COMP 6231- WINTER 2023

Distributed System Design

ASSIGNMENT #1

Movie Ticket Booking System

Submitted by: **Ganga Singh**

Instructor: **Dr. Rajagopalan Jayakumar**

Overview

The task is to make a software named Movie Ticket Booking System to which customers can book movie shows along with some other functionalities and similarly an Admin at theater can perform certain actions to Add, Delete movie shows on assigned theaters.

Actions performed by Admins:

1. Add movie shows.
2. Delete movie shows.
3. List movies show availability.
4. Book ticket. (Perform an action on behalf of the user)
5. Cancel ticket. (Perform an action on behalf of the user)
6. Show bookings. (Perform an action on behalf of the user)

Actions performed by Customers:

1. Book ticket.
2. Cancel ticket.
3. Show bookings.

Servers/Theaters	Movies
Atwater	Avatar, Avengers, Titanic
Verdun	Avatar, Avengers, Titanic
Outermont	Avatar, Avengers, Titanic

User ID	Movie ID
ATW/VER/OUT (Server Prefix)	ATW/VER/OUT (Server Prefix)
A/M (Admin/Customer)	M/A/E(Morning/A.Noon/Evening)
1234 (Four digit number)	190822 (ddMMyy)

Architecture and Implementation

1. Client and Server are communicating over Remote method Invocation technique.

2. Different ports associated with servers are listed below.

Server	Port (UDP)
Atwater	2501
Verdun	2502
Outermont	2503

3. Dependency Injection method is used to achieve inversion of control (**IOC**) in this software by moving objects binding from compile time to run time using service locator pattern.

4. Code pattern is structured in a way to reduce redundant code and facilitate debugging easily.

5. Logger is used to create logs separately for each customer/Admin and for each server for every operation.

6. Customer logs are located at: src/Log/Client

7. Theaters/Servers logs are located at: src/Log/Server

8. Concurrent HashMap is used to maximize concurrency and to store data for both Movies and Customers bookings

9. The Most Important part in the assignment is to obtain maximum concurrency and communication with other servers without hanging up in a UDP calls loop.

10. The challenging functionality was to implement the (*Delete movie shows*) in such a way that it should design the logic for finding the next available show as this requires sorting at each individual movie theater to find the next possible movie slot available.

11. Data Structure for Customer booking information used as below:

```
private Map<String, Map<String, MovieState>> customerBooking;
```

Map<*CustomerID*,Map<*MovieID*,*MovieState*>> customerBooking;

The *MovieState* class has the following information.

```
private Map<String,Integer> movieNamesAndTickets;
```

Map<*MovieName*,*NoOfTicketsBooked*> movieNamesAndTickets

Diagram Representation:

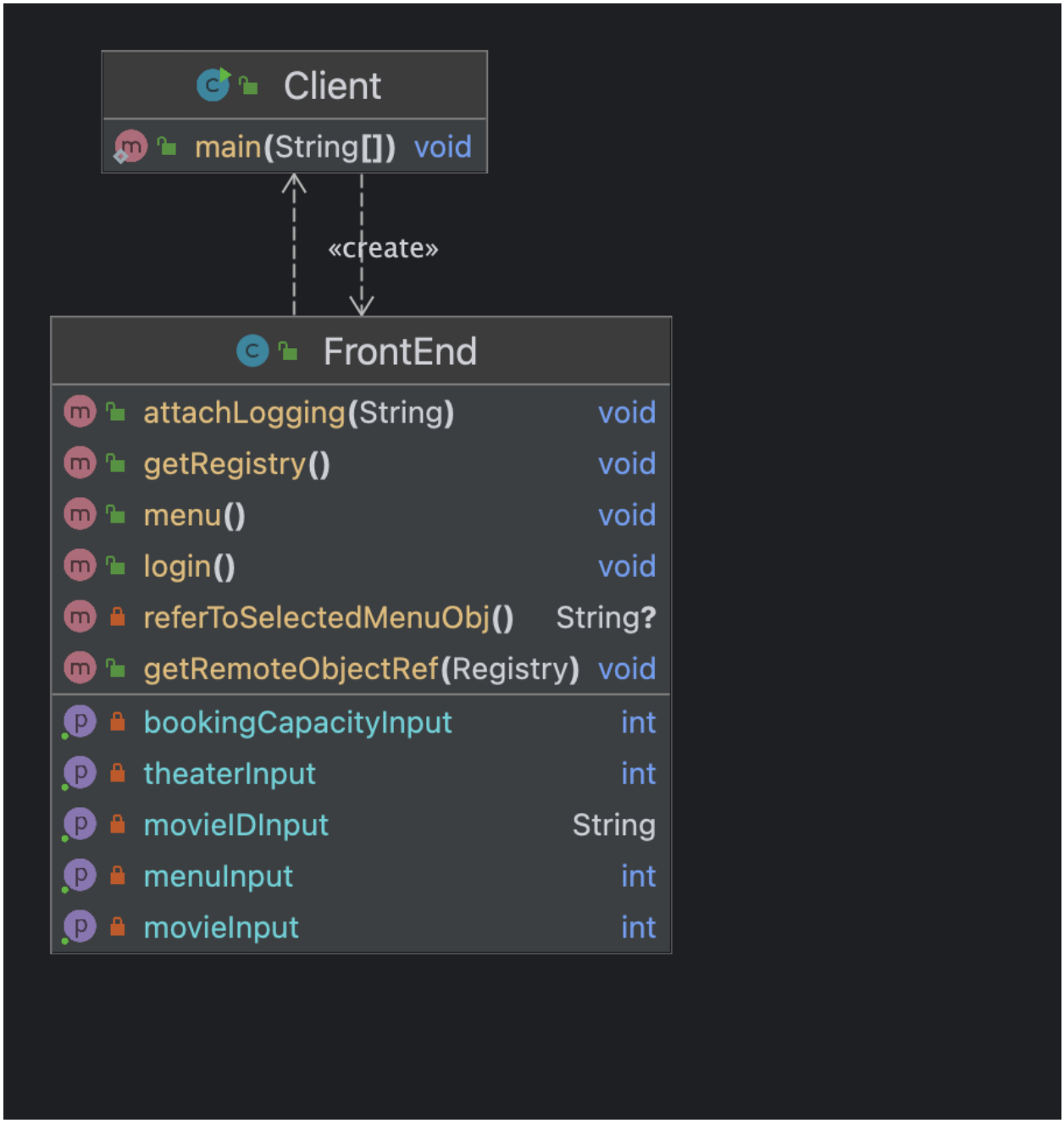
HashMap of customerBooking

CustomerID	*MovieID*	*MovieState*

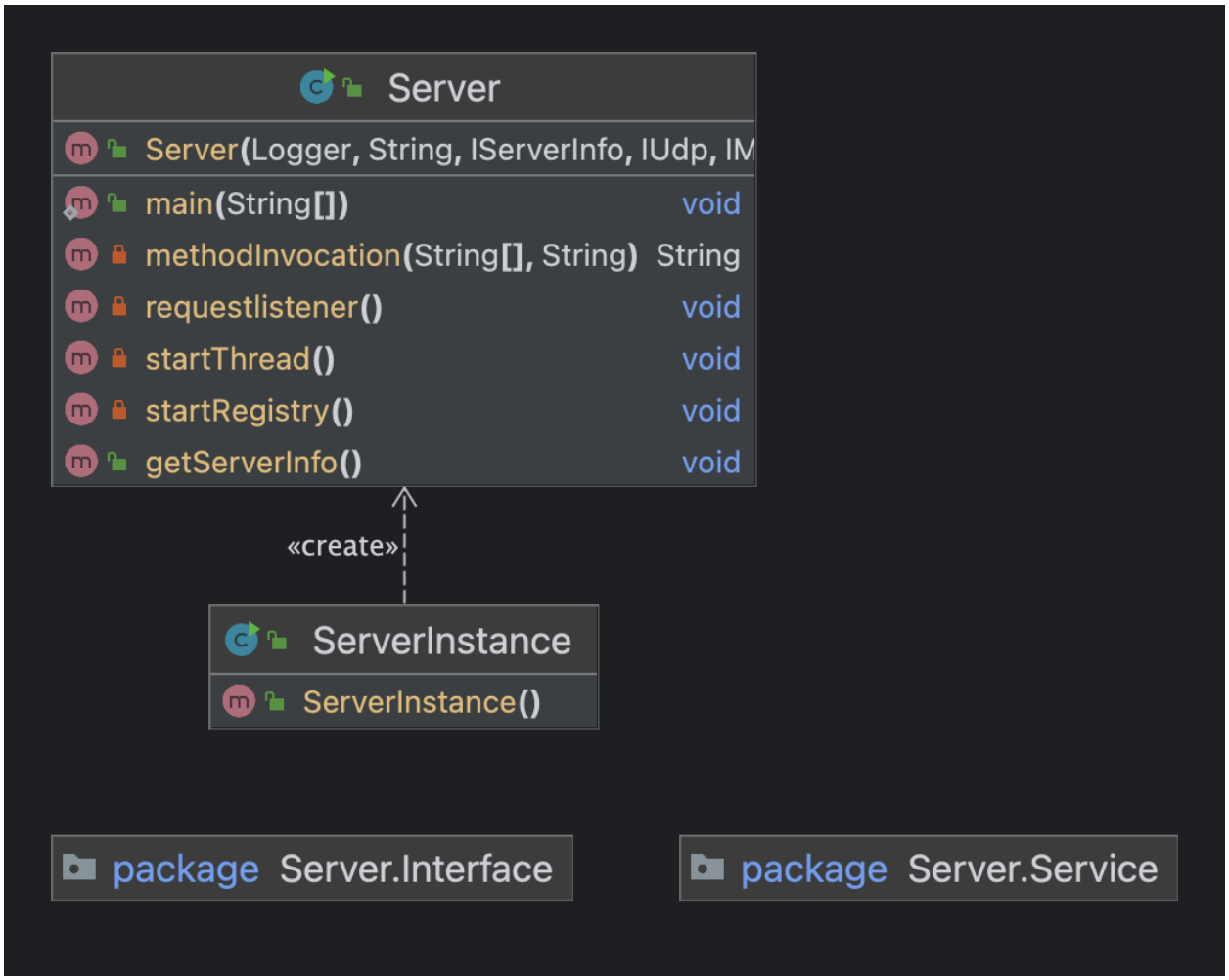
MovieState class has a field of type HashMap.
movieNamesAndTickets:

MovieName	*NoOfTicketsBooked*

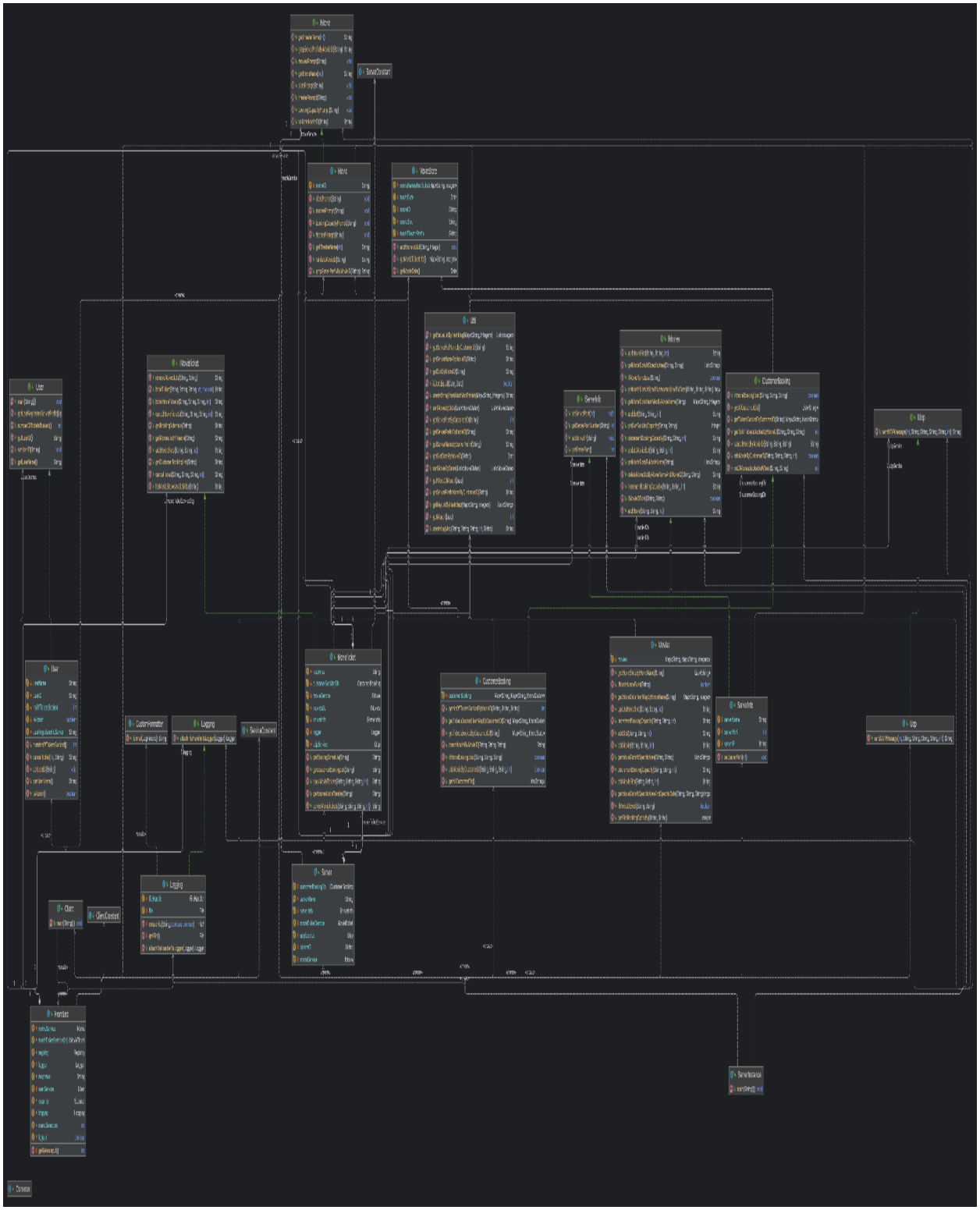
UML Diagram of Client



UML Diagram of Server



Full Project UML Diagram



Test Scenarios

S.No	Type	Input/Method	Cases
1.	Login	ID	1. Admin 2. Customer
2.	Logout	Option	Logout
3.	Admin	Add Slot	1. Valid Admin ID. 2. If movieID exist update else create new entry 3. Cannot add slots for unauthorized theaters. 4. Cannot add slot for one week later. 5. Update server response logs on server and client
4.	Admin	Remove Slot	1. Existing movieID should be deleted. 2. No client bookings for a movie then perform delete operation. 3. If a client booked the same slot which has to be deleted then assign the next available slot. 4. Previously occurring movie shows should not be deleted. 5. Other theater movies not to be deleted. 6. Update server response logs on

			server and client
5.	Admin	List Movies shows	1. Admin call to associated server only. 2. UDP calls to other servers to fetch movies. 3. Update server response logs on server and client
6.	Admin/ Customer	Book Movie	1. Ask Customer ID if(Admin operation) 2. Allowed on a registered server. 3. Allowed on other theaters. 4. Cannot book for more than 3 movies in a week. 5. Infinite bookings on a registered server. 6. Invalid Movie ID not allowed. 7. Update server response logs on server and client
7.	Admin/ Customer	Cancel bookings	1. Ask Customer ID if(Admin operation) 2. Only booking customerID can delete the movie. 3. Update server response logs on server and client

8.	Admin/ Customer	Get booking Schedule	1. Ask Customer ID if(Admin operation) 2. UDP calls to other servers and fetches all bookings from all servers by CustomerID. 3. Update server response logs on server and client