**Sistema de Gestión de Salud Comunitaria**

El presente proyecto tiene como objetivo desarrollar un sistema integral de gestión de salud comunitaria que facilite el acceso a servicios médicos y optimice la administración de información relacionada con la salud en comunidades locales. La problemática actual se centra en la dificultad que enfrentan muchos ciudadanos para acceder a servicios médicos adecuados, así como la falta de un sistema organizado que permita gestionar citas médicas, encuestas sobre salud e informes médicos.

Este sistema busca resolver estas dificultades al ofrecer una plataforma accesible donde los pacientes pueden registrarse, gestionar sus citas y acceder a su historial médico. Los doctores podrán administrar sus consultas y generar informes sobre el estado de salud de sus pacientes. Además, un panel administrativo permitirá gestionar usuarios y roles dentro del sistema, garantizando así un control adecuado sobre quién tiene acceso a qué información.

**Funcionalidades Principales**

1. Registro de Pacientes:

* Permitir a los usuarios registrarse y actualizar su información personal y médica.
* Mantener un historial médico accesible para los profesionales de la salud.

2. Gestión de Citas Médicas:

* Los pacientes pueden solicitar citas con médicos disponibles.
* Notificaciones automáticas sobre próximas citas.

3. Seguimiento del Estado de Salud:

* Registro de síntomas y condiciones médicas.
* Monitoreo del estado general de salud a través de encuestas periódicas.

4. Generación de Informes en PDF:

* Opción para generar informes que resuman el estado de salud individual o colectivo.
* Informes que pueden ser utilizados por las autoridades sanitarias para evaluar la situación en la comunidad.

Requerimientos Funcionales

1. Registro de Pacientes:

* El sistema debe permitir a los usuarios registrarse proporcionando su nombre, edad, género y una historia médica.
* Los usuarios deben poder cargar documentos como cédulas y fotografías durante el registro.

2. Gestión de Citas Médicas:

* Los pacientes deben poder solicitar citas médicas a través del sistema.
* El sistema debe mostrar una lista de citas disponibles y permitir a los pacientes seleccionar una fecha y hora.

3. Monitoreo del Estado de Salud:

* El sistema debe permitir la creación de encuestas periódicas para evaluar el estado de salud de los pacientes.
* Los pacientes deben poder reportar síntomas o condiciones médicas a través del sistema.

4. Generación de Informes en PDF:

* El sistema debe permitir a los profesionales generar informes en PDF que resuman el estado de salud individual o colectivo.
* Los informes deben incluir estadísticas sobre enfermedades prevalentes, tasas de vacunación, etc.

5. Almacenamiento y Gestión de Archivos:

* El sistema debe permitir la carga y almacenamiento seguro de documentos e imágenes (cédulas, fotos).
* Los archivos deben ser accesibles solo para personal autorizado y los propios pacientes.

6. Notificaciones:

* El sistema debe enviar notificaciones automáticas a los pacientes sobre próximas citas médicas.
* Debe haber alertas para recordar a los pacientes sobre encuestas o chequeos médicos.

7. Interfaz de Usuario:

* La interfaz debe ser intuitiva y fácil de usar, permitiendo navegación sencilla entre las diferentes funcionalidades del sistema.
* Debe haber formularios claros para el registro, gestión de citas y carga de archivos.

Requerimientos No Funcionales

1. Usabilidad:

* La interfaz del sistema debe ser amigable y accesible para usuarios con diferentes niveles de habilidad tecnológica.
* Debe incluir instrucciones claras y mensajes de error comprensibles.

2. Rendimiento:

* El sistema debe ser capaz de manejar al menos 1000 usuarios simultáneamente sin degradar su rendimiento.
* Las operaciones (carga de archivos, generación de informes) deben completarse en menos de 5 segundos en condiciones normales.

3. Escalabilidad:

* El sistema debe ser escalable para acomodar un aumento en el número de usuarios y datos sin necesidad de reestructuración significativa.
* Debe soportar la adición futura de nuevas funcionalidades sin afectar el rendimiento existente.

4. Seguridad:

* Todos los datos sensibles (información personal, historial médico) deben ser almacenados encriptados en la base de datos.
* El acceso a archivos cargados debe estar restringido a usuarios autorizados mediante autenticación robusta.

5. Mantenibilidad:

* El código del sistema debe seguir buenas prácticas para facilitar futuras actualizaciones y mantenimiento.
* La documentación técnica debe estar disponible para desarrolladores futuros que trabajen en el sistema.

6. Compatibilidad:

* El sistema debe ser compatible con navegadores web comunes (Chrome, Firefox, Safari).
* Debe ser accesible desde dispositivos móviles para facilitar el acceso a los usuarios en cualquier lugar.

7. Respaldo y Recuperación:

* Debe existir un mecanismo automático para respaldar la base de datos regularmente.
* En caso de fallo del sistema, debe haber un plan claro para la recuperación rápida sin pérdida significativa de datos.

**Viabilidad del Proyecto**

El proyecto de gestión de salud comunitaria es viable debido a:

* Necesidad Real: La demanda de sistemas que faciliten el acceso a servicios de salud es alta, especialmente en comunidades con limitaciones en infraestructura médica.
* Beneficio Social: Mejorar el acceso a la atención médica y la gestión de información de salud puede tener un impacto significativo en la calidad de vida de los ciudadanos.
* Escalabilidad: El diseño modular permite agregar nuevas funcionalidades y adaptarse a futuras necesidades sin comprometer el rendimiento.
* Tecnología Accesible: Las tecnologías web y bases de datos utilizadas son ampliamente soportadas y se pueden implementar fácilmente.

**Entidades para la Base de Datos**

Las entidades clave involucradas en este proyecto incluyen:

* User: Representa a todos los usuarios del sistema (pacientes, doctores y administradores).
* Role: Define los diferentes roles disponibles dentro del sistema.
* Profile: Relaciona cada usuario con su rol correspondiente, permitiendo una gestión adecuada de permisos.
* Paciente: Almacena información específica sobre cada paciente, incluyendo su historia médica y documentos relevantes.
* Cita, Encuesta, e Informe: Representan las interacciones clave entre pacientes y servicios médicos.

Entidades Principales

1. User

* id\_user: INT, PK
* name: VARCHAR
* email: VARCHAR
* status: ENUM ('activo', 'inactivo', 'suspendido')
* registration\_date: DATETIME
* withdrawal\_date: DATETIME

1. Role

* id\_role: INT, PK
* name\_role: VARCHAR

1. Profile

* id\_profile: INT, PK
* user\_id: INT, FK (relación con User)
* role\_id: INT, FK (relación con Role)
* estatus\_profile: ENUM ('activo', 'inactivo', 'suspendido')

1. Patient

* id\_patient: INT, PK
* user\_id: INT, FK (relación con User)
* medical\_history: TEXT
* citizenship \_url: VARCHAR
* photo\_url: VARCHAR

1. Citation

* id\_citation: INT, PK
* patient\_id: INT, FK (relación con Paciente)
* date: DATETIME
* doctor\_id: INT, FK (relación con User)
* document\_url: VARCHAR

1. Poll

* id\_poll: INT, PK
* patient\_id: INT, FK (relación con Paciente)
* date: DATETIME
* result: TEXT

1. Report

* id\_report: INT, PK
* patient\_id: INT, FK (relación con Paciente)
* generation\_date: DATETIME
* content: TEXT
* pdf\_url: VARCHAR

Entidades de Seguridad de Datos

1. AccessLog (Para registrar accesos al sistema y cambios en los estados de los usuarios.)

* id\_log: INT, PK
* user\_id: INT, FK (relación con User)
* action\_performed: VARCHAR
* date\_time: DATETIME

**Vistas y Apartados Necesarios**

Receptores del Servicio (Pacientes y Doctores)

1. Registro: Formulario para crear cuentas (pacientes y doctores).
2. Inicio de Sesión: Página para que los usuarios ingresen al sistema.
3. Gestión de Citas: Visualización y solicitud de citas médicas.
4. Encuestas: Formulario para completar encuestas sobre el estado de salud.
5. Informes: Opción para descargar informes en PDF sobre su estado de salud.

Emisores del Servicio (Administradores)

1. Panel Administrativo: Interfaz para gestionar usuarios y roles.
2. Visualización de Citas: Listado y gestión de todas las citas programadas.
3. Generación de Informes: Herramienta para generar informes sobre la salud pública en la comunidad.
4. Gestión de Encuestas: Visualización y análisis de resultados de encuestas completadas por pacientes.

Controlador: Flujo de Datos e Interacción

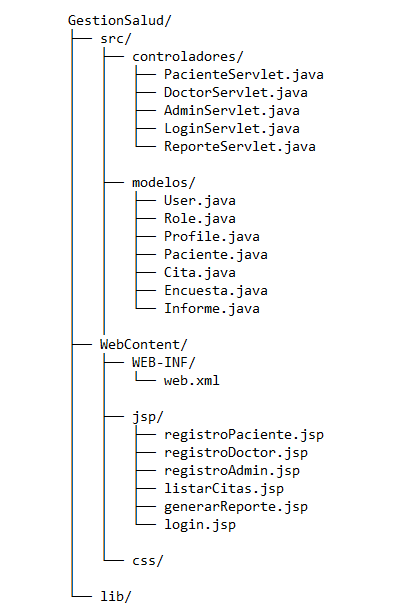
1. Flujo General

* Registro e Inicio de Sesión: Los usuarios acceden a formularios específicos para registrarse o iniciar sesión.
* Subida de Archivos: Durante el registro o en otros procesos, los usuarios pueden subir documentos e imágenes que se almacenan en el servidor o en una base de datos.
* Gestión y Consulta: Los pacientes pueden gestionar sus citas y completar encuestas a través del sistema.
* Generación y Descarga de Informes: Los pacientes pueden solicitar informes sobre su estado médico, que se generan en formato PDF.
* Administración: Los administradores gestionan usuarios y roles desde un panel administrativo, asegurando que los permisos sean correctos según el rol asignado.

1. Estabilidad del Proyecto

* Manejo de Errores: Implementar manejo adecuado de excepciones en los servlets para asegurar que cualquier error sea gestionado sin afectar la experiencia del usuario.
* Seguridad: Proteger las rutas sensibles y asegurar que solo los usuarios autorizados puedan acceder a ciertas funcionalidades mediante autenticación robusta.
* Respaldo Regular: Implementar un sistema automático para respaldar la base de datos regularmente para evitar pérdida de datos.

**Estructura en NetBeans**



1. Controladores (src/controladores/)

* PacienteServlet.java: Maneja el registro de pacientes, incluyendo la validación de datos y la subida de archivos (cédula y foto). Cumple con los requerimientos de gestión de pacientes y almacenamiento seguro de documentos.
* DoctorServlet.java: Permite el registro de doctores, gestionando su información y fotos. Asegura que los doctores tengan acceso a funciones específicas del sistema.
* AdminServlet.java: Facilita la gestión de usuarios y roles desde el panel administrativo, asegurando que se cumplan las políticas de seguridad y permisos.
* LoginServlet.java: Implementa la lógica de autenticación, asegurando que solo los usuarios autorizados accedan al sistema. Esto es crucial para la seguridad del sistema.
* ReporteServlet.java: Genera informes en PDF sobre el estado de salud, permitiendo a los pacientes descargar información relevante. Esto cumple con el requerimiento de generación de informes.

2. Modelos (src/modelos/)

* User.java: Define la entidad usuario, incluyendo atributos como nombre, email y estado. Es fundamental para gestionar la información básica de todos los usuarios.
* Role.java: Define los roles disponibles en el sistema (Paciente, Doctor, Administrador), permitiendo una gestión adecuada de permisos.
* Profile.java: Relaciona usuarios con roles y gestiona el estado del perfil, asegurando que se cumplan las políticas de acceso.
* Paciente.java: Almacena información específica sobre pacientes, como su historia médica y documentos relevantes. Es esencial para la atención médica efectiva.
* Cita.java, Encuesta.java, Informe.java: Estas clases representan las entidades relacionadas con citas médicas, encuestas sobre salud e informes generados, respectivamente. Son críticas para el seguimiento y gestión del estado de salud.

3. Vistas (WebContent/jsp/)

* registroPaciente.jsp, registroDoctor.jsp, registroAdmin.jsp: Proporcionan formularios para que los usuarios se registren en el sistema. Incluyen campos necesarios para cumplir con los requisitos funcionales.
* listarCitas.jsp: Permite a los pacientes ver sus citas programadas y gestionar cambios. Facilita la interacción entre pacientes y servicios médicos.
* generarReporte.jsp: Ofrece la funcionalidad para generar e imprimir informes en PDF sobre el estado de salud del paciente.
* login.jsp: Proporciona un formulario para que los usuarios inicien sesión en el sistema, asegurando que solo los autorizados puedan acceder.

4. Configuración (WebContent/WEB-INF/web.xml)

* Configura las rutas del servlet y las restricciones necesarias para garantizar que solo los usuarios autenticados accedan a ciertas funcionalidades.

5. CSS (WebContent/css/)

* Archivos CSS que mejoran la usabilidad y apariencia del sistema, cumpliendo con requisitos no funcionales relacionados con la experiencia del usuario.