**Messaging App**

**Adina Zafar: SP24-BSE-006**

**Aminah Tariq: SP2-BSE-011**

**Server End:**

**Classes:**

1. Person class
2. Sms class
3. Arabic Sms class
4. French Sms class
5. Spanish Sms class
6. Client Handler class
7. Sms Server class

**Attributes of Classes:**

Person class:

private String Name;  
private String Contact;  
private String Email;

Sms class:

private final String Id;  
private final Person Sender= new Person("aminah", "03701690844", "aminahtariq@gmail.com");  
private final Person Receiver= new Person("adina", "03771690844", "adina@gmail.com");;  
private String Content;  
private boolean Status;  
String receivedTime;  
String sendtime;  
static int *counter* = 0;

Client Handler class:

private Socket socket; private Socket socket;  
private ObjectInputStream input;  
private ObjectOutputStream output;

Sms Server class:

static ArrayList<Sms> *sms1* = new ArrayList<>();

**Client End:**

**Classes:**

1. Person class
2. Sms class
3. Arabic Sms class
4. French Sms class
5. Spanish Sms class
6. Console Input Handler class
7. Sms Client class

**Attributes of Classes:**

Person class:

private String Name;  
private String Contact;  
private String Email;

Sms class:

private final String Id;  
private final Person Sender= new Person("aminah", "03701690844", "aminahtariq@gmail.com");  
private final Person Receiver= new Person("adina", "03771690844", "adina@gmail.com");;  
private String Content;  
private boolean Status;  
String receivedTime;  
String sendtime;  
static int *counter* = 0;

Console Input Handler class:

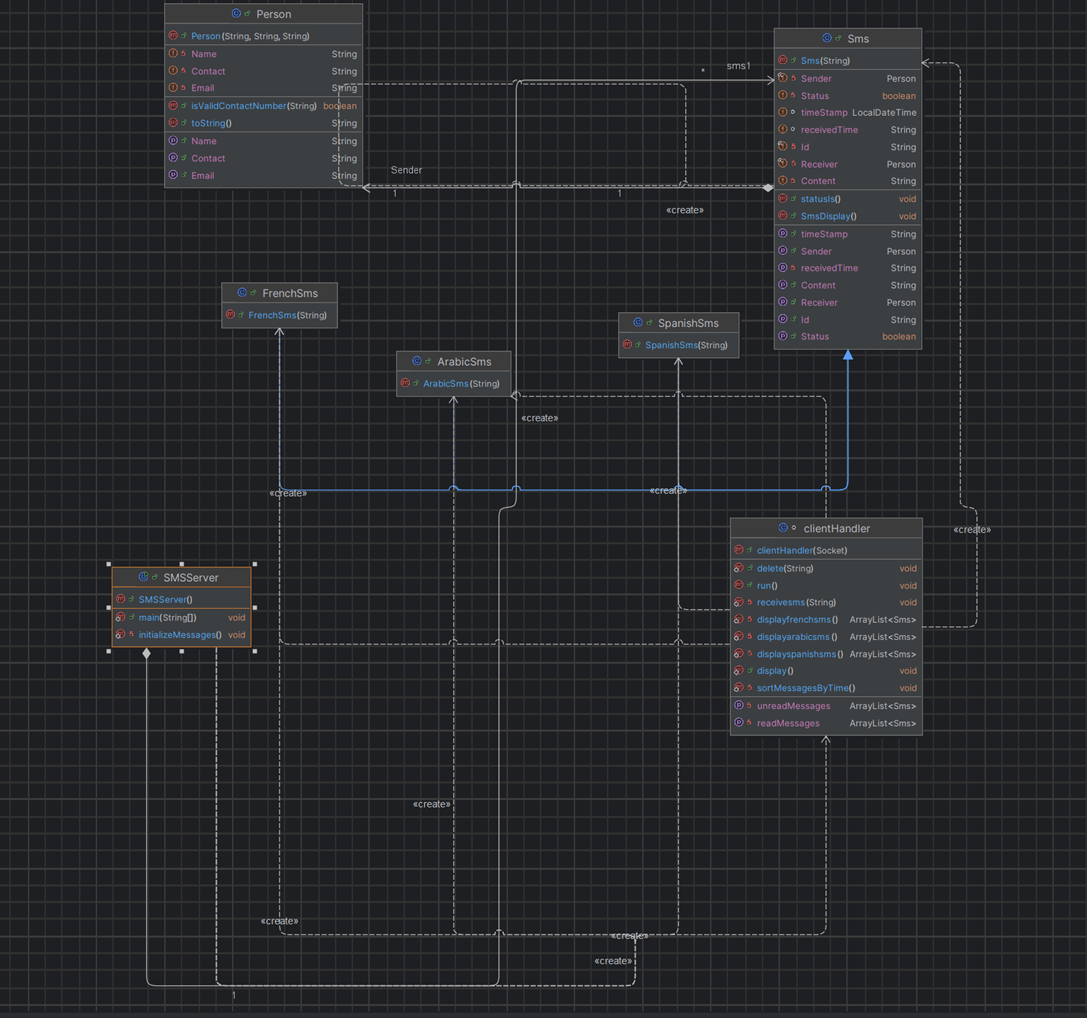
static ArrayList<Sms> sms1 = new ArrayList<>();  
  
 boolean running = true;

Sms Client class:

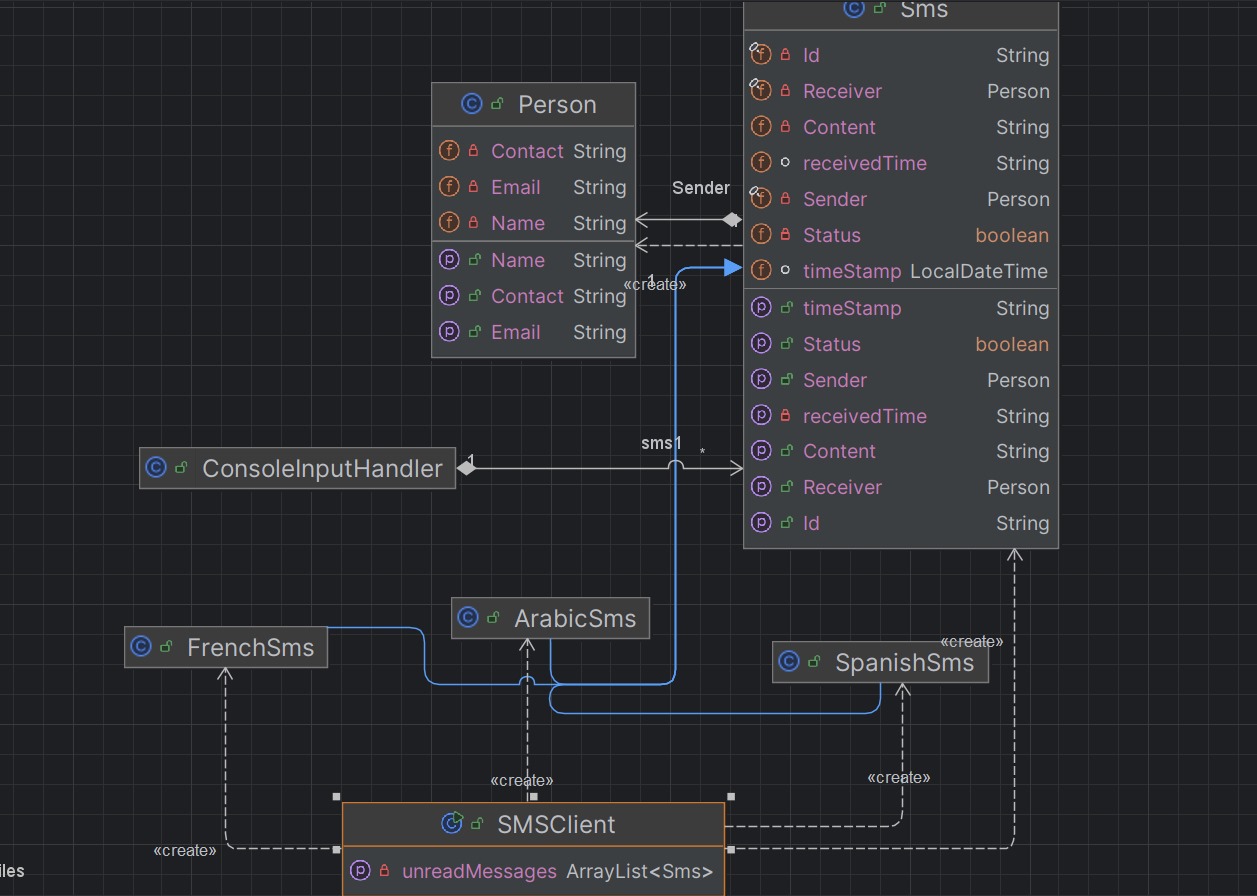
ObjectOutputStream output = new ObjectOutputStream(socket.getOutputStream());  
ObjectInputStream input = new ObjectInputStream(socket.getInputStream());  
Scanner scanner = new Scanner(System.in)

**UML Diagram:**

**Server End:**

****

**Client End:**



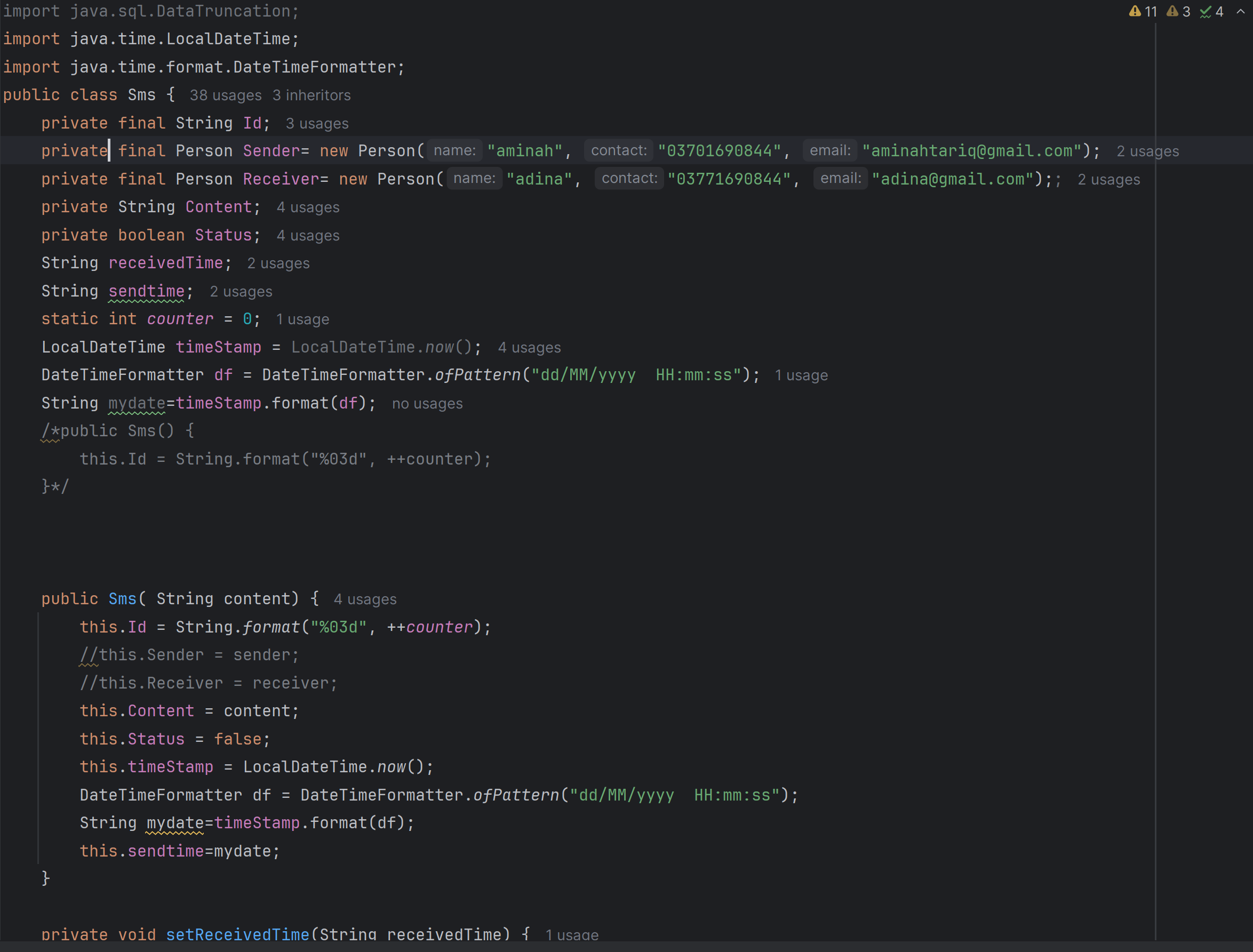
**Source Code:**

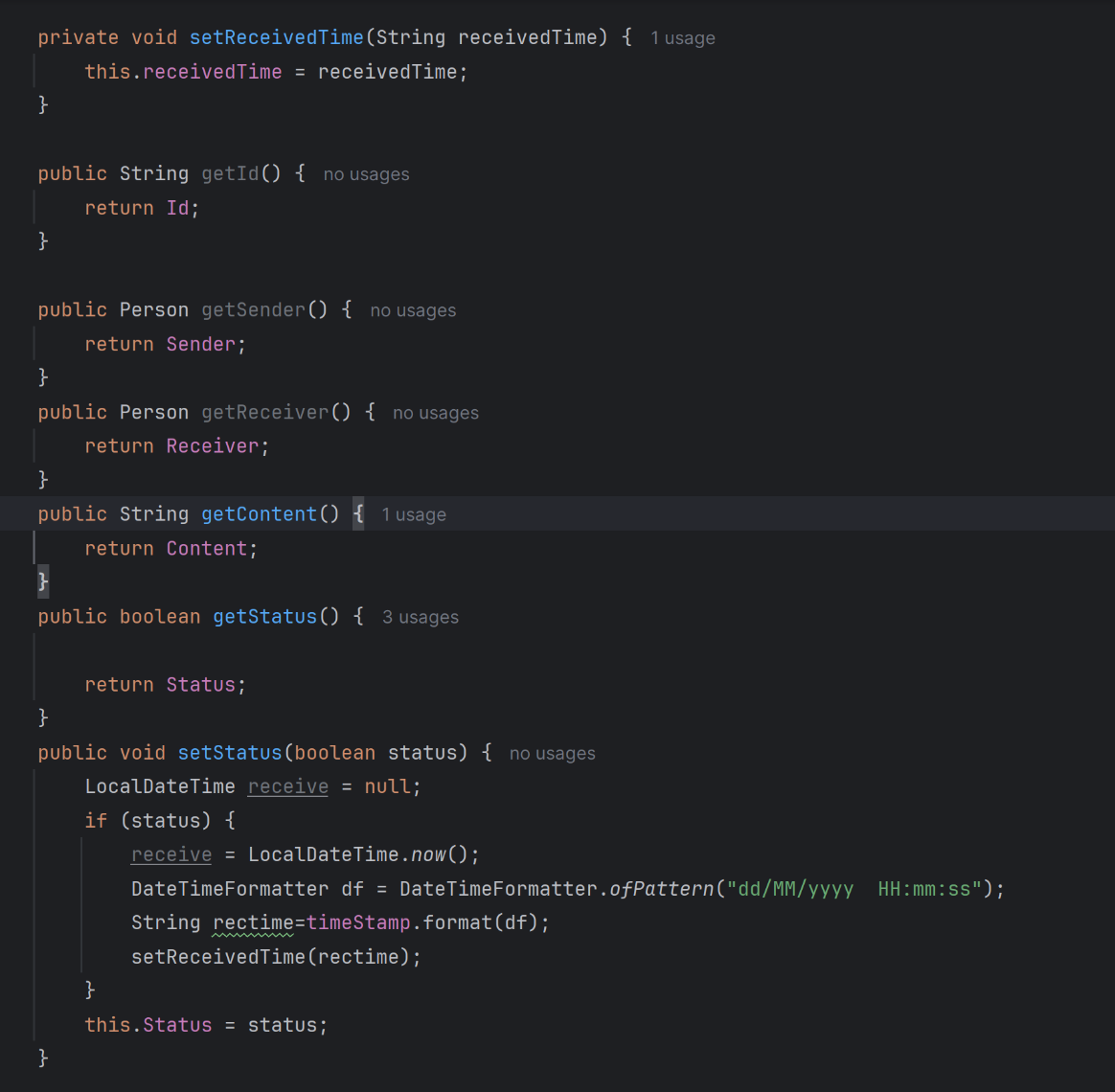
**Server End:**

**Person class:**

****

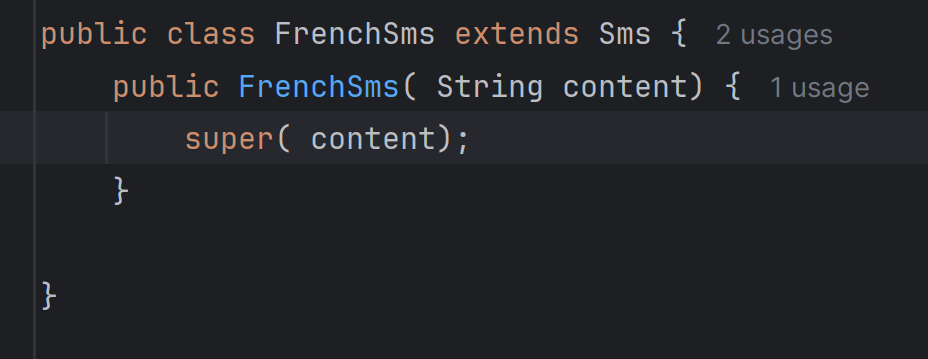
**SMS class:**



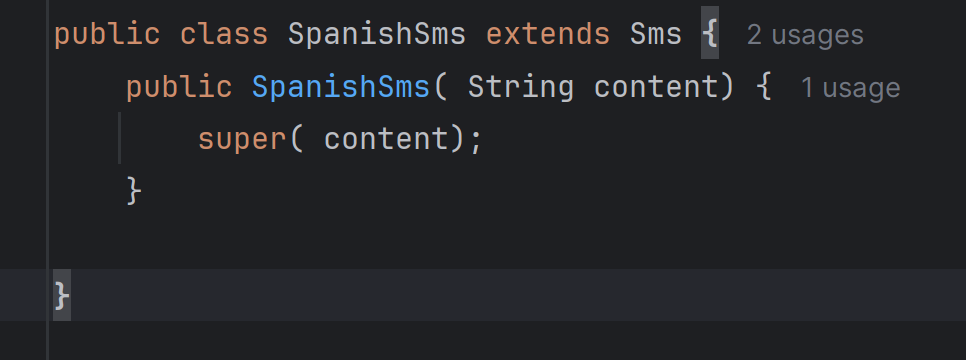




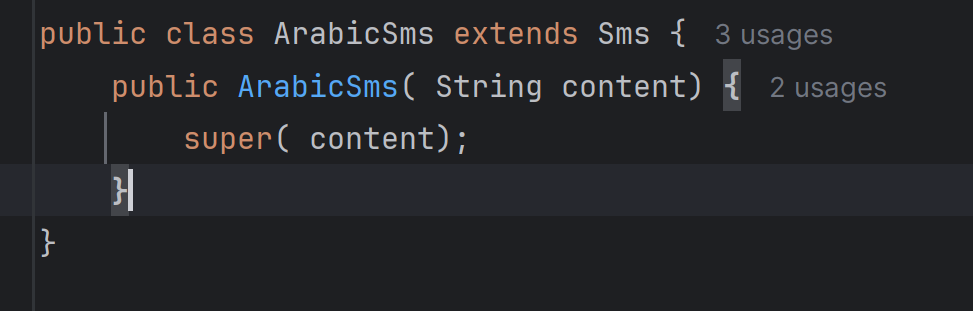
French Sms Class:



Spanish Sms Class:



Arabic Sms Class:



**Client Handler Class:**

import java.io.\*;  
import java.net.\*;  
import java.util.ArrayList;  
import java.util.Comparator;  
import java.util.Scanner;  
  
class clientHandler extends Thread {  
 private Socket socket;  
 private ObjectInputStream input;  
 private ObjectOutputStream output;  
 // private static ArrayList<Sms> sms = new ArrayList<>();  
 public clientHandler(Socket socket) {  
 this.socket = socket;  
 }  
 public void run() {  
 try {  
 input = new ObjectInputStream(socket.getInputStream());  
 output = new ObjectOutputStream(socket.getOutputStream());  
 output.writeObject("Welcome to the SMS Server!");  
  
 Thread receiveThread = new Thread(() -> {  
 try {  
 while (true) {  
 String message = (String) input.readObject();  
 System.*out*.println("Aminah: " + message);  
 //String content = (String) input.readObject();  
 Sms sms = new Sms(message);  
 // sms.SmsDisplay();  
 SMSServer.*sms1*.add(sms);  
 output.writeObject("Message received and stored.");

/\* if ("send".equalsIgnoreCase(message)) {  
 String content = (String) input.readObject();  
 Sms sms = new Sms(content);  
 sms.SmsDisplay();  
 SMSServer.sms1.add(sms);  
 output.writeObject("Message received and stored.");  
 }\*/  
 }  
 } catch (IOException | ClassNotFoundException ex) {  
 System.*out*.println("Client disconnected.");  
 }  
 });

receiveThread.start();  
 Scanner scanner = new Scanner(System.*in*);  
 while (true) {  
 System.*out*.println("1. Send SMS");  
 System.*out*.println("2. delete ");  
 System.*out*.println("3. display" );  
 System.*out*.println("4. sort by time");  
 System.*out*.println("5. receive message");  
 System.*out*.println("6. Exit");  
 System.*out*.print("Enter option: \n");  
 String sms = scanner.nextLine();  
  
 switch (sms) {  
 case "1":  
 System.*out*.println("choose language in which you want to send message:");  
 System.*out*.println("1. French");  
 System.*out*.println("2. spanish");  
 System.*out*.println("3. Arabic");  
 String sing4 = scanner.nextLine();  
 switch (sing4) {

case "1":  
 output.writeObject("sending message");  
 System.*out*.print("Enter message \n");  
 String sms2 = scanner.nextLine();  
 Sms smss = new FrenchSms(sms2);  
 SMSServer.*sms1*.add(smss);  
 System.*out*.println("Message sent");  
 break;

case "2":  
 output.writeObject("sending message");  
 System.*out*.print("Enter message \n");  
 String sms3 = scanner.nextLine();  
 Sms smsw = new SpanishSms(sms3);  
 SMSServer.*sms1*.add(smsw);  
 System.*out*.println("Message sent");  
 break;

case "3":  
 output.writeObject("sending message");  
 System.*out*.print("Enter message \n");  
 String sms9 = scanner.nextLine();  
 Sms smsq = new ArabicSms(sms9);  
 SMSServer.*sms1*.add(smsq);  
 System.*out*.println("Message sent");  
 break;  
 default:  
 System.*out*.println("Invalid option");  
 break;  
 }  
 break;

case "2":  
 System.*out*.print("Enter message you want to delete: ");  
 String message1 = scanner.nextLine();  
 *delete*(message1);  
 break;

case "3":  
 System.*out*.println("choose your option:");  
 System.*out*.println("1. display all sms");  
 System.*out*.println("2. display by status");  
 System.*out*.println("3. display by languages");  
 String sing = scanner.nextLine();  
 switch (sing) {

case "1":  
 for(Sms smss1 : SMSServer.*sms1*) {  
 smss1.SmsDisplay();  
 }  
 break;

case "2":  
 System.*out*.println("how do you want to display by status");  
 System.*out*.println("1. display unread");  
 System.*out*.println("2. display read");  
 String sing1 = scanner.nextLine();  
 switch (sing1) {  
 case "1":  
 ArrayList<Sms> smsr2 = *getUnreadMessages*();  
 for (Sms sms6 : smsr2) {  
 sms6.SmsDisplay();  
 }  
 break;  
 case "2":  
 ArrayList<Sms> sms3 = *getReadMessages*();  
 for (Sms smst : sms3) {  
 smst.SmsDisplay();  
 }  
 break;  
 default:  
 System.*out*.println("Invalid option");  
 break;  
 }  
 break;

case "3":  
 System.*out*.println("which language do you want to display:");  
 System.*out*.println("1. French");  
 System.*out*.println("2. spanish");  
 System.*out*.println("3. Arabic");  
 String sing3 = scanner.nextLine();  
 ArrayList<Sms> sms4= new ArrayList<>();  
 switch (sing3) {  
 case "1":  
 sms4 = *displayspanishsms*();  
 for (Sms sms5 : sms4) {  
 sms5.SmsDisplay();  
 }  
 break;  
 case "2":  
 sms4 = *displayfrenchsms*();  
 for (Sms sms5 : sms4) {  
 sms5.SmsDisplay();  
 }  
 break;  
 case "3":  
 sms4 = *displayarabicsms*();  
 for (Sms sms5 : sms4) {  
 sms5.SmsDisplay();  
 }  
 break;  
 default:  
 break;  
 }  
  
 break;  
 default:  
 break;  
 }  
 case "4":  
 *sortMessagesByTime*();  
  
 output.writeObject("Messages sorted by time.");  
 break;

case "5":  
 for(Sms smss1 : SMSServer.*sms1*) {  
 smss1.SmsDisplay();  
 }  
 System.*out*.print("\nEnter message you want to delete: ");  
 String message18 = scanner.nextLine();  
 *receivesms*(message18);  
 break;

case "6":  
 output.writeObject("Disconnecting...");  
 System.*out*.println("Client disconnected.");  
 socket.close();  
 return;  
 default:  
 output.writeObject("Invalid command.");  
 break;  
  
 }  
 if ("exit".equalsIgnoreCase(sms)) {  
 System.*out*.println("Disconnecting client...");  
 output.writeObject("Server disconnecting...");  
 socket.close();  
 break;  
 }  
  
 // output.writeObject(message);  
 output.flush();  
 }

} catch (IOException e) {  
 throw new RuntimeException(e);  
 }}

public static void delete(String message) {  
 Sms toDelete = null;  
 for (Sms msg : SMSServer.*sms1*) {  
 if (msg.getContent().equals(message)) {  
 toDelete = msg;  
 break;  
 } }

if (toDelete != null) {  
 SMSServer.*sms1*.remove(toDelete);  
 }// return toDelete;  
 }  
 public static void display(){  
 for (Sms msg: SMSServer.*sms1*){  
 msg.SmsDisplay();  
 }}

private static ArrayList<Sms> getUnreadMessages() {  
 ArrayList<Sms> unreadMessages = new ArrayList<>();  
 for (Sms msg : SMSServer.*sms1*) {  
 if (!msg.getStatus()) unreadMessages.add(msg);  
 }  
 return unreadMessages;  
 }  
  
 private static ArrayList<Sms> getReadMessages() {  
 ArrayList<Sms> readMessages = new ArrayList<>();  
 for (Sms msg : SMSServer.*sms1*) {  
 if (msg.getStatus()) readMessages.add(msg);  
 }  
 return readMessages;  
 }  
  
 private static void sortMessagesByTime() {  
 if (SMSServer.*sms1* != null) {  
 SMSServer.*sms1*.sort(Comparator.*comparing*(Sms::getTimeStamp));  
 for (Sms msg : SMSServer.*sms1*) {  
 msg.SmsDisplay();  
 }  
 } else {  
 System.*out*.println("No messages to sort.");  
 }}

private static ArrayList<Sms> displayfrenchsms(){  
 ArrayList<Sms> french = new ArrayList<>();  
 for(Sms sms : SMSServer.*sms1*){  
 if(sms instanceof FrenchSms){  
 // System.out.println(sms);  
 french.add(sms);  
 }}  
 return french;  
 }

private static ArrayList<Sms> displayspanishsms(){  
 ArrayList<Sms> spanish = new ArrayList<>();  
 for(Sms sms : SMSServer.*sms1*){  
 if(sms instanceof SpanishSms){  
 // System.out.println(sms);  
 spanish.add(sms);  
 }}  
 return spanish;  
 }

private static ArrayList<Sms> displayarabicsms(){  
 ArrayList<Sms> arabic = new ArrayList<>();  
 for(Sms sms : SMSServer.*sms1*){  
 if(sms instanceof ArabicSms){  
 //System.out.println(sms);  
 arabic.add(sms);  
 }}  
 return arabic;  
 }

private static void receivesms(String message) {  
 for (Sms msg : SMSServer.*sms1*) {  
 if (msg.getContent().equals(message)) {  
 if(!msg.getStatus()){  
 msg.setStatus(true);  
 System.*out*.println("SMS received");  
 } else {  
 System.*out*.println("message has already been received.");  
 }  
 } }  
 }}

SMS Server:

import java.io.\*;  
import java.net.\*;  
import java.util.ArrayList;  
public class SMSServer {  
 static ArrayList<Sms> *sms1* = new ArrayList<>();  
 public static void main(String[] args) {  
  
 try (ServerSocket serverSocket = new ServerSocket(5050)) {  
 System.*out*.println("Server is listening on port 5050");  
  
 while (true) {  
 Socket socket = serverSocket.accept();  
 System.*out*.println("New client connected");  
 *initializeMessages*();  
 clientHandler handler = new clientHandler(socket);  
 handler.start();  
 }  
 } catch (IOException ex) {  
 ex.printStackTrace();  
 }  
 }  
  
 private static void initializeMessages() {  
 SMSServer.*sms1*.add(new SpanishSms("Hola, cómo estás?"));  
 SMSServer.*sms1*.add(new FrenchSms("Bonjour, comment ça va?"));  
 SMSServer.*sms1*.add(new ArabicSms("مرحبا، كيف حالك؟"));  
 }  
}

**Client End**:

On the client end, person class, Sms class, SpanishSms class, FrenchSms Class, ArabicSms class are exactly the same as in Server end.

However, the source code for the clientSms and ConsoleInputHandler class is provided:

Console Input Handler Class:

import java.util.ArrayList;  
import java.util.Scanner;  
  
  
public class ConsoleInputHandler implements Runnable {  
 static ArrayList<Sms> sms1 = new ArrayList<>();  
  
 public void run() {  
 boolean running = true;  
 Scanner scanner = new Scanner(System.in);  
 while (running) {  
 System.out.println("Type a command (or 'shutdown' to stop the server):");  
 String input = scanner.nextLine();  
 if ("shutdown".equalsIgnoreCase(input)) {  
 running = false;  
 System.out.println("Shutting down server...");  
 break;  
 } else {  
 System.out.println("Unknown command.");  
 }  
 }  
 scanner.close();  
 }  
}

**Sms Client Class:**

import java.io.\*;  
import java.net.\*;  
import java.util.ArrayList;  
import java.util.Comparator;  
import java.util.Scanner;  
  
public class SMSClient {  
 public static void main(String[] args) {  
 //ArrayList<Sms> sms1 = new ArrayList<>();  
 ConsoleInputHandler.sms1.add(new SpanishSms("Hola, cómo estás?"));  
 ConsoleInputHandler.sms1.add(new FrenchSms("Bonjour, comment ça va?"));  
 ConsoleInputHandler.sms1.add(new ArabicSms("مرحبا، كيف حالك؟"));  
  
 try (Socket socket = new Socket("192.168.1.19", 5050);

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

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

Scanner scanner = new Scanner(System.in)) {  
 System.out.println("Connected to the server.");

System.out.println((String) input.readObject());

Thread receiveThread = new Thread(() -> {

try {  
 while (true) {  
 String message = (String) input.readObject();  
 System.out.println("Adina: " + message);  
 Sms sms = new Sms(message);  
 ConsoleInputHandler.sms1.add(sms);  
 System.out.println("message received and stored");  
 }

} catch (IOException | ClassNotFoundException ex) {  
 System.out.println("Connection closed.");  
 }  
 });

receiveThread.start();  
   
 while (true) {  
 System.out.println("1. Send SMS");  
 System.out.println("2. delete ");  
 System.out.println("3. display" );  
 System.out.println("4. sort by time");  
 System.out.println("5. receive message");  
 System.out.println("5. Exit");  
 System.out.print("Enter option: \n");  
 String command = scanner.nextLine();  
  
 switch (command) {

case "1":  
 System.out.println("choose language in which you want to send message:");  
 System.out.println("1. French");  
 System.out.println("2. spanish");  
 System.out.println("3. Arabic");  
 String sing4 = scanner.nextLine();  
 switch (sing4) {

case "1":  
 output.writeObject("sending message");  
 System.out.print("Enter message \n");  
 String sms2 = scanner.nextLine();  
 Sms smss = new FrenchSms(sms2);  
 ConsoleInputHandler.sms1.add(smss);  
 System.out.println("Message sent");  
 break;

case "2":  
 output.writeObject("sending message");  
 System.out.print("Enter message \n");  
 String sms3 = scanner.nextLine();  
 Sms smsw = new SpanishSms(sms3);

ConsoleInputHandler.sms1.add(smsw);  
 System.out.println("Message sent");  
 break;

case "3":  
 output.writeObject("sending message");  
 System.out.print("Enter message \n");

String sms9 = scanner.nextLine();  
 Sms smsq = new ArabicSms(sms9);  
 ConsoleInputHandler.sms1.add(smsq);  
 System.out.println("Message sent");  
 break;

default:  
 System.out.println("Invalid option");  
 break;  
 }

break;  
  
 case "2":  
 for(Sms smss1 : ConsoleInputHandler.sms1) {  
 smss1.SmsDisplay();  
 }  
 System.out.print("\nEnter message you want to delete: ");  
 String message = scanner.nextLine();  
 delete(message);  
 System.out.println("Message deleted.");  
 break;

case "3":  
 System.out.println("choose your option:");  
 System.out.println("1. display all sms");  
 System.out.println("2. display by status");  
 System.out.println("3. display by languages");  
 String sing = scanner.nextLine();

switch (sing) {

case "1":  
 for(Sms smss1 : ConsoleInputHandler.sms1) {  
 smss1.SmsDisplay();  
 }  
 break;

case "2":  
 System.out.println("how do you want to display by status");  
 System.out.println("1. display unread");  
 System.out.println("2. display read");  
 String sing12 = scanner.nextLine();

switch (sing12) {

case "1":  
 ArrayList<Sms> smsrw2 = getUnreadMessages();  
 for (Sms sms6 : smsrw2) {  
 sms6.SmsDisplay();  
 } break;

case "2":  
 getReadMessages();  
 /\* for (Sms smst : sms3) {  
 smst.SmsDisplay();  
 }\*/  
 break;  
 default:  
 System.out.println("Invalid option");  
 break;  
 } break;

case "3":  
 System.out.println("which language do you want to display:");  
 System.out.println("1. Spanish");  
 System.out.println("2. French");  
 System.out.println("3. Arabic");  
 String sing3 = scanner.nextLine();

ArrayList<Sms> sms4= new ArrayList<>();

switch (sing3) {  
 case "1":  
 sms4 = displayspanishsms();  
 for (Sms sms5 : sms4) {  
 sms5.SmsDisplay();  
 }  
 break;

case "2":  
 displayfrenchsms();

/\*for (Sms sms5 : sms4) {  
 sms5.SmsDisplay();  
 }\*/  
 break;

case "3":  
 sms4 = displayarabicsms();  
 for (Sms sms5 : sms4) {  
 sms5.SmsDisplay();  
 }  
 break;

default:  
 break;  
 } break;  
 default:  
 break;  
 }  
 case "4":  
 sortMessagesByTime();  
 break;

case "5":  
 for(Sms smss1 : ConsoleInputHandler.sms1) {  
 smss1.SmsDisplay();  
 }  
 System.out.print("\nEnter message you want to delete: ");  
 String message1 = scanner.nextLine();  
 receivesms(message1);  
 break;

case "6":  
 System.out.println("Disconnecting from server...");  
 output.flush();  
 break;

default:  
 System.out.println("Invalid option.");  
 break;  
 }}  
  
 } catch (IOException | ClassNotFoundException ex) {  
 ex.printStackTrace();  
 } }  
  
 public static Sms delete(String message) {  
 Sms toDelete = null;  
  
 for (Sms msg : ConsoleInputHandler.sms1) {  
 if (msg.getContent().equals(message)) {  
 toDelete = msg;  
 break;  
 } }  
  
 if (toDelete != null) {  
 ConsoleInputHandler.sms1.remove(toDelete);  
 }  
 return toDelete;  
 }  
  
 public static void display(){  
 for (Sms msg: ConsoleInputHandler.sms1){  
 msg.SmsDisplay();  
 }}  
  
 private static ArrayList<Sms> getUnreadMessages() {  
 ArrayList<Sms> unreadMessages = new ArrayList<>();  
 for (Sms msg : ConsoleInputHandler.sms1) {  
 if (!msg.getStatus()) unreadMessages.add(msg);  
 }  
 return unreadMessages;  
 }  
  
 private static void getReadMessages() {  
 // ArrayList<Sms> readMessages = new ArrayList<>();  
 for (Sms msg : ConsoleInputHandler.sms1) {  
 if(msg.getStatus()){  
 msg.SmsDisplay();  
 // System.out.println("SMS received");  
 }}  
 }  
  
 private static void sortMessagesByTime() {  
 if (ConsoleInputHandler.sms1 != null) {  
 ConsoleInputHandler.sms1.sort(Comparator.comparing(Sms::getTimeStamp));  
 for (Sms msg : ConsoleInputHandler.sms1) {  
 msg.SmsDisplay();  
 }  
 } else {  
 System.out.println("No messages to sort.");  
 } }

/\* private static ArrayList<Sms> displayfrenchsms(){  
 ArrayList<Sms> french = new ArrayList<>();  
 for(Sms sms : ConsoleInputHandler.sms1){  
 if(sms instanceof FrenchSms){  
  
 french.add(sms);  
 } }  
 return french;  
 }\*/  
  
 private static void displayfrenchsms() {  
 ArrayList<Sms> french = new ArrayList<>();  
 for (Sms sms : ConsoleInputHandler.sms1) {  
 if (sms instanceof FrenchSms) {  
 sms.SmsDisplay();  
 // FrenchSms frenchSms = (FrenchSms) sms;  
 // french.add(frenchSms);  
 }  
 } // return french;  
 }  
   
 private static ArrayList<Sms> displayspanishsms(){  
 ArrayList<Sms> spanish = new ArrayList<>();  
 for(Sms sms : ConsoleInputHandler.sms1){  
 if(sms instanceof SpanishSms){  
 spanish.add(sms);  
 } }  
 return spanish;  
 }

private static ArrayList<Sms> displayarabicsms(){  
 ArrayList<Sms> arabic = new ArrayList<>();  
 for(Sms sms : ConsoleInputHandler.sms1){  
 if(sms instanceof ArabicSms){  
 arabic.add(sms);  
 } }  
 return arabic;  
 }  
  
 private static void receivesms(String message) {  
 for (Sms msg : ConsoleInputHandler.sms1) {  
 if (msg.getContent().equals(message)) {  
 if(!msg.getStatus()){  
 msg.setStatus(true);  
 System.out.println("SMS received");  
 } else {  
 System.out.println("message has already been received.");  
 }}  
 }

}

}