Summarized Transcript:

Certainly, here is the alignment of the themed questions and the responses from the student:

Theme 1: Technical Debt Identification

4. Question: 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?

- Response: The stages typically involve ideation, prototyping, and challenges include lack of access to premium tools and limited mentorship.

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

- Response: I become aware of technical debt through strong typing and unit testing, which help discover bugs in the project.

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

- Response: Indicators include failing tests, poor documentation, code duplication, and complex code.

Theme 2: Technical Debt Measurement

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

- Response: I would like to use a tool that scans through the codebase to identify areas for improvement.

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

- Response: I use version control tools like Git, but tools like Step Size, Code Climate, and SonarQube are also mentioned for measuring technical debt.

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

- Response: I prioritize based on the criticality of the bugs in the product.

Theme 3: Technical Debt Impact Evaluation

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

- Response: Unresolved gaps can increase maintenance time, reduce productivity, and lead to system crashes.

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

- Response: Bugs in a project affected the user experience, causing users to leave mid-journey, creating a negative image.

Theme 4: Early Debt Repayment

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

- Response:Not aware of any practices but practices such as education, mentorship, and sensitizing students about the impact of technical debt would be ideal.

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

- Response: Incentives include having enough time for project completion, which allows for fixing technical gaps early.

Insight:

14. Question: Is there anything else you would like to add or any additional insights you would like to share regarding technical debt in software prototype development?

- Response: Sensitizing students about what technical debt is and the impact it can have on projects is crucial, along with mentorship and informative sessions to distinguish between good and bad practices.

**Full Transcript**

Interviewer: Hello, can you hear me?

Student: Yeah, sure.

Interviewer: My name is Mugoya Dihfahsih, and I'm doing a master's degree in software engineering at Makerere University, and I'm carrying out research on student final year projects. The main objective of this research is to identify key metrics for measuring technical debt in student prototypes as they develop their software.

Student: Have you ever heard about the term technical debt? Maybe you could elaborate more, but from the basis of what I understand, technical debt is probably something lacking in regards to the technicalities. Yes, it could be maybe the skills required to do the final year project, maybe they do lack certain knowledge of certain things that would support them through the final year project.

Interviewer: Yes, that's a good understanding. Can you please tell me your name, your course of study, and the project you're working on?

Student: Yeah, I'm Hassan, doing computer science at Uganda Christian University, and I'm going to be a software engineer. I didn't exclusively think about that because there were a couple of ideas, and I was still weighing them to see which is better. But for starters, the ideation was around a marketplace for farmers.

Interviewer: Okay, so this marketplace is going to be free for farmers, right?

Student: Yes, please.

Interviewer: Great. Let's talk a bit more about technical debt. Technical debt is a term used to describe the consequences of taking shortcuts or making compromises during the software development process, similar to financial debt. It accumulates over time and makes it hard to maintain and update the software in the future. It represents the extra work and challenges that arise when code is not properly designed, documented, or tested. Do you follow so far?

Student: Yes, I'm getting that. When you take shortcuts, you're going to keep certain things, and these same things will come back to haunt you at a later time. It becomes a technical debt.

Interviewer: Exactly. So, in your experience, what are the stages involved in developing software prototypes, and what are the key challenges you've encountered related to tools, standards, frameworks, programming languages, and conventions?

Student: For me, it usually starts with ideation, then moves to prototyping, and challenges include lack of access to premium tools and limited mentorship. Access to mentorship can be challenging when your project is in a field your supervisor isn't specialized in.

Interviewer: How do you identify or become aware of technical debt in your project?

Student: I become aware of technical debt through strong typing and unit testing, which help discover bugs in the project.

Interviewer: What are the indicators or red flags that suggest there is technical debt in your processes or product?

Student: Indicators include failing tests, poor documentation, code duplication, and complex code.

Interviewer: How would you like to measure technical gaps in your processes or product?

Student: I would like to use a tool that scans through the codebase to identify areas for improvement.

Interviewer: Which are the current tools, if any, or measurements you would use to measure technical debt?

Student: I use version control tools like Git. Tools like Step Size, Code Climate, and SonarQube are also mentioned for measuring technical debt.

Interviewer: How would you prioritize which technical gaps to address first?

Student: I prioritize based on the criticality of the bugs in the product.

Interviewer: How does unresolved software gaps affect the quality attributes of your software prototype, like reliability, performance, and maintainability?

Student: Unresolved gaps can increase maintenance time, reduce productivity, and lead to system crashes.

Interviewer: Can you provide specific examples of how technical gaps have affected the project outcomes or the end-user experience?

Student: Bugs in a project affected the user experience, causing users to leave mid-journey, creating a negative image.

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

Student: Practices include education, mentorship, and sensitizing students about the impact of technical debt.

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

Student: Incentives include having enough time for project completion, which allows for fixing technical gaps early.

Interviewer: Is there anything else you would like to add or any additional insights you would like to share regarding technical debt in software prototype development?

Student: Sensitizing students about what technical debt is and the impact it can have on projects is crucial, along with mentorship and informative sessions to distinguish between good and bad practices.

Interviewer: Thank you so much for sharing your insights, Hassan. This information will help us in our research to minimize technical debt in software prototype development.