How well did material covered in the project (code and non-code) reflect the material in lecture... (r0_Lec)

The non-code reflected lots of the materials in the lecture. (r1_Lec)

It did not reflect at all. It felt like the project and lectures were 2 different courses altogether. (r2_Lec)

Material on the quizzes seemed too vague at parts, and had arguably different answers than what was first shown (r3_Lec)

I would say the disconnect is a lot. Especially in the sense that you cannot apply design patterns at the time they're taught, because you're almost done with the backend by that point. (r4_Lec)

At a high level I guess the project reflected what we learned in class. However there were lots of aspects of the project that I found difficult due to not learning it in class. (r5_Lec)

I think learning about design principles is useful, but realistically by the point we learned about them we had already finished our implementation and since deadlines are a time crunch there was no opportunity to go back and fix things. Learning about APIs was helpful, but apart from that I think most of what we did for the project was just figuring stuff out as we went and going to office hours for help - not directly related to course content, but I still found it useful overall. (r6_Lec)

The materials covered in the project feels irrelevant to the materials covered in lecture. (r7_Lec)

I think the project is not super related to what we have learned in class other than the promises unit. Basically everything in the project I had to learn myself, the lectures did not help 99% of the time. (r8_Lec)

not much i think (r9_Lec)

The stuff we did in the project really lined up with what we learned in lectures and on quizzes. It was like seeing theory in action. (r10_Lec)

I would say that they are extremely different in terms of coding. However, I would still say that they were quite useful as all of the dynamics of working in a team and on a large project such as all the code smells that come up were very useful. However certain things like design patterns we were never forced to implement. The non-code artifacts on the other hand aligned quite well with the lecture and quiz material. (r11_Lec)

lectures somewhat, didn't help for quizzes though (r12_Lec)

I think the non-code artifacts were very good reflections of the lecture material. The code itself is less reflecting, as I don't think my group ever deliberately implemented a design pattern in the code. I can not really think of any link between the project and quizzes. (r13_Lec)

fairly well - but I know the course team intentionally introduces certain topics after a checkpoint has passed. For example it was interesting to see how the Composite design pattern could have been used to the performQuery implementation or when we briefly discussed GUIs or even I believe the mutant testing lecture came after the c0 deadline which I understand the possible pedagogical rationale but while working on c0 I was still so confused how the grading worked and what mutants were but it became more clear after the lecture + practice during quizzes. (r14_Lec)

I would say it was mildly relevant. (r15_Lec)

Non-code portions had some overlap, but mostly only on user stories. For the coding portion, I did not feel as though there was much explicit overlap (r16_Lec)

Somewhat, project generally did not use the practices (r17_Lec)

I feel as if there was some overlap (i.e. REST APIs, testing etc). but I did feel somewhat of a disconnect between lecture materials and the project contents. (r18_Lec)

50-60% (r19_Lec)

Not too related (r20_Lec)

not really much overlap (r21_Lec)

Not well, quite different (r22_Lec)

Somewhat well. The lectures on testing and REST were pretty applicable and useful. Otherwise there wasn't much overlap between the project and lecture. (r23_Lec)

I dont think that if covered it at all. The issue was that we mostly learned things that could be helpful to late too integrate them in our work (r24_Lec)

Not very well. The time constraints related to the project make it difficult to apply learned material to. (r25_Lec)

Fairly good (r26_Lec)

Blackbox and whitebox testing was very useful material in lecture that was directly applied in the project. Learning about working in a team was also directly applicable to the project. Also, learning about code smells and debt was helpful in ensuring that my project was maintainable. (r27_Lec)

Barely connected, except a bit of the test stuff and the REST APIs part. The whole "mock scrum" thing was a bit overblown. No need for a weekly scrum meeting for a two-person team that does all implementation in 3 day period. (r28_Lec)

I covered quite well. (r29_Lec)

Honestly I did not really use anything discussed in the lectures intentionally on the project. The quizzes quickly left my memory after taking them (r30_Lec)

Mainly the non-code artifacts on doing the user stories was where I noticed it was the only connection. We were implicitly doing black bloc and glass box testing but I never connected it with the content in class. (r31_Lec)

Fairly relevantly especially the non-code portions. Although it's still possible to submit non-principled code as AutoTest doesn't care about that. (r32_Lec)

Not very, it was pretty separate with minimal overlap. (r33_Lec)

Decently well I'd say. But the project is obv way more fun, enjoyable and overall there is a lot more learning to take from there than material in lecture which you could honestly teach yourself with the videos and readings. (r34_Lec)

I feel like we didn't really get a chance to practice a lot of the theory. For example, we didn't have to refactor our code if it already didnt have any code smells, and we also didn't really have to implement any high/low level design decisions. (r35_Lec)

70%. Good habits and methods translated well for each checkpoint (r36_Lec)

I feel like it is not as a good reflection. All the code smell/refactor parts I wasn't fully doing it as the project tasks were big that i didn't have time to go over and check it. Also the grading portion of it was too little in the project. (r37_Lec)

Not very much except maybe user stories (r38_Lec)

Very well. Solid is very useful. Helped with refactoring a lot. (r39_Lec)

I personally think the project serves as a practical application of the theoretical concepts discussed in class, providing an opportunity to reinforce and deepen understanding through hands-on experience. The project enables me to practice in various key areas such as refactoring techniques, identifying and addressing code smells, implementing integration tests, and more. (r40_Lec)

I feel like there isn't a strong connection between the project (code part) and material in lecture. The non code part is somewhat related and materials in lecture did provide some help. (r41_Lec)

There were a lot of topics from lecture that were covered in the project, but didn't help in its implementation. I found that the course material were more about the overarching techniques, methods, strategies, etc. to software engineering. Yes, it did include a large portion of unit testing/ blackbox testing, etc. but strategies on how to implement them were harder. I also found that they weren't always in sync with each other. (r42_Lec)

Some of the material of the project was really covered during lecture as with asynchronicity coding and design pattern as many of the in class question related to the project. (r43_Lec)

Not really. (r44_Lec)

i found a lot of the design stuff was covered too late to make it relevant to the project (r45_Lec)

Did not, the lecture and project seemed completely detached. The one thing I saw that was the same across the project and lecture was the API endpoints lecture because we actually used it in C3. (r46_Lec)

quizzes seem totally irrelevant to the project, except the user stories part (r47_Lec)

i didn't like some material was taught after or during when we were supposed to do them on the project, like testing. (r48_Lec)

Most material taught in lecture were not related to the project/quizzes. (r49_Lec)

The project wasn't really related to lecture or quizzes (r50_Lec)

Really well! As more time went on, I realized how a lot of these topics related to the project. (r51_Lec)

mainly reflect the lecture material through non-code part(e.g. refactoring). I feel like for code part it's more about interpretation of specs. (r52_Lec)

The high-level aspects of the project (e.g. scrum) demonstrated the material in lecture/quizzes fairly well. I thought C0 reflected the material on testing well. For C1, since our implementation details were open-ended, I found that we did not necessarily adhere to design principles, since it wasn't really required. I thought about design principles from class a bit more during the start of C2 when we refactored our code, but otherwise, it wasn't really a focus. As for C3, the API implementation component complemented the lecture material fairly well, but since we didn't cover MV* material until the end of C3, all we did for the frontend was try to keep our code clean and understandable. (r53_Lec)

'- The NCAs reflected the lecture material well.\- It is implied the code artifacts should be done in such a way to encompass lecture material (e.g. BVA, ECP, design patterns), but it's very possible and sort of in the nature of the deadlines not to do these extra refactorings. (r54_Lec)

Tangentially. (r55_Lec)

The material covered in the project is not really that related to the material in lecture. The parts that were related, such as process, cohesion, and coupling, were not explicitly included ion the project specification , it was just left to the appreciation of the individual developer. (r56_Lec)

I think the checkpoint materials in the project lagged a little behind the material covered in lecture. I think it would make more sense to have checkpoint deadlines for each concept due one week after coverage in lecture, so we get a full chance to learn it conceptually before having to implement in in code (r57_Lec)

I thought the lectures covered the content needed for most of the project. (r58_Lec)

Relatively well. There wasn't much that was needed to complete the project that I didn't feel was sufficiently covered in lectures. However, we did occasionally learn things we needed for the project a bit later than we needed them. This didn't really affect me though, because I had worked with most of the concepts and technologies before. (r59_Lec)

The lecture material related more to the non-code artifacts than the code-artifacts. (r60_Lec)

I do not feel that, in general, the material from the course assessments had overlap in quizzes. They felt very disjoint. Non-code artifacts were probably the only part that did (like the user stories for example). (r61_Lec)

Not very well. Lectures and quizzes felt very theory-heavy, while the project was implementation-heavy. (r62_Lec)

Project material aligned closely with lectures and quizzes. Concepts like data structures, algorithms, and design patterns were directly applicable within the project, reinforcing the theoretical knowledge from class. (r63_Lec)

I believe the project, particularly the code aspect, may not be directly linked to the lecture or quiz. However, it is reasonable to place emphasis on them separately. As for the non-code part, it does reflect some key points covered in the lecture. Some aspects may appear overly formal and artificial, which contrasts with the typically more organic process during the co-op. (r65_Lec)

While the material vaguely aligned, it felt like the project involved a significant amount of learning not covered in lecture. (r66_Lec)

The correspondence between lecture and the project is pretty strong in my view. Async and promises are an important part of processing datasets. Testing is a massive part of C0 (and hopefully subsequent checkpoints). Refactoring is a huge part of C2. REST and APIs make an appearance in C3. I would say it's harder to find applications of the design lectures. By the time design comes up we are somewhat late in the game and have fewer opportunities to apply that material. This is perhaps unfortunate because design is perhaps the highlight of the lectures but it gets the least reinforcement from the project. Teamwork and process do tend to permeate the whole project however (at least in non-dysfunctional teams).\\I didn't see a whole lot of correspondence between the project and the quizzes. I saw a lot of correspondence between the project and the lectures and the quizzes and the lectures, but I didn't perceive the correspondence as being transitive. (r64_Lec)

The code portion wasn’t that well covered, overall software development techniques however pretty well (r67_Lec)

I feel like the project is generally disjoint from the lectures and quizzes in this course. The choice to apply the learned material is on the student. It isn't needed to perform well (r68_Lec)

project does touch on some lecture material but is pretty detached from course (r69_Lec)

The lectures and quizzes felt disconnected with the project. (r70_Lec)

I think the project material made pretty good use of a lot of the lecture material, including promises, design and testing principles, etc. I think SOLID and the design patterns were introduced a little too late though, since our group had a large part of our program already completed before running into refactoring issues in c2 because we hadn't introduced enough separation by following DIP. (r71_Lec)

project and quizzes were largely independent, many techniques talked about in class were not used for the development of our project (r72_Lec)

Sort of (r73_Lec)

I didn't see much of a connection. The content in the lecture was related, but it did not feel relevant to completing the project at all. (r74_Lec)

I think the code and non-code portions of the project reflected the material well. (r75_Lec)

I could not easily make a connection between the material we used in the project to the material covered in lectures and quizzes. I think it would have been more worthwhile to introduce design patterns earlier for example so that we could utilize that information when creating our project, rather than learning about it afterwards and then noticing violations in our code that could have been designed more effectively. (r76_Lec)

the project did not reflect very well on the material in lecture and on quizzes. it was mostly up to the students to utilize the max of their knowledge of the class material to the project (r77_Lec)

The coding part has a lot more than what was covered in lecture, but I felt the non-code part tied pretty closely with the lecture material. It helped me reflect on what techniques from the lecture were being used in my development and what was missing. (r78_Lec)

Quiz questions felt slightly disconnected from the material covered in lecture at times. (r79_Lec)

The material covered in the project didn't reflect the material in lecture and quizzes. The lecture and quizzes had a great amount of disparity since it was very, very, theoretical and was based on definitions (not really reflective of software engineering). (r80_Lec)

The material covered in the project in a way somewhat reflected the material in the lecture. It definitely applied in terms of working with teams (Scrum and etc.) and design patterns + testing, but for some cases, I would have wished it was brought up earlier so i could have used that information to its full advantage. (r81_Lec)

Not very well, the lectures and quizzes feel very detached from the project. (r82_Lec)

In general, I felt that the quiz reflected the lecture material. There were some cases where there was a bit of a gap between what was mentioned in class, and how to interpret the example provided in the quiz. I will say, there were quite a few questions that seemed like "gotcha" style questions, where the wording of the question seemed intentionally misleading me towards thinking the wrong answer. An example which comes to mind is the user story exam questions. The true answer contradicted something that was taught in lecture (ie. that the user story shouldnt reflect something unmeasurable, like the emotional experience of the user), and a possibility that I was not considering in the first place (ie. that the story was lacking the quality of independence). I initially made the assumption that the note had already been implemented, and so the user story was independent. However, the MC option made me think that my assumption was inappropriate, since there was no evidence to indicate that the note had in fact been implemented already. (r83_Lec)

Refactoring into SOLID principles made significant impact on our project and made making changes and understanding different parts of our project easier. (r84_Lec)

not related at all, or very minimal connection on helping us being able to do the project better (r85_Lec)

It was a nice application of the content we saw in class. Although it wasn't always necessarily connected, I thought it was a good opportunity to learn how to create a full stack app. (r86_Lec)

Very well. (r87_Lec)

Unless we did a portion of the lecture specifically about the project and in specific reference to the project, I feel as though the project did not reflect the material covered in lecture and on quizzes. Much of the things we needed to know on the project was not covered in lectures so a lot of it was not relevant. (r88_Lec) not overlapping that much. (r89_Lec)

I feel like the material covered in the project did not reflect the material in lecture all that well. This might be due to the amount of freedom we were given to implement the project. It would have been nice to be introduced a material in the lecture while also have the ability to apply it in practice concurrently with the project. (r90_Lec)

Poorly. (r91_Lec)

It was very well, and the material that was covered was good. (r92_Lec)

I don't believe the project reflected well on the lecture and quizzes (although some did, I feel like it was two completely different courses). (r93_Lec)

I didn't think that it reflected the material very well. The coding parts were not so difficult as to require the strategies presented in the lecture, and the non-code exercises were not really useful for actually completing the project (we mostly completed them after the code parts were already done). (r94_Lec)

Very well. I would mention that the weekly meeting with the TA sometimes felt very useless; nothing were said, the TA would just ask us if we were on track and if we had any questions but not really ask us what we did, what we're working on and what we will work on, which was what I though those meeting would be. (r95_Lec)

I think they did a good job covering most of the project material in lecture. (r96_Lec)

I think they were pretty different, and the methods to try and link them together (like making us do user stories) were kinda meh. (r97_Lec)

For the most part, not very well. It did cover some stuff, but a large part of the lecture material was either not covered, not needed for the project, or not very well incorporated into the project. For example, the weekly scrum meetings with the TA didn't really feel like a scrum meeting. (r98_Lec)

Pretty well, especially in terms of testing, async, API requests, testing, etc. (r99_Lec)

80% (r100_Lec)

1) Overall it was probably aligned with the course material (r101_Lec)

The project did a good job in reflecting the material in lecture and quizzes. (r102_Lec)

It helps when you review your old code and when you start writing new code. You would begin to consider more like making some refactors and considering possible design patterns. (r103_Lec)

I found a lot of the refactoring techniques and information about code smells that we learned in lecture to be quite helpful in developing the project. However, other than that, I mostly had to learn everything on my own time, since I was not familiar with any of the languages or concepts used in this project, so I didn't find the lectures to be overall very helpful with the project (r104_Lec)

Some of it (REST, test writing, mutation testing, promises / async) have links between lecture and project. Mostly the links are not very strong (for example we did HTML+CSS+JavaScript for C3 front-end but that wasn't part of lecture at all; we did a lot of OOP design patterns in lecture, but those weren't in the project at all). (r105_Lec)

While the material was reflective of the standards set in industry which we talked about in class, I think the project content was not reflective of the class material. This is because a lot of it involved starting from scratch without much guidance which might be what the intention was to get student to build something from nothing. However, this was particularly hard as someone who had not worked with typescript before. It led to quite an overwhelming feeling and confusion on my end. For instance, I remember for C0 when I had to create my first ever test I was so confused about how the test is even written or where should I start. I think having a lecture on typescript and a guidance on how the work with it/ writes tests would benefit future students. (r106_Lec)

Much of it was content for non code artifacts for the project. The quizzes reflected lecture content but the true/false style of exams (and most multiple choice style questions in general) can make the exams somewhat luck based. (r107_Lec)

The material covered in the project reflected the lecture relatively well. However, when coding I noticed I didn't particularly consider the high level design principles too much, given at times we were under pressure to produce an MVP. (r108_Lec)

Honestly, it felt like the project was not connected to the lectures. Sure, there is some connection as we go over refactoring, and user stories, and things that they required us to do for the project. But overall, it's not like you needed to watch the lectures or read the slides to figure out the project. However, I may be an outlier here due to my prior software development experience. Generally speaking, the project and course material does line up, there are direct connections you can see to the content, but it's not like you need the knowledge of the lectures to do the project. The quizzes definitely reflect lecture content. (r109_Lec)

A little bit (r110_Lec)

They are pretty good to cover most in lecture and quizzes. (r111_Lec)

lecture material was applicable to the project (r112_Lec)

Not at all (r113_Lec)

Material in project covered most of the material in lecture/ quizzes - around 75%. (r114_Lec)

30% (r115_Lec)

Somewhat relevant to the project, however a lot of the content was not directly relevant or sometimes we would approach things in the project before learning about them completely, etc. (r116_Lec)

'- covered well, but the material covered in class was behind the project, so I found that it wasn't actually helpful when doing the project (r117_Lec)

I think they build upon one another instead of actually reflecting similar material. (r118_Lec)

I think they were decently well reflected. I don’t think the material on the quiz was well reflected at all. (r119_Lec)

Moderately (r120_Lec)

I would say that the project is somewhat separate from the lectures / quizzes, the topics were similar like when we refactored the project or used REST endpoints but I think the project is kind of it's own thing (r121_Lec)

The material covered in the project reflects the material well. (r122_Lec)

The project seemed mostly separate from lecture and quiz material. (r123_Lec)

Not very well in opinion. (r124_Lec)

Often, by the time we learned the strategies for good coding practices, the technical debt in the project was too large to reasonably address. Other than that, the lecture material was very applicable to the project. (r125_Lec)

Maybe 20%. (r126_Lec)

the non-code overlapped decently. the code portion barley resembled what we learned in lecture (r127_Lec)

not at all really (r128_Lec)

5/5 (r130_Lec)

not very well. I thought testing methods used in the project reflected the lectures, but other topics like design patterns were not explicitly covered in the project. Implementing design patterns probably made the project easier to work on, but wasn't required. (r129_Lec)

The coverage of REST was there but other concept were often introduced after we were involved with checkpoints. (r131_Lec)

The material in the project did not reflect much of the material in lecture and on quizzes. The only parts that were reflected were the refactoring sections, and a general overview of how to structure classes. (r132_Lec)

I think the project does not cover enough content in lectures and quizzes. (r133_Lec)

Material in and out of lecture are covered reasonably well. Due to the scope of materials covered, the learning experience could benefit from a summary article for each quizzed section. While this would risk students only reading the summary, I believe this will offer far more benefits for the students who properly prepare by reading all required materials. (r134_Lec)

There did not seem to be a lot of overlap, except maybe at the end with user stories and design patterns (r135_Lec)

i dont think it reflect at all, completetly separateed (r136_Lec)

It was somewhat applicable but largely quite isolated from the lecture and quizzes. (r137_Lec)

The materials are not very related but rather like two separate portions of the courses. The lecture materials are about ideal conditions for software development but there are few opportunities to fulfill them all in the project. (r138_Lec)

c (r139_Lec)

I would say 50%. The part that i struggle most is about some material in 304 i believe, such as read html, or unzip, or api. Also I struggle a lot for frontend, cuz i never learnt it before (r140_Lec)

Initial quizzes/project didnt match up great (r141_Lec)

I think the theoretical material in the project reflects the material in lecture/quizzes pretty well; however, there are much more actual coding in the project in which I feel like I don’t really realize they are connected to the course material while doing the project. In another word, the project’s connection to the lecture material is often more of an after thought or something we only think of during non-code artifact. (r142_Lec)

Up until the testing module it was well-reflected. I personally found the rest to be a bit lacklustre since we didn’t really have time and urgency to fix code smells and refactor. (r143_Lec)

It covered it all very well. the checkpoints supplemented the lectures well and helped me prepare for some quizzes. (r144_Lec)

Not very well (r145_Lec)

I don't think it reflected the material very well, while it was relevant to the lecture material, the project is very practical. This means we had to learn a lot of the stuff my ourselves, and that may or may not be what the instructors were looking for (r146_Lec)

The material covered in the project overlapped with the material in lecture very little, I would say there is about a 15-20% overlap at most. The time spent doing the project was mostly just learning how to use TypeScript, Promises, and debugging. Lecture focused mostly on high level techniques that CAN be applied to the project, but with the timeframe given and scope of the project it just didn't make sense to apply them, although I certainly can see how some of them can be applied.\\As for quizzes, I would argue that there was pretty much no overlap. The studying I did for the quiz was 100% on the slides, and I didn't see anything that I would've learned from project that I could apply to any quiz (with the exception of Quiz 0 perhaps, but by the time I learned enough from the project Quiz 0 had already passed) (r147_Lec)

Like somewhat, but also somewhat no. The stuff covered in class, especially the latter half of the class with design patterns did not apply to our project. The first half, however, did apply. The quizzes on the other hand did not really help with our project, but covered the stuff in the lecture that did not apply to the project. (r148_Lec)

The non-code section had some similarity, but the project didn't have that much overlap even though we used some of the ideas such as refactoring in our project. (r149_Lec)

I feel like I did not have to use much of what I learned in class during the project. Because of how relatively small the project is, it is usually faster to "brute force" it rather than thinking about some of the principles taught in class. (r150_Lec)

A lot of course topic couldn't be practiced in the project. There were not many opportunities to apply process or low level design patterns. There was perhaps a little bit of refactoring that could be done. (r151_Lec)

not that similar (r152_Lec)

i think the non code artifact reflected the lecture/quiz material more than the code artifacts did; the code artifact felt pretty detached from the rest of the course (r153_Lec)

Non-code covered a lot of the material in lecture and quizzes, however I felt that the project did not really reflected any material we had except for the black box testing, and the material was covered after or almost at the end of the deadline for the checkpoint so it was almost of no help in the project. Also we did cover asynchronous calls, but material in lecture wasn't really helpful in the project as it mostly was theoretical without actual practice . (r154_Lec)

I used absolutely none of the concepts covered in lecture in my project. (r156_Lec)

The project reflected the material in lectures and quizzes well, with real-world applications of the many concepts used throughout the project. (r155_Lec)

I think the lecture materials on refactoring, tech debt, as well as documentation like User Stories are quite relevant to the project. (r157_Lec)

Not well. I felt like project and lecture were basically two different loosely connected courses. Yes, we wrote tests when lecture taught tests, and like ostensibly found and corrected code-smells with refactors, but it felt force and ultimately unrelated. Project was primarily learning how to write code with a partner. Lecture was just memorize a bunch of terms. (r158_Lec)

I felt there was a pretty big disconnect between project coding artefacts and lecture material mainly because of timing. Even though we learned about design principles, I felt like I didn't really get the chance to apply them in my project due to time constraints - there was no way I would be able to refactor my messy c1 code fast enough and in time for c2 deadline for example. So while we learned important concepts in class, I felt like I could't really put them in practice. \\The non-coding artefacts related pretty well to some of the concepts discussed in class, especially with respect to writing good user stories and getting us to think about which code smells or design principles are present in our code. (r159_Lec)

I felt that the material in the lecture was not properly covered in material. It felt that each quiz did not properly cover the most recent material and was difficult to study for. (r160_Lec)

I felt like the material covered in the project wasn't a good reflection of the material in lecture and in the quizzes. (r161_Lec)

The material in lectures provides a high-level overview for the project, which is somewhat helpful. However the lecture material does not dive into details of coding, so we still need to figure things out by ourselves. (r162_Lec) lectures didnt reallly help much in coding part but helpful for documentation (r163_Lec)

The materials gives a guideline for us to write our code. It reminds us that we need to set up a proper user story/work frame before we start to develop new functions. (r164_Lec)

The material covered the quizzes well. The project issue are not covered. (r165_Lec)

Pretty well. The stage of development, teamwork, user stories, and code smells were all things reflected in the project and lecture. However, design patterns were not reflected in the project. (r166_Lec)

The non-code parts closely reflect the content in lectures and were quite synchronized. I do feel that the code part of the project was somewhat disconnected from the lecture content, specifically regarding c1 and c2 (c0 did correspond to async testing, and c3 covered REST and possibly some design patterns); it looked to me that c1 and c2 contents were mainly parsing and logical tasks, which did not directly reflect the lecture contents covered when these 2 stages were in progress. It was possible for us to adapt some relevant ideas, such as high-quality codes obeying appropriate design principles, (r167_Lec)

I think they were quite separate. (r168_Lec)

very well (r169_Lec)

I don't feel the lecture is sufficient for quizzes. The quizzes are harder than the practice questions too. (r170_Lec)

Very well (r171_Lec)

Not at all. (r172_Lec)

Quite well, especially when considering concepts like the Open/Closed principle and Interface Segregation Principle. (r173_Lec)

I felt like the material reflected some of the material in lecture, but I felt like I could have benefitted more if I learned proper software engineering design patterns earlier on, before C1. I felt like my code wasn't as streamlined and well-designed as I felt like it could have been, and if we had learned how to code to a certain standard earlier on, my program would have been a lot easier to work with in the latter checkpoints. (r174_Lec)

The code part of the project had very little to no relevance to the lecture materials. It was almost like taking 2 separate courses. The non-code aspect tried to bridge the gap between the project and the lecture material by making us recognize which concepts in lecture are being covered by each checkpoint (i.e. tests, user stories, etc). However the overall project still felt disjoint from the entire course. (r175_Lec)

non-code reflect material in lecture pretty well, but coding part doesn't have much relationship with lecture in my view (r176_Lec)

Not so well. The material in lecture, while important, does not really pertain to the project in my opinion. Particularly in constructing the project, having knowledge about the design patterns (r177_Lec)

I think some part of the project reflect the material really well, like process and testing, etc. But there are some missmatch on timeline, for example we would have to apply high level and low level design principle starting from C1, but we only get to cover those material in the second half of the term, of course we can do refactoring, but given the tight deadlines most of us can't get hands on experience on refactoring effectively. (r178_Lec)

The materials in lecture and quizzes are really abstract and does not specifically help with building the project. There are lots of vague description in the spec, with hyperlinks that seemed not-so important (but is actually really helpful), making it hard for me to start the project well. I think the materials taught in class are too general to be implemented in our code and non-code. (r179_Lec)

Partially (r180_Lec)

The material covered in the project was in theory similar to the lecture materials, but when working on implementation there seemed to be a lot of information we needed which we weren't provided so it took a lot of google searching, piazza searching and help from TA. (r181_Lec)

I feel like they aren't very related. I feel like I could do the project just fine without anything that is taught in class. There needs to be more interactive activities and practices for the lecture materials. (r182_Lec)

I found there to be a decently large disconnect between lecture content and project. (r183_Lec)

I felt that the material in the project did not reflect the material in the lecture very well, with the exception of a few non-code activities which referenced user stories and such which felt fairly inconsequential. (r184_Lec)

not much (r185_Lec)

I would say that material covered in the project did not really reflect lecture material. It felt like this class was 2 disjointed classes, one focused on the project and another on lecture material. Although there was some overlap such as refactoring and maybe the way we did the project in test driven development. (r186_Lec)

I don't think they are related, except for some design patterns, but it usually comes up when I saw some pattern in lecture, and I realized: oh I used this intuitively. (r187_Lec)

The material covered the process of coding but not really specifics. Sometimes it would be challenging to apply certain design patterns or techniques when you yourself didn’t know what was happening. (r188_Lec)

Somewhat reflected quiz material. (r189_Lec)

The project was great for getting hands-on experience with the concepts that were taught in lecture. (r190_Lec)

Hmmm it didn't feel very related; I felt that the project was an extreme extrapolation of some stuff covered. Although it was made pretty clear that there was a distinction between lecture and project material. (r191_Lec)

Well it vaguely covered it because we weren't really required to use the vast majority of the content we covered in class. One reason was probably because everyone approached it differently. Nevertheless, there were somethings that we still could have used in our project, especially the design principles and patterns (SOLID etc...), however I think that could have been more useful as a refactoring exercise. I also think that somehow after doing CPSC 110 and CPSC 210, one subconsciously/naturally/already incorporates some of the principles while doing their first/second attempt at the project, thus it was entirely optional for the students to even use most of the content for c1, c2 and c3.\\I say this based on the work I have done for the frontend, addDataset(), listDataset(), removeDataset(), and the HTML Parser (for c2) (r192_Lec)

reasonably well, however, sometimes I felt like we were learning the concepts in class/quizzes after we had implemented them in the project. It would have been helpful to learn about them in class first. (r193_Lec)

They have the right amount of overlap. Not a perfect reflection, but I don't even think it should aim to be, because I like how the project and the theory develop different skills. One feedback that I have is timing; we learn useful concepts for the project in lecture when it's too late. For example, how we should use classes for queries, and what design patterns to use. When I learned about that in lecture we've already gone too deep into the project to make that refactor. (r194_Lec)

pretty well (r195_Lec)

Not really. I feel the project and lecture barely have a connection. (r196_Lec)

not well at all, maybe 20% (r197_Lec)

There was a good amount of overlap between the project and the material in lecture and set me up well for each checkpoint. (r198_Lec)

Quite well. (r199_Lec)

The non-code artifact greatly reflects the material being covered in the lecture as it ask us to apply the concepts into our project and making changes to increase the overall quality. (r200_Lec)

The concept of testing was covered in lecture and applied practically in the project. Code smells and code refactoring are also pratices that were implemented in the project. (r201_Lec)

Mostly. I think some of the common design patterns were not well covered with the project, but the project was very useful for topics like refactoring, user stories/documentation, and testing. (r202_Lec)

the lectures provided useful guidelines. (r203_Lec)

There was some material in lecture that I found was related to the project, such as teamwork and user stories lectures. There were also some material that I felt was not related to the project. (r204_Lec)

Not very well. (r205_Lec)

The lecture material did not feel extremely relevant to the project. (r206_Lec)

For the most part the project did not require much of the lecture material, however lecture material did help inform better design decisions (r207_Lec)

It accurately covered the course materials. (r208_Lec)

Not very closely, to be honest. Design patterns for example, did not touch at all. Still worthwhile though, quite useful to know and recognize. (r209_Lec)

I felt like the lecture and quizzes felt completely irrelevent to the project - especially after classes 2-3. That is why I think many of us stopped attending the lectures. The quiz content felt like it could be acquired by mainly just reading through and understanding the content. The project was the major part of this course for my learning - I really enjoyed it and learned a lot. (r210_Lec)

The material in the lecture and quizzes felt extremely separate from the material covered in the project. (r211_Lec)

Yes, it related well. (r212_Lec)

Not very well, but it is hard to simulate a real world work environment (r213_Lec)

There was some overlap but the project obviously involved much more raw programming than the lecture or quiz material prepared us for. (r214_Lec)

on project, not clear most time, I need to talk with TA to adjust some questions (r215_Lec)

around 80-90% (r216_Lec)

I kind of felt like the project was its own course and the lecture material and quizzes were a whole different course. There were some connections but I definitely felt like they were more isolated. (r217_Lec)

Excellent in lecture, poorly in quizzes (r218_Lec)

marginally (r219_Lec)

I don't think the material covered in the project reflected the material in the lecture very much. I found the lecture material to be pretty self-explanatory, but I did not use a lot of the concepts taught in lecture to do things for my project. (r220_Lec)

good (r221_Lec)

A bit related but in holistic view went very well together (r222_Lec)

I think the non-code was more reflective of the course material, but the project could have been better because you did not need to follow the principles in class. (r223_Lec)

Overall the non-code part was quite relevant to the lecture material, especially like the user stories. But for the way we design and implement the entire project, it is quite hard to use what we have learned in the lecture immediately. Because of the difficulty of the project, we usually stuck with our own methods and couldn't care about the things we learned in the lecture. (r224_Lec)

Not so relevant. Spending a lot of time to understand the requirement for each checkpoint, and the grades only focused on the percentage of tests we passed, not the code quality. (r225_Lec)

The non-code helped a lot for parts of the quiz and the lecture helped the most when doing REST. (r226_Lec)

Not entirely well, maybe about 15-20% seemed directly relevant to the quizzes. (r227_Lec)

Not very well. I think it wasn't that good. (r228_Lec)

I would say that it cover alot of the topics we saw in class such as api dev, class design, etc. However, I feel that the project could have been done without ever going to class. (r229_Lec)

Moderately well (r230_Lec)

Somewhat well (r231_Lec)

Very Well (r232_Lec)

C0 was difficult since class material hadn't caught up with it (r233_Lec)

I'd say quite a lot, of course the fact that the course project is using tech stack that has never been introduced in previous courses add a lot of frustration, but I liked how the general concepts / overview of things we're doing in project are usually mentioned in lectures in sync with in project. (r234_Lec)

In my opinion, the code part of the project somewhat covered the material taught in lectures. For the code part of the project, there wasn't much in the lecture material to begin with. Besides the different types of testing, assessing quality of code, and fixing bugs in time before it gets too expensive, not much was covered in class related to the code part of the project. So while it did help, the main focus for us was to learn a new language (Typescript) and code in it. What helped the most from class material would be how promises and asynchronous programming in general works. I would add, the non-code artifacts relate very well to the course material. As someone who has interned at a Software company before, I do understand the importance of formal documentation when writing code for big software solutions. I feel like it would have been better tested if more group members are involved in a single group, though in a university setting, it works very well. (r235_Lec)

Project was related to the slides but not 1:1. (r236_Lec)

I think the project implicitly reflected the course content. I think things like design patterns and principles are things that make themselves known as a project evolves, and are not things you always have at the front of your mind. So I think as I was working on the project, I naturally gravitated towards using design principles just to make it easier for me to understand the code, even if I didn't set out to specifically use a design principle. (r237_Lec)

I didn't look at lecture material (r238_Lec)

well (r239_Lec)

The project felt relatively fragmented from the lecture/quizzes, apart from the user stories part. At times it felt like taking two separate courses and the content was a little hard to connect. (r240_Lec)

boht components moderately complement each other. (r241_Lec)

Personally, I felt there's about 50% coverage, as most of the detailed parts required self-learning, especially libraries and languages, which were first-time encounter for me. It took me some time to understand each concept before I could start coding for the checkpoints. The material in lecture only covers the big picture ideas. (r242_Lec)

The material covered in the project reflected lecture and quiz material fairly well, allowing me to get hands-on experience with user stories and the agile process. (r243_Lec)

Pretty well for code smells. (r244_Lec)

i think the lecture didn't cover everything in the project- there was a lot that was difficult to figure out (r245_Lec)

Honestly it was alright - better than one might expect. Not enough design pattern or fuzzing test practice though. (r246_Lec)

The project and lectures covered most of the important material, but I personally found CI/CD, which wasn't included in the project, quite interesting. (r247_Lec)

Personally felt that there was very little connection between project and lecture material (r248_Lec)

completely unrelated (r249_Lec)

The project covered material in lectures fairly well but not enough for quizzes. (r250_Lec)

I think they tied together fairly well. For example, during c0 we first covered testing, then in c1 and c2 we covered things like development teams, process, specs, refactoring and design principles, and in c3 we covered API design, Restful design, Devops and design patterns. So it felt like things covered in lecture and quizzes were fairly relevant to the project at the time. (r251_Lec)

I felt that the material in the project somewhat reflected the material in lecture, however it didn't provide much assistance when it came to doing quizzes. (r252_Lec)

Material covered in lecture helped me understand some concepts of the project. (r253_Lec)

The principles learnt in class are sometimes being applied to code writing. The team collaboration part really helped me building a team with my teammates. And the lecture material is often referred back to when I write my non-code artifact. (r254_Lec)

Not at all. They seem to be very unrelated. (r255_Lec)

The material in the project was a very good reflection of things we learned in lecture. However, the quizzes were not fair assesments of waht we learned in lecture or what we learend in the project. It seemed like the quizzes only tested speical cases that we do not ever ecounter in the project. (r256_Lec)

Not much, quite disconnected from the project to the lecture I would say (r257_Lec)

A large part of the introductory material is related to the code. For example the refactoring, Scrum and Kanban, types of testing. However when it came to more abstract theories such as covering ethics and high-level coding it was not as applicable to the checkpoint. (r258_Lec)

It was separate. But I liked learning about two different things at once, the coding project and also industry practices. (r259_Lec)

I think it was pretty good way of getting practical experience for the things discussed in class. Like it gave concrete examples of async, testing, etc. (r260_Lec)

Not very relevant (r261_Lec)

The project felt a lot more challenging and hands-on the the lecture material, which was more conceptual and abstract. I feel like learning more about the lecture material first, especially high-level and low-level design, code smells and refactoring would've helped for the project implementation (r262_Lec)

For the project, I did not feel like I could relate the code very well to the lectures because I had to learn the syntax first and just understand what the code was doing before I could even think about the more advanced techniques used in lecture. However, the lecture slides were really helpful for studying for quizzes. (r263_Lec)

Material covered in the project largely reflected the material in lecture and on quizzes. (r264_Lec)

some of it, the project covers, refactoring, test, high and low level design (r265_Lec)

Project is a good addition to the course material but I won't say they are exactly matched with each other. Many concepts of the low level design's not covered till project is pretty much done. (r266_Lec)

I felt like the project material was pretty seperate from the lecture material. In hindsight, I think they are definitely connected, but a lot of the pattens that would have been useful in our project were taught too late and the ones I did use were ones I had already known about from previous courses. (r267_Lec)

Code: I don't think the project and the lectures were connected at all.\Node Code: Team management techniques were explained well. (r268_Lec)

Pretty well, I can't think of anything that could be changed. (r269_Lec)

The project is a very good project, but I don't think ti reflects the material in lecture and quizzes very well. The project feels like a separate course overall from the lecture and quizzes. (r270_Lec)

the project material gave the opportunity to apply many of the lecture topics, although it was up to the student to choose whether to apply them or not (r271_Lec)

There was some overlap between the project material and that covered in lectures and quizzes. (r272_Lec)

totally different. Not well. Content during lectures and quizzes still helpful for knowledge and on project, but project not helpful for lectures and quizzes. (r273_Lec)

Good (r274_Lec)

Asynchrony was covered in c0 of the project (testing), but I don't feel the later technical topics reflected the material in the lecture all that much. As for non-technical topics, the Scrum processes and ideologies did appear in the project (user stores, code reviews). (r275_Lec)

Pretty well. Most of the questions on the quizzes can be mapped back directly to one or two sentences from the readings. (r276_Lec)

The code materials covered in the project are less related to lecture and quiz materials, and required more self-learning. The non-code portion are more related to the lecture materials and allow us to practice with concepts that may seem abstract when we first learned it. (r277_Lec)

I found the beginning of the course cover more about the project than towards the middle and the end. Course material was always relevant for quizzes, but the quizzes themselves weren't too relevant to the project. (r278_Lec)

Most of the course materials are applicable and useful when doing the project. The course materials covered not only basic coding knowledge for Typescripts like promises and async&await but also conceptual design choices and ways to resolve code smells and principle violations. (r279_Lec)

The project had some similarities to materials in lecture, but they were not very similar to the quizzes and lectures. (r280_Lec)

Not at all other than the use of agile (r281_Lec)

A large difficulty for the project was the use of typescript, as i have never used it before, not did we do much in lectures. however, for c0 specifically, the lectures covering promises was very important, as without it i would have struggled a lot to complete the early project checkpoints. (r282_Lec)

The checkpoints were linked well to what we were studying like learning about testing for c0 or RESTAPIs for c3. (r283_Lec)

Quite poorly. Given the constraints placed on us, with regards to time, it really was quite difficult to write "good" code by the standards of what was taught in class. (r284_Lec)

I think the material covered in the project reflected the material in lecture and the quizzes quite well. (r285_Lec)

I think that the testing module helped me with the project. other than that I think the project and lecture were fairly independent. (r286_Lec)

The project and material in lecture were pretty similar. (r287_Lec)

It was okay but not to a large extent. (r288_Lec)

I think one could do much of the project without following along in class, but the course project reinforces some of the concepts taught in class. For instance, having to do the refactoring in c2 was the first time I had designed and re-designed a non-trivial software project, and I gained a much better feeling for why we use design patterns. (r289_Lec)

Generally, the project material closely aligns with the theoretical concepts discussed in lectures. Lectures often provide the foundational knowledge needed to understand the problem space and the theoretical underpinnings of the tools and methods used in the project. This alignment ensures that when I begin to work on the project, I have a solid understanding of the necessary concepts. (r290_Lec)

I think they are some what relevant, but these relevance sometimes are overlooked when I am writing the project, because the implementing the functional part of the project is really what takes up 95% of the time and effort. (r291_Lec)

123 (r292_Lec)

The lecture material felt separate from the project. (r293_Lec)