



OST
Ostschweizer
Fachhochschule

PROJEKT 2 - INFORMATIONEN

CAS Front-End Engineering

Michael Gfeller

5. Juli 2021

Allgemeines

Start:	Heute
Präsentation:	12. Januar 2022
Abgabe:	16. Januar 2022
Gruppenarbeit:	2er Teams

Einschreibe-Liste:

https://ostch-my.sharepoint.com/:x:/g/personal/michael_gfeller_ost_ch/EeQGxf7oTFxMnsh0MYQJI80BS01WM-TRMkv9BrpFleANIA?e=qBLLQM

Gruppenfindung


- Wonder.me / BBB / ...
- Pause
- ...

Ziel: Single Page Applikation erstellen

- Angular
- React + Redux

Ablauf / Meilensteine

- **Ab jetzt**
 - Gruppen suchen, Ideenfindung
- **Bis nach den Sommerferien**
 - Gruppensuche abgeschlossen, Idee für Projekt 2 definiert
 - Wireframes / Mockups (Skizzen) erstellen
 - Start mit der Implementation (Client und Backend)
- **27. August**
 - Deadline: für die Einreichung der Projektidee mit Bestätigung durch den Betreuer
- **28 Oktober**
 - Deadline: Technischer Prototype mit Wireframes
- **15. Januar**
 - Präsentation
- **19. Januar**
 - Abgabe per Mail
- **ca. 2 Wochen später**
 - Persönliches Feedback zum Projektes 2 wird per Mail zugestellt.

Block 2 Case Studies				
05.07.21	Montag	8.U44/a	Unterrichtszeit: 09:00 – 18:00 Uhr	M. Gfeller M. Stolze Egli / Ritter
			Briefing für Projekt 2, Start Vorbereitungsphase Clean Coding mit JavaScript; Angular Tutorial (Teil 2)	
07.07.21	Mittwoch	1.209	React mit Test-Driven Development; React Router; Hooks basics	A. Kühne
10.07.21 - 08.08.21			Sommerferien	
09.08.21	Montag	1.209	Ausweichtermin (Reserve)	--
11.08.21	Mittwoch	1.209	Advanced React mit Context API, Hooks, TypeScript und Redux	D. Modalek
18.08.21	Mittwoch	1.209	Advanced React mit Context API, Hooks, TypeScript und Redux	D. Modalek
25.08.21	Mittwoch	1.209	Angular Praxis (1)	Egli / Ritter
01.09.21	Mittwoch	1.209	HTML + CSS 2 (Tools): Einsatz von Tools und Frameworks (z.B. SCSS, Image-post-processing, Bootstrap, Angular Material, React Material)	C. Sany
08.09.21	Mittwoch	1.209	Angular Praxis (2)	Egli / Ritter
15.09.21	Mittwoch	1.209	Angular Praxis (3) / PWA	Egli / Ritter
22.09.21	Mittwoch	1.209	Serverless Data-Management) / Kick-Off: offizieller Start Projekt 2	Stocker/Gfeller
29.09.21	Mittwoch	1.209	Komplexe Sites modular strukturieren	A. Dini
06.10.21	Mittwoch	1.209	Ausweichtermin (Reserve)	--
09.10.21 – 24.10.21			Herbstferien	
27.10.21	Mittwoch	1.209	Ausweichtermin (Reserve)	--
03.11.21	Mittwoch	1.209	HTML + CSS 3 (Praxis): Beispiel(e) von Herausforderungen und Lösungen im Bereich HTML, CSS und Vue bei Unic	C. Sany
10.11.21	Mittwoch	1.209	Ausweichtermin (Reserve)	--
11.11.21	Donnerstag	Aula	World Usability Day (OST) - fakultativ	

Ablauf / Meilensteine

- **Ansonsten**

- Arbeiten im Projekt-Team, in der weiteren Zeit- und Projektplanung ist das Team frei.
- Bei Team-Problemen frühzeitig Betreuer aufsuchen.
- Neue Inputs der Vorlesungen ins Projekt einfließen lassen z.B.
Fluides Design – Vorlesung: Projekt überprüfen ob «erfüllt» ansonsten versuchen dies zu berücksichtigen.

Deadline: Technischer Prototype

- **Zur Deadline muss ein technischer Prototype inkl. Wireframes vorhanden sein.**
 - Durchstich von User Interface bis zur "Datenbank".
 - z.B. Erfassen von X und darstellen.
 - Git-Repo muss vorhanden sein.
 - Jedes Gruppenmitglied muss aktiv gewesen sein.
 - Wireframes in Figma/Miro/... müssen vorhanden sein. Hinweis: Die Wireframes müssen nicht final sein.
- **Der technische Prototype soll sicherstellen...**
 - ... ob das Team funktioniert.
 - ... ob die richtige Technologie ausgewählt wurde.
 - ... das die Wireframes vorhanden sind für die weitere Entwicklung.
- **Bei nicht erfüllen, wird das weitere Vorgehen mit dem Team besprochen.**
 - Splitten vom Team
 - Pausieren
 - ...
- **Datum: 27. Oktober**

Projekt Ideen

- Reddit
- Webshop
- Spiel z.B. Online-Schach, 4 Gewinnt, Schiffchenversenken, ...
- Eigener Vorschlag

Allgemeine Anforderungen

- **Usability**
 - Funktionalität muss Userfriendly sein.
 - Benutzer Test (Bild machen als Beweis)
 - Fluid & Responsive Design
 - Tablet / PC und Smartphone
- **Code-Qualität**
 - Saubere Architektur auf dem Client
 - Sauberer Code
 - CSS-Qualität sicherstellen auch kein Copy und Paste z.B. mittels CSS-Preprozessoren
- **Projekt**
 - Installation soll einfach (oder beschrieben sein)
- **Tests**
 - Sinnvolle Unit-Tests für Domain-Klassen
 - User Interface Tests, falls sinnvoll

Schema von der letzten Durchführung

Funktionsumfang	
	User-Management
	Security Modul
	[Modul 1 z.B. Produkte Verwaltung]
	[Modul 2 z.B. Warenkorb]
	[Modul 3 z.B. Rating]
Client	
	Client-Architektur
Server	
	REST API
	Server-Architektur
User Experience	
	Fluides Design
	Bonus: Usability-Pattern verwendet z.B. Blank Slate
	[Projekt spezifisch 1 z.B. Spielspass]
	[Projekt spezifisch 2 z.B. Daten werden synchronisiert]

Tests	
	Sinnvolle Unit-Tests vorhanden
	Bonus: User Tests Die Erkenntnisse müssen dokumentiert sein
	Bonus: E2E Tests
Code Qualität	
	Besonders schlechte Konstrukte
	Besonders gute Konstrukte
	Einsatz von CSS-Präprozessoren (Sass / Less / ...)
Abgabe	
	Readme mit Installationsanleitung / Projektbeschreibung
Sonstiges	
	Bonus: Sinnvolle Code-Dokumentation
	Keine JS Errors
	Besonders Gut
	Besonders Schlecht

Hinweise zum Schema

- **Note 6** **Projektvorschlag mehr als erfüllt; Die Usability und die Qualität ist vorbildlich.**
- **Note 5** **Projektvorschlag erfüllt; Qualität vom Projekt geht in Ordnung.**
- **Note 4** **Projektvorschlag erfüllt; Mängel bei der Umsetzung.**
- **Ungenügend** **Projektvorschlag nicht erfüllt oder starke Mängel bei der Umsetzung.**

- **Der Funktionsumfang wird wie folgt bewertet:**
 - 100% der Punkte: Modul funktioniert und **hohe Usability**
 - 70-80% der Punkte: Modul funktioniert
 - 30-50% der Punkte: Modul funktioniert aber **schlechte Usability**
 - 0% Punkte: Modul funktioniert nicht / nicht vorhanden

Hinweise zum Schema

- Punkte im Schema können noch ergänzt/geändert werden
- Je nach Projekt / Implementation kann ein Punkt obsolet werden.
In diesem Fall wird dieser gestrichen oder durch einen alternativen Punkt ersetzt,
z.B. Firebase / Cloud Firestore verwendet.

In diesem Fall sollte man sich mit den «Best-Practices» von Firebase nachvollziehbar auseinandersetzen u.a. Security / Deployment
- Falls besondere Leistungen erbracht wurden, sollten diese im ReadMe erwähnt werden.
Als Beispiel: Hohe Accessibility / Offline Modus ...
Diese Hinweise werden berücksichtigt
- Die Module sind projekt spezifisch. Falls diese nicht klar ersichtlich sein sollte - sollten diese im ReadMe angegeben werden.
- Bei den Modulen wird auch die Usability bewertet - es sind User Tests zu empfehlen.
- Bei Unit-Tests sind 3-5 **sinnvolle** Unit-Tests zu definieren. Eine Testabdeckung von 100% ist nicht erforderlich.

Reddit

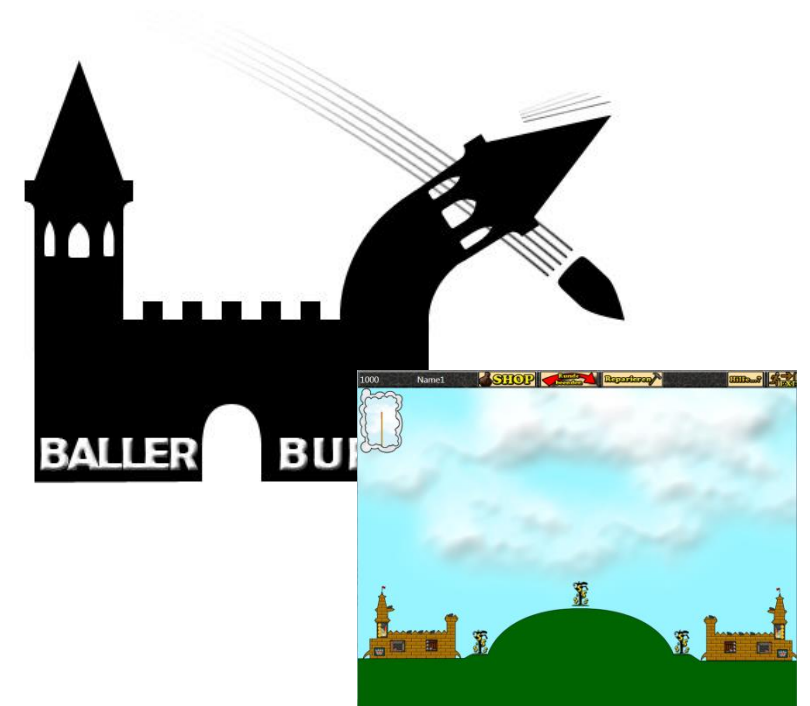
- **Link Modul**
 - Erfassen / löschen.
- **Kommentar Modul**
 - Links kommentieren.
 - Optional: Kommentare kommentieren können.
- **Rating Modul**
 - Links und Kommentare bewerten können.
 - Eigene Ratings wieder rückgängig machen.
 - Optional: Rating-Charts / Übersicht mit den «aktivsten/besten/...» Links.
- **User Modul**
 - User erfassen und verwalten.
- **Security Modul**
 - Login / Logout
 - Kommentare / Links / Ratings erstellen / löschen falls erlaubt.
- **Sonstiges**
 - Updaten von neuen Einträge via Web-Sockets z.B. wie Facebook.

Webshop

- **Produkt Modul**
 - Erfassen / löschen.
 - Warenhaltung
- **Warenkorb Modul**
 - Produkte hinzufügen / entfernen.
 - Checkout
- **Rating Modul**
 - Produkte bewerten / eigene Ratings wieder rückgängig machen.
- **User Modul**
 - User erfassen und verwalten.
 - Bestellungs-Verlauf
- **Security Modul**
 - Login / Logout
 - Optional: Auch anonymer Check Out möglich.
- **Sonstiges**
 - Falls Artikel nicht mehr an Lager soll der User benachrichtigt werden.

Spiel

- **Spiel-Idee**
 - Runden oder Echtzeit
 - Schach oder Risiko oder...
- **Spieler Management**
 - Spieler Verwaltung
 - Login / Logout
 - Statistik
- **Spiel-Bausteine als REST-Ressource**
- **Optional: Lobby**
- **Optional: Chat**
- **Nicht funktionale Anforderungen**
 - Spielspass



Verbesserung einer bestehender Applikation

- Eine bestehende App / Webseite auswählen
- UX-Probleme erkennen / UX-Tests durchführen
- Die (verbesserte) Eigen-Interpretation als SPA implementieren
- Gleiche Anforderungen wie ein «Eigenes Projekt»

Eigenes Projekt

- **Single Page Applikation**
- **Server anbinden**
- **5 Module**
 - User Management
 - Security
 - 3 Feature Module davon maximal 2 CRUD Module.
 - Optionale und nicht optionale Features andenken.
- **Sich nicht überschätzen**
- **Projekt mit Arbeitgeber möglich:**
 - Präsentation muss erlaubt sein.
 - Proojekt muss für sich alleine stehen.
 - Validation der Arbeit muss möglich sein.
 - Installation / Validation darf nicht zu aufwendig sein.

Eigenes Projekt: Einreichen

- **Mail an den Betreuer mit folgendem Inhalt:**
 - Kurze (!) Beschreibung
 - Beschreibungen der 5 Module-Blöcke
 - Änderungen an den Modulen ist nur erlaubt, falls die Komplexität und der Umfang sich nicht ändert.

Eigenes Projekt: Einreichen

Beispiel:

Silvan und ich (Michael) möchten gerne eine Pizza-Lieferdienst erstellen. Wir haben uns folgende Features überlegt:

- **User Module**
 - Bestellung soll mit eingeloggtem User aber auch anonym funktionieren.
- **Security**
 - Rollen: Admin, Normaler, Anonymer-User
- **Bestellung**
 - Bestell-Prozess / Checkout
- **Pizza CRUD / Pizza-Konfigurator**
 - Möglichkeit zum Erfassen von Standard-Pizza.
 - Optional: Eigene Pizza-Rezepte ermöglichen.
- **Bestellungen bearbeiten**
 - Pizza-Bäcker erhält eine Übersicht mit allen Bestellungen und kann diese bearbeiten.
 - Optional: Push Nachrichten bei Status-Änderungen.

Optionale Ideen

- Deployment in die Cloud
- Performance Optimierung
- Progressive Web Apps
 - Offline Modus
- Universal App mit Angular
- Multi Language
- Web Content Accessibility Guidelines (WCAG) 2.0
 - <http://www.w3.org/WAI/intro/wcag.php>
- SEO
- ...

Projekte aus den letzten Jahren

- **Markplatz**
 - <https://bartly.ch>
- **Travel Doodle**
 - <https://travel-doodle.firebaseio.com>
- **Rezepte Verwaltung**
 - <http://lechef.noerdli.ch/#/>
- **Time Tracker**
 - <https://protrack.firebaseio.com/>
- **Ernährungs-Berater**
 - <https://nuba-c3e84.firebaseio.com/journal>
- **Simulation**
 - <https://calc-app.firebaseio.com/simulation>

Ablauf: Nächste Schritte

1. Gruppenbildung

- Git Repository aufsetzen. Private oder public ist erlaubt, die Betreuer benötigen Zugriff.
 - <https://gitlab.ost.ch/> / <https://github.com/>
- Wichtig: Die Teammitglieder müssen **nachvollziehbar** einen ungefähr gleichen Aufwand fürs Projekt 2 aufbringen.

2. Projekt Definieren

- Projektvorschlag an michael.gfeller@ost.ch

3. Backend entscheiden

- Cloud Solutions
 - Google: <https://firebase.google.com/> und <https://firebase.google.com/docs/firestore/>
 - Amazon: <https://aws.amazon.com/de/dynamodb>
- Node.js
 - Fokus soll auf dem Frontend liegen!
- Bei anderen Technologien (ASP.NET, Springboot) muss ein Server zu Verfügung gestellt werden in der «Cloud».

4. Wireframes / Skizzen erstellen

- Wie könnte die Applikation aussehen.

Präsentation

- **Ablauf:**
 - ca. 7min Präsentation
 - ca. 3min Fragen
 - 2min Umbau
- **Inhalt der Präsentation:**
 - Thema kurz erläutern.
 - Auf folgende Punkte eingehen:
 - Besonders gut gelöst?
 - Würden "wir" anders machen?
 - Was haben "wir" gelernt?
- **Kleine Demo**
- **Mehr Details zum letzten Tag folgen.**

Fragen?

