PayMordomo - Documentação Completa

📋 Visão Geral

PayMordomo é uma aplicação web de gestão financeira cristã, desenvolvida em **HTML, CSS e JavaScript puros** (sem frameworks). O projeto oferece funcionalidades completas para controle de transações, metas financeiras, dízimos e muito mais.

Características Principais

✅ **Autenticação Segura** - Login com celular e senha via Supabase

✅ **Dashboard Inteligente** - Visão geral financeira com versículos e dicas

✅ **CRUD Completo** - Gerenciamento de transações, metas e dízimos

✅ **Armazenamento Local** - Dados sincronizados com localStorage

✅ **Responsivo** - Funciona perfeitamente em mobile, tablet e desktop

✅ **Logging Seguro** - Sistema de logs sem expor dados sensíveis

✅ **WhatsApp Integration** - Cadastro direto via WhatsApp

🏗️ Estrutura do Projeto

Plain Text

```
paymordomo-pure-js/
├── client/
│   ├── index.html              # Arquivo HTML principal
│   ├── css/
│   │   ├── style.css           # Estilos globais
│   │   ├── dashboard.css       # Estilos do dashboard
│   │   ├── transactions.css    # Estilos de transações
│   │   └── responsive.css      # Estilos responsivos
│   ├── js/
│   │   ├── app.js              # Aplicação principal
│   │   ├── auth.js             # Sistema de autenticação
│   │   ├── storage.js          # Gerenciador de armazenamento
│   │   ├── supabase-client.js  # Cliente Supabase
│   │   ├── ui.js               # Funções de UI
│   │   ├── utils/
│   │   │   ├── logger.js       # Sistema de logging
│   │   │   ├── formatter.js    # Formatadores de dados
│   │   │   └── validators.js   # Validadores
│   │   └── pages/
```

```
│   │       ├── dashboard.js    # Página Dashboard
│   │       ├── transactions.js # Página Transações
│   │       ├── goals.js        # Página Metas
│   │       ├── tithe.js        # Página Dízimos
│   │       ├── tips.js         # Página Dicas
│   │       └── reports.js      # Página Relatórios
│   └── public/                 # Arquivos estáticos
└── DOCUMENTACAO.md             # Este arquivo
```

🔐 Sistema de Autenticação

Login

A autenticação é feita via **Supabase** usando celular e senha:

1. **Formato do Celular**: `(11) 98765-4321` ou `11987654321`
2. **Conversão Interna**: O celular é convertido para email `11987654321@paymordomo.local`
3. **Integração Supabase**: Usa as credenciais fornecidas para autenticação

Credenciais Supabase (Seguras - Não expostas no Console)

JavaScript

```
SUPABASE_URL: 'https://fetimotrijqyswrfoyzz.supabase.co'
SUPABASE_ANON_KEY: 'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...'
```

Cadastro via WhatsApp

Usuários novos são redirecionados para WhatsApp:

Plain Text

```
https://wa.me/553287073537?
text=Olá%20MordomoPay%21%20Gostaria%20de%20criar%20uma%20conta
```

O número `553287073537` é o contato de suporte que faz o cadastro.

Logout

Ao fazer logout:

- Sessão é encerrada no Supabase
- Dados locais são limpos

- Usuário é redirecionado para login

📊 Páginas e Funcionalidades

1. Dashboard

Página inicial com resumo financeiro:

- **Estatísticas Rápidas**
  - Total de Receitas
  - Total de Despesas
  - Saldo Atual
  - Saúde Financeira (%)
  - Economia do Mês
  - Metas Ativas
- **Metas Financeiras**
  - Exibição de metas com barra de progresso
  - Percentual de conclusão
  - Prazo de cada meta
- **Conquistas**
  - Badges desbloqueadas
  - Conquistas em progresso
- **Versículo do Dia**
  - Versículos bíblicos relacionados a finanças
  - Navegação entre versículos
  - Atualização automática
- **Dica de Saúde Financeira**
  - Dicas práticas de gestão financeira
  - Navegação entre dicas
  - Atualização automática

2. Transações 💳

Gerenciamento completo de transações (CRUD):

Criar Transação

JavaScript

```
{
  description: "Salário",
  amount: 5200,
  type: "entrada",        // entrada ou saida
  category: "Renda",
  payee: "Empresa XYZ",
  date: "2025-01-05"
}
```

Funcionalidades

- ✅ **Incluir** - Novo botão abre modal
- ✅ **Editar** - Clique no ícone ✏️
- ✅ **Deletar** - Clique no ícone 🗑️
- ✅ **Filtrar** - Por tipo (entrada/saída) e categoria
- ✅ **Exportar** - Baixar como CSV

Categorias Disponíveis

- Alimentação
- Moradia
- Transporte
- Saúde
- Educação
- Lazer
- Espiritual
- Renda
- Renda Extra

3. Metas 🎯

Gerenciamento de metas financeiras:

Criar Meta

JavaScript

```
{
  name: "Fundo de Emergência",
  target: 5000,           // Valor alvo
  current: 3200,          // Valor atual
  category: "Segurança",
  deadline: "2025-12-31"
}
```

Funcionalidades

- ✅ **Criar** - Nova meta com valor alvo
- ✅ **Editar** - Atualizar valores e prazos
- ✅ **Deletar** - Remover meta
- ✅ **Progresso** - Barra visual de progresso
- ✅ **Prazo** - Acompanhamento de prazos

4. Dízimos 🙏

Registro e acompanhamento de dízimos:

Registrar Dízimo

JavaScript

```
{
  amount: 520,
  date: "2025-01-07",
  description: "Dízimo mensal"
}
```

Funcionalidades

- ✅ **Registrar** - Novo dízimo
- ✅ **Deletar** - Remover registro
- ✅ **Total** - Soma de todos os dízimos
- ✅ **Mês Atual** - Dízimos do mês
- ✅ **Meses Consecutivos** - Contagem de meses com dízimo

5. Dicas 💡

Dicas de saúde financeira:

- Registre todas as suas transações
- Estabeleça metas realistas
- Separe uma porcentagem para dízimo
- Crie um fundo de emergência
- Revise seus gastos mensalmente
- Invista em educação financeira
- Evite dívidas desnecessárias
- Pratique a gratidão

6. Relatórios 📈

Análise financeira completa:

- Total de Receitas
- Total de Despesas
- Saldo Final
- Taxa de Economia (%)

🛠️ Sistema de Logging

O sistema de logging é **seguro** e não expõe dados sensíveis:

Uso

JavaScript

```
// Informações
logger.info('Mensagem', { dados });

// Sucesso
logger.success('Operação realizada');

// Aviso
logger.warn('Atenção', { dados });

// Erro
logger.error('Erro ocorreu', { error: 'mensagem' });

// Debug
```

```
logger.debug('Informação de debug');

// Função
logger.function('NomeFuncao', 'ação realizada');

// API
logger.api('/endpoint', 'GET', 200);

// Autenticação (sem expor dados)
logger.auth('Login', true, { userId: '123' });

// Transação
logger.transaction('Create', 'id-123', 5200, 'entrada');

// Validação
logger.validation('email', true);
```

Dados Sensíveis Redatados

Campos automaticamente redatados:

- `password` → `***REDACTED***`
- `token` → `***REDACTED***`
- `apiKey` → `***REDACTED***`
- `phone` → `***REDACTED***`
- `email` → `***REDACTED***`
- `cpf` → `***REDACTED***`

📦 Armazenamento Local

Dados são salvos em `localStorage` com prefixo `paymordomo_`:

JavaScript

```
// Salvar dados
storage.set('transactions', transactionsArray);

// Obter dados
const transactions = storage.get('transactions');

// Remover dados
storage.remove('transactions');
```

```
// Limpar tudo
storage.clear();

// Dados específicos
storage.setUser(user);
storage.getUser();
storage.setTransactions(transactions);
storage.getTransactions();
storage.setGoals(goals);
storage.getGoals();
storage.setTithes(tithes);
storage.getTithes();
```

🎨 Formatadores de Dados

Funções para formatar dados para exibição:

JavaScript

```
// Moeda
Formatter.currency(5200)                    // R$ 5.200,00

// Data
Formatter.date('2025-01-05')               // 05/01/2025

// Data e Hora
Formatter.dateTime('2025-01-05T14:30:00')  // 05/01/2025 14:30:00

// Percentual
Formatter.percentage(72)                    // 72%

// Número
Formatter.number(1234.56)                   // 1.234,56

// Telefone
Formatter.phone('11987654321')              // (11) 98765-4321

// CPF
Formatter.cpf('12345678901')                // 123.456.789-01

// Texto
Formatter.capitalize('joão')                // João
Formatter.titleCase('joão silva')           // João Silva
Formatter.truncate('texto longo', 10)       // texto lo...

// Status
```

```
Formatter.status('ativo')                    // 🟢 Ativo

// Tempo relativo
Formatter.timeAgo('2025-01-05')              // 5 dias atrás

// Iniciais de nome
Formatter.initials('João Silva')             // JS
```

✅ Validadores

Funções para validar dados de entrada:

JavaScript

```
// Email
Validators.email('user@example.com')         // true/false

// Telefone
Validators.phone('11987654321')              // true/false

// CPF
Validators.cpf('12345678901')                // true/false

// CNPJ
Validators.cnpj('12345678901234')            // true/false

// Senha
Validators.password('Senha123')              // true/false

// URL
Validators.url('https://example.com' )       // true/false

// Número
Validators.number(123)                       // true/false
Validators.positiveNumber(123)               // true/false

// Texto
Validators.notEmpty('texto')                 // true/false
Validators.minLength('texto', 3)             // true/false
Validators.maxLength('texto', 10)            // true/false

// Comparação
Validators.equal(a, b)                       // true/false
Validators.between(50, 0, 100)               // true/false

// Transação
```

```
Validators.transactionType('entrada')       // true/false
Validators.transactionCategory('Renda')     // true/false
```

🎮 Funções de UI

Funções para interagir com a interface:

JavaScript

```
// Notificações
UI.showToast('Mensagem', 'success')         // success/error/info

// Loader
UI.showLoader()
UI.hideLoader()

// Modais
UI.showModal('modal-id')
UI.hideModal('modal-id')

// Formulários
UI.clearForm('form-id')
UI.validateForm('form-id')

// Elementos
UI.setValue('element-id', 'valor')
UI.getValue('element-id')
UI.show('element-id')
UI.hide('element-id')
UI.enable('element-id')
UI.disable('element-id')

// Classes
UI.addClass('element-id', 'class-name')
UI.removeClass('element-id', 'class-name')
UI.toggleClass('element-id', 'class-name')

// Navegação
UI.changePage('dashboard')
UI.changeContent('transactions')
UI.setActiveNav('transactions')

// Tabelas e Listas
UI.renderTable('table-id', data, columns)
UI.renderList('list-id', items, template)
UI.renderGrid('grid-id', items, template)
```

```
// Utilitários
UI.copyToClipboard('texto')
UI.scrollTo('element-id')
UI.focus('element-id')
UI.print('element-id')
```

🔄 Atualização Automática de Versículos e Dicas

Sistema Atual

Versículos e dicas são armazenados em arrays e navegáveis:

JavaScript

```
// Versículos
const verses = [
    { text: 'Versículo 1', ref: 'Referência 1' },
    { text: 'Versículo 2', ref: 'Referência 2' },
    // ...
];

// Dicas
const tips = [
    'Dica 1',
    'Dica 2',
    // ...
];
```

Como Adicionar Novos Versículos/Dicas

1. **Editar arquivo** `dashboard.js` :

JavaScript

```
this.verses = [
    // Adicione novos versículos aqui
    { text: 'Novo versículo', ref: 'Referência' }
];

this.tips = [
    // Adicione novas dicas aqui
    'Nova dica'
];
```

1. **Ou via API Supabase** (futuro):

```javascript
async loadVerses() {
    const { data } = await supabaseClient.select('verses');
    this.verses = data;
}
```

Atualização em Tempo Real

Para implementar atualização em tempo real via Supabase:

```javascript
// No arquivo dashboard.js
async init() {
    // ... código existente ...

    // Escuta mudanças em tempo real
    supabaseClient.onRealtimeChange('verses', (payload) => {
        if (payload.eventType === 'INSERT' || payload.eventType ===
'UPDATE') {
            this.verses = payload.new;
            this.renderVerse();
        }
    });
}
```

🚀 Como Usar

1. Acessar a Aplicação

Abra no navegador:

```
https://3000-i6y5tq3x0asl6q7bcg6ad-3d3774ff.us1.manus.computer
```

2. Fazer Login

- **Celular**: `11987654321` (ou qualquer número)

- **Senha**: `Teste123456` (ou qualquer senha válida)

Nota: A autenticação usa Supabase. Para criar um usuário real, use o WhatsApp.

3. Criar Conta via WhatsApp

Clique em "Criar Conta via WhatsApp" e será redirecionado para:

```
Plain Text

https://wa.me/553287073537?
text=Olá%20MordomoPay%21%20Gostaria%20de%20criar%20uma%20conta
```

4. Usar as Funcionalidades

- **Dashboard**: Visão geral e versículos/dicas
- **Transações**: Adicionar, editar, deletar transações
- **Metas**: Gerenciar metas financeiras
- **Dízimos**: Registrar dízimos
- **Dicas**: Ler dicas de saúde financeira
- **Relatórios**: Análise financeira

🔒 Segurança

Proteção de Dados

1. **Senhas**: Nunca são exibidas em logs
2. **Tokens**: Redatados automaticamente
3. **Dados Sensíveis**: CPF, email, telefone são redatados
4. **localStorage**: Dados locais são prefixados e isolados
5. **Supabase**: Usa autenticação segura com JWT

Boas Práticas

- ✅ Não exponha chaves de API no console
- ✅ Use HTTPS em produção
- ✅ Valide todos os dados de entrada
- ✅ Implemente rate limiting

- ✅ Monitore logs de segurança

🐛 Troubleshooting

Problema: Login não funciona

Solução:

1. Verifique se o Supabase está acessível
2. Confirme as credenciais
3. Verifique o console para erros
4. Limpe o localStorage: `storage.clear( )`

Problema: Dados não são salvos

Solução:

1. Verifique se localStorage está habilitado
2. Verifique o espaço disponível
3. Verifique o console para erros
4. Tente em modo anônimo

Problema: Interface não carrega

Solução:

1. Limpe o cache do navegador
2. Recarregue a página (Ctrl+Shift+R)
3. Verifique se todos os arquivos CSS/JS estão carregando
4. Verifique o console para erros

📱 Responsividade

A aplicação é totalmente responsiva:

- **Desktop** (>1024px): Layout completo com sidebar
- **Tablet** (768px-1024px): Layout adaptado
- **Mobile** (<768px): Menu toggle, layout otimizado

🎯 Próximas Melhorias

- [ ] Integração com gráficos (Chart.js)
- [ ] Backup automático de dados
- [ ] Sincronização em tempo real
- [ ] Notificações push
- [ ] Exportação em PDF
- [ ] Modo offline
- [ ] Temas personalizáveis
- [ ] Suporte a múltiplas contas

📞 Suporte

Para suporte, entre em contato via WhatsApp:

```
https://wa.me/553287073537
```

📄 Licença

Este projeto é de uso privado e proprietário.

✨ Desenvolvido com ❤️

PayMordomo - Gestão Cristã Inteligente

Desenvolvido em HTML, CSS e JavaScript puros, sem frameworks.

Última atualização: 10 de Janeiro de 2026