# Sistema de Tarifas Fijas para Préstamos

## Implementación Completa - 13 de Noviembre 2025

## 📋 Resumen Ejecutivo

Se implementó un sistema dual de cálculo de préstamos que permite elegir entre dos métodos:
- **Método de Interés**: Sistema tradicional basado en tasa de interés anual
- **Método de Tarifa Fija**: Sistema escalonado de tarifas por monto prestado

### Características Principales

✅ Selector de tipo de cálculo en formulario de creación de préstamos
✅ Cálculo automático según el método seleccionado
✅ Validación específica para cada tipo de cálculo
✅ Visualización diferenciada en detalles de préstamo
✅ Retrocompatibilidad con préstamos existentes

---

## 🔧 Cambios Técnicos Implementados

### 1. Base de Datos (Prisma)

**Archivo**: `app/prisma/schema.prisma`

```
enum LoanCalculationType {
  INTERES
  TARIFA_FIJA
}

model Loan {
  // ... otros campos
  loanCalculationType LoanCalculationType @default(INTERES)
}
```

**Migración**: `20251113064719_add_loan_calculation_type`
- Creó enum `LoanCalculationType`
- Agregó campo `loanCalculationType` con default `INTERES`
- Garantiza retrocompatibilidad con préstamos existentes

### 2. Lógica de Cálculo

**Archivo**: `app/lib/loan-calculations.ts`

#### Funciones Principales:

**a)** **`calculateInterestBasedPayment()`**

```
// Cálculo tradicional usando fórmula de amortización
// P * [r(1+r)^n] / [(1+r)^n - 1]
```

**b)** `calculateFixedFeePayment()`

```
// Sistema escalonado de tarifas:
// - $3,000 o menos: 16 pagos de $300
// - $4,000: 16 pagos de $425
// - $5,000: 16 pagos de $600
// - $5,000+: $600 base + $120 por cada mil adicional
```

**c)** `calculateLoanDetails()`

```
// Función principal que:
// 1. Determina el método de cálculo
// 2. Calcula monto de pago periódico
// 3. Calcula monto total a pagar
// 4. Calcula fecha de finalización
// 5. Calcula tasa efectiva (para tarifa fija)
```

**d)** `validateLoanParams()`

```
// Validaciones específicas por tipo:
// - INTERES: requiere tasa de interés válida
// - TARIFA_FIJA: monto entre $1,000 y $100,000
```

## 3. API de Préstamos

**Archivo**: `app/api/loans/route.ts`

**Cambios clave:**

```
// Extracción del tipo de cálculo
const { loanCalculationType = 'INTERES', ... } = await req.json();

// Validación condicional de interés
if (loanCalculationType === 'INTERES' && !interestRate) {
  return NextResponse.json(
    { error: 'La tasa de interés es requerida' },
    { status: 400 }
  );
}

// Cálculo usando la función unificada
const loanDetails = calculateLoanDetails({
  loanCalculationType,
  principalAmount,
  numberOfPayments,
  paymentFrequency,
  annualInterestRate: rate,
  startDate
});
```

## 4. Formulario de Creación

**Archivo**: `app/components/loans/new-loan-form.tsx`

**Nuevos elementos UI:**

```
// Selector de tipo de cálculo
<Select
  value={formData.loanCalculationType}
  onValueChange={(value) => handleInputChange('loanCalculationType', value)}
>
  <SelectItem value="INTERES">Por Interés</SelectItem>
  <SelectItem value="TARIFA_FIJA">Por Tarifa Fija</SelectItem>
</Select>

// Campos condicionales
{formData.loanCalculationType === 'INTERES' && (
  <div>
    <Label>Tasa de Interés Anual (%)</Label>
    <Input type="number" step="0.01" />
  </div>
)}

{formData.loanCalculationType === 'TARIFA_FIJA' && (
  <div className="text-sm text-muted-foreground">
    Sistema de tarifas escalonadas automático
  </div>
)}
```

**Cálculo en tiempo real:**

```
const calculationType = formData.loanCalculationType;
const loanDetails = calculateLoanDetails({
  loanCalculationType: calculationType,
  principalAmount: formData.loanAmount,
  numberOfPayments: formData.numberOfPayments,
  paymentFrequency: formData.paymentFrequency,
  annualInterestRate: formData.interestRate,
  startDate: new Date(formData.startDate)
});

// Actualizar vista previa
setCalculationPreview({
  paymentAmount: loanDetails.paymentAmount,
  totalAmount: loanDetails.totalAmount,
  effectiveRate: loanDetails.effectiveRate
});
```

## 5. Vista de Detalles

**Archivo**: `app/components/loans/loan-details.tsx`

**Visualización diferenciada:**

```
// Mostrar tipo de cálculo
<div>
  <span className="text-sm text-muted-foreground">Tipo de Cálculo</span>
  <p className="text-sm font-medium">
    {loan.loanCalculationType === 'INTERES'
      ? 'Por Interés'
      : 'Por Tarifa Fija'}
  </p>
</div>

// Mostrar tasa (interés o efectiva)
<div>
  <span className="text-sm text-muted-foreground">
    {loan.loanCalculationType === 'INTERES'
      ? 'Tasa de Interés'
      : 'Tasa Efectiva'}
  </span>
  <p className="text-sm font-medium">
    {loan.interestRate.toFixed(2)}%
  </p>
</div>
```

---

## 💰 Sistema de Tarifas Fijas - Detalles

### Tabla de Tarifas (16 Pagos)

| Monto Prestado | Pago Periódico | Total a Pagar | Tarifa Total |
|---|---|---|---|
| $1,000 - $3,000 | $300 | $4,800 | $1,800 |
| $4,000 | $425 | $6,800 | $2,800 |
| $5,000 | $600 | $9,600 | $4,600 |
| $6,000 | $720 | $11,520 | $5,520 |
| $7,000 | $840 | $13,440 | $6,440 |
| $8,000 | $960 | $15,360 | $7,360 |
| $10,000 | $1,200 | $19,200 | $9,200 |

### Fórmula para montos > $5,000

```
Pago Base = $600 (por el primer $5,000)
Miles Adicionales = CEIL((Monto - 5000) / 1000)
Tarifa Adicional = Miles Adicionales × $120
Pago Total = (Pago Base + Tarifa Adicional) × Número de Pagos
Pago Periódico = Pago Total / Número de Pagos
```

### Ejemplos de Cálculo

**Ejemplo 1: Préstamo de $3,000**

```
Monto: $3,000
Pagos: 16
Pago por periodo: $300
Total a pagar: $4,800
Tarifa: $1,800
Tasa efectiva: 60%
```

**Ejemplo 2: Préstamo de $7,500**

```
Monto: $7,500
Miles adicionales: CEIL((7500-5000)/1000) = 3
Pago base: $600
Tarifa adicional: 3 × $120 = $360
Pago por periodo: $600 + $360 = $960
Total (16 pagos): $15,360
Tarifa total: $7,860
Tasa efectiva: 104.8%
```

---

## 🔍 Validaciones Implementadas

### Para Método de Interés

✅ Tasa de interés requerida
✅ Tasa debe ser ≥ 0
✅ Monto principal > 0
✅ Número de pagos > 0

### Para Método de Tarifa Fija

✅ Monto mínimo: $1,000
✅ Monto máximo: $100,000
✅ Número de pagos > 0
✅ Periodicidad válida

---

## 📊 Comparación de Métodos

### Ventajas del Método de Interés

- Más transparente y regulado
- Tasa competitiva para buenos perfiles crediticios
- Permite plazos flexibles
- Estándar internacional

### Ventajas del Método de Tarifa Fija

- Simplicidad en el cálculo
- Pagos predecibles

- No requiere evaluación de tasa individual
- Procesamiento más rápido

---

# 🧪 Ejemplos de Uso en el Sistema

## Crear Préstamo con Interés

```
POST /api/loans
{
  "clientId": "...",
  "loanCalculationType": "INTERES",
  "loanAmount": 5000,
  "numberOfPayments": 16,
  "paymentFrequency": "QUINCENAL",
  "interestRate": 24.5,
  "startDate": "2025-11-13",
  "initialPayment": 500
}
```

**Resultado esperado:**
- Pago periódico: ~$385.42
- Total a pagar: ~$6,166.72
- Interés total: ~$1,166.72

## Crear Préstamo con Tarifa Fija

```
POST /api/loans
{
  "clientId": "...",
  "loanCalculationType": "TARIFA_FIJA",
  "loanAmount": 5000,
  "numberOfPayments": 16,
  "paymentFrequency": "QUINCENAL",
  "startDate": "2025-11-13",
  "initialPayment": 500
}
```

**Resultado esperado:**
- Pago periódico: $600
- Total a pagar: $9,600
- Tarifa total: $4,600
- Tasa efectiva: 92%

---

# 🔄 Retrocompatibilidad

## Préstamos Existentes

- Todos los préstamos existentes se marcan automáticamente como `INTERES`
- No se requiere migración de datos
- Los cálculos existentes permanecen intactos

### Migración suave

```
-- La migración establece INTERES como default
ALTER TABLE "loans"
ADD COLUMN "loanCalculationType" "LoanCalculationType"
NOT NULL DEFAULT 'INTERES';
```

---

## 📝 Notas de Implementación

### Consideraciones Importantes

1. **Tasa Efectiva en Tarifa Fija**: Se calcula y muestra solo como referencia comparativa
2. **Validación de Rangos**: Tarifa fija limitada a $1,000 - $100,000
3. **Número de Pagos Fijo**: El sistema de tarifas está diseñado para 16 pagos, pero acepta otros valores
4. **Redondeo**: Todos los montos se redondean a 2 decimales

### Posibles Mejoras Futuras

- [ ] Tarifas configurables por administrador
- [ ] Tabla de tarifas personalizable
- [ ] Diferentes estructuras según periodicidad
- [ ] Descuentos por pago anticipado
- [ ] Simulador de comparación de métodos

---

## ✅ Verificación de Implementación

### Checklist Técnico

- [x] Schema de Prisma actualizado
- [x] Migración ejecutada
- [x] Funciones de cálculo implementadas
- [x] API actualizada y probada
- [x] Formulario con selector de método
- [x] Validaciones específicas por método
- [x] Vista de detalles actualizada
- [x] Cálculo en tiempo real funcionando
- [x] Build exitoso sin errores
- [x] Retrocompatibilidad verificada

### Checklist Funcional

- [x] Crear préstamo con método de interés
- [x] Crear préstamo con método de tarifa fija
- [x] Visualizar detalles de cada tipo
- [x] Validación de campos requeridos
- [x] Cálculo correcto de pagos

- [x] Fecha de finalización correcta
- [x] Tasa efectiva mostrada correctamente

---

# 🚀 Próximos Pasos

## Para el Usuario

1. Probar la creación de préstamos con ambos métodos
2. Verificar los cálculos contra casos de prueba conocidos
3. Revisar la visualización en detalles de préstamo
4. Validar el comportamiento con diferentes periodicidades

## Para Despliegue

1. Hacer pull del último commit en EasyPanel
2. Limpiar caché de build
3. Reconstruir la aplicación
4. Verificar migración de base de datos
5. Probar funcionalidad en producción

---

# 📞 Soporte

Para preguntas o ajustes al sistema de tarifas, contactar al equipo de desarrollo.

---

**Documentación generada**: 13 de Noviembre 2025
**Versión del sistema**: EscalaFin MVP v1.5
**Autor**: DeepAgent - Abacus.AI