Developer Notes Repository Guide

GOAL
Create and manage a Git repository to store your notes, PDFs, and project documentation.
This repo will act as your personal archive for learning, reference, and version history.

README TEMPLATE
Copy this text into a file named README.md inside your dev-notes folder.

Brock's Dev Notes

A personal library of learning materials, project plans, and AI development logs.
This repository tracks progress across all learning phases — from Git fundamentals to building a local voice

■ Current Guides
File	Description
Git_CheatSheet_Brock.pdf	Core Git commands and workflow for version control.
Tailscale_RunPod_Workflow_Brock.pdf	Secure networking setup for Mac and RunPod integration.
Next_Steps_DiscordBot_RunPod.pdf	Step-by-step plan for connecting and deploying the Discord bot.
Local_Buddy_Roadmap_Brock.pdf	Long-term roadmap for the local AI buddy with voice, vision, and TTS.

■■ Folder Layout
dev-notes/
■■■ 01_Git_CheatSheet_Brock.pdf
■■■ 02_Tailscale_RunPod_Workflow_Brock.pdf
■■■ 03_Next_Steps_DiscordBot_RunPod.pdf
■■■ 04_Local_Buddy_Roadmap_Brock.pdf
■■■ daily_logs/
■   ■■■ YYYY-MM-DD.md
■   ■■■ ...
■■■ ideas/
    ■■■ feature-list.md
    ■■■ roadmap.md
    ■■■ questions.md

■ Purpose
This repo serves as a personal development archive — a record of learning how to:
- Build and deploy AI-driven bots and tools
- Securely connect local and remote systems
- Manage projects with Git and Docker
- Incrementally evolve from simple scripts to a full macOS “buddy” app

■ Update Routine
git add -A
git commit -m "update: new notes and progress"
git push

Keep updates small and frequent for easier version tracking.

TERMINAL COMMANDS

1. Navigate into your repo
   cd ~/Projects/dev-notes

2. Add your README and any PDFs
   git add -A

3. Commit the changes
   git commit -m "docs: added README and learning PDFs"

4. Push to GitHub
   git push

After pushing, refresh your GitHub page — the README will appear automatically as the repo’s description.

ENHANCEMENTS AND BEST PRACTICES
1. AUTOMATION

```
   Create a small update script called update.sh:
   #!/bin/zsh
   git add -A
   git commit -m "auto: daily update $(date '+%Y-%m-%d')"
   git push
   Make it executable: chmod +x update.sh

2. TAGGING PROGRESS
   Use Git tags to mark key milestones:
   git tag -a v1.0 -m "Initial note setup"
   git push origin v1.0

3. DAILY LOGS
   Keep a folder /daily_logs for Markdown notes:
   2025-10-18.md
   # Daily Progress
   - Completed Tailscale + Git setup
   - Added README to notes repo
   - Next: Connect RunPod to tailnet

4. BRANCHING FOR EXPERIMENTS
   Use feature branches for new note sections:
   git checkout -b notes-voicebuddy
   (write, commit, push, merge later)

5. BACKUPS
   - Keep repo synced with GitHub (push after each session).
   - Optional local backup with Time Machine or rsync to external drive.

6. OPTIONAL ENHANCEMENTS
   - Add a .gitignore to exclude junk files:
       .DS_Store
       __pycache__/
       *.log
   - Use Markdown headings consistently (#, ##, ###)
   - Add screenshots (PNG/JPEG) to illustrate steps or configs
   - Keep one PDF per major lesson for clarity

7. GROWTH PLAN
   - Phase 1: Use repo for notes & PDFs (done)
   - Phase 2: Add project readmes and code snippets
   - Phase 3: Add local scripts for automation (update.sh, cleanup.sh)
   - Phase 4: Integrate your AI learning logs and model configs

---

INSTRUCTOR NOTE
You now have a permanent knowledge repository — a system that grows as you do.
This habit transforms your learning into an organized, verifiable timeline.
Your future self will thank you when you can revisit old code, notes, and ideas with context.
```