Final Project

(EMPLOYEE MANAGEMENT SYSTEM)

Source Code:

```
from tkinter import *
from tkinter.messagebox import *
from tkinter.scrolledtext import *
from sqlite3 import *
import requests, json
import matplotlib.pyplot as plt


root = Tk()
root.title("Employee Management System")
root.geometry("950x900+1+1")
f = ("Century", 30)
ff = ("Arial", 40, "bold")

def f1():
	add.deiconify()
	root.withdraw()

def f2():
	view.deiconify()
	root.withdraw()
	vw_Data.delete(1.0, END)
	con = None
```

```
    try:
        con = connect("eems.db")
        cursor = con.cursor()
        sql = "select * from employee"
        cursor.execute(sql)
        data = cursor.fetchall()
        info = ""
        for d in data:
            info += f"Id: {str(d[0])}\nName: {str(d[1])}\nSalary: {str(d[2])}\n\n"
        vw_Data.insert(INSERT, info)
    except Exception as e:
        showerror ("ISSUE", e)
    finally:
        if con is not None:
            con.close()

def f3():
    update.deiconify()
    root.withdraw()

def f4():
    delete.deiconify()
    root.withdraw()

def f5():
    root.deiconify()
    add.withdraw()
```

```
def f6():
        root.deiconify()
        view.withdraw()

def f7():
        root.deiconify()
        update.withdraw()

def f8():
        root.deiconify()
        delete.withdraw()

def f9():
        employee_id = dt_idEntry.get()
        if not employee_id:
        showerror("ERROR", "Please enter the ID to delete.")
        return
        if not employee_id.isdigit():
        showerror("ERROR", "ID should contain only numbers.")
        return
        employee_id = int(employee_id)
        con = None
        try:
                con = connect("eems.db")
                cursor = con.cursor()
                cursor.execute(f"SELECT * FROM employee WHERE id =
{employee_id}")
```

```
            data = cursor.fetchone()
            if data:
                cursor.execute(f"DELETE FROM employee WHERE id = {employee_id}")
                con.commit()
                showinfo("SUCCESS", f"Employee with ID {employee_id} deleted successfully!")
            else:
                showerror("ERROR", f"No employee found with ID {employee_id}")
        except Exception as e:
            showerror("ISSUE", e)

        finally:
            if con is not None:
                con.close()

        dt_idEntry.delete(0, END)

def f10():
    employee_id = up_idEntry.get()
    new_name = up_nameEntry.get()
    new_salary = up_salaryEntry.get()

    if not employee_id:
        showerror("ERROR", "Please enter the ID.")
        return
```

```
    if not employee_id.isdigit():
        showerror("ERROR", "ID should contain only numbers.")
        return

    employee_id = int(employee_id)

    con = None
    try:
        con = connect("eems.db")
        cursor = con.cursor()
        cursor.execute(f"SELECT * FROM employee WHERE id = {employee_id}")
        data = cursor.fetchone()

        if data:
            update_query = "UPDATE employee SET "
            if new_name:
                update_query += f"name = '{new_name}', "
            if new_salary:
                update_query += f"salary = {int(new_salary)}, "
            update_query = update_query.rstrip(", ")

            update_query += f" WHERE id = {employee_id}"
            cursor.execute(update_query)
            con.commit()
            showinfo("SUCCESS", f"Employee with ID {employee_id} updated
successfully!")
        else:
            showerror("ERROR", f"No employee found with ID {employee_id}")
```

```
    except Exception as e:
        showerror("ISSUE", e)

    finally:
        if con is not None:
            con.close()

    up_idEntry.delete(0, END)
    up_nameEntry.delete(0, END)
    up_salaryEntry.delete(0, END)

def f11():
    top_5_data = get_top_5_employees()
    names = [row[0] for row in top_5_data]
    salaries = [row[1] for row in top_5_data]

    plt.figure(figsize=(8, 6))
    plt.bar(names, salaries, color='black')
    plt.xlabel("Employee Name")
    plt.ylabel("Salary")
    plt.title("Top 5 Employees with Highest Salaries")
    plt.xticks(rotation=45, ha='right')
    plt.tight_layout()
    plt.show()

def get_top_5_employees():
```

```
    con = None
    try:
        con = connect("eems.db")
        cursor = con.cursor()
        cursor.execute("SELECT name, salary FROM employee ORDER BY salary
DESC LIMIT 5")
        top_5_data = cursor.fetchall()
        return top_5_data
    except Exception as e:
        showerror("ISSUE", e)
    finally:
        if con is not None:
            con.close()


def save():
        id_text = add_idEntry.get()
        name_text = add_nameEntry.get()
        salary_text = add_salaryEntry.get()
        if not id_text:
                showerror("ERROR", "It should not be Empty")
        elif not id_text.isdigit():
                showerror("ERROR", "It should contain only Numbers")
        elif id_text == "":
                showerror("ERROR", "It should not be Empty")
        elif name_text == "":
                showerror("ERROR", "Name should not be Blank")
        elif not name_text:
                showerror("ERROR", "It should not be Empty")
```

```
elif name_text.isdigit():
        showerror("ERROR", "Name should contain Alphabet")
elif len(name_text) < 2:
        showerror("ERROR", "Name should contain more then 2 letters")
elif not salary_text:
        showerror("ERROR", "Salary should not be Empty")
elif salary_text == "":
        showerror("ERROR", "Salary should not be Empty")
elif not salary_text.isdigit():
        showerror("ERROR", "Salary should contain Integer")
else:
        con = None
        try:
                con = connect("eems.db")
                cursor = con.cursor()
                sql = "insert into employee values('%d', '%s', '%d')"
                id = int(add_idEntry.get())
                name =          add_nameEntry.get()
                salary = int(add_salaryEntry.get())
                cursor.execute(sql % (id, name, salary))
                con.commit()
                showinfo("SUCCESS", "Record inserted Successfully!!!")
                add_idEntry.delete(0, END)
                add_nameEntry.delete(0, END)
                add_salaryEntry.delete(0, END)
                add_idEntry.focus()
        except Exception as e:
```

```
            con.rollback()
            showerror("ISSUE", e)
        finally:
            if con is not None:
                con.close()

TitleLabel = Label(root, text = "EMPLOYEE MANAGEMENT SYSTEM", font = ff)
TitleLabel.place(x = 1, y = 1)

btn1 = Button(root, text = "Add Employee", font = f, command = f1)
btn1.place(x = 280, y = 100)

btn2 = Button(root, text = "View Employee", font = f, command = f2)
btn2.place(x = 280, y = 200)

btn3  = Button(root, text = "Update Employee", font = f, command = f3)
btn3.place(x = 260, y = 300)

btn4 = Button(root, text = "Delete Employee", font = f, command = f4)
btn4.place(x = 260, y = 400)

btn5 = Button(root, text = "Charts", font = f, command = f11)
btn5.place(x = 350, y = 500)

def f11():
    top_5_data = get_top_5_employees()
    names = [row[0] for row in top_5_data]
```

```
    salaries = [row[1] for row in top_5_data]

    plt.figure(figsize=(8, 6))
    plt.bar(names, salaries, color='lightblue')
    plt.xlabel("Employee Name")
    plt.ylabel("Salary")
    plt.title("Top 5 Employees with Highest Salaries")
    plt.xticks(rotation=45, ha='right')
    plt.tight_layout()
    plt.show()


add = Toplevel(root)
add.title("ADD EMPLOYEE")
add.geometry("950x900+1+1")


add_titleLabel = Label(add, text = "EMPLOYEE MANAGEMENT SYSTEM", font =
ff)
add_titleLabel.place(x = 1, y = 1)

add_idLabel = Label(add, text = "Enter Id: ", font = f)
add_idLabel.place(x = 380, y = 80)

add_idEntry = Entry(add, font = f)
add_idEntry.place(x = 250, y = 140)

add_nameLabel = Label(add, text = "Enter Name: ", font = f)
```

```
add_nameLabel.place(x = 380, y = 200)

add_nameEntry = Entry(add, font = f)
add_nameEntry.place(x = 250, y = 260)

add_salaryLabel = Label(add, text = "Enter Salary: ", font = f)
add_salaryLabel.place(x = 380, y = 320)

add_salaryEntry = Entry(add, font = f)
add_salaryEntry.place(x = 250, y = 380)

add_sbtn = Button(add, text = "Save", font = f, command = save)
add_sbtn.place(x = 390, y = 450)

add_bbtn = Button(add, text = "Back", font = f, command = f5)
add_bbtn.place(x = 390, y = 550)

add.withdraw()

view = Toplevel(root)
view.title("VIEW EMPLOYEE")
view.geometry("950x900+1+1")

vw_Data = ScrolledText(view, width = 40, height = 10, font = f)
vw_Data.place(x = 10, y = 30)
vw_BackBtn = Button(view, text = "BACK", font = f, command = f6)
vw_BackBtn.place(x = 390, y = 550)
```

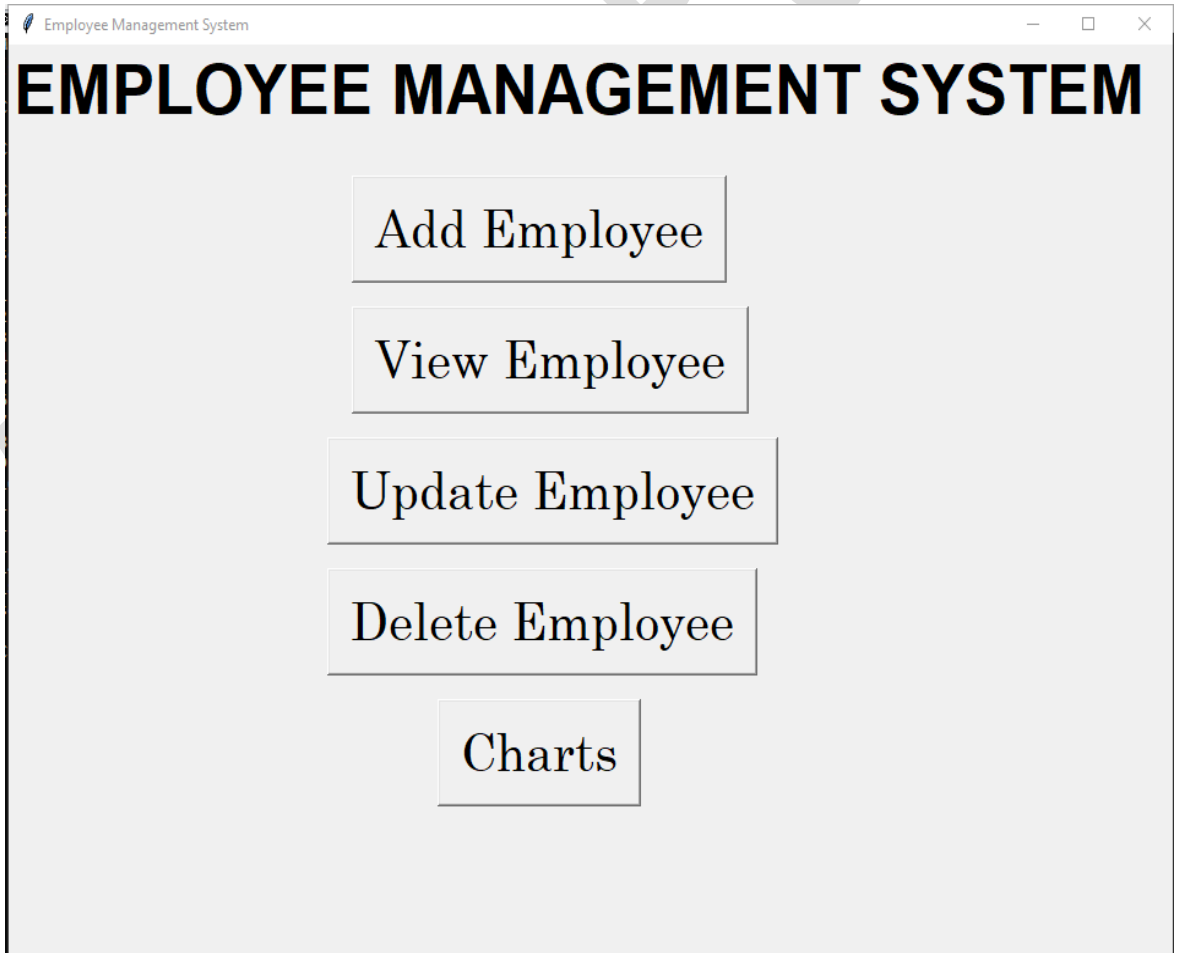
```
view.withdraw()

update = Toplevel(root)
update.title("UPDATE EMPLOYEE")
update.geometry("950x900+1+1")


up_titleLabel = Label(update, text = "EMPLOYEE MANAGEMENT SYSTEM", font
= ff)
up_titleLabel.place(x = 1, y = 1)

up_idLabel = Label(update, text = "Enter Id: ", font = f)
up_idLabel.place(x = 380, y = 80)

up_idEntry = Entry(update, font = f)
up_idEntry.place(x = 250, y = 140)

up_nameLabel = Label(update, text = "Enter Name: ", font = f)
up_nameLabel.place(x = 380, y = 200)

up_nameEntry = Entry(update, font = f)
up_nameEntry.place(x = 250, y = 260)

up_salaryLabel = Label(update, text = "Enter Salary: ", font = f)
up_salaryLabel.place(x = 380, y = 320)

up_salaryEntry = Entry(update, font = f)
```

```
up_salaryEntry.place(x = 250, y = 380)

up_sbtn = Button(update, text = "Update", font = f, command = f10)
up_sbtn.place(x = 390, y = 450)

up_bbtn = Button(update, text = "Back", font = f, command = f7)
up_bbtn.place(x = 390, y = 550)

update.withdraw()


delete = Toplevel(root)
delete.title("DELETE EMPLOYEE")
delete.geometry("950x900+1+1")


dt_titleLabel = Label(delete, text = "EMPLOYEE MANAGEMENT SYSTEM", font
= ff)
dt_titleLabel.place(x = 1, y = 1)

dt_idLabel = Label(delete, text = "Enter Id: ", font = f)
dt_idLabel.place(x = 380, y = 120)

dt_idEntry = Entry(delete, font = f)
dt_idEntry.place(x = 250, y = 180)

dt_sbtn = Button(delete, text = "Delete", font = f, command = f9)
dt_sbtn.place(x = 390, y = 350)
```

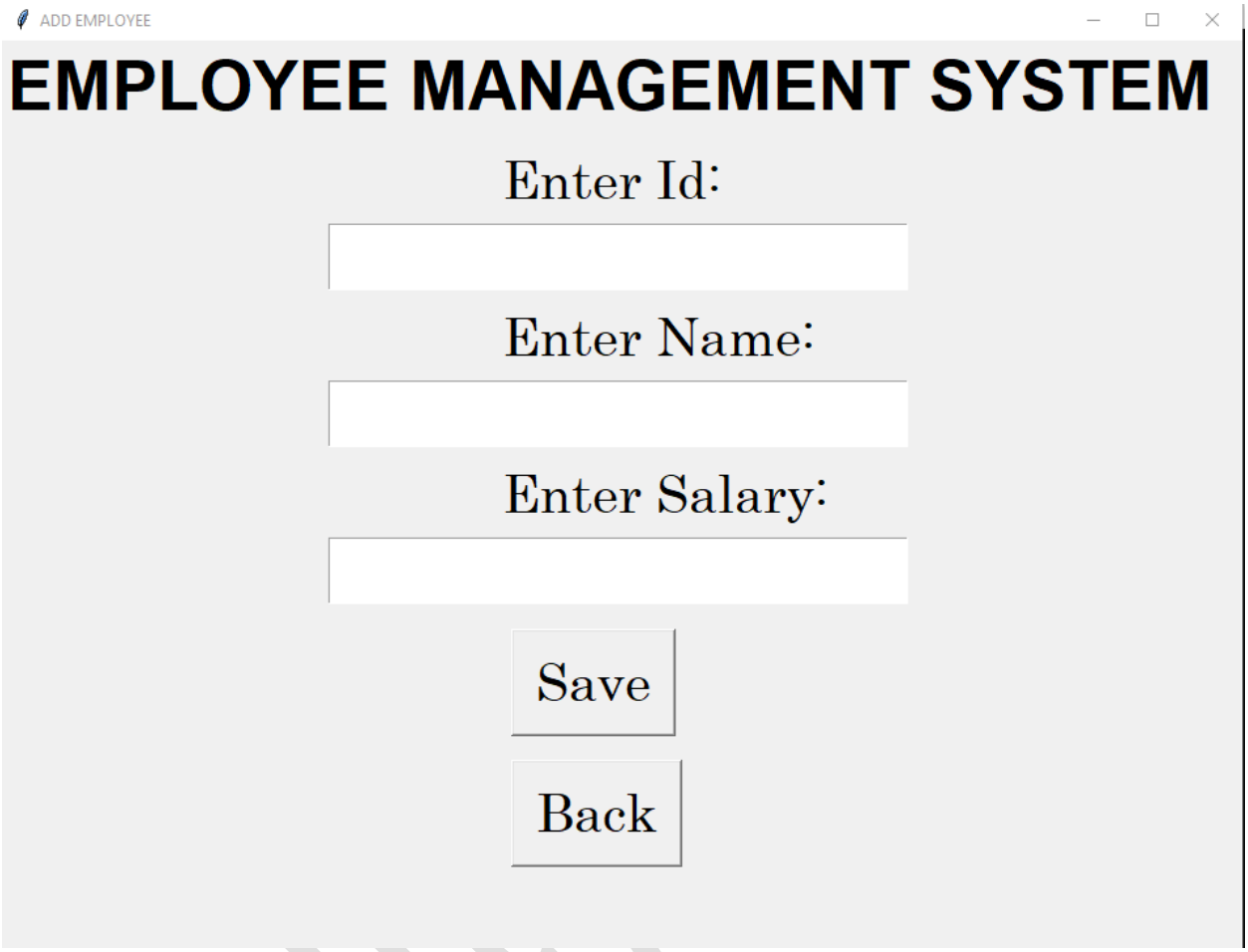
```
dt_bbtn = Button(delete, text = "Back", font = f, command = f8)
dt_bbtn.place(x = 390, y = 450)

delete.withdraw()


root.mainloop()
```

Output:

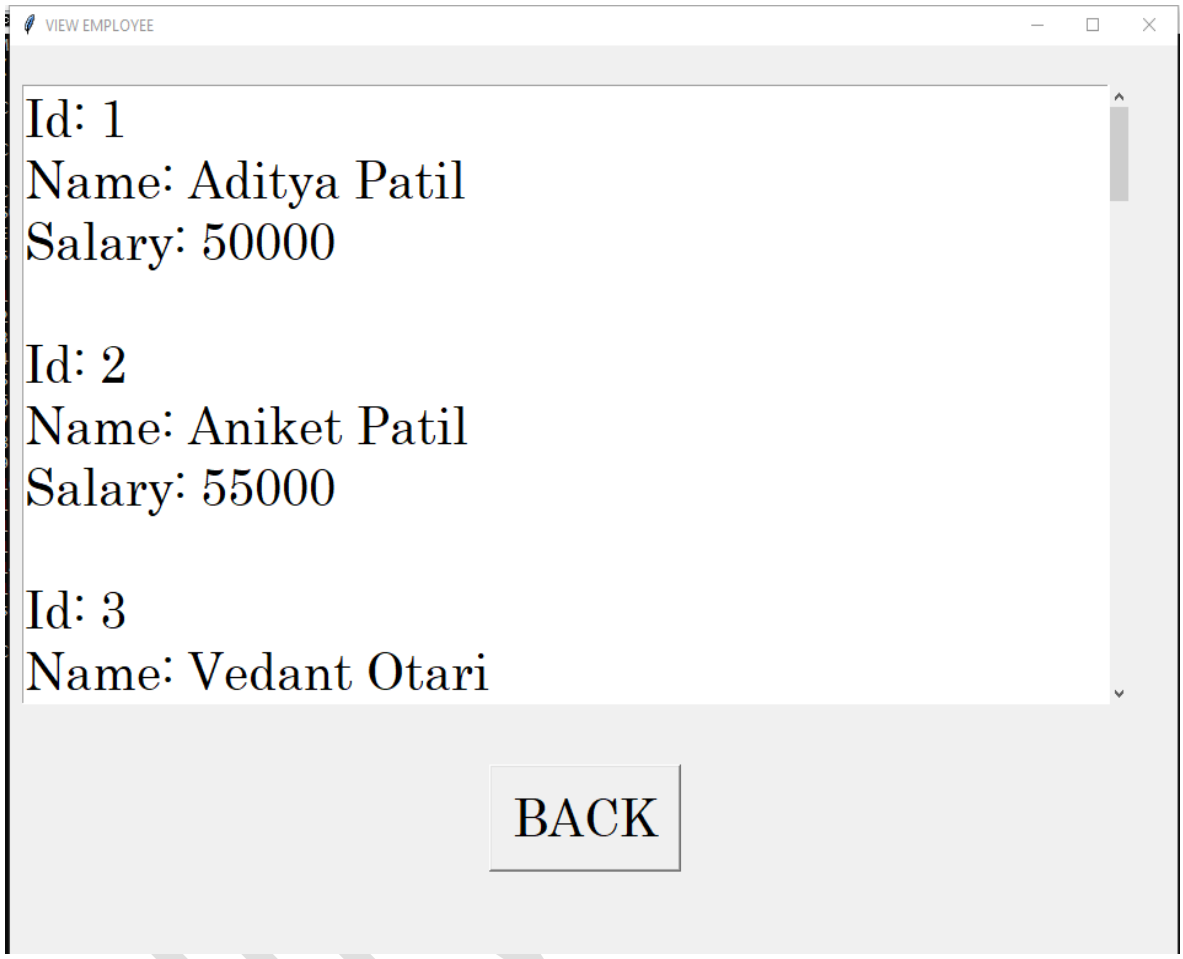
1) Home page:



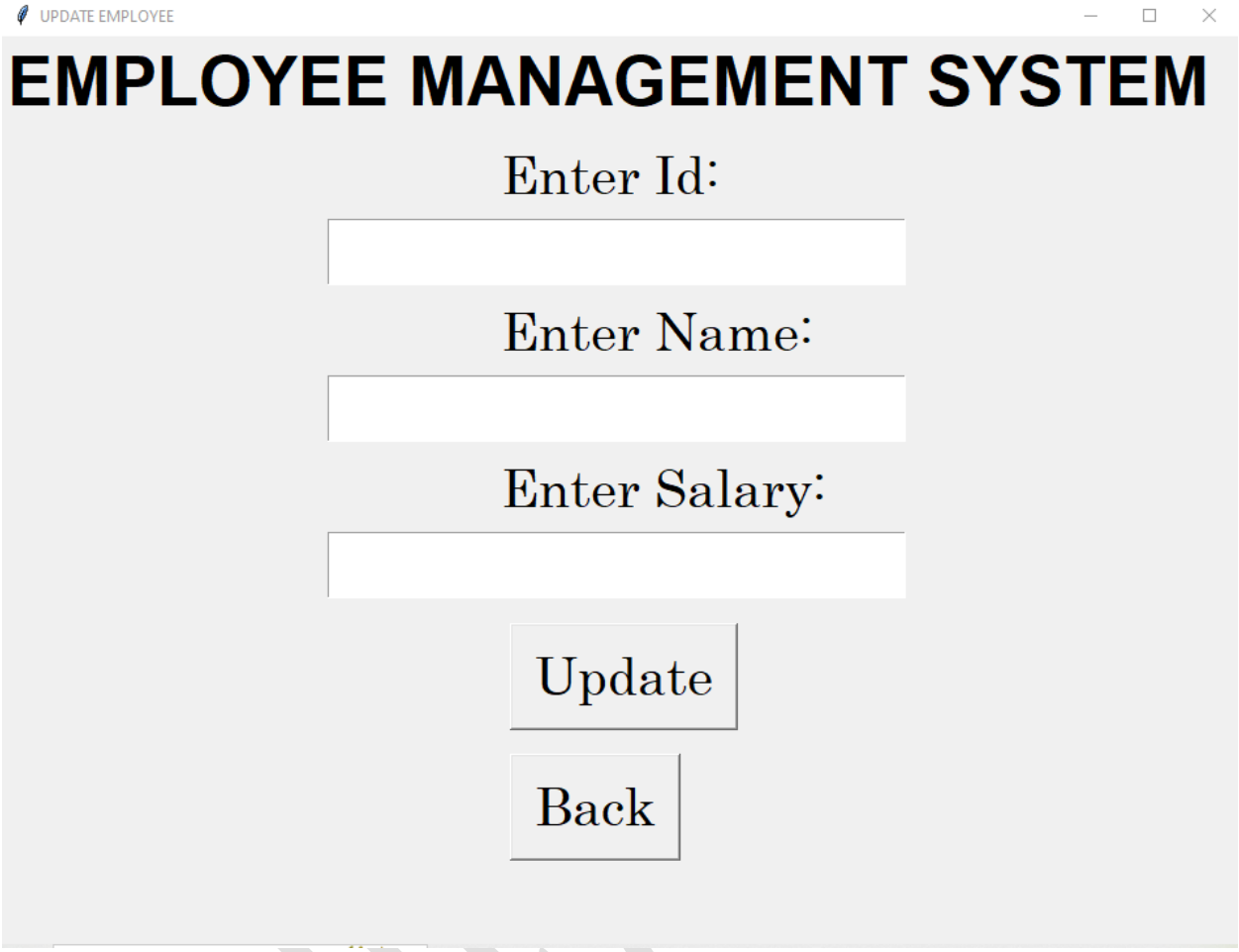
2) Add Employee Page:



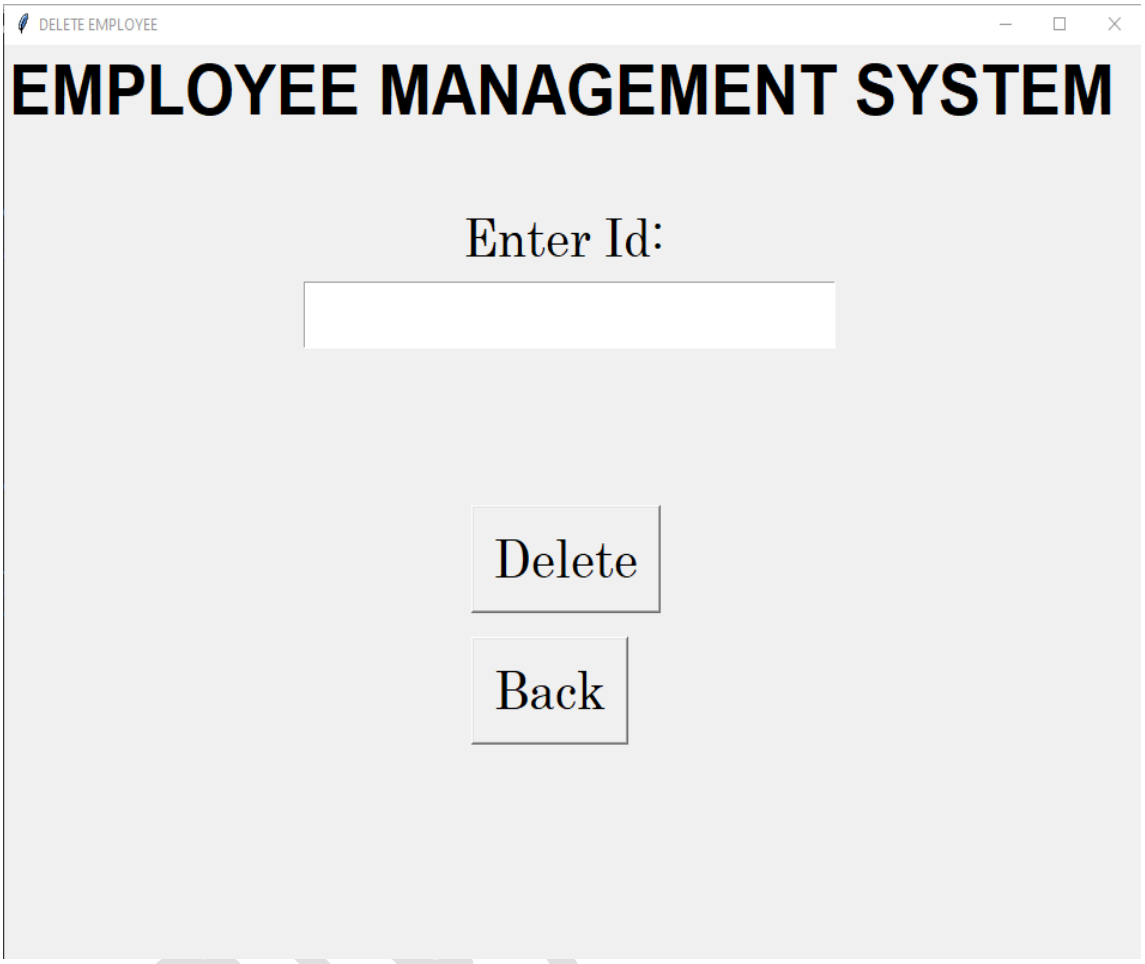
3) View Employee Page:



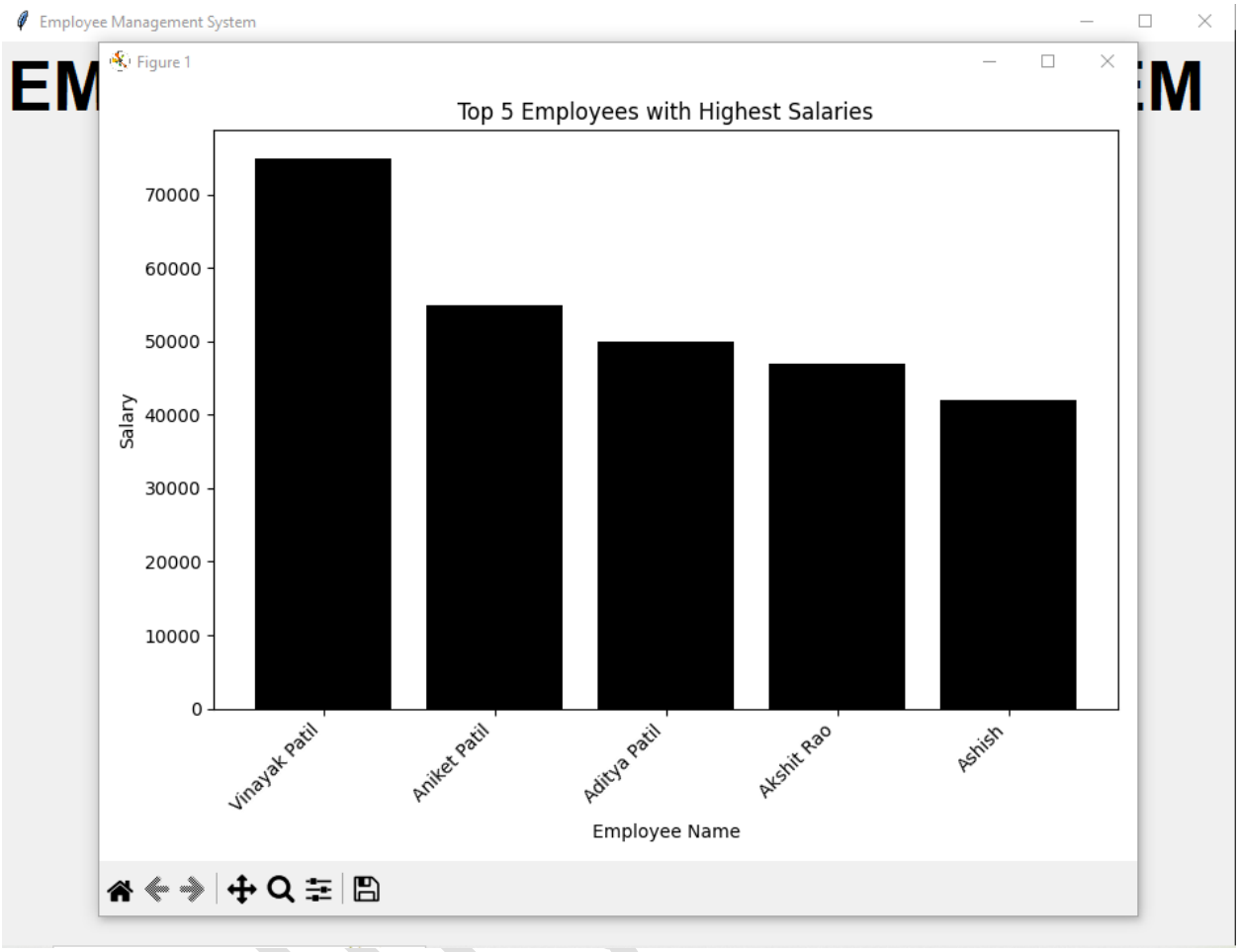
4) Update Employee Page:



5) Delete Employee Page:



6) Charts:


Employee Management System
Figure 1
Top 5 Employees with Highest Salaries
Salary
70000
60000
50000
40000
30000
20000
10000
0
Vinayak Patil
Aniket Patil
Aditya Patil
Akshit Rao
Ashish
Employee Name

7) Cmd sqlite3:

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.3930]
(c) Microsoft Corporation. All rights reserved.

C:\Users\dell\Desktop\Python\Project>notepad project_04.py

C:\Users\dell\Desktop\Python\Project>python project_04.py

C:\Users\dell\Desktop\Python\Project>sqlite3 eems.db
SQLite version 3.44.2 2023-11-24 11:41:44 (UTF-16 console I/O)
Enter ".help" for usage hints.
sqlite> select * from employee
   ...> ;
1|Aditya Patil|50000
2|Aniket Patil|55000
3|Vedant Otari|37000
4|Sameem Mandal|5000
5|Akshit Rao|47000
6|Sarvesh Pandit|20000
7|Ganesh Chavan|32000
8|Ashish|42000
9|Dipikesh |28000
10|Dhiraj|28000
11|Suraj|27000
12|Ajinkya|29500
13|Aditya|30500
14|Kartik|14000
18|Vinayak Patil|75000
sqlite> .exit

C:\Users\dell\Desktop\Python\Project>python project_04.py

C:\Users\dell\Desktop\Python\Project>
```