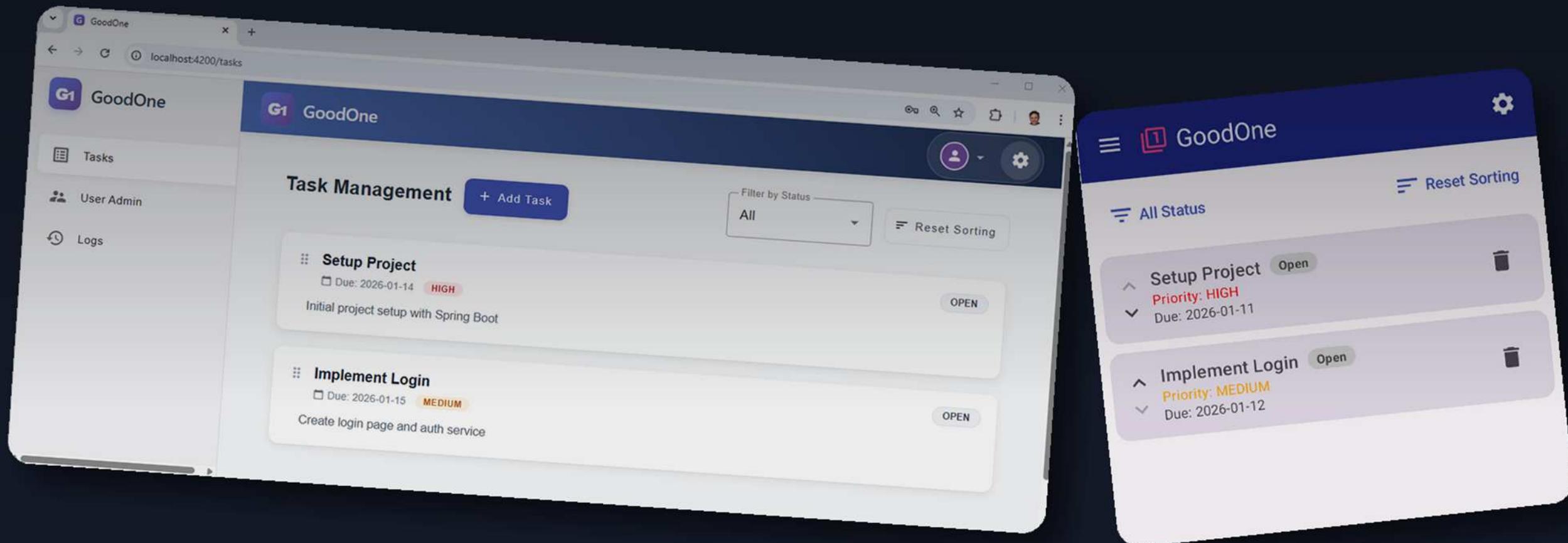


Software-Entwicklung mit AI



Agenda

Demo & Kontext

- Angular Web & Android App

AI-Werkzeuge

- Junie AI, ChatGPT

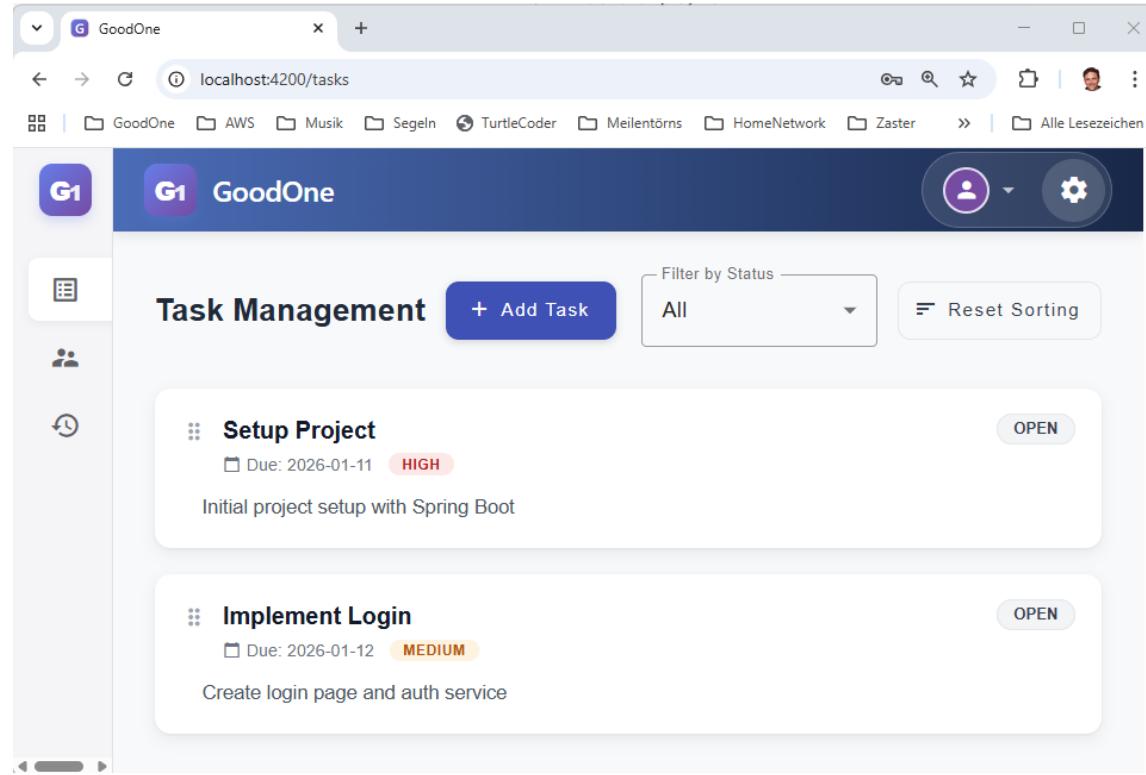
Praxis

- Vorgehensweise & Beispiele
- Wir implementierten ein Feature

Einordnung

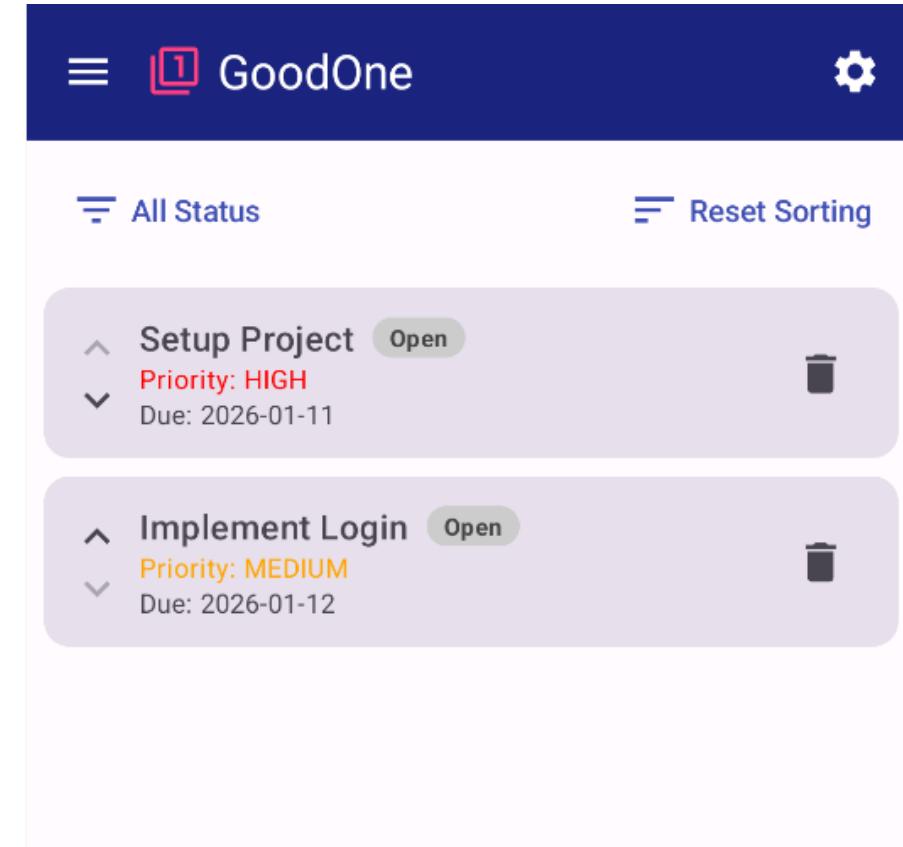
- Fazit & Ausblick

Anwendungsübersicht



The screenshot shows a web browser window for the 'GoodOne' application at localhost:4200/tasks. The interface includes a header with user profile and settings icons, a sidebar with navigation links, and a main 'Task Management' section. The 'Task Management' section features a 'Task List' with two items: 'Setup Project' (due 2026-01-11, priority HIGH) and 'Implement Login' (due 2026-01-12, priority MEDIUM). Each task card has an 'OPEN' button.

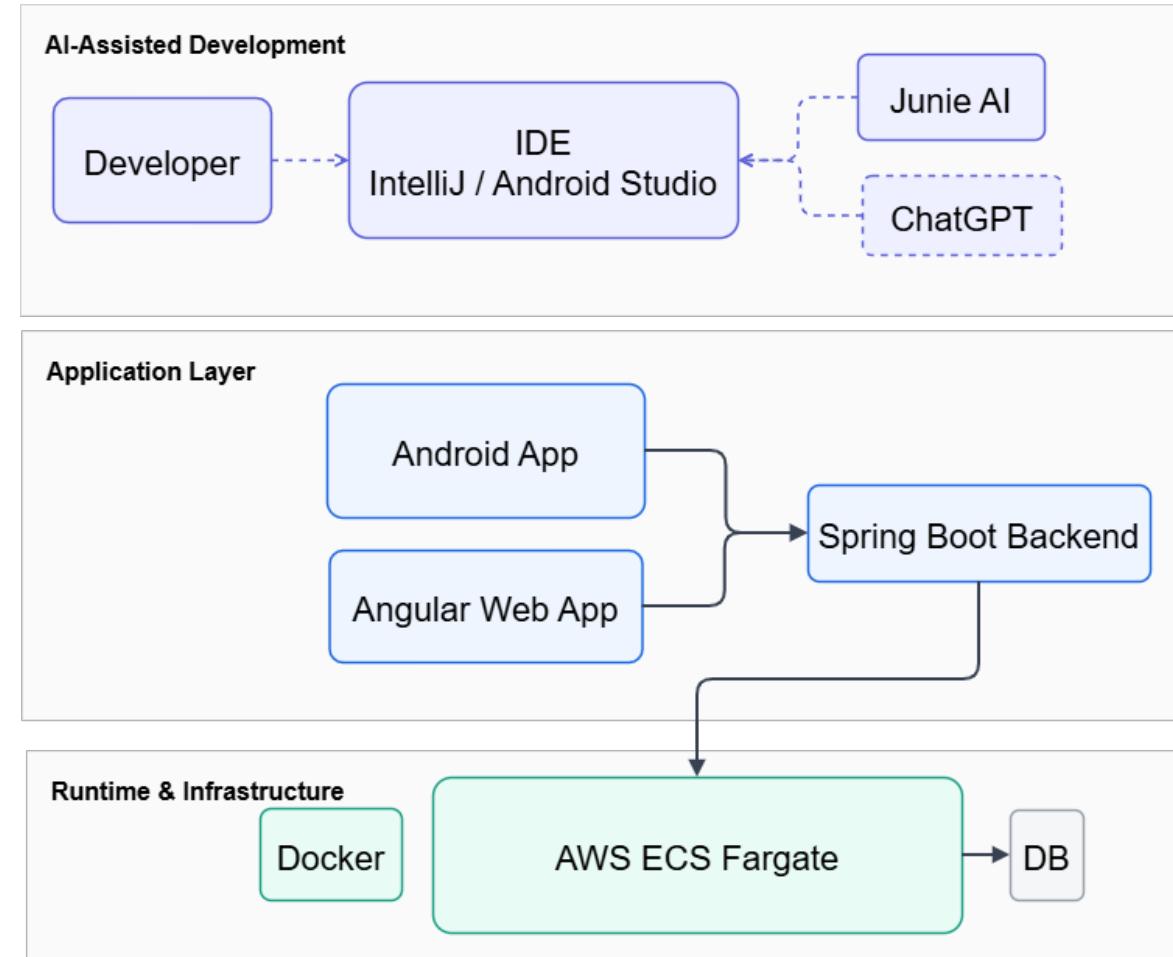
Angular Web-UI www.goodone.ch



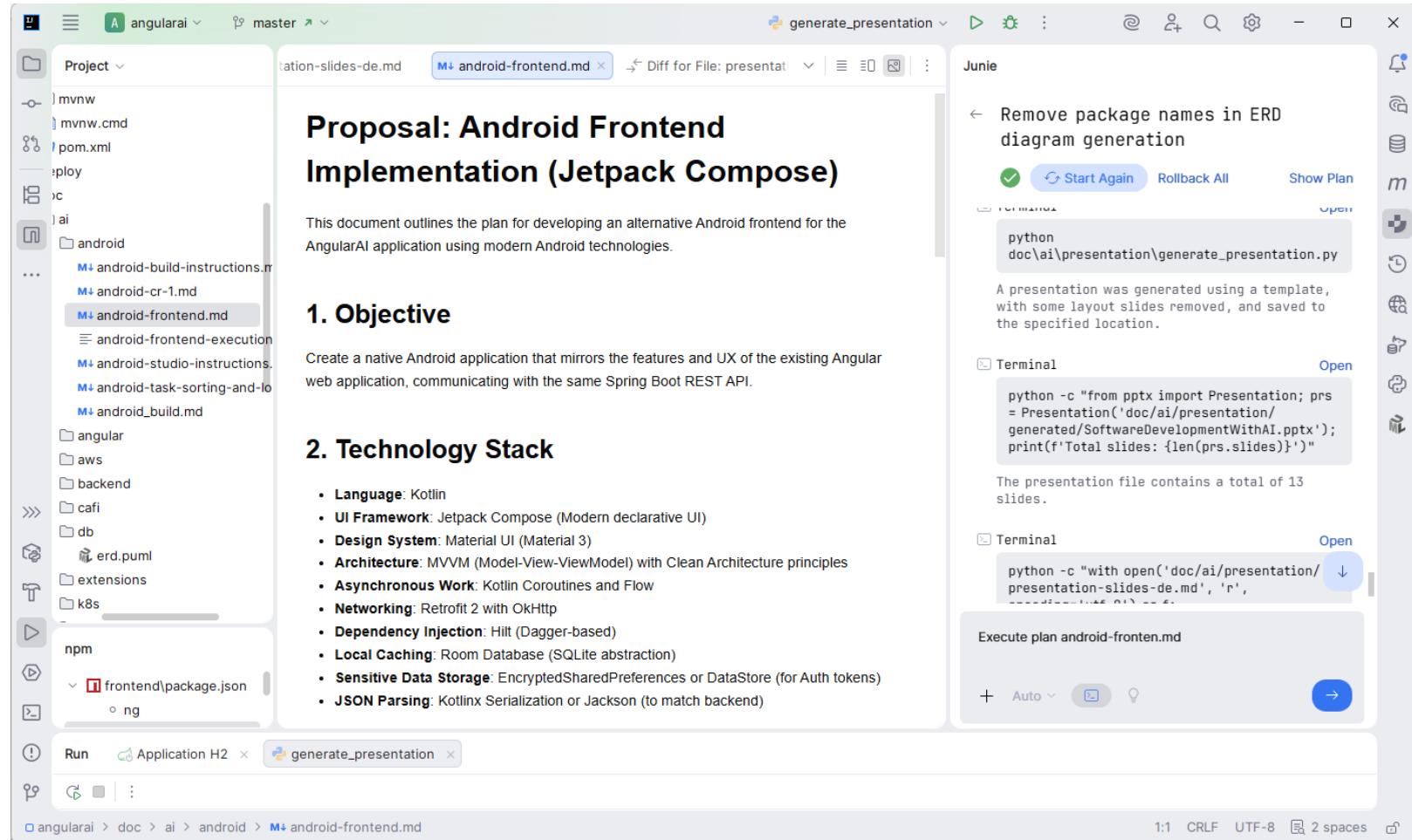
The screenshot shows the mobile application interface for 'GoodOne'. It features a dark blue header with the 'GoodOne' logo and a gear icon. Below the header are buttons for 'All Status' and 'Reset Sorting'. The main content area displays two tasks in a list format: 'Setup Project' (Priority: HIGH, Due: 2026-01-11) and 'Implement Login' (Priority: MEDIUM, Due: 2026-01-12). Each task card includes an 'Open' button and a trash bin icon.

Android App

Architekturübersicht – Infrastruktur



AI unterstützte Entwicklung



The screenshot shows a code editor interface with the following details:

- Project Structure:** The left sidebar displays a project tree for "angularai" with branches like mvnw, deploy, ai, android, angular, aws, backend, etc.
- Current File:** The main editor window shows a Markdown file titled "Proposal: Android Frontend Implementation (Jetpack Compose)".
- Content:** The file contains sections for "1. Objective" and "2. Technology Stack", listing various dependencies and frameworks.
- Terminal Log:** On the right, a terminal window shows the command "python doc\ai\presentation\generate_presentation.py" being run, resulting in a presentation file with 13 slides.
- Logs:** A log pane on the right shows a message from "Junie" about removing package names in an ERD diagram generation.
- Status Bar:** The bottom status bar indicates the file is 1:1 CRLF, UTF-8, 2 spaces.

Architekturübersicht – Logisch

Clients

- Angular Web-UI
- Android App (Jetpack Compose)
- Test-Daten-Generator

Backend

- Java, Spring Boot REST-API

Persistenz

- PostgreSQL (AWS RDS)
- Lokal: H2

Junie AI (JetBrains)

- Code-Generierung, Refactoring, Tests

ChatGPT (OpenAI)

- UX-Design, Architektur-Reviews, Konzeptarbeit

IDE-Integration

- IntelliJ IDEA, Android Studio

Vorgehen bei Design durch AI

1. **Prompting**

- Ziel und Kontext definieren

2. **Struktur**

- Entitäten, Attribute, Menüs

3. **Planung**

- Detaillierter Markdown-Plan

4. **Review**

- Entwickler prüft & ergänzt

5. **Implementierung**

- Code + Tests durch AI

6. **Qualität**

- Entwickler bleibt verantwortlich

Prompting-Beispiele

Implementiere ...

- Android App mit identischen Features wie Web
- Read-only Admin Access
- Behebe Defekt XYZ
- AWS Cloud Deployment
- UI-Integrationstests mit Cypress
- Architektur-Diagramm
- Confluence-Dokumentation
- Powerpoint-Päsentation

KI-gestützt mit Junie (IDE-zentriert)

- Direkte IDE-Integration
- Kontext aus Projekt & Code
- Schnelle Iterationen
- Fokus auf Implementierung

ChatGPT (extern & beratend)

- Architektur- & UX-Feedback
- Alternative Lösungsansätze
- Formulierung & Dokumentation
- Kein direkter Code-Zugriff

Einschränkungen im Firmenumfeld

Security & Compliance

- Keine Kundendaten extern

Tool-Vorgaben

- Zentrale AI-Freigaben

Technologie-Gap

- Verzögerter Zugang zu aktuellen Modellen

Know-how

- Prompting-Kompetenz nötig

Fazit

Demo bestätigt: AI ist produktionsreif

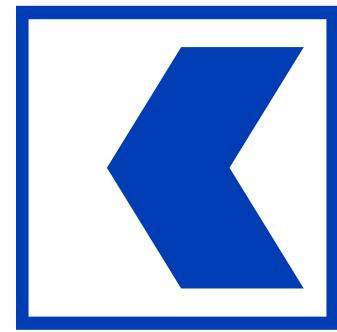
Werkzeuge: Junie & ChatGPT ergänzen sich ideal

Praxis: Weniger Code, mehr Qualität

Ausblick: AI-gestützte Entwicklung wird Standard

Schneller bauen. Besser verstehen. Qualität sichern.

Mit Junie AI & ChatGPT



Zürcher
Kantonalbank