APPENDIX 1

EMPLOYEE MANAGEMENT SYSTEM

PROJECT REPORT

By

Name – Sanjay Kumar
Roll No – 1803213128
Course – Btech
Branch – Information Technology
Date – February 1, 2020

ABES Engineering College
Ghaziabad, Pin – 201009

APPENDIX 2

Student Declaration

This is to declare that this report has been written by me. No part of the report is copied from other sources. All information included from other sources has been duly acknowledged. I aver that if any part of the report is found to be copied, I shall take full responsibility for it.

Name – Sanjay Kumar
Roll – 1803213128
Date – February 1, 2020
Course – BTech
Branch – Information Technology

Date: 28/10/2020

APPENDIX 3

TABLE OF CONTENTS

Background and objectives of the project :

Project assigned: EMPLOYEE MANAGEMENT SYSTEM

Project number: 22

Background: The project was done using the programming languages Python, MySQL, Tkinter(builtin python library)

Objectives: To manage and manipulate the basic employee data of an organization

Concrete Goals: Connecting database to the GUI programming to obtain a project like an employee management system

Motivation: I can do amazing software like employee management system by just knowing Tkinter, python, and MySQL commands

Project outcomes: In our Employee Management System project, one can manage and manipulate the data of employees like employee Id, name, number, email, address, gender, and salary of an employee. The provided data management

functions are, we can add new employee into the database and can update the existing record, we can search for a record to get details in case of larger database and also we can delete a record of unwanted employees, and we can also have an overview of a database having all data by just one click on show all records button. Simply we can have all controls on the basic information taking from an employee with ease. The main and important method is that we have a login system, where one can use MySQL credentials and login to the database and use it with secured data of employees.

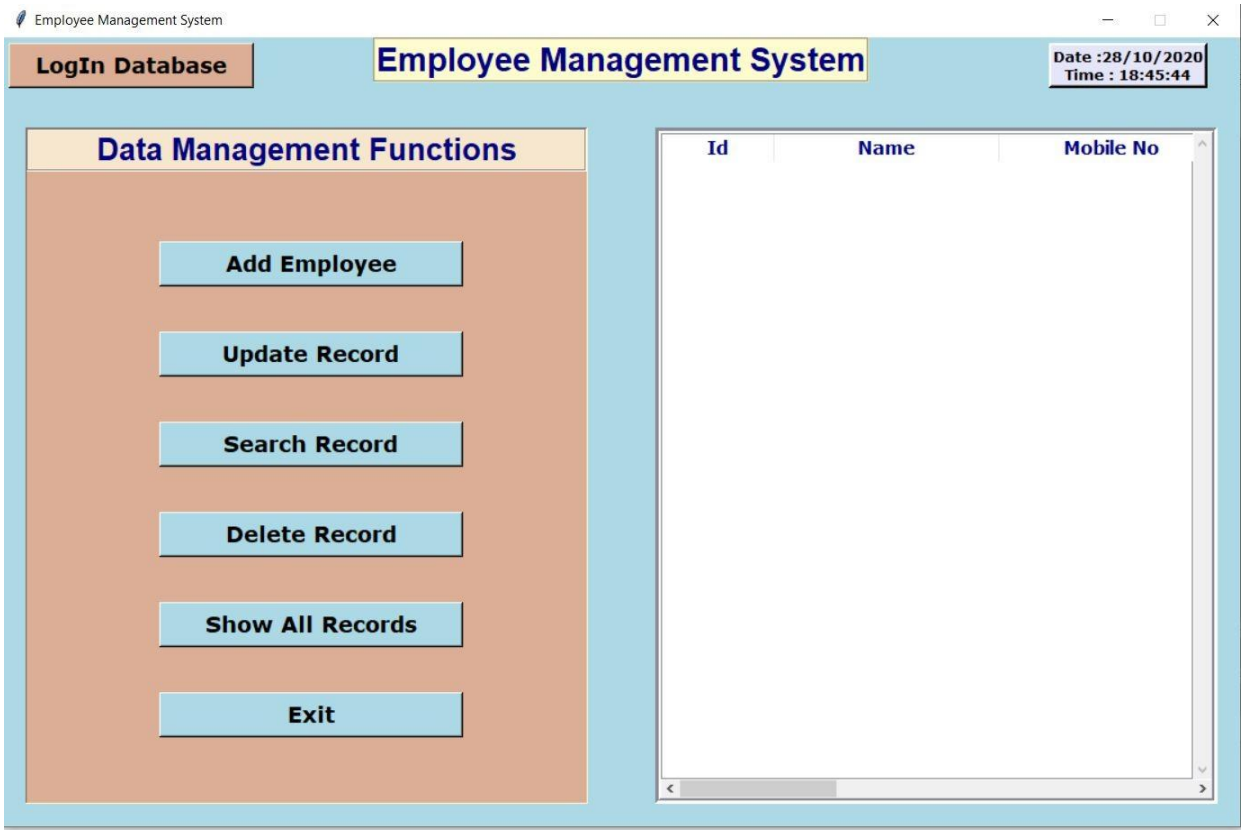
Description of project:

- Employee Management System, we should take details from the employee and store it into a particular database to have access over them
- We need a user interface application or GUI for operating the commands and functions that connecting to the database to manipulate and manage data
- We need a login system to secure the data or to get accessed by only limited members or admins
- We should add methods, which are required for managing and manipulating the employee data
- Methods like :
    - Adding a new employee data
    - Updating existing records
    - Searching existing records
    - Deleting a record in case of unwanted

- Fetching records from the database
- Showing the records on the console according to the function used
- Login system to connect database and to secure data
- Added time and date of a record

Implementation of scheduled work of Project :

Output and UI of the project:



Snap1: This is the mainwindow and UI of the project

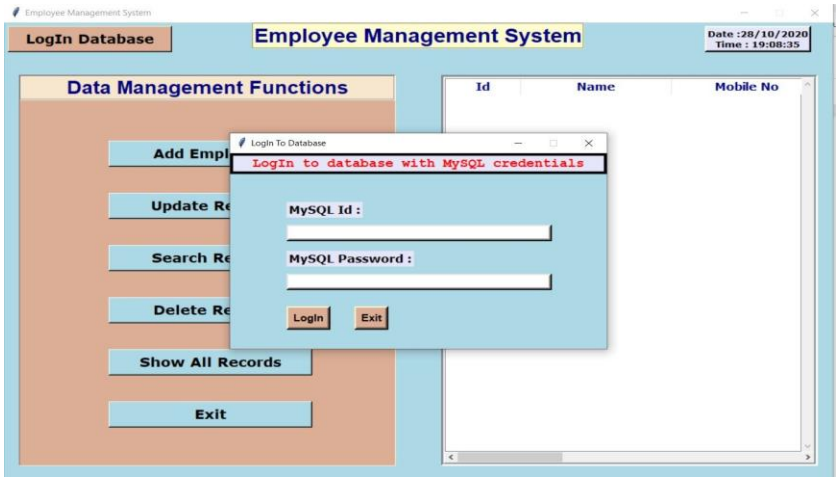
- About the geometry of this window, it contains one mainwindow named (root) and contains two frames named(DataEntryFrame and ShowDataFrame)
- This is the user interface of the project
- Add Employee, UpdateEmployee, Search Record, Delete record, ShowAllRecords, Exit are the buttons on DataEntryFrame and methods to manage and manipulate the data
- On the top left corner, “LogIn Database” button is on mainwindow (root)and it is the method to log in and connects to the database
- It has main label named “Employee Management System” situated in the root window
- On the top right corner , It is Time and Date, time calls it functions every 200milliseconds, so that time keeps on running until window exists
- The right part or ShowDataFrame will fetch records from the database and displays it like a treeview, it will display the data according to the MySQL query
- The root window is non-resizable & fixed in size

Code of mainwindow(root)&importings of tkinter:

```
from tkinter import *
from tkinter import Toplevel, messagebox, filedialog
from tkinter.ttk import Treeview
from tkinter import ttk
import pandas
import pymysql
import time

#---------------CREATING MAINWINDOW
root = Tk()
root.title("Employee Management System")
root.config(bg="light blue")
root.geometry("1100x700+200+50")
root.resizable(False, False)
```

Snap2: code of root window

1.LogIn Database :



- When we click on the login database button this popup window will open and no other button on the root window will work until we close this login window
- To login into the database, the user should have MySQL installed in the computer and also pymysql module installed in python libraries
- User should use MySQL user id and password to login into the database
  Eg: MySQL user id= root
  MySQL password= Umamahesh@123
  These are my MySQL credentials to log in to the database and the host is by default ‘localhost’ so that, anyone should use their MySQL credentials as LogIn Database details
- LogIn button connects to the database if credentials are correct else popup message will show that credentials are wrong
- Exit button will destroy the window and goes to mainwindow(root)

Code of connection to database and login button:

```
##****************************************  DEFINING LOGINDATABASE  *********************************************##
def Connectdb():

    #-------------------> defining login button & making connection to database <-------------------#
    def submitdb():
        global con, mycursor
        host = "localhost"
        user = userval.get()
        password = passwordval.get()
        try:
            con = pymysql.connect(host=host, user=user, password=password)
            mycursor = con.cursor()
        except:
            messagebox.showerror('Notifications', 'Data is incorrect please try again', parent=dbroot)
            return
        try:
            strr = 'create database employeemanagementsystem1'
            mycursor.execute(strr)
            strr = 'use employeemanagementsystem1'
            mycursor.execute(strr)
            strr = 'create table employeedata1(id int,name varchar(20),mobile varchar(12),email varchar(30),address varchar(100),
            mycursor.execute(strr)
            strr = 'alter table employeedata1 modify column id int not null'
            mycursor.execute(strr)
            strr = 'alter table employeedata1 modify column id int primary key'
            mycursor.execute(strr)
            messagebox.showinfo('Notification',
                                'database created and now you are connected connected to the database ....',
                                parent=dbroot)

        except:
            strr = 'use employeemanagementsystem1'
            mycursor.execute(strr)
            messagebox.showinfo('Notification', 'Now you are connected to the database ....', parent=dbroot)
        dbroot.destroy()
```

- This code will connect to the database and create a database named "employeemanagementsystem1" if it was the first time else it will use the created database
- This process is done by try and except statements to avoid errors recreating the existing database
- It also creates a table name "employeedata1"
- This will happen when we click on the login button in LogIn Database window

Code for date and time:

```
##****************************************  DEFINING TIME&DATE METHODS    *********************************************##
def tick():
    time_string = time.strftime("%H:%M:%S")
    date_string = time.strftime("%d/%m/%Y")
    clock.config(text='Date :' + date_string + "\n" + "Time : " + time_string)
    clock.after(200, tick)

##****************************************  END OF TIME&DATE METHODS    *********************************************##
```

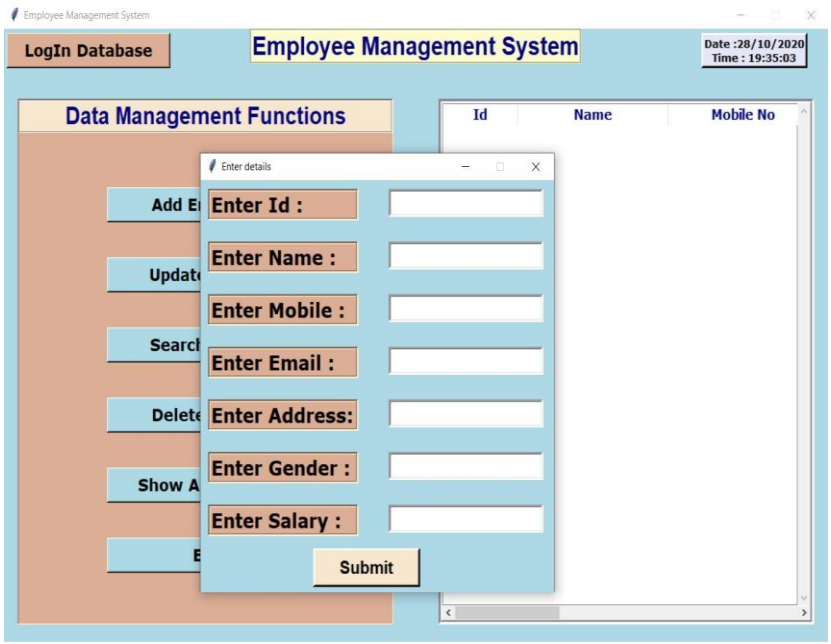
- 'clock.after(200,tick)' this will recall 'tick' function every 200milliseconds so time constantly runs on the root window

- And this also useful in adding added date and added time of a record to the database

2.Add Employee:

- When we click on Add Employee this window will open and it is not resizable and no other button on the root window will work until we close the Add Employee window
- Here Id part can not be null and can not be any other data type except Integer if any other datatype or null value is submitted the popup error message will open and it does not affect or change anything in the database
- Any other value in add employee form can be null and of any datatype except id
- Id is defined as the primary key in the database
- When we click on the submit button the data entered in the entry labels will be stored in the database and displayed in the ShowDataFrame
- Id is a unique value otherwise it will raise a popup error message

Code for addemployee & getting and storing entered data into database:



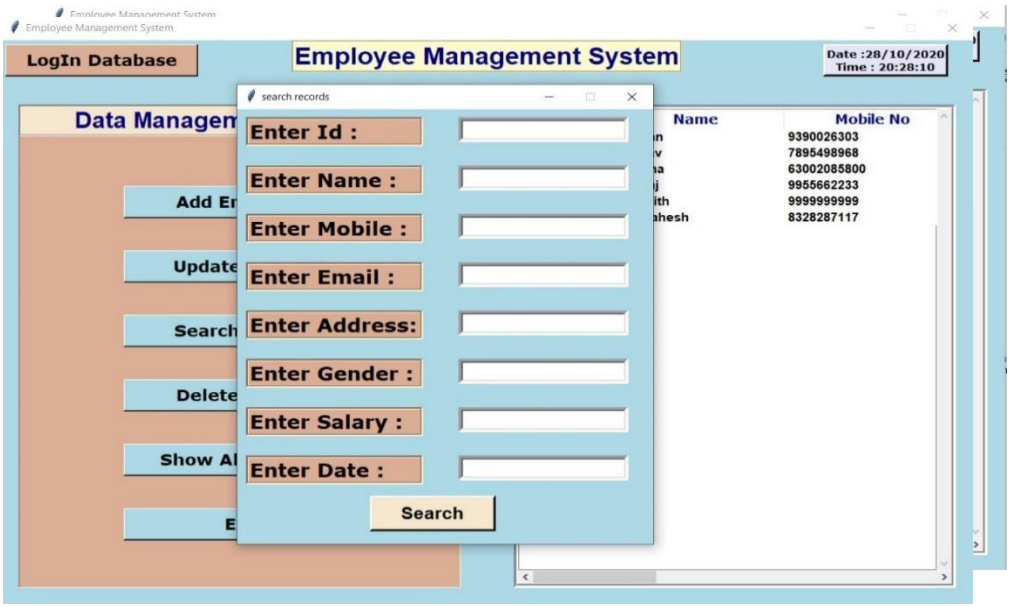
```
######################################    DEFINING ADDEMPLOYEE   ##########################################
def addemployee():

    #<<<------------ defining the submit button in addemployee form ------------>>>#
    def submitadd():
        id = idval.get()                        #getting data from entries...or entry lables
        name = nameval.get()
        mobile = mobileval.get()
        email = emailval.get()
        address = addressval.get()
        gender = genderval.get()
        salary = salaryval.get()
        addedtime = time.strftime("%H:%M:%S") #to get time
        addeddate = time.strftime("%d/%m/%Y") #to get date
        try:
            strr = 'insert into employeedata1 values(%s,%s,%s,%s,%s,%s,%s,%s,%s)'
            mycursor.execute(strr, (id, name, mobile, email, address, gender, salary, addeddate, addedtime))
            con.commit()
            res = messagebox.askyesnocancel('Notificatrions','Id {} Name {} Added sucessfully.. and want to clean the form'.forma
                                            parent=addroot)

            # clearing the entry form after adding details to database only if res==true
            if (res == True):
                idval.set('')
                nameval.set('')
                mobileval.set('')
                emailval.set('')
                addressval.set('')
                genderval.set('')
                salaryval.set('')
        except:
```

```
except:
    messagebox.showerror('Notifications', 'Id Already Exist try another id...', parent=addroot)
strr = 'select * from employeedata1'
mycursor.execute(strr)
datas = mycursor.fetchall()
employeetable.delete(*employeetable.get_children())
for i in datas:
    vv = [i[0], i[1], i[2], i[3], i[4], i[5], i[6], i[7], i[8]] #making data as list to print on treeview
    employeetable.insert('', END, values=vv)                    #to show data,employeetable treeview ,prints from star

#<<<---------------- end of defining sumbitbutton of addemployee ---------------->>>#
```

- This is the code for adding entered data into the database
- ‘strr= select * from employeedata1’ this will select all data from the database and the rest of the below code is used to show the added data in ShowDataFrame as a treeview
- ‘employeetable.delete(*employeetable.get_children()) “ this will clear the console in showdataframe and the code below it(vv=i[]) will print all the records and the newly added data also

3. Update record:

- When we click on Update Employee this window will open and it is not



resizable and no other button on the root window will work until we close the Update Employee window

- Here for updating the record id is mandatory and without id we cannot update the record because id is the primary key
- As similarly any other entry can be null except id
- On clicking the update button the record with provided id will be updated in database as well as the updated record will be shown in the ShowDataFrame
- Code for update record and adding the updated result to database is as same as addemployee

4.Search Record:

- When we click on Search Employee this window will open and it is not resizable and no other button on the root window will work until we close the Search Employee window
- Here for searching the record, we can search with any parameter
- We can search with only any one of the above parameters
- On searching the valid parameter the results will be displayed on the ShowDataFrame
- Search only with exact value do not search with half values

Code for search employee and displaying the data in ShowDataFrame:

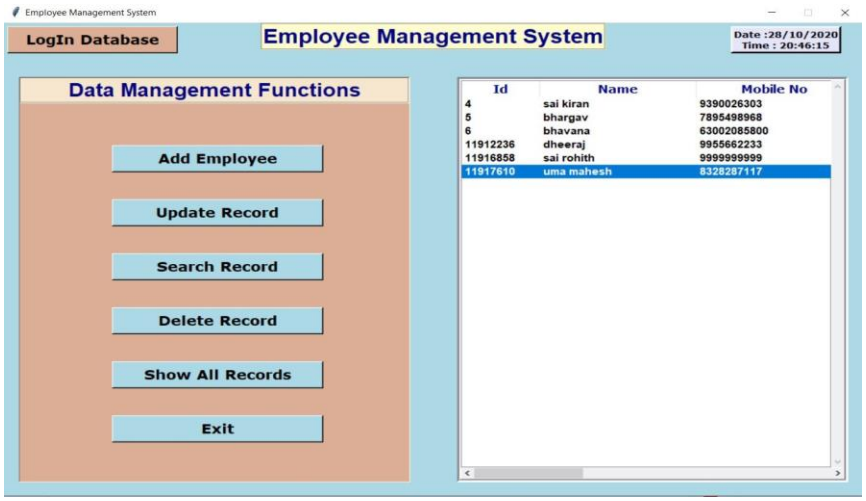
```
########################################   DEFINING SEARCHEMPLOYEE   ########################################
def searchemployee():

    #<<<-------------- defining the search button in searchemployee form ------------->>>#
    def search():
        id = idval.get()
        name = nameval.get()
        mobile = mobileval.get()
        email = emailval.get()
        address = addressval.get()
        gender = genderval.get()
        salary = salaryval.get()
        addeddate = time.strftime("%d/%m/%Y")

        #<<<--------------- code for searching records ------------->>>#
        if (id != ''):
            strr = 'select *from employeedata1 where id=%s'
            mycursor.execute(strr, (id))
            datas = mycursor.fetchall()
            employeetable.delete(*employeetable.get_children())
            for i in datas:
                vv = [i[0], i[1], i[2], i[3], i[4], i[5], i[6], i[7], i[8]]
                employeetable.insert('', END, values=vv)
        elif (name != ''):
            strr = 'select *from employeedata1 where name=%s'
            mycursor.execute(strr, (name))
            datas = mycursor.fetchall()
            employeetable.delete(*employeetable.get_children())
            for i in datas:
                vv = [i[0], i[1], i[2], i[3], i[4], i[5], i[6], i[7], i[8]]
                employeetable.insert('', END, values=vv)
```

```
        elif (mobile != ''):
            strr = 'select *from employeedata1 where mobile=%s'
            mycursor.execute(strr, (mobile))
            datas = mycursor.fetchall()
            employeetable.delete(*employeetable.get_children())
            for i in datas:
                vv = [i[0], i[1], i[2], i[3], i[4], i[5], i[6], i[7], i[8]]
                employeetable.insert('', END, values=vv)
        elif (email != ''):
            strr = 'select *from employeedata1 where email=%s'
            mycursor.execute(strr, (email))
            datas = mycursor.fetchall()
            employeetable.delete(*employeetable.get_children())
            for i in datas:
                vv = [i[0], i[1], i[2], i[3], i[4], i[5], i[6], i[7], i[8]]
                employeetable.insert('', END, values=vv)
        elif (address != ''):
            strr = 'select *from employeedata1 where address=%s'
            mycursor.execute(strr, (address))
            datas = mycursor.fetchall()
            employeetable.delete(*employeetable.get_children())
            for i in datas:
                vv = [i[0], i[1], i[2], i[3], i[4], i[5], i[6], i[7], i[8]]
                employeetable.insert('', END, values=vv)
        elif (gender != ''):
            strr = 'select *from employeedata1 where gender=%s'
            mycursor.execute(strr, (gender))
            datas = mycursor.fetchall()
            employeetable.delete(*employeetable.get_children())
            for i in datas:
                vv = [i[0], i[1], i[2], i[3], i[4], i[5], i[6], i[7], i[8]]
                employeetable.insert('', END, values=vv)
        elif (salary != ''):
            strr = 'select *from employeedata1 where salary=%s'
            mycursor.execute(strr, (salary))
            datas = mycursor.fetchall()
            employeetable.delete(*employeetable.get_children())
            for i in datas:
                vv = [i[0], i[1], i[2], i[3], i[4], i[5], i[6], i[7], i[8]]
```

- This is the code used to search the records

- At first, it will select the records with the details entered in the entry box, Next, it fetches all details of that record and next prints it into the showdataframe, this is the logic behind that code for searching the record
    - This will execute when we hit on the search button in search record window

5.Delete Record:



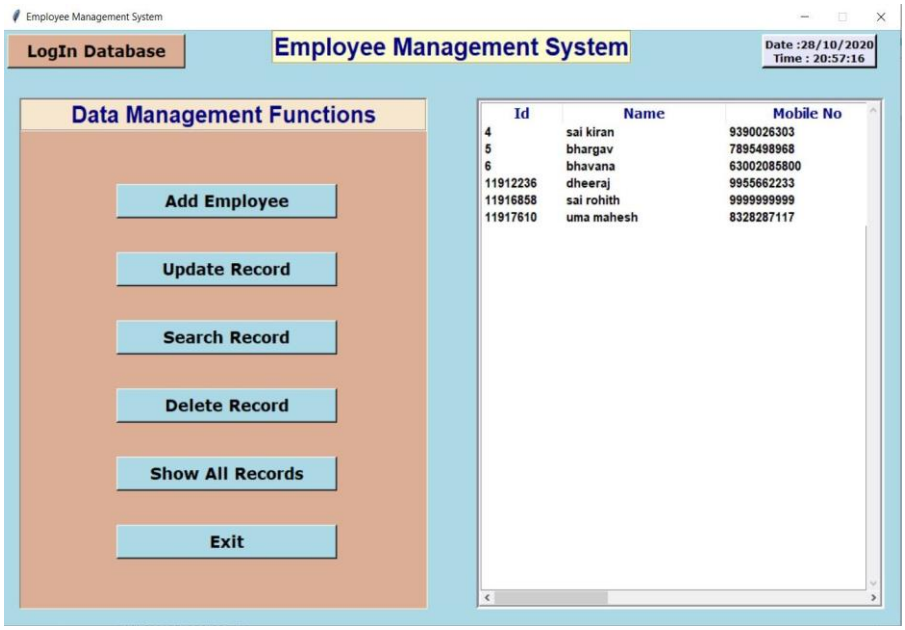
- To delete a record we should select the record as we selected in the snap, and next we should hit the delete button, then the record is deleted from database and it will also disappear from treeview or showdataframe
- We can also delete the record by searching it first and selecting it on showdatframe and hitting on the delete button

Code for delete employee and deleting the data from ShowDataFrame&database:

```
########################################  DEFINING DELETE EMPLOYEE   ##############################################
def deleteemployee():
    cc = employeetable.focus()         #focus on the record which we click and gets data of that record
    content = employeetable.item(cc)
    pp = content['values'][0]          #it gives the id of the particular record to delete
    strr = 'delete from employeedata1 where id=%s'
    mycursor.execute(strr, (pp))
    con.commit()
    messagebox.showinfo('Notifications', 'Id {} deleted sucessfully...'.format(pp))
    strr = 'select *from employeedata1'
    mycursor.execute(strr)
    datas = mycursor.fetchall()
    employeetable.delete(*employeetable.get_children())
    for i in datas:
        vv = [i[0], i[1], i[2], i[3], i[4], i[5], i[6], i[7], i[8]]
        employeetable.insert('', END, values=vv)

########################################  END OF DELETEEMPLOYEE   ##############################################
```

- This code will delete the data from database first and next, it clears the console and reprints the console by fetching details from database
- So the deleted record will be disappeared from the console or showdataframe

6. ShowAllRecords:



- The show all records button will print all data available in the database on showdataframe or console
- The code for show all records is simple it will get all records and puts on the console with just one click

7.Exit:

- Exit will just destroy the mainwindow(root) and code exits when we clickon it
- And when we exit the data base is automatically logged out and we need to login again when we run to fetch data and manipulate data

Technologies and Framework to be used :

IDE used : pycharm and jupyter notebook

Programming languages used: python, python(Tkinter), MySQL

- Python is used for methods and connecting GUI buttons, labels,entryforms, etc
- Tkinter is used to develop the graphical user interface
- MySQL is used to develop the database related quires and storing the data

Important:-

- To get a connection to database installing MySQL is mandatory
- After installing MySQL to connect to python ‘import pymysql’ command is written in the code
- So the module pymysql is also important to get installed in python libraries
- If pymysql is not installed in the python libraries do these steps:
    - If code is running in jupyter notebook
    - Goto anaconda command prompt-->givec ommand ‘pip install pymysql’
    - If code is running in pycharm
    - Go to settings after opening code -->go to project settings-->add interpreter-->search-->pymysql-->install-->run code

THANK YOU SIR, YOUR CONCEPTS MADE OUR PROJECT EASY BY YOUR TEACHING CONCEPTS OF PYTHON,TKINTER, DATABASE