Documentação da API: BTC-PERP Absolute Trader

API REST Completa

Versão: 1.0
Base URL: `http://localhost:5000/api/trading`
Formato: JSON

📋 Índice

🌐 Visão Geral

A API REST do BTC-PERP Absolute Trader fornece acesso programático completo ao sistema de trading algorítmico. Permite controlar o sistema, monitorar performance, executar ordens e acessar dados históricos.

Características

- **RESTful**: Segue princípios REST
- **JSON**: Formato de dados padrão
- **CORS**: Suporte a cross-origin requests
- **Real-time**: Dados atualizados em tempo real
- **Stateless**: Sem estado entre requisições

Base URL

```
http://localhost:5000/api/trading
```

Headers Padrão

```
Content-Type: application/json
Accept: application/json
```

🔐 Autenticação

Nota: A versão atual não requer autenticação. Em produção, implementar autenticação adequada.

Headers de Autenticação (Futuro)

```
Authorization: Bearer <token>
X-API-Key: <api-key>
```

🖥️ Endpoints de Sistema

Health Check

Verifica se a API está funcionando.

```
GET /health
```

Resposta:

```
{
  "status": "healthy",
  "timestamp": "2024-01-15T10:30:00Z",
  "version": "1.0.0"
}
```

Status do Sistema

Retorna o status atual do sistema de trading.

```
GET /status
```

Resposta:

```
{
  "status": "running",
  "balance": 98500.50,
  "total_equity": 99200.75,
  "open_orders": 2,
  "total_trades": 45,
  "positions": 1,
  "timestamp": "2024-01-15T10:30:00Z"
}
```

Status Possíveis: - `not_initialized`: Sistema não inicializado - `stopped`: Sistema parado - `running`: Sistema operando

Inicializar Sistema

Inicializa o sistema de trading com parâmetros específicos.

```
POST /initialize
Content-Type: application/json

{
  "initial_balance": 100000,
  "commission_rate": 0.001
}
```

Parâmetros: - `initial_balance` (number): Capital inicial em USD - `commission_rate` (number): Taxa de comissão (0.001 = 0.1%)

Resposta:

```
{
  "message": "Sistema inicializado com sucesso",
  "initial_balance": 100000,
  "commission_rate": 0.001
}
```

Iniciar Trading

Inicia o bot de trading.

```
POST /start
```

Resposta:

```
{
  "message": "Bot de trading iniciado"
}
```

Parar Trading

Para o bot de trading.

```
POST /stop
```

Resposta:

```
{
  "message": "Bot de trading parado"
}
```

📈 Endpoints de Trading

Colocar Ordem

Coloca uma nova ordem no sistema.

```
POST /place_order
Content-Type: application/json

{
  "symbol": "BTC-USDT",
  "side": "buy",
  "type": "market",
  "quantity": 0.1,
  "price": 50000.00
}
```

Parâmetros: - `symbol` (string): Par de trading (ex: "BTC-USDT") - `side` (string): "buy" ou "sell" - `type` (string): "market", "limit", "stop", "stop_limit" - `quantity` (number): Quantidade a negociar - `price` (number, opcional): Preço para ordens limit - `stop_price` (number, opcional): Preço de stop

Resposta:

```
{
  "message": "Ordem colocada com sucesso",
  "order_id": "ord_1234567890"
}
```

Cancelar Ordem

Cancela uma ordem específica.

```
DELETE /cancel_order/{order_id}
```

Parâmetros: - `order_id` (string): ID da ordem a cancelar

Resposta:

```
{
  "message": "Ordem cancelada com sucesso"
}
```

Listar Ordens

Retorna ordens abertas e histórico.

```
GET /orders
```

Resposta:


```
{
  "open_orders": [
    {
      "id": "ord_1234567890",
      "symbol": "BTC-USDT",
      "side": "buy",
      "type": "limit",
      "quantity": 0.1,
      "price": 49500.00,
      "status": "open",
      "timestamp": "2024-01-15T10:30:00Z"
    }
  ],
  "order_history": [
    {
      "id": "ord_0987654321",
      "symbol": "BTC-USDT",
      "side": "sell",
      "type": "market",
      "quantity": 0.05,
      "price": 50100.00,
      "status": "filled",
      "timestamp": "2024-01-15T09:15:00Z",
      "fill_timestamp": "2024-01-15T09:15:01Z"
    }
  ]
}
```


Listar Posições

Retorna posições atuais.

```
GET /positions
```

Resposta:


```
{
  "positions": [
    {
      "symbol": "BTC-USDT",
      "quantity": 0.05,
      "average_price": 49800.00,
      "unrealized_pnl": 150.00,
      "realized_pnl": 75.50,
      "side": "long"
    }
  ]
}
```

Performance

Retorna métricas de performance.

```
GET /performance
```

Resposta:

```
{
  "total_return": 0.125,
  "total_return_pct": 12.5,
  "annualized_return": 0.234,
  "volatility": 0.186,
  "sharpe_ratio": 1.67,
  "sortino_ratio": 2.31,
  "calmar_ratio": 2.04,
  "max_drawdown": -0.068,
  "max_drawdown_pct": -6.8,
  "win_rate": 0.613,
  "profit_factor": 1.58,
  "total_trades": 156,
  "winning_trades": 96,
  "losing_trades": 60,
  "average_win": 890.50,
  "average_loss": -625.30,
  "largest_win": 3200.00,
  "largest_loss": -1800.00
}
```

📊 Endpoints de Dados

Atualizar Dados de Mercado

Atualiza preços de mercado (para simulação).

```
 POST /market_data
Content-Type: application/json

{
  "symbol": "BTC-USDT",
  "price": 50000.00,
  "volume": 1500.50,
  "timestamp": "2024-01-15T10:30:00Z"
}
```

Parâmetros: - `symbol` (string): Par de trading - `price` (number): Preço atual - `volume` (number, opcional): Volume - `timestamp` (string, opcional): Timestamp ISO 8601

Resposta:

```
{
  "message": "Preço atualizado com sucesso"
}
```

Dados Históricos

Retorna dados históricos de preços.

```
GET /historical_data?symbol=BTC-USDT&timeframe=5m&limit=100
```

Parâmetros: - `symbol` (string): Par de trading - `timeframe` (string): Timeframe (1m, 5m, 15m, 1h, 4h, 1d) - `limit` (number): Número de registros (máximo 1000) - `start_time` (string, opcional): Data de início (ISO 8601) - `end_time` (string, opcional): Data de fim (ISO 8601)

Resposta:

```
{
  "symbol": "BTC-USDT",
  "timeframe": "5m",
  "data": [
    {
      "timestamp": "2024-01-15T10:30:00Z",
      "open": 49950.00,
      "high": 50100.00,
      "low": 49900.00,
      "close": 50000.00,
      "volume": 125.50
    }
  ]
}
```

⚖️ Endpoints de Risco

Métricas de Risco

Retorna métricas de risco atuais.

```
GET /risk_metrics
```

Resposta:

```
{
  "var_95": -0.021,
  "var_99": -0.034,
  "cvar_95": -0.028,
  "cvar_99": -0.042,
  "current_drawdown": -0.025,
  "max_drawdown": -0.068,
  "portfolio_risk": 0.15,
  "position_concentration": 0.08,
  "daily_pnl": 250.75,
  "risk_score": 3.2,
  "risk_level": "medium"
}
```

Limites de Risco

Retorna e permite atualizar limites de risco.

```
GET /risk_limits
```

Resposta:

```
{
  "max_position_size": 0.1,
  "max_portfolio_risk": 0.2,
  "max_daily_loss": 0.05,
  "max_drawdown": 0.15,
  "var_limit": 0.03,
  "stop_loss_pct": 0.02,
  "take_profit_pct": 0.04
}
```

Atualizar Limites de Risco

```
PUT /risk_limits
Content-Type: application/json

{
  "max_position_size": 0.08,
  "max_daily_loss": 0.03
}
```

Resposta:

```
{
  "message": "Limites de risco atualizados com sucesso"
}
```

Alertas de Risco

Retorna alertas de risco ativos.

```
GET /risk_alerts
```

Resposta:

```
{
  "alerts": [
    {
      "id": "alert_001",
      "type": "drawdown_warning",
      "level": "warning",
      "message": "Drawdown atual (4.2%) próximo do limite (5%)",
      "timestamp": "2024-01-15T10:30:00Z",
      "acknowledged": false
    }
  ]
}
```

📋 Modelos de Dados

Order

```
{
  "id": "string",
  "symbol": "string",
  "side": "buy|sell",
  "type": "market|limit|stop|stop_limit",
  "quantity": "number",
  "price": "number|null",
  "stop_price": "number|null",
  "status": "pending|open|filled|cancelled|rejected",
  "timestamp": "string (ISO 8601)",
  "fill_timestamp": "string (ISO 8601)|null",
  "commission": "number"
}
```

Position

```
{
  "symbol": "string",
  "quantity": "number",
  "average_price": "number",
  "unrealized_pnl": "number",
  "realized_pnl": "number",
  "side": "long|short|flat"
}
```

Trade

```
{
  "id": "string",
  "symbol": "string",
  "side": "buy|sell",
  "quantity": "number",
  "price": "number",
  "commission": "number",
  "timestamp": "string (ISO 8601)",
  "order_id": "string"
}
```

Market Data

```
{
  "symbol": "string",
  "timestamp": "string (ISO 8601)",
  "open": "number",
  "high": "number",
  "low": "number",
  "close": "number",
  "volume": "number"
}
```

Performance Metrics

```
{
  "total_return": "number",
  "total_return_pct": "number",
  "annualized_return": "number",
  "volatility": "number",
  "sharpe_ratio": "number",
  "sortino_ratio": "number",
  "calmar_ratio": "number",
  "max_drawdown": "number",
  "max_drawdown_pct": "number",
  "win_rate": "number",
  "profit_factor": "number",
  "total_trades": "number",
  "winning_trades": "number",
  "losing_trades": "number"
}
```

❌ Códigos de Erro

HTTP Status Codes

- **200 OK**: Requisição bem-sucedida
- **201 Created**: Recurso criado com sucesso
- **400 Bad Request**: Dados inválidos na requisição
- **401 Unauthorized**: Autenticação necessária
- **403 Forbidden**: Acesso negado
- **404 Not Found**: Recurso não encontrado
- **409 Conflict**: Conflito com estado atual
- **422 Unprocessable Entity**: Dados válidos mas não processáveis
- **500 Internal Server Error**: Erro interno do servidor

Formato de Erro

```
{
  "error": "string",
  "message": "string",
  "code": "string",
  "details": "object|null",
  "timestamp": "string (ISO 8601)"
}
```

Códigos de Erro Específicos

Código	Descrição
`SYSTEM_NOT_INITIALIZED`	Sistema não foi inicializado
`INVALID_ORDER_PARAMS`	Parâmetros de ordem inválidos
`INSUFFICIENT_BALANCE`	Saldo insuficiente
`ORDER_NOT_FOUND`	Ordem não encontrada
`POSITION_NOT_FOUND`	Posição não encontrada
`RISK_LIMIT_EXCEEDED`	Limite de risco excedido
`MARKET_CLOSED`	Mercado fechado
`INVALID_SYMBOL`	Símbolo inválido
`RATE_LIMIT_EXCEEDED`	Limite de taxa excedido

Exemplos de Erro

```
{
  "error": "Insufficient Balance",
  "message": "Saldo insuficiente para executar a ordem",
  "code": "INSUFFICIENT_BALANCE",
  "details": {
    "required": 5000.00,
    "available": 3500.50
  },
  "timestamp": "2024-01-15T10:30:00Z"
}
```

💡 Exemplos de Uso

Python

```python
import requests
import json

# Base URL
BASE_URL = "http://localhost:5000/api/trading"

# Inicializar sistema
response = requests.post(f"{BASE_URL}/initialize", json={
    "initial_balance": 100000,
    "commission_rate": 0.001
})
print(response.json())

# Verificar status
response = requests.get(f"{BASE_URL}/status")
status = response.json()
print(f"Status: {status['status']}")

# Iniciar trading
response = requests.post(f"{BASE_URL}/start")
print(response.json())

# Colocar ordem
order_data = {
    "symbol": "BTC-USDT",
    "side": "buy",
    "type": "market",
    "quantity": 0.1
}
response = requests.post(f"{BASE_URL}/place_order", json=order_data)
print(response.json())

# Verificar posições
response = requests.get(f"{BASE_URL}/positions")
positions = response.json()
print(f"Posições: {len(positions['positions'])}")

# Obter performance
response = requests.get(f"{BASE_URL}/performance")
performance = response.json()
print(f"Sharpe Ratio: {performance['sharpe_ratio']}")
```

JavaScript

```javascript
const BASE_URL = "http://localhost:5000/api/trading";

// Função auxiliar para requisições
async function apiRequest(endpoint, options = {}) {
    const response = await fetch(`${BASE_URL}${endpoint}`, {
        headers: {
            'Content-Type': 'application/json',
            ...options.headers
        },
        ...options
    });

    if (!response.ok) {
        throw new Error(`HTTP ${response.status}: ${response.statusText}`);
    }

    return response.json();
}

// Inicializar sistema
async function initializeSystem() {
    try {
        const result = await apiRequest('/initialize', {
            method: 'POST',
            body: JSON.stringify({
                initial_balance: 100000,
                commission_rate: 0.001
            })
        });
        console.log('Sistema inicializado:', result);
    } catch (error) {
        console.error('Erro:', error);
    }
}

// Verificar status
async function getStatus() {
    try {
        const status = await apiRequest('/status');
        console.log('Status do sistema:', status);
        return status;
    } catch (error) {
        console.error('Erro:', error);
    }
}

// Colocar ordem
async function placeOrder(orderData) {
    try {
        const result = await apiRequest('/place_order', {
            method: 'POST',
            body: JSON.stringify(orderData)
        });
        console.log('Ordem colocada:', result);
        return result;
    } catch (error) {
        console.error('Erro:', error);
    }
```

```
}

// Exemplo de uso
(async () => {
    await initializeSystem();
    await getStatus();

    await placeOrder({
        symbol: "BTC-USDT",
        side: "buy",
        type: "market",
        quantity: 0.1
    });
})();
```

cURL

```
# Inicializar sistema
curl -X POST http://localhost:5000/api/trading/initialize \
  -H "Content-Type: application/json" \
  -d '{"initial_balance": 100000, "commission_rate": 0.001}'

# Verificar status
curl -X GET http://localhost:5000/api/trading/status

# Iniciar trading
curl -X POST http://localhost:5000/api/trading/start

# Colocar ordem
curl -X POST http://localhost:5000/api/trading/place_order \
  -H "Content-Type: application/json" \
  -d '{
    "symbol": "BTC-USDT",
    "side": "buy",
    "type": "market",
    "quantity": 0.1
  }'

# Verificar posições
curl -X GET http://localhost:5000/api/trading/positions

# Obter performance
curl -X GET http://localhost:5000/api/trading/performance
```

🔄 WebSocket (Futuro)

Conexão WebSocket

```
const ws = new WebSocket('ws://localhost:5000/ws/trading');

ws.onopen = function(event) {
    console.log('Conectado ao WebSocket');
};

ws.onmessage = function(event) {
    const data = JSON.parse(event.data);
    console.log('Dados recebidos:', data);
};

ws.onclose = function(event) {
    console.log('Conexão fechada');
};
```

Tipos de Mensagem

- **market_data**: Dados de mercado em tempo real
- **order_update**: Atualizações de ordens
- **position_update**: Atualizações de posições
- **risk_alert**: Alertas de risco
- **system_status**: Mudanças de status do sistema

📚 Recursos Adicionais

Postman Collection

Uma collection do Postman está disponível em `docs/postman_collection.json` com todos os endpoints configurados.

OpenAPI Specification

A especificação OpenAPI 3.0 está disponível em `docs/openapi.yaml` para geração automática de clientes.

Rate Limiting

- **Limite**: 100 requisições por minuto por IP
- **Headers**: `X-RateLimit-Limit`, `X-RateLimit-Remaining`
- **Resposta**: HTTP 429 quando excedido

Versionamento

- **Versão Atual**: `v1`
- **Header**: `API-Version: v1`
- **URL**: `/api/v1/trading` (futuro)

Esta documentação é mantida atualizada com cada release do sistema. Para a versão mais recente, consulte o repositório oficial.

Documentação da API - BTC-PERP Absolute Trader v1.0
Última atualização: Janeiro 2025