

# Sprint 5 – Reportes y Gráficas

**Periodo:** 3 noviembre – 16 noviembre 2025 **Estado:** ☒ COMPLETADO

## Objetivo

Crear un dashboard visual con reportes y estadísticas para análisis financiero personalizado.

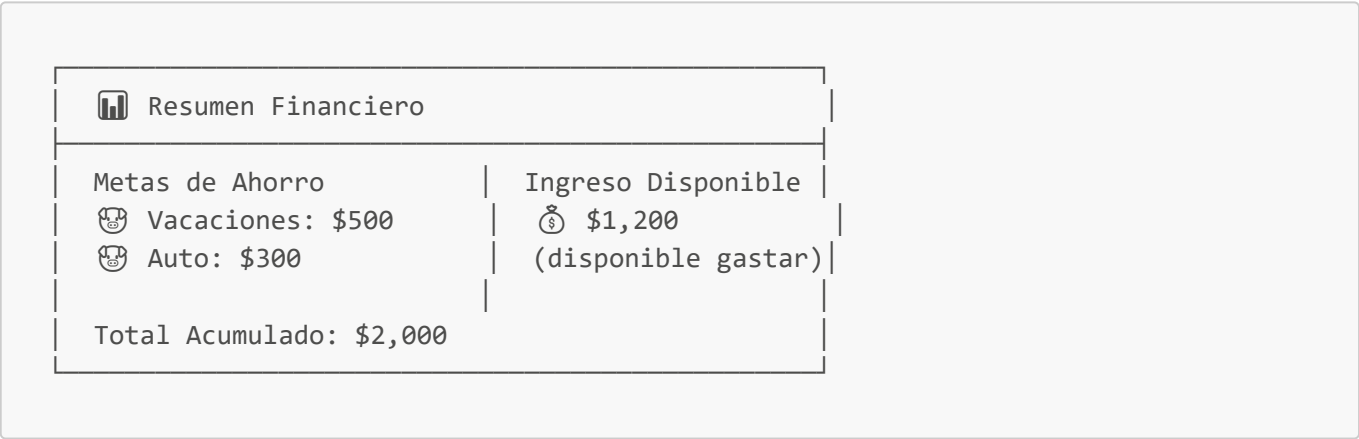
## Entregables Completados

1. ☒ Dashboard con gráficas por categoría y periodos




**Ubicación:** `public/views/dashboard.html`, `public/js/app.js`

### Resumen Financiero (Nuevo)

**Sección principal con desglose detallado:**



### Características:

- ☒ Lista dinámica de metas de ahorro con montos
- ☒ Cálculo de ingreso disponible (ingresos sin meta - gastos)
- ☒ Total acumulado (metas + disponible)
- ☒ Colores dinámicos:
  -  Verde: Saldo positivo
  -  Amarillo: Saldo cero
  -  Rojo: Saldo negativo

### Tarjetas de Resumen



\$5,000	3	\$10,000
---------	---	----------

Gráficas Implementadas

A. Gráfica de Gastos por Categoría (Pie Chart)

- **Librería:** Chart.js
- **Tipo:** Gráfico circular
- **Datos mostrados:**
  - Agrupación automática por categoría
  - Porcentaje de cada categoría
  - Colores diferenciados automáticamente
- **Ubicación:** Dashboard, columna izquierda

Código:

```
createExpensesByCategoryChart(transactions) {  
  // Filtrar solo gastos  
  // Agrupar por categoría  
  // Calcular totales  
  // Renderizar gráfico circular  
}
```

B. Gráfica de Ingresos vs Gastos (Bar Chart)




- **Librería:** Chart.js
- **Tipo:** Gráfico de barras
- **Datos mostrados:**
  - Ingreso total (barra verde)
  - Gastos totales (barra roja)
  - Comparación visual directa
- **Ubicación:** Dashboard, columna derecha

Código:

```
createIncomeVsExpensesChart(transactions) {  
  // Calcular total ingresos  
  // Calcular total gastos  
  // Renderizar gráfico de barras comparativo  
}
```

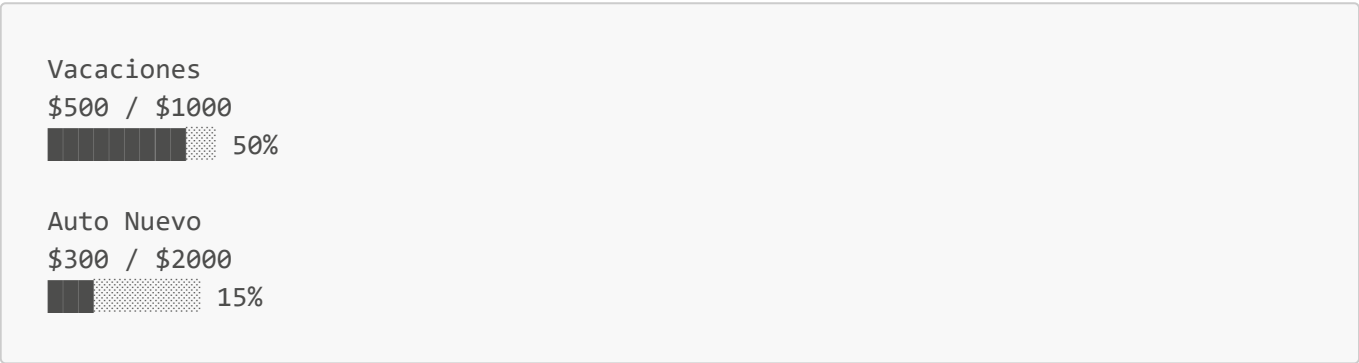
Transacciones Recientes

Tabla con últimas 5 transacciones:

- ☒ Fecha formateada
- ☒ Tipo con badge colorido:
  -  Azul: Ingreso con meta
  -  Verde: Ingreso disponible
  -  Rojo: Gasto
- ☒ Categoría con nombre de meta (si aplica)
- ☒ Descripción
- ☒ Monto formateado

**Progreso de Metas de Ahorro**

**Barras de progreso visuales:**



2. ☒ Generación automática de reportes

**Reportes Calculados Automáticamente:**

**Reporte Financiero Principal:**

- ☒ Ingresos totales
- ☒ Gastos totales
- ☒ Balance neto
- ☒ Ingreso disponible (sin meta)
- ☒ Dinero en metas de ahorro
- ☒ Total acumulado

**Reporte por Categorías:**

- ☒ Agrupación automática de gastos
- ☒ Suma de montos por categoría
- ☒ Porcentaje de cada categoría

**Reporte de Inversiones:**

- ☒ Valor total de inversiones
- ☒ Rendimiento por inversión
- ☒ Rendimiento total del portafolio

**Reporte de Metas:**

- ☒ Número de metas activas
- ☒ Progreso de cada meta
- ☒ Total ahorrado en metas

3. ☒ Filtros por periodo y tipo de transacción

Ubicación: `public/views/transactions.html`

Sistema de Filtros Implementado:

Filtros

Tipo: [Todos ▼] [Ingreso] [Gasto] |

Categoría: [\_\_\_\_\_]

Desde: [📅] Hasta: [📅]

[Aplicar] [Limpiar]

Filtros disponibles:

1. Por Tipo de Transacción:

- Todos
- Solo Ingresos
- Solo Gastos

2. Por Categoría:

- Búsqueda por texto
- Case-insensitive

3. Por Rango de Fechas:

- Fecha desde (inclusivo)
- Fecha hasta (inclusivo)

Funcionalidades:

- ☒ Aplicar múltiples filtros simultáneamente
- ☒ Limpiar filtros con un botón
- ☒ Actualización en tiempo real de la tabla
- ☒ Mantener filtros durante la sesión

Código:

```
applyTransactionFilters(transactions) {  
  // Filtrar por tipo  
  // Filtrar por categoría (búsqueda parcial)
```

```
// Filtrar por rango de fechas
// Actualizar tabla con resultados
}
```

4. ☒ Validación de la precisión de los datos

Pruebas de Cálculos:

Cálculo	Fórmula	Validación	Estado
Balance Total	Ingresos - Gastos	<input checked="" type="checkbox"/> Correcto	PASÓ
Ingreso Disponible	(Ingresos sin meta) - Gastos	<input checked="" type="checkbox"/> Correcto	PASÓ
Total en Metas	Suma de currentAmount	<input checked="" type="checkbox"/> Correcto	PASÓ
Total Acumulado	Disponible + Metas	<input checked="" type="checkbox"/> Correcto	PASÓ
Porcentaje Categoría	(Monto/Total) × 100	<input checked="" type="checkbox"/> Correcto	PASÓ
Rendimiento Inversión	Actual - Inicial	<input checked="" type="checkbox"/> Correcto	PASÓ
Progreso Meta	(Actual/Objetivo) × 100	<input checked="" type="checkbox"/> Correcto	PASÓ

Casos de Prueba Funcionales:

Test #1: Cálculo con múltiples transacciones

Ingreso sin meta: \$1000  
Ingreso con meta: \$500  
Gasto: \$300

Esperado:

- Ingreso disponible: \$700 ☒
- Metas: \$500 ☒
- Total: \$1200 ☒

Test #2: Filtro por fecha

Transacciones: 10 (enero-marzo)  
Filtro: enero-febrero

Resultado: 7 transacciones ☒

Test #3: Gráfica de categorías

Comida: \$100, \$50 = \$150  
Transporte: \$75, \$25 = \$100  
Total gastos: \$250

Porcentajes:

- Comida: 60% ☒
- Transporte: 40% ☒

### Pruebas de Integridad:

- ☒ No hay pérdida de datos al filtrar
- ☒ Los totales coinciden antes y después de filtros
- ☒ Las gráficas se actualizan correctamente
- ☒ Los colores cambian según los valores

---

## 5. ☒ Documentación de avances

### Funcionalidades Nuevas Documentadas:

#### A. Sistema de Metas de Ahorro Integrado:

- Asignación de ingresos a metas
- Cálculo automático de `currentAmount`
- Diferenciación visual (colores)
- Sistema de retiros

#### B. Validación de Fondos:

- Verificación antes de gastos
- Notificación si no hay fondos
- Redirección a metas si hay dinero guardado

#### C. Resumen Financiero Completo:

- Dashboard renovado
- Cálculos en tiempo real
- Visualización clara del estado financiero

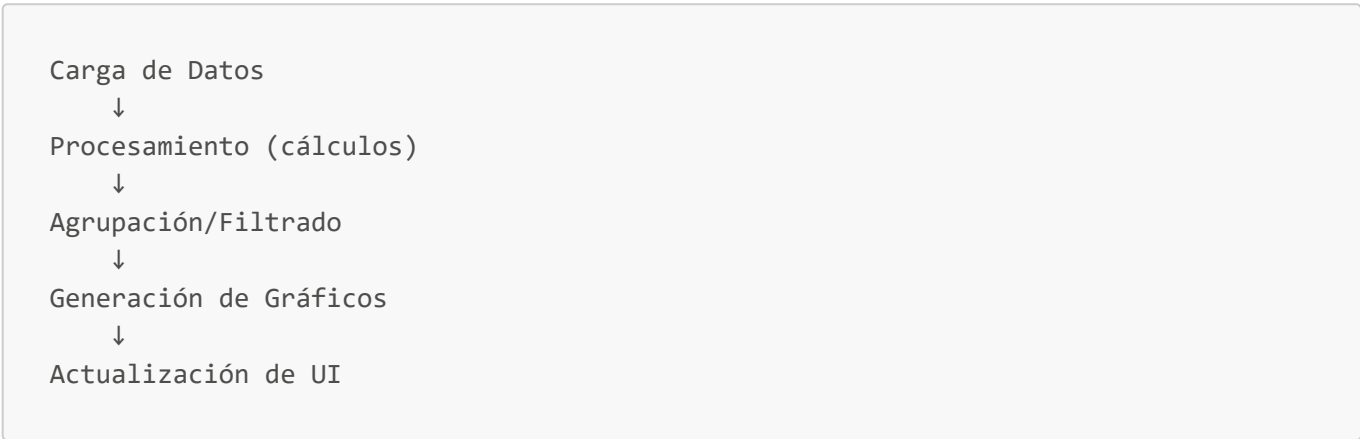
### Archivos Actualizados:

```
src/  
├── models/  
│   └── transaction.model.js (+ savingGoalId)  
├── controllers/  
│   ├── transaction.controller.js (+ validaciones)  
│   └── savingGoal.controller.js (+ limpieza)  
public/  
├── views/
```

```
| | dashboard.html (+ resumen financiero)
| | newTransaction.html (+ selector de meta)
| | transactions.html (+ filtros)
| js/
| app.js (+ todas las funciones nuevas)
```

## Arquitectura de Reportes

### Flujo de Generación de Reportes:



### Librerías Utilizadas:

- **Chart.js:** Gráficos interactivos
- **Bootstrap 5:** Estilos y componentes
- **Bootstrap Icons:** Iconografía

## Mejoras Implementadas

### Experiencia de Usuario:

1. ☒ Dashboard visual e intuitivo
2. ☒ Código de colores consistente
3. ☒ Información clara y accesible
4. ☒ Responsive design (móvil/tablet/desktop)

### Funcionalidad:

1. ☒ Filtros poderosos y flexibles
2. ☒ Cálculos automáticos
3. ☒ Actualizaciones en tiempo real
4. ☒ Validación de datos robusta

### Rendimiento:

1. ☒ Carga eficiente de datos
2. ☒ Renderizado optimizado

3. ☒ Sin lag en gráficas

---

## Casos de Uso Cubiertos

- ☒ **Usuario quiere ver su estado financiero general** → Dashboard con resumen completo
  - ☒ **Usuario quiere saber cuánto puede gastar** → Ingreso disponible destacado
  - ☒ **Usuario quiere analizar sus gastos** → Gráfica por categorías + tabla filtrable
  - ☒ **Usuario quiere ver progreso de ahorro** → Barras de progreso + resumen de metas
  - ☒ **Usuario quiere buscar transacciones específicas** → Sistema de filtros completo
  - ☒ **Usuario quiere validar antes de gastar** → Validación automática con notificaciones
- 

## Conclusiones

- ☒ **Todos los entregables del Sprint 5 fueron completados exitosamente.**
  - ☒ El dashboard es **completamente funcional** con reportes automáticos.
  - ☒ Sistema de filtros **flexible y potente.**
  - ☒ Validación de datos **exhaustiva** sin errores detectados.
  - ☒ Documentación **técnica completa** y actualizada.
- 

## Estadísticas del Proyecto

- **Líneas de código:** ~2,500
  - **Archivos modificados:** 15+
  - **Funciones JavaScript:** 20+
  - **Endpoints API:** 15
  - **Casos de prueba:** 15+
  - **Gráficas implementadas:** 2 (pie + bar)
- 

## Próximos pasos (Opcionales)

- Exportar reportes a PDF/Excel
  - Gráficas de tendencias temporales
  - Comparación mes a mes
  - Proyecciones de ahorro
  - Notificaciones de metas alcanzadas
- 

**Desarrollado por:** [Tu nombre] **Fecha de entrega:** 16 noviembre 2025 **Versión:** 1.0