

Documentación Sistema Smart-House

Problemática: Hoy en día las casas deben de manejarse de manera manual a través de una persona, quitándole tiempo el cual puede utilizar para algo más relevante, además, esto puede generar gastos de agua y eléctricos si esa persona se encuentra muy atareada con su trabajo o entornos sociales.

Solución: El sistema SmartHouse busca apoyar las actividades diarias del cliente automatizando de manera inteligente los diferentes comportamientos de los electrodomésticos de la casa, además, busca ahorrar recursos como el agua y la electricidad. El sistema contará con un centro de monitorización, notificación, trazabilidad, administración y seguridad en base a roles el cual permitirá al usuario “Administrador” ser consciente de todos los procesos que suceden en el sistema SmartHouse. Esta jerarquía fue elegida para que el control del sistema sea limitado según los permisos otorgados a cada rol.

Actores: Administrador, usuario estándar, invitado, sistema, electrodomésticos (cámaras, luces, lavarropas, aire acondicionado)

REQUERIMIENTOS FUNCIONALES:	REQUERIMIENTOS NO FUNCIONALES:
Agregar dispositivo	Interfaz responsiva
Modificar dispositivo	GUI personalizable
Buscar dispositivo	Diseño amigable para el usuario
Eliminar dispositivo	Soporte para múltiples idiomas
Listar dispositivos	Encriptación de datos
Agregar automatizaciones	Control de acceso por roles
Modificar automatizaciones	Autenticación multifactorial
Buscar automatizaciones	Logs de errores
Eliminar automatizaciones	Crear back-ups diarios
Listar automatizaciones	Alertas ante fallos críticos
Asignar roles	Compatibilidad con AI de voz
Notificar cambios de estado automatizaciones	Tiempo de respuesta menor a 10 segundos
Log-In	Soporte hasta 10 usuarios simultáneos
Registrar usuarios	Generación de estadísticas de uso

MVP DEMO: Registro de usuario, Log-In de usuario, Dispositivos (agregar, modificar, eliminar, buscar, listar), Automatizaciones (agregar, modificar, eliminar, buscar, listar), validaciones de prevención de datos no aceptados o errores críticos. Este MVP es puramente funcional y en consola.

SmartHouse: Sistema de automatización de dispositivos inteligentes desarrollado en Python + SQL como parte del ABP de la Tecnicatura en Desarrollo de Software.

Descripción: El sistema SmartHouse permite gestionar dispositivos, usuarios y automatizaciones dentro de un entorno de simulación. Incluye control de roles y seguridad básica, validaciones de entrada y operaciones CRUD completas.

Tecnologías utilizadas

- Python 3.10.
- SQL Server Management 20.
- PyODBC (Conector Python-SQL Server).
- Bcrypt (Hash de contraseñas).

Estructura del Proyecto

- Main.py (Punto de entrada/ejecución del programa principal).
- Interfaz.py (Manejo y navegación entre menús).
- Gestor.py (Procesos CRUD del programa de dispositivos, automatizaciones y roles de usuarios).
- Validaciones.py (Validaciones de datos de entrada, normalización, comparativa y control de usuarios).
- Imprimir.py (Funciones de impresión de diferentes listados).
- Conexión_base_de_datos.py (Conexión a la base de datos SQL).
- SmartHouse.sql (Script de creación y estructura de la base de datos).
- Carpeta Docs (Documentación y diagramas del sistema).

Instalación y configuración del sistema

1. Clonar el repositorio:
```bash  
git clone https://github.com/usuario/SmartHouse.git  
cd SmartHouse
2. Instalar dependencias: `pip install pyodbc bcrypt`
3. Configurar la base de datos:  
Crear la base de datos en SQL Server con el script `SmartHouse.sql`  
Revisar `conexión_base_de_datos.py` y configurar con su usuario y contraseña.
4. Ejecutar el programa Python `main.py`

## Roles de usuarios

ADMIN: Acceso total (gestionar usuarios, dispositivos y automatizaciones).

USUARIO: Puede ver, crear y modificar dispositivos y automatizaciones.

INVITADO: Solo puede ver y consultar dispositivos y automatizaciones.

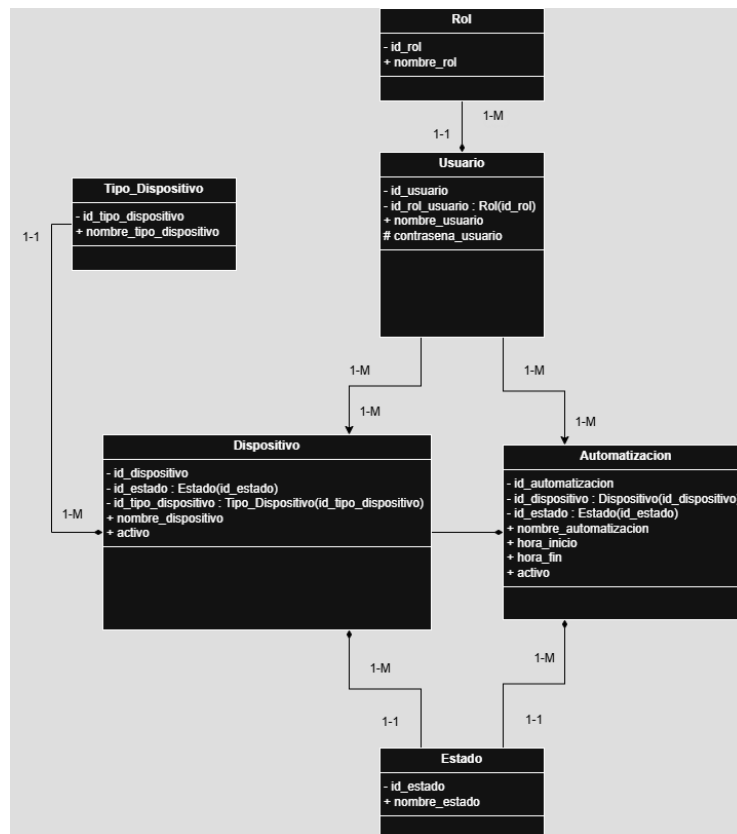
## Funcionalidades

- Registro y login de usuarios (con contraseñas encriptadas).
- Gestión de dispositivos (alta, baja, modificación, listado y búsqueda).
- Gestión de automatizaciones (programar horarios y estado de dispositivos).
- Gestión de roles de usuario (Solo puede hacerlo el usuario con rol ADMIN).

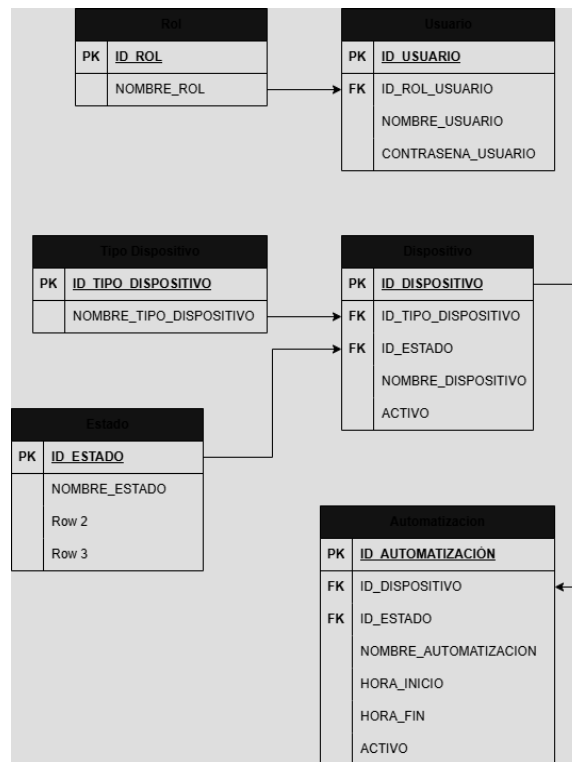
## Diagramas

(En docs pueden acceder a los diagramas en formato .drawio, .PNG y .XML).

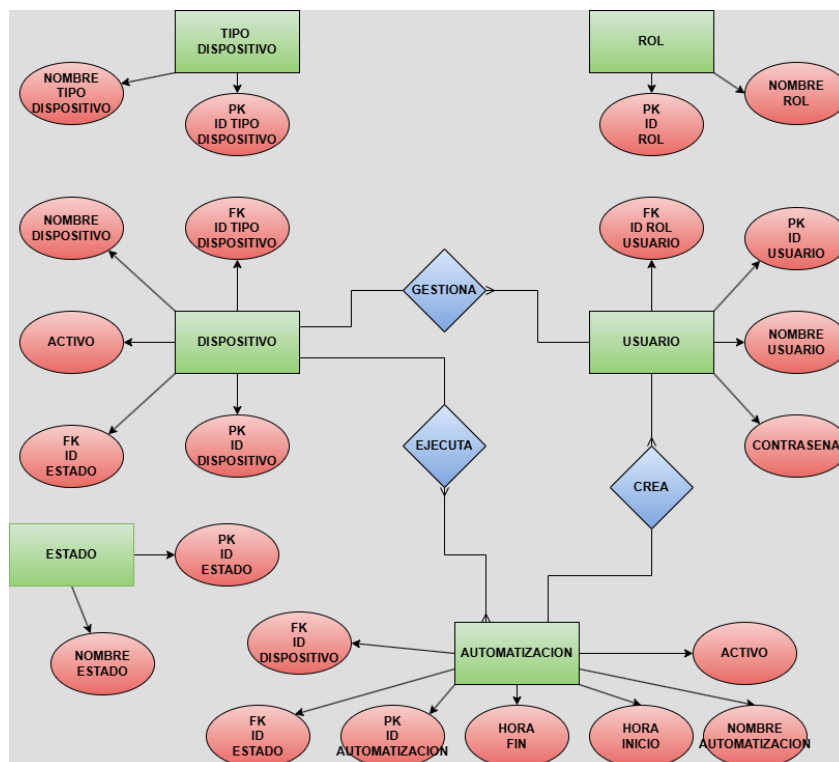
- Diagrama de clases:



- Diagrama Relacional:



- Diagrama Entidad Relación (DER):



## Próximos pasos

- Implementar monitoreo automático de eventos (sensores/cámaras).
- Agregar interfaz gráfica básica (Tkinter o web).
- Generar reportes de uso (KPI's).