Universidad Mariano Gálvez de Guatemala

TAREA ENTREGA 3: ADMINISTRACIÓN DE USUARIOS Y ROLES

SEGURIDAD Y AUDITORÍA DE SISTEMAS

Miguel Amilcar Mejía Tó
2290 21 2294
Johny David Chiroy Guarcax
2290 21 12939

Índice

Introducción

En el desarrollo de sistemas de información modernos, la administración de usuarios y roles se constituye como uno de los pilares fundamentales de la seguridad y auditoría. La correcta gestión de identidades digitales no solo permite controlar el acceso a los recursos de un sistema, sino que también garantiza la trazabilidad de las acciones realizadas, contribuyendo a la integridad y confiabilidad de la información.

El presente trabajo tiene como propósito analizar e implementar un módulo de administración de usuarios y roles en un proyecto desarrollado con C# ASP.NET Core 8 MVC, haciendo uso de un stack tecnológico que incluye Entity Framework Core, SQL Server, Bootstrap y JavaScript. El énfasis recae en cuatro aspectos principales:

- La creación de un formulario independiente para el registro y gestión de usuarios.
- La implementación de roles diferenciados (usuario, administrador y superadministrador).
- La visualización de usuarios en tablas con contraseñas cifradas (no visibles).
- La validación de una regla de seguridad consistente en la vigencia de 10 días para cambios de credenciales o roles.

Este enfoque busca integrar tanto fundamentos teóricos de seguridad informática como su aplicación práctica en el contexto de un proyecto real, orientado a las exigencias de un curso de Seguridad y Auditoría de Sistemas.

Administración de identidades

La administración de usuarios constituye el conjunto de políticas, procedimientos y tecnologías que permiten gestionar el ciclo de vida de las cuentas en un sistema informático: creación, modificación, suspensión, eliminación y auditoría. En ASP.NET Core MVC, este proceso se encuentra soportado por el componente Identity, el cual ofrece mecanismos nativos para autenticación, autorización y gestión de roles.

Principios de seguridad aplicados

- Para garantizar la robustez del módulo de usuarios, se aplican los siguientes principios:
- Mínimos privilegios: cada usuario solo debe poseer los permisos necesarios para cumplir su función.
- Separación de funciones: ningún usuario debe concentrar el control total de procesos críticos.
- No repudio: todas las acciones deben quedar registradas de manera auditable.
- Defensa en profundidad: se deben implementar controles de seguridad en todas las capas del sistema (base de datos, backend, frontend).

Creación de formularios independientes

En el sistema desarrollado con ASP.NET Core MVC 8, se opta por implementar formularios exclusivos para la gestión de usuarios. Estos permiten:

- Registrar nuevos usuarios con validaciones de unicidad (correo, CUI).
- Editar información personal y de contacto.
- Asignar roles y estados de cuenta.
- Gestionar suspensiones y eliminaciones lógicas.

El formulario está diseñado en Razor Views, empleando Bootstrap para mejorar la experiencia visual y validaciones en JavaScript para reforzar la usabilidad.

Implementación de roles en el sistema

El sistema contempla tres roles principales:

- Usuario: acceso restringido a las operaciones básicas asignadas a su perfil.
- Administrador: capacidad de gestionar catálogos, reportes y usuarios estándar.
- Superadministrador: privilegios globales, incluida la modificación de políticas de seguridad y la auditoría de registros.

En el código, la autorización se implementa mediante anotaciones:

```
[Authorize(Roles = "Administrador,SuperAdmin")]

public IActionResult GestionUsuarios()

{

    return View();

}
```

Cifrado de contraseñas

Una de las principales vulnerabilidades en sistemas mal diseñados es el almacenamiento de contraseñas en texto plano. En este proyecto, se adoptó el uso de PBKDF2 con salt aleatorio, configurado con un número elevado de iteraciones para incrementar la resistencia ante ataques de fuerza bruta.

El proceso aplicado es:

- Generar un salt único para cada usuario.
- Aplicar PBKDF2 sobre la contraseña ingresada.
- Almacenar en SQL Server únicamente el hash, el salt y las iteraciones.

Esto asegura que la contraseña original nunca se exponga ni siquiera a administradores de base de datos.

Visualización en tablas con información protegida

La vista Usuarios/Index.cshtml presenta los usuarios registrados, mostrando únicamente información no sensible:

- Nombre completo.
- Correo electrónico.
- Rol asignado.
- Estado de la cuenta.
- Fechas de creación y última modificación.

En lugar de exponer la contraseña, se presentan indicadores de seguridad como:

- “Último cambio de contraseña: dd/MM/yyyy”.
- “Autenticación multifactor: activada/desactivada”.

Validación de reglas: vigencia de 10 días

Se implementa una regla de seguridad que establece que toda modificación de credenciales o roles queda bajo un periodo de observación de 10 días.

- Durante dicho plazo, el sistema marca al usuario como “En revisión”.
- Si al cumplirse el plazo no se confirma el cambio por parte de un administrador, se aplica una acción correctiva, como la reversión del rol anterior o la solicitud de restablecimiento de contraseña.
- Este comportamiento se registra en la tabla AUDITORIA_USUARIO, que almacena los cambios efectuados, la fecha de modificación y el responsable.

```
if (usuario.FechaModificacion.AddDays(10) <= DateTime.Now && !usuario.RevisionConfirmada)

{

   usuario.Estado = "Revisión Pendiente";
```

```
    _context.Update(usuario);
    await _context.SaveChangesAsync();
}
```

La implementación del módulo de administración de usuarios bajo ASP.NET Core 8 MVC demuestra la capacidad del framework para integrar principios de seguridad informática en el desarrollo de aplicaciones web.

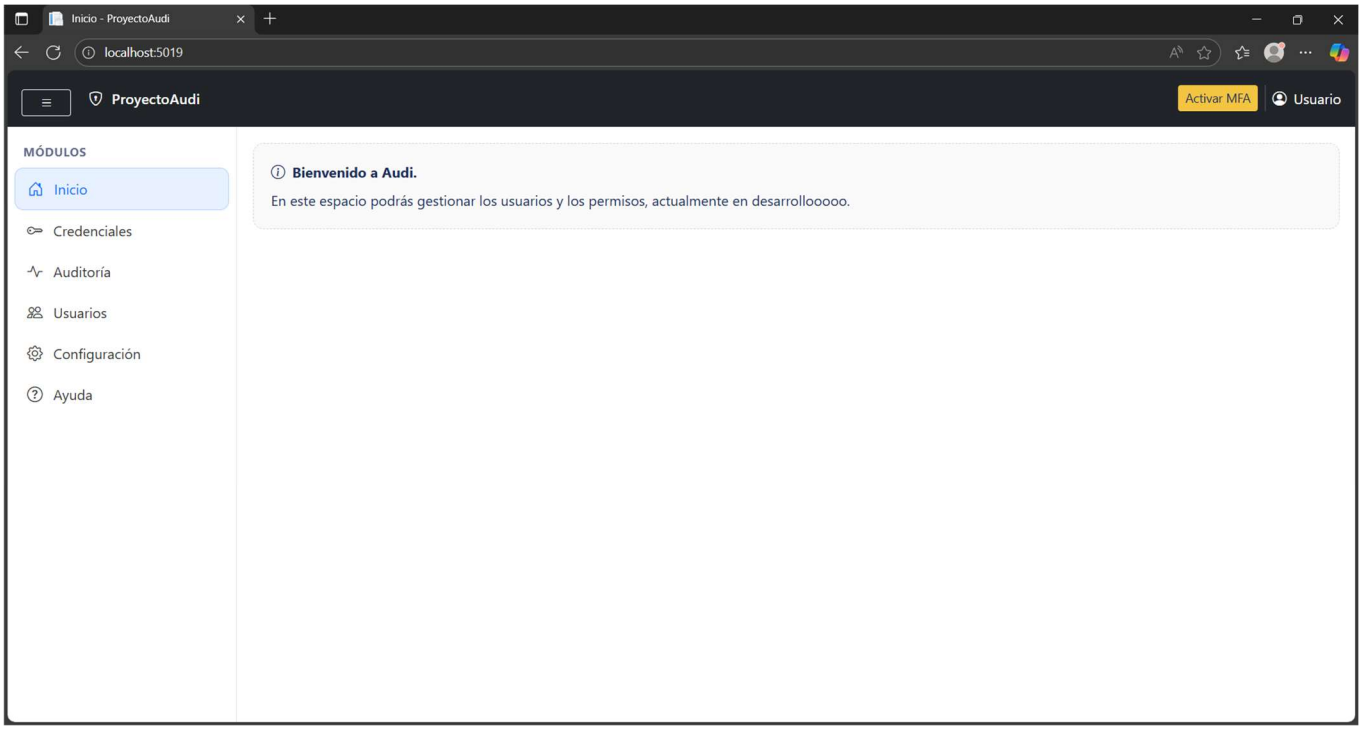
- La creación de formularios independientes refuerza la separación de funciones y mejora la trazabilidad.
- El modelo RBAC implementado garantiza escalabilidad y facilidad de mantenimiento.
- El uso de algoritmos de hash como PBKDF2 incrementa significativamente la seguridad de las credenciales en comparación con funciones simples como SHA-256.
- La regla de vigencia de 10 días introduce un mecanismo de control preventivo frente a cambios indebidos en credenciales o roles, alineado con buenas prácticas de auditoría.

No obstante, se identifican desafíos a considerar:

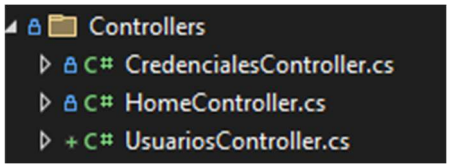
- La necesidad de capacitación de los administradores para gestionar adecuadamente el módulo.
- La implementación de autenticación multifactor como capa adicional de seguridad.
- La integración de alertas en tiempo real cuando se realizan cambios críticos.

Avances de la asignación

Creación de vista HOME.

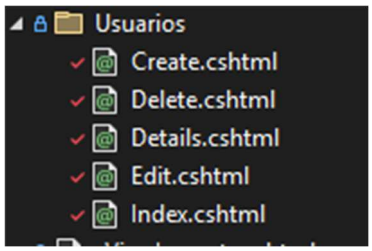


Creación de UsuarioControler.



Creación de UsuarioViewModel.

Creación de vistas Index, Create, Details, Edit y Delete



Conclusiones

La investigación demuestra que la administración de usuarios y roles en un sistema web desarrollado con ASP.NET Core 8 MVC no solo constituye una necesidad técnica, sino también un requisito estratégico para garantizar la seguridad organizacional.

El desarrollo de formularios independientes, la implementación del modelo RBAC, la protección criptográfica de contraseñas y la validación de reglas de vigencia de 10 días representan prácticas alineadas con estándares internacionales como ISO/IEC 27001. Estas medidas permiten fortalecer la gobernanza de accesos, reducir riesgos de abuso de privilegios y aumentar la confianza en la plataforma desarrollada.

Se concluye que la seguridad de un sistema no depende únicamente de la tecnología utilizada, sino de la correcta integración entre procesos, controles técnicos y auditoría continua.