






AMA 12 | インストールと環境構築のワークフロー

目的

Aéthaプロジェクトの一環として、AMA (Autonomous Memory Archive) システムをローカルで動作させるための初期セットアップ手順と環境構築ワークフローを明示する。すべての手順は手動確認後、Canvasに記録・更新しながら進行する。

1. 必要条件（前提環境）

-  OS : macOS（最新版推奨）
-  Python 3.10以降（Anacondaでも可）
-  Git（CLI操作可能な状態）
-  VSCode（または任意のエディタ）
-  ターミナル操作に慣れていること

2. 使用サービスと連携候補

サービス	目的	無料枠	今後の連携備考
Notion API	記憶ログの保存・可視化	○（要登録）	手動ログ出力との連携
LangChain	記憶管理チェーン・プロンプト制御	○（OSS）	FAISSとの併用可能
Chroma / FAISS	ローカル向けのVector DB	○	Pinecone（有料）との選択式
Google Sheets	JSON↔CSV連携	○	手動データバックアップ向け
GitHub	管理・アーカイブ	○	構成ファイルと統合インデックス保存先に使用

3. 初期セットアップ（ステップ別）

フォルダ構成の確認

- `ama-system/` および `eme-system/` ディレクトリを、アカウント内の正しい構成で設置済みか確認
- `scripts/`, `config/`, `prompts/`, `memory/`, `journal/` 各サブディレクトリの存在確認

Python環境構築

```
python3 -m venv venv
source venv/bin/activate
pip install -r requirements.txt
```

(※ requirements.txt は Canvas12 で作成予定)

APIキー等の環境変数設定

- `.env` ファイルに下記を記述（使用予定に応じて）

```
OPENAI_API_KEY=sk-xxx...
NOTION_API_KEY=...
NOTION_DATABASE_ID=...
```

4. 今後の連携・動作テスト


- Canvas12 にて：`test_prompt.py`, `test_store.py` などのスクリプトと動作検証手順を記述
- Canvas13 以降で：記憶書き込み／読み込み／日記更新フローを段階的に実装・テスト

5. 補足事項（開発ナビゲーション）

- 今はあくまでセットアップ準備フェーズ：実際の操作はCanvas構築完了後にタケが主導で進行
- Canvasに記録されるすべての内容は「初心者でも一通り読めば実行できる構成」を目指す
- 各ステップで不明点があれば、必ず止まって確認を取りながら進む

次アクション

-

 今後、綺羅との統合に向けて `index-ama-*.md` や共有テンプレートの調整を行う。各プロンプトやスクリプトもこの流れに沿って順次実装。