**Interviewer:**

Okay, so I wouldn't say good evening, but hello, X from Startup 2, and we are here to talk a bit about your startup first. Let's have some general question in the first part of the interview, and then some more specific questions related to Software Engineering practices you are doing. So, first of all what is the main service or product that you are developing at Startup 2?

**Interviewee:**

So the easiest thing would be to show you our application and kind of walk you through how our customers use that. So Startup 2 comes in solving a couple looming problems with job shop manufacturers. So let's say you're an engineer at Boeing or an engineer at a small startup, or anywhere between you're designing a new component for some system and you wanna prototype, or you want to enter a short run production nine times outta 10, you're not gonna machine that yourself. Nine times out of 10, your company is not gonna machine it. You're gonna outsource that to a short run production, company call the job shop. Sure. Those job shops are facing a serious labor shortage where it takes them, or they rely on trade skill machinists, and engineers to prepare the design files that they get from their customers into formats they can action the machine with. Takes on average, about 20 hours of highly skilled labor. And even if they can do it and they can't hire enough people, it's still a total pain. They can't hire anyone else because their engineers and machinists are just retiring. They're all like 55 or older, and there's no new generation stepping up that has this trade skill built up as an apprenticeship. So where Startup 2 comes in is we eliminate almost the entirety of this problem. Actually the entirety of the problem, we eliminate almost the entirety of this process. And the way we do it is we have an automated service where job shops, as soon as they get the first contact with their customer at Boeing or whoever else, they get the design file. They upload to fluidity. A couple minutes later, we give them all the data they need to immediately enter production with, and they can use that same data to also reply back to the customer, or that first request for quote, supply, a very, very specific, detailed, highly intuitive quote yeah to the customer. They know how much it's gonna cost 'em and what they should charge for that, and get a contract rolled out in a couple of minutes, a couple of hours, as opposed to days or weeks. So to show you exactly what this looks like, give you kind of, this is our itinerary page, we just rolled out, walk through the whole process from first customer contact the job shop uploads, a 3d CAD file, computer design file to Startup 2. We process it. Oops.

**Interviewer:**

You have to authenticate.

**Interviewee:**

Yeah, this is a demo. So it should just work fine. So customer uploads into Startup 2 we process it in which we get data back to the customer and they can send a quote back to their customer bone, whoever else. Once the contract is in place, the job shop can procure the tools. We tell 'em exactly what they need. As soon as the tools are delivered, they can start production, walk them through step by step, the change over time period for actually equipping the tools machines to machine part, what the cycle time is for action machining, each of the different steps and all the operations to that. Then if they need to refix that rotate the part in the machine, we tell them an estimate of the labour time for actually doing that. And all of these metrics are based on our simulations of running that program that we developed, okay. Our own virtual machine tools that we provide back to the customer. So these are almost guaranteed to be spot on for what it's actually gonna cost them.

**Interviewer:**

What's the error margin for this?

**Interviewee:**

We honestly just don't have enough benchmarks for it. And one of the reasons we haven't put the effort in for benchmarking what their margin is, it just doesn't matter to the customer. We say three minutes, 30 seconds. If we can say it takes one minute, between one minute and five minutes or one minute and six minutes, that's good enough. Okay. Their production lines are so short, it just doesn't make that much of a difference. They will wanna know, is this gonna take five minutes or two hours? So this is what this looks like on a per set-up basis. I don't know why this isn't loading. It was doing the same thing this morning. So this is the original CAD file that Boeing would've supplied the customer with. They upload it to Startup 2 and this is the interface they get. So you can see the CAD file. They can see the machine version, see if there are any changes, parts that we couldn't actually machine. Okay. They get some general contract data, quantity and whatnot. They also get this, hit that button again.They also get a bill of materials, which tells 'em exactly what tools they need to pull out of their tool crib, or if they don't already have that tool and it's needed for that part, we give them one click ordering button to procure it and have it drop shipped from one of the preferred vendors. We do the same thing with materials. We spec out the size of, let’s say this block of aluminum that they need, and they can order it directly through Startup 2. We also spec out their work holding. So this is using our [inaudible], and it does not need any custom fixtures, but if it did need custom fixture plates or soft draws to fit in that work holding device, then we’d tell them that and then get them instruction, how to machine those also. So let's go back to the itinerary page and go to setup one. This is what the part, this is just a solid block of aluminum. It's what it looks like at the start of the first setup, by the end of the setup, this is what the part will look like. Walk through that, we tell the customer, you need to load it in this specific device that we already called for. And that's how it'll end up on the end of that setup. We Give Them a list of all the tools that, we call for, for this specific setup and where to load that into the machine. And they give them a step by step list, for each of the operations so they can see what's at actually going. So in this case, the first operation is a facing operation. We have the tool path for machinery that overlay. We have some drilling operations in there. We have some pocketing operations of that centerpiece And then some profiling on the outside here. And the output for each setup is the input for the next setup. You can see the start of this setup, we start with this part in this format. This is how it's in the work holding. And by the end of that setup, we'll have something looks like this. And again, same tool, different tool table, perhaps maybe the same, this case is just the same. But all the operations are different. So for instance, the first operation is facing again. Then we have a pocket cut in there for all these overlay tool pass to generate, give, overriding parameters. So if you wanna change the rotating speed of the cutter or the feed rate engagement rate, you can change all that. And then each of those cases it's compile into the code that'll actually run in that tool and everything I've showed you up to this point is free of charge to the customer. The only thing we actually charge them for is the code when they're happy with how it's set up, when they have that contract signed by their customer, and then they go buy the code that we've already simulated. And that's the trigger. And that's the conversion point from a, free preview to an actual paid service.

**Interviewer:**

I see. Yeah. Nice business model. How many people are you in the team? You are the CEO, right?

**Interviewee:**

My background is in software mechatronics, electrical mechanical engineering. It's mostly mechatronics and systems engineering. Sure. They started out as a pain point for me. I couldn't develop complex systems fast enough. I couldn't outsource it to a job shop and I couldn't do it fast enough myself.

**Interviewer:**

Yeah. But the approach, I like it quite. Quite interesting.

**Interviewee:**

Yeah. It's, it's very niche, but it's very much needed in the industry. I wrote the first prototypes about a year ago is the last time I actually wrote any code brought on my co-founder Mike and he's one that has converted our first, very early, awful prototypes into actual MVPs. This is the brand new interface that is granted no UX expertise in it. But we're about to launch customers and it's just light years better than what they already have.

**Interviewer:**

Yeah. That's cool.

**Interviewee:**

Yeah. So it's Mike and I, and then he's doing, he's our sole software engineer on the team right now.

**Interviewer:**

Okay. So how many more or less?

**Interviewee:**

There's Ryan, who's working with us, he's an entrepreneur residence with Ann Arbor spark assigned to our team and will be converted to an equity role sometime in the next month. Then we have two contractors working with us also. One is a optimization engineer at U OFM and the other is manufacturing engineer in Chicago.

**Interviewer:**

Okay. So it's five people.

**Interviewee:**

About five. Yeah. Two full time, one close to full time and one part-time and then two contractors.

**Interviewer:**

Okay. That's cool. Do you have a gender balance? Are these all men?

**Interviewee:**

All men, unfortunately. The initial co-founder was a woman and it just didn't, she was still in school and it just was not where she wanted to be spending her time. She wanted be focused on school. We are very, very intentionally trying to have a gender balance.

**Interviewer:**

I'm not judging. I'm just asking. So yeah. It's, that's cool. The average age. What is the like, are you around your twenties?

**Interviewee:**

Yeah, I'm 26. Mike is 30.

**Interviewer:**

You look 22, by the way.

**Interviewee:**

Get that a lot, I get carded constantly. I feel like I appear younger and younger. Probably the more tired I get, just like the more like baby-ish my face gets. So I'm 26. Mike is just turned 30. Ryan is in his forties. Scott who's contracted with Austin is late thirties, early forties. Neil is in his late twenties.

**Interviewer:**

So I would say around the thirties?

**Interviewee:**

Yeah. About thirties is average.

**Interviewer:**

That's pretty cool. You have a very interesting team composition as well. Like

**Interviewee:**

Pretty much it was all based on referral. We haven't posted any job openings recently grad. We've posted a couple and just haven't actually needed to hire yet. Just kind of testing out the waters. Mike was a referral from one of our core advisors over at Arbor spark. Mike has worked with Ted previously very closely Ted highly, highly recommended him both Mike and Ted undersell old Mike. Just absolutely phenomenal, but that's same with Ryan was a referral to us. He's actually placed with our company through Ann Arbor spark. Ryan approached Ann Arbor spark just wanted to get out. So Ryan's background is investment banking in process manufacturing. He worked for about 10 to 13 years in plastics manufacturing for plant Miran Corporate finance in Detroit driving all around the country and around Canada financing and selling plastics manufacturers typically injection molds. I see, wanted to get out of frequent frequent travel, wanted to do something more earlier stage was leaning towards helping first engage with spark saying like, I have this type of experience, pair me with a startup that's ready to IPO or go through some investment banking activity and just kind of laughs like that's not the case. There aren't any companies that are ready to do that. But we have this other company called Startup 2 that has a complete different use case for your background. That's where got started off. So Ann Arbor spark actually pays for Ryan pays his salary for working with us. We're about to drain that fund completely and then switched over to the equity role.

**Interviewer:**

I see. Yeah. That's very interesting. You seem to have a lot of background and experience within the team, like various backgrounds and

**Interviewee:**

It's healthily diverse. Ryan's background has no technical background in what we're doing, but has great understanding of the business. He actually doesn't have much experience with our core customer base. But he has incredible experience with similar customers that he was very easily able to transfer over to our customer base.

**Interviewer:**

When were you founded? What year?

**Interviewee:**

So it started out more or less my last year of school at U of M in engineering. This started as I was working as a freelance machine and supporting myself and then also in mechatronics engineering positions at internships. So I started on this just as a personal project. It caught, our competitors, our current competitors. I tried just giving to them, say, please just commercialize this. I don't wanna maintain it. I don't wanna keep developing it, just run with it so I can use it. No one picked it up, they all said it was impossible. And that's when investors started catching wind of it. One in particular in the bay area and they looked at we gave them a demo. We had no idea they were investors at the time. They just wanted to see a demo of it at an event and followed up with us a couple days later and said, you got something here. We checked on the market, we think there's something here, have some cash we're gonna incentivize you to try to commercialize it and see if it's worth commercializing even. Yeah. And that was what kind of just a slap in the face is like we could commercialize it, I guess. Started doing intense customer discovery through center for entrepreneurship and later the Ary Institute, plus an Ann Arbor spark through their entrepreneurship bootcamp program and that's where kind of really, identified our core customer base, which was different than we actually thought. We thought it was gonna be a mechanical engineering tool for industrial design firms turns out there's this whole outsourcing industry that we had no experience with. I see. So just through customer discovery, regular interaction with the customer going into these shops every day, every other, or every week, every other week, keeping in touch in

**Interviewer:**

What year was this happening?

**Interviewee:**

This was late. We started it late. I'd say like November, 2016 was the first time we actually wrote any software. Then it was early 2017 that we started doing customer discovery by late 2017, we had some working prototypes and in 2018 was let's really, really crack down the business model. Let's really crack down the technology. Then January or February this year is when we actually started work. We're at the point that we could actually start running parts with customers.

**Interviewer:**

That's very cool.

**Interviewee:**

Thanks.

**Interviewer:**

The whole interface and everything. Are you located here in Ann Arbor mainly?

**Interviewee:**

We wanna stay in Ann Arbor, if at all possible if we were to move anywhere, it would be Chicago. It might make sense to open a satellite engineering slash sales office in Toronto at some point. But that's less likely it's decently likely that we would need to move Chicago in some case, but we wanna avoid that if possible. Uwe spend a lot, I spend a lot of time I'm in Chicago every week, if not every or every other week anyways, our contract manufacturing engineers out there or will expand our sales out there. It's very central to our customer base. Plus it's a lot easier to hire Tech knowledge in Chicago than in Arbor. Yeah. Cause it's larger pull.

**Interviewer:**

Thank you for the first part. Yeah, sure. This is, if you have something to add, please

**Interviewee:**

I can just keep talking, but I'll let you direct the conversation.

**Interviewer:**

So here we are again with X. I would like to ask a bit more about software engineering practices. How did you develop your minimal viable product that you mentioned? How did you go through different software steps and what tools did you use? What programming languages were more appropriate and so on, and what decisions did you make?

**Interviewee:**

So start with the question that you asked right before we started recording. Why is this a web application? Yeah, sure. Every one of our competitors in the industry has installable software that they require desktop interface. They sell it on a seat license base and they have annual maintenance charges and it all starts out with a rigorous onsite installation in training service. Well, not E service training operations that can go months if not longer with a shop. So there's a huge overhead adjust to changing software and adopting new camp software in these shops. We decided to go at the web interface because you can run on literally anything you can run on this iPad. You can run your phone around, around when the web browsers that they're running our competitor software on. Yeah. It requires no installation. Even this 3d render you see here, we're using software called X three D under the hood, which is an open source graphical rendering package on JavaScript for 3d models.

**Interviewer:**

Okay. Is it rendering on the client side.

**Interviewee:**

This is rendering client side. Okay. And it's the, the reason we picked this is it runs seamlessly on pretty much almost every web browser. You try it on, including IE six and versions before. Okay. Because our customers are going to be using that software and wanna make sure first time they try it on whatever device it's going to work and it's going to impress them, because the, our customers are so they they've been schooled for decades that there is no better option. It is their shop must run ONT, installable, crappy software. So if we give any indication that we're like that we lose our, a lot of our selling point on that. And we, it makes the a barrier entry

**Interviewer:**

Makes sense. How did you develop all these things? You said we have a team which is core team, three people. Most, some of you are software developers, I guess. And they're doing some CTO roles. I don't know.

**Interviewee:**

Yeah. I'm most I'm. I haven't, like I told you earlier, I haven't written any code for production, anything. Yeah. In about a year now, if not longer.

**Interviewer:**

Okay. But project management wise, like, do you have a,

**Interviewee:**

Yeah. So it's our, fortunately our team is so small. We can run very efficiently without having a lot of overhead. We tried building an overhead early on and realize this is just the long down. Really it's the biggest value point for us was hiring bringing on Mike, my co-founder early on who could has a lot of autonomy and just can move and make decisions on him own on his own for product and for technology stack on the, on the go plus regular communication. So Mike is actually out of town for a month working remotely and we have phone calls a couple times a day. We're in touch on slack. We have video calls, screen sharing constantly, and just having that open line of communication all across our team. Not with not necessarily set meetings, although we do have some, but mostly just if something comes up, we can immediately call one or another.

**Interviewee:**

Yeah. And talk right away. Having a lean enough team enables that without having some overarching task management software or anything on the technical development side. Yeah. We, the first prototypes, we had no interface, essentially, when we first started doing this, it was all focused on how do we take a 3d file and break this down into, well, we need a facing operation first. We need some pocketing operations. There are, we need a pocket there and there, and then we need to profile that corner out. I see. How do we automatically identify that thing pro that set up and that processing from in software where it's always been a human, trying to look at this. Okay.

**Interviewer:**

And so what was the initial programming language there?

**Interviewer:**

You say we didn't have an interface. Did you have some programming

**Interviewee:**

Language? You, yes. What did we initially do that in? It was Python. So the first thing we did was the inspiration of the company when the first prototype back in November, October, whenever it was in 2016 was a Python programming framework for controlling machine tools. Sure. And it was mostly just if I make this part, but I realized it doesn't fit and I move this bolt hole down three millimeters,

**Interviewee:**

It's gonna take another 20 hours to process this part. And you have to remake it the whole time. Yeah. The initial prototype we developed was just changing the machine code. So when you remake the part it'll move that pocket down slightly. Okay. we realized then that there wasn't really a market for that specific product. But there wasn't market it for this automated solution getting up to that point and just simplifying that 20 hour process. So you can't still make that design revision upstream and just prop get it down. And it was through customer discovery that we've learned that not even showing them our prototypes, just talking, just asking them very, very specific never loaded, but it's say loaded questions of trying to gauge whether or not this actually provide business value,

**Interviewer:**

But then you moved on to JavaScript and web technology. Yeah. So did you convert a code? What did you do?

**Interviewee:**

A lot of our backend code is working in Python because the some of the libraries that we're using are there and it's just the quickest to get started. The backend kernel is on C plus plus if you notice our tool path, generation software, we originally wrote our own tool path gken package that will generate these vectors on our own realize that's too difficult. We out. And then we license in some technology from Germany that we later realized all of our competitors already use anyways. And just partnered with them to generate all these tool PA softwares they're. We, we run an instance of offer locally their libraries and wrote a bridge and C plus Python just to call their libraries to compute this geometry.

**Interviewee:**

And then that, that bridge returns factors back that we load into the application. We're stalling. It we're stilling each of the steps in a database. So if you change the spin, the speed here this's broken in this demo, but pretend it works. You change the Spino speed here and switch over to the NC code tab. It'll prompt you say NC code out of date, click here to regenerate. You can make as many changes you want until we regenerate. And then it'll, it'll give you a change log of each of the changes back there. Sure. And the way they're pulling that off is each of the different operation here. Each of the different code, all the parameters set for this are all stored as singular objects in a larger database and NC code and the the application draws from that and organizes all that based on the data that's working the database. And anytime they make a change that database, it just propagates and update for the interface and update for the NC code regeneration. Yeah,

**Interviewer:**

Sure. That's pretty awesome. When you said MVP like this MVP is sort of joining with lean methodology. Like you wanna jump into a minefield with different options. Was this for you the same or,

**Interviewee:**

Well it's for, for how do it decide what the MVP is and what it needs to be, or

**Interviewer:**

Yeah, if you use the lean concepts for building the different things,

**Interviewee:**

Honestly, we're not.

**Interviewer:**

Really, you are not into that.

**Interviewee:**

No, t's not that we're not into it. It's just, we never consciously said, we're gonna use agile. We're gonna use lean. We're going to use scrum or anything. Okay. It's the focus is really on here's the business problem we need to solve right now. Here's the, the technology aspect of it is very intellectually stimulating. So at least we're not getting a bored, but how are we gonna refocus the technology to hit these business problems tomorrow or next week? And setting we've been setting like one week granularity. We have one week technical planning once a week. Mike and I, to set objectives of in order to move the business forward in boots, shopping phase for the next week, what do we need to accomplish in this week timeframe? And we, we talk about a lot. We set a lot on the back burner and then just say, this is what we need to focus on for the next 6, 7 days.

**Interviewer:**

Okay. Yeah. You didn't have to adopt a strategy. You, you kind of sort it out yourself. Yeah.

**Interviewee:**

And a lot. And a lot of that is we didn't need to because our team was so well built and we have such good communication, even though we were oftentimes working remotely.

**Interviewer:**

I see.

**Interviewee:**

There's just a lot of autonomy in there.

**Interviewer:**

That's very good. Did you do testing of the software sort of like if things were running properly, if what your business was a valid product or not? Yeah. Give me one for a second. Sure.

**Interviewee:**

Yes, we have.

**Interviewee:**

I just wanna show you. It was like, so this is the actual interface that our customers are use it. Awful.

**Interviewee:**

It's a cool. So if you notice, try posing those buttons.

**Speaker 3:**

Okay. Yeah.

**Interviewee:**

Now imagine that they beep at about 50 decibels. Okay. It's like, beep beep beep every time you press it, it's actually like that slapped up about face our like neck height on a machine tool. And you're pressing this button at that angle all day. Oh, you can't, you can do whatever you want else. It is usability us user experience wise. This is an absolutely awful machine tool. Yeah. It's one of the worst ones I think. But it's it more or less standard in the industry?

**Interviewer:**

Is this on your website as well? Like with you on the picture?

**Interviewee:**

Not oh, yes. So it's actually, you've seen that. Yeah, I did. It's not this it's actually, it's not a simulator. It's actually, one of the machine was one of our customers. Mm. Yeah. But it is, it looks identical. It's so same operator interface. Yeah.

**Interviewer:**

Very similar.

**Interviewee:**

Yeah. So we got this so we can test the actual code that we generate and just see if it errors out a lot of the problems we were focus and we were hitting up earlier on, is we just weren't getting the syntax, right. Because there's no pub, there's no good published syntax or language, documentation for the language that actually runs on this machine. I see. So we took, got this to see runner program. It's not gonna of cost collision cuz it's just simulating. See if it S out at all, see if there's anything that's illegal running in it. Some of the areas we caught up with early on running on actual customer sites was if you start a program mid cycle instead of at the top, then there's some codes that are running at the top that are triggered when you run it in the middle and then it causes collision stuff. So just trying to like seeing where breaks just on a very simplified interface that is not fragile.

**Interviewee:**

See. The other thing, the, the more interesting thing we did was just visually going through and seeing once we get this render of operations, does it actually make sense what's being and then also going to a customer and saying like with this work calling setup, could you actually pull this off? Could you touch off the machine tool at this point? So for this one, no, you couldn't really there send something physically there, but if we're generating code that requires a 3d probe to touch the origin off, then you could, then we could generate code that touches off that face, the back face and device.

**Interviewer:**

Yeah. But this is a part of a demo and I pretty believe you have reached a point that you have done quite good development and you can demo. Yeah. But until you get to that point, I guess you generate code afterwards that you say we are charging clients. Yeah. So until you get there, you need to test a bit on the software, right. On the, what this is being done, if it is correct.

**Interviewee:**

We had some earlier interfaces that

**Interviewer:**

You do, like integration, testing unit testing of the software,

**Interviewee:**

It wasn't unit testing. We tried to do unit testing and realize with our unique, what we're trying to do is so graphically intensive. There just, isn't a great way of unit testing yet. And it just requires a lot of overhead. Sure. I made a lot more sense to just move fast without unit testing and testing more Blanking the other term, not procedural testing system testing

**Interviewer:**

System testing, integrated testing,

**Interviewee:**

Or yeah, there oh there's a very important term for this.

**Interviewer:**

You mean for the interface testing or

**Interviewee:**

No, for an actual system, behavioral testing, behavioral testing.

**Interviewee:**

Yeah. So that's more of what, that's what you're testing. Okay. Not really algorithm, not really procedurally, but just making sure yeah. That the output actually matches what we're expecting. Okay.

**Interviewer:**

Is Wells the expectations? Yeah.

**Interviewee:**

Yeah. But a lot of it was, we were prototyping. We were less focused on actual testing and more focused on just solving. More generally like a certain element to this early on a certain elements we couldn't machine. We kind of ignore it and say like this package,

**Interviewer:**

But what is the Oracle telling that Oracle mean in this case that who's telling you that this thing is right or wrong,

**Interviewee:**

You have it's MI and then Mike also. So it's so for instance well it's a good example. We have previous one has a pocket cut.

**Interviewer:**

Okay. Because you have built something, you generate some code, then you have to figure out, okay, is this right? Is this wrong?

**Interviewee:**

So one of the things that we are doing is for testing from the product side and less, less the code side, but it's still important. Sure. This tool path has a number of step downs. And the reason for that is if you're rotating a cover, you can only gauge certain amount of material before you have too much load in the tool. And either the machine stalls or of the tool breaks. Of course, one of the first things we're doing is that was obvious to Mike, but Mike didn't come for a machining experience. So he was based on the step down based on number of step downs in a cut, not based on a calculation of the total maximum stepped on for that tool. Yeah. And that's one of the things that looking at just, just by observing what the tool pass looked like on the rendered, on the model we able to identify that's just wrong.

**Interviewee:**

And then dive back into actually where that error was. And we realized, oh, it's based on this false understanding of how this strategy act and the machine strategy actually works. Okay. Learn, we both learned about that and then fix the code based on that. But a lot of it for us is just graphically seeing what's going on in the machine and we can catch a lot of bugs and a lot of falsehoods of our strategy just by graphically seeing this is playing out same with customers. We can go to a customer and show them this show them an early render of this. Cuz we had renders going back in like last, last September. It was when we first started doing renders cuz okay. We realized that was what we needed to test with.

Interviewer:

Okay. To compare the results as well. Yeah. If I mentioned this is pretty exotic, but if I mention metamorphic testing, have you ever heard about

**Interviewee:**

That? I have. I couldn't honestly tell you really couldn't remember what it is. Yeah.

**Interviewer:**

You are kind of the first one that says yes. Out

**Interviewee:**

I definitely have heard it.

**Interviewer:**

Yeah. It's you know, this Tesla autonomous cars. Yep.

**Interviewer:**

Sometimes it's hard for the test case to have a good Oracle like to know what is the outcome or the protection of the result. So this autonomous cars, Tesla sometimes crash against the big track, which has a wide facet or a blue facet. Yep. And the reason is simple. The, the sensors and the AI doesn't distinguish that this isn't, the sky is just the face off a big truck and it just moves forward and crashes with a big truck in this case. Yep. So basically what meta metamorphic testing tries to do when you don't have a Oracle at hand, like you mentioned before is okay. It sort of tries to make sense of a salt, certain set of inputs and a certain set of output that is derived from, and they put a threshold and they say, okay, if the values are on this right side of the threshold, then most probably everything it's okay. But if the values are on the left side, then most, probably something has been wrong. We don't know for sure the exact value, but we know that here is the threshold. We can tell you that if you are getting this kind of strange values, then most, probably something is wrong. So this implies metamorphic testing is that it's not fixed testing, like unit testing. We have the value, the final value we can compare nine and 10 and you say they are not equal, but you have sort of subsets that you can compare

**Interviewee:**

Between and some continuum. Yeah.

**Interviewer:**

Yeah. And there is this very good workshop. I'm not an expert in the field, but there is very good workshop. I think in this IC conference, it was in Canada two weeks ago. It's one of the top level conferences in software engineering that we have. And basically this workshop has been running for four years, which is pretty new. The metamorphic concept is also pretty new with respect to the previous ones, but it's, it's finding out, you can test for things that other test cases don't, and you can find issues that other test cases cannot find those issues.

**Interviewee:**

So in the case of Tesla with our autonomous and, and the white truck phase, how does that, can you get some indication of what values they might have been testing for that, that would've been alright.

**Interviewer:**

I can't say for sure, the exact values there are for sure. Yeah. I confess this article. It's very good article. I think they, I met with the two workshop leaders and they were sort of organizing this workshop and they, they had pretty good knowledge about this. And they were saying, okay, with Tesla cars, it goes like this. And we have a, they had a pretty good article on describing what values they were tech testing for. Why shouldn't the car crash in that scenario? It doesn't tell to the car that this isn't the sky, but it tells to the car that look, maybe something is

Interviewee:

Wrong, not be the sky. It may not be the sky. Yeah. So this is a good, beneficial part of metamorphic testing. It looks for patterns.

**Interviewee:**

Yep.

**Interviewer:**

Like the sky is blue. Okay.

**Interviewer:**

It's pattern recognized, but we don't do this very often. You mentioned behavioral testing, which is somehow similar to that, but I know

**Interviewee:**

Behavioral testing is still it's unit testing at a larger scale. Really? Yeah. It's so it doesn't when you have all these small unit, when you have all these other actions going, is the output expected. Yeah.

**Interviewer:**

So the metamorphic testing, it's sort of like the person, like if you tell the person, okay, you have a big blue thing in front of you, is it the sky or is it something else? And what does the person do perhaps moves backwards and realize, okay, maybe this is not the sky. It's something else like a child that learns a bit on what's going on. So metamorphic testing is trying to solve testing based on patterns. So I can send you the workshop playing and the you, I definitely, you should have a look because yeah,

**Interviewee:**

I appreciate.

**Interviewer:**

It's very interesting. And maybe you are the first person that says yes. Out of

**Interviewee:**

Think it's my weeks two 80 class. So I'm a recent CS graduate of U of M. I switch into CS right before I graduated. I was electrical and mechanical work. Yeah. So my intro CS classes, which are very general and broad, I see I got are so very recent. So I think that might have been why that I heard that term before.

**Interviewer:**

Yeah. Yeah. It's it's still recent, but it's not that old meaning that you, you might have heard about it. I think

**Interviewee:**

It's probably on like 2016 that I of it.

**Interviewer:**

Yeah, it is. So basically what's going on, it's perhaps many systems that you cannot test for, you can use this kind of approach. Yes. Pattern, pattern recognition for what's going on for the Oracle.

**Interviewee:**

Yep.

**Interviewer:**

Okay. If I mention technical debt. Yes. I, I think you are aware of it. And what can you say? Do you have technical debt? Don't you, you don't have technical debt. I dunno. What's the situation. Yeah. It's how do you cope with it?

**Interviewee:**

It's oddly enough question of like, we don't really know that we have technical debt until suddenly you're like, well, crap, this doesn't work. We have to rewrite this but it's, we are, we very intentionally made it our technical stack language agnostic to give us the ability. And, and part of the reason of that was we don't wanna be locked up in just very specific on one program, one language or one set of libraries, of course, to the point that we're running a lot to make that work. And then it's, it's sudden not gonna work. We have to find another solution. Sure. We wanna make it a lot more modular, modular with some things that are running that we're doing locally, just cuz it's faster that, and just communicating with the cloud, going back and forth, we built up infrastructure for that flexibility.

**Interviewee:**

For one of the best cases of, I guess, I guess the, one of the strongest cases of actual technology debt that we had was building out our tool path library for generating these vectors and these okay. Tool paths or on, and we were pretty convinced that we could just do this. And we were told over again over again by manufacturing there's no, you can't, that's really hard. That's what all of your competitors are focused on. Like that's not that hard. Okay. And we were a, when we finally realized, oh crap, this is really hard. We can't do this. Or I see it's gonna take two years and we're not still not gonna beat our competitors to get the point we need. Because of the flexibility of error infrastructure we built in earlier on are able to find we connected with our manufacturing engineer, contractors.

**Interviewee:**

They connected us with the one company in Germany that writes all the tool about Jen for almost the entire industry. Now I see we flew, they typically do integration in about nine months times with our competitors for early MVP, prototypes stage, nothing in production. We flew out to Germany, got to speed with them and got something in production, running with customers in about about 10 days. Okay. so the flexibility built into our infrastructure and the understanding of how to, if we need to remove a system or re-engineer quickly helped us really avoid technical debt. Cuz there wasn't a lot. Our tool at Jen was so discreet and running as a, such a discreet system from everything else. I see. It was very easy to just pull it out and put something back right back in one of the first things that I did even before I brought Mike on that got us, our first bouts of investment was we didn't have functioning prototypes of this whole inter of not the interface for sure.

**Interviewee:**

The interface is only couple weeks old, the backend processing, nothing actually functioned full-blown with it. We had a couple prototypes for hot points of discussion that were complicated. I see. But the overarching infrastructure was all bought diagramed out. We have, I still have diagrams of what the infrastructure looks like. Okay. And the very specific notes on that, the specific module is that we're gonna build how we're gonna build them, what time Oxi and whatnot and all of the plumbing in between those modules modules and how we would replace them if need be. And that's what kind of, when we, our technical audits, everyone kind of checked off. Yeah. That will scale. And that's, if something goes wrong, you have a clear and easy way of handling that. Okay. And that was it's.

**Interviewer:**

So you are sort of managing, trying to make is the

**Interviewee:**

Technical debt. Yes.

**Interviewer:**

You knew that you had the technical debt.

**Interviewee:**

It's I knew that we would eventually that we would accrue technical debt and I wanted like, we're just going to accrue some technical debt over, over again. How are we going to mitigate that and keep it low enough that it's just not gonna run away with us. Okay.

**Interviewer:**

So it's not like you decided we don't want the technical debt at all.

**Interviewee:**

Or no, it's no, we need to move fast. We need to try things and we need to break things by creating dummy MVPs that help us validate requirements.

**Interviewer:**

So you needed to make a trade off.

**Interviewee:**

Yes.

**Interviewer:**

That's the bottom line.

**Interviewee:**

And we compromise some efficiency with actual processing and execution in order to and instead prioritize some modularity with the software right now, we, for still to this day, we haven't needed to fully automate the process. We have clear steps for in the future of how we're gonna do that. Okay. But it's just too much of a technical hurdle to actually implement it right now. Instead we have separate modules for when you upload apart we, we have a module, we have to elaborate that we'll take that part input throw a bounding box around it. And then another one that will identify what type of stock material size we should use for that. Okay. The output of that is passed to another module that the IPO graphically interface on a local computer or a web browser for foreign an operator first fluidity specialist to go in and say, this is the bottom. And we're just graphically clicking on that. Cuz we could algorithmically do that. We could build that all together, but it's much easier to have a separate module for that. And just ask that and then we can automate that out later and it just plug in the automated version and unplug the un-automated version. Yeah.

**Interviewer:**

That's smart. Do you do some sort of documentation or do you do a lot of documentations of the coding or of the system building

**Interviewee:**

Like as a decent amount of documentation just as making, making the code very readable. That's the biggest thing. Okay. Our team is small enough and it's the overall infrastructure, it's easy to explain to someone that needs to we're asking questions or anything. Yeah. But because of the size of the team and just the needs, we haven't focused on documentation, especially because so much of what we're trying is changing so frequently. It just doesn't make sense. Yeah. Later, later on we wanna do a lot more commenting, a lot more documentation with the programming. And then also just

**Interviewer:**

Would you, I stop some person to do this in particular or you

**Interviewee:**

No, the engineers that are doing it engineers. When, when we get PA, when we get like more into, into more rugged pilots, when we explain the team, it's gonna be the responsibility, the engineers to document and then write code right now because it's just try this, doesn't work, try this. It doesn't work. Try this. Oh, it works. There's no be a reason to write the documentation there. Okay. But when it, we have a more set pathway for specific feature then it makes a lot more sense to write the documentation including the tests and how the behavior of how it should work. Some template of what they think the infrastructure for that specific module should look like, okay. They build module and adjust the documentation based on what that is.

**Interviewer:**

Yeah. I

**Interviewee:**

See that that sets in stone, what the, how long it really should take to do that gives you a clear mindset of it actually be in there and it's something it was if the actual implementation doesn't match the documentation, that engineer that wrote that same documentation will be able to recognize that and say, what's going on with this? Is it just hardly thought? Or is it, am I going a destroy?

**Interviewer:**

One thing we don't teach students at university is how to protect their artifacts because they build a lot of software and a lot of systems, they try at least yes, but I, I think they are smart people. So they may be, sometimes they're doing a very good job and their project might have some value. And I bet you have a lot of artifacts and you have a strategy what to give out and what not to give out to the end client. So I'd really love to hear about it.

**Interviewee:**

Yeah. So the only thing, so we give this interface out to the client, we give the NC code out to the client. We give a list of the tool, table and stuff. The actual code that we provide to the customer though is exclusively the numerical code, the G code that will run in their embedded machine. I see. And there's, there's some really nice things going with this industry on. So taking a step back, if you're Boeing and you're contracting with a job shop, that job shop will oftentimes not even know they're working with Boeing, it'll be through like four sequential, third parties that are soliciting that job. And by the time that job shop gets it, even if they do know it's for Boeing, they don't know if it's for a Gued plate on a desk chair or it's going on a new 7 47 or it's going on a space launch system or anything.

**Interviewee:**

Same with all these industries. Oftentimes the customer will know what industry it might be going to. They will typically know what customer's going to. They will almost to never know what the actual component is. Okay. So there's some very natural abstraction on our behalf which is even better when we're actually generating the code. When we're talking to engineers and attorneys and insurance brokers and everything, they're like you're giving code to companies like how are you making sure that when you're given that it's not being stolen and reverse engineers, like, have you seen this code? It's impossible to tell what's going on with it. There's reason operators can't edit it. It's all it's compiled binary area. It doesn't communicate the actual design, the component.

**Interviewer:**

Okay. So you don't give the source code, you give the binary code at the end.

**Interviewee:**

It's the, I can, I can show you what it looks like.

**Interviewee:**

This is an older version, a production. This is what the code actually looks like. So it's literally some header file header information. It's an awful language header information on the contract that we can throw in and setting and configuring the the machine. And there's all just movement instruction on the machine to move to this location,

**Interviewer:**

Colocation coordinates right correctly.

**Interviewee:**

Yep. And you can reverse engineer some of the movement instructions into surface geometry, but there's no tolerances. There's no P build out of features.

**Interviewer:**

But this is not the source code. Sorry to interrupt you. This is sort of the output that you get from the code.

**Interviewee:**

Yes. So that's what, that's how we're producing. We're not giving the customer any of our source code. So like master camera gives camp. Some more competitors will actually provide the customer with that installable software. We're just giving them output. So in addition to making better user interface, it's so much easier protector technology because there's no installable software for them to actually

**Interviewer:**

Want, but you know that the JavaScript it's downloadable from the browsers,

**Interviewee:**

All that's in here is a 3d round, a 3d render of the, of the part it's, it's still none of our, it's none of the software that we're protecting. Okay. That

**Interviewer:**

We need to protect. So then the software that you are protecting it's on the backend. Yes. Rounding Python.

**Interviewer:**

Yeah. Python and C

Interviewee:

But six different language now. Yeah.

**Interviewer:**

That's cool. Yeah. That's very interesting. Yeah. So you're trying to, to protect what is of value, which is the code that is generating all these nice outputs at the end of the day. Yeah. And that's very good. We should teach students. I think when it comes to protection, there are some security, I would say, issues like this server, where is it located? Yep. Is it cloud based or whatever choices you have made? Do, can you mention some of the choices that make you feel assured that third party entities cannot?

**Interviewee:**

Yeah. The biggest honestly. So this is a little roundabout answer. Yeah. One of the biggest.

**Interviewee:**

Put them, one of the biggest security benefit fits to us early on was we didn't do, we did no marketing. We were, we, we made sure no one that would target us, knew we even existed. Okay. we did we even went to the extent of when we were testing things on AWS, we would set a timeout on about 30 minutes and then we would just shut down the, at those instances and spin them out on a different server in a different account. I see. And it just like, it gave us the flexibility to kind of play around and set up and run some task without worrying too much about over limiting security because no one was going to be targeting us. It just wasn't part of the threat assessment threat funnel. And then that gave that guy, us the ability to switch or to move our infrastructure into the co quickly, test it without being, without being a target try to break into it, see, make sure it's locked down, get some third party audits on it and then shut down and make some changes and re else

**Interviewer:**

Who told you this? Who told you to do this?

**Interviewee:**

It's my teammate and I, it was Y and I just talking it over and thinking of looking through a threat model. So I was I was working for a couple

**Interviewer:**

When I thought maybe a book, maybe not somehow, not a person. Did you come up by yourself?

**Interviewee:**

It was just, it, it was just seemed obvious to us. It was, it was just, we thought talked through. And so for instance, I was working at a company a couple years ago that was running stuff on AWS. Sure. And overnight their account was compromised and racked up about $5 million in process system fees. I see probably be just for Bitcoin mining. And we had a conversation about company the, the next day and said, yeah, we weren't heavily protected. We didn't have to factor allocation or anything running. But the big thing is up until this happened, we were never a target. No one knew that we were large enough company that had this large of a credit line in that they could actually get away with reminding this much, probably Bitcoin,

**Interviewer:**

At some point you'll have the problem. I'm pretty much sure. And yeah, I would love to teach people how to the value, first of all, because the value, you understand it at some point, but how to protect yourself as well.

**Interviewee:**

But the big thing for us was learning that we didn't need that level of protection because if I didn't have that experience, I didn't have that business frame of mind going into that. I would've said we're not running anything in cloud until we spent $50,000 on a security order to make sure it's rugged. And it's like, we didn't need to.

**Interviewer:**

How did you figure out that this piece of artifact here, which is the 3d rendering, rendering is not worth protecting, but rather hold it's worth protecting. How did you figure this out? Because we need to figure out what, what is worth protecting as well? Not just maybe we can protect everything, but maybe not because we wanna give out something.

**Interviewee:**

Yeah. So the, the key points of value in our product are the for the operator, the, the list of operations for how they would actually machine that part. Okay. so there's, so there's our source code and then there's the actual product that we're providing. So we wanna protect our source code for the processing. We also wanna protect the IP for that specific job so we can make a sale on it. We were looking at saying, we're providing tremendous value to the customer, just providing them step by step instruction, how to make this,

**Interviewee:**

But it's gonna be really hard to all that. If we can't get them on a per job basis, if we can't give them a preview of what that actually looks like and get buying from them. But the chance is that after they look at through this, that they would just purchase our code instead of spending another 20 hours doing this anyways. And just, just saying, well, this is great, but we're not gonna pay. We're just gonna go back and do the same way and spend that 20 hours instead of paying like 50, a hundred, $150, whatever for us. Sure. The chances of that happening are so low. And the chances of them just buying software from us are so high. We just we'll just charge them for the value they're getting for the process bonding when they buy the G code. Okay. And it's to say, we're comfortable giving them away the rest of it because there's, there's such a high likelihood that they're gonna make that transition period. And that's how we decided to we're comfortable releasing the IP on a per job basis. The source code for us. However so if looking at the renders and stuff, the, the renders on here are just 3d files. So there's no, there's no inherent technical property that we're giving away with that. Okay. It's all business property in this case that we're in business, I, that we're giving away.

**Interviewer:**

Okay. And these IPS, if we talk about since you mentioned and brought it up, I was gonna ask, but this intellectual property rights, IPRs, whatever. Do you have any thoughts about those? Like you have code, you don't want to be disclosed, but if somebody does, or maybe you give something outsource open source because you wanna contribute to the community or you get some call from the stack overflow community, and then you have to bring it back, perhaps I don't know how you cope with this.

**Interviewee:**

The biggest thing I'd say is we look at it from a business standpoint and say, most of what we're developing, just like we're small enough. We're underwriter are we have a detail, we have a threat assessment that says we don't need heavy patents. We're not gonna get sued out of existence cuz we're not a business wise. It doesn't make sense front us to Sue us and try to shut us down if we're infringing anything. So it doesn't make sense for us to build up patent repository for counter, for mutually short destruction because it's not gonna happen anyways. I see. But in the meantime, let's hold things as trade secret because there's no business reason that we would need to release it. And therefore it makes more sense to hold onto it in case we do wanna patent it or just to keep it as trade secret. Definitely. Okay. So it's, we've been very, we've been very heavy on trade. We've been keeping patent as patentability assessments on our radar as it's we, every couple weeks we say we kind of decide, are we at the point that we need to really start focusing on this? Or can we push it off? And it's always, you know, we can push it off up to this point and we'll see all of that happens in the next two months or so when we have of the conversation again. Yeah.

**Interviewer:**

That makes sense.

**Interviewee:**

It's I would say more and more in re in retrospect, the technology decisions we made have been almost 100% business focused of the technology is here to support this overall business. Like nothing that we're really, honestly, nothing that we're developing in. This is so novel from a technical standpoint. Okay. It's a lot of plumbing. It's a lot of here are the tools out here for managing complex compound geometry. Let's apply it in this new use case, but it really stands from this unique business model that no else in the industry has. Yeah.

**Interviewer:**

So you told me a little bit what you don't want to disclose to yes. To the end customers. And I really hope I could help future students to understand what are the things they are building. And it's not easy. I understand that because you must have a model, you must have some thoughts about it. What is of value and whatnot, but they should start thinking about that. Yes. Early phase. And I think it'll help them later on build a startup.

**Interviewee:**

Yeah. And, and again, the, the best thing for me that put that in context of how we should be focusing on protecting this is look at the business value and saying, this one can give what way this is. What's not worth getting away. Yeah. How should we protect this and how aggressive and how not aggressive should we be?

**Interviewer:**

Yeah, exactly. If I talk again a bit, a little bit, maybe the last things we discussed about the team. Yeah. You are the CEO and you are the co-founder and you have structured a team together, which pretty to have very nice team dynamics and so on. Yep. But when it comes to making decisions, are you guys, or are you in particular goal driven or resource driven?

**Interviewee:**

Can you elaborate on that?

**Interviewer:**

Yeah. I'll elaborate context.

**Interviewee:**

Yeah.

**Interviewer:**

So you started with fluidity. So you started this company and of course you had limited resources, I guess at the beginning and perhaps some opportunities came along and you had a milestone, a goal that you wanted to reach and then did you grasp lot of opportunities because maybe they were not really your milestone or your goal, but still look pretty good. And maybe you changed the goal a little bit and go for it or did you stick to your goal and is said, okay. I wait for more resources before I get

**Interviewee:**

There. So, so we're definitely guilty of future creep. Okay. But the we've definitely chased.

**Interviewer:**

Can you elaborate on the future creep part?

**Interviewee:**

Yes. So we've definitely been opportunity focused sometime and more, more so than not, it's been healthy opportunity focused and quickly testing us. Like there's this new feature that this customer really needs, or this there's this new opportunity where we could drastically increase our revenue stream with a new revenue model. Let's see, let's see how we can do that. And bill time technology. So the biggest thing is of feature creep were realized actually two cases two problems that we are facing from the business side was customer bringing up a new customer online with fluidity because they don't already have a, a digital render digital list of all the tools that they have in their shop. Yeah. They have back the napkin or they have in their head or they have a crappy Excel spreadsheet that isn't actually use useful.

**Interviewee:**

So one of the things you said is like, well, we need a way to show investors that can get past this hurdle. So let's make a barcode scanning app that they can go into, that they can download an iOS app or a web application. They go into their shop and fire for the camera and they scan all their tools and load it up into fluid and it'll be ready to use. And we did it and realized, well, that was a waste of a week and a half. We could have just told investors, we could have built this. Okay. And they said, yeah, that seems doable. And otherwise just gone to the shop and in an hour and a half. Yeah. Manually typed in all of the things or used our own internal tool and not built one for customer facing. Yeah. I see. Fortunately we've been very diligent on saying like, if something, if anything we're is gonna take more than two weeks to move us forward, we're just not gonna do it even when we so when we first started building up this earlier version of this interface and earlier version of the processing, we realized this new types of geometry we needed to handle and the new if a customer's gonna update the spin O speed or the feed rate or some other parameter, we had no way of actually backtracking, changing that parameter and then cycling me back through.

**Interviewee:**

And that's, that took about it's a good thing. Mike estimated that take about two weeks to make that change and up taking about five or four days to rewrite the back end. So it was updating SQL database, or I think it's SQL instead of just building it all in one, go in one instance. Sure. so we, and we probably wouldn't have made that change right there. We would've gotten the product a bit further and just built off that and realizing we need to make that change at some point let's avoid technical debt as much as we can building up more technical. But because we could do it within two weeks and it was more, it was pretty critical. We just had to pull a track on that. If it was longer than that we wouldn't have.

**Interviewer:**

Okay.

**Interviewee:**

That was a little sidetrack from the discussion, but

**Interviewer:**

Yeah, no, no, but you make your point example wise. So I think it's pretty good. And I mean, what I have noticed is that a lot of people say that we stick to the goals, but sometimes we have to, to fight with the resources that you have.

**Interviewee:**

Yeah. I'd see the, the biggest thing for us. And I'm definitely no expert at this. I just got a call with Ryan, our team about two hours ago or so when we were talking and saying like, yeah, we need to stop trying to focus on this one thing and just get more customers online, stop bringing a customer acquisition, just do everything we can.

**Interviewer:**

How many customers do you have right now?

**Interviewee:**

About three right now, three through piloting. And we need to just grow that and get them more regular use. And we need to not focus as much right now on lowering our cost of customer acquisition and our on-boarding costs. And just say, we'll figure that out later, let's just right now, get more testimonials and more regular use and get further and faster with that realized, we talked about about two months ago during a three hour ride back from Toronto, and we haven't really substantially made progress from that. And so it's, I'm, I'm definitely guilty of not doing this well all the time, but it's all of our technical incision to really focus on how are we going to risk to the company as much as possible in the next week or two weeks, or if we need to look a little further on that's where if our goals change and we come up with, we have new customer insight, we'll do a little bit more research to validate that it is a problem or an opportunity, and then we'll adjust the goals based on that. Okay. and pretty deep granularity. Really.

**Interviewer:**

That's very good answer. I'd say thank you. Do you have any final remarks? Because I have done, I have asked a lot and I'd like to thank you a lot for all the insights. Yeah.

**Interviewee:**

Not a problem.

**Interviewer:**

The beautiful things you have built, I'd say,

**Interviewee:**

And this ugly piece of machine

**Interviewer:**

You, you shouldn't say it's ugly. I find it all. I find it also a bit cumbersome because it's

**Interviewee:**

It looks very cool. Yeah. But started using it is so awful. Yeah. final remarks. We've been back and forth way before I heard Mike before burn anyone for our current team on the team. I, I was always sure from the beginning that I needed a core team, I could not do this alone. I was definitely right with that. And I wanted, and I also realized that I was not a software engineer. Not only I graduated software engineering degree. Oh, I see. I'm not a software engineer. I see. But I'm a great product manager and I wanna make sure we got software a very, we got excellent software engineers on our team early on. I initially started just going through my network of recent graduates from U OFM and current students trying to source. Yeah.

**Interviewer:**

What, what were you looking for when you found these people or so I found new people

**Interviewee:**

Eventually I was looking for, I was really advertising. This is a, I think a very promising business opportunity. I've also been told this way promising. Okay. And it's very intellectually stimulating software work. It is. It's hard problems. No. But they're all like small, hard problems that no one's ever thought about for yeah. Of course. Novel uses of existing technology and and applying them in, in a new novel way in, in a niche industry. Yeah. That been

**Interviewer:**

Make an easy example. If you get an intern in the future, like next year, what, what would you have him do? Like in soft skills, hard skills.

**Interviewee:**

Like, so that's one of the problems that we were facing. I was, I was looking, it was like,

**Interviewer:**

Get more women, first of all. Oh

**Interviewee:**

Yeah. Well, it's talking on the intern side. Yeah. So early on Mike and I are both very committed. We want to hire interns because it, the experience is so valuable. We want to get back to the community. Sure. But from an early startup standpoint, especially something very heavily technical based and very neat and very niche application, such a bad idea. Yes. and it's very unfortunate, but it really is the reason we got so far so quickly. And after, after about a year, struggling, trying to find the right technical expert to be in our team. I realized how important it was to hire Mike and or someone else in Mike's position that had many years of senior software experience. Okay. Had great leadership experience and great business mindset. And really can't do that with a recent graduate. There's some, all, some rock stars out there, but 99 times outta a hundred, you just won't find that you need someone that's been in the industry that has experienced this and has made mistakes and figure out how to go past that.

**Interviewee:**

Yeah. And really understands the, is a side of that. Mike has been phenomenal he's I mentioned like we wanna move the comedy forward in one to two week steps Mike's ability to estimate how long a job will actually take has always been exactly on par or better. It's always taken, been faster or just as long as it's set would take. And if I want the case, we wouldn't be here today. Sure. so hiring and the other thing is senior software engineers are much more, are not nearly as cash needy and are more interest in working on problems that they find stimulating and interesting than us getting a large salary really quickly past student loans and just have fun with. Yeah. And that realizing that retrospect made a lot of sense for us as a startup, just hiring Mike or some or other senior engineers, at least from the next year we plan on hiring mostly senior engineers. Yeah. Then moving into interns and recent grads okay. As needed.

**Interviewer:**

So you are looking for people with experience since you started. And then I see that the team had quite a good experience,

**Interviewee:**

I would say. Yeah. We, I really liked out with Mike and it's really set the groundwork for how we're gonna hire in the future. Yeah. the cultural wise also, but just I forget where I was gonna go with that. Yeah, no worries. I had some, a tangent.

**Interviewer:**

You said that you said so many things and I'm very happy with the whole conversation. I thank you so much, X. You're welcome.