Documento de Requisitos de Software (SRD)

Proyecto: App Web de Control de Finanzas StonkyStonk

Creado por Alejandro Hernández, Matías Cáceres y Diego Orellana.

1. Descripción del Proyecto

Nombre tentativo: *StonkyStonk*

Resumen:
StonkStonky es una aplicación web robusta de control financiero personal y pequeño empresarial. Permite a los usuarios registrar ingresos y gastos, detectar patrones financieros, definir metas, establecer presupuestos inteligentes, recibir alertas y obtener reportes de análisis comparativos y predictivos.

Propósito:
Proporcionar una solución integral de gestión financiera que automatice procesos, analice datos financieros de forma avanzada y promueva decisiones inteligentes de ahorro y gasto.

Alcance:
Incluye: autenticación, dashboard financiero, registros automáticos y manuales, visualizaciones avanzadas, alertas, metas, presupuestos, exportación de reportes y sistema de recomendaciones.
No incluye: integración bancaria directa en esta versión.

Público objetivo:
Usuarios individuales, emprendedores, estudiantes, y microempresarios que buscan tener un mejor control de sus finanzas.

2. Objetivos de Negocio

- Facilitar el control de gastos e ingresos sin hojas de cálculo.
- Detectar automáticamente gastos repetitivos o irregulares.
- Mejorar el hábito de ahorro mediante metas y presupuestos.
- Ayudar a proyectar el flujo de caja personal o de un negocio.

3. Requisitos del Usuario

ID	Historia de Usuario	Criterios de aceptación	Prioridad

RU01a	Como usuario deseo registrar ingresos manuales indicando monto, fecha y descripción.	• El sistema debe permitir ingresar monto (número positivo), fecha (por defecto la actual, editable) y descripción opcional. • El ingreso debe guardarse en la base de datos y actualizar el saldo automáticamente. • El usuario debe recibir un mensaje de confirmación tras registrar un ingreso.	
RU01b	Como usuario deseo registrar egresos manuales indicando monto, fecha y descripción.	• El sistema debe permitir ingresar monto (número positivo), fecha (por defecto la actual, editable) y descripción opcional. • El egreso debe guardarse en la base de datos y descontarse del saldo automáticamente. • El usuario debe recibir un mensaje de confirmación tras registrar un egreso.	
RU02	Como usuario quiero que el sistema detecte gastos repetitivos como suscripciones.	• El sistema debe identificar transacciones con mismo monto y categoría que se repitan de forma periódica (ej. mensual). • El sistema debe notificar al usuario cuando detecte una suscripción o gasto recurrente. • El usuario debe poder confirmar si un gasto es recurrente o descartarlo.	
RU03	Como usuario quiero recibir alertas si me excedo del presupuesto mensual.	• El sistema debe comparar gastos acumulados contra el presupuesto configurado. • El sistema debe enviar una alerta al llegar al 80% y otra al 100% del presupuesto. • La alerta debe mostrarse en el dashboard y enviarse por correo/push si el usuario lo habilita.	
RU04	Como usuario quiero establecer metas financieras con un objetivo y plazo.	• El usuario debe poder definir monto objetivo, fecha límite y categoría opcional. • La meta debe guardarse en la base de datos y mostrarse en el dashboard.	

		• El sistema debe calcular automáticamente el progreso de la meta.	
RU05	Como usuario deseo ver el progreso de mis metas en el dashboard.	• El sistema debe mostrar el porcentaje alcanzado, monto ahorrado y monto restante. • El progreso debe actualizarse automáticamente cuando se registren ingresos/egresos asociados. • El usuario debe ver todas sus metas activas en el dashboard.	
RU06a	Como usuario quiero analizar mi flujo de caja mensual.	• El sistema debe mostrar ingresos, egresos y saldo disponible en un rango mensual. • El flujo debe visualizarse en formato tabla y gráfico (ej. barras o líneas). • El usuario debe poder filtrar por mes y año.	
RU06b	Como usuario quiero recibir sugerencias automáticas basadas en mi flujo de caja mensual.	• El sistema debe generar al menos una sugerencia por mes. • Las sugerencias deben estar basadas en patrones (ej. reducción de gastos, optimización de ahorro). • El usuario debe ver las sugerencias en el dashboard y poder marcarlas como útiles o ignorarlas.	
RU07a	Como usuario quiero descargar reportes financieros en PDF.	• El sistema debe permitir seleccionar período (ej. mes, año, rango personalizado). • El archivo PDF debe incluir ingresos, egresos, saldo y gráficos. • El archivo debe generarse en menos de 10 segundos.	
RU07b	Como usuario quiero descargar reportes financieros en Excel.	• El sistema debe exportar transacciones detalladas en formato Excel (.xlsx). • El archivo debe incluir filtros por categorías y fechas. • El nombre del archivo debe incluir la fecha de generación.	

RU08	Como usuario quiero comparar mis gastos actuales con los de meses anteriores.	• El sistema debe mostrar un gráfico comparativo entre el mes seleccionado y los anteriores. • El usuario debe poder elegir qué meses comparar. • El sistema debe resaltar las categorías con mayor variación de gasto.	
RU09	Como usuario quiero configurar mi presupuesto mensual.	• El usuario debe poder establecer un monto total de presupuesto mensual. • El sistema debe permitir configurar presupuestos por categoría (ej. comida, transporte). • El presupuesto debe guardarse en la base de datos y reflejarse en alertas de gasto.	
RU10	Como usuario deseo poder registrarme con correo y contraseña.	• El formulario debe solicitar correo válido, contraseña y confirmación de contraseña. • El sistema debe validar que el correo no esté registrado previamente. • Tras registrarse, el usuario debe recibir un correo de confirmación.	
RU11	Como usuario deseo poder iniciar sesión con correo y contraseña.	• El sistema debe autenticar usando correo y contraseña registrados. • El inicio de sesión exitoso debe generar un token de sesión válido. • Si las credenciales son inválidas, debe mostrarse un mensaje de error claro.	
RU12	Como usuario deseo poder recuperar mi contraseña vía correo electrónico.	• El usuario debe poder solicitar la recuperación ingresando su correo. • El sistema debe enviar un enlace único y temporal para restablecer la contraseña. • El enlace debe expirar en un tiempo definido (ej. 24 horas). • El usuario debe poder establecer una nueva contraseña y confirmarla.	

4. Requisitos Funcionales

ID	Descripción	Criterios de aceptación	Prioridad
RF01a	El sistema debe permitir el registro de usuario con email, contraseña y confirmación de contraseña.	• El email debe validarse con formato correcto (ejemplo@dominio.com). • La contraseña debe cumplir requisitos mínimos (ej. al menos 8 caracteres, 1 mayúscula, 1 número). • El sistema debe verificar que el correo no exista en la base de datos. • El usuario debe recibir un mensaje de confirmación de registro.	
RF01b	El sistema debe enviar un enlace o código de verificación al correo del usuario registrado.	• El correo debe enviarse automáticamente después del registro. • El enlace de verificación debe tener validez limitada (ej. 24 horas). • El usuario no debe poder iniciar sesión sin haber verificado el correo.	
RF01c	El sistema debe permitir iniciar sesión con email y contraseña válidos.	• El sistema debe autenticar credenciales contra la base de datos. • Si son correctas, debe generarse un token JWT válido. • Si son incorrectas, debe mostrarse un error claro (“Correo o contraseña incorrectos”).	
RF01d	El sistema debe generar y validar tokens JWT para mantener sesiones seguras.	• El token debe generarse al iniciar sesión exitosamente. • El token debe tener fecha de expiración configurable (ej. 1 hora). • El sistema debe validar el token en cada solicitud a rutas protegidas.	
RF01e	El sistema debe incluir middleware de autenticación para proteger las rutas privadas.	• El middleware debe validar que el token JWT sea válido y no esté expirado. • Si el token es inválido o falta, la ruta debe devolver error 401 Unauthorized. • El middleware debe integrarse en todas las rutas privadas.	

RF01f	El sistema debe permitir la recuperación de contraseña vía correo electrónico.	• El usuario debe solicitar recuperación ingresando su correo. • El sistema debe enviar un enlace único y temporal (ej. válido por 24 horas). • El usuario debe poder establecer y confirmar una nueva contraseña.	
RF01g	El sistema debe permitir cerrar sesión invalidando el token JWT.	• El sistema debe eliminar o marcar como inválido el token usado. • El usuario debe ser redirigido a la pantalla de login tras cerrar sesión. • Si el usuario intenta usar un token inválido, debe recibir error 401 Unauthorized.	
RF02a	El dashboard debe mostrar el saldo actual (ingresos – gastos).	• El saldo debe calcularse automáticamente con cada transacción registrada. • El saldo debe mostrarse en formato moneda local (ej. CLP, USD). • El saldo debe actualizarse en tiempo real sin recargar la página.	
RF02b	El dashboard debe mostrar un gráfico de tendencias de gastos mensuales.	• El gráfico debe incluir al menos los últimos 6 meses. • El usuario debe poder elegir el rango de meses a visualizar. • El gráfico debe diferenciar ingresos y egresos con colores distintos.	
RF02c	El dashboard debe mostrar una lista de alertas financieras recientes.	• Las alertas deben incluir tipo (ej. presupuesto excedido, gasto recurrente detectado). • Cada alerta debe indicar fecha y descripción breve. • Las alertas deben poder marcarse como leídas o descartarse.	

RF02d	El dashboard debe mostrar un resumen de gastos por categoría en gráfico circular.	• El gráfico debe mostrar la distribución porcentual de cada categoría de gasto. • El usuario debe poder pasar el cursor sobre cada categoría para ver el detalle (monto y porcentaje). • El gráfico debe actualizarse automáticamente con cada transacción registrada.	
RF03a	El sistema debe permitir crear transacciones con monto, descripción, fecha, categoría y etiquetas personalizadas.	• El usuario debe poder registrar monto (positivo), descripción opcional, fecha (por defecto la actual), categoría y etiquetas libres. • El sistema debe validar que monto y fecha sean campos obligatorios. • La transacción debe guardarse en la base de datos y reflejarse en el saldo inmediatamente.	
RF03b	El sistema debe permitir editar transacciones existentes.	• El usuario debe poder modificar monto, descripción, fecha, categoría y etiquetas. • Los cambios deben guardarse en la base de datos y reflejarse en el saldo. • Debe registrarse la fecha de última modificación de la transacción.	
RF03c	El sistema debe permitir eliminar transacciones de forma suave o permanente.	• Eliminación suave: la transacción se marca como “inactiva” pero no se borra de la base de datos. • Eliminación permanente: la transacción se borra completamente. • El usuario debe confirmar antes de eliminar de forma permanente.	
RF03d	El sistema debe permitir listar transacciones con opciones de búsqueda y filtrado.	• El listado debe mostrar monto, descripción, categoría, etiquetas y fecha. • El usuario debe poder filtrar por rango de fechas, categoría y etiquetas. • El usuario debe poder buscar transacciones por palabras clave en la descripción. • El sistema debe ordenar los resultados por fecha (más reciente primero).	

RF04a	El sistema debe sugerir categorías automáticas al agregar una transacción.	• La sugerencia debe basarse en el historial de transacciones previas del usuario. • La categoría sugerida debe mostrarse por defecto en el formulario de creación de transacción. • El usuario debe poder aceptar la categoría sugerida o elegir otra manualmente.	
RF04b	El sistema debe permitir al usuario confirmar o corregir categorías sugeridas en una transacción.	• El sistema debe registrar si el usuario corrigió la categoría sugerida. • La corrección debe alimentar el algoritmo para mejorar futuras sugerencias. • El usuario debe poder cambiar la categoría en cualquier momento.	
RF04c	El sistema debe generar un reporte de patrones de gasto detectados en las transacciones.	• El reporte debe mostrar categorías con gastos recurrentes o elevados. • El usuario debe poder ver variaciones de gasto en períodos definidos (ej. semanal, mensual). • El reporte debe estar disponible en el dashboard y exportable a PDF/Excel.	
RF05a	El sistema debe permitir crear metas financieras con monto objetivo, fecha límite y categoría asociada.	• El formulario debe solicitar monto objetivo (número positivo), fecha límite (no puede ser anterior a la fecha actual) y categoría opcional. • El sistema debe validar que la fecha límite sea válida y el monto sea mayor a 0. • La meta debe guardarse en la base de datos con estado "activa". • El usuario debe recibir confirmación tras crear la meta.	
RF05b	El sistema debe calcular y mostrar el progreso de las metas automáticamente.	• El progreso debe calcularse como porcentaje: (monto ahorrado / monto objetivo) × 100. • El sistema debe mostrar monto objetivo, monto ahorrado actual y monto restante.	

		• El progreso debe actualizarse automáticamente cuando se registren transacciones relacionadas. • La visualización debe incluir barra de progreso visual.	
RF05c	El sistema debe generar alertas automáticas cuando se alcanzan hitos de progreso en las metas.	• El sistema debe enviar alerta al alcanzar 25%, 50%, 75% y 100% del objetivo. • Las alertas deben mostrarse en el dashboard y opcionalmente por correo/push. • El usuario debe poder configurar qué porcentajes de hito desea recibir. • Las alertas deben incluir nombre de la meta, progreso actual y monto restante.	
RF05d	El sistema debe permitir editar y gestionar metas existentes.	• El usuario debe poder modificar monto objetivo, fecha límite y categoría de metas activas. • El sistema debe permitir pausar, reactivar o eliminar metas. • Al modificar una meta, el progreso debe recalcularse automáticamente. • El sistema debe mantener historial de cambios en las metas.	
RF05e	El sistema debe mostrar un resumen visual de todas las metas en el dashboard.	• El dashboard debe mostrar todas las metas activas con su progreso. • Cada meta debe mostrar nombre, progreso porcentual, días restantes y estado. • Las metas próximas a vencer (menos de 7 días) deben resaltarse visualmente. • El usuario debe poder acceder a los detalles de cada meta desde el resumen.	
RF06a	El sistema debe analizar el historial de gastos del usuario para sugerir presupuestos automáticos por categoría.	• El sistema debe analizar al menos 3 meses de historial de transacciones. • El análisis debe calcular el promedio de gastos por categoría y sugerir un presupuesto basado en este dato. • El sistema debe mostrar la sugerencia junto con la justificación (ej. "Basado en tu promedio de $50,000 en Alimentación").	

RF06b	El sistema debe permitir al usuario configurar presupuestos mensuales totales y por categoría de forma manual.	• El formulario debe permitir establecer un presupuesto mensual total y presupuestos específicos por cada categoría. • El sistema debe validar que la suma de presupuestos por categoría no exceda el presupuesto total. • El usuario debe poder activar/desactivar el seguimiento de presupuesto por categoría. • Los presupuestos deben guardarse en la base de datos y aplicarse al mes actual y futuros.	
RF06c	El sistema debe generar alertas automáticas cuando el gasto en una categoría alcance porcentajes específicos del presupuesto.	• El sistema debe enviar alertas al alcanzar 50%, 80% y 100% del presupuesto de cada categoría. • Las alertas deben mostrarse en el dashboard con el porcentaje exacto usado y monto restante. • El usuario debe poder configurar los porcentajes de alerta personalizados por categoría. • Las alertas deben enviarse opcionalmente por correo o notificación push si está habilitado.	
RF06d	El sistema debe mostrar el progreso del presupuesto en tiempo real en el dashboard.	• El dashboard debe mostrar una barra de progreso para el presupuesto total y por cada categoría activa. • El progreso debe actualizarse inmediatamente al registrar una nueva transacción. • Las categorías que excedan el 100% deben mostrarse en color de alerta (ej. rojo). • El sistema debe mostrar días restantes del mes y proyección de gasto si continúa el ritmo actual.	
RF06e	El sistema debe permitir ajustar presupuestos durante el mes con recálculo automático de alertas.	• El usuario debe poder modificar presupuestos existentes en cualquier momento del mes. • Al modificar un presupuesto, el sistema debe recalcular el progreso actual y ajustar alertas futuras. • El sistema debe mantener historial de cambios de presupuesto para análisis posteriores.	

RF07a	El sistema debe generar gráficos comparativos de gastos entre diferentes períodos mensuales.	• El usuario debe poder seleccionar hasta 6 meses para comparar simultáneamente. • El gráfico debe mostrar gastos totales y por categoría en formato de barras o líneas. • El usuario debe poder alternar entre vista de meses consecutivos o mismo mes de diferentes años.	
RF07b	El sistema debe calcular variaciones porcentuales y absolutas entre períodos comparados.	• El sistema debe calcular variación porcentual: ((Mes Actual - Mes Anterior) / Mes Anterior) × 100. • También debe mostrar la variación absoluta en valores monetarios. • Las variaciones positivas (aumento de gasto) deben resaltarse en rojo y las negativas (ahorro) en verde. • El cálculo debe realizarse tanto para gastos totales como por cada categoría.	
RF07c	El sistema debe generar alertas automáticas cuando una categoría tenga variaciones significativas.	• El sistema debe alertar cuando una categoría aumente más del 20% respecto al mes anterior. • También debe alertar por aumentos del 30% respecto al mismo mes del año anterior. • Las alertas deben incluir la categoría afectada, porcentaje de variación y monto de la diferencia. • El usuario debe poder configurar los umbrales de alerta personalizados.	
RF07d	El sistema debe permitir excluir transacciones atípicas del análisis comparativo.	• El usuario debe poder marcar transacciones como "atípicas" o "extraordinarias". • El sistema debe permitir filtrar automáticamente transacciones que excedan 3 veces la desviación estándar de la categoría. • Las exclusiones deben ser configurables por el usuario y aplicarse consistentemente en todos los análisis. • El sistema debe mostrar claramente cuándo se están aplicando exclusiones en los reportes.	

RF07e	El sistema debe proporcionar insights automáticos basados en las comparativas mensuales.	• El sistema debe generar al menos 1 insights por mes basados en las variaciones detectadas. • Los insights deben incluir recomendaciones específicas (ej. "Tu gasto en Transporte aumentó 25%").	
RF08a	El sistema debe permitir filtrar reportes por múltiples criterios antes de generar la descarga.	• El usuario debe poder filtrar por rango de fechas, categorías específicas, etiquetas y rangos de monto. • Los filtros deben combinarse de manera acumulativa (AND lógico) para refinar los resultados. • El sistema debe mostrar una vista previa del reporte con la cantidad de transacciones incluidas antes de descargar.	
RF08b	El sistema debe generar reportes descargables en múltiples formatos con datos completos.	• El sistema debe soportar exportación en formato PDF (visual con gráficos), CSV y Excel (.xlsx). • Los reportes deben incluir todas las transacciones filtradas con monto, fecha, descripción, categoría y etiquetas. • El archivo debe generarse en menos de 10 segundos y el nombre debe incluir el rango de fechas y fecha de generación.	
RF08c	El sistema debe ofrecer plantillas de reportes predefinidas para casos de uso comunes.	• Debe incluir plantillas: "Resumen Mensual", "Gastos por Categoría", "Análisis Anual" y "Flujo de Caja". • Cada plantilla debe tener filtros y formato preconfigurados según su propósito específico. • El usuario debe poder personalizar las plantillas y guardar configuraciones personalizadas para uso futuro.	
RF08d	El sistema debe permitir programar reportes recurrentes automáticos.	• El usuario debe poder configurar reportes automáticos mensuales, trimestrales o anuales. • Los reportes programados deben enviarse automáticamente por correo electrónico en la fecha especificada. • El usuario debe poder activar, pausar o eliminar programaciones de reportes desde su perfil.	

RF08e	El sistema debe mantener un historial de reportes generados para fácil acceso.	• El sistema debe guardar los últimos 10 reportes generados por el usuario con fecha de creación. • El usuario debe poder redescargar reportes previos sin necesidad de regenerarlos. • El historial debe mostrar el tipo de reporte, filtros aplicados y tamaño del archivo.	

5. Requisitos Técnicos

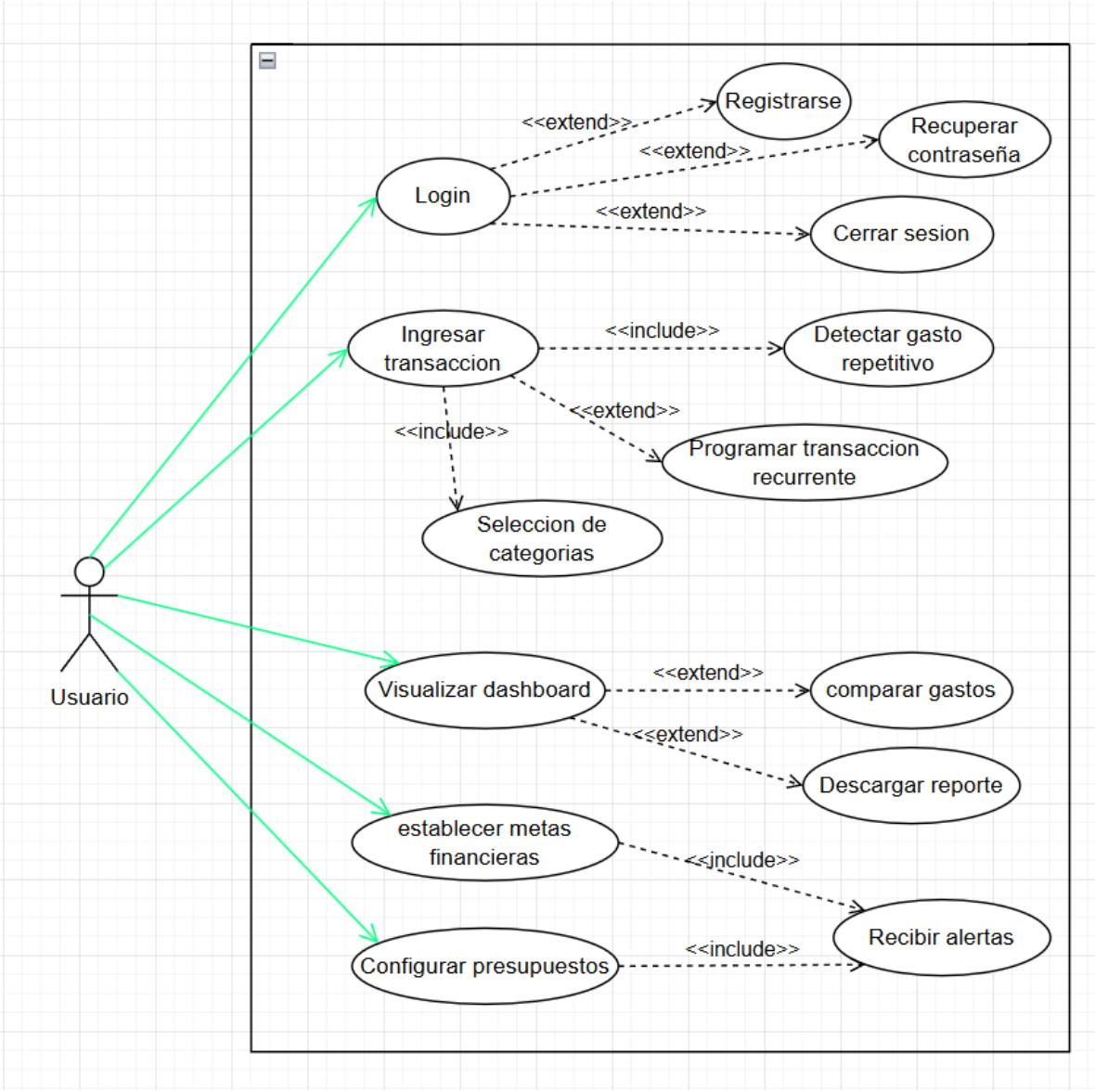
Frontend:	React + Tailwind CSS
Backend:	Node.js + Express + JWT (SimpleJWT)
Base de Datos:	PostgreSQL
Gráficos:	PostgreSQL
Infraestructura:	Separación de servicios, entorno seguro, backups automáticos.

6.Requisitos No Funcionales

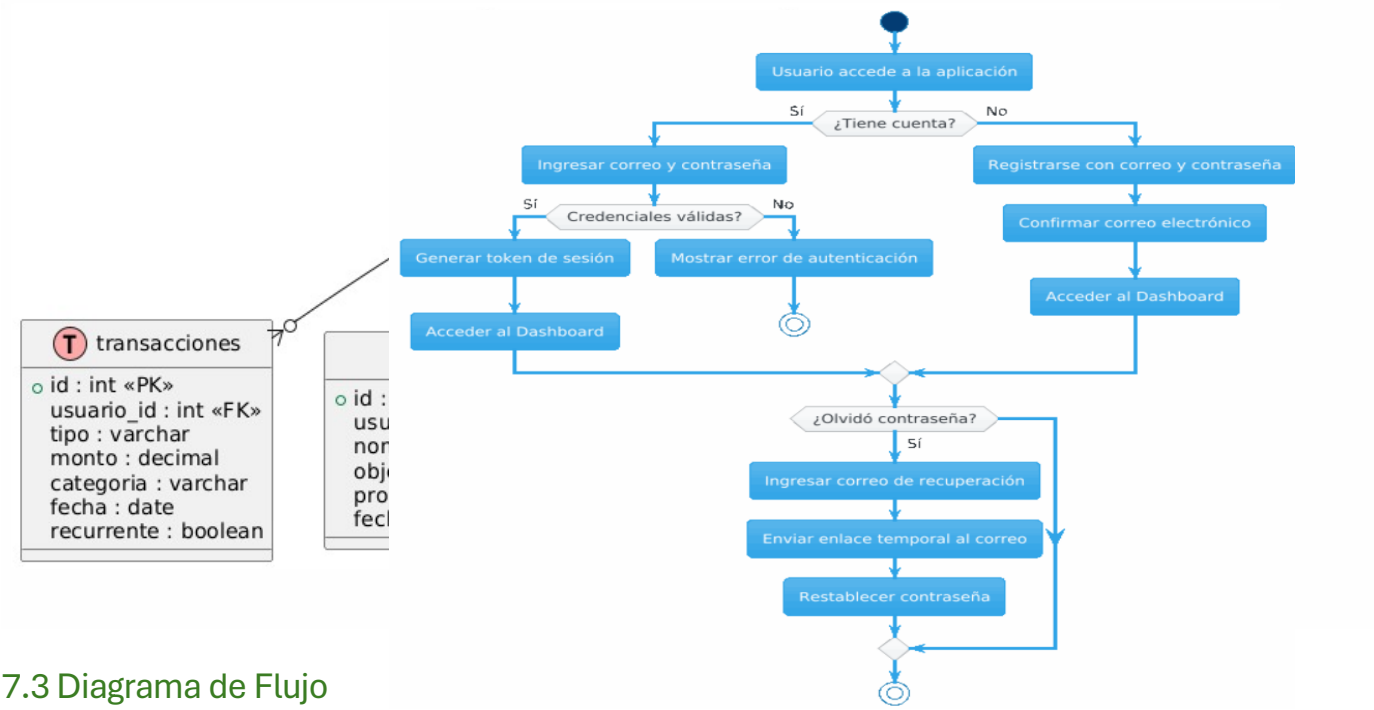
Categoría	Requisitos
Seguridad	HTTPS, encriptación de contraseñas, validación de entradas
Rendimiento	Dashboard < 5s de carga
Compatibilidad	Chrome, Firefox, Edge, Safari
Escalabilidad	Arquitectura modular, microservicios futuros
Usabilidad	UI accesible, intuitiva
Backup	Copias automáticas diarias de la base de datos
Mantenibilidad	Código comentado y modular con CI/CD

7.Diagramas

7.1Caso de Uso



7.2 Base de Datos



7.3 Diagrama de Flujo

7.4 Diagrama de Flujo 2