***NHA DATABASE***

**Database Design Document**

**V 3.0**

**By**

|  |  |  |
| --- | --- | --- |
| 1 | *Niyaz Ali* | *NUM-BSCS-2022-24* |
| 2 | *Eshna Eman* | *NUM-BSCS-2022-51* |
| 3 | *Malik Muhammad Hassan* | *NUM-BSCS-2022-47* |



**Department of Computer Sciences**

**Namal University**

**Mianwali, Pakistan**

**Submission Date: 30th June, 2024**

**REVISION HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Approved by** |
| *30/6/24* | *V.3.0* |  |  |
| *10/6/24* | *V.2.0* |  |  |
| *22/04/24* | *V 1.0* | *Specify the changes implemented subsequent to the submission of the previous document. These changes should be based on the suggestions given by the person who approved the document.* |  |

*Instructions:*

* *Place the latest revisions at the top of the table.*
* *The Revision History pertains only to changes in the document's content or any updates made after a suggestion from the approving authority. It does not apply to the template's formatting.*

**TABLE OF CONTENTS**

[CHAPTER 1: PROJECT OVERVIEW 4](#_Toc170133991)

[1.1. INTRODUCTION: 4](#_Toc170133992)

[1.2. PROBLEM STATEMENT: 4](#_Toc170133993)

[1.3. PROJECT OBJECTIVES: 4](#_Toc170133994)

[1.4. DOCUMENT OBJECTIVES: 4](#_Toc170133995)

[CHAPTER 2: DETAILED DATABASE DESIGN 5](#_Toc170133996)

[2.1. ENTITY: 5](#_Toc170133998)

[2.2. DATA DICTIONARY: 5](#_Toc170133999)

[2.3. RELATIONSHIPS: 5](#_Toc170134000)

[2.4. ENTITY RELATIONSHIP DIAGRAM: 6](#_Toc170134001)

[CHAPTER 3 : Logical DATABASE DESIGN 7](#_Toc170134002)

[3.1. RELATIONAL SCHEMA: 7](#_Toc170134003)

[3.2. FUNCTIONAL DEPENDENCIES: 7](#_Toc170134004)

[3.3. NORMALIZATION: 7](#_Toc170134005)

[CHAPTER 4 : Physical DATABASE DESIGN 8](#_Toc170134006)

[4.1. STRUCTURE OF THE TABLES: 8](#_Toc170134007)

[4.2. DATA SAMPLES INSIDE TABLES: 8](#_Toc170134008)

[4.3. QUERIES RESULTS: 8](#_Toc170134009)

[CHAPTER 5 : Interface Design 9](#_Toc170134010)

[5.1. LANGUAGE/FRAMEWORK: 9](#_Toc170134011)

[5.2. DATABASE CONNECTIVITY: 9](#_Toc170134012)

[5.3. STORED PROCEDURES AND FUNCTIONS: 9](#_Toc170134013)

[5.4. INTERFACES: 9](#_Toc170134014)

[CHAPTER 6 : CONCLUSION 10](#_Toc170134015)

[6.1. LESSONS LEARNED: 10](#_Toc170134016)

[6.2. CHALLENGES AND SOLUTIONS: 10](#_Toc170134017)

[6.3. FUTURE WORK AND IMPROVEMENTS: 10](#_Toc170134018)

[6.4. FINAL THOUGHTS: 10](#_Toc170134019)

[REFERENCES 11](#_Toc170134020)

# PROJECT OVERVIEW

## INTRODUCTION:

Our semester project focuses on developing a Database Management System (DBMS) for the National Highway Authority (NHA). The NHA, established in 2001, is responsible for the planning, development, operation, repair, and maintenance of National Highways and Strategic Roads in Pakistan. The project aims to streamline and enhance the management of information related to NHA's extensive network of national highways, motorways, expressways, and strategic routes. [1]

## PROBLEM STATEMENT:

National Interstate Specialist (NHA) of Pakistan supervises a endless organize of interstates and streets over the nation. In any case, overseeing the ventures, tenders, and contracts related with this framework may be a complex and challenging errand. The current framework needs a centralized database administration framework to effectively track and manage this information. Furthermore, there's a have to be screen and boycott development companies that come up short to meet measures or abuse controls. [1]

## PROJECT OBJECTIVES:

## Centralize NHA project, tender, contractor, and blacklist information.

## Ensure efficient data management with integrity and consistency.

## Provide easy access for informed decision-making.

## Generate detailed reports on projects and contractor performance.

## Automate administrative processes to reduce errors.

## Implement strong security for sensitive data.

[2],[3]

## DOCUMENT OBJECTIVES:

The document objectives are to provide a comprehensive database design for the National Highway Authority (NHA), facilitating efficient management of its extensive network of national highways, motorways, expressways, and strategic routes. It aims to outline the project's scope, objectives, and problem statement, ensuring alignment with NHA's goals. Additionally, the document aims to detail the database design, including entities, attributes, relationships, and an entity-relationship diagram (ERD), to enable informed decision-making and streamline project tracking and monitoring.

# DETAILED DATABASE DESIGN



## ENTITY:

|  |  |  |
| --- | --- | --- |
| 1. **Sr. No** | **Entity Name** | **Description** |
| 01 | Tender | A tender is a contract that a company get after wining the Bid of a project |
| 02 | Project | The Project represents a specific construction work undertaken by the National Highway Authority (NHA) within its jurisdiction. |
| 03 | Toll Taxes | Toll Taxes shows the collection of tolls fees or taxes levied on vehicle using National Highways and Motorways. |
| 04 | Blacklist | Blacklist entity represents a list of construction firms or companies that have been officially prohibited or restricted from engaging in projects overseen by the National Highway Authority (NHA) due to violations. |
| 05 | Remarks | The "Remarks" entity serves as a repository for comments or notes pertaining to the relationship between construction projects and blacklisted firms within the National Highway Authority (NHA) database system. |
| 06 | Power and Duties | The "Power and Duties" entity encapsulates the authoritative responsibilities and roles assigned to the National Highway Authority (NHA) within the context of managing national highways, motorways. |

## [1]

*.*

## DATA DICTIONARY:

* + 1. **Power and Duties:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No** | **Name** | **Data Type** | **Constraint** | **Description** |
| 01 | section | Varchar(255) | NOT NULL | This the section number or the rule of nha according to which the authorities are given |
| 02 | Section\_Allocation | Varchar(255) | NOT NULL/ NULL | Section Code for which power duties applies |
| 03 | power\_duty\_name | Varchar(255) | NOTNULL/ NULL | The department of NHA on which power duties applies |
| 04 | person\_Responsible | Varchar(255) | NOTNULL/ NULL | The witness from NHA who is responsible to Authority power duties relationship. |

* + 1. **Remarks:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No** | **Name** | **Data Type** | **Constraint** | **Description** |
| 01 | Contract\_owner | Varchar(255) | NOT NULL | The name of the the person get contract |
| 02 | Project\_type | Varchar(255) | NOT NULL | The type of project wheather is of highway or motorway. |
| 03 | Reason | Varchar(255) | NOT NULL / NULL | Reason why company black listed |

[2]

* + 1. **Toll\_taxes**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No** | **Name** | **Data Type** | **Constraint** | **Description** |
| 01 | SrNo | int | NOT NULL | It’s the serial number only and keep the total record numbers and is the primary key |
| 02 | Road\_name | Varchar(255) | NOT NULL / NULL | NOT NULL / NULLhis column holds the name of the roads(highways and motorways) on which the toll taxes are applies for different vehicals |
| 03 | Road\_distance | int | NOT NULL / NULL | Hold the length of motorway or high way length on which how many km per toll taxes applies |
| 04 | PPRA\_no | Varchar(20) | NOT NULL / NULL | Hold the tender code which receive or collects the taxes by hold or getting this tender |
| 05 | Car | int | NOT NULL / NULL | It holds the tax rate per kilometer for vehicle type car |
| 06 | Wagon | int | NOT NULL / NULL | It holds the tax rate per kilometer for vehicle type Wagon |
| 07 | MiniBus | int | NOT NULL / NULL | It holds the tax rate per kilometer for vehicle type MiniBus |
| 08 | Bus | int | NOT NULL / NULL | It holds the tax rate per kilometer for vehicle type Bus |
| 09 | TwoThreeAxleTruck | int | NOT NULL / NULL | It holds the tax rate per kilometer for vehicle type TwoThreeAxleTruck |
| 10 | ArticulatedTruck | int | NOT NULL / NULL | It holds the tax rate per kilometer for vehicle type ArticulatedTruck |

[3]

* + 1. **Project**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr. No | Name | Data Type | Constraint | Description |
| 01 | project\_title | varchar(255) | NOT NULL | Its holds the name of the project |
| 02 | Ppra\_no | varchar(20) | NOT NULL / NULL | Hold the uniques code of the project after its tender is get by some company |
| 03 | project\_length | float | NOT NULL / NULL | Hold that this length of this project |
| 04 | cost | int | NOT NULL / NULL | Hold that how much cost applies on it |
| 05 | project\_status | varchar(250) | NOT NULL / NULL | Hold that is it complete or under implementation of future upcomping |
| 06 | project\_type | varchar(255) | NOT NULL / NULL | Its holds that which type of government project it is |
| 07 | Completion\_Date | date | NOT NULL / NULL | Its holds that on which date this projects get completed or last time worked on |
| 08 | Physical\_Progress | int | NOT NULL / NULL | Its hold the percentage of physical progress of the project if more than 97% it means its completed |
| 09 | Financial\_Progress | int | NOT NULL / NULL | It holds the percentage of the financial progress of the project how much estimated amount is and how much had spent on a project |
| 10 | PD\_Name | varchar(255) | NOT NULL / NULL | Its holds the name of the contract under who’s supervision the project is taking place |
| 11 | PD\_Email | varchar(255) | NOT NULL / NULL | Its hold the contract email |

[4]

* + 1. **Tender:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No** | **Name** | **Data Type** | **Constraint** | **Description** |
| 01 | PPRA\_No | varchar(20) | NOT NULL | Act as project ID to identify different projects. |
| 02 | Title\_of\_tender | varchar(255) | NOT NULL / NULL | Hold the unique name of every tender which identify different tenders |
| 03 | Title\_procurement | varchar(255) | NOT NULL / NULL | Hold that how we obtain the and works to get the desire porcuerment |
| 04 | No\_of\_bids\_ | int | NOT NULL / NULL | Hold the number that how much we got the bids for this perticular tender |
| 05 | Contractor | varchar(255) | NOT NULL / NULL | Hold the contractor name |
| 06 | Estimated amount | int | NOT NULL / NULL | Hold the expected amount spend on the tender |
| 07 | Name\_of\_agency | varchar(255) | NOT NULL / NULL | Hold the name of the agency which receive the tender |
| 08 | Ppra\_no\_publish\_date | date | NOT NULL / NULL | Hold the date of when the tender is given and it receive a unque ppra code and it publish officially |

[5]

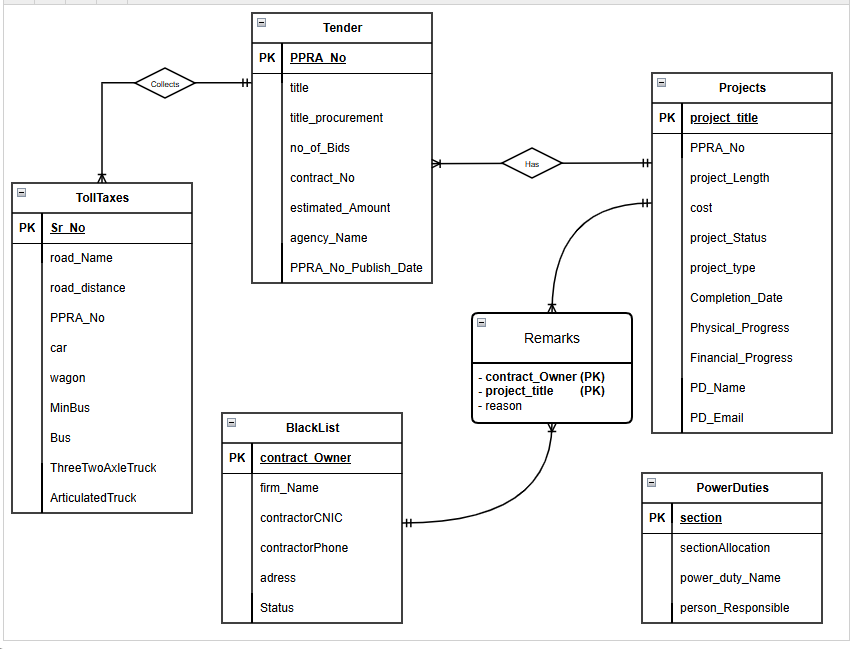
* + 1. **Blacklist:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No** | **Name** | **Data Type** | **Constraint** | **Description** |
| 01 | Contract\_owner | varchar(255) | NOT NULL | The name of contracted who frauded and get blacklisted.. |
| 02 | Name\_of\_firm | varchar(255) | NOT NULL | Name of Firm who have frauded in aproject. |
| 03 | Contractor\_CNIC | varchar(100) | NOT NULL | The CNIC of the person whose firm frauded. |
| 04 | Contractor\_NO | varchar(100) | NOT NULL | The phone number of the contractor. |
| 05 | Adress | varchar(255) | Can be NULL | The address of the person whose firm frauded |
| 06 | statuses | varchar(30) | NOT NULL | The time span for which the firm is BlackListed. |

## RELATIONSHIPS:

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No** | **Participating Entities** | **Relation** | **Business Rule** |
| 01 | TollTaxes , Tenders | Tenders collects Toll taxes | A Tender owner collects taxes from many toll plaza’s (Toll Taxes) but a Toll plaza under only one tender company. |
| 02 | Tenders, Projects | Project has Tenders | A Project can be done by one or many tenders. But one tender is only of one project. |
| 03 | Projects, Remarks | Project get Remarks | A single Project get one or many remarks. But one remarks in only for a single project. |
| 04 | Remarks, Blacklist | Blacklisted after Remarks | A blacklisted firm get many remarks but a single remark is only for one blacklisted company. |
| 06 | Power and duties | Power and Duties | Power and duties can be implemented by multiple authorities But single Authority can implement only a single kind of Power and duties. |

## ENTITY RELATIONSHIP DIAGRAM:



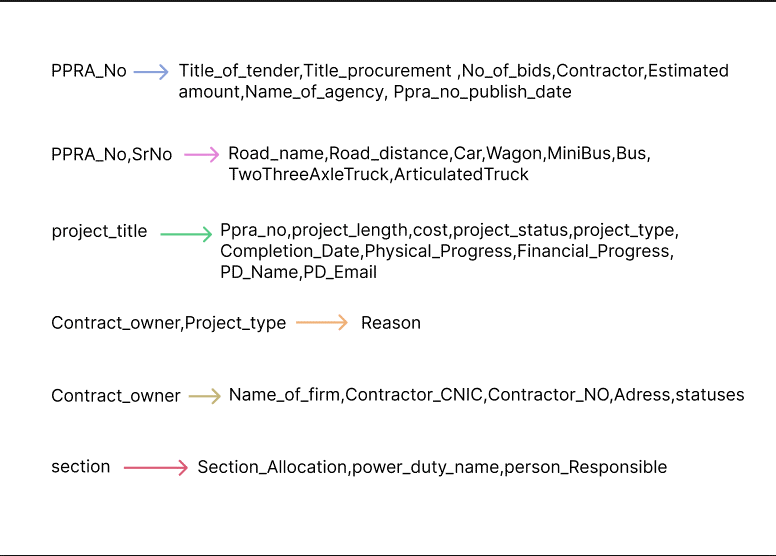
# : Logical DATABASE DESIGN

* 1. **RELATIONAL SCHEMA:**

A screenshot of a computer

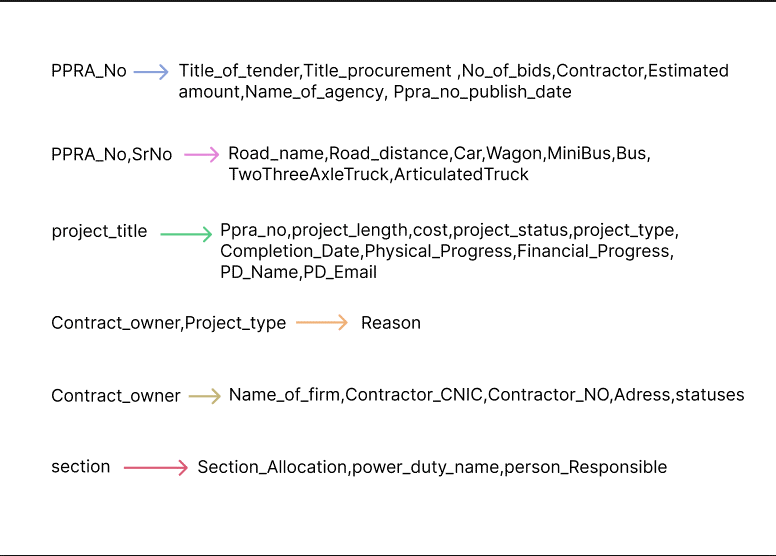
Description automatically generated

* 1. **FUNCTIONAL DEPENDENCIES:**



* 1. **NORMALIZATION:**

In this portion our database project required us to locate and show transitive and partial dependencies. Nevertheless, after carefully reviewing the structure of our database, we discovered that neither transitive nor partial dependencies are there. This is due to the fact that our design makes sure that no non-key attribute depends on any other non-key attribute and that every non-key attribute is totally functionally dependent on the primary key. As a result, the only dependencies our database displays are functional ones, which are explained in the section above. We have avoided redundancy and preserved data integrity by keeping our database in 3rd Normal Form (3NF) by adhering to this structure.



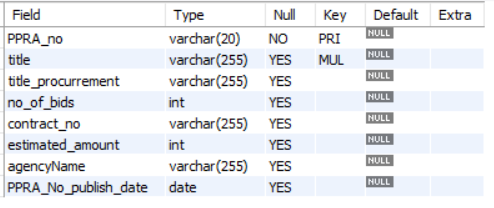
[1]

*.*

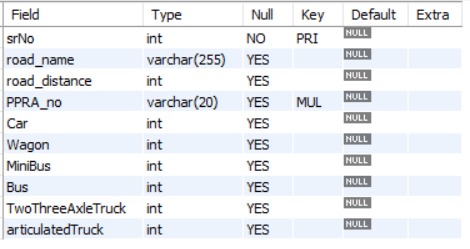
# : Physical DATABASE DESIGN

* 1. **STRUCTURE OF THE TABLES:**

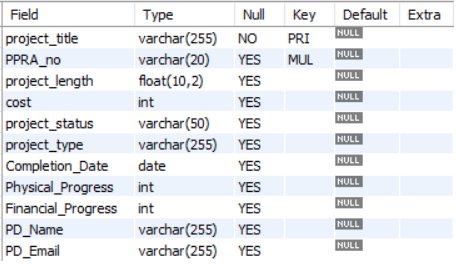
**describe tenders;**



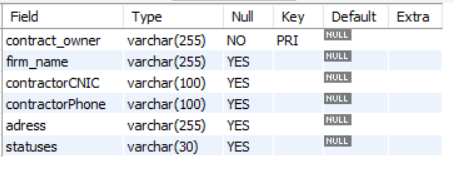
**describe tolltaxes;**



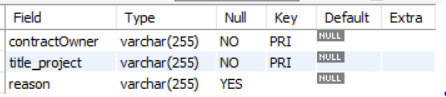
**describe projects;**



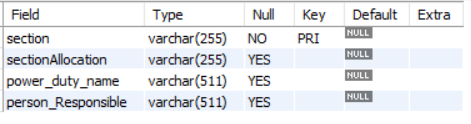
**describe blacklist;**



**describe remarks;**

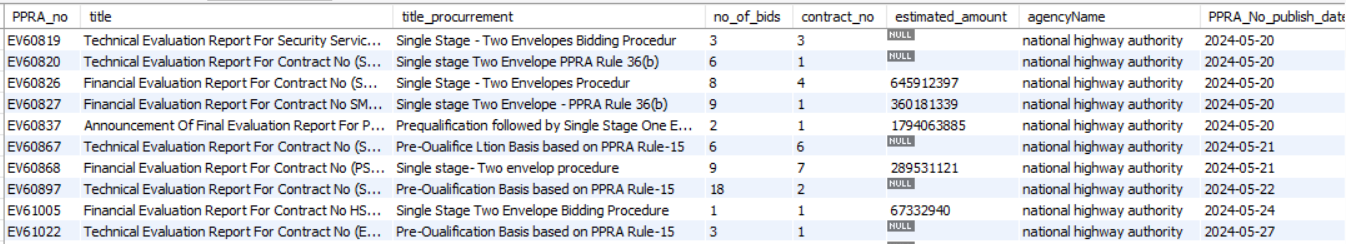


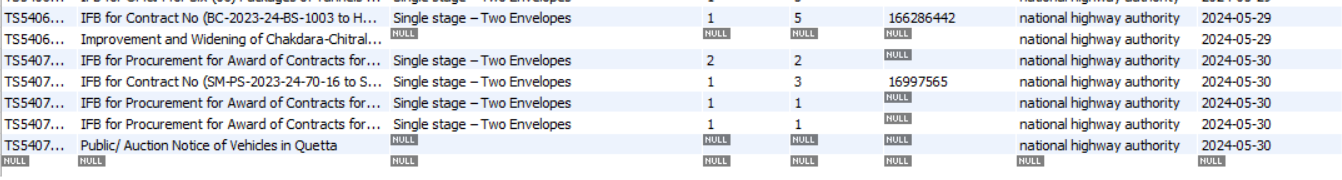
**describe powerduties;**



* 1. **DATA SAMPLES INSIDE TABLES:**

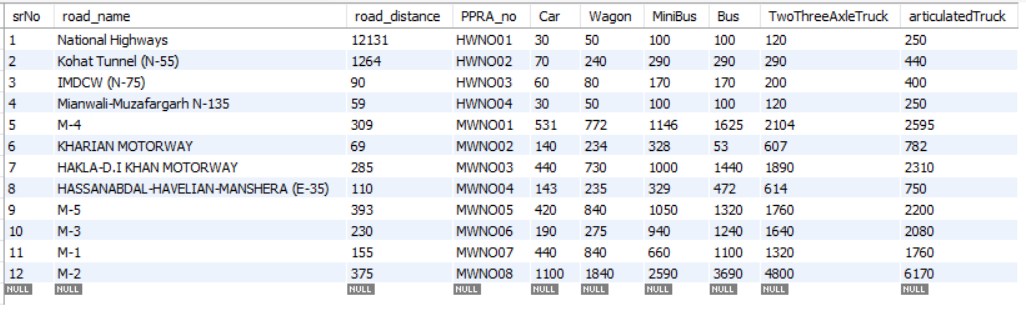
**select \* from tenders;**





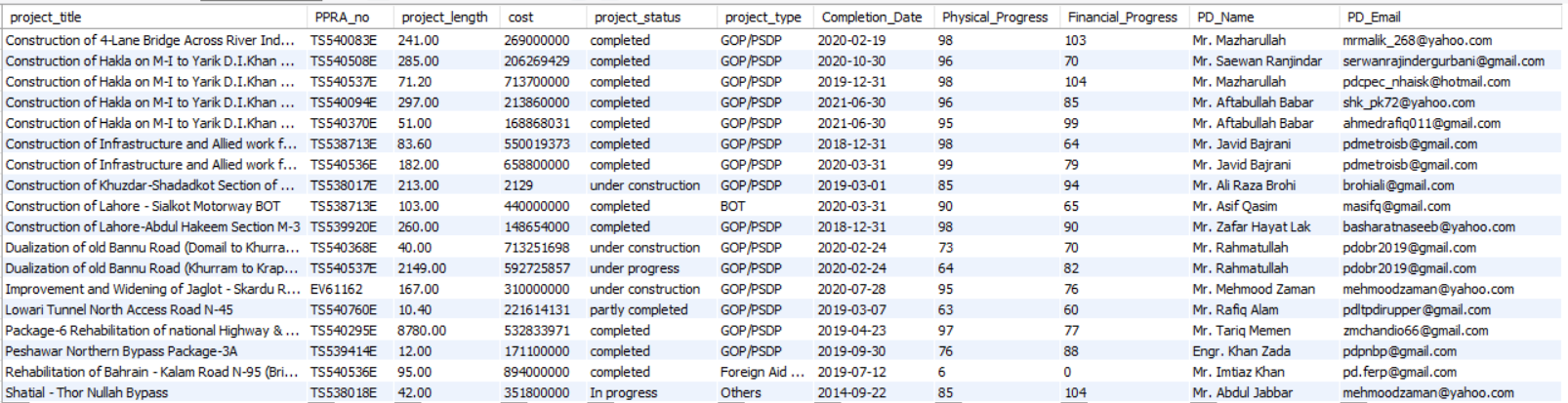


**select \* from tolltaxes;**



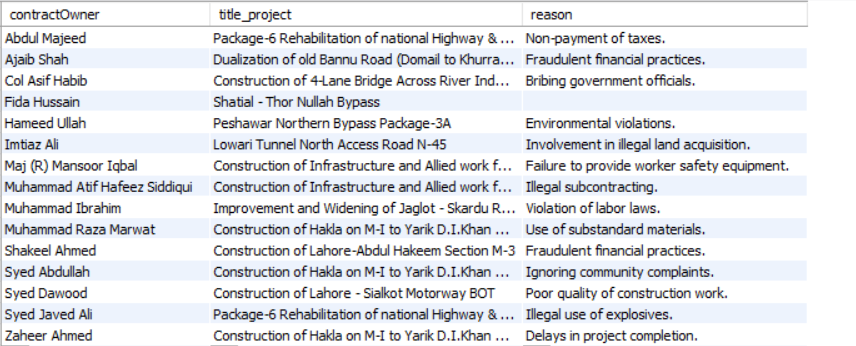


**select \* from projects;**





**select \* from remarks;**



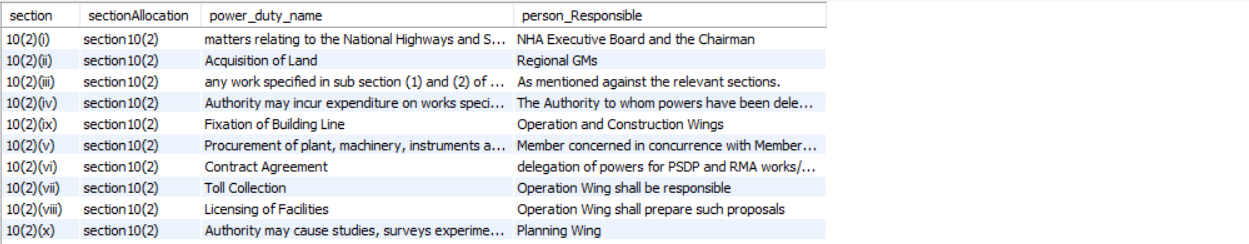


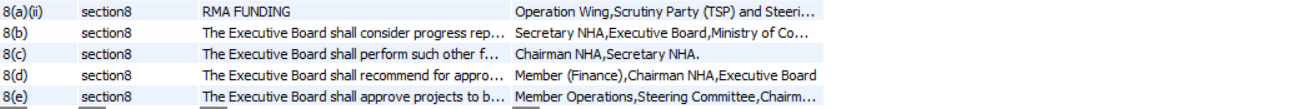
**select \* from blacklist;**





**select \* from powerduties;**





* 1. **QUERIES RESULTS:**

**Query 1 -- (JOIN on 3 Tables: tenders, project, remarks of banned firms):**

select t.PPRA\_No,p.project\_title,r.contractOwner,r.reason as Banned\_Reason

from tenders t

join projects p on t.PPRA\_No = p.PPRA\_No

join remarks r on p.project\_title = r.title\_project;

A screenshot of a computer

Description automatically generated

**Query 2 -- (Compaies Black listed for 3 years for which project):**

select p.PPRA\_No,p.project\_title,b.firm\_name,r.reason as Banned\_Reason,b.statuses

from projects p

join remarks r on p.project\_title = r.title\_project

join blacklist b on b.contract\_owner = r.contractOwner

where b.statuses = "3 Years";

A screenshot of a computer

Description automatically generated

**Query 3 -- (Using group by and Having Clause [No of Duties > 5]):**

select sectionAllocation,count(section) as No\_of\_Duties\_Assigned

from powerduties

group by sectionAllocation

having No\_of\_Duties\_Assigned > 5;

A screenshot of a computer

Description automatically generated

**Query 4 -- (Average Toll Taxes of Vehicles on Pakistani Roads):**

select avg(car) as AVG\_CAR\_TAX,avg(wagon) as AVG\_Wagon\_TAX,

avg(miniBus) as AVG\_MINIBus\_TAX,

avg(bus) as AVG\_Bus\_TAX,

avg(twothreeaxletruck) as AVG\_Truck\_TAX,

avg(articulatedTruck) as AVG\_ArticulatedTruck\_TAX

from tolltaxes;



**Query 5 -- (Most Expensive(Toll Tax) Road in Pakistan for Trucks):**

select road\_name,twothreeAxleTruck as HighestTax\_for\_Trucks

from tolltaxes

where twothreeAxleTruck = (select max(twothreeAxleTruck) from tolltaxes);

A close up of a number

Description automatically generated

**Query 6 -- (5 most Expensive Projects by NHA):**

select p.ppra\_no, project\_title,project\_length,cost,t.agencyname

from projects p,tenders t

where p.ppra\_no = t.ppra\_no and agencyname = "national highway authority"

order by cost desc limit 5;

A screenshot of a computer screen

Description automatically generated

**Query 7 -- (Create a function to calculate the AVG total toll tax):**

CREATE FUNCTION TotalTollTaxForRoad()

RETURNS DECIMAL(10, 2)

DETERMINISTIC

BEGIN

DECLARE totalTollTax DECIMAL(10, 2);

SELECT avg(car + wagon + MiniBus + Bus + twothreeAxleTruck + ArticulatedTruck) INTO totalTollTax

FROM TollTaxes;

RETURN totalTollTax;

END //

DELIMITER ;

select TotalTollTaxForRoad();

A close up of a text

Description automatically generated

**Query 8 -- (Find the projects that have more bids than the average number of bids across all tenders.)**

SELECT project\_title, P.PPRA\_No, no\_of\_Bids

FROM Projects P

JOIN Tenders T ON P.PPRA\_No = T.PPRA\_No

WHERE T.no\_of\_Bids > (select avg(no\_of\_bids) from tenders);

A screenshot of a computer

Description automatically generated

**Query 9 -- (List the names of contractors who have been blacklisted and are also associated with remarks.)**

SELECT firm\_Name FROM BlackList WHERE contract\_Owner in (SELECT contractOwner FROM Remarks);

*A black screen with white text

Description automatically generated***Query 10 -- (Find the average estimated amount of tenders for each agency.)**

SELECT agencyName, AVG(estimated\_Amount) AS average\_estimated\_amount FROM Tenders GROUP BY agencyName HAVING AVG(estimated\_Amount) > 100000;

A black screen with white text

Description automatically generated

# : Interface Design

## LANGUAGE/FRAMEWORK:

We have used the Tkinter Library of Python library, It is an excellent choice for developing GUI client Interfaces.

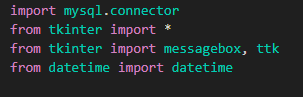
* Tkinter's syntax is beginner-friendly, making it a smooth curve for Python programmers. It uses Python classes and functions that closely resemble native Python objects, and creates widgets which contain content and finally gives a clean client Interfaces.
* Tkinter client applications are known for their speed and minimal resource usage, ideal for running efficiently even on low specs PC.
* No additional installation is required in Case of Tkinter for users with Python, streamlining deployment and avoiding compatibility issues.

Tkinter provides a rich set of standard GUI widgets like buttons, labels, text boxes, and more, allowing you to build a wide range of functionalities.

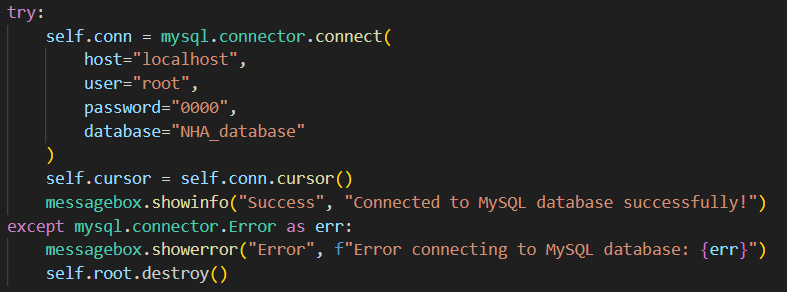
* 1. **DATABASE CONNECTIVITY:**

Tkinter itself doesn't handle database connections. Python provides the mySQL.connector library to establish a connection with a MySQL database.In our case we have connected to SQL Database in our Each Function like Insert\_data,Delete\_data etc. By this we got a smooth connection.

These Libraries help us to connect with SQL and Tkinter.



And Following code contain Database connection and proper Error Handling.



* 1. **STORED PROCEDURES AND FUNCTIONS:**

**Insert Function**

The insert\_project() function captures project details from a GUI, ensuring all fields like project title, PPRA number, cost, and completion status are filled before proceeding. It connects to a local MySQL database (nha\_database) using provided credentials (root and password), constructs an SQL INSERT query with placeholders for data security, and executes it to store the information. Upon successful insertion, it notifies the user, while error handling manages database issues and ensures proper closure of the database connection.

def insert\_project():

    project\_title = project\_title\_entry.get()

    ppra\_no = ppra\_no\_entry.get()

    project\_length = project\_length\_entry.get()

    cost = cost\_entry.get()

    project\_status = project\_status\_entry.get()

    project\_type = project\_type\_entry.get()

    completion\_date = completion\_date\_entry.get()

    physical\_progress = physical\_progress\_entry.get()

    financial\_progress = financial\_progress\_entry.get()

    pd\_name = pd\_name\_entry.get()

    pd\_email = pd\_email\_entry.get()

    if not project\_title or not ppra\_no or not project\_length or not cost or not project\_status or not project\_type or not completion\_date or not physical\_progress or not financial\_progress or not pd\_name or not pd\_email:

        messagebox.showwarning("Input Error", "Please fill in all fields")

        return

    try:

        connection = mysql.connector.connect(

            host='localhost',

            user='root',

            password='0000',

            database='nha\_database'

        )

        cursor = connection.cursor()

        sql\_insert\_query = """INSERT INTO Projects (project\_title, PPRA\_no, project\_length, cost, project\_status, project\_type, Completion\_Date, Physical\_Progress, Financial\_Progress, PD\_Name, PD\_Email) VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)"""

        cursor.execute(sql\_insert\_query, (project\_title, ppra\_no, project\_length, cost, project\_status, project\_type, completion\_date, physical\_progress, financial\_progress, pd\_name, pd\_email))

        connection.commit()

        messagebox.showinfo("Success", "Data submitted successfully")

        display\_projects()

    except mysql.connector.Error as err:

        messagebox.showerror("Error", f"Error: {err}")

    finally:

        if connection.is\_connected():

            cursor.close()

            connection.close()

            print("MySQL connection is closed")

**Delete Function**

The delete\_project() function allows users to delete project records from a MySQL database (nha\_database) based on a specified project title input. It first retrieves the project title from a GUI input field (delete\_project\_title\_entry) and validates its presence. Upon validation, it establishes a connection to the database using credentials (root and password) and constructs an SQL DELETE query (sql\_delete\_query) to remove records where the project title matches the input.

 project\_title = delete\_project\_title\_entry.get()

    if not project\_title:

        messagebox.showwarning("Input Error", "Please enter a Project Title to delete")

        return

    try:

        connection = mysql.connector.connect(

            host='localhost',

            user='root',

            password='eshna 123@',

            database='nha\_database'

        )

        cursor = connection.cursor()

        sql\_delete\_query = """DELETE FROM Projects WHERE project\_title = %s"""

        cursor.execute(sql\_delete\_query, (project\_title,))

        connection.commit()

        if cursor.rowcount == 0:

            messagebox.showinfo("Info", "No record found with the given Project Title")

        else:

            messagebox.showinfo("Success", "Data deleted successfully")

            display\_projects()

    except mysql.connector.Error as err:

        messagebox.showerror("Error", f"Error: {err}")

    finally:

        if connection.is\_connected():

            cursor.close()

            connection.close()

            print("MySQL connection is closed")

**Search Function**

The search\_project() function retrieves a project's details from a MySQL database based on user-provided project title input. It validates that the title is entered, displaying a warning if not. Upon connecting to nha\_database with root credentials, it executes an SQL SELECT query to fetch all columns from the Projects table where the title matches the input. If a result is found, it updates result\_label with formatted project details; if not, it indicates "No record found". Error handling manages potential database errors, showing detailed messages on failure, and ensures proper closure of the database connection for efficiency and data integrity.

def search\_project():

    project\_title = search\_project\_title\_entry.get()

    if not project\_title:

        messagebox.showwarning("Input Error", "Please enter a Project Title to search")

        return

    try:

        connection = mysql.connector.connect(

            host='localhost',

            user='root',

            password='eshna 123@',

            database='nha\_database'

        )

        cursor = connection.cursor()

        sql\_search\_query = """SELECT \* FROM Projects WHERE project\_title = %s"""

        cursor.execute(sql\_search\_query, (project\_title,))

        result = cursor.fetchone()

        if result:

            result\_label.config(text=f"Project Title: {result[0]}\nPPRA No: {result[1]}\nProject Length: {result[2]}\nCost: {result[3]}\nProject Status: {result[4]}\nProject Type: {result[5]}\nCompletion Date: {result[6]}\nPhysical Progress: {result[7]}\nFinancial Progress: {result[8]}\nPD Name: {result[9]}\nPD Email: {result[10]}")

        else:

            result\_label.config(text="No record found")

    except mysql.connector.Error as err:

        messagebox.showerror("Error", f"Error: {err}")

    finally:

        if connection.is\_connected():

            cursor.close()

            connection.close()

            print("MySQL connection is closed")

**Display Function**

The display\_projects function first clears any existing entries in the tree widget and then attempts to connect to a MySQL database named nha\_database using the provided credentials. Upon a successful connection, it retrieves all rows from the Projects table and inserts each row into the tree widget. If an error occurs during the database operations, an error message is displayed using a message box. Finally, the function ensures that the database connection is properly closed, printing a confirmation message if the connection is successfully terminated.

def display\_projects():

    for row in tree.get\_children():

        tree.delete(row)

    try:

        connection = mysql.connector.connect(

            host='localhost',

            user='root',

            password='eshna 123@',

            database='nha\_database'

        )

        cursor = connection.cursor()

        cursor.execute("SELECT \* FROM Projects")

        rows = cursor.fetchall()

        for row in rows:

            tree.insert("", END, values=row)

    except mysql.connector.Error as err:

        messagebox.showerror("Error", f"Error: {err}")

    finally:

        if connection.is\_connected():

            cursor.close()

            connection.close()

            print("MySQL connection is closed")

**Update Function**

The update\_project function begins by retrieving the project title from the project\_title\_entry widget. If no project title is provided, it displays a warning message prompting the user to select a record to update. The function then collects data from various entry widgets corresponding to the project details such as PPRA number, project length, cost, status, type, completion date, physical progress, financial progress, PD name, and PD email. It attempts to establish a connection to the nha\_database MySQL database using the provided credentials. Upon successful connection, it constructs and executes an SQL UPDATE query to modify the project details for the selected project title. If the operation is successful, it commits the changes to the database, displays a success message, and refreshes the project list by calling the display\_projects function. In case of an error during the database operations, an error message is displayed. Finally, the function ensures the database connection is closed properly, printing a confirmation message if the connection is successfully terminated.

def update\_project():

    project\_title = project\_title\_entry.get()

    if not project\_title:

        messagebox.showwarning("No Item Selected", "Please select a record to update")

        return

    ppra\_no = ppra\_no\_entry.get()

    project\_length = project\_length\_entry.get()

    cost = cost\_entry.get()

    project\_status = project\_status\_entry.get()

    project\_type = project\_type\_entry.get()

    completion\_date = completion\_date\_entry.get()

    physical\_progress = physical\_progress\_entry.get()

    financial\_progress = financial\_progress\_entry.get()

    pd\_name = pd\_name\_entry.get()

    pd\_email = pd\_email\_entry.get()

    try:

        connection = mysql.connector.connect(

            host='localhost',

            user='root',

            password='eshna 123@',

            database='nha\_database'

        )

        cursor = connection.cursor()

        sql\_update\_query = """UPDATE Projects SET PPRA\_no = %s, project\_length = %s, cost = %s, project\_status = %s, project\_type = %s, Completion\_Date = %s, Physical\_Progress = %s, Financial\_Progress = %s, PD\_Name = %s, PD\_Email = %s WHERE project\_title = %s"""

        cursor.execute(sql\_update\_query, (ppra\_no, project\_length, cost, project\_status, project\_type, completion\_date, physical\_progress, financial\_progress, pd\_name, pd\_email, project\_title))

        connection.commit()

        messagebox.showinfo("Success", "Data updated successfully")

        display\_projects()

    except mysql.connector.Error as err:

        messagebox.showerror("Error", f"Error: {err}")

    finally:

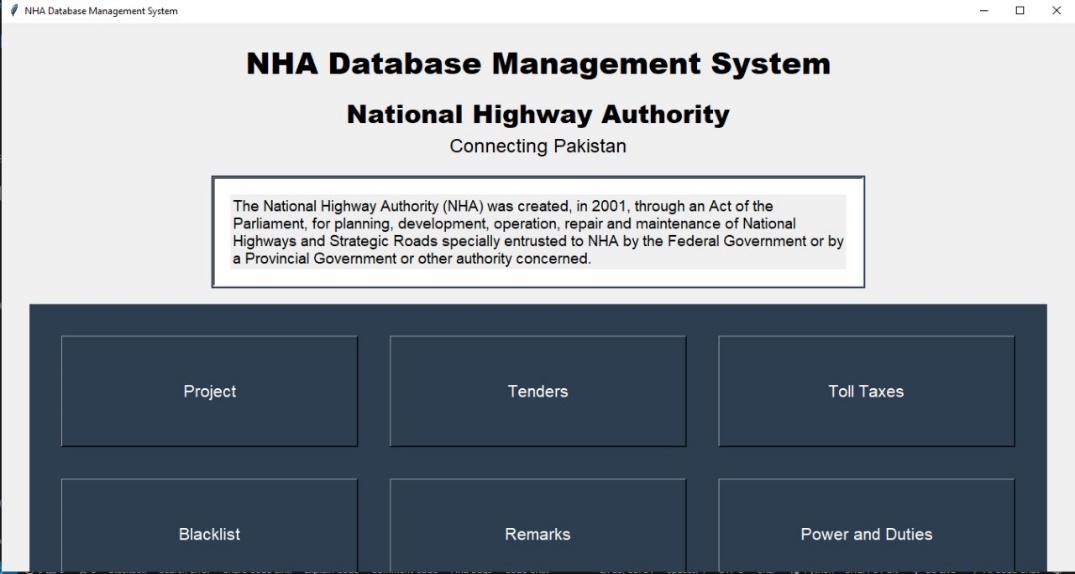
        if connection.is\_connected():

            cursor.close()

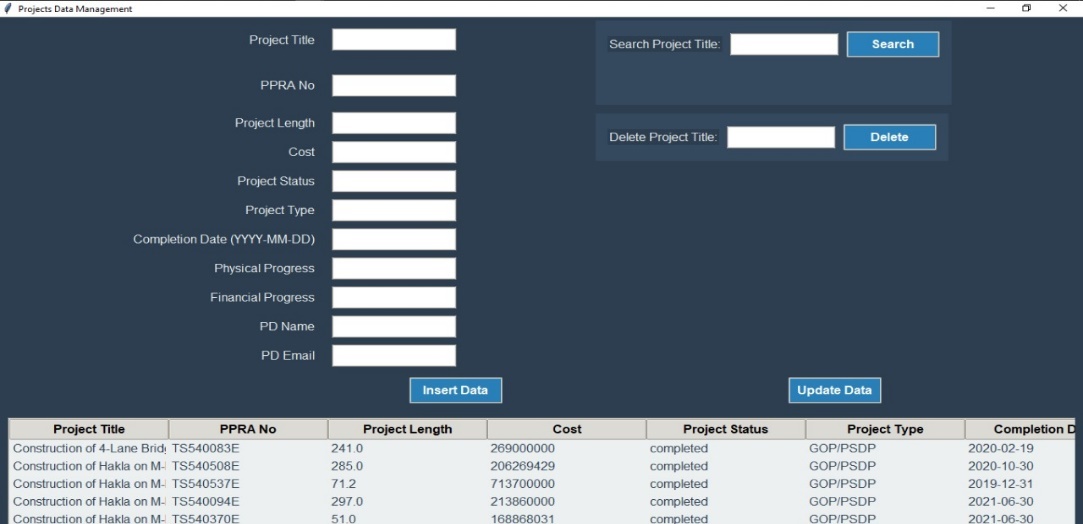
            connection.close()

            print("MySQL connection is closed")

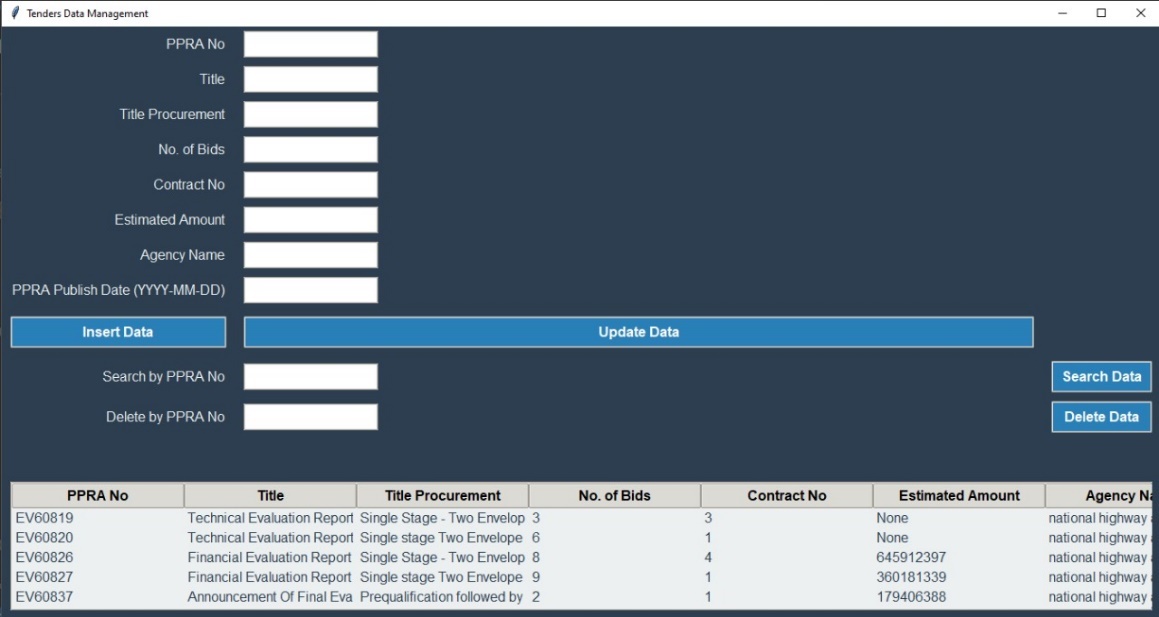
* 1. **INTERFACES:**



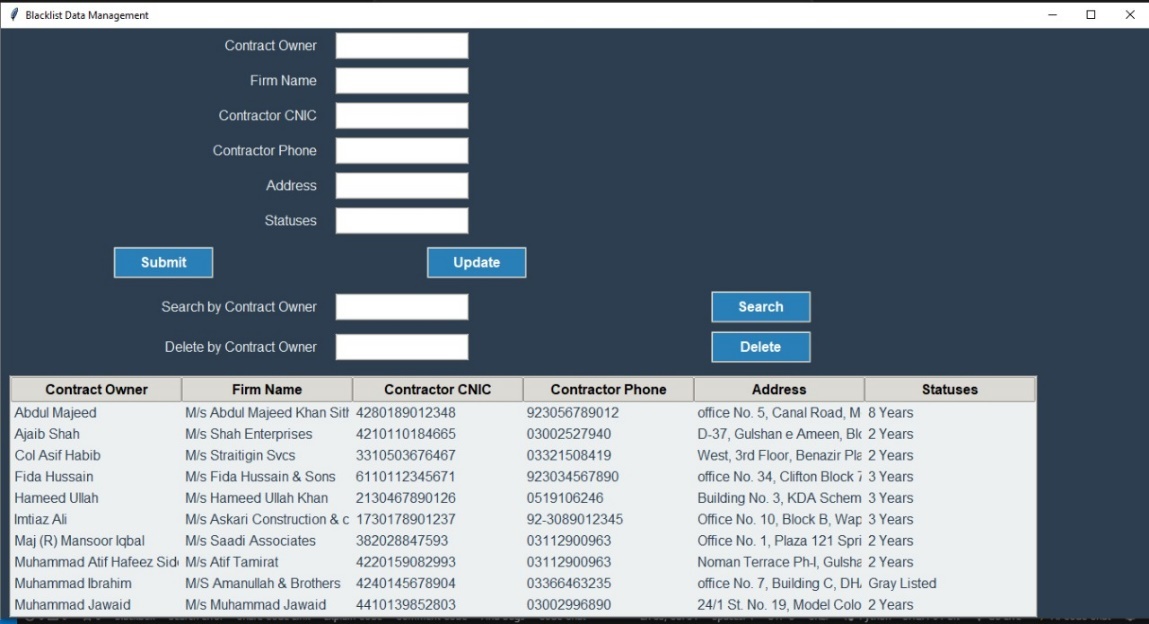
This main page of the application serves as the dashboard for the NHA Database Management System. It features a title and a description of the National Highway Authority (NHA), which outlines its purpose and functions. The page includes six large buttons labeled "Project," "Tenders," "Toll Taxes," "Blacklist," "Remarks," and "Power and Duties." Each button directs the user to a specific section of the application, allowing them to manage different aspects of the NHA database.



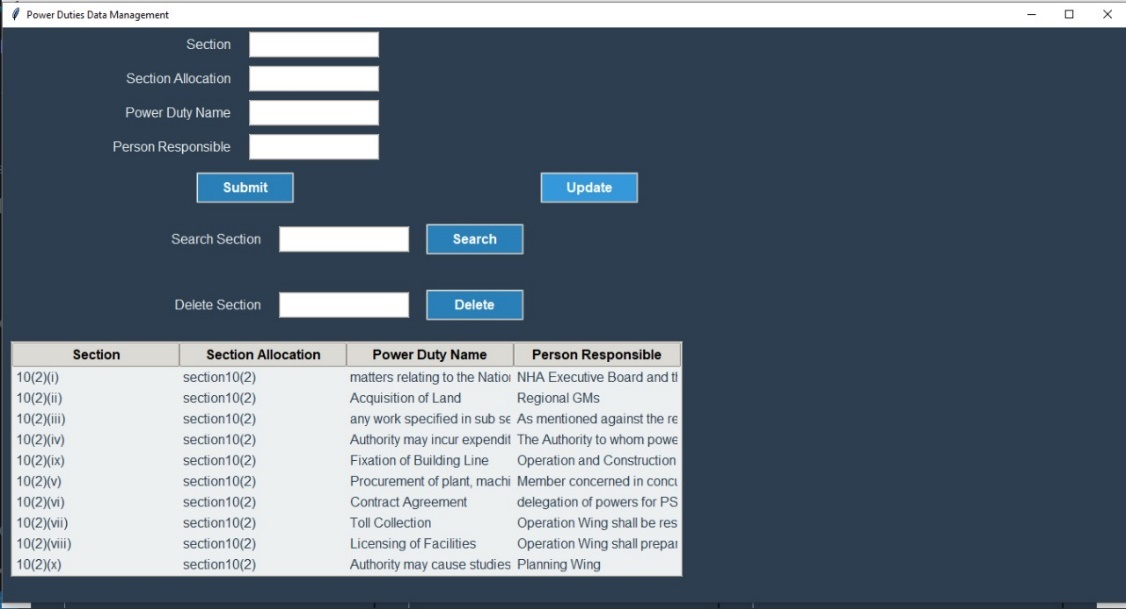
This page is committed to keeping the NHA database's project data organized. A number of project attributes, including "Project Title, PPRA No, Project Length, Cost," "Project Status," "Project Type," "Completion Date," "Physical Progress, Financial Progress, PD Name, and PD Email," are entered into the fields on the left. To add new project details to the database, users can enter them and then click Insert Data. A search function to locate projects by title and a delete function to remove projects by title are located in the right section. A table containing the current project records, including the project title, PPRA number, length, cost, status, type, and completion date, is located beneath these controls. By choosing a record and selecting Update Data, users can also make changes to the project's current information.



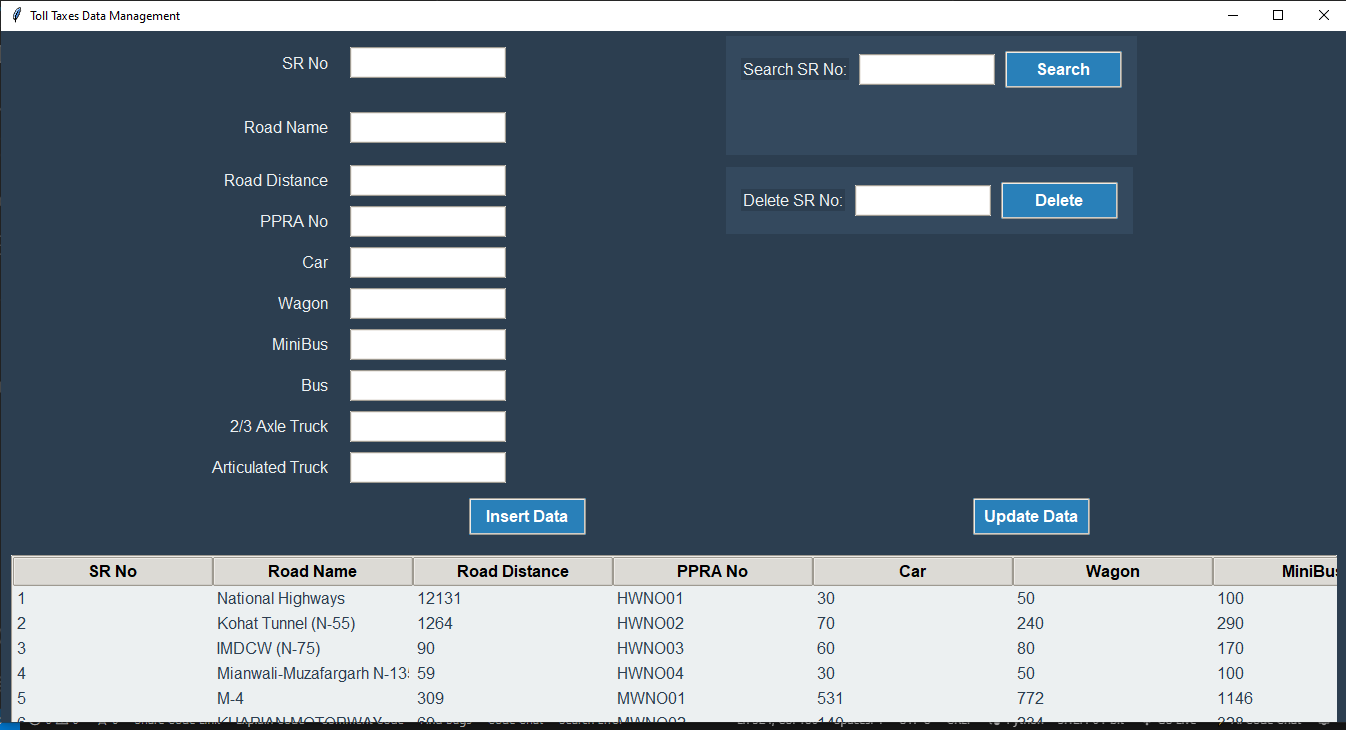
This page is dedicated to managing tender data within the NHA database. The left section contains entry fields for various tender attributes such as "PPRA\_no," "title," "title\_procurrement," "no\_of\_bids," "contract\_no," "estimated\_amount," "agencyName," and "PPRA\_No\_publish\_date." Users can enter new tender details and click "Insert Data" to add the information to the database. The right section features a search function to find tenders by PPRA number and a delete function to remove tenders by PPRA number. Below these controls, there's a table that displays existing tender records, showing details like PPRA\_no, title, title\_procurrement, no\_of\_bids, contract\_no, estimated\_amount, agencyName, and PPRA\_No\_publish\_date. Users can also update existing tender information by selecting a record and clicking "Update Data."

**

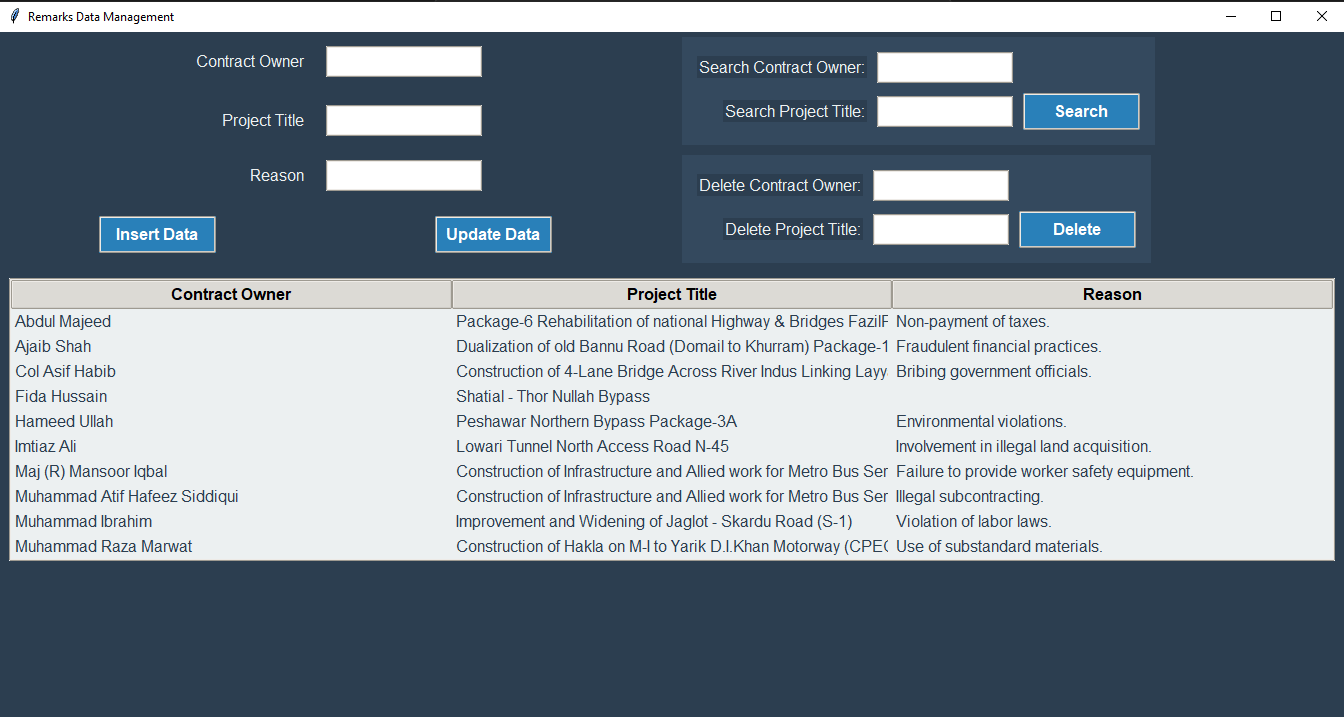
This page is dedicated to managing blacklist data within the NHA database. The left section contains entry fields for various blacklist attributes such as "contract\_owner," "firm\_name," "contractorCNIC," "contractorPhone," "address," and "statuses." Users can enter new blacklist details and click "Insert Data" to add the information to the database. The right section features a search function to find blacklisted companies by contract owner and a delete function to remove blacklisted companies by contract owner. Below these controls, there's a table that displays existing blacklist records, showing details like contract\_owner, firm\_name, contractorCNIC, contractorPhone, address, and statuses. Users can also update existing blacklist information by selecting a record and clicking "Update Data."

**

This page is dedicated to managing power and duties data within the NHA database. The left section contains entry fields for various power and duty attributes such as "section," "sectionAllocation," "power\_duty\_name," and "person\_Responsible." Users can enter new power and duty details and click "Insert Data" to add the information to the database. The right section features a search function to find duties by section and a delete function to remove duties by section. Below these controls, there's a table that displays existing power and duty records, showing details like section, sectionAllocation, power\_duty\_name, and person\_Responsible. Users can also update existing power and duty information by selecting a record and clicking "Update Data."

**

This page is dedicated to managing toll taxes data within the NHA database. The left section contains entry fields for various toll tax attributes such as "srNo," "road\_name," "road\_distance," "PPRA\_no," "Car," "Wagon," "MiniBus," "Bus," "TwoThreeAxleTruck," and "articulatedTruck." Users can enter new toll tax details and click "Insert Data" to add the information to the database. The right section features a search function to find toll taxes by serial number and a delete function to remove toll taxes by serial number. Below these controls, there's a table that displays existing toll tax records, showing details like srNo, road\_name, road\_distance, PPRA\_no, Car, Wagon, MiniBus, Bus, TwoThreeAxleTruck, and articulatedTruck. Users can also update existing toll tax information by selecting a record and clicking "Update Data."



This page is dedicated to managing remarks data within the NHA database. The left section contains entry fields for various remark attributes such as "contractOwner," "title\_project," and "reason." Users can enter new remark details and click "Insert Data" to add the information to the database. The right section features a search function to find remarks by contract owner and project title and a delete function to remove remarks by contract owner and project title. Below these controls, there's a table that displays existing remark records, showing details like contractOwner, title\_project, and reason. Users can also update existing remark information by selecting a record and clicking "Update Data."

# : CONCLUSION

## LESSONS LEARNED:

## Throughout the project focused on the "nha" database for the National Highway Authority, we learned how to integrate new technologies like Python with MySQL database management. Learning GUI development in Tkinter and connecting it seamlessly with backend operations was a significant achievement. Technically, effective database management within "nha" using MySQL and robust error handling ensured application stability. Project management-wise, modular development and effective task prioritization within "nha" were essential for meeting deadlines. These experiences also taught us valuable lessons in teamwork, time management, and professionalism, which were crucial for collaborative success and project execution within "nha".

* 1. **CHALLENGES AND SOLUTIONS:**

Throughout the project focused on the "nha" database for the National Highway Authority, we encountered various challenges that tested our skills and teamwork. Integrating Python with MySQL for database management initially posed complexity, addressed through thorough documentation study and leveraging `mysql-connector-python` for connectivity. Designing user-friendly GUIs in Tkinter required iterative refinement based on user feedback to improve usability. Robust error handling throughout the application ensured stability, while modular development and clear code organization minimized redundancy and facilitated maintenance. Effective teamwork and communication were pivotal in overcoming obstacles, ensuring task coordination and goal alignment. Rigorous testing procedures, including unit testing and user acceptance testing, validated functionality and reliability. These challenges underscored the importance of continuous learning, adaptability, and collaborative problem-solving in achieving project milestones effectively within the "nha" framework*.*

* 1. **FUTURE WORK AND IMPROVEMENTS:**

Looking ahead, there are several avenues for enhancing the "nha" database project for the National Highway Authority. Future improvements could focus on expanding functionality by incorporating advanced search filters and data visualization tools to provide deeper insights into tender, blacklist, remarks, toll taxes, and power and duties data. Optimizing database queries and performance tuning can enhance overall system responsiveness, particularly with large datasets. Additionally, integrating user authentication and access control mechanisms would bolster security. Further refinement could involve enhancing GUI aesthetics and responsiveness for a more intuitive user experience. Exploring cloud integration for scalability and real-time data updates could also extend the system's capabilities. These enhancements aim to elevate usability, performance, and security while aligning with evolving organizational needs within the "nha" framework*.*

* 1. **FINAL THOUGHTS:**

Reflecting on the "nha" database project for the National Highway Authority, it has been a journey of significant growth and accomplishment. From mastering Python and MySQL integration to developing robust GUIs in Tkinter, each challenge provided valuable learning opportunities. The project's impact lies in its contribution to streamlining data management processes within the NHA, enhancing efficiency, and fostering better decision-making through organized and accessible information. Personal insights gained include the importance of perseverance, teamwork, and continuous improvement in achieving project success. I am grateful for the support and collaboration of my team members and stakeholders throughout this endeavor, which has been instrumental in realizing our goals and delivering a solution that meets organizational needs effectively.

# REFERENCES

*Provide a list of references*

|  |
| --- |
| [1] "NHA," [Online]. Available: https://nha.gov.pk/ . |
| [2] "ADB," Asian Development Bank (ADB) information on specific projects., [Online]. Available: https://www.adb.org/where-we-work/pakistan/ . |
| [3] "PPRA," Pakistan Procurement Regulatory Authority (PPRA) rules and guidelines., [Online]. Available: https://ppra.org.pk/Rules.asp . |
| [4] "BLACKLIST (NHA)," [Online]. Available: https://nha.gov.pk/index.php/documents/topic/22270. |
| [5] "TOLL TAXES (NHA)," [Online]. Available: https://mail.nha.gov.pk/index.php/sitepages/topic/22566. |
| [6] "PROJECTS (NHA)," [Online]. Available: https://mail.nha.gov.pk/projects. |
| [7] "TENDERS (NHA)," [Online]. Available: https://mail.nha.gov.pk/tenders. |
| [8] "GOP/PSDP," Government of Pakistan Public Sector Development Programm, [Online]. Available: https://www.pc.gov.pk/uploads/archives/PSDP\_2022-23\_Final.pdf. |