CS 373 Spring 2024: Neha Desaraju

Blog #14: Final Entry | 04.29.2024

How well do you think the course conveyed those takeaways?

I think this course conveyed overall design patterns very well. I wish we had gotten to see more real-life use cases for these patterns, as I felt that some of the examples given in class and in the papers were very contrived. I wish there were more discussions about the concepts as well.

Were there any other particular takeaways for you?

Designing before you start writing and organizing your tasks is a very underrated skill and habit to get into. The projects really taught me about working with a team on an undirected project, and it helps to flowmap dependencies and tasks and design before we start coding so we don't have to do too much refactoring or fixing. Communicating design and software concepts is also an important skill.

What required tool did you not know and now find very useful?

Python testing tools such as unittest and pytest. They very quickly spin up tests in a clear and readable way!

What's the most helpful Web dev tool your group used that was not required?

Svelte! React was recommended, but I really love Svelte and its ability to use states and hooks natively compared to React. It feels like using raw HTML, CSS, and JS with superpowers. Components are readable and standard.

How did you feel about your group having to self-teach many technologies?

This was the part of the class I disliked the most. It really puts those who are familiar with the technologies at a disadvantage if the other group members don't know it — they must choose to either teach the other members or do it all themselves — and those who aren't also suffer because they don't learn well when this happens. I don't think it teaches self-learning as much as it seems like it should.

In the end, how much did you learn relative to other UT CS classes?

Not much; I learned many of these concepts previously (e.g. in an internship) and through other projects I'm more passionate about. Overall, this was a good exercise in working with a group, but I did not learn much computer science.

The daily quizzes were surprisingly useful in ensuring attendance (haha) and making sure I clearly pick up the concepts in class. I think the real benefit is in the spaced-repetition, since you have to recall what you learned in the class before in order to make a good score.

Having a customer and developer role made sense, but they should have just been in pairs (e.g. 15 is the developer for 16 and vice versa) instead of in a chain loop (e.g. 13 develops 14 develops 15, etc.). This would have made it less confusing.

CATME peer reviews were a great way to communicate feedback to a team member in a way that is anonymous. I really looked forward to getting feedback from my teammates as well because it let me know how they felt about my work.

I think the user stories aspect of the projects, however, was not as helpful. I see the value, but I think they became more of a nuisance (yet another random task on the rubric) than a lesson. The users tended to not know enough about the topic or project to really ask for something that is doable or made sense in the context of the topic.

The GitLab issues as rubrics was a great feature, though! I wish more CS classes did this, as it made it really easy to keep track of everything that was required.