EPISODE 1408

[INTRODUCTION]

[00:00:00] KP: When businesses share a common need, such as payroll, commercial offerings can compete for market share with software solutions that easily adapt to a variety of businesses. Not all tasks can be easily commoditized or standardized. Take content moderation as an example, every site that accepts user generated content is likely to have a unique and nuanced process to match their internal definitions of allowable content, and procedures for what to do with non-allowed content.

Many organizations create custom software solutions for their unique tasks. When they're central to the core mission, that's probably the right choice. When they're not, such projects are often a distraction for an engineering team that is already stretched too thin. That's where solutions like Flowdash come in. Flowdash is a low code, no code offering that enables businesses to rapidly develop custom solutions for tasks such as content moderation, and others. In this episode, I interview Nick Gervasi, co-founder and CTO of Flowdash.

[INTERVIEW]

[00:01:05] KP: Nick, welcome to Software Engineering Daily.

[00:01:08] NG: Thanks for having me.

[00:01:09] KP: Before we get into Flowdash, can you tell me a little bit about your history in software engineering?

[00:01:14] NG: Absolutely. So, I think I first started my journey into software when I was in elementary school and my father showed me a basic interpreter, was Quick Basic at the time. And he wrote a couple of digits on the screen and added them together. And I was just in shocked by how quickly the computer gave me the answer. I said, “Well, what if we use bigger numbers? What if we use bigger numbers?” And I was just like, amazed by what the computer could do. And I think that just like jump started my journey into software.

So, I've always been a self-taught programmer. I started with Quick Basic and MS DOS, moved on to Visual Basic, I loved video games. So, I taught myself C, C++, Direct X, OpenGL. I got into web programming a little later in my career. And yeah, have used several frameworks in production like ASP dotnet, most recently, Ruby on Rails for the past eight years.

[00:02:03] KP: Let's get straightaway into Flowdash. For listeners who don't know about the product, can you tell us what it is?

[00:02:08] NG: Absolutely. So, Flowdash is an ultra-customizable task management platform that's built specifically for repetitive manual tasks. In our experience working at startups and talking to a lot of startup founders and employees, we all know that engineering resources are often tight. And the engineering resources you do have, you want to spend on external facing product. That being said, there's always a lot of internal processes, internal tools, manual work that your ops team, your support team, your customer success team works through. It's either things you haven't automated yet, it's things that can't be automated, it's things that you want user intuition, or human intuition as part of the the loop.

So, we built Flowdash to help engineering teams build those kinds of tools faster for everything from doing KYC and AML, of New FinTech accounts to things like claims processing and education for health insurance companies. Just helping make it easier to build those manual workflows with less code.

[00:03:07] KP: So, is Flowdash a no code or a low code? Where do you see it amongst the popular buzzwords of the time?

[00:03:13] NG: That's a great question. I put it somewhere between no code and low code. I think it's a bit of a spectrum. And the reason I say that is, I think a lot of the tools in the market are either targeting helping making developers faster at their jobs, which are more of the low code variety, or enabling people who otherwise would not have been able to build applications to do so. And that’s kind of the no code bucket. The reason I put us a little bit in both is we're actually acknowledged that for a lot of these tools, there's two parties involved, both equally important. There are engineers that have to develop and maintain these tools and connect them

with upstream, downstream systems, databases, API's, et cetera. But also, the ops teams that use them every day and the ops managers that manage those teams.

What we wanted to enable with Flowdash was to allow it and make it easy for engineers to connect these workflows with their systems. But to also enable ops teams to tweak those workflows, add things like approval processes, or add escalations or SLAs, or I want to be alerted on Slack if something's been stuck in a particular stage for two days. Those are the kinds of changes that we enable ops teams to make with no code in our platform, while a lot of the integration with downstream systems and API's are more developer focused and as a result are a bit more of the low code variety.

[00:04:29] KP: So, how are users typically interacting with Flowdash?

[00:04:35] NG: Well, I'd say even the word user is a bit nuanced. We typically have two primary users in the product. We have developers who will be the ones generally to make the purchasing decision for Flowdash at their organization. Generally, it's replacing the need for them to build something bespoke from the ground up. So, the value there is that they can spend a ton less time implementing Flowdash then building something brand new.

The way they'll interface with it is setting it up, connecting it with their database and other systems, potentially building a few workflows as proof of concept. And then generally, they won't use it day to day. They now freed up a tons of dev resources to focus on the core product. Operations team is the one that would use it every day, say it's a FinTech company, and they're using the product to flag suspicious transactions that require additional review, an ops person would be alerted that one of these transactions came in would jump into Flowdash, see kind of some of the parameters of the transaction information about the customer, maybe past purchases, and ultimately make a decision whether to approve, reject the transaction, or maybe reach out to the customer for more information.

[00:05:40] KP: And is this handled transactionally? Or how do things fit together from like, an operational perspective? I guess, in terms of like workflows, so what is that unit of work? How does it get assigned? And does it have status and get complete? What's just the general flow of one of these tasks?

[00:05:56] NG: Yeah, that's a great question. So, I mean, the unit of work really depends on the specific use case and specific company and industry. And that's what's really powerful about the platform is its its general purpose, and you can use it for everything from healthcare to FinTech to insurance use cases. We even have a video production company using it to manage their production pipeline. So, unit of work typically gets into one of our queues. You can either push via API, we can pull from a database, you can connect to Zapier and create tasks on various triggers. And then once it's in the workflow, then someone in Flowdash will jump on, they'll either self-assign, or we have some auto assignment features. We have some round robin assignment features kind of depending on the scale of the team, but it's all customizable. And then from there, they can jump into a particular task, review some of the information. There's commenting and notifications, so they can collaborate with other folks on the team. And ultimately, they're moving it towards resolution. Resolution just means something different depending on each use case.

[00:06:58] KP: So, in putting something like this together, we've talked about engineers building the flows, but I'm also hearing a lot of business requirements, or maybe specific things to my company that I need to consider in building one. What's the typical collaboration like for a team putting a flow together?

[00:07:14] NG: Yeah, so maybe a concrete example would help. Say, for example, we took that that suspicious transaction example, I gave earlier for, we're building a FinTech product, we're allowing payments to be made between various parties. Occasionally, a payment comes up that just is above what we're used to seeing. Maybe it's over a certain threshold. So, what we might do as a developer is flag it as blocked in our core system. We actually won't make that payment. But we can also simultaneously post task over to Flowdash to have someone from the risk team manually review that payment before it goes through.

Now, at that point, it's kind of in the ops world. The way that the risk team wants to structure their process, the way they want to assign things across people, the way they want to escalate things, maybe even escalate to a higher level of risk person, if it's over a certain threshold, or looks riskier than other things. All of that can be conditionally defined in Flowdash without any code, which is what makes it really powerful. But ultimately, kind of the point where that loop

gets closed is when they decide, yes, I want to let this payment through or, no, I want to block that payment. That is where the developer would configure, say, a webhook URL back to their core app that finally flags the payment as, yes, let it through, or no, block it, which might trigger other things downstream from there.

[00:08:31] KP: And do companies need to consider change management? It sounds like it's pretty easy to update something in your system. That's not always the case in a typical enterprise software stack.

[00:08:41] NG: Yeah, change management is one of those challenging things, especially I think, in a lot of low code and no code tools. I'd say right now, it isn't really easy for anyone with the appropriate permissions in the system to make those changes. We've built a very robust permissioning system so that you can have that level of granularity of who's allowed to modify things and what parts they can change. We do have a lot more on the roadmap around being able to have proper version control around the workflows, having a paper trail of who changed those things in the configuration, as well as being able to support multiple environments, like for example, a staging environment and a production environment so you can test things before they get rolled out for everyone.

[00:09:24] KP: And when you go to do environments and rollouts like that, are there any best practices people are following in terms of like, do you fit into the typical release cycles? Or is this something done in like a non-sprint fashion that we're entering with a new low code tools available?

[00:09:39] NG: Sorry, just for that question, are you referring to our own change management, like a product features within Flowdash? Or do you mean that the rollout of new workflow changes that our customers are making for their teams?

[00:09:50] KP: Oh, both are interesting, but I meant the second about how your customers are using it.

[00:09:55] NG: Yeah, that's a great question. I mean, right now, a lot of the changes are immediate. I think I think what I meant by adding version control environments are things that

we have on the roadmap to support. I don't think we'd be mandating any specific best practices. I think it is helpful to have, especially when there's integrations with downstream systems, it's helpful to have effectively like an entire clone of your production stack in a staging environment. This is just independent of Flowdash or anything, really. It's just helpful to have a replica of what would be happening in production, to really just test things end to end. There are only so much that I found in my experience, you can do in a local environment, or even with containers. It's just always best to kind of have as closest simulation to production before rolling out major changes.

[00:10:43] KP: And I guess, as a consumer, I shouldn't really care what your internal tech stack is like. But since we touched on it, I'm curious to know a little bit about how you guys do engineering on the inside?

[00:10:53] NG: Yeah, absolutely. So, the main technology we use on the backend is Ruby on Rails. My co-founder and I have used Ruby and Rails for over eight years. So, I think one of the primary kind of decision factors when we're trying to choose our technology would be building Flowdash on top of early on was just familiarity of the founders with the technology. I think that's a really strong argument for an early stage startup, because you want to focus so much of your energy as much as possible on like, what makes your particular business or product unique. We didn't spend a whole lot of time evaluating what back end technology would be best suited for what we were building, as long as what we were choosing would fit the bill and rails absolutely does. That allowed us to move really, really quickly upfront because we weren't, you know, fighting the framework and trying to learn it, we were actually focusing all of our attention on our customers and, and building things that they need. On the front end, we're React. We use Apollo for state management, and our API layer across the back end. And front end is, as a result, Graph QL.

[00:11:56] KP: I'd love to zoom in a little bit more on some of the product features. If we think of an analogy of like a painter who's got a palette with colors on it, what are my colors when it comes to working on Flowdash?

[00:12:08] NG: Yeah, so I think there's a few core primitives when building something in Flowdash. The core app or unit of workflow is called a workflow. Within a workflow, things move

across different stages, those are kind of like the statuses or since it's an engineering podcast, you could think of it like a state machine. So, they are the states that a particular unit of work might be in. We have another concept called actions. And action is something that a human performs in the product that is analogous to a transition if you're talking in the finite state machine nomenclature. So, you define in our Flow Builder, the various stages, those stages might be, it's under review, it's been escalated, it's late, it's been approved, it's on hold, it's rejected. And then you define the actions which are maybe escalate to a manager or approve, and those are using our Flow Builder, you can visually string those together.

It's an entirely graphical editor and that helps you not only define what the users are allowed to perform at various stages, but also helps you and give you a visual representation of what the process is end to end, which acts as the documentation for it going forward within an action. So, there's obviously being able to transition it across stages. But what really makes Flowdash powerful is being able to connect that with other systems and those pieces of automation we call steps. So, a step might be send a Slack message, or call this API, or email a form to the customer to fill out. And so, you can kind of string, those are kind of some of the core primitives in Flowdash stages actions, we have automations, which are kind of like what if this, then that style, similar to something like Zapier and forms is another as well. So, there's a whole lot that goes into building these products, these internal tools and workflows. And yeah, we just love hearing more feedback from our customers and continuing to iterate and add new features to the platform.

[00:14:01] KP: So, it sounds like you own the management of the state then and if I want to query, what's the current state of this particular flow, I can get that through your API. Is there anything else you're persisting or does all the data and ancillary metadata gets stored somewhere else?

[00:14:16] NG: So, we're persisting the state, as you mentioned, as well as any ancillary metadata about that state. If you have customer data or transaction data, whatnot, if there's comments that are being made by the team on it, all that information is persisted. If there's assignments across individuals effectively, there's also an audit log of everything that's been done to every task or unit of work in the system. All of that is being persisted on our side. For many use cases, though, wanting to know, have a single source of truth of your data is

important. So, I think for some of our customers, Flowdash acts as that single source of truth for ops workflows. In other cases, they might connect directly to their Postgres or MySQL database, which is something that Flowdash supports, in which case that would act as the source of truth and we're kind of a layer on top of it.

[00:15:05] KP: So, we've got Postgres and MySQL, can you go through some of the other popular integrations?

[00:15:10] NG: Yeah, sure. I mean, we've got a ton that are being requested all the time. We've only built a few so far and we've kind of leaned on our Zapier integration for a lot of the others. But so far, some of the first party integrations include front, we have Slack, we have Twilio for sending SMS, we have SendGrid, coming this month as well, and Gmail shortly after that. Slack is a big one. I don't know if I mentioned that. But yeah, there's there's about a handful, and for everything else, a lot of our customers have been using our Zapier integration to trigger other actions and other products.

[00:15:46] KP: And what type of professional is typically bringing your product into the organization?

[00:15:50] NG: Yeah, so we kind of see a different kind of depends on the company. In some cases, it is a technical leader that discovers the product. And for them, they see it as a means to spend less engineering time building internal tools. And that's really helpful because, again, I mentioned tons of engineering teams, engineering resources are always limited, you want to focus those on your core product. And so, they see it as a means to spend less engineering time on these tools and enable the ops team to add features on their own without having to kind of make those requests engineering and wait months and months.

Other times, it might be the operations team who will discover the tool, recognize that, “Hey, this will finally help me stop getting waiting months and months and months for my feature request to get serviced by the answer. I can start doing a lot of these on my own.” So, they'll build a case internally and try to get by. And generally, since it does touch the technology stack, since it's usually integrated with other core systems and databases, we've seen that the final

purchasing decision is often within engineering or product. But really kind of the case, we've seen the case being built from different people within the organization.

[00:17:01] KP: Well, this is almost an unfair question on an audio podcast. But since we are talking about a WYSIWYG sort of GUI, can you describe a little bit about the experience? Maybe by analogy, what's it like to actually build one of these with my mouse?

[00:17:15] NG: Yeah, for sure. I mean, what when you start in a new workflow, we kind of kick you off in a very linear workflow that moves something from like not started to in progress to done. Three stages, two transitions between them, it's a straight line. So, if you were to log in, you would see a list of tasks. We start you off with three, and more just to illustrate kind of like what the interface is like. If you click the Flow Builder tab that's in our user interface, that's where you would see the graphical representation of the flow. And what you would see visually is five boxes drawn vertically on the screen, the first being not started, the second being moved to in progress, the third being the in progress stage, and the fourth being moved to done. And then the fifth being done.

What happens is, you can click any one of those, and that'll open a Properties panel where you can rename the stage, you can add a checklist that you want people to work through in order to move past that stage. If the thing you clicked was an action, then that's where you can add some of those steps that we talked about, like sending a Slack message calling an API, et cetera. And you can also drag additional stages and actions onto the canvas. Say we wanted to add a reject action, and then I want to be able to call that from in progress. Well, I would add the reject action onto the canvas and simply click my mouse under review or in progress, rather stage, and dragging that line is what tells the platform, when a task is in progress, you are allowed to reject it. That is one of the valid transitions from this point. So, you can visually string these things together and configure them in that interface. I hope that was my best attempt at describing a user interface on an audio podcast.

[00:18:58] KP: No well done, and I think there's a great video on the site, people can go check out to learn more and get the visual of it. Well, I'm wondering if next we could zoom in on one of the use cases that caught my eye in particular. And that's the human in the loop image labeling

for machine learning. When you see companies deploy this, can you give some examples of the types of labels they'd like to know?

[00:19:18] NG: Yeah, so I can think of one case where we ran, we built a proof of concept for a labeling company was where there were photos being taken within the companies trying to build a Amazon Go style checkout experience where you pick things up off of the shelf, and then kind of just like walking out the door. And then the AI automatically knows what you picked up and sends you a bill afterwards. They're trying to build that for not Amazon. Part of their labeling workflow is around identifying different objects of the shoppers as they're walking through the store. What is their right hand? What's their left hand? Where are their shoulders? Where's their head? Where's the product et cetera?

So, they were using – the proof of concept was around using Flowdash as a means for their labeling team to ingest photographs of shoppers and flag kind of where all these different key points in the image work. So, with Flowdash, that was generally just, again, wiring up the API, being able to like push images as they come in. And then we were able to build a block, block is kind of our word for like component or widget or things that you drag and drop into the user interface when you're working on a particular task. So, we had a block specifically for key point labeling that they were able to leverage to do all this, like labeling. And ultimately, that data got, they were able to query that data via our API to ingest into their machine learning models downstream from there.

[00:20:50] KP: Where are the humans in this human in the loop? Does your client provide the labelers? Or is that part of your service?

[00:20:56] NG: That's a good question. Yeah. So, at this time, we don't provide any of the human in the loop, like the actual human labor. Our clients have been providing them. We do partner with a company that provides the labor aspect. So, if you have one of these use cases, you want to use Flowdash as the software solution, but you still need to kind of augment your workforce to add just more bandwidth, more throughput, we can recommend some firms that we work with that would be able to provide that.

[00:21:25] KP: So, I've seen this done effectively in a batch format sort of offline to train a machine learning model. Can you help someone do this in real time?

[00:21:35] NG: Yeah, so I mean, there's nothing limiting the fact that whether it's done in batch. In the example, I gave for the shopping scenario that was done in batches, again, to just generate training data, but there's no reason you wouldn't be able to do real time, human in the loop with the platform. You'd basically post things to the endpoint, you could kind of think of it again, to do a software analogy. It's kind of a human powered background job, if you're using something like Rails developer, sidekick is kind of the ubiquitous background job processor these days. Kicking something off to there, except, rather than code running, it's an actual human looking at the inputs, doing some processing and making a decision on the output. That's sort of like the example I gave for flagging a transaction that's too large. Ultimately, like it is a human in the loop. The human is deciding whether to let the payment through or not and then that kind of resumes the execution from there.

[00:22:29] KP: And so that label, I'm going to bring my own fleet of labelers, to the table who I've trained, and they're going to use your software, maybe I staff in 24 hours, if that's the service I need. But I'm curious about the flow of their data. So, I guess, their app, or the store's technology needs to post the image to your system. Once I finally get the label, what are my options for sending that back into the rest of my tech stack?

[00:22:54] NG: Yeah, so the label itself, we store the labels as, it's kind of like a nested JSON structure of the various labels and the coordinates at which they were placed. You can fetch that via our API, or we can push that via our API call steps. So, kind of whichever you want. We can also even – if you're doing things in batches, you can export that whole thing kind of manually as a CSV file and do whatever processing you want to do to that offline. Now, if that format doesn't match what you need downstream, you'd have to write some transformation code to kind of ingest that into your downstream models and whatnot.

[00:23:30] KP: And then, I guess the process you described is a little bit more sophisticated than just a simple labeling. It's not like, hot dog, not hot dog. You're getting all that pose information. What's the interface like for my labelers, to be able to provide that?

[00:23:45] NG: Yeah, so in that case, we built the ability to define a set of labels, and they're associated, just colors, just to visually distinguish them. So, in that case, when configuring the workflow, the developers were able to define the labels, in which case, left shoulder, right shoulder, et cetera. And then when the operators or labelers, were using the tool, they would see the image and then on the right panel, kind of the various labels that they can drag onto the image and position.

[00:24:13] KP: And when you think about, I guess, best practices for building flows, obviously, this is a component, the flow is going to integrate with some other system. It doesn't sound like I ever would build a standalone business just on your platform, although maybe that's possible. But with that in mind, I guess my question is, around the integrations, like what's the typical lifecycle where someone says, “We've got to build this. It's going to take us two weeks or maybe two hours to get it done. Then this team has to integrate.” End to end, what's it usually take a team to get up and running?

[00:24:46] NG: Yeah, I think it generally depends on the complexity of the use case. To answer your previous question, can you build like entire things on Flowdash, we actually have seen folks do that, especially with our – we have an external facing formula that you can build similar to something like Type Form or Jot Form or Google Forms, building these forms, that's kind of like the ingest for data from your end users. And then, doing some processing on the back end. It’s an order form of some kind. We're also able to send additional forms to gather more information from the end user. It's not the typical use case.

I'll say, the main use cases, human in the loop, as you've already described, but it is possible to do something more end to end on the product as well. Now, the typical time to delivery I mean, we've often seen something that an engineering team had slated as two to three engineering months of work being reduced two days by building on top of our platform. I think, with a lot of these basically, like more like process driven internal tools, where there's a team, like items of work that arrive, people need to assign it to themselves. You need to coordinate it, you have to bring it to resolution. It's very easy to overlook a lot of the complexity that goes into building something like that. Just something simple, like assigning things and deciding when to assign things to particular people and load balancing them, and being notified when you've been

assigned something or being able to comment on something and have someone else be notified that you've commented and need their attention.

Once you do actually start putting something in production, the next question is like, “Well, how are we doing? Where are the bottlenecks in this process? Where's the room for improvements? What can we tweak?” And so, you start to look towards analytics and reporting. The advantage with Flowdash is all of that is built from the ground up. We've built the building blocks that are really catered for these like high volume, repetitive manual tasks, so that you don't have to build a lot of these primitives, and you can focus on the things that are more unique to your business.

[00:26:44] KP: Well, I'm considering now what a rollout might look like at a large social network, in particular, with the content management use case that I think we touched on and is covered on the site. Imagine I'm like the regional manager. So, I've got grandchildren, employees all over the place. I can't necessarily manage every person, but I need to know something about the operations. What sort of visibility can I get into things like number of open issues?

[00:27:09] NG: Yeah. We have built-in analytics feature where you can get an idea of within a given time period, how many issues were opened? How many moved to various stages? How many were resolved? And what were their resolutions? What is the average time for an individual unit of work to get from point A to B? What's the average time for something to get through the entire workflow? And how did that time change this month compared to last month compared to the month prior? So, a lot of this is kind of, we surface that in our analytics dashboard. But if you want to do any additional analytics on it, it's also all downloadable. We can get the raw data, and then you can bring that into a BI tool into Google Sheets or Excel and do whatever additional analysis you want to do to extract those insights.

[00:27:58] KP: And when you think a few years out, obviously, I think the market size for this sort of tools are growing, are we going to be writing less software in general? Or is there just more software altogether and a bigger market share going to tools like yours, or what's the future of software development?

[00:28:13] NG: Yeah, I think we'll continue to write more and more software. What I predict is that people that generally were not able to write software in the past are now being empowered

by a lot of these tools to build things that they otherwise would not have been able to build, we go back 10, 20, 30 years ago. A lot of people will categorize low code and no code as these newer entrants, these new buckets. For me, I kind of think of things like, I mentioned earlier, I got started on on Visual Basic when I was learning how to program originally. Lately, I've been using Ruby on Rails. I kind of think of Ruby on Rails and these frameworks as ways to get us from like absolutely nothing to working web app in much, much less time. It's kind of providing us a lot of the common building blocks so that we don't have to spend our energy on that. And then we can instead, spend our energy on what makes our product or service unique. I just think of low code and no code as an extension of that. And kind of widening the universe of who can be a developer, who can build apps by no longer requiring you to write as much or any code at all.

[00:29:20] KP: Yeah, I think there's a lot of enthusiasm for that and we're seeing a lot of entrants to the market, very popular, a lot of VC money flowing in. What differentiates your solution?

[00:29:30] NG: I think our solution, what's unique about it is I think a lot of these products that I've seen, are either focused entirely on developer persona, and you know, really having a high ceiling for the product. You can effectively build anything, but you need to write code, and there's a lot of complexity in doing that. Or on the other end of the spectrum, it's very cater to a non-developer, and again, it's providing you the tools to do something that you otherwise would not have been able to do. I've rarely seen a product that tries to address both personas, albeit in different ways. That's kind of where I see Flowdash fitting in is where a lot of these human in the loop processes have upstream and downstream systems that you have to integrate with in order to have those human decision decisions make their way back into your core system and that's where the developers come in, and that's where things like our API, like our web hooks, et cetera, our database integrations. Those are the side of the product that developers interface with, whereas a lot of the process management, a lot of the auto assignment SLAs, the alerting notifications, reporting and analytics are very ops or non-developer focused. So, I see Flowdash as being unique in that we cater to both of these audiences in different ways.

[00:30:48] KP: I'm curious if there's anything coming down the release pipeline you want to tease, or maybe any recent releases you're excited about?

[00:30:55] NG: Yeah, I mean, one in particular that we just released this week is, it might sound subtle or minor. But a lot of products have a form builder of some kind, like be it Air Table, or Google Sheets with Google Forms, Jot Form, Type Form, et cetera. What we've seen though, is a lot of these forms are typically for getting new data into a table. Aa lot of the customers we've talked to, a lot of the prospects were like, “Well, I already have a record for this customer. But I need them to provide me additional data. I want to send them a form that will just provide more information about themselves, whether it's a bank account number or an address, update an address, whatnot, and have that feed into the same row from where I'm already storing their customer information.” What we found is there wasn't a very good turnkey solution for that. Of course, you could accomplish it with like Zapier and stringing together these things, but we didn't see any of these products kind of have like a form that can update an existing record as like kind of a first party concept. So, we built it, and we just launched it this week, we're already seeing some great feedback from it. We're excited to see what folks build with it.

[00:32:09] KP: When you look across current adoption, I think there's a wide spectrum from enterprise development, medium, small business to indie, rogue weekend hacker, where have you seen the most interest coming in?

[00:32:21] NG: Yeah, we've seen the most interest as far as company size, generally, between series A and series C companies, and generally those that are have an operational component to them. So, that's generally companies in the FinTech space, companies in healthcare, where they're dealing with sensitive data, and there's kind of a human element to a lot of their processes. And the reason I say series A to C, generally, at the seed stage or early a, you're still kind of figuring things out, like shooting things over email is working, doing things that don't scale, you're not quite thinking about, how do I build out a team? How do I build out a process that's repeatable? Kind of not at that stage yet. Whereas post C and beyond, you've probably already established some tools and maybe bitten the bullet and built something custom. So, we've generally just had a little bit more trouble selling into those organizations for now. And our sweet spot has been kind of that transitionary phase between early stage and growth stage. I imagine it'll continue to expand from there, but that's at least our starting point.

[00:33:23] KP: What's the typical first step someone of that organization will take? Is it some of the lower end who hears about the product and wants to try it out and show the boss? Or do you see executive mandates coming down?

[00:33:35] NG: Usually, the former. So, someone will hear about us. It's often a developer, sometimes an ops person, but they'll hear about the tool, they'll poke around, they can sign up for a free trial work through the tutorial, we have a bunch of videos on our YouTube channel about how to do particular things, different examples. We have a template library they can start with, and generally, there'll be a moment where like, “Wow, I think we could use this for something.” We're always available jumping on on demo calls and in app chat with questions. But there is a point where they're like, “Hey, I think I have a use case and I want to build up a case internally for us to adopt this.” So generally, we seen kind of the purchase decision come from a little higher up, but rebuilding up of the case has happened more bottoms up than it's been tops down so far.

[00:34:20] KP: Is there a typical hello world? Or what's the first sort of test or integration that I can get up with really fast that shows the value of the platform?

[00:34:29] NG: That's a great question. I think, when you start out, there's our tutorial and our tutorial walks you through – five-minute introduction to the product that walks you through an example where you're a brand new startup, you have a waitlist for your product, so you're not even fully live yet. Or maybe it's like a request early access form. And those kind of enter into Flowdash and you ultimately are deciding which of those users you'd like to invite to the platform, versus which of those you want to maybe put on a waitlist for general release.

So, we kind of like walk you through building up that whole workflow over the course of a few minutes in our tutorials. I'd say that's probably one of the better introductions to just how to use the product. And then there's a few videos kind of on our channel about typical use cases. We have one around content moderation. We have another around KYC, and one around like reviewing transactions. I think those are all kind of typical examples that showcase, what's possible in the platform, or what kind of use cases it's particularly well suited to solve.

[00:35:31] KP: And when you think about scale and growth, obviously, it's more customers, more use cases. But from a technology point of view, are there any things you're chasing after?

[00:35:41] NG: Yeah. I think one surprise we had was just the volume of units of work that we're seeing our customers push through the product. I think we were expecting things to be on the order of hundreds and thousands of tasks. What we're starting to see more and more of his very high-volume workflows of tens of thousands, hundreds of thousands of tasks. So, from a technology standpoint, ask ourselves, how do we continue to scale our infrastructure, our performance? How do we leverage caching in various ways? How do we paginate particular user interface elements? And just, how do we keep the product snappy and productive, even as the number of records in these workflows continues to increase?

A number of our clients are actually running our product on premise. So, that's one thing I hadn't mentioned earlier. But apart from our cloud product, we also offer Flowdash to be deployed on premise with a dockerized deployment solution. And that's particularly well suited for companies that have really stringent data security requirements or data privacy requirements, those in industries like finance, like banking, like insurance and healthcare, where there might be you know, HIPAA requirements. Running Flowdash on premise would have been a nonstarter if it's not something we allowed. So, in those cases, how do we continue to make be able to support both our cloud customers and those that are on premise? And how do we make the on-premise deployment and upgrade process as easy and seamless as possible including things like migrating data and so on?

[00:37:12] KP: Well Nick, remind listeners where's the best place to learn more?

[00:37:17] NG: So, if you want to learn more about Flowdash, you can go to flowdash.com. You can watch a demo video there. You can also check out our YouTube channel, which you can just go on YouTube search Flowdash, I'm sure you'll find us. But those are probably two of the best places to learn a little bit more about the product and the company.

[00:37:34] KP: Sounds good. Well, thanks so much for taking the time to come on Software Engineering Daily.

[00:37:37] NG: Thanks for having me, Kyle.

[END]