Lumen

-Por Daniel Álvarez Morales y Miguel Ángel Montero Benítez.
DAM 2ºA.
IES Albarregas.
(0492 - Proyecto DAM - 2025/2026)
Docente: Mercedes Martinez Fragoso.
Fecha: [dd/mm/aaaa]
Versión documento: v1.0

Enlace a repositorio: EnlaceRepositoriGitHub
Enlace a tablero: EnlaceTableroTrello
Enlace a figma: EnlaceFigmaInterfaz
Enlace a documento: Enlace_PDF_DefincionProyecto

Este proyecto se distribuye bajo la licencia MIT License.
Copyright © 2025 Daniel Álvarez Morales y Miguel Ángel Montero Benítez.

(Se permite el uso, copia, modificación y distribución de este software con o sin modificaciones, siempre que se mantenga este aviso de derechos de autor y la licencia original.)

ÍNDICE

1.-Introducción

Hoy en día, la mayoría de las personas no reciben una educación financiera adecuada. Esto provoca que muchos ciudadanos, especialmente dentro de la clase trabajadora y las familias, tengan dificultades para controlar sus gastos y alcanzar sus metas de ahorro. En una época donde el costo de vida aumenta y el dinero parece desaparecer rápidamente, existe una clara oportunidad de ofrecer una herramienta digital que fomente la educación financiera práctica y el ahorro consciente.

2.- Resumen ejecutivo.

Teniendo en cuenta el problema mencionado, en Lumen hemos creado una sistema que ayuda a cumplir tus objetivos de ahorro, estableciendo límites en los diferentes sectores de gasto. Nuestro objetivo principal es ayudar al ciudadano promedio a ahorrar en una época donde parece que el dinero se esfuma.

Nuestros usuarios objetivo son la clase obrera, familias y cualquier persona que quiera ahorrar para invertir, todos ellos habituales usuarios de smartphones. La solución que proponemos es una aplicación de escritorio que analiza tus gastos, genera gráficas de en qué inviertes tu dinero y permite establecer limitadores diarios para no excederte.

Lumen es una aplicación diseñada para ayudar a los usuarios a gestionar mejor su economía personal. La app analiza los gastos del usuario, genera gráficas interactivas que muestran en qué se está utilizando el dinero y permite establecer límites personalizados en diferentes categorías de consumo. De esta forma, el usuario puede visualizar fácilmente su comportamiento financiero y adoptar hábitos de ahorro más sostenibles.

El desarrollo de Lumen se realizará utilizando Java/Flutter para la interfaz , con una base de datos mySql para el almacenamiento eficiente de la información financiera. Con una interfaz intuitiva y herramientas de análisis claras, Lumen convierte el ahorro en un proceso sencillo, educativo y accesible para todos.

Ayudar a personas con poco o nulo conocimiento financiero a que de una manera simple e intuitiva, puedan llegar a sus objetivos de ahorros, teniendo , mediante una interfaz muy sencilla, un control de gastos e ingresos

3 Justificación

Hemos elegido este modelo de aplicación porque nos parece interesante y sus desarrollo aporta los siguientes beneficios:

Beneficio técnico y educativo:

Nos permite aprender a desarrollar gráficos dentro de un programa, así como a interpretar operaciones financieras mediante una representación visual clara de los gastos e ingresos.

Beneficio de uso:

Ayuda al usuario a gestionar sus recursos monetarios a través de una interfaz amigable e intuitiva, mejorando la toma de decisiones financieras a largo plazo.

La aplicación puede integrarse en distintos ámbitos de la economía extremeña, especialmente entre pymes, autónomos y emprendedores, para facilitar la planificación y gestión de sus recursos financieros. Su uso puede adaptarse a varios sectores clave de la región:

Sectores agrícolas:

La aplicación podría ayudar a agricultores y cooperativas a registrar gastos en semillas, abonos, combustible o maquinaria, y a controlar los ingresos derivados de la venta de productos.
Esto facilita la planificación de campañas agrícolas y el control del flujo de caja de cada temporada

Ejemplo de caso de uso: Un agricultor utiliza la aplicación para registrar los gastos de gasóleo y abono, y conocer los beneficios obtenidos tras la venta de la cosecha.

PYMES:

Las pymes pueden utilizar la aplicación para llevar un control simple y visual de sus ingresos, gastos e inversiones menores, sin necesidad de software contable complejo.
Les permitiría tomar decisiones rápidas sobre presupuestos, detectar meses con más gasto y mejorar su rentabilidad.

Ejemplo de caso de uso: Una pyme de productos locales controla los ingresos de ventas semanales y los gastos de transporte para mejorar la planificación de pedidos.

Autónomos:

Los trabajadores por cuenta propia podrían usarla para organizar sus facturas y movimientos económicos diarios, separando gastos personales y profesionales.
Esto les ayudaría a planificar impuestos trimestrales y a mantener una visión clara de su actividad económica.

Ejemplo de caso de uso: Un autónomo del sector servicios utiliza la app para visualizar sus ingresos y gastos mensuales, facilitando la presentación de impuestos.

Clústeres:

La aplicación también podría integrarse con clústeres regionales, como el Clúster TIC Extremadura o el Clúster Agroalimentario, para fomentar la digitalización y modernización

de la gestión económica en las empresas asociadas.
De esta manera, el proyecto contribuiría a los objetivos de innovación y eficiencia que estos grupos empresariales promueven en la región.

El análisis de nuestro benchmark nos deja con los siguientes productos extremeños similares, aunque todos ellos pertenecen al ámbito de la banca:

A) **Caja Rural de Extremadura – App Ruralvía**
Aplicación de banca digital que permite consultar movimientos, realizar transferencias y gestionar tarjetas.
Aunque es funcional para la administración básica del dinero, no ofrece una **clasificación automática de gastos** ni análisis visual avanzado.
Su principal enfoque es la operativa bancaria, no la interpretación financiera del usuario.

B) **Caja Almendralejo – App de Banca Móvil**
Permite la gestión de cuentas y operaciones habituales desde el móvil.
La aplicación es estable y cercana para usuarios de la región, pero su enfoque sigue siendo transaccional y **carece de herramientas de planificación financiera** como presupuestos o alertas inteligentes.

C) **Unicaja Banco (antes Liberbank en Extremadura) – App Unicaja**
App con funciones modernas como pagos móviles, bizum y gestión de productos financieros.
A pesar de ello, su función de análisis del gasto es **limitada y poco visual** en comparación con gestores financieros especializados.
No incentiva el ahorro ni permite proyecciones de gastos a futuro.

D) **Ibercaja – App Ibercaja (ampliamente presente en el suroeste peninsular)**
Incluye gráficos de balance y categorización básica.
Sin embargo, la clasificación no siempre es automática o precisa, y la herramienta **no permite establecer metas de ahorro personalizadas**.

Nuestra app recoge todas estas ods de la agenda 2030:

ODS 4 → Educación de calidad

La app enseña a los usuarios a interpretar sus finanzas personales, fomentando la educación económica y la toma de decisiones responsables.

ODS 8 →Trabajo decente y crecimiento económico.

Ayuda a autónomos, pymes y emprendedores a gestionar mejor sus recursos financieros y mantener la estabilidad económica.

ODS 9 → Industria, innovación e infraestructura.

Promueve la digitalización de la gestión financiera en pequeñas empresas y sectores tradicionales (como el agrícola).

ODS 12 → Producción y consumo responsables.

Fomenta el control del gasto y la planificación, evitando el consumo impulsivo o irresponsable.

4. Historias de usuario

Los usuarios que tienen estas demandas expresadas en “historias de usuario” son las que satisfacen nuestras funcionalidades.

Historias principales de usuario del programa:

HU-001: Visualizar gráfico general

Como usuario altamente ocupado,quiero ver un gráfico sencillo y escueto con mis gastos, ingresos y ahorros,para poder controlar y planificar mi dinero según el periodo que seleccione (semana, mes o año).

HU-002: Agregar ingresos hipotéticos
Como usuario preocupado, quiero introducir posibles ingresos en la gráfica de finanzas para poder calcular con seguridad el pago de deudas ante posibles imprevistos.

HU-003: ver lista de gastos

Como usuario altamente ocupado, quiero una lista que refleje todos los gastos de un día, semana o mes, para identificar o tener en cuenta los días o picos más costosos.

HU-004: Crear gráficas alternativas

Como usuario que desea explorar diferentes escenarios, quiero poder crear gráficas alternativas con distintos gastos o ingresos simulados, para visualizar cómo cambiaría mi línea de ahorros sin modificar mi gráfica original.

HU-005: Gráficas por categorías

Como usuario que busca identificar sus hábitos de gasto, quiero ver gráficas que muestren en qué categorías gasto más, para poder detectar y reducir los gastos innecesarios.

HU-001
Ver gráfico de gastos, ingresos y ahorros → **Must** (Imprescindible)

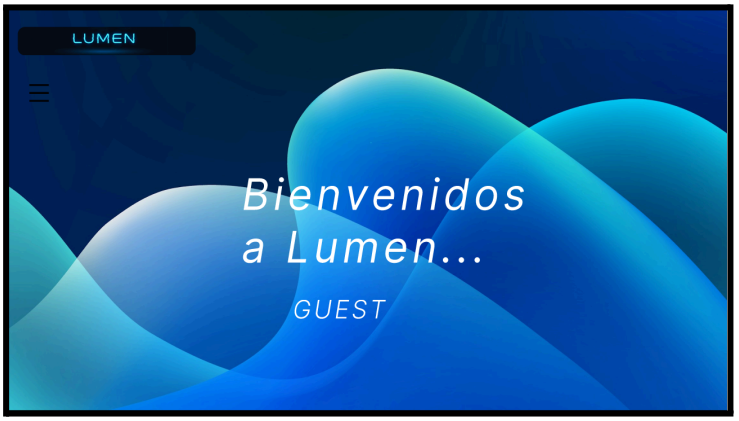
HU-002
Añadir ingresos hipotéticos a la gráfica de finanzas → **Could** (Conveniente)

HU-003
ver lista detallada de gastos → **Should** (Importante)

HU-004
Crear gráficas alternativas (modo premium) → **Could** (Conveniente)

Trazabilidad HU

HU-002: Agregar ingresos hipotéticos
Como usuario preocupado, quiero introducir posibles ingresos en la gráfica de finanzas para poder calcular con seguridad el pago de deudas ante posibles imprevistos.



(Inicio aplicación sesión iniciada)

(Menú desplegable inicio a gráficos)

(Pantalla de gráficos con las 3 listas a la lista ingresos)

(En las listas de ingresos agregamos ingreso)

(Agregando ingreso)

(No se pueden agregar ingresos negativos)

(Si agregas un ingreso negativo saldrá un mensaje informativo)

(Si agregas un ingresos positivo)

(Se agregará a lista de ingresos exitosamente)

Criterios aceptación (CA)

- El usuario puede introducir un ingreso manualmente con cantidad y categoría.
- El ingreso hipotético se refleja visualmente sin alterar los datos reales.
- El usuario puede eliminar o editar el ingreso simulado.

Pruebas

- Introducir un ingreso ficticio y comprobar que se añade a la simulación.
- Verificar que no modifica la base de datos real.

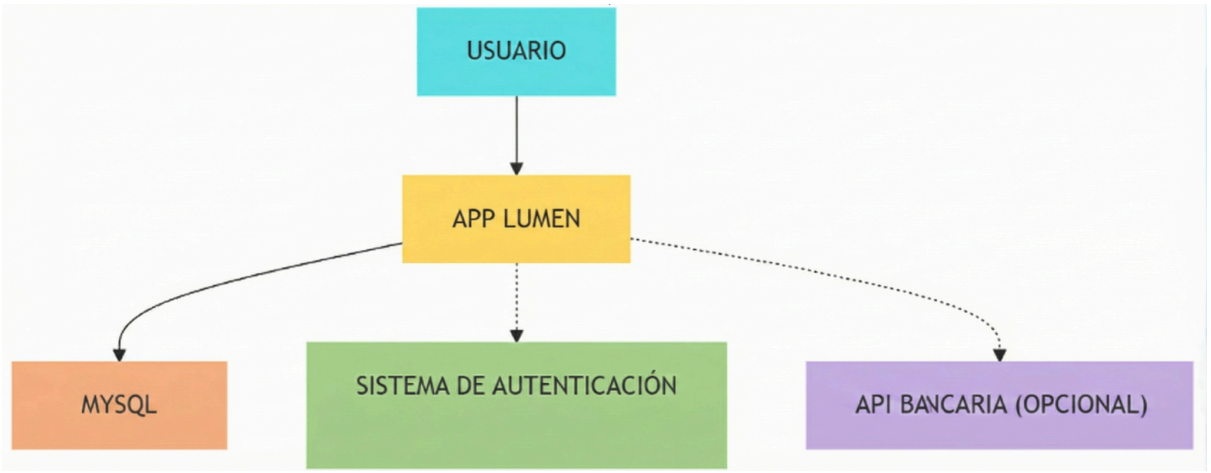
- Borrar el ingreso y confirmar que desaparece de la vista.

5. Arquitectura

Lumen es una aplicación móvil que ayuda a los usuarios a gestionar sus finanzas personales: registrar ingresos y gastos, establecer límites, generar gráficas y alcanzar objetivos de ahorro.

En el diagrama de contexto, hay varios agentes:

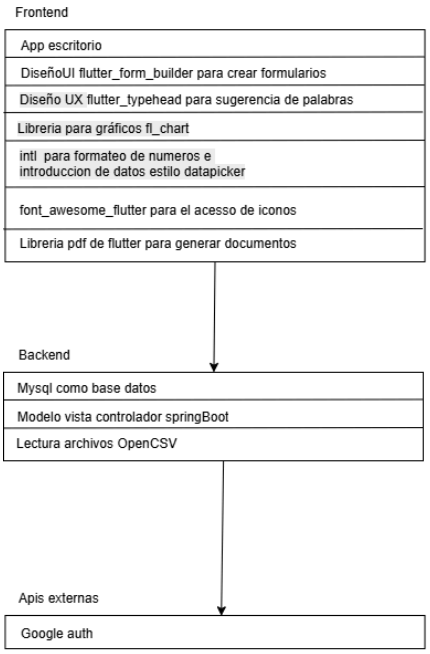
-Usuario final: individuos, familias o trabajadores que quieren controlar y optimizar su dinero.
-App Lumen: sistema central donde se procesan los datos financieros y se generan informes.
-Base de datos MySQL: almacenamiento de información financiera de manera segura y eficiente.
-Sistema de autenticación: gestiona el login y protege los datos del usuario.
-API bancaria (opcional/futuro): permite integrar cuentas bancarias para análisis automático de movimientos.



Podemos observar la rama principal que compone **Usuario-App Lumen-Sistema de autenticacion**

Y de ella cuelga la subrama **Sistema de autenticación-Api**

USUARIO —— APP LUMEN —— MySQL
                    |
                    ├── SISTEMA DE AUTENTICACIÓN(GOOGLE AUTH)
                    └── API BANCARIA (OPCIONAL)



ADR-001: Arquitectura Monolítica con Spring Boot (Backend)

Desarrollaremos el backend con una API monolítica spring boot que se comunique directamente con mySQL. Todo las operaciones de sistema en un solo artefacto desplegable.
Planteamos una organización simple en la aplicación para desarrollarla con éxito.
Es simple y básico para el despliegue y mantenimiento al concentrar todas las operaciones en un solo artefacto.

ADR-002: Arquitectura flutter (Frontend)

Desarrollaremos el fronted con Flutter.
Es una solución simple y básica para el despliegue y mantenimiento, ya que Flutter permite un único código para varias plataformas mientras que el backend concentra todas las operaciones en un solo artefacto.

ADR-003: Arquitectura MySQL (Base de datos)

Utilizaremos MySQL como sistema de gestión de base de datos principal.
Es una solución simple y estable para el despliegue y mantenimiento, ya que MySQL ofrece un entorno confiable y ampliamente soportado, permitiendo que el backend gestione todas las operaciones de forma centralizada dentro de un solo artefacto.

La aplicación de finanzas personales se apoyará en varias integraciones y componentes para funcionar correctamente y mantener un flujo de datos consistente:

Se integrará con la API de Google para login de usuarios, gestionando credenciales de forma segura mediante OAuth2.

Librerías de flutter como fl_chart para representación de gráficos, intl para formateo de numeros e introducción de datos estilo datepicker, font_awesome_flutter para el acceso de iconos, Librería pdf de flutter para generar documentos y otra librería lectura archivos “OpenCSV”.

Las credenciales y secretos sensibles se almacenarán únicamente en variables de entorno, nunca en el código ni en archivos de configuración.

Librerías lombok por ejemplo para ahorrar getter y setter , maven ya proporcionado por spring boot.

Durante el desarrollo de la aplicación de finanzas personales, se han identificado varios riesgos técnicos que podrían afectar la estabilidad, seguridad o mantenibilidad del sistema. A continuación se detallan los riesgos y las estrategias de mitigación:

Riesgo: Fallos en la integridad de datos durante la importación (Excel/CSV)

Descripción: La funcionalidad de carga masiva de movimientos mediante archivos externos es un punto crítico. Existe el riesgo de que el usuario suba archivos corruptos, con formatos de fecha no reconocidos o estructuras inválidas, lo que podría generar excepciones no controladas o dejar la base de datos en un estado inconsistente (ej: guardar solo la mitad de los movimientos).

Mitigación: Uso de librerías robustas como Apache POI u OpenCSV con validación estricta de tipos de datos antes de la persistencia.

Riesgo: Seguridad en la autenticación y gestión de sesiones

Descripción: Depender de una autenticación externa (Google Auth) requiere una gestión correcta de las redirecciones y del ciclo de vida de la sesión. Una configuración incorrecta podría exponer datos de usuario o permitir el secuestro de sesiones. Además, la exposición de credenciales del proveedor (Client Secret) es un riesgo crítico.

Mitigación: Integración de Google Authentication mediante el estándar OAuth2 de Spring Security. Almacenamiento de las credenciales sensibles (Google Client ID y Secret) exclusivamente en variables de entorno, nunca en el código fuente ni en el repositorio.

Riesgo: Problemas de visualización y compatibilidad en Flutter

Descripción: Al desarrollar una interfaz única para Escritorio y Web, existe el riesgo de que los elementos visuales (gráficas o tablas de datos) sufran desbordamientos ("overflow errors") al cambiar el tamaño de la ventana, rompiendo la experiencia de usuario.

Mitigación: Uso de widgets responsivos en Flutter (Expanded, LayoutBuilder, Flex) en lugar de dimensiones fijas. Pruebas manuales de redimensionamiento de ventana y uso de la librería intl para asegurar el formato correcto de fechas y monedas en cualquier sistema operativo.

Riesgo: Rendimiento en consultas de históricos

Descripción: A medida que el usuario acumula un gran volumen de transacciones, las consultas para generar gráficas o listar movimientos podrían volverse lentas.

Mitigación: Implementación de Paginación en los endpoints del Backend para cargar datos por bloques. Optimización de índices en la base de datos MySQL, específicamente en columnas de filtrado frecuente como fecha y usuario_id.

6. Requisitos no funcionales (NFR)

1. RNF-01:Rendimiento y latencia(Carga de datos)

- **Definición** :
    - Dado el RNF de rendimiento de carga de datos,
    - Cuando se lee el requisito,
    - Entonces indica:
        - Métrica: Tiempo de latencia desde la solicitud del usuario hasta el renderizado visual completo (Time-to-Interactive).
        - Umbral: Menos de 3 segundos para la carga de gráficos y datos.
        - Método de verificación: Pruebas de carga (Load Testing) con herramientas como JMeter o las DevTools del navegador en un entorno de red 4G/Wifi estándar.

2. RNF-02: Usabilidad y Accesibilidad (Contraste Visual)

Este requisito garantiza que la aplicación sea cómoda de usar y accesible, alineándose con el objetivo de "elementos armoniosos".

- **Definición** :
    - Dado el RNF de accesibilidad visual,
    - Cuando se lee el requisito,
    - Entonces indica:
        - Métrica: Ratio de contraste de color entre el texto/iconos y el fondo.
        - Método de verificación: Auditoría automática con herramientas como Google Lighthouse o Accessibility Scanner en Figma

3. RNF-03: Fiabilidad de Datos (Exactitud Visual)

Este requisito es general pero fundamental para una app de finanzas: lo que veo en la pantalla debe ser matemáticamente real.

- **Definición** :
    - Dado el RNF de exactitud en la representación de datos,
    - Cuando se lee el requisito,
    - Entonces indica:
        - Métrica: Discrepancia entre el valor almacenado en la base de datos y el valor mostrado en la lista o gráfico.
        - Umbral: 0% de error. El valor numérico en la pantalla debe ser idéntico al registro guardado en MySQL.
        - Método de verificación: Prueba de "Caja Negra": ingresar un gasto conocido (ej. 50€), consultar la base de datos para confirmar el guardado, y verificar que en la lista de la app aparece exactamente "50€".

4. RNF-04: Portabilidad (Entorno de Desarrollo)

Garantiza que cualquier nuevo desarrollador del equipo pueda ejecutar el proyecto inmediatamente, eliminando el problema de "en mi máquina funciona".

- **Definición** :
    - Dado el RNF de portabilidad del entorno,
    - Cuando se lee el requisito,
    - Entonces indica:
        - Métrica: Tasa de éxito en el despliegue local usando un solo comando.
        - Umbral: El comando `docker-compose up` debe levantar todos los servicios (API, BD) sin errores manuales de configuración en una instalación limpia.
        - Método de verificación**:** Ejecución del script de despliegue en una máquina "limpia" (sin dependencias previas de Java/MySQL instaladas localmente, solo Docker).

5. RNF-05: Adaptabilidad de Interfaz (Tema Claro/Oscuro)

Este es un requisito de diseño muy visual y fácil de entender: si el usuario cambia la configuración de su móvil, la app debe adaptarse.

- **Definición :**
    - Dado el RNF de adaptabilidad visual de la interfaz,
    - Cuando se lee el requisito,
    - Entonces indica:
        - Métrica**:** Porcentaje de elementos de la interfaz (fondos, textos, botones) que cambian de paleta de color correctamente.
        - Umbral: 100% de los elementos visibles deben adaptarse al tema seleccionado sin perder legibilidad.
        - Método de verificación: Inspección visual manual cambiando la configuración del sistema operativo (o un botón en la app) de "Modo Claro" a "Modo Oscuro" y viceversa.

6. RNF-06: Integridad de Capas MVC y Estándares de Código

Este requisito asegura que el código no se convierta en un "espagueti", manteniendo separadas las responsabilidades tal como definisteis en vuestra arquitectura (Controlador/ Servicio/ DAO).

- **Definición** :
    - Dado el RNF de calidad y adherencia al patrón MVC,
    - Cuando se analiza el código fuente del backend,
    - Entonces indica:
        - Métrica: Número de violaciones de reglas de arquitectura (ej: un Controlador accediendo directamente al DAO) y de estilo (naming conventions).
        - Umbral: 0 violaciones arquitectónicas (estricto cumplimiento de capas) y 0 errores de severidad alta en el linter.
        - Método de verificación:
            1. Estático: Ejecución automática de Checkstyle o SonarQube en el pipeline de compilación.
            2. Arquitectónico: Uso de una prueba unitaria con ArchUnit (librería Java) que verifique que el paquete controller solo accede a service, y service a dao.

7.RNF-07: Usabilidad de Inicio de Sesión (Google Auth)

Este requisito garantiza que el proceso de inicio de sesión con Google sea rápido y que la aplicación maneje correctamente las credenciales del usuario.

- **Definición:**
    - Dado el RNF de usabilidad del proceso de autenticación con Google,
    - Cuando el usuario pulsa el botón "Iniciar sesión con Google" por primera vez,
    - Entonces indica:
        - Métrica**:** Tiempo transcurrido desde que se pulsa el botón hasta que la pantalla principal de la aplicación es visible (incluyendo la comunicación con el backend para crear/verificar el usuario).
        - Umbral**:** Menos de 5 segundos.
        - Método de verificación**:** Pruebas de usabilidad (Usability Testing) con un cronómetro en la mano, o registro automático de eventos (Analytics/Logs de Firebase) en el dispositivo del usuario para medir el evento **login_start** hasta **home_screen_loaded**.