Aktualisierter Plan: 6. Januar 2026

Aktueller Stand

■ *Abgeschlossen*

1. **Netzwerk-Problem: Root Cause gefunden**
• Problem identifiziert: `cfgNetworks()` löscht alle Connections
• Lösung erstellt: `FIX_NETWORK_IN_DATABASE.sh`
• Dokumentation: Status Report erstellt

2. **Lokale KI-Setup gestartet**
• Ollama installiert
• Modell heruntergeladen
• Dokumentation erstellt

■ *In Arbeit*

1. **Open WebUI Installation** - läuft noch
2. **Netzwerk-Test** - wartet auf Pi-Boot

■ *Geplant*

1. RAG Setup
2. Agenten erstellen
3. Fine-tuning

Sofortige nächste Schritte

Schritt 1: Netzwerk-Lösung testen (HOCH)

Ziel: Netzwerk-Verbindung funktioniert nach Boot

Aktionen:
1. SD-Karte in Mac einstecken
2. `FIX_NETWORK_IN_DATABASE.sh` ausführen
3. SD-Karte sicher auswerfen
4. SD-Karte in Pi einstecken
5. Pi booten
6. Verbindung testen:

```
# Vom Mac aus:
ping 192.168.10.2
ssh andre@192.168.10.2
```

Erfolgskriterien:
• ■ Ethernet-Verbindung funktioniert (192.168.10.2)
• ■ SSH-Verbindung funktioniert
• ■ WiFi verbindet sich automatisch (falls verfügbar)

Bei Problemen:

• Logs sammeln: `network-mode-manager.sh` Logs prüfen
• Datenbank prüfen: `sqlite3 /Volumes/rootfs/var/local/www/db/moode-sqlite3.db`
• NetworkManager prüfen: `ls -la /Volumes/rootfs/etc/NetworkManager/system-connections/`

Schritt 2: Open WebUI abschließen (HOCH)

Ziel: Lokale KI läuft und ist nutzbar

Aktionen:
1. Prüfe ob Docker läuft: `docker ps`
2. Prüfe ob Open WebUI läuft: `curl http://localhost:3000`
3. Falls nicht: `./setup-open-webui.sh` ausführen
4. Browser öffnen: `http://localhost:3000`
5. Account erstellen
6. Ersten Test-Chat durchführen

Erfolgskriterien:
• ■ Open WebUI läuft
• ■ Kann mit KI chatten
• ■ Modell `llama3.2:3b` ist verfügbar

Schritt 3: RAG Setup vorbereiten (MITTEL)

Ziel: KI kann Fragen zu deinem Projekt beantworten

Aktionen:
1. In Open WebUI: Settings → RAG aktivieren
2. Wichtige Dateien sammeln:

```
# Dokumentation
docs/*.md

# Wichtige Scripts
FIX_NETWORK_IN_DATABASE.sh
tools/build.sh
tools/fix.sh

# moOde Source (wichtigste Teile)
moode-source/www/inc/network.php
moode-source/usr/local/bin/network-mode-manager.sh
```

3. Dateien in Open WebUI hochladen
4. Erste projekt-spezifische Fragen testen:
• "Wie funktioniert die Netzwerk-Konfiguration?"
• "Was macht cfgNetworks()?"
• "Wie deploye ich Änderungen?"

Erfolgskriterien:
• ■ RAG aktiviert
• ■ Dateien hochgeladen
• ■ KI kann projekt-spezifische Fragen beantworten

Mittelfristige Ziele (Diese Woche)

1. Network Config Agent erstellen

Ziel: Agent der automatisch Netzwerk-Probleme findet und fixt

Features:
• Prüft Netzwerk-Konfiguration
• Findet Probleme
• Erstellt Fixes
• Wendet Fixes an (mit Backup)

Verwendung:
```
Agent: network-config-agent
Task: "Prüfe die Netzwerk-Konfiguration und fixe alle Probleme"
```

---

2. Code-Review Agent

Ziel: Automatische Code-Reviews

Features:
• Analysiert Code
• Findet Bugs
• Prüft Best Practices
• Erstellt Review-Report

---

3. Documentation Agent

Ziel: Automatische Dokumentation

Features:
• Analysiert Code
• Generiert README.md
• Fügt Inline-Kommentare hinzu
• Erstellt Usage-Beispiele

Langfristige Ziele (Nächste Wochen)

1. Fine-tuning auf Code-Stil

Ziel: KI generiert Code in deinem Stil

Schritte:
1. Code-Beispiele sammeln
2. Trainingsdaten formatieren
3. Fine-tuning durchführen
4. Modell testen

---

2. Multi-Agent Systeme

Ziel: Mehrere Agenten arbeiten zusammen

Beispiel:

• Code Reviewer Agent
• Security Agent
• Performance Agent
• Documentation Agent
• Coordinator Agent

---

3. Deployment Automation

Ziel: Automatisches Deployment

Workflow:
1. Code-Review
2. Tests
3. Backup
4. Deployment
5. Verifizierung
6. Rollback bei Fehlern

Prioritäten-Matrix

Kritisch (Sofort)

• [ ] Netzwerk-Verbindung testen
• [ ] Open WebUI abschließen

Wichtig (Diese Woche)

• [ ] RAG Setup
• [ ] Ersten Agenten erstellen
• [ ] Code-Review Agent

Wünschenswert (Nächste Woche)

• [ ] Fine-tuning
• [ ] Multi-Agent Systeme
• [ ] Deployment Automation

Erfolgs-Metriken

Netzwerk

• ■ Verbindung funktioniert nach Boot
• ■ Keine manuellen Fixes nötig
• ■ Automatische Verbindung (Ethernet + WiFi)

Lokale KI

- ■ Open WebUI läuft
- ■ RAG funktioniert
- ■ Erster Agent erstellt
- ■ Agent kann Aufgaben erledigen

Code-Qualität

- ■ Automatische Code-Reviews
- ■ Dokumentation automatisch generiert
- ■ Bugs automatisch gefunden

Risiken & Mitigation

Risiko 1: Netzwerk-Lösung funktioniert nicht

Mitigation:
- Logs sammeln
- Datenbank prüfen
- Alternative Lösungen vorbereiten

Risiko 2: Open WebUI Installation schlägt fehl

Mitigation:
- Docker-Logs prüfen
- Alternative: LM Studio oder Jan.ai
- Manuelle Installation

Risiko 3: Agenten funktionieren nicht wie erwartet

Mitigation:
- Schrittweise vorgehen
- Einfache Agenten zuerst
- Tests nach jedem Schritt

Nächste Review

Wann: Nach Netzwerk-Test und Open WebUI Setup
Was prüfen:
- Netzwerk-Verbindung funktioniert?
- Open WebUI läuft?
- RAG Setup erfolgreich?
- Erster Agent erstellt?

Erstellt: 6. Januar 2026
Nächste Aktualisierung: Nach Netzwerk-Test