

Miniprojekt WED 3

Ihre Miniprojekt-Aufgabe besteht darin, eine Finance-Webapplikation in Angular 2 sowie React zu programmieren. Es soll möglich sein, Geld-Transaktionen zwischen verschiedenen Konten zu veranlassen.

Die Grundlagen bilden die Wireframes unten, welche den Funktionalitätsumfang zeigen. Zusätzlich definiert das Video „Testat-WED3.mp4“ die dynamische Ansicht der Webseite mit weiteren, konkreten Anforderungen.

Ihre Aufgabe ist den kompletten Funktionsumfang der Wireframes zu implementieren und die im Video visualisierten Feinheiten zu berücksichtigen. Die jeweiligen Vorlagen sind von der Technologie (Angular 2 / React) abhängig und werden in separaten Dokumenten im Übungsordner beschrieben. Die Server-Funktionalität wird durch eine Node.js Lösung zur Verfügung gestellt und liegt dieser Testat-Beschreibung bei. Die **Server API steht im Readme.md File** innerhalb des Testat-Server_vX.Y.Z.zip Files beschrieben.

Sie können am Aussehen Anpassungen vornehmen. Beachten Sie, dass das Design im Mindesten fluid sein sollte. Sie dürfen externe Libraries wie Bootstrap / Material Design / ... für reine Design-Zwecke einsetzen.

Login / Registrieren

Finance Portal

http://finance/welcome

Home Registrieren

Willkommen

Benutzername:

Passwort:

Login

http://finance/ verweist auf http://finance/welcome.
Registrier-Eingabe analog der Loginmaske.

Dashboard

Finance Portal

http://finance/dashboard

Home Konto Bewegungen Logout Michael Gfeller

Neue Bewegung

Von

Zu

chf

Überweisen

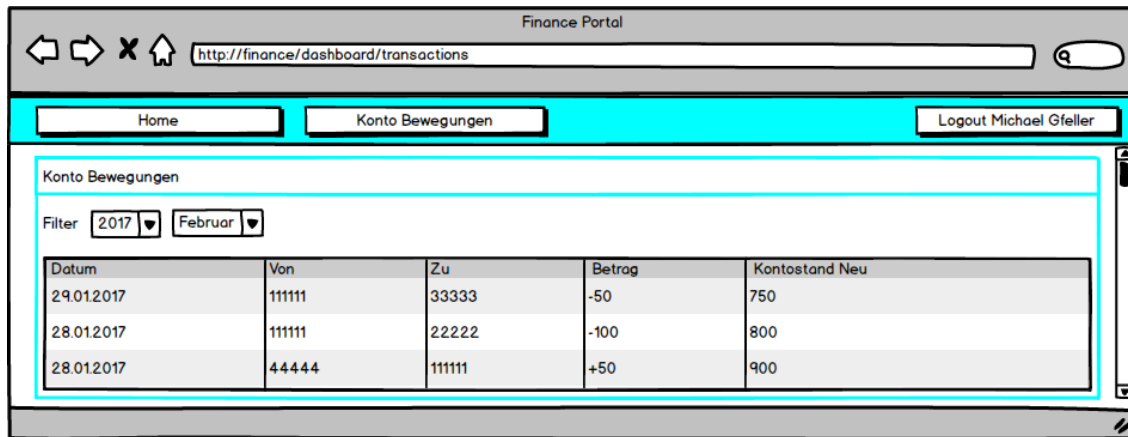
Letzte Bewegungen

Von	Zu	Betrag	Kontostand Neu
111111	33333	-50	750
111111	22222	-100	800
44444	11111	+50	900

Alle Anzeigen

Das Zielkonto (Zu) wird überprüft und der Name des Empfängers wird angezeigt - falls Nummer stimmt. Ansonsten Fehler.
Die letzten Bewegungen (rechts) müssen nach der erfolgreichen Transaktion updated werden.

Konto Bewegungen



Datum	Von	Zu	Betrag	Kontostand Neu
29.01.2017	111111	333333	-50	750
28.01.2017	111111	222222	-100	800
28.01.2017	444444	111111	+50	900

Filterung der Bewegungen nach Jahr und Monat.

Durchführung

Die Arbeit ist in 2er Teams durchzuführen.

1er und 3er Teams müssen bis zum 16. März bewilligt sein.

Bitte registrieren Sie Ihre Gruppe bis spätestens 16. März im folgenden Dokument:

<https://docs.google.com/spreadsheets/d/1CmSDaSUX9IDmvBAXCf1Xne67h7Vjn6ZtV5BVCEskUyM/edit#gid=0>

Review / Abgabe

Sie werden als Gruppe Ihre Lösung während der Übungsstunde dem Übungsleiter präsentieren. Alle Gruppenteilnehmer müssen anwesend sein.

Zusätzlich schicken Sie dem Übungsleiter Ihre Lösung als zip per E-Mail (OHNE den Ordner "node_modules"!).

Das Review der Lösung findet in der Woche 10 (Thema „Server: Grundlagen von ASP.NET MVC“) statt.

Für den Review-Termin können Sie sich im folgenden Dokument einschreiben. Bitte beachten Sie, dass Sie die Zulassung der Prüfung nur bei bestandenem Testat erhalten.

<https://docs.google.com/spreadsheets/d/1CmSDaSUX9IDmvBAXCf1Xne67h7Vjn6ZtV5BVCEskUyM/edit#gid=971143218>

Nachkorrektur

Jede Gruppe hat die Möglichkeit für eine Nachkorrektur. Diese erfolgt eine Woche später. Eine zweite Nachkorrektur wird nicht akzeptiert.

Nachkorrekturen müssen entweder nochmals präsentiert werden, oder bedürfen einer erneuten Abgabe der korrigierten Lösung per Mail an den Betreuer. Der Betreuer entscheidet über die für die Korrekturen erforderlichen Schritte.

Checkliste / Anforderungen

Folgende Punkte müssen erfüllt sein:

		Bemerkungen / geforderte Eigenschaften
User Management	Login	<ul style="list-style-type: none"> ✗ Validation vom Input (Benutzername / Passwort required und länger als drei Zeichen).
		<ul style="list-style-type: none"> • Login muss auf dem vom Server gelieferten JWT Token basieren.
		<ul style="list-style-type: none"> ✗ Das JWT Token im Local-Storage ablegen. ✗ Das erfolgreiche Login endet auf dem Dashboard.
	Registrierung	<ul style="list-style-type: none"> ✗ Validation von Vornamen, Nachnamen, Benutzernamen / Passwort / <u>analog Login</u>.
		<ul style="list-style-type: none"> • Passwort Confirmation Client-Seitig überprüfen. ✗ User wird auf dem Server angelegt und automatisch eingelogged.
Transaction Management	Logout	<ul style="list-style-type: none"> • Löscht das Client-seitig gespeicherte JWT Token. ✗ Benutzer befindet sich nach Logout auf der Welcome-Page.
		<ul style="list-style-type: none"> • Das JWT Token aus dem Local-Storage löschen.
	Zahlung auslösen	<ul style="list-style-type: none"> • Neue Zahlungen auf dem Dashboard auslösen. • Gültige Zahlungsempfänger (Vorname / Nachname) automatisch vom Server abfragen. • Validation vom Input (nur Beträge > 0.05 CHF erlaubt). • Vom Server bestätigte Transaktionen mittels einer „Transaction Erfolgreich“ -Info abschliessen.
	Zahlungs-übersicht darstellen	<ul style="list-style-type: none"> • Die letzten drei Transaktionen auf dem Dashboard darstellen. • Möglichkeit, auf die Ansicht mit sämtlichen Transaktionen zu wechseln. • Nach einer Transaktion soll diese in der Übersicht dargestellt werden.
	Alle Zahlungen darstellen und filtern	<ul style="list-style-type: none"> • Zahlungen nach Monaten filtern. • Filtermöglichkeit nach Monat des entsprechenden Jahres (die letzten drei Jahre auflisten). • Den Filter mithilfe der Server-API anwenden.
Einschränkungen	Fluides Design	<ul style="list-style-type: none"> • Die Cards auf dem Dashboard sollen im Verhalten fluid sein. Dazu können Libraries (siehe <i>Sonstiges</i>) verwendet werden.
	Sonstiges	<ul style="list-style-type: none"> • Der vorbereitete Server sowie die Vorlagen (Angular 2 / React spezifisch) sollen verwendet werden. • CSS Libraries (z.B. Bootstrap) dürfen eingesetzt werden. Diese müssen sich aber auf reines CSS beschränken oder explizit in der Vorlesung behandelt werden. • Vermeiden Sie Copy-Paste Code. Verwenden Sie Komponenten um den UI Code zu reduzieren und Services für wiederverwendbare Programm-Logik.