WRITE UP:

Assignment - 02.

* Title: Socket Programming
* Date of completion:
* Problem Statement: Enhance the above system. (assignment 1) with the help of socket programming. Use client server architecture
* Objective: To understand & implement socket programming in Java.
* Outcome: We understood socket programming in Java using client server architecture.
* S/W And H/W Requirements:
  IntelliJ IDE, Windows 10 64 bit, i5 processor.
* Prerequisites:
  Object Oriented Programming.
* Theory:

Java creates network applications using sockets. Program running on client m/c makes request to program running on server, involves networking services provides by transport layer. Transport layer has two protocols (TCP, UDP). These are use port to map incoming incoming data to a particular process on computer. Multiple services can be run on same time so use ports to distinguish

Teacher's Sign.:

between them

A socket is a software end point that establishes bidirectional communication between a server program & one or more client programs. The socket associates the server program with a specific hardware port on the machine where it runs on so any client program anywhere in the network with a socket associated with that same port can communicate with the server program

Socket is bound to specific port. Socket acts as interface between application & network

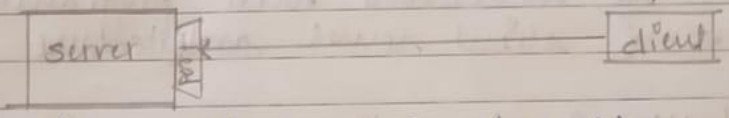

fig a client making connection request to the server.

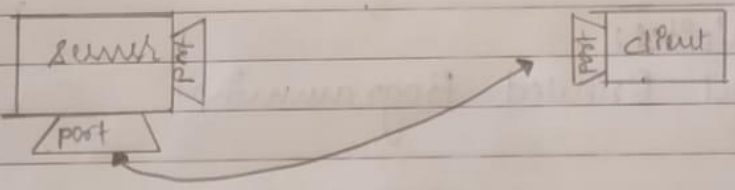

fig. session established with temporary ports.

- TCP/IP Socket Programming

a) steps for creating server programming

i) Open the server socket

```
ServerSocket server = new ServerSocket(port);
```

ii) Wait for the client Request

```
Socket client = server.accept();
```

iii) Create I/O streams for communicating to client
DataInputStream is = new DataInputStream(client.getInputStream());
DataOutputStream os = new DataOutputStream(client.getOutputStream());

iv) Perform communication with client
String line = is.readLine();
v) Send to client : os.writeBytes("Hello\n");
vi) Close socket:
client.close(); or server.close();

b) Steps for creating client program:
i) Create a socket Object:
Socket client = new Socket(server, port_id);
ii) Create I/O streams for communicating with server.
is = new DataInputStream(client.getInputStream());
os = new DataOutputStream(client.getOutputStream());
iii) Perform I/O with server:
String line = is.readLine();
iv) Send data to server: os.writeByte("Hi" + "\n");
v) Close socket when done:
client.close();

- UDP Socket Programming:

TCP guarantees the delivery of packets and preserves their order on destination.
Sometimes we can use a lighter transport protocol like UDP protocol.

Teacher's Sign.:

Page No.
Date

Service is accomplished by the UDP protocol which conveys datagram packets. Each packet contains message, length, host, port number. Java supports two packets Datagram socket, Datagram Packet. DatagramPacket has various constructors (ex. (byte [] buf, int length, InetAddress addr, int port) and key Methods : byte [] getData(), int getLength(); void setData(byte [] b), void length(int len)

* Test cases:

	Input	Expected Output	Result
i)	username: varunkarwa password: varunkarwa 1. Book a ticket Destination city: Mumbai Seat No: 10 Name: kaveri	Booked! Booking-id = 556739	Success.
ii)	Check Reservation Booking id: 556739	Your have Booking!	Success.
iii)	username: admin password: admin Cancel Bus: Nashik	Removed	Success
iv)	Add City: City: Solapur	Added!	Success.

Teacher's Sign.:

v) vanrakarwa logged in
Availability:
City: Nashik
Error! Success.

vi) Availability
City: Solapur
Displays bus seats Success.

vii) admin logged in
Add_Bus
City: Nashik
Added! Success.

viii) Delete City:
City: Jalgaon
Deleted! Success.

ix) vanrakarwa logged in
Availability
City: Nashik
Displays bus seat Success.

x) Display city list
Displayed Success

xi) Cancel Booking
Booking_id: 556739
cancelled! Success.

* Conclusion:-
Thus, implemented Bus Reservation System using Socket programming in Java.

Teacher's Sign:

Server Side Code:

```
import java.util.*;
import java.io.*;
import java.net.ServerSocket;
import java.net.Socket;

class User implements Serializable
{
   private String name;
   private String password;
   private String username;
   private String contactNo;

   User(String username,String name,String contactNumber,String password) {
      this.setName(name);
      this.setContactNumber(contactNumber);
      this.setPassword(password);
      this.setUsername(username);
   }
   User() {}
   public String getName() {
      return name;
   }

   public void setName(String name) {
      this.name = name;
   }

   public String getPassword() {
      return password;
   }

   public void setPassword(String password) {
      this.password = password;
   }

   public String getUsername() {
      return username;
   }

   public void setUsername(String username) {
      this.username = username;
   }

   public String getContactNumber() {
      return contactNo;
   }

   public void setContactNumber(String contactNumber) {
      this.contactNo = contactNumber;
   }

}

class Booking implements Serializable
{
   public ArrayList<String> Cities=new ArrayList<String>();
   public HashMap<String, ArrayList> buses = new HashMap<String, ArrayList>();
```

```
public ArrayList<customer> customer_list = new ArrayList<customer>();
public void add_cities()
{
   Cities.add("Mumbai");
   Cities.add("Ahemdabad");
   Cities.add("Kolhapur");
   Cities.add("Nashik");
   Cities.add("Indore");
   Cities.add("Nagpur");
   Cities.add("Jalgaon");
}
public void add_buses()
{
   for(String c:Cities)
   {
      ArrayList<Integer> bus = new ArrayList<Integer>();
      for(Integer i=0;i<40;i++)
         bus.add(0);
      buses.put(c,bus);
   }
}
public String add_city(String city)
{
   if(!Cities.contains(city))
   {
      Cities.add(city);
      add_bus(city);
      System.out.println("City added");
      return "Added!";
   }
   System.out.println("City already added");
   return "Error!";
}
public String delete_city(String city)
{
   if(Cities.contains(city))
   {
      Cities.remove(city);
      buses.remove(city);
      System.out.println("City deleted");
      return "Deleted!";
   }
   System.out.println("City not present!");
   return "Error!";
}
public String add_bus(String city)
{
   if(Cities.contains(city))
   {
      ArrayList<Integer> bus=new ArrayList<Integer>();
      for(Integer i=0;i<40;i++)
         bus.add(0);
      buses.put(city,bus);
      System.out.println("Bus added");
      return "Added!";
   }
   System.out.println("City not Available");
   return "Error!";
}
```

```
public String cancel_bus(String city)
{
   if(Cities.contains(city))
   {
      buses.remove(city);
      return "Removed";
   }
   System.out.println("City not present");
   return  "Error!";
}
public String book(customer c)
{
   int seat;
   Random rand=new Random();
   if(!Cities.contains(c.destination_city))
   {
      return "Sorry!! This city is not available.";
   }
   if(!buses.containsKey(c.destination_city))
   {
      return "Bus is cancelled for this city";
   }
   ArrayList <Integer> bus=buses.get(c.destination_city);
   //available(c.destination_city);
   if(!bus.contains(0))
   {
      return "This Bus is Full!!";
   }
   for(int i=0;i<c.number_of_passengers;i++)
   {
      seat=c.seat_no.get(i);
      if(bus.get(seat-1)==0)
      {
         bus.set(seat-1,1);
         c.seat_no.add(seat);
      }
      else
      {
         return "Seat is already booked!!";
      }
   }
   String city=c.destination_city;
   buses.replace(city,bus);
   c.booking_id=rand.nextInt(9999999);
   //System.out.println("Tickets are Booked!!");
   //c.print_details();
   customer_list.add(c);
   return "Booked!";
}
public Object check(int booking_id)
{
   for(customer cust:customer_list)
   {
      if(cust.booking_id==booking_id)
         return cust;
   }
   return null;
}
public String cancel(int booking_id)
```

```
   {
      customer c = (customer) check(booking_id);
      if(c==null)
         return "Error!";
      ArrayList<Integer> bus = buses.get(c.destination_city);
      for(int i=0;i<c.number_of_passengers;i++)
      {
         bus.set(c.seat_no.get(i)-1,0);
      }
      buses.replace(c.destination_city,bus);
      int i=0;
      for(customer cust:customer_list)
      {
         if(cust.booking_id==booking_id)
         {
            customer_list.remove(i);
            break;
         }
         i++;
      }
      return "Cancelled";
   }
   public ArrayList<Integer> available(String city) {
      if (Cities.contains(city)) {
         if (buses.containsKey(city)) {
            ArrayList<Integer> bus = buses.get(city);
            return bus;
         }
      }
      return null;
   }
}

public class Server
{
   public Vector<User> Users = new Vector<User>();
   Server() {
      User a = new User("varunkarwa","Varun","7617209448","varunkarwa");
      Users.add(a);
      User b = new User("mpm1712","Miti","8378099502","mitimehta");
      Users.add(b);
   }


   public void register_user () {

   }

   public boolean checkUser(String username) {
      for(User s : Users) {
         if(s.getUsername().equals(username)) {
            return true;
         }
      }
      return false;
   }

   public User validatingUser(String usern, String pass) {
      for(User s: Users) {
```

```
         if(s.getUsername().equals(usern)) {
            if(s.getPassword().equals(pass)) {
               return s;
            } else {
               return null;
            }
         }
      }

      return null;
   }

   public static <integer> void main(String args[]) throws Exception
   {
      String clientSentence;
      String serverSentence;
      ArrayList<Integer> bus = new ArrayList<Integer>();
      Booking b = new Booking();
      customer c = new customer();
      int booking_id;
      ServerSocket welcomeSocket = new ServerSocket(4353);
      System.out.println("ServerSocket awaiting connection...");
      Socket connectionSocket = welcomeSocket.accept();
      System.out.println("Connection from " + connectionSocket);
      BufferedReader inFromClient = new BufferedReader(new
InputStreamReader(connectionSocket.getInputStream()));
      ObjectInputStream is = new
ObjectInputStream(connectionSocket.getInputStream());
      ObjectOutputStream os = new
ObjectOutputStream(connectionSocket.getOutputStream());
      DataOutputStream outToClient = new
DataOutputStream(connectionSocket.getOutputStream());
      Server s = new Server();
      b.add_cities();
      b.add_buses();

      while(true) {
         System.out.println("Waiting for Input");
         boolean isAuth = false;
         boolean admin = false;
         while (!isAuth)
         {
            String username = inFromClient.readLine();
            if(username.equals("admin")) {
               System.out.println("Getting password for admin");
               //outToClient.writeBytes("Enter password for admin" + "\n");
               String password = inFromClient.readLine();
               if(password.equals("admin")) {
                  System.out.println("Welcome admin");
                  outToClient.writeBytes("Welcome" + "\n");
                  admin=true;
                  break;
               } else {
                  outToClient.writeBytes("Invalid password!!");
               }
            }
            System.out.println("Username:" + username);
            if(s.checkUser(username)){
               System.out.println("User Present");
```

```
            outToClient.writeBytes("Present" + "\n");
        } else {
            System.out.println("Username invalid");
            outToClient.writeBytes("Username invalid" + "\n");
            continue;
        }
        String pass = inFromClient.readLine();
        User p = s.validatingUser(username,pass);
        if(p != null) {
            outToClient.writeBytes("Successful" + "\n");
            os.writeObject(p);
            isAuth=true;
            break;
        } else {
            outToClient.writeBytes("Credentials false" + "\n");
        }
    }
    if(!admin)
    {
        while(true)
        {
            clientSentence = (String) inFromClient.readLine();
            System.out.println(clientSentence);
            if(clientSentence.equals("6"))
            {
                break;
            }
            switch(clientSentence) {
                case "1":
                    customer cust = (customer) is.readObject();
                    serverSentence = b.book(cust);
                    outToClient.writeBytes(serverSentence + "\n");
                    if(serverSentence.equals("Booked!"))
                        os.writeObject(cust);
                    break;
                case "2":
                    clientSentence = inFromClient.readLine();
                    booking_id = Integer.parseInt(clientSentence);
                    serverSentence = b.cancel(booking_id);
                    outToClient.writeBytes(serverSentence + "\n");
                    break;
                case "3":
                    clientSentence = inFromClient.readLine();
                    booking_id = Integer.parseInt(clientSentence);
                    c = (customer) b.check(booking_id);
                    if (c != null) {
                        outToClient.writeBytes("Yes" + "\n");
                        os.writeObject(c);
                    } else
                        outToClient.writeBytes("Error!");
                    break;
                case "4":
                    clientSentence = inFromClient.readLine();
                    bus = b.available(clientSentence);
                    if (bus != null) {
                        outToClient.writeBytes("Success" + "\n");
                        os.writeObject(bus);
                    } else
                        outToClient.writeBytes("Error!" + "\n");
```

```
                    break;
                case "5":
                    os.writeObject(b.Cities);
                    break;
            }
        }
    }
    else
    {
        while(true)
        {
            clientSentence = (String) inFromClient.readLine();
            System.out.println(clientSentence);
            if(clientSentence.equals("6"))
            {
                break;
            }
            switch(clientSentence)
            {
                case "1":
                    clientSentence = inFromClient.readLine();
                    serverSentence = b.add_city(clientSentence);
                    outToClient.writeBytes(serverSentence+"\n");
                    break;
                case "2":
                    clientSentence = inFromClient.readLine();
                    serverSentence = b.delete_city(clientSentence);
                    outToClient.writeBytes(serverSentence+"\n");
                    break;
                case "3":
                    clientSentence = inFromClient.readLine();
                    serverSentence = b.add_bus(clientSentence);
                    outToClient.writeBytes(serverSentence+"\n");
                    break;
                case "4":
                    clientSentence = inFromClient.readLine();
                    serverSentence = b.cancel_bus(clientSentence);
                    outToClient.writeBytes(serverSentence+"\n");
                    break;
                case "5":
                    os.writeObject(b.customer_list);
                    break;
            }
        }
    }

    clientSentence = inFromClient.readLine();
    if(clientSentence.equals("end")) {
        connectionSocket.close();
    } else {
        System.out.println("Connection Live");
    }
  }
 }
}
```

Client Side Code:

```
import java.io.*;
import java.util.*;
import java.net.Socket;
import java.net.SocketException;

class customer implements Serializable
{
   public String name,destination_city,contact_number;
   public int number_of_passengers,booking_id;
   ArrayList<String> passengers=new ArrayList<String>();
   ArrayList<Integer> seat_no = new ArrayList<Integer>();
   public void set_customer(User us)
   {
      this.name = us.getName();
      this.contact_number = us.getContactNumber();
   }
   public void print_customer()
   {
      System.out.println("--------Customer Details--------");
      System.out.println("Name of the Customer:" + name);
      System.out.println("Contact Number:" + contact_number);
   }
   public void book_ticket()
   {
      Scanner sc=new Scanner(System.in);
      System.out.println("Enter Destination City");
      destination_city=sc.next();
      System.out.println("Enter number of passengers");
      number_of_passengers=Integer.parseInt(sc.next());
      for(int i=0;i<number_of_passengers;i++)
      {
         System.out.println("Enter Passenger's Name");
         passengers.add(sc.next());
         System.out.println("Enter Seat No.:");
         seat_no.add(sc.nextInt());
      }
   }
   public void booking_details()
   {
      System.out.println("=====Customer Deatils=====");
      System.out.println("Booking Id:"+booking_id);
      System.out.println("Destination City:" + destination_city);
      System.out.println("List of Passenger's:");
      for(Integer i=0;i<number_of_passengers;i++)
      {
         System.out.println(passengers.get(i));
         System.out.println("Seat Number:"+seat_no.get(i));
      }
   }
   public void display_bus(ArrayList<Integer> bus)
   {
      System.out.println("Seat Availables(0 means Available):");
      int i=0;
      while(i<bus.size())
      {
         System.out.print("\t"+bus.get(i++)+"\t"+bus.get(i++));
         System.out.println("\t"+bus.get(i++)+"\t"+bus.get(i++));
```

```
      }
   }
   public void display_cities(ArrayList<String> cities)
   {
      for(String city:cities)
         System.out.println(city);
   }
   public void display_customers(ArrayList<customer> list)
   {
      int i=1;
      System.out.println("------Customer List------");
      for(customer c:list)
      {
         System.out.println(i++ +". "+c.name);
         System.out.println("Booking Details");
         c.booking_details();
      }
   }
}

public class Client
{
   public static void main(String args[]) throws Exception {
      String serverSentence, choice;
      String booking_id, city;
      ArrayList<customer> list = new ArrayList<customer>();
      customer c = new customer();
      Scanner sc = new Scanner(System.in);

      Socket clientSocket = new Socket("localhost", 4353);
      System.out.println("Connected to Server");

      BufferedReader inFromUser = new BufferedReader(new
InputStreamReader(System.in));

      DataOutputStream outToServer = new
DataOutputStream(clientSocket.getOutputStream());
      ObjectOutputStream os = new
ObjectOutputStream(clientSocket.getOutputStream());
      ObjectInputStream is = new ObjectInputStream(clientSocket.getInputStream());

      BufferedReader inFromServer = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));

      while (true) {
         String username, password;
         ArrayList<String> city_list = new ArrayList<String>();
         ArrayList<Integer> bus = new ArrayList<Integer>();
         User user = new User();
         boolean isAuth = false;
         boolean admin = false;
         while (!isAuth) {
            System.out.println("Enter username:");
            username = inFromUser.readLine();
            if (username.equals("admin")) {
               outToServer.writeBytes(username + "\n");
               System.out.println("Enter password:");
               password = inFromUser.readLine();
               outToServer.writeBytes(password + "\n");
```

```
                serverSentence = inFromServer.readLine();
                if (serverSentence.equals("Welcome")) {
                    admin = true;
                    break;
                } else {
                    System.out.println("Invalid Credentials");
                    continue;
                }
            }
            outToServer.writeBytes(username + "\n");
            serverSentence = inFromServer.readLine();
            if (serverSentence.equals("Present")) {
                System.out.println("Enter Password");
                password = inFromUser.readLine();
                outToServer.writeBytes(password + "\n");
            }
            else {
                System.out.println("Invalid Credentials");
                continue;
            }
            serverSentence = inFromServer.readLine();
            if (serverSentence.equals("Successful")) {
                user = (User) is.readObject();
                c.set_customer(user);
                System.out.println("Welcome " + user.getName());
                isAuth = true;
                break;
            }
        }

        if (!admin) {
            c.print_customer();
            while (true) {
                System.out.println("--------BUS RESERVATION SYSTEM--------");
                System.out.println("1.Book Tickets\n2.Cancel Ticket(s)\n3.Check
Reservation\n4.Seat Availability\n5.Cities Available\n6.Exit");
                choice = inFromUser.readLine();
                if (choice.equals("6")) {
                    outToServer.writeBytes("6" + "\n");
                    break;
                }
                switch (choice) {
                    case "1":
                        c.book_ticket();
                        outToServer.writeBytes("1" + "\n");
                        os.writeObject(c);
                        serverSentence = (String) inFromServer.readLine();
                        if (serverSentence.equals("Booked!")) {
                            c = (customer) is.readObject();
                            c.booking_details();
                        } else {
                            System.out.println("Error:" + serverSentence);
                        }
                        break;
                    case "2":
                        System.out.println("Enter booking_id:");
                        booking_id = inFromUser.readLine();
                        outToServer.writeBytes("2" + "\n");
                        outToServer.writeBytes(booking_id + "\n");
```

```
                    serverSentence = inFromServer.readLine();
                    System.out.println(serverSentence);
                    break;
                case "3":
                    System.out.println("Enter booking_id:");
                    booking_id = inFromUser.readLine();
                    outToServer.writeBytes("3" + "\n");
                    outToServer.writeBytes(booking_id + "\n");
                    serverSentence = inFromServer.readLine();
                    if (serverSentence.equals("Yes")) {
                        c = (customer) is.readObject();
                        c.booking_details();
                    } else {
                        System.out.println(serverSentence);
                    }
                    break;
                case "4":
                    System.out.println("Enter destination city:");
                    city = inFromUser.readLine();
                    outToServer.writeBytes("4" + "\n");
                    outToServer.writeBytes(city + "\n");
                    serverSentence = inFromServer.readLine();
                    if (serverSentence.equals("Error!")) {
                        System.out.println("Some Error has occurred!");
                    } else {
                        bus = (ArrayList) is.readObject();
                        c.display_bus(bus);
                    }
                    break;
                case "5":
                    outToServer.writeBytes("5" + "\n");
                    city_list = (ArrayList) is.readObject();
                    c.display_cities(city_list);
                    break;
            }
        }
    } else {
        while (true) {
            System.out.println("1.Add City\n2.Delete City\n3.Add Bus\n4.Cancel
Bus\n5.Customer List\n6.Exit");
            choice = inFromUser.readLine();
            if (choice.equals("6")) {
                outToServer.writeBytes("6" + "\n");
                break;
            }
            switch(choice)
            {
                case "1":
                    System.out.println("Enter City Name:");
                    city  = inFromUser.readLine();
                    outToServer.writeBytes("1"+"\n");
                    outToServer.writeBytes(city+"\n");
                    serverSentence = inFromServer.readLine();
                    System.out.println(serverSentence);
                    break;
                case "2":
                    System.out.println("Enter City Name:");
                    city = inFromUser.readLine();
                    outToServer.writeBytes("2"+"\n");
```

```
                    outToServer.writeBytes(city+"\n");
                    serverSentence = inFromServer.readLine();
                    System.out.println(serverSentence);
                    break;
                case "3":
                    System.out.println("Enter Destination City:");
                    city = inFromUser.readLine();
                    outToServer.writeBytes("3"+"\n");
                    outToServer.writeBytes(city+"\n");
                    serverSentence = inFromServer.readLine();
                    System.out.println(serverSentence);
                    break;
                case "4":
                    System.out.println("Enter Destination City:");
                    city = inFromUser.readLine();
                    outToServer.writeBytes("4"+"\n");
                    outToServer.writeBytes(city+"\n");
                    serverSentence = inFromServer.readLine();
                    System.out.println(serverSentence);
                    break;
                case "5":
                    outToServer.writeBytes("5"+"\n");
                    list = (ArrayList<customer>) is.readObject();
                    c.display_customers(list);
                    break;
                }
            }
        }
        System.out.println("Do you want to continue?(Y or N)");
        String choice1 = inFromUser.readLine();
        if(choice1.equals("N"))
        {
            outToServer.writeBytes("end" + "\n");
            clientSocket.close();
            break;
        }
        else
        {
            outToServer.writeBytes("Y"+"\n");
        }
    }
  }
}
```

OUTPUT:

Client Side:

Connected to Server

Enter username:

varunkarwa

Enter Password

varunkarwa

Welcome Varun

--------Customer Details--------

Name of the Customer:Varun

Contact Number:7617209448

---------BUS RESERVATION SYSTEM--------

1.Book Tickets

2.Cancel Ticket(s)

3.Check Reservation

4.Seat Availability

5.Cities Available

6.Exit

1

Enter Destination City

Mumbai

Enter number of passengers

1

Enter Passenger's Name

Kaveri

Enter Seat No.:

10

=====Customer Deatils=====

Booking Id:5535220

Destination City:Mumbai

List of Passenger's:

Kaveri

Seat Number:10

--------BUS RESERVATION SYSTEM--------

1.Book Tickets

2.Cancel Ticket(s)

3.Check Reservation

4.Seat Availability

5.Cities Available

6.Exit

3

Enter booking_id:

5535220

=====Customer Deatils=====

Booking Id:5535220

Destination City:Mumbai

List of Passenger's:

Kaveri

Seat Number:10

--------BUS RESERVATION SYSTEM--------

1.Book Tickets

2.Cancel Ticket(s)

3.Check Reservation

4.Seat Availability

5.Cities Available

6.Exit

6

Do you want to continue?(Y or N)

Y

Enter username:

admin

Enter password:

admin

1.Add City

2.Delete City

3.Add Bus

4.Cancel Bus

5.Customer List

6.Exit

4

Enter Destination City:

Nashik

Removed

1.Add City

2.Delete City

3.Add Bus

4.Cancel Bus

5.Customer List

6.Exit

1

Enter City Name:

Solapur

Added!

1.Add City

2.Delete City

3.Add Bus

4.Cancel Bus

5.Customer List

6.Exit

6

Do you want to continue?(Y or N)

Y

Enter username:

varunkarwa

Enter Password

varunkarwa

Welcome Varun

--------Customer Details--------

Name of the Customer:Varun

Contact Number:7617209448

--------BUS RESERVATION SYSTEM--------

1.Book Tickets

2.Cancel Ticket(s)

3.Check Reservation

4.Seat Availability

5.Cities Available

6.Exit

4

Enter destination city:

Solapur

Seat Availables(0 means Available):

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

--------BUS RESERVATION SYSTEM--------

1.Book Tickets

2.Cancel Ticket(s)

3.Check Reservation

4.Seat Availability

5.Cities Available

6.Exit

4

Enter destination city:

Nashik

Some Error has occurred!

--------BUS RESERVATION SYSTEM--------

1.Book Tickets

2.Cancel Ticket(s)

3.Check Reservation

4.Seat Availability

5.Cities Available

6.Exit

6

Do you want to continue?(Y or N)

Y

Enter username:

admin

Enter password:

admin

1.Add City

2.Delete City

3.Add Bus

4.Cancel Bus

5.Customer List

6.Exit

2

Enter City Name:

Jalgaon

Deleted!

1.Add City

2.Delete City

3.Add Bus

4.Cancel Bus

5.Customer List

6.Exit

3

Enter Destination City:

Nashik

Added!

1.Add City

2.Delete City

3.Add Bus

4.Cancel Bus

5.Customer List

6.Exit

5

------Customer List------

1. Varun

Booking Details

=====Customer Deatils=====

Booking Id:5535220

Destination City:Mumbai

List of Passenger's:

Kaveri

Seat Number:10

1.Add City

2.Delete City

3.Add Bus

4.Cancel Bus

5.Customer List

6.Exit

6

Do you want to continue?(Y or N)

Y

Enter username:

varunkarwa

Enter Password

varunkarwa

Welcome Varun

--------Customer Details--------

Name of the Customer:Varun

Contact Number:7617209448

--------BUS RESERVATION SYSTEM--------

1.Book Tickets

2.Cancel Ticket(s)

3.Check Reservation

4.Seat Availability

5.Cities Available

6.Exit

4

Enter destination city:

Nashik

Seat Availables(0 means Available):

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

0 0 0 0

0 0 0 0

0 0 0 0

--------BUS RESERVATION SYSTEM--------

1.Book Tickets

2.Cancel Ticket(s)

3.Check Reservation

4.Seat Availability

5.Cities Available

6.Exit

5

Mumbai

Ahemdabad

Kolhapur

Nashik

Indore

Nagpur

Solapur

--------BUS RESERVATION SYSTEM--------

1.Book Tickets

2.Cancel Ticket(s)

3.Check Reservation

4.Seat Availability

5.Cities Available

6.Exit

2

Enter booking_id:

5535220

Cancelled

--------BUS RESERVATION SYSTEM--------

1.Book Tickets

2.Cancel Ticket(s)

3.Check Reservation

4.Seat Availability

5.Cities Available

6.Exit

6

Do you want to continue?(Y or N)

N

Process finished with exit code 0

Server Side Output:

ServerSocket awaiting connection...

Connection from Socket[addr=/127.0.0.1,port=52020,localport=4353]

Waiting for Input

Username:varunkarwa

User Present

1

3

6

Connection Live

Waiting for Input

Getting password for admin

Welcome admin

4

1

Bus added

City added

6

Connection Live

Waiting for Input

Username:varunkarwa

User Present

4

4

6

Connection Live

Waiting for Input

Getting password for admin

Welcome admin

2

City deleted

3

Bus added

5

6

Connection Live

Waiting for Input

Username:varunkarwa

User Present

4

5

2

6

Connection Live

Waiting for Input

Getting password for admin

Welcome admin

5

```
6

Process finished with exit code 0
```