COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

MÁSTER EN BIG DATA

TRABAJO FIN DE MÁSTER

Diseño e implementación de un dashboard de inteligencia bancaria basado en datos agregados

Autor: Claudia Meana Iturri

Director: José Daniel Menendez Hernandéz

Madrid

Curso académico 2025/26

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Diseño e implementación de un dashboard de inteligencia bancaria basado en datos agregados

en la ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

de la UNIVERSIDAD PONTIFICIA COMILLAS en el curso académico 2025/26 es de mi autoría, original e inédito y no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente, y la información que ha sido tomada de otros documentos está debidamente referenciada.

MEANA ITURRI CLAUDIA - 29500571N

Firmado digitalmente por MEANA ITURRI CLAUDIA - 29500571N
Fecha: 2026.05.31 18:35:52 +02'00'

Fdo.: Claudia Meana Iturri Fecha: 08/06/2026

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

MENENDEZ HERNANDEZ JOSE DANIEL - 54494034L

Digitally signed by MENENDEZ HERNANDEZ JOSE DANIEL - 54494034L
Date: 2026.06.01 15:02:57 +02'00'

Fdo.: José Daniel Menendez Hernandéz Fecha: 08/06/2026

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

MÁSTER EN BIG DATA

TRABAJO FIN DE MÁSTER

Diseño e implementación de un dashboard de inteligencia bancaria basado en datos agregados

Autor: Claudia Meana Iturri

Director: José Daniel Menendez Hernandéz

Madrid

Curso académico 2025/26

Agradecimientos

Quiero dar las gracias, en primer lugar, a Dani, por su ayuda técnica durante las reuniones diarias y por sus consejos siempre acertados ante las dudas que han ido surgiendo a lo largo de este Trabajo Fin de Máster.

También quiero agradecer a Joan todo el conocimiento de negocio que me ha transmitido, así como la ayuda y el aprendizaje en habilidades profesionales que he adquirido trabajando contigo.

Al resto de personas de Oliver Wyman, gracias por el apoyo, el feedback y la disposición a ayudar durante el desarrollo del proyecto. Los comentarios y aportaciones han sido fundamentales para mejorar la herramienta y acercarla a un uso real dentro de la empresa.

A todos los profesores del máster, gracias por haberme permitido descubrir un campo que antes no conocía y que ahora me apasiona. Este año me ha ayudado a recuperar la ilusión por aprender y a disfrutar de ir a clase.

Finalmente, quiero dar las gracias a mi familia y a mis amigos, por aguantarme, apoyarme y estar siempre ahí durante todo este proceso.

Resumen del proyecto

Diseño e implementación de un dashboard de inteligencia bancaria basado en datos agregados

Autor: Claudia Meana Iturri.

Director: José Daniel Menendez Hernandéz.

Entidad colaboradora: ICAI – Universidad Pontificia Comillas.

Resumen

Este Trabajo Fin de Máster presenta el diseño e implementación de un dashboard de inteligencia bancaria basado en datos financieros agregados. El proyecto se desarrolla en el contexto de Oliver Wyman, una firma internacional de consultoría de gestión que trabaja con empresas e instituciones en retos estratégicos, operativos, tecnológicos y de transformación [1]. Dentro de su práctica de servicios financieros, la firma trabaja con bancos, entidades de crédito, compañías de pagos e instituciones de inversión [2, 3]. En el contexto de este proyecto, el foco se sitúa principalmente en el ámbito bancario en España, donde la explotación de datos agregados puede aportar valor tanto para análisis internos como para posibles propuestas a entidades financieras.

El punto de partida era una situación en la que la empresa ya disponía de datos financieros agregados procedentes de una fuente externa, pero su explotación dependía principalmente de análisis bajo demanda realizados por perfiles técnicos. Cuando un usuario de negocio necesitaba una métrica o visualización concreta, era necesario preparar el análisis mediante notebooks o herramientas locales. Aunque este enfoque permitía responder a preguntas puntuales, no era suficientemente accesible, reutilizable ni interactivo para usuarios de negocio.

El objetivo del proyecto fue desarrollar una herramienta analítica que permitiera consultar y comparar información bancaria de forma más autónoma. El dashboard permite seleccionar una entidad bancaria, compararla frente a otros bancos o grupos de bancos, consultar KPIs principales y analizar dimensiones como clientes, productos financieros, volumen, penetración, concentración, oportunidades de venta cruzada y salidas hacia neobancos.

Palabras clave: Inteligencia bancaria, Dashboard analítico, Datos financieros agregados, Streamlit, Arquitectura medallón.

1. Introducción

La motivación principal del proyecto surge de la diferencia entre disponer de datos y poder convertirlos en información útil para la toma de decisiones. En el sector bancario, los datos sobre clientes, productos, saldos y comportamiento

financiero pueden aportar valor estratégico, pero solo si se presentan de forma clara, comparable y accionable. En una firma de consultoría, esta necesidad es especialmente relevante, ya que el trabajo con entidades financieras exige transformar información compleja en diagnósticos, comparativas e ideas de negocio comprensibles.

2. Definición del proyecto

El proyecto consiste en el desarrollo de un dashboard analítico en Streamlit para la explotación de datos financieros agregados. La herramienta está orientada principalmente a usuarios internos de negocio, que pueden utilizarla para consultar métricas, comparar entidades y preparar análisis o presentaciones. Además, el dashboard puede servir como herramienta demostrativa para posibles clientes externos, especialmente bancos interesados en conocer su posición relativa frente al mercado.

El alcance incluye la preparación de datos, el diseño de la arquitectura, la implementación del dashboard, la creación de visualizaciones interactivas, la validación de métricas, la mejora del rendimiento, la incorporación de seguridad mediante usuario y contraseña y la preparación del código para futuras actualizaciones.

3. Descripción del sistema

La solución combina un pipeline de datos en Python con un dashboard interactivo en Streamlit. El procesamiento se organizó siguiendo una lógica inspirada en la arquitectura medallón, adaptada al flujo *raw* $\rightarrow$ *silver* $\rightarrow$ *gold* $\rightarrow$ *final_dashboard*. La capa *raw* conserva los datos originales, la capa *silver* limpia y enriquece la información, la capa *gold* estructura los datos mediante un modelo estrella y la capa *final_dashboard* genera ficheros precalculados para su consumo por Streamlit.

El dashboard se organiza en tres grandes páginas analíticas: *Visión General*, *Penetración y Concentración* y *Salidas hacia Neobancos*. Estas páginas permiten analizar KPIs principales, distribución de clientes y volumen, perfil medio del cliente, productos financieros, oportunidades de venta cruzada, concentración de volumen y transferencias hacia neobancos.

4. Resultados

El resultado principal fue una herramienta funcional, desplegada y utilizada en un contexto real. El dashboard permitió pasar de un modelo basado en notebooks y visualizaciones aisladas a una aplicación interactiva, organizada por páginas y orientada a usuarios de negocio.

También se construyó un pipeline de datos adaptado a las necesidades del dashboard. La creación de una capa final con ficheros precalculados permitió separar las tareas pesadas de procesamiento de la interacción del usuario con la aplicación. Esta decisión mejoró la usabilidad de la herramienta y facilitó su mantenimiento futuro.

Además, el despliegue temprano permitió recoger feedback operativo de usuarios internos. Este feedback ayudó a mejorar la narrativa de las páginas, aclarar el significado de determinadas métricas y corregir cálculos que no reflejaban completamente la lógica de negocio esperada.

5. Conclusiones

El proyecto demuestra que el valor de una herramienta de inteligencia bancaria no depende únicamente de mostrar gráficos, sino de construir una arquitectura de datos y una narrativa analítica que permitan transformar información agregada en conocimiento útil. La solución desarrollada facilita el acceso a datos financieros, reduce la dependencia de perfiles técnicos, homogeneiza métricas y ofrece una base para explorar posibles casos de uso comerciales con entidades bancarias.

Entre las principales limitaciones destacan la dependencia de ficheros CSV y Parquet, la validación parcialmente manual de cálculos, la ausencia de histórico completo para todas las métricas y la falta de una evaluación formal de usabilidad. Como trabajos futuros, se propone ampliar las visualizaciones, reforzar la trazabilidad de los cálculos, automatizar validaciones, evolucionar hacia una arquitectura de almacenamiento más robusta e incorporar permisos más avanzados.

Abstract

Diseño e implementación de un dashboard de inteligencia bancaria basado en datos agregados

Author: Claudia Meana Iturri.

Supervisor: José Daniel Menendez Hernandéz.

Collaborating entity: ICAI – Universidad Pontificia Comillas.

Abstract

This Master's Thesis presents the design and implementation of a banking intelligence dashboard based on aggregated financial data. The project was developed in the context of Oliver Wyman, an international management consulting firm that supports companies and institutions in strategic, operational, technological and transformation challenges [1]. Within its financial services practice, the firm works with banks, credit institutions, payment companies and investment firms [2, 3]. In this project, the focus is mainly on banking in Spain, where aggregated financial data can generate value for internal analysis and potential proposals to financial institutions.

The starting point was a situation in which the company already had access to aggregated financial data from an external source, but the use of that data depended mainly on ad hoc analysis performed by technical profiles. When a business user needed a specific metric or visualization, the analysis had to be prepared through notebooks or local tools. Although this approach was useful for specific questions, it was not sufficiently accessible, reusable or interactive for business users.

The objective of the project was to develop an analytical tool that would allow users to query and compare banking information more autonomously. The dashboard allows users to select a banking institution, compare it against other banks or groups of banks, consult key performance indicators and analyze dimensions such as customers, financial products, volume, penetration, concentration, cross-selling opportunities and outflows to neobanks.

Keywords: Banking intelligence, Analytical dashboard, Aggregated financial data, Streamlit, Medallion architecture.

1. Introduction

The main motivation of the project lies in the difference between having data and being able to transform it into useful information for decision-making. In the banking sector, data on customers, products, balances and financial behavior can provide strategic value, but only if it is presented in a clear, comparable and actionable way. In a consulting firm, this need is especially relevant, since work

with financial institutions requires transforming complex information into understandable diagnoses, comparisons and business insights.

2. Project definition

The project consists of the development of an analytical dashboard in Streamlit for the exploitation of aggregated financial data. The tool is mainly intended for internal business users, who can use it to consult metrics, compare institutions and prepare analyses or presentations. In addition, the dashboard can also serve as a demonstrative tool for potential external clients, especially banks interested in understanding their relative position in the market.

The scope includes data preparation, architecture design, dashboard implementation, interactive visualizations, metric validation, performance improvement, password-based access control and preparation of the code for future updates.

3. Description of the system

The solution combines a Python data pipeline with an interactive dashboard built in Streamlit. The data processing workflow follows a logic inspired by medallion architecture, adapted to the flow *raw* $\rightarrow$ *silver* $\rightarrow$ *gold* $\rightarrow$ *final_dashboard*. The *raw* layer stores the original data, the *silver* layer cleans and enriches the information, the *gold* layer structures the data through a star-schema logic, and the *final_dashboard* layer generates precomputed files for Streamlit.

The dashboard is organized into three main analytical pages: *General Overview*, *Penetration and Concentration*, and *Outflows to Neobanks*. These pages allow users to analyze key performance indicators, customer and volume distribution, average customer profile, financial products, cross-selling opportunities, volume concentration and transfers to neobanks.

4. Results

The main result was a functional dashboard, deployed and used in a real business context. The tool made it possible to move from a model based on notebooks and isolated visualizations to an interactive application organized around analytical pages and oriented toward business users.

A data pipeline adapted to the needs of the dashboard was also built. The creation of a final layer of precomputed files separated heavy data preparation tasks from user interaction with the application. This design improved the usability of the tool and made the system easier to maintain and update.

In addition, the early deployment of the dashboard allowed operational

feedback to be collected from internal users. This feedback helped refine the analytical narrative of the pages, clarify the meaning of specific metrics and correct calculations that did not fully reflect the expected business logic.

5. Conclusions

The project shows that the value of a banking intelligence tool does not depend only on displaying charts, but also on building a data architecture and analytical narrative capable of transforming aggregated information into useful knowledge. The developed solution improves access to financial data, reduces dependence on technical profiles, homogenizes metrics and provides a basis for exploring potential commercial use cases with banking institutions.

The main limitations are the reliance on CSV and Parquet files, partially manual validation of calculations, the absence of complete historical data for all metrics and the lack of a formal usability evaluation. Future work should include expanding visualizations, reinforcing calculation traceability, automating validation processes, evolving toward a more robust storage architecture and incorporating more advanced permission management.

Índice de la memoria

Índice de figuras

Índice de tablas

Parte I

Memoria

1. Introducción

Este capítulo introduce el proyecto desarrollado en el Trabajo Fin de Máster. En primer lugar, se presenta la motivación empresarial, tecnológica y analítica que justifica el desarrollo del dashboard. En segundo lugar, se explica el objetivo general de la memoria y el tipo de solución construida. Finalmente, se describe la estructura del documento para orientar al lector a lo largo de los capítulos posteriores.

1.1. Motivación del proyecto

El sector bancario es uno de los ámbitos donde el uso de datos tiene mayor relevancia estratégica. Las entidades financieras operan con grandes volúmenes de información sobre clientes, productos, saldos, transferencias y comportamiento financiero. Esta información puede utilizarse para comprender mejor la posición competitiva de una entidad, detectar oportunidades comerciales, analizar concentración de volumen, identificar patrones de vinculación de clientes o estudiar la aparición de nuevos competidores digitales. Sin embargo, el valor de estos datos depende de la capacidad de transformarlos en métricas claras, comparables y accionables.

Este proyecto se desarrolla en el contexto de Oliver Wyman, una firma internacional de consultoría de consultoría estratégica que trabaja con compañías e instituciones en retos estratégicos, operativos, tecnológicos y de transformación. La firma cuenta con una práctica relevante de servicios financieros, desde la que colabora con bancos, entidades de crédito, compañías de pagos e instituciones financieras en retos relacionados con regulación, transformación digital, estrategia, clientes y modelos de negocio [1, 2, 3]. En el contexto de este TFM, el proyecto se sitúa especialmente en el ámbito bancario, donde existe una necesidad clara de convertir datos financieros agregados en análisis útiles para usuarios de negocio.

La empresa ya disponía de datos financieros agregados procedentes de una fuente externa. Estos datos contenían información sobre clientes, entidades bancarias, productos financieros y volumen asociado a cada producto en cada entidad. Por tanto, el problema principal no era la ausencia de datos, sino la falta de una herramienta común, accesible e interactiva que permitiera explotarlos de forma sistemática. Antes del desarrollo del proyecto, muchas visualizaciones dependían de análisis bajo demanda realizados por perfiles técnicos mediante notebooks o herramientas locales. Aunque este enfoque era útil para responder a preguntas concretas, presentaba limitaciones de accesibilidad, reutilización, interactividad y escalabilidad.

La motivación principal del TFM surge de esa brecha entre disponibilidad de datos y capacidad de uso por parte de usuarios de negocio. Un conjunto de datos puede tener un valor elevado, pero si su explotación depende constantemente de perfiles técnicos, ese valor queda limitado. El dashboard desarrollado busca

reducir esta dependencia mediante una herramienta que permita consultar indicadores, comparar entidades bancarias, analizar productos, estudiar clientes y explorar patrones de comportamiento sin necesidad de modificar código ni acceder directamente a los ficheros originales.

Además, el proyecto tiene una motivación comercial. Al permitir comparar una entidad bancaria frente a otros bancos o grupos de bancos, el dashboard puede servir como herramienta demostrativa para potenciales clientes externos. Una entidad financiera puede utilizar este tipo de análisis para comprender mejor su posición relativa, detectar áreas estratégicas de mejora y acceder a información agregada que normalmente no estaría disponible dentro de sus propios sistemas internos. De esta forma, el proyecto no solo responde a una necesidad interna de eficiencia y autonomía, sino que también permite mostrar el valor potencial de los datos disponibles.

Desde el punto de vista tecnológico, el proyecto también se justifica por la necesidad de construir una solución eficiente y mantenible. El dashboard no podía limitarse a mostrar gráficos, sino que debía apoyarse en un flujo de datos capaz de limpiar, estructurar y precalcular métricas antes de su visualización. Esta necesidad fue especialmente importante porque el proyecto no se apoyaba en una base de datos analítica tradicional, sino en ficheros procesados previamente. Por ello, una parte central del trabajo consistió en diseñar una arquitectura de datos basada en capas y optimizada para reducir la latencia.

1.2. Objetivo general de la memoria

El objetivo general de esta memoria es describir el proceso de diseño, implementación y evaluación de un dashboard de inteligencia bancaria basado en datos financieros agregados. La memoria explica cómo se identificó la necesidad del proyecto, qué tecnologías y métodos se utilizaron, cómo se diseñó la arquitectura del sistema, cómo se implementó la herramienta y qué resultados se obtuvieron tras su desarrollo.

El trabajo no se limita a presentar una interfaz visual. La memoria muestra también las decisiones técnicas y metodológicas que hicieron posible la construcción del dashboard: el uso de Python y pandas para el procesamiento de datos, Streamlit para la aplicación web, Plotly para las visualizaciones interactivas, Figma para el prototipado inicial, una metodología iterativa basada en sprints y una arquitectura de datos inspirada en la lógica medallón.

Asimismo, la memoria analiza la evolución del modelo de datos desde una estructura inicial basada en modelo estrella hasta una capa final de ficheros precalculados orientada al consumo directo por parte del dashboard. Esta evolución fue necesaria para mejorar el rendimiento de la herramienta y reducir los tiempos de carga. Por ello, el documento presta especial atención a la relación entre arquitectura de datos, experiencia de usuario y utilidad empresarial.

Finalmente, la memoria evalúa los resultados obtenidos y presenta una discusión crítica sobre el alcance del proyecto. Se analizan los principales logros,

como el desarrollo de una herramienta funcional, la reducción de latencia, la incorporación de feedback operativo, la mejora de la accesibilidad a los datos y la preparación del sistema para futuras actualizaciones. También se reconocen las limitaciones del trabajo, especialmente la dependencia de ficheros, la validación parcialmente manual de métricas, la falta de histórico completo para algunos análisis y la ausencia de una evaluación formal de usabilidad.

1.3. Estructura del documento

La memoria se organiza en los siguientes capítulos:

- El Capítulo 2, *Descripción de las Tecnologías*, presenta las tecnologías, herramientas, métodos y marcos conceptuales necesarios para comprender el proyecto. Se describen Python, pandas, Streamlit, Plotly, Figma, la metodología ágil e iterativa, la arquitectura medallón y el modelo estrella.
- El Capítulo 3, *Estado de la Cuestión*, analiza la situación previa y las soluciones existentes. Se revisan las prácticas internas basadas en notebooks y visualizaciones bajo demanda, así como herramientas externas de inteligencia de negocio como Power BI, Tableau o Looker. Este capítulo permite identificar el hueco que cubre el TFM.
- El Capítulo 4, *Definición del Trabajo*, delimita el proyecto. En él se explican la justificación, los objetivos, la metodología seguida y la planificación temporal del desarrollo.
- El Capítulo 5, *Sistema Desarrollado*, describe el núcleo técnico del TFM. Se presentan los actores, datos de entrada y salida, requisitos funcionales y no funcionales, casos de uso, arquitectura general, navegación del dashboard, diseño funcional de páginas, modelo de datos, flujo de actualización e implementación.
- El Capítulo 6, *Análisis de Resultados*, evalúa los principales resultados del proyecto. Se analiza la construcción de la herramienta, la estructuración del pipeline de datos, la mejora de rendimiento, la validación mediante feedback, la incorporación de seguridad y el cumplimiento de los objetivos.
- El Capítulo 7, *Conclusiones y Trabajos Futuros*, sintetiza las principales conclusiones, aportaciones y limitaciones del proyecto. Además, propone líneas futuras de mejora relacionadas con nuevas visualizaciones, trazabilidad, automatización de validaciones, arquitectura de almacenamiento, permisos y validación comercial.
- Finalmente, la memoria incluye la bibliografía, donde pueden incorporarse materiales complementarios, documentación técnica, figuras adicionales o fragmentos de código relevantes.

2. Descripción de las Tecnologías

Este capítulo presenta los conceptos técnicos y metodológicos necesarios para comprender el desarrollo del dashboard analítico realizado en este Trabajo Fin de Máster. En concreto, se describen las principales tecnologías de desarrollo utilizadas, las librerías de análisis y visualización, la metodología de trabajo seguida, las herramientas de prototipado empleadas, las herramientas utilizadas para el despliegue y monitorización, y el marco teórico de modelado de datos que sirvió como referencia para estructurar la información.

2.1. Tecnologías y herramientas de desarrollo

2.1.1. Python y pandas

El desarrollo del proyecto se realizó principalmente en Python, ya que permitía integrar en un mismo entorno el procesamiento de datos, el cálculo de métricas y la construcción del dashboard. `pandas` permite trabajar con estructuras tipo `DataFrame`, que organizan la información en filas y columnas de forma similar a una tabla, pero con operaciones mucho más flexibles para filtrar, transformar, agrupar y combinar datos. Una de sus funcionalidades más importantes es `groupby`, que permite dividir los datos por una o varias variables, aplicar cálculos sobre cada grupo y recombinar los resultados en una nueva tabla agregada [4].

En este proyecto, `pandas` fue utilizado para preparar los datos antes de su visualización en el dashboard. Su uso aparece en tareas como la lectura de ficheros, la selección de columnas relevantes, la normalización de variables, la creación de agrupaciones y el cálculo de métricas agregadas. Esta librería fue especialmente importante porque el dashboard no se limita a mostrar datos brutos, sino que presenta indicadores ya transformados y orientados a preguntas de negocio, como distribución de productos, comparación entre entidades, retención de clientes o concentración de volumen.

2.1.2. Streamlit

Streamlit es un framework de Python que permite construir aplicaciones web interactivas orientadas a datos sin necesidad de desarrollar un frontend tradicional. Su principal ventaja es que permite transformar scripts de Python en aplicaciones visuales mediante una estructura sencilla, en la que los componentes de interfaz, los filtros, los gráficos y la lógica de cálculo se definen directamente desde el código [5]. Además, Streamlit incluye funcionalidades como gestión de estado de sesión y caché, relevantes para aplicaciones que necesitan mantener selecciones del usuario y evitar cálculos repetidos [6].

En este proyecto, Streamlit fue la herramienta elegida para desarrollar el dashboard final. La aplicación se estructuró en distintas páginas analíticas, cada una centrada en una dimensión del análisis, permitiendo al usuario navegar entre vistas, seleccionar entidades bancarias y comparar resultados de forma interactiva.

2.1.3. Plotly

Plotly es una librería de visualización interactiva que permite crear gráficos dinámicos en Python. A diferencia de una visualización estática, un gráfico interactivo permite al usuario explorar la información mediante tooltips, leyendas, zoom, filtros visuales o selección de elementos. Plotly ofrece gráficos de barras, líneas, mapas, heatmaps, diagramas Sankey y otras visualizaciones útiles para dashboards analíticos [7].

En el dashboard, Plotly se utilizó para representar visualmente las métricas calculadas a partir de los datos bancarios. Su función principal fue facilitar la interpretación de información agregada, permitiendo comparar entidades, productos, segmentos y comportamientos de clientes. Esta interactividad resultó relevante porque el usuario final no necesitaba únicamente consultar cifras, sino explorar patrones y contrastar información desde distintas perspectivas de negocio.

2.2. Métodologías de trabajo

Esta sección introduce la metodología utilizada durante el desarrollo del proyecto. En proyectos aplicados a empresa, especialmente cuando el objetivo es construir una herramienta analítica para usuarios reales, los requisitos iniciales rara vez están completamente definidos. Por ello, el trabajo no se planteó como un proceso cerrado desde el principio, sino como un desarrollo iterativo en el que las funcionalidades se fueron ajustando conforme aumentaba la comprensión del problema.

2.2.1. Metodología ágil e iterativa

Las metodologías ágiles se basan en el desarrollo incremental de soluciones, la revisión continua del trabajo realizado y la adaptación a nueva información. En particular, Scrum define un enfoque iterativo e incremental orientado a mejorar la predictibilidad y controlar el riesgo en contextos donde existen incertidumbre y cambios en los requisitos [8]. Esta lógica resulta especialmente adecuada en proyectos de consultoría y analítica aplicada, donde el cliente o usuario de negocio no siempre puede definir con precisión todas sus necesidades desde el inicio.

En este TFM, la forma de trabajo siguió una lógica inspirada en Scrum, organizada mediante sprints de dos semanas. Al inicio de cada sprint se definían los objetivos principales que debían abordarse durante ese periodo, priorizando las

funcionalidades o mejoras más relevantes para el avance del dashboard. Estos objetivos podían incluir el desarrollo de una nueva página, la mejora de una visualización, la corrección de una métrica, la incorporación de nuevos filtros o la optimización del rendimiento de una parte concreta de la herramienta.

Durante cada sprint, los nuevos requisitos, dudas o cambios detectados se añadían al *backlog* del proyecto. Este *backlog* funcionaba como una lista dinámica de tareas pendientes, en la que se recogían tanto nuevas necesidades de negocio como ajustes derivados del feedback recibido. De esta forma, el desarrollo no siguió una planificación rígida cerrada desde el principio, sino un proceso progresivo en el que las funcionalidades se fueron refinando conforme aumentaba la comprensión del problema y de las expectativas de los usuarios.

Esta metodología fue especialmente útil porque el dashboard se desarrolló en un contexto en el que las necesidades no estaban completamente cerradas al inicio. Primero se investigaron métricas potencialmente relevantes, después se contrastaron ideas con socios y usuarios de negocio, y posteriormente se implementaron funcionalidades de forma progresiva. A medida que se mostraban avances, surgían nuevos casos de uso, correcciones o cambios de enfoque. Por ello, el trabajo por sprints permitió priorizar, adaptar y mejorar la herramienta de manera continua, manteniendo siempre una orientación práctica hacia las necesidades reales de la empresa.

2.3. Herramientas específicas

Esta sección describe las herramientas que no forman parte directamente del código final del dashboard, pero que fueron relevantes para diseñar, validar o tomar decisiones durante el proyecto.

2.3.1. Figma

Figma es una herramienta de diseño colaborativo que permite crear prototipos visuales e interactivos de productos digitales. Sus funcionalidades de prototipado permiten simular flujos de navegación y explorar cómo interactuaría un usuario con una interfaz antes de que esta sea implementada técnicamente [9]. En proyectos de dashboard, este tipo de herramienta es útil porque permite validar estructura, jerarquía visual y experiencia de usuario sin necesidad de desarrollar todavía toda la aplicación.

En este proyecto, Figma se utilizó para construir una demo interactiva previa al desarrollo completo del dashboard. Esta demo permitió enseñar una primera propuesta visual, discutir la organización de las páginas y alinear expectativas con los usuarios de negocio. De esta forma, el diseño no surgió únicamente desde una perspectiva técnica, sino también desde la necesidad de que la herramienta fuera comprensible, navegable y útil para quienes iban a utilizarla.

2.4. Herramientas de despliegue, monitorización y gestión de secretos

Esta sección introduce las herramientas utilizadas para que el dashboard pudiera pasar de un desarrollo local a un entorno más cercano al uso operativo dentro de la empresa. Además de construir la aplicación y preparar los datos, era necesario desplegar la herramienta, monitorizar su funcionamiento y gestionar de forma segura las claves y secretos asociados a la configuración.

2.4.1. Azure App Service y Oryx Build

Azure App Service es un servicio de Azure orientado al despliegue y ejecución de aplicaciones web. En el caso de aplicaciones Python, Azure permite configurar el entorno de ejecución e instalar las dependencias necesarias para que la aplicación pueda ejecutarse en la nube [10]. Dentro de este proceso, Oryx Build es el sistema de construcción utilizado en entornos Azure para detectar el tipo de aplicación, instalar dependencias y generar los artefactos necesarios para su ejecución [11].

En este proyecto, el despliegue mediante Azure y Oryx Build permitió que el dashboard no quedara limitado a una ejecución local. Este enfoque facilitó la publicación de la aplicación en un entorno accesible para usuarios internos, reduciendo la necesidad de preparar manualmente el entorno de ejecución cada vez que se actualizaba la herramienta. De esta forma, el despliegue se integró como parte del ciclo operativo del dashboard.

2.4.2. Application Insights

Application Insights es una herramienta de monitorización de aplicaciones integrada dentro de Azure Monitor. Su función principal es recoger información de observabilidad sobre el comportamiento de una aplicación, incluyendo métricas de rendimiento, errores, disponibilidad y telemetría de uso [12]. Este tipo de herramienta resulta útil cuando una aplicación deja de ser un prototipo local y empieza a utilizarse en un entorno compartido.

En este proyecto, Application Insights se planteó como herramienta de monitorización del dashboard desplegado. Su incorporación permite disponer de una base para detectar errores, revisar problemas de rendimiento y analizar el comportamiento de la aplicación durante su uso. Además, esta monitorización se relaciona con una posible línea futura de trazabilidad, ya que permitiría registrar mejor qué ocurre cuando los usuarios interactúan con la herramienta.

2.4.3. Azure Key Vault

Azure Key Vault es un servicio de Azure diseñado para almacenar y controlar el acceso a secretos, claves y certificados [13]. Su uso permite evitar que

credenciales, contraseñas o claves de configuración queden escritas directamente en el código o en ficheros del repositorio. Esto es especialmente importante en aplicaciones empresariales que manejan información sensible o que requieren control de acceso.

En este proyecto, Key Vault se utilizó como referencia para la gestión segura de secretos asociados al despliegue y configuración del dashboard. Esta decisión refuerza la separación entre código y credenciales, facilita una gestión más segura de la configuración y contribuye a que la herramienta esté mejor preparada para un uso interno controlado dentro de la empresa.

2.5. Marcos teóricos y técnicas de modelado

Esta sección introduce los principales marcos conceptuales utilizados para organizar los datos del proyecto. En concreto, se describen dos ideas relevantes: la arquitectura medallón, utilizada para estructurar el flujo de datos en capas sucesivas, y el modelo estrella, empleado como referencia para organizar la capa analítica intermedia.

2.5.1. Arquitectura medallón

La arquitectura medallón es un patrón de diseño utilizado en entornos analíticos y lakehouse para organizar los datos en capas progresivas de calidad y preparación. Su lógica habitual distingue entre capas como *bronze*, *silver* y *gold*: la primera conserva los datos en un estado cercano al original, la segunda aplica procesos de limpieza y normalización, y la tercera contiene datos preparados para análisis, reporting o consumo de negocio [14, 15].

Este enfoque resulta útil porque separa las distintas fases del tratamiento de datos. En lugar de transformar toda la información directamente en la aplicación final, los datos avanzan por capas sucesivas, lo que permite controlar mejor la calidad, reutilizar transformaciones y reducir el coste computacional de las consultas. En proyectos de dashboard, esta separación es especialmente importante, ya que permite que la herramienta visual consuma datos previamente preparados en lugar de recalcular todas las métricas en cada interacción del usuario.

En este proyecto, la arquitectura medallón sirvió como referencia para organizar el flujo *raw* $\rightarrow$ *silver* $\rightarrow$ *gold* $\rightarrow$ *final_dashboard*. Aunque la terminología clásica suele utilizar las capas *bronze*, *silver* y *gold*, en este TFM se adapta la primera capa al nombre *raw* y se añade una capa final específica para el dashboard. Esta última capa contiene ficheros precalculados orientados al consumo directo por Streamlit, lo que permitió mejorar la latencia y facilitar la experiencia de uso.

2.5.2. Modelo estrella

El modelo estrella es un enfoque de modelado dimensional utilizado en sistemas analíticos y de inteligencia de negocio. Su estructura se basa en una tabla central de hechos, que contiene las medidas cuantitativas del proceso analizado, conectada con varias tablas de dimensiones, que contienen atributos descriptivos para interpretar esas medidas. Según Kimball, los esquemas estrella se caracterizan precisamente por conectar tablas de hechos con tablas de dimensiones mediante relaciones de claves primarias y foráneas [16]. Microsoft también destaca la relevancia de este tipo de diseño para crear modelos optimizados en términos de rendimiento y usabilidad en soluciones de BI [17].

En este proyecto, el modelo estrella sirvió como marco inicial para organizar la información bancaria dentro de la capa *gold*. La lógica consistía en separar entidades como usuarios, bancos, productos o aseguradoras de las medidas asociadas a su relación, como saldos, gastos, productos contratados o volúmenes. Esta estructura ayudó a conceptualizar los datos de forma más clara y a preparar el análisis multidimensional del dashboard. No obstante, debido a restricciones de rendimiento y a la ausencia de una base de datos analítica, posteriormente fue necesario adaptar esta estructura hacia ficheros más precalculados para mejorar la experiencia de uso.

2.6. Síntesis de la Descripción de las Tecnologías

En este capítulo se han presentado las tecnologías, métodos, herramientas y marcos conceptuales necesarios para entender el resto del trabajo. Python, pandas, Streamlit y Plotly constituyen la base tecnológica del dashboard; la metodología ágil explica la forma iterativa en que se desarrolló la solución; Figma permitió validar el diseño antes de la implementación; y las herramientas de Azure permitieron abordar el despliegue, la monitorización y la gestión segura de secretos.

También se han introducido las herramientas utilizadas para llevar la aplicación a un entorno operativo: Azure App Service y Oryx Build para el despliegue, Application Insights para la monitorización y Key Vault para la gestión de secretos. Por último, se han descrito los dos marcos principales de organización de datos utilizados en el proyecto. Por un lado, la arquitectura medallón permitió estructurar el flujo de datos en capas sucesivas de preparación, desde los datos originales hasta los ficheros finales consumidos por el dashboard. Por otro lado, el modelo estrella aportó una referencia conceptual para organizar la capa analítica intermedia mediante tablas de hechos y dimensiones.

A partir de esta base, los siguientes capítulos pueden centrarse en el estado de la cuestión, la definición del trabajo y el desarrollo del sistema sin tener que volver a explicar los conceptos técnicos fundamentales. De esta manera, el lector cuenta ya con el contexto necesario para comprender tanto las decisiones de diseño como las limitaciones técnicas que condicionaron el proyecto.

3. Estado de la Cuestión

Este capítulo analiza la situación previa y las soluciones existentes relacionadas con el proyecto. En primer lugar, se describen las formas de análisis que ya se utilizaban en el entorno de la empresa, principalmente basadas en notebooks y visualizaciones generadas bajo demanda. En segundo lugar, se presentan brevemente algunas soluciones generales de mercado orientadas a la visualización y explotación de datos, como Power BI, Tableau o Looker. Finalmente, se identifican las limitaciones que justifican el desarrollo de una herramienta propia, interactiva y accesible para usuarios de negocio.

3.1. Trabajos y soluciones existentes

3.1.1. Prácticas internas de análisis y visualización

Antes del desarrollo de este proyecto, la empresa ya disponía de datos financieros agregados procedentes de una fuente externa. Estos datos contenían información agregada sobre clientes, entidades bancarias, productos financieros y volumen asociado a cada producto en cada entidad. Por tanto, el problema principal no era la inexistencia de datos, sino la falta de una herramienta accesible que permitiera explorarlos de forma sistemática, interactiva y orientada a negocio para distintos usuarios de la empresa.

La forma habitual de responder a necesidades analíticas concretas consistía en que un perfil técnico, normalmente un ingeniero de datos, extrajera la información necesaria mediante un *Jupyter Notebook*. Este enfoque permitía realizar análisis específicos, generar visualizaciones puntuales y responder a preguntas concretas del negocio. Los notebooks son especialmente útiles para combinar código, texto explicativo y resultados visuales en un mismo documento, lo que los convierte en una herramienta adecuada para exploración, prototipado y análisis inicial [18].

Sin embargo, esta forma de trabajo presentaba una dependencia clara del equipo técnico. Cuando un usuario de negocio necesitaba una visualización o una métrica determinada, debía solicitarla a un perfil con capacidad para acceder a los datos, ejecutar el análisis y generar la salida correspondiente. Esta dinámica hacía que muchas preguntas dependieran de un proceso manual, poco escalable y difícil de reutilizar por otros usuarios.

El ingeniero encargado de estas visualizaciones desarrolló de forma interna un dashboard para agilizar la generación de las gráficas más habituales. No obstante, esta solución tenía un alcance limitado, ya que se encontraba en local y solo era accesible para la persona que la había construido. Aunque suponía un avance respecto a la generación manual de gráficos aislados, no resolvía todavía el problema de fondo: facilitar el acceso a la información y permitir que diferentes

usuarios pudieran explorar los datos sin depender constantemente de un intermediario técnico.

Este ingeniero también comenzó a desarrollar un chatbot para agilizar la preparación de propuestas y consultas internas. Sin embargo, esta iniciativa no llegó a completarse ni se planteó inicialmente como una herramienta destinada al uso general por parte de otros usuarios, sino como una forma de facilitar su propio trabajo.

3.1.2. Soluciones externas de inteligencia de negocio

En paralelo a estas soluciones internas, existen herramientas de inteligencia de negocio ampliamente utilizadas en el mercado. Power BI, por ejemplo, es la plataforma de análisis de negocio de Microsoft y permite conectar datos, crear informes, construir dashboards y compartir información dentro de una organización [19]. Su principal ventaja es la integración con el ecosistema Microsoft, lo que la convierte en una alternativa especialmente atractiva para empresas que ya trabajan con herramientas como Excel, Teams, SharePoint o Azure.

Otra solución relevante es Tableau, una plataforma de visualización e inteligencia de negocio orientada a conectar con distintas fuentes de datos, crear visualizaciones mediante una interfaz gráfica y compartir resultados con otros usuarios [20]. Del mismo modo, Looker, dentro del ecosistema de Google Cloud, ofrece funcionalidades de inteligencia de negocio, análisis embebido y definición de modelos de datos compartidos para facilitar la toma de decisiones en organizaciones [21].

Estas herramientas muestran que el mercado ya ofrece soluciones maduras para visualización y explotación de datos. Sin embargo, su existencia no elimina necesariamente la necesidad de una solución específica. En este proyecto, la decisión no dependía únicamente de elegir una herramienta estándar, sino de construir una aplicación adaptada a una estructura de datos concreta, a unas métricas bancarias específicas y a una forma de trabajo ya iniciada dentro de la empresa.

3.2. Limitaciones identificadas

3.2.1. Limitaciones de las prácticas internas existentes

La primera limitación identificada fue la dependencia de análisis manuales o semimanuales. Aunque los notebooks permitían responder a preguntas concretas, no constituían una solución adecuada para usuarios de negocio que necesitaban consultar información de forma recurrente. Un notebook exige conocimiento técnico, acceso al entorno de ejecución y capacidad para modificar o interpretar código. Por tanto, su utilidad para exploración interna no se traducía directamente en una herramienta operativa para usuarios no técnicos.

La segunda limitación fue la falta de accesibilidad. El dashboard existente antes del proyecto estaba desarrollado en local y solo podía ser utilizado por su creador. Adicionalemnte, estaba pensado solo para ayudar con las necesidades de un socio y no de todos los empleados. Esto impedía que otros usuarios consultaran las visualizaciones directamente, compararan entidades, cambiaran filtros o exploraran nuevas preguntas por sí mismos. En consecuencia, el conocimiento generado a partir de los datos quedaba concentrado en pocas personas y no se convertía en una herramienta compartida de apoyo a la toma de decisiones.

La tercera limitación tenía que ver con la interactividad. Las visualizaciones estáticas o generadas bajo demanda permiten responder a una pregunta concreta, pero no facilitan suficientemente la exploración. En un contexto de análisis bancario, el usuario puede querer comparar una entidad frente al mercado, seleccionar competidores concretos, analizar distintos productos, observar segmentos de clientes o estudiar patrones de retención y crecimiento. Estas necesidades requieren una herramienta interactiva, no únicamente gráficos puntuales.

3.2.2. Limitaciones de las soluciones externas en el contexto del proyecto

Para hablar de las limitaciones en las soluciones externas, estas eran la adaptación al caso de uso concreto. Herramientas como Power BI, Tableau o Looker ofrecen capacidades avanzadas de visualización y compartición, pero no siempre encajan directamente con una solución ya iniciada en Python ni con una arquitectura de datos preparada específicamente para cálculos propios. En este caso, ya existía trabajo previo en Streamlit dentro de la empresa, por lo que continuar con una solución personalizada permitía aprovechar ese punto de partida y adaptar la herramienta a las necesidades concretas del proyecto.

Finalmente, también existía una limitación de rendimiento. La explotación de datos agregados de clientes, productos y entidades podía generar problemas de latencia si las métricas se calculaban de forma dinámica cada vez que el usuario interactuaba con la aplicación. Por ello, el proyecto no solo debía crear una interfaz visual, sino también organizar los datos de manera que el dashboard pudiera responder con suficiente rapidez para ser utilizado en un entorno real.

3.3. Síntesis del estado de la cuestión

El análisis anterior muestra que el punto de partida no era la ausencia de datos ni la inexistencia total de visualizaciones. La empresa ya disponía de una fuente de datos financieros agregados y de capacidad técnica para analizarlos mediante notebooks. Además, algunas visualizaciones frecuentes ya habían sido agrupadas en un dashboard local. Sin embargo, estas soluciones no resolvían completamente las necesidades de accesibilidad, autonomía, interactividad, reutilización y escalabilidad.

Por otro lado, el mercado ofrece herramientas consolidadas de inteligencia de negocio, como Power BI, Tableau o Looker, que permiten crear dashboards y compartir información dentro de una organización. No obstante, la elección de una solución no puede evaluarse únicamente desde la existencia de estas plataformas. En este caso, la empresa ya contaba con trabajo previo en Streamlit y el proyecto requería una herramienta adaptada a métricas y datos específicos, lo que justificaba continuar con una aplicación personalizada.

El hueco que cubre este TFM se sitúa precisamente entre ambos extremos. Por un lado, supera la lógica de análisis aislado mediante notebooks y visualizaciones locales. Por otro lado, no se limita a implantar una herramienta genérica de BI, sino que desarrolla un dashboard específico, orientado a los datos y preguntas de negocio de la empresa. La solución propuesta busca transformar datos ya disponibles en una herramienta accesible, interactiva y útil para usuarios de negocio, reduciendo la dependencia de análisis manuales y facilitando una exploración más autónoma de la información.

Esta situación justifica el desarrollo del proyecto descrito en los capítulos siguientes. El TFM parte de una necesidad real: convertir datos financieros agregados en una aplicación analítica funcional, desplegable y alineada con la forma de trabajo de la empresa. A partir de esta necesidad se definen los objetivos, la metodología y la arquitectura del sistema desarrollado.

4. Definición del Trabajo

Este capítulo delimita el trabajo realizado en el proyecto. A partir de la situación descrita en el estado de la cuestión, se justifica la necesidad de desarrollar un dashboard analítico propio, se definen los objetivos generales y específicos, se explica la metodología seguida y se presenta una planificación temporal aproximada del desarrollo.

El proyecto se plantea como una solución aplicada a un contexto empresarial real, donde ya existían datos financieros agregados y ciertos análisis previos, pero no una herramienta suficientemente accesible, interactiva y orientada a usuarios de negocio. Por tanto, el objetivo de este capítulo es concretar qué problema se pretende resolver, qué alcance tiene la solución desarrollada y cómo se organiza el trabajo para alcanzarla.

4.1. Justificación

El desarrollo de este TFM se justifica por la necesidad de transformar datos financieros agregados, ya disponibles en la empresa, en una herramienta de análisis accesible, reutilizable e interactiva. Antes del proyecto, la explotación de estos datos dependía en gran medida de análisis realizados por perfiles técnicos mediante notebooks o herramientas locales. Aunque estas soluciones permitían responder a preguntas concretas, no facilitaban una consulta autónoma, recurrente y estructurada por parte de los usuarios de negocio.

Frente a esta situación, el proyecto aporta valor porque convierte un proceso principalmente manual en una aplicación analítica desplegable, organizada por módulos y orientada a la toma de decisiones. El dashboard permite explorar métricas sobre entidades bancarias, productos financieros, clientes, volúmenes, retención, estabilidad, crecimiento y comportamiento frente a neobancos, reduciendo la dependencia de peticiones individuales al equipo técnico.

Además, la herramienta no se plantea únicamente como una solución interna de visualización, sino también como una forma de mostrar el potencial comercial de los datos disponibles. Al permitir comparar entidades bancarias frente al mercado y frente a competidores, el dashboard puede servir como base para presentar a otros bancos posibles casos de uso de estos datos agregados. De esta forma, el proyecto ayuda a convertir una fuente de información ya existente en una herramienta demostrable, interpretable y potencialmente comercializable.

4.1.1. Necesidad de democratizar el acceso a la información

El primer argumento de justificación es la necesidad de facilitar el acceso a la información dentro de la empresa. Aunque los datos ya estaban disponibles, su

consulta requería conocimiento técnico, acceso al entorno de análisis y capacidad para ejecutar código. Esto hacía que muchas preguntas de negocio dependieran de una persona concreta, en lugar de poder resolverse directamente mediante una herramienta compartida.

El dashboard desarrollado busca reducir esta barrera. Al presentar los datos mediante una interfaz visual e interactiva, los usuarios pueden consultar métricas, comparar entidades, analizar productos y explorar distintos escenarios sin necesidad de modificar notebooks ni solicitar cada visualización de forma individual. De esta forma, el proyecto contribuye a convertir los datos existentes en una herramienta de uso más autónomo y recurrente.

4.1.2. Necesidad de homogeneizar métricas y cálculos

El segundo argumento de justificación está relacionado con la consistencia de las métricas y la fiabilidad de los cálculos. Cuando los análisis se generan de forma aislada, existe el riesgo de que los criterios de cálculo varíen entre visualizaciones o de que una misma métrica se interprete de forma distinta según el contexto. En un proyecto basado en datos financieros agregados, esta falta de homogeneidad puede dificultar la comparación entre entidades, productos o segmentos.

El dashboard permite centralizar las métricas principales en una misma aplicación, aplicando criterios de cálculo homogéneos y una estructura común de visualización. Esto resulta especialmente relevante para indicadores como productos por cliente, volumen asociado a cada entidad, retención, estabilidad, concentración o posibles oportunidades de crecimiento. La herramienta no solo muestra datos, sino que organiza la información bajo una lógica analítica común y reduce la probabilidad de inconsistencias entre análisis.

4.1.3. Necesidad de mostrar el valor comercial de los datos

El tercer argumento de justificación es la posibilidad de utilizar el dashboard como una herramienta demostrativa para terceros, especialmente entidades bancarias interesadas en conocer su posición relativa frente al mercado. Los datos agregados permiten analizar cómo se distribuyen productos, volúmenes y comportamientos entre bancos, lo que puede aportar valor a entidades que quieran entender mejor su situación competitiva.

En este sentido, el dashboard no solo resuelve una necesidad interna de análisis, sino que también permite enseñar de forma clara las posibles utilidades de los datos disponibles. Al tener cálculos prehechos, visualizaciones organizadas y comparaciones ya estructuradas, la herramienta facilita mostrar casos de uso concretos sin tener que construir cada análisis desde cero. Esto puede ser útil tanto para conversaciones internas como para presentar el potencial de la solución a otras entidades.

4.1.4. Necesidad de una solución adaptada al caso de uso

El cuarto argumento de justificación es la necesidad de construir una solución adaptada a los datos, restricciones y forma de trabajo de la empresa. Aunque existen herramientas externas de inteligencia de negocio, el proyecto requería una aplicación ajustada a una estructura de datos concreta, a métricas específicas y a un desarrollo previo ya iniciado en Streamlit.

Además, el proyecto debía responder a restricciones prácticas, especialmente en términos de rendimiento y seguridad. Al no trabajar con una base de datos analítica tradicional, fue necesario organizar los datos de forma que el dashboard pudiera responder con rapidez. Del mismo modo, al tratarse de información sensible, era necesario incorporar mecanismos de control de acceso para que únicamente personas autorizadas pudieran consultar la herramienta.

4.2. Objetivos

El objetivo general del proyecto es desarrollar un dashboard analítico interactivo que permita a usuarios autorizados explorar datos financieros agregados de forma accesible, estructurada y útil para la toma de decisiones.

A partir de este objetivo general, se definen los siguientes objetivos específicos:

- **O1.** Diseñar una herramienta visual e interactiva que permita consultar métricas relevantes sobre entidades bancarias, clientes, productos financieros, volúmenes, retención, estabilidad y crecimiento.
- **O2.** Reducir la dependencia de análisis manuales o semimanuales realizados por perfiles técnicos, sustituyendo visualizaciones aisladas por una aplicación accesible y reutilizable.
- **O3.** Homogeneizar el cálculo y presentación de las métricas principales, de forma que los usuarios puedan comparar entidades, productos y segmentos bajo criterios consistentes.
- **O4.** Comprobar que los datos y métricas calculadas sean coherentes, interpretables y adecuados para el análisis de negocio, incorporando revisiones y correcciones cuando se detecten inconsistencias.
- **O5.** Desarrollar una solución basada en Streamlit que aproveche el trabajo previo existente en la empresa y permita una evolución flexible mediante nuevas páginas, filtros y visualizaciones.
- **O6.** Diseñar y adaptar la arquitectura de datos necesaria para mejorar el rendimiento del dashboard, reduciendo cálculos pesados en tiempo real y favoreciendo el uso de datos precalculados.
- **O7.** Incorporar mecanismos de seguridad y control de acceso para que únicamente personas autorizadas dentro de la empresa puedan acceder a la información disponible en el dashboard.
- **O8.** Incorporar feedback de usuarios reales durante el desarrollo, priorizando

mejoras funcionales, visuales y de rendimiento según las necesidades detectadas.

O9. Desplegar una versión funcional del dashboard en una fase temprana del proyecto, con el objetivo de disponibilizar la herramienta, facilitar su uso por parte de usuarios reales y recoger feedback operativo.

O10. Explorar el potencial del dashboard como herramienta demostrativa para mostrar a otras entidades bancarias posibles utilidades de los datos agregados y comparaciones frente al mercado.

4.3. Metodología

La metodología seguida combinó investigación inicial, diseño de producto, desarrollo iterativo y validación con usuarios. El proyecto no partió de unos requisitos completamente cerrados, sino de una necesidad general: transformar datos financieros agregados en una herramienta de análisis útil para usuarios de negocio. Por ello, fue necesario adoptar una forma de trabajo flexible, capaz de incorporar cambios y nuevos requisitos a medida que avanzaba el desarrollo.

En una primera fase, se realizó una investigación de métricas potencialmente relevantes. El objetivo era identificar qué indicadores podían aportar valor a partir de los datos disponibles. Esta fase permitió estudiar posibles enfoques de análisis relacionados con productos financieros, volumen por entidad, retención, estabilidad, crecimiento, concentración y comportamiento de clientes.

Posteriormente, se mantuvieron conversaciones con socios y usuarios de negocio para contrastar estas ideas y entender mejor las necesidades reales de la empresa. Esta fase fue clave porque permitió pasar de una visión puramente técnica a una visión orientada a uso. Las conversaciones ayudaron a priorizar qué preguntas debía responder el dashboard y qué tipo de información resultaba más útil para los usuarios finales.

Tras esta validación inicial, se diseñó una demo interactiva en Figma. El objetivo del prototipo era representar de forma visual la estructura del dashboard antes de desarrollar completamente la aplicación. Esta demo permitió discutir la navegación, la organización de las páginas, la jerarquía visual y el tipo de información que debía aparecer en cada módulo.

A continuación, comenzó la implementación progresiva del dashboard en Streamlit. El trabajo se organizó en sprints de dos semanas. Al inicio de cada sprint se definían los objetivos principales que debían abordarse, como el desarrollo de una página, la mejora de una visualización, la incorporación de nuevos filtros, la corrección de una métrica o la optimización de rendimiento. Durante el sprint, los nuevos requisitos, dudas o cambios detectados se añadían al *backlog* del proyecto para ser revisados y priorizados posteriormente.

El desarrollo se organizó en cuatro sprints principales. Los dos primeros se centraron en construir la versión inicial del dashboard y desplegarla para que pudiera estar disponible cuanto antes. Este despliegue temprano no se planteó

como el cierre del proyecto, sino como una forma de permitir que usuarios reales interactuaran con la herramienta y pudieran aportar feedback desde una fase inicial.

Los dos sprints siguientes se centraron en la incorporación de mejoras derivadas del uso real. Entre estas mejoras se incluyeron ajustes en visualizaciones, revisión de cálculos, comprobación de coherencia de métricas, incorporación de nuevos requisitos, optimización de latencia, unificación de páginas y desarrollo de mecanismos de seguridad mediante usuario y contraseña. De esta forma, la herramienta fue evolucionando no solo a partir de una planificación inicial, sino también a partir de necesidades detectadas durante su uso.

Además del desarrollo visual, la metodología incluyó una fase de adaptación de la arquitectura de datos. Inicialmente, los datos se estructuraron siguiendo una lógica más cercana a un modelo estrella, pero esta solución generaba problemas de latencia al no trabajar con una base de datos analítica tradicional. Por ello, se evolucionó hacia una capa de datos más precalculada y orientada al consumo directo por el dashboard. Esta decisión permitió reducir cálculos en tiempo real y mejorar la experiencia de usuario.

Finalmente, el proyecto incluyó tareas de validación operativa y mejora continua. La herramienta fue utilizada por usuarios reales, lo que permitió detectar nuevas necesidades y realizar ajustes adicionales. Este enfoque permitió que el dashboard no fuera únicamente un prototipo técnico, sino una solución funcional alineada con los requisitos prácticos de la empresa y con posibles usos comerciales futuros.

4.4. Planificación

La planificación del proyecto se organizó en fases sucesivas, aunque con una lógica iterativa. La preparación inicial tuvo una duración aproximada de un mes, incluyendo la investigación de métricas, la definición preliminar del alcance, las conversaciones con usuarios de negocio y el diseño de la demo inicial. Posteriormente, el desarrollo principal se planificó en cuatro sprints de dos semanas, con una duración total aproximada de dos meses.

La Tabla 4.1 resume la planificación temporal del trabajo.

Fase	Descripción	Duración estimada
Fase 1	Investigación inicial de métricas, análisis del punto de partida y definición preliminar del alcance	2 semanas
Fase 2	Reuniones con usuarios de negocio, validación de necesidades y diseño de demo interactiva en Figma	1 semana
Sprint 1	Desarrollo inicial de la estructura del dashboard, navegación principal y primeras visualizaciones	2 semanas
Sprint 2	Desarrollo de páginas principales y despliegue temprano del dashboard para disponibilizar la herramienta y recibir feedback	2 semanas
Sprint 3	Incorporación de feedback, revisión de cálculos, ajustes de visualizaciones y mejora de coherencia de métricas	2 semanas
Sprint 4	Preparación del código para facilitar su uso por otros usuarios, actualización con nuevos datos, documentación del flujo de ejecución y ajustes finales derivados del uso real	2 semanas

Tabla 4.1: Planificación temporal aproximada del trabajo.

4.5. Síntesis del capítulo

En este capítulo se ha delimitado el alcance del proyecto y se ha explicado por qué el desarrollo del dashboard resulta necesario dentro del contexto de la empresa. La justificación principal parte de una situación en la que ya existían datos financieros agregados y ciertos análisis previos, pero no una herramienta suficientemente accesible, interactiva y reutilizable para usuarios de negocio.

También se han definido los objetivos del TFM, centrados en construir una herramienta visual que permita consultar métricas relevantes, reducir la dependencia de análisis manuales, homogeneizar cálculos, mejorar el rendimiento, incorporar mecanismos de seguridad y facilitar el uso del dashboard tanto para análisis internos como para posibles demostraciones del valor de los datos a otras entidades.

La metodología descrita muestra que el proyecto no siguió un desarrollo cerrado desde el inicio, sino un proceso iterativo basado en investigación inicial, validación con usuarios, diseño de una demo, desarrollo por sprints, despliegue temprano y mejora continua a partir del feedback recibido. Esta forma de trabajo permitió adaptar la herramienta a necesidades reales que fueron apareciendo durante el proyecto.

Finalmente, la planificación temporal resume las principales fases del trabajo,

desde la preparación inicial hasta los sprints de desarrollo, despliegue, revisión de cálculos, incorporación de feedback y preparación del código para futuras actualizaciones. Con esta definición del trabajo, los siguientes capítulos pueden centrarse ya en describir con mayor detalle la arquitectura, el diseño y la implementación del sistema desarrollado.

5. Sistema Desarrollado

Este capítulo describe el sistema desarrollado en el TFM. En él se presentan los requisitos que debía cumplir el dashboard, los actores que intervienen, los datos utilizados, la arquitectura diseñada y la forma en que se implementó la herramienta.

El sistema desarrollado consiste en un dashboard analítico interactivo construido en Streamlit, orientado a facilitar la exploración de datos financieros agregados. La herramienta permite a usuarios internos consultar métricas sobre entidades bancarias, productos financieros, clientes, volúmenes, penetración, concentración, oportunidades de venta cruzada y salidas hacia neobancos. Además, el sistema se diseñó para reducir la dependencia de análisis manuales, homogeneizar cálculos, mejorar la accesibilidad a los datos y permitir que la herramienta pudiera actualizarse con nuevos datos en el futuro.

5.1. Análisis del Sistema

El análisis del sistema parte de una necesidad concreta, la empresa disponía de datos financieros agregados, pero su explotación dependía principalmente de análisis bajo demanda realizados por perfiles técnicos. Aunque existían notebooks y algunas visualizaciones previas, no había una herramienta suficientemente accesible, interactiva y preparada para que distintos usuarios internos pudieran explorar la información de forma autónoma.

Por tanto, el sistema debía cumplir una doble función. En primer lugar, debía servir como herramienta interna de análisis para usuarios de negocio, permitiendo consultar indicadores ya calculados sin necesidad de ejecutar código. En segundo lugar, debía servir como herramienta demostrativa para mostrar posibles usos de los datos a entidades bancarias interesadas en conocer su posición relativa frente a otros bancos, grupos de bancos o segmentos concretos del mercado.

5.1.1. Actores del sistema

Los principales actores identificados en el sistema son los siguientes:

- **Usuarios internos de negocio**: perfiles de la empresa que necesitan consultar métricas, comparar entidades, interpretar resultados y utilizar el dashboard como apoyo para proyectos, análisis internos o presentaciones.
- **Equipo técnico**: perfiles encargados de preparar los datos, mantener el código, revisar cálculos, actualizar ficheros y garantizar que el dashboard pueda seguir utilizándose con nuevos datos.
- **Potenciales clientes externos**: entidades bancarias o perfiles externos a los que se podría presentar la herramienta como propuesta de valor, especialmente

para mostrar comparativas frente a otros bancos, grupos de bancos o patrones agregados de comportamiento financiero.

5.1.2. Datos de entrada

Los datos de entrada proceden de una fuente externa de datos financieros agregados. Estos datos contienen información sobre clientes, entidades bancarias, productos financieros y volumen asociado a cada producto en cada entidad. La información llega como base agregada para construir indicadores analíticos.

A partir de estos datos se generan distintas capas de procesamiento. En primer lugar, una capa inicial limpia y normaliza la información disponible. Posteriormente, los datos se estructuran siguiendo una lógica analítica, separando entidades como usuarios, bancos y productos. Finalmente, se generan ficheros optimizados para el consumo directo por parte del dashboard.

Una limitación relevante de los datos disponibles es que no se contaba con un histórico completo para todas las métricas utilizadas en el dashboard. Esta ausencia de histórico condicionó el tipo de análisis que podía realizarse, ya que impedía construir algunas métricas evolutivas o comparaciones temporales más profundas. Por ello, buena parte del sistema se orientó a explotar la información agregada disponible en un momento determinado, priorizando comparativas entre entidades, productos, perfiles de clientes y distribución de volumen.

Además, la información relativa a salidas hacia neobancos procedía de la misma fuente externa, pero fue recibida como un conjunto de datos separado respecto al utilizado en el resto de páginas del dashboard. Esto hizo necesario tratar esta parte como un módulo específico dentro de la herramienta, con una lógica de procesamiento y visualización propia.

5.1.3. Datos de salida

La salida principal del sistema es una aplicación web interactiva que presenta métricas y visualizaciones organizadas por módulos. Entre las salidas más relevantes se encuentran:

- comparativas entre una entidad seleccionada y otros bancos o grupos de bancos;
- una vista general del banco seleccionado mediante KPIs principales;
- análisis de penetración de clientes principales, *share of wallet* y riesgo de concentración de cartera;
- distribución de clientes principales por edad, renta y localización geográfica;
- métricas filtrables sobre perfiles de clientes principales frente a otros bancos;
- análisis de productos financieros, ticket medio, penetración y distribución del volumen;
- identificación de posibles oportunidades de venta cruzada;

- análisis de penetración y concentración de productos por cliente;
- análisis de concentración de volumen en determinados percentiles de usuarios;
- visualizaciones sobre dinero enviado hacia neobancos, frecuencia de operaciones, evolución disponible y distribución por entidad receptora;
- tablas y gráficos interactivos para facilitar la exploración de los datos.

Estas salidas están diseñadas para que los usuarios internos puedan interpretar la información sin necesidad de acceder directamente a los ficheros originales ni modificar código. En el caso de potenciales usuarios externos, el objetivo es ofrecer una visión comparativa de la posición de su entidad frente a otros bancos o grupos de bancos, identificar áreas estratégicas de mejora y extraer insights a partir de información agregada que normalmente no estaría disponible dentro de sus propios sistemas internos.

5.1.4. Requisitos funcionales

A partir de las necesidades identificadas, se definieron los siguientes requisitos funcionales:

RF1. Permitir la selección de una entidad bancaria principal para realizar el análisis.

RF2. Permitir la comparación de la entidad seleccionada frente a otros bancos o frente a un grupo de bancos.

RF3. Mostrar una vista general con los principales KPIs de la entidad seleccionada, incluyendo penetración de clientes principales, share of wallet y concentración de cartera.

RF4. Mostrar la distribución de clientes principales por variables como edad, renta y geografía.

RF5. Permitir el análisis comparativo de perfiles de clientes principales frente a otros bancos mediante métricas filtrables.

RF6. Incorporar análisis de productos financieros, incluyendo penetración, ticket medio y distribución del volumen.

RF7. Identificar posibles oportunidades de venta cruzada a partir de los productos y perfiles analizados.

RF8. Incorporar un módulo de penetración y concentración que permita estudiar productos por cliente principal, distribución del volumen y concentración en determinados percentiles de usuarios.

RF9. Incorporar un módulo específico para analizar salidas hacia neobancos, incluyendo volumen, frecuencia, evolución disponible y distribución por entidad receptora.

RF10. Presentar visualizaciones interactivas que permitan explorar la información de forma intuitiva.

RF11. Utilizar datos ya procesados o precalculados para reducir tiempos de carga.

RF12. Incorporar mecanismos de seguridad para limitar el acceso a personas de la empresa.

RF13. Permitir que el flujo de datos pueda actualizarse con nuevos ficheros en el futuro.

RF14. Permitir que usuarios puedan dar feedback para posibles mejoras en el futuro del dashboard.

RF15. Facilitar que otros usuarios técnicos puedan ejecutar, mantener o adaptar el dashboard sin depender exclusivamente de la persona que lo desarrolló inicialmente.

5.1.5. Requisitos no funcionales

Además de los requisitos funcionales, el sistema debía cumplir una serie de requisitos no funcionales:

RNF1. **Rendimiento**: el dashboard debía responder con suficiente rapidez para ser utilizado de forma práctica por usuarios reales.

RNF2. **Usabilidad**: la interfaz debía ser clara, visual y comprensible para usuarios de negocio no técnicos.

RNF3. **Mantenibilidad**: el código debía organizarse de forma que pudiera ser entendido, actualizado y utilizado por otras personas.

RNF4. **Escalabilidad funcional**: la solución debía permitir añadir nuevas páginas, métricas o visualizaciones en el futuro.

RNF5. **Seguridad**: el acceso debía estar limitado a personas de la empresa, dado el carácter sensible de la información analizada.

RNF6. **Consistencia**: las métricas debían calcularse de forma homogénea para evitar diferencias entre análisis o visualizaciones.

RNF7. **Actualización**: el flujo de procesamiento debía estar preparado para poder regenerar los datos del dashboard cuando se recibieran nuevos ficheros.

5.1.6. Casos de uso principales

Los casos de uso del sistema pueden agruparse según el tipo de actor que interactúa con la herramienta. Esta división permite distinguir entre el uso interno del dashboard para análisis y proyectos, el uso técnico para mantenimiento y actualización, y el uso potencial como herramienta demostrativa para clientes externos.

5.1.6.1. Usuarios internos de negocio

Los usuarios internos de negocio utilizan el dashboard como herramienta de consulta, análisis y apoyo para proyectos o presentaciones. Sus principales casos de uso son los siguientes:

CU1. **Consultar la situación general de una entidad**: el usuario selecciona una entidad bancaria y consulta sus principales KPIs, como penetración de clientes principales, *share of wallet* y concentración de cartera.

CU2. **Comparar con otros bancos o grupos de bancos**: el usuario compara la entidad seleccionada frente a una entidad concreta o frente a un grupo de bancos para analizar diferencias en clientes, productos, volumen o comportamiento.

CU3. **Analizar el perfil de clientes principales**: el usuario estudia cómo se distribuyen los clientes principales por edad, renta y geografía.

CU4. **Comparar perfiles de clientes**: el usuario filtra métricas sobre clientes principales y compara los resultados frente a otros bancos.

CU5. **Analizar productos financieros**: el usuario revisa la penetración, ticket medio y distribución del volumen por producto.

CU6. **Identificar oportunidades de venta cruzada**: el usuario analiza productos y segmentos donde pueden existir oportunidades comerciales.

CU7. **Analizar penetración y concentración**: el usuario estudia cómo se distribuyen los productos por cliente y cuánto volumen se concentra en determinados percentiles de usuarios.

CU8. **Analizar salidas hacia neobancos**: el usuario explora cuánto dinero se dirige hacia neobancos, con qué frecuencia, cómo evoluciona en los datos disponibles y hacia qué entidades se concentra.

CU9. **Preparar análisis o presentaciones internas**: el usuario utiliza las visualizaciones y métricas del dashboard como apoyo para proyectos, reuniones o propuestas dentro de la empresa.

5.1.6.2. Equipo técnico

El equipo técnico utiliza el sistema desde una perspectiva de mantenimiento, actualización y control de calidad. Sus casos de uso principales son los siguientes:

CU10. **Actualizar datos**: un usuario técnico sustituye o incorpora nuevos datos y ejecuta el flujo de procesamiento para generar los ficheros actualizados del dashboard.

CU11. **Ejecutar scripts de procesamiento**: el usuario técnico ejecuta los scripts necesarios para limpiar, transformar y preparar los datos antes de su consumo por la aplicación.

CU12. **Validar cálculos**: el usuario técnico revisa que las métricas generadas sean coherentes, estén correctamente calculadas y respondan a la lógica de

negocio definida.

CU13. Mantener el dashboard: el usuario técnico corrige errores, ajusta visualizaciones, mejora el rendimiento y asegura que la herramienta siga funcionando correctamente.

CU14. Preparar el código para futuras actualizaciones: el usuario técnico organiza el código y documenta el flujo de ejecución para que otras personas puedan utilizar o actualizar el dashboard en el futuro.

5.1.6.3. Potenciales clientes externos

Los potenciales clientes externos no interactúan con el sistema como responsables de mantenimiento, sino como destinatarios de una propuesta de valor basada en los datos agregados. Sus principales casos de uso son los siguientes:

CU15. Visualizar la propuesta de valor: el cliente externo observa cómo el dashboard permite transformar datos agregados en información interpretable y accionable.

CU16. Analizar su posicionamiento frente a otros bancos: el cliente externo puede entender cómo se sitúa su entidad en comparación con otros bancos o grupos de bancos.

CU17. Identificar áreas estratégicas de mejora: el cliente externo puede detectar diferencias relevantes en productos, clientes, volumen, penetración o concentración que ayuden a orientar decisiones comerciales.

CU18. Explorar posibles utilidades de los datos agregados: el cliente externo puede comprender qué tipo de análisis podrían realizarse con información a la que normalmente no tendría acceso directo desde sus propios sistemas internos.

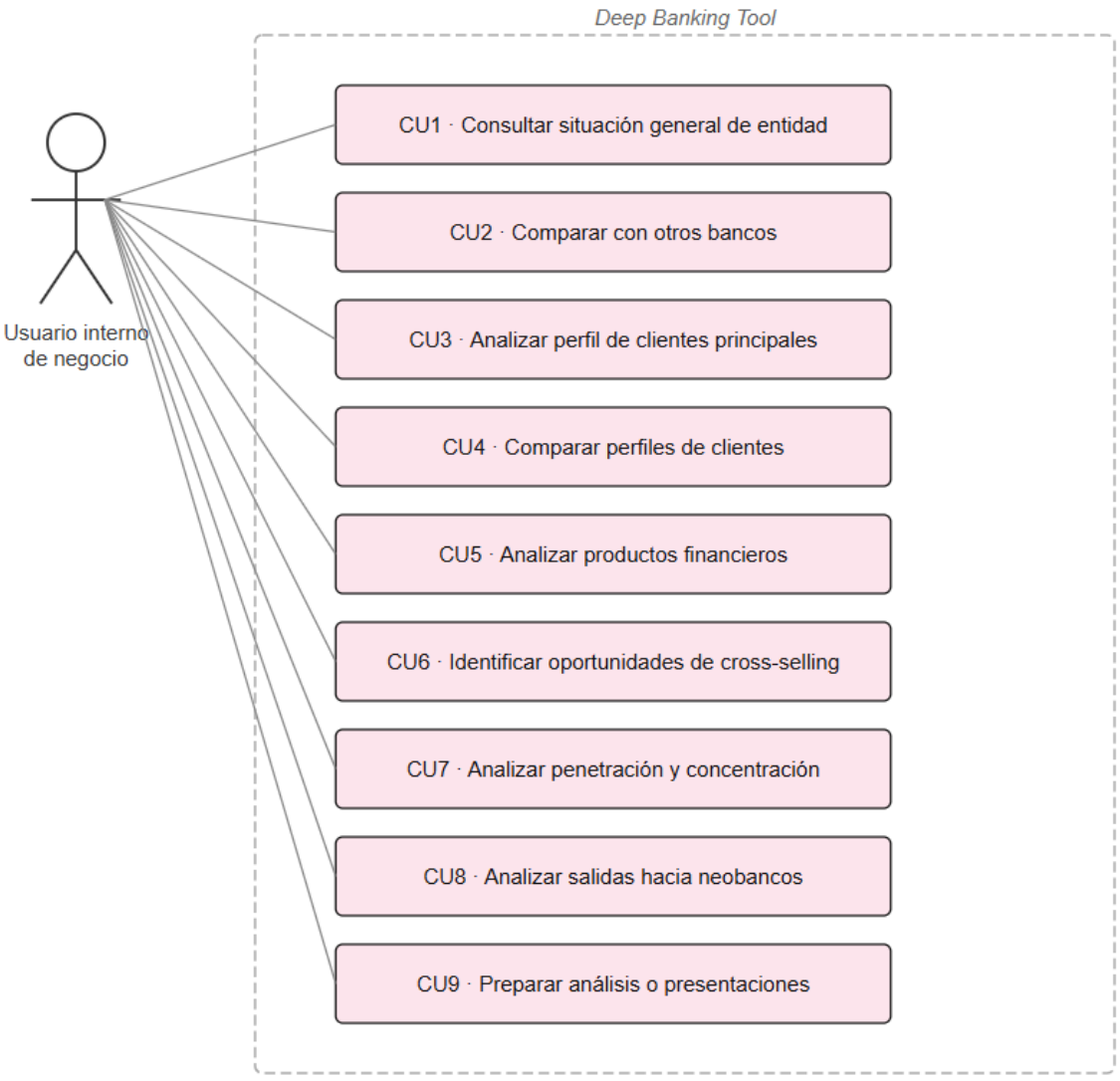

Figura 5.1: Casos de uso de los usuarios internos de negocio.

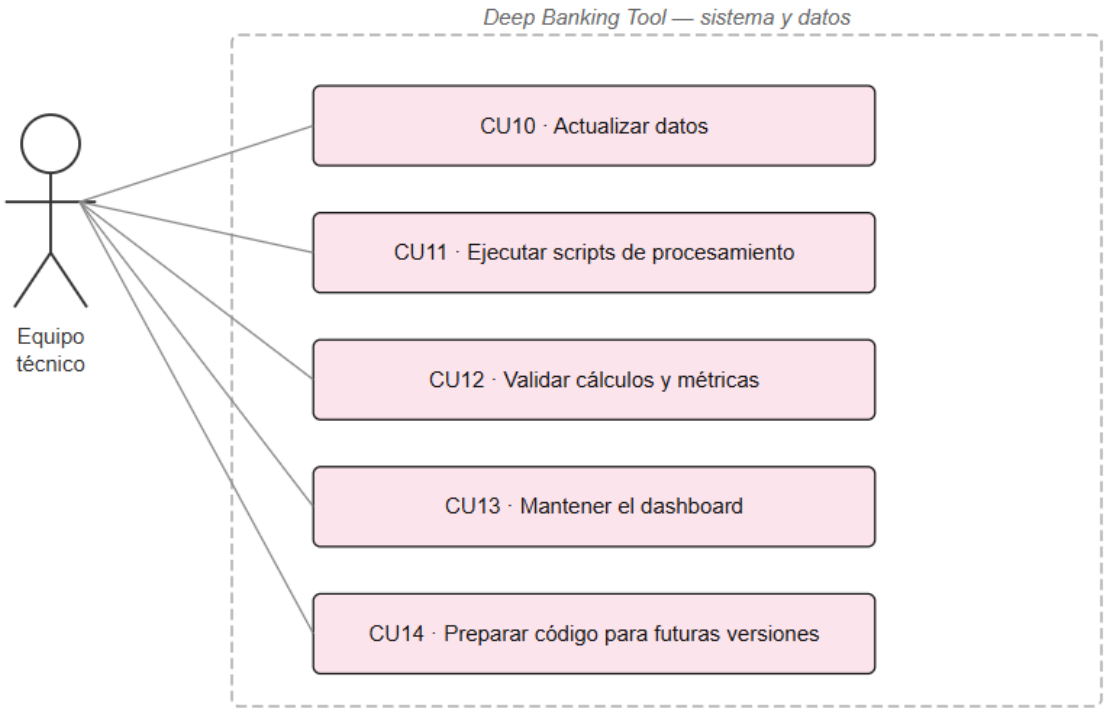

Figura 5.2: Casos de uso del equipo técnico.

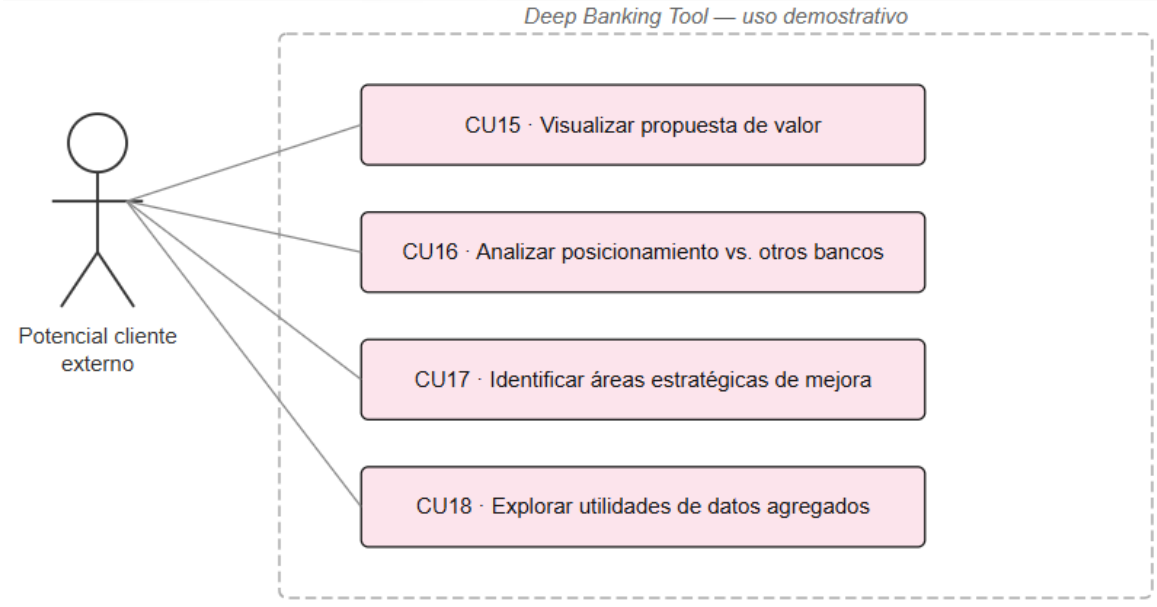

Figura 5.3: Casos de uso de potenciales clientes externos.

5.2. Diseño

El diseño del sistema se planteó a partir de tres decisiones principales. La primera fue construir una herramienta visual e interactiva en Streamlit, aprovechando el trabajo previo existente en la empresa. La segunda fue organizar

el dashboard por páginas o módulos analíticos, de forma que cada sección respondiera a una pregunta de negocio distinta. La tercera fue adaptar la arquitectura de datos para reducir la latencia, pasando de una estructura analítica inicial a ficheros finales precalculados para cada módulo del dashboard.

5.2.1. Arquitectura general del sistema

La arquitectura general del sistema muestra a alto nivel, cómo los datos pasan desde su origen hasta convertirse en visualizaciones dentro del dashboard.

El sistema parte de una fuente externa de datos financieros agregados. Estos datos llegan como ficheros de entrada y no se muestran directamente al usuario, ya que antes deben ser limpiados, transformados y adaptados a las métricas que necesita el dashboard. Para ello, se utilizan scripts de Python que preparan la información, calculan indicadores y generan ficheros finales optimizados.

Una decisión importante del diseño fue separar el procesamiento de datos de la interacción del usuario con la aplicación. Las operaciones más pesadas, como la limpieza, transformación y cálculo de métricas, se realizan antes de que el usuario utilice el dashboard. De esta forma, cuando el usuario accede a la herramienta, Streamlit no tiene que recalcular toda la información desde cero, sino que carga datos ya preparados y se centra en aplicar filtros, actualizar gráficos y mostrar resultados.

Esta arquitectura permite mejorar el rendimiento del sistema y hacer que el dashboard sea más usable en un contexto real. Además, facilita el mantenimiento futuro, ya que el equipo técnico puede actualizar los datos ejecutando de nuevo el flujo de procesamiento y regenerando los ficheros finales que consume la aplicación.

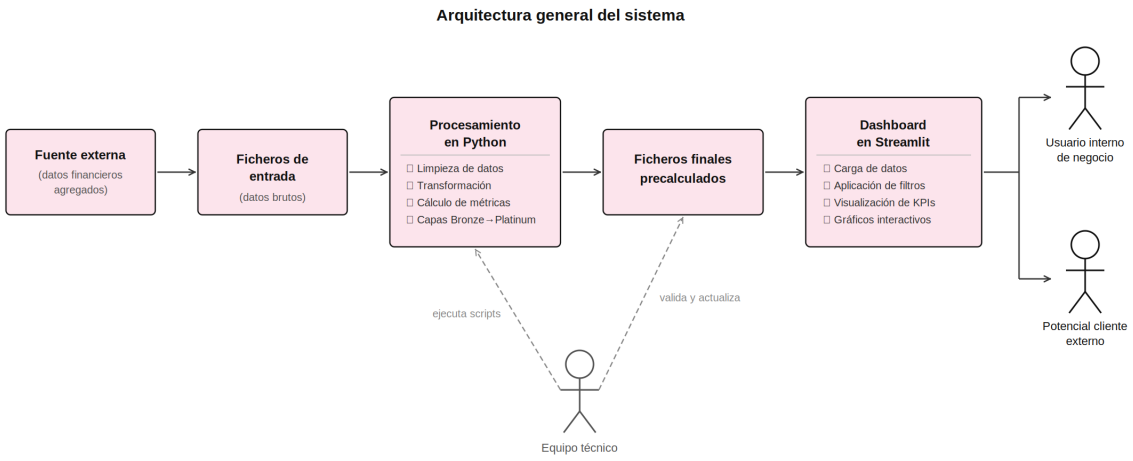

Figura 5.4: Arquitectura general del sistema desarrollado.

5.2.2. Diseño de la navegación del dashboard

El dashboard se diseñó como una aplicación multipágina. Esta decisión permite dividir el análisis en módulos diferenciados y evitar que toda la información aparezca concentrada en una única pantalla. Cada página responde a un bloque de análisis concreto y ayuda al usuario a navegar de forma ordenada por la herramienta.

La estructura principal incluye una página de inicio, una página de configuración y tres grandes páginas analíticas: *Visión General*, *Penetración y Concentración* y *Salidas hacia Neobancos*. La página de configuración permite seleccionar la entidad bancaria y definir el contexto de comparación, ya sea frente a otro banco concreto o frente a un grupo de bancos. A partir de esa selección, el resto de páginas muestran métricas y visualizaciones adaptadas a la entidad elegida.

La página de *Visión General* concentra la primera lectura del banco seleccionado. En ella se incluyen los KPIs principales, la distribución de clientes y volumen por renta, edad y provincia, el perfil medio del cliente, la distribución de productos por volumen, el análisis de productos y las oportunidades de venta cruzada. Por tanto, el análisis de crecimiento no se presenta como una página independiente, sino como parte de la visión general de la entidad.

La página de *Penetración y Concentración* agrupa dos dimensiones relacionadas. Por un lado, permite estudiar la penetración de clientes principales. Por otro lado, permite analizar la concentración de volumen, los percentiles de saldo y el perfil de los clientes asociados a dichos percentiles. Esta página ayuda a entender tanto el grado de vinculación de los clientes con la entidad como la distribución del volumen entre distintos grupos de usuarios.

Finalmente, la página de *Salidas hacia Neobancos* se diseñó como un módulo específico, ya que utiliza un conjunto de datos separado respecto al resto del dashboard. Esta página incluye visualizaciones sobre la evolución del volumen transferido a neobancos por usuario, la matriz de salida de volumen transaccional mensual hacia neobancos y la proporción de clientes totales que realizan transferencias a neobancos.

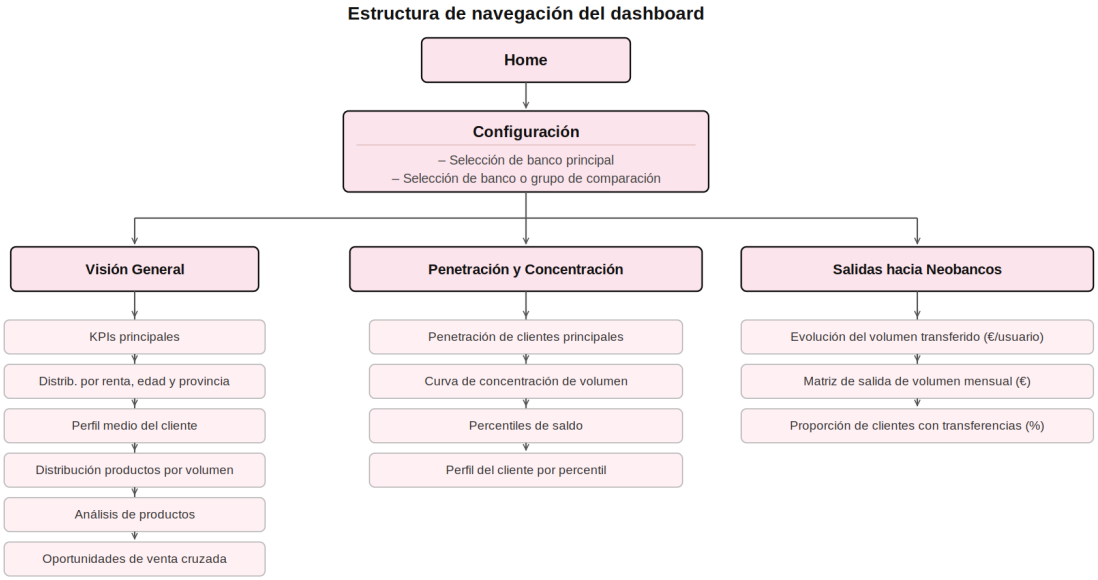

Figura 5.5: Estructura de navegación del dashboard.

5.2.3. Diseño funcional de las páginas

El diseño funcional del dashboard se organizó en torno a tres páginas analíticas principales. Cada una de ellas responde a una finalidad distinta: ofrecer una visión general del banco seleccionado, profundizar en la penetración y concentración de sus clientes, y analizar las salidas de dinero hacia neobancos.

5.2.3.1. Visión General

La página de *Visión General* se diseñó para ofrecer una primera lectura del estado de la entidad seleccionada. En ella se muestran los KPIs principales del banco, entre ellos la penetración de clientes principales, el share of wallet, los productos promedio por cliente y el riesgo de concentración del 10 % superior de clientes por volumen. Estos indicadores permiten obtener una visión rápida de la posición general de la entidad.

Además de estos KPIs, la página incluye una sección de *Distribución de clientes y volumen por renta, edad y provincia*. Esta parte permite analizar cómo se distribuyen los clientes principales y el volumen total según variables demográficas y geográficas. De esta forma, el usuario puede entender no solo cuánto volumen concentra una entidad, sino también qué perfiles de clientes explican dicho volumen.

Dentro de esta misma página se incorpora el *Perfil medio del cliente*. Este bloque resume características relevantes de los clientes analizados, como edad, renta, provincia, score financiero, porcentaje de autónomos, porcentaje de volumen dentro y fuera de cartera, productos únicos por cliente y producto más común. El objetivo es facilitar una lectura sintética del tipo de cliente predominante en la

entidad o segmento seleccionado.

La sección de *Distribución de productos por volumen* permite observar qué productos financieros concentran mayor peso dentro del volumen expuesto total. Esta visualización ayuda a identificar la composición de la cartera del banco y a distinguir qué productos explican una mayor parte del volumen agregado.

El bloque de *Análisis de productos* permite estudiar, para un producto seleccionado, métricas como la penetración entre clientes principales, el *share of wallet* del producto, los bancos externos con más clientes captados, los bancos externos con mayor volumen captado y la clasificación de clientes según tengan el producto de forma exclusiva, compartida o únicamente fuera de la entidad. También se analiza el ticket o saldo medio por categoría y segmento.

Finalmente, la sección de *Oportunidades de venta cruzada* compara productos relevantes del banco seleccionado frente al mercado o frente a un competidor. A partir de las diferencias de penetración, se estiman posibles clientes potenciales y se clasifica el nivel de oportunidad. Esta lógica permite transformar las diferencias observadas entre entidades en posibles líneas de actuación comercial.

5.2.3.2. Penetración y Concentración

La página de *Penetración y Concentración* se diseñó para estudiar la vinculación de los clientes con la entidad y la distribución del volumen entre distintos grupos de usuarios. La primera parte de la página se centra en la *Penetración de clientes principales*. Donde se ve para todos los clientes principales de un banco que porcentaje tiene un determinado producto dentro o fuera del banco.

La sección de *Curva de concentración de volumen* permite analizar qué proporción del volumen total está acumulada en los clientes con mayor volumen de dinero. Para ello, los clientes se ordenan de mayor a menor volumen y se calcula el porcentaje acumulado que representan distintos percentiles o grupos superiores de usuarios. Esta visualización permite identificar si el volumen de la entidad está distribuido de forma equilibrada o si depende de un conjunto reducido de clientes.

La sección de *Percentiles de saldo* muestra la distribución del saldo o volumen por cliente a través de distintos percentiles. Esta información permite entender los umbrales de volumen que separan a los clientes en distintos niveles y facilita la interpretación de la concentración observada en la curva anterior.

Por último, el *Perfil del cliente por percentil* permite analizar las características de los clientes situados por encima de un percentil seleccionado. Para este grupo se muestran variables como edad, renta, provincia, porcentaje de autónomos, saldo medio y saldo mínimo del percentil. Esta información ayuda a entender qué tipo de cliente concentra mayor volumen dentro de la entidad.

5.2.3.3. Salidas hacia Neobancos

La página de *Salidas hacia Neobancos* se diseñó como un módulo específico porque utiliza un conjunto de datos separado respecto al resto del dashboard.

Aunque procede de la misma fuente externa de datos financieros agregados, esta información responde a una lógica analítica distinta, centrada en transferencias y comportamiento transaccional hacia entidades digitales.

La visualización *Evolución del Volumen Transferido a Neobancos por Usuario (€)* permite analizar cómo cambia el volumen transferido hacia neobancos a lo largo del periodo disponible. Para ello, se utiliza una media ponderada del volumen mensual por usuario, empleando los usuarios activos como peso. Esta métrica ayuda a observar la intensidad con la que los usuarios transfieren dinero hacia entidades digitales.

La *Matriz de Salida de Volumen Transaccional Mensual hacia Neobancos (€)* combina distintas dimensiones del comportamiento transaccional. En ella se representan la frecuencia media mensual ponderada por usuarios activos, el ticket medio ponderado y el volumen mensual medio. Además, el tamaño de la burbuja representa el número de usuarios activos, lo que permite interpretar simultáneamente intensidad, volumen y base de usuarios que hacen transferencias hacia neobancos.

Finalmente, la visualización *Proporción de Clientes Totales que Realizan Transferencias a Neobancos (%)* muestra qué porcentaje de los clientes realiza transferencias hacia neobancos. Esta métrica permite comparar la presencia de distintas entidades digitales y analizar el grado de exposición de los clientes al crecimiento de nuevos competidores financieros.

5.2.4. Diseño del modelo y capas de datos

El diseño del modelo de datos se organizó siguiendo la lógica de arquitectura medallón introducida en la Sección 2.5.1. En el proyecto, esta lógica se adaptó al flujo *raw* $\rightarrow$ *silver* $\rightarrow$ *gold* $\rightarrow$ *final_dashboard*. Esta estructura permitió separar las distintas fases del tratamiento de datos: conservación de los datos originales, limpieza y enriquecimiento, estructuración analítica y generación de ficheros finales optimizados para el dashboard.

La capa *raw* contiene los datos originales recibidos desde la fuente externa de datos financieros agregados. En esta capa se conserva la información de partida sobre usuarios, bancos, productos financieros, saldos, gastos, edad, renta, provincia, score financiero, flags y fechas. Esta información no se utiliza directamente en el dashboard, ya que antes debe ser transformada para corregir inconsistencias, homogeneizar variables y preparar métricas analíticas.

A partir de la capa *raw*, se diseñó y construyó la capa *silver*. Esta capa constituye una fase de preprocesamiento en la que los datos se limpian, normalizan y enriquecen antes de ser utilizados. Aunque esta capa también sirve como base común para otros componentes del proyecto, como el chatbot desarrollado en paralelo, la construcción de esta capa fue una parte fundamental de este TFM, ya que permitió disponer de una base de datos coherente y preparada para el dashboard.

En la capa *silver* se realizaron varias transformaciones relevantes. Se eliminaron

columnas no utilizadas, se agruparon los tramos de renta en categorías, se agruparon las edades en segmentos, se transformó el score financiero en categorías interpretables, se imputaron valores faltantes del score utilizando la moda por segmentos de edad y renta, y se creó la variable que identifica el banco principal del usuario a partir del producto de ingresos recurrentes. También se marcaron variables relevantes como la existencia de hipoteca, se mensualizaron los gastos según la ventana temporal disponible y se unificaron saldos y gastos en una variable común de volumen.

Además, en esta capa se aplicaron transformaciones orientadas a mejorar la calidad analítica de los datos. El volumen se convirtió a valor absoluto, se creó la variable que indica si un producto se encuentra en el banco principal del usuario, se eliminaron outliers por producto mediante percentiles e IQR, y se calculó un coeficiente de ajuste por edad y renta. Este coeficiente permitió generar una variable de volumen normalizado, `normalized_volume`, con el objetivo de ajustar la muestra según la distribución de los perfiles de edad y renta.

Una vez construida la capa *silver*, los datos se estructuraron en la capa *gold*. Esta capa se diseñó siguiendo una lógica de modelo estrella, introducida en la Sección 2.5.2. El objetivo era separar entidades descriptivas, como usuarios, bancos, productos o aseguradoras, de los hechos o medidas cuantitativas utilizadas en el análisis.

La capa *gold* se construyó mediante varias tablas. La tabla `dim_usuario` contiene una fila por usuario e información demográfica, score, banco principal y distintos flags. La tabla `dim_banco` contiene una fila por entidad bancaria. La tabla `dim_producto` recoge el catálogo de productos financieros. Finalmente, la tabla `fact_userbancoproductos` recoge la relación entre usuario, banco, producto y volumen.

Esta estructura permitió ordenar conceptualmente los datos y preparar una base analítica común. En la versión actual, la capa *gold* contiene aproximadamente 340.000 filas en `dim_usuario`, 57 filas en `dim_banco`, 11 filas en `dim_producto` y 1.800.000 filas en `fact_userbancoproductos`. Estos tamaños reflejan la necesidad de estructurar los datos antes de su uso en el dashboard.

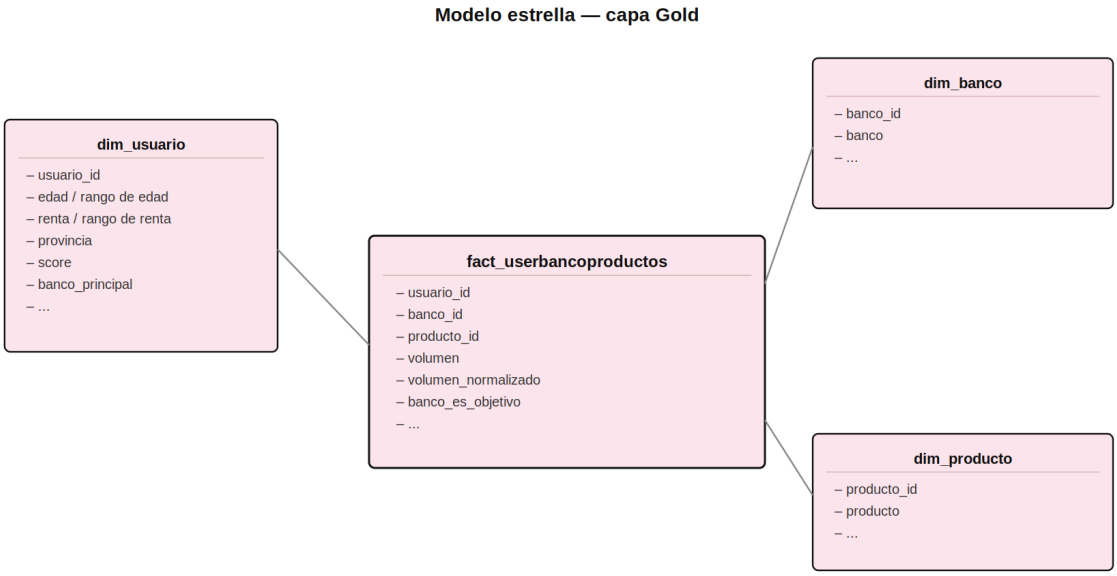

Figura 5.6: Modelo estrella inicial utilizado para estructurar la capa *gold*.

La Figura 5.6 muestra la estructura seguida en la capa *gold*. La tabla central, `fact_userbancoproductos`, recoge la relación entre usuarios, bancos, productos y volumen. Alrededor de ella se sitúan las dimensiones principales, como `dim_usuario`, `dim_banco` y `dim_producto`. Esta separación permitió organizar el dato de forma más clara y facilitar el análisis multidimensional antes de generar las tablas finales del dashboard.

Sin embargo, durante el desarrollo se comprobó que la capa *gold*, aunque adecuada desde un punto de vista analítico, no era suficiente para garantizar un rendimiento adecuado en la aplicación. Al no trabajar con una base de datos analítica tradicional, calcular métricas dinámicamente desde el dashboard generaba problemas de latencia. Por ello, se añadió una capa final específica, denominada *final_dashboard*, orientada al consumo directo por Streamlit.

La capa *final_dashboard* contiene ficheros precalculados por página o módulo, de forma que el dashboard no tenga que recalcular todas las métricas en cada interacción del usuario. Entre los principales ficheros generados se encuentran `overview.parquet`, `overview_distribution.parquet`, `growth_profile.parquet`, `growth_holdings.parquet`, `growth_summary.parquet`, `growth_cross_sell`, `retention_profile.parquet`, `retention_holdings.parquet`, `retention_market_distribution.parquet`, `retention_primary_shares.parquet`, `stability_profile.parquet`, `stability_holdings.parquet` y `fact_neobancos.csv`. Estos ficheros permiten que la aplicación cargue datos ya preparados y realice operaciones más ligeras de filtrado, agrupación o selección.

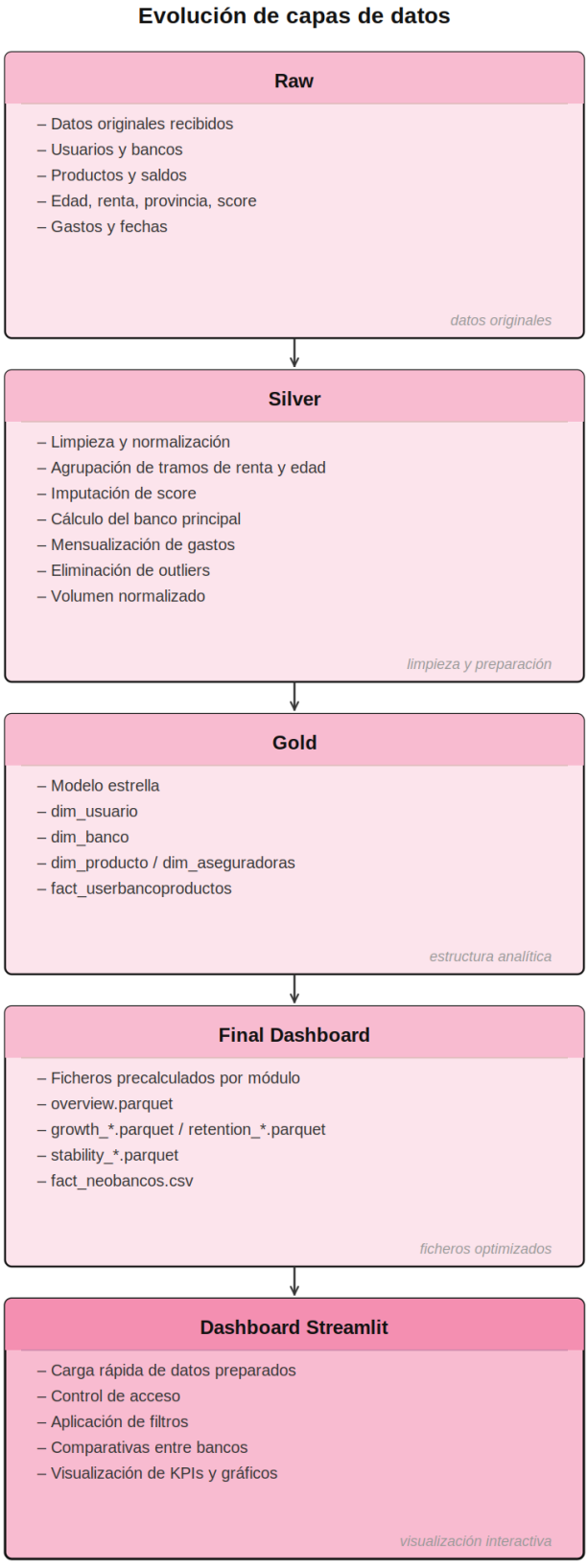

Figura 5.7: Evolución desde los datos originales hasta la capa final optimizada para el dashboard.

La Figura 5.7 resume visualmente la evolución completa del diseño de datos. En primer lugar, la capa *raw* conserva los datos originales tal como se reciben. A continuación, la capa *silver* transforma esos datos en una base más limpia, homogénea y enriquecida. Después, la capa *gold* estructura la información mediante un modelo estrella, separando dimensiones y hechos para facilitar el análisis. Finalmente, la capa *final_dashboard* genera ficheros optimizados para que Streamlit pueda cargar la información de forma rápida y mostrar KPIs, filtros, comparativas y visualizaciones interactivas. De esta forma, el diagrama sintetiza la principal decisión de diseño del modelo de datos: desplazar la complejidad del cálculo hacia fases previas de procesamiento para que la interacción del usuario con el dashboard sea más ágil.

5.2.5. Diseño de actualización de datos

Una parte importante del diseño fue preparar el sistema para que pudiera actualizarse con nuevos datos. El objetivo era que el dashboard no dependiera únicamente de una ejecución puntual, sino que el flujo de procesamiento pudiera repetirse cuando se recibieran nuevos ficheros. Para ello, el código se organizó en scripts que transforman los datos desde la capa *raw* hasta los ficheros finales consumidos por el dashboard.

El flujo de procesamiento se orquesta desde un script principal, que ejecuta las transformaciones necesarias para pasar de los datos originales a las capas *silver*, *gold* y *final_dashboard*. Esta organización facilita que otros usuarios técnicos puedan entender el proceso, ejecutar los scripts correspondientes y actualizar la información mostrada en la aplicación. De esta forma, el proyecto no se limita a crear un dashboard estático, sino que deja preparada una base para su mantenimiento y reutilización futura.

Antes de disponibilizar los datos actualizados en el dashboard, es necesario validar las métricas principales y comprobar que los resultados son coherentes. Esta validación resulta especialmente importante porque las métricas se utilizan para análisis de negocio y para posibles propuestas externas. Por ello, el flujo de actualización no termina únicamente con la generación de nuevos ficheros, sino que incluye una fase explícita de revisión antes de actualizar y desplegar la aplicación.

La Figura 5.8 representa este proceso de actualización. El flujo comienza con la recepción de nuevos datos y la sustitución de los ficheros correspondientes en la capa *raw*. A continuación, el equipo técnico ejecuta el script principal, que genera de forma secuencial las capas *silver*, *gold* y *final_dashboard*. Una vez creados los ficheros finales, se validan las métricas principales y, si los resultados son coherentes, se actualiza y despliega el dashboard en Streamlit. Finalmente, se comprueba el correcto funcionamiento de la herramienta para garantizar que los usuarios puedan consultar los datos actualizados sin errores.

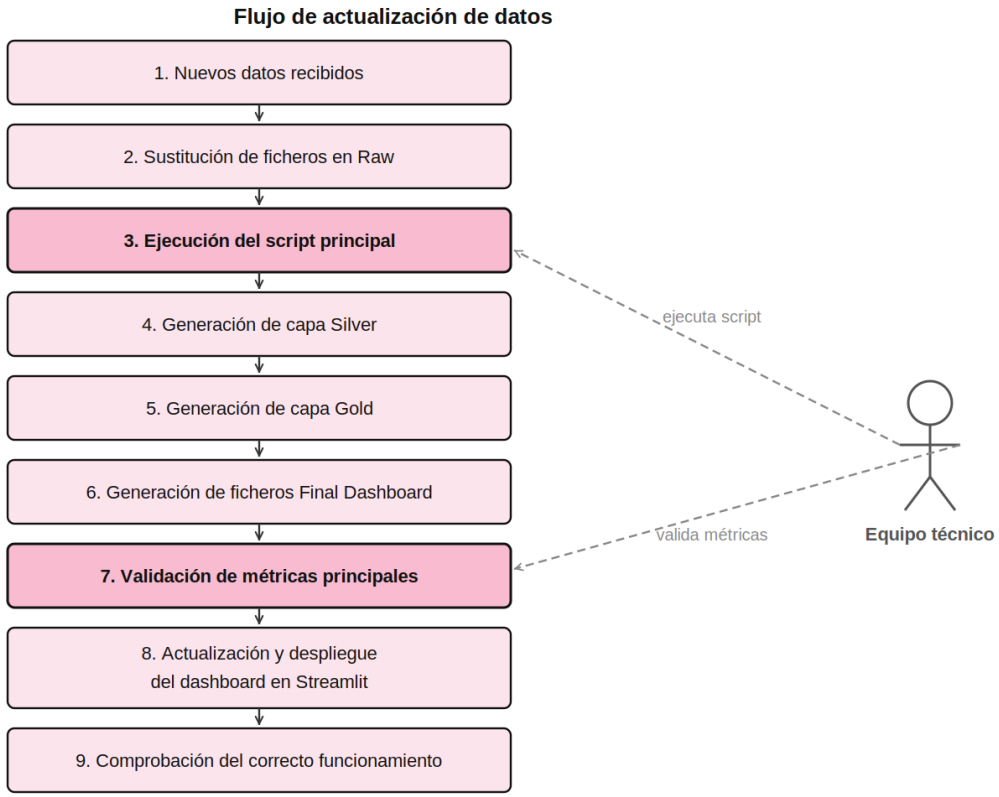

Figura 5.8: Flujo de actualización de datos del dashboard.

El diagrama muestra que la actualización del dashboard no se realiza modificando manualmente cada página, sino regenerando las capas de datos mediante un flujo ordenado. Esta decisión mejora la mantenibilidad del sistema, reduce el riesgo de inconsistencias y permite que la herramienta pueda seguir utilizándose cuando se reciban nuevos datos. Además, al separar la ejecución del procesamiento y la validación de métricas, se refuerza el control de calidad antes de que la información actualizada quede disponible para los usuarios.

5.3. Implementación

La implementación del sistema se realizó principalmente en Python, utilizando Streamlit para la aplicación web, pandas para el procesamiento de datos y Plotly para la visualización interactiva. El desarrollo se dividió en dos grandes bloques: por un lado, la preparación de los datos; por otro, la construcción de la interfaz del dashboard.

5.3.1. Implementación del procesamiento de datos

El procesamiento de datos se implementó mediante scripts de Python organizados por fases. El flujo activo transforma los datos desde la capa *raw* hasta la capa *final_dashboard*, pasando por las capas *silver* y *gold*. Esta organización permite separar la limpieza inicial, la estructuración analítica y la generación de ficheros optimizados para la aplicación.

En la capa *silver*, se realizan las principales transformaciones de limpieza y preparación. Entre ellas se incluyen la eliminación de columnas no utilizadas, la agrupación de tramos de renta y edad, la transformación del score financiero en categorías, la imputación de valores faltantes, la identificación del banco principal a partir de ingresos recurrentes, la mensualización de gastos, la unificación de saldos y gastos en una variable de volumen, la eliminación de outliers por producto y el cálculo de un volumen normalizado mediante un coeficiente de ajuste por edad y renta.

Posteriormente, en la capa *gold*, los datos se organizan siguiendo una lógica de modelo estrella. Esta capa contiene dimensiones de usuarios, bancos, productos y aseguradoras, además de una tabla de hechos que relaciona usuario, banco, producto y volumen. Esta estructura facilita la interpretación analítica de los datos y sirve como base intermedia para generar las tablas finales.

Finalmente, en la capa *final_dashboard*, se generan ficheros específicos para las páginas del dashboard. Estos ficheros contienen métricas y agregaciones ya preparadas, de forma que Streamlit pueda cargarlas directamente sin recalcular todo el pipeline. Esta implementación permitió separar las transformaciones pesadas de la aplicación visual y fue fundamental para mejorar la latencia.

5.3.2. Implementación del dashboard en Streamlit

La aplicación se implementó como un dashboard multipágina en Streamlit. La navegación se estructuró mediante páginas independientes, cada una dedicada a un bloque analítico concreto. Esta organización facilita el mantenimiento del código y permite que cada página tenga su propia lógica de carga de datos, cálculo y visualización.

La página de configuración ocupa un papel central, ya que permite al usuario seleccionar la entidad bancaria y definir el contexto de comparación. Estas selecciones se mantienen durante la navegación mediante el estado de sesión de Streamlit, de forma que las páginas posteriores puedan utilizar la misma entidad y los mismos criterios de análisis. La Figura 5.9 muestra la pantalla final del selector de banco implementada en Streamlit.

Figura 5.9: Pantalla de selección de banco y modo de comparación implementada en Streamlit.

Las páginas principales del dashboard se organizaron en torno a distintos objetivos de negocio. La página de *Visión General* resume la situación de la entidad seleccionada mediante KPIs principales, distribución de clientes y volumen por renta, edad y provincia, perfil medio del cliente, distribución de productos por volumen, análisis de productos y oportunidades de venta cruzada. La Figura 5.10 muestra una captura de esta página, donde se observa la combinación de KPIs, comparación frente a otra entidad.

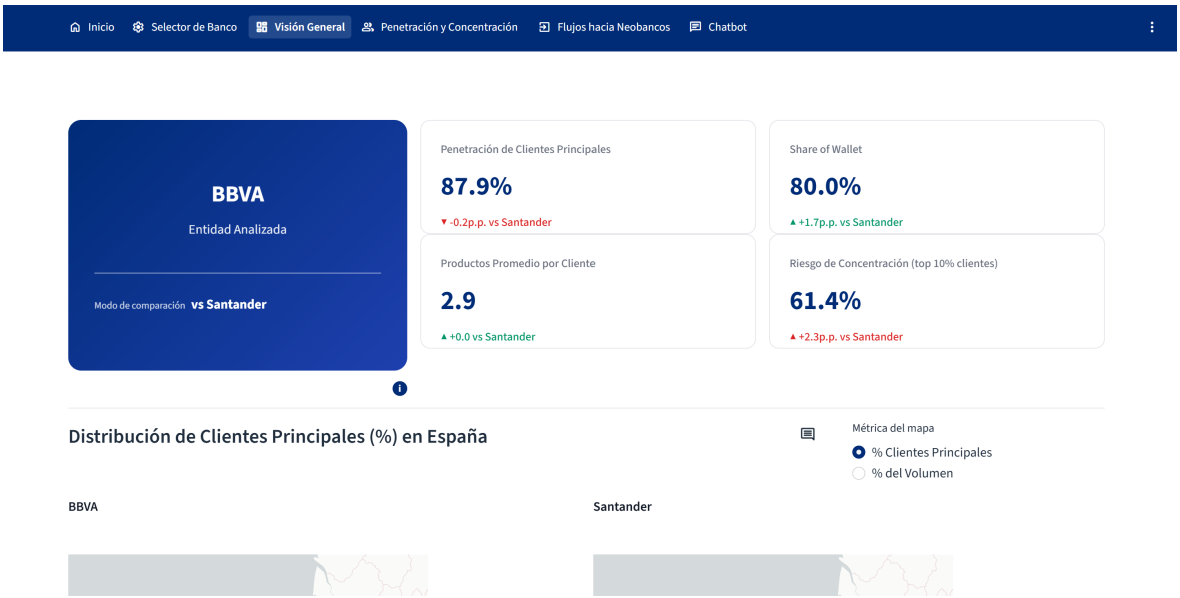

Figura 5.10: Página de *Visión General* implementada en Streamlit.

La página de *Penetración y Concentración* estudia la penetración de clientes principales, la curva de concentración de volumen, los percentiles de saldo y el

perfil del cliente por percentil. La Figura 5.11 muestra la parte de penetración de clientes principales, donde el usuario puede aplicar filtros y comparar la entidad analizada frente a otra entidad.

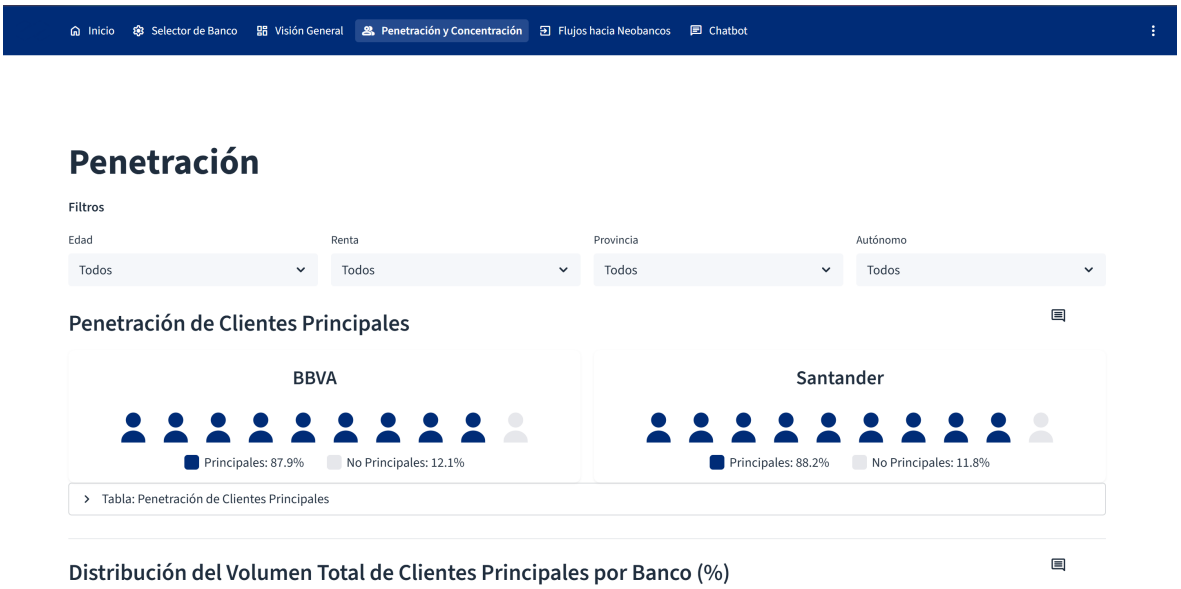

Figura 5.11: Página de *Penetración y Concentración*: sección de penetración de clientes principales.

Finalmente, la página de *Salidas hacia Neobancos* permite observar la evolución del volumen transferido a neobancos por usuario, la matriz de salida de volumen transaccional mensual hacia neobancos y la proporción de clientes totales que realizan transferencias a neobancos. La Figura 5.12 muestra una de las visualizaciones de esta página, centrada en la evolución del volumen transferido hacia neobancos.

Figura 5.12: Página de *Salidas hacia Neobancos*: evolución del volumen transferido por usuario.

5.3.3. Implementación de visualizaciones

Las visualizaciones se implementaron con Plotly, lo que permitió crear gráficos interactivos integrados dentro de Streamlit. Esta elección facilitó que el usuario pudiera explorar los datos mediante gráficos dinámicos, leyendas, tooltips y comparativas visuales.

Las visualizaciones se diseñaron con un criterio funcional: no se trataba únicamente de mostrar gráficos, sino de representar preguntas de negocio. Por ello, cada visualización debía ayudar a interpretar una dimensión concreta del análisis, como la posición relativa de una entidad, la distribución de clientes y volumen, el perfil medio del cliente, la distribución de productos por volumen, la penetración de clientes principales, la concentración de volumen o las salidas hacia neobancos.

5.3.4. Implementación de seguridad

Dado que el dashboard trabaja con información sensible y está orientado a un uso interno, se incorporaron mecanismos de control de acceso mediante usuario y contraseña. El objetivo era evitar que cualquier persona pudiera acceder libremente a la herramienta y garantizar que solo personas de la empresa pudieran consultar los datos.

Esta capa de seguridad no modifica la lógica analítica del dashboard, pero resulta necesaria para su uso operativo. La herramienta no se plantea como una visualización pública, sino como una aplicación restringida al entorno de la empresa.

5.3.5. Implementación del despliegue y monitorización

Además del desarrollo local del dashboard, una parte relevante del proyecto fue su despliegue en un entorno accesible para usuarios internos. El objetivo era que la herramienta no quedara limitada al equipo de desarrollo, sino que pudiera estar disponible para otras personas de la empresa, facilitando su uso real y la recogida de feedback operativo.

El despliegue se realizó en Azure, utilizando un flujo de construcción basado en Oryx Build, descrito en la Sección 2.4.1. En la práctica, este enfoque permitió automatizar parte del proceso de preparación de la aplicación: detección del proyecto, instalación de dependencias y generación del entorno necesario para ejecutar el dashboard. Esta decisión redujo el trabajo manual asociado al despliegue y facilitó que futuras versiones pudieran publicarse con menor fricción.

También se incorporó monitorización mediante Application Insights, introducido en la Sección 2.4.2. En el contexto del dashboard, esta monitorización permite disponer de información sobre errores, disponibilidad y comportamiento de la aplicación. Aunque el análisis detallado de uso queda como línea futura, la integración de monitorización deja preparada una base para controlar mejor el funcionamiento de la herramienta una vez desplegada.

Por último, la gestión de claves y secretos se planteó mediante Azure Key Vault, descrito en la Sección 2.4.3. Esta decisión permite separar credenciales y valores sensibles del código fuente, evitando que información de configuración quede expuesta directamente en el repositorio o en los scripts de la aplicación. Esta capa resulta coherente con el carácter interno del dashboard y con la necesidad de proteger el acceso a información sensible.

En conjunto, el despliegue, la monitorización y la gestión de secretos permitieron acercar el dashboard a un uso operativo dentro de la empresa. La herramienta no solo debía funcionar en local, sino estar disponible en un entorno controlado, monitorizable y preparado para futuras actualizaciones.

5.3.6. Implementación de validación y revisión de cálculos

Durante el desarrollo, una parte importante del trabajo consistió en revisar la coherencia de los cálculos y métricas mostradas. Al tratarse de indicadores utilizados para análisis de negocio, era necesario comprobar que los resultados fueran interpretables y que no aparecieran inconsistencias evidentes.

Esta validación se realizó a partir de revisiones internas y feedback de usuarios reales. Cuando una métrica no parecía tener sentido, o cuando una visualización no respondía correctamente a la pregunta de negocio, se revisaba el cálculo, la fuente de datos o la forma de presentación. Este proceso permitió mejorar progresivamente la fiabilidad del dashboard.

5.3.7. Implementación para mantenimiento y actualización futura

En la fase final del proyecto se trabajó en preparar el código para que pudiera ser utilizado y mantenido por otras personas. Esto implicó organizar scripts, estructurar carpetas, documentar el flujo de ejecución y facilitar que el dashboard pudiera actualizarse con nuevos datos.

Esta parte es importante porque el valor del proyecto no depende solo de la versión desarrollada durante el TFM, sino también de su capacidad para seguir utilizándose después. Una herramienta analítica pierde valor si no puede actualizarse con nueva información o si solo puede ser mantenida por la persona que la creó. Por ello, el sistema se preparó para que otros usuarios técnicos pudieran ejecutar el flujo de datos y mantener la aplicación en funcionamiento.

5.3.8. Resumen de la implementación

En conjunto, la implementación permitió transformar un proceso de análisis disperso en una herramienta estructurada. El sistema recibe datos financieros agregados, los procesa mediante scripts de Python, genera capas intermedias y ficheros finales optimizados, y los presenta a través de un dashboard interactivo en Streamlit. La solución incorpora navegación multipágina, visualizaciones interactivas, control de acceso, despliegue en Azure, monitorización, gestión de secretos, validación de métricas y un flujo preparado para futuras actualizaciones.

El resultado es una herramienta funcional que permite a usuarios internos explorar información financiera agregada de forma más autónoma, consistente y eficiente que mediante notebooks o visualizaciones aisladas. Además, el despliegue y los componentes de monitorización y gestión de secretos acercan la herramienta a un uso operativo real dentro de la empresa.

5.4. Síntesis del capítulo

En este capítulo se ha descrito el sistema desarrollado en el TFM, partiendo de las necesidades que debía cubrir y de los actores que intervienen en su uso y mantenimiento. El dashboard se plantea como una herramienta interna de análisis, pero también como una posible solución demostrativa para presentar a entidades bancarias el valor de los datos agregados y las comparativas que pueden obtenerse a partir de ellos.

También se han identificado los principales requisitos funcionales y no funcionales del sistema. Entre ellos destacan la posibilidad de seleccionar una entidad bancaria, comparar resultados frente a otros bancos o grupos de bancos, consultar KPIs generales, analizar la distribución de clientes y volumen por renta, edad y provincia, estudiar el perfil medio del cliente, analizar productos, identificar oportunidades de venta cruzada, analizar penetración y concentración, y observar salidas hacia neobancos. A nivel no funcional, el sistema debía ser usable, mantenible, seguro, consistente y suficientemente rápido para un uso real.

En cuanto al diseño, se ha explicado la navegación multipágina del dashboard, la lógica funcional de cada página y la evolución del modelo de datos. El flujo activo parte de datos originales, genera una capa *silver* de limpieza y normalización, una capa *gold* basada en modelo estrella y una capa *final_dashboard* con ficheros precalculados para cada módulo. Esta última capa fue necesaria para reducir la latencia y permitir que la aplicación consumiera datos ya preparados.

Finalmente, se ha descrito la implementación del sistema mediante Python, pandas, Streamlit y Plotly, así como las tareas de validación, seguridad, despliegue, monitorización, mantenimiento y actualización futura. Con ello, el capítulo muestra cómo el proyecto pasó de una necesidad de análisis interno a una herramienta funcional, desplegada y preparada para seguir evolucionando con nuevos datos, nuevos casos de uso y mayores requisitos operativos.

6. Análisis de Resultados

Este capítulo analiza los resultados obtenidos tras el desarrollo del dashboard. El análisis se organiza en dos partes. En primer lugar, se presentan los resultados principales del proyecto: la construcción de una herramienta funcional, la estructuración del pipeline de datos, la creación de páginas analíticas, la mejora de accesibilidad para usuarios internos, la reducción de latencia, la incorporación de seguridad y la preparación del código para futuras actualizaciones. En segundo lugar, se realiza una discusión crítica sobre el alcance real de estos resultados, sus limitaciones y su relación con las necesidades identificadas al inicio del trabajo.

6.1. Resultados principales

El resultado principal del proyecto fue el desarrollo de un dashboard analítico interactivo en Streamlit capaz de transformar datos financieros agregados en visualizaciones y métricas interpretables para usuarios de negocio. La herramienta permite seleccionar una entidad bancaria, compararla frente a otros bancos o grupos de bancos, analizar sus principales indicadores y explorar diferentes dimensiones relacionadas con clientes, productos, volumen, concentración y salidas hacia neobancos.

6.1.1. Desarrollo de una herramienta funcional y desplegada

El primer resultado relevante fue la construcción de una aplicación funcional, no limitada a un prototipo local. El dashboard fue desplegado en una fase temprana del proyecto para permitir su uso por parte de usuarios reales y recoger feedback operativo. Esta decisión permitió validar la utilidad de la herramienta en un contexto práctico y detectar errores, dudas o necesidades que no habrían aparecido únicamente durante el desarrollo técnico.

La herramienta final permite que usuarios internos consulten información sin necesidad de acceder directamente a los ficheros originales ni ejecutar código. Esto supone una mejora respecto a la situación previa, en la que muchas visualizaciones dependían de análisis manuales o semimanuales realizados por perfiles técnicos. El dashboard centraliza métricas, filtros y visualizaciones en una interfaz común, reduciendo la dependencia de notebooks y facilitando una exploración más autónoma de los datos.

Además, el dashboard también permite mostrar el valor potencial de los datos a posibles clientes externos. Al ofrecer comparativas frente a otros bancos o grupos de bancos, la herramienta puede servir como base para explicar cómo una entidad se sitúa frente al mercado, qué áreas estratégicas podría revisar y qué información agregada puede obtener a partir de datos que normalmente no tendría disponibles

en sus sistemas internos.

6.1.2. Estructuración del pipeline de datos

Otro resultado importante fue la construcción de un pipeline de datos estructurado siguiendo la lógica *raw* → *silver* → *gold* → *final_dashboard*. Esta organización permitió separar claramente las fases de recepción de datos, limpieza, estructuración analítica y generación de ficheros optimizados para la aplicación.

La capa *silver* permitió limpiar, normalizar y enriquecer los datos originales. En ella se agruparon variables como edad y renta, se transformó el score financiero, se imputaron valores faltantes, se identificó el banco principal de cada usuario, se mensualizaron gastos, se unificaron saldos y gastos en una variable común de volumen, se eliminaron outliers y se calculó el volumen normalizado. Esta capa fue clave para construir una base coherente tanto para el dashboard como para otros componentes relacionados del proyecto.

Posteriormente, la capa *gold* organizó los datos mediante una lógica de modelo estrella. Esta capa incluyó aproximadamente 340.000 usuarios, 57 bancos, 11 productos y 1.800.000 registros en la tabla de hechos. Aunque esta estructura resultó útil desde el punto de vista analítico, durante el desarrollo se comprobó que no era suficiente para garantizar una interacción fluida si las métricas se calculaban dinámicamente desde el dashboard.

Por ello, se generó una capa *final_dashboard* con ficheros precalculados por página o módulo. Esta capa permitió que Streamlit consumiera datos ya preparados, reduciendo el número de cálculos pesados durante la interacción del usuario. Este resultado fue especialmente importante porque el proyecto no se apoyaba en una base de datos analítica tradicional, sino en ficheros preparados previamente.

6.1.3. Construcción de páginas analíticas

El dashboard se organizó en tres grandes páginas analíticas: *Visión General*, *Penetración y Concentración* y *Salidas hacia Neobancos*. Esta estructura permitió ordenar la información según preguntas de negocio distintas y evitar una interfaz excesivamente concentrada.

La página de *Visión General* reúne los KPIs principales de la entidad seleccionada, como penetración de clientes principales, *share of wallet*, productos promedio por cliente y riesgo de concentración del 10 % superior. Además, incorpora la distribución de clientes y volumen por renta, edad y provincia, el perfil medio del cliente, la distribución de productos por volumen, el análisis de productos y las oportunidades de venta cruzada.

La página de *Penetración y Concentración* permite analizar, por un lado, la penetración de clientes principales y, por otro, la concentración del volumen entre los clientes de mayor saldo. Esta página incluye visualizaciones como la curva de concentración de volumen, los percentiles de saldo y el perfil del cliente por

percentil.

La página de *Salidas hacia Neobancos* constituye un módulo específico, basado en un conjunto de datos separado respecto al resto del dashboard. En ella se presentan visualizaciones como la evolución del volumen transferido a neobancos por usuario, la matriz de salida de volumen transaccional mensual hacia neobancos y la proporción de clientes totales que realizan transferencias a neobancos.

6.1.4. Mejora de rendimiento y reducción de latencia

Uno de los resultados más relevantes del proyecto fue la reducción del tiempo de carga del dashboard. Durante el desarrollo se probaron distintas estrategias de optimización: preagregación de métricas de venta cruzada, preagregación de métricas de penetración, uso de DuckDB, uso de ficheros Parquet y, finalmente, optimización de Parquet mediante índices y tipos de datos.

Los tiempos presentados en esta sección fueron medidos en un entorno local. Por tanto, deben interpretarse como una comparación interna entre versiones del dashboard, no como una medición definitiva de rendimiento en un entorno productivo. Aun así, son útiles para evaluar el impacto relativo de las decisiones de optimización aplicadas durante el proyecto.

La Tabla 6.1 resume la evolución del tiempo total de carga por página a lo largo de las distintas fases de optimización. Los resultados muestran una mejora clara: el tiempo total agregado pasó de 40 segundos en la versión original a 7 segundos en la versión final, lo que supone una reducción aproximada del 82,1 %.

Página	Original	Cross-sell	Penetración	DuckDB	Parquet	Final
Visión general	2,963	3,598	3,239	2,170	1,600	1,000
Crecimiento	19,221	6,323	6,879	5,678	2,700	2,500
Penetración	11,385	12,045	7,043	6,945	3,300	1,200
Concentración	6,310	6,633	6,305	5,934	5,300	2,300
Neobancos	0,161	0,161	0,232	0,165	0,165	0,165
Total	**40,040**	**28,760**	**23,698**	**20,892**	**13,065**	**7,165**

Tabla 6.1: Evolución de los tiempos de carga en local durante las fases de optimización, expresados en segundos.

La mejora más significativa se produjo en las páginas con mayor carga analítica. Como se observa en la Tabla 6.2, el bloque de crecimiento pasó de 19,221 segundos a 2,500 segundos, lo que representa una reducción aproximada del 87,0 %. La página de penetración pasó de 11,385 segundos a 1,200 segundos, con una reducción aproximada del 89,5 %. En el caso de visión general, el tiempo se redujo de 2,963 segundos a 1,000 segundo, mientras que concentración pasó de 6,310 segundos a 2,300 segundos. La página de neobancos apenas varió porque ya presentaba tiempos muy bajos desde el inicio.

Página	Tiempo original	Tiempo final	Reducción aproximada
Visión general	2,963 s	1,000 s	66,3 %
Crecimiento	19,221 s	2,500 s	87,0 %
Penetración	11,385 s	1,200 s	89,5 %
Concentración	6,310 s	2,300 s	63,5 %
Neobancos	0,161 s	0,165 s	Sin cambio relevante
Total	**40,040 s**	**7,165 s**	**82,1 %**

Tabla 6.2: Reducción aproximada de tiempos de carga en local entre la versión original y la versión final.

En conjunto, las Tablas 6.1 y 6.2 muestran que la creación de ficheros precalculados y optimizados fue una decisión adecuada. La mejora no se obtuvo únicamente cambiando el formato de almacenamiento, sino desplazando parte del coste computacional fuera del momento de interacción del usuario. En otras palabras, el dashboard dejó de calcular dinámicamente una parte relevante de las métricas y pasó a consumir datos preparados previamente.

6.1.5. Validación de métricas y mejora mediante feedback

Un resultado relevante del proyecto fue la incorporación de feedback real durante el desarrollo. Al desplegar la herramienta y permitir que usuarios internos interactuaran con ella, se detectaron métricas que necesitaban revisión, visualizaciones que debían presentarse de otra forma y cálculos que requerían validación adicional.

El feedback recibido fue principalmente operativo y cualitativo, no una evaluación formal mediante cuestionarios o entrevistas estructuradas. Aun así, resultó útil para mejorar la herramienta. Entre los comentarios recibidos se identificaron tres tipos principales de necesidades: explicar mejor la historia que debía contar cada página, aclarar cómo se calculaban determinadas métricas y corregir métricas cuyo cálculo no reflejaba correctamente la lógica de negocio esperada.

Estas mejoras fueron incorporadas progresivamente. En primer lugar, se trabajó en que el dashboard no fuera solo una sucesión de gráficos, sino una herramienta con una narrativa analítica más clara. Cada página debía ayudar al usuario a entender una pregunta concreta: cómo está posicionada una entidad, qué perfiles explican su volumen, dónde existen oportunidades de crecimiento, cómo se distribuye la concentración o qué comportamiento aparece frente a neobancos.

En segundo lugar, se mejoró la explicación de las métricas. Algunas métricas no resultaban suficientemente claras para los usuarios porque no era evidente cómo se calculaban o qué interpretación debían tener. Por ello, se incorporaron ajustes orientados a hacer más comprensible el significado de los indicadores y a reforzar la conexión entre cálculo técnico y lectura de negocio.

En tercer lugar, se revisaron y corrigieron cálculos cuando el feedback

mostraba que una métrica no representaba correctamente lo que se quería analizar. Este proceso fue importante porque permitió pasar de una herramienta técnicamente funcional a una herramienta más fiable desde el punto de vista de negocio. En un dashboard analítico, no basta con que el código se ejecute correctamente: las métricas deben ser coherentes, interpretables y útiles para la toma de decisiones.

6.1.6. Seguridad y control de acceso

Otro resultado del proyecto fue la incorporación de mecanismos de seguridad mediante usuario y contraseña. Dado que el dashboard trabaja con información sensible y está orientado a un uso interno, era necesario evitar que cualquier persona pudiera acceder libremente a la herramienta.

La incorporación del control de acceso permitió avanzar desde una aplicación puramente técnica hacia una solución más adecuada para un entorno empresarial. Aunque la seguridad no constituye el núcleo analítico del TFM, sí es una condición necesaria para que la herramienta pueda utilizarse en un contexto real.

6.1.7. Preparación para mantenimiento y actualización futura

En la fase final del proyecto se trabajó en preparar el código para que pudiera ser utilizado y actualizado por otras personas. Esto incluyó organizar scripts, estructurar carpetas, documentar el flujo de ejecución y facilitar la regeneración de los ficheros finales cuando se reciban nuevos datos.

Este resultado es especialmente relevante porque el valor del dashboard no depende únicamente de la versión entregada durante el TFM. Una herramienta analítica pierde utilidad si no puede actualizarse con nueva información o si solo puede ser mantenida por la persona que la desarrolló inicialmente. Por ello, el proyecto dejó preparada una base para que el equipo técnico pueda actualizar datos, ejecutar el pipeline y mantener la aplicación en funcionamiento.

6.1.8. Resumen de resultados frente a objetivos

La Tabla 6.3 resume la relación entre los objetivos definidos y los resultados obtenidos. Para facilitar su lectura, los objetivos se agrupan por áreas de resultado.

Área de objetivo	Resultado obtenido	Interpretación
Herramienta analítica interactiva	Se desarrolló un dashboard multipágina en Streamlit con visualizaciones interactivas en Plotly.	Permite explorar datos financieros agregados mediante filtros, KPIs y gráficos sin ejecutar código.
Accesibilidad y homogeneización	Las métricas principales se centralizaron en una única aplicación y se calcularon a partir de un pipeline común.	Reduce la dependencia de análisis manuales y mejora la consistencia respecto a visualizaciones aisladas.
Calidad y validación de métricas	Se revisaron cálculos a partir de feedback real y se corrigieron inconsistencias detectadas.	La validación incorporó criterios técnicos y de negocio, no solo comprobación de ejecución del código.
Rendimiento	Se redujo el tiempo total de carga en local de 40,040 segundos a 7,165 segundos.	La reducción aproximada del 82,1 % confirma que la capa *final_dashboard* mejoró sustancialmente la experiencia de uso.
Seguridad, mantenimiento y evolución	Se añadió control de acceso y se organizó el flujo para actualizar datos y regenerar ficheros finales.	La herramienta queda mejor preparada para uso interno, mantenimiento futuro y nuevas actualizaciones.
Potencial comercial	La herramienta permite comparar entidades y generar *insights* útiles para posibles propuestas externas.	El dashboard puede utilizarse como demostración del valor de los datos agregados para entidades bancarias.

Tabla 6.3: Resumen de resultados obtenidos frente a los objetivos del proyecto.

6.2. Discusión crítica

Los resultados obtenidos muestran que el proyecto logró transformar una explotación de datos principalmente técnica y bajo demanda en una herramienta analítica más accesible, estructurada y reutilizable. El dashboard permite que usuarios internos consulten información relevante sin depender de notebooks, y ofrece una base para demostrar a potenciales clientes externos el valor de los datos financieros agregados.

Uno de los aspectos más positivos del proyecto fue la combinación entre desarrollo técnico y validación práctica. El despliegue temprano permitió que usuarios reales interactuaran con la herramienta y generaran feedback operativo. Esto hizo posible detectar problemas que no se habrían identificado únicamente revisando el código, como métricas difíciles de interpretar, cálculos que requerían ajustes o visualizaciones que debían reorganizarse para responder mejor a las necesidades del negocio.

La mejora de rendimiento constituye una de las evidencias más sólidas del proyecto, ya que puede medirse de forma objetiva. La reducción del tiempo total

de carga en local de 40,040 segundos a 7,165 segundos muestra que la decisión de evolucionar hacia ficheros precalculados fue acertada. Además, las mayores mejoras se concentran precisamente en las páginas que inicialmente presentaban más latencia, como crecimiento y penetración. Esto refuerza la idea de que el problema no era únicamente visual, sino también arquitectónico: para que el dashboard fuera útil en un contexto real, era necesario rediseñar la preparación de datos y no limitarse a mejorar la interfaz.

También fue relevante la evolución de la arquitectura de datos. El modelo estrella aportó una estructura analítica clara, pero no resolvió por sí solo el problema de rendimiento. La necesidad de crear una capa *final_dashboard* muestra una conclusión importante del proyecto: una arquitectura conceptualmente correcta no siempre es suficiente si no se adapta a las restricciones reales de uso. En este caso, la ausencia de una base de datos analítica tradicional hizo necesario trabajar con ficheros precalculados para reducir la latencia.

En conjunto, los resultados son positivos porque el proyecto cumple su objetivo principal: convertir datos financieros agregados en una herramienta funcional, interactiva y preparada para uso real. Al mismo tiempo, el análisis muestra que la solución debe seguir evolucionando, especialmente en términos de actualización de datos, validación continua y posible escalabilidad técnica. Estas cuestiones se retoman de forma específica en el capítulo de conclusiones, donde se presentan las limitaciones y líneas de trabajo futuro.

6.3. Síntesis del capítulo

En este capítulo se han analizado los principales resultados obtenidos tras el desarrollo del dashboard. En primer lugar, se ha mostrado que la herramienta permitió pasar de una explotación de datos basada en notebooks y visualizaciones aisladas a una aplicación interactiva, desplegada y organizada por páginas analíticas.

También se ha explicado cómo la construcción del pipeline de datos permitió estructurar la información en capas sucesivas y generar ficheros finales optimizados para el dashboard. Esta arquitectura fue especialmente relevante para mejorar el rendimiento, como demuestra la reducción del tiempo total de carga en local de 40,040 segundos a 7,165 segundos.

Además, se ha destacado la importancia del feedback operativo recibido durante el desarrollo. Aunque no se recogió mediante una evaluación formal, permitió mejorar la narrativa de las páginas, aclarar el cálculo de métricas y corregir indicadores que no reflejaban adecuadamente la lógica de negocio esperada.

Finalmente, la discusión crítica ha permitido identificar las principales limitaciones del proyecto: ausencia de histórico completo, dependencia de los datos disponibles, necesidad de seguir validando métricas con usuarios y posible evolución futura hacia una infraestructura más escalable. A pesar de estas limitaciones, los resultados muestran que el TFM consiguió desarrollar una

herramienta funcional, útil para análisis interno y con potencial para demostrar el valor de los datos agregados a posibles clientes externos.

7. Conclusiones y Trabajos Futuros

Este capítulo sintetiza los principales resultados obtenidos, las aportaciones realizadas, las limitaciones identificadas y las posibles líneas de continuación.

7.1. Conclusiones

El proyecto ha permitido desarrollar un dashboard analítico interactivo orientado a la explotación de datos financieros agregados. El punto de partida era una situación en la que la empresa ya disponía de datos y de cierta capacidad técnica para analizarlos, pero no contaba con una herramienta común, accesible y reutilizable que permitiera a distintos usuarios internos explorar la información de forma autónoma. En este contexto, el TFM ha transformado un proceso basado en análisis bajo demanda y visualizaciones aisladas en una aplicación estructurada, desplegada y preparada para su uso operativo.

Una de las principales conclusiones del trabajo es que el valor del dashboard no reside únicamente en la visualización de datos, sino en la construcción de una capa analítica intermedia que permite convertir datos originales en métricas interpretables. Para ello, fue necesario diseñar un flujo de datos basado en capas: *raw*, *silver*, *gold* y *final_dashboard*. Esta estructura permitió separar la limpieza, normalización, modelado y optimización de los datos antes de su consumo por la aplicación.

También se concluye que la arquitectura inicial basada en modelo estrella fue útil para ordenar conceptualmente la información, pero no suficiente para garantizar una buena experiencia de uso. La ausencia de una base de datos analítica tradicional hizo necesario crear una capa final de ficheros precalculados. Esta decisión fue clave para reducir la latencia del dashboard, pasando de un tiempo total de carga en local de 40,040 segundos a 7,165 segundos, lo que supone una reducción aproximada del 82,1 %.

Otra conclusión relevante es que el despliegue temprano de la herramienta permitió mejorar el resultado final. Al disponibilizar el dashboard durante el desarrollo, se recibió feedback operativo de usuarios internos. Este feedback permitió ajustar la narrativa de las páginas, aclarar la forma de cálculo de determinadas métricas y corregir indicadores que no reflejaban correctamente la lógica de negocio esperada. Por tanto, el proyecto no siguió un proceso puramente técnico, sino una evolución iterativa guiada por necesidades reales.

Finalmente, el dashboard también abre una vía de valor comercial. Al permitir comparar una entidad bancaria frente a otros bancos o grupos de bancos, la herramienta puede utilizarse como demostración del potencial de los datos agregados. Esto permite mostrar a posibles clientes externos cómo se sitúa su entidad, qué diferencias presenta frente al mercado y qué *insights* podrían obtener

a partir de información que normalmente no estaría disponible en sus propios sistemas internos.

7.2. Aportaciones

La primera aportación del TFM es técnica. El proyecto ha construido una aplicación funcional en Streamlit, integrada con un pipeline de procesamiento de datos en Python y visualizaciones interactivas en Plotly. Esta aplicación permite consultar KPIs, distribuciones, comparativas, oportunidades de venta cruzada, concentración de volumen y salidas hacia neobancos desde una única interfaz.

La segunda aportación está relacionada con la arquitectura de datos. El trabajo ha adaptado una lógica de arquitectura medallón al contexto específico del proyecto, incorporando una capa *silver* de limpieza y enriquecimiento, una capa *gold* basada en modelo estrella y una capa *final_dashboard* orientada al consumo rápido por parte de Streamlit. Esta última capa constituye una adaptación práctica a las restricciones reales del proyecto, especialmente la ausencia de una base de datos analítica y la necesidad de reducir tiempos de carga.

La tercera aportación es metodológica. El proyecto se desarrolló mediante una lógica iterativa basada en sprints, incorporando feedback de usuarios internos y ajustando las funcionalidades conforme se detectaban nuevas necesidades. Esta forma de trabajo permitió construir una herramienta más cercana a los casos de uso reales de la empresa, en lugar de limitarse a una planificación cerrada definida al inicio.

La cuarta aportación es empresarial. El dashboard reduce la dependencia de perfiles técnicos para generar visualizaciones recurrentes y permite que usuarios internos consulten información sin modificar código ni acceder directamente a los ficheros originales. Además, al centralizar métricas y visualizaciones, favorece una interpretación más homogénea de los datos.

La quinta aportación es de mantenimiento y continuidad. El proyecto no se limitó a desarrollar una versión puntual del dashboard, sino que preparó el código para que otros perfiles técnicos pudieran actualizar datos, regenerar ficheros finales y mantener la herramienta. Esta aportación es importante porque una solución analítica solo tiene valor sostenido si puede evolucionar con nuevos datos y nuevas necesidades.

Por último, existe una aportación comercial potencial. La herramienta permite construir una narrativa clara sobre el valor de los datos agregados para entidades bancarias. Esto puede facilitar propuestas comerciales, demostraciones o conversaciones con clientes externos interesados en conocer su posicionamiento relativo frente al mercado.

7.3. Limitaciones

La primera limitación del proyecto está relacionada con el uso de ficheros como base de trabajo. Aunque el uso de CSV y Parquet permitió avanzar sin una base de datos analítica tradicional, esta decisión tiene restricciones. Los ficheros CSV no imponen de forma automática controles estrictos de esquema, tipos de datos, integridad o versionado, lo que puede aumentar el riesgo de errores si los datos de entrada cambian de formato o estructura. Además, trabajar con ficheros obliga a regenerar las capas procesadas cuando se actualizan los datos.

La segunda limitación es la ausencia de un histórico completo para todas las métricas. Esta restricción condicionó el tipo de análisis posible, ya que muchas métricas se centran en comparaciones entre entidades, productos, segmentos o perfiles, pero no permiten estudiar con profundidad la evolución temporal de ciertos indicadores. En un proyecto futuro, disponer de series históricas más completas permitiría construir análisis de tendencia, evolución de clientes y cambios en comportamiento financiero.

La tercera limitación es que la validación de cálculos sigue dependiendo en parte de revisión manual. Aunque se revisaron métricas y se corrigieron inconsistencias detectadas mediante feedback, no existe todavía un sistema completamente automatizado de validación, tests de regresión o control de calidad que alerte cuando un cálculo cambia de forma inesperada. Esto implica que, ante nuevos datos o nuevas métricas, el equipo técnico debe seguir comprobando manualmente que los resultados son coherentes.

La cuarta limitación está relacionada con la evaluación de usuarios. Aunque el feedback recibido fue útil y permitió mejorar la herramienta, la funcionalidad completa del dashboard se integró en una fase final del proyecto, por lo que no transcurrió tiempo suficiente para realizar una evaluación formal de usabilidad. En consecuencia, no puede afirmarse que exista una medición sistemática de satisfacción, facilidad de uso o impacto en productividad. El feedback debe entenderse como evidencia operativa recogida durante el uso inicial de la herramienta, no como una evaluación estadística o plenamente estructurada del dashboard.

La quinta limitación se refiere al rendimiento y escalabilidad futura. Aunque la capa *final_dashboard* redujo de forma significativa los tiempos de carga en local, esta medición no debe interpretarse como un benchmark definitivo en producción. Además, si el volumen de datos aumenta o si la frecuencia de actualización se intensifica, podría ser necesario evolucionar hacia una infraestructura más robusta, con una base de datos analítica o un sistema más automatizado de orquestación.

Finalmente, la herramienta depende de la calidad, granularidad y cobertura de los datos recibidos desde la fuente externa. Si determinadas variables no están disponibles, presentan errores o no tienen suficiente detalle, el dashboard no puede generar ciertos análisis sin modificar previamente la fuente de datos o incorporar datos adicionales.

7.4. Trabajos futuros

Una primera línea de trabajo futuro consiste en ampliar las visualizaciones disponibles. Durante el uso del dashboard y del chatbot asociado pueden surgir preguntas recurrentes de los usuarios. Aquellas consultas que aparezcan con frecuencia podrían transformarse en nuevas gráficas dentro del dashboard, de modo que la herramienta incorpore de forma estable los análisis más demandados y reduzca aún más la necesidad de peticiones ad hoc.

Una segunda línea de mejora sería reforzar la trazabilidad de los cálculos. Actualmente, la validación de métricas depende en parte de revisiones manuales. En el futuro podría incorporarse trazabilidad mediante herramientas como Application Insights en Azure, registrando qué cálculos se ejecutan, con qué datos, en qué momento y con qué resultado. Esto permitiría detectar errores, analizar comportamientos inesperados y mejorar la confianza en las métricas mostradas.

Una tercera línea de trabajo sería automatizar la validación de datos y métricas. Se podrían definir tests de calidad para comprobar esquemas, tipos de datos, valores nulos, rangos esperados, variaciones anómalas y coherencia entre agregados. También sería útil crear tests de regresión para asegurar que una nueva versión del pipeline no modifica métricas clave de forma no justificada.

Otra mejora relevante sería evolucionar la arquitectura de almacenamiento. Aunque el uso de ficheros fue adecuado para las restricciones del proyecto, una base de datos analítica o un entorno lakehouse permitiría mejorar la escalabilidad, el control de versiones, la actualización incremental y la consulta eficiente sobre mayores volúmenes de datos. Esto sería especialmente importante si el dashboard se convierte en una herramienta de uso recurrente dentro de la empresa.

También sería recomendable incorporar una capa más avanzada de permisos y roles. En la versión actual, el acceso se controla mediante usuario y contraseña. En el futuro podría diferenciarse entre perfiles de usuario, por ejemplo, usuarios de negocio, administradores técnicos o perfiles autorizados para ver determinadas entidades o módulos. Esto permitiría adaptar la herramienta a un uso más amplio sin comprometer la seguridad de la información.

Otra línea futura sería ampliar y sistematizar la explicación de las métricas dentro de la propia interfaz. Durante el proyecto ya se incorporaron elementos explicativos para aclarar el significado y la forma de cálculo de los indicadores, en respuesta al feedback recibido de los usuarios. No obstante, esta capa de ayuda podría seguir mejorándose mediante descripciones más completas, tooltips homogéneos, glosarios o paneles de ayuda que expliquen con mayor detalle cómo se calcula cada métrica y qué interpretación debe tener. Esto reforzaría la autonomía de los usuarios y reduciría dudas durante el uso, especialmente si el dashboard se amplía a nuevos perfiles o casos de uso.

Por último, sería conveniente validar el potencial comercial del dashboard con entidades externas. El proyecto ha demostrado que la herramienta puede servir para mostrar comparativas e *insights* agregados, pero sería necesario contrastar qué métricas interesan realmente a los bancos, qué nivel de detalle consideran útil y

cómo debería presentarse la información para convertirse en una propuesta comercial sólida.

En conjunto, los trabajos futuros se orientan en tres direcciones: ampliar el valor analítico de la herramienta, reforzar su robustez técnica y preparar su posible evolución hacia un producto más escalable, trazable y útil tanto para usuarios internos como para potenciales clientes externos.

7.5. Síntesis del capítulo

En este capítulo se han sintetizado las principales conclusiones del Trabajo Fin de Máster. El proyecto ha permitido transformar una forma de análisis basada en notebooks, visualizaciones aisladas y peticiones manuales en una herramienta analítica interactiva, desplegada y orientada a usuarios internos de negocio. El dashboard desarrollado permite consultar métricas financieras agregadas, comparar entidades bancarias, analizar productos, estudiar concentración de volumen y observar salidas hacia neobancos desde una interfaz común.

También se han identificado las principales aportaciones del trabajo. Desde el punto de vista técnico, se ha construido una aplicación en Streamlit integrada con un pipeline de procesamiento en Python. Desde el punto de vista arquitectónico, se ha adaptado una lógica de arquitectura medallón al contexto del proyecto, incorporando una capa *final_dashboard* con ficheros precalculados para mejorar el rendimiento. Desde el punto de vista empresarial, la herramienta reduce la dependencia de perfiles técnicos y permite mostrar el valor potencial de los datos agregados a posibles clientes externos.

Asimismo, se han expuesto las limitaciones del proyecto. Entre ellas destacan el uso de ficheros como base de almacenamiento, la ausencia de un histórico completo para todas las métricas, la validación todavía parcialmente manual de los cálculos, la falta de una evaluación formal de usabilidad y la posible necesidad de una arquitectura más escalable si aumentan el volumen de datos o la frecuencia de actualización.

Finalmente, se han propuesto distintas líneas de trabajo futuro, como ampliar las visualizaciones en función de las preguntas recurrentes de los usuarios, reforzar la trazabilidad de los cálculos, automatizar la validación de datos, evolucionar la arquitectura de almacenamiento, incorporar permisos más avanzados y validar el potencial comercial del dashboard con entidades externas. En conjunto, el capítulo muestra que el TFM no solo ha producido una herramienta funcional, sino también una base técnica y analítica sobre la que la empresa puede seguir construyendo.

8. Bibliografía

[1] Oliver Wyman. Oliver wyman: Impact-driven strategy and consulting. https://www.oliverwyman.com/index.html, 2026. Accessed: 2026-05-27.

[2] Oliver Wyman. Financial services. https://www.oliverwyman.com/our-expertise/industries/financial-services.html, 2026. Accessed: 2026-05-27.

[3] Oliver Wyman. Retail and business banking. https://www.oliverwyman.com/our-expertise/industries/financial-services/retail-and-business-banking.html, 2026. Accessed: 2026-05-27.

[4] pandas development team. pandas.dataframe.groupby. https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.groupby.html, 2026. Accessed: 2026-05-23.

[5] Streamlit. Define multipage apps with st.page and st.navigation. https://docs.streamlit.io/develop/concepts/multipage-apps/page-and-navigation, 2026. Accessed: 2026-05-23.

[6] Streamlit. Session state. https://docs.streamlit.io/develop/api-reference/caching-and-state/st.session_state, 2026. Accessed: 2026-05-23.

[7] Plotly. Plotly python graphing library. https://plotly.com/python/, 2026. Accessed: 2026-05-23.

[8] Ken Schwaber and Jeff Sutherland. The scrum guide. https://scrumguides.org/scrum-guide.html, 2020. Accessed: 2026-05-23.

[9] Figma. Guide to prototyping in figma. https://help.figma.com/hc/en-us/articles/360040314193-Guide-to-prototyping-in-Figma, 2026. Accessed: 2026-05-23.

[10] Microsoft. Configure a linux python app for azure app service. https://learn.microsoft.com/en-us/azure/app-service/configure-language-python, 2025. Accessed: 2026-06-01.

[11] Microsoft. Oryx: Build your repo automatically. https://github.com/microsoft/Oryx, 2026. Accessed: 2026-06-01.

[12] Microsoft. Application insights overview. https://learn.microsoft.com/en-us/azure/azure-monitor/app/app-insights-overview, 2026. Accessed: 2026-06-01.

[13] Microsoft. Azure key vault overview. https://learn.microsoft.com/en-us/azure/key-vault/general/overview, 2025. Accessed: 2026-06-01.

[14] Databricks. What is the medallion lakehouse architecture? https://docs.databricks.com/aws/en/lakehouse/medallion, 2026. Accessed: 2026-05-25.

[15] Microsoft. Implement medallion lakehouse architecture in fabric. https://learn.microsoft.com/en-us/fabric/onelake/onelake-medallion-lakehouse-architecture, 2026. Accessed: 2026-05-25.

[16] Kimball Group. Star schemas and olap cubes. https://www.kimballgroup.com/data-warehouse-business-intelligence-resources/kimball-techniques/dimensional-modeling-techniques/star-schema-olap-cube/, 2026. Accessed: 2026-05-23.

[17] Microsoft. Understand star schema and the importance for power bi. https://learn.microsoft.com/en-us/power-bi/guidance/star-schema, 2024. Accessed: 2026-05-23.

[18] Project Jupyter. Jupyter notebook documentation. https://jupyter-notebook.readthedocs.io/en/stable/notebook.html, 2026. Accessed: 2026-05-23.

[19] Microsoft. Power bi documentation. https://learn.microsoft.com/en-us/power-bi/, 2026. Accessed: 2026-05-23.

[20] Tableau. Business intelligence and analytics software. https://www.tableau.com/, 2026. Accessed: 2026-05-23.

[21] Google Cloud. Looker documentation. https://docs.cloud.google.com/looker/docs/intro, 2026. Accessed: 2026-05-23.