



TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE TLAXIACO

SEGURIDAD Y VIRTUALIZACIÓN

Nombre de los Integrantes de equipo:

Edwin López Santiago

Yanet González García

No. Control

21620123

21620273

Tema:

Práctica 1: Contraseñas y Certificados

Docente:

Ing. Osorio Salinas Edward

Carrera:

Ingeniería en Sistemas Computacionales

Grupo: 7US

Tlaxiaco, Oaxaca. A 26 de Agosto de 2024.



Índice

Índice	2
Introducción	4
Captura del código completo	5
CÓDIGO IMPLEMENTADO EN EL PROGRAMA	6
FUNCIONAMIENTO DEL CÓDIGO	7
EJECUCIÓN DEL PROGRAMA	8
2.- Crea un programa que me recomiende una contraseña segura. La contraseña debe cumplir con los criterios de la instrucción:	9
CAPTURA DEL CÓDIGO COMPLETO	10
CÓDIGO IMPLEMENTADO EN EL PROGRAMA	11
FUNCIONAMIENTO DEL CÓDIGO	12
EJECUCIÓN DEL PROGRAMA	13
3. Crea un certificado SSH, clave pública y clave privada, añade el certificado SSH a tu cuenta de GitHub y realiza un git clone de un repositorio nuevo utilizando la ruta SSH del repositorio.....	15
4. Crea un certificado SSL autofirmado con una validez de 365 días y añádelo a un servidor web local. Realiza una petición GET al servidor web local utilizando curl y muestra el certificado SSL.....	20
Investiga y describe los siguientes conceptos:	23
Contraseña	23
Certificado digital	23
Firma digital	23
Cifrado simétrico	24
Cifrado asimétrico	24
Hash	24
Encriptación	25
Investiga y describe los siguientes algoritmos de cifrado:	25
AES	25
RSA	25
SHA-256	26
Investiga y describe los siguientes estándares de cifrado:	27



SSL (Secure Sockets Layer)	27
TLS (Transport Layer Security)	27
Investiga y describe los siguientes protocolos de seguridad:.....	28
HTTPS (HyperText Transfer Protocol Secure)	28
SFTP (Secure File Transfer Protocol)	28
SSH (Secure Shell)	28
Conclusión	29
Bibliografía	30



Introducción

En la materia de "Seguridad y Virtualización", es fundamental entender cómo proteger la información mediante contraseñas seguras y certificados digitales. Esta práctica tiene como objetivo que aprendamos a crear programas, en este caso, en Java que validen la seguridad de una contraseña y nos recomienden contraseñas seguras. Además, nos adentraremos en la creación y uso de certificados SSH y SSL, que son esenciales para garantizar conexiones seguras en redes y aplicaciones. A través de esta práctica, no solo estamos aplicando conceptos teóricos, sino que también estamos desarrollando habilidades técnicas clave para asegurar la información en el mundo digital. A diario, usamos contraseñas para acceder a nuestras cuentas y servicios en línea, y confiamos en certificados digitales para asegurar nuestras comunicaciones. Sin embargo, muchas veces no comprendemos completamente cómo estas herramientas nos protegen. De esa manera es por la cual es muy importante realizar esta práctica e investigación sobre el tema de Contraseñas y Certificados para tener más claro cómo proteger nuestros datos.

Desarrollo

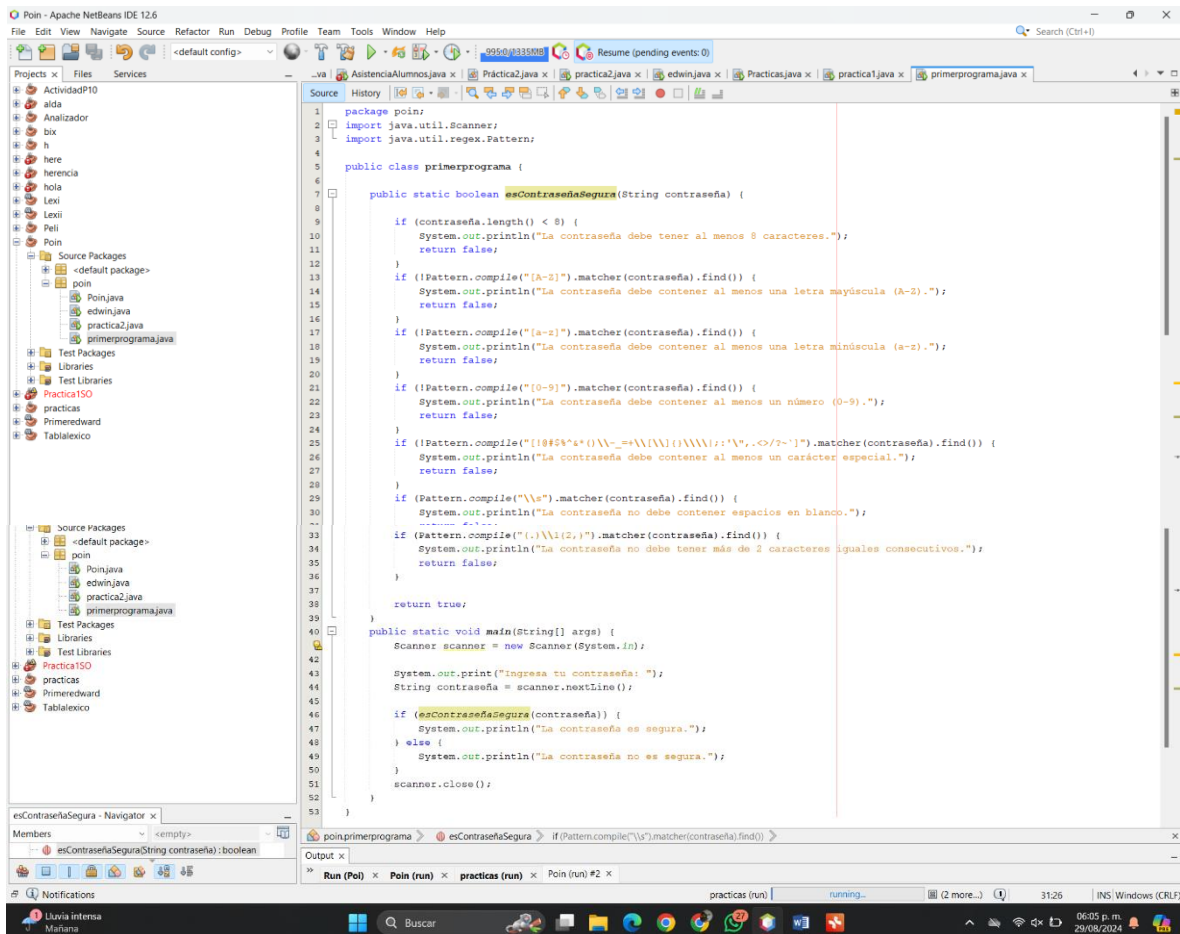
Práctica 1: Contraseñas y Certificados

Instrucción:

1.- Crea un programa en Python que permita al usuario ingresar una contraseña y que valide si la contraseña es segura o no. Una contraseña segura debe cumplir con los siguientes criterios basados en las recomendaciones de Google:

PROGRAMA EN JAVA

Captura del código completo





CÓDIGO IMPLEMENTADO EN EL PROGRAMA

```
package poin;

import java.util.Scanner;

import java.util.regex.Pattern;

public class primerprograma {

    public static boolean esContraseñaSegura(String contraseña) {

        if (contraseña.length() < 8) {

            System.out.println("La contraseña debe tener al menos 8 caracteres.");

            return false;

        }

        if (!Pattern.compile("[A-Z]").matcher(contraseña).find()) {

            System.out.println("La contraseña debe contener al menos una letra mayúscula (A-Z).");

            return false;

        }

        if (!Pattern.compile("[a-z]").matcher(contraseña).find()) {

            System.out.println("La contraseña debe contener al menos una letra minúscula (a-z).");

            return false;

        }

        if (!Pattern.compile("[0-9]").matcher(contraseña).find()) {

            System.out.println("La contraseña debe contener al menos un número (0-9).");

            return false;

        }

        if (!Pattern.compile("[!@#$%^&*()\\-_+=\\\\[\\\\]{}\\\\\\\\\\\\;:'\".,<>/?~`]").matcher(contraseña).find()) {

            System.out.println("La contraseña debe contener al menos un carácter especial.");

            return false;

        }

        if (Pattern.compile("\\s").matcher(contraseña).find()) {

            System.out.println("La contraseña no debe contener espacios en blanco.");

            return false;

        }

        if (Pattern.compile("(.)\\1{2,}").matcher(contraseña).find()) {
```



```
        System.out.println("La contraseña no debe tener más de 2 caracteres iguales consecutivos.");
        return false;
    }

    return true;
}

public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);

    System.out.print("Ingresa tu contraseña: ");
    String contraseña = scanner.nextLine();

    if (esContraseñaSegura(contraseña)) {
        System.out.println("La contraseña es segura.");
    } else {
        System.out.println("La contraseña no es segura.");
    }
    scanner.close();
}
}
```

FUNCIONAMIENTO DEL CÓDIGO

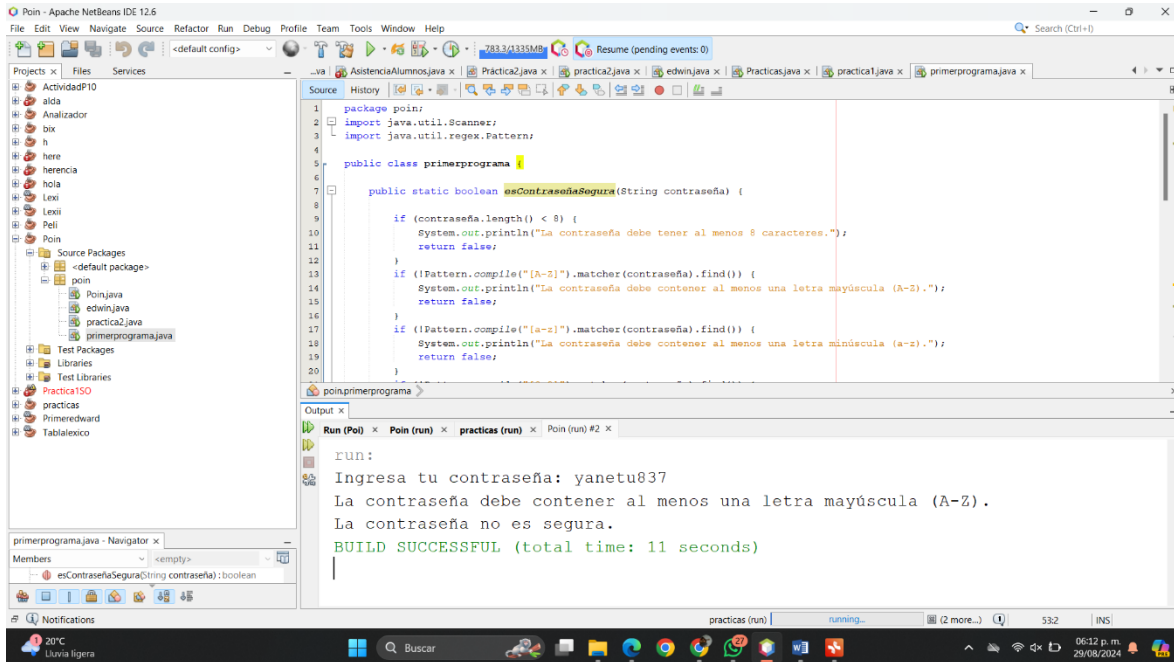
Este código en Java pide al usuario ingresar una contraseña y que valide si la contraseña es segura o no.

- Verifica si le hace falta algún carácter (!, @, #, \$, %, ^, &, *, (,), - , _ , =, +, [,] , { , } , | , \ , ; , ' , " , , , < , > , / , ? , ~ , `).
- Tiene al menos 8 caracteres.
- Contiene letras mayúsculas, minúsculas, números y caracteres especiales.
- No tiene espacios ni tres o más caracteres consecutivos iguales.

Si la contraseña es segura, el programa lo indica. Si no lo es, indica que no es segura.

EJECUCIÓN DEL PROGRAMA

1. Se ingresó una contraseña cualquiera, entonces nos indica que no es muy segura. La contraseña debe contener al menos una letra mayúscula (A-Z).



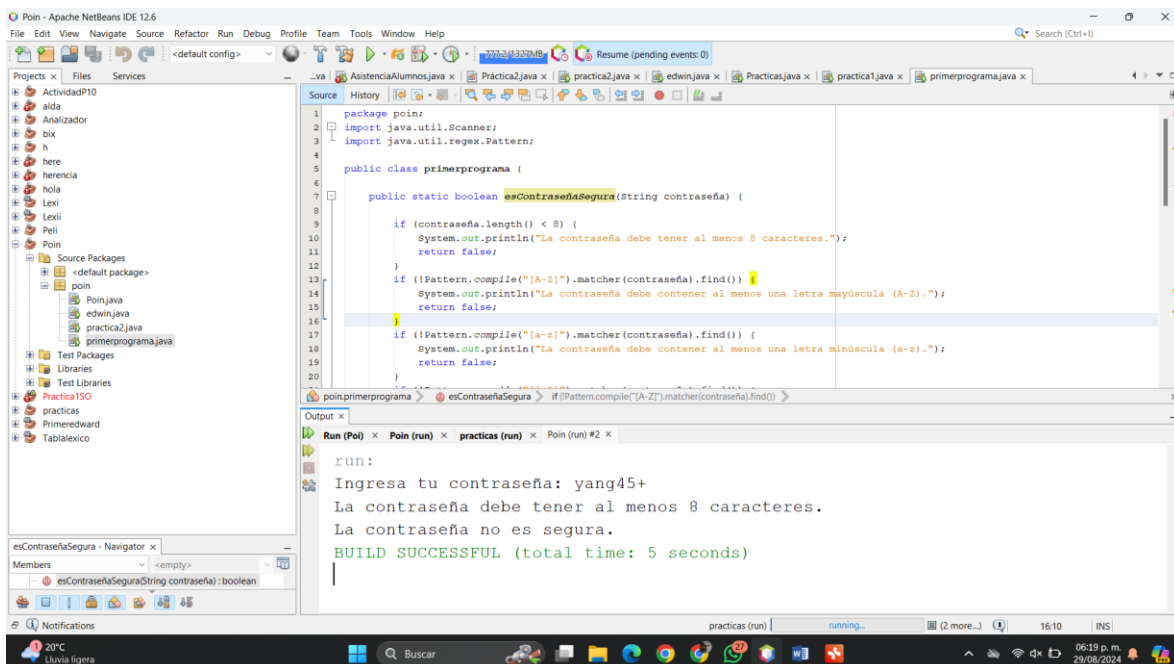
La imagen muestra la interfaz de NetBeans IDE 12.6. En el editor de código, se ve el archivo `primerprograma.java` con el siguiente código:

```
1 package poin;
2 import java.util.Scanner;
3 import java.util.regex.Pattern;
4
5 public class primerprograma {
6
7     public static boolean esContraseñaSegura(String contraseña) {
8
9         if (contraseña.length() < 8) {
10             System.out.println("La contraseña debe tener al menos 8 caracteres.");
11             return false;
12         }
13         if (!Pattern.compile("[A-Z]").matcher(contraseña).find()) {
14             System.out.println("La contraseña debe contener al menos una letra mayúscula (A-Z).");
15             return false;
16         }
17         if (!Pattern.compile("[a-z]").matcher(contraseña).find()) {
18             System.out.println("La contraseña debe contener al menos una letra minúscula (a-z).");
19             return false;
20         }
21     }
22 }
```

En la ventana de salida (Output), se muestra el resultado de la ejecución:

```
run:
Ingresa tu contraseña: yanetu837
La contraseña debe tener al menos 8 caracteres.
La contraseña no es segura.
BUILD SUCCESSFUL (total time: 11 seconds)
```

2. La contraseña debe tener al menos 8 caracteres. Si se ingresa menos de 8 nos indicará que la contraseña no es segura. Así como se muestra a continuación



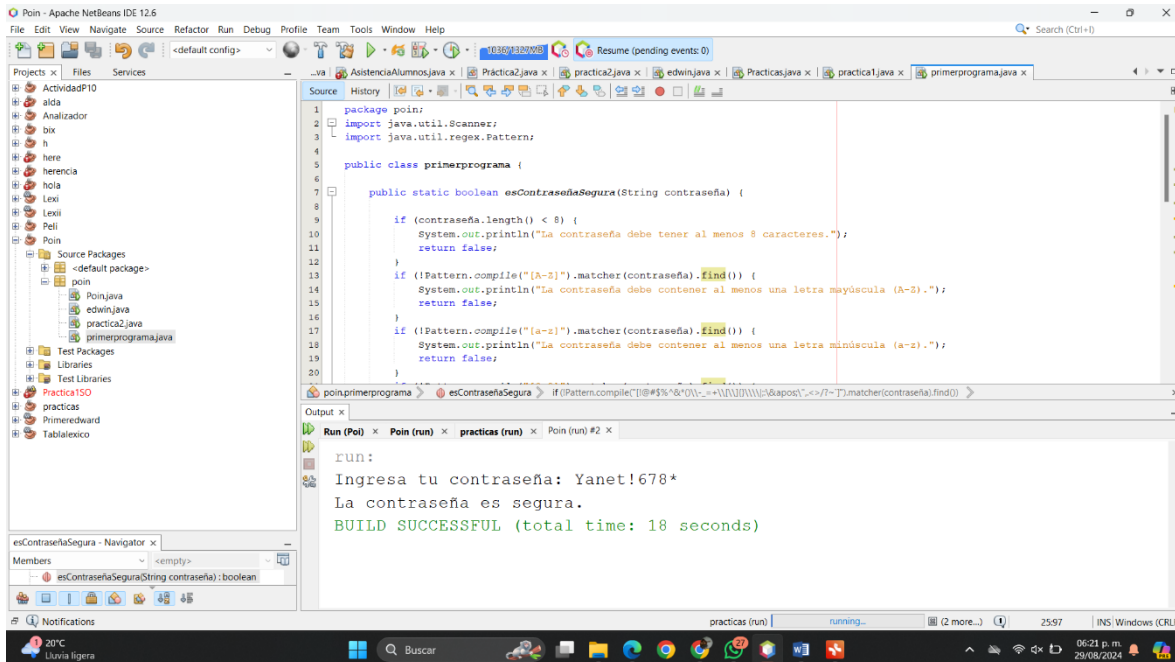
La imagen muestra la interfaz de NetBeans IDE 12.6. En el editor de código, se ve el archivo `primerprograma.java` con el siguiente código:

```
1 package poin;
2 import java.util.Scanner;
3 import java.util.regex.Pattern;
4
5 public class primerprograma {
6
7     public static boolean esContraseñaSegura(String contraseña) {
8
9         if (contraseña.length() < 8) {
10             System.out.println("La contraseña debe tener al menos 8 caracteres.");
11             return false;
12         }
13         if (!Pattern.compile("[A-Z]").matcher(contraseña).find()) {
14             System.out.println("La contraseña debe contener al menos una letra mayúscula (A-Z).");
15             return false;
16         }
17         if (!Pattern.compile("[a-z]").matcher(contraseña).find()) {
18             System.out.println("La contraseña debe contener al menos una letra minúscula (a-z).");
19             return false;
20         }
21     }
22 }
```

En la ventana de salida (Output), se muestra el resultado de la ejecución:

```
run:
Ingresa tu contraseña: yang45+
La contraseña debe tener al menos 8 caracteres.
La contraseña no es segura.
BUILD SUCCESSFUL (total time: 5 seconds)
```


3. Cuando ingresamos una contraseña compleja como la siguiente simplemente el programa nos dirá que es segura.



```
1 package poin;
2 import java.util.Scanner;
3 import java.util.regex.Pattern;
4
5 public class primesprograma {
6
7     public static boolean esContraseñaSegura(String contraseña) {
8
9         if (contraseña.length() < 8) {
10             System.out.println("La contraseña debe tener al menos 8 caracteres.");
11             return false;
12         }
13         if (!Pattern.compile("[A-Z]").matcher(contraseña).find()) {
14             System.out.println("La contraseña debe contener al menos una letra mayúscula (A-Z).");
15             return false;
16         }
17         if (!Pattern.compile("[a-z]").matcher(contraseña).find()) {
18             System.out.println("La contraseña debe contener al menos una letra minúscula (a-z).");
19             return false;
20         }
21     }
22 }
```

Run (Poi) x Poin (run) x practicas (run) x Poin (run) #2 x

run:

Ingresa tu contraseña: Yanet!678*

La contraseña es segura.

BUILD SUCCESSFUL (total time: 18 seconds)

2.- Crea un programa que me recomiende una contraseña segura. La contraseña debe cumplir con los criterios de la instrucción:

- ❖ Tener al menos 8 caracteres.
- ❖ Tener al menos una letra mayúscula (A-Z).
- ❖ Tener al menos una letra minúscula (a-z).
- ❖ Tener al menos un número (0-9).
- ❖ Tener al menos un carácter especial (!, @, #, \$, %, ^, &, *, (,), -, _, =, +, [,], {, }, |, \, ;, :, ', ", ,, ., <, >, /, ?, ~, `).
- ❖ No debe contener espacios en blanco.
- ❖ No debe tener más de 2 caracteres iguales consecutivos.
- ❖ Si la contraseña cumple con los criterios, el programa deberá mostrar un mensaje indicando que la contraseña es segura, de lo contrario, deberá mostrar un mensaje indicando que la contraseña no es segura.

CAPTURA DEL CÓDIGO COMPLETO

The image shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The toolbar contains icons for various development actions. The left sidebar displays the project structure for 'SegundoPrograma', showing source packages, test packages, and libraries. The main editor window shows the source code for 'SegundoPrograma.java'. The code includes imports for Random, Scanner, and Pattern. The 'esContraseñaSegura' method checks password length, character types (uppercase, lowercase, digits, special characters), and consecutive characters. The 'generarContraseñaSegura' method generates a random password using a StringBuilder. The 'main' method prompts the user for a password and checks if it is secure. The bottom status bar shows the system is running on Windows with the date 06/11 p.m. 29/08/2024.

```
import java.util.Random;
import java.util.Scanner;
import java.util.regex.Pattern;

public class SegundoPrograma {

    public static boolean esContraseñaSegura(String contraseña) {
        if (contraseña.length() < 8) {
            System.out.println("La contraseña debe tener al menos 8 caracteres.");
            return false;
        }

        if (!Pattern.compile("[A-Z]").matcher(contraseña).find()) {
            System.out.println("La contraseña debe contener al menos una letra mayúscula (A-Z).");
            return false;
        }

        if (!Pattern.compile("[a-z]").matcher(contraseña).find()) {
            System.out.println("La contraseña debe contener al menos una letra minúscula (a-z).");
            return false;
        }

        if (!Pattern.compile("[0-9]").matcher(contraseña).find()) {
            System.out.println("La contraseña debe contener al menos un número (0-9).");
            return false;
        }

        if (!Pattern.compile("[!@#%&*()\\-_=+\\\\[\\\\]{}\\\\\\\\\\\\;:;\\\\\"',.<.>/?~`]").matcher(contraseña).find()) {
            System.out.println("La contraseña debe contener al menos un carácter especial.");
            return false;
        }

        if (Pattern.compile("\\s").matcher(contraseña).find()) {
            System.out.println("La contraseña no debe contener espacios en blanco.");
            return false;
        }

        if (Pattern.compile("(.)\\1{2,}").matcher(contraseña).find()) {
            System.out.println("La contraseña no debe tener más de 2 caracteres iguales consecutivos.");
            return false;
        }

        return true;
    }

    public static String generarContraseñaSegura() {
        String mayusculas = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
        String minusculas = "abcdefghijklmnopqrstuvwxyz";
        String numeros = "0123456789";
        String caracteresEspeciales = "!@#%&*()\\-_=+[]{}\\\\\\\\\\\\;:;\\\\\"',.<.>/?~`";

        String todosLosCaracteres = mayusculas + minusculas + numeros + caracteresEspeciales;
        Random random = new Random();

        StringBuilder contraseña = new StringBuilder();

        contraseña.append(mayusculas.charAt(random.nextInt(mayusculas.length())));
        contraseña.append(minusculas.charAt(random.nextInt(minusculas.length())));
        contraseña.append(numeros.charAt(random.nextInt(numeros.length())));
        contraseña.append(caracteresEspeciales.charAt(random.nextInt(caracteresEspeciales.length())));
    }
}
```

```
for (int i = 4; i < 8; i++) {
    contraseña.append(todosLosCaracteres.charAt(random.nextInt(todosLosCaracteres.length())));
}

while (contraseña.length() < 12) {
    char nuevoCaracter = todosLosCaracteres.charAt(random.nextInt(todosLosCaracteres.length()));
    if (contraseña.length() < 2 || (nuevoCaracter != contraseña.charAt(contraseña.length() - 1)
        || nuevoCaracter != contraseña.charAt(contraseña.length() - 2))) {
        contraseña.append(nuevoCaracter);
    }
}

return contraseña.toString();
}

public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);

    System.out.print("Ingresa tu contraseña: ");
    String contraseña = scanner.nextLine();

    if (esContraseñaSegura(contraseña)) {
        System.out.println("La contraseña es segura.");
    } else {
        System.out.println("La contraseña no es segura.");
        String contraseñaRecomendada = generarContraseñaSegura();
        System.out.println("Te recomendamos la siguiente contraseña segura: " + contraseñaRecomendada);
    }

    scanner.close();
}
```

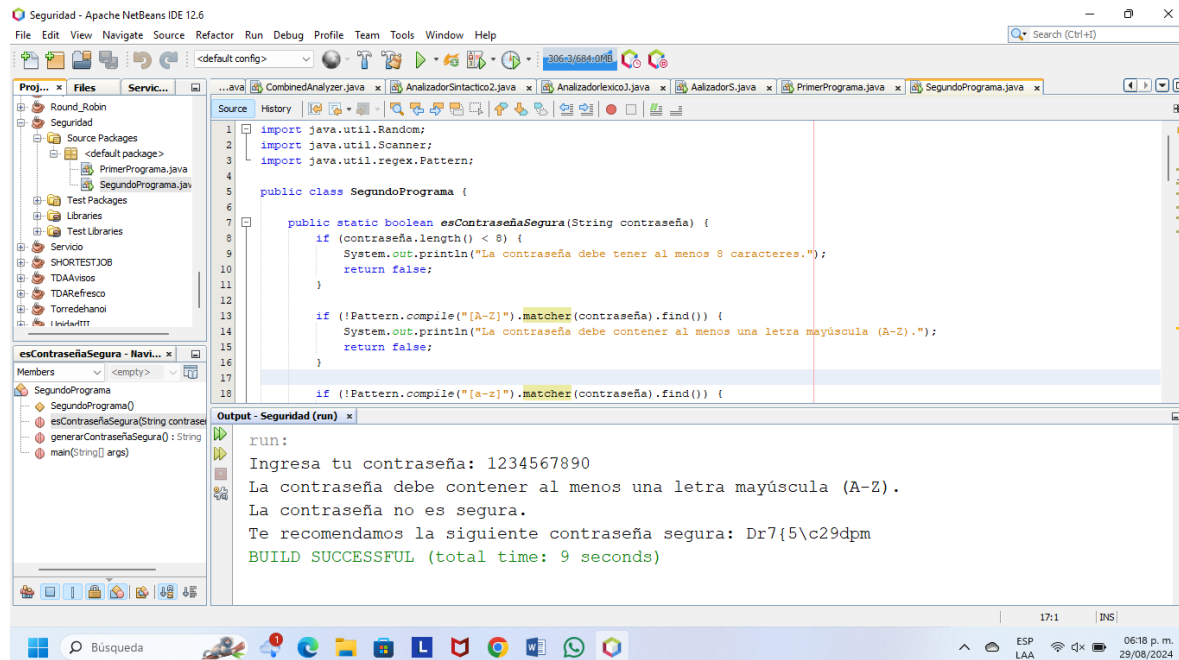
FUNCIONAMIENTO DEL CÓDIGO

Este código en Java pide al usuario que ingrese una contraseña y verifica si es segura. Una contraseña se considera segura si:

- Tener al menos 8 caracteres.
- Tener al menos una letra mayúscula (A-Z).
- Tener al menos una letra minúscula (a-z).
- Tener al menos un número (0-9).
- Tener al menos un carácter especial (!, @, #, \$, %, ^, &, *, (,), -, _ , =, +, [,], {, }, |, \, ;, :, ', ", ,, ., <, >, /, ?, ~, `).
- No debe contener espacios en blanco.
- No debe tener más de 2 caracteres iguales consecutivos.
- Si la contraseña cumple con los criterios, el programa deberá mostrar un mensaje indicando que la contraseña es segura, de lo contrario, deberá mostrar un mensaje indicando que la contraseña no es segura.

EJECUCIÓN DEL PROGRAMA

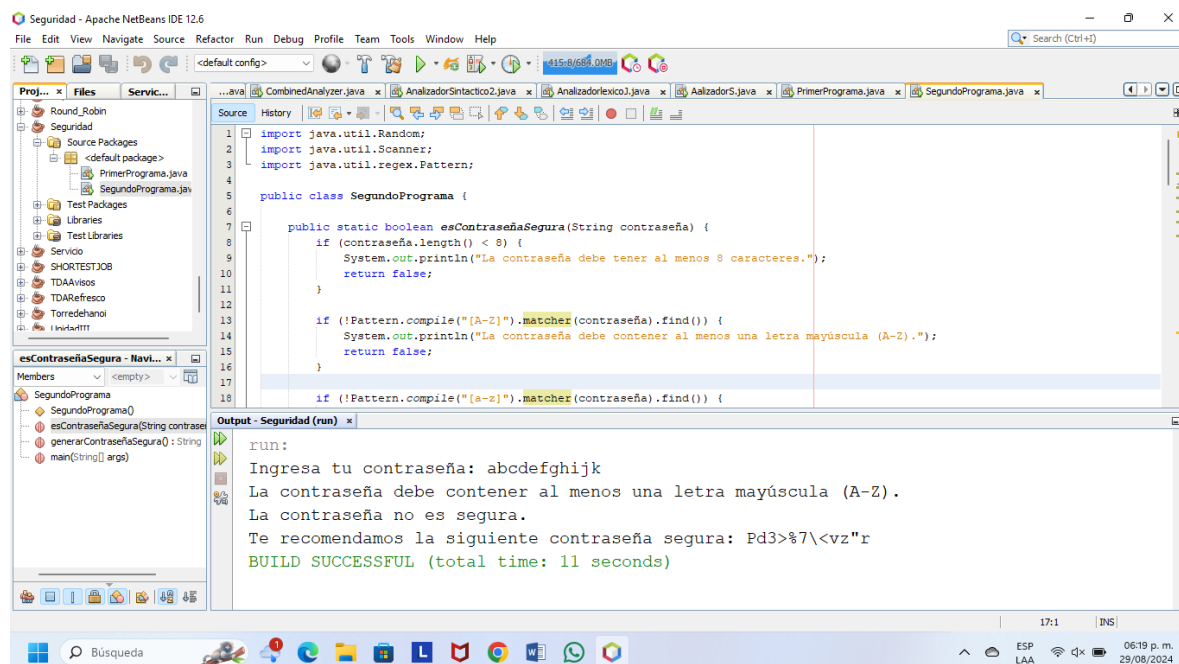
1. El programa no puede aceptar que se ingresen los números consecutivos y mandara una recomendación como se ve a continuación.



```
1 import java.util.Random;
2 import java.util.Scanner;
3 import java.util.regex.Pattern;
4
5 public class SegundoPrograma {
6
7     public static boolean esContraseñaSegura(String contraseña) {
8         if (contraseña.length() < 8) {
9             System.out.println("La contraseña debe tener al menos 8 caracteres.");
10            return false;
11        }
12
13        if (!Pattern.compile("[A-Z]").matcher(contraseña).find()) {
14            System.out.println("La contraseña debe contener al menos una letra mayúscula (A-Z).");
15            return false;
16        }
17
18        if (!Pattern.compile("[a-z]").matcher(contraseña).find()) {
19            System.out.println("La contraseña debe contener al menos una letra minúscula (a-z).");
20            return false;
21        }
22
23        return true;
24    }
25
26    public static void main(String[] args) {
27        Scanner scanner = new Scanner(System.in);
28        System.out.println("Ingresa tu contraseña:");
29        String contraseña = scanner.nextLine();
30        boolean esSegura = esContraseñaSegura(contraseña);
31        if (!esSegura) {
32            System.out.println("La contraseña no es segura.");
33            System.out.println("Te recomendamos la siguiente contraseña segura: Dr7{5\c29dpm");
34        } else {
35            System.out.println("¡Felicidades! Tu contraseña es segura.");
36        }
37    }
38}
```

run:
Ingresa tu contraseña: 1234567890
La contraseña debe contener al menos una letra mayúscula (A-Z).
La contraseña no es segura.
Te recomendamos la siguiente contraseña segura: Dr7{5\c29dpm
BUILD SUCCESSFUL (total time: 9 seconds)

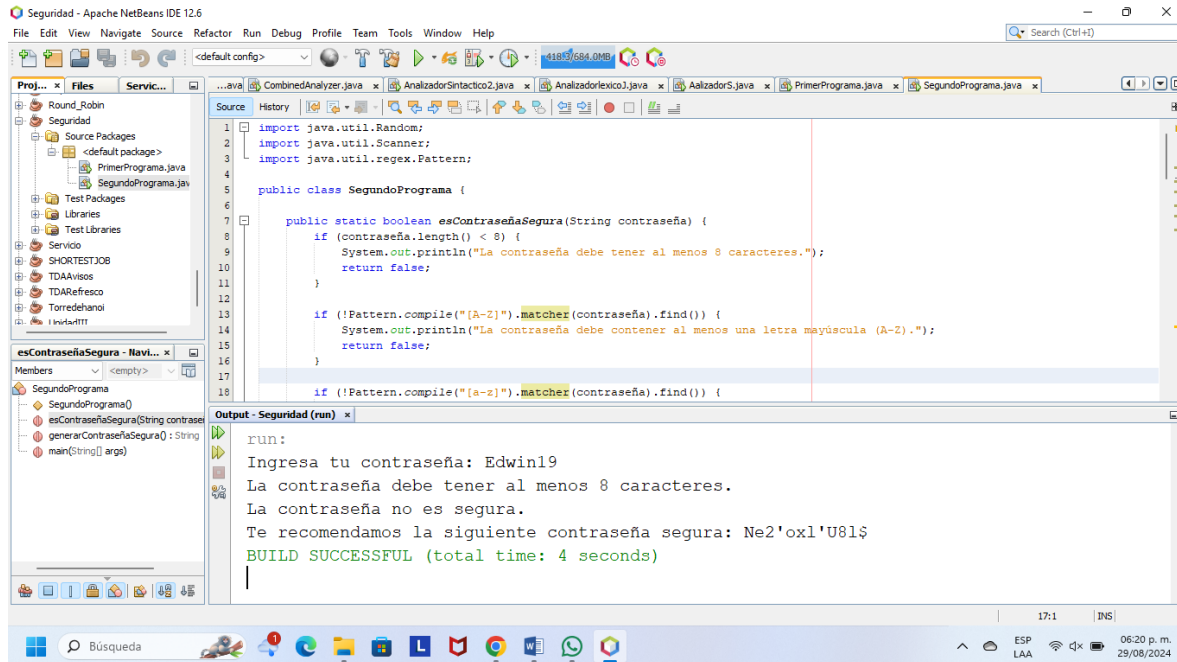
2. El programa no puede aceptar que se ingresen el orden del abecedario en minúscula o mayúsculas y nos mandara una recomendación como se ve a continuación.



```
1 import java.util.Random;
2 import java.util.Scanner;
3 import java.util.regex.Pattern;
4
5 public class SegundoPrograma {
6
7     public static boolean esContraseñaSegura(String contraseña) {
8         if (contraseña.length() < 8) {
9             System.out.println("La contraseña debe tener al menos 8 caracteres.");
10            return false;
11        }
12
13        if (!Pattern.compile("[A-Z]").matcher(contraseña).find()) {
14            System.out.println("La contraseña debe contener al menos una letra mayúscula (A-Z).");
15            return false;
16        }
17
18        if (!Pattern.compile("[a-z]").matcher(contraseña).find()) {
19            System.out.println("La contraseña debe contener al menos una letra minúscula (a-z).");
20            return false;
21        }
22
23        return true;
24    }
25
26    public static void main(String[] args) {
27        Scanner scanner = new Scanner(System.in);
28        System.out.println("Ingresa tu contraseña:");
29        String contraseña = scanner.nextLine();
30        boolean esSegura = esContraseñaSegura(contraseña);
31        if (!esSegura) {
32            System.out.println("La contraseña no es segura.");
33            System.out.println("Te recomendamos la siguiente contraseña segura: Pd3>%7\<vz"r");
34        } else {
35            System.out.println("¡Felicidades! Tu contraseña es segura.");
36        }
37    }
38}
```

run:
Ingresa tu contraseña: abcdefghijkl
La contraseña debe contener al menos una letra mayúscula (A-Z).
La contraseña no es segura.
Te recomendamos la siguiente contraseña segura: Pd3>%7\<vz"r
BUILD SUCCESSFUL (total time: 11 seconds)

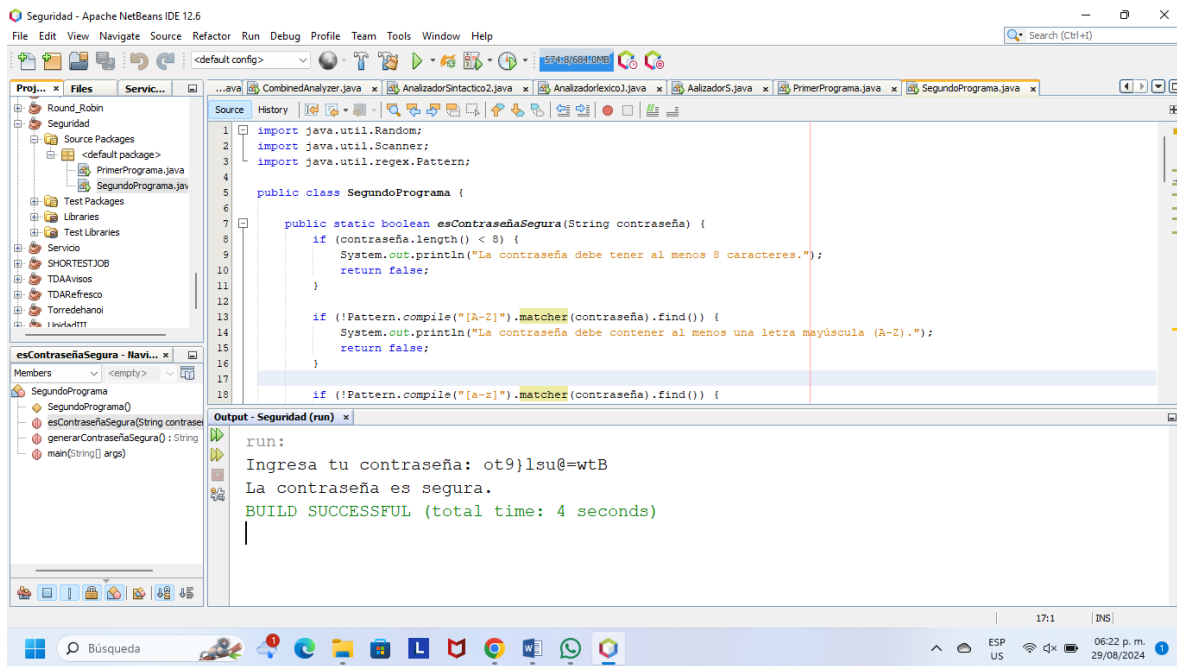
3. El programa no puede aceptar caracteres menos de 8 y que tenga caracteres como "!@#\$%^&*()-_+=+[]{}|\\;:\",<.>/?~ y mandara una recomendación como se ve a continuación.



```
1 import java.util.Random;
2 import java.util.Scanner;
3 import java.util.regex.Pattern;
4
5 public class SegundoPrograma {
6
7     public static boolean esContraseñaSegura(String contraseña) {
8         if (contraseña.length() < 8) {
9             System.out.println("La contraseña debe tener al menos 8 caracteres.");
10            return false;
11        }
12
13        if (!Pattern.compile("[A-Z]").matcher(contraseña).find()) {
14            System.out.println("La contraseña debe contener al menos una letra mayúscula (A-Z).");
15            return false;
16        }
17
18        if (!Pattern.compile("[a-z]").matcher(contraseña).find()) {
19            System.out.println("La contraseña debe contener al menos una letra minúscula (a-z).");
20            return false;
21        }
22
23        return true;
24    }
25
26    public static void main(String[] args) {
27        Scanner scanner = new Scanner(System.in);
28        System.out.println("Ingresa tu contraseña:");
29        String contraseña = scanner.nextLine();
30        boolean esSegura = esContraseñaSegura(contraseña);
31        if (esSegura) {
32            System.out.println("La contraseña es segura.");
33        } else {
34            System.out.println("La contraseña no es segura. Te recomendamos la siguiente contraseña segura: Ne2'ox1'U81$");
35        }
36    }
37 }
```

run:
Ingresa tu contraseña: Edwin19
La contraseña debe tener al menos 8 caracteres.
La contraseña no es segura.
Te recomendamos la siguiente contraseña segura: Ne2'ox1'U81\$
BUILD SUCCESSFUL (total time: 4 seconds)

4. Cuando ingresamos una contraseña compleja como la siguiente simplemente el programa nos dirá que es correcto.

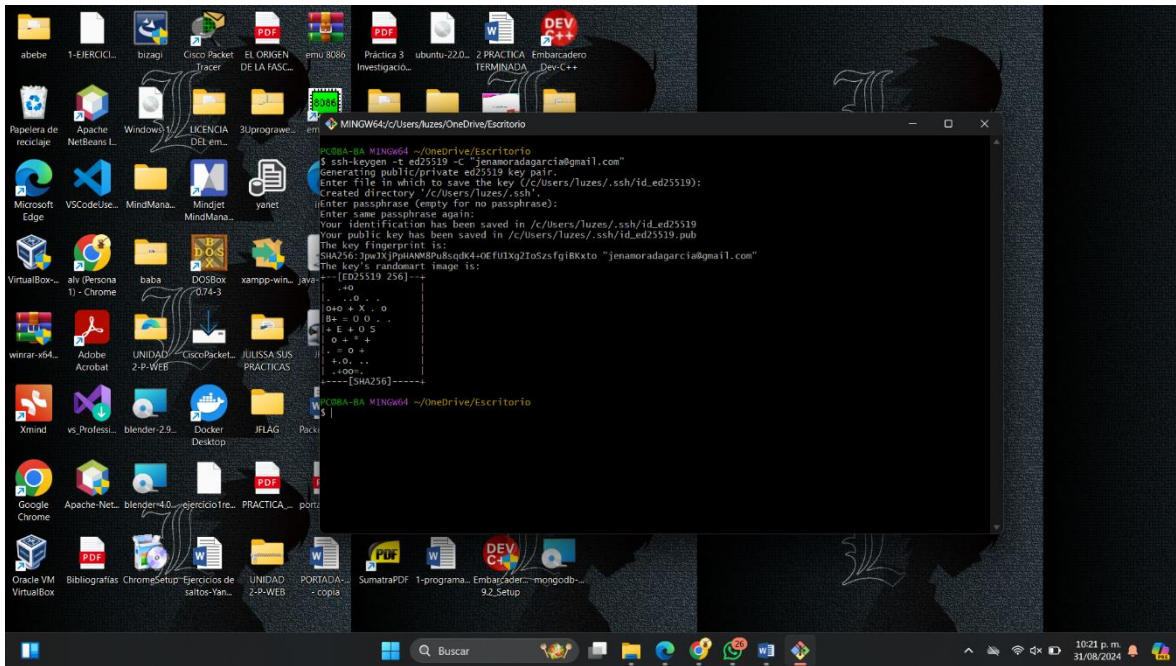


```
1 import java.util.Random;
2 import java.util.Scanner;
3 import java.util.regex.Pattern;
4
5 public class SegundoPrograma {
6
7     public static boolean esContraseñaSegura(String contraseña) {
8         if (contraseña.length() < 8) {
9             System.out.println("La contraseña debe tener al menos 8 caracteres.");
10            return false;
11        }
12
13        if (!Pattern.compile("[A-Z]").matcher(contraseña).find()) {
14            System.out.println("La contraseña debe contener al menos una letra mayúscula (A-Z).");
15            return false;
16        }
17
18        if (!Pattern.compile("[a-z]").matcher(contraseña).find()) {
19            System.out.println("La contraseña debe contener al menos una letra minúscula (a-z).");
20            return false;
21        }
22
23        return true;
24    }
25
26    public static void main(String[] args) {
27        Scanner scanner = new Scanner(System.in);
28        System.out.println("Ingresa tu contraseña:");
29        String contraseña = scanner.nextLine();
30        boolean esSegura = esContraseñaSegura(contraseña);
31        if (esSegura) {
32            System.out.println("La contraseña es segura.");
33        } else {
34            System.out.println("La contraseña no es segura. Te recomendamos la siguiente contraseña segura: Ne2'ox1'U81$");
35        }
36    }
37 }
```

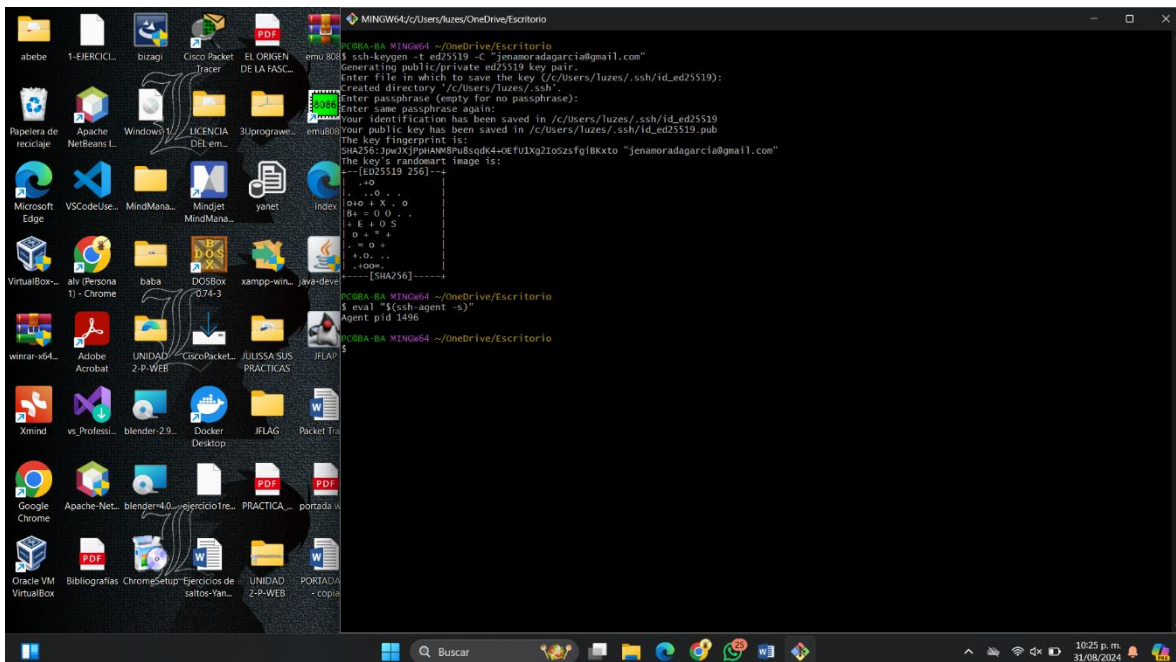
run:
Ingresa tu contraseña: ot9}lsu@=wtB
La contraseña es segura.
BUILD SUCCESSFUL (total time: 4 seconds)

3. Crea un certificado SSH, clave pública y clave privada, añade el certificado SSH a tu cuenta de GitHub y realiza un git clone de un repositorio nuevo utilizando la ruta SSH del repositorio.

1.- Primeramente, nos dirigimos a la terminal en Git Bash. Luego ejecutamos el siguiente comando **ssh-keygen -t ed25519 -C "jenamoradagarcia@gmail.com"** para generar una clave SSH. Y esperar a que se genere lo siguiente.

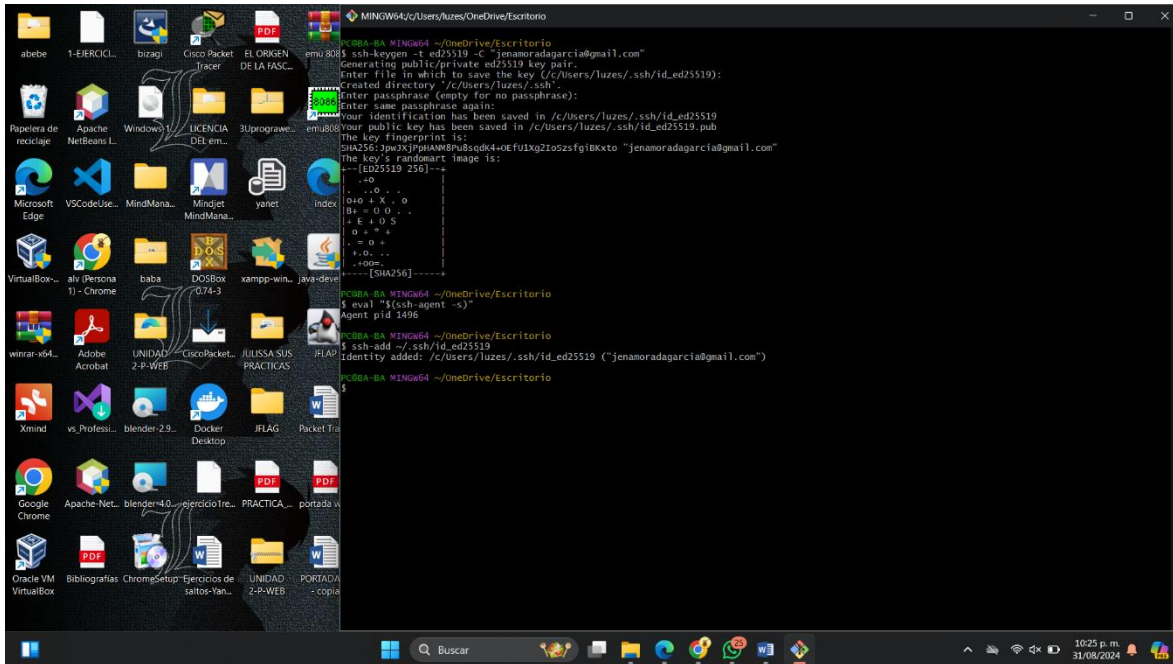


2.- Posteriormente para añadir la clave SSH a nuestro agente SSH, iniciamos el agente SSH con el comando **eval "\$(ssh-agent -s)"**.



3.- De ahí colocar el siguiente comando **ssh-add ~/.ssh/id_ed25519**

Este mensaje que se genera al colocar el comando, confirma que la clave privada especificada ha sido añadida con éxito al agente SSH.



```
PCBBA-BA MINGW64 ~/OneDrive/Escritorio
$ ssh-keygen -t ed25519 -C "jenamoradagarcia@gmail.com"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/c:/Users/luzes/.ssh/id_ed25519):
Created directory /c:/Users/luzes/.ssh.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c:/Users/luzes/.ssh/id_ed25519
Your public key has been saved in /c:/Users/luzes/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:3pw3XjppHAMBp8sQdK4+0EfU1Xg2I0s2sfgiBKxt0 "jenamoradagarcia@gmail.com"
The key's randomart image is:
+--[ED25519 256]--+
..o
..o..
o..o..
o..o..
+ E + O S
o + +
..o +
+..o..
+...+
--=[SHA256]--

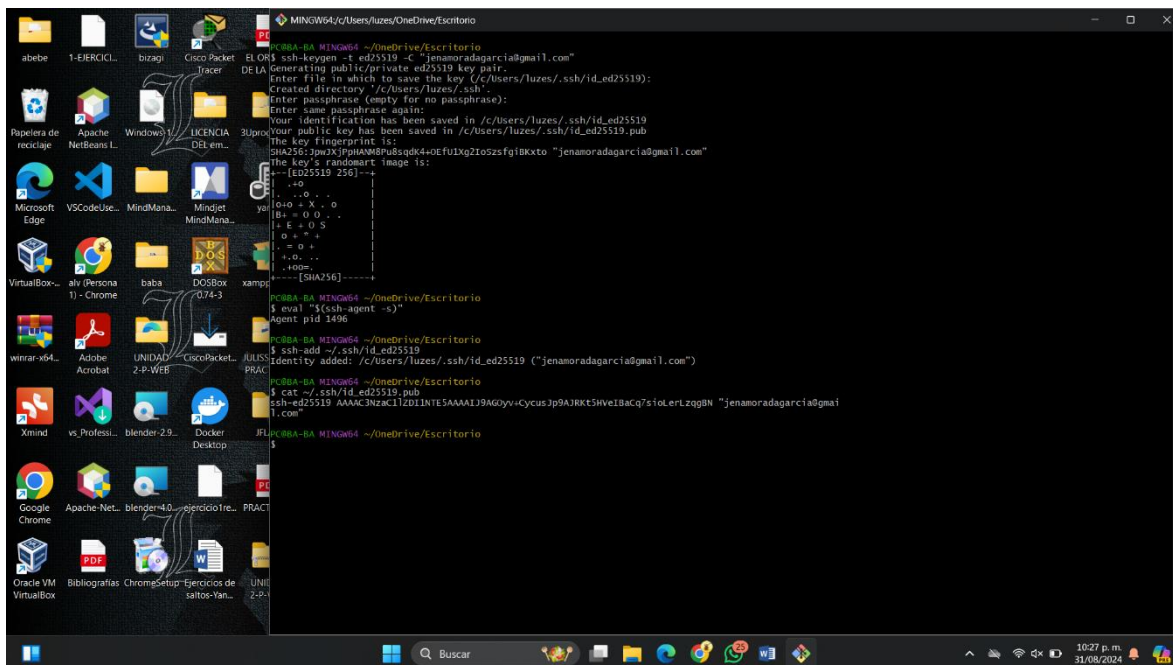
PCBBA-BA MINGW64 ~/OneDrive/Escritorio
$ eval "$(ssh-agent -s)"
Agent pid 1496

PCBBA-BA MINGW64 ~/OneDrive/Escritorio
$ ssh-add ~/.ssh/id_ed25519
Identity added: /c:/Users/luzes/.ssh/id_ed25519 ("jenamoradagarcia@gmail.com")

PCBBA-BA MINGW64 ~/OneDrive/Escritorio
$
```

4.- Después añadimos el comando **cat ~/.ssh/id_ed25519.pub** para obtener nuestra clave SSH, que es en este caso **ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIJ9AG0yv+CycusJp9AJRKt5HVeIBaCq7sioLe rLzqgBN "jenamoradagarcia@gmail.com"**

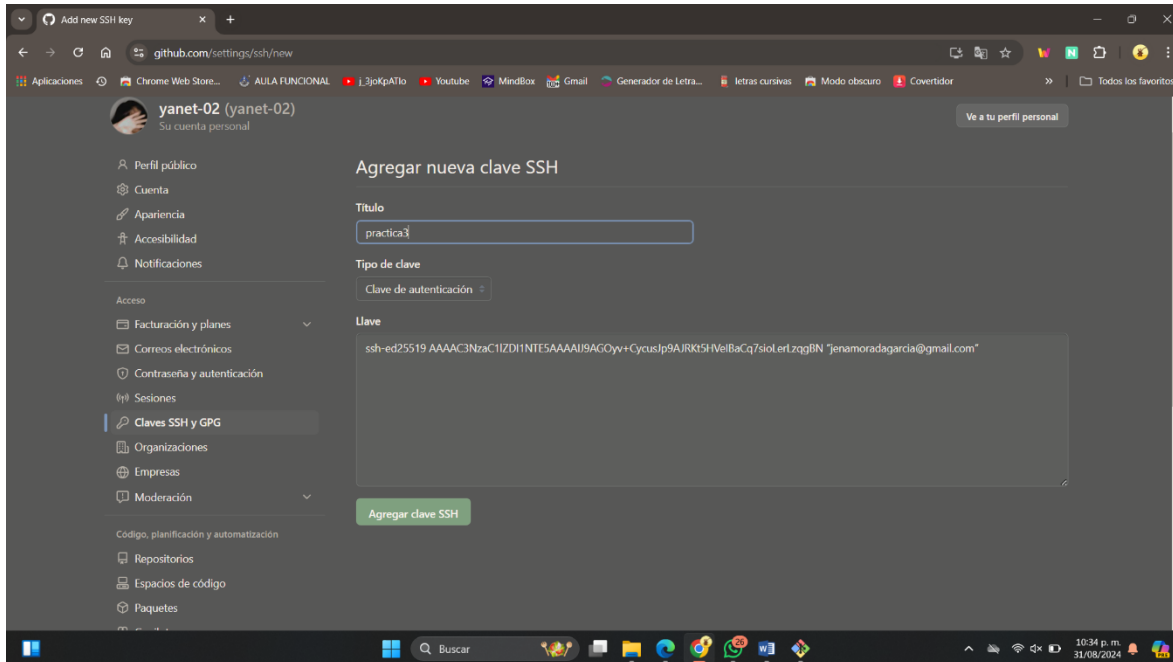
Debe aparecer como se muestra a continuación, con esto se verifica que el paso va correctamente.



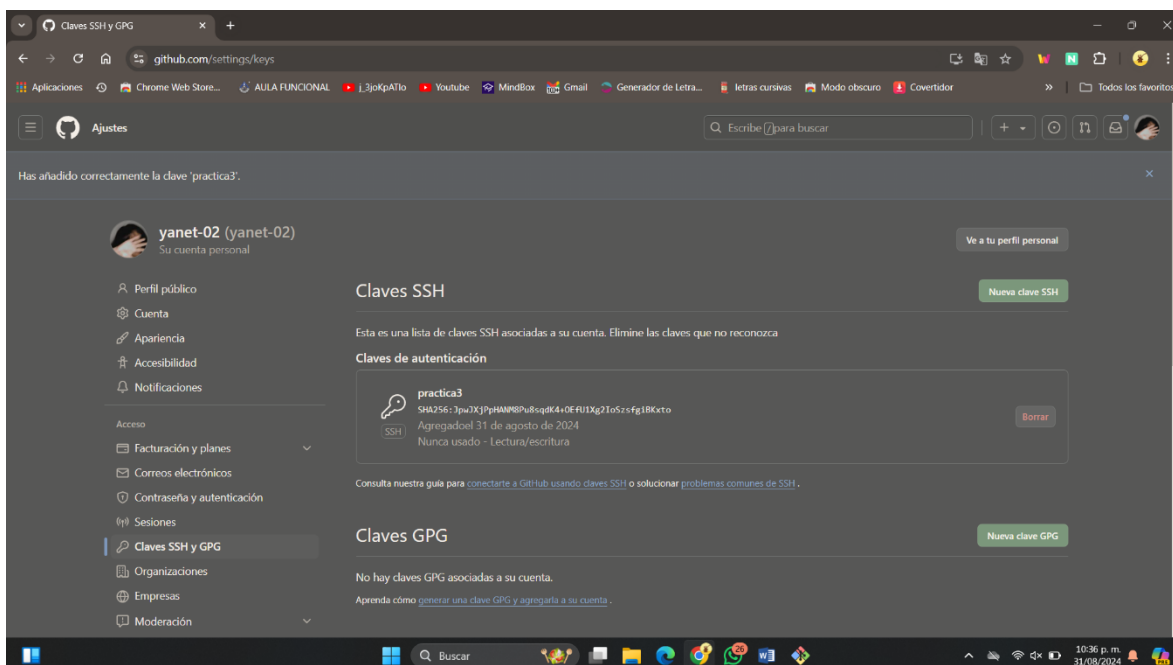
```
PCBBA-BA MINGW64 ~/OneDrive/Escritorio
$ cat ~/.ssh/id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIJ9AG0yv+CycusJp9AJRKt5HVeIBaCq7sioLe rLzqgBN "jenamoradagarcia@gmail.com"

PCBBA-BA MINGW64 ~/OneDrive/Escritorio
$
```

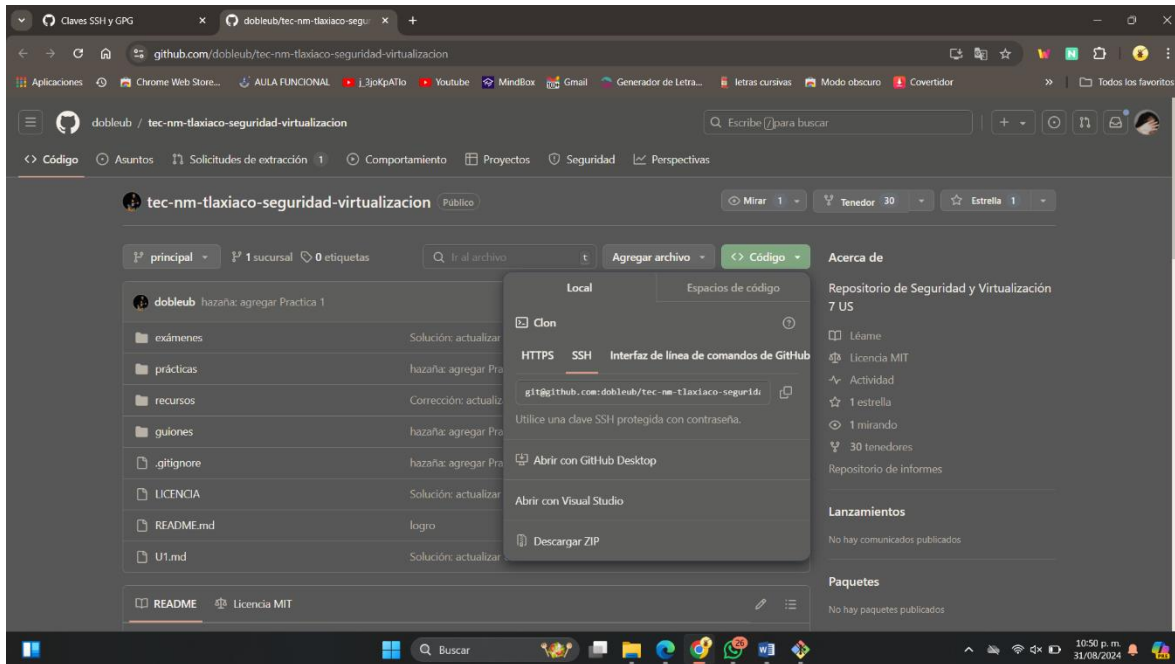

5.- En este paso se debe copiar la clave que se generó anteriormente para poder abrir en configuración del perfil de Git Hub, enseguida dirigimos en Claves SSH y GPG para poder agregar una nueva clave SSH. Se agrega el nombre de algún título que deseamos agregar, en este caso “practica3”, en la parte de Llave, pegar la clave copiada que nos generó en Git Bash. Damos clic en agregar clave SSH.



6.- Al finalizar nos muestra un mensaje que nos dice que la clave se ha añadido correctamente. Así como se visualiza a continuación.

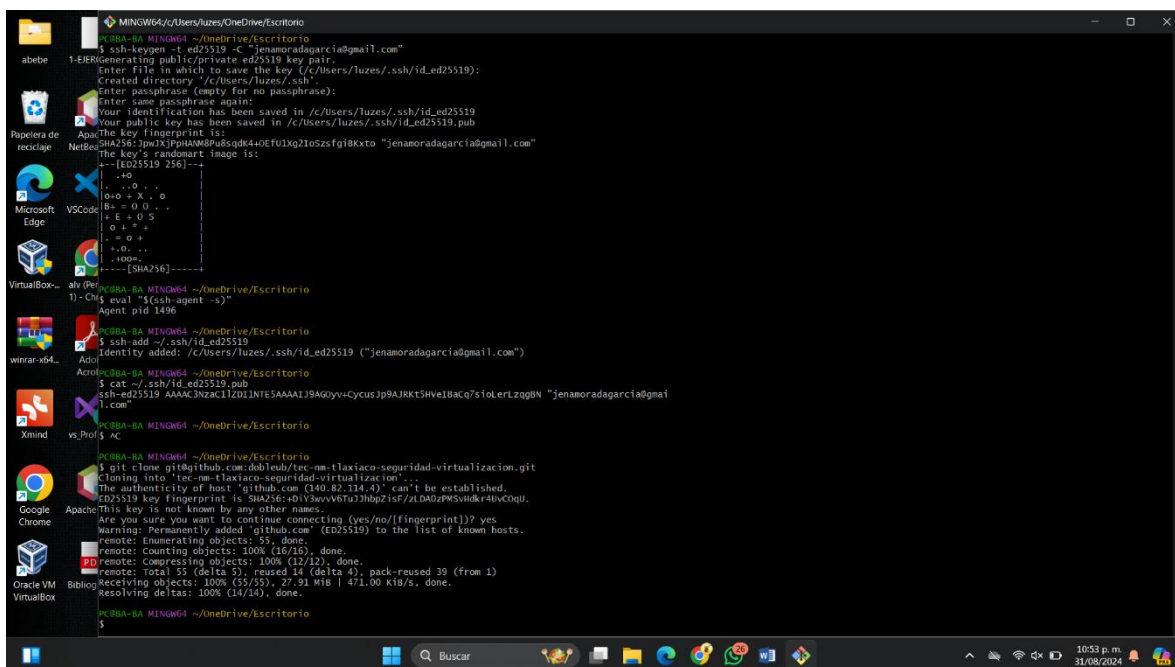


7.- Debemos verificar conectividad para empezar a clonar nuestro repositorio. En este caso, clonar cualquier repositorio que deseamos. Luego damos clic en (Código) y seleccionamos la opción SSH. Copiamos la URL SSH del repositorio.



8.- Posteriormente agregamos el comando git clone junto con el URL SSH.
git clone git@github.com:dobleub/tec-nm-tlaxiaco-seguridad-virtualizacion.git

Esto es para clonar el repositorio y después se puede observar cómo se empieza a clonar el repositorio. Una vez hecho lo dicho, debe quedar de la siguiente manera. Se observa que terminó de clonarse exitosamente el repositorio.

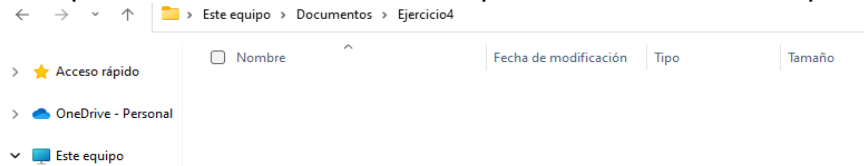




COMANDOS UTILIZADOS

- `ssh-keygen -t ed25519 -C" jenamoradagarcia@gmail.com"`
- `eval "$(ssh-agent -s)"`
- `ssh-add ~/.ssh/id_ed25519`
- `cat ~/.ssh/id_ed25519.pub`
- `git clone git@github.com:doubleub/tec-nm-tlaxiaco-seguridad-virtualizacion.git`

4. Crea un certificado SSL autofirmado con una validez de 365 días y añádelo a un servidor web local. Realiza una petición GET al servidor web local utilizando curl y muestra el certificado SSL.
- 1) Lo primero que hicimos fue crear una carpeta en una dirección que queramos.



- 2) Abrimos la terminal de Openssl para iniciar con el procedimiento.

```
Win64 OpenSSL Command Prompt
OpenSSL 3.3.1 4 Jun 2024 (Library: OpenSSL 3.3.1 4 Jun 2024)
built on: Tue Jun 4 17:39:01 2024 UTC
platform: VC-WIN64A
options: bn(64,64)
compiler: cl /Z7 /Fdssl_static.pdb /Gs0 /GF /Gy /MD /W3 /wd4090 /nologo /O2 -DL_ENDIAN -DOPENSSL_PIC -DOPENSSL_BUILDT
OCK_DEPRECATED_NO_WARNINGS -D"DEBUG" -D_WINSOCK_DEPRECATED_NO_WARNINGS -D_WIN32_WINNT=0x0502
ENGINESSDIR: "C:\Program Files\OpenSSL\lib\engines-3"
MODULESDIR: "C:\Program Files\OpenSSL\lib\ossl-modules"
Seeding source: os-specific
CPUINFO: OPENSSL_ia32cap=0xfffff38f:0xfffff38f:0x18485f6c:0xf3bfa7a9
C:\Users\S2953>
```

- 3) Luego navegaremos para configurar el certificado SSL en la carpeta como se muestra a continuación.

```
C:\Users\S2953>cd "C:\Users\S2953\Documents\Ejercicio4"
```

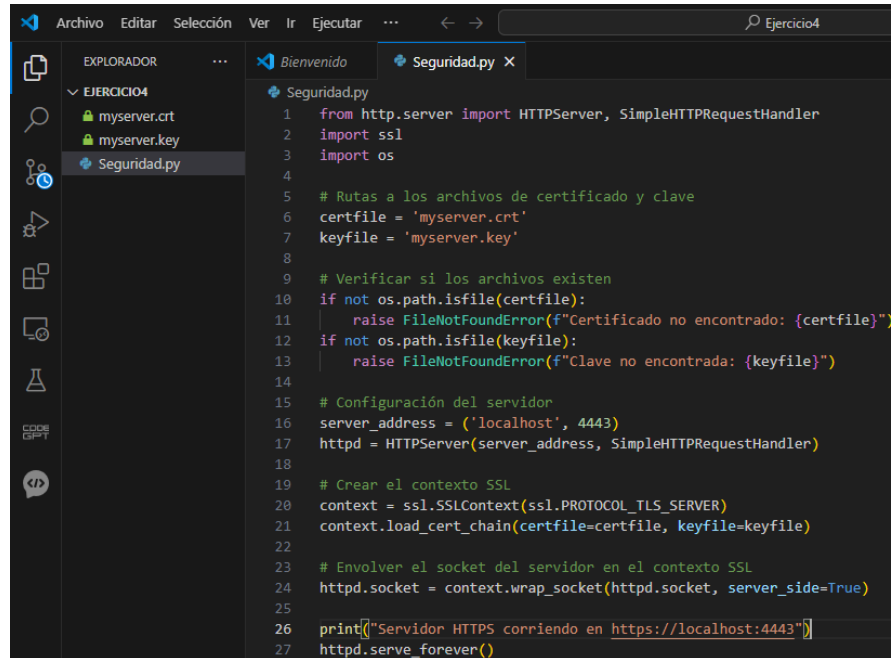
- 4) Una vez pegado la dirección nos permitirá crear el certificado SSL en la carpeta directamente y es aquí donde ya podemos interactuar creando el certificado. Procedemos a generar el Certificado y la clave. Con el siguiente comando **openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout myserver.key -out myserver.crt**

```
C:\Users\S2953\Documents\Ejercicio4>openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout myserver.key -out myserver.crt
```

- 5) Después procedimos a completar varios campos requeridos para identificar a la entidad solicitante del certificado. Estos campos incluyen el código del país, el nombre del estado o provincia, el nombre de la ciudad o localidad, el nombre de la organización o empresa, la unidad organizativa dentro de la entidad, el nombre común, y una dirección de correo electrónico de contacto.

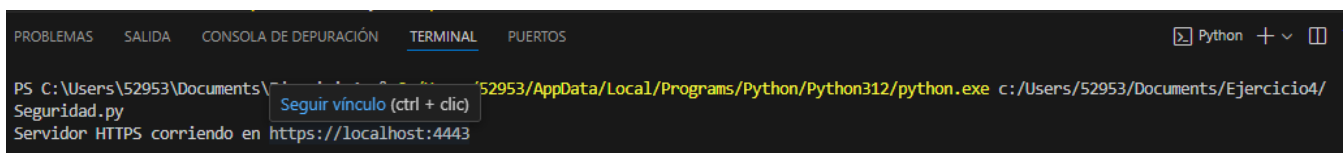
```
Country Name (2 letter code) [AU]:Mx
State or Province Name (full name) [Some-State]:Oaxaca
Locality Name (eg, city) []:Tlaxiaco
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Computer System
Organizational Unit Name (eg, section) []:System Integer
Common Name (e.g. server FQDN or YOUR name) []:Edwin
Email Address []:santiagoowin051@gmail.com
```

- 6) De ahí abrimos Visual Studio Code y creamos un nuevo script con punto python para Verificar la existencia de archivos, si los archivos están en el mismo directorio que el script. También para hacer dicho procedimiento debemos abrir la carpeta para crear el script.



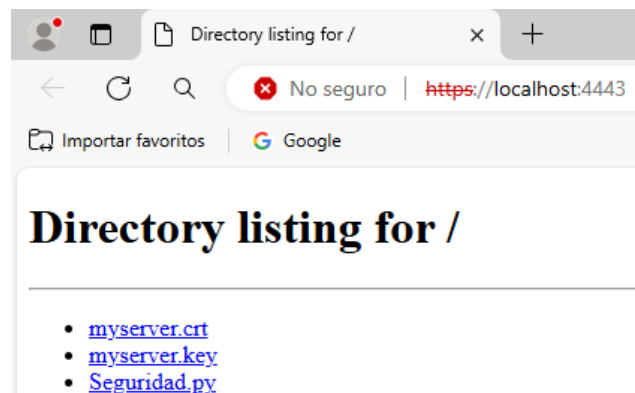
```
1 from http.server import HTTPServer, SimpleHTTPRequestHandler
2 import ssl
3 import os
4
5 # Rutas a los archivos de certificado y clave
6 certfile = 'myserver.crt'
7 keyfile = 'myserver.key'
8
9 # Verificar si los archivos existen
10 if not os.path.isfile(certfile):
11     raise FileNotFoundError(f"Certificado no encontrado: {certfile}")
12 if not os.path.isfile(keyfile):
13     raise FileNotFoundError(f"Clave no encontrada: {keyfile}")
14
15 # Configuración del servidor
16 server_address = ('localhost', 4443)
17 httpd = HTTPServer(server_address, SimpleHTTPRequestHandler)
18
19 # Crear el contexto SSL
20 context = ssl.SSLContext(ssl.PROTOCOL_TLS_SERVER)
21 context.load_cert_chain(certfile=certfile, keyfile=keyfile)
22
23 # Envolver el socket del servidor en el contexto SSL
24 httpd.socket = context.wrap_socket(httpd.socket, server_side=True)
25
26 print(f"Servidor HTTPS corriendo en https://localhost:4443")
27 httpd.serve_forever()
```

- 7) Una vez hecho el script procedemos a realizar la ejecución del script y como vemos no nos marcó error al ejecutar el script en python y nos muestra un mensaje que es la siguiente *Servidor HTTPS corriendo en <https://localhost:4443>* y le damos clic.



```
PS C:\Users\52953\Documents> python Seguridad.py
52953\AppData\Local\Programs\Python\Python312\python.exe c:/Users/52953/Documents/Ejercicio4/Seguridad.py
Servidor HTTPS corriendo en https://localhost:4443
```

- 8) Al darle clic al enlace que nos dio el script nos muestra la siguiente pantalla.



```

C:\Users\S29593\Documents\Ejercicio04\openssl s_client -connect localhost:4443 -showcerts
Connecting to 127.0.0.1:
CONNECTED(000001c4)
Can't use SSL get_servername
depth=0 c=MX, ST=Oaxaca, L=TLaxiaco, O=Computer System, OU=System Integer , CN=Edwin, emailAddress=santiagoedwin051@gmail.com
verify error:num=1:self-signed certificate
verify return:1
depth=0 c=MX, ST=Oaxaca, L=TLaxiaco, O=Computer System, OU=System Integer , CN=Edwin, emailAddress=santiagoedwin051@gmail.com
verify return:1

Certificate chain
 0:c=MX, ST=Oaxaca, L=TLaxiaco, O=Computer System, OU=System Integer , CN=Edwin, emailAddress=santiagoedwin051@gmail.com
 1:c=MX, ST=Oaxaca, L=TLaxiaco, O=Computer System, OU=System Integer , CN=Edwin, emailAddress=santiagoedwin051@gmail.com
a:PKCS: rsaEncryption, 2048 (bit); sigalg: RSA-SHA256
NoNotBefore: Sep 1 03:59:19 2024 GMT; NotAfter: Sep 1 03:59:19 2025 GMT
-----BEGIN CERTIFICATE-----
MIIEIzCCAsuagAwIBAgIUERc2EakaX18wRwRGPZg7DVdVlJFhMoQY7KcZlHvcNAQEL
BQAwAgAAcAAcA3BglbYk9AYTA1MQ8wOQYDVQIDAZPXXhyZXZlZXREAPBhNBAACFRS
YXhpYmVhcnRmRgVGVVQVQDAG9Bb21wXRI1BTEuXNOzWm8wQDABGVBASMDInSc3R1
BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1
Ym9uZW93aG4uMTF4Z211bnRwY29hbm9kY29hbm9kY29hbm9kY29hbm9kY29hbm9k
Y29hbm9kY29hbm9kY29hbm9kY29hbm9kY29hbm9kY29hbm9kY29hbm9kY29hbm9k
NTk0OV8wZG93aG4uMTF4Z211bnRwYk9AYTA1MQ8wOQYDVQIDAZPXXhyZXZlZXREAPBh
NBAACFRSAYXhpYmVhcnRmRgVGVVQVQDAG9Bb21wXRI1BTEuXNOzWm8wQDABGVBASMDIn
Sc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1
BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEu
XNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOz
Wm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8w
QDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDAB
GVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBAS
MDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDIn
Sc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1
BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEu
XNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOz
Wm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8w
QDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDAB
GVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBAS
MDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDIn
Sc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1
BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEu
XNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOz
Wm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8w
QDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDAB
GVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBAS
MDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDIn
Sc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1
BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEu
XNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOz
Wm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8w
QDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDAB
GVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBAS
MDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDIn
Sc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1
BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEu
XNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOz
Wm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8w
QDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDAB
GVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBAS
MDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDIn
Sc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1
BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEu
XNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOz
Wm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8w
QDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDAB
GVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBAS
MDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDIn
Sc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1
BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEu
XNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOz
Wm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8w
QDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDAB
GVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBAS
MDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDIn
Sc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1
BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEu
XNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOz
Wm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8w
QDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDAB
GVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBAS
MDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDIn
Sc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1
BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEu
XNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOz
Wm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8w
QDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDAB
GVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBAS
MDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDIn
Sc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1
BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEu
XNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOz
Wm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8w
QDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDAB
GVBASMDInSc3R1BTEuXNOzWm8wQDABGVBASMDInSc3R1BTEuXNOzWm8wQDABGVBAS
MDInSc3R1
```



Investiga y describe los siguientes conceptos:

- Contraseña
- Certificado digital
- Firma digital
- Cifrado simétrico
- Cifrado asimétrico
- Hash
- Encriptación

Contraseña: Es una secuencia secreta de caracteres (letras, números y símbolos) utilizada para proteger el acceso a un sistema, aplicación o recurso en línea. Su propósito es garantizar que solo las personas autorizadas puedan ingresar a un área restringida. Cuando un usuario intenta acceder a un sistema, introduce su contraseña, la cual se compara con la que está almacenada en el sistema. Si coinciden, se concede el acceso. Las contraseñas deben ser suficientemente complejas para resistir intentos de adivinanza o ataques automatizados, como el uso de programas que prueban combinaciones posibles. Las mejores prácticas incluyen el uso de contraseñas largas y variadas, así como el cambio regular de las mismas. Las contraseñas son esenciales para la seguridad digital, ya que ayudan a prevenir accesos no autorizados. Sin embargo, su efectividad depende de su complejidad; una contraseña débil puede ser fácilmente vulnerable a ataques, mientras que una fuerte, que combina letras, números y símbolos, ofrece mayor protección.

Certificado digital: Es un archivo electrónico que actúa como una credencial en el mundo digital. Contiene información sobre la identidad del titular (como su nombre y dirección de correo electrónico) y una clave pública utilizada para la encriptación. Además, está firmado digitalmente por una entidad de certificación (CA), que actúa como una autoridad de confianza que verifica la autenticidad del certificado. Cuando una persona recibe un certificado digital, puede usarlo para asegurar sus comunicaciones y transacciones en línea. El certificado garantiza que las comunicaciones están protegidas y que las partes involucradas son quienes dicen ser. Es una parte fundamental de la infraestructura de clave pública (PKI) utilizada para proteger datos y validar identidades en Internet. Este certificado asegura que la comunicación entre dos partes es segura y que las identidades involucradas son legítimas.

Firma digital: Asegura la autenticidad y la integridad de un mensaje o documento electrónico mediante el uso de criptografía de clave pública. El firmante utiliza su clave privada para crear una firma digital única para el documento. Esta firma se adjunta al documento y se puede verificar con la clave pública del firmante. Si el



documento es modificado después de la firma, la firma digital ya no será válida, lo que indica que el contenido ha sido alterado. La firma digital garantiza que el documento proviene de una fuente confiable y que no ha sido alterado durante la transmisión. Aplicar una firma digital proporciona una verificación clara de la identidad y el contenido, siendo crucial en transacciones electrónicas.

Cifrado simétrico: Es un método de encriptación que utiliza la misma clave para cifrar y descifrar datos. Es rápido y eficiente, adecuado para manejar grandes volúmenes de datos. La clave debe mantenerse en secreto y compartirse de manera segura entre las partes que necesitan acceder a los datos cifrados. Uno de los desafíos del cifrado simétrico es la distribución segura de la clave; si la clave es interceptada por una persona no autorizada, la seguridad del cifrado se ve comprometida. Ejemplos de algoritmos de cifrado simétrico incluyen AES (Estándar de Cifrado Avanzado) y DES (Estándar de Cifrado de Datos).

Cifrado asimétrico: Utiliza un par de claves relacionadas: una clave pública y una clave privada. La clave pública puede compartirse abiertamente, mientras que la clave privada se mantiene en secreto. Cuando alguien cifra un mensaje usando la clave pública, solo el titular de la clave privada correspondiente puede descifrarlo. Este método permite la encriptación de datos y la verificación de identidades sin necesidad de compartir una clave secreta. Además, el cifrado asimétrico es fundamental para la firma digital y el intercambio seguro de claves en cifrado simétrico. Algoritmos comunes de cifrado asimétrico incluyen RSA (Rivest-Shamir-Adleman) y ECC (Elliptic Curve Cryptography). Aunque es más lento que el cifrado simétrico, se utiliza ampliamente en la seguridad digital para asegurar la comunicación y permitir el intercambio seguro de información.

Hash: Es una función hash que toma una entrada (o mensaje) y produce una cadena de longitud fija, denominada valor hash o "digest", que representa los datos originales. Las funciones hash están diseñadas para ser rápidas de calcular, pero difíciles de revertir, lo que significa que es complicado reconstruir la entrada original a partir del valor hash. Además, debe ser improbable que dos entradas diferentes produzcan el mismo valor hash (colisiones). Los valores hash se utilizan en diversas aplicaciones, como verificar la integridad de los archivos y almacenar contraseñas de forma segura en las bases de datos. Aunque es muy difícil revertir un hash, los algoritmos débiles pueden ser vulnerables a colisiones, donde dos entradas diferentes producen el mismo hash. Ejemplos de algoritmos hash son SHA-256 (Secure Hash Algorithm) y MD5 (Message Digest Algorithm).

Encriptación: Es el proceso de transformar datos legibles en un formato codificado para proteger la información de accesos no autorizados. Existen dos tipos principales de encriptación: simétrica y asimétrica. En el cifrado simétrico, se utiliza la misma clave para cifrar y descifrar los datos, mientras que en el cifrado asimétrico se usan dos claves diferentes (una pública y una privada). La encriptación asegura que los datos sean ilegibles para cualquier persona que no tenga la clave adecuada para descifrarlos. Se emplea en diversas aplicaciones, desde la transmisión segura de datos en Internet hasta el almacenamiento de información confidencial. Es una herramienta crucial para mantener la privacidad y la seguridad en la comunicación digital.

Investiga y describe los siguientes algoritmos de cifrado:

- AES
- RSA
- SHA-256

AES (Advanced Encryption Standard)

Descripción: AES es un algoritmo de cifrado simétrico adoptado en 2001 por el Instituto Nacional de Estándares y Tecnología (NIST). Se utiliza ampliamente para proteger datos y reemplazó al DES (Data Encryption Standard).

Características:

- **Tipo:** Cifrado simétrico.
- **Tamaño de clave:** 128, 192 o 256 bits.
- **Bloques:** Opera sobre bloques de 128 bits de datos.
- **Rondas:** 10 rondas para claves de 128 bits, 12 para 192 bits, y 14 para 256 bits.

Ventajas:

- **Seguridad:** Muy seguro y eficiente.
- **Velocidad:** Rápido en software y hardware.
- **Estándar:** Ampliamente aceptado.

Desventajas:

- **Sin clave secreta:** La misma clave se usa para cifrar y descifrar.

RSA (Rivest-Shamir-Adleman)

Descripción: RSA es un algoritmo de cifrado asimétrico desarrollado en 1977. Se utiliza para la transmisión segura de datos y para firmas digitales, utilizando un par de claves: una pública y una privada.

Características:

- **Tipo:** Cifrado asimétrico.



- **Tamaño de clave:** 1024, 2048 o 4096 bits.
- **Proceso:** Usa una clave pública para cifrar y una privada para descifrar.
- **Seguridad:** Basado en la dificultad de factorizar grandes números primos.

Ventajas:

- **Seguridad:** Muy seguro con claves adecuadas.
- **Autenticación:** Permite firmas digitales y verificación de identidad.
- **Interoperabilidad:** Soporte en protocolos como SSL/TLS.

Desventajas:

- **Velocidad:** Más lento que el cifrado simétrico.
- **Tamaño:** Operaciones más costosas en términos de computación.

SHA-256 (Secure Hash Algorithm 256-bit)

Descripción: SHA-256 es una función de hash criptográfico de la familia SHA-2, que genera un valor hash de 256 bits a partir de una entrada de longitud variable. Se usa para garantizar la integridad y autenticidad de los datos.

Características:

- **Tipo:** Función de hash criptográfico.
- **Salida:** Hash de 256 bits (32 bytes).
- **Proceso:** Convierte una entrada de longitud variable en una cadena fija.
- **Seguridad:** Resistente a colisiones y ataques de preimagen.

Ventajas:

- **Seguridad:** Alta seguridad para la mayoría de las aplicaciones.
- **Integridad:** Verifica la integridad de los datos.
- **Popularidad:** Ampliamente utilizado en seguridad y criptografía.

Desventajas:

- **Rendimiento:** Menos eficiente en velocidad comparado con otras funciones de hash.
- **Tamaño:** Hash de 256 bits, que puede ser mayor que los generados por algoritmos más antiguos como MD5.

Investiga y describe los siguientes estándares de cifrado:

- SSL
- TLS

SSL (Secure Sockets Layer): Traducido es Capa de sockets seguros, es un protocolo de seguridad que protege la comunicación entre tu navegador y un sitio web. Desarrollado por Netscape en 1994, SSL cifra los datos enviados y recibidos para evitar que sean interceptados o alterados por atacantes. También autentica el servidor (y a veces al cliente) para asegurar que ambos están comunicándose con las partes correctas. Sin embargo, SSL ha sido reemplazado por un protocolo más seguro debido a sus vulnerabilidades.

TLS (Transport Layer Security): Traducido es Seguridad de la capa de transporte, es el sucesor de SSL, introducido en 1999 para abordar las debilidades de seguridad de SSL. TLS cifra la información de manera más avanzada y segura, y mejora la autenticación y la verificación de datos. Es el estándar actual para asegurar las comunicaciones en la web, utilizado por la mayoría de los navegadores y servidores modernos. Las versiones más recientes, como TLS 1.2 y TLS 1.3, ofrecen una mayor seguridad y rendimiento en comparación con las versiones anteriores de SSL.

SSL y TLS utilizan varios estándares de cifrado para proteger las comunicaciones en la web. Estos estándares se han desarrollado para asegurar que la información transmitida sea confidencial y esté protegida contra interceptaciones y alteraciones.

- **Cifrado Simétrico**
 - ❖ AES: Seguro y eficiente, usado en TLS 1.2 y 1.3.
 - ❖ 3DES y RC4: Menos seguros; RC4 está obsoleto.
- **Cifrado Asimétrico:**
 - ❖ RSA: Para intercambio de claves y autenticación.
 - ❖ ECC: Ofrece seguridad alta con claves más cortas, usado en versiones recientes.
- **Funciones de Hashing:**
 - ❖ SHA-256, SHA-384, SHA-512: Usados para verificar la integridad de los datos.
- **Intercambio de Claves:**
 - ❖ DH y ECDH: Para intercambio seguro de claves, ECDH es más eficiente.



Investiga y describe los siguientes protocolos de seguridad:

- HTTPS
- SFTP
- SSH

HTTPS (HyperText Transfer Protocol Secure): Por sus siglas significa Protocolo de transferencia de hipertexto seguro, HTTPS es una extensión segura del protocolo HTTP utilizado para la transferencia segura de datos a través de internet. Utiliza el protocolo SSL/TLS para cifrar la comunicación entre el navegador web del usuario y el servidor web, lo que garantiza que los datos transmitidos no puedan ser interceptados o modificados por terceros. Al implementar HTTPS, se utiliza un certificado digital que autentica la identidad del sitio web y establece una conexión segura, indicada por el ícono del candado en la barra de direcciones del navegador.

SFTP (Secure File Transfer Protocol): Por sus siglas significa Protocolo de transferencia segura de archivos, SFTP es un protocolo seguro utilizado para la transferencia segura de archivos a través de una red. A diferencia de FTP estándar, que transmite datos de manera no cifrada, SFTP utiliza SSH (Secure Shell) para cifrar tanto los comandos de control como los datos transmitidos. Esto proporciona autenticación segura y confidencialidad de la información durante la transferencia de archivos entre un cliente y un servidor remoto. SFTP es ampliamente utilizado en entornos donde se requiere una alta seguridad para la transferencia de datos sensibles. Es ampliamente utilizado en entornos donde la seguridad es una prioridad, garantizando que solo las personas autorizadas puedan acceder a los archivos.

SSH (Secure Shell): Por sus siglas significa Shell seguro, SSH es un protocolo de red que permite a los usuarios acceder de manera segura a sistemas remotos y ejecutar comandos de forma segura sobre una red no segura. Utiliza técnicas criptográficas para asegurar todas las comunicaciones entre el cliente SSH y el servidor SSH, incluyendo la autenticación del usuario, la integridad de los datos transmitidos y la confidencialidad. SSH se utiliza comúnmente para administrar servidores de manera remota a través de una interfaz de línea de comandos segura, y también soporta la transferencia segura de archivos (SFTP) y el reenvío seguro de puertos. Es una herramienta esencial para la administración remota de servidores y para garantizar la seguridad en redes inseguras como Internet.



Conclusión

Realizar esta práctica y la investigación asociada es esencial para comprender la importancia de proteger nuestra información en un entorno digital cada vez más vulnerable. Al crear programas que validan contraseñas y generan certificados, ponemos en práctica herramientas concretas para asegurar datos sensibles. Además, al investigar sobre cifrado y protocolos de seguridad, adquirimos un entendimiento más profundo de cómo estas tecnologías nos ayudan a mantener la confidencialidad e integridad de la información. Investigar conceptos como el cifrado y los protocolos de seguridad nos ha dado una base sólida para entender cómo funciona la seguridad en el mundo digital. Estos conocimientos son fundamentales para garantizar que la información permanezca privada y segura.

Esta práctica refuerza la idea de que la seguridad es un componente crucial en cualquier sistema y que, como futuros profesionales, debemos estar preparados para implementar y gestionar medidas de protección eficaces.



Bibliografía

<https://www.uach.cl/direccion-de-tecnologias-de-informacion/seguridad/que-es-una-contrasena-segura>
<https://www.xataka.com/basics/certificado-digital-que-que-tipos-hay-como-solicitarlo-activarlo>
<https://www.accessq.com.mx/diferencia-encryptacion-simetrica-asimetrica/#:~:text=%C2%BFQu%C3%A9%20Diferencia%20hay%20entre%20los,los%20mensajes%20cuando%20se%20comunican.>
<https://www.docusign.com/es-mx/blog/desarrolladores/hash>
https://www.sede.fnmt.gob.es/preguntas-frecuentes/otras-preguntas/-/asset_publisher/1RphW9leUoAH/content/1024-que-es-la-encryptacion-o-cifrado-#:~:text=La%20encryptaci%C3%B3n%20o%20cifrado%20es,un%20mensaje%20que%20estaba%20cifrado.
http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1815-59282013000300005
<https://www.redeszone.net/tutoriales/servidores/ftps-ftpes-sftp-caracteristicas-diferencias/>
<https://www.ssldragon.com/es/blog/rsa-aes-cifrado/>
<https://www.redeszone.net/tutoriales/seguridad/criptografia-algoritmos-clave-simetrica-asimetrica/>
<https://www.digicert.com/es/what-is-ssl-tls-and-https>
<https://aws.amazon.com/es/what-is/ssl-certificate/>
<https://www.ssl.com/es/art%C3%ADculo/que-es-ssl-tls-an-in-depth-guide/>
<https://es.hostzealot.com/blog/about-web-hosting/que-es-el-protocolo-de-transferencia-de-archivos-ssh-sftp>
<https://www.sybcodex.com/2021/04/que-son-los-protocolos-de-seguridad-https-sftp-ssh-smtps.html>
<https://www.cdmon.com/es/blog/protocolos-de-transferencia-de-archivos-ftp-sftp-y-ssh>