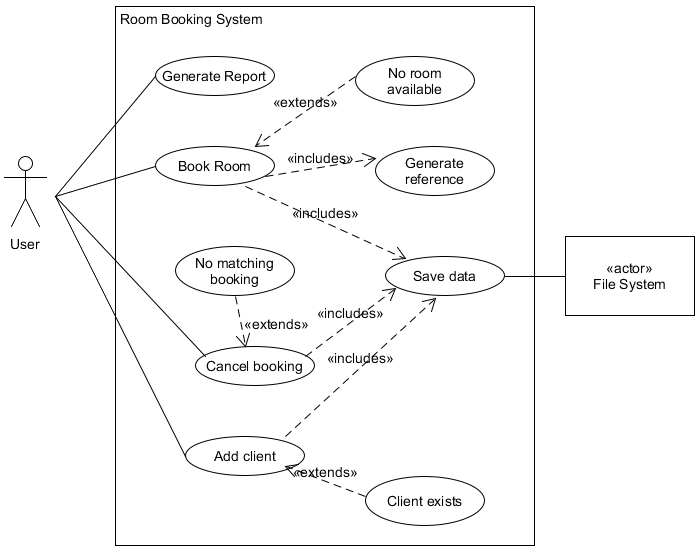
### Alternative Design Option

### Stage 1 – Use Case Diagram



### Stage 1 – Use Case Descriptions

#### Use Case 1

**Name**: Book Room

**Primary Actor(s)**: User, File System

**Description**: the user should be able to book a room for a client. They should specify the client name and the date, time and duration of the booking. They should also specify the number of workstation seats required in the room. The system should then search for available rooms and offer the best match for their requirements. The user may accept or decline the offer. If they accept the offer, they will be given a reference number. If there are no rooms available, the system should report that. The system should update the stored booking information.

**Pre-conditions**: the user has been presented with the input for a room booking.

**Post-conditions**: a room is successfully booked for the client and the updated information is stored.

**Trigger event:** the user confirms the details and requests a booking.

### Scenario

#### Normal Flow

1. The client name is retrieved from the ui.
2. The name is validated against the list of clients.
   1. The name must be present in the existing client list.
3. The required date is retrieved from the ui.
4. The date format is validated.
   1. The date must be in the ISO 8601 format (YYYY-MM-DD)
5. The required time is retrieved from the ui.
6. The time format is validated.
   1. The time must be in the ISO 8601 24 hour format (HH:MM)
7. The date/time is checked to be in the future.
8. The required number of workstations is retrieved from the ui.
9. The required number of seats is retrieved from the ui.
10. Find the best match for the client requirements.
    1. For each room in the system
       1. Check the room has enough seats to meet the requirement
       2. Check the room is bookable at the requested time
       3. If the room has fewer seats than the current best match, this room is now the best match
11. Offer the room to the client.
12. Book the room for the client.
13. Get the booking reference for the new booking.
14. Save the updated state of the system.

***Extension Flows***

1a The client name is missing from the ui.

1. Report the error to the user
2. terminate the use case.

2a The client does not exist in the list.

1. Report the error to the user
2. terminate the use case.

3a The date is missing from the ui.

1. Report the error to the user
2. terminate the use case.

4a The input cannot be converted to a properly formatted date.

1. Report the error to the user
2. terminate the use case.

5a The time is missing from the ui.

1. Report the error to the user
2. terminate the use case.

6a The input cannot be converted to a properly formatted time.

1. Report the error to the user
2. terminate the use case.

7a The time is in the past

1. Report the error to the user
2. terminate the use case

10a No match can be found

1. Report the issue to the user
2. terminate the use case

11a The client declines the booking

1. Terminate the use case

14a There is an IO error saving the state

1. Report the error to the user
2. Terminate the use case

#### Use Case 2

**Name**: Generate Client Booking Report

**Primary Actor(s)**: User

**Description**: the user should be able to ask for a report. They can ask for a report of all the bookings made by a particular client.

**Pre-conditions**: none.

**Post-conditions**: a formatted report is displayed to the user.

**Trigger event:** the user selects the report option.

### Scenario

#### Normal Flow

1. The client name is retrieved from the ui
2. The name is validated against the list of clients
   1. The name must be present in the existing client list
3. Find all the bookings for the given client
   1. Make an initially empty list of client bookings
   2. For each room in the system
      1. Get the list of bookings for that room
      2. For each booking in the room booking list
         1. If the booking’s client matches the given client then add the booking to the list of client bookings
4. Format the list of client bookings into a table
5. Display the table to the user

#### Extension Flows

1a The client name is missing from the ui.

1. Report the error to the user
2. terminate the use case.

2a The client does not exist in the list.

1. Report the error to the user
2. terminate the use case.

4a The client booking list is empty

1. Format an empty table.

#### Use Case 3

**Name**: Cancel Booking

**Primary Actor(s)**: User, File System

**Description**: the user should be able to enter a booking number and the system should delete the booking that corresponds to the given reference. The system should update the stored booking information.

**Pre-conditions**: none.

**Post-conditions**: the specified booking is removed from the system.

**Trigger event:** the user selects the cancel booking option.

### Scenario

#### Normal Flow

1. Get the booking reference from the ui
2. The number format is validated
   1. The number has to be a positive integer less than the current max reference value
3. Set cancelled flag to false
4. For each room in the system
   1. For each booking in the room
      1. If the reference of the room booking matches the given reference
         1. Remove the booking from the list
         2. Set cancelled flag to true
         3. Exit from step 4
5. If the cancelled flag is true
   1. Report the booking was cancelled
   2. Save the update state of the system.

#### Extensions

1a The booking reference is missing from the ui.

1. Report the error to the user
2. Terminate the use case.

2a The booking reference is not a number.

1. Report the error to the user
2. Terminate the use case.

5a The cancelled flag is false

1. Report that the booking could not be found.

#### Use Case 4

**Name**: Add Client

**Primary Actor(s)**: User, File System

**Description**: the user should be able to add a new client to the system. They must supply the family and given names of the client along with a contact email and contact phone number. The system should add the new client to the system and update the stored information.

**Pre-conditions**: the Add Client dialog has been displayed to the user.

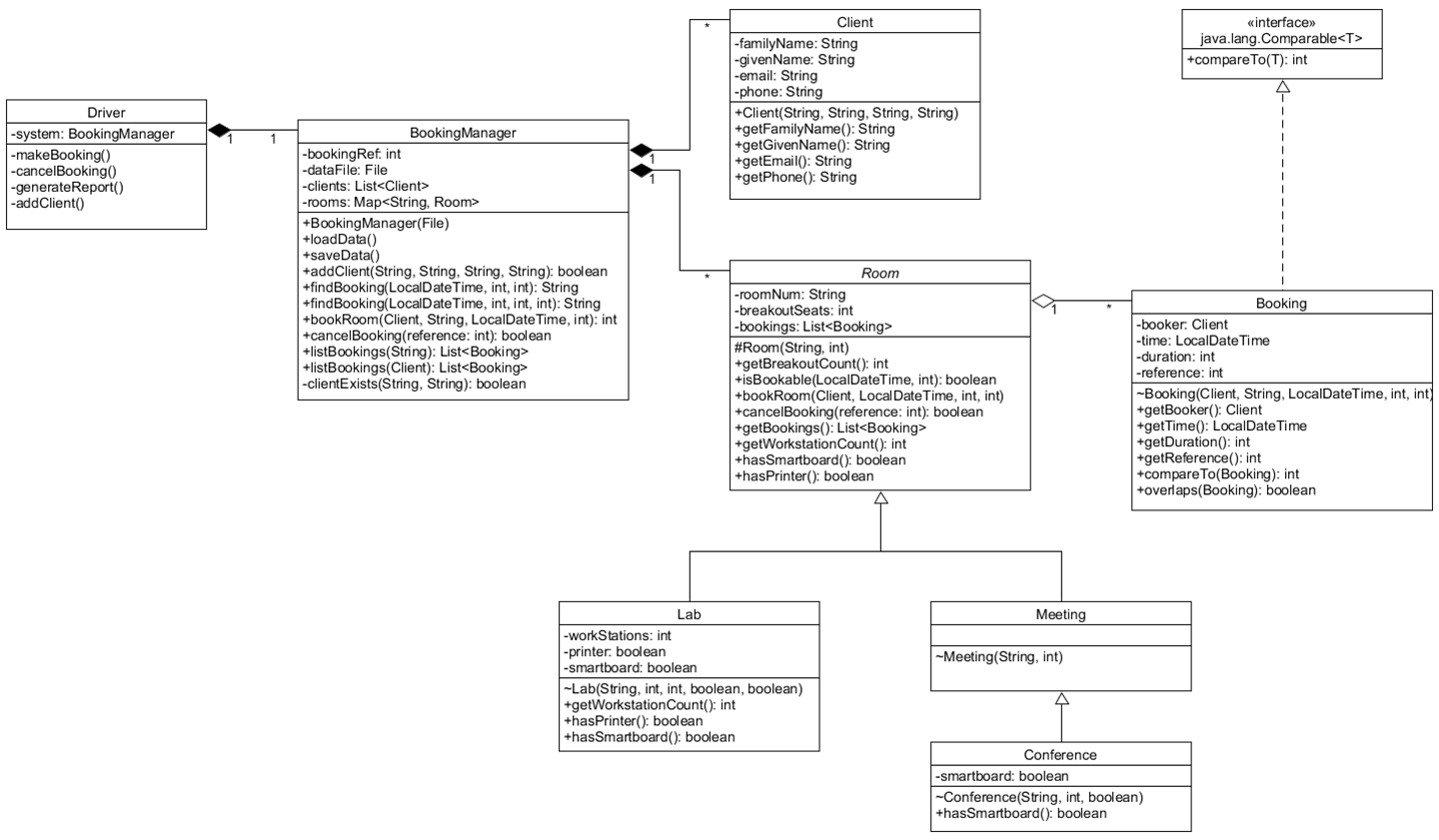
**Post-conditions**: the new client details are added to the system.

**Trigger event:** the user clicks the OK button on the add user dialog.

### Scenario

#### Normal Flow

1. The family name is retrieved from the ui.
2. The family name is validated.
   1. The name must be a non-empty string
   2. The first character must be an upper case letter
   3. The remaining characters must be lower case letters
3. The given name is retrieved from the ui.
4. The given name is validated.
   1. The name must be a non-empty string
   2. The first character must be an upper case letter
   3. The remaining characters must be lower case letters
5. The email is retrieved from the ui.
6. The email format is validated.
   1. The regular expression given below will be used for this[[1]](#footnote-1)
7. The phone number is retrieved from the ui.
8. The phone number is validated.
   1. The string must be 10 or 11 characters long
   2. The string must begin with the sequence “07”
   3. The remaining characters in the string must be digits
9. The combined family & given name is checked to be unique
   1. This combination must not appear in the existing client list
10. A new client with the given information is added to the system
11. A message confirming the new client is displayed
12. The updated state of the system is saved



1. The following regular expression from <https://www.regular-expressions.info/email.html> will be used to validate the email address: ^(?=[A-Za-z0-9][A-Za-z0-9@.\_%+-]{5,253}+$)[A-Za-z0-9.\_%+-]{1,64}+@(?:(?=[A-Za-z0-9-]{1,63}+\\.)[A-Za-z0-9]++(?:-[A-Za-z0-9]++)\*+\\.){1,8}+[A-Za-z]{2,63}+$ [↑](#footnote-ref-1)