

Web-Technologien-Projekt: TwitterLITE Dokumentation

Erdem Koc Matr.: xxxxxxxx

Ludwig Bergemann Matr.: xxxxxxxx

10.07.2020

Inhaltsverzeichnis

1	Einleitung.....	1
2	Vorstellung und Ergebnis	1
2.1	Funktionen und Use-Cases	1
2.1.1	Use Cases, die hinzugekommen sind oder erfüllte NiceToHave's	2
2.1.2	Use Cases, die verworfen wurden	2
2.1.3	Use Cases, die geändert wurden	3
2.2	Aktivitätsdiagramm	4
2.3	Datenbank	5
2.3.1	Aufbau	5
2.3.2	Trigger.....	6
2.4	Design	6
2.4.1	Login Page	6
2.4.2	Startseite	7
2.4.3	User Profil.....	8
2.5	Herausforderungen	8
2.6	Bewertung	9
2.7	Verteilung	9

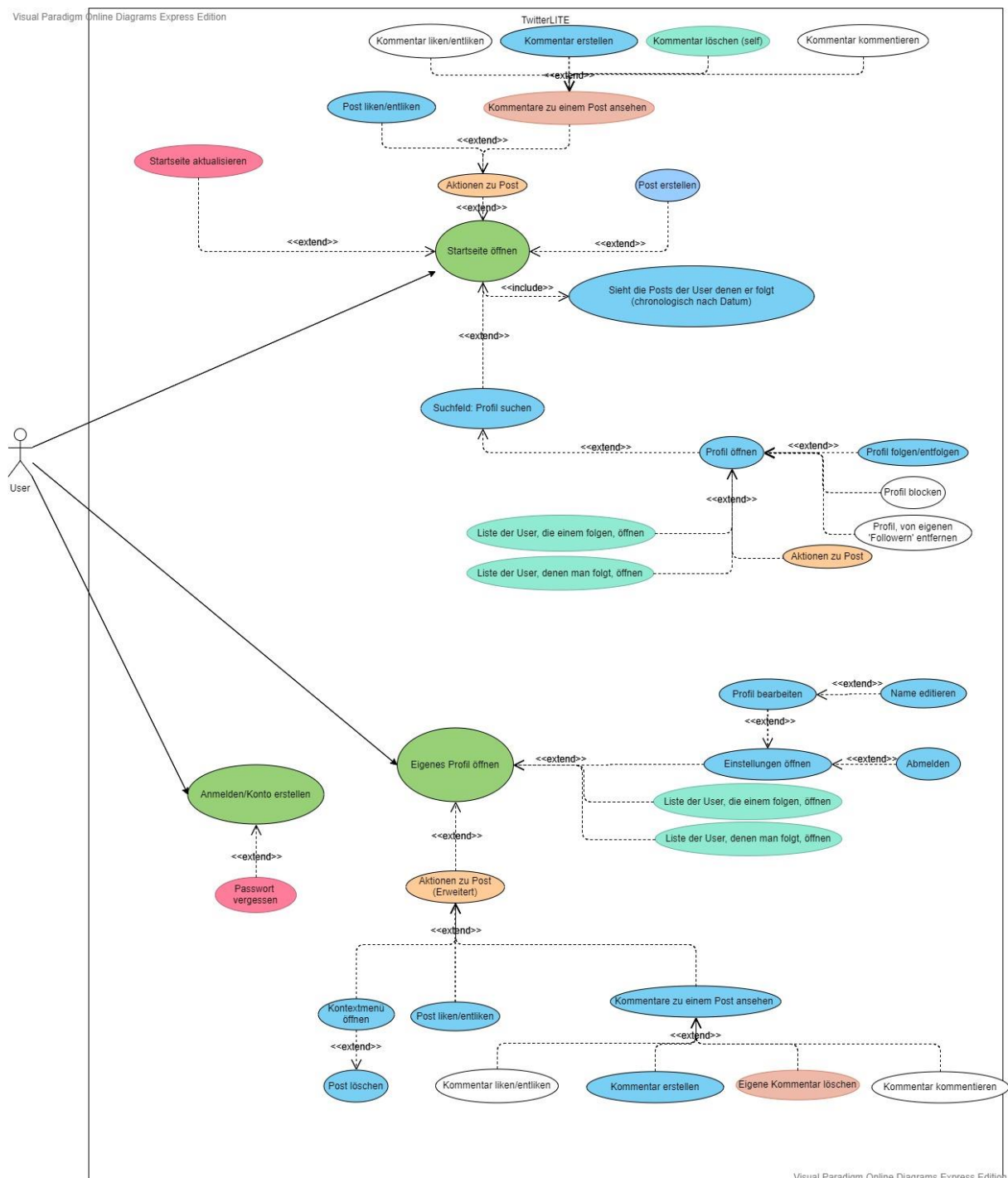
1 Einleitung

Das Projekt ist eine abgespeckte Version der bekannten sozialen Plattform „Twitter“.

Personen können sich miteinander vernetzen und mit ihren Followern Kurznachrichten teilen, welche geliked und kommentiert werden können.

2 Vorstellung und Ergebnis

2.1 Funktionen und Use-Cases



Legende:

- Grün: Hauptanwendungen
- Blau: Anwendungen (MustHave)
- Orange: Anwendungsgruppen
- Hellgrün: hinzugekommene Anwendungen oder erfüllte NiceToHave-Anwendungen
- Hellrot: geänderte Anwendungen
- Rot: verworfene MustHave-Anwendungen
- Weiß: weggelassene Anwendungen (NiceToHave)

2.1.1 Use Cases, die hinzugekommen sind oder erfüllte NiceToHave's

Die Use Cases, bei denen die Umsetzung oder der zeitliche Aufwand noch nicht ganz klar war, haben wir als NiceToHave deklariert.

- Use Case „Kommentar löschen (self)“: Diese Funktion hatten wir prinzipiell schon mit „Post löschen“ verbaut, wodurch uns die Implementierung nicht allzu viel Zeit kostete.
- Use Case „Liste der User, die einem folgen/denen man folgt, öffnen“: Da für uns die Umsetzung dieses Use Cases noch nicht eindeutig war (PopUpWindow, kleines Fenster auf der gleichen Seite, etc.), wollten wir uns nicht festlegen und es noch offen lassen. Letztendlich war die einfachste und sinnvollste Lösung auf einen neuen Tab zu verlinken.

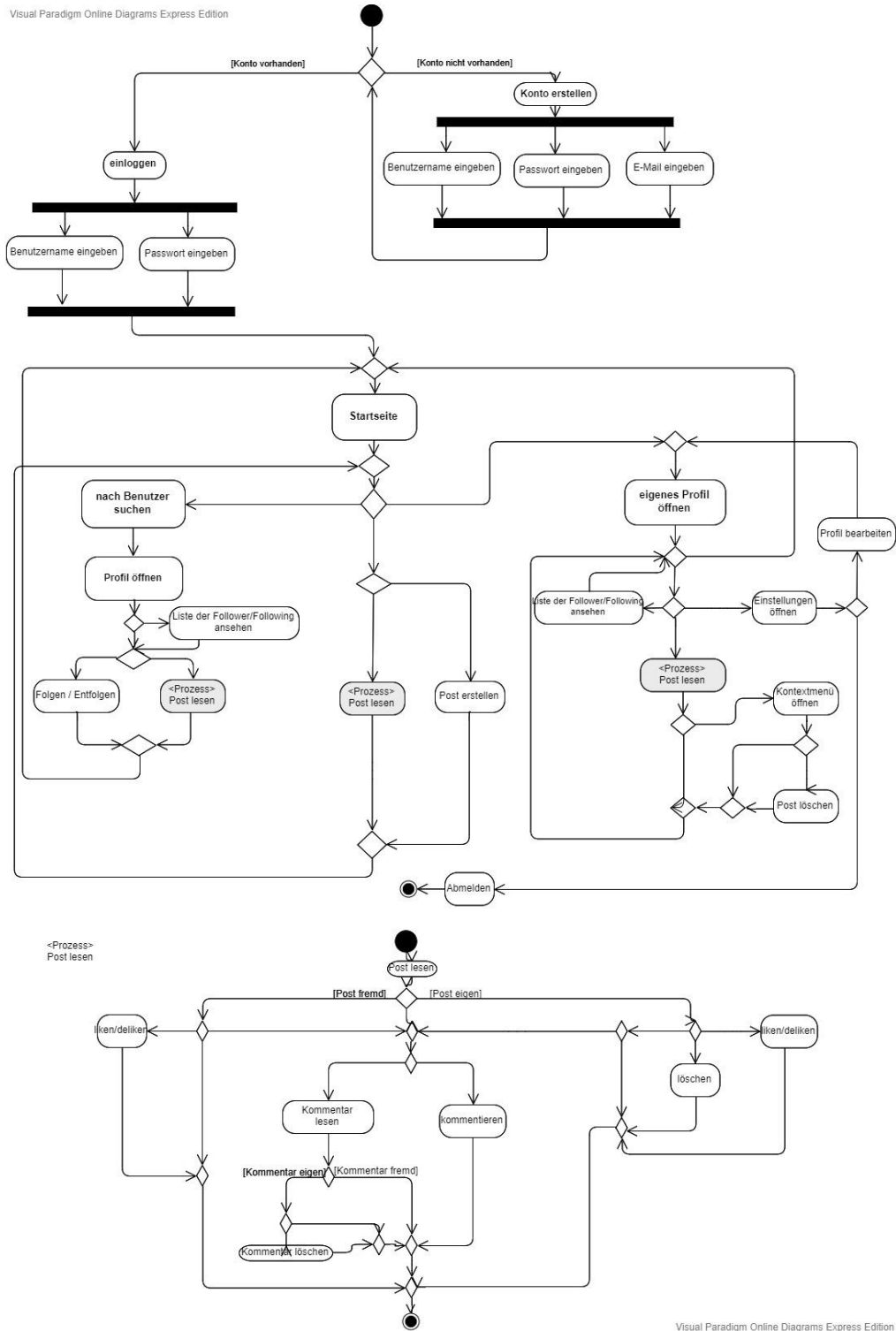
2.1.2 Use Cases, die verworfen wurden

- Use Case „Passwort vergessen“: Die Implementierung eines E-Mail-Systems, welches automatisch dem User die Möglichkeit bieten sollte sein Passwort zurückzusetzen, war dabei unsere Mindestvorstellung, was sich dann aber als zu arbeitsaufwendig und zu zeitintensiv zeigte. Eine einfachere Version kam für uns nicht in Frage.
- „Startseite aktualisieren“- Button: Der Button war ursprünglich dazu da, die auf der Startseite ausgegebenen Posts mit den aktualisierten Posts zu ersetzen, ohne die Seite neu laden zu müssen. Das Problem war das Ersetzen, die Posts wurden alle ein weiteres Mal unter die bestehenden Posts ausgegeben. Zum Lösen des Problems hätte man zahlreiche Java-Script-Funktionen einbauen und einen Teil des Codes neu ordnen müssen und, da der User auch so die Startseite einfach neu laden kann, entschieden wir uns dafür diesen zu verwerfen.

2.1.3 Use Cases, die geändert wurden

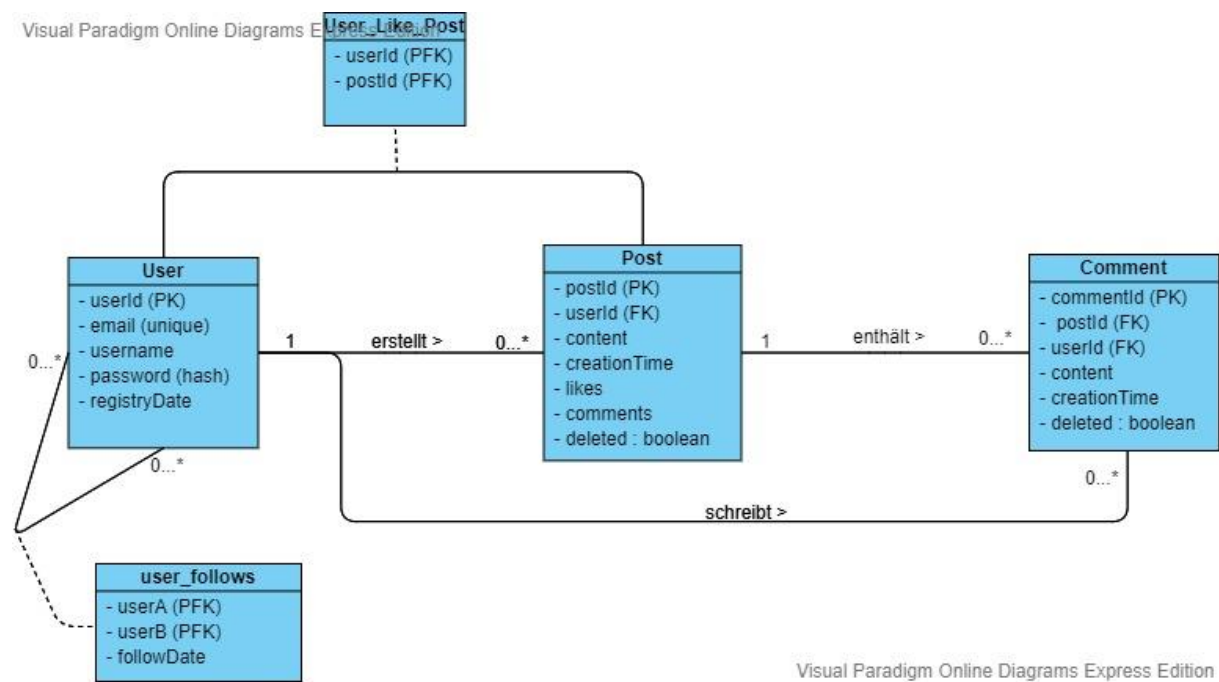
- Use Case „Kommentare zu einem Post ansehen“: Unter diesem Use Case haben wir uns einen Button vorgestellt, mit dem der User die Kommentare zu einem Post nach Belieben ein- oder ausklappen kann. Dieses Use-Case sahen wir mehr als eine spielerische Animation, und weniger als eine elementare Funktionalität, von der uns noch weitere gefehlt haben, weshalb wird das Ein-/Aufklappen erstmal aufgeschoben hatten. Da der User trotzdem die Möglichkeit hat, sich Kommentare anzusehen und die Animation ein zu großer Aufwand gewesen wäre, haben wir den Button letztendlich verworfen.
- Use-Case „Eigene Kommentar löschen“: Mit diesem Use-Case war erstens das Löschen der eigenen Kommentare und zweitens das Löschen von fremden Kommentaren unter den eigenen Posts gemeint. Den zweiten Punkt haben wir noch als NiceToHave offen gelassen, ihn könnte man noch einbauen.

2.2 Aktivitätsdiagramm



2.3 Datenbank

2.3.1 Aufbau



Dies ist das Klassendiagramm, worauf unsere Web-Anwendung basiert.

Da möglichst wenige Daten gelöscht werden sollen, haben wir einigen Relationen das Attribut „deleted“ hinzugefügt, welches entsprechend auf ‚True‘ oder ‚False‘ gesetzt wird.

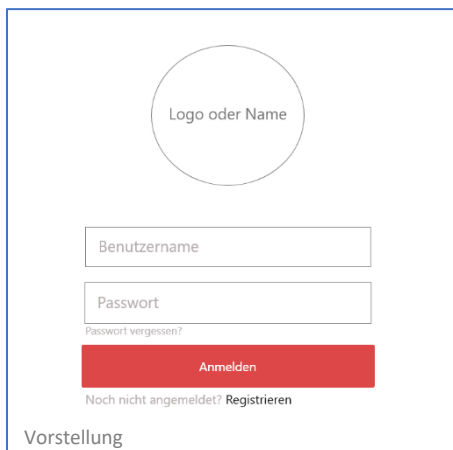
2.3.2 Trigger

- „user_like_post“ und „Post“: Wenn ein User einen Post liked/deliked wird dieser Prozess in user_like_post erfasst/entfernt, woraufhin in der Relation ‚Post‘ das Attribut ‚likes‘ dementsprechend aktualisiert wird.
- „Comment“ und „Post“: Wenn ein User einen neuen Kommentar zu einem Post verfasst/löscht (deleted true/false), wird in der Relation ‚Post‘ das Attribut ‚comments‘ dementsprechend aktualisiert.
Kommtar wird geshirbren oder gelöscht wird gezählt
- Wenn Post von User geliked wird

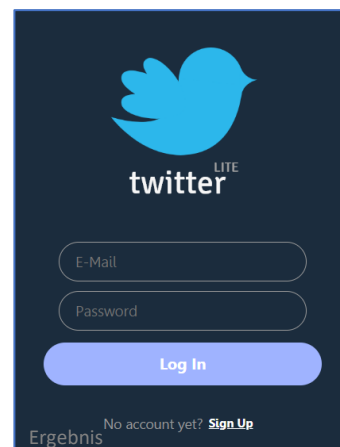
2.4 Design

Von Anfang an hatten wir eine genaue Vorstellung vom Layout, jedoch nicht von dem Design (color, fontstyle, etc.) der Webseite, weshalb wir flexibel bleiben wollten und uns das Design offen gehalten haben. Im Laufe des Entwicklungsprozesses orientierten wir uns immer mehr am Original.

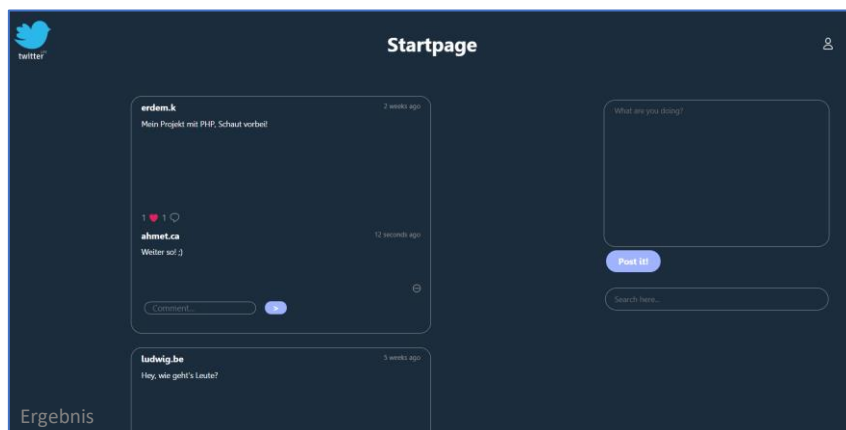
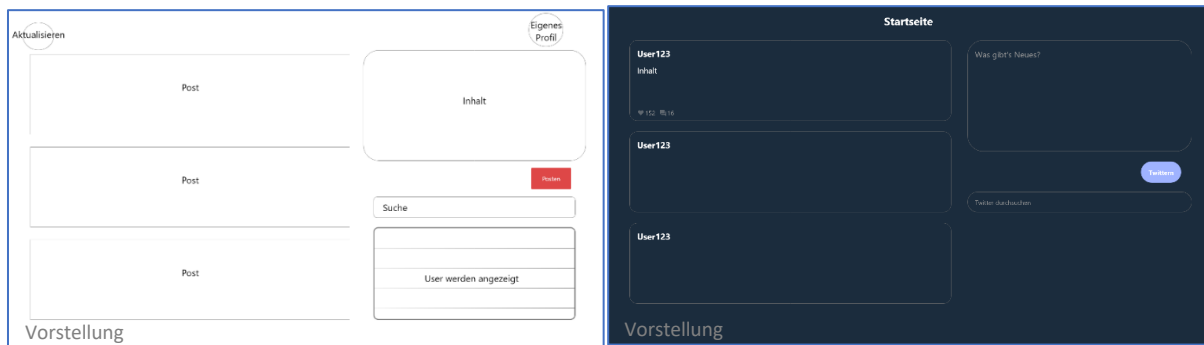
2.4.1 Login Page



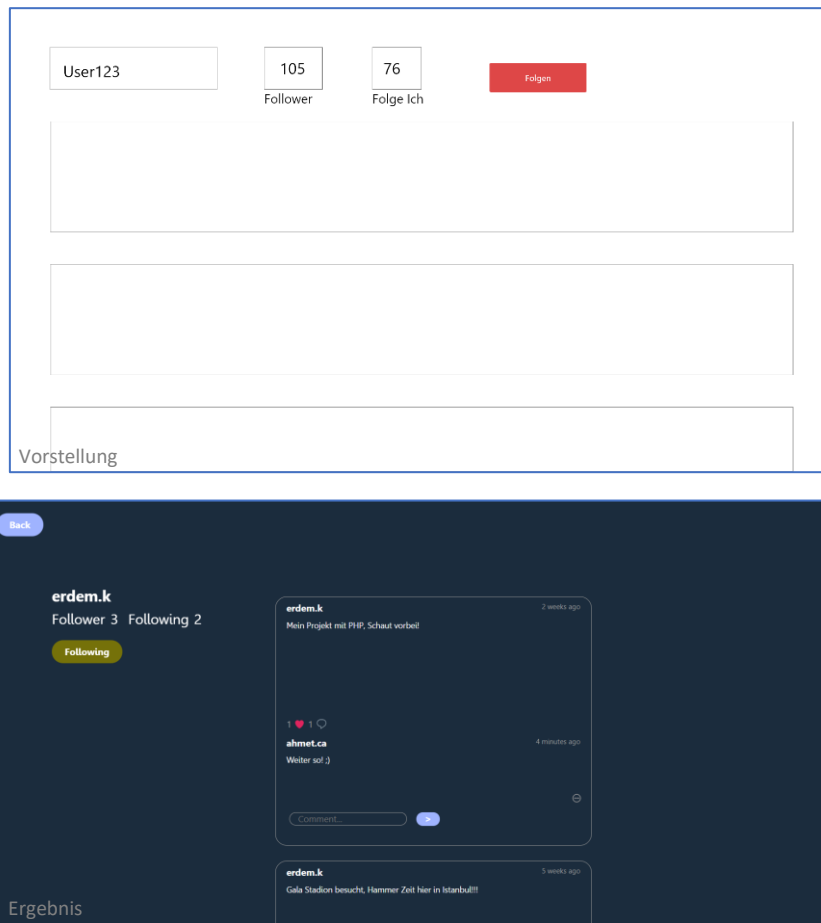
A wireframe diagram of a login page. At the top is a circular placeholder labeled "Logo oder Name". Below it are two rectangular input fields: "Benutzername" and "Passwort". Under the password field is a small link "Passwort vergessen?". Below these is a red rectangular button labeled "Anmelden". At the bottom left is a link "Noch nicht angemeldet? Registrieren". The entire wireframe is labeled "Vorstellung" at the bottom left.



2.4.2 Startseite



2.4.3 User Profil



2.5 Herausforderungen

- Die Kombination und das Verpacken der verschiedenen Sprachen (Php, JavaScript, Html, Css) in ein großes Ganzes hat manchmal zu Unverständnis geführt (z.B. die Verknüpfung von Html-Buttons mit Php-Funktionen)
- Der „Startseite aktualisieren“- Button hat zu Problemen geführt (siehe oben).
- Das Use Case „Kommentare zu einem Post ansehen“, also das Ein- und Ausklappen von Kommentaren wäre ein zu großer Aufwand gewesen (siehe oben).
- Das Design: Die Auswahl von Farben und Icons etc. war unproblematisch, da wir durch das Original eine anschauliche Orientierung hatten, allerdings war das Positionieren der verschiedenen Elemente sehr zeit- und nervenkostend.
- Scroll Position: Nach einigen User-Interaktionen, bei denen die Seite neu geladen wird, wird die Scroll Position nicht mehr gespeichert, wodurch der User nach z.B. einem Like wieder oben landet. Durch die JQuery Bibliothek hatten wir gegen Ende zwar einen Lösungsansatz parat, jedoch hätte dieser den Zeitrahmen gesprengt.
- Beim Arbeiten mit den verschiedenen Id's der einzelnen Html-Elemente (z.B. Post liken) musste man einzelne Schritte stark durchdenken.

Durch langes Grübeln, Recherchieren (stackoverflow.com, youtube.com), Testen und gegenseitige Unterstützung haben sich die meisten Probleme lösen lassen.

2.6 Bewertung

Wir bewerten unser Ergebnis als relativ in Ordnung, da wir die meisten unserer vorgenommenen Ziele erfüllt haben und die Seite ein modernes, reduziertes und ansprechendes Erscheinungsbild hat.

Eine Funktion, die für die User-Erfahrung noch sehr wichtig wäre, ist das Problem mit der Scroll Position (siehe oben). Ansonsten könnte man noch andere zahlreiche Funktionen vom Original einbauen (z.B. Retweet/Repost, Blockieren von Usern, Profilbild, andere Medienformate, Privatnachrichten etc.). Da der Wert von Sozialen Netzwerken in Abhängigkeit von der Anzahl der Benutzer wächst, ist es schwer und unratsam sich so früh nach der Entwicklung auf eine Preisspanne festzulegen.

2.7 Verteilung

Zu Beginn haben wir die einzelnen Arbeitsaufgaben präzise aufgeteilt. Da es für uns beide das erste Web-Projekt war, kam es oft zum Tauschen von Aufgaben und zu gegenseitiger Unterstützung, weswegen sich die Aufgabenverteilung nicht genau abgrenzen lässt. Alles in allem war die Arbeitsverteilung ausgeglichen, obwohl Ludwig gelegentlich etwas größeren Einsatz gezeigt hat