DHBW
Duale Hochschule
Baden-Württemberg
Karlsruhe

Daily Quotes

Software Engineering

Emil Carls, Jan Wilfert, Björn Wiedorn

Harald Ichters
20.06.2024

Gliederung

1. Projektziel
2. Architekturentscheidungen
3. Design Patterns
4. Qualitätssicherung
5. CI/CD Setup
6. Live-Demonstration
7. Fazit und gewonnene Erkenntnisse

Projektziel

- Benutzerfreundliche und inspirierende Plattform
- Tägliche Bereitstellung motivierender und interessanter Zitate
- Ermöglicht Speichern und Teilen von Zitaten
- Entwicklung einer Webapp und Android-App

Projektvision

- Führende Plattform für tägliche Inspiration
- Aufbau einer Gemeinschaft für persönliche Entwicklung
- Quelle der Motivation und positiven Veränderung

Projektvision

- Integrierter Teil des täglichen Lebens der Benutzer
- Nahtlose, intuitive Benutzererfahrung
- Ansprache eines breiten Publikums

Architekturentscheidungen - Anforderungen

1. **Zuverlässigkeit:** Verfügbarkeit
2. **Effizienz**: Performance bei der Anzeige von Zitaten
3. **Sicherheit:** Datensicherheit
4. **Benutzbarkeit:** Bedienbarkeit
5. **Wartbarkeit:** Erweiterbarkeit

1. Zuverlässigkeit: Verfügbarkeit

- Webapp und Android App sollen eine sehr hohe Zuverlässigkeit und Verfügbarkeit haben.
- Zusätzlich soll die Android App weitgehend Internet-unabhängig funktionieren.
- Verfügbarkeit > 99%

1. Zuverlässigkeit: Umsetzung

- Einsatz Cloud-basierte Services
- Skalierbarkeit
- Microservice-Architekturen

➢ Firebase

1. Zuverlässigkeit: Umsetzung

- Funktion bei temporären Störungen der Datenbank
- Offline Synchronisation von Datenbank (Android)

➢ Methode, um Datenbank lokal zu speichern

2. Effizienz: Performance

- Tägliche und gespeicherte Zitate sollen sehr schnell dem Nutzer angezeigt werden

→ Anzeigen in unter 2 Sekunden

2. Effizienz: Umsetzung

- Data Loader
  - Vorladen von Daten minimiert Wartezeit

➢ Offline Synchronisation der Datenbank (Android)

2. Effizienz: Umsetzung

- Passende Strukturierung der Datenbank und Daten

➢ Firebase von Google mit vordefinierten Strukturen

3. Sicherheit: Datensicherheit

- Nutzerdaten sollen nicht verloren gehen beziehungsweise nicht von Dritten eingesehen werden können
- Sensible Daten sollen nicht im Klartext gespeichert werden

3. Sicherheit: Umsetzung

- Verschlüsselte Übertragung
- Verschlüsselte Speicherung

➢ Firebase

4. Benutzbarkeit: Bedienbarkeit

- Die Benutzeroberfläche soll intuitiv sein
- Der Nutzer soll einfach zu den gewünschten Bereichen navigieren können
- Nutzer soll ohne das Studieren einer Anleitung die Webapp und die Android App sinnvoll und korrekt verwenden können

4. Benutzbarkeit: Umsetzung

- Verwendung von Design-Guidelines
- Verwendung von UI-Bibliotheken

➢ Google Material Design

5. Wartbarkeit: Erweiterbarkeit

- Der Code soll innerhalb von 1.5h verstanden werden können
- Aufteilung der Webapp und der Android App in einzelne Komponenten

→ Einfaches und schnelles hinzufügen einzelner Features im Nachhinein

5. Wartbarkeit: Umsetzung

- Microservice-Architekturen
- Einzelne Komponente mit Vue.js
- Wartbarkeit einzelner Komponente ohne komplettausfall der Anwendung

Architekturentscheidungen - MVVM

- Model:
  - Repräsentiert Daten und Geschäftslogik der Anwendung
- View:
  - Benutzeroberfläche der Anwendung
- ViewModel:
  - Verbindung zwischen Model und View

Architekturentscheidungen - MVVM

- Eigene Datei mit Geschäftslogik und Methoden
- Trennung der einzelnen Seiten in Komponente
- Eigene Methoden zur Interaktion zwischen Model und View

Architekturentscheidungen - MVVM

- Klare Trennung der Verantwortlichkeiten → Erleichtertes Warten, Testen und Skalieren
- Synchronisation zwischen View und Model durch Vue vereinfacht
- Einfache Erweiterbarkeit und Wiederverwendbarkeit durch Gliederung in Komponenten

Tech-Stack: Frontend-Entwicklung (WebApp)

- HTML 5: Strukturierung der Webseite
- SCSS (Sassy CSS): Erweiterung von CSS
- JavaScript
- Vue.js: progressives JavaScript-Framework

Tech-Stack: Entwicklungsplattformen (WebApp)

- Code-Editor: VS Code
- Versionskontrollsystem: GitHub
- Paketmanager: npm
- Dokumentationswerkzeuge: Md-Dateien in GitHub + Draw.io
- Projektmanagement-Tool: Jira

Tech-Stack: Design Patterns

- Single Responsibility Principle (**S**OLID)
- Jede Komponente hat eine klar definierte Aufgabe
- Vorteile:
  - Erhöhte Wartbarkeit
  - Skalierbarkeit und Erweiterbarkeit

Tech-Stack: Frontend-Entwicklung (Android)

- Kotlin
- Jetpack Compose: Erweiterung von Kotlin zur funktionalen Entwicklung der UI
- Material Design 3: UI-Bibliothek von Google

Tech-Stack: Entwicklungsplattformen (Android)

- Code-Editor: Android Studio
- Versionskontrollsystem: GitHub
- Build-Tool und Paketmanager: Gradle
- Dokumentationswerkzeuge: Md-Dateien in GitHub
- Projektmanagement-Tool: Jira

Qualitätssicherung

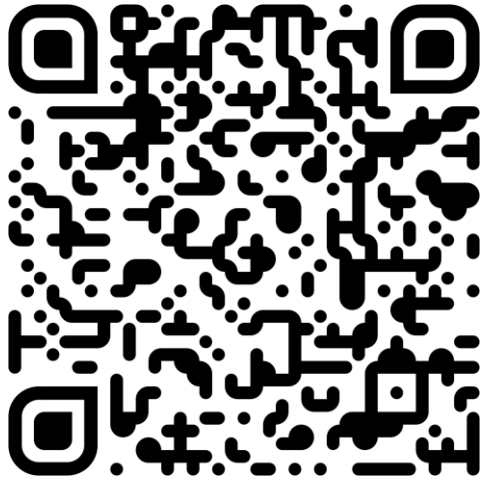
- Manuelle Tests
- Extensives Logging
- Vereinzelt automatisierte Tests

CI/CD Setup

- Node.js / CI build
  - Ausführung bei push/ pull request
  - Prüft ob Projekt gebuildet werden kann
- Lint Code Base
  - Analysiert Projekt
  - Meldet Probleme und Guideline Verstoße

DHBW
Duale Hochschule
Baden-Württemberg
Karlsruhe

Live Demonstration



https://play.google.com/store/apps/details?id=com.emil.dailyquotes

DHBW
Duale Hochschule
Baden-Württemberg
Karlsruhe

Live Demonstration

Fazit: Gewonnene Erkenntnisse

- Wichtigkeit einer effektiven Zeiteinteilung
- Notwendigkeit, Pufferzeiten für unvorhergesehene Probleme einplanen
- Vollständige UI-Mockups: Beschleunigen und vereinfachen Entwicklungsprozess

Fazit: Gewonnene Erkenntnisse

- Absprache und Zuverlässigkeit aller Teammitglieder
- Nutzen regelmäßiger Meetings zur Statusüberprüfung und Problemlösung
- Wichtigkeit von Code-Reviews zur Verbesserung der Codequalität

Danke für Eure Aufmerksamkeit