1:53

Alexey

**This week, we'll talk about building an open source data company, and not just building but… It's not the first time we have our guest, Adrian, on this podcast. Before, we spoke about being a data freelancer, so we'll talk about building an open source company as a data freelancer in the past, I guess. The special guest we have today is Adrian. Adrian started working in data quite some time ago – in 2012 for Berlin startups. Then he joined a corporation, and quickly found out that this is not what he likes, so then he decided to freelance.**

**He freelanced for quite some time. We have another interview with Adrian, as I mentioned, where he talks quite a lot about his experience. But today, we invited Adrian again to talk about what happened after freelancing. This is also a question where many people think, “Okay, now I can do freelance, but maybe I don't want to do this forever. What's next?” So today, we'll talk about that. Welcome, again, to our podcast.**

3:00

Adrian

Thank you, Alexey.

3:03

Alexey

**Yeah. The questions for today’s interviews were prepared mostly by Adrian, but also Johanna helped. Johanna always helps. Thanks, Johanna. I see you're here in the chat, so big heart. Thanks for doing that. Let's start.**

# Adrian’s background

3:19

Alexey

**Before we go into our main topic of building an open source data company, let's start with your background. Can you tell us about your career journey so far?**

3:28

Adrian

Sure. I just want to first mention that the questions were actually prepared thanks to Yovid, in case he’s in the audience. He basically asked me these questions last week, and I was able to just use most of them.

3:44

Alexey

**Who?**

3:45

Adrian

A data engineer, Yovid. I'm not sure if he's in the chat right now.

3:49

Alexey

**Okay.**

3:50

Adrian

Anyway, what was the question you asked? Why did I start freelancing, or?

3:55

Alexey

**No, the question was – for those who did not listen to our previous podcast, maybe you can give us an overview of your career journey so far.**

4:03

Adrian

Right. So as you were saying, I started working in data in 2012. I started freelancing some five, six years ago. I've already stopped by now. That was an amazing change of pace from employment, mostly because it gave me a lot of autonomy. It allowed me to really consider what I want to do with my life and with my time. It also put me in the position where I was able to save and invest, which also opened up different chapters and options in life.

I would say, I was on the freelancing path, which I think is entrepreneurial, for about five years’ time, during which I had quite a few learnings about what could be the next step. This is how we got here. I would guess the next logical step for me was taking more risk, actually – looking into how I can invest my time better than just freelancing.

5:15

Alexey

**You were freelancing as a data engineer, right?**

5:20

Adrian

Yes. I mean, I did all kinds of things, because I tried to figure out what's interesting to people and what's interesting to me. One of the things that I was doing as a freelancer was data engineering, of which I would say maybe half of the things were first-time setups – commonly called build and hire, where you build out a data warehouse and hire a team. The other half, I would say, were just generic data engineering projects. When I wasn't doing data engineering, I was doing a little bit of consulting. This was, basically, how you should do your data engineering, more or less, or how you should structure your team.

6:01

Alexey

**For me, freelancing is exchanging time for money. The image I have in my head is – a company needs somebody and then you say, “Okay, I charge 100 euros per hour,” or 1000 per day, or whatever. You say how much you charge and then they say, “Okay, we’ll hire you for three months to do this thing.” But I remember that in our previous interview, you also talked about other things. It was not just that, but other different kinds of freelancing, where sometimes it's more than that.**

**Maybe a company needs to build a data warehouse and you say, “Okay, it’s difficult to do for me alone, but I have my friends, who can also join. Instead of charging you per hour, I will charge you per project.” Then it's more entrepreneurial, right? It's not just “Give me 100 euros and I will just sit there, and then calculate how many hours I spent.” But you say, “This is the problem we have. This is the solution I can provide. This is what I need in exchange.” It's a bit different. You provide a solution to the problem, rather than just selling your time. For me, it was already quite entrepreneurial.**

7:18

Adrian

Exactly. The customer typically already wants to know how much they're roughly going to pay for what they get. They don't actually care about your hourly rate, generally. They care about the final outcome and what it costs them.

# The benefits of freelancing

7:34

Alexey

**Okay. You worked as a freelancer for some time. [Adrian agrees] Although, as we just talked about, there are all these different kinds of freelancing, where you just exchange time for money, where you sell projects, where you can bring your friends to help you, and so on. Why did you want to do something else? What triggered you to change from freelancing? It looked like you were already making a lot of impact. It can be quite satisfying when you don't just exchange time for money, but you also get paid for delivering the project. It's different – it can be pretty fulfilling, I guess.**

8:16

Adrian

It was. I can't complain about freelancing – I really enjoyed it. And I still think it's the, let's say, best lifestyle thing that somebody can do. Compared to employment, or compared to founding, it's probably the role where you have most autonomy. So if you are actually looking to also have a holistic life and invest in other areas, that's a pretty good position to be in.

8:41

Alexey

**Holistic life. That’s like when you can go fishing on a Wednesday.**

8:46

Adrian

Go fishing on a Wednesday, for example. You can decide every day what you're doing more or less. Of course, you need to be civilized and do it within the boundaries of other people working with you. But you do have a lot of autonomy and freedom. I would say I enjoyed that very much. But there is always something in the back of the head that goes, “I want more. I want something different.” Once you do something for a while, at least for me, it starts getting boring. I have to say that, you know, I built a lot of data warehouses and at some point it gets really old. Of course, when you are freelancing, people know you for what you do, so they will offer you more of the same work.

What I was doing in the later stages of my freelancing was also sub-contracting. Like you were describing, a customer has a need, and I didn't have some of the skills to deliver some of those things, but I could subcontract someone to help us to deliver. I did that. This changes your role a little bit from a single contributor, where you have a lot of autonomy, to something more like an agency manager, where you suddenly have a lot more communication and a lot more answering lines. Your autonomy goes down and your revenue goes up, but I would say that happiness also went down for me when starting in that direction. This basically brings it to the question of “What next?” More of the same gets old and if you look into how to improve things, you could go in the agency direction and make it a bigger business. But that doesn't seem fun, personally.

# Having an agency vs freelancing

10:38

Alexey

**In the case of an agency, you would actually have a company, you would hire people, you would negotiate with potential customers on what to deliver, and then people from your agency go and work, right?**

10:51

Adrian

Yeah. It's a different business at that point, because it's not only about you and your personal choices. Now you have employees, and you have some responsibilities around that. You need to make ends meet, because if you have an unpleasant customer, it's easy to fire them when you're a freelancer – you just go to a different project. But when you have a whole team of people and you need to keep them employed, it becomes more difficult. You're also managing the flow of people – it's a completely different job, fundamentally. There is also, let's say, a question of incentive.

When you're personally a freelancer, it's quite easy to just be aligned with your own system of values, whereas when you own an agency, and you have to pay the bills of the people that you've hired, I would say you start to have multiple responsibilities, some of which might conflict directly with your values. You might have to work with a customer for the good of your company, where you might have not chosen to work with this customer otherwise.

12:01

Alexey

**That's one of the potential next steps for a solo freelancer? When you're a solo freelancer, and you want to do something bigger, an agency is one of the options – but with all the pros and cons we discussed. The pros being having more money, and the cons, I guess, are all these things that you mentioned. For some, it might be fine, but not for everyone. Another option would be creating a product, right?**

12:31

Adrian

You could always go and do something more esoteric – that's different from core freelancing. You could go build a product on your own, but that sounds pretty risky. Or you could go down, let's say, the path of creating a company. Because fundamentally, I wanted to build a product. But in order to enable making that happen, you need to align multiple incentives with that. If I just want to build something by myself in isolation, for nobody's benefit, I can do that. But if you want things to end up in front of people, if you want them to be of a high quality, if you want people to help you, it's a different game. That's no longer, let's say, the resource drain – to make something like this happen takes more than the effort of one person. Also on the skills part.

# What let Adrian to switch over from freelancing

13:27

Alexey

**Okay. Usually, you don't just wake up and say, “Okay, now I quit freelancing. Let's try to build a product.” [Adrian agrees] It happens gradually, right? So there was something you saw that you were doing over and over again. Right?**

13:42

Adrian

I was telling you about building data warehouses over and over again – that does get pretty old. I would say that the reason why I didn't enjoy it anymore – it was not really challenging. The technical challenges get simple if you've been a data engineer for a few years and then it's just the people problem. It's kind of the same problem on repeat – educating stakeholders that “total margin” doesn't mean anything, and you need an actual metric, educating stakeholders that “number of customers” means different things to different teams. I can give an example where I was done building a data warehouse, technically, in two weeks, and then we spent two months getting everyone on the same page as to what “the customer” is and which ones they're tracking and which ones they're reporting on. It was literally mind-blowing. What happened for me was, I thought, “Okay, I don't want to do…” The technical challenge for me was easy to solve, but what I realized is that it's not easy for everyone else on the team.

Data engineers are often a bottleneck in organizations. What I was trying to do with my work of building and hiring was to always empower our others to take over. It wasn't a game where I wanted to be the centerpiece of data engineering for the next five years at this company. In order to do that, I was looking for easy ways in which other people can take on the engineering role. Like I said, I think it's a pretty low-complexity role in terms of technical requirements, but it's not necessarily easy to learn and to get there. If you actually have everything available to you, such as boilerplate code, then you could just use it as a data person, and you don't need all the engineering to happen. This is how the product idea actually came to be. Which I guess is more or less what you were asking, right?

15:48

Alexey

**Well, yeah – more or less. I was curious, what exactly you saw that made you think, “Okay, so many companies have this problem. If I put a solution to this problem in a box, instead of selling my services to these companies…” You could say, “Okay, this is the solution and you can actually use this solution on your own. You don't need me to use the solution.”**

16:16

Adrian

Yeah. I can give you some examples. You have a lot of really smart people working in the data field right now and even more smart people not working in the data field and wanting to. So you have all these new-generation Python users that… they're brilliant – they were exposed to programming languages earlier than the rest of us and they've had opportunities and chances that we dream of sometimes that we had ourselves. They're uniquely positioned to be able to solve these kinds of problems, but they need the right tools, because they don't have five years to develop the engineering skills and learnings that it might take for a data engineer.

What I noticed was a few patterns such as, these Python people can easily use Pandas, and they can get some data from an API, they can use pandas and SQL to load some data – fundamentally, the skills are there. It's just about the in-depth engineering knowledge that isn't quite there. There was another common antipattern – people were basically just throwing JSON strings into databases. This happens quite frequently. The reason why people do this is, because they have a problem and this is a solution. Right? No one's just being evil or shitty about code quality. Basically, I thought, “Okay, we need some good dev tooling to help us do what these people are trying to do. They're not doing things in a good way, in terms of engineering. [This tooling needs to] help them basically do it faster, better, harder, stronger.”

17:51

Alexey

**From what I heard from you… I know a bit about the product (the tool) you’re working on. What I heard is –when you have a data warehouse, you don't just have it for the sake of having it, you need to put some data in it. Then you have a bunch of JSON data coming from endpoints or from somewhere. Then, you need to take this JSON data and put it into a data warehouse or some sort of database and the easiest way of doing this is just to take the JSON as-is and put it to Postgres (because in Postgres, you have this JSON field or whatever). I think for all the major data warehouses, you can just put JSON data in there.**

18:41

Adrian

You can ignore it as a complex type, or you can just [audio cuts out] strings. [audio cuts out]

18:45

Alexey

**What people were doing is, naturally – you have this type, you have a bunch of JSON, “Hmm. The easiest way is just take the JSON and put it in the database.” Right? Because not everyone knows that maybe it's not the best solution, but it kind of works all right. So you saw that. You saw that people can actually do that –they can use Python – but this is not the best thing to do for a data warehouse, because it's expensive and you don't want to just put complex objects there because it's more difficult.**

19:18

Adrian

There are many problems – from cost, maintenance, robustness, and so on.

19:23

Alexey

**So you saw that and you thought, “Okay, what if there was a tool that could just easily enable them to do that? Then, instead of having a bunch of JSON fields in the database, they have something proper.” Right?**

19:38

Adrian

Yeah. Basically, databases like type flat tables. For example, JSON doesn't have date/time, so when you load JSON to a database, you're creating a date/time string, kind of. You're just creating strings. Then somebody has to come and say, “This is a timestamp. Let me extract it from the JSON. Let me give it the right type. Now we can actually use it.” So I tried to kind of jump over these steps so we don't have to have a human that is manually guessing what each data type is and unpacking this data and making it clean. Basically, this tool that I was working on facilitates just taking unknown JSON data and putting it in a tabular relational format in the database.

I say “relational” because JSON can have sub-structures, such as lists, within the JSON, which you cannot represent within a simple type. You need to either break it out into a new table, or keep it as some kind of array. Databases like tables, not arrays. This is kind of the idea – it just makes it easy. You have, let's say, declarative ways of loading – I can replace, I can append, I can upsert or merge the data. You also have a lot of tweaks that data engineers care about. I won't really go into a lot of details here about that, but… you put the JSON in the database, but the data engineer will worry about, “Okay, what's the distribution key? What are the primary key performance considerations? Maybe I want the data contract.” Things like that.

21:20

Alexey

**From what I heard – you said that people who are not necessarily experienced data engineers, but know some Python, can just take these JSONs and throw them into the data warehouse. But also, if I'm an experienced data engineer, this is kind of repetitive – I have this bunch of JSON files or JSON data and then I have to parse them and think, “Okay, for this, because it's nested, I need to create this table in this table. This is a one-to-many relationship.” And then you spend a week messing with JSON structures and creating a table structure, and then doing all the mapping.**

**Even if you have experience, it's kind of repetitive. Then you join another company, and then you have the same problem. Then you join another company as a freelancer, and, again, they have the same problem. So it feels super repetitive, right? Because you need to do the same thing over and over again. Even if you do it correctly, as a data engineer, you still end up doing a lot of stuff again, and again.**

22:24

Adrian

Yes, and ultimately, correctly, it's a matter of best guess, right? Because when you're referring to types from a weakly-typed data like JSON, you could be wrong. There's nothing preventing JSON from sending you a number today, and a string tomorrow – JSON internally doesn't have any kind of type consistency between records. You want something that also reduces maintenance, because we can guess what the data is, but we might find out that we are wrong two days later, when the data doesn't load. We don't want to do that. The more data you have, the more you try to avoid this, typically, and you try to curate it upfront, which generates a whole another set of problems. Now you have to talk to stakeholders and people and it takes so much longer.

# The conception of DLT (Growth Full Stack)

23:11

Alexey

**Was it born when you saw this pattern and you wanted to solve this problem for clients faster? That's why you created this tool? Or was it that you realized that there is a pattern, you stopped freelancing, and you focused on the tool?**

23:30

Adrian

It's a little more complex than that. I was already building some kind of data loading tool before – it's called Growth Full Stack. It's something like Fivetran for a specific vertical. There, I had the pleasure of playing with the concept a little bit. Okay, you have the side of what the right way for taking something to people so they can use it is, and then you have the building side of how it could be and how it should be. I would say it took me some time to actually formulate some stronger opinions about how it could be built. Once we started building it, there are multiple layers of abstractions of how you could build something like this – it could be built in a way that engineers love, or it could be built in a way that any Python person can understand.

Usually, to reach these good abstractions – you might have some ideas, but they're not going to be the best. Right? You want to validate them. We did a kind of incremental process, where we first built an engine, and with this engine, we started building pipelines, getting people's feedback, seeing what people could do or couldn't do. Then, we figured out that we need to simplify this and created another layer of abstraction on top, to enable the Python users to just easily use this.

24:53

Alexey

**You said, “Once we started building it,” you used “we”. Who are “we”? Who are you talking about? You and some other people?**

25:04

Adrian

Me and the co-founders, yes. Should I talk about them or?

25:09

Alexey

Yeah, I'm just curious – you were a freelancer, I guess you've worked with other freelancers, but how did you two meet? How did you actually find each other and decide, “Okay, let's focus on solving this problem.”?

25:23

Adrian

So it's a classic story – we met at work. On my last project, the guy that hired me had been working for this company for six years. He had previously founded some companies. And basically, I ended up working with him to build this Growth Full Stack solution that I mentioned. So we actually had one year of working together. We had also worked together on some smaller projects before. I went with him on a consulting and sales trip to Poland, for example, to sell this data engineering solution. I kind of realized, “Okay, I can work with this person. We can communicate well, and this person (I'll just call him Matt, because his name is Matt).”

Matt also had funding experience from before. I, personally, as a freelancer, wouldn't have jumped headfirst into all of this founding chaos without some kind of guidance. So this was a good opportunity. The rest of my team are basically people with whom Matt has founded before. There's Martin, who's our technical genius – who basically very much likes to hack things and figure out the simplest ways (and the most elegant ways or esoteric ways sometimes). There is also Anna, who joined in a more limited capacity, originally, to just help us with operations – registering the company, talking to the lawyers, figuring out all these kinds of things.

# The investment required to start a company

27:13

Alexey

**Before you started getting people on board, Anna and Matt, you wanted to know that this is the right thing to do – before starting a company? How did you decide that this is what you actually want to do? Like, “Let's start a company. Let's start getting money. Let's start hiring people?” Because it's a big investment, right?**

27:39

Adrian

Yeah, it's a big investment and it's a complex decision. I would say, for myself, I looked at it from an entrepreneurial perspective. I thought, “Okay, I'm a freelancer. There's so much that can be done with freelancing. I want to invest more.” As a freelancer, you actually get to earn a lot of money, and I had the opportunity to learn about investments. And when you are investing, there are multiple tiers of risk that you could take. You could buy a house and rent it out and make maybe 1% per year. Or you could buy an apartment, and make maybe 3% per year. Or maybe you invest in the stock market, you take more chances, and you might earn more. Or if you want to really go crazy, you could go into angel investing, which is very high risk and high reward. Sometimes it has a return of maybe 30% per year, statistically. And the next step would be founding, where you go all-in on something that's quite high risk. Of course, it’s also possibly high rewards.

This is, in a way, my way of investing in a way that I was able to. In another way, because I met Matt, and this team, it was kind of the perfect opportunity to go down a path that they were already familiar with, making things much easier to get it right. Also, having worked with them before, I had the understanding that this is a group of people that I can work with. Because probably when founding – when choosing co-founders, one of the most important things is that it's kind of like a family, in the sense that now you're bound together for the next six years or something. Then you will always need to figure out solutions to problems. If you don't, your problems will only grow. So I knew that this is the right team to do it with.

29:43

Alexey

**The word “investing” that I used when asking you that question, I meant more like time investing. But that’s interesting – it's interesting that you came from that angle. You can invest in a house, in an apartment, as an angel investor, or just starting a company.**

29:57

Adrian

I could have kept freelancing and earning money and [audio cuts out] so… [audio cuts out]

30:03

Alexey

**This is where I was going with that: if you work as a freelancer, it's probably a more natural thing to do as a next step – one thing is agency and then another thing is building the product. You don’t necessarily see it as an investment, but more like, “Okay, this is the next thing I will do.” But also, you need to kind of push money there… well not kind of, but you actually have to put money there. You need to eat something, right? Then, probably getting the first version… and this is something we should talk about, right? How did you…?**

**All of us humans need to eat, we need to live somewhere – it's the modern world, it costs money. So how did you solve this problem of finding something to eat and a place to sleep while being…? Because you bootstrapped at the beginning, right? You had your own capital, and you worked off of that capital, or how did it happen?**

31:08

Adrian

Basically, when you found a company, you need to start it with some capital. Then, there's your cost of living, which I would say is a moderate concern, in the sense that it's one of the costs that you will incur. Then there is the cost of company operation, because ultimately, you will want to do something with that company, and just keeping it with some money in the bank and not spending that money doesn't actually get you anywhere. What you want to do is invest and have access to build things. So you also want to get some money into the company. What we did was literally for one year, we didn't actually have a job or a salary. We just lived on savings. One of the advantages was that the cost of living in Berlin is not too high. It's manageable, if you manage to save up a little bit.

32:04

Alexey

**Every month, when I pay my…**

32:06

Adrian

It gets worse. [chuckles]

32:09

Alexey

**Yeah. I'm almost crying at how expensive… expensive compared to, I don't know, eight years ago. Of course, it's not expensive compared to New York.**

32:18

Adrian

When we were building this company, and costs kept going up like crazy, I was like, “I had a plan and I’m definitely not able to stick to it.” So it was definitely a little bit stressful. Luckily, there was still work to be found in the markets. So, at least we were able to earn money into the company a little bit to fund our operations. We basically, as I told you, first we built an engine. We use this engine with design partners, which also paid us for the work. We were able to get some feedback and also a little bit of funding for our company.

# Growth through the provision of services

33:02

Alexey

**So you were solving the problems they had, and building the tool at the same time, right? Because you were providing services, they were paying you. It was kind of like freelancing, in a way, right? Or was it more like, as a company, you provided services to them?**

33:19

Adrian

Exactly. It was kind of like freelancing in the sense that I was actually the only one directly involved in these projects, mostly. The difference, I would say, for me, is that somebody else was issuing the invoice for some other entity, mostly. But what was happening was – we got to actually test our tool with real data and we also had some really good learnings, which potentially couldn't be worth more in the prospect of a company building, with the little money that we got from these consulting teams.

33:55

Alexey

**But I guess this is not… You cannot do it that much. There is some money – you earn some money from this activity – when you consult, and then you get money, and you keep money in the company to fund yourself (to pay your and other people’s salary, pay for the office) but that is probably not enough, right? So you need to get more money.**

34:20

Adrian

What we did in the first year, actually – we squatted offices. As you know, the financial situation was changing, so a lot of companies basically – investors invested in them, let's say, 40X their yearly revenue value, where 20x would be normal. So when the “mini-crash,” let’s call it, came, lots of investors basically either halved the investment or doubled the goal to get back on track. And what this did was, companies were forced to let go of half of their staff. Unfortunately for them, but fortunately for us, there were lots of, let's say, empty rooms around Berlin offices at the time. We were able to use these rooms without paying for rent, which was super [cross-talk]

35:12

Alexey

**That’s why you said “squatted”?**

35:14

Adrian

Yes, exactly. We also didn't get…

35:18

Alexey

**But they knew that you were sitting there, right.**

35:20

Adrian

Sorry?

35:21

Alexey

**The company knew that you were sitting in the offices for free.**

35:26

Adrian

Yeah, yeah. It wasn't like Occupy Berlin or anything. We also didn't pay ourselves a salary. Because that wouldn't have been realistically possible. The pressure did increase at some point and we had to raise a pre-seed round. This was happening around the time when we were also creating a workshop for validation. I think you remember about that one because I was asking you if I can recruit some testers from your Slack group.

# Growth through teaching (product market fit)

35:59

Alexey

**Did you recruit anyone?**

36:00

Adrian

Yes. I can tell you. Basically, what happened was – we had this engine, and we wanted to figure out how to make a better interface for this engine that any Python user could use. Around that time, we were actually already raising the pre-seed round. It was quite chaotic, but we created this three-day workshop, “How to build a pipeline in six hours (two hours a day),” and we had some 60 people join us. To our amazement, they were all able to build an incremental pipeline. Back then, it was Twitter API, and now there is no more Twitter and there's no more free API either. Different times. But then, we realized, “Okay, we have success with this interface. This is what people can use and are willing to use and learn. It's a shallow-enough learning curve that people will just use it.”

36:55

Alexey

**So I guess people who came to the workshop, they didn’t necessarily come to give you feedback on the tool, but more to learn how to actually build pipelines, right? [Adrian agrees] You showed them how to do this, but you also learned from how they use it. Can you tell us more about this learning experience? How did you actually design the workshop in such a way that it was helpful for you and not just for the attendees?**

37:28

Adrian

Yeah. That's quite clever, actually. We had some help in designing this way of measuring. But one of the challenges was – when you're teaching, it's also hard to get live feedback from people, so I don't know how you manage it. But we figured that we actually need two people when teaching this course – one is teaching and the other is actually watching what the people are doing and helping. We have checkpoints. For every day, we had something like 20 checkpoints. We split the two hours into 10 segments and we were asking people to react on Slack to the checkpoint message to say, “Okay, if everyone has managed to do this step, please give an emoji to this Slack message.” This way we could actually see how many people have done it.

38:23

Alexey

**And how much time it took for each segment, right?**

38:26

Adrian

We weren't so concerned about time. We were concerned if somebody doesn't get it and if they cannot do it. Ultimately, we looked for the completion rate.

38:38

Alexey

**But then the role for the person – one is teaching, and the other is observing and helping – the one that’s helping, their task was to see the patterns, right? “X percent of students had problems with that checkpoint (with completing that part of the task).” Right?**

38:58

Adrian

They were basically moderating. I was teaching and they were saying, “Okay, enough people have finished this step. Let's move on to the next section.”

39:11

Alexey

**So if somebody's late, “I'm sorry.” Right?**

39:14

Adrian

Yeah. I mean, we waited. Some people asked questions, and they were also able to get live help, if they had some errors. One important aspect (speaking of errors) was actually preparing the workshop in an environment that's the same for everyone. We basically created a Code Spaces environment where everything was the same. So we had a pre–

39:39

Alexey

**Oh, you had Code Spaces? We should use that for our courses, too. Usually, we just say “Okay, just run an instance. This is how you configure stuff on that instance.” Then some of the students came up with this idea of, “Hmm… We can use Code Spaces.” They took initiative. So what you did was realized, “Okay, this thing is free, (or it costs something, but not much) and everyone has the same environment.” That's smart. The task was to get some JSON data from Twitter and build a warehouse from this data. Right?**

40:19

Adrian

It was basically to get the data from Twitter. The people that we were teaching were just Python users, so many had not actually done a web request. So it was also teaching them to extract the data, with authentication, pagination, concerns – so it was teaching best practices of how to do this, and also using the tool itself.

40:43

Alexey

**We didn't mention the name of the tool, right? Did we?**

40:45

Adrian

We didn’t.

40:48

Alexey

**So, what's the tool? [chuckles]**

40:50

Adrian

Yes. It's pretty simple. It's called Data Load Tool (DLT for short). I often like to tell people, “Don't think of it as a data loading tool, think of it as a pipeline building tool.” And the reason for this is because it's a developer tool made for developers to build pipelines easily. While it does load data, it does so because you built the pipeline.

# Moving on to creating docs

41:16

Alexey

**That workshop that we talked about – I think DLT already existed when the workshop happened, right?**

41:23

Adrian

Yes. Basically, at that point, we had just created this simple interface on top of the engine. But what is DLT, ultimately? A product is kind of a moving target. What we didn't have at the time were docs. You can imagine that without docs our product is unusable. I would say docs are just as much part of the product as the code itself. At that point, it was more like a research phase – it was very early.

41:55

Alexey

**As a result of that workshop… What did you actually learn from this workshop? Was it…**

42:01

Adrian

We learned less than we were hoping to. We were expecting that people would have problems with various parts of the workshop and it turned out that there were no problems. What we did learn was that this was a good abstraction for people to use, that it was really easy to understand, and the next step was to create docs that allow people to actually understand what’s happening and use it.

42:31

Alexey

**I see – because creating docs is a big investment. Because once you have the docs, and you change something in the tool, you need to redo all the docs. What you wanted to check was if the tool (the abstractions) you came up with were good enough so you could start building the docs. [Adrian agrees] How would you know that you need to do it this way?**

43:01

Adrian

It's not only about knowing. It's also… You're limited in bandwidth. There's only so much you can do. I can tell you that it probably took us three months before our docs were actually at a level where people were able to use them and another three months before they were the level where people were saying, “Hey, your docs are pretty good!” Yeah, we had quite a bit of negative feedback.

43:28

Alexey

**The purpose of this workshop was to help you… You had limited resources and the purpose of the workshop was to help you figure out what to do next.**

43:38

Adrian

Product market fit.

43:40

Alexey

**Product market fit. So either you focus on making the engine better right now because there were some checkpoints where many people struggled, or, as it turned out, there were no problems with the actual code. For you, this was a good signal that, “Okay, now it's time to invest in docs.” Right?**

44:00

Adrian

Exactly. The way you can think about it is: before you have product market fit – which I guess I will also have to explain what that is – usually, you keep going towards product market fit. Product market fit is basically a point where your product fits the needs of the market. You can generally tell by increasing adoption and people really wanting to use it. They say that one way to determine if you have product market fit is to take the solution away and see if anyone cries. If nobody cries, you don't have product market fit.

44:34

Alexey

**Interesting. How would you take the tool away? “Now, let’s try to do the same thing, but without the tool.” And you see how people react. Right? “Oh, no, I don't want to do that. Can you give it back?”**

44:50

Adrian

For example, one way is – you don't need to ask. You can see that it's used at the core of things. For example, we have an early adopter that decided to run the entire organization on DLT – the entire data stack – so if you take it away, they're gonna have to figure out how to do something else, but it would be a big pain, because it currently solves a lot of problems.

45:17

Alexey

**That company that you mentioned, they knew you were an early-stage startup, and that you were just experimenting with product market fit? Did they? Yet, they decided to put DLT at the core of their processes. It's a bit risky, right?**

45:35

Adrian

It depends on who you are and how you perceive risk, right? If you're a software engineer and you can analyze the code base that is open source, and you decide that this is something that looks good to you, that meets your criteria, then it's easy to make a decision. Because if something goes wrong, you can maintain this and you can keep using this, it's not just [audio cuts out].

45:59

Alexey

**This thing is easier than writing from scratch, right? You can just keep a copy of the code in your internal GitLab, whatever, and it's still easier than building a similar thing from scratch. Engineers saw value in that, because they can just open this thing and see, “Okay. Makes sense.” And then just use it.**

46:23

Adrian

But if you're the kind of person who's rather a tool user – if you use a tool like Segment, for example – where you expect that everything is done for you and you just pay for it, then this would be too early-stage for such a market.

46:39

Alexey

**So, for open source it’s different, because engineers adopt the tool and maybe the management doesn't care – as long as the problem is solved, it’s okay.**

46:51

Adrian

Yes. Also, it depends on the engineer. Because if the engineer can handle the code base on their own, then they don't have a problem. But if you expect something to happen in the future to the open source project, then it might not be a good bet.

# Adrian’s current role

47:06

Alexey

**What do you actually do these days? At the beginning… Because you had a co-founder, you had other people – so what did you do at the beginning and how did your role change over time?**

47:18

Adrian

My learning about roles in founding a company is that it's very different compared to what you think. While you might be able to work to your strengths in some areas, there will also be a lot of things that need to be done in the company that are nobody’s strength, but someone will have to do them – and it's going to be you, because you're the last line of defense.

47:46

Alexey

**But maybe your co-founder can do this. [chuckles]**

47:48

Adrian

Yeah, or my co-founders, of course. [audio cuts out]

47:51

Alexey

**Somebody – either you or him. Right?**

47:56

Adrian

Exactly. Basically, what this means is that you need to figure out what needs to happen next – figure out some kind of way to do it – and then try to get help to do more of it in a better way, if that pays off, kind of. So I'm doing a lot of things that are not in my strengths. I’m kind of inventing what we should be doing. But I'm not trying to reinvent any flat tires. I'm taking lots of cues from other people in the industry. To be specific about what I'm doing right now – I'm actually heading the go-to-market strategy for our library, which means communicating… You could call it marketing, in a way that helps the end-user understand about your product and that is aligned with the strategy. Specifically, our strategy is to go for bottom-up adoption. We don't want to be the solution that your non-Data Manager is buying, because they think it builds them a warehouse and bakes them a cake. We want to be that actual developer tool that you will come across and you will go like, “Wow, this is so much better than doing all this manual junk myself.”

49:13

Alexey

**You, as a data engineer, (or former data engineer… I don't know if you can ever be a former data engineer) but as somebody who has done this many times, who can speak the same language as other data engineers – you can explain what this thing is doing, right? This makes you a perfect fit for this position. Right?**

49:36

Adrian

Yes. Things like identifying the use cases for specific personas. For example, how do you reach audiences? You could go to where these audiences hang out, for example, on Reddit, or a data engineering Subreddit, or on your Slack group, or you could go to other tool groups that these people use. Right? If you go to this other tool group, you want to figure out, “Okay, do we have a use case that this audience is interested in?” Then you kind of need to figure out what they're doing, what they like, how they think, what problems they have, and then offer them a solution for that so they can relate to the content and maybe try it.

# Strategic partnerships and community growth through DocDB

50:24

Alexey

**Maybe a good example – I recently came to your office, you hosted a meetup with DocDB and I guess this is a good example of other tool groups. You can just hang out in the DocDB Slack and see what kind of problems people have, and see if DLT can solve some of these problems, right? And if it can't, then what do you actually do? Do you say, “Hey, have a look at DLT?” Or do you just take note and then see if you can improve these cases?**

50:53

Adrian

I would say that DocDB is a bit of a special case, because they have enabled us to do something, not slightly better, but zero to one. Specifically, because we're a library that runs in a notebook, and so are they, this means that we have the opportunity to run together in places where data pipelines and data engineers previously didn't go. This means we can, for example, create a simple demo on a Colab that just runs. It enables lots of easy testing and easy adoption for us. Also for development, right? Because DLT will generate schemas before the database. So if it's DocDB or something else, like BigQuery, DLT doesn't care – you will have the same schema. You can literally go for development between DocDB and something else. That event – we didn't do any content there, we just posted it. But we have a DocDB destination, we have another doc destination, and we are kind of their recommended solution for loading, which is helping.

52:01

Alexey

**I hope that DocDB users know that they recommend using DLT.**

52:10

Adrian

Come again? I didn't get the question.

52:12

Alexey

**You said that this is the recommended solution. It means that somewhere in the documents, they mention something like “If you have a specific use case (this specific problem) use DLT.”**

52:21

Adrian

Yes. I think it is there somewhere, maybe in… I don't know if we're actually mentioned in the documentation. But they did distribute us to their audience in the newsletter, for example. We have some demos we did together. [cross-talk]

52:41

Alexey

**How did you convince them to do this? This is interesting. What you have is a mutually beneficial… partnership? If I may say it this way. They help you because you can run both things locally, so you don't need to set up BigQuery, or whatever. And then for them, you solve some of their problems – then they can easily see the benefits of DocDB. So how did you find this partnership? How did you actually… not convince them, but how did they end up doing this mention? Did you ask them, “Hey, can you feature us in the newsletter?” Or did they want to do this themselves? Or how did it work?**

53:24

Adrian

You can think about this like dating? How do you move on from dating to marriage? It's a process. It's not a one-off point, right? I would say it was incremental. Partly, we just added DocDB because it was beneficial for us, so we added it as a destination. We started using it in our demos. Then we came to them and told them “Hey, guys. Look. It's a super useful solution for us and we also align with your product principles. One of them is being anti-platform – just being a library that doesn't plan to take over your entire stack.” This worked very well also with DocDB because what this means, if you're not trying to take over things, it means you're trying to integrate into things. It means that you can have healthy partnerships with the rest of the ecosystem, which DocDB also does very well.

# Plans for the future of DLT

54:25

Alexey

**Okay. So what are your plans for the future?**

54:29

Adrian

We are almost done closing the fundraising round. This will open a new chapter in the life of our company and that chapter will include working on a paid solution. I was telling you about the go-to-market fit. What you want to do before you take money is find product market fit. Because if you don't do that, investors don't care about your product market fit – they care about outgrowth.

55:02

Alexey

**They give you money because they want to get more money in return. If they don't see how their money will multiply, then they will not give you the money, right?**

55:10

Adrian

Yes, it's very hard to raise money for just research, right? Basically, we have got a product market fit with our library. Now we're working towards a paid solution. That paid solution would be something complimentary. It wouldn't limit the library in any way, it would rather add to it. Right now, we're basically working on user research to better figure out what the solution could be. We have, of course, some strong ideas, but we want to validate them before we just go out and build things. There are some other, let's say, open problems in the library space. One of them is taking contributions – this is a hard problem.

Basically, a community scales differently than a company. There can be a lot more people in the community, but if you are open to contributions, and they have to go through some company process, you end up being a bottleneck. This wouldn't work, basically. It would put a lot of burden on us that we wouldn't be able to take at the moment. How to solve this problem? Maybe LLMs will be involved in the future. The other problem that could be solved for a library would be – a lot of people just want sources. They don't want to build a pipeline from scratch. It's actually possible to generate many of these sources. But it's, again, not a very easy problem – not for everything. It's also raising questions of utility, maintenance, distribution, and so on. So maybe [cross-talk]

56:51

Alexey

**By that you mean – let's say there's the Twitter API (well, it existed)… Let’s say GitHub API. Then what we say is, “You select GitHub as a source somewhere, you select destination DocDB.” And then that's kind of it, right? You don't do anything.**

57:10

Adrian

Yeah. We've done several experiments in this direction. One of them is – there's an open API standard. You might have heard of Swagger, for example – Swagger tools for API's. Basically, in this open API specification, you have almost all the information you need to wrap an API. You can generate the entire pipeline code from the specification. We actually have them on our website, where our CTO does that from the Pokemon pipeline. And the generator is quite smart. It's only Python rule-based, there's no LLM involved. But it will do things like, for example, understand that, “this is a list resource, and this is a detail resource, and you first need to list, and then get the detail for this entity,” or something like that.

# DLT vs Airbyte vs Fivetran

57:56

Alexey

**So it puts you in competition with tools like Airbyte, right?**

58:00

Adrian

Yeah, I would say they're a distant competitor.

58:05

Alexey

**It’s what they do – they have a bunch of sources, a bunch of destinations, and then they connect them.**

58:11

Adrian

We don't really want to go… Airbyte is a platform. We’ll never be a platform in that way. Even if we do offer some kind of orchestration, that is not our selling point. We don't want to be another Fivetran. Airbyte, currently, is kind of trying to be another Fivetran. There is a question of product market fit. Who's going to be building and maintaining these pipelines? I would say we don't really compete with Airbyte in that way, because their builder is a UI-focused person. On the programming side, it's really hard to build with Airbyte. We also don't want to put this source building necessarily on the community, if not needed. There are multiple ways in which you can do this – I was telling you about the open API standard, but there's also LLMs used for generating. Here, we're actually uniquely positioned as well because if you use GPT with DLT docs, you can pretty easily get a pipeline just by asking for it. This is possible because it's a library. If this was some kind of monolithic application, it would be much harder.

# Adrian’s resource recommendations

59:29

Alexey

**Do you have time for one more question? [Adrian agrees] Can you recommend any resource, book, or course, or something to our listeners about this topic? If somebody wants to start what you did – create a product, (an open source product) where can they learn?**

59:50

Adrian

So unfortunately, there are no major simple resources that give you everything you need, but I can recommend reading about go-to-market and product market fit. I read this book called From Survival to Thrival, the Enterprise Product Market Fit, which describes how you can go from a startup to building an enterprise-ready product, kind of. For me as a data engineer, I never thought about things that way. It was just a very big eye-opener for me to understand that these topics exist, and I should consider them. I cannot necessarily recommend the book as being The authority, but it's definitely good to educate yourself on what is out there.

60:41

Alexey

**From Survival to Thrival, right?**

60:43

Adrian

Yes. This is like, “Something for Dummies.” It's a series. This one is about product market fit.

60:56

Alexey

**Okay. That's all we have time for today. We are a bit… We took three more minutes than we should have. Thanks a lot for joining us today and sharing your experience. I'm really curious. I think the last time we had an interview was two years ago – maybe slightly less. We should definitely meet again, maybe in a year and a half or two, and see what changed. [Adrian agrees] Yeah, that would be pretty interesting. Okay. So thanks again for joining us today. And thanks, everyone, for joining us today, too. Have a great week ahead.**

61:32

Adrian

Thank you. You too, Aleksey. See you on the podcast in a couple of years.

61:36

Alexey

**[chuckles] Yeah, maybe earlier.**