Theme 1: Introduction

Question: Can you introduce yourself, the course you're doing, your role in the project, and briefly explain the project?

Participants

- <A> - Mugoya Dihfahsih

- <B> - Apollo Amollo

**Theme 2: Technical Debt Identification**

Question 1 In your experience as you're developing the prototypes, what are some of the stages you go through when developing a prototype, and what challenges have you encountered?

Participant (B) Response

-

- Challenges primarily include data collection difficulties due to uncooperative stakeholders.

Question 2 How do you become aware of technical gaps in your prototype?

Participant (B) Response Gaps are identified through testing with friends and peers, who often help with code reviews.

Question 3 What indicators suggest technical difficulties in your project?

Participant (B) Response Indicators include code that stops and starts without apparent reason, leading to code reviews and checks for errors.

Theme 3: Technical Debt Measurement

Question 1 How would you like to measure the problems you encounter in your prototype?

Participant (B) Response Measurement is achieved through code reviews with peers who identify issues and suggest improvements.

Question 2 Are there any specific tools you use to identify problems in your code?

Participant (B) Response No specific tools mentioned, but open to using tools like SonarQube or Step Size if available.

Question 3 How do you prioritize technical gaps?

Participant (B) Response Prioritization depends on critical points in the project, particularly those related to system security like the login process.

Theme 4: Technical Debt Evaluation and Impact

Question How do unresolved technical debt issues impact your project in terms of performance, reliability, and maintenance?

Participant (B) Response Unresolved issues lead to decreased productivity, increased costs, and a loss of interest in software development.

Question Can you provide a specific example of how technical debt impacted your project?

Participant (B) Response While data collection was challenging, assuming data due to unresponsive stakeholders might lead to problems in the future.

Theme 5: Early Debt Repayment Strategies

Question 1 What encourages students to address technical debt early?

Participant (B) Response Early implementation helps avoid time issues and motivates students to pay attention to their code quality.

Question 2 Are there strategies in place to encourage early repayment?

Participant (B) Response No specific strategies mentioned, but post-exam checks may be implemented.

Question 3 How can students be made aware of technical debt and its importance?

Participant (B) Response Education and awareness are essential, as well as grants and scholarships, which motivate students.

Question 4 What incentives would help students fix their debt quickly?

Participant (B) Response Education and grants, combined with knowledge of the long-term benefits of proper coding practices, would incentivize students.

Question 5 What insights could be added to a framework for guiding students in software development practices?

Participant (B) Response Early project implementation should be emphasized in the framework, as it helps students understand code and avoid shortcuts.

This restructuring organizes the interview transcript by themes and questions for easier reference and analysis.

FULL TRANSCRIPT:

Certainly, here is the full interview transcript:

<A>: Okay, basically, my name is Mugoya Dihfahsih, I'm doing a master's in software engineering. I'm on my second year and I'm going out researching software architecture. Basically, looking at the metrics students normally use in identifying technical data and their prototypes. Maybe to bring you to the technical data, these are the course questions that students normally incur in developing software prototypes using sub-optimal solutions. Sub-optimal solutions would be like taking shortcuts in their prototypes. And technical data is compared to financial data. The more time you take, the more interest you pay. And this is due to, it's both about poor design, poor documentation, poor testing of the code and it really slows down the students' motivation at work. Maybe sometimes when they fail to graduate because of the technical problem, and then as well as it slows down their productivity.

And this may also extend to the working world, where they take the same scenarios, same shortcuts to the professional world in software development. So I just want you to introduce yourself, your name, the course you're doing, the role that you have on the project, and then briefly overview of the project that you are doing.

<B>: My name is Apollo Amollo and I'm doing BIST. That's a Bachelor of Information and System Technology. Our project is more on the idea recovery system. Actually the idea was purely mine. And our work is that ideas are very important, that students keep using ideas around the campus. So we want to come up with a solution whereby once you use an idea, you can easily go to a platform, put in your details, and then maybe if we have it on the platform, someone put it on the platform, you have a message telling you where it is and you pick it from there. So basically that's what we're doing.

<A>: Basically I'm going to look at this research in 4 themes i.e 1. technical debt identification, 2. technical debt measurement, 3. technical debt evaluation, the impact and 4. technical debt repayment

<A>: Theme 1: Technical Debt Identification

In your experience as you're developing the prototypes, what are some of the stages that you go through when you're developing a prototype? And what are some of the challenges that you might have encountered along the way?

<B>: I think the challenge we have encountered the most was data collection.

<A>: Data collection?

<B>: Yes, because some of you went to, for example, an office, you're reporting to somebody with the name of students in Makerere, and they also can't give you time.

And yet you're developing something that can be used by everyone because in terms of scalability, you can increase the number and what. So you really want to do something that's going to be sustainable, it's going to work for everyone. So data collection becomes hard when somebody doesn't give you the information you want.

<A>: When you get that requirements.

<B>: Yes, requirements in general.

<A>: So how do you normally become aware of these technical gaps in your prototype?

<B>: Well, we try them with our friends because you're going to give someone to check it out. Because they think the more you interact with something, the more you get more covers and the better you get to discover the errors in it. So you become aware of these technical difficulties.

Because we are programming, you give someone to check through your code, yourself you also check through, but most times it's more with your friends, people in your discussion group checking through. And also there is a core you know is not in your group, but it's better than you.

<A>: So they do a code review.

<B>: Yes, First of all, they check the code. They identify that this could be a problem, this one could cause you this, or I think you change this to this. It is running, yes, but you could change this to this.

<A>: So what could be one of the indicators that you really believe these are leading us to a technical difficulty? This is due to a technical difficulty in your project. What are the things that you have learnt in your prototype that help you identify technical difficulties? Or in your prototype that you are developing, what are the real facts that show the effect of technical difficulties?

<B>: Your question is that what you see in your system that you lets you know that something is wrong.

<A>: Yes, because these indicators are more of a code duplication, code complexity, code ownership, code coverage that is testing. So what are those indicators that you might say there is a problem in my code?

<B>: Well, sometimes the code just doesn't run. And yet it has been running. And yet it's not running. And then the next time you check it, it's running. So basically, when you have such, because you can't really know where the issue is. So basically, they are just keep stopping and running at some point and yet you have not changed anything. So that's when you have to look for someone to help you check through.

<A>: So basically, the red flag is at the implementation level?

<B>: When you are implementing your code, when you have errors to implement the system, you just run the code. Because sometimes it does run, sometimes it stops and you really don't know what is the issue. So you have to just look for someone who either knows more or you share it with your friends and they just check through.

<A>: Theme2: Technical Debt Measurement

So how would you like to measure these problems that you encounter in a prototype when you are developing it? How do you know you might want to measure it? The measurement of these technical debt? The options could be code analysis, engaging stakeholders, Then there are code reviews, code analysis and code versioning, git. How do you know you might want to measure your code in a prototype to identify the criticality of your errors?

<B>: I do reviews with friends and what.

<A>: So that's how you measure your debt.

<B>: Yes, yes.

<A>: So that's why you get someone who is better than you and you go,

<B>: Yeah, yeah, yeah. First time you check through this.

<A>: So they tell you that your code is at this level, it is going to affect your system.

<B>: Yeah, yeah.

<A>: So which are some of the tools that you might use when you are doing this identifying problems in your code? What kind of tools? Maybe you are in Visual Studio Code and there are some tools that you are basically using to identify problems in your code.

<B>: I just do some visual things because if there is any error, then you just indicate.

<A>: So you have not heard of certain things like using SonarQube', step size to identify your errors in your