Summarised from the transcript

**Theme 1: Technical Debt Identification**

4. In your experience, what are the stages involved in developing software prototypes and what are the key challenges you have encountered related to tools, standards, frameworks, programming languages, and conventions?

- The participant discussed the stages of developing a software prototype, emphasizing the importance of making the project usable for end-users and potential challenges, such as learning new programming languages/tools.

5. How do you identify or become aware of technical debt in your project?

- The participant mentioned that they identify technical debt by involving stakeholders, particularly experienced individuals, and by actively testing the prototype to ensure it meets its objectives.

6. What are the indicators/red flags that suggest that there is technical debt in your processes or product?

- The participant highlighted indicators like code duplication, complicated code, and

missing features as red flags for technical debt.

They emphasized the importance of maintaining documentation and addressing these issues during maintenance.

**Theme 2: Technical Debt Measurement**

7. How would you like to measure technical gaps in your processes or product?

- The participant suggested using tools like Visual Paradism to identify technical gaps and automate certain tasks in their project.

- They also stressed the importance of involving experienced individuals and performing code reviews.

8. Which are the current tools if any or measurements you would use to measure technical debt?

- Visual Paradism was mentioned as a tool for connecting the website to the database.

- The participant also mentioned using the internet for research and leveraging the university's free resources.

9. How would you prioritize which technical gaps to address first?

- The participant favored addressing technical debt during the maintenance phase, especially when scalability issues arise.

- They highlighted the significance of having good documentation to avoid misunderstandings.

**Theme 3: Technical Debt Impact Evaluation**

10. How does unresolved software gaps affect the quality attributes of your software prototype (e.g., reliability, performance, maintainability)?

- The participant

- explained that unresolved technical debt can impact the reliability and performance of a system.

- Changes made later may affect the system's initial design and stability.

11. Can you provide specific examples of how technical gap has affected the project outcomes or the end-user experience?

- The participant provided an example where they had to learn a new technology (Adreno) due to not having prior knowledge. This affected the project's progress and demonstrates how technical debt can arise.

**Theme 4: Early Debt Repayment**

12. Are there any practices or strategies in place to encourage early repayment or fixing of technical gaps during the software prototype development process?

- The participant mentioned that early allocation of supervisors who provide guidance and feedback on projects helps in identifying and addressing technical gaps early.

13. What incentives or mechanisms exist to motivate the team to actively manage and reduce these gaps?

- The participant highlighted that the availability of free resources like labs and internet access, along with a supportive academic environment, serves as an incentive for students to actively manage and reduce technical debt.

Insight:

- The participant emphasized the importance of orienting students before they begin projects to help them come up with innovative solutions to real-life problems.

- They suggested that students should be taught how to generate project ideas effectively.

**Full transcript**

Okay, so basically, my name is Mugoya Dihfahsih, and I'm a master's student at Wakilei University, offering software engineering and computer communication. That is the degree that I'm doing. I'm currently carrying out research in software architecture. Basically, the main objective of the research is to come up with a framework that will guide students in identifying, measuring, and evaluating technical data in their prototypes. Basically, a technical debt, these are consequences of you taking shortcuts, or these are consequences that you suffer in the long run after taking shortcuts in developing a prototype. These are related to a financial debt in which you have to pay interest after you're taking that debt. So, first of all, these technical debts, what they do to a prototype or development, they slow down the development, they kill the morale of software developers, and they really impact the software development in a negative way in the long run. In the short run, maybe you want to pass an exam, you take shortcuts, and you have a code that is running, but in the long run, you have to pay for it. Maybe you've gotten a job and you need to implement a project, they chase you out of the job and maybe you lost the employment. So, these are long-term repercussions of a technical debt. Maybe to start with, the question that I'm going to ask you for this interview, I would like to just know your name, the course, the year you are in, and then the project maybe you're working on or you've worked on. Yeah, so, maybe I'll start with that.

So, my name is Sserumaga Patrick. I'm a third-year student at Kyambogo University. In the course, I do Bachelor of Information Systems. Then, the role on the project, or the project that I'm planning to work on, it is a project about an online tool, as well as it has hardware, but it basically works on testing of blood groups. So, it is all about developing a personal kit that a person can use from anywhere to test his blood group, and the information will be recorded in certain database, for example, hospital, and like that.

So, what are the main objectives and requirements of that particular project?

So, the main objective is that the person to be able to know his or her blood group is to avoid this issue with the program that is here currently, that when you want to know a certain blood group, you have to go to the hospital, you make a choice of lines, sometimes you end up not being worked on, like that, but you are able to get that personal kit, and you have it anywhere with that problem. So, no delays. So, that's important. That's a great system, by the way.

In your experience in social engagement, what are those stages that involve yourself in developing a prototype, and what are some challenges that you normally encounter in using tools, standards, formats, programming languages? So, what are those stages of software development that you normally go through when you are developing a prototype?

On developing my prototype, I have to attach it with the basic information that the user will need so as to use it, not me coming up with my project, it's just this, for you to understand it, but on putting it out to the people, and those people want to be able to work with it, because if at all, the people you are making for the project or the product are not able to use it, it is kind of like, it is useless, yet you have to develop your project, it has to be, at first, beneficial even to you. So, challenges that I can face, there is, maybe you want to integrate some languages that could not make your work big, but you are not familiar with it, so you have to go that extra mile to learn those languages. So, the challenge is learning the programming language you are going to use on your project. Because you might be knowing a certain language, but it is not appropriate, or it will involve large codes, yet it is that simple a language that you can use to simplify your work.

Okay, so, how do you normally identify, maybe, technical gaps in your prototype? So, how I usually identify those gaps is by carrying, structuring some questionnaires, or plus taking my prototypes to those that are going to be my biggest stakeholders. For example, when I talk about my kit, taking it and explaining my information to, maybe, people like doctors in certain hospitals, and explaining what it does. So, as the doctor is experienced, he can tell you that this is lacking, this and this is lacking, so implement it. So, you basically contact the stakeholders? Ah, the stakeholders, because they are the people who are going to use that system. Oh, that's great. Because for you, you can think something is done, it is complete, but on taking your prototype to the people, you can identify like three or four things that need to be rectified. So, what are some of the indicators or red flags that would suggest that your prototype has those technical gaps? So, the red flags that could suggest, like not going to the stakeholder or the person who used it. I mean like when you invoked your project, the prototype, the code, what are some of the red flags or the indicators that show that your prototype is in a mess? Okay, so that can be carried out in my personal testing. Because I usually, me as the project manager, the person who wrote me the project, in my mind I know what the project has to do. But when testing it, maybe, for example, I want the person, like you test for your blood group, you get to know it, it is recorded in the system. Automatically, not by you, because if at all it is you, personally, it can be you inserted it. So, it has to be automatic. But if at all I want that thing to be inserted there automatically in the database, and yet after here, trying my prototype, I get the result, I see it, but when checking the database, there is nothing. So, I just know that there is something missing. So, maybe in terms of code duplication, code complexity, log lines of code, quality, maybe code ownership, maybe, yeah. So, in those which are maybe poor documentation of your code, maybe by coupling within the project code that you have used, so what could be some of those indicators that maybe you realize that if this happens to my code, then there is an issue with the project. If at all anything happens to the code I am using. Yeah, maybe code duplication, maybe you have duplicated the code, maybe you have written complicated code that you cannot understand, so for you, your development, what are those? So, some things that can bring out some mistakes. Yeah, yeah. That one happens that maybe I call it duplication, because you can be using a certain code to insert information from the website to the database, but you find

yourself that that code is somehow similar where you are going to use it on another page. So, duplicating it, you might miss some information and leave out the information of the other original code. Yeah, yeah. So, that one brings out some issues. So, that one really indicates that you are heading to a technical danger. So, we go to maybe theme number three of technical measurement. So, how would you like to measure technical gaps in your processes? How to measure? Yeah, in measuring I talk about maybe using version control, maybe also contacting stakeholders, maybe using tools, technical tools that help you to measure technical gaps. So, how would you like to measure technical gaps in your project? You can use these, let me say, those experienced personnels, like people that have been in the field before. Because here we develop projects, but they are like an additional step, something that already exists. So, I can try inquiring from those experienced people, because I can even be using a certain language or a certain tool. But when you inquire from the person who has spent some time in the field, there you can change and use this tool, you can change and integrate this language, even though you give your old language, but you can integrate this language. So, using that experienced person, you know, because for them they know better than you. So, you do what, you go into code reviews of your programming? That, plus you are searching on the internet. Also the searching, okay. So, what are some of the current tools that you normally use to identify technical gaps if there are any? That I use to identify technical gaps in my projects? Yes. So, I usually use these tools. We have software called Visual Paradism. So, that software helps you that the information you write in your website, it connects directly to the database. And if at all you have integrated your software, you have used that software, but still your information is not going the other side, meaning you have a problem with your system, because the software has to do it. Oh. Yeah, it has to do it. That's great. It's called what? Visual Paradism. Visual Paradism. For it, it identifies that whether you have… Yeah, it connects your database to the website. Okay. You insert the information in the website, it goes directly to the database. Oh. So, how do you normally prioritize which technical gaps to address first in your project? Maybe you have errors in the database, you have errors in the front end, you have errors in the logic. How do you normally prioritize which errors to handle first at which stage? Do you normally consider maybe I should first do a holistic view of the system? Maybe I should build a documentation? Maybe I should evaluate the criticality, whether the error is high, lower, medium? So, how do you normally prioritize which technical gaps to address first in your system? I usually prefer the documentation because the documentation part, that it contains all the information about your project. So, it has to start from the documentation, then bring out your final work. But when the documentation has some gaps, there will be misunderstanding between what maybe you're being reviewed, maybe on a panel, plus what you're showing the people physically. So, I prefer the documentation. It should be in line, then you create the physical part. Okay, so, how about, you know about SDLC, Software Development Lifecycle? Yes. So, you know we start with requirements gathering or requirements analysis, we design, then we implement, then we test, then we deploy, and then maintain. So, when you're doing, maybe, prioritizing which technical gaps to fix first, that lifecycle, at which stage do you want to fix that technical gap? What will be the appropriate stage? The appropriate stage is, like, maintaining. Because when your system is done, there's that one of testing. On testing, at first, because you have to build a system that has somehow, like, a scope. In the first stages, or the earlier days, that system might be working out properly. But as days come in, maybe the number of users is going on increasing. Then, when we go to the maintaining stage, it's when you see that, I had to build my system, having this certain scope. So, the system might be okay, but now, it's being employed by many people. So, it's scalability. Scalability. So, it comes at the maintenance, that's where you can prioritize that this is the area that I want to fix. Okay, thanks so much. Okay, we'll go to theme three. Theme three says, technical debt impact, the impact of technical debt. So, how would these unresolved gaps in your technical or in your prototype affect your software? Maybe in terms of reliability, maybe in terms of performance, or maybe in terms of interoperability. How would such affect you? These technical debts really disorganize the reliability of the system. In a way that, if at all you find out you missed out a point or something that requires, and the only thing that can help you is a certain language that you have to incorporate, you might find that incorporating that certain bit, it is going to lead to changes in some other stuff. So, the system that was, you might end up reformulating your system one way or the other, thereby affecting it. So, you have to study again what you have updated to find out that is it reliable, like the actual system that I had, that I had one thing missing like this, but now to put it there I have to make changes to it. So, it is affecting the performance of the system. Yes, sometimes the system has to be down, then you update it, something like that. That's one. Then can you provide a specific example in your programming where you interfaced the technical debt, or maybe it has affected the end user experience or the project outcomes? Technical debt. So, example, in my project or in your project? Yes, in the programming of your project. In the programming of my project. Yes. So, the technical debt, for example, in my prototype, I had to, as I have told you, that I want a system that you test for the blood group, then it records the information automatically. So, I came up with my, I had the prototype, but to integrate, to show that this information has to be carried on like this, because it is just a mere prototype. It came in that I had to find something that could really show the people that I want this and this. So, it involved me using some technologies. I used the temperature sensor to prototype that. But in my beginning, I didn't know how to integrate that system to build a prototype. I used the temperature sensor plus the software called Adreno to simulate that situation. So, in that period, to come up with the prototype, it involved me now go and I know how that Adreno works. So, that will affect the technical debt because… It involved me now to study that because I didn't know it. Yeah, I was building my system, but I'm coming to put it in the face of the people how it will work. It involved me going to know how Adreno works, programming in Adreno, because we just knew this programming may be on the website. So, that's a clear

example, by the way. So, we go to a thing called array debt repayment. So, are there any practices or strategies in place to encourage array repayment or fixing technical gaps? During the software prototype or development, maybe around the college or in your culture, are there any practices that you know that encourage you to fix your prototype technical debt as early as possible? Because we believe that fixing these technical debts as early as you identify it, maybe let me say at requirements gathering, maybe my code, my project need Adreno and I need to know prior knowledge before I even start implementing it. So, if you have that prior knowledge that if I fix this, it will not affect me in the future, that I have to come back and again read Adreno. So, are there any practices or strategies that enable you to do that? To find out that thing before the end? Or to anticipate that there would be a pause in your product and you need to fix them? This one can be, it is usually carried out by the early allocation of the supervisors. Because the final year projects are needed by the end of our third year. But they begin this activity as early as our first semester. So, they tell you, they always even allocate the time, the timetables, the evening times. Because my school, we don't usually study from 4 to… The day students do, but 4. So, after that time, they allocated project proposals. So, your person was… they allocated it, but nobody comes. So, that time they consider it that students should be using it for their projects beginning. Then, by the time that they… like three weeks or one month, the students have something, they allocate the supervisors. So, it is basically, students visit their supervisors, the supervisors reviews, just mere reviewing, not the final stuff. Now, the supervisor can help you out, tells you that there are loopholes. You say your system is doing this, but it provides maybe even unwanted information about the information you are telling us that you are revealing. So, by the time the students reach the other time, after using a full semester, just reporting, reporting and getting the errors and rectifying them. So, I think that is better. So, the supervisor is like a strategy where you can get this information right? Supervisors, maybe even trying out for the stakeholders that will be involved in your… Okay, so what are some of the incentives or mechanisms that you know that exist that encourage or motivate developers to fix their technical data as early as possible? Incentives that… Yeah, that normally motivate software developers to fix their technical gaps in prototypes as early as possible. The incentive that I would say is like, here at the university you have free platforms to do the research from, the labs, the internet is always free. So, if at all you find any technical gap and you have tried to look for solutions and you have somehow failed, nothing is not there on the internet. The internet is there, the labs are free, they are not that anytime you find it, there is no lecture you can use it to research anything you would like to know. So, there are really incentives. No payments, that is an incentive because when you find somebody outside, you might end up like, now I want to integrate this language in my system, but I have to always go to a cafe, pay. Yeah, pay, yeah, yeah. Yeah. So, for you… And remember, understanding is not first you can pay for one hour, study a little bit like this, you pay it even you have not understood, but here you come freely. To me that is an incentive. Okay, in conclusion, what are some of the insights that you believe you can include in such a framework or guideline whenever students have prototypes that maybe have minimal technical gaps in them? Yeah, I think students should be always oriented before coming up with those projects. Not me telling students, even though you are a computer student like this, not me telling a student to come up with a final project. They should be oriented because they have to bring out projects that work in real life, that solve real life problems. So, they should be taught how to come up with those ideas, think hard. What's coming up with a project is not something you just wake up that I'm going to do this. You have to think hard because you might even think about something, you call it your project, something is already in existence. You want to solve problems that we have solved a long time ago. Yeah. They should be taught the ways how to come up with the projects. Okay, so… I will be thinking, doing stuff, I think it is already in existence. So, that could be part of maybe career guidance. Yes, they should carry out that. Thank you so much for your taking your time and participating in this research. Your insights will come up in this framework and when this framework is out, we are done maybe the research and you come up with the guidelines. You will be contacted for validation because this framework has to be validated against the projects. If you follow these guidelines, can you help me come up with a solution that is technical? Of course, it can't be technical free because we know there is nothing like a system that is 100% free of bugs, right? But you have many more technical dates or fixing technical dates as possible in your project. So, just maybe give me your contact. Your number, your name and then of course… Okay. I can help. I'll call you.