PYTHON PROJECT – EMPLOYEE MANAGEMENT SYSTEM

VAISHNAVI PASALKAR-(206605)

DOCUMENTATION

CODE-

1) db.py

import mysql.connector

from mysql.connector import Error

class Database:

    def \_\_init\_\_(self, host, database, user, password):

        self.connection = mysql.connector.connect(

            host=host,

            database=database,

            user=user,

            password=password

        )

        self.cursor = self.connection.cursor()

        self.create\_table()

    def create\_table(self):

        create\_table\_query = """

        CREATE TABLE IF NOT EXISTS employees (

            id INT AUTO\_INCREMENT PRIMARY KEY,

            name VARCHAR(100),

            age INT,

            contact VARCHAR(15),

            address VARCHAR(20)

        )

        """

        self.cursor.execute(create\_table\_query)

    def insert(self, name, age, contact, address):

        insert\_query = """

        INSERT INTO employees (name, age, contact, address)

        VALUES (%s, %s, %s, %s)

        """

        self.cursor.execute(insert\_query, (name, age, contact, address))

        self.connection.commit()

    def fetch(self):

        fetch\_query = "SELECT \* FROM employees"

        self.cursor.execute(fetch\_query)

        return self.cursor.fetchall()

    def update(self, emp\_id, name, age, contact, address):

        update\_query = """

        UPDATE employees

        SET name = %s, age = %s, contact = %s, address = %s

        WHERE id = %s

        """

        self.cursor.execute(update\_query, (name, age, contact, address, emp\_id))

        self.connection.commit()

    def remove(self, emp\_id):

        delete\_query = "DELETE FROM employees WHERE id = %s"

        self.cursor.execute(delete\_query, (emp\_id,))

        self.connection.commit()

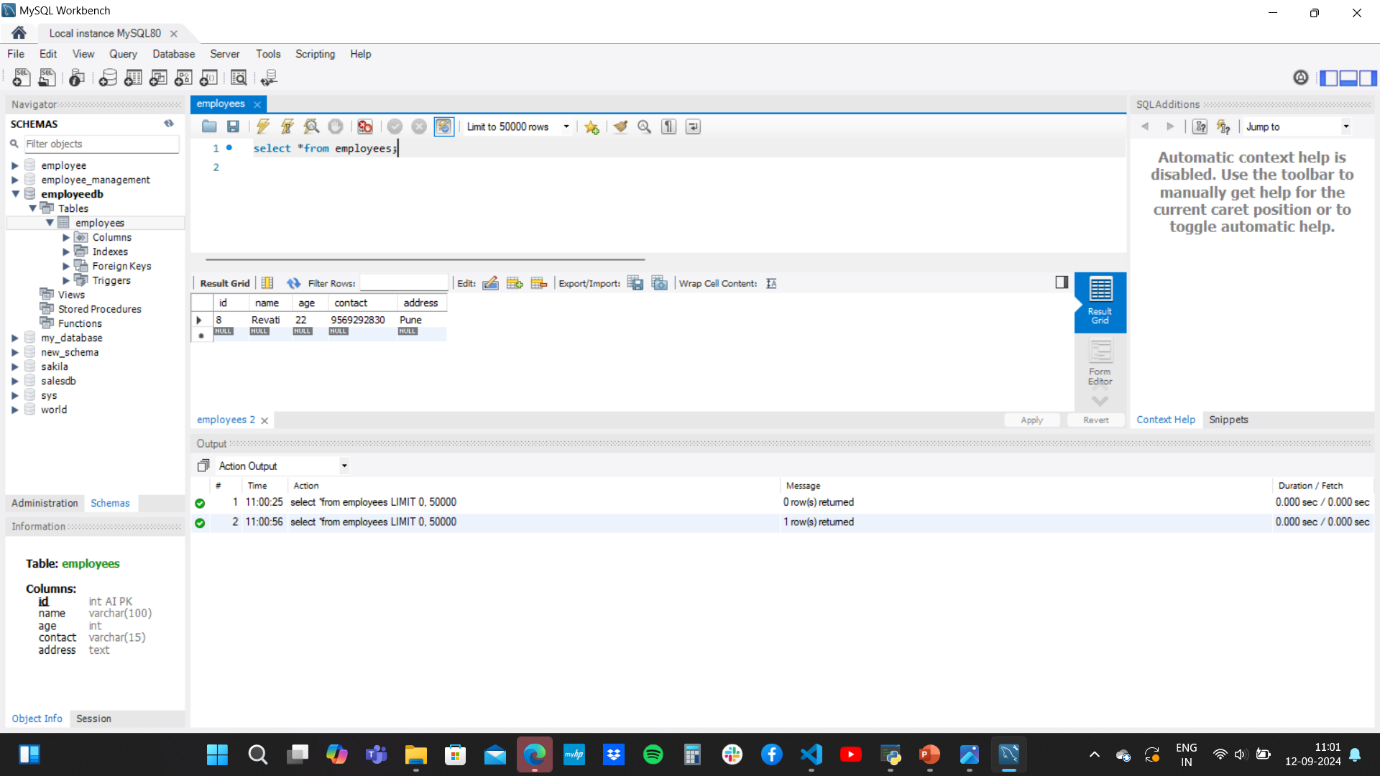
    def \_\_del\_\_(self):

        if self.connection.is\_connected():

            self.cursor.close()

            self.connection.close()

1. Main.py
2. from tkinter import \*
3. from tkinter import ttk
4. from tkinter import messagebox
5. from db import Database
6. # Initialize the Database
7. db = Database(host="localhost", database="EmployeeDB", user="root", password="root")
8. root = Tk()
9. root.title("Employee Management System")
10. root.geometry("1920x1080+0+0")
11. root.config(bg="#2c3e50")
12. root.state("zoomed")
13. name = StringVar()
14. age = StringVar()
15. contact = StringVar()
16. address = StringVar()
17. # Entries Frame
18. entries\_frame = Frame(root, bg="#535c68")
19. entries\_frame.pack(side=TOP, fill=X)
20. title = Label(entries\_frame, text="Employee Management System", font=("Calibri", 18, "bold"), bg="#535c68", fg="white")
21. title.grid(row=0, columnspan=2, padx=10, pady=20, sticky="w")
22. lblName = Label(entries\_frame, text="Name", font=("Calibri", 16), bg="#535c68", fg="white")
23. lblName.grid(row=1, column=0, padx=10, pady=10, sticky="w")
24. txtName = Entry(entries\_frame, textvariable=name, font=("Calibri", 16), width=30)
25. txtName.grid(row=1, column=1, padx=10, pady=10, sticky="w")
26. lblAge = Label(entries\_frame, text="Age", font=("Calibri", 16), bg="#535c68", fg="white")
27. lblAge.grid(row=1, column=2, padx=10, pady=10, sticky="w")
28. txtAge = Entry(entries\_frame, textvariable=age, font=("Calibri", 16), width=30)
29. txtAge.grid(row=1, column=3, padx=10, pady=10, sticky="w")
30. lblContact = Label(entries\_frame, text="Contact No", font=("Calibri", 16), bg="#535c68", fg="white")
31. lblContact.grid(row=2, column=0, padx=10, pady=10, sticky="w")
32. txtContact = Entry(entries\_frame, textvariable=contact, font=("Calibri", 16), width=30)
33. txtContact.grid(row=2, column=1, padx=10, pady=10, sticky="w")
34. lblAddress = Label(entries\_frame, text="Address", font=("Calibri", 16), bg="#535c68", fg="white")
35. lblAddress.grid(row=3, column=0, padx=10, pady=10, sticky="w")
36. txtAddress = Text(entries\_frame, width=85, height=5, font=("Calibri", 16))
37. txtAddress.grid(row=4, column=0, columnspan=4, padx=10, sticky="w")
38. def getData(event):
39. selected\_row = tv.focus()
40. data = tv.item(selected\_row)
41. global row
42. row = data["values"]
43. # Debug output
44. print("Selected row data:", row)
45. name.set(row[1])
46. age.set(row[2])
47. contact.set(row[3])
48. txtAddress.delete(1.0, END)
49. txtAddress.insert(END, row[4].strip())
50. def displayAll():
51. tv.delete(\*tv.get\_children())
52. for row in db.fetch():
53. print("Fetched row data:", row)  # Debug output
54. tv.insert("", END, values=row)
55. def add\_employee():
56. if txtName.get() == "" or txtAge.get() == "" or txtContact.get() == "" or txtAddress.get(1.0, END).strip() == "":
57. messagebox.showerror("Error in Input", "Please Fill All the Details")
58. return
59. db.insert(txtName.get(), txtAge.get(), txtContact.get(), txtAddress.get(1.0, END).strip())
60. messagebox.showinfo("Success", "Record Inserted")
61. clearAll()
62. displayAll()
63. def update\_employee():
64. if txtName.get() == "" or txtAge.get() == "" or txtContact.get() == "" or txtAddress.get(1.0, END).strip() == "":
65. messagebox.showerror("Error in Input", "Please Fill All the Details")
66. return
67. db.update(row[0], txtName.get(), txtAge.get(), txtContact.get(), txtAddress.get(1.0, END).strip())
68. messagebox.showinfo("Success", "Record Updated")
69. clearAll()
70. displayAll()
71. def delete\_employee():
72. db.remove(row[0])
73. clearAll()
74. displayAll()
75. def clearAll():
76. name.set("")
77. age.set("")
78. contact.set("")
79. txtAddress.delete(1.0, END)
80. btn\_frame = Frame(entries\_frame, bg="#535c68")
81. btn\_frame.grid(row=5, column=0, columnspan=4, padx=10, pady=10, sticky="w")
82. 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)
83. 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)
84. 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)
85. 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)
86. # Table Frame
87. tree\_frame = Frame(root, bg="#ecf0f1")
88. tree\_frame.place(x=0, y=480, width=1700, height=520)
89. style = ttk.Style()
90. style.configure("mystyle.Treeview", font=('Calibri', 18), rowheight=50)  # Modify the font of the body
91. style.configure("mystyle.Treeview.Heading", font=('Calibri', 18))  # Modify the font of the headings
92. tv = ttk.Treeview(tree\_frame, columns=(1, 2, 3, 4, 5), style="mystyle.Treeview")
93. tv.heading("1", text="ID")
94. tv.column("1", width=10, anchor='center')
95. tv.heading("2", text="Name")
96. tv.column("2", width=75, anchor='w')
97. tv.heading("3", text="Age")
98. tv.column("3", width=50, anchor='center')
99. tv.heading("4", text="Contact")
100. tv.column("4", width=100, anchor='w')
101. tv.heading("5", text="Address")
102. tv.column("5", width=950, anchor='center')  # Increase width for Address
103. tv['show'] = 'headings'
104. tv.bind("<ButtonRelease-1>", getData)
105. tv.pack(fill=X)
106. displayAll()
107. root.mainloop()

OUTPUT –

