

MindGear – App für mentale Stärke & Fokus



Inspiration für mentale Stärke – mit kuratierten YouTube-Videos zu Fokus & Selbstentwicklung.

Projektziel & Kontext

Warum MindGear?

- Entstand während meiner Weiterbildung zum Mobile Developer
- Ziel: Männer in schwierigen Lebensphasen mental unterstützen

Zielgruppe

- Männer zwischen 18–60 Jahren
- Nutzen Podcasts & YouTube zur Selbstentwicklung
- Verlieren sich im digitalen Überangebot

Zielsetzung der App

- Native iOS-App mit kuratierten YouTube-Inhalten
- Offline-fähig, minimalistisch, thematisch fokussiert

Technischer Rahmen

- SwiftUI · MVVM · SwiftData · YouTube API
- Dark-Mode-freundlich
- Deutsch & Englisch
- Vollständig selbstentwickelt

Systemübersicht: Wie funktioniert MindGear?

Die App kombiniert kuratierte Inhalte mit einer einfachen, performanten Struktur – optimiert für Inspiration im Alltag.

Was steckt drin?

- Datenquelle: YouTube API (Playlists)
- Speicherung: SwiftData (Favoriten & Verlauf)
- Lokale Inhalte: Farben, Texte, Icons

Wie funktioniert's?

- MVVM-Struktur für Übersicht & Wartbarkeit (z.B. APIManager)
- Services wie APIManager zur Anbindung
- SwiftData Models zur Offline-Verwaltung

Was sieht der Nutzer?

-  Videoliste mit Suchfunktion (Text + Mentorname)
-  Mentor-Profilansicht mit Avatar, Bio, Videoübersicht
-  Favoriten-Funktion: Speicherung mit SwiftData + @AppStorage



Projektziel & Kontext

Die MindGear-App ist funktional fertiggestellt – mit stabiler Architektur, performanter Suche & intuitivem UI.
Die App ist bereit zur Präsentation im Portfolio.

Fertige Features

- Videoliste mit Suchfunktion
- Favoriten mit SwiftData
- Mentoren-Detailseiten mit Video & Beschreibung
- Fokus auf Klartext statt UI-Lärm

Architektur

- MVVM mit SwiftUI
- YouTube Data API (v3)
- REST-Anbindung an YouTube API
- Offline-Modell mit SwiftData

Projektziel erreicht

- Inspiration & mentale Unterstützung
- Fokussiert & Dark-Mode-freundlich
- Selbstständig entwickelt (iOS-only)

Verifizierte Technologien aus dem Projekt

- ✓ SwiftUI · MVVM · SwiftData
- ✓ YouTube Data API v3 · REST (GET only)
- ✓ Cloudinary + SDWebImageSwiftUI
- ✓ URLSession · JSON Fallback
- ✓ NetworkMonitor



Einblick ins UI – Video, Mentoren, Favoriten



Videoliste & Suche

- Übersicht aller Videos mit Titel, Bild & Dauer
- Textsuche nach Video & Mentorname
- Schnellzugriff auf neue Inhalte



Mentor-Profil

- Avatar + Kurz-Bio
- Liste aller zugehörigen Videos
- Fokus auf Klarheit & Persönlichkeit
- Mentor-Detailseite vermittelt Persönlichkeit und thematische Ausrichtung.



Favoriten-System

- Speicherung mit SwiftData + @AppStorage
- Herz-Button mit Bounce-Animation
- Dark Mode freundlich gestaltet



UserFlow & View-Struktur

🎉 User Story 1

„Als Nutzer möchte ich schnell ein inspirierendes Video zu einem Thema wie Fokus oder mentale Stärke finden, damit ich neue Denkanstöße bekomme.“

→ Flow:

- **HomeView** › **CategoryView** › **VideoListView** › **VideoDetailView**

🎉 User Story 2

„Als Nutzer möchte ich mehr über einen Mentor erfahren, um zu sehen, ob er zu meinem Stil passt.“

→ Flow:

- **HomeView** › **MentorDetailView** › **VideoListView (gefiltert)**

🎉 User Story 3

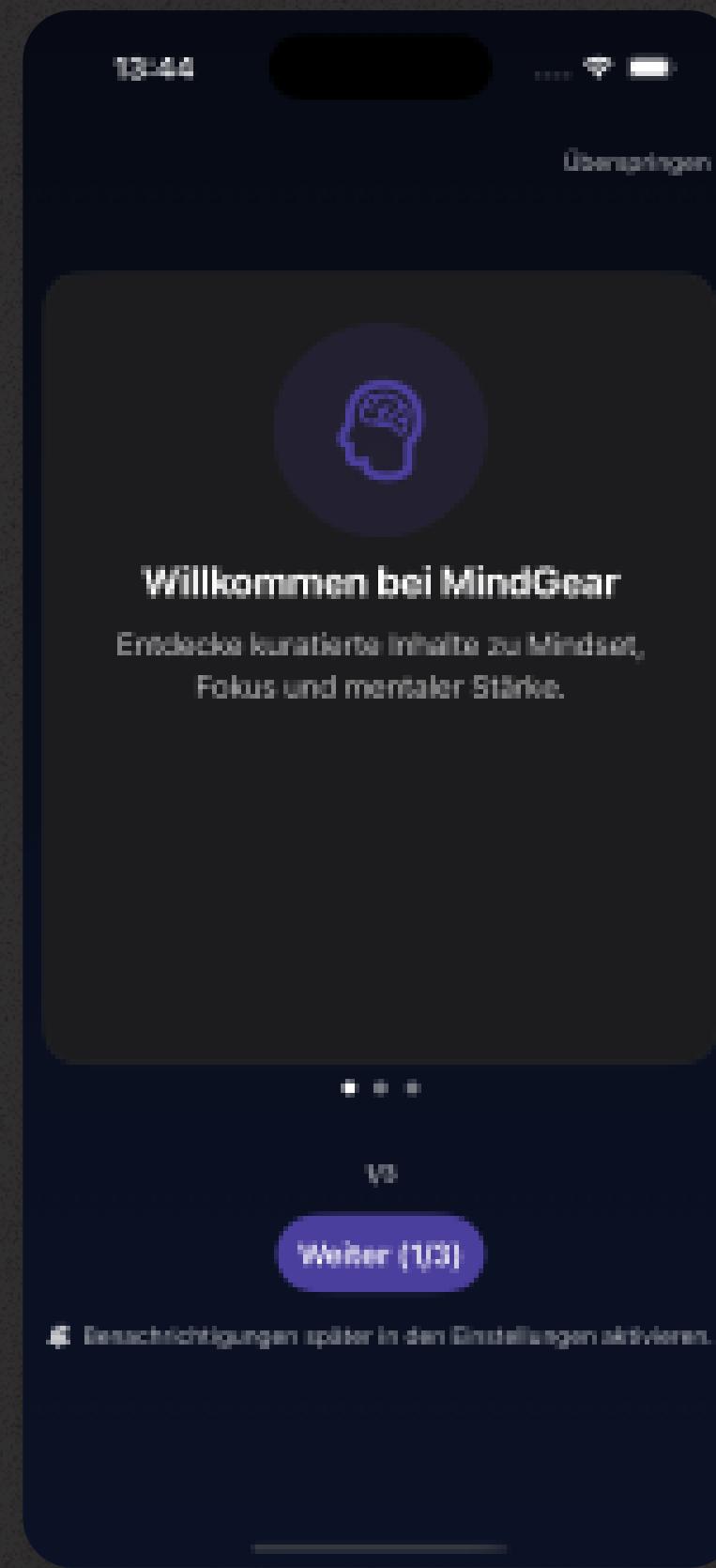
„Als Nutzer möchte ich Favoriten speichern, um inspirierende Videos später leicht wiederzufinden.“

→ Flow:

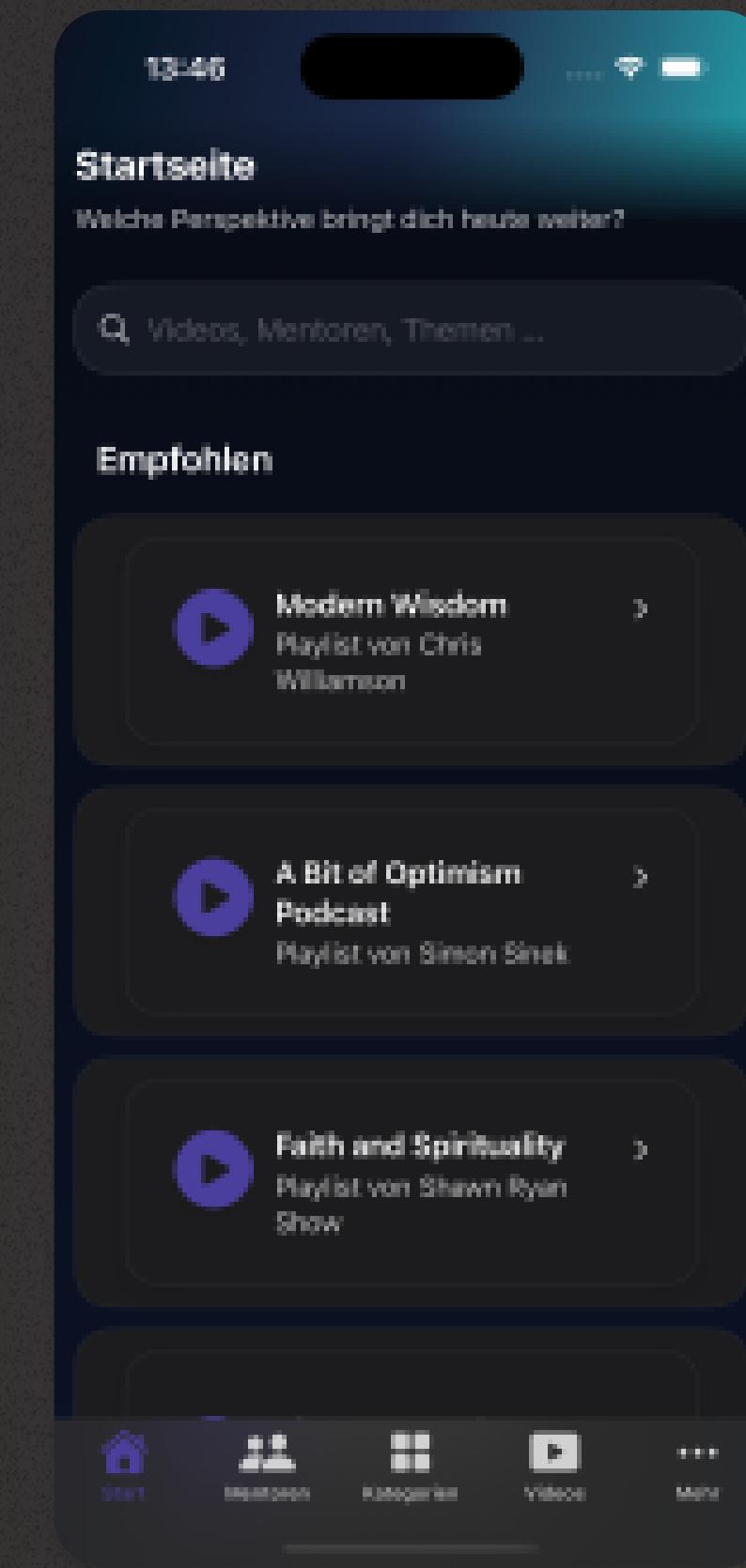
- **VideoDetailView** › ❤ **Favorite (Button)** › **FavoritesView**



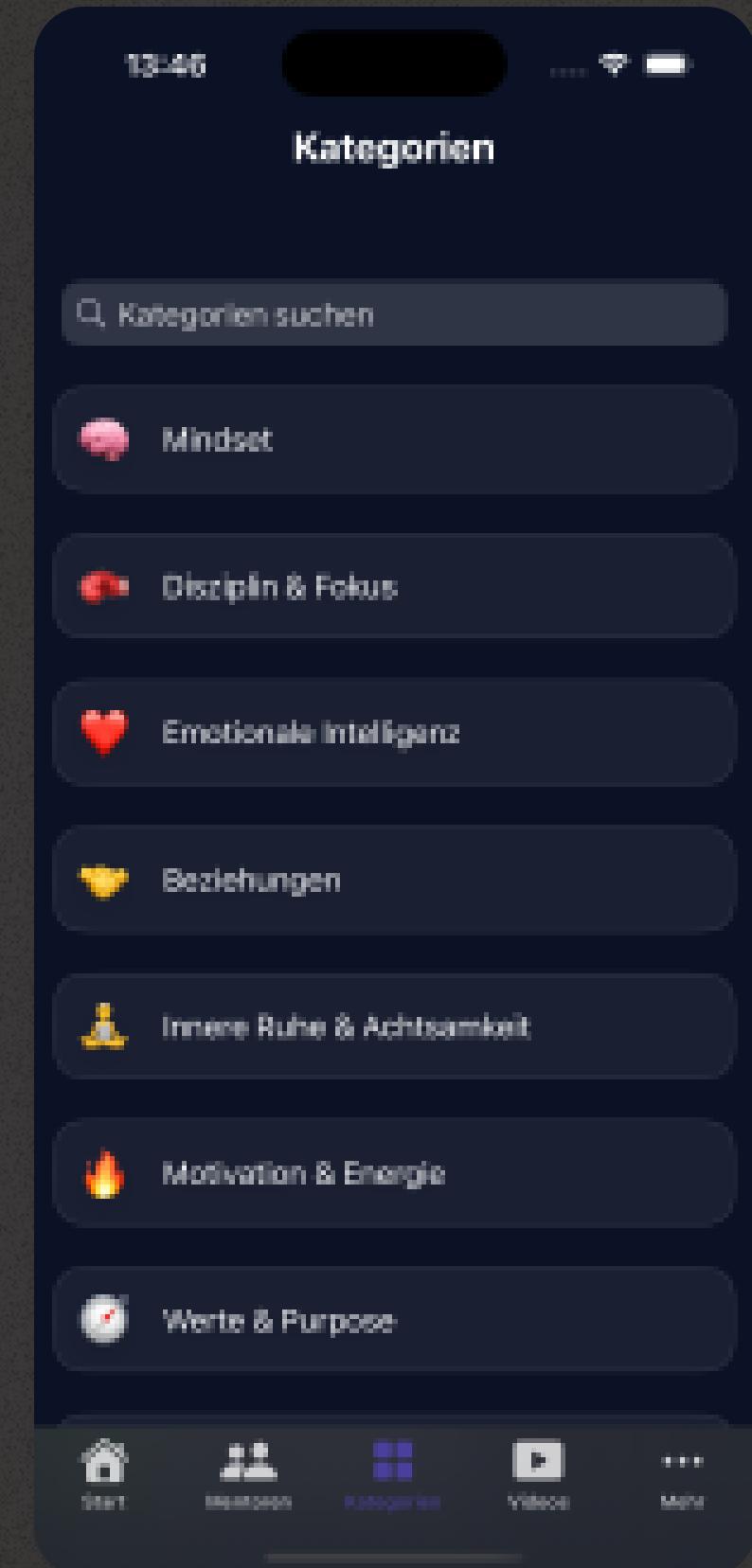
UI-Galerie – Screens aus der App



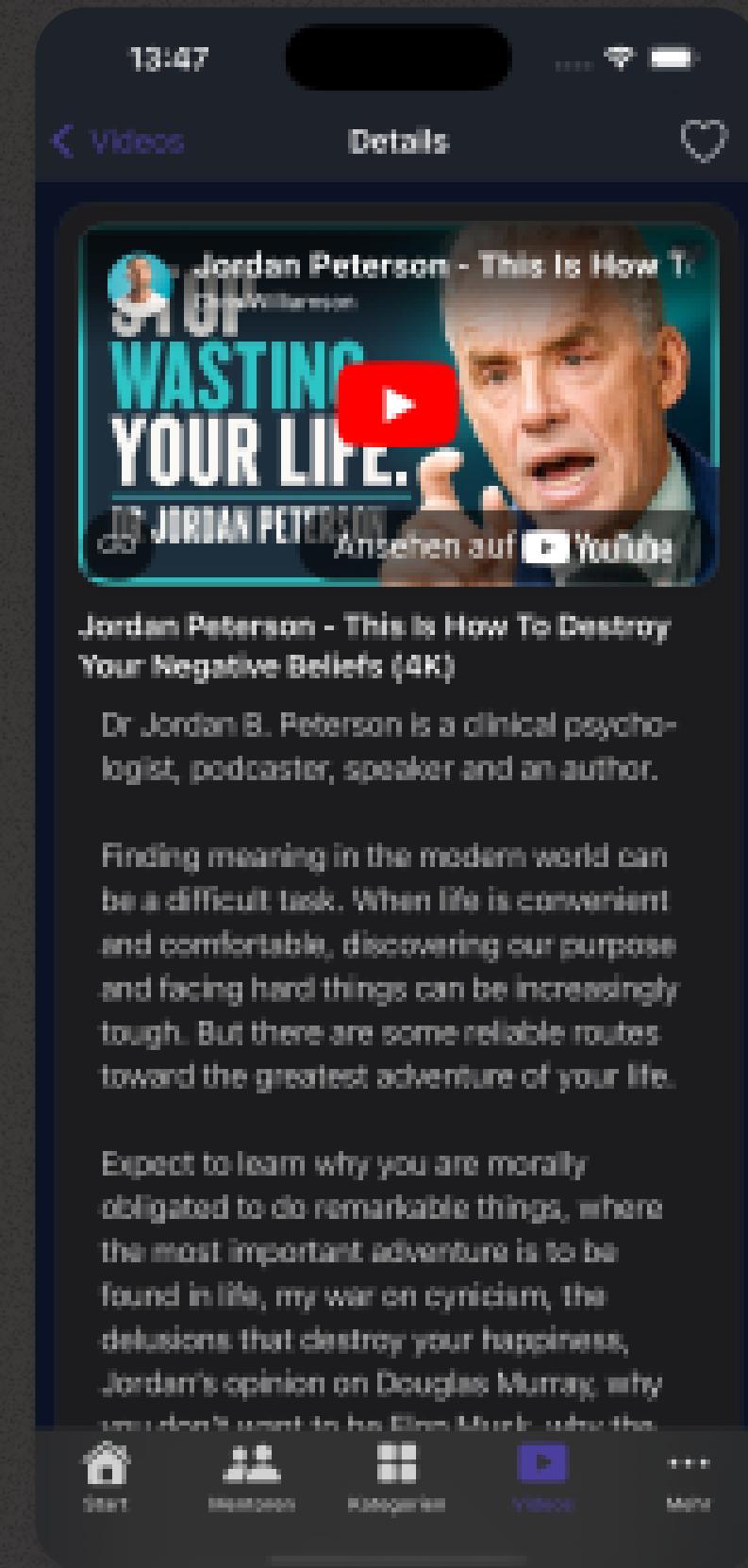
🧠 Onboarding



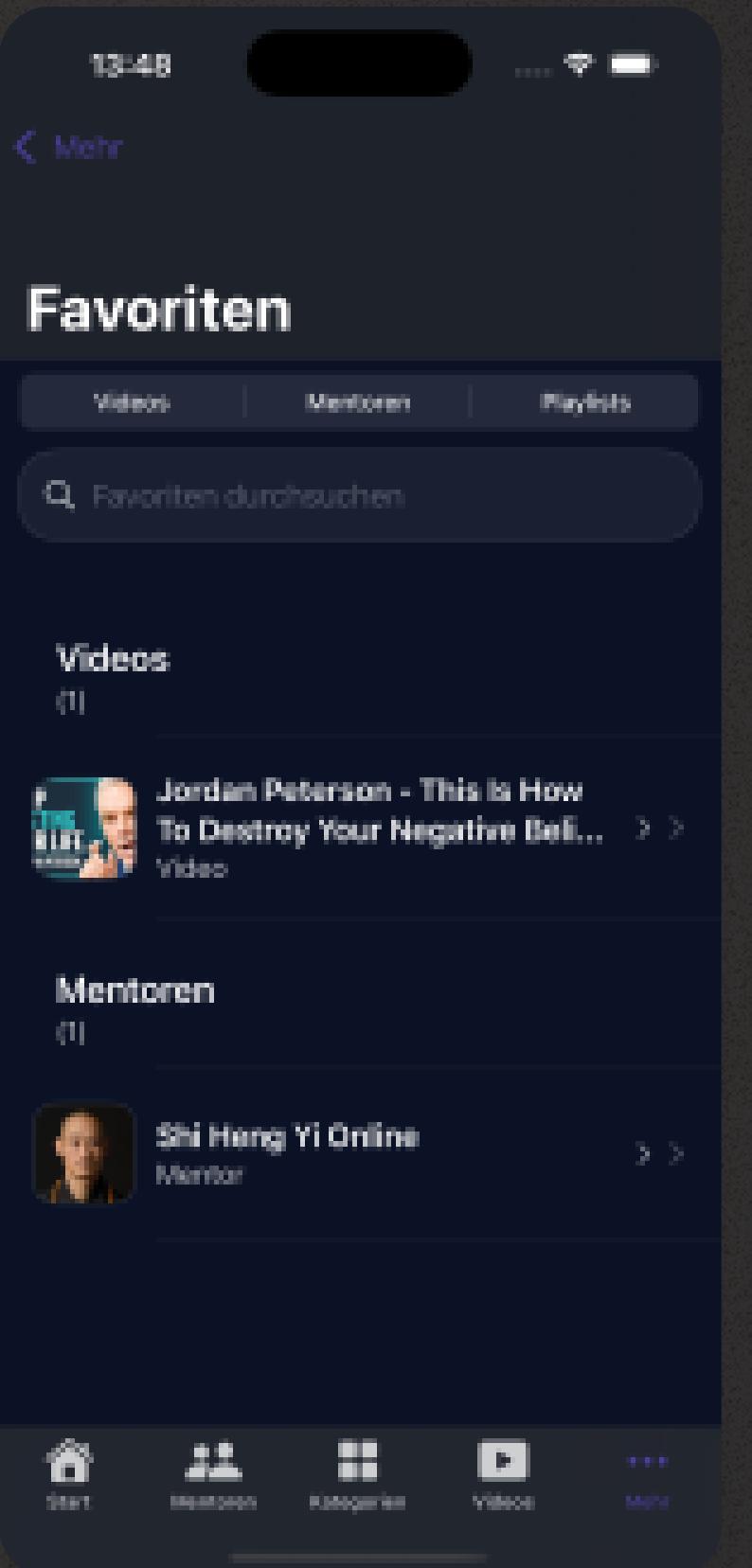
🏠 Startseite



📁 Kategorien



🎥 Videodetail



❤️ Favoriten (gefüllt)



Was ich gelernt habe

🧠 Projektplanung & Fokus:

Ich habe gelernt, wie wichtig eine klare Struktur und der Fokus auf den Nutzernutzen ist – besonders beim Entwickeln schlanker MVPs mit echtem Mehrwert.

🧩 SwiftData & API-Integration:

Ich habe SwiftData tiefer verstanden und gleichzeitig gelernt, wie ich eine externe REST API (YouTube Data API) zuverlässig einbinde – inklusive Fehlerbehandlung und Seed-/Fallback-Strategie bei Verbindungsproblemen.

🎨 UI-Polish & User Experience:

Ich habe gelernt, wie viel Einfluss kleine Details wie Animationen, Dark-Mode-Design und ein konsistentes Farbschema auf die Nutzererfahrung haben.

„MindGear war für mich mehr als nur ein Projekt – es war ein persönlicher Spiegel meines Wegs als Quereinsteiger.“



Eingesetzte Tools & Hilfsmittel



ChatGPT Plus & Codex :

- Unterstützung bei Architektur-Entscheidungen, Debugging & Textoptimierung.
- Einsatz von Codex Connector für Xcode-Review & Slide-Optimierung



Figma :

UI/UX-Design, Moodboards & Prototypen – inkl. Dark-Mode-Gestaltung.



Notion :

Projektplanung, Case Study-Planung & Dokumentation.



GitHub + GitHub Desktop :

- Versionierung, Commits & Pushes, Code-Reviews & README-Dokumentation.
- Branching-Strategie (feature/*), Commit-Konventionen & Markdown für Case Studies



Dank & Ausblick

**Vielen Dank für Ihr Interesse an meinem Projekt MindGear.
Für mich war diese App nicht nur eine technische Herausforderung –
sondern auch ein persönlicher Meilenstein auf meinem Weg als
Quereinsteiger.**

**Ich freue mich auf die nächsten Schritte als Mobile Developer –
gerne in einem Team, das Technik, Klarheit und Menschlichkeit verbindet.**

Vielleicht kreuzen sich unsere Wege – ich freue mich auf den Kontakt. 