PROCESOS DE SOFTWARE SEGURO

DESARROLLO DE SOFTWARE SEGURO

CONTRASEÑAS SEGURAS

CIFRADO DE CONTRASEÑA

- Jorge Ibarra Peña
- 20310025
- 7°P
- Desarrollo de Software
- 02/11/2023
- CETI COLOMOS

CONTRASEÑAS SEGURAS

Estándar:

ISO 27002 control 9.4.3 Gestión de contraseñas de usuario

Descripción:

Las contraseñas seguras son esenciales para proteger la seguridad en línea y la privacidad de las cuentas y la información personal. Ayudan a prevenir el acceso no autorizado, fraudes, robos de información y otros delitos en línea.

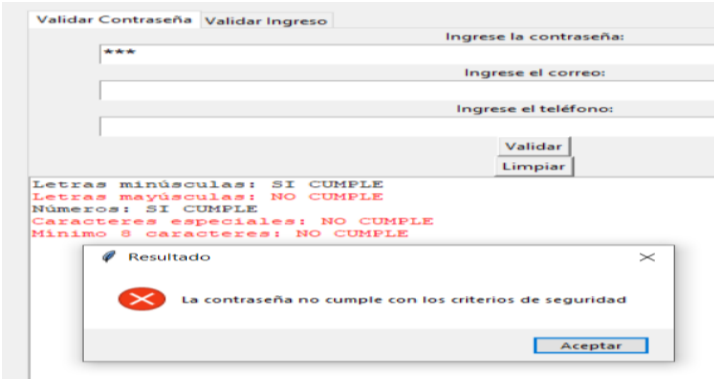
Las contraseñas sólidas son una primera línea de defensa contra amenazas cibernéticas y garantizan que tus cuentas y datos estén protegidos.

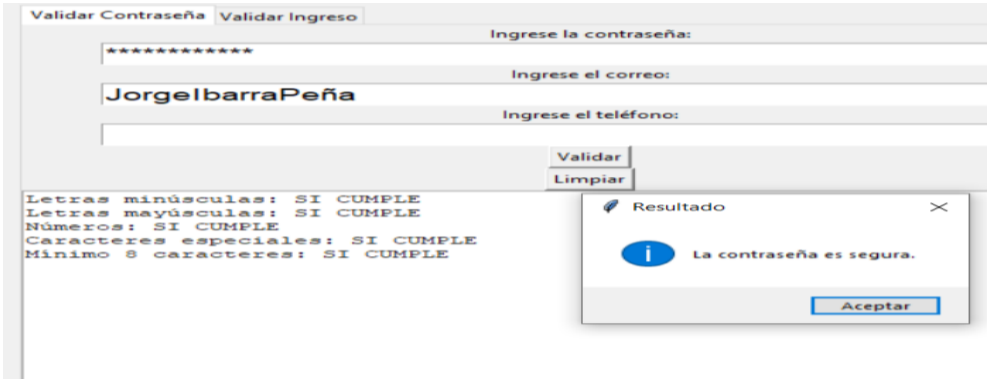
Para eso creamos un programa en Python el cual nos ayuda a hacer una contraseña segura donde nos piden que agreguemos letras minúsculas, mayúsculas, números y caracteres especiales. A continuación, la evidencia de lo realizado:

CODIGO:

```python
import re
import tkinter as tk
import sqlite3
from tkinter import messagebox
import hashlib
from tkinter import ttk

def validar_contrasena(contrasena):
    validaciones = {
        "Letras minúsculas": re.search("[a-z]", contrasena),
        "Letras mayúsculas": re.search("[A-Z]", contrasena),
        "Números": re.search("[0-9]", contrasena),
        "Caracteres especiales": re.search("[!@#$%&/()=;:_ñ<>¿'|°^.,]", contrasena),
        "Mínimo 8 caracteres": len(contrasena) >= 8
    }

    resultado_validacion = "\n".join([f"{criterio}: {'Cumple' if cumple else 'No cumple'}" for criterio, cumple in validaciones.items(

    return all(validaciones.values()), resultado_validacion

def guardar_datos(contrasena, correo, telefono):
    conn = sqlite3.connect("datos_usuarios.db")
    cursor = conn.cursor()
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS usuarios (
            id INTEGER PRIMARY KEY,
            contrasena TEXT,
            correo TEXT,
            telefono TEXT
        )
    ''')

    #CIFRADO DE CONTRASEÑA
    hash_contraseña = hashlib.sha256(contrasena.encode()).hexdigest()
    cursor.execute("INSERT INTO usuarios (contrasena, correo, telefono) VALUES (?, ?, ?)", (hash_contraseña, correo, telefono))
    conn.commit()
    conn.close()

def verificar_contrasena():
    contrasena = entrada_contraseña.get()
```

RESULTADO:





Cifrado de contraseñas

Estándar:

ISO 27002 Control 9.4.3 Gestión de contraseñas de usuario y 10 Cifrado

Descripción:

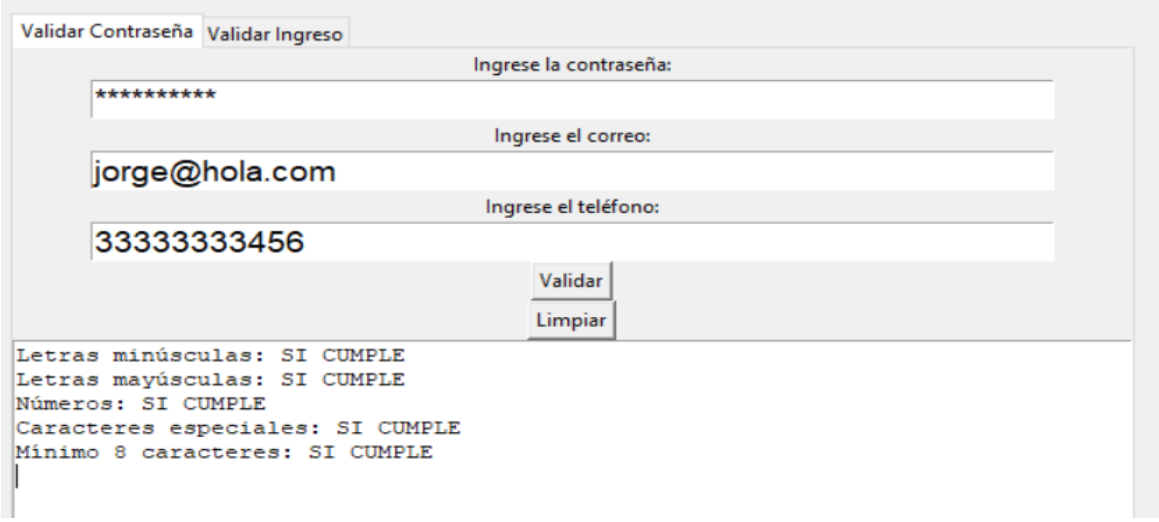
Bcrypt es una función de hash diseñada específicamente para el almacenamiento seguro de contraseñas. Cuando un usuario crea una cuenta o cambia su contraseña en una aplicación Laravel, la contraseña se pasa a través del algoritmo Bcrypt antes de almacenarla en la base de datos.

Para esto lo que haremos es una base de datos dentro del programa el cual cabe resaltar que fue desarrollado en Python incluyendo el sql y una vez creada lo que haremos es utilizar BD Browser for SQLite para poder administrar y comprobar todos los datos guardados en la base de datos. A continuación, el ejemplo:

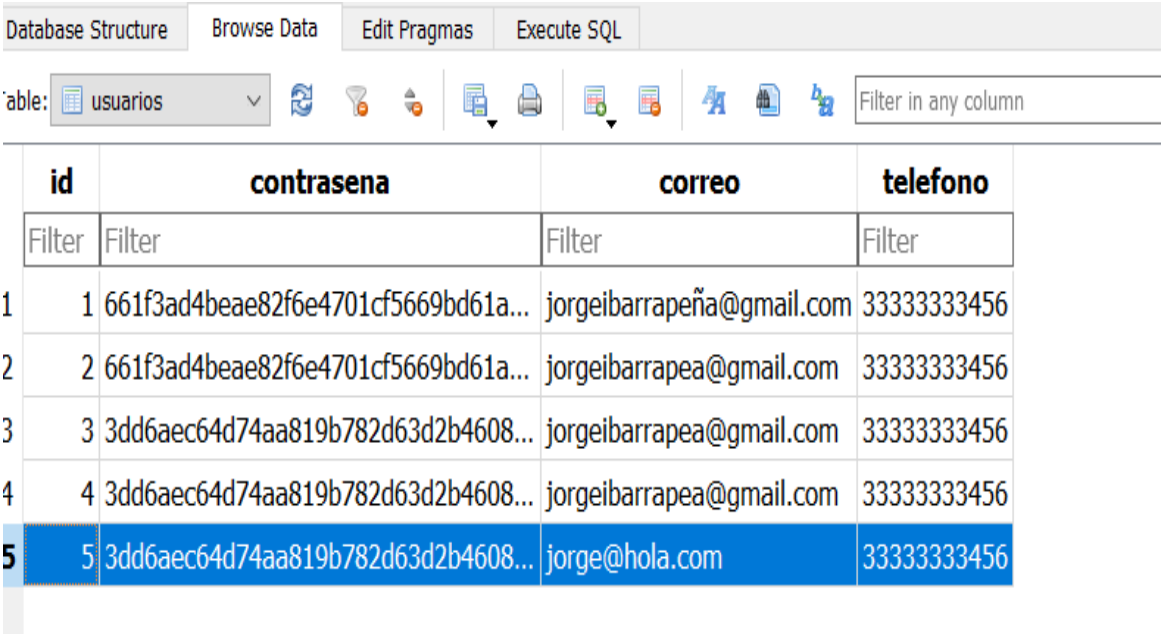
CODIGO:

```
    #CIFRADO DE CONTRASEÑA
    hash_contraseña = hashlib.sha256(contrasena.encode()).hexdigest()
    cursor.execute("INSERT INTO usuarios (contrasena, correo, telefono) VALUES (?, ?, ?)", (hash_contraseña, correo, telefono))
    conn.commit()
    conn.close()

def verificar_contrasena():
    contrasena = entrada_contraseña.get()
    correo = entrada_correo.get()
    telefono = entrada_telefono.get()  # Agrega el campo para el teléfono
    es_segura, resultado_validacion = validar_contrasena(contrasena)
    etiqueta_progreso.delete(1.0, tk.END)

    for linea in resultado_validacion.split("\n"):
        titulo, resultado = linea.split(": ")
        if resultado == "Cumple":
            etiqueta_progreso.insert(tk.END, f"{titulo}: SI CUMPLE\n", titulo)
        else:
            etiqueta_progreso.insert(tk.END, f"{titulo}: NO CUMPLE\n", "no_cumple")

    if es_segura:
        messagebox.showinfo("Resultado", "La contraseña es segura.")
    else:
        messagebox.showerror("Resultado", "La contraseña no cumple con los criterios de seguridad")

    guardar_datos(contrasena, correo, telefono)

def limpiar_campos():
    entrada_contraseña.delete(0, tk.END)
    entrada_correo.delete(0, tk.END)
    entrada_telefono.delete(0, tk.END)

def validar_ingreso():
    contrasena_ingresada = entrada_contraseña_ingresada.get()
    correo_ingresado = entrada_correo_ingresado.get()

    conn = sqlite3.connect("datos_usuarios.db")
    cursor = conn.cursor()
    cursor.execute("SELECT contrasena FROM usuarios WHERE correo = ?", (correo_ingresado,))
```

RESULTADOS:



Validamos que los datos se hayan guardado en la base de datos:



	id	contrasena	correo	telefono
	Filter	Filter	Filter	Filter
1	1	661f3ad4beae82f6e4701cf5669bd61a...	jorgeibarrapeña@gmail.com	33333333456
2	2	661f3ad4beae82f6e4701cf5669bd61a...	jorgeibarrapea@gmail.com	33333333456
3	3	3dd6aec64d74aa819b782d63d2b4608...	jorgeibarrapea@gmail.com	33333333456
4	4	3dd6aec64d74aa819b782d63d2b4608...	jorgeibarrapea@gmail.com	33333333456
5	5	3dd6aec64d74aa819b782d63d2b4608...	jorge@hola.com	33333333456

Y por último validamos si la contraseña y el correo son válidos: