LinkUp

Social-Media-App

Ein Bild, das Multimedia, Software, Screenshot, Computer enthält.

Automatisch generierte Beschreibung

|  |  |
| --- | --- |
| Autor | Simon Streuli |
| Erstelldatum | **05.06.2023** |
| Abgabedatum | **xx.xx.xx** |

Inhaltsverzeichnis

[Änderungstabelle 4](#_Toc138771118)

[1. Informieren 5](#_Toc138771119)

[1.1 Idee 5](#_Toc138771120)

[1.2 Ziele 5](#_Toc138771121)

[1.3 Technologien 6](#_Toc138771122)

[2. Planen 8](#_Toc138771123)

[2.1 Arbeitspakete 8](#_Toc138771124)

[2.2 Allfällige Risiken 9](#_Toc138771125)

[2.3 Use Cases 9](#_Toc138771126)

[2.3 Kanban-Board 10](#_Toc138771127)

[2.4 Zeitplanung 11](#_Toc138771128)

[2.5 Wireframe/UX, UI-Design 12](#_Toc138771129)

[2.6 Flow Chart 14](#_Toc138771130)

[2.7 Klassendiagramm 15](#_Toc138771131)

[2.8 Name & Logo 16](#_Toc138771132)

[3. Entscheiden 17](#_Toc138771133)

[3.1 Technologien 17](#_Toc138771134)

[3.2 Datenbank 17](#_Toc138771135)

[4. Realisieren 18](#_Toc138771136)

[4.1 Backend 18](#_Toc138771137)

[4.2 Datenbank 19](#_Toc138771138)

[4.3 Frontend 20](#_Toc138771139)

[5. Kontrollieren 22](#_Toc138771140)

[5.1 Testprotokoll 22](#_Toc138771141)

[6. Factsheet 27](#_Toc138771142)

[6.1 Factsheet 27](#_Toc138771143)

[7. Auswerten 28](#_Toc138771144)

[6.1 Reflexion 28](#_Toc138771145)

[8. Quellen 28](#_Toc138771146)

[8.2 Quellen 28](#_Toc138771147)

[8.3 Bildquellen 29](#_Toc138771148)

# Änderungstabelle

|  |  |  |  |
| --- | --- | --- | --- |
| Datum | Wer | Was | Checked |
| 05.06.2023 | Simon Streuli | Initialdokument erstellt |  |
| 05.06.2023 | Simon Streuli | Informieren und Planen erstellt |  |
| 06.06.2023 | Simon Streuli | Wireframe und Logo & Name Abschnitt hinzugefügt |  |
| 13.06.2023 | Simon Streuli | Entscheiden & Realisieren Abschnitt angefangen. |  |
| 21.06.2023 | Simon Streuli | Realisieren weiter gemacht |  |
| 26.06.2023 | Simon Streuli | Kapitel Frontend & Backend geschrieben. |  |
| 27.06.2023 | Simon Streuli | Prüfprotokoll hinzugefügt. |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 1. Informieren

### 1.1 Idee

Das Konzept besteht darin, eine soziale Medienplattform zu entwickeln. Die Anwendung soll es Benutzern ermöglichen, sich zu registrieren, Inhalte zu teilen, Inhalte zu kommentieren, anderen Benutzern zu folgen. Die App soll eine benutzerfreundliche Oberfläche besitzen und es Benutzern ermöglichen, sich zu vernetzten. Sie soll sowie auf Desktopgeräten und auch Mobilgeräten verfügbar sein.

### 1.2 Ziele

|  |  |
| --- | --- |
| Projektziele (Erwartete Resultate. Forderungen und Wünsche) | Prior.1 |
| * Entwicklung einer übersichtlichen und einfachen Benutzeroberfläche, die es Benutzern ermöglicht, sich miteinander zu vernetzten. | A |
| * Benutzern soll es möglich sein, Beiträge zu erstellen, Benutzern zu folgen, Beiträge zu liken und zu kommentieren. | A |
| * Es soll möglich sein, sich sicher zu registrieren und sich sicher anzumelden. | A |
| * Jeder Benutzer soll eine Einzelansicht zu seinem Profil haben. | B |
| * Benutzer sollen untereinander Chatten können. | C |
| * Es soll eine Startseite geben, bei der man die Posts von anderen Nutzern sehen kann. | A |

|  |  |
| --- | --- |
| Vorgehensziele (Forderungen und Rahmenbedingungen) | Prior.1 |
| IPERKA anwenden | B |
| Ausführlich und sauber Dokumentieren | A |
| UI/UX-Design lernen und anwenden | B |
| Frontend/Backend Kenntnisse vertiefen, neue Technologien kennenlernen (React.js) | A |
| Testing und Fehlerbehebung, um Funktionalität und Stabilität der App sicherzustellen | B |

|  |  |
| --- | --- |
| Lernziele (Welche Kompetenzen werden, angestrebt?) | Prior.1 |
| Vertiefte Kenntnisse in der Entwicklung von Webanwendungen | A |
| Node.js Kenntnisse erweitern | A |
| Sicherheitsaspekte in Webanwendungen verstehen und anwenden | C |
| HTML, CSS, JavaScript-Fähigkeiten erweitern, allenfalls React.js lernen | B |
| SQL oder MongoDB in Webanwendung integrieren | B |
| Projektmanagement und Organisationsfähigkeit | A |
| Test- und Fehlerbehebungskompetenz ausbauen | B |
| Präsentationskompetenz verbessern | B |

|  |  |  |
| --- | --- | --- |
| 1) A = Must have | B = Starkes Wunschziel, wenn möglich | C = “Nice to have“ |
|  |  |  |

### 1.3 Technologien

**Entwicklungsumgebung**

Es gibt viele verschiedene **IDE's**, die für die Entwicklung von Software in verschiedenen Programmiersprachen verwendet werden können. Einige der beliebtesten Optionen sind:

* Visual Studio: Eine IDE von Microsoft, die für die Entwicklung von Anwendungen in verschiedenen Sprachen verwendet werden kann, darunter C#, C++, Visual Basic und F#.
* Eclipse: Eine freie und quelloffene IDE, die für die Entwicklung von Anwendungen in verschiedenen Sprachen verwendet werden kann, darunter Java, C++ und PHP.
* IntelliJ IDEA: Eine kommerzielle IDE von JetBrains, die für die Entwicklung von Anwendungen in verschiedenen Sprachen verwendet werden kann, darunter Java, Kotlin, Python und Ruby.
* Xcode: Eine IDE von Apple, die für die Entwicklung von Anwendungen für das iOS- und macOS-Betriebssystem verwendet wird.

**Frontend**

Es gibt viele verschiedene Technologien und Frameworks, die für die Entwicklung von Frontend-Anwendungen verwendet werden können. Einige der beliebtesten Optionen sind:

* HTML: Eine Markup-Sprache, die verwendet wird, um die Struktur und den Inhalt von Webseiten zu definieren.
* React: Ein JavaScript-Framework, das von Facebook entwickelt wurde und häufig für die Entwicklung von Webanwendungen verwendet wird.
* Angular: Ein JavaScript-Framework, das von Google entwickelt wurde und häufig für die Entwicklung von Webanwendungen verwendet wird.
* Vue.js: Ein JavaScript-Framework, das für die Entwicklung von Benutzeroberflächen verwendet wird.
* Die Schwierigkeit darin besteht im Aneignen der Sprache.

**Backend**

Es gibt viele verschiedene Programmiersprachen und Frameworks, die für die Entwicklung von Backend-Anwendungen verwendet werden können. Einige der beliebtesten Optionen sind:

* Java: Eine beliebte objektorientierte Programmiersprache, die häufig für die Entwicklung von Backend-Anwendungen verwendet wird.
* Python: Eine leistungsstarke und flexible Programmiersprache, die häufig für Backend-Entwicklung, Datenanalyse und Machine Learning verwendet wird.
* Ruby: Eine objektorientierte Programmiersprache, die häufig für die Entwicklung von Webanwendungen verwendet wird.
* PHP: Eine weit verbreitete Server-Seiten-Programmiersprache, die häufig für die Entwicklung von Webanwendungen verwendet wird.
* .NET: Ein Framework von Microsoft, das für die Entwicklung von Desktop- und Webanwendungen verwendet wird.
* Node.js: Eine Server-Seiten-Plattform, die auf der JavaScript-Programmiersprache basiert und häufig für die Entwicklung von Backend-Anwendungen und Echtzeit-Anwendungen verwendet wird.

# 2. Planen

### 2.1 Arbeitspakete

|  |  |  |
| --- | --- | --- |
| Arbeitspaket | Beschreibung | Zeit |
| Anforderungsanalyse | Definition der Funktionalitäten und Designanforderungen | 2h |
| Informieren über das Thema | Sämmtliche Informationen über Thema sammeln (Programmiersprache, Entwicklungsumgebung, etc.) | 2h |
| Planung | Planen was, wann in der Projektlaufzeit gemacht wird. | 4h |
| Technologieauswahl | Auswahl von Technologien, Frameworks, Bibliotheken und Tools. | 2h |
| UI/UX-Design | Erstellung eines Designkonzepts für die Benutzeroberfläche der App. Erstellung von Wireframes und Prototypen. | 6h |
| Backend-Entwicklung | Aufsetzen der Node.js-Entwicklungsumgebung. Implementierung der Serverlogik für die App-Funktionen. Implementierung und Erstellung der Datenbank. | 20h |
| Frontend-Entwicklung | Umsetzung des UI/UX-Designs in HTML, CSS und JavaScript. Entwicklung der Benutzeroberfläche und Integration von Backend-Schnittstellen. | 20h |
| Datenschutz und Sicherheit | Implementierung von Sicherheitsmassnahmen wie Authentifizierung und Datenschutz | 4h |
| Testing und Fehlerbehebung | Durchführung umfangreicher Tests zur Funktionalität und Stabilität der App. Behebung von Fehlern. | 6h |
| Kontrollieren | Definierte Ziele anschauen und erreichte Ziele abhacken. | 4h |
| Dokumentation | Erstellung einer umfangreichen Dokumentation des Projekts. | 10h |
| Präsentation | Präsentation erstellen, lernen und vor Lehrmeister präsentieren. | 6h |
| Auswerten | Projekt bewerten und reflektieren. | 4h |
| Puffer | Puffer für Probleme oder Verzögerung | 8h |
| **Gesamt** |  | **90h** |

### 2.2 Allfällige Risiken

* Technische Schwierigkeiten beim Implementieren komplexer Funktionalitäten.
* Zeit könnte bei Problemen knapp werden.

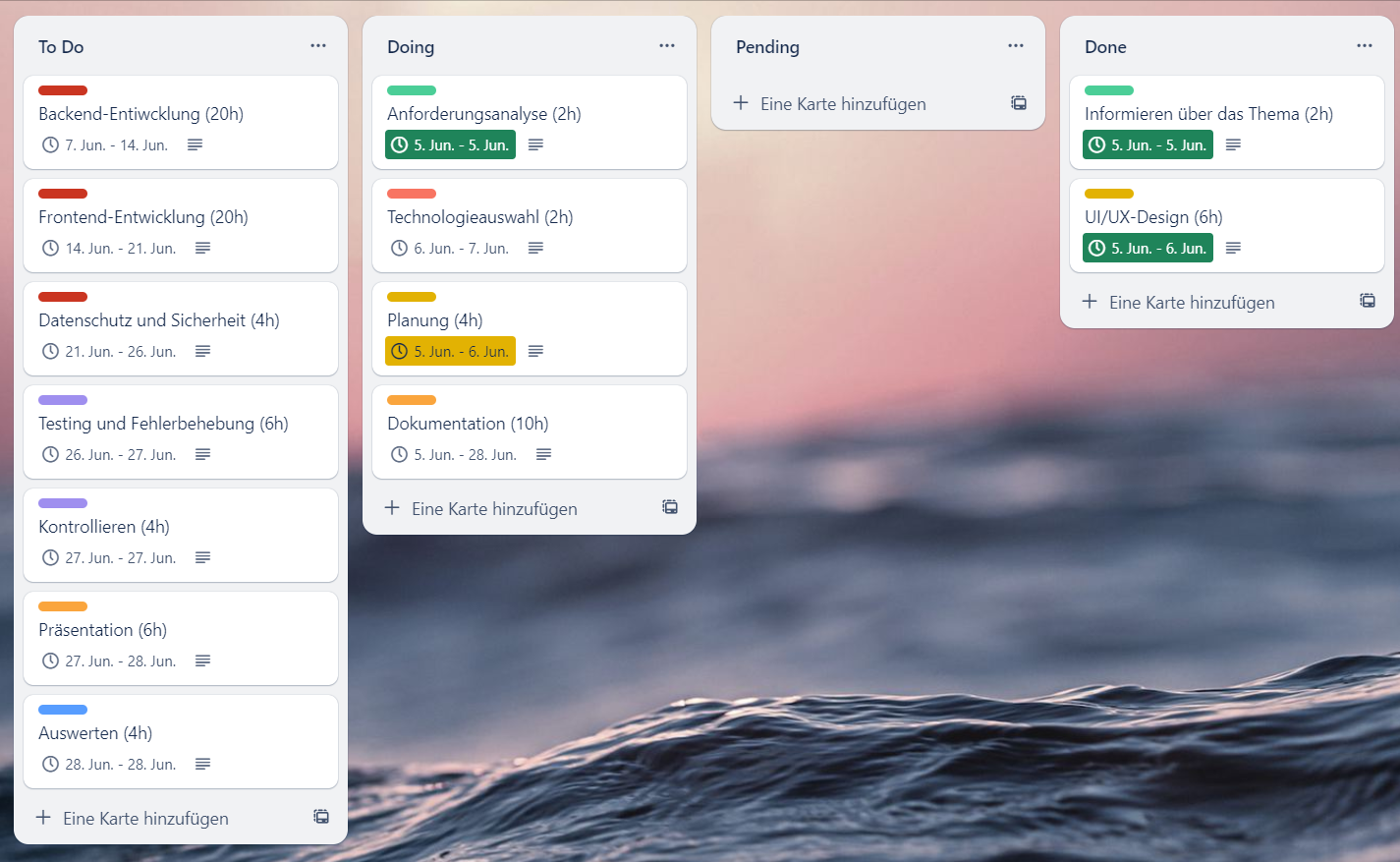
### 2.3 Use Cases

* Als User soll ich mich mit meinen Daten registrieren können.
* Als User soll ich mich Anmelden können.
* Als User soll ich Beiträge von anderen Usern sehen können.
* Als User soll ich Beiträge von anderen Usern liken und kommentieren können.
* Als User soll ich eigene Beiträge erstellen können.
* Als User soll ich anderen Usern folgen können.
* Als User soll ich mein eigenes Profil anschauen.
* Als User soll ich mein Beitrag löschen können.
* Als User soll ich Beiträge teilen können.
* Als User soll ich Benutzer suchen können.

### 2.3 Kanban-Board

Ich trage die Arbeitspakete in ein Kanban-Board ein, da es sinnvoll ist, den Fortschritt übersichtlich zu organisieren. Für die Verwaltung des Kanban-Boards verwende ich die Plattform Trello. Trello ermöglicht eine einfache und visuelle Darstellung der Arbeitspakete, deren Status und Zuweisung an Teammitglieder. Dadurch kann ich effizient arbeiten und den Fortschritt des Projekts im Blick behalten. Ich habe die Aufgaben in verschiedene Gruppen eingeteilt, sodass ich einen besseren Überblick habe, was mit was zusammenhängt.

Kanban-Board Trello



### 2.4 Zeitplanung

Die Zeitplanung habe ich ebenfalls mit Trello realisiert. Ich habe die Arbeitspakete mit ihrem Start- und Enddatum eingetragen. So habe ich einen genauen Überblick über den Projektverlauf.

Ein Bild, das Text, Screenshot, Software, Zahl enthält.

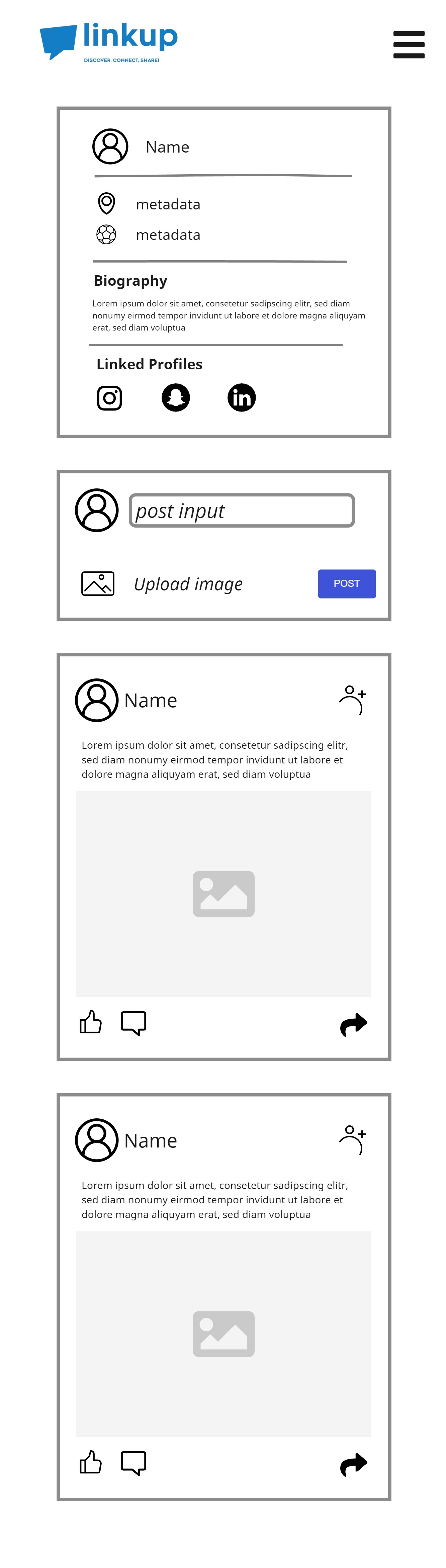
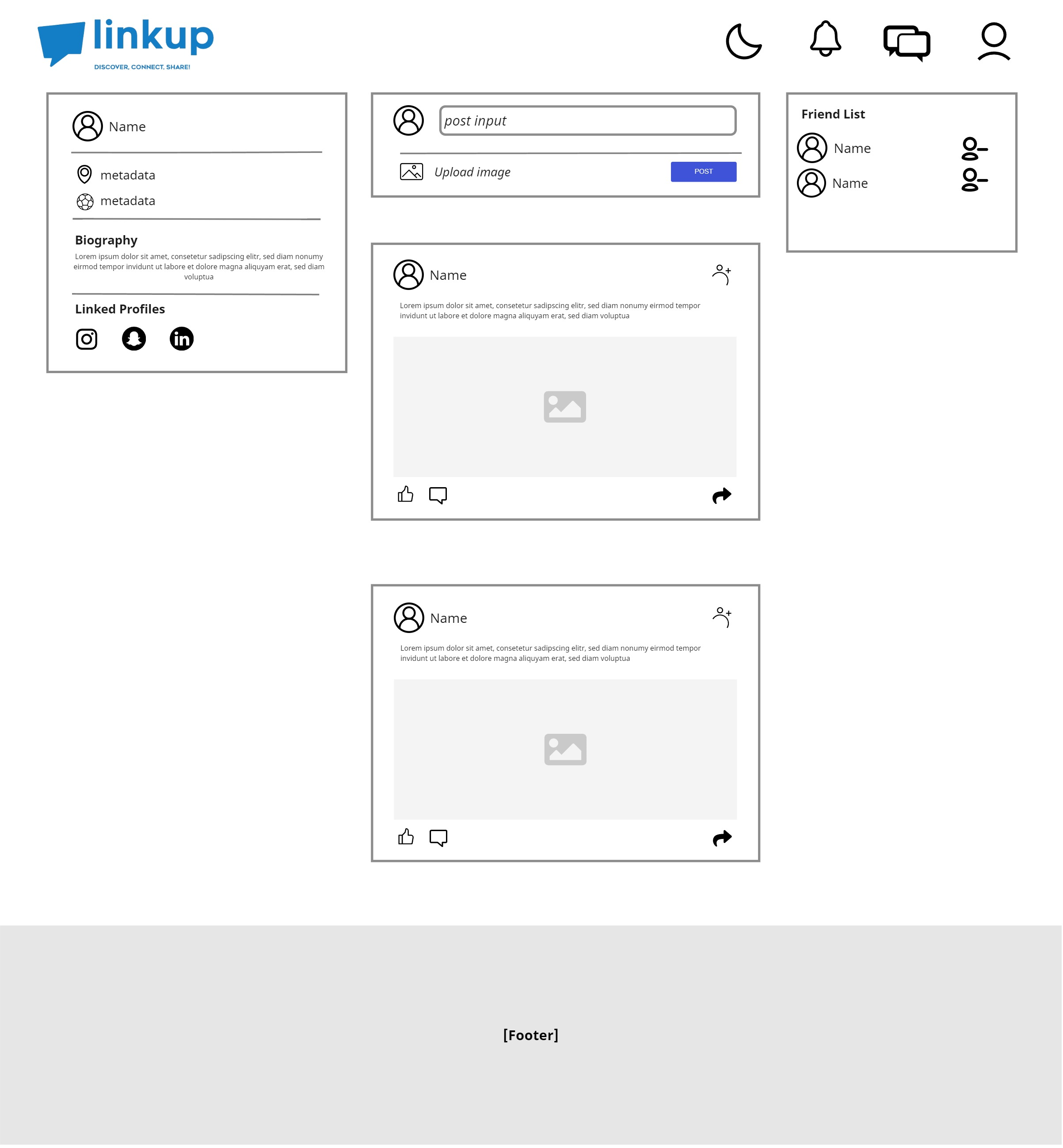
Automatisch generierte Beschreibung

Zeitplanung Trello

### 2.5 Wireframe/UX, UI-Design

Wireframes sind eine wichtige Komponente bei der Entwicklung einer Anwendung und spielen eine entscheidende Rolle in der Phase des UI/UX-Designs. Die Erstellung von Wireframes hat mir dabei geholfen, das grundlegende Layout, die Struktur und die Funktionalität meiner Social Media App zu visualisieren und zu planen.

LinkUp Startseite Desktop

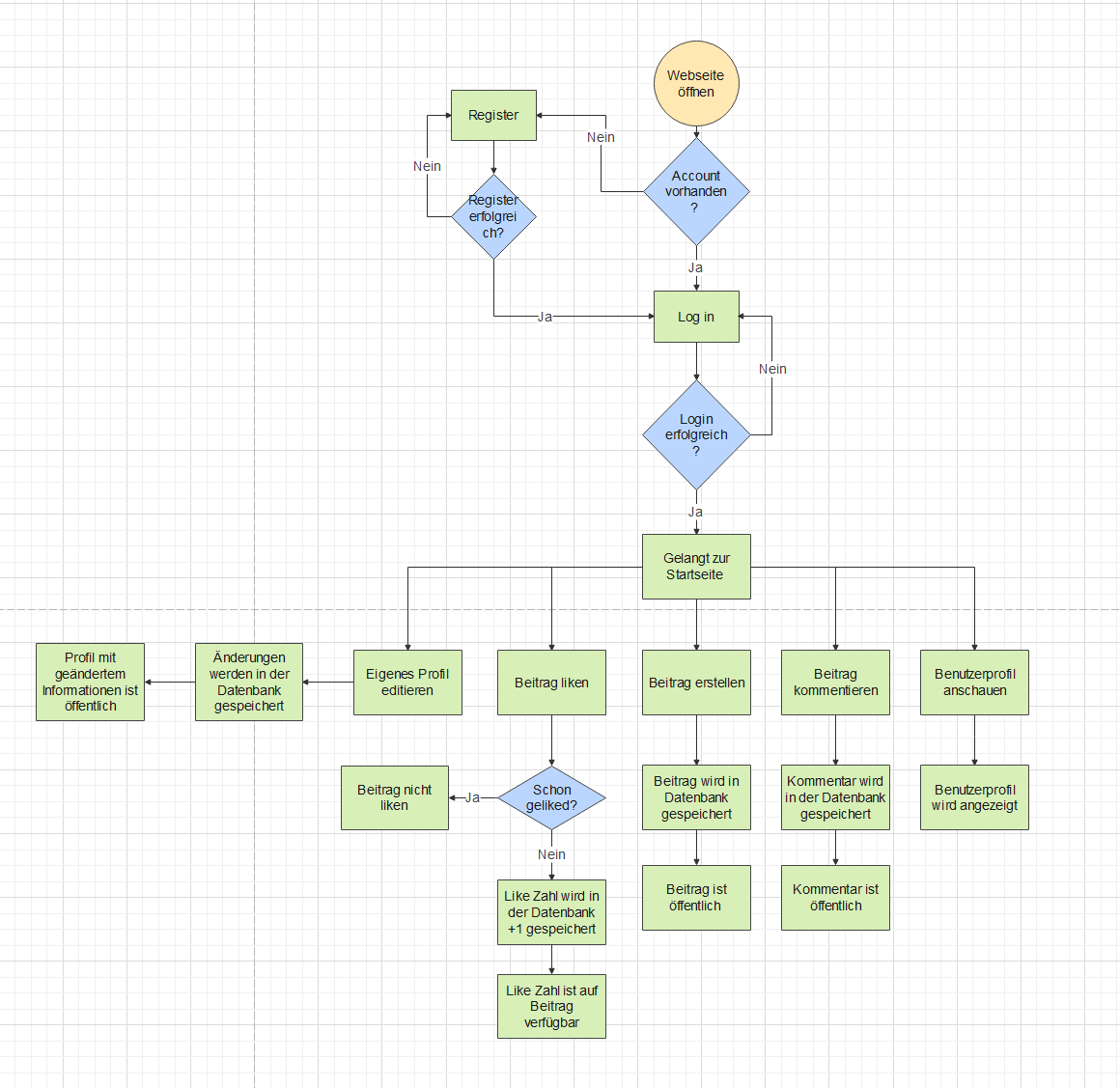


Mobil Version LinkUp Startseite

Mobil-Verison Wireframe

### 2.6 Flow Chart

Das Flowchart ist eine visuelle Darstellung des Ablaufs und der Entscheidungswege innerhalb meiner Social Media App. Es zeigt den Fluss der Benutzerinteraktionen und veranschaulicht, wie verschiedene Funktionen und Prozesse miteinander verbunden sind.



Flow Chart Social Media App

### 2.7 Klassendiagramm

Das Klassendiagramm dient als visuelle Darstellung der Struktur und der Beziehungen zwischen den Klassen in deiner Social Media App. Es bietet einen Überblick über die wichtigsten Klassen und deren Attribute und Methoden.



Klassendiagramm

### 2.8 Name & Logo

Für den Namen der Social Media App habe ich zur Inspiration ChatGPT gefragt und habe mich für den Namen "**LinkUp**" entschieden.

Ich habe mich für den Namen "LinkUp" entschieden, da er verschiedene Aspekte meiner Social Media App widerspiegelt. Der Name setzt sich aus den beiden Wörtern "Link" und "Up" zusammen.

Das Logo habe ich mit Looka erstellt. Ich habe mich für ein schlichtes, blaues Design mit dem Slogan „Discover, Connect, Share!“ entschieden. Das Logo sieht so aus:



LinkUp Logo

Ausserdem habe ich auch ein Icon für die Webseite erstellt. Das Icon sieht so aus:



LinkUp Logo Transparent



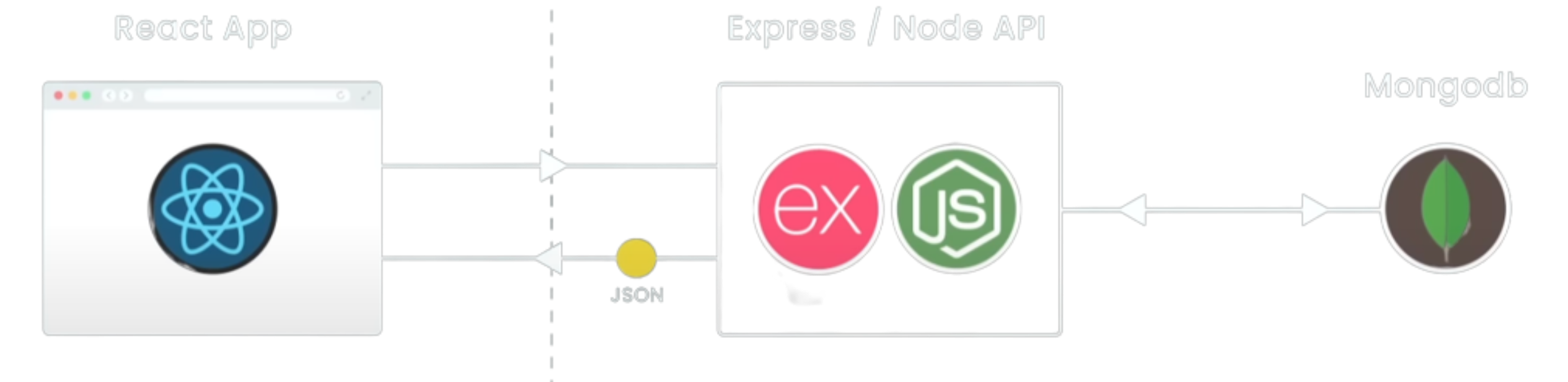
LinkUp Icon

# 3. Entscheiden

### 3.1 Technologien

**MERN-Stack** ist eine beliebte Technologiekombination für die Entwicklung von Webanwendungen, die aus MongoDB, Express.js, React und Node.js besteht. Ich habe mich entschieden, mein Projekt mit dem MERN-Stack zu realisieren, da er eine Reihe von Vorteilen bietet.

Zunächst ermöglicht der MERN-Stack eine effiziente und nahtlose Entwicklung. Mit MongoDB als NoSQL-Datenbank bietet er Flexibilität bei der Speicherung und Abfrage von Daten. Express.js dient als leichtgewichtiger Webframework für die Serverseite und ermöglicht eine einfache Handhabung von HTTP-Anfragen. React, eine leistungsstarke JavaScript-Bibliothek, ermöglicht eine schnelle und reaktive Benutzeroberfläche auf der Clientseite. Node.js, eine JavaScript-Laufzeitumgebung, ermöglicht die Ausführung des Servers und die Kommunikation zwischen Frontend und Backend.



MERN-Stack

### 3.2 Datenbank

Ein Bild, das Grafiken, Schrift, Grafikdesign, Logo enthält.

Automatisch generierte BeschreibungIch habe mich für MongoDB entschieden, da es eine flexible, skalierbare und leistungsstarke Datenbanklösung ist. Die dokumentenorientierte Struktur und die hohe Performance erfüllen meine Anforderungen optimal. Zudem bietet MongoDB eine aktive Community und umfangreiche Unterstützung, was die Entwicklung meines Projekts erleichtert. Ausserdem möchte ich Erfahrung mit einer Nicht-SQL Datenbank sammeln.

MongoDB

# 4. Realisieren

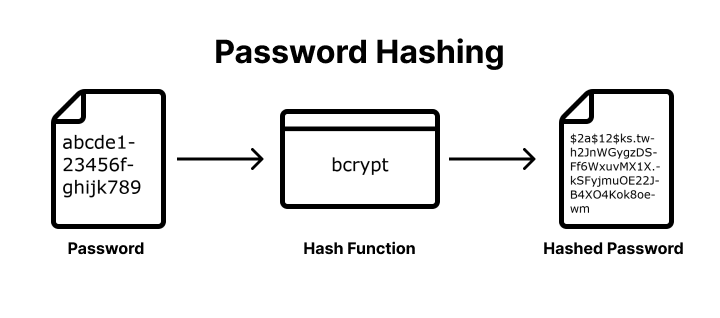
### 4.1 Backend

Um das Backend zu realisieren, habe ich Express.js verwendet, dass ein beliebtes Node.js-Framework ist. Express.js ermöglicht das Erstellen von APIs und das Handhaben von HTTP-Anfragen. Mit Express.js können Routen definiert werden, um verschiedene Endpunkte anzusprechen und die entsprechenden Aktionen auszuführen.

Ich habe Routes für User, Posts und die Authentifizierung erstellt. Bei den User und Posts Routes habe ich **CRUD-Methoden** implementiert, um User zu erstellen, User zu updaten und User zu löschen.

In meinem Projekt habe ich bcrypt verwendet, um die Sicherheit der Passwörter zu gewährleisten. bcrypt ist eine bewährte Methode zur Hashung und Überprüfung von Passwörtern. Durch die Verwendung von bcrypt werden die Passwörter meiner Benutzer sicher gehasht, sodass sie nicht im Klartext gespeichert werden.

Ein weiterer Vorteil von bcrypt ist die Verwendung von Salting. Beim Hashen wird automatisch ein zufälliger Salz-Wert generiert und mit dem Passwort kombiniert. Dadurch wird die Sicherheit weiter erhöht, da es schwieriger wird, die gehashten Passwörter durch vorberechnete Hash-Tabellen oder Brute-Force-Angriffe zu entschlüsseln.



Passwort Hashing

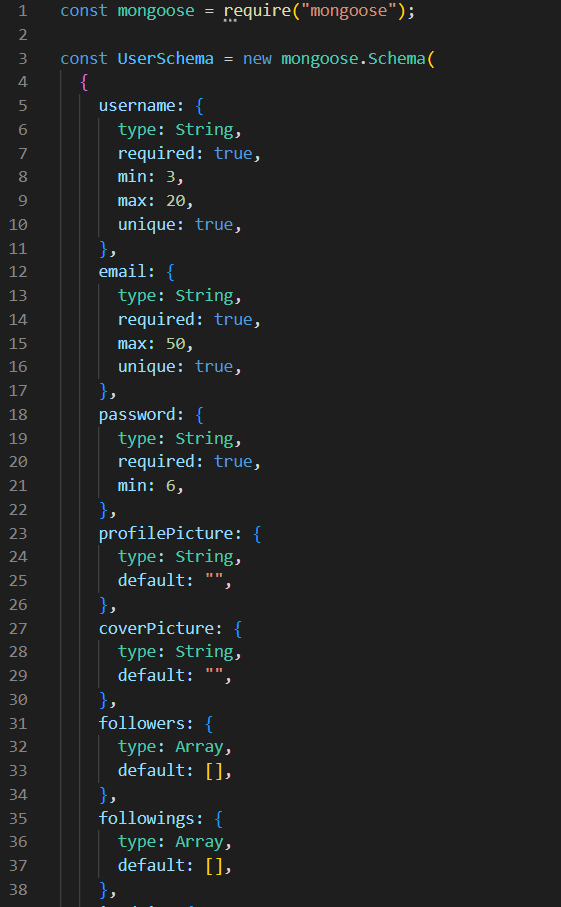
### 4.2 Datenbank

Für die Datenbank habe ich MongoDB verwendet. Ich bin auf die offizielle MongoDB Seite gegangen und habe ein Projekt erstellt, mit dem gratis Speicher. Danach habe ich geschaut, wie ich mich mit meiner Datenbank connecte das sieht so aus:  
Die Datenbank URL und der Port habe ich in das .env File geschrieben. Da meine Datenbank nicht öffentlich sein soll und man den Port beliebig anpassen können soll.



Datenbank Connection

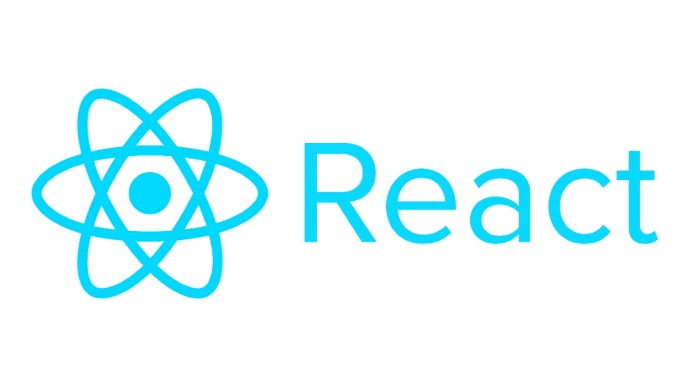
Ausserdem habe ich die Datenstruktur für User und Posts erstellt mit Models. Das sieht in etwa so aus:



User Modell

### 4.3 Frontend

Für die Entwicklung des Frontends in meinem Projekt habe ich moderne Webtechnologien verwendet. Ich habe mich für das React-Framework entschieden, da es eine leistungsstarke und flexible Lösung zur Erstellung von Benutzeroberflächen bietet.



React.js

Mit React konnte ich mein Frontend Komponenten aufteilen, um eine saubere und wiederverwendbare Codebasis zu schaffen. Ich habe JSX (JavaScript XML) verwendet, um die Komponenten zu erstellen und die Logik und das Design zu integrieren. Ich habe folgende Komponenten erstellt:

* Explorefeed
* Feed
* Navbar
* Post
* Share

Diese Komponenten habe ich in verschiedenen Pages verwendet. Ich habe folgende Pages:

* Home
* Explore
* Login
* Register
* Profile

Zur Verwaltung des Zustands meiner Anwendung habe ich den Context API von React genutzt. Dadurch konnte ich Daten zentral speichern und über verschiedene Komponenten hinweg darauf zugreifen. Die Verwendung des Context API hat mir geholfen, den Zustand meiner Anwendung effizient zu verwalten und die Kommunikation zwischen den Komponenten zu erleichtern.

Um mit dem Backend zu kommunizieren und Daten von den APIs abzurufen, habe ich das Axios-Paket verwendet. Mit Axios konnte ich HTTP-Anfragen an meine Backend-Server senden und die entsprechenden Daten verarbeiten.  
  
Ein Bild, das Schrift, Grafiken, Logo, Text enthält.

Automatisch generierte Beschreibung

Axios React

# 5. Kontrollieren

### 5.1 Testprotokoll

#### 5.11 Backend

|  |  |
| --- | --- |
| **Testfall Nr.** | **1** |
| Beschreibung | Im Backend mittels POST Request User registrieren. |
| Voraussetzungen | Backend gestartet und am Laufen. |
| Testschritte | * Öffne Postman und erstelle einen POST Request mit der URL <http://localhost:8800/api/auth/register> * Füge im Request-Body die folgenden Daten hinzu: * username: [Test-Benutzername] * email: [Test-E-Mail] * password: [Test-Passwort] * Request absenden |
| Erwartetes Ergebnis | Der Server gibt den Statuscode 200 zurück und sendet die Daten des erstellten Benutzers als Response. |
| Testergebnis | erfüllt |
| Testdatum | 27.06.2023 |
| Bemerkungen | - |

|  |  |
| --- | --- |
| **Testfall Nr.** | **2** |
| Beschreibung | Im Backend mit POST Request einloggen. |
| Voraussetzungen | Backend gestartet und am Laufen. |
| Testschritte | * Öffne Postman und erstelle einen POST Request mit der URL <http://localhost:8800/api/auth/login> * Füge im Request-Body die folgenden Daten hinzu: * email: [Test-E-Mail] * password: [Test-Passwort] * Request absenden |
| Erwartetes Ergebnis | Der Server gibt den Statuscode 200 zurück und sendet die Benutzerdaten als Response. |
| Testergebnis | erfüllt |
| Testdatum | 27.06.2023 |
| Bemerkungen | - |

|  |  |
| --- | --- |
| **Testfall Nr.** | **3** |
| Beschreibung | Im Backend mit einem PUT Request Benutzerdaten aktualisieren. |
| Voraussetzungen | Backend gestartet und am Laufen. |
| Testschritte | * Öffne Postman und erstelle einen PUT Request mit der URL [http://localhost:8800/api/users/{userID}](http://localhost:8800/api/users/%7buserID%7d) * Ersetze "{UserID}" in der URL durch die tatsächliche Benutzer-ID. * Füge im Request-Body die folgenden Daten hinzu: * Beschreibung: [Neue Benutzerbeschreibung] * Stadt: [Neue Stadt] * Ursprung: [Neuer Ursprung] * Beziehung: [Neue Beziehung] * Request absenden |
| Erwartetes Ergebnis | Der Server gibt den Statuscode 200 zurück und sendet die Meldung "Account updated" als Response. |
| Testergebnis | erfüllt |
| Testdatum | 27.06.2023 |
| Bemerkungen | - |

|  |  |
| --- | --- |
| **Testfall Nr.** | **4** |
| Beschreibung | Im Backend mit einem PUT Request demselben Benutzer folgen. |
| Voraussetzungen | Backend gestartet und am Laufen. |
| Testschritte | * Öffne Postman und erstelle einen PUT Request mit der URL http://localhost:8800/api/users/{UserID}/follow * Ersetze "{UserID}" in der URL durch die tatsächliche Benutzer-ID * Füge im Request-Body die folgenden Daten hinzu: * UserId: [UserId] * Request absenden |
| Erwartetes Ergebnis | Der Server gibt Statuscode 403 zurück und sendet die Meldung „you cant follow yourself“ als Response |
| Testergebnis | erfüllt |
| Testdatum | 27.06.2023 |
| Bemerkungen | - |

|  |  |
| --- | --- |
| **Testfall Nr.** | **5** |
| Beschreibung | Im Backend mit einem POST Request Post erstellen. |
| Voraussetzungen | Backend gestartet und am Laufen. |
| Testschritte | * Öffne Postman und erstelle einen POST Request mit der URL <http://localhost:8800/api/posts> * Füge im Request-Body die folgenden Daten hinzu: * userId: [Benutzer-ID] * desc: [Beitragstext] * Request absenden |
| Erwartetes Ergebnis | Der Server gibt den Statuscode 200 zurück und sendet den erstellten Beitrag als Response. |
| Testergebnis | erfüllt |
| Testdatum | 27.06.2023 |
| Bemerkungen | - |

#### 5.12 Frontend

|  |  |
| --- | --- |
| **Testfall Nr.** | **6** |
| Beschreibung | Einen neuen Benutzer über das Registrierungsformular im Frontend registrieren. |
| Voraussetzungen | Frontend und Backend gestartet und am Laufen. |
| Testschritte | * Öffne im Browser den URL <http://localhost:3000/register> * Auf der Registrierungsseite das Registrierungsformular ausfüllen. * Mit «SignUp» Formular abschicken. |
| Erwartetes Ergebnis | Registrierung erfolgreich abgeschlossen, und der Benutzer wird auf die Login-Seite weitergeleitet. |
| Testergebnis | erfüllt |
| Testdatum | 27.06.2023 |
| Bemerkungen | - |

|  |  |
| --- | --- |
| **Testfall Nr.** | **7** |
| Beschreibung | Einloggen mit einem registrierten Benutzer über das Login-Formular im Frontend. |
| Voraussetzungen | Frontend und Backend gestartet und am Laufen. |
| Testschritte | * Öffne im Browser den URL http://localhost:3000/login * Auf der Login Seite das Login Formular mit korrekten Daten ausfüllen. * Mit «Login» Formular abschicken. |
| Erwartetes Ergebnis | Registrierung erfolgreich abgeschlossen, und der Benutzer wird auf die Homeseite weitergeleitet. |
| Testergebnis | erfüllt |
| Testdatum | 27.06.2023 |
| Bemerkungen | - |

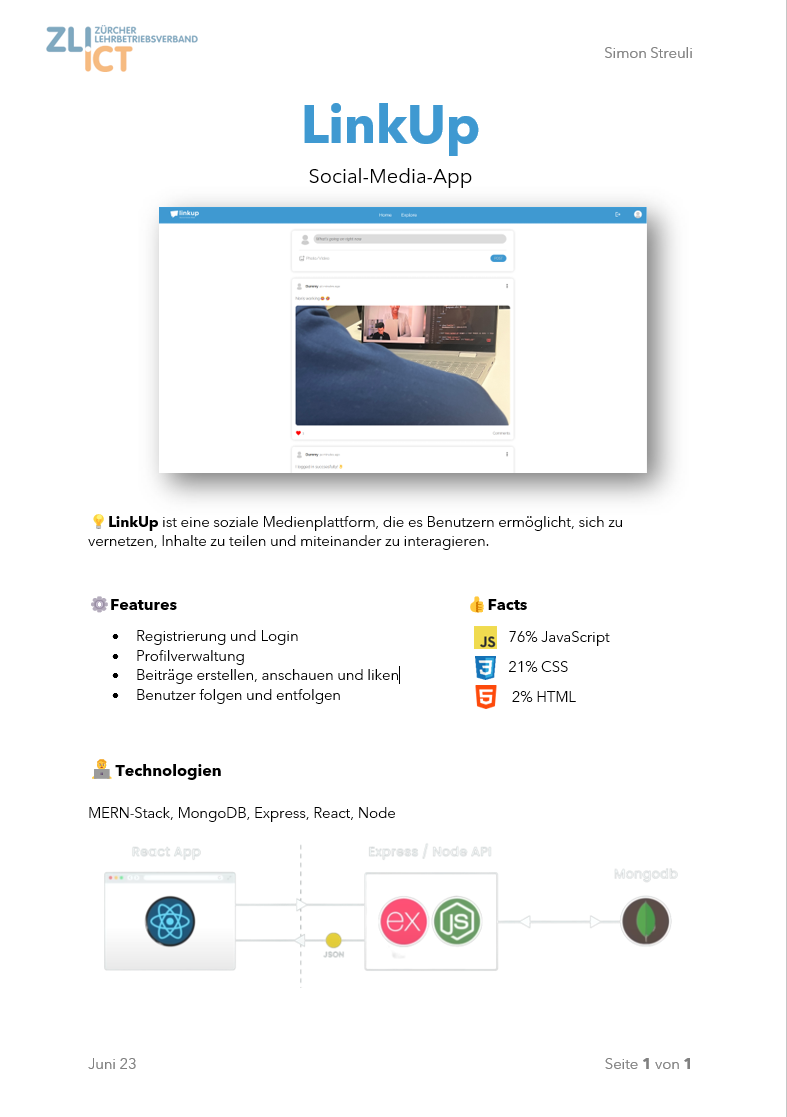
|  |  |
| --- | --- |
| **Testfall Nr.** | **8** |
| Beschreibung | Beschreibung Ändern der Benutzerinformationen im Profil im Frontend. |
| Voraussetzungen | Frontend und Backend gestartet und am Laufen. |
| Testschritte | * Öffne im Browser den URL http://localhost:3000/profile/{username} * Ersetzte username mit dem wirklichen Benutzernamen. * Klicke auf den «Follow» Button |
| Erwartetes Ergebnis | Änderungen werden erfolgreich gespeichert und im Profil angezeigt. |
| Testergebnis | erfüllt |
| Testdatum | 27.06.2023 |
| Bemerkungen | - |

|  |  |
| --- | --- |
| **Testfall Nr.** | **9** |
| Beschreibung | Einen anderen Benutzer im Frontend als Freund hinzufügen. |
| Voraussetzungen | Frontend und Backend gestartet, am Laufen und Benutzer eingeloggt. |
| Testschritte | * Öffne im Browser den URL http://localhost:3000/profile/{username} * Ersetzte username mit einem wirklichen Benutzernamen. * Füge im Request-Body die folgenden Daten hinzu: * userId: [Benutzer-ID] * desc: [Beitragstext] * Request absenden |
| Erwartetes Ergebnis | Benutzer wird erfolgreich gefolgt, Button ändert sich zu «unfollow» und die Followers Anzahl steigt beim gefolgten Benutzer. |
| Testergebnis | erfüllt |
| Testdatum | 27.06.2023 |
| Bemerkungen | - |

|  |  |
| --- | --- |
| **Testfall Nr.** | **10** |
| Beschreibung | Einen neuen Beitrag über das Eingabeformular im Frontend erstellen |
| Voraussetzungen | Frontend und Backend gestartet, am Laufen und eingeloggt. |
| Testschritte | * Öffne im Browser den URL http://localhost:3000/ * Schreibe im «What’s going on right now» Input die Post Description rein und füge ein Bild hinzu. * Schicke den Post mit dem «Post» Button ab. |
| Erwartetes Ergebnis | Post wird erfolgreich erstellt, der Post kann auf der Seite gesehen werden und hat ein Bild. Der Post hat keine Likes und ist von «a few seconds ago» |
| Testergebnis | erfüllt |
| Testdatum | 27.06.2023 |
| Bemerkungen | - |

# 6. Factsheet

### 6.1 Factsheet



Factsheet

# 7. Auswerten

### 7.1 Reflexion

# 8. Quellen

### 8.1 Quellen

|  |  |
| --- | --- |
| Quelle | Beschreibung |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

### 8.2 Bildquellen