AgenticCommerce API User Guide

Version 1.1.0 January 2026

Table of Contents

---

1. Introduction

AgenticCommerce is an autonomous AI agent commerce platform that enables:

- **AI Agent Management**: Create, manage, and run autonomous AI agents
- **X402 Payment Protocol**: HTTP 402-based micropayments using USDC
- **Circle Arc Blockchain**: Sub-second settlement on Circle's Arc network
- **Autonomous Execution**: Agents can research, decide, and execute payments

Key Features

Feature	Description
AI Agents	GPT-4o powered autonomous agents with budget management
X402 Payments	Spec-compliant V2 implementation with EIP-3009 signatures
Multi-Network	Support for Arc, Base, and Ethereum networks
Auto-Pay	Agents automatically handle 402 payment challenges

---

2. Getting Started

Base URL

```
Production: https://api.agenticcommerce.com
Development: https://localhost:7098
```

Quick Start

1. Check API health:

```
curl https://localhost:7098/health
```

2. Create an agent:

```
curl -X POST https://localhost:7098/api/agents \
  -H "Content-Type: application/json" \
  -d '{"name": "MyAgent", "budget": 10.0}'
```

3. Run a task:

```
curl -X POST https://localhost:7098/api/agents/{agentId}/run \
  -H "Content-Type: application/json" \
  -d '{"task": "Research AI API pricing and recommend the best option"}'
```

---

3. Authentication

The standard API currently uses wallet-based authentication via the X402 payment protocol. When calling paid endpoints, authentication is implicit in the signed payment payload.

Headers

Header	Description
`Content-Type`	`application/json` for all POST/PUT requests
`X-PAYMENT`	Base64-encoded signed payment payload (for paid endpoints)

---

4. API Endpoints

Health Check

GET /health

Check if the API is running.

Response:

```
Healthy
```

GET /

Get API information and available features.

Response:

```
{
  "service": "Agentic Commerce Backend",
  "version": "v1.1.0",
  "status": "Running",
  "features": [
    "Circle Arc Blockchain",
    "Circle Gateway (cross-chain USDC)",
    "X402 Payment Facilitation",
    "AI Agent Orchestration",
    "Policy Engine (Enterprise)"
  ],
  "endpoints": {
    "swagger": "/swagger",
    "health": "/health",
```

```
    "policies": "/api/policies"
  }
}
```

Agents API

GET /api/agents

List all agents.

Response:

```
[
  {
    "id": "agent_abc123",
    "name": "ResearchAgent",
    "description": "AI research assistant",
    "budget": 10.0,
    "currentBalance": 8.50,
    "status": "Active",
    "walletAddress": "0x6255d8dd3f84ec460fc8b07db58ab06384a2f487",
    "capabilities": ["research", "payments"],
    "createdAt": "2026-01-18T10:00:00Z"
  }
]
```

GET /api/agents/{agentId}

Get a specific agent by ID.

Parameters:

Parameter	Type	Description
agentId	string	The agent's unique identifier

Response: Same as list item above.

GET /api/agents/{agentId}/info

Get detailed agent information including transaction history.

Response:

```
{
  "id": "agent_abc123",
  "name": "ResearchAgent",
  "description": "AI research assistant",
  "budget": 10.0,
  "currentBalance": 8.50,
  "status": "Active",
  "walletAddress": "0x6255d8dd3f84ec460fc8b07db58ab06384a2f487",
  "capabilities": ["research", "payments"],
  "transactionCount": 3,
  "transactionIds": ["tx_1", "tx_2", "tx_3"],
  "createdAt": "2026-01-18T10:00:00Z",
```

```
  "lastActiveAt": "2026-01-18T14:30:00Z"
}
```

POST /api/agents

Create a new agent.

Request Body:

```
{
  "name": "MyAgent",
  "description": "Optional description",
  "budget": 10.0,
  "capabilities": ["research", "payments", "http"]
}
```

Field	Type	Required	Description
name	string	Yes	Agent display name
description	string	No	Agent description
budget	decimal	Yes	Initial USDC budget
capabilities	array	No	List of agent capabilities

Response: The created agent object.

POST /api/agents/{agentId}/run

Execute a task with an agent.

Request Body:

```
{
  "task": "Research AI API providers and recommend the best option for image generation under $50/month"
}
```

Response:

```
{
  "agentId": "agent_abc123",
  "success": true,
  "result": "Based on my research, I recommend Stability.ai for image generation...",
  "amountSpent": 0.01,
  "transactionIds": ["tx_abc123"],
  "startedAt": "2026-01-18T14:30:00Z",
  "completedAt": "2026-01-18T14:30:15Z"
}
```

DELETE /api/agents/{agentId}

Delete an agent.

Response: 204 No Content on success.

X402 Payment Protocol

The X402 protocol enables HTTP-based micropayments. When you request a paid resource without payment, you receive a 402 response with payment requirements.

GET /api/x402-example/simple

A simple paid endpoint requiring $0.01 USDC.

Without Payment:

```
HTTP/1.1 402 Payment Required
X-PAYMENT-REQUIRED: <base64-encoded requirements>
```

With Payment:

```
HTTP/1.1 200 OK
X-PAYMENT-RESPONSE: {"success": true, "transactionHash": "..."}

{"message": "Payment received!", "data": "..."}
```

GET /api/x402-example/free

A free endpoint (no payment required).

Response:

```
{
  "message": "This endpoint is free!"
}
```

GET /api/x402-example/ai-analysis

Premium AI analysis endpoint requiring $0.01 USDC.

Response (after payment):

```
{
  "analysis": "AI-powered analysis results...",
  "model": "gpt-4o",
  "timestamp": "2026-01-18T14:30:00Z"
}
```

Logs API

GET /api/logs

Get recent application logs.

Query Parameters:

Parameter	Type	Default	Description
count	int	100	Number of logs to return
level	string	null	Filter by log level

Response:

```
[
  {
    "id": 1,
    "timestamp": "2026-01-18T14:30:00Z",
    "level": "Information",
    "message": "Agent task completed",
    "source": "AgentService",
    "requestPath": "/api/agents/run"
  }
]
```

GET /api/logs/errors

Get recent error logs.

GET /api/logs/warnings

Get recent warning logs.

POST /api/logs/test

Test the logging system.

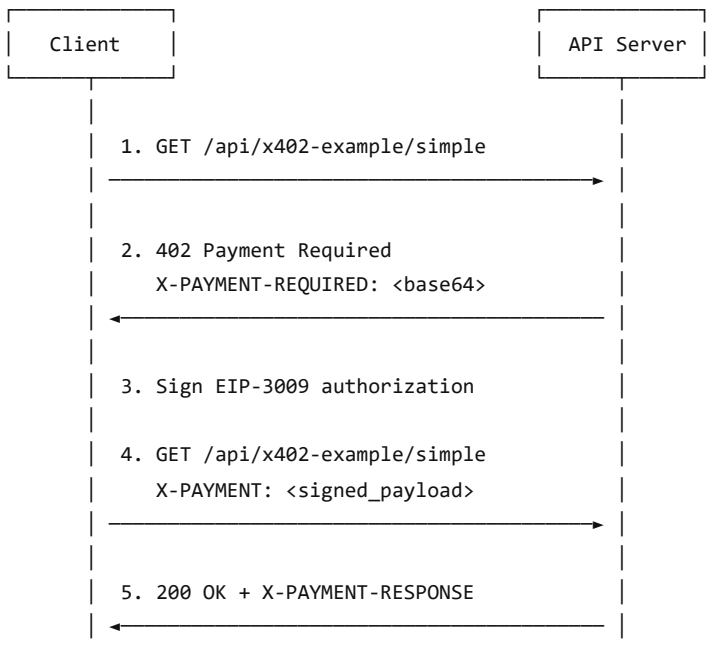
Request Body:

```
{
  "level": "Information",
  "message": "Test log entry"
}
```

---

5. X402 Payment Flow

Overview



Payment Required Response

When a payment is required, the server returns:

Headers:

- `X-PAYMENT-REQUIRED` : Base64-encoded JSON with payment options

Decoded X-PAYMENT-REQUIRED:

```
{
  "x402Version": 2,
  "accepts": [
    {
      "scheme": "exact",
      "network": "arc-testnet",
      "maxAmountRequired": "10000",
      "resource": "/api/x402-example/simple",
      "description": "API Request - $0.01 USDC",
      "payTo": "0x6255d8dd3f84ec460fc8b07db58ab06384a2f487",
      "asset": "0x3600000000000000000000000000000000000000"
    }
  ]
}
```

Payment Payload

To complete payment, send an `X-PAYMENT` header with a signed EIP-3009 authorization:

```
{
  "x402Version": 2,
  "scheme": "exact",
  "network": "arc-testnet",
  "payload": {
    "signature": "0x...",
    "authorization": {
      "from": "0xYourWallet",
      "to": "0xMerchant",
      "value": "10000",
      "validAfter": 0,
      "validBefore": 1768748280,
      "nonce": "0x..."
    }
  }
}
```

Supported Networks

Network	Chain ID	USDC Contract
arc-testnet	5042002	0x3600000000000000000000000000000000000000
arc-mainnet	TBD	TBD
base-sepolia	84532	0x036CbD53842c5426634e7929541eC2318f3dCF7e
base-mainnet	8453	0x833589fCD6eDb6E08f4c7C32D4f71b54bdA02913

---

6. Error Handling

HTTP Status Codes

Code	Description
200	Success
201	Created
204	No Content (successful delete)
400	Bad Request - Invalid input
402	Payment Required - X402 payment needed
404	Not Found
500	Internal Server Error

Error Response Format

```
{
  "type": "https://tools.ietf.org/html/rfc9110#section-15.5.1",
  "title": "Bad Request",
  "status": 400,
  "errors": {
    "name": ["The name field is required."]
  }
}
```

---

7. Code Examples

Python - Create and Run Agent

```
import requests

BASE_URL = "https://localhost:7098"

# Create agent
agent = requests.post(f"{BASE_URL}/api/agents", json={
    "name": "PythonAgent",
    "budget": 5.0,
    "capabilities": ["research", "http"]
}).json()

print(f"Created agent: {agent['id']}")

# Run task
result = requests.post(
    f"{BASE_URL}/api/agents/{agent['id']}/run",
    json={"task": "What are the top 3 AI APIs for text generation?"}
).json()
```

```
print(f"Result: {result['result']}")
print(f"Amount spent: ${result['amountSpent']}")
```

JavaScript - X402 Payment

```
const axios = require('axios');

async function callPaidEndpoint(url) {
  // First request - get payment requirements
  try {
    const response = await axios.get(url);
    return response.data;
  } catch (error) {
    if (error.response?.status === 402) {
      const paymentRequired = error.response.headers['x-payment-required'];
      const requirements = JSON.parse(
        Buffer.from(paymentRequired, 'base64').toString()
      );

      // Sign payment (implement your signing logic)
      const signedPayload = await signPayment(requirements);

      // Retry with payment
      const paidResponse = await axios.get(url, {
        headers: { 'X-PAYMENT': signedPayload }
      });

      return paidResponse.data;
    }
    throw error;
  }
}
```

cURL - Full Agent Workflow

```
# 1. Create agent
AGENT=$(curl -s -X POST https://localhost:7098/api/agents \
  -H "Content-Type: application/json" \
  -d '{"name": "CurlAgent", "budget": 1.0}')

AGENT_ID=$(echo $AGENT | jq -r '.id')
echo "Agent ID: $AGENT_ID"

# 2. Check agent info
curl -s https://localhost:7098/api/agents/$AGENT_ID/info | jq

# 3. Run a task
curl -s -X POST https://localhost:7098/api/agents/$AGENT_ID/run \
  -H "Content-Type: application/json" \
  -d '{"task": "Check the current USDC balance"}' | jq

# 4. Delete agent
curl -s -X DELETE https://localhost:7098/api/agents/$AGENT_ID
```

---

8. Appendix

Agent Capabilities

Capability	Description
research	Access to AI/API research tools
payments	Can execute USDC payments
http	Can make HTTP requests with x402 auto-pay
analysis	Cost analysis and comparison tools

Agent Status Values

Status	Description
Active	Agent is ready to accept tasks
Working	Agent is currently executing a task
Paused	Agent is temporarily disabled
Error	Agent encountered an error

USDC Decimals

USDC uses 6 decimal places:

- $1.00 = 1,000,000 smallest units
- $0.01 = 10,000 smallest units
- $0.001 = 1,000 smallest units

Rate Limits

Endpoint	Limit
Agent creation	10/minute
Task execution	30/minute
X402 payments	100/minute

Support

- **Documentation**: https://docs.agenticcommerce.com
- **API Status**: https://status.agenticcommerce.com
- **GitHub Issues**: https://github.com/kmatthewsio/AgenticCommerce/issues

---

AgenticCommerce API User Guide v1.1.0 - January 2026