Participant 11

SUMMARY KEYWORDS
people, issue, problem, software, pretty, team, code, colleague, quality, tests, fix, work, initiative, person, write, improve, agree, innovation, bit, read

SPEAKERS
Researcher
Participant 11

Researcher 00:16
Good morning, Participant 11. Can you hear me? Hello. Good morning. How are you?

Participant 11 00:28
Well, I'm well, I'm just trying to get by.

Researcher 00:33
Okay, take your time.

Participant 11 00:38
They just got a phone call on it. Would you like me to start my camera to add those to the meeting?

Researcher 00:51
No. It's not needed. It's entirely up to you. We don't need to see you,

Participant 11 01:01
Then I wouldn't bother us. It will take more minutes.

Researcher 01:04
Okay, sure, no problem. I'd like to thank you for the opportunity of the interview. I really appreciate specially this Sunday morning.

Participant 11 01:14
I think it's for you.

Researcher 01:17
Okay, I do have a lot of questions. And thank you for providing some answer by email. And I can see also, you have some good examples, which I like very much. I'd like to kick off the interview. So we can start discussing these examples. I'd like you to start with a brief introduction. Just tell us who you are. Mainly your education, your experience in software engineering.

Participant 11 01:50
Cool. So from the latest I have a master's in software engineering, then a bachelor's in computer science. And also did High School focused on AI information. Set study circumstance again. Yeah. And also when to contest only get us all support. That would be my educational background. Currently, I'm employed at different instruments as a software engineer. And being employed at software company with a total of three years.

Researcher 02:38
Okay, fantastic. The answer you're provided is based on your current team, the team you're working for at the moment?

Participant 11 02:40
Oh, yes.

Researcher 02:48
Okay. Yeah. So are you using Scrum or some sort of Agile method now?

Participant 11 02:55
So until let's say the past month we've been using Scrum come but the thing is, we're not doing it by the book. Okay, so it's a bit adapted to work with our the company culture and all the way we have the really structures also perfect. So we started pure Scrum and because we have very tight project now for 10 weeks, there was a proposal to go on Kanban. So we combining both.

Researcher 03:31
So yeah, this is a good opportunity to tell me how did you tailored Scrum to your own particularity? What did you change about it? Actually, it's a framework. It's a flexible framework, it doesn't dictate to you how to work so it is it is acceptable the team's tailor it to their own needs.

Participant 11 03:58
Well I worked so long in this method, I'm not saying anymore What's original and what's not. So I'm just gonna tell you how we do it now. Yeah. So we don't have demos anymore only ad hoc at the end of the iteration when all the project owners and let's say management, stop our own companies or free to review it. We do have some sync ups, not always with the people in the United States to let's say maybe once a week or once every two weeks depending on how free they are. To sync up on work that has been done. And if anything has come up, mostly they do not supervise us, but we go to them in case we have questions. So if you have no questions, nothing to sync up with the US team at all. Then we will have a backlog that's as usual. We did not have a spring meeting? Um, no. Okay, it was, and we call it planning, we should call it planning. So at the beginning of a sprint, we just plan everything we estimate. And that's it. We usually do not reorder the task only if either no, they're blocked or something very important has come up. The retrospective, yeah, with all the end, as usual. And we do have the daily standups. So it's pretty close. But sometimes we do alter from it either or, for example, we might extend the sprint, or we might consider like the cycle is extended developer, we don't, we don't officially close it. So okay, maybe by we might close the stories, but make some bugs to make up for it. Or we negotiate with the management to extend the project, or whatever to, to make up for issues that came

along. The roles might not, let's say it separate, maybe they collect for example, in case we don't have a scrum master or project owner or other on the manager might do them all. So highly dependent on where we are, for example, I have been part of two new teams each time I started with three people and ended up with 12. Each time, my manager starts with all the core roles for himself. And while we get settled, he started to spread them around. So yeah, it's highly dependent on where we are at the time.

Researcher 07:29
That's acceptable. That's not a major deviation from Scrum. Actually, it's that those little tweaking here and there are acceptable and even Scrum encourages those. Okay, thanks for sharing that. I will ask you a few brief question about the team just to get a little bit of context and we can move to the to the to the questions. I've sent an email. Okay, what is the size of the team?

Participant 11 07:59
So right now? I mean, I know. So it was the last year, it was pretty crazy. So I gave you example from both of these. So I guess a conversation explainable. The first team started with me and the other colleagues. We were challenged with improving the parent site on our backend service. For that, we managed to finish it after let's say three months. Afterwards, we were tasked with the front end application, again, for another few months, or like three or four or five. I don't remember at the end of the first part, the second part of the project for like full year. So in the second part of project we were a few months alone, then as it grew in scope of April, we were allocated towards the end of the past year, something like 90 members either newly hired or transfer because of rebalancing. Like each year we have a rebalancing the company people can choose from overall. So we got our repeated like people we got extended a lot. And we had a lot of parallels. That was amazing because otherwise we wouldn't have managed to finish what they what they wanted us to. And after we finished it finished the UI projects. It was left let's say with that team, and me and the two other colleagues that are in the core team at the beginning. So let's say the nine people that came with remained in that team and the core of that was me and the other two left to start the neither of you want which is handling for the secure right now. So it, they may read shows that they do not want to disturb the whole other people, we just we just got settled. And because like we are the ones that usually get the handle on faster, they just took us out and put us in a new team to bootstrap it. Again.

Researcher 10:18
Okay. So are you are you I understand that change the team dependent on the need and dependent on the project status cetera? So are you always cross functional? Or you always you always cross functional?

Participant 11 10:34
So all of us are developers and QAs And most of us know but this could potentially change in the team if needed.

Researcher 10:44

Okay, fantastic. I know, because we talked in emails. I now what type of software do you develop? But can you share with me? What type of software do you develop? An to what degree are you self-managed?

Participant 11 10:56
So we develop an asset management software. So for the same electronic business or manufacturing, they sell software for. So and I said before, and hardware also, for testing, components, computers, parts in factories. The newly acquired optimal plus company was specialised in analytics. So they're pretty much combining the two together. So with the NI expertise in manufacturing and testing, they are combining the analytic solution where Oh plus is specialised in that say, optimization, or of eels for digital components. So for example, like, you have a wafer map, testing, this takes a lot of time, in order to optimise that testing and spend less time doing it and maybe also have a better yield of components. You use the opium plus solution to do some estimations, put some EO AI over the statistics, and pretty much say, Okay, we've seen so many wafer maps from this product line, we can estimate this much are good and these dyes are bad. So we don't have to test all of them and waste time. So pretty much just starting to combine the two together to offer a better product.

Researcher 12:40
Okay, great, thank you, we will move to the to the to the core aspects of the core of the interview, but the most important one, we need to agree on the definition of software quality, because most Yeah, most of the discussion and the example you provided is about software quality. So you know, it can varies from one team to another, but we use one definition, which is the ISO standards definition. I'll read it to you and we can discuss it afterwards. Okay. Okay. So it says software quality is the degree to which the system satisfied the state and implied needs of its various stakeholder and does provide value. So this is the ISO definition. And also the ISO model proposes some non functional characteristic which the software products should adhere to. The main ones are performance, compatibility, usability, reliability, security, maintainability, and portability. So first, do you agree with this definition? Or you don't agree, or would you like to comment on it?

Participant 11 13:51
So first, I like to have it in front of me. Because I think I deviate so you could paste it somewhere. And

Researcher 13:57
Yeah, I pasted on the chat. Is that okay? Yeah.

Participant 11 14:01
Perfect. So I can come any comment on it line by line?

Researcher 14:05
These? Yeah, here it is.

Participant 11 14:09

Okay, so I took it with performance also here is really cool. Yeah. Usability, portability. Yeah, but I also add, let's say that also development side of things. So these are more like, I would say client related.

Researcher 14:31
Yes. It's very focused on the end user, isn't it? It doesn't. Yes, yeah. It doesn't focus on the internal aspect of the of the product. Yeah.

Participant 11 14:40
No, and that's, that's a big problem in my eyes, because let's say if we, if the development team doesn't have tests, if they do not have comments if they don't have documentation, the software may have a short life spam. I consider this also quality.

Researcher 15:24
Yeah. Yeah, I agree it is too much. It has very focused on the end user. Yeah. But let's keep in mind both, because I agree with your quality is not only the end user perspective, but it has to do with the scalability of the product, the quality of the code. So we and I can see that in your example. In your example, they mostly focus on the internal quality of the product. So there is no major disagreement here. Yeah.

Participant 11 16:03
I do have some examples of external quality if you want?

Researcher 16:08
Yeah, that's fine. You can add them later on. Okay. So I can see there is a lot of focus on quality, just judging from the example you provided. So how do you assure this quality?

Participant 11 16:29
So in an AI, we have what's called an LTI, T, low long term, I always forget initiative investment, or one of the two, or some other word, it's a long term thing, which they mostly focused on testing technologies, coming up with standards, choosing testing frameworks. And so on some of the old day, what they do is just experiment, and see if those experiments beat with what we work. After that they create testing guidelines, pipeline guidelines, release guidelines, so forth, they're not mandatory, that's an important point to make, you can alter them for yourself for your team. But they're recommended, as at least to see if they work on a large scale. So as many teams should use them, as long as it now it is not a burden. So it should not affect the development time or make, make it a hassle. After that, it is pretty much to the team itself to handle the quality of the software. So after you have those guidelines in place, you should apply them. PR to pretty personal requests. So as long as we're in the iteration, so once the your software has an owner is hurting you you're pretty much responsible for it. So one teammate will create a pull request. So it push the code online up to the to the repository, and another colleague will have to review it. Usually, the more senior colleagues give you the code. And we try to be as strict as possible about it. We also have like linting rules. So we do have some automatic checks. If we can we have we do it try to enforce a policy for unit tests for each component that we make. Even if it's a hassle sometime also like UI integration and end to end tests. We also have tried to deploy it internally like we try to do as much as possible to find issues as early as we can. So we try also to

make the deployment time to the internal service as short as possible, for example, so we can see like in an hour cost if there's an issue that could affect us. Now, we do have let's say there are more sensitive parts of the code in those parts. We do have owners that are from multiple teams. So if there's like a library or something that multiple people are using, they will have each representative in Informatica review. So this is automatic we don't have to add ourselves we have file which we write who's gonna be added automatically and will be added and we do hold meetings each week for multiple practice. So we have like a you know how in the old times they were like the school of for either no work, the School of farming the school, whatever not in school, like learning by the school, like a few scholars gather together talk about things. So we do have those kinds of things of testing, where development, and so on so forth. They have meetings each week for an hour, talk about what's happening in our company, what can be improved, what issues we had. And usually there are like six, seven of us pending on what we work there, like high level people, lower level people, everybody is open to work, whoever is interested to, to take part in the talks, where we gather everything we can and try to find ways to go around it. And either this will be the sole besides those meetings, or one of us will take responsibility for it, or the past two days, the I. That's pretty much what I know about handling quality in the company. If there's some other processes, I did not reach that yet.

Researcher 21:03
Okay, that's already very thorough and focused on quality. That's good to hear. So we will move to the core aspect of the interview. And from the answers you provided my assessment is that the work environment is relatively or even highly safe. What do I mean by safe? It's not physical safety, what I'll explain to you what do we mean by safe and we can discuss my assessment. So what we mean by safe is we mean that the work environments provide the work environment provide a sense of security from repercussions and so you feel that it is okay to admit mistakes, team member feel that it's okay to propose initiatives and discuss problems and there is a sense of confidence amongst the team that will not be embarrassed or reject or be punished for speaking up. So this confidence comes from the material respects and trust amongst the team member. So when I looked at your answer, I think you work in a team which is highly safe. Would you agree with me or disagree?

Participant 11 22:28
Yes, I agree.

Researcher 22:36
So to what extent do you agree Do you think you strongly agree, you agree, you disagree?

Participant 11 22:38
Strongly Agree.

Researcher 22:46
Okay, fantastic. Let's start with the examples. Sorry. Let's start with some of the examples. So the first one I ask if you make mistakes on your team, it is often held against you. You said no, and the answer practically it's a little bit long so I won't read it. But practically it says people accept mistakes and learn from them. This is what I understood. Before we discuss the good example you provided here can you elaborate a little bit how people come about admitting mistakes?

Participant 11 23:25
Um, well, first they have to get used to it some so some people which are from outside the company I'll give optimal plus for example now, so the company was bought so our new policies are a bit more efficient way to kind of keep it for so themselves did not predict help put that bother trying that first strength puller to let it out right. But in in So, what I saw from the beginning was even the senior members, the junior ones were telling every issue where me at the beginning was kind of keeping to myself or at least saying it only my mentor, not and I was not coming today, but slowly as I saw them doing it and they were being open and there was no problem and also like they're supporting the material that the company presents itself with. So we have internal see trainings on the company values and they do have like a little how they call it values booklet that we try to the company tries to give each employee as an example. This is what we try to be as company these are the values that we like to have and they're trying to do all kinds of exercises to socialize these values. It's a bit cringy for me sometimes it's a bit weird to be the saturate but they I guess that's the way that we corporate they do try to be open and let's say innovate and be fast and whatever. And as, as the more you stay in such in the easier to be for yourself to open, speak more freely. So after a while, I just started to edit a bit. I said, Okay, I have the issue issues I saw that you should issue over there. They're like, okay, what can we do to fix that I'm just based on for 10 minutes. And that will be it for like, what they found, or if I give a mistake, if I have a mistake, or just like, I would think, would be the most less, less, let's say, knowledgeable person in that field, I'll just put him in a meeting and explain to him like I this explored these fixes. And there'll be okay with it. Like, tell me though that like, Okay, you have to fix this, this and that and be done. You know, mostly we do not try to this is important, like, we do not try to fix issues for other people. That usually tends towards frustration, and the person who did it will not learn. So we try hand holding hands. So if somebody does something bad, we'll do pair programming will not fix his mistake for him, but will guide them towards it. We do not encourage giving straight answers, it is not needed. So we usually try to guide with questions. So did you try to do this? Or did you try to read that Windows dictation? Did you find something about this term? So we do give them like a few hours to fix it? If they can't, we will try to maybe do it more, let's say engaged. So we'll we will give them some answers. But it's it is as hands off as possible. So they learn to fix it by themselves next.

Researcher 26:56
That's a very healthy and intelligent way of doing it. I'd like to follow up before we go to the examples. I like to follow up on the values promoted by the organisation. Do you think those values they promote are the source of this safety that you experience in the workplace?

Participant 11 27:22
Well, it's nice to like to see that corporate, let's say management tries to embody those values but they should try to, to give an example of what they would like to see happening. But it's more important that the managers practice it. Because if they could say like, here's the perfect company, look how cool we are. So it highly matters that managers and project owners also embody this kind of behaviour. So it's not only about the company values, it's about the employees actively doing it.

Researcher 28:06

Yeah, I agree. And if the middle management does not embrace the same value, they become useless. But yes. Okay. I like the first example wishes about the AWS. The load balances, I think you did something to the configuration. And yeah, can you share with us that example? What happened? And can you can you tell us what happened first, before we discuss it.

Participant 11 28:35
So pretty much I had to deploy a new service. It was a front end application in Angular that I had to deploy, I have to write the load balancer for the Veritas cluster in AWS. And I put the wrong indentation and subito context somewhere else. So it was the wrong name in a place and the wrong station in other place. Um, I know realise it. My principal, like I have a colleague, which is a principal software engineer, he didn't realise it either. Either see it? It got passed through the PR into the pipeline. And there's a respect issue there that we cannot test those kinds of changes, you cannot build it you cannot simulate running it you have to run it. So then, we just, it just got through the pipeline to the development environments. So for CTSP Pro to the environment and the production, but this I think they're both like kind of people on their lap but I think the different stages, but I'm not sure. So you got into the environment. The environment is broke everything. Like nothing works anymore. I got the message from a calling like, this is failing. Okay, what can I do to fix it, I started another PR and made it immediately. Meanwhile, the other colleague, which was more senior, he rolled it back the change. We looked over my poor request to fix it, he approved it. And that was the story. So in less than an hour, we fix it.

Researcher 30:29
So how did you how did the team reacted to this to this mistake, which is a serious production mistake? How did they react to it? What's their approach to it? And

Participant 11 30:45
Nothing? So the reaction was okay, I didn't know but they knew. So they did react, said, okay, it happens. And let me be. And then as I started asking around, like, isn't the way we can test these things? Like, isn't the way to do this, this answer like, we thought about it, we tried. There's no way to do it. So that's why we have that development environment. So if this happens, it would crash there. So it was expected to crash.

Researcher 31:26
Okay, so you learn from this mistake?

Participant 11 31:32
Yes, No, the thing is, like, I mostly knew what I had right there. It was a mistake of not knowing some specifics. So I don't know, maybe I could have avoided if I had not experienced by the thing is like no one else could have done it. So that couldn't be avoided that way. And I did not have more experience this problem. And going around with, it wouldn't have worked. So I don't think there was much of a lesson to be learned. Sadly. Yes.

Researcher 32:11

Yes. That's, that's a knowledge you had to catch up with. And yeah, the mistake helps you to catch up with the with the with the knowledge, right.

Participant 11 32:20
Oh, yeah, I read more about a little bit on what they should have dropped there. But yeah, let's say now I think more in depth about those things.

Researcher 32:33
Okay, you have another example, which is even good for this, which is a colleague who came to the team and his standards of coding where we're not up to your team, can you share with us this example?

Participant 11 32:48
Okay, how specific you want me to be

Researcher 32:52
As specific as you like, as long as we understand what happened?

Participant 11 32:57
Okay, so only she came from optimal plus. So they were stuck for a very long time. And they never really caught up to enterprise level software quality practices in any way, shape, or form. We still have a pretty big issue with their code, mainly. So there, we're starting to sort. So they wrote code, when they had issues, they were disabling Linter rules. So like they're disabling automatic checks, so it doesn't fail anymore.

Researcher 33:50
So they can get away with it.

Participant 11 33:53
They didn't know how to fix it. That was the problem.

Researcher 33:58
Ah, wow. So they were disabling it? That's quite serious.

Participant 11 34:01
Well, yeah. But good luck to you if you put pull requests. So they will say no, we don't do that here. I have the they didn't know how to set up a project. So I have to teach them. I had to pretty much give them links or tissue documentation, teach them how to search for, let's say possible fixes everywhere. Teach them how to properly because things sometimes. It was a freak about them starting small to catch up. But I still get some time. It's very easy questions that could be read like in the first 10 rows of the documentation. So let's wait even I had a question about how do I make a script run automatically after The Angular application installs the packages it needs around. And if you like, if you Google that, to just that question, it will be like, you'll find the first thing is documentation. And in that name, the first five rows explained. And that was kind of disappointing to me, because I've been trying to help them for

a good while now, you know. So there's, there's also big culture of software quality in our team. So if you're not used to it, and if you did not experience it, and did not practice it for a while, you will have a hard time and long time trying to catch up.

Researcher 35:54
That's a good experience. So do you think that this colleague of yours, she felt safe to come about being less experience with quality and meeting your expectation quality, that's why she was coming to you to learn?

Participant 11 36:15
Well, suffice to say, because thing is, I thought that the school before, so I had experience with children. So I'm pretty much dealing with people as children, I have to assume they know nothing until they show me they do. So I've been trying to be as helpful as I can and as nice as possible as I can. So the problem is, right now is I've been too nice to her. And she just comes to me with every issue she has is trying instead of trying to struggle a little. So that's the ugly side of safety, is not is not enough, stress is too much trust. So I'm the problem solver, not the let's say the more experienced guy, which I go sometimes for help. And I'm trying to go back home. So I gave her everything she needed was still it's not your, it's not your fault, we're gonna do it together good night, are gonna make sure it's not going to happen. Again, this is how you do it. This is how we find what you need to do. That's happened for these issues. But now I'm trying to go back quickly to because I'm starting to have more people like this coming to me, and I have also long portfolio. So I'm trying to be less helpful on purpose right now.

Researcher 37:35
So how do you go about it? Because this is a helpful, healthy, constructive work environment? How do you go about it to reduce that frequency of coming and asking for help from you?

Participant 11 37:52
So to build that relationship, first, I answered every question and like, took time from my own work to do it. But to pull back from it and try to like use it for a more healthy level for me this time. I'm either like focusing on my work and like answering each hour or two hours. So I just ignore messages for a while. And just take them in a huge bulk, when for all of them at the time at that same time. So I do not stop each time I get the message because that's pretty much destroying my productivity. I'm also trying to, like I said, the beginning to what the other people are doing just answer with more questions. But I know the answer, I'm not going to give them direct answer. I'm going to give them more questions related to the issue, instead I give them hints. Like maybe it's a problem with the package manager, like forgetting the packages, maybe it's maybe you find a way to maybe the package widget has a way to transcripts, like pre and post the step, which pm posts are exactly the keywords you need to search on Google. Right. So I'm trying to make my answers in the form of questions and have them searched for it. So slowly in time, they will not bother me unless it's something like extremely niche or hard to find. Or that requires a lot of experience. But I'm still learning myself up to that. So I'm in process of learning how to do that.

Researcher 39:41

Okay, great. You have a third example, which is even better, I think one of your colleague introduce a bug to two production.

Participant 11 39:50
This was even before the two teams that I started. It was more than one and a half years ago. So I had no responsibility in that at that time. I also had the party but later so what happened was let me see if I remember properly so Aaron brings up what was the bug maybe I think some files were a misplaced some configuration files, I don't remember. So our software does expect some files to be in certain places. If they're not it will crash, it will not start up. So we have a lot of moving pieces, and one of those pieces was misplaced. We usually run these by hand so once we package them we run them locally on our computers and try to see if they work and install and or not but I guess maybe he found the issue fixed in any fault okay, and then just again because already fixed it. I don't know what the situation was. Exactly. But sometime a bug got into production so it was released. I think after one day or two I don't remember so little like a little while we got a ticket from a client that said okay, we did we installed this update on our like, they do some provisioning, so they have some like servers which first they set it up and then transferred like the databases so they do not like this their own factory for this. We start we started with has an issue does brand Braun This is the arrows they have, whatever. So it was in the area of our software that we made like the piece of software that we wrote. So pretty much the project manager came pull us all out all of us in the meeting show us a bug told us now figure it out. There's a customer having issues with the installer. The team also send us a ticket that there's an issue with the installer itself. And we look for the code the guide before we found was at fault the guy said I think I'm at fault. I'm gonna try to find a solution for it.

Researcher 42:35
So do you think he came forward and because of this safe work environment? He didn't feel like he would be blamed Do you think that's why he came forward?

Participant 11 42:50
Well, it's not like you wouldn't be playing which we would have found anyway who did it right. So there's a No It makes no sense to hide it's more of a question of let's say self worth or nor and I don't know how I should call it it's that you either have respect for your work mostly. And yeah, he also felt safe into saying it.

Researcher 43:27
Right? Yeah, so do you think as a team you learned from this experience and from this era and how

Participant 11 43:36
Well when you don't have a way of making it better sadly that like there's no way around like manual testing but testing it but we did try to promise each other that we would be more rigorous. We made an example like in an Excel which where we have steps and we have to sign at each step with our names so yes we learned and implemented a process gate to vet and avoid similar problems in the future.

Researcher 44:20

Okay, great. Let's move to the next example to the next item which is member of your team can bring problems and tough issues. And you said yes, we do that every time we think of something that might be a problem for quality. Anyone can be reached at any time to discuss problem and we frequently voice chat outside scheduled meeting. If you ever identify something widespread, we schedule a meeting and with all affected party and let the person we found the problem and present it. We value a keen and observant eye for quality. That's really An interesting way to say it. You do have an example here, which related to software quality. Can you share with us this example? The first one?

Participant 11 45:13
Okay, let me just read it this way, remember? Okay, yeah, that's a way from. So away from my PC, mostly, you know what, by the way, please?

Researcher 45:27
I know, it's our way. But sorry.

Participant 11 45:29
So we had the wait for map which had it was colour coded for each dice, die, to see if it's broken or not. And found that that's the colouring, the business logic for making that colouring and the throwing logic and everything was pretty much punched together. So if you wanted, for example, to display so so the way from my was capable of writing percentages, or numbers thing in there wasn't much. So if you wanted to do anything else, or anything else, maybe more custom colours, or to do column demo, custom logic, not all the logic was written in it, you couldn't. And this was working for one of the use cases we're trying to do right now. So I raised this issue and we talked about the splitting it up once to get approval, because right now we're in the process of let's say be working some of the software is part of their MVP. So once we have the MVP done, the all of the next part is rewriting that component to split it apart. So it has some business logic for colouring and yeah, thinking about what to draw and drawing are totally said that, so we can use it for anything else. And yeah, it was so well, well received, of course.

Researcher 47:18
So, you, you, you took this initiative to improve the reusability of this component, because you felt it's okay to do so in this thing.

Participant 11 47:30
Yeah. So we usually try to propose initiatives, we proposed a backlog for technical debt. So any kind of issues, we find we raise it to each other, so at least we all know about it, and then put it in there. We talk to management and discuss the items in the backlog and we prioritize them. But we do not, like assume they will. So we're hopeful, we just get rid of those technical debts. Sometimes the backlog gets huge, big enough. So they say like, okay, we can go on and start to fix. So we'll fix it. But sometimes they might get like overlooked for a little while, because they might be more priorities from the clients. So, for example, this would not have been a problem for the client, but the problem for the development teams. Because, as we discussed, we feel safe, we always negotiate dedicated Sprints for technical debts. We keep track and the quality improves.

Researcher 48:29
So that's bring us back to the definition of quality. This does not have necessarily back on the user because the feature will be working. But it has an impact on the reusability of the component. Right? And the code,

Participant 11 48:45
It will be maintainability. And portability.

Researcher 48:49
Yeah. So when you take initiatives like these ones, do you think as a team you learn from them?

Participant 11 48:57
Um, it highly depends from where that code came from. So if it's from outside, so for example, for example, this code was from outside of it. So it did not use the practices that we use. So we fought over the writing to use the practice that we saw nothing to learn from the request, just let's bring it up to practice, if it's from inside, so it's something that we wrote, for example, we had a service, the query service that I told you earlier about that I had to write the first part of the team. It was a bit rushed at the end. We had some the configuration part of it was It wasn't really meant maintainable. It was pretty coupled. It had no tests. That was a problem. because it was very hard to test we didn't really know how to. So after everything calmed down a little, I raise the issue that I also worked on it. So I know what I did around that. I said, like, this isn't really working for long term, we should do something about it. So we are looking at something like either it doesn't matter, I think doesn't have to say like one or two story points. So I say let's say one or two days for me to go over a terabyte it. So we can extend it further as the easier not to have to struggle with. So and then yeah, let's say we learn that it will be better to wait the first time. Good. But again, that's highly dependent on how much time you have. Uh, sometimes it's easier to write things specifically for the issue at hand. Right, and then refactor it to be more modular. But if you can write the module from the first time, if you have to sign in, you'll know that it will change. It's good to be like that from the first time. But if you don't know, just, it's easier to read it like an audit and see later what you can do.

Researcher 51:24
I agree, it is more efficient to have the practice upfront than refactor later on, because it's it is more efficient, of course, because refactoring is time consuming.

Participant 11 51:39
Yeah, but also, like, let's say, adding modularity very quickly, that way isn't needed, adds complexity and its scope where it might not be to exist. So it's a very fine line between the two sometimes.

Researcher 51:55
Yeah, I think the best way is to be pragmatic, right? Yeah. Thanks for that good example. We may go a little bit over time. Is that okay? If we take 15 minutes?

Participant 11 51:59
Don't worry about it. No problem. Okay.

Researcher 52:05
So the next one is being rejected, because of your ideas, right? The statement says people in your team sometimes reject other for being different. You said, I don't know any example. At least not when I'm working. So and you provided an example, which is you said we all are beat by the same quality practices. HBr needs, at least what we mean by rejection here is you having a different way of working or an innovative way of working you do things in a different way. When you do things in a different way. Do you get Do you feel like people get rejected in in the team?

Participant 11 53:12
This is a much different point going around, we'll have the standards pretty much set. So if we expect people to do things a certain way, all you know, take what the innovating, either goes in those innovation days mentioned later, or they're handed off to their AI teams. Or if you have a proposal, you're we're happy to present it. But we have to evaluate against our current standards and see if it's better or worse. So we expect you to do certain things a certain way.

Researcher 53:52
Yeah, so because it's highly standardised. It's yeah, the best practices are adhere to so you don't see people deviating from them?

Participant 11 54:04
No, not much. And if they deviate, they should be corrected. No, it's not the problem that we have an issue with working differently is a thing that all our practices came from mistakes that are made. So it's not like, don't do this, because I don't like it's like, don't do this because you end up doing that.

Researcher 54:25
So that's very interesting statement. That's all your practices come from mistakes. So you learn as you go, and you adjust your practices. Is this what I understand?

Participant 11 54:35
Yeah, that's pretty much what that we are doing. So they're looking at the, like, schools each week, when they're making they look like, okay. We, for example, the UI one was like, we're having issues with our styles because they're global, and they're overriding some of the styles and what not. And it was not the widespread over it was like one or two people that had it and they were more senior. So they pretty much started to figure out if they can make a standard are typically avoided. So that's how they get, they're getting bored. Slowly so they see an issue that they'll have something that might become issue they think about it, see if they can be can do it a certain way to avoid it. And if they can, do they make a standard. It might not get published officially, but it can go by the word of mouth sometimes depends on how important or big scale it is.

Researcher 55:33
Fantastic. I'll move to the next question. Or the item absent in the email wishes. It is safe to take risk and initiative on your team. And you said yes, and you explain about something very interesting, which you have innovation days, which you organised. Can you talk to me about this innovation days? Just explain to me what's the scope of an Innovation Day, and how you go about it, because this is very interesting.

Participant 11 56:05
Well, it's pretty much let everybody to whatever. The thing is, it's an opportunity for developers to try out new things, new technologies they're passionate about, maybe play with something new in another field. Try I don't know, maybe talk like AI some processes, or provide some service in another language or for another architecture, or they pretty much everybody's free to show off. Whatever they want to do. It's mainly we try to say, Okay, do it, but do it focused on our product. So look out at our product, CST, which might be of interest to you, and try to do whatever you want with it. So after innovation, they will have a presentation of a few hours, which everybody presents their experiments and our data and see if we can choose something that we want to keep long term, or we see a technology that might be of use somewhere else. And it's also like a time for the purpose of relaxing. So it's about fun. And yeah, innovation, everybody's going the crazy science.

Researcher 57:21
Do you think this all before we move to the example you provided, do you think this innovation days helps improving your quality and best practices?

Participant 11 57:34
I'm not sure about working best practices, because they do not apply to these at all. Most of the code is a mess when they exit innovation based or it's pretty much whatever a person could do in a day or two, as an experiment. So no tests, no nothing. It is just the pure of a bunch of lines of code put together. Yeah,

Researcher 57:55
I didn't mean the experimentation of programming. I agree. When you are in Innovation Day and trying something experimenting. You don't care about quality. What I meant is do people bring ideas in these innovation days to improve the quality or the best practices of quality?

Participant 11 58:17
Oh, yeah. Oh, for example. Let's mistake a bit. So other ones from what I wrote there?

Researcher 58:29
Yeah, we can work with the example you have here. The one about the UI test? Yeah. Can you talk to me about that? That's a very good example.

Participant 11 58:41

Yeah. So, we had an issue again, optimize plus, I hope this the company needs things doesn't get out. So we got the software, it wasn't tested, either. It was tested manually, and like end to end but helped me to learn when tests but didn't have like properly to I like UI similarly to unit test port for the UI will have like interaction tests for it like without them when we want to just purely the UI to see who works to run faster, and they don't want to like have to wait an hour or two to see if it works or not. So we had nothing, no framework for that was nothing else. So I have a colleague, which looked into a few technologies over those innovation days, he was curious, like, could we do something to lighten the workload, he was kind of getting bored and sick of trying things manually all the time to see if they work. So he spent time researching whatever he could he came up with three options. I don't remember there were three separate I think one was It was a deprecated, one from Microsoft, I don't remember the name. And then another one from related to Microsoft XR one employee neutrons. So what we ended up was choosing, I don't know, chose one, which was one of the terminologies that he experimented because he, he wrote tests in each one of them, right. So he made the like little report like, this is the issues we will have with each one of them. This is the benefits we can have from each one of them. This is easier to work with this is hard, but it's also a made like a little cream. He presented the test at the innovation day and was like, Hey, this is cool. This can help us. This will be let's say the responsibility of us for doing that manually was it will improve like the time spent on it. We'll have we don't have to spend like an hour to click around like, yes. So we pretty much make it made the feature after it afterwards. And we implemented it. Well, he led it. Some of us helped here. And there. He did.

Researcher 1:01:15
That's great. Do you think that your colleague in this example came forward? Because he felt it's okay to propose something new not experienced before?

Participant 11 1:01:26
Yeah. That's the role of the innovations. Yeah. So free to do whatever.

Researcher 1:01:30
Yeah. Fantastic. So do you think that his initiative has improved the UI tests standards in your team?

Participant 11 1:01:41
Yes, of course. We have UI tests now.

Researcher 1:01:45
Okay, fantastic. You have another example here? Which your work together or to reduce this to improve the speed of your queries and to reduce the complexity? That's another very good example. Yeah.

Participant 11 1:02:01
So the first team, when we first bought the three teams, or for the backend service for the querying. So it was already about improving the performance, which we managed. But I wasn't, I didn't like it, or something. Like what we managed to do, wasn't enough for me, I didn't like the solution. We didn't have time to find it. But we had the new innovation day. So I was like, I want to try to do something more radical. So I looked on the internet for the technology that helped me I found a graph database. Which

outflow had a many problem. Yeah, so we had many filters, or many values. And they were structured. So querying them was extremely slow. Anyway, we tried to do some caching and some merging of queries that improve our speed. But we didn't have time for anything more. So I took that time to, pretty much I took a week, that time for me. To rewrite the small query engine that would take I have an example database, epic, much past all of it. And put it through my custom query engine. That I pretty much wrote every query as a graph. And then mapped all the values of the data, we saw pretty much a pre ran on the base of the moment, and new data came in and ready for the from the query and pretty much placed it on the tree, that I placed the result of the tree. So if f to f anytime is looked for a way or for a query term, so for example, a product and an off factory, it will look on the tree to see if there is a way that contains all of those terms, and all their children. Right. And that improved query times for pretty much any query for elders. But he didn't get in the prototype because it was a bit crazy. It was nobody ever tried it. And it was totally out of scope. But it works for at least for the example that I had with marbles. It was I was proud of it.

Researcher 1:04:48
Yeah, that's, that's a great initiative. I mean, so did you share it with your example or with your colleague?

Participant 11 1:04:56
Of course yeah.

Researcher 1:05:01
So do you think by sharing this example, taking initiative share in this example? Did it influence your peers?

Participant 11 1:05:10
Um, I don't think so they already used to these kind of things. So maybe if there was a new year, I think yeah. But they're also being caught suffering with like, I don't have like an actual example. But they can, like, feels affordable returns. And so if you have a new colleague, who will also include them in this, so we'll just tell them, Okay, what you bought it, maybe we'll hold hands? And what are you passionate about? No, in fact, I don't have an example. So in the very beginning, when I was a new team member in the first team that they were working on in this company, I wasn't seeing a new one. And they, I worked alone by experimenting as I like workflow, but one of the new guys was paired with a senior. So if someone doesn't know what to do, or doesn't have something to do, they will be paired together. So it isn't about inspiring anything, because there any way they're going to do it. So if they're going to be part of it anyway, there's no way to not do it. You will, you'll do it. Nobody's working product those days.

Researcher 1:06:20
Okay, fantastic. So you have another initiative in the organisation, we should call long term initiative as a team dedicated for this long term initiative. And in your email, you said it's usually quality related. Can you explain to me the role of this LTI? And how do they, you say here, they research best practices, and guidelines guideline, and they show it to you or they presented to you to follow?

Participant 11 1:06:56

Well, all it really is pretty much each day. They take parts of our software, so we have the work on they work on everything. So from when a client complaints that, for example, the software stack is too big, they want a smaller deployment. What they do is look at our deployment, see the technology that we use, see how many servers will be and try to see if they can't keep cut it apart? And if they can, they'll make like features, discuss it with upper management, and see if it's a priority or not. So that's one thing they do they see if they can minimise software or change our software or whatnot. Like as a whole. And other time they look into what's the current market doing so they're looking, what's the latest technology? What are the, you know, those charts with? Where do you see we need to stop this, we need to watch this, we need to avoid this technology is also a codename of the charts. Are you familiar with them? Or should they lead first?

Researcher 1:08:13
And yes, a little bit not too much.

Participant 11 1:08:17
Think they think that Nemo talks itself and not sure.

Researcher 1:08:21
Sorry, what they could

Participant 11 1:08:23
know that not just I also think at all but yeah, I'm not sure this there's it's one of those charts with like little circles. Each circle is a company and it's like to the charts. And it's four quadrants. And it's like, we need to look at this new provider, market leaders market whatever, then multiple charts they use. So they don't ever teach one they see if we use a technology which will be avoided. If they find one, they look for replacements. If not, they look to other new tech, if it's worth it. If not, again, they start looking at what we have an Ico so they're pretty much doing anything they can bring our software without, let's say, and give us give us standards so we don't have to think about so if we also had to think about that, it will take even more time or more of our time to develop the software by having them do this work for us. Like and they do this eight hours a day. Right? You cannot do it better than two hours.

Researcher 1:09:42
Correct. It's they are dedicated not like you exactly yeah, yeah,

Participant 11 1:09:45
It helps a lot because I don't have to see okay, I have to Kubernetes stalker or other technologies to do this. No, I know they have we have to burn it. Does anybody know what they said? We don't want to direct one to use help and we pretty much changed all the software to hell in a few weeks. So it's much easier. So we don't have to think about it. And we did see improvements with a no why? Because we didn't ask, we know, we know, they said, Okay, this is the best practice to do. And we did it in words, we're happy.

Researcher 1:10:20

So these best practices they bring is based on examining your code and proposing something. Do Yeah, yeah. So do you have an example? They propose a best practices that influence the team quality?

Participant 11 1:10:42
Thing is, I'm not very close to that thing. No, nobody I know, is pretty much independent. Um,

Researcher 1:10:54
know, what I wanted to know is, did they put forward any initiative or that has influenced a best practice for quality?

Participant 11 1:11:06
The latest one I know is the Help box. So initially, we had Docker images for each surface that we want to deploy. Afterwards, we want to Kubernetes. But the promise, I don't know, what was the issue with this, or they didn't like something, I guess, for some, something great to deployment. But I don't remember, right. But what I know is that in the end, they went again, to Helm charts. And because in the Helm charts, they also have included some other configuration for load balancers and storage is also port. So there's even less configuration data. So what they're trying to do is pretty much adding service, one command away from being installed. So what does that target? So with Docker, there are multiple points they had to take care of, to provision then with Kubernetes. They were less but the defect still had to write multiple scripts. And with Helm, after the writer scripts is just one command to deploy the service. So yeah, the trying to reduce the amount of work, the person is mostly that that was the target. I guess.

Researcher 1:12:35
That's a good example. Thank you. Let's move to the next item, which is, it is difficult to ask other member of your team to help you said now, you can reach call any colleague at any work hour, I at least help around 510 times a day. Yeah, yeah. So that's excellent. Do you think you have you have an example here? Let's start with this example. Can you share this example with us?

Participant 11 1:13:10
Okay, let me do this again. Okay, so this was regarding to the time we started the, when the nine colleagues came. So the web development side. So from all the people there, I was asked is experienced one with Angular material development. And we have two interns, I think Sam's seniors, but they weren't up front. Then we had the DevOps person. So he knew cloud deployment stuff, I think the front end, we had a lot of people who didn't know. At that time, I need help. So I made the document for them with links. So where you can learn Angular, what you should learn for Angular, what links for our practices for when they're new links towards current documentation that we have for the explanation of the services that we already deployed. Pretty much anything to get them jobs jumpstarted. And after they read all of that, I took them one by one and address like whatever issues they had. So for example, depending on which side of the codes they go to signed on. They might even not have experience with. I don't know, it was not reactive state oriented state management. Yeah. They, you know, reactive state management. So how to go into production with them, someone explained them how it works, how our software uses it. In other parts, I had to explain how the

deployment works and explain the pipelines, how it was written and why it was written either because to write pipelines, neither side they had to explain how to create projects, or someone had to write the to create the project from scratch. And I told you how to which rules or which linting rules to copy, which lay how to configure the Angular project to work like the other source or Yeah, and I spent, like, half of my day, usually that for a little while, helping people around to get settled.

Researcher 1:15:58
So how did this initiative you took to help your colleague has helped the code of the quality of their code?

Participant 11 1:16:10
Oh, well, we had little to no mistakes coming into the pull request.

Participant 11 1:16:20
So we had little code to correct. So the pull request took less to review. This the speed at which you get out was much faster. So greatly, I guess.

Researcher 1:16:35
Okay, fantastic. I'm happy to hear that. Thanks for sharing that good example with us. We will move to the next item which said no one on my team would deliberately act in a way that undermine my effort you said true. We all consider that is one of us is doing bad, all of us will be affected in the end. So we try to act the best weekend. Even if something that we don't like happen, we often address both positive and negative feedback, either at retrospective without naming a person or if it's more personal or sensitive in a in a private with me the mediation. Alright, that's a very healthy work environment.

Researcher 1:17:32
This is a very healthy work environment.

Participant 11 1:17:37
Yeah, this is best thing about working here. So people come to this company because we're not afraid to do things to say things. Pretty much would not fail at trying anything.

Researcher 1:17:55
Okay. Do you have an example where? I see you have here an example? Can you share it with us?

Participant 11 1:18:05
The first one? Yeah, or the second one? Um,

Researcher 1:18:08
Yeah, we can talk about both. Let's start with the first one.

Participant 11 1:18:11
So yeah, you're in number the polygon talked about earlier. So they started disabling, hinting and not adding tests. So after a while, after writing the comments, I just stopped responding. And I just told

them, This is not gonna work like this for a long time, because I don't have time for this. Nobody has, we have find another way to do it. So I told her to implant to disable or deleting rows, to enable back deleting rows to see if she's missing anything I gave the example these are the rules that you have to have defined, so that it checks for them. I gave him another example for my other project you have to have at least tests for each component like two for the edge cases, at least the edge cases and one for the normal he's and I thought that like this is the policies not that I have, I want to get to your thoughts in nobody will approve. So if you like if you continue to do to do it like this, it will take more time you have to wait for us, you will just get more messages to fix it. So no, please read on our guidelines because everybody will enforce it. Right.

Participant 11 1:19:44
And, yeah, she started slowly doing it. But yeah, you pretty much have a choice. Otherwise you're you're conquering.

Researcher 1:19:56
So how did you approach it the tone of your voice? The approach with giving her feedback, how was it?

Participant 11 1:20:07
I'm direct. I'm a direct person I don't like to scuttle around with, if it's not something like I was hurt someone's feelings. It's it's not like, it's a problem with him personally, with or her personally, it's like, it's a problem with the code quality. It's not like I have a personal problem, it's like, it's not the issue I have with you. It's not the issue I have with working or something, it's, it's that it, it just won't get approved. So it's not something that I can do something about, right? It is totally up to you to do it, it's and I don't like it's, I understand that they might be weird or unusual, or it could be it's a shock still, because of the fast change of pace. So I do not expect to be like that, you know, expect to be quick change. But I need to see them being progress on it.

Researcher 1:21:11
So direct can be in a different way. I'm a direct person as well. It could be really sharp and rigid and a little bit cold. It could be constructive, warm and nice. So in this instance, where were you were you cold? Or were you all unconstructive?

Participant 11 1:21:33
I'm sure I guess a bit of both. I mean, I'm called on the side, like, the code is a problem, right? There's no way of going around, it won't pass, it won't get published. That's the cool part. And you cannot go around. Okay, so that's it. Now, the word Park understands it has an issue and standards, it's hard to you understand that takes time. I will follow up and see if I can help and have this needs to to try to fix it. The call side again is that I will not approve it. Until it's done.

Researcher 1:22:12
I think that's a very constructive way. It's a good style. It's a very constructive and healthy style actually, it's well balanced. Thanks for sharing that we will move to the next example. You have here can you share with us the second example?

Participant 11 1:22:32
Okay, so doing retrospectives we usually got the get the have they would try the lot of things around the mainly go around what my good what my bad what we what we'd like to have. So sometimes all the all these boards have to be all these post it notes. We do it on Europe. And we do it with Anonymous mode or so we do not know what prompted who wrote and why he wrote it. And we have the one person read them all right. And after the person read it reads it, he can assign the next person to discuss so we try to avoid a person talking by itself to set let's say not show himself. So try to be as anonymous as possible. So let's say if somebody says sensitive issue, it doesn't get attacked, have something in case it's something that somebody doesn't like, or somebody gets decent defensive about. So everything is anonymous. Nobody knows for what it started and why started and in most cases there are more than one person per period because it also so after we put all those anonymous posted notes, we go on voting and we vote on them anonymously and they get rated and we start from the most important ones police department and some issues have usually get more than five votes and we discuss them to see how they because sometimes a person might feel alone in in seeing that issue but they're not

Participant 11 1:24:24
and about people I did have a little problem with the PR which I both voice only to my manager because I felt like it was getting hard for me to manage.

Participant 11 1:24:46
By they just tell me sometimes happens. I should just continue what I'm doing in time. If it doesn't work at all. We'll see what we can later but we didn't take any action or Good morning. Yes. So on that I cannot wait. You don't have any, like, bad, like the bad feedback on everyone? Not yet. So I cannot have it's just technical issues or maybe bugs that someone started or we do not talk about the person started the bug we started, we talked about the bug.

Researcher 1:25:21
So I'm just curious, because I've never seen this. It's very, it's very progressive, and why the feedback is why the issue when you when you put it forward, is always anonymous. What's the logic behind it?

Participant 11 1:25:44
Well, in this meetings, also the managers are there. Okay. So we have a lot of people that are not small meetings, like we have the old, all the team and all our domains work with us on daily basis. And let's say, maybe it's not an issue that you saw in your coordinate is the issue that you saw in somebody else's code. So you don't want don't even if you do it in, like in good faith, right. So you're saying, I saw that code, that's a bad practice, I don't like something about it. That other person either might think that you don't like them, or might feel something that's I don't know. Some people that take their code, like themselves. So if then somebody talks about their code as bad they will take personally. And being anonymous, we cannot avoid the fact that maybe the auditor, it really is bad. Like, we cannot avoid that. But we will avoid the fact that he will know who maybe he was in somebody's target. Right. So that's the main thing we try to avoid with, we don't want to hold a grudge. So we still want open discussions in the deep between everybody. We still want sharing to share issues. And if you have the name that and you have a person so for example, maybe it's a for them, it's like me, this is a font that I always

comment on fronted Code F embed that I find that uncommon. But if somehow I comment the same person 10 times, you know, they've stopped eating before first for sure.

Researcher 1:27:33
Okay, I have just a quick follow up question before we conclude. Do you think these openness in talking about issues and bring them up in the retrospective or in other way, help a quality? Do you see that people learn and the quality of their code improve as a consequence of this healthy and constructive way of dealing with, with, with problems?

Participant 11 1:28:02
Yep, yep. So we also have action items out of this. So it ends up in the battle of time. So if we see a problem with test, we have an action item for a problem about to systemic interacting, we have action item with the user story that we have to face that we see a performance issue, there is an item for that and so forth. And like all these things have resulted in.

Researcher 1:28:34
So the actionable nature of the measures or the initiative, you put forward, incite, or motivate people to execute them and consequently improve. Right. Okay, Participant 11, that was a really interesting discussion. And thanks for the very good examples. I like them very much. And thanks for sharing those examples. I think it's a Sunday afternoon. It's all at lunchtime. I let you go and we stay in touch. I'll talk to you shortly. All right. Thank you very much. I appreciate it. Thank you. Bye. Have a nice day. Thank you.

Participant 11 1:29:02
Bye bye.