Software Engineering- Requirements specification

# Project Topic (e.g. Online Grocery Store Management System)

Group Members

|  |  |  |
| --- | --- | --- |
| Member Name | Member Surname | Student ID |
| Alican | Kaya | 49429 |
| Ege Bora | Güzel | 58351 |
| İrfan | Bag | 49279 |
| Cenker | Sule | 58566 |
| Efe Berk | Tancı | 59115 |
| Berke | Turk | 58169 |

# Introduction

Purpose:  
Define the functional and non-functional requirements of the “Hotel Booking and Management Platform.”

* Scope:  
  • Guests can browse availability, make and cancel bookings, and complete online payments.  
  • Hotel staff and administrators manage rooms, reservations, and site content via an admin panel.
* High-Level Objectives:
  1. User-friendly interface for both guests and administrators
  2. Secure payment processing (credit card, PayPal)
  3. ≥99.5% system availability and nightly data backups

# Stakeholder Identification

Identify and briefly describe **at least five key stakeholders** (e.g., end-users, clients, developers). Include their roles or relationships to the system. Summarize the primary needs or expectations of each stakeholder. This shows an understanding of who the system serves and why.

|  |  |  |
| --- | --- | --- |
| Stakeholder name | Stakeholder role (who it is) | Stakeholder’s primary need (what s/he expects from the system) |
| Guest | End user making reservations | Quick room search, easy booking and cancellation |
| Hotel Manager | Oversees hotel operations | Real-time occupancy reports, booking analytics |
| System Administrator | Maintains the application | User/room/reservation CRUD, system health monitoring |
| Payment Gateway | Processes guest payments | Secure, on-time transaction handling |
| Developer | Builds and maintains software | Clear requirements, stable database schema |

# Functional Requirements

List at least 18 specific, clear, and testable functional requirements. Each should describe a distinct system feature or behavior (e.g., "The system shall allow users to log in with a username and password"). Vague or untestable statements will reduce points.

Assign a priority level (e.g., high, medium, low) to each requirement and provide a brief justification (e.g., "High priority due to core functionality").

|  |  |
| --- | --- |
| Requirement statement | Priority |
| The system shall allow users to register with email and password. | High |
| Registered users shall log in using email and password. | High |
| The system shall display available rooms filtered by date range. | High |
| The system shall allow users to view, modify, or cancel their reservations. | Medium |
| The platform shall integrate with Google Maps API to display hotel location. | Low |

# Non-Functional Requirements

Define at least **three (three of each type):**

* Performance:

1. Login requests shall complete within 2 seconds.
2. The homepage shall load in under 1 second.
3. Database queries shall execute in less than 500 ms.

* Security:

1. Passwords shall be hashed using bcrypt.
2. All traffic shall use HTTPS/TLS.
3. Input validation and parameterized queries shall prevent SQL injection.

* Usability:

1. The UI shall be intuitive enough for new users to navigate without training.
2. Form validation errors shall display immediately inline.
3. The site shall be fully responsive on desktop and mobile devices.

* Reliability & Availability:

1. System uptime shall be ≥99.5% monthly.
2. Nightly full backups shall run automatically at 02:00 server time.
3. Incident alerting shall notify admins within 5 minutes of critical failures.

|  |  |
| --- | --- |
| Non-Fuctional Requirement | Related to (performance/ security/usability/reliability) |
| Login requests shall complete within 2 seconds. | Performance |
| Passwords shall be hashed using bcrypt. | Security |
| The UI shall be intuitive enough for new users to navigate without training. | Usability |

# Constraints

* Technical Constraints:

1. Backend must use PHP 8.1 and MySQL 8.0.
2. Front-end shall use Bootstrap 5.
3. Deployment target is Ubuntu 22.04 LTS on AWS.

* Business Constraints:

1. Total project budget shall not exceed $10,000.
2. Delivery deadline: June 15, 2025.
3. Must comply with GDPR and Turkish KVKK regulations.

# Assumptions and Dependencies

* Assumptions:

1. Guests will use modern browsers (Chrome, Firefox, Safari).
2. Hotel staff will access the admin panel during business hours.
3. Internet connectivity is stable and of sufficient bandwidth.

* Dependencies:

1. Third-party payment gateway (e.g., PayPal, Stripe).
2. SMTP service for transactional emails.
3. Google Maps API for location display.

# Evaluation Criteria

**1. Deadline ok (1 point)**

* **1 point: Document was made within the deadline**
* **0 points: Document was delayed**

**2. Stakeholder Identification (1 point)**

* **List of Stakeholders (0.5 points)**: Identify and briefly describe at least five key stakeholders (e.g., end-users, clients, developers). Include their roles or relationships to the system.
* **Stakeholder Needs (0.5 points)**: Summarize the primary needs or expectations of each stakeholder. This shows an understanding of who the system serves and why.

**3. Functional Requirements (3 points)**

* **Requirement Statements (2 points)**: List at least 18 specific, clear, and testable functional requirements. Each should describe a distinct system feature or behavior (e.g., "The system shall allow users to log in with a username and password"). Vague or untestable statements will reduce points.
* **Priority Levels (1 point)**: Assign a priority level (e.g., high, medium, low) to each requirement and provide a brief justification (e.g., "High priority due to core functionality").

**4. Non-Functional Requirements (2 points)**

* **Performance Requirements (0.5 points)**: Define at least three performance-related requirement (e.g., "The system shall process login requests within 2 seconds under normal load").
* **Security Requirements (0.5 points)**: Specify at least three security requirement (e.g., "The system shall encrypt user passwords using a secure hashing algorithm").
* **Usability Requirements (0.5 points)**: Include at least three usability requirement (e.g., "The interface shall be navigable by users with no prior training").
* **Reliability and Availability (0.5 points)**: Outline at least three reliability or availability requirement (e.g., "The system shall have 99% uptime during operational hours").

**5. Constraints (1 point)**

* **Technical Constraints (0.5 points)**: Identify at least three technical limitations (e.g., "The system must be developed using Python due to team expertise").
* **Business Constraints (0.5 points)**: Mention at least three business-related constraints (e.g., "The project must be completed within a budget of $5,000" or "The system must comply with GDPR regulations").

**6. Assumptions and Dependencies (1 point)**

* **Assumptions (0.5 points)**: List at least three assumptions made during requirement gathering (e.g., "It is assumed that users have access to a modern web browser").
* **Dependencies (0.5 points)**: Identify at least three external factors or system the project relies on (e.g., "The system depends on a third-party payment gateway").

**7. Document Quality and Clarity (1 point)**

* **Organization and Structure (0.5 points)**: The document must be well-organized, with clear headings, subheadings, and a logical flow between sections.
* **Language and Precision (0.5 points)**: Use precise, unambiguous language throughout. Avoid vague terms (e.g., "fast" or "good") and ensure consistent terminology.