Hotel Management System [HMS] Requirements Specification

Version 3.0

March 8th, 2025

Use this Requirements Specification template to document the requirements for your product or service, including priority and approval. Tailor the specification to suit your project, organizing the applicable sections in a way that works best, and use the checklist to record the decisions about what is applicable and what isn't.

The format of the requirements depends on what works best for your project.

This document contains instructions and examples which are for the benefit of the person writing the document and should be removed before the document is finalized.

To regenerate the TOC, select all (CTL-A) and press F9.

Table of Contents

1. Executive Summary

1.1 Project Overview

The Hotel Management System [HMS] is a management system designed to streamline the operations of a luxury hotel. This luxury hotel contains all of the normal services that every hotel is supposed to have but extending its capabilities by adding premium features such as casino, gym, restaurant and bar facilities.

2. Product/Service Description

2.1 Product Context

HMS is designed to be a mainly standalone system, but for a more up-to-date way of operating a hotel it includes external platforms such as:

Third-Party Integrations:

Online Travel Agencies (OTAs) for real-time room availability sync (e.g., Booking.com).

Payment Gateways (e.g., Stripe, PayPal) to process guest payments.

POS Systems for restaurants/bars to log orders and link charges to guest bills.

Mobile Guest Portal for self-service bookings, modifications, and service requests.

Digital keys generated by **Flexipass SDK**

The system as a whole is designed with these core modules in mind:

Core Modules and Functionalities:

1. **Booking Management Module**

   Purpose: Room reservations, modifications and cancellations.

   Workflow:

   - Guests search for rooms via OTAs or the Mobile Guest Portal.
   - The system checks real-time availability (synced with OTAs) and allows booking.
   - A unique BookingID is generated and stored in the database.

   Integration:

   - Syncs with OTAs for real-time availability updates.
   - Links to the billing module for payment processing.

2. **Check-In / Check Out Module**

   Purpose: Manages guest arrivals and departures.

   Workflow:

   - Receptionists verify guest details and assign rooms..
   - The system triggers the legacy keycard system to generate access.
   - At checkout, the system finalizes billing, , and updates room status.

   Integration:

   - Flexipass will handle the activation and deactivation of the digital keys for the rooms of the hotel.
   - Pulls data from the billing module for final payments.

The HMS is designed to cater to a diverse set of users (customers and staff), each with unique needs and technical expertise. Below are the general customer profiles:

User Characteristics

- *Guests:*

  Profile: Guests are the primary users of the system, interacting with it to manage their stay, access services, and handle payments. They may have varying levels of comfort with technology, so the interface must be intuitive and user-friendly.
  Experience: Varies from tech-savvy to novice.
  Technical Expertise: Basic.
  Other Characteristics: Use the mobile app for booking rooms, making service requests, and viewing bills.

- *Receptionists:*

  Profile: Receptionists are front-line staff who interact with the system daily to manage guest check-ins, check-outs, and payments. They require a system that is efficient and easy to navigate to ensure smooth operations.
  Experience: Familiar with hotel operations and customer service.
  Technical Expertise: Moderate; comfortable using software for reservations, billing, and room management.
  Other Characteristics: Verify guest IDs, assign rooms, process payments, and generate invoices.

- *Housekeeping Staff:*

Profile: Housekeeping staff use the system to update the status of rooms. They need a straightforward and minimalistic interface to quickly log room conditions without requiring advanced technical skills.
Experience: Skilled in cleaning operations.
Technical Expertise: Low (basic UI interaction).
Other Characteristics: Update room status (clean/occupied) via a simplified interface.

- *Hotel Managers:*

Profile: Managers of a department that use the system to oversee their own department's operations, analyze performance, and make strategic decisions. They need access to detailed analytics, reporting tools, and configuration options to optimize hotel operations. Oversee hotel operations, staff management, and financial reporting.
Experience: Experienced in hotel management and decision-making.
Technical Expertise: Moderate (analytics and reports).
Other Characteristics: Configure pricing, view occupancy and revenue reports, and manage promotions..

- *General Manager:*
  Profile: The General Manager oversees all of the departments. The General Manager manages the whole staff and is responsible for the whole operation and the important decisions.
  Experience: Experienced in hotel management and decision-making and leadership.
  Technical Expertise: Moderate (analytics and reports).
  Other Characteristics: Full access to every metric and employee data.

- *Facility Staff (Casino, Gym, Restaurant, Bar):*

  Profile: Facility staff interact with the system to check his own work schedule and verify guest access to specific amenities and log service usage. They require a simple and efficient tool to scan Booking IDs and update records.
  Experience: Operate gyms, casinos, or restaurants.
  Technical Expertise: Low (scan/verify Booking IDs).
  Other Characteristics: Require real-time updates on guest eligibility and usage charges.

- *Administrators:*

  Profile: Administrators are responsible for maintaining the system, ensuring security, and managing user access. They require advanced tools for technical configuration, monitoring, and troubleshooting to keep the system running smoothly.
  Experience: High (technical configuration).
  Technical Expertise: High (technical configuration).
  Other Characteristics: Manage user accounts, permissions, audit logs, and system backups.
  .

2.2 Assumptions

- **Internet Connectivity:** Staff and guests have reliable internet access for real-time system use.
- **Device Availability:** Staff have access to devices (PCs/tablets) to interact with the system.
- **Third-Party Systems**: OTAs, POS, and payment gateways are operational and accessible.
- **Staff Training:** Users receive basic training to navigate their assigned modules.
- **Guest Compliance:** Guests provide valid IDs and adhere to hotel policies during check-in.

2.3 Constraints and Dependencies

Technical Constraints:

- Must comply with GDPR/CCPA for guest data privacy.
- Role-based access control (RBAC) to restrict unauthorized actions (e.g., housekeeping cannot modify bills).

Dependencies:

Integration Dependencies:

- Payment processing requires functional third-party gateways.

- Real-time room availability depends on OTA sync.

Module Dependencies:

- Booking module must be completed before check-in/checkout workflows.
- Billing module requires integration with POS systems.
- Legacy Systems: Must operate alongside existing security systems (e.g., keycard access).

3. Requirements

- Describe all system requirements in enough detail for designers to design a system satisfying the requirements and testers to verify that the system satisfies requirements.
- Organize these requirements in a way that works best for your project. See Appendix DAppendix D, Organizing the Requirements for different ways to organize these requirements.
- Describe every input into the system, every output from the system, and every function performed by the system in response to an input or in support of an output. (Specify what functions are to be performed on what data to produce what results at what location for whom.)
- Each requirement should be numbered (or uniquely identifiable) and prioritized.
  See the sample requirements in Functional Requirements, and System Interface/Integration, as well as these example priority definitions:

  Priority Definitions

  The following definitions are intended as a guideline to prioritize requirements.

  - Priority 1 – The requirement is a “must have” as outlined by policy/law
  - Priority 2 – The requirement is needed for improved processing, and the fulfillment of the requirement will create immediate benefits
  - Priority 3 – The requirement is a “nice to have” which may include new functionality

  It may be helpful to phrase the requirement in terms of its priority, e.g., "The value of the employee status sent to DIS **must be** either A or I" or "It **would be nice** if the application warned the user that the expiration date was 3 business days away". Another approach would be to group requirements by priority category.
- A good requirement is:
  - Correct
  - Unambiguous (all statements have exactly one interpretation)
  - Complete (where TBDs are absolutely necessary, document why the information is unknown, who is responsible for resolution, and the deadline)
  - Consistent
  - Ranked for importance and/or stability
  - Verifiable (avoid soft descriptions like “works well”, “is user friendly”; use concrete terms and specify measurable quantities)
  - Modifiable (evolve the Requirements Specification only via a formal change process, preserving a complete audit trail of changes)
  - Does not specify any particular design
  - Traceable (cross-reference with source documents and spawned documents).

3.1 Functional Requirements

Req#	Requirement	Comments	Priority	Date Rvwd	SME Reviewed / Approved	System Specifications
	The system shall allow the guests to search for available rooms in by room type, check-in and check-out dates and price range.		1			The system shall receive as input the guest criterias(room type, check-in, check-out dates and price range). After receiving the input, the system validate the entered check-in and check-out dates ensuring that they are logical (check-in before check-out). After validating the system should display the filtered room inventory and it should display to the user the room inventory result.

Req#	Requirement	Comments	Priority	Date Rvwd	SME Reviewed / Approved	System Specifications
	The system shall allow the guests to book a room by providing personal details (full name, contract information, payment details)		1			After a booking request from a guest, the system shall receive the booking and personal details to validate. When each form is entered and valid, the system shall generate a unique BookingID to the guest and shall store the guest's personal information in the database. The system shall then send a confirmation message to the guest's mobile app (If the user books from an OTA then a confirmation email)..
	For					

3.1.1 Product Requirements

Requirements which specify that the delivered product must behave in a particular way e.g. execution speed, reliability, etc.

3.2.1.1 Usability Requirements

The system needs to be easy and intuitive for users to interact with the system, these are the usability requirements:

- The system's interface must be intuitive and easy to use, requiring no more than 1 hour of training for new employees to perform basic operations.
- The user interface should incorporate best practices in UX design, ensuring users can efficiently navigate through the system without extensive guidance.
- The system must be designed for accessibility, with settings like high contrast, text-to-speech, and alternative text for images.
- The system should have a complete documentation showcasing the detailed roles, privileges, and permissions for each user type, ensuring that users using the system are guided on their respective functionalities.

3.2.1.2 Performance Requirements

Scalability:

- The system shall be designed to handle up to 500 simultaneous users (guests and staff) without any degradation in performance.
- The system shall be scalable to accommodate future growth, supporting up to 1000 users without requiring major architectural changes.

Response Time:

- The system must process room booking requests, in less than 3 seconds per transaction under normal load conditions.
- All user interactions, including guest check-in/check-out, transaction processing, and room service requests, must be completed with minimal delays (within 2 seconds) to enhance the user experience.

System Performance:

- The system must be capable of handling up to 1000 database transactions per minute without slowdowns or system crashes.
- The database should be optimized to ensure fast query execution, especially during high-traffic periods (e.g., check-in/check-out times).
- Load testing must be conducted regularly to simulate peak usage and ensure the system meets performance benchmarks.

3.2.1.3 Availability

- The system shall maintain an uptime of 99.9% during business hours (6 AM–12 AM local time).
- Outside business hours, the system shall maintain an uptime of 99.5%.
- Critical modules (e.g., booking, check-in/out, payment processing) shall have 99.99% availability during peak hours (9 AM–11 AM and 6 PM–8 PM).
- **Scheduled Maintenance:** Maintenance windows shall not exceed 2 hours per month and shall occur during off-peak hours (2 AM–4 AM local time).
- **Scheduled Maintenance:** Users shall be notified of scheduled maintenance 72 hours in advance via email and in-system notifications.
- **Unscheduled Maintenance:** The system shall notify users immediately upon detecting an issue and provide regular updates until resolution.
- **Unscheduled Maintenance:** Unscheduled maintenance shall not exceed 1 hour per quarter**.**
- The system shall have a *Mean Time Between Failures* of ≥10,000 hours.

- The system shall not exceed 1 system-wide failure per quarter.
- For individual modules (e.g., booking, billing), the maximum permitted failures shall be ≤2 per month.

3.2.1.4 Security

The system shall follow the following requirement procedures:

Encryption:

- All sensitive data (e.g., guest details, payment information, keycard data) shall be encrypted using AES-256 encryption.

Activity:

- The system shall log 100% of user actions (e.g., logins, bookings, payments) with timestamps, user IDs, and IP addresses.
- Audit logs shall be retained for 7 years for compliance and auditing purposes.
- Audit logs shall be accessible only to authorized administrators and protected from tampering.

Role-Based Access Control:

- Modules shall communicate only with authorized modules based on predefined roles and permissions (e.g., housekeeping cannot access billing data).
- User sessions shall expire after 15 minutes of inactivity and require re-authentication.

Data Integrity Checks:

- The system shall perform daily checksum validation on critical data sets (e.g., guest profiles, booking records) to detect tampering.

Malware and Intrusion Prevention:

- The system shall run real-time antivirus and anti-malware scans on all servers and endpoints.
- The system shall include an IDS to detect and block unauthorized access attempts.

3.1.2 Organizational Requirements

The system needs to follow these following organizational policies and procedures:

Environmental Requirements:

- These requirements focus on sustainability, energy efficiency, and ethical considerations:
- The system shall minimize resource usage (e.g., CPU, memory) to reduce energy consumption.

Operational Requirements:

Security and Access Control

- All users (staff and administrators) shall use strong passwords (minimum 12 characters, including uppercase, lowercase, numbers, and special characters) that expire every 90 days.
- All security incidents (e.g., data breaches, unauthorized access) shall be reported to the IT security team within 15 minutes of detection.

Backup and Recovery

- The system shall perform daily automated backups of all critical data (e.g., bookings, guest details, transactions).

- A disaster recovery plan shall be in place, with a maximum recovery time objective (RTO) of 1 hour for critical modules.

Vendor and Third-Party Management

- All third-party vendors (e.g., payment gateways, OTAs) shall comply with the organization's security and privacy policies.
- Third-party vendors shall undergo annual security audits to ensure compliance with organizational standards.

Development Requirements:

- The system shall be implemented in phases, starting with core modules (e.g., booking, check-in/out) and gradually adding more advanced features.
- A pilot implementation shall be conducted at one hotel location for 3 months to identify and resolve issues before full rollout.
- The system shall be developed using Agile methodologies, with bi-weekly sprints, daily stand-ups, and sprint reviews.

3.1.3 External Requirements

Interoperability Requirements

Third-Party Integrations:

- The system shall integrate with Online Travel Agencies (OTAs) (e.g., Booking.com, Expedia) for real-time room availability synchronization.
- The system shall integrate with Point-of-Sale (POS) systems (e.g., Square, Toast) for restaurants and bars to log orders and link charges to guest bills.
- The system shall integrate with payment gateways (e.g., Stripe, PayPal) to process guest payments securely.

Legislative Requirements

GDPR/CCPA Compliance:

- The system shall provide tools for guest data consent, deletion requests, and access logs to comply with GDPR (General Data Protection Regulation) and CCPA (California Consumer Privacy Act).

PCI DSS Compliance:

- Payment processing modules shall comply with PCI DSS v4.0 standards to ensure secure handling of credit card information.
- The system shall automatically apply region-specific taxes (e.g., VAT, tourism taxes) to invoices to comply with local tax laws.

Globalization Requirements

- The system shall support 5 languages (e.g., English, Spanish, French, Mandarin, Arabic) for guest and staff interfaces.
- The system shall support 10 currencies for payments and billing, with real-time exchange rate updates.

4. User Scenarios/Use Cases

Provide a summary of the major functions that the product will perform. Organize the functions to be understandable to the customer or a first time reader. Include use cases and business scenarios, or provide a link to a separate document (or documents). A business scenario:

- Describes a significant business need
- Identifies, documents, and ranks the problem that is driving the scenario
- Describes the business and technical environment that will resolve the problem
- States the desired objectives
- Shows the "Actors" and where they fit in the business model
- Is specific, and measurable, and uses clear metrics for success

Use cases are associated with a particular Functional Requirement. Assuming you have the first functional requirement named BR_01, you will map it into the Use Case called UC_01 and user scenario US_01. Please keep this naming convention throughout all your use cases and diagrams.

5. Diagrams

In this section you are going to place all of the diagrams that you build throughout to the course, in following with the slides presented throughout the weeks.

5.1 ER Diagram

Standard ERD for your project. Not much but the skills gained in the DBMS course are required.

5.2 Use Case Diagram (general)

Use Case Diagram (only one, with all the use cases).

5.3 Activity Diagram

Each Activity Diagram should be associated with an use case, associated with a particular requirement which is further associated with a particular use-case. E.g BR_01 which becomes UC_01 which becomes AC_01.

5.4. Class diagram.

One class diagram (general) for all the classes. Edit it afterwards with the design pattern implemented in it.

5.5 State diagram

Place all the relevant state diagrams here.

5.6 Sequence diagram.

All sequence diagrams are associated with an Activity Diagram. A Sequence Diagram is built based on an activity diagram. If the activity diagram is named AC_07, the Sequence Diagram will be named SC_07.

5.7. Collaboration diagram

All collaboration diagrams directly relate to a sequence diagram. If a sequence diagram is named SC_07, then the collaboration diagram is named CC_07

6. Design Patterns

Choose the relevant design patterns for your project. For each, give a reasoning and the associated class and sequence diagram. These are NOT part of the above diagrams, and need not carry the following naming scheme.

7. Appendix.

Organizing the Requirements

This section is for information only as an aid in preparing the requirements document.

Detailed requirements tend to be extensive. Give careful consideration to your organization scheme. Some examples of organization schemes are described below:

By System Mode
Some systems behave quite differently depending on the mode of operation. For example, a control system may have different sets of functions depending on its mode: training, normal, or emergency.

By User Class
Some systems provide different sets of functions to different classes of users. For example, an elevator control system presents different capabilities to passengers, maintenance workers, and fire fighters.

By Objects
Objects are real-world entities that have a counterpart within the system. For example, in a patient monitoring system, objects include patients, sensors, nurses, rooms, physicians, medicines, etc. Associated with each object is a set of attributes (of that object) and functions (performed by that object). These functions are also called services, methods, or processes. Note that sets of objects may share attributes and services. These are grouped together as classes.

By Feature
A feature is an externally desired service by the system that may require a sequence of inputs to affect the desired result. For example, in a telephone system, features include local call, call forwarding, and conference call. Each feature is generally described in a sequence of stimulus-response pairs, and may include validity checks on inputs, exact sequencing of operations, responses to abnormal situations, including error handling and recovery, effects of parameters, relationships of inputs to outputs, including input/output sequences and formulas for input to output.

By Stimulus
Some systems can be best organized by describing their functions in terms of stimuli. For example, the functions of an automatic aircraft landing system may be organized into sections for loss of power, wind shear, sudden change in roll, vertical velocity excessive, etc.

By Response
Some systems can be best organized by describing all the functions in support of the generation of a response. For example, the functions of a personnel system may be organized into sections corresponding to all functions associated with generating paychecks, all functions associated with generating a current list of employees, etc.

By Functional Hierarchy
When none of the above organizational schemes prove helpful, the overall functionality can be organized into a hierarchy of functions organized by common inputs, common outputs, or common internal data access. Data flow diagrams and data dictionaries can be used to show the relationships between and among the functions and data.

Additional Comments

Whenever a new Requirements Specification is contemplated, more than one of the organizational techniques given above may be appropriate. In such cases, organize the specific requirements for multiple hierarchies tailored to the specific needs of the system under specification.

There are many notations, methods, and automated support tools available to aid in the documentation of requirements. For the most part, their usefulness is a function of organization. For example, when organizing by mode, finite state machines or state charts may prove helpful; when organizing by object, object-oriented analysis may prove helpful; when organizing by feature, stimulus-response sequences may prove helpful; and when organizing by functional hierarchy, data flow diagrams and data dictionaries may prove helpful.