A Project Report On

HOTEL MANAGEMENT SYSTEM

Submitted in partial fulfillment
for the award of the degree in

BACHELOR OF COMPUTER APPLICATION

By

ABHISHEK KUMAR SINGH
Enrollment No. AJU/190786/094

Under the esteemed guidance of

MS. SWETA KUMARI BARNWAL

DEPARTMENT OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY

ARKA JAIN UNIVERSITY, JHARKHAND

JAMSHEDPUR

2019-2022

A Project Report On

HOTEL MANAGEMENT SYSTEM

Submitted in partial fulfilment
for the award of the Degree in

BACHELOR OF COMPUTER APPLICATION

By

ABHISHEK KUMAR SINGH
REG NO. AJU/190786/094

Under the esteemed guidance of

MS. SWETA KUMARI BARNWAL

DEPARTMENT OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY

ARKA JAIN UNIVERSITY, JHARKHAND

JAMSHEDPUR

2019-2022

ARKA JAIN UNIVERSITY
JAMSHEDPUR, JHARKHAND
DEPARTMENT OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the project entitled **"HOTEL MANAGEMENT SYSTEM"**, is bona-fide work of **ABHISHEK KUMAR SINGH** bearing Registration Number: **AJU/190786** submitted in partial fulfilment of the requirement for the award of degree in BACHELOR OF COMPUTER APPLICATION from ARKA JAIN UNIVERSITY, JHARKHAND.

Internal Guide

Coordinator

Date: 25/5/22

University Seal

INTERNSHIP CERTIFICATE

ULTRAINFO TECHNOLOGIES
ENGINEERS

Ref. No: ITD-PI-0322-026 **Date:** 09-03-2022

PROJECT INTERNSHIP LETTER

TO WHOM IT MAY CONCERN

This is to certify that Mr. **Abhishek Kumar Singh**, a student of ARKA JAIN University, Jharkhand, Jamshedpur has successfully completed his Project Internship/Training (Virtual) with our organization as a **Python Developer** during the period of January 2022 to February 2022.

The topic of the project was:

"Hotel Management System"

The Frontend and backend used for project development was Python, MySQL.

During this period of internship/training, he was found to be honest, creative & able to perform all his duties perfectly on the project.

We wish him all the best in all his future accomplishments.

For ULTRAINFO TECHNOLOGIES

(Authorized Signatory)

Address: Rourkela, Odisha, India

Website: itdeveloper.in **E-mail:** info@itdeveloper.in

ABSTRACT

The Project Hotel Management System is a single user-based system application that allows the hotel manager to handle hotel activities technically with the help of receptionist. Interactive GUI and the ability to manage hotel bookings and rooms make this system very flexible and convenient.

The hotel manager is a very busy person and does not have the time to sit and manage the entire activities manually on paper. This application gives him the power and flexibility to manage the entire system from a single system.

Hotel Management project provides room booking, hotel details management and other necessary hotel management features. The system allows the manager to post available rooms in the system.

Administrator can view and book room & meal by system. Administrator has the power of either approving or disapproving the customer's booking request.

Hotel services can also be viewed by the administrator and can update them too. The system is hence useful for managers to portably manage the hotel activities.

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to several individuals and organization for supporting me throughout the completion of my project.

First, I wish to express my sincere gratitude to my mentor (**SWETA KUMARI BARNWAL**) for her enthusiasm patience, insightful comments, helpful information, practical advices and unceasing ideas that have helped me tremendously at all times in my Project and writing of this thesis. Her immense knowledge, profound experience and professional expertise in Backend has enabled me to complete this project successfully. Without her support and guidance, this project would not have been possible.

I am also thankful to our respected H.O.D **(Dr. ARVIND KUMAR PANDEY)** and all faculty members for loving inspiration and timely guidance. I also wish to express my sincere thanks to the Department of Computer science & Information technology of ARKA JAIN UNIVERSITY for accepting this project.

Thanks for all your encouragement

DECLARATION

I hereby declare that the Project entitled **"HOTEL MANAGEMENT SYSTEM"** done at ARKA JAIN UNIVERSITY has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

This project is done in partial fulfilment of the requirement for the award of degree of BACHELOR OF COMPUTER APPLICATION to be submitted as final semester project as part of our curriculum.

Abhishek

Signature

(ABHISHEK KUMAR SINGH)

Contents

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Hotel Management System is a system that provides us to reserving rooms, checking whether the rooms are vacant are or not etc by using software. This system is very useful to all especially for business people. For Business people they don't have sufficient time for these then they can use these types of Hotel Management Systems. By this project we will reduce the faults in bills of their expenditure and decrease time of delay to give the bills to the customers. We can also save the bills of the customer. By this project we can also include all the taxes on the bills according to their expenditures. It has a scope to reduce the errors in making the bills. Computerized bill can be printed within fraction of seconds. Booking is possible by using this software. This Project is based on python. If anyone wants to book the room for few days then they can specify the specific number by seeing the types of rooms we have. The bill of this booking is based on the type of room they can select is displayed.

OBJECTIVE:

- Easy maintenance of database and overall project.
- Adopting security measures, by maintaining the login of username and password.
- Reducing data redundancy.
- Reduce workload of staff.
- Reduce the delay in processing time.
- Provide user-friendliness in all possible ways.
- Store data in a centralized location to reduce redundancy and increase consistency.

PURPOSE AND SCOPE:

The manual work processes were time consuming and hence slow. Following are the main drawbacks of the existing system:

- The basic and major drawbacks in the existing system are the speed of retrieval of

data from files, which leads to delay.

- Maintenance of voluminous data is very cumbersome and laborious job.

- There are plenty of chances of duplicity of data andinformation.
- Updating is very tedious job.
- There is no central database from where one can get different statistical data at one place

- The purpose of this project is to overcome these problems by automating the system. Automation of the data maintenance would reduce the manpower and will result in accurate data & above all increase the efficiency of the system.

CHAPTER 2
SURVEY OF TECHNOLOGIES

Before beginning the project, a list of available software as well as hardware technologies was made.

In software we had an array of system-based application to choose but looking at the benefits and adaptability we went with python instead any other technology as there is plenty of good reasons to use python when developing an application. High speed, low cost, and vast language support are among the most significant benefits. python is built into the familiar Windows server environment, requiring less setup and configuration than other development platforms that must be installed and configured separately. The popularity of python makes online resources and skilled developers easy to find.

For database management we went with Microsoft SQL Server as it is really robust, powerful and has a very friendly GUI.

CHAPTER 3
REQUIREMENTS AND SYSTEM ANALYSIS

3.1. System Analysis

System analysis is a process of gathering and interpreting facts, diagnosing problems and the information to recommend improvements on the system. It is a problem-solving activity that requires intensive communication between the system users and system developers. System analysis or study is an important phase of any system development process. The system is studied to the minutest detail and analyzed. The system analyst plays the role of the interrogator and dwells deep into the working of the present system. The system is viewed as a whole and the input to the system are identified. The outputs from the organizations are traced to the various processes. System analysis is concerned with becoming aware of the problem, identifying the relevant and decisional variables, analyzing and synthesizing the various factors and determining an optimal or at least a satisfactory solution or program of action. A detailed study of the process must be made by various techniques like interviews, questionnaires etc. The data collected by these sources must be scrutinized to arrive to a conclusion. The conclusion is an understanding of how the system functions. This system is called the existing system. Now the existing system is subjected to close study and problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal. Preliminary study is the process of gathering and interpreting facts, using the information for further studies on the system. Preliminary study is problem solving activity that requires intensive communication between the system users and system developers. It does various feasibility studies. In these studies, a rough figure of the system activities can be obtained, from which the decision about the strategies to be followed for effective system study and analysis can be taken.

3.2. Existing System

In the existing system the exams are done only manually but in proposed system we have to computerize the exams using this application.

- Lack of security of data.
- More man power.
- Time consuming.
- Consumes large volume of pare work.
- Needs manual calculations.
- No direct role for the higher officials

3.3. Proposed System

The aim of proposed system is to develop a system of improved facilities. The proposed system can overcome all the limitations of the existing system. The system provides proper security and reduces the manual work.

- Security of data.
- Ensure data accuracies.
- Proper control of the higher officials.
- Minimize manual data entry.
- Minimum time needed for the various processing.
- Greater efficiency.
- Better service.
- User friendliness and interactive.
- Minimum time required

3.4. Feasibility study

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spend on it. Feasibility study lets the developer foresee the future of the project and the usefulness. A feasibility study of a system proposal is according to its workability, which is the impact on the organization, ability to meet their user needs and effective use of resources. Thus, when a new application is proposed it normally goes through a feasibility study. The document provides the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as Technical, Economic and Operational feasibilities. The following are its features:

3.4.1. Technical Feasibility

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs and procedures. Having identified an outline system, the investigation must go on to suggest the type of equipment, required method developing the system, of running the system once it has been designed.

3.4.2. Economic Feasibility

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.
The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also, all the resources are already available, it gives an indication of the system is economically possible for development.

3.4.3. Operational Feasibility

In this feasibility study it is determined whether there is need of well qualified operator or simple user. Is there need to train the operator or not? This project is supporting the Graphical User Interface; hence operating this project is so simple. Even a person who has a little knowledge of computer can easily handle this well. There is no need of trained operator.

3.5. Hardware Requirements

Name of Component	Specification
Processor	1.6 GHz or faster processor
RAM	1 GB of RAM (1.5 GB if running on a virtual machine)
Required hard disk drive	5400 RPM hard disk drive
Required hard disk space	10 GB of available hard disk space

3.6. Software Requirements

Name of Component	Specification
Operating System	Windows 7 and above
Front End	Python
Back End	Python
Database	MS SQL Server 2019

3.7. Conceptual Models

Conceptual models differ in the degrees of freedom given to the user in accomplishing tasks and achieving goals. In some cases, the task flow is highly structured and linear, thus giving the user less degrees of freedom in terms of alternative ways of the interaction. In other cases, the task flow is unstructured, thus giving the user more degree of freedom in terms of alternatives in how to perform the interaction.

3.7.1. SDLC

In order to develop the project **"HOTEL MANAGEMENT SYSTEM"** we have adopted the **Iterative Enhancement Model** also known as Incremental Model. This model removes the shortcoming of waterfall model. Since many facts of this system are already known. It is not a new concept and hence no research is required. A working version can be easily created and hence the system can start working. Rest of the functionalities can be implemented in the next iteration and can be delivered later. As the requirement analysis is also not required. It not being a new technology risk involved is also less. So, one need not perform detailed risk analysis. If redevelopment Admin is less than development can be started with a smaller number of people and in next increments others can be involved. As this model combines the advantage of waterfall model and prototyping, clients are always aware of the product being delivered and can always suggest changes and enhancements and can get them implemented. As less amount of customer communication is required one need not apply spiral model in which all types of analysis is done in detail. As the deadline is affordable one need not to for Rapid Application

Development model. Iterative enhancement model is useful when less manpower is available for software development and the release deadlines are specified. It is best suited for in house product development, where it is ensured that the user has something to start with. The complete product is divided into releases and the developer delivers the product release by release.

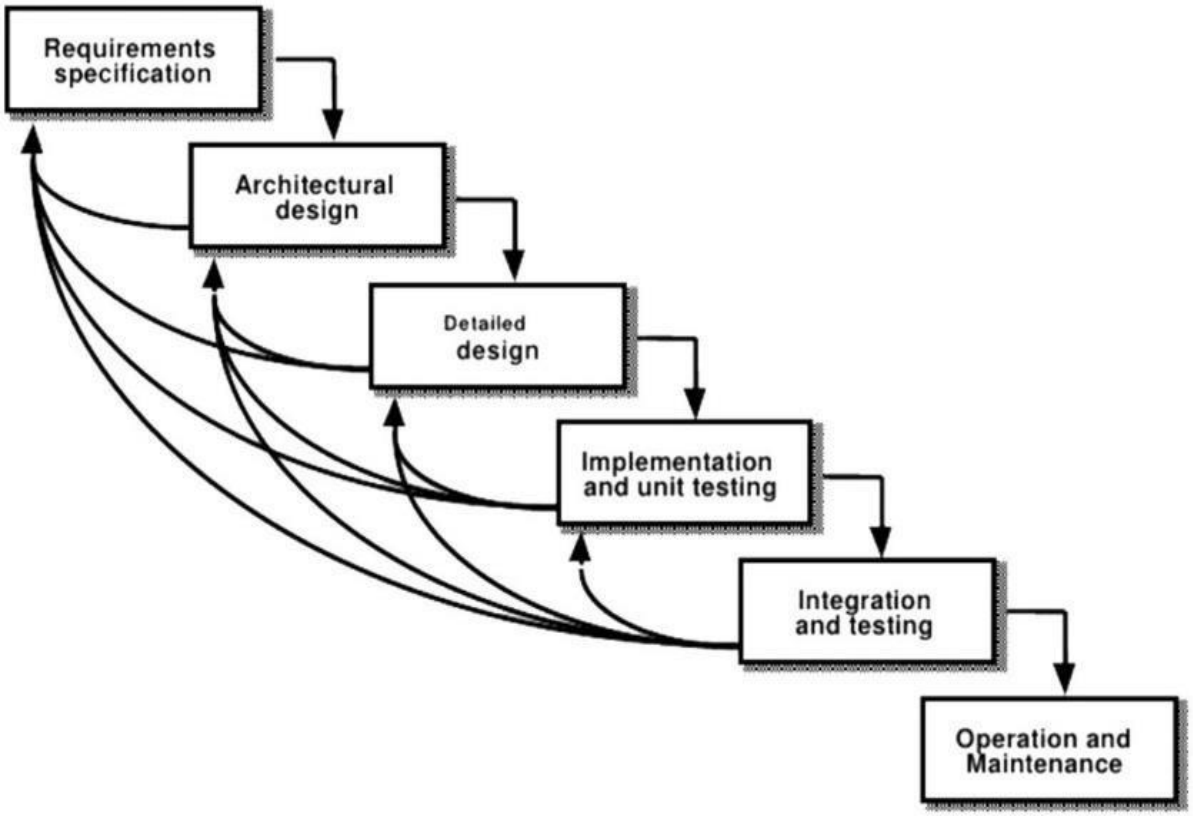


Fig 1: Iterative Enhancement Model

3.8. Data Flow Diagrams (DFDs)

3.8.1. Level 0 DFD

Fig 3: Level 0 DFD

3.8.2. Level 1 DFD

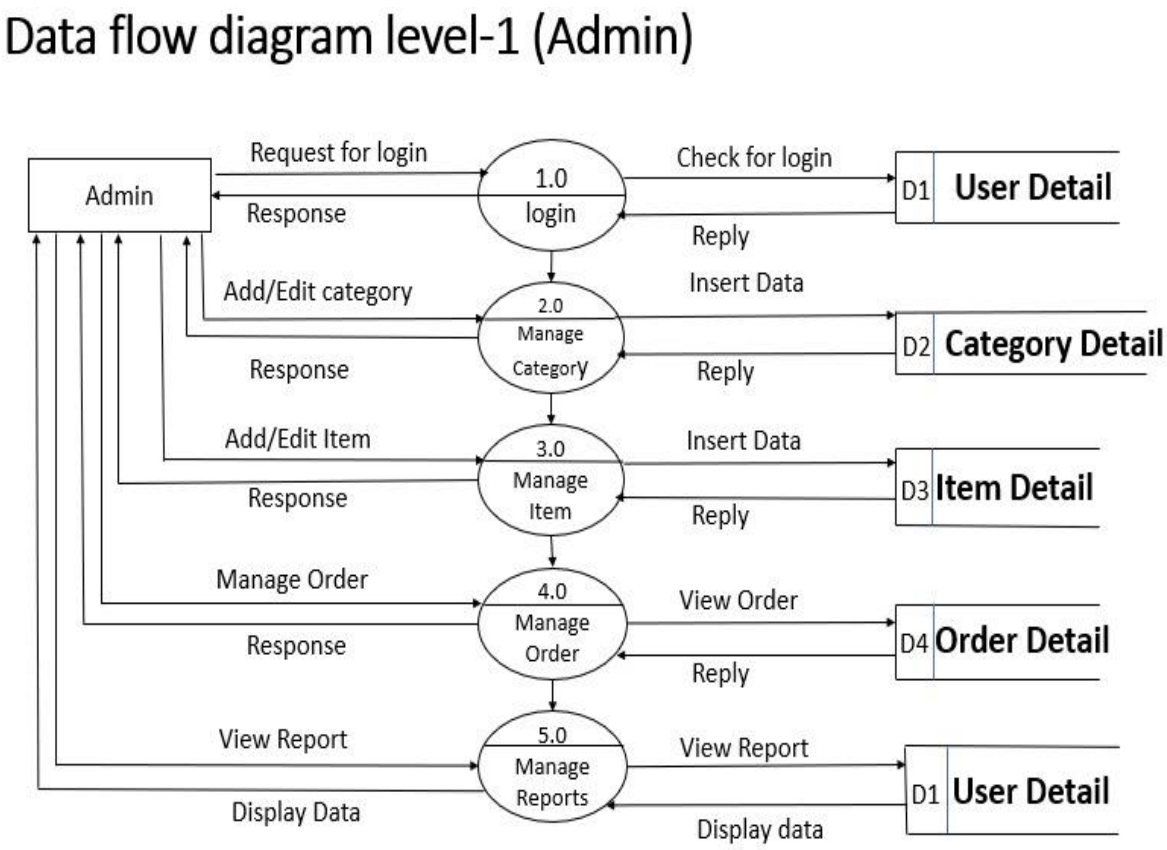


Fig 3.1: Level 1 Admin DFD

3.8.3. Level 2 DFD

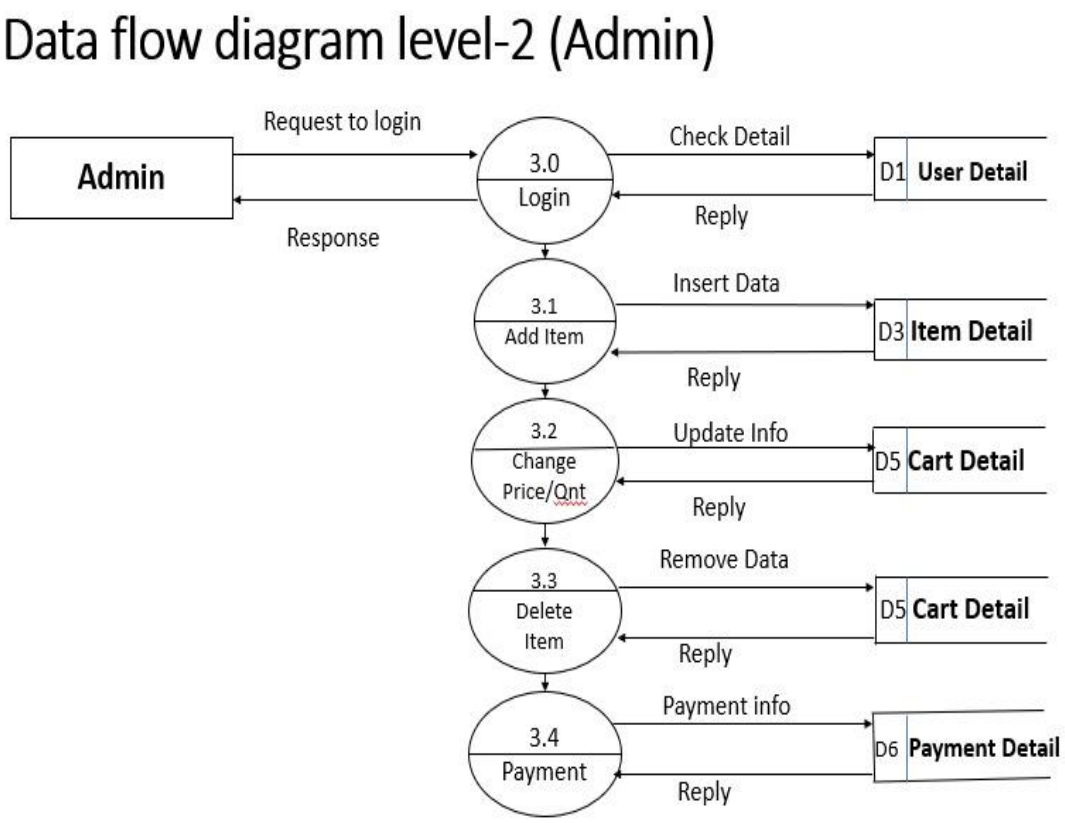


Fig 3.2: Level 1 Admin DFD

CHAPTER 4
SYSTEM DESIGN

It is a process of planning a new business system or replacing an existing system by defining its components or modules to satisfy the specific requirements. Before planning, you need to understand the old system thoroughly and determine how computers can best be used in order to operate efficiently.

System Design focuses on how to accomplish the objective of the system. It mainly focuses on:

- Systems
- Processes
- Technology

3.9. Basic Modules:

In this hotel management system, there are two types of users one is **Admin, Owner of hotel.**

Functionality of Admin/ Owner of Hotel:

Admin can manage all the system by login with his username and password.

Admin can register or add detail of hotel and create or add detail of number of rooms & meal of particular hotel.
After crating a hotel, allocate the house to room and generate username and password for all the member of this system. The member is an owner of hotel.

Admin can responsible for solve complain made by any member of the system. Admin can manage reports of hotel bill, and complain.

3.10. Data Dictionary

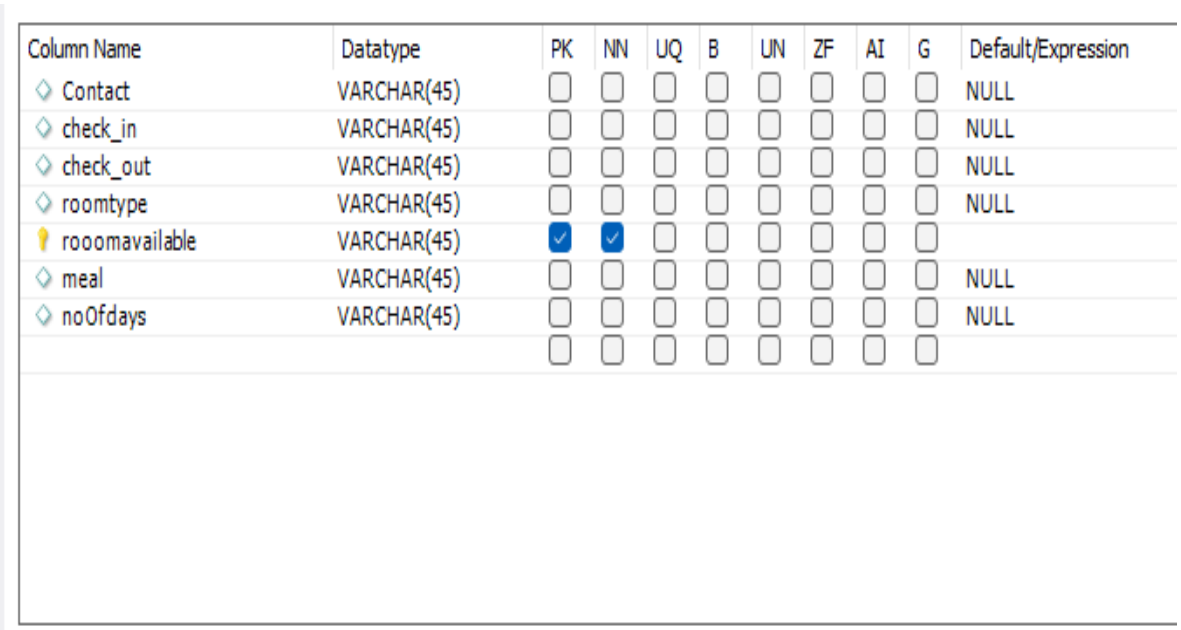
3.10.1. Table: Register/MySQL

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
fname	VARCHAR(45)		✓							
lname	VARCHAR(45)									NULL
contact	VARCHAR(45)									NULL
email	VARCHAR(45)	✓	✓							
securityQ	VARCHAR(45)									NULL
securityA	VARCHAR(45)									NULL
password	VARCHAR(45)									NULL

3.10.2. Table: Customer Details/MySQL

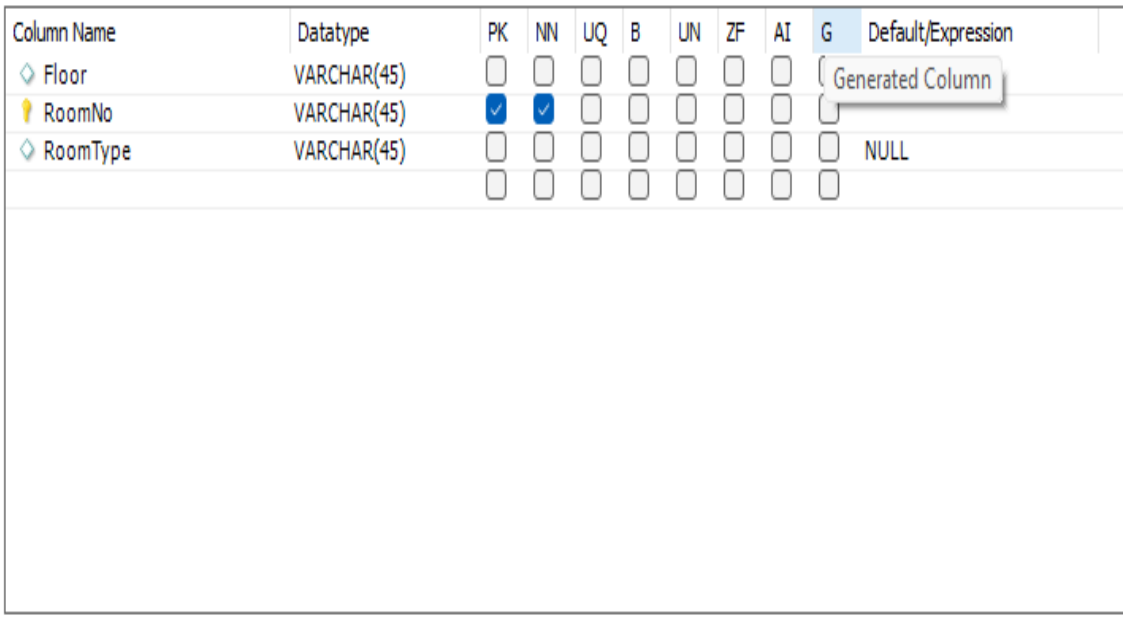
Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
Ref	INT	✓	✓							
Name	VARCHAR(45)									NULL
Mother	VARCHAR(45)									NULL
Gender	VARCHAR(45)									NULL
PostCode	VARCHAR(45)									NULL
Mobile	VARCHAR(45)									NULL
Email	VARCHAR(45)									NULL
Nationality	VARCHAR(45)									NULL
Idproof	VARCHAR(45)									NULL
Idnumber	VARCHAR(45)									NULL
Address	VARCHAR(45)									NULL

3.10.3. Table: Room/MySQL



Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
Contact	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
check_in	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
check_out	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
roomtype	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
rooomavailable	VARCHAR(45)	☑	☑	☐	☐	☐	☐	☐	☐	
meal	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
noOfdays	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
		☐	☐	☐	☐	☐	☐	☐	☐	

3.10.4. Table: Details/ MySQL



Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
Floor	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	
RoomNo	VARCHAR(45)	☑	☑	☐	☐	☐	☐	☐	☐	
RoomType	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
		☐	☐	☐	☐	☐	☐	☐	☐	

Generated Column

3.10.5. Table: Meal/MySQL

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
NoofMeal	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
MealinPlan	VARCHAR(45)	☑	☑	☐	☐	☐	☐	☐	☐	
PlanName	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
		☐	☐	☐	☐	☐	☐	☐	☐	

4.3.Flow Charts

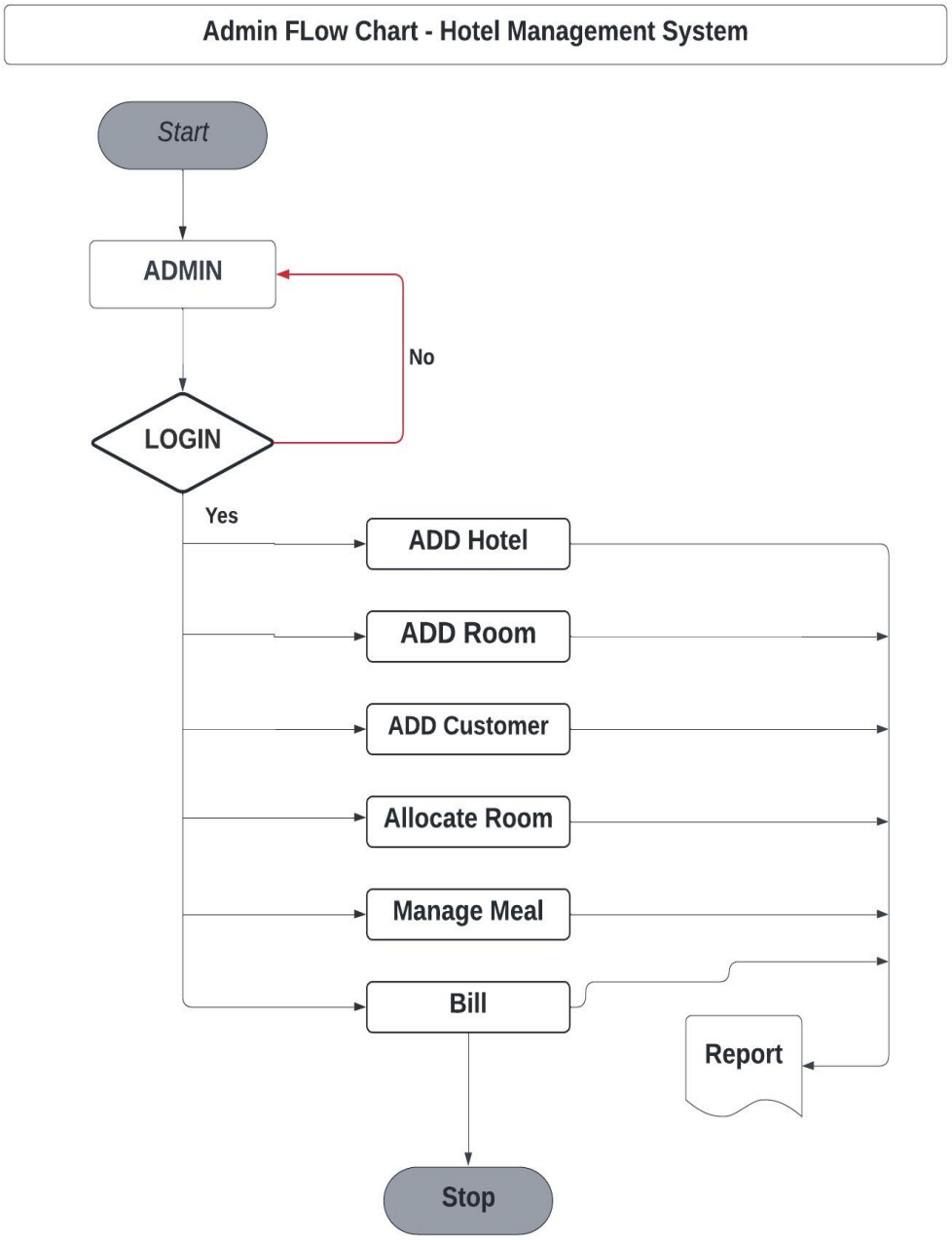


Fig 2: Admin Side Flow Chart

4.4.Gantt Chart

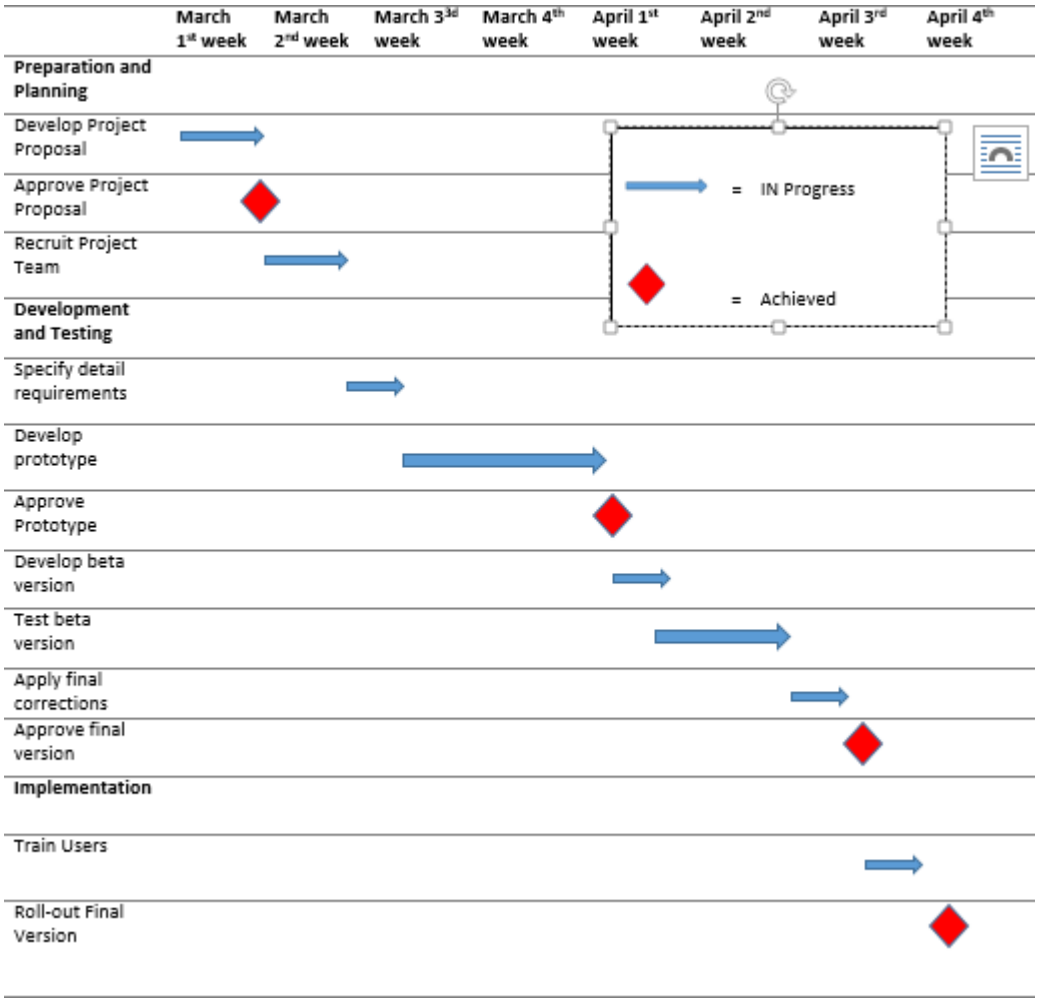


Fig 3: Gantt Chart

4.5. ER-Diagram

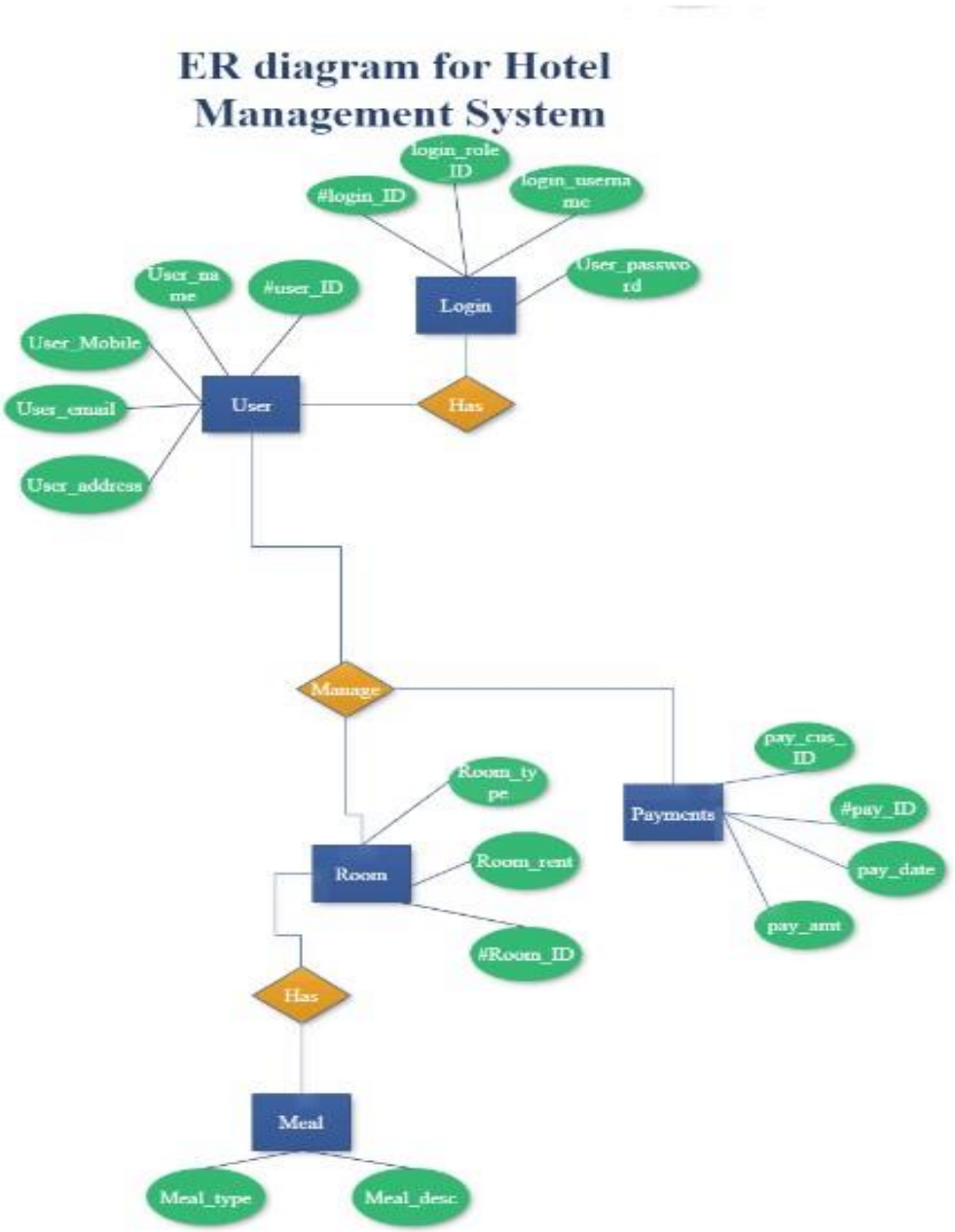


Fig 4.3: ER DIAGRAM

NORMALIZATION

Normalization is the process of organizing data into tables in such a way that the results of using the database are always unambiguous and as intended. Normalization may have the effect of duplicating data within the database and often results in the creation of additional tables. (While normalization tends to increase the duplication of data, it does not introduce redundancy, which is unnecessary duplication.) Normalization is typically a refinement process after the initial exercise of identifying the data objects that should be in the database, identifying their relationships, and defining the tables required and the columns within each table.

1. **Mapping 'Customer' entity**

   Customer

Customer ID	Name	MName	Gender	Country	City	Mobile

2. **Mapping 'Room' entity**

   Room

Room No	Room Type	Price	Max guest	Floor No

3. **Mapping 'Reservation' entity**

   Reservation

Check- in date	No of guests	Days	Check-Out date

4. **Mapping ‘reserve’ Association**

Reserve

Customer_ID	Check- in date	Check-Out date	Days

5. **Mapping ‘Book’ Association**

Book

Customer_ID	E_ID	Book Date

6. **Mapping ‘Add’ Association**

Add

A_ID	Room No

7. **Mapping ‘Meal’ Association**

Meal

A_ID	Meal

Un-Normalized Form

1. **Customer**

Customer_ID	Name	MName	Gender	Country	City	Mobile
2345	Ayele	Kebede	Male	India	Delhi	0912376330

2. **Room**

Room No	Room Type	Price	Max guest	Floor No

R-001	Single	150	72	1

3. Reservation

Check- in date	No of guests	Days	Check- Out date
5/5/05	14	2	9/5/05

4. Reserve

Customer-ID	Check- in date	Check- Out date	Duration	No of Guests
2345	04/05/05	06/05/05	2	1

5. Book

Customer-ID	E_ID	Book Date
2345	12x	07/05/05

6. Add

A_ID	Room No
A_ID_12x	13 12

7. Mapping 'Meal' Association

Meal

A_ID	Meal

1st Normal Form

1. Customer

Customer_ID	Name	MName	Gender	Country	City	Mobile
2345	Ayele	Kebede	Male	India	Delhi	0912376330

2. Room

Room No	Room Type	Price	Max guest	Floor No
R-001	Single	150	72	1

3. Reservation

Check- in date	No of guests	Days	Check- Out date
5/5/05	14	2	9/5/05

4. Reserve

Customer-ID	Check- in date	Check- Out date	Duration	No of Guests
2345	04/05/05	06/05/05	2	1

5. Book

Customer-ID	E_ID	Book Date
2346	12x	07/05/05

6. Add

A_ID	Room No
A_ID_12x	13 12

7. Meal

A-ID	Meal

2nd Normal Form

1. Customer

Customer_ID	Name	MName	Gender	Country	City	Mobile
2345	Ayele	Kebede	Male	India	Delhi	0912376330

2. Room

Room No	Room Type	Price	Max guest	Floor No
R-001	Single	150	72	1

3. Reservation

Check- in date	No of guests	Days	Check- Out date
5/5/05	14	2	9/5/05

4. Reserve

Customer-ID	Check- in date	Check- Out date	Duration	No of Guests
2345	04/05/05	06/05/05	2	1

5. Book

Customer-ID	A_ID	Book Date
2346	12x	07/05/05

6. Add

A-ID	Room No
A_ID_12x	13 12

7. Meal

A_ID	Meal

CHAPTER 5
IMPLEMENTATION AND TESTING

Testing is vital for the success of any software; no system design is ever perfect. Testing is also carried in two phases, first phase is during the software engineering that is during the module creation, second phase is after the completion of software, and this is system testing which verifies that the whole set of programs hanged together.

Coding Details and Code Efficiency

Front-End

Login Window

```
from tkinter import*
from tkinter import ttk
from PIL import  Image,ImageTk #pip install pillow]
from tkinter import messagebox
import mysql.connector
from cProfile import label
from webbrowser import get
import random
from time import strftime
```

```
from datetime import datetime
from cgitb import text
from logging import root
from optparse import Values

def main():
win=Tk()




app=Login_Window(win)
win.mainloop()
class Login_Window:
def __init__(self,root):
self.root=root
self.root.title("Login")
self.root.geometry("1265x635+0+0")

self.bg=ImageTk.PhotoImage(file=r"C:\Users\kumar\OneDrive\Desktop\my_hotel_ma
nagment_system\images\green.jpg")
lbl_bg=Label(self.root,image=self.bg)
lbl_bg.place(x=0,y=0,relwidth=1,relheight=1)

frame=Frame(self.root,bg="black")
frame.place(x=500,y=130,width=320,height=400)

img1=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\Avatar.png")
img1=img1.resize((80,80),Image.ANTIALIAS)
self.photoimg1=ImageTk.PhotoImage(img1)

lblimg1=Label(self.root,image=self.photoimg1,bg="black",borderwidth=0)
lblimg1.place(x=605,y=130,width=120,height=100)
```

```
get_str=Label(frame,text="Get Started",font=("times
of roman",15,"bold"),fg="white",bg="black")
get_str.place(x=105,y=93)
# label
username=lbl=Label(frame,text="Username",font=("times
new roman",13,"bold"),fg="white",bg="black")
username.place(x=60,y=130)




self.txtuser=ttk.Entry(frame,font=("times new roman",10,"bold"))
self.txtuser.place(x=30,y=160,width=270)
password=lbl=Label(frame,text="Password",font=("times
new roman",13,"bold"),fg="white",bg="black")
password.place(x=60,y=190)
self.txtpass=ttk.Entry(frame,font=("times new roman",10,"bold"))
self.txtpass.place(x=30,y=220,width=270)

# ================== Icon Image ======================
img2=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\Avatar.p
ng")
img2=img2.resize((25,25),Image.ANTIALIAS)
self.photoimg2=ImageTk.PhotoImage(img2)
lblimg1=Label(image=self.photoimg2,bg="black",borderwidth=0)
lblimg1.place(x=530,y=260,width=25,height=25)

img3=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\padlock-icon.png")

img3=img3.resize((25,25),Image.ANTIALIAS)
self.photoimg3=ImageTk.PhotoImage(img3)
```

```
lblimg1=Label(image=self.photoimg3,bg="black",borderwidth=50)
lblimg1.place(x=530,y=320,width=25,height=25)

# ==================Login Button=====================
btn_login=Button(frame,text="Login",borderwidth=3,relief=RAISED,command=self.l
ogin,font=("timesnewroman",15,"bold"),bd=3,fg="white",bg="red",activeforeground="
white",activebackground="red")
btn_login.place(x=100,y=250,width=120,height=33)
# Register Button
registerbtn=Button(frame,text="New
UserRegister",command=self.register_window,font=("timesnew



roman",10,"bold"),borderwidth=0,fg="white",bg="black",activeforeground="white",ac
tivebackground="black")
registerbtn.place(x=5,y=310,width=160)

# forgotpassbtn
registerbtn=Button(frame,text="Forget
Password",command=self.forgot_password_window,font=("times
new
roman",10,"bold"),borderwidth=0,fg="white",bg="black",activeforeground="white",ac
tivebackground="black")
registerbtn.place(x=0,y=330,width=160)
def register_window(self):
self.new_window=Toplevel(self.root)
self.app=Register(self.new_window)
def login(self):
if self.txtuser.get()=="" or self.txtpass.get()=="":
messagebox.showerror("Error","all field required",parent=self.root)
elif self.txtuser.get()=="kapu" and self.txtpass.get()=="ashu":
messagebox.showinfo("Success","Welcome to Hotel Managment System")
else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
```

```
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("select * from register where email=%s and password=%s",(
self.txtuser.get(),
self.txtpass.get()
))
row=my_cursor.fetchone()

#print(row)
if row==None:
messagebox.showerror("Error","Invalid Username & password")
else:



open_main=messagebox.askyesno("YesNo","Access only admin")
if open_main>0:
self.new_window=Toplevel(self.root)



self.app=HotelManagementSystem(self.new_window)
else:
if not open_main:
return
conn.commit()
conn.close()
#-------------------------------------reset password====================
def reset_pass(self):
if self.combo_security_Q.get()=="Select":
messagebox.showerror("Error","Select the security question",parent=self.root2)
elif self.txt_security.get()=="":
messagebox.showerror("Error","Please enter the answer",parent=self.root2)
elif self.txt_newpass.get()=="":
messagebox.showerror("Error","Please enter the new password",parent=self.root2)
else:
```

```
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
qury=("select * from register where email=%s and securityQ=%s and securityA=%s")
value=(self.txtuser.get(),self.combo_security_Q.get(),self.txt_security.get(),)
my_cursor.execute(qury,value)
row=my_cursor.fetchone()
if row==None:
messagebox.showerror("Error","Please enter corret Answer",parent=self.root2)
else:
query=("update register set password=%s where email=%s")
value=(self.txt_newpass.get(),self.txtuser.get())
my_cursor.execute(query,value)



conn.commit()
conn.close()
messagebox.showinfo("Info","Your password has been reset ,please login new
password",parent=self.root2)
self.root2.destroy()



#==================forget password window ====================
def forgot_password_window(self):
if self.txtuser.get()=="":
messagebox.showerror("Error","Please enter the email address to reset password")
else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
query=("select * from register where email=%s")
value=(self.txtuser.get(),)
```

```
my_cursor.execute(query,value)
row=my_cursor.fetchone()
if row==None:
messagebox.showerror("My Error","Please enter the valid user name")
else:
conn.close()
self.root2=Toplevel()
self.root2.title("Forget Password")
self.root2.geometry("330x380+490+160")
l=Label(self.root2,text="Forget Password",font=("times
new roman",18,"bold"),fg="red",bg="white")
l.place(x=0,y=10,relwidth=1)
security_Q=Label(self.root2,text="Select Security Question",font=("times
new roman",12,"bold"),bg="white",fg="black")



security_Q.place(x=50,y=70)
self.combo_security_Q=ttk.Combobox(self.root2,font=("times
new roman",12,"bold"),state="readonly")
self.combo_security_Q["values"]=("Select","Your Birth Place","Your
Nick Name","Your Pet Name")
self.combo_security_Q.place(x=50,y=100,width=230)
self.combo_security_Q.current(0)
security_A=Label(self.root2,text="Security Answer",font=("times
new roman",12,"bold"),bg="white",fg="black")

security_A.place(x=50,y=130)
self.txt_security=ttk.Entry(self.root2,font=("times new roman",12))
self.txt_security.place(x=50,y=155,width=230)
new_password=Label(self.root2,text="New Password",font=("times
new roman",12,"bold"),bg="white",fg="black")
new_password.place(x=50,y=185)
self.txt_newpass=ttk.Entry(self.root2,font=("times new roman",12))
self.txt_newpass.place(x=50,y=210,width=230)
```

```
btn=Button(self.root2,text="Reset",command=self.reset_pass,font=("times
new roman",14,"bold"),bg="green",fg="black")
btn.place(x=85,y=250,width=150)
```

Register

```
class Register:
def __init__(self,root):
self.root=root
self.root.title("Register")
self.root.geometry("1265x635+
# ==============variables====================
self.var_fname=StringVar()
self.var_lname=StringVar()




self.var_contact=StringVar()
self.var_email=StringVar()
self.var_securityQ=StringVar()
self.var_securityA=StringVar()
self.var_pass=StringVar()
self.var_confpass=StringVar()
# ================bg image=================
self.bg=ImageTk.PhotoImage(file=r"C:\Users\kumar\OneDrive\Desktop\my_hotel_ma
nagment_system\images\wallpaperflare.com_wallpaper.jpg")
bg_lbl=Label(self.root,image=self.bg)
bg_lbl.place(x=0,y=0,relwidth=1,relheight=1)
# ================left image=================
self.bg1=ImageTk.PhotoImage(file=r"C:\Users\kumar\OneDrive\Desktop\my_hotel_m
anagment_system\images\str1.jpg")
left_lbl=Label(self.root,image=self.bg1)
left_lbl.place(x=50,y=150,width=450,height=400)
# ================= MAin Frame================
```

```
frame=Frame(self.root,bg="white")
frame.place(x=501,y=150,width=675,height=400)
register_lbl=Label(frame,text="REGISTER HERE",font=("times
new roman",18,"bold"),fg="darkgreen",bg="white")
register_lbl.place(x=16,y=15
# ========================= label and entry================
# -------------------Row1
fname=Label(frame,text="First Name",font=("times
new roman",12,"bold"),bg="white")
fname.place(x=40,y=70)
self.fname_entry=ttk.Entry(frame,textvariable=self.var_fname,font=("times
new roman",12))
self.fname_entry.place(x=40,y=95,width=250)
l_name=Label(frame,text="Last Name",font=("times
new roman",12,"bold"),bg="white")
l_name.place(x=368,y=70)



self.txt_lname=ttk.Entry(frame,textvariable=self.var_lname,font=("times
new roman",12))
self.txt_lname.place(x=370,y=95,width=250)
# -------------------Row 2
contact=Label(frame,text="Contact No",font=("times
new roman",12,"bold"),bg="white")
contact.place(x=40,y=130)
self.txt_contact=ttk.Entry(frame,textvariable=self.var_contact,font=("times
new roman",12))
self.txt_contact.place(x=40,y=155,width=250)
email=Label(frame,text="Email",font=("times
new roman",12,"bold"),bg="white")
email.place(x=368,y=130)
self.txt_contact=ttk.Entry(frame,textvariable=self.var_email,font=("times
        new roman",13))
self.txt_contact.place(x=370,y=155,width=250)
```

```
# ------------- Row3
security_Q=Label(frame,text="Select Security Question",font=("times
new roman",12,"bold"),bg="white",fg="black")
security_Q.place(x=40,y=190)
self.combo_security_Q=ttk.Combobox(frame,textvariable=self.var_securityQ,
font=("times new roman",12,"bold"),state="readonly")
self.combo_security_Q["values"]=("Select","Your Birth Place","Your
Nick Name","Your Pet Name")
self.combo_security_Q.place(x=40,y=215,width=250)
self.combo_security_Q.current(0)
security_A=Label(frame,text="Security
Answer",font=("times new roman",12,"bold"),bg="white",fg="black")
security_A.place(x=368,y=190)
self.txt_security=ttk.Entry(frame,textvariable=self.var_securityA,font=("times new
roman",12))
self.txt_security.place(x=368,y=215,width=250)
# ----------------------- ROw 4



pswd=Label(frame,text="Password",font=("times
new roman",12,"bold"),bg="white",fg="black")
pswd.place(x=40,y=251)
self.txt_pswd=ttk.Entry(frame,textvariable=self.var_pass,font=("times
new roman",12))
self.txt_pswd.place(x=40,y=274,width=250)
confirm_pswd=Label(frame,text="Confirm Password",font=("times
new roman",12,"bold"),bg="white",fg="black")
confirm_pswd.place(x=368,y=251)
self.txt_confirm=ttk.Entry(frame,textvariable=self.var_confpass,font=("times
new roman",12))
self.txt_confirm.place(x=368,y=274,width=250)
# ============ Check Out Button ====================
self.var_check=IntVar()
checkbtn=Checkbutton(frame,variable=self.var_check,text="I Agree The
```

```
Terms &
Conditions",font=("times new roman",9,"bold"),onvalue=1,offvalue=0)
checkbtn.place(x=40,y=310)
#======================== Button ====================
img=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system\i
mages\register.jpg")
img=img.resize((200,55),Image.ANTIALIAS)
self.photoimage=ImageTk.PhotoImage(img)
b1=Button(frame,image=self.photoimage,command=self.register_data,borderwidth=0,
cursor="hand2",font=("times new roman",12,"bold"),bg="white")
b1.place(x=20,y=335,width=250)
img1=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\login now.png")
img1=img1.resize((200,55),Image.ANTIALIAS)
self.photoimage1=ImageTk.PhotoImage(img1)
b1=Button(frame,image=self.photoimage1,command=self.return_login,borderwidth=0,
cursor="hand2",font=("times new roman",12,"bold"),bg="white")
b1.place(x=380,y=335,width=250)



# ================== Function Declaration===============
def register_data(self):
if self.var_fname.get()=="" or self.var_email.get()==""
or self.var_securityQ.get()=="Select":
messagebox.showerror("Eroor","All fields are requaired",parent=self.root)
elif self.var_pass.get()!=self.var_confpass.get():
messagebox.showerror("Error","password & confirm password must be same")
elif self.var_check.get()==0:
messagebox.showerror("Error","Please agree our terms and condition")
else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
query=("select * from register where email=%s")
```

```
value=(self.var_email.get(),)
my_cursor.execute(query,value)
row=my_cursor.fetchone()
if row!=None:
messagebox.showerror("Error","User already exist,please try another email")
else:
my_cursor.execute("insert into register values(%s,%s,%s,%s,%s,%s,%s)",(
self.var_fname.get(),
                                                    self.var_lname.get(),
                                                    self.var_contact.get(),
                                                    self.var_email.get(),
                                                    self.var_securityQ.get(),
                                                    self.var_securityA.get(),
                                                    self.var_pass.get()

                                            ))
        conn.commit()
        conn.close()
messagebox.showinfo("Success","Register Successfully")



def return_login(self):
class Register:
def __init__(self,root):
self.root=root
self.root.title("Register")
self.root.geometry("1265x635+0+0")
# ==============variables====================
self.var_fname=StringVar()
self.var_lname=StringVar()
self.var_contact=StringVar()
self.var_email=StringVar()
self.var_securityQ=StringVar()
self.var_securityA=StringVar()
```

```
self.var_pass=StringVar()
self.var_confpass=StringV
# ================bg image==================
self.bg=ImageTk.PhotoImage(file=r"C:\Users\kumar\OneDrive\Desktop\my_hotel_ma
nagment_system\images\wallpaperflare.com_wallpaper.jpg")
bg_lbl=Label(self.root,image=self.bg)
bg_lbl.place(x=0,y=0,relwidth=1,relheight=1)
# ================left image==================
self.bg1=ImageTk.PhotoImage(file=r"C:\Users\kumar\OneDrive\Desktop\my_hotel_m
anagment_system\images\str1.jpg")
left_lbl=Label(self.root,image=self.bg1)
left_lbl.place(x=50,y=150,width=450,height=400)
# ================== MAin Frame================
frame=Frame(self.root,bg="white")
frame.place(x=501,y=150,width=675,height=400)
register_lbl=Label(frame,text="REGISTER HERE",font=("times
new roman",18,"bold"),fg="darkgreen",bg="white")
register_lbl.place(x=16,y=15
# ========================== label and entry================
# --------------------Row1



fname=Label(frame,text="First Name",font=("times
new roman",12,"bold"),bg="white")
fname.place(x=40,y=70)
self.fname_entry=ttk.Entry(frame,textvariable=self.var_fname,font=("times
new roman",12))
self.fname_entry.place(x=40,y=95,width=250)
l_name=Label(frame,text="Last Name",font=("times
new roman",12,"bold"),bg="white")
l_name.place(x=368,y=70)
self.txt_lname=ttk.Entry(frame,textvariable=self.var_lname,font=("times
new roman",12))
self.txt_lname.place(x=370,y=95,width=250)
```

```
# -------------------Row 2
contact=Label(frame,text="Contact No",font=("times
new roman",12,"bold"),bg="white")
contact.place(x=40,y=130)
self.txt_contact=ttk.Entry(frame,textvariable=self.var_contact,font=("times
new roman",12))
self.txt_contact.place(x=40,y=155,width=250)

email=Label(frame,text="Email",font=("times
new roman",12,"bold"),bg="white")
email.place(x=368,y=130)
self.txt_contact=ttk.Entry(frame,textvariable=self.var_email,font=("times
new roman",13))
self.txt_contact.place(x=370,y=155,width=250)
# ------------- Row3
security_Q=Label(frame,text="Select Security Question",font=("times
new roman",12,"bold"),bg="white",fg="black")
security_Q.place(x=40,y=190)
self.combo_security_Q=ttk.Combobox(frame,textvariable=self.var_securityQ,
font=("times new roman",12,"bold"),state="readonly")
self.combo_security_Q["values"]=("Select","Your Birth Place","Your



Nick Name","Your Pet Name")
self.combo_security_Q.place(x=40,y=215,width=250)
self.combo_security_Q.current(0)
security_A=Label(frame,text="Security
Answer",font=("times new roman",12,"bold"),bg="white",fg="black")
security_A.place(x=368,y=190)
self.txt_security=ttk.Entry(frame,textvariable=self.var_securityA,font=("times new
roman",12))
self.txt_security.place(x=368,y=215,width=250)
# ----------------------- ROw 4
pswd=Label(frame,text="Password",font=("times
```

```
new roman",12,"bold"),bg="white",fg="black")
pswd.place(x=40,y=251)
self.txt_pswd=ttk.Entry(frame,textvariable=self.var_pass,font=("times
new roman",12))
self.txt_pswd.place(x=40,y=274,width=250)
confirm_pswd=Label(frame,text="Confirm Password",font=("times
new roman",12,"bold"),bg="white",fg="black")
confirm_pswd.place(x=368,y=251)
self.txt_confirm=ttk.Entry(frame,textvariable=self.var_confpass,font=("times
new roman",12))
self.txt_confirm.place(x=368,y=274,width=250)
# ============= Check Out Button ======================
self.var_check=IntVar()
checkbtn=Checkbutton(frame,variable=self.var_check,text="I Agree
The Terms &
Conditions",font=("times new roman",9,"bold"),onvalue=1,offvalue=0)
checkbtn.place(x=40,y=310)
#======================== Button ====================
img=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system\i
mages\register.jpg")
img=img.resize((200,55),Image.ANTIALIAS)
self.photoimage=ImageTk.PhotoImage(img)



b1=Button(frame,image=self.photoimage,command=self.register_data,borderwidth=0,
cursor="hand2",font=("times new roman",12,"bold"),bg="white")
b1.place(x=20,y=335,width=250)
img1=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\login now.png")
img1=img1.resize((200,55),Image.ANTIALIAS)
self.photoimage1=ImageTk.PhotoImage(img1)
b1=Button(frame,image=self.photoimage1,command=self.return_login,borderwidth=0,
cursor="hand2",font=("times new roman",12,"bold"),bg="white")
b1.place(x=380,y=335,width=250)
```

```
# ================== Function Declaration===============
  def register_data(self):
  if self.var_fname.get()=="" or self.var_email.get()==""
  or self.var_securityQ.get()=="Select":
  messagebox.showerror("Eroor","All fields are requaired",parent=self.root)
  elif self.var_pass.get()!=self.var_confpass.get():
  messagebox.showerror("Error","password & confirm password must be same")
  elif self.var_check.get()==0:
  messagebox.showerror("Error","Please agree our terms and condition")
  else:
  conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
  015$$",database="managment")
  my_cursor=conn.cursor()
  query=("select * from register where email=%s")
  value=(self.var_email.get(),)
  my_cursor.execute(query,value)
  row=my_cursor.fetchone()
  if row!=None:
  messagebox.showerror("Error","User already exist,please try another email")
  else:
  my_cursor.execute("insert into register values(%s,%s,%s,%s,%s,%s,%s)",(
                                                    self.var_fname.get(),
                                                    self.var_lname.get(),



  self.var_contact.get(),
  self.var_email.get(),
  self.var_securityQ.get(),
  self.var_securityA.get(),
    self.var_pass.get()
     ))
  conn.commit()
  conn.close()
  messagebox.showinfo("Success","Register Successfully")
```

```
def return_login(self):
self.root.destroy()
```

(Home) Hotel Management System

```
class HotelManagementSystem:
def __init__(self,root):
self.root=root
self.root.title("Hotel Managment System")
self.root.geometry("1265x635+0+0")
# ====================1st img=============================
img1=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\hotel1.jpg")
img1=img1.resize((1265,150),Image.ANTIALIAS)
self.photoimg1=ImageTk.PhotoImage(img1)

lblimg=Label(self.root,image=self.photoimg1,bd=4,relief=RIDGE)
lblimg.place(x=0,y=0,width=1260,height=140)
# ========================logo=========================
img2=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\hotellogo.jpg")
img2=img2.resize((230,150),Image.ANTIALIAS)
self.photoimg2=ImageTk.PhotoImage(img2)
lblimg=Label(self.root,image=self.photoimg2,bd=4,relief=RIDGE)
lblimg.place(x=0,y=0,width=230,height=140)



# ====================title=================================
lbl_title=Label(self.root,text="HOTEL MANAGMENT SYSTEM",font=("times
new roman",30,"bold"),bg="black",fg="gold",bd=4,relief=RIDGE)
lbl_title.place(x=0,y=120,width=1260,height=50)
#======================main frame=======================
main_frame=Frame(self.root,bd=4,relief=RIDGE)
main_frame.place(x=0,y=160,width=1260,height=470)
#=======================menu====================
```

```
lbl_menu=Label(main_frame,text="MENU",font=("times
new roman",18,"bold"),bg="black",fg="gold",bd=4,relief=RIDGE)
lbl_menu.place(x=0,y=0,width=230)
# ======================Btn frame================
btn_frame=Frame(main_frame,bd=4,relief=RIDGE)
btn_frame.place(x=0,y=30,width=225,height=160)
cust_btn=Button(btn_frame,text="CUSTOMER",command=self.cust_details,width=27
,font=("times
new roman",12,"bold"),bg="black",fg="gold",bd=0,cursor="hand1")
cust_btn.grid(row=0,column=0,pady=1)
room_btn=Button(btn_frame,text="ROOM",command=self.roombooking,width=27,fo
nt=("times
new roman",12,"bold"),bg="black",fg="gold",bd=0,cursor="hand1")
room_btn.grid(row=1,column=0,pady=1)
details_btn=Button(btn_frame,text="DETAILS",command=self.details_room,width=2
7,font=("times
new roman",12,"bold"),bg="black",fg="gold",bd=0,cursor="hand1")
details_btn.grid(row=2,column=0,pady=1)
report_btn=Button(btn_frame,text="REPORT",width=27,font=("times
new roman",12,"bold"),bg="black",fg="gold",bd=0,cursor="hand1")
report_btn.grid(row=3,column=0,pady=1)
logout_btn=Button(btn_frame,text="LOGOUT",command=self.logout,width=27,font=
("times new roman",12,"bold"),bg="black",fg="gold",bd=0,cursor="hand1")
logout_btn.grid(row=4,column=0,pady=1)
==========RIGHT SIDE IMAGE============================



img3=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\hotels.jpg")
img3=img3.resize((1310,590),Image.ANTIALIAS)
self.photoimg3=ImageTk.PhotoImage(img3)
lblimg1=Label(main_frame,image=self.photoimg3,bd=4,relief=RIDGE)
lblimg1.place(x=225,y=0,width=1030,height=465)
# ==============DOWN IMAGES============================
```

```
  img4=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
  \images\hotelfront.jpg")
  img4=img4.resize((220,210),Image.ANTIALIAS)
  self.photoimg4=ImageTk.PhotoImage(img4)
  lblimg1=Label(main_frame,image=self.photoimg4,bd=4,relief=RIDGE)
  lblimg1.place(x=0,y=190,width=230,height=150)
  img5=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
  \images\hotelfood.jpg")
  img5=img5.resize((230,190),Image.ANTIALIAS)
  self.photoimg5=ImageTk.PhotoImage(img5)
  lblimg1=Label(main_frame,image=self.photoimg5,bd=4,relief=RIDGE)
  lblimg1.place(x=0,y=320,width=230,height=145)
  def cust_details(self):
  self.new_window=Toplevel(self.root)
  self.app=Cust_Win(self.new_window)
  def roombooking(self):
  self.new_window=Toplevel(self.root)
  self.app=Roombooking(self.new_window)
def details_room(self):
self.new_window=Toplevel(self.root)
self.app=DeatailsRoom(self.new_window)
def logout(self):
self.root.destroy()oom Booking
  class Roombooking:
  def __init__(self,root):
  self.root=root



  self.root.title("Hotel Managment System")
  self.root.geometry("1025x424+233+200")
  # =============== variable==============================
  self.var_contact=StringVar()
  self.var_checkin=StringVar()
  self.var_checkout=StringVar()
```

```
self.var_roomtype=StringVar()
self.var_rooomavailable=StringVar()
self.var_meal=StringVar()
self.noOfdays=StringVar()
self.var_paidtax=StringVar()
self.var_actualtotal=StringVar()
self.var_total=StringVar()
# ======================title==========================
lbl_title=Label(self.root,text="ROOM BOOKING DETAILS",font=("times new
roman",15,"bold"),bg="black",fg="gold",bd=4,relief=RIDGE)
lbl_title.place(x=0,y=0,width=1295,height=35)
# ========================logo==========================
img2=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\hotellogo.jpg")
img2=img2.resize((100,40),Image.ANTIALIAS)
self.photoimg2=ImageTk.PhotoImage(img2)
lblimg=Label(self.root,image=self.photoimg2,bd=0,relief=RIDGE)
lblimg.place(x=5,y=2,width=80,height=30)
# ====================LABLE FRAME=======================
labelframeleft=LabelFrame(self.root,bd=2,relief=RIDGE,text="Roombooking
Dtails",font=("times new roman",11,"bold"),padx=2)
labelframeleft.place(x=5,y=35,width=340,height=400)
#  ====================Lable & Entry=====================
#  Customer contact
lbl_cust_contact=Label(labelframeleft,text="Customer
Contact",font=("arial",9,"bold"),padx=2,pady=6)
lbl_cust_contact.grid(row=0,column=0,sticky=W)



enty_contact=ttk.Entry(labelframeleft,textvariable=self.var_contact,font=("arial",10,"b
old"),width=21)
enty_contact.grid(row=0,column=1,sticky=W)
# Fetch Data Button
btnFetchData=Button(labelframeleft,command=self.Fetch_contact,text="Fetch
```

```
Data",font=("arial",8,"bold"),bg="black",fg="gold",width=8)
btnFetchData.place(x=266,y=4)
#  Check_in Data
check_in_date=Label(labelframeleft,font=("arial",9,"bold"),text="Check_in
Date:",padx=2,pady=6)
check_in_date.grid(row=1,column=0,sticky=W)
textcheck_in_date=ttk.Entry(labelframeleft,textvariable=self.var_checkin,font=("arial",
10,"bold"),width=29)
textcheck_in_date.grid(row=1,column=1)
#  Check_out Data
lbl_Check_out=Label(labelframeleft,font=("arial",9,"bold"),text="Check_Out
Date:",padx=2,pady=6)
lbl_Check_out.grid(row=2,column=0,sticky=W)
text_Check_out=ttk.Entry(labelframeleft,textvariable=self.var_checkout,font=("arial",1
0,"bold"),width=29)
text_Check_out.grid(row=2,column=1)
#  Room Type
label_RoomType=Label(labelframeleft,font=("arial",9,"bold"),text="Room
Type:",padx=2,pady=6)
label_RoomType.grid(row=3,column=0,sticky=W)
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("select RoomType from details")
ide=my_cursor.fetchall()
combo_RoomType=ttk.Combobox(labelframeleft,textvariable=self.var_roomtype,font
=("arial",10,"bold"),width=27,state="readonly")
combo_RoomType["value"]=ide



combo_RoomType.current(0)
combo_RoomType.grid(row=3,column=1)
#  Available Room
lblRoomAvailable=Label(labelframeleft,font=("arial",9,"bold"),text="Available
```

```
Room:",padx=2,pady=6)
lblRoomAvailable.grid(row=4,column=0,sticky=W)
#textRoomAvailable=ttk.Entry(labelframeleft,textvariable=self.var_rooomavailable,fo
nt=("arial",13,"bold"),width=29)
#textRoomAvailable.grid(row=4,column=1)
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("select RoomNo from details")
rows=my_cursor.fetchall()
combo_RoomNo=ttk.Combobox(labelframeleft,textvariable=self.var_rooomavailable,f
ont=("arial",10,"bold"),width=27,state="readonly")
combo_RoomNo["value"]=rows
combo_RoomNo.current(0)
combo_RoomNo.grid(row=4,column=1)

        #  Meal
lblMeal=Label(labelframeleft,font=("arial",9,"bold"),text="Meal:",padx=2,pady=6)
lblMeal.grid(row=5,column=0,sticky=W)
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("select PlanName from meal")
ipl=my_cursor.fetchall()
combo_Meal=ttk.Combobox(labelframeleft,textvariable=self.var_meal,font=("arial",10
,"bold"),width=27,state="readonly")
combo_Meal["value"]=ipl
combo_Meal.current(0)
combo_Meal.grid(row=5,column=1)



#  No Of Days
lblNoOfDays=Label(labelframeleft,font=("arial",9,"bold"),text="No of
Days:",padx=2,pady=6)
```

```
lblNoOfDays.grid(row=6,column=0,sticky=W)
textNoOfDayslblNoOfDays=ttk.Entry(labelframeleft,textvariable=self.noOfdays,font=("arial",10,"bold"),width=29)
textNoOfDayslblNoOfDays.grid(row=6,column=1)

#  Paid Tax
lblNoOfDays=Label(labelframeleft,font=("arial",9,"bold"),text="Paid Tax:",padx=2,pady=6)
lblNoOfDays.grid(row=7,column=0,sticky=W)
textNoOfDayslblNoOfDays=ttk.Entry(labelframeleft,textvariable=self.var_paidtax,font=("arial",10,"bold"),width=29)
textNoOfDayslblNoOfDays.grid(row=7,column=1)

# Sub Total
lblNoOfDays=Label(labelframeleft,font=("arial",9,"bold"),text="Sub Total:",padx=2,pady=6)
lblNoOfDays.grid(row=8,column=0,sticky=W)
textNoOfDayslblNoOfDays=ttk.Entry(labelframeleft,textvariable=self.var_actualtotal,font=("arial",10,"bold"),width=29)
textNoOfDayslblNoOfDays.grid(row=8,column=1)

# Total Cost
lblNoOfDays=Label(labelframeleft,font=("arial",9,"bold"),text="Total Cost:",padx=2,pady=6)
lblNoOfDays.grid(row=9,column=0,sticky=W)
textNoOfDayslblNoOfDays=ttk.Entry(labelframeleft,textvariable=self.var_total,font=("arial",10,"bold"),width=29)
textNoOfDayslblNoOfDays.grid(row=9,column=1)
# ====================== Bill Button ======================
btnBill=Button(labelframeleft,text="Bill",command=self.total,font=("arial",9,"bold"),bg="black",fg="gold",width=10)
btnBill.grid(row=10,column=0,padx=1,sticky=W)
```

```
# ======================Btns==============================
btn_frame=Frame(labelframeleft,bd=2,relief=RIDGE)
btn_frame.place(x=0,y=335,width=333,height=35)
btnAdd=Button(btn_frame,text="Add",command=self.add_data,font=("arial",9,"bold")
,bg="black",fg="gold",width=10)
btnAdd.grid(row=0,column=0,padx=1)
btnUpdate=Button(btn_frame,text="Update",command=self.update,font=("arial",9,"bol
d"),bg="black",fg="gold",width=10)
btnUpdate.grid(row=0,column=1,padx=1)
btnDelete=Button(btn_frame,text="Delete",command=self.mDelete,font=("arial",9,"bo
ld"),bg="black",fg="gold",width=10)
btnDelete.grid(row=0,column=2,padx=1)
btnReset=Button(btn_frame,text="Reset",command=self.reset,font=("arial",9,"bold"),b
g="black",fg="gold",width=10)
btnReset.grid(row=0,column=3,padx=1)
# ======================Right side Image==================
img3=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\bed.jpg")
img3=img3.resize((520,300),Image.ANTIALIAS)
self.photoimg3=ImageTk.PhotoImage(img3)
lblimg=Label(self.root,image=self.photoimg3,bd=0,relief=RIDGE)
lblimg.place(x=670,y=37,width=350,height=210)

# ====================Table frame search system==========
Table_Frame=LabelFrame(self.root,bd=2,relief=RIDGE,text="View Details And
Search System",font=("arial",11,"bold"),padx=2)
Table_Frame.place(x=350,y=215,width=670,height=200)
lblSearchBy=Label(Table_Frame,font=("arial",11,"bold"),text="Search
By:",bg="red",fg="white")
lblSearchBy.grid(row=0,column=0,sticky=W)
self.search_var=StringVar()



combo_Search=ttk.Combobox(Table_Frame,textvariable=self.search_var,font=("arial",
```

```
11,"bold"),width=17,state="readonly")
combo_Search["value"]=("Contact","Room")
combo_Search.current(0)
combo_Search.grid(row=0,column=1,padx=2)
self.txt_search=StringVar()
txtSearch=ttk.Entry(Table_Frame,textvariable=self.txt_search,font=("arial",12,"bold"),
width=17)
txtSearch.grid(row=0,column=2,padx=2)
btnSearch=Button(Table_Frame,text="Search",command=self.search,font=("arial",11,"
bold"),bg="black",fg="gold",width=9)
btnSearch.grid(row=0,column=3,padx=1)
btnShowAll=Button(Table_Frame,text="Show
All",command=self.fetch_data,font=("arial",11,"bold"),bg="black",fg="gold",width=9)
btnShowAll.grid(row=0,column=4,padx=1)
# ======================SHOW DATA====================
details_table=Frame(Table_Frame,bd=2,relief=RIDGE)
details_table.place(x=0,y=50,width=660,height=125)
scroll_x=ttk.Scrollbar(details_table,orient=HORIZONTAL)
scroll_y=ttk.Scrollbar(details_table,orient=VERTICAL)
self.room_table=ttk.Treeview(details_table,column=("contact","checkin","checkout","r
oomtype","rooomavailable","meal","noOfdays"),xscrollcommand=scroll_x.set,yscrollc
ommand=scroll_x.set)
scroll_x.pack(side=BOTTOM,fill=X)
scroll_y.pack(side=RIGHT,fill=Y)
scroll_x.config(command=self.room_table.xview)
scroll_y.config(command=self.room_table.yview)
self.room_table.heading("contact",text="Contact")
self.room_table.heading("checkin",text="Check-in")
self.room_table.heading("checkout",text="Check-out")
self.room_table.heading("roomtype",text="Room Type")
self.room_table.heading("rooomavailable",text="Room No")
self.room_table.heading("meal",text="Meal")
```

```
self.room_table.heading("noOfdays",text="NoOfDays")
self.room_table["show"]="headings"
self.room_table.column("contact",width=100)
self.room_table.column("checkin",width=100)
self.room_table.column("checkout",width=100)
self.room_table.column("roomtype",width=100)
self.room_table.column("rooomavailable",width=100)
self.room_table.column("meal",width=100)
self.room_table.column("noOfdays",width=100)
self.room_table.pack(fill=BOTH,expand=1)
self.room_table.bind("<ButtonRelease-1>",self.get_cuersor)
self.fetch_data()
# ======================add data============================
def add_data(self):
if self.var_contact.get()=="" or self.var_checkin.get()=="":
messagebox.showerror("Eroor","All fields are requaired",parent=self.root)
else:
try:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("insert into room values(%s,%s,%s,%s,%s,%s,%s)",(
self.var_contact.get(),
self.var_checkin.get(),
self.var_checkout.get(),
self.var_roomtype.get(),
self.var_rooomavailable.get(),
self.var_meal.get(),
self.noOfdays.get()
))
conn.commit()
self.fetch_data()
conn.close()
```

```
messagebox.showinfo("Success","Room Booked",parent=self.root)
except Exception as es:
messagebox.showwarning("Warning",f"Some thing went
wrong:{str(es)}",parent=self.root)

# fetch data
def fetch_data(self):
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("select * from room")
rows=my_cursor.fetchall()
if len(rows)!=0:
self.room_table.delete(*self.room_table.get_children())
for i in rows:
self.room_table.insert("",END,values=i)
conn.commit()
conn.close()

#get cursor
def get_cuersor(self,event=""):
cursor_row=self.room_table.focus()
content=self.room_table.item(cursor_row)
row=content["values"]
self.var_contact.set(row[0])
self.var_checkin.set(row[1])
self.var_checkout.set(row[2])
self.var_roomtype.set(row[3])
self.var_rooomavailable.set(row[4])
self.var_meal.set(row[5])
self.noOfdays.set(row[6])
#  update function
def update(self):
```

```
if self.var_contact.get()=="":
messagebox.showerror("Error","Please enter mobile number",parent=self.root)
else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("update room set
check_in=%s,check_out=%s,roomtype=%s,rooomavailable=%s,meal=%s,noOfdays=
%s where Contact=%s",(
self.var_checkin.get()
self.var_checkout.get(),
self.var_roomtype.get(),
self.var_rooomavailable.get(),
self.var_meal.get(),
self.noOfdays.get(),
        self.var_contact.get()
))
conn.commit()
self.fetch_data()
conn.close()
messagebox.showinfo("Update","Room details has been updated
successfully",parent=self.root)
def mDelete(self):
mDelete=messagebox.askyesno("Hotel Managment System","DO you want to delete
this customer details",parent=self.root)
if mDelete>0:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
query="delete from room where Contact=%s"
value=(self.var_contact.get(),)
my_cursor.execute(query,value)
else:
```

```
if not mDelete:
return
conn.commit()
self.fetch_data()
conn.close()
def reset(self):
self.var_contact.set("")
self.var_checkin.set("")
self.var_checkout.set("")
self.var_roomtype.set("")
self.var_rooomavailable.set("")
self.var_meal.set("")
self.noOfdays.set("")
self.var_paidtax.set("")
self.var_actualtotal.set("")
self.var_total.set("")

# ========================All  Data Fetch ==================
def Fetch_contact(self):
if self.var_contact.get()=="":
messagebox.showerror("Error","Please enter Contact Number",parent=self.root)
else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
query=("select Name from customer where Mobile=%s")
value=(self.var_contact.get(),)
my_cursor.execute(query,value)
row=my_cursor.fetchone()
if row==None:
messagebox.showerror("Error","This number Not Found",parent=self.root)
else:
conn.commit()
conn.close()
showDataframe=Frame(self.root,bd=4,relief=RIDGE,padx=2)
showDataframe.place(x=350,y=45,width=310,height=165)
lblName=Label(showDataframe,text="Name:",font=("arial","12","bold"))
lblName.place(x=0,y=0)
lbl=Label(showDataframe,text=row,font=("arial","12","bold"))
lbl.place(x=90,y=0)
# ===========Gender===============
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
```

```
015$$",database="managment")



my_cursor=conn.cursor()
query=("select Gender from customer where Mobile=%s")
value=(self.var_contact.get(),)
my_cursor.execute(query,value)
row=my_cursor.fetchone()
lblGender=Label(showDataframe,text="Gender:",font=("arial","12","bold"))
lblGender.place(x=0,y=30)
lbl2=Label(showDataframe,text=row,font=("arial","12","bold"))
lbl2.place(x=90,y=30)
# ==============email=========================
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
query=("select Email from customer where Mobile=%s")
value=(self.var_contact.get(),)
my_cursor.execute(query,value)
row=my_cursor.fetchone()
lbl3=Label(showDataframe,text=row,font=("arial","12","bold"))
lbl3.place(x=90,y=60)
# ==============Nationality=========================
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
query=("select Nationality from customer where Mobile=%s")
value=(self.var_contact.get(),)
my_cursor.execute(query,value)
row=my_cursor.fetchone()
lblNationality=Label(showDataframe,text="Nationality:",font=("arial","12","bold"))
lblNationality.place(x=0,y=90)
lbl4=Label(showDataframe,text=row,font=("arial","12","bold"))
lbl4.place(x=90,y=90)
# ==============Address=========================
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
```

```
my_cursor=conn.cursor()



query=("select Address from customer where Mobile=%s")
value=(self.var_contact.get(),)
my_cursor.execute(query,value)
row=my_cursor.fetchone()
lbladdress=Label(showDataframe,text="Address:",font=("arial","12","bold"))
lbladdress.place(x=0,y=120)lbl1=Label(showDataframe,text=row,font=("arial","12","b
old"))
lbl1.place(x=90,y=120)
# search System
def search(self):
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("select * from room where "+str(self.search_var.get())+" LIKE
'%"+str(self.txt_search.get())+"%'")
rows=my_cursor.fetchall()
if len (rows)!=0:
self.room_table.delete(*self.room_table.get_children())
for i in rows:
self.room_table.insert("",END,values=i)
conn.commit()
conn.close()
def total(self):
inDate=self.var_checkin.get()
outDate=self.var_checkout.get()
inDate=datetime.strptime(inDate,"%d/%m/%Y")
outDate=datetime.strptime(outDate,"%d/%m/%Y")
self.noOfdays.set(abs(outDate-inDate).days)
if (self.var_meal.get()=="'American Plan"' and
self.var_roomtype.get()=="Luxury","Single","Deluxe"):
q1=float(2000)
```

```
q2=float(1500)
q3=float(self.noOfdays.get())



q4=float(q1+q2)
q5=float(q3*q4)
Tax="Rs."+str("%.2f"%((q5)*0.09))
ST="Rs."+str("%.2f"%((q5)))
TT="Rs."+str("%.2f"%(q5+((q5)*0.09)))
self.var_paidtax.set(Tax)
self.var_actualtotal.set(ST)
self.var_total.set(TT)
elif (self.var_meal.get()=="'Europe Plan'" and
self.var_roomtype.get()=="Single","Luxury","Deluxe"):
q1=float(1800)
q2=float(1300)
q3=float(self.noOfdays.get())
q4=float(q1+q2)
q5=float(q3*q4)
Tax="Rs."+str("%.2f"%((q5)*0.09))
ST="Rs."+str("%.2f"%((q5)))
TT="Rs."+str("%.2f"%(q5+((q5)*0.09)))
self.var_paidtax.set(Tax)
self.var_actualtotal.set(ST)
self.var_total.set(TT)
elif (self.var_meal.get()=="'Indian Plan'" and
self.var_roomtype.get()=="Deluxe","Single","Luxury"):
q1=float(1600)
q2=float(1200)
q3=float(self.noOfdays.get())
q4=float(q1+q2)
q5=float(q3*q4)
Tax="Rs."+str("%.2f"%((q5)*0.09))
ST="Rs."+str("%.2f"%((q5)))
```

```
TT="Rs."+str("%.2f"%(q5+((q5)*0.09)))
self.var_paidtax.set(Tax)
self.var_actualtotal.set(ST)



self.var_total.set(TT)
```

Hotel Details

```
class Cust_Win:
def __init__(self,root):
self.root=root
self.root.title("Hotel Managment System")
self.root.geometry("1025x424+233+200")
#=================variable================
self.var_ref=StringVar()
x=random.randint(1000,9999)
self.var_ref.set(str(x))
self.var_cust_name=StringVar()
self.var_mother=StringVar()
self.var_gender=StringVar()
self.var_post=StringVar()
self.var_mobile=StringVar()
self.var_email=StringVar()
self.var_nationality=StringVar()
self.var_address=StringVar()
self.var_id_proof=StringVar()
self.var_id_number=StringVar()

# ======================title============================
lbl_title=Label(self.root,text="ADD CUSTOMER DETAILS",font=("times new
roman",15,"bold"),bg="black",fg="gold",bd=4,relief=RIDGE)
lbl_title.place(x=0,y=0,width=1295,height=35)
# ========================logo============================
img2=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\hotellogo.jpg")
```

```
img2=img2.resize((100,40),Image.ANTIALIAS)
self.photoimg2=ImageTk.PhotoImage(img2)
lblimg=Label(self.root,image=self.photoimg2,bd=0,relief=RIDGE)
lblimg.place(x=5,y=2,width=80,height=30)



# ====================LABLE FRAME====================
labelframeleft=LabelFrame(self.root,bd=2,relief=RIDGE,text="Customer
Dtails",font=("times new roman",11,"bold"),padx=2)
labelframeleft.place(x=5,y=35,width=340,height=400)
#  ========================Lable & Entry====================
#  custRef
lbl_cust_ref=Label(labelframeleft,text="Customer
Ref",font=("arial",9,"bold"),padx=2,pady=6)
lbl_cust_ref.grid(row=0,column=0,sticky=W)
enty_ref=ttk.Entry(labelframeleft,textvariable=self.var_ref,font=("arial",10,"bold"),wid
th=29,state="readonly")
enty_ref.grid(row=0,column=1)
# cust name
cname=Label(labelframeleft,font=("arial",9,"bold"),text="Customer
Name:",padx=2,pady=6)
cname.grid(row=1,column=0,sticky=W)
txtmname=ttk.Entry(labelframeleft,textvariable=self.var_cust_name,font=("arial",10,"b
old"),width=29)
txtmname.grid(row=1,column=1)
# mother name
lblmname=Label(labelframeleft,font=("arial",9,"bold"),text="Mother
Name:",padx=2,pady=6)
lblmname.grid(row=2,column=0,sticky=W)
txtmname=ttk.Entry(labelframeleft,textvariable=self.var_mother,font=("arial",10,"bold
"),width=29)
txtmname.grid(row=2,column=1)
# gender combox
label_gender=Label(labelframeleft,font=("arial",9,"bold"),text="Gender:",padx=2,pady
```

```
=5)
label_gender.grid(row=3,column=0,sticky=W)
combo_gender=ttk.Combobox(labelframeleft,textvariable=self.var_gender,font=("arial
",10,"bold"),width=27,state="readonly")
combo_gender["value"]=("Male","Female","Other")



combo_gender.current(0)
combo_gender.grid(row=3,column=1)
# postcode
lblPostCode=Label(labelframeleft,font=("arial",9,"bold"),text="PostCode:",padx=2,pa
dy=5)
lblPostCode.grid(row=4,column=0,sticky=W)
txtMobile=ttk.Entry(labelframeleft,textvariable=self.var_post,font=("arial",10,"bold"),
width=29)
txtMobile.grid(row=4,column=1)
# mobilenumber
lblMobile=Label(labelframeleft,font=("arial",9,"bold"),text="Mobile:",padx=2,pady=)
lblMobile.grid(row=5,column=0,sticky=W)
txtMobile=ttk.Entry(labelframeleft,textvariable=self.var_mobile,font=("arial",10,"bold
"),width=29)txtMobile.grid(row=5,column=1)

# email

lblEmail=Label(labelframeleft,font=("arial",9,"bold"),text="Email:",padx=2,pady=5)
lblEmail.grid(row=6,column=0,sticky=W)
txtEmail=ttk.Entry(labelframeleft,textvariable=self.var_email,font=("arial",10,"bold"),
width=29)
txtEmail.grid(row=6,column=1)

# nationality
lblNationality=Label(labelframeleft,font=("arial",9,"bold"),text="Nationality:",padx=2,
pady=5)
lblNationality.grid(row=7,column=0,sticky=W)
```

```
combo_Nationality=ttk.Combobox(labelframeleft,textvariable=self.var_nationality,font
=("arial",10,"bold"),width=27,state="readonly")
combo_Nationality["value"]=("Indian","American","British")
combo_Nationality.current(0)
combo_Nationality.grid(row=7,column=1)




# idproof typr combobox
lblIdProof=Label(labelframeleft,font=("arial",9,"bold"),text="Id Proof
Type:",padx=2,pady=5)
lblIdProof.grid(row=8,column=0,sticky=W)
combo_Id=ttk.Combobox(labelframeleft,textvariable=self.var_id_proof,font=("arial",1
0,"bold"),width=27,state="readonly")
combo_Id["value"]=("Aadhar Card","Driving Licence","Passport")
combo_Id.current(0)
combo_Id.grid(row=8,column=1)
# id number
lblIdNumber=Label(labelframeleft,font=("arial",9,"bold"),text="Id
Number:",padx=2,pady=5)
lblIdNumber.grid(row=9,column=0,sticky=W)

txtIdNumber=ttk.Entry(labelframeleft,textvariable=self.var_id_number,font=("arial",1
0,"bold"),width=29)
txtIdNumber.grid(row=9,column=1)
# address
lblAddress=Label(labelframeleft,font=("arial",9,"bold"),text="Address:",padx=2,pady=
5)
lblAddress.grid(row=10,column=0,sticky=W)
txtAddress=ttk.Entry(labelframeleft,textvariable=self.var_address,font=("arial",10,"bol
d"),width=29)
txtAddress.grid(row=10,column=1)
# ==================Btns================================
btn_frame=Frame(labelframeleft,bd=2,relief=RIDGE)
```

```
btn_frame.place(x=0,y=330,width=333,height=35)
btnAdd=Button(btn_frame,text="Add",command=self.add_data,font=("arial",9,"bold")
,bg="black",fg="gold",width=10)
btnAdd.grid(row=0,column=0,padx=1)
btnupdate=Button(btn_frame,text="Update",command=self.update,font=("arial",9,"bol
d"),bg="black",fg="gold",width=10)
btnupdate.grid(row=0,column=1,padx=1)



btnDelete=Button(btn_frame,text="Delete",command=self.mDelete,font=("arial",9,"bo
ld"),bg="black",fg="gold",width=10)
btnDelete.grid(row=0,column=2,padx=1)
btnReset=Button(btn_frame,text="Reset",command=self.reset,font=("arial",9,"bold"),b
g="black",fg="gold",width=10)
btnReset.grid(row=0,column=3,padx=1)
# ======================table frame search system==============
Table_Frame=LabelFrame(self.root,bd=2,relief=RIDGE,text="View Details And
Search System ",font=("arial",11,"bold"),padx=2)
Table_Frame.place(x=350,y=35,width=666,height=387)
lblSearchBy=Label(Table_Frame,font=("arial",11,"bold"),text="Search
By:",bg="red",fg="white")
lblSearchBy.grid(row=0,column=0,sticky=W)
self.search_var=StringVar()
combo_Search=ttk.Combobox(Table_Frame,textvariable=self.search_var,font=("arial",
11,"bold"),width=17,state="readonly")
combo_Search["value"]=("Mobile","Ref")
combo_Search.current(0)
combo_Search.grid(row=0,column=1,padx=2)
self.txt_search=StringVar()
txtSearch=ttk.Entry(Table_Frame,textvariable=self.txt_search,font=("arial",12,"bold"),
width=17)
txtSearch.grid(row=0,column=2,padx=2)
btnSearch=Button(Table_Frame,text="Search",command=self.search,font=("arial",11,"
bold"),bg="black",fg="gold",width=9)
```

```
btnSearch.grid(row=0,column=3,padx=1)
btnShowAll=Button(Table_Frame,text="Show
All",command=self.fetch_data,font=("arial",11,"bold"),bg="black",fg="gold",width=9)
btnShowAll.grid(row=0,column=4,padx=1)
# ====================SHOW DATA=======================
details_table=Frame(Table_Frame,bd=2,relief=RIDGE)
details_table.place(x=0,y=50,width=655,height=300)
scroll_x=ttk.Scrollbar(details_table,orient=HORIZONTAL)



scroll_y=ttk.Scrollbar(details_table,orient=VERTICAL)

self.Cust_Details_Table=ttk.Treeview(details_table,column=("ref","name","mother","g
ender","post","mobile","email","nationality","idproof","idnumber","address"),xscrollco
mmand=scroll_x.set,yscrollcommand=scroll_x.set)

scroll_x.pack(side=BOTTOM,fill=X)
scroll_y.pack(side=RIGHT,fill=Y)

scroll_x.config(command=self.Cust_Details_Table.xview)
scroll_y.config(command=self.Cust_Details_Table.yview)
self.Cust_Details_Table.heading("ref",text="Refer No")
self.Cust_Details_Table.heading("name",text="Name")
self.Cust_Details_Table.heading("mother",text="Mother Name")
self.Cust_Details_Table.heading("gender",text="Gender")
self.Cust_Details_Table.heading("post",text="PostCode")
self.Cust_Details_Table.heading("mobile",text="Mobile")
self.Cust_Details_Table.heading("email",text="Email")
self.Cust_Details_Table.heading("nationality",text="Nationality")
self.Cust_Details_Table.heading("idproof",text="Id Proof")
self.Cust_Details_Table.heading("idnumber",text="Id Number")
self.Cust_Details_Table.heading("address",text="Address")
self.Cust_Details_Table["show"]="headings"
self.Cust_Details_Table.column("ref",width=100)
self.Cust_Details_Table.column("name",width=100)
self.Cust_Details_Table.column("mother",width=100)
self.Cust_Details_Table.column("gender",width=100)
self.Cust_Details_Table.column("post",width=100)
self.Cust_Details_Table.column("mobile",width=100)
self.Cust_Details_Table.column("email",width=100)
self.Cust_Details_Table.column("nationality",width=100)
self.Cust_Details_Table.column("idproof",width=100)
self.Cust_Details_Table.column("idnumber",width=100)
self.Cust_Details_Table.column("address",width=100)
self.Cust_Details_Table.pack(fill=BOTH,expand=1)
self.Cust_Details_Table.bind("<ButtonRelease-1>",self.get_cuersor)
self.fetch_data()
```

```
def add_data(self):
if self.var_mobile.get()=="" or self.var_mother.get()=="":
messagebox.showerror("Error","All Fields are required",parent=self.root)
else:
try:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("insert into customer
values(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)",(
self.var_ref.get(),
self.var_cust_name.get(),
self.var_mother.get(),
self.var_gender.get(),




self.var_post.get(),
self.var_mobile.get(),
self.var_email.get(),
self.var_nationality.get(),
self.var_id_proof.get(),
self.var_id_number.get(),
self.var_address.get(),
))
conn.commit()
self.fetch_data()
conn.close()
messagebox.showinfo("Success","customer has been added",parent=self.root)
except Exception as es:
messagebox.showwarning("Warning",f"Some thing went
wrong: {str(es)}",parent=self.root)
def fetch_data(self):
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("select * from customer")
rows=my_cursor.fetchall()
if len(rows)!=0:
self.Cust_Details_Table.delete(*self.Cust_Details_Table.get_children())
```

```
for i in rows:
self.Cust_Details_Table.insert("",END,values=i)
conn.commit()
conn.close()
def get_cuersor(self,eveny=""):
cursor_row=self.Cust_Details_Table.focus()
content=self.Cust_Details_Table.item(cursor_row)
row=content["values"]
self.var_ref.set(row[0]),
self.var_cust_name.set(row[1]),



self.var_mother.set(row[2]),
self.var_gender.set(row[3]),
self.var_post.set(row[4]),
self.var_mobile.set(row[5]),
self.var_email.set(row[6]),
self.var_nationality.set(row[7]),
self.var_id_proof.set(row[8]),
self.var_id_number.set(row[9]),
self.var_address.set(row[10])
def update(self):
if self.var_mobile.get()=="":
messagebox.showerror("Eroor","Please enter mobile number",parent=self.root)
else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("update customer set
Name=%s,Mother=%s,Gender=%s,PostCode=%s,Mobile=%s,Email=%s,Nationality=
%s,Idproof=%s,Idnumber=%s,Address=%s where Ref=%s",(
self.var_cust_name.get(),
self.var_mother.get(),
self.var_gender.get(),
```

```
self.var_post.get(),
self.var_mobile.get(),
self.var_email.get(),
self.var_nationality.get(),
self.var_id_proof.get(),
self.var_id_number.get(),
self.var_address.get(),
self.var_ref.get()
))
conn.commit()
self.fetch_data()



conn.close()
messagebox.showinfo("Update","Customer details has been updated
successfully",parent=self.root)
def mDelete(self):
mDelete=messagebox.askyesno("Hotel Managment System","DO you want to delete
this customer details",parent=self.root)
if mDelete>0:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
query="delete from customer where Ref=%s"
value=(self.var_ref.get(),)
my_cursor.execute(query,value)
else:
if not mDelete:
return
conn.commit()
self.fetch_data()
conn.close()
def reset(self):
self.var_ref.set(""),
```

```
self.var_cust_name.set(""),
self.var_mother.set(""),
# self.var_gender.set(""),
self.var_post.set(""),
self.var_mobile.set(""),
self.var_email.set(""),
# self.var_nationality.set(""),
# self.var_id_proof.set(""),
self.var_id_number.set(""),
self.var_address.set("")
x=random.randint(1000,9999)
self.var_ref.set(str(x))



def search(self):
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("select * from customer where "+str(self.search_var.get())+" LIKE
'%"+str(self.txt_search.get())+"%'")
rows=my_cursor.fetchall()
if len (rows)!=0:
self.Cust_Details_Table.delete(*self.Cust_Details_Table.get_children())
for i in rows:
self.Cust_Details_Table.insert("",END,values=i)
conn.commit()
conn.close()
```

Details

```
class DeatailsRoom:
def __init__(self,root):
self.root=root
self.root.title("Hotel Managment System")
self.root.geometry("1025x424+233+200")

# ======================title==============================
lbl_title=Label(self.root,text="ROOM BOOKING DETAILS",font=("times new
roman",18,"bold"),bg="black",fg="gold",bd=4,relief=RIDGE)
```

```
lbl_title.place(x=0,y=0,width=1100,height=50)

# ========================logo============================
img2=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\hotellogo.jpg")
img2=img2.resize((100,40),Image.ANTIALIAS)
self.photoimg2=ImageTk.PhotoImage(img2)

lblimg=Label(self.root,image=self.photoimg2,bd=0,relief=RIDGE)
lblimg.place(x=5,y=2,width=100,height=40)

# ======================LABLE FRAME====================
labelframeleft=LabelFrame(self.root,bd=2,relief=RIDGE,text="New Room
Add",font=("times new roman",14,"bold"),padx=2)
labelframeleft.place(x=5,y=50,width=425,height=180)

#  Floor
lbl_floor=Label(labelframeleft,text="Floor:",font=("arial",12,"bold"),padx=2,pady=6)
lbl_floor.grid(row=0,column=0,sticky=W)




self.var_floor=StringVar()
enty_floor=ttk.Entry(labelframeleft,textvariable=self.var_floor,font=("arial",13,"bold")
,width=20)
enty_floor.grid(row=0,column=1,sticky=W)
#  Room No
lbl_RoomNo=Label(labelframeleft,text="Room
no:",font=("arial",12,"bold"),padx=2,pady=6)
lbl_RoomNo.grid(row=1,column=0,sticky=W)


self.var_roomNo=StringVar()
enty_RoomNo=ttk.Entry(labelframeleft,textvariable=self.var_roomNo,font=("arial",13,
"bold"),width=20)
enty_RoomNo.grid(row=1,column=1,sticky=W)


#  Room Type
lbl_RoomType=Label(labelframeleft,text="Room
Type:",font=("arial",12,"bold"),padx=2,pady=6)
lbl_RoomType.grid(row=2,column=0,sticky=W)


self.var_RoomType=StringVar()
```

```
enty_RoomType=ttk.Entry(labelframeleft,textvariable=self.var_RoomType,font=("aria
l",13,"bold"),width=20)
enty_RoomType.grid(row=2,column=1,sticky=W)

#====================meal frame====================
Mealframeleft=LabelFrame(self.root,bd=2,relief=RIDGE,text="Add
Meal",font=("times new roman",14,"bold"),padx=2)
Mealframeleft.place(x=5,y=230,width=425,height=180)

#  NO of Meal in Plan
lbl_NO_of_Meal=Label(Mealframeleft,text="No Of
Meals:",font=("arial",12,"bold"),padx=2,pady=6)
lbl_NO_of_Meal.grid(row=0,column=0,sticky=W)




self.var_NoofMeal=StringVar()
enty_No_of_meal=ttk.Entry(Mealframeleft,textvariable=self.var_NoofMeal,font=("ari
al",13,"bold"),width=20)
enty_No_of_meal.grid(row=0,column=1,sticky=W)

#  Meal Name
lbl_Meal_Plan=Label(Mealframeleft,text="Meal in
Plan:",font=("arial",12,"bold"),padx=2,pady=6)
lbl_Meal_Plan.grid(row=1,column=0,sticky=W)
self.var_MealinPlan=StringVar()
enty_Meal_Plan=ttk.Entry(Mealframeleft,textvariable=self.var_MealinPlan,font=("aria
l",13,"bold"),width=20)
enty_Meal_Plan.grid(row=1,column=1,sticky=W)
#  Plan Name
lbl_Plan_Name=Label(Mealframeleft,text="Plan
Name:",font=("arial",12,"bold"),padx=2,pady=6)
lbl_Plan_Name.grid(row=2,column=0,sticky=W)
self.var_PlanName=StringVar()
```

```
enty_Plan_Name=ttk.Entry(Mealframeleft,textvariable=self.var_PlanName,font=("aria
l",13,"bold"),width=20)
enty_Plan_Name.grid(row=2,column=1,sticky=W)

# ==================Btns====================
btn_frame=Frame(labelframeleft,bd=2,relief=RIDGE)
btn_frame.place(x=1,y=110,width=415,height=40)
btnAdd=Button(btn_frame,text="Add",command=self.add_data,font=("arial",11,"bold"
),bg="black",fg="gold",width=10)
btnAdd.grid(row=0,column=0,padx=1)
btnupdate=Button(btn_frame,text="Update",command=self.update,font=("arial",11,"bo
ld"),bg="black",fg="gold",width=10)
btnupdate.grid(row=0,column=1,padx=1)
btnDelete=Button(btn_frame,text="Delete",command=self.mDelete,font=("arial",11,"b
old"),bg="black",fg="gold",width=10)



btnDelete.grid(row=0,column=2,padx=1)
btnReset=Button(btn_frame,text="Reset",command=self.reset_data,font=("arial",11,"b
old"),bg="black",fg="gold",width=10)
btnReset.grid(row=0,column=3,padx=1)

# ==========================Meal Btns====================
btn_frame1=Frame(Mealframeleft,bd=2,relief=RIDGE)
btn_frame1.place(x=1,y=110,width=415,height=40)
btnAdd1=Button(btn_frame1,text="Add",command=self.add_data1,font=("arial",11,"b
old"),bg="black",fg="gold",width=10)
btnAdd1.grid(row=0,column=0,padx=1)
btnupdate1=Button(btn_frame1,text="Update",command=self.update1,font=("arial",11
,"bold"),bg="black",fg="gold",width=10)
btnupdate1.grid(row=0,column=1,padx=1)
btnDelete1=Button(btn_frame1,text="Delete",command=self.mDelete1,font=("arial",1
1,"bold"),bg="black",fg="gold",width=10)
btnDelete1.grid(row=0,column=2,padx=1)
```

```
btnReset1=Button(btn_frame1,text="Reset",command=self.reset_data1,font=("arial",1
1,"bold"),bg="black",fg="gold",width=10)
btnReset1.grid(row=0,column=3,padx=1)

# ====================table frame search system===============
Table_Frame=LabelFrame(self.root,bd=2,relief=RIDGE,text="Show Room
Details",font=("arial",12,"bold"),padx=2)
Table_Frame.place(x=440,y=50,width=590,height=180)
scroll_x=ttk.Scrollbar(Table_Frame,orient=HORIZONTAL)
scroll_y=ttk.Scrollbar(Table_Frame,orient=VERTICAL)
self.room_table=ttk.Treeview(Table_Frame,column=("floor","roomno","roomtype"),xs
crollcommand=scroll_x.set,yscrollcommand=scroll_x.set)
scroll_x.pack(side=BOTTOM,fill=X)
scroll_y.pack(side=RIGHT,fill=Y)
scroll_x.config(command=self.room_table.xview)
scroll_y.config(command=self.room_table.yview)



self.room_table.heading("floor",text="Floor")
self.room_table.heading("roomno",text="Room No")
self.room_table.heading("roomtype",text="Room Type")
self.room_table["show"]="headings"
self.room_table.column("floor",width=100)
self.room_table.column("roomno",width=100)
self.room_table.column("roomtype",width=100)
self.room_table.pack(fill=BOTH,expand=1)
self.room_table.bind("<ButtonRelease-1>",self.get_cuersor)
self.fetch_data()

# ================Meal table frame search system===============

Meal_Table_Frame1=LabelFrame(self.root,bd=2,relief=RIDGE,text="Show Meal
Details",font=("arial",12,"bold"),padx=2)
Meal_Table_Frame1.place(x=440,y=220,width=590,height=180)
```

```
scroll_x=ttk.Scrollbar(Meal_Table_Frame1,orient=HORIZONTAL)
scroll_y=ttk.Scrollbar(Meal_Table_Frame1,orient=VERTICAL)
self.meal_table=ttk.Treeview(Meal_Table_Frame1,column=("NoofMeal","MealinPlan
","PlanName"),xscrollcommand=scroll_x.set,yscrollcommand=scroll_x.set)
scroll_x.pack(side=BOTTOM,fill=X)
scroll_y.pack(side=RIGHT,fill=Y)
self.meal_table.heading("NoofMeal",text="No of Meal")
self.meal_table.heading("MealinPlan",text="Meal in Plan")
self.meal_table.heading("PlanName",text="Plan Name")
self.meal_table["show"]="headings"
self.meal_table.column("NoofMeal",width=100)
self.meal_table.column("MealinPlan",width=100)
self.meal_table.column("PlanName",width=100)
self.meal_table.pack(fill=BOTH,expand=1)
self.meal_table.bind("<ButtonRelease-1>",self.get_cuersor1)
self.fetch_data1()



def add_data(self):
if self.var_floor.get()=="" or self.var_RoomType.get()=="":
messagebox.showerror("Eroor","All fields are requaired",parent=self.root)
else:
try:
        conn=mysql.connector.connect(host="localhost",username="root",password="
Happy2015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("insert into details values(%s,%s,%s)",(
self.var_floor.get(),
self.var_roomNo.get(),
self.var_RoomType.get()
))
conn.commit()
self.fetch_data()
```

```
conn.close()
messagebox.showinfo("Success","New Room Added Successfully",parent=self.root)
except Exception as es:
messagebox.showwarning("Warning",f"Some thing went
wrong:{str(es)}",parent=self.root)
# fetch data
def fetch_data(self):
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")

my_cursor=conn.cursor()
my_cursor.execute("select * from details")
rows=my_cursor.fetchall()
if len(rows)!=0:
self.room_table.delete(*self.room_table.get_children())
for i in rows:
self.room_table.insert("",END,values=i)
conn.commit()



conn.close()

#get cursor
def get_cuersor(self,event=""):
cursor_row=self.room_table.focus()
content=self.room_table.item(cursor_row)
row=content["values"]

self.var_floor.set(row[0]),
self.var_roomNo.set(row[1]),
self.var_RoomType.set(row[2])
def add_data1(self):
if self.var_NoofMeal.get()=="" or self.var_PlanName.get()=="":
messagebox.showerror("Eroor","All fields are requaired",parent=self.root)
```

```
else:
try:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("insert into meal values(%s,%s,%s)",(
self.var_NoofMeal.get(),
self.var_MealinPlan.get(),
self.var_PlanName.get()
))
conn.commit()
self.fetch_data1()
conn.close()
messagebox.showinfo("Success","New Meal Plan Added
Successfully",parent=self.root)
except Exception as es:
messagebox.showwarning("Warning",f"Some thing went
wrong:{str(es)}",parent=self.root)



# fetch data
def fetch_data1(self):
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("select * from meal")
rows=my_cursor.fetchall()
if len(rows)!=0:
self.meal_table.delete(*self.meal_table.get_children())
for i in rows:
self.meal_table.insert("",END,values=i)
conn.commit()
conn.close()
```

```
#get cursor
def get_cuersor1(self,event=""):
cursor_row=self.meal_table.focus()
content=self.meal_table.item(cursor_row)
row=content["values"]
self.var_NoofMeal.set(row[0]),
self.var_MealinPlan.set(row[1]),
self.var_PlanName.set(row[2])
#  update function
def update(self):
if self.var_floor.get()=="":
messagebox.showerror("Error","Please enter Floor number",parent=self.root)
else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("update details set Floor=%s,RoomType=%s where
RoomNo=%s",(
self.var_floor.get(),



self.var_RoomType.get(),
self.var_roomNo.get(),
))
conn.commit()
self.fetch_data()
conn.close()
messagebox.showinfo("Update","New Room details has been updated
successfully",parent=self.root)
#  update function
def update1(self):
if self.var_NoofMeal.get()=="":
messagebox.showerror("Error","Please enter Meal number",parent=self.root)
```

```
else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database=
"managment")
my_cursor=conn.cursor()
my_cursor.execute("update meal set NoofMeal=%s,PlanName=%s where
MealinPlan=%s",(
self.var_NoofMeal.get(),
self.var_PlanName.get(),
self.var_MealinPlan.get(),
))
conn.commit()
self.fetch_data1()
conn.close()
messagebox.showinfo("Update","New Meal details has been updated
successfully",parent=self.root)
def mDelete(self):
mDelete=messagebox.askyesno("Hotel Managment System","DO you want to delete
this Room Details",parent=self.root)
if mDelete>0:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
query="delete from details where RoomNo=%s"
value=(self.var_roomNo.get(),)
my_cursor.execute(query,value)
else:
if not mDelete:
return
conn.commit()
self.fetch_data()
conn.close()
```

```
def mDelete1(self):
mDelete1=messagebox.askyesno("Hotel Managment System","DO you want to delete
this Meal Details",parent=self.root)
if mDelete1>0:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
query="delete from meal where MealinPlan=%s"
value=(self.var_MealinPlan.get(),)
my_cursor.execute(query,value)
else:
if not mDelete1:
return
conn.commit()
self.fetch_data1()
conn.close()
def reset_data(self):
self.var_floor.set(""),
self.var_roomNo.set(""),
self.var_RoomType.set("")
def reset_data1(self):
self.var_NoofMeal.set(""),




self.var_MealinPlan.set(""),
self.var_PlanName.set("")
```

Back-End

Login Window

```
from tkinter import*
from tkinter import ttk
from PIL import  Image,ImageTk #pip install pillow]
from tkinter import messagebox
```

```
import mysql.connector
from cProfile import label
from webbrowser import get
import random
from time import strftime
from datetime import datetime
from cgitb import text
from logging import root
from optparse import Values

def main():
win=Tk()
app=Login_Window(win)
win.mainloop()

class Login_Window:
def __init__(self,root):
self.root=root
self.root.title("Login")
self.root.geometry("1265x635+0+0")
self.bg=ImageTk.PhotoImage(file=r"C:\Users\kumar\OneDrive\Desktop\my_hotel_ma
nagment_system\images\green.jpg")
lbl_bg=Label(self.root,image=self.bg)



lbl_bg.place(x=0,y=0,relwidth=1,relheight=1)
frame=Frame(self.root,bg="black")
frame.place(x=500,y=130,width=320,height=400)
img1=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\Avatar.png")
img1=img1.resize((80,80),Image.ANTIALIAS)
self.photoimg1=ImageTk.PhotoImage(img1)
lblimg1=Label(self.root,image=self.photoimg1,bg="black",borderwidth=0)
lblimg1.place(x=605,y=130,width=120,height=100)
```

```
get_str=Label(frame,text="Get Started",font=("times
of roman",15,"bold"),fg="white",bg="black")
get_str.place(x=105,y=93)

# label
username=lbl=Label(frame,text="Username",font=("times
new roman",13,"bold"),fg="white",bg="black")
username.place(x=60,y=130)
self.txtuser=ttk.Entry(frame,font=("times new roman",10,"bold"))
self.txtuser.place(x=30,y=160,width=270)
password=lbl=Label(frame,text="Password",font=("times
new roman",13,"bold"),fg="white",bg="black")
password.place(x=60,y=190)
self.txtpass=ttk.Entry(frame,font=("times new roman",10,"bold"))
self.txtpass.place(x=30,y=220,width=270)
# ================== Icon Image ========================
img2=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\Avatar.p
ng")
img2=img2.resize((25,25),Image.ANTIALIAS)
self.photoimg2=ImageTk.PhotoImage(img2)
lblimg1=Label(image=self.photoimg2,bg="black",borderwidth=0)
lblimg1.place(x=530,y=260,width=25,height=25)



img3=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\padlock-icon.png")
img3=img3.resize((25,25),Image.ANTIALIAS)
self.photoimg3=ImageTk.PhotoImage(img3)
lblimg1=Label(image=self.photoimg3,bg="black",borderwidth=50)
lblimg1.place(x=530,y=320,width=25,height=25)

# ==================Login Button=====================
```

```
btn_login=Button(frame,text="Login",borderwidth=3,relief=RAISED,command=self.l
ogin,font=("times new
roman",15,"bold"),bd=3,fg="white",bg="red",activeforeground="white",activebackgro
und="red")
btn_login.place(x=100,y=250,width=120,height=33)

# Register Button
registerbtn=Button(frame,text="New
UserRegister",command=self.register_window,font=("timesnew
roman",10,"bold"),borderwidth=0,fg="white",bg="black",activeforeground="white",ac
tivebackground="black")
registerbtn.place(x=5,y=310,width=160)

# forgotpassbtn
registerbtn=Button(frame,text="Forget
Password",command=self.forgot_password_window,font=("times
new
roman",10,"bold"),borderwidth=0,fg="white",bg="black",activeforeground="white",ac
tivebackground="black")
registerbtn.place(x=0,y=330,width=160)
def register_window(self):
self.new_window=Toplevel(self.root)
self.app=Register(self.new_window)
def login(self):




if self.txtuser.get()=="" or self.txtpass.get()=="":
messagebox.showerror("Error","all field required",parent=self.root)
elif self.txtuser.get()=="kapu" and self.txtpass.get()=="ashu":
messagebox.showinfo("Success","Welcome to Hotel Managment System")
else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
```

```
my_cursor=conn.cursor()
my_cursor.execute("select * from register where email=%s and password=%s",(
self.txtuser.get(),
self.txtpass.get()
))
row=my_cursor.fetchone()
#print(row)
if row==None:
messagebox.showerror("Error","Invalid Username & password")
else:
open_main=messagebox.askyesno("YesNo","Access only admin")
if open_main>0:
self.new_window=Toplevel(self.root)
self.app=HotelManagementSystem(self.new_window)
else:
if not open_main:
return
conn.commit()
conn.close()

#---------------------------------------reset password====================
def reset_pass(self):
if self.combo_security_Q.get()=="Select":
messagebox.showerror("Error","Select the security question",parent=self.root2)
elif self.txt_security.get()=="":
messagebox.showerror("Error","Please enter the answer",parent=self.root2)



elif self.txt_newpass.get()=="":
messagebox.showerror("Error","Please enter the new password",parent=self.root2)
else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
```

```
qury=("select * from register where email=%s and securityQ=%s and securityA=%s")
value=(self.txtuser.get(),self.combo_security_Q.get(),self.txt_security.get(),)
my_cursor.execute(qury,value)
row=my_cursor.fetchone()
if row==None:
messagebox.showerror("Error","Please enter corret Answer",parent=self.root2)
else:
query=("update register set password=%s where email=%s")
value=(self.txt_newpass.get(),self.txtuser.get())
my_cursor.execute(query,value)
conn.commit()
conn.close()
messagebox.showinfo("Info","Your password has been reset ,please login new password",parent=self.root2)
self.root2.destroy()
#==============forget password window ==============
def forgot_password_window(self):
if self.txtuser.get()=="":
messagebox.showerror("Error","Please enter the email address to reset password")

else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2015$$",database="managment")
my_cursor=conn.cursor()
query=("select * from register where email=%s")
value=(self.txtuser.get(),)
my_cursor.execute(query,value)



row=my_cursor.fetchone()
if row==None:
messagebox.showerror("My Error","Please enter the valid user name")
else:
conn.close()
```

```
self.root2=Toplevel()
self.root2.title("Forget Password")
self.root2.geometry("330x380+490+160")
l=Label(self.root2,text="Forget Password",font=("times
new roman",18,"bold"),fg="red",bg="white")
l.place(x=0,y=10,relwidth=1)
security_Q=Label(self.root2,text="Select Security Question",font=("times
new roman",12,"bold"),bg="white",fg="black")
security_Q.place(x=50,y=70)

self.combo_security_Q=ttk.Combobox(self.root2,font=("times
new roman",12,"bold"),state="readonly")

self.combo_security_Q["values"]=("Select","Your Birth Place","Your
Nick Name","Your Pet Name")

self.combo_security_Q.place(x=50,y=100,width=230)
self.combo_security_Q.current(0)
security_A=Label(self.root2,text="Security Answer",font=("times
new roman",12,"bold"),bg="white",fg="black")
security_A.place(x=50,y=130)
self.txt_security=ttk.Entry(self.root2,font=("times new roman",12))
self.txt_security.place(x=50,y=155,width=230)
new_password=Label(self.root2,text="New Password",font=("times
new roman",12,"bold"),bg="white",fg="black")
new_password.place(x=50,y=185)
self.txt_newpass=ttk.Entry(self.root2,font=("times new roman",12))
self.txt_newpass.place(x=50,y=210,width=230)



btn=Button(self.root2,text="Reset",command=self.reset_pass,font=("times
new roman",14,"bold"),bg="green",fg="black")
btn.place(x=85,y=250,width=150)
```

Register

```
class Register:
def __init__(self,root):
self.root=root
self.root.title("Register")
self.root.geometry("1265x635+
# ==============variables====================
self.var_fname=StringVar()
self.var_lname=StringVar()
self.var_contact=StringVar()
self.var_email=StringVar()
self.var_securityQ=StringVar()
self.var_securityA=StringVar()
self.var_pass=StringVar()
self.var_confpass=StringVar()
# ================bg image==================
self.bg=ImageTk.PhotoImage(file=r"C:\Users\kumar\OneDrive\Desktop\my_hotel_ma
nagment_system\images\wallpaperflare.com_wallpaper.jpg")
bg_lbl=Label(self.root,image=self.bg)
bg_lbl.place(x=0,y=0,relwidth=1,relheight=1)
# ================left image==================
self.bg1=ImageTk.PhotoImage(file=r"C:\Users\kumar\OneDrive\Desktop\my_hotel_m
anagment_system\images\str1.jpg")
left_lbl=Label(self.root,image=self.bg1)
left_lbl.place(x=50,y=150,width=450,height=400)
# ================== MAin Frame=================
frame=Frame(self.root,bg="white")
frame.place(x=501,y=150,width=675,height=400)



register_lbl=Label(frame,text="REGISTER HERE",font=("times
new roman",18,"bold"),fg="darkgreen",bg="white")
register_lbl.place(x=16,y=15
```

```
# ========================= label and entry================
# -------------------Row1
fname=Label(frame,text="First Name",font=("times
new roman",12,"bold"),bg="white")
fname.place(x=40,y=70)
self.fname_entry=ttk.Entry(frame,textvariable=self.var_fname,font=("times
new roman",12))
self.fname_entry.place(x=40,y=95,width=250)
l_name=Label(frame,text="Last Name",font=("times
new roman",12,"bold"),bg="white")
l_name.place(x=368,y=70)
self.txt_lname=ttk.Entry(frame,textvariable=self.var_lname,font=("times
new roman",12))
self.txt_lname.place(x=370,y=95,width=250)
# -------------------Row 2
contact=Label(frame,text="Contact No",font=("times
new roman",12,"bold"),bg="white")
contact.place(x=40,y=130)
self.txt_contact=ttk.Entry(frame,textvariable=self.var_contact,font=("times
new roman",12))
self.txt_contact.place(x=40,y=155,width=250)
email=Label(frame,text="Email",font=("times
new roman",12,"bold"),bg="white")
email.place(x=368,y=130)
self.txt_contact=ttk.Entry(frame,textvariable=self.var_email,font=("times
new roman",13))
self.txt_contact.place(x=370,y=155,width=250)
# ------------- Row3
security_Q=Label(frame,text="Select Security Question",font=("times
new roman",12,"bold"),bg="white",fg="black")



security_Q.place(x=40,y=190)
self.combo_security_Q=ttk.Combobox(frame,textvariable=self.var_securityQ,
```

```
font=("times new roman",12,"bold"),state="readonly")
self.combo_security_Q["values"]=("Select","Your Birth Place","Your
Nick Name","Your Pet Name")
self.combo_security_Q.place(x=40,y=215,width=250)
self.combo_security_Q.current(0)
security_A=Label(frame,text="Security
Answer",font=("times new roman",12,"bold"),bg="white",fg="black")
security_A.place(x=368,y=190)
self.txt_security=ttk.Entry(frame,textvariable=self.var_securityA,font=("times new
roman",12))
self.txt_security.place(x=368,y=215,width=250)

# ---------------------- Row 4
pswd=Label(frame,text="Password",font=("times
new roman",12,"bold"),bg="white",fg="black")
pswd.place(x=40,y=251)
self.txt_pswd=ttk.Entry(frame,textvariable=self.var_pass,font=("times
new roman",12))
self.txt_pswd.place(x=40,y=274,width=250)
confirm_pswd=Label(frame,text="Confirm Password",font=("times
new roman",12,"bold"),bg="white",fg="black")
confirm_pswd.place(x=368,y=251)
self.txt_confirm=ttk.Entry(frame,textvariable=self.var_confpass,font=("times
new roman",12))
self.txt_confirm.place(x=368,y=274,width=250)
# ============= Check Out Button =====================
self.var_check=IntVar()
checkbtn=Checkbutton(frame,variable=self.var_check,text="I Agree The Terms &
Conditions",font=("times new roman",9,"bold"),onvalue=1,offvalue=0)
checkbtn.place(x=40,y=310)



#======================== Button ===================
```

```
img=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system\images\register.jpg")
img=img.resize((200,55),Image.ANTIALIAS)
self.photoimage=ImageTk.PhotoImage(img)
b1=Button(frame,image=self.photoimage,command=self.register_data,borderwidth=0,cursor="hand2",font=("times new roman",12,"bold"),bg="white")
b1.place(x=20,y=335,width=250)
img1=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system\images\login now.png")
img1=img1.resize((200,55),Image.ANTIALIAS)
self.photoimage1=ImageTk.PhotoImage(img1)
b1=Button(frame,image=self.photoimage1,command=self.return_login,borderwidth=0,cursor="hand2",font=("times new roman",12,"bold"),bg="white")
b1.place(x=380,y=335,width=250)
# ================== Function Declaration===============
def register_data(self):
if self.var_fname.get()=="" or self.var_email.get()==""
or self.var_securityQ.get()=="Select":
messagebox.showerror("Eroor","All fields are requaired",parent=self.root)
elif self.var_pass.get()!=self.var_confpass.get():
messagebox.showerror("Error","password & confirm password must be same")
elif self.var_check.get()==0:
messagebox.showerror("Error","Please agree our terms and condition")
else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2015$$",database="managment")
my_cursor=conn.cursor()
query=("select * from register where email=%s")
value=(self.var_email.get(),)
my_cursor.execute(query,value)
row=my_cursor.fetchone()
if row!=None:
```

```
messagebox.showerror("Error","User already exist,please try another email")
else:
my_cursor.execute("insert into register values(%s,%s,%s,%s,%s,%s,%s)",(
self.var_fname.get(),
self.var_lname.get(),
self.var_contact.get(),
self.var_email.get(),
self.var_securityQ.get(),
self.var_securityA.get(),
self.var_pass.get()
))
conn.commit()
conn.close()
messagebox.showinfo("Success","Register Successfully")
def return_login(self):
class Register:
def __init__(self,root):
self.root=root
self.root.title("Register")
self.root.geometry("1265x635+0+0")
# ==============variables====================
self.var_fname=StringVar()
self.var_lname=StringVar()
self.var_contact=StringVar()
self.var_email=StringVar()
self.var_securityQ=StringVar()
self.var_securityA=StringVar()
self.var_pass=StringVar()
self.var_confpass=StringV

# ================bg image=================
self.bg=ImageTk.PhotoImage(file=r"C:\Users\kumar\OneDrive\Desktop\my_hotel_ma
nagment_system\images\wallpaperflare.com_wallpaper.jpg")
```

```
bg_lbl=Label(self.root,image=self.bg)
bg_lbl.place(x=0,y=0,relwidth=1,relheight=1)
# ================left image==================
self.bg1=ImageTk.PhotoImage(file=r"C:\Users\kumar\OneDrive\Desktop\my_hotel_m
anagment_system\images\str1.jpg")
left_lbl=Label(self.root,image=self.bg1)
left_lbl.place(x=50,y=150,width=450,height=400)
# ================== MAin Frame=================
frame=Frame(self.root,bg="white")
frame.place(x=501,y=150,width=675,height=400)
register_lbl=Label(frame,text="REGISTER HERE",font=("times
new roman",18,"bold"),fg="darkgreen",bg="white")
register_lbl.place(x=16,y=15
# ======================== label and entry================
# -------------------Row1
fname=Label(frame,text="First Name",font=("times
new roman",12,"bold"),bg="white")
fname.place(x=40,y=70)
self.fname_entry=ttk.Entry(frame,textvariable=self.var_fname,font=("times
new roman",12))
self.fname_entry.place(x=40,y=95,width=250)
l_name=Label(frame,text="Last Name",font=("times
new roman",12,"bold"),bg="white")
l_name.place(x=368,y=70)
self.txt_lname=ttk.Entry(frame,textvariable=self.var_lname,font=("times
new roman",12))
self.txt_lname.place(x=370,y=95,width=250)
# -------------------Row 2
contact=Label(frame,text="Contact No",font=("times
new roman",12,"bold"),bg="white")
contact.place(x=40,y=130)
self.txt_contact=ttk.Entry(frame,textvariable=self.var_contact,font=("times
new roman",12))
```

```
self.txt_contact.place(x=40,y=155,width=250)
email=Label(frame,text="Email",font=("times
new roman",12,"bold"),bg="white")
email.place(x=368,y=130)
self.txt_contact=ttk.Entry(frame,textvariable=self.var_email,font=("times
new roman",13))
self.txt_contact.place(x=370,y=155,width=250)
# ------------- Row3
security_Q=Label(frame,text="Select Security Question",font=("times
new roman",12,"bold"),bg="white",fg="black")
security_Q.place(x=40,y=190)
self.combo_security_Q=ttk.Combobox(frame,textvariable=self.var_securityQ,
font=("times new roman",12,"bold"),state="readonly")
self.combo_security_Q["values"]=("Select","Your Birth Place","Your
Nick Name","Your Pet Name")
self.combo_security_Q.place(x=40,y=215,width=250)
self.combo_security_Q.current(0)
security_A=Label(frame,text="Security
Answer",font=("times new roman",12,"bold"),bg="white",fg="black")
security_A.place(x=368,y=190)
self.txt_security=ttk.Entry(frame,textvariable=self.var_securityA,font=("times new
roman",12))
self.txt_security.place(x=368,y=215,width=250)
# ------------------------ ROw 4
pswd=Label(frame,text="Password",font=("times
new roman",12,"bold"),bg="white",fg="black")
pswd.place(x=40,y=251)
self.txt_pswd=ttk.Entry(frame,textvariable=self.var_pass,font=("times
new roman",12))
self.txt_pswd.place(x=40,y=274,width=250)
confirm_pswd=Label(frame,text="Confirm Password",font=("times
new roman",12,"bold"),bg="white",fg="black")
confirm_pswd.place(x=368,y=251)
```

```
self.txt_confirm=ttk.Entry(frame,textvariable=self.var_confpass,font=("times
new roman",12))
self.txt_confirm.place(x=368,y=274,width=250)
# ============= Check Out Button ======================
self.var_check=IntVar()
checkbtn=Checkbutton(frame,variable=self.var_check,text="I Agree
The Terms &
Conditions",font=("times new roman",9,"bold"),onvalue=1,offvalue=0)
checkbtn.place(x=40,y=310)
#======================== Button ==================
img=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system\i
mages\register.jpg")
img=img.resize((200,55),Image.ANTIALIAS)
self.photoimage=ImageTk.PhotoImage(img)
b1=Button(frame,image=self.photoimage,command=self.register_data,borderwidth=0,
cursor="hand2",font=("times new roman",12,"bold"),bg="white")
b1.place(x=20,y=335,width=250)
img1=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\login now.png")
img1=img1.resize((200,55),Image.ANTIALIAS)
self.photoimage1=ImageTk.PhotoImage(img1)
b1=Button(frame,image=self.photoimage1,command=self.return_login,borderwidth=0,
cursor="hand2",font=("times new roman",12,"bold"),bg="white")
b1.place(x=380,y=335,width=250)
# ================== Function Declaration===============
def register_data(self):
if self.var_fname.get()=="" or self.var_email.get()==""
or self.var_securityQ.get()=="Select":
messagebox.showerror("Eroor","All fields are requaired",parent=self.root)
elif self.var_pass.get()!=self.var_confpass.get():
messagebox.showerror("Error","password & confirm password must be same")
elif self.var_check.get()==0:
```

```
messagebox.showerror("Error","Please agree our terms and condition")



else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
query=("select * from register where email=%s")
value=(self.var_email.get(),)
my_cursor.execute(query,value)
row=my_cursor.fetchone()
if row!=None:
messagebox.showerror("Error","User already exist,please try another email")
else:
my_cursor.execute("insert into register values(%s,%s,%s,%s,%s,%s,%s)",(
self.var_fname.get(),
self.var_lname.get(),
self.var_contact.get(),
self.var_email.get(),
self.var_securityQ.get(),
self.var_securityA.get(),
self.var_pass.get()
))
conn.commit()
conn.close()
messagebox.showinfo("Success","Register Successfully")
def return_login(self):
self.root.destroy()
```

(Home) Hotel Management System

```
class HotelManagementSystem:
def __init__(self,root):
self.root=root
```

```
self.root.title("Hotel Managment System")
self.root.geometry("1265x635+0+0")



# ==================1st img=============================
img1=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\hotel1.jpg")
img1=img1.resize((1265,150),Image.ANTIALIAS)
self.photoimg1=ImageTk.PhotoImage(img1)
lblimg=Label(self.root,image=self.photoimg1,bd=4,relief=RIDGE)
lblimg.place(x=0,y=0,width=1260,height=140)
# ========================logo=========================
img2=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\hotellogo.jpg")
img2=img2.resize((230,150),Image.ANTIALIAS)
self.photoimg2=ImageTk.PhotoImage(img2)
lblimg=Label(self.root,image=self.photoimg2,bd=4,relief=RIDGE)
lblimg.place(x=0,y=0,width=230,height=140)
# ==================title=================================
lbl_title=Label(self.root,text="HOTEL MANAGMENT SYSTEM",font=("times
new roman",30,"bold"),bg="black",fg="gold",bd=4,relief=RIDGE)
lbl_title.place(x=0,y=120,width=1260,height=50)
#======================main frame======================
main_frame=Frame(self.root,bd=4,relief=RIDGE)
main_frame.place(x=0,y=160,width=1260,height=470)
#======================menu====================
lbl_menu=Label(main_frame,text="MENU",font=("times
new roman",18,"bold"),bg="black",fg="gold",bd=4,relief=RIDGE)
lbl_menu.place(x=0,y=0,width=230)
# ======================Btn frame================
btn_frame=Frame(main_frame,bd=4,relief=RIDGE)
btn_frame.place(x=0,y=30,width=225,height=160)
cust_btn=Button(btn_frame,text="CUSTOMER",command=self.cust_details,width=27
,font=("times
```

```
new roman",12,"bold"),bg="black",fg="gold",bd=0,cursor="hand1")
cust_btn.grid(row=0,column=0,pady=1)



room_btn=Button(btn_frame,text="ROOM",command=self.roombooking,width=27,fo
nt=("times
new roman",12,"bold"),bg="black",fg="gold",bd=0,cursor="hand1")
room_btn.grid(row=1,column=0,pady=1)
details_btn=Button(btn_frame,text="DETAILS",command=self.details_room,width=2
7,font=("times
new roman",12,"bold"),bg="black",fg="gold",bd=0,cursor="hand1")
details_btn.grid(row=2,column=0,pady=1)
report_btn=Button(btn_frame,text="REPORT",width=27,font=("times
new roman",12,"bold"),bg="black",fg="gold",bd=0,cursor="hand1")
report_btn.grid(row=3,column=0,pady=1)
logout_btn=Button(btn_frame,text="LOGOUT",command=self.logout,width=27,font=
("times
new roman",12,"bold"),bg="black",fg="gold",bd=0,cursor="hand1")
logout_btn.grid(row=4,column=0,pady=1)
==========RIGHT SIDE IMAGE==========
img3=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\hotels.jpg")
img3=img3.resize((1310,590),Image.ANTIALIAS)
self.photoimg3=ImageTk.PhotoImage(img3)
lblimg1=Label(main_frame,image=self.photoimg3,bd=4,relief=RIDGE)
lblimg1.place(x=225,y=0,width=1030,height=465)
# ==========DOWN IMAGES==========
img4=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\hotelfront.jpg")
img4=img4.resize((220,210),Image.ANTIALIAS)
self.photoimg4=ImageTk.PhotoImage(img4)
lblimg1=Label(main_frame,image=self.photoimg4,bd=4,relief=RIDGE)
lblimg1.place(x=0,y=190,width=230,height=150)
```

```
img5=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\hotelfood.jpg")
img5=img5.resize((230,190),Image.ANTIALIAS)
self.photoimg5=ImageTk.PhotoImage(img5)



lblimg1=Label(main_frame,image=self.photoimg5,bd=4,relief=RIDGE)
lblimg1.place(x=0,y=320,width=230,height=145)
def cust_details(self):
self.new_window=Toplevel(self.root)
self.app=Cust_Win(self.new_window)
def roombooking(self):
self.new_window=Toplevel(self.root)
self.app=Roombooking(self.new_window)
def details_room(self):
self.new_window=Toplevel(self.root)
self.app=DeatailsRoom(self.new_window)
def logout(self):
self.root.destroy()
```

Room Booking

```
class Roombooking:
def __init__(self,root):
self.root=root
self.root.title("Hotel Managment System")
self.root.geometry("1025x424+233+200")

# ============== variable==============================
self.var_contact=StringVar()
self.var_checkin=StringVar()
self.var_checkout=StringVar()
self.var_roomtype=StringVar()
self.var_rooomavailable=StringVar()
```

```
self.var_meal=StringVar()
self.noOfdays=StringVar()
self.var_paidtax=StringVar()
self.var_actualtotal=StringVar()
self.var_total=StringVar()



# ======================title==========================
lbl_title=Label(self.root,text="ROOM BOOKING DETAILS",font=("times new roman",15,"bold"),bg="black",fg="gold",bd=4,relief=RIDGE)
lbl_title.place(x=0,y=0,width=1295,height=35)
# ========================logo==========================
img2=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system\images\hotellogo.jpg")
img2=img2.resize((100,40),Image.ANTIALIAS)
self.photoimg2=ImageTk.PhotoImage(img2)
lblimg=Label(self.root,image=self.photoimg2,bd=0,relief=RIDGE)
lblimg.place(x=5,y=2,width=80,height=30)
# ====================LABLE FRAME=======================
labelframeleft=LabelFrame(self.root,bd=2,relief=RIDGE,text="Roombooking Dtails",font=("times new roman",11,"bold"),padx=2)
labelframeleft.place(x=5,y=35,width=340,height=400)
#  ====================Lable & Entry=====================
#  Customer contact
lbl_cust_contact=Label(labelframeleft,text="Customer Contact",font=("arial",9,"bold"),padx=2,pady=6)
lbl_cust_contact.grid(row=0,column=0,sticky=W)
enty_contact=ttk.Entry(labelframeleft,textvariable=self.var_contact,font=("arial",10,"bold"),width=21)
enty_contact.grid(row=0,column=1,sticky=W)
# Fetch Data Button
btnFetchData=Button(labelframeleft,command=self.Fetch_contact,text="Fetch Data",font=("arial",8,"bold"),bg="black",fg="gold",width=8)
btnFetchData.place(x=266,y=4)
```

```
#  Check_in Data
check_in_date=Label(labelframeleft,font=("arial",9,"bold"),text="Check_in
Date:",padx=2,pady=6)
check_in_date.grid(row=1,column=0,sticky=W)
textcheck_in_date=ttk.Entry(labelframeleft,textvariable=self.var_checkin,font=("arial",
10,"bold"),width=29)



textcheck_in_date.grid(row=1,column=1)
#  Check_out Data
lbl_Check_out=Label(labelframeleft,font=("arial",9,"bold"),text="Check_Out
Date:",padx=2,pady=6)
lbl_Check_out.grid(row=2,column=0,sticky=W)
text_Check_out=ttk.Entry(labelframeleft,textvariable=self.var_checkout,font=("arial",1
0,"bold"),width=29)
text_Check_out.grid(row=2,column=1)

#  Room Type
label_RoomType=Label(labelframeleft,font=("arial",9,"bold"),text="Room
Type:",padx=2,pady=6)
label_RoomType.grid(row=3,column=0,sticky=W)
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("select RoomType from details")
ide=my_cursor.fetchall()
combo_RoomType=ttk.Combobox(labelframeleft,textvariable=self.var_roomtype,font
=("arial",10,"bold"),width=27,state="readonly")
combo_RoomType["value"]=ide
combo_RoomType.current(0)
combo_RoomType.grid(row=3,column=1)

#  Available Room
lblRoomAvailable=Label(labelframeleft,font=("arial",9,"bold"),text="Available
```

```
Room:",padx=2,pady=6)
lblRoomAvailable.grid(row=4,column=0,sticky=W)
#textRoomAvailable=ttk.Entry(labelframeleft,textvariable=self.var_rooomavailable,fo
nt=("arial",13,"bold"),width=29)
#textRoomAvailable.grid(row=4,column=1)
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")



my_cursor=conn.cursor()
my_cursor.execute("select RoomNo from details")
rows=my_cursor.fetchall()
combo_RoomNo=ttk.Combobox(labelframeleft,textvariable=self.var_rooomavailable,f
ont=("arial",10,"bold"),width=27,state="readonly")
combo_RoomNo["value"]=rows
combo_RoomNo.current(0)
combo_RoomNo.grid(row=4,column=1)

#  Meal
lblMeal=Label(labelframeleft,font=("arial",9,"bold"),text="Meal:",padx=2,pady=6)
lblMeal.grid(row=5,column=0,sticky=W)
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("select PlanName from meal")
ipl=my_cursor.fetchall()
combo_Meal=ttk.Combobox(labelframeleft,textvariable=self.var_meal,font=("arial",10
,"bold"),width=27,state="readonly")
combo_Meal["value"]=ipl
combo_Meal.current(0)
combo_Meal.grid(row=5,column=1)

#  No Of Days
lblNoOfDays=Label(labelframeleft,font=("arial",9,"bold"),text="No of
```

```
Days:",padx=2,pady=6)
lblNoOfDays.grid(row=6,column=0,sticky=W)
textNoOfDayslblNoOfDays=ttk.Entry(labelframeleft,textvariable=self.noOfdays,font=
("arial",10,"bold"),width=29)
textNoOfDayslblNoOfDays.grid(row=6,column=1)

#  Paid Tax
lblNoOfDays=Label(labelframeleft,font=("arial",9,"bold"),text="Paid




Tax:",padx=2,pady=6)
lblNoOfDays.grid(row=7,column=0,sticky=W)
textNoOfDayslblNoOfDays=ttk.Entry(labelframeleft,textvariable=self.var_paidtax,font
=("arial",10,"bold"),width=29)
textNoOfDayslblNoOfDays.grid(row=7,column=1)

# Sub Total
lblNoOfDays=Label(labelframeleft,font=("arial",9,"bold"),text="Sub
Total:",padx=2,pady=6)
lblNoOfDays.grid(row=8,column=0,sticky=W)
textNoOfDayslblNoOfDays=ttk.Entry(labelframeleft,textvariable=self.var_actualtotal,f
ont=("arial",10,"bold"),width=29)
textNoOfDayslblNoOfDays.grid(row=8,column=1)

# Total Cost
lblNoOfDays=Label(labelframeleft,font=("arial",9,"bold"),text="Total
Cost:",padx=2,pady=6)
lblNoOfDays.grid(row=9,column=0,sticky=W)
textNoOfDayslblNoOfDays=ttk.Entry(labelframeleft,textvariable=self.var_total,font=(
"arial",10,"bold"),width=29)
textNoOfDayslblNoOfDays.grid(row=9,column=1)
# ====================== Bill Button ======================
btnBill=Button(labelframeleft,text="Bill",command=self.total,font=("arial",9,"bold"),b
g="black",fg="gold",width=10)
```

```
btnBill.grid(row=10,column=0,padx=1,sticky=W)
# ======================Btns==============================
btn_frame=Frame(labelframeleft,bd=2,relief=RIDGE)
btn_frame.place(x=0,y=335,width=333,height=35)
btnAdd=Button(btn_frame,text="Add",command=self.add_data,font=("arial",9,"bold")
,bg="black",fg="gold",width=10)
btnAdd.grid(row=0,column=0,padx=1)
btnUpdate=Button(btn_frame,text="Update",command=self.update,font=("arial",9,"bol
d"),bg="black",fg="gold",width=10)



btnUpdate.grid(row=0,column=1,padx=1)
btnDelete=Button(btn_frame,text="Delete",command=self.mDelete,font=("arial",9,"bo
ld"),bg="black",fg="gold",width=10)
btnDelete.grid(row=0,column=2,padx=1)
btnReset=Button(btn_frame,text="Reset",command=self.reset,font=("arial",9,"bold"),b
g="black",fg="gold",width=10)
btnReset.grid(row=0,column=3,padx=1)
# ======================Right side Image==================
img3=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\bed.jpg")
img3=img3.resize((520,300),Image.ANTIALIAS)
self.photoimg3=ImageTk.PhotoImage(img3)
lblimg=Label(self.root,image=self.photoimg3,bd=0,relief=RIDGE)
lblimg.place(x=670,y=37,width=350,height=210)

# ====================Table frame search system==========
Table_Frame=LabelFrame(self.root,bd=2,relief=RIDGE,text="View Details And
Search System",font=("arial",11,"bold"),padx=2)
Table_Frame.place(x=350,y=215,width=670,height=200)
lblSearchBy=Label(Table_Frame,font=("arial",11,"bold"),text="Search
By:",bg="red",fg="white")
lblSearchBy.grid(row=0,column=0,sticky=W)
self.search_var=StringVar()
```

```
combo_Search=ttk.Combobox(Table_Frame,textvariable=self.search_var,font=("arial",
11,"bold"),width=17,state="readonly")
combo_Search["value"]=("Contact","Room")
combo_Search.current(0)
combo_Search.grid(row=0,column=1,padx=2)
self.txt_search=StringVar()
txtSearch=ttk.Entry(Table_Frame,textvariable=self.txt_search,font=("arial",12,"bold"),
width=17)
txtSearch.grid(row=0,column=2,padx=2)
btnSearch=Button(Table_Frame,text="Search",command=self.search,font=("arial",11,"
```

```
bold"),bg="black",fg="gold",width=9)
btnSearch.grid(row=0,column=3,padx=1)
btnShowAll=Button(Table_Frame,text="Show
All",command=self.fetch_data,font=("arial",11,"bold"),bg="black",fg="gold",width=9)
btnShowAll.grid(row=0,column=4,padx=1)
# ======================SHOW DATA====================
details_table=Frame(Table_Frame,bd=2,relief=RIDGE)
details_table.place(x=0,y=50,width=660,height=125)
scroll_x=ttk.Scrollbar(details_table,orient=HORIZONTAL)
scroll_y=ttk.Scrollbar(details_table,orient=VERTICAL)
self.room_table=ttk.Treeview(details_table,column=("contact","checkin","checkout","r
oomtype","rooomavailable","meal","noOfdays"),xscrollcommand=scroll_x.set,yscrollc
ommand=scroll_x.set)
scroll_x.pack(side=BOTTOM,fill=X)
scroll_y.pack(side=RIGHT,fill=Y)
scroll_x.config(command=self.room_table.xview)
scroll_y.config(command=self.room_table.yview)

self.room_table.heading("contact",text="Contact")
self.room_table.heading("checkin",text="Check-in")
self.room_table.heading("checkout",text="Check-out")
self.room_table.heading("roomtype",text="Room Type")
```

```
self.room_table.heading("rooomavailable",text="Room No")
self.room_table.heading("meal",text="Meal")
self.room_table.heading("noOfdays",text="NoOfDays")
self.room_table["show"]="headings"
self.room_table.column("contact",width=100)
self.room_table.column("checkin",width=100)
self.room_table.column("checkout",width=100)
self.room_table.column("roomtype",width=100)
self.room_table.column("rooomavailable",width=100)
self.room_table.column("meal",width=100)
self.room_table.column("noOfdays",width=100)



self.room_table.pack(fill=BOTH,expand=1)
self.room_table.bind("<ButtonRelease-1>",self.get_cuersor)
self.fetch_data()
# ======================add data============================
def add_data(self):
if self.var_contact.get()=="" or self.var_checkin.get()=="":
messagebox.showerror("Eroor","All fields are requaired",parent=self.root)
else:
try:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("insert into room values(%s,%s,%s,%s,%s,%s,%s)",(
self.var_contact.get(),
self.var_checkin.get(),
self.var_checkout.get(),
self.var_roomtype.get(),
self.var_rooomavailable.get(),
self.var_meal.get(),
self.noOfdays.get()
))
```

```
conn.commit()
self.fetch_data()
conn.close()
messagebox.showinfo("Success","Room Booked",parent=self.root)
except Exception as es:
messagebox.showwarning("Warning",f"Some thing went
wrong:{str(es)}",parent=self.root)

# fetch data
def fetch_data(self):
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")



my_cursor=conn.cursor()
my_cursor.execute("select * from room")
rows=my_cursor.fetchall()
if len(rows)!=0:
self.room_table.delete(*self.room_table.get_children())
for i in rows:
self.room_table.insert("",END,values=i)
conn.commit()
conn.close()
#get cursor
def get_cuersor(self,event=""):
cursor_row=self.room_table.focus()
content=self.room_table.item(cursor_row)
row=content["values"]

self.var_contact.set(row[0])
self.var_checkin.set(row[1])
self.var_checkout.set(row[2])
self.var_roomtype.set(row[3])
self.var_rooomavailable.set(row[4])
```

```
self.var_meal.set(row[5])
self.noOfdays.set(row[6])

#  update function
def update(self):
if self.var_contact.get()=="":
messagebox.showerror("Error","Please enter mobile number",parent=self.root)
else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("update room set
check_in=%s,check_out=%s,roomtype=%s,rooomavailable=%s,meal=%s,noOfdays=



%s where Contact=%s",(
self.var_checkin.get()
self.var_checkout.get(),
self.var_roomtype.get(),
self.var_rooomavailable.get(),
self.var_meal.get(),
self.noOfdays.get(),
self.var_contact.get()
))
conn.commit()
self.fetch_data()
conn.close()
messagebox.showinfo("Update","Room details has been updated
successfully",parent=self.root)
def mDelete(self):
mDelete=messagebox.askyesno("Hotel Managment System","DO you want to delete
this customer details",parent=self.root)
if mDelete>0:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
```

```
015$$",database="managment")
my_cursor=conn.cursor()
query="delete from room where Contact=%s"
value=(self.var_contact.get(),)
my_cursor.execute(query,value)
else:
if not mDelete:
return
conn.commit()
self.fetch_data()
conn.close()
def reset(self):
self.var_contact.set("")
self.var_checkin.set("")



self.var_checkout.set("")
self.var_roomtype.set("")
self.var_rooomavailable.set("")
self.var_meal.set("")
self.noOfdays.set("")
self.var_paidtax.set("")
self.var_actualtotal.set("")
self.var_total.set("")
# ========================All  Data Fetch ==================
def Fetch_contact(self):
if self.var_contact.get()=="":
messagebox.showerror("Error","Please enter Contact Number",parent=self.root)
else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
query=("select Name from customer where Mobile=%s")
value=(self.var_contact.get(),)
```

```
my_cursor.execute(query,value)
row=my_cursor.fetchone()
if row==None:
messagebox.showerror("Error","This number Not Found",parent=self.root)
else:
conn.commit()
conn.close()
showDataframe=Frame(self.root,bd=4,relief=RIDGE,padx=2)
showDataframe.place(x=350,y=45,width=310,height=165)
lblName=Label(showDataframe,text="Name:",font=("arial","12","bold"))
lblName.place(x=0,y=0)
lbl=Label(showDataframe,text=row,font=("arial","12","bold"))
lbl.place(x=90,y=0)
# ==========Gender==============
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
```

```
015$$",database="managment")
my_cursor=conn.cursor()
query=("select Gender from customer where Mobile=%s")
value=(self.var_contact.get(),)
my_cursor.execute(query,value)
row=my_cursor.fetchone()
lblGender=Label(showDataframe,text="Gender:",font=("arial","12","bold"))
lblGender.place(x=0,y=30)
lbl2=Label(showDataframe,text=row,font=("arial","12","bold"))
lbl2.place(x=90,y=30)
# ==============email========================
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
query=("select Email from customer where Mobile=%s")
value=(self.var_contact.get(),)
my_cursor.execute(query,value)
```

```
row=my_cursor.fetchone()
lbl3=Label(showDataframe,text=row,font=("arial","12","bold"))
lbl3.place(x=90,y=60)
# ==============Nationality=========================
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
query=("select Nationality from customer where Mobile=%s")
value=(self.var_contact.get(),)
my_cursor.execute(query,value)
row=my_cursor.fetchone()
lblNationality=Label(showDataframe,text="Nationality:",font=("arial","12","bold"))
lblNationality.place(x=0,y=90)
lbl4=Label(showDataframe,text=row,font=("arial","12","bold"))
lbl4.place(x=90,y=90)
# ==============Address=========================



conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
query=("select Address from customer where Mobile=%s")
value=(self.var_contact.get(),)
my_cursor.execute(query,value)
row=my_cursor.fetchone()
lbladdress=Label(showDataframe,text="Address:",font=("arial","12","bold"))
lbladdress.place(x=0,y=120)
lbl1=Label(showDataframe,text=row,font=("arial","12","bold"))
lbl1.place(x=90,y=120)
# search System
def search(self):
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
```

```
my_cursor.execute("select * from room where "+str(self.search_var.get())+" LIKE
'%"+str(self.txt_search.get())+"%'")
rows=my_cursor.fetchall()
if len (rows)!=0:
self.room_table.delete(*self.room_table.get_children())
for i in rows:
self.room_table.insert("",END,values=i)
conn.commit()
conn.close()
def total(self):
inDate=self.var_checkin.get()
outDate=self.var_checkout.get()
inDate=datetime.strptime(inDate,"%d/%m/%Y")
outDate=datetime.strptime(outDate,"%d/%m/%Y")
self.noOfdays.set(abs(outDate-inDate).days)
if (self.var_meal.get()=="'American Plan"' and
self.var_roomtype.get()=="Luxury","Single","Deluxe"):



q1=float(2000)
q2=float(1500)
q3=float(self.noOfdays.get())
q4=float(q1+q2)
q5=float(q3*q4)
Tax="Rs."+str("%.2f"%((q5)*0.09))
ST="Rs."+str("%.2f"%((q5)))
TT="Rs."+str("%.2f"%(q5+((q5)*0.09)))
self.var_paidtax.set(Tax)
self.var_actualtotal.set(ST)
self.var_total.set(TT)
elif (self.var_meal.get()=="'Europe Plan"' and
self.var_roomtype.get()=="Single","Luxury","Deluxe"):
q1=float(1800)
q2=float(1300)
```

```
q3=float(self.noOfdays.get())
q4=float(q1+q2)
q5=float(q3*q4)
Tax="Rs."+str("%.2f"%((q5)*0.09))
ST="Rs."+str("%.2f"%((q5)))
TT="Rs."+str("%.2f"%(q5+((q5)*0.09)))
self.var_paidtax.set(Tax)
self.var_actualtotal.set(ST)
self.var_total.set(TT)
elif (self.var_meal.get()=="'Indian Plan'" and
self.var_roomtype.get()=="Deluxe","Single","Luxury"):
q1=float(1600)
q2=float(1200)
q3=float(self.noOfdays.get())
q4=float(q1+q2)
q5=float(q3*q4)
Tax="Rs."+str("%.2f"%((q5)*0.09))
ST="Rs."+str("%.2f"%((q5)))



TT="Rs."+str("%.2f"%(q5+((q5)*0.09)))
self.var_paidtax.set(Tax)
self.var_actualtotal.set(ST)
self.var_total.set(TT)
```

Hotel Details

```
class Cust_Win:
def __init__(self,root):
self.root=root
self.root.title("Hotel Managment System")
self.root.geometry("1025x424+233+200")
#=================variable================
self.var_ref=StringVar()
```

```
x=random.randint(1000,9999)
self.var_ref.set(str(x))
self.var_cust_name=StringVar()
self.var_mother=StringVar()
self.var_gender=StringVar()
self.var_post=StringVar()
self.var_mobile=StringVar()
self.var_email=StringVar()
self.var_nationality=StringVar()
self.var_address=StringVar()
self.var_id_proof=StringVar()
self.var_id_number=StringVar()
# ======================title============================
lbl_title=Label(self.root,text="ADD CUSTOMER DETAILS",font=("times new roman",15,"bold"),bg="black",fg="gold",bd=4,relief=RIDGE)
lbl_title.place(x=0,y=0,width=1295,height=35)
# ========================logo==============================
img2=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system\images\hotellogo.jpg")



img2=img2.resize((100,40),Image.ANTIALIAS)
self.photoimg2=ImageTk.PhotoImage(img2)
lblimg=Label(self.root,image=self.photoimg2,bd=0,relief=RIDGE)
lblimg.place(x=5,y=2,width=80,height=30)
# ====================LABLE FRAME====================
labelframeleft=LabelFrame(self.root,bd=2,relief=RIDGE,text="Customer Dtails",font=("times new roman",11,"bold"),padx=2)
labelframeleft.place(x=5,y=35,width=340,height=400)
#  ======================Lable & Entry====================
#  custRef
lbl_cust_ref=Label(labelframeleft,text="Customer Ref",font=("arial",9,"bold"),padx=2,pady=6)
lbl_cust_ref.grid(row=0,column=0,sticky=W)
```

```
enty_ref=ttk.Entry(labelframeleft,textvariable=self.var_ref,font=("arial",10,"bold"),wid
th=29,state="readonly")
enty_ref.grid(row=0,column=1)

# cust name
cname=Label(labelframeleft,font=("arial",9,"bold"),text="Customer
Name:",padx=2,pady=6)
cname.grid(row=1,column=0,sticky=W)
txtmname=ttk.Entry(labelframeleft,textvariable=self.var_cust_name,font=("arial",10,"b
old"),width=29)
txtmname.grid(row=1,column=1)

# mother name
lblmname=Label(labelframeleft,font=("arial",9,"bold"),text="Mother
Name:",padx=2,pady=6)
lblmname.grid(row=2,column=0,sticky=W)
txtmname=ttk.Entry(labelframeleft,textvariable=self.var_mother,font=("arial",10,"bold
"),width=29)
txtmname.grid(row=2,column=1)



# gender combox
label_gender=Label(labelframeleft,font=("arial",9,"bold"),text="Gender:",padx=2,pady
=5)
label_gender.grid(row=3,column=0,sticky=W)
combo_gender=ttk.Combobox(labelframeleft,textvariable=self.var_gender,font=("arial
",10,"bold"),width=27,state="readonly")
combo_gender["value"]=("Male","Female","Other")
combo_gender.current(0)
combo_gender.grid(row=3,column=1)
# postcode
lblPostCode=Label(labelframeleft,font=("arial",9,"bold"),text="PostCode:",padx=2,pa
dy=5)
```

```
lblPostCode.grid(row=4,column=0,sticky=W)
txtMobile=ttk.Entry(labelframeleft,textvariable=self.var_post,font=("arial",10,"bold"),
width=29)
txtMobile.grid(row=4,column=1)
# mobilenumber
lblMobile=Label(labelframeleft,font=("arial",9,"bold"),text="Mobile:",padx=2,pady=5
)
lblMobile.grid(row=5,column=0,sticky=W)
txtMobile=ttk.Entry(labelframeleft,textvariable=self.var_mobile,font=("arial",10,"bold
"),width=29)
txtMobile.grid(row=5,column=1)

# email
lblEmail=Label(labelframeleft,font=("arial",9,"bold"),text="Email:",padx=2,pady=5)
lblEmail.grid(row=6,column=0,sticky=W)
txtEmail=ttk.Entry(labelframeleft,textvariable=self.var_email,font=("arial",10,"bold"),
width=29)
txtEmail.grid(row=6,column=1)

# nationality
lblNationality=Label(labelframeleft,font=("arial",9,"bold"),text="Nationality:",padx=2,
```

```
pady=5)
lblNationality.grid(row=7,column=0,sticky=W)
combo_Nationality=ttk.Combobox(labelframeleft,textvariable=self.var_nationality,font
=("arial",10,"bold"),width=27,state="readonly")
combo_Nationality["value"]=("Indian","American","British")
combo_Nationality.current(0)
combo_Nationality.grid(row=7,column=1)
# idproof typr combobox
lblIdProof=Label(labelframeleft,font=("arial",9,"bold"),text="Id Proof
Type:",padx=2,pady=5)
lblIdProof.grid(row=8,column=0,sticky=W)
```

```
combo_Id=ttk.Combobox(labelframeleft,textvariable=self.var_id_proof,font=("arial",10,"bold"),width=27,state="readonly")
combo_Id["value"]=("Aadhar Card","Driving Licence","Passport")
combo_Id.current(0)
combo_Id.grid(row=8,column=1)

# id number
lblIdNumber=Label(labelframeleft,font=("arial",9,"bold"),text="Id Number:",padx=2,pady=5)
lblIdNumber.grid(row=9,column=0,sticky=W)
txtIdNumber=ttk.Entry(labelframeleft,textvariable=self.var_id_number,font=("arial",10,"bold"),width=29)
txtIdNumber.grid(row=9,column=1)

# address
lblAddress=Label(labelframeleft,font=("arial",9,"bold"),text="Address:",padx=2,pady=5)
lblAddress.grid(row=10,column=0,sticky=W)
txtAddress=ttk.Entry(labelframeleft,textvariable=self.var_address,font=("arial",10,"bold"),width=29)
txtAddress.grid(row=10,column=1)




# ==================Btns==================================
btn_frame=Frame(labelframeleft,bd=2,relief=RIDGE)
btn_frame.place(x=0,y=330,width=333,height=35)
btnAdd=Button(btn_frame,text="Add",command=self.add_data,font=("arial",9,"bold"),bg="black",fg="gold",width=10)
btnAdd.grid(row=0,column=0,padx=1)
btnupdate=Button(btn_frame,text="Update",command=self.update,font=("arial",9,"bold"),bg="black",fg="gold",width=10)
btnupdate.grid(row=0,column=1,padx=1)
btnDelete=Button(btn_frame,text="Delete",command=self.mDelete,font=("arial",9,"bo
```

```
ld"),bg="black",fg="gold",width=10)
btnDelete.grid(row=0,column=2,padx=1)
btnReset=Button(btn_frame,text="Reset",command=self.reset,font=("arial",9,"bold"),b
g="black",fg="gold",width=10)
btnReset.grid(row=0,column=3,padx=1)
# ======================table frame search system==============
Table_Frame=LabelFrame(self.root,bd=2,relief=RIDGE,text="View Details And
Search System ",font=("arial",11,"bold"),padx=2)
Table_Frame.place(x=350,y=35,width=666,height=387)
lblSearchBy=Label(Table_Frame,font=("arial",11,"bold"),text="Search
By:",bg="red",fg="white")
lblSearchBy.grid(row=0,column=0,sticky=W)
self.search_var=StringVar()
combo_Search=ttk.Combobox(Table_Frame,textvariable=self.search_var,font=("arial",
11,"bold"),width=17,state="readonly")
combo_Search["value"]=("Mobile","Ref")
combo_Search.current(0)
combo_Search.grid(row=0,column=1,padx=2)
self.txt_search=StringVar()
txtSearch=ttk.Entry(Table_Frame,textvariable=self.txt_search,font=("arial",12,"bold"),
width=17)
txtSearch.grid(row=0,column=2,padx=2)
btnSearch=Button(Table_Frame,text="Search",command=self.search,font=("arial",11,"
bold"),bg="black",fg="gold",width=9)



btnSearch.grid(row=0,column=3,padx=1)
btnShowAll=Button(Table_Frame,text="Show
All",command=self.fetch_data,font=("arial",11,"bold"),bg="black",fg="gold",width=9)
btnShowAll.grid(row=0,column=4,padx=1)
# ====================SHOW DATA======================
details_table=Frame(Table_Frame,bd=2,relief=RIDGE)
details_table.place(x=0,y=50,width=655,height=300)
scroll_x=ttk.Scrollbar(details_table,orient=HORIZONTAL)
scroll_y=ttk.Scrollbar(details_table,orient=VERTICAL)
```

```
self.Cust_Details_Table=ttk.Treeview(details_table,column=("ref","name","mother","g
ender","post","mobile","email","nationality","idproof","idnumber","address"),xscrollco
mmand=scroll_x.set,yscrollcommand=scroll_x.set)

scroll_x.pack(side=BOTTOM,fill=X)
scroll_y.pack(side=RIGHT,fill=Y)

scroll_x.config(command=self.Cust_Details_Table.xview)
scroll_y.config(command=self.Cust_Details_Table.yview)

self.Cust_Details_Table.heading("ref",text="Refer No")
self.Cust_Details_Table.heading("name",text="Name")
self.Cust_Details_Table.heading("mother",text="Mother Name")
self.Cust_Details_Table.heading("gender",text="Gender")
self.Cust_Details_Table.heading("post",text="PostCode")
self.Cust_Details_Table.heading("mobile",text="Mobile")
self.Cust_Details_Table.heading("email",text="Email")
self.Cust_Details_Table.heading("nationality",text="Nationality")
self.Cust_Details_Table.heading("idproof",text="Id Proof")
self.Cust_Details_Table.heading("idnumber",text="Id Number")
self.Cust_Details_Table.heading("address",text="Address")
self.Cust_Details_Table["show"]="headings"
self.Cust_Details_Table.column("ref",width=100)



self.Cust_Details_Table.column("name",width=100)
self.Cust_Details_Table.column("mother",width=100)
self.Cust_Details_Table.column("gender",width=100)
self.Cust_Details_Table.column("post",width=100)
self.Cust_Details_Table.column("mobile",width=100)
self.Cust_Details_Table.column("email",width=100)
self.Cust_Details_Table.column("nationality",width=100)
self.Cust_Details_Table.column("idproof",width=100)
```

```
self.Cust_Details_Table.column("idnumber",width=100)
self.Cust_Details_Table.column("address",width=100)
self.Cust_Details_Table.pack(fill=BOTH,expand=1)
self.Cust_Details_Table.bind("<ButtonRelease-1>",self.get_cuersor)
self.fetch_data()
def add_data(self):
if self.var_mobile.get()=="" or self.var_mother.get()=="":
messagebox.showerror("Error","All Fields are required",parent=self.root)
else:
try:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("insert into customer
values(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)",(
self.var_ref.get(),
self.var_cust_name.get(),
self.var_mother.get(),
self.var_gender.get(),
self.var_post.get(),
self.var_mobile.get(),
self.var_email.get(),
self.var_nationality.get(),
self.var_id_proof.get(),
self.var_id_number.get(),




self.var_address.get(),
))
conn.commit()
self.fetch_data()
conn.close()
messagebox.showinfo("Success","customer has been added",parent=self.root)
except Exception as es:
```

```
messagebox.showwarning("Warning",f"Some thing went
wrong: {str(es)}",parent=self.root)
def fetch_data(self):
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("select * from customer")
rows=my_cursor.fetchall()
if len(rows)!=0:
self.Cust_Details_Table.delete(*self.Cust_Details_Table.get_children())
for i in rows:
self.Cust_Details_Table.insert("",END,values=i)
conn.commit()
conn.close()
def get_cuersor(self,eveny=""):
cursor_row=self.Cust_Details_Table.focus()
content=self.Cust_Details_Table.item(cursor_row)
row=content["values"]
self.var_ref.set(row[0]),
self.var_cust_name.set(row[1]),
self.var_mother.set(row[2]),
self.var_gender.set(row[3]),
self.var_post.set(row[4]),
self.var_mobile.set(row[5]),
self.var_email.set(row[6]),
self.var_nationality.set(row[7]),



self.var_id_proof.set(row[8]),
self.var_id_number.set(row[9]),
self.var_address.set(row[10])
def update(self):
if self.var_mobile.get()=="":
messagebox.showerror("Eroor","Please enter mobile number",parent=self.root)
```

```
  else:
  conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
  015$$",database="managment")
  my_cursor=conn.cursor()
  my_cursor.execute("update customer set
  Name=%s,Mother=%s,Gender=%s,PostCode=%s,Mobile=%s,Email=%s,Nationality=
  %s,Idproof=%s,Idnumber=%s,Address=%s where Ref=%s",(
 self.var_cust_name.get(),
 self.var_mother.get(),
 self.var_gender.get(),
 self.var_post.get(),
 self.var_mobile.get(),
 self.var_email.get(),
 self.var_nationality.get(),
 self.var_id_proof.get(),
 self.var_id_number.get(),
self.var_address.get(),
self.var_ref.get()
))
conn.commit()
  self.fetch_data()
  conn.close()
  messagebox.showinfo("Update","Customer details has been updated
  successfully",parent=self.root)
  def mDelete(self):
  mDelete=messagebox.askyesno("Hotel Managment System","DO you want to delete
  this customer details",parent=self.root)

  if mDelete>0:
  conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
  015$$",database="managment")
  my_cursor=conn.cursor()
  query="delete from customer where Ref=%s"
  value=(self.var_ref.get(),)
  my_cursor.execute(query,value)
```

```
else:
if not mDelete:
return
conn.commit()
self.fetch_data()
conn.close()
def reset(self):
self.var_ref.set(""),
self.var_cust_name.set(""),
self.var_mother.set(""),
# self.var_gender.set(""),
self.var_post.set(""),
self.var_mobile.set(""),
self.var_email.set(""),
# self.var_nationality.set(""),
# self.var_id_proof.set(""),
self.var_id_number.set(""),
self.var_address.set("")
x=random.randint(1000,9999)
self.var_ref.set(str(x))
def search(self):
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("select * from customer where "+str(self.search_var.get())+" LIKE
'%"+str(self.txt_search.get())+"%'")
rows=my_cursor.fetchall()
if len (rows)!=0:
self.Cust_Details_Table.delete(*self.Cust_Details_Table.get_children())
for i in rows:
self.Cust_Details_Table.insert("",END,values=i)
conn.commit()
conn.close()
```

Details

```
class DeatailsRoom:
def __init__(self,root):
self.root=root
self.root.title("Hotel Managment System")
self.root.geometry("1025x424+233+200")
# ======================title==============================
lbl_title=Label(self.root,text="ROOM BOOKING DETAILS",font=("times new
roman",18,"bold"),bg="black",fg="gold",bd=4,relief=RIDGE)
lbl_title.place(x=0,y=0,width=1100,height=50)
# ========================logo============================
img2=Image.open(r"C:\Users\kumar\OneDrive\Desktop\my_hotel_managment_system
\images\hotellogo.jpg")
img2=img2.resize((100,40),Image.ANTIALIAS)
self.photoimg2=ImageTk.PhotoImage(img2)
lblimg=Label(self.root,image=self.photoimg2,bd=0,relief=RIDGE)
lblimg.place(x=5,y=2,width=100,height=40)
# ======================LABLE FRAME====================
labelframeleft=LabelFrame(self.root,bd=2,relief=RIDGE,text="New Room
Add",font=("times new roman",14,"bold"),padx=2)
labelframeleft.place(x=5,y=50,width=425,height=180)

#  Floor
lbl_floor=Label(labelframeleft,text="Floor:",font=("arial",12,"bold"),padx=2,pady=6)



lbl_floor.grid(row=0,column=0,sticky=W)
self.var_floor=StringVar()
enty_floor=ttk.Entry(labelframeleft,textvariable=self.var_floor,font=("arial",13,"bold")
,width=20)
enty_floor.grid(row=0,column=1,sticky=W)

#  Room No
lbl_RoomNo=Label(labelframeleft,text="Room
```

```
no:",font=("arial",12,"bold"),padx=2,pady=6)
lbl_RoomNo.grid(row=1,column=0,sticky=W)
self.var_roomNo=StringVar()
enty_RoomNo=ttk.Entry(labelframeleft,textvariable=self.var_roomNo,font=("arial",13,
"bold"),width=20)
enty_RoomNo.grid(row=1,column=1,sticky=W)

#  Room Type
lbl_RoomType=Label(labelframeleft,text="Room
Type:",font=("arial",12,"bold"),padx=2,pady=6)
lbl_RoomType.grid(row=2,column=0,sticky=W)
self.var_RoomType=StringVar()
enty_RoomType=ttk.Entry(labelframeleft,textvariable=self.var_RoomType,font=("aria
l",13,"bold"),width=20)
enty_RoomType.grid(row=2,column=1,sticky=W)
#====================meal frame====================
Mealframeleft=LabelFrame(self.root,bd=2,relief=RIDGE,text="Add
Meal",font=("times new roman",14,"bold"),padx=2)
Mealframeleft.place(x=5,y=230,width=425,height=180)

#  NO of Meal in Plan
lbl_NO_of_Meal=Label(Mealframeleft,text="No Of
Meals:",font=("arial",12,"bold"),padx=2,pady=6)
lbl_NO_of_Meal.grid(row=0,column=0,sticky=W)
self.var_NoofMeal=StringVar()



enty_No_of_meal=ttk.Entry(Mealframeleft,textvariable=self.var_NoofMeal,font=("ari
al",13,"bold"),width=20)
enty_No_of_meal.grid(row=0,column=1,sticky=W)
#  Meal Name
lbl_Meal_Plan=Label(Mealframeleft,text="Meal in
Plan:",font=("arial",12,"bold"),padx=2,pady=6)
lbl_Meal_Plan.grid(row=1,column=0,sticky=W)
```

```
self.var_MealinPlan=StringVar()
enty_Meal_Plan=ttk.Entry(Mealframeleft,textvariable=self.var_MealinPlan,font=("aria
l",13,"bold"),width=20)
enty_Meal_Plan.grid(row=1,column=1,sticky=W)
#  Plan Name
lbl_Plan_Name=Label(Mealframeleft,text="Plan
Name:",font=("arial",12,"bold"),padx=2,pady=6)
lbl_Plan_Name.grid(row=2,column=0,sticky=W)
self.var_PlanName=StringVar()
enty_Plan_Name=ttk.Entry(Mealframeleft,textvariable=self.var_PlanName,font=("aria
l",13,"bold"),width=20)
enty_Plan_Name.grid(row=2,column=1,sticky=W)
# ==================Btns==================
btn_frame=Frame(labelframeleft,bd=2,relief=RIDGE)
btn_frame.place(x=1,y=110,width=415,height=40)
btnAdd=Button(btn_frame,text="Add",command=self.add_data,font=("arial",11,"bold"
),bg="black",fg="gold",width=10)
btnAdd.grid(row=0,column=0,padx=1)
btnupdate=Button(btn_frame,text="Update",command=self.update,font=("arial",11,"bo
ld"),bg="black",fg="gold",width=10)
btnupdate.grid(row=0,column=1,padx=1)
btnDelete=Button(btn_frame,text="Delete",command=self.mDelete,font=("arial",11,"b
old"),bg="black",fg="gold",width=10)
btnDelete.grid(row=0,column=2,padx=1)
btnReset=Button(btn_frame,text="Reset",command=self.reset_data,font=("arial",11,"b
old"),bg="black",fg="gold",width=10)



btnReset.grid(row=0,column=3,padx=1)
# ========================Meal Btns==================
btn_frame1=Frame(Mealframeleft,bd=2,relief=RIDGE)
btn_frame1.place(x=1,y=110,width=415,height=40)
btnAdd1=Button(btn_frame1,text="Add",command=self.add_data1,font=("arial",11,"b
old"),bg="black",fg="gold",width=10)
```

```
btnAdd1.grid(row=0,column=0,padx=1)
btnupdate1=Button(btn_frame1,text="Update",command=self.update1,font=("arial",11
,"bold"),bg="black",fg="gold",width=10)
btnupdate1.grid(row=0,column=1,padx=1)
btnDelete1=Button(btn_frame1,text="Delete",command=self.mDelete1,font=("arial",1
1,"bold"),bg="black",fg="gold",width=10)
btnDelete1.grid(row=0,column=2,padx=1)
btnReset1=Button(btn_frame1,text="Reset",command=self.reset_data1,font=("arial",1
1,"bold"),bg="black",fg="gold",width=10)
btnReset1.grid(row=0,column=3,padx=1)
# ====================table frame search system===============
Table_Frame=LabelFrame(self.root,bd=2,relief=RIDGE,text="Show Room
Details",font=("arial",12,"bold"),padx=2)
Table_Frame.place(x=440,y=50,width=590,height=180)

scroll_x=ttk.Scrollbar(Table_Frame,orient=HORIZONTAL)
scroll_y=ttk.Scrollbar(Table_Frame,orient=VERTICAL)
self.room_table=ttk.Treeview(Table_Frame,column=("floor","roomno","roomtype"),xs
crollcommand=scroll_x.set,yscrollcommand=scroll_x.set)
scroll_x.pack(side=BOTTOM,fill=X)
scroll_y.pack(side=RIGHT,fill=Y)
scroll_x.config(command=self.room_table.xview)
scroll_y.config(command=self.room_table.yview)
self.room_table.heading("floor",text="Floor")
self.room_table.heading("roomno",text="Room No")
self.room_table.heading("roomtype",text="Room Type")
self.room_table["show"]="headings"



self.room_table.column("floor",width=100)
self.room_table.column("roomno",width=100)
self.room_table.column("roomtype",width=100)
self.room_table.pack(fill=BOTH,expand=1)
self.room_table.bind("<ButtonRelease-1>",self.get_cuersor)
```

```
self.fetch_data()
# =================Meal table frame search system===============
Meal_Table_Frame1=LabelFrame(self.root,bd=2,relief=RIDGE,text="Show Meal
Details",font=("arial",12,"bold"),padx=2)
Meal_Table_Frame1.place(x=440,y=220,width=590,height=180)
scroll_x=ttk.Scrollbar(Meal_Table_Frame1,orient=HORIZONTAL)
scroll_y=ttk.Scrollbar(Meal_Table_Frame1,orient=VERTICAL)
self.meal_table=ttk.Treeview(Meal_Table_Frame1,column=("NoofMeal","MealinPlan
","PlanName"),xscrollcommand=scroll_x.set,yscrollcommand=scroll_x.set)
scroll_x.pack(side=BOTTOM,fill=X)
scroll_y.pack(side=RIGHT,fill=Y)
self.meal_table.heading("NoofMeal",text="No of Meal")
self.meal_table.heading("MealinPlan",text="Meal in Plan")
self.meal_table.heading("PlanName",text="Plan Name")
self.meal_table["show"]="headings"
self.meal_table.column("NoofMeal",width=100)
self.meal_table.column("MealinPlan",width=100)
self.meal_table.column("PlanName",width=100)
self.meal_table.pack(fill=BOTH,expand=1)
self.meal_table.bind("<ButtonRelease-1>",self.get_cuersor1)
self.fetch_data1()
def add_data(self):
if self.var_floor.get()=="" or self.var_RoomType.get()=="":
messagebox.showerror("Eroor","All fields are requaired",parent=self.root)
else:
try:
        conn=mysql.connector.connect(host="localhost",username="root",password="
Happy2015$$",database="managment")



my_cursor=conn.cursor()
my_cursor.execute("insert into details values(%s,%s,%s)",(
self.var_floor.get(),
self.var_roomNo.get(),
```

```
self.var_RoomType.get()
))
conn.commit()
self.fetch_data()
conn.close()
messagebox.showinfo("Success","New Room Added Successfully",parent=self.root)
except Exception as es:
messagebox.showwarning("Warning",f"Some thing went
wrong:{str(es)}",parent=self.root)
# fetch data
def fetch_data(self):
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("select * from details")
rows=my_cursor.fetchall()
if len(rows)!=0:
self.room_table.delete(*self.room_table.get_children())
for i in rows:
self.room_table.insert("",END,values=i)
conn.commit()
conn.close()

#get cursor
def get_cuersor(self,event=""):
cursor_row=self.room_table.focus()
content=self.room_table.item(cursor_row)
row=content["values"]
self.var_floor.set(row[0]),




self.var_roomNo.set(row[1]),
self.var_RoomType.set(row[2])
def add_data1(self):
```

```
if self.var_NoofMeal.get()=="" or self.var_PlanName.get()=="":
messagebox.showerror("Eroor","All fields are requaired",parent=self.root)
else:
try:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("insert into meal values(%s,%s,%s)",(
self.var_NoofMeal.get(),
self.var_MealinPlan.get(),
self.var_PlanName.get()
))
conn.commit()
self.fetch_data1()
conn.close()
messagebox.showinfo("Success","New Meal Plan Added
Successfully",parent=self.root)
except Exception as es:
messagebox.showwarning("Warning",f"Some thing went
wrong:{str(es)}",parent=self.root)

# fetch data
def fetch_data1(self):
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("select * from meal")
rows=my_cursor.fetchall()
if len(rows)!=0:
self.meal_table.delete(*self.meal_table.get_children())



for i in rows:
self.meal_table.insert("",END,values=i)
```

```
conn.commit()
conn.close()

#get cursor
def get_cuersor1(self,event=""):

cursor_row=self.meal_table.focus()
content=self.meal_table.item(cursor_row)
row=content["values"]
self.var_NoofMeal.set(row[0]),
self.var_MealinPlan.set(row[1]),
self.var_PlanName.set(row[2])

#  update function
def update(self):
if self.var_floor.get()=="":
messagebox.showerror("Error","Please enter Floor number",parent=self.root)
else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
my_cursor.execute("update details set Floor=%s,RoomType=%s where
RoomNo=%s",(

self.var_floor.get(),
self.var_RoomType.get(),
self.var_roomNo.get(),
))
conn.commit()
self.fetch_data()
conn.close()



messagebox.showinfo("Update","New Room details has been updated
```

```
successfully",parent=self.root)
#  update function
def update1(self):
if self.var_NoofMeal.get()=="":
messagebox.showerror("Error","Please enter Meal number",parent=self.root)
else:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database=
"managment")
my_cursor=conn.cursor()
my_cursor.execute("update meal set NoofMeal=%s,PlanName=%s where
MealinPlan=%s",(
self.var_NoofMeal.get(),
self.var_PlanName.get(),
self.var_MealinPlan.get(),
))
conn.commit()
self.fetch_data1()
conn.close()
messagebox.showinfo("Update","New Meal details has been updated
successfully",parent=self.root)
def mDelete(self):
mDelete=messagebox.askyesno("Hotel Managment System","DO you want to delete
this Room Details",parent=self.root)
if mDelete>0:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
my_cursor=conn.cursor()
query=delete from details where RoomNo=%s"
value=(self.var_roomNo.get(),)
my_cursor.execute(query,value)
else:
```

```
if not mDelete:
return
conn.commit()
self.fetch_data()
conn.close()
def mDelete1(self):
mDelete1=messagebox.askyesno("Hotel Managment System","DO you want to delete
this Meal Details",parent=self.root)
if mDelete1>0:
conn=mysql.connector.connect(host="localhost",username="root",password="Happy2
015$$",database="managment")
value=(self.var_MealinPlan.get(),)
my_cursor.execute(query,value)
else:
if not mDelete1:
return
conn.commit()
self.fetch_data1()
conn.close()
def reset_data(self):
self.var_floor.set(""),
self.var_roomNo.set(""),
self.var_RoomType.set("")
def reset_data1(self):
self.var_NoofMeal.set(""),
self.var_MealinPlan.set(""),
self.var_PlanName.set("")
def reset_data(self):
self.var_floor.set(""),
self.var_roomNo.set(""),
self.var_NoofMeal.set(""),
self.var_MealinPlan.set(""),
self.var_PlanName.set("")
```

Testing Approach

Software testing is a process used to identify the correctness, completeness and quality of developed computer software. It includes a set of activities conducted with the intent of finding errors in software so that it could be corrected before the product is released to the end users. In other word software testing is an activity to check that the software system is defect free.

Software testing is primarily a broad process that is composed of several interlinked processes. The primary objective of software testing is to measure software health along with its completeness in terms of core requirements. Software testing involves examining and checking software through different testing processes.

The objectives of these processes can include:

- **Completeness** - Verifying software completeness in regards to functional/business requirements
- **Errors Free** - Identifying technical bugs/errors and ensuring the software is error-free
- **Stability** - Assessing usability, performance, security, localization, compatibility and installation

  This phase determines the error in the project. If there is any error then it must be removed before delivery of the project.

Type of Testing

For determining errors various types of test action are performed: -

Unit testing: - Unit testing focuses verification effort on the smallest unit of software design – the module. Using the detail design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The relative complexity of tests and the errors detected as a result is limited by the constrained scope established for unit testing. The unit test is always white box oriented, and the step can be conducted in parallel for multiple modules.

Unit testing is normally considered an adjunct to the coding step. After source level code has been developed, reviewed, and verified for correct syntax, unit test case design begins.

Integration Testing - A level of the software testing process where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units.

System Testing: - Software is only one element of a larger computer-based system. Ultimately, software is incorporated with other system elements (e.g., new hardware, information), and a series of system integration and validation tests are conducted. Steps taken during software design and testing can greatly improve the probability of successful software integration in the larger system.

A classics system testing problem is “finger pointing”. This occurs when a defect is uncovered, and one system element developer blames another for the problem. The software engineer should anticipate potential interfacing problems and design error handling paths that test all information coming from other elements of the system, conduct a series of tests that simulate bad data or other potential errors at the software interface, record the results or tests to use as “evidence” if finger pointing does occur, participate in the planning and design of system test to ensure that software is adequately tested.

There are many types of system tests that are worthwhile for software-based systems: -

Usability Testing - Usability Testing is a type of testing done from an end-user’s perspective to determine if the system is easily usable.

Functionality testing - Tests all functionalities of the software against the requirement.

Performance testing – Performance testing is designed to test the run-time performance of software within the context of an integrated system

Security testing – Security testing attempts to verify that protection mechanisms built into a system will protect it from improper penetration

Stress tests – Stress tests are designed to confront programs with abnormal situations.

Use Case

A use case diagram is essentially a picture showing system behavior along with the key actors that interact with the system. The use case represents complete functionality. Use case diagram can be imagined as a black box where only the input, output, and the function of the black box are known. Use Case elements are used to make test cases when performing the testing. The use case should contain all system activities that have significance to the users. A use case can be thought of as a collection of possible scenarios related to a particular goal, indeed. Use cases can be employed during several stages of software development, such as planning system requirements, validating design and testing software.

Use case Diagram Objects

Use case diagrams mostly consist of 3 objects: -

Actor - Actor is a use case diagram is any entity that performs a role in one given system. This could be a person, organization or an external system.

Use Case - A Use case represents a function or an action within the system. its drawn as an oval and named with the function.

System - System is used to define the define the scope of the use case and drawn as rectangle.

There are two functions: -

Include – This represents required. Symbol of this function is dashed arrow and arrow is labeled with the keyword <<include>>

---------->

Extend – This represents optional and it is also shown with dashed arrow the arrow is labeled with the keyword <<extend>>

<----------

Use Case Diagram

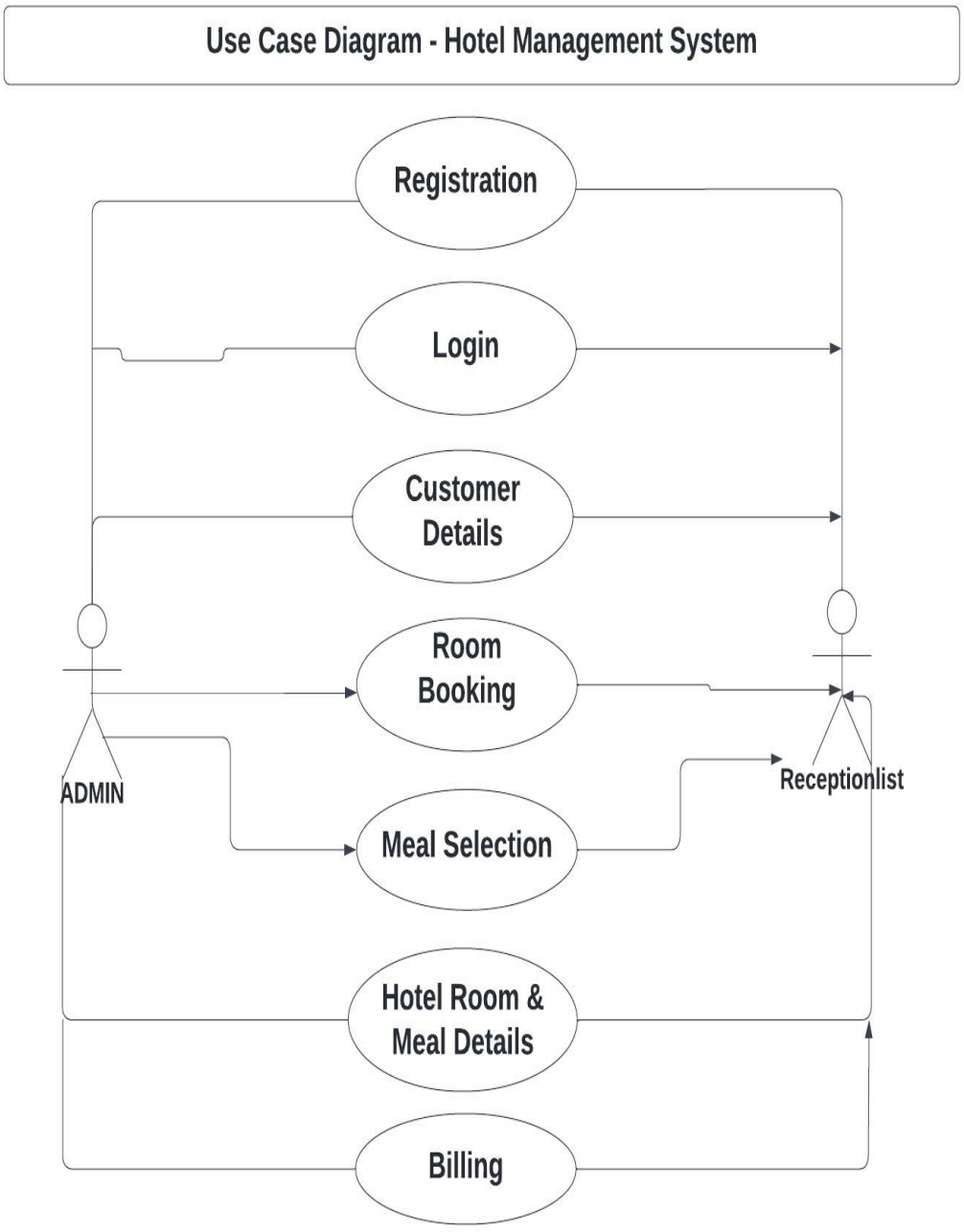


Fig 5: Use Case Diagram

Test case

Test case for Login Page. Test Case Id - TC00I

Sr.No.	Input/ Action	Expected Result	Actual Result	Remark
1	Leave text/ Field empty.	Will show error message “Enter Username" “Enter Password "	Error message “Enter Username" “Enter Password "	Pass
2	Entered Invalid username or password.	Will show error message " “Invalid User "	Error message "Invalid User"	Pass
3	Entered Valid username.	Will accept the data.	Data accepted	Pass

Test case for Search Test Case Id - TC002

Sr. No	Input/Action	Expected Result	Actual Result	Remark
1	Leave text/ field empty	Will show error message ” Select proper search criteria"	Error message “Select proper search criteria"	Pass
2	Enterd Valid data.	Will accept the data.	Data accepted	Pass

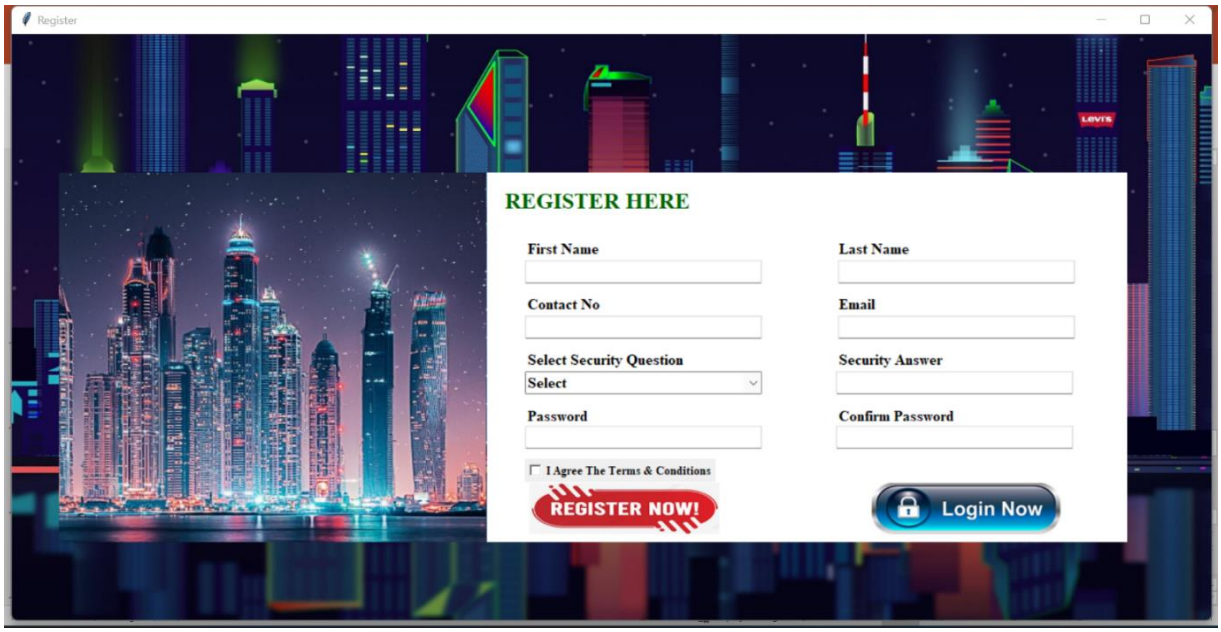
Test case for Report Test Case Id -TC003

Sr.No.	Input/Action	Expected Result	Actual Result	Remark
1.	Leave text/ field empty	Will show error message “0- Record Found”	Error message “0- Record Found”	Pass
2.	Entered Valid data	Will accept the data.	Data accepted	Pass

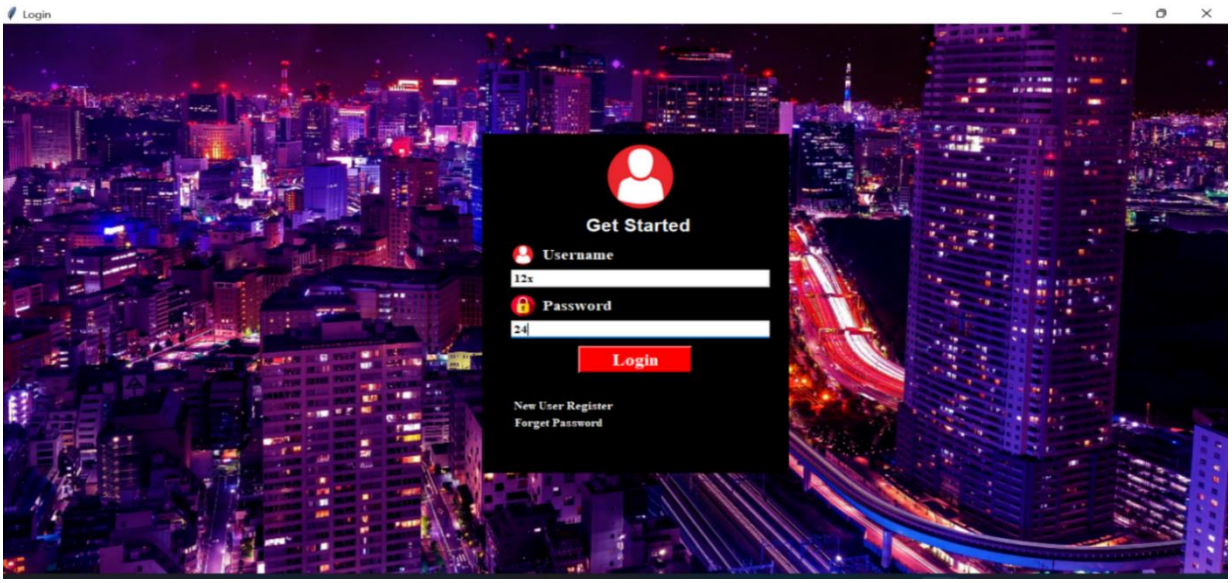
CHAPTER 6:
RESULTS AND DISCUSSION

User Documentation

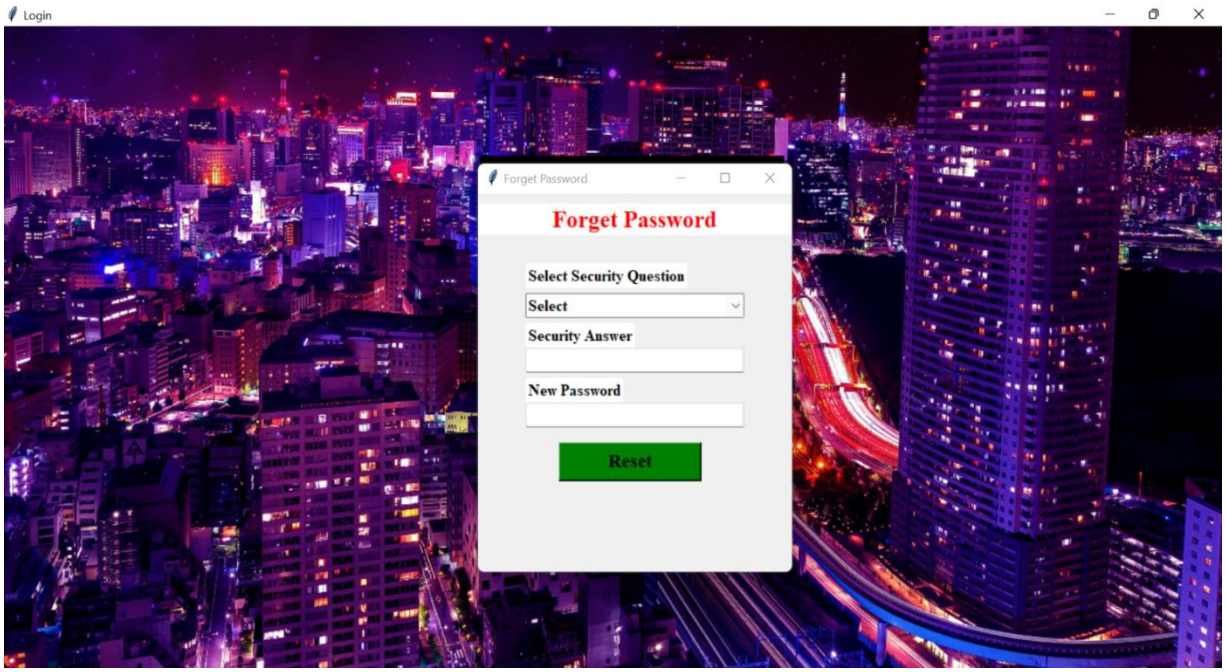
5.1 Register



5.2 Admin Login

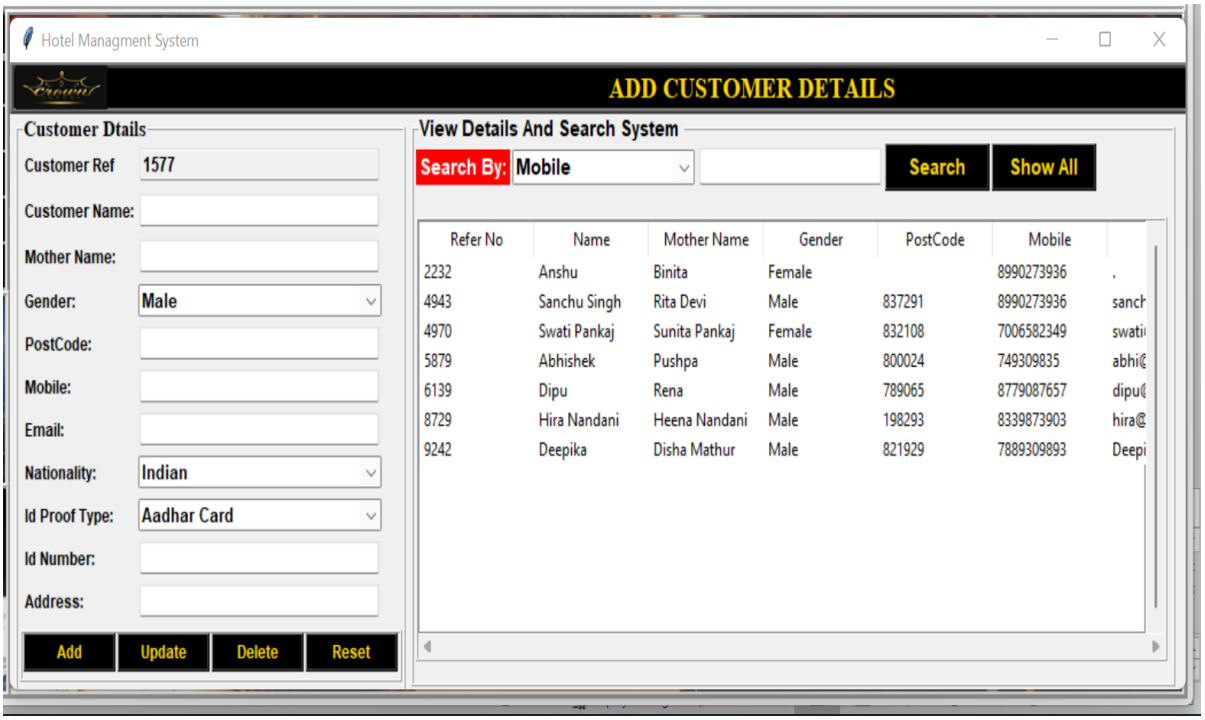


5.3 Forgot Password

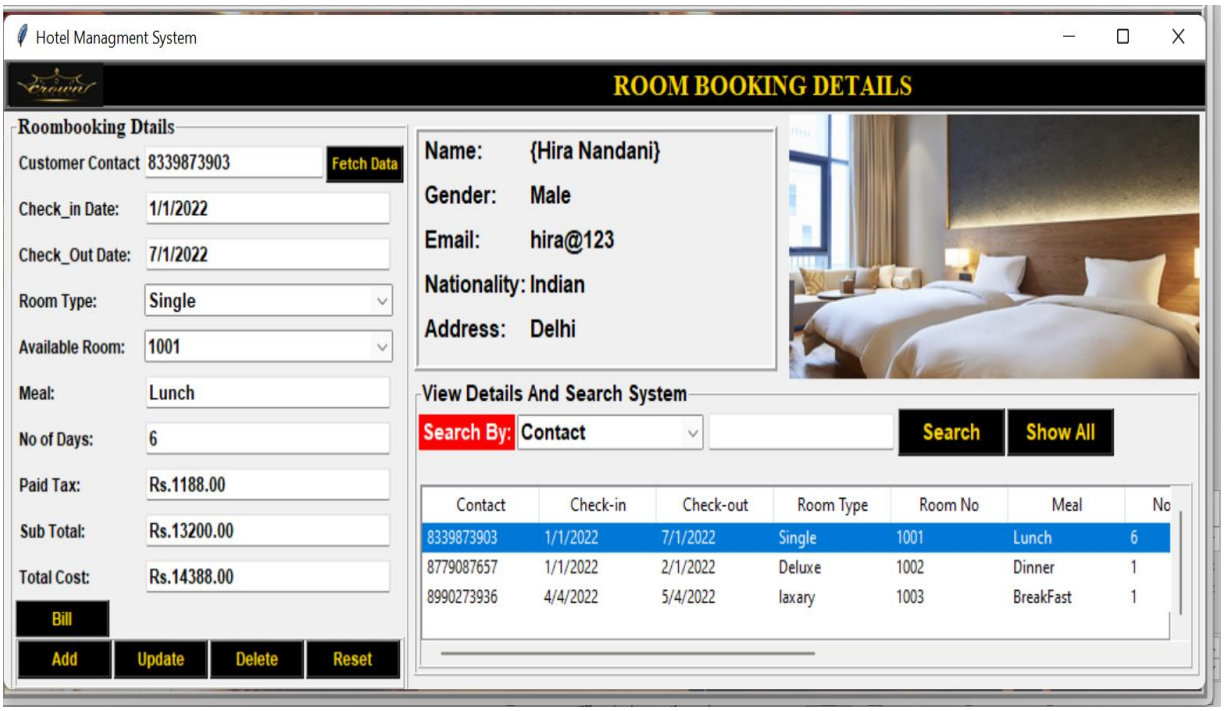


5.4 Home Page

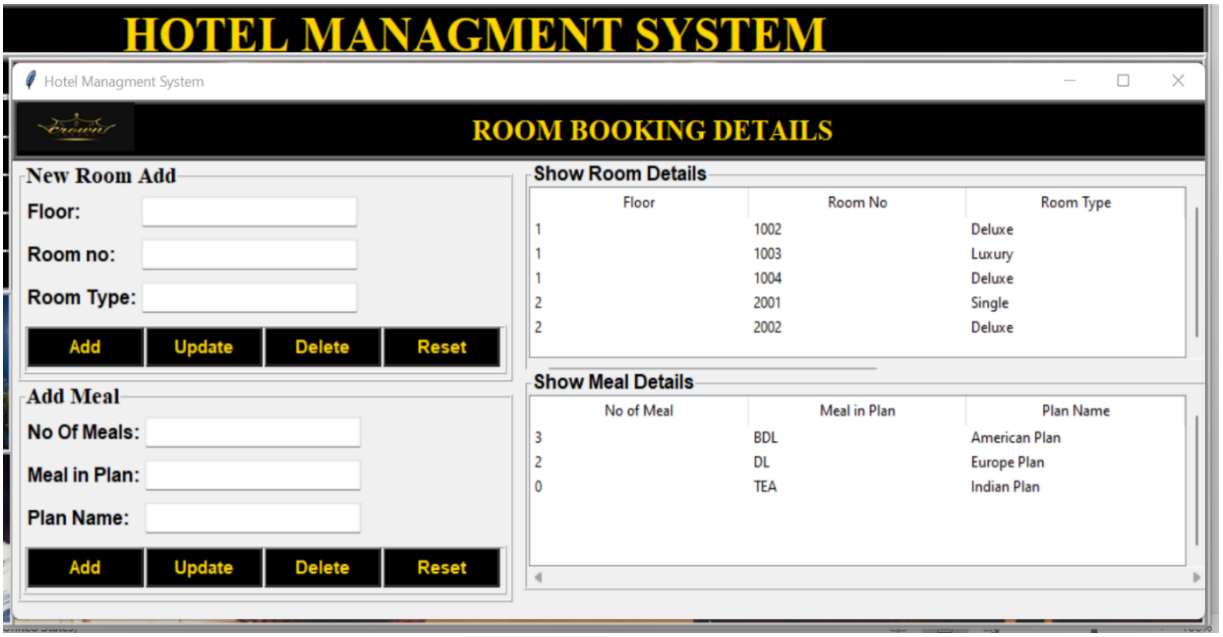
5.5 Add Customer



5.6 Room Allotment & Billing



5.7 Add/Update Room & Meal Details



5.8 Logout

CHAPTER 7
CONCLUSION

Conclusion

The “HOTEL MANAGEMENT SYSTEM" has been computed successfully and was also tested successfully by taking "Test Cases". It is user friendly, and has required options, which can be utilized by the user to perform the desired operations.

The Software is developed using Python as front end & back end and Microsoft SQL Server as the database in windows environment.

The goals that are achieved by the software are:

- Simplification of the operations
- Less processing time and getting required information
- User friendly
- Portable and flexible for further enhancement

Future scope of the project

Today, the market place is flooded with several Hotel Management options for hotels and their members to choose from. A variety of members are being offered worst services by the hotels. In the last couple of years, the growth of IT Industry has been phenomenal as more have started discovering the benefits of using this platform. Therefore, keeping in mind every requirement of a house member we built this system. In future, the UI can be made more intuitive for better user experience. And we will make this system live and provide Software as Service (SAS).

REFERENCES

The following reference has been used to develop the project “HOTEL MANAGEMENT SYSTEM”:

Books: -

- Python Programming: An Introduction to Computer Science.
- SQL QuickStart Guide: The Simplified Beginner’s Guide to Managing, Analyzing, and Manipulating Data With SQL

Web Source: -

- https://www.google.com
- https://www.youtube.com
- https://gighub.club/
- https:// stackoverflow.com
- https://geeksforgeeks.org/