*** PROVISIONAL REPORT ***

UNIVERSITY OF TEXAS AT AUSTIN — COURSE-INSTRUCTOR SURVEY — Fall 2017 DEPARTMENT COPY
Downing, Glenn P — C S373 — 51740 — SOFTWARE ENGINEERING — Enrollment = 58
E100 EXPANDED — Surveys Returned = 58

NUMBER CHOOSING EACH RESPONSE

	Str Disag	Disagree	Neutral	Agree	Str Agree	NO. REPLIES THIS ITEM	AVG.
1 COURSE OBJECTIVES DEFINED-EXPLAINED	0	2	3	24	29	58	4.4
2 INSTRUCTOR PREPARED	0	0	1	7	50	58	4.8
3 COMMUNICATED INFORMATION EFFECTIVELY	0	2	3	19	34	58	4.5
4 STUDENTS ENCOURAGED-ACTIVE ROLE	0	0	1	14	43	58	4.7
5 INSTRUCTOR AVAILABILITY	0	1	4	16	37	58	4.5
6 COURSE WELL-ORGANIZED	0	3	4	24	26	57	4.3
7 STUDENT FREEDOM OF EXPRESSION	0	2	2	17	37	58	4.5
8 CLASS PARTICIPATION ENCOURAGED	0	0	2	6	50	58	4.8
9 ENGAGING INSTRUCTION	0	1	7	23	27	58	4.3
10 INST. HAD THOROUGH KNOWLEDGE OF SUBJECT	0	1	4	15	38	58	4.6
11 INSTRUCTOR EXPLANATIONS CLEAR	0	0	3	24	31	58	4.5
12 GENUINELY INTERESTED IN TEACHING COURSE	0	0	1	13	44	58	4.7
13 HELPFUL COURSE MATERIALS	0	4	17	18	19	58	3.9
14 ADEQUATE INSTRUCTIONS FOR ASSIGNMENTS	0	9	12	22	15	58	3.7
15 ASSIGNMENTS AND TESTS RETURNED PROMPTLY	0	1	9	22	26	58	4.3
16 ASSIGNMENTS USUALLY WORTHWHILE	0	2	3	24	28	57	4.4
17 STUDENT PERFORMANCE EVALUATED FAIRLY	0	3	12	21	22	58	4.1
18 STUDENT PERCEPTION OF AMOUNT LEARNED	0	3	4	19	32	58	4.4

	Vry Unsat	Unsat	Satisfact	Very Good	Excellent	NO. REPLIES THIS ITEM	AVG.
19 OVERALL INSTRUCTOR RATING	0	2	3	22	31	58	4.4
20 OVERALL COURSE RATING	0	3	9	22	24	58	4.2

	Excessive	High	Right	Light	Insuff	NO. REPLIES THIS ITEM
21 STUDENT RATING OF COURSE WORKLOAD	1	31	25	0	0	57

	Less 2.00	2.00-2.49	2.50-2.99	3.00-3.49	3.50-4.00	NO. REPLIES THIS ITEM
22 OVERALL UT GRADE POINT AVERAGE	0	1	3	19	35	58

	__A__	__B__	__C__	__D__	__F	NO. REPLIES THIS ITEM
23 PROBABLE COURSE GRADE	28	20	10	0	0	58

For the computation of averages, values were assigned on a 5-point scale so that the most favorable response was assigned a value of 5 and the least favorable response was assigned a value of 1.

COMMENTS:
Total Number of Comments: 33

---

1. Simon's blog post on Test 1. I feel the time allocated was insufficient for such a novel test. I feel the amount of people present at the end of the exam supports this claim.   See Jared Jones' blog for week 11 which I feel sums up the sql portion of the class.   This class is all over the place with tools, it felt like using a phone with lots of bloatware. Lots of stuff forced on me that I don't want. - Travis/unittest : I still have no clue how to truly use these in respect to our project, you go over it in respect to Collatz and then ... nothing. You wouldn't even answer questions properly on Piazza about what to test. "whatever you think is best" is not an answer - Plan It Poker : wtf. Otherwise, I liked cold-calling/quizzes/format.

---

2. I wish more was covered in lecture in regards to how to actually develop (directory structuring, elementary introductions to Flask, React, etc.) The only reason our team survived is that one of our members already happened to know how to do front-end, and absolutely carried the team in that regard. Everything relating to tools and methods was helpful (although it felt like I was taking OOP at times), but the workload was incredibly high - I should've taken it a different semester. I just wish Ilearned more about front-end.

---

3. I really like the course and I learned a lot. I just don't think that our midterm graded reflect all the work that we put into the semester long project. I also think that the hackerrank portion of the exam was long and counted too much towards our overall exam grade. I think that hackerrank should have counted a quarter of the exam and not half of the exam. In the end, I really liked working on the project since a lot of the tools that we used are really useful in the real world.

---

4. One of my most valuable takeaways in this class were the wide variety of tools I learned how to use by working on the group projects. There were many components to the projects and I found them all to be very useful: the database (SQL), API (flask/SQLAlchemy/python), front-end (react), unit tests (Postman, Selenium). The specs for each project could have been better articulated on the class website. Also, this class can be a bit stressful based on how dependable your group is. I hope this is anonymous because I feel like not all of my group members pulled their weight throughout the semester, and for that I feel like not everyone deserves the same grade. Maybe in future semesters there could be a peer evaluation.

---

5. Although we covered a lot of tools in class, I believe that we went through them "briefly" and not too much in depth. I thoroughly enjoyed the lectures and everything was explained clearly and effectively. This is one of my favorite classes at UT!

---

6. Professor Downing's class is definitely one with a heavy workload that you have to keep up with, but I do feel like I learned a lot in the class.

---

7. I've taken another course of Downing's and I think he is a great instructor. My only qualms with the course are 1) I think the quizzes should perhaps be a little more fair...sometimes they can be a little tough. I think the fact that they actually impact our grade beyond participation is unnecessary. I can understand utilizing them for practice and attendance, but to weigh them based on results diminishes the ability to feel comfortable learning and trying. It instead becomes a more strict testof knowledge and doesn't result in a positive learning experience in my opinion. 2) I think when utilizing new technologies like GCP or Postman, it is important to be more flexible in how those tools affect the grade students receive on projects.

---

8. Working on the projects was very useful. Gives a taste of working in industry. Very practical class.

---

9. I don't like how the project specs often changed due to something not being specified/unclear. I don't like having to learn tools that aren't commonly used in industry or anywhere else (GitPitch, PlanItPoker) when better alternatives are available. Graders need to give better feedback for the projects. Also it's hard to work on your project while you're waiting for the graders to finish grading. I would have preferred a talk on setting up local dev instead of the react talk which I thought was not useful. I didn't use Docker at all, some more explanation as to the usefulness of Docker would have been appreciated but I personally saw no reason to use it. Speakers were good.

---

10. A little more detail for PlanItPoker is needed for IDB 1, people lost points because their internship experience was different from what the graders expected for user stories. GCP is good, but has a steep learning curve. A detailed deployment lecture would be nice.

---

11. Very engaging and open to questions. Really wants you to learn the material. Not a big fan of the daily quizzes, puts unnecessary stress sometimes.

---

12. I know that professor Downing is usually very receptive in terms of student feedback. However, his reluctance to change GCP, one of the tools used in the class, was frustrating at times.

---

13. Overall, I think this was an excellent course, perhaps the best I've taken at UT. The group project was incredibly relevant and I learned a ton. I felt very unprepared for the test, however. I did excellent on the multiple choice section of the test, but I did very poorly on the HackerRank section. I felt as though I had practiced programming all of the topics discussed, but found myself unable to effectively answer the coding questions. Considering the projects rely so heavily on coding, and actually producing a functional full-stack application, I do not feel the coding section is necessary for the test. However, if I was going to do it again I would just request practice problems, similar in complexity and topic, to the exam.

---

14. Not a fan of GCP, but I see that next semester the class will be using AWS. I appreciate all the new technologies I was exposed, which were a learning curve. I highly value the guest lectures and hope you continue to do them in future semesters. Thechoice to pick our own groups is a good idea for future semesters. Thank you for sharing your knowledge with us. Definitely a hard class but mirrors the industry.

---

15. .THE TOOLS. I am thankful to have been introduced to so many tools, especially Python-specific tools like unittest. Now I have a more complete knowledge of what to use when testing my code, professional or otherwise. .COLLATZ. That was a nice introduction to using tools and applying the theory learned in class to an actual assignment. .GROUP PROJECTS. I think the GitBook criticism on Piazza was valid, namely that a team typically does not change their documentation tools midway. Otherwise, no complaints. .SPEAKERS. I believe the most important topic was the ethics talk. My opinion is that there needs to me more of that talk because, outside of this class, the only other ethics talk is in CS439 with professor Norman for 0.5hrs. Not enough.

---

16. Overall I love taking courses with Downing. This course, however, felt really disorganized at times surrounding the "team

project". I felt often times that the project was poorly planned out, portions felt clunky in an attempt to simulate the "real world" but in the end failed at that and made the course unnecessarily confusing. I think taking a better look at what it means to work on a sprint team today would be really valuable. My main complaint is that planning phases are really valuable, the first phase should have been focused on making the API and planning the structure of the project not wasting a ton of time on a bad front-end.

---

17. The class taught me a lot about the JS/Python/SQL web stack. I enjoyed learning about React, Flask, and SQL and using tools such as TravisCI, coverage, Postman, Trello, and Slack but I did not like GCP (complicated, expensive, many known bugs), GitBook, GitPitch (I'm not a fan of using markdown languages instead of WYSIWYG office applications), and Plan It Poker. I didn't learn a ton for Collatz and the suggested workflow that we followed for that first individual project did not really help for the following group projects. I think a light-weight, first week group project would be more helpful in completing IDB.

---

18. Professor Downing is very effective at conveying a great deal of knowledge, from Python intricacies to SQL. I believe a bit more meta-course information would make the course more meaningful. For example, a full-semester overview of the topics covered by day in the course and a brief introduction to each day's topic would give the course more motivation. Furthermore, I am not sure if the group project structure is more beneficial than detrimental, as the group structure allows individuals to silo into different specialties; perhaps a structure more like OOP with every student completing something in SQL, something in Python, something in React, etc. would give students more breadth.

---

19. I wish tools used were more reflective of current technologies. PlanItPoker seemed excessive.

---

20. This was one of the most engaging and inspiring courses I have taken so far at UT, and by far my favorite course in computer science. The project was overwhelming but effective in teaching me how to approach overwhelming problems. A great course.

---

21. Canvas was a good tool for quizzes and exams. It also made group members clearn. Docker was a bit confusing to use at first, but helpful in getting the correct environment for the first project. Git and GitHub have been the most used tools on theprojects, which is useful for version control. HackerRank is so much better than writing code in Canvas. I like the premade tests I can check against.   Collatz was a nice primer into using Python. GCP was interesting to learn at first, but then we delegated only one person to handling GCP. The rest of us don't know much about it.   Gitbook is nice for hosting the technical doc, but the UML was a pain. Trello was great for organization. Atlassian and Goldman Sachs were very interesting talks

---

22. The course was well taught. Lectures were engaging, especially being cold-called during instruction. Made me perform my readings prior to class. Learned a great deal about software engineering.

---

23. I had a couple of worries coming into course. On one end, I was extremely excited about taking SWE since I had a wonderful time taking OOP. However, I was caught off guard over the sheer amount of work needed for the project. It also is definitely much more front end intensive and React intensive. Groups play a major part in how the projects will play out, so it is important to find a responsive and cohesive group. Other than that, the tests and quizzes are fairly given and assessed. I recommend anyone interested in learning how to work on a full scale project with teammates to take this course, but also be aware of the workload because it can catch you off guard!

---

24. The class was a great intro into how to do software development (web dev in particular). It would have been nice to focus less on language specifics however, and more on the big picture of how to write good software / structure a project / deployment practices. Sorely missing from the Python lectures were how imports work / pythonic directory structure.   I think some dedicated lessons on React would be hugely beneficial. I really liked how even though we all did a similar project, we were able to choose the specific tools and implementations that we wanted e.g. we still had freedom.   GCP was somewhat convoluted - AWS or possibly Azure could be better solution

---

25. GCP was overly complicated for our project. Although it is probably extremely useful for projects of a larger scale, it caused a lot of headaches to many groups in the class. In the following semesters of this class, I would look into using Heroku instead of GCP to host our projects.

---

26. Overall, the class was informative and useful. Learning the intricacies of Python was a neat bonus, but the main learning experience for me was the group project, the tools, and the reading on refactoring. Having taken OOP before, there was a considerable amount of overlap, but the differing language and the vastly different project structure made this class worthwhile alongside OOP. We (like many) had issues with GCP, but many of the tools were new to me and ended up interesting. Postman and Selenium in particular I found useful for testing. Plan It Poker may make more sense in a traditional workplace setting, but we found it lacking. Having Hackerrank coding sections on the tests I found more valuable than pure MC (as when I took OOP).

---

27. Excellent class

---

28. I think the tools we used were neat and useful. Although, I think that we hardly got to use things like Travis CI, and autopep8, etc. since we only used those for Collatz. Perhaps a nice thing to maybe try and do in the future is trying to use continuous integration in some fashion for our IDB project. The group project is probably the best part of the class due to how much you learn just by developing a website basically from scratch all on your own. Although, I suggest using a platform otherthan GCP to save some headaches. The talks were interesting and were worthwhile to go to because you get a free quiz grade. Overall, I think the class was a worthwhile learning experience.

---

29. Everything about this course was fair except for the way the exams were done.

---

30. I liked how Professor Downing gave us some starting tools but allowed us the option of using alternatives. I did not like pylint much. When making my meta cache, it put every value on a new line, which resulted in a massive python file. Collatz seemed kind of random, but I liked it. It taught us some python basics and let us ease in to the true rigor of the course. I learned a lot about full stack development from IDB. Of the tools we used, I was not a fan of GCP, GitBook, and PlanItPoker. GCP had a steep learning curve that was unnecessary for the scope of this project. GitBook felt more restrictive than Google Docs. And then there's PlanItPoker. I liked the talks, and I appreciate Professor Downing going out of his way to arrange them.

---

31. As a professor and lecturer, I thought Prof. Downing was amazing. His lectures were very engaging and I always learned something new and random I had never previously encountered or thought of. Some of the things we learned were also really interesting and applicable to my own programming experience. The projects, on the other hand, were good but sometimes felt a bit too much to keep up with. The result is that we learned a lot in a quick amount of time, but the actual learning experience is heavily group-dependent. Still, I found them fun and worthwhile. The tests seem a bit arbitrary, and are probably my least favorite aspect of the class. I think they should be a bit more high-level and board in material coverage.

```
--------------------------------------------------------------------------------------------------------------------------------
32. The main problem I had with this class is that the projects and the course material are miles apart from each other. Maybe you
can have 15 or 20 minutes on Friday to go over any questions and maybe show a demo or two of a new tool we're supposed to use.
The instructions on the project are too loose for interpretation on such a large task. We needed more guidance on the project,
and also the final wasn't great. We didn't cover any test material in the last 4 classes we had. That is not a good way to
prepare your students for an exam, you need to be refreshing the material in class instead of talking about it once and never
returning to it.
--------------------------------------------------------------------------------------------------------------------------------
33. I found this class to be a very worthwhile experience and the projects were great. That being said, at times I wasn't certain on
all the requirements, nor on why they were there (for example, the additional 10 user stories per projects seemed moot).
--------------------------------------------------------------------------------------------------------------------------------
```