AgentPM Web Application – How It Works

Overview

AgentPM is a full-stack web application that simulates the key responsibilities of a Product Manager (PM) using AI agents. Each agent is responsible for a specific PM function—from hiring and planning to launch and feedback analysis. The system supports both manual interactions via a frontend UI and automated coordination via a backend orchestration engine.

System Architecture

- **Frontend**: React (Vite) + TailwindCSS
- **Backend**: FastAPI + Python services
- **AI Layer**: OpenAI GPT-4o + FAISS for semantic matching
- **Agent Framework**: Modular REST agents with planned CrewAI orchestration
- **Data Storage**: PostgreSQL, mock data, and vector DB (FAISS)
- **Deployment**: Dockerized via `docker-compose`

Core Functional Modules (Agents)

1. Talent Acquisition Agent

- **Input**: Job Description, Candidate Resumes
- **Function**: Parses resumes, matches against JD, ranks candidates, schedules interviews.
- **Tools Used**: Embeddings, FAISS, Google Calendar (planned)

2. Roadmap Planning Agent

- **Input**: User feedback, Competitor benchmarks
- **Function**: Clusters feedback, uses RICE scoring, builds a 3-month roadmap.
- **Tools Used**: LLMs, prompt-based logic, Trello/Jira APIs (planned)

3. Progress Monitoring Agent

- **Input**: Jira/Trello task data, Roadmap status
- **Function**: Tracks execution, detects blockers, generates weekly reports.
- **Tools Used**: Recharts/Streamlit, Slack API (planned)

4. GTM Strategy Agent

- **Input**: Feature list, User personas
- **Function**: Generates marketing content, schedules campaign posts.
- **Tools Used**: GPT-4o, Google Trends, Mailchimp/Hootsuite (planned)

5. Sales & Feedback Agent

- **Input**: Analytics, Reviews, CRM data

- **Function**: Analyzes usage & feedback, suggests roadmap updates.
- **Tools Used**: Sentiment analysis, Typeform, Zendesk, Stripe

---

How the Web Application Works

Frontend

- Users navigate through pages like `Talent`, `Roadmap`, `Progress`, etc.
- Each page is linked to a backend endpoint using `axios.js`.
- Upload forms allow JD/resumes, feedback, and other data to be submitted.
- Dashboards display results such as ranked candidates or roadmap kanban.

Backend

- FastAPI routes for each agent handle API requests.
- Each agent is a service module that uses LLMs, tools, or APIs to generate output.
- Mock data is used for testing, with test cases defined in the `tests/` folder.

Orchestration (Planned)

- A centralized orchestration script (`orchestrator.py`) is scaffolded using CrewAI.
- This script runs all agents in sequence for full lifecycle simulation.

Example Use Case Flow

1. Upload JD and resumes → Talent Agent ranks candidates.
2. Upload feedback CSV → Roadmap Agent clusters and creates a plan.
3. Monitor progress via synced sprints → Progress Agent reports blockers.
4. Launch content auto-generated → GTM Agent creates campaign.
5. After launch, feedback collected → Sales & Feedback Agent suggests improvements.

Key Features

- Multi-agent system, each with its own logic and API
- Clean frontend for interaction and uploads
- Dockerized deployment
- Full test coverage with `pytest`
- Extensible orchestration support with CrewAI

Next Steps

- Hook into real-time APIs (Jira, Stripe, Google Calendar)
- Expand CrewAI agent memory and workflow logic
- Deploy to cloud (e.g., Render, Railway, Vercel)

Authors

Developed by [Your Name] as part of a multi-agent AI product management simulation project.

For more detailed architecture or to contribute, visit the README or orchestrator scripts in the `/backend/app/orchestration/` directory.