**PROYECTO**

**Sistema de Autenticación Centralizado (SAC)**

**Introducción**

El control de acceso es un mecanismo fundamental en los sistemas de información que limita el acceso a datos y funcionalidades según el perfil del usuario. A través de la definición de roles y permisos, se garantiza la seguridad de la información y la eficiencia de los procesos.

Sin embargo, en muchos sistemas actuales se ha perdido la comprensión de cómo funciona este mecanismo a nivel técnico. La proliferación de herramientas y frameworks que automatizan gran parte del proceso ha llevado a una desconexión entre los desarrolladores y los principios básicos del control de acceso.

**Objetivo**

El proyecto SAC tiene como objetivo recuperar y sistematizar el conocimiento sobre el control de acceso, proporcionando una plataforma centralizada que permita:

* **Definir roles y permisos:** crear una estructura jerárquica de roles y asignar permisos específicos a cada uno.
* **Gestionar usuarios:** crear, modificar y eliminar cuentas de usuario, asociándolas a los roles correspondientes.
* **Integrar con otros sistemas:** proporcionar interfaces de programación (APIs) para que otros sistemas puedan autenticar usuarios y autorizar sus acciones.
* **Documentar el proceso:** crear una documentación detallada que explique el funcionamiento del sistema y los conceptos clave del control de acceso.

**Beneficios esperados**

* **Seguridad**: reducción del riesgo de accesos no autorizados.
* **Eficiencia**: simplificación de los procesos de gestión de usuarios y permisos.
* **Flexibilidad**: adaptabilidad a diferentes necesidades y escenarios.
* **Cumplimiento**: facilidad para cumplir con las normativas de seguridad de la información.

**Componentes clave**

[*agregar aquí un diagrama que muestre la relación entre usuarios, roles, permisos y sistemas*]

* **Repositorio de usuarios**: almacena la información de los usuarios, incluyendo credenciales y asociaciones a roles.
* **Motor de autorización**: evalúa las solicitudes de acceso y determina si el usuario tiene los permisos necesarios.
* **API de autenticación**: proporciona servicios para autenticar usuarios y obtener tokens de acceso.

**Metodología**

[*agregar aquí la descripción sobre la metodología a seguir en el proyecto*]

|  |
| --- |
| **Considerando la tecnología:** MySQL como base de datos y PHP para la lógica de negocio, se propone la siguiente metodología, combinando elementos de la metodología Ágil y Waterfall:  **Fase 1: Planificación y Diseño**   * **Análisis de requerimientos:**   + Definición detallada de los roles y permisos necesarios.   + Diseño de la estructura de la base de datos MySQL (tablas, relaciones, índices).   + Diseño de la arquitectura del sistema, incluyendo componentes y flujo de datos.   + Identificación de las interfaces de usuario (API REST). * **Diseño técnico:**   + Selección de frameworks PHP (Laravel, Symfony, etc.) para agilizar el desarrollo.   + Diseño de las clases y funciones PHP para la gestión de usuarios, roles, permisos y autenticación.   + Definición de las reglas de negocio para la autorización. * **Planificación de tareas:**   + Creación de un diagrama de Gantt para visualizar el cronograma del proyecto.   + Estimación del tiempo requerido para cada tarea.   **Fase 2: Desarrollo**   * **Desarrollo de la base de datos:**   + Creación de las tablas en MySQL según el diseño.   + Carga de datos iniciales (si aplica). * **Desarrollo del backend:**   + Implementación de las funcionalidades en PHP:     - Registro de usuarios.     - Inicio de sesión.     - Gestión de roles y permisos.     - Motor de autorización (evaluación de permisos).     - API REST para la interacción con otros sistemas. * **Desarrollo del frontend (opcional):**   + Si se requiere una interfaz de usuario, se desarrollará utilizando tecnologías como HTML, CSS y JavaScript.   **Fase 3: Pruebas**   * **Pruebas unitarias:**   + Verificación del correcto funcionamiento de cada módulo de forma aislada. * **Pruebas de integración:**   + Validación de la interacción entre los diferentes componentes del sistema.   + Pruebas de las API REST. * **Pruebas de seguridad:**   + Evaluación de la vulnerabilidad del sistema a ataques comunes (inyección SQL, XSS, etc.). * **Pruebas de rendimiento:**   + Medición de la capacidad del sistema para manejar un gran número de usuarios y solicitudes.   **Fase 4: Implementación y Mantenimiento**   * **Deployment:**   + Instalación del sistema en el entorno de producción.   + Configuración de servidores web y base de datos. * **Capacitación:**   + Capacitación de los usuarios finales en el uso del sistema. * **Mantenimiento:**   + Corrección de errores y mejoras continuas.   + Monitoreo del sistema para detectar problemas y optimizar el rendimiento.   **Metodología Ágil**   * **Iteraciones cortas:** Dividir el proyecto en iteraciones cortas (sprints) para obtener feedback temprano y realizar ajustes. * **Desarrollo incremental:** Entregar funcionalidades de forma incremental para obtener valor de negocio más rápido. * **Colaboración:** Fomentar la colaboración entre los miembros del equipo a través de reuniones diarias y revisiones de código.   **Consideraciones Adicionales**   * **Seguridad:** Implementar medidas de seguridad robustas para proteger la información de los usuarios (encriptación, gestión de contraseñas seguras, etc.). * **Escalabilidad:** Diseñar el sistema para que pueda escalar horizontalmente en caso de crecimiento. * **Documentación:** Crear una documentación completa del sistema, incluyendo la arquitectura, el código y las guías de usuario. * **Versionamiento:** Utilizar un sistema de control de versiones (Git) para gestionar los cambios en el código.   **Herramientas Recomendadas:**   * **Base de datos:** MySQL Workbench para diseño y administración. * **Desarrollo PHP:** PhpStorm, Visual Studio Code. * **Control de versiones:** Git. * **Gestión de proyectos:** Trello, Jira.   **Adaptación de la Metodología:**  Esta propuesta es un punto de partida y puede adaptarse según las necesidades específicas del proyecto. Por ejemplo, si el proyecto es muy grande, se puede considerar dividirlo en subproyectos más pequeños.  **¿Qué te parece esta propuesta? ¿Hay algún aspecto que te gustaría profundizar o alguna otra metodología que te interese?**  **Adicionalmente, podríamos explorar temas como:**   * **Implementación de autenticación de dos factores.** * **Integración con servicios de directorio activo.** * **Gestión de sesiones y tokens de acceso.** * **Mejores prácticas para la seguridad en aplicaciones web.** |

**Desarrollo**

[*agregar aquí el detalle para el desarrollo del proyecto*]

**Conclusiones**

[*agregar aquí las conclusiones sobre el proyecto o los resultados finales*]