WEEK : 9

Write a Java program about Socket Program for network chatting and display it .

SERVER.JAVA

```
import java.io.*;
import java.net.*;
import java.util.HashMap;
import java.util.Map;

public class Server {
    private static final int PORT = 12306;
    private static final Map<String, String> responses = new HashMap<>();

    static {
        loadResponsesFromFile("C:\\Users\\user\\Desktop\\chat bot answer.txt");
    }

    public static void main(String[] args) {
        try {
            ServerSocket serverSocket = new ServerSocket(PORT);
            System.out.println("Server listening on port " + PORT + "...");

            while (true) {
                Socket clientSocket = serverSocket.accept();
                System.out.println("Client connected: " + clientSocket);

                Thread clientThread = new ClientThread(clientSocket);
                clientThread.start();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    static class ClientThread extends Thread {
```

```
        private Socket clientSocket;
        private PrintWriter out;
        private BufferedReader in;

        public ClientThread(Socket socket) {
            this.clientSocket = socket;
        }

        public void run() {
            try {
                out = new PrintWriter(clientSocket.getOutputStream(), true);
                in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));

                String message;
                while ((message = in.readLine()) != null) {
                    System.out.println("Received from client: " + message);

                    // Process the user's message using the responses map
                    String response = responses.getOrDefault(message.toLowerCase(),
"I'm not sure how to respond to that. Please ask another question.");

                    out.println(response);
                }
            } catch (IOException e) {
                e.printStackTrace();
            } finally {
                try {
                    in.close();
                    out.close();
                    clientSocket.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }
    }
```

```
    private static void loadResponsesFromFile(String filePath) {
        try (BufferedReader reader = new BufferedReader(new
FileReader(filePath))) {
            String line;
            while ((line = reader.readLine()) != null) {
                String[] parts = line.split("\\|");
                if (parts.length == 2) {
                    String question = parts[0].trim();
                    String response = parts[1].trim();
                    responses.put(question.toLowerCase(), response);
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

CLIENT .JAVA

```
import java.io.*;
import java.net.*;
 import javafx.application.Application;
import javafx.beans.property.SimpleListProperty;
import javafx.collections.FXCollections;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;
import javafx.util.Duration;
import java.net.Socket;
public class Clientt extends Application{

    private Socket socket;
    private PrintWriter out;
    private BufferedReader in;
```

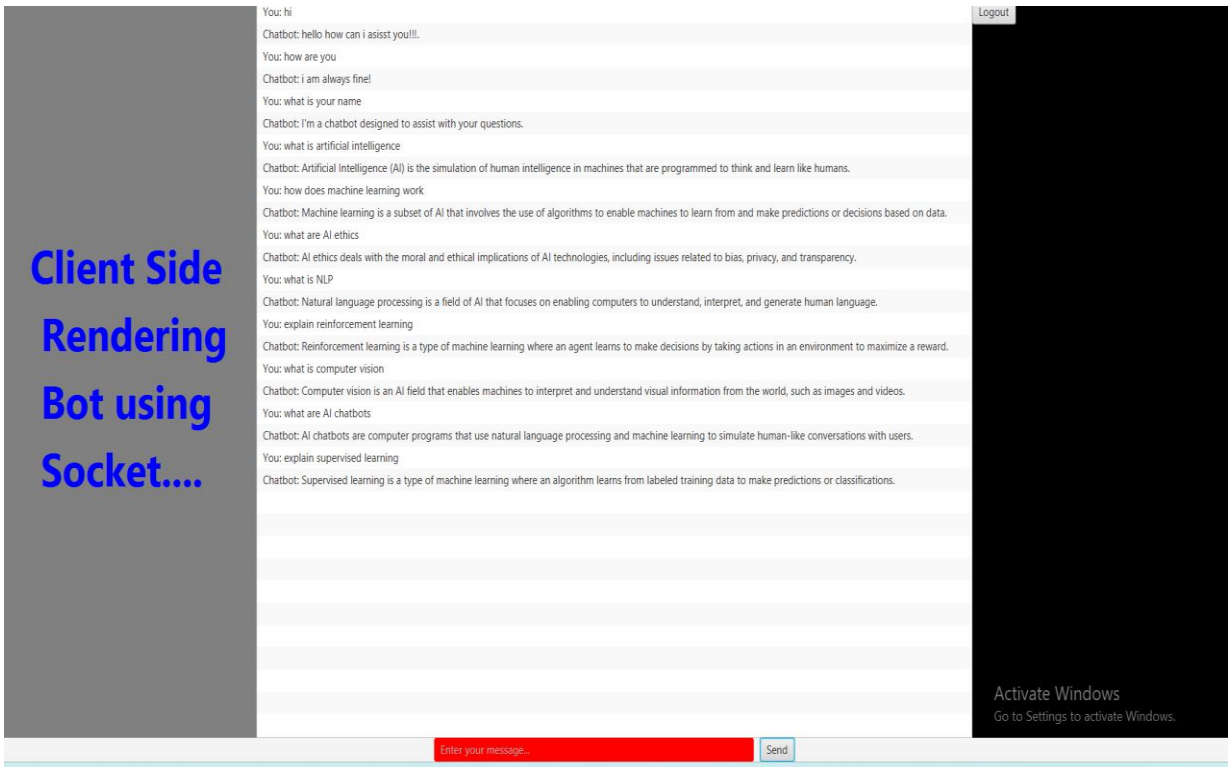
```
    private ListView<String> chatListView;
    private TextField messageField;
    private Button sendButton;

    private SimpleListProperty<String> chatMessages = new
SimpleListProperty<>(FXCollections.observableArrayList());

    public static void main(String[] args) {
        launch(args);
    }

    @Override
    public void start(Stage primaryStage) {
        primaryStage.setTitle("Chat Client");

        chatListView = new ListView<>();
        chatListView.setItems(chatMessages);

        messageField = new TextField();
        messageField.setPromptText("Enter your message...");
          messageField.setStyle("-fx-background-color: red;");
          messageField.setPrefWidth(500);
        messageField.setOnAction(e -> sendMessage());

        sendButton = new Button("Send");
        sendButton.setOnAction(e -> sendMessage());

       HBox inputBox = new HBox(10);
          inputBox.setAlignment(Pos.CENTER); // Center horizontally
          inputBox.getChildren().addAll(messageField, sendButton);
        BorderPane borderPane = new BorderPane();
        borderPane.setCenter(chatListView);
        borderPane.setBottom(inputBox);
          Label leftLabel = new Label("Client Side\n Rendering\n Bot using\n
Socket....");
          StackPane root = new StackPane(leftLabel);
```

```
        Scene scenes = new Scene(root, 300, 200);

        primaryStage.setTitle("Sequential Dot Effect");
        primaryStage.setScene(scenes);
        primaryStage.show();

        // Create a timeline for the dot animation
        Timeline dotAnimation = new Timeline(
                new KeyFrame(Duration.ZERO, new
KeyValue(leftLabel.textProperty(), "Client Side\n Rendering\n Bot using\n
Socket.")),
                new KeyFrame(Duration.seconds(0.5), new
KeyValue(leftLabel.textProperty(), "Client Side\n Rendering\n Bot using\n
Socket..")),
                new KeyFrame(Duration.seconds(1), new
KeyValue(leftLabel.textProperty(), "Client Side\n Rendering\n Bot using\n
Socket...")),
                new KeyFrame(Duration.seconds(1.5), new
KeyValue(leftLabel.textProperty(), "Client Side\n Rendering\n Bot using\n
Socket....")),
                    new KeyFrame(Duration.seconds(2), new
KeyValue(leftLabel.textProperty(), "Client Side\n Rendering\n Bot using\n
Socket.....'))
        );

        dotAnimation.setCycleCount(Timeline.INDEFINITE);
        dotAnimation.play();
          leftLabel.setStyle("-fx-font-size: 60px; -fx-font-weight: bold; -fx-text-fill:
blue;");
        // Create left and right panes
        VBox leftPane = new VBox();
        leftPane.setStyle("-fx-background-color: gray;");
        leftPane.setPrefWidth(400);
          leftPane.setAlignment(Pos.CENTER); // Center the label vertically
          leftPane.getChildren().add(leftLabel);
```

```
        VBox rightPane = new VBox();
        rightPane.setStyle("-fx-background-color: black;");
        rightPane.setPrefWidth(400);
          Button logoutButton = new Button("Logout");
        Scene scene = new Scene(borderPane, 800, 600);

        primaryStage.setScene(scene);
        primaryStage.show();

        connectToServer();
    }
    private void receiveMessages() {
        try {
            String message;
            while ((message = in.readLine()) != null) {
                chatMessages.add("Chatbot: " + message);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    private void sendMessage() {
        String message = messageField.getText();
        if (!message.isEmpty()) {
            out.println(message);
            chatMessages.add("You: " + message);
            messageField.clear();
        }
    }
        }
        System.exit(0); // Terminate the application
    }
}
```

OUTPUT:

```
run:
Server listening on port 12306...
Client connected: Socket[addr=/127.0.0.1,port=64312,localport=12306]
Received from client: hi
Received from client: how are you
Received from client: what is your name
Received from client: what is artificial intelligence
Received from client: what is deep learning
Received from client: what is NLP
Received from client: explain reinforcement learning
Received from client: what is computer visio
Received from client: what are AI chatbots
Received from client: what are neural networks
```



RESULT: