Summary

**Theme 1: Technical Debt Identification**

4. Response:

Software prototype development stages include problem evaluation, solution identification, implementation, and possible training.

Key challenges: tools, standards, frameworks, and keeping up with evolving technologies.

5. Response: echnical debt is identified through evaluating code understanding and using code from external sources without comprehension.T

6. Response: Indicators of technical debt include code duplication, complexity, and inconsistent coding standards.

**Theme 2: Technical Debt Measurement**

7. Response: Technical gaps are measured using code metrics, including code duplication, complexity, ownership, and code quality.

8. Response: Common tools for measuring technical debt include SonarQube and Code Climate, but integrated development environments (IDEs) often have built-in extensions for identification.

9. Response: Prioritization is based on the criticality of issues that could stall the project or harm reliability.

**Theme 3: Technical Debt Impact Evaluation**

10. Response: Unresolved technical debt affects reliability, performance, and maintainability, leading to errors and decreased software quality.

11. Response: Technical debt example: Incorporating external code introduced bugs, delayed fixes, increased costs, and impacted user experience.

**Theme 4: Early Debt Repayment**

12. Response: Practices include continuous learning, seeking help when needed, and maintaining transparency and integrity.

13. Response: Incentives: Financial support for courses, emphasizing integrity, and promoting innovation through events like hackathons.

**Insight:**

14. Response: Provide career guidance, exposure to career paths, encourage innovation, and create a learning-eager environment for students.

Full transcription

Here's the transcription of the second participant's responses to the themed questions:

Participant: Okay, maybe I'll introduce myself. My name is Mugoya Dihfahsih. I'm a student from Makerere University, and I'm doing research in software architecture. The main objective of this research is to identify key metrics for measuring technical debt in software prototypes. For software developers and high-speed developers, we may include these. Have you ever heard of self-taught developers? And also students that are doing finance projects. So, as we understand well, maybe I take this through what I mean with technical debt. These are consequences of taking shortcuts or compromises in your software development. This technical debt normally slows your development, they slow your productivity, they slow your level of engagement with the project itself. So, those are the consequences of technical debt. This project, the interview, is themed in four. We have identification of technical debt, we have impacts of technical debt, we have measurement, and then we have repayment of technical debt. So, I will start with the first one.

Please, where you introduce yourself, tell me your name, what you're doing, and your role.

Participant: My name is Timothy. I'm a student of Kyambogo University. I'm pursuing a bachelor's in financial studies, mainly problem-solving softwares.

Interviewer: Institution-based?

Participant: Yes.

Interviewer: Mobile?

Participant: It's going to be institutional, meaning they're going to be locally.

Interviewer: So, they're going to be desktop applications?

Participant: Exactly.

Interviewer: Okay, so what are the main objectives and requirements? Just overview, not the detailed stuff.

Participant: The main objectives for the project in particular are to reduce costs, to reduce the time spent performing these tasks. Three is the workload, efficiency, effectiveness of the results, as well as credibility.

Interviewer: Yes, proficiency.

Participant: You said objectives and something else. You said the main goals or the requirements?

Participant: Requirements. Okay, it will need a particular skill set of software. Right now, the university doesn't provide this structure, but they give us parameters for languages, which as well gives us a basis on how to maneuver, which as well helps a lot.

Interviewer: So, it's just a matter of...

Participant: Okay, so like I said, technical debt really slows down the software development. It introduces bugs and requires efforts when you're adding features to it, to a product, and also maybe when you're fixing or improving the software. So, you have to put in a lot of efforts when you do that. So, that's technical debt.

Interviewer: So, in Theme 1, we are going to look at technical debt identification. In your experience, what are some of the stages that you normally involve when you're developing a software product?

Participant: The stages? One, we look at the problem.

Interviewer: Yes.

Participant: Problem evaluation, possible solutions.

Interviewer: Possible solutions, you have to zero down to the actual solutions that are really going to be helped.

Participant: Implementation, which comes down to planning and scheduling how these things are going to be done. And then, the part of training, in case of any languages or software that we currently don't know very well, or we need to learn other frameworks, particularly in order to implement this. As well as getting exposed to the various libraries, so that the work is simplified and put on the first track.

Interviewer: So, how on the projects you're working on, what's your role? What do you normally play on the projects?

Participant: Normally, I am a strategist. I come up with a framework of implementation of activities. I break down, as you know, I try and assign tasks, but not in a very administrative way.

Interviewer: But it's mainly just depending on what people are saying.

Participant: I mean, in the software development.

Interviewer: I'm a developer.

Participant: Yeah, a software developer. Basically, that's what I wanted.

Interviewer: Because I didn't want to ask you different things yet.

Participant: You say in the project, you're specific.

Interviewer: Okay, great.

Participant: So, what are some of the key challenges that you normally encounter related to tools, standards, frameworks, programming languages, and the other compilations of tools?

Interviewer: What challenges do I have to get to them, or what challenges do I face?

Participant: Yeah, when you're relating with these tools, the programming languages and the frameworks, because you're talking about the library.

Interviewer: Yeah, yeah, yeah.

Participant: So, now, one, if you're not well, if you don't know what you're doing, yeah?

Interviewer: If I'm to say, if I'm trying to do something that has in line with maybe data science or machine learning.

Participant: Yes, yes, yes.

Interviewer: And you're not really very competent with the whole scope of the knowledge and the whole discipline.

Participant: You have a challenge of, you do not exactly know what is going to exactly be the best library to use.

Interviewer: You're going to look for something that looks like it's going to solve the problem at the moment.

Participant: Yes, it will, but there might be a much better, unified, better way, library, within that whole library that you can use.

Interviewer: Then, with the languages, languages keep on having very many frameworks that are coming up.

Participant: Yes, yes, yes.

Interviewer: These frameworks are very good and very efficient in what the main objective.

Participant: So, the problem is keeping up, keeping up with these frameworks and, as well, getting exposed and learning.

Interviewer: The time-lapse is not so friendly, and as well as, you see it.

Participant: You mean the time-lapse between the learning?

Interviewer: Learning and implementing them in a project is not really very favorable, as well as the financing.

Participant: Because, you see, when you're learning some of these things, it's much better if...

Interviewer: What we have, what the issue we face mainly, as university students, we can't pay for the short courses externally.

Participant: And yet, the short courses online are a much better...

Interviewer: Courses, yes.

Participant: Yes, and a free one.

Interviewer: The free ones just give you an introductory...

Participant: Yes, class.

Interviewer: Yes, and they do not give you well-structured projects to design.

Participant: The projects are very, maybe, might be very tight and might not really be realistic for us, who normally sit in the office.

Interviewer: And might not equally help us.

Participant: Okay.

Interviewer: So, they send us back to what you are trying to address.

Participant: So, you're saying the problem is the time for learning, and then maybe the financial resources that are available for you to learn.

Interviewer: Okay, so how do you normally identify or become aware

of technical gaps in your prototype?

Participant: How do I?

Interviewer: How do you become aware of technical gaps in your prototype?

Participant: How do I become aware?

Interviewer: That is quite simple. You just know what you don't know.

Participant: It's very straightforward because what you don't know, you don't know.

Interviewer: You can't be like, I don't know this, and you have to speak about it. Do I actually know it?

Participant: Actually, I meant maybe the techniques you normally use, but that enable you to identify technical gaps in your prototype when you're developing it.

Interviewer: Ah, technical gaps.

Participant: Okay, this comes back to the problem.

Interviewer: When you're looking at the solution, the solutions for the problem, yeah, sample solutions.

Participant: You look at all these solutions and technically, you know that in this area, this will require this.

Interviewer: Yes, yes, yes.

Participant: But how much do I know this?

Interviewer: So, in that way, you'll obviously identify that, you know what, there is a particular module in this that I need to do it.

Participant: I either need to just go and get it from somewhere, depending on the stringent time frame that the work needs to be done at,

Interviewer: Or I have to go and learn if the time frame for completion of the project is longer.

Participant: Yeah, so it's just looking at it and evaluating yourself personally with the skills that you have.

Interviewer: So, basically, what you do here, you might do personal projects, right? Not team work?

Participant: If it is a final year project, yeah, it is personal.

Interviewer: But if it is an external project, normally, it is a group thing, it's a team thing.

Participant: So, for the final project, because basically that's what I'm looking at.

Interviewer: So, we have these, let me say, guidelines, maybe contacting the stakeholders, maybe reviewing the whole prototype.

Participant: Maybe supervisor, maybe going through the code reviews, you know the code reviews?

Interviewer: Yes, yes.

Participant: Give the code to someone to review it.

Interviewer: Maybe during, maybe pair programming, pair programming is like, maybe you're a junior, you have a senior person to consult, something like that.

Participant: So, at which of those stages do you prefer identifying or how do you use to identify technical gaps?

Participant: That can be more preferable for you.

Interviewer: When you're addressing stakeholders, because now, when you go to review your code, you've already built the code.

Participant: Yes, yes.

Interviewer: And you see, the explanation of technical debt-wise, the way I understood it, the way you really brought it up was,

Participant: Technically getting, going to sites like GitHub, you get code that does a particular bug, but you don't know what it actually,

Interviewer: You don't know what it actually means and how it can be performed, then you bring it and use it in your project.

Participant: Now, when someone reviews your code, they're going to see that.

Interviewer: They will know that everything is flowing, everything is working, so that can't work.

Participant: So, you have to protect the trust.

Interviewer: Yes, when you go to your supervisor and you're reviewing your project, he's going to see the thing is working, he's not going to ask many questions,

Participant: He's going to, as long as you're sure that you know what you're doing, and technically, if you copied something from somewhere,

Interviewer: And you know that this module was copied, you can technically explain, like this, this, this, this, you can explain a page in.

Participant: But when you're getting, when an evaluation is taken, this means that you're evaluating one, you as a developer,

Interviewer: Two, the users.

Participant: Now, there you'll identify that, you know what, what the user wants, I don't know,

Interviewer: And that means I have to go and copy it from somewhere, meaning the user, take the code out and copy it.

Participant: So, what are some of the indicators, yes? Indicators, I mean, maybe code metrics. Code metrics?

Interviewer: Yes, code duplication, we have code complexity, we have code ownership, and then we have code, the quality of the code.

Participant: Yes.

Interviewer: So, those are the metrics. So, what are those indicators that suggest that they're taking a good hit in your product?

Participant: It might be a technical indicator.

Interviewer: Indicator is something, it's like a red flag.

Participant: Yeah, yeah, yeah.

Interviewer: This is, this has been reason and it's showing that you have a technical gap, you know.

Participant: What are those indicators?

Interviewer: Yeah, these ones that I've listed, maybe duplication.

Participant: I think it is after, after, okay, you see, now you see the way you brought it up, you're mentioning a metric that is,

Interviewer: You want to use to determine whether you have a technical debt.

Participant: Am I now, you see, you're trying to say that as a person implementing, because now I'm the one on the project,

Interviewer: I cannot, I cannot kind of, okay, I cannot actually put myself on a scale to determine whether I have a technical debt.

Participant: Or not.

Interviewer: Yeah, or not, yet I'm the one on the actual project and I'm the one going to do it, who actually knows where the technical debt is.

Participant: So, it is, it is more like running in circles, like I know I put something there, you know, I have a technical debt in my project.

Interviewer: And then I develop a metric, or I implement a metric to determine whether I have a technical debt inside my project.

Participant: No, I mean, like, metric I'm talking about, it could be a red flag, maybe you have long lines of code, maybe you have a single class, maybe, let's say.

Interviewer: Are you trying to talk about errors?

Participant: Exactly, technical debt could be, remember when I talked about the bugs.

Interviewer: But now you say a technical debt can result in two bugs,

failure for projects to run, you didn't really say that it as well has bugs.

Participant: Of course that's technical debt.

Interviewer: You have bugs.

Participant: Now, advantages, advantages, those indicators are very simple.

Interviewer: The, the, the, the, how do you call that?

Participant: Yes, code, I'm forgetting how they call those tools that we use.

Interviewer: Extensions?

Participant: No, no, not extensions.

Interviewer: Which tools are you talking about?

Participant: I'm talking about notepad.

Interviewer: Those are called IDEs, they have the integrated.

Participant: Yes.

Interviewer: And now then, automatically there is how, if you, now you see for every language you're going to write, especially in VS Code.

Participant: Okay, there's a particular word I'm looking for.

Interviewer: Okay, actually that would be, that was going to be my question.

Participant: There is, you have extensions.

Interviewer: Yes.

Participant: Now these extensions, yeah, they help you identify.

Interviewer: Because my next question was going to be, which are the current tools that you normally use to identify or measure technical debt in your prototype?

Participant: But now these ones are just, these tools here, when you get an extension for a language, yeah, they automatically come with it.

Interviewer: Because these extensions here has an indicator of sort.

Participant: Because it will know that, man, you've written this code in this language, but it doesn't, this is not the law of this language, so this is wrong.

Interviewer: You've missed something here, this is wrong.

Participant: So it will always highlight for you.

Interviewer: Automatically.

Participant: It has an inbuilt function in that extension.

Interviewer: Okay, so for code indicators, like the red flags, that suggest that you have a technical debt for you, look at maybe the tools that you use.

Participant: Yes.

Interviewer: Okay, so that could be maybe code coverage, something like that, or code quality.

Participant: So which of those ones you've mentioned, which tools do you normally use, use VS Code?

Interviewer: No, VS Code is, the M and the Y have escaped totally.

Participant: It is a platform, more like an application that enables us developers to develop, to use various languages.

Interviewer: In the same environment.

Participant: Yes, in the same environment, without having different platforms, that just work for particular languages.

Interviewer: Like the way you see Python, you can go up it using PyCharm, that one is about specific.

Participant: Yes.

Interviewer: Like, yeah.

Participant: Now, VS Code just helps you to develop, use various languages and frameworks to develop on a one unified environment.

Interviewer: Maybe that was what I was talking about, that I identified technical debt.

Participant: We have step size, sonar cubes, code climate, and so many of them.

Interviewer: Have you ever interfaced to identify technical debt in your prototype?

Participant: I doubt actually I would even need to use them, except if I'm at a managerial position.

Interviewer: But as a developer, I doubt, because, as I said before, this is something that you as a person know that you want.

Participant: Technically, no.

Interviewer: What you're doing?

Participant: Of course, technical debt, maybe I never explained the types of technical debt.

Interviewer: Intentional technical debt.

Participant: Intentional.

Interviewer: Yeah, then there is one unintentional data, the one we don't, we have no control.

Participant: So maybe you have a tool that identifies that this code has a technical gap with it.

Interviewer: So I'm saying maybe, have you used any tool?

Participant: No, I've not used any.

Interviewer: Okay, but you know there are those tools that exist that can help you do that, right?

Participant: Yes.

Interviewer: Okay, so how do you normally prioritize which technical debt to address first in your prototypes?

Participant: How do I normally address?

Interviewer: How do you prioritize?

Participant: How do I prioritize?

Interviewer: Yes.

Participant: Normally, it is that that is going to really affect the project.

Interviewer: So you consider the criticality?

Participant: Criticality, like the one that is gravely going to put the project at a standstill.

Interviewer: So at which, because I'm looking at SDLC, software development lifecycle, so at which step do you normally do this?

Participant: Development.

Interviewer: The development part, that's the implementation.

Participant: Implementation.

Interviewer: Okay, so let's look at the technical debt, but how does these unresolved gaps affect your software products?

Participant: Maybe in terms of reliability, performance and maintainability.

Interviewer: So it's time.

Participant: One, the time you take, you spend looking for the right framework.

Interviewer: Yes, yes, yes.

Participant: There's a lot to, you don't know what you've done, exactly, you don't know what it actually does.

Interviewer: So you're not quite credible, meaning that your software is not reliable.

Participant: Yes, yes.

Interviewer: The quality, the quality might not be as good because you can't maintain it.

Participant: Yes, yes.

Interviewer: So how about if you provide me a specific example of how you maybe interface technical debt in your project as you're developing?

Participant: And it limited me?

Interviewer: No, maybe it was like, if I remember I was doing a project, but I did this, this, this, maybe I never documented something,

Participant: but when I needed to maybe to integrate a functionality in it, I had to again to go through the code base,

Interviewer: I understand it before I implement a feature to it.

Participant: So it cost me maybe time, maybe resources.

Interviewer: No, it is always time. Time is a main factor.

Participant: Yes, actually, it also, the muller itself, if you were not doing anything.

Interviewer: Yes.

Participant: Okay, so let's look at the last thing, that is repayment of the technical debt.

Interviewer: So, are there any strategies that you know of that encourage repayment or fixing these technical debts during software development process as early as possible?

Participant: Do I know of any?

Interviewer: Strategies?

Participant: Strategies?

Interviewer: Or something like practices?

Participant: Oh, practices. One, get as much exposure. One, that's the best. Get as much exposure in the various technologies.

Interviewer: Not just languages or art, but in the various technologies.

Participant: Because right now, ChatGPT is helping people out very much because it

's a machine learning, machine learning that can do natural languages.

Interviewer: Yeah, chatbot.

Participant: Yeah, chatbot, like this. So it's very helpful to very many people. So, one, if you get a chance to get to as many technologies as possible, because that is a loan, that is in that sense.

Interviewer: That could help you in future projects.

Participant: That is future, but if you're a master of something, you can decide that I'm going to do it for money, but I'm going to use it in my future projects.

Interviewer: In your current project, yes.

Participant: Yeah.

Interviewer: I'm going to use this.

Participant: So, that's one.

Interviewer: As a means of paying it.

Participant: Then, I think there's two things that I said. In case you're a master of a certain technology, just do that in your free time, maybe you work for people, yes. Then, as well, you have your project that you're doing. You know that I'm not conversant with this, but you know that if I put a certain project here, it is going to take so much time to be able to do it. So you're going to have that amount of time for you to get to learn it, get to learn that particular language or technology.

Interviewer: Okay, so if I provide you with a scenario where maybe you're working on a software prototype, and maybe it's in the middle of development, and you've identified a critical technical debt, what would you do in such a situation?

Participant: If it's in the middle of development, this is a perfect scenario. If it's in the middle of development, one, I'm very sure of the work, because I'm the developer, I'm the one doing the work. If it's in the middle of development, then this is the best scenario, because this means that I am very familiar with what I'm doing.

Interviewer: Yes, yes.

Participant: Now, if I'm a managerial level, and maybe I'm still in the middle of development, it depends on the project.

Interviewer: Okay.

Participant: Yeah, because you know, I'm running on time.

Interviewer: Yes, yes.

Participant: I'm running on time, I need to get to deliver my project on time. Then I have a debt to pay off, and maybe this is a big project that I have to get to deliver. So what do I do? Depending on how much of that particular language that I know, I can decide either to assign it to someone who has a, or has experience in that language, in that technology.

Interviewer: Yes, yes, yes.

Participant: Because they can deliver within a shorter time frame, or I can decide to get to learn it on my own, or at least get to get exposure to it, because my project is not yet due.

Interviewer: Okay, thank you so much. This has been a very great conversation, and I really appreciate your time and your insights on this topic.

Participant: Okay, it was nice.

Interviewer: Thank you.