PromptDojo Operator Setup

This document serves as the central blueprint for building, maintaining, and scaling **PromptDojo** as an Operator-native GPT product.

🧠 PRODUCT OVERVIEW

PromptDojo is a GPT-powered, dojo-style training experience for AI evaluation jobs. It is structured as a 7-day interactive curriculum with slash commands, XP/rank system, and a PromptPass certification at the end.

Mission: Train beginners into prompt evaluation pros through guided drills, live simulations, and feedback — no prior experience needed.

Current Version: v5.2.2\ **Planned Next:** v5.3 Operator Launch

---

👈🏼 FILE STRUCTURE (TO BE CREATED IN GITHUB)

```
📁 promptdojo-operator
├── README.md
├── /config
│   └── promptdojo_v5.3.json
├── /docs
│   ├── operator_blueprint.md
│   ├── prompts_and_rubric.md
│   ├── gpt_setup_guide.md
│   └── changelog.md
└── /assets
    ├── logo.png
    ├── badge_promptpass.png
    └── banner_twitter.png
```

👈🏼 WHAT THIS SYSTEM FIXES

- Solves **GPT memory reset issues** by using persistent `.md` files as live role definitions
- Each assistant (Signal, Forge, Marketing) reads raw GitHub to know its duties, without needing reprompting
- Roles like `Product Coach`, `Dev`, `Marketing` can be created in new chats instantly by referencing their GitHub doc

---

👈🏽OPERATOR BLUEPRINT

Core Commands:

- `/start` – Kicks off training
- `/rubric` – Explains 5-part grading system
- `/day1` – `/day7` – Structured content pacing
- `/outlier` – Real evaluator-style tasks
- `/promptpass` – Final cert challenge (5 prompts)
- `/rank` – Tracks XP, dojo level, rank title
- `/support` – Donation prompt via Ko-fi
- `/roadmap` – Shows future features & use of donations
- `/sensei` – Coaching hub for retry/reflection
- `/shinobi` – Optional paywalled Easter egg mode

Certification Logic:

- PromptPass = 5 graded responses in a row > 20/25
- Generates PDF certificate customized to user

XP/Ranks:

- 5 XP per drill
- Ranks: Student → Apprentice → Evaluator → Shinobi

😇WIRED FEATURES

- GitHub `.md` structure → Operator memory & logic
- JSON config contains all logic/state
- Ko-fi integration for `/support`
- AI adapts feedback and tone based on user history

👋🏼GITHUB INTEGRATION TODO

-

😟NEXT ACTIONS

1. **Create GitHub repo + file structure**
2. **Push current .json config and docs**
3. **Move GPT setup to clean new chat**
4. **Update slash commands to match Operator logic**
5. **Lock in v5.3 JSON and launch officially**

Once that’s done, you’ll have:

- A full Operator-native, monetizable product
- Persistent memory and slash logic
- Public roadmap, ranks, coaching system, and donation loop