**ALL SAINT’S CHURCH SR.SEC. SCHOOL**

**M.I. ROAD , JAIPUR**

**A PROJECT REPORT ON**

**SCHOOL MANAGEMENT SYSTEM**

**SUBJECT: INFORMATICS PRACTICES(065)**

**Session: 2023-2024**

**SUBMITTED BY- SUBMITTED TO-**

**Mohd. Shayyan Mrs. Sharon Amus**

**ACKNOWLEDGMENT**

**I would like to express our special thanks of**

**Gratitude to my teacher Mrs. Sharon Amus for**

**Mentoring me for this project work. I also thank**

**our principal Mrs. Shabnam Haque for her motivation and guidance.**

**My project is titled as School Management System and**

**It enables me to do a lot of research and I came to Know about so many new things.**

**Secondly I would also like to thank our parents and friends who helped us a lot in finalizing this project Within the limited time frame.**

**Mohd. Shayyan**

**XII SCI**

**CONTENT**

1. **Introduction**
2. **System Implementation**
3. **Screenshots**
4. **User Output**
5. **SQL Queries**
6. **User Interface Code**
7. **Testing**
8. **Bibliography**

**INTRODUCTION**

**The “School Management System” undertaken as a project under IP is based on PYTHON AND MYSQL.**

**It’s an attempt to automate the existing system.**

**The project enables its user to perform few**

**operations pertaining to management of School.**

**The Project Enables its user to:**

**1.) Add a new Student, new Staff and new Fee record’s.**

**2.) Delete Student, Staff and Fee record’s.**

**3.) Update Student, Staff and Fee record’s.**

**4.) View Student, Staff and Fee record’s from the Database.**

**System Implementation**

**The hardware used =>**

**--------------------------| System |---------------------------**

**Processor- Intel(R) Core(TM) i5**

**7300U CPU 2.60GHz**

**2.71 GHz**

**Installed memory[RAM]- 8.00 GB(7.88 GB usable)**

**System Type- 64-bit operating system,**

**X64-based processor**

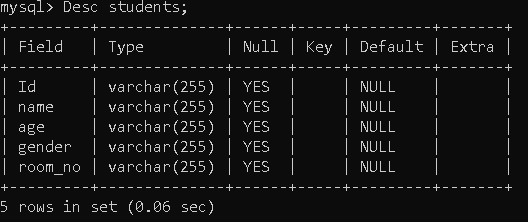
**Pen and Touch- No pen or touch input is**

**Available for this display**

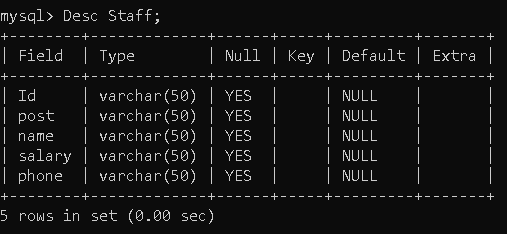
**Screenshots**

**PROJECT TITLE- “SCHOOL MANAGEMENT” DBMS: MySQL Host : localhost User: root Password: root Database: School Table Structure: As per the Screenshot given below:**

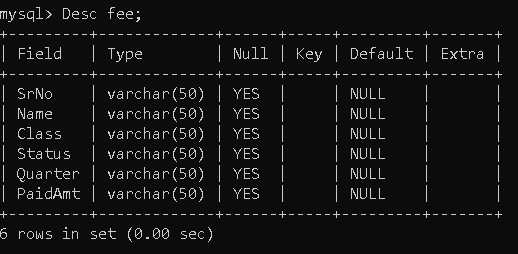
**Table: Student**

****

**Table: Staff**

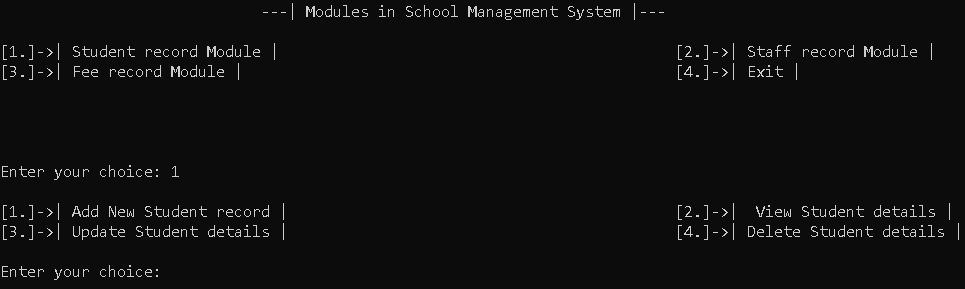
****

**Table: Fee**

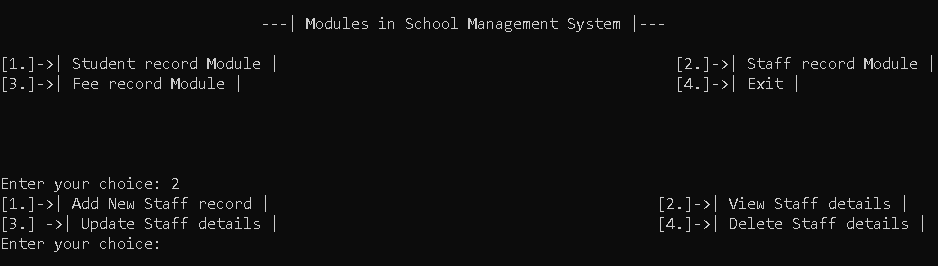
****

**USER OUTPUT**

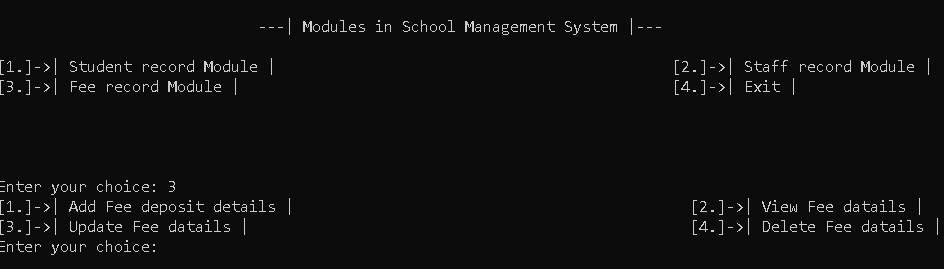
**STUDENT DETAILS:**

****

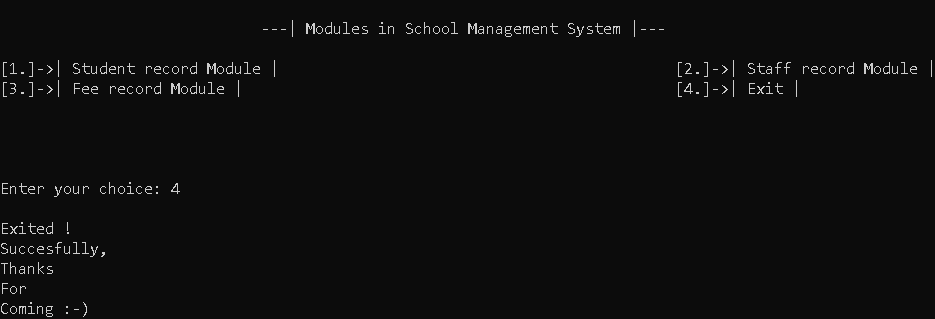
**STAFF DETAILS:**

****

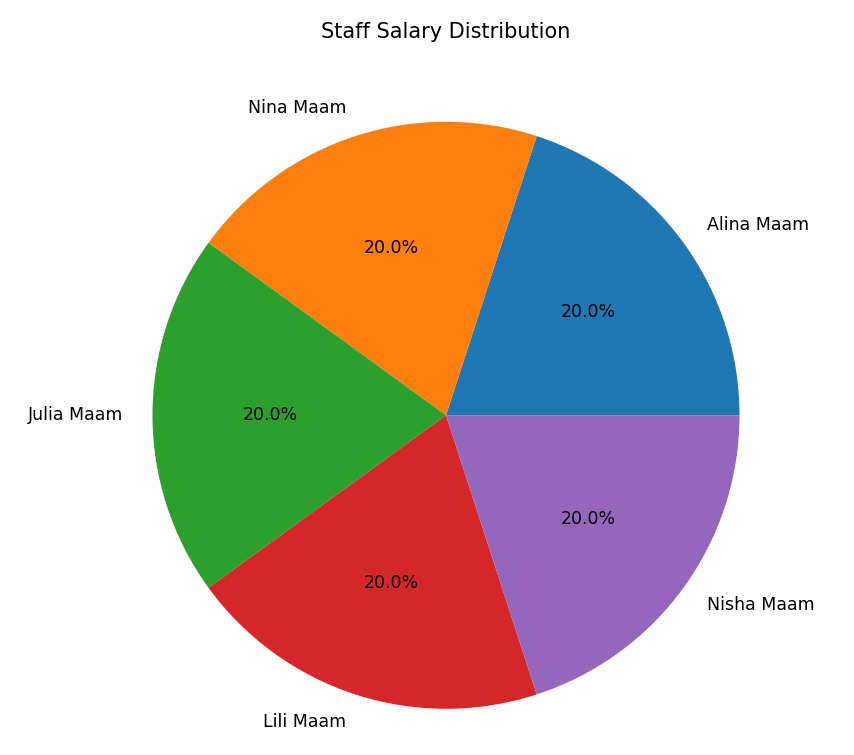
**FEE DETAILS:**

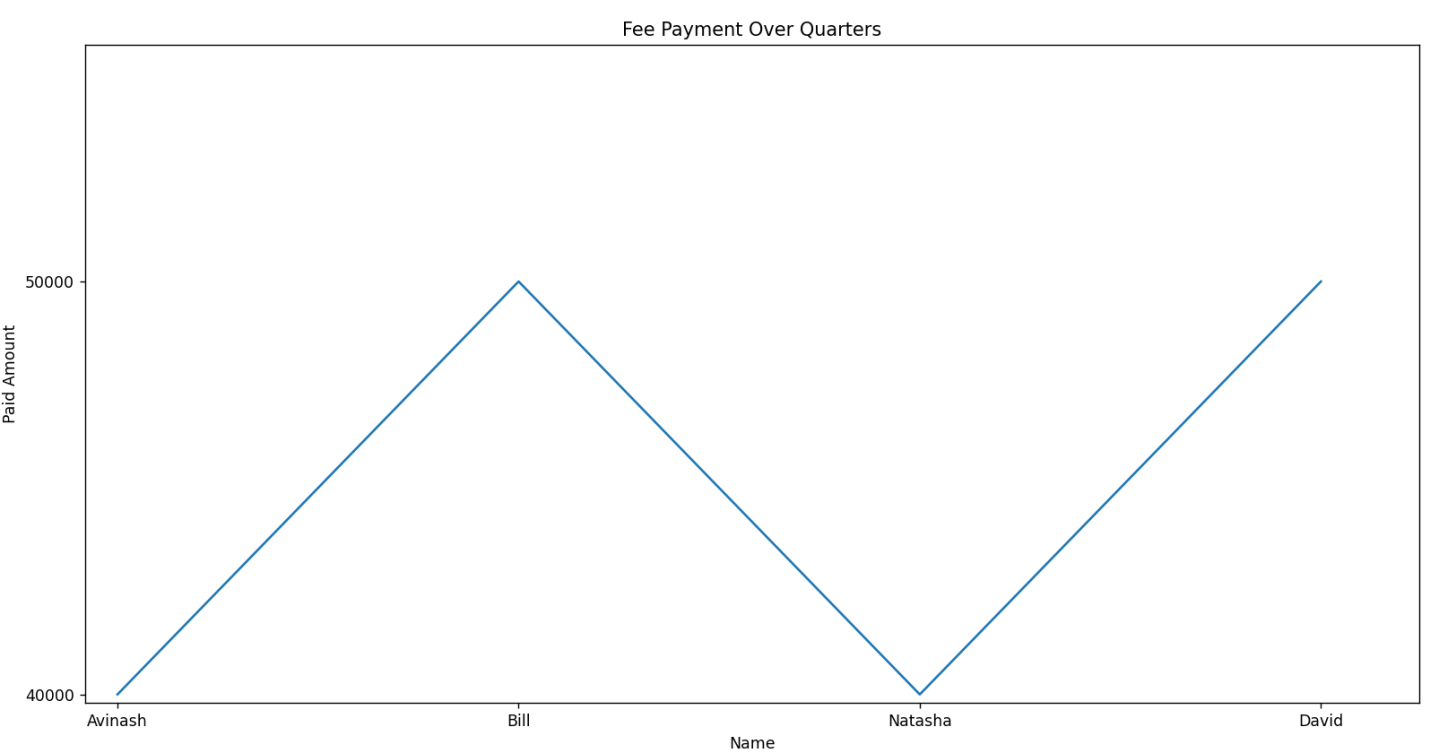
****

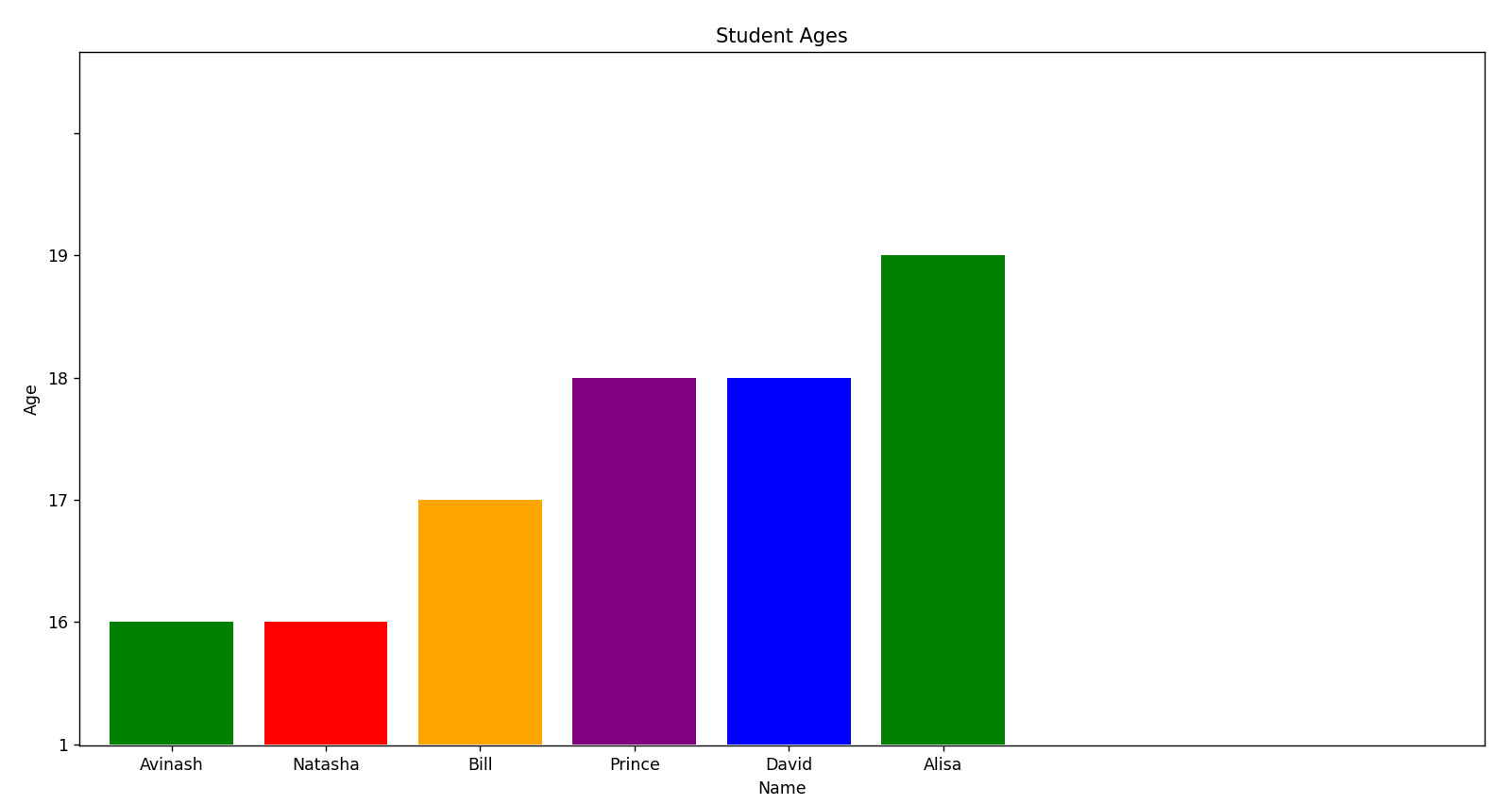
**EXIT DETAILS:**

****

**Chart’s**

**Chart b/w Staff Name & Salary: **

**Chart b/w Students Name & Paid Amount: **

**Chart b/w Students Name & Ages: **

**SQL QUERIES**

**Create database school;**

**use school;**

**CREATE TABLE students (Id VARCHAR(255),name VARCHAR(255), age VARCHAR(255), gender VARCHAR(255), room\_no VARCHAR(255));**

**Desc students;**

**use school;**

**create table Staff(Id varchar(50),post varchar(50), name varchar(50),salary varchar(50),phone varchar(50))**

**Desc Staff;**

**use school;**

**create table fee(SrNo varchar(50),Name varchar(50),Class varchar(50),Status varchar(50), Quarter varchar(50),PaidAmt varchar(50));**

**Desc fee;**

**USER INTERFACE CODE**

**mydb = mysql.connector.connect(**

**host="localhost",**

**user='root',**

**password='root')**

**print(mydb,"connected to server")**

**print("\n")**

**print("-" \*100)**

**print(" Welcome to School Management System")**

**def menu():**

**print("-" \* 100)**

**print("----------------Modules in School Management System ---------")**

**print("Module\_1: Student record Module ")**

**print("Module\_2: Staff record Module")**

**print("Module\_3: Fee record Module")**

**print("Module\_4: Exit from the system")**

**print("\_" \* 100)**

**# Get the user's choice:**

**# if option first:**

**def getchoice():**

**while True:**

**create\_database()**

**menu()**

**print("")**

**ch = input("Enter your choice: ")**

**if ch=='1':**

**print("PRESS (a): To Add New Student record PRESS (b): View Student details")**

**print("PRESS (c): To Update Student details PRESS (d):Delete Student details")**

**ch = input("Enter your choice: ")**

**create\_students()**

**if ch=='a':**

**add\_student()**

**input("Press ENTER KEY to continue.....")**

**print()**

**elif ch=='b':**

**view\_students()**

**input("Press ENTER KEY to continue.....")**

**print()**

**elif ch=='c':**

**update\_student()**

**input("Press ENTER KEY to continue.....")**

**print()**

**elif ch=='d':**

**delete\_student()**

**input("Press ENTER KEY to continue.....")**

**print()**

**## if option Second:**

**elif ch=='2':**

**print("PRESS (e) : Add New Staff record PRESS (f) : View Staff details | ")**

**print("PRESS (g) : Delete Staff details PRESS (h) : UPDATE Staff details ")**

**opp =input("Enter your choice: ")**

**create\_Staff()**

**if opp=='e':**

**add\_staff()**

**input("Press ENTER KEY to continue.....")**

**print()**

**elif opp=='f':**

**view\_staff()**

**input("Press ENTER KEY to continue.....")**

**print()**

**elif opp=='g':**

**update\_staff()**

**input("Press ENTER KEY to continue.....")**

**print()**

**elif opp=='h':**

**delete\_staff()**

**input("Press ENTER KEY to continue.....")**

**print()**

**### if option Third:**

**elif ch=='3':**

**print("PRESS (i): Add Fee deposit details PRESS (j): View Fee details ")**

**print("PRESS (k): Update Fee details PRESS (l): Delete Fee details")**

**opp = input("Enter your choice: ")**

**create\_fee()**

**if opp=='i':**

**fee()**

**input("Press ENTER KEY to continue.....")**

**print()**

**elif opp=='j':**

**view\_fee()**

**input("Press ENTER KEY to continue.....")**

**print()**

**elif opp=='k':**

**update\_fee()**

**input("Press ENTER KEY to continue.....")**

**print()**

**elif opp=='l':**

**delete\_fee()**

**input("Press ENTER KEY to continue.....")**

**print()**

**#### if option Fourth:**

**elif ch=='4':**

**print()**

**print("Exited !")**

**print("Succesfully,")**

**print("Thanks")**

**print("For")**

**print("Coming :-)")**

**print()**

**print()**

**print()**

**print()**

**break**

**ADD STUDENT RECORD**

**# CREATING A TABLE**

**def create\_students():**

**cursor = mydb.cursor()**

**cursor.execute('CREATE TABLE IF NOT EXISTS students (Id VARCHAR(255),name VARCHAR(255), age VARCHAR(255), gender VARCHAR(255), room\_no VARCHAR(255))')**

**# Define the function to add a new student**

**def add\_student():**

**Id=input("Enter Student SrNo: ")**

**name = input("Enter student Name: ")**

**age = input("Enter student DOB: ")**

**gender = input("Enter student gender: ")**

**room\_no = input("Enter student Class: ")**

**cursor = mydb.cursor()**

**# Inserting Values**

**sql = "INSERT INTO students (Id,name, age, gender, room\_no) VALUES (%s,%s, %s, %s, %s)"**

**val = (Id,name, age, gender, room\_no)**

**cursor.execute(sql, val)**

**mydb.commit()**

**print(cursor.rowcount, "record(s) inserted.")**

**DELETE STUDENT RECORD**

**# Define the function to delete student details**

**def delete\_student():**

**id = input("Enter student SrNo: ")**

**cursor = mydb.cursor()**

**sql = "DELETE FROM students WHERE Id = %s"**

**val = (id,)**

**cursor.execute(sql, val)**

**mydb.commit()**

**print(cursor.rowcount, "record(s) deleted.")**

**VIEW STUDENT RECORD**

**# Define the function to view student details**

**def view\_students():**

**cursor = mydb.cursor()**

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

**result = cursor.fetchall()**

**print("Press (f) to see in the form of DataFrame")**

**print("Press (i) to see the Saprate Index values")**

**print("Press (l) to see in the form of list")**

**ch = input("Enter your choice: ")**

**if ch=='i':**

**result\_list = [list(row) for row in result]**

**lst1 = [row[0] for row in result\_list]**

**print("lst1 is:", lst1)**

**lst2 = [row[1] for row in result\_list]**

**print('lst 2 is:', lst2)**

**lst3 = [row[2] for row in result\_list]**

**print('lst 3', lst3)**

**lst4 = [row[3] for row in result\_list]**

**print('lst 4 is: ', lst4)**

**lst5 = [row[4] for row in result\_list]**

**print('list 5', lst5)**

**elif ch=='l':**

**result\_list = [list(row) for row in result]**

**print(result\_list)**

**elif ch=='f':**

**result\_list = [list(row) for row in result]**

**lst1 = [row[0] for row in result\_list]**

**lst2 = [row[1] for row in result\_list]**

**lst3 = [row[2] for row in result\_list]**

**lst4 = [row[3] for row in result\_list]**

**lst5 = [row[4] for row in result\_list]**

**df = pd.DataFrame({'ID': lst1, 'Name': lst2, 'Age': lst3, 'Gender': lst4, 'Room\_No': lst5})**

**print(df.to\_markdown())**

**elif ch == 'g':**

**result\_list = [list(row) for row in result]**

**lst1 = [row[0] for row in result\_list]**

**lst2 = [row[1] for row in result\_list]**

**lst3 = [row[2] for row in result\_list]**

**lst4 = [row[3] for row in result\_list]**

**lst5 = [row[4] for row in result\_list]**

**df = pd.DataFrame({'ID': lst1, 'Name': lst2, 'Age': lst3, 'Gender': lst4, 'Class': lst5})**

**# Sort the dataframe by Age in ascending order**

**df\_sorted = df.sort\_values(by='Age')**

**# Get the sorted values for 'Name' and 'Age'**

**Name = df\_sorted['Name'].tolist()**

**Age = df\_sorted['Age'].tolist()**

**# Create the bar chart**

**plt.bar(Name, Age, color=['blue', 'green', 'red', 'orange', 'purple'])**

**plt.xlabel('Name')**

**plt.ylabel('Age')**

**plt.title('Student Ages')**

**# Set the y-axis limits and ticks**

**plt.ylim(0, 18) # Set the y-axis limits from 0 to 18**

**plt.yticks(range(19)) # Set the y-axis ticks from 0 to 18**

**plt.show()**

**UPDATE STUDENT RECORD**

**# Define the function to update student details**

**def update\_student():**

**id = input("Enter student SrNo: ")**

**name = input("Enter student Name: ")**

**age = input("Enter student DOB: ")**

**gender = input("Enter student gender: ")**

**room\_no = input("Enter student Class: ")**

**cursor = mydb.cursor()**

**sql\_up = "update students set name = %s, age = %s, gender = %s, room\_no = %s where id = %s"**

**val\_up = (name, age, gender, room\_no,id)**

**cursor.execute(sql\_up, val\_up)**

**mydb.commit()**

**print(cursor.rowcount, "record(s) updated.")**

**ADD STAFF RECORD**

**# CREATING A TABLE**

**def create\_Staff():**

**cursor = mydb.cursor()**

**cursor.execute('CREATE TABLE IF NOT EXISTS Staff(Id varchar(50) primary key,post varchar(50),name varchar(50),salary varchar(50),phone varchar(50))')**

**# Define the function to add a new staff**

**def add\_staff():**

**Id=input("Enter staff ID: ")**

**post=input("Enter staff Post: ")**

**name = input("Enter staff Name: ")**

**salary = input("Enter staff Salary: ")**

**phone = input("Enter staff Phone no: ")**

**cursor = mydb.cursor()**

**# Inserting Values**

**sqls = "INSERT INTO staff (Id,post,name,salary,phone) VALUES (%s,%s,%s, %s, %s)"**

**vals = (Id,post,name,salary,phone)**

**cursor.execute(sqls, vals)**

**mydb.commit()**

**print(cursor.rowcount, "record(s) inserted.")**

**UPDATE STAFF RECORD**

**# Define the function to update staff details**

**def update\_staff():**

**Id=input("Enter staff ID: ")**

**post=input("Enter staff Post: ")**

**name = input("Enter staff Name: ")**

**salary = input("Enter staff Salary: ")**

**phone = input("Enter staff Phone no: ")**

**cursor = mydb.cursor()**

**sql = "UPDATE staff SET post= %s, name = %s, salary = %s, phone = %s WHERE Id = %s"**

**val = (Id,post,name,salary, phone)**

**cursor.execute(sql, val) mydb.commit()**

**print(cursor.rowcount, "record(s) updated.")**

**DELETE STAFF RECORD**

**# Define the function to delete staff details**

**def delete\_staff():**

**Id = input("Enter staff ID: ")**

**cursor = mydb.cursor()**

**sql = "DELETE FROM staff WHERE Id = %s"**

**val = (Id,) cursor.execute(sql, val)**

**mydb.commit() print(cursor.rowcount, "record(s) deleted.")**

**VIEW STAFF RECORD**

**# Define the function to view student details**

**def view\_staff():**

**cursor = mydb.cursor()**

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

**result = cursor.fetchall()**

**print("Press (f) to see in the form of DataFrame")**

**print("Press (i) to see the Saprate Index values")**

**print("Press (l) to see in the form of list")**

**ch = input("Enter your choice: ")**

**if ch=='i':**

**result\_list = [list(row) for row in result]**

**lst1 = [row[0] for row in result\_list]**

**print("lst1 is:", lst1)**

**lst2 = [row[1] for row in result\_list]**

**print('lst 2 is:', lst2)**

**lst3 = [row[2] for row in result\_list]**

**print('lst 3', lst3)**

**lst4 = [row[3] for row in result\_list]**

**print('lst 4 is: ', lst4)**

**lst5 = [row[4] for row in result\_list]**

**print('list 5', lst5)**

**elif ch=='l':**

**result\_list = [list(row) for row in result]**

**print(result\_list)**

**elif ch=='f':**

**result\_list = [list(row) for row in result]**

**lst1 = [row[0] for row in result\_list]**

**lst2 = [row[1] for row in result\_list]**

**lst3 = [row[2] for row in result\_list]**

**lst4 = [row[3] for row in result\_list]**

**lst5 = [row[4] for row in result\_list]**

**df=pd.DataFrame({'ID':lst1,'POST':lst2,'NAME':lst3,'SALARY':lst4,'PHONE':lst5})**

**print(df.to\_markdown())**

**# Plotting pie chart**

**elif ch=='g':**

**result\_list = [list(row) for row in result]**

**lst1 = [row[0] for row in result\_list]**

**lst2 = [row[1] for row in result\_list]**

**lst3 = [row[2] for row in result\_list]**

**lst4 = [row[3] for row in result\_list]**

**lst5 = [row[4] for row in result\_list]**

**df=pd.DataFrame({'ID':lst1,'POST':lst2,'NAME':lst3,'SALARY':lst4,'PHONE':lst5})**

**plt.pie(df['SALARY'], labels=df['NAME'], autopct='%1.1f%%')**

**plt.title('Staff Salary Distribution')**

**plt.show()**

**ADD FEE RECORD**

**# CREATING A TABLE**

**def create\_fee():**

**cursor = mydb.cursor()**

**cursor.execute('CREATE TABLE IF NOT EXISTS fee(SrNo varchar(50) primary key,Name varchar(50),Class varchar(50),Status varchar(50),Quarter varchar(50),PaidAmt varchar(50))')**

**# Define the function to add Fee details**

**def fee():**

**student\_id =input("Enter Payer SrNo: ")**

**Name = input("Enter Payer Name: ")**

**Class = input("Enter Payer Class: ")**

**Status= input("Enter Status(Paid/Due) : ")**

**Quarter= input("Enter Quarter : ")**

**PaidAmt= input("Enter PaidAmt : ")**

**cursor = mydb.cursor()**

**# Inserting Values**

**msql = "INSERT INTO fee (student\_id,Name,Class,Status,Quarter,PaidAmt) VALUES (%s,%s, %s, %s,**

**%s,%s)"**

**valu = (student\_id,Name,Class,Status,Quarter,PaidAmt)**

**cursor.execute(msql, valu)**

**mydb.commit()**

**print(cursor.rowcount, "record(s) inserted.")**

**VIEW FEE RECORD**

**# Define the function to view Fee details**

**def view\_fee():**

**cursor = mydb.cursor()**

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

**result = cursor.fetchall()**

**result = cursor.fetchall()**

**print("Press (f) to see in the form of DataFrame")**

**print("Press (i) to see the Saprate Index values")**

**print("Press (l) to see in the form of list")**

**ch = input("Enter your choice: ")**

**if ch=='i':**

**result\_list = [list(row) for row in result]**

**lst1 = [row[0] for row in result\_list]**

**print("lst1 is:", lst1)**

**lst2 = [row[1] for row in result\_list]**

**print('lst 2 is:', lst2)**

**lst3 = [row[2] for row in result\_list]**

**print('lst 3', lst3)**

**lst4 = [row[3] for row in result\_list]**

**print('lst 4 is: ', lst4)**

**lst5 = [row[4] for row in result\_list]**

**print('list 5', lst5)**

**lst6 = [row[5] for row in result\_list]**

**print('list 6', lst6)**

**lst7 = [row[6] for row in result\_list]**

**print('lst 7 is: ',lst7)**

**elif ch=='l':**

**result\_list = [list(row) for row in result]**

**print(result\_list)**

**elif ch=='f':**

**result\_list = [list(row) for row in result]**

**lst1 = [row[0] for row in result\_list]**

**lst2 = [row[1] for row in result\_list]**

**lst3 = [row[2] for row in result\_list]**

**lst4 = [row[3] for row in result\_list]**

**lst5 = [row[4] for row in result\_list]**

**lst6 = [row[5] for row in result\_list]**

**lst7 = [row[6] for row in result\_list]**

**df=pd.DataFrame({'SrNo':lst1,'Name':lst2,'Class':lst3,'':lst4,'Status':lst5,**

**'Quarter':lst6,'PaidAmt':lst7})**

**print(df.to\_markdown())**

**# Plotting line chart**

**elif ch == 'g':**

**result\_list = [list(row) for row in result]**

**lst1 = [row[0] for row in result\_list]**

**lst2 = [row[1] for row in result\_list]**

**lst3 = [row[2] for row in result\_list]**

**lst4 = [row[3] for row in result\_list]**

**lst5 = [row[4] for row in result\_list]**

**lst6 = [row[5] for row in result\_list]**

**df=pd.DataFrame({'SrNo':lst1,'Name':lst2,'Class':lst3,'Status':lst4,'Quarter':lst5,'PaidAmt':lst6 })**

**# Sort the DataFrame by Quarter in ascending order**

**df.sort\_values(by='PaidAmt')**

**Name = df['Name']**

**PaidAmt = df['PaidAmt']**

**plt.plot(Name, PaidAmt)**

**plt.xlabel('Name')**

**plt.ylabel('Paid Amount')**

**plt.title('Fee Payment Over Quarters')**

**plt.show()**

**DELETE FEE RECORD**

**# Define the function to delete Fee details**

**def delete\_fee():**

**SrNo = input("Enter student SrNo: ")**

**cursor = mydb.cursor()**

**sqle = "DELETE FROM fee WHERE SrNo = %s"**

**vale = (SrNo,)**

**cursor.execute(sqle, vale)**

**mydb.commit()**

**print(cursor.rowcount, "record(s) deleted.")**

**UPDATE FEE RECORD**

**# Define the function to update Fee details**

**def update\_fee():**

**SrNo = input("Enter student SrNo: ")**

**Name = input("Enter student Name: ")**

**Class = input("Enter student Class: ")**

**Status = input("Enter student Status(Paid/Due): ")**

**Quarter = input("Enter student Quarter: ")**

**PaidAmt = input("Enter student PaidAmount: ")**

**cursor = mydb.cursor()**

**sqlx = "UPDATE fee SET Name = %s, Class = %s, Status = %s, Quarter = %s,PaidAmt = %s WHERE SrNo = %s" valx = (Name,Class,Status,Quarter,PaidAmt,SrNo)**

**cursor.execute(sqlx, valx) print(cursor.rowcount, "record(s) deleted.")**

**TESTING**

**Software testing is an empirical investigation conducted to provide skateholders with information**

**About the quality of the product or service under test, with respect to the context in which it is**

**Intended to operate. Software testing also provides an, independent view of the software to allow**

**The business to appreciate and understand the risk at implementation of the software.**

**Test techniques include, but are not limited to the process of executing a programme or**

**Application with the intent of finding software bugbugs.**

**It can also be stated as the process of validating and verifying that a software programme/**

**Application / product meets the business and technical requirements that guided the**

**its design and development, so that it works a expected and can be implemented with**

**the same characteristics. Software testing, depending on the testing method employed,**

**can be implemented at anytime in the development process however the most test**

**effort is employed after the requirements have been defined and coding process has**

**been completed.**

**BIBLOGRAPHY**

* **Google for Research**
* **www.wikipedia.com**
* **www.geeksforgeeks.org**