Hotel Management System Project Report

Files:

https://drive.google.com/file/d/1f-7z0olchynkdFvA8eVZjKoUOlxvlqXl/view?usp=sharing

---

Table of Contents

---

1. Introduction

This report documents a comprehensive Hotel Management System developed using **Python** and **MySQL**. The system provides a complete solution for managing hotel operations including customer registration, room booking, room management, and contact information. The application features a graphical user interface (GUI) built with **Tkinter** and integrates with a **MySQL database** for persistent data storage.

---

2. System Architecture

The system follows a three-tier architecture:

- **Presentation Layer**: Tkinter-based GUI
- **Application Layer**: Python business logic
- **Data Layer**: MySQL database

Key components:

- `hotel.py`: Main application window
- `custmer.py`: Customer management module
- `room.py`: Room booking module

- `details.py`: Room management module
- `contactUs.py`: Contact information module

---

3. Database Design

The system uses **MySQL Workbench** with the following tables:

Customer Table

Stores customer personal information
Fields: Ref (PK), Name, Mother, Gender, PostCode, Mobile, Email, Nationality, IdProof, IdNumber, Address

Room Details Table

Stores room inventory information
Fields: floor, roomno (PK), roomType

Room Booking Table

Manages room reservations
Fields: contact (FK), checkinDate, checkoutDate, roomType, room, meal, noOfDays

Entity-Relationship Diagram:

- One-to-many relationship between Customer and Room Booking

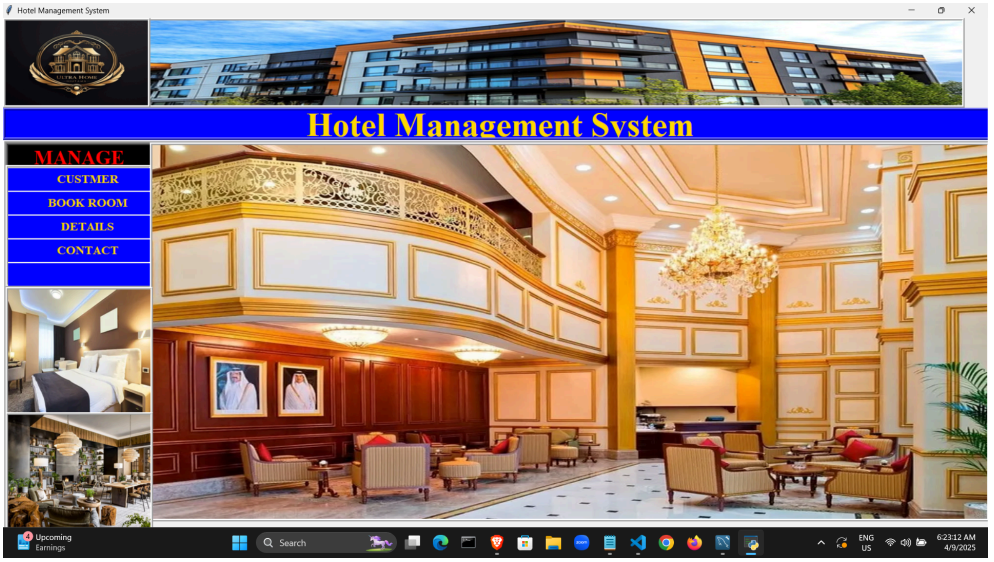
- Many-to-one relationship between Room Details and Room Booking

---

4. GUI Implementation

Main Window (hotel.py)

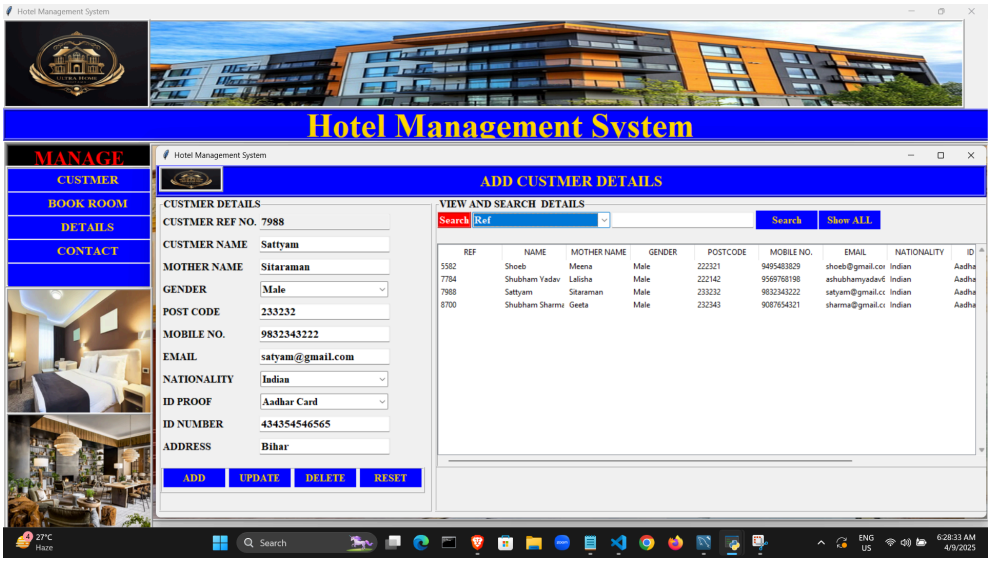
- Features a modern dashboard with navigation buttons
- Displays hotel images for visual appeal
- Provides access to all system modules



Customer Management (custmer.py)

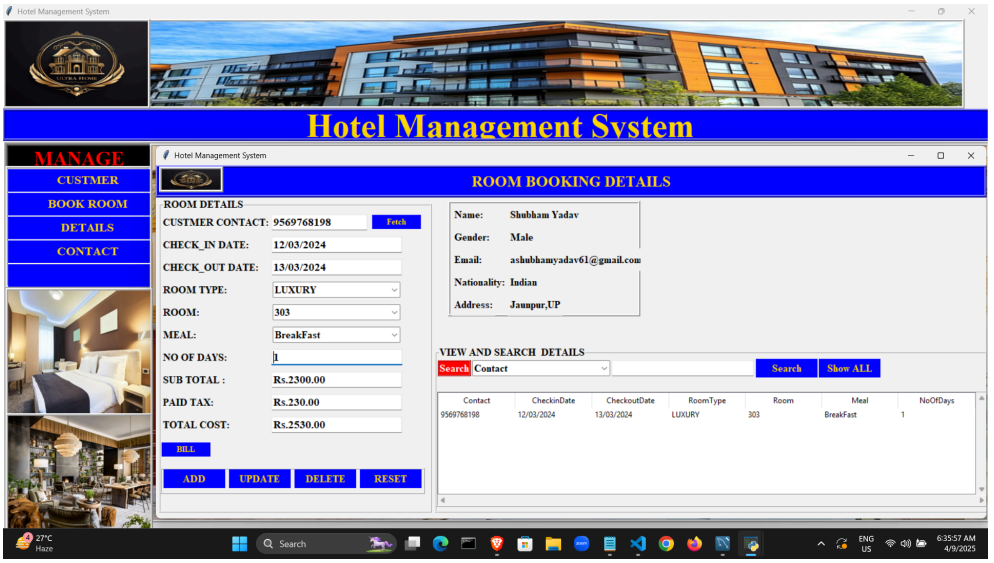
- Form for capturing customer details
- Search functionality with multiple filters

- Data table for viewing all customers
- CRUD operations (Create, Read, Update, Delete)



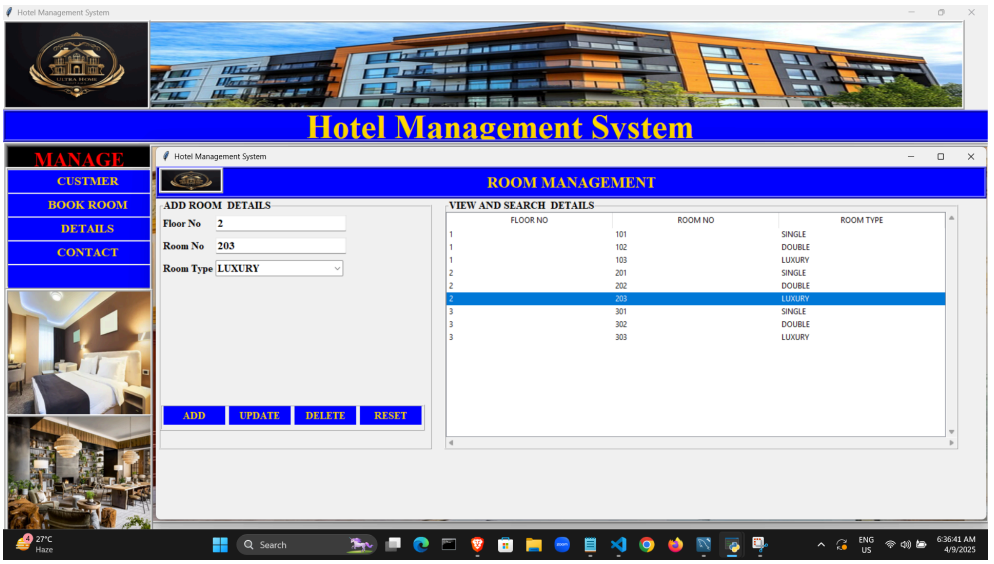
Room Booking (room.py)

- Integrated with customer and room databases
- Bill calculation with tax computation
- Date difference calculation for stay duration
- Room availability checking



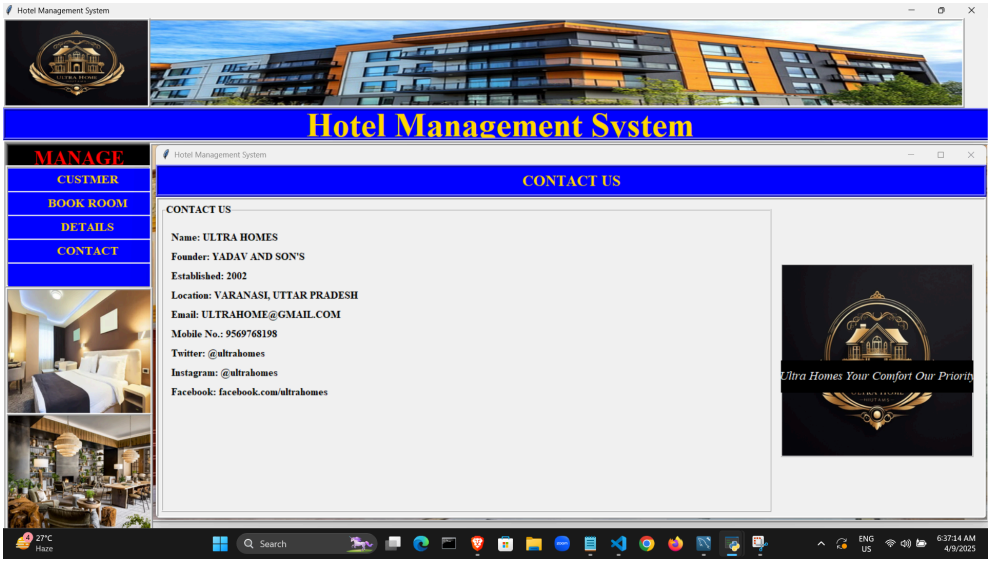
Room Management (details.py)

- Add/remove rooms from inventory
- Set room types (SINGLE, DOUBLE, LUXURY)
- Floor management system



Contact Us (contactUs.py)

- Displays hotel contact information
- Features animated text effects
- Shows company logo



---

5. Functionality Overview

Core Features:

- **Customer Registration**
    - Auto-generated reference numbers
    - Comprehensive personal details
    - ID proof management
- **Room Booking System**

    - Integrated with customer database
    - Automatic bill calculation
    - Meal options (Breakfast, Lunch, Dinner)
- **Room Inventory Management**
    - Add/remove rooms
    - Categorize by type and floor
    - View all available rooms
- **Search Functionality**
    - Search across all tables
    - Multiple filter options
    - Real-time results display

---

6. Database Operations

Connection Management:

```
conn = mysql.connector.connect(
    host="localhost",
    username="root",
    password="3030",
```

```
    database="new_schema"
)
```

CRUD Operations:

- **Create (Insert):**

```
my_cursor.execute(
    "INSERT INTO customer VALUES (%s, %s, ...,
%s)",
    (self.var_ref.get(), self.var_cust.get(), ...)
)
```

- **Read (Select):**

```
my_cursor.execute("select * from customer")
rows = my_cursor.fetchall()
```

- **Update:**

```
my_cursor.execute("update customer set
Name=%s,... where Ref=%s",
    (self.var_cust.get(), ..., self.var_ref.get()))
```

- **Delete:**

```
my_cursor.execute("delete from customer where
Ref=%s",
    (self.var_ref.get(),))
```

Data Binding:

- Tkinter variables (`StringVar`, `IntVar`) bound to database fields
- Automatic type conversion between GUI and database

---

7. Cross-Table Operations

Customer-Room Integration:

```
# In room booking module
query = ("SELECT Name FROM customer WHERE
Mobile=%s")
value = (self.var_contact.get(),)
my_cursor.execute(query, value)
```

Room Availability Checking:

```
# Get all rooms from details table
my_cursor.execute("select roomno from details")
rows = my_cursor.fetchall()
```

Bill Calculation:

- Combines room rates and meal costs
- Calculates tax automatically
- Computes based on stay duration

---

8. Error Handling

Input Validation:

```
if self.var_mobile.get() == "" or self.var_mother.get()
== "":
    messagebox.showerror("Error", "Please Enter
Required Fields")
```

Database Error Handling:

```
try:
    # database operations
except mysql.connector.Error as err:
    messagebox.showerror("Error", f"Database error:
{str(err)}")
except Exception as e:
    messagebox.showerror("Error", f"Unexpected error:
{str(e)}")
```

Transaction Management:

- Explicit commit/rollback

- Connection cleanup in finally blocks (if implemented)

---

9. Conclusion

This Hotel Management System demonstrates:

- Effective integration of Python GUI with MySQL database
- Implementation of complex business logic (room booking, billing)
- Robust data management with proper error handling
- User-friendly interface with visual elements

It serves as a real-life example of how hotel operations can be digitized and managed efficiently through software.