Develop a hotel management system using ASP.NET Core. The system will allow for the registration of Customers, management of room reservations, and viewing of reservation details. It should have functionalities for room and reservation management, including creating, editing, and listing rooms and reservations.

Database Structure

* Rooms Table: Store room details. Include fields such as RoomId, RoomNumber, Type, Capacity, DailyCost.
* Customers Table: Store Customer details. Include fields like CustomerId, Name, Email, Phone.
* Reservations Table: Store reservation details. Include fields like ReservationId, CustomerId, RoomId, CheckInDate, CheckOutDate.

Model Classes and Properties

* Room
* RoomId (int, primary key)
* RoomNumber (string)
* Type (string)
* Capacity (int)
* IsAvailable(bool)
* DailyCost (decimal)
* Customer
* CustomerId (int, primary key)
* Name (string)
* Email (string)
* Phone (string)
* Reservation
* ReservationId (int, primary key)
* CustomerId (int, foreign key)
* RoomId (int, foreign key)
* CheckInDate (DateTime)
* CheckOutDate (DateTime)

Relationships

* A Room can have multiple Reservations.
* A Reservation can have only one room
* A Customer can have multiple Reservations.
* A Reservation can have only one Customer
* Use Entity Framework Core to configure one-to-many relationships between these entities.

CRUD Operations for Pages

* Rooms: Create, View All , View Details, Edit, Delete room entries.
* Customers: Register (Create), View All , View Details (Read), Update, Delete Customers.
* Reservations: Create, View all , View Details, Edit, Cancel (Delete) reservations.

Validation Requirements

* Room: RoomNumber should be unique.
* Customer: Email should be a valid email format and unique.
* Reservation: CheckOutDate must be later than CheckInDate.

Business Logic Requirements

* Check room availability before confirming a reservation.
* Calculate total cost based on the number of days and room's daily cost.
* Prevent overlapping reservations for the same room.

Additional Requirements

* Exception Handling with ModelState
* Use ModelState to display error messages in views.
* DAL (Data Access Layer) Implementation
* Use Entity Framework Core for database operations.
* BLL (Business Logic Layer) Implementation