# YOUR PROJECT TITLE

#### Video Demo: <URL HERE>

#### Description:

Home Care in Mexico

This is a web base application to communicate between a patient, the nurse and the nurse´s supervisor in home care. It is designed to work in Mexico, because of the gap between homecare workers and the need of them.

## Enfoque

El foco de este Sistema es la comunicación. Es importante compartir datos e información sobre situaciones que pueden incidir la salud de un paciente.

Es deseable que el paciente comparta los datos relaticos a síntomas, eventos, resultados de estudios y tomas de somatometría.

Por su parte, la enfermera debe registrar el plan de cuidados, incluyendo las indicaciones médicas, y podrá registrar también prescripciones y somatometría. También deberá incluir en el Plan las acciones de enfermería (curaciones, detecciones, colocación de sondas, toma de signos vitales, aplicación de medicamentos por vía prescrita, …)

Por su parte el supervisor revisará y aprobará los planes de cuidados, teniendo acceso a todo el expediente y comunicación directa con la enfermera.

Los alcances y posibilidades de desarrollo a partir de este esquema son muchas y se consideran fuera del alcance de este proyecto.

Tan solo para citar algunos:

* del lado del paciente, la comunicación con “gadgets” de salud pueden alimentar el expediente de cuidados, mismo que puede consultarse por el o los médicos y personal de confianza que cada paciente decida.
* También se podría usar inteligencia artificial con los síntomas y eventos y, en caso de algun riesgo, alertar a la enfermera y al supervisor.
* Del lado de la enfermera, se puede llevar el Plan de Cuidados a hojas diarias de tareas, llevar el registro de actividades cumplidas y alertar de tareas pendientes tanto a enfermería como a supervisión. También se puede apoyar a la enfermera mediante acceso a marco legal aplicable y procedimientos específicos.
* Las prescripciones se pueden programar en el calendario y pedir confirmación. Para los medicamentos relativos a diagnósticos crónicos, se puede registrar stock y consumo para alertar con la anticipación deseada sobre la necesidad de resurtimiento.
* Para el rol de supervisión se pueden hacer análisis tanto del estado de salud del paciente, como de la eficiencia y eficacia de cada enfermera.

Por lo pronto, la funcionalidad por rol de esta app queda así:

#Rol de paciente.

El paciente podrá registrar eventos que pueden impactar su salud (sucesos o hechos observables, como fiebre, sangrado, caída) y síntomas (manifestación subjetiva no cuantificable como dolor, ansiedad, mareo). También podrá autorizar a otros usuarios a ver su expediente (y revocar la autorización).

También le será visible el Plan de Cuidados y las prescripciones.

#Rol de enfermeria

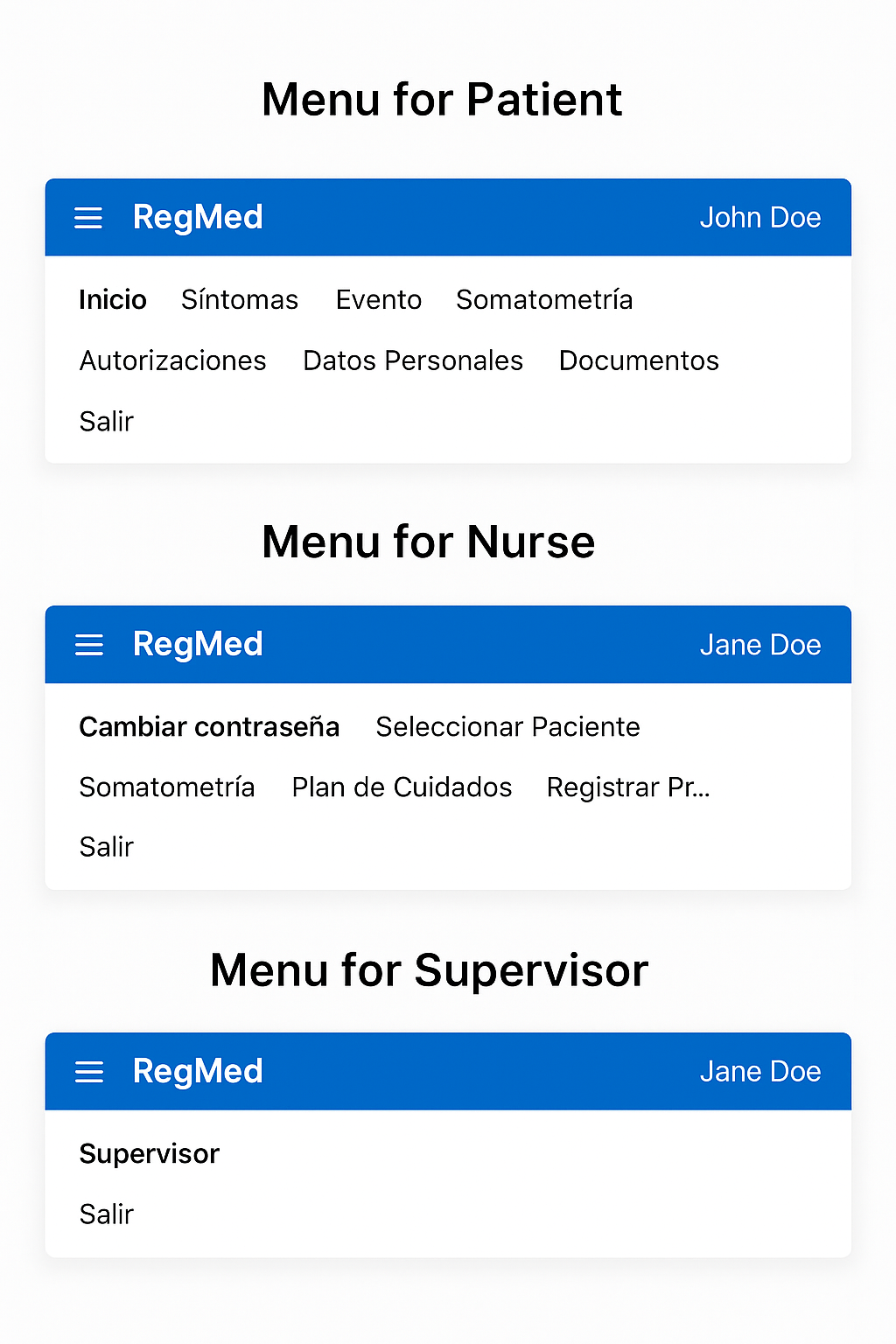
La enfermera podrá registrar nuevo Plan de Cuidados cuando sea requerido. Podrá ver los expedientes de todos los pacientes. También podrá registrar la somatometría del paciente.

Podrá consultar los eventos y síntomas registrados por el paciente.

Podrá ver en el Plan de Cuidados los comentarios del supervisor, en caso de que lo rechace. Podrá hacer los ajustes y enviar comentarios al supervisor hasta que sea aceptado.

#Rol de supervisor

El supervisor tendrá a la vista los Planes de cuidados pendientes de aprobación. Tendrá a la vista y podrá consultar el resto de la información. Puede aceptar o rechazar los planes. En este caso se abre un campo para presentárselo a enfermería y explicar el motivo del rechazo.



#ARCHIVOS DEL PROYECTO

##TEMPLATES

###apology.html

Mensajes de error

###autacceso.html

Autorización, por parte del paciente, para que otra persona acceda al expediente de cuidados.

###datospersonales.html

Permite al paciente registrar sus datos personales.

###documentos.html

Permite al paciente subir al expediente documentos en formato PDF relevantes para el cuidado, tales como resultados de estudios auxiliares.

###evento.html

Permite al paciente registrar eventos importantes como enfermedades, eventos traumáticos, tratamientos médicos, ambientales, sociales,...

###index.html

###layout.html

Plantilla base que adapta la interfaz según el usuario que está logueado, mostrando un menú diferente para enfermería, pacientes y supervisores.

###login.html

Rutina para verificar password y buscar rol del usuario

###macros.html

Macros para resaltar los riesgos de un paciente (alegrías, úlceras,…)

###password.html

Rutina para que el usuario cambie su password

###prescripciones.html

Permite a enfermera registrar la o las prescripciones médicas, para darles seguimiento

###register.html

Pantalla de inicio de sesión, que pide usuario y contraseña

###registro\_enfermeria.html

Esta es la pantalla de trabajo de la enfermera. Permite visualizar el Plan de Cuidados, así como registrar prescripciones, somatometría y propuestas de ajustes al Plan de Cuidados.

###seleccionar\_paciente.html

Permite a la enfermera seleccionar con que paciente va a trabajar

###sintomas.html

Permite al paciente registrar síntomas.

###somatometria.html

Permte a enfermera y a paciente registrar síntomas.

##PYTHON

###App.py

1. **Configuración de la aplicación Flask**
   * + Inicializa Flask y configura la sesión.
     + Configura carpeta de uploads y filtros personalizados para Jinja.
2. **Conexión a la base de datos**
   * + Función [get\_db\_connection()](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) para conectar a SQLite y crear tablas si no existen.
3. **Rutas principales (endpoints)**
   * /
     + Métodos: GET, POST
     + Función: [index](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Muestra el dashboard principal con datos del usuario, paciente, planes de cuidados, somatometría, síntomas, eventos, documentos, autorizaciones y prescripciones.
   * /datospersonales
     + Métodos: GET, POST
     + Función: [datospersonales](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Permite al paciente registrar o actualizar sus datos personales.
   * /autacceso
     + Métodos: GET, POST
     + Función: [autacceso](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Permite registrar y mostrar autorizaciones de acceso.
   * /editar\_ultimo\_enfermeria
     + Método: POST
     + Permite a enfermería editar el último plan de cuidados rechazado.
   * /login
     + Métodos: GET, POST
     + Función: [login](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Permite iniciar sesión.
   * /password
     + Métodos: GET, POST
     + Función: [password](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Permite cambiar la contraseña del usuario.
   * /register
     + Métodos: GET, POST
     + Función: [register](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Permite registrar un nuevo usuario y, si es paciente, lo redirige a completar sus datos personales.
   * /registro\_enfermeria
     + Métodos: GET, POST
     + Función: [registro\_enfermeria](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Permite a enfermería registrar un nuevo plan de cuidados para un paciente seleccionado.
   * /seleccionar\_paciente
     + Métodos: GET, POST
     + Función: [seleccionar\_paciente](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Permite a enfermería seleccionar el paciente sobre el que va a trabajar.
   * /sintomas
     + Métodos: GET, POST
     + Función: [sintomas](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Permite al paciente registrar y ver sus síntomas.
   * /evento
     + Métodos: GET, POST
     + Función: [evento](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Permite al paciente registrar eventos y autorizaciones.
   * /somatometria
     + Métodos: GET, POST
     + Función: [somatometria](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Permite registrar y mostrar datos de somatometría (peso, talla, presión, etc.) tanto por paciente como por enfermería.
   * /documentos
     + Métodos: GET, POST
     + Función: [documentos](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Permite al paciente subir y ver documentos.
   * /prescripciones
     + Métodos: GET, POST
     + Función: [prescripciones](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Permite registrar y mostrar prescripciones médicas.
   * /logout
     + Método: GET
     + Función: [logout](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Cierra la sesión del usuario.
   * /validar\_ultimo
     + Método: POST
     + Permite validar, comentar o cancelar el último plan de cuidados.
4. **Funciones auxiliares**
   * [obtener\_planes\_de\_cuidados(paciente\_id)](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Obtiene los planes de cuidados de un paciente.
   * [registrar\_usuario(username, password, rol, nombre\_completo)](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)
     + Registra un usuario y, si es paciente, lo agrega a la tabla de pacientes.
5. **Gestión de sesiones y contexto**
   * Inyecta el nombre de usuario en los templates.
   * Configura cabeceras para evitar caché.

###Helpers.py

1. **valida(usuario, db)**
   * Verifica si el usuario existe en la base de datos.
2. **apology(message, code=400)**
   * Muestra una página de error personalizada ([apology.html](vscode-file://vscode-app/c:/Users/Mauricio/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)) con el mensaje y código de error.
3. **login\_required(f)**
   * Decorador para proteger rutas; obliga a que el usuario esté logueado para acceder.
4. **mxn(value)**
   * Formatea un valor numérico como moneda en pesos mexicanos (MXN).
5. **rows\_to\_dict(rows)**
   * Convierte una lista de filas de SQLite (sqlite3.Row) en una lista

##BASE DE DATOS

pacientes.db has the next .SCHEMA

CREATE TABLE sqlite\_sequence(name,seq);

CREATE TABLE resumen (id INTEGER PRIMARY KEY AUTOINCREMENT, users\_id INTEGER NOT NULL, symbol TEXT NOT NULL, qty INTEGER NOT NULL);

CREATE TABLE Pacientes ( usersId INTEGER PRIMARY KEY AUTOINCREMENT, Usuario TEXT, Contrasena TEXT, NombreCompleto TEXT, Expediente TEXT, Nombre TEXT, PrimerApellido TEXT, SegundoApellido TEXT, FechaNacimiento TEXT, -- YYYY-MM-DD EntidadDeNacimiento INTEGER, SexoCURP INTEGER, SexoBiologico INTEGER, Genero REAL, CURP TEXT, EstadoMental TEXT, GrupoRH TEXT, -- cambié "Grupo y RH" (no admite espacios en nombre) DiagnosticoPrimario TEXT, -- sin espacios CIEPrimario TEXT, -- char(4) → TEXT SegundoDiagnostico TEXT, SegundoCIE TEXT, TercerDiagnostico TEXT, TercerCIE TEXT, Domicilio INTEGER, eMailPaciente TEXT, TelefonoPaciente TEXT, WhatsAppPaciente TEXT, NombreContactoTutor TEXT, ParentescoContactoTutor TEXT, TelefonoContactoTutor TEXT, WhatsAppContactoTutor TEXT, PrimerContacto INTEGER DEFAULT 1, PacienteActivo INTEGER NOT NULL, FechaUltimoMovimiento TEXT -- YYYY-MM-DD

, Alergico INTEGER DEFAULT 0, RiesgoCaidas INTEGER DEFAULT 0, RiesgoUlceras INTEGER DEFAULT 0);

CREATE TABLE movimiento ( id INTEGER PRIMARY KEY AUTOINCREMENT, date NOT NULL, users\_id INTEGER NOT NULL, evento TEXT NOT NULL, type INTEGER NOT NULL );

CREATE TABLE sintomas ( id INTEGER PRIMARY KEY AUTOINCREMENT, date TEXT NOT NULL, users\_id INTEGER NOT NULL, tipo TEXT NOT NULL, duracion TEXT, intensidad TEXT, descripcion TEXT );

CREATE TABLE users (id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, username TEXT NOT NULL, hash TEXT NOT NULL, rol TEXT NOT NULL DEFAULT 'paciente');

CREATE TABLE eventos ( id INTEGER PRIMARY KEY AUTOINCREMENT, date TEXT NOT NULL, users\_id INTEGER NOT NULL, tipo INTEGER NOT NULL, evento TEXT NOT NULL, type INTEGER NOT NULL );

CREATE TABLE documentos ( id INTEGER PRIMARY KEY AUTOINCREMENT, users\_id INTEGER NOT NULL, fecha TEXT NOT NULL, tema TEXT NOT NULL, comentarios TEXT, filename TEXT NOT NULL );

CREATE TABLE IF NOT EXISTS "planes\_de\_cuidados" ( id INTEGER PRIMARY KEY AUTOINCREMENT, users\_id INTEGER NOT NULL, fecha TEXT NOT NULL, riesgo\_caidas TEXT, estado\_mental TEXT, riesgo\_ulceras TEXT, riesgo\_pie\_diabetico TEXT, heridas TEXT, estomas TEXT, aseo TEXT, medidas\_posturales TEXT, balance\_liquidos TEXT, dispositivos TEXT, cuidados\_via\_aerea TEXT , status TEXT NOT NULL DEFAULT 'Pendiente de revisión', comentarios TEXT, dieta TEXT, rehabilitacion TEXT, alergico TEXT, detecciones TEXT, acciones TEXT);

CREATE TABLE prescripciones ( id INTEGER PRIMARY KEY AUTOINCREMENT, users\_id INTEGER NOT NULL, medicamento TEXT NOT NULL, dosis TEXT NOT NULL, cantidad TEXT NOT NULL, via TEXT NOT NULL, cada\_cantidad TEXT NOT NULL, cada\_unidad TEXT NOT NULL, unidad\_medida TEXT NOT NULL, desde TEXT NOT NULL, durante\_cantidad TEXT NOT NULL, durante\_unidad TEXT NOT NULL , frecuencia TEXT, observaciones TEXT, vigente TEXT);

CREATE TABLE somatometria ( id INTEGER PRIMARY KEY AUTOINCREMENT, fecha TEXT, peso REAL, talla REAL, imc REAL, circ\_abdominal REAL, temp REAL, sistolica INTEGER, diastolica INTEGER, fcard INTEGER, fresp INTEGER, o2 INTEGER, glucemia REAL, users\_id INTEGER ,registrado\_por INTEGER);

##REQUIREMENTS

Requirements.txt has the requirements for this project:

* Flask
* Flask-Session
* pytz
* SQLAlchemy

##README

This is the readme.md

#Rutas y roles

| **Ruta** | **Métodos** | **Función** | **Rol requerido / Acceso** |
| --- | --- | --- | --- |
| / | GET, POST | index | **Todos** (según rol: supervisor, enfermería o paciente) |
| /datospersonales | GET, POST | datospersonales | **Paciente** |
| /autacceso | GET, POST | autacceso | **Paciente** |
| /editar\_ultimo\_enfermeria | POST | editar\_ultimo\_enfermeria | **Enfermería** |
| /login | GET, POST | login | Público (sin sesión) |
| /register | GET, POST | register | Público (sin sesión) |
| /registro\_enfermeria | GET, POST | registro\_enfermeria | **Enfermería** |
| /seleccionar\_paciente | GET, POST | seleccionar\_paciente | **Enfermería** |
| /sintomas | GET, POST | sintomas | **Paciente** |
| /evento | GET, POST | evento | **Paciente** |
| /somatometria | GET, POST | somatometria | **Paciente y Enfermería** |
| /documentos | GET, POST | documentos | **Paciente** |
| /prescripciones | GET, POST | prescripciones | **Paciente y Enfermería** |
| /logout | GET | logout | **Todos con sesión** |
| /validar\_ultimo | POST | validar\_ultimo | **Supervisor** |