Distributed Systems - Assignment 3

Distributed Player Status System (DPSS) using Web-Services

Assignment -3
Distributed Player Status System (DPSS) – Web-Services

Submitted to:
Dr. Mohamed Taleb
Email: mohamed.taleb@concordia.ca
Office: S-EV 3233

Submitted by:
Vaibhav Malhotra
ID: 40079373

Development tools

All code is written in IntelliJ IDE, Java JDK version 8.

This project implements **Option 1** (Build the end point files using the wsgen command before publishing the service. Import the wsdl files using the wsimport command.)

Build and Run

IDE (IntelliJ)

- Open the project in folder DPSS_WebServices
- Setup the SDK

Generating artifacts and endpoints using wsgen

- Compile Server Implementations files in Servers folder
- Open terminal at DPSS_WebServices
- Run the command below three commands to generate wsdl files
    - `wsdl -cp . src.main.java.Servers.America.AmericanServerImpl -wsdl`
    - `wsdl -cp . src.main.java.Servers.Asia.AsianServerImpl -wsdl`
    - `wsdl -cp . src.main.java.Servers.Europe.EuropeanServerImpl -wsdl`

- Go to ‘http://localhost:8080/server?wsdl to verify.

Import wsdl files using wsimport

- Open terminal at DPSS_WebServices
- Run the command below three commands to import wsdl files
    - `wsimport -keep -d . -p main.java.Client` https://localhost:8080/server?wsdl
    - `wsimport -keep -d . -p main.java.Client` https://localhost:8081/server?wsdl
    - `wsimport -keep -d . -p main.java.Client` https://localhost:8082/server?wsdl

Running the project

- Run: AmericaGameServer, EuropeGameServer, AsiaGameServer.
- Run: PlayerClient (to launch a player window)
- Run: AdminClient (to launch a Admin window)
- To run multiple clients change the configuration to “Allow parallel run”.

Command Line

- Move to *DPSS* directoy: **cd DPSS_WebServices**.
- Create a new folder named *dist* in the current folder : **mkdir dist**
- Compile the code (outputting into dist folder): **javac -d dist src/**/*.java**
- Move to *dist* folder: **cd dist**

Generating artifacts and endpoints using wsgen

- Compile Server Implementations files in Servers folder
- Open terminal at DPSS_WebServices
- Run the command below three commands to generate wsdl files
    - `wsdl -cp . src.main.java.Servers.America.AmericanServerImpl -wsdl`
    - `wsdl -cp . src.main.java.Servers.Asia.AsianServerImpl -wsdl`
    - `wsdl -cp . src.main.java.Servers.Europe.EuropeanServerImpl -wsdl`

- Go to ‘http://**local**host:8080/server?wsdl to verify.

Import wsdl files using wsimport

- Open terminal at DPSS_WebServices
- Run the command below three commands to import wsdl files
    - `wsimport -keep -d . -p main.java.Client` https://localhost:8080/server?wsdl
    - `wsimport -keep -d . -p main.java.Client` https://localhost:8081/server?wsdl
    - `wsimport -keep -d . -p main.java.Client` https://localhost:8082/server?wsdl

Start Servers:

```
start java Server.America.AmericaServer
start java Server.Asia.AsiaServer
start java Server.Europe.EuropeanServer
```

- Run the clients (will have to open in different terminals):

```
java PlayerClient
java AdminClient
```

Ports

- AmericanServer – 8080
- EuropeanServer - 8081
- AsianServer – 8082

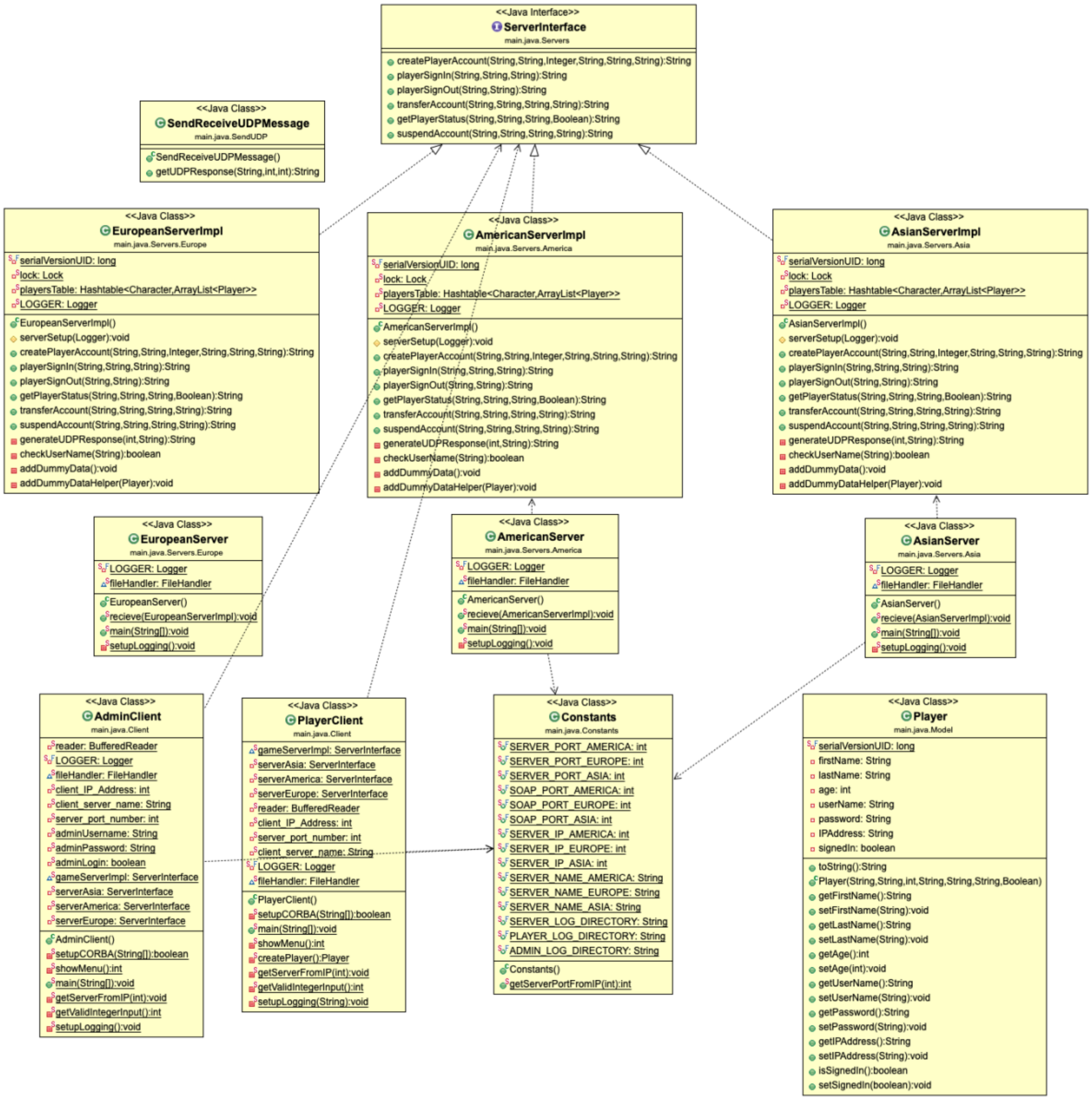
The ports can be changed in Constants file.

Architecture

Three different servers (America, Europe and Asia) are started, which then start their own UDP sockets for communication. Each of these servers bind their implementation object with unique names under Naming Context.

Intercommunication between the servers is handled through UDP requests. These will be discussed in detail later.

Class diagram:


<<Java Interface>>
ServerInterface
main.java.Servers
createPlayerAccount(String,String,Integer,String,String,String):String
playerSignIn(String,String,String):String
playerSignOut(String,String):String
transferAccount(String,String,String,String):String
getPlayerStatus(String,String,String,Boolean):String
suspendAccount(String,String,String,String):String
<<Java Class>>
SendReceiveUDPMessage
main.java.SendUDP
SendReceiveUDPMessage()
getUDPResponse(String,int,int):String
<<Java Class>>
EuropeanServerImpl
main.java.Servers.Europe
serialVersionUID: long
lock: Lock
playersTable: Hashtable<Character,ArrayList<Player>>
LOGGER: Logger
EuropeanServerImpl()
serverSetup(Logger):void
createPlayerAccount(String,String,Integer,String,String,String):String
playerSignIn(String,String,String):String
playerSignOut(String,String):String
getPlayerStatus(String,String,String,Boolean):String
transferAccount(String,String,String,String):String
suspendAccount(String,String,String,String):String
generateUDPResponse(int,String):String
checkUserName(String):boolean
addDummyData():void
addDummyDataHelper(Player):void
<<Java Class>>
AmericanServerImpl
main.java.Servers.America
serialVersionUID: long
lock: Lock
playersTable: Hashtable<Character,ArrayList<Player>>
LOGGER: Logger
AmericanServerImpl()
serverSetup(Logger):void
createPlayerAccount(String,String,Integer,String,String,String):String
playerSignIn(String,String,String):String
playerSignOut(String,String):String
getPlayerStatus(String,String,String,Boolean):String
transferAccount(String,String,String,String):String
suspendAccount(String,String,String,String):String
generateUDPResponse(int,String):String
checkUserName(String):boolean
addDummyData():void
addDummyDataHelper(Player):void
<<Java Class>>
AsianServerImpl
main.java.Servers.Asia
serialVersionUID: long
lock: Lock
playersTable: Hashtable<Character,ArrayList<Player>>
LOGGER: Logger
AsianServerImpl()
serverSetup(Logger):void
createPlayerAccount(String,String,Integer,String,String,String):String
playerSignIn(String,String,String):String
playerSignOut(String,String):String
getPlayerStatus(String,String,String,Boolean):String
transferAccount(String,String,String,String):String
suspendAccount(String,String,String,String):String
generateUDPResponse(int,String):String
checkUserName(String):boolean
addDummyData():void
addDummyDataHelper(Player):void
<<Java Class>>
EuropeanServer
main.java.Servers.Europe
LOGGER: Logger
fileHandler: FileHandler
EuropeanServer()
recieve(EuropeanServerImpl):void
main(String[]):void
setupLogging():void
<<Java Class>>
AmericanServer
main.java.Servers.America
LOGGER: Logger
fileHandler: FileHandler
AmericanServer()
recieve(AmericanServerImpl):void
main(String[]):void
setupLogging():void
<<Java Class>>
AsianServer
main.java.Servers.Asia
LOGGER: Logger
fileHandler: FileHandler
AsianServer()
recieve(AsianServerImpl):void
main(String[]):void
setupLogging():void
<<Java Class>>
AdminClient
main.java.Client
reader: BufferedReader
LOGGER: Logger
fileHandler: FileHandler
client_IP_Address: int
client_server_name: String
server_port_number: int
adminUsername: String
adminPassword: String
adminLogin: boolean
gameServerImpl: ServerInterface
serverAsia: ServerInterface
serverAmerica: ServerInterface
serverEurope: ServerInterface
AdminClient()
setupCORBA(String[]):boolean
showMenu():int
main(String[]):void
getServerFromIP(int):void
getValidIntegerInput():int
setupLogging():void
<<Java Class>>
PlayerClient
main.java.Client
gameServerImpl: ServerInterface
serverAsia: ServerInterface
serverAmerica: ServerInterface
serverEurope: ServerInterface
reader: BufferedReader
client_IP_Address: int
server_port_number: int
client_server_name: String
LOGGER: Logger
fileHandler: FileHandler
PlayerClient()
setupCORBA(String[]):boolean
main(String[]):void
showMenu():int
createPlayer():Player
getServerFromIP(int):void
getValidIntegerInput():int
setupLogging(String):void
<<Java Class>>
Constants
main.java.Constants
SERVER_PORT_AMERICA: int
SERVER_PORT_EUROPE: int
SERVER_PORT_ASIA: int
SOAP_PORT_AMERICA: int
SOAP_PORT_EUROPE: int
SOAP_PORT_ASIA: int
SERVER_IP_AMERICA: int
SERVER_IP_EUROPE: int
SERVER_IP_ASIA: int
SERVER_NAME_AMERICA: String
SERVER_NAME_EUROPE: String
SERVER_NAME_ASIA: String
SERVER_LOG_DIRECTORY: String
PLAYER_LOG_DIRECTORY: String
ADMIN_LOG_DIRECTORY: String
Constants()
getServerPortFromIP(int):int
<<Java Class>>
Player
main.java.Model
serialVersionUID: long
firstName: String
lastName: String
age: int
userName: String
password: String
IPAddress: String
signedIn: boolean
toString():String
Player(String,String,int,String,String,String,Boolean)
getFirstName():String
setFirstName(String):void
getLastName():String
setLastName(String):void
getAge():int
setAge(int):void
getUserName():String
setUserName(String):void
getPassword():String
setPassword(String):void
getIPAddress():String
setIPAddress(String):void
isSignedIn():boolean
setSignedIn(boolean):void

Techniques Used

1. JAX-WS (Java API for XML Web Services)

- Used JAX-WS to develop a web services for DPSS. The communication is happens using SOAP which is a XML-based protocol.
- Used WSDL (Web Service Description Language) to generate required artifacts to deploy the service.

2. UDP

For below communication between server UDP is used:

- **transferAccount()** - When a user request to transfer account to another server, a UDP request with user information is generated which instructs the other server to add player to it's database and responds with the status. If successful the player is removed from old server.
- **getPlayerStatus()** – When admin requests this method on a server, that server sends UDP request to other two servers to get the player info.
- **createPlayer()** – When a user tries to create a new player on a server, that server sends a UDP request to other servers to check if Username already exists.

2. Multi-threading

- All servers run on their individual thread
- All UDP requests are sent on a new thread
- All client requests are sent on a new thread

3. HashTables - Data Structure

- Player data on server are stored in a Hashtables. Hashtables are thread-safe and promote concurrency.

4. Locks

- Lock (ReentrantLock) is used for proper synchronization to allow multiple users to perform operations for the same or different accounts at the same time.

Vaibhav Malhotra – 40079373

Most Challenging part

Running the wsgen command:

While running **wsgen** command to generate wsdl files for the servers I encountered the below issue.

“Missing SEI”
This issue was because I was not providing the right path for my java implementation of the server file. The below link helped me resolve the issue:

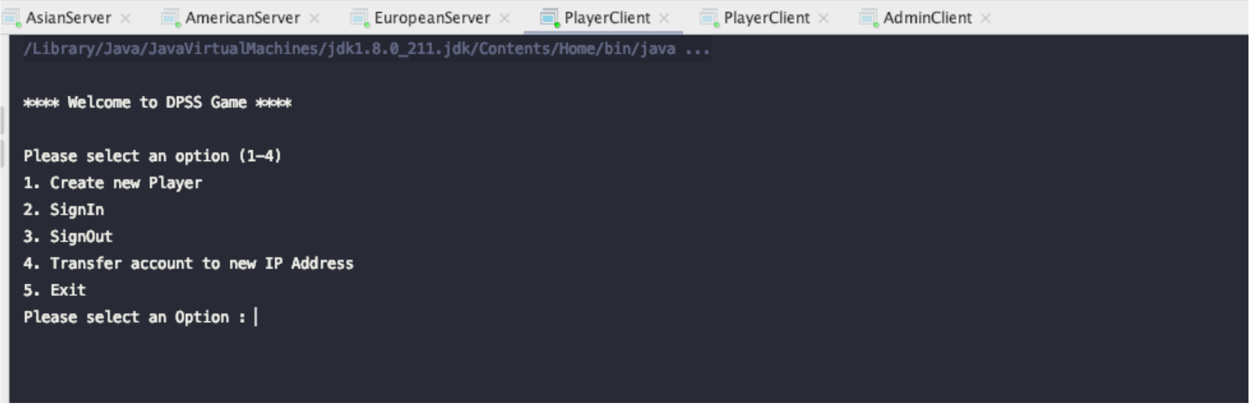
https://stackoverflow.com/questions/13180271/missing-sei-when-generating-artifacts-with-wsgen

Vaibhav Malhotra – 40079373

Testing

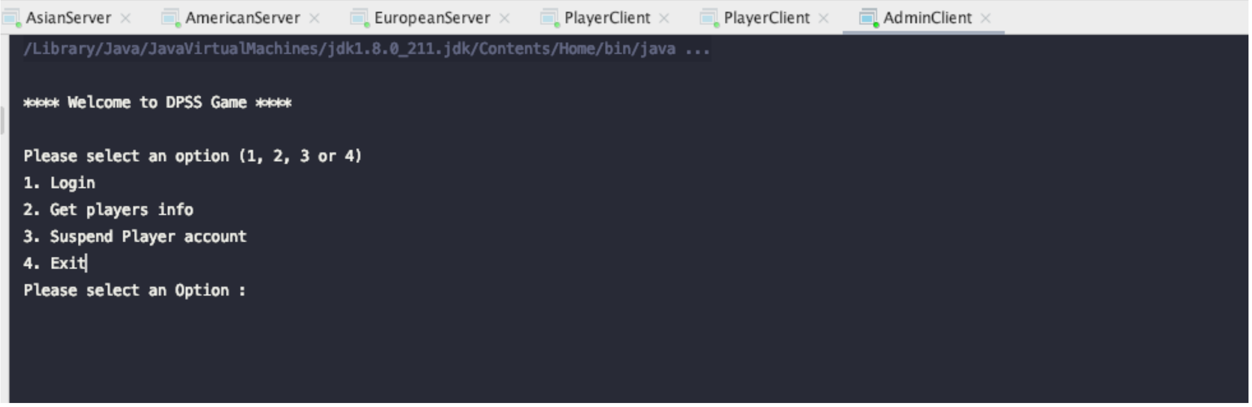
Intro screens:

Player Screen:



```
/Library/Java/JavaVirtualMachines/jdk1.8.0_211.jdk/Contents/Home/bin/java ...

**** Welcome to DPSS Game ****

Please select an option (1-4)
1. Create new Player
2. SignIn
3. SignOut
4. Transfer account to new IP Address
5. Exit
Please select an Option : |
```

Admin Screen:



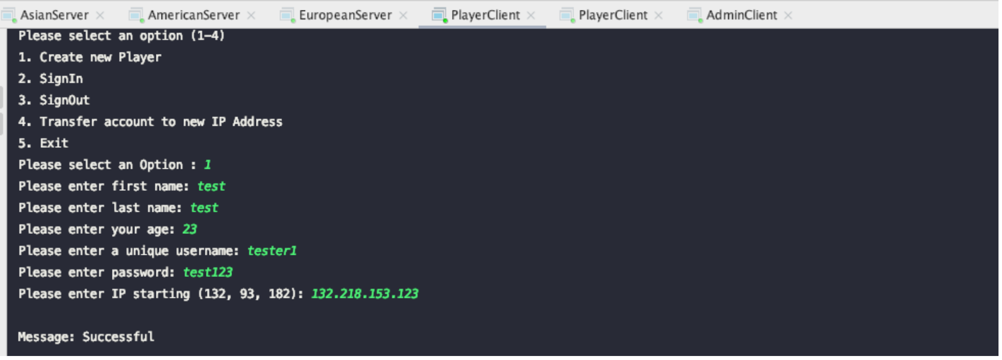
```
/Library/Java/JavaVirtualMachines/jdk1.8.0_211.jdk/Contents/Home/bin/java ...

**** Welcome to DPSS Game ****

Please select an option (1, 2, 3 or 4)
1. Login
2. Get players info
3. Suspend Player account
4. Exit
Please select an Option :
```

Player testing:

In this section we will be testing Player Client functionalities.

Test Number	Scenario	Reason
1	Create player	Major functionality
2	Login player	Major functionality
3	Login player who is already logged in	Major functionality, Concurrent access
4	Logout player	Major functionality
5	Transfer Player to another server	Major functionality
6	Create a user (Already on another server)	Major functionality

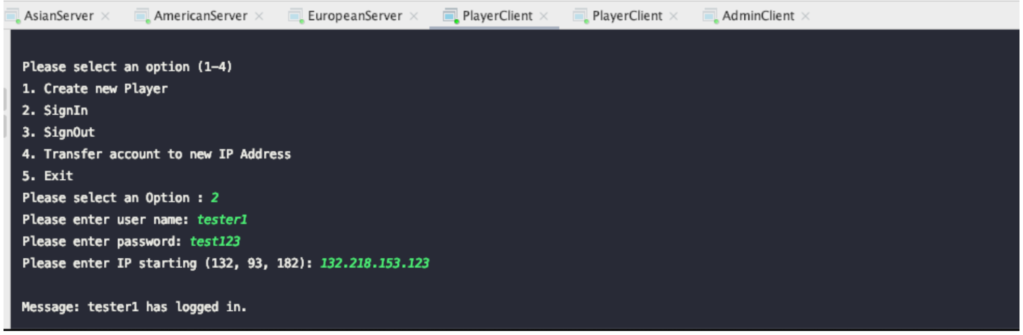
1. **Create player - *createPlayerAccount* (*FirstName, LastName, Age, Username, Password, IP Address*)**



```
Please select an option (1-4)
1. Create new Player
2. SignIn
3. SignOut
4. Transfer account to new IP Address
5. Exit
Please select an Option : 1
Please enter first name: test
Please enter last name: test
Please enter your age: 23
Please enter a unique username: tester1
Please enter password: test123
Please enter IP starting (132, 93, 182): 132.218.153.123

Message: Successful
```

Input data is validated.
Player created successfully and message is displayed.

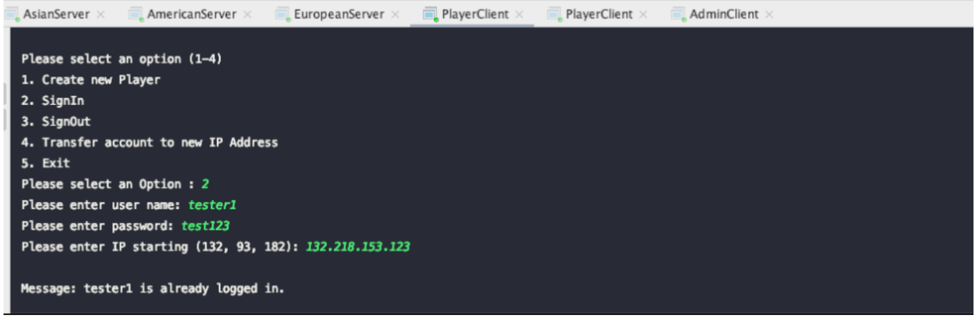
2. **Login player - *playerSignIn* (*Username, Password, IPAddress*)**



```
Please select an option (1-4)
1. Create new Player
2. SignIn
3. SignOut
4. Transfer account to new IP Address
5. Exit
Please select an Option : 2
Please enter user name: tester1
Please enter password: test123
Please enter IP starting (132, 93, 182): 132.218.153.123

Message: tester1 has logged in.
```

Player data is validated. Player logged in successfully and message is displayed.

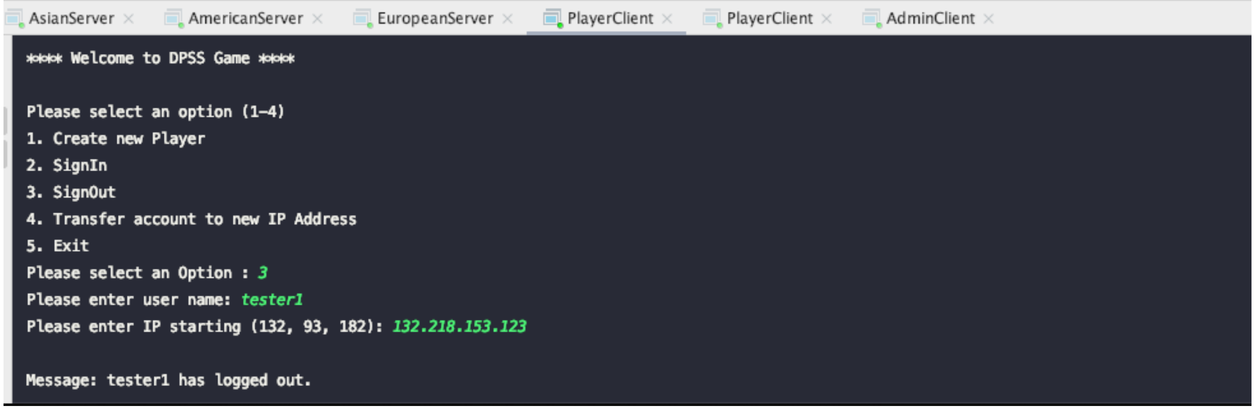
3. **Login player who is already logged in.**



```
Please select an option (1-4)
1. Create new Player
2. SignIn
3. SignOut
4. Transfer account to new IP Address
5. Exit
Please select an Option : 2
Please enter user name: tester1
Please enter password: test123
Please enter IP starting (132, 93, 182): 132.218.153.123

Message: tester1 is already logged in.
```

Player log in unsuccessfully because player is already logged in and message is displayed.

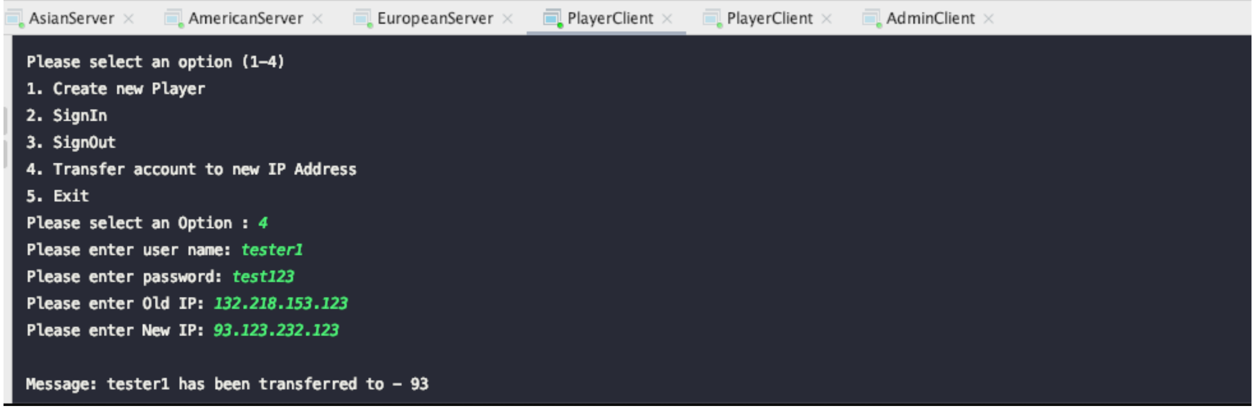
4. Logout player - *playerSignOut* (*Username, IPAddress*)



```
**** Welcome to DPSS Game ****

Please select an option (1-4)
1. Create new Player
2. SignIn
3. SignOut
4. Transfer account to new IP Address
5. Exit
Please select an Option : 3
Please enter user name: tester1
Please enter IP starting (132, 93, 182): 132.218.153.123

Message: tester1 has logged out.
```

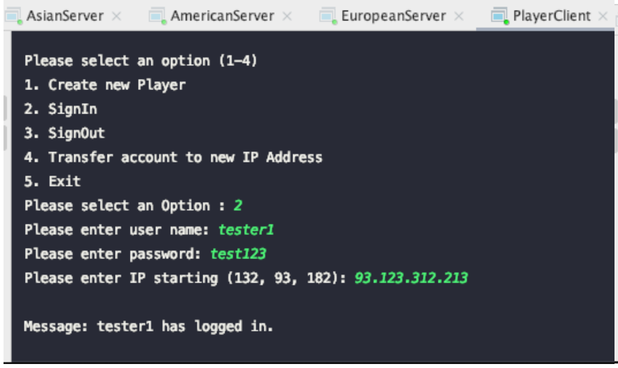
Player data is validated.
Player logged out successfully and message is displayed.

5. Transfer Player to another server - *playerSignOut* (*Username, IPAddress*)

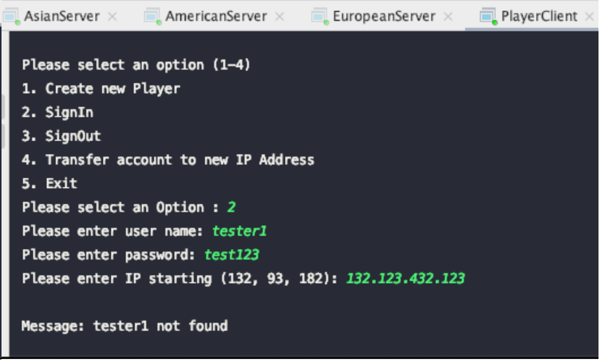


```
Please select an option (1-4)
1. Create new Player
2. SignIn
3. SignOut
4. Transfer account to new IP Address
5. Exit
Please select an Option : 4
Please enter user name: tester1
Please enter password: test123
Please enter Old IP: 132.218.153.123
Please enter New IP: 93.123.232.123

Message: tester1 has been transferred to - 93
```

The player is transferred to it's new server and then removed from old server.



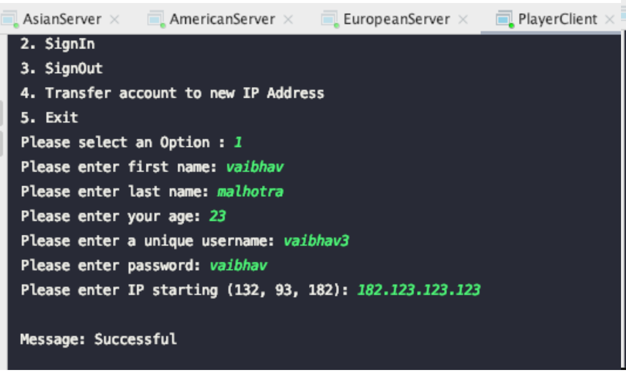
```
Please select an option (1-4)
1. Create new Player
2. SignIn
3. SignOut
4. Transfer account to new IP Address
5. Exit
Please select an Option : 2
Please enter user name: tester1
Please enter password: test123
Please enter IP starting (132, 93, 182): 93.123.312.213

Message: tester1 has logged in.
```



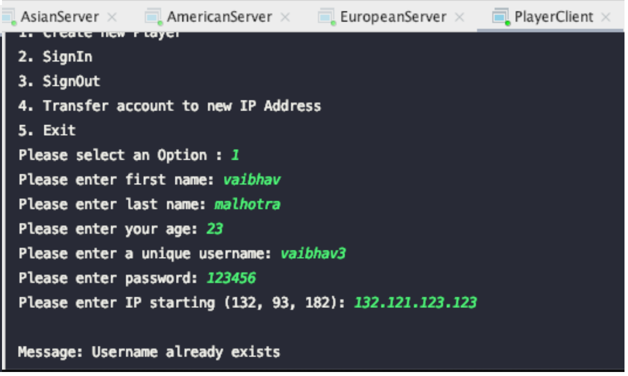
```
Please select an option (1-4)
1. Create new Player
2. SignIn
3. SignOut
4. Transfer account to new IP Address
5. Exit
Please select an Option : 2
Please enter user name: tester1
Please enter password: test123
Please enter IP starting (132, 93, 182): 132.123.432.123

Message: tester1 not found
```

The player is now on server IP 93(New Server). Player not available on server IP 132(Old Server)

6. Try to create a user (Already on another server)



```
2. SignIn
3. SignOut
4. Transfer account to new IP Address
5. Exit
Please select an Option : 1
Please enter first name: vaibhav
Please enter last name: malhotra
Please enter your age: 23
Please enter a unique username: vaibhav3
Please enter password: vaibhav
Please enter IP starting (132, 93, 182): 182.123.123.123

Message: Successful
```



```
1. Create new Player
2. SignIn
3. SignOut
4. Transfer account to new IP Address
5. Exit
Please select an Option : 1
Please enter first name: vaibhav
Please enter last name: malhotra
Please enter your age: 23
Please enter a unique username: vaibhav3
Please enter password: 123456
Please enter IP starting (132, 93, 182): 132.121.123.123

Message: Username already exists
```

Player creation unsuccessful on server 132, Because player already existed on server 182.

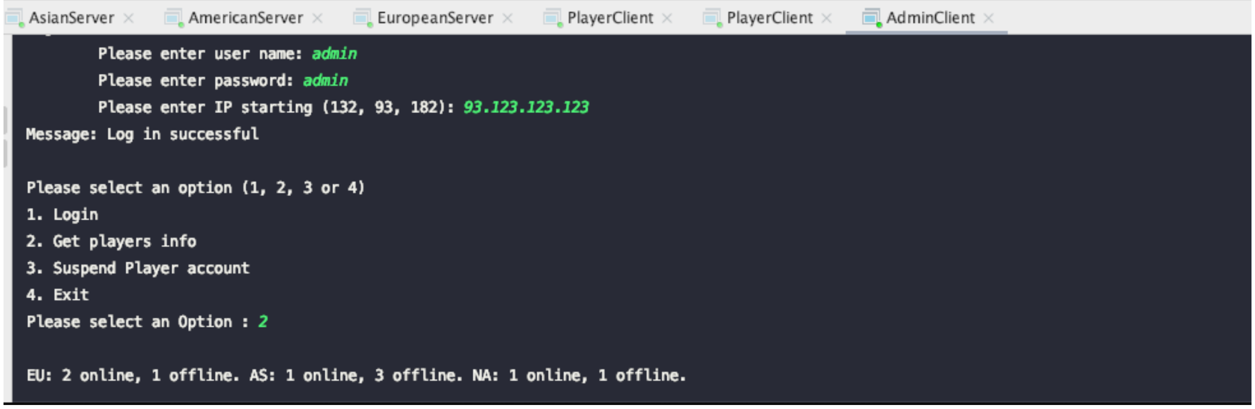
Admin testing:

In this section we will be testing Admin Client functionalities.

Test Number	Scenario	Reason
7	Get player status from different servers	Major functionality, Concurrent access
8	Suspend player account	Major functionality

7. Get Player status from different servers - *getPlayerStatus* (*AdminUsername, AdminPassword, IPAddress*)

a. On server IP 93

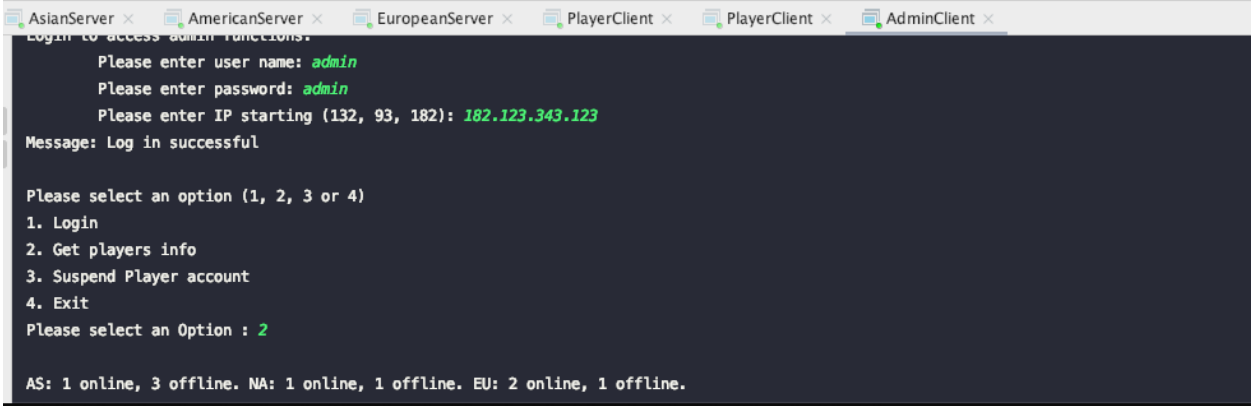


```
        Please enter user name: admin
        Please enter password: admin
        Please enter IP starting (132, 93, 182): 93.123.123.123
Message: Log in successful

Please select an option (1, 2, 3 or 4)
1. Login
2. Get players info
3. Suspend Player account
4. Exit
Please select an Option : 2

EU: 2 online, 1 offline. AS: 1 online, 3 offline. NA: 1 online, 1 offline.
```

Admin has logged in successfully on server IP 93 i.e server Europe.

Europe server sends request to Asia and America Server to gets their player status, appends it's result and displays.

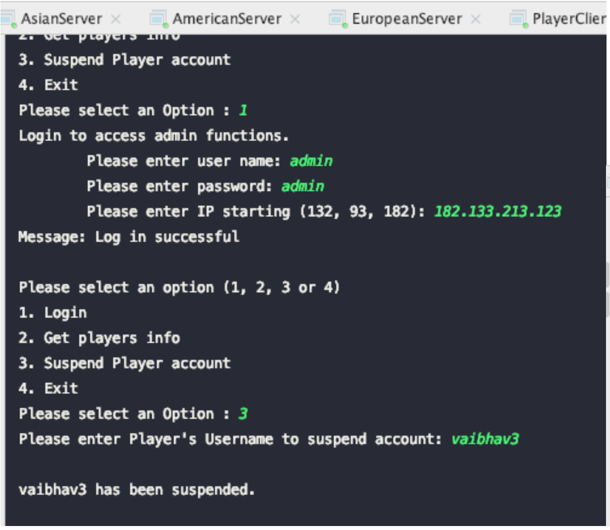
b. On server IP 182



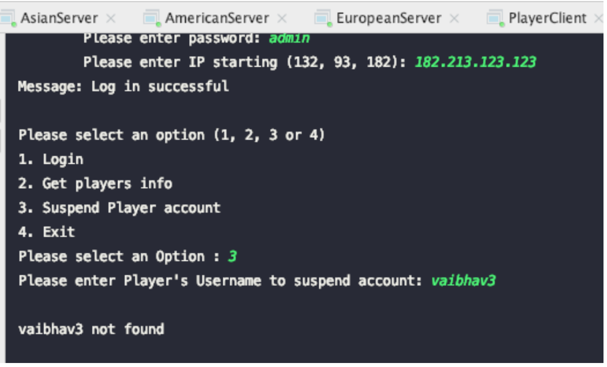
```
Login to access admin functions.
        Please enter user name: admin
        Please enter password: admin
        Please enter IP starting (132, 93, 182): 182.123.343.123
Message: Log in successful

Please select an option (1, 2, 3 or 4)
1. Login
2. Get players info
3. Suspend Player account
4. Exit
Please select an Option : 2

AS: 1 online, 3 offline. NA: 1 online, 1 offline. EU: 2 online, 1 offline.
```

Admin has logged in successfully on IP 182 i.e on server Asia.
Asia server sends request to Europe and America Server to get player status, appends result and displays.

8. Suspend Player account (AdminUsername, AdminPassword, AdminIP, UsernameToSuspend)



```
2. Get players info
3. Suspend Player account
4. Exit
Please select an Option : 1
Login to access admin functions.
        Please enter user name: admin
        Please enter password: admin
        Please enter IP starting (132, 93, 182): 182.133.213.123
Message: Log in successful

Please select an option (1, 2, 3 or 4)
1. Login
2. Get players info
3. Suspend Player account
4. Exit
Please select an Option : 3
Please enter Player's Username to suspend account: vaibhav3

vaibhav3 has been suspended.
```



```
        Please enter password: admin
        Please enter IP starting (132, 93, 182): 182.213.123.123
Message: Log in successful

Please select an option (1, 2, 3 or 4)
1. Login
2. Get players info
3. Suspend Player account
4. Exit
Please select an Option : 3
Please enter Player's Username to suspend account: vaibhav3

vaibhav3 not found
```

Admin suspended player with username – *vaibhav3* on server IP *182.*
Player could not login with username – *vaibhav3* on server IP *182* because it has been suspended.

Multithreading and atomicity testing:

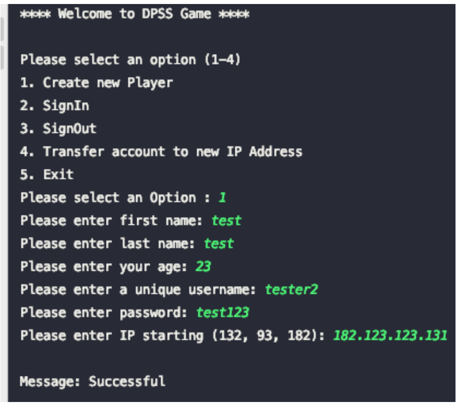
Test Number	Scenario	Reason
9	Multiple player account creation	Test Concurrency
10	Suspend player account	Test Atomicity

9. Multiple account creation

Scenario: 3 PlayerClients are trying to create new accounts simultaneously
Player1 – Username – tester2, server – 182 (Asia)
Player2 – Username – tester2, server – 93 (Europe)

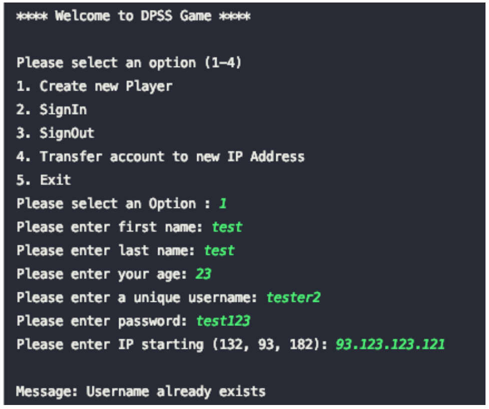
As we can see that Player1 was successful but as Username – tester2 was already taken at Asian server, Player2 was unable to create an account. The time difference between 2 calls is of microseconds. This tests the concurrent access to the data-structures in the project.

PLayerClient1



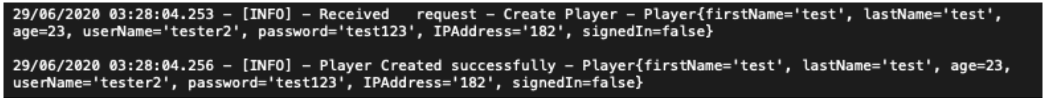
```
**** Welcome to DPSS Game ****

Please select an option (1-4)
1. Create new Player
2. SignIn
3. SignOut
4. Transfer account to new IP Address
5. Exit
Please select an Option : 1
Please enter first name: test
Please enter last name: test
Please enter your age: 23
Please enter a unique username: tester2
Please enter password: test123
Please enter IP starting (132, 93, 182): 182.123.123.131

Message: Successful
```

PlayerClient2



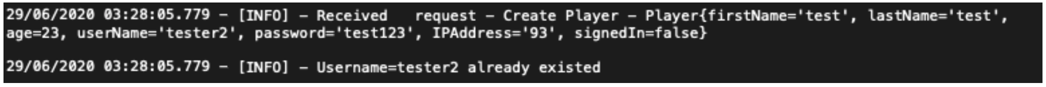
```
**** Welcome to DPSS Game ****

Please select an option (1-4)
1. Create new Player
2. SignIn
3. SignOut
4. Transfer account to new IP Address
5. Exit
Please select an Option : 1
Please enter first name: test
Please enter last name: test
Please enter your age: 23
Please enter a unique username: tester2
Please enter password: test123
Please enter IP starting (132, 93, 182): 93.123.123.121

Message: Username already exists
```

Logs from Asia



```
29/06/2020 03:28:04.253 - [INFO] - Received   request - Create Player - Player{firstName='test', lastName='test',
age=23, userName='tester2', password='test123', IPAddress='182', signedIn=false}

29/06/2020 03:28:04.256 - [INFO] - Player Created successfully - Player{firstName='test', lastName='test', age=23,
userName='tester2', password='test123', IPAddress='182', signedIn=false}
```

Logs from Europe

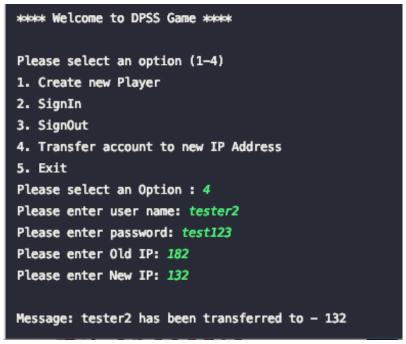


```
29/06/2020 03:28:05.779 - [INFO] - Received   request - Create Player - Player{firstName='test', lastName='test',
age=23, userName='tester2', password='test123', IPAddress='93', signedIn=false}

29/06/2020 03:28:05.779 - [INFO] - Username=tester2 already existed
```

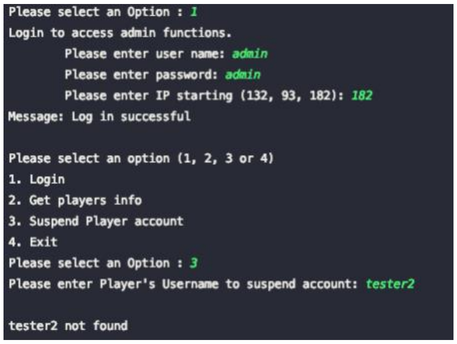
10. Suspending and transferring account at the same time

Scenario: Transfer request and suspend request are generated at the same time. For atomicity only one of the requests should be successful. In our case transfer was successful.

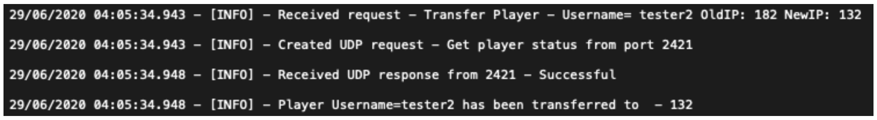
Vaibhav Malhotra – 40079373



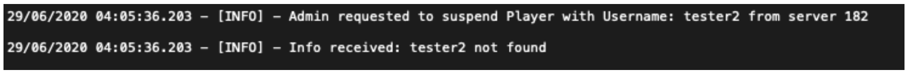
```
**** Welcome to DPSS Game ****

Please select an option (1-4)
1. Create new Player
2. SignIn
3. SignOut
4. Transfer account to new IP Address
5. Exit
Please select an Option : 4
Please enter user name: tester2
Please enter password: test123
Please enter Old IP: 182
Please enter New IP: 132

Message: tester2 has been transferred to - 132
```



```
Please select an Option : 1
Login to access admin functions.
        Please enter user name: admin
        Please enter password: admin
        Please enter IP starting (132, 93, 182): 182
Message: Log in successful

Please select an option (1, 2, 3 or 4)
1. Login
2. Get players info
3. Suspend Player account
4. Exit
Please select an Option : 3
Please enter Player's Username to suspend account: tester2

tester2 not found
```



```
29/06/2020 04:05:34.943 - [INFO] - Received request - Transfer Player - Username= tester2 OldIP: 182 NewIP: 132
29/06/2020 04:05:34.943 - [INFO] - Created UDP request - Get player status from port 2421
29/06/2020 04:05:34.948 - [INFO] - Received UDP response from 2421 - Successful
29/06/2020 04:05:34.948 - [INFO] - Player Username=tester2 has been transferred to  - 132
```

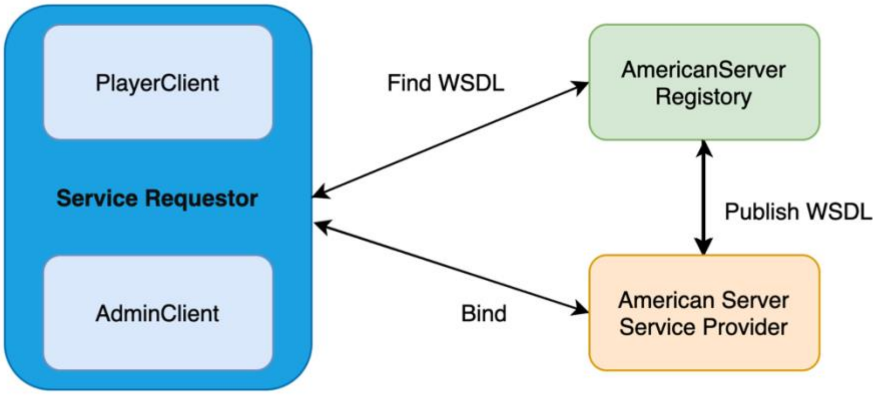


```
29/06/2020 04:05:36.203 - [INFO] - Admin requested to suspend Player with Username: tester2 from server 182
29/06/2020 04:05:36.203 - [INFO] - Info received: tester2 not found
```

Vaibhav Malhotra – 40079373

Server Architecture Design

For simplicity only one server is shown in below diagram:



Service Provider:

From an architectural perspective, it is the platform that hosts the services.

Service Requestor

Service requestor is the application that is looking for and invoking or initiating an interaction with a service. The browser plays the requester role, driven by a consumer or a program without a user interface.

Service Registry

Service requestors find service and obtain binding information for services during development.

Steps followed on start of each server:

- Code and compile the server classes
- Use wsgen to generate required artifacts to deploy them as a service.
- Use wsimport to generate and compile stub files.

Steps followed by each client:

- Create a URL objects with address of the servers
- Use namespaceURI and localPart to generate **QName** objects.
- Create a service object using the URL and QName objects.
- Get the service from the designated port number.