STUDY SPACE

Concurrent and Distributed Systems project

Lotano Sara

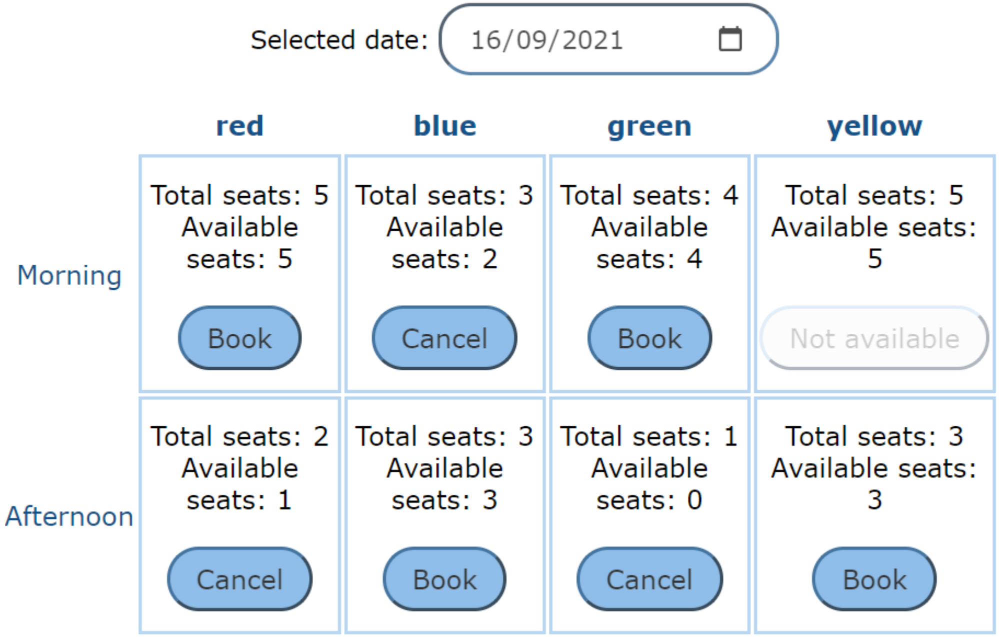
Academic Year 2021/2022

APPLICATION OVERVIEW

- **Study Space** is a web application that allows users to reserve a timeslot in a library in order to consult books, use library services or just to study in a quiet place.

- The library consists of multiple rooms (**red**, **blue, green and yellow**)

- There are two different timeslots: **Morning** and **Afternoon**



Selected date: 16/09/2021

	red	blue	green	yellow
Morning	Total seats: 5 Available seats: 5 Book	Total seats: 3 Available seats: 2 Cancel	Total seats: 4 Available seats: 4 Book	Total seats: 5 Available seats: 5 Not available
Afternoon	Total seats: 2 Available seats: 1 Cancel	Total seats: 3 Available seats: 3 Book	Total seats: 1 Available seats: 0 Cancel	Total seats: 3 Available seats: 3 Book

APPLICATION ACTORS

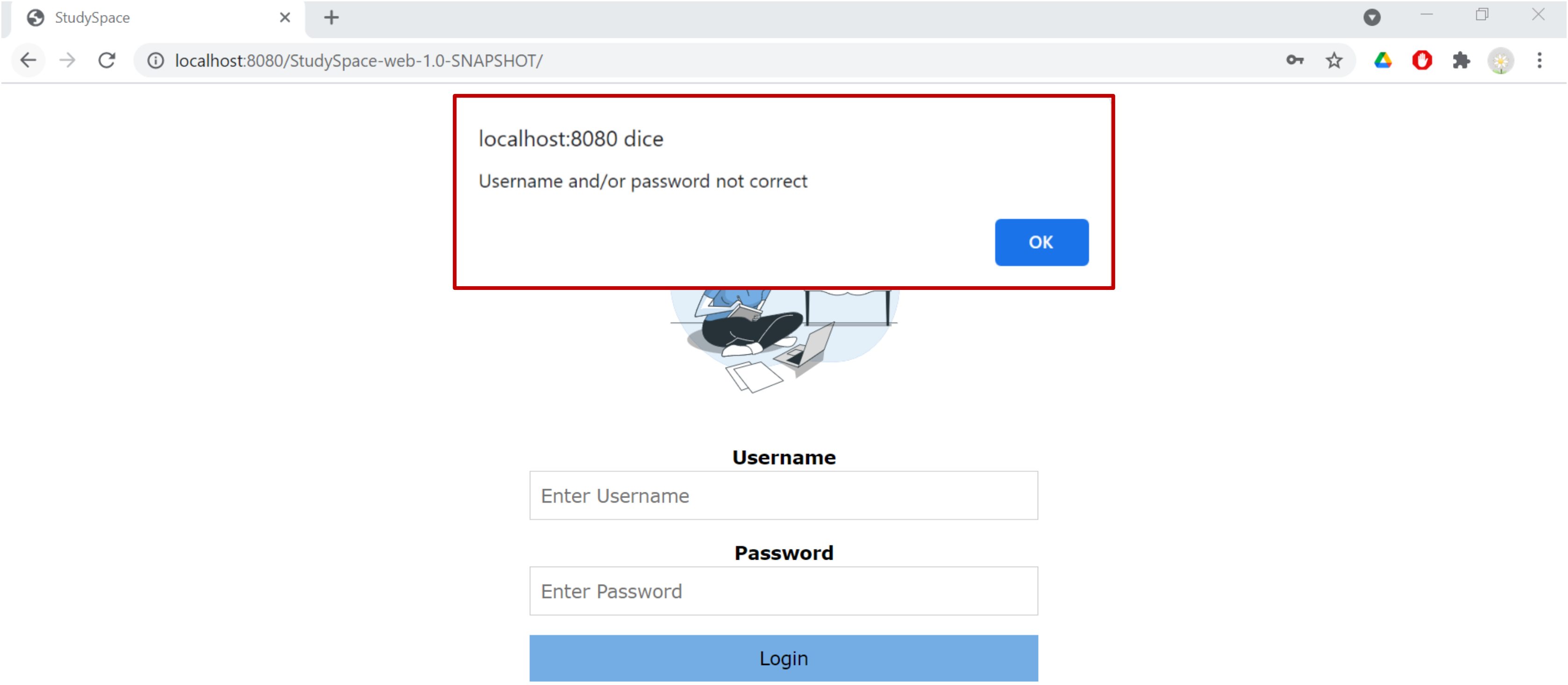
User

- Login/Logout
- Book seat
- Delete reservation

Admin

- Login/Logout
- Modify room capacity
- Modify room availability

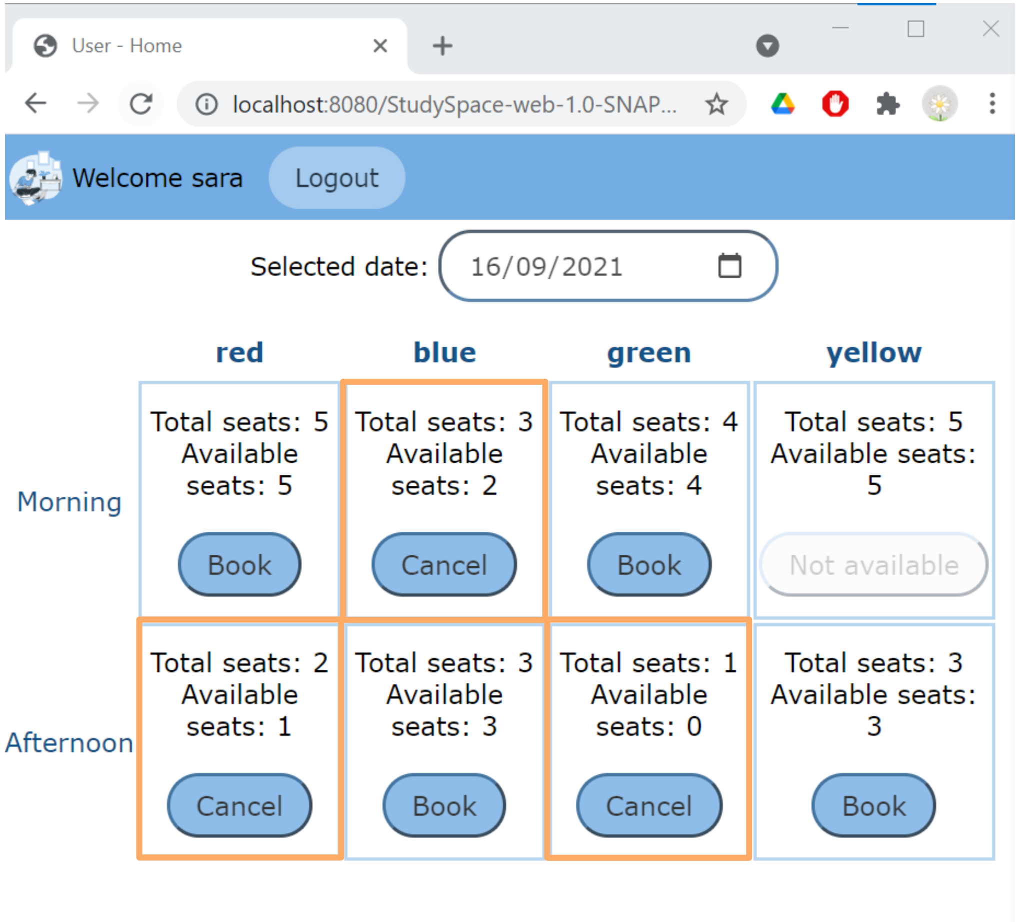
USE CASE: LOGIN



localhost:8080 dice

Username and/or password not correct

USER HOME



User - Home

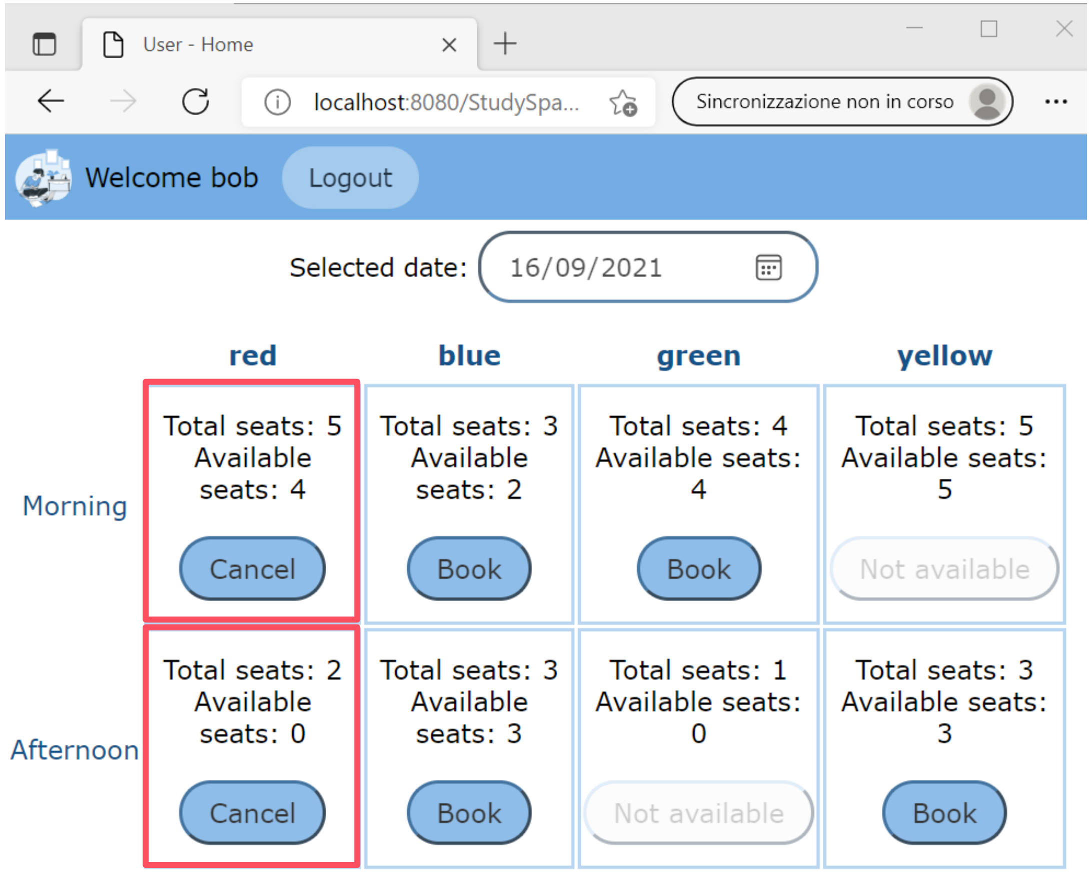
localhost:8080/StudySpace-web-1.0-SNAP...

Welcome sara

Logout

Selected date: 16/09/2021

	red	blue	green	yellow
Morning	Total seats: 5 Available seats: 5 Book	Total seats: 3 Available seats: 2 Cancel	Total seats: 4 Available seats: 4 Book	Total seats: 5 Available seats: 5 Not available
Afternoon	Total seats: 2 Available seats: 1 Cancel	Total seats: 3 Available seats: 3 Book	Total seats: 1 Available seats: 0 Cancel	Total seats: 3 Available seats: 3 Book



User - Home

localhost:8080/StudySpa...

Sincronizzazione non in corso

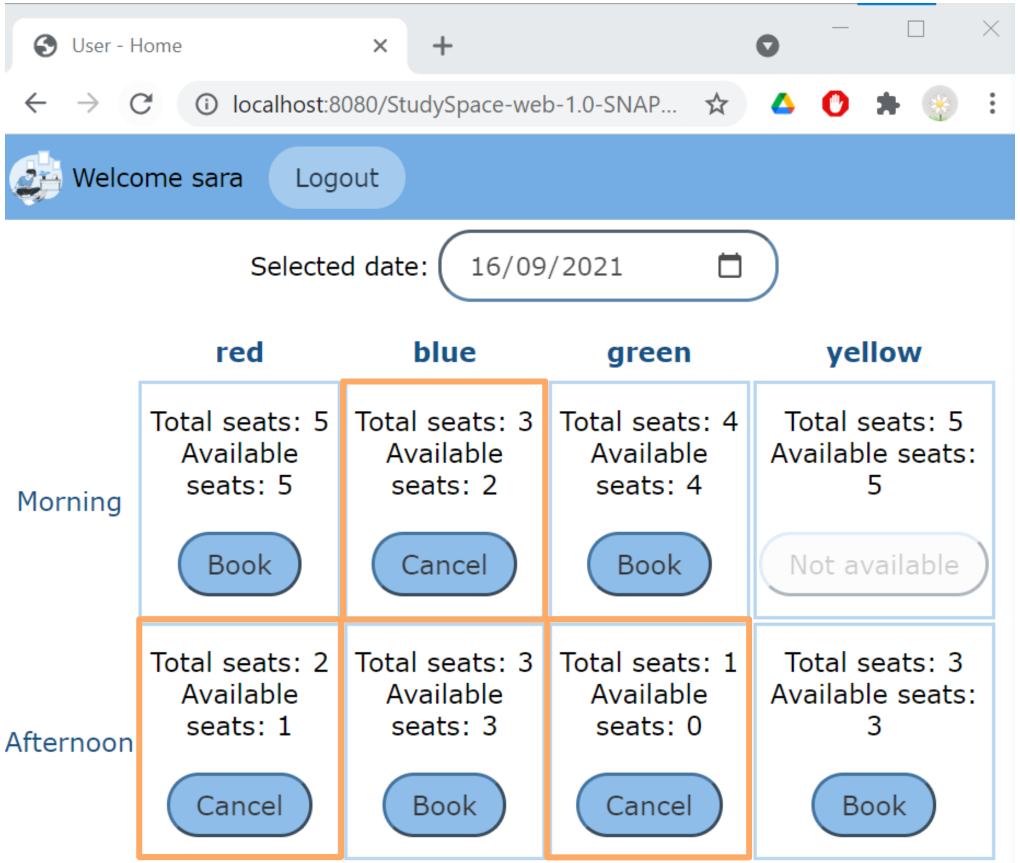
Welcome bob

Logout

Selected date: 16/09/2021

	red	blue	green	yellow
Morning	Total seats: 5 Available seats: 4 Cancel	Total seats: 3 Available seats: 2 Book	Total seats: 4 Available seats: 4 Book	Total seats: 5 Available seats: 5 Not available
Afternoon	Total seats: 2 Available seats: 0 Cancel	Total seats: 3 Available seats: 3 Book	Total seats: 1 Available seats: 0 Not available	Total seats: 3 Available seats: 3 Book

USE CASE: BOOK A SEAT



User - Home

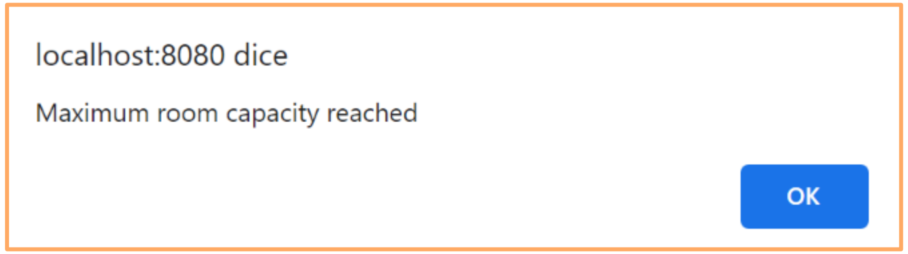
localhost:8080/StudySpace-web-1.0-SNAP...

Welcome sara Logout

Selected date: 16/09/2021

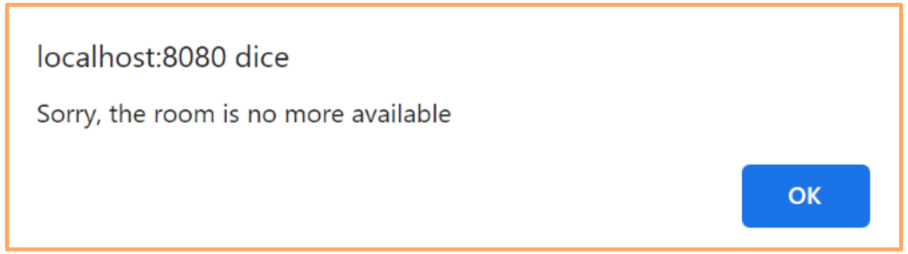
	red	blue	green	yellow
Morning	Total seats: 5 Available seats: 5 Book	Total seats: 3 Available seats: 2 Cancel	Total seats: 4 Available seats: 4 Book	Total seats: 5 Available seats: 5 Not available
Afternoon	Total seats: 2 Available seats: 1 Cancel	Total seats: 3 Available seats: 3 Book	Total seats: 1 Available seats: 0 Cancel	Total seats: 3 Available seats: 3 Book

Possible alerts:



localhost:8080 dice

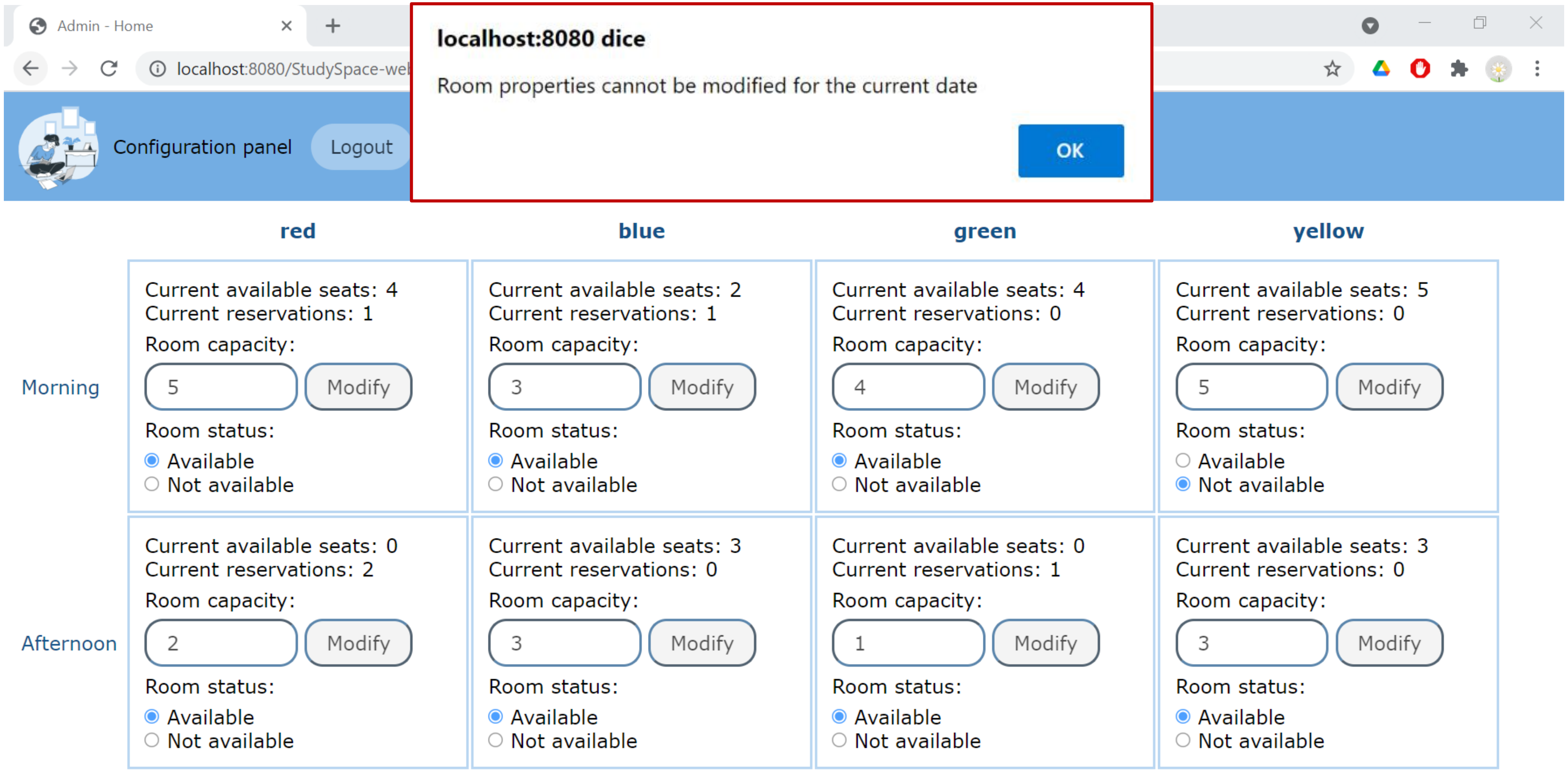
Maximum room capacity reached



localhost:8080 dice

Sorry, the room is no more available

ADMIN HOME



Admin - Home

localhost:8080/StudySpace-wel

Configuration panel

Logout

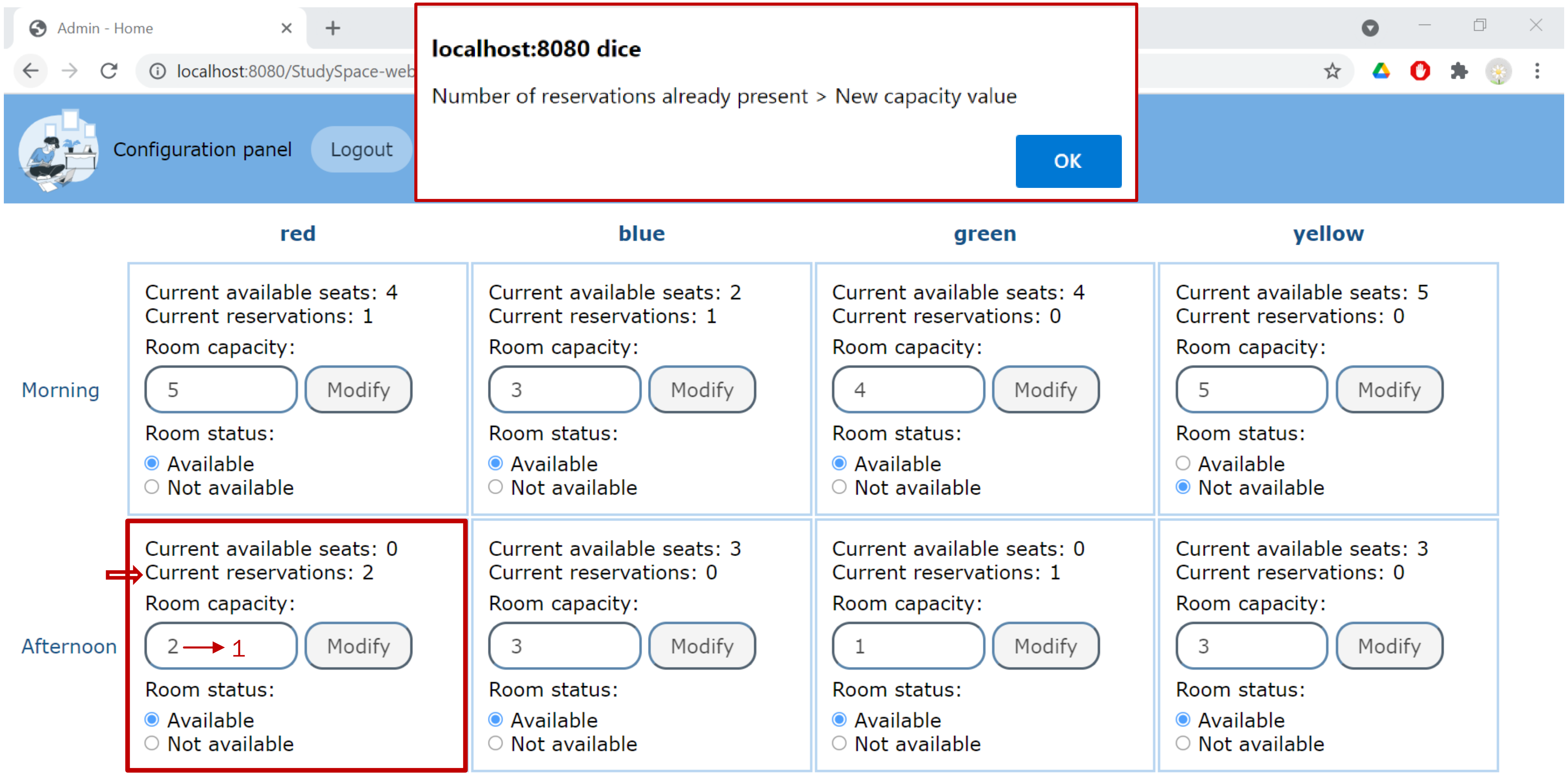
localhost:8080 dice

Room properties cannot be modified for the current date

OK

	red	blue	green	yellow
Morning	Current available seats: 4 Current reservations: 1 Room capacity: 5 Modify Room status: ● Available ○ Not available	Current available seats: 2 Current reservations: 1 Room capacity: 3 Modify Room status: ● Available ○ Not available	Current available seats: 4 Current reservations: 0 Room capacity: 4 Modify Room status: ● Available ○ Not available	Current available seats: 5 Current reservations: 0 Room capacity: 5 Modify Room status: ○ Available ● Not available
Afternoon	Current available seats: 0 Current reservations: 2 Room capacity: 2 Modify Room status: ● Available ○ Not available	Current available seats: 3 Current reservations: 0 Room capacity: 3 Modify Room status: ● Available ○ Not available	Current available seats: 0 Current reservations: 1 Room capacity: 1 Modify Room status: ● Available ○ Not available	Current available seats: 3 Current reservations: 0 Room capacity: 3 Modify Room status: ● Available ○ Not available

USE CASE: MODIFY ROOM CAPACITY



localhost:8080 dice

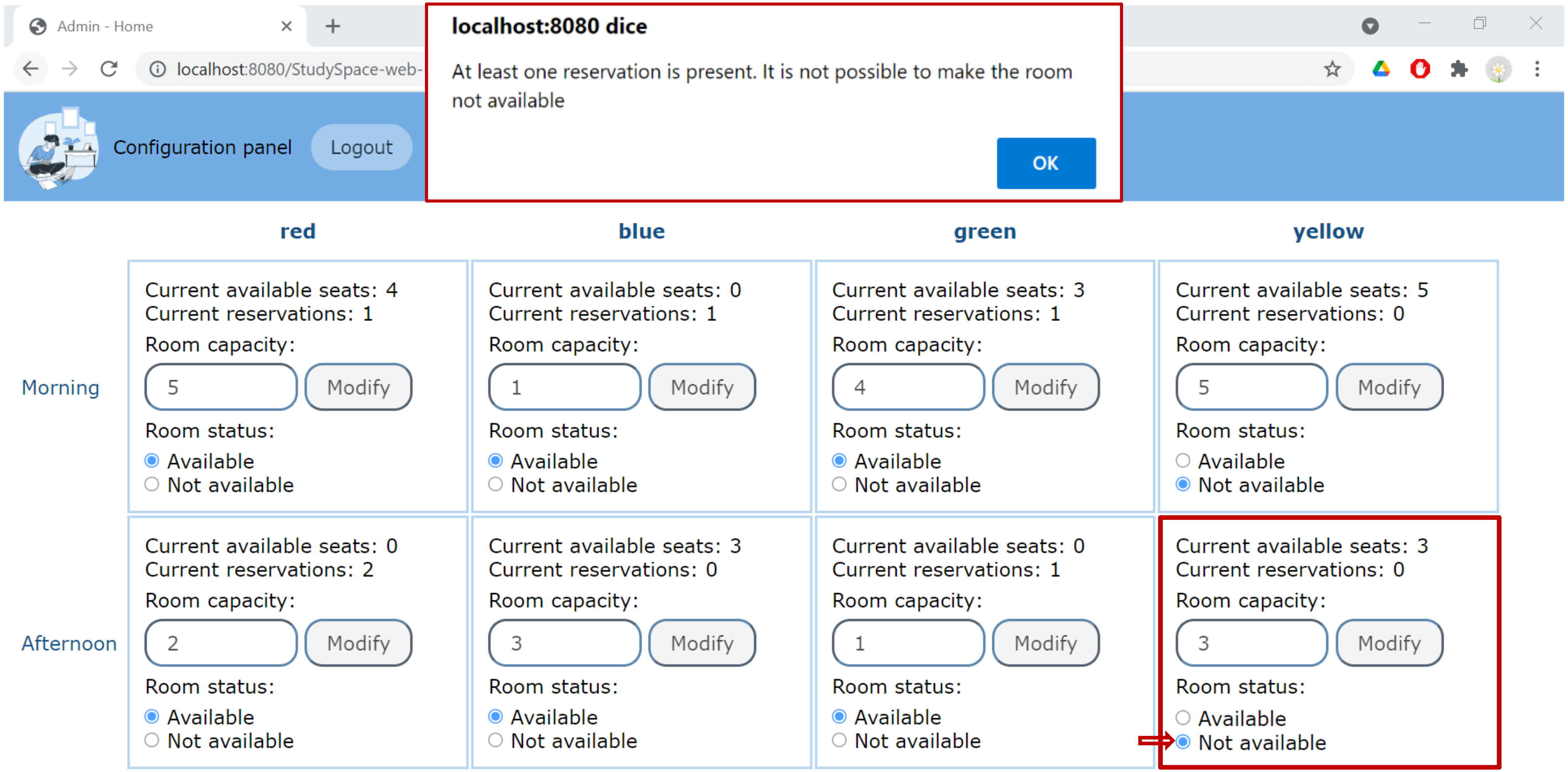
Number of reservations already present > New capacity value

OK

	red	blue	green	yellow
Morning	Current available seats: 4 Current reservations: 1 Room capacity: 5 Modify Room status: ● Available ○ Not available	Current available seats: 2 Current reservations: 1 Room capacity: 3 Modify Room status: ● Available ○ Not available	Current available seats: 4 Current reservations: 0 Room capacity: 4 Modify Room status: ● Available ○ Not available	Current available seats: 5 Current reservations: 0 Room capacity: 5 Modify Room status: ○ Available ● Not available
Afternoon	Current available seats: 0 ⇒ Current reservations: 2 Room capacity: 2 → 1 Modify Room status: ● Available ○ Not available	Current available seats: 3 Current reservations: 0 Room capacity: 3 Modify Room status: ● Available ○ Not available	Current available seats: 0 Current reservations: 1 Room capacity: 1 Modify Room status: ● Available ○ Not available	Current available seats: 3 Current reservations: 0 Room capacity: 3 Modify Room status: ● Available ○ Not available

USE CASE: MODIFY ROOM AVAILABILITY

Possible alert:



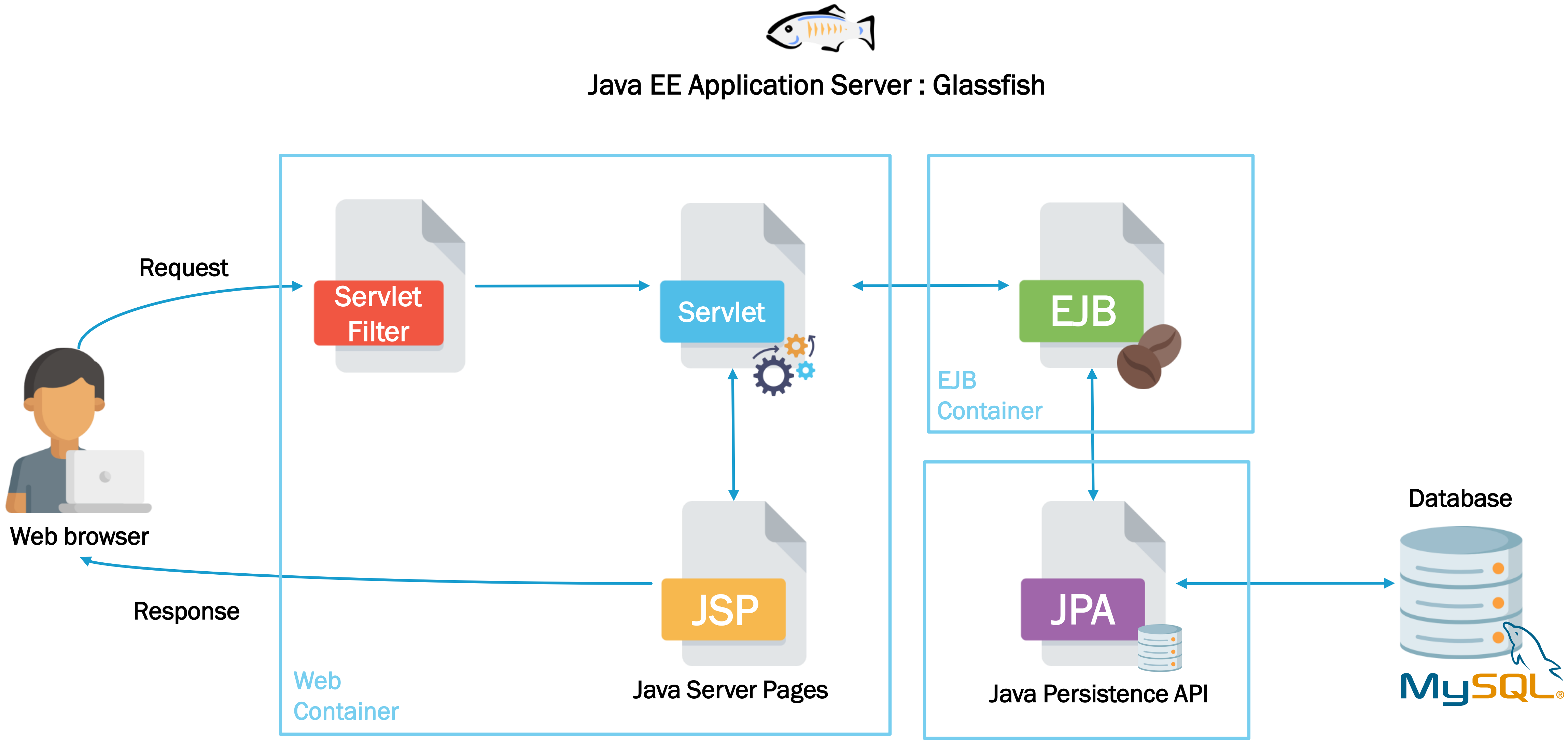
localhost:8080 dice

At least one reservation is present. It is not possible to make the room not available

	red	blue	green	yellow
Morning	Current available seats: 4 Current reservations: 1 Room capacity: 5 Modify Room status: ● Available ○ Not available	Current available seats: 0 Current reservations: 1 Room capacity: 1 Modify Room status: ● Available ○ Not available	Current available seats: 3 Current reservations: 1 Room capacity: 4 Modify Room status: ● Available ○ Not available	Current available seats: 5 Current reservations: 0 Room capacity: 5 Modify Room status: ○ Available ● Not available
Afternoon	Current available seats: 0 Current reservations: 2 Room capacity: 2 Modify Room status: ● Available ○ Not available	Current available seats: 3 Current reservations: 0 Room capacity: 3 Modify Room status: ● Available ○ Not available	Current available seats: 0 Current reservations: 1 Room capacity: 1 Modify Room status: ● Available ○ Not available	Current available seats: 3 Current reservations: 0 Room capacity: 3 Modify Room status: ○ Available ● Not available

IMPLEMENTATION

SYSTEM ARCHITECTURE



SERVLETS

Servlet classes

- controllerLogin: checks the correctness of username and password
- controllerLogout: invalidates the user session
- controllerCalendar: shows the room data for the selected date
- controllerReservation: allows the user to create or cancel a reservation
- controllerAdmin: allows the admin to modify the properties of the rooms

Servlet Filter classes

- adminAuthenticationFilter: prevents access to the admin page if you are not an admin
- userAuthenticationFilter: prevents access to the user's personal page if the user is not correctly logged into

ENTERPRISE JAVA BEANS

- The application uses two different categories of Session Beans:

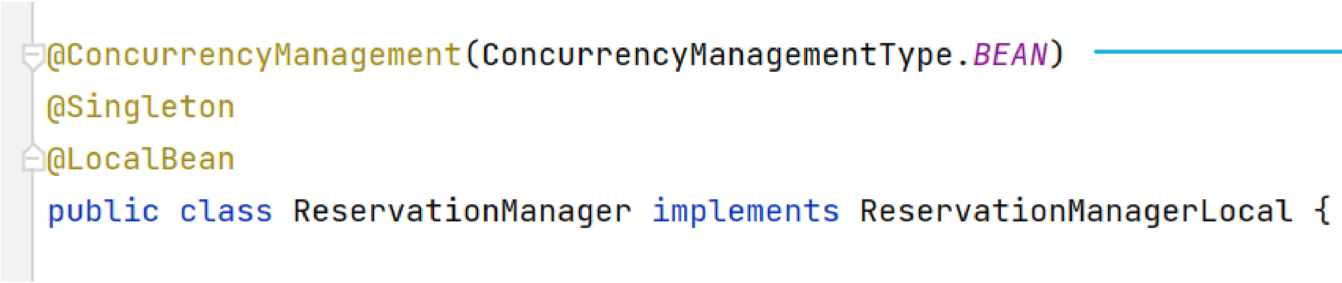
Stateless Session Beans

- Admin
- User
- Room
- Reservation

Singleton Session Bean

- ReservationManager ← How can we manage the concurrency?

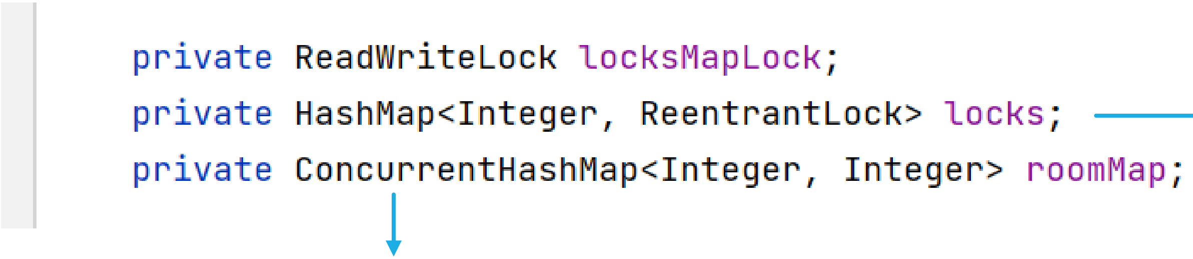
CONCURRENCY MANAGEMENT TYPE



```
@ConcurrencyManagement(ConcurrencyManagementType.BEAN)
@Singleton
@LocalBean
public class ReservationManager implements ReservationManagerLocal {
```

“Bean developer is responsible for managing concurrent access to the bean instance.”

CONCURRENT DATA STRUCTURES

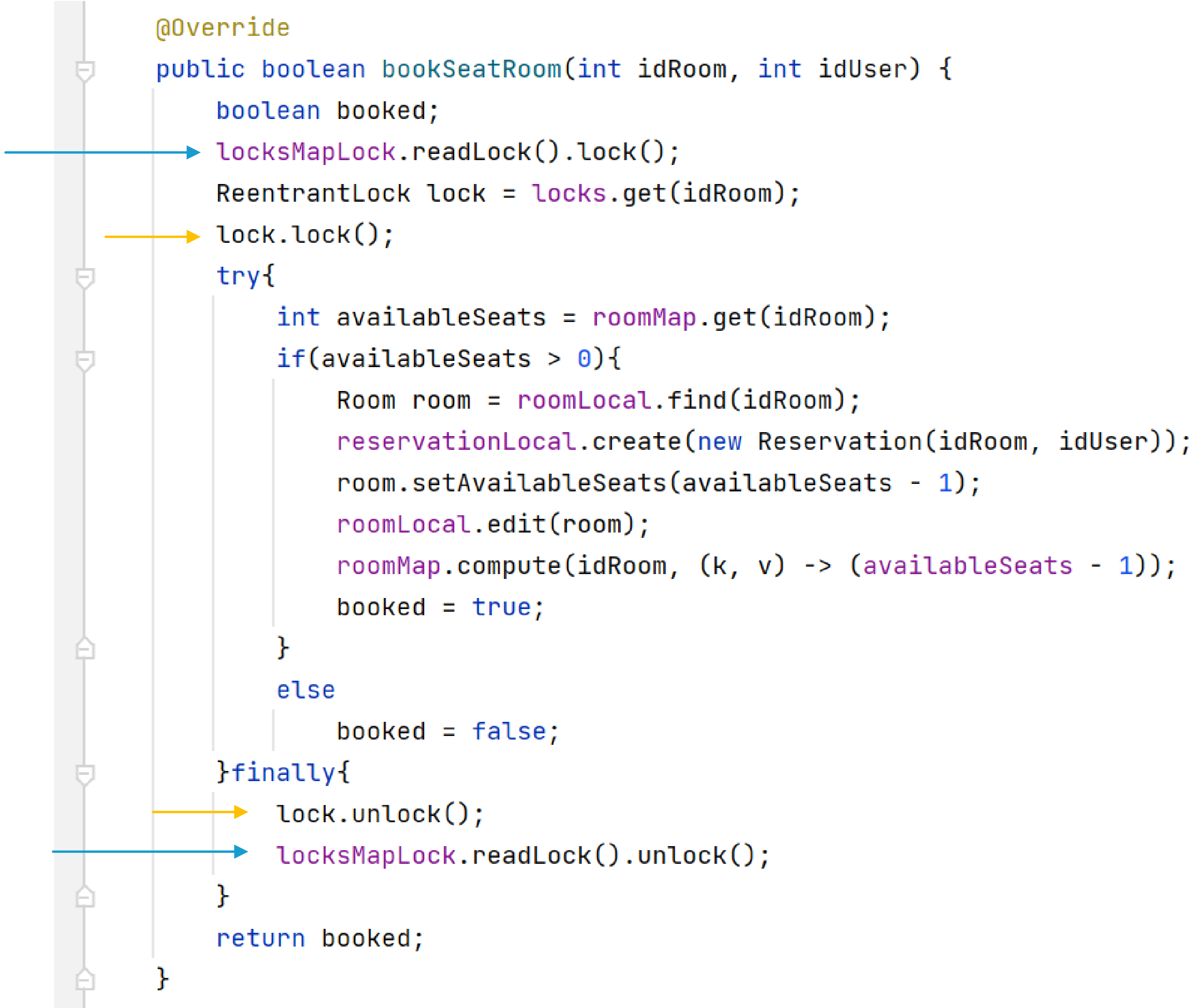


```
    private ReadWriteLock locksMapLock;
    private HashMap<Integer, ReentrantLock> locks;
    private ConcurrentHashMap<Integer, Integer> roomMap;
```

The HashMap contains a ReentrantLock for each room.
The **key** is the **idRoom**.

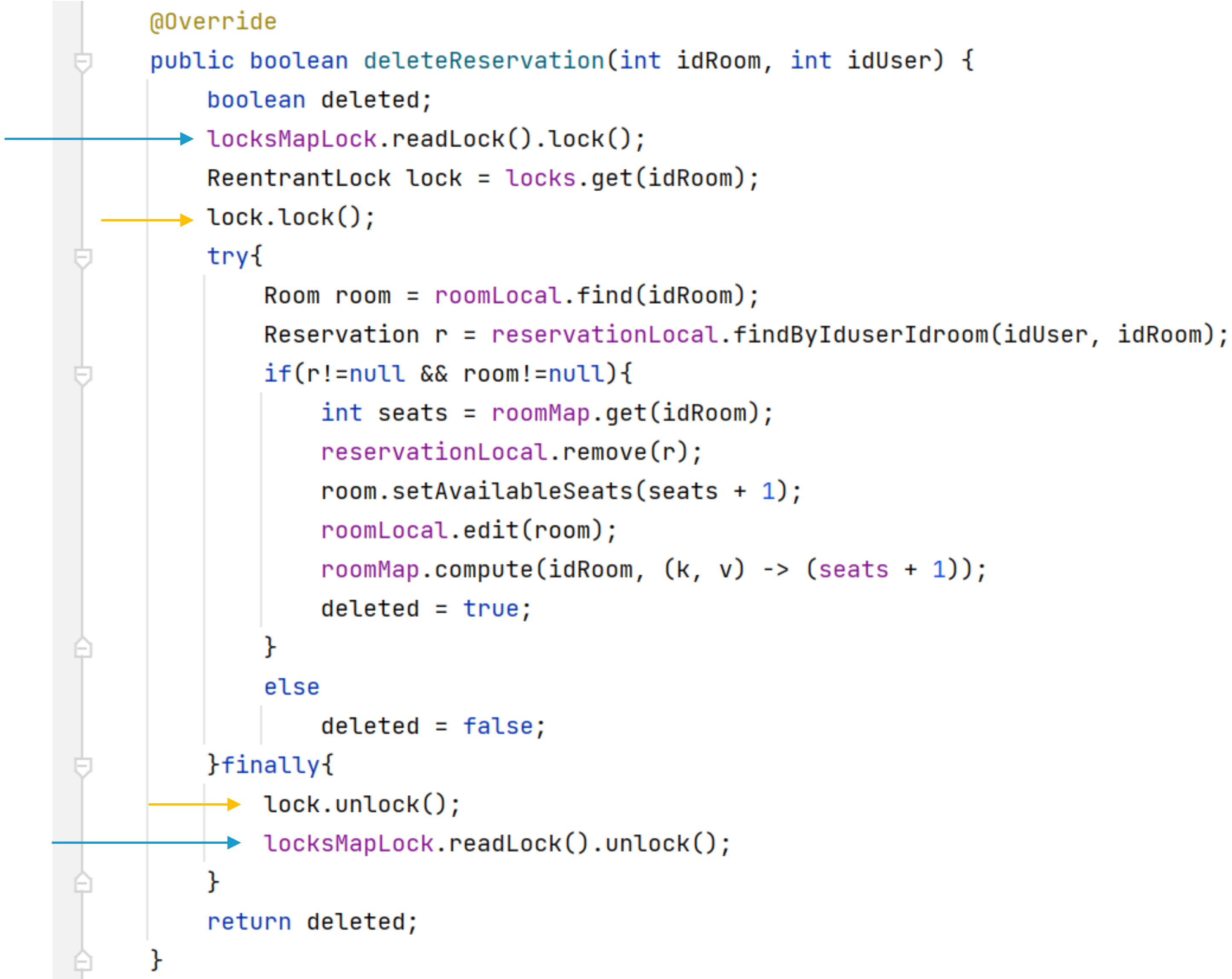
The ConcurrentHashMap contains for each room (**key: idRoom**) the number of current available seats (**value: availableSeats**).

CONCURRENT METHODS: BOOK A SEAT



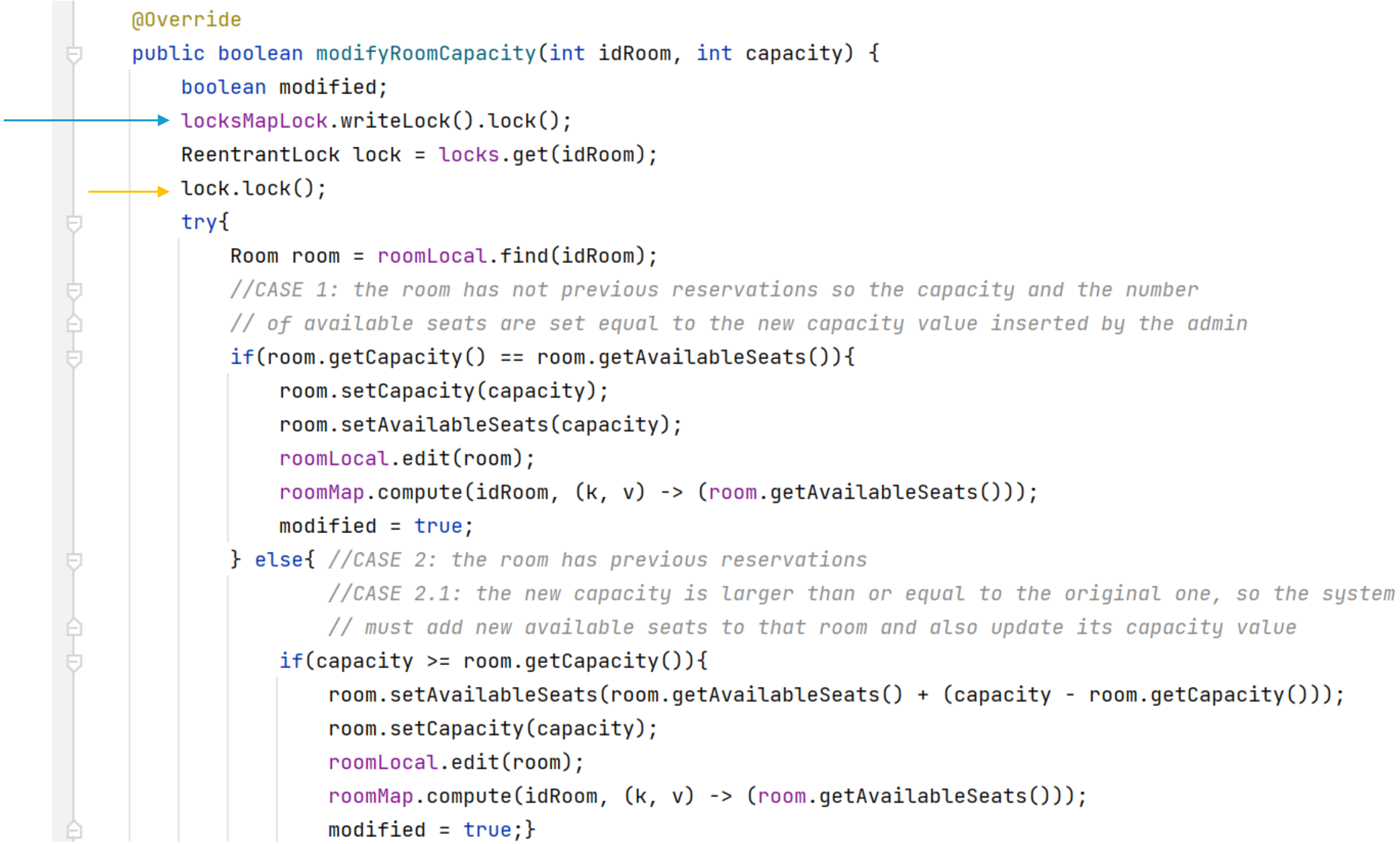
```java
@Override
public boolean bookSeatRoom(int idRoom, int idUser) {
    boolean booked;
    locksMapLock.readLock().lock();
    ReentrantLock lock = locks.get(idRoom);
    lock.lock();
    try{
        int availableSeats = roomMap.get(idRoom);
        if(availableSeats > 0){
            Room room = roomLocal.find(idRoom);
            reservationLocal.create(new Reservation(idRoom, idUser));
            room.setAvailableSeats(availableSeats - 1);
            roomLocal.edit(room);
            roomMap.compute(idRoom, (k, v) -> (availableSeats - 1));
            booked = true;
        }
        else
            booked = false;
    }finally{
        lock.unlock();
        locksMapLock.readLock().unlock();
    }
    return booked;
}
```

CONCURRENT METHODS: DELETE RESERVATION



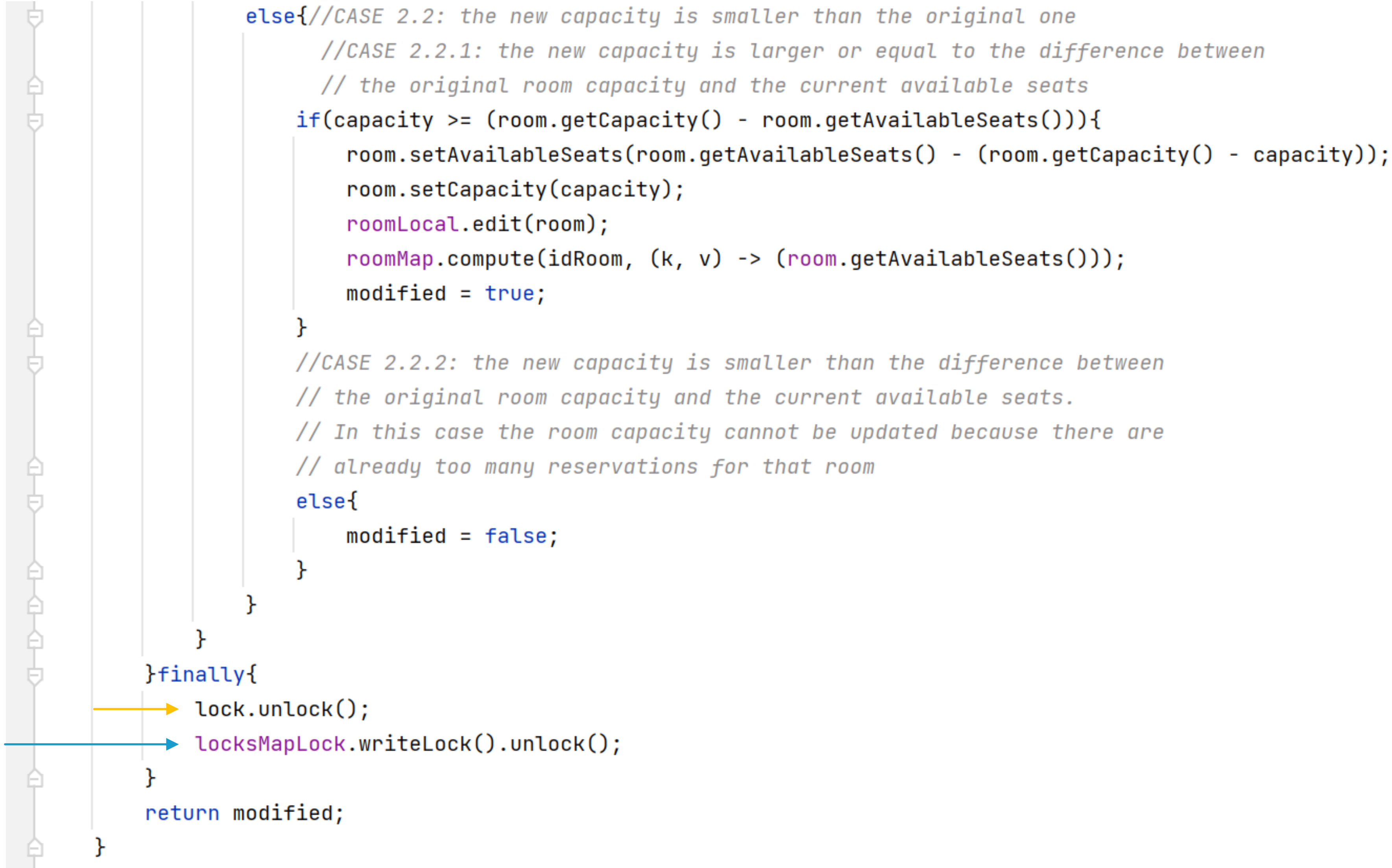
```java
@Override
public boolean deleteReservation(int idRoom, int idUser) {
    boolean deleted;
    locksMapLock.readLock().lock();
    ReentrantLock lock = locks.get(idRoom);
    lock.lock();
    try{
        Room room = roomLocal.find(idRoom);
        Reservation r = reservationLocal.findByIduserIdroom(idUser, idRoom);
        if(r!=null && room!=null){
            int seats = roomMap.get(idRoom);
            reservationLocal.remove(r);
            room.setAvailableSeats(seats + 1);
            roomLocal.edit(room);
            roomMap.compute(idRoom, (k, v) -> (seats + 1));
            deleted = true;
        }
        else
            deleted = false;
    }finally{
        lock.unlock();
        locksMapLock.readLock().unlock();
    }
    return deleted;
}
```

CONCURRENT METHODS: MODIFY ROOM CAPACITY (1/2)



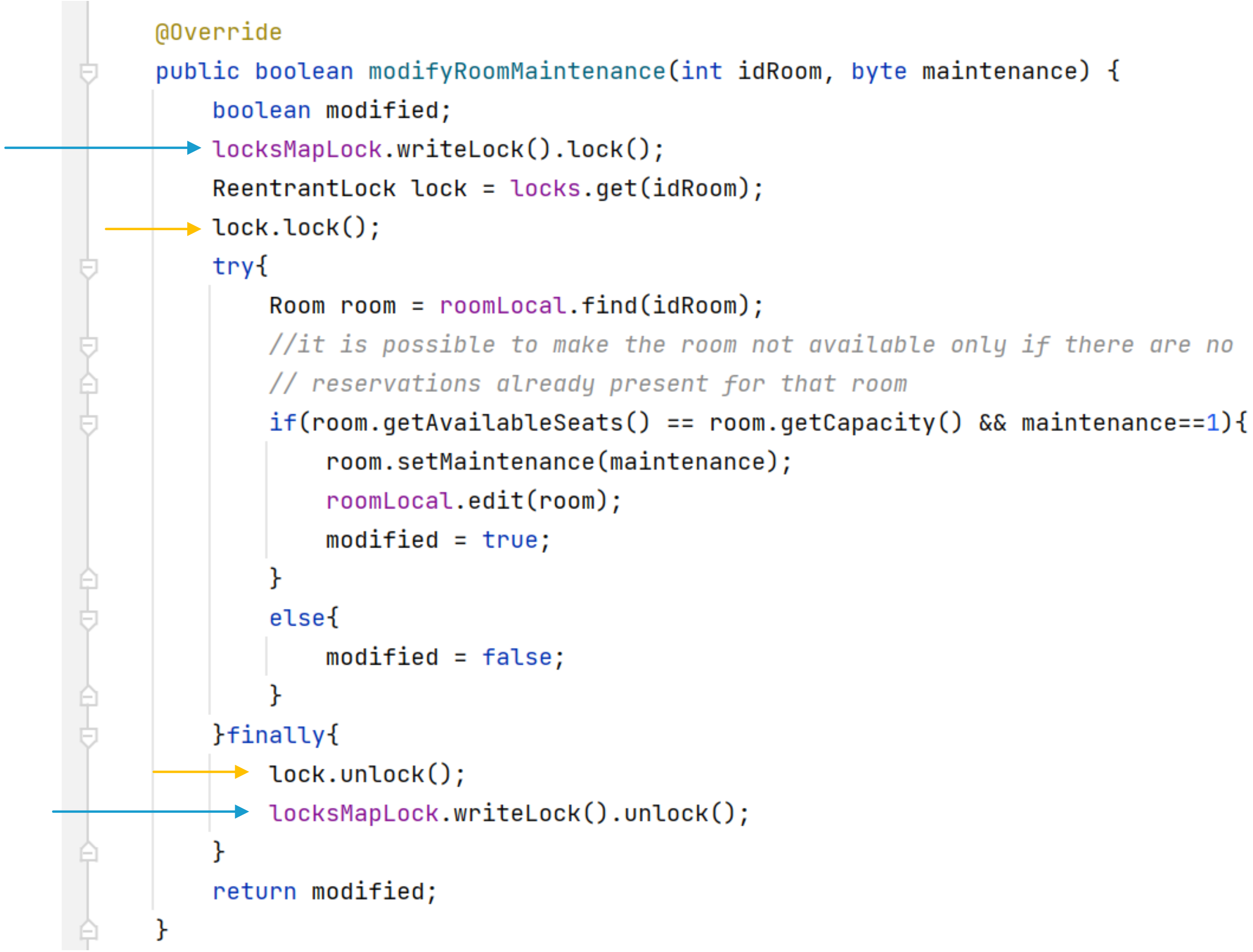
```java
@Override
public boolean modifyRoomCapacity(int idRoom, int capacity) {
    boolean modified;
    locksMapLock.writeLock().lock();
    ReentrantLock lock = locks.get(idRoom);
    lock.lock();
    try{
        Room room = roomLocal.find(idRoom);
        //CASE 1: the room has not previous reservations so the capacity and the number
        // of available seats are set equal to the new capacity value inserted by the admin
        if(room.getCapacity() == room.getAvailableSeats()){
            room.setCapacity(capacity);
            room.setAvailableSeats(capacity);
            roomLocal.edit(room);
            roomMap.compute(idRoom, (k, v) -> (room.getAvailableSeats()));
            modified = true;
        } else{ //CASE 2: the room has previous reservations
                //CASE 2.1: the new capacity is larger than or equal to the original one, so the system
                // must add new available seats to that room and also update its capacity value
            if(capacity >= room.getCapacity()){
                room.setAvailableSeats(room.getAvailableSeats() + (capacity - room.getCapacity()));
                room.setCapacity(capacity);
                roomLocal.edit(room);
                roomMap.compute(idRoom, (k, v) -> (room.getAvailableSeats()));
                modified = true;}
```

CONCURRENT METHODS: MODIFY ROOM CAPACITY (2/2)



```
            else{//CASE 2.2: the new capacity is smaller than the original one
                  //CASE 2.2.1: the new capacity is larger or equal to the difference between
                  // the original room capacity and the current available seats
                if(capacity >= (room.getCapacity() - room.getAvailableSeats())){
                    room.setAvailableSeats(room.getAvailableSeats() - (room.getCapacity() - capacity));
                    room.setCapacity(capacity);
                    roomLocal.edit(room);
                    roomMap.compute(idRoom, (k, v) -> (room.getAvailableSeats()));
                    modified = true;
                }
                //CASE 2.2.2: the new capacity is smaller than the difference between
                // the original room capacity and the current available seats.
                // In this case the room capacity cannot be updated because there are
                // already too many reservations for that room
                else{
                    modified = false;
                }
            }
        }
    }finally{
        lock.unlock();
        locksMapLock.writeLock().unlock();
    }
    return modified;
}
```

CONCURRENT METHODS: MODIFY ROOM MAINTENANCE



```java
@Override
public boolean modifyRoomMaintenance(int idRoom, byte maintenance) {
    boolean modified;
    locksMapLock.writeLock().lock();
    ReentrantLock lock = locks.get(idRoom);
    lock.lock();
    try{
        Room room = roomLocal.find(idRoom);
        //it is possible to make the room not available only if there are no
        // reservations already present for that room
        if(room.getAvailableSeats() == room.getCapacity() && maintenance==1){
            room.setMaintenance(maintenance);
            roomLocal.edit(room);
            modified = true;
        }
        else{
            modified = false;
        }
    }finally{
        lock.unlock();
        locksMapLock.writeLock().unlock();
    }
    return modified;
}
```

THANK YOU FOR THE ATTENTION