Homework 01

Ohad Ragolsky 7381351, Gruppe 4

**Exercise 1. (Stakeholders) - (a),(b):**

1. **Department of Computer Science at the University of Cologne:**
    - **Power(High):** This department acts as the project sponsor, overseeing the project, providing funding, coordinating overall activities, and ensuring alignment with institutional goals.
    - **Interest in Development (High):** The department is interested in promoting the project to increase student satisfaction through a fair distribution across all exercise groups and to reduce the number of students left without placement in a group.
2. **Students (Users):**
    - Students are the primary users of the EGD system and register for exercise groups through it.
    - **Interest in Development (High):** They want a fair and efficient system that considers their availability and minimizes scheduling conflicts with other courses or personal commitments
    - **Power(Low):** Individually, students have limited influence, but collectively, their feedback is important for the system's usability,
3. **Lecturers**
    - Lecturers use the system to create and manage exercise groups for their courses. The system's effectiveness impacts their workload.
    - **Interest in Development (High):** They want a system that simplifies group formation and ensures balanced group sizes.
    - **Power(High):** Lecturers significantly influence system design, as it must meet their needs.
4. **Chair of Software & Systems Engineering**
    - This chair oversees the development of the EGD system and involves students as developers and testers.
    - **Interest in Development (High):** The chair is committed to the system’s successful development and implementation while providing students with practical experience through the project.
    - **Power(Medium-High):** The chair leads the development.
5. University IT Department (System Administration)
    - The IT department supports the integration of the system into existing university infrastructures, such as Shibboleth authentication.
    - **Interest in Development (High):** The department need to ensure the system is compatible, secure, and scalable.
    - **Power(High):** If the system is inadequately developed, insecure, or incompatible with other university systems, the IT department can forbid it use.

6. **University Administration**

- The university's higher administration, beyond the Computer Science Department, might be involved if the system is deployed university wide.
- **Interest in Development (Medium):** They are interested in the system's potential university-wide application and its impact on overall student satisfaction.
- **Power(Medium-High):** They could influence funding and policy if the system is adopted university-wide.

(**b**)

**Power/Interest Grid**

| Power / Interest | High Interest | Medium/Low Interest |
|---|---|---|
| High Power | Department of CS<br>Lecturers<br>Chair SSE<br>IT-Department | University-Administration |
| Medium/Low Power | Students (Users) | |

**Explanation of Classification:**

- **Lecturers, Department of CS, Chair of SSE, University IT Department (High Power, High Interest)** These stakeholders have significant influence over the project and a strong interest in its outcomes. They should be closely involved to ensure their needs are met and their feedback is incorporated into project development.

- **University-Administration (High Power, Low Interest):**
  **The University Administration** has the power to deploy the system university-wide. However, changes in big organizations are not always welcome.

- **Students (Users) (Low Power, High Interest):** As end-users, students have a keen interest in the system's functionality and fairness, though little influence on project decisions. They should be informed of project progress and how it benefits them.

**Exercise 2. Requirements**

a. **functional requirements:**
   1. A user shall be able to Register as lecturer, Student or Administrator
   2. Lecturers shall be able to create exercise groups for their courses
   3. Lecturers shall be able to set group details like session times and the number of students each group can handle.
   4. Students log in to view available sessions.
   5. Student shall be able to register all courses they want to attend exercises for in one semester.
   6. The system shell send Notifications that will let students know which groups they've been assigned to.

b. **3 quality requirements and their respective quality attribute**
   1. The system shall distribute students evenly among the groups.
   2. The System shall minimize the number of students that could not be assigned a group
   3. The system shall minimize the number of manual administration processes

c. **1 constraint**
   1. The system should be developed in Java.

d. **1 project requirement**
   1. The system should be deployed in winter semester 2026/27, with first test Versions ready at the beginning of winter semester 2025/26.

e. **1 process requirement.**
   1. Students should participate in the development of the system, both as developers as well as testers.

**Exercise 3. Requirements Validation**

**functional requirements:**

- A.1 User shall be able to Register as a lecturer, Student or Administrator
    - **Validity Check**: This requirement is valid as it aligns with the real need for differentiating user roles and providing appropriate access.
    - **Consistency Check**: It is consistent with other user-related requirements; ensure there are no contradictions in role-specific functionalities
    - **Completeness Check**: Consider specifying the registration details required for each user type, such as credentials and role-specific fields.
    - **Realism Check**: The registration feature is realistic and can be implemented easily within the system's scope.
    - **Verifiability Check**: The requirement is verifiable by testing each user type's registration process to ensure successful role-specific account creation
- A.2 Lecturers shall be able to create exercise groups for their courses
    - **Validity Check**: This requirement is valid as it meets the lecturers' need to manage exercise groups effectively.
    - **Consistency Check**: Ensure consistency with related requirements, such as group management and student registration, without conflicts.
    - **Completeness Check**: The requirement should specify whether lecturers can also modify or delete groups for completeness.
    - **Realism Check**: Creating exercise groups is feasible within the project scope and timeline.
    - **Verifiability Check**: This requirement is verifiable through testing by confirming that lecturers can create exercise groups as intended.

- A.3 Lecturers shall be able to set group details like session times and the number of students each group can handle.
    - **Validity Check**: This requirement is valid as it addresses the need for lecturers to manage group capacities and schedules.
    - **Consistency Check**: It is consistent with other group management features, but ensures no contradictions with student preferences or availability.
    - **Completeness Check**: The requirement could be more complete by specifying additional details, such as location or prerequisites for the group.
    - **Realism Check**: Setting group details is realistic.
    - **Verifiability Check**: This requirement is verifiable by testing the ability of lecturers to set and modify group details as described.
- A.4 Students log in to view available sessions.
    - **Validity Check**: The requirement is valid as it meets the students' need to access and choose sessions that fit their schedules.
    - **Consistency Check**: Ensure this requirement is consistent with other user-related requirements like registration and exercise group assignment.
    - **Completeness Check**: Specify if students can filter sessions by specific criteria such as availability or preferred time slots
    - **Realism Check**: Viewing available sessions is feasible within the system's planned functionality.

    - **Verifiability Check**: This requirement is verifiable by testing the login process and ensuring students can successfully view all available sessions.

- A.5 Students shall be able to register for all courses they want to attend exercises for in one semester.
    - **Validity Check**: This requirement is valid as it reflects the need for students to manage their semester schedule efficiently.
    - **Consistency Check**: Ensure it is consistent with other requirements, such as group availability and student preferences.
    - **Completeness Check**: Specify if there are any restrictions or prerequisites for course registration.
    - **Realism Check**: Registering for courses is realistic given the current scope of the system.
    - **Verifiability Check**: This requirement is verifiable by testing the registration process to confirm students can select all desired courses without issues.

- A.6 The system shell sends Notifications that will let students know which groups they've been assigned to.
    - **Validity Check**: This requirement is valid as it ensures students are informed about their group assignments, which is essential for planning.
    - **Consistency Check**: Ensure consistency with other requirements regarding communication and group assignment processes
    - **Completeness Check**: Specify how the notifications will be sent (e.g., email, in-app message) to provide clarity.
    - **Realism Check**: Sending notifications is feasible within the scope of the system, if reliable communication mechanisms are implemented.
    - **Verifiability Check**: This requirement is verifiable by testing the system's ability to send notifications correctly to students after group assignments are made.

### Quality requirements and their respective quality attribute

- B.1 The system shall distribute students evenly among the groups.
    - **Validity Check**: This requirement ensures fairness for students and reflects their needs, making it a valid objective.
    - **Consistency Check**: There are no conflicts but verify alignment with other requirements like student preferences and group sizes.
    - **Completeness Check**: The requirement could be more complete by specifying how the system handles scenarios where even distribution is impossible.
    - **Realism Check**: Even distribution is feasible but may require manual adjustments for edge cases.
    - **Verifiability Check**: This requirement is verifiable by defining metrics (maximum difference between group sizes) and testing the system under various scenarios.
- B.2 The System shall minimize the number of students that could not be assigned a group
    - **Validity Check**: This requirement addresses the need to provide group access to as many students as possible, making it valid

    - **Consistency Check**: The requirement aligns with other system goals, such as fair group distribution and conflict management, without contradictions.
    - **Completeness Check**: Specify what actions should be taken for students who remain unassigned (e.g., notify them or provide manual assignment options).
    - **Realism Check**: Minimizing unassigned students is realistic with efficient algorithms, but some manual intervention may be necessary for difficult cases.
    - **Verifiability Check**: Verifiable by measuring the number of unassigned students after each distribution run and ensuring it is minimized to an acceptable level.
- B.3 The system shall minimize the number of manual administration processes
    - **Validity Check**: This requirement reflects the need to reduce administrative workload, making the system more efficient for lecturers and admins.
    - **Consistency Check**: There are no contradictions but ensure consistency with requirements involving automated scheduling and notifications.
    - **Completeness Check**: Specify which manual tasks should be automated to achieve minimal manual administration
    - **Realism Check**: Reducing manual processes is feasible within the project scope, but requires adequate automation features and validation
    - **Verifiability Check**: Verifiable by comparing the number of manual tasks before and after system implementation, ensuring a reduction in manual workload.

### Constraint

The system should be developed in Java.

- **Validity Check:** This requirement is valid as it specifies the preferred programming language, ensuring alignment with the development team's expertise and compatibility with university systems.
- **Consistency Check:** There are no conflicts with other requirements, but make sure other technical requirements do not impose restrictions incompatible with Java.
- **Completeness Check:** The requirement could be more complete by specifying the Java version or any specific frameworks that need to be used.
- **Realism Check:** Developing in Java is realistic given its widespread use, especially in educational settings. Ensure that the development team has the necessary expertise.
- **Verifiability Check:** This requirement is verifiable by reviewing the source code to confirm that all components are developed in Java.

### project requirement

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

- **Validity Check**: This requirement is valid as it aligns with the project timeline, providing clear deadlines for testing and deployment phases.
- **Consistency Check**: There are no conflicts with other requirements; ensure all tasks align with the set timeline.
- **Completeness Check**: Specify whether there are milestones or intermediate deadlines for different phases leading up to the deployment.

- **Realism Check**: The timeline is realistic given adequate resources and planning. Confirm that the scope can be achieved within the given timeframe.
- **Verifiability Check**: This requirement is verifiable by tracking project progress to ensure milestones and deadlines are met according to the schedule.

### process requirement

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

- **Validity Check:** This requirement is valid as it provides an opportunity for students to gain hands-on experience, aligning with the educational goals of the university.
- **Consistency Check:** It is consistent with other project requirements and objectives, ensuring that students' roles are integrated into the development process.
- **Completeness Check:** Specify how students will be involved (e.g., specific phases, tasks, or roles) to ensure the participation is well-structured.
- **Realism Check:** This is realistic as long as the appropriate guidance and training are provided to students.
- **Verifiability Check:** Verifiable by tracking students' participation and contributions throughout the development and testing phases.

**Exercise 4: Use case**

| | |
|---|---|
| Title | Assign course |
| Actors | Student, Course coordinator, lectures |
| Pre-conditions | [What must be true or happen before the use case run]<br>Student must be enrolled in one or more courses for the semester.<br>The EGD system has Current information about available exercise groups, lab sessions, and their respective time slots. |
| Post-conditions | [What must be true or happen after the use case run]<br>• The student is assigned to exercise groups or lab sessions with minimal scheduling conflicts.<br>• If conflict arise, the student must be informate about it and get a solution suggestion (or next step suggestion)<br>• The student receives a final schedule, confirmed via email. |
| Trigger | [This is the event that cause the use case to be initiated]<br>The student logs into the EGD system to schedule their exercise groups. |
| | **Step Description** |
| Main Success scenario | Basic Flow when nothing goes wrong<br>1. **Student:** Enter Log in data<br>2. **System:** verify authority<br>3. **System:** display an overview of available courses (and labs slot)<br>4. **Student:** Enter unavailability times (submitting her availability)<br>5. **System:** confirm no scheduling conflicts<br>6. **System:** assigns the student to the courses<br>7. **System:** send confirmation Email<br>8. **Student:** receives confirmation Email with the information about the courses und schedules for the semester |
| Alternative Paths | Alternative Flow vatiations oft he main scenario<br><br>5.a [System: recognize conflict]<br>5.a.2 **System:** Show Conflict Massage<br>5.a.3 **System:** assigns the student to the courses and send (or show) Student message to contact Course Coordinator (or lecture)<br>-> use case ends<br>_____<br><br>5.a.3 [System cannot assign the student to the courses]<br>5.a.3.II **System:** send (or show) Student message to contact Course Coordinator (or lecture)<br>-> use case ends |

| | |
|---|---|
| | |