1. Document Purpose

This Database Design Document describes the logical and physical database design for **Bookify**, a hotel reservation system built using **ASP.NET Core** with **Microsoft SQL Server** as the target DBMS. It covers requirements, the conceptual model, logical schema, physical schema (DDL), indexing, normalization considerations, sample queries, backup & maintenance, and security guidelines.

2. Requirements & Assumptions

Functional requirements (DB-related):

- Store hotel, room, and availability information.
- Support user (customer) accounts and administrative users.
- Record bookings/reservations with status lifecycle.
- Record payments (or payment simulation), invoices, and transactions.
- Support hotel reviews and ratings.
- Track room pricing (base price + seasonal/special rates) and room amenities.
- Maintain audit fields (created_by, created_at, updated_by, updated_at).
- Soft-delete support for key entities.

Non-functional requirements:

- ACID compliance for booking/payment operations.
- Reasonable performance for search and booking queries.
- Scalable design to support multi-hotel and multi-city scenarios.
- Secure storage of sensitive fields (no plain-text payment card storage).

Assumptions:

- Payments are processed through a third-party gateway; only tokens/transaction IDs and minimal metadata are stored.

- One hotel has many rooms; one room type may be reused across hotels (optional).
- Time zones are simplified to UTC in DB; application layer handles local displays.

3. Conceptual Overview (ER Summary)

Main entities:

- User (customers and admin users)
- Hotel
- Room (individual room or room type)
- Amenity (and HotelAmenity / RoomAmenity mapping)
- Booking (reservation)
- BookingItem (optional: for multi-room bookings)
- Payment / Transaction
- Review
- Rate (price rules, seasonal rates)
- City / Location (optional)
- Audit / Logs (optional)

Relationships (high-level):

- User (1) — (M) Booking
- Hotel (1) — (M) Room
- Room (1) — (M) BookingItem (a booking can include one or more rooms)
- Booking (1) — (1..1) Payment (a booking may have 0..1 payment records)
- User (1) — (M) Review — (1) Hotel
- Hotel (M) — (M) Amenity (through HotelAmenity)

4. Logical Schema (Tables & Columns)

Below are the main tables with key columns. Types are for **Microsoft SQL Server**.

4.1 Users

- **Users**
    - `UserId` INT IDENTITY(1,1) PRIMARY KEY
    - `Email` NVARCHAR(256) NOT NULL UNIQUE
    - `PasswordHash` NVARCHAR(512) NOT NULL
    - `FirstName` NVARCHAR(100)
    - `LastName` NVARCHAR(100)
    - `Phone` NVARCHAR(30)
    - `Role` NVARCHAR(50) NOT NULL -- e.g., Customer, Admin, HotelManager
    - `IsActive` BIT DEFAULT 1
    - `CreatedAt` DATETIME2 DEFAULT SYSUTCDATETIME()
    - `UpdatedAt` DATETIME2 NULL
    - `DeletedAt` DATETIME2 NULL

Notes: Store password hashes (and salts) only; consider using ASP.NET Identity.

4.2 Hotels

- **Hotels**
    - `HotelId` INT IDENTITY PRIMARY KEY
    - `Name` NVARCHAR(256) NOT NULL
    - `Description` NVARCHAR(MAX) NULL
    - `Address` NVARCHAR(512)

- `City` NVARCHAR(100)
- `Country` NVARCHAR(100)
- `Latitude` DECIMAL(9,6) NULL
- `Longitude` DECIMAL(9,6) NULL
- `Phone` NVARCHAR(30) NULL
- `Email` NVARCHAR(256) NULL
- `IsActive` BIT DEFAULT 1
- `CreatedAt`, `UpdatedAt`, `DeletedAt` DATETIME2 as above

4.3 Rooms

- **Rooms**
  - `RoomId` INT IDENTITY PRIMARY KEY
  - `HotelId` INT NOT NULL FOREIGN KEY → Hotels(HotelId)
  - `RoomNumber` NVARCHAR(50) NULL -- optional if tracking physical room
  - `RoomType` NVARCHAR(100) NOT NULL -- e.g., Single, Double, Suite
  - `MaxOccupancy` INT DEFAULT 2
  - `Description` NVARCHAR(MAX)
  - `BasePrice` DECIMAL(10,2) NOT NULL
  - `IsAvailable` BIT DEFAULT 1
  - `CreatedAt`, `UpdatedAt`, `DeletedAt`

Alternative: Use `RoomTypes` table + `Rooms` referencing `RoomTypeId` if you want normalized room types reused.

4.4 Amenities

- **Amenities**
    - `AmenityId` INT IDENTITY PRIMARY KEY
    - `Name` NVARCHAR(100)
    - `Description` NVARCHAR(256)
- **HotelAmenities**
    - `HotelAmenityId` INT IDENTITY PRIMARY KEY
    - `HotelId` INT FOREIGN KEY → Hotels(HotelId)
    - `AmenityId` INT FOREIGN KEY → Amenities(AmenityId)
- **RoomAmenities** (optional if amenities can be at room level)
    - `RoomAmenityId` INT IDENTITY PRIMARY KEY
    - `RoomId` INT FOREIGN KEY → Rooms(RoomId)
    - `AmenityId` INT FOREIGN KEY → Amenities(AmenityId)

4.5 Rates (Seasonal / Special Pricing)

- **Rates**
    - `RateId` INT IDENTITY PRIMARY KEY
    - `RoomId` INT FOREIGN KEY → Rooms(RoomId)
    - `StartDate` DATE NOT NULL
    - `EndDate` DATE NOT NULL
    - `Price` DECIMAL(10,2) NOT NULL
    - `IsActive` BIT DEFAULT 1
    - `CreatedAt`, `UpdatedAt`

Behavior: Application chooses the most specific active rate for a booking date range; otherwise `Rooms.BasePrice` is used.

4.6 Bookings & BookingItems

- **Bookings**
    - `BookingId` INT IDENTITY PRIMARY KEY
    - `UserId` INT NOT NULL FOREIGN KEY → Users(UserId)
    - `HotelId` INT NOT NULL FOREIGN KEY → Hotels(HotelId)
    - `BookingReference` NVARCHAR(50) UNIQUE -- human-friendly ref
    - `CheckInDate` DATE NOT NULL
    - `CheckOutDate` DATE NOT NULL
    - `TotalAmount` DECIMAL(12,2) NOT NULL
    - `Status` NVARCHAR(50) NOT NULL -- e.g., Pending, Confirmed, Cancelled, Completed
    - `CreatedAt`, `UpdatedAt`, `CancelledAt` DATETIME2 NULL
- **BookingItems** (for multi-room per booking)
    - `BookingItemId` INT IDENTITY PRIMARY KEY
    - `BookingId` INT FOREIGN KEY → Bookings(BookingId)
    - `RoomId` INT FOREIGN KEY → Rooms(RoomId)
    - `PricePerNight` DECIMAL(10,2)
    - `Nights` INT
    - `Amount` DECIMAL(12,2)

4.7 Payments / Transactions

- **Payments**

- `PaymentId` INT IDENTITY PRIMARY KEY
- `BookingId` INT FOREIGN KEY → Bookings(BookingId) NULL
- `UserId` INT FOREIGN KEY → Users(UserId)
- `TransactionId` NVARCHAR(100) NULL -- gateway transaction id
- `PaymentMethod` NVARCHAR(50) -- e.g., Card, Wallet
- `Amount` DECIMAL(12,2)
- `Currency` NVARCHAR(10) DEFAULT 'EGP'
- `Status` NVARCHAR(50) -- Pending, Paid, Failed, Refunded
- `CreatedAt` DATETIME2
- `Metadata` NVARCHAR(MAX) NULL -- store gateway response (JSON) if needed

4.8 Reviews

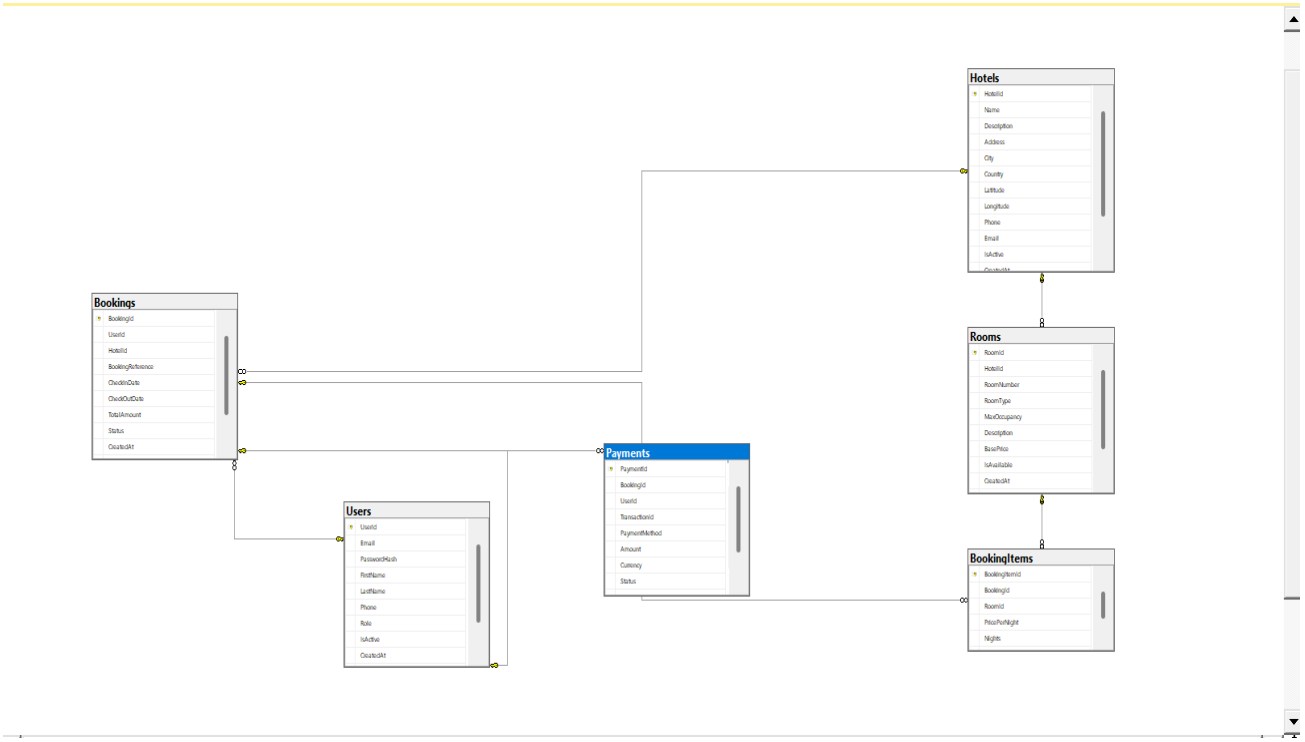
- **Reviews**
  - `ReviewId` INT IDENTITY PRIMARY KEY
  - `HotelId` INT FOREIGN KEY → Hotels(HotelId)
  - `UserId` INT FOREIGN KEY → Users(UserId)
  - `Rating` TINYINT NOT NULL -- 1..5
  - `Title` NVARCHAR(200)
  - `Comment` NVARCHAR(MAX)
  - `IsApproved` BIT DEFAULT 0 -- moderation flag
  - `CreatedAt` DATETIME2

4.9 Audit / Logs (Optional)

- **AuditLogs**
    - `AuditId` INT IDENTITY PRIMARY KEY
    - `EntityName` NVARCHAR(100)
    - `EntityId` NVARCHAR(50)
    - `Action` NVARCHAR(50)
    - `PerformedBy` NVARCHAR(256) NULL
    - `PerformedAt` DATETIME2 DEFAULT SYSUTCDATETIME()
    - `Details` NVARCHAR(MAX) NULL

4. SQLServer DataBase (Diagram & Queries)

4.1 Diagram



4.1 creating tables queries

~vs2708.sql - RAH...hma elhakime (55))* SQLQuery1.sql - RA...hma elhakime (53))* RAHMAELHAKIM.Hotel - Diagram_0*

```
);

-- Rooms
CREATE TABLE dbo.Rooms (
    RoomId INT IDENTITY(1,1) PRIMARY KEY,
    HotelId INT NOT NULL,
    RoomNumber NVARCHAR(50) NULL,
    RoomType NVARCHAR(100) NOT NULL,
    MaxOccupancy INT DEFAULT 2,
    Description NVARCHAR(MAX) NULL,
    BasePrice DECIMAL(10,2) NOT NULL,
    IsAvailable BIT DEFAULT 1,
    CreatedAt DATETIME2 DEFAULT SYSUTCDATETIME(),
    UpdatedAt DATETIME2 NULL,
    DeletedAt DATETIME2 NULL,
    CONSTRAINT FK_Rooms_Hotels FOREIGN KEY (HotelId) REFERENCES dbo.Hotels(HotelId)
);

-- Bookings
CREATE TABLE dbo.Bookings (
    BookingId INT IDENTITY(1,1) PRIMARY KEY,
    UserId INT NOT NULL,
    HotelId INT NOT NULL,
    BookingReference NVARCHAR(50) UNIQUE,
    CheckInDate DATE NOT NULL,
    CheckOutDate DATE NOT NULL,
    TotalAmount DECIMAL(12,2) NOT NULL,
    Status NVARCHAR(50) NOT NULL,
    CreatedAt DATETIME2 DEFAULT SYSUTCDATETIME(),
    UpdatedAt DATETIME2 NULL,
    CancelledAt DATETIME2 NULL,
    CONSTRAINT FK_Bookings_Users FOREIGN KEY (UserId) REFERENCES dbo.Users(UserId),
    CONSTRAINT FK_Bookings_Hotels FOREIGN KEY (HotelId) REFERENCES dbo.Hotels(HotelId)
);
```

~vs2708.sql - RAH...hma elhakime (55))* SQLQuery1.sql - RA...hma elhakime (53))* RAHMAELHAKIM.Hotel - Diagram_0*

```
);

-- BookingItems
CREATE TABLE dbo.BookingItems (
    BookingItemId INT IDENTITY(1,1) PRIMARY KEY,
    BookingId INT NOT NULL,
    RoomId INT NOT NULL,
    PricePerNight DECIMAL(10,2),
    Nights INT,
    Amount DECIMAL(12,2),
    CONSTRAINT FK_BookingItems_Bookings FOREIGN KEY (BookingId) REFERENCES dbo.Bookings(BookingId),
    CONSTRAINT FK_BookingItems_Rooms FOREIGN KEY (RoomId) REFERENCES dbo.Rooms(RoomId)
);

-- Payments
CREATE TABLE dbo.Payments (
    PaymentId INT IDENTITY(1,1) PRIMARY KEY,
    BookingId INT NULL,
    UserId INT NOT NULL,
    TransactionId NVARCHAR(100) NULL,
    PaymentMethod NVARCHAR(50),
    Amount DECIMAL(12,2),
    Currency NVARCHAR(10) DEFAULT 'EGP',
    Status NVARCHAR(50),
    CreatedAt DATETIME2 DEFAULT SYSUTCDATETIME(),
    Metadata NVARCHAR(MAX) NULL,
    CONSTRAINT FK_Payments_Users FOREIGN KEY (UserId) REFERENCES dbo.Users(UserId),
    CONSTRAINT FK_Payments_Bookings FOREIGN KEY (BookingId) REFERENCES dbo.Bookings(BookingId)
);
```

~vs2708.sql - RAH...hma elhakime (55))* | SQLQuery1.sql - RA...hma elhakime (53))* | RAHMAELHAKIM.Hotel - Diagram_0*

```sql
create database Hotel;

-- Users
CREATE TABLE dbo.Users (
    UserId INT IDENTITY(1,1) PRIMARY KEY,
    Email NVARCHAR(256) NOT NULL UNIQUE,
    PasswordHash NVARCHAR(512) NOT NULL,
    FirstName NVARCHAR(100),
    LastName NVARCHAR(100),
    Phone NVARCHAR(30),
    Role NVARCHAR(50) NOT NULL,
    IsActive BIT DEFAULT 1,
    CreatedAt DATETIME2 DEFAULT SYSUTCDATETIME(),
    UpdatedAt DATETIME2 NULL,
    DeletedAt DATETIME2 NULL
);

-- Hotels
CREATE TABLE dbo.Hotels (
    HotelId INT IDENTITY(1,1) PRIMARY KEY,
    Name NVARCHAR(256) NOT NULL,
    Description NVARCHAR(MAX) NULL,
    Address NVARCHAR(512),
    City NVARCHAR(100),
    Country NVARCHAR(100),
    Latitude DECIMAL(9,6) NULL,
    Longitude DECIMAL(9,6) NULL,
    Phone NVARCHAR(30) NULL,
    Email NVARCHAR(256) NULL,
    IsActive BIT DEFAULT 1,
    CreatedAt DATETIME2 DEFAULT SYSUTCDATETIME(),
    UpdatedAt DATETIME2 NULL,
    DeletedAt DATETIME2 NULL
);
```