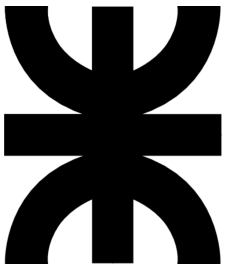

Univesidad Tecnologica Nacional

Instituto Nacional de Profesorado Tecnico

Proyecto final

Autor:
Fernandez Lautaro, Fullana Matias

Profesor/es:
Miguel Silva

Comicion:
2.603

24 de noviembre de 2025

Índice

1. Consigna **2**

2. Descripción General del Proyecto **3**
2.1. Tema . . . 3
2.2. Objetivos del Sistema . . . 3
2.3. Descripción de Actores y Roles . . . 3

3. Casos de Uso y Diagramas **3**
3.1. Casos de Uso Comunes . . . 3
3.2. Actor: Oficial . . . 4
3.3. Actor: Suboficial . . . 5
3.4. Actor: Soldado . . . 5

4. Arquitectura y Desarrollo **6**
4.1. Descripción de las Capas del Sistema . . . 6
4.1.1. 1. Capa de Presentación (Frontend) . . . 6
4.1.2. Capa de Controladores (Controller Layer) . . . 6
4.1.3. Capa de Acceso a Datos (DAO Layer) . . . 7
4.1.4. Capa de Persistencia (Database Layer) . . . 7
4.2. Tecnologías y Herramientas Utilizadas . . . 7
4.2.1. Backend (Lado del Servidor) . . . 7
4.2.2. Frontend (Lado del Cliente) . . . 7
4.2.3. Base de Datos y Herramientas . . . 8

5. Lógica de Negocio y Algoritmos **8**
5.1. Cálculo de Estado de Servicios . . . 8
5.2. Refactorización Frontend (Principio DRY) . . . 8

6. Base de Datos **8**
6.1. Diagrama Entidad-Relación (DER) . . . 8
6.2. Diccionario de Datos . . . 8
6.2.1. Tablas Maestras y de Usuarios . . . 8
6.2.2. Tablas Estructurales (Organización) . . . 9
6.2.3. Tabla Transaccional (El Corazón del Sistema) . . . 9
6.3. Relaciones y Cardinalidad . . . 9

7. Manual de Usuario **10**
7.1. Acceso General al Sistema . . . 10
7.2. Manual del Oficial (Administrador) . . . 10
7.2.1. Gestión de Entidades . . . 10
7.2.2. Alta de Personal . . . 10
7.2.3. Auditoría de Servicios . . . 10
7.3. Manual del Suboficial . . . 10
7.3.1. Creación y Asignación de Tareas . . . 10
7.3.2. Gestión de Tropa . . . 11
7.4. Manual del Soldado . . . 11
7.4.1. Visualización de Perfil . . . 11
7.4.2. Gestión de Servicios Asignados . . . 11
7.4.3. Seguridad de la Cuenta . . . 11

8. Repositorio del Proyecto **11**

1. Consigna

Desarrolle un sistema para gestionar la información de los soldados que realizan el servicio militar, teniendo en cuenta que:

- Un soldado se define por su código de soldado (único), su nombre y apellidos, y su graduación
- Existen varios cuarteles, cada uno se define por su código de cuartel, nombre y ubicación.
- Hay que tener en cuenta que existen diferentes Cuerpos del Ejército (Infantería, Artillería, Caballería...), y cada uno se define por un código de Cuerpo y denominación.
- Los soldados están agrupados en compañías, siendo significativa para cada una de éstas, el número de compañía y la actividad principal que realiza
- Se desea controlar los servicios que realizan los soldados (correr, limpiar, barrer...), y se definen por el código de servicio y descripción.

Consideraciones de diseño:

- Un soldado pertenece a un único cuerpo y a una única compañía, durante todo el servicio militar. A una compañía pueden pertenecer soldados de diferentes cuerpos, no habiendo relación directa entre compañías y cuerpos.
- Los soldados de una misma compañía pueden estar destinados en diferentes cuarteles, es decir, una compañía puede estar ubicada en varios cuarteles, y en un cuartel puede haber varias compañías. Eso sí, un soldado sólo está en un cuartel.
- Un soldado realiza varios servicios a lo largo de su paso por el servicio militar. Un mismo servicio puede ser realizado por más de un soldado (con independencia de la compañía), siendo significativa la fecha de realización.
- Al sistema podrán acceder tres tipos de usuarios: soldados (que sólo lo podrán consultar), suboficiales (que administrarán a los soldados y los servicios) y oficiales (que administrarán todo).

Para ello:

- Analice los requerimientos anteriores.
- Determine los objetos requeridos para implementar ese sistema.
- Establezca los atributos que deben tener estos objetos.
- Fije los comportamientos que exhibirán estos objetos.
- Especifique la forma en que los objetos deben interactuar entre sí para cumplir con los requerimientos del sistema.

El sistema deberá utilizar abstracción, encapsulamiento, herencia, polimorfismo y persistencia (no BD). La E/S del sistema será exclusivamente por consola (no GUI). Se deberán subir a GitHub el ejecutable (en formato jar), el código fuente, la documentación (generada con javadoc) y los diagramas UML de caso-uso y de clases (generados con `http://plantuml.com/es` o `https://www.umletino.com/umletino.html` y grabados en formato png).

2. Descripción General del Proyecto

2.1. Tema

El proyecto consiste en un **Sistema de Gestión Integral para una Unidad Militar**. El sistema administra la jerarquía de personal, la estructura organizativa (Cuerpos, Compañías, Cuarteles) y, fundamentalmente, el ciclo de vida de las asignaciones de servicios y tareas operativas.

2.2. Objetivos del Sistema

El objetivo principal es digitalizar la administración del cuartel, permitiendo un control estricto sobre quién realiza qué tarea y cuándo.

- **Para Oficiales:** Proveer una plataforma de administración total para gestionar la infraestructura y el alta de personal con seguridad criptográfica.
- **Para Suboficiales:** Ofrecer herramientas operativas para la asignación de tareas a la tropa, con validaciones de rango para evitar errores jerárquicos.
- **Para Soldados:** Brindar un portal personal para consultar sus órdenes del día, ver su ubicación y reportar el cumplimiento de servicios en tiempo real.

2.3. Descripción de Actores y Roles

El sistema implementa una lógica de seguridad basada en roles con permisos escalonados:

1. **OFICIAL (Administrador):**
   - Gestión completa (Alta/Baja/Modificación) de la estructura militar (Cuerpos, Cuarteles, Compañías).
   - Alta de nuevos usuarios con asignación de roles y hash de contraseñas.
   - Auditoría global de servicios de cualquier efectivo.
2. **SUBOFICIAL (Supervisor):**
   - Creación de nuevos tipos de servicios (ej. Guardia, Cocina).
   - Asignación de tareas a Soldados y otros Suboficiales.
   - *Restricción de Seguridad:* El sistema impide por backend que un Suboficial asigne tareas a un Oficial.
3. **SOLDADO (Usuario Final):**
   - Consulta de perfil y ubicación asignada.
   - Visualización de servicios asignados con estado dinámico (Pendiente/Realizado/Vencido).
   - Acción de marcar servicios como Çompletados".

3. Casos de Uso y Diagramas

Esta sección detalla las funcionalidades disponibles para cada actor del sistema. Los casos de uso están segregados según el rol del usuario autenticado, garantizando el principio de menor privilegio.

3.1. Casos de Uso Comunes

Existen funcionalidades transversales a todos los usuarios, independientemente de su jerarquía:

- **Autenticarse (Login):** Todos los usuarios deben ingresar sus credenciales (email y contraseña) para acceder al sistema. El backend valida el rol y redirige al menú correspondiente.
- **Consultar Perfil:** Visualización de datos personales básicos.

3.2. Actor: Oficial

El Oficial posee el rol de administrador total del sistema. Sus casos de uso se centran en la gestión de la estructura militar y el personal.

- **Gestionar Cuerpos, Cuarteles y Compañías (ABM):** Permite dar de alta, baja o modificar las entidades organizativas del ejército. Es el primer paso para poblar la base de datos.
- **Gestionar Servicios (ABM):** Capacidad de definir qué tipos de tareas pueden realizarse en el cuartel.
- **Gestionar Soldados (ABM):** Incluye el alta de nuevos usuarios en el sistema, asignándoles un rol (Soldado, Suboficial, Oficial) y generando sus credenciales de acceso seguras.
- **Gestionar Asignaciones:** El Oficial es el encargado de vincular a la tropa con su destino.
  - *Soldados - Servicio:* Asignación directa de tareas.
  - *Soldados - Compañía/Cuerpo/Cuartel:* Definición de la ubicación y pertenencia del personal.

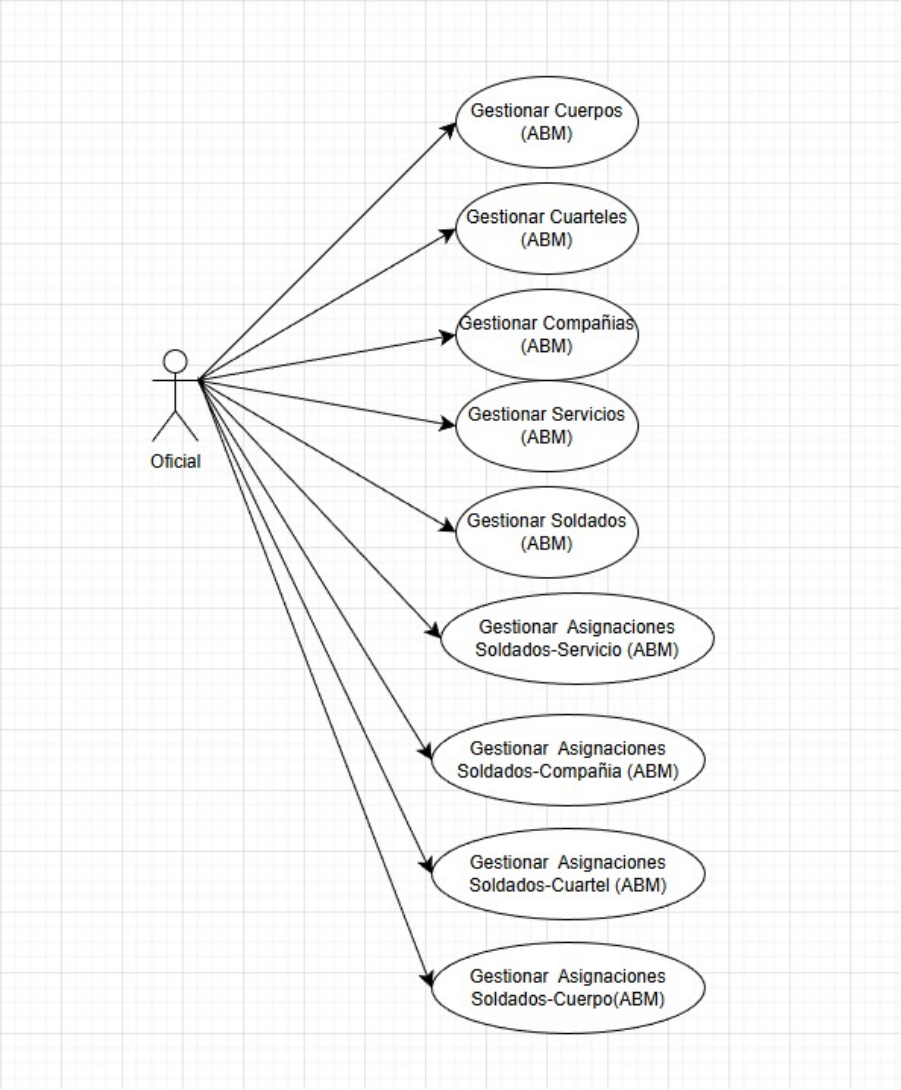

Figura 1: Diagrama de Casos de Uso - Oficial

3.3. Actor: Suboficial

El Suboficial actúa como un mando intermedio u operativo. Tiene permisos de gestión sobre la tropa, pero también posee atributos de Üsuarioçon asignaciones propias.

- **Consultas Personales:** Al igual que un soldado, el Suboficial puede consultar sus propios servicios, cuerpo, compañía y cuartel asignados.
- **Gestionar Servicios (ABM):** Tiene permisos para crear nuevas instancias de servicios operativos.
- **Gestionar Soldados (ABM):** En el contexto de la aplicación, esto se refiere a la capacidad de visualizar el listado de tropa a su cargo y asignarles tareas específicas (siempre respetando la restricción de no asignar tareas a un Oficial).

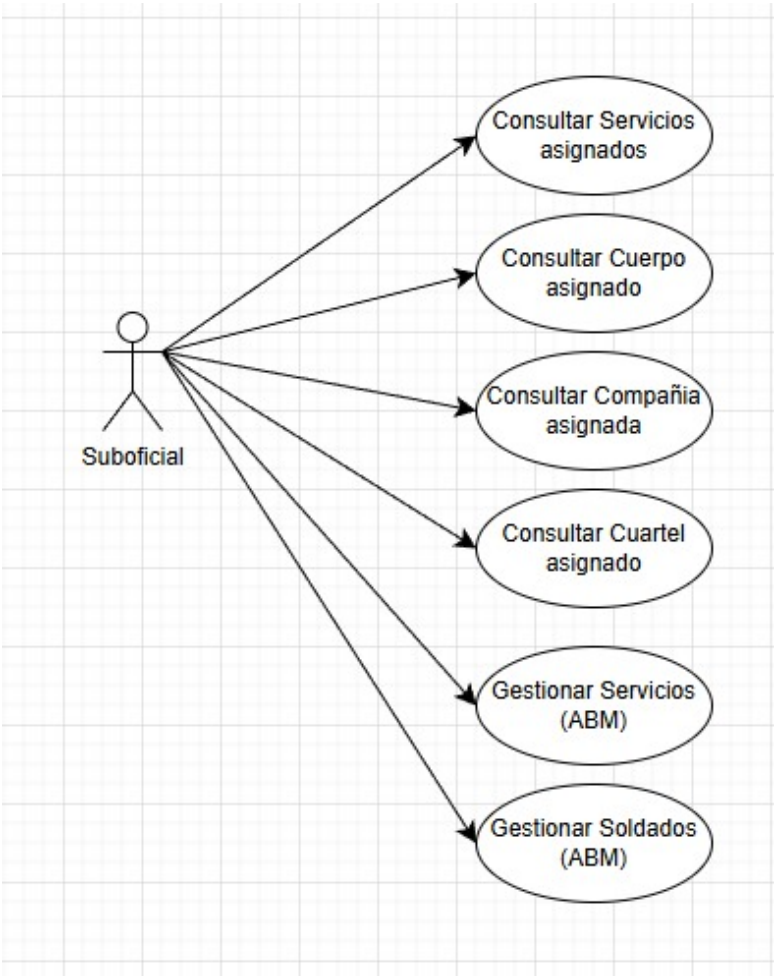

Figura 2: Diagrama de Casos de Uso - Suboficial

3.4. Actor: Soldado

Es el usuario final del sistema. Sus permisos son mayoritariamente de lectura (consulta) sobre su propia información, con interacción limitada a la actualización de estado de sus tareas.

- **Consultar Servicios Asignados:** Visualización de la lista de órdenes. Desde aquí se deriva la acción de marcar un servicio como Realizado"(registrando la fecha de completado).
- **Consultar Cuerpo Asignado:** Visualización de su pertenencia a un cuerpo específico (ej. Infantería).
- **Consultar Compañía Asignada:** Visualización de su compañía.
- **Consultar Cuartel Asignado:** Información sobre su ubicación física actual.

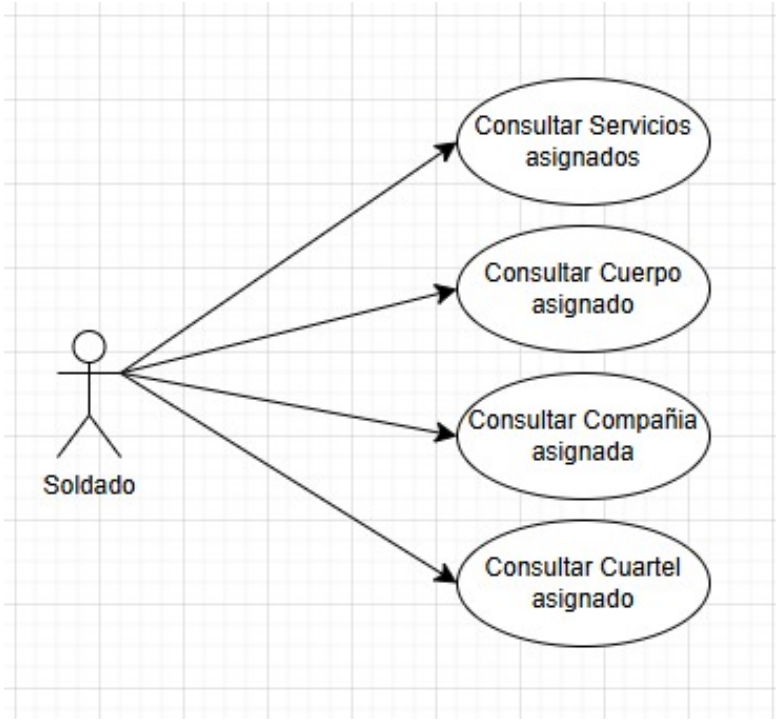

Figura 3: Diagrama de Casos de Uso - Soldado

4. Arquitectura y Desarrollo

4.1. Descripción de las Capas del Sistema

El proyecto se ha diseñado siguiendo un patrón de arquitectura en capas (Layered Architecture) bajo un modelo Cliente-Servidor desacoplado. Esta estructura asegura la separación de responsabilidades (Separation of Concerns), facilitando el mantenimiento y la escalabilidad del software.

4.1.1. 1. Capa de Presentación (Frontend)

Esta capa es la interfaz visual con la que interactúan los usuarios (Oficiales, Suboficiales y Soldados).

- **Responsabilidad:** Renderizar la interfaz de usuario y capturar eventos. No procesa reglas de negocio críticas, sino que delega esa responsabilidad al servidor.
- **Comunicación:** Se comunica con el Backend exclusivamente a través de llamadas asíncronas (`AJAX/Fetch API`), enviando y recibiendo datos en formato **JSON**.
- **Gestión de Sesión:** Utiliza `sessionStorage` en el navegador para persistir temporalmente la información del usuario autenticado y su rol, permitiendo la protección de rutas a nivel de cliente (redirecciones si no está logueado).

4.1.2. Capa de Controladores (Controller Layer)

Es el punto de entrada a la lógica del servidor. Se implementa utilizando la anotación `@RestController` de Spring Boot.

- **Responsabilidad:** Exponer los *endpoints* (puntos de acceso) de la API REST. Recibe las peticiones HTTP (GET, POST, PUT), deserializa el JSON entrante a objetos Java (DTOs o Entidades) y delega la acción a la capa de datos.
- **Manejo de Errores:** Gestiona las respuestas HTTP adecuadas (200 OK, 403 Forbidden, 500 Internal Server Error) según el resultado de la operación.

4.1.3. Capa de Acceso a Datos (DAO Layer)

A diferencia de los frameworks ORM automáticos, este proyecto implementa el patrón **DAO (Data Access Object)** utilizando JDBC puro.

- **Responsabilidad:** Abstraer y encapsular todo el acceso a la fuente de datos. Aquí reside la lógica SQL.
- **Seguridad y Rendimiento:** Se utiliza `PreparedStatement` en todas las consultas. Esto no solo precompila la sentencia SQL mejorando la velocidad, sino que es la barrera principal contra ataques de Inyección SQL.
- **Lógica de Negocio:** En esta arquitectura, el DAO también encapsula reglas de negocio relacionadas con los datos, como el cálculo dinámico del estado de un servicio (comparación de fechas `Timestamp` vs `LocalDate`) antes de devolver el objeto a la capa superior.

4.1.4. Capa de Persistencia (Database Layer)

El nivel más bajo de la arquitectura, encargado del almacenamiento físico de la información relacional, garantizando la integridad referencial mediante claves foráneas (FK).

4.2. Tecnologías y Herramientas Utilizadas

La selección tecnológica priorizó el rendimiento, el control sobre los datos y la seguridad estándar de la industria.

4.2.1. Backend (Lado del Servidor)

- **Java 17 (LTS):** Lenguaje base del proyecto, seleccionado por su tipado fuerte y robustez en entornos empresariales.
- **Spring Boot 3:** Framework que actúa como contenedor de la aplicación. Provee:
  - **Servidor Embebido:** Apache Tomcat, permitiendo que la aplicación sea autoejecutable.
  - **Inyección de Dependencias:** Gestión automática del ciclo de vida de los objetos (Controllers y Beans).
- **Spring Security 6:** Framework de seguridad encargado de la protección de la aplicación.
  - Se configuró un `SecurityFilterChain` personalizado para proteger los endpoints de la API.
  - Implementación del algoritmo de hashing **BCrypt**. Este algoritmo aplica un "salt.ªleatorio a las contraseñas antes de guardarlas, haciendo que sean irreversibles y seguras contra ataques de diccionario.
- **JDBC (Java Database Connectivity):** API de bajo nivel para la conexión a base de datos. Se eligió sobre JPA/Hibernate para tener un control manual y optimizado sobre las sentencias SQL complejas, especialmente en los reportes de servicios.

4.2.2. Frontend (Lado del Cliente)

- **HTML5 Semántico:** Estructura base de las vistas (`soldado.html`, `oficial.html`).
- **CSS3 (Vanilla):** Se diseñó una hoja de estilos personalizada (`oficial-estilos.css`) implementando un diseño responsivo y una paleta de colores en "Modo Oscuro"para reducir la fatiga visual operativa.
- **JavaScript (ES6+):** Lógica del cliente. Se destaca el uso de:
  - **Async/Await:** Para manejar las promesas de las peticiones a la red de forma limpia.
  - **DOM Manipulation:** Para renderizar dinámicamente las tablas de soldados y listas de servicios sin recargar la página.
  - **Módulos Compartidos:** Implementación de `shared.js` para reutilizar lógica transversal (DRY).

4.2.3. Base de Datos y Herramientas

- **MySQL 8:** Motor de base de datos relacional.
- **Apache Maven:** Gestor de dependencias y automatización de la construcción del proyecto (`pom.xml`).
- **HeidiSQL / DBeaver:** Herramientas utilizadas para el diseño del esquema y pruebas de consultas SQL directas.

5. Lógica de Negocio y Algoritmos

5.1. Cálculo de Estado de Servicios

El sistema no almacena simplemente si una tarea está "hecha.º no. El DAO implementa una lógica temporal al momento de la consulta:

- Si `estado` es falso y la fecha actual es posterior a la asignada: **VENCIDO**.
- Si `estado` es verdadero y la fecha de completado es igual a la asignada: **REALIZADO EN FECHA**.
- Si `estado` es verdadero pero la fecha difiere: **REALIZADO CON DEMORA**.

5.2. Refactorización Frontend (Principio DRY)

Se implementó un archivo `shared.js` para centralizar la lógica común, cumpliendo con el principio *Don't Repeat Yourself*:

- Validación de sesión y roles.
- Renderizado dinámico de listas de servicios.
- Manejo de alertas y logout.

6. Base de Datos

El sistema se sustenta sobre una base de datos relacional diseñada en MySQL. El modelo garantiza la integridad referencial y la normalización de datos hasta la tercera forma normal (3NF).

6.1. Diagrama Entidad-Relación (DER)

A continuación, se presenta el esquema lógico de la base de datos, ilustrando las entidades, sus atributos y las relaciones cardinales entre ellas.

6.2. Diccionario de Datos

Descripción técnica de las tablas principales mostradas en la Figura 4:

6.2.1. Tablas Maestras y de Usuarios

- **usuario:** Es la entidad central del sistema.
  - `id_user` (PK): Identificador único.
  - `rol`: Definido como un `ENUM`, restringe los valores posibles a los perfiles del sistema (SOLDADO, SUBOFICIAL, OFICIAL), asegurando la consistencia de la seguridad.
  - `contrasenia`: Almacena el hash BCrypt (varchar de longitud suficiente).
  - **Claves Foráneas (FK):** Mantiene relaciones N:1 con `compania`, `cuerpo` y `cuartel`, ubicando al efectivo dentro de la jerarquía militar.
- **servicios:** Catálogo de tareas disponibles.
  - `nombre_servicio`: Título corto de la tarea (ej. "Guardia").
  - `descripcion`: Detalle operativo de la tarea.

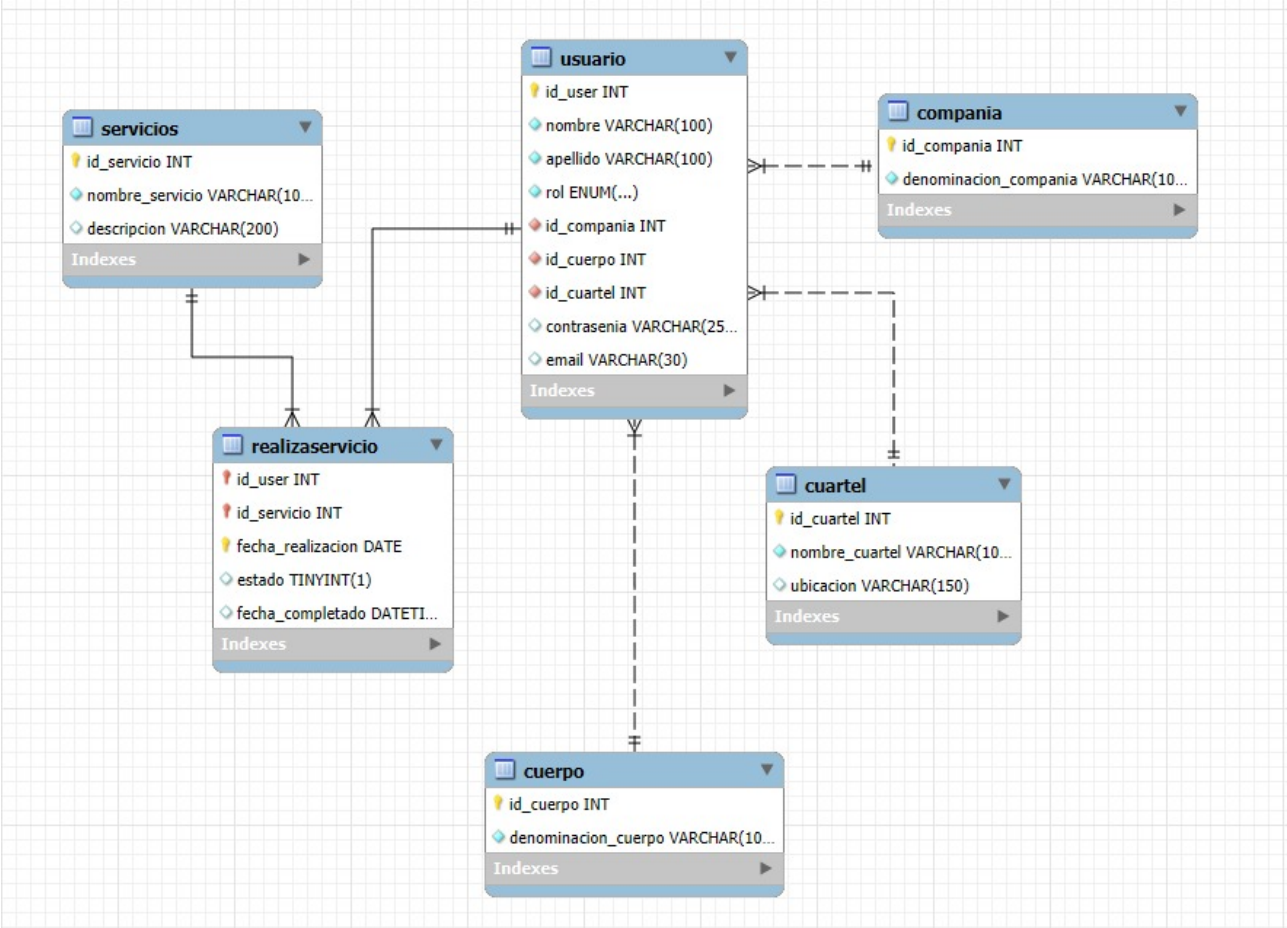

Figura 4: Diagrama Entidad-Relación del Sistema Militar

6.2.2. Tablas Estructurales (Organización)

Estas tablas actúan como catálogos para normalizar la estructura de la unidad militar, evitando la redundancia de datos en la tabla de usuarios.

- **cuerpo:** Define la rama o especialidad (ej. Ïnfantería").
- **compania:** Define la subdivisión operativa.
- **cuartel:** Define la ubicación física (`nombre` y `ubicacion`).

6.2.3. Tabla Transaccional (El Corazón del Sistema)

- **realizaservicio:** Esta tabla resuelve la relación de cardinalidad Muchos a Muchos (N:M) entre *Usuarios* y *Servicios*. Es donde reside la lógica de negocio temporal.
  - `id_user` y `id_servicio`: Claves foráneas que vinculan al ejecutante con la tarea.
  - `fecha_realizacion` (DATE): Indica cuándo se **asignó** la tarea.
  - `estado` (TINYINT): Actúa como booleano (0 = Pendiente, 1 = Realizado).
  - `fecha_completado` (DATETIME): Registra el momento exacto (fecha y hora) en que el soldado marcó la tarea como terminada. Este campo permite calcular si la tarea se realizó en tiempo y forma o con demora.

6.3. Relaciones y Cardinalidad

El modelo implementa dos tipos principales de relaciones:

1. **Uno a Muchos (1:N):** Un Cuartel/Cuerpo/Compañía puede tener asignados múltiples Usuarios, pero un Usuario pertenece a una sola instancia de cada uno de estos.
2. **Muchos a Muchos (N:M):** Un Usuario puede realizar múltiples Servicios a lo largo del tiempo, y un tipo de Servicio puede ser realizado por múltiples Usuarios. Esta relación se gestiona mediante la tabla intermedia `realizaservicio`.

7. Manual de Usuario

Esta sección describe el flujo de operación del sistema para cada uno de los perfiles de usuario. La interfaz ha sido unificada bajo un diseño visual de alto contraste ("Modo Oscuro") para facilitar su uso en entornos operativos.

7.1. Acceso General al Sistema

1. **Ingreso:** Navegue a la URL de la aplicación
2. **Autenticación:** Ingrese su correo electrónico y contraseña en el formulario de Login.
3. **Redirección:** El sistema detectará automáticamente su rango (Oficial, Suboficial o Soldado) y lo redirigirá al portal correspondiente. Si las credenciales son inválidas, se mostrará una alerta en pantalla.

7.2. Manual del Oficial (Administrador)

El Oficial tiene acceso al panel de control total.

7.2.1. Gestión de Entidades

En la sección principal del tablero, encontrará formularios para expandir la infraestructura:

- **Crear Cuerpo/Compañía/Cuartel:** Ingrese el nombre y ubicación. Al pulsar el botón azul Çrear", la entidad se guardará inmediatamente en la base de datos para ser utilizada en nuevas asignaciones.

7.2.2. Alta de Personal

Para registrar nuevos efectivos en el sistema:

1. Localice la tarjeta .ªlta de Personal".
2. Complete los datos personales (Nombre, Apellido, Email).
3. **Importante:** Seleccione el ROL en la lista desplegable (Soldado, Suboficial, Oficial).
4. Ingrese los IDs numéricos de su destino (Cuerpo, Compañía, Cuartel).
5. Al hacer clic en Registrar Usuario", el sistema creará la cuenta y cifrará la contraseña automáticamente.

7.2.3. Auditoría de Servicios

- **Buscador:** Ingrese el ID de *cualquier* usuario (sin restricciones de rango) en el campo de búsqueda.
- **Resultados:** Se desplegará el historial completo de servicios de ese efectivo, indicando si están pendientes o realizados.

7.3. Manual del Suboficial

El Suboficial opera como supervisor de la tropa.

7.3.1. Creación y Asignación de Tareas

- **Crear Nuevo Servicio:** Permite definir nuevas tareas en el catálogo (ej. "Guardia Nocturna").
- **Asignar Servicio:** Ingrese el ID del Usuario destino y el ID del Servicio.

- **Validación de Rango:** Si intenta asignar una tarea a un ID correspondiente a un Oficial, el sistema bloqueará la operación y mostrará una alerta de error: "No tiene permiso para asignar servicios a un Oficial".

7.3.2. Gestión de Tropa

- **Listar Soldados:** El botón ."Actualizar Lista"descarga en tiempo real la nómina de soldados registrados, mostrando sus datos básicos para facilitar la asignación de tareas.

7.4. Manual del Soldado

El portal del soldado está optimizado para la visualización rápida de órdenes y estado.

7.4.1. Visualización de Perfil

Al ingresar, la tarjeta "Mis Datos"mostrará automáticamente su nombre, rango y la ubicación (Cuartel/Compañía) a la que ha sido asignado por el Oficial.

7.4.2. Gestión de Servicios Asignados

En la sección "Mis Servicios Asignados", verá su lista de tareas con indicadores de estado inteligentes:

- **Pendiente:** La tarea debe realizarse. Aparecerá un botón verde **"Marcar Realizado"**.
- **Acción de Completar:** Al pulsar el botón, el sistema registra la fecha y hora exacta.
- **Estados Finales:** Una vez completada, la tarea cambiará de estado visualmente a:
  - **Realizado en fecha:** Si se completó el mismo día de la asignación.
  - **Realizado con demora:** Si la fecha de completado es posterior a la asignación.
  - **Vencido:** Si la tarea sigue pendiente y la fecha ya pasó.

7.4.3. Seguridad de la Cuenta

El soldado dispone de un formulario para actualizar su contraseña. El sistema validará que la nueva contraseña y la confirmación coincidan antes de enviar el cambio cifrado al servidor.

8. Repositorio del Proyecto

`https://github.com/M4vixs/ProyectoFinalPro2`