EPISODE 1301

[INTRODUCTION]

[00:00:00] JM: Apache Druid is an open source analytics platform that came out of the company Metamarkets, which was an advertising technology company. Today, Imply.io is one of the leading new analytics providers for companies that want to do what is commonly called operational analytics. This is something where perhaps the data is coming in fast, you want to have slicing and dicing occur, you want to perhaps build interesting interfaces around that slicing and dicing of data. And Jad Naous is one of the people who is in charge of making this company tick. He's the VP of Engineering and Product at Imply. He's a friend, and he's an engineer and investor that I deeply respect.

In today's episode, we talk about how Imply is used to handle large scale analytic workloads. And we also talk a lot about market positioning. What are the different players in the big data analytics market today? What does Snowflake do? What does BigQuery do? What does Databricks do? Who are the different constituents? And what are their use cases? What's the Venn diagram of these different big data analytics use cases? It was a great wide ranging conversation. And I, as always, enjoy talking to Jod.

[INTERVIEW]

[00:01:20] JM: Jad , welcome back to the show.

[00:01:21] JN: Thank you for having me again.

[00:01:23] JM: I am thrilled to have you again. I respect your opinion in infrastructure at a very deep level. And so I'd like to start with a fairly open discussion about a piece of data infrastructure. And this is just a broad-based data infrastructure question. I asked it to get just your general temperature reading of the market. And gradually, we'll get into more specific topics. But I've been really thinking deeply about what the ecosystem difference between Snowflake and Databricks is, and the Spark offering, that stack. Do you have a perspective on those two sides of the market?

[00:02:00] JN: So Databricks is today trying to refashion itself as a data warehouse, but Spark is a general processing engine. And so if you look at Databricks' history over time, they kind of started with, "Hey, we're a general processing engine," but they couldn't really figure out what to sell there. They did a lot of ETL, because that's really what requires a lot of generalized processing. Then they started to talk a lot about data science and machine learning. And much more recently, you hear that they're talking a lot more about data warehousing.

With Spark being such a general processing engine, they could do all these things. And they could go and talk about all these different pieces, where Snowflake knew what it was all the way from the very beginning. They're a data warehouse. They are replacement for all the data warehouses that are on-prem, but in the cloud. They're cheaper. They're easier to use. Scale out with, start, and so on.

A lot of Databricks' is kind of use cases today are still ETL. Like that's their bread and butter. But from a high-perspective, from like a marketing and mission perspective, there's a big momentum in the market today behind data warehousing. And so that's why they're moving there, and they're playing there. This isn't a judgment on whether they are a great data warehouse or not. I don't really have much of a say there. But from a marketing perspective, that's how I perceive it.

But what I'm really interested in, to me, they're both Databricks, and Snowflake, and BigQuery, Redshift and all these systems, all live in the same area of the market. All live in the same, I guess, quadrant of the Gartner, data analytics magic quadrant, which is basically to say they're mostly focused on cold analytics. They're mostly focused on reporting queries, on things that professional analysts need or professional data engineers need to build out dashboards for executives, or for business users. These are mainly batch-oriented reporting workflows that work on large quantities of data. They are not highly concurrent workflows. They don't have hundreds or thousands of users who are trying to access this data at the same time. They don't have a ton of ad hoc queries where people are hundreds or thousands of people at the same time trying to slice and dice the data. These are what I like to call cold analytics. Kind of they really focused on doing reporting workflows that don't necessarily need to be real time. Queries could take minutes, hours, maybe an order of days, but that doesn't matter as much, because the workflows are much more reporting-oriented.

The area of the market that I'm more interested in is kind of like the other end of the spectrum. And this is where Imply really lives, what we do. It's our bread and butter. And I call that area the hot analytic space. And here, in this world, it's a different use case. It's a different user. It's a different workflow. And I'll give you a little bit of a kind of more concrete idea of what I'm talking about. Over the past five years, I've been noticing a really interesting trend in SaaS applications. You can like see that more and more sass applications have some kind of analytics inside them. It's a little bit freaky, because five years ago, we wouldn't think about a SaaS application. I'm talking about enterprise applications, of course. Like you wouldn't think about them necessarily doing anything other than the function that they were built to do. But now, having analytics inside your SaaS application, it's kind of standard more or less. And if you don't have analytics inside the SaaS application that you're using, it's kind of like, "Well, look this is upcoming," or this is like you got to pay for the enterprise feature. And the enterprise feature has the analytical capability. But I think this is a trend that's going to continue. And the reason for that is I think more and more enterprise users really need data in order to do their day-to-day jobs. And they have to ask hundreds of questions out of data. They have to answer all those questions immediately. They have to explore and understand what's going on. And these people are not professional analysts. They cannot look at data in the abstract. They need data to live in their domain to speak their language. And that's why we're seeing analytics kind of move inside SaaS applications, inside the tools where people work, because it's how they think. It's kind of there right at their fingertips. It's the world that they live in. The questions that they can ask are formulated not in SQL, but formulated in the language of the application or the domain that they work in. And I expect that this is a trend that's going to continue and we're going to pretty much see analytics in pretty much every SaaS application out there.

The future that I see is one where when you go and build an application, you have your transactional database or your metadata, like MySQL, or Postgres, or whatever. And then you have your analytical database sitting right next to it, and that's Druid, hopefully. And this is very similar to the way that kind of Elasticsearch has become ubiquitous and has become kind of synonymous with search, like, "Hey, I'm building an application. I need search." Boom! Elasticsearch. And so that's how I see this world playing out. And the interesting piece here is you scale out and build systems where you have hundreds or thousands of users executing analytics ad hoc on your data, you need systems that are really built and designed to be able to do these kinds of hot analytics use cases. So from a landscape perspective, specifically around analytics, I kind of divide the world into these two areas, the cold analytic space and the hot analytic space.

[00:08:12] JM: And we can go a little bit deeper there. But I actually just want to zoom on a different facet of analytics, and that is embedded analytics. You touched on this a little bit. But essentially, the idea that you want to make it very easy for users to embed analytic experiences into applications. So like I think the great idea I think about is like the quantified self, right? The ultimate quantified self-platform where you've got a bunch of sensors hooked up to your body. You've got the dream toilet that analyzes all of your waste materials. You've got sensors on your sink. It's the best. That's the dream vision. That's the moonshot. The toilet moonshot. But yeah, you have all these things like instrumenting all this data, and it has to get munged in data engineer and stuff. And then you have to have a really good analytics layer, ideally, an interactive analytics layer. Even if you're just a “non-technical user” who wants to use this out of the box quantified self-platform, you want really cool graphics, and slice and dice ability. So if we're talking about that kind of domain, is that the sweet spot for Imply?

[00:09:19] JN: That's a good question. I think that, for us, we have two sweet spots. One is this embedded analytics where people want to build out applications that have analytics in them at large scale. And so they need something like Imply to be able to handle that level of analytics level, that level. I guess the key requirement here for these embedded analytics is that they need to be low latency. They need to be real time. They need to work kind of in a way similar to like what users, what consumers want out of applications. They want to know now what's happening right now. They want to relate it to their current reality in the outside world. And if something, if you're looking at data inside your app that's a day-old, it doesn't gel very well with a consumer-like experience.

And these people are also not very – They're not experts on data. So they want to explore and kind of like dig in and drill down into that data and look at interesting trends and slice the data in certain interesting ways. So it's not like a professional analyst who constructs a SQL query that needs to execute in a certain way in order to get the data that's required across multiple tables that need to be joined, right?

So from an embedded analytics standpoint, Druid and Imply has this sweet spot, which is where people require real time analytics, both in the sense that it's immediate in terms of what's happening in the outside world. Like you can see something in the outside world and it's already recorded in your dashboard. And it's real time in the sense of how quickly you can get to insights, how quickly you can get answers to your questions. And so that's what I call the infrastructure side of Imply, the infrastructure side of what we do, which is helping people build analytics into their applications.

But Imply also has another part of the business, which is the more – You could call it next generation BI. And that's where we provide horizontal type analytics to users at a visual level. And so we have an application that we call Pivot, a product that we call Pivot. And you can think of Pivot as the equivalent of Excel for – Like it's a visual Excel for a large scale analytics. It allows you to create all sorts of views, visual views on data, and kind of slice and dice and create formulas and so on so you can get the answers that you need in an aggregated visual way. And so in that space, it's less about building applications and more about, "Hey, I'm a marketing person. I want to figure out like what's happening with my app right now. My team doesn't need to go and build an application to do this for me. I can just go open up Pivot, and look at the marketing data and figure out what's going on." And so Pivot kind of allows people to build walled gardens around which they can go and like explore data and interact with it and figure out what's going on.

[00:12:29] JM: Hearing you talk about this makes me think a lot about the engineering problems that go into building this product. It's why I enjoyed our last conversation. And it's why I'm looking forward to exploring this more deeply with you right now. I saw a tweet from George Fraser from a few days ago, George Fraser, the CEO of Fivetran. And he was essentially critiquing the fact that Uber Eats seems to occasionally drop packets. I don't know if you order on Uber Eats much, but occasionally, like it times out in weird ways, or like a payment kind of fails. And you get a sense that something a little shaky is going on behind the scenes. Doordash is a little bit more reliable. Do you know what I'm talking about? Or is that foreign language to you? You can't play around meals.

[00:13:15] JN: Well, actually, I don't use Uber Eats very often. I often just use Seamless, or Grubhub.

[00:13:21] JM: Oh man! You don’t use Doordash?

[00:13:24] JN: No. So I'm a little bit cheap. And all these companies ask you to pay a delivery fee, and Seamless and Grubhub don't. And so I just got used to it.

[00:13:35] JM: That's powerful actually. Are the mobile apps okay?

[00:13:37] JN: I use the web. Why do you need a mobile app for something like ordering food?

[00:13:42] JM: Well, a lot of times I'm leaving a location. Like I'm leaving the South Bay, and I want to order and I just want to have kick-off the async and then have my food waiting for me as soon as I arrive home. You know what I’m talking about?

[00:13:54] JN: Yeah. I mean, it's not that fancy. It's not going to like track your driver and figure out where they are on the map.

[00:14:03] JM: Look, I'm not going to open up a mobile web view in order food through that. You're not going to –

[00:14:09] JN: I think that exposes the age difference between us, yeah.

[00:14:12] JM: I guess so. But anyway, seriously, like he’s criticizing Kafka. He basically posted this, and I hope George hears this, and maybe he’ll prove me wrong about this, but I think this is what he was implying, is that in this supply chain of data, we know that Kafka is the weak point, and this must be causing problems for Uber. And basically he's trying to conclude from a user interface experience that Kafka is causing data inconsistency problems. I'm not sure George Fraser knows how much potential there is for inconsistency across an entire distributed system. I mean, you can have failures in React, right? Like essentially on a single node system, you can have data inconsistency issues. So I say this just to tee up a conversation about building this end-to-end distributed systems experience. To me, it seems very challenging to make interactive distributed systems with basically a data warehouse platform. Am I mistaken?

[00:15:11] JN: There’s a pretty big deal. So let me maybe first comment on George's tweet. I think he’s just being tongue in cheek here. I don’t think he –

[00:15:20] JM: I know he is. I mean, I'm giving him a total hard time, a really total hard time. I'm being a massive troll. I feel like George is somebody who respects trolls. So I’m kind of a massive troll.

[00:15:31] JN: But the second piece is that actually building real time distributed systems at massive scales are huge. It’s very difficult. It's a huge undertaking. And it's not just about like, “Okay, it's a distributed system that can deliver a query right now at the performance that you're looking for? And like we do this, like we run into this issue in the POCs all the time, which is like, “Oh, yeah, let's look at this system versus that system and like figure out what the query performance is like. Great, okay, maybe like they're comparable.” But the question is what happens when you need to change something in your system? What happens when you need to increase the number of nodes? What happens when you need to decrease the number of nodes? What happens when you need to change the query patterns or the load that's actually executing, and so on? And so it's not just about like delivering the performance, but also delivering the day two operations of how you expand. How do you evolve? How do you move forward?

And Druid – So a lot of people will say, “Well, Apache Druid is hard to actually get started with.” That's actually true. And that's mainly because it was built to run at scale. It was built to handle all these kind of day two type operations, and maybe like the 90-day operations where you start thinking about, “Hey, let's go scale, and let's go do something else.”

Generally, like we find that like a lot of people forget that aspect of running distributed systems at scale is important. Trying to make sure that these systems continue to run and that they're flexible. And I guess it's always important to keep that in mind. But to your question earlier, yes, it is very hard. And one of the things that we're actually doing right now at Imply is we're redesigning our platform. And we are breaking down every layer of how Imply delivers functionality to our users, including Druid, because Druid is part of our bigger platform. And we’re re-designing it, keeping in mind time to value.

And so this next generation of Imply is going to be a SaaS platform. It's going to be more of a system where send us your data and we'll give you sub-second queries. It's going to be – We're at like starting with the ingestion piece. One of the biggest things that we're doing is moving from a pull model to a push model, where you can send us the files, you can push the files to us, or you can push individual events to us. That's huge, because it means that now you can send your events from anywhere. You don't have to do all this integration in terms of opening up your infrastructure to get the data into Druid and so on.

But the second piece is because it's SaaS, it's going to allow us to deliver a much tighter performance management experience. One of the interesting things that I've noticed in the data infrastructure space is that performance doesn't seem like a core value prop from a user experience perspective, if that makes sense. Like what I'm talking about is like there aren't that many really great performance, like tuning user experiences in the data world. But for Imply what we do is, literally, like our core value prop is to give you the best performance you can get for your analytics, and being able to beat all these other systems.

And so a performance workflow is really core. And we're working on a dedicated performance workflow that allows you to manage the performance of your queries and be able to say things like, "I want an SLA of X on this particular set of queries. And I want an SLA of Y on that particular set of queries, and for Imply to automatically handle and figure out how to deliver that SLA for you behind the scenes. This is a pretty huge undertaking. But from a technical perspective, we have all of the pieces there to actually be able to deliver on that capability. And that's what we're going to be working on over the next 18 months. So to all the listeners there who might be really interested and excited about something like that, hit us up. We've got a lot of positions open for working on this cool stuff.

[00:19:52] JM: Absolutely. It's got to be one of the top-tier companies related to data warehousing, data infrastructure, obviously analytics. I want to reverse troll George Fraser now. So Fivetran represents fairly high-level data infrastructure. And this is pretty powerful. I think that's representative of the broader landscape of data infrastructure today. It's getting high-level. It's getting easier to work with. It's certainly easier. What were you doing during the Hadoop days? Were you tracking data space during Hadoop days?

[00:20:26] JN: During the Hadoop days, I was at AppDynamics. And we were at the time trying to build a metric system on top of HBase. That was really hard. HBase sucks. No offense to HBase people, but –

[00:20:42] JM: Well, I just find it funny. Like that's trivial today, right? The inequivalent applications trivial today. It's like some SaaS solution, right?

[00:20:49] JN: Yeah. Yeah. There's like a ton of time series databases. They just like pop-off and put in.

[00:20:54] JM: Okay. So, strategically, Imply wants to like raise the level of abstraction as high as possible. And you're describing things that are very, very hard. I'm very curious, what's it like to play in the sandbox with today's data infrastructure abstractions? Like what's useful to you?

[00:21:09] JN: Like from an application developer perspective or from –

[00:21:12] JM: Yeah. Like are you guys making use of things like Fivetran, or Kafka, or Kinesis, or Redshift? Like I have no idea what goes on behind the scenes.

[00:21:20] JN: So that's a very good question. For us, our core bread and butter is Apache Druid, right? So this is the open source database. It's all custom code built from the ground up –

[00:21:32] JM: Absolutely. Founded in Metamarkets.

[00:21:33] JN: Yep, found in Metamarkets a while ago, spun out. Netflix, I think, was the first big user. They wrote up a bunch of blogs recently about their use of Druid. But for a long time, we used to rely on Hadoop for ingesting data into Imply for doing batch ingestion. And that was a big pain point for a lot of our customers and a lot of the Apache Druid users. And so we ended up building native ingestion within Druid in order to alleviate that pain. And so now we think our native ingestion systems are actually better than Hadoop. It's not as performant, but most of the users that we know of on Apache Druid today or at Imply are actually using native ingestion. And so that's one piece of software that we want dependency from a big data system that we've kind of dropped out.

Now, when we start talking about push, as mentioning how our next generation products is going to have the ability for you to just push events directly to us. Like we need some system like Kafka. We need a queueing system that allows us to hold these events and then distribute them to multiple parallel processors and so on. So you want to reinvent the wheel on something like that. And the answer is no. We'll just use Kafka.

And the question is always for somebody who is building a database or a data system. What are the things that you just take off the shelf, and what are the things that you build from the ground up? We have me and Gian Merlino, the CTO and cofounder at Imply, we have this constant running debate about whether we want to build out fact-to-fact joins in Druid. And the question for us is always, "Well, do we go and build it out from scratch? Like from the ground up? To be something that's really tightly integrated into Druid that works really well with it superfast, kind of really built out for how Druid works? Or do we go and adopt something like Trino and maybe fork it out and then integrate it over time into what we do? I don't think we have a great answer at this point around which one of these actually works better, because it's not yet clear to us, for example, whether things like fact-to-fact joins are kind of like what we call data warehousing query capabilities are commoditized enough for us to not care about making it a proprietary, or not really proprietary, but more like a core strength of what Druid does.

So you're asking a little bit about abstractions, and what are the different pieces. Like what are the tools in the sandbox that we think about? So Kafka is probably one of the most interesting ones, because at some point you're going to have queueing. And you need something that does it at scale, that allows you to distribute execution across multiple systems, so Kafka or something. Stream processing has been something that I've always wanted to see how we could use, but never ended up in a situation where we would use a stream processor rather than build a custom consumer from these queueing systems. And that could be something that we end up using for our next generation platform.

One area that I've never really understood around these stream processors or like how they get used as their attempt to be databases at the same time as stream processing engines. That's always been something I never fully groked. And then, of course, there's like the hit data syncs, the data warehouses, or the data lakes, or databases, analytical databases like Druid. I mean, I could go on and on in terms of all the various abstractions that could be useful. So like there's the general processing engines like Spark or, really, Hadoop in the older world, or streaming processors in the new world. But it'll be hard to kind of just paint the full picture of the full landscape in the short time that we have.

[00:25:37] JM: Trino. Trino sounds really familiar. Is that the Starburst fork?

[00:25:41] JN: Yeah.

[00:25:41] JM: Or sorry. Sorry, the Presto fork.

[00:25:43] JN: Yes. That's the Presto fork.

[00:25:44] JM: What happened there again? Can you find me? This open source drama, I can't take it. I can't handle it. I need like a people magazine for open source.

[00:25:55] JN: Yeah. So what happened there? Well, Facebook disagreed with the three cofounders of the Presto project. They weren't happy with how things were moving. They left. They started a foundation. They forked-off Presto. Then they joined Starburst. And they renamed Presto to Trina is basically how it played out.

[00:26:19] JM: Gotcha. So Trino, did they diverged meaningfully at this point?

[00:26:24] JN: I think they have diverged. I'm not sure to what degree? I'm not sure exactly of all the differences at this point.

[00:26:31] JM: Gotcha. Okay. So you mentioned Trino just in the context of kind of the next act of Imply, I guess? Or like where you could expand into? Did I understand that correctly?

[00:26:41] JN: Well, so Trino is like a general – It’s like a Query Federation –

[00:26:47] JM: Yeah. Yeah. Yeah. Yeah. I know what it is.

[00:26:49] JN: Well, I just wanted to explain it in the sense of like Query Federation engine, we are a database. So we would be something that sits under Trino, as opposed to like us trying to build Trino. Like we don't want to go and federate queries with other systems. But Trino does have a really interesting capability, which is execute, shuffle joins on multi-stage queries, and so on. And so that particular query capability, that engine piece, is something that could be very helpful for us.

[00:27:17] JM: You wouldn't build that into Druid. You're saying you would slot yourself under Presto and put that within imply infrastructure.

[00:27:25] JN: Well, we're not really sure what we would do there. The question is like, “Okay, well, can we build off of what other really smart and great people have already built? And how can we use that? And there're a lot of options. We don't want to appropriate somebody else's work, but at the same time –

[00:27:41] JM: Here's a question. Let's say you do have some crazy ambitious project, like you want to build Trino on top of Druid and have – Basically, I mean, have the end user experience be the same, but you insert some like querying middleware. Maybe it lets you federate queries to GraphQL for – I don't know, whatever you want to do. In that kind of project, are you able to go to – Or let's say just for the sake of argument. Let's say you do it with Starburst. You're going to do it with Presto. Is that the kind of situation where you can go to Starburst and say, “Hey, Starburst, we want you to give us a solutions architect and help us build this thing.” I've been trying to understand like corporate dynamics for this kind of thing. Can you leverage help from the other big corp?

[00:28:25] JN: Well, it depends on whether or not it helps them with their go to market and with their sales. So if you think of Presto or Trino as a platform for integrating with other databases and allowing you to have a uniform querying experience across all these different systems, then one of the strategies that they could pursue is partnerships with data stores where they could go and query data from. There has been attempts in the past to enable Trino, I think, or maybe Presto, I forgot which one, to query data from Druid. There's a ton of integrations and connectors for Presto and to drop Presto and Trino and two other data stores.

So I don't know whether this is a strategy that they're going after immediately, or like whether Druid is particularly interesting. Most people who use Trino use it on – Or like the biggest use case is using it on data files, like ORC or Parquet, as opposed to using it on other data warehouses or other data systems like Druid.

[00:29:29] JM: So you've been at Imply for two years at this point, or 18 months, something like that?

[00:29:33] JN: Yeah, almost two years. Two years next month.

[00:29:35] JM: Okay, so you're an experienced engineering leader at this point. You've got a lot of experience. What kind of new stuff have you learned over the last year?

[00:29:44] JN: So let me maybe talk a little bit about – Well, do you want to talk about – Like should I talk about the technical stuff? Or should I talk about like organizational stuff, and leadership, and what are you interested in?

[00:29:55] JM: I mean, anything just ordered in terms of impact to how you see the world personally.

[00:30:00] JN: Yeah. Well, one of the biggest things that have happened over the past two years is COVID.

[00:30:08] JM: I remember that.

[00:30:08] JN: You remember? Yeah.

[00:30:10] JM: I remember that. But, I mean, seriously. I'll give a shout out on the air to how nice of a guy you are and how generous you were to me. It was a very difficult time for me. And it was nice to have a friend such as yourself.

[00:30:21] JN: I appreciate it. Thanks, Jeff. And, basically – So one of the interesting things is like before COVID happened, I was actually trying to get more people into the office. We were trying to centralize more in the Bay Area so that we can gain more locality efficiencies, like people working together. And what's happened over the past year is that we were kind of like – We’re forced to go back into, “Okay, let's go redistribute people. Let's go – Everything is remote now.”

And luckily, for us, I mean, we had remote work before. We were mainly a remote culture. A lot of the work that we were doing was open source. And so like there were systems in place to make that work possible. And so it wasn't too hard for us to continue in a remote work situation. But one thing that I've learned over the past year, or like, really, over the past six months, as we were trying to figure out what Imply – What the next generation of Imply looks like. What the SaaS project will look like, is that it's really, really hard to align people on an ambiguous project across multiple teams in a remote world. Like the biggest difference between working in an office altogether and working on Zoom is that you cannot have an ad hoc conversation. You cannot like poke the person next to you and be like, “Hey, look at this. What do you think?” And then like look around you and pull them in, and so on. And that has led to a lot of churn in terms of figuring out what we do, how we do it. Like we would write a PRD and then kind of tens of comments would come in and then keeping track of which comments were answered, not answered, and whether we fixed the PRD to match the comments that were resolved and so on. Just like this churn cycle has been really hard. And I'm surprised. I mean, there are some productivity tools that we've decided are just not good for this kind of stuff.

[00:32:24] JM: Oh man! I don't want to guess. But I guess you can't tell me. Oh man! Oh man! That's a hot one.

[00:32:33] JN: And it's really related to how real time the communication and the changes need to be. Like when you do a comment, how quickly do you see it on your screen? Do you even see it on your screen at all without having to reload the page? Like that is such an important piece of the UX for remote collaboration, that now like we're like thinking, “Okay, well, what do we use? How do we fix this? How do we move forward there? But yeah, there's a lot of like thinking now about how do we work together better remotely with better tools?

[00:33:06] JM: Man, we got to wrap up. But I think we have to wrap up, right? It's been like 45 minutes, or 50 minutes, or something.

[00:33:12] JN: Yeah, 41 minutes.

[00:33:14] JM: Yeah, I wish we could talk longer. I got to go to something. Physical space matters these days. You actually have to travel through physical space to get to places that you need to go to.

[00:33:23] JN: Yes.

[00:33:23] JM: This is a novel development for me.

[00:33:28] JN: Why? You can't Zoom it in. You can have a remote lunch?

[00:33:32] JM: No. Actually not. It's very interesting. Why is your background your office? It's ridiculous, man.

[00:33:43] JN: Because why not? Like it's people are reopening offices, right?

[00:33:47] JM: I actually really like it. I mean, most people like choose an island or like a beach or like some cool pattern in the background. And you're just in an office.

[00:33:54] JN: Yeah, man. We're reopening offices.

[00:33:59] JM: You move to a place where they're still in-office? Or is there an apply office near you? No. You're remote now. But you guys are reopening offices, right?

[00:34:06] JN: Yeah. Yeah, we're reopening offices.

[00:34:08] JM: There's no way that's going to work, right? Like who's going to come back? No offense to your team. I don't mean to undermine you on air. It's just I don't think that's going to work for you. I don't think it's good for Amazon.

[00:34:18] JN: I think you're under estimating the number of people who like to actually meet and work together. Like humans are social beings.

[00:34:28] JM: Alright. Okay. Alright.

[00:34:29] JN: Like think of it this way. Like you're a young person, move, left their parents' home. You're still a bachelor working like – Well, you know. In your situation, you're so used to it, because like you don't work in an office. But like for other people, it's like the office is like where they go and mingle and like have conversations and like make friends, especially like when they move to a new city or like they move out, right? Like that's where they go and build a life, like build a personal life. Like that's where it's all starts, like work. Otherwise, like where would you go make people? It's a lot harder.

And I think like, you can maybe think about the social dynamics at a bigger scale like, "Well, should people ever –" Like when people graduate from school, should they leave their hometowns at all and like go into places like the Bay Area where like they would work on similar things as everybody else? Well, that's a different. Like that's a bigger story. If people don't move out of their parents' houses or like if they don't move out of their hometowns, like they have social lives, and they have these networks that already exist. And at that point, like it becomes a much more interesting question. But I think that's a dynamic that'll only play out over a much longer period of time, as opposed to right now. I also don't think like – Alright, let me just clarify. Imply is not going to ask all the people who are remote to come back to the office. And like our current, like the way it's going to work, it's going to be more optional, and like people will come in like two or three days a week to work with the teams that are local if they want to.

[00:36:00] JM: Alright. I mean, I would be willing to bet you an eth on this, if you want.

[00:36:04] JN: That what? That nobody will come?

[00:36:06] JM: I mean, it's not going to work. I just don't think people are going to want to come back to the office. I think it's going to be mutiny. So we could bet an eth. We could bet a Bitcoin if you want to go high stakes. We could bet a sandwich.

[00:36:19] JN: You know, I'm head of engineering, right? So I could be like, "Hey guys, we'll split this eth, but all of you have to come to the office for like one day every week so we can rob, Jeff."

[00:36:31] JM: I'm sorry. So one day a week? Okay, wait. So you're just trying to get people to come back one day a week?

[00:36:36] JN: Yeah, it's like one to three days a week. Like it's kind of optional.

[00:36:40] JM: Yeah. Okay. That's, that's actually kind of appealing. I mean, do you try to sync everybody? Or you just say like go one day and just like you're in a ghost office?

[00:36:46] JN: We probably want to sync people who are willing to work together.

[00:36:50] JM: Yeah. Yeah. Okay. Huh! Yeah, that actually makes a lot of sense. I like that idea.

[00:36:55] JN: Well, that's what a lot of companies are doing.

[00:36:57] JM: Okay. Alright. I guess I was mistaken. Okay. It's still like – Then if you're at Google, you're in like an office that has 1/5th of the population it supposed to have. It's kind of spooky, right?

[00:37:08] JN: Well, so Google, I think is like only allowing 20%. I mean, for these big companies on these big campuses, they have a problem. So Salesforce, for example, I think is now subletting part of their office space, because it's like this massive tower, where they were expecting like everybody to come in, but now they're only going to have part-time people. So there's definitely like a big dynamic that's going to change, and how offices will work and so on. So for us, like all our desks, our hotel desks, so you like you have to reserve it if you want to use it there and go to the office.

[00:37:40] JM: Oh, that's cool, actually. I like that.

[00:37:42] JN: Yeah.

[00:37:43] JM: Okay. Well, I got to go. I wish we could do this longer. Are you going to be in town anytime soon? Could we do in-person? I got a studio space now. I’m totally serious.

[00:37:51] JN: Maybe next time. I'm planning to start flying into the Bay Area like once a month.

[00:37:56] JM: Oh, awesome.

[00:37:57] JN: Like spending a week here and then going back. Like my job is literally every day kind of talk to people in meetings. Like from the morning until the evenings, like all meetings. And like just doing it over Zoom, it’s like disgusting.

[00:38:10] JM: Yeah, it’s not good.

[00:38:11] JN: Jeff, great chatting with you, man.

[00:38:12] JM: Yeah, it's a real pleasure. I really can't wait to see you. It's such a pleasure talking to you during the pandemic. So, yeah –

[00:38:19] JN: Well, thank you again for having me.

[00:38:21] JM: Okay, anytime. Talk soon, Jad.

[END]