




VCTT-AGI Endpoint Connection Status




Last Updated: 2025-11-22 after conversation history implementation

Summary

Category	Total End-points	 Connected	 Docu-mented Only	 Frontend Calls But Miss-ing
Core Chat	3	3	0	0
IDE/DeepAgent	12	7	5	0
Goals	10	2	8	0
Safety	16	2	14	0
Analytics	8	2	6	0
Knowledge Graph	11	0	11	0
Memory	8	0	8	0
Scheduler	6	0	6	0
Coach & Skills	4	0	0	4
LLM Committee	2	0	2	0
Truth Mycelium	3	0	3	0
Health	1	1	0	0
TOTAL	84	17	63	4

CONNECTED & ACTIVELY USED (17)



Core Chat (3/3)

-  POST /api/v1/session/start - ChatbotLanding.tsx:152
-  POST /api/v1/session/step - ChatbotLanding.tsx:215
-  GET /api/v1/session/:id - ChatbotLanding.tsx:77 (via api.getSessionHistory)



IDE/DeepAgent (7/12)

-  GET /health - DeepAgent.tsx:159 (startup health check)
-  POST /api/deep/execute - DeepAgent.tsx:387 (terminal commands)
-  POST /api/ide/workspace/load - DeepAgent.tsx:205 (workspace context)
-  POST /api/ide/file-operation - FileTree.tsx, FileTreeWithIcons.tsx (file CRUD)
-  POST /api/ide/code-edit - AIChat.tsx:234 (Cmd+K AI editing)
-  POST /api/ide/search-files - SearchModal.tsx:35 (Cmd+Shift+F)
-  GET /api/ide/file-tree - FileTree components (file browser)



Goals (2/10)

-  GET /api/goals/active - AIChat.tsx:196 (active goals display)
-  GET /api/goals - AIChat.tsx:193 (all goals with status filter)

Safety (2/16)

-  GET /api/safety/status - AIChat.tsx:194 (safety status indicator)
-  GET /api/safety/charter - Public safety charter page (planned)

Analytics (2/8)

-  GET /api/v1/analytics/sessions - ChatbotLanding.tsx:42 (conversation history list)
-  GET /api/v1/analytics/sessions/:id/history - ChatbotLanding.tsx:125 (full conversation)

DOCUMENTED BUT NOT CONNECTED YET (63)

These endpoints exist in the backend and have Swagger docs, but no frontend component uses them yet.

IDE/DeepAgent (5)

- POST /api/ide/run-tests
- POST /api/ide/code-analysis
- GET /api/ide/deployment-status
- POST /api/ide/deploy
- GET /api/ide/image-preview

Goals (8)

- POST /api/goals
- GET /api/goals/tree
- GET /api/goals/state-awareness
- GET /api/goals/:id
- PUT /api/goals/:id
- POST /api/goals/:id/status
- POST /api/goals/:id/progress
- DELETE /api/goals/:id

Safety (14)

- POST /api/safety/kill-switch
- POST /api/safety/kill-switch/deactivate

- `POST /api/safety/mode`
- `POST /api/safety/memory/enable`
- `POST /api/safety/memory/disable`
- `GET /api/safety/audit`
- `GET /api/safety/audit-log`
- `POST /api/safety/config`
- `GET /api/safety/summary`
- `GET /api/safety/checks`
- `GET /api/safety/checks/goal`
- `GET /api/safety/checks/knowledge`
- `GET /api/safety/checks/coach`
- `POST /api/safety/checks`
- `GET /api/safety/guardian/status`

Knowledge Graph (11)

- `POST /api/knowledge/extract`
- `POST /api/knowledge/entity`
- `GET /api/knowledge/entity/:id`
- `POST /api/knowledge/relationship`
- `GET /api/knowledge/query`
- `GET /api/knowledge/subgraph/:id`
- `GET /api/knowledge/concepts`
- `GET /api/knowledge/concepts/:id/entities`
- `DELETE /api/knowledge/entity/:id`
- `GET /api/knowledge/graph`
- `GET /api/knowledge/search`

Memory (8)

- `POST /api/memory/consent/grant`
- `POST /api/memory/consent/revoke`
- `GET /api/memory/consent/:userId`
- `POST /api/memory/store`
- `GET /api/memory/retrieve`
- `DELETE /api/memory/:memoryId`
- `DELETE /api/memory/all/:userId`
- `GET /api/memory/export/:userId`

Scheduler (6)

- `POST /api/scheduler/schedule`
- `PUT /api/scheduler/approve/:id`
- `DELETE /api/scheduler/cancel/:id`
- `GET /api/scheduler/pending`
- `GET /api/scheduler/goal/:goalId`
- `GET /api/scheduler/status/:id`

Analytics (6)

- GET /api/v1/analytics/trust-metrics
- GET /api/v1/analytics/aggregate
- GET /api/v1/analytics/cross-session-patterns
- GET /api/v1/analytics/cost
- GET /api/v1/analytics/performance
- GET /api/v1/analytics/export

LLM Committee (2)

- GET /api/v1/analytics/llm-committee/session/:id
- GET /api/v1/analytics/llm-committee/global

Truth Mycelium (3)

- GET /truth-mycelium/
- GET /truth-mycelium/stats
- GET /truth-mycelium/health

✗ FRONTEND CALLS BUT BACKEND MISSING (4)

These are called by the frontend but return 404 because controllers don't exist!


Coach & Skills (4)

- ✗ GET /api/coach/proposals?status=pending - CoachDashboard.tsx:27
- ✗ GET /api/skills/candidates?minTau=0.85&minCount=3 - CoachDashboard.tsx:40
- ✗ POST /api/coach/proposals/:id/approve - CoachDashboard.tsx:58
- ✗ POST /api/coach/proposals/:id/reject - CoachDashboard.tsx:71

Fix Required: Create `coach.controller.ts` and `skills.controller.ts`

Implementation Priority

High Priority (User-Facing Features)

1. **Goals Panel** - Connect remaining 8 goal endpoints to enable full CRUD
2. **Coach/Skills** - Implement missing controllers (4 endpoints) to fix 404s
3. **Conversation Search** -  **IMPLEMENTED** (now live with search bar)

Medium Priority (Admin Features)

1. **Safety Dashboard** - Connect 14 safety endpoints for admin controls
2. **Analytics Dashboard** - Connect 6 analytics endpoints for metrics
3. **Scheduler UI** - Connect 6 scheduler endpoints for task management

Low Priority (Advanced Features)

1. **Knowledge Graph UI** - Connect 11 knowledge endpoints for graph visualization
2. **Memory System UI** - Connect 8 memory endpoints for GDPR management
3. **LLM Committee Stats** - Connect 2 LLM Committee endpoints for Jazz Team analytics

Recently Completed

Conversation History & Search (Nov 22, 2025)

Problem: User reported “Last 50 questions” visible in LLM panel but no UI to access them

Solution:

- Connected `GET /api/v1/analytics/sessions` to load conversation list
- Connected `GET /api/v1/analytics/sessions/:id/history` to load full conversations
- Added search bar to LeftSidebar (search by title or message content)
- Shows conversation metadata (message count, trust score, timestamp)
- Lazy-loads full messages when clicking on conversation

Before: Only current session visible


After: Full ChatGPT-style conversation history with search

Notes

Why So Many Unconnected Endpoints?

- Backend was built comprehensively to support all planned features
- Frontend is being built iteratively based on user priority
- Having backend ready allows rapid frontend iteration

How to Connect a New Endpoint

1. Check `src/services/api.ts` for existing service method
2. Import and call method in component
3. Handle loading states and errors
4. Update this document with  status
5. Update `API_ENDPOINTS_DOCUMENTATION.md` with usage location

Testing Endpoints Without UI

Use Swagger documentation: <https://vctt-agi-backend.onrender.com/api-docs>

Or use curl/Postman to test directly