contents. It also communicates with other tiers and puts out the results to the browser/client tier and to all other tiers in the network. In simple terms, it is a layer which users can access directly (such as a web page, or an operating system's GUI).

Business Layer

Business layer or domain logic is the part of the program that encodes the real-world business rules which determine how data can be created, stored, and changed. It is contrasted with the remainder of the software that might be concerned with lower-level details of managing a database or displaying the user interface, system infrastructure, or generally connecting various parts of the program.

Data access layer

A Data Access Layer (DAL) in computer software, is a layer of computer program which provides simplified access to data stored in persistent storage. For example, the DAL might return a reference to an object (in terms of object-oriented programming) with its attributes instead of a row of fields from a database table. This allows the client (or user) modules to be created with a higher level of abstraction. This kind of model could be implemented by creating a class of data access methods that directly reference a corresponding set of database stored procedures. Another implementation could potentially retrieve or write records to or from a file system. The DAL hides the complexity of the underlying data store from the external world.

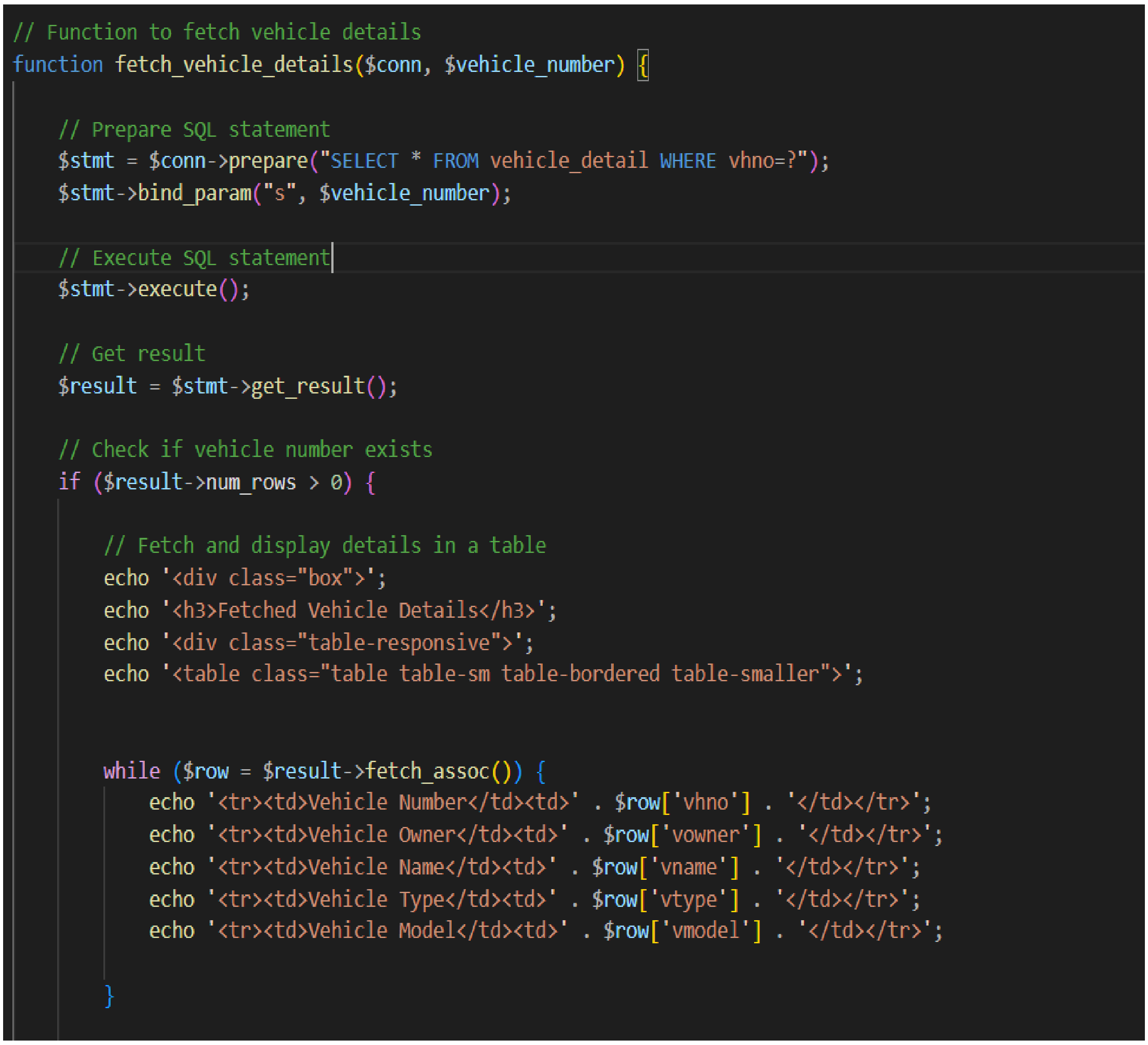
Control layer

The control layer is responsible for the communication between business and presentation layer. It connects logic and data with each other and provides a better connectivity and separation between layers

4.2.1 Pseudo Code For Major Functionalities

Reporting offense page: Allows to report offense on vehicle

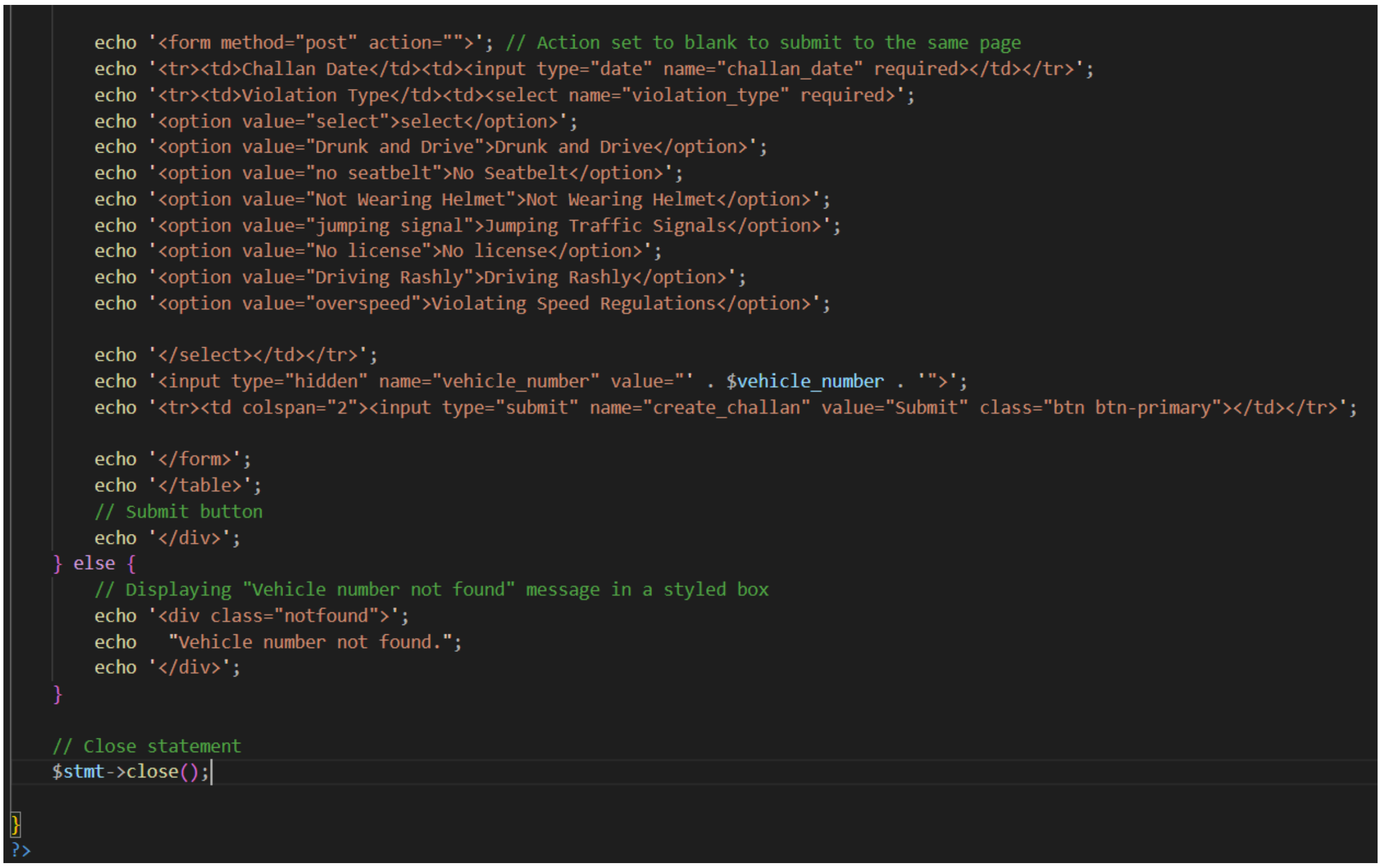
1.Fetch Vehicle details of violator:



```
// Function to fetch vehicle details
function fetch_vehicle_details($conn, $vehicle_number) {

    // Prepare SQL statement
    $stmt = $conn->prepare("SELECT * FROM vehicle_detail WHERE vhno=?");
    $stmt->bind_param("s", $vehicle_number);

    // Execute SQL statement
    $stmt->execute();

    // Get result
    $result = $stmt->get_result();

    // Check if vehicle number exists
    if ($result->num_rows > 0) {

        // Fetch and display details in a table
        echo '<div class="box">';
        echo '<h3>Fetched Vehicle Details</h3>';
        echo '<div class="table-responsive">';
        echo '<table class="table table-sm table-bordered table-smaller">';


        while ($row = $result->fetch_assoc()) {
            echo '<tr><td>Vehicle Number</td><td>' . $row['vhno'] . '</td></tr>';
            echo '<tr><td>Vehicle Owner</td><td>' . $row['vowner'] . '</td></tr>';
            echo '<tr><td>Vehicle Name</td><td>' . $row['vname'] . '</td></tr>';
            echo '<tr><td>Vehicle Type</td><td>' . $row['vtype'] . '</td></tr>';
            echo '<tr><td>Vehicle Model</td><td>' . $row['vmodel'] . '</td></tr>';

        }
```

Fig.3.2. Vehicle details of violator

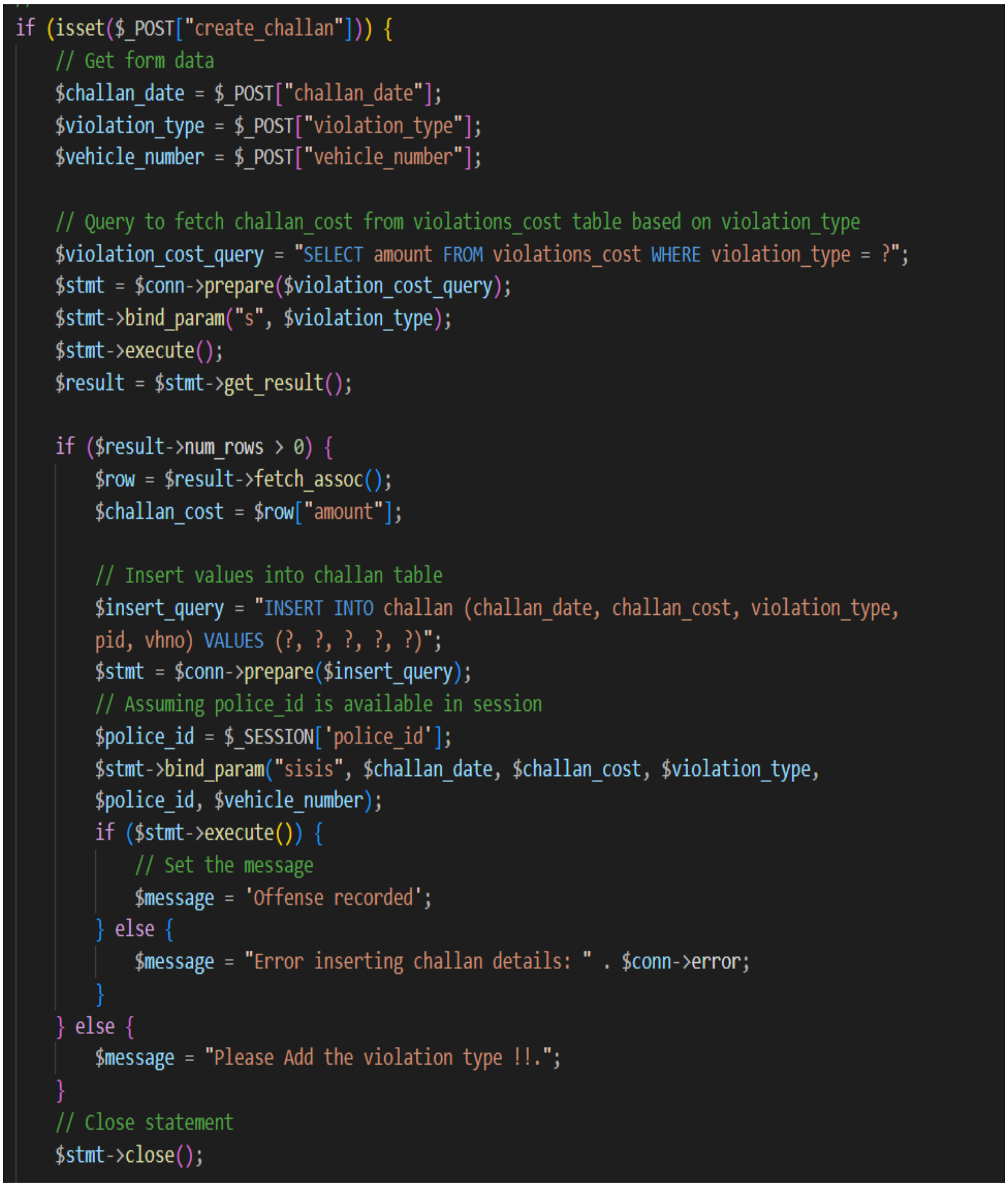
2.Reporting violation_type and date:



```
        echo '<form method="post" action="">'; // Action set to blank to submit to the same page
        echo '<tr><td>Challan Date</td><td><input type="date" name="challan_date" required></td></tr>';
        echo '<tr><td>Violation Type</td><td><select name="violation_type" required>';
        echo '<option value="select">select</option>';
        echo '<option value="Drunk and Drive">Drunk and Drive</option>';
        echo '<option value="no seatbelt">No Seatbelt</option>';
        echo '<option value="Not Wearing Helmet">Not Wearing Helmet</option>';
        echo '<option value="jumping signal">Jumping Traffic Signals</option>';
        echo '<option value="No license">No license</option>';
        echo '<option value="Driving Rashly">Driving Rashly</option>';
        echo '<option value="overspeed">Violating Speed Regulations</option>';

        echo '</select></td></tr>';
        echo '<input type="hidden" name="vehicle_number" value="' . $vehicle_number . '">';
        echo '<tr><td colspan="2"><input type="submit" name="create_challan" value="Submit" class="btn btn-primary"></td></tr>';

        echo '</form>';
        echo '</table>';
        // Submit button
        echo '</div>';
    } else {
        // Displaying "Vehicle number not found" message in a styled box
        echo '<div class="notfound">';
        echo   "Vehicle number not found.";
        echo '</div>';
    }

    // Close statement
    $stmt->close();

}
?>
```

Fig.3.3. Report violation_type and date

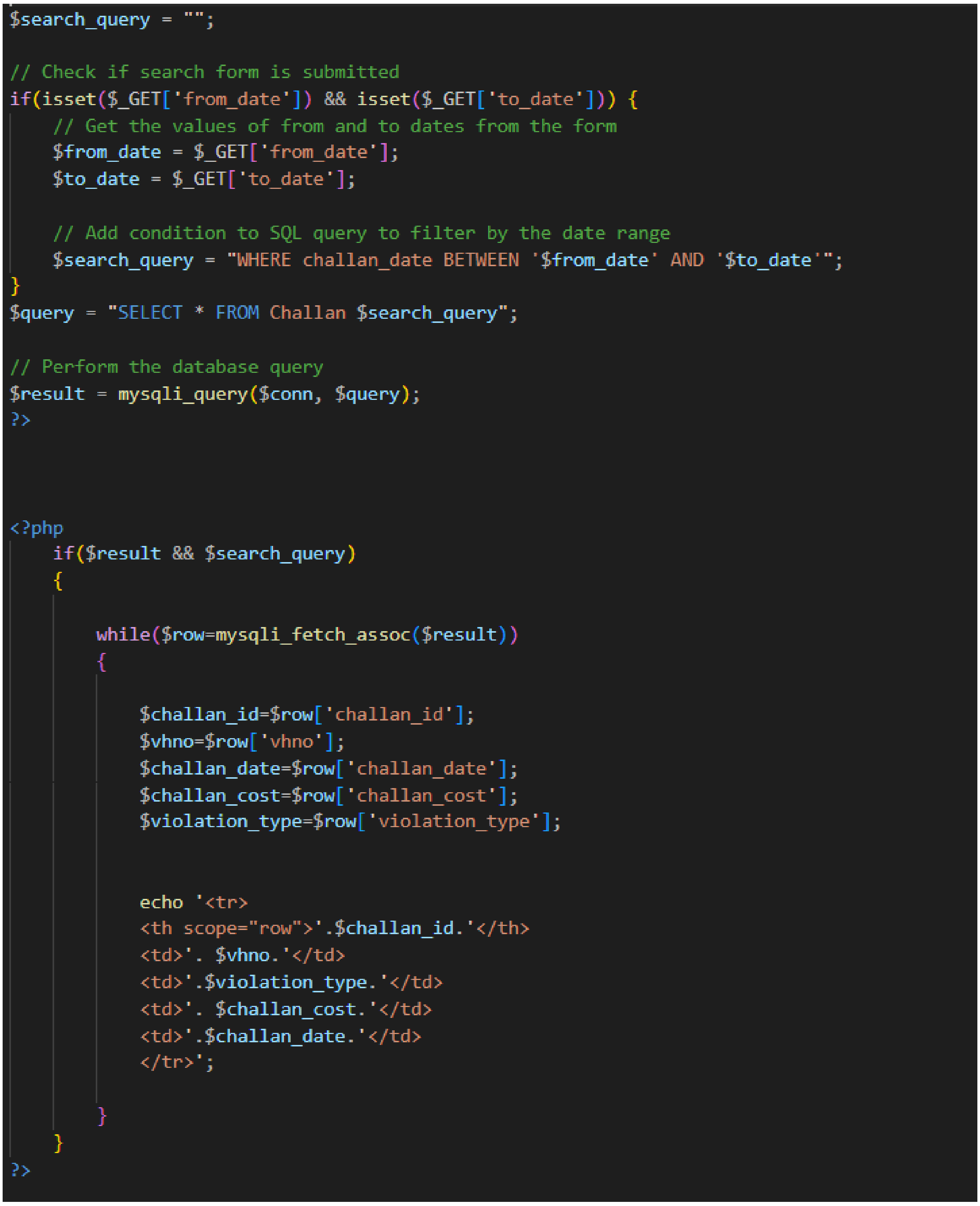
3.Submission of Challan:



```
if (isset($_POST["create_challan"])) {
    // Get form data
    $challan_date = $_POST["challan_date"];
    $violation_type = $_POST["violation_type"];
    $vehicle_number = $_POST["vehicle_number"];

    // Query to fetch challan_cost from violations_cost table based on violation_type
    $violation_cost_query = "SELECT amount FROM violations_cost WHERE violation_type = ?";
    $stmt = $conn->prepare($violation_cost_query);
    $stmt->bind_param("s", $violation_type);
    $stmt->execute();
    $result = $stmt->get_result();

    if ($result->num_rows > 0) {
        $row = $result->fetch_assoc();
        $challan_cost = $row["amount"];

        // Insert values into challan table
        $insert_query = "INSERT INTO challan (challan_date, challan_cost, violation_type,
        pid, vhno) VALUES (?, ?, ?, ?, ?)";
        $stmt = $conn->prepare($insert_query);
        // Assuming police_id is available in session
        $police_id = $_SESSION['police_id'];
        $stmt->bind_param("sisis", $challan_date, $challan_cost, $violation_type,
        $police_id, $vehicle_number);
        if ($stmt->execute()) {
            // Set the message
            $message = 'Offense recorded';
        } else {
            $message = "Error inserting challan details: " . $conn->error;
        }
    } else {
        $message = "Please Add the violation type !!.";
    }
    // Close statement
    $stmt->close();
```

Fig.3.3. Submission of Challan

Search By Challan_Date page: Allows to search challans by specific Date



```
$search_query = "";

// Check if search form is submitted
if(isset($_GET['from_date']) && isset($_GET['to_date'])) {
    // Get the values of from and to dates from the form
    $from_date = $_GET['from_date'];
    $to_date = $_GET['to_date'];

    // Add condition to SQL query to filter by the date range
    $search_query = "WHERE challan_date BETWEEN '$from_date' AND '$to_date'";
}
$query = "SELECT * FROM Challan $search_query";

// Perform the database query
$result = mysqli_query($conn, $query);
?>



<?php
    if($result && $search_query)
    {

        while($row=mysqli_fetch_assoc($result))
        {

            $challan_id=$row['challan_id'];
            $vhno=$row['vhno'];
            $challan_date=$row['challan_date'];
            $challan_cost=$row['challan_cost'];
            $violation_type=$row['violation_type'];


            echo '<tr>
            <th scope="row">'.$challan_id.'</th>
            <td>'. $vhno.'</td>
            <td>'.$violation_type.'</td>
            <td>'. $challan_cost.'</td>
            <td>'.$challan_date.'</td>
            </tr>';

        }
    }
?>
```

Fig.3.4. Search By Challan_Date

Chapter 5

Results & Snapshots

Home page

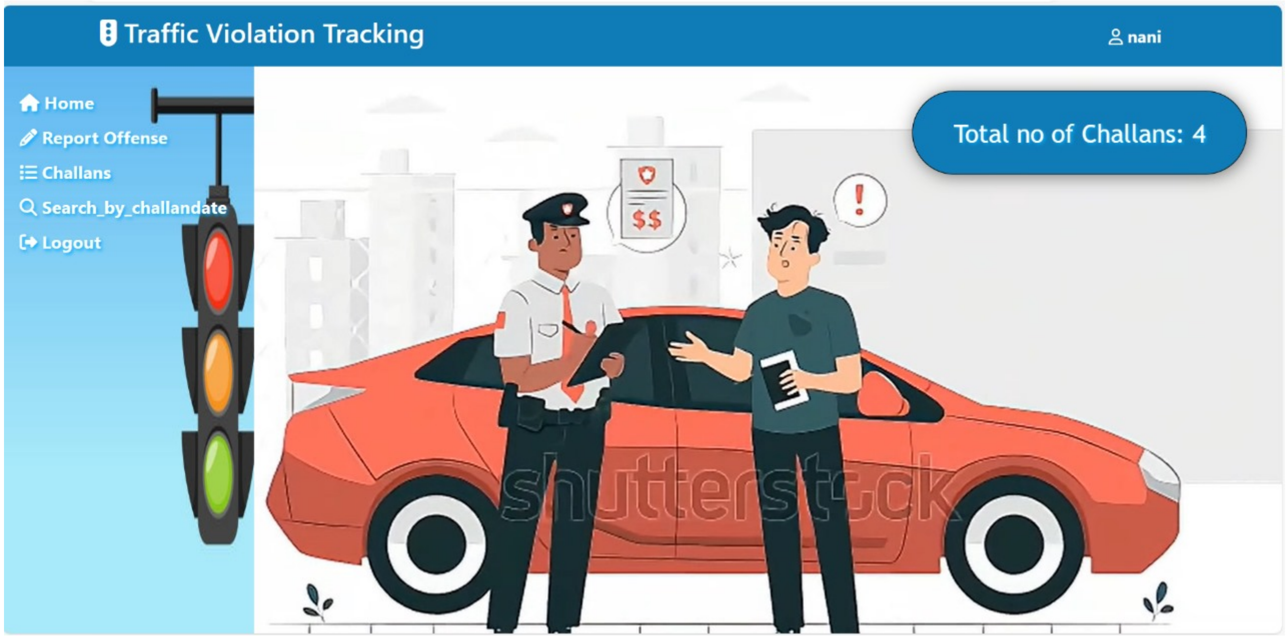
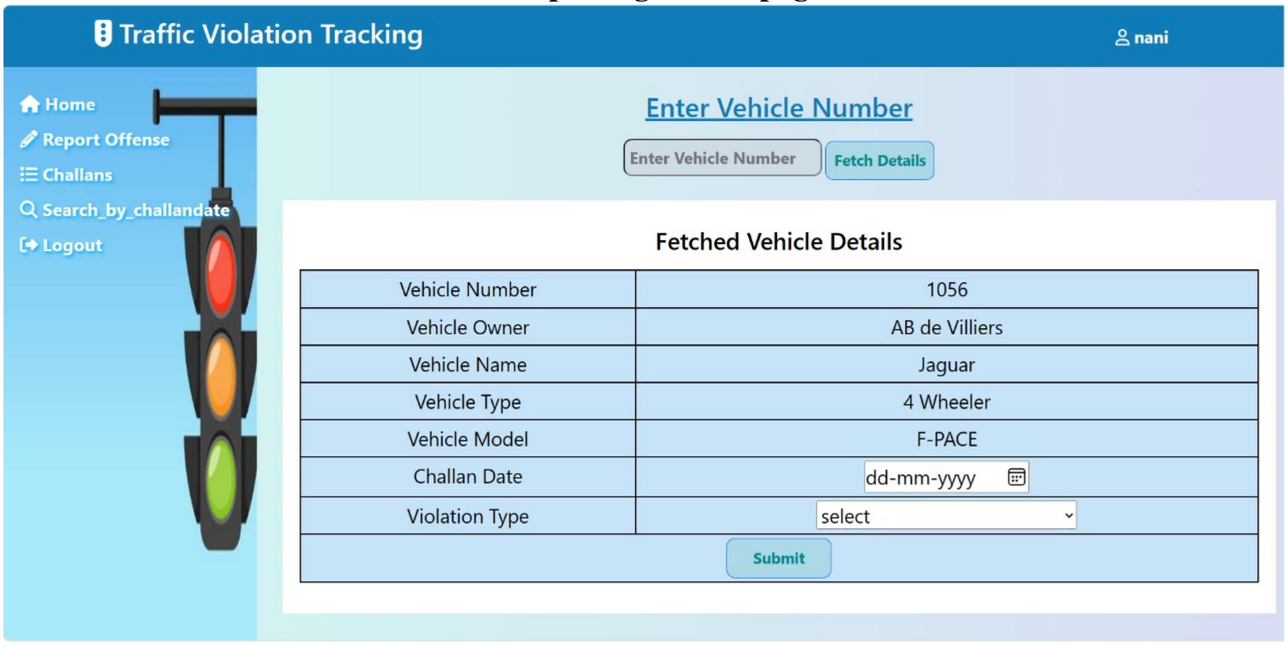


Fig.5.1. Home page

The PHP script establishes a connection to a database and queries the number of challans stored. It retrieves this count and displays it on a webpage. The HTML and CSS sections create a visually appealing interface with a video background. The challan count is showcased in a box with a stylish design. The backend PHP code handles database interaction and data retrieval. The frontend HTML/CSS part ensures an engaging user experience. Users can view the total number of challans dynamically updated on the webpage. This integration of backend and frontend technologies delivers a seamless user interaction. The script utilizes PHP's database capabilities to fetch real-time data. The frontend design enhances user engagement with visual elements. Overall, the script combines backend functionality with frontend aesthetics to provide a comprehensive web experience.

Reporting offense page



Fetched Vehicle Details

Vehicle Number	1056
Vehicle Owner	AB de Villiers
Vehicle Name	Jaguar
Vehicle Type	4 Wheeler
Vehicle Model	F-PACE
Challan Date	dd-mm-yyyy
Violation Type	select

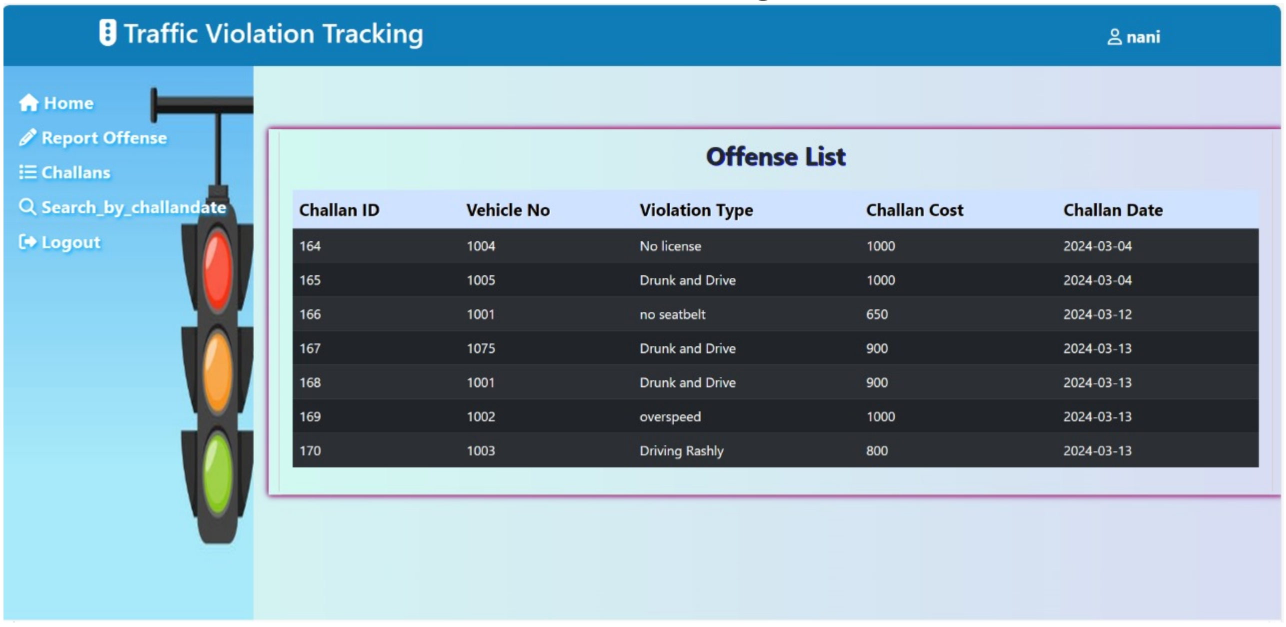
Fig.5.2. Reporting Offense page

The interface begins with a form prompting users to enter a vehicle number. Upon submission, the script fetches and displays vehicle details if the vehicle number exists in the database. It then presents a dropdown selection for violation types and a form to create a challan for the selected vehicle and violation type.

The backend logic handles form submission, executing SQL queries to fetch violation costs and insert challan details into the database. It also manages error handling, ensuring that appropriate messages are displayed to the user in case of success or failure.

Styling is implemented using CSS, with Bootstrap classes utilized for consistent styling of components such as buttons and tables. The script effectively differentiates between fetched vehicle details and error messages using styled boxes with distinct colors, enhancing user experience. Overall, this web application offers a robust solution for recording traffic violations and efficiently generating challans.

Offense List Page



Offense List

Challan ID	Vehicle No	Violation Type	Challan Cost	Challan Date
164	1004	No license	1000	2024-03-04
165	1005	Drunk and Drive	1000	2024-03-04
166	1001	no seatbelt	650	2024-03-12
167	1075	Drunk and Drive	900	2024-03-13
168	1001	Drunk and Drive	900	2024-03-13
169	1002	overspeed	1000	2024-03-13
170	1003	Driving Rashly	800	2024-03-13

Fig.5.3. Offense List Page

The PHP script generates a web page to display a list of traffic offense records (challans) stored in a database. It starts by initiating a session and including the necessary configuration file (‘config.php‘). Then, it executes an SQL query to retrieve all records from the ‘Challan‘ table.

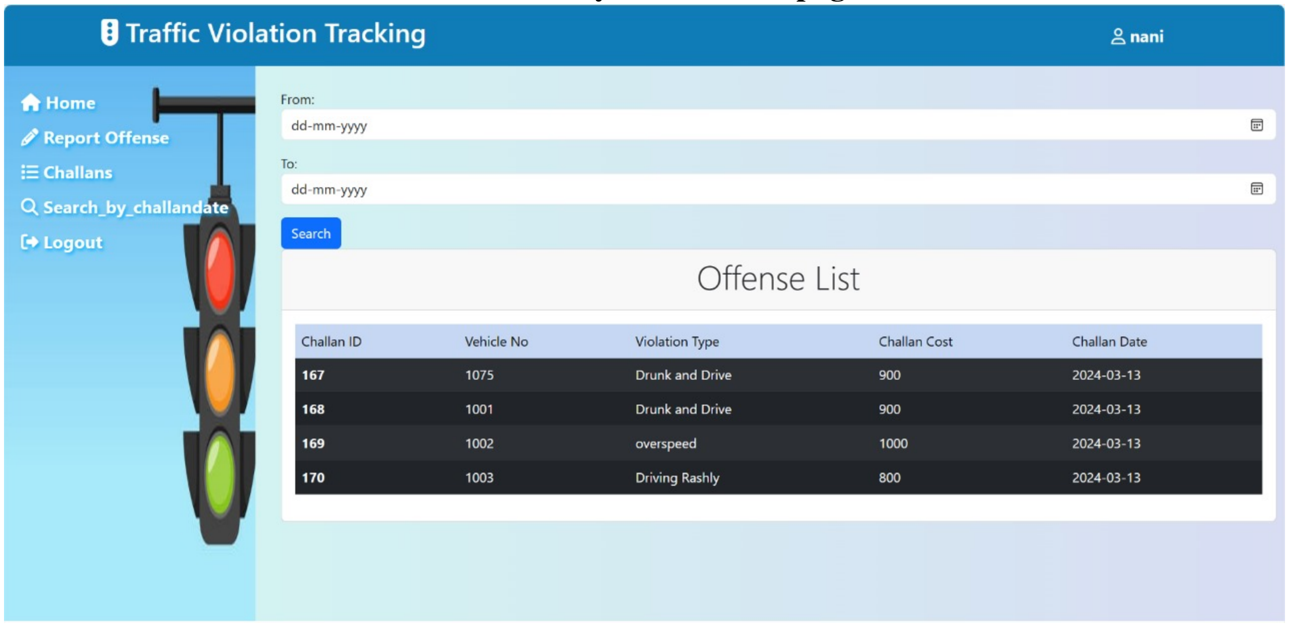
The retrieved data is then displayed in an HTML table format using Bootstrap for styling. Each row in the table represents a single offense record, displaying details such as Challan ID, Vehicle Number, Violation Type, Challan Cost, and Challan Date.

The table is styled using Bootstrap classes for better presentation, with alternating row colors for improved readability. The header row is highlighted with a different background color to distinguish it from the data rows.

The script dynamically populates the table rows using a while loop that iterates over the result set obtained from the database query. Inside the loop, it extracts the relevant information from each row and inserts it into the table cells.

Overall, this script provides a user-friendly interface for viewing and managing traffic offense records, enhancing accessibility and usability for users interacting with the system.

Search by challan_date page



Challan ID	Vehicle No	Violation Type	Challan Cost	Challan Date
167	1075	Drunk and Drive	900	2024-03-13
168	1001	Drunk and Drive	900	2024-03-13
169	1002	overspeed	1000	2024-03-13
170	1003	Driving Rashly	800	2024-03-13

Fig.5.3. search by challan_date page

The PHP script generates a web page to display a list of traffic offense records (challans) stored in a database. It allows users to search for offenses within a specific date range.

The script begins by including the necessary configuration file (‘config.php‘) and initializing search variables. It checks if the search form has been submitted using the ‘GET‘ method and retrieves the values of the ‘from_date‘ and ‘to_date‘ inputs.

Based on the submitted form data, it constructs a SQL query to filter the offense records based on the date range provided. The constructed query is then executed to fetch the relevant records from the database.

The HTML structure of the page includes a search form with input fields for ‘from_date‘ and ‘to_date‘, allowing users to specify the date range for the search. Upon submitting the form, the page reloads with the filtered offense records displayed in a table format.

The table presents details of each offense record, including Challan ID, Vehicle Number, Violation Type, Challan Cost, and Challan Date. Each row in the table corresponds to a single offense record retrieved from the database.

The table is styled using Bootstrap classes for a clean and organized appearance, with alternating row colors for improved readability. Additionally, the header row is highlighted with a different background color to distinguish it from the data rows.

Overall, this script provides users with a convenient interface to search and view traffic offense records within a specified date range, enhancing usability and accessibility for managing traffic-related data.

Chapter 6

Conclusion

In conclusion, the Traffic Violation Tracking System (TVTS) project represents a comprehensive and efficient solution for managing traffic offense records and ensuring road safety. Throughout the development and implementation process, several key features and functionalities have been integrated to address the complex challenges associated with traffic management.

The TVTS system offers users the ability to record, track, and analyze traffic violations seamlessly. By leveraging technologies such as PHP, MySQL, and Bootstrap, the system provides a user-friendly interface for both administrators and law enforcement personnel to manage offense data effectively.

Key highlights of the TVTS project include:

- **User-friendly Interface:** The system offers an intuitive and easy-to-use interface, allowing users to navigate through different functionalities effortlessly.
- **Search and Filtering:** Users can search and filter offense records based on various criteria such as date range, vehicle number, and violation type, enhancing accessibility and data retrieval efficiency.
- **Record Management:** TVTS enables the efficient recording and management of offense records, including details such as Challan ID, Vehicle Number, Violation Type, Challan Cost, and Challan Date.
- **Data Analysis:** The system facilitates data analysis and reporting, empowering authorities to identify patterns, trends, and areas of concern, leading to informed decision-making and targeted interventions.

Chapter 7

Future Enhancements

In order to further enhance the Traffic Violation Tracking System (TVTS), there are several avenues for future development focused on engaging users and promoting road safety. One approach is the development of a mobile application, offering users convenient access to violation history, educational resources, and safety tips on their smartphones. Interactive training modules can be incorporated to simulate driving scenarios, providing real-time feedback and personalized recommendations for safer driving practices. Additionally, community engagement initiatives such as workshops and awareness campaigns can empower users to actively participate in promoting road safety within their communities. Integration with ride-sharing platforms and incentive programs for safe driving can further incentivize responsible behavior, while collaboration with law enforcement agencies and feedback mechanisms can ensure continuous improvement based on user input. Through these future enhancements, the TVTS platform aims to create a more user-centric experience, fostering a culture of responsible driving and contributing to overall road safety efforts.

References

[1] Ramez Elmarsi and Shamkant B. Navathe,*”Fundamentals of Database Systems”* , Pearson, 7th edition,2017.

[2] Abraham Silberschatz,Henry F. Korth, S. Sudarshan *Database System Concepts*,MC Graw Hill,7th Edition, 2021

[3] Mark L. Gillenson,*Fundamentals of Database Management Systems*, 3rd Edition,Wiley,2023.

[4] @online MySQL official, `https://www.mysql.com/`