Gestor de Gastos

Prototipo

Se plantea el desarrollo de un gestor que permita a los usuarios registrar movimientos financieros, que permita controlar el consumo según el presupuesto que defina el usuario. Además de datos básicos como la fecha, monto, tipo de movimiento, clasificación y moneda utilizada, se incluyen también datos de geolocalización tanto para movimiento y el tercero, que permiten identificar dónde y para quién fue realizado el movimiento. El gestor permite operar con monedas extranjeras, integrando automáticamente la conversión de valores a la moneda local según el tipo de cambio vigente al momento del registro. El diseño minimalista y la funcionalidad sin conexión eliminan demoras a la hora de realizar el registro, permitiendo al usuario realizar clasificar y completar detalles de los movimientos luego de su registro. Con los datos completos, el sistema genera resúmenes periódicos para facilitar el análisis del consumo.

Requerimientos

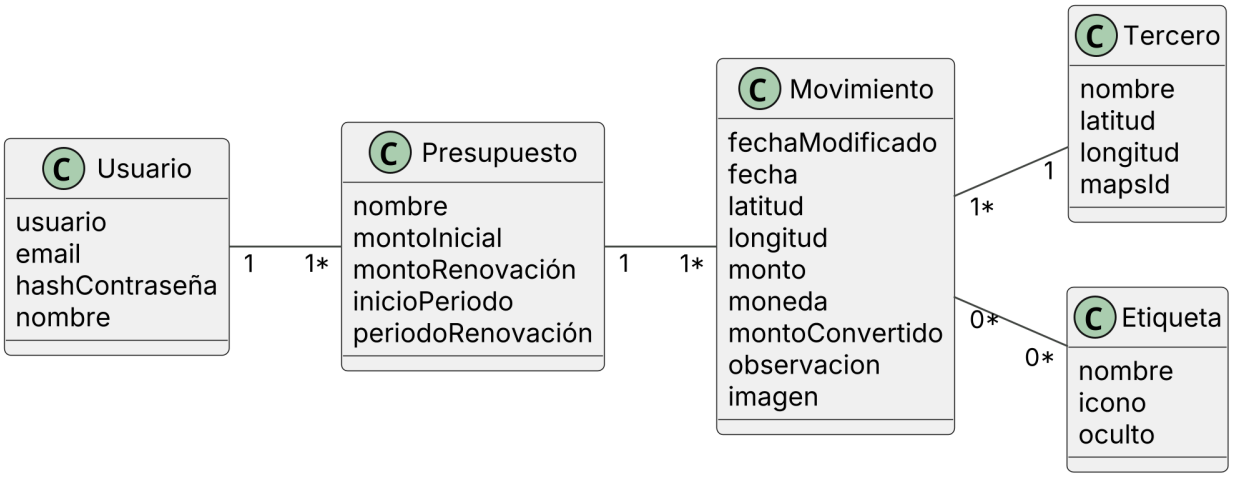
Funcionales: (se destacan los requisitos trabajados en el prototipo actual)

- Controlar los consumos e ingresos del usuario comparando con un presupuesto estimado por el usuario que se renueva automáticamente cada cierto periodo de tiempo.
- **Registrar los datos básicos de un movimiento, si es consumo o ingreso, la fecha, el lugar, el monto y que moneda.**
- Editar datos de movimientos existentes para corregir y agregar más información, la índole del movimiento, datos del tercero, etiquetas definidas por el usuario, comprobantes o notas.
- **Saber cuándo, dónde y con qué tercero se realizó cierto movimiento.**
- Registrar movimientos usando un comprobante de compra o transferencia.
- Exportar e importar datos en formato de tabla.
- **Comparar el valor en moneda local, histórico y actual para movimientos en moneda extranjera.**
- Generar resúmenes de los movimientos para distintos periodos de tiempo, con totales, cuáles fueron los mayores consumos o ingresos y agrupándolos por índole.

No funcionales:

- Debe funcionar sin conexión a internet.
- Debe realizar un respaldo periódico y privado de los datos fuera del dispositivo.
- La importación de los datos debe tener retrocompatibilidad
- La carga debe ser rápida para el usuario.

Diseño



Caso de Uso: Carga de un movimiento		
ID: 1	Fecha: 08/05/2025	
Descripción: El usuario ingresa los datos de un movimiento para registrar.		
Actores Principales: Usuario	Actores Secundarios: Maps API, Dólar API	
Observaciones:		
Precondiciones: Existe un **Presupuesto** para asociar el movimiento		
Post- Condiciones	Éxito: Se registra un nuevo movimiento en la base de datos	
	Fracaso: No se registra el movimiento	
Flujo principal		Flujo Alternativo
1. El caso de uso comienza cuando el usuario presiona “Cargar” en la vista de movimientos de un presupuesto		
2. La aplicación solicita permiso para recolectar datos de geolocalización		
3. El usuario otorga el permiso de geolocalización, la aplicación toma la latitud y longitud		3.1 El usuario no otorga el permiso. 3.2 Se bloquea la carga del movimiento. 3.3 Finaliza el caso de uso
4. El usuario ingresa el monto para el movimiento realizado y selecciona la moneda.		
5. El usuario no decide buscar lugares cercanos a su ubicación actual		5.1 El usuario decide buscar lugares cercanos a su ubicación actual 5.2 Envía la ubicación del usuario a la API de Google Maps, recupera un listado de lugares cercanos 5.3 Crea una nueva instancia de **Tercero** guardando la *mapsId* (identifica el lugar en la API), *nombre*, *latitud* y *longitud*
6. El usuario carga el movimiento.		

7. El movimiento se realizó en moneda local y no requiere de conversión	7.1 El movimiento se realizó en moneda extranjera (dólares) 7.2 Recupera con éxito los valores de cotización de la Dólar API 7.3 Convierte el monto a moneda local usando el valor de venta si se trata de un egreso, o el valor de compra si se trata de un egreso.
8. El usuario carga el movimiento, se crea una nueva instancia de **Movimiento** usando la *fecha* de hoy, la *latitud* y *longitud* obtenidas previamente, la *moneda* y el *monto* ingresado, el *montoConvertido* si aplica, dejando *observacion* e *imagen* con valor nulo. Asocia la nivea instancia al **Presupuesto** seleccionado.	
9. Finaliza el caso de uso	

Caso de Uso: Lista de movimientos		
ID: 2	Fecha: 08/05/2025	
Descripción: El usuario decide ver sus movimientos		
Actores Principales: Usuario	Actores Secundarios: Dólar API	
Observaciones:		
Precondiciones: Existe un **Presupuesto**.		
Post- Condiciones	Éxito: Muestra el listado de los movimientos	
	Fracaso: No muestra el listado	
Flujo principal		Flujo Alternativo
1. El caso de uso comienza cuando ingresa a la vista de movimientos de un **Presupuesto**		
2. La aplicación recupera los movimientos asignados al **Presupuesto** actual.		
3. Todos los **Movimientos** hechos en moneda extranjera (dólar) tienen un *montoConvertido* no nulo		3.1 Existen movimientos hechos en moneda extranjera (dólar) con *montoConvertido* nulo 3.2 Recupera con éxito los valores de cotización históricos de la Dólar API para la *fecha* del **Movimiento** 3.3 Convierte el monto a moneda local usando el valor de venta si se trata de un egreso, o el valor de compra si se trata de un egreso. 3.4 Actualiza la instancia de **Movimiento**.
4. Muestra la *moneda*, el *monto*, *montoConvertido* si aplica y *fecha* de cada **Movimiento**.		

5. Finaliza el caso de uso.	

Caso de Uso: Información de un movimiento		
ID: 3		Fecha: 08/05/2025
Descripción: El usuario decide ver el detalle de un movimiento		
Actores Principales: Usuario		Actores Secundarios: Maps API
Observaciones:		
Precondiciones: Existe el **Movimiento**.		
Post- Condiciones	Éxito: Muestra información del movimiento	
	Fracaso: No muestra la información	
Flujo principal		Flujo Alternativo
1. El caso de uso comienza presionando un movimiento en el listado de movimientos.		
2. La aplicación recupera los datos del movimiento seleccionado.		
3. No tiene asignado un **Tercero**, utiliza los datos de *latitud* y *longitud* del **Movimiento**.		3.1 Tiene asignado un **Tercero**, utiliza los datos de *latitud*, *longitud* del mismo. 3.2 Muestra el *nombre* del tercero
4. Usando la API de Google Maps, muestra un mapa marcando la ubicación según los valores de *latitud* y *longitud* determinados y el *nombre* si aplica		
5. Muestra la *id*, la *moneda*, el *monto*, *montoConvertido* si aplica y *fecha* del **Movimiento**.		
6. Finaliza el caso de uso		

Validación

Casos de Testing

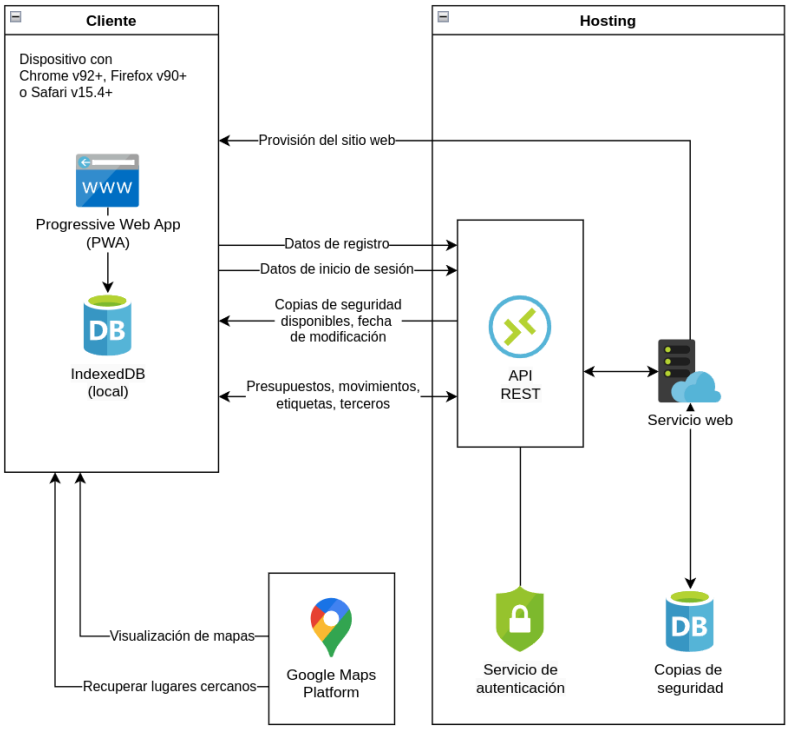
Mockup

Mockup Gestor de Costos

Arquitectura

Emplea la arquitectura Cliente-Servidor. En el servidor se almacenan los presupuestos, movimientos, etiquetas y datos de terceros cargados por los usuarios. Cada usuario puede crear múltiples presupuestos, y solo puede recuperar datos de presupuestos propios. La autenticación se realiza usando Firebase Authentication.

El cliente es una Aplicación Web Progresiva (PWA), esto permite agregar funcionamiento sin conexión a la aplicación. Se almacenan en caché los últimos movimientos, etiquetas y terceros pertenecientes a un presupuesto y permanecen en solo lectura. No es posible editar movimientos sin conexión, si es posible crear nuevos movimientos, estos se guardan localmente hasta que se vuelve a obtener conexión con el servidor.



El cliente se comunica con el servidor a través del protocolo HTTP, transfiriendo datos codificados en JSON (las fechas siendo strings que respetan la ISO 8601) a una API REST.

Método	Ruta	Descripción
POST	/api/register	Registrar a un usuario, toma el usuario, email, contraseña y el resultado de un captcha.
POST	/api/login	Iniciar sesión, toma usuario/email y contraseña, devuelve un JSON Web Token en una cookie.
GET	/api/yo	Devuelve los datos del usuario
GET	/api/presupuestos	Devuelve los presupuestos que tiene el usuario
GET	/api/presupuestos/{p_id}	Devuelve los datos de un presupuesto en particular y movimientos
POST	/api/presupuestos/{p_id}	Crea un nuevo presupuesto
PUT	/api/presupuestos/{p_id}	Actualiza los datos de un presupuesto
GET	/api/presupuestos/{p_id}/movimientos	Devuelve una lista de los movimientos (solo monto, fecha y etiquetas) asignados a un presupuesto, en páginas, con parámetros de filtrado
POST	/api/presupuestos/{p_id}/movimientos	Crea un nuevo movimiento, y lo asigna al presupuesto determinado
GET	/api/presupuestos/{p_id}/etiquetas	Devuelve las etiquetas que existen en un presupuesto
POST	/api/presupuestos/{p_id}/etiquetas	Crea una nueva etiqueta, y la asigna al presupuesto determinado
GET	/api/presupuestos/{p_id}/terceros	Devuelve los terceros que existen en un presupuesto
POST	/api/presupuestos/{p_id}/terceros	Crea un nuevo tercero, y lo asigna al presupuesto determinado
GET	/api/presupuestos/{p_id}/reporte	Devuelve un reporte de gastos e ingresos para un periodo determinado
GET	/api/movimientos/{m_id}	Devuelve los datos de un movimiento
PUT	/api/movimientos/{m_id}	Actualiza los datos de un movimiento
DELETE	/api/movimientos/{m_id}	Borra un movimiento

PUT	/api/etiquetas/{e_id}	Actualiza los datos de una etiqueta existentes
DELETE	/api/etiquetas/{e_id}	Borra una etiqueta y sus referencias en movimientos asociados a ella
GET	/api/terceros/{t_id}	Devuelve los datos de un tercero
PUT	/api/terceros/{t_id}	Actualiza los datos de un tercero
DELETE	/api/terceros/{t_id}	Borra un tercero y sus referencias en movimientos asociados a ella