Product Specification & Commercial Brief: AI Sidecar Command Agent

# Document Overview

Document Version: 1.0

Author: RickPublic / ChatGPT

Date: August 2025

Purpose: This document defines the requirements, architecture, and business case for a Command Agent that interprets AI-issued instructions and performs local file and Git operations.

# 1. Introduction

AI tools like ChatGPT lack persistent memory and file access. This tool enables local execution of structured AI-generated commands, acting as a 'sidecar brain' for repo and content governance.

# 2. Functional Requirements

- Parse [COMMAND] instructions from ChatGPT.

- Execute file and Git operations securely.

- Track file lineage and maintain logs.

- Operate in a sandboxed directory environment.

- Allow manual override, dry-run preview, and rollback logs.

# 3. Non-Functional Requirements

- Cross-platform compatibility (Linux, macOS, Windows).

- Minimal dependencies for the MVP (Python-based).

- Configurable command mapping and safety modes.

- Expandable through plugins or GUI in later phases.

# 4. System Architecture

Component 1: Command Parser

Component 2: Shell/Git Executor

Component 3: Audit Logger

Component 4: Metadata Tracker (JSON-based)

Optional Component 5: Feedback Agent to AI

# 5. Setup Instructions

1. Install Git and Python 3.8+.

2. Create sidecar folder structure: /incoming, /staging, /archive, /registry.

3. Run script to parse and execute [COMMAND]s.

4. Review logs and state file after each batch.

# 6. Time and Cost Estimates

- MVP: 10–15 hours (core file + Git ops).

- Logging/dry-run: +4–6 hours.

- Feedback loop or browser overlay: +10–15 hours.

- Optional GUI: +20–40 hours.

Total Estimated Cost: $1,500–3,000 USD.

# 7. Commercial Viability

User Market: AI creators, Substackers, Git maintainers, civic coders.

Revenue Models:

- Free version on GitHub (basic features).

- Paid version with GUI, GitHub integration, GPT plugin support.

- Tiered pricing (e.g., $200–500/user).

Projected Revenue:

- 500 users at $200 = $100,000

- Add-on modules could increase LTV.

# 8. Marketing Strategy

Primary Channels: Hacker News, Product Hunt, Reddit, AI forums.

Video Content: Screencasts showing AI directing local files.

SEO: Keywords like 'give ChatGPT memory', 'AI file tools', 'AI-powered Git'.

Launch Strategy: Open-source drop + optional paid pro tier.

# 9. Branding Concepts

Name Ideas:

- AISA (AI Sidecar Agent)

- GhostHand

- PromptPilot

- CmdBridge

- EchoDO

Logo & UX Style: Minimalist, retro-terminal aesthetic with clear icons.

Placeholder: [Insert logo concept sketch here]

# 10. UI Mockups (Placeholder)

[Mockup: Command Panel showing parsed command queue, preview toggle, live Git status]

[Mockup: File history viewer with version tree]

[Mockup: Main dashboard for AI-directed session playback]

# 11. Financial Scenarios

Open Source: 0 revenue (adoption metric only)

MVP Sales (200 users x $200): $40,000

Pro Add-ons (50 users x $500): $25,000

Enterprise Licenses (5 clients x $2,000): $10,000

Total Addressable Revenue: $75,000+ in Phase 1

# 12. Competitive Landscape

- Notion AI: No file ops or Git integration.

- LangChain: Dev-centric, complex to configure.

- Copilot: IDE-bound, not command-structured.

- No direct competitor offering structured AI-to-local-agent orchestration.

# 13. Future Expansion Opportunities

- Native ChatGPT plugin version.

- Mobile/voice assistant integration.

- Custom workflows: Substack prep, repo restructuring, content governance.

- Smart archive management based on AI-observed usage patterns.

# 14. Call to Action

Looking for collaborators:

- Python/Node developer for MVP.

- UI/UX designer for mockups and user flow.

- Early testers and marketers familiar with AI productivity pain points.

AI Sidecar Command Agent

Product Specification & Commercial Prospectus  
Version 1.1 – August 2025

# Executive Summary

This proposal outlines the AI Sidecar Command Agent (AISA), a lightweight, local tool that allows AI assistants like ChatGPT to perform structured file and Git operations through interpreted command blocks. The MVP leverages Python to securely manage content and repository workflows, offering an expandable platform for AI-human collaboration in knowledge curation, coding, and publishing.

# 2. Functional Requirements (Expanded)

|  |  |  |
| --- | --- | --- |
| Requirement | Status | Description |
| Parse [COMMAND] blocks | MVP | Extract structured directives embedded in plain text or .md files. |
| Execute file and Git ops | MVP | Run file operations (create/move/edit/delete) and Git commits. |
| Track lineage/logs | MVP | Maintain changelogs for each operation set. |
| Sandboxed directories | MVP | Ensure all actions remain within a known local structure. |
| Dry-run & rollback | MVP | Simulate commands before execution; enable undo logs. |
| Manual override | MVP | Allow users to approve or modify generated actions. |

# 4. System Architecture

[DIAGRAM PLACEHOLDER: Component diagram showing Command Parser, Shell/Git Executor, Audit Logger, Metadata Tracker, Feedback Agent]

# Original content sections and improvements continue below: