

## LINQ Queries in Hotel Reservation System

- 1) In SpecializedRepositories.cs-It filters users to include only Receptionists and Hotel Managers, then sorts them alphabetically by last name.

```
2 references
public async Task<IEnumerable<User>> GetStaffUsersAsync()
{
    return await _dbSet
        .Where(u => u.Role == "Receptionist" || u.Role == "HotelManager")
        .OrderBy(u => u.LastName)
        .ToListAsync();
}
```

- 2) In SpecializedRepositories.cs-It filters hotels to return only those whose **City name** contains the given city text, ignoring letter case.

```
2 references
public async Task<IEnumerable<Hotel>> GetHotelsByCityAsync(string city)
{
    return await _dbSet
        .Where(h => h.City.ToLower().Contains(city.ToLower()))
        .ToListAsync();
}
```

- 3) In SpecializedRepositories.cs-It fetches reservations for a specific user and sorts them from **most recent to oldest**.

```

public async Task<IEnumerable<Reservation>> GetReservationsByUserAsync(int userId)
{
    return await _dbSet
        .Include(r => r.Room)
        .ThenInclude(rm => rm.Hotel)
        .Include(r => r.User)
        .Where(r => r.UserId == userId)
        .OrderByDescending(r => r.CreatedAt)
        .ToListAsync();
}

```

4) In DataSeeder.cs-It checks **asynchronously** whether **any user record exists** in the Users table and returns **true** or **false**.

```

// 2. Seed Users (Now we can assign HotelId)
if (!await context.Users.AnyAsync())
{
    var firstHotel = await context.Hotels.FirstOrDefaultAsync();
    var hotelId = firstHotel?.Id;

    var users = new List<User>
    {
        new User { FirstName = "Admin", LastName = "User", Email = "admin@hotel.com", Password = "123456", Role = "Admin", HotelId = hotelId },
        // Assign Manager to First Hotel
        new User { FirstName = "Hotel", LastName = "Manager", Email = "manager@hotel.com", Password = "123456", Role = "Manager", HotelId = hotelId },
        // Assign Receptionist to First Hotel
        new User { FirstName = "Front", LastName = "Desk", Email = "receptionist@hotel.com", Password = "123456", Role = "Receptionist", HotelId = hotelId },
        new User { FirstName = "John", LastName = "Doe", Email = "john.doe@email.com", Password = "123456", Role = "User", HotelId = hotelId },
        new User { FirstName = "Jane", LastName = "Smith", Email = "jane.smith@email.com", Password = "123456", Role = "User", HotelId = hotelId },
    };
    await context.Users.AddRangeAsync(users);
    await context.SaveChangesAsync();
}

```

5) In DataSeeder.cs-It **asynchronously** retrieves the first hotel record from the database, or returns **null** if no hotel exists.

```

if (!await context.Users.AnyAsync())
{
    var firstHotel = await context.Hotels.FirstOrDefaultAsync();
    var hotelId = firstHotel?.Id;

    var users = new List<User>
    {
        new User { FirstName = "Admin", LastName = "User", Email = "admin@hotel.com", PasswordHash = "1qaz!@WSX" },
        // Assign Manager to First Hotel
        new User { FirstName = "Hotel", LastName = "Manager", Email = "manager@hotel.com", PasswordHash = "1qaz!@WSX" },
        // Assign Receptionist to First Hotel
        new User { FirstName = "Front", LastName = "Desk", Email = "receptionist@hotel.com", PasswordHash = "1qaz!@WSX" },
        new User { FirstName = "John", LastName = "Doe", Email = "john.doe@email.com", PasswordHash = "1qaz!@WSX" },
        new User { FirstName = "Jane", LastName = "Smith", Email = "jane.smith@email.com", PasswordHash = "1qaz!@WSX" },
    };
    await context.Users.AddRangeAsync(users);
    await context.SaveChangesAsync();
}

```

6) In DataSeeder.cs-It filters the users to include **only those with the role “Guest”** and converts the result into a list.

```

var guestUsers = users.Where(u => u.Role == "Guest").ToList();
var availableRooms = rooms.Take(5).ToList();
var reservations = new List<Reservation>();

```

7) In DataSeeder.cs-It selects the **first 5 rooms** from the collection and returns them as a list.

```

var users = await context.Users.ToListAsync(); // Fetch existing users
var rooms = await context.Rooms.ToListAsync(); // Fetch existing rooms
var guestUsers = users.Where(u => u.Role == "Guest").ToList();
var availableRooms = rooms.Take(5).ToList();
var reservations = new List<Reservation>();

```