Vaibhav Pundir

Full Stack Developer

vaibhavpundir07@gmail.com 9639727959 Shamli linkedIn github.com/pundir-07

Software Engineer with production experience building server-side systems and frontend UIs for an AI-powered developer tooling platform. Expert in the Node.js/TypeScript ecosystem, with a deep focus on agentic AI architecture, LLM tool parsing, and the Model Context Protocol (MCP). Skilled in designing reliable, multi-tenant backend systems driven by job queues (BullMQ), caching (Redis), and vector databases. Strong quantitative foundation (JEE Advanced, IIT BHU) with a proven ability to lead core architectural decisions-from state protocol design to token-aware context management.

TECHNICAL SKILLS

Languages: JavaScript, TypeScript, Python
Backend: Node.js, Express.js, RESTful API Design, Server-Sent Events (SSE), Real-time Data Streaming, Redis (caching, session/hot-data paths), BullMQ (job queues, async event processing), Python APIs, FastAPI
Databases: MongoDB (Mongoose), PostgreSQL, Redis, Vector Databases
API and Contracts: API Contract Design, Request/Response Schema Validation (Zod), Versioned Message Protocols, REST/SSE Integration with Frontend & DevOps Teams
AI/LLMs: LLM Integration (Anthropic Claude, OpenAI, OpenRouter), RAG Pipelines, Agentic Orchestration, Tool Orchestration, MCP (Model Context Protocol), Context Window Management
DevOps: Kubernetes, Docker, GitHub Actions, CI/CD Pipelines, AWS (deployment & observability), Vitest, pnpm monorepo, Turbo
Frontend: React, Next.js, Angular, Tailwind CSS, VS Code Webview API, Vite

PROFESSIONAL EXPERIENCE

Software Developer. AI-Powered Developer Tooling (VS Code Extension) 06/2025 – 07/2026
Codemate.ai

- Built and maintained the React-based webview UI (chat interface, task history, settings panels) using TypeScript, implementing real-time streaming of LLM responses via VS Code's message-passing API between the extension host and webview.
- Designed and extended the core agent orchestration layer in Node.js/TypeScript, handling tool-call parsing (XML and native function-calling formats), multi-turn conversation state, and provider-agnostic integration across LLM APIs (Anthropic, OpenAI, OpenRouter, local models).
- Implemented file system and terminal tool handlers (read/write/diff, shell execution, browser automation) with permission-gating and diff-preview UI, enabling safe autonomous code edits within the editor.
- Optimized context management for large codebases by building token-aware truncation, sliding-window conversation history, and workspace file indexing to keep the agent within model context limits.
- Contributed to the extension's build and packaging pipeline (esbuild, VS Code Marketplace publishing) and added telemetry/error-tracking to monitor tool-call success rates and reduce agent failure loops in production.

PROJECTS

Cobraine - Telegram-Based Second-Brain AI Agent (Personal Project)
Tech Stack: Node.js, PostgreSQL, pgvector, Redis, BullMQ, Ollama (Nomic Embeddings), RAG

- Designed and built a backend service ingesting heterogeneous real-time inputs (notes, reminders, links, PDFs, images) via a Telegram webhook, processed asynchronously through BullMQ job queues, mirroring high-throughput event ingestion pipeline patterns.
- Implemented semantic search using a RAG pipeline with VoyageAI embeddings and pgvector over PostgreSQL, including hand-written SQL for schema and query design (no ORM, by deliberate choice) - reflecting deep, hands-on database design and query optimization experience.
- Used Redis for caching and coordinating async job state across the ingestion pipeline.
- Implemented token-based authentication via OAuth for using Google Workspace MCP server.
- Deployed and managed the application on an AWS EC2 instance, using Docker Compose for PostgreSQL and Redis, and PM2 to supervise the Node.js application and background worker processes.
- Architected the system with clear service boundaries between ingestion, embedding, storage, and retrieval layers, designed for reliability and extensibility.

Google Workspace MCP server
Tech Stack: Node.js, TypeScript, Model Context Protocol (MCP), Google Workspace APIs, OAuth 2.0, HTTP/REST

- Built a stateless, multi-tenant Model Context Protocol (MCP) server exposing 45+ Google Workspace actions (Drive, Gmail, Calendar, Docs) for AI agents.
- Migrated the standard MCP architecture from local stdio to a centralized HTTP service to support thousands of concurrent users in a Telegram bot.
- Engineered a connection-level authentication system that intercepts OAuth tokens from HTTP headers to spin up ephemeral, per-request API clients without tying the service to a specific database schema.

EDUCATION

MCA 2025 – Present
Chandigarh University Chandigarh, India

BA English 2018 – 2022
Delhi University Delhi, India

ACHIEVEMENTS AND EXTRAS

: Cleared **JEE Advanced 2017** and secured admission to IIT BHU strong quantitative, analytical, and problem-solving foundation

: Mountaineering and allied adventure sports

: Music/ Literature