



INDIVIDUELLE PRODUKTIVARBEIT

Internes soziales Netzwerk

Probelauf

1 Inhaltsverzeichnis

1	Inhaltsverzeichnis	1
2	Management Summary	2
3	Informieren	7
4	Planen	10
5	Entscheiden	21
6	Realisieren	23
7	Kontrollieren	32
8	Auswerten	38
9	Arbeitsprotokolle	41
10	Verzeichnisse	46
11	Anhang	47

2 Management Summary

2.1 Aufgabenstellung

2.1.1 Beschreibung der Applikation

Kleines soziales Netzwerk, bei dem ein Benutzer sich registrieren kann. Er muss bei der Registrierung einen Benutzernamen, eine E-Mail-Adresse, den Vornamen, Nachnamen und ein Passwort in Textform angeben. Freiwillig ist das Setzen eines Profilbildes. Einloggen kann sich der Benutzer ausschliesslich nur mit dem Benutzernamen oder E-Mail und mit dem Textpasswort über ein Login-Fenster mit zwei Eingabefeldern. Als eingeloggter Benutzer kann er einen Textbeitrag verfassen. Er kann beim Erstellen eines Beitrags einen Text mit begrenzter Zeichenzahl schreiben und den Beitrag veröffentlichen. Er kann den Beitrag auch wieder löschen. Dazu kann er seine und andere Beiträge mit «Gefällt mir» markieren, indem er beim Beitrag auf den entsprechenden Knopf drückt und nur mit Text unter dem Beitrag kommentieren. Die Kommentare können auch wieder vom Ersteller gelöscht werden. Die Benutzer sehen alle Beiträge von allen Benutzern. Andere Benutzer können andere Beiträge melden indem sie beim Beitrag auf die Melden-Option klicken. Die Benutzerrollen (Admin, Troubleshooter und Konsument) haben ihr eigenes Panel.

Der Zweck der Applikation ist es, in einem kleinen Unternehmen den globalen Informationsaustausch einfach und abgegrenzt zu ermöglichen.

2.1.2 Kriterien

2.1.2.1 Funktioneller Umfang

- Multiuser-Applikation
- Benutzerrollen
 - Konsument (Registrierter Benutzer)
 - Konsument kann sich registrieren
 - Pflicht:
 - Benutzername
 - Passwort und Passwort wiederholen
 - Wird verschlüsselt
 - Passwort wiederholen
 - E-Mail-Adresse
 - Vorname
 - Nachname
 - Kann:
 - Profilbild
 - Konsument kann sich einloggen
 - Textfeld für Benutzername oder E-Mail
 - Passworttextfeld für Passwort
 - Konsument kann sein Profil bearbeiten
 - Profilbild ändern
 - Vorname
 - Nachname
 - Benutzername
 - Passwort
 - E-Mail-Adresse
 - Konsument kann Beiträge in Form von Text verfassen, seine eigenen bearbeiten und löschen.
 - Nur Text im Beitrag
 - Bearbeiten mit Option direkt beim Beitrag

- Löschen mit Option direkt beim Beitrag
- «Gefällt-Mir-Funktion» und Kommentarfunktion für alle Konsumenten bei allen Beiträgen
 - Kommentar nur in Form von Text
 - Gefällt-Mir direkt beim Beitrag an- und abwählbar
- Beiträge können gemeldet werden
 - Werden von Troubleshooter verwaltet
- Admin
 - Admin hat Zugriff auf alle Hintergrundfunktionen (Eigenes Admin-Panel mit allen Funktionen auf einer Seite):
 - Beiträge ansehen
 - Beiträge löschen
 - Verwaltung der gemeldeten Beiträge
 - Benutzer hinzufügen/löschen/sperren
- Troubleshooter
 - Troubleshooter hat nur Zugriff auf die Verwaltung der gemeldeten Beiträge (Eigenes Panel)
 - Beiträge ansehen
 - Beiträge löschen
 - Benutzer sperren
- Passant (Nicht eingeloggter Benutzer)
 - Passant hat keinen Zugriff auf die Funktionen des Konsumenten und wird direkt zur Login-Seite weitergeleitet

2.1.2.2 Technischer Umfang

- Datenbankverbindung (MySQL/JDBC)
- Muss objektorientiert realisiert werden
- Produkt ist eine Java Servlet Page (JSP)

Die einzelnen Schritte müssen geplant, durchgeführt und dokumentiert werden.

2.1.3 Mittel und Methoden

- 1 Laptop
- Entwicklungsumgebung Eclipse
- Ausgabesprachen HTML und CSS für GUI
- Programmiersprachen Java und JavaScript für die Hintergrundfunktionen
- MySQL-Datenbank für die Datenablage
- SQL für die Datenabfrage und Datenmanipulation
- GitHub für das Sichern der Backups für das Projekt

2.1.4 Vorarbeiten

Im Eclipse muss die Erweiterung für Webapplikationen und der Tomcat-Server installiert, auf die eiwandfreie Funktionalität getestet und konfiguriert werden.

Dokumentation hat die Grundstruktur und der Zeitplan ist erstellt und die einzelnen Schritte geplant.

2.2 Vorkenntnisse

Während des 3. Lehrjahrs bekam ich einen vertieften Einblick in die Webentwicklung mit Java, HTML, CSS und JavaScript mit Verbindung zu MySQL.

Sprache	Anwendung	Kenntnis
HTML, CSS	In der gesamten Lehrzeit habe ich HTML, CSS immer wieder in Webapplikationen angewendet.	Sehr gut
SQL	Seit dem 2. Lehrjahr vertieft in Webapplikationen angewendet	Sehr gut
JavaScript	Seit dem 2. Lehrjahr in fast jeder Webapplikation vertieft angewendet	Gut
Java	Seit dem 3. Lehrjahr vertieft in Webapplikationen angewendet	Gut

Tabelle 1 - Auflistung Vorkenntnisse

2.3 Vorarbeiten

Die folgenden Arbeitsschritte wurden als Vorbereitung durchgeführt:

- Eclipse JSP-Erweiterung installiert
- Tomcat-Server für Eclipse installiert
- Einrichten von GitHub mit Eclipse
- Test-Verbindung zu Datenbank
- Test-Verbindung zu Tomcat-Server

2.4 Projektmanagement

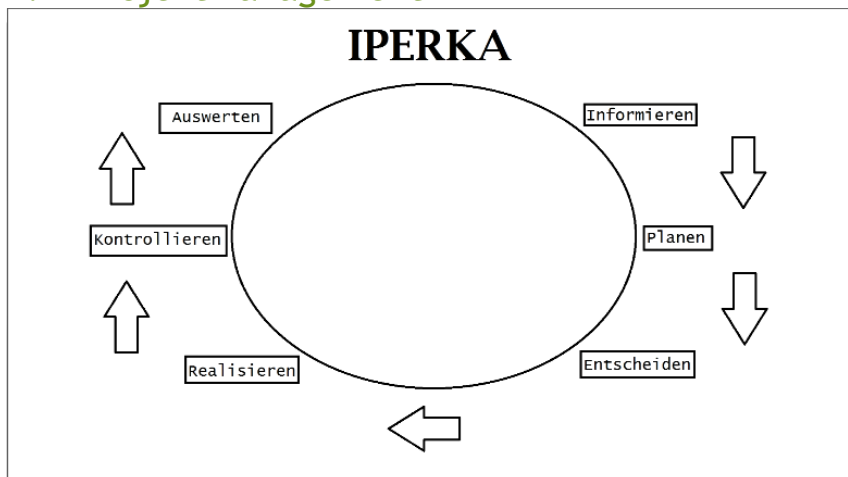


Abbildung 1 - IPERKA-Modell¹

In der oberen Grafik (Abbildung 1) sieht man den Strukturierten Ablauf von IPERKA. Der Ablauf ist klar strukturiert, dh. dass zuerst alle Schritte genau geplant werden, bevor sie dann durchgeführt werden. Somit kann man im Verlaufe des Projekts keine neuen Arbeitsschritte hinzufügen und neu planen.

¹ Quelle: <https://projekte.bbbaden.ch/2010/12/23/projekttag-der-film/iperka/>
Dennis Schächli
Siemens Schweiz AG

2.5 Zeitplan

Nr.	Tätigkeit	31.10.2018			01.11.2018			02.11.2018			07.11.2018			09.11.2018		
		2	4	6	2	4	6	2	4	6	2	4	6	2	4	6
I	Informieren															
1	Aufgabenstellung															
2	Management Summary															
P	Planen															
1	Zeitplan															
2	GUI Design															
3	Planung Programmablauf und -umfang															
4	Datenbankdesign															
E	Entscheiden															
1	Wahl der Datenbankumgebung															
2	Wahl der Serverseitigen Programmiersprache															
3	Wahl der Entwicklungsumgebung															
R	Realisieren															
1	Implementierung Login-Seite															
2	Implementierung Registrierungsseite															
3	Implementierung Beitragsseite															
4	Implementierung "Beitrag erstellen und bearbeiten"															
5	Implementierung "Mein Profil"															
6	Implementierung Control Panels															
K	Kontrollieren															
1	Testing Normalfall															
2	Testing Extremfall															
A	Auswerten															
1	Ergebnis															
2	Gelungen															
3	Nicht gelungen															
4	Reflexion															
D	Dokumentation															
W	Wiederkehrende Aufgaben															
D	Arbeitsjournal															

Abbildung 2 - Zeitplan

2.5.1 Meilensteine

Nr.	Meilenstein	Aufgaben	Datum
1	Informieren	Die Dokumentation beinhaltet den kompletten Teil, des Informierens.	31.10.2018
2	Planen & Entscheiden	Die gesamte Planung ist in der Dokumentation vorhanden. Dazu gehören das GUI-Mockup, das Klassendiagramm und das Datenbankdesign.	02.11.2018
3	Realisieren	Die Aufgaben orientieren sich an den Teil des Realisierens im IPERKA-Modell	07.11.2018
4	Kontrollieren	Die Aufgaben orientieren sich an den Teil des Kontrollierens im IPERKA-Modell	09.11.2018
5	Auswerten und Abgabe	Die Aufgaben orientieren sich an den Teil des Auswertens im IPERKA-Modell Abgabe der Probe-IPA	09.11.2018

Tabelle 2 - Auflistung Meilensteine

2.6 Organisation

Die verantwortliche Fachkraft ist für die Zulassung der Aufgabenstellung zuständig. Die Experten sind für die Bewertung dieser Dokumentation, der Präsentation und für das Fachgespräch zuständig.

2.7 Projektübersicht

Diese Dokumentation orientiert sich an den einzelnen Phasen meiner gewählten Projektmanagementmethode:

- Informieren
- Planen
- Entscheiden
- Realisieren
- Kontrollieren
- Auswerten

Diese zusätzlichen Informationen sind in dieser Dokumentation auch vorhanden:

- Arbeitsprotokoll
- Anhang

2.8 Lehrbetrieb

Siemens Schweiz AG
Freilagerstrasse 40
8047 Zürich

2.9 Involvierte Personen

Lernender	Dennis Schächli
Verantwortliche Fachkraft / Hauptexperte	Remo Steinmann
Berufsbildner	Jonas Knoll

Tabelle 3 - Involvierte Personen

3 Informieren

3.1 Ausgangslage

Die Applikation wird von Grund auf neu programmiert. Die Entwicklungsumgebung wurde so eingerichtet, dass man direkt mit der Realisierung anfangen könnte.

3.2 Umgebung

- Lokal auf privatem Gerät
- Datenbank läuft lokal unter XAMPP
- Git-Repository auf GitHub

3.3 Rechtschreibung und Grammatik

Für die Rechtschreibung wird das Korrekturprogramm von MS Word verwendet. Zusätzlich lese ich diese Dokumentation auch mehrmals durch und korrigiere alle Fehler manuell.

3.4 Technologien

- Entwickelt mit Java, HTML, CSS und JavaScript
- MySQL

3.5 UseCase

Dieses Use-Case-Diagramm wurde mit MS Visio erstellt und als PNG-Bild exportiert. Visio konnte ich von Microsoft kostenlos herunterladen, da ich mich als TBZ-Schüler dort anmelden konnte und wir bekamen sämtliche Office-Programme kostenlos. Visio habe ich schon während meiner ganzen Lehrzeit angewendet und bin mit diesem Tool am meisten vertraut.

Für die Erklärung dieses UseCase-Diagramms nehme ich den Bezug zur Beschreibung in der Applikation in der Aufgabenstellung:

«Er muss bei der Registrierung einen Benutzernamen, eine E-Mail-Adresse, den Vornamen, Nachnamen und ein Passwort in Textform angeben. Freiwillig ist das Setzen eines Profilbildes. Einloggen kann sich der Benutzer ausschliesslich nur mit dem Benutzernamen oder E-Mail und mit dem Textpasswort über ein Login-Fenster mit zwei Eingabefeldern. Als eingeloggter Benutzer kann er einen Textbeitrag verfassen. Er kann beim Erstellen eines Beitrags einen Text mit begrenzter Zeichenzahl schreiben und den Beitrag veröffentlichen. Er kann den Beitrag auch wieder löschen. Dazu kann er seine und andere Beiträge mit «Gefällt mir» markieren, indem er beim Beitrag auf den entsprechenden Knopf drückt und nur mit Text unter dem Beitrag kommentieren. Die Kommentare können auch wieder vom Ersteller gelöscht werden. Die Benutzer sehen alle Beiträge von allen Benutzern. Andere Benutzer können andere Beiträge melden indem sie beim Beitrag auf die Melden-Option klicken. Die Benutzerrollen (Admin, Troubleshooter und Konsument) haben ihr eigenes Panel.»

Das UseCase-Diagramm befindet sich auf der nächsten Seite.

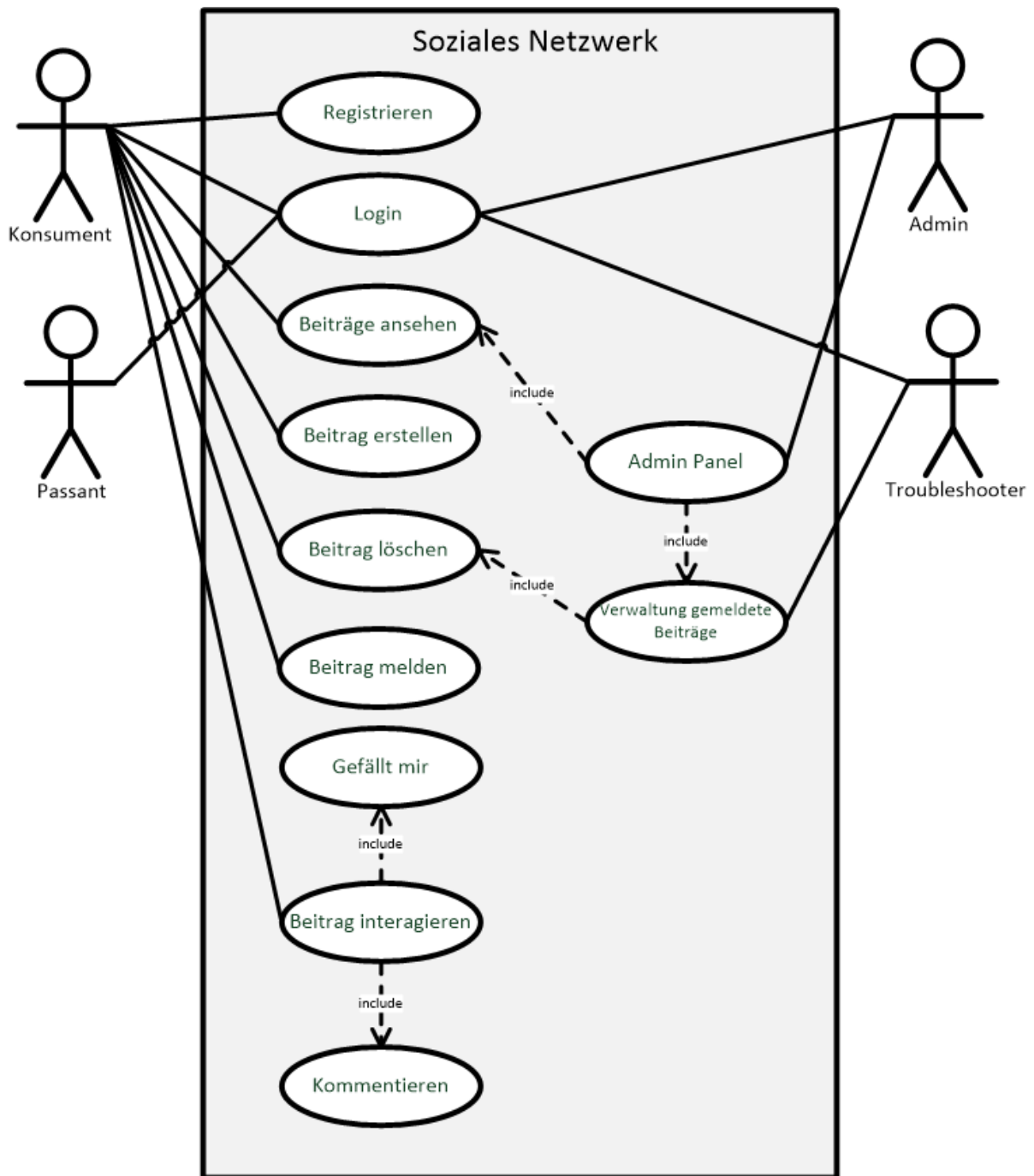


Abbildung 3 - UseCase-Diagramm

3.6 Datenbank

Damit ich das Datenbankmodell erstellen kann, muss ich zuerst analysieren, was alles in die Datenbank kommt und wie es aufgebaut ist. Folgende Tabellen konnte anhand des UseCase-Diagramms (Abbildung 3) definieren:

Tabelle	Spalte	Datentyp	Einschränkung
Benutzerrolle	ID Benutzerrolle	Ganzzahl	
	Bezeichnung	Text	Max. 20 Zeichen
Berechtigungen	ID Berechtigung	Ganzzahl	
	Bezeichnung	Text	Max. 20 Zeichen
Zuteilung Berechtigung	ID Zuteilung	Ganzzahl	
	FS Benutzerrolle	Ganzzahl	
	FS Berechtigung	Ganzzahl	
Benutzer	ID Benutzer	Ganzzahl	
	Benutzername	Text	Max. 20 Zeichen
	Vorname	Text	Max. 30 Zeichen
	Nachname	Text	Max. 30 Zeichen
	Email	Text	Max. 50 Zeichen
	Passwort (Verschlüsselt)	Text	Max. 255 Zeichen
	FS Benutzerrolle	Ganzzahl	
	Gesperrt	Ganzzahl	Nur 0 oder 1
Beitrag	ID Beitrag	Ganzzahl	
	Titel	Text	Max. 30 Zeichen
	Beschreibung	Text	Max. 300 Zeichen
	Datum des Posts	Datum/Uhrzeit	
	FS Benutzer	Ganzzahl	
	Gemeldet	Ganzzahl	Nur 0 oder 1
Kommentare	ID Kommentar	Ganzzahl	
	FS Beitrag	Ganzzahl	
	FS Benutzer	Ganzzahl	
	Text	Text	Max. 100 Zeichen
	Datum des Kommentars	Datum/Uhrzeit	
	Gemeldet	Ganzzahl	Nur 0 oder 1
Gefällt Mir	ID Gefällt Mir	Ganzzahl	
	FS Benutzer	Ganzzahl	
	FS Beitrag	Ganzzahl	

Tabelle 4 - Detailplanung Tabelle für Datenbank

Alle Primärschlüssel sind **rot** markiert und alle Fremdschlüssel sind **blau** markiert.

3.7 Berechtigungen

	Passant	Konsument	Admin	Troubleshooter
Login-Seite	✓	✓	✓	✓
Beitragsseite	x	✓	✓	✓
Mein Profil	x	✓	✓	✓
Registrieren-Seite	✓	✓	✓	✓
Neuer Beitrag erstellen	x	✓	✓	✓
Admin Control Panel	x	x	✓	x
Troubleshooter Panel	x	x	x	✓

Tabelle 5 - Berechtigungsmatrix Benutzerrollen

4 Planen

4.1 Datenbank ERM

Nach dem ich mich über die nötigen Tabellen und Felder informieren konnte, konnte ich nun das Datenbankmodell erstellen, damit ich beim Realisieren nur noch nach diesem Modell die Datenbank aufsetzen kann. Um das Datenbankmodell besser verstehen zu können, habe ich folgende Legende:

PK	Primärschlüssel
FK	Fremdschlüssel
←	Die Beziehungsverknüpfung endet mit einem Pfeil, wenn sie auf den Primärschlüssel verweist. (1)
—	Die Beziehungsverknüpfung endet mit einer Linie, wenn sie auf die Fremdschlüssel verweist. (n)
Normal	Die normal gedruckten Wörter zeigen, dass die Werte in der Spalte NULL sein dürfen.
Fett	Die fett gedruckten Wörter zeigen, dass die Werte in der Spalte nicht NULL sein dürfen.

Tabelle 6 - Legende für Abbildung 4

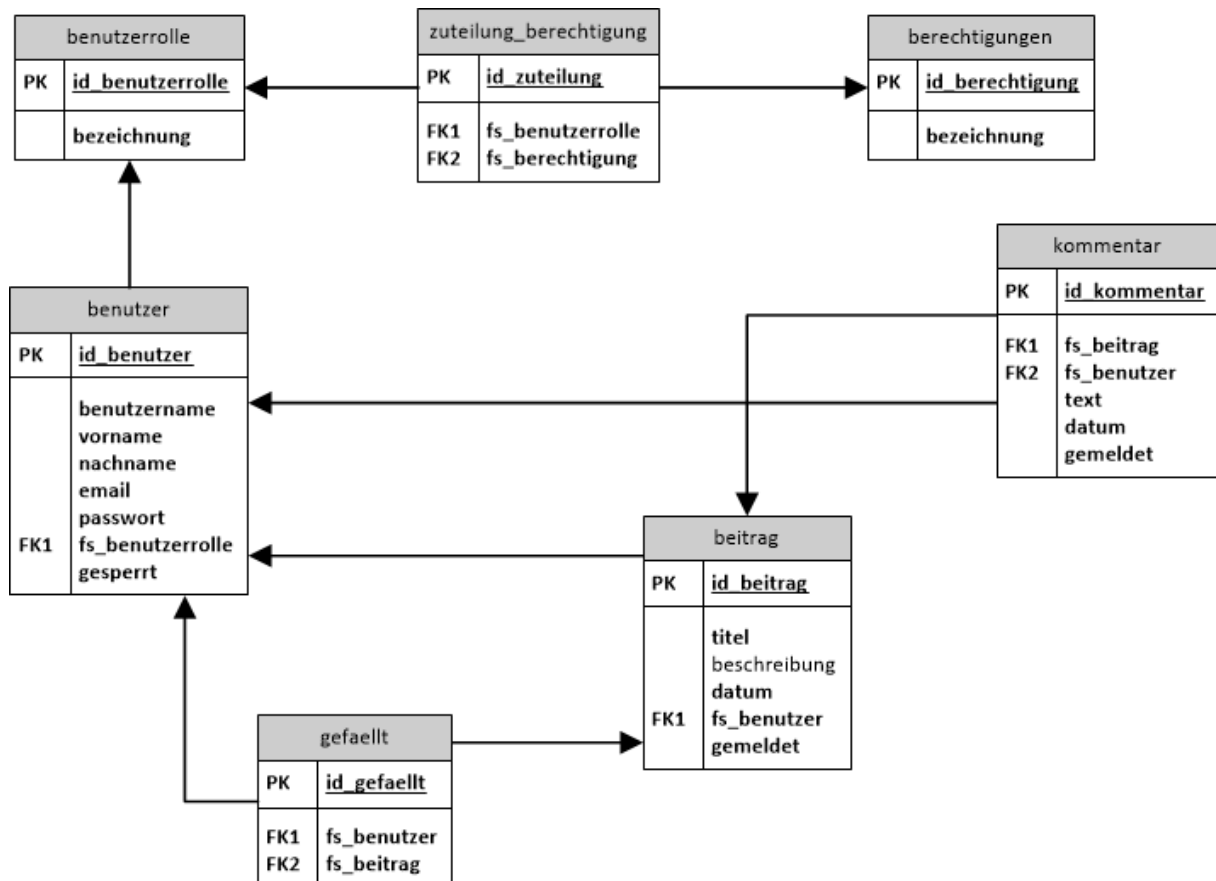


Abbildung 4 - Datenmodell Planung

4.2 Stored Procedures

Damit SQL-Funktionen und Java-Code sauber getrennt sind, verwende ich für das Aufrufen der Funktionen Stored Procedures.

Name	Beschreibung	Seite
loadBenutzer	Lädt alle Benutzerdaten	Login/Mein Profil
register	Schreibt den neuen Benutzer in die Datenbank	Registrieren
loadBeitraege	Lädt alle nicht gemeldeten Beiträge	Beiträge
loadAlleBenutzer		Admin Control Panel
loadGemeldeteBeitraege	Lädt alle gemeldeten Beiträge	Troubleshooter Panel
setGefaeilt	Erfasst ein «Gefällt Mir» beim Beitrag	Beiträge
entferneGefaeilt	Entfernt ein «Gefällt Mir» beim Beitrag	Beiträge
meldeBeitrag	Setzt den Beitrag auf gemeldet	Beiträge
sperreBenutzer	Setzt den Benutzer auf gesperrt	Admin Control Panel
entsperreBenutzer	Setzt den Benutzer auf nichtgesperrt	Admin Control Panel
loescheBeitrag	Löscht den Beitrag aus der Datenbank	Beiträge
erstelleBeitrag	Erfasst einen neuen Beitrag	Neuer Beitrag
loadKommentare	Lädt alle Kommentare eines Beitrags	Beiträge
schreibeKommentar	Schreibt einen Kommentar unter den entsprechenden Beitrag	Beiträge

Tabelle 7 - Planung Stored Procedures

4.3 Erlaubte Zeichen in den Feldern

Bei der Registrierung hat der Benutzer bestimmte Einschränkungen bei der Zeicheneingabe:

Feld	Erlaubte Zeichen
Benutzername	Nur Buchstaben (ohne äöü) und Zahlen
Vorname	Nur Buchstaben
Nachname	Nur Buchstaben
Email	Nur Buchstaben (ohne äöü), Zahlen, Punkt, Binde- und Unterstrich und @
Passwort	Nur Buchstaben, Zahlen und folgende Sonderzeichen: ?!\$&%-_.:
Passwort wiederholen	Nur Buchstaben, Zahlen und folgende Sonderzeichen: ?!\$&%-_.:

Tabelle 8 - Auflistung erlaubter Zeichen beim Registrieren

4.4 Back-End Klassendiagramm

Ich habe mich am UseCase-Diagramm orientiert und so das Klassendiagramm erstellt. Zuerst habe ich die Klassen grob definiert und dann für jede Klassen die relevanten Funktionen aufgelistet:

- Benutzer
 - Die Klasse Benutzer speichert alle Benutzerdaten und beinhaltet die Optionen, welche mit dem Benutzer angestellt werden können.
 - Registrieren
 - Sperren
 - Identifizieren
- Beitrag
 - Die Klasse Beitrag speichert alle Informationen eines Beitrags und beinhaltet die Optionen, welche mit dem Beitrag angestellt werden können:
 - Melden
 - Löschen
 - Bearbeiten
 - Gefällt Mir
- Login
 - Login
 - Logout
 - Benutzerdaten überprüfen
 - Passwort verschlüsseln
- Controller
 - Beiträge aus Datenbank laden
 - Benutzer der Beiträge und angemeldeter Benutzer identifizieren
- Datenbank
 - Verbindung zur Datenbank herstellen

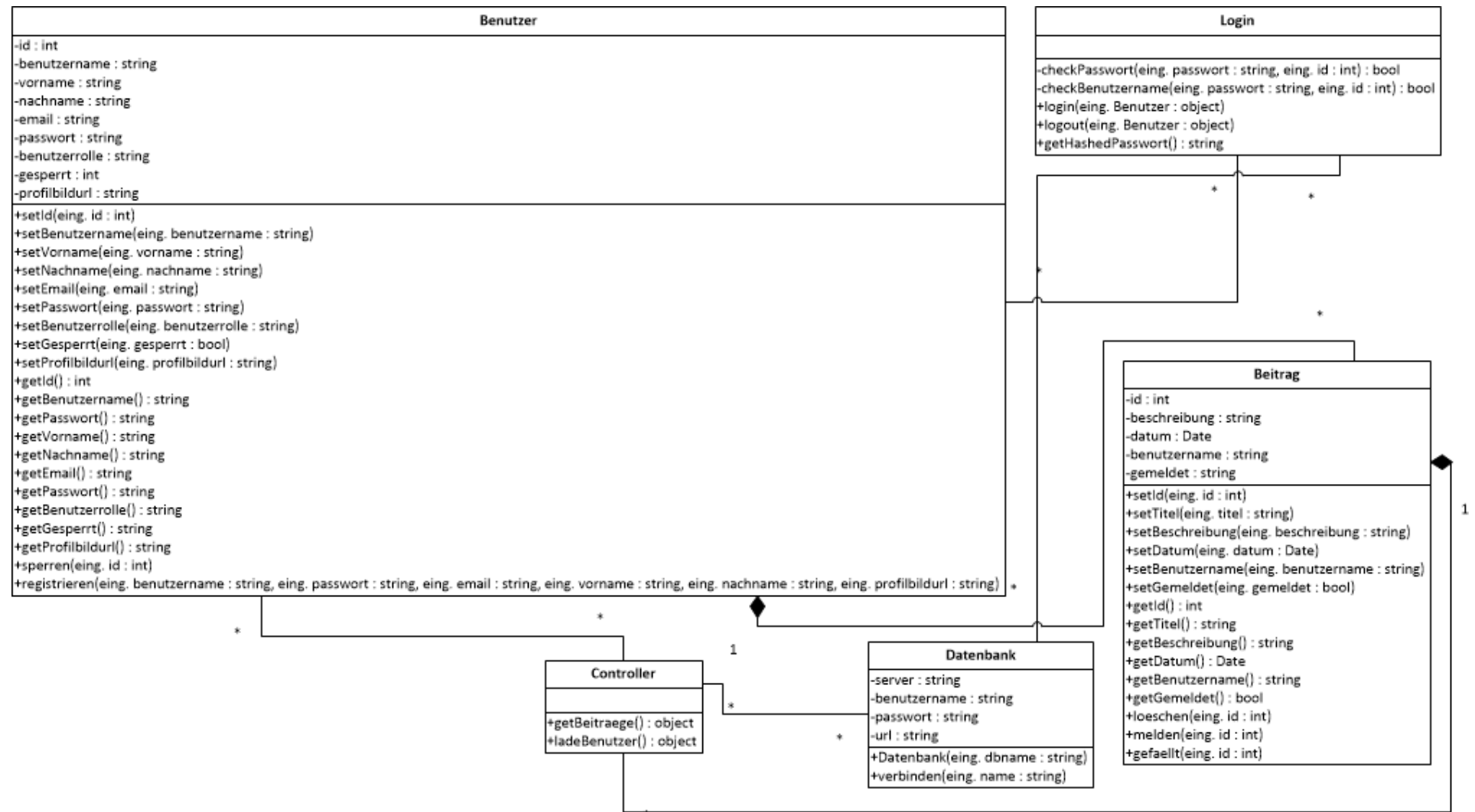


Abbildung 5 - Klassendiagramm

4.5 GUI Mockup

4.5.1 Registrieren

Auf dieser Seite (Abbildung 6) kann der Benutzer folgende Daten eingeben:

- Benutzername (Textfeld)
- Vorname (Textfeld)
- Nachname (Textfeld)
- Email (Textfeld)
- Passwort (Passworttextfeld)
- Passwort wiederholen (Passworttextfeld)

Der Benutzer ist auch bei der Registration dazu in der Lage, ein Profilbild für sein Profil auszuwählen.

Mit einem Klick auf «Senden kann er das Formular abschicken und der Benutzer wird registriert.

The mockup shows a registration form titled "Registrieren". It contains six text input fields stacked vertically: "Benutzername", "Vorname", "Nachname", "Email", "Passwort", and "Passwort wiederholen". Below these fields are two buttons: "Profilbild auswählen" and "Senden".

Registrieren	
Benutzername	
Vorname	
Nachname	
Email	
Passwort	
Passwort wiederholen	
Profilbild auswählen	
Senden	

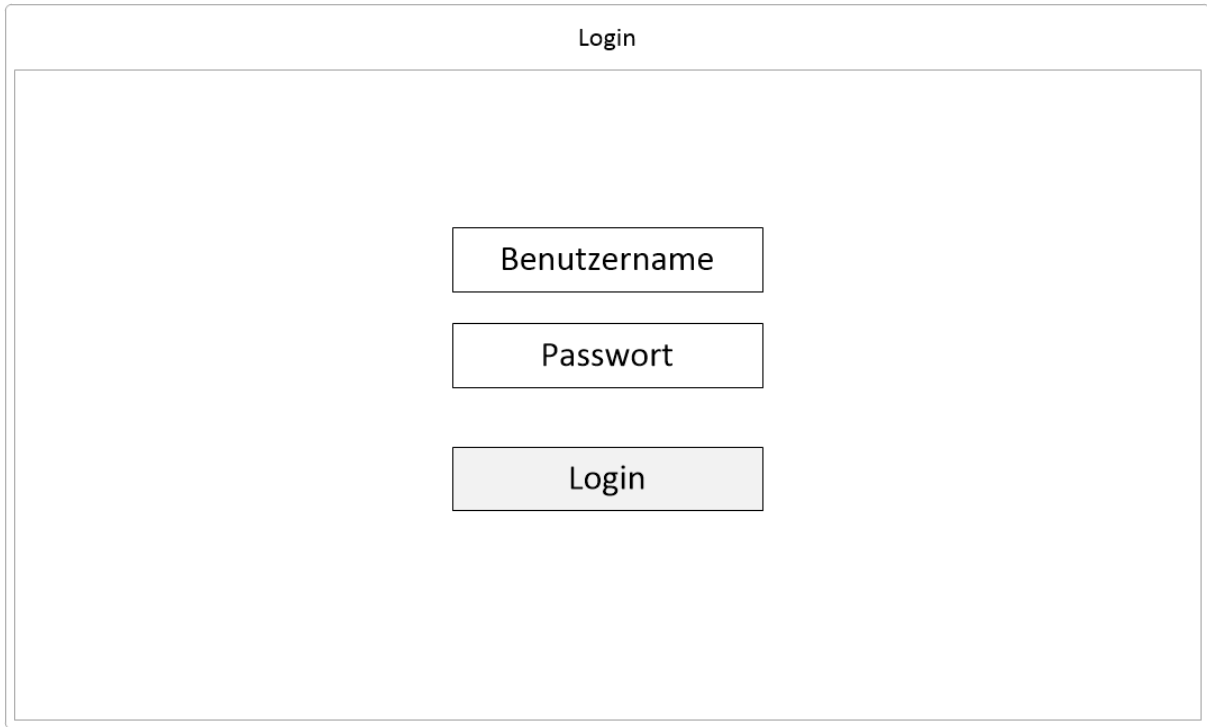
Abbildung 6 - GUI Registrieren

4.5.2 Login

Auf dieser Seite (Abbildung 7) kann der registrierte Benutzer folgende Daten eingeben:.

- Benutzername oder E-Mail-Adresse (Textfeld)
- Passwort (Passworttextfeld)

Mit einem Klick auf «Login» werden zuerst die angegebenen Daten überprüft und dann bei korrekter Prüfung angemeldet.



The diagram shows a login interface within a window titled "Login". Inside the window, there is a large rectangular area containing three vertically stacked input fields. The top field is labeled "Benutzername", the middle field is labeled "Passwort", and the bottom field is a button labeled "Login".

Abbildung 7 - GUI Login

4.5.3 Beiträge

Auf dieser Seite (Abbildung 8) werden alle Beiträge aufgelistet, die Konsumenten erstellt haben. Ganz oben links beim Beitrag wird angezeigt, wer ihn gepostet hat und wann er gepostet wurde. Oben rechts werden die Optionen angezeigt, jedem Benutzer das Melden und dem Ersteller das Löschen. Unten kann der Benutzer den Beitrag mit «Gefällt Mir» markieren. Ganz unten kann der Benutzer in ein Kommentarfeld (Textfeld) einen Kommentar verfassen.

Beiträge	
Von: [Benutzername]	Gepostet am: [Datum/Zeit]
Melden Löschen	
Titel	
Text...	
Gefällt Mir	
Kommentare	

Abbildung 8 - GUI Beiträge

4.5.4 Neuer Beitrag

Auf dieser Seite (Abbildung 9) kann der Konsument einen neuen Beitrag erstellen. Er kann folgendes definieren:

- Titel (Textfeld)
- Text (Textfeld)

Nachdem der Benutzer den Beitrag fertig geschrieben hat, kann er auf «Erstellen» klicken und der Beitrag wird gepostet.



The image shows a user interface for creating a new post. It is titled "Neuer Beitrag" at the top. Below the title is a large rectangular area containing two text input fields. The first field is labeled "Titel" and the second field is labeled "Text...". Below these fields is a button labeled "Erstellen".

Abbildung 9 - GUI Neuer Beitrag

4.5.5 Mein Profil

Auf dieser Seite (Abbildung 10) kann der Benutzer sein Profil bearbeiten, dabei kann er folgendes ändern:

- Benutzername (Textfeld)
- Vorname (Textfeld)
- Nachname (Textfeld)
- Profilbild (Image)

Mit einem Klick auf «speichern» kann der Benutzer seine Änderungen speichern.

Mein Profil	
Benutzername:	
Vorname:	
Nachname:	
Email:	
Profilbild:	
<div>Speichern</div>	

Abbildung 10 - GUI Mein Profil

4.5.6 Admin Control Panel

Auf dieser Seite (Abbildung 11) kann nur der Admin zugreifen und er sieht eine Auflistung aller Benutzer und eine Liste aller Beiträge. Bei den Beiträgen sieht er auch, ob dieser gemeldet wurde. Der Admin kann Benutzer sperren und Beiträge löschen.

Admin Control Panel

Liste Benutzer

Benutzername	sperren

Liste Beiträge

Titel	Beschreibung	gemeldet	löschen

Abbildung 11 - GUI Admin Control Panel

4.5.7 Troubleshooter Panel

Auf dieser Seite (Abbildung 12) kann nur der Troubleshooter zugreifen und er sieht eine Liste mit allen gemeldeten Beiträgen. Er kann diese löschen.

Troubleshooter Panel

Liste gemeldete Beiträge		
Titel	Beschreibung	löschen

Abbildung 12 - GUI Troubleshooter Panel

5 Entscheiden

5.1 Datenbankumgebung

Ich habe die folgenden 2 Möglichkeiten:

- MySQL
- MSSQL

Kriterien:

Kriterium	Gewichtung
Bedienbarkeit	1x
Kosten	2x
Unterstützung Java	3x

Tabelle 9 - Auflistung Kriterien Datenbankumgebung

Meine Entscheidung:

	MySQL	MSSQL
Bedienbarkeit	1	1
Kosten	2	0
Unterstützung Java	3	3
Punkte	6	4

Tabelle 10 - Entscheidungsmatrix Datenbankumgebung

Ich habe mich für MySQL entschieden, weil ich die Probe-IPA auf einem privaten Gerät mache und deshalb eine kostenlose Datenbankumgebung haben will.

5.2 Entwicklungsumgebung

Ich habe bis jetzt mit den folgenden 2 Entwicklungsumgebungen gearbeitet:

- Eclipse
- Visual Studio

Kriterien:

Kriterium	Gewichtung
Bedienbarkeit	1x
Kosten	2x
Unterstützung Java	3x

Tabelle 11 - Auflistung Kriterien Entwicklungsumgebung

Meine Entscheidung:

	Eclipse	Visual Studio
Bedienbarkeit	1	1
Kosten	2	0
Unterstützung Java	3	3
Punkte	6	4

Tabelle 12 - Entscheidungsmatrix Entwicklungsumgebung

Ich habe mich für Eclipse entschieden, weil ich die Probe-IPA auf dem privaten Gerät mache und deshalb eine kostenlose Entwicklungsumgebung will.

5.3 Dynamisch und statisch

Ich musste mich für die Art entscheiden, wie ich diese Applikation realisieren werde. Ich hatte die Wahl zwischen einer statischen oder dynamischen Webapplikation.

Kriterien:

Kriterium	Gewichtung
Datenbankverbindung	2x
Unterstützung Java	3x

Tabelle 13 - Auflistung Kriterien Statisch/Dynamisch

Meine Entscheidung:

	Statisch	Dynamisch
Datenbankverbindung	0	2
Unterstützung Java	0	3
Punkte	0	5

Tabelle 14 - Entscheidungsmatrix Statisch/Dynamisch

Ich habe mich für eine dynamische Applikation entschieden, weil nur hier das Verbinden zu einer Datenbank möglich ist und bei Dynamisch auch eine serverseitige Programmiersprache eingebunden werden kann.

6 Realisieren

6.1 Datenbank aufsetzen

Als erstes fing ich damit an, die Datenbank aufzusetzen. Die Angaben zur Datenbank:

Name der Datenbank:	probeipa_sozialesnetzwerk
Server:	localhost
Benutzer:	root
Passwort:	

Tabelle 15 - Zugriffsdaten Datenbank

Damit es keine Komplikationen mit dem Fremdschlüsseln gibt, habe ich die Tabellen in dieser Reihenfolge erzeugt:

1. Benutzerrolle
2. Berechtigungen
3. Benutzer
4. Beitrag
5. Kommentar
6. Gefaellt
7. Zuteilung_Berechtigung

Danach musste ich noch die nötigen Beziehungen von Fremdschlüssel zu Primärschlüssel setzen und das ERM sah dann so aus:

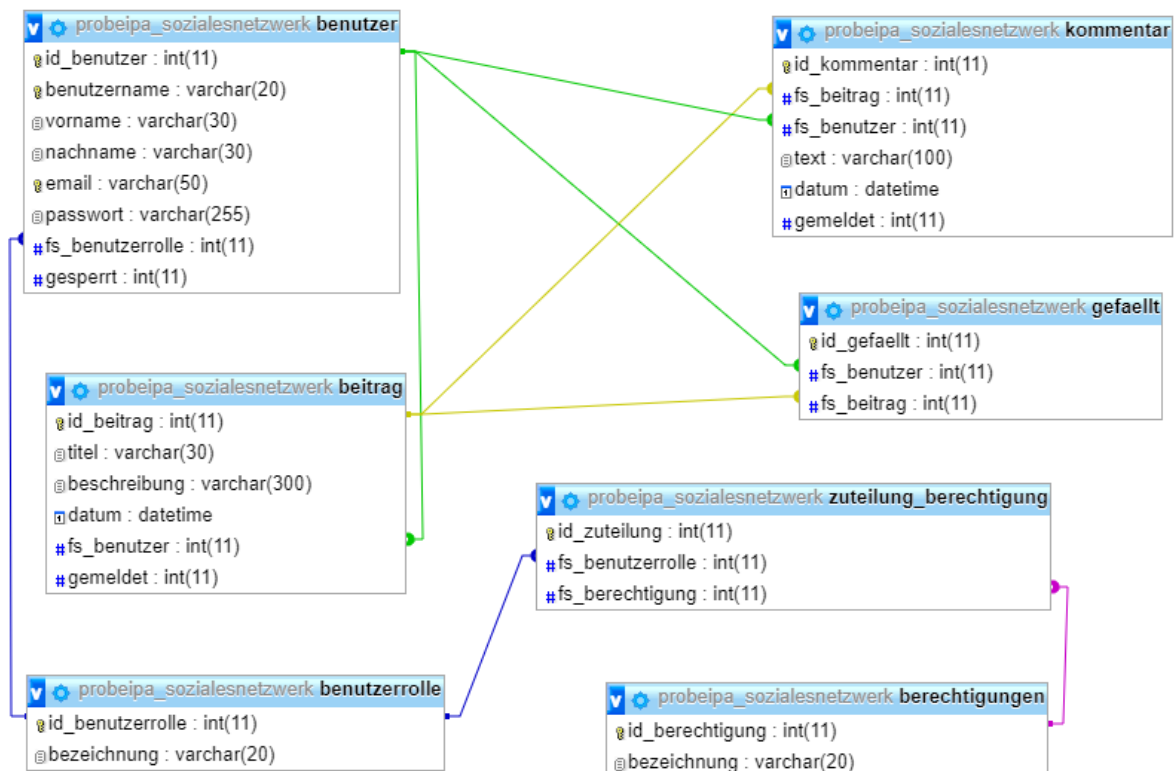


Abbildung 13 - Datenmodell nach Realisierung

Nachdem ich alle diese Tabellen erzeugt habe, musste ich die Grunddaten bei der Tabelle «Berechtigungen» einschreiben, denn das kann später nicht über das GUI gemacht werden.

Folgende Daten waren notwendig für die Tabelle «Berechtigungen», dabei orientiere ich mich aus der Berechtigungsmatrix aus dem Kapitel 3.7:

- Beitrag entfernen
- Beitrag melden
- Beitrag erstellen
- Beitrag bearbeiten
- Benutzer sperren

Auch die Grunddaten für die Tabelle «Benutzerrolle» war notwendig:

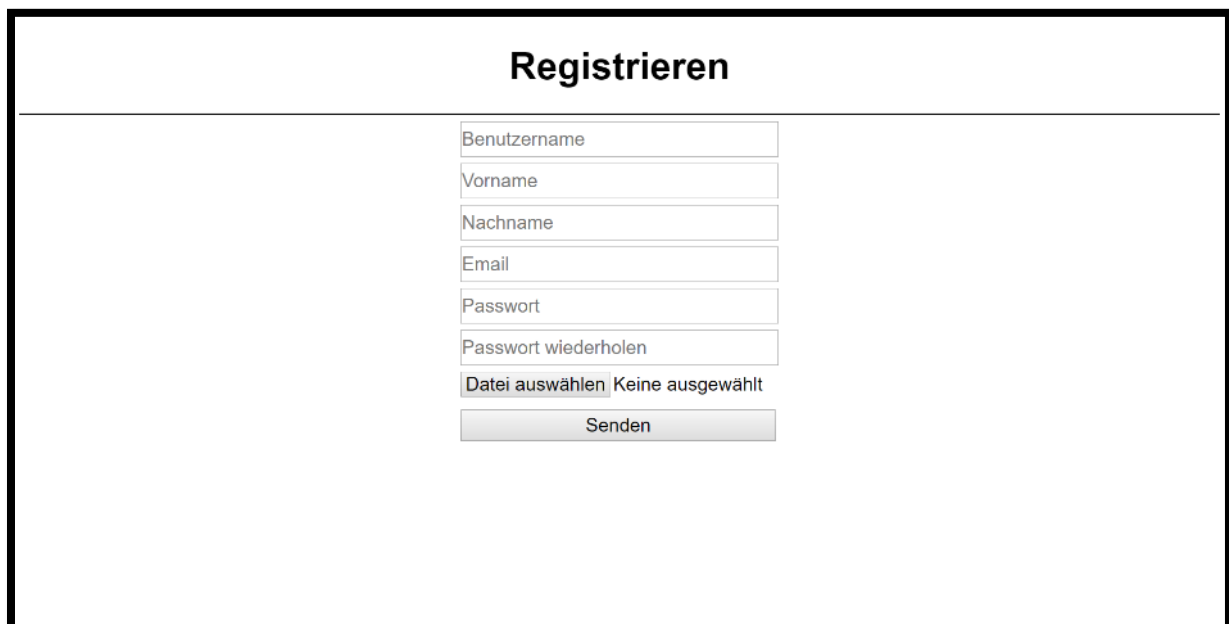
- Konsument
- Troubleshooter
- Admin

6.2 Implementierung Frontend

Als erstes habe ich alle HTML-Elemente bei den Seiten hinzugefügt, ohne das CSS zu berücksichtigen. Im zweiten Schritt habe ich das CSS dem Design der Mockups angepasst.

6.2.1 Registrieren

Die Seite Registrieren wurde im Frontend nach Plan umgesetzt und gemäss Mockup auch so angezeigt. Dazu gehören die Eingabe Felder Benutzername, Vorname, Nachname, E-Mail, Passwort und Passwort wiederholen, das Dialogfeld Profilbild auswählen und der Absende-Knopf.



Registrieren

Benutzername

Vorname

Nachname

Email

Passwort

Passwort wiederholen

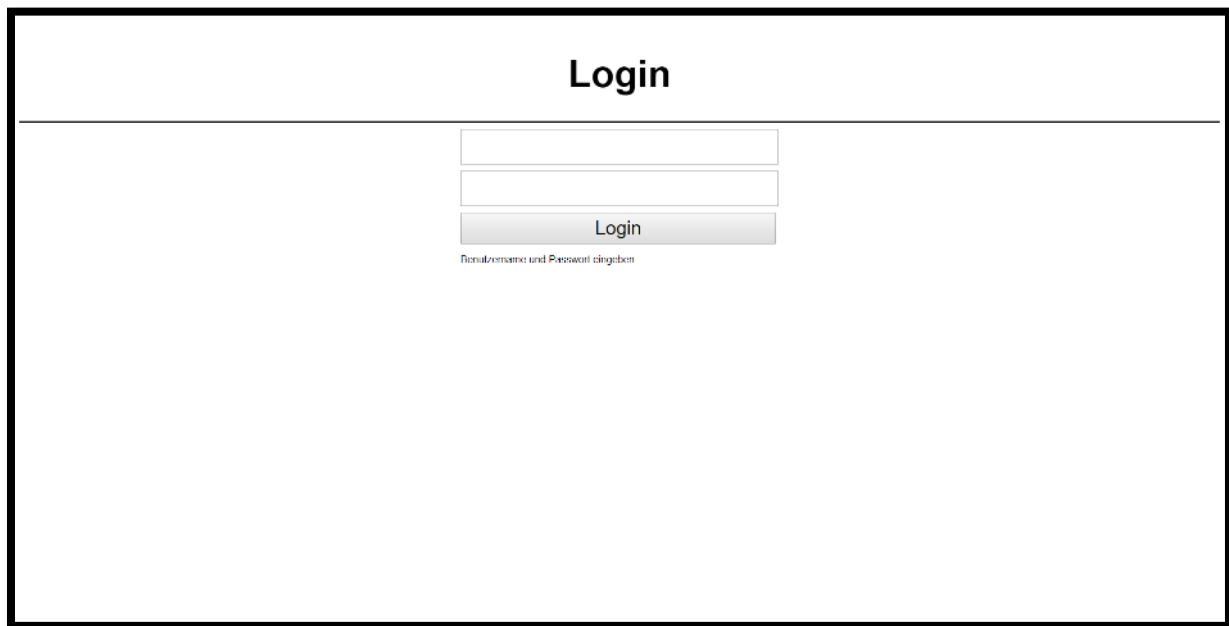
Datei auswählen Keine ausgewählt

Senden

Abbildung 14 - Ergebnis Seite Registrieren

6.2.2 Login

Die Seite Login wurde im Frontend nach Plan ungesetzt und gemäss Mockup auch so dargestellt. Dazu gehören die Eingabefelder Benutzername und Passwort, und der Absende-Knopf für das Login.



The mockup shows a login page with a white background and a black border. At the top center, the word "Login" is displayed in a bold, black font. Below this, there is a horizontal line. Under the line, there are two input fields: the first is for the username and the second is for the password. Below these fields is a button labeled "Login". At the bottom of the input fields, there is a small, faint text label that reads "Benutzername und Passwort eingeben".

Abbildung 15 - Ergebnis Seite Login

6.2.3 Beitrag

Die Seite Beitrag wurde um Frontend nur teilweise nach Plan umgesetzt und gemäss Mockup nicht ganz so dargestellt wie geplant. Umgesetzt wurde:

- Info wer den Beitrag gepostet hat
- Info wann der Beitrag gepostet wurde (nur Datumsgenau)
- Titel
- Beschreibung
- Kommentarfeld und der dazugehörige Knopf zum absenden
- Anzeige der Kommentare

Nicht umgesetzt wurde:

- Option Beitrag melden
- Option Beitrag löschen
- Gefällt Mir

Design Abweichungen:

- Beiträge nicht zentriert sondern linksbündig

Beiträge

Benutzer-ID: 11 [Neuer Beitrag](#) [Mein Profil](#) [Beiträge](#) [Logout](#)

Von: testor Gepostet am 2016-11-09

test
test

Kommentar schreiben
Senden

Von: testor Gepostet am 2016-11-09

Test
Test

Kommentar schreiben
Senden

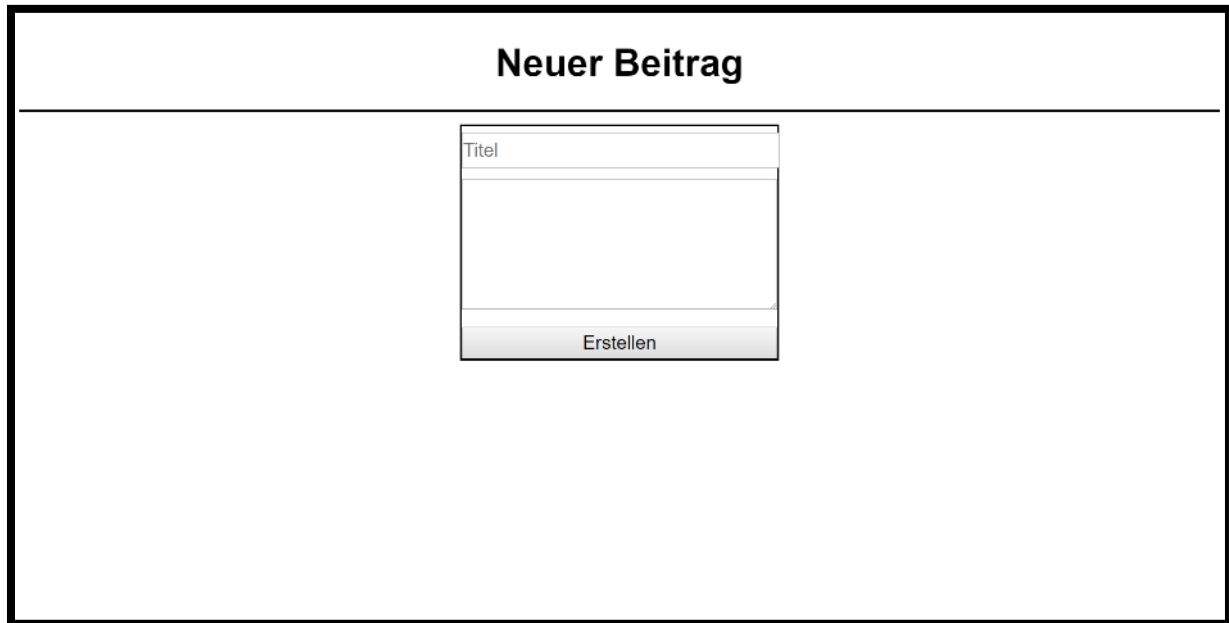
Abbildung 16 - Ergebnis Seite Beiträge

6.2.4 Neuer Beitrag

Die Seite Neuer Beitrag wurde im Frontend nach Plan umgesetzt und gemäss Mockup mit minimalen Abweichungen auch so dargestellt. Dazu gehören die Eingabefelder Titel und Beschreibung, und der Absende-Knopf für das Erstellen des Beitrags.

Design Abweichungen:

- Das Formular wird von einem sichtbaren Rahmen umschlossen



The mockup shows a web form titled "Neuer Beitrag". The form is contained within a large rectangular frame with a thick black border. At the top of this frame, the title "Neuer Beitrag" is centered. Below the title is a horizontal line. In the center of the frame is a smaller, rounded rectangular box. This inner box contains a text input field labeled "Titel", a larger text area for the description, and a button labeled "Erstellen" at the bottom.

Abbildung 17 - Ergebnis Seite Neuer Beitrag

6.2.5 Mein Profil

Die Seite Mein Profil wurde im Frontend nach Plan mit minimalen Abweichungen umgesetzt und gemäss Mockup auch teilweise so dargestellt.

Umgesetzt wurde:

- Eingabefelder
 - Benutzername
 - Vorname
 - Nachname
 - Email
 - Profilbild auswählen
- Knopf zum speichern

Nicht umgesetzt wurde:

- Anzeige des aktuellen Profilbilds

Design Abweichungen:

- Tabellenlinien sind nicht sichtbar
- Profilbild wird nicht angezeigt

Mein Profil

Benutzer-ID: 11 [Neuer Beitrag](#) [Mein Profil](#) [Beiträge](#) [Logout](#)

Benutzername:

Vorname:

Nachname:

Email:

Profilbild: Keine ausgewählt

Abbildung 18 - Ergebnis Seite Mein Profil

6.2.6 Control Panel

Die Seite Control Panel wurde im Frontend nach Plan mit minimalen Abweichungen umgesetzt und gemäss Mockup mit Abweichungen auch so dargestellt.

Umgesetzt wurde:

- Liste Benutzer
 - Benutzername
 - Option Benutzer sperren
- Liste Beiträge
 - Beschreibung
 - Option Beitrag löschen
- Liste gemeldete Beiträge
 - Beschreibung
 - Option Beitrag löschen

Nicht umgesetzt wurde:

- Liste Beiträge
 - Keine Information, ob gemeldet

Design Abweichungen:

- Liste Beiträge und Liste gemeldete Beiträge: Anstatt Titel steht Benutzername
- Seite ist nicht zentriert sondern linksbündig

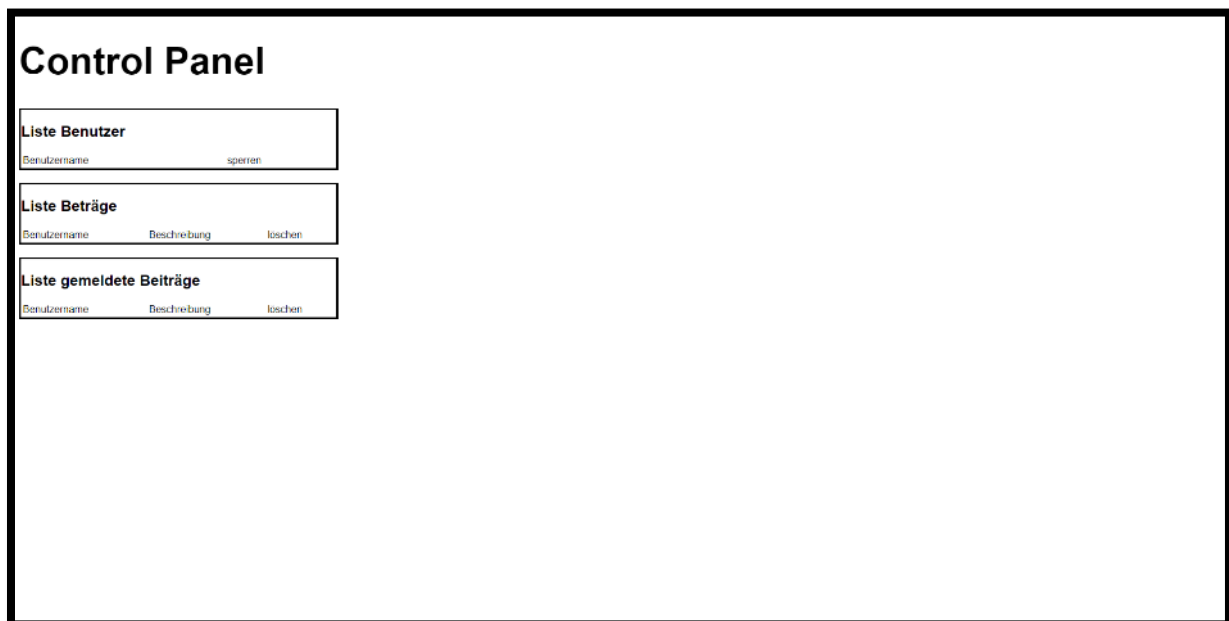


Abbildung 19 - Ergebnis Seite Control Panel

6.3 Implementierung Backend

Weil ich im Zeitplan ein wenig im Rückstand bin, habe ich mich im ersten Schritt nur auf die wesentlichen Funktionen beschränkt. Zuerst habe ich nachdem Klassendiagramm alle Java-Klassen hinzugefügt.

6.3.1 Registrieren

Die Seite Registrieren wurde im Backend nach Plan teilweise umgesetzt.

Umgesetzt wurde:

- Verbindung mit der Datenbank
- Speichern von den Informationen im Benutzer-Objekt
- Registrieren-Funktion des Benutzers
- Passwortverschlüsselung
- Serverseitige Feldüberprüfung auf leeres Feld
- Weiterleitung auf Loginseite

Nicht umgesetzt wurde:

- Weiterverarbeitung des selektierten Profilbildes → Hochladen
- Serverseitige Feldüberprüfung auf fehlerhafte Eingabe
- Vergleich zwischen Passwort und Passwort wiederholen
- Fehlermeldungen bei fehlerhaften Eingaben

6.3.2 Login

Die Seite Login wurde im Backend nach Plan umgesetzt.

Umgesetzt wurde:

- Das Überprüfen des Passwortes mit der Datenbank
- Serverseitige Feldüberprüfung auf leeres Feld
- Fehlermeldung
- Weiterleitung auf Beitragsseite
- Setzen der Session mit den Benutzerinformationen: ID und Rolle

6.3.3 Beitrag

Die Seite Beitrag wurde im Backend nach Plan teilweise umgesetzt.

Umgesetzt wurde:

- Überprüfen ob Benutzer eingeloggt
- Laden und Anzeigen aller Beiträge aus der Datenbank
- Laden und Anzeigen aller Kommentare aus der Datenbank
- Funktion Kommentar Eintrag

Nicht umgesetzt wurde:

- Funktion Beitrag melden
- Funktion Beitrag löschen
- Funktion Gefällt Mir

6.3.4 Neuer Beitrag

Die Seite Neuer Beitrag wurde im Backend nach Plan mehrheitlich umgesetzt.

Umgesetzt wurde:

- Überprüfen ob Benutzer eingeloggt
- Funktion Beitrag erstellen (Eintrag in Datenbank)
- Serverseitige Feldüberprüfung auf leeres Feld
- Weiterleitung auf Beitragsseite

Nicht umgesetzt wurde:

- Fehlermeldungen bei fehlerhafter Eingabe

6.3.5 Mein Profil

Die Seite Mein Profil wurde im Backend nach Plan teilweise umgesetzt.

Umgesetzt wurde:

- Überprüfen ob Benutzer eingeloggt
- Laden der Benutzerdaten aus der Datenbank gemäss eingeloggter Benutzer
- Funktion Benutzerdaten aktualisieren in der Datenbank

Nicht umgesetzt wurde:

- Laden des Profilbildes
- Ändern des Profilbildes
- Serverseitige Überprüfung auf leeres Feld und fehlerhafte Eingaben

6.3.6 Control Panel

Die Seite Control Panel wurde im Backend gar nicht umgesetzt.

Nicht umgesetzt wurde:

- Laden der Benutzer aus der Datenbank
- Laden der Beiträge aus der Datenbank
- Funktion für das Sperren eines Benutzers
- Funktion für das Löschen eines Beitrags
- Überprüfen ob Benutzer eingeloggt
- Überprüfung der Zugriffsrechte

6.3.7 Allgemeine Abweichungen

Nicht in der Aufgabenstellung definiert und doch umgesetzt wurde das Logout. Das Logout war notwendig um im Testing überprüfen zu können, ob der Benutzer danach noch Zugriff auf einzelne Seiten hat.

7 Kontrollieren

7.1 Testing Normalfall

	Testfall Nr. 1	Name Tester: Dennis Schächli
--	----------------	------------------------------

Testname:	Probe-IPA Testing
Absicht:	Normalfall
Eingabedaten:	Buchstaben, Zahlen, Sonderzeichen
Soll-Ergebnis:	Alle Tests laufen durch

Schritt Nr.	Vorgehen	Erwartetes Ergebnis	Abweichungen
1	Registrieren: Eingabe Benutzername: "test"	Daten werden im Input-Feld angezeigt	keine
2	Registrieren: Eingabe Vorname: "test"	Daten werden im Input-Feld angezeigt	keine
3	Registrieren: Eingabe Nachname: "test"	Daten werden im Input-Feld angezeigt	keine
4	Registrieren: Eingabe Email: "test@test.com"	Daten werden im Input-Feld angezeigt	keine
5	Registrieren: Eingabe Passwort: "test"	Daten werden im Input-Feld angezeigt	keine
6	Registrieren: Eingabe Passwort wiederholen: "test"	Daten werden im Input-Feld angezeigt	keine
7	Registrieren: Eingabe Profilbild auswählen: "test.png"	Daten werden im Input-Feld angezeigt	keine
8	Registrieren Formular absenden	Benutzer wird registriert und auf Login-Seite weitergeleitet	keine

9	Login: Eingabe Benutzername: "test"	Daten werden im Input-Feld angezeigt	keine
10	Login: Eingabe Passwort: "test"	Daten werden im Input-Feld angezeigt	keine
11	Login Formular absenden	Benutzer wird eingeloggt und wird auf Beitragsseite weitergeleitet	keine
12	Login Formular absenden mit Benutzername "test@test.com"	Benutzer wird eingeloggt und wird auf Beitragsseite weitergeleitet	Login schlägt fehl
13	Beitrag: Seite aufrufen	Seite wird geladen	keine
14	Beitrag melden	Beitrag wird gemeldet	wurde nicht umgesetzt
15	Beitrag löschen	Beitrag wird gelöscht	wurde nicht umgesetzt
16	Gefällt Mir setzen	Beitrag bekommt ein Gefäll mir	wurde nicht umgesetzt
17	Kommentar schreiben: "Cool!"	Kommentarfeld wird ausgefüllt	keine
18	Kommentar abschicken	Kommentar wird gepostet	Kommentar erscheint nicht, es passiert nichts
19	Neuer Beitrag Seite aufrufen	Seite wird geladen	keine
20	Neuer Beitrag : Titel "test"	Daten werden im Input-Feld angezeigt	keine
21	Neuer Beitrag Beschreibung "test"	Daten werden im Input-Feld angezeigt	keine

22	Neuer Beitrag Formular abschicken	Beitrag wird erstellt und auf Beitragsseite weitergeleitet	keine
23	Mein Profil aufrufen	Seite und entsprechende Benutzerdaten werden geladen	keine
24	Benutzername ändern auf "tester"	Wird in Datenbank geändert	keine
25	Vorname ändern auf "Dennis"	Wird in Datenbank geändert	keine
26	Nachname ändern auf "Schäppi"	Wird in Datenbank geändert	keine
27	Email ändern auf "dennis.schaeppi@siemens.com"	Wird in Datenbank geändert	keine
28	Profilbild ändern	Wird geändert	wurde nicht umgesetzt
29	Admin Control Panel aufrufen	Seite wird geladen	keine
30	Admin Control Panel: Benutzer sperren	Benutzer wird gesperrt	wurde nicht umgesetzt
31	Admin Control Panel: Beitrag löschen	Beitrag wird gelöscht	wurde nicht umgesetzt
32	Troubleshooter Control Panel aufrufen	Seite wird geladen	keine
33	Troubleshooter Control Panel: Beitrag löschen	Beitrag wird gelöscht	wurde nicht umgesetzt

Tabelle 16 - Testing Normalfall

7.2 Testing Extremfall

	Testfall Nr. 2	Name Tester: Dennis Schächli
--	----------------	------------------------------

Testname:	Probe-IPA Testing
Absicht:	Extremfall
Eingabedaten:	Buchstaben, Zahlen, Sonderzeichen
Soll-Ergebnis:	Alle Tests laufen durch

Schritt Nr.	Vorgehen	Erwartetes Ergebnis	Abweichungen
1	Registrieren: Eingabe Benutzername: "täst" und absenden	Wird nicht abgesendet	Benutzer wird trotzdem registriert
2	Registrieren: Eingabe Vorname: "test123" und absenden	Wird nicht abgesendet	Benutzer wird trotzdem registriert
3	Registrieren: Eingabe Nachname: "test" und absenden	Wird nicht abgesendet	Benutzer wird trotzdem registriert
4	Registrieren: Eingabe Email: "test@?täst.com" und absenden	Wird nicht abgesendet	Benutzer wird trotzdem registriert
5	Registrieren: Eingabe Passwort: "test;" und absenden	Wird nicht abgesendet	Benutzer wird trotzdem registriert
6	Registrieren: Eingabe Passwort wiederholen: "test;" und absenden	Wird nicht abgesendet	Benutzer wird trotzdem registriert
7	Registrieren: Profilbild auswählen: "test.xlsx" und absenden	Wird nicht abgesendet	Benutzer wird trotzdem registriert
8	Registrieren Passwort: "test" und Passwort wiederholen: "Test123"	Wird nicht abgesendet	Benutzer wird trotzdem registriert

9	Registrieren Formular leer absenden	Wird nicht abgesendet	keine
10	Login: Formular leer absenden	Wird nicht abgesendet	keine
11	Kommentar leer absenden	Wird nicht abgesendet	Da die Kommentare allgemein nicht gepostet werden und nichts passiert kann nicht beurteilt werden, ob es leer abgesendet wurde oder nicht.
12	Neuer Beitrag Formular leer absenden	Wird nicht abgesendet	keine
13	Mein Profil Benutzername leer absenden	Wird nicht abgesendet	Wird trotzdem abgesendet
14	Mein Profil Vorname leer absenden	Wird nicht abgesendet	Wird trotzdem abgesendet
15	Mein Profil Nachname leer absenden	Wird nicht abgesendet	Wird trotzdem abgesendet
16	Mein Profil Email leer absenden	Wird nicht abgesendet	Wird trotzdem abgesendet
17	Mein Profil: Eingabe Benutzername: "täst" und absenden	Wird nicht abgesendet	Wird trotzdem abgesendet
18	Mein Profil: Eingabe Vorname: "test123" und absenden	Wird nicht abgesendet	Wird trotzdem abgesendet
19	Mein Profil: Eingabe Nachname: "test" und absenden	Wird nicht abgesendet	Wird trotzdem abgesendet
20	Mein Profil: Eingabe Email: "test@?täst.com" und absenden	Wird nicht abgesendet	Wird trotzdem abgesendet
21	Mein Profil: Eingabe Passwort: "test;" und absenden	Wird nicht abgesendet	Wird trotzdem abgesendet

22	Mein Profil: Profilbild auswählen: "test.xlsx" du absenden	Wird nicht abgesendet	Wird trotzdem abgesendet
23	Admin Control Panel als Konsument, Passant und Troubleshooter aufrufen	Zu Login weitergeleitet	Seite wird trotzdem geladen
24	Troubleshooter Control Panel als Konsument, Passant und Admin aufrufen	Zu Login weitergeleitet	Seite wird trotzdem geladen
25	Beitragsseite als Passant aufrufen	Zu Login weitergeleitet	Internal Server Error
26	Neuer Beitrag als Passant aufrufen	Zu Login weitergeleitet	Internal Server Error

Tabelle 17 - Testing Extremfall

7.3 Zusammenfassung der Testings

Art des Testings	Anzahl Tests gesamt	Erfolgreiche Tests	Gescheiterte Tests
Normalfall	33	23	10
Extremfall	26	3	23

Tabelle 18 - Zusammenfassung Testings

Von insgesamt 59 Tests sind 33 Tests gescheitert. Es handelt sich hauptsächlich um Dinge, die aus Zeitgründen nicht realisiert werden konnten oder um Fehlermeldungen innerhalb des Programms.

8 Auswerten

8.1 Endresultat Applikation

8.1.1 Gelungen

Im Grossen und Ganzen ist mir die Applikation nur teilweise gelungen, die wichtigsten Funktionen konnte ich aber umsetzen, dass bei der Demo das ganze verständlich gezeigt werden kann. Der Benutzer kann sich registrieren, einloggen, einen Beitrag verfassen, seine Profildaten aktualisieren und sich ausloggen. Die folgenden Punkte (gemäss Aufgabenstellung Kapitel 2.1.2.1 auf Seite 2) sind mir gelungen und laufen in der Applikation reibungslos:

Nr.	Was	Begründung
1	Multiuser-Applikation	Ja, es handelt sich um eine Multi-User-Applikation, weil sich mehrere Benutzer registrieren und einloggen können.
2	Benutzerrollen	Die einzelnen Benutzerrollen wurden definiert.
2.1	Konsument kann sich registrieren <ul style="list-style-type: none"> • Benutzername • Passwort <ul style="list-style-type: none"> ○ Wird verschlüsselt • E-Mail-Adresse • Vorname • Nachname 	Der Konsument kann sich die daneben genannten Punkte eingeben und sich so registrieren.
2.2	Konsument kann sich einloggen	Der Konsument kann sich mit dem Benutzernamen einloggen.
2.3	Konsument kann sein Profil bearbeiten <ul style="list-style-type: none"> • Vorname • Nachname • Benutzername • Passwort • E-Mail-Adresse 	Der Konsument kann sein Profil bearbeiten und die daneben genannten Punkte ändern.
2.4	Konsument kann Beiträge in Form von Text verfassen	Der Konsument kann einen Beitrag verfassen

8.1.2 Nicht gelungen

Bei der Realisierung musste ich besonders aus Zeitgründen einige Funktionen streichen und Programmfehler stehen lassen, da ich sonst keine Zeit für die Dokumentation gehabt hätte.

Ich konnte diese Punkte nicht realisieren, weil die Zeit zu knapp war und ich für gewisse Programmfehler keine Zeit hatte diese zu realisieren. Ich musste auch aus Zeitgründen gewisse Punkte in der Aufgabenstellung streichen.

Die folgenden Punkte (gemäss Aufgabenstellung Kapitel 2.1.2.1 auf Seite 2) sind mir nicht gelungen:

Nr.	Was	Begründung
1	Konsument kann sich registrieren <ul style="list-style-type: none"> Profilbild 	Der Benutzer kann sein Profilbild nicht setzen, weil ich aus Zeitgründen diese Funktion ausgelassen habe und die Funktion in der Aufgabenstellung nur als Kann-Kriterium definiert habe und keine hohe Priorität hat.
2	Konsument kann sich registrieren <ul style="list-style-type: none"> Passwort wiederholen 	Die beiden Passwörter werden nicht miteinander verglichen. Ich konnte aus Zeitgründen diese Funktion nicht mehr einbauen.
3	Konsument kann sich einloggen	Der Konsument kann sich nicht mit der Email einloggen und ich konnte diesem Problem aus Zeitgründen nicht mehr nachgehen. Wenn ein nicht existierender Benutzername eingegeben wird, dann schlägt das Programm fehl und gibt eine interne Fehlermeldung aus.
4	Konsument kann sein Profil bearbeiten <ul style="list-style-type: none"> Profilbild ändern 	Da ich das Hochladen des Profilbildes nicht realisiert habe, kann der Benutzer sein Profilbild auch nicht ändern, weil ich diese Funktion auch aus Zeitgründen gestrichen habe.
5	Konsument seine eigenen Beträge bearbeiten und löschen. <ul style="list-style-type: none"> Nur Text im Beitrag Bearbeiten mit Option direkt beim Beitrag Löschen mit Option direkt beim Beitrag 	Der Ersteller eines Posts kann seinen Beitrag nicht löschen und bearbeiten. Aus Zeitgründen habe ich diese Funktion gestrichen, weil für die Demo das nichtexistieren dieser Funktionen keinen negativen Einfluss hat.
6	Gefällt-Mir-Funktion» und Kommentarfunktion für alle Konsumenten bei allen Beiträgen	Die Gefällt Mir Funktion wurde ebenfalls ausgelassen, weil ich aus Zeitgründen dies nicht mehr realisieren konnte. Die Kommentare werden nicht in die Datenbank geschrieben und es kommt eine interne Fehlermeldung. Auf die Demo hat dies keinen grossen Einfluss.
7	Beiträge können gemeldet werden	Die Melden-Funktion wurde ebenfalls aus Zeitgründen nicht realisiert.
8	Control Panels	Alle Control Panels wurden auch aus Zeitgründen nicht realisiert, da habe ich die Zeitplanung unterschätzt und musste dies streichen. Für die Demo ist es unvorteilhaft, da dies einer der

		spannendsten aber nicht dringendsten Seiten war.
9	Benutzer haben keine Zugriffsrechte auf bestimmte Seiten	In jedem Fall wird der Benutzer, wenn er keine Berechtigung für diese Seite hat, nicht auf die Login-Seite weitergeleitet, sondern es kommt ein interner Programmfehler. Aus Zeitgründen konnte ich dies nicht mehr vertieft anschauen und ich musste es so lassen.

8.2 Reflexion

8.2.1 Was war gut

Ich habe das Projekt gut planen können, ich habe Diagramme gezeichnet und nach diesem das Projekt realisiert. Dies hat den Prozess beim Realisieren beschleunigt. Für die Dokumentation habe ich genügend Zeit eingeplant und auch viel herausholen. Das Erstellen aller Grafischen Oberflächen verlief auch sehr gut und ich konnte wie im Fluss anhand meiner Mockups die GUIs erstellen. Ich habe es geschafft wieder in die Soll Zeit zu kommen und konnte daraus viel lernen, dass es Abweichungen geben kann. Das Implementieren der Klassen konnte ich nach meinem erstellten Klassendiagramm erstellen und die Variablen und Methoden auch so erstellen. Aus meinen Arbeitsjournalen konnte ich den Tag zurückblicken und konnte am nächsten Tag wieder einsteigen, und das mit ganz anderen Ansichten und Gedanken. Im Grossen und Ganzen verliefen die Planung, das Entscheiden, das Kontrollieren und das Auswerten relativ gut.

8.2.2 Was war nicht gut

Das Projekt verlief mehrheitlich nicht so gut. Bei der Zeitplanung habe den Zeitaufwand unterschätzt und ich wusste nicht, dass man mehr für die Planung investieren musste und das Realisieren nicht so gross war wie das Planen. Das hatte die Konsequenzen, dass ich mehrere Teile aus der Aufgabenstellung streichen musste und Programmfehler nicht vertieft anschauen und lösen konnte. Bei der Planungsphase war ich mit den Diagrammen so stark beschäftigt, dass ich auch hier viel Zeit verlor. Beim Realisieren war ich mit einem kleineren Schönheitsfehler so stark beschäftigt, dass ich ziemlich stark im Rückzug war. Für die restlichen Funktionen hat es deshalb nicht ganz gereicht.

8.2.3 Was habe ich gelernt

Ich habe gelernt, dass ich die Planung auf keinen Fall unterschätzen darf und deshalb das Planen grösser als das Realisieren ist. Ich habe gelernt, dass das Use-Case-Diagramm in die Phase des Informierens gehört und dass auch solche Diagramme nicht unterschätzt werden dürfen. Gelernt habe ich auch, dass die Grafiken viel einfacher dargestellt werden können und ich nicht so einen Aufwand hätte betreiben müssen.

8.2.4 Fazit

Insgesamt war diese Probe-IPA ein sehr lehrreiches Projekt für mich.

9 Arbeitsprotokolle

9.1 Arbeitsprotokoll vom 31.10.2018

Nr.	Phase	Tätigkeiten	Soll	Ist
1	Informieren	Aufgabenstellung	1.5h	1h
2	Informieren	Management Summary	1.5h	1.5h
3	Planen	Zeitplan	1h	1h
4	Dokumentation	Informieren dokumentiert und ergänzt	2h	3h
Total			6h	6.5h

9.1.1 Erfolge und Misserfolge

Heute war mein erster Tag bei der Probe-IPA. Am Morgen erstellte ich einen Zeitplan und überarbeitete meine Aufgabenstellung, da diese noch zu wenig ausführlich war. Die verantwortliche Fachkraft bestätigte mir, dass die Aufgabenstellung so in Ordnung war. Der Start war ein wenig turbulent, da wir am Morgen zuerst den Zeitplan erstellen mussten und erst danach mit der Dokumentation anfangen konnten, weil die verantwortliche Fachkraft dies so haben wollte. Deshalb musste ich die Informieren- und Planen-Phase ein wenig mischen, jedoch half mir der Zeitplan sehr, mich an den bevorstehenden Aufgaben zu orientieren.

9.1.2 Probleme

Ich habe den Zeitaufwand beim Informieren unterschätzt und plante deshalb auch, heute das GUI-Design, das Use-Case-Diagramm und das Klassendiagramm fertig zu haben.

9.1.3 Überzeiten

Heute musste ich eine halbe Stunde Überzeit machen, weil ich den Teil des Informierens unbedingt fertig haben wollte.

9.1.4 Reflexion

Der heutige Tag verlief im Grossen und Ganzen relativ gut. Ich bin im Zeitplan im Rückstand. Ich hätte heute bereits das GUI-Design, das Use-Case-Diagramm und das Klassendiagramm fertig haben sollen.

Die Planung werde ich morgen fertig haben und mit einem kleinen Teil in der Realisierung beginnen können.

9.2 Arbeitsprotokoll vom 01.11.2018

Nr.	Phase	Tätigkeiten	Soll	Ist
1	Informieren	UseCase-Diagramm	1h	1h
2	Planen	Datenbankmodell	1h	2h
3	Planen	Klassendiagramm	1.5h	2h
4	Planen	GUI Mockup	1h	1h
5	Dokumentation	Planen Grafiken hinzugefügt und teilweise dokumentiert.	1h	1h
6	Expertengespräch	Hilfreiche Tipps bekommen	0.5h	0.5h
Total			6h	7.5h

9.2.1 Erfolge und Misserfolge

Der zweite Tag der Probe-IPA verlief sehr produktiv. Ich hatte das erste Gespräch mit dem Experten und habe mir alles Wichtige notiert bei der Besprechung, um diese Arbeit noch mehr zu optimieren. Das Gespräch verlief gut. Leider konnte ich ihm noch nicht viel zeigen, aber dafür war das Gespräch nicht zu lange und ich konnte wieder zurück an die Arbeit. Ich konnte das UseCase- und Klassendiagramm, das Datenbankmodell und das GUI Mockup erstellen. Beim Datenbankdesign habe ich sehr viel Zeit verloren, weil ich im Visio immer zwischen den Optionen wechseln musste. Das Gleiche war auch beim Klassendiagramm der Fall. Das Mockup konnte ich schnell erstellen und ich kam beim Mockup gut durch. Die Dokumentation musste ich laufend ergänzen und ich wurde mit dem auch nicht ganz fertig, was bedeutet, dass ich noch mehr Zeit verliere. Ich konnte nicht mit der Entscheidungsphase beginnen.

9.2.2 Probleme

Das Erstellen des Datenbankmodells und des Klassendiagramms war ein Problem, denn Visio verschob mir alle Elemente immer wieder herum, so dass ich die Dokumente mehrmals wiederherstellen musste. Ich habe die gesamte Zeitplanung unterschätzt und es wird voraussichtlich ein Teil beim Realisieren wegfallen.

9.2.3 Überzeiten

Heute musste ich anderthalb Stunden Überzeit machen, weil ich beim Datenbankmodell und beim Klassendiagramm den Zeitaufwand unterschätzt habe.

9.2.4 Reflexion

Der heutige Tag verlief am Anfang gut und am Ende nicht mehr so gut, weil mir ein Teil der Dokumentation fehlte und ich nicht mit dem Entscheiden beginnen konnte, was heute hätte fertig werden müssen. Am Morgen hatte ich ein gutes Expertengespräch, welches nicht zu lange dauerte. Morgen werde im ersten Block die Dokumentation für das Planen und Entscheiden abschliessen und werde im zweiten und dritten Block mit dem Realisieren starten. In dem ich heute sehr viele hilfreiche Diagramme erstellt habe, werde ich genau wissen, was wo implementiert wird und das beschleunigt den Prozess beim Realisieren. Ich darf auch nicht vergessen, immer wieder zu dokumentieren, denn das ist das Wichtigste bei dieser Arbeit. Ich habe geplant, dass ich in der letzten Stunde nur noch dokumentieren werde und das Arbeitsjournal des Tages zu führen.

9.3 Arbeitsprotokoll vom 02.11.2018

Nr.	Phase	Tätigkeiten	Soll	Ist
1	Entscheiden	Visual Studio oder Eclipse Dynamisch oder statisch SQL oder MSSQL	0.5h	0.5h
2	Realisieren	GUI Frontend erstellt	2h	1.5h
3	Realisieren	Backend Login implementiert	2.5h	3.5h
4	Dokumentation	Entscheidung und Prozess dokumentiert	1h	1h
Total			6h	6.5h

9.3.1 Erfolge und Misserfolge

Heute konnte ich das Planen abschliessen, ich musste noch etwas Kleines im Klassendiagramm ändern. Danach konnte ich direkt mit dem Entscheiden beginnen, wo ich zuerst für die Datenbankumgebung, für die Entwicklungsumgebung und für dynamische oder statische Website. Ich fing danach mit dem Realisieren an, gleich mit dem Aufsetzen der Datenbank, was ohne Probleme funktioniert hat. Ich hatte ein schönes ERM von der Planung vor mir und wusste genau, was ich tun musste. Beim Realisieren hatte ich heute auch einige Misserfolge, denn ich hatte bereits beim Login Probleme mit dem Backend. Der grösste Misserfolg heute war, dass ich zu lange an einem Punkt stecken geblieben bin, dass ich nicht weiterging. Das Gute beim Realisieren heute war, dass ich alle Seiten mit den nötigen HTML-Elementen hinzufügen konnte, ohne das CSS und den Hintergrundfunktionen.

9.3.2 Probleme

Beim Implementieren des Backends für das Login gab es Probleme bei der falschen Eingabe, ein Error. Ich beharrte darauf, dass ich den Fehler finden musste, dass ich vergessen habe, die anderen fälligen Punkte zu realisieren. Die Folge: Stress und Rückstand. Das Realisieren nach dem Klassendiagramm hat einige Abweichungen und deshalb verlangsamt sich dieser Prozess auch, wenn ich die Funktionen ändern muss.

9.3.3 Überzeiten

Heute musste ich eine halbe Stunde Überzeit machen, weil ich beim Login zu sehr am Backend hängengeblieben war.

9.3.4 Reflexion

Der heutige Tag verlief beim Entscheiden gut und danach eher schlechter. Ich hing zu lange am Login mit dem Backend fest, so dass ich vergessen habe, dass an einer IPA nicht alles perfekt sein muss, sondern entweder realisiert oder dokumentiert. Ich werde im nächsten Schritt alle Nebenfunktionen streichen und nur das Wichtigste machen. Jetzt, wo ich die erste Datenbankabfrage gemacht habe, habe ich eine Vorlage für alle anderen Abfragen, die bevorstehen, hier werde ich viel Zeit sparen können und jetzt bin ich wieder mit Programmieren routinierter, denn ich habe in den letzten zwei Tagen fast nur Word, Excel und Visio gesehen. Ich werde aber auch genügend Zeit für das Kontrollieren und Auswerten investieren, am besten einen halben oder ganzen Tag.

9.4 Arbeitsprotokoll vom 07.11.2018

Nr.	Phase	Tätigkeiten	Soll	Ist
1	Realisieren	CSS und Funktionen Beitragsseite	1h	2h
2	Realisieren	CSS und Funktionen Mein Profil	1h	1h
3	Realisieren	CSS und Funktionen Registrieren	1h	1.5h
4	Realisieren	CSS und Funktionen Neuer Beitrag	1h	1h
5	Realisieren	Logout	0.25h	0.25h
6	Realisieren	CSS Login	0.25h	0.25h
7	Realisieren	Kommentare im Code	0.5h	0.5h
8	Dokumentation	Realisieren dokumentiert	1h	0.5h
Total			6h	6h

9.4.1 Erfolge und Misserfolge

Heute konnte ich den ganzen Tag programmieren und das verlief ohne grosse Probleme. Beim Implementieren der Funktionen geriet ich im Code ein wenig in ein Chaos, was nach dem auskommentieren wieder übersichtlicher war. Nicht alle Funktionen sind schön gekapselt und das hätte ich auch schöner gemacht, wenn ich mehr Zeit gehabt hätte. Die geplanten Funktionen haben am Ende alle funktioniert, nur bei den Extremfällen schlägt das Programm fehl. Ich konnte heute einige Sachen aus meinem Programm streichen, weil ich sonst nicht damit fertig werde. Ich habe zusammen mit den Funktionen auch das Design umgesetzt, was auch nicht ganz so herausgekommen ist wie geplant und ich wollte keine Zeit verlieren, denn die Dokumentation war mir wichtiger. Dies aber eher Schönheitsfehler, es war das Positionieren in der Mitte.

9.4.2 Probleme

Bei der Implementation gab es mehrere Crashes im Programm. Das Programm erträgt kein Testing mit Extremwerten, was nicht gut ist. Ich musste auch mehr als die Hälfte der Funktionen streichen, weil ich sonst die Arbeit nicht in der vorgegebenen Zeit abschliessen kann.

9.4.3 Überzeiten

Heute konnte ich die 6 Stunden auf Punktlandung füllen, ich habe bei einigen Dingen mehr Zeitgebraucht, diese aber auch wieder kompensiert.

9.4.4 Reflexion

Der heutige Tag verlief im Grossen und Ganzen gut. Ausser von ein paar Problemen beim Backend konnte ich alles ruckelfrei implementieren. Ich musste Mehrere Funktionen streichen, weil ich die Arbeit sonst nicht fertigbringe. Ich habe nur die wichtigsten Funktionen implementiert: Login, Beitrag, Neuer Beitrag, Mein Profil und Logout. Beiträge melden und löschen wurde gestrichen, weil die Zeit zu knapp wäre. Das Control Panel sehe ich auch nicht als wichtige Funktion und deshalb habe ich dies auch aus Zeitgründen gestrichen. Das Schreiben der Kommentare endet in einem Crash und nach 30 Minuten Fehlersuche musste ich das aufgeben und mit den anderen Funktionen fortfahren, da ich sonst fast nichts vom Programm gehabt hätte. Das weitere Vorgehen ist auf jeden Fall die Dokumentation zu beenden. Ich werde im Programm nichts mehr ändern, ich lasse es so und dokumentiere weiter und liste alle Funktionen auf, die nicht funktioniert haben, das einzige, was ich noch mache, ist den Code zu kommentieren. Ich werde im letzten Tag das Programm testen und auswerten, dann bin ich auch wieder im Zeitplan drinnen.

9.5 Arbeitsprotokoll vom 09.11.2018

Nr.	Phase	Tätigkeiten	Soll	Ist
1	Realisieren	Code Kommentiert	1h	1h
2	Dokumentation	Realisieren dokumentiert	3h	3h
3	Expertengespräch		0h	0,25
4	Kontrollieren	Testing	1h	1h
5	Auswerten	Rückblick & Reflexion	0,5h	1h
6	Auswerten	Code exportieren	0,5h	0,25
Total			6h	7h

9.5.1 Erfolge und Misserfolge

Heute war der Endspurt dieser Arbeit. Ich habe noch letzte Stellen im Code kommentiert und das Testing durchgeführt, das Testing verlief gut und ich konnte die Probleme alle beschreiben. Ich hatte heute ein Expertengespräch und wir konnten die allfälligen Probleme besprechen. Das Kopieren des Codes im Eclipse verlief gut, ausser mit dem SQL-Code, der nicht farbig markiert ist.

9.5.2 Probleme

Heute gab es keine grossen Probleme mehr, ich geriet am Ende ein wenig in den Zeitdruck. Das Kopieren des SQL-Codes war wegen des Editors nicht möglich und wegen Zeitdruck nicht mehr

9.5.3 Überzeiten

Heute habe ich viel Überzeit gemacht, weil heute Abgabe war.

9.5.4 Reflexion

Der heutige Tag war sehr turbulent, weil die Abgabe war. Der Morgen verlief eher entspannend und der Nachmittag war ein wenig turbulenter, wegen dem Zeitdruck. Das Projekt war für mich im Grossen und Ganzen sehr lehrreich und habe viel daraus gelernt. Mit der Dokumentation geriet ich am Enden noch ein wenig in Zeitdruck, weil ich nicht so schnell schreiben konnte, wenn ich gestresst war. An der richtigen IPA kann ich mir sehr gut vorstellen, dass man für das Realisieren mehr Zeit hat und das Planen gleich lange dauert wie hier an der Probe-IPA.

10 Verzeichnisse

10.1 Glossar

ERM	Datenmodell
GUI	Grafische Benutzeroberfläche
UseCase	Anwendungsfall

10.2 Abbildungsverzeichnis

Abbildung 1 - IPERKA-Modell	4
Abbildung 2 - Zeitplan	5
Abbildung 3 - UseCase-Diagramm	8
Abbildung 4 - Datenmodell Planung	10
Abbildung 5 - Klassendiagramm	13
Abbildung 6 - GUI Registrieren	14
Abbildung 7 - GUI Login	15
Abbildung 8 - GUI Beiträge	16
Abbildung 9 - GUI Neuer Beitrag	17
Abbildung 10 - GUI Mein Profil	18
Abbildung 11 - GUI Admin Control Panel	19
Abbildung 12 - GUI Troubleshooter Panel	20
Abbildung 13 - Datenmodell nach Realisierung	23
Abbildung 14 - Ergebnis Seite Registrieren	24
Abbildung 15 - Ergebnis Seite Login	25
Abbildung 16 - Ergebnis Seite Beiträge	26
Abbildung 17 - Ergebnis Seite Neuer Beitrag	27
Abbildung 18 - Ergebnis Seite Mein Profil	28
Abbildung 19 - Ergebnis Seite Control Panel	29

10.3 Tabellenverzeichnis

Tabelle 1 - Auflistung Vorkenntnisse	4
Tabelle 2 - Auflistung Meilensteine	6
Tabelle 3 - Involvierte Personen	6
Tabelle 4 - Detailplanung Tabelle für Datenbank	9
Tabelle 5 - Berechtigungsmatrix Benutzerrollen	9
Tabelle 6 - Legende für Abbildung 4	10
Tabelle 7 - Planung Stored Procedures	11
Tabelle 8 - Auflistung erlaubter Zeichen beim Registrieren	11
Tabelle 9 - Auflistung Kriterien Datenbankumgebung	21
Tabelle 10 - Entscheidungsmatrix Datenbankumgebung	21
Tabelle 11 - Auflistung Kriterien Entwicklungsumgebung	21
Tabelle 12 - Entscheidungsmatrix Entwicklungsumgebung	21
Tabelle 13 - Auflistung Kriterien Statisch/Dynamisch	22
Tabelle 14 - Entscheidungsmatrix Statisch/Dynamisch	22
Tabelle 15 - Zugriffsdaten Datenbank	23
Tabelle 16 - Testing Normalfall	34
Tabelle 17 - Testing Extremfall	37
Tabelle 18 - Zusammenfassung Testings	37

11 Anhang

11.1 Klassen

11.1.1 Beitrag.java

```
1 package App;
2
3 import java.sql.Connection;
4
5
6 public class Beitrag {
7     //Gekapselte Variablen: Informationen zum Beitrag
8     private int id;
9     private String titel;
10    private String beschreibung;
11    private Date datum;
12    private String benutzername;
13    private int gemeldet;
14
15    //Variablen für Datenbankabfragen
16    private Datenbank db = new Datenbank();
17    private PreparedStatement ps;
18    private Connection con;
19    private ResultSet rs;
20
21    //Konstruktor
22    public Beitrag(Connection con){
23        this.con = con;
24    }
25
26    //Getter und Setter für Id
27    public int getId() {
28        return id;
29    }
30    public void setId(int id) {
31        this.id = id;
32    }
```



```
42 }
43
44 //Getter und Setter für titel
45 public String getTitel() {
46     return titel;
47 }
48 public void setTitel(String titel) {
49     this.titel = titel;
50 }
51
52 //Getter und Setter für Beschreibung
53 public String getBeschreibung() {
54     return beschreibung;
55 }
56 public void setBeschreibung(String beschreibung) {
57     this.beschreibung = beschreibung;
58 }
59
60 //Getter und Setter für Datum
61 public Date getDatum() {
62     return datum;
63 }
64 public void setDatum(Date datum) {
65     this.datum = datum;
66 }
67
68 //Getter und Setter für Benutzername
69 public String getBenutzername() {
70     return benutzername;
71 }
72 public void setBenutzername(String benutzername) {
73     this.benutzername = benutzername;
74 }
75
76 //Getter und Setter für Gemeldet
```

```
77 public int getGemeldet() {
78 return gemeldet;
79 }
80
81 public void setGemeldet(int gemeldet) {
82 this.gemeldet = gemeldet;
83 }
84
85 //Schreibt einen Kommentar
86 public void schreibeKommentar(int beitrag, int benutzer, String text){
87 //Aktuelles Datum ermitteln
88 Calendar cal = new GregorianCalendar();
89 java.util.Date dt = cal.getTime();
90 Date sdt = new Date(dt.getTime());
91
92 //Führt Datenmanipulation aus: schreibeKommentar()
93 try {
94 ps = con.prepareStatement("CALL schreibeKommentar(?, ?, ?, ?)");
95 ps.setInt(1, beitrag);
96 ps.setInt(2, benutzer);
97 ps.setString(3, text);
98 ps.setDate(4, sdt);
99 ps.executeUpdate();
100
101 } catch (SQLException e) {
102 e.printStackTrace();
103 }
104 }
105
106 //Erstellt einen neuen Beitrag
107 public void neu(String titel, String beschreibung, Date datum, int benutzer){
108
109 //Führt Datenmanipulation aus: erstelleBeitrag()
110 try {
111 ps = con.prepareStatement("CALL erstelleBeitrag(?, ?, ?, ?)");
112 ps.setString(1, titel);
113 ps.setString(2, beschreibung);
```

```
114 ps.setDate(3, datum);
115 ps.setInt(4, benutzer);
116 ps.executeUpdate();
117 } catch (SQLException e) {
118 e.printStackTrace();
119 }
120 }
121
122
123 }
124
```

11.1.2 Benutzer.java

```
1 package App;
2
3 import java.sql.Connection;
4
5
6
7
8
9
10 public class Benutzer {
11 //Gekapselte Variablen: Benutzerinformationen
12 private int id;
13 private String benutzername;
14 private String vorname;
15 private String nachname;
16 private String email;
17 private String password;
18 private String benutzerrolle;
19 private int gesperrt;
20 private String profilbildurl;
21
22 //Variablen für Datenbankabfragen
23 private Datenbank db = new Datenbank();
24 private Login login = new Login();
25 private PreparedStatement ps;
26 private Connection con = db.getConnect();
27 private ResultSet rs;
28
```

```
29 //Getter und Setter für Id
30 public int getId() {
31     return id;
32 }
33 public void setId(int id) {
34     this.id = id;
35 }
36
37 //Getter und Setter für Benutzername
38 public String getBenutzername() {
39     return benutzername;
40 }
41 public void setBenutzername(String benutzername) {
42     this.benutzername = benutzername;
43 }
44
45 //Getter und Setter für Vorname
46 public String getVorname() {
47     return vorname;
48 }
49 public void setVorname(String vorname) {
50     this.vorname = vorname;
51 }
52
53 //Getter und Setter für Nachname
54 public String getNachname() {
55     return nachname;
56 }
57 public void setNachname(String nachname) {
58     this.nachname = nachname;
59 }
60
61 //Getter und Setter für Email
62 public String getEmail() {
63     return email;
64 }
65 public void setEmail(String email) {
```

```
66 this.email = email;
67 }
68
69 //Getter und Setter für Passwort
70 public String getPassword() {
71     return password;
72 }
73 public void setPassword(String password) {
74     this.password = password;
75 }
76
77 //Getter und Setter für Benutzerrolle
78 public String getBenutzerrolle() {
79     return benutzerrolle;
80 }
81 public void setBenutzerrolle(String benutzerrolle) {
82     this.benutzerrolle = benutzerrolle;
83 }
84
85 //Getter und Setter für Gesperrt
86 public int getGesperrt() {
87     return gesperrt;
88 }
89 public void setGesperrt(int gesperrt) {
90     this.gesperrt = gesperrt;
91 }
92
93 //Getter und Setter für Profilbild URL
94 public String getProfilbildurl() {
95     return profilbildurl;
96 }
97 public void setProfilbildurl(String profilbildurl) {
98     this.profilbildurl = profilbildurl;
99 }
100
101 //Sperrt einen Benutzer
102 public void sperren(int id){
```

```
103
104 }
105
106 //Aktualisiert die Benutzerdaten
107 public void datenAendern(int id, String bn, String vorname, String nachname, String
email){
108
109 //Führt Datenmanipulation aus: aendereBenutzerdaten()
110 try {
111 ps = con.prepareStatement("CALL aendereBenutzerdaten(?, ?, ?, ?, ?)");
112 ps.setInt(1, id);
113 ps.setString(2, bn);
114 ps.setString(3, vorname);
115 ps.setString(4, nachname);
116 ps.setString(5, email);
117 ps.executeUpdate();
118
119 } catch (SQLException e) {
120 e.printStackTrace();
121 }
122 }
123
124 //Registriert einen neuen Benutzer
125 public void registrieren(String benutzername, String passwort, String email, String
vorname, String nachname, String profilbildurl){
126
127 //Führt Datenmanipulation aus: register()
128 try {
129 ps = con.prepareStatement("CALL register(?, ?, ?, ?, ?)");
130 ps.setString(1, benutzername);
131 ps.setString(2, vorname);
132 ps.setString(3, nachname);
133 ps.setString(4, email);
134 ps.setString(5, "-");
135 ps.executeUpdate();
```

```
136
137 //Lädt den neu registrierten Benutzer aus der Datenbank
138 ps = con.prepareStatement("CALL loadBenutzer(?");
139 ps.setString(1, benutzername);
140 rs = ps.executeQuery();
141
142
143 if(rs.next()){
144 try{
145 //Setzt die Id des neuen Benutzers
146 setId(rs.getInt("id_benutzer"));
147
148 //Fügt das verschlüsselte Passwort ein
149 ps = con.prepareStatement("CALL registerPasswort(?, ?)");
150 ps.setInt(1, getId());
151 ps.setString(2, login.getHashedPasswort(getId()+password)); //Passwort
verschlüsselt mit ID und Passworttext
152 ps.executeUpdate();
153
154 }catch(NumberFormatException e){
155 e.printStackTrace();
156 }
157 }
158
159 } catch (SQLException e) {
160 e.printStackTrace();
161 }
162
163
164 }
165
166
167
168 }
169
```

11.1.3 Login.java

Login.java

```
1 package App;
2
3 import java.security.MessageDigest;
4
5
6
7
8
9
10
11
12 public class Login {
13
14 //Variablen für Datenbankabfragen
15 private Datenbank db = new Datenbank();
16 private PreparedStatement ps;
17 private Connection con;
18 private ResultSet rs;
19
20 //Gekapselte Variablen: Logininformationen
21 private String username = null;
22 private String password = null;
23 private Integer id2;
24 private Boolean logged = false;
25 private int rolle;
26
27 //Konstruktor 1: Für Login
28 public Login(String benutzername, String passwort){
29 con = db.getConnect();//Verbindet mit Datenbank
30 setLogged(checkValidUser(benutzername, passwort));//Überprüft Benutzerdaten und
    Passwort
31
32 }
33
34 //Konstruktor 2: Lediglich für Funktionsaufruf
35 public Login(){
36
37 }
38
39 //Überprüft Benutzerdaten und Passwort
40 private Boolean checkValidUser(String benutzername, String passwort){
```



```
41 try {
42   ///Lädt Benutzer aus Datenbank
43   ps = con.prepareStatement("CALL loadBenutzer(?)");
44   ps.setString(1, benutzername);
45   rs = ps.executeQuery();
46
47   ///Setzt die Variablen mit den ermittelten Benutzerdaten
48   if(rs.next()){
49     id2 = rs.getInt("id_benutzer");
50     username = rs.getString("benutzername");
51     password = rs.getString("passwort");
52     rolle = rs.getInt("fs_benutzerrolle");
53
54     System.out.println("BN: "+username);
55   }
56
57   if(password.equals(getHashedPasswort(id2+password))){
58     return true;
59   }
60
61   } catch (SQLException e) {
62
63   }
64   return false;
65 }
66
67 ///Verschlüsselt das Passwort
68 public String getHashedPasswort(String password){
69   try {
70     MessageDigest md5 = MessageDigest.getInstance("MD5"); ///Einwegverschlüsselungsart
71     auswählen
72     byte[] array = md5.digest(password.getBytes());///Wandelt das eingegebene Passwort
73     in Bytes um
74     StringBuffer sb = new StringBuffer();
75     for (int i = 0; i < array.length; ++i) {
```

```
74 sb.append(Integer.toHexString((array[i] & 0xFF) | 0x100).substring(1,3));
//Wandelt jedes Zeichen in HEX um
75 }
76 return sb.toString();
77 } catch (NoSuchAlgorithmException e) {
78 }
79 return null;
80 }
81
82 //Getter und Setter für Logged
83 public void setLogged(Boolean b){
84 this.logged = b;
85 }
86
87 public Boolean getLogged(){
88 return logged;
89 }
90
91 //Getter für Id
92 public int getId(){
93 return id2;
94 }
95
96 //Getter für Benutzerrolle
97 public int getRolle(){
98 return rolle;
99 }
100
101
102 }
103
Page 2
```

11.1.4 Datenbank.java

```
1 package Database;
2
3 import java.sql.Connection;
```

```
10
11 public class Datenbank {
12 //Zugangsdaten für Datenbank
13 private String server = "localhost";
14 private String user = "root";
15 private String password = "";
16 private String datasource = "probeipa_sozialesnetzwerk";
17 private Connection connect;
18
19 public Datenbank(){
20 //URL zum Verbinden zu MySQL
21 String url = "jdbc:mysql://" + server + "/" + datasource + "?user=" + user +
"&password=" + password + "&serverTimezone=UTC";
22 try {
23 //JDBC-Treiber laden
24 Class.forName("com.mysql.cj.jdbc.Driver");
25 try {
26 //Verbindung aufbauen
27 connect = DriverManager.getConnection(url);
28 } catch (SQLException e) {
29 e.printStackTrace();
30 }
31 } catch (ClassNotFoundException e) {
32 e.printStackTrace();
33 }
34
35 }
36
37 //Verbindung aufbauen
38 public Connection getConnect(){
39 return this.connect;
40 }
41 }
42
```

11.2 Websites

11.2.1 Beitrag.jsp

```
<%@page import="java.util.Enumeration"%>
<%@page import="java.util.Map"%>
<%@page import="java.sql.Connection"%>
<%@page import="java.util.HashMap"%>
<%@page import="App.Beitrag"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.sql.PreparedStatement"%>
<%@page import="Database.Datenbank"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>

<%
    //Überprüfen ob eingeloggt
    if(session.getAttribute("id").toString().isEmpty()){
        response.sendRedirect("login.jsp");
    }

    //Datenbankverbindung herstellen
    Datenbank db = new Datenbank();
    Connection con = db.getConnect();

    //Benutzer-ID gesetzt
    int benutzer = Integer.parseInt(session.getAttribute("id").toString());

    //Variablen für Datenbankabfragen
    PreparedStatement ps;
    PreparedStatement psCom;
    ResultSet rs;
    ResultSet rsCom;

    //Liste der vorhandenen Beiträge
    HashMap<Integer, Beitrag> beitraege = new HashMap<Integer, Beitrag>();

    //Lade alle Beiträge
    ps = con.prepareStatement("CALL loadBeitraege()");
```

```
rs = ps.executeQuery();

while(rs.next()){
    //Erstelle ein neues Objekt für Beitrag
    Beitrag beitrage = new Beitrag(db.getConnection());
    //Die ermittelten Informationen in das Objekt speichern
    beitrage.setBenutzername(rs.getString("benutzername"));
    beitrage.setTitel(rs.getString("titel"));
    beitrage.setBeschreibung(rs.getString("beschreibung"));
    beitrage.setDatum(rs.getDate("datum"));
    beitrage.setId(rs.getInt("id_beitrage"));

    //Schreibe den Beitrag in die Liste
    beitraege.put(beitrage.getId(), beitrage);
}

//Ermitteln aller Formular-Elemente
Enumeration<String> en = request.getParameterNames();
while(en.hasMoreElements()){

    String parameter = en.nextElement();

    //Wenn Kommentar abgesendet dann in DB schreiben
    if(parameter.contains("comment-writer")){
        int beitrage = Integer.parseInt(parameter.split("-")[2]);
        Beitrag b = new Beitrag(db.getConnection());

        b.schreibeKommentar(beitrage, benutzer, request.getParameter(parameter));
    }
}

%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
    <link rel="stylesheet" type="text/css" href="style/style.css">
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
    <title>Beiträge</title>
</head>
```

```

<body>
  <div class="title"><h1>Beiträge</h1></div>

  <div class="menu">
    <p class="logged">Benutzer-ID: <%=session.getAttribute("id") %></p>
    <p><a href="neuerbeitrag.jsp">Neuer Beitrag</a><p>
    <p><a href="meinprofil.jsp">Mein Profil</a><p>
    <p><a href="beitrag.jsp">Beiträge</a><p>
    <p><a href="logout.jsp">Logout</a><p>
  </div>

  <% for(Map.Entry<Integer, Beitrag> hmb : beitraege.entrySet()){
    int id = hmb.getKey();
    Beitrag b = hmb.getValue();
  %>

    <div class="container">
      <div class="infohead">
        <p>Von: <%=b.getBenutzername() %>      Gepostet am: <%=b.getDatum() %></p>
        <!-- <p><a>Melden</a><a>Löschen</a></p>-->
      </div>
      <div class="content-beitrag">
        <h3><%=b.getTitel() %></h3>
        <p>
          <%=b.getBeschreibung() %>
        </p>
      </div>
      <!-- <div class="gefaellt">
        <p>Gefällt</p>
      </div>-->
      <div class="comment">
        <div class="write-comment">
          <form method="post">
            <input name="comment-writer-<%=b.getId() %>" type="text" placeholder="Kommentar
schreiben">
            <input type="submit" value="Senden">
          </form>
        </div>
      </div>
    </div>
  }

```

```

        <div class="comment-container">
            <%
                psCom = con.prepareStatement("CALL loadKommentare(?)");
                psCom.setInt(1, b.getId());
                rsCom = psCom.executeQuery();

                while(rsCom.next()){
                    <p>Kommentar von: <%=rsCom.getString("benutzername") %>
                        <br>
                        <%=rsCom.getString("text") %>
                    </p>
                }
            <%>
        </div>
    </div>
</div>

<% } %>
</body>
</html>

```

11.2.2 Controlpanel.jsp

```

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
    <link rel="stylesheet" type="text/css" href="style/style.css">
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
    <title>Login</title>
</head>
<body>
    <h1>Control Panel</h1>
    <div class="container">

```

```
<h2>Liste Benutzer</h2>
<table>
  <tr>
    <td>Benutzername</td>
    <td><a>sperr</a></td>
  </tr>
</table>
</div>
<div class="container">
  <h2>Liste Beträge</h2>
  <table>
    <tr>
      <td>Benutzername</td>
      <td>Beschreibung</td>
      <td><a>löschen</a></td>
    </tr>
  </table>
</div>
<div class="container">
  <h2>Liste gemeldete Beiträge</h2>
  <table>
    <tr>
      <td>Benutzername</td>
      <td>Beschreibung</td>
      <td><a>löschen</a></td>
    </tr>
  </table>
</div>
<form method="post">

</form>
```



```
</body>
</html>
```

11.2.3 Login.jsp

```
<%@page import="App.Login"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%

//Anmeldeinformationen in Parameter speichern
String benutzername = request.getParameter("benutzername");
String password = request.getParameter("password");

//Rückmeldung für das Login
String message = "Anmelden";

//Anmeldeinformationen überprüfen
if(benutzername != null && password != null && benutzername!= "" && password != ""){
    Login login = new Login(benutzername, password);
    //Wenn Login erfolgreich, dann Session setzen und weiterleiten
    if(login.getLogged()){
        HttpSession s = request.getSession(true);
        session.setAttribute("id", login.getId());
        session.setAttribute("rolle", login.getRolle());
        response.sendRedirect("beitrag.jsp");
    }
}else{
    message="Benutzername und Passwort eingeben";
}

%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
```

```

<link rel="stylesheet" type="text/css" href="style/style.css">
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Login</title>
</head>
<body>
  <div class="container-Login">
    <div class="title"><h1>Login</h1></div>
    <div class="content">
      <form method="post">
        <input type="text" name="benutzername" maxlength="30"><br>
        <input type="password" name="passwort"><br>
        <input type="submit" value="Login">
        <p><%=message %></p>
      </form>
    </div>
  </div>
</body>
</html>

```

11.2.4 Logout.jsp

```

<%@page import="App.Login"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%
//Session beenden und alle Variablen löschen
request.getSession().invalidate();
request.getSession().setAttribute("id", null);
request.getSession().setAttribute("rolle", null);

if(session!=null){
    session = null;
}

//Zurück zum Login
response.sendRedirect("login.jsp");

```

```
%>
```

11.2.5 Meinprofil.jsp

```
<%@page import="java.util.Enumeration"%>
<%@page import="App.Benutzer"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.sql.PreparedStatement"%>
<%@page import="java.sql.Connection"%>
<%@page import="Database.Datenbank"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>

<%
    //Überprüfen ob eingeloggt
    if(session.getAttribute("id").toString().isEmpty()){
        response.sendRedirect("login.jsp");
    }

    //Datenbankverbindung
    Datenbank db = new Datenbank();
    Connection con = db.getConnect();

    //Variablen für Datenbankabfragen
    PreparedStatement ps;
    ResultSet rs;
    Benutzer profil = new Benutzer();

    //Benutzer-ID gesetzt
    Integer benutzer = Integer.parseInt(session.getAttribute("id").toString());

    //Lade den angemeldeten Benutzer nach der Session ID
    ps = con.prepareStatement("CALL loadBenutzer(?)");
    ps.setString(1, benutzer.toString());
    rs = ps.executeQuery();

    //Setze die Informationen zum Benutzer
    if(rs.next()){
```

```
        profil.setBenutzername(rs.getString("benutzername"));
        profil.setVorname(rs.getString("vorname"));
        profil.setNachname(rs.getString("nachname"));
        profil.setEmail(rs.getString("email"));
    }

    //String benutzername = "";

    //Ermitteln aller Formular-Elemente
    Enumeration<String> en = request.getParameterNames();

    while(en.hasMoreElements()){
        String parameter = en.nextElement();
        //Wenn Profil aktualisieren abgesendet, dann in DB aktualisieren
        if(parameter.equals("aendern")){
            System.out.println("ändern");
            profil.datenAendern(benutzer, request.getParameter("benutzername"), request.getParameter("vorname"),
request.getParameter("nachname"), request.getParameter("email"));
        }
    }

%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
    <link rel="stylesheet" type="text/css" href="style/style.css">
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
    <title>Login</title>
</head>
<body>
    <div class="title"><h1>Mein Profil</h1></div>
    <div class="menu">
        <p class="logged">Benutzer-ID: <%=session.getAttribute("id") %></p>
        <p><a href="neuerbeitrag.jsp">Neuer Beitrag</a><p>
        <p><a href="meinprofil.jsp">Mein Profil</a><p>
        <p><a href="beitrag.jsp">Beiträge</a><p>
        <p><a href="logout.jsp">Logout</a><p>
```

```

</div>
<div class="content">
  <form method="post">
    <table>

      <tr>
        <td><span>Benutzername:</span></td>
        <td><input name="benutzername" type="text" maxlength="20" placeholder="Benutzername"
value="<%=profil.getBenutzername() %>"></td>
      </tr>
      <tr>
        <td><span>Vorname:</span></td>
        <td><input name="vorname" type="text" maxlength="30" placeholder="Vorname"
value="<%=profil.getVorname() %>"></td>
      </tr>
      <tr>
        <td><span>Nachname:</span></td>
        <td><input name="nachname" type="text" maxlength="30" placeholder="Nachname"
value="<%=profil.getNachname() %>"></td>
      </tr>
      <tr>
        <td><span>Email:</span></td>
        <td><input name="email" type="email" maxlength="50" placeholder="Email"
value="<%=profil.getEmail() %>"></td>
      </tr>
      <tr>
        <td><span>Profilbild:</span></td>
        <td><img src=""><br><input type="file" value="Profilbild ändern"></td>
      </tr>
    </table>
    <input name="aendern" type="submit" value="Speichern">
  </form>
</div>

```

```
</body>
</html>
```

11.2.6 Neuerbeitrag.jsp

```
<%@page import="Database.Datenbank"%>
<%@page import="App.Beitrag"%>
<%@page import="java.util.Calendar"%>
<%@page import="java.util.GregorianCalendar"%>
<%@page import="java.sql.Date"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%
    //Überprüfen ob eingeloggt
    if(session.getAttribute("id").toString().isEmpty()){
        response.sendRedirect("login.jsp");
    }

    Datenbank db = new Datenbank();

    //Variablen für den neuen Beitrag
    String titel = request.getParameter("titel");
    String beschreibung = request.getParameter("beschreibung");

    //Aktuelles Datum ermitteln
    Calendar cal = new GregorianCalendar();
    Date dt = new Date(cal.getTime().getTime());

    //Benutzer-ID gesetzt
    int benutzer = Integer.parseInt(session.getAttribute("id").toString());

    //Neues Objekt für neuen Beitrag erstellen
    Beitrag b = new Beitrag(db.getConnect());
```

```

//Felder überprüfen und Informationen in Datenbank schreiben (Beitrag posten)
if(titel!=null && beschreibung!=null){
    if(titel!="" && beschreibung!=""){
        b.neu(titel, beschreibung, dt, benutzer);
        response.sendRedirect("beitrag.jsp");
    }
}
%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
    <link rel="stylesheet" type="text/css" href="style/style.css">
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
    <title>Login</title>
</head>
<body>
    <div class="title"><h1>Neuer Beitrag</h1></div>
    <div class="content">
        <form method="post">
            <div class="container">
                <input name="titel" type="text" placeholder="Titel"><br>
                <textarea name="beschreibung"></textarea><br>
                <input type="Submit" value="Erstellen">
            </div>
        </form>
    </div>
</body>
</html>

```

11.2.7 Registrieren.jsp

```

<%@page import="App.Benutzer"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%
    //Variablen für die Benutzereingaben
    String benutzername = request.getParameter("benutzername");
    String vorname = request.getParameter("vorname");

```

```
String nachname = request.getParameter("nachname");
String email = request.getParameter("email");
String passwort = request.getParameter("passwort");
String passwort_r = request.getParameter("passwort_r");
String profilbildurl="";

//Neues Benutzer-Objekt erstellen
Benutzer b = new Benutzer();
//Eigabedaten überprüfen und dann in DB eintragen
if(benutzername!=null && vorname!=null && nachname!=null && email!=null && passwort!=null && passwort_r!=null){
    if(benutzername!="" && vorname!="" && nachname!="" && email!="" && passwort!="" && passwort_r!=""){
        b.registrieren(benutzername, passwort, email, vorname, nachname, profilbildurl);
        response.sendRedirect("login.jsp");//Weiterleitung zum Login
    }
}

%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
    <link rel="stylesheet" type="text/css" href="style/style.css">
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
    <title>Login</title>
</head>
<body>
    <div class="title"><h1>Registrieren</h1></div>
    <div class="content">
        <form method="post">
            <input type="text" name="benutzername" maxlength="20" placeholder="Benutzername"><br>
            <input type="text" name="vorname" maxlength="30" placeholder="Vorname"><br>
            <input type="text" name="nachname" maxlength="30" placeholder="Nachname"><br>
            <input type="text" name="email" maxlength="50" placeholder="Email"><br>
            <input type="password" name="passwort" maxlength="30" placeholder="Passwort"><br>
            <input type="password" name="passwort_r" maxlength="30" placeholder="Passwort wiederholen"><br>
            <input type="file"><br>
            <input type="submit" value="Senden">
        </form>
    </div>
```



```
</body>  
</html>
```

11.2.8 Style.css

```
body {  
    font-family: arial;  
  
}  
  
h1{  
    font-size: 60px;  
}  
  
.container{  
    border: solid;  
    display: flex;  
    flex-direction: column;  
    justify-content: center;  
    width: 500px;  
    margin-top: 20px;  
}  
  
.container-Login{  
    display: flex;  
    flex-direction: column;  
    justify-content: center;  
    margin-top: 20px;  
}  
  
.title{  
    display: flex;  
    justify-content: center;  
    border-bottom: solid;  
}  
  
.content{  
    /*border: solid;*/
```

```
        display: flex;
        justify-content: center;
    }

    .comment{
        border-top: solid;
    }

    .comment input{
        width: 300px;
        height: 30px;
        font-size: 20px;
    }

    input{
        width: 500px;
        height: 50px;
        margin-top: 10px;
        font-size: 30px;
    }

    /*.container{
        border: solid;
        display: flex;
        justify-content: center;
    }*/

    .infohead{
        border-bottom: solid;
    }

    .content-beitrag{
        text-align: center;
    }

    td span{
```

```
        font-size: 30px;
    }

    textarea{
        height: 200px;
    }

    .menu p{
        float: left;
        margin-left: 30px;
    }

    .menu{
        width: 100%;
        height: 60px;
        border-bottom: solid;
    }
```

11.3 Datenbank

```
-- phpMyAdmin SQL Dump
-- version 4.5.1
-- http://www.phpmyadmin.net
--
-- Host: 127.0.0.1
-- Erstellungszeit: 09. Nov 2018 um 16:34
-- Server-Version: 10.1.16-MariaDB
-- PHP-Version: 7.0.9
```

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
```

```
SET time_zone = "+00:00";
```

```
/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;  
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;  
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;  
/*!40101 SET NAMES utf8mb4 */;
```

```
--  
-- Datenbank: `probeipa_sozialesnetzwerk`  
--
```

```
DELIMITER $$
```

```
--  
-- Prozeduren  
--
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `aendereBenutzerdaten` (IN `$id` INT, IN `$bn` VARCHAR(20), IN `$vorname` VARCHAR(30), IN  
`$nachname` VARCHAR(30), IN `$email` VARCHAR(50)) UPDATE benutzer SET benutzername=$bn, vorname=$vorname, nachname=$nachname,  
email=$email WHERE id_benutzer = $id$$
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `ensperreBenutzer` (IN `$id` INT) UPDATE benutzer SET gesperrt = 0 WHERE id_benutzer = $id$$
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `entferneGefaeht` (IN `$benutzer` INT, IN `$beitrag` INT) DELETE FROM gefaeht WHERE  
fs_benutzer = $benutzer AND fs_beitrag = $beitrag$$
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `erstelleBeitrag` (IN `$titel` VARCHAR(30), IN `$beschreibung` VARCHAR(300), IN `$datum` DATETIME, IN `$benutzer` INT) INSERT INTO beitrag(titel, beschreibung, datum, fs_benutzer) VALUES ($titel, $beschreibung, $datum, $benutzer)$$
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `loadAlleBenutzer` () SELECT u.id_benutzer, u.benutzername, u.email FROM benutzer u$$
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `loadBeitraege` () SELECT b.id_beitrag, b.titel, b.beschreibung, b.datum, b.gemeldet, u.benutzername, u.id_benutzer FROM beitrag b INNER JOIN benutzer u ON b.fs_benutzer = u.id_benutzer WHERE b.gemeldet = 0 ORDER BY b.datum DESC$$
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `loadBenutzer` (IN `$benutzername` VARCHAR(20)) SELECT b.id_benutzer, b.benutzername, b.vorname, b.nachname, b.email, b.passwort, r.bezeichnung, b.gesperrt, b.fs_benutzerrolle FROM benutzer b INNER JOIN benutzerrolle r ON b.fs_benutzerrolle = r.id_benutzerrolle WHERE b.benutzername = $benutzername OR b.email = $benutzername OR b.id_benutzer = $benutzername$$
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `loadGemeldeteBeitraege` () SELECT b.id_beitrag, b.titel, b.beschreibung, b.datum, b.gemeldet, u.benutzername, u.id_benutzer FROM beitrag b INNER JOIN benutzer u ON b.fs_benutzer = u.id_benutzer WHERE b.gemeldet = 1$$
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `loadKommentare` (IN `$beitrag` INT) SELECT k.fs_beitrag, b.benutzername, k.text, k.datum FROM kommentar k INNER JOIN benutzer b ON k.fs_benutzer = b.id_benutzer WHERE k.gemeldet=0 AND k.fs_beitrag = $beitrag$$
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `loescheBeitrag` (IN `$id` INT) DELETE FROM beitrag WHERE id_beitrag = $id$$
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `meldeBeitrag` (IN `$id` INT) UPDATE beitrag SET gemeldet = 1 WHERE id_beitrag = $id$$
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `register` (IN `$benutzername` VARCHAR(20), IN `$vorname` VARCHAR(30), IN `$nachname` VARCHAR(30), IN `$email` VARCHAR(50), IN `$passwort` VARCHAR(255)) INSERT INTO benutzer(benutzername, vorname, nachname, email, passwort) values($benutzername, $vorname, $nachname, $email, $passwort)$$
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `registerPasswort` (IN `$id` INT, IN `$pw` VARCHAR(255)) UPDATE benutzer SET passwort = $pw WHERE id_benutzer = $id$$
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `schreibeKommentar` (IN `$benutzer` INT, IN `$beitrag` INT, IN `$text` VARCHAR(100), IN `$datum` DATETIME) INSERT INTO kommentar(fs_beitrag, fs_benutzer, text, datum) VALUES ($beitrag, $benutzer, $text, $datum)$$
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `setGefaeellt` (IN `$benutzer` INT, IN `$beitrag` INT) INSERT INTO gefaeellt(fs_benutzer, fs_beitrag) VALUES($benutzer, $beitrag)$$
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `sperreBenutzer` (IN `$id` INT) UPDATE benutzer SET gesperrt = 1 WHERE id_benutzer = $id$$
```

```
DELIMITER ;
```

```
-- -----
```

```
--
```

```
-- Tabellenstruktur für Tabelle `beitrag`
```

```
--
```

```
CREATE TABLE `beitrag` (  
  `id_beitrag` int(11) NOT NULL,
```

```
`titel` varchar(30) NOT NULL,  
`beschreibung` varchar(300) DEFAULT NULL,  
`datum` datetime NOT NULL,  
`fs_benutzer` int(11) NOT NULL,  
`gemeldet` int(11) NOT NULL DEFAULT '0'  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
  
--  
-- Daten für Tabelle `beitrag`  
--  
  
INSERT INTO `beitrag` (`id_beitrag`, `titel`, `beschreibung`, `datum`, `fs_benutzer`, `gemeldet`) VALUES  
(11, 'test', 'test', '2018-11-09 00:00:00', 11, 0),  
(12, 'Test', 'Test', '2018-11-09 00:00:00', 11, 0);  
  
-----  
  
--  
-- Tabellenstruktur für Tabelle `benutzer`  
--  
  
CREATE TABLE `benutzer` (  
  `id_benutzer` int(11) NOT NULL,
```

```
`benutzername` varchar(20) NOT NULL,  
`vorname` varchar(30) NOT NULL,  
`nachname` varchar(30) NOT NULL,  
`email` varchar(50) NOT NULL,  
`passwort` varchar(255) NOT NULL,  
`fs_benutzerrolle` int(11) NOT NULL DEFAULT '1',  
`gesperrt` int(11) NOT NULL DEFAULT '0'  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

--

-- Daten für Tabelle `benutzer`

--

```
INSERT INTO `benutzer` (`id_benutzer`, `benutzername`, `vorname`, `nachname`, `email`, `passwort`, `fs_benutzerrolle`, `gesperrt`) VALUES  
(1, 'felix', 'he', 'asd', 'asd@fsfdsdf', 'ba88c155ba898fc8b5099893036ef205', 3, 0),  
(9, 'dennis', 'Dennis', 'Weibel', 'ssd@test.com', 'e62020afa72eb54a15725473e3a8475b', 1, 0),  
(10, 'webgoat', 'Dennis', 'Schäppi', 'dennisschaeppi@outlook.com', '623ec00ca1d92ed6e2bbaa00ee2ec140', 1, 0),  
(11, 'tester', 'Dennis', 'Schäppi', 'dennis.schaeppi@siemens.com', '08a68eec37af94301db96679e95673ca', 1, 0),  
(12, 'täst', 'test123', 'test', 'test@xn--tst-qla.com', '8b634156edde77e407764d5166e34d20', 1, 0);
```

-- -----

--


```
-- Tabellenstruktur für Tabelle `benutzerrolle`
```

```
--
```

```
CREATE TABLE `benutzerrolle` (  
  `id_benutzerrolle` int(11) NOT NULL,  
  `bezeichnung` varchar(20) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
--
```

```
-- Daten für Tabelle `benutzerrolle`
```

```
--
```

```
INSERT INTO `benutzerrolle` (`id_benutzerrolle`, `bezeichnung`) VALUES  
(1, 'Konsument'),  
(2, 'Troubleshooter'),  
(3, 'Admin');
```

```
-- -----
```

```
--
```

```
-- Tabellenstruktur für Tabelle `berechtigungen`
```

```
--
```

```
CREATE TABLE `berechtigungen` (  
  `id_berechtigung` int(11) NOT NULL,  
  `bezeichnung` varchar(20) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
  
--  
-- Daten für Tabelle `berechtigungen`  
--  
  
INSERT INTO `berechtigungen` (`id_berechtigung`, `bezeichnung`) VALUES  
(1, 'Beitrag löschen'),  
(2, 'Beitrag melden'),  
(3, 'Beitrag erstellen'),  
(4, 'Beitrag bearbeiten'),  
(5, 'Benutzer sperren');  
  
-----  
  
--  
-- Tabellenstruktur für Tabelle `gefaellt`  
--  
  
CREATE TABLE `gefaellt` (  

```

```
`id_gefaellt` int(11) NOT NULL,  
`fs_benutzer` int(11) NOT NULL,  
`fs_beitrag` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

-- -----

--

-- Tabellenstruktur für Tabelle `kommentar`

--

```
CREATE TABLE `kommentar` (  
  `id_kommentar` int(11) NOT NULL,  
  `fs_beitrag` int(11) NOT NULL,  
  `fs_benutzer` int(11) NOT NULL,  
  `text` varchar(100) NOT NULL,  
  `datum` datetime NOT NULL,  
  `gemeldet` int(11) NOT NULL DEFAULT '0'  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

-- -----

--

```
-- Tabellenstruktur für Tabelle `zuteilung_berechtigung`
```

```
--
```

```
CREATE TABLE `zuteilung_berechtigung` (  
  `id_zuteilung` int(11) NOT NULL,  
  `fs_benutzerrolle` int(11) NOT NULL,  
  `fs_berechtigung` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
--
```

```
-- Indizes der exportierten Tabellen
```

```
--
```

```
--
```

```
-- Indizes für die Tabelle `beitrag`
```

```
--
```

```
ALTER TABLE `beitrag`  
  ADD PRIMARY KEY (`id_beitrag`),  
  ADD KEY `fs_benutzer` (`fs_benutzer`);
```

```
--
```

```
-- Indizes für die Tabelle `benutzer`
```

```
--
```

```
ALTER TABLE `benutzer`  
  ADD PRIMARY KEY (`id_benutzer`),  
  ADD UNIQUE KEY `email` (`email`),  
  ADD UNIQUE KEY `benutzername` (`benutzername`),  
  ADD KEY `fs_benutzerrolle` (`fs_benutzerrolle`);
```

```
--
```

```
-- Indizes für die Tabelle `benutzerrolle`
```

```
--
```

```
ALTER TABLE `benutzerrolle`  
  ADD PRIMARY KEY (`id_benutzerrolle`);
```

```
--
```

```
-- Indizes für die Tabelle `berechtigungen`
```

```
--
```

```
ALTER TABLE `berechtigungen`  
  ADD PRIMARY KEY (`id_berechtigung`);
```

```
--
```

```
-- Indizes für die Tabelle `gefaellt`
```

```
--
```

```
ALTER TABLE `gefaellt`  
  ADD PRIMARY KEY (`id_gefaellt`),
```

```
ADD KEY `fs_benutzer` (`fs_benutzer`),
ADD KEY `fs_beitrag` (`fs_beitrag`);

--

-- Indizes für die Tabelle `kommentar`
--

ALTER TABLE `kommentar`
  ADD PRIMARY KEY (`id_kommentar`),
  ADD KEY `fs_beitrag` (`fs_beitrag`),
  ADD KEY `fs_benutzer` (`fs_benutzer`);

--

-- Indizes für die Tabelle `zuteilung_berechtigung`
--

ALTER TABLE `zuteilung_berechtigung`
  ADD PRIMARY KEY (`id_zuteilung`),
  ADD KEY `fs_benutzerrolle` (`fs_benutzerrolle`),
  ADD KEY `fs_berechtigung` (`fs_berechtigung`);

--

-- AUTO_INCREMENT für exportierte Tabellen
--
```

```
--  
-- AUTO_INCREMENT für Tabelle `beitrag`  
--  
ALTER TABLE `beitrag`  
  MODIFY `id_beitrag` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=13;  
--  
-- AUTO_INCREMENT für Tabelle `benutzer`  
--  
ALTER TABLE `benutzer`  
  MODIFY `id_benutzer` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=13;  
--  
-- AUTO_INCREMENT für Tabelle `benutzerrolle`  
--  
ALTER TABLE `benutzerrolle`  
  MODIFY `id_benutzerrolle` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=4;  
--  
-- AUTO_INCREMENT für Tabelle `berechtigungen`  
--  
ALTER TABLE `berechtigungen`  
  MODIFY `id_berechtigung` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=6;  
--  
-- AUTO_INCREMENT für Tabelle `gefaellt`  
--
```

```
ALTER TABLE `gefaellt`  
  MODIFY `id_gefaellt` int(11) NOT NULL AUTO_INCREMENT;  
--  
-- AUTO_INCREMENT für Tabelle `kommentar`  
--  
ALTER TABLE `kommentar`  
  MODIFY `id_kommentar` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=11;  
--  
-- AUTO_INCREMENT für Tabelle `zuteilung_berechtigung`  
--  
ALTER TABLE `zuteilung_berechtigung`  
  MODIFY `id_zuteilung` int(11) NOT NULL AUTO_INCREMENT;  
--  
-- Constraints der exportierten Tabellen  
--  
--  
-- Constraints der Tabelle `beitrag`  
--  
ALTER TABLE `beitrag`  
  ADD CONSTRAINT `beitrag_ibfk_1` FOREIGN KEY (`fs_benutzer`) REFERENCES `benutzer` (`id_benutzer`);  
--
```



```
-- Constraints der Tabelle `benutzer`
```

```
--
```

```
ALTER TABLE `benutzer`
```

```
  ADD CONSTRAINT `benutzer_ibfk_1` FOREIGN KEY (`fs_benutzerrolle`) REFERENCES `benutzerrolle` (`id_benutzerrolle`);
```

```
--
```

```
-- Constraints der Tabelle `gefaellt`
```

```
--
```

```
ALTER TABLE `gefaellt`
```

```
  ADD CONSTRAINT `gefaellt_ibfk_1` FOREIGN KEY (`fs_benutzer`) REFERENCES `benutzer` (`id_benutzer`),
```

```
  ADD CONSTRAINT `gefaellt_ibfk_2` FOREIGN KEY (`fs_beitrag`) REFERENCES `beitrag` (`id_beitrag`);
```

```
--
```

```
-- Constraints der Tabelle `kommentar`
```

```
--
```

```
ALTER TABLE `kommentar`
```

```
  ADD CONSTRAINT `kommentar_ibfk_1` FOREIGN KEY (`fs_beitrag`) REFERENCES `beitrag` (`id_beitrag`),
```

```
  ADD CONSTRAINT `kommentar_ibfk_2` FOREIGN KEY (`fs_benutzer`) REFERENCES `benutzer` (`id_benutzer`);
```

```
--
```

```
-- Constraints der Tabelle `zuteilung_berechtigung`
```

```
--
```

```
ALTER TABLE `zuteilung_berechtigung`
```

```
ADD CONSTRAINT `zuteilung_berechtigung_ibfk_1` FOREIGN KEY (`fs_benutzerrolle`) REFERENCES `benutzerrolle` (`id_benutzerrolle`),  
ADD CONSTRAINT `zuteilung_berechtigung_ibfk_2` FOREIGN KEY (`fs_berechtigung`) REFERENCES `berechtigungen` (`id_berechtigung`);
```

```
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;  
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;  
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */
```

