

# Dokumentation

## Student-Management-System

**Datum:** 6. Januar 2024

**Verfasser:** Pascal Fritschi

## Inhaltsverzeichnis

Projektbeschreibung.....	3
Anforderungskatalog.....	4
Klassendiagramm.....	5
Klassen und ihre Beziehungen	
Storyboard.....	6
Screen-Mockups.....	7-8
REST-Schnittstellen.....	9-10
Testplan.....	11-12
Testziele	
Testumgebung	
Testprotokoll	
Testresultate	
Installationsanleitung.....	13
Schritt-für-Schritt-Installation	
Hilfestellungen.....	14
Referenzen	

## 1.0 Projektbeschreibung

Das Ziel ist die Entwicklung eines Studenten-Management-Systems, das die Verwaltung von Studenten und Klassen zulässt. Das System ermöglicht das mühelose Hinzufügen und Entfernen von Studenten mit detaillierten Informationen wie Name, Geburtstag und Kontaktdaten. Es besteht auch die Möglichkeit, Daten einzelner Studenten zu verändern und durch eine simple Suchfunktion können Benutzer Studenten schnell anhand von Kriterien wie den Namen finden.

Die Klassenverwaltung bietet eine unkomplizierte Erstellung von Klassen. Benutzer können Studenten problemlos zu Klassen hinzufügen oder entfernen, was eine effektive Klassenorganisation unterstützt. Die Anzeige aller Schüler in einer Klasse ermöglicht einen klaren Überblick für eine effiziente Klassenverwaltung.

## 2.0 Anforderungskatalog

### **1. Studentenverwaltung:**

#### **1.1 Hinzufügen von Studenten:**

- Das System ermöglicht es Benutzern, neue Studenten hinzuzufügen.
- Benutzer müssen relevante Informationen wie Name, Geburtstag, Kontaktdaten usw. für die Registrierung eines neuen Studenten angeben.

#### **1.2 Entfernen von Studenten:**

- Benutzer haben die Möglichkeit, Studenten aus dem System zu entfernen.
- Die Entfernungsaktion sollte sicherstellen, dass alle zugehörigen Daten gelöscht werden.

#### **1.3 Datenänderungen:**

- Benutzer können die Daten einzelner Studenten bearbeiten.
- Das Bearbeiten umfasst Aktualisierungen oder Korrekturen von Informationen, um die Genauigkeit zu gewährleisten.

#### **1.4 Suchfunktion:**

- Das System stellt eine Suchfunktion bereit.
- Benutzer können nach bestimmten Studenten anhand von Kriterien wie Name oder ID suchen.

### **2. Klassenverwaltung:**

#### **2.1 Erstellung von Klassen:**

- Benutzer können Klassen erstellen.

#### **2.2 Hinzufügen von Studenten zu Klassen:**

- Benutzer können Studenten zu bestimmten Klassen hinzufügen.
- Die Zuordnung sollte einfach und effizient sein.

#### **2.3 Entfernen von Studenten aus Klassen:**

- Die Möglichkeit, Studenten aus Klassen zu entfernen, sollte vorhanden sein.
- Diese Funktion bietet Flexibilität in der Klassenorganisation.

#### **2.4 Anzeige aller Schüler in einer Klasse:**

- Das System ermöglicht die Anzeige einer Übersicht aller Schüler in einer bestimmten Klasse.
- Dies gewährleistet eine effektive Klassenverwaltung und -überwachung.

### **3. Allgemeine Anforderungen:**

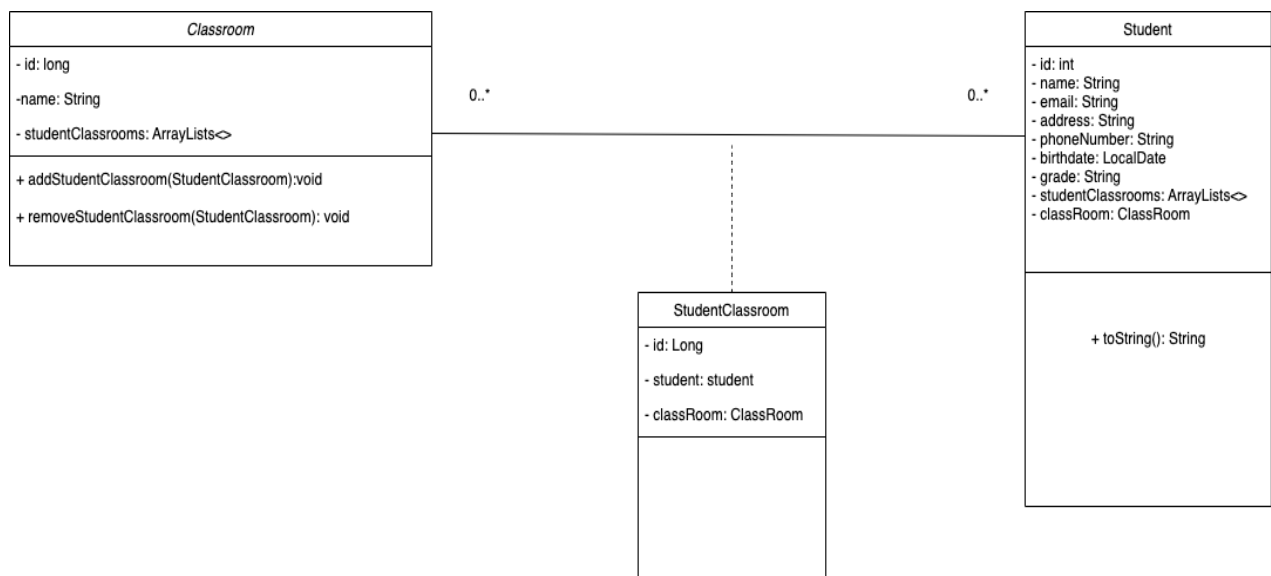
#### **3.1 Benutzeroberfläche:**

- Die Benutzeroberfläche sollte benutzerfreundlich und intuitiv gestaltet sein.
- Sie muss klare Navigationswege und verständliche Funktionen bieten.
- Die Benutzeroberfläche sollte responsiv sein und auf verschiedenen Bildschirmgrößen gut funktionieren.

## 3.0 Klassendiagramm

### 3.1 Modell-Klassen und deren Beziehung zueinander

Das folgende UML-Diagramm beschreibt die Beziehung zwischen der «Student»-Klasse und der «Classroom»-Klasse. Jeder Student kann entweder keiner Klasse, einer Klasse oder mehreren Klassen (0..\*) zugeordnet werden, dasselbe gilt für die «Classroom»-Klasse.



## 4.0 Storyboards

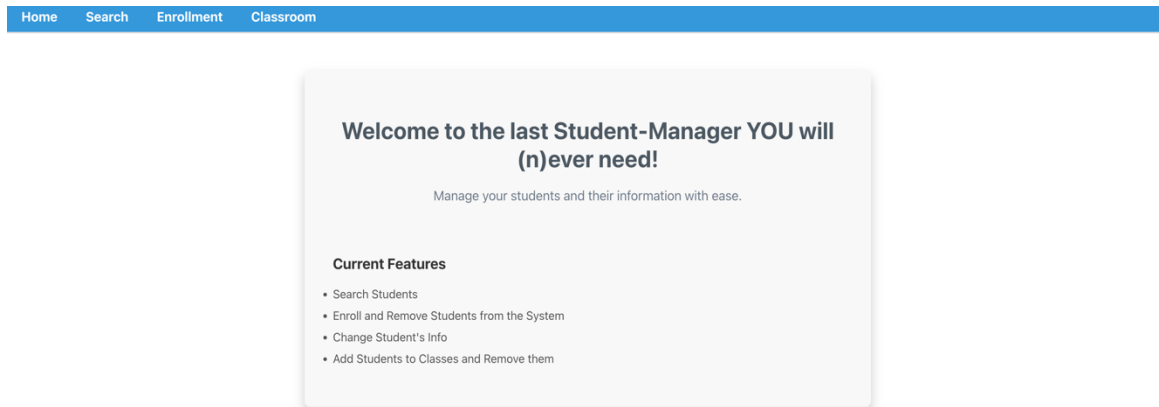
Die folgenden Storyboards beschreiben, die verschiedenen Funktionen der Web-Applikation:



## 5.0 Screen-Mockups (Student-Management-System)

Die folgenden Mockups zeigen die vorhandenen Screens in der Student-Management Applikation:

### Home-Screen



The Home-Screen mockup features a blue navigation bar at the top with the links: Home, Search, Enrollment, and Classroom. The main content area is a light gray box with a white background. It contains a welcome message, a sub-header, a list of current features, and a search bar.

**Welcome to the last Student-Manager YOU will (n)ever need!**

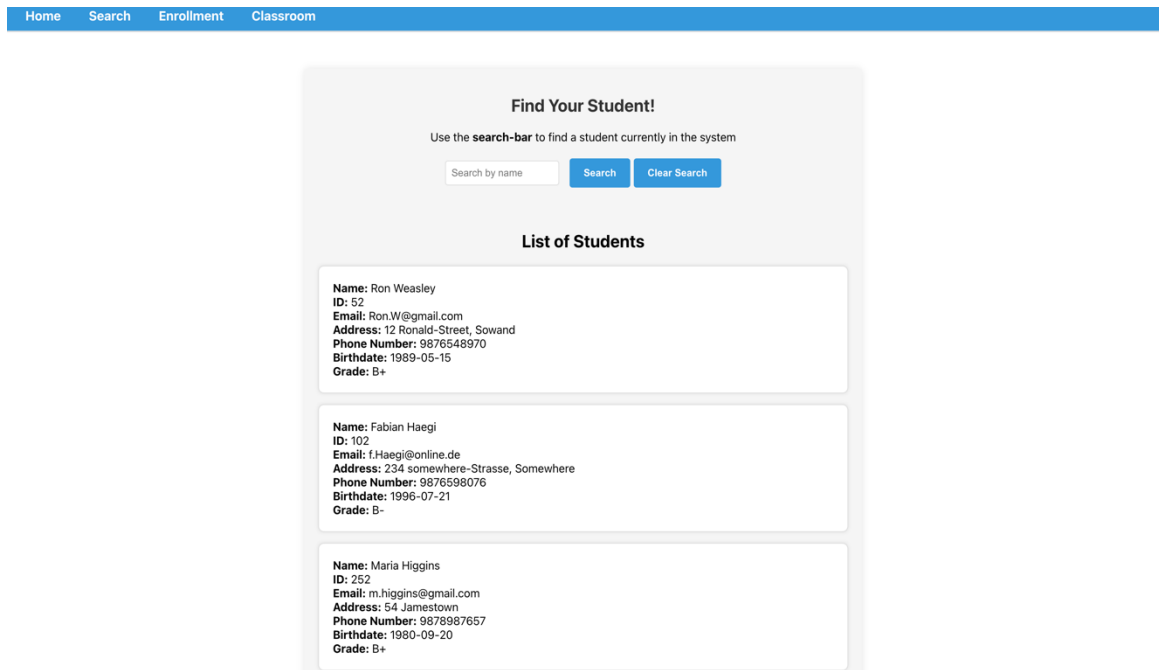
Manage your students and their information with ease.

**Current Features**

- Search Students
- Enroll and Remove Students from the System
- Change Student's Info
- Add Students to Classes and Remove them

Search by name

### Search-Screen



The Search-Screen mockup features a blue navigation bar at the top with the links: Home, Search, Enrollment, and Classroom. The main content area is a light gray box with a white background. It contains a search bar, a list of students, and a search button.

**Find Your Student!**

Use the **search-bar** to find a student currently in the system

Search by name

**List of Students**

**Name:** Ron Weasley  
**ID:** 52  
**Email:** Ron.W@gmail.com  
**Address:** 12 Ronald-Street, Sowand  
**Phone Number:** 9876548970  
**Birthdate:** 1989-05-15  
**Grade:** B+

**Name:** Fabian Haegi  
**ID:** 102  
**Email:** f.haegi@online.de  
**Address:** 234 somewhere-Strasse, Somewhere  
**Phone Number:** 9876598076  
**Birthdate:** 1996-07-21  
**Grade:** B-

**Name:** Maria Higgins  
**ID:** 252  
**Email:** m.higgins@gmail.com  
**Address:** 54 Jamestown  
**Phone Number:** 9876987657  
**Birthdate:** 1980-09-20  
**Grade:** B+

## Student-List-Screen

[Home](#) [Search](#) [Enrollment](#) [Classroom](#)

### Enrollment

Use the Form on the left to **add** a student to the System.

#### Student Form

Add Student

#### Ron Weasley

Email: Ron.W@gmail.com

Address: 12 Ronald-Street, Sowand

Phone Number: 9876548970

Birthdate: 1989-05-15

Grade: B+

DeleteUpdate

#### Fabian Haegi

Email: f.Haegi@online.de

Address: 234 somewhere-Strasse, Somewhere

Phone Number: 9876598076

Birthdate: 1996-07-21

Grade: B-

DeleteUpdate

#### Maria Higgins

Email: m.higgins@gmail.com

Address: 54 Jamestown

Phone Number: 9878987657

Birthdate: 1980-09-20

Grade: B+

DeleteUpdate

#### Severus Snape

Email: s.snape@gmail.com

Address: 23 Schlangenstrasse, HighCastle

Phone Number: 9988779889

Birthdate: 1969-05-12

Grade: A+

DeleteUpdate

## Classroom-Enrollment-Screen

[Home](#) [Search](#) [Enrollment](#) [Classroom](#)

### Classroom Enrollment

Enter a name to create a class

Create Classroom

#### Classrooms

HistoryDelete

ChemistryDelete

Select a classroom from the dropdown menu to view enrolled students

Adding students: Choose a **classroom** and a **student** to assign the student to the class

ChemistrySelect Student

Add Student

Students in Selected Classroom

Fabian HaegiDelete



## 6.0 REST-Schnittstellen

Die folgende Auflistung zeigt die vorhandenen REST-Schnittstellen im Backend der Applikation

### 1. ClassroomController:

Nr.	HTTP-Methode	URI	Beschreibung	Anforderungskörper	Rückgabewert
1	GET	/api/classroom/getClassrooms	Ruft alle Klassen aus der Datenbank ab.	-	List<ClassRoom>
2	POST	/api/classroom/createClassroom	Erstellt eine neue Klasse.	ClassRoom	ClassRoom
3	DELETE	/api/classroom/{classroom_id}	Löscht eine Klasse anhand der ID.	-	Void
4	GET	/api/classroom/{classroom_id}/students	Ruft alle Schüler in einer Klasse anhand der ID ab.	-	List<Student>

### 2. StudentClassroomController:

Nr.	HTTP-Methode	URI	Beschreibung	Anforderungskörper	Rückgabewert
1	GET	/api/studentclassroom/getAll/{classroom_id}	Ruft alle Student-Klassenraum-Beziehungen in einer Klasse ab.	-	List<StudentClassroom>
2	POST	/api/studentclassroom/add/{student_id}/{classroom_id}	Fügt einen Schüler zu einer Klasse hinzu.	StudentClassroom	StudentClassroom
3	DELETE	/api/studentclassroom/remove/{student_id}/{classroom_id}	Entfernt einen Schüler aus einer Klasse.	-	Void

### 3. StudentController

Nr.	HTTP-Methode	URI	Beschreibung	Anforderungskörper	Rückgabewert
1	GET	/api/student	Ruft alle Schüler aus der Datenbank ab.	-	List<Student>
2	POST	/api/student	Fügt einen neuen Schüler zur Datenbank hinzu.	Student	Student
3	GET	/api/student/search?studentId={student_id}	Sucht einen Schüler anhand der ID.	-	Student
4	PUT	/api/student/update	Aktualisiert einen vorhandenen Schüler in der Datenbank.	Student	Student
5	DELETE	/api/student/delete?studentId={student_id}	Löscht einen Schüler anhand der ID.	-	String

## 7.0 Testplan

### 7.1 Testziele

- Überprüfung der Funktionalität zur Hinzufügung von Schülern zu Klassen.
- Validierung der korrekten Entfernung von Schülern aus Klassen.
- Gewährleistung der ordnungsgemäßen Anzeige aller Schüler in einer Klasse.
- Bestätigung der erfolgreichen Erstellung neuer Klassen.
- Überprüfung der Funktionalität der Schülersuche.

### 7.2. Testumgebung

- Entwicklungsumgebung: Eclipse (mit Java 8.0 oder höher)
- Testumgebung: lokale Entwicklungsumgebung

### 7.3 Testprotokoll

Testfall	Schritte	Erwartetes Ergebnis	Bestanden (Pass/Not)
Erfolgreiches Hinzufügen eines Schülers zu einer Klasse	1. Einen neuen Schüler erstellen. 2. Eine vorhandene Klasse auswählen. 3. Schüler auswählen 4. Den Schüler zur Klasse hinzufügen.	Der Schüler wird erfolgreich zur ausgewählten Klasse hinzugefügt.	<input type="checkbox"/> Pass
Testfall	Schritte	Erwartetes Ergebnis	Bestanden (Pass/Not)
Validierung der korrekten Entfernung von Schülern aus Klassen	1. Einen vorhandenen Schüler in einer Klasse auswählen. 2. Den Schüler aus der Klasse entfernen.	Der Schüler ist nicht mehr mit der ausgewählten Klasse verbunden.	<input type="checkbox"/> Pass
Testfall	Schritte	Erwartetes Ergebnis	Bestanden (Pass/Not)
Gewährleistung der ordnungsgemäßen Anzeige aller Schüler in einer Klasse	1. Eine bestimmte Klasse auswählen. 2. Die Liste der Schüler in dieser Klasse abrufen.	Die Anwendung zeigt eine Liste der Schüler, die zu der ausgewählten Klasse gehören.	<input type="checkbox"/> Pass
Testfall	Schritte	Erwartetes Ergebnis	Bestanden (Pass/Not)
Bestätigung der erfolgreichen Erstellung neuer Klassen	1. Details für eine neue Klasse bereitstellen. 2. Die neue Klasse erstellen.	Die Anwendung erstellt eine neue Klasse, die in der Liste der Klassen sichtbar ist.	<input type="checkbox"/> Pass
Testfall	Schritte	Erwartetes Ergebnis	Bestanden (Pass/Not)
Überprüfung der Funktionalität der Schülersuche	1. Eine Suche mit einem bestimmten Schülernamen starten.	Die Anwendung gibt den oder die Schüler zurück, die den Suchkriterien entsprechen.	<input type="checkbox"/> Pass

7.4 Testresultate				
Tester: Pascal Fritschi		Test-Datum: 6. Januar 2024		
Testfall	Schritte	Erwartetes Ergebnis	Ergebnis	Bestanden (Pass/Not)
Erfolgreiches Hinzufügen eines Schülers zu einer Klasse	1. Einen neuen Schüler erstellen. 2. Eine vorhandene Klasse auswählen. 3. Schüler auswählen 4. Den Schüler zur Klasse hinzufügen.	Der Schüler wird erfolgreich zur ausgewählten Klasse hinzugefügt.	Schüler wird erfolgreich zur Klasse hinzugefügt	[ x] Pass
Testfall	Schritte	Erwartetes Ergebnis	Ergebnis	Bestanden (Pass/Not)
Validierung der korrekten Entfernung von Schülern aus Klassen	1. Einen vorhandenen Schüler in einer Klasse auswählen. 2. Den Schüler aus der Klasse entfernen.	Der Schüler ist nicht mehr mit der ausgewählten Klasse verbunden.	Der Student wird korrekt aus der Klasse entfernt	[ x] Pass
Testfall	Schritte	Erwartetes Ergebnis	Ergebnis	Bestanden (Pass/Not)
Gewährleistung der ordnungsgemäßen Anzeige aller Schüler in einer Klasse	1. Eine bestimmte Klasse auswählen. 2. Die Liste der Schüler in dieser Klasse abrufen.	Die Anwendung zeigt eine Liste der Schüler, die zu der ausgewählten Klasse gehören.	Liste der Schüler in einer Klasse funktioniert	[ x] Pass
Testfall	Schritte	Erwartetes Ergebnis	Ergebnis	Bestanden (Pass/Not)
Bestätigung der erfolgreichen Erstellung neuer Klassen	1. Details für eine neue Klasse bereitstellen. 2. Die neue Klasse erstellen.	Die Anwendung erstellt eine neue Klasse, die in der Liste der Klassen sichtbar ist.	Klasse wird erfolgreich erstellt und ist in der Liste sichtbar	[ x] Pass
Testfall	Schritte	Erwartetes Ergebnis	Ergebnis	Bestanden (Pass/Not)
Überprüfung der Funktionalität der Schülersuche	1. Eine Suche mit einem bestimmten Schülernamen starten.	Die Anwendung gibt den oder die Schüler zurück, die den Suchkriterien entsprechen.	Suchfunktion gibt den gesuchten Schüler aus	[ x] Pass

## 8.0 Installationsanleitung

### 8.1 Schritt-für-Schritt Anleitung

#### 1. Datenbank erstellen:

- Öffne deine MySQL-Datenbank-Verwaltungskonsole.
- Führe den folgenden SQL-Befehl aus, um die Datenbank "StudentManagement" zu erstellen: `CREATE DATABASE IF NOT EXISTS StudentManagement;`

#### 2. Projekt in Eclipse importieren:

- Öffne Eclipse und wähle "File" > "Import...".
- Wähle "Maven" > "Existing Maven Projects" und klicke auf "Next".
- Wähle das heruntergeladene Student-Management-System aus und klicke auf "Finish".

#### 3. Datenbankkonfiguration im Projekt aktualisieren:

- Öffne die Datei `src/main/resources/application.properties` im Projekt.
- Aktualisiere die Datenbankverbindungsinformationen (Benutzername und Passwort).

#### 4. Maven-Build ausführen:

- Klicke mit der rechten Maustaste auf das Projekt und wähle "Run As" > "Maven build...".
- Gib "spring-boot:run" als Goal ein und klicke auf "Run".

#### 5. React-Build ausführen

- Gehe in den Frontend-Projekt-Ordner und tippe «npm start» in die Konsole ein. (Falls «npm start» nicht funktioniert, benutze «npm install» und versuche es erneut mit «npm start»)

#### 6. Öffne die Anwendung:

- Öffne deinen Webbrowser und gehe zu <http://localhost:3000> um das Frontend zu erreichen, oder <http://localhost:8080> um die API zu erreichen.

## 9.0 Hilfestellungen

### 9.1 Referenzen

#### Internet-Recherche

- <https://react.dev/reference/react/useState> (useState)
- <https://react.dev/reference/react/useEffect> (useEffect)
- Unterricht-Unterlagen (Modul 294 & 295) auf Teams/OneNote
- <https://www.freecodecamp.org/news/javascript-map-how-to-use-the-js-map-function-array-method/> (map-function-array-method)

#### **Beispiel aus dem Projekt (JavaScript map):**

```
<h2>Students in Selected Classroom</h2>
{console.log('Classroom Students:', classroomStudents)}
{classroomStudents.length > 0 ? (
  <ul className="classroom-list">
    {classroomStudents.map((studentObj) => (
      <li key={studentObj.id} className="classroom-list-item">
        {studentObj.student.name}
        <button onClick={() => deleteStudentFromClassroom(studentObj.student.id, selectedClassRoomId)}>Delete</button>
      </li>
    ))}
  </ul>
) : (
  <p>No students in this classroom.</p>
)}
```

- <https://www.developer.com/java/java-optional-object/> (use of Optional in Java)
- <https://javabeginners.de/Grundlagen/Datentypen/Optionals.php>