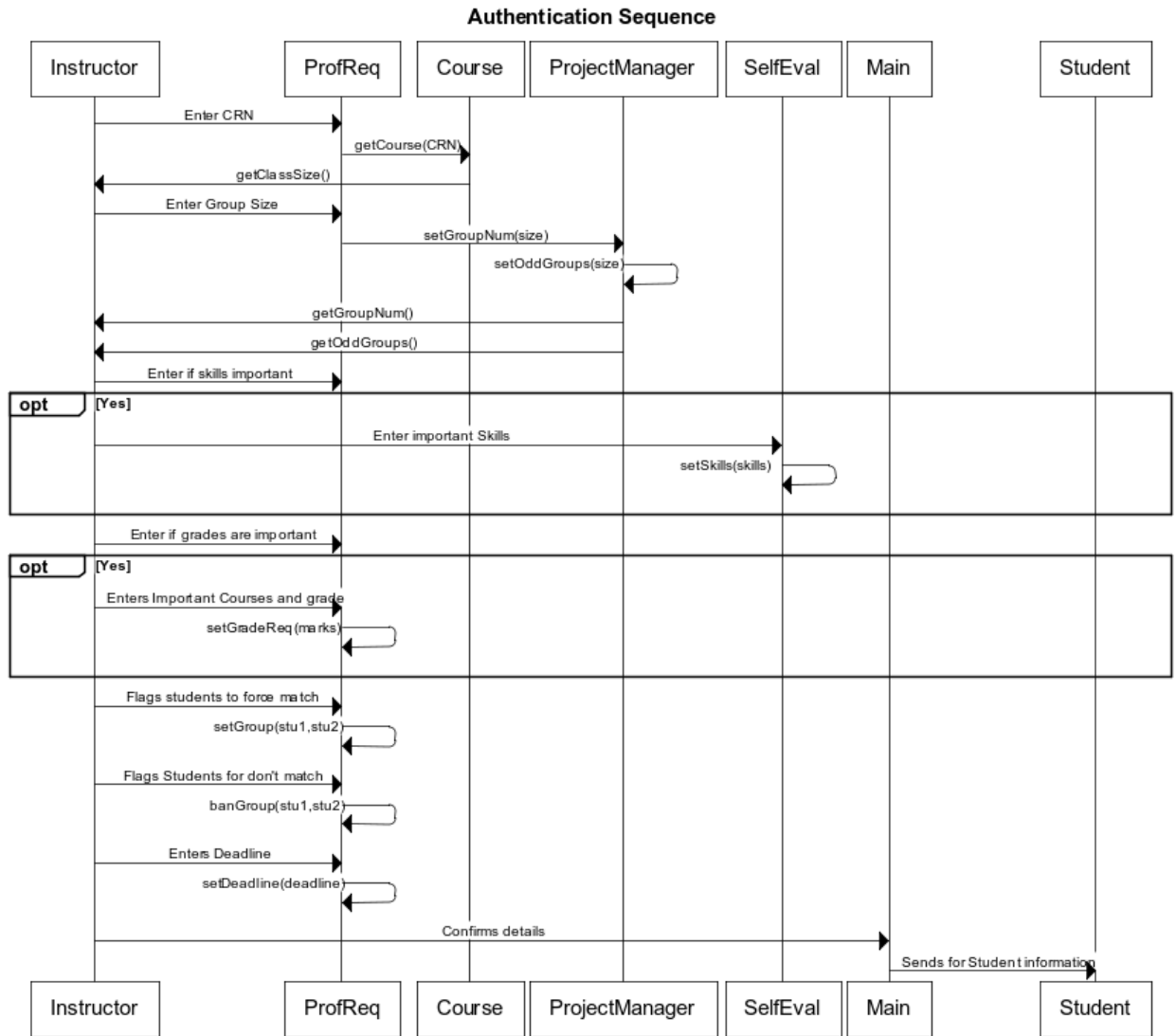Why the classes Marks, SelfEval, Skills, ProjManager, Main and Overview don't have attributes? It makes sense to have an interface (Overview) of the Group. However, the rest of the classes seem to require some attributes. Example: Answers for the SelfEval class.

Are you sure you don't need to store information of the instructor? What if two different grouping tasks need to be done at the same time? How are you going to control the access to each of them?
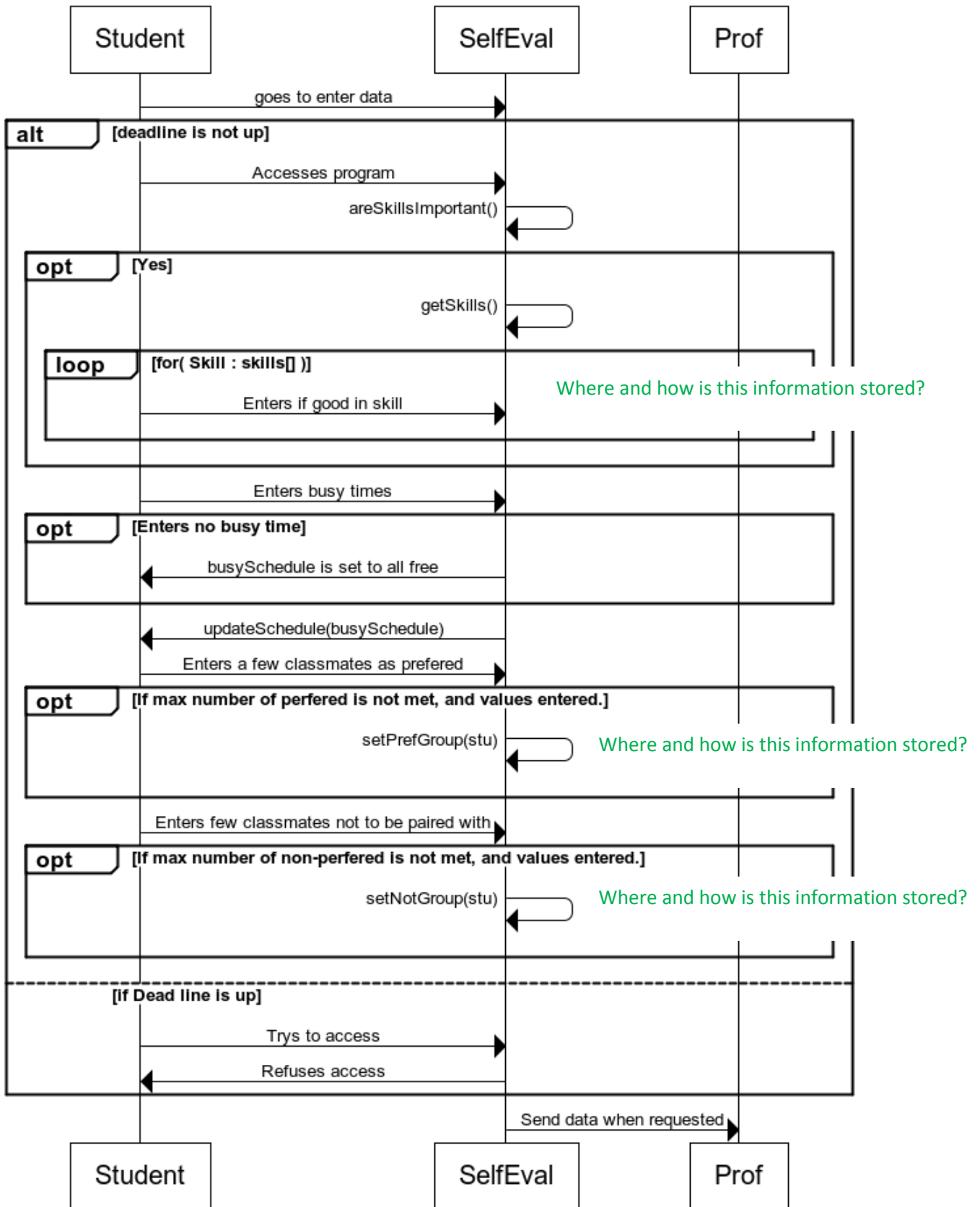
Where are you storing grouping preferences of the Professor and the Students?

Where are you storing the deadline?

89/100

Authentication Sequence

Instructor	ProfReq	Course	ProjectManager	SelfEval	Main	Student

Instructor → ProfReq: Enter CRN

ProfReq → Course: getCourse(CRN)

ProfReq → Instructor: getClassSize()

Instructor → ProfReq: Enter Group Size

ProfReq → ProjectManager: setGroupNum(size)

ProjectManager → ProjectManager: setOddGroups(size)

ProjectManager → Instructor: getGroupNum()

ProjectManager → Instructor: getOddGroups()

Instructor → ProfReq: Enter if skills important

opt [Yes]

ProfReq → SelfEval: Enter important Skills

SelfEval → SelfEval: setSkills(skills)

Instructor → ProfReq: Enter if grades are important

opt [Yes]

Instructor → ProfReq: Enters Important Courses and grade

ProfReq → ProfReq: setGradeReq(marks)

Instructor → ProfReq: Flags students to force match

ProfReq → ProfReq: setGroup(stu1,stu2)

Instructor → ProfReq: Flags Students for don't match

ProfReq → ProfReq: banGroup(stu1,stu2)

Instructor → ProfReq: Enters Deadline

ProfReq → ProfReq: setDeadline(deadline)

Instructor → Main: Confirms details

Main → Student: Sends for Student information

Instructor	ProfReq	Course	ProjectManager	SelfEval	Main	Student

www.websequencediagrams.com

Where are matches, deadline and gradeRequests being stored?

You need to differentiate between the actor Student and the class Student. The first one is who triggers this Use Case. The second one is part of the System and you need to define its attributes and methods

Instructor | Main | ProfReqs | Course | SelfEval | ProjectManager | ProfReq | Group | Overview | Student

Tries to initiate Program

Fetches deadline

getDeadline()

alt [Deadline not met yet]

[Deadline has occured]

getCourse()

getClassList()

areSkillsImportant()

opt [yes]

getSkills()

getGroupNums()

getOddGroups()

getProfMatches()

getProfBans()

loop [for(Group g:profMatches)]

Search for group with existing member

alt [If one is found]

add(otherStudent)

[not found]

add(student1)

add(student2)

are thay allow to have prefered partners

opt [yes]

Make preferd groups

Compare Students

makeGroups

viewGroups()

opt [Instructor wishes to change group]

editGroup(groups)

g1.remove(student1)

g2.remove(student2)

g1.add(student2)

g2.add(student1)

getGroup()

Instructor | Main | ProfReqs | Course | SelfEval | ProjectManager | ProfReq | Group | Overview | Student

www.websequencediagrams.com