Jan Kremer; Matrikel Nummer: 7398656

Homework 1. Requirements

Exercise 1. Stakeholders

a) Identifying and Explaining Stakeholders

1. **Students**

   - **Interest**: High - Students are directly impacted by the new system because it will determine the distribution of exercise groups, potentially affecting their schedules, workload, and course conflicts.

   - **Power**: Low - While they have a high interest in a fair and functional system, they have limited influence over the system's development, except through feedback and testing.

2. **Lecturers**

   - **Interest**: High - Lecturers are involved in creating exercise groups and setting the parameters for group distribution. A well-functioning system can reduce their administrative workload.

   - **Power**: Medium - Lecturers have some influence as they determine the requirements and constraints of the exercise groups.

3. **Department of Computer Science Administration**

   - **Interest**: Medium - The department oversees the general management and coordination of courses. An efficient system can streamline administrative tasks and improve student satisfaction.

   - **Power**: High - The administration has significant influence over funding, implementation decisions, and adoption of the system.

4. **System Administrators (IT Staff)**

   - **Interest**: Medium - They will be responsible for the technical deployment, maintenance, and support of the system.

   - **Power**: Medium - Their role involves implementing and managing the system, giving them some influence over technical decisions.

5. **Chair of Software & Systems Engineering**

   - **Interest**: High - The chair is driving the initiative to develop the system, providing guidance on system requirements and quality expectations.

   - **Power**: High - As a major decision-maker, they influence the development process, including design decisions and timelines.

6. **Future University-wide Users (Other Departments)**

- **Interest**: Low (Current) - At present, other departments are not directly involved.

- **Power**: Low (Current) - They currently have no influence, but their interest and power may increase if the system is expanded university-wide.

b) Power/Interest Grid Classification
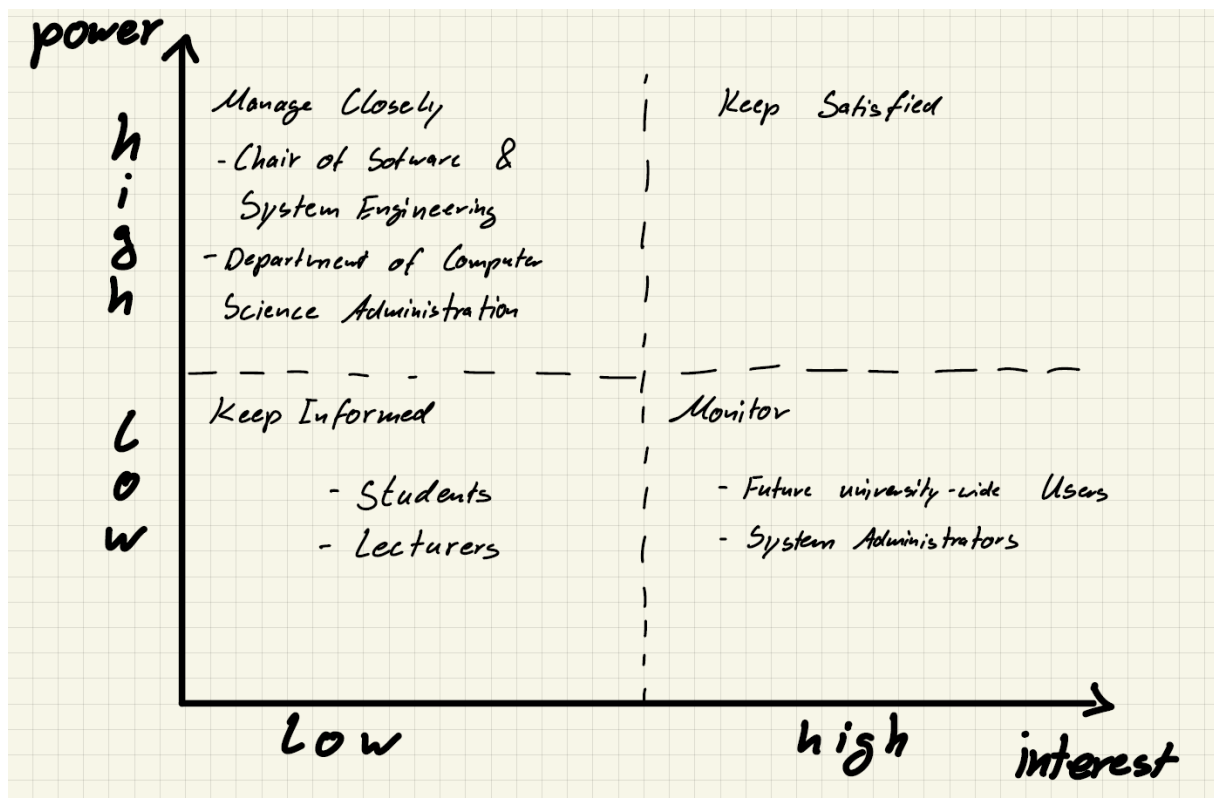
Exercise 2. Requirements

a) Functional Requirements

1. The system shall allow lecturers to create exercise groups for their courses, including setting session times and specifying the number of students per group.
2. The system shall enable students to register for multiple courses' exercise sessions in a single semester.
3. The system shall allow students to mark times when they are unavailable due to other commitments.
4. The system shall automatically distribute students across available exercise groups, considering their indicated availability and schedule conflicts.

5. The system shall notify students of their assigned exercise groups after the distribution process is complete.
6. The system shall support manual administration for cases where students could not be assigned to any group.

b) Quality Requirements and Their Respective Quality Attribute

1. **Usability**: The system should be easy to use for both students and lecturers, with a user-friendly interface that simplifies navigation and registration.
2. **Scalability**: The system should be able to handle thousands of students, especially during peak registration periods.
3. **Security**: The system should ensure that access is controlled through university credentials (Shibboleth) and protect personal data from unauthorized access.

c) Constraint

- The system must be developed in Java.

d) Project Requirement

- The system should be deployed in the winter semester of 2026/27, with first test versions ready by the beginning of the winter semester 2025/26.

e) Process Requirement

- Students should participate in the development of the system, both as developers and testers.

Exercise 3. Requirements Validation

Functional Requirements

1. **Requirement**: "The system shall allow lecturers to create exercise groups for their courses, including setting session times and specifying the number of students per group."

   - **Precision**: Partially fulfilled. It specifies the actions lecturers can take, but lacks details about how this is done (e.g., through an interface or form).

   - **Consistency**: Fulfilled. There are no conflicting requirements in the list.

   - **Verifiability**: Partially fulfilled. It is verifiable whether lecturers can create groups, but not whether the interface or process is easy or intuitive.

   - **Validity**: Fulfilled. It aligns with the system's goal to manage exercise group distribution.

- o **Improvement**: "The system shall provide an interface for lecturers to create exercise groups by filling out a form, specifying session times, location, and the maximum number of students."

2. **Requirement**: "The system shall enable students to register for multiple courses' exercise sessions in a single semester."

   - o **Precision**: Partially fulfilled. The term "register" could be more specific (e.g., add/edit/delete registrations).

   - o **Consistency**: Fulfilled. It fits well with other requirements.

   - o **Verifiability**: Fulfilled. It can be tested whether students can register for multiple courses.

   - o **Validity**: Fulfilled. This feature is necessary for the system's main purpose.

   - o **Improvement**: "The system shall allow students to register for, edit, and delete exercise session registrations for multiple courses within a semester."

3. **Requirement**: "The system shall allow students to mark times when they are unavailable due to other commitments."

   - o **Precision**: Partially fulfilled. It should clarify how unavailability is marked (e.g., calendar interface).

   - o **Consistency**: Fulfilled. This does not contradict other requirements.

   - o **Verifiability**: Fulfilled. It can be checked whether students can indicate unavailable times.

   - o **Validity**: Fulfilled. This helps ensure fair group distribution.

   - o **Improvement**: "The system shall provide a calendar interface for students to mark specific times or days when they are unavailable due to other commitments."

4. **Requirement**: "The system shall automatically distribute students across available exercise groups, considering their indicated availability and schedule conflicts."

   - o **Precision**: Partially fulfilled. It doesn't specify the criteria for distribution or prioritization rules.

   - o **Consistency**: Fulfilled. It aligns with other requirements.

   - o **Verifiability**: Partially fulfilled. It can be tested if the system distributes students, but the fairness or effectiveness of the algorithm may be subjective.

- o **Validity**: Fulfilled. It addresses the core problem the system aims to solve.

- o **Improvement**: "The system shall use an algorithm to automatically distribute students across available exercise groups, prioritizing group assignment based on their indicated availability and minimizing schedule conflicts."

5. **Requirement**: "The system shall notify students of their assigned exercise groups after the distribution process is complete."

   - o **Precision**: Partially fulfilled. It does not specify how notifications will be sent (e.g., email, in-app notification).

   - o **Consistency**: Fulfilled. It complements other requirements.

   - o **Verifiability**: Fulfilled. It can be verified if students receive notifications.

   - o **Validity**: Fulfilled. It is necessary for informing students of the results.

   - o **Improvement**: "The system shall notify students of their assigned exercise groups via email and in-app notifications after the distribution process is complete."

6. **Requirement**: "The system shall support manual administration for cases where students could not be assigned to any group."

   - o **Precision**: Partially fulfilled. It should describe what kind of manual administration is possible (e.g., reassigning students, creating new groups).

   - o **Consistency**: Fulfilled. It doesn't conflict with other requirements.

   - o **Verifiability**: Fulfilled. Manual intervention can be tested.

   - o **Validity**: Fulfilled. Manual administration is a fallback option in line with system goals.

   - o **Improvement**: "The system shall provide an interface for manual administration, allowing authorized staff to reassign students or create additional exercise groups if students could not be assigned automatically."

Quality Requirements

1. **Requirement**: "The system should be easy to use for both students and lecturers, with a user-friendly interface that simplifies navigation and registration."

   - o **Precision**: Partially fulfilled. "Easy to use" and "user-friendly" are vague terms.

- o **Consistency**: Fulfilled. There are no conflicting quality requirements.

- o **Verifiability**: Not fulfilled. There is no specific metric to measure ease of use.

- o **Validity**: Fulfilled. It aligns with the goal of making the system accessible.

- o **Improvement**: "The system should have a user-friendly interface, where common tasks (e.g., registering for a group) can be completed in no more than three steps."

2. **Requirement**: "The system should be able to handle thousands of students, especially during peak registration periods."

- o **Precision**: Partially fulfilled. It should specify expected load (e.g., number of users within a timeframe).

- o **Consistency**: Fulfilled. It does not conflict with other requirements.

- o **Verifiability**: Partially fulfilled. It can be tested for high load, but the definition of "peak" could be clearer.

- o **Validity**: Fulfilled. Scalability is critical for system performance.

- o **Improvement**: "The system should handle up to 5,000 concurrent users during peak registration periods without performance degradation."

3. **Requirement**: "The system should ensure that access is controlled through university credentials (Shibboleth) and protect personal data from unauthorized access."

- o **Precision**: Fulfilled. It specifies the method of access control.

- o **Consistency**: Fulfilled. It aligns with security requirements.

- o **Verifiability**: Fulfilled. Security features can be tested.

- o **Validity**: Fulfilled. Data protection is essential for compliance.

- o **Improvement**: No improvement needed; this requirement is precise and testable.

Constraint

- **Requirement**: "The system must be developed in Java."

- o **Precision**: Fulfilled. It clearly specifies the programming language.

- o **Consistency**: Fulfilled. There are no conflicting requirements.

- o **Verifiability**: Fulfilled. It can be checked whether the system is developed in Java.

- o **Validity**: Fulfilled. The constraint aligns with the department's goal for student participation.

- o **Improvement**: No improvement needed; this is already a clear constraint.

Project Requirement

- **Requirement**: "The system should be deployed in the winter semester of 2026/27, with first test versions ready by the beginning of the winter semester 2025/26."

  - o **Precision**: Fulfilled. It provides a clear timeline.

  - o **Consistency**: Fulfilled. The timeline is aligned with development goals.

  - o **Verifiability**: Fulfilled. The deployment dates can be tracked.

  - o **Validity**: Fulfilled. The timeline matches the project schedule.

  - o **Improvement**: No improvement needed; this requirement is already clear.

Process Requirement

- **Requirement**: "Students should participate in the development of the system, both as developers and testers."

  - o **Precision**: Partially fulfilled. It could clarify the level of involvement expected.

  - o **Consistency**: Fulfilled. It aligns with the goal of student participation.

  - o **Verifiability**: Partially fulfilled. Measuring "participation" may be subjective.

  - o **Validity**: Fulfilled. It is consistent with the educational purpose of the project.

  - o **Improvement**: "Students should participate in the development of the system, contributing at least 20% of the codebase and conducting user testing during key development phases."

  - o

Exercise 4. Use Case

Title	Register for Exercise Groups	
Actors	- **Students:** Register for exercise groups based on course requirements and personal availability. - **Lecturers:** Ensure fair distribution of students across groups and assist in manual conflict resolution.	

	- **Course Coordinators:** May help resolve scheduling conflicts and arrange special sessions.
Preconditions	- The student is enrolled in one or more courses. - The EGD system is accessible and functioning. - The student knows their course schedule and other time commitments.
Trigger	-The student opens the page to register for exercise groups.
Main succes scenario	1. **Log in to the EGD system:** The student logs in using university credentials. 2. **View available exercise groups:** The student sees exercise group options for all enrolled courses. 3. **Input availability:** The student specifies times when they are not available. 4. **Automatic group assignment:** The system tries to assign the student to groups based on availability and course requirements. 5. **Notify student of results:** The system notifies the student if assignment was successful or if there are unresolved conflicts. 6. **Manual conflict resolution (if needed):** If conflicts remain, the student contacts the lecturer or course coordinator to arrange a solution.
Alternative paths	- **A1. Unable to Assign Any Groups:** If no groups can be assigned, the student is informed and guided to contact relevant lecturers or coordinators. - **A2. Conflicting Group Times:** If the student has conflicting schedules for different courses, the system prioritizes one course and asks the student to resolve the other manually. - **A3. Successful Automatic Rescheduling:** If conflicts are detected, the system may attempt alternative scheduling based on different available slots.