18CSC207J
ADVANCED PROGRAMMING PRACTICE

Project title: Employee Management System

Submitted by:

1)Bharathwaj.M(RA2011026020065)

2)Prawin R.P(RA2011026020107)

Aim:

Employee Management System created to record all data of the employees. This system provides the organization (or) the employer to manage and track their employee data from different location.

Design:

The **Employee Management System** is a python project. this system offers the user/employer to monitor and update tasks depending upon the situation of the employee job status and also saves money & time through automation.

Database-driven Employee Management System in Python that will store the information in the MySQL Database. The script will contain the following operations:

- Add Employee
- Remove Employee
- Promote Employee
- Display Employees

The idea is that we perform different changes in our Employee Record by using different functions for example the Add Employee will insert a new row in our Employee, also, we will create a Remove Employee Function which will delete the record of any particular existing employee in our employee table. This System works on the concepts of taking the information from the database making required changes in the fetched data and applying the changes in the record which we will see in our Promote Employee System. We can also have the information about all the existing employees by using the Display Employee function. The main advantage of connecting our program to the database is that the information becomes lossless even after closing our program a number of times.

Procedure:

To implement this project, we use basic concepts of python, tkinter and SQLite.

- **Tkinter** is standard GUI library and it is one of the easiest ways to build a GUI Application.
- **SQLite** is a server-less database and is self-contained. This is also referred to as an embedded database which means the DB engine runs as a part of the app.

To install the required modules run pip installer command on the command line:

1. Pip install tkinter
2. Pip install pysqlite3 (**or**) Pip install pysqlite.

Attributes the employer needs for monitoring their job status:

- Name, age, gender, contact, address. (personal)
- Id, doj (date of joining), email address. (professional)

Source code:

main.py

```
from tkinter import *
from tkinter import ttk
from tkinter import messagebox
from db import Database

db = Database("Employee.db")
root = Tk()
root.title("Employee Management System")
root.geometry("1920x1080+0+0")
root.config(bg="#2c3e50")
root.state("zoomed")

name = StringVar()
age =  StringVar()
doj = StringVar()
gender = StringVar()
email = StringVar()
contact = StringVar()

# Entries Frame
entries_frame = Frame(root, bg="#535c68")
entries_frame.pack(side=TOP, fill=X)
title = Label(entries_frame, text="Employee Management System", font=("Calibri", 18, "bold"), bg="#535c68", fg="white")
title.grid(row=0, columnspan=2, padx=10, pady=20, sticky="w")

lblName = Label(entries_frame, text="Name", font=("Calibri", 16), bg="#535c68", fg="white")
lblName.grid(row=1, column=0, padx=10, pady=10, sticky="w")
```

```
txtName = Entry(entries_frame, textvariable=name, font=("Calibri", 16), width=30)
txtName.grid(row=1, column=1, padx=10, pady=10, sticky="w")

lblAge = Label(entries_frame, text="Age", font=("Calibri", 16), bg="#535c68", fg="white")
lblAge.grid(row=1, column=2, padx=10, pady=10, sticky="w")
txtAge = Entry(entries_frame, textvariable=age, font=("Calibri", 16), width=30)
txtAge.grid(row=1, column=3, padx=10, pady=10, sticky="w")

lbldoj = Label(entries_frame, text="D.O.J", font=("Calibri", 16), bg="#535c68", fg="white")
lbldoj.grid(row=2, column=0, padx=10, pady=10, sticky="w")
txtDoj = Entry(entries_frame, textvariable=doj, font=("Calibri", 16), width=30)
txtDoj.grid(row=2, column=1, padx=10, pady=10, sticky="w")

lblEmail = Label(entries_frame, text="Email", font=("Calibri", 16), bg="#535c68", fg="white")
lblEmail.grid(row=2, column=2, padx=10, pady=10, sticky="w")
txtEmail = Entry(entries_frame, textvariable=email, font=("Calibri", 16), width=30)
txtEmail.grid(row=2, column=3, padx=10, pady=10, sticky="w")

lblGender = Label(entries_frame, text="Gender", font=("Calibri", 16), bg="#535c68", fg="white")
lblGender.grid(row=3, column=0, padx=10, pady=10, sticky="w")
comboGender = ttk.Combobox(entries_frame, font=("Calibri", 16), width=28, textvariable=gender, state="readonly")
comboGender['values'] = ("Male", "Female")
comboGender.grid(row=3, column=1, padx=10, sticky="w")

lblContact = Label(entries_frame, text="Contact No", font=("Calibri", 16), bg="#535c68", fg="white")
lblContact.grid(row=3, column=2, padx=10, pady=10, sticky="w")
txtContact = Entry(entries_frame, textvariable=contact, font=("Calibri", 16), width=30)
```

```
txtContact.grid(row=3, column=3, padx=10, sticky="w")

lblAddress = Label(entries_frame, text="Address", font=("Calibri", 16), bg="#535c68", fg="white")
lblAddress.grid(row=4, column=0, padx=10, pady=10, sticky="w")

txtAddress = Text(entries_frame, width=85, height=5, font=("Calibri", 16))
txtAddress.grid(row=5, column=0, columnspan=4, padx=10, sticky="w")

def getData(event):
    selected_row = tv.focus()
    data = tv.item(selected_row)
    global row
    row = data["values"]
    #print(row)
    name.set(row[1])
    age.set(row[2])
    doj.set(row[3])
    email.set(row[4])
    gender.set(row[5])
    contact.set(row[6])
    txtAddress.delete(1.0, END)
    txtAddress.insert(END, row[7])

def displayAll():
    tv.delete(*tv.get_children())
    for row in db.fetch():
        tv.insert("", END, values=row)
```

```
def add_employee():
    if txtName.get() == "" or txtAge.get() == "" or txtDoj.get() == "" or txtEmail.get() == ""
or comboGender.get() == "" or txtContact.get() == "" or txtAddress.get(
            1.0, END) == "":
        messagebox.showerror("Erorr in Input", "Please Fill All the Details")
        return
    db.insert(txtName.get(),txtAge.get(), txtDoj.get() , txtEmail.get() ,comboGender.get(),
txtContact.get(), txtAddress.get(
            1.0, END))
    messagebox.showinfo("Success", "Record Inserted")
    clearAll()
    displayAll()



def update_employee():
    if txtName.get() == "" or txtAge.get() == "" or txtDoj.get() == "" or txtEmail.get() == ""
or comboGender.get() == "" or txtContact.get() == "" or txtAddress.get(
            1.0, END) == "":
        messagebox.showerror("Erorr in Input", "Please Fill All the Details")
        return
    db.update(row[0],txtName.get(), txtAge.get(), txtDoj.get(), txtEmail.get(),
comboGender.get(), txtContact.get(),
              txtAddress.get(
                  1.0, END))
    messagebox.showinfo("Success", "Record Update")
    clearAll()
    displayAll()
```

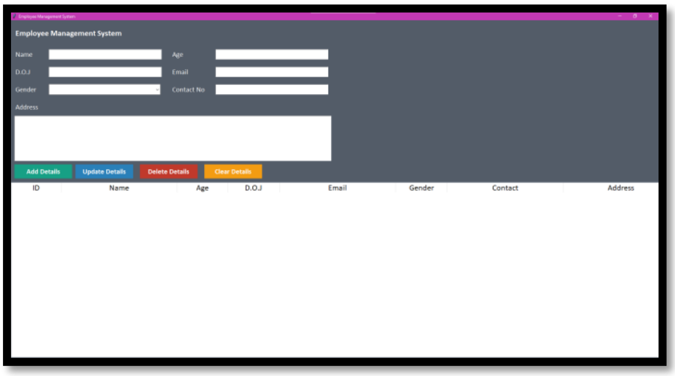
```
def delete_employee():
    db.remove(row[0])
    clearAll()
    displayAll()



def clearAll():
    name.set("")
    age.set("")
    doj.set("")
    gender.set("")
    email.set("")
    contact.set("")
    txtAddress.delete(1.0, END)



btn_frame = Frame(entries_frame, bg="#535c68")
btn_frame.grid(row=6, column=0, columnspan=4, padx=10, pady=10, sticky="w")
btnAdd = Button(btn_frame, command=add_employee, text="Add Details", width=15, font=("Calibri", 16, "bold"), fg="white",
                bg="#16a085", bd=0).grid(row=0, column=0)
btnEdit = Button(btn_frame, command=update_employee, text="Update Details", width=15, font=("Calibri", 16, "bold"),
                 fg="white", bg="#2980b9",
                 bd=0).grid(row=0, column=1, padx=10)
btnDelete = Button(btn_frame, command=delete_employee, text="Delete Details", width=15, font=("Calibri", 16, "bold"),
                   fg="white", bg="#c0392b",
                   bd=0).grid(row=0, column=2, padx=10)
btnClear = Button(btn_frame, command=clearAll, text="Clear Details", width=15, font=("Calibri", 16, "bold"), fg="white",
                bg="#f39c12",
                bd=0).grid(row=0, column=3, padx=10)

# Table Frame
tree_frame = Frame(root, bg="#ecf0f1")
tree_frame.place(x=0, y=480, width=1980, height=520)
style = ttk.Style()
style.configure("mystyle.Treeview", font=('Calibri', 18),
               rowheight=50)  # Modify the font of the body
style.configure("mystyle.Treeview.Heading", font=('Calibri', 18))  # Modify the font of the headings
tv = ttk.Treeview(tree_frame, columns=(1, 2, 3, 4, 5, 6, 7, 8), style="mystyle.Treeview")
tv.heading("1", text="ID")
tv.column("1", width=5)
tv.heading("2", text="Name")
tv.heading("3", text="Age")
tv.column("3", width=5)
tv.heading("4", text="D.O.B")
tv.column("4", width=10)
tv.heading("5", text="Email")
tv.heading("6", text="Gender")
tv.column("6", width=10)
tv.heading("7", text="Contact")
tv.heading("8", text="Address")
tv['show'] = 'headings'
tv.bind("<ButtonRelease-1>", getData)
tv.pack(fill=X)

displayAll()
root.mainloop()
```

db.py

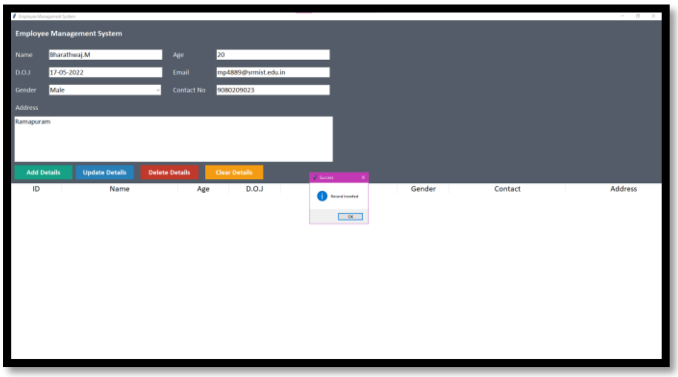
```
import sqlite3


class Database:
    def__init__(self, db):
        self.con = sqlite3.connect(db)
        self.cur = self.con.cursor()
        sql = """
        CREATE TABLE IF NOT EXISTS employees(
            id Integer Primary Key,
            name text,
            age  text,
            doj text,
            email text,
            gender text,
            contact text,
            address text
        )
        """
        self.cur.execute(sql)
        self.con.commit()

    # Insert Function
    def insert(self, name, age, doj, email, gender, contact, address):
        self.cur.execute("insert into employees values (NULL,?,?,?,?,?,?,?)",
                         (name, age, doj, email, gender, contact, address))
        self.con.commit()
    # Fetch All Data from DB
    def fetch(self):
        self.cur.execute("SELECT * from employees")
        rows = self.cur.fetchall()
        # print(rows)
        return rows

    # Delete a Record in DB
    def remove(self, id):
        self.cur.execute("delete from employees where id=?", (id,))
        self.con.commit()

    # Update a Record in DB
    def update(self, id, name, age, doj, email, gender, contact, address):
        self.cur.execute(
            "update employees set name=?, age=?, doj=?, email=?, gender=?, contact=?, address=? where id=?",
            (name, age, doj, email, gender, contact, address, id))
        self.con.commit()
```

Screenshots:

Record Displayed:

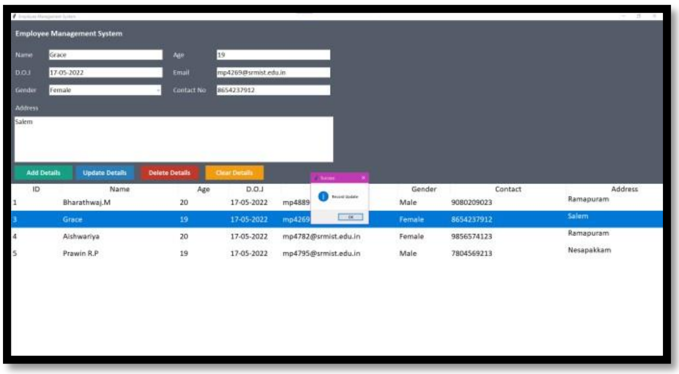


Record Inserted:



Name Bharathwaj.M
Age 20
D.O.J 17-05-2022
Email mp4889@srmist.edu.in
Gender Male
Contact No 9080209023
Address Ramapuram

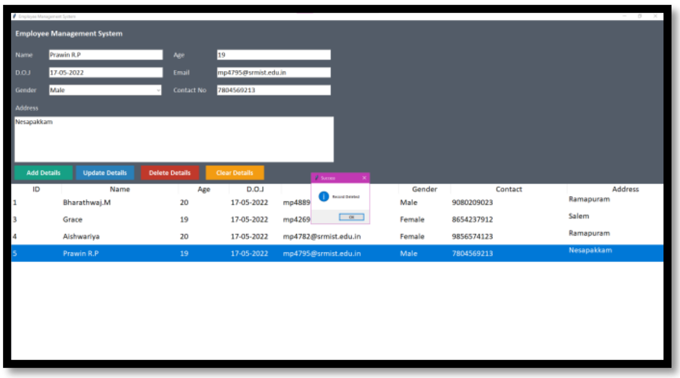
Record Updated:



Name Grace
Age 19
D.O.J 17-05-2022
Email mp4269@srmist.edu.in
Gender Female
Contact No 8654237912
Address Salem

ID	Name	Age	D.O.J		Gender	Contact	Address
1	Bharathwaj.M	20	17-05-2022	mp4889	Male	9080209023	Ramapuram
3	Grace	19	17-05-2022	mp4269	Female	8654237912	Salem
4	Aishwariya	20	17-05-2022	mp4782@srmist.edu.in	Female	9856574123	Ramapuram
5	Prawin R.P	19	17-05-2022	mp4795@srmist.edu.in	Male	7804569213	Nesapakkam

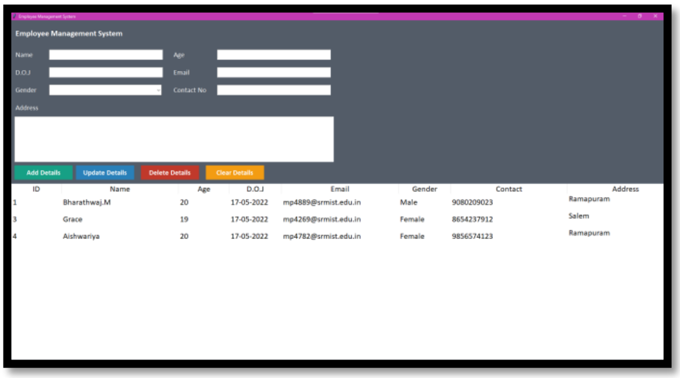
Record Deleted:



Name Prawin R.P
Age 19
D.O.J 17-05-2022
Email mp4795@srmist.edu.in
Gender Male
Contact No 7804569213
Address Nesapakkam

ID	Name	Age	D.O.J		Gender	Contact	Address
1	Bharathwaj.M	20	17-05-2022	mp4889	Male	9080209023	Ramapuram
3	Grace	19	17-05-2022	mp4269	Female	8654237912	Salem
4	Aishwariya	20	17-05-2022	mp4782@srmist.edu.in	Female	9856574123	Ramapuram
5	Prawin R.P	19	17-05-2022	mp4795@srmist.edu.in	Male	7804569213	Nesapakkam

Final List:



ID	Name	Age	D.O.J	Email	Gender	Contact	Address
1	Bharathwaj.M	20	17-05-2022	mp4889@srmist.edu.in	Male	9080209023	Ramapuram
3	Grace	19	17-05-2022	mp4269@srmist.edu.in	Female	8654237912	Salem
4	Aishwariya	20	17-05-2022	mp4782@srmist.edu.in	Female	9856574123	Ramapuram

Result:

We have successfully developed the **Employee Management System** using python. We have used the popular Tkinter library that used for rendering graphics. We used SQLite for serverless Database.