**Room Booker**

4/6/2024

Version 1

**Presented To:**

**DR/ Mohamed Ramadan**

**Submitted By:**

*Sayed Ragab*

*Mohamed Ashraf*

*Elsayed Mahmoud*

*Ali Zenhom*

**REVISION HISTORY**

| **Date** | **Author** | **Distributed to** | **Version** | **Description** |
| --- | --- | --- | --- | --- |
| 4/6/2024 | Ali Zenhom | Sayed Ragab | Version 1 | This is the first addition to the project |

**TABLE OF CONTENTS**

**1. PRODUCT DESCRIPTION**

Description: "Room Booker" is a web application designed to streamline the process of booking bedrooms in a hotel. It offers a user-friendly interface for guests to browse available rooms, check availability, make reservations, and manage their bookings conveniently online.

**2. TEAM DESCRIPTION**

*Sayed Ragab (frontend developer)*

*Mohamed Ashraf (backend Developer)*

*Elsayed Mahmoud (frontend developer)*

*Ali Zenhom (database developer)*

**3. SOFTWARE PROCESS MODEL DESCRIPTION**

We will use the Agile development model for this project, "Room Booking." Agile is chosen for its ability to adapt to continuous changes in project requirements and deliver continuous value to customers through iterative delivery of demonstrable features. Additionally, the Agile model allows for continuous interaction with customers and continuous product improvement based on feedback, helping to better meet user needs and expectations.

**4. Product Definition:**

Below is a description of the users and user environment for the hotel room booking system:

Users:

1. Guests: These are individuals who wish to book rooms at the hotel for their stay.

2. Hotel staff: This category includes receptionists and reservation staff who manage and execute booking operations.

User Environment:

1. Electronic website: The room booking system is accessed through an electronic application that operates on smartphones and tablet devices.

2. Hotel Reception: Guests can also make bookings directly by visiting the hotel reception and speaking to a receptionist.

Context Diagram:

The context diagram displays the system scope, indicating what is inside the system (such as the electronic website and reservation management system) and what is outside the system (such as end users and external payment systems). It also illustrates the flow of information between the system and external users, such as the room booking process, confirmation of bookings, and sending invoices.

Examples of Personas:

1. A couple planning to spend a weekend getaway at a beachfront hotel.

2. A businessman needing to book a room for accommodation during a business trip.

3. A hotel receptionist assisting guests and managing reservations.

These are the key elements that comprise the product definition for the hotel room booking system.

User Stories:

1. As a guest, I want to be able to search for available rooms on specific dates, so I can plan my stay accordingly.

2. As a guest, I want to be able to view detailed information about each available room, including amenities and pricing, to help me make an informed decision.

3. As a guest, I want to be able to easily book a room online, with a simple and intuitive booking process.

4. As a guest, I want to receive confirmation of my booking via email or SMS, to ensure my reservation is secured.

5. As a hotel staff member, I want to be able to manage room availability and reservations efficiently, to ensure optimal occupancy and customer satisfaction.

6. As a hotel staff member, I want to be able to check-in guests smoothly upon their arrival, providing them with a seamless experience.

7. As a hotel staff member, I want to be able to handle booking cancellations or modifications effectively, while maintaining clear communication with guests.

High Level Use Cases:

1. Search for Available Rooms:

- Unique Name: SearchRooms

- Participating Actors: Guest

- Entry Conditions: Guest accesses the room booking system.

- Exit Conditions: Guest receives a list of available rooms matching the search criteria.

- Flow of Events:

1. Guest enters the desired dates and other search criteria.

2. System retrieves available rooms based on the entered criteria.

3. System displays a list of available rooms to the guest.

2. Book a Room:

- Unique Name: BookRoom

- Participating Actors: Guest

- Entry Conditions: Guest selects a room for booking.

- Exit Conditions: Guest receives booking confirmation.

- Flow of Events:

1. Guest selects a specific room from the list of available rooms.

2. Guest provides necessary booking details (e.g., name, contact information).

3. System confirms room availability and generates a booking.

4. Guest receives booking confirmation via email or SMS.

Use Case Descriptions:

SearchRooms:

- Unique Name: SearchRooms

- Participating Actors: Guest

- Entry Conditions: Guest accesses the room booking system.

- Exit Conditions: Guest receives a list of available rooms matching the search criteria.

- Flow of Events:

1. Guest enters the desired dates and other search criteria.

2. System retrieves available rooms based on the entered criteria.

3. System displays a list of available rooms to the guest.

- Special Requirements: The system should allow guests to filter search results based on room type, price range, and other relevant criteria.

BookRoom:

- Unique Name: BookRoom

- Participating Actors: Guest

- Entry Conditions: Guest selects a room for booking.

- Exit Conditions: Guest receives booking confirmation.

- Flow of Events:

1. Guest selects a specific room from the list of available rooms.

2. Guest provides necessary booking details (e.g., name, contact information).

3. System confirms room availability and generates a booking.

4. Guest receives booking confirmation via email or SMS.

- Special Requirements: The system should provide a user-friendly interface for guests to input booking details and ensure secure payment processing.

**5. USER EXPERIENCE WIREFRAMES**

1. **Homepage:**
   * The homepage typically includes the hotel logo, navigation menu, search bar, and possibly featured promotions or deals.
2. **Search Results:**
   * After entering search criteria (e.g., location, dates, number of guests), users are presented with a list of available rooms matching their preferences.
   * Each room listing includes basic details like room type, price, and a thumbnail image. Users can click on a listing to view more details.
3. **Room Details:**
   * This page displays comprehensive information about a selected room, including amenities, description, and larger images.
   * Users can choose their preferred dates and select additional options like breakfast or room upgrades.
4. **Booking Summary:**
   * After selecting dates and options, users proceed to the booking summary page where they can review their selections and finalize the booking.
   * This page displays the total cost, along with any applicable taxes or fees.
5. **Checkout:**
   * The checkout page collects user information required for booking confirmation, such as name, email, and payment details.
   * Users may also have the option to create an account for faster booking in the future.
6. **Confirmation:**
   * After completing the booking, users receive a confirmation message along with their booking details.
   * This page may also include a call-to-action for users to share their booking on social media or download a PDF confirmation.

These wireframes provide a basic visual representation of the user journey through the hotel room booking process. Actual designs may vary based on specific branding, functionality, and user experience considerations.

Top of Form



Bottom of Form

**6.Project Organization:**

Breakdown of major tasks and schedule:

1. Requirements Analysis:

- Analyzing user needs

- Identifying functional and non-functional requirements

2. Design:

- User interface design

- Database structure identification

3. Development:

- Developing the online booking application

- Developing the reservation management system

4. Testing:

- Unit testing

- System testing

- Performance testing

5. Deployment and Maintenance:

- Deploying the application on servers

- Providing technical support and ongoing maintenance

Matrix of Responsibilities:

| | |

|  |  |
| --- | --- |
| Task | Responsible |
| Requirements Analysis | Systems Analyst |
| Design | User Interface Designer and Database Developer |
| Development | Website Developer and Reservation System Developer |
| Testing | Testing Team |
| Deployment and Maintenance | Hosting System Administrator and Technical Support Team |

PERT / Gantt Chart:

An initial schedule will be determined using PERT and Gantt chart. This schedule may include specified tasks and expected deadlines for completion.

**7. VALIDATION PLAN**

Test Strategy:

The test strategy for the room booking project will involve comprehensive testing procedures to ensure the functionality, usability, and reliability of the system. This includes various types of testing such as unit testing, integration testing, system testing, and user acceptance testing (UAT).

Definition of Done:

The definition of done for the room booking project means that all planned features and functionalities have been implemented and thoroughly tested. It also involves meeting predefined acceptance criteria and ensuring that the system meets the requirements outlined in the project specifications.

Success Criteria:

Success for the room booking project will be determined by the following criteria:

1. Functional Requirements: All functional requirements specified for the system are implemented correctly and operate as intended. This includes features such as room search, booking, cancellation, and modification.

2. Usability: The system is user-friendly, intuitive, and easy to navigate for both guests and hotel staff. Users should be able to complete booking processes without encountering major obstacles or confusion.

3. Reliability: The system operates reliably under normal conditions, without frequent crashes, errors, or disruptions. It should be able to handle simultaneous user interactions without significant performance degradation.

4. Security: The system ensures the security and confidentiality of user data, including personal information and payment details. It should be protected against unauthorized access, data breaches, and other security threats.

5. Performance: The system performs efficiently, with acceptable response times for user interactions such as searching for rooms, making bookings, and viewing reservation details.

6. Scalability: The system should be able to scale to accommodate increased demand, such as peak booking periods or higher traffic volumes, without significant degradation in performance or functionality.

Meeting these success criteria indicates that the room booking system has been successfully developed and validated, ready for deployment and use by guests and hotel staff.

**8. FEASIBILITY STUDY:**

Risk Identification:

1. Technical Risks:

- Integration Complexity: Integrating the room booking system with existing hotel management systems may pose technical challenges.

- Scalability: Ensuring that the system can handle a large volume of concurrent users during peak booking periods.

- Security Vulnerabilities: Risks related to data breaches, unauthorized access, and payment security.

2. Operational Risks:

- Staff Training: Ensuring that hotel staff are adequately trained to use the new booking system.

- System Downtime: Risks of system failures or downtime impacting guest bookings and hotel operations.

3. Financial Risks:

- Development Costs: Overestimation of development costs leading to budget overruns.

- Return on Investment: Uncertainty regarding the return on investment and revenue generation from the new booking system.

Risk Prioritization:

1. Security Vulnerabilities

2. Integration Complexity

3. Scalability

4. Staff Training

5. System Downtime

6. Development Costs

7. Return on Investment

Risk Mitigation:

1. Security Vulnerabilities: Implement robust security measures such as encryption, access controls, and regular security audits. Regularly update security protocols to address emerging threats.

2. Integration Complexity: Conduct thorough testing and collaboration with the vendors of existing systems to ensure seamless integration. Develop clear communication channels to address any integration issues promptly.

3. Scalability: Design the system with scalability in mind, utilizing cloud-based infrastructure and load balancing techniques. Perform stress testing to identify potential bottlenecks and optimize system performance.

4. Staff Training: Develop comprehensive training programs for hotel staff, including user manuals and hands-on workshops. Provide ongoing support and refresher courses to ensure staff proficiency.

5. System Downtime: Implement redundant systems and backup procedures to minimize the impact of system failures. Establish a rapid response team to address technical issues and restore services promptly.

6. Development Costs: Conduct thorough cost-benefit analysis and budget planning to accurately estimate development costs. Prioritize essential features and functionalities to ensure efficient resource allocation.

7. Return on Investment: Continuously monitor key performance indicators (KPIs) such as booking conversion rates and revenue growth. Implement marketing strategies to promote the new booking system and attract more guests.

Prototype:

A prototype of the room booking system will be developed to validate the feasibility of the project and gather feedback from stakeholders. The prototype will include essential features such as room search, booking functionality, and user authentication. It will be a simplified version of the final product, focusing on core functionalities to demonstrate its potential value.

**9. CONFIGURATION AND VERSION CONTROL**

Process and Attributes:

1. Version Control System (VCS):

- Utilize a centralized version control system such as Git to manage all project artifacts, including code, documentation, and configuration files.

- Establish a central repository hosted on a platform like GitHub or GitLab to store and track changes to project files.

2. Branching Strategy:

- Implement a branching strategy to organize development efforts and facilitate collaboration among team members.

- Use feature branches for developing new features or enhancements, ensuring that each feature is isolated and can be easily reviewed and merged into the main branch.

- Maintain a main branch (e.g., master or main) that reflects the stable, production-ready version of the software.

3. Versioning Scheme:

- Adopt a semantic versioning scheme (e.g., MAJOR.MINOR.PATCH) to assign version numbers to releases.

- Increment the MAJOR version for incompatible API changes, MINOR version for backward-compatible feature additions, and PATCH version for backward-compatible bug fixes.

4. Commit Guidelines:

- Enforce descriptive commit messages that clearly communicate the purpose of each change.

- Include references to relevant issues or user stories in commit messages to provide context and traceability.

5. Code Reviews:

- Mandate code reviews for all changes before merging them into the main branch.

- Conduct thorough reviews to ensure code quality, adherence to coding standards, and alignment with project objectives.

6. Continuous Integration (CI) and Continuous Deployment (CD):

- Integrate CI/CD pipelines into the version control process to automate testing, building, and deployment tasks.

- Configure automated tests to run on each commit to detect and prevent regressions early in the development life cycle.

7. Access Control:

- Define granular access controls within the version control system to restrict access to sensitive information and limit the actions users can perform.

- Grant permissions based on role and responsibility, ensuring that only authorized personnel can modify critical files.

8. Documentation:

- Maintain comprehensive documentation within the version control system, including project requirements, design specifications, and user manuals.

- Ensure that documentation is versioned along with the codebase to maintain consistency and traceability.

By following these processes and attributes for version control, the room booking project can effectively manage changes, track progress, and ensure the integrity and reliability of project artifacts throughout the development life cycle.

**10. TOOLS**

1.programming languages:

* JavaScript(node.js): for backend development
* HTML/CSS/JavaScript: for front end development

2. **Frontend Frameworks**:

* **React.js** or **Vue.js**: For building dynamic and interactive user interfaces.

3. **Backend Frameworks**:

* **Express.js**: A lightweight Node.js framework for building the backend RESTful APIs.

4. **Database**:

* **MongoDB** or **MySQL**: For storing hotel room details, booking information, user data, etc.

**11. ARCHITECTURE**

1. **Server Infrastructure**:
   * **Web Server**: Hardware capable of running a web server software like Apache or Nginx to serve the web application.
   * **Database Server**: Hardware to host the database server (e.g., MySQL, PostgreSQL, MongoDB) to store hotel room data, user information, and booking details.
   * **Load Balancer (optional)**: If the application experiences high traffic, a load balancer distributes incoming web traffic across multiple servers to ensure high availability and reliability.
2. **Networking Equipment**:
   * **Router and Switches**: Hardware components for network connectivity within the server infrastructure.
   * **Firewall**: To protect the server infrastructure from unauthorized access and potential security threats.
3. **Storage**:
   * **Hard Drives or Solid State Drives (SSDs)**: Storage devices to store the web application files, database files, and any other data related to the application.
   * **Network-Attached Storage (NAS)**: Optional storage solution for backups and data redundancy.
4. **Client Devices**:
   * **Desktops/Laptops**: Devices used by users to access the web application for browsing available rooms, making bookings, and managing reservations.
   * **Smartphones/Tablets**: Mobile devices that should be compatible with the web application for users who prefer to book rooms using their mobile devices.
   * **Point-of-Sale (POS) Terminals (optional)**: For hotel staff to manage bookings and reservations directly from the hotel's reception.
5. **Peripheral Devices (Optional)**:
   * **Printers**: To print booking confirmations, invoices, or other documents related to room reservations.
   * **Barcode/QR Code Scanners**: For efficient check-in and check-out processes, especially in larger hotels.
6. **Power Backup (UPS)**:
   * Uninterruptible Power Supply (UPS) systems to provide backup power in case of power outages or fluctuations, ensuring continuous operation of the server infrastructure and networking equipment.
7. **Monitoring and Management Tools**:
   * **Server Monitoring Software**: To monitor server performance, track resource usage, and identify potential issues proactively.
   * **Remote Management Tools**: Software solutions for remote administration and management of server infrastructure.
8. **Security Systems**:
   * **Surveillance Cameras (CCTV)**: For physical security monitoring of the server room and other critical areas.
   * **Intrusion Detection Systems (IDS)** and **Intrusion Prevention Systems (IPS)**: To detect and prevent unauthorized access and potential security breaches.
9. **Environmental Controls**:
   * **Cooling Systems**: To maintain optimal temperature and humidity levels in the server room, preventing overheating and hardware failures.
   * **Environmental Monitoring Sensors**: To monitor temperature, humidity, and other environmental factors in the server room.
10. **Backup and Disaster Recovery Solutions**:
    * **Regular Backup Systems**: Automated backup solutions to backup critical data and configurations regularly.
    * **Disaster Recovery Plans**: Strategies and procedures to recover the application and data in the event of a disaster or system failure.

.