Ticketing
Web Service

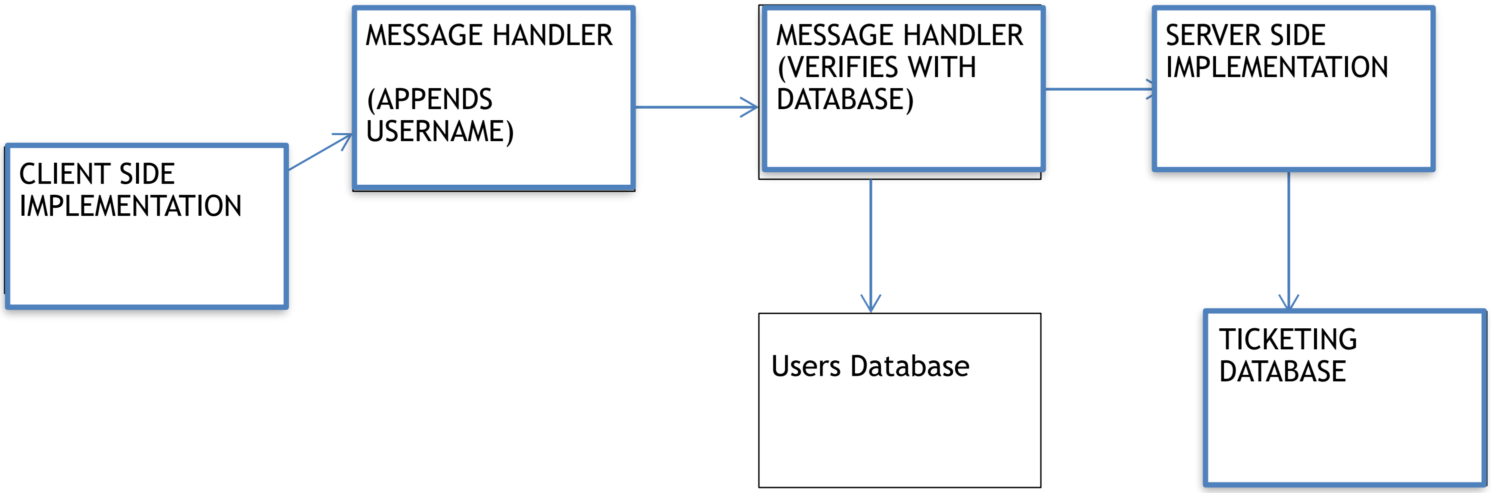
Introduction

The following is the documentation for an application Ticket booking application for a stadium, which has four stands and with each stand having the tickets from A to F. The application is deployed as a SOAP Web service so that the client can access the server code irrespective of the platform. An additional coordination protocol is implemented to prevent unauthorized users from accessing the application.

The documentation includes the design of the web service, how a coordination protocol is implemented, client side and server side implementation and the screen shots of the running application.

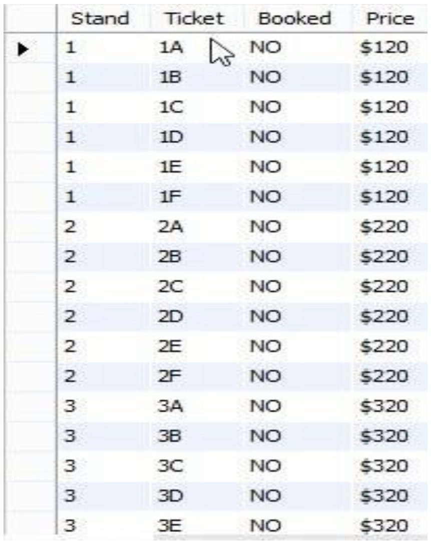
Design of Web Service

The figure given below describes the structure of how the created web service is called from the client
side and the implementation of the coordination protocol.


CLIENT SIDE IMPLEMENTATION
MESSAGE HANDLER (APPENDS USERNAME)
MESSAGE HANDLER (VERIFIES WITH DATABASE)
SERVER SIDE IMPLEMENTATION
Users Database
TICKETING DATABASE

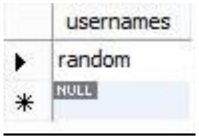
Before running the application MySQL test bench is loaded on to the local machine and two databases are created namely User_List and Tickets. The various columns are loaded onto the databases as follows. Then once the project is loaded into NetBeans IDE MySql driver jars are added to the project.

Tickets



Stand	Ticket	Booked	Price
1	1A	NO	$120
1	1B	NO	$120
1	1C	NO	$120
1	1D	NO	$120
1	1E	NO	$120
1	1F	NO	$120
2	2A	NO	$220
2	2B	NO	$220
2	2C	NO	$220
2	2D	NO	$220
2	2E	NO	$220
2	2F	NO	$220
3	3A	NO	$320
3	3B	NO	$320
3	3C	NO	$320
3	3D	NO	$320
3	3E	NO	$320

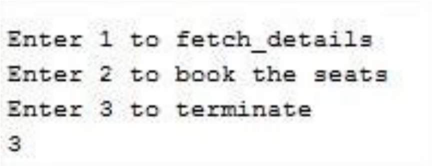
User_List



usernames
random
NULL

CLIENT SIDE IMPLEMENTATION

The client side application asks for user input from the console. Based on the options, the corresponding methods on the server side will be called.



```
Enter 1 to fetch_details
Enter 2 to book the seats
Enter 3 to terminate
3
```

The inputs will be asked from the user continuously unless the input 3 is given to terminate the application. A message handler is provided at client side that appends the username before sending the message to the server.

SERVER SIDE IMPLEMENTATION

At the server side there is a handler that verifies with the user's database that checks if he/she is registered with the database. If registered with database the message is passed onto the server side that provides the response based on the input from the user console. The server side implementation then fetches the required details from the Tickets database and updates the database.

COORDINATION PROTOCOL (AUTHORIZATION CHECK)

The coordination protocol between the server and the client is provided by means of two message handlers at the client side and server side. The client side handler appends the username before any message that is transmitted to the server. The username is obtained from the System by means of a Java method.

Once the user name is appended the message is then passed to the server side to call the message. Before any message reaches the server side implementation, it has to pass through the server side handler. At the server side, the handler verifies with user_list table to check if he/she is registered with the database, if not an exception is thrown to the client side that is caught. If the user is registered, the message is transferred to the server side that provides the response based on the details obtained from Tickets table.

TOOLS THAT WERE USED.

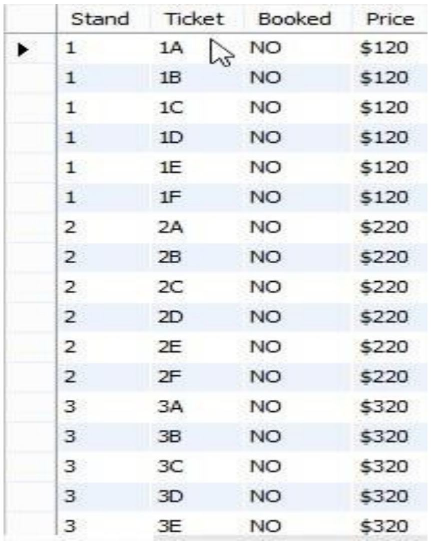
Programming Language : Java, NetBeans 8.0 IDE

Database : MySQL

DATABASE

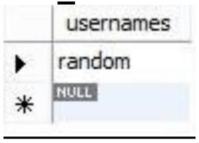
SCREENHOTS

Tickets



	Stand	Ticket	Booked	Price
▶	1	1A	NO	$120
	1	1B	NO	$120
	1	1C	NO	$120
	1	1D	NO	$120
	1	1E	NO	$120
	1	1F	NO	$120
	2	2A	NO	$220
	2	2B	NO	$220
	2	2C	NO	$220
	2	2D	NO	$220
	2	2E	NO	$220
	2	2F	NO	$220
	3	3A	NO	$320
	3	3B	NO	$320
	3	3C	NO	$320
	3	3D	NO	$320
	3	3E	NO	$320

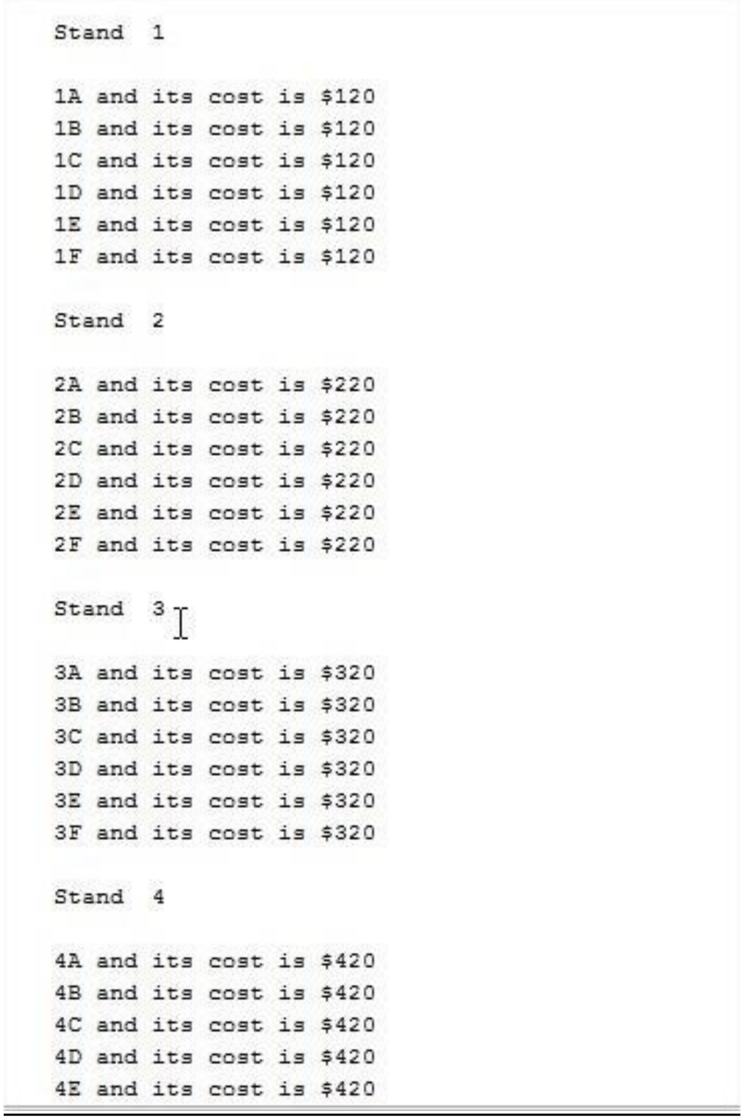
User_List



	usernames
▶	random
*	NULL

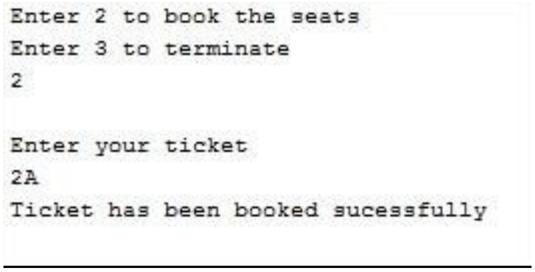
SCREENSHOTS OF THE RUNNING APPLICATION

1) FETCH THE DETAILS



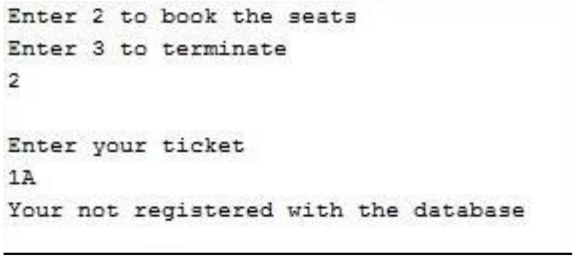
```
Stand  1

1A and its cost is $120
1B and its cost is $120
1C and its cost is $120
1D and its cost is $120
1E and its cost is $120
1F and its cost is $120

Stand  2

2A and its cost is $220
2B and its cost is $220
2C and its cost is $220
2D and its cost is $220
2E and its cost is $220
2F and its cost is $220

Stand  3

3A and its cost is $320
3B and its cost is $320
3C and its cost is $320
3D and its cost is $320
3E and its cost is $320
3F and its cost is $320

Stand  4

4A and its cost is $420
4B and its cost is $420
4C and its cost is $420
4D and its cost is $420
4E and its cost is $420
```

2) Ticket Booked Successfully



```
Enter 2 to book the seats
Enter 3 to terminate
2

Enter your ticket
2A
Ticket has been booked sucessfully
```

3) Not registered



```
Enter 2 to book the seats
Enter 3 to terminate
2

Enter your ticket
1A
Your not registered with the database
```