[00:10] RESEARCHER:

Hi, how are you?

[00:11] PARTTICIPANT 18:

Hi Researcher.

[00:13] RESEARCHER:

Can you hear me? Sorry about this morning. I think I've sent the invites in the wrong time. I'm really sorry about that.

[00:22] PARTTICIPANT 18:

No, trouble. When I saw the time, I was surprised. I was like, okay, five am. It's not a problem to get up but no, everything is fine.

[00:36] RESEARCHER:

I'm really sorry again.

[00:38] PARTTICIPANT 18:

It's okay really.

[00:41] RESEARCHER:

Do you have any questions for me before we start?

[00:45] PARTTICIPANT 18:

I think I don't have. I saw the documents and the type of questions which you will ask me. And I'm perfectly fine with all of those.

[00:55] RESEARCHER:

Fantastic. If that's the case, let's start with the questions. The first question is just an introduction. If you can introduce yourself and your experience. Talk to us about your experience, what you've been doing, etc... so, briefly.

[01:15] PARTTICIPANT 18:

So, my name is PARTTICIPANT 18 and I'm in the I.T. industry and my experience is seven plus years. I've been working as a QA lead for three years and before that I worked as a QA analyst, QA software engineer. Call it what you want. It's the same role. I started in outsourced company as a junior QA. And there I spent two years. I started to learn the basics. And then I started to see what Agile is, since my first company we applied Agile at the beginning. After that, I changed my company and improving my testing skills in manual and automation testing. And after a couple of years I became the team lead.

[02:25] RESEARCHER:

Okay, fantastic. What do you think of Agile?

[02:31] PARTTICIPANT 18:

Agile is a super software development framework, let's say, which allows us to deliver faster and to allow our customers to be more satisfied and to see the progress of the project. Let's say faster. So, when we speak about Agile and Waterfall before, Waterfall projects took several months, even several years after they were finished. And with Agile, we can have small iterative periods after we deploy something and then the customer can see actually what we have done is that aligned with their wishes and with their expectations. And we can fix very, very fast what customer are expecting from us?

[03:31] RESEARCHER:

You mentioned fast delivery. Does it create pressure on the team?

[03:37] PARTTICIPANT 18

It shouldn't create pressure because if things are aligned properly and if they're estimated properly, teams shouldn't have any pressure.

[03:52] RESEARCHER:

But estimation in software engineering is very difficult?

[03:58] PARTTICIPANT 18

Yes, it is. It can’t be estimated properly always. But they're learning during the time. And if you are long time on the project, you are better in estimations because you know the core, you know the project, you know the technology. And your experience is much bigger. And with time you are better and better in estimations.

[04:24] RESEARCHER:

So, it's from the learning loop that you become better at estimating.

[04:31] PARTTICIPANT 18:

Yeah, exactly.

[04:32] RESEARCHER:

Another one, I'd like to touch on is customer satisfaction. How is the customer involved in Agile?

[04:39] PARTTICIPANT 18

The customer is involved in Agile via the product owners. So, we have a man or woman in the team which is called product owner and he or she is communicating with the customers and try to listen to the customers' wishes. Product owners then sits with the customer and let's say negotiates in which timeframe we can deliver something and then when we deliver that, the customer can see actually things if what they ordered is okay. Is it according to their wishes and if it is, the customer is very satisfied. But if it isn't, then we can fix it very quickly for the customer. So, it's not like he waited for a year or two to see the result of what we created. Actually, he can see very, very fast the output, how the product will look like and can impact already, for example, if you want something to change, something to improve, etc.

[05:50] RESEARCHER:

So, when the customer is involved, they usually don't understand software development, how it works. They keep changing requirements. Well, how do you handle that in an Agile sprint if they change the requirement for you?

[06:07] PARTTICIPANT 18:

It's important that the change of the requirements came in the next sprint so the current sprint can't be interrupted. So, sprints can be long, from two up to four weeks. In my company, we have sprints which last two weeks, and then it's important if the customer has some change requirements, we can improve it or change it inside the next sprint. So, we can estimate and see how it will impact in the next software development.

[06:41] RESEARCHER:

You said that Agile is super. It's a little bit optimistic. Too optimistic, isn't it?

[06:52] PARTTICIPANT 18

Well, it depends. Comparing to the waterfall, isn't super for the 21st century. Why? Because things are changing all the time. Technology definitely. Your customers want now and want immediately everything. And for nowadays it’s not acceptable to wait to the product to be finished in two years because competition will invent something you wanted. They will go to the market with a product you have or a better product. So, it's really important to be fast and to deliver with high quality. So, we don't have to wait for a long period of time to go to the market. So, we are going very fast market, see our customer feedback. If something isn't, would you change it, improve it? If it's good grade and we continue developing on the good grade product we have.

[08:00] RESEARCHER:

You mentioned quality, which is good. How do you define quality in the context of agile software development?

[08:08] PARTTICIPANT 18:

Agile advocates for quality but it doesn’t necessarily define it. My understanding is that quality in agile means clean code and sustainable design. Clean code is subjective, but we know at least it should be simple, readable and maintainable. We know a clean code when we see it! In agile, we also keen in creating software design that is sustainable; we can easily change it in the future to accommodate new requirements.

[08:30] RESEARCHER:

That's good. You mentioned something that I like. Which is deliver high quality. How does Agile deliver high quality? Your experience is Scrum, so how does Scrum help achieving quality?

[08:42] PARTTICIPANT 18:

Yes, my experience is all Scrum, But I prefer the word Agile. So, Scrum delivers high quality on, let's say, one way which I experienced. So, if you have sprints of two weeks and we have a small piece of code to test. It's much easier to observe the bugs and to observe the defects inside the code. If you have a small piece of code to test and a small amount time to deliver. So, when you have a large piece of code to test, let's say code, which was developing for several months, it's quite hard to test it perfectly. And some bugs can be found in the production. So, the quality is decreasing, but then you test small piece of code, you tested it regularly, you test it all the time. Then you have high quality and the customers are more satisfied.

[09:00] RESEARCHER:

But the piece of code is not always small, it's not always modular.

[09:04] PARTTICIPANT 18:

Oh, we are trying to be modular nowadays. So now we are going to implement best practices. Like if you have a model like application, you're going to split it into micro services. Micro services start small and they're scalable and they're testable, which is really, really important. So, if you'll follow the best practices and new technologies, then it's quite possible to have small code tested and delivered.

[09:39] RESEARCHER:

Do you think modularity is an Agile thing or a software engineering practice?

[09:45] PARTTICIPANT 18:

I think it's software engineering practice more than Agile.

[09:49] RESEARCHER:

Yeah, fantastic. Let's move to the next question. Can you describe your Scrum environment from your experience? You can pick up in any one of your projects and talk to us about it.

[10:09] PARTTICIPANT 18:

I can say that in every company I was working, Agile and Scrum were implemented differently. I didn't see that Agile and Scrum were implemented by the book. Well, every team implemented it in a way that was properly for them. For example, in my current company, we have sprints which are two weeks and we have a team as a dependent component. Every team has five to eight, up to ten members. So, by the book is like eight members max. But in one team we have ten members. We have a product owner, but we don't have a scrum master, for example. And if you observe scrum, we should have product owners, scrum master and a team which is independent of all of them. So, from my experience in previous company, for example, we had a product owner, scrum master and a team. But our sprints were lasting four weeks. If you observe, every company creates an environment which is best for them to work in which they're feeling comfortable and in which they can deliver best quality for the customers.

[11:40] RESEARCHER:

So, you are a QA lead. Can you describe to me a little bit in detail how QA is part of the team and how do they work in the team?

[11:55] PARTTICIPANT 18:

Yes, so I had my team as an independent component. So, we are all QA. It's four us, sorry, five. Every QA is dedicated to one development team. So, it's a part of that team, but it's a part of the QA team also. When you have QA inside the team, QA has multiple functions in the team. First and primary function is to find bugs and to keep code quality on the high level. Other function is to communicate with all members from the business side and from the development side. For example, QA gets some tasks to test, he can observe that some things could be improved. For example, product owner didn't remember which place one button should be implemented and then QA can communicate with the product owner, "Okay. Is this task complete? Should we improve it?" Its leading quality from the development perspective. Also, in front of product perspective. Also, QA is there to keep quality, not only test it manual and automation. You can do a code review for the developers to understand the code and to learn new things. So that's the role of QA in one Agile environment.

[13:35] RESEARCHER:

Is QA involved from the beginning from the beginning?

[13:38] PARTTICIPANT 18:

Yes, yes, it is. In my company, it is. We have sprint planning, sprint grooming. In sprint grooming, there is only the team leader of the team and product owner and they're discussing big features which will be developed in the sprint. The next day we have sprint planning where we have the product owner, QA, designer team as an independent component. And they're estimating tasks, which the product owner defined previously. And the developers are estimating their work and QA estimating the QA work. And after that, we assume its full estimation of one task.

[14:38] RESEARCHER:

How does it make your team member and you feel when you are engaged from the beginning?

[14:46] PARTTICIPANT 18:

You feel very proud, to be honest. So, we can improve some things from the very first beginning. And we know what to expect in the next sprint so we can plan our time, our KPIs accordingly. We become better familiar with the requirements and what to test.

[15:03] RESEARCHER:

How does this help quality?

[15:15] PARTTICIPANT 18:

We know what to test we assume less which help us identify real bugs and less bugs.

[15:18] RESEARCHER:

So you become more efficient at testing and you know what the business want better?

[15:20] PARTTICIPANT 18:

Exactly! That’s a good summary.

[15:21] RESEARCHER:

This setup that you explained to me, do you think it's a good implementation of Scrum and why?

[15:30] PARTTICIPANT 18:

It is working. I think it's not the best implementation of Scrum. I think some things could be improved. But it's like experience. We need time to figure out what is missing. Because sometimes, as you mentioned, estimations can't be a hundred percent sure and then it's estimated wrongly, for example. And then we have a problem with the product owner explaining that we estimated wrongly and that we need more time. That's the part that's tricky. So, when you cannot deliver on time. So, if you ask me the implementation is good, but it can be better.

[16:25] RESEARCHER:

How can we make it better? How can you make it better, for example?

[16:30] PARTTICIPANT 18:

To have more flexible customers and the product owner.

[16:43] RESEARCHER:

It's agile. It should be flexible.

[16:46] PARTTICIPANT 18:

Agile is flexible, but people are not.

[16:54] RESEARCHER:

Can you give me an example?

[16:57] PARTTICIPANT 18:

For example, Agile allows you in a small piece of time to change things faster. But then customers want something and it's really important to him to deliver up to the first day in a month. Then you really have to deliver until then. So, we are flexible in terms of changing things and improving things. But when it comes to deadlines and deadlines are not quite flexible.

[17:30] RESEARCHER:

How long has this team been working together?

[17:33] PARTTICIPANT 18:

Our team is working together... I'm in the company three years. But during those three years, we changed a lot of processes and agile implementations. The way that we handle tasks, how we handle deadlines. The problem is that people are changing all the time in I.T. They're going from one company to another. And then the problem is that you always have a pool of new people. People which need to learn the project, the technology, and the way that estimations are done. So that's the challenge in I.T. in general that you don't have people who stay for a long period of time in the company. So, like developers or I.T. experts stay in one company from two to five years.

[18:23] RESEARCHER:

How does it affect the Scrum process in place?

[18:28] PARTTICIPANT 18:

It's not like it affects it a lot but it is a challenge, to be honest. It is a challenge every time when someone leaves, you first have to find proper people to replace them and then to train them, then to educate them, then he or she gets involved in the project. It's quite a challenge, to be honest.

[19:02] RESEARCHER:

So, it affects the maturity of the team because the team, the longer they stay together, the more that gets mature and the process gets matured as well.

[19:12] PARTTICIPANT 18:

Exactly.

[19:15] RESEARCHER:

That's a very important aspect of any software development process.

[19:19] PARTTICIPANT 18:

Yes.

[19:21] RESEARCHER:

We'll move to the next question. Can you take me through the journey of a requirement or a feature from inception to release?

[19:34] PARTTICIPANT 18

So, the journey's like this. The customer first talks with the product owner and explains what is needed. What kind of feature is needed? So, the customer and the product owner speaks and then the product owner knows what the customer is expecting from us. The product owner is thinking of the feature and writes the feature. So, the product owner then explains the feature in some project management tool. We are using Jira, for example, or Trello or other management tools which they're using. They describe the feature. For example, as a customer, I want to have a yellow button placed on that page. So, the task is defined and then the product owner and the team lead has a spring grooming. The team leader is the most experienced developer inside the team. They observe the task and say to the product owner, what kind of obstacles we can see during the development. If everything goes okay, we go to sprint planning and other team members observe the feature description. And on the sprint planning, that feature is split in several technical tasks. So technical tasks are there for developers cause one feature can be implemented from several developers. One is the front end. Other is the back end. And one feature is split into a frontend part and the backend part. Estimations are done, the sprint has started. We start with the feature implementation. When developers finish with writing the code, it goes to a code review.

[21:58] PARTTICIPANT 18:

If the code review is passed, the code goes to the testing. QA implements testing phase. First of all, they test manually. Then if everything is okay, if the feature works completely, they write automation test cases. If the feature is broken, it doesn't complete requirements from the written task, we return it to developers for the feature to be improved. For example, a button is working on five pages, but on the sixth page, the button doesn't work. It throws some server error. We have in that period, we can have a ping pong effect between developers and the QA. Especially developers who doesn't want to listen and doesn't want to listen to the QA. So, they're not reading tasks properly, there's no detail during the implementation of that task so there can be a ping pong effect. It's not so often, but it can be observed. If the task is okay and it's tested properly, it can be delivered. It can be delivered together with other tests from the sprint or it can be delivered as a single module. It depends on how the team implemented the delivery, then tasks are delivered. The product owner informs the customer and the customer is observing what we prepared for them. And if they're satisfied, great. If they're not, they're complaining to the product owner on what can be improved. And in the next sprint to be are improving what we did in the previous sprint.

[23:51] RESEARCHER:

You touch a little bit on something very important, which is the QA and developer relationship. Can you elaborate a little bit further on that?

[24:02] PARTTICIPANT 18:

Sure. In some companies I saw and what I propagate actually in my team is that we have to have good relationships with developers. So, we shouldn't be people that are fighting, we should be people who are collaborating. Because every member of the team is responsible for the quality, team quality and the project quality actually. So, because of that, dev and QA have to work together, they have to work as a team and not as opposite sides. I saw in previous companies that developers are not satisfied when someone finds a bug in their code. They are not happy. And sometimes they ignore QA. But during this time and I think that's changed, to be honest. Me as a team leader have an important role to explain to people that are not opposite sides. We are one side which should collaborate together. And if the team lead propagates that we are working as a team, I think that things can be improved.

[25:27] RESEARCHER:

How do you do that?

[25:29] PARTTICIPANT 18:

Well, I have several approaches. So, I always say to my team members, "Don't just return the feature to the developer and write inside the task what is missing. Go to that person ask how they are feeling and explain the problem in person. So, face to face, make a coffee for him or she and discuss about the problem". QA has to have good people skills, so communicative skills. So, you need to find a way to approach the developer and to explain it friendly what is the problem because nobody wants to hear that something in which they invest their time is not working. It's bad. And we have to stop this way of thinking. And we need to explain to people that we are there to collaborate as a team and to find proper ways to approach those people and to make them listen to us.

[26:46] RESEARCHER:

It is difficult, yeah?

[26:48] PARTTICIPANT 18:

It is difficult but it's possible.

[26:56] RESEARCHER:

Do you think Scrum facilitates it, makes it easier?

[27:00] PARTTICIPANT 18

I think Scrum or Agile isn't involved in this.

[27:07] RESEARCHER:

Yeah, but it's it is collaborative.

[27:10] PARTTICIPANT 18

It's collaborative for sure. But I think it's more up to people, to be honest, up to their personalities. Are they willing to collaborate or are they collaborative persons?

[27:27] RESEARCHER:

But if they are not collaborative, they shouldn't be working in an Agile team, right?

[27:31] PARTTICIPANT 18:

They shouldn't be working in anything. They should be freelancers and work alone.

[27:40] RESEARCHER:

They should stay home, right?

[27:42] PARTTICIPANT 18:

Yes. Yes. I really mean that.

[27:44] RESEARCHER:

Yeah, you're right. Because everything is based on collaboration these days.

[27:52] PARTTICIPANT 18:

Exactly.

[27:53] RESEARCHER:

Or working in a research lab with rats and with insects.

[28:01] PARTTICIPANT 18:

They're a good company too, to be honest.

[28:05] RESEARCHER:

Let's move to the next question. What do you do to assure software quality in this Scrum setup?

[28:14] PARTTICIPANT 18

Okay, to be honest, I try to assure one hundred percent software quality, but it's quite impossible because there is no software without bugs. There is software in which parts are not a hundred percent visible. But software is that without bugs doesn't exist. I tried to explain to my people don't be stressed out. If some bug goes to production, we should learn every time from our mistakes. What we didn't see, what we can improve, so every mistake should be lessons learned for us. We are implementing several testing practices. When some feature comes to testing, first of all, we are implementing manual test cases and manual testing. Why is that? Because with manual testing, you're observing a product from the user and the customer perspective. You're observing functionalities and you're seeing the colors, paddings, and everything else which customers will see. If everything is okay with the manual testing, then we write automation test cases, which will help us in the next sprints. When we implement a new feature, which can jeopardize other code, We are letting out our automation test cases to be executed and we see this new code is impacting our own code.

[29:53] PARTTICIPANT 18:

So, if we created a new feature, is that feature breaking some other old stuff? There automation test cases are helping us. This is regression testing. We have monitor tools which we are using. Those monitor tools are following the CPU usage on the server side, memory user services on the server side and production errors, which we can see and observe and analyze during the time. And see from the DevOps point of view, so we have some misleading and some leaks on the system. Also including QA from the very beginning increases quality of the product because as I mentioned, QA can see some things and can observe some things which the product owner forgot. QA sees that product every day, a couple of times a day, and you know the product like your fingers. And you are the first one who can explain how that product works and can remember the places where some features can be important. So those are the ways we increase product quality.

[31:21] RESEARCHER:

Fantastic. I've noticed that you do mostly quality control activities to assure quality.

[31:33] PARTTICIPANT 18

Well, yes.

[31:36] RESEARCHER:

You know the difference between quality assurance and quality control? You are a QA expert. Why are there less QA and more QA quality control because I've noticed that in most software engineering teams, or software development teams they do, for example, code review. It is a quality assurance practice, but it is rarely a practice in closed source environments. Why is that, in your opinion?

[32:17] PARTTICIPANT 18

We are doing that more to learn. It's not like to control someone, it's more to learn. Because if you observe how other people are writing their code, first you'll learn development practices which you can implement in your automation test cases. That's the first step. The other is you learn from the code how the feature is implemented. And then, for example, this task isn't described very well. But you take a look at the code, you can see all the places the developer made the changes. And then according that, you can improve your testing phase. You don't have to go to developer and ask him, "hey, can you explain to me how you implemented this feature, or on which places you put this button? And I'm not quite understanding from the task." You can actually see that and learn from a simple request, you can observe from the code review. So that's my point of view. It's not controlling anyone. We want to learn more.

[33:30] RESEARCHER:

So, do you think this is Scrum setup you've been talking about produces quality software?

[33:37] PARTTICIPANT 18

I think it is, yes.

[33:40] RESEARCHER:

Why and how?

[33:43] PARTTICIPANT 18

Well, as I mentioned, when you deliver small pieces of code, an example, two or three features in a sprint. You're able to control and to follow all the bugs and to correct most of them. But for example, if you have a time frame of a year, and after a year you have to test something, it's more likely that you will pass into the production more bugs and more defects because it was quite a large period of time. It's a bunch of code. Then after fixing the bugs, you can cause new bugs and it's really a mess. When you have a small period of two or three weeks, you have a small piece of code which is testable, which is iterative, it allows you to produce better code quality.

[34:45] RESEARCHER:

All right. Fantastic. Can you share with me a positive story about Scrum and quality?

[34:54] PARTTICIPANT 18

Well, for me, every successful project is a positive story. I never worked in a waterfall environment, so I can't compare one hundred percent with Agile. I always worked in Agile. And if we finish our project on time, and the customer is satisfied, then that is definitely a positive story. So, if you are completing customer satisfaction? We are doing a good job.

[35:30] RESEARCHER:

Fantastic. What is customer satisfaction? How do you satisfy the customer?

[35:36] PARTTICIPANT 18

Customer satisfaction I can describe this way. We are implementing something. It's fulfilling their expectation, they pay us money, and they're continually collaborating with us. And that's the best description of customer satisfaction.

[35:55] RESEARCHER:

Unfortunately, things don't go well all the time. So, do you have a negative story to share with us?

[36:05] PARTTICIPANT 18:

Negative story is when we don't fulfill the client’s expectations. And I think the biggest role of the product owner is to protect the team when the team breaks the deadline. That's the tricky thing because if you agree with customers one thing and you didn't deliver it or you deliver it late, then the product owner has to be that person, which will be like a sponge. He will get feedback from the customer, but he will try to protect the team. It's his role to protect the team because he was there, and the estimations were done. He committed on some timeline. And then he is that person that needs to be on the front line between the customer and that team. So, the negative story is when you don't hit the deadline when you just break it.

[37:08] RESEARCHER:

But the in case of agile, you always have something to deliver in two weeks or in one week.

[37:18] PARTTICIPANT 18

So, for example if some big feature is in progress and it cannot be done in two weeks, but we said it will be. That's the tricky part.

[37:30] RESEARCHER:

Yeah. So that's misleading the customer?

[37:32] PARTTICIPANT 18:

Yes.

[37:35] RESEARCHER:

The last question. It's a little bit of a provocative question, but the purpose is not to create disagreement, but to get your opinion. What do you think of this statement: Agile produces poor software?

[37:57] PARTTICIPANT 18

I didn't experience that. I cannot say that it produces poor software. It produces good software. It's software where you can have impact all the time. You can change things fast. Maybe you are making some mistakes in that period. But definitely it's good software. From my experience, really. I worked on several projects which were very successful, and they've been created in Scrum.

[38:34] RESEARCHER:

So, what's the commonalities in your opinion? Because you've been working for successful projects. What are the commonalities across this project? There must be something common across this project?

[38:47] PARTTICIPANT 18:

People collaboration, team engagement, because without that, you cannot create successful product.

[39:00] RESEARCHER:

So, what is team engagement? What's the difference between engagement and collaboration?

[39:06] PARTTICIPANT 18:

So, when you have a collaboration, it's like we can speak properly. We respect each other and we can collaborate. But when you have team engagement, you're engaged to some deadline and to some delivery. And they're working as a team. So as one mechanism. They are engaged in a way that they will do anything to reach the target, to reach the deadline. They will work overtime if it's needed, and they will complete their engagement and their estimations. So, it's not enough sometimes to work eight hours. In Agile, if you miss estimations, then there is some pressure. But if you have an engaged team, then you don't feel that pressure because everyone is working as one person, as one mechanism. And if you have a team collaboration and team engagement, then you will create a successful product.

[40:15] RESEARCHER:

Some people told me that a poor Scrum implementation can produce poor software. What do you think?

[40:25] PARTTICIPANT 18

Maybe that can be true in some instances.

[40:35] RESEARCHER:

So, if the implementation of Scrum is not good, then the software will be poor, right?

[40:44] PARTTICIPANT 18

Exactly. So, we changed several times our Scrum implementation until we saw what is best for us. That's the best thing about Agile in general, it can be changed any time.

[41:00] RESEARCHER:

Keep you keep learning and you keep adapting it to your needs until you reach a level of maturity with the process, right?

[41:08] PARTTICIPANT 18:

Exactly. That's the point.

[41:11] RESEARCHER:

How about the people maturity as a software developer or as software professionals?

[41:19] PARTTICIPANT 18

It's quite different. Let's say six years ago, I knew very serious software developers. They're thirty plus, forty plus years. Now, they were mature in their heads, their minds. Now I can see a bunch of people which are not mature. They're like nineteen, twenty years. They're not grown up. They don't have the maturity. And they're like children. They want to code, but they don't have that team engagement. They don't have team responsibility. They're late for the meetings, they not fulfilling their work on time. They're good coders, but their company engagement is a disaster. They are like kids, they don't have that engineering approach.

[42:21] PARTTICIPANT 18

So now you can see everything. You can see good software engineers and you can see coders. Coders are like people who don't have engineering minds. They only type code, they don't have responsibility, they don't respect their colleagues. They're late for the meetings. They are committing to some work that they cannot fulfill. But from the other side, you have software engineers who are open, who are precise, who have architectural points of view. They're responsible. They never commit to work that they cannot fulfill. They're not late for the meetings. And you have two totally opposite sides.

[43:09] RESEARCHER:

It's fantastic. That brings us to the end of the interview. Thank you very much.

[43:15] PARTTICIPANT 18:

Hopefully, I helped.

[43:19] RESEARCHER:

Yeah, it was very helpful. It was very insightful. Thank you. Do you have any questions for me?

[43:25] PARTTICIPANT 18:

Can you tell me about the research you're working on?

[43:30] RESEARCHER:

Yes, I'm interested in software quality in general, but this time I'm looking exactly at how agile deals with quality. Software quality is more than testing. Testing is not the only measure that we use in software to produce quality. There is the human factor, there is the process factor, and there is the social factor as well, which you talked about a little bit, which is the relationship between the QA and the developer, for example. That's very interesting. We're looking at these enablers and these attributes that help in achieving quality in Agile. If you look at the manifesto, which started everything. the Agile manifesto, it doesn't make clear reference to quality. It doesn't. It's more highlights and guidelines for the process, but it makes reference to excellence, for example, but excellence is not necessarily quality. You could have a very high achieving team, but they don't produce quality. So that's make us question. And when we look at the scrum guide, the scrum guide doesn't make reference to quality either. So, we question for example, if the father of everything, the scrum guides, the manifesto doesn't call directly for quality, then how do Agile teams in practice make quality. We like to understand that. Your interview was very insightful. There was a lot of pickups from it and we will analyze it and make conclusions.

[45:50] PARTTICIPANT 18:

I'm glad that I could help. I read some books which are very interesting. Have you seen the book Agile Testing by Lisa Crispin? Let me send you the link. It could be helpful for you.

[46:21] RESEARCHER:

Okay, sure.

[46:31] PARTTICIPANT 18:

It has two books. Agile Testing and More Agile Testing.

[46:47] RESEARCHER:

I'm reading her first book, which is Scrum. I will read this one next. I like her writing.

[46:57] PARTTICIPANT 18:

She is full of understanding and I listened to her at some conferences and she is really good.

[47:10] RESEARCHER:

I like her books. They are straightforward. Thank you very much. I wish you a really good day. And I'm really sorry about earlier.

[47:21] PARTTICIPANT 18:

No, really everything's fine.

[47:23] RESEARCHER:

Thank you PARTTICIPANT 18. Have a good day.