SOURCES CODE

B.TECH 3rd YR.(6th SEM.) 1st PROJECT

```
from tkinter import*
from tkinter import messagebox, ttk
import mysql.connector
import time
class EmployeeSystem:
   def init (self,root):
       self.root=root
       root.title("EMPLOYEE PAYROLL MANAGEMENT SYSTEM | BY SRIHARI KAUSHIK
BTECH. (CSE) (3rd YR.)")
       root.geometry('1350x700')
       root.config(bg="white")
       title=Label(self.root,text="EMPLOYEE PAYROLL MANAGEMENT
SYSTEM", font=('times new roman', 30, 'bold'), bg='#262626', fg='white')
       title.place(x=0, y=0, relwidth=1)
       btn emp=Button(root,text="ALL EMPLOYEE'S
DETAILS", command=self.employee frame, font=('times new roman', 10), bg='slate
blue',fg='blue')
       btn emp.place(x=1100, y=14)
       #======Variables======
       self.var e id=StringVar()
       self.var designation=StringVar()
       self.var name=StringVar()
       self.var age=StringVar()
       self.var gender=StringVar()
       self.var email=StringVar()
       self.var hr location=StringVar()
       self.var dob=StringVar()
       self.var doj=StringVar()
       self.var proof id=StringVar()
```

```
self.var contact=StringVar()
        self.var status=StringVar()
        self.var experience=StringVar()
        self.var address=StringVar()
        Frame1=Frame(self.root,bd=5,relief=RIDGE,bg='white')
        Frame1.place (x=10, y=70, width=715, height=560)
        title2=Label(Frame1,text="Employee Details",font=('times new
romans',20),bg='lightgray',fg='black',anchor="w",padx=10)
        title2.place(x=0, y=0, relwidth=1)
        #======ROW1======
        lbl code=Label(Frame1, text="Emp.Code:", font=('times new
roman',16),bg='white',fg='black')
        lbl code.place (x=10, y=65)
        self.txt code=Entry(Frame1, font=('times new
roman',13),textvariable=self.var e id,bg='lightyellow',fg='black')
        self.txt code.place(x=150,y=65,width=200)
btn search=Button(Frame1,text="Search",command=self.search,font=('times new
roman',16),bg='lightgreen',fg='black')
        btn search.place(x=390, y=55)
        #======ROW2======
        lbl designation=Label(Frame1, text="Designation:", font=('times new
roman',16),bg='white',fg='black')
        lbl designation.place(x=10,y=115)
        txt designation=Entry(Frame1, font=('times new
roman',13),textvariable=self.var designation,bq='lightyellow',fq='black')
        txt designation.place(x=150,y=115,width=200)
        lbl DOB=Label(Frame1, text="D.O.B:", font=('times new
roman',16),bg='white',fg='black')
        lbl DOB.place(x=400, y=115)
```

```
txt DOB=Entry(Frame1, font=('times new
roman',13),textvariable=self.var dob,bg='lightyellow',fg='black')
        txt DOB.place (x=500, y=115)
        #======ROW3=======
        lbl name=Label(Frame1, text="Name:", font=('times new
roman',16),bg='white',fg='black')
        lbl name.place(x=10, y=165)
        txt name=Entry(Frame1, font=('times new
roman',13),textvariable=self.var name,bg='lightyellow',fg='black')
        txt name.place(x=150, y=165, width=200)
        lbl DOJ=Label(Frame1, text="D.O.J:", font=('times new
roman',16),bg='white',fg='black')
        lbl DOJ.place(x=400, y=165)
        txt DOJ=Entry(Frame1, font=('times new
roman',13),textvariable=self.var doj,bg='lightyellow',fg='black')
        txt DOJ.place(x=500, y=165)
        #======ROW4======
        lbl age=Label(Frame1, text="Age:", font=('times new
roman',16),bg='white',fg='black')
        lbl age.place(x=10, y=215)
        txt age=Entry(Frame1, font=('times new
roman',13),textvariable=self.var age,bg='lightyellow',fg='black')
        txt_age.place(x=150, y=215, width=200)
        lbl experience=Label(Frame1, text="Experience:", font=('times new
roman',16),bg='white',fg='black')
        lbl experience.place(x=380,y=215)
        txt experience=Entry(Frame1, font=('times new
roman',13),textvariable=self.var experience,bg='lightyellow',fg='black')
        txt experience.place (x=500, y=215)
```

```
#======ROW5======
        lbl gender=Label(Frame1, text="Gender:", font=('times new
roman',16),bg='white',fg='black')
        lbl gender.place(x=10, y=265)
        txt gender=Entry(Frame1, font=('times new
roman',13),textvariable=self.var gender,bg='lightyellow',fg='black')
        txt gender.place(x=150,y=265,width=200)
        lbl id=Label(Frame1, text="Proof ID:", font=('times new
roman',16),bg='white',fg='black')
        lbl id.place(x=380, y=265)
        txt id=Entry(Frame1, font=('times new
roman',13),textvariable=self.var proof id,bg='lightyellow',fg='black')
        txt id.place (x=500, y=265)
        #======ROW6======
        lbl Email=Label(Frame1, text="Email:", font=('times new
roman',16),bg='white',fg='black')
        lbl Email.place(x=10, y=315)
        txt Email=Entry(Frame1, font=('times new
roman',13),textvariable=self.var email,bg='lightyellow',fg='black')
        txt Email.place (x=150, y=315, width=200)
        lbl number=Label(Frame1,text="Contact No.:",font=('times new
roman',16),bg='white',fg='black')
        lbl number.place(x=380, y=315)
        txt number=Entry(Frame1, font=('times new
roman',13),textvariable=self.var contact,bg='lightyellow',fg='black')
        txt number.place(x=500, y=315)
        #======ROW7=======
```

```
lbl address=Label(Frame1, text="Address:", font=('times new
roman',16),bg='white',fg='black')
       lbl address.place (x=10, y=415)
       txt address=Entry(Frame1, font=('times new
roman',13),textvariable=self.var address,bg='lightyellow',fg='black')
       txt address.place(x=150,y=415,width=535,height=60)
       title0=Label(Frame1,text="BY --SRIHARI KAUSHIK [B.TECH 3rd
YR.]", font=('times new
roman',20),bg='LightGoldenrod2',fg='black',anchor="w",padx=10).place(x=0,y=
514, relwidth=1)
       #======ROW8======
       lbl location=Label(Frame1, text="HR Location:", font=('times new
roman',16),bg='white',fg='black')
       lbl location.place(x=10, y=365)
       txt location=Entry(Frame1, font=('times new
roman',13),textvariable=self.var hr location,bg='lightyellow',fg='black')
       txt location.place(x=150, y=365, width=200)
       lbl status=Label(Frame1, text="Status:", font=('times new
roman',16),bg='white',fg='black')
       lbl status.place(x=400, y=365)
       txt status=Entry(Frame1, font=('times new
roman',13),textvariable=self.var status,bg='lightyellow',fg='black')
       txt status.place(x=500, y=365)
       self.var month=StringVar()
       self.var year=StringVar()
       self.var salary=StringVar()
       self.var t days=StringVar()
       self.var absent=StringVar()
       self.var medical=StringVar()
```

```
self.var pf=StringVar()
        self.var convence=StringVar()
        self.var net salary=StringVar()
        Frame2=Frame(root,bd=3,relief=RIDGE,bg='white')
        Frame2.place (x=730, y=70, width=540, height=300)
        tile3=Label(Frame2,text="Employee Salary Details:",font=('times new
roman',20),bg='lightgray',fg='black',anchor="w",padx=10)
        tile3.place(x=0, y=0, relwidth=1)
        #======2ROW1=======
        lbl Month=Label(Frame2,text="Month:",font=('times new
roman',13),bg='white',fg='black')
        lbl Month.place(x=10, y=60)
        txt Month=Entry(Frame2, font=('times new
roman',12),textvariable=self.var month,bg='lightyellow',fg='black')
        txt Month.place (x=70, y=60, width=90)
        lbl year=Label(Frame2, text="Year:", font=('times new
roman',13),bg='white',fg='black')
        lbl year.place(x=200, y=60)
        txt year=Entry(Frame2, font=('times new
roman',12),textvariable=self.var year,bg='lightyellow',fg='black')
        txt year.place (x=250, y=60, width=90)
        lbl salary=Label(Frame2, text="Basic:", font=('times new
roman',13),bg='white',fg='black')
        lbl salary.place(x=380, y=60)
```

```
txt salary=Entry(Frame2, font=('times new
roman',12),textvariable=self.var salary,bg='lightyellow',fg='black')
        txt salary.place(x=430,y=60,width=90)
        #======2ROW2======
        lbl Days=Label(Frame2,text="Total Days:",font=('times new
roman',13),bg='white',fg='black')
        lbl Days.place (x=10, y=100)
        txt Days=Entry(Frame2, font=('times new
roman',13),textvariable=self.var t days,bg='lightyellow',fg='black')
        txt Days.place(x=100, y=100, width=150)
        lbl Absent=Label(Frame2, text="Days Absent:", font=('times new
roman',13),bg='white',fg='black')
        lbl Absent.place(x=265, y=100)
        txt Absent=Entry(Frame2, font=('times new
roman',13),textvariable=self.var absent,bg='lightyellow',fg='black')
        txt Absent.place (x=370, y=100, width=150)
        #======2ROW3=======
        lbl medical=Label(Frame2,text="Medical:",font=('times new
roman',13),bg='white',fg='black')
        lbl medical.place(x=10, y=140)
        txt medical=Entry(Frame2, font=('times new
roman',13),textvariable=self.var medical,bg='lightyellow',fg='black')
        txt medical.place(x=100,y=140,width=150)
        lbl pf=Label(Frame2, text="P.F.:", font=('times new
roman',13),bg='white',fg='black')
        lbl pf.place(x=320, y=140)
        txt pf=Entry(Frame2, font=('times new
roman',13),textvariable=self.var pf,bg='lightyellow',fg='black')
```

```
txt pf.place (x=370, y=140, width=150)
        #======2ROW4=======
        lbl convence=Label(Frame2,text="Convence:",font=('times new
roman',13),bg='white',fg='black')
        lbl convence.place(x=10,y=180)
        txt convence=Entry(Frame2, font=('times new
roman',13),textvariable=self.var convence,bg='lightyellow',fg='black')
        txt convence.place(x=100, y=180, width=150)
        lbl net=Label(Frame2, text="Net Salary:", font=('times new
roman',13),bg='white',fg='black')
        lbl net.place (x=275, y=180)
        txt net=Entry(Frame2, font=('times new
roman',13),textvariable=self.var net salary,bg='lightyellow',fg='black')
        txt net.place(x=370, y=180, width=150)
btn calculate=Button(Frame2,text="Calculate",command=self.calculate,font=('
times new roman', 13), bg='lightblue', fg='black')
        btn calculate.place(x=138, y=240)
self.btn save=Button(Frame2,text="Save",command=self.save,font=('times new
roman',13),bg='light green',fg='black')
        self.btn save.place(x=239, y=240)
self.btn clear=Button(Frame2, text="Clear", command=self.clear, font=('times
new roman',13),bg='light blue',fg='black')
        self.btn clear.place(x=465, y=240)
```

```
self.btn update=Button(Frame2,text="Update",state=NORMAL,command=self.updat
e, font=('times new roman',13),bg='light green',fg='black')
       self.btn update.place(x=305, y=240)
self.btn delete=Button(Frame2,text="Delete",state=NORMAL,command=self.delet
e, font=('times new roman',13),bg='orange',fg='black')
       self.btn_delete.place(x=385,y=240)
       Frame3=Frame(self.root,bd=5,relief=RIDGE,bg='white')
       Frame3.place (x=730, y=380, width=540, height=250)
       self.var txt=StringVar()
       self.var operator=''
       def btn click(num):
          self.var operator=self.var operator+str(num)
          self.var txt.set(self.var operator)
       def result():
          res=str(eval(self.var operator))
          self.var txt.set(res)
          self.var operator=''
       def clear cal():
          self.var txt.set('')
          self.var operator=''
       cal frame=Frame(Frame3,bg="antique white",bd=2,relief=RIDGE)
       cal frame.place(x=5, y=5, width=262, height=230)
```

```
text result=Entry(cal frame, bg='snow', textvariable=self.var txt, font=("time
s new roman",20,"bold"))
       text_result.place(x=5,y=5,width=250,height=40)
       btn 7=Button(cal frame,text="7",command=lambda:btn click(7),font=("times
new roman",12,"bold"))
       btn 7.place(x=8,y=50,width='60',height='40')
btn_8=Button(cal_frame,text="8",command=lambda:btn_click(8),font=("times
new roman",12,"bold"))
       btn 8.place(x=68,y=50,width="60",height="40")
btn_9=Button(cal_frame,text="9",command=lambda:btn_click(9),font=("times
new roman",12,"bold"))
       btn 9.place(x=128,y=50,width="60",height="40")
btn div=Button(cal frame,text="/",command=lambda:btn click("/"),font=("time
s new roman",20,"bold"))
       btn div.place(x=188,y=50,width="60",height="40")
        btn 4=Button(cal frame, text="4", command=lambda:btn click(4), font=("times
new roman",12,"bold"))
       btn 4.place(x=8, y=92, width="60", height="40")
```

```
btn 5=Button(cal frame,text="5",command=lambda:btn click(5),font=("times
new roman",12,"bold"))
      btn 5.place(x=68, y=92, width="60", height="40")
btn 6=Button(cal frame, text="6", command=lambda:btn click(6), font=("times
new roman",12,"bold"))
      btn 6.place(x=128,y=92,width="60",height="40")
s new roman",20,"bold"))
      btn mul.place(x=188,y=92,width="60",height="40")
       btn 1=Button(cal frame,text="1",command=lambda:btn click(1),font=("times
new roman",12,"bold"))
      btn 1.place(x=8, y=135, width="60", height="40")
btn 2=Button(cal frame,text="2",command=lambda:btn click(2),font=("times
new roman",12,"bold"))
      btn 2.place(x=68, y=135, width="60", height="40")
btn 3=Button(cal frame, text="3", command=lambda:btn click(3), font=("times
new roman",12,"bold"))
      btn 3.place(x=128,y=135,width="60",height="40")
      btn min=Button(cal frame, text="-", command=lambda:btn click("-
"), font=("times new roman", 20, "bold"))
      btn min.place(x=188,y=135,width="60",height="40")
```

```
btn_0=Button(cal_frame,text="0",command=lambda:btn click(0),font=("times
new roman",12,"bold"))
       btn 0.place(x=8,y=177,width="60",height="40")
       btn dot=Button(cal frame, text="C", command=clear cal, font=("times
new roman",12,"bold"))
       btn dot.place(x=68,y=177,width="60",height="40")
btn sum=Button(cal frame,text="+",command=lambda:btn click("+"),font=("time
s new roman",12,"bold"))
       btn sum.place(x=128,y=177,width="60",height="40")
       btn_equal=Button(cal_frame,text="=",command=result,font=("times new
roman",20,"bold"))
       btn equal.place (x=188, y=177, width="60", height="40")
       sal frame=Frame(Frame3, bg="white", bd=2, relief=RIDGE)
       sal frame.place (x=265, y=5, width=262, height=230)
       title4=Label(sal frame, text="Salary Receipt", font=('times new
roman',20),bg='lightgray',fg='black',anchor="w",padx=10)
       title4.place(x=0, y=0, relwidth=1)
       sal frame2=Frame(sal frame,bg="white",bd=2,relief=RIDGE)
       sal frame2.place(x=5, y=40, width=250, height=180)
       self.sample=f'''\tINTERNATION SCHOOL\n Address:B-12
Kamlanagar, Floor-4
Employee ID\t\t: ID
Employee Name\t\t: ----
```

```
Emp. Designation\t\t: ----
Salary Of \t\t: MON/YYYY
Generated On\t\t: DD-MM-YYYY
Total Days\t\t: DD
Total Present\t\t: DD
Total Absent\t\t: DD
Convence\t\t: Rs.----
Medical\t\t: Rs.----
PF \t\t: Rs.----
Gross Payment\t\t: Rs.----
Net Salary\t\t: Rs.----
This is computer genarated slip,
not required any signature'''
       scroll y=Scrollbar(sal frame2, orient=VERTICAL)
       scroll y.pack(side=RIGHT,fill=Y)
       self.txt salary receipt=Text(sal frame2, font=("times new
roman",11),bg="lightyellow",yscrollcommand=scroll y.set)
       self.txt salary receipt.pack(side=LEFT,fill="both",expand=1)
       scroll y.config(command=self.txt salary receipt.yview)
       self.txt salary receipt.insert(END, self.sample)
    def search(self):
       try:
           con=mysql.connector.connect(host='localhost',user='root',
                                     password='kaushik@',db='project')
```

```
c=con.cursor()
            c.execute('SELECT * FROM employee data WHERE
e id=%s',(str(self.var_e_id.get()),))
            row=c.fetchone()
            if row is not None:
                self.var e id.set(row[0])
                self.var designation.set(row[1])
                self.var name.set(row[2])
                self.var age.set(row[3])
                self.var gender.set(row[4])
                self.var email.set(row[5])
                self.var hr location.set(row[6])
                self.var doj.set(row[7])
                self.var_dob.set(row[8])
                self.var experience.set(row[9])
                self.var_proof_id.set(row[10])
                self.var contact.set(row[11])
                self.var status.set(row[12])
                #self.var address.delete('1.0',END)
                #self.var address.insert(END, row[13])
                self.var month.set(row[14])
                self.var year.set(row[15])
                self.var salary.set(row[16])
                self.var t days.set(row[17])
                self.var absent.set(row[18])
                self.var medical.set(row[19])
                self.var_pf.set(row[20])
                self.var convence.set(row[21])
                self.var net salary.set(row[22])
                file=open('Salary receipt/'+str(row[23]),'r')
                self.txt salary receipt.delete('1.0',END)
```

```
for i in file:
                    self.txt salary receipt.read(END,i)
                    #self.txt salary receipt.insert(END,i)
                file.close()
                self.btn save.config(state=NORMAL)
                self.btn update.config(state=NORMAL)
                self.btn delete.config(state=NORMAL)
                self.txt code.config(state='readonly')
            else:
                messagebox.showerror("Error", 'Invalid Employee ID, TRY AGAIN
with anothor ID',parent=self.root)
        except Exception as ex:
            messagebox.showerror("Error",f'Error due to: {str(ex)}')
    def clear(self):
        self.btn save.config(state=NORMAL)
        self.btn update.config(state=NORMAL)
        self.btn delete.config(state=NORMAL)
        self.txt code.config(state=NORMAL)
        entry widgets=(self.var e id, self.var designation, self.var name,
self.var age, self.var gender,
                       self.var email, self.var hr location, self.var dob,
self.var doj, self.var proof id,
                       self.var contact, self.var_status,
self.var experience, self.var address, self.var month,
                       self.var year, self.var salary, self.var t days,
self.var absent, self.var medical,
                       self.var pf, self.var convence, self.var net salary)
        for entry in entry widgets:
            entry.set("")
```

```
self.txt salary receipt.delete("1.0", "end")
        self.txt salary receipt.insert(END, self.sample)
    def delete(self):
        if self.var e id.get() == '':
            messagebox.showerror("Error", 'EMPLOYEE ID IS REQUIRED')
        else:
            try:
con=mysql.connector.connect(host='localhost',user='root',password='kaushik@
',db='project')
                c=con.cursor()
                c.execute('SELECT * FROM employee data WHERE
e id=%s',(str(self.var e id.get()),))
                row=c.fetchone()
                if row==None:
                    messagebox.showerror("Error",'Invalid Employee ID, try
again with anothor ID', parent=self.root)
                else:
                    op=messagebox.askyesno("CONFIRM", 'DO YOU REALLY WANT TO
DELETE?')
                    if op==True:
                         c.execute("DELETE FROM employee data WHERE
e id=%s",(str(self.var e id.get()),))
                         con.commit()
                         messagebox.showinfo('SUCCESS','DATA DELETED
SUCCESSFULLY')
                         self.clear()
            except Exception as ex:
                messagebox.showerror("Error", 'Error due to: {str(ex)}')
    def calculate(self):
        if (self.var month.get()=='' or self.var year.get()=='' or
            self.var salary.get() == '' or self.var t days.get() == '' or
            self.var absent.get() == '' or self.var e id.get() == '' or
```

```
self.var name.get() == '' or self.var status.get() == ''):
            messagebox.showerror('Error','All fields are required')
        else:
            per day=int(self.var salary.get())/int(self.var t days.get())
            work days=int(self.var t days.get()) -
int(self.var absent.get())
            sal = (per day) * (work days)
            deduct=int(self.var medical.get())+int(self.var pf.get())
            addition=int(self.var convence.get())
            net sal=(sal -deduct) + (addition)
            self.var net salary.set(str(round(net sal,2)))
    #=======UPDATE THE RECEIPT===================================
            new_sample=f'''\tINTERNATIONAL SCHOOL\n Address:B-12
Kamlanagar, Floor-4
Employee ID\t\t: {self.var e id.get()}
Employee Name\t\t: {self.var name.get()}
Emp. Designation\t\t: {self.var designation.get()}
Salary Of \t\t: {self.var month.get()}-{self.var year.get()}
Generated On\t\t: {str(time.strftime("%d-%m-%y"))}
Total Days\t\t: {self.var t days.get()}
Total Present\t\t: {str(int(self.var t days.get())-
int(self.var absent.get()))}
Total Absent\t\t: {self.var absent.get()}
Convence\t\t: Rs. {self.var convence.get()}
Medical\t\t : Rs. {self.var medical.get()}
PF \t\t : Rs. {self.var pf.get()}
Gross Payment\t\t: Rs. {self.var salary.get()}
Net Salary\t\t: Rs. {self.var net salary.get()}
```

```
This is computer genarated slip,
not required any signature
. . .
            self.txt salary receipt.delete('1.0',END)
            self.txt salary receipt.insert(END, new sample)
    def save(self):
        if (self.var_e_id.get() == '' or self.var_name.get() == '' or
self.var designation.get() == ""):
            messagebox.showerror("Error", "ALL EMPLOYEE DETAILS ARE
REQUIRED")
        else:
            try:
                #connection=mysql.connector.connect(host='localhost',
user='root', #password='kaushik@')
                #c=connection.cursor()
                #query 1="CREATE DATABASE project"
                #c.execute(query 1)
                #con.commit()
                #print("DATABASE CREATED")
                #connection=mysql.connector.connect(host='localhost',
user='root', #password='kaushik@',db='project')
                #c=connection.cursor()
                #query 2=("CREATE TABLE employee data (E ID INT(10) NOT
NULL PRIMARY KEY, DESIGNATION TINYTEXT, NAME TINYTEXT, AGE TINYTEXT, GENDER
TINYTEXT, EMAIL TINYTEXT, HR LOCATION TINYTEXT, DOJ TINYTEXT, DOB
```

```
TINYTEXT, EXPERIENCE TINYTEXT, PROOF ID TINYTEXT, CONTACT TINYTEXT, STATUS
TINYTEXT, ADDRESS TINYTEXT, MONTH TINYTEXT, YEAR TINYTEXT, BASIC SALARY
TINYTEXT, T DAYS TINYTEXT, ABSENT DAYS TINYTEXT, MEDICAL TINYTEXT, PF
TINYTEXT, CONVENCE TINYTEXT, NET SALARY TINYTEXT, SALARY RECEIPT TEXT)")
                #c.execute(query 2)
                #connection.commit
                #print("Table Created")
                connection=mysql.connector.connect(host='localhost',
user='root',
password='kaushik@',db='project')
                c=connection.cursor()
                c.execute("SELECT * FROM employee_data WHERE
e id=%s",(str(self.var_e_id.get()),))
                row=c.fetchone()
                if row!=None:
                    messagebox.showerror("ERROR","THIS EMPLOYEE ID IS
ALREADY AVAILABLE IN OUR RECORDS")
                else:
                    e id=self.var e id.get()
                    designation=self.var designation.get()
                    name=self.var name.get()
                    age=self.var age.get()
                    gender=self.var gender.get()
                    email=self.var email.get()
                    location=self.var hr location.get()
                    doj=self.var_doj.get()
                    dob=self.var dob.get()
                    experience=self.var experience.get()
                    proof=self.var proof id.get()
                    contact=self.var contact.get()
                    status=self.var status.get()
                    address=self.var address.get()
```

```
month=self.var month.get()
                   year=self.var year.get()
                   basic=self.var salary.get()
                   total=self.var_t_days.get()
                   absent=self.var absent.get()
                   medical=self.var medical.get()
                   pf=self.var pf.get()
                   convence=self.var convence.get()
                   net=self.var net salary.get()
                   #receipt=self.var e id.get().txt
receipt="Salary receipt/"+str(self.var e id.get()+'.txt')
                   insert_query= "INSERT INTO employee data
(e id, designation, name, age, gender, email, hr location, doj, dob, experience, proo
f id, contact, status, address, month, year, basic salary, t days, absent days, medi
cal,pf,convence,net_salary,salary_receipt) VALUES
vals=(e id,designation,name,age,gender,email,location,doj,dob,experience,pr
oof, contact, status, address, month, year, basic, total, absent, medical, pf, convenc
e, net, receipt)
                   c.execute(insert query, vals)
                   connection.commit()
                   connection.close()
file=open("Salary_receipt/"+str(self.var e id.get()+'.txt'),'w')
                   file.write(self.txt salary receipt.get('1.0',END))
                   file.close()
```

```
messagebox.showinfo("Success", "RECORD ADDED
SUCCESSFULLY")
            except Exception as ex:
                messagebox.showerror("Error",f'Error due to:{str(ex)}')
    def update(self):
        if self.var e id.get() =="" and self.var name.get() == '':
            messagebox.showerror("ERROR","EMPLOYEE ID AND NAME ARE
REQUIRED")
        else:
            try:
                connection=mysql.connector.connect(host='localhost',
user='root',
password='kaushik@',db='project')
                c=connection.cursor()
                #c.execute("update employee data
set=",(str(self.var e id.get()),))
                c.execute("SELECT * FROM employee data WHERE
e id=%s",(str(self.var e id.get()),))
                row=c.fetchone()
                if row==None:
                    messagebox.showerror("ERROR","THIS EMPLOYEE ID IS
INVALID, TRY AGAIN WITH ANOTHER ID")
                else:
                    e id=self.var e id.get()
                    designation=self.var designation.get()
                    name=self.var name.get()
```

```
age=self.var age.get()
                     gender=self.var gender.get()
                     email=self.var email.get()
                     location=self.var hr location.get()
                     doj=self.var doj.get()
                     dob=self.var dob.get()
                     experience=self.var experience.get()
                     proof=self.var proof id.get()
                     contact=self.var contact.get()
                     status=self.var status.get()
                     address=self.var address.get()
                    month=self.var month.get()
                     year=self.var year.get()
                     basic=self.var salary.get()
                     total=self.var t days.get()
                     absent=self.var absent.get()
                     medical=self.var medical.get()
                     pf=self.var pf.get()
                     convence=self.var convence.get()
                     net=self.var net salary.get()
                     receipt=str(self.var e id.get()) + ".txt"
                     update query=("UPDATE employee data SET
designation=%s,name=%s,age=%s,gender=%s,email=%s,hr loaction=%s,doj=%s,dob=
%s,experience=%s,proof id=%s,contact=%s,status=%s,address=%s,month=%s,year=
%s, basic salary=%s, t days=%s, absent days=%s, medical=%s, pf=%s, convence=%s, ne
t salary=%s, salary receipt=%s WHERE e id=%s")
vals=(designation, name, age, gender, email, location, doj, dob, experience, proof, c
ontact, status, address, month, year, basic, total, absent, medical, pf, convence, net
,e id)
                     c.execute(update query, vals)
                     connection.commit()
```

```
#connect.close()
                    file=open('Salary receipt/'+receipt,'w')
                    file.write(self.txt_salary_receipt.get('1.0',END))
                    file.close()
                    messagebox.showinfo("Success", "RECORD UPDATED
SUCCESSFULLY")
            except Exception as ex:
                messagebox.showerror("Error",f'Error due to:{str(ex)}')
    def show(self):
        try:
            connection=mysql.connector.connect(host='localhost',
user='root',
password='kaushik@',db='project')
            c=connection.cursor()
            c.execute("SELECT * FROM employee data")
            row=c.fetchall()
            self.employee_tree.delete(*self.employee_tree.get_children())
            for row in row:
                self.employee tree.insert('',END,values=row)
        except Exception as ex:
            messagebox.showerror("ERROR",f'ERROR DUE TO:{str(ex)}')
```

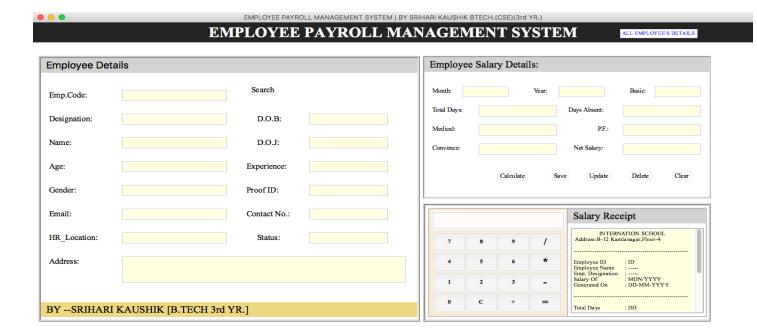
```
def employee frame(self):
        self.root2=Toplevel(self.root)
        self.root2.title("EMPLOYEE PAYROLL MANAGEMENT SYSTEM | BY SRIHARI
KAUSHIK BTECH. (CSE) (3rd YR.)")
        self.root2.geometry("900x500+200+60")
        self.root2.config(bg="white")
        title=Label(self.root2,text="Employee Payroll Management
System", font=('times new roman', 30, 'bold'), bg='lightblue', fg='black')
        title.pack(side=TOP, fill=X)
        self.root2.focus force()
        scrolly=Scrollbar(self.root2,orient=VERTICAL)
        scrolly.pack(side=RIGHT,fill=Y)
        scrollx=Scrollbar(self.root2, orient=HORIZONTAL)
        scrollx.pack(side=BOTTOM, fill=X)
self.employee tree=ttk.Treeview(self.root2,column=('e id','designation','na
me','age','gender','email','hr location','doj','dob','experience',
'proof id', 'contact', 'status', 'address', 'month', 'year', 'basic salary', 't da
ys', 'absent days', 'medical',
'pf','convence','net_salary','salary_receipt'),
                                         yscrollcommand=scrolly.set,
                                         xscrollcommand=scrollx.set)
        self.employee tree.heading('e id',text='EMPLOYEE ID')
        self.employee tree.heading('designation',text='DESIGNATION')
        self.employee tree.heading('name',text='NAME')
```

```
self.employee tree.heading('age',text='AGE')
self.employee tree.heading('gender',text='GENDER')
self.employee tree.heading('email',text='EMAIL')
self.employee tree.heading('hr location',text='HR LOCATION')
self.employee tree.heading('doj',text='DOJ')
self.employee tree.heading('dob',text='DOB')
self.employee tree.heading('experience',text='EXPERIENCE')
self.employee tree.heading('proof id',text='PROOF ID')
self.employee tree.heading('contact',text='CONTACT')
self.employee tree.heading('status',text='STATUS')
self.employee tree.heading('address',text='ADDRESS')
self.employee tree.heading('month',text='MONTH')
self.employee tree.heading('year',text='YEAR')
self.employee tree.heading('basic salary',text='BASIC SALARY')
self.employee tree.heading('t days',text='TOTAL DAYS')
self.employee tree.heading('absent days',text='ABSENT DAYS')
self.employee tree.heading('medical',text='MEDICAL')
self.employee tree.heading('pf',text='PF')
self.employee tree.heading('convence',text='CONVENCE')
self.employee tree.heading('net salary',text='NET SALARY')
self.employee tree.heading('salary receipt',text='SALARY RECEIPT')
self.employee tree ['show']='headings'
```

```
self.employee_tree.column('e_id',width=100)
self.employee_tree.column('designation',width=100)
self.employee_tree.column('name',width=200)
self.employee_tree.column('age',width=100)
self.employee_tree.column('gender',width=100)
self.employee_tree.column('email',width=150)
self.employee_tree.column('hr_location',width=100)
self.employee_tree.column('doj',width=100)
self.employee_tree.column('dob',width=100)
self.employee_tree.column('experience',width=100)
self.employee_tree.column('proof_id',width=100)
```

```
self.employee tree.column('contact', width=100)
        self.employee tree.column('status', width=100)
        self.employee tree.column('address', width=300)
        self.employee tree.column('month', width=100)
        self.employee tree.column('year', width=100)
        self.employee tree.column('basic salary', width=100)
        self.employee tree.column('t days',width=100)
        self.employee tree.column('absent days', width=100)
        self.employee tree.column('medical', width=100)
        self.employee tree.column('pf', width=100)
        self.employee tree.column('convence', width=100)
        self.employee tree.column('net salary', width=100)
        self.employee tree.column('salary receipt', width=100)
        scrollx.config(command=self.employee tree.xview)
        scrolly.config(command=self.employee tree.yview)
        self.employee tree.pack(fill=BOTH,expand=1)
        self.show()
        self.root2.mainloop()
root=Tk()
obj= EmployeeSystem(root)
root.mainloop()
```

OUTPUTS

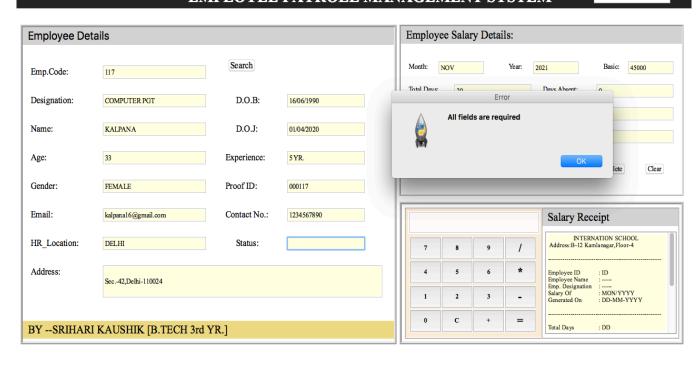


mysql> DESC employee_data;

	+	+	+	+	+
Field	Туре	Null	Key	Default	Extra
E_ID	int(10)	NO	PRI	NULL	+
DESIGNATION	tinytext	YES	j i	NULL	İ
NAME	tinytext	YES	j i	NULL	İ
AGE	tinytext	YES	ĺ	NULL	ĺ
GENDER	tinytext	YES		NULL	
EMAIL	tinytext	YES		NULL	l
HR_LOCATION	tinytext	YES		NULL	l
DOJ	tinytext	YES		NULL	l
DOB	tinytext	YES		NULL	l
EXPERIENCE	tinytext	YES		NULL	
PROOF_ID	tinytext	YES		NULL	l
CONTACT	tinytext	YES		NULL	
STATUS	tinytext	YES		NULL	
ADDRESS	tinytext	YES		NULL	
MONTH	tinytext	YES		NULL	
YEAR	tinytext	YES		NULL	
BASIC_SALARY	tinytext	YES		NULL	
T_DAYS	tinytext	YES		NULL	
ABSENT_DAYS	tinytext	YES		NULL	
MEDICAL	tinytext	YES		NULL	
PF	tinytext	YES		NULL	
CONVENCE	tinytext	YES		NULL	
NET_SALARY	tinytext	YES		NULL	
SALARY_RECEIPT	text	YES		NULL	
	+	+	+		+

EMPLOYEE PAYROLL MANAGEMENT SYSTEM

LL EMPLOYEE'S DETAILS

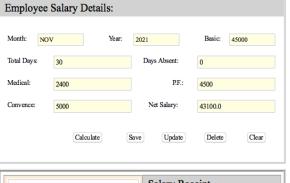


EMPLOYEE PAYROLL MANAGEMENT SYSTEM | BY SRIHARI KAUSHIK BTECH.(CSE)(3rd YR.)

EMPLOYEE PAYROLL MANAGEMENT SYSTEM

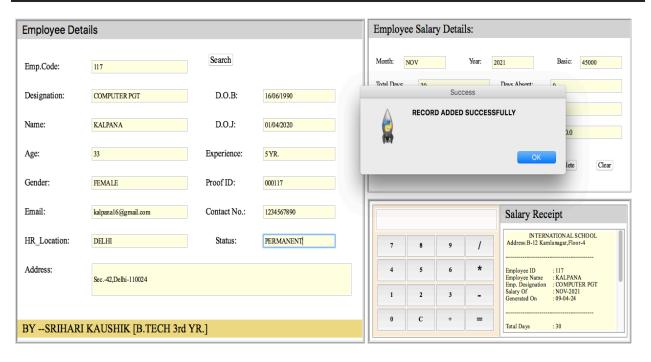
ALL EMPLOYEE'S DETAILS







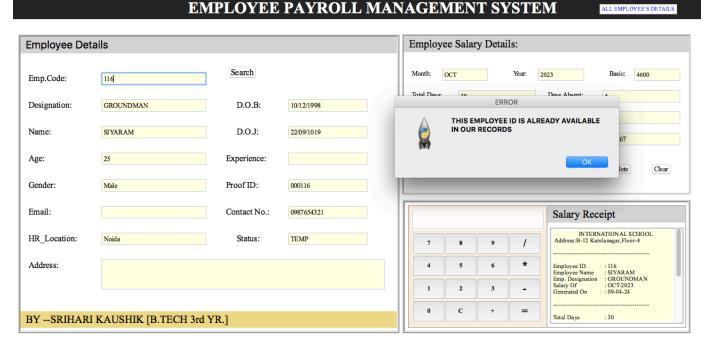
© © © EMPLOYEE PAYROLL MANAGEMENT SYSTEM | BY SRIHARI KAUSHIK BTECH, (CSE) (3rd YR.) EMPLOYEE PAYROLL MANAGEMENT SYSTEM ALLEMPLOYEE'S DETAILS.



EMPLOYEE PAYROLL MANAGEMENT SYSTEM | BY SRIHARI KAUSHIK BTECH.(CSE)(3rd YR.) EMPLOYEE PAYROLL MANAGEMENT SYSTEM **Employee Details** Employee Salary Details: Month: JAN 2022 Basic: 35000 Emp.Code: 118 Designation: D.O.B: 11/10/1995 RECORD ADDED SUCCESSFULLY D.O.J: Name: MUKUL 01/04/2020 Experience: Age: Gender: MALE Proof ID: 000118 Email: mukul11@gmail.com Contact No.: 0987654321 Salary Receipt HR Location: DELHI Status: PROBATION Total Days Total Present Total Absent Convence Medical 1 : 8 : Rs. 5000 : Rs. 2400 : Rs. 4500 : Rs. 35000 : Rs. 23766.67 * Address: 6 Sec.-56,Delhi-110065 2 3 0 C = BY -- SRIHARI KAUSHIK [B.TECH 3rd YR.]

EMPLOYEE PAYROLL MANAGEMENT SYSTEM | BY SRIHARI KAUSHIK BTECH.(CSE)(3rd YR.)

EMPLOYEE PAYROLL MANAGEMENT SYSTEM



EMPLOYEE PAYROLL MANAGEMENT SYSTEM | BY SRIHARI KAUSHIK BTECH.(CSE)(3rd YR.) EMPLOYEE PAYROLL MANAGEMENT SYSTEM ALL EMPLOYEE'S DETAILS **Employee Details** Employee Salary Details: Search Month: Emp.Code: Dave Absent Total Dave Error D.O.B: Designation: YOGA COACH ALL EMPLOYEE DETAILS ARE REQUIRED Name: ASHA D.O.J: Age: Experience: Clear Proof ID: Gender: Email: Contact No .: Salary Receipt INTERNATION SCHOOL Address:B-12 Kamlanagar,Floor-4 HR_Location: Status: Address: Employee ID Employee Name Emp. Designation Salary Of Generated On 6 : ID MON/YYYY DD-MM-YYYY 2 3 C Total Days BY --SRIHARI KAUSHIK [B.TECH 3rd YR.]



118.txt — Edited ~

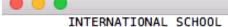
INTERNATIONAL SCHOOL Address:B-12 Kamlanagar

Employee ID : 117
Employee Name : KALPANA
Emp. Designation : COMPUTER PGT
Salary Of : NOV-2021
Generated On : 09-04-24

Total Days Total Present
Total Absent
Convence

: 30 : 30 : 0 : Rs. 5000 : Rs. 2400 Convence Medical PF : Rs. 4500 Gross Payment : Rs. 45000 Net Salary : Rs. 43100.0

This is computer generated slip, not required any signature



Address:B-12 Kamlanagar

Employee ID : 118
Employee Name : MUKUL
Emp. Designation : ENGLISH PGT
Salary Of : JAN-2022
Generated On : 09-04-24

: 30 : 22 : 8 : Rs. 5000 Total Days Total Days
Total Present
Total Absent Total Absent

Convence Medical : Rs. 2400 PF : Rs. 4500 Gross Payment : Rs. 35000 Net Salary : Rs. 23766.67

This is computer generated slip, not required any signature

EMPLOYEE PAYROLL MANAGEMENT SYSTEM

Employee Deta	ils	Employee	e Salary D	etails:			
Emp.Code:	Search	Month:		Year:		Basic:	
Designation:	D.O.B:	Total Days: Medical:			Days Absent:		
Name:	D.O.J:	Convence:			Net Salary:		
Age:	Experience:		Calc	ulate	Save Update	Delete	Clear
Gender:	Proof ID:						
Email:	Contact No.:	Salary Receipt					
HR_Location:	Status:	7	8	9 /	Address:B-12 Kar	INTERNATION SCHOOL Address:B-12 Kamlanagar,Floor-4	
Address:		4	5	6 *	Employee ID Employee Name Emp. Designation	: ID :	
		1	2	3 -	Salary Of Generated On	: MON/YYYY : DD-MM-YYYY	
BYSRIHARI I	KAUSHIK [B.TECH 3rd YR.]	0	С	+ =	Total Days	: DD	

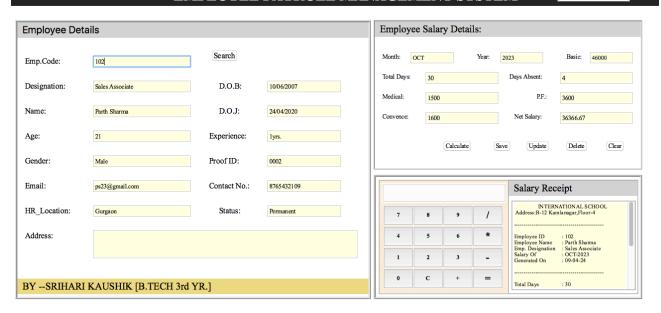
7	8	9	1
4	5	6	*
1	2	3	-
0	С	+	=

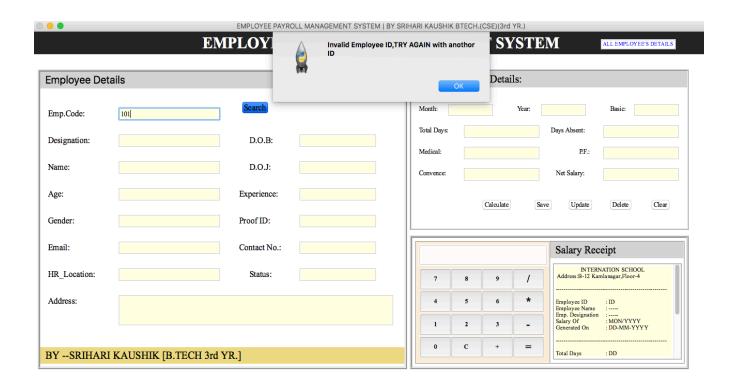


6			
7	8	9	/
4	5	6	*
1	2	3	-
0	C	+	=



EMPLOYEE PAYROLL MANAGEMENT SYSTEM

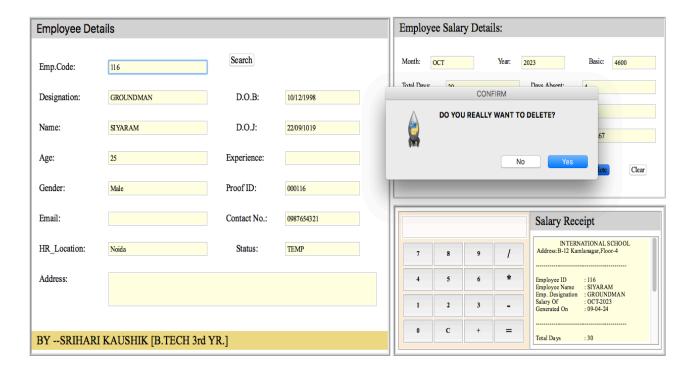




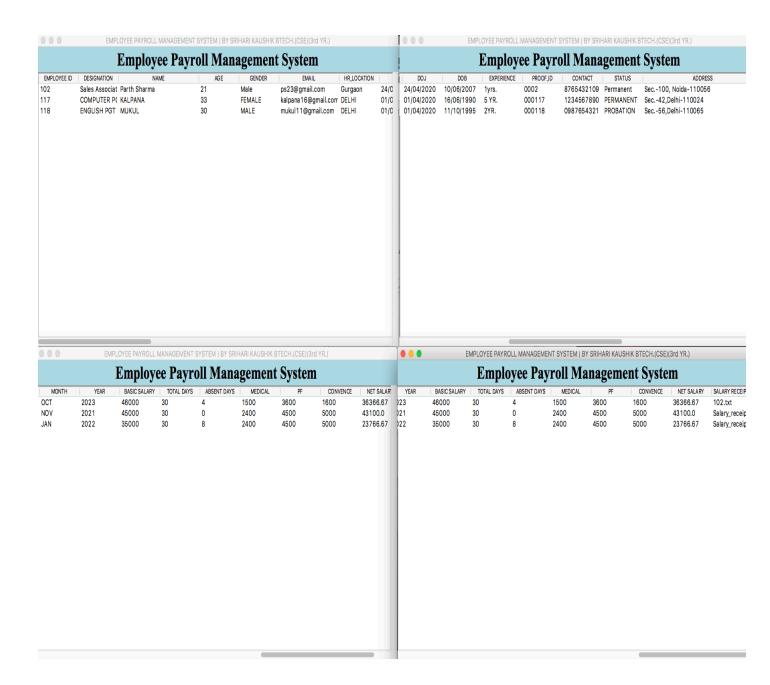
EMPLOYEE PAYROLL MANAGEMENT SYSTEM | BY SRIHARI KAUSHIK BTECH.(CSE)(3rd YR.)

000

EMPLOYEE PAYROLL MANAGEMENT SYSTEM



EMPLOYEE PAYROLL MANAGEMENT SYSTEM | BY SRIHARI KAUSHIK BTECH.(CSE)(3rd YR.) EMPLOYEE PAYROLL MANAGEMENT SYSTEM ALL EMPLOYEE'S DETAILS **Employee Details** Employee Salary Details: Search Month: OCT 2023 Emp.Code: Total Days SUCCESS Designation: GROUNDMAN D.O.B: 10/12/1998 DATA DELETED SUCCESSFULLY D.O.J: 22/09/1019 Name: SIYARAM Experience: Age: Clear Proof ID: Gender: 000116 Email: Contact No.: 0987654321 Salary Receipt INTERNATIONAL SCHOOL Address:B-12 Kamlanagar,Floor-4 HR_Location: Employee ID Employee Name Emp. Designation Salary Of Generated On Address: : 116 : SIYARAM : GROUNDMAN : OCT-2023 : 09-04-24 2 3 C BY --SRIHARI KAUSHIK [B.TECH 3rd YR.]



CONCLUTION

In conclusion, this employee payroll system offers an efficient solution for managing employee data and payroll processing. By leveraging Python and MySQL, it provides a user-friendly interface for storing employee information, calculating salaries, and generating pay slips. This system streamlines HR tasks, making it easier to maintain records, update information, and ensure accurate compensation for employees.

The code's modular design allows for easy customization and scalability, making it adaptable to various organizational needs. It showcases the power of open-source technologies in building robust business applications.

As startups and small organizations strive to enhance efficiency and reduce manual processes, this payroll system serves as a valuable tool. It not only simplifies HR operations but also minimizes errors, ensuring that employees are compensated accurately and on time.

In the ever-evolving world of business and technology, tools like this employee payroll system play a vital role in supporting modern HR practices and contributing to the success of organizations.

FUTURE SCOPE

- 1. Integration with External Systems: One potential future enhancement is to integrate the payroll system with other HR and finance software, such as attendance management, tax calculation, and accounting systems. This integration can streamline data flow, reducing manual data entry and potential errors.
- 2. Security Enhancements: Implement advanced security features to protect sensitive employee data. This may involve role-based access control and encryption to ensure data privacy and compliance with data protection regulations.
- 3. Mobile Application: Develop a mobile application to allow HR personnel and employees to access and update their information on the go. This can improve accessibility and convenience.
- 4. Cloud-Based Deployment: Offer a cloud-based version of the software, allowing organizations to access and manage payroll data from anywhere, while also facilitating automatic updates and backups.
- 5. Localization and Multilingual Support: Extend the system to support multiple languages and comply with local tax and labor laws in different regions, making it suitable for global organizations.
- # As businesses evolve and technology advances, this payroll system can stay relevant and valuable by incorporating these and some other future enhancements to meet the growing demands of modern HR and payroll management.

ADVANTAGES

- 1. Efficient Employee Data Management: The code allows for the centralized and organized storage of employee details, making it easy to update and retrieve information. This efficiency reduces the administrative burden on HR departments.
- 2. Accurate Salary Calculation: The code automates the process of salary calculation, reducing the chances of errors. This leads to precise salary disbursements and minimizes disputes.
- 3. Streamlined Payroll Processing: With this code, generating salary receipts becomes a straightforward task. This streamlining of payroll processing saves time and ensures that employees are paid on time.
- 4. User-Friendly Interface: The system offers a user-friendly interface that simplifies interactions for HR personnel and administrators. This means minimal training is required for users to operate the system effectively.
- 5. Customization: The code can be customized to fit the specific needs and requirements of an organization. This adaptability ensures it aligns with unique payroll processes.

BIBLIOGRAPHY

- . Computer Science with Preeti Arora
- . www.stackoverflow.com
- . www.geeksforfeeks.org
- . www.github.com
- . www.youtube.com
- .www.useblackbox.io