⚙️Automation with Ag — AFFiNE-Powered Micro-Trading MVP

> **Production-Ready** micro-SaaS: AFFiNE + TypeScript + Next.js + OpenRouter + Supabase + Ollama + Binance API

Project Context

Transform AFFiNE's real-time workspace into an **agentic trading & devops hub**:

- **AFFiNE UI**: Custom panels (`/trading-agent`, `/dev-bot`) embedded in the editor.
- **Agent Core**: LangChain AgentExecutor → self-hosted OpenRouter → Ollama LLM.
- **Trading Engine**: Binance Spot API for dip-buy & gain-sell logic.
- **Auth & Data**: Supabase (Auth, RLS, Edge Functions, Postgres).
- **Monetization**: Stripe tiered plans with webhook-driven access control.
- **Infra**: Docker Compose local dev; deploy to Fly.io, GCP Cloud Run, or AWS ECS/Fargate.
- **Security & Monitoring**: Sentry, Snyk, Helmet.js, Zod, Prisma, API gateway.

🗜️High-Level Architecture

```
flowchart LR
  subgraph UI
    A[AFFiNE Editor] -->|REST/Socket| B[Agent API]
    A -->|Webhooks| C[Bot Interfaces]
  end

  subgraph AgentLayer
    B --> D[LangChain AgentExecutor]
    D --> E[Ollama LLM
(local Docker)]
    D --> F[OpenRouter API]
    E -->|Response| D
    F -->|Fallback| D
  end

  subgraph Bots
    C --> G[Telegram Bot]
    C --> H[Discord Bot]
  end

  subgraph Infra
    B --> I[Supabase
(RLS, Edge Fn, Postgres)]
    B --> J[Stripe
(Webhooks)]
    B --> K[Logging & Monitoring]
  end

  style UI fill:#f0f9ff,stroke:#3b82f6
  style AgentLayer fill:#f0fff4,stroke:#10b981
  style Bots fill:#fffbeb,stroke:#f59e0b
  style Infra fill:#fff1f2,stroke:#ef4444
```

🔗Core Features

Category	Feature
UI	AFFiNE panels for strategy config & logs
Trading Agent	LangChain + Ollama auto buy/sell
Code Review	Gemini CLI via OpenRouter for inline fixes
Bots	Telegram & Discord triggers & notifications
Auth & DB	Supabase RLS, sessions, agent & user configs
Payments	Stripe checkout & webhook-led tier gating
Infra	Docker (local), Fly.io/GCP/AWS (prod)
Security	Zod & Prisma validation, Helmet.js, rate limits

🛠️Tech Stack

```
graph TD
  subgraph Frontend
    UI[AFFiNE (Next.js + Tailwind)]
  end
  subgraph Agent
    AC[Agent API (Node.js)]
    LC[LangChain Executor]
    OL[Ollama LLM]
    OR[OpenRouter]
  end
  subgraph Services
    SB[Supabase]
    ST[Stripe]
    BB[Binance API]
    SN[Sentry & Snyk]
  end
  subgraph Deploy
    DC[Docker Compose]
    FO[Fly.io / GCP / AWS]
  end
  UI --> AC --> LC --> OL
  LC --> OR
  AC --> SB
  AC --> ST
  AC --> BB
  AC --> SN
  DC --> FO
```

🔐Security Measures

Application

- Helmet.js for HTTP headers
- Strict CORS & CSRF tokens
- Input sanitization & XSS filtering

Agent Layer

- Zod schemas to validate prompts & responses
- LangChain guardrails against injection
- Rate limiting on tool usage

Database

- Supabase Row-Level Security policies
- Prisma ORM with parameterized queries
- Audit logs via Edge Functions

API & Infra

- API gateway: key validation, IP allowlists
- Docker secrets, non-root containers
- Snyk vulnerability scans

📠docker-compose.yml

```
version: '3.8'
services:
  affine:
    build: ./apps/affine
    ports:
      - 3000:3000
    env_file: .env

  supabase:
    image: supabase/postgres:latest
    ports:
      - 54321:5432
    volumes:
      - supabase-data:/var/lib/postgresql/data

  ollama:
    image: ollama/ollama
    ports:
      - 11434:11434

  openrouter:
    image: openrouterai/openrouter
    ports:
      - 3001:3000
    env_file: .env

  agent-runner:
    build: ./apps/agent-runner
    ports:
      - 4000:4000
    depends_on:
      - supabase
      - ollama
      - openrouter

volumes:
  supabase-data:
```

🧰Key Snippets

```
import { z } from 'zod';

export const TradeSchema = z.object({
  symbol: z.string().min(3),
  action: z.enum(['BUY','SELL']),
  amount: z.number().positive().max(1000),
});
```

```
model User {
  id         String   @id @default(uuid())
  email      String   @unique
  tier       String
  logs       Log[]
}

model Log {
  id        String   @id @default(uuid())
  userId    String
  action    String
  result    Json
  createdAt DateTime @default(now())
  User      User     @relation(fields: [userId], references: [id])
}
```

💷Demo & Testing

1. **Local Prompt Test**\ `ts-node apps/agent-runner/demo.ts`

2. **Supabase Logs**\ View `agent_logs` table for telemetry.

3. **Bot Trigger**\ Send `/trade BTCUSDT` in Telegram → inspect logs.

📈Next Moves

- **LangSmith** for eval metrics
- **AFFiNE Plugin** surface inline Gemini feedback
- **CI/CD** across Fly.io, GCP, AWS with GitHub Actions

Clean. Modern. Mission-ready. 🚀