JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2019.Fall

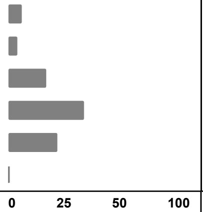
Report: OOSE

Response Rate: 84/85 (98.82 %)

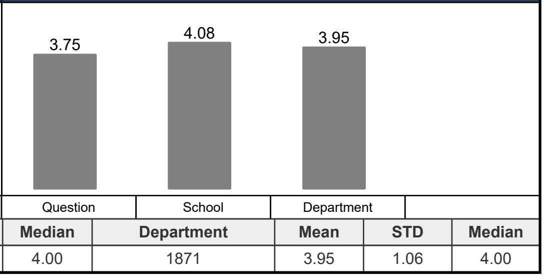
1 - The overall quality of this course is:

Response Option	Weight	Frequency	Percent
Poor	(1)	6	7.23%
Weak	(2)	4	4.82%
Satisfactory	(3)	17	20.48%
Good	(4)	34	40.96%
Excellent	(5)	22	26.51%
N/A	(0)	0	0.00%

Percent Responses



Means

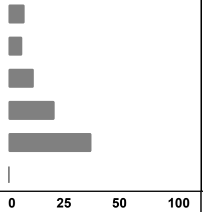


Response Rate	Mean	STD	Median	School	Mean	STD	Median	Department	Mean	STD	Median
83/85 (97.65%)	3.75	1.12	4.00	11005	4.08	0.99	4.00	1871	3.95	1.06	4.00

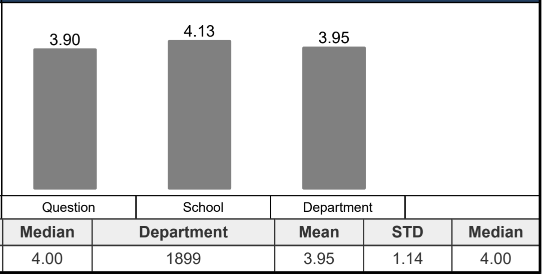
2 - The instructor's teaching effectiveness is:

Response Option	Weight	Frequency	Percent
Poor	(1)	7	8.75%
Weak	(2)	6	7.50%
Satisfactory	(3)	11	13.75%
Good	(4)	20	25.00%
Excellent	(5)	36	45.00%
N/A	(0)	0	0.00%

Percent Responses



Means

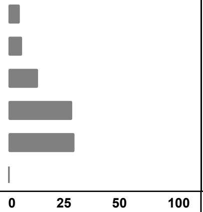


Response Rate	Mean	STD	Median	School	Mean	STD	Median	Department	Mean	STD	Median
80/85 (94.12%)	3.90	1.30	4.00	12078	4.13	1.03	4.00	1899	3.95	1.14	4.00

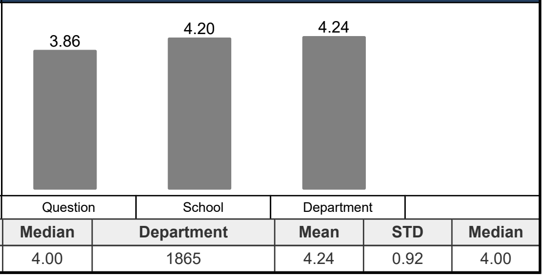
3 - The intellectual challenge of this course is:

Response Option	Weight	Frequency	Percent
Poor	(1)	5	6.17%
Weak	(2)	6	7.41%
Satisfactory	(3)	13	16.05%
Good	(4)	28	34.57%
Excellent	(5)	29	35.80%
N/A	(0)	0	0.00%

Percent Responses



Means

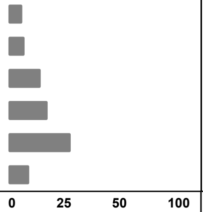


Response Rate	Mean	STD	Median	School	Mean	STD	Median	Department	Mean	STD	Median
81/85 (95.29%)	3.86	1.17	4.00	10929	4.20	0.88	4.00	1865	4.24	0.92	4.00

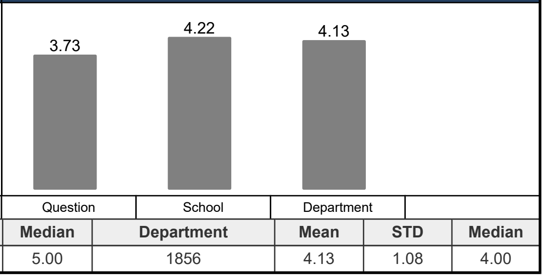
4 - The teaching assistant for this course is:

Response Option	Weight	Frequency	Percent
Poor	(1)	6	7.50%
Weak	(2)	7	8.75%
Satisfactory	(3)	14	17.50%
Good	(4)	17	21.25%
Excellent	(5)	27	33.75%
N/A	(0)	9	11.25%

Percent Responses



Means



Response Rate	Mean	STD	Median	School	Mean	STD	Median	Department	Mean	STD	Median
80/85 (94.12%)	3.73	1.30	4.00	10896	4.22	1.00	5.00	1856	4.13	1.08	4.00

5 - Please enter the name of the TA you evaluated in question 4:

Response Rate	52/85 (61.18%)

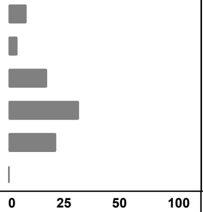
- Adeshola Lawal
- All
- Anand
- Anuraag
- Anuraag Baishya
- Anuraag Baishya
- Anuraag Baishya
- Anuraag Baishya
- Anuraag Baishya
- Anuraag Baishya
- Brandom Lim
- I'm not sure, but the TAs need to be told that they're not managers of a team of engineers. They're TAs. They're supposed to help, and teach, rather than be obnoxious and demanding.
- Kalyani
- Kalyani Pawar
- Liza
- Liza
- Liza
- Liza
- Liza
- Muskaan
- Muskaan
- Muskaan Kalra
- Muskaan Kalra
- N/A
- Paranv
- Pranav
- Pranav
- Pranav
- Pranav Shirke
- Pranav Shirke
- Shola
- Shola
- Shola
- Shola
- Shola Lawal
- Shreyas
- Shreyas
- Shreyas
- Shreyas
- Shreyas
- Shreyas
- SHREYAS
- Shreyas Aiyar
- Shreyas Ayar
- Stephan Kemper
- Stephen Kemper (Skemper)
- Yash
- yash
- Yash Kumar Lai
- Yash Kumar Lal
- Yash Kumar Lal

- Yash Kumar Lal

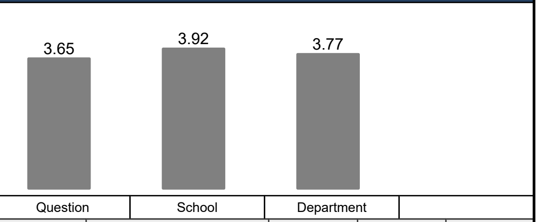
6 - Feedback on my work for this course is useful:

Response Option	Weight	Frequency	Percent
Disagree strongly	(1)	8	9.88%
Disagree somewhat	(2)	4	4.94%
Neither agree nor disagree	(3)	17	20.99%
Agree somewhat	(4)	31	38.27%
Agree strongly	(5)	21	25.93%
N/A	(0)	0	0.00%

Percent Responses



Means

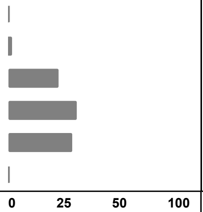


Response Rate	Mean	STD	Median	School	Mean	STD	Median	Department	Mean	STD	Median
81/85 (95.29%)	3.65	1.21	4.00	10874	3.92	1.07	4.00	1843	3.77	1.12	4.00

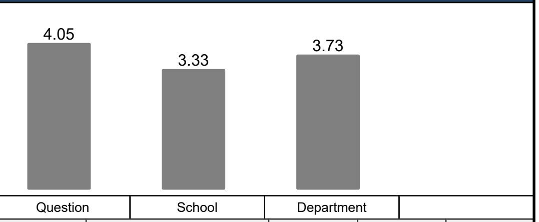
7 - Compared to other Hopkins courses at this level, the workload for this course is:

Response Option	Weight	Frequency	Percent
Much lighter	(1)	0	0.00%
Somewhat lighter	(2)	1	1.23%
Typical	(3)	22	27.16%
Somewhat heavier	(4)	30	37.04%
Much heavier	(5)	28	34.57%
N/A	(0)	0	0.00%

Percent Responses



Means



Response Rate	Mean	STD	Median	School	Mean	STD	Median	Department	Mean	STD	Median
81/85 (95.29%)	4.05	0.82	4.00	10886	3.33	1.02	3.00	1847	3.73	0.95	4.00

8 - What are the best aspects of this course?	
Response Rate	60/85 (70.59%)

- A well organized course covering plenty of interesting topics.
- Ability to work on an extended team project, this course is as close to "real-life" software development as any course at Hopkins. Professor asks for and generally seems to incorporate feedback well.
- Being able to work on a semester long project with a group of friends.
- Building an app throughout the entire year
- Building your own project and seeing it work.
- Choosing your own project
- Complete control on the group project. The TA is really helpful during lab session.
- Exposure to lots of tools.
- Forced you to learn new technologies
- getting to create an app
- good course organization
- Good introduction to many aspects of software engineering
- Hands on development of a project all throughout the semester. One class a week was dedicated to working on the project.
- Hands on I guess?
- Hands-on experience on Developement
- He recorded his lectures
- I learned a lot with the project (probably one of the classes I learned the most). Professor Facchinetti was definitely very dedicated to this course and wanted to make it as good as possible. He was an amazing resource for help, I always felt he had a deep understanding of what he was talking about, and he could even help with things he wasn't necessarily familiar with.
- I like the wide range of software engineering topics that are covered
- Independent project
- It involves comprehensive aspects of software engineering.
- It teaches us a lot about softare engineering. I learned a lot.
- It's a great courses.
- Leandro is a STAR, give him a raise PLEASE
- Leandro is most likely the best professor I've had at Hopkins. It's clear he is deeply passionate about teaching and computer science, which shows through both in his creative assignments and lectures. Every time I speak with him I feel I learn something new about CS as a whole, and is an inspiring figure.
- Leandro is very passionate about the material. I just think the execution was poor. We definitely should've spent more time learning material. Homeworks also need to be completely revamped. There needs to be more intro to javascript in the course.
- Leandro really tried to make sure that everyone understood the concepts fully.
- Learn how to build a web application from scratch
- Learn useful topics for software engineering. Leandro is enthusiastic and seems to care.
- Learning a wide variety on the subject of OOSE. Working in teams to create a complete, working product. The professor is very passionate about the topic.
- Learning about the high level design that goes into software engineering. Useful and practical, and working on group project develops skills
- Learning full-stack development is an essential skill and this is the only course at Hopkins that provides that opportunity. Working in a group was also very useful, especially for mastering git.
- Learning new technology
- Making our own app
- Nothing
- Project was fun.
- Project, Practical nature of the course
- Putting into practice a lot of tools that I've seen are needed in software engineering internships/jobs etc. Also the instructor is super passionate and he is willing to help you past his office hours.
- semester long Project, open to do whatever you want
- Super passionate teacher, most relevant CS course offered, actually learn coding practices you will use in real life
- The beast aspects of this course were the ability to create a useless, meaningful application that hopefully by the end of the project, has the potential to become a real application. You will likely learn a lot in the course if you aren't familiar with the tools before and will gain a lot of experience that will directly translate to your ability to work in a software engineering job.
- The best aspect of this course is the fact that we learn industry skills that are not taught elsewhere in the CS department, namely JavaScript, React, and other aspects of web development. The instructor genuinely cares about this course and makes every effort to solicit feedback to improve this course.
- The breadth of material covered and the assignments requiring hands on implementation that requires subject knowledge
- The course instructor is excellent.
- the group project! Very fun. Our TA was also an angel.
- The group project.
- The lectures are very helpful- they are informative and the instructor makes each concept clear. The class provides a good introduction to most tools in the toolbox and provides an in depth look on different design patterns and principles of software engineering. The videos taken for each lecture are also a great resource.

- The professor genuinely cared, but that didn't necessarily translate to effective teaching. It could have been great. He has the heart, but perhaps he just needs to teach more in order to practice his skills. The actual project was by far the best part. It was a fun idea.
- The project is fun and a really good experience
- The semester-long group project and the TA who helped us with our project and always encouraged us
- This course offers students a unique opportunity to design and develop a coding project in a somewhat structured setting with other students. I learned a lot about frontend software development, which isn't taught by many other courses at JHU.
- This is the first course in cs that I feel like has a lot of external application that I will need after I graduate. I really appreciated learning about the toolbox and being exposed to everything the course offers.
- Using skills and technology that is the industry standard
- Valuable course topics for real world software engineering jobs
- Videos of todoose, instructor's personality, coherence of the course
- We learned a lot of new tools that I know are very practical and I need to know for my CS career.
- work on self-designed projects with choice of tools and environments to use OOP
- Work on your own app
- Working on an interesting project of our choosing over the course of the semester
- Working on group projects and the professor's enthusiasm.
- You are teaming up to complete a project. Learned a lot & fun

9 - What are the worst aspects of this course?

Response Rate	63/85 (74.12%)

- A large portion of the course was heavily focused on individual assignments, instead of the group project. During the first 10 weeks of the course, when we had individual assignments, I had a difficult time focusing on both the individual assignments and the group project, because each took a considerable amount of time to complete. Additionally, the course expects students to learn about advanced applications of languages like JavaScript, React, and JSX without first providing students with an introduction to how these languages work. This left me and many of my group members very confused at first, and forced us to learn about many of these concepts largely on our own. I feel like I now know how these languages work, but I wish I had more foundational knowledge about each language so that I could more thoroughly understand them.
- Assignments could be easier on average since the midterm would test the concepts.
- Bad group chemistry
- can be a lot of work
- Everything I learned I had to learn on my own it felt like.
- Heavy workload with unpredictable spikes and unclear evaluation criteria
- Homework was not.
- Homeworks were extremely hard and required you to spend at least 20 hours outside of class in addition to group assignments.
- I am confident that the instructor spends 24-7 of his time on this course; but he has no right to expect the same from us. The workload is unbearable; the due dates are criminal; the TA's are divided between HW and projects and are therefore unreliable and difficult to find; the instructor is neither engaging nor knows not how to write decent, feasible assignments OR lecture notes.This class was diluted by meaningless assignments created in vacuum.
- I didn't feel like the material was well taught and I felt like we need more support throughout each week that the TAs just weren't able to give. Also there was an incredible amount of work. It was really just unfair.
- I think it's taught a bit too specifically to the tools that are used. This isn't bad, since many of these tools are common, but for example this course does not really prepare you for object oriented software engineering projects that are not written in java / react.
- If you get stuck with a bad group this class will be really hard
- Individual assignments eat up time that could've been better spent on the team projects. Some assignments repeatedly emphasize concepts to the point that they become busywork.
- individual assignments often interfere with group project scheduling concerns
- It is very difficult to keep up with work on the weeks when both assignments and project iterations are due. The learning curve at the beginning of the course is extremely steep for someone who is unfamiliar with most of the tools in the class toolbox. There is a lot of learning that must be done outside of the course in addition to the assignments and projects that are already due.
- Lack of direction. I get that that's the point, but a lot of times I just felt completely lost, especially in the group projects.
- Lectures in class could have covered more content
- Lectures were fun but not very helpful on the homework. We really struggled through homework because we just weren't taught how to do them which felt super unnecessary. I feel like everything I learned I pretty much taught myself.
- Material isn't very interesting and not really related to the project; many important aspects of software development not taught; material is not taught in a way that makes it easy to retain
- n/a
- needf a motivated group to keep on track with the project.
- None
- Nothing actually, I loved this course!
- Often the weekly assignments would consume a lot of time paired with the project iterations. Lectures were also not as useful for completing assignments/designing your own project.
- Pointless assignments
- poorly structured homework assignments. Quiz that is vaugely discussed and way too long. Unclear expectations. Multiple assignments and iterations due on the same day.
- Professor could be very condescending and oftentimes the way assignments were written would be completely at odds with how they were graded (e.g. an assignment would say: "Do not include Getters and Setters" and on the feedback for the assignment points would be taken off for not including getters and setters)
- Skemper was a very annoying TA.
- some individual is way more heavier that could be less in the future
- Sometimes I wasn't sure what the lectures and HW's had to do with the project. Also too many homeworks.
- Sometimes tedious work
- Strange, incoherent lectures
- The assignments are creative but sometimes vague.
- The course load was very heavy. We often had multiple large assignments due at the same time. The homeworks were often vague—and not in a way that encourages critical thinking—yet they demanded specific answers.
- The depth is not all that good
- The expectations for the quiz/final were unclear.
- The first half of the homeworks seemed useless and jumped too quickly into very foreign material . The "quiz" was supposed to take "less than 45 minutes" yet every person in the class stayed for a full 1.5 hours to complete an 8 page written essay final exam
- The grade may not be objective enough.
- The homework assignments were often vague or tedious and I did not get much out of them
- The homework description is vague sometimes, and takes a lot of time and effort. The lecture content is very interesting, but the professor teaching style is not engaging sometimes. The quiz is very difficult.
- The homework was tedious, boring, time-consuming, and poorly designed.
- The homeworks felt very pointless at times, and they were also pretty vague. Grading felt a little subjective sometimes. Since the projects can vary so much, I didn't feel the TAs really helped too much when I had questions, since they also didn't really understand what some of the homeworks asked. The quiz was dumb.

- The instruction of the group project is really vague. In times we are notified the requirements of each phace/iteration of our project two days before the deadline, or even after the deadline.
- The nature of the group can vary, depending on who you're paired with. Also the individual assignments can be very hard to do combined with group work
- The TA for our project doesn't seem to understand much about games and doesn't try to adjust her point of view to do so.
- The teaching assistants for this course were absolutely terrible. They were extremely bad at conveying expectations and would go from "everything looks great" to "you're falling behind and not doing enough" in just one week. They had zero understanding of issues we were having and had no interest in actually helping us. This was the worst experience by a huge margin with teaching assistants in my four years at Hopkins. Additionally, there was way too much work expected of this class because on top of working on the course long group project, there were weekly assignments that just felt like busy work and hindered the amount of time we would work on the course project. I've never had a coding class that has had weekly assignments and it's obvious why it should never be done. They were generous with the late days but late days don't mean anything if you have another assignment less than a week later. Additionally, many of these assignments were never released more than a week ahead of when they were due so there was no opportunity to get ahead in the class.
- The unclear lectures and the pointless, extremely time-consuming weekly homeworks
- The weekly assignments - they're very vague and exhausting to complete especially since we have group project iterations due as well
- The weekly individual assignments. They take quite a lot of time and this time could be used towards our projects instead.
- The workload for this class was totally disproportionate - why were there assignments due weekly AND bi-weekly on the same day? The individual assignments often felt tedious and unhelpful.
- The workload is too high. I spend a lot of times on this course and our team meet 3 times a week but I still feel frastrated.
- The worst aspect of this course are the individual homework assignments. While the intent is that they reinforce the concepts we learn in lecture, in practice, they amount to little more than busywork that distract us from our final projects (which most people would rather work on). Furthermore, many of the TAs (including Yash, the head TA) don't have a strong grasp of React, HTTP requests, or other course concepts, making going to most office hours useless (the instructor's office hours are an exception).
- There is way too much work all at once. Individual and group assignments are due at the same time, and at points it was hard to manage. I also do not think we are prepared properly for the group project and that the project would be much more successful if we got more support throughout the process.
- Too many assignments which detracted from overall group project focuses. Sometimes misguided directions by TA/Advisor
- too much work and does not provide enough support for the front-end
- Too much work and new stuffs
- Too much workload. It surprises me that this course is a 3-credit given how much time we have to devote to both individual assignment and the group project.
- Too much workload..
- Topics are not covered deep enough
- Way too much work and assignments were way too vague. Some of them even had bugs. I think every other week of individual assignments would've been nice. However, we had both that and the group assignment and so barely had time to even do teh group assignment. The individual assignments literally took days of sifting through code of language we weren't even taught. Even teh TAs didn't know teh language Leandro used....
- Way too much work. It felt unrealistic expecting students to watch the 31-part video series along with the individual assignments and team projects. The individual assignments took a great deal of time, which took away from time that would have been spent on the team project.
- Weekly homeworks take away time from working on main project and are not always related.
- Workload is quite heavy.

Report: OOSE

Response Rate: 84/85 (98.82 %)

10 - What would most improve this class?

Response Rate	58/85 (68.24%)

- A introduction to React, like an online reference course, for those who are new to it.
- A practice quiz, and a further increased emphasis on the group projects.
- Actually teach how to do the homework in class.
- add a pre course on front-end development
- Alternating homework assignments and group project iterations.
- Better communication of the instructor, and lessen interactive session to put forward more material
- Choose to have students work on the project instead of the assignment. Decrease the frequency of the assignments.
- Clearer standards for work, especially presentations
- Cover some more "useful" topics in more depth
- Cut the number of individual assignment in half. Either load them up in the front of the class or make it an every other week kind of deal. If you still want to cover all the same topics, then eliminate the useless questions in each assignment and combine the two. Preferably, make it so that the individual assignments and the group assignments are not due on the same day. Also, find a better way of recruiting teaching assistants because the current ones are terrible. Aside from the head TA who was very kind, supportive, and helping, the remaining were all condescending, arrogant, and unhelpful.
- DO NOT HAVE GROUP AND INDIVIDUAL ASSIGNMENTS DUE ON THE SAME DAY
- Don't have homeworks and iterations due at the same time, it can be really overwhelming at times. Also, the TAs were not usually useful during QA sessions about the homework, either because they haven't yet looked at the homework beforehand or they also are not sure how to approach the homework.
- Exposure to more languages. Go deeper into object oriented code.
- Fewer topics covered in lecture and in more depth
- First, make sure that the TAs actually know the concepts that are covered in the course. Perhaps better vetting should be implemented to weed out applications who don't know anything. TAs don't need to know everything, but they should have knowledge in at least one area so that we can say (for example) "Mr. X is the React TA, while Ms. Y is the Javalin TA." In addition, assignments should be reworked. Assignments can be shortened in lieu of higher expectations on the final projects (eg. asking us if we used any design principles or patterns mentioned in lecture in our project).
- good
- Group projects and individual assignments due at different times. More material covered before jumping right into semester project.
- Have only one person leading the todoose video series, the presence of a second (Yash) is distracting and uninformative. Course grading expectations (such as what numerical course grade corresponds to what letter grade) are extremely unclear.
- Having less homework and making it just a group project.
- Having the individual assignments be much shorter, about 2 hours to complete, rather than the average of 4-6 hours.
- I would appreciate making the individual assignments less intensive so that students have a greater ability to focus on the group project, which is the backbone of this course. There are plenty of other computer science courses that give students weekly assignments; what makes this class unique is its focus on a collaborative, long-term project, which teaches skills that are incredibly useful for any software engineering career. By introducing time-intensive individual assignments, the course unnecessarily distracts students from the group project and makes the impact of this course weaker overall. Introduce lectures that provide students with more foundational knowledge of the new languages introduced in this course, such as JavaScript, React, and JSX.
- If you want to continue to have individual assignments and group iterations do not have them due the same week. So alternate weeks for due dates. Having both an individual assignment due and a group iteration is just not feasible with the Hopkins workload.
- Increasing conceptual content covered during class
- It is unclear what exactly is required for each project iteration; even a vague rubric would be helpful so that everyone knows how much work is expected for each iteration. It would also be nice to grade group members individually based on their contributions (some group members don't always do their work). The TAs seem a bit unfamiliar with some of the toolbox; it would be helpful if they could provide more support when groups hit a snag.
- Lectures could be better to help in working on projects.
- Less homeworks since were also working on the project at the same time.
- Less homeworks, clearer lectures that involve less class participation and more direct lecturing
- Less individual assignment
- less individual assignments
- Less individual assignments more focus on the project
- Less individual assignments, more focus on group project
- Less individual assignments. Perhaps biweekly instead of weekly.
- Lighter individual assignments.
- Make the workload lighter.
- More consistent checks during the semester to make sure every member is contributing
- More explanation of material
- More focus on how to build scalable software, less focus on random tools/technologies
- More knowledge during lecture & less lab time
- More manageable homework’s and fewer concurrent assignments.
- More resources to read and sharpen the skills required to work on projects
- n/a
- No homework on weeks that iterations are due. Need to walk students through implementation aspect before doing design elements (learning JavaScript, server etc)

Report: OOSE

Response Rate: 84/85 (98.82 %)

- No lab sessions and more lectures, especially ones teaching more specific things related to software development rather than an overview of general design principles. Things such as version control and the development lifecycle, different types of stacks, different technologies and hardware, integrating different parts of a project, etc. Also if we had more guidance choosing a project so that we could have a better sense of what type or size of project is reasonable, I think the projects would be more interesting and we would be more invested in it.
- Not having homework when iterations are due, not having a quiz on the same day as an iteration is due, making a quiz that's not 11 pages long for a time frame of 45 minutes, giving smaller but more meaningful assignments.
- Not having homeworks due on the same day as iterations. Shorter quiz. Better homework instructions
- nothing
- Perhaps match project TAs to the project students are working on.
- Picking a different group
- Reduce work load
- Reduce workload and difficulty of the homework, and give more time to focus on the group project.
- reducing the number of homework assignments I also wish the course focused less on specific frameworks/libraries like React or Javalin as opposed to more fundamental software engineering principles and systems.
- Reevaluate the workload maybe
- Showing more example code from more sources rather than one larger project to succinctly express design patterns/principles
- Significantly less workload, a more knowledgeable instructor--preferably with a PhD--and a useful example app (TODOOSE is worthless, unhelpful, and cumbersome) that we design together in class. The videos released for making the app are risibly ineffective at teaching app design. A more engaging approach would be for the class to decide on and build an app together (start to finish) over the semester rather than the instructor selfishly creating his own app and assigning TEN extremely esoteric assignments.
- Simpler project but with two design phases (1. develop application, 2. ops/maintain)
- Smaller groups, or smaller individual projects on top of the large group project. These groups were too big, unecessarily so. Most of the time it felt like our group of 5 was doing the half the work that one of us could do on our own
- The grading system and the whole thing about the lab/group project.
- We could make a better group project if we could devote all our time to it.

Report: OOSE

Response Rate: 84/85 (98.82 %)

11 - What should prospective students know about this course before enrolling? (You may comment on any aspect of this course such as assumed background, readings, grading systems, and so on.)

Response Rate	52/85 (61.18%)

- A lot of work but you'll learn a lot. A must take for software engineers
- Be prepared to be self-organized in the team project. And it would be so much better if you enroll with a group in mind.
- Be prepared to teach yourself everything
- Be ready to commit about 15 hours a week to this class, if not more. This class should be about 6 credits so don't let the 3 credits fool you. Also, be ready to learn a LOT on your own. If you don't have any experience with javascript, SQL, HTML, be ready to spend countless hours googling stuff and watching videos on your own time because they will NOT teach it to you. They even repeating told us in class that we were expected to learn and figure out things on our own. Also, make sure you find a group to work with before the class starts and not during the class. Who you work with and how compatible you guys are will directly relate to how well the project turns out and how much of a pain it will be at times.
- Be ware that this course will take a lot of time if you want the perfect. The course is amazing and the instructer is so good at organizing materials and lectures!
- Be warned, a lot of work goes into this class.
- depending on your group/project, this may be a lot of work or not that much
- Do not take this class with Leandro Facchinetti. He has no teaching abilities or respect for students' time.
- experience with web or mobile development beforehand will go a long way
- Find some good teammates
- good class to take.
- Half of the class was amazing, and even for that the class is worth taking if you have friends to work on the project with.
- Have a group when you go in. Its hard to form a group with randos.
- Honestly you're much better off if you have a background in javascript / frontend stuff. He assumes you kinda know what you're doing even though it is definitely not a prerequisite.
- I would recommend having some familiarity with web development before joining; otherwise, the class may be more difficult/time consuming.
- If it is another instructor, it might be fine, but this course literally did not actually teach me anything I just had to self learn everything which I guess is like software engineering but then what's the point of having a course like this.
- If you're not expert in java script, fullstack development, and don't have too much time to put on this course, then you are in trouble...
- It's a chill class. It will probably be very useful to you if you have never written full-stack code before.
- Its an immense amount of work - requires you to spend 20-30 hours a week learning
- Java,HTML, CSS, MySQL
- Know web basics: HTML, CSS, js, react, etc.
- Know your group before you go into the class
- Make sure to take it with people you want to work with!
- Make sure you choose good groupmates because you will be with them for the entire semester.
- More work than previous years seem to have been
- Must tailor your choice of technologies to the LCD of knowledge within group to work effectively
- n/a
- NA
- no background needed except coding experience
- Pick a good group
- Prepare to learn valuable software engineering concepts, and to apply them. You will have to devote a lot of time to your final project to end up with something that you are happy with.
- Prepare to work on individual assignments every week and group assignments every two weeks
- Prospective students should know that most work is done outside of lecture (which only met once per week - this may change under different instructors). The project will take a lot of time, so make sure you have good teammates. Finally, in this iteration of the course there was a "quiz" that was in actuality a final exam, so make sure you're prepared for that if it's mentioned in the syllabus.
- Recommend for people with previous experience in web app or mobile app development.
- The course load is extremely heavy!
- The project is great but other than that the class isn't very compelling. If you want the experience of working collaboratively on a software design project then it's a good class but otherwise it's not particularly helpful.
- The workload is really big.
- There was a lot of work and if the course load doesn't change, make sure you don't have a lot of other heavy loaded classes.
- This class is a lot of work. Be prepared to spend a lot of hours learning how to use new tools and programming.
- This course is very valuable if you want to do front-end development or web-dev, but otherwise there is no reason to take this course.
- This course teaches you software engineering skills that are very practical, but you will have to teach yourself a lot and will be busy with a lot of different work.
- This is a good course to take if you know multiple friends who will also be taking the course. It is really fun to be able to complete a semester long project.
- This is a very useful class for learning about software engineering and group work. This is great preparation for working in a group with a deadline several months from now. The lecturer is really passionate about his students learning and I would definitely recommend this course for those interested in software engineering.
- This is amazing course to be able to understand and build a web application and covers wide range of topics required for a software engineer role in the industry
- Very good project based class to get to know more about group based coding projects and software engineering. You also get to learn about functional programming and security.
- Will learn lots of new skills

- Workload is quite heavy.
- would definitely require some programming experience beforehand
- You need to have basic understanding towards web application, front end or back end, know how to build framework
- You should have a fair idea of the different applications used, and be effective to master them quickly since you'll require to work on them throughout the course.
- you should have some grasp of javascript before taking this course.
- You'd better be proficient in Java, Javacript, backend or frontend. If you know nothing about these, you will have a very unhealthy semester.