Flow of events

LOGIN

SignIn

Id and name: SignIn

Primary actor: User

Description: User signs into account

Trigger: -

Preconditions: Faculty staff created the account

Postconditions: User can do other actions

Normal Flow:

1. User logs in using the unique username and the password
2. App checks if the credentials are correct
3. If the credentials are correct, the user can access the account

Alternative Flos:

3.1 If the credentials are not correct, the message “wrong email/password” is shown.

Exceptions:

Account has been deleted

UpdateProfile

Id and name: UpdateProfile

Primary actor: User

Description: User updates the information in his/her account

Preconditions: User is signed in

Postconditions: The personal information is changed

Normal Flow:

1. User goes to the page where their profile is displayed.
2. App displays the personal information that is already on their account (if there is any)
3. User updates/uploads the personal information he/she wants to be on their profile
4. User clicks the “Save” button, so that the changes are saved

Alternative Flow:

4.1 User clicks ”Cancel”

5. App goes back to the menu

LogOut:

Id and name: LogOut

Primary actor: User

Description: User logs out of their account

Trigger: -

Preconditions: User is signed in

Postconditions: User is unable to make any other actions on app

Normal Flow:

1. User clicks the “Log out” button
2. App displays a message “Are you sure you want to log out?”
3. User clicks “Yes”.
4. App logs the user out

Alternative Flow:

3.1 User clicks “No”

4. App shows the user back to the main page

COURSES

EnrollYear:

Id and name: EnrollYear

Primary actor: Student

Description: Student enrolls in the study years they want to take part in

Trigger: App shows notification before the beginning of the year

Preconditions: User signed in

Postconditions: Curriculum for the selected year/years is saved

Normal Flow:

1. Student selects the number of years, and which years to enroll in (At most 2).
2. App saves the curriculum for the selected years (so that the user will be able to see it when he/she wants)

Alternative Flow:

1.1 Student does not select any years to enroll in

2. App automatically enrolls student into just one year (previous passed +1)

Exceptions: Student drops out of university

ViewCurriculum:

Id and name: ViewCurriculum

Primary actor: Student

Description: Student sees the curriculum for the chosen year

Preconditions: User signed in, EnrollYear

Postconditions: -

Normal Flow:

1. Student selects year
2. App displays the chosen curriculum

ListOptionals:

Id and name: ListOptionals

Primary actor: Student

Description: Student selects the optional courses for that year

Trigger: Notification “It’s time to choose your optional courses” pops up/

Preconditions: User signed in, EnrollYear, ViewCurriculum, ApproveOptionals

Postconditions: App generates final schedule

Normal Flow:

1. Student selects optional course from the curriculum one by one.
2. App displays a draft of the list of preferred optional courses in decreasing order. Each element on the list is added as the user clicks on it and removed from the curriculum.
3. If the Student wants to remove an optional from the draft displayed, he clicks on the optional, the draft is updated, and the optional is back on the curriculum.
4. After the draft is correct, Student clicks “Save”

Alternative Flow:

1.1 Student does not select any optional course before the set deadline

2. App generates final curriculum based on what places are left, alphabetically.

Exception: Student drops out

SignContract:

Id and name: SignContract

Primary actor: Student

Description: Student signs and uploads contract

Trigger:

Preconditions: User signed in, EnrollYear, ListOptionals

Postconditions: -

Normal Flow:

1. Student selects year
2. App generates a contract
3. Student fills in and signs the contract
4. Student uploads contract
5. Save

ViewGrades

Id and name: ViewGrades

Primary actor: Student

Description: Student views the grades

Preconditions: SignIn, EnrollYear, ListOptionals, SignContract

Normal Flow:

1. Student selects year
2. App displays list of disciplines and the corresponding grades (students can only see their grades)

ViewGrantStatus:

Id and name: ViewGrantStatus

Primary actor: Student

Description: Student checks if their grant status

Trigger:

Preconditions: User signed in, EnrollYear

Normal Flow

1. Student selects year
2. App displays the status for the scholar grant

ProposeOptionals

Id and name: ProposeOptionals

Primary actor: Teacher

Secondary actor: Chief of department

Description: Teacher uploads the proposed optional courses

Trigger: -

Preconditions: SignIn

Postconditions: Chief of the department receives proposed optional courses

Normal Flow:

1. Teacher uploads the name and description of at most 2 optional courses
2. App sends the list to the page of the chief of the department.

GradeStudent

Id and name: GradeStudent

Primary actor: Teacher

Secondary actor: Student

Description: Teacher uploads the grades

Preconditions: SignIn

Postconditions: Student and chief of the department receives grades

Normal Flow:

1. App displays the list of all students.
2. Teacher selects the name of a student
3. App displays a section where the grade can be filled in
4. Teacher fills in with the corresponding grade.
5. Teacher saves the changes made
6. App shows the student and the chief of the department the corresponding grade.
7. App goes back to the list of all students

ViewAssignedGrades:

Id and name: ViewAssignedGrades

Primary actor: Teacher

Description: Teacher views all the grades

Preconditions: SignIn

Postconditions: -

Normal Flow:

1. App displays the list of all students and their grades in the teacher's subject.

ViewOptionals:

Id and name: ViewOptionals

Primary actor: Chief of department

Description: Chief of department views the optionals

Preconditions: SignIn, ProposeOptionals

Postconditions:

Normal Flow:

1. App displays the list of all optional courses proposed by all the teachers in the given department.

ApproveOptionals

Id and name: ApproveOptionals

Primary actor: Chief of department

Secondary actor: Teacher, Student

Description: Chief of department makes the final list of optional courses

Trigger: Deadline for proposing optional courses expired.

Preconditions: SignIn, ProposedOptionals

Postconditions: Students and Teachers receive final list with the optional courses for that year.

Normal Flow:

1. App displays list of optional courses
2. Chief of department selects optional courses that are not approved
3. Chief of department selects a bin button
4. Trigger pops up with the text “Are you sure you want to delete this optional?”
5. Chief of department selects yes.
6. App deletes the optional from the list of optional courses
7. Chief of department selects save. (Creates final list)
8. App lets teachers know which optional courses have been approved
9. App puts approved optional courses on the curriculum.

Alternative Flow:

5.1 Chief of department selects no.

6. App does not delete optional course
7. Chief of department selects save. (Creates final list)
8. App lets teachers know which optional courses have been approved
9. App puts approved optional courses on the curriculum.

Exception: No optional courses have been proposed

SpecifyMaxStudents

Id and name: SpecifyMaxStudents

Primary actor: Chief of department

Description: Teacher sets a limit to the number of students that can take an optional class

Preconditions: SignIn, ApproveOptionals

Postconditions:

Normal Flow:

1. App displays the final list of approved optional courses
2. Chief of department selects an optional course
3. App displays a section where the maximum number of students can be filled in.
4. Chief of department fills in the maximum number of students that can join the given course.
5. Chief of department saves the changes made.
6. App goes back to the list of all optional courses

ViewTeacherPerformance

Id and name: ViewTeacherPerformance

Primary actor: Chief of department

Description: Chief of department checks the performance of the teachers

Preconditions: SignIn, GradeStudents

Postconditions: -

Normal Flow:

1. App displays the list of all teachers in the given department
2. Chief of department selects a teacher
3. App displays a list of all courses the teacher
4. Chief of department selects a course
5. App displays all the results obtained at that course.

Alternative Flow:

2.1 Chief of department selects sort by results
3. App displays a list of all disciplines with teachers sorted by result in decreasing order

Exception: Students have not been graded yet

ViewDisciplines:

Id and name: ViewDisciplines

Primary actor: Chief of department

Description: Chief of department views the list of all disciplines

Preconditions: SignIn

Postconditions: -

Normal Flow:

1. App displays the list of all teachers in the given department.
2. Teacher selects a teacher
3. App displays the list of all the course's name given by the teacher.

ViewStudentsByGroups

Id and name: ViewStudentsByGroups

Primary actor: Faculty administrative staff:

Description: Faculty administrative staff views all students ordered by grades in a group

Preconditions: SignIn, GradeStudents

Postconditions: -

Normal Flow:

1. App displays list of years of study
2. Faculty administrative staff selects a year
3. App displays the list of all groups from the given year
4. Faculty administrative staff selects a group
5. App displays a list of students ordered alphabetically
6. Faculty administrative staff selects the option "Sort by grades"
7. App displays the list ordered by grades in decreasing order

Exception: Students have not been graded yet

PrintStudentsByGroups

Id and name: PrintStudentsByGroups

Primary actor: Faculty administrative staff:

Description: Faculty administrative staff prints all students ordered by grades in a group

Preconditions: SignIn, GradeStudents,ViewStudentsByGrades

Postconditions: -

Normal Flow:

1. Teacher selects the print option
2. System prints the list of students

Exception: Printer is not connected/ does not work

ViewStudentsByYear

Id and name: ViewStudentsByYear

Primary actor: Faculty administrative staff:

Description: Faculty administrative staff views all students with the yearly average withing a selected interval

Preconditions: SignIn, GradeStudents

Postconditions: -

Normal Flow:

1. App displays list of years of study
2. Faculty administrative staff selects a year
3. App displays the list of all groups from the given year
4. App displays a list of students ordered alphabetically
5. Faculty administrative staff selects the option “Filter by grades”
6. Faculty administrative staff selects interval
7. App displays the list of students with results within set interval

Alternative Flow:

7.1 App displays message: “There are no students within the given interval”

Exception: Students have not been graded yet

PrintStudentsByYear

Id and name: PrintStudentsByYear

Primary actor: Faculty administrative staff:

Description: Faculty administrative staff prints list of all students with the yearly average withing a selected interval

Preconditions: SignIn, GradeStudents, ViewStudentsByYear

Postconditions: -

Normal Flow:

1. Teacher selects the print option
2. System prints the list of students

Exception: Printer is not connected/ does not work

GenerateGrantsList:

Id and name: GenerateGrantsList

Primary actor: Faculty administrative staff:

Secondary actor: Student

Description: Faculty administrative creates list with students who will receive grants.

Preconditions: SignIn, GradeStudents, ViewStudentsByYear

Postconditions: -

Normal Flow:

1. Faculty administrative staff checks the funding level to determine how many grants there are
2. Faculty administrative staff selects each student that receives a grant.
3. Faculty administrative staff changes the status student.
4. Faculty administrative staff saves the changes.
5. App up updates the grant status of the given students.

AssignStudents

Id and name: AssignStudents

Secondary actor: Student

Trigger: Deadline for ListOptionals expires

Description: This is the algorithm by which the app creates the final curriculum

Preconditions: ListOptionals, ApproveOptionals, SpecifyMaxStudents

Postconditions: Final schedule is created

Normal Flow:

1. App goes through each student's optional courses list in order of first sent.

2. App check availability for first choice
3. After each student's list has been checked there is a minimum of 20 students and optional is approved
4. App creates final schedule for student

Alternative Flow:

3.1 After each student's list has been checked there aren't 20 students and the optional is not approved

3.2 App repeats this step for the second/third choice

Exception: None of the choices is selected, app redistributes optional randomly

GenerateContract

Id and name: AssignStudents

Secondary actor: Student

Trigger: Deadline for ListOptionals expires

Description: This is the algorithm by which the app creates the contract

Preconditions: ListOptionals, AssignStudents

Postconditions: Contracts are available for Students

Normal Flow:

1. App builds a contract using a template with the corresponding curriculum.
1. App makes the contract available on the student's page