Appendix

Below are the responses from the completed questionnaires. The *italicized* text indicates the labels assigned to the answers during the thematic analysis.

Questionaire from Participant 1

Question 1: Could you briefly introduce yourself and share your background in software development?

I have been working as a software engineer for over 15 years now. Most of these years i spent as a consultant in various roles, being a full stack cloud engineer, backend developer, software coach and architect.

Question 2: Please briefly describe the system that you are currently working on

Currently i work on a system for a large home improvement chain in the Netherlands. Specifically the customer loyalty program. One of the most important things we enable is the loyalty program: a customer can save points when they buy something, and these points can be spent later on for example a discount. Other things include the "my account" stuff: letting customers manage their personal information, receipts, online orders, their paints and so on.

Question 3: How do you envision integrating the proposed process into your day-to-day work?

The company i work for has several teams working on various systems, and to encourage cooporation across all teams there are several guilds responsible for different things. One of the guilds i am part of is the architecture guild, where we aim to have a cohesive architecture and guidelines for all teams to follow. I can see views generated by applying the process and using Architecture Lens being a great aid in gaining insight in the various systems during these architecture guild meetings, while also providing a base for discussion *(alignment with design, improve understanding of system)*.

Question 4: Considering our proposed process, how do you think the ability to scope views might impact your overall comprehension of your system architecture?

Some parts of the system are more critical then others, so i can definitely see that creating scoped, more narrow views of areas of the system that are more business critical then others will be very valueable.*(focus in critical parts of system, scoped views)* It is very important to me to reduce my cognitive load, so i can focus on the stuff that really matters to me.

Question 5: In your opinion, what effect could our proposed process of incorporating diagram generation as part of a continuous process have on your system's architectural documentation?

Currently, the architectural documentation is being maintained by hand by someone. All the stereotypical problems are there: this is a person that does not work on each of these systems, so he needs to get the information 3rd hand and trust that it's accurate.*(Lack of human error)* The documentation does not get updated all that often, which further reduces accuracy. Also, it is hard to find: it's hidden on a sharepoint drive somewhere, and you can never be sure you have the correct version. Finally, it is a single image with

all components on it, which makes it impossible to parse at first glance.

I am a big proponent of generating documentation and publishing it as part of the deliverables of a project. Just like you would package up the source code in maybe a deployable docker image, generating and publishing the documentation that goes with that deployable would make a lot of sense to me. This would mitigate a lot of the issues described above. Therefore, Applying to the proposed process and continiously generating views of the system using Architecture Lens which can be stored on github can have a huge impact.

Question 6: What are your thoughts on showcasing differences between the "main" branch and your feature branch when creating a pull request using our proposed process?

Our pull requests are mostly reviewed in the diff view, to see what code got removed and what code got added. Just by looking at code changes it can be challenging to really understand the consequences these changes might have. So, just like how we rely on a CI pipeline to verify the changes did not break the build, having something as part of the pull request that shows the impact on the architecture will help in providing a better code review.*(alignment with design)*

Question 7: How well did our proposed process support your workflow and assist you in maintaining up-to-date architectural documentation?

Very well. Having several diagrams generated as part of the build pipeline feels right at home with generating and publishing the other artifacts. We can even keep track what version of the architecture is in production, and which is on staging to really have a clear picture of what is going on.*(improve understanding of system)*

Question 8: How well did our proposed process support your workflow and assist you in improving the comprehensibility of your architectural documentation?

It does take some experimentation find scoped views that are applicable for the system as a whole, and creating scoped views that really zoom in on the part that is under development. But, dialing in the diagrams also helped me understand what parts i cared and what parts were less relevant, which in turn also increased my understanding of the system as a whole. *(improve understanding of system)*

Question 9: Do you have any suggestions to improve the tool to make the diagrams even more comprehensible or valuable?

Expand the process to also be able to include connected systems. In todays microservice world, it would be awesome to generate documentation that shows how the microservices are connected to one another. Including REST calls, topics, queues, databases, etc.*(improvements)*

Question 10: What additional features or future research directions would you like to see explored in Architectural Lens?

- Make a VSCode and/or PyCharm plugin that can show the diagrams directly in the IDE.
- Make the diagrams interactive, allowing users to click through and zoom the diagrams. Additionally, provide the capability to save scoped views in the configuration.

- Have the tool analyze the architecture and give suggestions on how to improve based on best practices.*(improvements)*

Questionaire from Participant 2

Question 1: Could you briefly introduce yourself and share your background in software development?

I'm a senior developer with over 5 years of experience in software development, mostly working on Python applications. I've been involved in various projects, mainly involving web development and data analysis.

Question 2: Please briefly describe the system that you are currently working on

At the moment, I'm working on a web-based data visualization platform, allowing users to explore and analyze large datasets through interactive visualizations, like generating heatmaps or creating custom graphs to compare data points.

Question 3: How do you envision integrating the proposed process into your day-to-day work?

I attended a workshop on Architectural Lens, and it was pretty straightforward to set up and start using in my own project.*(easy to setup)* Over the past week, I've been integrating it into my daily work, and it's been helpful in automating the generation of architectural diagrams which i and my colleagues understand and keeping them up-to-date.

Question 4: Considering our proposed process, how do you think the ability to scope views might impact your overall comprehension of your system architecture?

Although i already knew the system quite well, after using the ability to scope views in the tool while developing as the process states, my understanding of the system architecture has improved.*(scoped views)* It allowed me to focus on specific parts of the system that are most relevant to my work, like the components responsible for data processing or user authentication.*(focus in critical parts of system)*

Question 5: In your opinion, what effect could our proposed process of incorporating diagram generation as part of a continuous process have on your system's architectural documentation?

Incorporating the diagram generation into a continuous process has helped ensure that our architectural documentation stays up-to-date and accurately reflects the current state of the system. *(up-to-dateness)* This has made it easier to troubleshoot issues or onboard new team members.*(onboarding)*

Question 6: What are your thoughts on showcasing differences between the "main" branch and your feature branch when creating a pull request using our proposed process?

After trying the feature that showcases differences between the "main" branch and my feature branch when creating a pull request, I found it useful in identifying potential issues or inconsistencies in the architecture, For instance, I was able to spot a new feature inadvertently introducing a circular dependency, which I addressed before merging the changes.*(spot errors with diff view)* Additionally it was useful just to understand what changes in your system before allowing it to be part of your code-repository, the color highlighting made that very easy to see. *(improve understanding of system)*

Question 7: How well did our proposed process support your workflow and assist you in maintaining up-to-date architectural documentation?

The process and tool has streamlined my workflow and helped me maintain up-to-date architectural documentation.*(up-to-dateness)* I no longer have to spend as much time manually updating diagrams, allowing me to focus on development tasks. The documentation has however not yet been accepted by my bosses yet, so it is only me using it right now, I think if it is to be really useful, my colleagues need to do something similar.

Question 8: How well did our proposed process support your workflow and assist you in improving the comprehensibility of your architectural documentation?

Generating customized views tailored to specific aspects of the system, like showing only the components related to "data storage" or "user management", has made it easier to understand the architecture.*(scoped views)* But again, I think its important to note, that for this process to really take effect, it is important that the entire team atleast adopts the idea, much like we have with unit testing, which I was informed that the process was inspired by.*(team adoption)*

Question 9: Do you have any suggestions to improve the tool to make the diagrams even more comprehensible or valuable?

One suggestion for improvement might be to add more customization options for the diagrams, such as different visual styles or layout options, to further enhance their readability and appeal.*(improvements)* For example, I'd like to have the option to color-code components based on their purpose or layer in the architecture.

Question 10: What additional features or future research directions would you like to see explored in Architectural Lens?

I'd be interested to see Architectural Lens and the proposed process extended to support other programming languages and development environments. It would also be useful to explore ways to integrate the tool more seamlessly with existing development tools and platforms, like incorporating it into popular IDEs or version control systems, but it is nice that it is currently available through pip.*(improvements)*

Questionaire from Participant 3

Question 1: Could you briefly introduce yourself and share your background in software development?

Master's degree in science and engineering, 4 years of experience working as a software developer on Python, Scala, Java projects in banking and game development.

Question 2: Please briefly describe the system that you are currently working on

It is dockerized Python system designed to transfers files over SFTP as either client or server. The users of the system are external banking partners that all have different demands on file formats and encryption standards making configurability and modularity a big focus of the system.

Question 3: How do you envision integrating the proposed process into your day-to-day work?

I see two main modes of operation:

1. As a local tool for prototyping sweeping changes to the system (large refactorings or new modular functionality).*(large refactor)* The tool would be run often on my local machine, as with unit tests.
2. As a collaborative tool to ensure that other team members align with my design and that my changes don't introduce undesired coupling.*(alignment with design)* This would take place in a space such as a pull request.

Question 4: Considering our proposed process, how do you think the ability to scope views might impact your overall comprehension of your system architecture?

I think scoped views is what makes the tool usable for a system of moderate to high complexity.*(scoped views)* Being able to pick apart the system and focus on one specific area at a time helps when onboarding people*(onboarding)* (as that process is naturally gradual) and also when discussing how specific parts of the system are implemented. If you could not scope the views, and only had a top-level view, it would not be as helpful since details would easily be missed.

Question 5: In your opinion, what effect could our proposed process of incorporating diagram generation as part of a continuous process have on your system's architectural documentation?

Documentation tends to have a higher quality when it is often iterated upon. Generating an architectural view of a system with every change vastly increases the odds of the documentation being correct and precise*(up-to-dateness)*. I think it would also lower the barrier for people to start engaging in maintaining proper documentation, which further spreads knowledge of a system through a team of developers and designers.*(easy to setup)*

Question 6: What are your thoughts on showcasing differences between the "main" branch and your feature branch when creating a pull request using our proposed process?

This is the most exciting feature for me as it is, due to the following reasons:

1. Checked in and audited in version control.
2. Automated and integrated into build pipelines.
3. Automatically reflecting the changes of the raised pull request (PR), providing a mutual understanding and solid basis for discussion between reviewers.*(alignment with design)*
4. Automatically updated upon further code changes based on the evolution of the PR, providing a solid trail of changes for new reviewers.*(evolution of design in PR)*

Question 7: How well did our proposed process support your workflow and assist you in maintaining up-to-date architectural documentation?

The proposed process required minimal modification to my workflow and was easy to integrate.*(easy to setup)* The updates to the architectural documentation are naturally iterative and updated often, granting

the documentation a higher degree of confidence. In addition, as the documentation was generated frequently, it assisted me in spotting architectural errors before commiting them to the codebase, which in the long run may help my system quality improve, as it is less complex and has less architectural errors.*(helps spot architectural errors)*

Question 8: How well did our proposed process support your workflow and assist you in improving the comprehensibility of your architectural documentation?

Following the process and applying Architectural Lens helped me gradually gain a more fine-grained understanding of the system and its components.*(improve understanding of system)*

Question 9: Do you have any suggestions to improve the tool to make the diagrams even more comprehensible or valuable?

Searchable diagrams would be of large help as it aids in breaking down top-level diagrams (which are still useful if they can be navigated).
Circular dependencies could probably be highlighted with thicker arrows.*(improvements)* More often than not you do not want them, if the tool can identify them that'd be a plus.

Question 10: What additional features or future research directions would you like to see explored in Architectural Lens?

It'd be nice to see what data types flow between modules most often. This is perhaps quite dependant on the language being analyzed, but seeing if certain types leak very far outside of the module where they are defined might indicate a leaky abstraction.*(improvements)*

Questionaire from Participant 4

Question 1: Could you briefly introduce yourself and share your background in software development?

Advanced Higher Vocational Education Diploma in Software Development and have been working in software development for a little bit more than 7 years as a developer and tech lead across various platforms and languages.

Question 2: Please briefly describe the system that you are currently working on

I work for a large financial institute with tightly coupled Python systems, maintained by several teams serving, which handles all the business needs for accounting and book keeping purposes.

Question 3: How do you envision integrating the proposed process into your day-to-day work?

I can see it being extremely useful in the change process to evaluate impact and scope of proposed changes to the system, which otherwise usually relies on manually updated (and often outdated) documentation or the knowledge of experienced developers.*(alignment with design)* If we could more easily identify our change impact, which is made difficult by having many integrations, we would could ease the burden of senior developers but also in turn more efficiently onboard new developers, speeding up the change process at the

same time and hopefully lowering our risks related to changes.*(helps spot architectural errors)* It could also similarly ease the onboarding of new colleagues by simply serving as an alternative to exploring the system in an illustrative way rather than by code.*(onboarding)*

Question 4: Considering our proposed process, how do you think the ability to scope views might impact your overall comprehension of your system architecture?

The scoped view functionality is key for the process serving as a multi-purpose tool.*(scoped views)* With scoped views it could be used for all steps in the development process. A low level scope can be used by the developer immediately to identify unintended side effects.*(helps spot architectural errors)* Whilst a top level view allows designers, testers and architects to properly evaluate module coupling.*(improve understanding of system)*

Question 5: In your opinion, what effect could our proposed process of incorporating diagram generation as part of a continuous process have on your system's architectural documentation?

In my experience, the only documentation that always stays up to date is the one that is done automatically and when it's integrated into the change process this ensures that our documentation always up to date and factual, automatically.*(up-to-dateness)* In turn I would hope this would further incentivize developers to take part in documentation.

Question 6: What are your thoughts on showcasing differences between the "main" branch and your feature branch when creating a pull request using our proposed process?

This is essential for the tool to be used in the change process and offers a lot of value for stakeholders as it can be used to more accurately measure impact and risk for proposed changes.*(helps spot architectural errors)* This means it could also be used to troubleshoot or better understand divergences in system output or state after a change by comparing previous views to the current one.

Question 7: How well did our proposed process support your workflow and assist you in maintaining up-to-date architectural documentation?

Because it was so easy to integrate I was up and running almost immediately and started generating architectural views based on live changes.*(easy to setup)* As it was done automatically from the codebase it was less prone to mistakes or inaccuracies and in turn also earlier in the development process.*(lack of human error)* Traditionally documentation would be done, by hand, after the change was in place but now we could fully explore architectural differences in a much earlier stage.

Question 8: How well did our proposed process support your workflow and assist you in improving the comprehensibility of your architectural documentation?

The proposed process allowed me to in an illustrative way explore our system in an up-to-date fashion.*(up-to-dateness)*

Question 9: Do you have any suggestions to improve the tool to make the diagrams even more comprehensible or valuable?

Perhaps the views could distinctly showcase different types of dependencies (e.g. circular dependencies, bi-directional dependenices, uni-directional dependencies).*(improvements)*

Question 10: What additional features or future research directions would you like to see explored in Architectural Lens?

If it could also export in some raw data format, rather than views, users could more easily parse it with code for analysis purposes depending on use case, allowing even more flexibility for the tool.*(improvements)* It would also mean that users could, potentially, build their own presentation/view layer for the architectural views which potentially would fit their own use case better.

Questionaire from Participant 5

Question 1: Could you briefly introduce yourself and share your background in software development?

Bachelor in software engineering and Software developer/operations engineer with 4 years of experience working in python backend systems

Question 2: Please briefly describe the system that you are currently working on

Web based monitoring system using precision temperature and humidity sensor hardware to monitor and validate manufactoring processes for food and pharma industries

Question 3: How do you envision integrating the proposed process into your day-to-day work?

Implementing views to current documentation and most valuable is hooking renders up to PRs and seeing the diff views.*(Difference views)* After attending the workshop, it was easy for me to get started using the process and tool.

Question 4: Considering our proposed process, how do you think the ability to scope views might impact your overall comprehension of your system architecture?

I think it'll mostly help me with keeping eye on maintability of the system. To easily spot if any new dependencies on a new PR has arisen and make sure to avoid any unwanted coupling from happening before merging.*(helps spot architectural errors)*

Additionally, when it comes to overall team efficiency and onboarding, I think that visual learning is an incredibly powerful way to quickly understand and get an overview of a new project. Having pre-generated and precise views of our system for onboarding of new employees could help with reducing man hours used on onboarding both for the new employee and the senior developers.*(onboarding)* While a new employee should never be afraid to ask questions to understand a new system, this also takes time off the senior developers having to explain the system, which in itself can be hard to do. Being able to easily divide your project into subsystems makes it easier to explain, and possibly reduce questions to senior devs how the system works. *(improve understanding of system)*

Question 5: In your opinion, what effect could our proposed process of incorporating diagram generation as part of a continuous process have on your system's architectural documentation?

Having this as the source for documentation makes sure its always updated, and no manual work apart from making new views when needed helps reducing time spent on documentation (which most developers prefer not to spend time on).*(up-to-dateness)* We've tried using automated tools in the past, but have not found any option which gave us anything of value (entire system views are hard to understand, only found tools which can do this).*(automation)* I like that you can create specific views in the json file, meaning that i only have to create the json file once, and then i can always generate the updated views when changes are made and save them for others to see.*(scoped views)*

Question 6: What are your thoughts on showcasing differences between the "main" branch and your feature branch when creating a pull request using our proposed process?

As previously mentioned, seems like a very nice way of keeping maintainability by visually being able to spot dependency changes, I tried using it with my colleague to spot errors in our pull requests, it feels like a clear visual way of validating your architecture continiously.*(Difference views)*

Question 7: How well did our proposed process support your workflow and assist you in maintaining up-to-date architectural documentation?

This reduces time spent on updating system diagrams, which was done once in a while. With it happening automatic, this ensures you have an up to date diagram, and developers don't have to question whether diagram is incorrect either from human errors or just not recently updated.*(lack of human error, up-to-dateness)* I have started storing some of the render views in my github, where they are relevant, it would be nice if you could specify where they are saved and have a github action in addition which refreshes them each time a commit is made.*(improvements)*

Question 8: How well did our proposed process support your workflow and assist you in improving the comprehensibility of your architectural documentation?

Being able to segment into views helps with focusing on what part of the system you're working on now, instead of seeing the whole system in itself and get lost in an enormous diagram. *(scoped views)*

Question 9: Do you have any suggestions to improve the tool to make the diagrams even more comprehensible or valuable?

I could see it being very handy to have renders being interactive. So you're able to see the entire system. From there on click and choose views you'd like to see, highlighting the once you've chosen, and grey out/lower opaque of the modules you haven't chosen in the view. Additionally as previously mentioned, being able to give render views a path where they get saved and then have them refreshed on each commit would be nice to keep them up to date, after saving them.*(improvements)*

Question 10: What additional features or future research directions would you like to see explored in Architectural Lens?

This being implemented to other git platforums such as Azure Devops, gitbucket etc. *(improvements)*