Yangzhen Fu

TARGET ROLE

Agentic AI Engineer

CONTACT

Location	Suzhou, China
Phone	+86 18506200312
Email	fuyangzhen@gmail.com
Website	me.mintmind.io
LinkedIn	YangzhenFu
GitHub	0xmoat
Twitter/X	@moat_mintmind

CORE SKILLS

Agent Architecture & Orchestration

LangGraph · ReAct · Plan-and-Execute · StateGraph · DAG Execution · Tool Use · MCP · Multi-step Reasoning

Memory & Knowledge Systems

mem0 · Vector Retrieval (pgvector, ChromaDB) · Hybrid Search · RRF Reranking · Long-term Memory

LLM Engineering

Claude / OpenAI API · Prompt Engineering · Context Engineering · Structured Output (Instructor) · Fine-tuning (LoRA via LLaMA Factory)

Evaluation & Observability

LLM-as-Judge · Langfuse Tracing · Prometheus · Grafana

Code Agent

Claude Code · Cursor · Spec-Driven Development · Reusable Skills

Backend

Python · C# · FastAPI · PostgreSQL · Docker

Frontend

React · TypeScript · Next.js · Tailwind CSS

EDUCATION

Beihang University
M.Eng. in Software Reliability Eng.
Sep 2015 – Apr 2018

China Jiliang University
B.Eng. in Quality Management Eng.
Sep 2011 – Jun 2015

SUMMARY

7 years at Microsoft delivering AI products such as Teams Meeting Recap; hands-on experience integrating LLM Agents end-to-end into existing enterprise workflows — led the end-to-end design and rollout of an internal on-call agent — **Incident Copilot**, closing the full Agent loop of ReAct + tool use + long-term memory in a real enterprise setting. Independently built and shipped a personal Agent project, **JobHunter** (live): a LangGraph-based ReAct + Plan-and-Execute dual-StateGraph architecture, with engineering work on the Plan subgraph's executable DAG transformation, three-layer circuit breakers for long-running tasks, and cross-channel orchestration; backed by mem0 cross-session memory, Prometheus + Langfuse end-to-end observability, and a 7-dimension LLM-as-Judge evaluation pipeline. Proven design-to-production experience across the five core Agent modules — memory, knowledge base, tool use, task planning, and evaluation loop; familiar with Agent Harness engineering principles and continuously tracking and applying SOTA Agent practices in own projects.

EXPERIENCE

MintMind

Oct 2025 – May 2026 · Suzhou

Founding Team Member

- Led the GEO (Generative Engine Optimization) technical roadmap: built an AI search visibility evaluation framework, quantifying brand citation rate and ranking position within answers from major AI search engines such as DeepSeek and Doubao.
- Served 20+ enterprise clients across 30+ products, delivering an end-to-end loop from visibility assessment to optimization strategy iteration; owned technical design and delivery.

Microsoft

May 2018 – Sep 2025 · Suzhou

Software Engineer II

- **Incident Copilot — Internal On-Call Incident Triage Agent (Designed & Built)**
  - Designed and independently delivered an internal on-call incident triage agent, integrating a LLM-driven Agent end-to-end into the existing incident response toolchain; adopted by the on-call rotation team as a daily tool, cutting average triage time on recurring incident patterns by ~50%.
  - Designed a ReAct dynamic tool-decision architecture: the LLM autonomously selects the order and timing of three tools — `search_docs` / `get_incident_summary` / `get_incident_detail` — accumulating context across rounds until information suffices for root-cause reasoning.
  - Built an engineering knowledge base on ChromaDB; `search_docs` lets the LLM construct queries on its own, with multi-vector retrieval + RRF reranking.
  - Integrated mem0 to persist long-term incident memory: when an incident closes, structured memory (metadata / root cause / teams involved / related docs) is auto-distilled from logs and discussion comments and recalled as background context in future sessions.
  - Integrated a Ragas evaluation pipeline into CI: covering Faithfulness / Answer Relevancy / Context Precision across 50 regression test cases; every PR triggers evaluation, and merges are auto-blocked when any metric falls below the absolute threshold or regresses >5% against the main baseline — preventing silent quality regressions from prompt / tool / retrieval changes.
  - Exposed retrieval over MCP (SSE transport) + Azure AD auth; team members can ask inside Code Copilot / Cursor and have the Agent auto-retrieve historical Root Cause Analysis, covering both development and on-call triage scenarios.
- **Meeting Content Platform — Cross-DC Distributed Meeting Content Platform**
  - Owned core modules of a cross-datacenter distributed meeting-content platform, unifying meeting artifacts such as notes, recording summaries, and shared chat files.
  - Provided metadata read/write capabilities to the global Teams frontend and multiple downstream AI services (including Meeting Recap), serving billions of metadata requests per day as the data backbone of the Teams AI content-generation pipeline.
- **Teams Meeting Recap Frontend**
  - Drove the structured presentation of AI-generated meeting outputs, delivering meeting notes, topics, action items, related people, and documents to end users via layered card-based pages.
- **Carina QnA Studio — Microsoft AI Configurable QnA Bot Platform**
  - Designed and built an ElasticSearch-based NLP QnA engine with multi-path hybrid retrieval (keyword + semantic) and RRF reranking — landing a hybrid-retrieval paradigm aligned with today's RAG well before the LLM wave; powering 20+ Microsoft internal business teams and 100+ customized bots in production.
  - Built a generic bot development framework and component library, providing a low-code experience for enterprise customers to quickly assemble custom knowledge-search bots.
- **O365 Dashboard — Outlook Internal Diagnostics Platform**
  - Designed and built a real-time log add-in with a configurable alert system and adaptive error classification, cutting average production-issue detection time by ~50% and reducing manual triage effort by ~40%, accelerating historical trend analysis.

PERSONAL PROJECTS

- **JobHunter** *Oct 2025 – Present*

  Full-Lifecycle Job Search Agent

  Designed & built solo (live) → jobhunter.mintmind.io

  Context: Job hunting spans search, JD analysis, resume tailoring, cover-letter writing, and application tracking; information is scattered across LinkedIn, job boards, and personal notes, forcing candidates to switch tools repeatedly with no way to compound a personalized strategy from past activity.
  Goal: Build an AI Agent that handles real-time conversation and multi-step batch tasks in one system, with cross-session long-term memory and quantifiable evaluation, covering the full job-search lifecycle end to end.
  Approach:
  - Designed a LangGraph dual-StateGraph architecture: a chat channel runs ReAct for instant responses; a batch channel uses Planner / Executor / Replanner to handle multi-step tasks.
  - Engineered the P&E subgraph: auto-converts the LLM's plan into an executable DAG (with validation / auto-repair / topological wave scheduling / serial fallback when not schedulable).
  - Implemented a three-layer circuit breaker for long-running tasks: step timeout, bounded recursion, and semantic-level loop detection — keeping batch execution stable.
  - Designed cross-channel tool calls: ReAct decides on its own when to hand off a multi-step task to the P&E subgraph, avoiding hard-coded routing.
  - Designed a scalable Tool registration and orchestration architecture, currently integrating 14 domain tools across 4 categories: intelligence (job search / JD gap analysis), document generation (tailored resume + PDF / cover letter), application tracking, and cross-channel control; new tools register through a unified interface and become auto-discoverable by the Agent.
  - Job retrieval applies SEO landing-page filtering + Instructor-based structured rerank, keeping noise out of the Agent context.
  - Integrated mem0 + pgvector for cross-session user-profile persistence and semantic retrieval (mem0 search P50 ~300ms).
  - Used differentiated memory strategies for the two channels' latency profiles: chat channel retrieves dynamically per turn, batch channel preloads once at entry.
  - Built end-to-end observability: Langfuse traces node-level spans; Prometheus instruments TTFT / TPOT / E2E latency / tool and mem0 timings; Grafana surfaces P50 / P99 curves.
  - Built a 7-dimension LLM-as-Judge evaluation pipeline (51 cases × 5 generic dimensions + P&E-specific plan_quality / replan_decision); tool_appropriateness uses a dual-track scoring scheme (Jaccard for ReAct to penalize redundant calls, recall for P&E to tolerate parallel expansion).

  Outcome: Trace analysis surfaced rubric mismatches in the judge; iterating the judge prompt recalibrated the quality baseline — relevancy mean went from 0.585 to 0.830, pass rate from 57% to 83%; mem0 search P50 stays around 300ms; the system is live with the SSE-based front-to-back streaming pipeline running stably in production.