Exercise 1: Stakeholders

(a) Stakeholder Identification and Analysis

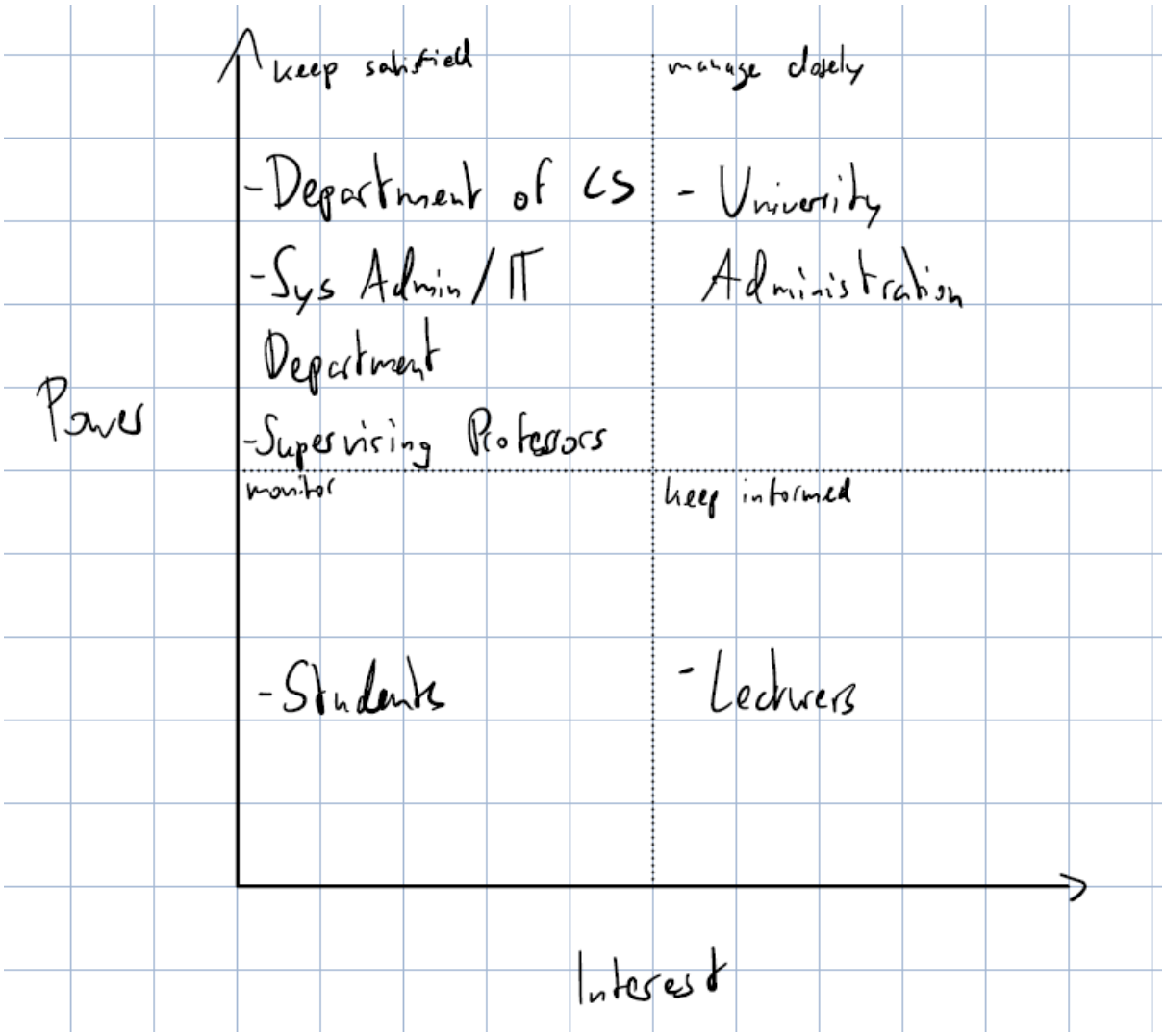
1. **Students**
    - **Role and Interest:** Students are the primary users of the EGD system. They have a strong interest in the project because the system directly impacts their schedule flexibility, group allocation, and fairness in distribution. This new system can alleviate scheduling conflicts, improve group assignment fairness, and simplify registration.
    - **Power:** Low power but high interest, as they are directly affected by the system's functionality and fairness.
2. **Lecturers**
    - **Role and Interest:** Lecturers will use the system to manage and create exercise groups, set session times, and monitor registration. Their interest lies in having a smooth, efficient system that reduces administrative workload and ensures students are appropriately allocated based on availability.
    - **Power:** Medium power, as they influence group creation and need to ensure that sessions run efficiently.
3. **System Administrators/IT Department**
    - **Role and Interest:** Responsible for maintaining the system’s functionality, data security, and system integration with university login credentials (e.g., Shibboleth). Their interest is in ensuring the system operates reliably and securely, with minimal disruptions.
    - **Power:** High power and high interest, as they control the system's infrastructure and security aspects.
4. **Department of Computer Science**
    - **Role and Interest:** Oversees the EGD project development and is responsible for budget allocation, ensuring that the project aligns with departmental goals (e.g., fairness and ease of use). They are also interested in student satisfaction and may be motivated to expand the system university-wide if successful.
    - **Power:** High power and high interest, as they are decision-makers for the project and are invested in its success.
5. **University Administration**
    - **Role and Interest:** University Administration might consider scaling the system university-wide if it proves beneficial. Although they are not directly involved in the initial rollout, they are interested in the system’s potential to streamline administrative processes.
    - **Power:** Medium power, as they could influence future funding or expansion decisions but have low interest at this stage of development.
6. **Supervising Professors and Development Team**

- **Role and Interest:** Leads the project and supervises development, with a vested interest in meeting project goals on time, within budget, and involving students as developers and testers. They ensure that the system meets technical and functional requirements.
- **Power:** High power, as they oversee the development and have control over project direction, budget allocation, and timeline adherence.

✓

3/3P.

(b) Power/Interest Grid



✓

1/1P.

Σ4/4P.

Exercise 2: Requirements

(a) Functional Requirements

1. The system must allow lecturers to create exercise groups for their courses, specifying session times and capacity.
2. The system must enable students to log in and view available sessions for each course.
3. The system must allow students to register for multiple courses within one semester.
4. The system must provide an option for students to mark times when they are unavailable.
5. The system must automatically assign students to groups based on availability and minimize scheduling conflicts.
6. The system must notify students of their assigned groups after allocation is complete.

(b) Quality Requirements

1. **Usability:** The system should be easy to use for both students and lecturers, with a user-friendly interface.
2. **Scalability:** The system should handle up to several thousand users during peak registration periods.
3. **Security:** The system should protect personal data from unauthorized access, with secure access controlled through university credentials (e.g., Shibboleth).

(c) Constraint

- The system must be developed in Java.

(d) Project Requirement

- The total development budget for the system must not exceed 70,000 euros.

(e) Process Requirement

- The system must be ready for testing at the beginning of the winter semester 2025/26.

Exercise 3: Requirements Validation

(a) Functional Requirements Validation

1. **Lecturer Group Creation**
    - **Precision:** Partially fulfilled. The requirement describes group creation but lacks detail on what specific information should be entered.
    - **Consistency:** Fulfilled, as it does not contradict other requirements.
    - **Verifiability:** Partially fulfilled; it's unclear how "specifying session times and capacity" should be validated.

- **Validity:** Fulfilled; creating groups is essential for the system.
- **Improvement Example:** "The system must allow lecturers to create exercise groups by specifying the session name, day and time, capacity, and additional notes for each course."

2. **Student Session Viewing**
    - **Precision:** Partially fulfilled; details about filtering or organizing session views are not specified.
    - **Consistency:** Fulfilled.
    - **Verifiability:** Partially fulfilled, as it's not clear how the completeness of session information display should be tested.
    - **Validity:** Fulfilled, as viewing sessions is a necessary function.
    - **Improvement Example:** "The system must enable students to log in and view a list of available sessions for each course, including session name, day, time, and capacity."
3. **Multi-Course Registration**
    - **Precision:** Fulfilled, but could be clarified further.
    - **Consistency:** Fulfilled.
    - **Verifiability:** Fulfilled; can be verified by testing the multi-course selection feature.
    - **Validity:** Fulfilled; multi-course registration is essential for a practical scheduling system.
    - **Improvement Example:** "The system must allow students to register for exercise groups across multiple courses within one semester."
4. **Marking Unavailability**
    - **Precision:** Partially fulfilled; needs clarification on the format of unavailability input (e.g., days and times).
    - **Consistency:** Fulfilled.
    - **Verifiability:** Partially fulfilled; could be improved by specifying how unavailability should be recorded and displayed.
    - **Validity:** Fulfilled; it addresses the need to consider student schedules.
    - **Improvement Example:** "The system must allow students to mark days and times they are unavailable, using a time-block interface, to minimize scheduling conflicts."
5. **Automatic Group Assignment**
    - **Precision:** Partially fulfilled; lacks clarity on how conflicts are minimized and the criteria for assignment.
    - **Consistency:** Fulfilled.
    - **Verifiability:** Partially fulfilled; testing conflict minimization may be difficult without more specifics.
    - **Validity:** Fulfilled; aligns with the system's purpose.
    - **Improvement Example:** "The system must automatically assign students to groups based on availability, prioritizing minimal conflicts and maximizing course fit."
6. **Assignment Notifications**

- **Precision:** Partially fulfilled; the form of notifications (e.g., email, in-app) is not specified.
- **Consistency:** Fulfilled.
- **Verifiability:** Partially fulfilled; could specify the notification process.
- **Validity:** Fulfilled; notifying students of their assignments is necessary.
- **Improvement Example:** "The system must notify students of their assigned groups via in-app notifications and email after the allocation process is complete."

(b) Quality Requirements Validation

1. **Usability**
    - **Precision:** Partially fulfilled; “easy to use” could be subjective.
    - **Consistency:** Fulfilled.
    - **Verifiability:** Partially fulfilled; testing usability requires specific metrics.
    - **Validity:** Fulfilled; usability is crucial.
    - **Improvement Example:** "The system should achieve a usability score of at least 80% in student and lecturer satisfaction surveys."
2. **Scalability**
    - **Precision:** Fulfilled; clearly mentions handling thousands of users.
    - **Consistency:** Fulfilled.
    - **Verifiability:** Fulfilled; can be tested through load testing.
    - **Validity:** Fulfilled; system scalability is crucial for potential university-wide use.
    - **Improvement Example:** No change needed; it’s clear and verifiable.
3. **Security**
    - **Precision:** Partially fulfilled; does not specify specific security standards.
    - **Consistency:** Fulfilled.
    - **Verifiability:** Partially fulfilled; security could be tested more effectively with specifics.
    - **Validity:** Fulfilled; security is essential due to sensitive data.
    - **Improvement Example:** "The system should protect personal data by implementing data encryption, two-factor authentication, and access control through Shibboleth."

(c) Constraint Validation

1. **Java Development**
    - **Precision:** Fulfilled; specified without ambiguity.
    - **Consistency:** Fulfilled.
    - **Verifiability:** Fulfilled; checking for Java development is straightforward.
    - **Validity:** Fulfilled; aligns with departmental requirements.
    - **Improvement Example:** No change needed.

(d) Project Requirement Validation

1. **Budget Constraint**
    - **Precision:** Fulfilled; specified clearly.
    - **Consistency:** Fulfilled.
    - **Verifiability:** Fulfilled; the budget can be monitored.
    - **Validity:** Fulfilled; aligns with resource management goals.
    - **Improvement Example:** No change needed.

(e) Process Requirement Validation

1. **Testing Schedule**
    - **Precision:** Fulfilled; specifies a clear timeframe.
    - **Consistency:** Fulfilled.
    - **Verifiability:** Fulfilled; the timeline can be monitored.
    - **Validity:** Fulfilled; aligns with project schedule goals.
    - **Improvement Example:** "The system must be ready for beta testing by the beginning of the winter semester 2025/26 to ensure timely feedback."

Exercise 4: Use Case

Use Case Name:
Student Exercise Group Selection and Assignment

Primary Actor:
Student

Stakeholders and Interests:

1. **Student** - Wants to be assigned to exercise groups that avoid conflicts with other scheduled commitments.
2. **Lecturer** - Needs to manage group capacities effectively and support students in case of scheduling conflicts.
3. **System Administrators** - Ensure the system functions efficiently, allowing fair distribution of students across exercise groups.
4. **Department of Computer Science** - Interested in a system that improves the fairness and satisfaction of group assignments among students.

Preconditions:

- The student must be enrolled in the courses they wish to register for.
- The student must have login access to the EGD system (assumed through university credentials, e.g., Shibboleth).
- Lecturers have created and published exercise groups with session times and capacities in the EGD.

Postconditions:

- The student is assigned to available, non-conflicting exercise groups if possible.
- The student is notified of any remaining conflicts and receives information for resolution (e.g., contacting lecturers).

Main Success Scenario:

1. The student logs into the EGD system.
2. The student selects their enrolled courses and views the available exercise group sessions for each course.
3. The student enters their availability (or unavailability) times, indicating when they cannot attend sessions.
4. The EGD system attempts to assign the student to exercise groups for each course, considering:
    - The student's availability.
    - The schedule of the selected exercise groups.
    - The capacity of each group.
5. **(Success Case):** If the system finds conflict-free group assignments for all courses:
    - The system assigns the student to appropriate exercise groups.
    - The system sends a confirmation email to the student with details of their group assignments.

Alternative Scenarios:

- **Alternative Path 1: Partial Assignment with Conflict Notification**
    - **4a.** If the system cannot assign the student to an exercise group for one or more courses due to conflicts:
        - The system notifies the student of the conflict(s) and indicates which course(s) require manual resolution.
        - The student may contact the relevant lecturers or course coordinators to negotiate an alternative solution (e.g., creating a new session or adjusting their availability).
- **Alternative Path 2: Conflict within Available Groups**
    - **4b.** If there are limited group options and a conflict occurs between groups in two different courses:
        - The system assigns the student to one of the conflicting groups (based on available capacity and priority).
        - The system notifies the student to contact the course coordinator of the unassigned group to negotiate an alternative.

Exceptions:

1. **Login Failure** - If the student cannot log into the EGD system, they are advised to contact IT support.
2. **Capacity Limit Exceeded** - If all groups are at maximum capacity, the student is notified to contact the course coordinator for alternative arrangements.
3. **Unavailable Groups** - If there are no groups available due to scheduling conflicts or other issues, the student is notified and directed to manual resolution.

Assumptions:

- The system has a mechanism to store and manage each student's availability input.
- Notifications to students are delivered via email and in-app messages.
- In cases where manual conflict resolution is needed, there are support processes in place (e.g., instructor contact, possible special sessions).

Σ S/SP.