Trust-me Backend API Documentation

Visão Geral

Este é o backend do sistema Trust-me migrado para Laravel/MySQL, seguindo a estrutura do projeto de referência. O sistema oferece APIs para autenticação, gestão de planos, assinaturas, pagamentos e conteúdo.

Configuração

Requisitos

- PHP 8.1+
- MySQL 8.0+
- Composer
- Laravel 10.x

Instalação

```
# Clone o projeto
git clone <repository>
cd trust-me-laravel

# Instale as dependências
composer install

# Configure o ambiente
cp .env.example .env
php artisan key:generate

# Configure o banco de dados no .env
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=trustme
DB_USERNAME=trustme
DB_PASSWORD=trustme123

# Execute as migrations e seeders
php artisan migrate
php artisan db:seed

# Inicie o servidor
php artisan serve --host=0.0.0.0 --port=8000
```

Autenticação

O sistema utiliza Laravel Sanctum para autenticação via tokens.

Endpoints de Autenticação

POST /api/auth/login

Realiza login do usuário.

Request:

```
{
    "email": "admin@trustme.com",
    "password": "admin123"
}
```

Response:

```
{
    "user": {
        "id": 1,
        "name": "Administrador",
        "email": "admin@trustme.com",
        "role": "admin"
    },
    "token": "1|token_string_here",
    "token_type": "Bearer"
}
```

POST /api/auth/register

Registra novo usuário.

Request:

```
{
    "name": "João Silva",
    "email": "joao@email.com",
    "password": "senha123",
    "password_confirmation": "senha123"
}
```

POST /api/auth/logout

Realiza logout (requer autenticação).

GET /api/auth/me

Retorna dados do usuário autenticado.

POST /api/auth/forgot-password

Envia email de recuperação de senha.

POST /api/auth/reset-password

Redefine senha com token.

Planos

GET /api/plans

Lista todos os planos ativos.

Response:


```
{
    "success": true,
    "data": [
        {
            "id": 1,
            "name": "Básico",
            "description": "Plano ideal para pequenos negócios",
            "monthly_price": "29.90",
            "semiannual_price": "99.90",
            "annual_price": "199.90",
            "seals_limit": 1,
            "contracts_limit": 1,
            "features": [
                "1 selo digital",
                "1 contrato",
                "Suporte por email"
            ]
        }
    ]
}
```


GET /api/plans/{id}

Retorna detalhes de um plano específico.

Assinaturas

GET /api/user/subscriptions

Lista assinaturas do usuário autenticado.

POST /api/subscriptions (Admin)

Cria nova assinatura.

Request:


```
{
    "user_id": 1,
    "plan_id": 1,
    "billing_cycle": "monthly",
    "payment_method": "mercado_pago",
    "payment_id": "payment_123"
}
```

Pagamentos

GET /api/payment/methods

Lista métodos de pagamento disponíveis.

POST /api/payment/create-preference

Cria preferência de pagamento no Mercado Pago.

Request:

```
{
    "plan_id": 1,
    "billing_cycle": "monthly"
}
```

Response:

```
{
    "success": true,
    "data": {
        "id": "PREF_123456",
        "init_point": "https://www.mercadopago.com.br/checkout/v1/redirect?
pref_id=PREF_123456",
        "items": [...]
    }
}
```

POST /api/payment/process

Processa pagamento aprovado.

POST /api/payment/webhook

Webhook para notificações do Mercado Pago.

Conteúdo

GET /api/faqs

Lista FAQs ativas.

GET /api/testimonials

Lista depoimentos ativos.

POST /api/contacts

Envia mensagem de contato.

Request:

```
{
    "name": "João Silva",
    "email": "joao@email.com",
    "phone": "(11) 99999-9999",
    "subject": "Dúvida sobre planos",
    "message": "Gostaria de saber mais sobre..."
}
```

Administração

Todas as rotas administrativas requerem autenticação e role “admin”.

Dashboard

- GET /api/admin/dashboard - Estatísticas gerais
- GET /api/admin/users - Lista usuários
- GET /api/admin/subscriptions - Lista assinaturas
- GET /api/admin/contacts - Lista contatos
- GET /api/admin/reports - Relatórios

Gestão de Usuários

- GET /api/users - Lista usuários
- POST /api/users - Cria usuário
- GET /api/users/{id} - Detalhes do usuário
- PUT /api/users/{id} - Atualiza usuário
- DELETE /api/users/{id} - Remove usuário

Gestão de Planos

- POST /api/plans - Cria plano
- PUT /api/plans/{id} - Atualiza plano
- DELETE /api/plans/{id} - Desativa plano

Gestão de FAQs

- POST /api/faqs - Cria FAQ
- PUT /api/faqs/{id} - Atualiza FAQ
- DELETE /api/faqs/{id} - Remove FAQ

Gestão de Depoimentos

- POST /api/testimonials - Cria depoimento
- PUT /api/testimonials/{id} - Atualiza depoimento
- DELETE /api/testimonials/{id} - Remove depoimento

Gestão de Contatos

- GET /api/contacts - Lista contatos
- POST /api/contacts/{id}/respond - Responde contato
- PUT /api/contacts/{id}/status - Atualiza status

Configurações do Site

- GET /api/site-settings - Lista configurações

- POST /api/site-settings - Cria configuração
- PUT /api/site-settings/{key} - Atualiza configuração
- POST /api/site-settings/bulk-update - Atualização em lote

Códigos de Status

- 200: Sucesso
- 201: Criado com sucesso
- 400: Erro na requisição
- 401: Não autenticado
- 403: Acesso negado
- 404: Não encontrado
- 422: Erro de validação
- 500: Erro interno do servidor

Headers Necessários

Para rotas autenticadas:

```
Authorization: Bearer {token}
Content-Type: application/json
Accept: application/json
```

Dados de Teste

Usuários

- Admin: admin@trustme.com / admin123
- User: user@trustme.com / user123

Planos

1. Básico: R$ 29,90/mês - 1 selo + 1 contrato
2. Intermediário: R$ 49,90/mês - 3 selos + 3 contratos
3. Plus: R$ 69,90/mês - Ilimitado

Integração com Frontend NextJS

Para conectar com o frontend NextJS:

1. Configure a URL base da API: `http://localhost:8000/api`
2. Use o token retornado no login para autenticação
3. Implemente interceptors para adicionar o token automaticamente
4. Trate os códigos de erro adequadamente

Exemplo de uso (JavaScript):

```javascript
// Login
const response = await fetch('http://localhost:8000/api/auth/login', {
    method: 'POST',
    headers: {
        'Content-Type': 'application/json',
    },
    body: JSON.stringify({
        email: 'admin@trustme.com',
        password: 'admin123'
    })
});

const data = await response.json();
localStorage.setItem('token', data.token);

// Requisições autenticadas
const authResponse = await fetch('http://localhost:8000/api/user/profile', {
    headers: {
        'Authorization': `Bearer ${localStorage.getItem('token')}`,
        'Content-Type': 'application/json',
    }
});
```

Estrutura do Projeto

```
trust-me-laravel/
├── app/
│   ├── Http/
│   │   ├── Controllers/
│   │   │   ├── AuthController.php
│   │   │   ├── PlanController.php
│   │   │   ├── SubscriptionController.php
│   │   │   ├── PaymentController.php
│   │   │   └── ...
│   │   └── Middleware/
│   │       ├── CheckAdmin.php
│   │       └── CheckEmpresa.php
│   └── Models/
│       ├── User.php
│       ├── Plan.php
│       ├── Subscription.php
│       └── ...
├── database/
│   ├── migrations/
│   └── seeders/
├── routes/
│   └── api.php
└── ...
```

Próximos Passos

1. Implementar integração real com Mercado Pago

2. Adicionar sistema de notificações por email
3. Implementar logs de auditoria
4. Adicionar testes automatizados
5. Configurar CI/CD
6. Implementar cache Redis
7. Adicionar documentação Swagger/OpenAPI