

**Web Engineering**

# **Projekt-Kickoff**

Adrian Herzog

# Übersicht

**Ziel:** *Entwickeln einer «realistischen» Web-App,  
basierend auf eigener Idee.*

Inspiration: <https://flaviocopes.com/sample-app-ideas/>

teilweise deutlich  
zu einfach

**Modus:** Einzelarbeit

**Umfang:** ca. 30 Stunden

**Abgabe-Format:** Spring-Boot-Applikation über GitHub Classroom

# Zeitlicher Ablauf

heute, 1.4.2025	<b>GitHub Classroom</b> Einladung annehmen (kommt per Teams)
<b>spätestens Montag, 7.4.</b>	<b>Applikations-Vorschlag</b> per E-Mail an mich (ca. 5 Sätze). Ich bestätige den Vorschlag oder schlage Anpassungen vor.
ab 8.4.	Arbeit am Projekt ausserhalb der Unterrichtszeit
13.5.	Projekt-Unterstützung (und Thema REST APIs)
20.5.	Projekt-Unterstützung, Input zu vermeidbaren Fehlern
27.5.	Projekt-Unterstützung
<b>27.5. um 23:59</b>	<b>Spätester Zeitpunkt für Abgabe (via GitHub)</b> <i>Letzter Push zählt. Stelle sicher, dass in dieser Version alles funktioniert (auch die Tests).</i> <i>Last-Minute-Änderungen sind keine gute Idee!</i>
3.6.	<i>Präsentation der Projekte, 7 Minuten pro Person</i>

# Verwendung von externem Code

## Erlaubt:

- Kopieren von kleinen Codestücken von Stack Overflow, Tutorials, usw.
- Verwenden von Drittbibliotheken (siehe Erlaubte Technologien.xlsx)
- Diskutieren mit anderen Personen, Beratung durch Andere, Copilot und Chatbots

## Kopierter oder generierter Code muss klar markiert sein!

- Kommentar in Code (oder Commit-Message), inkl. «EXTERNAL» und Quelle
- Ausnahme: single line code completion muss nicht deklariert werden

## Nicht erlaubt:

- JavaScript Single Page App Frameworks (React, Angular, Svelte, Vue.js, etc.)
- CSS-Frameworks (Bootstrap, Tailwind, etc.)
- Kopieren von Code aus den Übungen oder Musterlösungen  
(Ausnahme: kleine Dinge, die man «nicht anders machen kann»,  
z.B. Einstellungen in application.properties)

# Bewertung

Siehe Bewertungsraster in Excel (Bewertungsraster.xlsx)

## 50 Punkte möglich

- Funktionalität: 20 Punkte
- Engineering: 20 Punkte
- Individuelles Zusatzthema: 5 Punkte
- Präsentation: 5 Punkte

## Note

- Skala: 1 + Punkte / 10
- Betrugsversuch  
(z.B. nicht deklarerter externer Code) Note 1

*schon passiert!*


# Zusatzthema

- Auswahl eines Themas aus Liste von Ideen (siehe [Bewertungsraster.xlsx](#)).
- Alternativ sind auch eigene Ideen möglich, müssen aber von mir per Mail zusammen mit der Applikations-Idee genehmigt werden.
- Es wird nur ein Zusatzthema bewertet, auch wenn mehrere umgesetzt werden. ***Das zu bewertende Zusatzthema muss auf der Info-Seite der Applikation als solches angegeben werden.***

# Präsentation

- 7 Minuten pro Person
- Hat Einfluss auf die Note (siehe nächste Folie)
- Inhalt:
  - Ziel und Zweck der Applikation
  - Demo (1 bis 2 Minuten)
  - Zusatzthema
  - Reflexion
    - Herausforderungen und wie ich damit fertig wurde
    - Wo hab ich am meisten gelernt?
    - Was würde ich anders machen?
  - Fragen der Zuhörenden beantworten

# Beispiel: Q&A App



bitte nicht  
verwenden

3 Entitäten: *Fragen*, *Antworten*, *Tags* (Themen)

- Eine Frage kann mehrere Tags enthalten (N:M)
- Zu einer Frage kann es mehrere Antworten geben (1:N)

CRUD-Unterstützung für Fragen, Erstellen & Löschen von Antworten.  
Vordefinierte Tags in DB (werden beim ersten Starten reingeladen)

Engineering-Aspekte:

- Sinnvolle Unit-Tests, Integration-Tests, E2E-Tests
- Valides HTML, semantische Elemente
- Sauberer Code & saubere Git-History



# Erlaubte Technologien

Siehe Erlaubte Technologien.xlsx

Wenn unsicher, bitte einfach nachfragen, Liste wird dann ergänzt

	Vorlesungs-Stack	Erlaubte Alternativen	Nicht erlaubt
<b>Frontend</b>	HTML	keine	
	CSS ohne Libraries	keine	Bootstrap, Tailwind, etc.
	Kein JavaScript	<ul style="list-style-type: none"> <li>- Selber geschriebenes JavaScript</li> <li>- JS Libraries zur Darstellung von Charts und Graphen (z.B. Cytosape.js)</li> <li>- JS Libraries zur Verwendung von WebSockets</li> <li>- JS Rich Text Editor (z.B. Quill)</li> <li>- Weitere spezifische Libraries (z.B. Tagify)</li> </ul>	<ul style="list-style-type: none"> <li>- JS SPA-Libraries wie Vue, Svelte, React, Angular</li> <li>- Alles was einen Frontend-Build benötigt und nicht einfach direkt eingebunden werden kann.</li> </ul>
<b>Backend</b>	Java 21	Kotlin	
	Spring Boot	Quarkus	
<b>Templating</b>	Pebble	Qute (mit Quarkus), kotlinx.html (mit Kotlin)	
<b>Datenbank</b>	JPA / Spring Data		
	H2 DB		Datenbank, die man zuerst installieren muss
<b>E2E Tests</b>	Selenium	Playwright, HtmlUnit, Cypress	
<b>Deployment</b>	JVM	keine	
<b>Weiteres</b>		Lombok für Java	
		Libraries für Java oder Kotlin (z.B. Um Grafiken zu generieren oder ein GraphQL API zu konsumieren)	
		Font Awesome Icons	

# Git & Hilfe



Nur was auf **master** ist,  
zählt!

## Zusätzliche Anforderungen:

- Die gesamte Entwicklung muss über das persönliche Git Repository in GitHub Classroom laufen
- Pushe deinen neusten Entwicklungsstand regelmässig auf GitHub

*Commit early, commit often!*

## Hilfe erhalten:

- 20 Minuten selber versuchen, Problem zu lösen (Google, Stack Overflow, Chatbots)
- Wenn möglich 20 Minuten mit Mitstudierenden versuchen, Problem zu lösen
- E-Mail an mich, mit Problem-Beschreibung (inkl. Compiler-Fehler oder vollständigem Stack Trace) und *Links zu Code auf GitHub!*