# Use case specification

## Use case: Update user information

*Brief description*

This use case describes how the university staff updates the information of a specific user.

*Basic flows*

The use case begins when an employee of the university wants to update information of a user.

1. The employee selects to update user information.
2. The system asks for the new user information.
3. The employee inputs the required information.
4. The system updates the database with the new information.
5. The system displays a confirmation message.
6. The use case terminates.

*Preconditions*

The employee has to be logged with an admin account. The user must previously exist in the database.

*Postconditions*

Upon successful operation, the database contains the newly added information about the client.

### Use case: Assign optional courses

*Brief description*

This use case describes how the university staff assigns the optional courses.

*Basic flow*

The use case begins when an employee of the university has to assign the optional courses for the new year.

1. The employee selects to assign the optional courses.
2. The system shows a list of optional courses with more than 20 followers and a list of optional courses with less than 20 followers. (The order the courses are not organized?)
3. The employee selects to redistribute the followers based on the organization rules.
4. The system selects the list of the courses that will be organized.
5. The system shows the list of official courses and the number of followers.
6. The use case terminates.

*Preconditions*

The employee has to be logged with an admin account.

*Postconditions*

New records for the courses are created.

## Use case: Sign contract

*Brief description*

This use case describes how the students sign the contract with the university.

*Basic flow*

The use case begins when a student has to choose the courses at which will attend for the new year.

1. The student selects to see the courses for the new year.
2. The system shows a list of mandatory courses and a list of optional courses.
3. The student orders by preference the optional courses.
4. The student selects to send the contract information. (?)
5. The system saves the student’s preferences.
6. The use case terminates.

*Preconditions*

The student has to be logged with a student account.

*Postconditions*

The system has set a new follower for the top course selected by the student.

## Use case: Upload exam results

*Brief description*

This use case describes how the teachers (?) upload the exam grades for the students attending their course.

*Basic flow*

The use case begins when a teacher wants to upload the grades for each student attending his course.

1. The teacher selects to upload grades.
2. The system shows a list of courses that the he teaches.
3. The teacher chooses a course.
4. The system shows a list of students and asks the teacher to input grades.
5. The teacher inputs grade for the students and selects to upload the modification.
6. The system updates the database.
7. The system displays a confirmation message.
8. The use case terminates.

*Preconditions*

The teacher has to be logged with a teacher account.

*Postconditions*

Given every students grade, the status of students at that course is set.

## Use case: Group approved courses

*Brief description*

This use case describes how the teachers group the approved courses into several groups.

*Basic flow*

The use case begins when a teacher wants to group the approved courses.

1. The teacher selects to group courses.
2. The system shows the list of created groups.
3. The teacher groups courses.
4. The teacher selects to finish the operation.
5. The system updates the database.
6. The use case terminates.

*Alternate flows*

A1. Courses not grouped

In step 4 of the basic flow there is one or more courses that have not been assign to a group. The system displays a message notifying the teacher and asks if he wants to proceed with saving the data as it is.

1. If the teacher wants to continue assigning courses into different groups, the system shows the screen with the approved courses.
2. If the teacher wants to end the operation the use case continues.

A2. Create new groups

In the first step of the subflow ‘Group courses’ the may want to add a new course.

1. The teacher selects to add a new group.
2. The system creates the group.

The subflows continues.

*Subflows*

S1. Group courses

1. The teacher selects a group.
2. The system shows the list of currently added courses in the group.
3. The teacher selects to add a new course to the group.
4. The systems shows the list of optional courses not yet grouped.
5. The teacher select course and selects to save the course in the current group.
6. The system updates the list of currently added courses in the group.
7. The teacher selects to end modifications to this group.

*Preconditions*

The teacher has to be logged with a teacher account and has to be a chief of department.

*Postconditions*

Upon grouping every approved course, the students can start choosing the courses for the new year.