CAI × ERC-8004 Framework – 完整 API 文档

版本: v1.0.0
最后更新: 2025-10-20
Base URL: `https://api.cai-framework.eth` (生产环境) | `http://localhost:3001` (开发环境)

📋 目录

概述

CAI Framework API 提供一套 RESTful 接口，用于：

- 签发和验证可验证凭证 (Verifiable Credentials)
- 管理 AHIN 哈希链和 Merkle 证明
- 生成签名审计报告
- 查询链上交易状态
- 管理去中心化身份 (DID)

技术栈

- **协议**: HTTPS (TLS 1.3)
- **格式**: JSON (application/json)
- **编码**: UTF-8
- **认证**: Bearer Token (JWT)

核心特性

- ✅ RESTful 设计原则

- ✅ 幂等性保证
- ✅ 请求签名验证
- ✅ 速率限制保护
- ✅ 完整错误信息
- ✅ Webhook 事件通知

认证与安全

API 密钥

所有请求需在 HTTP Header 中包含 API 密钥：

http

```
Authorization: Bearer YOUR_API_KEY
Content-Type: application/json
```

获取 API 密钥

bash

```
# 开发环境
curl -X POST http://localhost:3001/api/auth/register \
  -H "Content-Type: application/json" \
  -d '{
    "address": "0x1234567890123456789012345678901234567890",
    "signature": "0xsigned_message"
  }'
```

响应:

json

```
{
  "apiKey": "cai_sk_live_1a2b3c4d5e6f...",
  "expiresAt": "2026-10-20T00:00:00Z",
  "permissions": ["vc:issue", "ahin:read", "audit:generate"]
}
```

请求签名

对于敏感操作，需要额外的请求签名：

javascript

```
const signature = ethers.utils.keccak256(
  ethers.utils.solidityPack(
    ['string', 'string', 'uint256'],
    [method, path, timestamp]
  )
);

// 添加到 Header
headers['X-Signature'] = signature;
headers['X-Timestamp'] = timestamp.toString();
```

通用响应格式

成功响应

json

```
{
  "success": true,
  "data": { /* 响应数据 */ },
  "meta": {
    "timestamp": "2025-10-20T14:32:15.123Z",
    "requestId": "req_1a2b3c4d",
    "version": "1.0.0"
  }
}
```

分页响应

json

```
{
  "success": true,
  "data": [ /* 数据数组 */ ],
  "pagination": {
    "page": 1,
    "pageSize": 20,
    "totalPages": 5,
    "totalItems": 97,
    "hasNext": true,
    "hasPrev": false
  },
  "meta": { /* 元数据 */ }
}
```

错误处理

错误响应格式

json

```
{
  "success": false,
  "error": {
    "code": "VALIDATION_ERROR",
    "message": "Invalid address format",
    "details": [
      {
        "field": "agent",
        "message": "Must be a valid Ethereum address",
        "value": "invalid_address"
      }
    ],
    "timestamp": "2025-10-20T14:32:15.123Z",
    "requestId": "req_1a2b3c4d",
    "docs": "https://docs.cai-framework.eth/errors/validation-error"
  }
}
```


错误代码列表

代码	HTTP Status	说明	解决方案
VALIDATION_ERROR	400	请求参数验证失败	检查请求参数格式
UNAUTHORIZED	401	缺少或无效的 API 密钥	提供有效的 Bearer Token
FORBIDDEN	403	权限不足	确认 API 密钥有对应权限
NOT_FOUND	404	资源不存在	检查资源 ID 是否正确
RATE_LIMIT_EXCEEDED	429	超过速率限制	减少请求频率或升级配额
INTERNAL_ERROR	500	服务器内部错误	联系技术支持
CONTRACT_ERROR	503	智能合约调用失败	检查网络状态和 Gas 余额
VC_EXPIRED	400	凭证已过期	使用有效期内的凭证
SIGNATURE_INVALID	400	签名验证失败	检查签名算法和私钥
DID_NOT_FOUND	404	DID 不存在	先注册 DID
INSUFFICIENT_BALANCE	402	余额不足	充值账户

速率限制

限制规则

层级	请求数	时间窗口	适用端点
免费	100	15 分钟	所有公开端点
基础	1,000	15 分钟	所有端点
专业	10,000	15 分钟	所有端点 + 优先支持
企业	无限制	-	所有端点 + SLA 保证

响应 Headers

http

```
X-RateLimit-Limit: 1000
X-RateLimit-Remaining: 847
X-RateLimit-Reset: 1697723471
Retry-After: 900
```

超限响应

json

```
{
  "success": false,
  "error": {
    "code": "RATE_LIMIT_EXCEEDED",
    "message": "Too many requests",
    "retryAfter": 900,
    "limit": 1000,
    "remaining": 0
  }
}
```

API 端点

6.1 可验证凭证 (VC)

6.1.1 创建 Mandate VC

端点: `POST /api/v1/vc/mandate`

描述: 签发授权凭证，定义 AI Agent 的交易权限

请求 Headers:

http

```
Authorization: Bearer YOUR_API_KEY
Content-Type: application/json
```

请求 Body:

json

```
{
  "subject": "0x1234567890123456789012345678901234567890",
  "agent": "0xabcdefabcdefabcdefabcdefabcdefabcdefabcd",
  "budget": "100000000000000000000",
  "expiry": 86400,
  "whitelist": ["merchant1.eth", "merchant2.eth"],
  "metadata": {
    "purpose": "AI training data purchase",
    "category": "research"
  }
}
```

字段说明:

字段	类型	必填	说明
`subject`	`string`	✅	用户 DID 地址 (`0x...`)
`agent`	`string`	✅	Agent DID 地址 (`0x...`)
`budget`	`string`	✅	预算金额 (wei，字符串格式)
`expiry`	`number`	✅	有效期 (秒)
`whitelist`	`array`	❌	白名单商户列表
`metadata`	`object`	❌	额外元数据

响应: `201 Created`

json

```
{
  "success": true,
  "data": {
    "vc": {
      "@context": ["https://www.w3.org/2018/credentials/v1"],
      "id": "urn:uuid:3f7a8b2c-9d1e-4f5a-8c3b-2a1d9e8f7c6b",
      "type": ["VerifiableCredential", "MandateVC"],
      "issuer": "did:ethr:0x9876543210987654321098765432109876543210",
      "issuanceDate": "2025-10-20T14:32:15.123Z",
      "expirationDate": "2025-10-21T14:32:15.123Z",
      "credentialSubject": {
        "id": "0x1234567890123456789012345678901234567890",
        "agent": "0xabcdefabcdefabcdefabcdefabcdefabcdefabcd",
        "budget": "100000000000000000000",
        "expiry": 86400,
        "whitelist": ["merchant1.eth", "merchant2.eth"]
      },
      "proof": {
        "type": "EcdsaSecp256k1Signature2019",
        "created": "2025-10-20T14:32:15.123Z",
        "verificationMethod": "did:ethr:0x9876...#key-1",
        "proofPurpose": "assertionMethod",
        "signatureValue": "0xabcdef1234567890abcdef1234567890abcdef1234567890abcdef123456789
      }
    },
    "vcHash": "0x7f3e8b9a2c1d4e5f6a7b8c9d0e1f2a3b4c5d6e7f8a9b0c1d2e3f4a5b6c7d8e9f",
    "signature": "0x1a2b3c4d5e6f7a8b9c0d1e2f3a4b5c6d7e8f9a0b1c2d3e4f5a6b7c8d9e0f1a2b"
  },
  "meta": {
    "timestamp": "2025-10-20T14:32:15.123Z",
    "requestId": "req_mandate_001"
  }
}
```


查询参数:

- `onchain=true`: 同时在链上签发 VC（需要额外 Gas 费用）

示例:

bash

```
curl -X POST https://api.cai-framework.eth/api/v1/vc/mandate \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -H "Content-Type: application/json" \
  -d '{
    "subject": "0x1234567890123456789012345678901234567890",
    "agent": "0xabcdefabcdefabcdefabcdefabcdefabcdefabcd",
    "budget": "100000000000000000000",
    "expiry": 86400
  }'
```

6.1.2 创建 Cart VC

端点: `POST /api/v1/vc/cart`

描述: 生成购物车凭证，记录交易商品明细

请求 Body:

json

```
{
  "subject": "0xabcdefabcdefabcdefabcdefabcdefabcdefabcd",
  "cartHash": "0x9a8b7c6d5e4f3a2b1c0d9e8f7a6b5c4d3e2f1a0b9c8d7e6f5a4b3c2d1e0f9a8b",
  "items": [
    {
      "id": "item_001",
      "name": "AI Training Dataset",
      "description": "10GB labeled images",
      "price": "50000000000000000000",
      "quantity": 1,
      "metadata": {
        "format": "PNG",
        "resolution": "1920x1080"
      }
    },
    {
      "id": "item_002",
      "name": "GPU Compute Hours",
      "description": "NVIDIA A100 rental",
      "price": "45000000000000000000",
      "quantity": 10,
      "unit": "hour"
    }
  ],
  "totalAmount": "95000000000000000000",
  "merchant": "0xmerchant1234567890123456789012345678901234"
}
```


响应: `201 Created`

json


```
{
  "success": true,
  "data": {
    "vc": { /* W3C VC 结构 */ },
    "vcHash": "0x...",
    "cartHash": "0x9a8b7c6d5e4f3a2b1c0d9e8f7a6b5c4d3e2f1a0b...",
    "signature": "0x..."
  }
}
```

6.1.3 验证 VC

端点: `GET /api/v1/vc/verify/:vcHash`

描述: 验证凭证有效性（链上 + 链下）

路径参数:

- `vcHash`: 凭证哈希 (0x...)

响应: `200 OK`

json


```
{
  "success": true,
  "data": {
    "vcHash": "0x7f3e8b9a2c1d4e5f6a7b8c9d0e1f2a3b4c5d6e7f...",
    "isValid": true,
    "checks": {
      "signatureValid": true,
      "notExpired": true,
      "notRevoked": true,
      "issuerTrusted": true,
      "subjectActive": true
    },
    "credential": {
      "type": "MandateVC",
      "issuer": "did:ethr:0x9876...",
      "subject": "0x1234...",
      "issuedAt": "2025-10-20T14:32:15.123Z",
      "expiresAt": "2025-10-21T14:32:15.123Z"
    }
  }
}
```


错误响应: `404 Not Found`

json

```
{
  "success": false,
  "error": {
    "code": "VC_NOT_FOUND",
    "message": "Credential not found or revoked"
  }
}
```

6.1.4 撤销 VC

端点: `POST /api/v1/vc/revoke`

描述: 撤销已签发的凭证

请求 Body:

json

```
{
  "vcHash": "0x7f3e8b9a2c1d4e5f6a7b8c9d0e1f2a3b4c5d6e7f...",
  "reason": "User requested revocation",
  "signature": "0xissuer_signature"
}
```

响应: `200 OK`

json

```
{
  "success": true,
  "data": {
    "vcHash": "0x7f3e...",
    "revoked": true,
    "revokedAt": "2025-10-20T15:00:00.000Z",
    "txHash": "0xrevoke_tx_hash"
  }
}
```

6.2 AHIN 服务

6.2.1 添加交易到队列

端点: `POST /api/v1/ahin/transaction`

描述: 将交易数据加入 AHIN 批处理队列

请求 Body:

```
json

{
  "transactionId": "0x1a2b3c4d5e6f7a8b9c0d1e2f3a4b5c6d7e8f9a0b...",
  "mandateVC": "0x7f3e8b9a2c1d4e5f...",
  "cartHash": "0x9a8b7c6d5e4f3a2b...",
  "receiptHash": "0xfedcba0987654321...",
  "metadata": {
    "agent": "0xabcd...",
    "merchant": "0x1234...",
    "amount": "950000000000000000"
  }
}
```

响应: `202 Accepted`

```
json

{
  "success": true,
  "data": {
    "ahinTxId": "0xahin_tx_3f7a8b2c...",
    "status": "queued",
    "queuePosition": 23,
    "estimatedAnchorTime": "2025-10-20T14:37:00.000Z",
    "batchNumber": 15
  }
}
```

6.2.2 获取 Merkle 证明

端点: `GET /api/v1/ahin/proof/:txId`

描述: 获取交易的 Merkle 树证明路径

路径参数:

- `txId`: AHIN 交易 ID

响应: `200 OK`

json


```
{
  "success": true,
  "data": {
    "txId": "0xahin_tx_3f7a8b2c...",
    "proof": [
      "0x1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef",
      "0xfedcba0987654321fedcba0987654321fedcba0987654321fedcba0987654321",
      "0xabcdef1234567890abcdef1234567890abcdef1234567890abcdef1234567890"
    ],
    "blockNumber": 15,
    "merkleRoot": "0x9a8b7c6d5e4f3a2b1c0d9e8f7a6b5c4d3e2f1a0b...",
    "anchorTxHash": "0xetherscan_tx_hash",
    "verified": true
  }
}
```


验证示例:

javascript

```
// 使用返回的 proof 验证交易
const isValid = await ahinContract.verifyTransaction(
  blockNumber,
  txHash,
  proof
);
```

6.2.3 查询 AHIN 统计

端点: `GET /api/v1/ahin/stats`

描述: 获取 AHIN 系统统计信息

响应: `200 OK`

json

```
{
  "success": true,
  "data": {
    "onChainBlocks": 12,
    "offChainBlocks": 15,
    "pendingTransactions": 23,
    "totalTransactionsAnchored": 1847,
    "lastAnchorTimestamp": "2025-10-20T14:30:00.000Z",
    "nextAnchorEstimate": "2025-10-20T14:35:00.000Z",
    "averageAnchorTime": "4.5 minutes",
    "systemStatus": "healthy"
  }
}
```

6.2.4 手动触发锚定

端点: `POST /api/v1/ahin/anchor`

描述: 手动触发批量锚定到区块链（需要管理员权限）

请求 Headers:

http

```
Authorization: Bearer ADMIN_API_KEY
X-Signature: 0xadmin_signature
```

响应: `200 OK`

json

```
{
  "success": true,
  "data": {
    "blockNumber": 16,
    "merkleRoot": "0x9f8b7a6c5d4e3f2a1b0c9d8e7f6a5b4c3d2e1f0a...",
    "transactionCount": 45,
    "txHash": "0xsepolia_tx_hash",
    "gasUsed": "123456",
    "metadataURI": "ipfs://Qm..."
  }
}
```

6.3 审计服务

6.3.1 生成审计报告

端点: `POST /api/v1/audit/bundle`

描述: 生成签名的审计报告包

请求 Body:

json


```
{
  "transactionId": "0x1a2b3c4d5e6f7a8b9c0d1e2f3a4b5c6d...",
  "mandateVC": "0x7f3e8b9a2c1d4e5f6a7b8c9d0e1f2a3b...",
  "cartHash": "0x9a8b7c6d5e4f3a2b1c0d9e8f7a6b5c4d...",
  "receiptHash": "0xfedcba0987654321fedcba0987654321...",
  "includeProof": true
}
```


响应: `200 OK`

json

```
{
  "success": true,
  "data": {
    "bundleId": "0xbundle_unique_id",
    "timestamp": 1697723471000,
    "transactionId": "0x1a2b3c4d...",
    "verificationChain": [
      {
        "step": 1,
        "name": "Mandate VC",
        "hash": "0x7f3e...",
        "status": "verified",
        "signature": "0xmandate_sig"
      },
      {
        "step": 2,
        "name": "Cart VC",
        "hash": "0x9a8b...",
        "status": "verified",
        "budgetCheck": "95 DAI ≤ 100 DAI"
      },
      {
        "step": 3,
        "name": "Payment",
        "hash": "0x1a2b...",
        "status": "completed",
        "amount": "95000000000000000000"
      },
      {
        "step": 4,
        "name": "Receipt",
        "hash": "0xfedc...",
        "status": "verified",
        "provider": "PayPal"
      },
      {
        "step": 5,
        "name": "AHIN Anchor",
        "blockNumber": 15,
        "merkleRoot": "0x9f8b...",
        "status": "anchored"
      }
    ],
    "merkleProof": [ /* Merkle path */ ],
```

```
    "signature": "0xbundle_signature",
    "metadata": {
      "version": "1.0.0",
      "network": "sepolia",
      "generatedBy": "CAI Backend v1.0.0"
    }
  }
}
```

下载选项:

bash

```
# 下载 JSON 格式
curl -X POST https://api.cai-framework.eth/api/v1/audit/bundle \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{"transactionId": "0x..."}' \
  -o audit-bundle.json

# 下载 PDF 格式
curl -X POST https://api.cai-framework.eth/api/v1/audit/bundle?format=pdf \
  -H "Authorization: Bearer YOUR_API_KEY" \
  -d '{"transactionId": "0x..."}' \
  -o audit-report.pdf
```

6.3.2 验证审计报告

端点: `POST /api/v1/audit/verify`

描述: 验证审计报告的完整性和签名

请求 Body:

json

```
{
  "bundleId": "0xbundle_unique_id",
  "signature": "0xbundle_signature",
  "merkleRoot": "0x9f8b7a6c5d4e3f2a..."
}
```

响应: `200 OK`

json


```
{
  "success": true,
  "data": {
    "valid": true,
    "checks": {
      "signatureValid": true,
      "merkleRootMatch": true,
      "timestampValid": true,
      "chainIntact": true
    },
    "issuer": "did:ethr:0x9876...",
    "issuedAt": "2025-10-20T14:32:15.123Z"
  }
}
```

6.4 DID 管理

6.4.1 注册 DID

端点: `POST /api/v1/did/register`

请求 Body:

json


```
{
  "address": "0x1234567890123456789012345678901234567890",
  "didDocument": "ipfs://QmDidDocument123456",
  "signature": "0xsigned_registration"
}
```


响应: `201 Created`

json

```
{
  "success": true,
  "data": {
    "did": "did:ethr:0x1234567890123456789012345678901234567890",
    "didDocument": "ipfs://QmDidDocument123456",
    "status": "active",
    "createdAt": "2025-10-20T14:32:15.123Z",
    "txHash": "0xregistration_tx"
  }
}
```

6.4.2 查询 DID 信息

端点: `GET /api/v1/did/:address`

响应: `200 OK`

json


```
{
  "success": true,
  "data": {
    "did": "did:ethr:0x1234...",
    "owner": "0x1234...",
    "didDocument": "ipfs://Qm...",
    "status": "active",
    "createdAt": "2025-10-01T00:00:00.000Z",
    "updatedAt": "2025-10-20T14:32:15.123Z",
    "credentialCount": 3,
    "transactionCount": 12,
    "reputation": 95
  }
}
```

6.5 交易查询

6.5.1 查询交易详情

端点: `GET /api/v1/transactions/:txId`

响应: `200 OK`

json


```
{
  "success": true,
  "data": {
    "transactionId": "0x1a2b...",
    "agent": "0xabcd...",
    "merchant": "0x1234...",
    "user": "0x5678...",
    "amount": "950000000000000000",
    "cartHash": "0x9a8b...",
    "receiptHash": "0xfedc...",
    "status": "completed",
    "createdAt": "2025-10-20T14:30:00.000Z",
    "completedAt": "2025-10-20T14:32:00.000Z",
    "verificationStatus": {
      "mandateVerified": true,
      "cartVerified": true,
      "paymentVerified": true,
      "receiptVerified": true,
      "ahinAnchored": true
    }
  }
}
```

6.5.2 查询交易列表

端点: `GET /api/v1/transactions`

查询参数:

- `agent`: 过滤 Agent 地址
- `status`: 过滤状态 (pending/completed/disputed)
- `page`: 页码 (默认 1)
- `pageSize`: 每页数量 (默认 20，最大 100)
- `sort`: 排序字段 (createdAt/amount)
- `order`: 排序方向 (asc/desc)

示例:

bash

```
GET /api/v1/transactions?agent=0xabcd...&status=completed&page=1&pageSize=20
```

响应: `200 OK`

json

```
{
  "success": true,
  "data": [
    { /* 交易对象 */ },
    { /* 交易对象 */ }
  ],
  "pagination": {
    "page": 1,
    "pageSize": 20,
    "totalPages": 5,
    "totalItems": 97
  }
}
```

6.6 统计数据

6.6.1 系统统计

端点: `GET /api/v1/stats/system`

响应: `200 OK`

json

```
{
  "success": true,
  "data": {
    "totalDIDs": 127,
    "totalCredentials": 342,
    "totalTransactions": 1847,
    "totalAHINBlocks": 45,
    "activeAgents": 23,
    "last24hTransactions": 156,
    "averageTransactionValue": "87500000000000000000",
    "systemHealth": {
      "api": "healthy",
      "blockchain": "healthy",
      "database": "healthy",
      "ahinService": "healthy"
    }
  }
}
```

6.6.2 Agent 统计

端点: `GET /api/v1/stats/agent/:address`

响应: `200 OK`

json

```
{
  "success": true,
  "data": {
    "agent": "0xabcdefabcdefabcdefabcdefabcdefabcdefabcd",
    "reputation": 95,
    "rank": 12,
    "totalTransactions": 847,
    "completedTransactions": 831,
    "successRate": 98.1,
    "totalVolume": "741250000000000000000000",
    "averageTransactionValue": "875000000000000000000",
    "credentials": {
      "issued": 15,
      "revoked": 0,
      "expired": 2
    },
    "activity": {
      "last7Days": 34,
      "last30Days": 156
    }
  }
}
```

Webhooks

配置 Webhook

端点: `POST /api/v1/webhooks`

请求 Body:

json

```
{
  "url": "https://your-server.com/webhook",
  "events": [
    "transaction.completed",
    "vc.issued",
    "vc.revoked",
    "ahin.anchored"
  ],
  "secret": "your_webhook_secret"
}
```

响应: `201 Created`

json

```
{
  "success": true,
  "data": {
    "webhookId": "wh_1a2b3c4d",
    "url": "https://your-server.com/webhook",
    "events": ["transaction.completed", "vc.issued"],
    "secret": "your_webhook_secret",
    "status": "active"
  }
}
```

Webhook 事件格式

所有 Webhook 请求包含以下 Headers:

http

```
Content-Type: application/json
X-CAI-Event: transaction.completed
X-CAI-Signature: sha256=abc123...
X-CAI-Timestamp: 1697723471
```

transaction.completed

json

```
{
  "event": "transaction.completed",
  "timestamp": "2025-10-20T14:32:15.123Z",
  "data": {
    "transactionId": "0x1a2b...",
    "agent": "0xabcd...",
    "merchant": "0x1234...",
    "amount": "950000000000000000",
    "status": "completed",
    "completedAt": "2025-10-20T14:32:15.123Z"
  }
}
```

vc.issued

json

```
{
  "event": "vc.issued",
  "timestamp": "2025-10-20T14:32:15.123Z",
  "data": {
    "vcHash": "0x7f3e...",
    "type": "MandateVC",
    "subject": "0x1234...",
    "issuer": "did:ethr:0x9876...",
    "expiresAt": "2025-10-21T14:32:15.123Z"
  }
}
```

ahin.anchored

json

```
{
  "event": "ahin.anchored",
  "timestamp": "2025-10-20T14:35:00.000Z",
  "data": {
    "blockNumber": 16,
    "merkleRoot": "0x9f8b...",
    "transactionCount": 45,
    "txHash": "0xetherscan_tx_hash"
  }
}
```

验证 Webhook 签名

javascript

```javascript
const crypto = require('crypto');

function verifyWebhookSignature(payload, signature, secret) {
  const expectedSignature = crypto
    .createHmac('sha256', secret)
    .update(JSON.stringify(payload))
    .digest('hex');

  return `sha256=${expectedSignature}` === signature;
}

// 使用示例
app.post('/webhook', (req, res) => {
  const signature = req.headers['x-cai-signature'];
  const isValid = verifyWebhookSignature(
    req.body,
    signature,
    process.env.WEBHOOK_SECRET
  );

  if (!isValid) {
    return res.status(401).send('Invalid signature');
  }

  // 处理事件
  console.log('Event:', req.body.event);
  res.status(200).send('OK');
});
```

SDK 示例

JavaScript/TypeScript

安装

bash

```bash
npm install @cai-framework/sdk
```

初始化

typescript

```typescript
import { CAIClient } from '@cai-framework/sdk';

const client = new CAIClient({
  apiKey: process.env.CAI_API_KEY,
  network: 'sepolia',
  baseURL: 'https://api.cai-framework.eth'
});
```

创建 Mandate VC

typescript

```typescript
const mandate = await client.vc.createMandate({
  subject: '0x1234...',
  agent: '0xabcd...',
  budget: '100000000000000000000',
  expiry: 86400,
  whitelist: ['merchant1.eth']
});

console.log('VC Hash:', mandate.vcHash);
console.log('Signature:', mandate.signature);
```

验证 VC

typescript

```typescript
const isValid = await client.vc.verify(vcHash);
console.log('Is valid:', isValid);
```

添加到 AHIN

typescript

```typescript
const ahinTx = await client.ahin.addTransaction({
  transactionId: '0x1a2b...',
  mandateVC: '0x7f3e...',
  cartHash: '0x9a8b...',
  receiptHash: '0xfedc...'
});

console.log('Queue position:', ahinTx.queuePosition);
```

生成审计报告

typescript

```typescript
const bundle = await client.audit.generateBundle({
  transactionId: '0x1a2b...',
  mandateVC: '0x7f3e...',
  cartHash: '0x9a8b...',
  receiptHash: '0xfedc...'
});

// 下载为文件
await bundle.download('audit-report.json');
```

查询统计

typescript

```typescript
const stats = await client.stats.getSystem();
console.log('Total DIDs:', stats.totalDIDs);
console.log('Total Transactions:', stats.totalTransactions);

const agentStats = await client.stats.getAgent('0xabcd...');
console.log('Agent reputation:', agentStats.reputation);
console.log('Success rate:', agentStats.successRate);
```

Python

安装

bash

```
pip install cai-framework-sdk
```

使用示例

python

```python
from cai_framework import CAIClient

# 初始化客户端
client = CAIClient(
    api_key='YOUR_API_KEY',
    network='sepolia'
)

# 创建 Mandate VC
mandate = client.vc.create_mandate(
    subject='0x1234...',
    agent='0xabcd...',
    budget='100000000000000000000',
    expiry=86400
)

print(f"VC Hash: {mandate['vcHash']}")

# 验证 VC
is_valid = client.vc.verify(mandate['vcHash'])
print(f"Is valid: {is_valid}")

# 添加到 AHIN
ahin_tx = client.ahin.add_transaction(
    transaction_id='0x1a2b...',
    mandate_vc='0x7f3e...',
    cart_hash='0x9a8b...',
    receipt_hash='0xfedc...'
)

print(f"Queue position: {ahin_tx['queuePosition']}")

# 查询统计
stats = client.stats.get_system()
print(f"Total DIDs: {stats['totalDIDs']}")

agent_stats = client.stats.get_agent('0xabcd...')
print(f"Agent reputation: {agent_stats['reputation']}")
```

---

cURL 完整示例

bash

```bash
#!/bin/bash

API_KEY="YOUR_API_KEY"
BASE_URL="https://api.cai-framework.eth/api/v1"

# 1. 创建 Mandate VC
echo "Creating Mandate VC..."
MANDATE_RESPONSE=$(curl -s -X POST "$BASE_URL/vc/mandate" \
  -H "Authorization: Bearer $API_KEY" \
  -H "Content-Type: application/json" \
  -d '{
    "subject": "0x1234567890123456789012345678901234567890",
    "agent": "0xabcdefabcdefabcdefabcdefabcdefabcdefabcd",
    "budget": "100000000000000000000",
    "expiry": 86400
  }')

VC_HASH=$(echo $MANDATE_RESPONSE | jq -r '.data.vcHash')
echo "VC Hash: $VC_HASH"

# 2. 验证 VC
echo "Verifying VC..."
curl -s -X GET "$BASE_URL/vc/verify/$VC_HASH" \
  -H "Authorization: Bearer $API_KEY" | jq

# 3. 创建 Cart VC
echo "Creating Cart VC..."
CART_RESPONSE=$(curl -s -X POST "$BASE_URL/vc/cart" \
  -H "Authorization: Bearer $API_KEY" \
  -H "Content-Type: application/json" \
  -d '{
    "subject": "0xabcdefabcdefabcdefabcdefabcdefabcdefabcd",
    "cartHash": "0x9a8b7c6d5e4f3a2b1c0d9e8f7a6b5c4d3e2f1a0b",
    "items": [
      {
        "id": "item_001",
        "name": "AI Training Dataset",
        "price": "500000000000000000",
        "quantity": 1
      }
    ],
    "totalAmount": "500000000000000000"
  }')
```

```
CART_HASH=$(echo $CART_RESPONSE | jq -r '.data.cartHash')
echo "Cart Hash: $CART_HASH"

# 4. 添加到 AHIN
echo "Adding to AHIN..."
AHIN_RESPONSE=$(curl -s -X POST "$BASE_URL/ahin/transaction" \
  -H "Authorization: Bearer $API_KEY" \
  -H "Content-Type: application/json" \
  -d "{
    \"transactionId\": \"0x1a2b3c4d5e6f7a8b9c0d1e2f3a4b5c6d\",
    \"mandateVC\": \"$VC_HASH\",
    \"cartHash\": \"$CART_HASH\",
    \"receiptHash\": \"0xfedcba0987654321fedcba0987654321\"
  }")

AHIN_TX_ID=$(echo $AHIN_RESPONSE | jq -r '.data.ahinTxId')
echo "AHIN Tx ID: $AHIN_TX_ID"

# 5. 查询 AHIN 统计
echo "Getting AHIN stats..."
curl -s -X GET "$BASE_URL/ahin/stats" \
  -H "Authorization: Bearer $API_KEY" | jq

# 6. 生成审计报告
echo "Generating audit bundle..."
curl -s -X POST "$BASE_URL/audit/bundle" \
  -H "Authorization: Bearer $API_KEY" \
  -H "Content-Type: application/json" \
  -d "{
    \"transactionId\": \"0x1a2b3c4d5e6f7a8b9c0d1e2f3a4b5c6d\",
    \"mandateVC\": \"$VC_HASH\",
    \"cartHash\": \"$CART_HASH\",
    \"receiptHash\": \"0xfedcba0987654321\"
  }" | jq > audit-bundle.json

echo "Audit bundle saved to audit-bundle.json"
```

测试环境

Sandbox API

开发和测试使用 Sandbox 环境：

Base URL: https://sandbox.cai-framework.eth/api/v1

测试账户

json

```
{
  "testAccounts": [
    {
      "address": "0x1234567890123456789012345678901234567890",
      "privateKey": "0xtest_key_1",
      "role": "user",
      "balance": "100000000000000000000"
    },
    {
      "address": "0xabcdefabcdefabcdefabcdefabcdefabcdefabcd",
      "privateKey": "0xtest_key_2",
      "role": "agent",
      "balance": "500000000000000000000"
    }
  ]
}
```

测试 API Key

Sandbox Key: cai_sk_test_1a2b3c4d5e6f7a8b9c0d

变更日志

v1.0.0 (2025-10-20)

新增:

- ✅ 完整的 VC 签发和验证 API
- ✅ AHIN 批量锚定服务

- ✅ 审计报告生成功能
- ✅ DID 注册和查询
- ✅ 交易查询和统计
- ✅ Webhook 事件通知
- ✅ 速率限制和错误处理

改进:

- 优化 Merkle 证明生成性能
- 增强请求验证
- 改进错误信息

支持

文档

- 完整文档: https://docs.cai-framework.eth
- API 参考: https://docs.cai-framework.eth/api
- 教程: https://docs.cai-framework.eth/tutorials

社区

- Discord: https://discord.gg/cai-framework
- GitHub: https://github.com/cai-framework
- Twitter: @CAI_Framework

联系方式

- 技术支持: dev@cai-framework.eth
- 商务合作: bd@cai-framework.eth
- 安全问题: security@cai-framework.eth

附录

A. 状态码快速参考

状态码	含义	常见场景
200	OK	成功请求
201	Created	资源创建成功
202	Accepted	请求已接受，异步处理
400	Bad Request	参数验证失败
401	Unauthorized	缺少或无效的 API Key
403	Forbidden	权限不足
404	Not Found	资源不存在
429	Too Many Requests	超过速率限制
500	Internal Server Error	服务器错误
503	Service Unavailable	服务暂时不可用

B. 时间格式

所有时间戳使用 ISO 8601 格式：

```
2025-10-20T14:32:15.123Z
```

Unix 时间戳（秒）：

```
1697723471
```

C. 金额格式

所有金额使用 wei 单位，字符串格式：

json

```
{
  "amount": "10000000000000000000"
}
```

转换示例：

javascript

```
// ETH to wei
const amountWei = ethers.utils.parseEther("100").toString();

// wei to ETH
const amountEth = ethers.utils.formatEther("100000000000000000000");
```

D. 地址格式

所有以太坊地址使用校验和格式（EIP-55）：

```
0x1234567890123456789012345678901234567890
```

验证地址：

javascript

```
const isValid = ethers.utils.isAddress(address);
const checksummed = ethers.utils.getAddress(address);
```

文档版本: 1.0.0
生成时间: 2025-10-20
维护者: CAI Framework Team