Sistema de Gestión de Diagnósticos Asistidos por Imágenes Médicas

Descripción del Proyecto

El proyecto consiste en desarrollar una aplicación para gestionar los resultados de diagnósticos médicos realizados mediante análisis de imágenes de resonancia magnética (MRI), tomografía computarizada (CT) y rayos X. Estos estudios están vinculados a pacientes con enfermedades crónicas como cáncer y enfermedades cardíacas. El sistema permitirá almacenar, visualizar y administrar tanto los datos de los pacientes como los diagnósticos generados por un modelo de IA. Este sistema tendrá una base de datos SQL para la gestión estructurada de pacientes y sus diagnósticos, y una base de datos MongoDB para almacenar imágenes y reportes no estructurados. Igualmente se gestiona la información de los usuarios (administrador, medico, técnico)

Para este proyecto , deben investigar un poco sobre imágenes biomédicas y su manejo, y qué son los metadatos, a modo informativo para para que estén contextualizados y entiendan que tipo de metadatos van a almacenar en la base de datos NoSql (MONGODB)

Funcionalidades Requeridas

1. **Login y roles de Usuario:**
    a) Login con autenticación de roles (Administrador, Médico, Técnico).
    b) Solo el administrador podrá añadir, eliminar o modificar usuarios y permisos (CRUD) , estos datos, en base de datos SQL.
    c) Los médicos podrán ver y actualizar diagnósticos e información de los pacientes, pero no pueden borrarla ni crearla, ni gestionar usuarios, ni cargar imágenes, mientras que los técnicos solo podrán gestionar las imágenes(sus metadatos) y reportes, es decir que los técnicos pueden cargar imágenes, moverlas, eliminar, añadir notas técnicas a dichos reportes, pero no tienen acceso a la información de pacientes o diagnósticos.
2. **Gestión de Pacientes y Diagnósticos:**
    a) Crear, leer, actualizar y eliminar registros de pacientes según los especificado por los diferentes roles.

b) Registrar, actualizar y ver los resultados de diagnósticos por cada paciente. Los diagnósticos incluyen:

- Resultado de probabilidad de enfermedad generada por IA (en %).
- Tipo de imagen (MRI, CT, Rayos X).
- Fecha de la toma de imagen y fecha del diagnóstico.
- Estado de revisión por el médico (Sí/No).

3. **Almacenamiento y "Visualización" de Imágenes:**

a) Las imágenes médicas serán almacenadas en la base de datos NoSQL (MongoDB) junto con metadatos, como:

- Identificación del paciente.
- Tipo y fecha de la imagen.
- Resultado preliminar del análisis por IA (en %).
- Información de la imagen diagnóstica (nota del técnico relacionada con el tipo de captura; contraste, posicionamiento del paciente, resolución espacial, frecuencia de muestreo, etc..., consultar al respecto)
- Zona de estudio de la imagen (Abdomen, cabeza, extremidades, etc.)

b) Los técnicos pueden ver y actualizar el estado de revisión de cada imagen.

4. **Reportes Médicos:**

a) Los médicos podrán generar reportes personalizados con el estado de los pacientes y adjuntar notas de diagnóstico.

b) Los reportes se almacenarán MongoDB en un formato de documento estructurado y estarán relacionados con el paciente.

Ejemplo:

```
{
  "id_imagen": "img456",
  "id_paciente": "12345",
  "fecha": "2024-05-14",
  "tipo_imagen": "MRI",
  "parte_cuerpo": "Cerebro",
  "image_path": "/images/mri/2024/05/14/img456_mri_brain.jpg",
  "analisis_IA": {
    " condicion_sugerida": "Glioblastoma multiforme",
    "probabilidad_%": 92.3,
    "notas": "La masa tiene bordes irregulares y alta densidad en las áreas observadas."
  },
```

```
"notas_tecnicas": [  {
    "id_tecnica": "tech789",
    "fecha_nota ": "2024-05-14",
    "texto": "Imagen capturada correctamente con contraste. No se reportaron problemas en la
   captura."
  }  ]
}
```


5. **Submenú de búsqueda:**

    a) Permite buscar pacientes por su ID, ver historial de diagnósticos, y consultar imágenes asociadas.

6. **CRUD para Usuarios, Pacientes, Diagnósticos e Imágenes:**

    a) Las funcionalidades CRUD deben realizarse tanto en SQL como en MongoDB, asegurando la sincronización entre las bases de datos.

Requerimientos Técnicos

1. **Bases de Datos Relacionales (SQL) y NoSQL (MongoDB):**

    a) La base de datos SQL (MySQL) contendrá:

        - Tabla de usuarios (administrador, médico, técnico), user_id, username, password, role.
        - Tabla de pacientes (ID, nombre, edad, género, historial de diagnósticos).
        - Tabla de diagnósticos (ID de paciente, tipo de imagen, resultado de IA, fecha de diagnóstico, estado).

    b) La base de datos MongoDB contendrá:

        - Colección de imágenes médicas(Paths de ubicaciones donde van a estar las imágenes), con referencias al ID del paciente y metadatos de imagen.
        - Colección de reportes médicos con información del diagnóstico y comentarios adicionales.

2. **Funciones Específicas:**

    a) Las funciones CRUD para cada entidad deben estar implementadas para MySQL y MongoDB.

    NOTA: Ambas bases de datos MySQL y MONGO deberá estar configurada con los siguientes parámetros:

- Nombre de la base de datos: Informatica1_PF
- Usuario: informatica1
- Contraseña: info20242
- Tablas: estarán nombradas y configuradas según lo considerado por cada equipo

b) Implementación de módulos de funciones para manejo de validación de datos (alfabéticos, numéricos, etc.) en un modulo aparte.

c) Submenús específicos para acceder y visualizar información de ambas bases de datos.

3. **Manejo de Excepciones:**

a) Validaciones de los campos de entrada y manejo de errores utilizando try/except.

b) Implementar mensajes de error personalizados y validaciones en cada campo.

4. **Documentación:**

a) El código debe estar debidamente documentado con explicaciones de funciones y lógica, accesible mediante la función help().

5. **Formato de Entrega y Sustentación:**

a) Entregar el código fuente y las bases de datos pre-cargadas en SQL y MongoDB. Es decir que las tablas ya deben contar con información previamente ingresada.

b) Sustentación del proyecto, presentación en vivo, explicando cada funcionalidad.

OBSERVACIONES DE ENTREGA

1. El trabajo se realizará en grupos de 2 a 3 INTEGRANTES
2. Deben subir el código a GitHub y adicionar link en la entrega

3. Se deben entregar todos los archivos (.py) diseñados en Spyder o Visual Studio o en Jupyter Notebook (.ipynb) y el archivo de la base de datos en MySQL y MONGO, incluirá las tablas y la colecciones de información previamente cargada para probar. Deben asegurarse de enviar correctamente los archivos ya que la revisión se realizará única y exclusivamente con los archivos cargados en la tarea de TEAMS que se abrirá para tal efecto.

4. En la cabecera del script se deben colocar los nombres de los autores como un comentario (integrantes del grupo) y el código debe estar debidamente documentado. Descripción de las variables, descripción de los procesos y resultados a mostrar.

5. El trabajo debe ser cargado en TEAMS, solo una vez, es decir, uno de los dos integrantes hará la carga . POR DEFINIR
6. Cualquier intento de fraude o copia de código entre los diferentes equipos, por mínimo que sea, una función, un ciclo, lo que sea..; implicará una calificación de 0.0 para todos los involucrados sin posibilidad de recuperación.

7. Si al ejecutar el script hay algún error, sin importar cual sea, el trabajo se calificará sobre 3.73.

Ejemplo

Funcionamiento del Login:

1. **Inicio de sesión**:
    - El sistema solicita al usuario su username y password.
    - La contraseña se compara con el password almacenado en la base de datos.
    - Si el usuario es autenticado correctamente, el sistema verifica el rol asignado y activa los permisos correspondientes.
2. **Interfaz de Menú Según Rol**:
    - **Administrador**:
        - Acceso completo, incluyendo la creación, modificación y eliminación de usuarios, y configuración de permisos.
    - **Médico**:
        - Puede ver y editar información de pacientes y diagnósticos, y acceder a reportes médicos.
    - **Técnico**:
        - Permiso para cargar imágenes médicas y añadir notas técnicas, pero sin acceso a editar diagnósticos o ver reportes médicos.

Ejemplo de Flujo del Sistema para Usuarios

1. **Login**:
    - **Usuario**: jdoe
    - **Contraseña**: hOLa53523
2. **Validación**:
    - El sistema valida el username y password.

- Encuentra que el usuario jdoe tiene el rol de **“Médico”**.

3. **Interfaz de Médico**:
    - Muestra opciones como:
        - Ver pacientes
        - Editar diagnósticos
        - Consultar reportes médicos
    - Oculta opciones de carga de imágenes y acceso administrativo, que no están permitidos para el rol de **Médico**.

Ejemplo de Flujo para el Administrador

1. **Login**:
    - **Usuario**: X
    - **Contraseña**: admin_password
2. **Validación**:
    - El sistema valida el usuario y password.
    - Encuentra que el usuario X tiene el rol de **Administrador**.
3. **Interfaz de Administrador**:
    - Opciones disponibles:
        - Gestión completa de usuarios y permisos.
        - Configuración de la base de datos y roles.
        - Acceso a todos los módulos del sistema.