COMPUTER SCIENCE PROJECT FILE
ON

Student Record Management

(2023-2024)

OPG WORLD SCHOOL

Submitted By:
Name: Prakhar Jha
Class: XII C
Roll No: 20

CERTIFICATE

This is to certify that Prakhar Jha of class XII-C, OPG World School has completed his project file of Computer Science as prescribed by CBSE in the year 2023-24 under the guidance of **Ms. Ankita Yadav**.

Date:

Signature
(Internal Examiner)

Signature
(External Examiner)

ACKNOWLEDGEMENT

I would like to express a deep sense of gratitude to my practical guide **Ms. Ankita Yadav** for guiding me immensely through the course of the Year. She always displayed a keen interest in my work. Her constructive advice and constant motivation have been responsible for successfully completing this project file. I also thank my parents for their motivation and support. I must thank my classmates for their timely help and support in the compilation of this file.

I would also like to express my gratitude towards our principal **Mrs. Namrata Datta** for providing vital inputs and giving various insights about the subject.

INDEX

Student Record Management

Overview

The Student Record Management project is a Python-based application that facilitates the storage, retrieval, modification, and deletion of CS student records in the school The project incorporates the use of a MySQL database for data storage and manipulation. Matplotlib is used for generating graphs, providing a comprehensive solution for student data management.

Purpose

The primary purpose of this project is to provide a comprehensive solution for addressing the administrative challenges associated with student data management. By combining Python, MySQL, and Matplotlib, it offers a unified platform for record-keeping, data analysis, and data visualization in the educational sector.

Scope

The scope of the Student Record Management project encompasses the following key aspects:

- Storing student records efficiently.
- Retrieving and displaying student information.
- Updating and deleting records as needed.
- Visualizing data using Matplotlib to gain insights into student populations.

Project Structure

The project consists of a Python program that provides the following functionalities:
The project consists of several sections:

1. Database Creation and Table Initialization:
    - The initial part of the program establishes a connection to the MySQL database server hosted locally.
    - If the connection is successful, it proceeds to create a new database named "mcs" and a table named "st" within that database. The "st" table is designed to store essential student information, including admission number, name, class, section, address, and house name.
    - After creating the database and table, the changes are committed to the database.

2. Updating Student Records:

    - In this section, the program allows you to update student records. You can select a student by name and then choose the field (name, class, section, or address) that you want to change. Afterward, you input the new data, and the program updates the records accordingly.

3. Deleting Student Records:
    - This section enables you to delete student records based on various criteria. You can specify a field (e.g., admission number, name, class) and the corresponding value you want to delete.
4. Graph Generation
    - This section uses a python library Matplotlib and mysql.connector
      Matplotlib significantly enhances the Student Record Management project by providing a means to visualize data effectively. In this context, it allows administrators to gain insights into the distribution of students among different house names. This visualization facilitates data analysis, aiding in decision-making processes and administrative tasks

LIBRARIES USED

The project report outlines the utilization of two fundamental libraries, *mysql.connector* and *Matplotlib*, in the context of a student data analysis and visualization program. These libraries serve as the backbone of the project, enabling data understanding, manipulation, analysis, and the creation ofinsightful visualizations.mysql.connector: This library is used for database connectivity and executing SQL operations.

The mysql.connector library is essential for establishing, managing, and interacting with MySQL databases in the Python code. It allows us to execute SQL queries, retrieve and manipulate data, and ensure the integrity of our database transactions. In the graph generation code, mysql.connector is used to connect to the database to fetch data from the "st" table for the graph. It retrieves the class and name information from the database, processes the data, and then creates graphs using Matplotlib to visualize the relationship between class and student names.

Matplotlib: Data visualization using Matplotlib can help project stakeholders, such as school administrators or teachers, gain insights from the data. For example, the bar chart that displays the relationship between class and student names can highlight patterns, trends, or anomalies in the student population. It can answer questions like, "How many students are in each class?" or "Are there any students with the same name in different classes?" The project's goal is to manage student records effectively. Matplotlib assists in this by providing a visual representation of data, enabling educational institutions to make informed decisions regarding class distribution, student performance, and other related aspects. This ultimately contributes to the improvement of administrative processes and educational outcomes.

PYTHON CODE:

 # Connecting python to sql and creating table

create.py - C:\Users\ASUS\OneDrive\Desktop\Project\create.py (3.8.10)

File Edit Format Run Options Window Help

```python
import mysql.connector

# Establish a connection to the MySQL database
con = mysql.connector.connect(user="root", password="Shambhavi@401", host="local

if con.is_connected():
    # Create a cursor to execute SQL commands
    cur = con.cursor()

    # Create the 'mcs' database if it doesn't exist
    cur.execute("CREATE DATABASE IF NOT EXISTS mcs")

    # Switch to the 'mcs' database
    cur.execute("USE mcs")

    # Create the 'st' table with specified columns
    table_creation_query = """
    CREATE TABLE IF NOT EXISTS st (
        admno INT NOT NULL PRIMARY KEY,
        name VARCHAR(20) NOT NULL,
        class INT NOT NULL,
        section VARCHAR(1) NOT NULL,
        address VARCHAR(100),
        house_name VARCHAR(10)
    )
    """
    cur.execute(table_creation_query)

    # Commit the changes to the database
    con.commit()
else:
    print("Sorry, can't connect to the database.")

# Close the connection
con.close()
```

Taking and inserting records

inserting record.py - C:\Users\ASUS\OneDrive\Desktop\Project\inserting record.py (3.8.10)

File Edit Format Run Options Window Help

```
import mysql.connector

def get_student_data():
    # Prompt user for student details
    name = input("Enter name: ")
    admno = int(input("Enter admission no.: "))
    class1 = int(input("Enter class: "))
    section = input("Enter section: ")
    house = input("Enter house name: ")
    address = input("Enter address: ")

    return (admno, name, class1, section, address, house)

# Establish a connection to the MySQL database
con = mysql.connector.connect(user="root", password="Shambhavi@401", host="local

if con.is_connected():
    cur = con.cursor()

    for _ in range(30):
        # Get student data
        student_data = get_student_data()

        # SQL query to insert the student record
        insert_query = "INSERT INTO st (admno, name, class, section, address, ho

        # Execute the query and commit the transaction
        cur.execute(insert_query, student_data)
        con.commit()

        print("Record accepted. Thank you!")

    con.close()
else:
    print("Sorry, can't connect to the database.")
```

Updating Records

update.py - C:\Users\ASUS\OneDrive\Desktop\Project\update.py (3.8.10)

File Edit Format Run Options Window Help

```python
import mysql.connector

# Establish a connection to the MySQL database
con = mysql.connector.connect(user="root", password="Shambhavi@401", host="local

# Check if the connection is established
if con.is_connected():
    cur = con.cursor()

    # Get the name of the student whose record you want to change
    student_name = input("Whose record do you want to change: ")

    # Fetch the student records
    cur.execute("SELECT * FROM st")
    for record in cur:
        if record[1] == student_name:
            print("Student Record:")
            print(f"Admission No.: {record[0]}")
            print(f"Name: {record[1]}")
            print(f"Class: {record[2]}")
            print(f"Section: {record[3]}")
            print(f"Address: {record[4]}")

    # List the fields that can be changed
    print("\nFields that can be changed:")
    print("1. Name")
    print("2. Class")
    print("3. Section")
    print("4. Address")

    # Prompt the user to choose the field to change
    field_choice = input("Which field do you want to change (1/2/3/4): ")

    if field_choice == "1":
        # Change the student's name
        new_name = input("Enter the new name: ")
        query = "UPDATE st SET name = %s WHERE name = %s"
        params = (new_name, student_name)
    elif field_choice == "2":
        # Change the student's class
```

Ln: 63 Col: 0

update.py - C:\Users\ASUS\OneDrive\Desktop\Project\update.py (3.8.10)

File Edit Format Run Options Window Help

```python
    # List the fields that can be changed
    print("\nFields that can be changed:")
    print("1. Name")
    print("2. Class")
    print("3. Section")
    print("4. Address")

    # Prompt the user to choose the field to change
    field_choice = input("Which field do you want to change (1/2/3/4): ")

    if field_choice == "1":
        # Change the student's name
        new_name = input("Enter the new name: ")
        query = "UPDATE st SET name = %s WHERE name = %s"
        params = (new_name, student_name)
    elif field_choice == "2":
        # Change the student's class
        new_class = int(input("Enter the new class: "))
        query = "UPDATE st SET class = %s WHERE name = %s"
        params = (new_class, student_name)
    elif field_choice == "3":
        # Change the student's section
        new_section = input("Enter the new section: ")
        query = "UPDATE st SET section = %s WHERE name = %s"
        params = (new_section, student_name)
    elif field_choice == "4":
        # Change the student's address
        new_address = input("Enter the new address: ")
        query = "UPDATE st SET address = %s WHERE name = %s"
        params = (new_address, student_name)

    # Execute the update query
    cur.execute(query, params)
    print("Record UPDATED")

    # Commit the changes to the database
    con.commit()
else:
    print("Sorry, can't connect to the database.")
```

Display Records

display.py - C:\Users\ASUS\OneDrive\Desktop\Project\display.py (3.8.10)

File Edit Format Run Options Window Help

```
import mysql.connector

# Establish a connection to the MySQL database
con = mysql.connector.connect(user="root", password="Shambhavi@401", host="local

if con.is_connected():
    cur = con.cursor()

    # User menu
    print("DISPLAY\n(a) All records\n(b) Specific record")

    # User's choice
    choice = input("Choose from the above: ")

    cur.execute("SELECT * FROM st")

    if choice.lower() == 'a':
        # Display all records
        for record in cur:
            print(record)
    elif choice.lower() == 'b':
        # Display a specific record by admission number
        admission_no = int(input("Enter admission no.: "))
        for record in cur:
            if record[0] == admission_no:
                print(record)

    con.commit()
    cur.close()
    con.close()
else:
    print("Sorry, can't connect to the database.")
```

Ln: 34 Col: 15

Delete Records

del.py - C:\Users\ASUS\OneDrive\Desktop\Project\del.py (3.8.10)

File Edit Format Run Options Window Help

```python
import mysql.connector

# Establish a connection to the MySQL database
con = mysql.connector.connect(user="root", password="Shambhavi@401", host="localhost", database="mcs")

if con.is_connected():
    cur = con.cursor()

    print("Fields to Delete:", "ADMNO.", "NAME", "CLASS", "SECTION", "ADDRESS", "HOUSE", sep="\n")
    field = input("Choose the field by which you want to delete a record: ").strip()
    value = input("Enter the value to specify the record to delete: ").strip()

    # Dictionary to map field names to table column names
    field_column_mapping = {
        "admno": "admno",
        "name": "name",
        "class": "class",
        "section": "section",
        "address": "address",
        "house": "house_name"
    }

    # Check if the specified field is valid
    if field.lower() in field_column_mapping:
        column_name = field_column_mapping[field.lower()]

        # Create the SQL statement for deletion
        delete_query = f"DELETE FROM st WHERE {column_name} = %s"

        cur.execute(delete_query, (value,))
        con.commit()
        print("Record deleted.")
    else:
        print("Invalid field specified. No records were deleted.")

    con.close()
else:
    print("Sorry, can't connect to the database.")
```

Drop Records

drop.py - C:\Users\ASUS\OneDrive\Desktop\Project\drop.py (3.8.10)

File Edit Format Run Options Window Help

```
import mysql.connector

# Establish a connection to the MySQL database
con = mysql.connector.connect(user="root", password="Shambhavi@401", host="local

if con.is_connected():
    cur = con.cursor()

    # Provide a clear message to the user
    print("Processing to delete the student table.")

    # Ask for user confirmation
    a = input("Are you sure you want to delete the student table? (yes/no): ")

    # Check user input
    if a.lower() == 'yes':
        # Drop the 'st' table
        cur.execute("DROP TABLE st")
        print("Table 'st' DELETED")

    con.commit()
else:
    print("Sorry, can't connect to the database.")
```

Using Matplotlib for Graphs

To find the number of students from each section

prj.py - C:/Users/ASUS/OneDrive/Desktop/Project/prj.py (3.8.10)

File Edit Format Run Options Window Help

```
import mysql.connector
import matplotlib.pyplot as plt

# Establish a connection to the MySQL database
con = mysql.connector.connect(user="root", password="Shambhavi@401", host="localhost", database="mcs"

if con.is_connected():
    cur = con.cursor()

    # Select the data from the "st" table
    cur.execute("SELECT section, COUNT(*) FROM st GROUP BY section")

    # Fetch the data
    data = cur.fetchall()

    # Separate sections and student counts into separate lists
    sections = [record[0] for record in data]
    student_counts = [record[1] for record in data]

    con.close()

    # Create a line graph to visualize the data
    plt.figure(figsize=(10, 6))
    plt.plot(sections, student_counts, marker='o', linestyle='-', color='b', markerfacecolor='r')

    # Add labels and title
    plt.xlabel('Sections')
    plt.ylabel('Number of Students')
    plt.title('Number of Students in Each Section')

    # Show the plot
    plt.grid(True)
    plt.tight_layout()
    plt.show()

else:
    print("Sorry, can't connect to the database.")
```

To find the number of students from each region

prj.py - C:/Users/ASUS/OneDrive/Desktop/Project/prj.py (3.8.10)

File Edit Format Run Options Window Help

```python
import mysql.connector
import matplotlib.pyplot as plt

# Establish a connection to the MySQL database
con = mysql.connector.connect(user="root", password="Shambhavi@401", host="localhost", database="mcs"

if con.is_connected():
    cur = con.cursor()

    # Select the data from the "st" table
    cur.execute("SELECT address, COUNT(*) AS count FROM st GROUP BY address")

    # Fetch all the records
    data = cur.fetchall()

    # Separate addresses and student counts into separate lists
    addresses = [record[0] for record in data]
    student_counts = [record[1] for record in data]

    con.close()

    # Create a bar chart to visualize the data
    plt.figure(figsize=(12, 6))
    plt.bar(addresses, student_counts)

    # Add labels and title
    plt.xlabel('Address')
    plt.ylabel('Number of Students')
    plt.title('Number of Students from Each Address')

    # Rotate x-axis labels for better readability
    plt.xticks(rotation=45, ha='right')

    # Show the plot
    plt.tight_layout()
    plt.show()

else:
    print("Sorry, can't connect to the database.")
```

To find the no of students vs House Block

prj.py - C:/Users/ASUS/OneDrive/Desktop/Project/prj.py (3.8.10)

File Edit Format Run Options Window Help

```
import mysql.connector
import matplotlib.pyplot as plt
# Establish a connection to the MySQL database
con = mysql.connector.connect(user="root", password="Shambhavi@401", host="localhost", database="mcs"
if con.is_connected():
    cur = con.cursor()
    # Select the data from the "st" table
    cur.execute("SELECT house_name FROM st")
    # Fetch all the records
    data = cur.fetchall()
    # Close the database connection
    con.close()
    # Create a dictionary to count the occurrence of house name first letters
    house_letter_count = {}

    # Iterate through the records and count the first letter of each house name
    for record in data:
        house_name = record[0]
        first_letter = house_name[0].upper()  # Get the first letter and convert to uppercase
        if first_letter not in house_letter_count:
            house_letter_count[first_letter] = 1
        else:
            house_letter_count[first_letter] += 1
    # Extract data for plotting
    letters = list(house_letter_count.keys())
    counts = list(house_letter_count.values())
    # Create a bar chart to visualize the data
    plt.figure(figsize=(10, 6))
    plt.bar(letters, counts)

    # Add labels and title
    plt.xlabel('House Name First Letter')
    plt.ylabel('Number of Students')
    plt.title('Number of Students vs. House Name First Letter')
    # Show the plot
    plt.tight_layout()
    plt.show()
else:
    print("Sorry, can't connect to the database.")
```

OUTPUT SCREEN

 # Inserting Records

IDLE Shell 3.8.10

File Edit Shell Debug Options Window Help

```
Python 3.8.10 (tags/v3.8.10:3d8993a, May  3 2021, 11:48:03) [MSC v.1928 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Users\ASUS\OneDrive\Desktop\Project\create.py ==========
>>>
====== RESTART: C:\Users\ASUS\OneDrive\Desktop\Project\inserting record.py =====
Enter name: Akshay
Enter admission no.: 101
Enter class: 12
Enter section: A
Enter house name: A-102
Enter address: Dwarka,Delhi
Record accepted. Thank you!
Enter name: Akshat
Enter admission no.: 105
Enter class: 12
Enter section: C
Enter house name: B-403
Enter address: Dwarka,Delhi
Record accepted. Thank you!
Enter name: Ananya
Enter admission no.: 301
Enter class: 12
Enter section: C
Enter house name: A-401
Enter address: Dwarka,Delhi
Record accepted. Thank you!
Enter name: Prakhar Jha
Enter admission no.: 303
Enter class: 12
Enter section: C
Enter house name: A-401
Enter address: Dwarka,Delhi
Record accepted. Thank you!
```

IDLE Shell 3.8.10

File Edit Shell Debug Options Window Help

```
Enter name: Vaishnavi
Enter admission no.: 404
Enter class: 12
Enter section: B
Enter house name: A-903
Enter address: Dwarka,Delhi
Record accepted. Thank you!
Enter name: Palak
Enter admission no.: 892
Enter class: 12
Enter section: B
Enter house name: L-102
Enter address: Janakpuri,Delhi
Record accepted. Thank you!
Enter name: Chehak
Enter admission no.: 511
Enter class: 12
Enter section: A
Enter house name: A-511
Enter address: Dwarka,Delhi
Record accepted. Thank you!
Enter name: Puneet
Enter admission no.: 512
Enter class: 12
Enter section: B
Enter house name: B-512
Enter address: Janakpuri,Delhi
Record accepted. Thank you!
Enter name: Rushil
Enter admission no.: 569
Enter class: 12
Enter section: B
Enter house name: X-769
Enter address: Dwarka Mor
Record accepted. Thank you!
Enter name: Navya
Enter admission no.: 561
Enter class: 12
Enter section: A
Enter house name: w-111
```

IDLE Shell 3.8.10

File Edit Shell Debug Options Window Help

```
Enter name: Navya
Enter admission no.: 561
Enter class: 12
Enter section: A
Enter house name: w-111
Enter address: Dwarka Mor
Record accepted. Thank you!
Enter name: Utsmaya
Enter admission no.: 402
Enter class: 12
Enter section: C
Enter house name: A-402
Enter address: Dwarka,Delhi
Record accepted. Thank you!
Enter name: Vadhish
Enter admission no.: 999
Enter class: 12
Enter section: A
Enter house name: Q-918
Enter address: Janakpuri,Delhi
Record accepted. Thank you!
Enter name: Rudra
Enter admission no.: 469
Enter class: 12
Enter section: A
Enter house name: B-004
Enter address: Dwarka,Delhi
Record accepted. Thank you!
Enter name: Purunjay
Enter admission no.: 774
Enter class: 12
Enter section: A
Enter house name: A-103
Enter address: Dwarka,Delhi
Record accepted. Thank you!
Enter name: Akshita
Enter admission no.: 222
Enter class: 12
Enter section: B
Enter house name: Q-003
```

IDLE Shell 3.8.10

File Edit Shell Debug Options Window Help

```
Enter name: Akshita
Enter admission no.: 222
Enter class: 12
Enter section: B
Enter house name: Q-003
Enter address: Janakpuri,Delhi
Record accepted. Thank you!
Enter name: Vidur
Enter admission no.: 343
Enter class: 12
Enter section: C
Enter house name: X-999
Enter address: Dwarka Mor
Record accepted. Thank you!
Enter name: Kashish
Enter admission no.: 121
Enter class: 12
Enter section: A
Enter house name: E-404
Enter address: Dwarka,Delhi
Record accepted. Thank you!
Enter name: Bharat
Enter admission no.: 498
Enter class: 12
Enter section: B
Enter house name: Z-111
Enter address: Janakpuri,Delhi
Record accepted. Thank you!
Enter name: Saraswati
Enter admission no.: 174
Enter class: 12
Enter section: A
Enter house name: Q-105
Enter address: Janakpuri,Delhi
Record accepted. Thank you!
Enter name: Riya
Enter admission no.: 419
Enter class: 12
Enter section: C
Enter house name: A-890
```

```
Enter name: Riya
Enter admission no.: 419
Enter class: 12
Enter section: C
Enter house name: A-890
Enter address: Dwarka,Delhi
Record accepted. Thank you!
Enter name: Ashmit
Enter admission no.: 134
Enter class: 12
Enter section: C
Enter house name: A-212
Enter address: Dwarka,Delhi
Record accepted. Thank you!
Enter name: Madhur
Enter admission no.: 767
Enter class: 12
Enter section: B
Enter house name: A-313
Enter address: Dwarka,Delhi
Record accepted. Thank you!
Enter name: Madhuri
Enter admission no.: 767
Enter class: 12
Enter section: B
```

IDLE Shell 3.8.10

File Edit Shell Debug Options Window Help

```
====== RESTART: C:\Users\ASUS\OneDrive\Desktop\Project\inserting record.py =====
Enter name: Madhuri
Enter admission no.: 768
Enter class: 12
Enter section: B
Enter house name: A-314
Enter address: Dwarka,Delhi
Record accepted. Thank you!
Enter name: Ali
Enter admission no.: 207
Enter class: 12
Enter section: A
Enter house name: C-123
Enter address: Dwarka,Delhi
Record accepted. Thank you!
Enter name: Qudrat
Enter admission no.: 991
Enter class: 12
Enter section: B
Enter house name: D-300
Enter address: Dwarka Mor
Record accepted. Thank you!
Enter name: Keshav
Enter admission no.: 227
Enter class: 12
Enter section: B
Enter house name: C-111
Enter address: Dwarka Mor
Record accepted. Thank you!
Enter name: Krishna
Enter admission no.: 352
Enter class: 12
Enter section: A
Enter house name: C-999
Enter address: Dwarka Mor
Record accepted. Thank you!
Enter name: Birbal
Enter admission no.: 894
Enter class: 12
Enter section: C
```

IDLE Shell 3.8.10

File Edit Shell Debug Options Window Help

```
Enter section: B
Enter house name: D-300
Enter address: Dwarka Mor
Record accepted. Thank you!
Enter name: Keshav
Enter admission no.: 227
Enter class: 12
Enter section: B
Enter house name: C-111
Enter address: Dwarka Mor
Record accepted. Thank you!
Enter name: Krishna
Enter admission no.: 352
Enter class: 12
Enter section: A
Enter house name: C-999
Enter address: Dwarka Mor
Record accepted. Thank you!
Enter name: Birbal
Enter admission no.: 894
Enter class: 12
Enter section: C
Enter house name: C-777
Enter address: Dwarka Mor
Record accepted. Thank you!
Enter name: Wasim
Enter admission no.: 770
Enter class: 12
Enter section: A
Enter house name: A-401
Enter address: Dwarka,Delhi
Record accepted. Thank you!
Enter name: Anushka
Enter admission no.: 001
Enter class: 12
Enter section: A
Enter house name: A-001
Enter address: Dwarka,Delhi
Record accepted. Thank you!
>>>
```

Command Prompt - mysql -uroot -p

```
mysql> select * from st;
+-------+-------------+-------+---------+-----------------+------------+
| admno | name        | class | section | address         | house_name |
+-------+-------------+-------+---------+-----------------+------------+
|     1 | Anushka     |    12 | A       | Dwarka,Delhi    | A-001      |
|   101 | Akshay      |    12 | A       | Dwarka,Delhi    | A-102      |
|   105 | Akshat      |    12 | C       | Dwarka,Delhi    | B-403      |
|   121 | Kashish     |    12 | A       | Dwarka,Delhi    | E-404      |
|   134 | Ashmit      |    12 | C       | Dwarka,Delhi    | A-212      |
|   174 | Saraswati   |    12 | A       | Janakpuri,Delhi | Q-105      |
|   207 | Ali         |    12 | A       | Dwarka,Delhi    | C-123      |
|   222 | Akshita     |    12 | B       | Janakpuri,Delhi | Q-003      |
|   227 | Keshav      |    12 | B       | Dwarka Mor      | C-111      |
|   301 | Ananya      |    12 | C       | Dwarka,Delhi    | A-401      |
|   303 | Prakhar Jha |    12 | C       | Dwarka,Delhi    | A-401      |
|   343 | Vidur       |    12 | C       | Dwarka Mor      | X-999      |
|   352 | Krishna     |    12 | A       | Dwarka Mor      | C-999      |
|   402 | Utsmaya     |    12 | C       | Dwarka,Delhi    | A-402      |
|   404 | Vaishnavi   |    12 | B       | Dwarka,Delhi    | A-903      |
|   419 | Riya        |    12 | C       | Dwarka,Delhi    | A-890      |
|   469 | Rudra       |    12 | A       | Dwarka,Delhi    | B-004      |
|   498 | Bharat      |    12 | B       | Janakpuri,Delhi | Z-111      |
|   511 | Chehak      |    12 | A       | Dwarka,Delhi    | A-511      |
|   512 | Puneet      |    12 | B       | Janakpuri,Delhi | B-512      |
|   561 | Navya       |    12 | A       | Dwarka Mor      | w-111      |
|   569 | Rushil      |    12 | B       | Dwarka Mor      | X-769      |
|   767 | Madhur      |    12 | B       | Dwarka,Delhi    | A-313      |
|   768 | Madhuri     |    12 | B       | Dwarka,Delhi    | A-314      |
|   770 | Wasim       |    12 | A       | Dwarka,Delhi    | A-401      |
|   774 | Purunjay    |    12 | A       | Dwarka,Delhi    | A-103      |
|   892 | Palak       |    12 | B       | Janakpuri,Delhi | L-102      |
|   894 | Birbal      |    12 | C       | Dwarka Mor      | C-777      |
|   991 | Qudrat      |    12 | B       | Dwarka Mor      | D-300      |
|   999 | Vadhish     |    12 | A       | Janakpuri,Delhi | Q-918      |
+-------+-------------+-------+---------+-----------------+------------+
30 rows in set (0.00 sec)

mysql>
```

Display Records

```
IDLE Shell 3.8.10
File  Edit  Shell  Debug  Options  Window  Help
Python 3.8.10 (tags/v3.8.10:3d8993a, May  3 2021, 11:48:03) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Users\ASUS\OneDrive\Desktop\Project\display.py ==========
DISPLAY
(a) All records
(b) Specific record
Choose from the above: b
Enter admission no.: 768
(768, 'Madhuri', 12, 'B', 'Dwarka,Delhi', 'A-314')
>>>
========== RESTART: C:\Users\ASUS\OneDrive\Desktop\Project\display.py ==========
DISPLAY
(a) All records
(b) Specific record
Choose from the above: a
(1, 'Anushka', 12, 'A', 'Dwarka,Delhi', 'A-001')
(101, 'Akshay', 12, 'A', 'Dwarka,Delhi', 'A-102')
(105, 'Akshat', 12, 'C', 'Dwarka,Delhi', 'B-403')
(121, 'Kashish', 12, 'A', 'Dwarka,Delhi', 'E-404')
(134, 'Ashmit', 12, 'C', 'Dwarka,Delhi', 'A-212')
(174, 'Saraswati', 12, 'A', 'Janakpuri,Delhi', 'Q-105')
(207, 'Ali', 12, 'A', 'Dwarka,Delhi', 'C-123')
(222, 'Akshita', 12, 'B', 'Janakpuri,Delhi', 'Q-003')
(227, 'Keshav', 12, 'B', 'Dwarka Mor', 'C-111')
(301, 'Ananya', 12, 'C', 'Dwarka,Delhi', 'A-401')
(303, 'Prakhar Jha', 12, 'C', 'Dwarka,Delhi', 'A-401')
(343, 'Vidur', 12, 'C', 'Dwarka Mor', 'X-999')
(352, 'Krishna', 12, 'A', 'Dwarka Mor', 'C-999')
(402, 'Utsmaya', 12, 'C', 'Dwarka,Delhi', 'A-402')
(404, 'Vaishnavi', 12, 'B', 'Dwarka,Delhi', 'A-903')
(419, 'Riya', 12, 'C', 'Dwarka,Delhi', 'A-890')
(469, 'Rudra', 12, 'A', 'Dwarka,Delhi', 'B-004')
(498, 'Bharat', 12, 'B', 'Janakpuri,Delhi', 'Z-111')
(511, 'Chehak', 12, 'A', 'Dwarka,Delhi', 'A-511')
(512, 'Puneet ', 12, 'B', 'Janakpuri,Delhi', 'B-512')
(561, 'Navya', 12, 'A', 'Dwarka Mor', 'w-111')
(569, 'Rushil', 12, 'B', 'Dwarka Mor', 'X-769')
(767, 'Madhur', 12, 'B', 'Dwarka,Delhi', 'A-313')
(768, 'Madhuri', 12, 'B', 'Dwarka,Delhi', 'A-314')
(770, 'Wasim', 12, 'A', 'Dwarka,Delhi', 'A-401')
(774, 'Purunjay', 12, 'A', 'Dwarka,Delhi', 'A-103')
(892, 'Palak', 12, 'B', 'Janakpuri,Delhi', 'L-102')
(894, 'Birbal', 12, 'C', 'Dwarka Mor', 'C-777')
(991, 'Qudrat', 12, 'B', 'Dwarka Mor', 'D-300')
(999, 'Vadhish', 12, 'A', 'Janakpuri,Delhi', 'Q-918')
>>>
```

Update Records

```
=========== RESTART: C:\Users\ASUS\OneDrive\Desktop\Project\update.py ==========
Whose record do you want to change: Anushka
Student Record:
Admission No.: 1
Name: Anushka
Class: 12
Section: A
Address: Dwarka,Delhi

Fields that can be changed:
1. Name
2. Class
3. Section
4. Address
Which field do you want to change (1/2/3/4): 3
Enter the new section: A
Record UPDATED
>>>
```

Ln: 255 Col: 4

Delete Records

IDLE Shell 3.8.10

File Edit Shell Debug Options Window Help

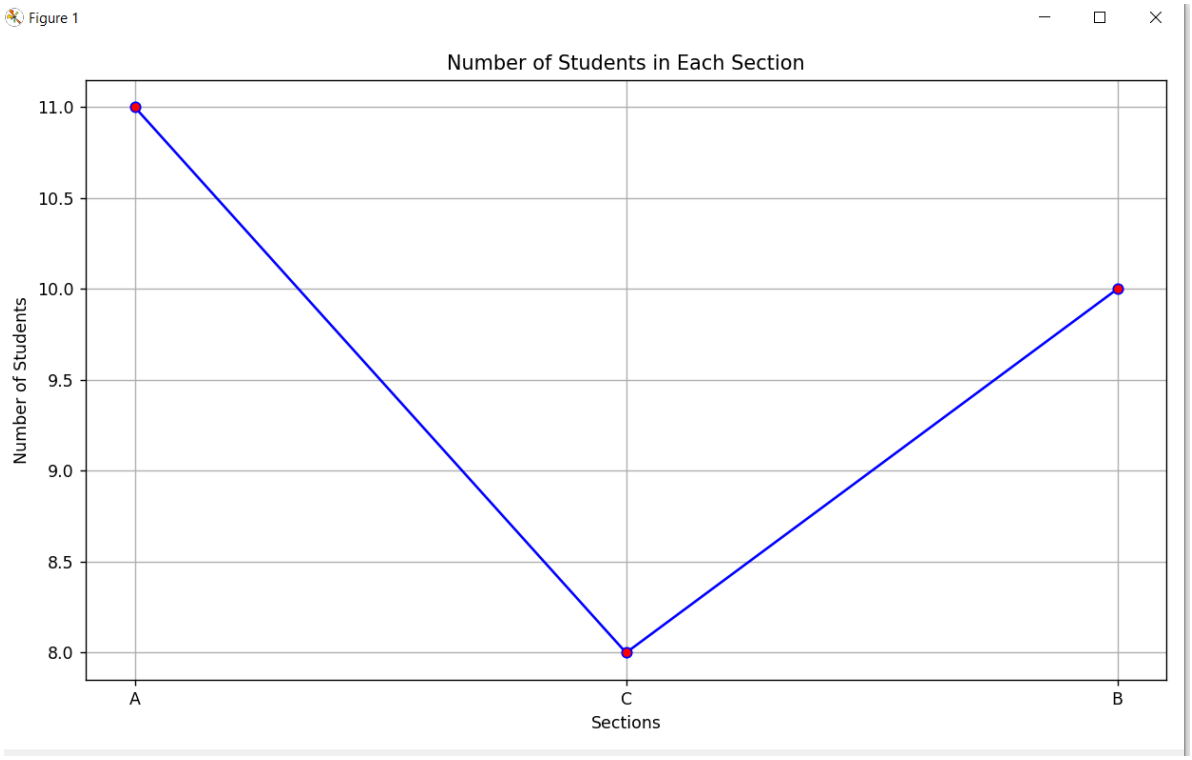
```
Python 3.8.10 (tags/v3.8.10:3d8993a, May  3 2021, 11:48:03) [MSC v.1928 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============ RESTART: C:\Users\ASUS\OneDrive\Desktop\Project\del.py ============
Fields to Delete:
ADMNO.
NAME
CLASS
SECTION
ADDRESS
HOUSE
Choose the field by which you want to delete a record: admno
Enter the value to specify the record to delete: 207
Record deleted.
>>>
```

Drop Records

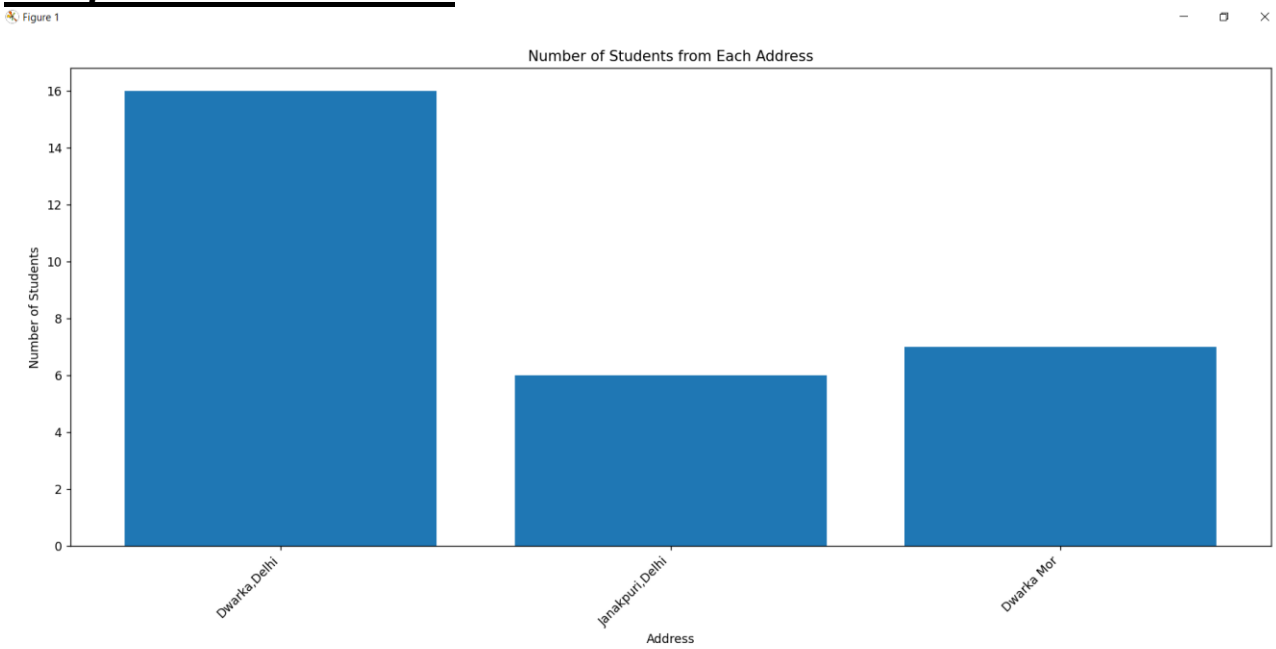
```
IDLE Shell 3.8.10
File  Edit  Shell  Debug  Options  Window  Help
Python 3.8.10 (tags/v3.8.10:3d8993a, May  3 2021, 11:48:03) [MSC v.1928 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============ RESTART: C:\Users\ASUS\OneDrive\Desktop\Project\drop.py ===========
Processing to delete the student table.
Are you sure you want to delete the student table? (yes/no): no
>>>
```

Matplotlib

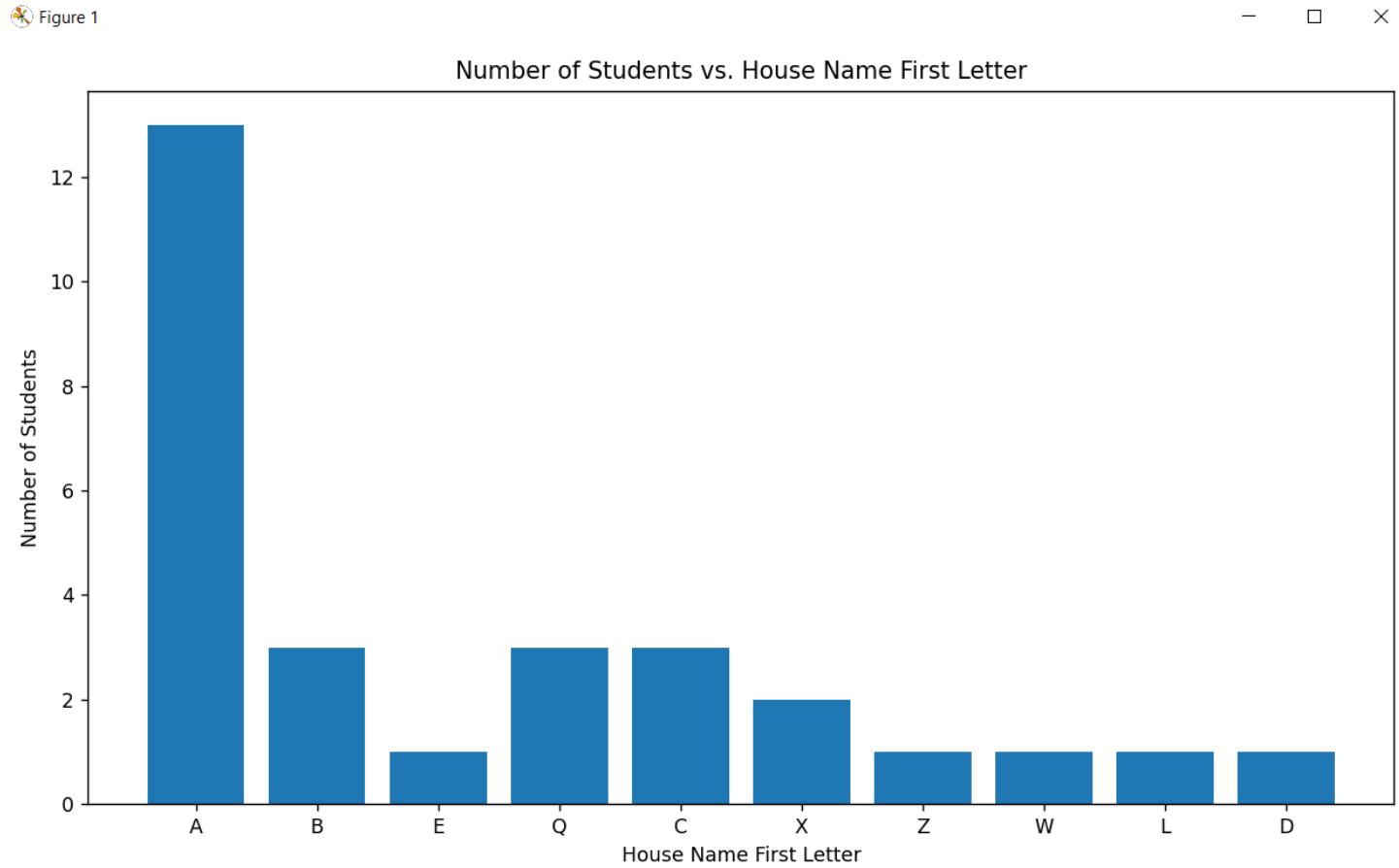
No of students vs section



No of students vs Address



No of students vs House Block



CONCLUSION

The Student Record Management project marks a significant milestone in the realm of educational administration and data management. This robust project, powered by Python, MySQL, Matplotlib, and various other tools, has been meticulously crafted to streamline the process of storing, retrieving, and visualizing student data.

The Student Record Management project is a testament to the power of technology in modern educational institutions. It stands as a comprehensive solution that addresses the administrative challenges of managing student data. As this project unfolds, it showcases how Python, MySQL, and Matplotlib can work in unison to simplify record-keeping, data analysis, and visualization. Moreover, it is a demonstration of the endless possibilities when technology and education converge.

Looking ahead, there is ample room for expansion and enhancement. With a solid foundation in place, the project can be further developed to cater to the ever-evolving needs of educational institutions. Whether it's creating more advanced data visualizations, implementing user authentication, or integrating additional features, the Student Record Management project offers an exciting platform for future growth.
In conclusion, the Student Record Management project is not just a program; it's an embodiment of progress in educational administration. It represents a commitment to accuracy, efficiency, and the ever-improving quest to provide quality education. It is a testament to the power of technology to revolutionize traditional systems and to make a significant impact in the world of education.

BIBLIOGRAPHY

1. http://www.matplotlib.org/
2. Computer Science with python by Sumita Arora
3. Class X AI Facilitator's handbook