Aufgabe 1: Stakeholder-Analyse

a)

1. **Studenten**
   - **Interesse**: Die Studenten sind die primären Nutzer des Systems und haben ein großes Interesse daran, dass sie fair und flexibel in Gruppen eingeteilt werden, die ihrem Zeitplan entsprechen, um Überschneidungen zu vermeiden. ✓
2. **Dozenten/Lehrpersonal**
   - **Interesse**: Dozenten wollen eine gerechte Verteilung der Studenten auf die Übungsgruppen und einen effizienten Zugang zu den Informationen über Teilnehmerzahlen und Zeitpläne ihrer Gruppen. ✓
3. **Systemadministratoren**
   - **Interesse**: Diese Stakeholder sind für die technische Wartung des Systems verantwortlich. Sie wollen sicherstellen, dass das System stabil, sicher und benutzerfreundlich läuft und dass die Registrierung und Authentifizierung reibungslos funktioniert. ✓
4. **Softwareentwickler**
   - **Interesse**: Die Entwickler arbeiten am Entwurf und der Implementierung des Systems und sind daran interessiert, dass Anforderungen und Wünsche der Endnutzer (Studenten und Dozenten) klar definiert sind, damit sie das System passend gestalten können. ✓
5. **IT-Sicherheitsbeauftragte der Universität**
   - **Interesse**: Sie sorgen dafür, dass alle personenbezogenen Daten geschützt und sicher sind und dass der Zugriff über sichere Universitätsanmeldedaten erfolgt. ✓
6. **Verwaltung/Management der Universität**
   - **Interesse**: Das Management hat ein Interesse an der Effizienzsteigerung durch das System, um die Fairness bei der Gruppenzuordnung zu erhöhen und die Zufriedenheit der Studenten zu fördern. Eine erfolgreiche Einführung könnte die Reputation der Universität verbessern. ✓

3/3P.

b) Power-Interest-Grid

Power	High Interest	Low Interest
High	Verwaltung/Management, Dozenten	IT-Sicherheitsbeauftragte
Low	Studenten, Systemadministratoren	Softwareentwickler

✓ 1/1P.

Σ 4/4P.

Aufgabe 2:

a) 6 functional requirements

1. Das System muss es den Dozenten ermöglichen, Übungsgruppen mit spezifischen Zeiten und Teilnehmergrenzen anzulegen. ✓
2. Studenten können sich über das System für alle gewünschten Kurse und deren Übungsgruppen in einem Semester anmelden. mit ihren Uni Zugangsdaten -0,5P.
3. Studenten sollen Zeiten markieren können, in denen sie für Übungen nicht verfügbar sind, damit das System diese bei der Gruppenzuteilung berücksichtigen kann. ✓
4. Das System muss die Studenten auf die Übungsgruppen aufteilen und versuchen, Überschneidungen mit anderen Kursen zu vermeiden. ✓
5. Das System soll Studenten benachrichtigen, in welche Übungsgruppe sie zugeteilt wurden. ✓
6. Das System muss eine Option zur manuellen Nachbearbeitung bieten, falls ein Student keiner Übungsgruppe zugewiesen werden konnte. ✓

b) 3 quality requirements

1. **Benutzerfreundlichkeit**
   - Das System muss eine intuitive Benutzeroberfläche haben, die es sowohl Studenten als auch Dozenten leicht ermöglicht, ihre Aufgaben (Anmeldung, Gruppenerstellung) zu erledigen.

   steht so nicht im Text. -0,5P.
2. **Sicherheit**
   - Der Zugriff auf das System muss durch die Shibboleth-Authentifizierung abgesichert sein, um den Datenschutz und die Sicherheit der Benutzerinformationen zu gewährleisten. ✓
3. **Skalierbarkeit**
   - Das System muss in der Lage sein, bis zu mehrere tausend Studenten zu verwalten, insbesondere während der Stoßzeiten zu Semesterbeginn. ✓

c) 1 constraint

- Das System muss in der Programmiersprache Java entwickelt werden, um die Beteiligung der Studenten an der Entwicklung und den Testphasen zu unterstützen. ✓

d) 1 project requirement

- Die erste Testversion des Systems muss bis zum Beginn des Wintersemesters 2025/26 bereitgestellt werden, um den geplanten Rollout im Wintersemester 2026/27 zu ermöglichen. ✓

e) 1 process requirement

- Studenten sollen als Entwickler und Tester in den Entwicklungsprozess eingebunden werden, um das System auf Benutzerfreundlichkeit und Funktionalität zu prüfen. ✓

Σ 5/6P.

Verbesserungen sollen als Requirement ausformuliert werden. dass dann testbar und präzise ist.

Aufgabe 3: requirement validation

Anforderung 1: ***"Das System muss es den Dozenten ermöglichen, Übungsgruppen mit spezifischen Zeiten und Teilnehmergrenzen anzulegen."***

- **Präzision**: Diese Anforderung ist präzise, da sie klar beschreibt, was von der Funktion erwartet wird.
- **Konsistenz**: Keine Konflikte mit anderen Anforderungen.
- **Verifizierbarkeit**: Die Funktion ist leicht testbar, indem überprüft wird, ob Dozenten Gruppen mit Zeiten und Teilnehmergrenzen erstellen können. ✓
- **Gültigkeit**: Entspricht den Erwartungen der Stakeholder.

Verbesserung: Ergänzung spezifischer Details, z. B., ob es eine maximale oder minimale Teilnehmerzahl gibt.

Anforderung 2: ***"Studenten müssen sich über das System für alle gewünschten Kurse und deren Übungsgruppen in einem Semester anmelden können."***

- **Präzision**: Eindeutig, aber könnte präzisiert werden (z. B. „alle gewünschten Kurse“ auf „zugelassene Kurse für das Semester“ spezifizieren). + Loginmethode
- **Konsistenz**: Konsistent mit anderen Anforderungen.
- **Verifizierbarkeit**: Testbar, indem geprüft wird, ob Anmeldungen für mehrere Kurse und Gruppen möglich sind.
- **Gültigkeit**: Entspricht den Interessen der Studenten und Dozenten.

Tf

Verbesserung: Angabe einer maximalen Anzahl an Kursen oder Gruppen, falls es solche Beschränkungen gibt.

Anforderung 3: ***"Studenten sollen Zeiten markieren können, in denen sie für Übungen nicht verfügbar sind, damit das System diese bei der Gruppenzuteilung berücksichtigen kann."***

- **Präzision**: Präzise formuliert.
- **Konsistenz**: Konsistent mit dem Ziel der fairen Verteilung.
- **Verifizierbarkeit**: Verifizierbar, indem man sicherstellt, dass das System solche Zeiten akzeptiert und die Verteilung entsprechend anpasst. ✓
- **Gültigkeit**: Entspricht den Anforderungen der Studenten.

Verbesserung: Hinzufügen von Details zur Art der Zeiterfassung (z. B. Uhrzeiten, Wochentage).

Anforderung 4: ***"Das System muss die Studenten auf die Übungsgruppen aufteilen und versuchen, Überschneidungen mit anderen Kursen zu vermeiden."***

- **Präzision**: Recht präzise, könnte jedoch genauer formuliert werden, wie Überschneidungen vermieden werden.
- **Konsistenz**: Konsistent mit den Zielen des Systems. ✓
- **Verifizierbarkeit**: Testbar durch Simulation verschiedener Studentenkonfigurationen und deren Zuteilung.
- **Gültigkeit**: Wichtig für die Zielgruppe der Studenten.

Verbesserung: Klarere Spezifikation der Regeln zur Priorisierung bei Überschneidungen.

Anforderung 5: ***"Das System soll Studenten benachrichtigen, in welche Übungsgruppe sie zugeteilt wurden."***

- **Präzision**: Präzise.
- **Konsistenz**: Passt zur Logik des Systems.
- **Verifizierbarkeit**: Testbar durch Prüfung, ob Benachrichtigungen korrekt verschickt werden.
- **Gültigkeit**: Entspricht den Erwartungen der Nutzer.

Verbesserung: Angabe, wie die Benachrichtigung erfolgt (z. B. per E-Mail).

Anforderung 6: ***"Das System muss eine Option zur manuellen Nachbearbeitung bieten, falls ein Student keiner Übungsgruppe zugewiesen werden konnte."***

- **Präzision**: Präzise formuliert.
- **Konsistenz**: Konsistent mit den anderen Anforderungen und Zielen.
- **Verifizierbarkeit**: Testbar durch Prüfung der manuellen Zuweisungsoptionen.
- **Gültigkeit**: Wichtig für reibungslose Systemnutzung.

Verbesserung: Definition, wer berechtigt ist, die manuelle Zuweisung vorzunehmen.

ΣS/SP.

USE CASE ATTRIBUTE	DETAILS
USE CASE NAME	Übungsgruppen-Anmeldung und Zuweisung
PRIMARY ACTOR	Student
SECONDARY ACTORS	1. Systemadministrator (bei Konflikten) 2. Dozenten (zur manuellen Lösung von Zuweisungen)
PRECONDITIONS	1. Der Student ist im EGD-System authentifiziert (über Universitäts-Login). 2. Der Student ist für mindestens einen Kurs angemeldet, der Übungsgruppen benötigt.
POSTCONDITIONS	Der Student ist einer oder mehreren Übungsgruppen erfolgreich zugewiesen, oder er erhält eine Mitteilung, dass er die Übungszeit manuell abstimmen muss.
MAIN SUCCESS SCENARIO	1. Der Student loggt sich in das EGD-System ein. 2. Das System zeigt alle Kurse und Übungsgruppenoptionen an, für die der Student angemeldet ist. 3. Der Student gibt seine Verfügbarkeiten an, indem er Zeiten auswählt, in denen er nicht verfügbar ist. 4. Das System berechnet basierend auf den Verfügbarkeiten des Studenten eine Zuweisung zu den Übungsgruppen. 5. Das System weist den Studenten einer passenden Übungsgruppe zu und vermeidet dabei Überschneidungen, wenn möglich. 6. Der Student wird über seine Gruppenzuteilung informiert, z. B. per E-Mail oder innerhalb des Systems.
EXTENSIONS (ALTERNATIVE SCENARIOS)	5a. Wenn keine Übungsgruppe ohne Zeitkonflikte gefunden wird: 1. Das System benachrichtigt den Studenten über den Konflikt und schlägt vor, sich an den zuständigen Dozenten zu wenden. 2. Der Student kontaktiert den Dozenten, um eine Lösung zu finden, z. B. durch eine manuelle Zuweisung zu einer anderen Übungszeit. 5b. Wenn der Kurs nur eine begrenzte Anzahl an Übungsgruppen hat und keine Überschneidungsfreie Option verfügbar ist: 1. Das System weist den Studenten automatisch einer Gruppe in einem konfliktfreien Kurs zu und gibt eine Anweisung zur manuellen Abstimmung mit dem zweiten Kurs.
TRIGGER	Der Student öffnet das EGD-System

ES/SP