EMPLOYEES SCHEDULING SYSTEM

ODIWUOR LAMECK

Odiwuorlameck2019@gmail.com

BSCIT-05-0208/2019

INTRODUCTION

Based on my observation I was able to identify a problem faced by the business community.

Both the Employees and the Employer where facing great difficulty in keeping Track of time

The employees that arrived to work, the time they live, the duration taken by the employees during working hours, the population of Employees of an organization both male and female. The exact numbers of employees present and those absent at any given date and time.

PROBLEM RECOGNITION

The current System that is used to manage Employees Schedule is a manual process that makes managing of employees daily work routine very difficult.

Records indicating the time employees arrive to work ,the time they live from work ,Employees Absent ,Employees who apologies for not coming to work are kept in books that take up lager physical spaces .This books holding this kind of records can easily be misplaced ,papers can get old and tear off as time goes by.

The records held in such books are also difficult to manage since they are not made to synchronize Analysis of Employees Attendance can take ages.

Files Holding Employees Schedule can easily be mishandled or even get out of scope hence lost information.

The Employees Schedule information is not also safe since every employee can easily have access to their book of record ,employee can easily give in false information by signing for the next day's arrival to work ,without proper validation .

Signing of books by employees and manually writing the time they have arrived and the time they live from work can also lead to unnecessary time wastage making the process of signing in and signing off slower .

Using the manual system were a pen and book is involved can lead to records that cannot be clearly read.

The manual system does not clearly mark the balance between The Male and Female Employees Population; one has to manually count to get accurate information by going through the records manually using a pen and a paper.

Records of the employees that apologies for not coming to work and those of arrival and departure time are kept in separate books in separate places handle by separate individuals. This make accessing both the information contained in such books for analysis of an employee's rate of coming to work difficult.

As the volume of Employees Schedule records increase the amount of data becomes difficult to manage overtime , more physical space is also required to store such records.

PROBLEM SOLUTION

The new system is meant to overcome all the challenges that come with the old system of managing employees Schedule records.

The new system will be implementing the concept of a database, where records of the employees schedule are kept in an Electrical Database managed by a fully customized database management system.

Database administrators will have an easy time dealing and handling records of the time employees report to work, the time they live for home, the number of hours they should take at work. Records of Employees that have not come to work ,those present and those absent are also kept in one database hence performing data mining and data analysis is as easy as clicking a button .

Employees Schedule data retrieval and data entry will be managed by good specially designed Graphical user interface Application.

Analysis of an employee's attendance will be possible in real time.

Data Visualization will also be put in place in the new computerized system by use of Electronic charts such as pie chart ,bar chart scatter chart to enable as to see the graphical representation of the kind of the records held in employees records database

An employee's attendance to be analyzed will be done in real time by the use of a special work id that will be uniquely assigned to each employee.

Creation, Deleting, Updating, Inserting of Employees records will be possible by the use of a well-designed Graphical user interface application.

The specially assigned work id will be used by employees to sign in during arrival to work and sign out when departing from work. All this will be possible through a well-designed graphical user interface application.

When an employee arrive to work he or she should hand in the work id .The database administrator will key in the work id number and press a button written arrived.

When Employee lives for home he or she again hands in the work id and the database administrator keys in the work id into the Employees Scheduling system Departed GUI form and press the button Departed.

The time and Date that employees arrive and depart from work and there work id is automatically recorded by the database as soon as the buttons are pressed.

The new system will also be able to accommodate descriptive details of employees, such as first name ,second name ,third name, work id ,national id ,phone number ,Gender ,place of work, working hours and the job title.

Database administrators will also be able to run a quick query and be able to retrieve summarized Employees information.

The database holding the employees records will only be accessible by the use of the employees Scheduling System Application by the use of a username, local host and a password as the credentials for logging into the database server.

Database Administrators will also be able to retrieve the full list of employees details ,list of employees present ,list of employees absent, list of employees absent with apology ,list of male and female employees absent and present .

In a case where an employee sends an apology for not coming to work ,all the database administrator needs to do is to key in the employees work id and press the button written apologies and the date and time will be automatically inserted into the database. This indicates who and when the apology was made.

SYSTEM REQUIREMENTS

List of things we need to be able to make this system come to life:

To be able to build Employees scheduling system will be using python as our programming language of choice.

Download python from there official website www.python.org.

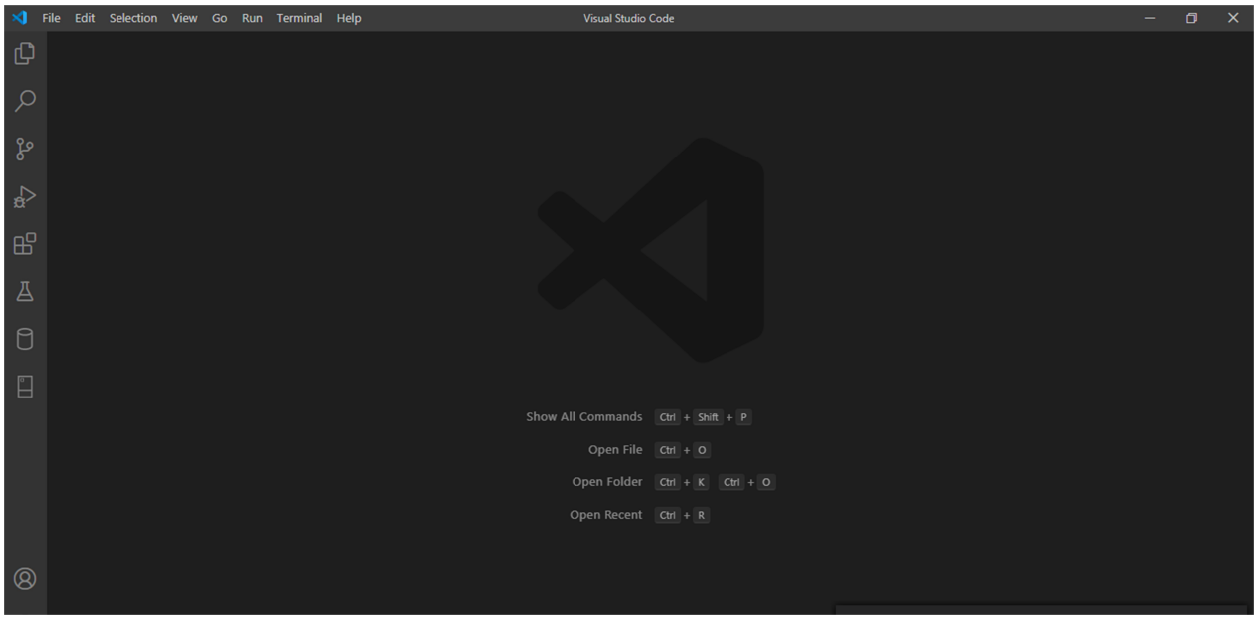
Install python in your computer system.

Install the modules listed below using pip3or pip or even pipenv.

```
import tkinter as tk

import numpy as np
from tkinter import Menu
from tkinter import scrolledtext
from tkinter import ttk
from tkinter import messagebox as msg
from tkinter import *
from PIL import Image
from PIL import ImageTk
from PyQt5.QtWidgets import *
import sys
import datetime

Import matplotlib.pyplot as plt
import mysql.connector as mysql
```

Download Mysql Community server and install in your machine.

You may also need a text Editor like visual studio code:



SYSTEM ANALYSIS

Having installed all the modules and the database server we need .The process of building Employees Scheduling system will be as painless as possible.

Ensure that your modules are working and that the database server is up and running.

SYSTEM DESIGN

PSEUDO CODE OF EMPLOYEES SCHEDULING SYSTEM.

When an organization employ's an new employee ,they Create a new record for them by recording their three names that is the first ,second third name,national Id number,phone number ,work id ,Gender ,Job title,Place of work and working hours.

Employees Work id is unique for every employee and is used to manage Employees Schedule ,that is the time the employee arrives to work the time they Depart from work and those employees absent and those who hava apologiesed .

Datetime is used to accurately map the schedule of an employee using there work id.

Data visualization helps to represent employees' records using bar charts, line charts, pie charts and scatter charts

Employees' records can easily be updated.

Every time an employee reports to work date and time is automatically recorded.When employees live from work there sign off using there work id.

SYSTEM CODING AND TESTING

Open mysql community server that you have installed and create the following :

Step one:

```
CREATE DATABASE  employees_records;
```

Step two:

```
USE employees_records;
CREATE TABLE new_employees(
First_Name  varchar(50),
 Second_Name  varchar(50),
 Third_Name  varchar(50)   NULL ,
 Id_Number int PRIMARY KEY,
 Work_Id int,
 Phone_Number int,
 Gender_      varchar(7),
 Job_Title    varchar(70),
 Place_Of_Work  varchar(80),
 Working_Hours varchar(2));
```

Step three:

```
USE employees_records;
CREATE TABLE employees_absent(work_id  varchar(50) not null,
absent_date    datetime     default   current_timestamp
);
```

Step four :

```
USE employees_records;
CREATE table   arrival_time(
arrival_date datetime  DEFAULT CURRENT_TIMESTAMP ,
work_id  varchar(50)
);
```

Step five:

```
USE Employees_Records;

CREATE TABLE apologies(
work_id   varchar(80),
apologatic_date  datetime DEFAULT CURRENT_TIMESTAMP
);
```

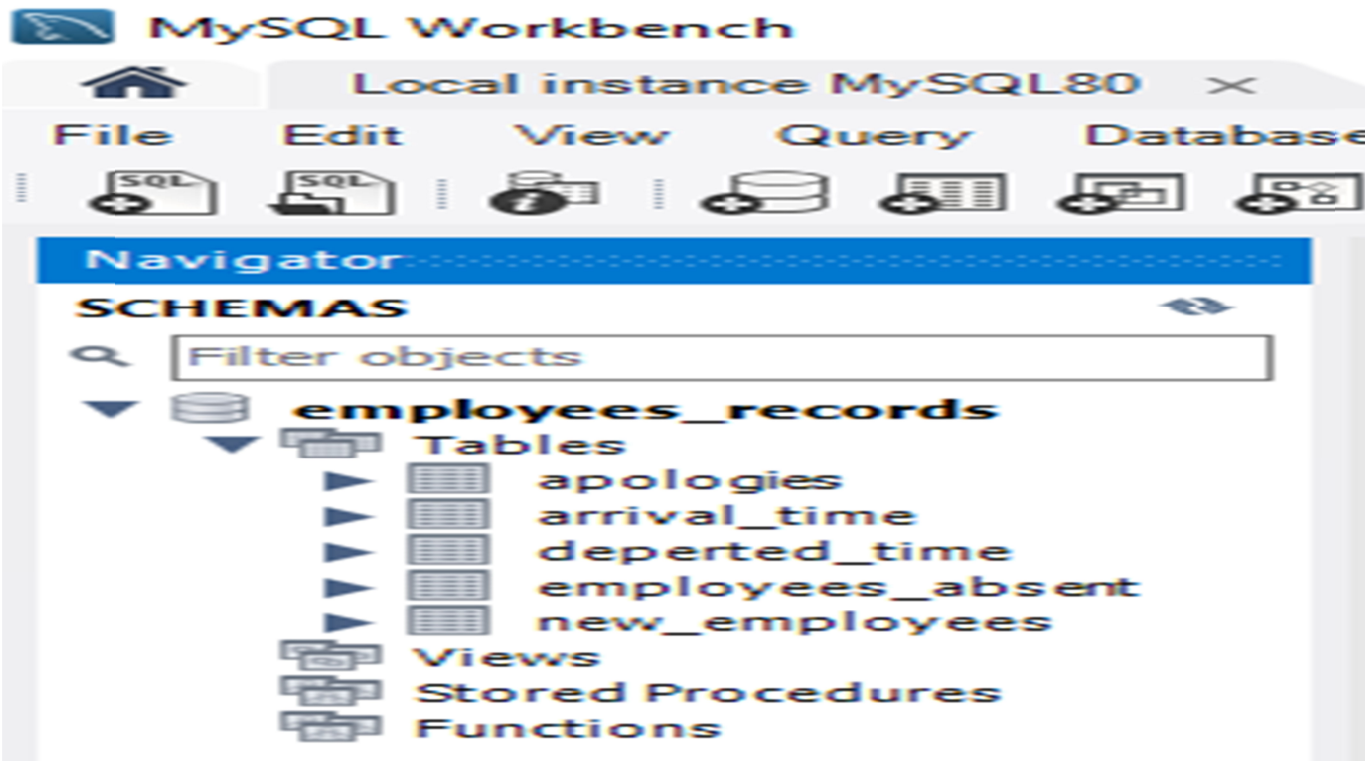
Step six:

```
USE Employees_Records;

CREATE TABLE deperted_time(
work_id  varchar(80),
departed_date  datetime DEFAULT CURRENT_TIMESTAMP
  );
```

Check if all the tables have been created successfully:

```
USE Employees_Records;
show tables;
```

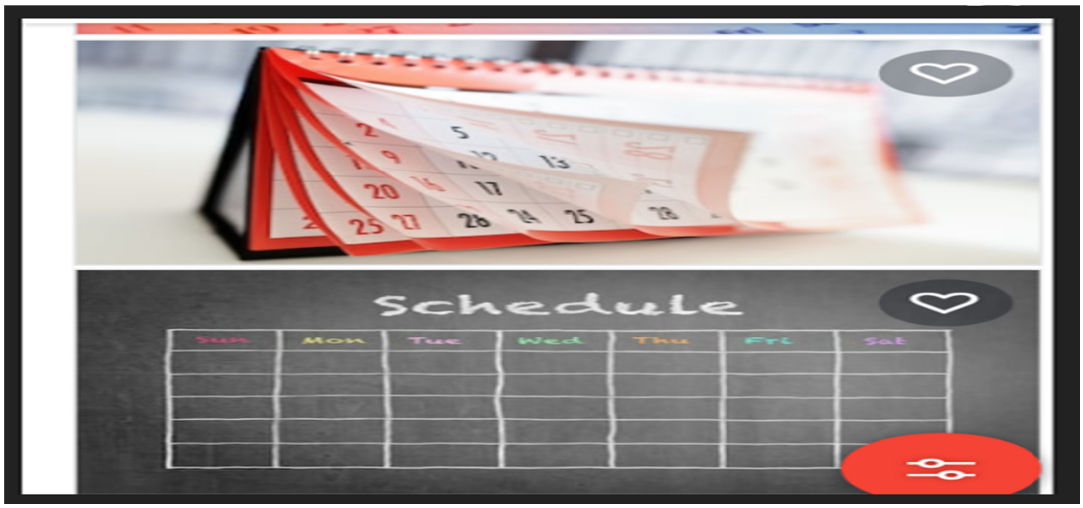
if all is well then you should have something like this :



In your local folder conferm that you have the following :

Also include this three images into that folder:

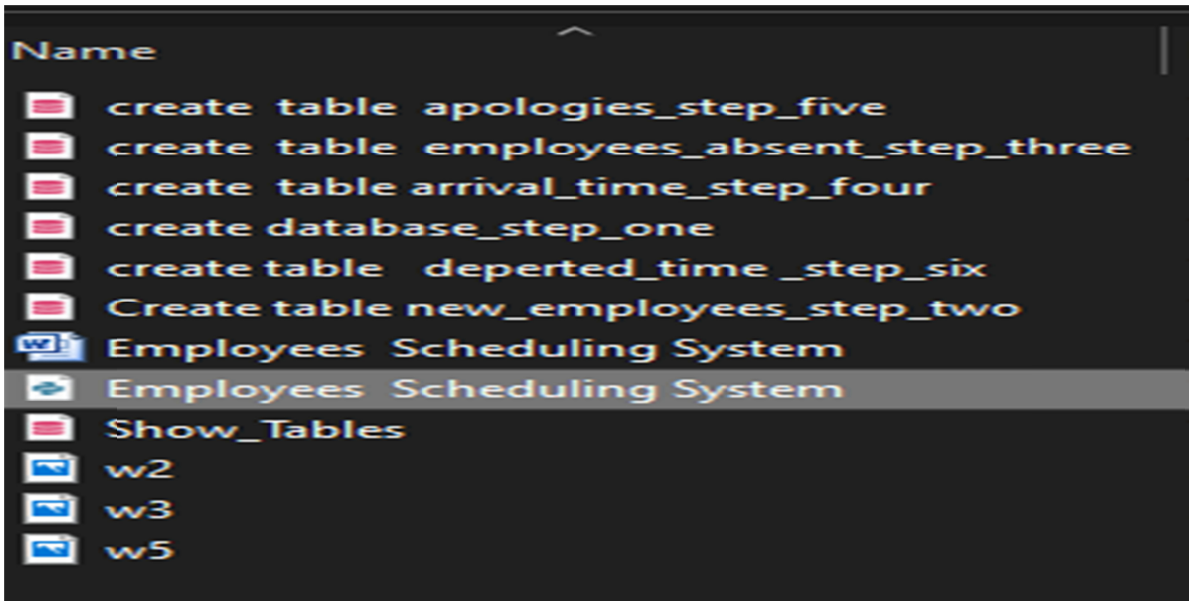
W3.png:



w2.png:

W5.png:

Finaly check if your folder contains all this :



Create a folder and in that folder create a file named Employees Scheduling System.py

Open your IDE or text editor and type the codes below:

```
import tkinter as tk
import numpy as np
from tkinter import Menu
from tkinter import scrolledtext
from tkinter import ttk
from tkinter import messagebox as msg
from tkinter import *
from PIL import Image
from PIL import ImageTk
from PyQt5.QtWidgets import *
import sys
import datetime

import matplotlib.pyplot as plt
import mysql.connector as mysql
#Authentification



#user name and a pass word.
class authentification:
    def __init__(self):
        self.user ="user"
        self.password="password"
        self.host="host"
```

```
    def Authentification(self):
        user=self.user=user_name1_log.get()
        host=self.host=host_name1_log.get()
        password=self.password=password_name1_log.get()

        #Test connection to mysql.

        try:
            conn=mysql.connect(user=user,password=password,host=host)
            msg.showinfo("Authentification :","You have successfully logged in.,welcome ")
            tabcontrol.select(tab1)
        except mysql.Error as err:
            msg.showerror("Showing Connection Status to the Database:","connection
fail!!!"+str(err))



#Initializing window.

window=tk.Tk()

#Exit from program.
```

```
def _quit():
    window.quit()
    window.destroy()



#Creating tabcontrol.



tabcontrol=ttk.Notebook(window)
tab=tk.Frame(tabcontrol, relief=tk.RAISED,bg='gray')
tab0=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab1=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab2=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab3=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab4=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab5=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab6=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab7=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab8=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab9=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab10=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab11=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab12=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab13=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
```

```
tab14=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab15=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab16=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tab17=tk.Frame(tabcontrol,relief=tk.RAISED,bg='blue')
tabcontrol.add(tab,text="Log In:")
tabcontrol.add(tab0,text="Info")
tabcontrol.add(tab1,text="Create")
tabcontrol.add(tab2,text="Insert")
tabcontrol.add(tab3,text="Delete ")
tabcontrol.add(tab4,text="Update")
tabcontrol.add(tab5,text="Analyse")
tabcontrol.add(tab6,text="Employees")
tabcontrol.add(tab7,text="Query")
tabcontrol.add(tab8,text="Charts")
tabcontrol.add(tab9,text="Guide")
tabcontrol.add(tab10,text="Arrival Time")
tabcontrol.add(tab11,text="Deperture Time")
tabcontrol.add(tab12,text="Working Hours")
tabcontrol.add(tab13,text="Employees Present")
tabcontrol.add(tab14,text="Employees Absent")
tabcontrol.add(tab15,text="Employees Absent with Aplogies")
tabcontrol.add(tab16,text="Male Employees")
tabcontrol.add(tab17,text="Female Employees")
tabcontrol.pack(expand=1,fill="both")#pack to make visible.
```

#Creating anew employees record by clicking on the file menu button new record.

```
class Create:
      #New window for the newrecord class.
        def show(self):
            tabcontrol.select(tab1)
        def delete(self):
            tabcontrol.select(tab3)
        def update(self):
           tabcontrol.select(tab4)
        def insert(self):
           tabcontrol.select(tab2)

file_create=Create()
```

#instance of a class .

```
#Login credantials


##########################################################################################
##########################################################################################
#############

labelframe=tk.LabelFrame(tab,text="LOG IN:",bg='yellow')

labelframe.pack(fill="both",side=LEFT,expand=1)

#helping to fill up spaces.

fill_name=tk.Label(labelframe,text="  ",bg='yellow')

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

fill_name=tk.Label(labelframe,text="  ",bg='yellow')

fill_name.grid(column=0,row=2)

fill_name=tk.Label(labelframe,text="  ",bg='yellow')

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

fill_name=tk.Label(labelframe,text="  ",bg='yellow')

fill_name.grid(column=0,row=4)

fill_name=tk.Label(labelframe,text="  ",bg='yellow')

fill_name.grid(column=0,row=5)

fill_name=tk.Label(labelframe,text="  ",bg='yellow')

fill_name.grid(column=0,row=6)



name1_=tk.StringVar()

user_name_log=tk.Label(labelframe,text="User Name:",bg='yellow',fg='blue')

user_name_log.grid(column=0,row=7)
```

```
user_name1_log=tk.Entry(labelframe,relief=tk.SUNKEN,textvariable=name1_,width=50,show=
"*")

user_name1_log.focus()

user_name1_log.grid(column=1,row=7)

name2_=tk.StringVar()

name3_=tk.StringVar()

fill_name=tk.Label(labelframe,text="  ",bg='yellow')

fill_name.grid(column=0,row=9)

host_name=tk.Label(labelframe,text="Host :",bg='yellow',fg='blue')

host_name.grid(column=0,row=10)

host_name1_log=tk.Entry(labelframe,relief=tk.SUNKEN,textvariable=name2_,width=50,show=
"*")

host_name1_log.grid(column=1,row=10)

fill_name=tk.Label(labelframe,text="  ",bg='yellow')

fill_name.grid(column=1,row=11)

password_name=tk.Label(labelframe,text="Password :",bg='yellow',fg='blue')

password_name.grid(column=0,row=12)

password_name1_log=tk.Entry(labelframe,relief=tk.SUNKEN,textvariable=name3_,width=50,s
how="*")

password_name1_log.grid(column=1,row=12)

fill_name=tk.Label(labelframe,text="  ",bg='yellow')

fill_name.grid(column=1,row=14)

aut=authentification()

create=tk.Button(labelframe,text=" LOG IN
",font='yellow',bg='blue',command=aut.Authentification)


create.grid(column=1,row=16,sticky='WE')
```

```
#Loading images



#Loading calendar Class.

class Window(QWidget):
    def __init__(self):
        QWidget.__init__(self)


        layout=QGridLayout()
        self.setWindowTitle("ESS : Calendar ")
        self.setLayout(layout)
        calendar=QCalendarWidget()
        calendar.showToday()
        layout.addWidget(calendar)



def Calendar():
      app=QApplication(sys.argv)
      screen=Window()
      screen.show()
      app.exec_()

canvas1=Canvas(tab0,width=1400,height=600)
```

```
button=tk.Button(tab0,text="Click here to Show
Calendar",command=Calendar,fg="yellow",bg="blue",font=("arial",14))

button.pack(fill=BOTH,expand=1)

canvas1.pack()

img1=ImageTk.PhotoImage(Image.open("w2.png"))


canvas1.create_image(0,0,anchor=NW ,image=img1)

canvas=Canvas(tab,width=700,height=700)

canvas.pack(side=RIGHT)

img=ImageTk.PhotoImage(Image.open("w5.png"))


canvas.create_image(0,0,anchor=NW ,image=img)


##########################################################################################
##########################################################################################
#############
```

#Function to create new record.

```
def Create_Record():

conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    cursor=conn.cursor()
    cursor.execute("USE employees_records")

    sql_insert_quary="""
    INSERT INTO new_employees( First_Name, Second_Name  ,Third_Name
,Id_Number,work_id,Phone_Number  ,Gender_ ,Job_Title  ,Place_Of_Work ,Working_Hours)
    VALUES(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s );"""

insert_data=(first_name1_.get(),second_name1_.get(),third_name1_.get(),id_number1_.get(),wor
k_id1_.get(),phone_number1_.get(),gender1_.get(),job_title1_.get(),
    place_of_work2_.get(),working_duration1_.get())
    cursor.execute(sql_insert_quary,insert_data)
    msg.showinfo("Validation Successful","The Record You have entered is valied.")
    conn.commit()

#Function to create new record.
def insert_Record():

conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    cursor=conn.cursor()
    cursor.execute("USE employees_records")

    sql_insert_quary="""
```

```
    INSERT INTO new_employees( First_Name, Second_Name  ,Third_Name
,Id_Number,work_id,Phone_Number  ,Gender_ ,Job_Title  ,Place_Of_Work ,Working_Hours)

    VALUES(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s );"""


insert_data=(firstname1.get(),secondname1.get(),thirdname1.get(),idnumber1.get(),workid1.get()
,phonenumber1.get(),Gender1.get(),jobtitle1.get(),

    placeofwork1.get(),workingduration1.get())

    cursor.execute(sql_insert_quary,insert_data)

    msg.showinfo("Validation Successful","The Record You have entered is valied.")

    conn.commit()

#Function to Delete record.

def Delete_Record():


conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())

    cursor=conn.cursor()

    cursor.execute("USE employees_records")

    delete_data=(_id_number1.get())

    cursor.execute("DELETE FROM new_employees WHERE  Id_Number={}
;".format(delete_data))

    msg.showinfo("Removal Successful","The Record You have entered is Deleted successfully.")

    conn.commit()

#Function to update records

def Update_Record():


conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())

    cursor=conn.cursor()
```

```
    cursor.execute("USE employees_records")
    Data="""UPDATE  new_employees
    SET First_Name=%s,Second_Name
=%s,Third_Name=%s,Id_Number=%s,work_id=%s,Phone_Number=%s,
    Gender_=%s,Job_Title =%s,Place_Of_Work=%s,Working_Hours=%s  WHERE
Id_Number=%s;
    """
    Update_Data=(__first_name1.get(),__second_name1.get(),__third_name1.get(),
    __id_number1.get(),__work_id1.get(),__phone_number1.get(),__combobox111.get(),
    __jobtitle1.get(),__placeofwork1.get(),__workingduration1.get(),__id_number1.get())
    cursor.execute(Data,Update_Data)
    conn.commit()
    msg.showinfo("Update Successful","The Record You have entered is Updated successfully.")
#creating the menubar and placing it in window.

menubar=Menu(window)
window.configure(menu=menubar)
#Create Employees records.

labelframe=tk.LabelFrame(tab1,text="Create New Employees Records:",bg='yellow')
labelframe.grid(column=0,row=0)
name1=tk.StringVar()
first_name=tk.Label(labelframe,text="First Name:",bg='yellow',fg='blue')
first_name.grid(column=0,row=0)
first_name1_=tk.Entry(labelframe,relief=tk.SUNKEN,textvariable=name1)
first_name1_.focus()
```

```
first_name1_.grid(column=1,row=0)

name2=tk.StringVar()
second_name=tk.Label(labelframe,text="Second Name:",bg='yellow',fg='blue')
second_name.grid(column=2,row=0)
second_name1_=tk.Entry(labelframe,relief=tk.SUNKEN,textvariable=name2)
second_name1_.grid(column=3,row=0)

name3=tk.StringVar()
third_name=tk.Label(labelframe,text="Third Name:",bg='yellow',fg='blue')
third_name.grid(column=4,row=0)
third_name1_=tk.Entry(labelframe,relief=tk.SUNKEN,textvariable=name3)
third_name1_.grid(column=5,row=0)

name4=tk.StringVar()
id_number=tk.Label(labelframe,text="ID Number:",bg='yellow',fg='blue')
id_number.grid(column=6,row=0)
id_number1_=tk.Entry(labelframe,relief=tk.SUNKEN,textvariable=name4)
id_number1_.grid(column=7,row=0)

name5=tk.StringVar()
work_id=tk.Label(labelframe,text="   Work ID:",bg='yellow',fg='blue')
work_id.grid(column=8,row=0)
work_id2_=tk.Entry(labelframe,relief=tk.SUNKEN,textvariable=name5)
work_id2_.grid(column=9,row=0)
```

```
name6=tk.StringVar()
phone_number=tk.Label(labelframe,text="Phone Number:",bg='yellow',fg='blue')
phone_number.grid(column=0,row=2)
phone_number1_=tk.Entry(labelframe,relief=tk.SUNKEN,textvariable=name6)
phone_number1_.grid(column=1,row=2)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=0,row=1)

name7=tk.StringVar()
gender=tk.Label(labelframe,text="Gender:  ",bg='yellow',fg='blue')
gender.grid(column=2,row=2)

gender1_=ttk.Combobox(labelframe,state='readonly',font='blue',textvariable=name7)
gender1_['values']=('Male','Female')
gender1_.grid(column=3,row=2)

name8=tk.StringVar()
job_title=tk.Label(labelframe,text="   Job Title:",bg='yellow',fg='blue')
job_title.grid(column=4,row=2)
job_title1_=ttk.Entry(labelframe,textvariable=name8)
job_title1_.grid(column=5,row=2)


name9=tk.StringVar()
```

```
place_of_work=tk.Label(labelframe,text="  Place of Work:",bg='yellow',fg='blue')
place_of_work.grid(column=6,row=2)
place_of_work1_=ttk.Entry(labelframe,textvariable=name9)
place_of_work1_.grid(column=7,row=2)



name10=tk.StringVar()
working_duration=tk.Label(labelframe,text="  working hours:",bg='yellow',fg='blue')
working_duration.grid(column=8,row=2)
working_duration1_=tk.Entry(labelframe,textvariable=name10)
working_duration1_.grid(column=9,row=2)
#hellping to fill space.hence good layout.
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=3)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=4)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=5)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=6)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=7)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=8)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
```

```
label_1.grid(column=9,row=10)




create=tk.Button(labelframe,text=" New Record
",font='yellow',bg='blue',command=Create_Record)
create.grid(column=9,row=9)




#creating the filemenu.


window.title("Employees Scheduling system:(Ess)")
filemenu=Menu(menubar,tearoff=0)
filemenu.add_command(label="New Record",command=file_create.show)
filemenu.add_separator()
filemenu.add_command(label="Save As",command=Create_Record)
filemenu.add_separator()
filemenu.add_command(label="Save",command=Create_Record)
filemenu.add_separator()
filemenu.add_command(label="UPDate",command=file_create.update)
filemenu.add_separator()
filemenu.add_command(label="Exit",command=_quit)
menubar.add_cascade(label="File",menu=filemenu)
```

```
#creating the editmenu.

editmenu=Menu(menubar,tearoff=0)
editmenu.add_command(label="Copy")
editmenu.add_separator()
editmenu.add_command(label="Cut")
editmenu.add_separator()
editmenu.add_command(label="Delete",command= file_create.delete)
editmenu.add_separator()
editmenu.add_command(label="insert",command=file_create.insert)
menubar.add_cascade(label="Edit",menu=editmenu)

#creating viewmenu.

viewmenu=Menu(menubar,tearoff=0)
viewmenu.add_command(label="Show Databases")
viewmenu.add_separator()
viewmenu.add_command(label="Show Tables")
menubar.add_cascade(label="View",menu=viewmenu)




#insert's tab values and widgets.
```

```
labelframe=tk.LabelFrame(tab2,text="Insert New Employees Records:",bg='yellow')
labelframe.grid(column=0,row=0)

firstname=tk.Label(labelframe,text="First Name:",bg='yellow',fg='blue')
firstname.grid(column=0,row=0)
firstname1=tk.Entry(labelframe,relief=tk.SUNKEN)
firstname1.focus()
firstname1.grid(column=1,row=0)

secondname=tk.Label(labelframe,text="Second Name:",bg='yellow',fg='blue')
secondname.grid(column=2,row=0)
secondname1=tk.Entry(labelframe,relief=tk.SUNKEN)
secondname1.grid(column=3,row=0)

thirdname=tk.Label(labelframe,text="Third Name:",bg='yellow',fg='blue')
thirdname.grid(column=4,row=0)
thirdname1=tk.Entry(labelframe,relief=tk.SUNKEN)
thirdname1.grid(column=5,row=0)

idnumber=tk.Label(labelframe,text="ID Number:",bg='yellow',fg='blue')
idnumber.grid(column=6,row=0)
idnumber1=tk.Entry(labelframe,relief=tk.SUNKEN)
idnumber1.grid(column=7,row=0)
workid=tk.Label(labelframe,text="   Work ID:",bg='yellow',fg='blue')
```

```
workid.grid(column=8,row=0)
workid1=tk.Entry(labelframe,relief=tk.SUNKEN)
workid1.grid(column=9,row=0)

phonenumber=tk.Label(labelframe,text="Phone Number:",bg='yellow',fg='blue')
phonenumber.grid(column=0,row=2)
phonenumber1=tk.Entry(labelframe,relief=tk.SUNKEN)
phonenumber1.grid(column=1,row=2)
label1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label1.grid(column=0,row=1)

Gender=tk.Label(labelframe,text="Gender:  ",bg='yellow',fg='blue')
Gender.grid(column=2,row=2)

Gender1=ttk.Combobox(labelframe,state='readonly',font='blue')
Gender1['values']=('Male','Female')
Gender1.grid(column=3,row=2)

jobtitle=tk.Label(labelframe,text="   Job Title:",bg='yellow',fg='blue')
jobtitle.grid(column=4,row=2)
jobtitle1=ttk.Entry(labelframe)
jobtitle1.grid(column=5,row=2)

placeofwork=tk.Label(labelframe,text="  Place of Work:",bg='yellow',fg='blue')
placeofwork.grid(column=6,row=2)
```

```
placeofwork1=ttk.Entry(labelframe)
placeofwork1.grid(column=7,row=2)

workingduration=tk.Label(labelframe,text="  working hours:",bg='yellow',fg='blue')
workingduration.grid(column=8,row=2)
workingduration1=tk.Entry(labelframe)
workingduration1.grid(column=9,row=2)
#hellping to fill space.hence good layout.
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=3)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=4)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=5)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=6)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=7)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=8)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=10)

create=tk.Button(labelframe,text=" Insert Record
",font='yellow',bg='blue',command=insert_Record)
```

```
create.grid(column=9,row=9)



#creating new controltab in tab8.

tabcontrol1=ttk.Notebook(tab8)
chart_tab0=tk.Frame(tabcontrol1,relief=tk.RAISED,bg='green')
chart_tab1=tk.Frame(tabcontrol1,relief=tk.RAISED,bg='green')
chart_tab2=tk.Frame(tabcontrol1,relief=tk.RAISED,bg='green')
chart_tab3=tk.Frame(tabcontrol1,relief=tk.RAISED,bg='green')

tabcontrol1.add(chart_tab0,text="Employee's Attendance:")
tabcontrol1.add(chart_tab1,text="Bar Chart")
tabcontrol1.add(chart_tab2,text="Pie Chart")
tabcontrol1.add(chart_tab3,text="Scatter Chart")
tabcontrol1.pack(expand=1,fill="both")

#Employees Attendance.

labelframe=tk.LabelFrame(chart_tab0,text="Input Employees ID to be Analysed:",bg='yellow')
labelframe.grid(column=0,row=0)

work_id=tk.Label(labelframe,text="   Work ID:",bg='yellow',fg='blue')
work_id.grid(column=8,row=0)
```

```
work_id1_=tk.Entry(labelframe,relief=tk.SUNKEN)

work_id1_.focus()

work_id1_.grid(column=9,row=0)

#Analysis of Employee using work id .

def Employee_Pie_Chart_Analysis_():


conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())

    fig=plt.figure(facecolor="yellow")

    ax=fig.add_axes([0.1,0.1,0.8,0.8])

    ax.axis('equal')

    Attendance=["Present","Absent","Apologies"]

    #Work_id=work_id1_.get()

    #print(Work_id)


    cursor=conn.cursor()

    cursor.execute("USE employees_records")

    cursor.execute(" SELECT count(work_id) FROM arrival_time WHERE
work_id={}".format(work_id1_.get()))


    Arrival_count=cursor.fetchall()

    cursor.execute(" SELECT DISTINCT count(*) FROM employees_absent WHERE
work_id={}".format(work_id1_.get()))


    Absent_count=cursor.fetchall()
```

```
    cursor.execute(" SELECT count(work_id) FROM Apologies WHERE
work_id={}".format(work_id1_.get()))
    Apologies_count=cursor.fetchall()


   #chech if List is all ready displayed.


    conn.commit()
    employees=[Arrival_count,Absent_count,Apologies_count]
    ax.pie(employees,labels=Attendance,autopct='%1.2f%%')
    ax.legend(labels=Attendance)
    ax.set_title("Employee Attendance Analysis :")
    plt.show()


#Pie_analysis.
create=tk.Button(chart_tab2,text=" Pie  Chart Analysis
",font='yellow',bg='blue',command=Employee_Pie_Chart_Analysis_)
create.grid(column=0,row=0)
#Scatter Attendance Analysis of An Employee.
def Employee_Scatter_Chart_Analysis_():

conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    cursor=conn.cursor()
    cursor.execute("USE employees_records")
    cursor.execute(" SELECT count(work_id) FROM arrival_time WHERE
work_id={}".format(work_id1_.get()))
```

```
    present_count=cursor.fetchall()

    cursor.execute(" SELECT DISTINCT count(*) FROM employees_absent WHERE
work_id={}".format(work_id1_.get()))


    Absent_count=cursor.fetchall()


    cursor.execute(" SELECT count(work_id) FROM Apologies WHERE
work_id={}".format(work_id1_.get()))


    Apologies_count=cursor.fetchall()


   #chech if List is all ready displayed.


    conn.commit()


    fig=plt.figure(facecolor="yellow")
    ax=fig.add_axes([0.1,0.1,0.8,0.8])
    ax.scatter("Present",present_count,color='r')
    ax.scatter("Absent",Absent_count,color='b')
    ax.scatter("Apologies",Apologies_count,color='g')


    ax.set_xlabel("Employee Attendance ")
    ax.set_ylabel('Number Of Attendance')
    ax.set_title('Employee  Attendance Analysis')
    ax.legend(labels=('Present','Absent','Apologies'))
    plt.show()
```

```
#Scatter_analysis.

create=tk.Button(chart_tab3,text=" Scatter  Chart Analysis
",font='yellow',bg='blue',command=Employee_Scatter_Chart_Analysis_)

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




#Scatter Attendance Analysis of An Employee.

def Employee_Bar_Chart_Analysis_():


conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())

    cursor=conn.cursor()

    cursor.execute("USE employees_records")

    cursor.execute(" SELECT count(work_id) FROM arrival_time WHERE
work_id={}".format(work_id1_.get()))


    present_count=cursor.fetchall()

    cursor.execute(" SELECT DISTINCT  count(*) FROM employees_absent WHERE
work_id={}".format(work_id1_.get()))


    Absent_count=cursor.fetchall()


    cursor.execute(" SELECT count(work_id) FROM Apologies WHERE
work_id={}".format(work_id1_.get()))
```

```
    Apologies_count=cursor.fetchall()
    #commit to the database.
    conn.commit()


    fig=plt.figure(facecolor="yellow")
    ax=fig.add_axes([0.1,0.1,0.8,0.8])
    ax.bar("Present",present_count[0],color='r')
    ax.bar("Absent",Absent_count[0],color='b')
    ax.bar("Apologies",Apologies_count[0],color='g')


    ax.set_xlabel("Employee Attendance ")
    ax.set_ylabel('Number Of Attendance')
    ax.set_title('Employee  Attendance Analysis')
    ax.legend(labels=('Present','Absent','Apologies'))
    plt.show()


#Bar_analysis.
create=tk.Button(chart_tab1,text=" Bar  Chart Analysis
",font='yellow',bg='blue',command=Employee_Bar_Chart_Analysis_)
create.grid(column=0,row=0)
```

#Analyse All Employees in tab5.

```
tabcontrol1=ttk.Notebook(tab5)
chart_tab_=tk.Frame(tabcontrol1,relief=tk.RAISED,bg='green')
chart_tab_1=tk.Frame(tabcontrol1,relief=tk.RAISED,bg='green')
chart_tab_2=tk.Frame(tabcontrol1,relief=tk.RAISED,bg='green')
chart_tab3=tk.Frame(tabcontrol1,relief=tk.RAISED,bg='green')
chart_tab_4=tk.Frame(tabcontrol1,relief=tk.RAISED,bg='green')
chart_tab_5=tk.Frame(tabcontrol1,relief=tk.RAISED,bg='green')
chart_tab_6=tk.Frame(tabcontrol1,relief=tk.RAISED,bg='green')
```

```
tabcontrol1.add(chart_tab_,text="Scattered Chart")
tabcontrol1.add(chart_tab_1,text="Bar Chart")
tabcontrol1.add(chart_tab_2,text="Pie Chart")
tabcontrol1.add(chart_tab3,text="Line Chart")
tabcontrol1.add(chart_tab_4,text=" Scatter Chart: Employees Present  Today ")
tabcontrol1.add(chart_tab_5,text=" Bar Chart: Employees Present Today")
```

```
tabcontrol1.add(chart_tab_6,text=" Pie Chart: Employees Present Today")



tabcontrol1.pack(expand=1,fill="both")

#Pie chart of list of Employees;

def PieChart_of_list_of_employees():


conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())

    fig=plt.figure(facecolor="yellow")

    ax=fig.add_axes([0.1,0.1,0.8,0.8])

    ax.axis('equal')

    Gender=["Male","Female"]


    cursor=conn.cursor()

    cursor.execute("USE employees_records")

    cursor.execute(" SELECT count(work_id) FROM new_employees WHERE
Gender_='Male'")


    male_count=cursor.fetchall()

    cursor.execute(" SELECT count(work_id) FROM new_employees WHERE
Gender_='Female'")


    female_count=cursor.fetchall()


   #chech if List is all ready displayed.
```

```
    conn.commit()
    employees=[male_count,female_count]
    ax.pie(employees,labels=Gender,autopct='%1.2f%%')
    ax.legend(labels=Gender)
    ax.set_title("Population Of Employees :")
    plt.show()


#Pie_analysis.
create=tk.Button(chart_tab_2,text=" Pie  Chart Analysis
",font='yellow',bg='blue',command=PieChart_of_list_of_employees)
create.grid(column=0,row=0)


#Scatter chart 0fEmployees.
def Scatter_Chart_of_Employees():

conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
      cursor=conn.cursor()
      cursor.execute("USE employees_records")
      cursor.execute(" SELECT count(work_id) FROM new_employees WHERE
Gender_='Male'")

      male_count=cursor.fetchall()
      cursor.execute(" SELECT count(work_id) FROM new_employees WHERE
Gender_='Female'")

      female_count=cursor.fetchall()
```

```
    #check if List is all ready displayed.


        conn.commit()
        fig=plt.figure(facecolor="yellow")
        ax=fig.add_axes([0.1,0.1,0.8,0.8])
        ax.scatter("Male",male_count,color='r')
        ax.scatter("Female",female_count,color='b')


        ax.set_xlabel("Employee's Gender")
        ax.set_ylabel('Number Of Employees')
        ax.set_title('scatter plot of  The Population of Employees')
        ax.legend(labels=('Male','Female'))
        plt.show()



#Scatter Analysis.
create=tk.Button(chart_tab_,text=" Scatter Analysis
",font='yellow',bg='blue',command=Scatter_Chart_of_Employees)
create.grid(column=0,row=0)


#Bar Chart Analysis.
def Bar_Chart_Analysis_of_Employees():


conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
```

```
    cursor=conn.cursor()
    cursor.execute("USE employees_records")
    cursor.execute(" SELECT count(work_id) FROM new_employees WHERE Gender_='Male'")

    male_count=cursor.fetchall()
    cursor.execute(" SELECT count(work_id) FROM new_employees WHERE
Gender_='Female'")

    female_count=cursor.fetchall()

    #check if List is all ready displayed.

    conn.commit()



    fig=plt.figure(facecolor="yellow")
    ax=fig.add_axes([0.1,0.1,0.8,0.8])

    for female in female_count:
        Female=female
    ax.bar("Female", Female,color='g',width=0.25)
    for male in male_count:
        Male=male

    ax.bar("Male", Male,color='b',width=0.25)
```

```
    ax.set_xlabel("Employees Gender")
    ax.set_ylabel("Number of Employees")
    ax.set_title("Population Of Employees")
    ax.legend(labels=['Female','Male'])
    plt.show()
# Chart Analysis of the list of Employees.


create=tk.Button(chart_tab_1,text=" Bar Chart Analysis
",font='yellow',bg='blue',command=Bar_Chart_Analysis_of_Employees)
create.grid(column=0,row=0)
#Line Chart Analysis.
def Line_Chart_Analysis_of_Employees():

conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    cursor=conn.cursor()
    cursor.execute("USE employees_records")
    cursor.execute(" SELECT count(work_id) FROM new_employees WHERE Gender_='Male'")

    male_count=cursor.fetchall()
    cursor.execute(" SELECT count(work_id) FROM new_employees WHERE
Gender_='Female'")

    female_count=cursor.fetchall()

    #check if List is all ready displayed.
```

```
    conn.commit()



    fig=plt.figure(facecolor="yellow")
    ax=fig.add_axes([0.1,0.1,0.8,0.8])

    for female in female_count:
        Female=female
    ax.plot("Female",Female)
    for male in male_count:
        Male=male

    ax.plot("Male",Male)
    ax.set_xlabel("Employees Gender")
    ax.set_ylabel("Number of Employees")
    ax.set_title("Number Of Employees")
    ax.legend(labels=['Female','Male'])
    plt.show()
#Line Chart Analysis of the list of Employees.
create=tk.Button(chart_tab3,text=" Line Chart Analysis
",font='yellow',bg='blue',command=Line_Chart_Analysis_of_Employees)
create.grid(column=0,row=0)
#Scatter Chart: Employees Present And  Absent
def Scatter_Chart_Employees_Present_Today():
```

```
conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    cursor=conn.cursor()
    cursor.execute("USE employees_records")


    i=0
    female_present_=0
    cursor.execute("SELECT  arrival_date FROM arrival_time WHERE work_id in (SELECT
work_id FROM new_employees WHERE Gender_='Female');")
    all_the_date=cursor.fetchall()
    number_of_female=np.array(all_the_date).size
    while(i< number_of_female):
        _time = all_the_date[i][0]
        _year=_time.year
        _month=_time.month
        _day=_time.day
        #print(_time.year)
        t=datetime.datetime.now()
        YEAR=_year
        MONTH=_month
        DAY=_day
        if(t.year==YEAR and t.month==MONTH and t.day==DAY):
                    female_present_+=1
        i+=1
       #check if List is all ready displayed.
```

```
    cursor.execute("SELECT  arrival_date FROM arrival_time WHERE work_id in (SELECT
work_id FROM new_employees WHERE Gender_='Male');")
    all_the_dates=cursor.fetchall()
    nuber_of_dates=np.array(all_the_dates).size
    j=0
    male_present_=0
    while(j<nuber_of_dates):
        _time = all_the_dates[j][0]
        _year=_time.year
        _month=_time.month
        _day=_time.day
        #print(_time.year)
        t=datetime.datetime.now()
        YEAR=_year
        MONTH=_month
        DAY=_day
        if(t.year==YEAR and t.month==MONTH and t.day==DAY):
                    male_present_+=1
        j+=1
       #chech if List is all ready displayed.


    conn.commit()
    fig=plt.figure(facecolor="yellow")
    ax=fig.add_axes([0.1,0.1,0.8,0.8])
    ax.scatter("Male",male_present_,color='r')
```

```
    ax.scatter("Female",female_present_,color='b')


    ax.set_xlabel("Employee's Gender")

    ax.set_ylabel('Number Of Employees')

    ax.set_title('Population Of Employees Present Today')

    ax.legend(labels=('Male','Female'))

    plt.show()

#Scatter Analysis.

create=tk.Button(chart_tab_4,text=" Scatter Chart Analysis
",font='yellow',bg='blue',command=Scatter_Chart_Employees_Present_Today)

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


#Bar Chart: Employees Present Today

def Bar_Chart_Employees_Present_Today():


conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())

    cursor=conn.cursor()

    cursor.execute("USE employees_records")


    i=0

    female_present_=0

    cursor.execute("SELECT  arrival_date FROM arrival_time WHERE work_id in (SELECT
work_id FROM new_employees WHERE Gender_='Female');")

    all_the_date=cursor.fetchall()

    number_of_female=np.array(all_the_date).size

    while(i< number_of_female):
```

```
        _time = all_the_date[i][0]
        _year=_time.year
        _month=_time.month
        _day=_time.day
        #print(_time.year)
        t=datetime.datetime.now()
        YEAR=_year
        MONTH=_month
        DAY=_day
        if(t.year==YEAR and t.month==MONTH and t.day==DAY):
                        female_present_+=1
        i+=1
       #check if List is all ready displayed.
    cursor.execute("SELECT  arrival_date FROM arrival_time WHERE work_id in (SELECT
work_id FROM new_employees WHERE Gender_='Male');")
    all_the_dates=cursor.fetchall()
    nuber_of_dates=np.array(all_the_dates).size
    j=0
    male_present_=0
    while(j<nuber_of_dates):
        _time = all_the_dates[j][0]
        _year=_time.year
        _month=_time.month
        _day=_time.day
        #print(_time.year)
```

```
        t=datetime.datetime.now()
        YEAR=_year
        MONTH=_month
        DAY=_day
        if(t.year==YEAR and t.month==MONTH and t.day==DAY):
                male_present_+=1
        j+=1
       #check if List is all ready displayed.


    conn.commit()
    fig=plt.figure(facecolor="yellow")
    ax=fig.add_axes([0.1,0.1,0.8,0.8])
    ax.bar("Female", female_present_,color='g',width=0.25)
    ax.bar("Male",  male_present_,color='b',width=0.25)
    ax.set_xlabel("Employees Gender")
    ax.set_ylabel("Number of Employees")
    ax.set_title("Population Of Employees Present Today")
    ax.legend(labels=['Female','Male'])
    plt.show()
#Bar Chart Analysis.
create=tk.Button(chart_tab_5,text=" Bar Chart  Analysis
",font='yellow',bg='blue',command=Bar_Chart_Employees_Present_Today)
create.grid(column=0,row=0)
#Pie Chart: Employees Present Today
def Pie_Chart_Employees_Present_Today():
```

```
conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    cursor=conn.cursor()
    cursor.execute("USE employees_records")


    i=0
    female_present_=0
    cursor.execute("SELECT  arrival_date FROM arrival_time WHERE work_id in (SELECT
work_id FROM new_employees WHERE Gender_='Female');")
    all_the_date=cursor.fetchall()
    number_of_female=np.array(all_the_date).size
    while(i< number_of_female):
        _time = all_the_date[i][0]
        _year=_time.year
        _month=_time.month
        _day=_time.day
        #print(_time.year)
        t=datetime.datetime.now()
        YEAR=_year
        MONTH=_month
        DAY=_day
        if(t.year==YEAR and t.month==MONTH and t.day==DAY):
                    female_present_+=1
        i+=1
       #check if List is all ready displayed.
```

```
    cursor.execute("SELECT  arrival_date FROM arrival_time WHERE work_id in (SELECT
work_id FROM new_employees WHERE Gender_='Male');")
    all_the_dates=cursor.fetchall()
    nuber_of_dates=np.array(all_the_dates).size
    j=0
    male_present_=0
    while(j<nuber_of_dates):
        _time = all_the_dates[j][0]
        _year=_time.year
        _month=_time.month
        _day=_time.day
        #print(_time.year)
        t=datetime.datetime.now()
        YEAR=_year
        MONTH=_month
        DAY=_day
        if(t.year==YEAR and t.month==MONTH and t.day==DAY):
                    male_present_+=1
        j+=1
       #check if List is all ready displayed.


    conn.commit()
    fig=plt.figure(facecolor="yellow")
    ax=fig.add_axes([0.1,0.1,0.8,0.8])
    ax.axis('equal')
```

```
    Gender=["Male","Female"]
    employees=[male_present_,female_present_]
    ax.pie(employees,labels=Gender,autopct='%1.2f%%')
    ax.legend(labels=Gender)
    ax.set_title("Population Of Employees Present Today :")
    plt.show()
```

```
#Pie Chart Analysis of employees.
create=tk.Button(chart_tab_6,text=" Pie Chart  Analysis
",font='yellow',bg='blue',command=Pie_Chart_Employees_Present_Today)
create.grid(column=0,row=0)
```

```
#male workers tabcontrol.
tabcontrol2=ttk.Notebook(tab16)
male_tab0=tk.Frame(tabcontrol2,relief=tk.RAISED,bg='green')
male_tab1=tk.Frame(tabcontrol2,relief=tk.RAISED,bg='green')
tabcontrol2.add(male_tab0,text="Male present Employees:")
tabcontrol2.add(male_tab1,text="Male Absent Employees:")
tabcontrol2.pack(expand=1,fill="both")
```

```
#Female present tabcontrol.

tabcontrol2=ttk.Notebook(tab17)
female_tab0=tk.Frame(tabcontrol2,relief=tk.RAISED,bg='green')
female_tab1=tk.Frame(tabcontrol2,relief=tk.RAISED,bg='green')
tabcontrol2.add(female_tab0,text="Female present Employees:")
tabcontrol2.add(female_tab1,text="Female Absent Employees:")
tabcontrol2.pack(expand=1,fill="both")




#insert's tab values and widgets.

labelframe=tk.LabelFrame(tab2,text="Insert New Employees Records:",bg='yellow')
labelframe.grid(column=0,row=0)

firstname=tk.Label(labelframe,text="First Name:",bg='yellow',fg='blue')
firstname.grid(column=0,row=0)
firstname1=tk.Entry(labelframe,relief=tk.SUNKEN)
firstname1.focus()
firstname1.grid(column=1,row=0)

secondname=tk.Label(labelframe,text="Second Name:",bg='yellow',fg='blue')
```

```
secondname.grid(column=2,row=0)
secondname1=tk.Entry(labelframe,relief=tk.SUNKEN)
secondname1.grid(column=3,row=0)

thirdname=tk.Label(labelframe,text="Third Name:",bg='yellow',fg='blue')
thirdname.grid(column=4,row=0)
thirdname1=tk.Entry(labelframe,relief=tk.SUNKEN)
thirdname1.grid(column=5,row=0)

idnumber=tk.Label(labelframe,text="ID Number:",bg='yellow',fg='blue')
idnumber.grid(column=6,row=0)
idnumber1=tk.Entry(labelframe,relief=tk.SUNKEN)
idnumber1.grid(column=7,row=0)
workid=tk.Label(labelframe,text="   Work ID:",bg='yellow',fg='blue')
workid.grid(column=8,row=0)
workid1=tk.Entry(labelframe,relief=tk.SUNKEN)
workid1.grid(column=9,row=0)

phonenumber=tk.Label(labelframe,text="Phone Number:",bg='yellow',fg='blue')
phonenumber.grid(column=0,row=2)
phonenumber1=tk.Entry(labelframe,relief=tk.SUNKEN)
phonenumber1.grid(column=1,row=2)
label1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label1.grid(column=0,row=1)
```

```
Gender=tk.Label(labelframe,text="Gender:  ",bg='yellow',fg='blue')
Gender.grid(column=2,row=2)

Gender1=ttk.Combobox(labelframe,state='readonly',font='blue')
Gender1['values']=('Male','Female')
Gender1.grid(column=3,row=2)

jobtitle=tk.Label(labelframe,text="   Job Title:",bg='yellow',fg='blue')
jobtitle.grid(column=4,row=2)
jobtitle1=ttk.Entry(labelframe)
jobtitle1.grid(column=5,row=2)

placeofwork=tk.Label(labelframe,text="  Place of Work:",bg='yellow',fg='blue')
placeofwork.grid(column=6,row=2)
placeofwork1=ttk.Entry(labelframe)
placeofwork1.grid(column=7,row=2)

workingduration=tk.Label(labelframe,text="  working hours:",bg='yellow',fg='blue')
workingduration.grid(column=8,row=2)
workingduration1=tk.Entry(labelframe)
workingduration1.grid(column=9,row=2)
#helping to fill space.hence good layout.
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=3)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
```

```
label_1.grid(column=9,row=4)

label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')

label_1.grid(column=9,row=5)

label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')

label_1.grid(column=9,row=6)

label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')

label_1.grid(column=9,row=7)

label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')

label_1.grid(column=9,row=8)

label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')

label_1.grid(column=9,row=10)


create=tk.Button(labelframe,text=" Insert Record
",font='yellow',bg='blue',command=insert_Record)

create.grid(column=9,row=9)


#Delete employees records tab.


labelframe=tk.LabelFrame(tab3,text="Delete Employees Records:",bg='yellow')

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


_first_name=tk.Label(labelframe,text="First Name:",bg='yellow',fg='blue')

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

_first_name1=tk.Entry(labelframe,relief=tk.SUNKEN)

_first_name1.focus()
```

```
_first_name1.grid(column=1,row=0)

_second_name=tk.Label(labelframe,text="Second Name:",bg='yellow',fg='blue')
_second_name.grid(column=2,row=0)
_second_name1=tk.Entry(labelframe,relief=tk.SUNKEN)
_second_name1.grid(column=3,row=0)

_third_name=tk.Label(labelframe,text="Third Name:",bg='yellow',fg='blue')
_third_name.grid(column=4,row=0)
_third_name1=tk.Entry(labelframe,relief=tk.SUNKEN)
_third_name1.grid(column=5,row=0)

_id_number=tk.Label(labelframe,text="ID Number:",bg='yellow',fg='blue')
_id_number.grid(column=6,row=0)
_id_number1=tk.Entry(labelframe,relief=tk.SUNKEN)
_id_number1.grid(column=7,row=0)
_work_id=tk.Label(labelframe,text="    Work ID:",bg='yellow',fg='blue')
_work_id.grid(column=8,row=0)
_work_id1=tk.Entry(labelframe,relief=tk.SUNKEN)
_work_id1.grid(column=9,row=0)

_phone_number=tk.Label(labelframe,text="Phone Number:",bg='yellow',fg='blue')
_phone_number.grid(column=0,row=2)
_phone_number1=tk.Entry(labelframe,relief=tk.SUNKEN)
_phone_number1.grid(column=1,row=2)
```

```
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=0,row=1)

Gender=tk.Label(labelframe,text="Gender:  ",bg='yellow',fg='blue')
Gender.grid(column=2,row=2)

_combo_box1=ttk.Combobox(labelframe,state='readonly',font='blue')
_combo_box1['values']=('Male','Female')
_combo_box1.grid(column=3,row=2)

_jobtitle=tk.Label(labelframe,text="   Job Title:",bg='yellow',fg='blue')
_jobtitle.grid(column=4,row=2)
_jobtitle1=ttk.Entry(labelframe)
_jobtitle1.grid(column=5,row=2)

_placeofwork=tk.Label(labelframe,text="  Place of Work:",bg='yellow',fg='blue')
_placeofwork.grid(column=6,row=2)
_placeofwork1=ttk.Entry(labelframe)
_placeofwork1.grid(column=7,row=2)

_workingduration=tk.Label(labelframe,text="  working hours:",bg='yellow',fg='blue')
_workingduration.grid(column=8,row=2)
_workingduration1=tk.Entry(labelframe)
_workingduration1.grid(column=9,row=2)
#hellping to fill space.hence good layout.
```

```
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=3)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=4)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=5)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=6)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=7)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=8)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=10)
create=tk.Button(labelframe,text=" Delete Record
",font='yellow',bg='blue',command=Delete_Record)
create.grid(column=9,row=9)



#update tab's widgets.
labelframe=tk.LabelFrame(tab4,text="Update Employees Records:",bg='yellow')
labelframe.grid(column=0,row=0)
__first_name=tk.Label(labelframe,text="First Name:",bg='yellow',fg='blue')
__first_name.grid(column=0,row=0)
__first_name1=tk.Entry(labelframe,relief=tk.SUNKEN)
```

```
__first_name1.focus()
__first_name1.grid(column=1,row=0)

__second_name=tk.Label(labelframe,text="Second Name:",bg='yellow',fg='blue')
__second_name.grid(column=2,row=0)
__second_name1=tk.Entry(labelframe,relief=tk.SUNKEN)
__second_name1.grid(column=3,row=0)

__third_name=tk.Label(labelframe,text="Third Name:",bg='yellow',fg='blue')
__third_name.grid(column=4,row=0)
__third_name1=tk.Entry(labelframe,relief=tk.SUNKEN)
__third_name1.grid(column=5,row=0)

__id_number=tk.Label(labelframe,text="ID Number:",bg='yellow',fg='blue')
__id_number.grid(column=6,row=0)
__id_number1=tk.Entry(labelframe,relief=tk.SUNKEN)
__id_number1.grid(column=7,row=0)
__work_id=tk.Label(labelframe,text="    Work ID:",bg='yellow',fg='blue')
__work_id.grid(column=8,row=0)
__work_id1=tk.Entry(labelframe,relief=tk.SUNKEN)
__work_id1.grid(column=9,row=0)

__phone_number=tk.Label(labelframe,text="Phone Number:",bg='yellow',fg='blue')
__phone_number.grid(column=0,row=2)
__phone_number1=tk.Entry(labelframe,relief=tk.SUNKEN)
```

```
__phone_number1.grid(column=1,row=2)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=0,row=1)

__Gender=tk.Label(labelframe,text="Gender:  ",bg='yellow',fg='blue')
__Gender.grid(column=2,row=2)

__combobox111=ttk.Combobox(labelframe,state='readonly',font='blue')
__combobox111['values']=('Male','Female')
__combobox111.grid(column=3,row=2)

__jobtitle=tk.Label(labelframe,text="   Job Title:",bg='yellow',fg='blue')
__jobtitle.grid(column=4,row=2)
__jobtitle1=ttk.Entry(labelframe)
__jobtitle1.grid(column=5,row=2)

__placeofwork=tk.Label(labelframe,text="  Place of Work:",bg='yellow',fg='blue')
__placeofwork.grid(column=6,row=2)
__placeofwork1=ttk.Entry(labelframe)
__placeofwork1.grid(column=7,row=2)

__workingduration=tk.Label(labelframe,text="  working hours:",bg='yellow',fg='blue')
__workingduration.grid(column=8,row=2)
__workingduration1=tk.Entry(labelframe)
__workingduration1.grid(column=9,row=2)
```

```
#hellping to fill space.hence good layout.
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=3)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=4)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=5)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=6)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=7)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=8)
label_1=tk.Label(labelframe,text="      ",bg='yellow',fg='blue')
label_1.grid(column=9,row=10)
create=tk.Button(labelframe,text=" Update Record
",font='yellow',bg='blue',command=Update_Record)
create.grid(column=9,row=9)



#List Employees Records.


labelframe=tk.LabelFrame(tab6,text="List of Employees:",bg='yellow')
labelframe.grid(column=200,row=0)
```

```
#Listing Employees Records.


def List_Employees():


conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    cursor=conn.cursor()
    cursor.execute("USE employees_records")
    cursor.execute(" SELECT * FROM new_employees")


    List=cursor.fetchall()
    L=np.array(List)
    z=L.size
    L.reshape(z,1)
    #check if List is all ready displayed.
    if(str(L) not in scrol_List.get('1.0',tk.END)):
       scrol_List.insert(tk.INSERT,L)
    conn.commit()



#Statistics_of list_of_Employees;
def Statistics_of_List_Employees():


conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    cursor=conn.cursor()
```

```
    cursor.execute("USE employees_records")
    cursor.execute(" SELECT * FROM new_employees")

    List=cursor.fetchall()
    L=np.array(List)
    z=L.size
    L.reshape(z,1)
    #chech if List is all ready displayed.
    print(L)
    return L
    conn.commit()

#Running quary.
#Running quary.
def Run_Quary():

conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    cursor=conn.cursor()
    cursor.execute("USE employees_records")
    q= scrol_query.get('1.0',tk.END)
    try:
      cursor.execute(q)
    except mysql.Error as error:
       msg.showwarning("Syntax error",str(error))
    List=cursor.fetchall()
```

```
    L=np.array(List)
    z=L.size
    L.reshape(z,1)
    #chech if List is all ready displayed.
    if(str(L) not in scrol_query.get('1.0',tk.END)):
      scrol_query.insert(tk.INSERT,L)
    conn.commit()
#Listing male  Employee's Records.


def Male_Employees_Present():


conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    cursor=conn.cursor()
    cursor.execute("USE employees_records")
    cursor.execute(" SELECT * FROM Arrival_time WHERE work_id in (SELECT work_id
FROM new_employees WHERE Gender_='Male' ) ")


    _List=cursor.fetchall()
    L=np.array(_List)
    z=L.size
    L.reshape(z,1)
    #chech if List is all ready displayed.
    if(str(L) not in scrol_Male_Employees_Present.get('1.0',tk.END)):
      scrol_Male_Employees_Present.insert(tk.INSERT,L )
    conn.commit()
```

```
def Male_Employees_Absent():


conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    t=datetime.datetime.now()
    cursor=conn.cursor()
    cursor.execute("USE employees_records")
    cursor.execute("SELECT * FROM arrival_time  ")
    arrival_time=cursor.fetchall()
    cursor.execute("SELECT  COUNT(work_id)FROM arrival_time")
    arrival_count=cursor.fetchall()
    arrival_no=0
    for arrival in arrival_count:
        arrival_no=arrival[0]
    i=0


    work_id_for_employees_present_today=[]
    Employees_Absent_Today=[]
    cursor.execute("SELECT arrival_date  FROM arrival_time  ")
    all_the_dates=cursor.fetchall()
    cursor.execute("SELECT work_id FROM  arrival_time;")
    all_the_work_ids=cursor.fetchall()
    while(i<arrival_no):
        _time = all_the_dates[i][0]
        _year=_time.year
        _month=_time.month
```

```
_day=_time.day
t=datetime.datetime.now()
YEAR=_year
MONTH=_month
DAY=_day
if(t.year==YEAR and t.month==MONTH and t.day==DAY):
        #print(all_the_work_ids[i][0])
        work_id_for_employees_present_today.append(all_the_work_ids[i][0])
        #print(work_id_for_employees_present_today)
        cursor.execute("SELECT work_id FROM new_employees")
        Employees_Absent_Today=cursor.fetchall()
        #print(Employees_Absent_Today)
        _present=[]
        _Absent=[]
        for present_ in work_id_for_employees_present_today:
                _present.append(present_)
        for Absent_ in  Employees_Absent_Today:
           _Absent.append(Absent_[0])
        for Present  in _Absent:
           if Present not in _present:
               if(str(Present) not in str(work_id_for_employees_present_today)):
                   cursor.execute("INSERT INTO employees_absent(work_id)
VALUES({})".format(Present))
                   conn.commit()
```

```
        i+=1
    cursor.execute("USE employees_records")
    cursor.execute(" SELECT * FROM Arrival_time WHERE IdNumber==(SELECT Id_Number
FROM new_employees WHERE Gender_='Male' ) ")

    List=cursor.fetchall()
    L=np.array(List)
    z=L.size
    L.reshape(z,1)
    #check if List is all ready displayed.
    if(str(L) not in scrol_Male_Employees_Absent.get('1.0',tk.END)):
       scrol_Male_Employees_Absent.insert(tk.INSERT,L)
    conn.commit()
#Listing Female Employees Records.

def Female_Employees_Present():

conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    cursor=conn.cursor()
    cursor.execute("USE employees_records")
    cursor.execute(" SELECT * FROM Arrival_time WHERE work_id in (SELECT work_id
FROM new_employees WHERE Gender_='Female' ) ")

    List=cursor.fetchall()
```

```
    L=np.array(List)
    z=L.size
    L.reshape(z,1)
    #check if List is all ready displayed.
    if(str(L) not in scrol_Female_Employees_Present.get('1.0',tk.END)):
      scrol_Female_Employees_Present.insert(tk.INSERT,L)
    conn.commit()
```

```
show_employees=tk.Button(labelframe,text="List
Employees",font='yellow',bg='blue',command=List_Employees)
show_employees.grid(column=0,row=0,sticky='W')

scrol_List=scrolledtext.ScrolledText(labelframe,width=166,height=34,fg='blue',font=('arial',11),
wrap=tk.WORD)
scrol_List.grid(column=0,row=12,sticky='WS')

#Employees present.

#

labelframe=tk.LabelFrame(tab13,text="List of Employees  present:",bg='yellow')
labelframe.grid(column=0,row=0,sticky='W')

def Employees_Present():
```

```
conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
   cursor=conn.cursor()
   cursor.execute("USE employees_records")
   cursor.execute(" SELECT work_id,arrival_date FROM arrival_time")
   List=cursor.fetchall()
   L=np.array(List)
   z=L.size
   L.reshape(z,1)

   #check if List is all ready displayed.
   if(str(L)  not in  Employees_Present.get('1.0',tk.END)):
         Employees_Present.insert(tk.INSERT,L )
   conn.commit()

show_employees3=tk.Button(labelframe,text="Present
Employees",font='yellow',bg='blue',command=Employees_Present)
show_employees3.grid(column=0,row=0,sticky='w')

Employees_Present=scrolledtext.ScrolledText(labelframe,width=166,height=51,fg='blue',wrap=t
k.WORD,font=("arial",11))
Employees_Present.grid(column=0,row=10)

#Employees Absent.
#Employees Absent
def Employees_Absent():
```

```
conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    t=datetime.datetime.now()
    cursor=conn.cursor()
    cursor.execute("USE employees_records")
    cursor.execute("SELECT * FROM arrival_time  ")
    arrival_time=cursor.fetchall()
    cursor.execute("SELECT  COUNT(work_id)FROM arrival_time")
    arrival_count=cursor.fetchall()
    arrival_no=0
    for arrival in arrival_count:
        arrival_no=arrival[0]
    i=0

    work_id_for_employees_present_today=[]
    Employees_Absent_Today=[]
    cursor.execute("SELECT arrival_date  FROM arrival_time  ")
    all_the_dates=cursor.fetchall()
    cursor.execute("SELECT work_id FROM  arrival_time;")
    all_the_work_ids=cursor.fetchall()
    while(i<arrival_no):
        _time = all_the_dates[i][0]
        _year=_time.year
        _month=_time.month
        _day=_time.day
```

```
t=datetime.datetime.now()
YEAR=_year
MONTH=_month
DAY=_day
if(t.year==YEAR and t.month==MONTH and t.day==DAY):
        #print(all_the_work_ids[i][0])
        work_id_for_employees_present_today.append(all_the_work_ids[i][0])
        #print(work_id_for_employees_present_today)
        cursor.execute("SELECT work_id FROM new_employees")
        Employees_Absent_Today=cursor.fetchall()
        #print(Employees_Absent_Today)
        _present=[]
        _Absent=[]
        for present_ in work_id_for_employees_present_today:
                _present.append(present_)
        for Absent_ in  Employees_Absent_Today:
           _Absent.append(Absent_[0])
        for Present  in _Absent:
           if Present not in _present:
               if(str(Present) not in str(work_id_for_employees_present_today)):
                    cursor.execute("INSERT INTO employees_absent(work_id)
VALUES({})".format(Present))
```

```
        i+=1


    cursor.execute("SELECT  * FROM employees_absent ;")
    all_employees_absent=cursor.fetchall()
    List=all_employees_absent
    L=np.array(List)
    z=L.size
    L.reshape(z,1)
    #check if List is all ready displayed.
    if(str(L) not in scrol_Employees_absent.get('1.0',tk.END)):
       scrol_Employees_absent.insert(tk.INSERT,L)
    conn.commit()


labelframe=tk.LabelFrame(tab14,text="List of Employees  Absent:",bg='yellow')
labelframe.grid(column=200,row=0)


show_employees=tk.Button(labelframe,text="Absent
Emplyees",font='yellow',bg='blue',command=Employees_Absent)
show_employees.grid(column=0,row=0,sticky='w')
scrol_Employees_absent=scrolledtext.ScrolledText(labelframe,width=166,height=34,fg='blue',fo
nt=("arial",11))
scrol_Employees_absent.grid(column=0,row=10)


#Employees Absent with Aplologies.
def Apologies():
```

```
conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    cursor=conn.cursor()
    cursor.execute("USE employees_records")
    cursor.execute("SELECT Work_Id  FROM new_employees")


    test=cursor.fetchall()
    test2=work_id_apologie.get()
    if test2 in str(test):
        cursor.execute(" INSERT INTO apologies(work_id)
VALUES({});".format(work_id_apologie.get()))
        cursor.execute("SELECT first_name,second_name,third_name FROM new_employees
WHERE work_id={}".format(test2))
        name=cursor.fetchall()
        msg.showinfo("Absent with Aplology: ",'Name: '+str(name)+'\n Aplogiesed for not
comming today.')
        conn.commit()
    else:
       msg.showwarning("Authentification Error:","Work Id you have entered does not exist!!!")
def List_Employees_Absent_with_apologies():

conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    cursor=conn.cursor()
    cursor.execute("USE employees_records")
    cursor.execute(" SELECT * FROM apologies")
```

```
    List=cursor.fetchall()
    L=np.array(List)
    z=L.size
    L.reshape(z,1)
    #check if List is all ready displayed.
    if(str(L) not in scrol_apologies.get('1.0',tk.END)):
       scrol_apologies.insert(tk.INSERT,L)
    conn.commit()
labelframe=tk.LabelFrame(tab15,text="List of Employees  Absent With Apology:",bg='yellow')
labelframe.grid(column=200,row=0)
show_employees=tk.Button(labelframe,text="Apologetic
Emplyees",font='yellow',bg='blue',width=30,command=List_Employees_Absent_with_apologie
s)
show_employees.grid(column=0,row=2,sticky='W')
show_employees=tk.Button(labelframe,text="Apologies",font='yellow',bg='blue',width=30,com
mand=Apologies)
show_employees.grid(column=0,row=1,sticky='W')
work_id_apologie=tk.Entry(labelframe,relief=tk.SUNKEN,width=46)
work_id_apologie.focus()
work_id_apologie.grid(column=0,row=0,sticky='W')

scrol_apologies=scrolledtext.ScrolledText(labelframe,width=166,height=34,fg='blue',font=("aria
l",11))
scrol_apologies.grid(column=0,row=10)
```

#Male present Employees.

```
labelframe=tk.LabelFrame(male_tab0,text="Male Present Employees:",bg='yellow')

labelframe.grid(column=200,row=0)

show_employees=tk.Button(labelframe,text="Male
Present",font='yellow',bg='blue',command=Male_Employees_Present)

show_employees.grid(column=0,row=0,sticky='w')


scrol_Male_Employees_Present=scrolledtext.ScrolledText(labelframe,width=166,height=34,fg='
blue',font=("arial",11))

scrol_Male_Employees_Present.grid(column=0,row=10)
```

#Male Absent Employees.

```
# Male Absent.

def Male_Absent_employees():


conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())

   cursor=conn.cursor()

   cursor.execute("USE employees_records")

   cursor.execute("SELECT * FROM employees_absent WHERE work_id  in  (SELECT
work_id FROM new_employees WHERE Gender_='Male');")

   all_male_absent=cursor.fetchall()

   List=all_male_absent

   L=np.array(List)
```

```
    z=L.size
    L.reshape(z,1)
    #chech if List is all ready displayed.
    if(str(L) not in scrol_Male_Employees_absent.get('1.0',tk.END)):
      scrol_Male_Employees_absent.insert(tk.INSERT,L)
    conn.commit()


labelframe=tk.LabelFrame(male_tab1,text="Male Absent  Employees:",bg='yellow')
labelframe.grid(column=200,row=0)
show_employees=tk.Button(labelframe,text="Male Absent",font='yellow',bg='blue'
,command=Male_Absent_employees)
show_employees.grid(column=0,row=0,sticky='w')


scrol_Male_Employees_absent=scrolledtext.ScrolledText(labelframe,width=166,height=34,fg='b
lue',font=("arial",11))
scrol_Male_Employees_absent.grid(column=0,row=10)




#Female present Employees.



labelframe=tk.LabelFrame(female_tab0,text="Female Present Employees:",bg='yellow')
labelframe.grid(column=200,row=0)
```

```
show_employees=tk.Button(labelframe,text="Female
Present",font='yellow',bg='blue',command=Female_Employees_Present)

show_employees.grid(column=0,row=0,sticky='w')


scrol_Female_Employees_Present=scrolledtext.ScrolledText(labelframe,width=166,height=34,f
g='blue',font=("arial",11))

scrol_Female_Employees_Present.grid(column=0,row=10)



#Female Absent Employees.

def Female_Absent_employees():


conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())

   cursor=conn.cursor()

   cursor.execute("USE employees_records")

   cursor.execute("SELECT * FROM employees_absent WHERE work_id  in  (SELECT
work_id FROM new_employees WHERE Gender_='Female');")

   all_female_absent=cursor.fetchall()

   List=all_female_absent

   L=np.array(List)

   z=L.size

   L.reshape(z,1)

   #chech if List is all ready displayed.

   if(str(L) not in scrol_female_absent.get('1.0',tk.END)):

     scrol_female_absent.insert(tk.INSERT,L)

   conn.commit()
```

```
labelframe=tk.LabelFrame(female_tab1,text="Female Absent  Employees:",bg='yellow')

labelframe.grid(column=200,row=0)

show_employees=tk.Button(labelframe,text="Female
Absent",font='yellow',bg='blue',command=Female_Absent_employees)

show_employees.grid(column=0,row=0,sticky='w')


scrol_female_absent=scrolledtext.ScrolledText(labelframe,width=166,height=34,fg='blue',font=(
"arial",11))

scrol_female_absent.grid(column=0,row=10)



#Run Quary widgets.


labelframe=tk.LabelFrame(tab7,text="Run A Query:[Type your Query bellow]",bg='yellow')

labelframe.grid(column=200,row=0)

show_employees=tk.Button(labelframe,text="Run
Query:",font='yellow',bg='blue',command=Run_Quary)

show_employees.grid(column=0,row=0,sticky='w')


scrol_query=scrolledtext.ScrolledText(labelframe,width=166,height=34,fg='blue',font=('arial',11
))

scrol_query.grid(column=0,row=10)
```

```
#Arrival time tab.
def arrival():

conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
    cursor=conn.cursor()
    cursor.execute("USE employees_records")
    cursor.execute("SELECT Work_Id  FROM new_employees")


    test=cursor.fetchall()
    test2=arrival_.get()
    if test2 in str(test):
        cursor.execute(" INSERT INTO Arrival_time(work_id)
VALUES({});".format(arrival_.get()))
        cursor.execute("SELECT first_name,second_name,third_name FROM new_employees
WHERE work_id={}".format(test2))
        name=cursor.fetchall()
        msg.showinfo("Arrived",'Name: '+str(name)+'Arrived Now')
        conn.commit()
    else:
       msg.showwarning("Authentification Error:","Work Id you have entered does not exist!!!")


labelframe=tk.LabelFrame(tab10,text="Enter Employee's Work ID:",bg='yellow')
labelframe.grid(column=0,row=0)
```

```
first_name=tk.Label(labelframe,text="work ID:",bg='yellow',fg='blue')

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

arrival_=tk.Entry(labelframe,relief=tk.SUNKEN)

arrival_.focus()

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


create=tk.Button(labelframe,text=" Arrived ",font='yellow',bg='blue',command=arrival)

create.grid(column=9,row=9)


#Deperture time.

def Deperted():


conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())

   cursor=conn.cursor()

   cursor.execute("USE employees_records")

   cursor.execute("SELECT Work_Id  FROM new_employees")

   test=cursor.fetchall()

   cursor.execute("SELECT work_id  FROM arrival_time")

   test1=cursor.fetchall()

   test2=deperted_time.get()

   if test2 in str(test) and test2 in str(test1):

      cursor.execute(" INSERT INTO deperted_time(work_id)
VALUES({});".format(deperted_time.get()))

      cursor.execute("SELECT first_name,second_name,third_name FROM new_employees
WHERE work_id={}".format(test2))

      name=cursor.fetchall()
```

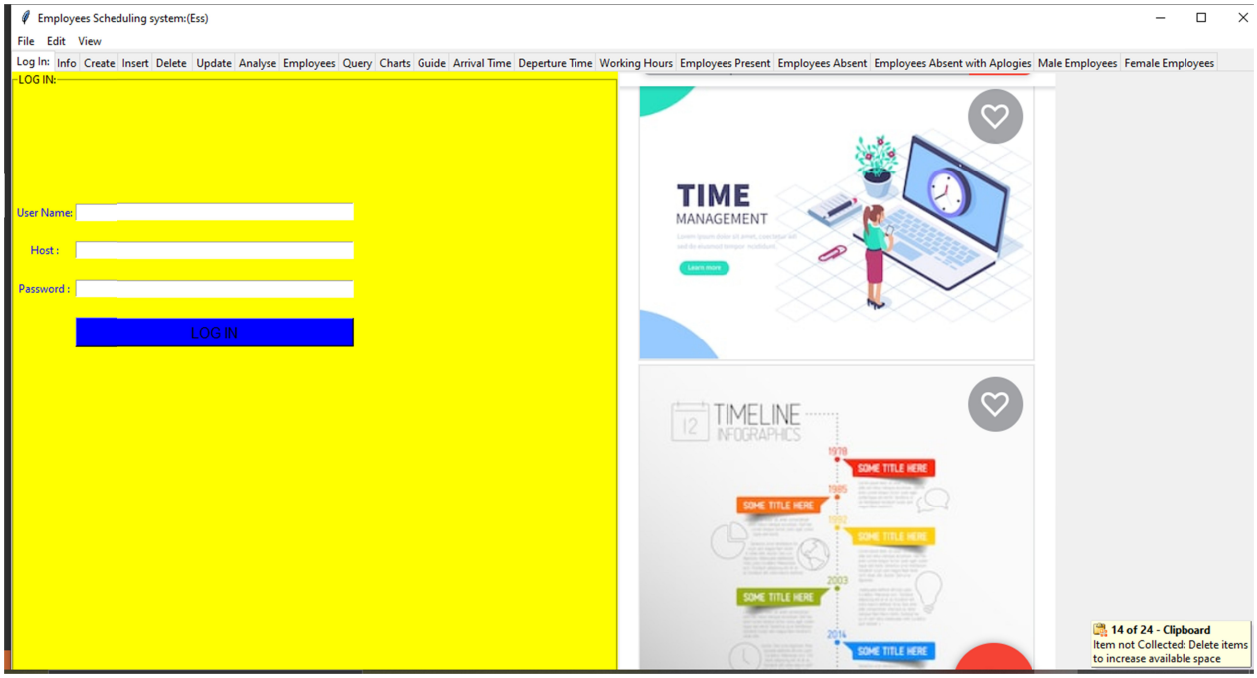
```
        msg.showinfo("Deperted",'Name: '+str(name)+'Has just signed out right Now')
        conn.commit()
    elif test2  not in str(test1) and test2 in str(test):
        cursor.execute("SELECT first_name,second_name,third_name FROM new_employees
WHERE work_id={}".format(test2))
        name=cursor.fetchall()
        msg.showinfo("Deperture Error:",'Name: '+str(name)+'did not even arrive today.')
    else :
       msg.showwarning("Authentification Error:","Work Id you have entered does not exist!!!")


labelframe=tk.LabelFrame(tab11,text="Enter Employee's Work ID:",bg='yellow')
labelframe.grid(column=0,row=0)


first_name=tk.Label(labelframe,text="work ID:",bg='yellow',fg='blue')
first_name.grid(column=0,row=0)
deperted_time=tk.Entry(labelframe,relief=tk.SUNKEN)
deperted_time.focus()
deperted_time.grid(column=1,row=0)


create=tk.Button(labelframe,text=" Deperted ",font='yellow',bg='blue',command=Deperted)
create.grid(column=9,row=9)



#Working hours.
def set_workingours():
```

```
conn=mysql.connect(user=user_name1_log.get(),password=password_name1_log.get(),host=hos
t_name1_log.get())
   cursor=conn.cursor()
   cursor.execute("USE employees_records")
   cursor.execute("SELECT work_id FROM new_employees")
   test_work_id=cursor.fetchall()
   test1_work_id=work_id1.get()
   if test1_work_id in str(test_work_id):
       cursor.execute("UPDATE new_employees SET working_hours={} WHERE
work_id={}".format(working_Hours.get(),work_id1.get()))
       msg.showinfo("Updating working Hours:","Change has been made successfully.")
       conn.commit()
   else:
       msg.showwarning("Authentification Error:","The work_id you have entered :
["+str(test1_work_id)+"] is unknown please try another Work ID.")
labelframe=tk.LabelFrame(tab12,text="Enter Employee's Work ID:",bg='yellow')
labelframe.grid(column=0,row=0)

first_name=tk.Label(labelframe,text="work ID:",bg='yellow',fg='blue')
first_name.grid(column=0,row=0)
work_id1=tk.Entry(labelframe,relief=tk.SUNKEN)
work_id1.focus()
work_id1.grid(column=1,row=0)

first_name=tk.Label(labelframe,text="Set Working Hours:",bg='yellow',fg='blue')
first_name.grid(column=2,row=0)
```

```
working_Hours=tk.Entry(labelframe,relief=tk.SUNKEN)

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



create=tk.Button(labelframe,text=" Set Working
Hours",font='yellow',bg='blue',command=set_workingours)

create.grid(column=9,row=9)



window.mainloop()
```

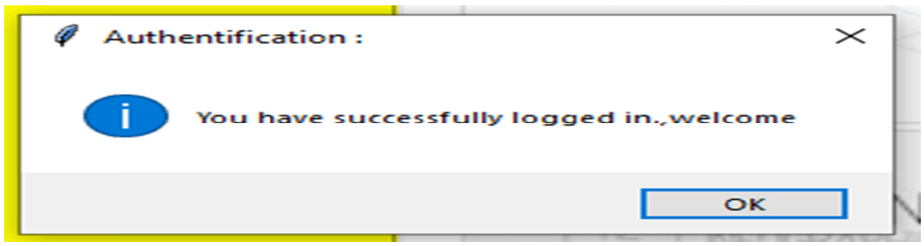
SYSTEM IMPLIMENTATION

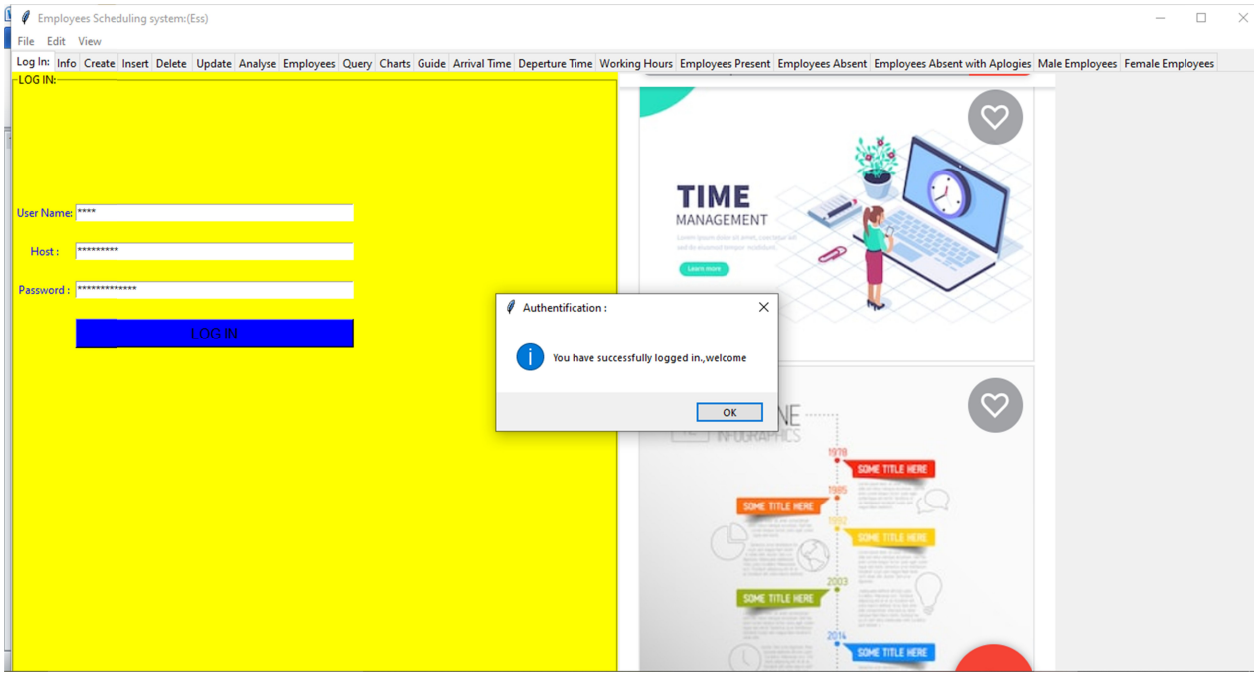
When you follow the instructions correctly and run the program then you should get:



Log in with a username ,Host and Password of the database running on your local computer System:

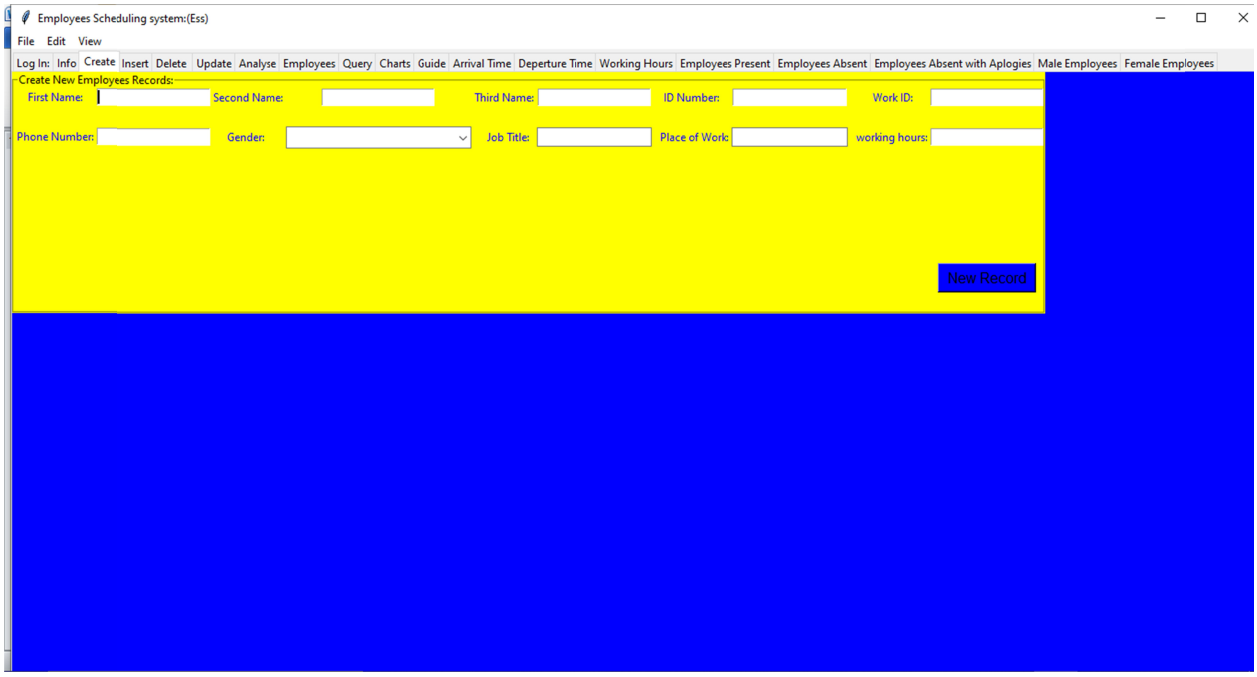
If you use the right credentials then you will get the nitification below:



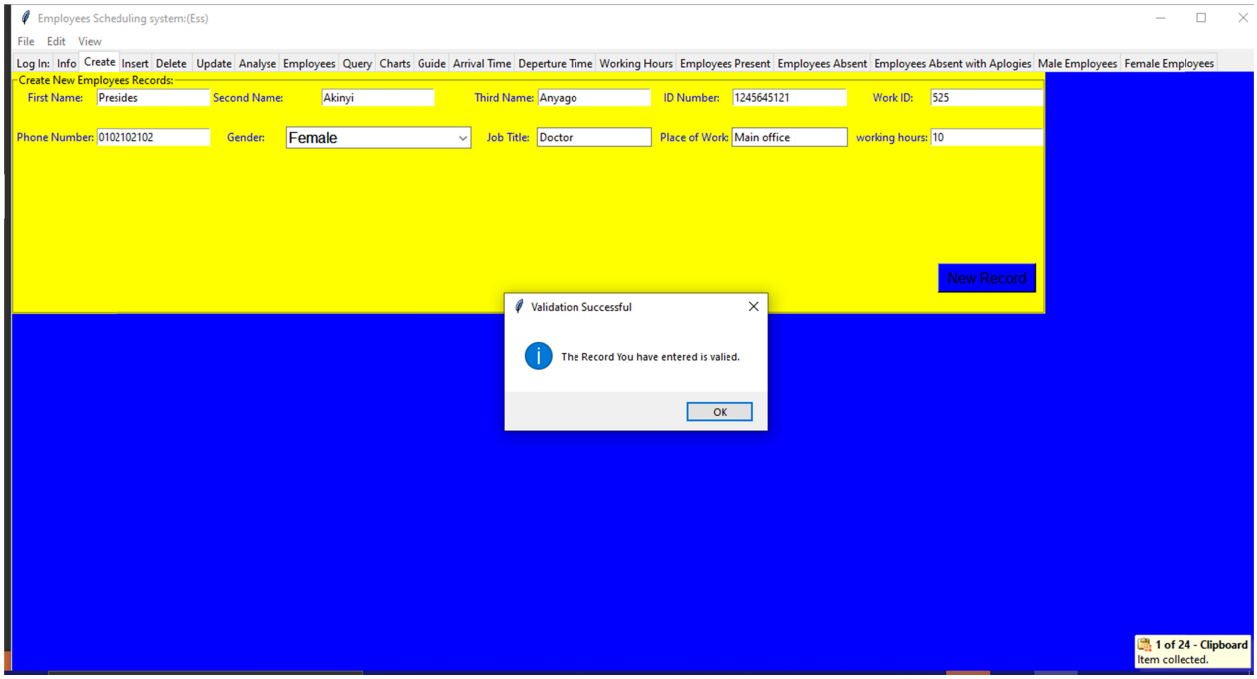


Click on the **ok** button on the Authentification MassageBox.

You will be able to see a Create tab opening up.



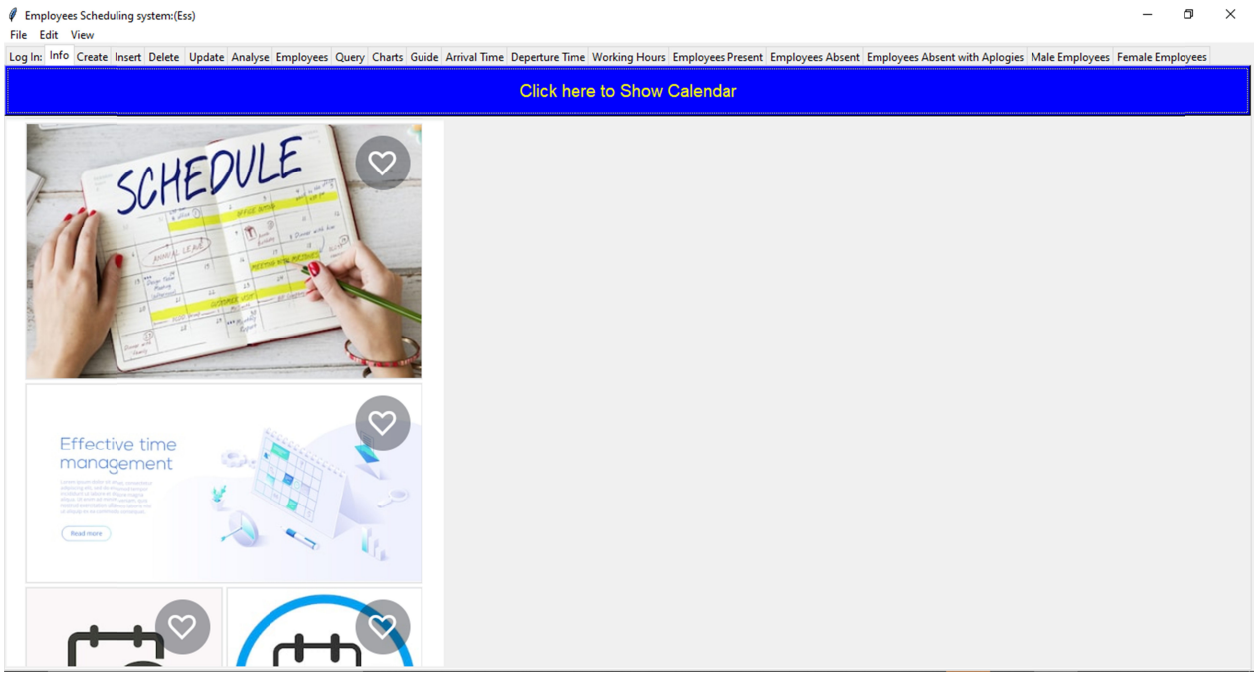
Fill in the form details and click on the button New Record:



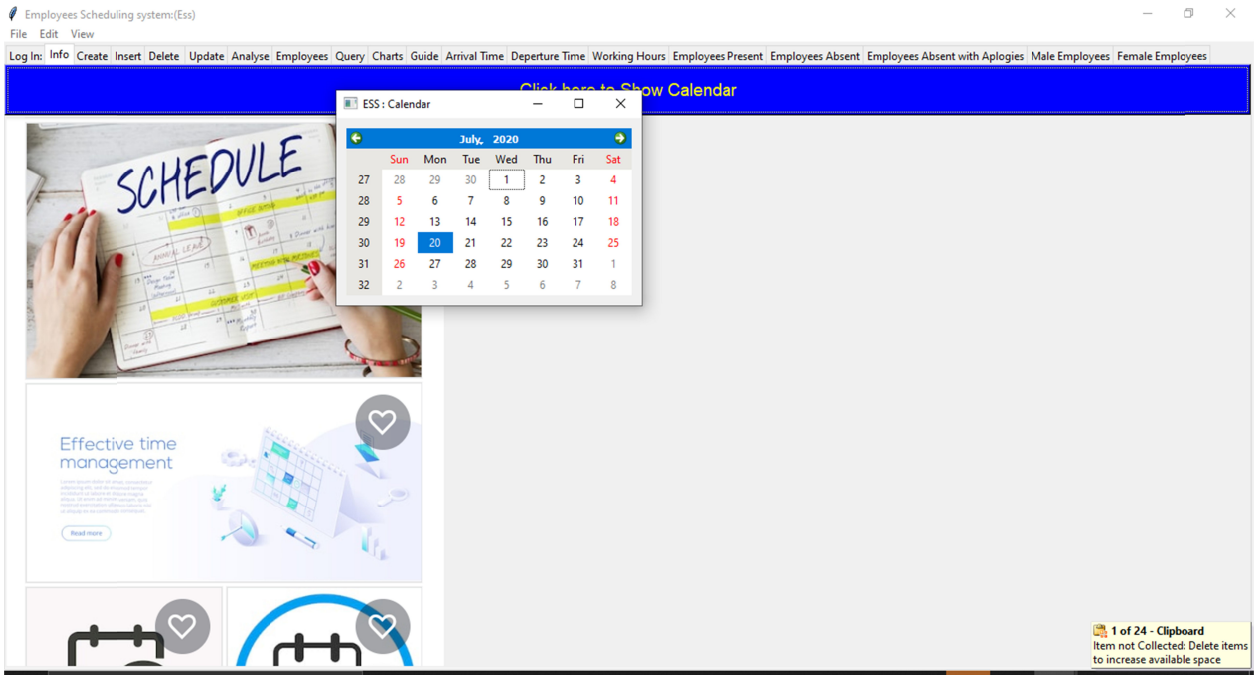
Navigate to to the Employees Tab:

Click on the list Employees Button to show the list of employees.

If you want to display the Employees Scheduling system Calendar click on the Info tab .

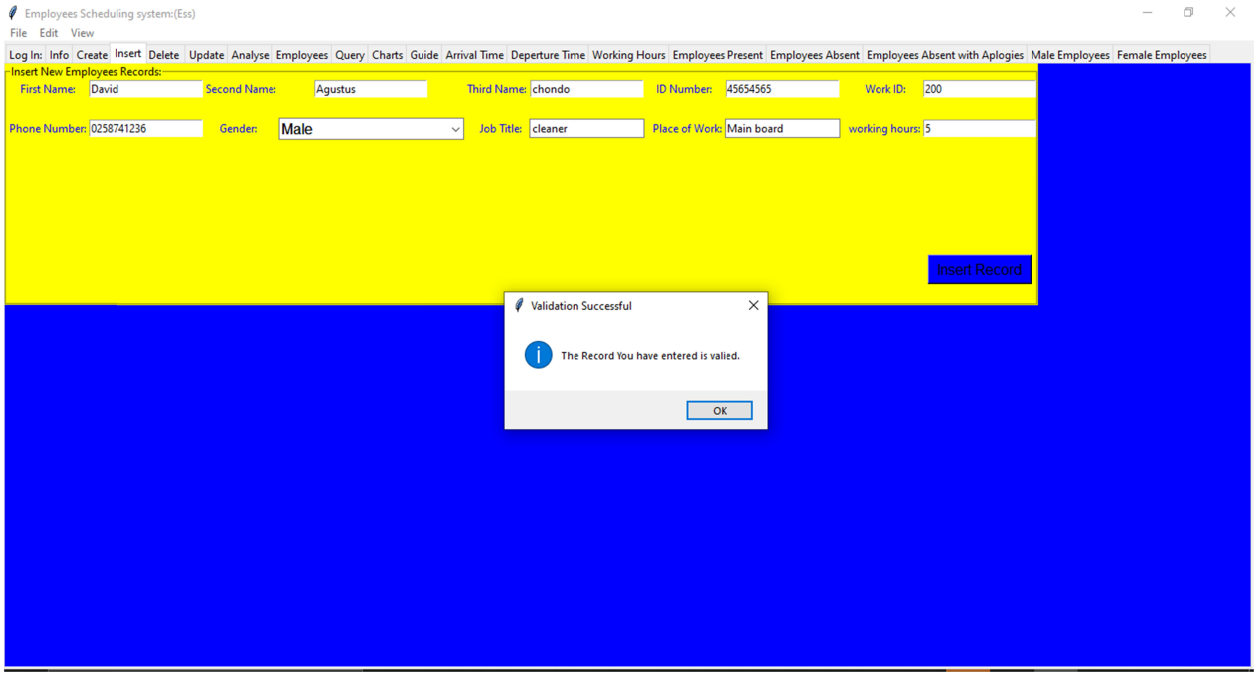


Then click on the **click here to show Calendar .**

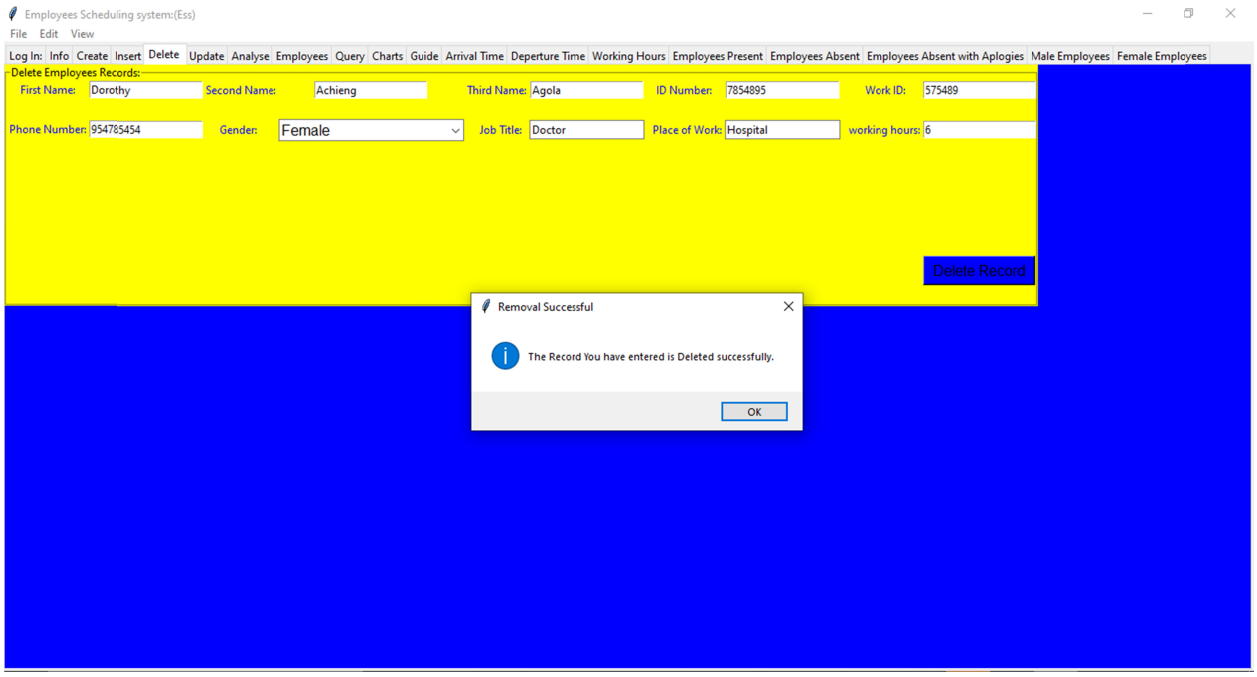


In case you click on the Show Calendar Button and you don't see the calendar just navigate into the status bar and point to the employees scheduling System Icon on the status bar.

Alternatively you can click on Insert tab to create New employees Records.

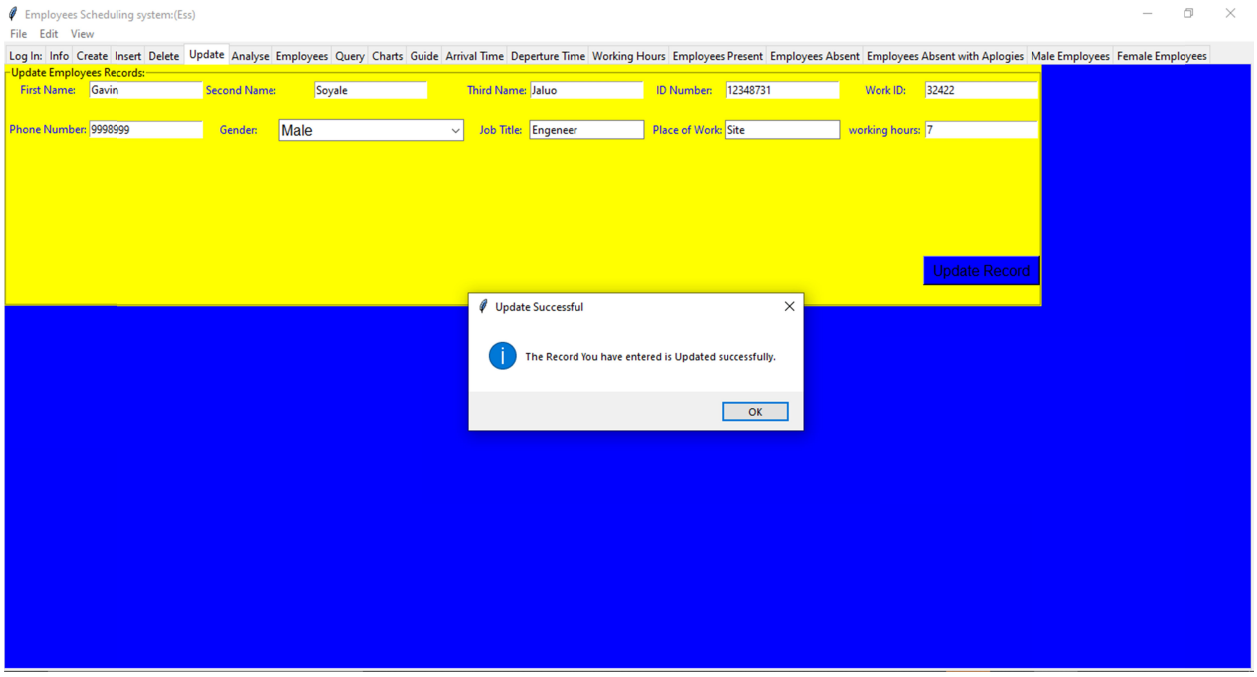


When there is need to delete an Employees Record navigate to the Delete tab and input the employees Details to be Deleted then click on the **Delete Record .**



To update The employees Records With new Records navigate to the Update tab.

Input the Employees Records to be updated then Click on the button **Update Record.**



To run a Query and be able to retreave more summerised information navigate on the query tab and type the query you want to be sent to the database .

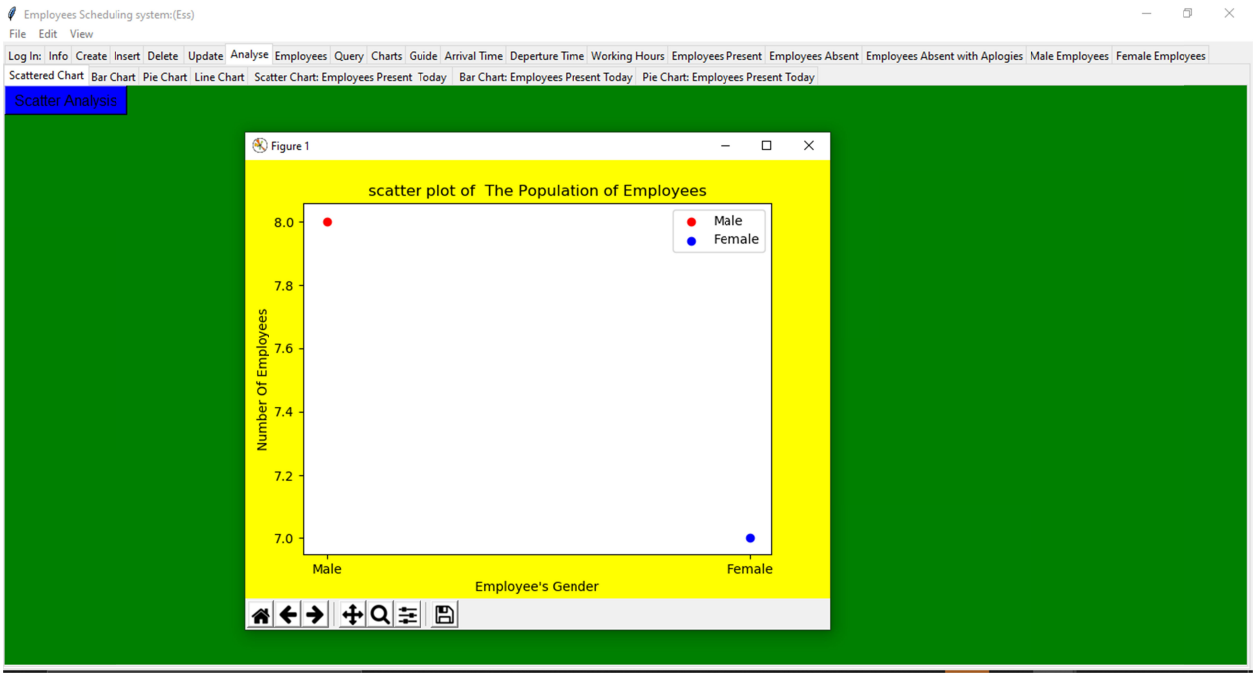
Note :you can only type one complete statement a time .

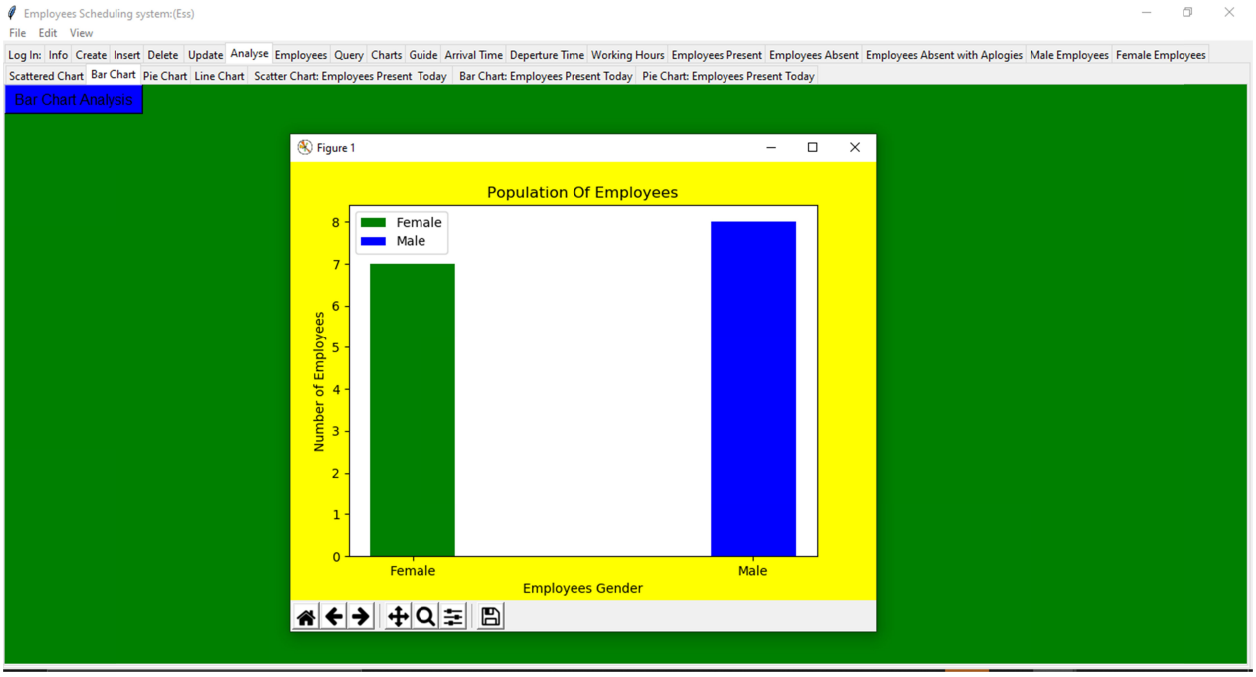
After typing your Query click on enter on the keyboard.

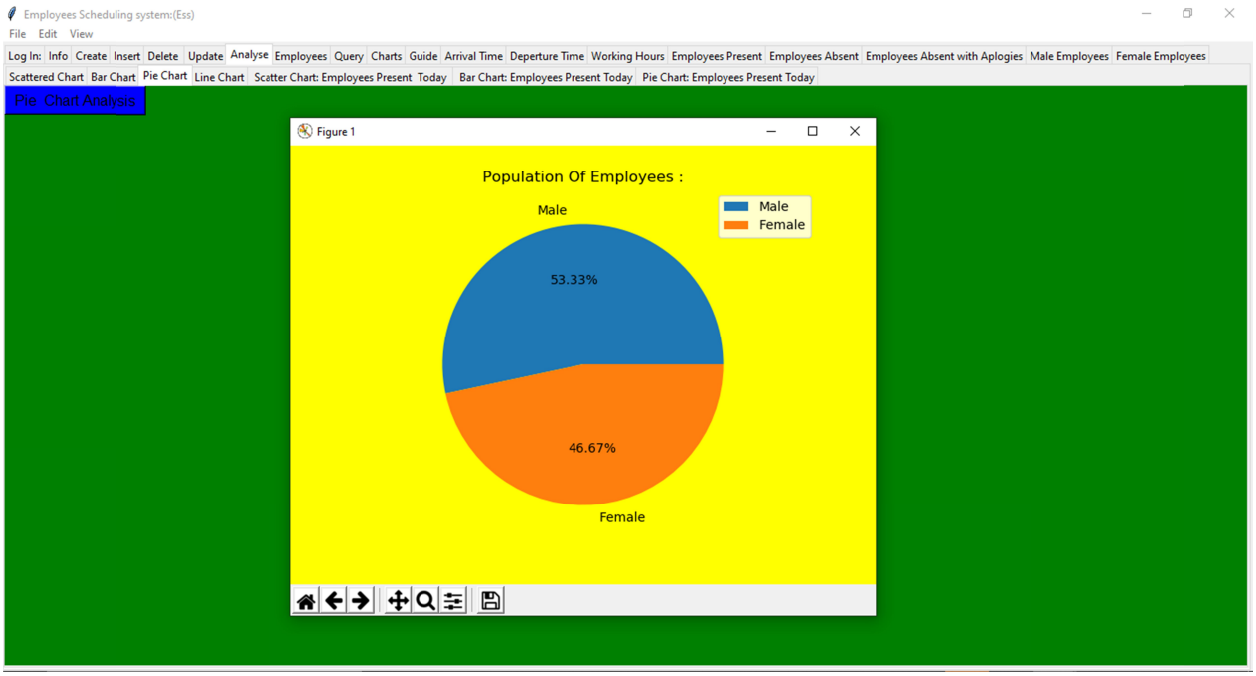
After typing your Query click on the **Run Query .**

Using Data Visualization to Analyse Employees Records click on Analysis tab.

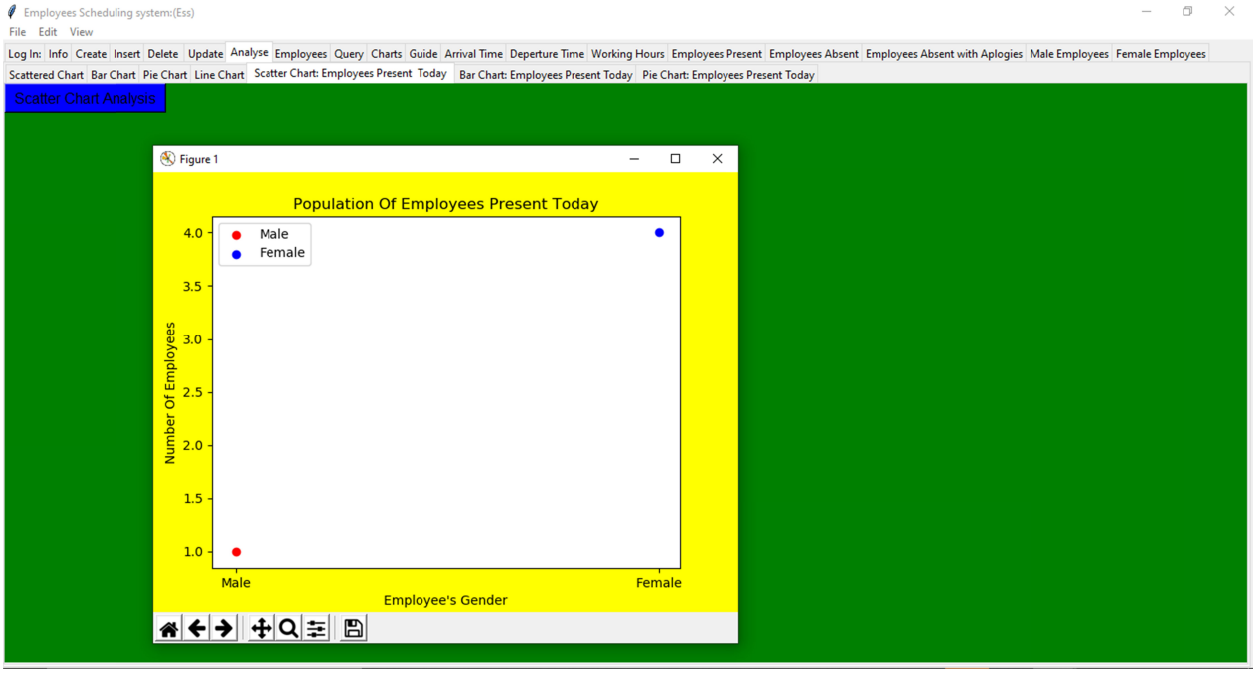
Clicking on the Scatter Chart ,Bar Chart ,Pie Chart ,and Line Chart tabs gives you a graphical representation of employees records detailes interms of population of male and female employees.

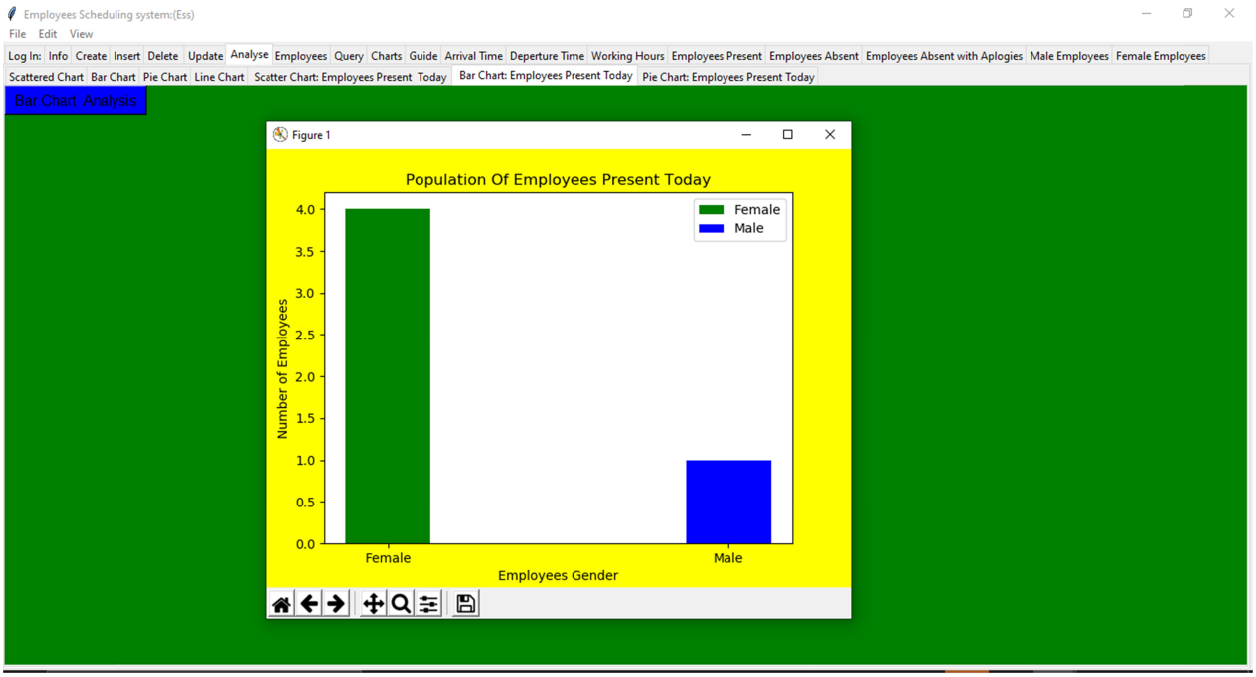

Employees Scheduling system:(Ess)
File Edit View
Log In: Info Create Insert Delete Update Analyse Employees Query Charts Guide Arrival Time Deperture Time Working Hours Employees Present Employees Absent Employees Absent with Aplogies Male Employees Female Employees
Scattered Chart Bar Chart Pie Chart Line Chart Scatter Chart: Employees Present Today Bar Chart: Employees Present Today Pie Chart: Employees Present Today
Scatter Analysis
Figure 1
scatter plot of The Population of Employees
Male
Female
Number Of Employees
8.0
7.8
7.6
7.4
7.2
7.0
Male
Female
Employee's Gender

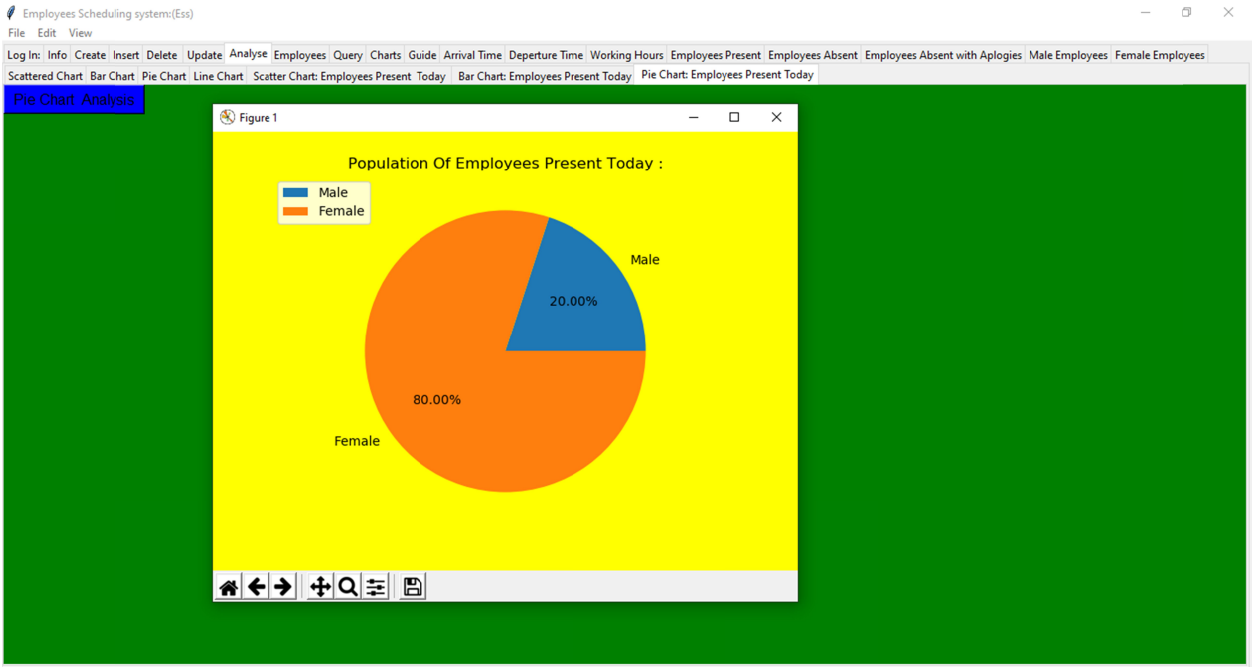

Employees Scheduling system:(Ess)
File Edit View
Log In: Info Create Insert Delete Update Analyse Employees Query Charts Guide Arrival Time Deperture Time Working Hours Employees Present Employees Absent Employees Absent with Aplogies Male Employees Female Employees
Scattered Chart Bar Chart Pie Chart Line Chart Scatter Chart: Employees Present Today Bar Chart: Employees Present Today Pie Chart: Employees Present Today
Bar Chart Analysis
Figure 1
Population Of Employees
Female
Male
Number of Employees
8
7
6
5
4
3
2
1
0
Female
Male
Employees Gender



If you want to Visualise the number Employees present (Those who came to work today) navigate to the tabs Scatter chart :Employees Present Today,Bar chart :Employees Present Today and Pie chart :Employees Present today.



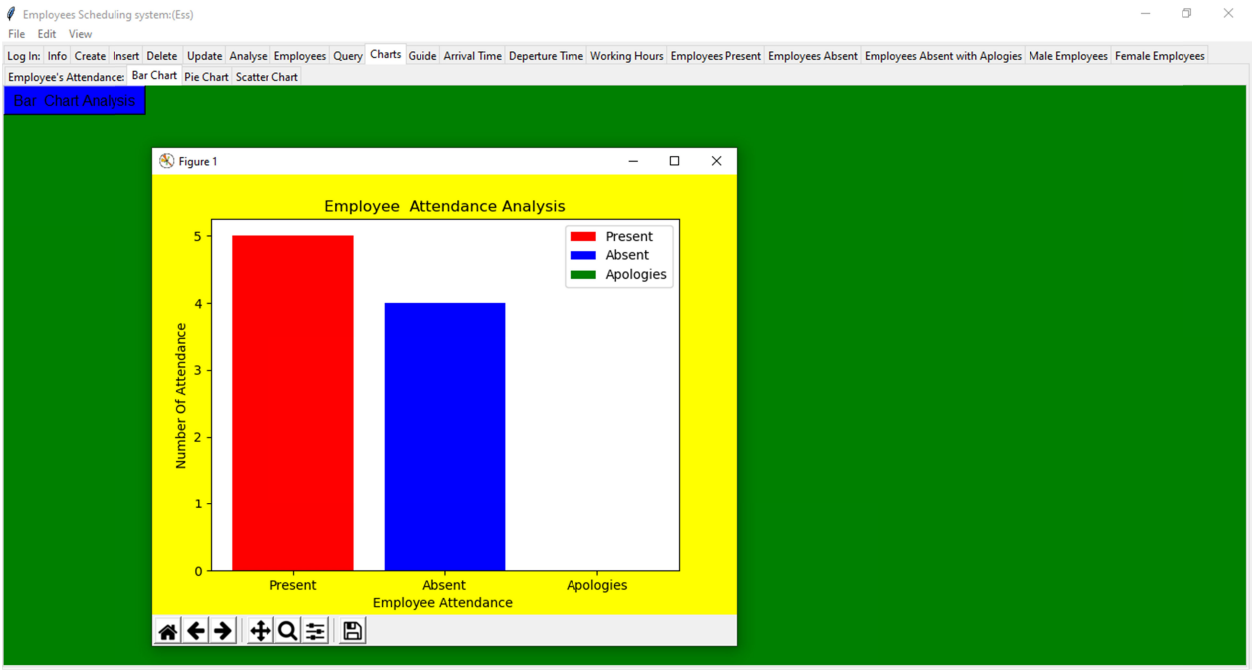


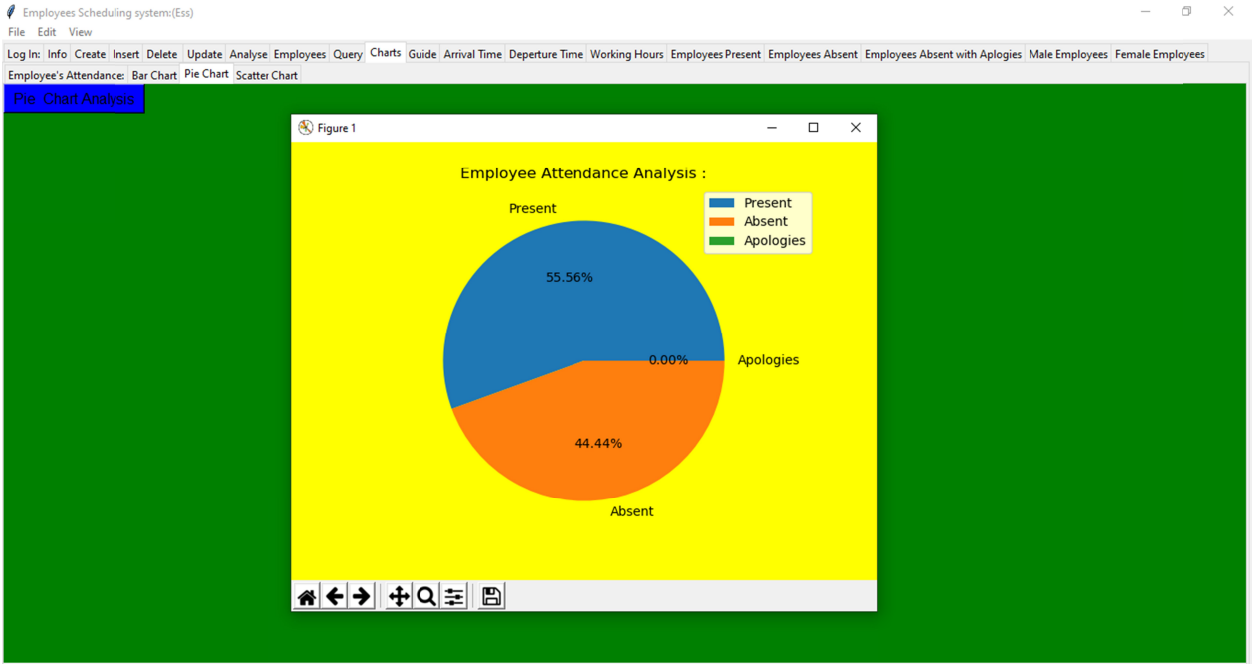


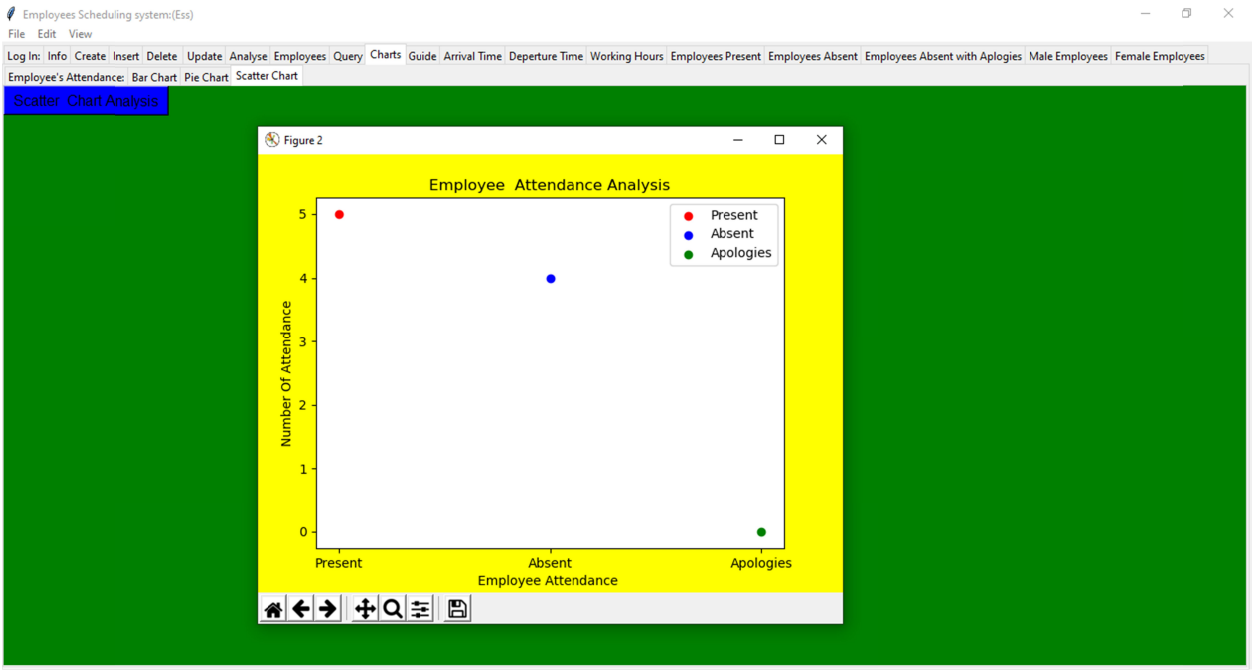
To Perform personalized Employees Attendance analysis click on **charts** tab and input the employees work id to be analysed .

Enter Employees Work id to be Analysed.

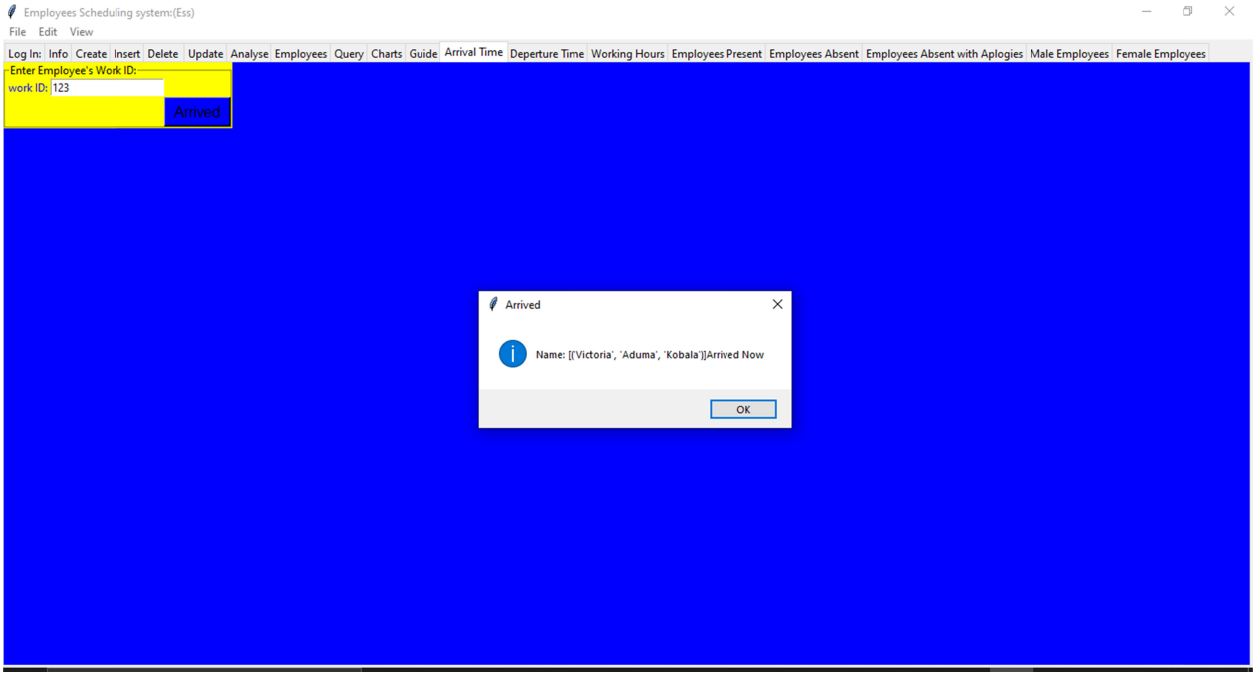
Navigate to the Bar chart,Pie chart and Scatter chart tabs to view employees Results in Visualised charts.


Employees Scheduling system:(Ess)
File Edit View
Log In: Info Create Insert Delete Update Analyse Employees Query Charts Guide Arrival Time Deperture Time Working Hours Employees Present Employees Absent Employees Absent with Aplogies Male Employees Female Employees
Employee's Attendance: Bar Chart Pie Chart Scatter Chart
Bar Chart Analysis
Figure 1
Employee Attendance Analysis
Present
Absent
Apologies
Number Of Attendance
0
1
2
3
4
5
Present
Absent
Apologies
Employee Attendance

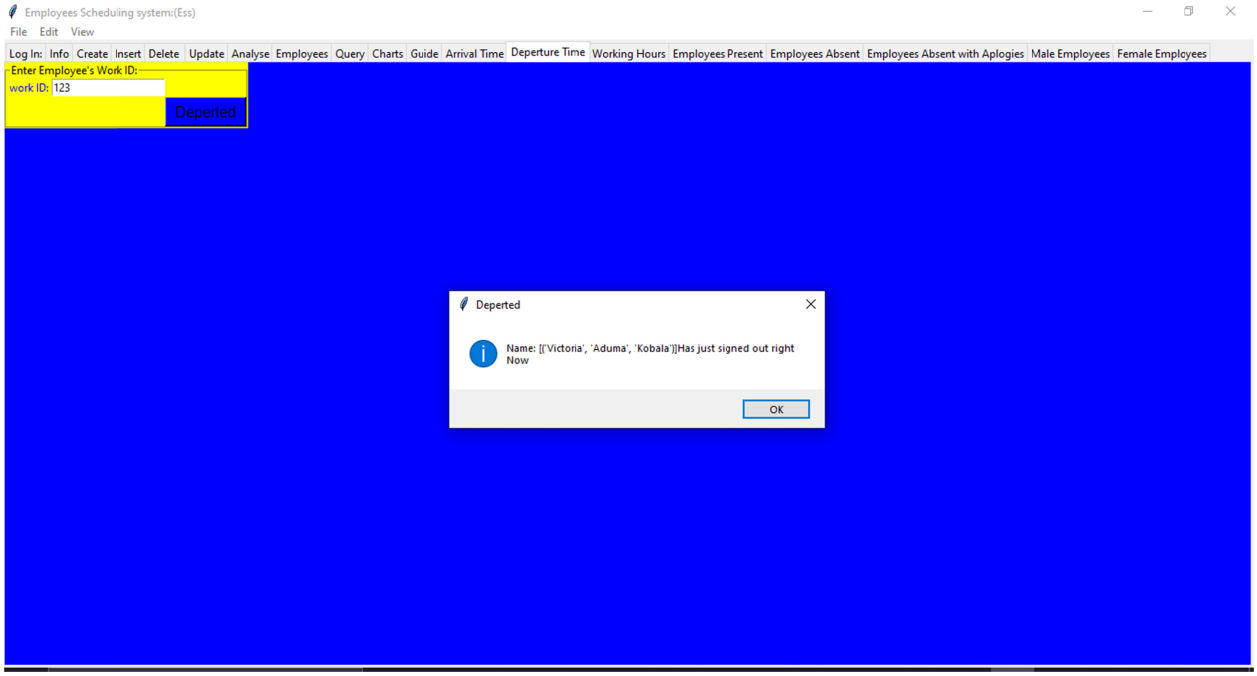

Employees Scheduling system:(Ess)
File Edit View
Log In: Info Create Insert Delete Update Analyse Employees Query Charts Guide Arrival Time Deperture Time Working Hours Employees Present Employees Absent Employees Absent with Aplogies Male Employees Female Employees
Employee's Attendance: Bar Chart Pie Chart Scatter Chart
Pie Chart Analysis
Figure 1
Employee Attendance Analysis :
Present
Absent
Apologies
55.56%
0.00%
44.44%
Apologies
Absent



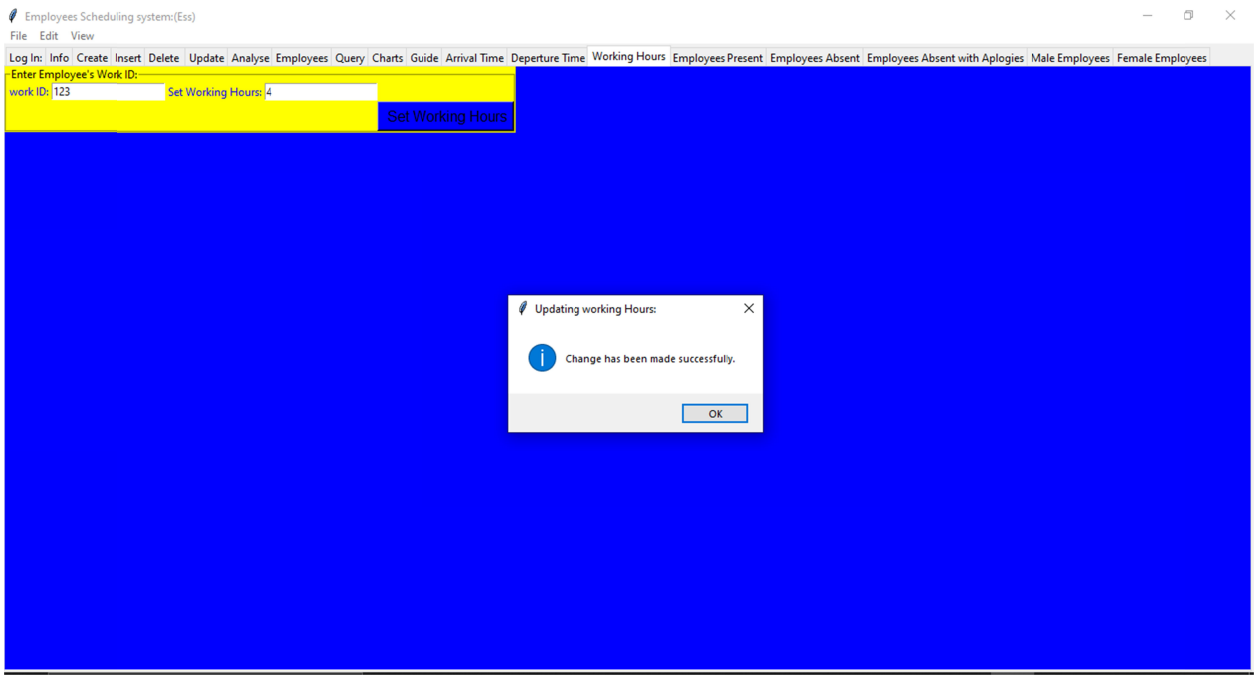
When an employee reports to work navigate to arrival time tab ,enter the work id and press **Arrived** button.



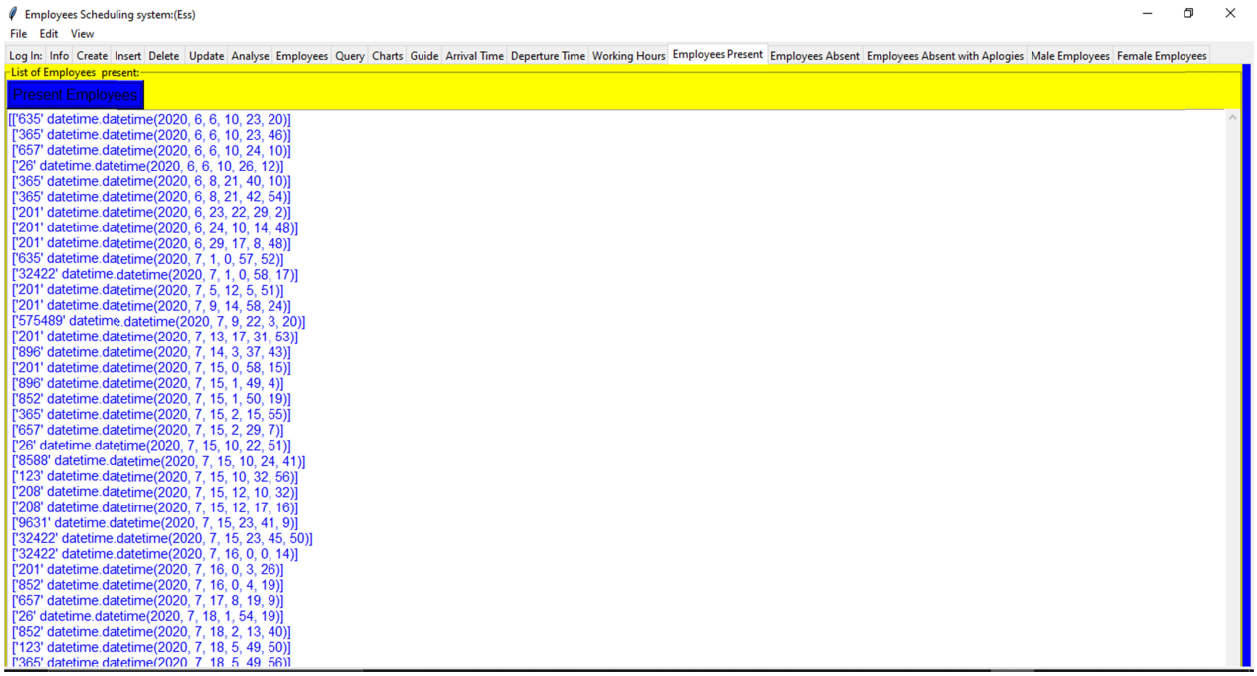
When an employee lives for home navigates to the Departed Time tab and enter the work id of the employee that is living for home.



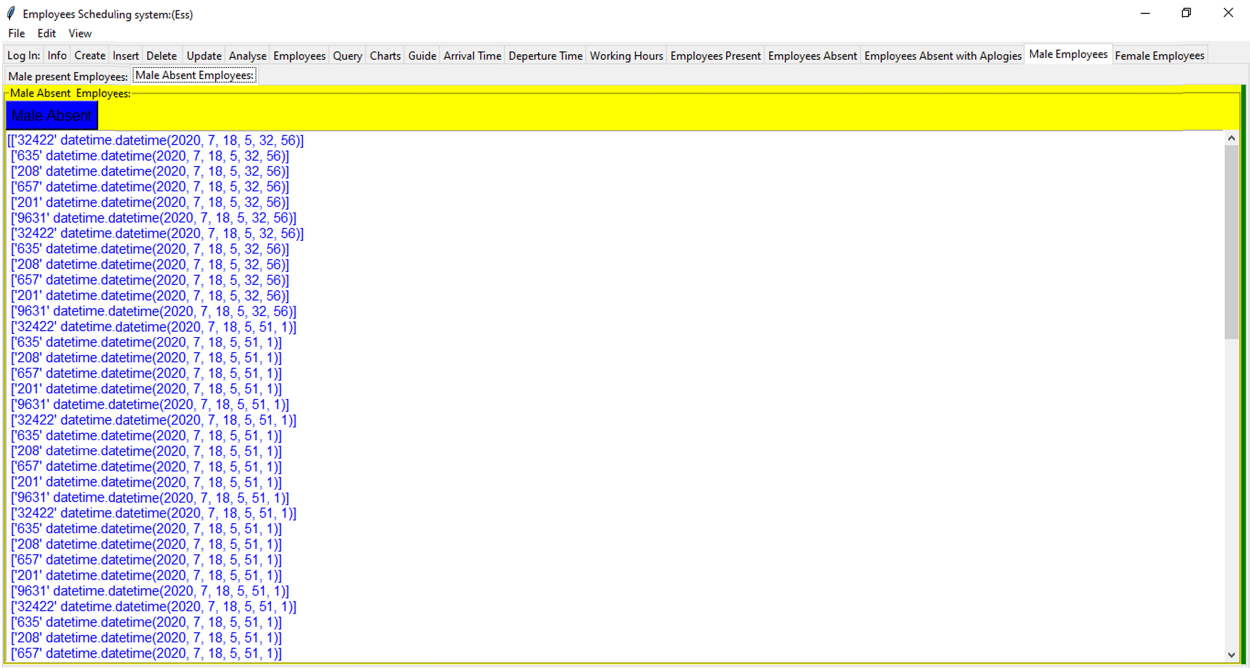
To specifically update the working hours of an Employee navigate to Working hours tab.



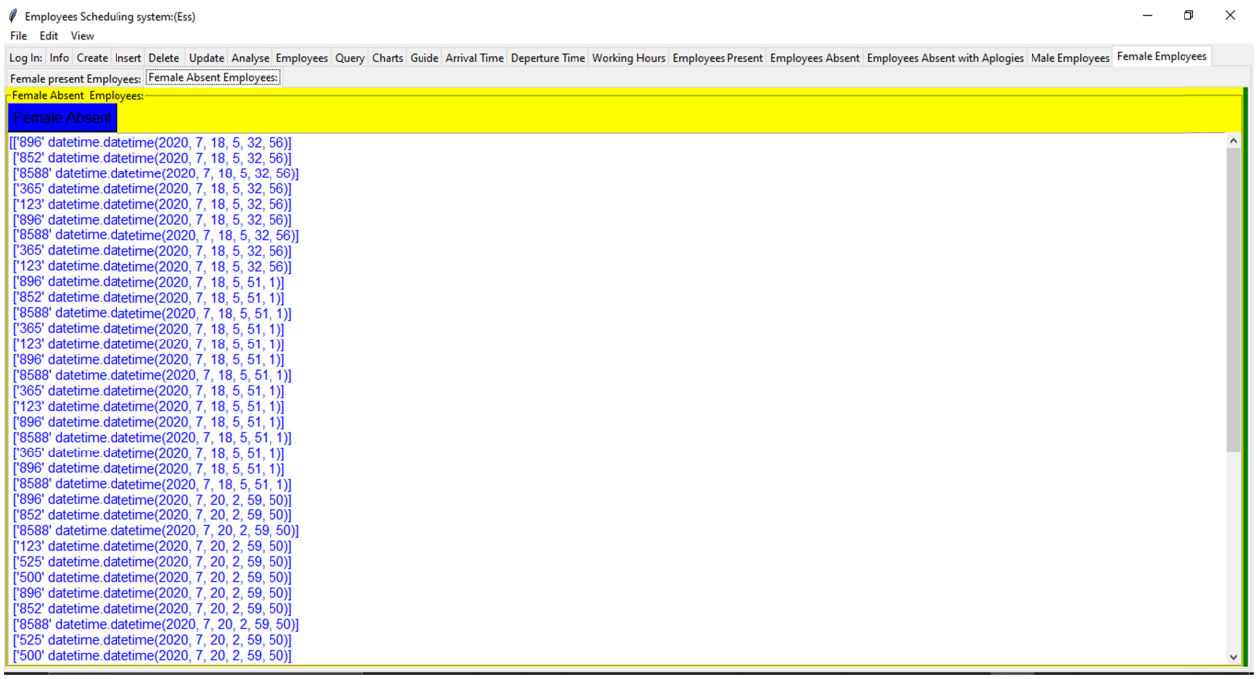
Navigate to Employees Present tab and click on the **Present Employees** button to display the list of Employees present,there work id ,date and time they reported to work.



Navigate to Male Absent Employees and click on to Male Absent to obtain the list of male employees and the date they were absent using there work id.



The same information can also be obtained for the female employees by clicking on the Female Employees tab .



Conclusion

Employees Scheduling system is a good piece of software and will help in keeping employees schedules up to date.However one can make improvements to its Algorithms to even make it much better.

REFRENCES

www.python.org

www.stackoverflow.com

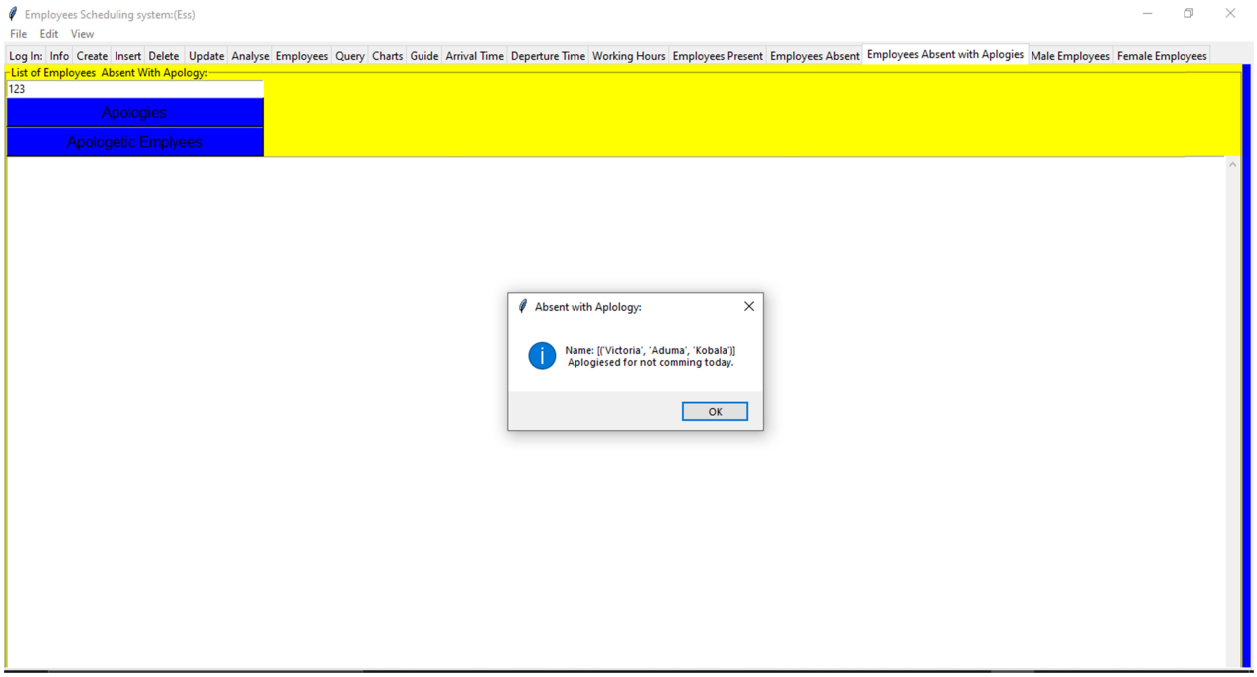
dev.msql.com

To list Employees Absent using there work id and the date and time they were absent click on Employees Absent tab.

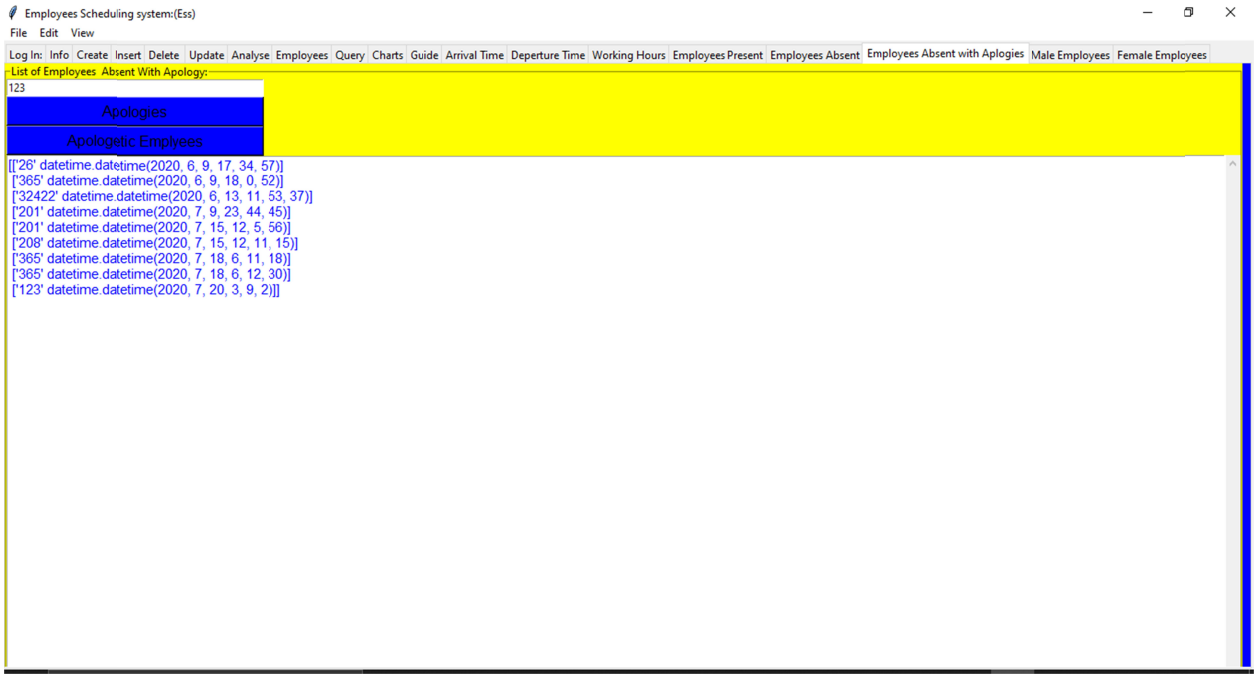
Note : Button Absent Employees can only be pressed once every day .

When dealing with employees that are not coming to work navigate to Employees Absent with Apologies tab.

Enter the Employees work id that has apologiesed for not coming to work.Press Enter.

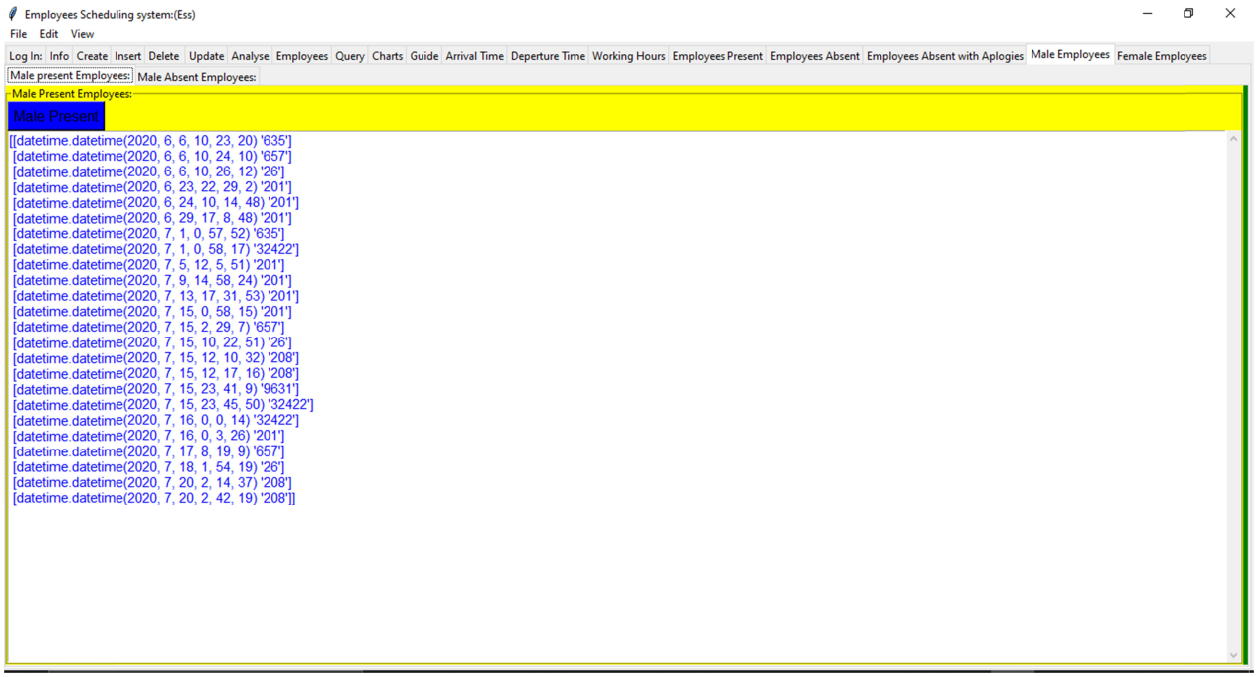


To list the Employees that have apologiesed for not coming to work in the resent past click on **Apologetic Employees.**



To Specifically show the list of Male present and Absent on the Employees tabs click on the Male Employees Tab.

On the male Employees present tab click male present to list Males present using there work id and datetime they came to work.


Employees Scheduling system:(Ess)
File Edit View
Log In: Info Create Insert Delete Update Analyse Employees Query Charts Guide Arrival Time Deperture Time Working Hours Employees Present Employees Absent Employees Absent with Aplogies Male Employees Female Employees
Male present Employees: Male Absent Employees:
Male Present Employees:
Male Present
[[datetime.datetime(2020, 6, 6, 10, 23, 20) '635']
[datetime.datetime(2020, 6, 6, 10, 24, 10) '657']
[datetime.datetime(2020, 6, 6, 10, 26, 12) '26']
[datetime.datetime(2020, 6, 23, 22, 29, 2) '201']
[datetime.datetime(2020, 6, 24, 10, 14, 48) '201']
[datetime.datetime(2020, 6, 29, 17, 8, 48) '201']
[datetime.datetime(2020, 7, 1, 0, 57, 52) '635']
[datetime.datetime(2020, 7, 1, 0, 58, 17) '32422']
[datetime.datetime(2020, 7, 5, 12, 5, 51) '201']
[datetime.datetime(2020, 7, 9, 14, 58, 24) '201']
[datetime.datetime(2020, 7, 13, 17, 31, 53) '201']
[datetime.datetime(2020, 7, 15, 0, 58, 15) '201']
[datetime.datetime(2020, 7, 15, 2, 29, 7) '657']
[datetime.datetime(2020, 7, 15, 10, 22, 51) '26']
[datetime.datetime(2020, 7, 15, 12, 10, 32) '208']
[datetime.datetime(2020, 7, 15, 12, 17, 16) '208']
[datetime.datetime(2020, 7, 15, 23, 41, 9) '9631']
[datetime.datetime(2020, 7, 15, 23, 45, 50) '32422']
[datetime.datetime(2020, 7, 16, 0, 0, 14) '32422']
[datetime.datetime(2020, 7, 16, 0, 3, 26) '201']
[datetime.datetime(2020, 7, 17, 8, 19, 9) '657']
[datetime.datetime(2020, 7, 18, 1, 54, 19) '26']
[datetime.datetime(2020, 7, 20, 2, 14, 37) '208']
[datetime.datetime(2020, 7, 20, 2, 42, 19) '208']]