ABSTRACT

The Word guessing game is a user interactive game/puzzle, where the user is required to guess the letters of the word generated. The user is given the option to choose individual letters of the alphabet one at a time and if the guessed letter is present in the word, then the gameinterface displays the letters and the position of the letters in that word. The more correct guesses the user has, the closer he/she understands the random word generated.

The user has to guess the word within a specified error range. That is the user is allowed only 6 incorrect guesses while trying to determine the word. The incorrect guesses need notbe simultaneous. As the number of incorrect guesses progress, the hangman frame will also progress. When the hangman is completely visible the game ends, that is when the user chooses the 8th incorrect letter that is not present in the word, the user has failed to guess the word correctly.

Buttons are used to help the user guess the letters. Already guessed letters cannot be chosen again. When the chosen alphabet is present in the word the button turns green indicating that the alphabet was present in the word. If the button turns red, it indicates thealphabet is not present in the generated word.

INTRODUCTION

1.1 PROPOSED WORK

The Word guessing application project aims to implement Transfer Control Protocol based connection between the client and the server. The application of the game will interact with the server through a socket connection and will provide a Graphical User Interface to the client. Java isused to implement the socket connection and developing the User Interface.

1.2 MODELS OF NETWORKING

There are mainly 2 types of networking Models

1. Client server model
2. Peer to Peer model

1.2.1 Client Server Model

The Client-server model is a distributed application structure that partitions task or workload between the providers of a resource or service, called servers, and service requesters called clients.

In the client-server architecture, when the client computer sends a request for data to the server through the internet, the server accepts the requested process and deliver the data packets requested back to the client.

Clients do not share any of their resources. Examples of Client-Server Model are Email, World Wide Web, etc.

1.2.2 Peer to Peer model

In a peer-to-peer model, all the end-systems are in a non-hierarchical layout, hence centralized management and administrative security cannot be implemented.

Peer-to-peer networks have nodes that both consume and give resources. When a result, asthe number of nodes grows, the peer-to-peer network's resource sharing capacity grows as well.

This differs from client-server networks, in which the server becomes overburdened as the number of nodes grows.

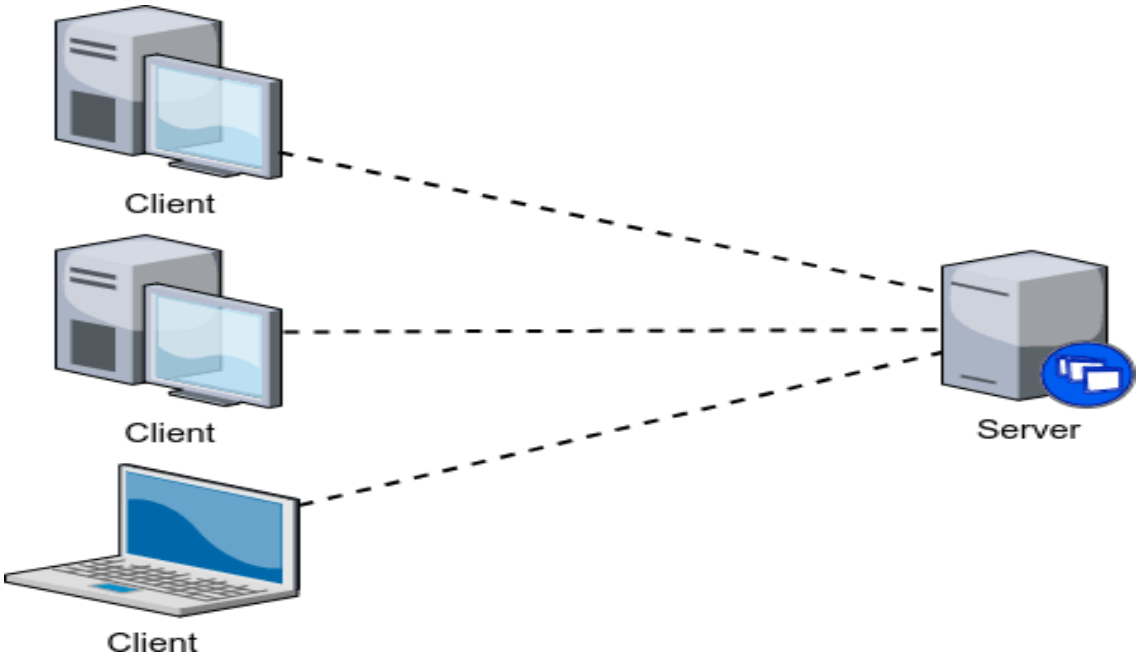


Fig. 1: An illustration of a client-server model.

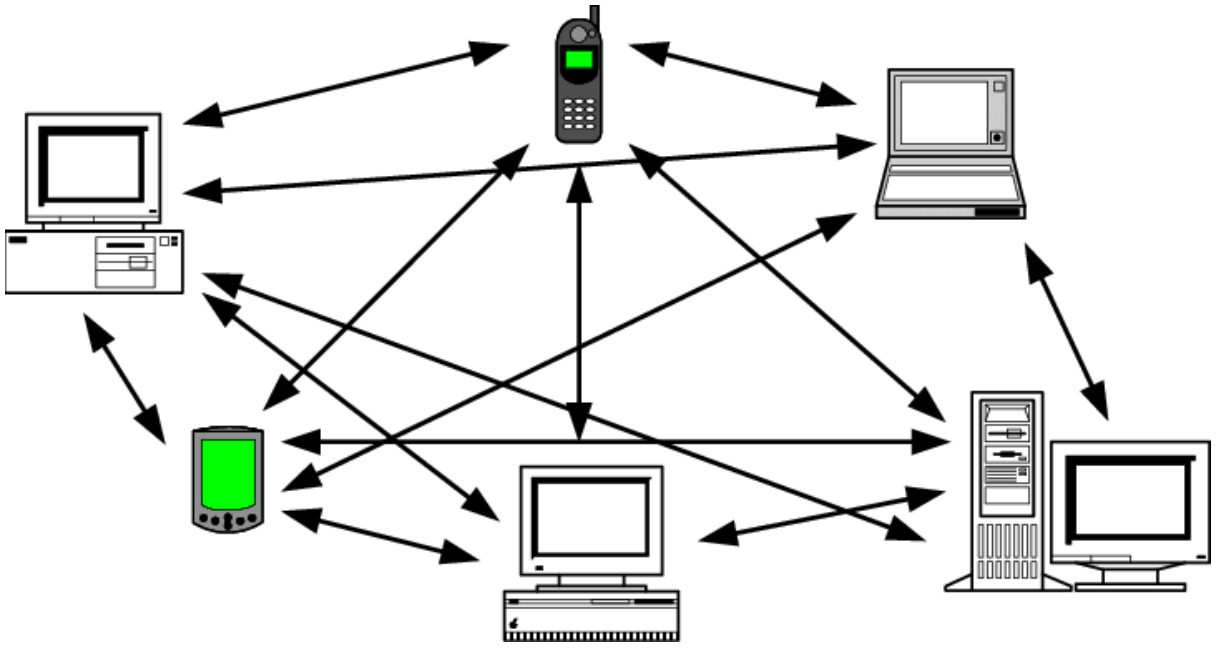


Fig. 2: Illustration of a peer-to-peer network model.

1.3 DIFFERENCE BETWEEN CLIENT SERVER AND PEER TO PEER

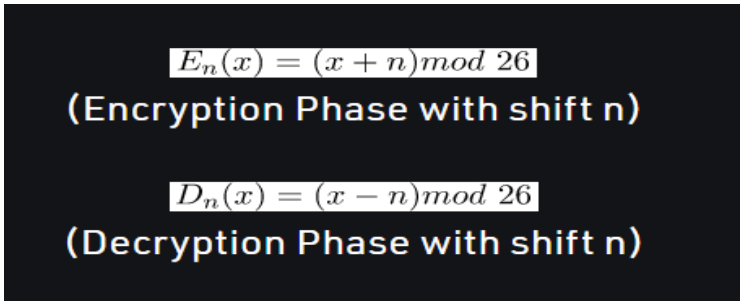
	CLIENT SERVER	PEER TO PEER
	In Client-Server Network, Clients and server are differentiated, Specific server and clients are present.	In Peer-to-Peer Network, Clients and server are not differentiated.
	Client-Server Network focuses on information sharing.	While Peer-to-Peer Network focuses on connectivity.
	In Client-Server Network, Centralized server is used to store the data.	While in Peer-to-Peer Network, Each peer has its own data.
	In Client-Server Network, Server respond the services which is request by Client.	While in Peer-to-Peer Network, Each and every node can do both request and respond for the services.
	Client-Server Network are costlier than Peer-to-Peer Network.	While Peer-to-Peer Network are less costlier than Client-Server Network.
	Client-Server Network are more stable than Peer-to-Peer Network.	While Peer-to-Peer Network are less stable if number of peer is increase.
	Client-Server Network is used for both small and large networks.	While Peer-to-Peer Network is generally suited for small networks with fewer than 10 computers.

1.4 NETWORK SECURITY (CRYPTOGRAPHY)

Cryptography is the study and practice of techniques for secure communication in the presence of third parties called adversaries. It deals with developing and analyzing protocols which prevents malicious third parties from retrieving information being shared between two entities thereby following the various aspects of information security.

In this project, the cryptographic technique used is Caesar Cipher.

The Caesar Cipher technique is one of the earliest and simplest methods of encryption technique. It's simply a type of substitution cipher, i.e., each letter of a given text is replaced by a letter with a fixed number of positions down the alphabet. For example with a shift of 1, A would be replaced by B, B would become C, and so on.
Thus to cipher a given text we need an integer value, known as a shift which indicates the number of positions each letter of the text has been moved down.
The encryption can be represented using modular arithmetic by first transforming the letters into numbers, according to the scheme, A = 0, B = 1,…, Z = 25. Encryption of a letter by a shift n can be described mathematically as.



Ceasar Cipher Code (C++)

```
string encrypt(string text, int s)
{
    string result = "";

    // traverse text
    for (int i = 0; i < text.length(); i++) {
        // apply transformation to each character
        // Encrypt Uppercase letters
        if (isupper(text[i]))
            result += char(int(text[i] + s - 65) % 26 + 65);

        // Encrypt Lowercase letters
        else
            result += char(int(text[i] + s - 97) % 26 + 97);
    }

    // Return the resulting string
    return result;
}
```

1.5 TRANSMISSION CONTROL PROTOCOL

TCP (Transmission Control Protocol) is one of the main protocols of the Internet protocol suite. It lies between the Application and Network Layers which are used in providing reliable delivery services. It is a connection-oriented protocol for communications that helps in the exchange of messages between the different devices over a network.

To make sure that each message reaches its target location intact, the TCP/IP model breaks down the data into small bundles and afterward reassembles the bundles into the original message on the opposite end. Sending the information in little bundles of information makes it simpler to maintain efficiency as opposed to sending everything in one go.

After a particular message is broken down into bundles, these bundles may travel along multiple routes if one route is jammed but the destination remains the same.

FEATURES OF TCP/IP :

- Segment numbering system
- Flow control
- Error control
- Congestion control

ADVANTAGES :

- It is a reliable protocol.
- It provides an error-checking mechanism as well as one for recovery.
- It gives flow control.
- It makes sure that the data reaches the proper destination in the exact order that it was sent.
- Open Protocol, not owned by any organization or individual.
- It assigns an IP address to each computer on the network and a domain name to each site thus making each device site to be distinguishable over the network.

DISADVANTAGES :

- TCP is made for Wide Area Networks, thus its size can become an issue for small networks with low resources.
- TCP runs several layers so it can slow down the speed of the network.
- It is not generic in nature. Meaning, it cannot represent any protocol stack other than the TCP/IP suite. E.g., it cannot work with a Bluetooth connection.
- No modifications since their development around 30 years ago.

1.6 NETWORK SOCKETS

A network socket is a software structure within a network node of a computer network that serves as an endpoint for sending and receiving data across the network. The structure and properties of a socket are defined by an API for the networking architecture. Sockets are created only during the lifetime of a process of an application running in the node.

Because of the standardization of the TCP/IP protocols in the development of the Internet, the term network socket is most commonly used in the context of the Internet protocol suite, and is therefore often also referred to as Internet socket. In this context, a socket is externally identified to other hosts by its socket address, which is the triad of transport protocol, IP address, and port number.

The term socket is also used for the software endpoint of node-internal IPC, which often uses the same API as a network socket.

The IP address of the system and the port it uses make a network socket. For example, a website with an IP address of 127.1.1.1, the socket for the HTTP server for that site is given by 127.1.1:87. In our project, since the same system is used to demonstrate the working of the connection, the IP address or the host name by default is 127.0.0.1. The port number of the server system is specified as 3287 in our implementation.

Types of sockets

Datagram sockets :

It is associated with UDP protocol, and implies that packets will be sent in the datagram format, which is asynchronous in nature. Data can be transmitted across UDP sockets without establishing a secure connection. As a result, this is referred to as "connection-less". It does support bi-directional flow of data. The data delivery is not always guaranteed in this connection.

Stream sockets :

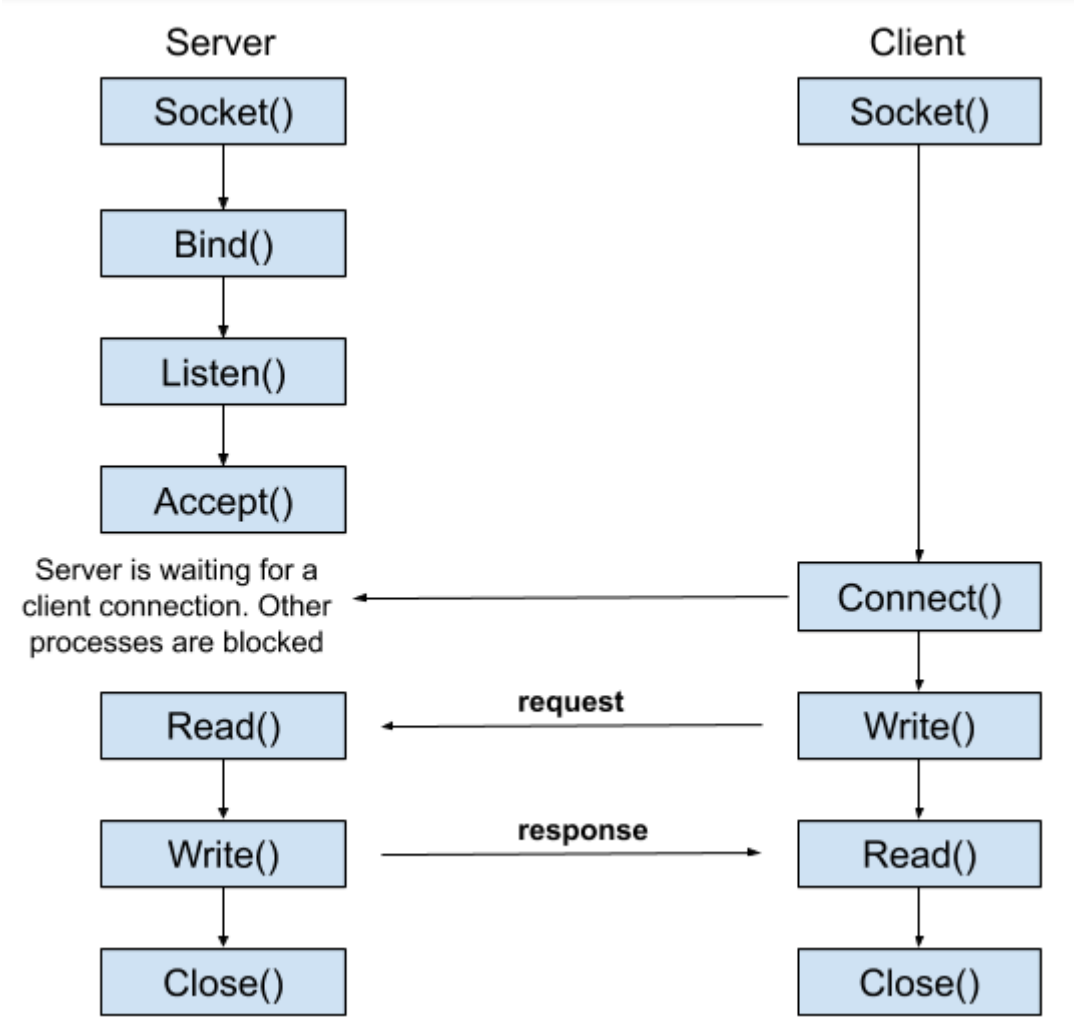
It's related to the TCP protocol and ensures data security during transmission and reception. Before we can transfer data in connection-oriented communication, we need to create a channel. That is a secure connection is established between the two end-systems involved in the socket connection. After the connection has been established, data can be read from and written to these sockets as a byte stream. A stream socket provides bidirectional, reliable, sequenced, and unduplicated flow of data with no record boundaries.

1.7 WORKING

This project aims to establish a TCP based socket connection between the server and its clients, for the application of the Word guessing game. The server code is run first. It creates a new server socket with the port number 3287 and waits patiently for a client to establish a connection. The java socket library helps in the creation and establishment of this connection, as it already possesses the necessary methods and classes to do the same.

In java, a socket connection is established by the following steps;

1. The server code creates a server socket with a specific port number on which the communication transpires.
2. The server uses the accept() method to validate an incoming request to the socket.
3. The client then builds a Socket object with the server name and port number specified. Note that the client needs to have the server's IP address and the port number before hand in order to establish the connection with the server.
4. The client and server can then communicate using I/O streams if the connection is successful.
5. The I/O streams are handled by the client and server socket classes.
6. The client's OutputStream relays data to the server's InputStream, and the server's OutputStream relays data to the client's InputStream.



1.8 CLASSES USED IN SOURCE CODE

1. Chat_server - To implement server-side application. To create a socket with the specified port number and listens to establish a connection. It generates the random word that is required to be sent to the client side. It does so after converting it to a cipher text using a pre-defined key.

2. Server – It instantiates a chat_server class object and runs it.

3. Main – It creates the client-side socket and establishes a connection with the available server. The server IP address and the specific port number is passed to the client. It requests the necessary data from the server side by sending the required request message.

4. Guesspanel – It is used to construct the GUI required by the user to interact with, in order to avail the chance to guess the random word generated by the server.

5. Hangmanpanel – it is used to construct the GUI required by the user to efficiently comprehend the status of the game the user is playing, whether he is guessing the correct word or not.

SOURCE CODE

Chat_server.java

```
import java.io.EOFException;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.net.ServerSocket;
import java.net.Socket;

public class chat_server {
   private ObjectOutputStream output;
   private ObjectInputStream input;
   private Socket connection; // for client using TCP communication
   private ServerSocket server; // listen for clients
   private int totalClients = 50;
   private int port = 3287;

   private static final String[] words = {
         "SENTENCE","LINE","MAN","BOY","FARM","LETTER","POINT","MOTHER",
"AMERICA","FATHER","TREE",

"CITY","EXAMPLE","PAPER","SECOND","DIFFERENT","HARD","IMPORTANT",
"WHITE",
         "TOGETHER","ALWAYS","BELOW","NEVER",

"STRUCTURE","HONEST","CURTAIN","CLAY","FACTORY","INVASION","FREE",
"COLOURFUL","LEADERSHIP",

"PROPOSAL","COMPUTER","MONITOR","COW","RAIN","BOOK","BOTH","YOUNG",
"BOLD",

"ABOVE","BELOW","SCHOOL","COLLEGE","OFFICE","INTERNSHIP","NOTEBOOK",
         "EXAM","SEMESTER","SUMMER","WINTER","HOLIDAYS","VACATION"
         ,"SOPHOMORE","FRESHER","TECHNOLOGY","MECHANICS","PHYSICS",
         "BIOLOGY","ENGLISH","SCIENCE","MATHEMATICS","ENGINEERING",
         "MEDICINE","LAW","RESEARCH","FEST","CULTURE","ORGANISATION",
         "INTERNAL","EXTERNAL","PRATICALS","LABS","RECORD","MUSIC",
         "ENTERTAINMENT","ICE CREAM","HARRY POTTER","CORAL REEF",
         "ELON MUSK","ABDUL KALAM",
"COMPUTER NETWORK","ROUTER","CLASS","FREE HOUR","LUNCH","HOSTEL"
   };

   public void startRunning() {
      try {
         server = new ServerSocket(port, totalClients);
         while (true) {
            try {
               connection = server.accept(); // establishes connection and waits for the client
```

```
output = new ObjectOutputStream(connection.getOutputStream());
                output.flush();
                input = new ObjectInputStream(connection.getInputStream());

                whileChatting();
            } catch (EOFException eofException) {
            }

        }
      } catch (IOException ioException) {
        ioException.printStackTrace();
      }
   }

   private void whileChatting() throws IOException {
      String message_r = "", message_s = "";
      String request1 = "new";

      do {
        try {
          message_r = (String) input.readObject();
          if (message_r.equals(request1)) {
            message_s = words[(int) (Math.random() * words.length)];
            StringBuffer str = new StringBuffer(message_s);
            for (int i = 0; i < message_s.length(); i++) {
              if (Character.isUpperCase(message_s.charAt(i))) {
                char ch = (char) (((int) message_s.charAt(i) +  4 - 65) % 26 + 65);
                str.append(ch);
              } else {
                char ch = (char) (((int) message_s.charAt(i) + 4 - 97) % 26 + 97);
                str.append(ch);
              }
            }
            // str.setCharAt(i, (char) (message_s.charAt(i) + key[(j++) % 4]));
            message_s = str.toString();
                        //String m = "";
                        //for(int i=0;i<message_s.length()/4;i++){
                        //        m = m + message_s.charAt(i);
                        //}
                        //message_s = m;
            output.writeObject(message_s);
          }
        } catch (ClassNotFoundException classNotFoundException) {
        }
      } while (true);

   }
}
```

Main.java

```
import javax.swing.*;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.net.Socket;
import java.io.IOException;
import java.net.InetAddress;
import javax.swing.JOptionPane;

public class Main {
   private static String word;
   private static ObjectOutputStream output;
   private static ObjectInputStream input;
   // private String message = "";
   private static String serverIP = "127.0.0.1";
   private static Socket connection; // for client using TCP communication
   private static int port = 3287;

   public static void main(String[] args) {
      {
         try {
            connection = new Socket(InetAddress.getByName(serverIP), port);
            output = new ObjectOutputStream(connection.getOutputStream());
            output.flush();
            input = new ObjectInputStream(connection.getInputStream());
         } catch (IOException ioException) {
            ioException.printStackTrace();
         }
      }
      int dialogButton = JOptionPane.YES_NO_OPTION;
      int dialogResult = JOptionPane.showConfirmDialog(null, "Would you like to pick a
random server generated word?", "Word Guessing application",
            dialogButton);
      if (dialogResult == 0) {
         word = getRandomWord();
      } else {
         JPasswordField wordField = new JPasswordField();
         int okOrCancel = JOptionPane.showConfirmDialog(null, wordField, "Word",
               JOptionPane.OK_CANCEL_OPTION, JOptionPane.PLAIN_MESSAGE);
         if (okOrCancel == JOptionPane.OK_OPTION) {
            word = String.valueOf(wordField.getPassword()).trim().toUpperCase();
         }
      }
      System.err.println("Your word is " + word);
      new HangmanFrame();
   }
```

```
public static String getWord() {
        return word;
    }



 public static String getRandomWord() {
        String word_n = "", sm = "new";
        try {
            output.writeObject(sm);
            output.flush();
            word_n = (String) input.readObject();
            StringBuffer str = new StringBuffer(word_n);
            for (int i = 0; i < word_n.length(); i++) {
                if (Character.isUpperCase(word_n.charAt(i))) {
                    char ch = (char) (((int) word_n.charAt(i) + 4 - 65) % 26 + 65);
                    str.append(ch);
                } else {
                    char ch = (char) (((int) word_n.charAt(i) + 4 - 97) % 26 + 97);
                    str.append(ch);
                }
            }
            // str.setCharAt(i, (char) (word_n.charAt(i) - key[(j++) % 4]));
            word_n = str.toString();
                        String m = "";
                                for(int i=0;i<word_n.length()/4;i++){
                                        m = m + word_n.charAt(i);
                                }
                                word_n = m;
        } catch (Exception E) {
        }
        return word_n;
    }

    public static void setWord(String word) {
        Main.word = word;
    }
}
```

Server.java

```
public class Server {
    public static void main(String[] args) {
        chat_server myServer = new chat_server();
        myServer.startRunning();
    }
}
```

GuessPanel.java

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class GuessPanel extends JPanel implements ActionListener {
   private static final JButton[] letters = new JButton[26];

   GuessPanel() {
      int j = 0;
      for (char i = 'A'; i <= 'Z'; i++) {
         letters[j] = new JButton(String.valueOf(i));
         letters[j].setFocusable(false);
         letters[j].setFont(new Font("Comic Sans",Font.BOLD, 20));
         letters[j].addActionListener(this);
         this.add(letters[j]);
         j++;
      }
      this.setBackground(new Color(0, 84, 209));
      this.setOpaque(true);
   }

   public static void replaceAsterisks(char c) {
      char[] temp = HangmanPanel.getWordWithHiddenLetters().toCharArray();
      for (int i = 0; i < Main.getWord().length(); i++) {
         if (Main.getWord().charAt(i) == c) {
            temp[i] = c;
         }
      }
      HangmanPanel.setWordWithHiddenLetters(String.valueOf(temp));
   }

   @Override
   public void actionPerformed(ActionEvent e) {
      if (!HangmanPanel.isGameOver()) {
         JButton pressedButton = (JButton) e.getSource();
         if (Main.getWord().contains(pressedButton.getText())) {
            replaceAsterisks(pressedButton.getText().charAt(0));
            pressedButton.setBackground(Color.GREEN);
            pressedButton.setEnabled(false);
            if (HangmanPanel.getWordWithHiddenLetters().equals(Main.getWord())) {
               HangmanPanel.setGameOver(true);
               HangmanPanel.victory();
               HangmanPanel.resetSetup();
            }
         }
```

```
else {
            pressedButton.setBackground(Color.RED);
            pressedButton.setEnabled(false);



HangmanPanel.increaseMistakesCount();
            if (HangmanPanel.getMistakesCount() == 7) {
               HangmanPanel.guessfail();
               HangmanPanel.setWordWithHiddenLetters(Main.getWord());
               HangmanPanel.setGameOver(true);
               HangmanPanel.resetSetup();
            }
         }
      }
   }

   public static void reset() {
      for (int i = 0; i < letters.length; i++) {
         letters[i].setEnabled(true);
         letters[i].setBackground(new JButton().getBackground());
      }
   }
}
```

HangmanFrame.java

```
import javax.swing.*;

public class HangmanFrame extends JFrame {
   HangmanFrame() {
      this.setSize(1100, 800);
      this.setTitle("Word guessing application");
      //this.setIconImage(new ImageIcon("rope.png").getImage());
      this.setLayout(null);
      this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      HangmanPanel hangmanPanel = new HangmanPanel();
      hangmanPanel.setBounds(0, 0, 700, 800);
      GuessPanel guessPanel = new GuessPanel();
      guessPanel.setBounds(700, 0, 400, 800);
      this.add(hangmanPanel);
      this.add(guessPanel);
      this.setResizable(false);
      this.setVisible(true);
   }
}
```

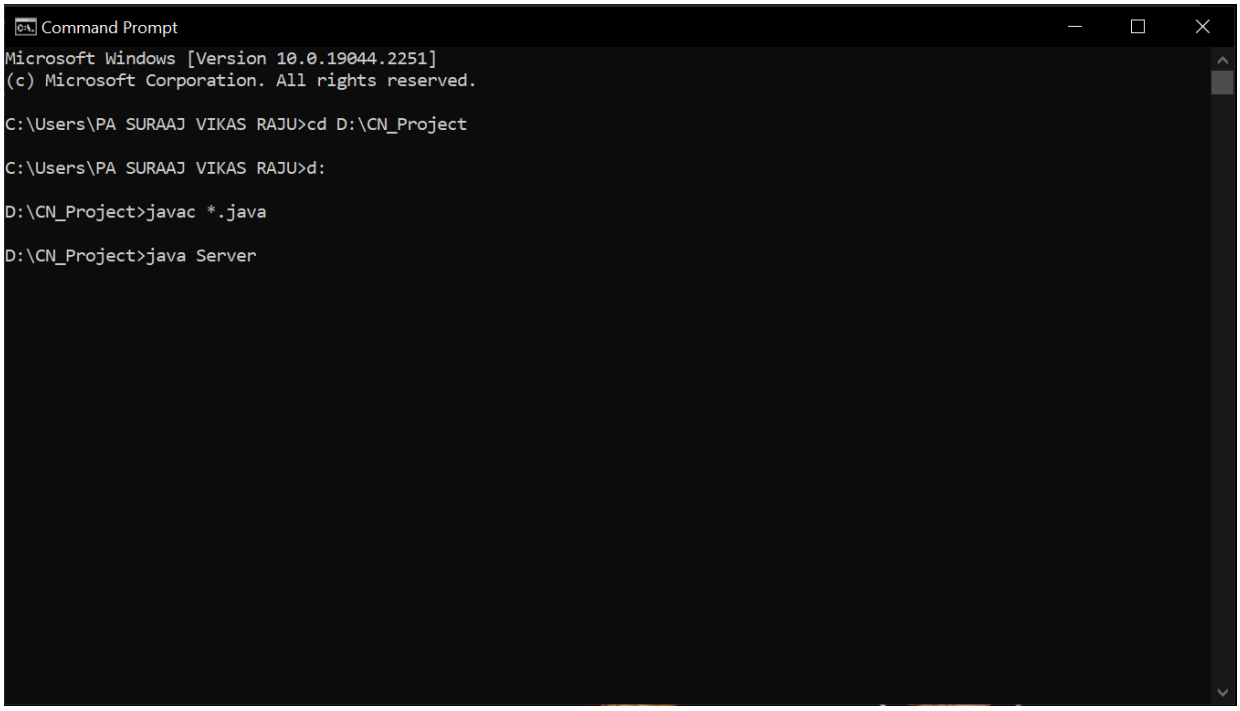
HangmanPanel.java

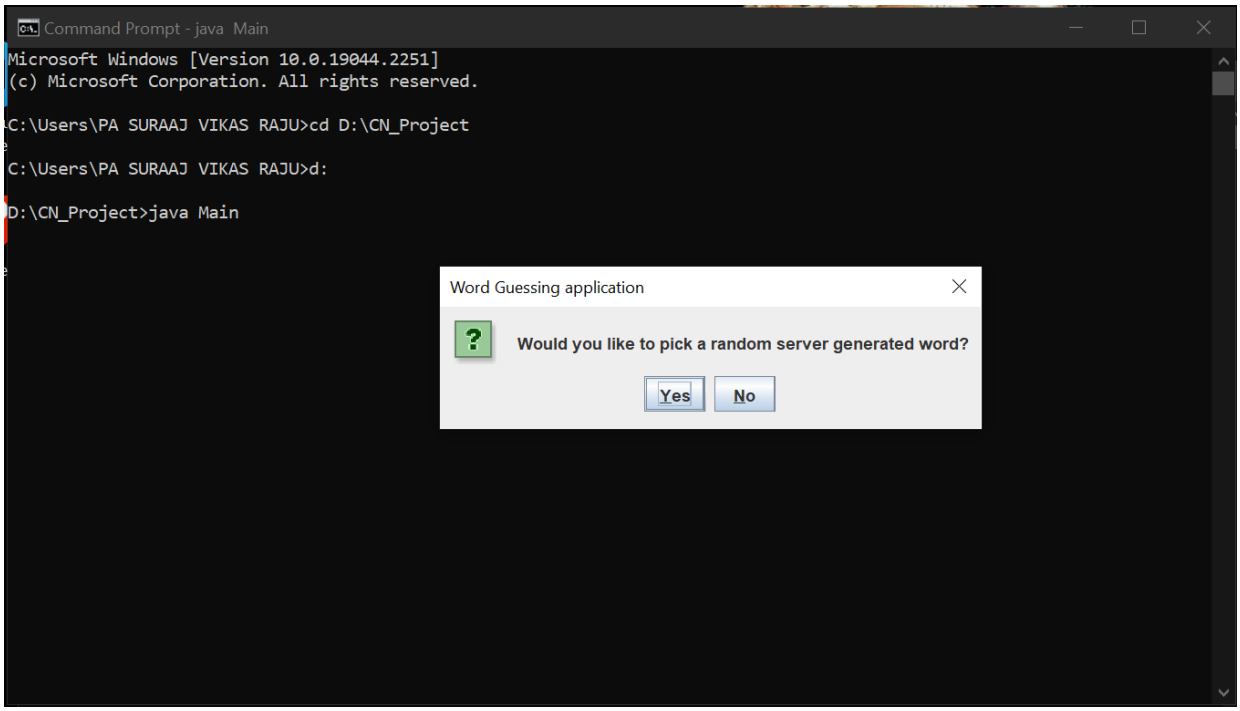
```
import javax.swing.*;
import java.awt.*;

public class HangmanPanel extends JPanel {
   private static final ImageIcon[] hangmanImages = {
         new ImageIcon("h1.png"),
         new ImageIcon("h2.png"),
         new ImageIcon("h3.png"),
         new ImageIcon("h4.png"),
         new ImageIcon("h5.png"),
         new ImageIcon("h6.png"),
         new ImageIcon("h7.png"),
         new ImageIcon("h8.png")
   };
   private static int mistakesCount = 0;
   private static String wordWithHiddenLetters;
   private static JLabel jLabel;
   private static boolean gameOver;

HangmanPanel() {
gameOver = false;
this.setBackground(new Color(201, 181, 84));
this.setOpaque(true);
StringBuilder sb = new StringBuilder();
for (int i = 0; i < Main.getWord().length(); i++) {
sb.append(Main.getWord().charAt(i) == ' ' ? ' ' : '*');
}
wordWithHiddenLetters = sb.toString();
jLabel = new JLabel(wordWithHiddenLetters);
jLabel.setFont(new Font("Comic Sans", Font.BOLD, 60));
jLabel.setBackground(new Color(186, 230, 255));
jLabel.setOpaque(true);
jLabel.setVerticalTextPosition(JLabel.TOP);
jLabel.setHorizontalTextPosition(JLabel.CENTER);
this.add(jLabel);
}

   public static String getWordWithHiddenLetters() {
      return wordWithHiddenLetters;
   }

   public static void setWordWithHiddenLetters(String wordWithHiddenLetters) {
      HangmanPanel.wordWithHiddenLetters = wordWithHiddenLetters;
      jLabel.setText(wordWithHiddenLetters);
   }

  public static void increaseMistakesCount() {
      mistakesCount++;
      jLabel.setIcon(hangmanImages[mistakesCount - 1]);
   }
```

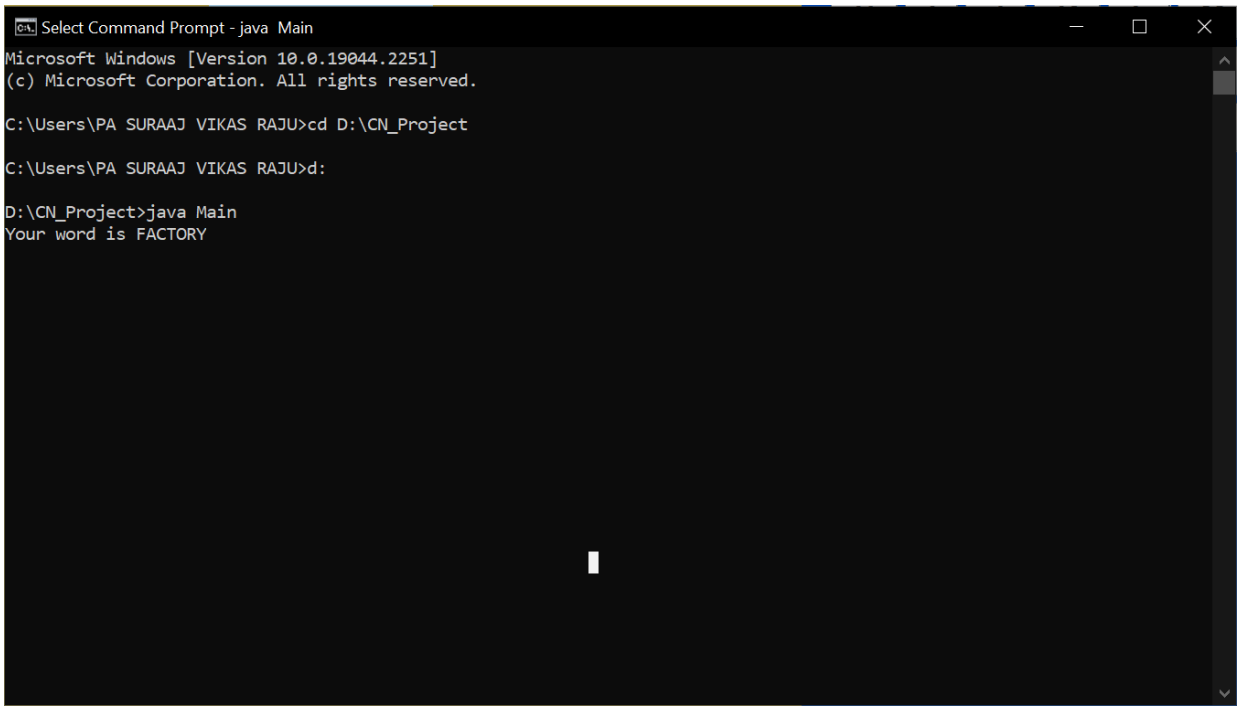
```
    public static int getMistakesCount() {
        return mistakesCount;
    }

    public static boolean isGameOver() {
        return gameOver;
    }

    public static void setGameOver(boolean gameOver) {
        HangmanPanel.gameOver = gameOver;
    }

    public static void victory() {
        jLabel.setIcon(new ImageIcon("good_job.jpg"));
    }

    public static void guessfail() {
        jLabel.setIcon(new ImageIcon("bg.png"));
    }

    public static void reset() {
        gameOver = false;
        mistakesCount = 0;
        jLabel.setIcon(null);
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < Main.getWord().length(); i++) {
            sb.append(Main.getWord().charAt(i) == ' ' ? ' ' : '*');
        }
        wordWithHiddenLetters = sb.toString();
        jLabel.setText(wordWithHiddenLetters);
    }

    public static void resetSetup() {
        int dialogButton = JOptionPane.YES_NO_OPTION;
        int dialogResult = JOptionPane.showConfirmDialog(null, "Would you like to start again
?", "Word Guessing application",
                dialogButton);
        if (dialogResult == 0) {
            dialogButton = JOptionPane.YES_NO_OPTION;
            dialogResult = JOptionPane.showConfirmDialog(null, "Would you like to pick a
random server generated word?", "Word Guessing application",
                    dialogButton);

if (dialogResult == 0) {
                Main.setWord(Main.getRandomWord());
            }
```

```
else {
            JPasswordField wordField = new JPasswordField();
            int okOrCancel = JOptionPane.showConfirmDialog(null, wordField, "Word",
                  JOptionPane.OK_CANCEL_OPTION, JOptionPane.PLAIN_MESSAGE);
            if (okOrCancel == JOptionPane.OK_OPTION) {
               Main.setWord(String.valueOf(wordField.getPassword()).trim().toUpperCase());
            }
         }
         GuessPanel.reset();
         reset();

         System.err.println("Your word is " + Main.getWord());
      }
   }
}
```

OUTPUT

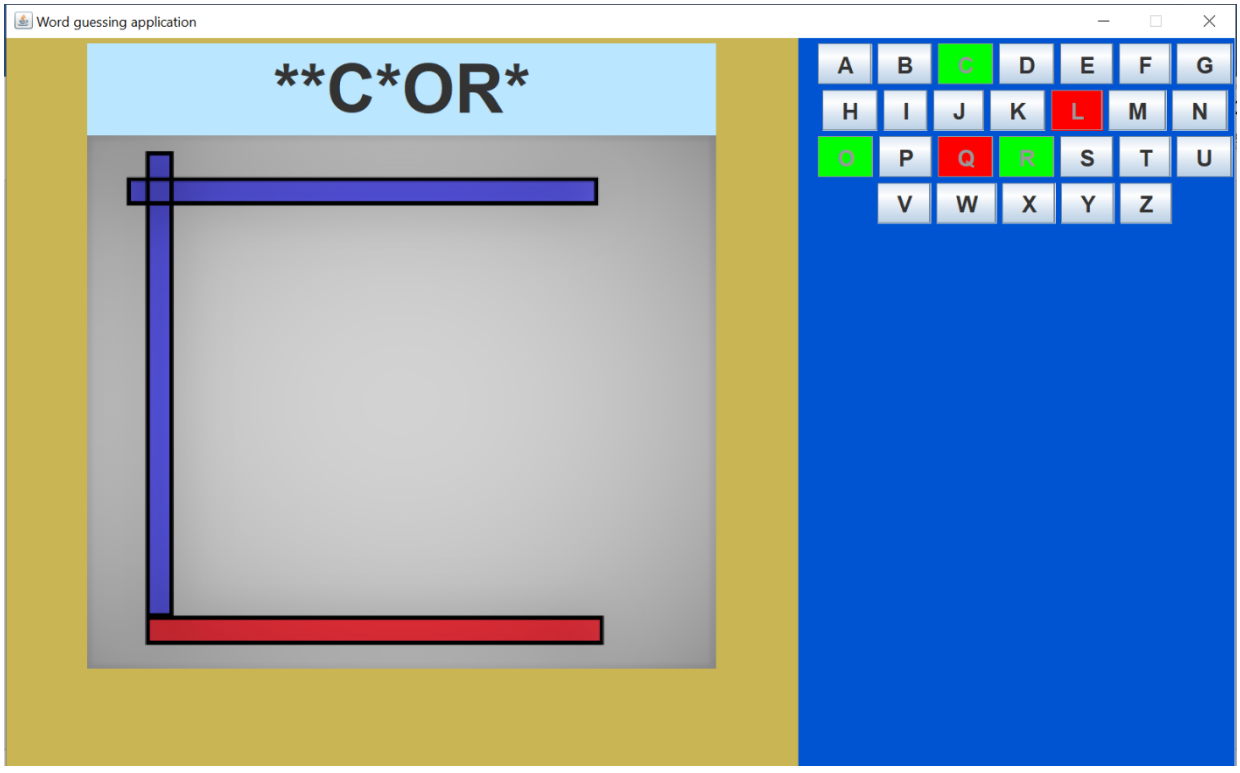

Command Prompt
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.
C:\Users\PA SURAAJ VIKAS RAJU>cd D:\CN_Project
C:\Users\PA SURAAJ VIKAS RAJU>d:
D:\CN_Project>javac *.java
D:\CN_Project>java Server

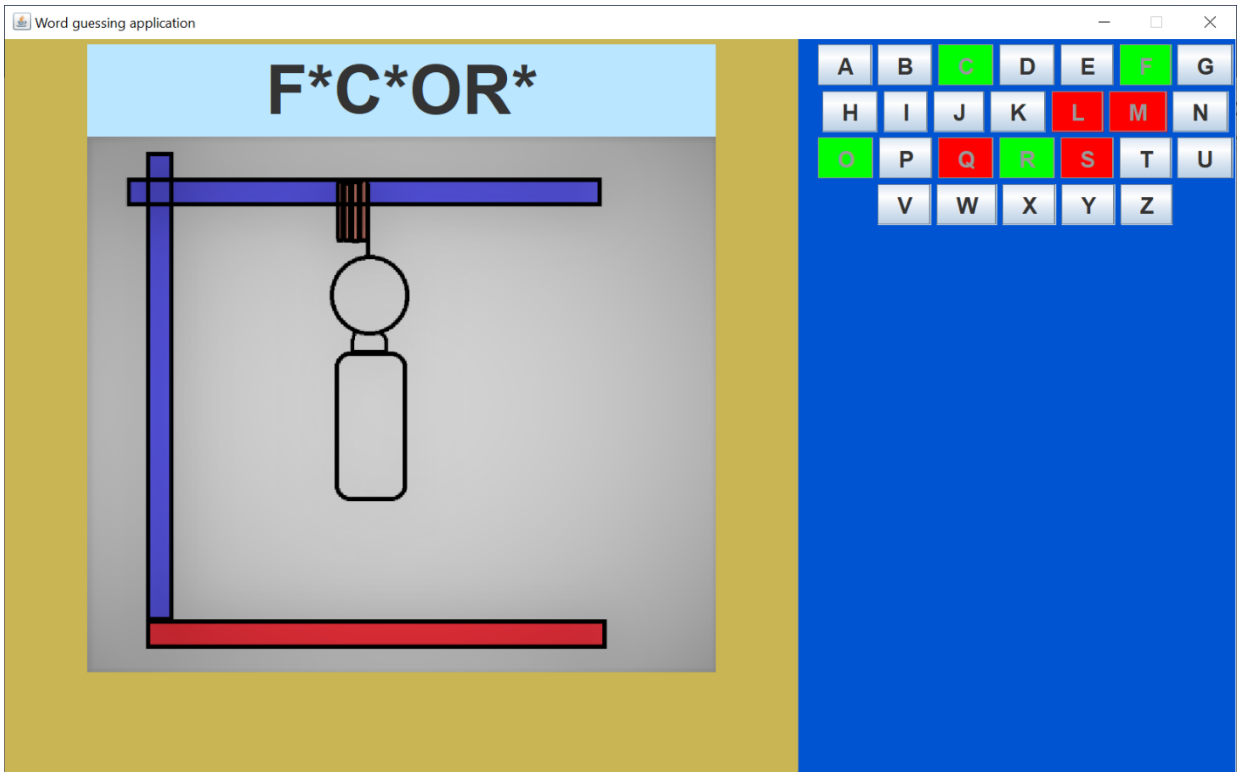

Command Prompt - java Main
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.
C:\Users\PA SURAAJ VIKAS RAJU>cd D:\CN_Project
C:\Users\PA SURAAJ VIKAS RAJU>d:
D:\CN_Project>java Main
Word Guessing application
Would you like to pick a random server generated word?
Yes
No

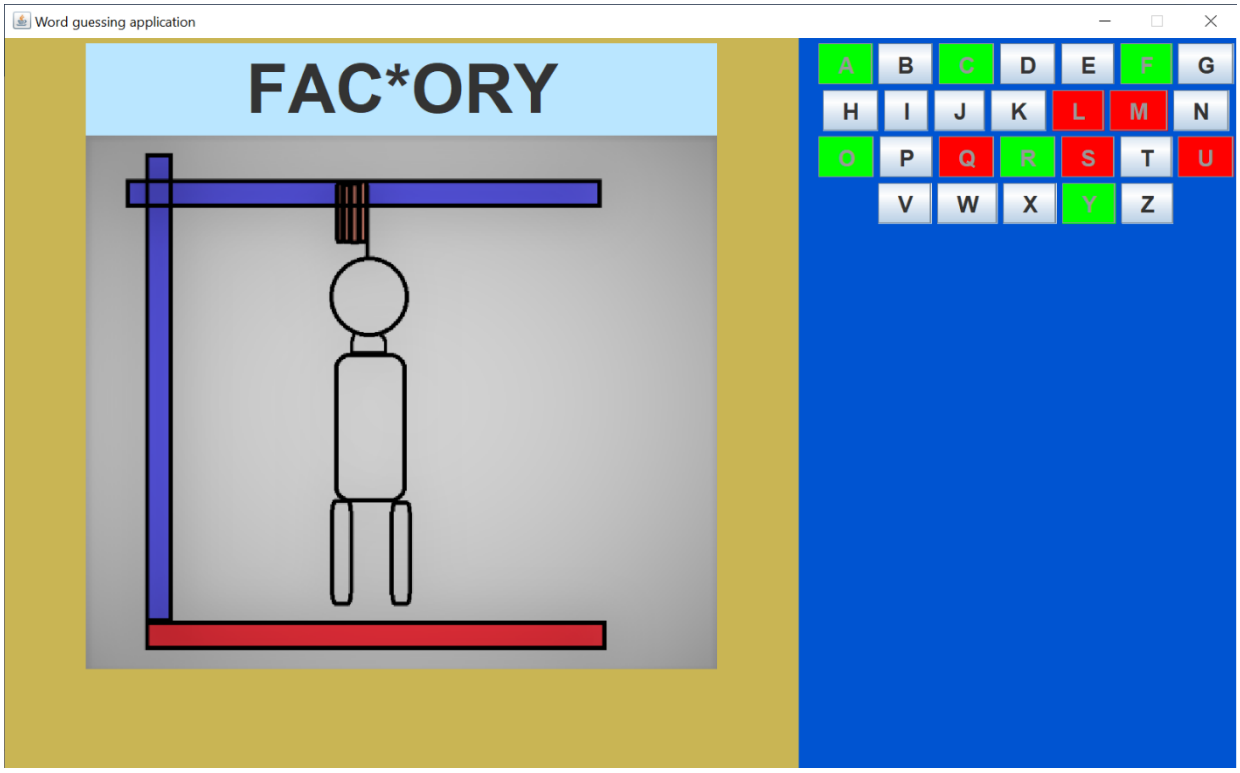

Select Command Prompt - java Main
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.
C:\Users\PA SURAAJ VIKAS RAJU>cd D:\CN_Project
C:\Users\PA SURAAJ VIKAS RAJU>d:
D:\CN_Project>java Main
Your word is FACTORY


Word guessing application
****
A B C D E F G
H I J K L M N
O P Q R S T U
V W X Y Z

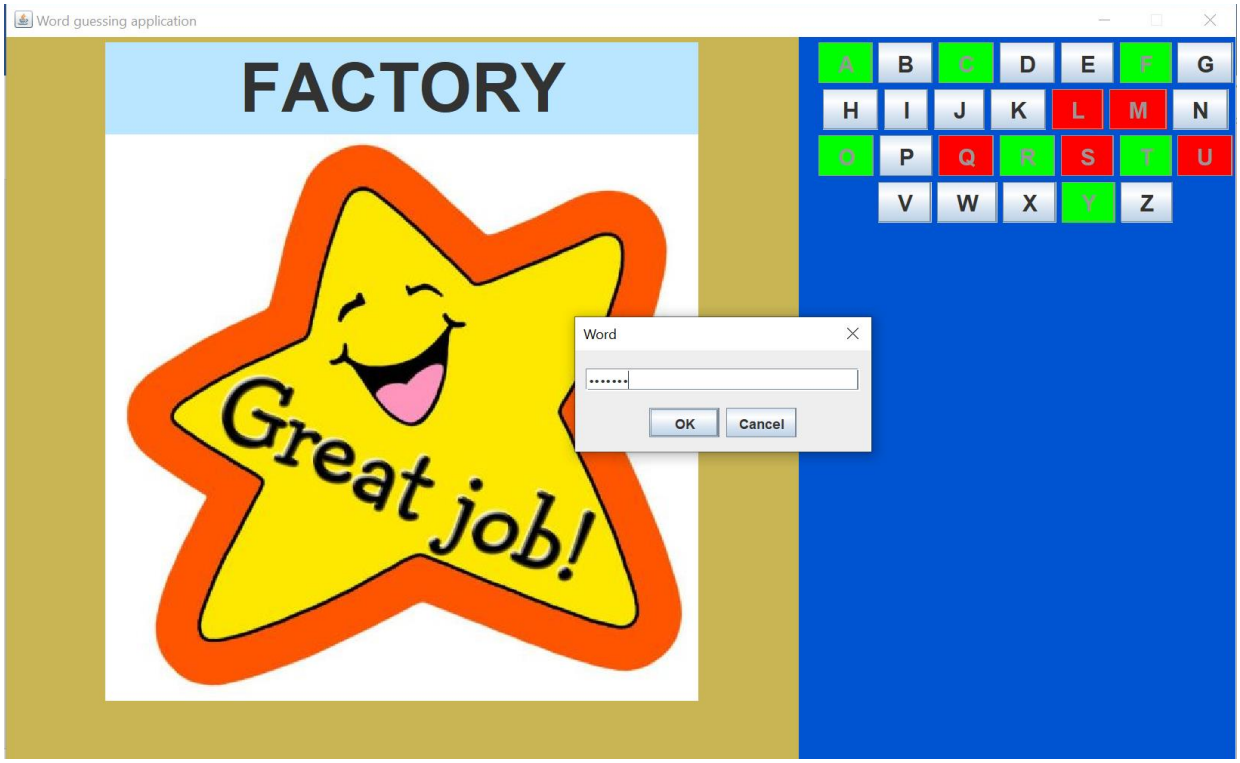

Word guessing application
**C*OR*
A B C D E F G
H I J K L M N
O P Q R S T U
V W X Y Z

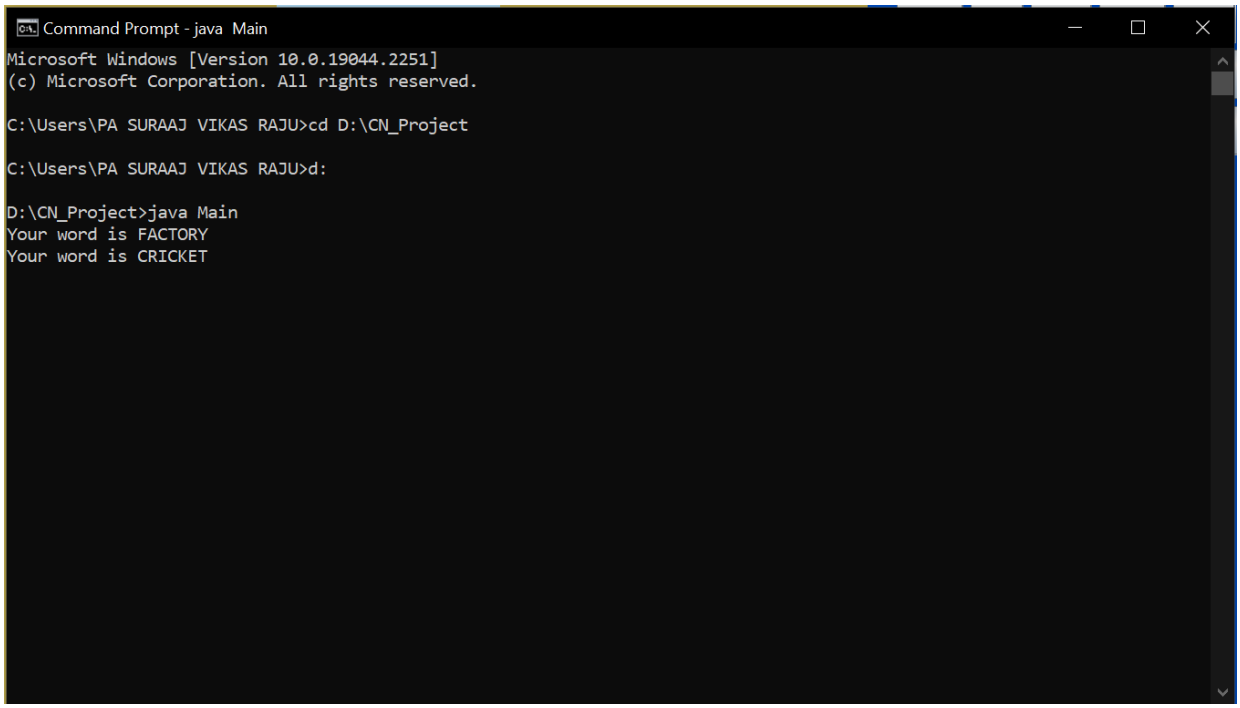

Word guessing application
**C*OR*
A B C D E F G
H I J K L M N
O P Q R S T U
V W X Y Z

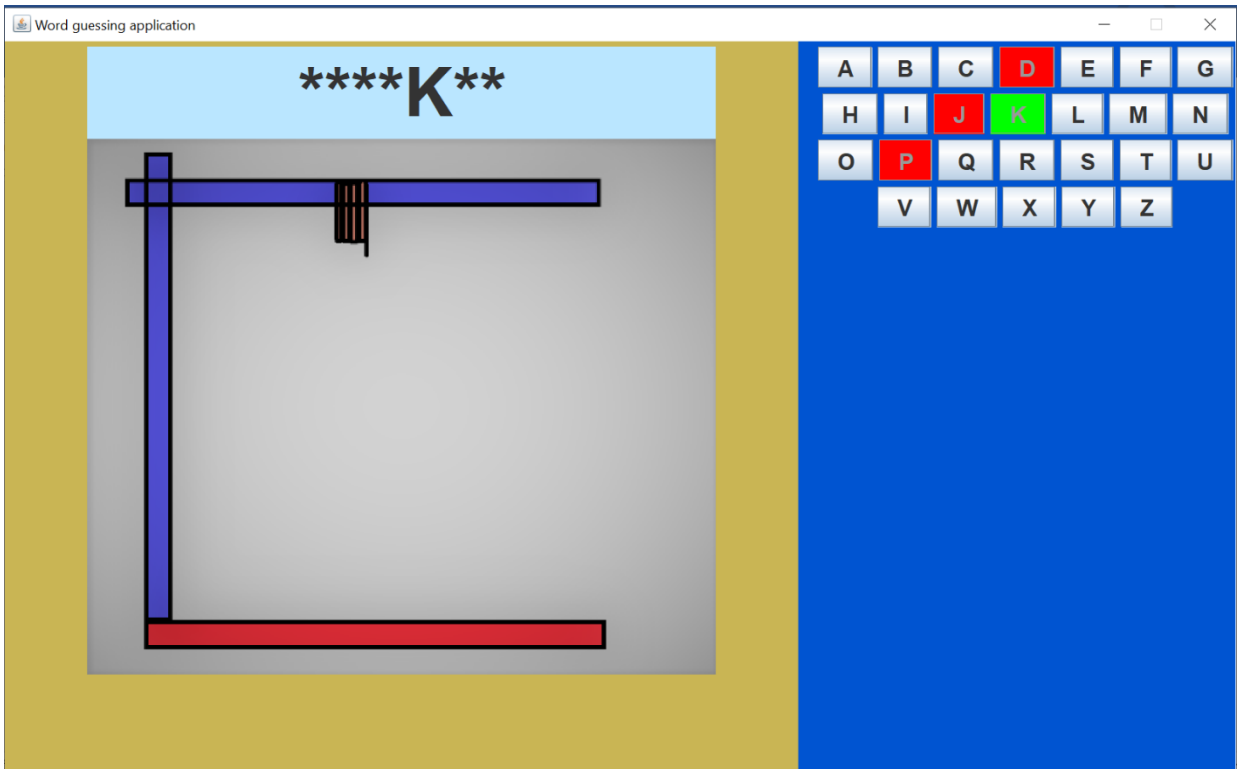

Word guessing application
F*C*OR*
A B C D E F G
H I J K L M N
O P Q R S T U
V W X Y Z

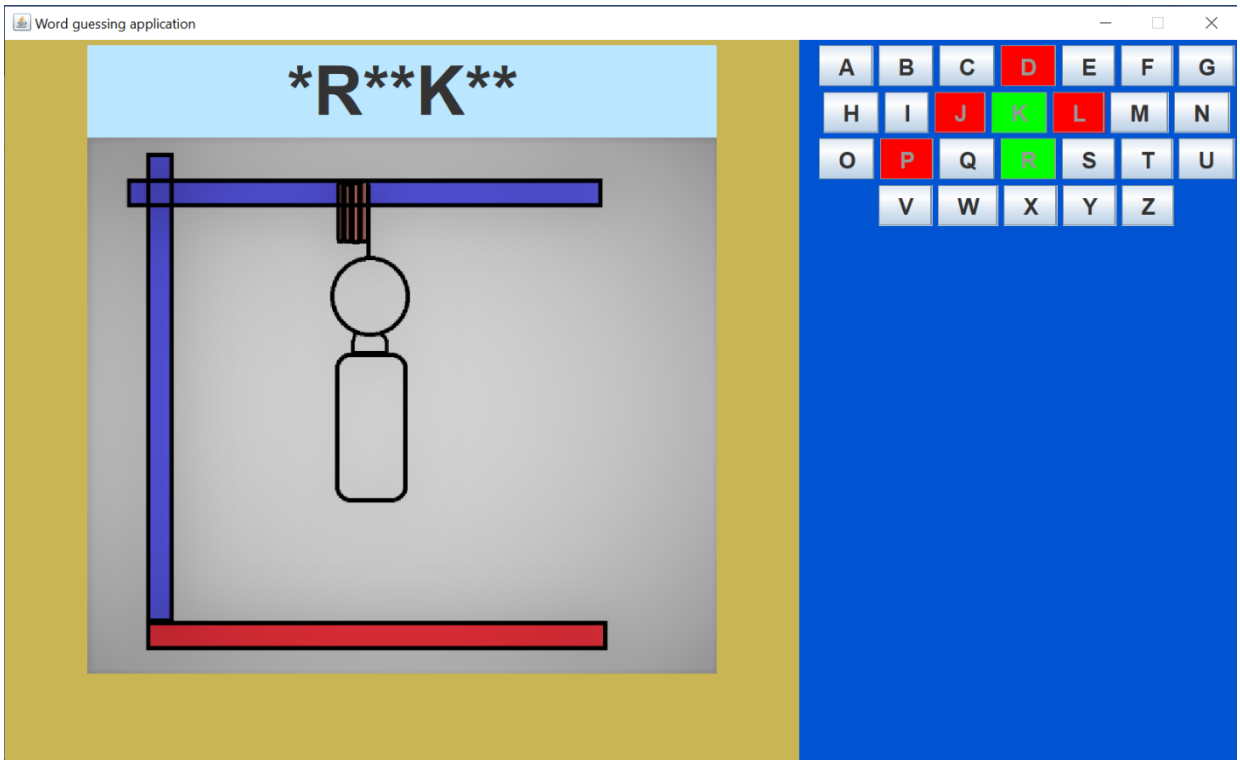

Word guessing application
FAC*ORY
A B C D E F G
H I J K L M N
O P Q R S T U
V W X Y Z

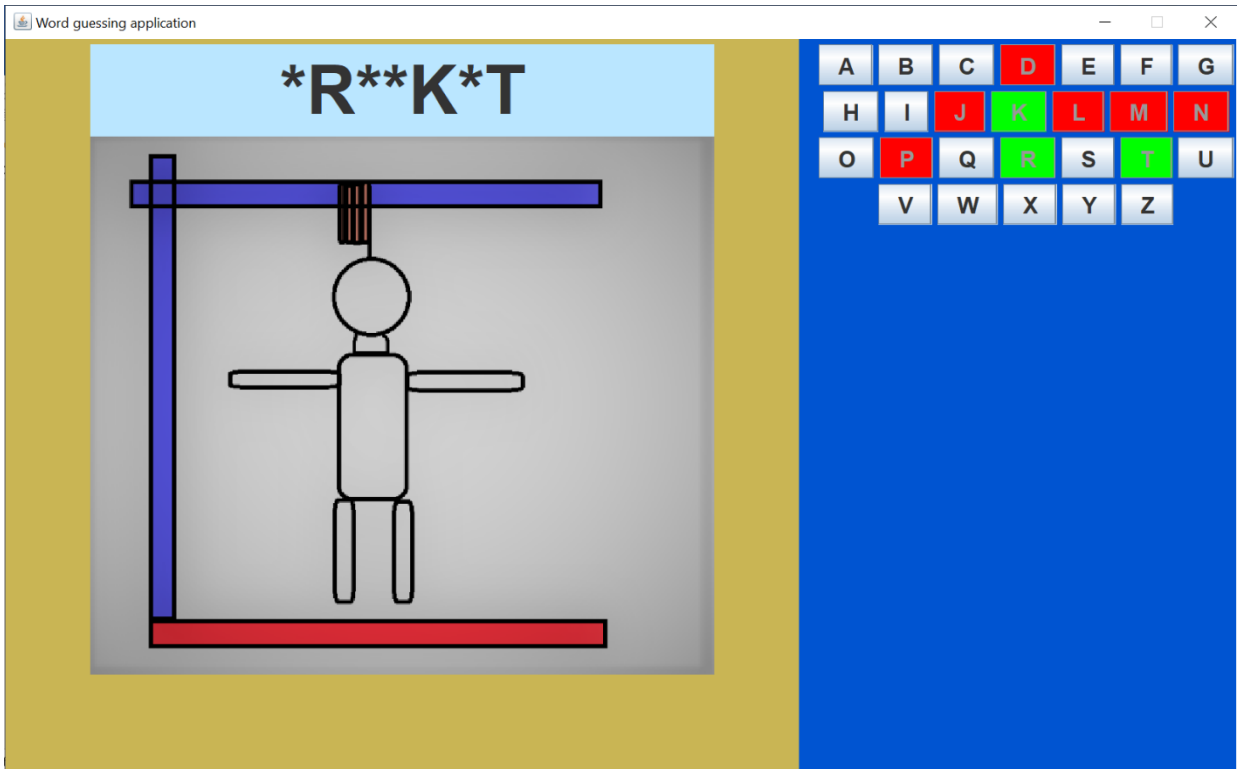

Word guessing application
FACTORY
Great job!
A B C D E F G
H I J K L M N
O P Q R S T U
V W X Y Z
Word Guessing application
Would you like to start again ?
Yes
No


Word guessing application
FACTORY
Great job!
A B C D E F G
H I J K L M N
O P Q R S T U
V W X Y Z
Word
OK
Cancel

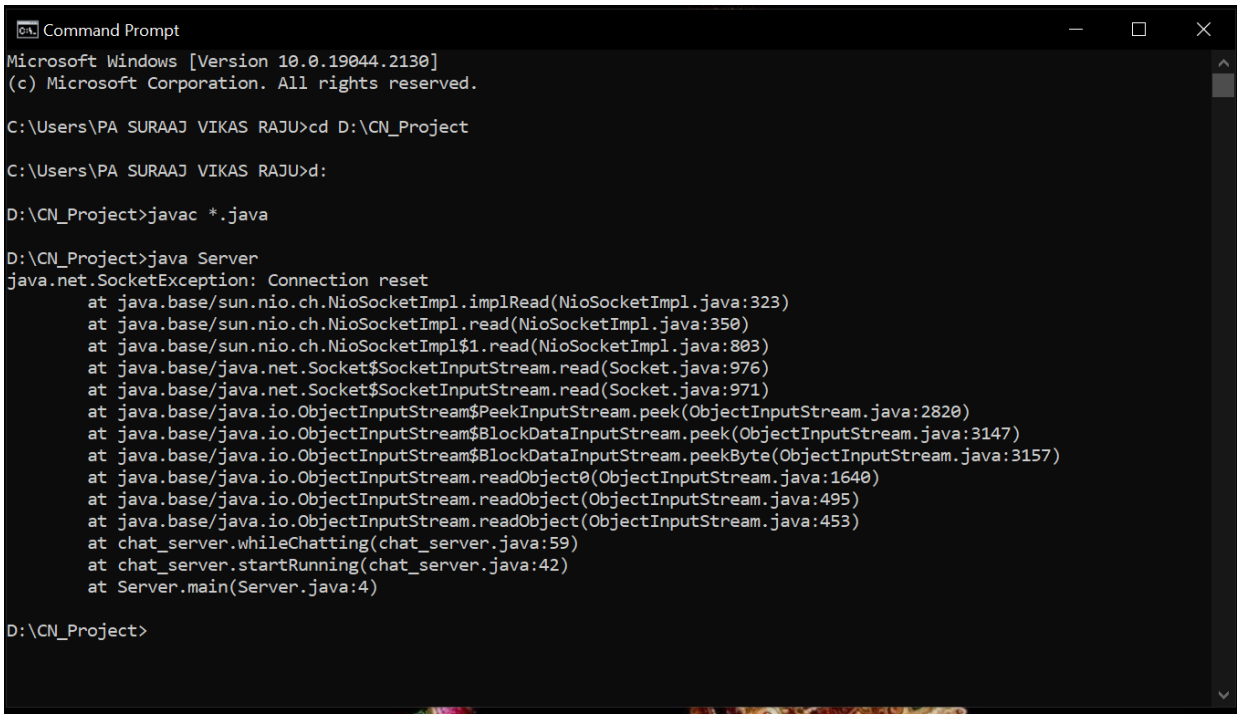

Command Prompt - java Main
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.
C:\Users\PA SURAAJ VIKAS RAJU>cd D:\CN_Project
C:\Users\PA SURAAJ VIKAS RAJU>d:
D:\CN_Project>java Main
Your word is FACTORY
Your word is CRICKET


Word guessing application
****K**
A B C D E F G
H I J K L M N
O P Q R S T U
V W X Y Z


Word guessing application
*R**K**
A B C D E F G
H I J K L M N
O P Q R S T U
V W X Y Z


Word guessing application
*R**K*T
A B C D E F G
H I J K L M N
O P Q R S T U
V W X Y Z


Word guessing application
CRICKET
SORR
BETTER LUCK
NEXT TIME
A B C D E F G
H I J K L M N
O P Q R S T U
V W X Y Z
Word Guessing application
Would you like to start again ?
Yes
No


Command Prompt
Microsoft Windows [Version 10.0.19044.2130]
(c) Microsoft Corporation. All rights reserved.
C:\Users\PA SURAAJ VIKAS RAJU>cd D:\CN_Project
C:\Users\PA SURAAJ VIKAS RAJU>d:
D:\CN_Project>javac *.java
D:\CN_Project>java Server
java.net.SocketException: Connection reset
at java.base/sun.nio.ch.NioSocketImpl.implRead(NioSocketImpl.java:323)
at java.base/sun.nio.ch.NioSocketImpl.read(NioSocketImpl.java:350)
at java.base/sun.nio.ch.NioSocketImpl$1.read(NioSocketImpl.java:803)
at java.base/java.net.Socket$SocketInputStream.read(Socket.java:976)
at java.base/java.net.Socket$SocketInputStream.read(Socket.java:971)
at java.base/java.io.ObjectInputStream$PeekInputStream.peek(ObjectInputStream.java:2820)
at java.base/java.io.ObjectInputStream$BlockDataInputStream.peek(ObjectInputStream.java:3147)
at java.base/java.io.ObjectInputStream$BlockDataInputStream.peekByte(ObjectInputStream.java:3157)
at java.base/java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1640)
at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:495)
at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:453)
at chat_server.whileChatting(chat_server.java:59)
at chat_server.startRunning(chat_server.java:42)
at Server.main(Server.java:4)
D:\CN_Project>

CONCLUSION

This project illustrates the concepts and working features of TCP based socket connection.
It proposes and implements a cipher text method (Ceasar cipher) to securely transfer data from server to client.
This project shows us how java socket programming is used to establish a connection between the clients and the server. The GUI of the java programming helps the user interact with the application effortlessly.
Further developments can be made to the code. The words can be categorized, into several directories like fruits, vegetables, pronouns, verbs and so on.
Based on which the user can request a word to guess. A score can be maintained to determine the number of correct words guessed simultaneously, without any break.
A time quantum can be introduced to determine the speed in which the user is able to guess the words correctly.

REFERENCES

https://www.javatpoint.com/socket-programming

https://www.javatpoint.com/java-oops-concepts

https://www.javatpoint.com/java-swing

https://www.geeksforgeeks.org/caesar-cipher-in-cryptography/

https://www.geeksforgeeks.org/difference-between-client-server-and-peer-to-peer-network/

https://en.wikipedia.org/wiki/Transmission_Control_Protocol