VCTT-AGI API Documentation

Version: 2.0.0 (Phase 2 Complete)
Base URL: `http://localhost:8000` (development) or your production URL

Overview

The VCTT-AGI API provides endpoints for:
1. **Session Management** - Create and manage conversation sessions
2. **Analytics** - Retrieve insights, trust metrics, and patterns (NEW in Phase 2)
3. **System** - Health checks and documentation

Authentication

Currently no authentication required. Will be added in Phase 3.

Session Management

POST /api/v1/session/start

Create a new conversation session.

Request Body:

```json
{
  "user_id": "string (required)",
  "input": "string (required) - First message"
}
```

Response (201 Created):

```json
{
  "session_id": "uuid-string"
}
```

Example:

```bash
curl -X POST http://localhost:8000/api/v1/session/start \
  -H "Content-Type: application/json" \
  -d '{"user_id":"user123","input":"Hello!"}'
```

POST /api/v1/session/step

Send a message in an existing session.

Request Body:

```
{
  "session_id": "string (required)",
  "input": "string (required)"
}
```

Response (200 OK):

```
{
  "response": "AI-generated response",
  "internal_state": {
    "trust_tau": 0.95,
    "sim": {
      "tension": 0.1,
      "uncertainty": 0.05,
      "emotional_intensity": 0.2
    },
    "regulation": "normal",
    "repair_count": 0,
    "contradiction": 0.04
  },
  "agent_logs": ["Agent activity logs..."]
}
```

GET /api/v1/session/:id

Retrieve session details and message history.

Response (200 OK):

```
{
  "session_id": "uuid",
  "user_id": "user123",
  "created_at": "2025-11-17T10:00:00Z",
  "messages": [
    {
      "id": "msg-uuid",
      "role": "user",
      "content": "Hello!",
      "timestamp": "2025-11-17T10:00:01Z"
    },
    {
      "id": "msg-uuid",
      "role": "assistant",
      "content": "Hi! How can I help?",
      "timestamp": "2025-11-17T10:00:02Z"
    }
  ],
  "internal_state": {...}
}
```

Analytics Endpoints (Phase 2)

GET /analytics/sessions

List all sessions with metadata.

Query Parameters:
- `user_id` (optional) - Filter by user
- `limit` (optional, default: 50) - Maximum sessions to return

Response (200 OK):

```
{
  "total": 25,
  "sessions": [
    {
      "session_id": "uuid",
      "user_id": "user123",
      "created_at": "2025-11-17T10:00:00Z",
      "message_count": 10,
      "last_activity": "2025-11-17T10:15:00Z",
      "trust_tau": 0.92,
      "repair_count": 1
    }
  ]
}
```

GET /analytics/sessions/:sessionId/history

Get full conversation history including messages and internal states.

Response (200 OK):

```
{
  "session_id": "uuid",
  "user_id": "user123",
  "created_at": "2025-11-17T10:00:00Z",
  "messages": [...],
  "internal_state": {
    "trust_tau": 0.92,
    "sim": {...},
    "regulation": "normal",
    "repair_count": 1
  },
  "last_updated": "2025-11-17T10:15:00Z"
}
```

GET /analytics/trust-metrics

Track trust evolution over time.

Query Parameters:
- `user_id` (optional) - Filter by user
- `sessionId` (optional) - Single session metrics

Response (200 OK):

```json
{
  "total": 50,
  "metrics": [
    {
      "session_id": "uuid",
      "user_id": "user123",
      "timestamp": "2025-11-17T10:00:00Z",
      "trust_tau": 0.95,
      "contradiction": 0.03,
      "sim": {
        "tension": 0.1,
        "uncertainty": 0.05,
        "emotional_intensity": 0.15
      },
      "regulation": "normal",
      "repair_count": 0
    }
  ]
}
```

GET /analytics/aggregate

Get aggregate statistics across all sessions.

Query Parameters:
- `user_id` (optional) - Filter by user

Response (200 OK):

```
{
  "overview": {
    "total_sessions": 100,
    "total_messages": 850,
    "avg_messages_per_session": 8.5
  },
  "trust_metrics": {
    "average_trust_tau": 0.91,
    "min_trust": 0.75,
    "max_trust": 1.0
  },
  "repair_metrics": {
    "total_repairs": 15,
    "avg_repairs_per_session": 0.15,
    "max_repairs": 3
  },
  "regulation": {
    "normal": 85,
    "clarify": 10,
    "slow_down": 5
  },
  "timestamp": "2025-11-17T11:00:00Z"
}
```

GET /analytics/cross-session-patterns

AI-powered pattern analysis across sessions.

Query Parameters:
- `user_id` (optional) - Filter by user

Response (200 OK):

```
{
  "total_sessions": 100,
  "patterns": [
    {
      "type": "trust_evolution",
      "data": [
        {
          "session_id": "uuid",
          "created_at": "2025-11-17T10:00:00Z",
          "trust_tau": 0.88
        }
      ]
    },
    {
      "type": "repair_frequency",
      "data": [...]
    },
    {
      "type": "engagement",
      "data": [
        {
          "session_id": "uuid",
          "user_messages": 5,
          "assistant_messages": 5,
          "total_exchanges": 5
        }
      ]
    }
  ],
  "insights": [
    "Trust has increased by 7.2% over time, indicating improving conversation qual-
ity",
    "Average conversation length: 4.2 exchanges per session",
    "Average repair count is 0.15, indicating low conversational complexity"
  ],
  "timestamp": "2025-11-17T11:00:00Z"
}
```

System Endpoints

GET /health

System health check.

Response (200 OK):

```
{
  "status": "healthy",
  "timestamp": "2025-11-17T11:00:00Z",
  "service": "VCTT-AGI Coherence Kernel",
  "version": "1.0.0"
}
```

GET /api

Interactive Swagger UI documentation.

Access at: `http://localhost:8000/api`

Error Responses

All errors follow this format:

```json
{
  "statusCode": 404,
  "message": "Session not found",
  "error": "Not Found"
}
```

Common Status Codes:
- `400` - Bad Request (invalid input)
- `404` - Not Found (session doesn’t exist)
- `429` - Too Many Requests (rate limit)
- `500` - Internal Server Error

Rate Limiting

To be implemented in Phase 3

WebSocket Support

Planned for Phase 3

Examples

Complete Session Flow

```
# 1. Create session
SESSION_ID=$(curl -s -X POST http://localhost:8000/api/v1/session/start \
  -H "Content-Type: application/json" \
  -d '{"user_id":"demo","input":"Hello"}' | jq -r '.session_id')

# 2. Send message
curl -X POST http://localhost:8000/api/v1/session/step \
  -H "Content-Type: application/json" \
  -d "{\"session_id\":\"$SESSION_ID\",\"input\":\"How does trust work?\"}"

# 3. Get session history
curl http://localhost:8000/analytics/sessions/$SESSION_ID/history | jq '.'

# 4. View trust metrics
curl http://localhost:8000/analytics/trust-metrics?sessionId=$SESSION_ID | jq '.'
```

Change Log

v2.0.0 (Phase 2 - November 17, 2025)

- ✅ Added PostgreSQL persistence
- ✅ Added 5 analytics endpoints
- ✅ Added cross-session learning
- ✅ Added trust evolution tracking

v1.0.0 (Phase 1)

- ✅ Initial session management
- ✅ Multi-agent orchestration
- ✅ Trust metric calculation

Support

For issues or questions:
- Check Swagger UI: `/api`
- Review test results: `./test-phase2.sh`
- See Phase 2 documentation: `PHASE_2_STATUS.md`

Last Updated: November 17, 2025
API Version: 2.0.0-phase2