**Theme 1: Technical Debt Identification**

1. What is your role in the software development team and how does it contribute to the overall project?

Hussein Kizza serves as a team lead in the software development team. His role involves contributing to project development, providing leadership, and ensuring the team's collaboration and adherence to coding conventions and standards.

2. Can you describe the software prototype you are currently working on, including the type of product and its intended functionality?

The software prototype is primarily focused on web development, specifically in the areas of e-commerce, groceries, fast food, and beverages. Its primary function is to provide a platform for users to order groceries and food items online.

3. What are the main objectives and requirements of the software prototype?

The main objectives of the software prototype are to provide a user-friendly platform for online ordering, streamline the e-commerce process for groceries, fast food, and beverages, and offer a convenient experience for customers.

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?

Hussein Kizza highlights

* The importance of clear documentation, collaboration, and defining coding standards when developing software prototypes.
* He mentions the challenge of maintaining collaboration and conventions, especially in remote teams.

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

He suggests that technical debt becomes evident when the project reaches a stage beyond the minimum viable product (MVP). It's crucial to address technical debt once market-fit products are identified.

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

Indicators of technical debt include the loss of time, wasted resources, reduced morale and productivity, decreased product performance, and reduced product quality due to code shortcuts, lack of documentation, or unaddressed technical issues.

**Theme 2: Technical Debt Measurement**

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

They suggest that a clear measurement strategy for technical debt should be developed to assess the criticality of different technical debt components.

This includes determining which technical debt items to address first.

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

He mentions the use of Notion as a project management tool and documentation platform but acknowledges the need for more precise measurement tools for technical debt, particularly in prioritizing which items to address first.

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

Prioritization of technical debt should be based on criticality and impact.

Clear guidelines and metrics should be developed to help teams decide which technical debt components are most urgent to address.

**Theme 3: Technical Debt Impact Evaluation**

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

Unresolved software gaps can negatively impact software quality attributes such as reliability, performance, maintainability, and overall product quality. For example, performance issues may arise from unoptimized code, leading to reduced user experience.

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

Hussein Kizza shares an example of how a lack of performance measurement in a web application resulted in the application freezing when uploading images. This issue could have been prevented with proper performance measurement and management.

**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?

Hussein Kizza recommends

-Documenting identified technical debt items, prioritizing them based on criticality, and developing a clear repayment strategy.

Suggests introducing measures to encourage the team to address technical debt early.

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

Incentives can include

Suggests that a clear roadmap or staging area for addressing technical debt, where team members can track and address technical debt items as the project progresses.

This helps keep the team motivated and organised in tackling technical debt.

**Conclusion:**

14. 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? If yes, please share with me.

Hussein Kizza emphasizes

The need for a guideline or model that helps developers and teams prioritize technical debt items based on criticality and impact. Such a model would assist in managing technical debt more effectively and lead to better software prototype development outcomes.