

Correção: Campo de Meta Opcional em Aportes de Investimentos

Data: 01/11/2025 05:45 UTC







Commit: 91cd9b7

Status:  CORRIGIDO

Problema Reportado

“O botão de adicionar novo aporte não responde até eu selecionar uma meta. O campo ‘Meta’ deve ser realmente opcional, permitindo registrar aportes sem associar a uma meta existente.”


Comportamento Anterior:


-  Usuário clicava em “Novo Aporte”
-  Tentava preencher os campos obrigatórios (Nome, Valor, Categoria)
-  Deixava o campo “Meta” em branco (opcional)
-  Clicava em “Adicionar Aporte”
-  **NADA ACONTECIA** (botão não respondia)
-  Apenas funcionava se uma meta fosse selecionada

Diagnóstico

Causa Raiz

O componente `Select` do React estava com um valor inválido quando nenhuma meta era selecionada:

```
//  Problema: valor inicial vazio causa estado inválido
const [selectedGoalId, setSelectedGoalId] = useState<string>('')

//  Select tentando usar valor vazio
<Select value={selectedGoalId || 'no-goal'} onChange={(value) => setSelectedGoalId(value === 'no-goal' ? '' : value)}>
```

Por que isso causava o problema?

1. **Estado Inválido:** String vazia (`''`) não é um valor válido para o `Select`
2. **Validação Implícita:** React detectava o estado inválido e bloqueava a interação
3. **Lógica Confusa:** Conversão bidirecional entre `''` e `'no-goal'` criava inconsistências

✓ Solução Aplicada

1. Definir Valor Padrão Válido

```
// ✓ Valor inicial válido
const [selectedGoalId, setSelectedGoalId] = useState<string>('no-goal')
```

2. Simplificar o Select

```
// ✓ Select com valor sempre válido
<Select value={selectedGoalId} onValueChange={setSelectedGoalId}>
  <SelectTrigger className="bg-[#0d0d0d] border-[#2a2a2a] text-white">
    <SelectValue placeholder="Nenhuma meta" />
  </SelectTrigger>
  <SelectContent className="bg-[#1a1a1a] border-[#2a2a2a]">
    <SelectItem value="no-goal" className="text-white hover:bg-[#2a2a2a]">
      Sem meta (apenas aporte) ✓
    </SelectItem>
    {goals.map((goal) => (
      <SelectItem key={goal.id} value={goal.id} className="text-white hover:bg-[#2a2a2a]">
        {goal.name}
      </SelectItem>
    ))}
  </SelectContent>
</Select>
```

3. Converter para API

```
// ✓ Enviar null para API quando 'no-goal' estiver selecionado
body: JSON.stringify({
  name: investmentName,
  amount: parseFloat(investmentAmount),
  category: investmentCategory,
  date: investmentDate,
  goalId: selectedGoalId === 'no-goal' ? null : selectedGoalId, // ✓
})
```

4. Atualizar Lógica de Sucesso

```
// ✓ Mensagem adequada baseada na meta
if (selectedGoalId && selectedGoalId !== 'no-goal') {
  const updatedGoals = await fetch('/api/goals').then(res => res.json())
  setGoals(updatedGoals)
  toast.success('Aporte adicionado e progresso da meta atualizado! 🎉')
} else {
  toast.success('Aporte adicionado com sucesso! 💰')
}
```

5. Limpar Formulário Corretamente

```
// ✅ Resetar para valor padrão válido
setInvestmentName('')
setInvestmentAmount('')
setInvestmentCategory('')
setSelectedGoalId('no-goal') // ✅ Não mais ''
setInvestmentDate(new Date().toISOString().split('T')[0])
setIsInvestmentDialogOpen(false)
```

Resultado

Comportamento Agora:

- ✅ Usuário clica em “Novo Aporte”
- ✅ Preenche campos obrigatórios (Nome, Valor, Categoria)
- ✅ Deixa “Meta” como “Sem meta (apenas aporte)” (padrão)
- ✅ Clica em “Adicionar Aporte”
- ✅ **FUNCIONA PERFEITAMENTE!** 🎉
- ✅ Aporte é registrado sem vincular a nenhuma meta
- ✅ Se desejar, pode escolher uma meta da lista



Benefícios da Solução

1. UX Melhorada

- Campo de meta é realmente opcional
- Texto mais claro: “Sem meta (apenas aporte)”
- Usuário não precisa “adivinhar” o que fazer

2. Código Mais Limpo

- Sem conversões bidirecionais confusas
- Estado sempre válido
- Lógica mais simples e direta

3. Sem Warnings do React

- Select sempre tem valor válido
- Não há estados inválidos
- Console limpo

4. Flexibilidade

- Usuário pode adicionar aportes sem metas
- Ou pode associar a metas existentes
- Liberdade de escolha

Validação

Build Local

```
cd nextjs_space
yarn build
```

✓ **Resultado:** Build passou com sucesso

Testes Manuais

- ✓ Criar aporte sem meta
- ✓ Criar aporte com meta
- ✓ Verificar que progresso da meta é atualizado
- ✓ Verificar que aportes sem meta aparecem na lista

Arquivos Alterados

/components/investments/investments-client.tsx

1. **Linha 104:** Estado inicial alterado

```
tsx
// Antes: useState<string>>('')
// Depois: useState<string>('no-goal')
```

2. **Linha 233:** Lógica de envio ajustada

```
tsx
// Antes: goalId: selectedGoalId || null
// Depois: goalId: selectedGoalId === 'no-goal' ? null : selectedGoalId
```

3. **Linha 242:** Validação de meta aprimorada

```
tsx
// Antes: if (selectedGoalId)
// Depois: if (selectedGoalId && selectedGoalId !== 'no-goal')
```

4. **Linha 254:** Reset do formulário corrigido

```
tsx
// Antes: setSelectedGoalId('')
// Depois: setSelectedGoalId('no-goal')
```

5. **Linha 821:** Select simplificado

```
tsx
// Antes: value={selectedGoalId || 'no-goal'} onChange={...conversão complexa...}
// Depois: value={selectedGoalId} onChange={setSelectedGoalId}
```

6. **Linha 826:** Texto melhorado

```
tsx
// Antes: "Nenhuma meta"
// Depois: "Sem meta (apenas aporte)"
```

Lições Aprendidas

Para Componentes Select

1. Sempre use valores válidos

- ❌ Nunca use string vazia `''` como valor
- ✅ Use um valor padrão como `'none'`, `'no-selection'`, `'no-goal'`

2. Estado Controlado Simples

- ❌ Evite conversões bidirecionais complexas
- ✅ Use o valor diretamente do estado

3. Validação Clara

- ❌ Não confie apenas no valor do Select
- ✅ Valide explicitamente antes de enviar para API

4. Feedback ao Usuário

- ❌ Não use textos genéricos como “Nenhuma”
 - ✅ Seja específico: “Sem meta (apenas aporte)”
-

Impacto

- **Experiência do Usuário:** 🎯 Significativamente melhorada
 - **Flexibilidade:** ✅ Campo de meta agora é verdadeiramente opcional
 - **Manutenibilidade:** ✅ Código mais simples e claro
 - **Bugs:** ✅ Zero warnings ou erros no console
-

Referências

- **Commit:** `91cd9b7`
 - **Branch:** `main`
 - **Deploy:** Vercel (automático após push)
 - **URL:** <https://orcamento-planejado.abacusai.app>
-

✅ Checklist de Validação

- [x] Build local aprovado
 - [x] Commit realizado
 - [x] Push para GitHub
 - [x] Deploy Vercel em andamento
 - [x] Documentação criada
 - [x] Relatório principal atualizado
-

Última atualização: 01/11/2025 05:45 UTC

Status:  **RESOLVIDO E DOCUMENTADO**

Próximo deploy: Automático via Vercel