Hotel Management System – Specification Document v2.0

Authors: ESSETTI Yasmine, MASMOUDI Nour
Date: 2024/2025

1. Static Aspects - Class Diagram

Business Classes Identified:

1. **User** (Abstract)
    - Attributes: userId, firstName, lastName, email, phoneNumber, password
    - Subclasses:
        - **Guest**
        - **Administrator**

User
- String email - String password - int userId - String firstName - String lastName - String phoneNumber

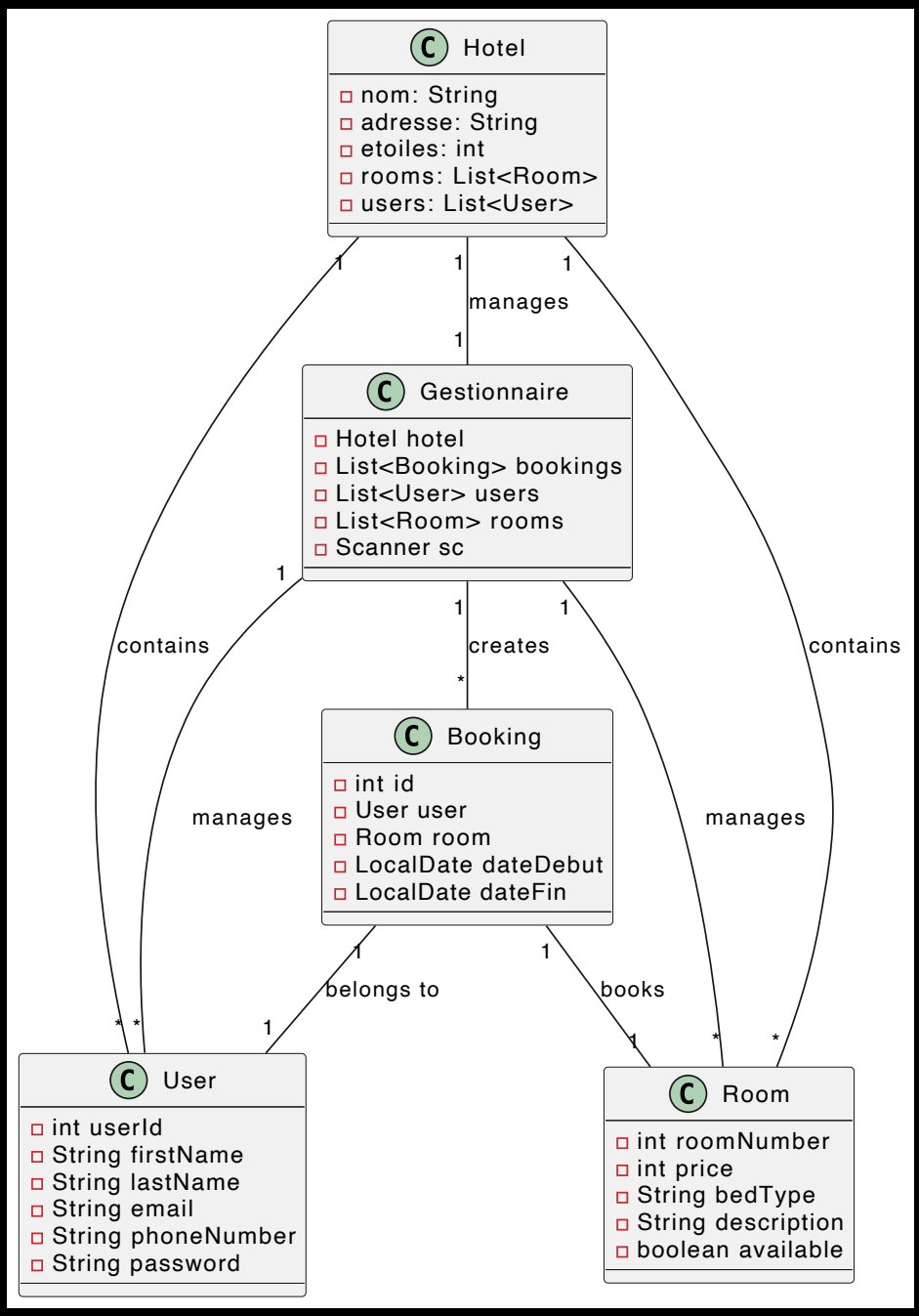
2. **Room**
    - Attributes: roomNumber, price, bedType, description, available (boolean)

Room
- int roomNumber - int price - String bedType - String description - boolean available

3. **Booking**
    - Attributes: bookingId, startDate, endDate, totalPrice, status
    - Associations:
        - 1 User (either Guest or Admin)
        - 1 Room

Booking
- int id - User user - Room room - LocalDate dateDebut - LocalDate dateFin - boolean paid

Class Diagram

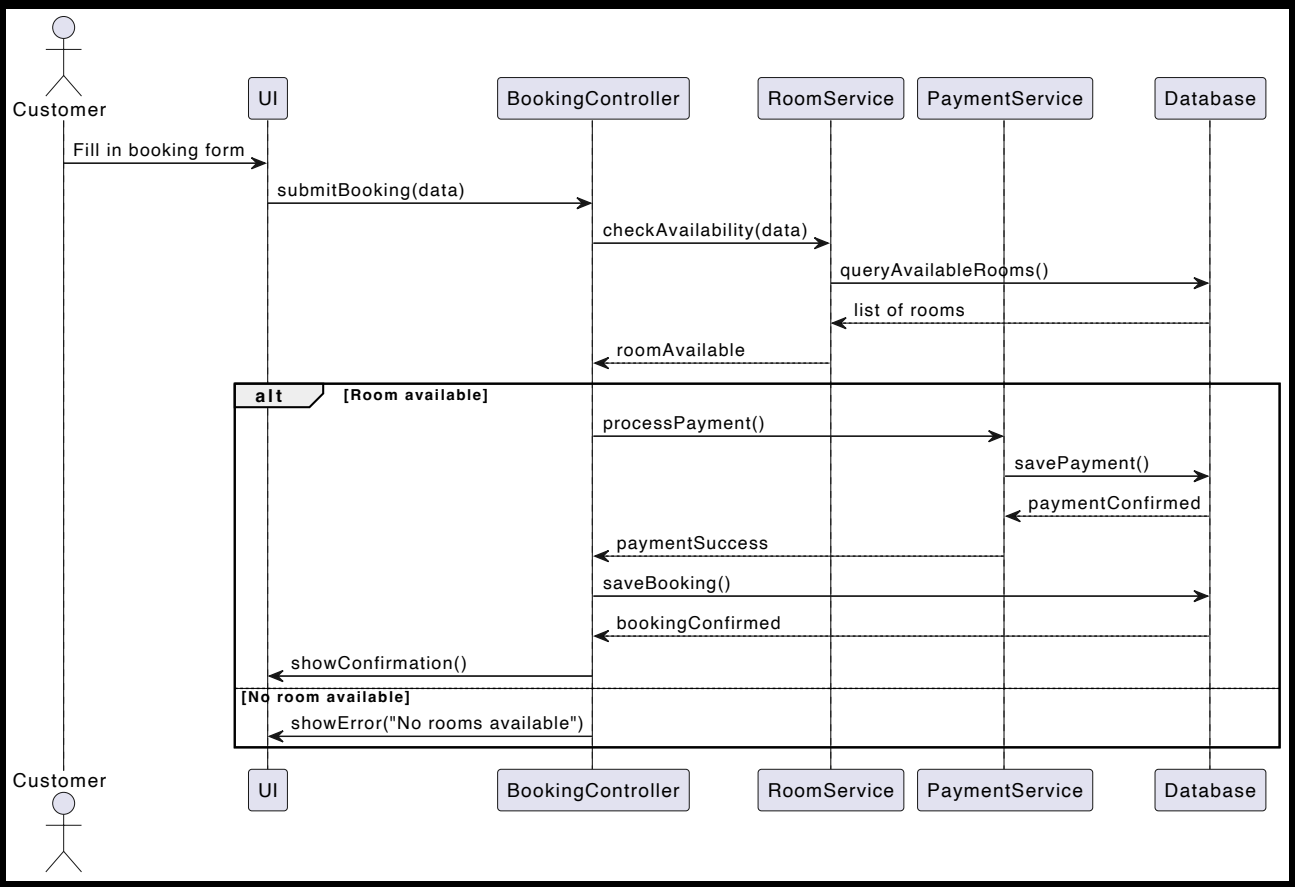


2. Dynamic Aspects - Sequence Diagrams

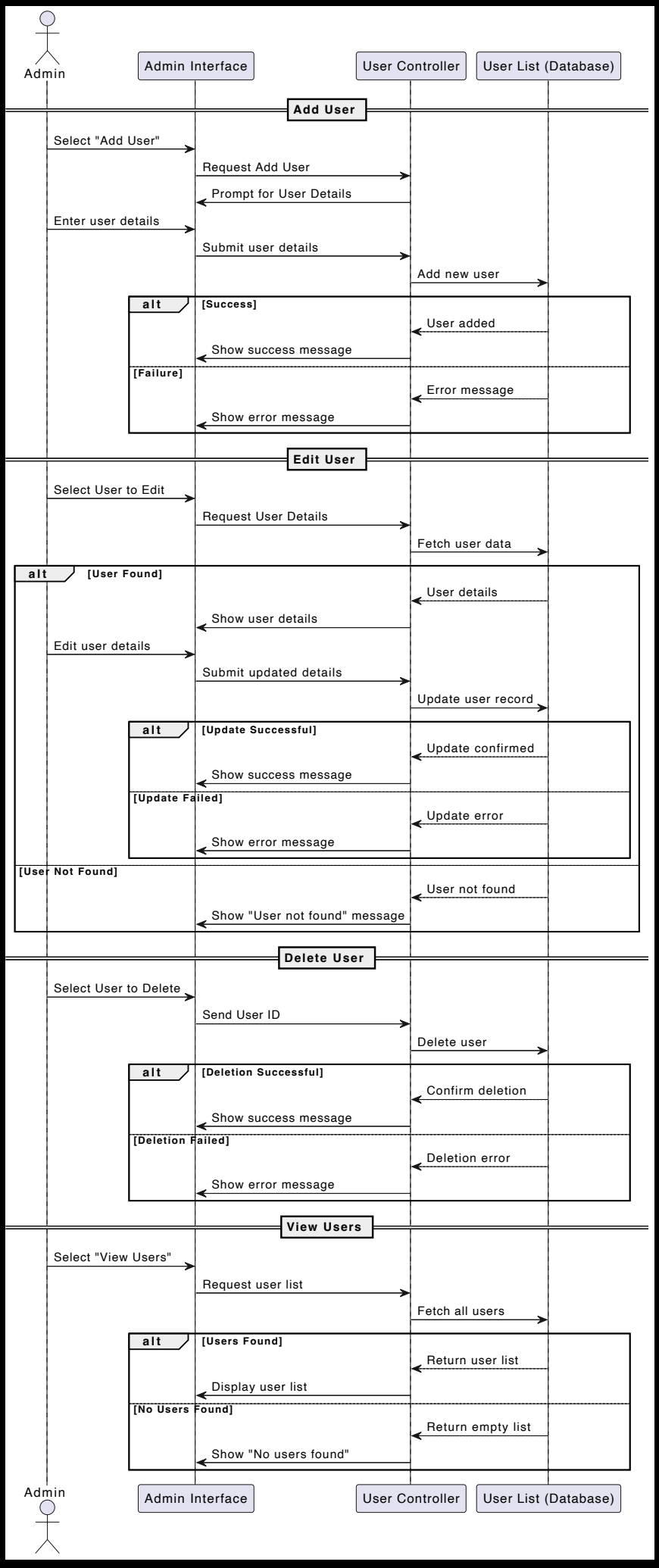
Selected Use Cases for Sprint 1:

1. Room Booking
2. User management

2.1 Room Booking (DSUC1)



2.2 User management (DSUC2)


Admin
Admin Interface
User Controller
User List (Database)
Add User
Select "Add User"
Request Add User
Prompt for User Details
Enter user details
Submit user details
Add new user
alt
[Success]
User added
Show success message
[Failure]
Error message
Show error message
Edit User
Select User to Edit
Request User Details
Fetch user data
alt
[User Found]
User details
Show user details
Edit user details
Submit updated details
Update user record
alt
[Update Successful]
Update confirmed
Show success message
[Update Failed]
Update error
Show error message
[User Not Found]
User not found
Show "User not found" message
Delete User
Select User to Delete
Send User ID
Delete user
alt
[Deletion Successful]
Confirm deletion
Show success message
[Deletion Failed]
Deletion error
Show error message
View Users
Select "View Users"
Request user list
Fetch all users
alt
[Users Found]
Return user list
Display user list
[No Users Found]
Return empty list
Show "No users found"
Admin
Admin Interface
User Controller
User List (Database)

Admin Use Cases

1. **Admin Login**
   - Description: Allows the administrator to securely log into the system.
   - Precondition: Admin credentials must exist.

PlantUML:

```
@startuml
actor Admin
participant "Login Interface" as UI
participant "Auth Controller" as Controller
participant "Admin DB" as DB

Admin -> UI : Enter username and password
UI -> Controller : Submit login credentials
Controller -> DB : Validate credentials

alt Login Successful
    DB --> Controller : Credentials valid
    Controller -> UI : Redirect to Admin Dashboard
    UI -> Admin : Access granted
else Login Failed
    DB --> Controller : Invalid credentials
    Controller -> UI : Show error message
    UI -> Admin : Display "Login failed"
end
@enduml
```

2. **Manage Users**
   - Description: Allows the administrator to add, remove, or update user information.
   - Precondition: Admin must be logged in.

PlantUML:

```
@startuml
actor Admin
participant "Admin Interface" as UI
participant "User Controller" as Controller
participant "User List (Database)" as DB

== Add User ==
Admin -> UI : Select "Add User"
UI -> Controller : Request Add User
```

```
Controller -> UI : Prompt for User Details
Admin -> UI : Enter user details
UI -> Controller : Submit user details
Controller -> DB : Add new user

alt Success
    DB --> Controller : User added
    Controller -> UI : Show success message
else Failure
    DB --> Controller : Error message
    Controller -> UI : Show error message
end

== Edit User ==
Admin -> UI : Select User to Edit
UI -> Controller : Request User Details
Controller -> DB : Fetch user data

alt User Found
    DB --> Controller : User details
    Controller -> UI : Show user details
    Admin -> UI : Edit user details
    UI -> Controller : Submit updated details
    Controller -> DB : Update user record

    alt Update Successful
        DB --> Controller : Update confirmed
        Controller -> UI : Show success message
    else Update Failed
        DB --> Controller : Update error
        Controller -> UI : Show error message
    end

else User Not Found
    DB --> Controller : User not found
    Controller -> UI : Show "User not found" message
end
```

```
== Delete User ==
Admin -> UI : Select User to Delete
UI -> Controller : Send User ID
Controller -> DB : Delete user

alt Deletion Successful
    DB --> Controller : Confirm deletion
    Controller -> UI : Show success message
else Deletion Failed
    DB --> Controller : Deletion error
    Controller -> UI : Show error message
end

== View Users ==
Admin -> UI : Select "View Users"
UI -> Controller : Request user list
Controller -> DB : Fetch all users

alt Users Found
    DB --> Controller : Return user list
    Controller -> UI : Display user list
else No Users Found
    DB --> Controller : Return empty list
    Controller -> UI : Show "No users found"
end
@enduml
```

3. **Manage Rooms**
   - Description: Allows the administrator to add, update, or remove rooms.
   - Precondition: Admin must be logged in.

PlantUML:

```
@startuml
actor Admin
participant "Admin Interface" as UI
participant "Room Controller" as Controller
participant "Room Database" as DB

== Add Room ==
```

```
Admin -> UI : Select "Add Room"
UI -> Controller : Request Room Details
Controller -> UI : Prompt for room info
Admin -> UI : Enter room details
UI -> Controller : Submit room details
Controller -> DB : Add room

alt Success
    DB --> Controller : Room added
    Controller -> UI : Show success message
else Failure
    DB --> Controller : Error adding room
    Controller -> UI : Show error message
end

== Edit Room ==
Admin -> UI : Select Room to Edit
UI -> Controller : Request Room Data
Controller -> DB : Fetch Room Details

alt Room Found
    DB --> Controller : Room data
    Controller -> UI : Display Room Info
    Admin -> UI : Update room info
    UI -> Controller : Submit updated details
    Controller -> DB : Update Room Info

    alt Success
        DB --> Controller : Update confirmed
        Controller -> UI : Show success message
    else Failure
        DB --> Controller : Update error
        Controller -> UI : Show error message
    end

else Room Not Found
    DB --> Controller : Room not found
    Controller -> UI : Show "Room not found"
```

```
end

== Delete Room ==
Admin -> UI : Select Room to Delete
UI -> Controller : Send Room ID
Controller -> DB : Delete Room

alt Success
    DB --> Controller : Deletion confirmed
    Controller -> UI : Show success message
else Failure
    DB --> Controller : Deletion error
    Controller -> UI : Show error message
end

== View Rooms ==
Admin -> UI : Select "View Rooms"
UI -> Controller : Request Room List
Controller -> DB : Fetch all rooms

alt Rooms Found
    DB --> Controller : List of rooms
    Controller -> UI : Display room list
else No Rooms Found
    DB --> Controller : Empty list
    Controller -> UI : Show "No rooms available"
end
@enduml
```

2. User Use Cases

2.1. User Registration

Description: Allows new users to create an account. Precondition: User must provide required information.

PlantUML:

```
@startuml
actor Guest
participant "Registration Interface" as UI
participant "User Controller" as Controller
```

```
participant "User Database" as DB

Guest -> UI : Open Registration Form
UI -> Guest : Prompt for user details
Guest -> UI : Enter name, email, password, etc.
UI -> Controller : Submit registration details
Controller -> DB : Check if user already exists

alt User Exists
    DB --> Controller : Email already in use
    Controller -> UI : Show error "User already exists"
    UI -> Guest : Display error
else User Does Not Exist
    Controller -> DB : Create new user account

    alt Success
        DB --> Controller : User created
        Controller -> UI : Show success message
        UI -> Guest : "Registration successful"
    else Failure
        DB --> Controller : Error during creation
        Controller -> UI : Show error message
        UI -> Guest : "Registration failed"
    end
end
@enduml
```

2.4. **Book Room**

- Description: Allows users to book a room after checking its availability.
- Precondition: User must be logged in, and room must be available.

PlantUML:

```
@startuml
actor Customer
participant UI
participant BookingController
participant RoomService
participant PaymentService
participant Database
```

```
Customer -> UI : Fill in booking form
UI -> BookingController : submitBooking(data)
BookingController -> RoomService : checkAvailability(data)
RoomService -> Database : queryAvailableRooms()
Database --> RoomService : list of rooms
RoomService --> BookingController : roomAvailable

alt Room available
  BookingController -> PaymentService : processPayment()
  PaymentService -> Database : savePayment()
  Database --> PaymentService : paymentConfirmed
  PaymentService --> BookingController : paymentSuccess
  BookingController -> Database : saveBooking()
  Database --> BookingController : bookingConfirmed
  BookingController -> UI : showConfirmation()
else No room available
  BookingController -> UI : showError("No rooms available")
end
@enduml
```

2.5. **Cancel Booking**

Description: Allows users to cancel their Booking.

Precondition: User must be logged in, and booking must be valid.

PlantUML:

```
@startuml
actor User
participant "Booking UI" as UI
participant "Booking Controller" as Controller
participant "Booking Database" as DB

User -> UI : Select "My Bookings"
UI -> Controller : Request user bookings
Controller -> DB : Fetch bookings by user ID
DB --> Controller : Return booking list
Controller -> UI : Display booking list

User -> UI : Click "Cancel" on booking
UI -> Controller : Request cancellation
```

```
Controller -> DB : Verify booking status

alt Booking Exists and is Cancellable
    DB --> Controller : Booking active
    Controller -> DB : Cancel booking
    DB --> Controller : Cancellation confirmed
    Controller -> UI : Show success message
    UI -> User : "Booking cancelled successfully"

else Booking Already Used / Not Found
    DB --> Controller : Not cancellable
    Controller -> UI : Show error message
    UI -> User : "Cancellation failed"
end
@enduml
```