Tuwaiq Academy أكاديمية طويق

Lab-6

MyApartment

MyApartment
A system for booking apartments

Instructor: Majd Alzahrani

Osama Alahmadi

Table of Content

Introduction

Tourism is one of the most important sectors in the kingdom in Saudi Arabia. The sector has grown significantly, boosting the national economy and increasing both domestic and international tourism, as well as attracting more investors and creating more economic and investment opportunities.

Tourism applications are among the most important drivers of tourism economic growth, as they enable and facilitate the provision of tourism services using technology. Multiple digital platforms are available that offer various tourism services, such as hotel booking applications, flight booking applications, applications for tourist activities and events, and applications for restaurant and venue reviews.

And that's where our system come in use.

System brief

My Apartment should provide housing and help tourist and people book apartment online, as well as allowing apartments owner to list their apartment for customers to book.

The system should have 7 models:

1- Customer: users that wants to book apartments.

2- Apartment-Owners: users that lists their apartments.

3- Reservation: Customers reservations.

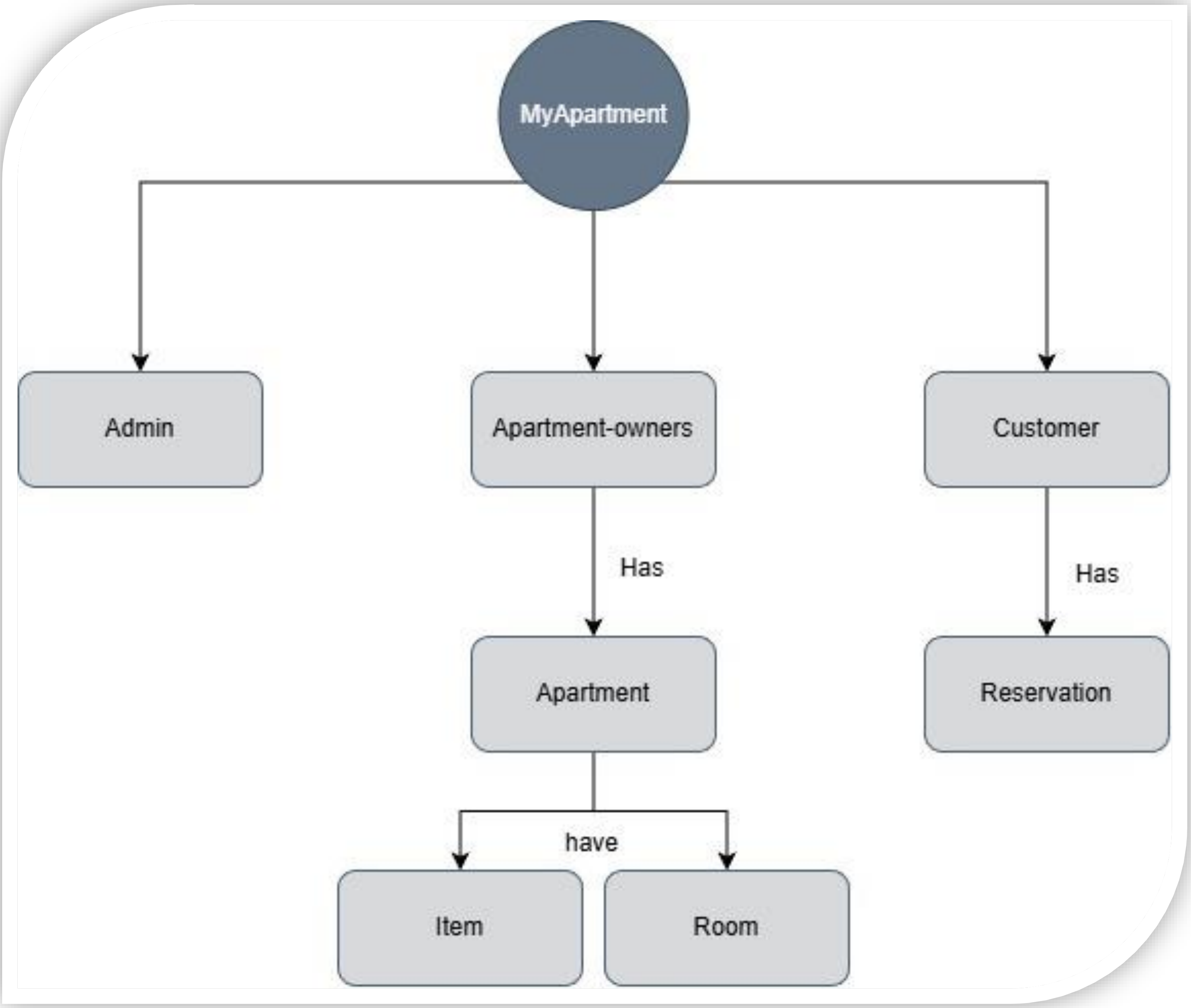
4- Apartment: apartments listed.

5- Room: room information in each apartment

6- Item: items provided for each apartment (food, cloths, pillows, …)

7-Admin: for system owners.

System Components and their relation



Models and validation

When creating those models, we must specify their data and the validation applied to them, so we will define some of those models as well as showing their validations rules and criteria.

Customer: the model has: ID, Username, password, email, phone number, and penalty (money the customer owes).

Validation

ID:
- Cannot be null
- Length must be exactly 6
- Can only be digits

Username:
- Cannot be null
- Length must be more than 5
- Can include characters, number and special characters.

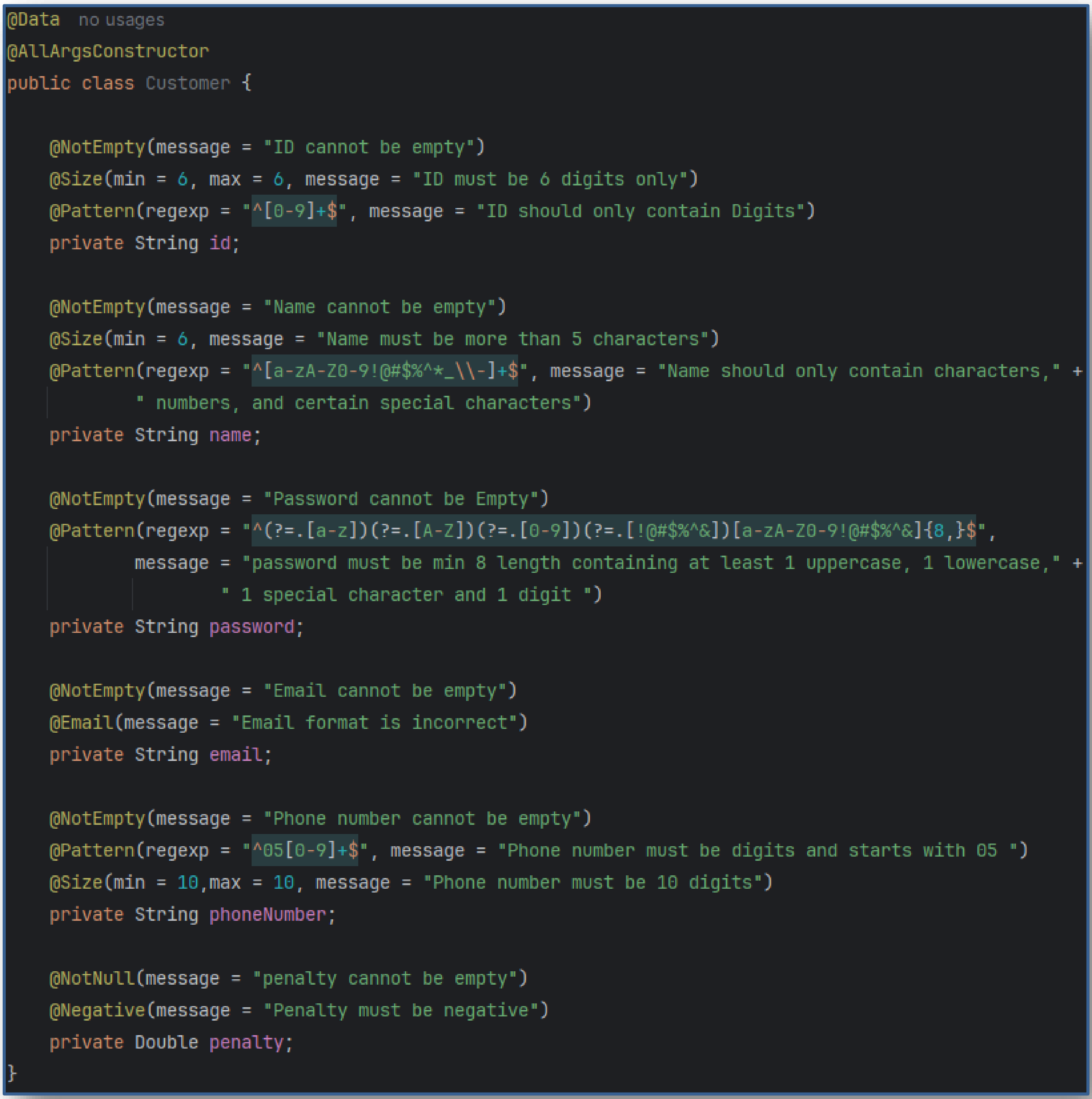
Password:
- Cannot be null
- Length must be more than
- Can include characters, number and special characters.

Email:
- Cannot be null
- Must be Email formatted

Phone number:
- Cannot be null
- Must starts with 05 and followed by 8 digits

Penalty:
- Cannot be null
- Cannot be positive

Customer Validation code



```java
@Data  no usages
@AllArgsConstructor
public class Customer {

    @NotEmpty(message = "ID cannot be empty")
    @Size(min = 6, max = 6, message = "ID must be 6 digits only")
    @Pattern(regexp = "^[0-9]+$", message = "ID should only contain Digits")
    private String id;

    @NotEmpty(message = "Name cannot be empty")
    @Size(min = 6, message = "Name must be more than 5 characters")
    @Pattern(regexp = "^[a-zA-Z0-9!@#$%^*_\\-]+$", message = "Name should only contain characters," +
            " numbers, and certain special characters")
    private String name;

    @NotEmpty(message = "Password cannot be Empty")
    @Pattern(regexp = "^(?=.[a-z])(?=.[A-Z])(?=.[0-9])(?=.[!@#$%^&])[a-zA-Z0-9!@#$%^&]{8,}$",
            message = "password must be min 8 length containing at least 1 uppercase, 1 lowercase," +
                    " 1 special character and 1 digit ")
    private String password;

    @NotEmpty(message = "Email cannot be empty")
    @Email(message = "Email format is incorrect")
    private String email;

    @NotEmpty(message = "Phone number cannot be empty")
    @Pattern(regexp = "^05[0-9]+$", message = "Phone number must be digits and starts with 05 ")
    @Size(min = 10,max = 10, message = "Phone number must be 10 digits")
    private String phoneNumber;

    @NotNull(message = "penalty cannot be empty")
    @Negative(message = "Penalty must be negative")
    private Double penalty;
}
```

Reservation: the model has: reservartionID, bookingDate, startDate, endDate, price, status, and Customer (Relation).

Validation

reservartionID:
- Cannot be null
- Length must be exactly 10
- Can only be digits

bookingDate:
- Cannot be null

startDate:
- Cannot be null
- Date Should be in the Future or present

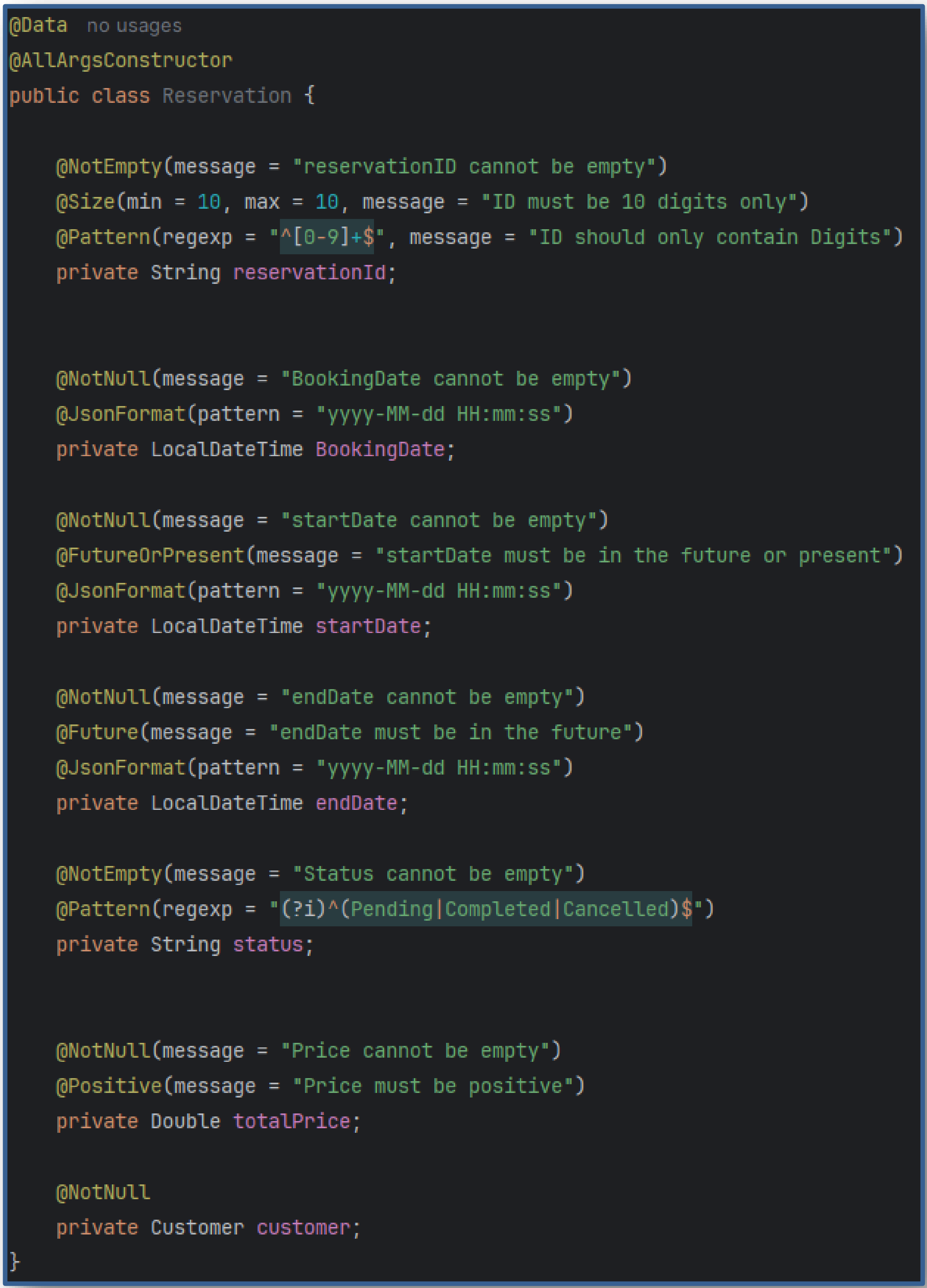
EndDate:
- Cannot be null
- Date Should be in the Future.

TotalPrice:
- Cannot be null
- must be positive

Status:
- Cannot be null
- Must be Pending or Completed or Cancelled only

Customer:
- Cannot be null

Reservation Validation code



```
@Data  no usages
@AllArgsConstructor
public class Reservation {

    @NotEmpty(message = "reservationID cannot be empty")
    @Size(min = 10, max = 10, message = "ID must be 10 digits only")
    @Pattern(regexp = "^[0-9]+$", message = "ID should only contain Digits")
    private String reservationId;


    @NotNull(message = "BookingDate cannot be empty")
    @JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss")
    private LocalDateTime BookingDate;

    @NotNull(message = "startDate cannot be empty")
    @FutureOrPresent(message = "startDate must be in the future or present")
    @JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss")
    private LocalDateTime startDate;

    @NotNull(message = "endDate cannot be empty")
    @Future(message = "endDate must be in the future")
    @JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss")
    private LocalDateTime endDate;

    @NotEmpty(message = "Status cannot be empty")
    @Pattern(regexp = "(?i)^(Pending|Completed|Cancelled)$")
    private String status;


    @NotNull(message = "Price cannot be empty")
    @Positive(message = "Price must be positive")
    private Double totalPrice;

    @NotNull
    private Customer customer;
}
```

Apartment: the model has: ApartmentID, City, Location (inside the city), status(is it available or not).

Validation

ApartmentID:
- Cannot be null
- Length must be exactly 6
- Can only be digits

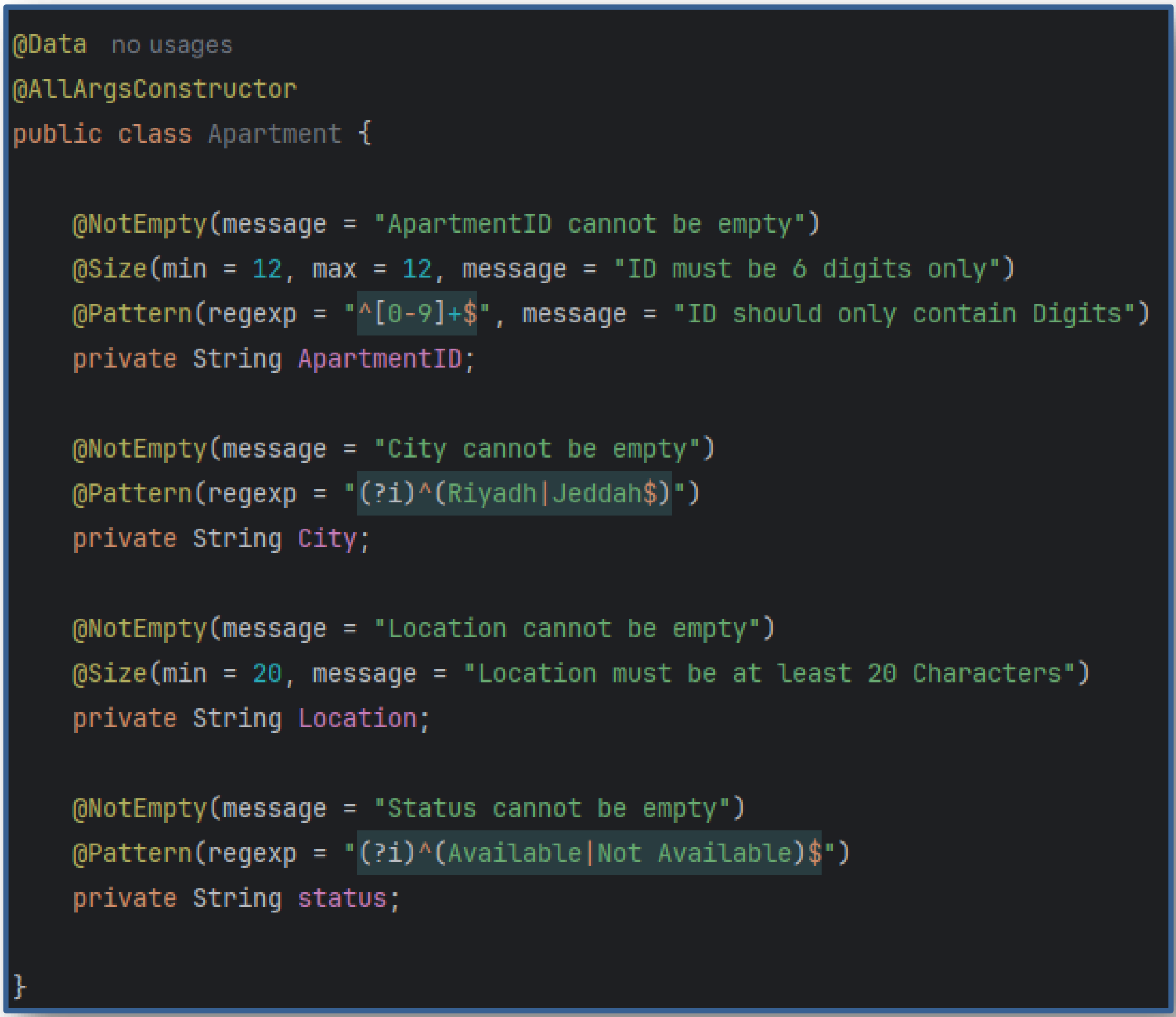
City:
- Cannot be null
- Must be Riyadh or Jedda only

Location:
- Cannot be null
- Length must be at least 20 characters

Status:
- Cannot be null
- Must be Available or not Available only

MyApartment

Apartment Validation code



```
@Data  no usages
@AllArgsConstructor
public class Apartment {

    @NotEmpty(message = "ApartmentID cannot be empty")
    @Size(min = 12, max = 12, message = "ID must be 6 digits only")
    @Pattern(regexp = "^[0-9]+$", message = "ID should only contain Digits")
    private String ApartmentID;

    @NotEmpty(message = "City cannot be empty")
    @Pattern(regexp = "(?i)^(Riyadh|Jeddah$)")
    private String City;

    @NotEmpty(message = "Location cannot be empty")
    @Size(min = 20, message = "Location must be at least 20 Characters")
    private String Location;

    @NotEmpty(message = "Status cannot be empty")
    @Pattern(regexp = "(?i)^(Available|Not Available)$")
    private String status;

}
```