Salary Management System

Veda Pattar, Fiza Taskeen N, Deepa U S

Department of Electronics & Communication Engineering

TONTADARYA COLLEGE OF ENGINEERING Mundaragi Road gadag

Corresponding Author: Lohit javali

Abstract - Small and Medium-sized Enterprises (SMEs) in the process of wage entry, calculation, the total number are needed to be done manually in the past, the data volume is quite large, processing speed is low, and it is easy to make error, which is resulting in low efficiency. The main purpose of writing this paper is to present the basis of salary management system, establish a scientific database, the computer payroll system, using the computer instead of a lot of past manual work in order to reduce duplication of staff labor, it will improve working efficiency. This system combines the actual needs of SMEs, through in-depth study and practice of the C/S mode, PowerBuilder10.0 development tools, databases and SQL language, Completed a payroll system needs analysis, database design, application design and development work. Wages, departments, units and personnel database file are included in this system, and have data management, department management, personnel management and other functions, through the control and management of the database query, add, delete, modify, and other functions can be realized. This system is reasonable design, a more complete function, stable operation has been tested .

***Key Word*:** SMEs

1.INTRODUCTION

The basic objective behind developing this project “SALARY MANAGEMENT SYSTEM” is as follows:

- To store basic information regarding employees of the organization.
- To store salary information of employees such as working hours, salary per hour, salary before tax, tax percentage, total amount of tax paid, salary after tax, on monthly basis.
- To keep record of salary to be paid to employee grade wise.
- To calculate the employee salary with leave and without leave.
- Salary slips can be sent to the employees upon request.

2. The basic objective behind developing this project “SALARY MANAGEMENT SYSTEM” is as follows:

- To store basic information regarding employees of the organization.
- To store salary information of employees such as working hours, salary per hour, salary before tax, tax percentage, total amount of tax paid, salary after tax, on monthly basis.

The basic objective behind developing this project "SALARY MANAGEMENT SYSTEM" is as follows:

- To store basic information regarding employees of the organization.
- To store salary information of employees such as working hours, salary per hour, salary before tax, tax percentage, total amount of tax paid, salary after tax, on monthly basis.
- To keep record of salary to be paid to employee grade wise.
- To calculate the employee salary with leave and without leave.
- Salary slips can be sent to the employees upon request.

REQUIREMENT ANALYSIS

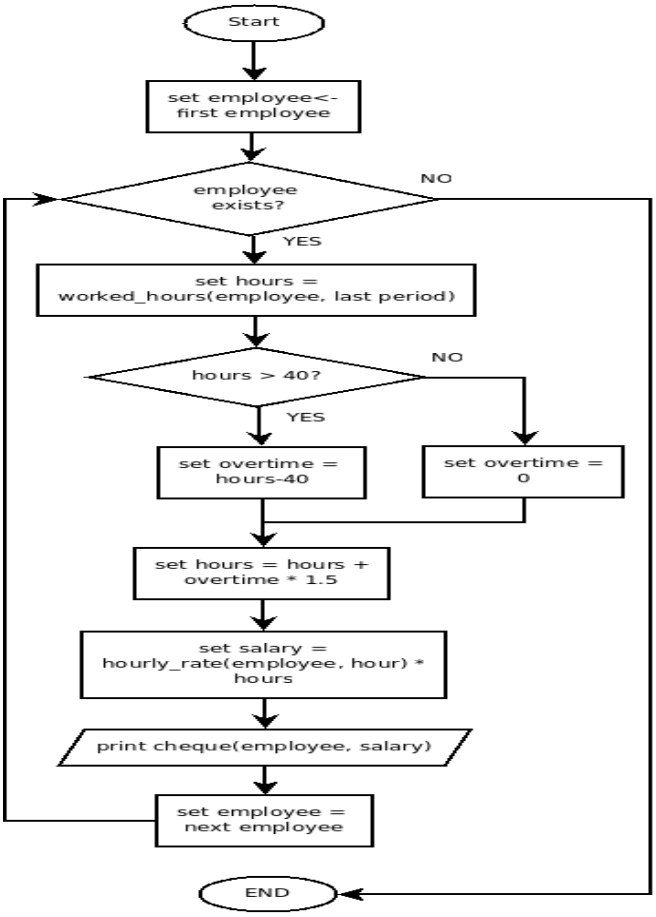
Software Requirements:

Operating system: Windows 10.
Programming language: Python.
Tool: Pycharm and Tkinter platform.

Hardware Requirements:

Processor : Intel(R) Core (TM)i5-8250U CPU
CPU clock :1.60GHz
Main Memory :8.00 GB
Secondary Storage:120GB

FLOW CHAT:



SOURCE CODE :

```
import time
import tkinter.messagebox
from tkinter import *
root = Tk()
root.title("Salary Management System")
root.geometry('1370x720+0+0')
root.maxsize(width=1370, height=720)
root.minsize(width=1370, height=720)
root.configure(background="dark gray")
Tops = Frame(root, width=1350, height=50, bd=8,
bg="dark blue")
Tops.pack(side=TOP)
f1 = Frame(root, width=600, height=600, bd=8,
bg="dark gray")
f1.pack(side=LEFT)
f2 = Frame(root, width=300, height=700, bd=8,
bg="dark blue")
```

```
f2.pack(side=RIGHT)
fla = Frame(f1, width=600, height=200, bd=8,
bg="dark blue")
fla.pack(side=TOP)
flb = Frame(f1, width=300, height=600, bd=8,
bg="dark blue")
flb.pack(side=TOP)
lbl_information = Label(Tops, font=('arial', 45,
'bold'), text="Salary Management System",
relief=GROOVE, bd=10, bg="Dark Gray",
fg="Black")
lbl_information.grid(row=0, column=0)
def Exit():
    wayOut =
tkinter.messagebox.askyesno("Salary Management
System", "Do you want to exit the system")
    if wayOut > 0:
        root.destroy()
        return
def Reset():
    FullName.set("")
    Address.set("")
    Wrk_Hrs.set("")
    Hrs_Wage.set("")
    Payable.set("")
    Taxable.set("")
    NetPayable.set("")
    GrossPayable.set("")
    OverTimeBonus.set("")
    CompanyAgency.set("")
    PhoneNumber.set("")
    txtPaymentSlip.delete("1.0", END)
def InformationEntry():
    txtPaymentSlip.delete("1.0", END)
    txtPaymentSlip.insert(END, "\t\tPay
Slip\n\n")
    txtPaymentSlip.insert(END, "Full
Name:\t\t" + FullName.get()+"\n\n")
    txtPaymentSlip.insert(END, "Home
Address:\t\t" + Address.get()+"\n\n")
    txtPaymentSlip.insert(END,
"Company/Agency:\t\t" +
CompanyAgency.get()+"\n\n")
    txtPaymentSlip.insert(END, "Phone
Number:\t\t" + PhoneNumber.get()+"\n\n")
    txtPaymentSlip.insert(END, "Hours
Worked:\t\t" + Wrk_Hrs.get()+"\n\n")
    txtPaymentSlip.insert(END, "Net
Payable:\t\t" + NetPayable.get()+"\n\n")
    txtPaymentSlip.insert(END, "Wages per
hour:\t\t" + Hrs_Wage.get()+"\n\n")
    txtPaymentSlip.insert(END, "Tax Paid:\t\t"
+ Taxable.get()+"\n\n")
    txtPaymentSlip.insert(END, "Payable:\t\t" +
Payable.get()+"\n\n")
def WagesForWeekly():
    txtPaymentSlip.delete("1.0", END)
    hrs_wrk_per_wek = float(Wrk_Hrs.get())
    hrs_per_wgs = float(Hrs_Wage.get())
    DuePayment=hrs_per_wgs *
hrs_wrk_per_wek
    PaymentDue="P" + str('%.2f'
%DuePayment)
    Payable.set(PaymentDue)
    tax = DuePayment * 0.12
    taxable = "P" + str('%.2f'%tax)
    Taxable.set(taxable)
    PaymentNet = DuePayment - tax
    NetPayments = "P" + str('%.2f'
%PaymentNet)
    NetPayable.set(NetPayments)
    if hrs_wrk_per_wek > 40:
```

```
            HoursTimeOver =
(hrs_wrk_per_wek - 40) + hrs_per_wgs * 1.5
            OverTime = "P" + str('%.2f'
%HoursTimeOver)
            OverTimeBonus.set(OverTime)
        elif hrs_wrk_per_wek <= 40:
            PaymentOverTime = 0
            HoursOverTime = "P" + str('%.2f'
%PaymentOverTime)

        OverTimeBonus.set(HoursOverTime)
        return

# Variables
FullName = StringVar()
Address = StringVar()
Hrs_Wage = StringVar()
Wrk_Hrs = StringVar()
Payable = StringVar()
Taxable = StringVar()
NetPayable = StringVar()
GrossPayable = StringVar()
OverTimeBonus = StringVar()
CompanyAgency = StringVar()
PhoneNumber = StringVar()
TimeOfOrder = StringVar()
DateOfOrder = StringVar()
DateOfOrder.set(time.strftime("%d/%m/%Y"))
# Label Widget
labelFirstName = Label(fla, text="Full Name", font
= ('arial', 16, 'bold'), bd=20, fg="white", bg="dark
blue").grid(row=0, column=0)
labelAddress = Label(fla, text="Home Address",
font = ('arial', 16, 'bold'), bd=20, fg="white",
bg="dark blue").grid(row=0, column=2)
labelCompanyAgency = Label(fla,
text="Company/Agency", font=('arial', 16, 'bold'),
bd=20, fg="white", bg="dark blue").grid(row=1,
column=0)
labelPhoneNumber = Label(fla, text="Phone
Number", font=('arial', 16, 'bold'), bd=20,
fg="white", bg="dark blue").grid(row=1,
column=2)
labelHoursWorked = Label(fla, text="Hours
Worked", font=('arial', 16, 'bold'), bd=20,
fg="white", bg="dark blue").grid(row=2,
column=0)
labelHourlyRate = Label(fla, text="Hourly Rate",
font=('arial', 16, 'bold'), bd=20, fg="white",
bg="dark blue").grid(row=2, column=2)
labelTax = Label(fla, text="Tax", font=('arial', 16,
'bold'), bd=20, fg="white", bg="dark
blue").grid(row=3, column=0)

labelOverTime = Label(fla, text="Over Time",
font=('arial', 16, 'bold'), bd=20, fg="white",
bg="dark blue").grid(row=3, column=2)
labelGrossPay = Label(fla, text="Gross Pay",
font=('arial', 16, 'bold'), bd=20, fg="white",
bg="dark blue").grid(row=4, column=0)
labelNetPay = Label(fla, text="Net Pay",
font=('arial', 16, 'bold'), bd=20, fg="white",
bg="dark blue").grid(row=4, column=2)
# Entry Widget
txtFullname = Entry(fla, textvariable=FullName,
font=('arial', 16, 'bold'), bd=16, width=22,
justify='left')
txtFullname.grid(row=0, column=1)
txtAddress = Entry(fla, textvariable=Address,
font=('arial', 16, 'bold'), bd=16, width=22,
justify='left')
txtAddress.grid(row=0, column=3)
```

```
txtCompanyAgency = Entry(fla,
textvariable=CompanyAgency, font=('arial', 16,
'bold'), bd=16, width=22, justify='left')
txtCompanyAgency.grid(row=1, column=1)
txtWrk_hrs = Entry(fla, textvariable=Wrk_Hrs,
font=('arial', 16, 'bold'), bd=16, width=22,
justify='left')
txtWrk_hrs.grid(row=2, column=1)
txtHrs_Wages = Entry(fla, textvariable=Hrs_Wage,
font=('arial', 16, 'bold'), bd=16, width=22,
justify='left')
txtHrs_Wages.grid(row=2, column=3)
txtPhoneNumber = Entry(fla,
textvariable=PhoneNumber, font=('arial', 16,
'bold'), bd=16, width=22, justify='left')
txtPhoneNumber.grid(row=1, column=3)
txtGrossPayment = Entry(fla, textvariable=Payable,
font=('arial', 16, 'bold'), bd=16, width=22,
justify='left')
txtGrossPayment.grid(row=4, column=1)
txtNetPayable = Entry(fla,
textvariable=NetPayable, font=('arial', 16, 'bold'),
bd=16, width=22, justify='left')
txtNetPayable.grid(row=4, column=3)

txtTaxable = Entry(fla, textvariable=Taxable,
font=('arial', 16, 'bold'), bd=16, width=22,
justify='left')
txtTaxable.grid(row=3, column=1)

txtOverTimeBonus = Entry(fla,
textvariable=OverTimeBonus, font=('arial', 16,
'bold'), bd=16, width=22, justify='left')
txtOverTimeBonus.grid(row=3, column=3)


# Text Widget

payslip = Label(f2, textvariable=DateOfOrder,
font=('arial', 21, 'bold'), fg="white", bg="dark
blue").grid(row=0, column=0)

txtPaymentSlip = Text(f2, height=22, width=34,
bd=16, font=('arial', 13, 'bold'), fg="black",
bg="white")
txtPaymentSlip.grid(row=1, column=0)

# buttons

ButtonSalary = Button(flb, text='WeeklySalary',
padx=16, pady=16, bd=15, font=('arial', 16, 'bold'),
relief="groove", width=14, fg="black", bg="dark
gray",
        command=WagesForWeekly).grid(row=0,
column=0)


ButtonReset = Button(flb, text='Reset', padx=16,
pady=16, bd=15, font=('arial', 16, 'bold'),
relief="groove", width=14, command=Reset,
fg="black",
        bg="dark gray").grid(row=0, column=1)


ButtonPaySlip = Button(flb, text='View Payslip',
padx=16, pady=16, bd=15, font=('arial', 16, 'bold'),
relief="groove", width=14,
command=InformationEntry,
        fg="black" ,bg="dark gray").grid(row=0,
column=2)
```

```
ButtonExit = Button(flb, text='Exit System',
padx=16, pady=16, bd=15, font=('arial', 16, 'bold'),
relief="groove", width=14, command=Exit,
        fg="black", bg="dark gray").grid(row=0,
column=3)

root.mainloop()
```

RESULT:

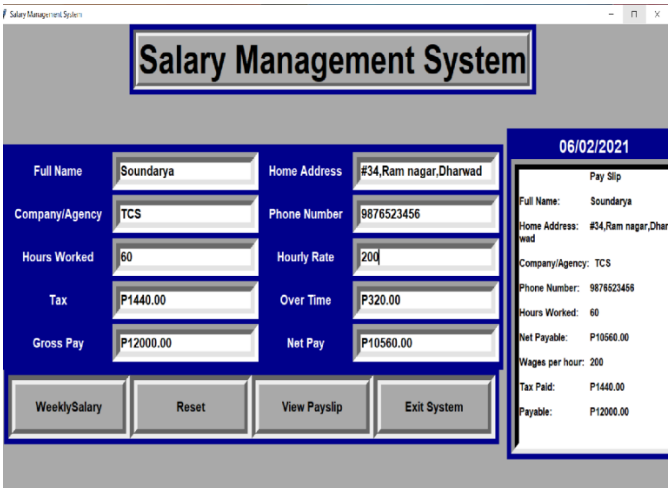


Fig 4.1: Shows the payment slip of an employee where 12% of tax, gross payment, net payment and overtime bonus of 1.5% (hours worked > 40) is calculated

Fig 4.2: Shows the payment slip of an employee where 12% of tax, gross payment, net payment is calculated (overtime bonus is 0 here because hours worked < 40).

CONCLUSIONS

The **SALARY MANAGEMENT SYSTEM** software has been designed to achieve maximum efficiency and reduce the time taken to handle the activity of calculation of salaries. The implementation of the system in the organization will considerably reduce data entry, time and also provide readily calculated reports.

ACKNOWLEDGEMENT

The heading should be treated as a 3rd level heading and should not be assigned a number.

REFERENCES

[1] Think Python by Allen B. Downey. 2nd edition.

[2] Text Book: Automate the Boring Stuff with Python by AI Sweigart.

[3]https://itsourcecode.com/free-projects/python-projects/payroll-management-systemproject-in-python-with-source-