**Interviewer:**

Okay, so hello again, we are here at Startup 3. We are starting the interview. It's Startup 3.com, right?

**Interviewee:**

Yes. Startup 3.com.

**Interviewer:**

Okay. We are gonna discuss a bit about software engineering practices. Okay. I can get some information from here, but I'm with X, which is the CEO of the company, we would love to discuss about software engineering practices, but at the beginning, I would really love to ask some general questions regarding the start up. So can you tell us a bit more, what is your core product or service that you're providing?

**Interviewee:**

Yeah. So what happened is, as I mentioned, I got my PhD from University of Michigan. So we are experiencing getting a traditional program and accelerated like 10 to hundred times by using graphic scores or GPUs. Okay. So that was all my PhD was about like getting a traditional program that people write and then converted to utilize the graphic scores. So that way we accelerated. Okay. And then we start looking around to see who needs this type of acceleration, who needs to finish their program faster. Everyone wants that, but who is really in real need. And we found out that genomics market is a good target for us because you have lots of, lots of raw data and that raw data needs to be analyzed to find a meaningful result out of that. And that thing takes like two, three days right now on a big CPU server. But by replacing that CPU server, with a GPU server and using our solution, we can do the same computation in around 45 minutes. So two days to 45 minutes, that's what we provide.

**Interviewer:**

Okay. Yeah. That's very cool. When did you start?

**Interviewee:**

In 2015 officially, but on this project 2016.

**Interviewer:**

Okay. Very good. And you are located at Ann Arbor?

**Interviewee:**

Right? Yes. We graduated from University of Michigan and stayed here.

**Interviewer:**

Hmm, very cool. So I don't want to imagine, I know the role now you are the CEO, so I don't have to ask what is your role? What is the type of ecosystem you are working right now? Is this incubation center?

**Interviewee:**

So its called startup garage. So what Melo did, Melo innovation is a company here. They assign some part of their company to startups. So they provide office space internet, electricity.

**Interviewer:**

Any consulting?

**Interviewee:**

No, but if you want, you can use, use their services. So providing good interface and stuff like that, you can work with them to do that. But nothing comes with being in startup garage. What we get is space and all things related to that.

**Interviewer:**

Cool. How many employees are you right now?

**Interviewee:**

So we are right now six full time. Two part-time. So eight.

**Interviewer:**

What's the gender balance?

**Interviewee:**

Unfortunately it's all male.

**Interviewer:**

I was guessing you'd say all female.

**Interviewee:**

Unfortunately. All male.

**Interviewer:**

And what is the average employee age like between their thirties? Twenties?

**Interviewee:**

So average is around 30 something, but we have people with 25. We have people with 55 years old

**Interviewer:**

Listen anyone between the 35 and 40

**Interviewee:**

Me and X.

**Interviewer:**

Okay. That's good. Can you tell me a bit of team composition? Like you are doing some are technical.

**Interviewee:**

X and me, we are half and half. We are doing half development and half business. Okay. Those two part-time are both business and the rest of four are developers. So it's half and half.

**Interviewer:**

Very good. Okay. I got some demographics, I would say I'm just gonna pause now because it's easier to transcribe and I'll start the second part of the interview, which are a bit more specific question to software Engineering practices.

**Interviewer:**

Okay, so here we are again I would like to ask a bit of open-ended questions because I don't want to structure things like I know everything. I don't know everything. I know part of it. So what are the software engineering processes, models, methods, and tools that you are using right now? I said a lot, but you can manage what are the development practices that you are?

**Interviewee:**

So there are multiple categories. One is when we are developing, we are using different ideas, like interface tools to when we are writing the code. We use VS code for C plus plus that we are working, we use net beans for Java code that we are doing and brackets we use when we are doing some website developments. Okay. So these are ideas that we use for doing. For Versioning we useGitlab we are some GitHub, but we are mostly on Gitlab. For acceleration these are all C based. These are all C based, the main languages that we use is CC ++, and Kuda. Kuda is the language specifically designed for GPU, which is C ++ with some keyboards.

**Interviewer:**

Okay. Yeah. So it sounds like that. Yeah. How about the programming schedule? How do you manage the past.

**Interviewee:**

So what happened is we have the baseline code, so it's a little bit different from other companies. We have a baseline code that some biotic wrote it, which is slow, and we want to accelerate it. There are multiple steps in it. So if they are in Java, in Python, we convert it to CC plus plus. So that's one task that we need to do when we do it in CC plus, plus what we do, we do all these performance analysis to find a bottleneck. What is, which parts of it take a long time? And then first we start optimizing those parts by using normal CPUs, like SDL threads or pets. And if the performance is not what we want, we convert those parts to the GP code and we write good and code for it. Then, as you can imagine, now that bottleneck is not bottleneck anymore because it's running much faster. Now, bottleneck moves somewhere else. Then we go back and do this process again. And then we go back and do this process again. And sometimes bottleneck might not be in the computation part. It might be reading from this. It might be accessing memory. So we have this large number of optimization tools that we use to make sure that if for each bottleneck we decide based on that bottleneck, how to optimize it.

**Interviewer:**

Yeah. Who is doing project management.

**Interviewee:**

So, so each one of us has their own project, so we have different set of project and right now, but I will oversee all the projects going forward.

**Interviewer:**

Okay. And these are projects for clients I imagine. Or is it, what, what are you optimizing? Which quote for whom.

**Interviewee:**

Oh, okay. So there are multiple standard tools that everyone is using. Okay. So we are optimizing those and selling those to everyone. Okay. So it's a software company. It's not a consulting company. I see. So we make the software and sell it to the,

**Interviewer:**

So you don't have like customers that are coming to you and asking

**Interviewee:**

There are few customization that we do for some of the customers, but mostly it's all product based and sellable.

**Interviewer:**

What's your project management wise. You don't have any specific how to follow up for instance on how, how is doing and yourself or the other developers. How do you follow up?

**Interviewee:**

With the developers or with the customers,

**Interviewer:**

With the developers, with the customers? Do you have some practice in place?

**Interviewee:**

So with the customers, we usually doing it through email. So they send us the customization and we have the list of task. We don't have really fancy project management software. Okay. So we use Google tasks and assign, assign tasks to everyone.

**Interviewer:**

Yeah. That sounds fine. You mentioned a lot of things, programming languages too. So I don't have to ask that. How about testing? Are you doing testing of the software?

**Interviewee:**

Yes. So what sort of testing? So everyone is responsible for testing their own product, so they do it all the time. But any release that we have, we have a regression test that we started before the unit level, right? Yes. At unit level and we test, so it's not one product, it's a framework that has all these pieces. Of course. So we, we have all these scenarios that different things might happen. So we test all of them during the weekend on Amazon cloud. So we started come back on Monday and see if anything fail every weekend with new developments that we do. Okay. And everyone who is adding a new product to this software suite, they will make the test cases to add to regression. Yeah. Okay. So even the test cases to add to the regression, they make it and then regression now is getting bigger and then we do it again.

**Interviewer:**

Okay. Sounds cool. Do you have like I would say documentation for the testing and for the development, do you write some documents?

**Interviewee:**

Or yes. So we have a customer facing documentation. Yeah. And we have our own inside documentation, like what to do, how to do it, stuff like that. So one of the main purposes of is to warm up, whoever joins the company, like at the beginning, they need to look at that and read and learn how we do stuff and how to do the testing, to do the code writing stuff.

**Interviewer:**

And this documentation everybody is still responsible for is part of the project or the code, or do you have somebody that is responsible for it?

**Interviewee:**

Everyone is responsible for doing their own documenting.

**Interviewer:**

Yeah. Either being code level testing level or…

**Interviewee:**

Yeah. So, so as I mentioned, each one of us has their own product. Okay. So you are responsible for developing testing documentation and debugging when the customer comes back and say, this is not working. Yeah. That guy is responsible to do that, but at the same time, and I are part of all of the projects too. So we have in enough knowledge about all of the projects that if, if the guy is busy or something, we can do it or something. Yeah.

**Interviewer:**

Have some insights. Yeah.

**Interviewee:**

Yeah.

**Interviewer:**

Makes sense. How do you protect the artifacts? You have all this documents being generated called being generated. You store it somewhere on the heat lab, but on the long run, how do you protect some things that you deem? Because the software, I think it's very relevant, right? That you are selling, you are accelerating yourself from one week or two days to 45 minutes. Yeah. So that's a big gain. I would say if somebody needs to run some complex algorithms and, and so on, and I think it's very beneficial, but then you are spending time, right. And you are producing artifacts, like code testing code documents that you write for the customer or whatever other things you're doing. So how do you protect these things? Like you put them on a safe repository.

**Interviewee:**

Okay, perfect. Yeah. So for repository, it's all private. So anyone has access to their own product repository. So that's how we basically manage who has to watch repository. But in terms of delivery to the customer, everything is executable encrypted and inside the Docker. So everything is containerized. So you don't easily have access to the binary. And even binary is encrypted and we are giving the encrypted binary to the customers. I see. So they

**Interviewer:**

Can compile version.

**Interviewee:**

Yeah. Yeah. Even encoded compiled version.

**Interviewer:**

How about the, I would say the part that you have, like I piece on the code because you are working eight people or six people and say development wise. So you have some proprietary rights on the code, right. As a company. How do you deal with that? Because let me rephrase why I'm asking these questions. We teach a lot of things for in software engineering students, but we don't really teach the, the value of what they are producing in their courses. Sometimes like they're producing this piece of software. How are you? Are you going to protect it? So do you have an approach that we can sort of integrate in our courses or

**Interviewee:**

So it's, it's really hard to do that. It is, it's, it's really hard to protect, but yeah, I, I don't think we have any good way of protecting the intellectual property, but what so the code that they write, they can take it some like our developers. So we have all these contracts noncompeting to IP contracts, but yeah. If someone wants to do it, they can do it. Nothing stops to,

**Interviewer:**

Okay. But your team is valuable, right. For the company. It's a value

**Interviewee:**

To yeah. So, and if you want to push too many of these limitations to the team, then working together will be really hard to do so you have to trust each other. Yeah. We trust each other. We trust everyone

**Interviewer:**

Let's accept that part. Meaning that that part is not considered like you are a team and you trust each other, you are working together. Yeah. But what I I'm talking about is what are the things that you mentioned a little bit, the things that you don't want to tell or don't wanna give away to the end customer or to a third party, and you said, I'm compiling the code. I am encrypting the code, et cetera. And this is a protection way I would say to, to not give away the source code, which has more work done behind it. And then you said, I put it on GitLab. It's private, right? Yes. But let's say somebody sort of want take that code or takes that code because you see breaches the security of GitHub and he has access to your SU or gets your password. And this is not within your team because I understand your team. Yeah. You are working together. Yeah. It's a third party entity.

**Interviewer:**

So do you have any way of protecting yourself or?

**Interviewee:**

So, yes. So the, I think the main protection against Sumter party take our code is our relationship with the customer. Okay. So what happened is having that code is useless without that relationship because we are up keep updating the same code and, and the relationship with the customer is when they, they see anything that they want to come back to us. So that code two months ago, now it changed a lot from two months ago, it's updated based on customer preferences and it's updated with the new piece of code. It still, it hurts that someone else has access to that. But we have this roadmap ahead of us that we are keep adding more and more features to our softwares, to it, which makes the old version obsolete. Yeah.

**Interviewer:**

And how do you ask, which is important and which is, is not to, to protect. How do you say that? Okay. This piece of code is important. This is maybe not important. We use the open source library and we can share back. I don't know.

**Interviewee:**

So some, some of the things that,

**Interviewer:**

Or this document is classified, this is less classified. How do you ask?

**Interviewee:**

So based on the complexity of the optimization that we do. Okay. So more complex is means we should protect it more, but if we do something easier, and we think that by making it open source, we can give back to the community. Okay. We do that.

**Interviewer:**

Okay. You do that. Yeah. Even when you take some code, perhaps somewhere it from type overflow and then you update it because you find the issues and

**Interviewee:**

Yeah. And you put it back. Yeah. And we keep giving feedback to the open source community. Okay. So like, if we find a bug, if we find a better way to do things, we provide all those comments to the open source.

**Interviewer:**

If I say technical debt.

**Interviewee:**

Yeah.

**Interviewer:**

What do you think?

**Interviewee:**

Technical debt. Okay.

**Interviewer:**

Technical debt is like this. Sometimes you stick to good practices because you know, these are the good engineering practices, but they might take longer than you think. Yeah. And then you have to find a shortcut. Yeah. Like a faster solution to things that you are doing. But that shortcut might also mean that your code is not that robust. Yeah. And it might have inserted some box that you are not aware of. And if you do this, like a, a lot of times, then you might end up with a software that is very buggy and it costs more to fix it. Yes. Rather than create a new one. Yes. And this is the technical gap that you get. Yeah. Okay. You cannot repay it with money or whatever, but you have to sort of start everything from scratch, because this has a lot of issues as a code because you took a lot of shortcut and you didn't follow to good software engineering practices. Yes. So what do you do in

**Interviewee:**

Regards? So at the beginning of the small startup company,

**Interviewer:**

When you say beginning, small start is your startup.

**Interviewee:**

Yeah. Or startup. So I think, but it's, I think it's valid for most of the startup that start. Okay. We took a lot of shortcuts. So it was all about shortcuts to get to the first prototype sooner. Okay. So at that point, we didn't care about robust engineering and wanted the minimum viable product as fast as we can, because we wanted to prove ourself like with something that is boggy, but works in some cases just to prove what we can do. But now that we are getting bigger we are trying to use good software techniques, like to make sure that everything is robust. We are trying to make all the codes look like each other all the different guidelines that we need to follow. We are now sure we can develop anything but not everything within the given time limitations. So. Okay. Yeah. So, I think at some point you need to take shortcuts still. Yeah. To move things faster.

**Interviewer:**

Okay. So the technical debt can be a good and a bad thing.

**Interviewee:**

Yes. Based, based on the situation. Yes.

**Interviewer:**

And perhaps you take some trade-offs.

**Interviewee:**

Yes. We still, of course like a company, like Google should not take a shortcut at any point, because we don't know.

**Interviewer:**

I haven't been to those guys. Yeah. But I can check. Yeah. Yeah. I'm pretty sure.

You, you are right. But you never know.

**Interviewer:**

So When it comes to employees, I, I wanna ask this because you are hired already eight people, you are. Yeah. What are your expectations? If there is a new employee intern that could to you, or we send the students back and forth and we'd love them to work for startups to work for companies and what do you expect from them?

**Interviewee:**

So they should, so one of the most important things for me is they should be able to break the problem into smaller problems. Mm. So if you always have…

**Interviewer:**

So you say problem solving…

**Interviewee:**

Problem, solving, like how they do problem solving. You always have big problems, but if you break it into small problem, it would be much, much easier to do it. And you see the progress yourself. So you are happier because you feel happy because now I'm fixing this problem, fixing that problem, fixing that problem till the end. You see, there is no big problem. It's all fixed. You

**Interviewer:**

Want to, you want them to work like GPUs.

**Interviewee:**

Yes, exactly. So that's really important for me and communication. Communication is a big, big thing for us. When

**Interviewer:**

You say communication is like teamwork,

**Interviewee:**

Teamwork, skills, teamwork. No, just teamwork. So, okay. I don't want anyone talking a problem for a week and I don't know about it. Okay. So if, if you have a big problem and you cannot solve it, just let me know. I ask I either I have a solution or I ask someone else. So talking about problems inside this group is really important.

**Interviewer:**

How do you do that? How do you talk with pro about problems? Do you have daily routine or…?

**Interviewee:**

So I, I usually go around, as you see, you are sitting in a circle type of thing. I usually go around and ask them whats of what's happening, like, like once a day.

**Interviewer:**

Okay. That's cool. Are you as a startup because you have been some years now are you goal driven or resource driven and I will make an example

**Interviewee:**

About this. Okay. Yeah, please do. It's hard.

**Interviewer:**

Go resource driven. And goal driven are more or less like this. When you have goal, you know, that you stick to that goal, even though you don't have the resources and when the resources are lacking, most of the time they do, when you are a startup, then instead of changing a goal or optimizing goal, you just try to find more resources to get there. Okay. And a good example of this would be like, I am building this software, very nice software to optimize things, but we need a lot of skilled people. Right. We have a couple, but we need a couple more. What do we do? Do we take a shortcut? We do a seminar software, which has less benefit or do we sort of wait to find more budget, to get more people involved that have the competence and build the right software or instead resource driven. It'll be like, okay, we have this benefits, short term benefits, like, okay, we did a version MVP, it looks good. It can be better, but we wanna sell it right away because we think it gets us some resources, even though it's not our final goal, we didn't want to get that minimum viable product to be sold. We wanted something better and we needed to wait. So what drives

**Interviewee:**

So we definitely want to be goal driven.

**Interviewee:**

We want to be goal driven and we consider ourself goal driven, but sometimes because of lack of some resources, you there's a trade off. So you sacrifice some of the things that you want again. But we are always looking for more resources to get to our goal. So it's and looking for more resources it's happening right now in our startup. So I would say we are goal driven.

**Interviewer:**

Okay. So when you started, you had a goal. What I could imagine from the first background conversation, you said I had a PhD and then I was working on this. I created a startup. And this was because I had a background on kind of optimization on the point. And it sort of, it was a goal to make programs run faster. Yes. Right.

**Interviewee:**

That was the goal for me personally, but for our job is provide better data as the fastest pace that's possible for, to bioinformatician. Okay. So that's our goal. So we do whatever we can to provide that for bioinformaticians.

**Interviewer:**

Okay. So whenever you take an important decision, key decision, you sort of if you have the resources, you stick to the goal, but if you don't have the resources, what do you do

**Interviewee:**

So that that's, That's become problematic sometimes, as you said, you, you define it. You define my list on, instead of the final goal, you define my, and try to get there while you are looking for more resources. So basically the same way that we break problem, we break the gold into smaller pieces and try to get to each piece at each time. Yeah,

**Interviewer:**

Sure. You break in several objectives.

**Interviewee:**

Yeah, exactly. Okay.

**Interviewer:**

Yeah. I see. I have asked a lot of things I believe. We have mentioned a lot of stuff. I don't know if you have some remarks regarding something you'd like to say, like, I can ask a bit about this KPIs key performance indicators that you have within the startup. Like do you document this KPIs as well? Like you document the software and the code and the,

**Interviewee:**

So, so we have on the market, it's

**Interviewer:**

A bit business question, but

**Interviewee:**

Yeah. And the marketing size, we have some so basically we released this press release. How many people, how many visits came to our website, for example,

**Interviewer:**

Do you document this stuff?

**Interviewee:**

We, we have all those. So basically the Google keep it for us, but okay. But we have some understanding of what works, what doesn't work. So we do webinars, we do white papers. We do different type of marketing and we compare the, their efficiency with each other. Yeah.

**Interviewer:**

Well, one final thing that I'm wondering, because I didn't ask this earlier, you have a code that you are developing, right. And you said it's tailored to the customer and then yeah. Nobody can take it or whatever, unless it comes some not useful. But are you adopting a bit of reusability in the code quality assurance that you can sort of verify, validate you do this because you test, but reusability of the code. Have you thought about it or

**Interviewee:**

How visible for yourself? Yeah.

**Interviewer:**

For yourself. Yeah. Developing version one. You 1.1, 1.2.

**Interviewee:**

Yeah.

**Interviewer:**

So are you using code how that is managed?

**Interviewee:**

Yeah. So at again at the beginning, it wasn't like that, but right now we are, we are doing our best to not write the same code twice. Okay. So that's one of the good software design approaches. So right. Each code once. So we are extracting these common patterns in all of our tools and we provide this standalone tool that can be called from other software. So we are trying to make it modularized. And now everyone who wants to work on a new product, they can use these modules to.

**Interviewer:**

Okay. So when you are writing these modules, do you document them right away on the code for usability purposes? How do you document?

**Interviewee:**

So there are three type of documentation. One is right. Think the code in a way that anyone looking at it, they understand it. Okay. Like choose a good variable name, choose a good structure, stuff like that. The other one is put comments. Okay. Inside the code. And the third one is update it on our VicU page, basically how to do it. So we try to do all three of them, but sometimes you are latching and here, and then

**Interviewer:**

I see. Yeah. That's good. Well, I don't know. I think we have covered a lot result if you have something else to tell me. So otherwise we can something you'd like, no,

**Interviewee:**

I'm good. If you have any more questions, just contact me.

**Interviewer:**

Yeah, sure. I will. If something misses, but hopefully not. You have said a lot of good information, I would say. I mean you can, you mentioned a lot of stuff. I have one exotic question. Okay.

**Interviewee:**

Go ahead.

**Interviewer:**

It's called metamorphic testing. Okay. Have you ever heard about it? No. No. Especially for companies doing AI development artificial intelligence, machine learning, et cetera, like these Tesla cars that you might have heard of. Yeah. They are pretty autonomous, but sometimes they crash. Yeah. And this sometimes sort of happens in a strange way. And one of these strange ways is because you get a big truck with a wide facet

**Interviewer:**

And the car doesn't distinguish this big face in front of it. If it is actually the sky or a blue face or whatever, is it the sky or is it just this big truck in front of it? And it just crashes. And sometimes testing is not that you mentioned, we do regression testing, unit testing, white box, that box, perhaps doesn't matter, you have a coverage of the code, et cetera. But sometimes testing is not that obvious because you have some inputs and you get some outputs and those outputs need to be compared against an Oracle. We call it Oracle. Yeah. For comparing, okay. Is the result correct? Or something went wrong, but this results are sometimes not that trivial. The Oracle is not that trivial to say, this is gonna be that or comparing just to numbers. So basically with metamorphic testing, you put some thresholds like doing a simulation and you say, okay, for this set of inputs, we know that the output should be on the left side or on the right side of this threshold. And if it falls in this kind of reasonable outputs, then we say, okay, then things are good. Otherwise there is some ambiguity and we need to figure out what is producing this ambiguity, which is usually a bug. Yeah. Right in the software. So metamorphic testing, it sort of works like this. It looks at patterns. It look how the cold is behaving at the end. And then you try to reason based on the outputs, rather than just having a fixed Oracle saying, oh, the result would be 10, but it's nine, 10 and nine are not equal to each other. And we figured that out. But no, it has a threshold you say on this sort of range of values is good. Outside of that range is not good. That's why it's sort of called metamorphic rather new. It's becoming part of the finding out and discovering scenarios that have not been discovered or are not able to be tested with regular classical test cases.

**Interviewee:**

So, so far we had this fixed Oracle, so we know what, so we have a baseline curve,

**Interviewer:**

Where do you take it?

**Interviewee:**

There, there is a code that we call it, the standard. Even if that code is wrong, our code should be wrong. Okay. So we, we okay. Because everyone get is used to that code. Yeah. So even if it's wrong

**Interviewer:**

Has never occurred that there are some scenarios that you don't understand why it's not.

**Interviewee:**

Yes, but everyone in the field, they learn how to deal with it when it's cause problem. And if you don't cause that problem, they don't know how to deal with it. So we even imitate the problems in the original code. Okay. Just, but we are getting in this game of improving accuracy right now. Okay. So we are getting better. We want to get better than original code. So in that case, yes, we don't have the, so, and we need to look at the result and basically understand it ourself, if it's good enough result or not.

**Interviewer:**

Okay. Maybe I'll share something after

**Interviewee:**

That. Okay.

**Interviewer:**

Thank you so much myself. Yes.