Krish (products.snowpal.com) (00:01.121)
Hey there, in this video, we're gonna take a look at one of our APIs. APIs serve as building blocks and one of our APIs is actually called a building blocks API. Let's spend, you know, just a couple of minutes, maybe a few minutes at most and try to understand a little bit about this API and what the API actually offers. We'll take an example, a real life example, not a manufactured one and we'll...

you know, go through some of the items that would be relevant to you as an application developer. Now, we're not gonna look at a microservice or a mobile app in this particular video. We could, but you know, I'm just gonna try and keep it short. So we're gonna go and look at a web application, one of our products that we built at Snowpal. You could do the exact same set of things for mobile apps and microservices. We're gonna be creating a series of videos.

So we'll cover the other types of clients, mobile clients and microservices, if I could loosely call that a client, in the subsequent podcast. We're gonna do more than one for just the web app in itself, just so it's clear. This is the first of those. Okay, so what are building blocks? So to get to this API documentation, you can just go to developers .snowpal .com. You'll be taken to the homepage.

You can look through it at leisure, but let me go to building blocks API. And I can actually either go to the API reference here at the top or click postman collection. So let me just go to API reference for a second. And then you can see a large list of end points here. These are collapsed. But if I actually expanded each of these folders, I believe we have closed like 375, maybe even close to 400 end points.

at the time of this recording, that number just keeps increasing. And the idea behind publishing this API, much like the other APIs that are part of our API product suite, is to essentially help you as a developer, as an architect, as a product owner, as an entrepreneur, who's building a solution, go to market a whole lot sooner, essentially reducing your time to market. So it's a much smaller number.

Krish (products.snowpal.com) (02:23.937)
even better than the one that you might actually have in mind is our primary purpose. Because short of having these APIs or such APIs being available, you're going to have to have, not to trivialize the problem, but you're going to have to actually have someone build a user interface in your team and another set of developers and team members building the server -side microservices. There's more pieces to the puzzle. I'm just going to trivialize this for ease of conversation.

Let's assume that all that you had to do is two pieces. One is the web app itself and the other one is the API. You're going to have to design, architect design, implement, develop, even staff a team if you don't have one, deploy and maintain all of those things. And again, continually improve your features and do whatnot. In other words, you're going to have to do a lot more than what you actually have to do if you did not leverage APIs. That's the purpose of...

of an API for strategy and if someone else publishes the APIs, the more the merrier. That's where we fit, we come into play as a company. Now let's take an example. Now here you can browse through the API documentation for one of the APIs by going to developers .snowpal .com or you can also actually go, let's go to the homepage here. Let's go to building blocks. You can hit postman collection. It's gonna open a postman collection. I already have that open on the first tab. So this is, yeah.

You know, we've seen that product owners typically like this documentation, the way this is laid out on this page that I'm sharing here, with guides, recipes, API references. Developers tend to

prefer Postman because it makes it much easier to run this in your native Postman client and start developing within the first 10, 15 minutes of you being exposed to this API. That's, you know, all that you'll have to do is go subscribe to this API and you can do that in one of many ways.

We actually sell this in multiple, more than one, we support more than one pricing model. You can license the APIs if you want to have complete control over them. You can just purchase commercial licenses and use our APIs. It's totally fine. That is one of the models we support. Or you could pay by use, like pay per request on AWS Marketplace or by subscription. Those are options. Or if you wanted us to provision these APIs,

Krish (products.snowpal.com) (04:51.361)
in your infrastructure, whether it's AWS or on Google Cloud or Azure, we are happy to do that as well. That way you will have kind of the exact same setup that we do. We have a multi -tenant setup, but you may just have it for yourself, but it runs in your infrastructure and becomes part of your AWS building. Again, there's more than one model we support. You can pick and choose the one that works best for you. Subscribing to the API is very simple. It takes few clicks.

You can look at the purchase options and then after you do those two or three clicks, you would submit and you will actually receive an email within the first five minutes that has an API key and the product code. The product code, because we support more than one product, we have multiple API products as products. So you'll get the product code and the API key and at that point, your dev team should be able to hit the ground running. It's literally as simple as that. Honestly, I'm not exaggerating here.

That's literally what the experience is for our customers. So let's take a look at an app that was built using this API. And then so you get an idea. Now, the app that I'm going to show you is a project management application. If you go to snowpal .com, you can check out the application. Now this is a production account. So it's got production data. Unfortunately, I'm not going to able to share that. So what I'm going to do is run this app locally. And then we'll actually check out how this.

Let me actually sign out of here.

Krish (products.snowpal.com) (06:23.393)
is all test data, so just take the data of the grain of salt. So this is a project management application that was built using the building blocks API. Now what you want to build is entirely up to you. You could be building an application for the pharmaceutical industry or for the restaurant and hospitality industry or for fintech, you know, for the food industry. It honestly doesn't matter. Whatever your problem is, the API should come in real handy. All that you have to do is understand.

our terminologies is going to take you a couple of hours to get your sort of your feet wet. And once you get to understanding our terminologies, you can map them between your business problems and your terminologies that relate to your domain and your problem. And our product terminologies here. And once you do the mapping, you should be good to go. Like what we call a key, which is essentially a project or a board, might be something else for you. So that's literally all that you would have to.

spend that initial time understanding how to present this to your end users, so to speak. With that said, let me show, hopefully I'm recording here, yeah. Let's say this is a project management app and on the first pages you're exposed to is the dashboard. And if I go to the

app here, you see a dashboard directory in that collection. And it's got a bunch of endpoints, it says get dashboard details, get an unread count, recently modified keys, blogs due shortly.

and so on and all of that data helps to render these items here on this particular dashboard. Now I'm gonna close that and go to dashboard chats. There's a bunch of endpoints here for dashboard chats. If I go to chats, it's gonna pull up the chats. Again, this is test data, so you're gonna see real test names like CK1 and CA that don't mean anything but in the real world, like if I go to my production data, I can show you it'll have something to do with taxes, with Snowpal projects.

travel plans and whatnot, whatever your project and your data happens to be. These chart related end points are documented under Dashboard 2. Our naming convention essentially is we have the domain, it basically tells you the feature, a product feature, a horizontal feature if you will. The .1, .2 is we just increment that within the scope of a feature. So for Dashboard there's two directories for.

Krish (products.snowpal.com) (08:47.553)
keys, there's one, two, three, four, five. For key pods is six, for blocks there is like seven. Some of them there could be a couple and others there might just be one. That's the naming convention and then we have these endpoints numbered so it's very easy for you to socialize these endpoints as you work with the rest of your team members. Let's say you're building a React app with a couple of other developers and you have your product manager as part of your team. I mean you're socializing these endpoints and having these conversations.

you can just go point to these numbers. That's the idea behind naming them in such a manner with that pattern, essentially. So those are a couple of directories. Now we can, when you go to keys, if I go, let's say if I go to this page, actually if I go to keys, a key on Snowpal is essentially an entry point. It's a project, it's a board. Let's say if I'm going to travel to, I don't know, let's say a trip to Scotland.

I can create a trip to Scotland and that becomes my entry point to that project, quote unquote. Everything is a project if you look at it that way. It's not just at work, even your personal life, even as we enter, and we are like a month and a half or two months away from the tax season. So you're planning to file taxes, you wanna gather all your tax documents. You could again pick a different key type if you wanted to. You could say taxes 2024.

and then you can go in and you go into taxes, you're gonna add a block here, you could actually say, I don't know, 1040, and then you might have state taxes and whatnot. So that's a block, and then when you go into a block, you'll see that we have the next set of endpoints for blocks. Blocks and keys, they support a notion of attachments and checklists, comments, notes, the fundamental building blocks, so to speak. So if I were to draw this out really quickly, let's say, let's do that.

before we finish this video. Let's say I go here. I'm gonna call it building blocks API. The purpose of this API, much like our other APIs, is to serve as the foundational building blocks. So you can build whatever your building is gonna be, it's left to you. These are the building blocks, and you can think of them as Lego pieces. Creativity or imagination is the limit, or your business problems are literally your limit. You can solve your business problems.

Krish (products.snowpal.com) (11:13.107)
You know, our API is a domain agnostic. So again, like I said earlier, your domain and the industry that you're in honestly doesn't matter, makes no difference. You just need to be able to map your problems to our terminologies and endpoints, and you're good to go. And it's going to save you all the time and money and risk of not having to build, maintain, and improve and

enhance your back end systems and microservices. So here we had building blocks. And I'm going to.

or a couple of more things. Let me pick a different shape here.

We saw a dashboard endpoint, a dashboard directory, sorry. And then we looked at keys. That's like I said, close to like 400 endpoints. We're not gonna go through all of them, not in this initial video. Let's say we just took three things, dashboard, keys and blocks. So I'm gonna draw these arrows here.

Krish (products.snowpal.com) (12:11.169)
And then each of them, like we saw dashboard for instance had two directories, keys has like five directories. Let's actually say dashboard and dashboard charts. I'll just pick a different.

Krish (products.snowpal.com) (12:44.033)
call it dashboard details, I'll just call it.

Krish (products.snowpal.com) (12:55.233)
make the font smaller and I type more and then this is going to be dashboard shots actually let me call it details and shots so those are the two sub directories if you will that we saw in postman postman doesn't support a hierarchy of directories so we just have a flat list here so the next one is keys keys has a number of them details the default I'm just calling it details

and then Shards, Checklists, Notes and Tasks.

Krish (products.snowpal.com) (13:33.441)
tails.

Krish (products.snowpal.com) (13:42.241)
Checklist.

Krish (products.snowpal.com) (13:58.561)
arrows here for the sake of completeness.

Krish (products.snowpal.com) (14:09.057)
All of that should have actually been keys, not blocks. You know what, in the interest of time, I'm just gonna ignore blocks. So we just do two of them, keys. So we saw more than this, but let's say you take, so this is the postman, this is the API documentation layout, API.

Whether it's Postman or the other developer documentation, we went to the API reference. And you see the naming convention is pretty much the same. It's OpenAPI, YAML, or OpenAPI JSON. It's basically OpenAPI specification that we support. So it looks quite identical, no matter where you consume this. If you.

Again, I said product owners because we have guides with more documentation as we continue to update this. There's plenty of articles that speak to our APIs, how you have to approach them, and how you do the mapping and whatnot. And we're starting to write more recipes. So this page, again, developers would use it as well, the developer documentation. But the product owners seem to take liking to it as well. The postman version, I've only seen developers use it. Not to say that product owners may or may not.

use this or like it. It's just in my experience, I've seen that this appeals a bit more to developers because they are able to actually run this locally in the client and start coding. But again, it doesn't matter what your role is. You can figure out what appeals more to you and use that

interface if you know what I mean. So I just want to draw this out for at least for a couple of degrees so you get the idea. So each of them has a number of endpoints. Now we saw details and shards and if you go to details, you see all these other endpoints.

So at this level, you have a bunch of endpoints. I don't want to draw this because there's a number of endpoints. Even if you looked at these two directories that we took, you're going to see like 20, 25 endpoints, or maybe even upwards of that. So if you're building something for the food industry, now again, you may not need all 350 endpoints to implement it. Maybe your application uses a combination of APIs. We'll talk about that in a subsequent video. Or you are.

Krish (products.snowpal.com) (16:21.857)
you know, you have your own API backend teams for all you know, maybe some team, maybe not all of them, or even if you have a larger team, you are better off using your team and resources to building what you need to your problems, to the problems that you're trying to solve, to the solution that you're building, instead of reinventing the wheel because that doesn't add any value. So you're gonna have your backend members leverage our Snowpal APIs for doing what...

they don't have to redo and you're only gonna let them, you know, spend their time meaningfully in building the pieces that we don't offer because it's not generic, it's not fundamental building blocks necessarily, and it's very specific to your business. The rest of it, you can use our endpoints either as is or you do the mapping. Maybe we call it keys and you could call it something else. You might call it restaurants. So you call an endpoint that says get keys or add a key.

But then what you're actually doing is persisting a restaurant and you return it to your client, whether it's a web app. In this case, we're talking a web app. You'll return a list of keys, but then your client, your web app, the React app chooses to render it using a different language. That's the mapping that you would do because it's very specific to your problem. But what it saved you by virtue of using APIs and endpoints eventually is...

You don't have to implement this. You don't have to design any of this. Imagine the work that would be involved if you had to actually implement all of these APIs, or at least the ones you needed, even if it's a smaller subset, and then actually having someone else pick up and work on it in your UI teams. Now, they could do it in parallel by stubbing out these APIs and by stubbing out these endpoints, but still the work needs to be done eventually, and there's a lot of nuances and challenges that it brings to the table. So that's the whole idea.

I think with this, I'm just going to end this video. We talked about our developer documentation as a quick recap. We talked about postman collections. We drew up a quick diagram. We saw the AWS marketplace. So you just go to aws .snowpal .com to actually see all the products we have available. We looked at the building blocks SaaS product in this video. And I also showed you a quick real life example of a product that's in production, used by thousands of customers that's built.

Krish (products.snowpal.com) (18:46.177)
using and relying on this very same API that we looked at here. That's it for this video. Talk to you soon. Bye bye.