CONTENTS

INTRODUCTION

This project is a user-friendly hotel management system meant for employee and administrator use. Its functions as a customer detail database and also provides a way for new customers to be added during check-in. Existing customer data can be updated as well; for example, if a customer needs a room change or extension of stay. Finally, this program also calculates the room bill for every room allowing for an easy calculation of the amount to be paid during check-out. It uses an attractive and minimalistic GUI as the interface between user and system. Being simple in build and functionality it is user friendly and easy to operate.

During coding and design of the software Project, Python-3.7 IDE, a powerful front-end tool is used for getting Graphical User Interface (GUI) based integrated platform and coding simplicity. As a back-end a powerful, open-source RDBMS, My SQL is used as per requirement of the CBSE curriculum of Computer Science Course

REQUIREMENT ANALYSIS

2.1) Problem Definition:

To develop a software platform for a hotel named Hotel Sacerdote to automize the check-in and check-out process and digitize their customer records which can be accessed by staff accounts and managed with an admin account.

2.2) Advantages of Proposed System:

This project is a user-friendly hotel management system meant for employee and administrator use. Its functions as a customer detail database and also provides a way for new customers to be added during check-in. Existing customer data can be updated as well; for example, if a customer needs a room change or extension of stay. Finally, this program also calculates the room bill for every room allowing for an easy calculation of the amount to be paid during check-out. It uses an attractive and minimalistic GUI as the interface between user and system. Being simple in build and functionality it is user friendly and easy to operate.

2.3) Project Description:

This program aims to digitize and make simple the check-in and checkout process at Hotel Sacerdote and maintain an easily accessible database for customer details. It manages the following data and interactions:

- Accounts of staff and admin
- Customer/staff details
- Room no and room type
- Room service details

- Calculation and display of bill

SYSTEM ANALYSIS AND DESIGN

3.1) Theoretical background:

- WHAT IS PYTHON?

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

- WHAT IS MySQL?

The management of data in a database system is done by means of a general-purpose software package called a Database Management System (DBMS). Some commercially available RDBMS are MS SQL Server, MS ACCESS, INGRES, ORACLE, and Sybase. MySQL, the most popular Open-Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. It is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL Database Server is very fast, reliable, and easy to use. MySQL Server can handle large databases much faster than existing solutions and is successfully used in highly demanding production environments for

several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.

WHAT IS PYTHON IDLE?

Every Python installation comes with an Integrated Development and Learning Environment, which you'll see shortened to IDLE or even IDE. These are a class of applications that help you write code more efficiently. While there are many IDEs for you to choose from, Python IDLE is very bare-bones, which makes it the perfect tool for a beginning programmer. Python IDLE as an interactive interpreter or as a file editor. The Python shell is an excellent place to experiment with small code snippets. You can access it through the terminal or command line app on your machine. You can simplify your workflow with Python IDLE, which will immediately start a Python shell when you open it. Every programmer needs to be able to edit and save text files. Python programs are files with the .py extension that contain lines of Python code. Python IDLE gives you the ability to create and edit these files with ease. Python IDLE also provides several useful features that you'll see in professional IDEs, like basic syntax highlighting, code completion, and auto-indentation.

WHAT IS TKINTER?

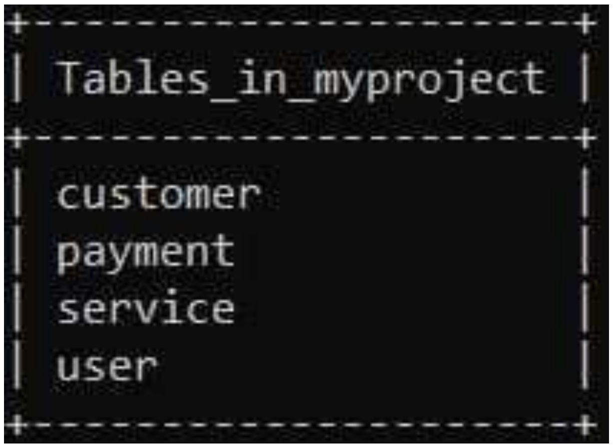
Tkinter stands for Tk interface is a Python binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit, and is Python's de facto standard GUI. Tkinter is included with standard Linux, Microsoft Windows and Mac OS X installs of Python. The name Tkinter comes from Tk interface.

3.2) Table Design:

Database Design-

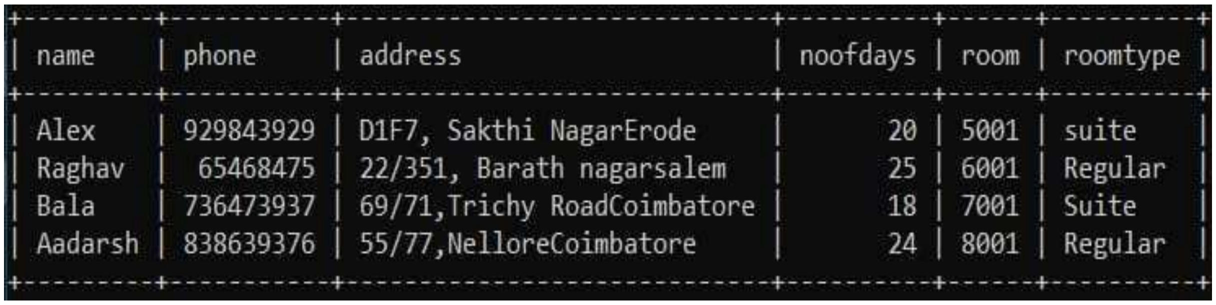
An important aspect of system design is the design of data storage structure. To begin with a logical model of data structure is developed first. A database is a container object which contains tables, queries, reports and data validation policies enforcement rules or constraints etc. A logical data often represented as records are kept in different tables after reducing anomalies and redundancies. The goodness of data base design lies in the table structure and its relationship. This software project maintains a database named ‘myproject’ which contains the following tables.

Database: myproject

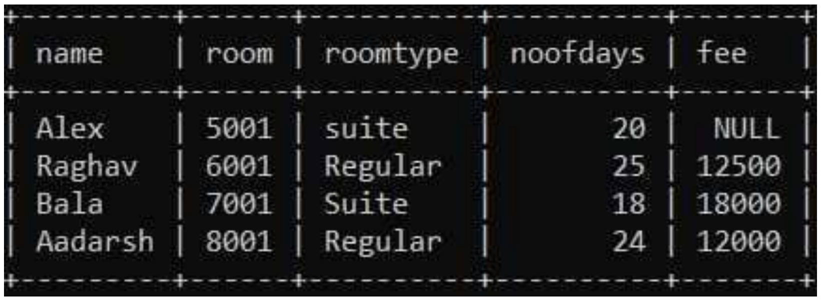


```
+---------------------+
| Tables_in_myproject |
+---------------------+
| customer            |
| payment             |
| service             |
| user                |
+---------------------+
```

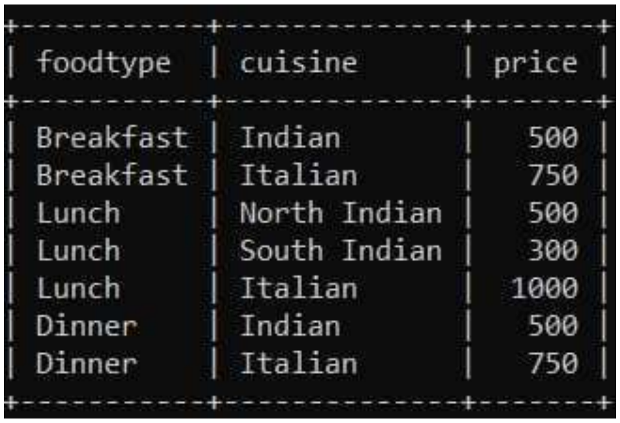
Customer table:


```
+---------+-----------+---------------------------+----------+------+----------+
| name    | phone     | address                   | noofdays | room | roomtype |
+---------+-----------+---------------------------+----------+------+----------+
| Alex    | 929843929 | D1F7, Sakthi NagarErode   |       20 | 5001 | suite    |
| Raghav  |  65468475 | 22/351, Barath nagarsalem |       25 | 6001 | Regular  |
| Bala    | 736473937 | 69/71,Trichy RoadCoimbatore |     18 | 7001 | Suite    |
| Aadarsh | 838639376 | 55/77,NelloreCoimbatore   |       24 | 8001 | Regular  |
+---------+-----------+---------------------------+----------+------+----------+
```

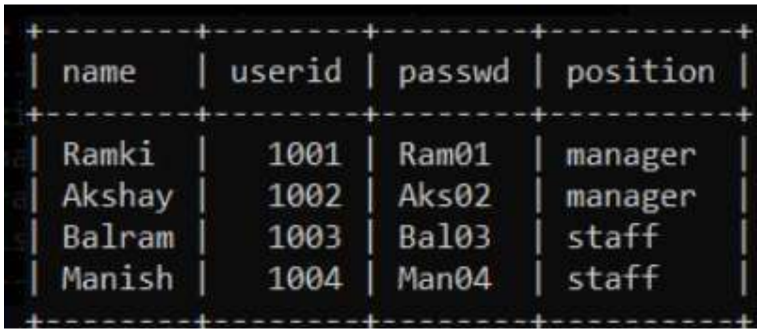
Payment Table:


```
+---------+------+----------+----------+-------+
| name    | room | roomtype | noofdays | fee   |
+---------+------+----------+----------+-------+
| Alex    | 5001 | suite    |       20 |  NULL |
| Raghav  | 6001 | Regular  |       25 | 12500 |
| Bala    | 7001 | Suite    |       18 | 18000 |
| Aadarsh | 8001 | Regular  |       24 | 12000 |
+---------+------+----------+----------+-------+
```

Service Table:


```
+-----------+--------------+-------+
| foodtype  | cuisine      | price |
+-----------+--------------+-------+
| Breakfast | Indian       |   500 |
| Breakfast | Italian      |   750 |
| Lunch     | North Indian |   500 |
| Lunch     | South Indian |   300 |
| Lunch     | Italian      |  1000 |
| Dinner    | Indian       |   500 |
| Dinner    | Italian      |   750 |
+-----------+--------------+-------+
```

User table:


```
+--------+--------+--------+----------+
| name   | userid | passwd | position |
+--------+--------+--------+----------+
| Ramki  |   1001 | Ram01  | manager  |
| Akshay |   1002 | Aks02  | manager  |
| Balram |   1003 | Bal03  | staff    |
| Manish |   1004 | Man04  | staff    |
+--------+--------+--------+----------+
```

3.3) System Requirements:

Hardware requirements:

Intel Core 2 Duo processor or higher

Hard disk: 32GB or higher

RAM: 3GB or higher

Software Requirements:

Operating System: Windows 7 or higher

Python IDLE with Tkinter as frontend

MySQL as backend

Hardware used:

Processor: Intel(R) Core™ i5-8250U CPU @ 1.60GHz 1.8GHz

RAM: 8.00 GB

System type: 64-bit Operating System, x64 based processor

Windows edition: Windows 10 Home Single Language

Software used:

Microsoft Windows® 10 as Operating System.
Python 3.7 as Front-end for GUI development

MySQL as Back-end for data management MS-Word 2010

for documentation

SOURCE CODE

```
from tkinter import *

import tkinter as tk

from tkinter import messagebox

import mysql.connector as sqltor

from PIL import ImageTk

mycon=sqltor.connect(host="localhost",user="root",passwd="root")

mc=mycon.cursor()


mc.execute("create database if not exists myproject")

mc.execute("use myproject")

mc.execute("create table if not exists user(name varchar(30),userid int(6) primary key,passwd
varchar(20),position varchar(20))")

q1="insert into user(name,userid,passwd,position) values(%s,%s,%s,%s)"

v1=[("Ramki",1001,"Ram01","manager"),("Akshay",1002,"Aks02","manager"),("Balram",1003,"Ba
l03","staff"),("Manish",1004,"Man04","staff")]

mc.executemany(q1,v1)

mc.execute("create table if not exists customer(name varchar(30),phone int(15),address
varchar(50),noofdays int(3),room int(10) primary key,roomtype varchar(20) )")

mc.execute("create table if not exists payment as select name,room,roomtype,noofdays from
customer")

mc.execute("alter table payment add fee int")

mc.execute("create table if not exists service(foodtype varchar(30),cuisine varchar(30),price int(6))")

q="insert into service(foodtype,cuisine,price) values (%s,%s,%s)"

v=[("Breakfast","Indian",500),("Breakfast","Italian",750),("Lunch","North
Indian",500),("Lunch","South
Indian",300),("Lunch","Italian",1000),("Dinner","Indian",500),("Dinner","Italian",750)]
```

```
mc.executemany(q,v)

root=Tk()
root.title("Hotel Sacerdote")
root.geometry("1177x615")
root.configure(bg="#00FFFF")
root.resizable(width=False,height=False)
usr=StringVar()
pwd=StringVar()
name=StringVar()
phone=StringVar()
address1=StringVar()
address2=StringVar()
noofdays=StringVar()
roomno=StringVar()
roomtype=StringVar()
khana=StringVar()
mlc=StringVar()
Pri=StringVar()
rtno=StringVar()

BGImage = ImageTk.PhotoImage(file = r"C:\Users\Asus\Downloads\test\hotelbg1.jpg")
BG = Label(root,image = BGImage)
BG.place(x=0,y=0)

def check(rt):
 us=usr.get()
 pw=pwd.get()
 mc.execute('select * from user')
 r=mc.fetchall()
 for i in r:
   us=int(us)
   if i[1]==us and i[2]==pw:
       if i[3]=='manager' or i[3]=='staff' :
           rt.destroy()
           return mainpage()

 else:
     messagebox.showinfo('warning','incorrect username or password')

def rtype(a,ch):
   if a=="Suite":
       return ch*1000
   if a=="Regular":
       return ch*500



def mainpage(*event):


   root1 = Tk()
   root1.title('reservation booking')
   root1.geometry("1022x609")
   root1.configure(bg="#FAD8AD")
   root1.resizable(width=False,height=False)

   BGImage1 = ImageTk.PhotoImage(file = r"C:\Users\Asus\Downloads\test\hall.jpg")
```

```
    BG1 = Label(root1,image = BGImage1)
    BG1.place(x=0,y=0)


    newcust=Button(root1,text='New Customer',padx=10,pady=30,height = 1,width =
10,bg='#FFD58F',fg='black',command=lambda:booking(root1))
    newcust.place(x=300,y=50)
    newcust.tkraise()
    detail=Button(root1,text='Room Details',padx=10,pady=30,height = 1,width =
10,bg='#FFD58F',fg='black',command=lambda:details(root1))
    detail.place(x=600,y=50)
    detail.tkraise()
    payment=Button(root1,text='Check Out',padx=10,pady=30,height = 1,width =
10,bg='#FFD58F',fg='black',command=lambda:pay(root1))
    payment.place(x=300,y=350)
    payment.tkraise()
    display1=Button(root1,text='Staff Details',padx=10,pady=30,height = 1,width =
10,bg='#FFD58F',fg='black',command=lambda:display(root1))
    display1.place(x=600,y=350)
    display1.tkraise()
    roomsv=Button(root1,text='Room Service',padx=10,pady=30,height = 1,width =
10,bg='#FFD58F',fg='black',command=lambda:rservice(root1))
    roomsv.place(x=465,y=200)
    roomsv.tkraise()


    root1.mainloop()




def booking(r):
    def submit(rt):
        na=l1.get()

        ph=l2.get()
        p=int(ph)
        a1=l3.get()
        a2=l4.get()
        ad=a1+a2
        che=l5.get()
        ch=int(che)
        rn=l6.get()
        r=int(rn)
        ty=l7.get()
        f=rtype(ty,ch)

        q2="insert into customer(name,phone,address,noofdays,room,roomtype)
values(%s,%s,%s,%s,%s,%s)"

        q3="insert into payment(name,room,roomtype,noofdays,fee) values(%s,%s,%s,%s,%s)"

        v2=(na,p,ad,ch,r,ty)
        v3=(na,r,ty,ch,f)
        mc.execute(q2,v2)
        mc.execute(q3,v3)
        mycon.commit()
        rt.destroy()


    root2=Tk()
```

```
root2.geometry("1100x1100")
root2.configure(bg="#FAD8AD")
root2.resizable(width=False,height=False)
x0=Label(root2,text='CUSTOMER DETAILS',font=('arial',14),bg='#FFFF80')
x0.place(x=450,y=110)
x1=Label(root2,text='NAME',font=('arial',14),bg='#FFFF80')
x1.place(x=395,y=210)
x2=Label(root2,text='PHONE NUMBER',font=('arial',14),bg='#FFFF80')
x2.place(x=395,y=260)
x3=Label(root2,text='ADDRESS(in 2 lines)',font=('arial',14),bg='#FFFF80')
x3.place(x=395,y=310)
x4=Label(root2,text='NO OF DAYS',font=('arial',14),bg='#FFFF80')
x4.place(x=395,y=410)
x5=Label(root2,text='ROOM-NO',font=('arial',14),bg='#FFFF80')
x5.place(x=395,y=460)
x6=Label(root2,text='ROOM TYPE',font=('arial',14),bg='#FFFF80')
x6.place(x=395,y=510)
x7=Label(root2,text='(Suite/Regular)',font=('arial',10),bg='#FFFF80')
x7.place(x=395,y=530)

l1=Entry(root2,width=20,bd=3,textvariable=name,font=('arial',14),bg='#FFFF80')
l1.place(x=610,y=210)
l2=Entry(root2,width=20,bd=3,textvariable=phone,font=('arial',14),bg='#FFFF80')
l2.place(x=610,y=260)
l3=Entry(root2,width=20,bd=3,textvariable=address1,font=('arial',14),bg='#FFFF80')
l3.place(x=610,y=310)
l4=Entry(root2,width=20,bd=3,textvariable=address2,font=('arial',14),bg='#FFFF80')
l4.place(x=610,y=360)
```

```
    l5=Entry(root2,width=20,bd=3,textvariable=noofdays,font=('arial',14),bg='#FFFF80')
    l5.place(x=610,y=410)
    l6=Entry(root2,width=20,bd=3,textvariable=roomno,font=('arial',14),bg='#FFFF80')
    l6.place(x=610,y=460)
    l7=Entry(root2,width=20,bd=3,textvariable=roomtype,font=('arial',14),bg='#FFFF80')
    l7.place(x=610,y=510)

    back=Button(root2,text='Entry',padx=40,pady=20,height = 1,width =
5,bg='#FFD58F',fg='black',command=lambda:submit(root2))
    back.place(x=620,y=570)



def details(r):

    root2=Tk()
    root2.geometry("1000x1000")
    frame=Frame(root2,height=700,width=1000,bg='#d3d3d3')
    frame.place(x=0,y=0)
    mc.execute('select * from customer')
    r1=mc.fetchall()

    listBox=Text(root2,height = 38, width = 155,fg="white",bg="blue",font="Helvetica")
    listBox.grid(row = 2,column= 0, columnspan = 2)
    listBox.insert(END, "  \t\t\t Customer information\t\t\n\n")
    listBox.insert(END," -----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------\n\n")
    listBox.insert(END,"\tName\t\tPhone\t\tAddress\t\t\t\tNo of days\t\tRoom no\t\tRoom Type\n")
    listBox.insert(END," -----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------\n\n")
```

```
    for i in r1:
        listBox.insert(END,"\t")
        listBox.insert(END,i[0])
        listBox.insert(END,"\t\t")
        listBox.insert(END,i[1])
        listBox.insert(END,"\t\t")
        listBox.insert(END,i[2])
        listBox.insert(END,"\t\t\t\t")
        listBox.insert(END,i[3])
        listBox.insert(END,"\t\t")
        listBox.insert(END,i[4])
        listBox.insert(END,"\t\t")
        listBox.insert(END,i[5])
        listBox.insert(END,"\n\n")
        listBox.insert(END," ----------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------\n\n")


def stype(a,ch,ch1):
    mc.execute('select * from service where foodtype = "{}"'.format(a.split()[0]))
    r1=mc.fetchall()
    if len(r1) != 0:
        mc.execute('update payment set fee={}+fee where room={}'.format(ch,ch1))
        messagebox.showinfo('ON THE WAY','ON THE WAY')
    else:
        messagebox.showinfo('warning','invalid input')



def rservice(r):
```

```
root2=Tk()
root2.geometry("1200x1200")
frame=Frame(root2,height=700,width=1000,bg='#d3d3d3')
frame.place(x=0,y=0)

mc.execute('select * from service')
r1=mc.fetchall()
listBox=Text(root2,height = 38, width = 155,fg="white",bg="blue",font="Helvetica")
listBox.grid(row = 2,column= 0, columnspan = 5)
listBox.insert(END, "  \t\t\t Service \t\t\n\n")
listBox.insert(END," ----------------------------------------------------------------------------------------------------
----------------------------------------------------\n\n")
listBox.insert(END,"\tMeal\t\tCuisine\t\tPrice\n")
listBox.insert(END," ----------------------------------------------------------------------------------------------------
------------------------------------------------------\n\n")
for i in r1:
    listBox.insert(END,"\t")
    listBox.insert(END,i[0])
    listBox.insert(END,"\t\t")
    listBox.insert(END,i[1])
    listBox.insert(END,"\t\t")
    listBox.insert(END,i[2])
    listBox.insert(END,"\n\n")
    listBox.insert(END," -------------------------------------------------------------------------------------------------
---------------------------------------------------\n\n")

y=Label(root2,text='Meal and cuisine',font=('arial',12))
y.place(x=800,y=600)
z=Entry(root2,textvariable=khana,font=('arial',12))
```

```
    z.place(x=940,y=600)
    y1=Label(root2,text='Price',font=('arial',12))
    y1.place(x=800,y=625)
    z1=Entry(root2,textvariable=Pri,font=('arial',12))
    z1.place(x=940,y=625)
    y2=Label(root2,text='Room no',font=('arial',12))
    y2.place(x=800,y=650)
    z2=Entry(root2,textvariable=rtno,font=('arial',12))
    z2.place(x=940,y=650)

    #v=int(v)
    #v1=int(v1)
    B1 = Button(root2,text = "Confirm",command = lambda:stype(z.get(),z1.get(),z2.get()))
    B1.place(x=995,y=675)



def pay(r):
    root2=Tk()
    root2.geometry("1000x1000")
    frame=Frame(root2,height=700,width=1000,bg='#d3d3d3')
    frame.place(x=0,y=0)
    mc.execute('select * from payment')
    r1=mc.fetchall()

    listBox=Text(root2,height = 38, width = 155,fg="white",bg="blue",font="Helvetica")
    listBox.grid(row = 2,column= 0, columnspan = 5)
    listBox.insert(END, "  \t\t\t Check out \t\t\n\n")
```

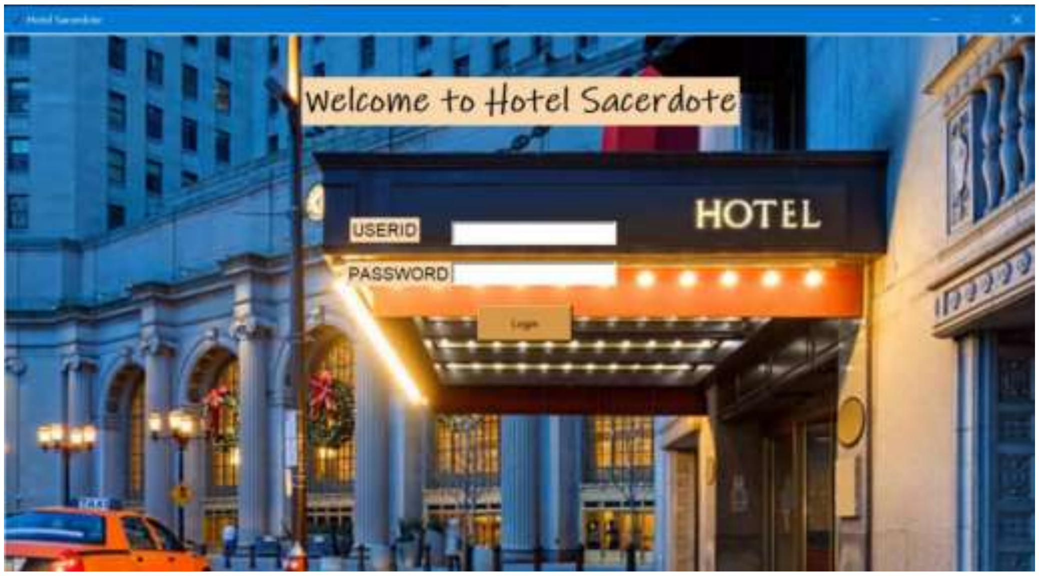
```
   listBox.insert(END," ------------------------------------------------------------------------------------------
--------------------------------------------------\n\n")
   listBox.insert(END,"\tName\t\tRoom\t\tRoom Type\t\tNo of days\t\tFee\n")
   listBox.insert(END," ------------------------------------------------------------------------------------------
----------------------------------------------------\n\n")
   for i in r1:
      listBox.insert(END,"\t")
      listBox.insert(END,i[0])
      listBox.insert(END,"\t\t")
      listBox.insert(END,i[1])
      listBox.insert(END,"\t\t")
      listBox.insert(END,i[2])
      listBox.insert(END,"\t\t")
      listBox.insert(END,i[3])
      listBox.insert(END,"\t\t")
      listBox.insert(END,i[4])
      listBox.insert(END,"\n\n")
      listBox.insert(END," --------------------------------------------------------------------------------------
-------------------------------------------------\n\n")



   e=Label(root2,text = "Select Room Number",font=('arial',15))
   e.place(x=200,y=500)
   E1 = Entry(root2,font=('arial',14))
   E1.place(x=400,y=500)
   B1 = Button(root2,height = 2,width = 15,text = "Confirm",command = lambda:
remove(E1.get(),root2))
   B1.place(x=400,y=550)
```

```
def remove(x,rt):
    mc.execute("delete from payment where room = {}".format(int(x)))
    mc.execute("delete from customer where room = {}".format(int(x)))
    rt.destroy()
    mycon.commit()

def display(r):
    root2 = Tk()
    root2.geometry("1190x650")
    root2.configure(bg="#FAD8AD")
    root2.resizable(width=False,height=False)



    m=Label(root2,text='#ONLY MANAGERS CAN 
ACCESS',font=('Algerian',30,'bold'),bg='#FAD8AD')
    m.place(x=220,y=100)
    m.tkraise()
    h=Button(root2,text='STAFF',padx=30,pady=30,height = 1,width = 
5,bg='#FFD58F',fg='black',command=lambda:staff(root2))
    h.place(x=720,y=420)
    h.tkraise()
    b1=Label(root2,text='USERID',font=('arial',14,),bg='#d3d3d3')
    b1.place(x=395,y=400)
    b1.tkraise()
    a2=Entry(root2,width=20,bd=3,textvariable=usr,font=('arial',13))
    a2.place(x=510,y=400)
    a2.tkraise()
    b2=Label(root2,text='PASSWORD',font=('arial',14),bg='#d3d3d3')
    b2.place(x=390,y=500)
    b2.tkraise()

    a3=Entry(root2,width=20,bd=3,textvariable=pwd,show='*',font=('arial',13))

    a3.place(x=510,y=500)

    a3.tkraise()

    login=Button(root2,text='Login',height = 2,width =
14,bg='#D59949',fg='black',command=lambda:check1(root2))

    login.place(x=540,y=550)

    login.tkraise()


    root2.mainloop()


def check1(rt):

 us=usr.get()

 pw=pwd.get()

 mc.execute('select * from user')

 r=mc.fetchall()

 for i in r:

    us=int(us)

    if i[1]==us and i[2]==pw:

        if i[3]=='manager' or i[3]=='staff' :

            rt.destroy()

            return manager()


 else:

      messagebox.showinfo('warning','incorrect username or password')



def staff(r):
    root3=Tk()
    root3.geometry("1000x1000")
    frame=Frame(root3,height=700,width=1000,bg='#d3d3d3')
    frame.place(x=0,y=0)

    mc.execute('select* from user')
    r1=mc.fetchall()
    listBox=Text(root3,height = 38, width = 155,fg="white",bg="black",font="Helvetica")
    listBox.grid(row = 2,column= 0, columnspan = 5)
    listBox.insert(END, "  \t\t\t Staff details \t\t\n\n")
    listBox.insert(END," -----------------------------------------------------------------------------------------------
-------------------------------------------------\n\n")
    listBox.insert(END,"\tName\t\tUserid\t\tPosition\n")
    listBox.insert(END," -----------------------------------------------------------------------------------------------
---------------------------------------------------\n\n")
    for i in r1:
        listBox.insert(END,"\t")
        listBox.insert(END,i[0])
        listBox.insert(END,"\t\t")
        listBox.insert(END,i[1])
        listBox.insert(END,"\t\t")
        listBox.insert(END,i[3])
        listBox.insert(END,"\n\n")
        listBox.insert(END," ------------------------------------------------------------------------------------------
------------------------------------------------\n\n")

def manager(*event):
    root3=Tk()
    root3.geometry("1000x1000")
    frame=Frame(root3,height=700,width=1000,bg='#d3d3d3')
    frame.place(x=0,y=0)
    mc.execute('select* from user')
    r1=mc.fetchall()
    listBox=Text(root3,height = 38, width = 155,fg="white",bg="black",font="Helvetica")
    listBox.grid(row = 2,column= 0, columnspan = 5)
    listBox.insert(END, "  \t\t\t Staff details \t\t\n\n")
    listBox.insert(END," -------------------------------------------------------------------------------------------------------------------------------------------------------------------\n\n")
    listBox.insert(END,"\tName\t\tUserid\t\tPassword\t\tPosition\n")
    listBox.insert(END," ---------------------------------------------------------------------------------------------------------------------------------------------------------------------\n\n")
    for i in r1:
        listBox.insert(END,"\t")
        listBox.insert(END,i[0])
        listBox.insert(END,"\t\t")
        listBox.insert(END,i[1])
        listBox.insert(END,"\t\t")
        listBox.insert(END,i[2])
        listBox.insert(END,"\t\t")
        listBox.insert(END,i[3])
        listBox.insert(END,"\n\n")
        listBox.insert(END," ---------------------------------------------------------------------------------------------------------------------------------------------------------------\n\n")

wc=Label(root,text='Welcome to Hotel Sacerdote',font=('Ink Free',30, 'bold'),bg='#FAD8AD',)
wc.place(x=340,y=50)
uid=Label(root,text='USERID',font=('arial',14,),bg='#FAD8AD')
uid.place(x=395,y=210)
```

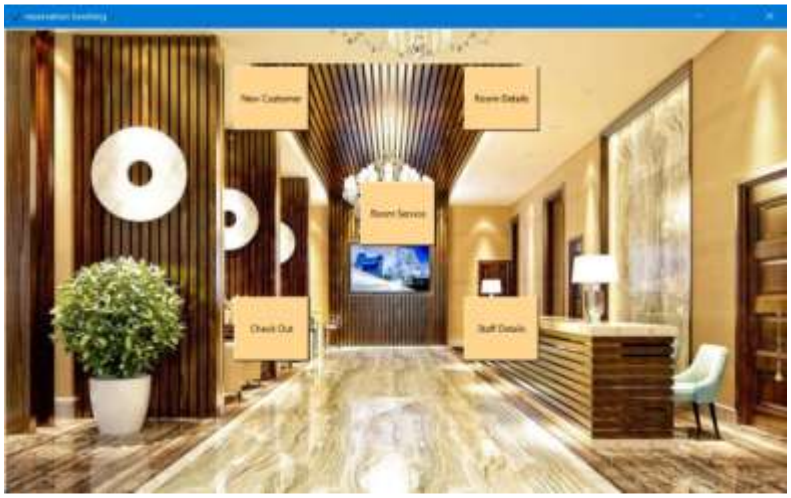
```
a1=Entry(root,width=20,bd=3,textvariable=usr,font=('arial',13))
a1.place(x=510,y=214)
pd=Label(root,text='PASSWORD',font=('arial',14),bg='#FAD8AD')
pd.place(x=390,y=260)
a2=Entry(root,width=20,bd=3,textvariable=pwd,show='*',font=('arial',13))
a2.place(x=510,y=260)
login=Button(root,text='Login',height = 2,width =
14,bg='#D59949',fg='black',command=lambda:check(root))
login.place(x=540,y=310)
root.mainloop()
mc.close()
mycon.close()
```

I/O FORM DESIGNS

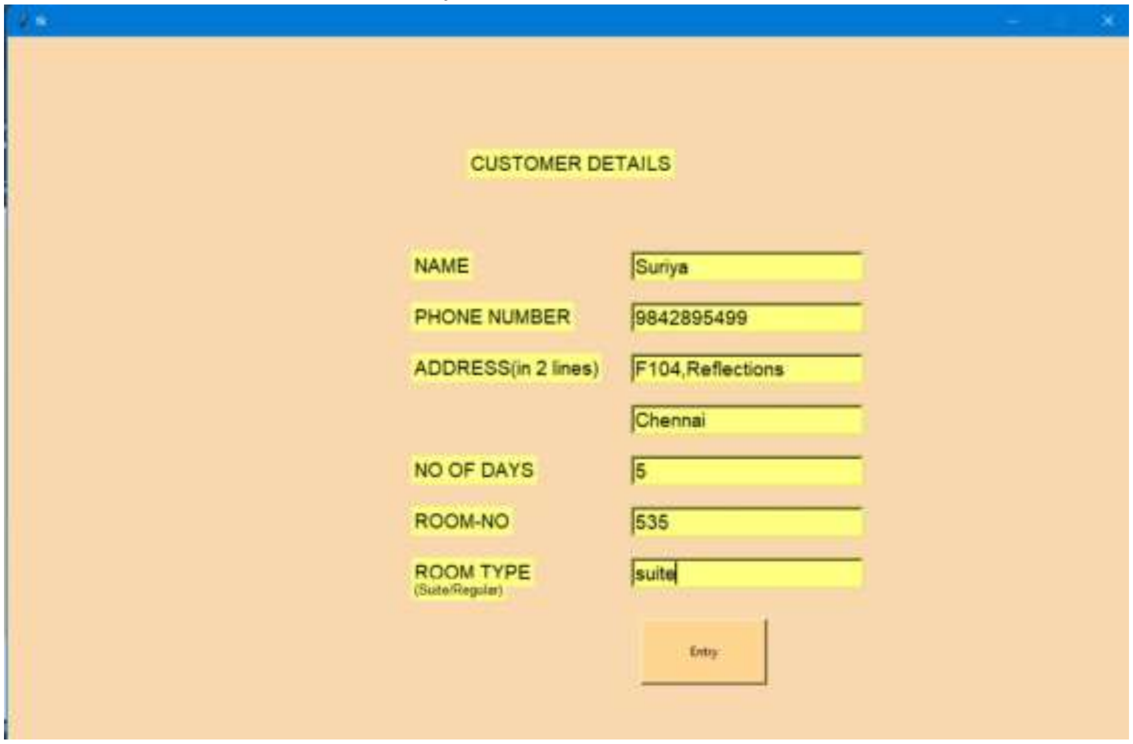
Main login window



Booking window



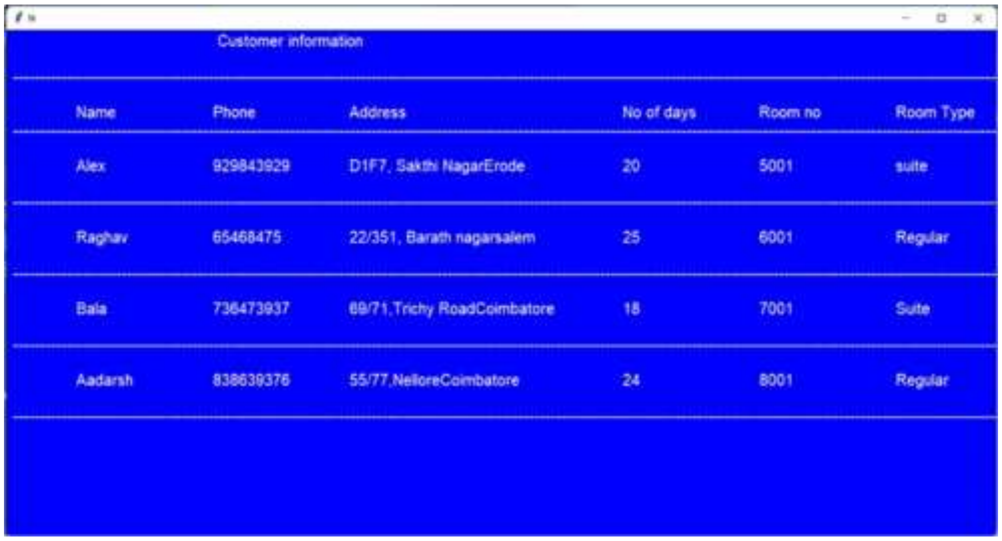
Customer details entry window



CUSTOMER DETAILS

NAME	Suriya
PHONE NUMBER	9842895499
ADDRESS(in 2 lines)	F104,Reflections
	Chennai
NO OF DAYS	5
ROOM-NO	535
ROOM TYPE (Suite/Regular)	suite

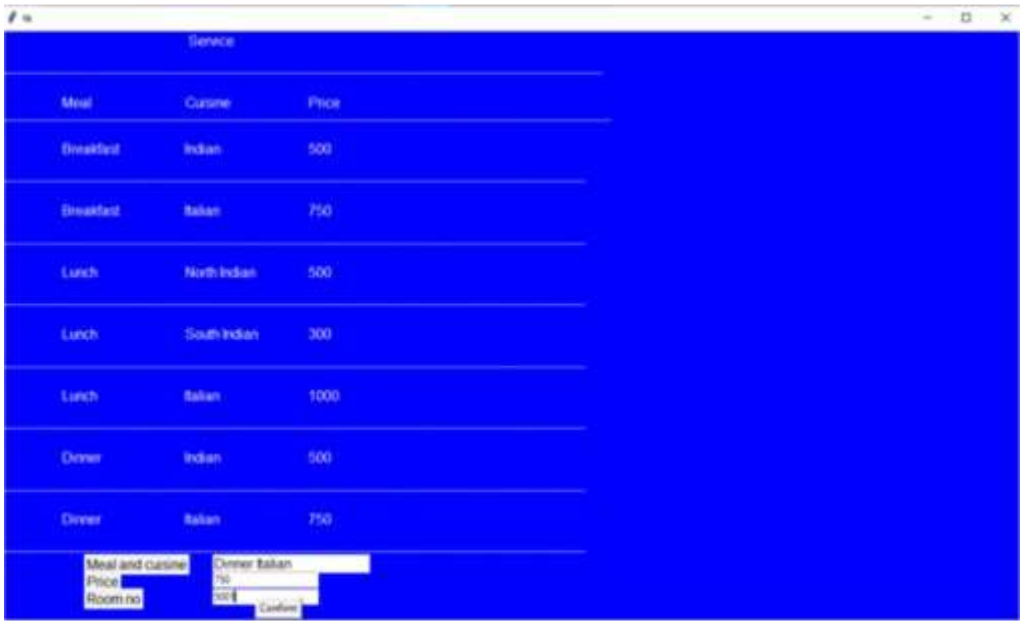
Room details Window



Customer information

Name	Phone	Address	No of days	Room no	Room Type
Alex	929843929	D1F7, Sakthi NagarErode	20	5001	suite
Raghav	65468475	22/351, Barath nagarsalem	25	6001	Regular
Bala	736473937	69/71,Trichy RoadCoimbatore	18	7001	Suite
Aadarsh	838639376	55/77,NelloreCoimbatore	24	8001	Regular

Room service window

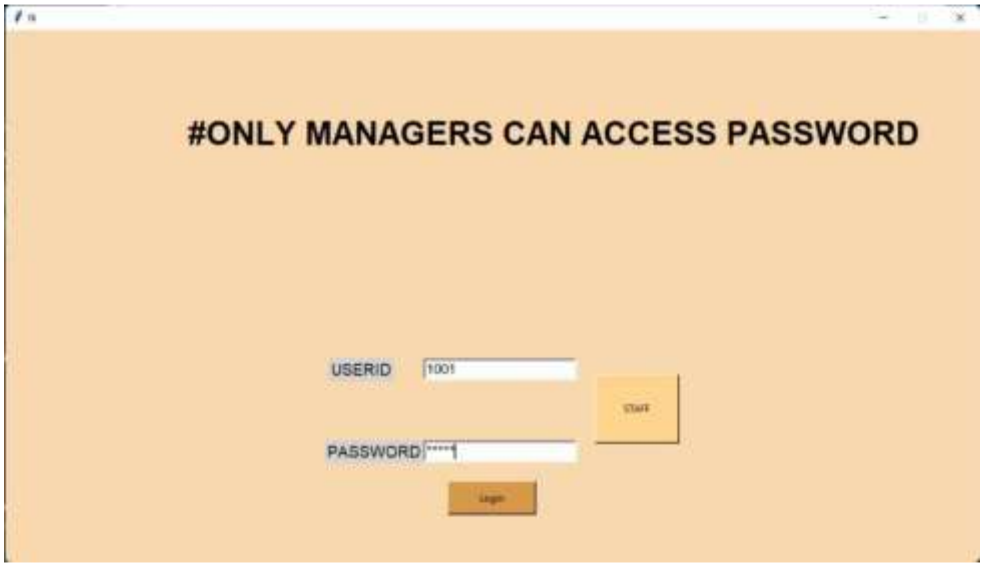


Service

Meal	Cuisine	Price
Breakfast	Indian	500
Breakfast	Italian	750
Lunch	North Indian	500
Lunch	South Indian	300
Lunch	Italian	1000
Dinner	Indian	500
Dinner	Italian	750

Meal and cuisine	Dinner Italian
Price	750
Room no	500

Manager Access Window

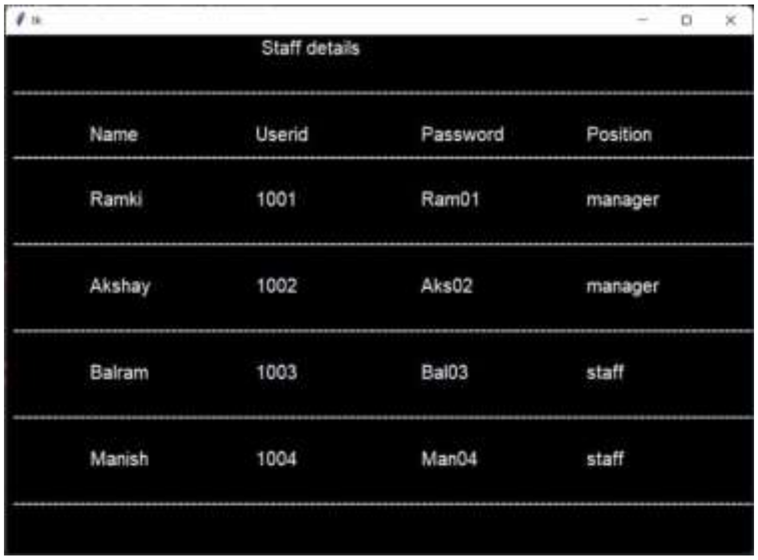


#ONLY MANAGERS CAN ACCESS PASSWORD

USERID 1001

PASSWORD *****

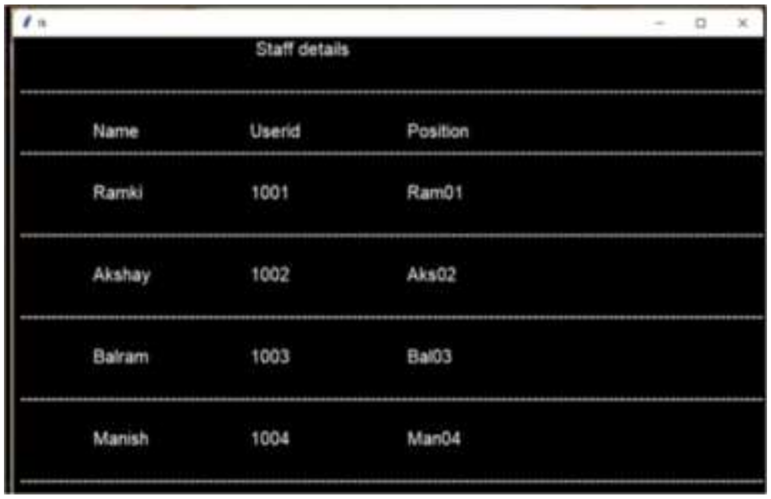
Staff details window (ManagerLogin)



Staff details

Name	Userid	Password	Position
Ramki	1001	Ram01	manager
Akshay	1002	Aks02	manager
Balram	1003	Bal03	staff
Manish	1004	Man04	staff

Staff details window (Staff login)



Staff details

Name	Userid	Position
Ramki	1001	Ram01
Akshay	1002	Aks02
Balram	1003	Bal03
Manish	1004	Man04

CONCLUSION

A Multiuser environment is provided in this application which provides an ease of access.

During the course of project development, a lot of problems were faced with database connectivity of MySQL in Python to access and retrieve data from the tables.

But after completion of the project, all these were clearly understood. The concepts of functions, loops and modules were also understood. This project also included additional learning of new concepts like tkinter and its application.

BIBLIOGRAPHY

BOOKS:

Computer science with python textbook for class XII -Sumitha Arora

-Published by: DHANPAT RAI & CO. (Pvt.) Ltd ; 2021 edition

Computer science with python textbook for class XI -Sumitha Arora

-Published by: DHANPAT RAI & CO. (Pvt.) Ltd ; 2021 edition

WEBSITES:

https://stackoverflow.com/questions/60640638/tkinter-listbox-make-aselection

https://docs.python.org/3/library/dialog.html

https://docs.python.org/3/library/tkinter.ttk.html

https://www.google.com/