Participant 10

SUMMARY KEYWORDS

quality, team, gs, feature, developers, implementing, case, discuss, java, agree, important, company, customer, technical background, scrum, product, helps, application, generate, approach

SPEAKERS

Researcher
Participant 10

Researcher 00:13
Hello, hi, Participant 10, how are you?

Participant 10 00:16
Thank you. I'm good. How are you?

Researcher 00:18
I'm doing very well. I'd like to thank you for the opportunity to talk to you for this interview. And especially it's a Sunday afternoon. Thank you a lot. Thanks for those answers you've sent in the email. I'd like to go ahead and start because there are a lot of things we need to go through questions and things to discuss. I'd like to start with a brief introduction. Can you introduce yourself briefly, mainly your education and your experience?

Participant 10 00:49
Yeah. Graduates from Computer Engineering in 2005 now, and afterwards, it's been almost 13 years working as a software engineer, or senior software engineer or technical team lead with different positions that I worked. Can you hear me? Correct? Yeah,

Researcher 01:16
I can hear you perfectly. Thank you.

Participant 10 01:18
Sorry. Sorry. And yeah, I worked about 10 years in Turkey. And afterwards, I, it has been 3 years. I'm working in Germany now.

Researcher 01:32
Okay, fantastic. Thanks for that. We will talk briefly about your team, the current team. So the answer you provided are based on your current team. Okay.

Participant 10 01:47

Actually, my former company as well, why we had some, I had some, as far as pays on my former company as well.

Researcher 01:58
Okay. Well, each time we discuss the example, can you make reference to the previous or the current? Team, please? Yeah. So currently, you're working on a Scrum and right. Yeah. Yeah. And the previous team was also Scrum.

Participant 10 02:19
Yeah, we were trying to do with Scrum.

Researcher 02:22
okay. How many member in your team currently?

Participant 10 02:33
Six, six developers, and one designer, one QA on Product Management? Okay, we got about nine.

Researcher 02:45
Okay, how long have you been working together?

Participant 10 02:48
It's been almost two years. Okay. Of course, there are some numbers are new or some numbers left to company.

Researcher 02:57
Okay. And what type of software do you develop?

Participant 10 03:03
We are implementing internal communication application for companies or for big groups, like can be university or can be a company, etc.

Researcher 03:20
Okay. So it's usually a custom software product for Yeah. Okay, great. Most of the discussion will be around software quality. So ideally, we'd like to agree on the depth on the definition and and after that, we will move forward with the questions. Yeah, we use the ISO standard definition for quality. I'll read it to you and we can discuss it afterwards. So software quality is the degree to which the system satisfies the stated and implied needs of its various stakeholders and thus provide value and ISO model or standards also suggests some non functional characteristic that the software should adhere to. So mainly performance compatibility, usability, reliability, security, maintainability and portability. So first, do you agree or disagree with this, this definition? And would you like to comment on it?

Participant 10 04:34

Well, I don't like to use this word, but it depends sometimes be it depends what you're developing or what is your product or for the customer. Actually, sometimes it can be. I mean, for example, you're implementing an internal application with security might not be that crucial for the For the customer side, the features will be important and using that features immediately will be important for example, but in general, I would say it depends on what you are implementing or what are the flopping so sometimes what you say and sometimes you focus on feature and less on internal quality, right, because the most important is the feature. Yeah. So sometimes you also compromise scalability just to deliver more features as well.

Participant 10 05:41
Yeah. It's also about the customer as well. customer pays money to get some service. You know, for customer size, it is important to get the service they sometimes don't care about the security or quality or code quality, etc, etc. It's kind of depends on customers as well.

Researcher 06:12
So, they focus on features right, especially on Scrum, when you have a product owner sometimes participating he or she make push for more feature and less other aspects of the software quality. I agree. Sir, so, what do you do in your team to assure quality I mean, what type of processes or practices you use to assure quality?

Participant 10 06:40
Yeah, there's different type of quality measurements like quality or feature quality or the or, I mean, like, having some box can be quality issue or sometimes the quality means like, customer wants something and you are implementing something a bit different, that's also complex called guide with

Researcher 07:15
Okay. So, what do you use for code quality?

Participant 10 07:19
For code? Yeah, for code quality, we have some quality gates in the pipeline, we have like SonarQube, or CheckStyle or JDK system rules or we have some basically some tools to make sure the code quality is at one level for example, also unit tests or integration tests should be at least higher than 60 70%. Coverage, I mean, the coverage and we have this kind of complicates.

Researcher 07:59
Okay, and for features what do you use you use user testing do you use it you do you have

Participant 10 08:07
For users we have like first of all, we have parents deployments for every new feature and we branch deployments first developer or called your beaver will do actually at the first point that Loper will do a one test either on his or her local and afterwards there will be branch deployments with the Contra view and the also another developer will do a test and afterwards they will be through a testing another process and the creator will do a bit broad test perspective like multiple browsers or multiple platforms like iOS, Android or desktop application, desktop tests test and afterwards they will be added it will

feature will be merged to main branch and main branch will deploy to a UHF environment then we have an external company that makes sure the all overall features are working correctly together. That's and afterwards they will be smoke check internally and after smoke check it will deploy to staging environment. We are using that staging environment as internal communication like we are implementing company internal communication to which we are we are the first user of the system. The staging environment will serve us to our company and afterwards we are deploying to production and customers will use it sometimes we have feature toggles some customers can use or some customers do Try to wait for LTS version.

Researcher 10:04
Okay, that's very thorough and advanced quality processes. Yeah, I'd moved to the core aspect of the interview, which is based on the answer you provided in your email. So looking at those answers, I had mixed feelings regarding what we're looking for. So I'll explain to you what we're looking for. And I'll get your assessment, or so what are we looking for? We're looking to assess the level of safety of a software development team. I'll explain to you what we mean by safety. We mean that the work environment provides a sense of security from repercussions. So what does it mean? It means that you feel it's okay to admit mistakes, you feel that it's okay to propose initiative, it is okay to discuss problem, there is a sense of confidence amongst the team member that they won't be embarrassed, they won't be rejected or punished for speaking up. So there is a confidence amongst the team member because they respect each other, they trust each other. So when I look at your answer, I had a mixed feeling sometimes it feels like it is safe as per this definition, sometimes it doesn't feel safe.

Participant 10 11:24
Yeah, this, I feel same as it always depends on topic or depends on the timing maybe? Or

Researcher 11:36
Can you give me a scenario when you feel it safe. And another scenario when it is not safe?

Participant 10 11:43
For example, if you affect to, like you're implementing a feature, and you've spent maybe two weeks and you have only three weeks to implement that feature, and you are end of second week, and you're thinking or saying that we did everything wrong, or we or maybe you as a person you did a mistake in at the beginning, and which will affect hold the future? In the in that case? It's kind of hard to. So of course, it depends. How did you explain the situation? Or how did you communicate this. But sometimes, the deadlines always kills the quality. To be honest, if you have Doppler, if you have customer pressure, that means it's a you shouldn't do a mistake, or you shouldn't change your idea about design or for anything. That's one case. And another case, if you if you did that, beforehand, it's completely fine. I mean, you can say, or explain your thoughts or issues about anything.

Researcher 13:14
So I've seen this example before, it seems like it is safe as long as you don't compromise the customer satisfaction. Yeah, yeah. Yeah. So as long as the customer is happy, you are safe. But if you compromise that, that satisfaction, you are somehow in trouble. smoothly, okay. So is this feeling in your current team or previous team?

Participant 10 13:49
Actually, for my current team, my former team, we, we had the we were, we had a wrong understanding of HR, for example, and we were thinking to, to, it's like HR was an approach to do implementation or to build the last month, but Scrum is a framework of the better approach. But back then we were thinking like, Scrum is the main goal. We have to do this, and we have to stick to the scrum. It doesn't matter or product or features or we should do the these tasks. And it's it wasn't the also main goal to do to make customers happy. That's that wasn't it? Maybe we were a bit confused. Maybe we're wrong. And in that case, we had some issues, mainly a lot of issues. And now it's better. And some companies things like Scrum is everything or sprint is everything or it's, it doesn't matter, especially for big companies, it doesn't matter. What did you bring to customer? Or what? What did you implement? It's important to stick to the rules of the internal process or internal scrum stuff, etc. That's the the main conflict, I guess.

Researcher 15:33
So, do you think your current team is safer than the previous team? That's what you

Participant 10 15:43
Yeah, yeah. Because the now our main focus is what we are implementing, and immediately we are meeting that feature with the customers. But that's, that should be the main goal. Always.

Researcher 16:00
Okay, fantastic. Let's move on to do you think this improvement of safety in your current team, how it came about, I mean, how it's promoted, what make what make your current team better than the previous team when it comes to safety.

Participant 10 16:22
Actually, we were back down to maybe two years ago, or can help years ago, we were, we completely built this team together, and all the new members were hired, and no internal member, like to enter a member and the rest of the team was hired later. And we built our culture internally, like, we decided how to do things. Or, for example, no one told us to use this process or do like this, or we did whatever suits to our team, like we have different approaches, or etc. Our company was only interested in our output, not our how we work, they gave us total freedom. and that sometimes can be different for different teams. So for example, internally, we also have in our company in our especially in our engineering department that we have different teams, some of them using Scrum, some of them using Kanban, or some kind of Scrum bomb powerful. And that's the culture should be built in the team. Maybe

Researcher 17:47
That's, that's, that's interesting. And from what I heard from you, my understanding is they have given you a significant leeway, a significant self governance opportunity. And you took that self governance opportunity and you created this culture of safety, right? Yep. Exactly. Okay. Let's move to the example you provided in the email. So the first one, if you make mistakes on your team, it is often held against you. You answered yes. If you were in planning phase, then you you can be sick on vacation, etc, or implementation. So this answer is based in your previous team or you.

Participant 10 18:37
The that was based on my previous team. And it's the because the sprint was important, the scrum was important than the people who are not important or that I'm developers or customer were not very important for MD. Maybe on the high level management, it was important to what we bring to customer but mid level on mid level side, it was very important to stick to Scrum or anything. That's that why we had for example, some plantings for the spin. And when one or more one or more developers left for vacation or for sickness or any issues, then the most probably the Sprint will fail because we didn't have like, like buffer for the for this case, and that was like kind of a big issue.

Researcher 19:46
So do you have an example from the previous team where somebody came forward and admitted mistake or he or she didn't come forward to admit mistakes? And what happened? Because it was not from what I hear it was not very safe?

Participant 10 20:06
Yeah. Especially for very big companies. Like, for its food chain, this is the case in my former company and the we were not able to freely speak about, what and how should we do. And this is how we should command this. That was the case. And in that case, it wasn't possible to speak or, especially for juniors or for like beginners. I mean, the poor developers, it was hard, more hard, harder to, to say their feelings or their thoughts or their ideas.

Researcher 21:06
So how did that affect the team?

Participant 10 21:12
Everybody left on the long run.

Researcher 21:18
So do you think it's also impacted the quality of the teamwork?

Participant 10 21:24
Sure, it did. We had, like, everybody was trying to the resolve tickets. As may tickets and stories as possible, it wasn't important to think about the quality or and we had lots of bugs. And after, for example, five, six sprints, we took two three Sprint's to fix just bugs. And we also failed, because that brought more bugs because the quality wasn't very good. And that somehow in the end, was like, trash product.

Researcher 22:07
Yeah, this is a good opportunity for you an I to compare, because now you move to a better team. Have you seen now you are self-govern? You created this culture of better safety? Do you see a difference?

Participant 10 22:23

Sure, of course. For example, in our team, it's, it's been three years and only one member left. And also, for example, for production, we don't see that much bugs, we have some very minor issues, but we don't see bugs at all. And also same for the maintainability of code, and it's very huge, there's a huge difference.

Researcher 23:11
What do you think has is helping in your current team, do you think this level of safety is helping and how it is helping?

Participant 10 23:25
Because it's the we are, we have a good process to test or to make sure the core features or, etc, is very good. And we have that buffer. And also, it is not important to especially we are we don't have any sprint or some kind at moment. And it's very good to take a ticket to I saw this issue on last feature, and you can always bring your ticket and just we need to agree with the product management and we have some kind of weekly meetings or once a week, be weekly. And we are discussing this together. And when we decide this is important to tackle immediately and we're just taking the ticket and just solving the issue that's that brings like, good quality. And then also it's I mean like when you bring some topics that doesn't have any output in perspective of Product Management. It's it shouldn't be implemented in like product will always think that it shouldn't be done until it's not important because the As mobile not seen, but at the moment we have the kind of approach when we bring something the product management also believe or trust developers to that if developer brings this topic, then it should be important. And And also, we have we are a bit lucky about we have product management manager who has technical bank background, that might be the case.

Researcher 25:35
Just taking notes because I'd like to follow up that's a good example. You mentioned something you twice which is the buffer, what do you mean by the buffer? Is it a buffer for quality? Or what do you mean by a buffer?

Participant 10 25:49
Buffer is the workload when we take workload we always plan on five developers but we are six developers and one developer should be always like kind of buffer to tackle immediate production issues or maybe some technical issues or maybe some sick leaves are formed for this cases.

Researcher 26:18
So you make you make you make you make a plan for uncertainty right but yeah, yeah, do you think compared to the other team, it takes pressure from you and it helps people to focus on quality.

Participant 10 26:40
In the end, you will be measured by some kind of management will always look into ticket numbers or story points you handle or the feature is kind of not visible for them to measure the team in that case it's it affects

Researcher 27:08

So I've noticed that the developers and the current teams have a significant level of freedom to make decision how to go about work right? Yeah, yeah. When you have this level of freedom, do you think it helps quality?

Participant 10 27:28
Of course when you have this freedom and safety we created, you actually the most importantly the motivation will be good and if developer motivation is high, then he will implement anything with high quality. If you give money or anything else, it doesn't matter. I would say most important thing is team motivation.

Researcher 28:11
Do you think this culture of safety you created in your team together helps motivation?

Participant 10 28:18
Yeah, sure. Because we decide and we push ourselves forward.

Researcher 28:33
Okay, fantastic. Thank you for sharing those good examples. We will move to the next item which is about talking about problem bringing up issues. So the statement in the email says member of your team can bring up problem and tough issues. You said yes, this is natural for our job. Was it the same thing? I'm assuming you answer this based on your current team isn't it? Was it the same thing in the previous team?

Participant 10 29:06
No. That wasn't the case.

Researcher 29:09
So I'd like you to tell me how it makes a difference for the quality of your work. For example, not talking about problem and now talking about problem How did how does it helps to allow you as a developer to achieve the quality you want.

Participant 10 29:31
The for example in our sector, the everything chains very fast and we need to adopt everything always like the our team type to do I mean our technology stack can change or everything can change in time or all the processes everything but when you have something very specific for you. For example, for my former company, everything was very strict. And we had to stick to the same kind of rules like Kim level rules. I'm, I'm mean, like, what I mean about the rules, it's like, it's also a culture, but we have work and, and in that case, when we think to bring something new, for example, a technologist in context of technology, it is already decided by the preferred companies, most of time, this is very strict, it's already decided that you can change this. In the end, it's hard to and also the learning new things, or using new things, motivates the lower parts as well. And whenever you it's also kind of the good for company, good for product and good for developer. And whenever you bring something new, if it's extracted by your project or your company or your product management, that will always motivate you, it will bring

product to a high level or federal level and also for the team motivation. I mean, in general, it will help a lot for different cases.

Researcher 31:42
Yeah, so what I understood is when you allow people to bring their own initiative, you empower them and they become more motivated, right? That's what you mean. Yeah. So do you have an example where a team member in this current team we will we will you if we can start with the current team, a team member or yourself brought an initiative and the team embrace the initiative and that initiative has helped to some extent to for better quality?

Participant 10 32:19
For example, we like we are team we have the half of the team has good background with the JavaScript and so we decided to use for example Node JS for backend side and in for rest of the project we have only Java for example. But with no JS every party like for who implements something on backend side they will they were excited to use this and pass this or experienced this new technology stack on the stack on production or how for the development side it was kind of good motivation for key then we just implemented and we were we get good output from them be bad actually, it's not only how we decided is a bit complex, it's not only we like to bring new something new that's not the case it's like we had some reasons like it's the Java container is taken too much resource at the beginning and we have a small service and we're excited to use it and it was very good use case for Node JS and we used it and it was very successful. I know that motivated everybody in the in the team. Maybe you don't have technical background.

Researcher 33:57
No I developed software. I know I got the answer because I used to develop software I know how we did improved quality, but I like more detail to tell you in this interview how we did the transition to not

Researcher 34:15
For the purpose of the research I'd like to tell me how node JS has held the code I understood what happens from quality perspective but I like more detail if you don't mind exactly tell me how Node JS has helped?

Participant 10 34:31
Yeah. For example, for our also we did some fine tunings or on modules application and also in the container as well. For Java side we that we were using Spring Boot and we needed to put minimum 300 megabytes of RAM to the container and also minimum half virtual CPU on the server. With this one, and we, we had this new service that new service only gets, for example, on production for, for we have about 2 million user for one environment. And for 2 million users, we have daily maximum 10,000 requests and maybe one request per second, or maybe half requests per second. We had this kind of metrics. And we thought, the maybe 50 megabytes of RAM or 100 megabytes of RAM 3%, or three of one, CPU, will be enough for the service. And we used it. We started with this, and we even reduced to we had a bin, maybe 48 megabytes of RAM for that container, and it was perfectly serving the service requests.

Researcher 36:11
Fantastic. Thanks for those detail I appreciate. Now let's go back to the previous team. Do you have an example where somebody brought a serious problem that's affecting quality? And it was rejected? Or it was not welcome? And what were the consequences?

Participant 10 36:29
Yeah, in there as well. We have also, we had same example with no chairs. And we were some kind of generating codes, we were trying to implement No, no code platform. And we had some we were generating some quotes and we bought it, we had option to use only Java. And with what was the there was an Apache there was a framework for your framework to generate the template engine. Yeah, there was a template engine from Apache or remember the name right. And we, we had to use that because we had Java stack. It was very slow, we when the generating of one application was taken too much, as well as the Java is kind of typesafe. And we had to the generator, all the types for the generation side. That's why it was very hard. It was taking more than normal effort to generate the generated codes or the backend, aka we were just kind of generating backend API s programmatically. And with Node JS, the afterwards we excellent we had discussion and with we couldn't use, and we got that message. Like, we don't have enough experts on that side that we can't use it. That was the only answer. And after afterwards, we just did some checks. And we did some research for over that topic. And they it was very easy to do with Node JS, but it was too late. We also did some small POC and it was very easy because we were kind of sending some Jason's to the backend. And that Jason's for Java, we you need to define them strictly. Otherwise, it was of course you can do it without types, like you can use objects or stuff. But in that case, it's hard to process that object, unknown type. But with no Node JS, it was more easy to process that unknown types like on Jason's you can easily do stuff, like easy to process. And not just was very good idea. But it took us very longer. Not just maybe, but we'll continue with the Java, unfortunately.

Researcher 39:34
So who objected to the proposal to move to Node JS,

Participant 10 39:39
Yeah, we had some kind of architects in another team, like to three architects actually. And they were already inside of the we will use Java. That's it. most momentous decision.

Researcher 40:02
So my understanding you did stayed with Java did slow the efficiency or did also slow the quality.

Participant 10 40:12
Yeah, of course. But no one was aware of it.

Researcher 40:20
Yeah, yeah. It's not visible, maybe the feature was working. Thanks for that good example. Yeah, I think the next one is, I think in the email, you didn't understand what we mean by reject, but I read the statement and we can discuss it and explain to you what we mean by reject here, it could be interpreted in many ways I agree with you. So it says people in your team sometimes reject other for being

different, you said now, not sure what you mean by different. So in your current team, which you have this culture of safety, sometimes we do things differently. So I may have a different approach to to go about some programming decisions, some design in decision design decision? And do you think that in this safe work environment, people would reject each other for having a different approach to how they do things?

Participant 10 41:32
No. You can always bring your ideas or approaches on right time. And also, we all should be agree on that or on something in the team, we have this rule, we shouldn't have different approaches for different things, you either you should agree. Either I should agree for your idea, or you should agree with my idea. We must be consistent. And always discuss in the team.

Participant 10 42:32
So we need to have only single approach for our processes. Like, it can be technical, it can be product, or it can be implementation, or it can be how we work. And we should only have one approach to four for all.

Researcher 42:54
So I agree in software engineering you need to be consistent, and that consistency become a best practice. I agree with you. So how do you come about good? Agree, and for example, Researcher, I could bring an idea. Participant 10, you could bring an idea how do you go about getting a consensus? And in the same time maintaining or sustaining this culture of safety?

Participant 10 43:23
Are we if it's related to on the technical side, and developers will whoever bring that idea? They, he or she need to create a meeting and all the developers need to discuss that internally? And if it's related to product level and also the include them to and then it's we will discuss that what is the benefits? What is the downsides or etc, we discuss if just maybe I should search for correct decision making.

Participant 10 44:03
Sorry, just a quick, convince he, either he or she need to convince me or I should convince him why it is wrong, or why he should convince me why this is right. So, in the end, we always have a rule of meeting the we need to have consensus, or we need to have output of who the how to continue or how to proceed. We have this rule. And if we agree, we can change but it's it's also about the for example feature. This should be discussed at the beginning of feature and you can change in the way on the way that's the That's how we approach to it. But if it's general technical, we can discuss that any time. Or if it's process wise, we can discuss any time, etc, etc. Like, the right time and with after discussion.

Researcher 45:19
Okay, great, thank you. Do you have an example where something was brought up by someone? And you had a meeting you discussed?

Participant 10 45:32
Good example. Okay.

Researcher 45:35
I'm excited to hear.

Participant 10 45:38
We had some kind of, like a topic, how we will separate our monolith application, I mean, how we convert our application to micro services, where we had a kind of discussion, we should, some of the developers wanted to use another approach than me, and we discussed a lot. And I couldn't bring reasons. I mean, reasons they have the reasons were more than my reasons. And they went on the discussion. And we use, I stick to the rules, and we used kind of modularization. Internally, we recently restructured the code, but my only reason was this will take forever, and there is no output on product management side, I had only two reasons. And maybe I couldn't explain, Well, maybe they didn't underestimate this reasons. And in the end, we decided with their approach, and in the end, we failed. And after, but after two and a half years, we decided not to continue with when the other, like, find a part to how the to extract it, like, find the possible business part, like logical, the part that makes specific things. And also it makes hard to maintain multiplication. Like, for example, in a big application a pretty generation, time to time, you will get some image and you will process the previews and generate some image. But this is only some generate some spikes on the production time to time when you upload big file, for example. And we need to find that part and extract it. That was my idea. We are at that point.

Researcher 48:09
So in this example, I wasn't able to follow most of it, but it did help the scalability of the product, right?

Participant 10 48:19
Yeah. But I couldn't convince them to continue with my idea. That's why I stick to their idea. And at the end we fail.

Researcher 48:33
Well, sorry, you failed at the end.

Participant 10 48:36
Yeah, we all fail, because that approach was wrong. And I was right. But unfortunately, I couldn't convince them.

Researcher 48:47
You couldn't that's the ugly side of this style of working, if you could be right, and you might not able to convince them. And so as a team, what did you learn from this experience?

Participant 10 49:07
Actually, we also have some gaming's from them from this approach. But in the end, we lost time. We could be at better point in context of that, that approach or that implementation, and we all met last time and company time as well.

Researcher 49:35
So what's happened at the end? Did you go back to your idea, or you continue?

Participant 10 49:40
Oh, yeah, yeah. Yeah. We went back there.

Researcher 49:43
So there was a learning from the experience, right, even though it was a failure, right? Yeah, yeah. So. So safety. Either way, it creates an environment a healthy environment where you can learn and you still can catch up even though you last sometimes, right?

Participant 10 50:01
Yeah. The only thing was like we learn a bit late or we took a bit longer, like we shouldn't have, or we shouldn't try, we should find a way to try that in a short time. That was the downside. Like, maybe we should try it. We should have tried, like six months on, but it took two years.

Researcher 50:29
Yeah. Well, I see it in a positive way. Because in a different team in your previous team, you wouldn't even have the opportunity to talk about it in the first place.

Participant 10 50:41
Oh, sure. Yeah. And, and we wouldn't be able to see, we were wrong.

Researcher 50:49
Yes. Yes. Yeah, you stay you stay in that mentality, which is there is no growth course. I think in this team, there is growth you grow in? I mean, yeah, sure. Yeah. Thank you for that good example, I move to the next item, which is quite relevant to software development, which is helping each other I mean, it says it is difficult to ask for help. And you said no, for my team, it is always welcome to ask for help. That is normal in this current team? Yeah, can you can you elaborate, how do you help each other?

Participant 10 51:32
If you I mean, if you do that also depends a bit on the your verbal, if you have very strict verbal, you have to solve this numbers of taking the number of tickets, then you won't be able to help others. That's the main part main point for that, I guess. And we are a bit like free to do this together. This will, we kind of agree on that with, with our managers or etc. Because if we are at the same level or something, I mean, technically, at the same level with the older developers, that will help a lot on the long run. That's the mentality behind this. And we are always helping trying to help each other like peering together. Or maybe if sometimes someone stuck with a technical difficulty, and we will jump in and just help each other.

Researcher 52:47
Okay, great. Do you have an example in the current team where you helped each other either you are a team member helps another one. And it did help somehow the code quality or some aspect of quality.

Participant 10 53:06
Even in remote situation, like Corona, and we have kind of this course, application, we are always online. With any small questions, we are just hey, anyone here just I have question or have difficulty here. And they will go to another chatroom or voice chat room. And they will just discuss or they will just share screens and do it together five minutes, maybe 10 minutes, we'll solve that small ticket. But in the long run, you will not have big issues in that case. But when you have don't have that much communication. In that case, you need to spend like one day to solve a big issue or just spend today that's not our case. We are always in we are kind of connected to each other with like communicating with each other all the time. And that's why it's very good to the We Are we don't have the big topics that I can give example to there was some issue. Anytime we are just having like small chats and small questions and one, two minutes, maybe five minutes, maximum half an hour time to do something together. And that helps us to provide the long things that I can say, Yeah, we did this, but I don't have that kind of example.

Researcher 54:40
No, that's okay. I understand. I mean, if you jump in and you help somebody, I mean you help them to solve a problem. And when you help them to solve a problem, you contribute it to a better design. And maybe for example they jump in and they share with you their code that piece of code and you help them to make it better. Right? Yeah. Yeah. So that things happened quite often in the team. I mean,

Participant 10 55:11
Whenever someone asks, looking for this class, I was thinking to do this, like this. And that's why I'm looking for this course. But I couldn't find, for example, we just say, no, you need to do like this. You don't need that class that solves issue that prevents him to spend like one day and afterwards, it will be hard to it will take more time to fix that issue. When someone spend time on it, like when you are not going to run away, and you're not going far from the way. Forks help. Yeah, so that's why it saves time.

Researcher 56:01
Yeah, it doesn't only save time, or what I understand from your explanation, it's always prevent people from deviating from going away from your own standards. Yeah. So this helping each other collectively. I mean, the knowledge is circulating in the team. Do you think this helping people learn from each other's? And do you think this fast learning from each other helps quality?

Participant 10 56:34
Yeah, of course, like, we don't have that much technical debt, for example, because of that. Or wrong designs Implementation. And that, that is the most important thing for code quality. It's not very important that you have some small stuff in your code. It's important to not have like, huge design mistakes or like, when you try to fix you need to spend two weeks or maybe months that's have serious consequences. We reduced technical debt significantly but helping and making decisions together.

Researcher 57:29
Because technical debt, for example, and design issue, have more economical costs like yours, you have more economical cost, and the bugs they produce are more severe. And if they produce severe bugs in production, again, the economical cost is higher. Yeah, yeah. So yeah. So we talked a little bit

about the first team and the second team, I just like you to give me an assessment of the level of safety, as we discussed in this interview for the previous and the current team. So for the previous team, how how do you agree that it was not safe? Do you strongly agree, do you agree neutral? Disagree? You strongly disagree?

Participant 10 58:17
I mean, what do you mean by agreed? Like, is it? Was it good or bad?

Researcher 58:24
So how? Let's let's take it this way. Was it a safe work environment? So the statement is positive? Do you strongly disagree, disagree? Neutral? Agree or strongly agree? not agree. So you these are good. It wasn't safe. It wasn't safe environment. So you would disagree or strongly Disagree?

Participant 10 58:51
Disagree. Okay.

Researcher 58:52
And for the current team, do you strongly agree, agree?

Participant 10 58:55
Strongly Agree.

Researcher 58:58
Okay. Fantastic. Thank you very much. I really appreciate it was a very good discussion, and very good examples. Before we conclude, let you go. This is Sunday afternoon. So if there's anything you want to add in this discussion we didn't cover that makes a difference to be working in the safety environment and how does it help achieving quality?

Participant 10 59:31
We maybe would be nice to have like managers or management side who has technical background a bit, at least a little bit. Otherwise. For example, you understood me in a very short time, because you have technical background. If you don't, if you you have like business administration, it Will, it might be impossible to agree or to be on the same page?

Researcher 1:00:07
I agree with you the migration to Node JS wouldn't make sense to anybody with non technical background. It doesn't even, it doesn't even make the slightly. I don't know how to express the slightly remote idea that would improve performance and scalability. I agree with you. But do you think I think the good thing is that you have now is this level of self governance? Yeah, they give you a lot of freedom to make your own decisions?

Participant 10 1:00:37
Actually, that that's coming from the management side, we discussed that to do this we need freedom. We convinced them to do this, we need to make our own decisions about anything technical.

Researcher 1:01:13
Do you think they accepted to give you the self governance capability because they are open and they are willing to listen?

Participant 10 1:01:22
Yeah, that's, I believe, we agreed because they understood us because they had technical background.

Researcher 1:01:32
Okay.

Participant 10 1:01:33
That's the main takeaway, maybe because otherwise, it will be really important. I mean, it will be impossible to convince someone who doesn't understand you.

Researcher 1:01:53
Yeah, and especially these technical decisions that are really difficult to digest. Yeah. And especially, I mean, economically, sometimes they can be costly and costly, and sometimes it's difficult to digest. It doesn't help them to make the decision. Yeah, I know it's Sunday afternoon. I'd like to thank you a lot. And have a good afternoon. Thanks a lot for everything. I will stay in touch