Bookify Hotel Reservation System Documentation

1. Project Planning

1.1 Project Overview

Bookify is a hotel reservation web application built using ASP.NET Core MVC, Entity Framework Core, and N-Tier architecture. The system allows customers to browse rooms, book reservations, and complete payments. Admin users manage rooms, room types, and bookings.

1.2 Objectives

- Build a scalable, maintainable hotel booking platform.
- Implement N-Tier Architecture, Repository Pattern, and Unit of Work.
- Provide secure authentication, role-based access, and Stripe payment integration.
- Deliver a responsive UI/UX for customers and admins.

1.3 Project Scope

- Public room listing and search.
- Room details and reservation flow.
- User authentication & authorization.
- Admin dashboard for management.
- Booking confirmation with Stripe payments.
- Logging, health checks, and responsive UI.

1.4 Timeline (4 Weeks)

- **Week 1:** Architecture setup, database design, room listing.
- **Week 2:** Reservation flow, RBAC, Admin Panel.
- **Week 3:** Booking confirmation, Stripe payments, user profiles.
- **Week 4:** Logging, health checks, UI improvements, final testing.

---

2. Stakeholder Analysis

2.1 Stakeholder List

1. Customers – Browse rooms, make bookings, complete payments, manage profiles.
2. Admin Users – Manage rooms, room types, bookings, and system content.
3. Hotel Management – Oversees the system, reviews reports.

4. Developers – Responsible for development, maintenance, and updates.
5. Payment Gateway Provider (Stripe) – Handles secure online payments.

2.2 Stakeholder Interests & Needs

Stakeholder	Needs
Customers:	Simple UI, fast booking, secure payments, booking history
Admin Users:	Easy management tools, secure access, real-time updates
Management:	Reports, accurate data, system stability
Developers:	Clear architecture, maintainability, logs, monitoring
Stripe:	Secure API transactions, proper payment workflows

---

3. Database Design

3.1 Main Entities

RoomTypes – Defines the category and pricing model for rooms.
Rooms – Physical room instances that belong to room types.
Bookings – Customer reservations linked to rooms and users.
Users – Application users handled via ASP.NET Identity.
Payments – Stores Stripe transaction IDs and statuses.

3.2 ERD Structure (Text-Based)

```
RoomTypes
- Id (PK)
- Name
- Description
- Price
- Capacity

Rooms
- Id (PK)
- RoomNumber
- RoomTypeId (FK)
- Status

Bookings
- Id (PK)
- UserId (FK)
- RoomId (FK)
- CheckInDate
- CheckOutDate
- TotalPrice
- PaymentStatus

Payments
```

```
- Id (PK)
- BookingId (FK)
- StripeTransactionId
- Amount
- CreatedAt
```

3.3 Relationships

- One **RoomType** ↔ Many **Rooms**
- One **Room** ↔ Many **Bookings**
- One **Booking** ↔ One **Payment**
- One **User** ↔ Many **Bookings**

---

4. UI/UX Design

4.1 Design Principles

- Clean, modern UI using Bootstrap.
- Minimalist layouts with clear navigation.
- Fast browsing and searching.
- Admin dashboard with tables, filters, and CRUD modals.

4.2 Main Screens

Customer Side:

1. **Home / Room Listings** – Grid view of available rooms.
2. **Room Details** – Images, description, availability.
3. **Reservation Cart** – Selected room and dates.
4. **Checkout + Stripe Payment** – Final booking process.
5. **Profile Page** – Personal info + booking history.

Admin Side:

1. **Admin Dashboard** – Statistics, recent bookings.
2. **Room Types Management** – Add/Edit/Delete room types.
3. **Rooms Management** – CRUD for rooms.
4. **Bookings Management** – View, update status.
5. **Users Management** – Role assignments.

4.3 Wireframe Descriptions

Room Listing Page: Card layout → image, name, price, book button.
Admin Table Pages: DataTables → search, pagination, export.

5. Conclusion

This documentation outlines the full planning, stakeholders, database structure, and UI/UX structure for the Bookify Hotel Reservation System. It prepares the foundation for successful implementation following N-Tier Architecture, Repository Pattern, and best practices.