

MVP (PRODUCTO MÍNIMO VIABLE)

El siguiente documento presenta la estructura, planificación, diseño y componentes esenciales del Producto Mínimo Viable (MVP) desarrollado por el equipo del espacio académico.

En él se describe la información general del proyecto, el contexto del problema abordado, la propuesta tecnológica seleccionada, el modelo de datos, los módulos funcionales y los resultados obtenidos durante el proceso de desarrollo. Igualmente, se incluyen los anexos correspondientes que permiten evidenciar el funcionamiento del sistema, tales como la presentación, el video demostrativo y el repositorio del código fuente.

Este MVP busca dar solución a una necesidad real, aportando valor al Hospital San Rafael de Tunja (HSRT) mediante un prototipo funcional que demuestra la viabilidad técnica de la propuesta abordada en el espacio académico en calidad de proyecto integrador.

INFORMACIÓN GENERAL

Integrantes (máximo 5):

- 1.Johan Hurtado
- 2.William Cruz
- 3.Alejandro Torres
- 4.Juan José Rincón
- 5.Santiago Ordoñez

Espacio académico:

CONTEXTO DEL PROBLEMA ABORDADO

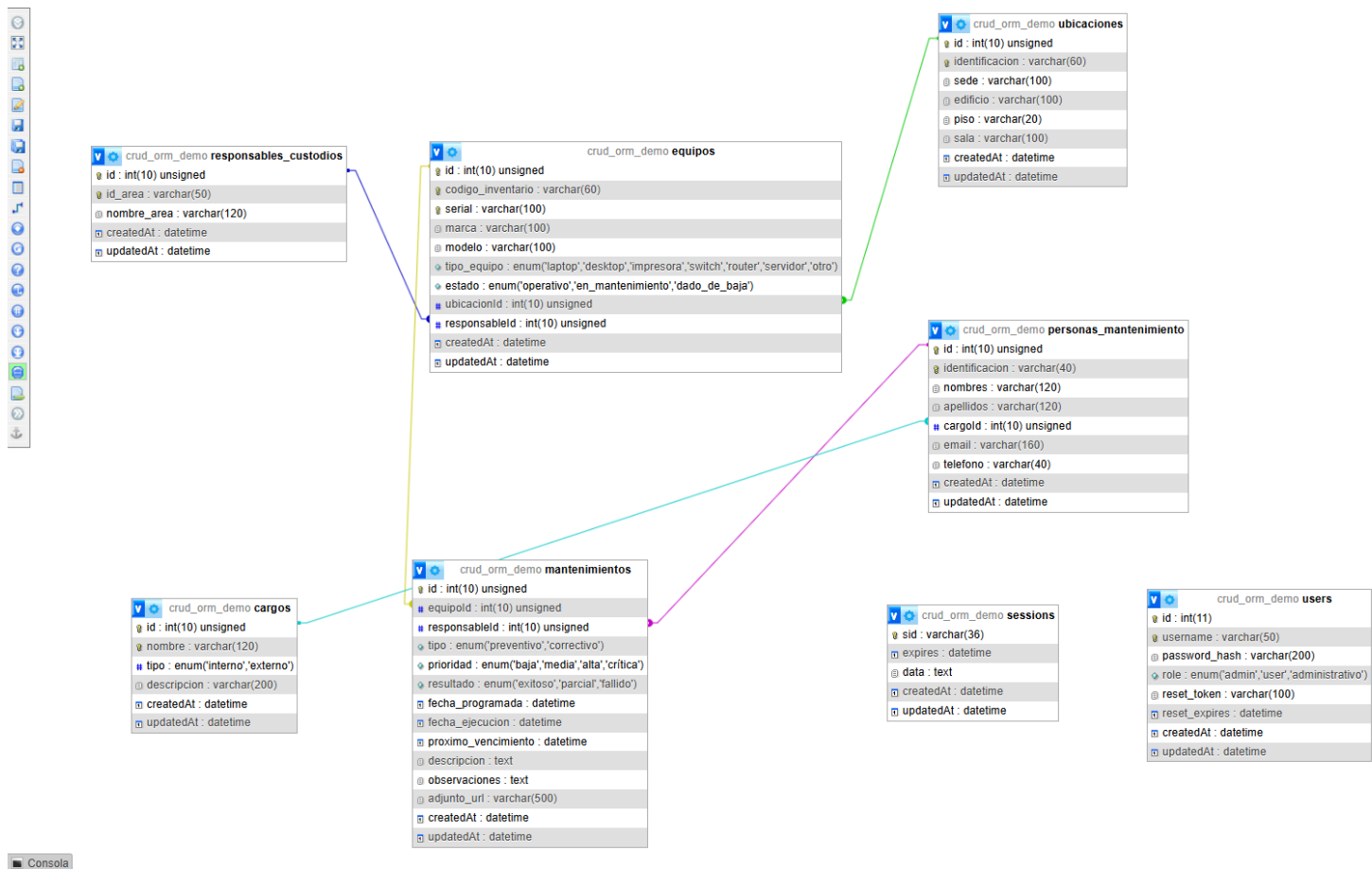
- ❖ Código del proyecto: PDS-001
- ❖ Nombre del proyecto: *Módulo de gestión de inventario y registro de mantenimientos de equipos informáticos*
- ❖ ODS relacionados (a consideración del equipo):
- ❖ Usuario objetivo: enunciar cómo se beneficiará el Hospital San Rafael de Tunja HSRT

PLANEACIÓN DEL PROYECTO

- ❖ Objetivo general: Desarrollar un sistema integral para la gestión, control y seguimiento del inventario y mantenimiento de equipos biomédicos, que permita administrar ubicaciones hospitalarias, responsables clínicos, personal de soporte, cargas laborales y procesos de mantenimiento, garantizando la trazabilidad, organización y disponibilidad de la información para optimizar la operación y la seguridad tecnológica dentro del Hospital San Rafael de Tunja
- ❖ Framework de FE: EJS (Server-side rendering) + Vanilla JavaScript (no SPA framework)
- ❖ Framework de BE: Express (Node.js)

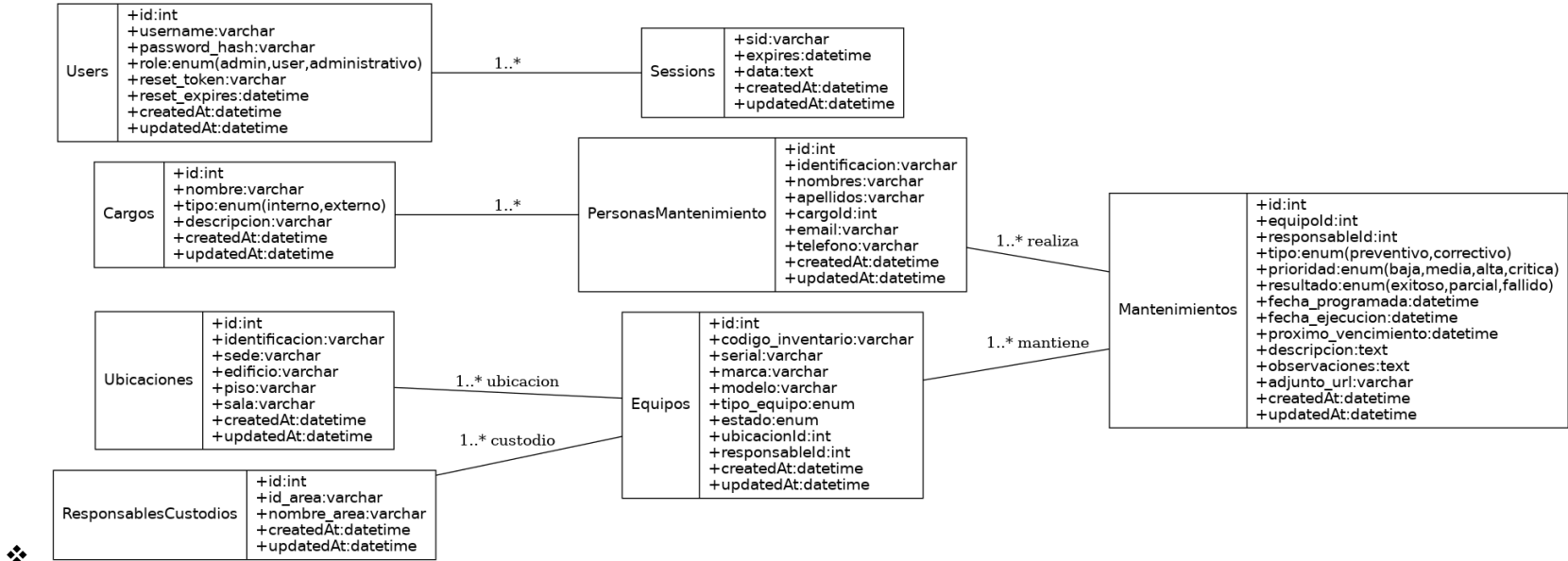
BASE DE DATOS (BAAS O AUTOGESTIONADA)

- ❖ Modelo relacional



RESULTADOS

- ❖ Diagramas UML (casos de uso, clases, componentes / otro aplicable)



❖❖ **Módulos funcionales – funcionalidades (Core):**
Módulos funcionales – funcionalidades (Core):

- Autenticación y roles: Login, logout, recuperación/recambio de contraseña, gestión de roles (admin, user, administrativo), protección de rutas.
- Gestión de Usuarios: CRUD de usuarios, cambio de contraseñas, validaciones de unicidad y seguridad.
- Gestión de Equipos: CRUD de equipos (código inventario, serial, marca, modelo, tipo, estado), búsquedas y validaciones de unicidad.
- Gestión de Ubicaciones: CRUD de ubicaciones físicas (sede, edificio, piso, sala), listados y restricciones al eliminar.
- Gestión de responsables / Custodios: CRUD de responsables, asociación con equipos.
- Gestión de Cargos y Personas de Mantenimiento: CRUD de cargos; CRUD de personas de mantenimiento con relación a cargos.
- Mantenimientos: Planificación y ejecución (preventivo/correctivo), asignación de responsable, estados/resultados, validaciones de fechas, historiales.
- Dashboard y accesos rápidos: Página principal con indicadores clave (total equipos, equipos en mantenimiento, próximos mantenimientos, número de ubicaciones, usuarios activos), gráficas simples (estado por tipo/ubicación) y panel de "accesos rápidos" con enlaces a acciones comunes (crear equipo, nueva orden de mantenimiento, exportar reporte, gestionar usuarios).
- Módulo de Reportes: Generación y descarga de reportes (por ejemplo, mantenimientos) con filtros (fecha desde/hasta, equipo, responsable, tipo, prioridad), exportación a Excel (exceljs) y endpoint de descarga (/reportes/mantenimientos.xlsx).
- Uploads y almacenamiento: Subida segura de archivos (PDF/JPG/PNG), almacenamiento en uploads y servir recursos estáticos.
- Sesiones y persistencia: Gestión de sesiones con express-session y connect-session-sequelize (sesiones en BD).
- API REST + Vistas SSR: Endpoints JSON para consumo AJAX y rutas que renderizan vistas EJS para la UI.
- Salud y utilidades: Endpoint de health (/api/health), formateo de fechas y DTOs sanitizados.

ANEXOS

- URL de la presentación: [Presentacion Construccion.pptx](#)
- URL del video YouTube: <https://youtu.be/LDcQ2R-xhLw>
- URL del repositorio: https://github.com/johanHurtado/Project_Integrador4z