Informe final de cumplimiento

Introducción

Con este informe queremos dar a conocer todas las especificaciones que incluye nuestro proyecto además de los cambios producidos respecto a los anteriores documentos entregados.
Para realizar este informe nos basamos en el nuevo documento de requisitos de software llamado FarmacIA-ERS-v1.1.

1. Requisitos del ERS que sí se implementan en la entrega

Estos son los requisitos que se implementan en la aplicación:

- Requisito 001 – Interfaz de Usuario
- Requisito 002 – RAM mínima
- Requisito 003 – Procesador mínimo
- Requisito 004 – Almacenamiento mínimo
- Requisito 005 – Interfaces de software
- Requisito 006 – Interfaz de software con IA
- Requisito 007 – Comunicación con base de datos
- Requisito 008 – Lista de medicamentos
- Requisito 009 – Sistema de introducción de medicamentos (pastillero)
- Requisito 010 – Caducidad de medicamento
- Requisito 011 – Aviso por caducidad
- Requisito 012 – Manejo y aviso de error de medicamento
- Requisito 013 – Registro de usuario
- Requisito 014 – Inicio de sesión
- Requisito 015 – Barra de búsqueda
- Requisito 016 – Autocompletar de la barra de búsqueda
- Requisito 017 – Visualizar información desde cuenta administrativa externa
- Requisito 018 – Cambiar información desde cuenta administrativa externa
- Requisito 019 – Resumen Inteligente
- Requisito 020 – Consultas con IA
- Requisito 021 – Mi Calendario
- Requisito 022 – Tiempos de reacción
- Requisito 023 – Cifrado de contraseñas
- Requisito 024 – Errores de conexión
- Requisito 025 – Errores menores
- Requisito 026 – Mantenimiento mensual
- Requisito 027 – Idioma de la aplicación

2. Requisitos del ERS que no se implementan en esta entrega o que se han modificado

Estos son los requisitos que han sido borrados de la primera versión del ERS y no se implementarán:

- Requisito (Antes era el 022) – Registros: Este requisito se ha borrado para la nueva versión ya que es una aplicación que no usará registros ni logs para registrar la actividad de los usuarios y del sistema.
  También se ha borrado todo lo relacionado con este requisito en el documento, en las acciones de otros requisitos que implementan logs, por lo que hay varios requisitos que han sufrido modificaciones menores quitando el registro.

Estos son los requisitos que han sido modificados de la primera versión del ERS:

- Requisito 008 – Lista de medicamentos: Cambio menor en la explicación de la acción del ERS.

- Requisito 013 – Registro de usuario: Cambio menor en la explicación de la acción del ERS y se elimina el requerimiento de 6 carácteres mínimo para establecer una contraseña.

- Requisito 016 – Autocompletar de la barra de búsqueda: En la nueva versión se especifica que para que se efectúe la búsqueda se necesita pulsar el botón necesario para ello, no se completa a tiempo real.

3. Requisitos nuevos que sí están implementados en esta entrega

Estos son los requisitos que han sido añadidos a la aplicación, mostrados en la segunda versión (v1.1) del documento ERS:

- Requisito 019 – Resumen Inteligente: Sistema que genere automáticamente un resumen claro y estructurado a partir de información sanitaria del usuario disponible en la app para facilitar la comprensión y consulta rápida.

- Requisito 020 – Consultas con IA: Sistema que permita al usuario realizar consultas predeterminadas a un motor de IA sobre su medicación, mediante prompts guiados, limitados a tres tipos de consulta sanitaria segura y controlada.

- Requisito 021 – Mi Calendario: Sistema que muestre al usuario un calendario semanal de medicación, accesible desde el menú principal mediante el botón “Mi calendario”, donde se visualicen los medicamentos que debe tomar organizados por día y hora.

4. Comentarios sobre cambios realizados en la versión del proyecto entregada

En la nueva versión de FarmacIA-ERS-v1.1 hemos implementado estos cambios. Primero hemos hecho cambios en el índice, ya que el proyecto al ser modificado ha cambiado donde se ubican ciertos puntos. También se ha implementado un cambio del logotipo, por motivos estéticos y el apéndice ha sido eliminado.
Centrándonos en puntos más específicos, en el apartado 2.1 Perspectiva del producto, se ha modificado el contenido, con una perspectiva más profesional y ambiciosa. Además en el 2.2 se ha añadido una funcionalidad, de Inteligencia Artificial, apoyando los nuevos cambios. Y por último, en el 2.5 se ha añadido una nueva dependencia, que es la API de Gemini.

5. Patrones de diseño GOF utilizados en la realización de la práctica

5.1. **Data Access Object (DAO):** Tenemos clases como UserDAO, PillboxDAO y CareGiverDAO. Su función en la aplicación es separar la lógica de acceso a datos (consultas SQL e inserciones) de los objetos de negocio. Y permite que las Actividades no tengan que saber nada sobre SQL, solo llaman a métodos como UserDAO.login().

5.2. **Singleton (Variación):** Aunque en Android se suele gestionar a través del contexto, nuestra clase DatabaseHelper actúa en la práctica como un Singleton para la base de datos. Se asegura que solo exista una instancia de un objeto en toda la aplicación (en este caso, una única conexión a la base de datos para evitar bloqueos).

5.3. **Template Method:** Este patrón está implementado en la clase DatabaseHelper que hereda de SQLiteOpenHelper. Define el esqueleto de un algoritmo en una operación, delegando algunos pasos a las subclases. El framework de Android llama a onCreate() y onUpgrade() en momentos específicos. Y se rellenan los huecos con la lógica de creación de tablas, pero el "cuándo" y "cómo" se ejecutan lo decide la clase padre.

5.4. **Adapter:** Lo aplicamos en la clase CimaAdapter dentro de CalendarActivity. Permite que interfaces incompatibles trabajen juntas. Convierte una lista de objetos de datos (Toma o Medicamento) en una vista visual (View) que el RecyclerView puede mostrar.

5.5. **Strategy (Variación):** Se observa ligeramente en cómo manejas los tipos de usuario. Permite definir una familia de algoritmos y hacerlos intercambiables. Dependiendo de si un usuario es Administrador o Usuario, el sistema puede comportarse de forma distinta, aunque lo manejas principalmente mediante herencia de clases.