# Bio-Quantum Multi-Agent API Bridge Specification

## Overview

The Bio-Quantum Multi-Agent API Bridge enables secure, coordinated collaboration among Claude 3, Grok 4, GPT-4o, and Manus to develop the Bio-Quantum AI Trading Platform. It supports shared resource access, session memory sync, file versioning, and task queue management, aligning with the Tier 1 demo goals (real-time UI, mock AI logic, investor-ready narrative).

## Core Components

## 1. Agent Gateway

Purpose: Authenticates agents and routes tasks.  
Endpoint: POST /bridge/agent-task  
  
Request:  
{  
 "agent": "string (Claude\_3, Grok\_4, GPT\_4o, Manus)",  
 "task": "string (e.g., document\_review, animate\_ui)",  
 "file": "string (e.g., HLDD\_GENESIS\_AIX\_JULY2025\_UPDATED.docx)",  
 "context": "string (e.g., documentation, ui)",  
 "output\_format": "string (e.g., summary\_pdf, react\_component)"  
}  
  
Response:  
{  
 "taskId": "string",  
 "status": "Task queued",  
 "details": { ... }  
}  
  
Security: Role-based access via agent\_roles.json.

## 2. Context Sync Engine

Purpose: Maintains session memory and task history.  
Endpoint: GET /bridge/context-sync?agent={agent}  
  
Response:  
{  
 "agent": "string",  
 "context": [{ "taskId": "string", "task": "string", "context": "string" }]  
}  
  
Storage: MongoDB/Redis (mocked as in-memory for demo).  
Data: Syncs project\_goals, file\_manifest.json, version\_control\_summary.json.

## 3. File Exchange Hub

Purpose: Tracks and serves files in /mnt/data/.  
Endpoint: GET /bridge/files?filter={filter}&tag={tag}  
  
Response:  
{  
 "files": {  
 "filename": { "agent": "string", "tags": ["string"], "updated": "string" }  
 }  
}  
  
Filters: updated\_today, tag=for\_review.

## 4. Task Queue API

Purpose: Manages agent-specific task lists.  
Endpoint: GET /bridge/task-queue?agent={agent}  
  
Response:  
{  
 "agent": "string",  
 "tasks": ["string"]  
}  
  
Tasks: e.g., animate\_portfolio\_dashboard\_UI, simulate\_AAPL\_AI\_trade\_recommendation.

## 5. Security & Control Layer

Features:  
- Role-based permissions (agent\_roles.json).  
- Action logging (agent, file, timestamp).  
- Rate limiting (5 requests/minute per agent).  
  
Example Roles:  
{  
 "Claude\_3": { "permissions": ["read\_docs", "write\_docs"], "scope": "documentation" },  
 "Grok\_4": { "permissions": ["read\_code", "write\_code", "simulate"], "scope": "dev" }  
}

## Implementation

Stack: Node.js/Express (integrates with existing middleware).  
Dependencies: express, express-rate-limit, MongoDB/Redis (mocked for Tier 1).  
Deployment: Dockerized for xAI compatibility.  
Endpoints: POST /bridge/simulate-ai: Mock AI recommendations (47% return narrative).  
Extensible for Tier 2 (e.g., feedback\_channel, sandbox\_exec).

## Integration with Tier 1 Demo

UI: React dashboard consumes /bridge/simulate-ai for AI popups.  
APIs: Polygon.io data routed via /bridge/files for shared access.  
Investor Narrative: “Multi-agent collaboration delivers a world-class trading platform in 4 weeks.”

## Future Extensions

- Feedback Channel: Agent critique (Tier 2).  
- Sandbox Execution: Isolated code testing (Tier 2).  
- Token Meter: Track compute usage (Tier 2).  
- Manual Override UI: Richard reassigns tasks (Tier 2).