Доброго дня! Сьогодні я представлю проект **Hotel Booking System API** – для управління готелем. Ця API дозволяє адміністраторам керувати номерами та користувачами, а клієнтам – бронювати номери та переглядати свої бронювання.

**1. Архітектура проєкту**

Він побудований за **багатошаровою архітектурою**:

**Перший шар HotelRoomReserved.API** → Відповідає за отримання та обробку HTTP-запитів.

**Другий шар HotelRoomReserved.Core** → Містить бізнес-логіку (DTO, сервіси, інтерфейси, мапінг, безпеку).

**Третій шар HotelRoomReserved.Storage** → Відповідає за взаємодію з базою даних через **EF Core**.

**Чертвертий шар HotelRoomReserved.Entities** → Містить моделі даних, які визначають структуру бази.

**2. Моделі та їх зв’язки**

У системі використовуються три основні моделі:

**User (Користувач)**

Id, FirstName, LastName, Email, Password, Role, CreatedAt

Один **User** може мати багато **Reservation**.

**Room (Номер)**

Id, RoomNumber, Type, Price, IsAvailable

Один **Room** може мати багато **Reservation**.

**Reservation (Бронювання)**

Id, RoomId, UserId, CheckInDate, CheckOutDate, Status

Відноситься до одного **User** і одного **Room**.

Зв’язки:  
🔹 **Один користувач** може мати **багато бронювань**  
🔹 **Одна кімната** може мати **багато бронювань**

**3. Контекст бази даних (HotelContext)**

Контекст HotelContext керує збереженням і взаємодією з базою:

DbSet<User> Users

DbSet<Room> Rooms

DbSet<Reservation> Reservations

**Та для забезпечення створення міграцій.**

**Особливості:**  
✔️ **OnModelCreating** налаштовує зв’язки, та **DeleteBehavior.Restrict** забороняє видаляти номери/користувачів, якщо є пов'язані бронювання

**4. Аутентифікація та безпека**

**JWT Authentication** використовується для захисту API.

Користувач отримує Токен після входу.

Токен використовується для ідентифікації користувача.

Всі дії вимагають авторизації (крім перегляду списку номерів).

**Додано Хешування паролів:**

Паролі не зберігаються у відкритому вигляді.  
Алгоритм дії:

✔️ Використовується **SHA-256**  
✔️ Додається **сіль** (CreatedAt)  
✔️ Хешований пароль зберігається в базі

**5. Бізнес-логіка (Services)**

Усі операції виконуються через сервіси:

**AuthService (Реєстрація, вхід, керування користувачами)**

✔️ Реєстрація користувачів (RegisterUser, RegisterAdmin)  
✔️ Логін та генерація токена (Login)  
✔️ Оновлення ролі користувача (UpdateUserRole)

✔️ Оновлення даних користувача (ім’я та прізвище)  
✔️ Видалення користувача  
✔️ Отримання списку користувачів (з пагінацією)

**RoomService (Керування номерами, тільки для Admin)**

✔️ Додавання номера (AddRoom)  
✔️ Оновлення номера (UpdateRoom)  
✔️ Видалення номера (DeleteRoom)  
✔️ Отримання списку номерів (GetRooms, GetRoomById)

**ReservationService (Керування бронюванням)**

✔️ Користувач може створювати бронювання (CreateReservation)  
✔️ Користувач може переглядати свої бронювання (GetUserReservations)  
✔️ Адміністратор може переглядати всі бронювання (GetAllReservations)  
✔️ Користувач може скасовувати бронювання (CancelReservation)

**6. Контролери**

Контролери обробляють запити та взаємодіють із сервісами.

**AuthController**

✔️ POST /auth/register-admin – Реєстрація адміністратора

✔️ POST /auth/register-user – Реєстрація користувача (може тільки Admin)  
✔️ POST /auth/login – Вхід та отримання JWT

✔️ GET /auth/users – Отримання списку користувачів (Admin)

✔️ PUT /auth/update-role/{id} – Оновлення ролі користувача (Admin)

✔️ PUT /auth/update-user/{id} – Оновлення даних користувача (Admin)  
✔️ DELETE /auth/delete-user/{id} – Видалення користувача (Admin)

**RoomController**

✔️ POST /rooms – Додати новий номер (Admin)  
✔️ PUT /rooms/{id} – Оновити номер (Admin)  
✔️ DELETE /rooms/{id} – Видалити номер (Admin)  
✔️ GET /rooms – Отримати список номерів (Доступно всім)

✔️ GET /rooms/{id} – Отримання номера за ID (Доступно всім)

**ReservationController**

✔️ POST /reservations – Створити бронювання (Користувач)  
✔️ GET /reservations/user – Переглянути свої бронювання  
✔️ GET /reservations – Переглянути всі бронювання (Admin)  
✔️ PUT /reservations/{id} – Скасувати бронювання

**7. Middleware та Exception Filter**

🔹 **LoggingMiddleware** – логування всіх запитів та відповідей  
🔹 **ExceptionFilter** – для обробки помилок (400, 404, 500)

**8. Висновки**

**Hotel Booking System API** – це сучасне рішення для керування бронюваннями номерів у готелі.

**Чому цей API вартий використання?:**

✔️ **Безпека** (JWT + хешування паролів)  
✔️ **Гнучкість** (поділ на шари, використання інтерфейсів)  
✔️ **Розширюваність** (легко додати нові функції)  
✔️ **Чистий код** (використання Dependency Injection, AutoMapper, DTO)

**Дякую за увагу!**