Thank you for reviewing this application; it was an interesting challenge to solve.
I will be happy to discuss the code in greater detail however I thought of sharing some design decision for the application and along with some other notes. The README file contains the necessary instructions to run the application.

1. The application architecture follows the Onion Architecture pattern, promoting separation of concerns.
2. Dependencies in the UI layer are kept minimal; service dependencies are managed within the Services project.
3. To assist with automation, I have included a Postman collection with requests and tests as part of the repository. Please import RoomNest.API.postman_collection.json and run the tests against a fresh database.
4. The database schema uses generous types like nvarchar(max) for flexibility. For production use, these should be carefully reviewed and adjusted according to business rules.
5. modelBuilder.ApplyConfigurationsFromAssembly(Assembly.GetExecutingAssembly()) is used to automatically apply entity configurations, so developers do not need to register new entities manually.
6. ModelState.IsValid validation is intentionally omitted at this stage. It is planned for the next iteration to include checks for over-posting, HTTPS enforcement, CORS policies, and other security features.
7. The current implementation focuses primarily on the happy path; more edge case handling will be added in the next iteration for robustness.
8. Hosting on Azure has not been done yet but is straightforward using Azure Kubernetes Service (AKS) or App Service. The application can run in Docker, although Kubernetes has some known issues; however deployment scripts are provided as references.
9. Comprehensive unit testing is not yet implemented. However, examples of testing different layers—API, Service, and in-memory database testing—are included. These examples use Automapper and NSubstitute to guide future test coverage improvements.
10. API versioning is not implemented but can be included in subsequent iterations.
11. As per instructions, all seeded hotels contain six rooms.
12. More advanced booking logic—such as check-in/check-out times and allowing same-day check-in when another guest checks out—is not implemented in this iteration.
13. Auto-migration is configured to simplify database updates when running the application directly in Visual Studio. Please verify and modify the connection string as needed. LocalDB typically comes with Visual Studio, so no additional infrastructure is required unless running in Docker or a full SQL Server environment.