Project Delivery 6 and 7

Spring 2023
Software Engineering

Maíra Marques Samary PhD

Project – TA Application System

The goal of the project is to create a system that allows any department to manage students applying to work as TAs on its courses. At the end of the project the system will be connected with EagleApps, where students will only be able to apply to courses they have already done. And courses will be populated automatically from EagleApps.

Requirements didn't change!

Critical Requirements:

1) A user can be an instructor, a student or an admin
2) An instructor/admin will have at least name and email as required fields
3) An instructor should be able to create/edit a course. An admin should also be able to edit courses.
4) An instructor should be able to edit only the courses he/she created. But the admin should be able to edit everything.
5) A student person that has name, email, relation to CS department (major BA/BS or minor), Eagled, cohort (freshmen, sophomore, junior, senior), state (open to work, hired).
6) All courses will have a number, a name, a description, a section, an instructor, date, time and number of TAs needed. There will be 2 types of courses with discussion and without discussion. Courses should also have Boolean or flag marking homework/assignments graded in meetings (yes/no), number of office hours required per week, and an extra field (string) where the instructor can add relevant information about the course.
7) The courses with discussions can have multiple discussion session, and each ones have its own number, date and time.
8) Students should be able to see all courses and to easily figure out courses have open positions - I call this summary. Students' summary can be the landing page after the summary is working.
9) Admin should be able to easily see all courses, how many TA's positions are still open, how many are closed and get into a course info and/or student application easily from the summary page. The summary can be the landing page once it is functional.

10) The system should handle states: open for application/closed for application. The state should be managed as a whole (the whole system and all its applications) and by course and section.
11) The system state will be changed to "open for application" by the admin, that will do that as soon as all instructor have added their courses, sections and discussions. As soon as all TA's positions are filled the administrator will be able to change the system state to "closed".
12) Application should be done by students, selecting course and/or instructor they want to apply.
13) Students should also be able to write something in their application, so that the instructor can see and have a better idea of who the student is.
14) Students can have a maximum of 5 open applications for a TA position - not hard coded, it should be field in a table
15) As soon as a course/section has applications, the instructor should be able to see the applications and should be able to select the TA's.
16) When the instructor selects a TA, the system should send an email to the student inviting him/her to be the TA for that course and instructor, and the student can agree or decline the invitation.
17) If the student agrees and the number of open TA's for that course is reached, than that course/section state should be closed.
18) If the students decline, the instructor should get a message and the system has to "save" and show to the instructor that the student decline the invitation.
19) When the number of TA's he selects matches the number defined in the course/section, the course/section state should change to closed, and the students selected will receive an email informing that they were selected and they can agree or not to work as TAs for this course/section.
20) When an instructor finish choosing TA's for their course and students accept the invitation all students that applied should be notified that they were not accepted, and their open applications should be changed to a smaller number
21) As soon as the student agrees to be a TA for one course/section the student state should be changed to hired, and all other applications he/she made should become "invisible". And instructors should not be able to select or see their application in other courses/sections.
22) When a TA is selected to a course, he/she can't be selected to be a TA in any other course unless they decline the invitation received

Non-Functional Requirements:

- The system must be done using Django / HTML/ CSS / Bootstrap only
- The system should store the information a user entered and show again to the user, if the system gets offline or if anything happens in the middle of the process, he/she is doing.

Non-Critical Requirements:

- Students should be able to select one or multiple courses to apply at the same time.

For Delivery 6 we have a couple of formal deliverables:

- Report
- Presentation
- Link of deployed system

The report must have:

- Summary of compliance with the requirements: which requirements are satisfied / partially satisfied / not satisfied
- Description of the methodology followed by the team: tools used, frequency of meetings, work planning, etc.
- Description of your data model (UML class diagram)
- Interaction Overview Diagram (UML) https://en.wikipedia.org/wiki/Interaction_overview_diagram
- Post-Morten analysis:
    - Are you proud of our finished deliverables? If yes, what made them great? If no, what was wrong or missing?
    - Did we get the results we wanted and did it make impact?
    - Which of our methods or processes worked particularly well?
    - Which of our methods or processes were difficult or frustrating to use?
    - How would you do things differently next time to avoid this frustration?
    - What else could we do better next time?
    - What was the most gratifying or professionally satisfying part of the project?
    - What would you or the team do differently with the knowledge you have now? In terms of technology and in terms of people (team)?

Presentation (20 minutes):

Summary of everything you did on the report and the demo. In the demo you have to show all the process from a professor creating a course until a student being accepted or not to be a TA in a course. The demo must be done from Railway (or another service that you are familiar).

About the deployment:

Here is a link to a sample tutorial deployment in Railway. You can deploy in another place, if you know and are familiar with other web services:

https://github.com/CSCI3356Spring2023/DeployExample

Be aware that deployment is highly dependable of everything you have done in your code, versions of libraries/modules, installing everything…. So, the tutorial is just a starting point on what do.

If a team is capable of finishing all the requirements and doing everything requested on Delivery 6 well. They are not required to do Delivery 7. The grade of Delivery 6 will be repeated on Delivery 7. Otherwise, you have to do/present to Delivery 7.

Project Delivery 7 (each group has to schedule a meeting with me before May 14 – the whole team must attend):

RUBRIC – Delivery 6

Item	100%	70%	40%	20%	0%	Max Points
Functional Requirements * The list of requirements being considered here is according to what each team showed on Delivery 5. Teams that don't finish the requirements by Delivery 6 will have to show new requirements (functionality) working, not necessarily everything.	All done and working	Almost all done and working	Around half of it done	A bit done	Nothing done	2
Non- Functional Requirements	All done and working	Almost all done and working	Around half of it done	A bit done	Nothing done	1
Report • Summary of compliance with the requirements: which requirements are satisfied / partially satisfied / not satisfied • Description of the methodology followed by the team: tools used, frequency of meetings, work planning, etc. • Description of your data model (UML class diagram) • Interaction Overview Diagram (UML)	All items of the report well done. A thorough report.	One or two items of the report could be improved or are not very accurate but in general very well done	Three items of the report could be improved or are not very accurate. But the others are ok	All items of the report have something missing or are not accurate.	Very poor report of no report at all	3

• Post-Morten analysis						
Presentation	Presentation well done. Clear summary of everything. Demo working with no glitches that shows all the flow from creating a course to a student's knowing he/she was accepted as a TA.	Presentation could be better Or the summary was not good enough The demo had glitches	Both things are not great, but ok	Very poor presentation Or very bad summary Or too many glitches or problems on the demo		2
GitHub	Everyone on the team contributing	Almost everyone	Half of the team	Not everyone at all		1
Deployed	Yes	-	-	-	No	1

RUBRIC – Delivery 7

Item	**100%**	**70%**	**40%**	**20%**	**0%**	**Max Points**
Functional Requirements	All done and working	Almost all done and working	Around half of it done	A bit done	Nothing done	2
Non- Functional Requirements	All done and working	Almost all done and working	Around half of it done	A bit done	Nothing done	1
Report (updated according to what else was done) • Summary of compliance with the requirements: which requirements are satisfied / partially satisfied / not satisfied • Description of the methodology followed by the team: tools used, frequency of meetings, work planning, etc. • Description of your data model (UML class diagram) • Interaction Overview Diagram (UML) • Post-Morten analysis	All items of the report well done. A thorough report.	One or two items of the report could be improved or are not very accurate but in general very well done	Three items of the report could be improved or are not very accurate. But the others are ok	All items of the report have something missing or are not accurate.	Very poor report of no report at all	3

Presentation	Demo working with no glitches that shows all the flow from creating a course to a student's knowing he/she was accepted as a TA.	The demo had a few glitches	Demo with glitches	Too many glitches or problems on the demo		2
GitHub	Everyone on the team contributing	Almost everyone	Half of the team	Not everyone at all		1
Deployed	Yes	-	-	-	No	1