Exam System for VIA University College – Analysis Document

Summary

VIA University College Software Engineering needs a system that can arrange all exam dates, time and rooms needed. The exam type needs to be specific (Oral or Written) in order to work with our system.

The system is focusing only on scheduling exams for Software Engineering students between 1st and 4th semester, but it can be adjusted for other study programs in the university. The exam system needs to be easily accessible in order to arrange the proper time, date and room needed for any given exam. The program should be able to handle names and store information about rooms (when they are free and when they are used for classes or something else.)

System will be used mostly from the secretaries, so we put them as our main target user, design wise should be simplistic. The program should be easy to use and understand.

Functional requirements

**Critical priority:**

1. As a secretary, I want to be able to add an exam at a specific date and room, in order to prevent overlapping.

2. As a secretary, I want to be able to make lists of exams and students attending the exams, for

them to know where and when the exam is held.

3. As a secretary, I want to have a list of examiners, their availability and which exams they are attending, in order to add them in the system.

4. As a secretary, I want to be able to add, edit and delete scheduled individual exams, in order to make, change and remove data and exams if necessary.

5. As a secretary, I want to be able to change and update the schedule at any given time, in order to prevent overlapping on dates and rooms.

6. As a secretary, I want to be able to check and edit if necessary, whether a class has more than one written exam in one day, in order to not exceed the limit of one written exam per day per student.

**High Priority:**

7. As a secretary, I want to have access to student information, in order to be able to arrange proper oral exams. Also, this information can be used to reach the student via email or phone number.

8. As a secretary, I want to have a list with equipment for each class, in order to make sure oral exams can take place there, e. g.: has HDMI.

9. As a secretary, I want to be able to add teacher names for the exams and change when

necessary, in order to keep track of number of examiners attending.

**Low Priority:**

10. As a secretary, I want to be able to set written exams before the oral ones, in order to let students be more prepared.

11. As a secretary, I want to be able to arrange a 3-day SEP exam, as this is the standard

procedure.

12. As a secretary, I want to be able to notify students for upcoming exams, room numbers and dates and send them a reminder message 3 days before the exam, in order to make sure they

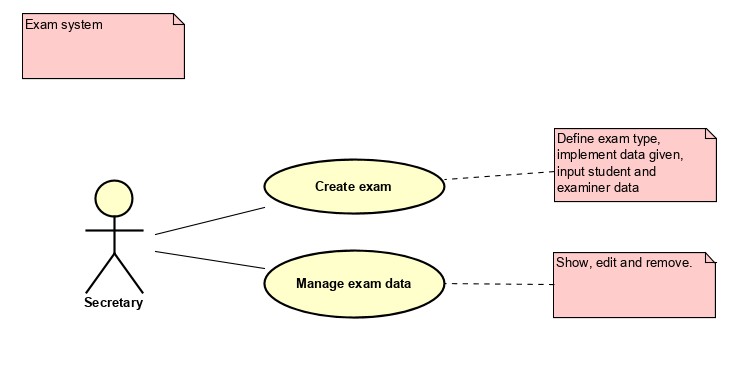
are aware of the schedule.

13. As a secretary, I want to be able to let the janitors know when they should enter a given room and clean it before/between/after exams. This will be possible if the program has a function that shows the exact time when rooms are available, and/or for how long.

Non-functional requirements

14. User interface should be practical and easy to use.

User Case Diagram



Use case descriptions

**Use Case Create exam**

**Summary** Add an exam and add information for the exam – semester, date, room, participants and what type of exam it is.

**Actor** Secretary

**Precondition** Information should be available, or created (date, room, type of exam, participants, semester)

**Postcondition** An exam has been created. The information about the exam has been added, which includes date, room, participants and participants’ semester.

**Base sequence** 1. Add the exam depending on what type it is – written or oral

2. Add the participants’ semester.

3. Add date for the exam.

4. Add available room for exam.

5. Add examiners for the exam.

6. Add students for the exam from the specified semester.

7. Approve the exam.

**Exception**

**Sequence**

If the user tries to create the same exam for a subject that has already been created once:

Step 1-10 as base sequence.

The process should be terminated, to prevent overlapping of data.

**Note** The process can be canceled at any time.

If any information written and scheduled is wrong, the information will be easy to change and put in properly in the next use case.

**Use Case Manage exam data**

**Summary** Edit and remove exam data

**Actor** Secretary

**Precondition** The exam should be created.

**Postcondition** The exam data is now visible to users. Wrong data can be changed or removed.

**Base sequence**  SHOW:

1. If the current date is between February and June, the system shows a calendar of June and available dates to add an exam.
2. If the current date is between July and January, the system shows a calendar of January and available dates to add an exam.

EDIT:

In case of wrong input, change any information needed in the next steps:

1. If you want to delete an exam, proceed to step 1 in the removing base sequence.

2. Change exam date, if not needed proceed.

3. Change room, if not needed proceed.

4. Change examiners, if they are wrong or choose guards, if the exam is written. If not needed, proceed.

5. Validate and delete changes, if not needed go to step 1 again or terminate the program.

**Exception**

**Sequence**

REMOVE:

1. Validate and delete it from the system.
2. Terminate the program.

If the user adds an exam and a student/examiner already has/is assigned to an exam or the room is occupied. The process should be terminated in order to prevent overlapping.

**Note** For example, if the user chooses the wrong examiner and realizes the mistake after saving the information, the user will easily access the same data about the examiner and change to the right one.

If a room is reserved for another exam already, the program will give the user a notice and make the user change the room to an available one from a list.

Link between requirements and use cases

**Use Case Covered requirements**

Create exam 1, 2, 3, 4, 5, 6, 7, 9

Manage exam data 1, 4, 6, 7, 8, 10, 11, 12, 13

**Requirements Related Use Case**

1 Create exam

2 Create exam

3 Create exam

4 Manage exam data

5 Manage exam data

6 Manage exam data

7 Manage exam data

8 Create exam

9 Manage exam data

10 Create exam, Manage exam data

11 Create exam

12 Create exam

13 Manage exam data

Activity diagram

