

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

INTRODUCCIÓN A LA PROGRAMACIÓN Y COMPUTACIÓN 1

CATEDRÁTICO: ING. Neftalí Calderón

TUTOR ACADÉMICO: JOSUÉ RODOLFO MORALES CASTILLO



David Norberto Fabro Guzmán

CARNÉ: 202307499

SECCIÓN: A

GUATEMALA, 13 DE JUNIO DEL 2,024

# ÍNDICE

<b>ÍNDICE</b>	<b>1</b>
<b>INTRODUCCIÓN</b>	<b>1</b>
<b>OBJETIVOS</b>	<b>1</b>
1. GENERAL	1
2. ESPECÍFICOS	1
<b>ALCANCES DEL SISTEMA</b>	<b>1</b>
<b>ESPECIFICACIÓN TÉCNICA</b>	<b>1</b>
• REQUISITOS DE HARDWARE	1
• REQUISITOS DE SOFTWARE	1
<b>DESCRIPCIÓN DE LA SOLUCIÓN</b>	<b>2</b>
<b>LÓGICA DEL PROGRAMA</b>	<b>2</b>
❖ NOMBRE DE LA CLASE	
Captura de las librerías usadas	2
➤ Librerías	2
➤ Variables Globales de la clase _(El nombre de su clase actual)	3
➤ Función Main	3
➤ Métodos y Funciones utilizadas	3

# **INTRODUCCIÓN**

Este manual tiene como fin proporcionar una guía detallada y comprensiva para los desarrolladores y usuarios del sistema de chat seguro CYPHERCHAT. En él se describen los objetivos del sistema, los requisitos necesarios para su funcionamiento, la lógica y estructura del código, así como las funciones y métodos utilizados. El propósito es facilitar la comprensión y uso del sistema, garantizando que los usuarios puedan aprovechar al máximo sus funcionalidades y que los desarrolladores puedan mantener y expandir el código de manera efectiva.

## **OBJETIVOS**

### **1. GENERAL**

**1.1** El objetivo general de este manual es proporcionar una descripción exhaustiva y clara de la funcionalidad y estructura del sistema CYPHERCHAT, detallando cómo se ha implementado y cómo debe ser utilizado y mantenido.

### **2. ESPECÍFICOS**

**2.1.** Objetivo 1: Proveer una explicación detallada de la lógica del programa y la estructura del código, incluyendo la descripción de las clases, métodos y variables globales, para que los desarrolladores comprendan cómo interactúan los diferentes componentes del sistema.

**2.2.** Objetivo 2: Instruir sobre los requisitos de hardware y software necesarios para la correcta ejecución del sistema, así como proporcionar ejemplos de uso y configuraciones recomendadas, asegurando que los usuarios puedan instalar y operar el sistema sin inconvenientes.

## ALCANCES DEL SISTEMA

El objetivo del manual es proporcionar una guía detallada sobre la implementación y funcionamiento del sistema de chat denominado CYPHERCHAT. Este sistema está diseñado para permitir la comunicación segura entre usuarios mediante la encriptación de mensajes, la gestión de contactos y la administración de usuarios.

## ESPECIFICACIÓN TÉCNICA

### ● REQUISITOS DE HARDWARE

- Procesador: Intel Core i3 o superior.
- Memoria RAM: 4 GB mínimo.
- Espacio en Disco: Al menos 500 MB de espacio libre.
- Pantalla: Resolución mínima de 1366x768.
- Conectividad: Conexión a Internet para la comunicación entre usuarios.

### REQUISITOS DE SOFTWARE

- **Sistema Operativo:** Windows 7 o superior, macOS, Linux.
- **Java Runtime Environment (JRE):** Version 8 o superior.
- **IDE:** NetBeans o IntelliJ IDEA para desarrollo.
- **Bibliotecas:** Swing para la interfaz gráfica de usuario

## **DESCRIPCIÓN DE LA SOLUCIÓN**

La solución se pensó y analizó a partir de los requerimientos del enunciado con un enfoque en la seguridad y facilidad de uso. Se desarrolló una aplicación de chat que permite la comunicación en tiempo real entre usuarios con funcionalidades de encriptación de mensajes. Se incluyó una interfaz de usuario intuitiva utilizando Swing y se estructuró el código en clases para facilitar la mantenibilidad y expansión futura.

# LÓGICA DEL PROGRAMA

## ❖ CLASES

```
package ventana;

import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Font;
import java.awt.Image;
import java.awt.Panel;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;
import javax.security.auth.spi.LoginModule;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JCheckBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;
import javax.swing.SwingConstants;
import javax.swing.text.StyledEditorKit;
import static ventana.UsuarioLogic.adminPanel;
```

### ➤ Librerías

- java.awt.BorderLayout: Se usa para el manejo de layouts en la interfaz gráfica.
- java.awt.Color: Se usa para definir colores en la interfaz gráfica.
- java.awt.Font: Se usa para definir fuentes de texto.
- java.awt.Image: Se usa para manejar imágenes.
- java.awt.Panel: Se usa para crear paneles en la interfaz gráfica.
- java.awt.event.ActionEvent: Se usa para manejar eventos de acciones.
- java.awt.event.ActionListener: Se usa para definir escuchadores de eventos de acciones.

- `java.util.ArrayList`: Se usa para manejar listas dinámicas de elementos.
- `javax.swing.ImageIcon`: Se usa para manejar iconos de imágenes.
- `javax.swing.JButton`: Se usa para crear botones.
- `javax.swing.JCheckBox`: Se usa para crear casillas de verificación.
- `javax.swing.JFrame`: Se usa para crear ventanas.
- `javax.swing.JLabel`: Se usa para crear etiquetas de texto.
- `javax.swing.JOptionPane`: Se usa para crear cuadros de diálogo emergentes.
- `javax.swing.JPanel`: Se usa para crear paneles en la interfaz gráfica.
- `javax.swing.JPasswordField`: Se usa para crear campos de texto para contraseñas.
- `javax.swing.JTextField`: Se usa para crear campos de texto.
- `javax.swing.SwingConstants`: Se usa para definir constantes de alineación de componentes.

#### ➤ Variables Globales de las clases

```
String adminID = "202307499";
String adminP = "plIPC1";
```

Clase ventana. Las variables globales en esta clase se utilizan principalmente para almacenar credenciales de administrador que se usan para iniciar sesión y gestionar los permisos en la aplicación.

```
public class UsuarioOB {
    public static List<Usuarios> usuario = new ArrayList<>();
```

Clase Usuarios. Se crea de manera global un arraylist en el cual se almacenaran los nuevos usuarios, los que se vayan registrando.

#### ➤ Función Main

La función main se utiliza para crear una instancia de la clase Ventana y hacerla visible, iniciando así la aplicación.


```

public static void main(String[] args) {
    Ventana vl = new Ventana();


    vl.setVisible(true);
}

```

## ➤ Procedimientos, métodos y Funciones utilizadas

- 121  `private void sendMessage() {`

Clase ChatWindow. Envía el mensaje escrito por el usuario. Actualmente, encripta el mensaje antes de enviarlo, agrega una marca de tiempo, lo añade a la lista de mensajes y limpia el campo de entrada del mensaje.

- 133  `private void loadMatrixA() {`

Clase ChatWindow. Carga la matriz A desde un archivo de texto seleccionado por el usuario. Lee el archivo línea por línea, convierte los elementos a enteros y los almacena en una matriz.

- 

```

public String getContra() {
    return contra;
}


public void setContra(String contra) {
    this.contra = contra;
}

```

Clase Usuarios. Getters And Setters. Obtienen el código, nombre, apellido, teléfono, edad, género de la persona que se registra.

- 

```

9  public boolean insertar(Usuarios nuevoUsuario) {
10     return usuario.add(nuevoUsuario);
11 }

```

Clase UsuarioOB. Inserta un nuevo usuario en la lista.



- `13` `public boolean modificar(Usuarios usuarioModificado) {...9 lines }`

Clase UsuarioOB. Modifica un usuario existente en la lista.

- `23` `public boolean eliminar(String codigo) {`

Clase UsuarioOB. Elimina un usuario de la lista basado en su código.

- `27` `public Usuarios obtener(String codigo) {`

Clase UsuarioOB. Obtiene un usuario de la lista basado en su código.

- `36` `public int buscar(String codigo) {`

Clase UsuarioOB. Busca el índice de un usuario en la lista basado en su código.

- `43` `public static boolean autentificar(String codigo, String contra) {`

Clase UsuarioLogic. Autentifica a un usuario comparando su código y contraseña.