**EMPLOYEE PAYMENT MANAGEMENT SYSTEM USING PYTHON PROGRAMMING**

**PROJECT REPORT**

***BY***

**AADARSH KUMAR PATEL**

**RA1711008020034**

**Department of Information and Technology**

**To**

**Mrs. R. Mythili**

****

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**RAMPURAM CHENNAI 600089 TN**

**April 2020**

**CONTENTS:**

1. Abstract
2. Project Description

* Introduction
* Objective of this project
* Scope of this project

1. Source code
2. System Specification
3. Screenshots
4. Conclusion

**ABSTRACT:**

In this project I have used several kinds of Python feature such as Tkinter+ database(sqlite3) etc. The name of my project is EMPLOYEE PAYMENT MANAGEMENT SYSTEM. I have used PyCharm software to write the code and run on command prompt. There are several kinds of functions available in project like we can add monthly wages, hourly rate, overtime, tax, Gross pay etc. Employee Management System is a python-based project. I have developed Employee Management System using PyCharm and MySQL. The main modules available in this project are Salary module which manages the functionality of Salary, Employee is normally used for managing Employee, Login contains all the functionality related to Login, Information’s manages the Information functionality, Experience has all the features of Experience and Leaves module manages the functionality of Leaves.

As we know Python Projects, are trending topics for academic python project development. So, we had chosen python3 for developing the Employee Management System. In this project we developing features for Salary, Employee, Experience etc, which reduces the human efforts and increase the efficiency.

**Keyword:** PyCharm, MySQL, Tkinter+ etc.

**PROJECT DESCRIPTION:**

**INTRODUCTION:**

Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.

It is used for:

* web development (server-side),
* software development,
* mathematics,
* system scripting.

What can Python do:

* Python can be used on a server to create web applications.
* Python can be used alongside software to create workflows.
* Python can connect to database systems. It can also read and modify files.
* Python can be used to handle big data and perform complex mathematics.
* Python can be used for rapid prototyping, or for production-ready software development.

This text editor GUI will consist of various menu like file and edit, using which all functionalities like saving the file, opening a file, editing, cut and paste can be done.

Now for creating this notepad, Python 3 and Tkinter should already be installed in your system. You can download suitable [python package](https://www.python.org/downloads/) as per system requirement. After you have successfully installed python you need to install Tkinter (a Python’s GUI package).

The [tkinter](https://docs.python.org/3/library/tkinter.html" \l "module-tkinter" \o "tkinter: Interface to Tcl/Tk for graphical user interfaces) package (“Tk interface”) is the standard Python interface to the Tk GUI toolkit. Both Tk and [tkinter](https://docs.python.org/3/library/tkinter.html" \l "module-tkinter" \o "tkinter: Interface to Tcl/Tk for graphical user interfaces) are available on most Unix platforms, as well as on Windows systems. (Tk itself is not part of Python; it is maintained at Active State.)

Running python -m tkinter from the command line should open a window demonstrating a simple Tk interface, letting you know that [tkinter](https://docs.python.org/3/library/tkinter.html" \l "module-tkinter" \o "tkinter: Interface to Tcl/Tk for graphical user interfaces) is properly installed on your system, and also showing what version of Tcl/Tk is installed, so you can read the Tcl/Tk documentation specific to that version. Most of the time, [tkinter](https://docs.python.org/3/library/tkinter.html" \l "module-tkinter" \o "tkinter: Interface to Tcl/Tk for graphical user interfaces) is all you really need, but a number of additional modules are available as well. The Tk interface is located in a binary module named \_tkinter. This module contains the low-level interface to Tk, and should never be used directly by application programmers. It is usually a shared library (or DLL), but might in some cases be statically linked with the Python interpreter.

SQLite is an in-process library that implements a [self-contained](https://www.sqlite.org/selfcontained.html), [serverless](https://www.sqlite.org/serverless.html), [zero-configuration](https://www.sqlite.org/zeroconf.html), [transactional](https://www.sqlite.org/transactional.html) SQL database engine. The code for SQLite is in the [public domain](https://www.sqlite.org/copyright.html) and is thus free for use for any purpose, commercial or private. SQLite is the [most widely deployed](https://www.sqlite.org/mostdeployed.html) database in the world with more applications than we can count, including several [high-profile projects.](https://www.sqlite.org/famous.html)

SQLite is an embedded SQL database engine. Unlike most other SQL databases, SQLite does not have a separate server process. SQLite reads and writes directly to ordinary disk files. A complete SQL database with multiple tables, indices, triggers, and views, is contained in a single disk file. The database [file format](https://www.sqlite.org/fileformat2.html) is cross-platform - you can freely copy a database between 32-bit and 64-bit systems or between [big-endian](http://en.wikipedia.org/wiki/Endianness) and [little-endian](http://en.wikipedia.org/wiki/Endianness) architectures. These features make SQLite a popular choice as an [Application File Format](https://www.sqlite.org/appfileformat.html). SQLite database files are a [recommended storage format](https://www.sqlite.org/locrsf.html) by the US Library of Congress. Think of SQLite not as a replacement for [Oracle](http://www.oracle.com/database/index.html) but as a replacement for [fopen()](http://man.he.net/man3/fopen)

SQLite is a compact library. With all features enabled, the [library size](https://www.sqlite.org/footprint.html) can be less than 600KiB, depending on the target platform and compiler optimization settings. (64-bit code is larger. And some compiler optimizations such as aggressive function in lining and loop unrolling can cause the object code to be much larger.) There is a trade-off between memory usage and speed. SQLite generally runs faster the more memory you give it. Nevertheless, performance is usually quite good even in low-memory environments. Depending on how it is used, SQLite can be [faster than direct filesystem I/O](https://www.sqlite.org/fasterthanfs.html).

**PyCharm** is an [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) used in [computer programming](https://en.wikipedia.org/wiki/Computer_programming), specifically for the [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) language. It is developed by the Czech company [JetBrains](https://en.wikipedia.org/wiki/JetBrains).[[6]](https://en.wikipedia.org/wiki/PyCharm#cite_note-6) It provides code analysis, a graphical debugger, an integrated unit tester, integration with [version control systems](https://en.wikipedia.org/wiki/Revision_control) (VCSes), and supports web development with [Django](https://en.wikipedia.org/wiki/Django_(web_framework)) as well as [Data Science](https://en.wikipedia.org/wiki/Data_science) with [Anaconda](https://en.wikipedia.org/wiki/Anaconda_(Python_distribution)).[[7]](https://en.wikipedia.org/wiki/PyCharm#cite_note-7)

PyCharm is [cross-platform](https://en.wikipedia.org/wiki/Cross-platform), with [Windows](https://en.wikipedia.org/wiki/Windows), [macOS](https://en.wikipedia.org/wiki/MacOS) and [Linux](https://en.wikipedia.org/wiki/Linux) versions. The Community Edition is released under the [Apache License](https://en.wikipedia.org/wiki/Apache_License),[[8]](https://en.wikipedia.org/wiki/PyCharm#cite_note-community-8) and there is also Professional Edition with extra features – released under a [proprietary license](https://en.wikipedia.org/wiki/Proprietary_software).

**OBJECTIVE OF THIS PROJECT:**

* Time and Cost Effective – Software is making the process of keeping database time and cost effective by making it easy to alter or update payment and also keeping that safe.
* Database Creation - A database of the registered users will be created and this will help the users to fetch the details of Computer Center.
* Flexibility – Database provides flexibility to administrator. Records can be inserted, deleted or updated with just a click.
* Informative - The software has all the necessary details about the Students and Courses and provides all the relevant information therefore. For example, searching of appropriate Student for like admission and then the information of admission is added to Students records.
* Security - Proper authorization and authentication provisions have been made for the security of the software so that only the registered administrators can look the Information. Without proper login no one is allowed to access this software.

**SCOPE OF THIS PROJECT:**

For all the s/w there is always a scope of future enhancements. There are few enhancements which are pointed out in the proposed system. They are as follows:

* Module for Employee Loan.
* Modules for proper arrangement of payment of employees.
* Introducing online version of the Software.
* Introducing software for Employees side also.
* More proper salary calculations based on various factors.

**SOURCE CODE:**

import time

import datetime

from tkinter import \*

import tkinter.messagebox

root=Tk()

root.title("Employee payroll system")

root.geometry('1350x650+0+0')

root.configure(background="powder blue")

Tops=Frame(root,width=1350,height=50,bd=8,bg="powder blue")

Tops.pack(side=TOP)

f1=Frame(root,width=600,height=600,bd=8,bg="powder blue")

f1.pack(side=LEFT)

f2=Frame(root,width=300,height=700,bd=8,bg="powder blue")

f2.pack(side=RIGHT)

fla=Frame(f1,width=600,height=200,bd=8,bg="powder blue")

fla.pack(side=TOP)

flb=Frame(f1,width=300,height=600,bd=8,bg="powder blue")

flb.pack(side=TOP)

lblinfo=Label(Tops,font=('arial',45,'bold'),text="Employee Payment Management system ",bd=10,fg="green")

lblinfo.grid(row=0,column=0)

def exit():

exit=tkinter.messagebox.askyesno("Employee system","Do you want to exit the system")

if exit>0:

root.destroy()

return

def reset():

Name.set("")

Address.set("")

HoursWorked.set("")

wageshour.set("")

Payable.set("")

Taxable.set("")

NetPayable.set("")

GrossPayable.set("")

OverTimeBonus.set("")

Employer.set("")

NINumber.set("")

txtpayslip.delete("1.0",END)

def enterinfo():

txtpayslip.delete("1.0",END)

txtpayslip.insert(END,"\t\tPay Slip\n\n")

txtpayslip.insert(END,"Name :\t\t"+Name.get()+"\n\n")

txtpayslip.insert(END,"Address :\t\t"+Address.get()+"\n\n")

txtpayslip.insert(END,"Employer :\t\t"+Employer.get()+"\n\n")

txtpayslip.insert(END,"NI Number :\t\t"+NINumber.get()+"\n\n")

txtpayslip.insert(END,"Hours Worked :\t\t"+HoursWorked.get()+"\n\n")

txtpayslip.insert(END,"Net Payable :\t\t"+NetPayable.get()+"\n\n")

txtpayslip.insert(END,"Wages per hour :\t\t"+wageshour.get()+"\n\n")

txtpayslip.insert(END,"Tax Paid :\t\t"+Taxable.get()+"\n\n")

txtpayslip.insert(END,"Payable :\t\t"+Payable.get()+"\n\n")

def weeklywages():

txtpayslip.delete("1.0",END)

hoursworkedperweek=float(HoursWorked.get())

wagesperhours=float(wageshour.get())

paydue=wagesperhours\*hoursworkedperweek

paymentdue="INR",str('%.2f'%(paydue))

Payable.set(paymentdue)

tax=paydue\*0.2

taxable="INR",str('%.2f'%(tax))

Taxable.set(taxable)

netpay=paydue-tax

netpays="INR",str('%.2f'%(netpay))

NetPayable.set(netpays)

if hoursworkedperweek > 40:

overtimehours=(hoursworkedperweek-40)+wagesperhours\*1.5

overtime="INR",str('%.2f'%(overtimehours))

OverTimeBonus.set(overtime)

elif hoursworkedperweek<=40:

overtimepay=(hoursworkedperweek-40)+wagesperhours\*1.5

overtimehrs="INR",str('%.2f'%(overtimepay))

OverTimeBonus.set(overtimehrs)

return

#=============================== Variables ========================================================

Name=StringVar()

Address=StringVar()

HoursWorked=StringVar()

wageshour=StringVar()

Payable=StringVar()

Taxable=StringVar()

NetPayable=StringVar()

GrossPayable=StringVar()

OverTimeBonus=StringVar()

Employer=StringVar()

NINumber=StringVar()

TimeOfOrder=StringVar()

DateOfOrder=StringVar()

DateOfOrder.set(time.strftime("%d/%m/%Y"))

#================================ Label Widget =================================================

lblName=Label(fla,text="Name",font=('arial',16,'bold'),bd=20,fg="red",bg="powder blue").grid(row=0,column=0)

lblAddress=Label(fla,text="Address",font=('arial',16,'bold'),bd=20,fg="red",bg="powder blue").grid(row=0,column=2)

lblEmployer=Label(fla,text="Employer",font=('arial',16,'bold'),bd=20,fg="red",bg="powder blue").grid(row=1,column=0)

lblNINumber=Label(fla,text="NI Number",font=('arial',16,'bold'),bd=20,fg="red",bg="powder blue").grid(row=1,column=2)

lblHoursWorked=Label(fla,text="Hours

Worked",font=('arial',16,'bold'),bd=20,fg="red",bg="powder blue").grid(row=2,column=0)

lblHourlyRate=Label(fla,text="Hourly Rate",font=('arial',16,'bold'),bd=20,fg="red",bg="powder blue").grid(row=2,column=2)

lblTax=Label(fla,text="Tax",font=('arial',16,'bold'),bd=20,anchor='w',fg="red",bg="powder blue").grid(row=3,column=0)

lblOverTime=Label(fla,text="OverTime",font=('arial',16,'bold'),bd=20,fg="red",bg="powder blue").grid(row=3,column=2)

lblGrossPay=Label(fla,text="GrossPay",font=('arial',16,'bold'),bd=20,fg="red",bg="powder blue").grid(row=4,column=0)

lblNetPay=Label(fla,text="Net Pay",font=('arial',16,'bold'),bd=20,fg="red",bg="powder blue").grid(row=4,column=2)

#=============================== Entry Widget =================================================

etxname=Entry(fla,textvariable=Name,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxname.grid(row=0,column=1)

etxaddress=Entry(fla,textvariable=Address,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxaddress.grid(row=0,column=3)

etxemployer=Entry(fla,textvariable=Employer,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxemployer.grid(row=1,column=1)

etxhoursworked=Entry(fla,textvariable=HoursWorked,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxhoursworked.grid(row=2,column=1)

etxwagesperhours=Entry(fla,textvariable=wageshour,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxwagesperhours.grid(row=2,column=3)

etxnin=Entry(fla,textvariable=NINumber,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxnin.grid(row=1,column=3)

etxgrosspay=Entry(fla,textvariable=Payable,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxgrosspay.grid(row=4,column=1)

etxnetpay=Entry(fla,textvariable=NetPayable,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxnetpay.grid(row=4,column=3)

etxtax=Entry(fla,textvariable=Taxable,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxtax.grid(row=3,column=1)

etxovertime=Entry(fla,textvariable=OverTimeBonus,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxovertime.grid(row=3,column=3)

#=============================== Text Widget ============================================================

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

txtpayslip=Text(f2,height=22,width=34,bd=16,font=('arial',13,'bold'),fg="green",bg="powder blue")

txtpayslip.grid(row=1,column=0)

#=============================== buttons ===============================================================

btnsalary=Button(flb,text='Weekly Salary',padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=14,fg="red",bg="powder blue",command=weeklywages).grid(row=0,column=0)

btnreset=Button(flb,text='Reset',padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=14,command=reset,fg="red",bg="powder blue").grid(row=0,column=1)

btnpayslip=Button(flb,text='View Payslip',padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=14,command=enterinfo,fg="red",bg="powder blue").grid(row=0,column=2)

btnexit=Button(flb,text='Exit

System',padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=14,command=exit,fg="red",bg="powder blue").grid(row=0,column=3)

root.mainloop()

etxnin.grid(row=1,column=3)

etxgrosspay=Entry(fla,textvariable=Payable,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxgrosspay.grid(row=4,column=1)

etxnetpay=Entry(fla,textvariable=NetPayable,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxnetpay.grid(row=4,column=3)

etxtax=Entry(fla,textvariable=Taxable,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxtax.grid(row=3,column=1)

etxovertime=Entry(fla,textvariable=OverTimeBonus,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxovertime.grid(row=3,column=3)

#=============================== Text Widget ============================================================

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

txtpayslip=Text(f2,height=22,width=34,bd=16,font=('arial',13,'bold'),fg="green",bg="powder blue")

txtpayslip.grid(row=1,column=0)

#=============================== buttons ===============================================================

btnsalary=Button(flb,text='Weekly Salary',padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=14,fg="red",bg="powder blue",command=weeklywages).grid(row=0,column=0)

btnreset=Button(flb,text='Reset',padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=14,command=reset,fg="red",bg="powder blue").grid(row=0,column=1)

btnpayslip=Button(flb,text='View Payslip',padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=14,command=enterinfo,fg="red",bg="powder blue").grid(row=0,column=2)

btnexit=Button(flb,text='Exit

etxnin.grid(row=1,column=3)

etxgrosspay=Entry(fla,textvariable=Payable,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxgrosspay.grid(row=4,column=1)

etxnetpay=Entry(fla,textvariable=NetPayable,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxnetpay.grid(row=4,column=3)

etxtax=Entry(fla,textvariable=Taxable,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxtax.grid(row=3,column=1)

etxovertime=Entry(fla,textvariable=OverTimeBonus,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxovertime.grid(row=3,column=3)

#=============================== Text Widget ============================================================

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

txtpayslip=Text(f2,height=22,width=34,bd=16,font=('arial',13,'bold'),fg="green",bg="powder blue")

txtpayslip.grid(row=1,column=0)

#=============================== buttons ===============================================================

btnsalary=Button(flb,text='Weekly Salary',padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=14,fg="red",bg="powder blue",command=weeklywages).grid(row=0,column=0)

btnreset=Button(flb,text='Reset',padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=14,command=reset,fg="red",bg="powder blue").grid(row=0,column=1)

btnpayslip=Button(flb,text='View Payslip',padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=14,command=enterinfo,fg="red",bg="powder blue").grid(row=0,column=2)

btnexit=Button(flb,text='Exit

etxnin.grid(row=1,column=3)

etxgrosspay=Entry(fla,textvariable=Payable,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxgrosspay.grid(row=4,column=1)

etxnetpay=Entry(fla,textvariable=NetPayable,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxnetpay.grid(row=4,column=3)

etxtax=Entry(fla,textvariable=Taxable,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxtax.grid(row=3,column=1)

etxovertime=Entry(fla,textvariable=OverTimeBonus,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxovertime.grid(row=3,column=3)

#=============================== Text Widget ============================================================

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

txtpayslip=Text(f2,height=22,width=34,bd=16,font=('arial',13,'bold'),fg="green",bg="powder blue")

txtpayslip.grid(row=1,column=0)

#=============================== buttons ===============================================================

btnsalary=Button(flb,text='Weekly Salary',padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=14,fg="red",bg="powder blue",command=weeklywages).grid(row=0,column=0)

btnreset=Button(flb,text='Reset',padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=14,command=reset,fg="red",bg="powder blue").grid(row=0,column=1)

btnpayslip=Button(flb,text='View Payslip',padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=14,command=enterinfo,fg="red",bg="powder blue").grid(row=0,column=2)

btnexit=Button(flb,text='Exit

System',padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=14,command=exit,fg="red",bg="powder blue").grid(row=0,column=3)

root.mainloop()

etxnin.grid(row=1,column=3)

etxgrosspay=Entry(fla,textvariable=Payable,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxgrosspay.grid(row=4,column=1)

etxnetpay=Entry(fla,textvariable=NetPayable,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxnetpay.grid(row=4,column=3)

etxtax=Entry(fla,textvariable=Taxable,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxtax.grid(row=3,column=1)

etxovertime=Entry(fla,textvariable=OverTimeBonus,font=('arial',16,'bold'),bd=16,width=22,justify='left')

etxovertime.grid(row=3,column=3)

**SYSTEM SPECIFICATION:**

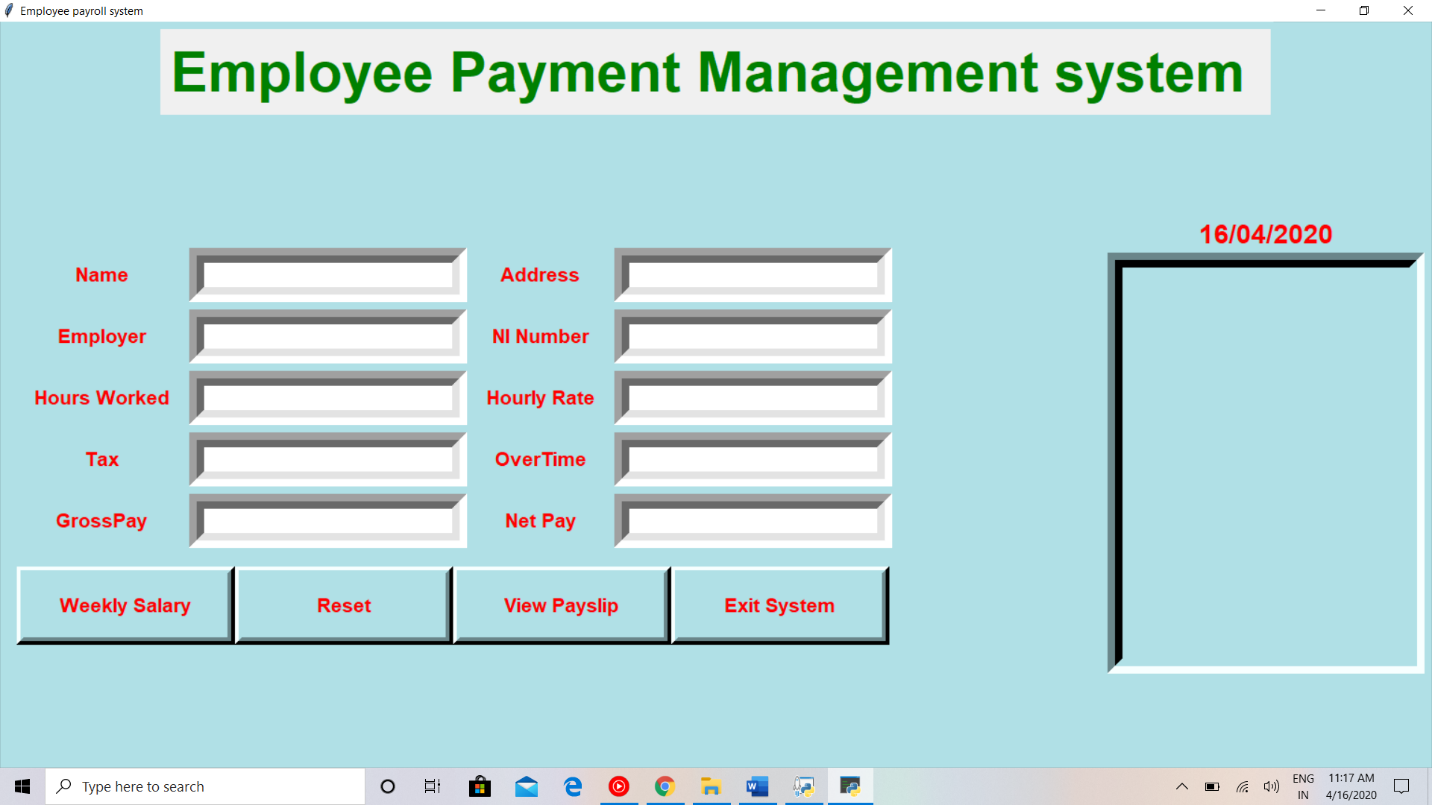
**Recommended System Requirements:**

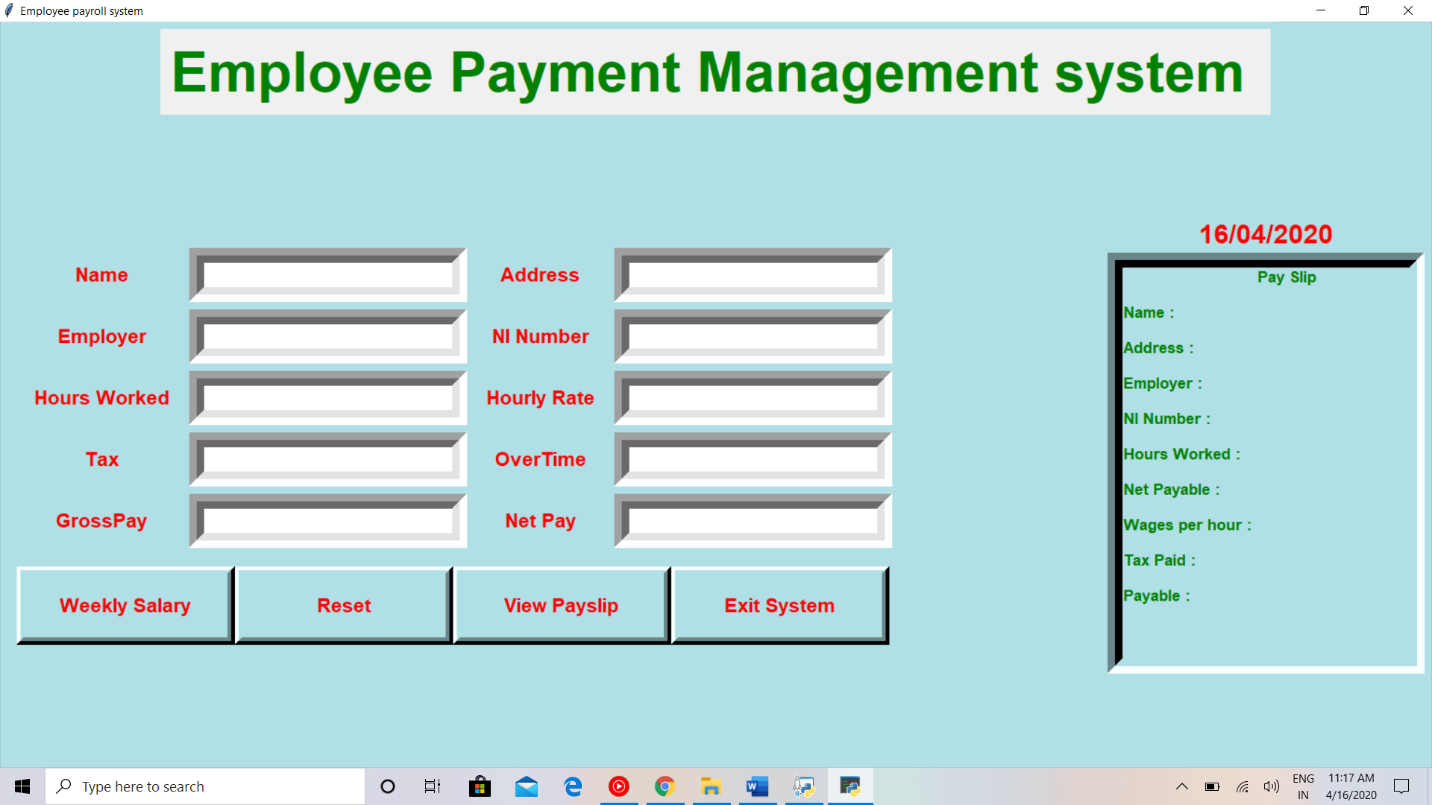
* Processors:Intel® Core™ i5 processor 4300M at 2.60 GHz or 2.59 GHz (1 socket, 2 cores, 2 threads per core), 8 GB of DRAMIntel® Xeon® processor E5-2698 v3 at 2.30 GHz (2 sockets, 16 cores each, 1 thread per core), 64 GB of DRAMIntel® Xeon Phi™ processor 7210 at 1.30 GHz (1 socket, 64 cores, 4 threads per core), 32 GB of DRAM, 16 GB of MCDRAM (flat mode enabled)
* Disk space: 2 to 3 GB
* Operating systems: Windows® 10, macOS\*, and Linux\*
* Notepad++
* PyCharm

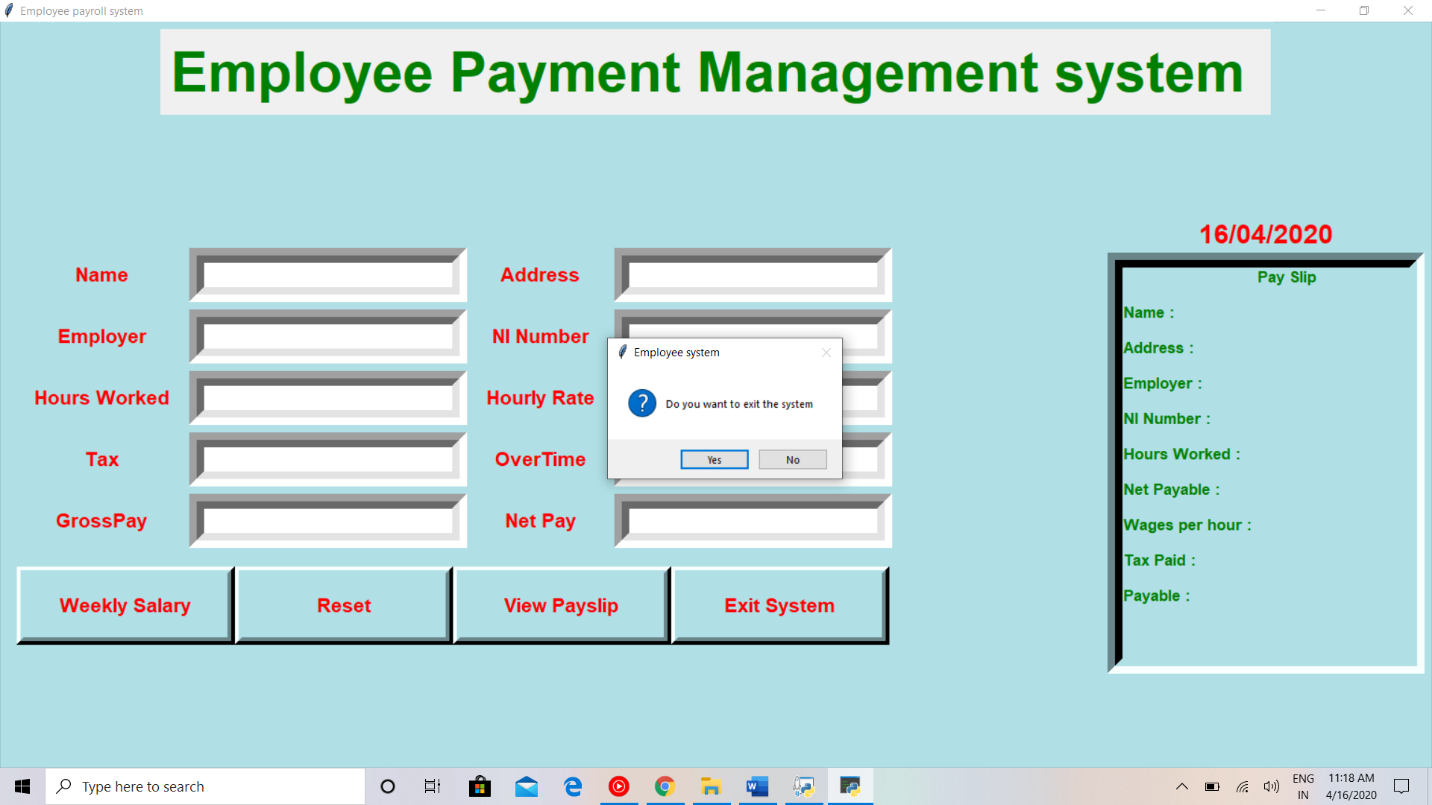
**Minimum System Requirements:**

* Processors: Intel Atom® processor or Intel® Core™ i3 processor
* Disk space: 1 GB
* Operating systems: Windows\* 7 or later, macOS, and Linux
* Python\* versions: 2.7.X, 3.6.X
* PyCharm
* Notepad++
* MySql

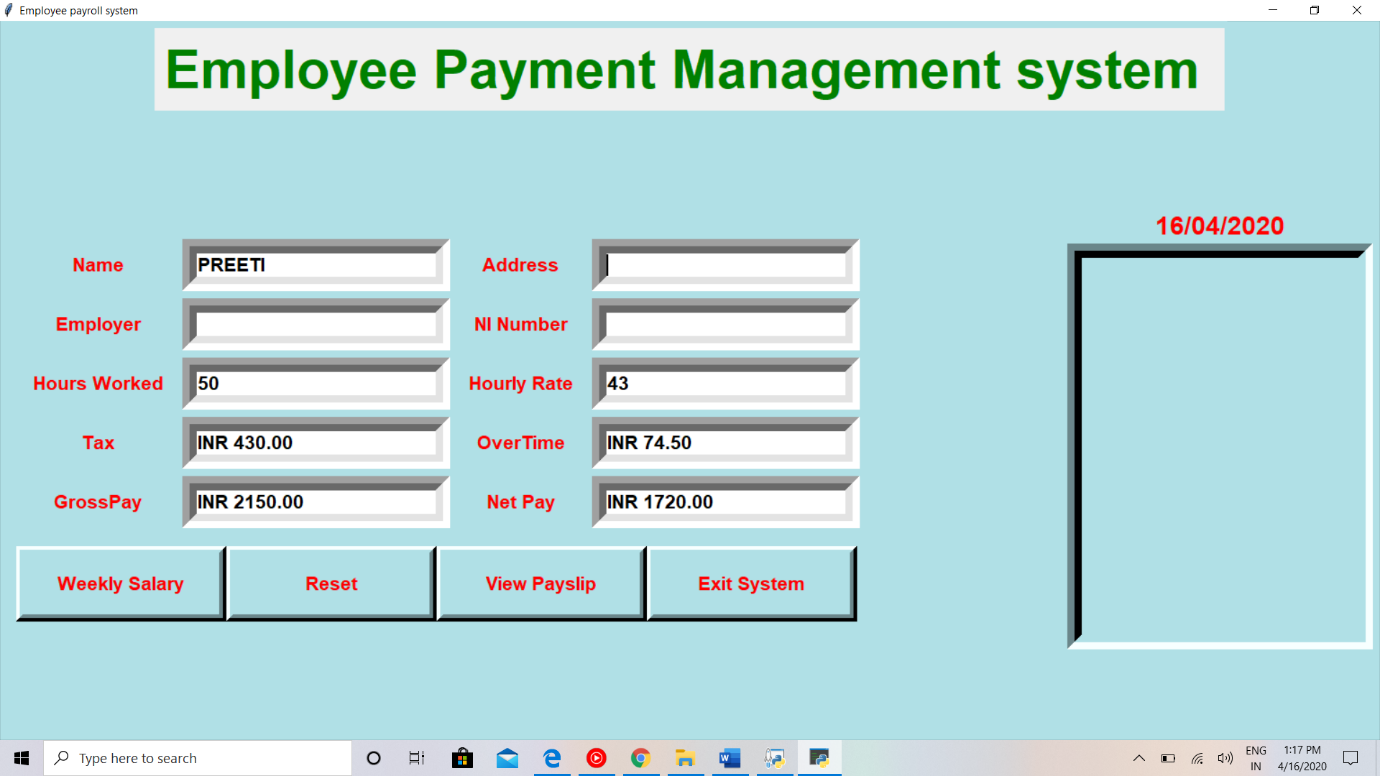
**SCREENSHOTS:**

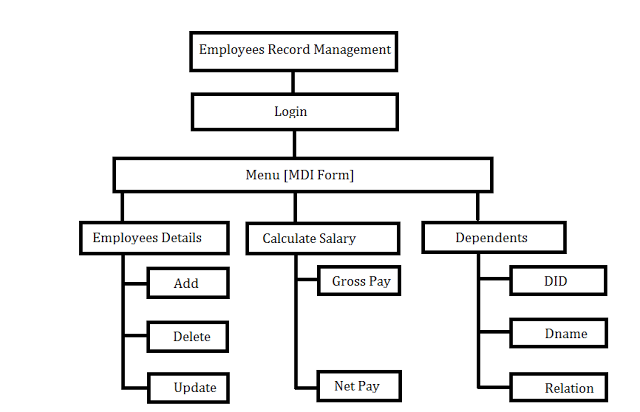












**CONCLUSION:**

The project is to digitalize the database of Employees in Organizations and enabling Administrators to have benefit from Computers. Software acts as a Information System between Employees and administrators. Here the user can keep his/her database secure and safe for a unlimited period of time.

As by using this project it become easy to manage the payment of employee without any problems.Withe the help of this we can easily manage the employee payment.