UNIVERSIDAD CENTRAL DEL ECUADOR

FACULTAD DE INGENIERÍA Y CIENCIAS

SERVICIO WEB Y APLICACIÓN ASP.NET

DOCUMENTACIÓN DE USO

SERVICIO GENERADOR Y GESTOR DE CONTRASEÑAS SEGURAS

Servicio Web Generador y Gestor de Contraseñas Seguras

Descripción general del servicio

El gestor de contraseñas es un servicio web que permite a los usuarios crear y guardar contraseñas seguras para proteger sus cuentas en línea. El servicio utiliza algoritmos de cifrado y técnicas de hashing para proteger la información de la contraseña del usuario.

El proceso de creación de una contraseña segura comienza con la definición de la longitud de la contraseña y la selección de los caracteres especiales y números que se incluirán. El servicio utiliza un generador de contraseñas aleatorias con una longitud estándar mínima de 8 caracteres, extendiéndose hasta los necesarios, para crear una contraseña que cumpla con los criterios establecidos.

Además, el servicio cuenta con las operaciones CRUD necesarias para el acceso, registro y actualización de usuarios consumidores del servicio, además de las mismas operaciones CRUD, pero enfocadas al almacenamiento de las contraseñas escogidas y almacenadas en el Gestor

De modo que La cuenta de usuario se puede proteger con una contraseña maestra fuerte, que es la única que el usuario debe recordar para acceder a todas las contraseñas almacenadas en la plataforma.

Como utilizarlo

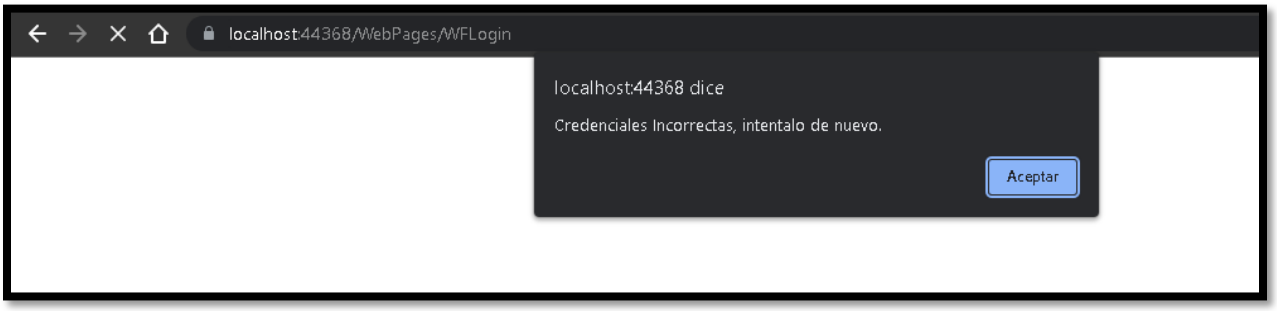
1 .- Acceso de Usuario

De entrada, el usuario se encontrará con la ventana de Login o Acceso de usuario:

En la que para entrar al sistema ingresará su correo electrónico y contraseña previamente registrados, en los que se validará que estos campos no estén vacíos para su comprobación en la base de Datos, además de que el campo Correo Electrónico cuenta con la validación de solo admitir formatos de email, de modo que el usuario estará obligado a llenar los campos.

Como resultado del intento de ingreso se podrán dar 2 eventos:

1.- Si las credenciales son incorrectas reflejará el mensaje “Credenciales incorrectas”, dando más oportunidades de acceso:



2.- Si las credenciales son correctas el usuario accederá a la página principal de sesión de usuario.

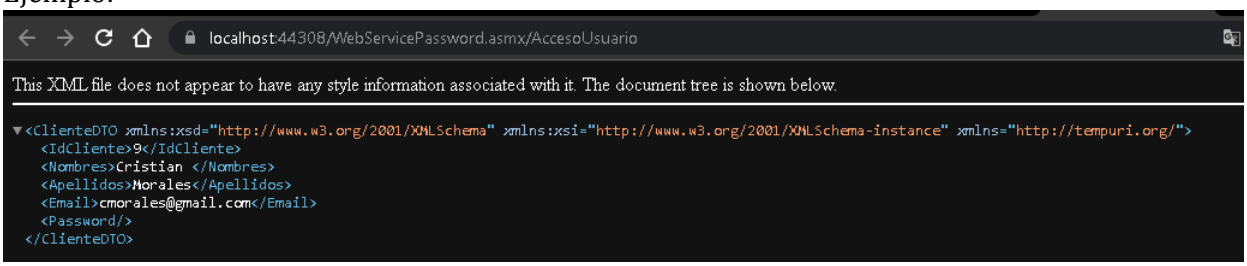
Referente al Consumo del Servicio Web:

Para poder ejecutar esta operación de acceso se consume el servicio Web que tiene la siguiente estructura:

```
[WebMethod]
0 referencias
public ClienteDTO AccesoUsuario(string email, string password)// login
{
    //variable que almacena la respuesta de la capa Service
    ClienteDTO clienteEncontrado;
    // método ClienteLogin de la capa ClienteService para procesar la información de entrada y acceder al sistema
    clienteEncontrado = this.serviceCliente.ClienteLogin(email, password);
    return clienteEncontrado;
}
```

El método se denomina AccesoUsuario que retorna un objeto de tipo ClienteDTO con el objetivo de usar su información durante la sesión de usuario, recibe como parámetros el email, y la contraseña con valores primitivos string, entonces si la capa de servicio encuentra al usuario este lo retorna junto a su información, por el contrario, simplemente retornará un objeto nulo.

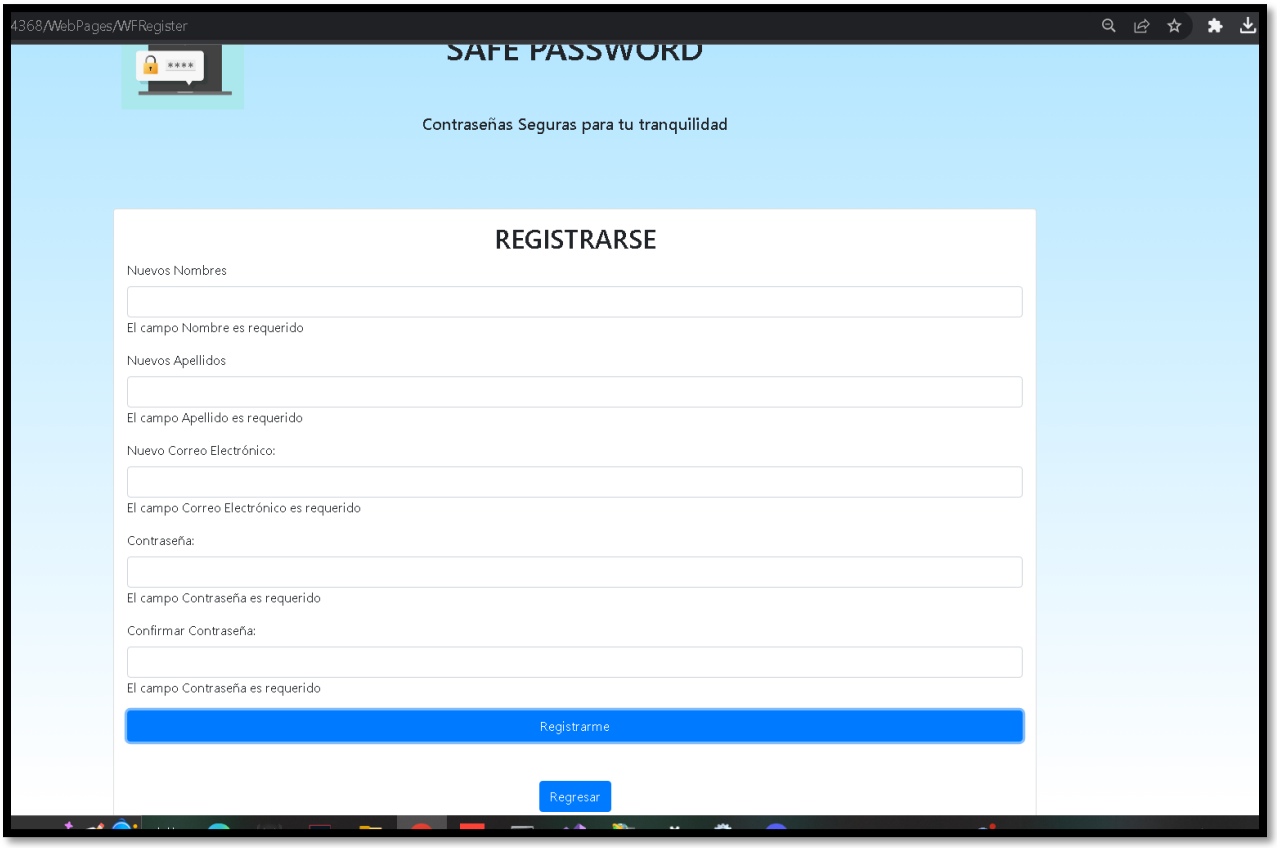
Ejemplo:



Es necesario Recalcar que el servicio Web envía la información del cliente, omitiendo su contraseña, para mejorar la seguridad.

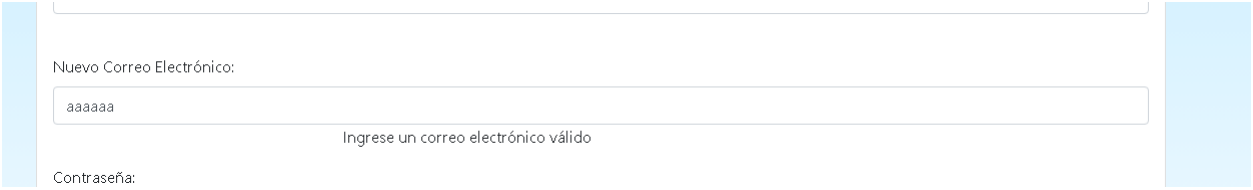
2 .- Registro de usuario

En la parte de registro el usuario tiene que llenar todos los campos de manea obligatoria caso contrario no podrá registrarse tiene que llenar parámetros de: nombres, apellidos, correo electrónico y contraseña.

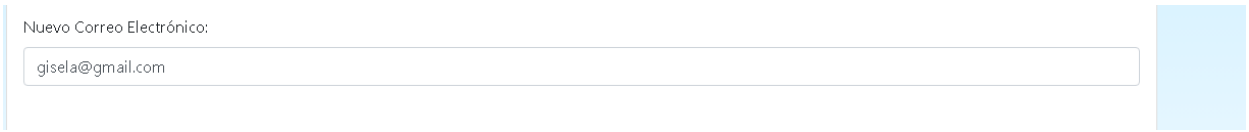


Correo electrónico:

El usuario debe escribir un correo electrónico valido para poder continuar con su registro es decir tiene que contar con los parámetros de: @gmail.com o @hotmail.com



Esta es la manera correcta el usuario debe escribir su correo electrónico:

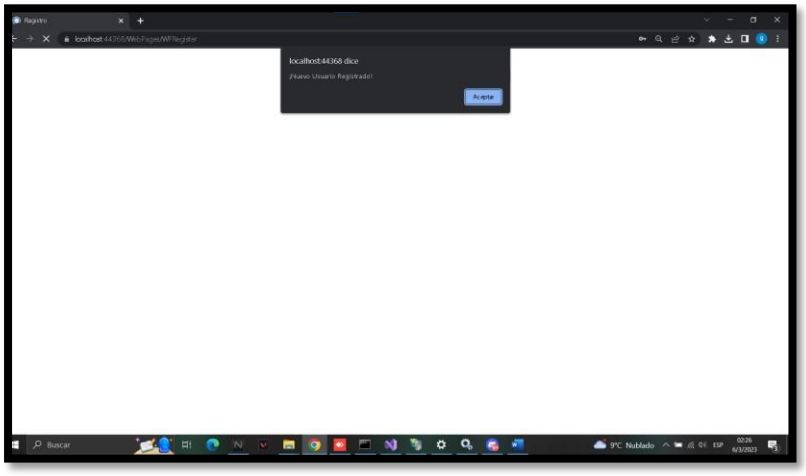


Contraseña:

El usuario no puede dejar el campo de contraseña vacío y los otros campos llenos ya que no podrá registrarse

Finalmente:

Una vez que el usuario llene todos los campos tiene que dar click en el botón registrarme y se cargará una ventana con el mensaje de “nuevo usuario registrado” y el usuario ya podrá ir a iniciar sesión



Referente al Consumo del Servicio Web:

Este es un método que recibe un objeto ClienteDTO llamado clienteRegistro como parámetro y devuelve un valor booleano.

Dentro del método, se declara una variable bool llamada respuesta, que se utilizará para almacenar la respuesta del método ClienteRegistro de la capa ClienteService.

Luego, se llama al método ClienteRegistro de la capa ClienteService, pasando como parámetro el objeto clienteRegistro recibido. La respuesta del método ClienteRegistro se asigna a la variable respuesta.

Finalmente, se devuelve la variable respuesta como resultado del método RegistroUsuario.

```
[WebMethod]
0 referencias
public Boolean RegistroUsuario(ClienteDTO clienteRegistro)
{
    //variable que almacena la respuesta de la capa Service
     bool respuesta;
     // método ClienteRegistro de la capa ClienteService para procesar la información de entrada
       respuesta = this.serviceCliente.ClienteRegistro(clienteRegistro);

     return respuesta;
}
```

3.- Sesión de usuario

Para la sesión de usuario el cliente contará con la siguiente interfaz:

Como primer punto obtenemos los datos de quien ha iniciado una sesión acompañado del texto "Bienvenid@" esto con el objetivo de comprobar que estamos en una sesión de usuario, además posee un navegador, con las opciones actualizar información y opción de cerrar la sesión actual.

Tenemos 2 secciones la de la izquierda pertenece a la funcionalidad de generación de contraseñas seguras, donde el usuario podrá personalizar las contraseñas a su comodidad, desde solo tener caracteres, a tener caracteres especiales y números, además de una longitud personalizada con un mínimo de 8 caracteres hasta 32 para lograr almacenarlos en su banco de contraseñas.

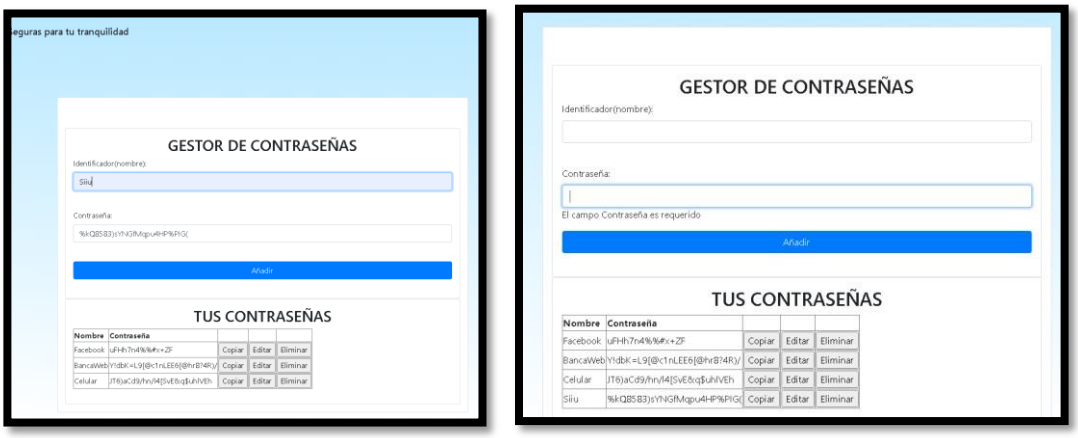
Generador de contraseñas seguras:

Esta sección esta conformada por un campo no editable, llamado "tu contraseña" el cual dará el resultado de la contraseña, con la opción de copiar su contenido directo al registro de contraseña del cual se hablará más adelante.

Posteriormente encontramos las opciones de personalización de nuestra contraseña, en ella el usuario podrá elegir que elementos quiero incluir en su contraseña, además de la longitud necesaria.

Por otro lado, tenemos el botón Fácil de recordar, y esta nos generará una contraseña conformada de 2 palabras y 2 números de 2 dígitos como máximo de este modo el usuario también tendrá la oportunidad de recordar su contraseña.

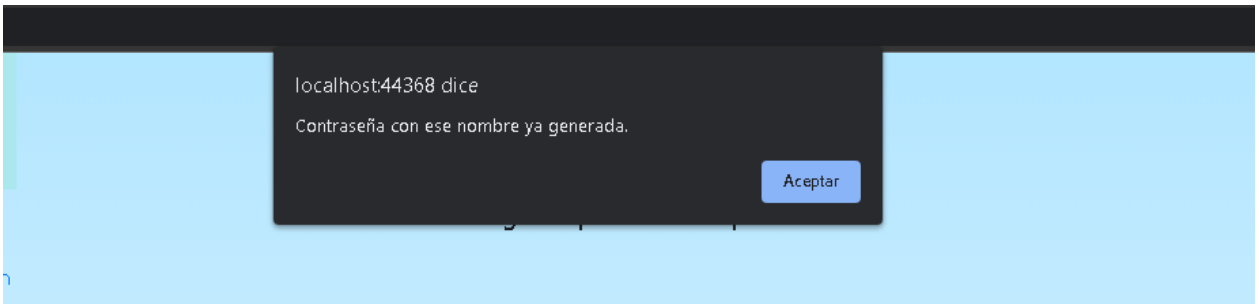
Pasando a la otra Sección:



Con el botón Copiar junto al resultado del generador, podremos enviar directamente la contraseña de elección al campo para registrarlo, y presionando el botón añadir podremos registrar nuestra nueva contraseña asociada a un nombre especifico, cabe recalcar que para realizar un nuevo registro se verifica que los campo no estén vacíos.

Errores a tomar en cuenta:

Si un usuario ingresa una contraseña con un nombre similar al de algún registro, el sistema deniega el nuevo registro, con el siguiente mensaje.



Referente al Consumo del Servicio Web:

Para la funcionalidad de Generar una contraseña fácil de recordar utilizamos el método GenerarPasswordFacilRecordar, que no solicita ningún parámetro , y que nos devuelve una cadena compuesta de 2 palabra y 2 números.

```
[WebMethod]
0 referencias
public string GenerarPasswordFacilRecordar()
{
    String passWordResultante = this.generadorPassword.PasswordFacilRecordar();
    return passWordResultante;
}
```

Para la funcionalidad de Generar una contraseña Personalizada utilizamos el método GenerarPassword, en el que enviaremos como parámetro, la longitud solicitada por el usuario, estados de verdadero o falso para los campos números, y símbolos de modo que según la elección desde la vista retornará un resultado deseado

```
[WebMethod]
0 referencias
public string GenerarPassword(int longitud, Boolean numeros, Boolean simbolos)
{

    String passWordResultante = this.generadorPassword.GenerarPassword(longitud, simbolos, numeros);
    return passWordResultante;

}
```

La funcionalidad de Gestor de contraseñas incluye los siguientes métodos del Web Service:

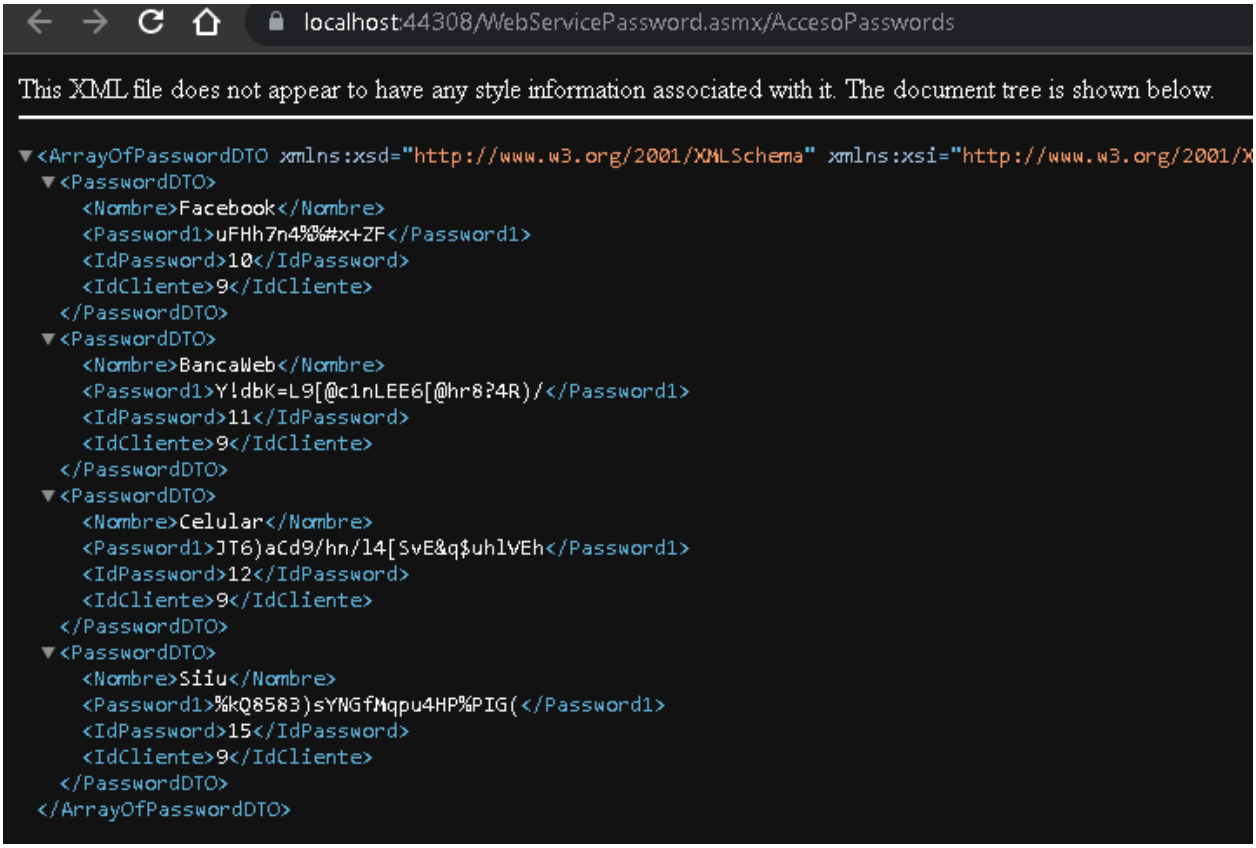
Para El registro de contraseñas el método RegistroPassword que recibe como parámetro un objeto de tipo PasswordDTO, y que retorna un valor booleano según el éxito en el registro de la nueva contraseña, por lo tanto, el método devolverá true siempre que se haya ejecutado la operación CRUD de inserción, por el contrario, si devuelve un valor false significa que no ha registrado la nueva contraseña, ya que el nombre para ese registro ya existe.

```
public bool RegistroPassword(PasswordDTO passwordRegistro)
{
    bool respuesta = false;
    // método PasswordRegistro de la capa PasswordService para procesar la información de entrada
    respuesta = this.servicePassword.PasswordRegistro(passwordRegistro);
    return respuesta;
}
```

Para mostrar los registros de contraseñas de un usuario se utiliza el siguiente método, que requiere como parámetro el ID del cliente en sesión, y que nos retornará una lista con todas aquellas contraseñas registradas

```
[WebMethod]
0 referencias
public List<PasswordDTO> AccesoPasswords(int idCliente)// login
{
    //variable que almacena la respuesta de la capa Service
    List<PasswordDTO> passwords;
    // método PasswordBuscar de la capa PasswordService para obtener las contraseñas de un cliente
    passwords = this.servicePassword.PasswordBuscar(idCliente);
    return passwords;
}
```

Dando un resultado asi:



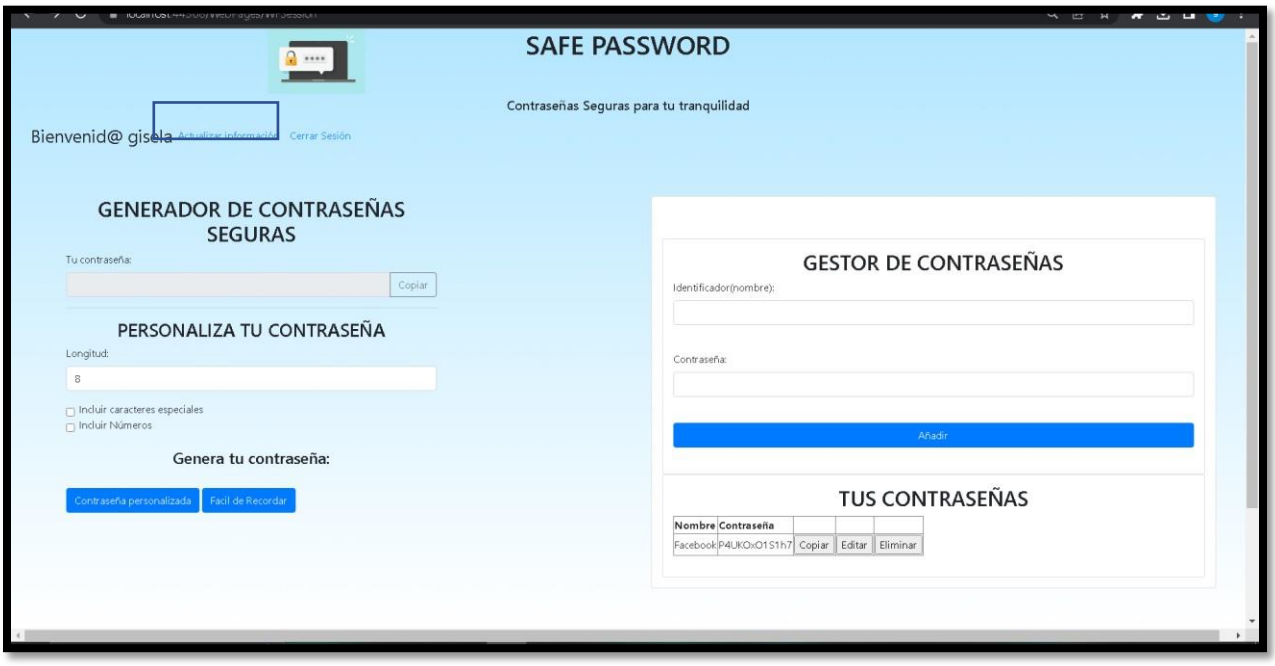
Para la actualización y eliminación de un registro de contraseña utilizamos los siguientes métodos:

```
[WebMethod]
0 referencias
public void EditarPasword(PasswordDTO passwordEdit)
{
    // método PasswordEditar de la capa PasswordService para actualizar el registro de una contraseña
    this.servicePassword.PasswordEditar(passwordEdit);

}

[WebMethod]
0 referencias
public void EliminarPassword(int idPassword)
{
    // método PasswordEliminar de la capa PasswordService para procesar la información de entrada
    this.servicePassword.PasswordEliminar(idPassword);

}
```

Los cuales una vez ejecutadas sus responsabilidades no retornan ningún valor.

4 .- Editar información

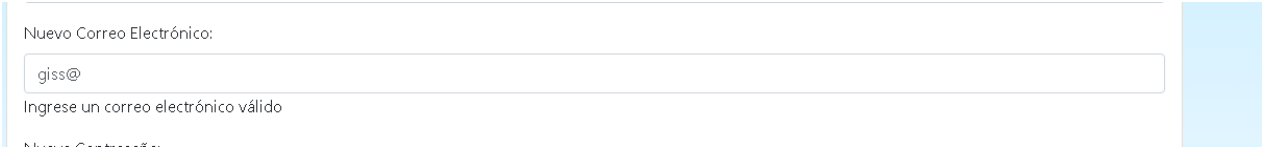
Una vez el usuario ingrese a su cuenta tiene una ventana de sesión con apartados de actualizar información al dar click en actualizar información le carga una página con el nombre editar información



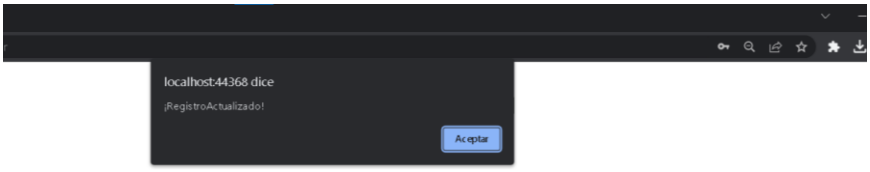
El usuario tendrá que llenar parámetros como: nombres, apellidos, nuevo correo electrónico, nueva contraseña, confirmar nueva contraseña y contraseña actual.

Los campos obligatorios que deben estar completos son los de nueva contraseña, confirmar nueva contraseña y contraseña actual.

El correo electrónico tiene que cumplir con los parámetros de tener @gmail.com o @hotmail.com caso contrario le saldrá un mensaje de "ingrese un correo electrónico válido"



Una vez llene todos los campos vacíos el usuario tiene que dar click al botón actualizar y le saldrá un mensaje de "registro actualizado"



Referente al Consumo del Servicio Web:

Este es un método que recibe un objeto ClienteDTO llamado cliente y una cadena de texto llamada passwordActual como parámetros, y devuelve un valor booleano.

Dentro del método, se declara una variable bool llamada respuesta, que se utilizará para almacenar la respuesta del método ClienteEditar de la capa ClienteService.

Luego, se llama al método ClienteEditar de la capa ClienteService, pasando como parámetros el objeto cliente y la cadena passwordActual. Este método procesará la información de entrada y actualizará el registro correspondiente.

La respuesta del método ClienteEditar se asigna a la variable respuesta. Finalmente se devuelve la variable respuesta como resultado del método EditarUsuario.

Es decir este método es responsable de llamar al método ClienteEditar de la capa ClienteService y devolver su respuesta. El propósito de este método es permitir la edición de un registro de cliente existente en el sistema, proporcionando la información de entrada necesaria para realizar la actualización y validar la contraseña actual del usuario que realiza la edición.

```
[WebMethod]
0 referencias
public Boolean EditarUsuario(ClienteDTO cliente, string passwordActual)
{
    //variable que almacena la respuesta de la capa Service
    bool respuesta;
    // método ClienteEditar de la capa ClienteService para procesar la información de entrada y editar el resgistro
    respuesta = this.serviceCliente.ClienteEditar(cliente, passwordActual);

    return respuesta;
}
```

Autenticación y autorización

Al estar el servicio web en periodo de pruebas y desarrollo, no existe una autorización, ni método de autenticación disponible para el público en general

Formato de datos

La manera de trasmitir los datos de este servicio web es mediante XML, debido a que es el método de preferencia para .NET

Seguridad

Contamos con algoritmos de encriptación unidireccionales para la seguridad de la clave maestra, mientras que para el almacenamiento del gestor se ha implementado algoritmos de encriptación bidireccionales, con el objetivo de resguardar la información de los clientes en la base de datos

En que se puede Aplicar nuestro servicio web:

Autenticación de dos factores: El gestor de contraseñas podría implementar la autenticación de dos factores para proporcionar una capa adicional de seguridad. Los usuarios pueden habilitar esta opción y recibir un código de autenticación único en su dispositivo móvil para completar el proceso de inicio de sesión.

Integración con navegadores web: El gestor de contraseñas podría integrarse con navegadores web para facilitar el inicio de sesión en línea. Los usuarios pueden guardar sus contraseñas en el gestor de contraseñas y permitir que el navegador web las utilice automáticamente al acceder a un sitio web.

Empresas: Las empresas pueden utilizar un gestor de contraseñas para mantener seguras las contraseñas de sus empleados y proteger sus datos confidenciales. Los administradores de la empresa pueden crear cuentas de usuario para los empleados y asignarles permisos específicos para acceder a ciertas contraseñas.

Familias: Las familias pueden utilizar un gestor de contraseñas para mantener seguras las contraseñas de sus cuentas en línea. Los padres pueden crear una cuenta de gestor de contraseñas y compartir las contraseñas con sus hijos para que puedan acceder a las cuentas familiares de manera segura.

Usuarios individuales: Los usuarios individuales pueden utilizar un gestor de contraseñas para almacenar y proteger sus contraseñas en línea. El gestor de contraseñas puede proporcionar una capa adicional de seguridad al autenticar la identidad del usuario a través de la autenticación de dos factores.

Equipos de trabajo: Los equipos de trabajo pueden utilizar un gestor de contraseñas para compartir contraseñas entre los miembros del equipo y mantener la información de la empresa segura. Los administradores pueden asignar permisos específicos a los miembros del equipo para acceder a ciertas contraseñas.