PROCESO DE GESTIÓN DE FORMACIÓN PROFESIONAL INTEGRAL

FORMATO GUÍA DE APRENDIZAJE

1. IDENTIFICACIÓN DE LA GUIA DE APRENDIZAJE

- **Denominación del Programa de Formación**: Análisis y Desarrollo de Software ADSO.
- **Código del Programa de Formación:** 228118
- **Ficha:** 2931558
- **Nombre del Proyecto**: Sistema de Información para manejo de análisis físico químico y microbiológico de calidad de aguas para consumo humano para laboratorio de aguas ubicado en el centro de gestión agroempresarial del oriente (Código 1139209).
- **Fase del Proyecto**: Planeación
- **Actividad de Proyecto**: Diseñar la estructura tecnológica del sistema de información.
- **Competencia**: Diseñar el sistema de acuerdo con los requisitos del cliente.
- **Resultados de Aprendizaje Alcanzar**: Aplicar políticas y mecanismos de control en el diseño del sistema de información, mediante el análisis de la vulnerabilidad de la información, siguiendo los parámetros establecidos por la organización.
- **Duración de la Guía**: 30h presencial + 12h LMS

2. PRESENTACION

En esta actividad de aprendizaje usted diseñará los mecanismos de seguridad como password y control del sistema de información *como las sesiones en su proyecto.*

Existen muchas definiciones del término seguridad. Simplificando, se puede definir la seguridad como la "Característica que indica que un sistema esta libre de todo peligro, daño o riesgo." Villalón (2007).

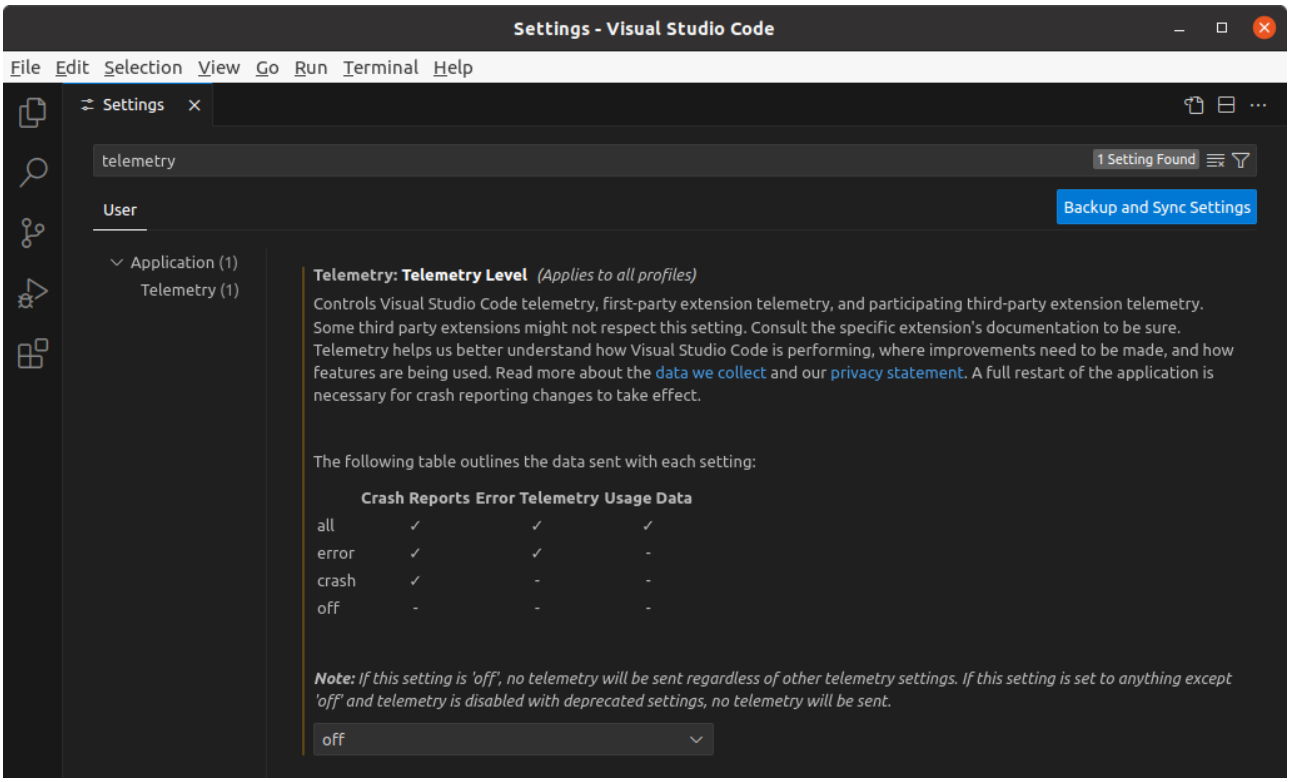
En toda actividad se hace necesario, no solo planear y ejecutar las actividades, sino efectuar procedimientos de control que vayan encaminados a asegurar que dichas actividades han sido ejecutadas de acuerdo a los parámetros que se habían establecido con anterioridad.

Es por ello, que para Evaluar la seguridad de los sistemas de información se requiere que en las diferentes fases del ciclo de vida de los sistemas de información, se planteen protocolos claros que permitan lograr un buen nivel de calidad en el software.

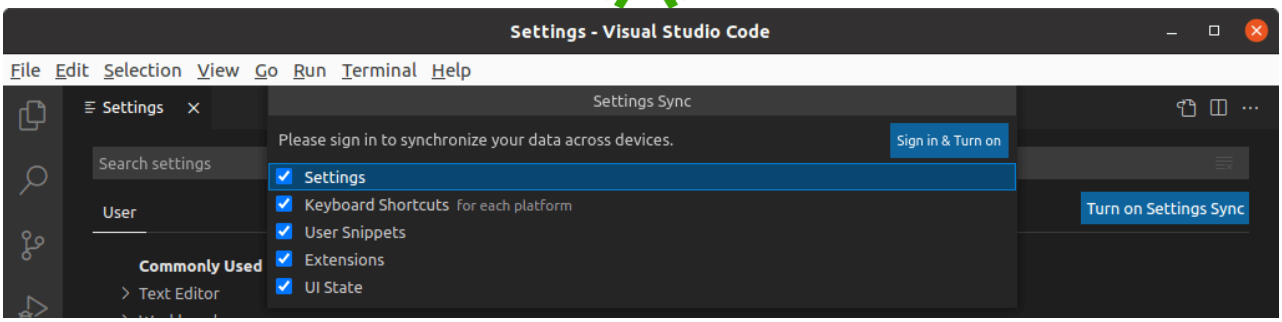
Para el diseño de estos mecanismos de seguridad y control, se debe ir de la mano de la seguridad informática, es por ello que en este tema tocaremos varios conceptos claves de este tema tan de moda actualmente, para evitar caer en la "inseguridad informática".

Pero antes vamos a configurar nuestro visual studio code para pyton el cual dare a conocer en las siguientes paginas.

2.1. Cuando entramos por primera vez a VS Code, nos pide con qué tema queremos usar el programa (a elegir entre claro, oscuro o alto contraste). También nos notifica sobre la recolección de datos de uso por parte de Microsoft. Si no queremos compartir esta información podemos desactivarla accediendo a **File** → **Preferences** → **Settings** (o con el atajo de teclado **Ctrl+**), buscar *telemetry* en la barra de búsqueda y seleccionar la opción *off* en el menú desplegable de la parte inferior.



Si tenemos pensado usar VS Code en distintos ordenadores es buena idea utilizar la sincronización de preferencias. De este modo no vamos a tener que repetir el proceso de configuración en cada nueva instalación, aunque para ello debemos tener una cuenta de Microsoft o de GitHub. Para activar esta característica accedemos igualmente a **Settings** y pinchamos el botón **Turn on Settings Sync**. Se nos abre una ventana con un listado de los ajustes que podemos sincronizar y el botón **Sign in & Turn on**, el cual al pincharle nos pide si para la sincronización vamos a usar una cuenta de Microsoft o de GitHub.



Además, si al iniciar VS Code tenemos nuestra computadora en español nos aparece momentáneamente una notificación preguntándonos si queremos instalar el paquete de idioma español. En caso de no hacerlo no pasa nada ya que el idioma es en realidad una extensión que podemos instalar en cualquier momento. Al final del post te explico cómo hacerlo.

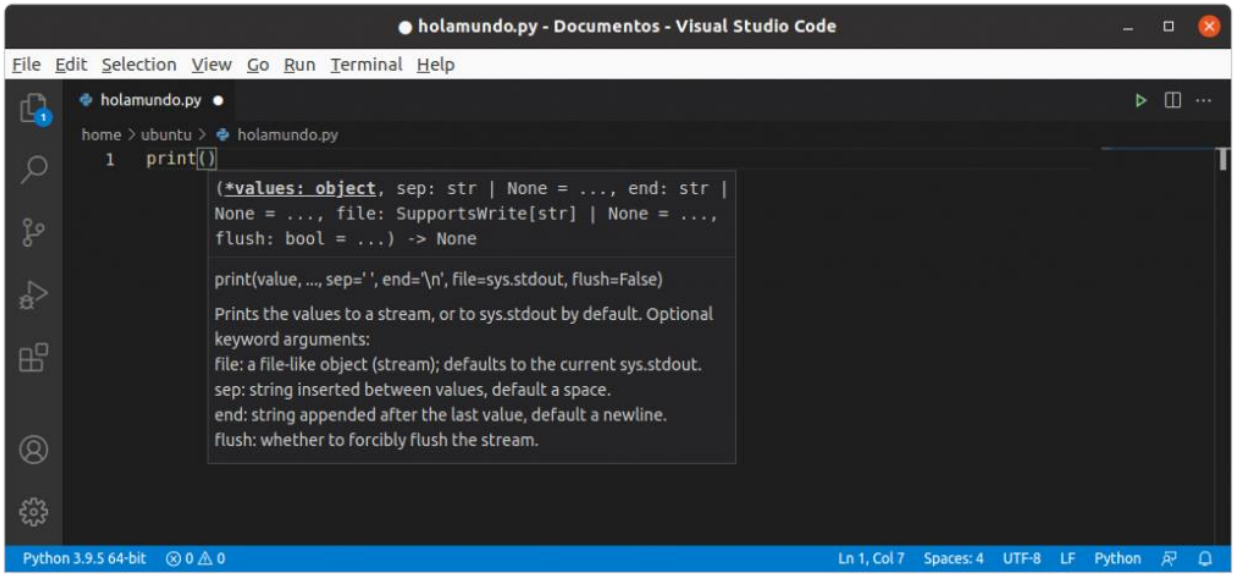
Extensiones recomendadas

Instalar extensiones a VS Code es tan fácil como pinchar en el botón correspondiente del menú de la izquierda o acceder directamente con el atajo de teclado **Ctrl+Shift+X**. Esto nos abre una nueva sección en la parte izquierda del programa que contiene un buscador, para que podamos buscar extensiones por su nombre, y un listado de extensiones agrupadas en tres categorías: instaladas, populares y recomendadas. Si vas a programar en Python a continuación te voy a dar mis recomendaciones.

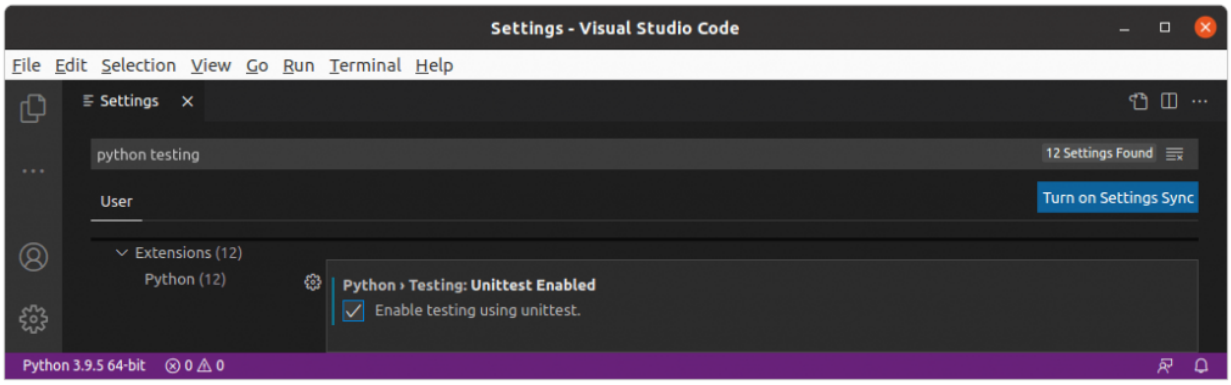
Python

Esta extensión, desarrollada por Microsoft, añade a VS Code muchas funcionalidades relacionadas con Python como el autocompletado y formateo del código, herramientas para hacer *debugging* y testear nuestro código Python, y la gestión de entornos virtuales, entre otras.

Una vez instalada, al trabajar con archivos .py, nos indica la versión de Python que tenemos instalada en la parte inferior izquierda. En mi caso, como se ve en la imagen inferior, la versión que tengo es la 3.9.5 64-bit. Además, mientras programamos nos proporciona ayuda sobre los parámetros que aceptan las funciones, tal y como se muestra a continuación.

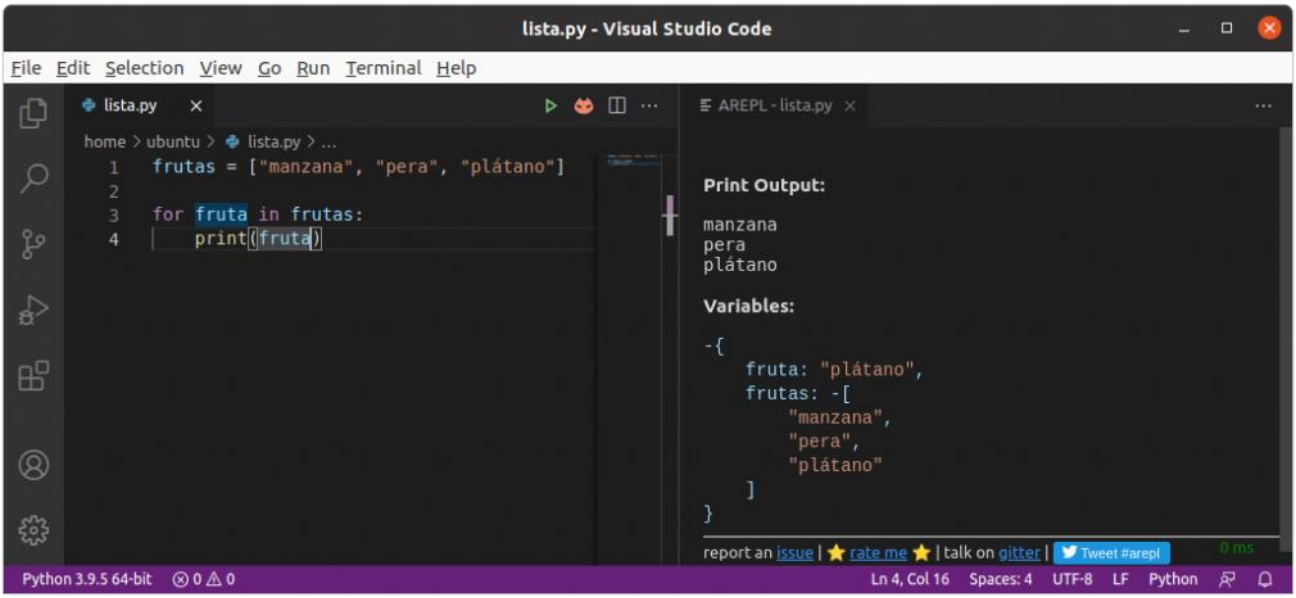


Otra consideración a tener en cuenta es activar la herramienta de testeo que vayamos a utilizar, a escoger entre: unittest, pytest o nose. Esto lo podemos hacer fácilmente accediendo a **Settings** y teclear python testing en la barra de búsqueda. En mi caso como utilizo unittest marco la casilla correspondiente para activarlo.



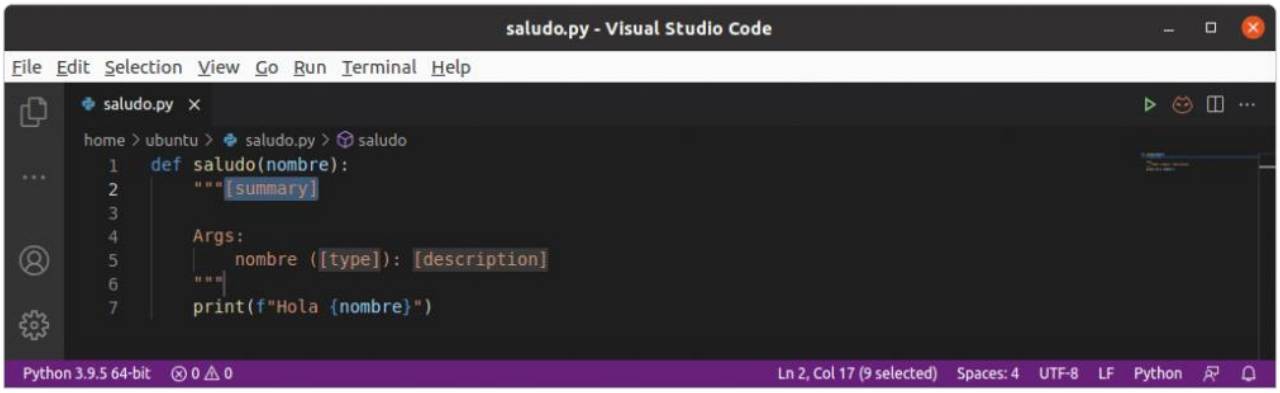
AREPL for Python

Esta extensión es muy interesante ya que evalúa nuestro código mientras lo escribimos. Al instalarla aparece un icono en la parte superior derecha del editor, y al pinchar sobre él se nos abre una ventana donde van apareciendo los resultados. En el siguiente ejemplo vemos el resultado que nos da al iterar una lista.



Python Docstring Generator

Esta extensión, desarrollada por Nils Werner, hace exactamente lo que dice, es decir, nos ayuda a generar de manera eficiente *docstrings* de nuestros métodos y funciones. Para usarla sólo hay que abrir triples comillas dobles en la primera línea de nuestro método o función y pulsar intro. Esto nos genera una plantilla con los parámetros a rellenar, y que podemos recorrer fácilmente con la ayuda del tabulador. Se trata sin duda de una extensión imprescindible para mejorar nuestra productividad a la hora de documentar código.

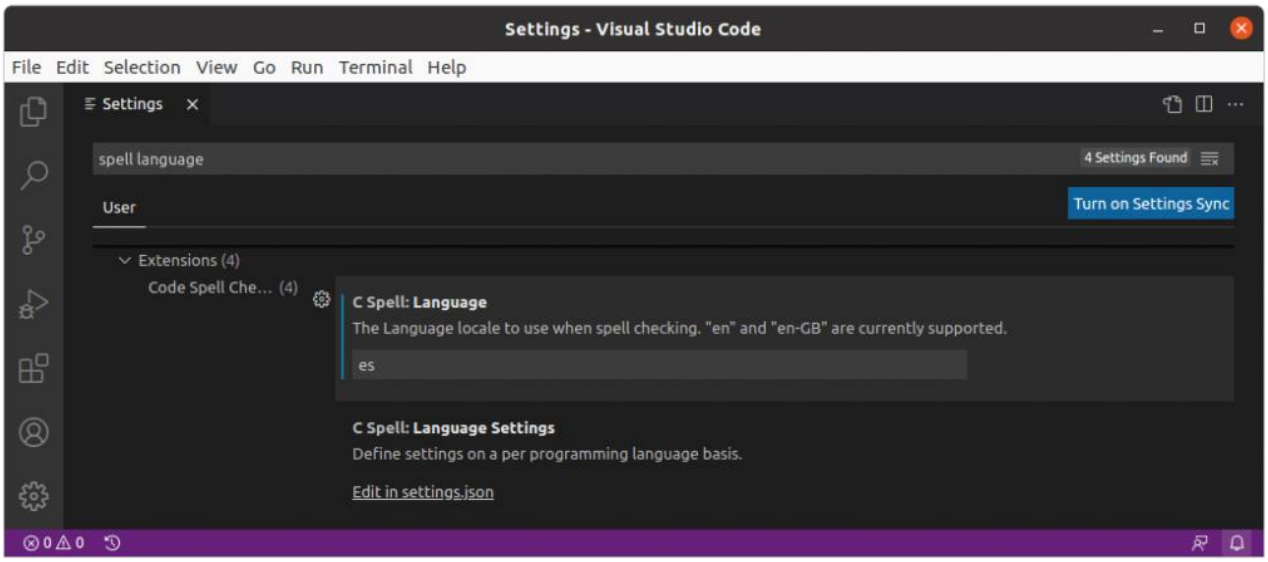


Configuración de la interfaz en español

VS Code se instala por defecto en inglés, pero si lo deseamos podemos instalarle una extensión para cambiar el idioma al español. Para ello sólo tenemos que acceder al

menú de instalación de extensiones, buscar *spanish* e instalar la extensión desarrollada por Microsoft.

Adicionalemnte, si al programar usamos nombres de variables en español, podemos instalar la extensión **Spanish – Code Spell Checker**, que lo que hace es indicarnos errores ortográficos en nuestro código. Una vez instalada hay que indicarle que use el diccionario español (es) indicándolo en el ajuste *C Spell: Language* tal y como se muestra a continuación.



3. FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE

3.1 Descifre (desencripte) la siguiente información:
APRENDIZ DESENCRIPTE LAS SIGUIENTES LINEAS PUEDE USAR JAVA O PHP O C SHARP
ZKIVMWRA WVHVMXIRKGV OZH HRTFRVMGVH ORMVZH KFVWV FHZI QZEZ L KSK L X HSZIK
OZ XLMURWVMXRZORWZW RMULINZXRLM HLOL KFVWV HVI IVEVOZWZ Z FHFZIRLH
ZFGLIRAZWLH
OZ RMGVTIRWZW VH OZ NLWRURXZXRLM WV RMULINZXRLM WVYV HVI XLMGILOZWZ
OZ WRHKLMRYRORWZW VH OZ ZHRTMZXRLM WV ILOVH KZIZ ZXXVWVI Z RMULINZXRLM
Una vez conocido el algoritmo cifre (encripte) la siguiente información:
ABCDEFGHIJKLMNOPQRSTUVWXYZ
LA INFORMACION ES IMPORTANTE PARA EL CLIENTE ES MEJOR PROTEJERLA
Intente generar el algoritmo de cifrado y descifrado en Java o PHP o C#
En java puede usar texto.charAt(3);
En PHP puede usar $texto{3} o $texto[3] y la función ord($letra);
En C# puede usar texto[3];
Tip: Puede buscar en internet algoritmo Cesar
Pista: 'A' + 'Z' - letra

3.2 Haga una lectura comprensiva del siguiente texto:
La protección de los datos como claves o contraseñas se puede dar desde el mismo servidor de base de datos o desde el mismo lenguaje de programación, en ambos utilizando hash (funcion de resumen o de picadillo), generalmente se prefiere desde el mismo lenguaje de programación ya que el control del logs (registros de uso) lo debe hacer el mismo programador mientras que el servidor dependiendo de la configuración lo hace de manera automática y algún log podrá contener la contraseña del cliente en texto claro, es decir, sin haberle aplicado la función hash.

Hash desde MySQL

La funcion hash SHA2(clave,k) – donde el segundo argumento tiene que ser 224 o 256 o 384 o 512 ej:
SELECT SHA2("C1av3De1U5uari0",224)
Retornará: 2ea70d238a20e10f9c383fa70257e0fac7f162c9bb1cee09572b86e8

Por lo anterior si va a crear una tabla que va a guardar la clave o contraseña del usuario con hash 224 debe crear un campo VARBINARY(28) o BLOB(28) para guardar la clave ej. (obviamente faltan mas campos)
EJ.

idUsuario	correoUsuario	claveUsuario	nombreUsuario
1	sofita@misena.edu.co	2ea70d238a20e10f9c383fa70257 e0fac7f162c9bb1cee09572b86e8	Sofia Rodriguez

(Para hash 256 el campo tendrá que ser de 32 – que sale de dividir 256 entre 8-, para hash 384 el campo de 48 y para hash 512 el campo de 64 de largo)

Es decir, la captura basica de datos podría darse así:

```
<form action="controladorLogin.php" method="post">
<input type="email" placeholder="ejemplo@correo.co" required>
<input type="password" placeholder="123456" required>
<input type="submit" value="Ingresar">
</form>
```

Y en alguna línea del login.php (en el modelo) podría quedar asi:

```php
<?php
$verificacion = mysqli_query("SELECT nombreUsuario
FROM Usuarios
WHERE (correoUsuario LIKE $this->_correoDelFormulario)
AND (claveUsuario = SHA2($this->_claveDelFormulario,224))");
...
if(mysqli_num_rows($verificacion)){
    echo "Usuario si existe y la clave es correcta";
    echo "el programador podria hacer un return true";
    echo "en el controlador tendra que redireccionarlo al menu",
}else{
    echo "Usuario no existe o la clave es incorrecta"
    echo "el programador podría hacer un return false";
    echo "el controlador lo redireccionara al index"
}
?>
```

El correo se pide en caso que tengamos que volver a generarle una clave nueva al usuario (generalmente por que el usuario la olvida) esta clave nueva se le enviará al correo usando la funcion mailto() de php, en Java se podría usar javax.mail.jar, en C# se podría usar el servicio de mail.

3.3 Usando MySQL desde PHPMyAdmin (recuerde iniciar el servidor XAMPP) genere hash de 10 claves usando sha2 de 224 bits, 256 bits, 384 bits y 512 bits (es decir 40 hashes) en las claves mezcle letras con numeros y mayúsculas con minúsculas y llene la siguiente tabla:

Clave	SHA2(clave,224)	SHA2(clave,256)	SHA2(clave,384)	SHA2(clave,512)
SuperClavepj				

3.4 Cree una base de datos llamado encripcionUno, dentro de esa base de datos cree una tabla llamada Usuarios con los campos id (autoincrement), nombreUsuario, emailUsuario, claveUsuario, rolUsuario, celularUsuario y almacene 10 usuarios diferentes con claves diferentes tenga en cuenta que para el campo claveUsuario deberia ser VARBINARY(56) si va usar SHA2(clave,224) ej.

```
INSERT INTO Usuarios VALUES (null,'German Cruz', 'cruzgerman@sena.co', SHA2('superClav3',224), 1,
'3125144324')
```

3.5 Realice una lectura compresiva al documento oc_seguridad.pdf.

La generación de un hash para resumir la clave también es posible desde dentro del lenguaje de programación cada lenguaje de programación tiene su manera particular de implementar estos algoritmos de hashing algunos ya vienen incorporados como funciones

en el lenguaje mientras que en otros lenguajes se requiere de importar o usar librerias externas.

Hash desde PHP

Triturar contraseñas en php es muy sencillo ya que tiene una similitud con la función usada en MySql que ya fue vista en la anterior actividad solo hay que tener en cuenta que las funciones hash devolverán datos en binario por lo que si queremos observarlas entonces debemos usar la función bin2hex($clave_hashed):

```
<?php
    echo hash('sha256','SuperClaveLarga');
?>
```

Existe otra cantidad de algoritmos de hash como por ejemplo (cada uno genera una longitud diferente)
'md2', 'md4', 'md5', 'sha1', 'sha256', 'sha384', 'sha512', 'ripemd128', 'ripemd160', 'ripemd256', 'ripemd320', 'whirlpool', 'tiger128,3', 'tiger160,3', 'tiger192,3', 'tiger128,4', 'tiger160,4', 'tiger192,4', 'snefru', 'gost', 'adler32', 'crc32', 'crc32b', 'haval224,4', 'haval256,4', 'haval128,5', 'haval256,5', entre otros.

3.6 En brackets.io o visual studio code cree una pagina llamada resumenes.php y dentro de ella muestre aplique hash para 10 claves diferentes (no olvide iniciar su servidor XAMPP y también al archivo php agregar los tags HTML) ej.

```
<form>
    <label>La clave es: SuperClaveLarga </label>
    <input type="password" value="SuperClaveLarga">
<?php
    echo '<label>Hash = </label>'.hash('sha256','SuperClaveLarga');
?>
<br>
<input type="submit">
</form>
```

En la práctica el hash generado debe guardarse en la base de datos preferiblemente en un campo de la tabla Usuarios y no debe mostrarse al usuario final.

3.7 Opcional: Realizar una pequeña aplicación php en una pagina llamada generadorDeHash.php que tenga un form action="generadorDeHash.php" y dentro de este: tenga un input tipo password, un <textarea> para mostrar el hash y un input de tipo submit llamado "generar el hash".

Hash desde JAVA

La implementación en java requiere un poco más de código ya que debemos cargar primero el algoritmo de hashing, luego el password en valor String volverlo a bytes para que el algoritmo pueda digerir el password y entregar un hash de tipo byte.
Ej.

```
import java.security.NoSuchAlgorithmException;
import java.security.MessageDigest;
```

```
public byte[] generarHash(String clave) throws NoSuchAlgorithmException {
    MessageDigest sha256 = MessageDigest.getInstance("SHA-256");
    byte[] claveEnBytes = clave.getBytes();
    byte[] claveEnHash = sha256.digest(claveEnBytes);
    return claveEnHash;
}
//para mostrarlo nuevamente convertimos el hash de bytes en hexadecimal para mostrarlo
public void mostrarHash() throws NoSuchAlgorithmException {
    byte[] hashClave = generarHash("SuperClaveLarga");
    String hashParaMostrar="";
    for (byte cadaByte: hashClave){
        hashParaMostrar += String.format("%02X",cadaByte);
    }
    System.out.println("El hash ="+ hashParaMostrar);
}
```

Tip: Al momento de generar el hash puede agregar sal (en inglés salt) para mejorar la protección.

3.8 Haga una lectura compresiva del siguiente texto:
Hemos visto como proteger en cierta medida la clave y la aplicación web, si usamos logueo con usuario y contraseña usando hash, ahora bien, debemos empezar a proteger las funciones de la aplicación pues no queremos que nuestros usuarios accedan a todas las funciones de la aplicación web, es decir, debemos definir cuales usuarios podrá acceder a ciertas funciones y de alguna manera ocultar funciones que no queremos mostrar a otros usuarios, esto se le llama privilegios de acceso y se maneja mediante roles de acceso.

Manejo de Roles usando la variable $_SESSION en PHP

Lo primero es definir que tipos de usuarios vamos a tener: generalmente en una aplicación pequeña tendremos dos roles: el admin y el usuario.
El usuario: es aquella persona que tiene el privilegio de usar algunas funciones o acceder a algunos formularios de la aplicación web, enfatizando en que no puede acceder al formulario de manejo de todos los usuarios.
El admin: es aquella persona que tiene el privilegio de acceder a todas las funciones y de acceso a todos los formularios teniendo claro que el admin no puede editar o eliminar los registros de auditoria.

Tip: En aplicaciones grandes puede que se creen muchos usuario a toda hora y algunos simplemente crean un usuario para probar la aplicación web y otros ya no vuelven a usar la aplicación por lo tanto es buena idea borrar usuario con más de años sin acceso.

Ejemplo de una tabla de roles con campos crud (permiso de **c**rear, permiso de lista**r**, permiso de modificar y permiso de eliminar)

idRol	nombreRol	productoRol	ventaRol	auditoriaRol	usuarioRol	rolRol

1	Admin	crud	crud	crud	crud	crud
2	Usuario	ru	crud	r		

En otras aplicaciones podemos tener más roles ej. El admin, coordinador, supervisor, vendedor, usuario, gerente, profesor, estudiante, agricultor, ingeniero, jefe, etc…

Una vez definido que roles que va a tener nuestra aplicación y que permisos se le van a dar en cada formulario necesitamos tener creada en nuestra base de datos una tabla llamada usuarios -ojo el campo claveUsuario será de VARBINARY(32) ya que vamos a usar sha256, recuerde que el 32 sale de dividir 256 entre 8- con su respectivo MVC para gestionar usuarios ej. formularioUsuarios.php y también el MVC para el login.php que bien o puede ser una pagina completa para solo login o puede ser un <section> en algún lado del index que llame a login.php veamos un ej.

Lo primero es crear dos usuario usando el formularioUsuario.php para probar el login, uno con rol de admin que será 1 y otro con rol de usuario que será 2.

Se requiere también crear en la vista el formularioLogin.php, en el controlador el controladorLogin.php y en el modelo el Login.php.

formularioLogin.php o index.php

```php
<form action="../controlador/controladorLogin.php" method="post">
    <h2>Ingrese al sistema</h2>
    <input name="fEmail" type="email" maxlength="60" placeholder="nombre@sucorreo.co" required autofocus>
    <input name="fClave" type="password" placeholder="Password" required>
    <button name="fEnviar" type="submit" value="Ingresar">Ingresar</button>
</form>
<?php
    @$mensaje = $_GET['mensaje'];
    if (isset($mensaje)){
        if($mensaje=='incorrecto'){
            echo '<div class="alert alert-danger" role="alert">Usuario o clave incorrecto</div>';
        }
    }
?>
```

controladorLogin.php

```php
$emailUsuario = $_POST["fEmail"];
$claveUsuario = $_POST["fClave"];

include_once("../modelo/Conexion.php");
$objetoConexion = new Conexion();
$conexion = $objetoConexion->conectar();

$emailUsuario = mysqli_real_escape_string($conexion, $emailUsuario);

include_once("../modelo/Login.php");
$objetoLogin = new Login($conexion, $emailUsuario, $claveUsuario);
$usuarioEsValido = $objetoLogin->verificarUsuario();
```

```php
    $objetoConexion->desconectar();
    if($usuarioEsValido==true){
        session_start();
        $_SESSION['id']     = $objetoLogin->getIdUsuario();
        $_SESSION['nombre'] = $objetoLogin->getIdNombreUsuario();
        $_SESSION['rol']    = $objetoLogin->getRolUsuario();
        header("location:../vista/formularioVentas.php");
    }else{
        header("location:../vista/formularioLogin.php?mensaje=incorrecto");
    }
```

Login.php

```php
<?php
class Login{
    private $_conexion;
    private $_idUsuario;
    private $_emailUsuario;
    private $_hashedClaveUsuario;
    private $_nombreUsuario;
    private $_rolUsuario;

    function __construct($conexion, $correo, $clave){
        $this->_conexion           = $conexion;
        $this->_emailUsuario       = $correo;
        $this->_hashedClaveUsuario = hash('sha256', $clave);
    }

    function verificarUsuario(){
        $verificacion = mysqli_query($this->_conexion,"SELECT idUsuario, nombreUsuario,
        rolUsuario FROM Usuario WHERE correoUsuario LIKE '$this->_emailUsuario' AND
        CONVERT(claveUsuario, CHAR(100)) LIKE '$this->_hashedClaveUsuario'");

        if(mysqli_num_rows($verificacion)){
            $unUsuario = mysqli_fetch_array($verificacion);
            $this->_idUsuario     = $unUsuario["idUsuario"];
            $this->_nombreUsuario = $unUsuario["nombreUsuario"];
            $this->_rolUsuario    = $unUsuario["rolUsuario"];
            return true;
        }
        return false;
    }

    function getId(){
        return $this->_idUsuario;
    }

    function getNombre(){
        return $this->_nombreUsuario;
    }

```

```
37      function getRol(){
38          return $this->_rolUsuario;
39      }
40  }
41  ?>
```

Cuando el usuario logre ingresar (por medio de un login) a un formulario cualquiera una de las primeras sentencias en el código php debería ser session_start(), lo que permite usar la variable $_SESSION

Cuando el usuario de clic en cerrar sesión en la aplicación se debe usar session_unset() y session_destroy();

Suba todos los archivos generados al link designado por el instructor en la plataforma LMS.

Para desarrollar la anteriores actividades se requiere de un ambiente tipo aula con mesas y equipos de computo con conexión a internet, UPS, sillas ergonómicas, televisor con entrada HDMI o MHL, tablero acrílico, extintor blanco, escoba, recogedor, papeleras de reciclaje.

Para desarrollar la anterior actividad se requiere de los siguientes materiales: Tener en el computador instalado cualquier sistema operativo con interfaz gráfica (Microsoft windows, linux, macos), navegadores web: firefox, google chrome, opera; IDEs como visual studio code, brackets.io, notepad++, netbeans y servidores como xampp o easyphp, suite ofimatica libreoffice o microsoft office.

4. ACTIVIDADES DE EVALUACIÓN

4.1 Genere el login para su sistema de información (debe crear la tabla usuarios si no lo ha hecho) la evidencia para esta actividad deberá subir el login.php, la base de datos exportada.sql, el controladorLogin.php y el formularioLogin.php ó index.php (si tiene el login incrustado en el index.php –recuerde que puede fabricar un index usando launchaco.com-)
4.2 Ajuste el modelo usuarios.php para que puede insertar y modificar claves con hash.
4.3 Ajuste los formularios para que los usuarios no puedan acceder saltándose el login para ello use en las primeras líneas de su formulario:

```
1  <?php
2  ··session_start();
3  ··if (isset($_SESSION['id'])){
4  ?>
```

y en las ultimas líneas (donde cierra el html):

```
126     </body>
127   </html>
128   <?php
129     }else{
130       header("location:../index.php");
131     }
```

4.3 Defina roles para su sistema de información modificando el menú para ocultar/mostrar las opciones según el rol y ocultar/mostrar aquellos formularios al rol correspondiente usando las variables de session, para ello puede agregar la siguiente función en los modelos usando php:

```php
function getPermiso($idUsuario){
    $permisos=mysqli_query($this->conexion,"SELECT ".static::class."rol AS elPermiso FROM roles
    WHERE idRol IN(SELECT idRolUsuario FROM Usuario WHERE idUsuario = $idUsuario");
    $unRegistro=$cantidadBloques->mysqli_fetch_array($permisos);
    return $unRegistro["elPermiso"];
}
```

Y

Evidencias de Aprendizaje	Criterios de Evaluación	Técnicas e Instrumentos de Evaluación
Evidencias de Conocimiento : No Evidencias de Desempeño: No. **Evidencias de Producto**: - Login del proyecto - Manejo de roles usando sesión en el proyecto	- Controla la seguridad del diseño del sistema de información, aplicando las políticas y protocolos establecidos, según normas y procedimientos de la organización.	Técnica de Evaluación: Formulación de Preguntas Instrumento de Evaluación: Cuestionario Técnica de Evaluación: Observación Sistemática Instrumento de Evaluación: Lista de Chequeo **Técnica de Evaluación**: Valoración de producto **Instrumento de Evaluación**: Lista de Verificación

5. GLOSARIO DE TERMINOS

Cifrado: También llamado encripción. Es un proceso para convertir la información a un formato más seguro. En otras palabras, los datos que están en un formato claro, o sea entendible, se convierten mediante un proceso matemático a un formato encriptado o codificado, o sea ininteligible. Una vez que llegan a su destino, se decodifican para poder ser legibles de nuevo, se descifran.

Hashes: Los hash o funciones de resumen son algoritmos que consiguen crear a partir de una entrada (ya sea un texto, una contraseña o un archivo, por ejemplo) una salida

alfanumérica de longitud normalmente fija que representa un resumen de toda la información que se le ha dado (es decir, a partir de los datos de la entrada crea una cadena que sólo puede volverse a crear con esos mismos datos).
Token: También llamado componente léxico, es una cadena de caracteres que tiene un significado coherente en cierto lenguaje de programación. Ejemplos de tokens podrían ser palabras clave (if, else, while, int, ...), identificadores, números, signos, o un operador de varios caracteres, (por ejemplo, :=).
Radius: Acrónimo en inglés de Remote Authentication Dial-In User Server, es un protocolo de autenticación y autorización para aplicaciones de acceso a la red o movilidad IP. Utiliza el puerto 1812 UDP para establecer sus conexiones..

6. REFERENTES BIBLIOGRAFICOS

Foundations of Security: What Every Programmer Needs to Know. Christoph Kern, Anita Kesavan, Neil Daswani. Apress. 2007
Essential PHP Security. A guide to building secure web applications. Chris Shiflett. O'Reilly. 2005
Beginning ASP.NET Security. Barry Dorrans. Wrox. 2010
https://www.owasp.org/index.php/Session_Management_Cheat_Sheet

7. CONTROL DEL DOCUMENTO

	Nombre	Cargo	Dependencia	Fecha
Autor (es)	Julio Cesar Argaez Lenis	Instructor	Coordinación Académica	15 de febrero 2025

8. CONTROL DE CAMBIOS

	Nombre	Cargo	Dependencia	Fecha	Razón del Cambio
Autor (es)					