

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**(ALLENHOUSE INSTITUTE OF TECHNOLOGY, KANPUR)**

**(Dr. A. P. J. Abdul Kalam Technical University, Lucknow, Uttar Pradesh)**

**MINI PROJECT REPORT**

**(KCS-554)**

**On**

**HOTEL MANAGEMENT SYSTEM**



**BACHELOR OF TECHNOLOGY**

**Computer Science and Engineering (AI&ML)**

**2023-24**

**SUBMITTED TO**

Dr. Pradeep Kumar  
(Mini Project Coordinator)

**SUBMITTED BY**

Pranshu Pal  
(2105051530075)

## **Table of Content**

• Title Page.....	1
• Table of Content.....	2
• Abstract.....	3
• Introduction.....	4
1. Feasibility Study.....	5
1.1. Technical Feasibility.....	5
1.2. Economic Feasibility.....	5
1.3. Organizational Feasibility.....	5
1.4. Operational Feasibility.....	5
2. Requirement Specification.....	6
2.1 Hardware Requirements.....	6
2.2. Software Requirements.....	6
3. Designing.....	7
3.1. Module Description.....	7
3.2. ER Diagram.....	8
3.3. Screenshots.....	9
4. Testing.....	24
4.1. Types of Testing.....	25
4.2. Testing Method.....	25
5. Maintenance.....	29
5.1. Implementation of Security.....	29
5.2. Database Security.....	30
6. Conclusion .....	32
7. Future Scope.....	33
8. References.....	34

## **ABSTRACT**

The Hotel Management System is a comprehensive software solution designed to streamline and automate various operations within a hotel, enhancing overall efficiency and guest satisfaction. This project addresses the complexities associated with managing different facets of hotel operations, including reservation management, room allocation, billing, inventory tracking, and reporting.

### **Key Features:**

The system allows for easy and efficient reservation handling, providing a user-friendly interface for both guests and hotel staff. It enables real-time room availability checks and ensures accurate booking information.

Automated room allocation based on guest preferences, availability, and other relevant factors helps optimize room utilization. The system also manages inventory for other hotel resources such as conferences rooms, banquet halls, and amenities.

The software simplifies the billing process by automating the calculation of charges based on room rates, additional services, and duration of stay. It generates accurate invoices and supports various payment methods, enhancing financial transparency.

A centralized database maintains guest profiles, storing preferences and historical data to personalize services. This facilitates a more tailored and enjoyable experience for returning guests.

Role-based access control ensures that only authorized personnel can access specific features. The system employs robust security measures to protect sensitive guest information and maintain data integrity.

Seamless integration with online booking platforms enables real-time updates on room availability and rates, expanding the hotel's reach and enhancing its online presence.

The system facilitates effective communication with guests, allowing the hotel to manage feedback, address concerns, and build lasting relationships.

# **INTRODUCTION**

## **❖ OBJECTIVE:**

- The main objective of the project is to design and develop a user friendly-system
- Easy to use and an efficient computerized system.
- To develop an accurate and flexible system, it will eliminate data redundancy.
- To study the functioning of Students management System.
- To make a software fast in processing, with good user interface.
- To make software with good user interface so that user can change it and it should be used for a long time without error and maintenance.
- To provide synchronized and centralized farmer and seller database.
- Computerization can be helpful as a means of saving time and money.
- To provide better Graphical User Interface (GUI).
- Less chances of information leakage.
- Provides Security to the data by using login and password method.
- To provide immediate storage and retrieval of data and information.
- Improving arrangements for students' coordination.
- Reducing paperwork.

## **❖ LIMITATIONS:**

- Time consumption in data entry as the records are to be manually maintained faculties a lot of time.
- Lot of paper work is involved as the records are maintained in the files and registers.
- Storage Requires as files and registers are used the storage space requirement is increased.
- Less Reliable use of papers for storing valuable data information is not at all reliable.
- Valid Id Proof linkage with the official Government Identity database has not been done.

# **FEASIBILITY STUDY**

Feasibility study is conducted to select the best system that meets the performance requirements. This entails an identification, description and evaluation of the candidate system, and the selection of the best system for the job, many feasibility studies are disillusioning for both user and analyst. First the study often pre-supposes that when feasibility of the documents is being prepared, the analyst is in position to evaluate solutions. Second most studies tend to overlook the confusion inherent in the system development the constraints and the assumed attitudes. If the feasibility study is to serve as decision documents, it must answer three key questions:

- Is there new and better way to do a job that will benefit the user?
- What are the cost and savings of the alternatives?
- What is recommended?

**Their main considerations are involved in feasibility analysis are:**

- **Technical Feasibility:** It refers to whether the software that is available in the market fully supports the present application. It studies the pros and cons of using particular software for the development and its feasibility. It also studies the additional training needed to be given to the people to make the application work.
- **Economic Feasibility:** It refers to the benefits or outcomes we deriving from the product compared to the total cost we are spending for developing the product.  
In the present system , the development of the new product greatly enhances the accuracy of the system and cuts short the delay in the processing of application.
- **Organizational Feasibility:** This involves questions such as whether the system has enough support to be implemented successfully, whether it brings an excessive amount of change, and whether the organization is changing too rapidly to absorb it.
- **Operational Feasibility:** It refers to the feasibility of the product to be operational. Some products may work very well at design and implementation but may fail in the real time environment. It includes the study of additional human resource required and their technical expertise. In the present system, all the operations can be performed easily compared to existing system and supports for the backlog data.

# **REQUIREMENT SPECIFICATION**

## ▪ **Hardware Specifications:**

- Computer with a 1.1 GHz or faster processor
- Minimum 2GB of RAM or more
- 2.5 GB of available hard-disk space
- 5400 RPM hard drive
- 1366 × 768 or higher-resolution display
- DVD-ROM drive

## ▪ **Software Specifications:**

- Operating System: Windows 10
- Google Chrome/Internet Explorer
- XAMPP (Version-3.7)
- Python main editor (user interface): PyCharm Community
- workspace editor: Sublime text 3

# **DESIGNING**

## **Modules Description**

This module has following module:

- Administrator Module
- Customer Module

### **❖ Administrator Module**

- In administrator module admin manage the bookings made.
- Administrator can see the status of the booking.
- Administrator can allow only authorized genuine customer or staff is added by admin itself.
- Administrator can manage the staff and customer.
- Administrator can check customer and staff's session logs.
- Administrator can change the password anytime.
- Administrator can see the feedbacks send by the visitors of the site.

### **❖ Customer Module:**

- To book the room according to their requirements.
- After successful registration user can login to proceed ahead.
- User can book the room and food according to their requirement on the specific date and time.
- User can search and booked multiple rooms and foods.

### **➤ ER Diagram:**

An Entity-Relationship (ER) diagram is a visual representation of the data model that depicts the entities, attributes, relationships, and cardinality constraints within a system. It is commonly used in database design to illustrate the logical structure of a database.

Here's a brief explanation of the key components typically found in an ER diagram:

- **Entities:**  
Represented by rectangles.

Each entity corresponds to a table in the database.

Examples: "Customer," "Room," "Reservation."

- **Attributes:**

Represented by ovals and connected to entities.

Describes the properties or characteristics of entities.

Examples: "CustomerID" as an attribute of the "Customer" entity.

- **Relationships:**

Represented by diamond shapes and connected to related entities.

Describe how entities are related to each other.

Examples: "Books" relationship between "Customer" and "Room" entities indicating a reservation.

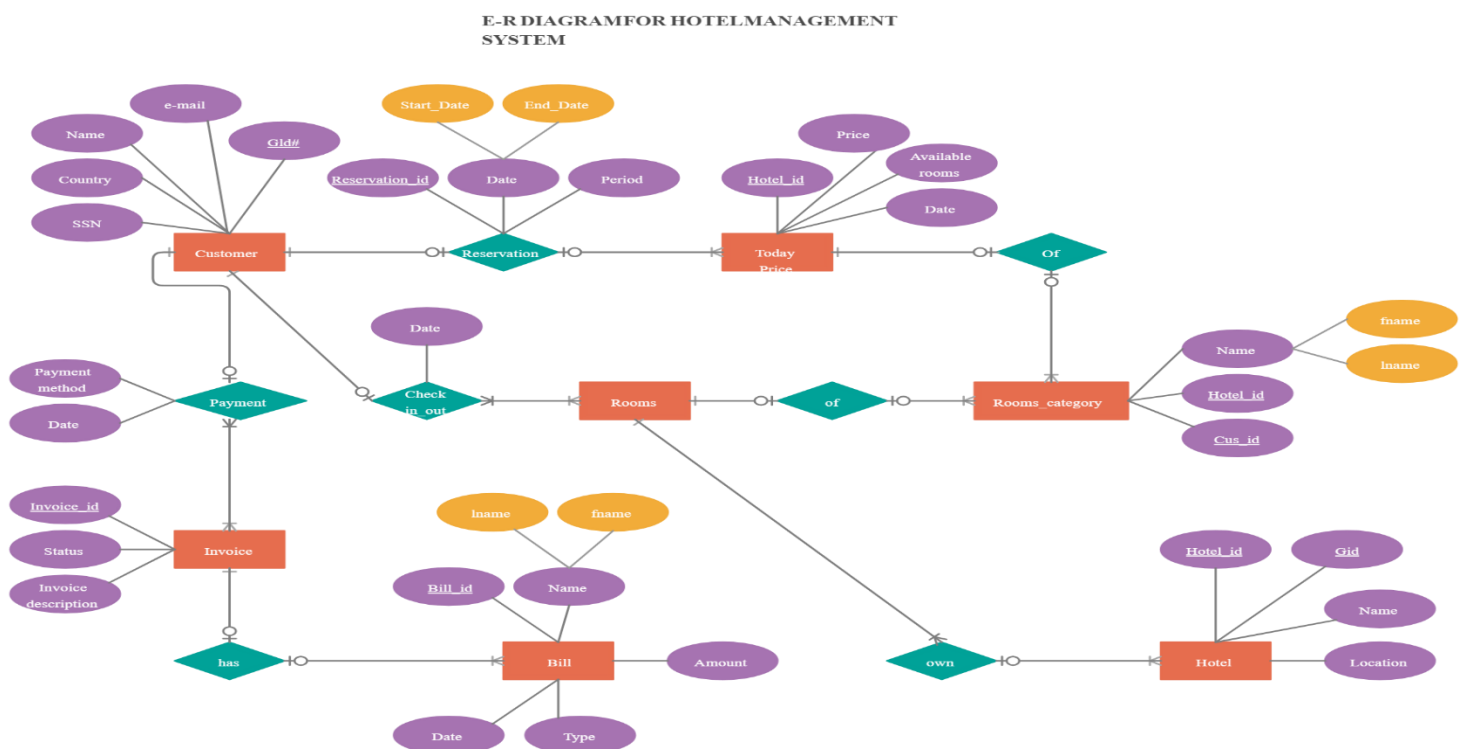
- **Cardinality:**

Indicates the maximum number of instances of one entity that can be associated with another entity.

Represented by lines or crow's feet near the relationship lines.

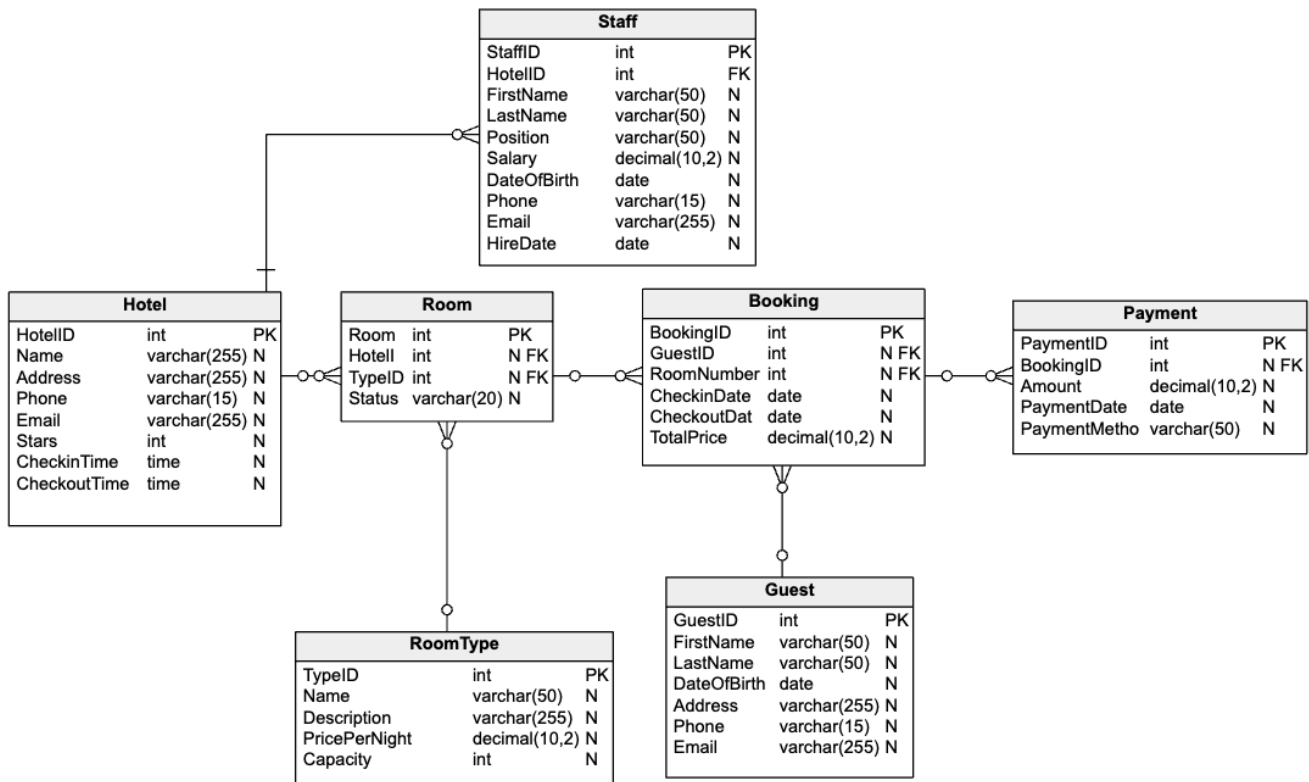
Examples: "0..1" indicating zero or one, "0..n" indicating zero to many.

## **ER Diagram:**





## Schema Diagram:



## Screenshots of Implementation:

```
from tkinter import*
from tkinter import ttk
from PIL import Image,ImageTk
from customer import Customer_Window
from room import Roombooking
from details import DetailsRoom
from report import Report
from tkinter import messagebox
import mysql.connector

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("1550x800+0+0")

self.bg=ImageTk.PhotoImage(file=r"D:\Attenance_Login\hotel images\SDT_Zoom-Back-
grounds_April-8_Windansea-1-logo-1.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=610,y=170,width=340,height=450)

img1=Image.open(r"D:\Attenance_Login\hotel images\LoginIconAppl.png")
img1=img1.resize((100,100),Image.LANCZOS)
self.photoimage1=ImageTk.PhotoImage(img1)
lblimg1=Label(image=self.photoimage1,bg="black",borderwidth=0)
lblimg1.place(x=730,y=175,width=100,height=100)

get_str=Label(frame,text="Get Started",font=("times new ro-
man",20,"bold"),fg="white",bg="black")
get_str.place(x=95,y=100)

#Label
username=lbl=Label(frame,text="Username",font=("times new ro-
man",15,"bold"),fg="white",bg="black")
username.place(x=70,y=155)

self.txtuser=ttk.Entry(frame,font=("times new roman",15,"bold"))
self.txtuser.place(x=40,y=180,width=270)

password=lbl=Label(frame,text="Password",font=("times new ro-
man",15,"bold"),fg="white",bg="black")
password.place(x=70,y=225)

self.txtpass=ttk.Entry(frame,font=("times new roman",15,"bold"))
self.txtpass.place(x=40,y=250,width=270)

#=====Icon Image=====
img2=Image.open(r"D:\Attenance_Login\hotel images\LoginIconAppl.png")
img2=img2.resize((25,25),Image.LANCZOS)
self.photoimage2=ImageTk.PhotoImage(img2)
lblimg1=Label(image=self.photoimage2,bg="black",borderwidth=0)
lblimg1.place(x=650,y=323,width=25,height=25)

img3=Image.open(r"D:\Attenance_Login\hotel images\lock-512.png")
img3=img3.resize((25,25),Image.LANCZOS)
self.photoimage3=ImageTk.PhotoImage(img3)
lblimg1=Label(image=self.photoimage3,bg="black",borderwidth=0)
lblimg1.place(x=650,y=395,width=25,height=25)

```

```

        #Login Buttonloginbtn=Button(frame,com-
mand=self.login,text="Login",font=("times new roman",15,"bold"),bd=3,rel-
ief=RIDGE,fg="white",bg="red",activeforeground="white",activeback-
ground="red")

        loginbtn.place(x=110,y=300,width=120,height=35)

        #Registration Button
        registerbtn=Button(frame,text="New User Register",command=self.regis-
ter_window,font=("times new roman",10,"bold"),border-
width=0,fg="white",bg="black",activeforeground="white",activebackground="black")
        registerbtn.place(x=15,y=350,width=160)

        #forgetpassbtn
        registerbtn=Button(frame,text="Forget Password",command=self.forgot_pass-
word_window,font=("times new roman",10,"bold"),border-
width=0,fg="white",bg="black",activeforeground="white",activebackground="black")
        registerbtn.place(x=10,y=370,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 fields are required")
        elif self.txtuser.get() == "kapu" and self.txtpass.get() == "ashu":
            messagebox.showinfo("Success", "The Great Grand Hotel")
        else:
            conn = mysql.connector.connect(host="localhost", user="root", pass-
word="Pranshu@9918", database="mydata")
            my_cursor = conn.cursor()
            my_cursor.execute("select * from registration where email=%s and pass-
word=%s", (self.txtuser.get(), self.txtpass.get()))
            row = my_cursor.fetchone()
            if row is 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",par-
ent=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", user="root", password="Pranshu@9918", database="mydata")
            my_cursor = conn.cursor()
            query=("select * from registration 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(query,value)
            row=my_cursor.fetchone()
            if row==None:
                messagebox.showerror("Error","Please enter the correct Answer",parent=self.root2)
            else:
                query=("update registration 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 to reset password")
    else:
        conn = mysql.connector.connect(host="localhost", user="root", password="Pranshu@9918", database="mydata")
        my_cursor = conn.cursor()
        query=("select * from registration where email=%s")
        value=(self.txtuser.get(),)
        my_cursor.execute(query,value)
        row=my_cursor.fetchone()
        #print(row)

        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("340x450+610+170")

        l=Label(self.root2,text="Forget Password",font=("times new ro
man",20,"bold"),fg="red",bg="white")
        l.place(x=0,y=10,relwidth=1)

        security_Q=Label(self.root2,text="Select Security Ques-
tions",font=("times new roman",15,"bold"),bg="white",fg="black")
        security_Q.place(x=50,y=80)

        self.combo_security_Q=ttk.Combobox(self.root2,font=("times new ro
man",15,"bold"),state="readonly")
        self.combo_security_Q['values']=("Select","Your Birth Place","Your
Favourite Person","Your Pet Name")
        self.combo_security_Q.place(x=50,y=110,width=250)
        self.combo_security_Q.current(0)

        security_A=Label(self.root2,text="Security Answer",font=("times
new roman",15,"bold"),bg="white",fg="black")
        security_A.place(x=50,y=150)

        self.txt_security=ttk.Entry(self.root2,font=("times new ro
man",15))
        self.txt_security.place(x=50,y=180,width=250)

        new_password=Label(self.root2,text="New Password",font=("times new
roman",15,"bold"),bg="white",fg="black")
        new_password.place(x=50,y=220)

        self.txt_newpass=ttk.Entry(self.root2,font=("times new roman",15))
        self.txt_newpass.place(x=50,y=250,width=250)

        btn=Button(self.root2,text="Reset",command=self.re-
set_pass,font=("times new roman",15,"bold"),fg="white",bg="green")
        btn.place(x=100,y=290)

class Register:
    def __init__(self,root):
        self.root=root
        self.root.title("Register")
        self.root.geometry("1600x900+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=StringVar()

#Background Image
self.bg=ImageTk.PhotoImage(file=r"D:\Attenance_Login\hotel images\0-
3450_3d-nature-wallpaper-hd-1080p-free-download-new.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"D:\Attenance_Login\hotel im-
ages\thought-good-morning-messages-LoveSove.jpg")
left_lbl=Label(self.root,image=self.bg1)
left_lbl.place(x=50,y=100,width=470,height=550)
#mainframe
frame=Frame(self.root,bg="white")
frame.place(x=520,y=100,width=800,height=550)

register_lbl=Label(frame,text="Register Here",font=("times new ro-
man",20,"bold"),fg="darkgreen",bg="white")
register_lbl.place(x=20,y=20)

#Labels and entry
fname_label = Label(frame, text="First Name", font=("times new roman", 15,
"bold"), bg="white", fg="black")
fname_label.place(x=50, y=100)

fname_entry = ttk.Entry(frame,textvariable=self.var_fname, font=("times
new roman", 15, "bold"))
fname_entry.place(x=50, y=130, width=250)

l_name_label = Label(frame, text="Last Name", font=("times new roman", 15,
"bold"), bg="white", fg="black")
l_name_label.place(x=370, y=100)

self.txt_lname=ttk.Entry(frame,textvariable=self.var_lname,font=("times
new roman",15))
self.txt_lname.place(x=370,y=130,width=250)

#row2
contact=Label(frame,text="Contact No",font=("times new ro-
man",15,"bold"),bg="white",fg="black")
contact.place(x=50,y=170)

self.txt_contact=ttk.Entry(frame,textvariable=self.var_con-
tact,font=("times new roman",15))
self.txt_contact.place(x=50,y=200,width=250)

email=Label(frame,text="Email",font=("times new ro-
man",15,"bold"),bg="white",fg="black")
email.place(x=370,y=170)

```

```

        self.txt_email=ttk.Entry(frame,textvariable=self.var_email,font=("times
new roman",15))
        self.txt_email.place(x=370,y=200,width=250)

        #row3
        security_Q=Label(frame,text="Select Security Questions",font=("times new
roman",15,"bold"),bg="white",fg="black")
        security_Q.place(x=50,y=240)

        self.combo_security_Q=ttk.Combobox(frame,textvariable=self.var_securi-
tyQ,font=("times new roman",15,"bold"),state="readonly")
        self.combo_security_Q['values']=("Select","Your Birth Place","Your Favour-
ite Person","Your Pet Name")
        self.combo_security_Q.place(x=50,y=270,width=250)
        self.combo_security_Q.current(0)

        security_A=Label(frame,text="Security Answer",font=("times new ro-
man",15,"bold"),bg="white",fg="black")
        security_A.place(x=370,y=240)

        self.txt_security=ttk.Entry(frame,textvariable=self.var_Secu-
rityA,font=("times new roman",15))
        self.txt_security.place(x=370,y=270,width=250)

        #row4
        pswd=Label(frame,text="Password",font=("times new ro-
man",15,"bold"),bg="white",fg="black")
        pswd.place(x=50,y=310)

        self.txt_pswd=ttk.Entry(frame,textvariable=self.var_pass,font=("times new
roman",15))
        self.txt_pswd.place(x=50,y=340,width=250)

        confirm_pswd=Label(frame,text="Confirm Password",font=("times new ro-
man",15,"bold"),bg="white",fg="black")
        confirm_pswd.place(x=370,y=310)

        self.txt_confirm_pswd=ttk.Entry(frame,textvaria-
ble=self.var_confpass,font=("times new roman",15))
        self.txt_confirm_pswd.place(x=370,y=340,width=250)

        #checkboxbutton
        self.var_check=IntVar()
        checkbtn=Checkbutton(frame,variable=self.var_check,text="I Agree The Terms
& Conditions",font=("times new roman",12,"bold"),onvalue=1,offvalue=0)
        checkbtn.place(x=50,y=380)

        #Buttons
        img=Image.open(r"D:\Attenance_Login\hotel images\register-now-but-
ton1.jpg")
        img=img.resize((200,50),Image.LANCZOS)

```

```

        self.photoimage = ImageTk.PhotoImage(image=img)
        b1=Button(frame,image=self.photoimage,command=self.register_data,border-
width=0,cursor="hand2")
        b1.place(x=10,y=420,width=200)

        img1=Image.open(r"D:\Attenance_Login\hotel images\loginpng.png")
        img1=img1.resize((200,50),Image.LANCZOS)
        self.photoimage1 = ImageTk.PhotoImage(image=img1)
        b1=Button(frame,image=self.photoimage1,command=self.return_login,border-
width=0,cursor="hand2")
        b1.place(x=330,y=420,width=200)

    #function declaration
    def register_data(self):
        if self.var_fname.get()==" " or self.var_email.get()==" " or self.var_secu-
rityQ.get()=="Select":
            messagebox.showerror("Error","All fields are required")
        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",user="root",pass-
word="Pranshu@9918",database="mydata")
            my_cursor=conn.cursor()
            query=("select * from registration 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 exists,please try an-
other email")
            else:
                my_cursor.execute("insert into registration val-
ues(%s,%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!!!")

#Login
def return_login(self):
    self.root.destroy()

#manage.py code
class HotelManagementSystem:
    def __init__(self, root):
        self.root = root
        self.root.title("The Great Grand Hotel")
        self.root.geometry("1550x800+0+0")

        #=====First Image=====

        img1 = Image.open(r"D:\Attenance_Login\hotel images\hotel1.png")
        img1 = img1.resize((1550, 140),Image.LANCZOS)
        self.photoimg1 = ImageTk.PhotoImage(img1)

        lblimg = Label(self.root, image=self.photoimg1, bd=4, relief=RIDGE)
        lblimg.place(x=0, y=0, width=1550, height=140)

        #=====Logo=====

        img2 = Image.open(r"D:\Attenance_Login\hotel images\logohotel.png")
        img2 = img2.resize((230, 140),Image.LANCZOS)
        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="THE GREAT GRAND HOTEL" ,font=("TIMES NEW ROMAN",40,"bold"),bg="black",fg="gold",bd=4,relief=RIDGE)
        lbl_title.place(x=0,y=140,width=1550,height=50)

        #=====main frame=====
        main_frame=Frame(self.root,bd=4,relief=RIDGE)
        main_frame.place(x=0,y=190,width=1550,height=620)

        #=====Menu=====
        lbl_menu=Label(main_frame,text="MENU" ,font=("TIMES NEW ROMAN",20,"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=35,width=228,height=190)

cust_btn=Button(btn_frame,text="CUSTOMER",command=self.cust_details,width=22,font=("TIMES NEW ROMAN",14,"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=22,font=("TIMES NEW ROMAN",14,"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=22,font=("TIMES NEW ROMAN",14,"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",command=self.developer_report,width=22,font=("TIMES NEW ROMAN",14,"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=22,font=("TIMES NEW ROMAN",14,"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"D:\Attenance_Login\hotel images\slide3.jpg")
img3= img3.resize((1310, 590),Image.LANCZOS)
self.photoimg3 = ImageTk.PhotoImage(img3)

lblimg1 = Label(main_frame, image=self.photoimg3, bd=4, relief=RIDGE)
lblimg1.place(x=225, y=0, width=1300, height=590)

#=====Down Image=====
img4 = Image.open(r"D:\Attenance_Login\hotel images\myh.jpg")
img4= img4.resize((230, 210),Image.LANCZOS)
self.photoimg4 = ImageTk.PhotoImage(img4)

lblimg1 = Label(main_frame, image=self.photoimg4, bd=4, relief=RIDGE)
lblimg1.place(x=0, y=225, width=230, height=210)

img5 = Image.open(r"D:\Attenance_Login\hotel images\khana.jpg")
img5= img5.resize((230, 190),Image.LANCZOS)
self.photoimg5 = ImageTk.PhotoImage(img5)

lblimg1 = Label(main_frame, image=self.photoimg5, bd=4, relief=RIDGE)
lblimg1.place(x=0, y=420, width=230, height=190)

```

```

def cust_details(self):
    self.new_window=Toplevel(self.root)
    self.app=Customer_Window(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=DetailsRoom(self.new_window)

def developer_report(self):
    self.new_window=Toplevel(self.root)
    self.app=Report(self.new_window)

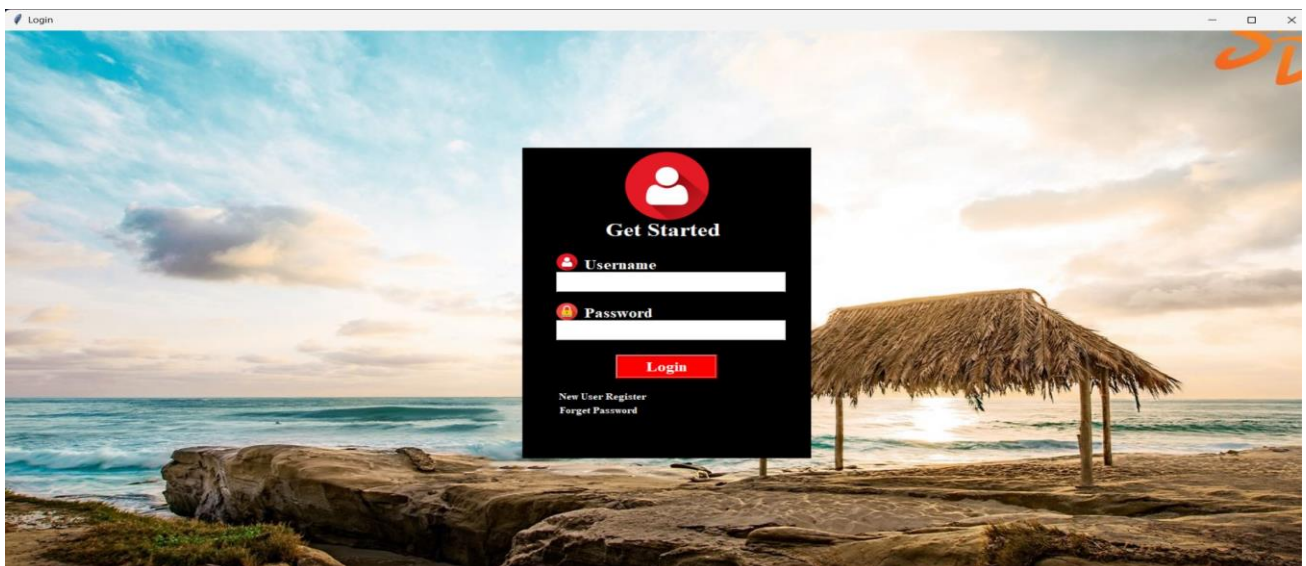
def logout(self):
    self.root.destroy()

if __name__=="__main__":
    main()

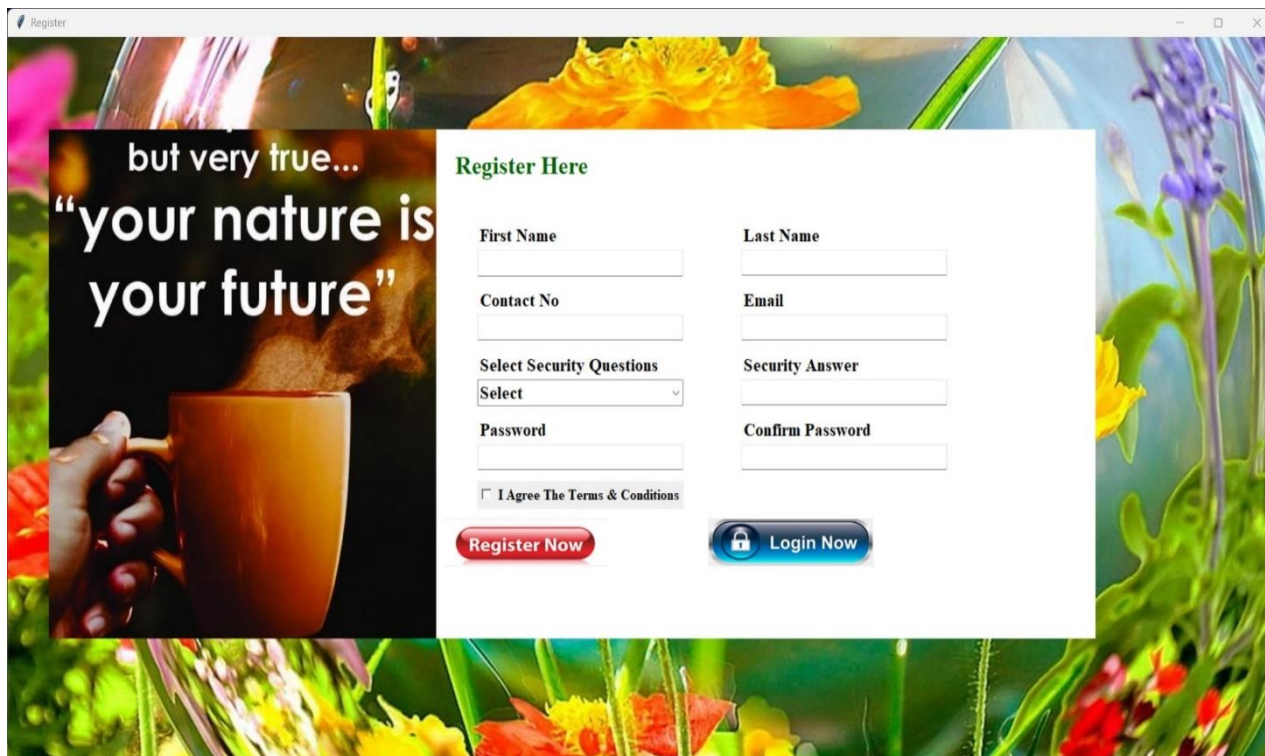
```

## Output:

### ▪ Login Window:



- **New User Window:**



The Register window features a vibrant background of various flowers. On the left, a dark rectangular box contains the text "but very true..." in a small font, followed by "your nature is your future" in a large, bold, white font. A hand holding a steaming mug is visible at the bottom left of this box. The main registration form is titled "Register Here" in green. It includes input fields for First Name, Last Name, Contact No, Email, Password, and Confirm Password. A dropdown menu for "Select Security Questions" is set to "Select". Below the fields is a checkbox for "I Agree The Terms & Conditions". At the bottom, there are two buttons: a red "Register Now" button and a blue "Login Now" button with a lock icon.

Register Here

First Name

Last Name

Contact No

Email

Select Security Questions  
Select

Password

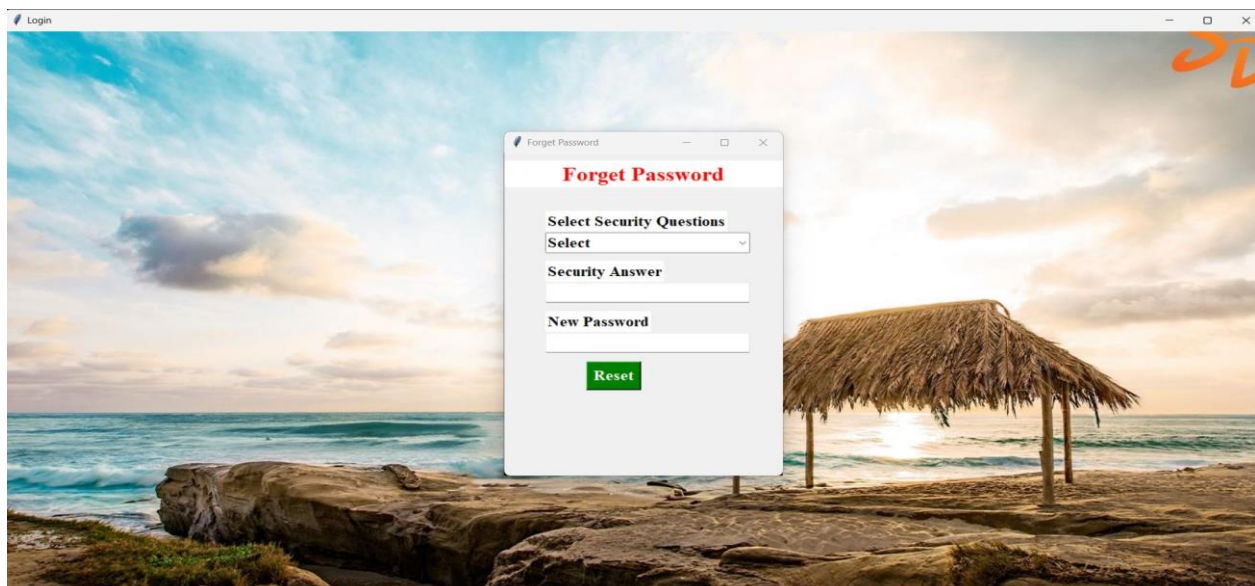
Security Answer

Confirm Password

☐ I Agree The Terms & Conditions

Register Now Login Now

- **Forget Window**



The Forget Password window is displayed over a scenic background of a beach at sunset, with waves crashing against rocks and a thatched hut on the shore. The window is titled "Forget Password" in red. It contains a dropdown menu for "Select Security Questions" set to "Select", a text field for "Security Answer", and a text field for "New Password". A green "Reset" button is located at the bottom of the form.

Forget Password

Select Security Questions  
Select

Security Answer

New Password

Reset



## ■ Home Window:



## ■ Customer Window:

The Great Grand Hotel

# THE GREAT GRAND HOTEL

**MENU**

**CUSTOMER**

**ROOM**

**DETAILS**

**REPORT**

**LOGOUT**

**ADD CUSTOMER DETAILS**

**Customer Details**

Customer Ref: 4978

Customer Name: Raman

Mother Name: Roohi

Gender: Male

PostCode: 202575

Mobile: 2005254190

Email: raman@gmail.com

Nationality: Indian

Id Proof Type: AadharCard

Id Number: 44464687864332

Address: Uttar Pradesh

**Add Update Delete Reset**

**View Details And Search System**



Search By: Mobile

**Search Show All**

Refer No	Name	Mother Name	Gender	PostCode	Mobile	Email	Nationality	Id
3044	Divesh	Aliza	Male	647	9144219	dives@gmail.co	Indian	Aadha
4978	Raman	Roohi	Male	202575	2005254190	raman@gmail.co	Indian	Aadha
5722	Alisa	Aliza	Female	46498	654963169	alisa05@gmail.co	British	Passpc
7395	Navya	Nisha	Female	45782	1234567809	navya@gmail.co	Indian	Passpc



## ▪ RoomBooking Window:

The Great Grand Hotel

# THE GREAT GRAND HOTEL

**MENU**  
**CUSTOMER**  
**ROOM**  
**DETAILS**  
**REPORT**  
**LOGOUT**

The Hospital Room

## ROOMBOOKING DETAILS

RoomBooking Details

Customer Contact: 9632587410 [Fetch Data](#)

Check\_in Date: 18/11/2023

Check\_Out Date: 30/12/2023

Room Type: Duplex+

Available Room: 3001

Meal: Breakfast

No Of Days: 42


Paid Tax: Rs.154.20

Sub Total: Rs.1542.00

Total Cost: Rs.1696.20

[Bill](#)

[Add](#) [Update](#) [Delete](#) [Reset](#)



View Details And Search System



Search By: Contact  [Search](#) [Show All](#)

Contact	Check-in	Room Type	Check-out	Room No	Meal	NoOfDays
1234567809	19/11/2023	25/11/2023	Single	1001	Lunch	6
9140200	01/05/2025	08/05/2025	luxury	115	Dinner	7
9144219	19/11/2023	25/11/2023	Single	2001	Lunch	6
9632587410	18/11/2023	30/12/2023	Duplex+	3001	Breakfast	42

[Show desktop](#)



## ▪ RoomDetails Window:

The Great Grand Hotel

# THE GREAT GRAND HOTEL

**MENU**  
**CUSTOMER**  
**ROOM**  
**DETAILS**  
**REPORT**  
**LOGOUT**

The Hospital Room

## ROOMBOOKING DETAILS

New Room Add

Floor: 1

Room No: 1002

Room Type: Double

[Add](#) [Update](#) [Delete](#) [Reset](#)

Show Room Details

Floor	Room No	Room Type
1	1001	Luxury
1	1002	Double
1	1003	Single
2	2001	Single
2	2002	Luxury
2	2003	Double
3	3001	Duplex+

Great Learning Online Courses on the App Store

## ▪ Application Details Window:





# **TESTING**

Software testing is a critical element of software quality assurance and represent the ultimate review of specification design, coding, purpose of product testing is to verify and validate various work products via unit integrated unit, final product to ensure that they meet their requirements.

## **Testing Objectives:**

Basically, testing is done for the following purposes.

- Testing is a process of executing program with the intent of finding an error.
- A good test case is one that has a high probability of finding an as yet undiscovered error.
- Successful test case is one that uncovers an as yet undiscovered error.

Our objective is to design test cases that systematically uncover different classes of error and do so with a minimum amount of time and effort. This process has two parts:

- **Planning:**

This involves writing and reviewing unit integration, functional, validation and acceptance test plans.

- **Execution:**

This involves executing these test plans, measuring Collecting data and very fine if it meets the quality criteria. Data collected is used to make appropriate changes in the plans related to development and testing. The quality of a product or item can be achieved by ensuring that the product meets the requirements by planning and conducting the following tests at various stages.

## **4.1. Unit Testing:**

Unit testing is essentially for the verification of the code produced during the coding phase and the goal is test the internal logic of the module/program. Int the Generic code project, the unit testing is done during coding phase of data entry forms whether the functions are working properly or not. In this phase all the drivers are tested they are rightly connected or not.

All the tested modules are combined into subsystems, which are then tested. The goal is to see if the modules are properly integrated, and the emphasis being on the testing interfaces between the modules. The generic code integration testing is done mainly on table creation module and insertion module.



#### **4.1.1. Types of testing software:**

The main types of software testing are:

##### **1. Component Testing:**

Starting from the bottom the first test level is 'Component Testing', sometimes called Unit testing. It involves checking that each feature specifies in the 'Component Design' has been implemented in the component. In theory an independent tester should do this, but in practice the developer usually does it, as they are the only people who understand how a component works. The problem with a component is that it performs only a small part of the functionality of a system, and it relies on co-operating with other parts of the system, which may not have been built yet. To overcome this, the developer either builds, or uses special software to trick the component into believing it is working in a fully functional system.

##### **2. Interface Testing:**

As the components are constructed and tested they are then linked together to check if they work with each other. It is fact that two components that have passed all their tests, when connected to each other produce one new component full of faults. These tests can be done by specialists, or by the developers.

- 1). what a component can expect from another component in terms of services.
- 2). How these services will be asked for.
- 3). How they will be given.
- 4). How to handle nonstandard conditions, i.e. errors.

##### **3. Release Testing:**

Even if a system meets all its requirements, there is still a case to be answered that it will benefit the business.

- 1). Does it affect any other system running on the hardware?
- 2). is it compatible with other system?
- 3). Does it have acceptable performance under load?

These tests are usually run by the computer operations team in a business it would appear obvious that the operation of a new system may have.

#### **4.1.2. Types of testing techniques:**

- White box Testing
- Black Box Testing

#### **4.1.2.1 White box testing:**

White box test focus on the program control structure Test cases are derived to ensure that all statement in the program has been executed at least once during testing and that all logical condition has been exercised. Basic path testing, a white box testing, makes use of program graph to derive the set of linearly independent test that will ensure coverage.

#### **Conditional testing:**

Condition testing is tests case design method that exercises the logical condition in a program module. A simple condition is a Boolean variable or a relational expression.

#### **Data flow testing:**

We have used data flow testing due to check the path of program according to the locations of definitions and uses of variables in the program.

#### **4.1.2.2. Black box testing:**

Black box testing focuses on the functional requirements of the software. That is, black-box testing enable the software engineer to derive set of input conditions that will fully exercise all functional requirements for a program.

#### **Graph-base testing method:**

We have used graph-based testing method errors associated with relationships. The first step is to understand the objects that are modeled in software in software and the relationship that connect these objects.

#### **Equivalence partitioning:**

This testing is used for the following reason:

- Specific numeric values
- Range of values
- Set of related values
- Boolean condition

#### **Boundary value analysis:**

Boundary value analysis is a test case design techniques that complements equivalence partitioning. Rather than selecting any element of equivalence class the selection of test cases at the edges of the class. Rather than focusing solely on the input condition.

The point of equivalence partitioning as:

- An input condition specifies a range boundary by values a and b, test cases should be design with

values, test cases should be design with values a and b and just below a and b.

- An input condition specifies a number of values, test cases should be developed that exercise the minimum and maximum number.

#### **Acceptance testing:**

When the function test was completed, we involved the user to make sure that the system worked according to the user's expectation. Thus the user did the final acceptance test.

#### **4.2. Integration Testing:**

Testing can be done in two ways:

1. Bottom up approach
2. Top down approach

##### **1. Bottom up approach:**

Testing can be performed starting from smallest and lowest level modules and proceeding one at a time. For each module in bottom up testing a short program executes the module and provides the needed data so that the module is asked to perform the way it will when embedded within the larger system. When bottom level modules are tested attention turns to those on the next level that use the lower level ones they are tested individually and then linked with the previously examined lower level modules.

##### **2. Top-down approach:**

This type of testing starts from upper-level modules, since the detailed activities usually performed in the lower-level routines are not provided stubs are written. A stub is a module shell called by upper-level module and that when reached properly will return a message to the calling module indicating that proper interaction occurred. No attempt is made to verify the correctness of the lower level module.

#### **4.3. System Testing:**

Once the entire system has been built then it has to be tested against the 'System Specification' to check if it delivers the features required. It is still developer focused, although specialist developers known as system testers are normally employed to do it. In essence System Testing is not about checking the individual parts of the design, but checking the system as a whole. In effect it is one giant component.

System testing can involve isn't of specialist types to see if all the functional and non-functional requirements these may include the following types of testing for the non-functional requirements.

- 1). Performance- Are the performance criteria met?

- 2). Volume- Can large volumes of information be handled?
- 3). Stress- Can speak volumes of information be handled?
- 4). Documentation- Is the documentation usable for the system?
- 5). Robustness- Does the system remain stable under adverse circumstances?

There are many others, the needs for which are dictated by how the system is supposed to perform.

## **MAINTENANCE**

Software maintenance is the modification of a software product after delivery to correct faults, to improve performance or other attributes, or to adapt the product to a modified environment. System maintenance denotes any changes made to a software product after it has been delivered to the customer. Maintenance is inevitable for almost any kind of product. However, most products need maintenance due to the wear and tear caused by use. On the other hand, software products do not need maintenance on this count, but need maintenance to correct errors, enhance features, port to new platforms, etc.

Software maintenance is becoming an important activity of large number of organizations. This is no surprise, given the rate of hardware obsolescence, the immortality of a software product per se, and the demand of the user community to see the existing software products run on newer platforms, run in newer environments, and/or with enhanced features. When the hardware platform changes and a software product perform some low-level functions, maintenance is necessary. Also, whenever the support environment of a software changes, the software product requires re-work to cope with the newer interface. Thus, every software product continues to evolve after its development through maintenance efforts.

Maintenance covers a wide range of activities including correcting, coding and designing errors, updating documentation and test data and upgrading user support. Enhancement means adding, modifying or re-developing the code to support changes in the specifications. It is necessary to keep up with the changing user needs and the operational environment. The software is designed with the view of easy updating to the software.

Future advancements can be done easily through the review of product. The software architecture is strong enough to provide enhancement in functionality, performance and reliability. Functions in the software are designed in such a way that they dynamically update on addition of new and revised modules. To append new features in this software, the databases for student details, examination details are sufficient enough for the task. Thus the software is capable enough of being enhanced easily whenever requires by the user.

### **5.1. Implementation of Security:**

#### **User access control:**

Security comprises user authentication, implying user identification (against defined users) and user's authorization indicating actions and permissions within the application. The application will incorporate security for all the modules in a hierarchal form. These levels are required to be identified by the application owners.

- The users will be classified in groups and each group will be provided access to identify screens and reports. This access will be across different modules.
- The group authorization right will also define whether they are permitted to view the information only or can modify it as well.

#### **User authentication:**

Every registered user will have a login ID and password. The system administrator, using the functionality provided by login control module, will create new login IDs and maintain existing IDs using the user registration screen. Whenever a new user is registered, a new login ID and password will be created. Users can change their own passwords. The passwords will not be displayed on the screen while they are being entered into the applications.

#### **User authorization:**

For data entry and modification, permissions will be defined for the entry screens. Administrators and Users both will be having different screens and different access rights. Administrator would be able to access all the features available on the website.

#### **User session management:**

The USERNAME of the user will be displayed on the next page, and any item needed on the different page will also be maintained using Session management. Browser Session will expire and terminate user session after a certain period of inactivity. The following data will be maintained throughout an active session.

#### **Exception handling:**

Validation will be enforced at the UI as well as the business layer of application. If errors occur, appropriate messages will be displayed to the user for taking necessary action. These errors will be handled by the application using Visual Studio Exception Handling.

### **5.2. Database Security:**

In this particular software our back end used is MYSQL. So all the inbuilt security aspects provided by the MYSQL –Server database is used here. It provides a strong feature of security so that it will be difficult to change, modify any personal or universal data. Information is vital to success but when damaged or in the wrong hands, it can threaten success. MYSQL provides extensive security features to safe guard your information from both unauthorized viewing and intentional or inadvertent damage. This security is provided by granting or revoking privileges on a person-by-person and privilege-by-privilege basis.

## **Limitation**

The portal industry is several years old, and vendors come into and out of the market every month. Since typical licensing and development costs are several hundred thousand dollars or more, vendor selection is high risk. (In addition to some eight major vendors, a higher-education consortium is in the process of developing an open framework called the JA-SIG portal.) The current volatility of the portal market and the lack of agreed upon standards argues for institutions to wait to jump into a portal unless there is a clear need or benefit that requires one.

Developing a campus portal is a key strategic technology decision that will impact the entire campus community and every other strategic technology program such as CMS. The decision on a portal strategy requires careful analysis of long-term and short-term needs.

Campuses that do intend to begin the process of developing a portal need to consider the following issues:

- What short-term problem does the campus intend to solve with a portal, and is a portal the best solution?
- Is executive management willing to mandate a single portal for the campus?
- Does executive management understand that a portal represents an ongoing commitment rather than a onetime investment?
- Who owns and manages the portal?
- Is advertising appropriate? E-commerce.

An emerging consensus regarding portal development includes the following major best practices and considerations:

- There should be one and only one horizontal portal on campus;
- Portals should be developed iteratively;
- The portal should support “single sign-on”; that is, with a single user id and password, each user can access all the applications and data that she or he is allowed to use;
- Campuses should consider integration with both legacy systems and CMS;
- Courseware management tools should be integrated with the portal; and
- While revenue generation should not drive the development of a portal, the design should allow advertising and e-commerce if desirable and appropriate.

In addition, careful consideration of security, privacy, and protection of intellectual property must be part of the portal development project.

## **CONCLUSION**

In conclusion, the Hotel Management project has been a comprehensive exploration into the intricacies of managing a successful hotel operation. Throughout the course of this project, we delved into various aspects, ranging from efficient reservation systems and customer service strategies to effective staff management and technology integration.

One of the key findings of this project is the pivotal role that technology plays in enhancing the overall guest experience and streamlining hotel operations. The implementation of a robust reservation system not only increased booking efficiency but also contributed to improved customer satisfaction. Additionally, the utilization of data analytics proved instrumental in making informed decisions, optimizing resource allocation, and identifying trends to stay competitive in the dynamic hospitality industry.

Our study also underscored the significance of a well-trained and motivated staff in delivering exceptional service. Staff training programs and employee engagement initiatives were identified as crucial elements in maintaining high service standards, leading to increased guest loyalty and positive online reviews.

Furthermore, the project emphasized the importance of sustainability in modern hotel management. Adopting eco-friendly practices not only aligns with global environmental goals but also resonates positively with eco-conscious guests, providing a competitive edge in the market.

As we conclude, it is evident that the successful management of a hotel requires a holistic approach, considering technological advancements, human resource dynamics, and environmental responsibilities. The insights gained from this project serve as a valuable resource for aspiring hotel managers and industry professionals seeking to adapt to the evolving landscape of hospitality.

In essence, the Hotel Management project has provided a nuanced understanding of the multifaceted challenges and opportunities in the field, and its findings contribute to the ongoing discourse on optimizing hotel operations for a sustainable and customer-centric future.



## **FUTURE SCOPE**

The future scope of a Hotel Management project can be quite extensive, as the hospitality industry continues to evolve and adopt new technologies. Here are some potential areas of growth and development for Hotel Management projects:

Continued integration of technology for seamless hotel operations, including reservation systems, online booking platforms, and mobile check-in/check-out processes.

Implementation of smart room technology, such as IoT devices for climate control, lighting, and entertainment systems.

Utilizing data analytics tools to gather and analyze customer preferences, behavior, and trends to enhance personalized services.

Implementing business intelligence solutions for better decision-making and optimizing hotel performance.

Developing and enhancing mobile apps for hotel management, allowing guests to easily book rooms, access hotel services, and receive personalized recommendations.

Exploring the use of blockchain for secure and transparent transactions, especially in areas like payment processing and guest identity verification.

Integrating AI-powered chatbots for customer support, reservations, and FAQs, providing instant responses and improving overall customer experience.

Implementing VR/AR for virtual tours, enabling guests to explore hotel facilities and rooms before making a reservation.

Incorporating sustainable practices and green technologies to appeal to environmentally conscious travelers.

AI is being used to automate tasks, personalize guest experiences, and make better decisions. For example, AI-powered chatbots can answer guest questions and provide recommendations, while AI-based analytics can help hotels identify trends and improve their operations.

Cloud-based HMS are becoming increasingly popular, as they offer scalability, flexibility, and cost savings. Cloud-based HMS can be accessed from anywhere in the world, and they can be easily updated with new features and functionality.

HMS are being integrated with other systems, such as property management systems (PMS), point-of-sale (POS) systems, and revenue management systems. This integration is helping to create a seamless guest experience and improve operational efficiency.

IoT devices are being used to collect data about guests, their preferences, and their interactions with the hotel. This data can be used to personalize guest experiences and improve operational efficiency.

## **REFERENCES**

- **<https://www.youtube.com>**
- **<https://www.w3schools.com/sql>**
- **<https://www.google.com>**
- **<http://projectmanagementdocs.com/>**
- **John Zukowski, “Mastering Java2” (2000), BPB Publications**
- **Stefen Denninger, “Enterprise Java Beans-2.1” Author’s Press**
- **Rajeev mall ‘Software engineering’**
- **Elmasri Navathe ‘Fundamentals of database systems’**

