23-2-2024

CGI BBDD Python

Implantación de Aplicaciones Web 2º ASIR

**Andres Catalina Blázquez**

Contenido De La Memoria

[**1.** **Creación de BBDD** 2](#_Toc159582714)

[**2.** **Creación de formularios** 2](#_Toc159582715)

[**a)** **Registrarse** 2](#_Toc159582716)

[**b)** **Login** 3](#_Toc159582717)

[**3.** **Python** 4](#_Toc159582718)

[**a)** **Registrarse** 4](#_Toc159582719)

[**I.** **Iniciar página y recoger datos** 4](#_Toc159582720)

[**II.** **Conexión a la BBDD y encriptar contraseña** 5](#_Toc159582721)

[**III.** **Insertar datos a la BBDD** 5](#_Toc159582722)

[**IV.** **Excepciones** 6](#_Toc159582723)

[**b)** **Login** 7](#_Toc159582724)

[**4.** **Conclusión y explicaciones** 9](#_Toc159582725)

[**a)** **Modificaciones en httpd.conf de Apache** 9](#_Toc159582726)

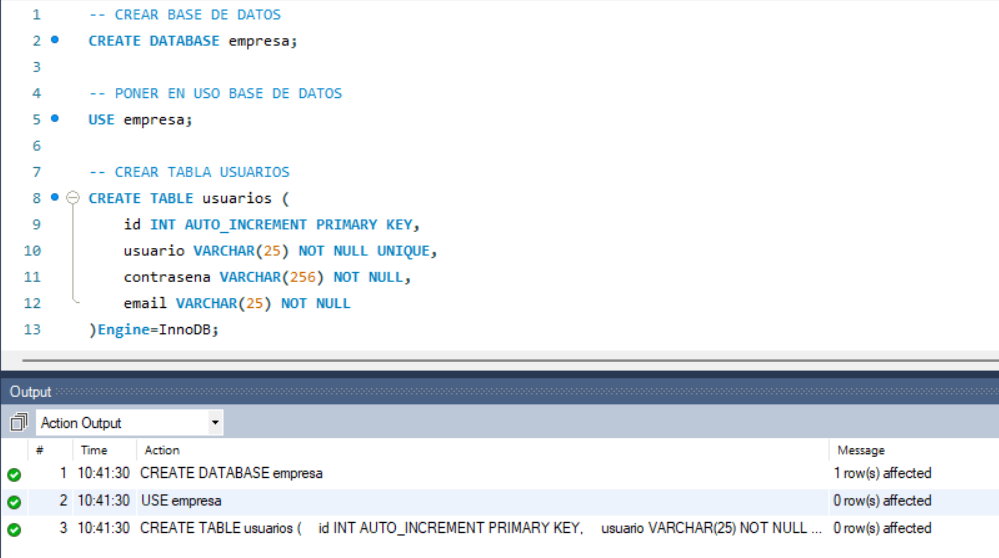
[**I.** **Módulo de CGI** 9](#_Toc159582727)

[**II.** **Crear Directorio** 10](#_Toc159582728)

[**I.** **Crear Directorio** 10](#_Toc159582729)

[**5.** **Pruebas de que todo funciona correctamente y errores** 11](#_Toc159582730)

# **Creación de BBDD**

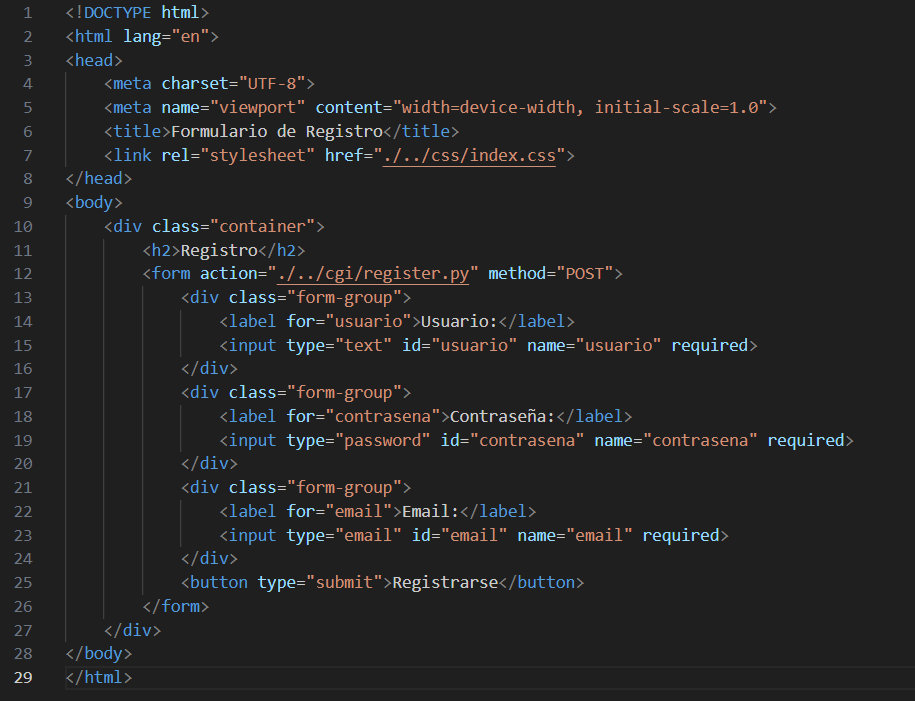


# **Creación de formularios**

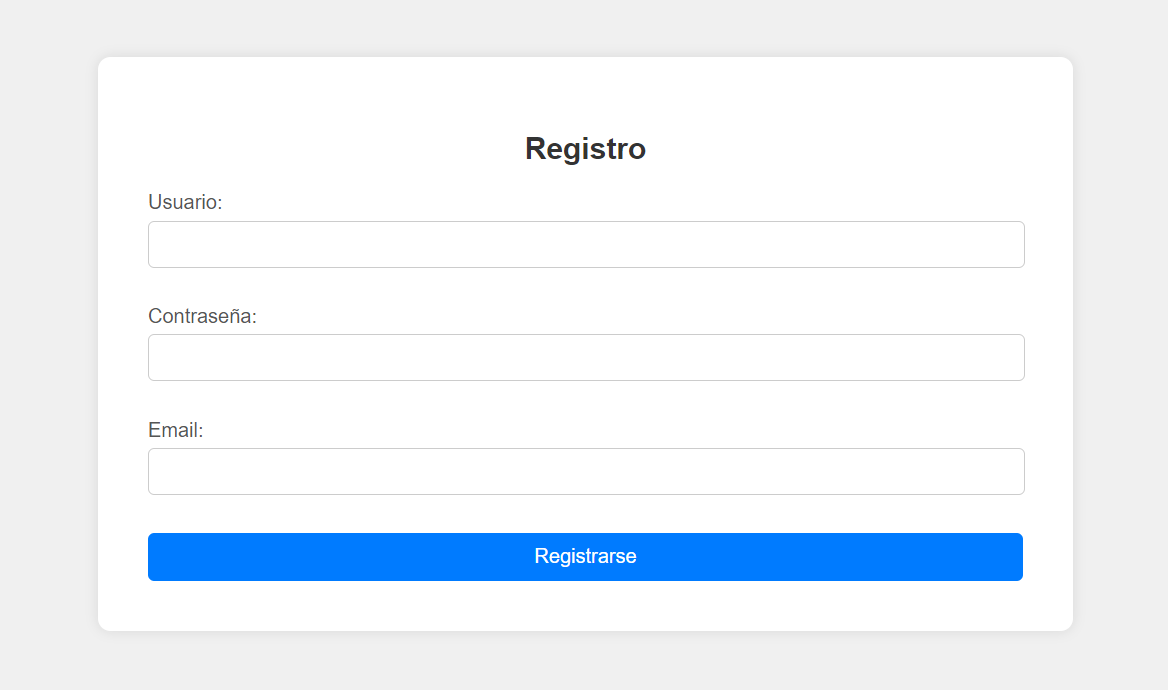
En mi caso he optado por 2 formularios (Registrarse y Login) para poder registrar usuarios en la BBDD y además luego poder comprobar que los usuarios pueden acceder a una pagina que se ha creado previamente. (La pagina esta desarrollada de otros años anteriores y mejorada con ayuda de inteligencia artificial)

## **Registrarse**

Código de la pagina

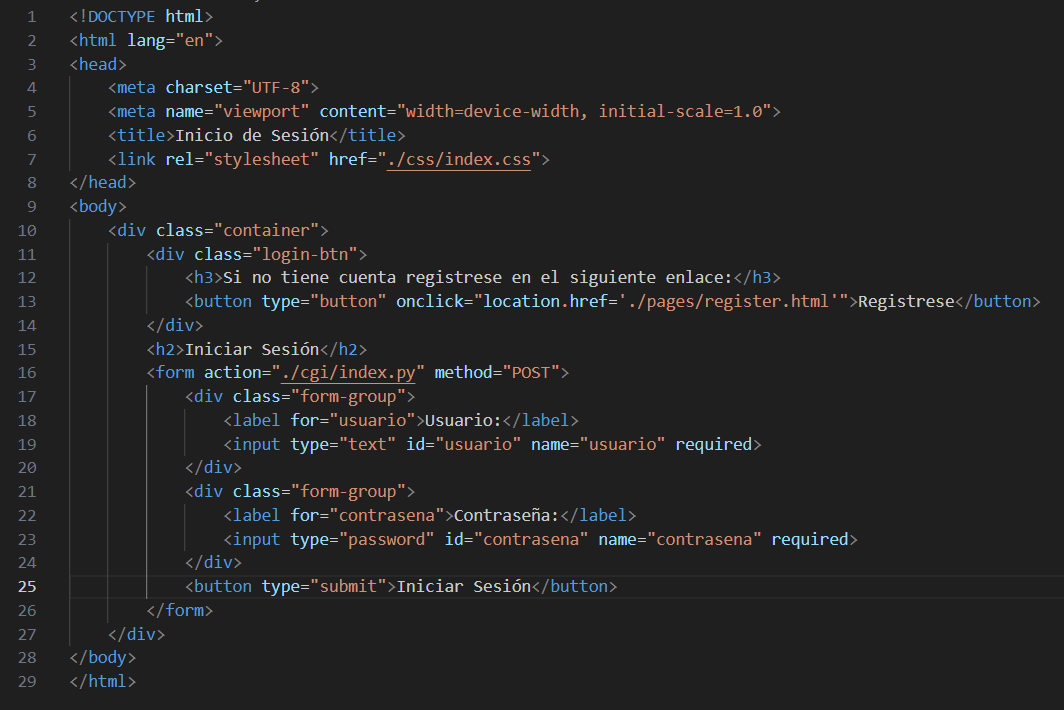


Vista desde forma visual para el cliente



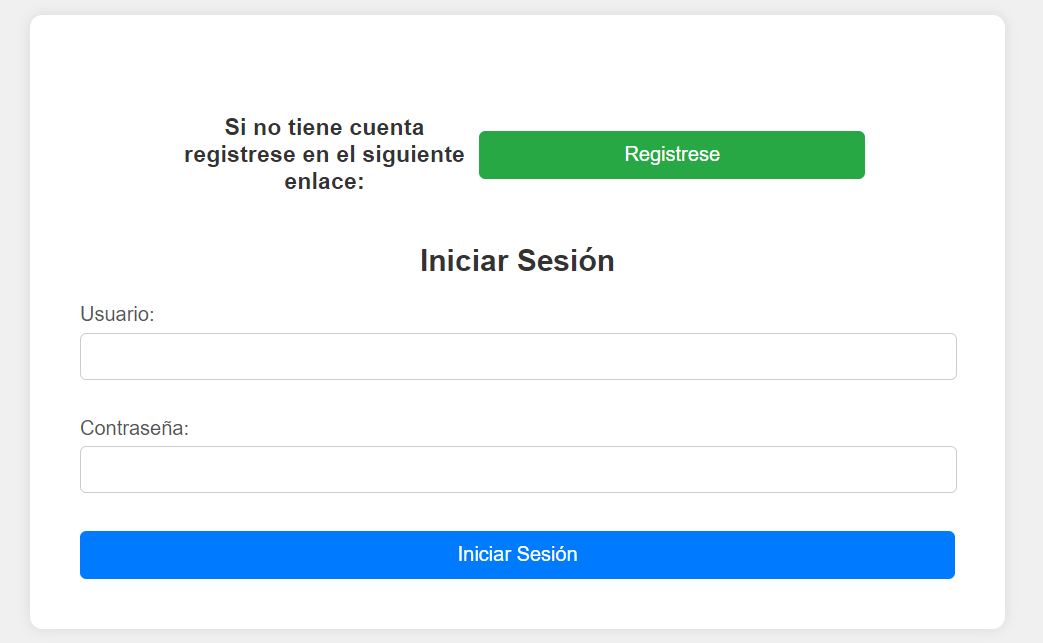
## **Login**

Código de la pagina



Vista desde forma visual para el cliente.

(Hemos añadido un botón que redirige a la página de Registrarse por si no tienes cuenta ya que la página de inicio de sesión será la página principal index.html)

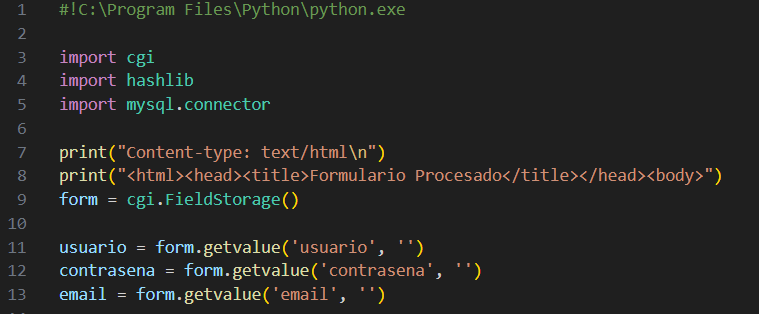


# **Python**

## **Registrarse**

### **Iniciar página y recoger datos**

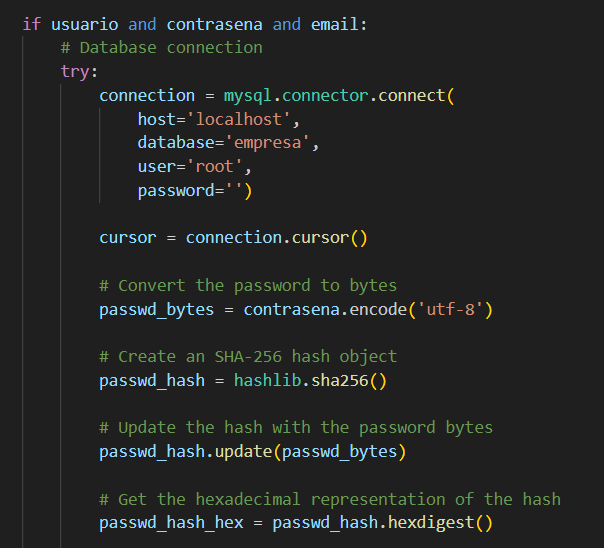
En el archivo de Python importamos las librerías (import cgi/ import hashlib/ import mysql.connector), iniciamos el HTML con un print (print("Content-type: text/html\n")). Nos traemos los datos del HTML con getvalue(), habiendo cogido las librerías cgi.



### **Conexión a la BBDD y encriptar contraseña**

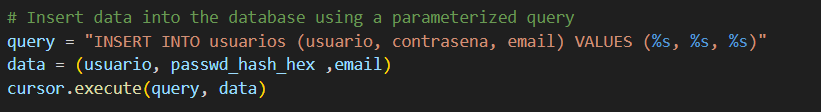
En este paso indicamos que, si existe “usuario”, “contraseña” y “email” abra un cursor que realice la conexiona la BBDD y poder comenzar a realizar operaciones en la misma.

Además, como opción de seguridad he puesto que las contraseñas se encripten en formato hexadecimal antes de guardarlas en la BBDD, así no se guardan en texto plano y en caso de que alguien acceda a la BBDD no pueda visualizar las contraseñas de forma clara. Para realizar ese paso tenemos que importar previamente la librería **hashlib.**

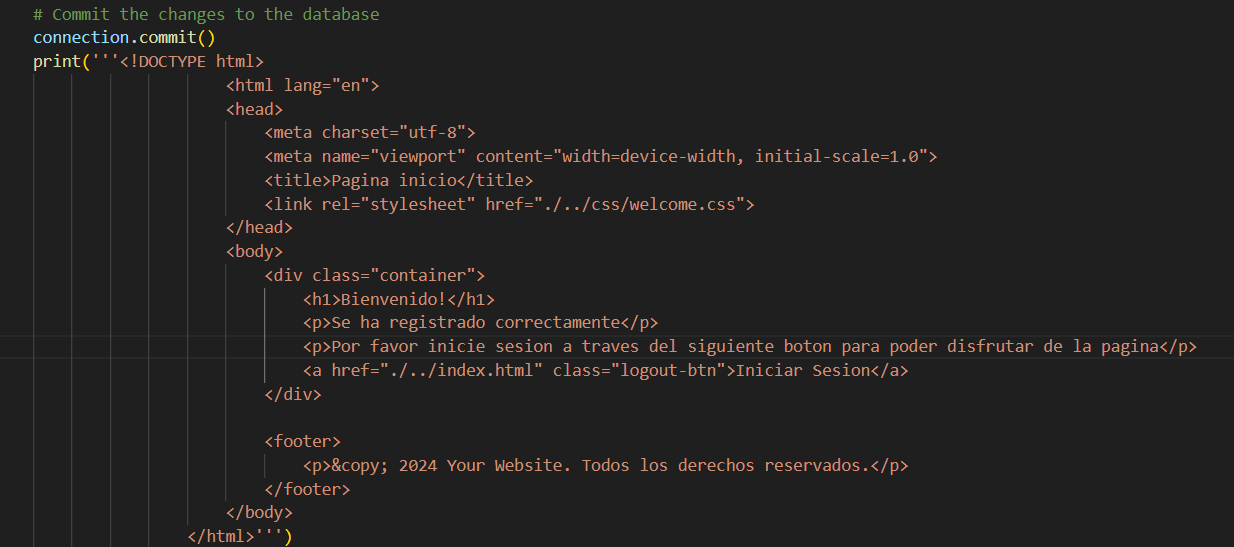


### **Insertar datos a la BBDD**

Una vez hemos recogido todos los parámetros, generamos una variable llamada query y metemos la inserción, además, en otra variable llamada data introducimos los datos que hemos recogido previamente y lanzamos el cursor para que ejecute la inserción de los datos facilitados por el cliente



En caso de que sea correcta la inserción hemos puesto que se muestre una interfaz que le indique al cliente que se ha registrado correctamente.

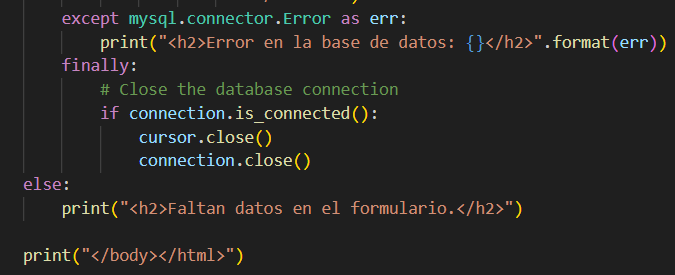


### **Excepciones**

Capturamos excepciones con except.

Nos ha interesado capturar los errores de MySQL y mostrarlos por pantalla

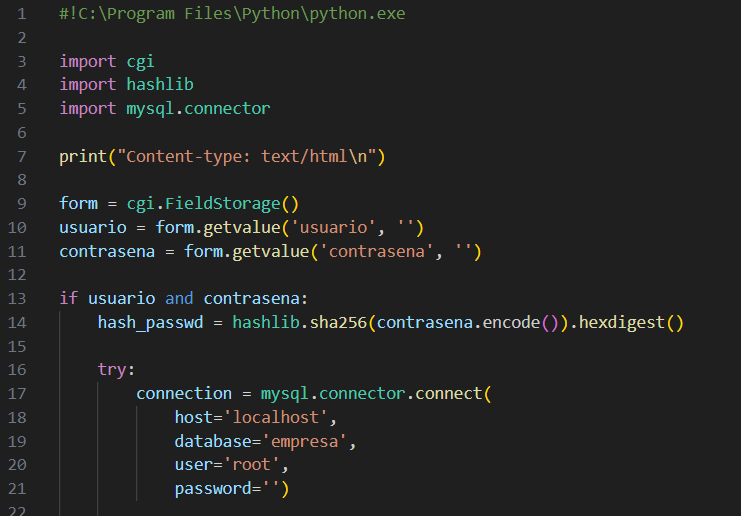
Con el finally, se haya ejecutado el try o el except, cerramos el cursor y la conexión a MySQL.

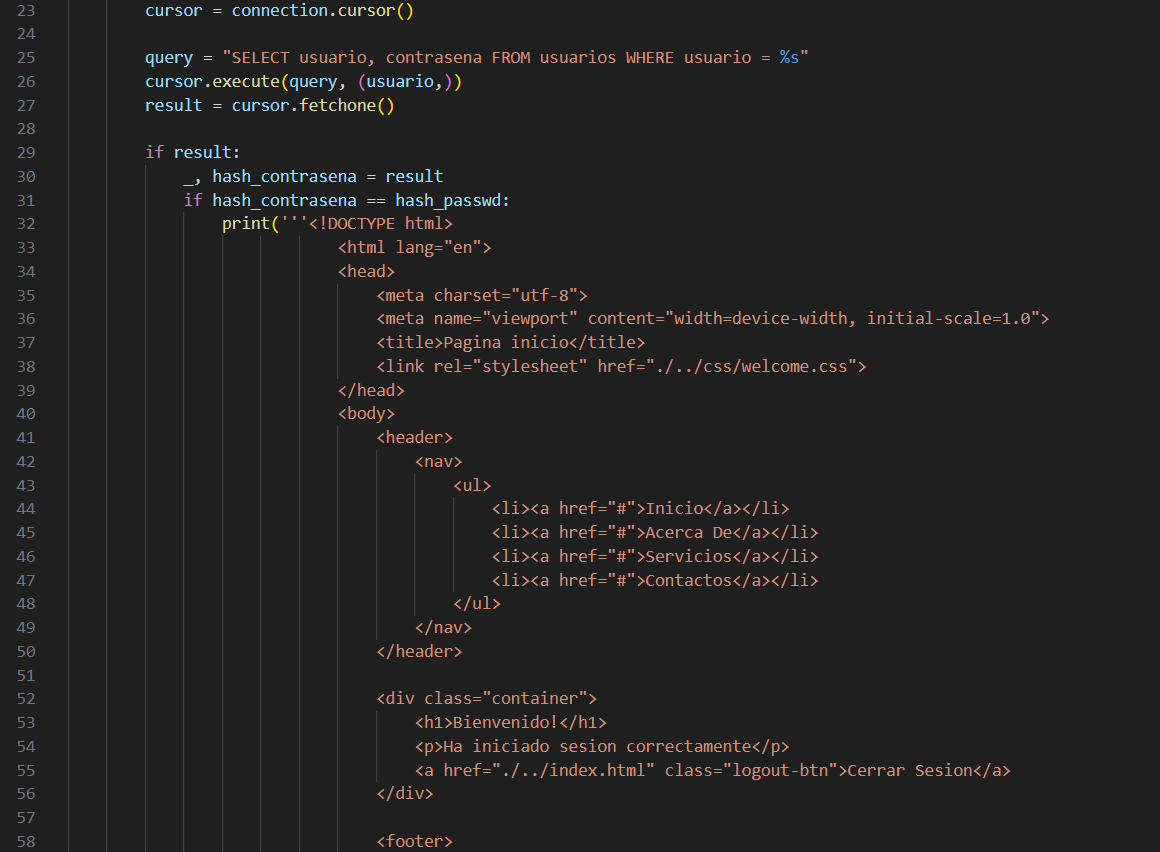


## **Login**

Al igual que en anterior este archivo de Python lleva una estructura similar.

Primero recogemos los datos, si existen esos datos abre un cursor, realiza la conexión a la BBDD, realiza la SELECT de los usuarios con el nombre que se ingresa en el formulario y se guarda en una variable. Si esa variable tiene algún valor significa que ese usuario existe entonces pasamos a comprobar la contraseña.





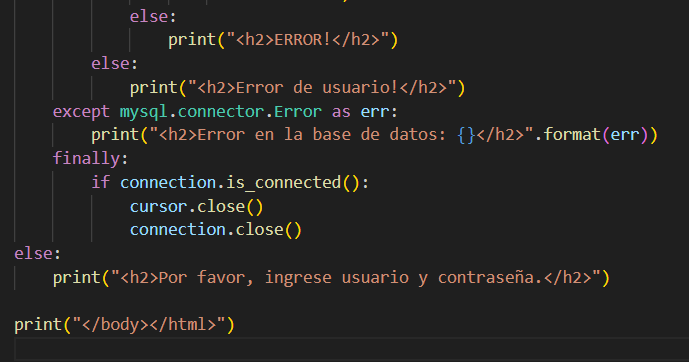
**\* ( \_, hash\_contrasena = result) 🡪 Con esto lo que realizamos es la desestructuración de tuplas, es decir, result es una tupla que contiene dos elementos: el primero corresponde al nombre de usuario que se asigna a \_ y se descarta y el segundo corresponde a la contraseña que se asigna a hash\_contrasena(En Python La barra baja ‘\_’ se utiliza para indicar que el valor no se utilizará) \***



Posteriormente para comprobar la contraseña hemos comprobado que la contraseña que esta almacenada en la BBDD sea el mismo hash que el que realizamos mas arriba hasheando la contraseña que nos facilita el usuario.

Si ambas comprobaciones son correctas, muestra una página que será la pagina principal, pero en el caso de que la contraseña no sea correcta lanza por pantalla un mensaje de ERROR sin especificar que la contraseña es incorrecta ya que lo veo un fallo de seguridad. Además, en caso de que la SELECT no encuentre nada al realizar la búsqueda en la BBDD le vamos a indicar que saque por pantalla un mensaje que indique error de usuario.

En este archivo también tenemos las mismas excepciones que antes y finalizamos de la misma forma.



# **Conclusión y explicaciones**

Esta practica me ha parecido muy interesante ya que hemos comprobado a través de una librería de Python como se pueden tratar los datos que se ingresan por el cliente a través de una interfaz gráfica.

Con esta practica hemos tenido que modificar y preparar el archivo de configuración de Apache.

## **Modificaciones en httpd.conf de Apache**

### **Módulo de CGI**

Cargar el módulo responsable de manejar la ejecución de scripts CGI en el servidor web Apache.



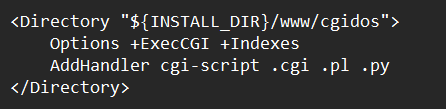
### **Crear Directorio**

Creamos la configuración para el directorio donde va a estar alojado nuestra pagina

**+ExecCGI:** indica que los scripts CGI en este directorio pueden ser ejecutados

**+Indexes**: permite la generación de listas de directorios si no se encuentra un archivo de índice.

**AddHandler cgi-script .cgi .pl .py:** cuando un cliente solicite un archivo con una de estas extensiones en este directorio, el servidor tratará el archivo como un script CGI y lo ejecutará en lugar de servirlo como un archivo estático.



### **Crear Directorio**

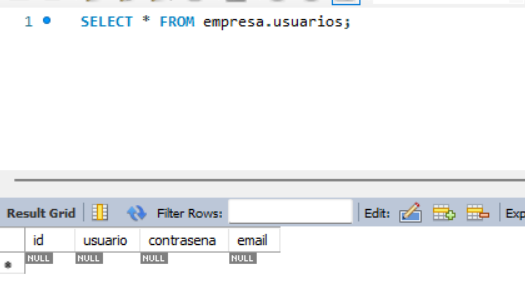
Configurar para que cuando un cliente solicite uno de estos archivos sean manejados como scripts CGI por el servidor web Apache.



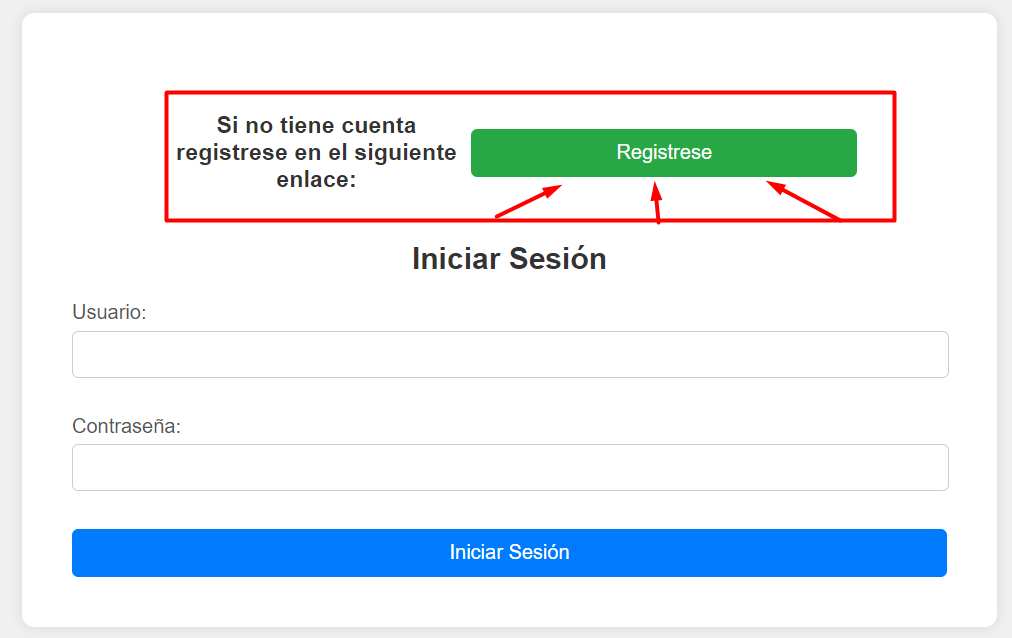
Para esta práctica me han tenido que ayudar varios compañeros además de buscar información por internet y pedir información y ayuda de configuración a inteligencia artificial además de preguntar a personas que se dedican al desarrollo de aplicaciones web.

# **Pruebas de que todo funciona correctamente y errores**

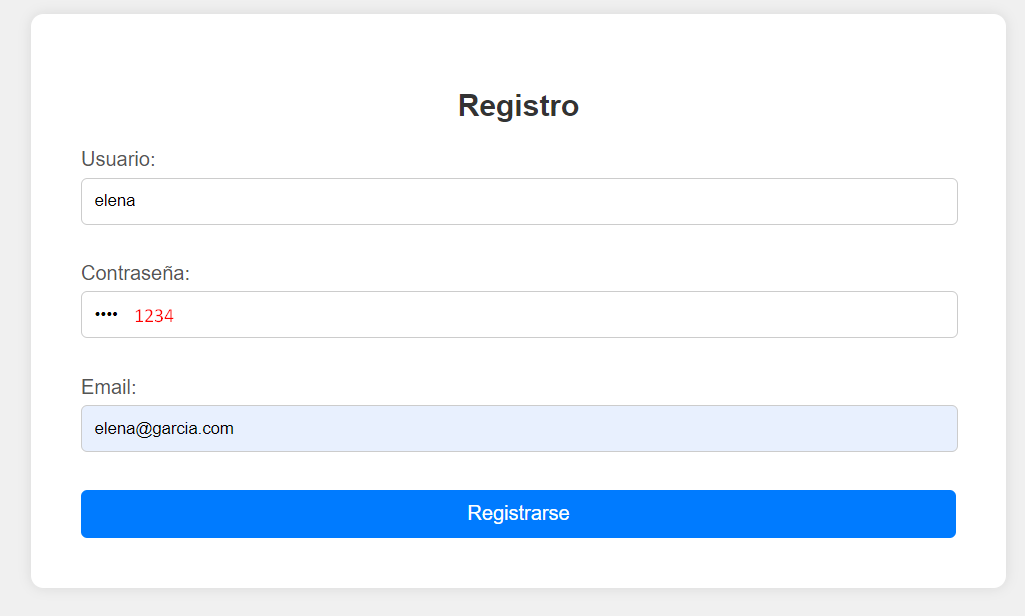
La base de datos no tiene ningún usuario, vamos a registrar uno



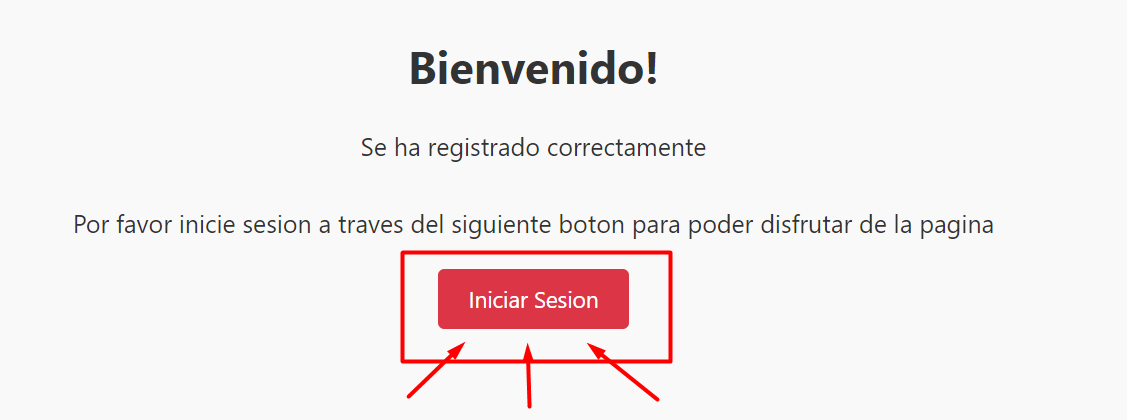
Accedemos a la pagina y como no tenemos cuenta pulsamos en Registrese



Rellenamos los datos que nos solicitan y pulsamos en Registrarse

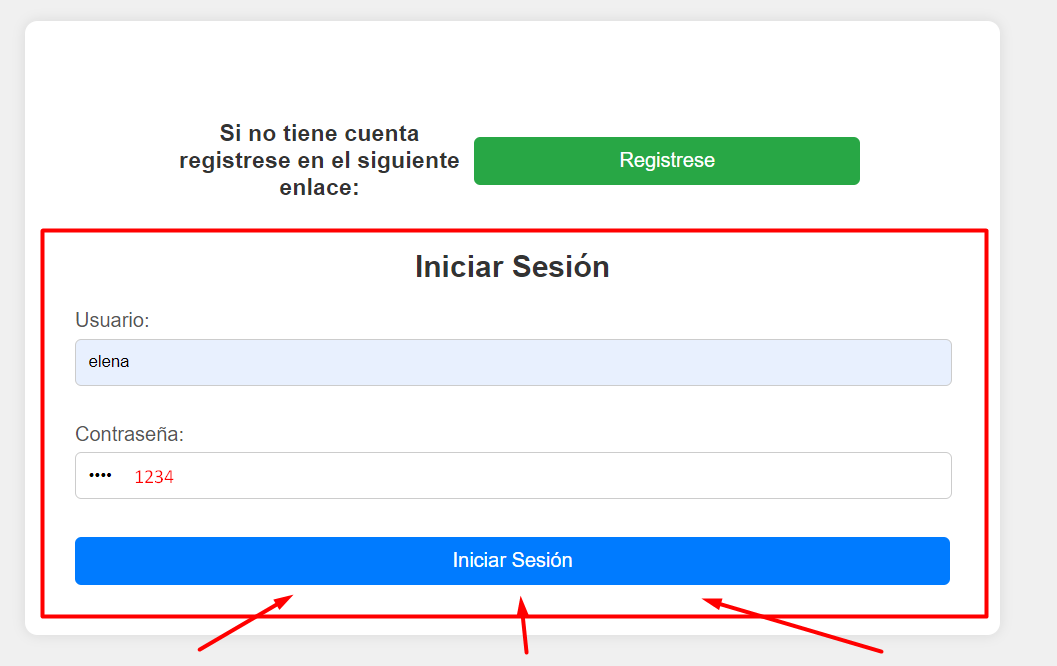


Cuando le damos nos sale una interfaz con un texto y un botón, pulsamos en iniciar sesión y ese botón redirige de nuevo a index.html

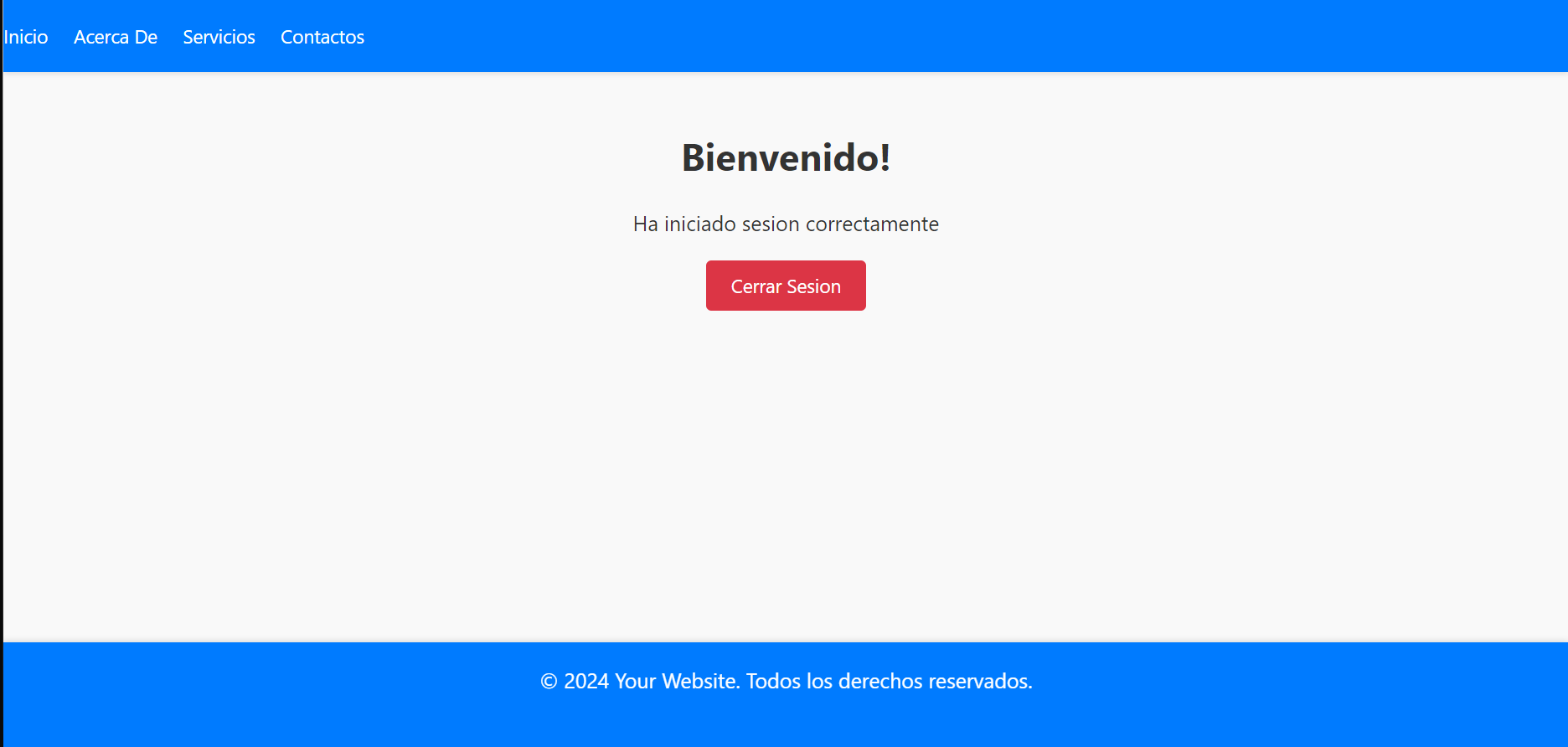


Rellenamos los datos (usuario y contraseña) y pulsamos en iniciar sesión.

Automáticamente cargara la pagina



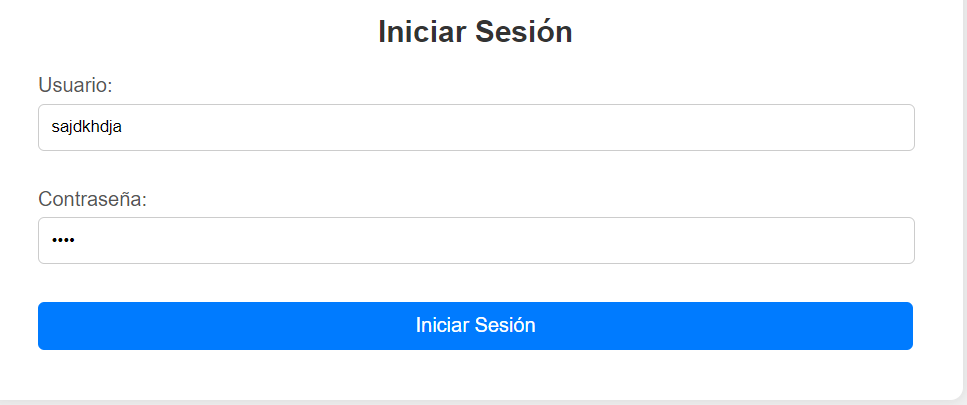
Esta es la página que carga

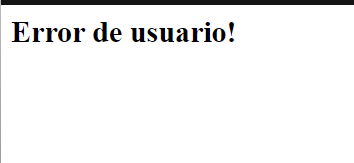


AHORA VAMOS A PROBOCAR FALLOS PARA VER LOS ERRORES QUE TENEMOS

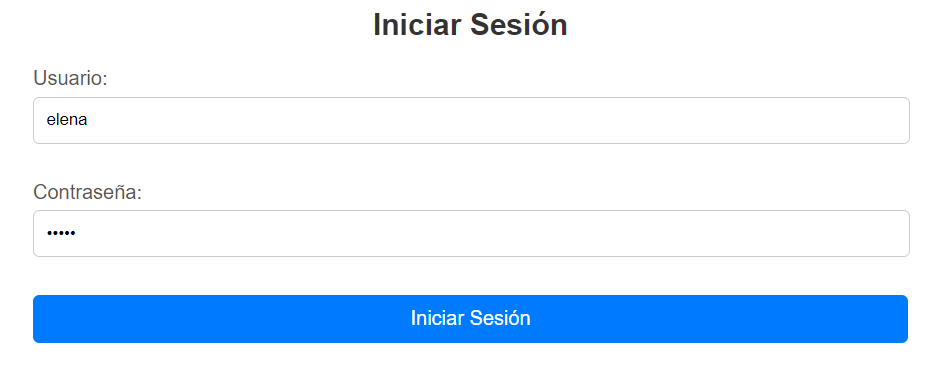
El primer error es de usuario y contraseña incorrectos al iniciar sesión

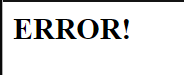
***Error de usuario***





***Error de contraseña***





El segundo error es de intentar registrar el mismo usuario

