```java
package com.nxkundu.server.bo;

import java.io.IOException;
import java.io.Serializable;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.net.UnknownHostException;

/**
 *
 * @author nxkundu
 *
 * @email nxk161830@utdallas.edu
 * @name Nirmallya Kundu
 *
 * Server Class holds all the server information
 * This is a singleton class
 * As we need to create the Server object only once
 * and whenever we need server information we call get the
 * reference to the server object and retrieve the server information
 *
 * Methods:
 *
 * 1> getInstance() - This method returns the server object if already created
 * else, create a object and return it
 *
 * 2> startServer() -  This method starts the server at the defined port
 *
 * 3> connectToServer() -  This method allows the Client to
 * Connect to the server
 *
 */
public class Server implements Serializable{

    /**
     *
     */
```

```
38      private static final long serialVersionUID = 1L;
39
40      private DatagramSocket datagramSocket;
41
42      private int port;
43      private String hostName;
44      private InetAddress inetAddress;
45
46      public static final String SERVER_USERNAME = "SERVER";
47
48      public static Server server;
49
50      /****************************** Constructors ******************************************/
51
52      private Server() throws UnknownHostException {
53          super();
54
55          this.hostName = "localhost";
56          this.inetAddress = InetAddress.getByName(this.hostName);
57          this.port = 8005;
58
59      }
60
61      /****************************** Object Methods ****************************************/
62
63      /**
64       * This method returns the server object if already created
65       * else, create a object and return it
66       * @return
67       * @throws UnknownHostException
68       */
69      public static Server getInstance() throws UnknownHostException {
70
71          if(server == null) {
72              server = new Server();
73          }
74
```

```
75          return server;
76      }
77
78      /**
79       * This method starts the server at the defined port
80       * @throws IOException
81       */
82      public void startServer() throws IOException {
83
84          System.out.println("Server Started @ Port = " + this.port);
85          this.datagramSocket = new DatagramSocket(this.port);
86      }
87
88      /**
89       * This method allows the Client to
90       * Connect to the server
91       * @throws IOException
92       */
93      public void connectToServer() throws IOException {
94
95          if(this.datagramSocket == null) {
96
97              this.datagramSocket = new DatagramSocket();
98          }
99      }
100
101
102     /****************************** Getters and Setters ************************************/
103
104     public DatagramSocket getDatagramSocket() {
105         return datagramSocket;
106     }
107
108     public void setDatagramSocket(DatagramSocket datagramSocket) {
109         this.datagramSocket = datagramSocket;
110     }
111
```

```
112     public int getPort() {
113         return port;
114     }
115 
116     public void setPort(int port) {
117         this.port = port;
118     }
119 
120     public String getHostName() {
121         return hostName;
122     }
123 
124     public void setHostName(String hostName) {
125         this.hostName = hostName;
126     }
127 
128     public InetAddress getInetAddress() {
129         return inetAddress;
130     }
131 
132     public void setInetAddress(InetAddress inetAddress) {
133         this.inetAddress = inetAddress;
134     }
135 }
136 
```