package com.chatapp;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.\*;

import java.net.\*;

public class ServerGUI extends JFrame {

private JTextArea chatArea;

private JTextField messageField;

private JButton sendButton;

private ServerSocket serverSocket;

private Socket socket;

private DataInputStream input;

private DataOutputStream output;

public ServerGUI() {

setTitle("Server Chat");

setSize(400, 500);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

chatArea = new JTextArea();

chatArea.setEditable(false);

JScrollPane scrollPane = new JScrollPane(chatArea);

messageField = new JTextField();

sendButton = new JButton("Send");

sendButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

sendMessage();

}

});

JPanel panel = new JPanel(new BorderLayout());

panel.add(messageField, BorderLayout.CENTER);

panel.add(sendButton, BorderLayout.EAST);

add(scrollPane, BorderLayout.CENTER);

add(panel, BorderLayout.SOUTH);

setVisible(true);

startServer();

}

private void startServer() {

try {

serverSocket = new ServerSocket(1234);

chatArea.append("Server started. Waiting for client...\n");

socket = serverSocket.accept();

chatArea.append("Client connected.\n");

input = new DataInputStream(socket.getInputStream());

output = new DataOutputStream(socket.getOutputStream());

AESutil.generateKey();

output.writeUTF(AESutil.getSecretKey());

new Thread(() -> receiveMessages()).start();

} catch (Exception e) {

chatArea.append("Error: " + e.getMessage() + "\n");

}

}

private void sendMessage() {

try {

String message = messageField.getText();

if (!message.isEmpty()) {

String encryptedMessage = AESutil.encrypt(message);

output.writeUTF(encryptedMessage);

chatArea.append("You: " + message + "\n");

messageField.setText("");

}

} catch (Exception e) {

chatArea.append("Error: " + e.getMessage() + "\n");

}

}

private void receiveMessages() {

try {

while (true) {

String encryptedMessage = input.readUTF();

String decryptedMessage = AESutil.decrypt(encryptedMessage);

chatArea.append("Client: " + decryptedMessage + "\n");

}

} catch (Exception e) {

chatArea.append("Connection closed.\n");

}

}

public static void main(String[] args) {

new ServerGUI();

}

}