Analysis and advise.

# Interview with the developers to Analyze the problems they are facing.

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Description | Author |
| 1.0 | 02-02-2023 | Initial Draft of the “Analyze and Advise document” | Lokesh Agnihotri |
| 1.1 | 02-8-2023 | Made the interview with the developers and added the problems they face | Lokesh Agnihotri |
| 1.2 | 02-15-2023 | Created a Scope for solving the problems | Lokesh Agnihotri |
| 1.3 | 2-20-2023 | Created prototype | Lokesh Agnihotri |
| 1.4 | 2-22-2023 | Got feedback on prototype | Lokesh Agnihotri |
| 1.5 | 2-25-2023 | Got another feedback and changed the approach on prototype CLI to app based | Lokesh Agnihotri |
| 1.6 | 2-28-2023 | Showed prototype to developers and they did not like the simple design | Lokesh Agnihotri |
| 2.0 | 03-02-2023 | Updated the prototype to show as a dashboard. | Lokesh Agnihotri |

## Abstract

The following document is a research analysis on the problem that the developers face and with the GitHub, GitHub repositories and while using common dependencies between their multiple projects.

I will be interviewing multiple software developers from different places and locations, in order to understand the problems that they are facing, and how would they like it to be solved.

I will also take a dive into their choice of User interface of the solution that they prefer.

After making multiple interviews with the many developers, I will make my advice on how the final solution should look like and share it with the developers for the feedback. During this process I will have to make my own choices and might have to turn down some suggestions while accepting others. The reason for declining certain choices will be based on what most people desire to have in the application and how it should look.

Since all the choices cannot be catered or fulfilled, I will be picking the most accepted chosen choice to solve the problem. However, the reason for making certain choices will also be provided with the prototype.

In the following document I will be sharing the multiple interviews I had. I will also share the link where the interviews can be found. The document will talk about the essence of the interview and things that stood out form the interview as what the developer wants to fix and what is his preference on the GUI.

## Questions:

1. Tell me about yourself.
2. Do you see yourself as a software developer, tester, or anything else?
3. Do you work with GitHub?
4. Do you have repositories on GitHub?
5. Do you know what dependencies are?
6. Do you encounter any problems with respect to dependencies in your projects on GitHub?
7. What problems do you encounter?
8. How do you solve them if you solve them?
9. What is the problem that you still face and would like to be solved.
10. How would you like to see the software that solve your problems? In CLI or Web based or browser extension or any other suggestions?
11. What is your preference on UI?
12. Do you prefer the UI or just notification when something is wrong with dependencies?
13. What language do you most work on.

## Prototype: https://vjimjj.axshare.com

## Criteria Table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Criteria | Weight | Rating Scale | Ignas | Wouter | Pieter | Mohit | Bastiaan | Valerian |
| Knowledge of Dependency | 10% | 1-5 | 3 | 5 | 1 | 5 | 5 | 1 |
| Experience with dependency conflicts | 25% | 1-5 | 3 | 5 | 1 | 4 | 4 | 2 |
| Ability to identify and solve dependency issues | 25% | 1-5 | 2 | 4 | 1 | 4 | 3 | 1 |
| UI knowledge | 25% | 1-5 | 3 | 2 | 5 | 3 | 3 | 5 |
| New Suggestions to solve the dependency problem with vulnerability | 15% | 1-5 | 2 | 3 | 5 | 3 | 2 | 1 |
| Total Score | 100% | -- | 2.6 | 3.7 | 2.6 | 3.7 | 3.3 | 2.25 |

## Interview One: Ignas Apsega, to be graduate Fontys Student in Semester 8.

<https://photos.app.goo.gl/uRpDeLzfJ9Nt7oUv9>

He uses GitHub and uses dependencies.  
He also use common dependencies in multiple GitHub repositories.

He does not update the dependencies so it is possible that his projects will not work if the dependencies are not updated or are not suitable together.

**Problem he faces**: He gets email from GitHub that his dependencies should be updated and or are vulnerable.

**What he wants**: Double click on icon, application launches, and he then enter the username, the application then creates a graph of common dependencies between GitHub repositories and see the vulnerable ones in different color.

**Preference on UI**: He prefers the icon and the application to launch as User Interface.

He does not prefer the CLI.

**Feedback on Prototype (1):**   
Good: Simple.

Bad: Make it somehow explicit that the 3 charts of critical, healthy, minor are my projects of GitHub.

Improved: Forgot to add - 'Back' button in the healthy and minor views. The sum of percentages of critical, healthy, minor is above 100%.

Does it solve your issue: It does.

**Final Feedback on prototype (2)**

## Interview Two: Wouter Pennings, semester 6, Developer and a teacher

<https://photos.app.goo.gl/XnKiHsYGEkDRSzpw9>

He likes to decrease the use of dependencies between his projects.

**Problems he sees:**  Malicious code in the dependencies can put the whole project in the jeopardy.

Dependencies can go very deep and can have threats in them in case you use the nonstandard dependencies.

**What he wants:**  How fast can you find a vulnerability?

How fast can this information be visualized and how fast can user be notified? Maybe help user to show where it is and how could it be fixed?

He believes that the problem generally lies in the sub-sub-sub-sub dependencies.

**Preference on UI:** He prefers a CLI tool. But he does not want to run this tool every day.

He wants a notification only.

Maybe as an extension that sends notification automatically if there is a problem.

So, he does not prefer having a UI for the tool.

But when he gets a notification, he prefers a notification with the picture showing the trouble in vulnerabilities.

## Interview Three: Wels Pieter P., Coach at Open Learning

<https://photos.app.goo.gl/wN4PbApzMNY8uutu9>

He is a designer. Not exactly a software developer.

**Problem he sees:** Bad handled code, malware etc. in a dependency.

**What he wants:** To be able to visualize problem in dependency.

**Preference on UI:**  He would like to see vulnerabilities in color coded based on how hardcode the threat is.

Also give feedback when everything is correct. Like as score.

**Initial Feedback:**

Pieter shared no feedback after asking multiple times. He just ignored the messages.

Graphical user interface, application

Description automatically generated

## Interview four: Mohit (Product Owner, Signify, Eindhoven)

<https://stichtingfontys-my.sharepoint.com/personal/394616_student_fontys_nl/Documents/Opnamen/Meeting%20with%20the%20Sinify%20Developer-20230217_160919-Meeting%20Recording.mp4?web=1>

Product owner and lead developer. Uses multiple languages like C, C++ and python.

Team of mix developers and new developers come in.

Difficult to visualize how different modules are interacting with each other.

Hard to give this information to new employee.

This is done manually and is never correct. The diagram changes everyday when someone add new module or new dependency is added or something else.

Hence it is hard to visualize the dependency between module in system and keep it up to date.

They do not have the tool.

**Problem to solve:** They want to have a tool that shows the updated view of modules and dependencies used there. They do not monitor their dependencies.

**What he wants:** They want to have a tool that shows modules and dependencies and stays updated.

**Preference on UI:** They do not care about GUI. They do not care about GitHub as well.

They do not care if the tool is script or a batch file.

They do not want a GUI since they are bad to be automated.

They prefer a CLI.

They want to see the dependencies on Demand using the CLI. And look at it when needed.

They do not want browser extension as they use machines that do not even use the browsers.

**Feedback on prototype**:

Waiiting..

## Interview Five: Lichtenbelt,Bastiaan B.O.

<https://stichtingfontys-my.sharepoint.com/:v:/g/personal/394616_student_fontys_nl/ETu-bYu74FNKvh4AxRc4aPEBzzWE_Cd4igFGrLf-KDsLkw?email=b.lichtenbelt%40student.fontys.nl>

User faces the problem of having same dependencies with different code inside them which can create problem. Different dependencies with same name.

UI preference: A script

**Feedback**: 1) Like the original graph (the one that coach rejected.)

2) feel like the graph should be added to the current version to locate critical position for errors, this will help locate the most likely place for major errors or problems.

3) the graph could be limited by setting up a depth limit, or a way to hide areas of the graph

4) The bar graph is useful to see how much you messed up, it's useful that it shows what is in critical state or healthy

**Final Feedback on Prototype:**

## Interview six: Valerian Mirzac

<https://photos.google.com/photo/AF1QipONbzdCgH2Jt5k4kGVxjMtN5yMk1jCEJ5aLcEUY>

Valerian is not very adept with using the dependencies or GitHub or maintaining the dependencies as he is mostly from the hardware side.

Problem he faces: It is hard to keep the structure and dependencies organized when the software grew.

Solution he wants: He wants a solution that can help him visualize the dependencies and it will be great if it can refer the user to the documentation.

**Initial Feedback on Prototype**: It is too simple, and it is not good. Does not look good at all.

**Final Feedback on the prototype:**

## Conclusion

Ignas exhibits a strong understanding of software dependencies and their associated conflicts. While he has yet to address such conflicts directly, he displays an adeptness with the tools available to identify and monitor potential vulnerabilities within his GitHub repositories. Additionally, Ignas exhibits a solid grasp of user interface design, as evidenced by his ability to clearly articulate his desired features and functionality. However, he did not offer any novel solutions or suggestions to improve the project beyond the immediate issue at hand.

Wouter exhibits a comprehensive understanding of software dependencies and demonstrates considerable expertise in resolving conflicts through a variety of methods, as highlighted in the video. Additionally, Wouter displays a solid understanding of user interface design, although his suggested UI solution was not well-suited to the current project requirements. Nevertheless, he quickly grasped the problem at hand and was able to provide valuable insights and ideas to address the issue effectively.

During the recent interview, Pieter's contributions were notable for their strengths and weaknesses. While he demonstrated a lack of expertise in managing dependencies and how to address them, his suggestions for improving the user interface (UI) of the application were insightful and valuable. He did offer a unique perspective on how to visualize dependencies within the application. Specifically, he suggested that dependencies could be rated on their health, providing a visual representation of the components that are functioning correctly and those that require attention. On the other hand, Pieter's contributions to the discussion about UI were impressive. He proposed innovative ideas for streamlining the user experience and making the application more user-friendly. His suggestions were backed up by research and analysis, and his ability to communicate his ideas clearly was commendable. He displayed a keen eye for detail.

During the interview, Mohit demonstrated an impressive level of knowledge regarding GitHub, repositories, and dependencies. He emphasized the importance of visualizing dependencies and the potential issues that may arise if dependencies are not managed properly. One specific challenge that Signify faced was explaining the code structure to new employees, including which dependencies were related and how they developed as the software grew. Mohit tackled this challenge with confidence and expertise, sharing his deep understanding of the potential issues that can arise with poorly managed dependencies. He was able to provide clear direction and vision for the application.

During our recent discussions, Bastian demonstrated an impressive level of knowledge regarding dependencies and related issues. In particular, he suggested a new problem that needs to be fixed: that during team development, users can sometimes have dependencies with the same name, but different code inside them. This creates confusion and inefficiencies and can be a significant challenge for our team.

Valerian demonstrated good knowledge about his understanding of the dependencies and problems that can create when the dependencies increase with the size of the software. He mentioned that it is hard to keep a track of all the dependencies as software grow and it will be nicer to have a tool that can visualize it as well as refer to the documentation of the dependency that is affected.

Valerian showed great insight in the UI and suggested many changes that can make it more user interactive. One idea that stood out was to immediately show the graph of the GitHub user as soon as his username is entered in the tool, so that an immediate action can be taken, and this idea saves many click and make job of the user quick.

After conducting interview with multiple users and developers I come to conclusion that it is important to have a tool. The tool should run every certain period and show the affected repositories. When the user can login to the tool and share the username of the profile he wants to check, the first thing he needs to see is how many repositories are in critical threat. This way he can prioritize his work to update the repositories dependences.

Once he clicks on the graph (critical, minor, healthy) he can then look at the repositories that are critically affected. And then got to the GitHub library to solve the error.