CAS SD: Projekt EQualS

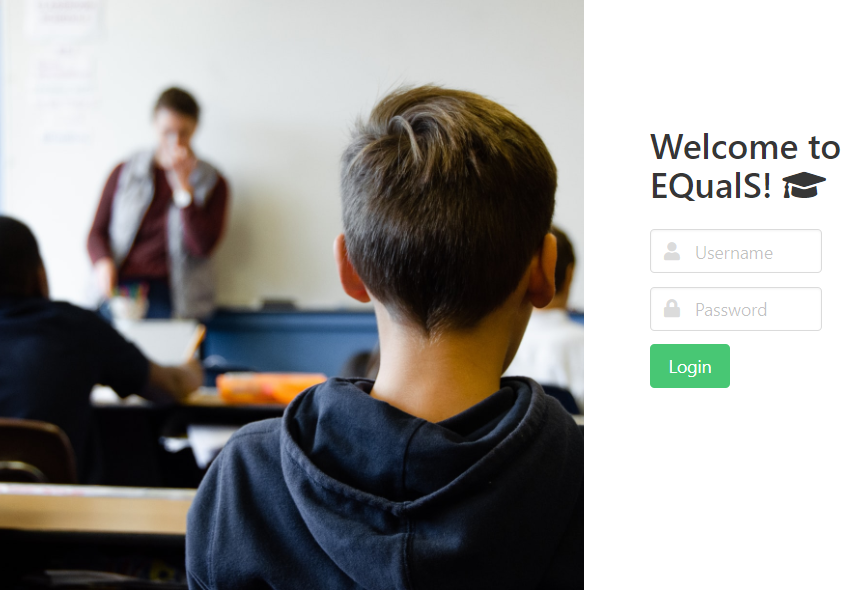
Datum: 9. März 2021

Autoren: Raphael Gerber, Christophe Leupi, Sabina Löffel, Igor Stojanovic

# Projekt EQualS

# Gruppe 1

# Raphael Gerber, Christophe Leupi, Igor Stojanovic, Sabina Löffel



[1. Use Cases 2](#_Toc66179794)

[1.1. Use Case Diagramm 2](#_Toc66179795)

[2. Datenmodell 3](#_Toc66179796)

[2.1. ER-Modell 3](#_Toc66179797)

[2.2. Datenbank-Schema 4](#_Toc66179798)

[2.3. Datenbank-Schema: Konsolidierung 5](#_Toc66179799)

[3. Graphische Benutzeroberfläche 6](#_Toc66179800)

[3.1. Ergonomie-Überlegungen 6](#_Toc66179801)

[3.2. GUI-Prototyp 6](#_Toc66179802)

[4. Gesamtdesign 7](#_Toc66179803)

[4.1. Design-Überlegungen 7](#_Toc66179804)

[4.2. Architektur 7](#_Toc66179805)

[4.3. Rest-Schnittstelle 7](#_Toc66179806)

[5. Implementation 8](#_Toc66179807)

[5.1. Zugriffskontrolle 8](#_Toc66179808)

[5.2. Notenberechnung 8](#_Toc66179809)

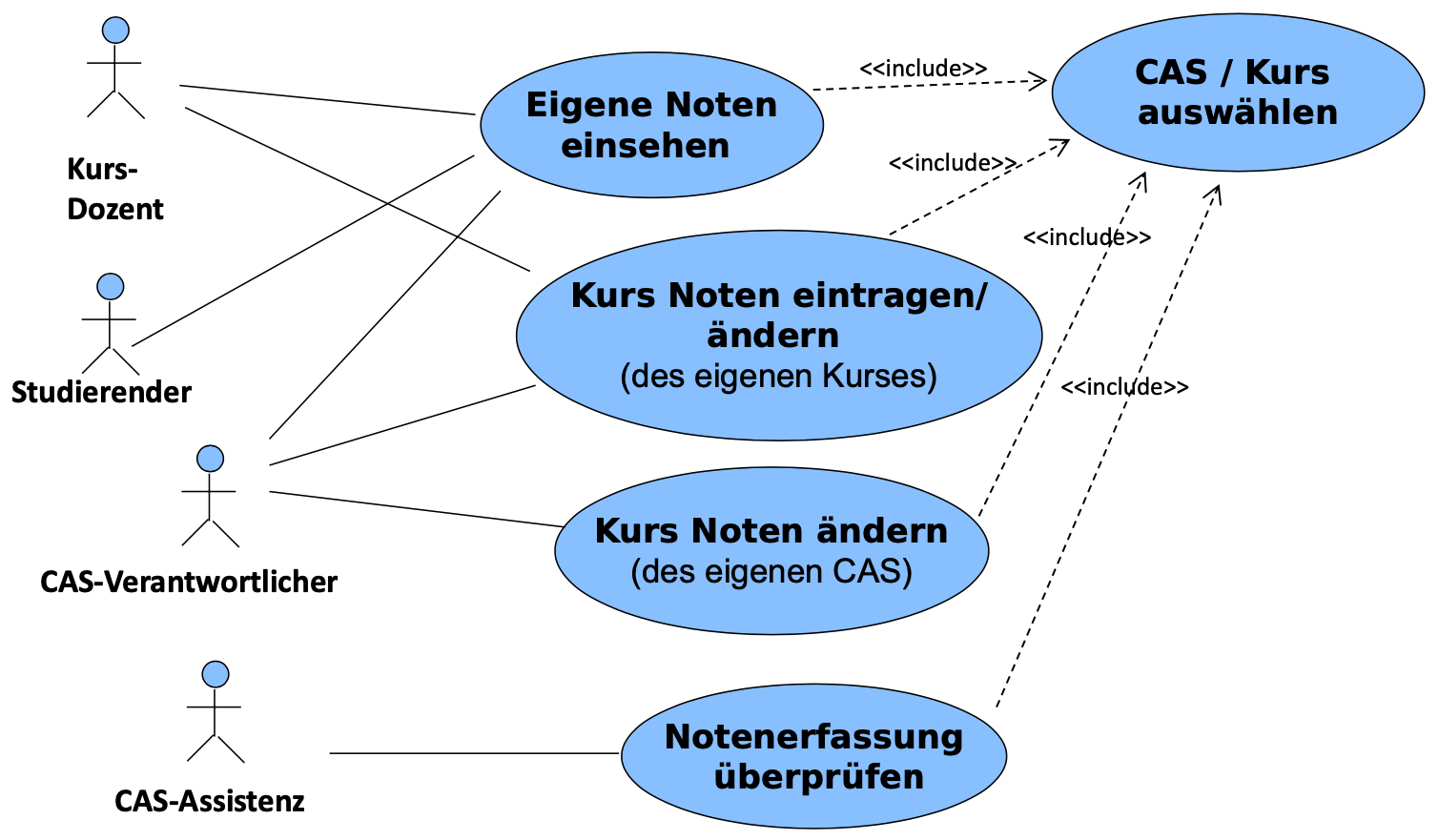
[6. Erreichte Ziele und gemachte Erfahrungen 9](#_Toc66179810)

[7. Glossar 10](#_Toc66179811)

# Use Cases

## Use Case Diagramm

Die Projektgruppe erarbeitete die Projektarbeit auf der Grundlage der Aufgabenstellung:

**

**Kurs-Dozenten (Teacher)**

Kursdozenten müssen sich in der Applikation einloggen, einen Kurs auswählen und sind dann berechtigt Noten in den Kursen, die sie unterrichten, zu erfassen, einzusehen und zu modifizieren. Auf anderen Kursen haben sie keinen Zugriff.

**Studierende**

Studierende müssen sich in der Applikation einloggen, ein CAS selektieren und die eigenen Kursnoten einsehen.

**CAS-Verantwortliche**

CAS-Verantwortliche müssen sich in der Applikation einloggen, ein CAS, welches von ihnen verantwortet wird, selektieren und alle Kursnoten aller Studierenden einsehen und ändern. Die CAS-Verantwortlichen können nur Noten der Kurse erfassen, welche sie selbst unterrichten (analog Kurs-Dozenten). Pro CAS gibt es nur eine verantwortliche Person.

**CAS-Assistenz**

Die CAS-Assistenz kann in allen CAS, welche von ihr assistiert werden, die Noten einsehen und die Dozenten auf fehlende Noten aufmerksam machen. Sie verfügen nur über Lese- und über keine Schreibrechte. Pro CAS gibt es nur eine Assistenz.

**Generell:**

Es gilt zu beachten, dass eine Person mehrere Rollen haben kann in unterschiedlichen CAS. So ist es möglich, dass ein Dozent einen Kurs unterrichtet, in einem anderen CAS als Student tätig sein. Auch ist es so möglich, dass eine CAS-Assistenz in einem anderen CAS als Studierende/r eingeschrieben ist.

# Datenmodell

## ER-Modell

Unser ER-Modell fasst die Rollen Student / Dozierender / CAS-Verantwortlicher / CAS-Administration in eine Rolle, namentlich Person, zusammen. Die Möglichkeiten der Rollen sind im Kapitel 1 «Use Cases» zusammengefasst. Dieses Modell bietet die Flexibilität, dass keine Mehrfacheinträge von gleichen Personen in verschiedenen Rollen in der Datenbank vorhanden sind, wenn z.B. ein CAS-Verantwortlicher auch gleichzeitig ein Dozent oder ein Dozent gleichzeitig ein Student ist.

Im ER-Modell ist dadurch eine zirkuläre Abhängigkeit entstanden, welche auf den ersten Blick problematisch wirkt, jedoch bei einem Projekt dieser Grösse durchaus erfolgreich umgesetzt werden kann. Daraus ergibt sich gleichzeitig auch eine Schwäche im Modell, welche es ermöglicht, einen Studierenden als CAS Verantwortliche/n einzutragen. Die Projektgruppe ist sich dieser Schwäche bewusst, ist jedoch überzeugt, dass es sich lohnt ein einfaches Modell für die Umsetzung bereitzustellen (KISS – Keep it Simple and Short).

Ein CAS-Verantwortliche/r kann mehrere CAS leiten, wobei ein CAS nur eine/n Verantwortliche/n haben kann (analog CAS-Assistenz). Ein CAS besteht aus mehreren Kursen und ein Kurs kann genau zu einem CAS gehören. Dies ist diskutierbar und könnte vorausschauend anders aufgebaut werden, die Projektgruppe einigte sich jedoch im Rahmen des Projektes dies so beizubehalten. Studierende können sich in mehrere CAS einschreiben, wobei auch mehrere Studierende in einem CAS-Kurs eingeschrieben sein können. Die CAS-Note ist ein kalkulierter Eintrag, der die Gewichtung der Kurse (quantifier) berücksichtigt und erst berechnet wird, wenn in allen zu einem CAS zugehörigen Kursen mit Noten ausgestattet wurden.

**

## Datenbank-Schema



Das Datenbank-Schema basiert auf dem ER-Modell, wurde jedoch um weitere Attribute bei den Entitäten und Relationen ergänzt. Die Projektgruppe verzichtete aufgrund Komplexitäts-minderungen bewusst auf die Aufnahme von boolean-Spalten in der Personen-Entität (z.B. isStudent, isTeacher, isCASResponsible, isAssistant). Die Constraints im Modell verunmöglichen es, z.B. Teacher in der Tabelle Course\_Teacher zu erfassen, ohne dass es die entsprechende ID in der Personentabelle gibt. Hier folgen noch einige Testqueries für das erarbeitete Modell:

**Beispiel 1:**

SELECT p.lastName, cs.grade FROM Person p  
Inner join Course\_Student cs on cs.studentId = p.id  
Inner join CAS\_Student cass on cass.StudentId = p.id  
Inner join CAS cas on cass.CASId = cas.id  
WHERE cas.id = 1 AND p.id = 1;

*Nachname und Noten der Besuchten Kurse einer Person mit der CAS ID 1 und Personen ID 1*

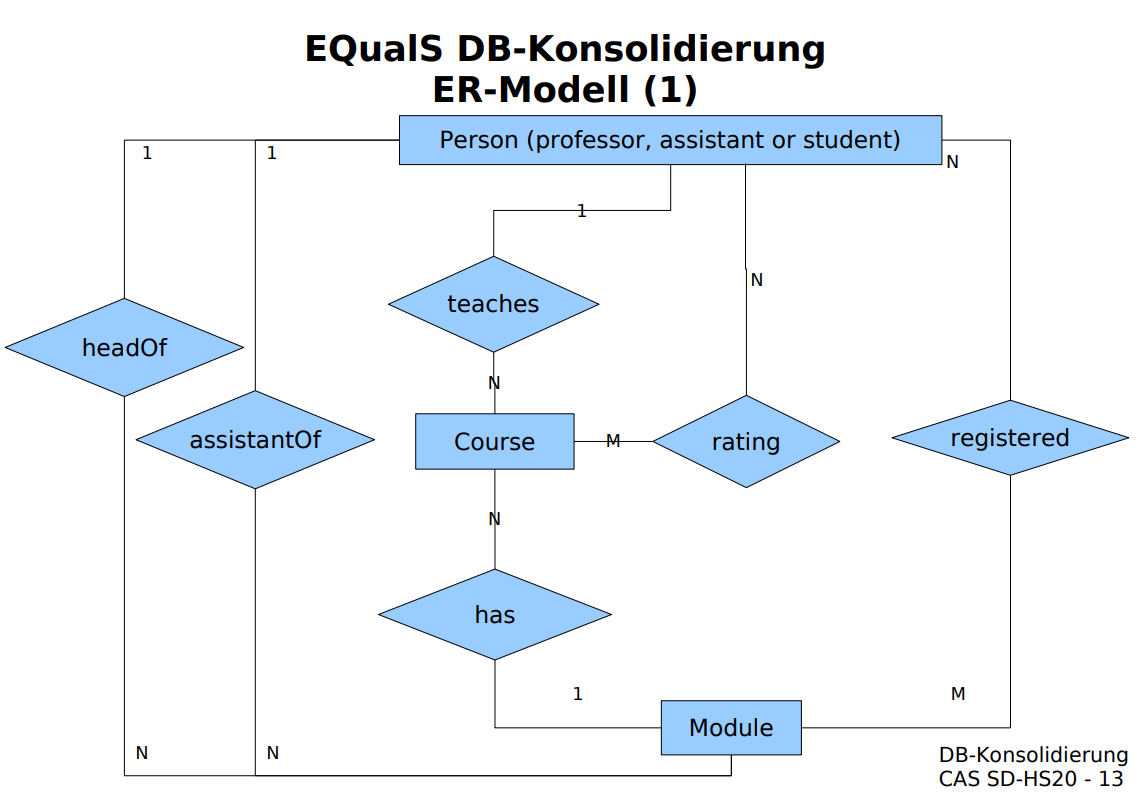
**Beispiel 2:**

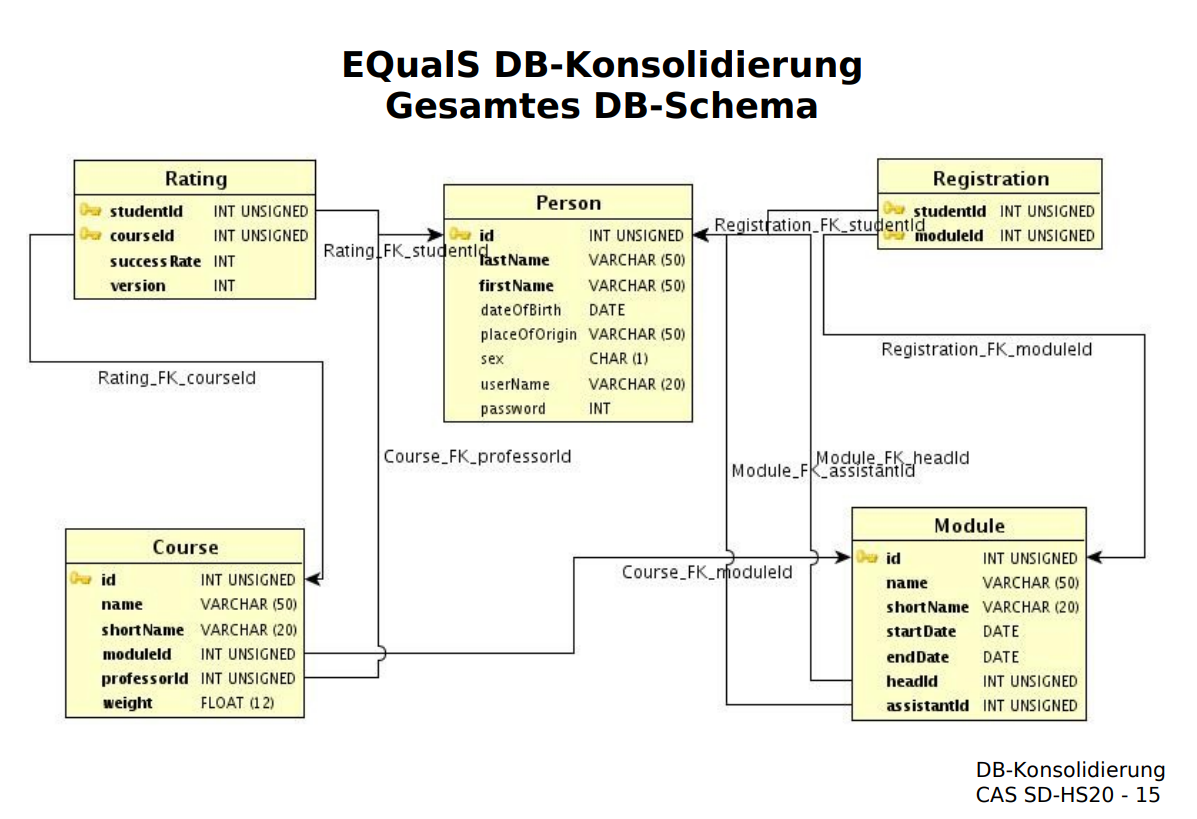
SELECT p.lastName FROM Person p  
Inner join Course\_Teacher teacher on p.id = teacher.teacherId  
Inner join Course c on teacher.courseId = c.id  
Inner join CAS cas on c.CASId = cas.id  
WHERE cas.id = 22;

*Alle Personen-Nachnamen, welche im CAS mit der ID 22 unterrichten (Teacher).*

## Datenbank-Schema: Konsolidierung

Nach Abgabe des eigenen Datenbank-Schemas wurde für die Umsetzung des Projekts ein Datenbank-Schema vorgegeben.





# Graphische Benutzeroberfläche

## Ergonomie-Überlegungen

*// Begründungen für die gewählten Masken, Navigation, Bedienung, Benutzergruppen, …*

Die Login Maske sieht für alle verschiedenen Rollen identisch aus. Der Benutzer kann sich mit dem Usernamen und Passwort einloggen. Nach erfolgtem Login sind die Modules in einer Übersicht als Kacheln ersichtlich. Dabei werden jeweils nur die Modules angezeigt, worauf der Benutzer mit der entsprechenden Berechtigung Zugriff hat.

**Save Button auf der Module Overview**

Der Save Button wird initial ausgegraut angezeigt (nach erfolgtem Login und Navigation auf die Übersicht). Wenn Werte aus der Liste verändert werden, wird der Button aktiviert und kann selektiert werden. Die Werte werden erst gespeichert, nach dem Abschicken vom Save Button. Bei der Eingabe der Werte gibt es Formatprüfungen auf die zulässigen Werte, die dem Benutzer als Orientierung dienen. Wenn unzulässige Werte eingegeben werden, kann der Request nicht abgeschickt werden und es wird ein Fehler geworfen.

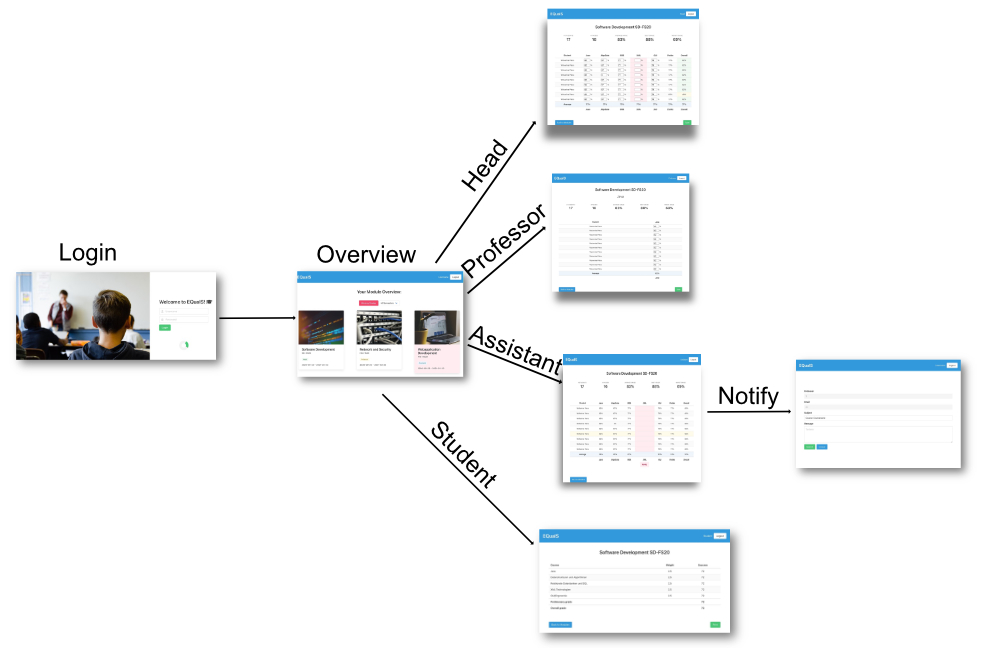
Die CAS-Assistenz sieht nach dem Login anhand der rot hinterlegten Modules, wo noch Noten fehlen.

## GUI-Prototyp

|  |  |
| --- | --- |
| Login Mock |  |
| Module Overview Mock |  |
| Full Rating Overview Mock  (Head view) |  |
| Rating Overview Mock  (Professor view) |  |
| Full Rating Overview Mock  (Assistant view) |  |
| Notification Moc (Assistant View) |  |
| Assestment Mock (Student View) |  |

## App Walk-Through

Als zentrale Anlaufstelle in der App dient die Overview-Komponente. Hier können alle User, unabhängig von der Rolle zwischen den einzelnen Modulen navigieren. Auf einem Blick stehen dem User alle Module zur Verfügung, wo eine Rolle wahrgenommen wird. Allen User-Rollen, ausser dem Student, steht eine Filterung zur Verfügung, da eine grosse Anzahl Modulen bei Studenten nicht erwartet wird. Für den CAS-Verantwortlichen, Professor und Assistent wird dieselbe Courses-Komponente aufgerufen, wobei sich die Ansicht je nach Rolle verändert wird (Anzahl Kurse, Inputfelder, etc.). Dem Assistenten wird zusätzlich ein Button angeboten, wo er direkt bei fehlenden Noten auf den Professor eines Kurses eine Nachricht (Email wird aufgerufen bei Submit) senden kann. Beim Student wird die Assessment-Komponente aufgerufen, welche eine unterschiedliche Darstellung aufzeigt, als bei den anderen Rollen. Das untenstehende Bild visualisiert den Page-Flow.



# Gesamtdesign

## Design-Überlegungen

*Begründung für das gewählte Design: Vorteile/Nachteile*

## Architektur

*Z.B. Grob-Klassendiagramm mit Erklärungen, …*

## Rest-Schnittstelle

*Dokumentation, Klassendiagramm mit Erklärungen, ….*

# Implementation

*Besonderes und Erwähnenswertes aus der Implementationsphase wie zum Beispiel:*

## Zugriffskontrolle

## Notenberechnung

Die Notenberechnung wird aktuell im Frontend durchgeführt.

# Erreichte Ziele und gemachte Erfahrungen

# Glossar

Wir haben im EQualS folgende Begriffsdefinitionen angewendet:

|  |  |
| --- | --- |
| **Module** | CAS |
| **Course** | Kurs |
| **Grade** | Note / |
| **Raiting** | Quote, Erfolgsquote / Einzelnote pro Kurs |
| **Preliminary** | Zwischenerfolg |
| **Overall** | Gesamterfolgsquote/note |