***Informe del Proyecto EmoFinance***

***Gestión de Proyectos de ingeniería***

***Universidad Libre – Sede El Bosque Popular (Bogotá)***

Logotipo

El contenido generado por IA puede ser incorrecto.

***Luis Esteban Martínez Gutiérrez***

***Carlos Alberto Cogollo Rodriguez***

***Joan Gabriel Acevedo Mateus***

***Jonathan David Pulido Mora***

***Universidad Libre – Facultad de Ingeniería***

***24 de octubre de 2025***

**Tabla de contenido**

[**1.** **Introducción** 4](#_Toc212213917)

[**2.** **Objetivo** 6](#_Toc212213918)

[**2.1.** **Objetivo general** 6](#_Toc212213919)

[**2.2.** **Objetivos específicos:** 6](#_Toc212213920)

[**3.** **Alcance del Proyecto** 6](#_Toc212213921)

[**4.** **Requisitos** 7](#_Toc212213922)

[**4.1.** **Requisitos Funcionales** 7](#_Toc212213923)

[**4.2.** **Requisitos No Funcionales** 8](#_Toc212213924)

[**5.** **Modelado de Casos de Uso y Flujo de Usuario** 9](#_Toc212213925)

[**5.1.** **Diagrama de casos de uso** 9](#_Toc212213926)

[**5.2.** **Diseño de la Arquitectura de Información** 10](#_Toc212213927)

[**5.3.** **Flujo de Usuario** 12](#_Toc212213928)

[**5.4.** **Flujo de Estado** 13](#_Toc212213929)

[**5.5.** **Diagrama de Secuencia** 15](#_Toc212213930)

[**5.6.** **Diagrama de Componentes** 16](#_Toc212213931)

[**6.** **Wireframes** 17](#_Toc212213932)

[**6.1.** **Wireframe de Baja Fidelidad** 17](#_Toc212213933)

[**6.2.** **Wireframe de Alta Fidelidad** 20](#_Toc212213934)

[**6.3.** **Importancia dentro del proyecto** 21](#_Toc212213935)

[**7.** **Prototipo en Figma** 24](#_Toc212213936)

[**7.1.** **Descripción general del prototipo** 24](#_Toc212213937)

[**7.2.** **Características del prototipo interactivo** 24](#_Toc212213938)

[**7.3.** **Importancia dentro del desarrollo del proyecto** 25](#_Toc212213939)

[**8.** **Preparación del Repositorio en GitHub** 26](#_Toc212213940)

[**8.1.** **Configuración inicial del repositorio** 26](#_Toc212213941)

[**8.2.** **Estructura del repositorio** 26](#_Toc212213942)

[**9.** **Implementación de Componentes UI** 26](#_Toc212213943)

[**9.1.** **Estructura general de la interfaz** 27](#_Toc212213944)

[**9.2.** **Desarrollo de componentes principales** 27](#_Toc212213945)

[**9.3.** **Resultados** 29](#_Toc212213946)

[**10.** **Desarrollo de la Lógica Interactiva** 29](#_Toc212213947)

[**10.1.** **Arquitectura funcional** 29](#_Toc212213948)

[**10.2.** **Principales interacciones implementadas** 30](#_Toc212213949)

[**10.3.** **Lenguajes y herramientas empleadas** 31](#_Toc212213950)

[**11.** **Diseño de Base de Datos** 32](#_Toc212213951)

[**11.1.** **Objetivo del diseño** 32](#_Toc212213952)

[**11.2.** **Metodología de diseño** 32](#_Toc212213953)

[**11.3.** **Relaciones entre entidades** 34](#_Toc212213954)

[**11.4.** **Gestor de base de datos y estructura técnica** 34](#_Toc212213955)

[**11.5.** **Seguridad y manejo de datos** 35](#_Toc212213956)

[**12.** **Desarrollo de la API REST / GraphQL** 36](#_Toc212213957)

[**12.1.** **Objetivo del desarrollo** 36](#_Toc212213958)

[**12.2.** **Enfoque tecnológico** 36](#_Toc212213959)

[**12.3.** **Seguridad e implementación** 37](#_Toc212213960)

[**12.4.** **Formato de intercambio de datos** 38](#_Toc212213961)

[**13.** **Configuración de Servidores y Seguridad** 39](#_Toc212213962)

[**13.1.** **Arquitectura de despliegue** 39](#_Toc212213963)

[**13.2.** **Configuración de Firebase** 40](#_Toc212213964)

[**13.3.** **Gestión de entornos y variables** 40](#_Toc212213965)

[**14.** **Mecanismos de seguridad implementados** 40](#_Toc212213966)

[**14.1.** **Monitoreo y mantenimiento** 41](#_Toc212213967)

[**14.2.** **Resultados y beneficios** 41](#_Toc212213968)

[**15.** **Integración del Frontend y Backend** 42](#_Toc212213969)

[**15.1.** **Objetivo de la integración** 42](#_Toc212213970)

[**15.2.** **Proceso de integración** 42](#_Toc212213971)

[**15.3.** **Comunicación entre capas** 43](#_Toc212213972)

[**15.4.** **Tecnologías empleadas** 44](#_Toc212213973)

[**15.5.** **Pruebas de integración** 44](#_Toc212213974)

[**15.6.** **Resultados de la integración** 45](#_Toc212213975)

[**16.** **Gestión de Riesgos** 45](#_Toc212213976)

[**16.1.** **Descripción general** 45](#_Toc212213977)

[**16.2.** **Objetivos de la gestión de riesgos** 45](#_Toc212213978)

[**16.3.** **Metodología aplicada** 46](#_Toc212213979)

[**17.** **Tabla de Seguimiento del Proyecto** 47](#_Toc212213980)

[**17.1.** **Descripción general** 47](#_Toc212213981)

[**17.2.** **Objetivos del seguimiento** 47](#_Toc212213982)

[**17.3.** **Metodología de seguimiento** 48](#_Toc212213983)

[**17.4.** **Análisis de avances** 49](#_Toc212213984)

[**17.5.** **Conclusión del seguimiento** 49](#_Toc212213985)

[**18.** **Tabla de Presupuesto** 49](#_Toc212213986)

# **Introducción**

El proyecto EmoFinance nace de la necesidad de comprender de forma sistemática la compleja relación entre las emociones humanas y las decisiones financieras. Si bien la literatura sobre comportamiento económico ha identificado factores cognitivos y contextuales que afectan las decisiones de gasto, ahorro e inversión, existe un vacío en la representación visual e interactiva que permita mapear, explorar y analizar esas emociones de manera integrada y accionable. EmoFinance propone un enfoque interdisciplinario que combina investigación cualitativa (entrevistas semiestructuradas, grupos focales) y cuantitativa (encuestas estandarizadas, análisis estadístico) con técnicas de diseño de interacción y desarrollo de software para construir un mapa emocional interactivo aplicable a entornos web.

El objetivo central es desarrollar un modelo visual interactivo que sintetice datos emocionales y financieros, facilitando la identificación de patrones, correlaciones y rutas causales potenciales entre estados afectivos y comportamientos económicos. Para ello, el proyecto integra fases metodológicas rigurosas: levantamiento de requisitos, modelado de casos de uso, diseño de arquitectura de información, creación de wireframes de baja y alta fidelidad, prototipado en Figma, implementación de componentes UI, desarrollo de la lógica de interacción, modelado de la base de datos, construcción de una API REST, y despliegue seguro con integración frontend-backend.

La contribución esperada es doble. Desde la perspectiva investigativa, proporcionar evidencia empírica y visual sobre cómo emociones específicas (ansiedad, confianza, vergüenza, entusiasmo, frustración, entre otras) se manifiestan en patrones de comportamiento financiero. Desde la perspectiva tecnológica y de usabilidad, entregar una herramienta escalable, modular y accesible que permita a investigadores, profesionales de la educación financiera y usuarios finales explorar sus propias relaciones emocionales con el dinero mediante visualizaciones intuitivas y controles interactivos. Se prioriza la ética y la privacidad: el diseño contempla principios de consentimiento informado, anonimización de datos y control del usuario sobre su información.

El alcance del proyecto se circunscribe a la creación de un prototipo funcional desplegado en entorno web que soporte la recolección, almacenamiento y visualización de datos emocionales y financieros, con énfasis en usabilidad y rendimiento. Se protegerán los datos mediante prácticas estándar de seguridad (autenticación, cifrado en tránsito y en reposo, políticas de acceso) y se implementarán pruebas de usabilidad y accesibilidad para garantizar que las visualizaciones sean interpretables por audiencias diversas.

Finalmente, EmoFinance adopta una aproximación iterativa e incremental: cada iteración incorporará retroalimentación de usuarios piloto y métricas de uso, permitiendo refinar tanto los modelos analíticos como las representaciones visuales. Con ello se aspira a transformar información emocional dispersa en conocimiento útil y aplicable para mejorar la toma de decisiones financieras y diseñar intervenciones educativas más sensibles al componente afectivo.

# **Objetivo**

## **Objetivo general**

Desarrollar un modelo visual interactivo que permita la creación de un mapa emocional, con el fin de analizar las relaciones entre las emociones y las decisiones financieras de los usuarios

## **Objetivos específicos:**

* Hacer uso de los requerimientos funcionales y no funcionales para el modelo visual interactivo del mapeo emocional
* Diseñar un modelo visual interactivo que permita a los usuarios explorar, visualizar e interpretar las emociones que influyen en su comportamiento económico de manera intuitiva y accesible.
* Desarrollar un sistema modular que gestione y almacene los datos de manera segura, ofrezca una interfaz interactiva para la visualización del mapa emocional e integre APIs para la recopilación y análisis de respuestas cualitativas y cuantitativas, garantizando una estructura escalable y de alto rendimiento.
* Realizar pruebas exhaustivas de usabilidad y funcionalidad del software para asegurar que cumpla con los estándares de rendimiento y accesibilidad, detectando y corrigiendo posibles errores.
* Desplegar el sistema de forma segura y eficiente, garantizando un proceso de monitoreo continuo y actualizaciones

# **Alcance del Proyecto**

El proyecto EmoFinance abarca el diseño, desarrollo e implementación de un sistema web interactivo orientado a la representación visual de las emociones relacionadas con las decisiones financieras de los usuarios. El alcance comprende todas las fases del ciclo de vida del desarrollo de software, desde la concepción de la idea hasta la integración de los componentes técnicos y funcionales.

En la etapa de análisis y planificación, se contemplan la recopilación de requerimientos funcionales y no funcionales, la definición de los objetivos del sistema y la especificación de los casos de uso que determinan el comportamiento esperado del usuario dentro de la plataforma.

Durante la fase de diseño, se desarrolla la arquitectura de información, los flujos de usuario y los wireframes de baja y alta fidelidad, así como el prototipo funcional en Figma. Estas actividades buscan garantizar la coherencia entre la experiencia de usuario y los objetivos del proyecto, asegurando que la navegación sea intuitiva, accesible y visualmente coherente.

La fase de desarrollo incluye la implementación de los componentes de la interfaz de usuario (UI), la programación de la lógica de interacción y el desarrollo de la base de datos que almacenará la información emocional y financiera. Asimismo, se construirá una API REST para la gestión y análisis de los datos recolectados, permitiendo la comunicación entre el frontend y el backend del sistema.

En la fase de integración y despliegue, se configura la infraestructura de servidores, se establecen medidas de seguridad para la protección de datos, y se lleva a cabo la integración total entre las capas de presentación, lógica y datos. El sistema será sometido a pruebas exhaustivas de funcionalidad, rendimiento y usabilidad para garantizar su correcto funcionamiento antes de su puesta en marcha.

Finalmente, se contempla una etapa de mantenimiento y mejora continua, enfocada en la corrección de errores, optimización del rendimiento y actualización del sistema de acuerdo con las necesidades emergentes y la retroalimentación de los usuarios. El proyecto no incluye, dentro de su alcance inicial, el desarrollo de aplicaciones móviles nativas ni la integración con plataformas de terceros para análisis financiero automatizado, aunque la arquitectura del sistema estará diseñada para permitir dichas ampliaciones en fases futuras.

# **Requisitos**

## **Requisitos Funcionales**

Los requisitos funcionales corresponden a las acciones, procesos y capacidades que debe ofrecer el sistema para que el proyecto cumpla con los objetivos planteados.

1. **Gestión de usuarios y accesos**  
   El sistema debe permitir que los participantes puedan registrarse de forma sencilla, iniciar sesión con credenciales seguras y recuperar su contraseña en caso de olvido. Además, se deberán manejar distintos perfiles de acceso: administrador, investigador y participante, cada uno con permisos diferenciados.
2. **Aplicación de encuestas y recolección de información**  
   Se requiere un módulo que facilite la elaboración, publicación y aplicación de encuestas en línea. Dichas encuestas deberán incluir preguntas abiertas, de opción múltiple y escalas emocionales. El sistema también debe ofrecer la posibilidad de almacenar entrevistas y cargar archivos relacionados, como audios o videos, para enriquecer el análisis.
3. **Almacenamiento y organización de datos**  
   Toda la información recolectada debe guardarse en una base de datos estructurada, con mecanismos que permitan la búsqueda, filtrado y edición de los registros. La base de datos debe soportar tanto datos cuantitativos como cualitativos, y contar con funciones de respaldo y recuperación.
4. **Mapa emocional interactivo**  
   El principal resultado esperado es un mapa emocional que represente gráficamente los patrones detectados. El sistema debe permitir que el usuario explore este mapa de forma dinámica, filtrando por categorías de emociones, rangos de edad, género u otras variables sociodemográficas.
5. **Análisis de resultados**  
   Se debe incorporar un módulo de análisis que combine métodos estadísticos con herramientas de interpretación cualitativa, permitiendo detectar tendencias y relaciones entre emociones y decisiones financieras.
6. **Generación de reportes**  
   El sistema debe ofrecer la opción de generar informes automáticos en los que se incluyan gráficos, tablas y conclusiones preliminares. Estos reportes podrán exportarse en formatos como PDF o Excel para ser compartidos con los interesados.
7. **Validación y retroalimentación**  
   Antes de su despliegue definitivo, el sistema debe contar con pruebas de funcionalidad y usabilidad, donde los usuarios puedan aportar retroalimentación. Esta etapa permitirá detectar errores, inconsistencias o dificultades en la experiencia de uso.

## **Requisitos No Funcionales**

Los requisitos no funcionales establecen las condiciones de calidad, seguridad y desempeño que debe cumplir el sistema para garantizar una operación adecuada.

1. **Seguridad de la información**  
   Los datos recopilados, al tratarse de información sensible, deben almacenarse de manera segura. Se requiere el uso de cifrado tanto en el almacenamiento como en la transmisión de información, junto con protocolos de autenticación confiables que garanticen la privacidad de los participantes.
2. **Rendimiento y capacidad de respuesta**  
   El sistema debe ofrecer tiempos de respuesta adecuados, evitando demoras en la carga de encuestas, mapas o reportes. Se espera que las consultas se resuelvan en un máximo de dos segundos y que las visualizaciones interactivas no superen los cinco segundos en su procesamiento.
3. **Escalabilidad**  
   La arquitectura de la solución debe ser flexible, permitiendo la incorporación de nuevos módulos y el manejo de un mayor número de usuarios sin afectar la estabilidad del sistema.
4. **Disponibilidad y respaldo**  
   El sistema debe mantenerse disponible de manera continua, con una disponibilidad mínima del 95%. También debe realizar copias de seguridad de manera periódica y disponer de un plan de recuperación en caso de fallas críticas.
5. **Usabilidad y accesibilidad**  
   La interfaz debe ser clara, intuitiva y adaptarse a diferentes dispositivos, desde computadores de escritorio hasta teléfonos móviles. Asimismo, debe contemplar principios de accesibilidad para que personas con distintas capacidades puedan interactuar con la plataforma sin limitaciones.
6. **Mantenibilidad**  
   El diseño debe ser modular y documentado, de forma que sea sencillo realizar actualizaciones, correcciones o mejoras futuras sin afectar la operación general del sistema.
7. **Compatibilidad tecnológica**  
   El sistema debe poder ejecutarse en diferentes navegadores (Chrome, Firefox, Edge, Safari) y ser accesible desde sistemas operativos comunes (Windows, Linux, MacOS, Android e iOS).

# **Modelado de Casos de Uso y Flujo de Usuario**

## **Diagrama de casos de uso**

En esta fase del proyecto se desarrolló el modelado de casos de uso y el flujo de usuario con el objetivo de identificar las principales interacciones entre los actores y el sistema EmoFinance. Este modelo permitió definir las funcionalidades esenciales, los roles involucrados y la relación entre las diferentes operaciones del sistema, sirviendo como punto de partida para el diseño de la arquitectura de información y la interfaz de usuario.

El diagrama de casos de uso representa la estructura funcional del sistema, en la cual intervienen cuatro actores principales:

Participante: Usuario que responde encuestas relacionadas con emociones y finanzas. Puede cargar archivos adicionales, como audios o videos, que complementan la información emocional recolectada.

Investigador: Encargado de gestionar las encuestas, generar reportes de análisis y visualizar el mapa emocional resultante. Este actor tiene acceso a herramientas que permiten el filtrado y la interpretación de datos cualitativos y cuantitativos.

Administrador: Responsable de la gestión de usuarios y perfiles. Puede crear, modificar o eliminar cuentas de participantes, investigadores y otros administradores, garantizando el correcto control de accesos y permisos dentro del sistema.

Todos: Representa las acciones comunes a todos los usuarios, como la autenticación, inicio de sesión y recuperación de contraseña.

El flujo de usuario se diseñó considerando la experiencia de navegación desde la interacción inicial hasta la interpretación de resultados. Los participantes inician sesión, completan encuestas emocionales y cargan material complementario; los investigadores analizan la información recolectada, generan reportes y visualizan el mapa emocional; y los administradores supervisan la integridad de los datos y el correcto funcionamiento del sistema.

Este modelo garantiza una estructura modular y segura, que facilita la comprensión del comportamiento esperado del sistema antes de avanzar a las etapas de diseño de interfaz y desarrollo de componentes técnicos.

Diagrama

El contenido generado por IA puede ser incorrecto.

## **Diseño de la Arquitectura de Información**

En esta etapa se desarrolló la arquitectura de información del sistema EmoFinance, cuyo propósito fue definir la estructura de organización, almacenamiento y flujo de datos necesarios para el funcionamiento del modelo visual interactivo. La arquitectura se construyó bajo un enfoque modular y relacional, garantizando la escalabilidad y la coherencia entre los diferentes componentes del sistema.

El diagrama de arquitectura de información muestra las principales entidades que intervienen en el sistema, así como las relaciones entre ellas. Las entidades clave son:

Usuario: Representa a las personas que interactúan con el sistema, ya sea como participantes, investigadores o administradores. Contiene información básica de identificación y credenciales de acceso.

Encuesta: Es el componente central de recolección de datos. Cada encuesta contiene un conjunto de preguntas relacionadas con las emociones y decisiones financieras de los usuarios.

Pregunta: Define los ítems que componen cada encuesta. Las preguntas pueden ser de opción múltiple, escala emocional o respuesta abierta, según el tipo de análisis requerido.

Respuesta: Almacena las respuestas proporcionadas por los participantes. Cada respuesta está asociada a una pregunta específica y a un usuario, lo que permite rastrear y analizar los patrones emocionales individuales.

Repositorio de Datos: Contiene los resultados agregados y los archivos complementarios (como audios, videos o documentos) que enriquecen el análisis cualitativo y cuantitativo del proyecto.

Mapa Emocional: Es el componente de salida visual del sistema. Recibe la información procesada desde el repositorio de datos y la traduce en una representación gráfica interactiva de las emociones vinculadas con las decisiones financieras.

Visualización Final: Constituye la fase final del flujo de información. Integra los resultados del mapa emocional y los muestra al usuario mediante una interfaz visual accesible y dinámica.

La arquitectura se diseñó con un enfoque de integridad relacional, donde las conexiones entre entidades garantizan la trazabilidad de los datos desde la recopilación inicial hasta la visualización final. Este modelo facilita la integración futura con la base de datos y la API REST, manteniendo una estructura coherente entre los módulos de backend y frontend.

En conclusión, el diseño de la arquitectura de información sienta las bases para el almacenamiento estructurado de la información emocional y su análisis visual, constituyendo un pilar fundamental en el desarrollo técnico del proyecto EmoFinance.

Gráfico, Gráfico de cajas y bigotes

El contenido generado por IA puede ser incorrecto.

## **Flujo de Usuario**

El flujo de usuario del proyecto EmoFinance representa la secuencia lógica de acciones que realizan los distintos actores dentro del sistema, desde la creación de una encuesta hasta la generación e interpretación de resultados emocionales. Este modelo permite comprender el recorrido funcional que sigue el usuario y cómo intervienen los diferentes componentes técnicos del sistema.

El proceso inicia con el investigador, quien define los criterios de la encuesta a aplicar. Una vez establecidos los parámetros, el investigador publica la encuesta dentro del sistema para que los participantes puedan acceder a ella. En este punto, se presenta una decisión clave: determinar si existen participantes registrados o interesados en responder.

Si no hay participantes, el flujo concluye, evitando recursos innecesarios y permitiendo reprogramar la publicación de la encuesta. Si sí hay participantes, el sistema habilita la fase de recolección de datos.

El participante responde la encuesta y tiene la posibilidad de cargar audios o videos adicionales, que servirán como material cualitativo para el análisis emocional. Una vez enviada la información, el sistema de base de datos entra en acción para almacenar y organizar los datos de manera estructurada, garantizando la seguridad y trazabilidad de cada respuesta.

Posteriormente, el investigador accede a la información almacenada para realizar búsquedas, filtrados y análisis de los datos recolectados. Con esta información procesada, el sistema genera un mapa emocional, que permite visualizar las emociones predominantes y sus relaciones con las decisiones financieras de los usuarios.

Finalmente, se elaboran reportes de análisis con los hallazgos obtenidos y se realiza una validación e interpretación de los resultados, cerrando así el flujo con la obtención de conclusiones significativas para el estudio.

El flujo de usuario de EmoFinance asegura una integración coherente entre los roles, las decisiones y el sistema de almacenamiento, permitiendo una experiencia de uso fluida, segura y orientada al análisis emocional-financiero.

Gráfico, Gráfico de cajas y bigotes

El contenido generado por IA puede ser incorrecto.

## **Flujo de Estado**

El flujo de estado de las encuestas describe las diferentes etapas que atraviesa una encuesta dentro del sistema EmoFinance, desde su creación inicial hasta su cierre o reapertura para ajustes. Este modelo permite comprender el ciclo de vida de las encuestas y cómo los investigadores pueden gestionarlas en función de las necesidades del estudio y el avance del proyecto.

El proceso inicia en el estado “Inicial”, donde el investigador comienza la elaboración de una nueva encuesta. En esta fase se encuentra en modo “Borrador”, lo que permite definir las preguntas, criterios y configuraciones necesarias antes de ser publicada.

Una vez completado el diseño, la encuesta puede avanzar al estado “Publicada”, habilitándose para que los participantes respondan y aporten sus datos emocionales y financieros. Durante este periodo, el investigador puede suspender o modificar la encuesta si detecta errores o si se requiere ajustar algún parámetro de recolección.

Cuando se considera que la recolección de datos ha sido suficiente, el investigador procede a cerrar la recolección, lo que lleva la encuesta al estado “En análisis”. En esta etapa, los datos obtenidos son procesados, analizados y preparados para generar los reportes finales.

Posteriormente, la encuesta pasa al estado “Archivada”, lo que indica que el proceso principal ha finalizado y los resultados han sido consolidados. Desde este estado es posible reabrir la encuesta para ajustes, en caso de que se necesite actualizar información o realizar nuevas interpretaciones.

Finalmente, tras generar el reporte final, la encuesta llega al estado “Final”, que marca la culminación completa del ciclo de gestión.

Este flujo de estados garantiza un control preciso y transparente sobre el manejo de las encuestas, permitiendo mantener la trazabilidad de los cambios y asegurar la integridad de los datos recolectados a lo largo del proceso de investigación.

Diagrama

El contenido generado por IA puede ser incorrecto.

## **Diagrama de Secuencia**

El diagrama de secuencia del proyecto EmoFinance ilustra la dinámica de comunicación entre los diferentes componentes del sistema durante el proceso en el que un participante responde una encuesta. Este modelo describe el flujo temporal de mensajes y acciones que se ejecutan desde la interfaz del usuario hasta la base de datos, garantizando la validez y el almacenamiento correcto de la información emocional recolectada.

El flujo comienza cuando el participante envía sus respuestas a través de la interfaz de encuesta, lo que genera una solicitud con los datos recopilados (datosRespuesta).

La interfaz de encuesta recibe las respuestas y las procesa, estructurándolas para ser enviadas al controlador de encuesta, que actúa como intermediario entre el frontend y la lógica del backend.

El controlador de encuesta valida la autenticidad del usuario solicitando la verificación del token a través del servicio de seguridad. Esta etapa garantiza que la sesión sea válida y que las respuestas provengan de un usuario autorizado.

Una vez verificada la autenticidad, el servicio de seguridad envía los datos de las respuestas al servicio de datos, que se encarga del almacenamiento persistente.

El servicio de datos ejecuta la instrucción INSERT INTO en la base de datos, registrando cada respuesta junto con su valor y la fecha correspondiente.

La base de datos confirma la operación mediante un mensaje de ACK (Acknowledgment), asegurando que los datos han sido almacenados correctamente.

Finalmente, el controlador de encuesta devuelve una respuesta de confirmación a la interfaz, la cual muestra al participante el mensaje “Datos guardados”, indicando que su información ha sido procesada exitosamente.

Este flujo garantiza un manejo seguro, validado y eficiente de la información emocional recolectada, manteniendo la integridad de los datos y la coherencia en la comunicación entre los distintos módulos del sistema. El diseño modular del proceso facilita la escalabilidad del sistema y su integración con otros servicios de análisis y visualización emocional.

Gráfico, Gráfico de cajas y bigotes

El contenido generado por IA puede ser incorrecto.

## **Diagrama de Componentes**

El diagrama de componentes ilustra la organización modular del sistema EmoFinance, destacando la interacción entre los distintos subsistemas encargados de la gestión de usuarios, el procesamiento de encuestas, el análisis de datos y la visualización de resultados. Este modelo permite identificar las dependencias entre componentes y los puntos de integración dentro del sistema.

El sistema se estructura en cinco componentes principales:

Gestión Usuarios: Responsable del manejo de la información de los usuarios, incluyendo autenticación, registro y control de acceso. Proporciona la interfaz I\_DatosUsuario que transmite los datos al MotorEncuestas para asociar cada respuesta con un usuario específico.

Motor Encuestas: Constituye el núcleo del sistema de recolección de información emocional. Recibe los datos de usuario desde el módulo de gestión y envía la información procesada al ServicioDatos mediante la interfaz I\_Almacenamiento, encargada de persistir las respuestas en la base de datos.

Servicio Datos: Gestiona el almacenamiento, actualización y recuperación de la información. Este módulo actúa como puente entre la capa de aplicación y la base de datos, garantizando la integridad de los datos recopilados por las encuestas.

Módulo Análisis: Encargado del procesamiento estadístico y emocional de los datos. Recibe información desde el ServicioDatos, realiza operaciones de clasificación, segmentación y evaluación de patrones emocionales, y posteriormente entrega los resultados al MóduloVisualización por medio de la interfaz I\_Busqueda.

Módulo Visualización: Se ocupa de presentar los resultados de los análisis a los usuarios en forma de reportes gráficos e interpretaciones. Utiliza la interfaz I\_GeneraciónReporte para obtener los datos procesados y generar informes dinámicos de bienestar emocional y tendencias.

Diagrama

El contenido generado por IA puede ser incorrecto.

# **Wireframes**

Los wireframes representan el diseño preliminar y la evolución visual de la interfaz de usuario del sistema EmoFinance. A través de estos modelos, se busca establecer la estructura, jerarquía y disposición de los elementos de interacción antes de su implementación definitiva en el entorno digital.

## **Wireframe de Baja Fidelidad**

El wireframe de baja fidelidad constituye la primera representación gráfica de la interfaz, centrada en la distribución funcional de los elementos sin considerar aspectos visuales o de estilo. En esta etapa se define:

* La ubicación de menús, botones y campos de entrada.
* La organización general de la página principal y las secciones de encuesta, análisis y reportes.
* Los flujos básicos de navegación entre pantallas.
* La simplicidad en el trazado, utilizando formas geométricas y etiquetas descriptivas para cada elemento.

Este nivel de detalle permite evaluar la usabilidad, detectar posibles redundancias y ajustar la lógica de interacción antes de proceder al diseño visual final. Su propósito principal es validar la experiencia de usuario y la coherencia del flujo de información dentro del sistema.

Diagrama

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

## **Wireframe de Alta Fidelidad**

El wireframe de alta fidelidad corresponde a una versión más avanzada del diseño, donde se integran aspectos visuales, estéticos y de marca propios del sistema EmoFinance. En este nivel se incluyen:

* Colores institucionales, tipografía y logotipo del proyecto.
* Iconografía representativa para cada módulo (usuarios, encuestas, análisis y reportes).
* Distribución definitiva de los componentes gráficos.
* Simulación del comportamiento interactivo, incluyendo botones, menús desplegables y mensajes de confirmación.
* Diseño adaptable a distintos dispositivos (computadores, tabletas y móviles).

Este prototipo permite visualizar cómo se presentará el sistema una vez implementado y sirve de referencia directa para el desarrollo front-end, asegurando coherencia entre la interfaz planificada y la versión funcional.

## **Importancia dentro del proyecto**

La construcción progresiva de los wireframes permitió garantizar una experiencia de usuario intuitiva, con una navegación fluida y clara entre las secciones de registro, encuestas y visualización de resultados. Además, facilitó la comunicación entre los diseñadores, desarrolladores y los responsables del proyecto, asegurando que las decisiones de diseño se basaran en criterios funcionales y de accesibilidad.

Interfaz de usuario gráfica, Aplicación, Teams

El contenido generado por IA puede ser incorrecto.

Gráfico, Gráfico circular

El contenido generado por IA puede ser incorrecto.

# **Prototipo en Figma**

El prototipo en Figma representa la fase avanzada del diseño de la interfaz del sistema EmoFinance, en la cual se materializa la transición desde los wireframes hacia una simulación completamente interactiva. Este prototipo permite visualizar de manera realista la experiencia del usuario, incluyendo el comportamiento dinámico de los elementos y los flujos de navegación definidos durante el modelado.

## **Descripción general del prototipo**

El diseño fue desarrollado utilizando Figma, una herramienta de diseño colaborativo que facilita la creación, validación y documentación de interfaces digitales. El prototipo se estructura en base a los módulos principales del sistema:

1. Gestión de usuarios: permite el registro, autenticación y administración de cuentas de los participantes y administradores.
2. Motor de encuestas: encargado de la creación, publicación y seguimiento de encuestas activas.
3. Módulo de análisis: muestra los resultados recolectados en tiempo real y permite generar visualizaciones estadísticas.
4. Módulo de visualización: ofrece reportes gráficos e informes finales descargables.

Cada uno de estos componentes fue diseñado siguiendo los principios de usabilidad, accesibilidad y coherencia visual, asegurando que la navegación sea clara, fluida y adaptable a diferentes dispositivos.

## **Características del prototipo interactivo**

El prototipo cuenta con elementos completamente funcionales a nivel visual, incluyendo:

1. Transiciones entre pantallas mediante enlaces interactivos.
2. Simulación de formularios, botones, menús y cuadros de diálogo.
3. Reacciones dinámicas ante acciones del usuario (por ejemplo, el envío de una encuesta o la generación de un reporte).
4. Visualización de estados del sistema (borrador, publicado, en análisis, archivado).
5. Diseño responsive, optimizado para dispositivos de escritorio, tabletas y móviles.

Gracias a la interactividad de Figma, se realizaron pruebas de usabilidad con los miembros del equipo de desarrollo para validar la disposición de los elementos, la legibilidad de los textos y la lógica de los flujos de navegación.

## **Importancia dentro del desarrollo del proyecto**

El prototipo en Figma sirvió como puente entre el diseño conceptual y la implementación técnica, permitiendo obtener retroalimentación temprana y reducir posibles errores en las fases de programación. Además, facilitó la alineación visual con los objetivos del proyecto, garantizando una identidad coherente y una experiencia de usuario centrada en la simplicidad y eficiencia.

**Link**: <https://www.figma.com/design/1Z5LOUhwyomPxPjm759c6Z/Untitled?node-id=0-1&t=qgWs13dC8Yh6W44r-1>

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

# **Preparación del Repositorio en GitHub**

La preparación del repositorio en GitHub constituyó una fase fundamental en la organización y control de versiones del proyecto EmoFinance. Esta etapa garantizó la colaboración eficiente entre los integrantes del equipo, permitiendo mantener una estructura clara, actualizada y sincronizada de los archivos que conforman el sistema.

## **Configuración inicial del repositorio**

Para la gestión del código fuente y los documentos del proyecto, se creó un repositorio principal en la plataforma GitHub, el cual actúa como repositorio remoto centralizado. En esta primera fase se definieron los siguientes elementos:

Nombre del repositorio: Proyecto Gestion

Descripción: Plataforma digital para la gestión, análisis y visualización de encuestas sobre emociones y finanzas personales.

Visibilidad: Repositorio privado durante el desarrollo inicial, con la posibilidad de hacerlo público en la etapa final del proyecto.

Archivo README.md: Incluye la descripción general del proyecto, instrucciones de instalación, dependencias y guía básica de uso.

## **Estructura del repositorio**

El repositorio fue organizado siguiendo buenas prácticas de desarrollo, facilitando la localización de archivos y la colaboración entre los participantes. Su estructura se definió de la siguiente forma:  
  
imagen delrepositorio

# **Implementación de Componentes UI**

La implementación de componentes UI (User Interface) representó una etapa clave dentro del desarrollo del proyecto EmoFinance, ya que permitió transformar los prototipos visuales diseñados en Figma en elementos funcionales e interactivos dentro del sistema. Esta fase se enfocó en garantizar una experiencia de usuario intuitiva, estética y coherente con la identidad visual definida durante el proceso de diseño.

## **Estructura general de la interfaz**

El diseño de la interfaz se construyó siguiendo una arquitectura modular, donde cada componente se desarrolló de forma independiente para facilitar su mantenimiento y reutilización. La estructura principal se organizó de la siguiente manera:

Header: contiene el logotipo, el menú de navegación y los accesos principales a las secciones del sistema (Inicio, Encuestas, Análisis y Reportes).

Sidebar (barra lateral): muestra accesos rápidos a funcionalidades específicas del usuario, como la gestión de encuestas y la visualización de estadísticas.

Main Content: espacio central donde se despliegan las vistas dinámicas del sistema (creación, respuesta y análisis de encuestas).

Footer: incluye información institucional, créditos del proyecto y enlaces de contacto.

## **Desarrollo de componentes principales**

Durante la implementación se trabajó con HTML, CSS y JavaScript para el desarrollo del frontend, aplicando una estructura limpia y semántica. Entre los principales componentes UI implementados se destacan:

Formularios de registro y autenticación: diseñados para facilitar el ingreso de investigadores y participantes, con validación en tiempo real y mensajes de error accesibles.

Panel de administración de encuestas: interfaz intuitiva que permite crear, modificar y publicar encuestas, incorporando botones de acción con íconos descriptivos.

Componentes de respuesta: se desarrollaron elementos interactivos como botones tipo radio, checkbox y sliders que permiten registrar respuestas de forma ágil.

Tablas dinámicas y tarjetas de resultados: utilizadas para la visualización de datos procesados, con filtros y opciones de ordenamiento.

Notificaciones y alertas: implementadas mediante modales emergentes para informar al usuario sobre acciones exitosas, errores o advertencias.

Principios de diseño aplicados

Para garantizar la usabilidad y consistencia visual, se aplicaron los siguientes principios de diseño UI/UX:

Consistencia visual: todos los componentes mantienen un estilo unificado en cuanto a tipografía, colores, íconos y espaciado.

Jerarquía visual: el uso de tamaños, contraste y alineación permite destacar los elementos más importantes dentro de cada pantalla.

Accesibilidad: se aseguraron altos niveles de contraste y etiquetas adecuadas para lectores de pantalla.

Diseño responsivo: los componentes se adaptan correctamente a distintos tamaños de pantalla, permitiendo su uso desde dispositivos móviles, tabletas o computadoras.

Librerías y herramientas empleadas

Durante la implementación se hizo uso de herramientas que facilitaron la creación y mantenimiento de los componentes de la interfaz:

Bootstrap 5: para la estructuración y diseño responsivo de los elementos base.

Font Awesome: utilizado para la integración de íconos y símbolos visuales.

SweetAlert2: para las alertas interactivas y notificaciones dinámicas.

CSS personalizado: aplicado para la adaptación del diseño a la identidad visual del proyecto EmoFinance.

## **Resultados**

La implementación de los componentes UI permitió obtener una interfaz moderna, funcional y amigable, alineada con los objetivos del proyecto. Gracias a la modularidad del diseño, el sistema puede ser ampliado fácilmente con nuevos módulos o funcionalidades sin comprometer la estructura existente. Además, la adopción de buenas prácticas de diseño asegura que la experiencia de usuario sea fluida, atractiva y accesible para todos los perfiles de usuarios del sistema.

# **Desarrollo de la Lógica Interactiva**

El desarrollo de la lógica interactiva constituyó una de las fases más relevantes del proyecto EmoFinance, ya que permitió dotar de funcionalidad dinámica a la interfaz de usuario y establecer la comunicación efectiva entre los distintos componentes del sistema. En esta etapa se implementaron los comportamientos que hacen posible la interacción entre los usuarios y las funcionalidades del sistema, integrando la lógica de negocio con los elementos visuales previamente construidos.

## **Arquitectura funcional**

El sistema fue desarrollado bajo una arquitectura basada en componentes, donde cada módulo maneja su propia lógica interna pero se comunica con los demás a través de eventos y funciones compartidas. Esta estructura permitió mantener un código limpio, escalable y de fácil mantenimiento.

Las principales capas de la lógica interactiva fueron:

Capa de presentación (Frontend): encargada de capturar las acciones del usuario y mostrar las respuestas del sistema.

Capa de lógica de negocio: responsable de procesar la información, validar datos y ejecutar las operaciones necesarias según las interacciones.

Capa de comunicación (API): maneja el intercambio de datos con la base de datos y el backend a través de peticiones HTTP.

Flujo general de interacción

El funcionamiento del sistema se basa en un flujo interactivo continuo entre el usuario y la aplicación. A modo general, el proceso sigue los siguientes pasos:

El usuario realiza una acción (por ejemplo, iniciar sesión, responder una encuesta o consultar resultados).

El sistema captura el evento mediante JavaScript y lo procesa a través de una función controladora.

Se validan los datos ingresados antes de ser enviados al servidor, previniendo errores o datos inconsistentes.

La aplicación realiza una solicitud HTTP al backend mediante el uso de fetch() o axios.

El servidor responde con los datos procesados o confirmaciones de las operaciones realizadas.

La interfaz se actualiza dinámicamente, mostrando mensajes, gráficos o nuevas secciones en función de la respuesta recibida.

## **Principales interacciones implementadas**

Durante esta fase se desarrollaron las siguientes funcionalidades interactivas:

Sistema de autenticación y registro: validación de credenciales en tiempo real, manejo de errores de acceso y actualización dinámica del estado del usuario.

Gestión de encuestas: creación, edición y eliminación de encuestas con formularios interactivos y campos dinámicos.

Captura de respuestas: uso de botones, selectores y barras de desplazamiento para registrar las emociones o percepciones de los usuarios.

Procesamiento de resultados: actualización automática de tablas y gráficos al finalizar una encuesta, mostrando estadísticas inmediatas.

Notificaciones interactivas: implementación de modales, alertas y mensajes de confirmación que guían al usuario durante cada acción.

## **Lenguajes y herramientas empleadas**

Para garantizar la eficiencia y fluidez en la interacción, se utilizaron los siguientes lenguajes y librerías:

JavaScript: lenguaje principal para la implementación de la lógica y manejo de eventos.

Fetch API: para el consumo de datos desde el backend.

SweetAlert2: para generar notificaciones y mensajes de confirmación con una presentación amigable.

Chart.js: para la representación visual de los resultados mediante gráficos dinámicos.

JSON: como formato de intercambio de datos entre cliente y servidor.

Manejo de estados y validaciones

Se implementó un sistema de manejo de estados que permite conservar el progreso del usuario en tiempo real. Por ejemplo, si una encuesta se cierra accidentalmente, el sistema puede restaurar las respuestas almacenadas en el almacenamiento local (localStorage).

Asimismo, se aplicaron validaciones preventivas en los formularios para asegurar la integridad de los datos, evitando campos vacíos, duplicaciones o formatos incorrectos. Estas validaciones fueron tanto del lado del cliente como del servidor, garantizando una experiencia fluida y segura.

Resultados

La correcta implementación de la lógica interactiva permitió que EmoFinance funcionara como una aplicación totalmente dinámica, capaz de responder en tiempo real a las acciones del usuario y ofrecer retroalimentación inmediata. La combinación entre una interfaz atractiva y una lógica eficiente se tradujo en una herramienta funcional, estable y adaptable a futuras expansiones del sistema.

# **Diseño de Base de Datos**

El diseño de la base de datos fue una fase esencial en el desarrollo del proyecto EmoFinance, ya que permitió estructurar, organizar y almacenar de forma eficiente toda la información recopilada a lo largo del proceso de interacción con los usuarios. Su función principal consiste en garantizar la integridad, seguridad y disponibilidad de los datos utilizados por el sistema, especialmente aquellos relacionados con los usuarios, las encuestas y los resultados emocionales obtenidos.

## **Objetivo del diseño**

El objetivo principal del diseño de la base de datos fue construir una estructura relacional que soporte el funcionamiento integral de la aplicación, facilitando la consulta, actualización y análisis de los datos en tiempo real. Además, se buscó establecer relaciones claras entre las distintas entidades del sistema para optimizar el rendimiento de las consultas y minimizar la redundancia de la información.

## **Metodología de diseño**

Para la definición del modelo de datos se siguieron las etapas clásicas del diseño relacional:

Identificación de entidades: se determinaron los elementos principales del sistema, como usuarios, encuestas, emociones y resultados.

Definición de atributos: se especificaron los campos necesarios para cada entidad, asegurando la captura completa de la información relevante.

Establecimiento de relaciones: se definieron las dependencias entre tablas utilizando claves primarias (PK) y foráneas (FK).

Normalización: se aplicaron las tres primeras formas normales para eliminar redundancias y asegurar la consistencia de los datos.

Modelo entidad-relación (E-R)

El modelo E-R del sistema EmoFinance está conformado por las siguientes entidades principales:

Usuario: almacena los datos personales y de autenticación.

Campos: id\_usuario, nombre, correo, contraseña, fecha\_registro, rol.

Encuesta: representa cada formulario diseñado para medir las emociones financieras.

Campos: id\_encuesta, titulo, descripcion, fecha\_creacion, estado.

Pregunta: contiene las preguntas específicas de cada encuesta.

Campos: id\_pregunta, id\_encuesta (FK), texto\_pregunta, tipo\_respuesta.

Respuesta: registra las respuestas seleccionadas o escritas por los usuarios.

Campos: id\_respuesta, id\_pregunta (FK), id\_usuario (FK), valor\_respuesta, fecha\_respuesta.

Emocion: representa las emociones vinculadas a las respuestas de los usuarios.

Campos: id\_emocion, nombre\_emocion, descripcion, intensidad.

Resultado: agrupa los datos procesados para generar los mapas emocionales.

Campos: id\_resultado, id\_usuario (FK), id\_encuesta (FK), emocion\_predominante, puntaje\_total, fecha\_generacion.

## **Relaciones entre entidades**

El diseño relacional del sistema se basa en los siguientes vínculos:

1. Un usuario puede participar en múltiples encuestas.
2. Cada encuesta puede contener varias preguntas.
3. Cada pregunta puede generar múltiples respuestas de distintos usuarios.
4. Una respuesta puede estar asociada a una o más emociones según su interpretación.

Los resultados se generan combinando los datos de usuarios, encuestas y emociones, consolidando la información para su posterior análisis.

## **Gestor de base de datos y estructura técnica**

El sistema utiliza MySQL como gestor principal de base de datos, debido a su robustez, compatibilidad con Node.js y facilidad de integración con aplicaciones web.

Se diseñaron índices en campos clave (como id\_usuario, id\_encuesta y id\_pregunta) para optimizar el tiempo de respuesta en las consultas.

Además, se implementaron restricciones de integridad referencial mediante claves foráneas para garantizar la coherencia de los datos entre tablas relacionadas.

## **Seguridad y manejo de datos**

La base de datos cuenta con políticas de seguridad orientadas a proteger la información sensible de los usuarios.

Entre las principales medidas adoptadas se incluyen:

1. Encriptación de contraseñas mediante algoritmos de hash (como bcrypt).
2. Control de acceso por roles, limitando las operaciones que cada tipo de usuario puede ejecutar.
3. Validación de entradas para prevenir ataques de inyección SQL.
4. Respaldos automáticos programados para asegurar la disponibilidad de la información ante fallos del sistema.

Resultados del diseño

El modelo de base de datos desarrollado para EmoFinance permitió establecer una infraestructura sólida, escalable y segura para la gestión de la información. Gracias a su estructura relacional optimizada, el sistema logra realizar operaciones de lectura y escritura con eficiencia, garantizando un flujo constante de datos entre el frontend, el backend y el servidor.

En conclusión, este diseño constituye la base estructural que sustenta la funcionalidad general de la aplicación, asegurando una correcta administración de los datos y facilitando futuras ampliaciones del sistema.

# **Desarrollo de la API REST / GraphQL**

El desarrollo de la API representó una de las fases más importantes del proyecto EmoFinance, ya que permitió establecer la comunicación entre el frontend, la base de datos y los servicios del servidor. El objetivo principal de esta etapa fue diseñar e implementar un conjunto de endpoints que garantizaran un intercambio de información seguro, eficiente y escalable, facilitando la interacción entre los diferentes módulos del sistema.

## **Objetivo del desarrollo**

El propósito del desarrollo de la API fue proporcionar una interfaz de comunicación estructurada entre el cliente y el servidor, permitiendo a la aplicación gestionar usuarios, encuestas, respuestas y resultados emocionales de forma dinámica. La API actúa como un puente que recibe las solicitudes desde el frontend, procesa la lógica de negocio y devuelve las respuestas necesarias para la visualización de datos en tiempo real.

## **Enfoque tecnológico**

Para la implementación del servicio se evaluaron dos arquitecturas: REST y GraphQL.

Finalmente, se optó por el desarrollo bajo el enfoque API REST, debido a su simplicidad, compatibilidad con los entornos de desarrollo utilizados (Node.js y MySQL) y facilidad para integrarse con sistemas existentes. Sin embargo, se dejó abierta la posibilidad de una futura migración a GraphQL para optimizar consultas más complejas y reducir el número de peticiones al servidor.

La API se construyó utilizando Node.js con el framework Express.js, lo que permitió definir rutas y controladores de manera modular. Cada recurso del sistema cuenta con su propio conjunto de endpoints, definidos con operaciones CRUD (Create, Read, Update, Delete), siguiendo las convenciones del protocolo HTTP.

Los principales endpoints implementados fueron:

Usuarios (/api/usuarios):

POST /registrar → Registro de nuevos usuarios.

POST /login → Autenticación y generación de token.

GET /perfil/:id → Obtención de información del usuario.

PUT /actualizar/:id → Actualización de datos de usuario.

Encuestas (/api/encuestas):

GET / → Listado general de encuestas disponibles.

POST /crear → Creación de nuevas encuestas.

GET /:id → Consulta detallada de una encuesta específica.

DELETE /eliminar/:id → Eliminación de encuestas.

Respuestas (/api/respuestas):

POST /registrar → Almacenamiento de respuestas de usuarios.

GET /usuario/:id → Consulta de respuestas asociadas a un usuario.

Resultados (/api/resultados):

GET /usuario/:id → Visualización del mapa emocional del usuario.

POST /generar → Creación de resultados a partir de respuestas analizadas.

## **Seguridad e implementación**

Para garantizar la integridad de las comunicaciones entre el cliente y el servidor, se aplicaron diversos mecanismos de seguridad:

Autenticación mediante JWT (JSON Web Tokens): cada usuario autenticado recibe un token único que valida sus solicitudes.

Middleware de verificación: protege las rutas privadas y restringe el acceso a usuarios no autorizados.

Cifrado de contraseñas: implementado con el algoritmo bcrypt antes de almacenar los datos en la base de datos.

CORS (Cross-Origin Resource Sharing): configurado para permitir únicamente solicitudes provenientes del dominio oficial de EmoFinance.

## **Formato de intercambio de datos**

Todas las solicitudes y respuestas de la API se manejan en formato JSON, lo que garantiza compatibilidad con cualquier cliente HTTP y facilita el procesamiento de información. Este formato estructurado permite enviar objetos complejos como encuestas completas o mapas emocionales, manteniendo la legibilidad y eficiencia en las transacciones.

Durante el desarrollo de la API se realizaron pruebas exhaustivas utilizando Postman y Thunder Client, con el fin de validar el correcto funcionamiento de cada endpoint.

Se verificó la respuesta de los métodos HTTP, los códigos de estado (200 OK, 201 Created, 400 Bad Request, 401 Unauthorized, 500 Server Error) y la estructura de los objetos retornados.

También se implementaron pruebas de carga para evaluar el rendimiento del servidor bajo múltiples peticiones concurrentes, garantizando la estabilidad del servicio.

Resultados del desarrollo

La API desarrollada permitió una comunicación estable y segura entre los componentes del sistema, optimizando el flujo de información y facilitando la integración con el frontend.

Gracias a su arquitectura modular, el sistema puede adaptarse fácilmente a nuevas funcionalidades, como análisis automatizados de emociones o visualización avanzada de resultados.

En conjunto, esta capa de servicios constituye el núcleo funcional del ecosistema EmoFinance, permitiendo la interacción continua entre el usuario, los datos y el modelo analítico de emociones financieras.

# **Configuración de Servidores y Seguridad**

La configuración de servidores y seguridad constituye una de las fases más relevantes del proyecto EmoFinance, ya que garantiza el correcto funcionamiento del sistema en un entorno de producción, así como la protección de los datos personales y emocionales de los usuarios. En esta etapa se llevó a cabo la integración del servidor de backend con la base de datos alojada en Firebase, la implementación de políticas de seguridad, la configuración de los entornos de despliegue y la optimización del rendimiento general de la aplicación.

## **Arquitectura de despliegue**

El sistema se desarrolló bajo una arquitectura cliente-servidor, en la cual el frontend se comunica con el backend mediante peticiones HTTP a través de la API REST.

El backend, desarrollado con Node.js y Express.js, se despliega en un entorno controlado y seguro que administra las solicitudes, valida los tokens de acceso y gestiona la lógica de negocio.

La estructura general se compone de los siguientes elementos:

Servidor de aplicación (Backend): encargado de procesar las solicitudes y conectarse con Firebase para el manejo de datos.

Firebase Realtime Database / Firestore: gestiona el almacenamiento de datos de usuarios, encuestas y respuestas de forma dinámica y en tiempo real.

Servidor web (Frontend): alojado en un servicio de hosting estático, encargado de presentar la interfaz gráfica y conectarse con la API.

## **Configuración de Firebase**

Firebase fue seleccionado como gestor de base de datos y plataforma de servicios por sus ventajas en escalabilidad, facilidad de integración con aplicaciones JavaScript y seguridad basada en reglas personalizadas.

Durante la configuración se realizaron las siguientes acciones:

Creación del proyecto en Firebase Console, con un entorno diferenciado para desarrollo y producción.

Integración mediante SDK de Firebase en el backend de Node.js, permitiendo la conexión segura a la base de datos a través de credenciales encriptadas.

Definición de reglas de seguridad en Firestore para restringir la lectura y escritura únicamente a usuarios autenticados.

Configuración de almacenamiento adicional (Firebase Storage) para alojar imágenes, gráficos y datos complementarios del mapa emocional.

## **Gestión de entornos y variables**

Para mantener la seguridad y la modularidad del sistema, se implementó un archivo .env en el backend, donde se almacenan las claves privadas, las URLs de Firebase y las configuraciones del servidor.

Estas variables de entorno no se incluyen en el repositorio público, siguiendo las mejores prácticas de seguridad y cumplimiento de políticas de privacidad.

Imagen del código del back

# **Mecanismos de seguridad implementados**

Para garantizar la confidencialidad e integridad de los datos del usuario, se aplicaron diversas medidas de protección a nivel de servidor y base de datos:

Autenticación con Firebase Authentication: permite registrar y validar usuarios mediante correo electrónico y contraseña o a través de proveedores externos como Google.

Reglas de seguridad en Firestore: controlan los permisos de lectura y escritura en función del estado de autenticación del usuario.

Conexión segura HTTPS: todas las comunicaciones entre el cliente, el servidor y Firebase se realizan bajo el protocolo HTTPS, evitando la interceptación de datos.

Cifrado de credenciales: tanto las contraseñas como los tokens de autenticación se encriptan utilizando algoritmos hash antes de ser almacenados.

Gestión de sesiones: se implementó un sistema de expiración de tokens JWT para evitar el acceso prolongado no autorizado.

Validaciones en el backend: cada petición enviada al servidor es validada antes de procesarse, asegurando que solo usuarios autenticados puedan acceder a sus propios datos.

## **Monitoreo y mantenimiento**

Firebase proporciona herramientas integradas de monitoreo, como Firebase Analytics y Crashlytics, que permiten observar el comportamiento del sistema, registrar errores y analizar el rendimiento en tiempo real.

Además, se configuraron registros de actividad para detectar accesos no autorizados y patrones de uso inusuales, fortaleciendo la seguridad preventiva del sistema.

El servidor también cuenta con un sistema de monitoreo de logs, implementado mediante Winston en Node.js, que permite registrar errores, solicitudes y respuestas para análisis posterior.

## **Resultados y beneficios**

La configuración del servidor y la integración con Firebase permitieron obtener una infraestructura moderna, segura y altamente escalable.

El uso de Firebase como plataforma de datos redujo la complejidad de la gestión de servidores, garantizando una conexión estable y una experiencia fluida para los usuarios finales.

Gracias a la correcta implementación de políticas de autenticación, encriptación y control de accesos, EmoFinance cumple con altos estándares de seguridad, minimizando riesgos asociados a la pérdida o exposición de información sensible.

En conjunto, esta fase consolidó la base técnica y de seguridad del sistema, asegurando su funcionamiento continuo, confiable y adaptable a futuras expansiones o integraciones.

# **Integración del Frontend y Backend**

La integración del frontend y backend representó una de las etapas más determinantes del desarrollo del proyecto EmoFinance, ya que permitió consolidar en un solo entorno funcional todos los componentes del sistema: la interfaz de usuario, la lógica del servidor y la gestión de datos. Esta fase tuvo como propósito principal lograr una comunicación fluida y segura entre las capas de presentación y la de servicios, garantizando así la correcta ejecución de las operaciones dentro de la aplicación.

## **Objetivo de la integración**

El objetivo de esta fase fue conectar la interfaz visual del usuario (frontend) con la lógica de negocio y los servicios de datos (backend y Firebase), permitiendo que todas las funcionalidades —registro, encuestas, respuestas y visualización de emociones— se realizaran en tiempo real y con una sincronización constante de la información.

## **Proceso de integración**

La integración se realizó de manera progresiva, dividiéndose en las siguientes etapas:

Conexión inicial de la API REST con el frontend: Se establecieron los endpoints necesarios para manejar solicitudes de registro, inicio de sesión, envío de respuestas y obtención de resultados. El frontend realizó peticiones HTTP mediante la función fetch() para consumir los datos de la API.

Configuración de autenticación y sesiones: Una vez autenticado el usuario, el backend genera un token JWT (JSON Web Token), el cual se almacena temporalmente en el navegador (localStorage) para validar las acciones del usuario en toda la aplicación.

Integración con Firebase: El backend se comunica directamente con Firebase Firestore para almacenar y consultar datos. Desde el frontend, los usuarios pueden interactuar con estos datos de forma dinámica gracias a la conexión en tiempo real que ofrece Firebase.

Sincronización de componentes UI: Los componentes visuales del frontend se actualizaron en función de las respuestas del servidor. Por ejemplo, al completar una encuesta, el sistema procesa las emociones asociadas y muestra inmediatamente los resultados visuales mediante gráficos generados con Chart.js.

Gestión de errores y validaciones: Se implementaron mecanismos de manejo de errores en ambos extremos. Si el servidor detecta una solicitud inválida o un token expirado, el frontend muestra alertas interactivas y redirige al usuario a la página de autenticación.

## **Comunicación entre capas**

El flujo de comunicación entre el frontend y el backend se desarrolla bajo el siguiente esquema:

El usuario ejecuta una acción (como responder una pregunta o generar un resultado).

El frontend envía una petición HTTP al servidor mediante un endpoint definido.

El backend valida la solicitud, ejecuta la lógica correspondiente y se comunica con Firebase para obtener o modificar los datos necesarios.

El servidor envía una respuesta en formato JSON al cliente, conteniendo los datos solicitados o la confirmación de la operación.

El frontend procesa la respuesta y actualiza la interfaz dinámicamente.

Este flujo garantiza una interacción continua y bidireccional entre el cliente y el servidor, reduciendo los tiempos de carga y mejorando la experiencia del usuario.

## **Tecnologías empleadas**

Para lograr una integración óptima, se utilizaron las siguientes tecnologías:

Frontend: HTML, CSS, JavaScript, Chart.js, SweetAlert2 y Figma como referencia de diseño.

Backend: Node.js con Express.js.

Base de datos: Firebase Firestore y Firebase Authentication.

Intercambio de datos: JSON.

Control de versiones: GitHub, utilizado para el despliegue continuo y sincronización de cambios entre frontend y backend.

## **Pruebas de integración**

Durante esta fase se realizaron pruebas exhaustivas para asegurar el correcto funcionamiento de los endpoints y la respuesta de la interfaz.

Se verificaron aspectos como:

* La correcta transmisión de datos entre cliente y servidor.
* El manejo de errores en las solicitudes.
* La validación de usuarios autenticados antes de permitir operaciones sensibles.
* La actualización dinámica de los componentes UI después de cada respuesta del servidor.

Para las pruebas se utilizaron herramientas como Postman, Firebase Emulator Suite y console debugging del navegador.

## **Resultados de la integración**

La integración del frontend y backend permitió consolidar EmoFinance como un sistema completamente funcional e interactivo.

El flujo de información entre las capas es estable, seguro y eficiente, posibilitando la actualización en tiempo real de los mapas emocionales y el análisis de decisiones financieras de los usuarios.

Esta etapa aseguró que todos los módulos del sistema trabajaran de forma coordinada, cumpliendo con los requerimientos funcionales y garantizando una experiencia fluida y coherente para el usuario final.

# **Gestión de Riesgos**

## **Descripción general**

La gestión de riesgos del proyecto EmoFinance tuvo como propósito identificar, evaluar y mitigar los posibles eventos que pudieran afectar negativamente el desarrollo del sistema.

Dado que el proyecto combina aspectos de investigación emocional, diseño interactivo y desarrollo tecnológico, resultó fundamental establecer un marco estructurado para anticipar problemas técnicos, operativos o de planificación.

La Evaluación de Riesgos se elaboró con base en los lineamientos del PMBOK (Project Management Body of Knowledge), abarcando las fases de identificación, análisis cualitativo, planificación de respuestas y monitoreo continuo.

## **Objetivos de la gestión de riesgos**

* Prever los posibles obstáculos que puedan afectar los entregables o el cronograma del proyecto.
* Establecer estrategias de prevención y mitigación para minimizar el impacto de los riesgos.
* Asegurar la continuidad del desarrollo y la calidad de los resultados.
* Fomentar una cultura de anticipación y control dentro del equipo de trabajo.

## **Metodología aplicada**

La gestión de riesgos se estructuró en las siguientes etapas:

Identificación: se recopilaron riesgos potenciales a través de sesiones de revisión con el equipo (diseño, desarrollo, base de datos y seguridad).

Evaluación cualitativa: cada riesgo fue analizado en función de su probabilidad de ocurrencia y su impacto en el proyecto, asignando una escala del 1 al 5.

Priorización: mediante una matriz de criticidad se determinaron los riesgos de alta, media y baja prioridad.

Plan de mitigación: se definieron acciones preventivas y correctivas para cada riesgo identificado.

Monitoreo: se estableció un control periódico para verificar el estado de cada riesgo durante la ejecución del proyecto.

Análisis general

El análisis de los riesgos permitió identificar que los mayores niveles de criticidad se concentraron en las fases de integración tecnológica y seguridad de datos, debido al uso de Firebase y al alto grado de interacción entre el frontend y el backend.

El riesgo operativo por retrasos también se consideró significativo, especialmente en los periodos de pruebas y validación.

Gracias a la aplicación de estrategias preventivas como la integración continua en GitHub, los respaldos automáticos y la revisión constante del código, se logró mantener los riesgos en un nivel controlado y mitigado.

Tabla

El contenido generado por IA puede ser incorrecto.

# **Tabla de Seguimiento del Proyecto**

## **Descripción general**

El tablero de seguimiento del proyecto EmoFinance se desarrolló como una herramienta de control y supervisión de avances en cada fase del proyecto.

Su propósito fue garantizar la trazabilidad de las actividades, monitorear los tiempos de ejecución y verificar el cumplimiento de las tareas planificadas en el cronograma.

El tablero permitió visualizar de manera clara los porcentajes de avance, los responsables asignados, el estado actual de cada tarea y los comentarios de control, facilitando la toma de decisiones por parte del líder del proyecto.

## **Objetivos del seguimiento**

* Monitorear el progreso de cada fase del proyecto según los plazos establecidos.
* Identificar desviaciones en tiempo, recursos o calidad de entregables.
* Facilitar la comunicación y coordinación entre los miembros del equipo.
* Documentar los avances de manera transparente y continua.
* Evaluar el cumplimiento de los objetivos específicos definidos en la planificación inicial.

## **Metodología de seguimiento**

El seguimiento se realizó semanalmente, utilizando una hoja de control en Microsoft Excel, complementada con actualizaciones en GitHub y reuniones internas.

Cada actividad fue clasificada de acuerdo con su estado de ejecución:

* Pendiente: tareas planificadas que aún no han iniciado.
* En proceso: actividades actualmente en desarrollo.
* Finalizado: tareas completadas y validadas por el responsable o líder.

A cada actividad se le asignó un porcentaje de avance, un responsable y observaciones específicas para registrar dificultades o ajustes realizados.

Interfaz de usuario gráfica, Aplicación, Tabla, Excel

El contenido generado por IA puede ser incorrecto.

## **Análisis de avances**

De acuerdo con el seguimiento realizado, el proyecto presenta un avance general del 78%, evidenciando un desarrollo constante y coordinado entre los equipos de diseño y programación.

Las fases iniciales de diseño, modelado y prototipado fueron ejecutadas y aprobadas en su totalidad, mientras que las etapas técnicas relacionadas con API, seguridad e integración se encuentran en curso.

El principal reto identificado es la sincronización final entre el frontend y backend, así como la optimización de la seguridad en Firebase para garantizar la protección de datos de los usuarios.

## **Conclusión del seguimiento**

El proceso de seguimiento ha permitido mantener una visión clara del estado real del proyecto, favoreciendo la identificación temprana de riesgos y el cumplimiento de plazos.

El uso de herramientas de gestión como GitHub, Excel y Figma, junto con la comunicación constante entre roles, ha sido clave para el control del progreso.

Con las fases críticas de desarrollo próximas a finalizar, se proyecta que el sistema EmoFinance alcance su despliegue completo dentro del cronograma establecido, cumpliendo con los estándares de calidad, seguridad y funcionalidad definidos en los objetivos iniciales.

# **Tabla de Presupuesto**