LibreChat Augmented

Organization: Dogpatch Labs
Supervisor: Patrick Curran/Gleb Sapunenko
Duration: 2 Weeks

Objective

Install, configure, and deploy **LibreChat**, a self-hosted ChatGPT alternative. Ensure it is fully operational with **SSO (Single Sign-On)**, **PWA/mobile app functionality**, and **augmented features**. Explore further functionality and how it can be expanded beyond a standard LLM chat interface by looking at MCP workflows and RAG across other data flows. Propose name & logo for our internal chat assistant.

Scope of Work

1. **LibreChat Documentation Review**
    - https://www.librechat.ai/docs > Features
    - Q: What are some interesting features that could be useful for the Dogpatch Labs team? Or the Dogpatch coworking community?
    - Look into other possible capabilities of LibreChat:
        i. Can we integrate with Slack to make easily accessible to the community?
        ii. How can we integrate with other tools we use as data sources?
            1. E.g. Can we integrate with Gmail to pull emails?
            2. Can we integrate with Hubspot and get it to search contacts?
        iii. How can we use it to trigger agent workflows that may be interesting?

2. **Setup & Installation**
    - Install LibreChat on Dogpatch Labs hosting environment via Docker (See Appendix 1)
    - Use provided **OpenAI API Key** for LLM functionality. (See Appendix 1)
    - Use provided **SSH Hosting Key** for access. (See Appendix 1)
    - Configure environment, dependencies, and security best practices.

3. **Configuration**
    - Enable and test **SSO login (Google/Microsoft preferred)**.
    - Configure environment variables and integrations as required.
    - Ensure admin dashboard access.

4. **Testing**
    - Test **core chat features** (history, prompts, formatting, conversations).
    - Test **mobile/PWA experience** (installable on iOS/Android, notifications, offline support). Explore and document **extended features** (plugins, multi-model support, sharing, collaboration, pipelines, bots, etc).

5. **Expansion**
    - Model System-Prompt self-improvement
        i. Implement bot/agent/tool that can self improve the system prompt. [see Appendix 2]
    - Explore further features of LibreChat, for example:
        i. How can we integrate with data sources (e.g. Google Drive) and utilise RAG to obtain more relevant answers?
        ii. How can we use MCP to implement agent workflows across our different software tools
    - Feel free to use your imagination on capabilities!

6. **Branding**
    - Create name, icon
    - Improve & customise CSS/UX as required

Expected Deliverables

- Fully functioning **LibreChat instance** deployed and accessible.
- **SSO-enabled login flow** tested and documented.
- **Mobile/PWA experience report** with screenshots and feedback.
- **Feature exploration document** highlighting key findings.
- **Name proposals** with reasoning behind each suggestion.
- **Final presentation/demo** of the deployed system.

Check-ins & Support

- **Day 1**: Kickoff call (review goals, clarify setup).
- **Midpoint (Day 3)**: Progress review
- **Final Day (Day 5)**: Demo session & deliverables handover.
- Slack channel for ongoing Q&A and troubleshooting.

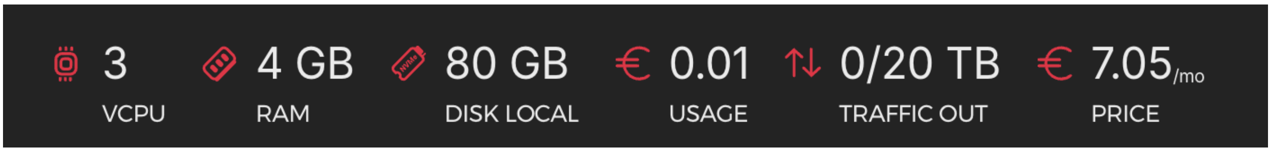
Outcome

By the end of this task, Dogpatch Labs will have:
✅ A deployed **LibreChat** instance with SSO and mobile/PWA capabilities.
✅ Clear documentation for setup, usage, and features.
✅ A shortlist of potential names for our internal chatbot.

Appendix 1: Technical Details

Hosting Hetzner

Hosting is through Hetzner. It can be accessed using the SSH details below:

Specs	 3 VCPU 4 GB RAM 80 GB DISK LOCAL € 0.01 USAGE 0/20 TB TRAFFIC OUT € 7.05/mo PRICE
IP	95.217.131.11

Hetzner SSH Key:

Private Key	`-----BEGIN OPENSSH PRIVATE KEY-----`
	`b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAAAMwAAAAtzc2gtZW`
	`QyNTUxOQAAACDPRuTWlyU1WBktX6ntiZ3Pvp80UY9POnFXKfL2E0485QAAAJBeIT2zXiE9`
	`swAAAAtzc2gtZWQyNTUxOQAAACDPRuTWlyU1WBktX6ntiZ3Pvp80UY9POnFXKfL2E0485Q`
	`AAAED217iBO1/11e9uBg3eQnfHXsNS/ogOu0ns/fSzYivy/s9G5NaXJTVYGS1fqe2Jnc++`
	`nzRRj086cVcp8vYTTjzlAAAAB0lTRTIwMjUBAgMEBQY=`
	`-----END OPENSSH PRIVATE KEY-----`

Public Key	`ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIM9G5NaXJTVYGS1fqe2Jnc++nzRRj086cVcp8vYTTjzl`
	`ISE2025`

OpenAI API Key:

```
sk-proj-zY9Ldr0_VorHStkJc56nRmqlqaFBFF0A1_R-A1R71F508OMmnOmb2A3AiaYTKzzgAPQMh
Cp1uST3BlbkFJvKwoSvMe0P2S7YklK0FpSgHZfku8fevEX34POIzMGlGzvLC5ocVtM5QSnMTzDmMO
Ro2pfijCoA
```

Appendix 2: SystemPrompt Self-Improvement

Implement bot/agent/tool that can self improve its own system prompt (or another bot's)

Usage:

1. Use a model as usual
2. Tag the tool/bot saying what needs to be improved in its prompt (e.g. add this routing table)
3. Bot takes input, reviews existing prompt, drafts new system prompt and asks user if that's better
4. If user confirms, bot updates prompt and adds to change log the changes it made. Confirms to user the update success.

Example:

1. User: who do i ask for annual leave?
2. Model: this is beyond my knowledge.
3. User: I've asked HR and they told me I need to ask on the Personio platform @improvebot
4. Feel free to get creative with how to implement this in practice.
5. Consider other possible methods of automatic prompt improving or engineering.