Brian Gracely (00:01.393)
And we're back and excited to dive into this one today because we've been talking a lot about, obviously, a lot of things on the show. We talk about developers. We talk about APIs. We talk about AI. We talk about cloud. And I feel like sometimes we sort of silo what we're getting into. We don't really talk about a lot of the intersections between these things. And right now, so much is evolving and so much is changing, whether it's how code's going to be built, how teams are interacting around more complex applications.

All those types of things. And we kind of wanted to take an opportunity to really dig into what some of those interesting interactions are today. And so really excited to have Taylor Bukacic, who is head of product for API Collaboration Core at Postman, joining us today from Boston. Taylor, welcome to the show. Great to have you on.

Taylor Pechacek (00:45.73)
Hey Brian, really glad to be here. Thanks for having me.

Brian Gracely (00:48.005)
Yeah, no, excited to have you because I know, you we've been obviously doing some doing some work with Postman here recently. Before we dive into a lot of this, give us a little bit of your background, kind of, you know, your background, but also what you focus on these days at Postman.

Taylor Pechacek (01:01.976)
Sure. Yeah. So I did a few startups coming out of grad school, kind of in the healthcare space, all in the product management side, thinking about front end development, product management. And then I joined it last year in about, about seven and a half years ago, eight years ago, and just had a really incredible journey there building out products and was really on the side of that company that was thinking about how software teams actually collaborate, how they build things, how they operate stuff. And that was really working on things with Injera Software.

I looked at a lot of the integrations work. So how does get information, CI, CD, feature flags, you essentially answering those questions of like, Hey, is this issue deployed or not? Right. How do we know the status of work? Because the work was kind of happening across all of these tools, but it wasn't really happening back in JIRA. And so how do you kind of bring that context and, and help people just move work forward. And then the last three years was really fascinating. We built a product from the ground up called Compass at Alassian.

And that was really in this internal developer portal space. And we were competing with things like Backstage from Spotify, which is an open source version. There's a handful of startups there, Cortex, Port, a few others. And Atlassian had been on that same journey internally. A lot of microservices moving to the cloud, increasing complexity for developers. They had also invested heavily in kind of their internal compute platform.

you know, called Micros, and then we wanted to build a front-end portal for that. And we thought we saw so many customers also investing in developer experience, the portal side of things, and how to really make sense across all of these different tools that were being given to developers internally. And we packaged that up and that was built out as Compass. And so had a wonderful time there. Really loved that company, a lot of good memories made. so I it was the right time for me to leave. Had a really great role and opportunity to join Postman.

And I've been there for about seven or eight months. And what was really fascinating about Postman was that they had this amazing tool. And let me pause there for a second and tell a little bit about what Postman is before I continue. So Postman, for those who don't know, is really this kind of end-to-end collaboration platform for APIs, right? So if you're developing APIs, designing APIs, operating them,

Taylor Pechacek (03:24.652)
It's really going to handle everything you need to be able to track those, to build against them, to test them. And you can do that with yourself, of course, and you can do that with now a group of people and even across the entire company. Right. And it really started many years ago and kind of sending this request response from an API. And so an API being an application programming interface, you would hit that API. So if we're using, Atlassian or Atlassian Jira, Hey, get issue. It will return the results.

from a data perspective of what's on that Jira issue, know, and the assignee and stuff. And so what I saw really through my time at Alassian was more and more people were more and more developers and software teams were spending time trying to figure out how to, how to deal with two or more services or APIs or parts of these products. And they needed to stitch them together. They needed to assemble them.

And Postman is just right here in the middle of this because it's the entire communication layer of how technology kind of communicates back and forth, right? And it's through your HTTP REST API's. You can move into other things that are more event driven. There's new standards, of course, with GraphQL and stuff. so yeah, joining Postman now, what I think about a lot, still software teams, still collaboration, but really thinking about it from the perspective of how can we help people assemble that software?

Brian Gracely (04:31.271)
Yep. Yep.

Taylor Pechacek (04:50.574)
through the use of these APIs. And so I think a lot about how a group of 10 people might want to use Postman to make their lives easier, to be able to build faster, to be able to build higher quality APIs in that world.

Brian Gracely (05:02.279)

Yeah, no. And that's a great place to start because so much of, as we get a chance to talk to development teams, there's always interest in the thing that they're building. And then there's always this sort of like pause or a sigh. they go, but I spent so little of my time writing code. There's so many other things that I've got to keep up with. It's this system. It's that system. It's almost like.

they've got flows in their mind. They've got sort of workflow in their mind. And they're always constantly kind of, I don't want to say fighting, like kind of struggling with, know, just do certain tools kind of help them, you know, accelerate what they're doing. Are they kind of pushing against it and so forth? it feels like we're going to get into a lot of pretty interesting stuff today. Let's, again, just so we kind of can level set with folks that are listening, let's start with some real basic concepts.

If you're a developer, you're a small team, you're working on a project or a service, where in today's world are they interacting with software typically? They're building something versus more so they're interacting with APIs. They're having to bring together an entire experience. What are the most important things in today's modern developer world in which those things, how do they think about those two things either separately or together?

Taylor Pechacek (06:22.914)
Yeah, that's a great, great place to start and a really good question. I think, I think one way to kind of frame this is we can kind of go through the life cycle of let's say we're starting something really small and we're creating our first service together. So you and I are in a startup, we're building something, we want to create something. We'll probably consist of a couple of things, right? We'll have our backend, we'll have some compute, we'll have some storage, we'll have some networking, and then we'll have our front end. We'll probably render that to people through UI of some, of some kind.

And a lot of times when you start with that, you're going to build development where you're essentially interacting with data in some way. And how you want to test that is, you know, can go much quicker by just using a REST client like Postman to be able to mock out, hey, let's get some data. Let's post some data. Let's modify that data. Let's see how it looks in the UI. And so we're actually going to use that. That is, you could call it testing, but I actually really think it's much more

better way to describe it might be that you're debugging or you're building and you're kind of testing the limits of said system. And so you're kind of really in this, this build mode that you're trying to understand what you're working with. And as you, as you kind of build that out, you know, postman has a lot of mechanisms to be able to even test that API to the second person. So I may not have like deployed my local host application yet.

But the contract I've created through the API is there. And I want to know if you on the front end who are going to go work on building out this UI with maybe a list of things and different buttons that you can press to update stuff. You can trust that contract, right? And you can build against

it. Now we can start to do things in parallel. And so I can go work on my back end. might change even my entire language. I might change my entire system design, but you have that contract to rely on. And so the APIs are really there to help people understand

Hey, what capabilities, you know, what capabilities does this piece of technology offer? And it is the interface that describes that capability. And that gives a really, really a large amount of power in terms of working with other people. now if we expand out and we have 10 people in our team, right? How do we kind of come back to those APIs, understand which ones are meant to be trusted? APIs themselves are often blueprints. So you might have 20 parameters that you could pass in, right?

Taylor Pechacek (08:44.312)
What are some of those patterns? know, hey, get me the issues or give me, give me some data from Slack and all these channels that have these filters or maybe bulk update a thing or notify across a set of channels. There's probably a few ways you might use that API that you can start to impose as examples and people can pick that up. They have all the different authentication mechanisms with them within that, and they can really quickly get up and going to start to build against that. So I think like if I were to zoom way out.

you know, before we get to kind of the, the, standards and the governance at like large scales, within that one to 10 person team, it's really this, this back and forth between the underlying software, like the compute, like what is it doing and the interface that it gives you, like even AI, if we talk about AI, you know, you're calling an API to essentially give a query and you're going to get some non-deterministic result kind of back, right? But at end of the day, it's still this, I have a request.

something's going to do something, some black box that's there. And I think it's more black than before, you know? And so it's getting darker. And so, but it's going to give me a result. And how I work around that is really how we think about APIs and the interfaces. And then it becomes, you know, mapping that out across a sequence diagram, or if you're familiar with, people are familiar with sequence diagrams, you know, you're going to call this API, then you're going to call this one, then you'll do some, you know, compute on your data and then you'll call the next one and so on.

And that's a great way to think about that in terms of these flows or these sequence diagrams that you want to work with. And Postman thinks a lot about how those group of people work on that.

Brian Gracely (10:22.983)
Yeah, no, think you hit on a couple of things that to me resonated a ton because I have this saying I try and explain to people all the time that nowadays every sort of business project is a technical project sort of waiting behind it. business team sits around, they come up with some idea for getting into the market, delivering something new for customers. there's always, again, it's no longer like,

go to a physical store, do this thing. It's how are we going to implement this in the mobile app, in the kiosk, in whatever. And so there's an aspect of it that the business owner is going like, I'd like this experience to happen. I can imagine in my mind what this experience would happen. And that's, to me, very much the things you were talking about, like the ability to experiment, to validate that things work, to do it in small teams, but to validate it with external groups. But at the same time, once you've sort of

proven that to somebody that you can show that to somebody. mean, their immediate expectation is like, OK, great. When can this go into production? And that's, to me, very much the whole idea of these APIs sort of service contracts for what that overall experience is going to look like. You can build security around it, but you're very much understanding that's my interaction with those people. And you can build process and flow and things around that. I think the way that you explained that was fantastic, because it helped me

kind of connect the dots between the way a business owner, whether it's a product manager or even a general manager working with people, trying to map that down into the technology and then the way that software developers have to kind of, again, build it, reverse engineer it, and then lifecycle it so that it can go out and scale and do stuff is really interesting. Yeah.

Taylor Pechacek (12:09.9)
One thing if I might add there before we move on to the next question is that I think to your point, like thinking in this way, AI is accelerating that mental model because you have the ability for this, this thing, this like smart bot or agent or whatever you want to call it, that is able to kind of, write the underlying Lego pieces, right? The underlying code that's going to execute on your idea. You can call it vibe coding. You just go for it, you know, and just.

Brian Gracely (12:33.523)
Mm-hmm.

Taylor Pechacek (12:39.682)
Just let it ride. But at the end of the day, you do no matter what you do, you have to have some interface that is going to access that thing. There's some like, yeah, translation layer, right. And that's always going to exist, whether that's AI, whether it's rest APIs. and I think that's really interesting that that, that, that that's going to always exist. And so what you can do is you can start to be a little bit more API first. In your thinking, you can start to be a little bit more, Hey, this is what I want to achieve.

And a lot of the technology we're building is actually coming into support that abstraction. most of technology is adding a new layers of abstraction. And I think we're kind of coming to that point. You have it with AI and being able to abstract across all this knowledge base. But then also with APIs, think having its moment in terms of being able to access, and we'll get into that with how AI calls these APIs, they call them tools. Right.

Brian Gracely (13:14.643)
Mm-hmm.

Brian Gracely (13:36.051)
Mm-hmm.

Taylor Pechacek (13:38.99)
people on the ground, people who may not be developers, they have business problems. Everybody has like these problems they want to solve. And I think the faster we can help translate that into some working piece of software, it is really exciting. And it's usually going to be the combination of two or more of these discrete pieces of technology.

Brian Gracely (13:50.685)
Mm-hmm. Yeah.

Brian Gracely (13:57.683)
Yeah, yeah, no, absolutely. I feel like about once a year or so, we kind of go explore the API space. And we've had a chance to have a number of people on, similar to yourself, sort of live in this space all the time. It's not a side thing. And I always feel like there are so many new kind of concepts coming out. Like you mentioned GraphQL, and there's certain specifications. We talk about REST APIs and others.

Are we seeing more standardization around APIs? Are we seeing more consistent frameworks? Or is it still kind of the way it is with programming languages? People kind of pick the ones they like, and stuff can be made up of whatever programming language.

Taylor Pechacek (14:43.022)
Yeah, OK. I think there's definitely isn't as broad as like maybe the languages and other things. There's definitely some dominant patterns. I think if you look at kind of the different protocols that are out there, we can bucket those into kind of two large camps and maybe a third now with AI. But the first camp is kind of this request response, right? And that's HTTP, GraphQL, these types of formats. And then there's also kind of event driven. So with async.

Brian Gracely (15:04.435)
Mm-hmm.

Brian Gracely (15:11.537)
Okay. Yep.

Taylor Pechacek (15:12.64)
API and those, those two kind of have their own paths here and HTTP still being very dominant. I think the reason that GraphQL came out was that you wanted to optimize for the consumer to not have to stitch together six different REST APIs, but to say, I actually just need name, field data. It's like a, it's like an auto join of a table. It goes and does all of its stuff. And then cool. It's

optimizes for people consuming. So again, really improved, big improvement on the developer experience.

Brian Gracely (15:27.933)
Yeah. Yeah.

Taylor Pechacek (15:42.51)
And so that's all a lot of traction, which is great. There. then the third one, I think would be kind of AI. I think this new kind of not the non-deterministic, you know, I don't really know what I'm going to get back, but I'm going to get something back and I might do something with it. That introduces all these interesting problems with how we design systems that historically have been very deterministic. So, you know, I get a request, I send it back. kind of expect what I'm going to get so can write tests and other things around it.

Brian Gracely (16:11.281)
Yeah. Yeah. So it's interesting. And we're going to dive into some more of the AIA stuff in a bit. you actually think about these queries into LLMs and stuff as almost like an API, like a non-deterministic API. I mean, is that kind of a way you think about it as a category, maybe?

Taylor Pechacek (16:11.714)
That's going to change quite a bit with AI on the third side.

Taylor Pechacek (16:31.31)
Kind of. Yeah. I don't think we've really nailed that down internally. It's part of a lot of discussion. I mean, even if you take flows, which is one of the, one of the products that we have within Postman and you're able to chain together various different APIs to create kind of a workflow, right? And automation and what we, what we've kind of launched recently is looking into these AI agent builders and really the unlock layer is that you can add a step in that, in that flow that

Brian Gracely (16:53.458)
Okay.

Taylor Pechacek (17:00.17)
You can ask an LLM to do to like make some call. Tell me whether it's this or that, depending on what the LLM tells you. So let's say like, here's a bunch of data. I'll give you real example. Here's a bunch of data from Slack, a JIRA issue, you know, something in the observability tool. You tell me what severity this should be as an incident. And, and you don't really know, you don't really know what it's going to tell you, right? You know, and over time you have to train. This is what I mean by.

Brian Gracely (17:18.715)
Okay.

Brian Gracely (17:22.738)

Yeah. Yeah.

Taylor Pechacek (17:26.798)
How do you train that? is it like trained on? How does it start working? How do you get to a really good level that's maybe just, okay, now it's good to go. But you don't really know. And so you can think about that problem and you can apply that to so many areas that you might want to predict something and you can embed that in your flow. And so now you have this, let's say there are 10 steps and three of those may fork at any given time depending on how open-ended the responses are and how many paths you want to kind of add to that.

And so yeah, I think there's non-deterministic nature of you give it a bunch of context, you ask it a question, and then if it does something, that's why giving it access to tools is such a fascinating problem because it's really cool at first glance. But what does that mean in terms of security, wiping out, you're actually wiping out your entire database because these APIs are giving you keys to the kingdom for many of these APIs.

Brian Gracely (18:13.297)
Yeah. Yeah.

Right, right. Yeah, no, that's interesting. I hadn't thought about it that I think maybe I had thought about it as like, people were building things on top of, like the OpenAI API is sort of a thing. But the rest of it was just sort of data interaction. But I guess, like you said, if you just sort of break down what an API is fundamentally doing, it's sort of requesting or interacting with data and then doing something upon it. So I like that.

Taylor Pechacek (18:44.334)
Hmm.

I want a couple more to add to the standards of maybe if I could come back to that. Yeah, sorry. So I think the other thing within the standard, so let's let's take the HTTP rest API, which is still such a dominant force in the world. You really want to think about a few standards there that have come out around the open API spec. And why did that kind of why does that exist? Why does this specification exist? I think what why it exists is that you want something that can be very portable. That's very consistent and is really kind of.

Brian Gracely (18:47.436)
Let's. Yeah, go ahead. Yeah, go ahead.

Taylor Pechacek (19:17.07)
tool agnostic, let's call it, right? And so a lot of times when you're working with APIs and Postman and you're building all these out and you're saving them as your collections, so you might have a collection that's like a, if you're a library, you might have like a book collection so you can update books, read books, find, you have all these different endpoints and these

interfaces that give you that. Being able to export that or vice versa, like compile that from the code, you have the underlying endpoints, which is

We talked about that. That's the actual code, the implementation, there's compute behind that. And then you need something to represent that. And that is usually this open API spec, which has been pretty dominant, as well as open collections. A lot of people kind of use those collections and open API specs kind of together. But again, most software teams already have Git, right? We've all standardized on this and there's the kind of three major players, GitHub, Bitbucket and GitLab, I would say, and GitHub being by far the dominant force in that world.

Brian Gracely (19:56.712)
Okay.

Brian Gracely (20:04.104)
Okay.

Yep. Yep.

Taylor Pechacek (20:16.724)
And so being able to work with Git and those protocols, which means, what does that mean? That means that you have access to the pull request flows and on every pull request, you have access to be able to shift left. You have the ability to run tests on every PR change. You have the ability to kind of get embedded in that, that really core developer workflow as they get, move things more towards production. And so being able to check for breaking changes, being able to understand the endpoints.

That's a powerful mechanism there that you want to be able to work really closely with. so Postman is thinking a lot about how they work closely with Git and Postman, because you're going to do all of your, the executable, the documentation, how you actually work with those API, all of that's happening in Postman.

Brian Gracely (21:01.405)
Yeah, yeah. We've been talking a bunch about sort of development within a team, smaller teams interacting with a couple of services. As the team either grows or as things become more pervasive in larger organizations, you start to get into kind of best practices, governance, all those types of things. What are the kind of important things that people should be thinking about, whether it's

around usage patterns and governance, or some of the things that Postpan's doing to, again, like we talked about, sort of make things easier for developers.

Taylor Pechacek (21:37.102)
I love this question. So I spent many years thinking about like, let's call it common developer problems. Like what is governance? Governance is like, I have a hundred out of a thousand

developers. I've got a thousand repos, a thousand workspaces in Postman. How do I deal with that complexity? And I think it's this really great model. think platform engineering has driven this a lot forward. Kind of this, you had Agile, you DevOps. think platform engineering has added a lot to that conversation.

Brian Gracely (21:43.251)
Mm-hmm.

Taylor Pechacek (22:05.132)
And what it's added is really around being able to build happy paths or build like things that are, here's the right way to do things is this way. And it's really easy for you. And if you want to break out of that, of course you need to innovate, you need to go do different stuff. But for most of the development, most of these teams, they have a lot already built out for them. And so I would make the argument that like, taking Google or Meta,

You know, their developers, like even a new developer coming in is probably like a level 99 warlock, you know, super powerful comes in and just is able to work quite effectively relative to maybe a startup that doesn't have the same tool set, the same investment. and so, and that's relative to the size of the code base and complexity. And so when you think about this governance, you know, one of the things we're doing within Postman is thinking about how you have all these endpoints, right? So it starts with endpoints, starts with these requests.

And what you want to really do is start to think about the ownership of those requests and actually grouping them into what we call collections. And a collection is essentially a piece of technology that has those interfaces. And so this kind of gets into service design or domain driven design, but you want to think about like, you know, okay, I have a library. Let's go back to the library example. I have something with books. I have something with users. I probably have something around search.

And so you might want to build a collection for each of those building blocks of your app. And each building block has a set of APIs by a set of endpoints. And those are the interfaces that access the capabilities in that. And so when you think about this collection, this is where Postman gets really interesting. That collection, you can start to give it a life of its own. You can start to test against that. So you can say, I want to make sure all these endpoints are always running effectively. I want to make sure that they actually have.

good tests at game, good tests. so when I get results back, expect certain results. And so if someone changes something in the code, you're actually going to get different results back. You can catch that. Right. And then, and then you can even check that ahead of time during your pull request flow. But the point is that you build out this collection in terms of the quality and the state of that thing. And you have this now this long running piece of technology that's like, wow, you know, if someone wants to deal with books in my company, I should probably bring them to the books collection and they'll have everything they need to do.

Taylor Pechacek (24:28.09)
They can fork that, can work with it. And so one of the things that we're working on now, which I'm really excited to kind of announce that's coming out, is two things. The first one is what we're calling specification hub or spec hub. And that's the ability to kind of integrate really tightly with Git. What I was just talking about with some of these portable open API specifications and connect that back to the collection. So this books collection that we have, it is represented in Git.

Any changes I make in code are represented in the collection in Postman as I worked against it. And vice versa, as people work with that collection in Postman, it's going to be represented back in the code. So they have this kind of running kind of source of truth there across both. And so that's really exciting. You'll be able to kind of author these things. You'll be able to design these specs. You'll be able to generate. So you'll be actually be able to take an entire open API spec, generate collections from it, generate the inputs, and then work with that, run those APIs, test them.

Brian Gracely (25:22.6)
Mm-hmm.

Taylor Pechacek (25:26.944)
I give them to colleagues and kind of work with and collaborate together. And then the second thing that's really exciting is around typed collections. And this is just such a great, you know, thing that should have come in. You should have been out for a while, but it's just such an awesome piece of technology. And that is essentially adding linting to these collections. And so how many times have you, there's so many people have told us they come to a collection. come, you Hey, come check out this API, this books API, right?

And you get there and you have no idea if it's required fields, if it's optional fields, you start hitting the API, you're getting 400s, you start getting frustrated. And so what the type collection does is it adds these like linting parameters to say, Hey, this is what you need to fill out. If you didn't fill this out, it's going to give those error warnings back to you. And so the developer experience of using an API that you're not familiar with is going to be a lot, it's going to be a lot better. And you can kind of set these rules, these governance.

kind of aspects across, you know, hundreds of teams, hundreds of workspaces within Postman. And so it gives them a lot of kind of leeway to apply this, this governance, but still let people kind of work in their own way.

Brian Gracely (26:35.859)
No, that's very cool. Very cool. Yeah, no, mean, just the amount of collaboration you end up doing between teams, and do we have the right versions, and when do they get onboarded, and just all those things that you go, those should be simple. should be like they just, they're never as simple as you think they are. So yeah, the fact that you guys are thinking about those in

ways that are repeatable, shareable, versionable, no, that's fantastic. Very, very cool to see. Yeah.

Taylor Pechacek (27:03.566)
And one last thing there, Sarah, Brian, and it just reminded me like the, documentation piece is actually really important here because I think most developers today, they're still kind of have like the documentation up to the right. And they're probably using Postman here to the left and maybe their VS code or some type of coding thing. And they're kind of working across the set here. And so they're looking at the docs to try and figure out, you know, what's required, what's not, how do I work with it? I think Postman really views that as like, how do we collapse it?

Brian Gracely (27:16.083)
Mm-hmm.

Taylor Pechacek (27:32.568)
So the documentation and the executable are just the same thing. Why is that different? That's so weird, right? Like we should get to a world where that's the same thing and then we're collaborating against that and evolving it and making sure that it has a stable representation, you know, back in Git and things like that.

Brian Gracely (27:46.963)
Right. Yeah, no, that's very, very cool. I want to come back to, we kind of scratched the surface on AI. We were kind of talking about it as kind of a non-deterministic way of thinking about APIs or just its interaction. But I'm curious just more holistically, because like you said, you spend a lot of your time thinking about developer interactions, how to make things easier, kind of what they're dealing with. What are you seeing in terms of how AI is

is changing developers' work and how much that's going to impact APIs. Obviously, we hear a lot about co-pilots and helping to write code, or even now autonomously writing some code. But how are you seeing it, or how are you thinking about what AI is starting to do in the developer space or lifecycle space?

Taylor Pechacek (28:38.478)
There's so many ways to go about this question. love it. so let me start with maybe the first thing I'll say is that AI is very much this horizontal purpose technology. I mean, I think we really believe that AI can be applied at every single step of the journey. And when you think about that, you think about like, is, where is it somewhere I can predict, you know, something's off track boom prediction. Ooh, some things that shouldn't be an incident or not prediction. Right. and so you want to think about kind of broadly.

these workflows and where you can start to embed AI within those. And then the other thing is that I think AI is helping developers write more code, right? And they're able to write more like code faster. We can debate the quality of that, I think for a long time. I think we'll kind of sort that

out over time. There's agents that will actually check, you know, what the person writing the code or the AI writing the code. So they'll just start to do stuff without us, I suppose.

Um, but, you know, to your point earlier, 30, let's say 30 to 40 % of a developer's time is spent actually coding. The rest of the time is kind of understanding specifications, thinking about the system that they're working within. How are they going to go about this? Should they, should they add an endpoint to an existing service? Should they create an entirely new service? Um, you know, how do they think about it? Everyone becomes kind of this mini architect and the AI is there kind of helping them write this code to be able to fulfill that.

And so one, think fundamentally that that's going to create a lot more software. That's going to create a lot more discrete pieces of software and it's going to create a lot more interfaces to them. And hence, I think there's going to be a tremendous explosion of APIs in the world and how we reason about those, how we, how people work with that complexity is a big question. You know, which one do I trust? Like, you know, which model should I use? Like, how do I navigate all the stuff that's happening so quickly and how do I test these things out?

So that's one, think we want to give people the ability to just work kind of what what Postman did in its origins was like, Hey, there's all these APIs out here. Like how do I test it and validate that this is the one I need. Right. Okay. Yep. That's the data I get back. That's how it works. Cool. I'll pull this into my system. I think we want to do the same thing with AI in that regard. And then the second thing going, going beyond that is if they're going to be all this new software and new discrete units, you want to help people craft or assemble software.

Taylor Pechacek (31:05.184)
And so we have a simple mechanism in Postman, kind of like Git, but you're able to take an API and you're able to fork that into your own workflows, right? I need to use the Books API. I will fork that and I can actually edit it. I can make changes to it and I can kind of push that back to the original API owner, right? And then it can kind of, maybe they might update the underlying code and other things there where they can push back what they want, what they want to occur, the interface. And so what happens when you start to say, well, actually, you know what?

Brian Gracely (31:12.754)
Okay.

Taylor Pechacek (31:34.572)
Let me take my sequence diagram and I know that I need to work across eight, eight, eight APIs across eight different teams in this company. What if I could go into postman and just, you know, let me grab that one, that one, that one, that one fork into a working canvas. Right. And so now I'm working with like the, the interfaces that I need and I'm starting to assemble the small feature, this application that I'm kind of working on using those different APIs. And I have all the authentication sorted for me. I've got environments and variables sorted out for me.

I can turn that into a flows or a visual diagram to kind of understand how this is actually working. have all the debugging and responses I'm getting back to be able to work with that. So I think you're really starting to improve the developer experience when actually working across two or more of these systems. And I think it's a slightly better version than vibe coding. I think it's a form of vibe coding, but you have a little bit of safety and quality that's kind of baked in there.

Brian Gracely (32:24.701)
Yeah.

Taylor Pechacek (32:35.092)
Expressing through Postman, platform, right? Like the companies impose that governance and they set those standards and authentication. so yeah, developers shouldn't have to worry about that. They should be able to build.

Brian Gracely (32:43.313)
Yeah. No, that's cool. want to, I kind of want, like you said, we could dive into a lot of things. I kind of want to, want to wrap up and hit two topics with sort of one question. Obviously we're, seeing a lot, a lot discussed these days and people are excited about this idea of, AI agents or agentic AI. And at the same time, you know, we're starting to see this new flurry of kind of protocols around this, right? One of them being

MCP, Model Context Protocol. Google just came out with one called A2A, but I kind of want to focus on MCP because it seems to be at least in the immediate term on mid-April here, 2025, the one that a lot of different companies and projects are kind of rallying around. As you're thinking about this from an API perspective or even just kind of the context, way that you think about stuff getting built.

What are the big things that people should be thinking about in terms of, do you think about agents as like a user? Do you think of it as a service? Have people started to wrap their head around, what does that mean from a security perspective? I don't know, holistically, where do you see that space right now? And what are the things in that space that you're thinking about the most?

Taylor Pechacek (34:01.87)
Yeah, it is moving really fast. I think the A2A announcement, got to go read up on to understand a bit more and even the model context protocol, we're all kind of catching up. So I think with AI, think with the agents, we started with a couple of things at Postman, which we're really excited about. The first one, we announced this a little bit a while ago, but the first one is that you can literally build agents and you can interact with AI.

within Postman directly. So what do I mean by that? The first thing is that you can come in and there's a request type. So instead of calling HTTP, you're actually calling AI and you're calling Anthropics, you're calling chat, open AIs, and you're able to kind of like interact with these, your token counts, see your response rates. You know, you're able to get data around the

effectiveness of which model you should use and what context. And then as I mentioned before, you can actually group this into a collection.

And you can run entire test suites against this to see which one's going to cost you the less to run against a set of APIs. Like very easy. Yeah. there's some really cool stuff that we've, we've built there that I would encourage people to check out. So that's just being able to interact with AI and understand which models do I use? Like, how do I work with this kind of like I would with any, with any API. The second thing is that flows itself started as, it is this visual low code, no code builder.

Brian Gracely (35:05.795)
that's nice.

Brian Gracely (35:20.413)
Yep. Yep.

Taylor Pechacek (35:29.144)
this workflow builder. But what's really cool is that Postman has this API network, right? And so not only is it just, here's a bunch of stuff out there, but it's high quality. So this is where PayPal hosts their APIs. This is where Salesforce hosts their APIs. And so while you're actually building, you can use Slack dot and it will add the APIs that are high quality in the network that those companies have put onto Postman's network.

And developers can come in and quickly chain those together to build these workflows, these flows, right? And now what you've done is you've been able to add essentially non-deterministic, non-deterministic AI calls. And so you can take, know, grab this data from Slack, you know, do this thing from Salesforce and then, by the way, go pass this to this, this LLM code block and come back and do something.

And then you can continue the flow. So that gives this like really incredible superpower to just these workflows, these flows. Now the question becomes, how do you up level that to agents? Right? Now the agents get really interesting here where now you want the agent to be able to have access to the API. So it's no longer an Oracle. So before we talked about kind of the AI is more of an Oracle where in this step, tell me yes or no, or categorize it in one, two, three, four.

And then I will do something with whatever you say. This now is moving to the world where the agent itself is starting to act on the world and how they act on the world is through those interfaces. And that's where I think model context protocol comes in, in that it is, it is giving a consistent standardized way to be able to access these various API resources. And what we call that essentially is like tools, right? You want to give access to the tools. So if I build an agent.

It doesn't have access to my Dropbox at the moment. It doesn't have access to my Figma. You know, it doesn't have access to my tools. And so how do I give it access to tools and, you know,

in what capacity? And so think this is the big security question. It's kind of like, remember all the off flows where like those evolved so much where, Hey, just off, you know, give Slack everything you want or give it last everything you want versus, Hey, here's like the three things it's going to take. And let me let you check boxes on, know, do you want read, writes on this versus this versus that?

Taylor Pechacek (37:52.664)
I think you'll probably need to have something similar with what exactly can this AI do? What are the mechanisms to replay those events? Like what happens if it does make wrong decisions? Right. And so I think the model context protocol just provides, it's almost like, if I were to think about it, it's almost like what GraphQL did to HTTP. You want to optimize for the consumer. So you're optimizing for AI to not go stitch together all of these rest endpoints.

Brian Gracely (38:02.714)
yeah. Yeah.

Taylor Pechacek (38:20.558)
You know, for a given company, you're not going to expose a hundred percent of your endpoints. So you think about like, Hey, what's the set I should expose and how do I kind of group that? That expression is kind of the model context protocols. You're expressing it in like a consistent format that these AI agents know to work with. And so I think it's just a huge advantage for the, the, for the whole industry to be able to reason across this and Postman's, you know, allowing you, you know, we're already working on this, you know, being able to.

Brian Gracely (38:23.933)
Right. Right.

Taylor Pechacek (38:50.318)
call out to those MCP and get results back and then start to add those to your agents. So as you build that in flows and you build your agents, you can actually manage what tools you give access to. And so that's what people are trying to solve. Like how do you put all this together really so that you can build these effective agents? They are solving actual business problems, you know, and you can manage those ongoing. A lot of people forget about the cost ongoing and what happens to your point or what I saying earlier, like if it makes a mistake, where does that go? How does it get audited?

Brian Gracely (39:08.593)
Right. Right.

Brian Gracely (39:19.367)
Right, right, right. Yeah, no, a ton of things. It's a fascinating time, because we are giving so many more people the ability to have the power to go create this stuff. AI is making it somewhat easier. The beauty of it is so many of the things behind the scenes, like you said, how you do

governance, how you do versioning, how you make it efficient. You guys have solved a ton of those problems, and now it's just bringing that to.

Taylor Pechacek (39:19.962)
Yeah.

Brian Gracely (39:46.429)
kind of a different context. so it's very, very cool to see. It's awesome to see people that are just super passionate about this space and are constantly thinking about how to make things simpler for developers because so many things are kind of on their shoulders these days. There's just so much kind of challenges that they have in terms of keeping up and learning new stuff. so it's very cool to hear all this. I know we dove into a million things. We will put a ton of stuff in the show notes for folks because

You know if you're like me Taylor gave you a million things to go think about and you're gonna want to go go read about them So a last question You know obviously Go to the website to get stuff But are there places if people are are engaging with you or engaging with some of the new stuff? That's like sort of the best place to go to either learn about it or you know kind of get engaged with the teams

Taylor Pechacek (40:36.418)
yeah, great question. it would just be, you can go to postman.com. We have our community where you can see some of our kind of drops, like our newsletters, where you can see all the new capabilities coming out kind of each month. I would encourage you to check that out. And then we do have like the Academy or the learning area where you can just dive in and you want to see, know, how do I work with APIs? And then we're slowly adding more content to that. That's related to some of these newer aspects around AI and other things, but the core postman is still there and we're making it better every day. So that's the place to check it out.

Brian Gracely (41:05.933)
Cool, very cool. Good stuff. Taylor, thank you so much for the time today. I learned a ton of stuff. I know we went into a whole lot of topics for folks. for everybody who listens, thank you. On behalf of Aaron and I, we want to thank Taylor for his time. Again, there will be a bunch of things in the show notes, so you can kind of catch up to all the things that we talked about. With that, I want to thank everybody for listening. Again, I want to thank Taylor for his time. And we'll wrap up, and we'll talk to you next week.