 Senior project tittle:

Guest House Renting Management System

Group Members:

Name ID

Eyob Debebe RCS/304/21

Mengstu Merech RCS/320/21

Mesay Mesfin RCS/322/21

Advisor:

Mr. Belilign Teferi

Jun, 2025 G.C

**Chapter 1: Introduction**

In today’s fast-paced world, where convenience and comfort are increasingly prioritized, many travelers and property owners are turning to digital solutions to streamline short-term accommodation.  
Traditional room booking methods are often manual, inefficient, and prone to errors causing double bookings, missed reservations, and guest dissatisfaction.

So far, only a few guest houses have adopted fully automated, tech-enabled management systems.  
However, the growing demand for smooth, contactless, and reliable booking experiences makes this system a practical and forward-thinking solution.

This project proposes a model that focuses on:

* Convenience
* Efficiency
* User Satisfaction

**1.1Background**

* The Guest House Renting Management System is being developed to modernize and improve the management of guest house accommodations in Ethiopia. Traditionally, guest house services have been managed manually, relying heavily on phone bookings, paper records, and walk-in reservations. This has limited the visibility, accessibility, and efficiency of the sector, especially in regions with growing tourism and travel needs.
* Guest houses play a crucial role in supporting both domestic and international travelers by offering affordable and accessible lodging options. However, the lack of a centralized and automated system has made it challenging to manage bookings, track availability, and ensure service quality across different locations.
* A Guest House Renting Management System (GHRMS) is especially important for Ethiopia for the following reasons:
  + Fragmented Accommodation Sector: Guest house information is often scattered, unregistered, or inconsistently maintained, which limits opportunities for both guests and property owners.
  + Manual Booking Systems: Many guest houses still use paper-based booking and record-keeping, which can lead to double-booking, lost reservations, and inefficient guest management.
  + Limited Market Access for Owners: Without digital visibility, many local guest house operators miss out on potential guests, especially from outside their immediate area or region.

**1.2. Statement of Problem**

some of the problems in not having computerized guest house renting system are:

* Lack of computerized system Currently most guest house owners use the manual system in recording and maintaining their property and customers data
* Data security is not assured in a manual way, data is recorded on books/papers which may easily get damaged leading to loss of data.
* Limited Accessibility: Accessing the guest house renting management system can be challenging, especially for users with limited technical knowledge. The system may not be versatile enough to be compatible with different devices and browsers, limiting its accessibility.
* Inaccurate and Unreliable Information: The current guest house renting management system may contain inaccurate and unreliable information that does not represent a fair description of the rooms or amenities available in the guest house. This can lead to customer dissatisfaction and a decline in customer loyalty.
* Poor Communication and Coordination: Communication and coordination between guest house owners and management and potential customers is often inadequate, resulting in possible booking conflicts and a negative experience for guests.

**1.3 Purpose of the Project:**

The guest house renting management system is significant for various reasons:

* **Optimizes space utilization:** Guest house renting management system helps in optimizing the utilization of space. Homeowners can make use of empty spaces in their homes to provide a temporary stay for guests.
* **Additional income:** Renting out a guest house can provide an additional source of income. This income can be used to maintain the guest house, pay bills or even save for other investments.
* **Better customer service:** By renting out a guest house, you are providing guests with better customer service as compared to staying in a hotel. Guests can experience the comfort of a home and feel more welcomed.
* **Flexibility:** The guest house renting management system offers more flexibility as guests can book the guest house for a shorter or longer period, as per their requirement.
* **Cost savings:** The cost of renting a guest house is generally lower than staying in a hotel. Guests can save money and still enjoy the comforts of home.
* **Tourism promotion:** Guest houses can help in promoting tourism in the local area. Visitors who stay in guest houses are more likely to explore the local attractions and spend money in the local economy.
* **Social interaction:** Staying in a guest house offers guests the opportunity to interact with the hosts and learn more about the local culture and way of life. This can enhance the guest's overall experience.

**1.4 Objective of the Project**

**1.4.1. General objective of the Project**

The general objective of the project is to build guest house renting management system.

**1.4.2. Specific objective of the Project**

The Specific objectives of the project are:

* Gathering required information for the proposed system.
* Specifying functional and non-functional requirements of the proposed System
* To design and implement the user interface of the system
* Deployment of a database for the proposed system
* Testing and validating the proposed system

**1.5 Scope and Limitation of the project**

**1.5.1 Scope of the Project**

A **Guest House Renting M**anagement **System (GHRMS)** offers a complete digital solution aimed at enhancing the management, reservation, and communication processes of guest house operations. It bridges the gap between guests and administrators, ensuring smooth and efficient handling of room bookings, payments, and overall guest services. The system encompasses several core functionalities as outlined below:

* **User Management:** The GHRMS ensures a secure and user-friendly environment by implementing robust user management capabilities. Both administrators and guests can register and log into the system through secure authentication methods, including encrypted passwords and verification protocols. Guests can maintain personalized user profiles that store preferences, booking history, and contact information, streamlining future interactions. Meanwhile, admins have access to dashboards that allow them to monitor user activity, manage roles, and ensure system integrity. The role-based access control system ensures that sensitive functions like pricing adjustments and room inventory updates are only available to authorized personnel, fostering data security and operational efficiency.
* **Room Management:** Efficient room management lies at the heart of the system. Administrators can easily add new rooms, edit existing listings, or remove outdated ones. Each room listing can include detailed information such as pricing, number of beds, amenities (Wi-Fi, air conditioning, etc.), and high-quality photos. Room availability can be managed in real-time, ensuring that potential guests always view the most current status. This digital approach replaces traditional manual logs with an intuitive interface, significantly reducing errors and administrative overhead.
* **Booking Module:** Guests benefit from a responsive and real-time booking engine. The system checks room availability instantly, allowing guests to secure their preferred rooms with immediate confirmation. Once booked, users have the flexibility to cancel or modify their reservations within the policy limits. The module ensures smooth scheduling without overlaps, and confirms reservations via email to reassure guests. This real-time automation reduces double bookings, minimizes administrative bottlenecks, and enhances the overall customer experience.
* **Payment Integration:** The GHRMS integrates secure online payment gateways, enabling guests to pay conveniently using credit/debit cards, net banking, or digital wallets. All transactions are encrypted and compliant with industry standards to ensure payment security. Once a booking is made, the system generates automated invoices and receipts, providing clear financial documentation for both guests and admins. This transparency simplifies accounting and boosts trust in the platform.
* **Communication Tools:** Communication is central to excellent guest service. The system sends real-time notifications via email regarding booking confirmations, reminders, cancellations, or special offers. Guests are encouraged to leave reviews and feedback after their stay, which helps administrators improve service quality and build credibility. These tools ensure a two-way communication channel that keeps users informed while giving administrators valuable insights into guest satisfaction.
* **Reports and Analytics:** Beyond basic operations, the GHRMS offers powerful analytics to track and optimize performance. Administrators can access visual reports on booking history, room occupancy rates, revenue trends, and guest preferences. These insights help identify peak seasons, monitor underperforming rooms, and adjust pricing strategies accordingly. Predictive analytics can even suggest inventory and staffing adjustments, enabling the guest house to prepare proactively for fluctuating demand. This data-driven approach ensures more effective management and higher guest satisfaction.

.

**1.5.2 Limitation of the Project**

Despite the numerous benefits of developing a new Guest House Renting Management System, there are certain limitations that may affect the project's full potential. One major challenge is the lack of standardized or digitized data across different guest houses, which makes it difficult to implement a system that can scale to multiple locations. As students, we also face time constraints, which limit our ability to explore all features and expand the project in depth. Furthermore, our limited budget restricts our capacity to conduct extensive research on guest house operations in various regions. In addition, the system may involve complex development processes such as online payment integration, secure user authentication, and real-time booking functionalities each requiring advanced technical knowledge and experience in software engineering, which may exceed our current expertise.

**1.6 Tools**

Any item that aids in the completion of a task is considered a tool. It could be a physical instrument, such as a piece of software, such as a web browser.

|  |  |  |  |
| --- | --- | --- | --- |
| **Category** | **Tool** | | **Description** |
| Software | Planning  and Organization | Microsoft Word | Document creation with templates, formatting, and collaboration features. |
| Microsoft PowerPoint | Presentation software for creating visually appealing slideshows. |
| Microsoft Project | Project management tool for planning, organizing, and tracking tasks |
| Drawing and Design | Draw.io | Free online tool for creating diagrams, mind maps, and flowcharts. |
| Figma | Web-based tool for designing websites and apps with real-time collaboration features. |
| Frontend  Development | HTML | Hypertext Markup Language for defining website structure and content. |
| Typescript | Superset of JavaScript with optional typing for improved code readability and error prevention. |
| CSS | Cascading Style Sheets defines website visual style (fonts, colors, layouts, etc.). |
| Backend Development | Express.js | Web framework built on Node.js for simplified web application and API development. |
| Postgress | Open-source relational database management system for storing and organizing data. |
| Version Control | GitHub | Platform for hosting code, collaborating on projects, and sharing code publicly or privately. |
| Hardware | Smartphones | |  |
| Laptops | | Lenovo, HP,Dell |
| Desktops | | Dell with specifications processor, RAM, storage |
| Flash drives | |  |
|
| RG45 cables | |  |
| Operating Systems | Windows 10 pro | |  |
| Windows 11 pro | |  |

#### Table ‑ Tool Description

**1.7 Methodology**

**1.7.1 System development life cycle**

The System Development Life Cycle (SDLC) is a road map for developing software. The SDLC is divided into phases: planning and analysis, design, development, testing, implementation and deployment, and maintenance. This disciplined method keeps everything organized, eliminates risks, and ensures a high-quality end result.



#### Figure ‑ System Development lifecycle

**1.7.2 Data collection methodology**

Data collection is one of important tasks to analyze how activities done in existing system and developed the new system. The following data collection methods are we used to gather the data from the blood and tissue service.

* Interview

It is a structured conversation where one person (the interviewer) asks questions and another person (the interviewee) provides answers. So we used this method to get the information from the real able person that has more experience and knowledge about the current system. This helps us to get detail information and well defined answer.

* Observation

We used to gather additional data by observing the actual work being done by the staff and consolidated with what was obtained through interviewer.

* Document analysis

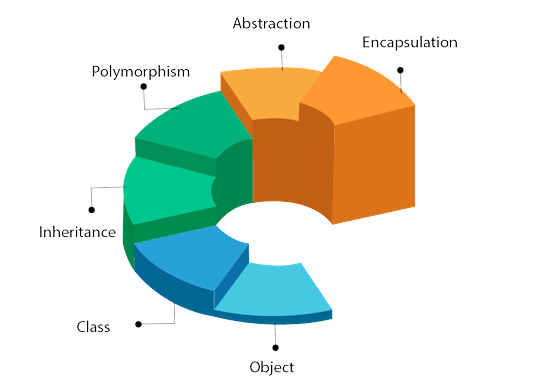
The method of examining various documents that include concepts connected to our project and the rationale behind selecting it is Documents don't always need costly data collection techniques and are frequently easily accessible.

**1.7.3 Analysis and design methodology**

We used object-oriented analysis and design approach for modeling and designing the modules of proposed system.

Hybrid design methodology combines elements from different design approaches to create a system that leverages the strengths of each method. It avoids the limitations of a single approach and aims for a more robust and efficient development process. So we chose this methodology because of:

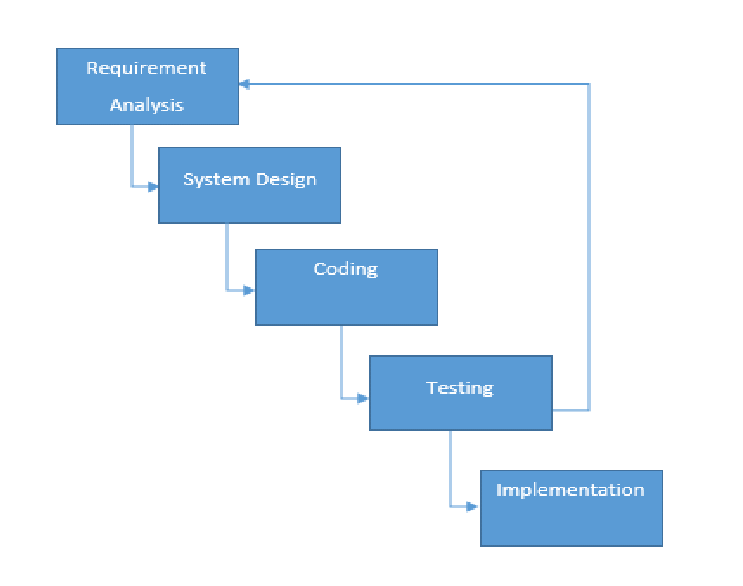
* **Flexibility and Adaptability:** Can adapt to changing project requirements or new technologies that emerge during development.
* **Reduced Risk:** Early planning (from Waterfall) mitigates risks, while frequent feedback loops (from Agile) allow for course correction.
* **Improved Efficiency:** Leverages the strengths of both approaches to streamline development and avoid rework.
* **Better Communication and Collaboration:** Promotes communication between stakeholders and encourages collaboration throughout the development process.



#### Figure ‑ object-oriented Diagram

**1.7.4 Software development Model**

The iterative waterfall approach combines structure and flexibility. It divides projects into mini-waterfalls (plan, design, develop, test) for each feature. After each mini-waterfall, input is used to revise preparations before proceeding. It provides greater versatility than the classic waterfall paradigm while retaining some predictability. Consider it like building a house one part at a time, with the flexibility to change the designs based on what you learn along the way.  
The Waterfall Iterative Model In contrast to the strict Waterfall methodology, this method permits going back and reviewing previous iteration cycles. This implies that as we grow, we might take feedback into account and modify to meet new specifications. Every iteration adds a useful feature to the project, promoting user involvement and reducing risk. All things considered, it's a great fit for our projects with a core set of demands that could alter. It offers structure while allowing for flexibility, striking a good balance between rigidity and complete flexibility.



#### Figure ‑ Iterative Waterfall model Diagram

**1.8 Time Schedule of the Project**

A timeline for the project implementation will be established, including milestones for system development, testing, training, and deployment. Regular monitoring and evaluation will be conducted to ensure project timelines are met.

The robust project management software we utilize, like Microsoft Project Management Software, facilitates the development of our advanced planning software. This ensures we stay on track while embracing changing requirements.

Microsoft Project is a powerful project management software tool developed and sold by Microsoft. It's widely used by individuals and teams to plan, manage, and track projects of various sizes and complexities.



#### Figure ‑ Gantt chart

**Chapter 2: Requirement Analysis**

**2.1 Introduction**

In response to the growing demand for efficient guest accommodation services, we propose the development of a comprehensive Guest House Renting Management System (GHRS). This system aims to streamline the end-to-end process of guest house operations, including room reservations, pricing management, customer communication, and payment processing. By offering a centralized digital platform, GHRS enables guests to search, book, and pay for rooms online, while providing administrators with tools to manage room availability, monitor earnings, and engage with customers efficiently. Key features of the system include secure user management, dynamic room listings, real-time booking capabilities, integrated payment solutions, automated notifications, and robust analytics. This Requirement Analysis outlines the functional and non-functional requirements essential for the successful deployment of the GHRS, with the goal of improving operational efficiency, enhancing guest satisfaction, and optimizing the use of resources across the guest house rental ecosystem.

**2.2 Current System**

The current guest house renting system in our country is mostly manual. When a customer wants to book a room, he/she must visit the guest house in person or call the owner to check availability and make a reservation. This process is also used for checking room information, prices, and amenities. There is no centralized or automated system to handle bookings, payments, or manage room inventory, which often leads to errors, double bookings, and customer dissatisfaction.

**2.2.1 Report generated by current System**

So far, guest house renting systems in Ethiopia do not follow a common reporting format. Because of this, each guest house uses its own custom style to generate and print transaction or performance reports.

**2.2.2 Forms and Documents Used in Current System**

In the current system we cannot get an organized form and document. But we got this guest house report form document:

Guest House Report Form:

The guest registration form is a key document used to gather important information from guests during check-in at the guest house. It starts by recording the guest house name, address, contact and guest’s full name, gender, job, address, nationality, and contact number to ensure accurate identification and communication. Additionally, the form includes fields for official identification details such as an ID or passport number, which helps verify the guest’s identity and maintain security. the form also captures room number, stay duration, including check-in and check-out dates.

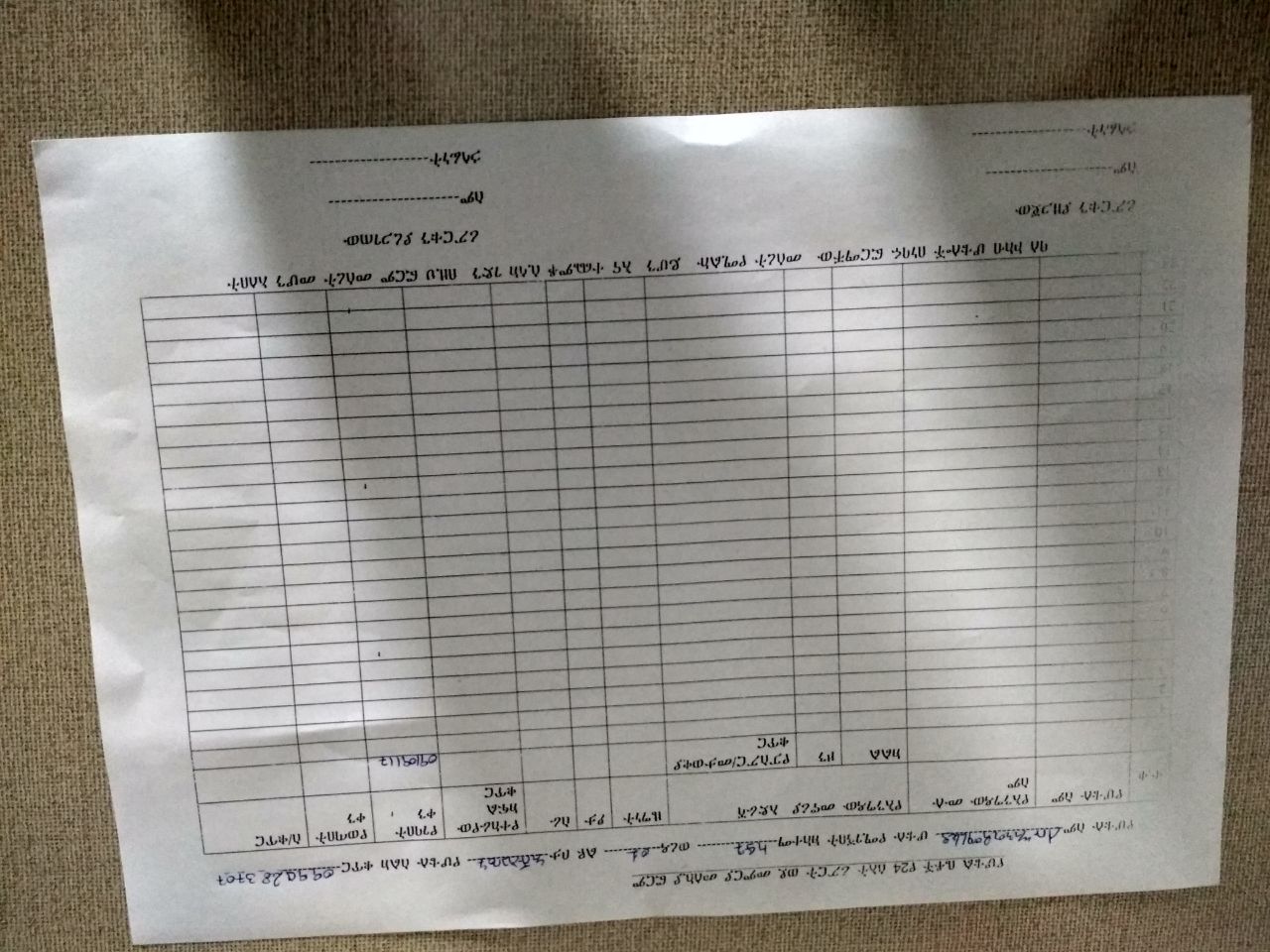


Figure 2-1 Guest house report form

**2.2.3 Players in the Existing System**

The main players in the existing system are there, and those are:

**Manager**: the manager is the one in charge of the whole guest house. They make decisions, plan staff schedules, check reports from the front desk and cashier people, and promote the guest house to attract more customers. If something big comes up, the manager handles it.

**Guest: the guest** is the main customer of the guest house. Their roles include checking room availability (which is prepared or communicated by the front desk staff), selecting the type of room they wish to stay in, making the required payment, and receiving a receipt.

**Front Desk Staff (Resumption):** the front desk staff is the person you meet when you enter the guest house. They help guests check in and check out, tell them which rooms are free, collect payments, and give receipts. They also keep guest records and write daily reports for the manager.

**Supervisor:** The supervisor checks rooms after guests leave to ensure they are clean, beds are made, and towels are replacedand make sure everything is ready for the next guest. They report any damage or issues to the front desk or manager.

**Cashier: the cashier** is responsible for handling the financial records and transactions of the guest house. Their duties include processing payments made by guests (via cash, or mobile money), managing income reports, preparing financial statements, and submitting summaries to the manager for review.

**2.2.4 Problems of the Current System**

As we analyzed the current system form the information we got, we identify and generalize the problems as follows:

**Consume of time:** the current renting management process consumes a great deal of time due to its heavy reliance on manual procedures. Staff members must check room availability by flipping through logbooks, record guest information by hand, and manually generate bills and receipts. These tasks are not only time-consuming but also prone to delays during busy periods, causing long queues at the front desk. The lack of automation slows down guest check-in and check-out, room assignments, and service coordination, leading to poor customer experiences and inefficiencies in day-to-day operations.

**Manual system:** the existing system depends almost entirely on paper-based records and manual data entry. Guest registrations, room bookings, service requests, and payment logs are all recorded on paper or basic spreadsheets, which are vulnerable to human error, duplication, and inconsistency. Physical records are also at risk of being lost, torn, or damaged, which can result in incomplete or missing guest histories and financial data. Without a digital system, even simple tasks like generating reports or viewing current room occupancy require tedious, error-prone manual work.

**Management:** from a management perspective, the lack of a centralized digital system makes it difficult for managers to access up-to-date information on guest activities, room status, financial performance, or staff workload. Managers must rely on reports submitted manually by front desk or finance staff, which can be delayed or inaccurate. This limits their ability to make timely, informed decisions, and prevents them from tracking trends or performance metrics effectively. Moreover, the absence of real-time dashboards hinders overall visibility and control over operations**.**

**Security:** Security is another major concern in the current system. Guest details such as personal identification numbers, contact information, and payment history are stored in unsecured paper files or loosely organized spreadsheets. This makes the data vulnerable to unauthorized access, loss, or misuse. There is no role-based access control, meaning any staff member could potentially access or alter sensitive guest information. These weaknesses put guest privacy at risk and expose the guest house to potential legal and reputational damage.

**Post ponding schedule:** Once a guest books a room, there’s usually no simple way to change or reschedule their stay if something comes up. For example, if someone made a booking for next weekend but then needs to delay it by a day, the current system doesn’t allow them to easily request a change. Because everything is done manually, there’s no flexibility. Guests either lose their booking, have to come in person to make changes, or call and hope someone writes it down correctly. This creates frustration for guests and confusion for staff especially when it leads to empty rooms or overlapping bookings. Without a proper system to manage reschedules or cancellations, it’s hard to keep things running smoothly.

**2.2.5 Practices to be preserved from the Current System**

We agreed on to preserve all practices from the current system except the payment system because we are not shore that we can include these systems while there are no police to support in. And also, we got some difficulties.

**2.3Proposed System**

**2.3.1 Overview**

A Guest House Renting Management System is a software solution designed to streamline guest house operations, manage bookings, monitor room availability, coordinate housekeeping, and handle billing in a centralized, efficient way. The system helps both guests and staff by automating daily activities, reducing errors, and enhancing the overall experience.

**Essential Features:**

**Guest Management**

Guest information stored in Database: Maintains a centralized record of all guests including contact details, identity verification, booking history, preferences, and special requests.

Registration and Profile Management: Enables easy registration of new guests and editing of existing guest information. Guests can log in to view their booking history and upcoming reservations.

Feedback Collection: Allows guests to submit reviews and feedback after their stay, which helps improve service quality.

**Room Booking & Reservation**

Real-Time Availability: Displays up-to-date information on room availability, rates, and special offers. Prevents double-booking by synchronizing room status instantly.

Online and Front Desk Booking: Supports room booking both through an online portal and by front desk staff, ensuring flexibility and convenience.

Booking Modification and Cancellation: Allows guests or staff to modify or cancel bookings based on set policies, helping to reduce no-shows or scheduling conflicts.

**Supervised the guest house**

Task Assignment: notify cleaning tasks to supervisor after check-out or before a guest arrives.

Status Monitoring: Tracks room cleanliness and readiness in real-time. Notifies front desk when rooms are ready for occupancy.

Issue Reporting: Allows Supervisor to report maintenance issues that require attention.

**Billing and Payment Management**

Multi-Payment Options: Supports cash, mobile money, and digital wallet payments.

Discounts and Refunds: Handles promotions, loyalty discounts, and refunds for cancellations.

**Reporting and Analytics**

Occupancy Reports: Generates daily, weekly, and monthly reports showing room occupancy, vacancy, and average stay duration.

Customer Behavior Reports: Analyzes repeat guests, peak booking times, and popular room types to help with marketing and pricing decisions.

**User Management**

Role-Based Access: Provides secure login for different types of users Manager, Front Desk Staff, Supervisor, and Cashier each with specific permissions.

**Online Portal and Integration**

Guest Portal: Guests can log in to book rooms, view receipts, update personal information, or request services.

Third-Party Integration: Connects with payment gateways, email systems for notifications.

**2.3.2 Functional Requirements for Gust House Renting Management System (GHRMS)**

**2.3.2.1 Business Rule**

|  |  |
| --- | --- |
| BR Id | Description |
| BR01 | Guests must provide valid identification (ID/passport) when booking or checking in to ensure security and legal compliance. |
| BR02 | A room cannot be booked if it is already reserved or occupied during the selected dates. |
| BR03 | Guests can only cancel or change their reservation within a defined time window, otherwise a cancellation fee applies. |
| BR04 | A booking cannot proceed if no rooms are available for the requested period. |
| BR05 | Room records can only be deleted if the room is not currently booked. |
| BR06 | When a guest checks in, the system must update the room status to “Occupied” and log the check-in date/time. |
| BR07 | A reservation must be linked to at least one room. If no rooms are linked, the reservation is considered incomplete. |
| BR08 | A reservation can only be canceled if it has not yet been checked in. If checked in, the guest must check out before canceling. |
| BR09 | Room details can be edited only if there are no active reservations for that room. |
| BR10 | Room cleaning schedules can only be assigned after guest check-out or based on supervisor’s requests. |
| BR11 | If a room is reported damaged during a guest stay, it must be flagged as “Under Maintenance” and removed from booking availability. |
| BR12 | If a guest causes property damage, the supervisor notifies the damage report to the manager for action. |
| BR13 | Before check-in, the guest's booking must be verified along with payment confirmation. |
| BR14 | A guest cannot extend their stay if the room is already booked by another guest after their original checkout date. |
| BR15 | Rooms must be cleaned and inspected before being assigned to a new guest. |
| BR16 | If a room is flagged as unsafe or under maintenance, it must be hidden from the guest portal until fixed. |
| BR17 | Bookings that are inactive for a long period without confirmation should be automatically marked as expired. |
| BR18 | Staff can only access system features based on their roles. |
| BR19 | All payment records must be retained, and any change to a financial record must be traceable. |
| BR20 | If a reservation or payment is mistakenly canceled or edited, it must be possible to roll it back rather than delete it. |

**2.3.2.2Functional Requirements that dividing in case of Actor:**

1. Guest:

* **FR-1:** The system allows new guests to register by providing personal information and contact details.
* **FR-2:** The system allows guests to browse available rooms based on location, dates, room type, and price.
* **FR-3:** The system allows guests to book a room, make payments, and receive booking confirmation and invoices.
* **FR-4:** The system allows guests to cancel or modify their reservation before the check-in date (if allowed by policy).
* **FR-5:** The system provides booking history and receipts for past stays.
* **FR-6:** The system allows guests to submit feedback or requests.

2. Front Desk Staff:

* **FR-7:** The system allows staff to check guests in and out, update room status accordingly.
* **FR-8:** The system allows staff to view, edit, and manage guest bookings and assign rooms.
* **FR-9:** The system alerts staff to upcoming check-ins, check-outs, and late arrivals.
* **FR-10:** The system enables communication with guests about booking changes or room availability.
* **FR-11:** The system allows staff to generate and print guest invoices and receipts.

3. Supervisor:

* **FR-12:** The reception (front desk staff) notifies supervisor that need cleaning after guest check-out.
* **FR-13:** The system allows supervisor to report damages or maintenance issues directly.

4. Manager:

* **FR-14:** The system provides a dashboard showing occupancy rates, income reports, and guest feedback.
* **FR-15:** The system allows the manager to adjust room pricing, set promotional discounts, and change room availability.
* **FR-16:** The system allows the manager to review and approve maintenance or damage reports.
* **FR-17:** The system provides reports on performance metrics like revenue, bookings, and customer satisfaction.

5. Cashier:

* **FR-18:** The system tracks all financial transactions including bookings, cancellations, and refunds.
* **FR-19:** The system generates financial reports (e.g., income statements, tax summaries).
* **FR-20:** The system allows finance staff to manage guest payments and update payment status.
* **FR-21:** The system alerts the finance team about failed or pending transactions.

**2.3.3. Non-Functional Requirements for Guest House Renting Management System (GHRMS)**

1. Performance

The system should respond quickly to user requests, such as room booking, check-in/out, and payment processing. It must handle multiple guest requests and transactions at the same time without delays or slowdowns, especially during peak hours.

2. Security

The system should use strong encryption and secure login methods to protect sensitive guest information, including personal details and payment data. It should comply with privacy regulations and ensure only authorized staff access specific parts of the system.

3. Scalability

The system should be designed in a way that it can easily grow. As more branches or rooms are added, or more guests start using it, the system should continue to function smoothly without needing major rework.

4. Reliability

The system must be stable and reliable. It should not lose or corrupt reservation or payment data. Regular backups and a clear recovery plan must be in place so that operations can resume quickly after any failure.

5. Usability

The system should be easy to use for all users’ guests, front desk staff, managers, and finance teams. The interface should be clean and intuitive. Training and help guides should be provided to make sure everyone can use the system effectively without confusion.

**2.4 Use Case Diagram**

A use case diagram is a visual representation of how users (actors) interact with a system. It displays the system's functionalities (use cases) and how different actors interact with them to achieve their objectives.

The components of use case for Guest House Renting Management System are here below:

**Actors:** These are the external entities (people, organizations, or systems) that interact with the system. They are often portrayed as stick figures. The actors of our systems are:

**Use Cases:** These are the functionalities or features provided by the system. They appear as ellipses or oval forms. The Guest House Renting Management System Use cases are here below:

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID** | **Use Case Name** | **Actor** | **Description** |
| UC01 | Register | Guest | Create an account to book rooms and manage reservations. |
| UC02 | Log In | All Actors | Log into the system to access functionalities. |
| UC03 | Forgot Password | All Actors | Recover account access using email. |
| UC04 | Log Out | All Actors | Securely exit the system. |
| UC05 | Manage Guest House Account | Super Admin | Create, update, or manage guest house accounts. |
| UC06 | Manage System Admin Account | Super Admin | Create, update, or manage system admin user accounts. |
| UC07 | Manage Staff Account | System Admin | Create, update, or manage staff accounts. |
| UC08 | Manage Guest Account | Reception | Create, update, or manage guest profiles manually. |
| UC09 | Search Room Availability | Guest, Reception | Search available rooms by date, type, or guest preference. |
| UC10 | Book Room | Guest, Reception | Book an available room with real-time confirmation. |
| UC11 | Cancel Booking | Guest | Cancel an existing booking according to policy. |
| UC12 | View Booking History | Guest | View previous and upcoming bookings. |
| UC13 | Check in Guest | Reception | Confirm guest arrival and update room status. |
| UC14 | Check Out Guest | Reception | Complete check-out process and update room status. |
| UC15 | Assign Room Cleaning Task | Reception | Assign cleaning tasks to supervisor. |
| UC16 | Update Room Status | Supervisor | Change room status to cleaned, occupied, or under maintenance. |
| UC17 | Report Room Issues | Supervisor | Log or report maintenance issues for a room. |
| UC18 | Process Payment / Pay | Guest | Process room payment through supported payment methods. |
| UC19 | Issue Refund | Cashier | Process refund requests for canceled or disputed stays. |
| UC20 | Verify Payment Status | Cashier | Confirm whether payment has been made or completed. |
| UC21 | View Payment History | Guest | View all previous and current payments for bookings. |
| UC22 | View Booking Reports | Manager | View occupancy statistics and booking trends. |
| UC23 | Manage Pricing & Room Types | Manager | Adjust pricing, room descriptions, and promotional offers. |
| UC24 | Backup Data | System | Automatically back up all system and user data. |
| UC25 | Send Notifications | System | Send alerts for booking, payment, feedback, or promotions. |
| UC26 | View Room Issue Reports | Manager | Review logged maintenance and issue reports for rooms. |

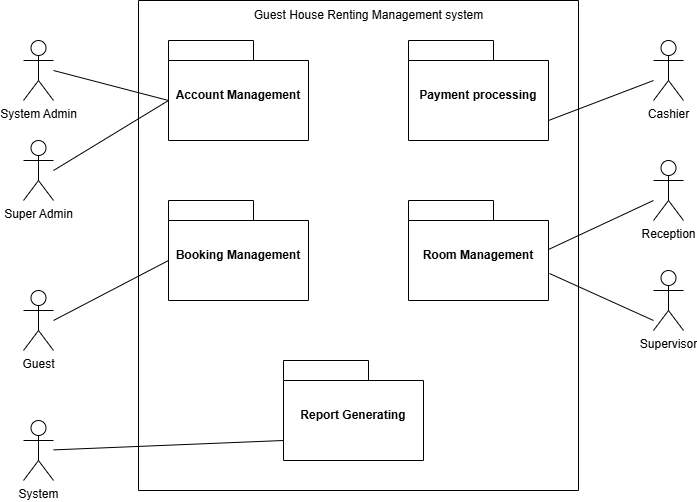


Figure 2-4 package

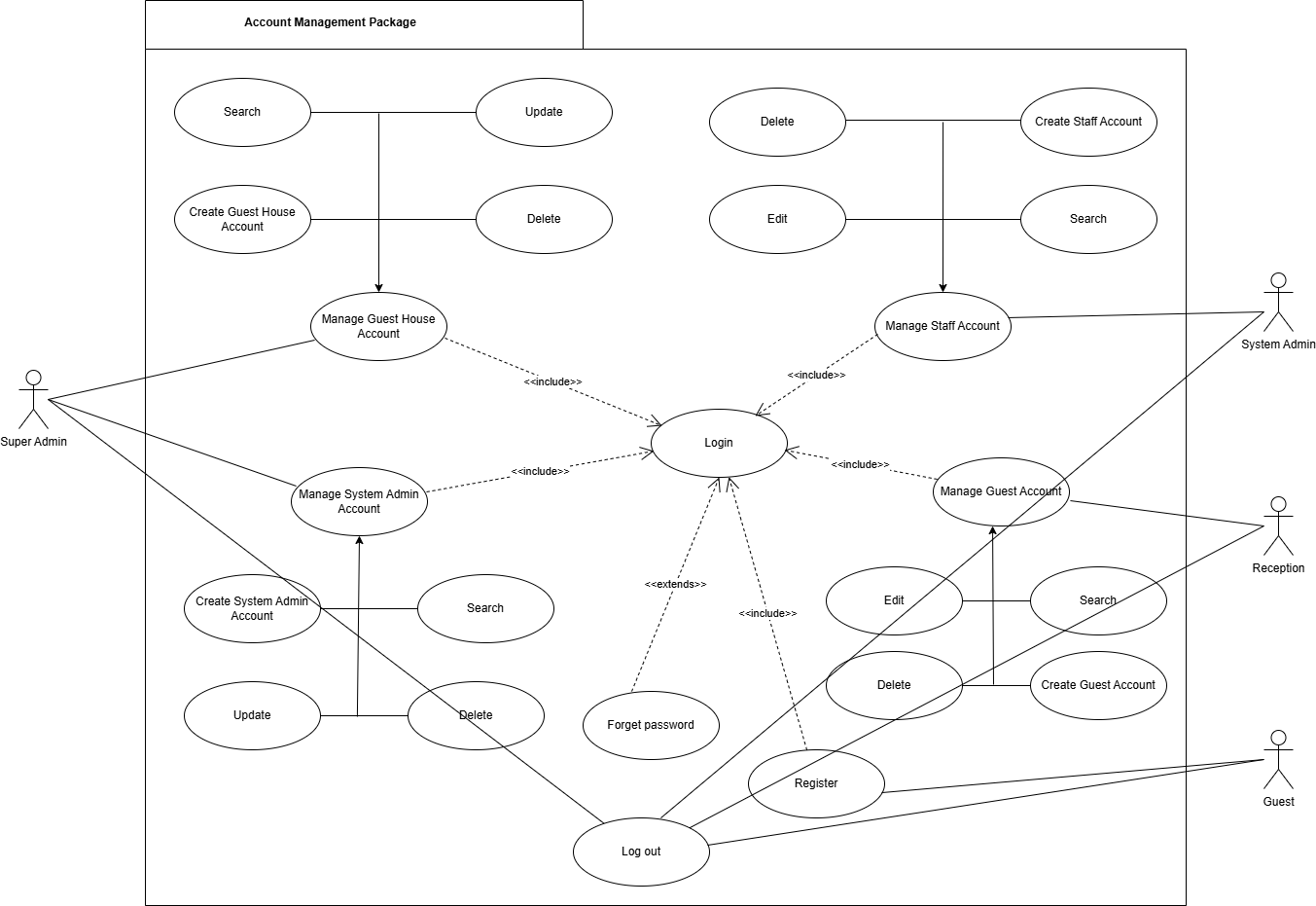


Figure 2-5 account management package

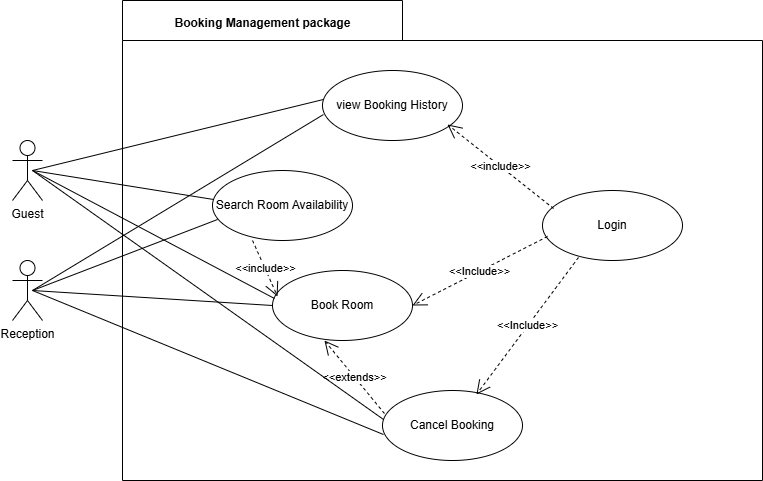


Figure 2-6 booking management package

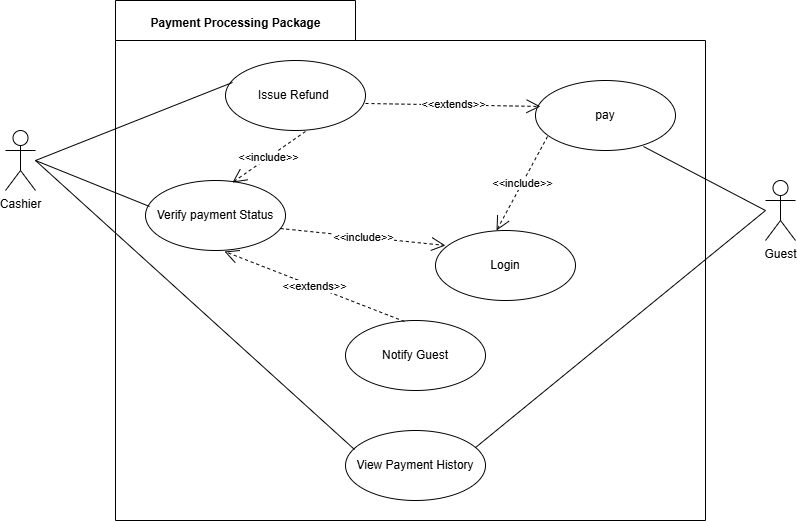


Figure 2-7 payment processing package

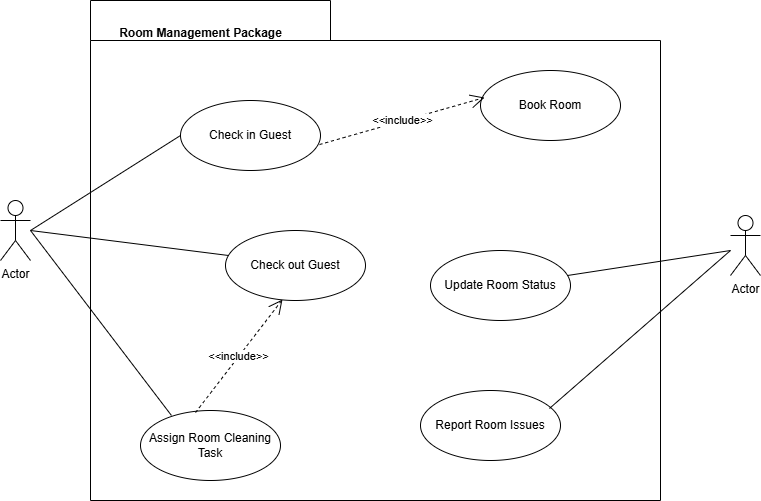


Figure 2-8 room management package

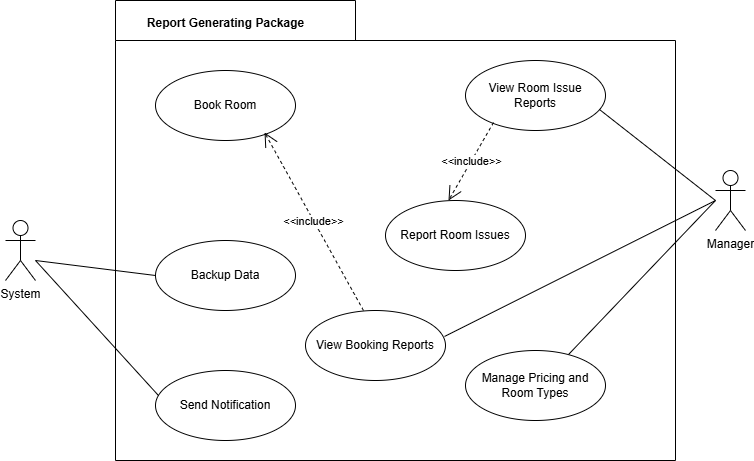


Figure 2-9 report generating package

|  |  |
| --- | --- |
| Use Case ID: | UC01 |
| Use Case Name: | Register |
| Actors: | Guest |
| Description: | This use case describes the process by which a new guest creates an account on the Guest House Renting website to be able to search, book, and manage room reservations. |
| Preconditions: | 1. The guest is not currently registered in the system. 2. The guest has access to the internet and a web browser. 3. The website is online and accessible. |
| Include: | Login |
| Extend: |  |
| Basic Course of Action: | 1. The guest opens the guest house renting website in their browser. 2. The guest clicks the "Register" or "Create Account" link on the homepage. 3. The system displays a registration form requiring details such as full name, email, phone number, and password. 4. The guest fills in the form and clicks the “Submit” button. 5. The system validates the input (e.g., required fields, valid email, password strength). 6. If validation is successful, the system creates a new guest account and stores it in the database. 7. The system sends a confirmation email to the guest. 8. The system redirects the guest to the login page or displays a success message. 9. The use case ends. |
| Alternative Course of Action: | 5A. Validation Fails:  If the system detects missing or invalid data (e.g., empty fields, weak password), it displays relevant error messages and prompts the guest to correct and resubmit the form.  6A. Email Already Exists:  If the provided email address is already registered, the system notifies the guest and suggests logging in instead. |
| Postconditions: | 1. The guest’s account is successfully created and stored in the system database. 2. The guest receives confirmation of registration. 3. The guest can proceed to log in and access system features like searching and booking rooms. |

Table 2-1 Register

|  |  |
| --- | --- |
| Use Case ID: | UC02 |
| Use Case Name: | Log In |
| Actors: | All Actors |
| Description: | This use case describes how registered users log into the Guest House Renting website to access their respective system features based on roles. |
| Preconditions: | 1. The user (any actor) has already registered and has valid login credentials (email/username and password). 2. The user has access to the website via a browser. 3. The system is online and functional. |
| Include: |  |
| Extend: |  |
| Basic Course of Action: | 1. The user opens the Guest House Renting website in a browser. 2. The user clicks the “Log In” button or link on the homepage. 3. The system displays the login form with fields for email/username and password. 4. The user enters their credentials and clicks the “Submit” or “Log In” button. 5. The system validates the credentials against its database. 6. If the credentials are valid, the system logs in the user and redirects them to the appropriate dashboard based on their role (e.g., Guest Dashboard, Reception Panel). 7. The user can now access the system features permitted to their role. 8. The use case ends. |
| Alternative Course of Action: | 5A. Invalid Credentials:  If the login credentials are incorrect, the system displays an error message and allows the user to retry.  5B. Account Suspended or Deactivated:  If the user’s account has been deactivated by an admin, the system prevents login and displays an account suspension notice.  5C. Forgot Password:  The user may click the “Forgot Password” link, which will trigger UC03: Forgot Password for recovery. |
| Postconditions: | 1. The user is authenticated and redirected to their respective dashboard or homepage. 2. The user session is securely initiated and maintained during the active period. 3. Unsuccessful login attempts are logged. |

Table 2-2 Login

|  |  |
| --- | --- |
| Use Case ID: | UC03 |
| Use Case Name: | Forgot Password |
| Actors: | All Actors |
| Description: | This use case describes how users who have forgotten their passwords can securely recover access to their accounts on the Guest House Renting website. |
| Preconditions: | 1. The user has previously registered an account with a valid email address or phone number. 2. The user can access the Guest House Renting website via a browser. 3. The system’s email or SMS service is operational. |
| Include: |  |
| Extend: | Log In |
| Basic Course of Action: | 1. The user visits the Guest House Renting website and clicks on the “Log In” button. 2. On the login page, the user clicks the “Forgot Password?” link. 3. The system displays a password recovery form asking for the user’s registered email or phone number. 4. The user enters their email/phone number and submits the form. 5. The system verifies whether the contact information exists in the database. 6. If it does, the system generates a secure password reset link or verification code. 7. The system sends the reset link/code to the user via the provided method (email or SMS). 8. The user follows the link or enters the verification code and is taken to a reset password form. 9. The user enters and confirms a new password and submits it. 10. The system updates the password and shows a success message. 11. The user is redirected to the login page to sign in with the new password. 12. The use case ends. |
| Alternative Course of Action: | 5A. Email/Phone Not Found:  If the entered email or phone number does not match any registered user, the system shows an error and prompts the user to retry.  7A. Email/SMS Fails to Send:  If the reset message cannot be delivered due to server issues, the system shows a message instructing the user to try again later or contact support.  9A. Passwords Do Not Match:  If the new password and confirmation do not match, the system prompts the user to re-enter them. |
| Postconditions: | 1. The user receives a password reset link or code via email or SMS. 2. The user is able to reset their password and log in again. 3. A password change confirmation is sent to the user’s registered contact. |

Table 2-3 Forget password

|  |  |
| --- | --- |
| Use Case ID: | UC04 |
| Use Case Name: | Log Out |
| Actors: | All Actors |
| Description: | This use case describes how a logged-in user securely ends their session and exits the Guest House Renting website system. |
| Preconditions: | 1. The user is currently logged in to the Guest House Renting website. 2. The user's session is active in a supported web browser. |
| Include: |  |
| Extend: |  |
| Basic Course of Action: | 1. The user clicks on the “Log Out” option from the website’s navigation bar or profile menu. 2. The system immediately ends the user's session. 3. All authentication tokens and temporary session data are invalidated and removed. 4. The user is redirected to the homepage or login page. 5. A confirmation message is displayed: “You have been successfully logged out.” 6. The use case ends. |
| Alternative Course of Action: | 2A. Session Timeout or Error:  If the session has already expired or was inactive too long, the system may automatically log out the user and display a message:  “Your session has expired due to inactivity. Please log in again.” |
| Postconditions: | 1. The user’s session is terminated. 2. All session-related data is cleared from the browser. 3. The user is redirected to the website's homepage or login page. |

Table 2-4 Log out

|  |  |
| --- | --- |
| Use Case ID: | UC05 |
| Use Case Name | Manage Guest House Account |
| Actors: | Super Admin |
| Description: | This use case details the functions available to the Super Admin for managing guest-house accounts on the website. The Super Admin may create new guest-house accounts, edit existing ones, delate accounts, or simply view account details. |
| Preconditions: | 1. The Super Admin already has a registered administrator account. 2. The Super Admin must be logged into the system. |
| Include: | Log In |
| Extend: |  |
| Basic Course of Action: | 1. The Super Admin logs into the system. 2. The Super Admin navigates to the Guest House Management module from the dashboard. 3. The Super Admin views a list of guest houses. 4. The Super Admin decides whether to create, edit, deactivate, or view a guest house account:   ➔ If creating a new guest house account, execute Sub-flow S-1.  ➔ If editing an existing guest house account, execute Sub-flow S-2.  ➔ If deactivating an account, execute Sub-flow S-3.  ➔ If viewing an account, execute Sub-flow S-4.   1. The system updates the account and displays a confirmation message. 2. Use case ends. |
| Sub-flow: | S-1: Create Guest House Account   1. Super Admin clicks on "Add New Guest House". 2. Super Admin enters required details such as:    - Guest House Name   - Location   - Contact Info   - Description and capacity   1. Super Admin submits the form. 2. System confirms creation and stores the new account.   S-2: Edit Guest House Account   1. Super Admin selects an existing guest house from the list. 2. Super Admin updates any necessary information. 3. Super Admin submits the changes. 4. System confirms updates and saves the modified details.   S-3: Deactivate Guest House Account   1. Super Admin selects an active guest house to deactivate. 2. Super Admin confirms the deactivation. 3. System disables the account and reflects it in the system status.   S-4: View Guest House Account   1. Super Admin selects a guest house to view. 2. System displays all stored information about that guest house. |
| Alternate Course of Action: | A6. System error occurs or validation fails (e.g., required fields are missing or invalid).  A7. The system prompts: "Guest house account could not be updated. Please check inputs."  A8. System discards unsaved changes.  A9. The use case ends. |
| Postcondition: | Guest house account for use in the system is either created, updated, viewed, or delete successfully. |

Table 2-5 Manage guest house account

|  |  |
| --- | --- |
| Use Case ID: | UC06 |
| Use Case Name: | Manage System Admin Account |
| Actors: | Super Admin |
| Description: | This use case outlines the process used by the Super Admin to manage accounts of System Administrators within the website. It includes creating new system admin accounts, editing their information, delating accounts, and viewing account details when needed. |
| Include: | Log In |
| Extend: |  |
| Basic Course of Action: | 1. The Super Admin logs into the system. 2. The Super Admin navigates to the System Admin Management module from the dashboard. 3. The Super Admin views a list of existing system admin accounts. 4. The Super Admin selects an action:   ➔ If creating a new system admin account, execute Sub-flow S-1.  ➔ If editing an existing system admin account, execute Sub-flow S-2.  ➔ If deactivating an account, execute Sub-flow S-3.  ➔ If viewing an account, execute Sub-flow S-4.   1. The system processes the selected action and displays a confirmation message. 2. Use case ends. |
| Sub-flow: | S-1: Create System Admin Account   1. Super Admin clicks “Add New Admin.” 2. Super Admin enters required details, such as:   Full Name  Email Address  Username and Password  Assigned Role/Permissions   1. Super Admin submits the form. 2. System validates input and creates a new admin account. 3. Confirmation is shown on screen.   S-2: Edit System Admin Account   1. Super Admin selects a system admin from the list. 2. Super Admin updates editable fields such as name, role, or credentials. 3. Super Admin submits the changes. 4. System saves the changes and displays a success message.   S-3: Deactivate System Admin Account   1. Super Admin selects an active admin account. 2. Super Admin clicks “Deactivate” and confirms the action. 3. System flags the account as inactive and prevents login.   S-4: View System Admin Account   1. Super Admin selects an admin account to view. 2. System displays detailed information for that account, including role and status. |
| Alternative Course of Action: | A6. If required fields are missing or invalid:  → The system displays an error and prompts for correction.  A7. If the selected account is already deactivated or not found:  → The system shows a warning and blocks the action.  A8. If a system/database error occurs:  → The system aborts the action, logs the issue, and notifies the Super Admin.  A9. If duplicate usernames or emails are entered:  → The system prevents creation and shows a duplication warning. |
| Postcondition: | System admin account is either created, updated, viewed, or marked as inactive and saved in the system accordingly. |

Table 2-6 Manage system admin account

|  |  |
| --- | --- |
| Use Case ID: | UC07 |
| Use Case Name: | Manage Staff Account |
| Actor: | System Admin |
| Description: | This use case details how a System Admin can manage Staff Accounts on the website-based Guest House Renting Management System. Actions include creating new staff accounts, editing existing ones, deactivating accounts, and viewing staff profiles. |
| Precondition: | 1. The System Admin must have a valid and active login account. 2. The System Admin is authenticated and authorized to access the Staff Management module. |
| Include: | Log In |
| Extend: |  |
| Basic Course of Action: | 1. The System Admin logs in to the system. 2. The System Admin navigates to the Staff Account Management section on the dashboard. 3. The System Admin views a list of existing staff accounts. 4. The System Admin chooses one of the available actions:   ➔ If creating a new staff account, execute Sub-flow S-1.  ➔ If editing an existing staff account, execute Sub-flow S-2.  ➔ If deactivating an account, execute Sub-flow S-3.  ➔ If viewing an account, execute Sub-flow S-4.   1. The system executes the requested operation and shows a confirmation message. 2. The use case ends. |
| Sub-flow: | S-1: Create Staff Account   1. The System Admin clicks "Add New Staff Account." 2. The System Admin enters necessary staff details, including:   Full Name  Username and Password  Assigned Role (e.g., Reception, Cashier, Supervisor)  Contact Information   1. The System Admin submits the form. 2. The system validates the input, creates the staff account, and shows a confirmation message.   S-2: Edit Staff Account   1. The System Admin selects a staff account to update. 2. The System Admin edits details such as role, contact info, or username. 3. The System Admin submits the changes. 4. The system updates the record and confirms the update.   S-3: Deactivate Staff Account   1. The System Admin selects an active staff account. 2. The System Admin clicks “Deactivate” and confirms the action. 3. The system marks the account as inactive and blocks future logins.   S-4: View Staff Account   1. The System Admin selects a staff account to view. 2. The system displays the staff member’s full profile and status. |
| Alternative Course of Action: | A6. If input validation fails (e.g., missing required fields):  → The system highlights the errors and prevents submission.  A7. If a duplicate username or email is detected:  → The system stops the operation and displays a conflict message.  A8. If the selected account no longer exists or is already deactivated:  → The system informs the user and cancels the action.  A9. If a server or database error occurs during the operation:  → The system logs the issue and notifies the System Admin of the failure. |
| Postcondition: | A staff account is successfully created, modified, deactivated, or viewed, and the system reflects the updated data. |

Table 2-7 Manage staff account

|  |  |
| --- | --- |
| Use Case ID: | UC08 |
| Use Case Name: | Manage Guest Account |
| Actor: | Reception |
| Description: | This use case explains how the Reception staff manages Guest Accounts through the website-based Guest House Renting Management System. This includes creating new guest records, updating existing profiles, deactivating accounts, or viewing guest details. |
| Precondition: | 1. The Reception staff must be logged into the system with a valid and active account. 2. The system is accessible and operational. |
| Include: | Log In |
| Extend: |  |
| Basic Course of Action: | 1. The Reception logs into the system. 2. The Reception navigates to the Guest Account Management section from the dashboard. 3. A list of registered guest accounts is displayed. 4. The Reception selects one of the management actions:   ➔ If creating a new guest account, execute Sub-flow S-1.  ➔ If editing an existing guest account, execute Sub-flow S-2.  ➔ If deactivating an account, execute Sub-flow S-3.  ➔ If viewing a guest profile, execute Sub-flow S-4.   1. The system performs the operation and displays a confirmation. 2. The use case ends. |
| Sub-flow: | S-1: Create Guest Account   1. Reception clicks on "Add New Guest." 2. Reception enters guest details including:   Full Name  Contact Information  Identification Details (e.g., ID/passport number)  Optional notes or preferences   1. Reception submits the form. 2. The system validates and stores the guest profile.   S-2: Edit Guest Account   1. Reception selects a guest profile from the list. 2. Reception updates contact info, preferences, or identification data. 3. Reception submits the changes. 4. The system confirms the update and saves the changes.   S-3: Deactivate Guest Account   1. Reception selects an active guest account. 2. Reception clicks "Deactivate" and confirms the action. 3. The system deactivates the account, restricting its use for future bookings.   S-4: View Guest Account   1. Reception selects a guest to view. 2. The system displays full profile details, booking history, and contact info. |
| Alternative Course of Action: | A6. Required fields are missing or invalid:  → The system prevents submission and highlights missing data.  A7. Duplicate contact or ID information:  → The system blocks submission and alerts the Reception.  A8. Attempting to modify a deactivated account:  → The system denies the action and prompts a warning message.  A9. Network or system error during submission:  → The system logs the error and notifies the Reception of failure. |
| Postcondition: | A guest account is successfully created, updated, viewed, or deactivated, and the system reflects the changes accordingly. |

Table 2-8 Manage guest account

|  |  |
| --- | --- |
| Use Case ID: | UC09 |
| Use Case Name: | Search Room Availability |
| Actors: | Guest, Reception |
| Description: | This use case describes how users (Guest or Reception) search for available rooms based on date, location, and guest house within the Guest House Renting website system. |
| Preconditions: | 1. The user (Guest or Reception) is logged into the system or is accessing the public room search interface. 2. The Guest House Renting website is accessible through a web browser. 3. Room data is already stored in the system, including availability status and booking calendar. |
| Include: | Log In |
| Extend: | Book Room |
| Basic Course of Action: | 1. The user navigates to the “Search Rooms” section on the website. 2. The user enters the required search filters:   Check-in and check-out dates  Location  Number of guests  Guest house (optional)   1. The system validates the search criteria (e.g., date format, future date, non-empty fields). 2. The system queries the database for rooms matching the criteria that are marked as available. 3. A list of available rooms is displayed with key details:   Room type and number  Price per night  Guest house name and address  Availability status   1. The user may click “Book Now” to continue to booking (UC10). 2. The use case ends. |
| Alternative Course of Action: | 3A – Invalid or incomplete search input:  If the user leaves required fields empty or uses invalid dates, the system displays an error message:  “Please enter valid check-in and check-out dates.”  4A – No matching rooms found:  If no rooms match the search filters, the system displays:  “No available rooms found for the selected dates and criteria. Please try different dates or filters.”  5A – System error or database timeout:  If the system fails to retrieve room data, it shows:  “Unable to retrieve room availability at this time. Please try again later.” |
| Postconditions: | 1. The system displays a list of available rooms that match the search criteria. 2. The user may proceed to book a room based on availability (see UC10: Book Room). |

Table 2-9 Search room availability

|  |  |
| --- | --- |
| Use Case ID: | UC10 |
| Use Case Name: | Book Room |
| Actors: | Guest, Reception |
| Description: | This use case describes how a Guest or Reception user books an available room on the Guest House Renting website system, including selecting room details, entering personal information (for Guests), and confirming the reservation. |
| Preconditions: | 1. The user has successfully searched for available rooms (via UC09: Search Room Availability). 2. The room selected for booking is currently available. 3. If the actor is a Guest, they must be logged in. 4. For Reception, they are logged into their authorized dashboard. |
| Include: | Log In |
| Extend: | Cancel Booking |
| Basic Course of Action: | 1. The user (Guest or Reception) selects an available room from the search results (UC09). 2. The system loads the Booking Details page for the selected room. 3. The user fills in or verifies the required booking information:   Check-in and check-out dates  Guest name and contact information  Special requests (optional)   1. The user confirms the reservation. 2. The system:   Validates the booking data  Reserves the room for the specified dates  Creates a new booking record   1. The system displays a booking confirmation page with booking ID and summary. 2. Optionally, the system redirects to UC18: Process Payment / Pay. 3. The use case ends. |
| Alternative Course of Action: | 3A – Required fields missing or invalid:  The system displays an error message:  “Please complete all required fields to continue with the booking.”  4A – Room becomes unavailable during booking:  If another user books the room at the same time, the system notifies:  “The selected room is no longer available. Please choose a different room.”  5A – System failure or timeout:  If the database or reservation engine fails, the system returns:  “Booking could not be completed due to a system error. Please try again later.” |
| Postconditions: | 1. A new booking record is created and stored in the system. 2. The selected room is marked as “Booked” for the chosen dates. 3. The guest receives a confirmation message or receipt. |

Table 2-10 Book room

|  |  |
| --- | --- |
| Use Case ID: | UC11 |
| Use Case Name: | Cancel Booking |
| Actors: | Guest |
| Description: | This use case allows a logged-in Guest to cancel an existing room booking they have previously made, according to the cancellation policy defined by the Guest House Renting website system. |
| Preconditions: | 1. The user is logged in to their Guest account (via UC02: Log In). 2. The user has at least one active booking in the system. 3. The booking to be canceled is still within the allowed cancellation window (if applicable). |
| Include: | Log In, View Booking History |
| Extend: | Issue Refund |
| Basic Course of Action: | 1. The Guest logs into their account via UC02. 2. The Guest navigates to the “My Bookings” or “Booking History” section. 3. The system displays a list of the Guest’s current and past bookings. 4. The Guest selects an active booking to cancel. 5. The Guest clicks the “Cancel Booking” button. 6. The system displays a confirmation message and any applicable cancellation policies or fees. 7. The Guest confirms the cancellation. 8. The system:   Marks the booking as “Canceled”  Frees up the room for others to book  Triggers UC19: Issue Refund if eligible   1. A success message is displayed: “Your booking has been successfully canceled.” 2. The use case ends. |
| Alternative Course of Action: | 4A – Booking already expired or completed:  The system notifies:  “This booking has already passed and cannot be canceled.”  5A – Cancellation not allowed due to policy (e.g., too late):  The system displays:  “This booking cannot be canceled as it falls outside the allowed cancellation window.”  8A – System error during cancellation:  The system notifies the user:  “An error occurred while canceling your booking. Please try again later or contact support.” |
| Postconditions: | 1. The selected booking is marked as “Canceled” in the system. 2. The room becomes available again for future bookings. 3. If eligible, a refund process may be initiated (see UC19: Issue Refund). |

Table 2-11 Cancel booking

|  |  |
| --- | --- |
| Use Case ID: | UC12 |
| Use Case Name: | View Booking History |
| Actors: | Guest |
| Description: | This use case enables a Guest to view all of their past and upcoming room bookings through their account on the Guest House Renting website. |
| Preconditions: | 1. The user is logged into a valid Guest account (via UC02: Log In). 2. The user has made at least one booking in the system. |
| Include: | Log In |
| Extend: | Cancel Booking |
| Basic Course of Action: | 1. The Guest logs into their account through the website (via UC02). 2. From the dashboard or profile menu, the Guest navigates to “My Bookings” or “Booking History.” 3. The system fetches and displays a list of all bookings made by the Guest, categorized by:   Upcoming bookings  Past bookings  Canceled bookings   1. Each booking entry includes key details such as:   Booking ID  Guest House Name  Room Type  Check-in / Check-out dates  Booking status   1. The Guest can click on a booking to view full details, including:   Price paid  Payment status  Cancellation options (if available)   1. The use case ends. |
| Alternative Course of Action: | 3A – No bookings found:  The system displays:  “You have not made any bookings yet.”  Optionally, the user is offered a button to Search Rooms (UC09).  4A – System error or failed data fetch:  The system shows:  “Unable to load booking history. Please try again later.” |
| Postconditions: | 1. A list of the Guest’s bookings is displayed, including booking status (e.g., confirmed, canceled, completed). 2. The Guest can view detailed information for each booking. |

Table 2-12 View booking history

|  |  |
| --- | --- |
| Use Case ID: | UC13 |
| Use Case Name: | Check In Guest |
| Actors: | Reception |
| Description: | This use case allows the Receptionist to officially check in a guest upon their arrival at the guest house, confirm their booking, and update the room status in the system. |
| Preconditions: | 1. The Receptionist is logged into the system (via UC02: Log In). 2. The Guest has a valid, confirmed booking in the system. 3. The check-in date matches today’s date or is within the allowable check-in window. |
| Include: | Log In, Book Room |
| Extend: | Report Room Issues |
| Basic Course of Action: | 1. The Receptionist logs into the system through the admin dashboard. 2. From the dashboard, the receptionist navigates to the “Today’s Check-ins” or “Bookings” section. 3. The receptionist searches for the Guest’s booking using name, booking ID, or check-in date. 4. The system displays the guest's booking information. 5. The receptionist verifies the guest's identity and booking details. 6. The receptionist clicks the “Check In” button. 7. The system updates the guest’s status to “Checked In” and marks the room as “Occupied.” 8. A confirmation screen is shown, and optionally, a printable or digital check-in receipt is generated. 9. The use case ends. |
| Alternative Course of Action: | 3A – Booking not found or invalid:  The system displays:  “No valid booking found for this guest. Please verify the information.”  5A – Guest arrives earlier than the allowed check-in time:  The system alerts:  “This guest is attempting to check in before the official check-in time.”  The receptionist can choose to override this restriction if permitted.  7A – Room not ready (e.g., not cleaned or under maintenance):  The system blocks the check-in and displays:  “The assigned room is not ready for check-in. Please assign a different room or wait until the room is available.”  The receptionist may choose to assign another room or notify the guest. |
| Postconditions: | 1. The guest is marked as “checked in.” 2. The booked room's status is updated to “Occupied.” 3. A digital or printed check-in confirmation is available if required. |

Table 2-13 Check in guest

|  |  |
| --- | --- |
| Use Case ID: | UC14 |
| Use Case Name: | Check Out Guest |
| Actors: | Reception |
| Description: | This use case allows the Receptionist to complete a guest's stay by processing their check-out. It involves updating the guest's status, ensuring the room is marked for cleaning or available, and confirming that any outstanding payments have been addressed. |
| Preconditions: | 1. The Receptionist is logged into the system (via UC02: Log In). 2. The Guest has previously checked in and has an active booking status. 3. All services used during the stay are recorded and available for review. 4. Payment status is either already cleared or pending final processing. |
| Include: | Log In |
| Extend: | Assign Room Cleaning Task |
| Basic Course of Action: | 1. The Receptionist logs into the system. 2. They navigate to the “Current Stays” or “Check Out” section of the dashboard. 3. The guest is identified using their booking ID, room number, or name. 4. The system displays the guest’s stay summary, including dates, services used, and payment history. 5. The receptionist verifies that all dues are settled.   If payment is pending, they redirect to UC18: Process Payment.   1. Once cleared, the receptionist clicks the “Check Out” button. 2. The system updates the guest’s status to “Checked Out” and marks the room as “Needs Cleaning.” 3. The guest is thanked, and a check-out receipt is optionally printed or emailed. 4. The use case ends. |
| Alternative Course of Action: | 3A – Guest record not found or invalid:  System shows:  “Guest check-in record not found. Please verify the booking ID or name.”  5A – Unpaid dues detected:  System alerts:  “Outstanding balance detected. Please settle the bill before proceeding.”  Receptionist must ensure payment is completed via UC18.  7A – Room status fails to update (system error):  System shows:  “Unable to update room status. Please try again or contact technical support.”  Receptionist retries or logs the issue manually. |
| Postconditions: | 1. The guest is officially checked out of the system. 2. The room is marked as “Needs Cleaning” or queued for a new booking. 3. Final payment and check-out receipt are recorded and stored in the system. |

Table 2-14 Check out guest

|  |  |
| --- | --- |
| Use Case ID: | UC15 |
| Use Case Name: | Assign Room Cleaning Task |
| Actors: | Supervisor |
| Description: | This use case describes how the Supervisor assigns cleaning duties to housekeeping staff after a guest checks out or when a room is flagged as needing cleaning. It ensures the room is cleaned and ready for the next occupant. |
| Preconditions: | 1. The Supervisor is logged into the system (via UC02: Log In). 2. At least one room has been flagged as “Needs Cleaning.” 3. Housekeeping staff members are registered in the system and available for task assignment. |
| Include: | Log In, Check Out Guest |
| Extend: |  |
| Basic Course of Action: | 1. The Supervisor logs into the system. 2. They navigate to the “Room Cleaning Tasks” module on the dashboard. 3. The system displays a list of all rooms flagged as “Needs Cleaning.” 4. The Supervisor selects a room and assigns it to an available housekeeping staff member. 5. The Supervisor sets the cleaning priority (e.g., urgent, normal) and expected completion time. 6. The system notifies the assigned staff member and updates the task list. 7. The room’s cleaning status is set to “In Progress.” 8. The use case ends. |
| Alternative Course of Action: | 3A – No rooms listed as needing cleaning:  System displays:  “All rooms are currently clean. No pending tasks available.”  Supervisor may exit or review status.  4A – No housekeeping staff available or online:  System displays:  “No staff currently available. Please try again later or contact HR.”  Supervisor can assign later or manually track.  6A – Notification fails:  System shows:  “Task assigned, but staff notification failed. Please inform the staff manually.” |
| Postconditions: | 1. The room is officially assigned for cleaning. 2. The housekeeping staff member is notified and begins cleaning. 3. The room’s status is updated in the system as “In Progress” until cleaning is completed. |

Table 2-15 Assign room cleaning task

|  |  |
| --- | --- |
| Use Case ID: | UC16 |
| Use Case Name: | Update Room Status |
| Actors: | Supervisor |
| Description: | This use case allows the Supervisor to update the current status of a room (e.g., cleaned, occupied, under maintenance) after inspection or maintenance follow-up. Keeping room statuses up to date ensures accurate availability for bookings and operational planning. |
| Preconditions: | 1. The supervisor must be logged into the system via a valid account. 2. The room being updated must already exist in the system. |
| Include: | Log In |
| Extend: |  |
| Basic Course of Action: | The Supervisor logs in to the Guest House Renting System via UC02: Log In.  From the dashboard, the Supervisor navigates to the “Room Management” section.  The system displays a list of rooms with their current statuses.  The Supervisor selects a specific room to update.  The Supervisor chooses a new status from a predefined list:  Cleaned  Occupied  Under Maintenance  Ready for Check-In  The Supervisor submits the status update.  The system updates the room’s status and displays a success confirmation.  The use case ends. |
| Alternative Course of Action: | 4A. Invalid Room Selection:  - If the selected room ID does not exist or is unavailable, the system shows an error message:   “Selected room not found. Please try again.”  6A. Invalid or Empty Status:  - If no status is selected or the input is invalid, the system prompts:   “Please select a valid room status before submitting.”  7A. System Error or Database Failure:   - If an internal error occurs, the system shows:    “Room status update failed. Please try again later.” |
| Postconditions: | 1. The updated room status is saved in the system database. 2. The new room status is reflected for receptionists and managers for operational awareness. 3. Availability updates take effect immediately (if status affects booking). |

Table 2-16 Update room status

|  |  |
| --- | --- |
| Use Case ID: | UC17 |
| Use Case Name: | Report Room Issues |
| Actors: | Supervisor |
| Description: | This use case enables the Supervisor to report problems or maintenance issues found in rooms (e.g., broken fixtures, unclean conditions, plumbing issues). These reports ensure timely resolution by alerting the management and maintenance teams. |
| Preconditions: | 1. The supervisor must be logged into the system using valid credentials. 2. The room for which an issue is being reported must exist in the system. |
| Include: | Log In |
| Extend: |  |
| Basic Course of Action: | 1. The Supervisor logs in to the system via UC02: Log In. 2. The Supervisor navigates to the “Room Issue Reporting” section from the dashboard. 3. The system displays a list of rooms assigned to the supervisor. 4. The Supervisor selects the room with the issue. 5. A reporting form appears where the Supervisor fills in:   Issue Type (e.g., Plumbing, Electrical, Cleanliness, Furniture, etc.)  Description of the issue  Priority Level (Low, Medium, High)   1. The Supervisor submits the report. 2. The system stores the report and notifies the Manager for review. 3. A confirmation message is shown: “Issue reported successfully.” 4. The use case ends. |
| Alternative Course of Action: | 4A. Room not assigned or not available:  - The system shows: “You do not have permission to report issues for this room.”  5A. Missing required fields:  - If the form is incomplete, the system prompts: “Please fill in all required fields before submitting.”  7A. Notification failure or backend error:  - If the system fails to notify the manager, it logs the report but shows:   “Issue reported, but notification failed. Please inform the manager manually.” |
| Postconditions: | 1. The room issue report is saved and associated with the selected room. 2. The report is viewable by both the Manager and Receptionist (if needed). 3. The room’s status may be flagged automatically for maintenance. |

Table 2-17 Report room issues

|  |  |
| --- | --- |
| Use Case ID: | UC18 |
| Use Case Name: | Process Payment / Pay |
| Actors: | Guest |
| Description: | This use case allows a Guest to complete payment for a booking using available payment methods such as credit card, debit card, mobile payment, or online banking. The system securely processes the transaction and confirms the payment. |
| Preconditions: | 1. The guest must be logged into the system with a valid account. 2. A confirmed booking must exist for the guest. 3. The payment gateway must be operational. |
| Include: | Log In, Book Room |
| Extend: | Verify Payment Status |
| Basic Course of Action: | 1. The guest logs in using UC02: Log In. 2. The guest navigates to the “My Bookings” section. 3. The system displays the list of unpaid or pending bookings. 4. The guest selects a booking and clicks “Pay Now.” 5. The system shows available payment options. 6. The guest chooses a payment method and enters required payment details. 7. The guest confirms the payment. 8. The system processes the payment through a secure payment gateway. 9. Upon success, the system updates the booking status to “Paid”. 10. A confirmation message and digital receipt are displayed and/or emailed to the guest. 11. The use case ends. |
| Alternative Course of Action: | 3A. No pending bookings:  - The system shows: “No bookings available for payment.”  6A. Invalid or missing payment information:  - The system prompts the guest to correct the form:   “Please check your card number or payment details.”  8A. Payment failed due to technical error or declined transaction:  - The system shows: “Payment failed. Please try again with another method.”  9A. Partial payment not allowed:  - If the guest attempts a partial payment, the system prompts:   “Full payment is required to complete the booking.” |
| Postconditions: | 1. The booking status is updated to Paid. 2. The guest receives a digital receipt via email and/or on-screen. 3. The payment record is stored in the system and linked to the guest’s account. |

Table 2-18 Process payment/ pay

|  |  |
| --- | --- |
| Use Case ID: | UC19 |
| Use Case Name: | Issue Refund |
| Actors: | Cashier |
| Description: | This use case describes how the Cashier handles refund requests from guests due to booking cancellations, system errors, or service issues. The process ensures that valid refund claims are verified and processed back to the guest’s original payment method. |
| Preconditions: | 1. The guest must have completed a payment (UC18). 2. A refund request must be logged or triggered through cancellation or admin action. 3. The Cashier must be logged in with valid credentials. 4. The booking must be eligible for a refund based on the refund policy. |
| Include: | Log In, Cancel Booking |
| Extend: | Verify Payment Status |
| Basic Course of Action: | 1. The Cashier logs into the system. 2. The Cashier navigates to the “Payments” or “Refund Requests” section. 3. The system displays a list of eligible refund requests or canceled bookings. 4. The Cashier selects a booking and reviews the payment and cancellation status. 5. The Cashier clicks “Issue Refund.” 6. The system prompts the Cashier to confirm the amount and refund method. 7. The Cashier confirms and submits the refund request. 8. The system processes the refund via the original payment channel (e.g., card, mobile). 9. A refund confirmation message is displayed. 10. The system logs the refund transaction and notifies the guest. 11. The use case ends. |
| Alternative Course of Action: | 3A. No refund-eligible bookings found:  – The system displays: “No bookings eligible for refund.”  6A. Refund amount does not match policy:  – The system blocks submission and shows:    “Refund amount must match policy guidelines.”  8A. Payment channel unavailable or refund fails:  – The system displays: “Unable to process refund at this time. Please try again later.”  4A. Booking flagged as fraudulent or disputed:  – The system prevents refund and alerts admin. |
| Postconditions: | 1. Refund is issued successfully and recorded in the system. 2. Guest is notified via email or system alert. 3. Refund details are added to the guest’s payment history. |

Table 2-19 Issue refund

|  |  |
| --- | --- |
| Use Case ID: | UC20 |
| Use Case Name: | Verify Payment Status |
| Actors: | Cashier |
| Description: | This use case outlines how the Cashier verifies whether a guest has successfully completed payment for a booking. This verification is essential before check-in, refunds, or other financial processes can proceed. |
| Preconditions: | 1. The guest must have made a booking that requires payment. 2. The Cashier is logged into the system with proper authorization. 3. The payment gateway or payment records system is online and accessible. |
| Include: | Log In |
| Extend: | Issue Refund, Check in Guest, Process Payment |
| Basic Course of Action: | 1. The Cashier logs into the system using UC02: Log In. 2. The Cashier navigates to the “Verify Payment” or “Booking Management” section. 3. The Cashier enters or selects the booking ID, guest ID, or payment transaction number. 4. The system retrieves and displays the payment status:    1. - Paid    2. - Pending    3. - Failed    4. - Refunded 5. If the status is “Paid”, the cashier proceeds with any necessary next step (e.g., issuing receipt, check-in, or refund). 6. The use case ends. |
| Alternative Course of Action: | 3A. Invalid or missing booking ID entered:  – System displays: “Booking not found. Please check the ID and try again.”  4A. Payment status cannot be retrieved due to gateway error:  – System displays: “Unable to retrieve payment status. Try again later or contact support.”  4B. Payment shows as “Pending” or “Failed”:  – System blocks further action and notifies cashier: “Payment incomplete. Please confirm before proceeding.” |
| Postconditions: | 1. The system displays the verified payment status for the selected booking. 2. The cashier can take appropriate next steps (e.g., proceed with refund, check-in, or flag payment issue). 3. The action is logged in the system audit trail. |

Table 2-20 Verify payment status

|  |  |
| --- | --- |
| Use Case ID: | UC21 |
| Use Case Name: | View Payment History |
| Actors: | Guest |
| Description: | This use case describes how a Guest can view a history of all payments made through the Guest House Renting Management website. This helps users track transactions related to bookings, cancellations, and refunds. |
| Preconditions: | 1. The guest has a registered account on the platform. 2. The guest is logged into the system. 3. There are payment records linked to the guest’s account. |
| Include: | Log In |
| Extend: | Book Room, Cancel Booking, Issue Refund |
| Basic Course of Action: | 1. The Guest logs into their account using UC02: Log In. 2. From the dashboard, the Guest navigates to “My Payments” or “Payment History.” 3. The system retrieves a list of all payments associated with the guest’s account. 4. The system displays payment details, including:    1. - Booking ID    2. - Payment Date    3. - Amount Paid    4. - Payment Method    5. - Status (Paid / Refunded / Failed) 5. The Guest may click on a specific transaction to view more details. 6. The use case ends. |
| Alternative Course of Action: | 3A. No payment records found:  – System displays: “No payment history available.”  – Optionally, it may offer help or suggestions like: “Have you made any bookings yet?”  4A. System error during retrieval:  – System shows an error: “Unable to load payment history. Please try again later.” |
| Postconditions: | 1. The guest successfully views past payment records. 2. Optionally, the guest can download receipts or contact support regarding specific transactions. 3. The action is recorded in the system’s activity logs for auditing purposes. |

Table 2-21 View payment history

|  |  |
| --- | --- |
| Use Case ID: | UC22 |
| Use Case Name: | View Booking Reports |
| Actors: | Manager |
| Description: | This use case describes how a Manager can access and review detailed booking reports. These reports provide insights into occupancy trends, booking frequency, guest preferences, and seasonal patterns to support informed decision-making. |
| Preconditions: | 1. The Manager has a valid registered account. 2. The Manager is logged into the system. 3. Booking data exists in the system. |
| Include: | Log In |
| Extend: | Book Room, View Booking History, Assign Room Cleaning Task |
| Basic Course of Action: | 1. The Manager logs into the system using UC02: Log In. 2. The Manager navigates to the “Reports” section from the dashboard. 3. The Manager selects “Booking Reports” from the available report types. 4. The system presents filters and options (e.g., date range, guest house, room type, etc.). 5. The Manager applies filters and submits the query. 6. The system generates a report based on the selected parameters. 7. The system displays summarized data such as:    * Total bookings by date    * Occupancy rate per room or guest house    * Guest types (local/international, returning/new)    * Booking sources (website, staff-assisted) 8. The Manager can export or print the report if needed. 9. The use case ends. |
| Alternative Course of Action: | 6A. No data available for selected filters:  – System displays a message: “No bookings found for the selected criteria.”  7A. System error during report generation:  – System displays: “Unable to load booking report. Please try again later.” |
| Postconditions: | 1. The Manager successfully views detailed booking insights. 2. The report may be used for operational or strategic decisions. 3. An exportable record of the report is available for archiving or further analysis. |

Table 2-22 View booking reports

|  |  |
| --- | --- |
| Use Case ID: | UC23 |
| Use Case Name: | Manage Pricing & Room Types |
| Actors: | Manager |
| Description: | This use case outlines how the Manager can update pricing structures, define or modify room types, and apply promotional rates. This ensures competitive pricing and accurate representation of rooms on the Guest House Renting Management System website. |
| Preconditions: | 1. The Manager is a registered and authenticated user. 2. The system contains existing room types and pricing data. 3. There are rooms assigned to a guest house managed under the system. |
| Include: | Log In |
| Extend: | Search Room Availability, Book Room, Send Notifications |
| Basic Course of Action: | 1. The Manager logs into the system using UC02: Log In. 2. The Manager navigates to the “Room Management” or “Pricing” section. 3. The system displays a list of room types and their current prices. 4. The Manager selects an action:   ➔ To Add New Room Type:   Clicks on "Add Room Type"  Enters type name, description, capacity, amenities  Saves the new room type  ➔ To Edit Room Type:  Selects an existing type  Updates details such as description or features  Saves changes  ➔ To Update Pricing:  Selects a room type  Inputs new base price, seasonal price, or promo rate  Sets duration (if promo) and submits   1. The system updates the data and displays a success confirmation. 2. The use case ends. |
| Alternative Course of Action: | 4A. Missing or invalid input:  – System highlights required fields and prompts: “Please fill all required fields correctly.”  5A. Database error or update conflict:   – System shows: “Unable to save changes. Please try again or contact admin.” |
| Postconditions: | 1. Room types and pricing information are updated in the system. 2. Any changes reflect in the booking interface and search results. 3. Promotional changes may trigger automated user notifications. |

Table 2-23 Manage room types and pricing

|  |  |
| --- | --- |
| Use Case ID: | UC24 |
| Use Case Name: | Backup Data |
| Actors: | System |
| Description: | This use case defines how the System automatically performs regular data backups, including user data, booking records, payments, and guest house information, to ensure data integrity and recovery in case of failure or system crash. |
| Preconditions: | 1. The system is running and scheduled backup configuration is enabled. 2. Storage destination (local or cloud) is properly configured. 3. There is active internet connectivity (for cloud backups). |
| Include: |  |
| Extend: | Send Notifications, View Room Issue Reports |
| Basic Course of Action: | 1. At a scheduled interval (e.g., daily at midnight), the system initiates the backup process. 2. The system collects and compiles all key data from the database, including:    * User accounts and roles    * Booking records    * Payment transactions    * Guest house and room configurations    * System logs and reports 3. The system compresses and encrypts the data archive. 4. The backup is saved to the configured storage location (e.g., remote server or cloud service). 5. A success log is created, and optionally, the Super Admin is notified. 6. Use case ends. |
| Alternative Course of Action: | 2A. Data access error:  – System logs error and retries once before aborting.  4A. Storage not available or out of space:  – System logs failure and sends alert to the Super Admin:   “Backup failed: Insufficient storage space or network issue.”  5A. Partial backup or corruption detected:  – System discards the file and retries or schedules next attempt with alert. |
| Postconditions: | 1. A secure and up-to-date backup is created and stored. 2. Backup integrity is logged for auditing. 3. Super Admin is notified of success or failure, if configured. |

Table 2-24 Backup data

|  |  |
| --- | --- |
| Use Case ID: | UC25 |
| Use Case Name: | Send Notifications |
| Actors: | System |
| Description: | This use case describes how the System automatically sends various notifications to users (e.g., Guest, Reception, Manager, Super Admin) regarding events like bookings, payments, cancellations, feedback, account changes, promotions, or system updates. |
| Preconditions: | 1. The user has a registered account with valid contact information (e.g., email or phone number). 2. Notification settings are properly configured in the system. 3. Triggering event (e.g., booking confirmation, payment received) has occurred. |
| Include: |  |
| Extend: | Book Room, Cancel Booking, Process Payment / Pay, Issue Refund, Backup Data, Manage Pricing & Room Types |
| Basic Course of Action: | 1. A triggering event occurs in the system (e.g., new booking created). 2. The system retrieves the relevant user and their notification preferences. 3. The system generates a message based on the type of event, including:    * Event title (e.g., "Booking Confirmed", "Payment Successful")    * Date and time    * Action details (e.g., room type, amount paid) 4. The message is sent through the appropriate channel (e.g., email, SMS, in-app). 5. The system logs the notification delivery status. 6. The use case ends. |
| Alternative Course of Action: | 3A. Message template missing or invalid:  – The system uses a default fallback message or logs an error.  4A. Delivery fails due to network issue or invalid contact:  – The system retries sending once.  – If it fails again, an error is logged, and the notification is queued for later. |
| Postconditions: | 1. The user is informed about system events or actions through their preferred notification channel. 2. Notification delivery is logged for auditing and troubleshooting. |

Table 2-25 Send Notification

|  |  |
| --- | --- |
| Use Case ID: | UC26 |
| Use Case Name: | View Room Issue Reports |
| Actor: | Manager |
| Description: | This use case allows the Manager to view and monitor all reported issues or maintenance problems related to guest house rooms. It supports better decision-making and ensures that room-related issues are tracked and resolved efficiently. |
| Preconditions: | 1. The Manager is successfully logged into the Guest House Renting website system. 2. At least one issue has been reported via UC17: Report Room Issues. |
| Include: | Log In |
| Extend: | Report Room Issues |
| Basic Course of Action: | 1. The Manager logs into the system using UC02: Log In. 2. The Manager navigates to the “Room Issue Reports” section from the dashboard. 3. The system displays a list of all reported issues related to rooms. 4. The Manager filters the list based on various criteria:   Room number  Issue type (e.g., plumbing, electrical, cleaning)  Report date  Status (e.g., unresolved, in progress, resolved)   1. The Manager clicks on a specific issue report to view more details. 2. The system shows full details of the selected issue, including:   Description of the issue  Date and time of report  Name of the reporting Supervisor  Current resolution status  Any attached photos or notes   1. The Manager can choose to download or export the report. 2. The use case ends. |
| Alternative Course of Action: | 3A. No issue reports found:  – The system displays a message: “No room issues have been reported.”  5A. Error loading selected report:  – The system shows: “Unable to load report details. Please try again later.”  – The Manager may return to the issue list or refresh the page. |
| Postconditions: | 1. The Manager successfully views one or more room issue reports. 2. Room issue information is accessible for review, analysis, or maintenance planning. |

Table 2-26 View room issue reports

* 1. **Sequence diagram**

One kind of UML (Unified Modeling Language) interaction diagram that shows how objects interact in a particular situation or use case is the sequence diagram. It shows the order in which messages are sent and received by various objects over time.

The essential components of a sequence diagram are broken down as follows:

**Lifelines:** The objects involved in the interaction are represented by these vertical lines. Every object in the diagram has a distinct name and a lifeline that indicates its presence during the scenario.

**Messages:** These are requests or messages conveyed from one object to another represented by horizontal arrows drawn between lifelines. Labels on arrows usually indicate the content of the message (data or function call).

**Activation Bars:** Lifelines are marked with narrow rectangles that show how long an object is actively.