EPISODE 1513

[INTRODUCTION]

[00:00:00] ANNOUNCER: This episode is hosted by Sean Falconer. Sean's been an academic founder and Googler. He has published works covering a wide range of topics from information visualization to quantum computing. Currently, Sean is Head of Developer Relations and Product Marketing at Skyflow, and host of the podcast Partiality Redacted, a podcast about privacy and security engineering.

Today, we spoke to Toma Puljak and Vedran Jukic of Codeanywhere. We discuss cloud-based Dev environments, cloud-based IDEs, infrastructure as code, Dev containers, and live collaboration.

[INTERVIEW]

[00:00:40] SF: Toma, Vedran, welcome to the show.

[00:00:43] TP: Thank you.

[00:00:43] SF: Awesome. So, this is my first time doing actually an in-person podcast recording, and also my first time with multiple guests, and my first time doing a podcast in Croatia, and your first time on the show. So, lots of first today.

Let's just start off with some basics. Can you introduce yourself? Give a little bit of background for how you ended up where you are today?

[00:01:02] VJ: Yeah, sure. Toma is younger.

[00:01:05] TP: Okay. Toma Puljak. I'm 22-years-old. I've been a software developer for about six years now I guess. I've been at Codeanywhere for the past, I think, three or four years. Where now I'm part of the core team for b2b integrations and IDE customizations. And how it ended up here, basically, software development has been a part of most of my life. And I met

the Vedran through one of my friends four years ago. And then this is how that story basically started.

[00:01:35] SF: Yeah. Wow! I'm surprised that you waited until you were 16 to start your software engineering career.

[00:01:41] VJ: I'm Vedran. I'm 41. As well, I've been a developer for as long as I remember. I started this project as a pet project. I think, it's 2009 or '8. And it got traction really fast. And then Ivan Burazin, he's also a founder of Shift Conference, he joined me back then. And we started the company. So, in 2012, we incorporated in US. And the rest, we'll talk about here.

[00:02:09] SF: Great. Yeah, I definitely want to get into the history of Codeanywhere and discuss some of the evolution of the products through the last like 12 years. But before we kind of jump too deep into that, we've been at the Infobip Shift Conference in Zadar, Croatia all week. My first time here, as I mentioned. It's Game of Thrones country, and it's been absolutely great experience for me so far. But how has the conference been for you? And I guess how does it compare to prior years?

[00:02:35] VJ: I was, from the beginning, involved in conference. Not actively. But as I said, Ivan was my co-founder in Codeanywhere. So, from the first humble beginning, this conference in the last few years started to look very, very serious. I mean, I would compare it to biggest conferences in Europe.

[00:02:50] SF: Yeah. I mean, from my experience my first time here, I don't see any difference between coming to this conference versus any other top-tier tech conference. I think it's a very high-quality presenters, high-quality companies that are participating.

So you, Toma, you presented new features of Codeanywhere. And I definitely want to get into that. But before we get too deep, can you first explain what is Codeanywhere?

[00:03:13] TP: So, code anywhere is basically a development platform that aims to standardize move development environments to the cloud. And we use a concept called the infrastructure as code that's been previously associated with DevOps. But we translate that concept into

development environments. And we basically aim to revolutionize how software development works.

[00:03:33] SF: Yeah. And we all attended Sean Wang's talk this week as well on the idea of end the local hosts, which is essentially the idea that development environment should be moving in the cloud. And he was saying, in a lot of ways, this is still early days. And there's still problems to be solved in the space. And it's an emerging area.

But as you mentioned, Vedran, incredibly, the first Codeanywhere prototype was started in 2009, which is over 10 years ago. And we're talking about it being early days now. So, you were super super early to this idea of coding in the browser. Can you talk a little bit about the initial idea and the initial prototype and some of the history of the company and the platform?

[00:04:11] VJ: Yep. So, it was really early. And I think that's one reason that I got the traction so fast. And I was doing a lot of web development back then. And I was kind of frustrated that I couldn't move around easily. And there was this Google Docs starting to become popular. And it offered the ability to Gmail Google Docs. And they offered the ability to access your documents or email whatever you are. You just needed a device.

And that got me thinking, "I need some tool for development that I can quickly and do some fixes, updates to a website." So, that was the part of my life. And I did a lot of different projects. They were not serious projects. There were no pipelines for a deployment. So, it was very basic FTPing to servers and making changes. And that's how I started the first prototype. It was basically an FTP client in a browser that you could easily make some changes on existing sites.

And I remember, I released a project one night and I got up tomorrow and there was like 60 people already signed up. And by the end of the week, there was more than 300 people. And then I said like, "Well, this is becoming like a –" And then I joined the forces with Ivan. Ivan, back then, he was very entrepreneurship. He had this mindset of he wanted to go to Silicon Valley to make something big. He needed a product. I didn't know that he was just like a guy that was really good. And we hanged out.

When he saw the idea, he said like, "Wow! This is a big – This will change development now." I was like, "I don't believe so." But after a year, we grew. The users kept coming. We added more and more features. I mean that was very primitive back then.

So, later, we moved on to introduce containers. You could run code in the cloud, but that was again Codeanywhere. What I'm talking about, this pre-day, it's Codeanywhere. It was a tool called PHP Anywhere. And it was meant for PHP developers that were doing WordPress and some for PHP sites. So, that's kind of how we started.

[00:06:09] SF: Yeah. And I imagine like the editor abilities back in 2009 must have been pretty rudimentary compared to what we're able to do today.

[00:06:17] VJ: Oh, yeah. Absolutely. So, that was the time when Adobe Flash was, I think. So, I remember that I was kind of thinking that I should do this in a Flash. Thank, God, I didn't go that way. Because I was frustrated that you couldn't do so much in the browser. Of course, you could do much more than I could at that time. But still, it was very, very different to be a developer. At that time, tooling was just emerging for JavaScript and for browser-based applications. jQuery was the thing. So, there was no React. There were no workflows that we are using now and we take for granted. They didn't exist back then. It was very a different time.

[00:06:54] SF: Yeah. I mean, I was a engineer during that time as well. And a lot of the frontend development was either very, very basic, or you basically got yourself into a place where it was just like a spectrum of spaghetti code because there's not really any framework for managing. And it's just like jQuery files and stuff all over the place.

So, you talked a little bit, you mentioned infrastructure as code. And that's something that we've covered on the show several times most recently with a company called Bridgecrew. But you're essentially applying that to development workflows. Can you elaborate a little bit further on how applying infrastructure as code concept to the development workflows with Codeanywhere?

[00:07:29] TP: Well basically, infrastructure as code is just code that describes some infrastructure. And we basically use it for development environments in a sense that you describe your development environment configuration through code.

And the benefits of that are, first, because it's code, it can become part of your source code. So, you can just version control it using Git. And now, then you basically know what's going on in that configuration. Why it's there? Why it changed.

But also, recently, we adopted the dev container standard from Microsoft that also includes configurations for your IDE as well. So, those IDE configurations can become part of your source code again, but be isolated from basically all of your code in a sense that it's contained in one configuration folder. And basically, that configuration folder is your, let's say, infrastructures code concept translated to development environments.

[00:08:22] SF: Where did the dev container standard started at Microsoft and that's now a standard? Do you know sort of the background in history on where that standard came from or was developed at Microsoft?

[00:08:33] TP: So, they developed it for basically allowing people to spin up containerized development environments from VS Code, I think. I think that was the initial idea. So, they wanted to make it easier to basically create any containerized development environment just from clicking a few buttons from VS Code. And they now offer like templates for creating different environments from VS Code, C#, Node and such. And they basically use that standard not only VS Code but for GitHub code spaces as well. So, we saw the potential there to implement that standard rather than implement our own.

[00:09:10] SF: Mm-hmm. Right. And then when I started my career, much like you in engineering like 20 years ago, when you started a new job on your first day or maybe even your first couple days, you spent most of that time just downloading an IDE, setting up a compiler, setting up a local host environment. Getting ready to work essentially on your local machine.

And you fast-forward 20 years, and most people are doing the same thing when they start a new job and they're spending all that time basically setting up their machines. Why do you think that we've kind of held on to this like antiquated way of doing things? So much stuff has moved to the cloud. We do documents. We do spreadsheets. We do taxes. Like, all this type of stuff.

Email have moved to the Cloud. Yet we're still holding on to this like localhost environment for the people who are actually building the products that are now existing in the cloud.

[00:09:58] VJ: I would just compare again to a DevOps revolution that happened, I'd say like 10 years ago. For containers, shipping code was for non-enterprise products and companies. It was difficult. You need to have a dedicated engineer who would need to set up a server or a cloud environment. It doesn't matter. But he would have to do that by your instructions. And there was no guarantee that it will be set up in exact way that your software is written for.

So, now, we take for granted. Like, you package your code, you create artifact, and you deploy. And you know it works. And that wasn't the thing. As far as I remember, we struggled a lot when deploying code. There was a difference between staging environments, between production environments. So, you take that for granted.

And, still we develop on a local machine, as you said, and we still have those problems. If we are working in a team and we don't have the same environment, there are always issues with compiling code, with running code, with previewing your application. So, it's all those little things that can take so much time. And they drain energy. They can really be a turn off for entire team. And that's something that we also experienced as a company and talking to – especially about in this journey with Codeanywhere, we had a lot of contacts with other developers, with other companies. And we figured out that the main pain point is onboarding. That's one thing that we could contribute to as a product.

We are not alone in this journey, as Toma said. Even Microsoft has put a great effort on how to solve that problem. And I think that they're doing a great job with container. It's the first standard that we know of that's trying to standardize cloud development environment is described.

So, when we figured out that we could run, that we could execute this containerization configuration inside our environment, it was kind of no-brainer for us that we should move that way and offered pretty much what code spaces offers.

[00:11:54] SF: So, how is, I guess, adopting the dev container standard rather than doing your own version? Like, transform the products? As it just made essentially your development my

cycle much easier because you're following the standard? Or what were some of the advantages that you got from using the Microsoft standard versus doing your own thing?

[00:12:11] VJ: Our standard was we could describe pretty decent and complex environments. And still, it was very similar to Docker Compose. But we figured out it's missing code editor configuration that you could configure your preferences and extensions and so on. It was missing so much stuff. And we were looking at the containerization and we saw, "Okay, they have it all here. It's transparent. It's open sourced." So, it was kind of no-brainer. There's no point to trying to promote another standard. Especially, we are a company that can't be compared to Microsoft.

But we don't believe that it's – we don't even need our standard. We just need something that can be useful. And you can use that to describe your development environment. And we just need to support it in our product.

[00:12:54] SF: From part of what's happening when someone's using Codeanywhere, is you're creating these like ephemeral environments, right? So, I guess what makes that different than just local Dev environments?

[00:13:05] TP: So, with ephemeral development environments, the idea basically is to spin up the environment whenever you need to contribute to the project. And then when you're done, you just release all the resources. And the main difference is that every time you do so, you start from scratch basically. And when you start from scratch, there's no way for the environment to be different from anyone's machine. Because if you start from scratch, you follow the same steps to get to that environment. You should end up in the same environment. Whereas, locally, you're not sure if – Do you want to start over?

[00:13:38] SF: Yeah. Maybe I can set you up again. With Codeanywhere, creating these ephemeral environments. So, how is that different than just a regular local development environment?

[00:13:48] TP: So, the thing is, with ephemeral development environments, the idea is whenever you need to contribute to a project, like opening a pull request or such, you spin up a

fresh environment from scratch. And when you're done, you basically destroy the environment, or the environment destroys itself, and that's it.

And the difference between that and local environment is that you basically start from scratch, so you always know what you're going to get set up with. Whereas with local environments, you're not sure what each machine has. Because, I don't know, if maybe I installed Java 18 on my machine, you had Java 11, those environments are very different. And going from those environments to the same point is not the same journey. So, with the ephemeral environment, you're always guaranteed that the environment you're going to end up with is exactly the same.

[00:14:35] SF: Does that still allow, I guess, like personal customization? A lot of people, a lot of developers, like, "Oh, I want my IDE to look a certain way. Or I want certain combination of keystrokes that do certain things." Can I still have that level of customization?

[00:14:48] TP: Yeah, of course. The dev container standard, as we said, actually includes configurations and customizations for IDEs currently for VS Code and code spaces. And those preferences are meant to be set for, let's say, project-specific preferences. So, I don't know if you're developing in Python, you need a Python extension. And you would set up the Python extension preferences just for that project.

But personal preferences, like, I don't know, themes, key maps or such, can be again personalized, and they can move with you. Because from Codeanywhere's perspective, you can configure user preferences for your account specifically that will just be for your account. And they'll be basically waiting for you each time you spin up an environment. And those preferences aren't shared. Whereas those, let's say, Dev container preferences are shared and are project-specific.

[00:15:40] SF: Yeah. And I guess one advantage of that is that, just like my email or something like that, I can basically move from machine to machine and I'm going to have the exact same experience.

I guess the fact that you're spinning up these new environments each time and tearing them down, is that like a inefficient process for someone, I guess? Because you're essentially booting

up this entire environment each time, and then it goes away when the person's like done, whatever that they're doing.

[00:16:05] TP: So, it would be inefficient if it would – it was just so literal, like, setting up a laptop and then destroying the laptop. So, that would be inefficient. But when talking about this technology that uses containers, well, containers can be spun up and destroyed pretty fast and easily. And it actually is more efficient to spin up the environment each time and destroy it each time. Because, first, you're basically saving resources. You aren't using them when the developer is not really contributing to the project.

And again, we're moving back to the ephemeral workflow where you need to start from scratch each time to get to the same point of the environment. So, if you keep reusing the same environment in the cloud, let's say, for a couple of weeks, you end up with the same problem of different environments, because you're going to install something in it and such. But when starting from scratch, each time you create a commit or anything like that, you are basically guaranteed to end up in the same environment.

[00:17:03] SF: Right. And I guess even if there is some like boot up cost in comparison to the kind of like technical debt that a company's taking on each time, like someone makes a library update that then breaks other people's code when they do the check-in to their local environment and they have to do the updates, you're saving that cost significantly by essentially building all these things in the cloud.

So, beyond some of the challenges that you've mentioned that you're essentially helping development teams with essentially the standardization, and essentially everybody can kind of get the same infrastructure regardless of the constraints of their machine, what other challenges are you helping teams solve over a traditional way of developing software?

[00:17:46] VJ: As Toma said, onboarding is very frustrating for developers. And it can consume so much time. And it can drain energy. And it's very demotivating for developers, especially if it takes so much time. Also, when you're working on a project, you need to check out to a different branch. For example, if you have old version of your software that you still have to maintain and you change so much in your infrastructure, development infrastructure, for example, database

version, language runtime and so on, you need to make all those changes on your local environment usually. And those can take, again, so much time. They can cause another set of problems.

You then apply a fix or you make some changes to your code. Then you have to move back to your current version. And you still have to, again, repeat all that process. And you do that again and again. And it's just a waste of time. And I think for developers' time, time is money. And also, not just money. It's how they feel. And it's about their happiness.

I think if you provide them with a way that they can just work, that's enormous benefit. And there is of course some trade-off, as you said. You need to wait some time until that environment is spun-up. But there are ways to compensate. So, we can talk about it later. There is a pre-made environment that can be created in a background. So, when you commit something to your branch, who can be triggered? And an environment can be created and is just waiting to be started. So, that reduces a time from, let's say, minutes to milliseconds to spin-up a fresh environment.

[00:19:17] SF: I guess that's kind of similar to how like frontend code is like prefetching. You're essentially prefetching the environment beforehand setting it up, so that's like ready to go in the background?

[00:19:26] VJ: Yeah, it's passes through all those steps that needs to be for complete environment. For example, it can pull all Docker images that you need for databases and other infrastructure primitives. And it can also run in its script. So, all packages are already pre-pulled. And some steps are already done. And you end up with ready-to-go environments. So, you just need to spin up the environment that already has everything in it.

[00:19:53] SF: And there's a bunch of different potential use cases of Codeanywhere, from like straightforward, like software engineering, we're just going to move to Cloud. We're going to do our development there. To using it as like potentially an interview platform too. There's also a number of like collaboration features. What are the typical use cases that you're seeing from some of your customers?

[00:20:14] TP: So, software development is the obvious one. Companies using it for internal development. But we also saw potential use cases in companies that are attracting developers to their platforms, for example SDKs or API platforms. They can use this such of setup to basically create really fast environments for their examples on their documentation site.

For example, you offer an SDK to developers. And it takes some time to set up that SDK locally. Again, you have to go through some process. Download the SDK. Configure what needs to be done. With this kind of setup, you can just click a button in the documentation. Get a fully featured environment, an IDE in the browser, and actually start coding on that SDK.

And we basically call that time, or it's called the time to first Hello World for – that's the metric, basically. And we take that time the first Hello World from basically hours to seconds, because it just takes that click of a button. You get a fully featured environment. You can get your API keys already set up in the environment.

You mentioned, of course, interviewing platforms because of the collaboration features. So, we can essentially remove the need to basically code in Google Docs, because Google Docs is a collaborative. And if you have a fully featured collaborative IDE, you can actually test out, I don't know, terminal skills with the collaboration terminal and coding skills inside the text editor.

Another potential use case is we work with a company that uses this to present their platform to their clients so they just offer the environments to their clients so they can play around for a week, for example, and then test out their platform using our environments. Because, again, it removes the needs for their clients to go through the setup process. And they can just really make a seamless experience in trying out the platform.

[00:22:11] SF: Yeah. And I think one of the things, Toma, that you said during your talk was that you're really transforming this experience for developers from essentially **[inaudible 00:22:21]** read through readmes, go through a configuration. Try to run it. Doesn't work. Then you go back and read the readme again and go through all this configuration to essentially click a button, start coding. Which I think is also – like, you can kind of see the inspiration that came from Google Docs. Because Google Docs, like, there's a lot of things that you can do in it. But you

can give it to almost anybody, and they can start typing and find value in it. And it seems like you're sort of taking that philosophy to essentially the development world.

So, what are the types of businesses that you're actually seeing make this shift from local development to essentially cloud-based development? Is that newer tech startups? Or are these older legacy systems? Or products as well? Or even large enterprises? Are they all making this shift? Or is it you're seeing sort of this concentration of people moving the cloud in certain verticals?

[00:23:13] VJ: That's a really good question. And we are constantly discussing this with our team. It's funny that in the last year a lot of, let's say, old companies that are in business for more than two decades, they approached us when they saw a new product that we still didn't release for b2c.

In our b2b contents, we had a lot of inbound from older companies that are struggling with keeping their environments ready to code. They have a lot of problems with onboarding. Keeping developers these days is hard. And I feel that like companies are now fighting for developers. And they want their developers to be happy and to remain in the company.

And we see that they approached us with how to solve the problem of keeping their environments ready to code. I was very skeptical about these older companies as they have these traditional way of doing development. But seems that times are really changing. And I believe that all companies, no matter size, can benefit from this approach.

[00:24:17] SF: Yeah. And I think one of the points that Sean Wang made in his talk was that you don't need the highest end laptop to do development if your development essentially exists in the cloud. Because then you're not depending on all the resources of the laptop to essentially run the code locally. So, there's – I think beyond the efficiency gains that a company is getting, you're potentially getting essentially cost savings, because you're not having to give all your developers a ten-thousand-dollar machine that you have to then replace every single year.

So, how does a company go about like making this shift, I guess? Like, what's the process? Is there downtime? What's your recommendation for someone who's kind of moving from –

Essentially, it's like a movement from on-prem to cloud, but they're moving on-prem development environments to a cloud-based development environment.

[00:25:06] TP: I think the first requirement is to have this mindset that you need to package your application. And if you are already using containers in that process, I don't believe there is a blocker for moving to ephemeral environments. As long as you already have those pipelines set up, those can be translated to development environments.

And then if you implement those in development environments, you make that process easier for your DevOps guys, because you already create a lot of configuration in your development environment that they can look into. You don't need to communicate through them by, I don't know, emails, or writing documentation, or instructions. Many, many of these dependencies are already included in your development environment configuration. But if you have some other ways of deployment, it might be a problem. So, I think having containers as a part of your workflow is a requirement to move this way.

[00:26:00] SF: And then does it help if someone's already using like an infrastructure as code type of deployment through like Terraform or something like that to make this migration to using Codeanywhere?

[00:26:10] VJ: I'm not sure about how it's related to Terraform. And Terraform can be used in multiple ways. But if you are packaging your application services in containers, and if you are already creating those artifacts, then you are aware of all of those goodies that kind of packaging offers you. So, it's the same way for development. And if you can describe your runtime environment to build in a container image, I think you are good to go for making this shift in development environments.

[00:26:36] SF: I see. And then if you have like certain requirements, like, MySQL database for application storage, Redis for caching, Redshift for a warehouse, how does the configuration and setup work for getting started with Codeanywhere?

[00:26:50] TP: Basically, exactly the same as you would set it up locally, but translate it to code. So, the dev container standard actually relies on Docker and Docker compose. So, basically

anything that Docker compose can run, it can be run in the cloud. And not only that. If you move your environment to the cloud, actually, if you need to, you can connect to some shared database if that's a requirement. So, you're not – And that again can be standardized and included in parts of your configuration so you can make the development process easier even if you're packaging the entire development environment into that cloud environment or if you're connecting to Redis externally or a database externally.

[00:27:33] SF: And then I guess you're – as part of the configuration, you're essentially supplying whatever credentials you need to connect to those different services.

[00:27:42] TP: Yeah, of course. Because each user that tries to create an environment needs to be authenticated in the system. We can basically pull any credentials from their account that need to be pulled and inject them into the environment with, I don't know, environment variables or anything similar.

[00:27:57] SF: I see. Yeah. You're, I guess, kind of mimicking the same experience that you would get with developing locally, where you can customize some of those things to connect to like your instance of MySQL. But you can do that essentially in the cloud.

[00:28:08] TP: Of course.

[00:28:09] SF: What are some of the main, if any, limitations of running your development environment in the cloud versus developing locally? Are there limitations?

[00:28:18] TP: So, the current limitations – I mean, people always ask us this question. And there are questions like what about native development? and what about Mac development? Windows development, and such?

And currently, what we are 100 sure is that everything that runs on Linux will run inside these containers. And with native development, it, of course, can be done, but it would require some additional setup using, again, some external tools that will allow you to actually see what's going on with these applications.

But other than that, I mean, of course, again, Apple software is locked into Apple software. So, that of course isn't really easy to transfer to the cloud and develop in the cloud. And these limitations are unfortunately out of our control. But we're trying to do everything we can to support basically any development workflow.

[00:29:10] SF: Yeah. So, something like Android development, essentially, could run an emulator to run your code.

[00:29:16] TP: Yeah, of course. You can run an emulator and then connect that emulator from your local machine again.

[00:29:21] SF: And then I think one of the like seemingly, I guess, unique features of Codeanywhere is also the live collaboration. So, why did you build this feature? What was sort of the inspiration? And how are you seeing people use it?

[00:29:34] TP: So, live collaboration, I basically was, I think, brought onto the company to work on the collaboration. So, we started off working on a VS Code extension for collaboration. And that was at the time that actually I think Live Share was starting up and starting to gain traction. And we basically wanted to create a VS Code extension that we can then port into our online IDE. Because, of course, collaboration should be a part of your workflow. There shouldn't be a need to ever copy-paste blocks of code into Slack or Meet or whatever. Your coding tool should be your collaboration tool. So, we decided to build every collaboration feature we can into the IDE to actually allow you to get the full collaborative experience into IDE.

And then what we did in the end, we moved from that VS Code extension to create our own Theia IDE extension, because Theia is the editor we use in the browser. And I think it worked out pretty well because we got a lot of inbound actually questions about that used for interviewing platforms. Or we actually have some schools that use it for lectures, where the professor can just connect to his workspace and then all the students can come along to his workspace, follow his cursor, looked at what he's editing, collaborate on terminals and such. So, it really creates the seamless collaboration experience all the way around.

[00:31:00] SF: Yeah, that's amazing. I wish I had had something like that when I was doing my undergrad. We were still turning in our assignments with floppy disks back then. So, it's good to see that the university education is starting to modernize as well. So, you mentioned the – You've mentioned the Dev container standard for Microsoft. What are some of the other technologies, your tech stack essentially, that you're using to build Codeanywhere?

[00:31:25] VJ: We are using Kubernetes as our workspace orchestrator. And we are using Kata Containers and Firecrackers to create those isolated pods. I mean, basically, a workspace is in our name for development environment. And translate it to infrastructure, it's a pod. So, those are technologies that we use to spin those in the cloud.

And we do JavaScript – I mean, Typescript and Go languages as our development languages for Codeanywhere. I'm not sure about – We use some databases, messaging technologies like **[inaudible 00:31:59]** and – I'm not sure.

[00:32:01] TP: Yeah. That's it basically. Yeah.

[00:32:03] VJ: It's very techy. Like, infrastructure product. So, there is, of course, a programming language and development. But we rely heavily on technology.

[00:32:11] SF: And then how does, I guess, Codeanywhere compare to some of the competitors in the market? Companies like – Or products like Gitpod, and coder, GitHub, Codespaces?

[00:32:20] VJ: Good question. We started a new product – So, just a disclaimer. Before, as I was talking how we started, we were an online IDE company for a while. We created a product that was really useful for web developer students. I would say really small teams. But we just couldn't scale for more serious development. But we got a lot of inbound on requests to basically make Codeanywhere more powerful.

We also evolved as a company and as a team in the last decade. And we started to work on a new product with this ephemeral idea about three years ago. And it's funny, in that time, Gitpod was gaining traction. Microsoft announced Codespaces. Coder was the guys – they did the VS

Code in the browser. And it's called Codeserver or something. And it got a great traction. And they started the Coder company.

So, before that, there was a Cloud9, there was Nitros, there was – I can't remember. But one company particularly inspiring for us was Codeenvy. And they are now called CodeReady.

[00:33:27] TP: Oh, yeah. **[inaudible 00:33:28]** CodeReady from Eclipse. Yeah.

[00:33:31] VJ: And they were the first that kind of envisioned this orchestration of development environments that was like a wow effect for me. And then as we worked on our product, other companies also offered their products. We saw that it's very similar. And this time, we didn't want to go any other direction. And we ended up in, I would say, very similar products. So, Gitpod, Codespaces, Coder is maybe a bit different. But they offer very, very similar feature sets. They use underlying different technologies. But I can't really make big conceptual differences. And I don't think that's a bad thing. It shows that they're all like on the same way to create a better experience for development environments. And it's a validation that we are moving in a good direction.

[00:34:20] SF: Right. Yeah. And if you compare even local development, like IDEs, purely Jbuilder, to Eclipse during its heyday. And functionally they're not that different. It really comes down to what makes sense for a company. And there's little features that maybe are better fit for some people than other.

So, you've been – Vedran, you've been at this for a number of years now. What are, I guess, some of the lessons that you've learned as both an engineer working on this product as well as an entrepreneur developing Codeanywhere?

[00:34:54] VJ: Yeah, it's hard to describe this in a few sentences. But one of the things that I figured out is very valuable, it's having a good team, and good and flexible team, and that understands each other, and that you can have a good conversation and understanding. So, that's one of the things that I would say are crucial for a company to succeed. Especially if you are doing tooling, it's very hard. It's very hard. You are selling your product to folks that are probably better than you. And many of them have more experience, more knowledge. And you

have to cover many aspects of this business. So, having a good team, it's very crucial. And also, ability to adapt and to change.

I think if you move to a wrong direction, and it happened to us a lot during this 10 years in Codeanywhere, you lose precious time. And it's really hard to catch up later. So, one thing that you need to be aware is to always question your product. Are you moving to a good direction and have a good relation with your customer? Good feedback. So, those are all important things that I wish I knew before.

[00:36:00] SF: Yeah, I think that's the hard lessons of being entrepreneur, is you learn a lot of these things along the way, and you wish you could go back and tell your younger self these things. But that's why the second company is a little bit easier.

As I mentioned, this is my first time in Croatia. And in the past life, I did a lot of work with Infobip, which is also headquartered in Croatia, and it's a monster company now. It's worth over a billion dollars. Completely bootstrapped.

But beyond, I guess, Codeanywhere and Infobip, what's the like Croatian tech scene in start-up scene like? Is this an emerging market? And for not only talent, but I guess for companies as well?

[00:36:39] VJ: I think we can both give our opinion, especially since there is so much age difference. I was around as a software developer for 20 years. It's about the time that I got into college. And I started working immediately. So, back then it was a very, very different time in Croatia. There were no meetups. There was no scene, you know, like these days.

And being a developer was usually meant that you work for a company and you work on a product. Entrepreneurship was a word that I only discovered later. But in last 10 years, I see that it's very different. It's evolving very fast. It's still not on a level that you will see in the US or I would say Western Europe. But at Croatia, as well as the other parts of Eastern Europe in last decade, it's very, very different, different story. Younger people, they have this mentality of an entrepreneurship and socializing. And they do meetups. They do events. And now, it's much, much better than 20 years ago.

[00:37:42] SF: Mm-hmm. And Toma?

[00:37:44] TP: Yeah. I mean, I agree. I mean, I can't speak for 10 years ago, because I was – I don't know which grade. Not even high school. But I think, really, the development community grows each time. I mean, we see conferences popping up in a couple of cities each year and meetups as well. And especially in Zagreb. Zagreb has a really nice tech scene. You can probably go to meetup every day.

And we have, I don't know, I think two unicorns in Croatia currently. And I think if any developer shouldn't even have a problem finding a job here, I don't know, you can just probably walk into any company because everyone's looking for developers currently here, because every company is growing. But developers are still catching up.

But I think even amongst my peers, I can see that everyone is actually working while studying essentially. And even some high school – I started in high school as well. And I think that shows some trend of like the development community growing actually in Croatia.

[00:38:46] SF: Yeah. And I think you know the Internet and all the things that we have available to us to sort of stay connected and also tapped into different areas of the world has really sort of democratized the idea of like entrepreneurship. You're seeing a lot of entrepreneurship in the tech scene in parts of the world that are not typically associated with that, whether it's Eastern Europe. Or there's a lot of growth in Africa as well. A lot of fintech companies are coming out of parts of Africa.

So, I guess, like, what's next for Codeanywhere? Are there things that that you can share in terms of future roadmap items or things that you're really excited about?

[00:39:17] VJ: Yeah, we are going to definitely keep moving in this direction. At the moment, we just have a new business-to-business product that can be installed on-prem on your environment. The plan is to offer this product for our b2c customers by the end of the year. We will also keep supporting the old product. IDE will still be available for all our existing users. And

if they don't want to make a shift, it's fine. We are not going to kill the product. But the company itself will be 100% focused on a new product from now on.

And we'll be adding more and more features. We plan to support Dev container standard to its full extent. So, at the moment, we support major features. But there are still like configuration options that we need to support. We also have inbound for supporting OpenShift, which is kind of hard. But enterprise companies are – many enterprise companies are using OpenShift's platform. So, that's also one of our goals for b2b product. And it's early to say, but we have serious plans for open sourcing products. So, that's also on our map.

[00:40:26] SF: Well, that's exciting. I guess is there anything else that you would invite the audience to know as we start to wrap up here?

[00:40:32] TP: Well, I mean, nothing major. I just want to say that we really are invested. And we think that development environments are moving to the cloud because we see a lot of benefits. And we just hope that more and more developers will continue with this shift towards the cloud, because we see it as something that will provide a more seamless flow and allow developers to basically just focus on their work and basically just be developers. And that's our vision and what we're invested in moving forward.

[00:41:02] VJ: I would add also, that the end of localhost doesn't mean that you're not going to use a local environment. It's just the way that you can use it in, I'd say, a different way that you can utilize containers to really boost up your development experience and make it easier. And then cloud component is just an add-on to that workflow.

[00:41:24] SF: Yeah. I mean, I think it's something that makes a ton of sense. And if you look at some of the biggest technology companies in the world, like Facebook and Google, they've made this move internally with their own versions of these types of products. And I think a lot of times looking at those really huge companies is a good way to sort of triangulate on where the tech scene is moving, because they are typically hitting scale problems much, much earlier than other tech companies. So, it's not that many tech companies have a hundred thousand engineers like they have at Google.

So, I want to thank you both for coming on the show. It was great to be able to do this in person. And I really enjoyed the conversation. And I'm excited to see where Codeanywhere goes and see how it continues to develop.

[00:42:02] TP: Thanks for having us.

[00:42:03] VJ: Yeah, thank you. And hope we talk sometime soon.

[END]