SISTEMA DE CONTROL DE ACCESO PARA EL COMPLEJO EMPRESARIAL ZONA ACME

ALVARO ANDRES MARTINEZ ALCINA

JUAN DAVID CONDE MARTINEZ

MICHEL ADRIAN TORRADO ROA

SKILLS

TRAINER

CARLOS RUEDA

Campulands

Floridablanca

2024

TABLA DE CONTENIDO

INTRODUCCIÓN

Actualmente, la seguridad y el control de acceso son fundamentales para el funcionamiento de los complejos empresariales, por lo que, se hace indispensable contar con un sistema que optimice estos procesos. El *"*Sistema de Control de Acceso para el Complejo Empresarial Zona ACME" busca garantizar un acceso seguro, organizado y eficiente para todos los usuarios del complejo.

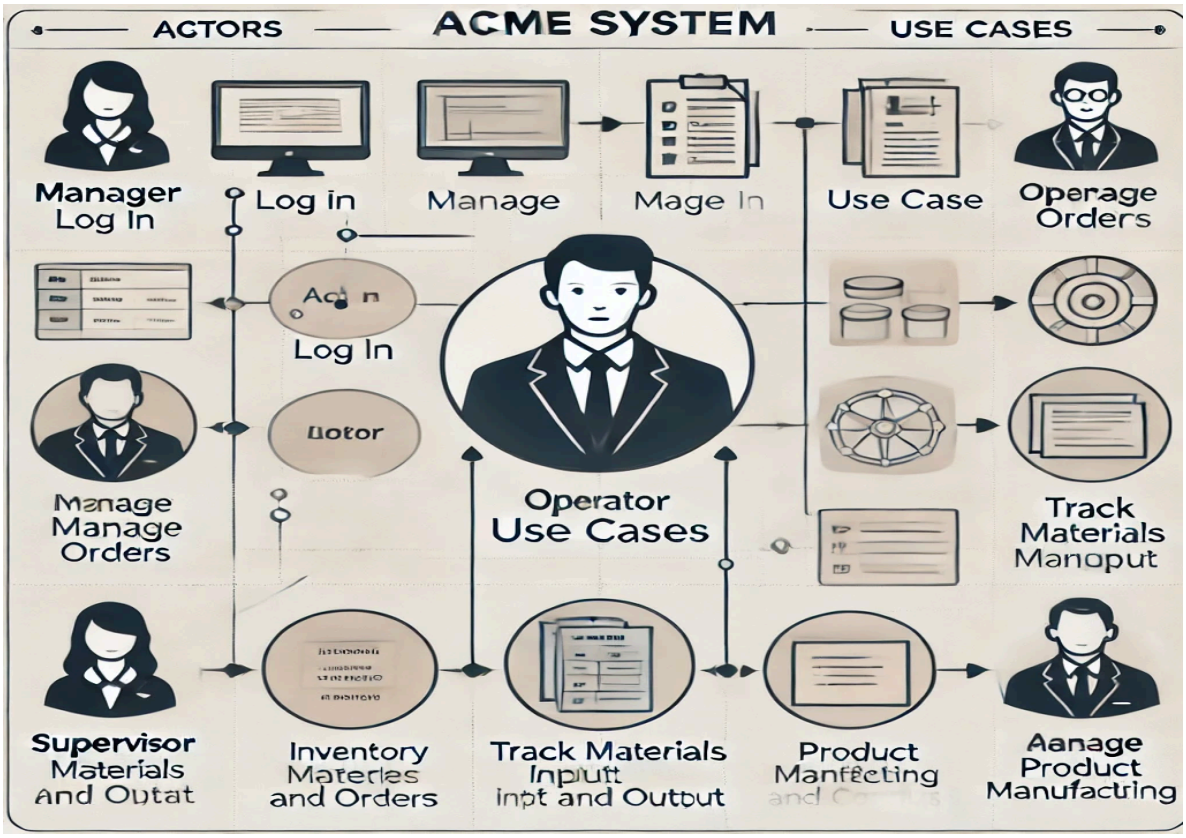
Este proyecto tiene como objetivo principal el desarrollo de una plataforma que registre y supervise las actividades de entrada y salida de los empleados, vehículos e invitados al complejo, asegurando la trazabilidad de los eventos y proporcionando funcionalidades avanzadas de gestión para los diferentes roles como: superusuarios, supervisores de seguridad, guardas, y gerentes de empresas.

El sistema usa una arquitectura basada en los principios de diseño MVC, utilizando MySQL como gestor de base de datos y Java como lenguaje de programación principal. Asimismo, se implementan patrones de diseño, técnicas de programación funcional y una interfaz gráfica intuitiva, cumpliendo con los estándares modernos de escalabilidad, rendimiento y seguridad.

DESCRIPCIÓN

El sistema contempla la interacción de varios perfiles de usuario, diseñados para cumplir roles específicos:

- Superusuario: Responsable de la configuración inicial del sistema, incluyendo la creación de supervisores de seguridad y la gestión de roles.
- Supervisores de seguridad: Administran a los guardas, manejan incidencias, y supervisan los accesos en tiempo real.
- Guardas de seguridad: Ejecutan el registro de entradas y salidas de personas y vehículos, consultan información relevante sobre los usuarios e implementan decisiones operativas basadas en anotaciones.
- Funcionarios de empresas(Gerentes): Representan a cada empresa en el complejo, siendo responsables de registrar a sus trabajadores e invitados, y de gestionar incidentes relacionados con su personal.
- Trabajadores e invitados: Usuarios finales que interactúan con el sistema en los puntos de control para registrar su entrada y salida.



REQUERIMIENTOS

- Requerimientos funcionales:
    1. El sistema manejara módulos de superusuario, supervisores de seguridad, guardas de seguridad, funcionarios de empresas.
    2. Debe existir un rol gerente que dará privilegios y manejara todas las funciones comunes que un funcionario de empresa cuenta.
    3. Un superusuario crea a los supervisores de seguridad.
    4. Los supervisores de seguridad crean a los guardas de seguridad y a los funcionarios de empresas.
    5. El rol gerente del funcionario de empresa gestiona a los trabajadores e invitados asociados a la empresa.

6. Los usuarios creados no se pueden eliminar, se debe inactivar, manejando consistencia de datos.
7. En el módulo de acceso, los usuarios deben presentar su documento de identificación.
8. El ingreso de identidad del usuario debe ser manualmente registrado por el guarda de seguridad, si no se encuentra registrado, se deberá notificar el incidente.
9. Las anotaciones deben ser visibles por los guardas de seguridad.
10. Las anotaciones de prohibición deben ser las primeras en mostrar.
11. Al salir, el usuario debe presentar su documento para realizar el registro de salida correctamente.
12. Al no realizarse el registro de salida correctamente, debe registrarse en el sistema con los respectivos datos como fecha y hora, quien autoriza y la anotación.
13. Si se ingresa en un vehículo el usuario debe registrar la matrícula correspondiente al conductor.
14. Cada persona dentro del vehículo debe ser registrada por el guarda de seguridad.
15. Los supervisores deben registrar las anotaciones de comportamientos indebidos.
16. Las anotaciones deben manejar categorías como la prohibición de acceso al complejo empresarial.
17. El levantamiento de la restricción debe ser justificado y registrado por el gerente.
18. El levantamiento de la restricción solo lo podrá realizar el superusuario.
19. Generar reportes detallados sobre los usuarios activos e inactivos(supervisores, guardas de seguridad y funcionarios).
20. Realizar un listado de trabajadores e invitados de cada empresa.

21. Presentar informes de trazabilidad de acceso de trabajadores y funcionarios en rangos de fechas.
22. El supervisor podrá monitorear en tiempo real las actividades en la entrada en una pantalla del sistema.
23. Los funcionarios de empresas podrán visualizar el estado actual de trabajadores e invitados del complejo empresarial, también podrán saber quienes ingresaron al complejo y a que oficinas se dirigen.
24. Los guardas y supervisores de seguridad podrán ver en tiempo real en sus pantallas cuando los gerentes autoricen o cambien de estado algún trabajador o invitado que esté en la entrada.
25. El superusuario podrá configurar en donde se encontrará la base de datos.

- Requerimientos no funcionales:
    1. Un buen manejo de la base de datos relacional con las debidas normalizaciones, debe constatar de un sistema consistente y eficiente.
    2. El sistema de gestión de base de datos que se dispondrá por el complejo empresarial zona ACME es MySQL.
    3. El código debe ser escalable, funcional y comentado, implementando los principios SOLID y patrones de diseño.
    4. Uso correcto y adecuado de la paleta de colores para las vistas.
    5. Manejo de interfaces intuitivas, amigables, ordenadas, y entendibles para cualquier tipo de usuario.
    6. Rendimiento óptimo y eficaz.
    7. Seguridad de autentificación de roles con controles de acceso y restricciones.

8. Disponibilidad y fiabilidad.
9. El proyecto se va a desplegar en un repositorio de GitHub.

MATRIZ DE RESPONSABILIDADES

A continuación, las siguientes tablas que representan la distribución de las tareas que se realizó para el desarrollo de cada requerimientos al mismo tiempo se visualiza los responsables de los ítems.

#	Requerimientos	Tareas	Responsable(s)
1	El sistema manejara módulos de superusuario, supervisores de seguridad, guardas de seguridad, funcionarios de empresas.	❖ Desarrollo de diagrama de clases ❖ Desarrollo del diseño relacional. ❖ Desarrollo del diagrama UML. ❖ Creación de la base de datos. ❖ Implementación de la Arquitectura MVC. ❖ Desarrollar interfaz login ❖ Crear clases y servicios para gestión de usuarios y roles.	Todos
2	Debe existir un rol gerente que dará privilegios y manejara todas las funciones comunes que un funcionario de empresa cuenta.	❖ Añadir el rol Gerente al modelo de datos. ❖ Crear servicios de asignación de privilegios. ❖ Configurar proxy para manejar roles en el sistema.	Alvaro

3	Un superusuario crea a los supervisores de seguridad.	❖ Desarrollar interfaz para superusuario ❖ Implementar módulo registro de supervisores de seguridad ❖ Crear servicios para registro de supervisores.	Alvaro, Conde
4	Los supervisores de seguridad crean a los guardas de seguridad y a los funcionarios de empresas.	❖ Desarrollar interfaz supervisor de seguridad ❖ Implementar módulo registro de guarda seguridad y funcionario de empresas. ❖ Crear servicios para registro de guarda de seguridad y funcionario de empresa.	Conde, Michel
5	El rol gerente del funcionario de empresa gestiona a los trabajadores e invitados asociados a la empresa.	❖ Desarrollar interfaz funcionario de empresa. ❖ Implementar módulo registro de trabajador e invitados. ❖ Crear servicios para registro de trabajador e invitado.	Conde, Michel

6	Los usuarios creados no se pueden eliminar, se debe inactivar, manejando consistencia de datos.	❖ Implementar funcionalidad de desactivación de estado del usuario ❖ Implementar Log de registro de cambio de estado	Alvaro
7	En el módulo de acceso, los usuarios deben presentar su documento de identificación.	❖ Implementar interfaz de registro de ingreso en el módulo guardas de seguridad. ❖ Crear servicios para registro de de acceso para el usuario.	Conde, Michel
8	El ingreso de identidad del usuario debe ser manualmente registrado por el guarda de seguridad, si no se encuentra registrado, se deberá notificar el incidente.	❖ Integrar funcionalidad de notificación al rol gerente de la empresa que el visitante desea ingresar. ❖ Registrar el incidente durante el proceso de notificación.. ❖ Implementar sistema de notificaciones al módulo funcionario de empresa.	Alvaro
9	Las anotaciones deben ser visibles por los guardas de seguridad.	❖ Integrar módulo vista anotación usuario.	Conde, Michel

		❖ Integrar funcionalidad para listar las anotaciones.	
10	Las anotaciones de prohibición deben ser las primeras en mostrar.	❖ Implementar funcionalidad de ordenamiento de las anotaciones según la clasificación tipo prohibición.	Michel
11	Al salir, el usuario debe presentar su documento para realizar el registro de salida correctamente.	❖ Diseñar un flujo de validación para salida. ❖ Actualizar registros en la base con timestamp.	Alvaro
12	Al no realizarse el registro de salida correctamente, debe registrarse en el sistema con los respectivos datos como fecha y hora, quien autoriza y la anotación.	❖ Implementar lógica para capturar datos adicionales.	Alvaro
13	Si se ingresa en un vehículo el usuario debe registrar la matrícula correspondiente al conductor.	❖ Desarrollar interfaz módulo en guarda de seguridad. ❖ Integrar registro de ingreso del vehículo según la matrícula. ❖ Implementar validación en la BD la existencia del vehículo	Conde, Michel

14	Cada persona dentro del vehículo debe ser registrada por el guarda de seguridad.	❖ Diseñar sección para ingreso de datos múltiples.	Conde
15	Los supervisores deben registrar las anotaciones de comportamientos indebidos.	❖ Integrar sección registro de anotaciones por comportamientos indebidos. ❖ Clasificar anotaciones según los comportamientos indebidos.	Conde, Michel
16	Las anotaciones deben manejar categorías como la prohibición de acceso al complejo empresarial.	❖ Diseñar base de datos para categorías de anotaciones. ❖ Crear lógica para filtrado y búsqueda por categorías.	Alvaro
17	El levantamiento de la restricción debe ser justificado y registrado por el gerente.	❖ Integrar sección Justificación de levantamiento de restricciones.	Conde, Michel
18	El levantamiento de la restricción solo lo podrá realizar el superusuario.	❖ Implementación módulo restricciones en superusuario. ❖ Establecer opción de cambiar estado de la restricción.	Conde
19	Generar reportes detallados sobre los usuarios activos e inactivos(supervisores, guardas de seguridad y funcionarios).	❖ Implementar servicios de generación de reportes.	Michel

20	Realizar un listado de trabajadores e invitados de cada empresa.	❖ Implementar listados de trabajadores e invitados de cada empresa de manera diario.	Alvaro
21	Presentar informes de trazabilidad de acceso de trabajadores y funcionarios en rangos de fechas.	❖ Implementar filtros por fecha en las consultas.	Michel
22	El supervisor podrá monitorear en tiempo real las actividades en la entrada en una pantalla del sistema.	❖ Integrar sección en tiempo real sobre las actividades en la entrada en el módulo supervisor.	Conde
23	Los funcionarios de empresas podrán visualizar el estado actual de trabajadores e invitados del complejo empresarial, también podrán saber quienes ingresaron al complejo y a que oficinas se dirigen.	❖ Implementar sección visualización del estado actual de los trabajadores e invitados.	Conde
24	Los guardas y supervisores de seguridad podrán ver en tiempo real en sus pantallas cuando los gerentes autoricen o cambien de estado algún trabajador o invitado que esté en la entrada.	❖ Implementar sección visualización del estado actual de los usuarios.	Conde

25	El superusuario podrá configurar en donde se encontrará la base de datos.	❖ Integrar interfaz configuración de IP de la BD en el módulo superusuario. ❖ Implementar funcionalidad de intercambio de IP para un mejor uso de la BD en otra red	Conde, Alvaro

Todos			
#	**Requerimiento**	**Tareas**	**Nivel**
1	El sistema manejara módulos de superusuario, supervisores de seguridad, guardas de seguridad, funcionarios de empresas.	Desarrollo de diagrama de clases	Alto
		Desarrollo del diseño relacional.	
		Desarrollo del diagrama UML.	
		Creación de la base de datos.	
		Implementación de la Arquitectura MVC.	
		Crear clases y servicios para gestión de usuarios y roles.	

Conde			
#	**Requerimiento**	**Tareas**	**Nivel**
1	El sistema manejara módulos de superusuario, supervisores de seguridad, guardas de seguridad, funcionarios de empresas.	Desarrollar interfaz login	Alto
3	Un superusuario crea a los supervisores de seguridad.	Desarrollar interfaz para superusuario	Alto
		Implementar módulo registro de supervisores de seguridad	
4	Los supervisores de seguridad crean a los guardas de seguridad y a los funcionarios de empresas.	Desarrollar interfaz supervisor de seguridad	Alto
		Implementar módulo registro de guarda seguridad y funcionario de empresas.	
5	El rol gerente del funcionario de empresa gestiona a los trabajadores e invitados asociados a la empresa.	Desarrollar interfaz funcionario de empresa.	Alto
		Implementar módulo registro de trabajador e invitados.	

7	En el módulo de acceso, los usuarios deben presentar su documento de identificación.	Implementar interfaz de registro de ingreso en el módulo guardas de seguridad.	Alto
9	Las anotaciones deben ser visibles por los guardas de seguridad.	Integrar módulo vista anotación usuario.	Alto
13	Si se ingresa en un vehículo el usuario debe registrar la matrícula correspondiente al conductor.	Desarrollar interfaz módulo en guarda de seguridad. Integrar registro de ingreso del vehículo según la matrícula.	Medio
14	Cada persona dentro del vehículo debe ser registrada por el guarda de seguridad.	Diseñar sección para ingreso de datos múltiples.	Medio
15	Los supervisores deben registrar las anotaciones de comportamientos indebidos.	Integrar sección registro de anotaciones por comportamientos indebidos.	Medio
17	El levantamiento de la restricción debe ser justificado y registrado por el gerente.	Integrar sección Justificación de levantamiento de restricciones.	Bajo

18	El levantamiento de la restricción solo lo podrá realizar el superusuario.	Implementación módulo restricciones en superusuario.	Bajo
		Establecer opción de cambiar estado de la restricción.	
22	El supervisor podrá monitorear en tiempo real las actividades en la entrada en una pantalla del sistema.	Integrar sección en tiempo real sobre las actividades en la entrada en el módulo supervisor.	Bajo
23	Los funcionarios de empresas podrán visualizar el estado actual de trabajadores e invitados del complejo empresarial, también podrán saber quienes ingresaron al complejo y a que oficinas se dirigen.	Implementar sección visualización del estado actual de los trabajadores e invitados.	Bajo
24	Los guardas y supervisores de seguridad podrán ver en tiempo real en sus pantallas cuando los gerentes autoricen o cambien de estado algún	Implementar sección visualización del estado actual de los usuarios.	Bajo

	trabajador o invitado que esté en la entrada.		
25	El superusuario podrá configurar en donde se encontrará la base de datos.	Integrar interfaz configuración de IP de la BD en el módulo superusuario.	Bajo

Michel			
#	**Requerimiento**	**Tareas**	**Nivel**
4	Los supervisores de seguridad crean a los guardas de seguridad y a los funcionarios de empresas.	Crear servicios para registro de guarda de seguridad y funcionario de empresa.	Alto
5	El rol gerente del funcionario de empresa gestiona a los trabajadores e invitados asociados a la empresa.	Crear servicios para registro de trabajador e invitado.	Alto
7	En el módulo de acceso, los usuarios deben presentar su documento de identificación.	Crear servicios para registro de de acceso para el usuario.	Alto

9	Las anotaciones deben ser visibles por los guardas de seguridad.	Integrar funcionalidad para listar las anotaciones.	Alto
10	Las anotaciones de prohibición deben ser las primeras en mostrar.	Implementar funcionalidad de ordenamiento de las anotaciones según la clasificación tipo prohibición.	Alto
13	Si se ingresa en un vehículo el usuario debe registrar la matrícula correspondiente al conductor.	Implementar validación en la BD la existencia del vehículo	Medio
15	Los supervisores deben registrar las anotaciones de comportamientos indebidos.	Clasificar anotaciones según los comportamientos indebidos.	Medio
19	Generar reportes detallados sobre los usuarios activos e inactivos(supervisores, guardas de seguridad y funcionarios).	Implementar servicios de generación de reportes.	Bajo
21	Presentar informes de trazabilidad de acceso de	Implementar filtros por fecha en las consultas.	Bajo

	trabajadores y funcionarios en rangos de fechas.		

Alvaro			
#	**Requerimiento**	**Tareas**	**Nivel**
2	Debe existir un rol gerente que dará privilegios y manejara todas las funciones comunes que un funcionario de empresa cuenta.	Añadir el rol Gerente al modelo de datos.	Alto
		Crear servicios de asignación de privilegios.	
		Configurar proxy para manejar roles en el sistema.	
3	Un superusuario crea a los supervisores de seguridad.	Crear servicios para registro de supervisores.	Alto
6	Los usuarios creados no se pueden eliminar, se debe inactivar, manejando consistencia de datos.	Implementar funcionalidad de desactivación de estado del usuario	Alto
		Implementar Log de registro de cambio de estado	

8	El ingreso de identidad del usuario debe ser manualmente registrado por el guarda de seguridad, si no se encuentra registrado, se deberá notificar el incidente.	Integrar funcionalidad de notificación al rol gerente de la empresa que el visitante desea ingresar.	Alto
		Registrar el incidente durante el proceso de notificación..	
		Implementar sistema de notificaciones al módulo funcionario de empresa.	
11	Al salir, el usuario debe presentar su documento para realizar el registro de salida correctamente.	Diseñar un flujo de validación para salida.	Alto
		Actualizar registros en la base con timestamp.	
12	Al no realizarse el registro de salida correctamente, debe registrarse en el sistema con los respectivos datos como fecha y hora, quien autoriza y la anotación.	Implementar lógica para capturar datos adicionales.	Alto
16	Las anotaciones deben manejar categorías como la	Diseñar base de datos para categorías de anotaciones.	Medio
		Crear lógica para filtrado y búsqueda por	

	prohibición de acceso al complejo empresarial.	categorías.	
20	Realizar un listado de trabajadores e invitados de cada empresa.	Implementar listados de trabajadores e invitados de cada empresa de manera diario.	Bajo
25	El superusuario podrá configurar en donde se encontrará la base de datos.	El superusuario podrá configurar en donde se encontrará la base de datos. ❖ Implementar funcionalidad de intercambio de IP para un mejor uso de la BD en otra red	Bajo

DISEÑO RELACIONAL

Planteamiento del Diseño Relacional

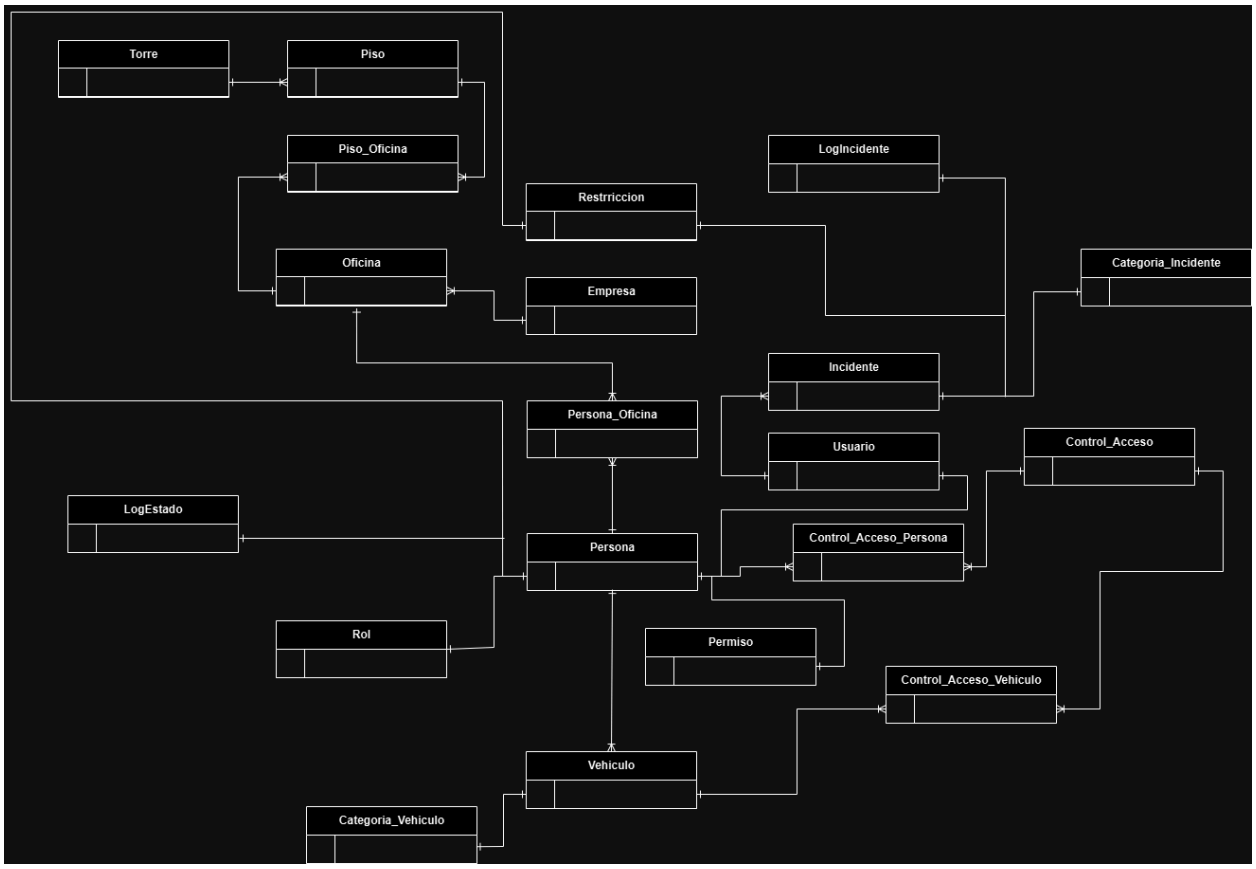
Identificación de Entidades

El primer paso en el diseño relacional fue identificar las entidades relacionales que deben ser representadas en la base de datos. En este caso, las entidades principales fueron:

- **Usuarios:** Representa a las personas que tienen acceso al sistema de control del edificio.
- **Control Accesos:** Registra la entrada y salida de los usuarios al edificio.
- **Vehículos:** Representa los vehículos de los usuarios que pueden ingresar al estacionamiento del edificio.

- **Incidentes:** Registra cualquier incidente o problema relacionado con los usuarios dentro del edificio.
- **Personas:** Representa a las personas que se encuentran registradas en el sistema para entrar al complejo y sus datos básicos.

Primer nivel:



Entidades:

- ***Categoría Vehículo:*** La tabla Categoría Vehículo permite clasificar los diferentes tipos de vehículos que pueden ser registrados en el sistema. Estas categorías podrían incluir "automóvil", "motocicleta", "camión", "bus", entre otros.

- ***Rol:*** La tabla Rol define los dos tipos de persona que están registradas en el sistema que son “Trabajadores” e “Invitados”.
- ***Permiso:*** La tabla permiso guarda los permisos que se le dan a los invitados para entrar al complejo.
- ***Categoría Incidente:*** La tabla Categoría Incidente clasifica los diferentes tipos de incidentes que pueden ocurrir dentro del edificio.
- ***Restricción:*** La tabla restricción se encarga de guardar las prohibiciones entras activas o inactivas que se encuentran.
- ***Empresa:*** En la tabla Empresa se puede encontrar toda la información de las empresas dentro del complejo ACME.
- ***Oficina:*** En la tabla oficina se encontrará el nombre hacia dónde va las personas dentro del complejo
- ***Piso:*** En la tabla Piso se encontrará el número o cantidad de pisos que cuenta el edificio.
- ***Torre:*** En la tabla Torre se encontrará los nombres de los edificios que hay distribuidos dentro del complejo.
- ***Log Estado:*** La tabla de Log-Estado se encargará de guardar los cambios que se realice de una persona que esté en el sistema del estado de activo a inactivo o viceversa.
- ***Log Incidente:*** En la tabla Log-Incidente se guardará la información del proceso que lleve una persona.

Relaciones:

- ***Control Acceso-Vehiculo:*** En la tabla de Control Acceso-Vehículo, se maneja una de tipo intermedia debido a que la relación es de muchos a muchos, por lo consiguiente, se podrán encontrar las llaves foráneas.

- ***Control Acceso-Persona:*** En la tabla de Control Acceso-Persona, se maneja una de tipo intermedia debido a que la relación es de muchos a muchos, por lo consiguiente, se podrán encontrar las llaves foráneas.
- ***Persona-Oficina:*** En la tabla intermedia de Persona-Oficina, se encarga de manejar la información de hacia dónde va una persona que está entrando al complejo, por lo que es una la relación de muchos a muchos, por lo consiguiente, se podrán encontrar las llaves foráneas.
- ***Piso-Oficina:*** En la tabla intermedia de Piso-Oficina, se maneja una relación de muchos a muchos, por lo consiguiente, se podrán encontrar las llaves foráneas que nos indicará en qué piso se encuentra la oficina.

Las relaciones de **UNO A UNO** la manejas las ***CATEGORÍAS, ROL y los LOGS*** ya que solamente se le puede asignar o insertar en el sistema a los eventos y/o personas que se vayan a guardar en el sistema, el resto de relaciones son de **UNO A MUCHOS**.

Último nivel o Diseño UML:

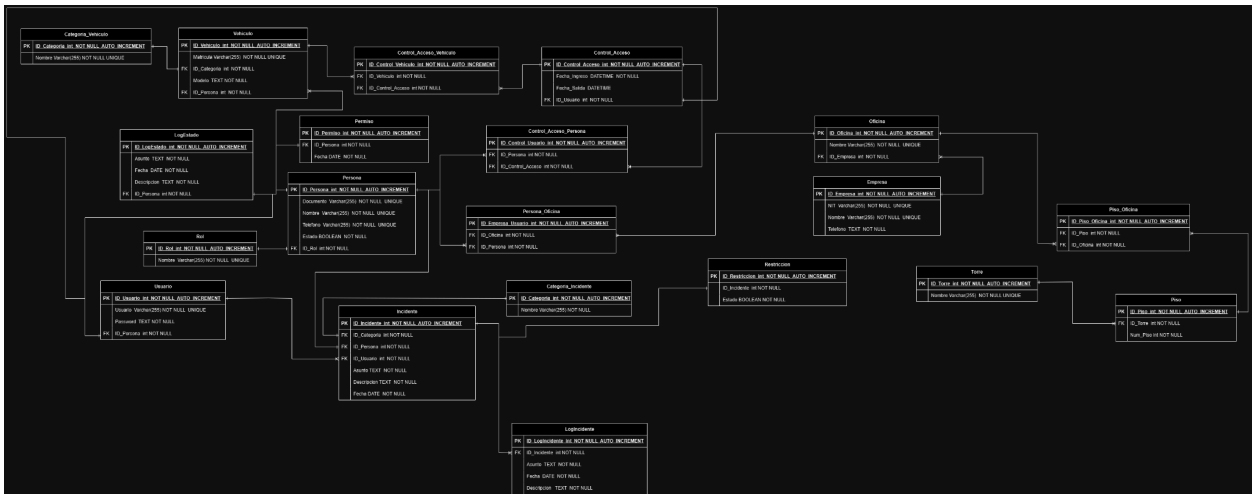


TABLA DE TRIGGERS Y FUNCIONES

Nombre	Tipo	Evento	Tabla Afectada	Acción Realizada	Propósito
registrar_actualizar_estado	Trigger	AFTER UPDATE	Persona	Inserta un registro en LogEstado cuando se hace un cambio en Estado	Llevar un registro de los cambios en el estado de una persona (activo/inactivo).
trg_check_control_acceso	Trigger	BEFORE INSERT	control_acceso	Verifica que Fecha_Salida sea mayor que Fecha_Ingreso.	Garantizar que las fechas de acceso sean lógicas y correctas.
trg_update_persona_estado	Trigger	AFTER INSERT	logestado	Actualiza el campo Estado en la tabla Persona basándose en el valor del campo Asunto.	Sincronizar el estado de la persona dependiendo de los registros en logestado.
validar_persona_en_vehiculo	Trigger	BEFORE INSERT	vehículo	Verifica que la Persona_idPersona asignada al vehículo exista en la tabla persona.	Garantizar la integridad de datos: no asignar un vehículo a personas no registradas.

validar_documento	Función	-	-	Verifica si el documento de una persona ya existe en la tabla Persona. Devuelve TRUE si no existe.	Garantizar la unicidad de los documentos antes de insertar una nueva persona.

ARQUITECTURA DEL SOFTWARE(MVC)

La relación que se utilizó en la arquitectura del software fue la siguiente:

- Modelo (Model):

  El modelo representa la lógica de negocio y los datos. En relación con DAO, el modelo utiliza el DAO para realizar la persistencia de datos en la base de datos.

- Vista (View):

  La vista es responsable de la interfaz de usuario. Presenta los datos del modelo y permite la interacción del usuario.

  a. Paquete: View

  b. Ejemplo: Clases o archivos que muestran la interfaz gráfica o textual.

  c. Relación con Controller: La vista se comunica con el controlador para notificar eventos generados por el usuario.

- Controlador (Controller):

  El controlador actúa como intermediario entre la vista y el modelo. Recibe las entradas del usuario a través de la vista, las procesa y actualiza el modelo.

  a. Paquete: Controller

  b. Ejemplo: Clases que manejan las peticiones y coordinan la lógica del negocio.

c. Relación con Model y View: El controlador recibe las acciones del usuario y actualiza la vista según los cambios en el modelo.

PATRONES DE DISEÑO

1. Patrón Proxy: Validación del inicio de sesión

El patrón Proxy se utiliza para proporcionar un acceso controlado a un objeto. En este caso, se usa para gestionar la validación del inicio de sesión del usuario. El proxy actúa como un intermediario entre el usuario y el sistema, interceptando las solicitudes de acceso y asegurándose de que la autenticación y autorización sean correctas antes de permitir el acceso al sistema. Este patrón ayuda a mejorar la seguridad y la eficiencia del sistema al gestionar las credenciales y permisos de forma centralizada.

Uso en el proyecto:

- El proxy realiza una validación del inicio de sesión para asegurarse de que el usuario tenga los permisos adecuados antes de acceder a las funcionalidades del sistema.
- Este patrón ayuda a mejorar el control de acceso y puede ser útil para gestionar la autenticación y las sesiones.

2. Patrón Strategy: Generar estrategias según el rol de la persona

El patrón Strategy permite definir una familia de algoritmos, encapsular cada uno y hacerlos intercambiables. En lugar de tener múltiples condicionales if o switch, el patrón Strategy permite que se seleccionen comportamientos o estrategias dinámicamente en función de una condición, como el rol de la persona en el sistema.

Uso en el proyecto:

- Dependiendo del rol del usuario (por ejemplo, administrador, supervisor, empleado), el sistema selecciona una estrategia para generar la interfaz correspondiente.
- Las diferentes estrategias definen las acciones o la interfaz de usuario que se presentan según el rol del usuario, facilitando la personalización y mejorando la experiencia del usuario.

3. Patrón DTO (Data Transfer Object):

El patrón DTO se utiliza para transferir datos entre sistemas o capas, reduciendo la cantidad de llamadas entre componentes y simplificando la comunicación de datos. Los DTOs son objetos que contienen sólo datos, sin lógica de negocio, y se utilizan para agrupar información y transferirla eficientemente.

Uso en el proyecto:

- **VehiculoDTO, PersonaDTO, UsuarioDTO, EmpresaDTO, ConfiguracionDTO, ControlAccesoDTO y ReportesDTO** son objetos que encapsulan los datos de cada entidad del sistema y facilitan la transferencia entre la capa de presentación y la capa de persistencia.
- Cada DTO se utiliza para representar las entidades en el sistema y se pasa entre capas sin lógica adicional, lo que mejora la eficiencia y facilita el mantenimiento del sistema.

4. Patrón DAO (Data Access Object):

El patrón DAO abstrae la interacción con la base de datos y proporciona una interfaz para acceder a los datos sin exponer los detalles de la implementación. El patrón DAO se utiliza para separar la lógica de negocio de la lógica de persistencia de datos.

Uso en el proyecto:

- **UsuarioDAO, SuperUsuarioDAO, SesionDAO, GuardaDAO, PersonaDAO, SupervisorDAO y EmpresaDAO** son clases que encapsulan la lógica de acceso a datos para sus respectivas entidades.
- Estas clases proporcionan métodos específicos para acceder a la base de datos, lo que facilita la manipulación de datos y hace que la capa de persistencia sea fácilmente intercambiable o modificable sin afectar al resto del sistema.

5. Patrón Singleton: Conexión MySQL

El patrón Singleton garantiza que una clase tenga una única instancia y proporciona un punto global de acceso a esa instancia. Este patrón se utiliza para controlar el acceso a recursos compartidos, como bases de datos, conexiones de red o archivos, asegurando que no haya duplicación de instancias.

Uso en el proyecto:

- El patrón Singleton se utiliza en el controller de conexión MySQL para garantizar que solo haya una instancia de la conexión a la base de datos durante la ejecución del sistema.
- De esta manera, se evita la creación de múltiples conexiones y se asegura que todas las operaciones de acceso a la base de datos se realicen de manera eficiente y centralizada, mejorando el rendimiento y el control de la conexión.

PRINCIPIOS SOLID

1. S: Principio de Responsabilidad Única (Single Responsibility Principle - SRP)

Este principio establece que una clase debe tener una única razón para cambiar, es decir, debe tener una única responsabilidad. Si una clase realiza más de una tarea, cualquier cambio en una de esas responsabilidades podría afectar a la clase de manera indeseada.

Aplicación en el proyecto:

- **DAO (Data Access Object):** Cada clase DAO (como UsuarioDAO, PersonaDAO) tiene la única responsabilidad de interactuar con la base de datos para la entidad correspondiente. No debe manejar lógica de negocio o de presentación.
- **DTOs (Data Transfer Objects):** Los DTOs contienen solo los datos necesarios para representar las entidades y no contienen lógica de negocio. Esto cumple con el principio de tener una única responsabilidad: representar datos.

2. O: Principio de Abierto/Cerrado (Open/Closed Principle - OCP)

El principio de abierto/cerrado establece que las clases deben estar abiertas para su extensión pero cerradas para su modificación. Esto significa que se debe poder agregar nuevas funcionalidades sin modificar el código existente.

Aplicación en el proyecto:

- **Estrategias con el patrón Strategy:** Cuando se agregan nuevos roles de usuario, se pueden agregar nuevas estrategias sin modificar las clases existentes. Por ejemplo, si se

agrega un nuevo tipo de usuario con una interfaz diferente, se puede crear una nueva estrategia sin necesidad de alterar el código de la interfaz de usuario.

- **DAO:** Si se necesita agregar nuevos tipos de acceso a datos, es posible extender los DAOs actuales sin cambiar su estructura básica. Nuevas clases DAO pueden ser añadidas sin afectar a las existentes, manteniendo la funcionalidad intacta.

3. L: Principio de Sustitución de Liskov (Liskov Substitution Principle - LSP)

Este principio establece que los objetos de una clase derivada deben poder reemplazar a los de la clase base sin afectar el comportamiento del programa. En otras palabras, una clase hija debe ser sustituible por su clase madre sin alterar el funcionamiento del sistema.

Aplicación en el proyecto:

- **Polimorfismo en patrones como Strategy y DAO:** Cuando se utilizan estrategias basadas en roles, las subclases de un tipo de estrategia deben ser sustituibles por su clase base sin que el comportamiento del sistema se vea afectado. Por ejemplo, si AdminStrategy extiende de RoleStrategy, se debe poder sustituir por otro tipo de estrategia sin que el sistema falle.
- **Herencia en el patrón DAO:** Las clases específicas de cada entidad pueden extender una clase DAO genérica, permitiendo que cualquier nueva clase hija pueda ser usada en lugar de la clase base, sin modificar el comportamiento del acceso a datos.

4. I: Principio de Segregación de la Interfaz (Interface Segregation Principle - ISP)

El principio de segregación de interfaces sugiere que no se debe forzar a los clientes a depender de interfaces que no utilizan. En lugar de tener una interfaz grande y generalizada, se deben crear interfaces más pequeñas y específicas.

Aplicación en el proyecto:

- **Interfaces específicas para diferentes estrategias y DAOs:** En lugar de tener una única interfaz que abarque todas las operaciones de un DAO o Strategy, cada interfaz está segregada en partes más pequeñas y especializadas. Por ejemplo, la interfaz de ControlAccesoDAO solo contiene los métodos necesarios para interactuar con los registros de control de acceso, no otros métodos de manipulación de datos.
- **Estrategias diferenciadas por roles:** Cada tipo de rol puede tener su propia interfaz que defina las acciones que ese rol puede realizar, evitando que todos los roles dependan de una interfaz demasiado amplia.

5. D: Principio de Inversión de Dependencias (Dependency Inversion Principle - DIP)

Este principio establece que las clases de alto nivel no deben depender de clases de bajo nivel, sino de abstracciones (interfaces). Las clases de bajo nivel deben depender de abstracciones, no de clases concretas.

Aplicación en el proyecto:

- **Uso de Interfaces en DAOs y Estrategias:** En lugar de que el código de negocio dependa directamente de una implementación concreta (por ejemplo, acceso a la base de datos), se define una interfaz que abstrae la funcionalidad de acceso a datos. Las clases

concretas implementan esta interfaz, y las clases de alto nivel (como los controladores o los servicios) interactúan con estas interfaces.

- **Controladores y Conexión a la Base de Datos:** El controlador de la conexión MySQL sigue el principio DIP, ya que depende de la interfaz Connection para gestionar las conexiones a la base de datos, en lugar de depender de una clase de conexión concreta.

IMPLEMENTACIÓN DEL SOFTWARE

Gestión de Roles y Credenciales en el Software

Los roles permiten **segregar responsabilidades** y asegurar que cada usuario tenga acceso únicamente a las funcionalidades necesarias según su perfil. A continuación, se describen los roles definidos para el sistema:

Rol	Descripción
Superusuario	Tiene control total del sistema. Puede administrar usuarios, permisos y toda la base de datos.
Supervisor	Gestiona usuarios, incidentes, anotaciones y monitorea las actividades del sistema.
Guarda	Se encarga de registrar accesos, salidas y consultar restricciones e incidentes.
Gerente	Gestiona la información de trabajadores e invitados asociados a la empresa.

Credenciales de Usuarios:

Se han definido credenciales de acceso para los usuarios asociados a cada rol. Estas credenciales se crean utilizando comandos SQL (CREATE USER) y se les asignan los roles correspondientes mediante GRANT.

Rol	Usuario	Contraseña	Asignación de Rol
Superusuario	superadmin	password_superadmin	GRANT Superusuario TO 'superadmin'@'%'
Supervisor	supervisor1	password_supervisor1	GRANT Supervisor TO 'supervisor1'@'%'
Supervisor	supervisor2	password_supervisor2	GRANT Supervisor TO 'supervisor2'@'%'
Guarda	guarda1	password_guarda1	GRANT Guarda TO 'guarda1'@'%'
Guarda	guarda2	password_guarda2	GRANT Guarda TO 'guarda2'@'%'
Gerente	gerente1	password_gerente1	GRANT Gerente TO 'gerente1'@'%'
Gerente	gerente2	password_gerente2	GRANT Gerente TO 'gerente2'@'%'

Permisos Asociados a Cada Rol

Los permisos controlan qué operaciones (SELECT, INSERT, UPDATE, DELETE) puede realizar cada rol sobre las tablas específicas.

Rol	Tabla	Permisos
Superusuario	*.*	ALL PRIVILEGES
Supervisor	usuario, rol, control_acceso, permiso, incidente, logestado,categoriavehiculo	SELECT, INSERT, UPDATE
	torre, piso, oficina, piso_oficina, logestado,	SELECT

	logincidente, control_acceso_persona, control_acceso_vehiculo	
	vehículo	SELECT, INSERT
Guarda	control_acceso_persona, control_acceso_vehiculo, vehiculo	SELECT, INSERT
	restriccion, incidente, control_acceso	SELECT
Gerente	persona, oficina_persona, categoria_incidente	SELECT, INSERT, UPDATE
	restriccion	SELECT, UPDATE

EXPLICACIÓN DE LOS MÓDULOS

La interfaz del software está organizada en módulos específicos según las funcionalidades y roles del sistema. A continuación, se detalla cada uno de ellos:

1. **Inicio de Sesión**

El módulo de **Inicio de Sesión** es el punto de acceso principal al sistema. Permite a los usuarios autenticarse con sus credenciales (usuario y contraseña) y, dependiendo del rol asignado, serán redirigidos a su respectiva área de trabajo dentro del sistema.

Flujo del Módulo:

1. **Autenticación:** El usuario ingresa su nombre de usuario y contraseña.
2. **Validación:** El sistema verifica las credenciales en la base de datos.
3. **Redirección:**
    a. Si las credenciales son correctas, se redirige al usuario a su módulo correspondiente:
        - **Superusuario**

- **Supervisor**
- **Guarda**
- **Gerente**

b. Si las credenciales son incorrectas, se muestra un mensaje de error.

2. Superusuario

El módulo del Superusuario tiene el control total del sistema. Desde este módulo se pueden realizar las siguientes acciones:

- Crear, insertar y modificar registros de cualquier módulo.
- Gestionar usuarios y asignar roles.
- Monitorear el funcionamiento general del sistema.
- Acceso a todas las funcionalidades sin restricciones.

3. Supervisor

El módulo del Supervisor permite gestionar y supervisar las operaciones diarias. Sus principales funcionalidades incluyen:

- **Gestión de Guardas:** Creación, actualización y consulta de registros.
- **Gestión de Incidentes:** Registro y monitoreo de anotaciones e incidentes.
- **Control de Acceso:** Gestión de accesos y salidas al sistema.
- **Monitoreo:** Acceso a logs y estados generales del sistema.

4. Guarda

El módulo del Guarda está diseñado para facilitar las tareas relacionadas con el control de acceso. Sus funcionalidades son:

- **Registro de accesos y salidas:**
    a. Personas
    b. Vehículos
- **Consulta de restricciones:** Verificación de permisos y restricciones.
- **Registro de incidentes:** Reporte básico de situaciones específicas.

5. Gerente

El módulo del **Gerente** está enfocado en la gestión de personal e invitados dentro de la empresa. Sus principales funciones son:

- **Gestión de trabajadores e invitados:** Registro y actualización de información.
- **Consultas de restricciones:** Acceso a permisos y restricciones de personal.
- **Acceso a reportes:** Visualización de incidentes y registros específicos.

-