

Fullstack Engineering Assessment

Overview

Build the fullstack project for an AI-powered customer support system with a multi-agent architecture. A router agent should analyze incoming queries and delegate to specialized sub-agents, each with access to relevant tools. The system should maintain conversational context across messages to enable more accurate and personalized responses.

Time Allocation	Submission
12-48 hours	GitHub repo + Loom demo

Requirements

Architecture

1. Controller-Service pattern
2. Clean separation of concerns
3. Proper error handling throughout (Recommended use a middleware)

Multi-Agent System

Router Agent (Parent)

- Analyzes incoming customer queries
- Classifies intent and delegates to appropriate sub-agent
- Handles fallback for unclassified queries

Sub-Agents (implement all three):

1. Support Agent

- Handles general support inquiries, FAQs, troubleshooting
- Tools: query conversation history

2. Order Agent

- Handles order status, tracking, modifications, cancellations
- Tools: fetch order details, check delivery status

3. Billing Agent

- Handles payment issues, refunds, invoices, subscription queries
- Tools: get invoice details, check refund status

Agent Tools

- 4. Each sub-agent must have tools
- 5. Tools should query actual data from database (mock data is enough)
- 6. Seed database with sample orders, payments, conversations
- 7. The agents should have a conversation context. On the User's previous contexts.

API & Database

- 8. RESTful API endpoints for chat interactions
- 9. Streaming responses from AI agents (Recommended)
- 10. Conversation and message persistence
- 11. Real-time agent is typing indicator

API Routes



Tech Stack

Frontend	Basic UI (React / Vite)
Backend	Hono.dev
Database	PostgreSQL
ORM	Prisma / Drizzle
AI	Vercel AI SDK

Bonus Points

Hono RPC + Monorepo Setup (+30 points Guaranteed)

Set up a monorepo with Hono RPC for end-to-end type safety between backend and frontend. Use <https://turborepo.dev/> for monorepo management.

Other Bonuses

1. Rate limiting implementation
2. Usage of <https://useworkflow.dev>
3. Unit/integration tests
4. Context management / Compaction (When running out of tokens)
5. Show reasoning of the AI. Or a loader of random words like “Thinking”, “Searching”
6. Deployed live demo

Evaluation Focus

Backend architecture and code organization

Multi-agent system design and routing logic

Tool implementation and data flow

API design and error handling

Submission

12. GitHub repository with README
13. Loom video walkthrough (2-5 min)
14. Working setup instructions

Important Note

We don't want you to vibe code this entirely. Your submission will be thoroughly reviewed, and we'll ask you to walk through your code and explain your decisions. Please write the code yourself and AI assistants are fine for help, but the architecture and implementation should be yours.

Good luck !!