EPISODE 1099

[INTRODUCTION]

[00:00:00] JM: WordPress has been a dominant force in the world of online publishing for many years because of how battle tested it is. WordPress is the definitive leader in CMS technology, but there have always been alternatives; Drupal, Ghost and other open source CMSs are alternatives to WordPress. And more recently, there have been the emergence of the headless CMS category such as Contentful, which decouples the CMS backend from the frontend presentation layer.

Strapi is a popular open source headless CMS. Pierre Burgy is the founder of Strapi and he joins the show to talk about the CMS category, the role that Strapi fills and the technology behind Starpi. I hope you enjoy the episode.

[SPONSOR MESSAGE]

[00:00:49] JM: This episode of Software Engineering Daily is brought to you by Datadog, a full stack monitoring platform that integrates with over 350 technologies like Gremlin, PagerDuty, AWS Lambda, Spinnaker and more. With the rich visualizations and algorithmic alerts, Datadog can help you monitor the effects of chaos experiments. It can also identify weaknesses and improve the reliability of your systems. Visit softwareengineeringdaily.com/datadog to start a free 14-day trial and receive one of Datadog's famously cozy T-shirts. That softwareengineeringdaily.com/datadog.

Thank you to Datadog for being a long-running sponsor of Software Engineering Daily.

[INTERVIEW]

[00:01:44] JM: Pierre, welcome to the show.

[00:01:45] PB: Hi, Jeff. Thanks a lot.

[00:01:47] JM: So we're talking about Strapi today, and Strapi is a CMS, or you might say it's a framework. And if we're talking about this domain, we have to talk about WordPress first. WordPress is the dominant CMS. It's been the dominant CMS for a long time. Powers a lot of websites in the Internet. How has the world of the CMS evolved since the early days of WordPress?

[00:02:13] PB: Yeah. WordPress is definitely the leading CMS. It powers 30% of the Internet. And so it's [inaudible 00:02:21] to build websites. But now if you [inaudible 00:02:24] to be displayed not only on the websites, but also in mobile applications, connected devices and even the way to create a website really changed in the past few years, because developers want to use modern technologies such as React, Vue.js, Angular, and all of these technologies are designed to consume the content from APIs and to display it their own way.

In a traditional CMS like WordPress, you really have two things. You have a backend [inaudible 00:02:54] content and a frontend to display it, but we have this new approach, the frontend is really separated from the backend. So it's a new architecture, and the frontend consume the content and the CMS now is useful to [inaudible 00:03:11] content. And instead of having a frontend, it has an API, which make the content available to any platform.

[00:03:19] JM: Strapi is a headless CMS, you might say, as you're talking about this decoupling between the backend and the frontend. Strapi would be, I guess, arguably the frontend, right? But there're also some backend management components to it. I mean, the way I look at Strapi is it's more like you migrate your data sources to it and then you have a decoupled CMS system. Can you just describe the workflow for taking an existing website and porting it to Strapi?

[00:03:54] PB: Yeah, sure. Exactly. That's correct. Strapi is a headless CMS, which means it has no frontend. When you decide to migrate a website from a traditional CMS to a modern stack, which is often called the Jamstacks, which have a script API and markup, which is a new architect, you are going to use a modern technology for the frontend. So it can be React, Vue.js, Angular or any framework built on top of these frameworks. And so you start developing the frontend. And according to the frontend, you are going to configure the headless CMS. If you're

building a blog, for example, [inaudible 00:04:36] that every blog post should have a title, should have content and should have an image. That's just an example. That's the configuration step.

You are going to add some content in your CMS and you are going to configure your frontend to connect it actually to the CMS, to the CMS API. So then every time you update the content in the CMS, it's going to be displayed in the frontend. That's the typical workflow. And then the developer, you will probably give access to the marketing team, for example, to the CMS so they can update the content.

[00:05:15] JM: The data sources that we might have in Strapi, we can define relationships between these data sources to make an inheritance hierarchy across them. Can you define the – Or I guess not necessarily inheritance hierarchy, but relational hierarchy. Can you define the idea of relations between different data types?

[00:05:40] PB: Yeah, sure. If you are building a blog, again, because it's a simple example, you will probably [inaudible 00:05:46] for all your blog posts. Maybe you want to say each blog post kind of have one category or maybe you want each blog post to have several categories. So a CMS gives you the flexibility to define this, and that way you can have a very clear and complex data structure so you can create relationships. That's the basic way to create relations between the content types.

[00:06:15] JM: So if I set up a Strapi website with – Let's say I have a WordPress blog and I migrate it to Strapi, why is that advantageous. What can I get out of the changing of my CMS infrastructure?

[00:06:33] PB: I think it's mostly performances for many reasons. First of all, in your frontend, because you use modern technologies such as React, or Gatsby, your website is going to be much faster because there is no page reload every time the user click on the link. And also, this website [inaudible 00:06:51] static site generator, for example, can be hosted on Netlify or Vercel and it can be available anywhere, thanks to the CDN.

So you will have way better performances. And also, when you go from a page to another, so even if you are not choosing a static website generator, the web [inaudible 00:07:14] is not

going to regress an entire webpage, but only the content. So there is less data in the payload, which make the website much faster. That's definitely the main advantage, but headless CMS is also more secure, because in most of these traditional CMS, the security issues come from the template system because it exposes a lot of functions to the templates. But with a headless CMS, you don't have these template systems, which removes a lot of potential issues.

The last thing is maintainability over time, because even if you want to create a new website in the future and you want to keep the content, with a headless CMS it's really easy, because you keep the content in the headless CMS and you just have to plug your new frontend.

[00:08:05] JM: Okay. And Strapi allows for some flexibility, some configurability with frontend frameworks. Can you describe how Strapi integrates with – Like if I wanted to have a React frontend or a Vue frontend and I want to configure some of the code in my CMS – I mean, historically, WordPress, if I want to design my WordPress templates, there's a little bit of work involved in understanding where the bounds of WordPress and the component architecture that you might want in React. So could you just tell me about the relationship between the frontend component libraries and the CMS infrastructure?

[00:08:55] PB: Yeah, sure. You can basically regress Strapi from any frontend technology and you can do it in any way, because Strapi is just an API. You can use the regressed library of your choice. You can use [inaudible 00:09:11] or any other library. So freedom is really part of our values in the product. So you can do it your own way. That's the first thing.

But if you want to use something like Gatsby, Gatsby is built on a plug-in architecture with different data sources. For example, we do have a plug-in for Gatsby, which makes things really easier. Again, Strapi has an API which is well-documented and it can be used from anywhere. And we support both Rust and GraphQL. So you can use the API of your choice.

[00:09:46] JM: There's a number of these headless CMS stacks that have come up in the last five years or so. What would you say is the differentiator for Strapi? How does it compare to the other popular headless CMS stacks?

[00:10:02] PB: That's correct. There are a lot of headless CMSs out there now, and it's really becoming a thing, and I think it's getting used for the market and it definitely proved that the traditional CMS market is shifting to the headless CMS approach.

So most of the headless CMS are SaaS, so software as a service, which is nice because you can easily sign up and get your product up and running. But when we created Strapi, so it was in 2015, we were working as freelance developers. We did a lot of websites using traditional CMSs. And then we were convinced that if you want to want to make something with a CMS, it totally makes sense to build something with an open source CMS. We made Strapi open source from the beginning, and our traction definitely proves that it's a very good way to reach the headless CMS users. And the reason behind this is that most of the big companies still [inaudible 00:11:03] their CMS on their servers, especially for privacy reasons. So we do have a lot of [inaudible 00:11:09] and insurance issues in Strapi.

The other reason is that developers want to have the ability to customize the CMS, and on the open source can offer such a good way to customize the CMS. That's the two main reasons. So open source, and [inaudible 00:11:27] reasons that Strapi is completely customizable. Thanks to a plug-in system. Strapi is also completely based on JavaScript. So we use NodeJS for the API and React for the dashboard, which means the developers can use the same language for the frontend and the customization of the headless CMS, which is very [inaudible 00:11:47].

Another thing is the flexibility. So having headless CMS, which is very flexible is extremely important, because if you are building a blog, then the content structure is very easy. You have blog posts with data content and so on. But if you are building a corporate website with a homepage and you have a tagline, then some testimonials, then a slider and then a footer. So your content structure is quickly becoming very complex.

We decided to focus on this on Strapi and the other we made, one of the most flexible headless CMS. For example, we introduced the component system in the CMS, which means you have some pieces of content that you can reuse, request different content types. So it could be a slider, for example. So you can say that one component, which is a slider, which include another component, which is a slide. [inaudible 00:12:46] an image and the link and you are going to be able to use the slider the product, in the blog post or anything else. That's extremely important

for complex projects. And we also introduced the dynamics and concepts, because if you are building a webpage, for example, you don't necessarily know what is going to be the structure of your page, because maybe you – Yeah, just for example, when you are building your page, maybe on the fly, you want to have the possibility to say I want to have the slider on that window, I want to add a paragraph, and so on and so on. Thanks to the dynamics zones, you can really build your page on the fly, and that adds a lot of flexibility.

[00:13:29] JM: Give me a little bit more explanation for the Jamstack use case. So like I know when I'm talking to developers, usually they just want to build everything from scratch. If they're building a website and they know how to write react component, they're going to do everything in react. They're going to make all their own component or they're going to take some off the shelf from GitHub, but they are not going to use a CMS. My sense is that this is changing somewhat and that just like people look at Ruby on Rails or people look at VueJS or NextJS, people are comfortable looking at a CMS as basically a framework now. But I'm just trying to understand to what extent developers want to use a CMS as the basis for often times what they're trying to make as a platform.

[00:14:27] PB: Yeah. Developers don't necessarily want to use a CMS. Most of the times, they prefer to add the content directly in the code, which is fine, but if someone else in our team need to update the content, then it doesn't scale, because the marketing team will want to have the possibility to update the content themselves. That's exactly when a CMS makes sense, when you have a group of people who want to manage the content.

In that situation, the developer is going to kind of make dynamic specific portion of all the entire website. Then in this case, it really makes sense to connect the frontend to a CMS. And the good thing is that the developer can still use all of their favorite technologies.

[00:15:17] JM: And the typical customer use case. So take me through a typical customer use case or some of the prototypical customer use cases for people who are working with Strapi.

[00:15:29] PB: Yeah. Sure. So the main use case we have is editorial website. People are going to manage blog posts. Most of the time in a [inaudible 00:15:39] way. But it's typically the

blog post use case. For example, there is an entire section of ibm.com which is powered by Strapi and they do manage content to publish it on their website.

The second use case is corporate website. So it's really the .com website of a company where you are going to manage your content [inaudible 00:16:03] page. And then, for example, at least the features, some testimonials and these kind of things. And the third one is ecommerce website where people are going to manage products in Strapi and display it on their website or mobile application and they will handle the payment system. Most of the time we use something like [inaudible 00:16:24], for example, and that's the three main use cases between – Also, we have some IoT use cases, which is very interesting, and the headless CMS is definitely open new doors for these use cases. So some users [inaudible 00:16:40] mobile application – Not mobile application, but an IoT device for Fireman. And when [inaudible 00:16:47] connected device actually gets some information and it is linked to Strapi.

[00:16:53] JM: Interesting. So the IoT use case, like less of a publishing use case. That's more like a low code application use case.

[00:17:05] PB: Yeah, exactly. But to give you another example, which is really more easier to understand, the MIT decided to build a connected device, which is in this tweet. In case of an earthquake, this connected device is going to spread a message. And the message comes from Strapi.

[00:17:23] JM: Amazing. To that point, you have plugins. You have integrations with Redis, and SendGrid, and Algolia. And the integration workflow for these different plugins, where do they hook in?

[00:17:40] PB: Yeah. We truly believe in the power of plugins, because if you take a look at WordPress, for example, what made the success of WordPress is not WordPress itself, but it's really the ecosystem of plugins around it. So that's part of our strategy from the beginning. And we already have some plugins made by both the team and the community. You can develop your own plugin using internal APIs. And so you can extend both the API, for example, to synchronize your data with Algoria, and also the dashboard. So you can, for example, add an

entire section and add maybe a dashboard to analytics about your content can be something like this.

[SPONSOR MESSAGE]

[00:18:31] JM: I've recently started working with X-Team. X-Team is a company that can help you scale your team with new engineers. X-Team has been helping me out with softwaredaily.com and they have thousands of proven developers in over 50 countries ready to join your team and they can provide an immediate positive impact and lets you get back to focusing on what's most important, which is moving your team forward.

X-Team is able to support a wide range of needs. If you need DevOps, or mobile engineers, or backend architecture, or ecommerce, or frontend development, X-Team can help you with what you need. They've got a full-range of technologists who can help with AWS, and Go lang, and Shopify, and JavaScript, and Java. Whatever your engineering team needs to get to the points of scale that you want to get to, X-Team can help you grow your team. They offer flexible options if you're looking to grow your team efficiently, and their model allows for seamless integration with companies and teams of all sizes. Whether you're a gigantic company like Riot Games, or Coinbase, or Google, or if you're a tiny company like Software Daily. You can get help with the technologies that you need. If you're interested, you can go to x-team.com/sedaily. That's x-team.com/sedaily to learn about getting some help with your engineering projects from X-Team.

Thank you to X-Team for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[00:20:14] JM: The decoupling of the frontend and the backend for this kind of headless CMS infrastructure, at least to a well-defined architecture, rather than a monolithic architecture, you've got better separation of concerns, I think. Can you tell me about the architecture of Strapi itself?

[00:20:36] PB: Yeah, sure. Strapi is based on NodeJS. So it's actually a simple NodeJS application, which can easily be deployed on any cloud or any platform as a service like Heroku. We decided to use Quora as the main Framework. And Strapi is database agnostic, which means you can connect it to MySQL, Postgres, MongoDB, or SQLite, or even develop your own connector.

Regarding the dashboard, Strapi is based on React. So we did our own web pack configuration, and that's the main technology. And we also developed our own React component library so people can develop their own plugin using the same design.

[00:21:23] JM: In Strapi, there is content delivery that happens over an API, and Strapi mainly has the content API. But there are two separate content, the collective and the single content type. Can you explain the difference between those two content types?

[00:21:40] PB: Yeah, sure. If you're building a blog or an ecommerce website, you are going to manage lists of products, lists of blog posts. So that's typically a correction type. Most of the time, our users choose to go with a correction type. But if you're a building a corporate website and you have a homepage, for example, then you are going to have only one homepage on your website.

So you are going to use a single type, which means this content isn't repeatable. So you can't have [inaudible 00:22:14] home pages. In this specific content type, you are going to say the structure is first a tagline and then a slider and so on and so on. This structure would be only for this specific content.

[00:22:27] JM: Got it. What' the typical deployment model for Strapi?

[00:22:33] PB: There are tons of possibilities, and we only see different options according to the company using Strapi. We do have a lot of users deploying on Heroku, because it's really easy. But we also see some companies developing on Kubernetes, because Strapi can be Dockerized. So there are really a lot of options.

We do recommend a few options. We do have some guides, but at the end, anyone can do what they want.

[00:23:03] JM: Am I typically deploying it in a Docker container or on an EC2 instance or are there any recommendations, or do you have a hosting model that if I'm using your company to deploy it, you have a hosting model that you can describe?

[00:23:22] PB: So we don't provide any hosting model at the moment. We will probably add this in the future to make our users life easier. But at the moment, we just provide some guides, about one guide for each main provider. So AWS, GCP and Microsoft Azure. So that it's for the moment, but we'll see what happens in the future.

[00:23:43] JM: So Strapi is built on Node. Do you use any other frameworks? Are there any other open source projects that are key in Strapi?

[00:23:53] PB: Yeah. The framework we use is Quora, which is very similar to Express. We use Bookshelf as [inaudible 00:24:00] to regress the SQL databases, and we use Mongo for MongoDB. So those are the main libraries. We also use [inaudible 00:24:09] for testing.

[00:24:12] JM: How big is the team?

[00:24:13] PB: We are 17 in the team.

[00:24:15] JM: Okay. And what are the priorities? The engineering priorities right now across the company?

[00:24:21] PB: Yeah. We raised 10 million last week. So we do have some priorities for the next 18 months. First of all, we are going to focus on the community. We are community and product-driven from the beginning. So we really want to make the best open source headless CMS and we'll continue to do this, because this is really the impact we want to have.

Then we are going to start setting the enterprise edition, which is actually an extension of the community edition. And this enterprise edition will include some advanced features, especially

for marketing teams. Typically it will include Word base access control, single sign on, content internationalization and contribution workflow. So that's a new challenge for us, because until now we didn't do any revenue. But we have lots of requests for all of these features. So we are confident.

The last one is [inaudible 00:25:22] the ecosystem, because as we said earlier, plugins are very important in the CMS. So we are going to improve our plugin system to make super easy for anyone to develop a Strapi plugin, and we will also provide marketplace so anyone can share their own plugins and then with plugins from the community, and we are sure that we will see very impressive plugins and the product will go faster that we contribute to it. So that's going to be exciting.

[00:25:58] JM: What's the business model at this point?

[00:26:00] PB: We don't have any.

[00:26:01] JM: Okay. Right. So it's like hopefully you want to get to hosting eventually, I assume.

[00:26:07] PB: Yeah. The first business model will be the enterprise edition with all of these advanced features. And later, we will probably have the hosting.

[00:26:15] JM: Got it. Yeah, I guess that makes sense, because the enterprise model, that's probably a more profitable or more lucrative business at least in the short term.

[00:26:24] PB: Yes. And there are lots of SaaS out there. So we don't have that much value if we provide a hosted version of Strapi. There are lots of other solutions for this. So we really want to focus on our users and we have tons of requests for all of these advanced features and we really want to make what our users want. And I think that's probably the best way to develop the product.

[00:26:49] JM: That is intelligent. So the people who are using Strapi, they're not all developers. You have many content creators. You have non-technical people who interact with CMSs. How

does that affect you as a developer of a system? What design decisions do you make to accommodate non-technical users?

[00:27:10] PB: That's super interesting. From the beginning, we target developers just because we were developers at the beginning. So it was just the best way to target this audience. And then we actually realized that developers have a very big impact in the decision process when a company ask to [inaudible 00:27:30] headless CMS.

Most of the time, it's inside a team, which is managed by a product manager. So we are to interact with a product manager. Then when the project is done, the developers give access to the content editors. Currently, we still focus on developers. That's what we do. It's a bottom-up approach. Then we also see a lot of product managers contacted us for all of these enterprise features, and also regarding – For questions related to content editors. So that's the typical workflow. Developers first, then product managers, and then content editors.

[00:28:10] JM: Got it. The static site generators, like Gatsby and Jekyll, these have also become popular. What's the build path for a Strapi instance? Does it get to a completely static site or is there some dynamic element that still have some latency associated with it?

[00:28:33] PB: Yeah. Strapi is dynamic. It's a NodeJS application. You cannot build it like a static website generator. So you can deploy it on Heroku, for example, but not on Netlify, because Netlify doesn't support the NodeJS application. Also, you need a database for Strapi. So there are some [inaudible 00:28:52] based CMSs, but the problem with that, it doesn't really scale because when you start doing having lots of content and lots of relations between content, then it's not really good even if it can be hosted on a static hosting provider. So that's why we decided to go dynamic with Strapi.

[00:29:11] JM: Just to revisit the question of other CMSs. When you look at Strapi and its architecture and the differentiation from the other CMS platforms, what do you think you've done right? What has differentiated you the most from those other platforms?

[00:29:31] JM: I think we really made something for developers. Again, they have a very impact in the decision process. So we really did something which is very good for developers. We really

focused on the developer experience. And so that's why we have built such a big community. [inaudible 00:29:48].

[00:29:49] JM: Tell me more about the engineering of the backend. So what is the architecture like and what are the problems that you're working on right now?

[00:30:00] PB: Yeah. So on the backend, the main challenges are definitely the plugin system. So we do have built very scalable plugin system. So anyone can use internal APIs. That's quite a challenge. And we also did lots of improvements in the last few weeks for the stable release of Strapi. So we drastically reduced the number of files in the Strapi project. So it's much more lightweight and it makes updates easier, because when we released the version of Strapi, it has to be easy to update your current project. So the next challenge is I think would be to make things even more simple. Again, it's really about deploying in system.

[00:30:46] JM: Tell me more about the plugin system. How do you create the right ecosystem? The right hooks into a plugin ecosystem?

[00:30:55] PB: I think you have to start from the zero and manage it like a product, which means you start with the need from users. And so for example, someone is going to say, "I need to build this plugins. So I need to plug to Strapi specifically at this moment." For example, it can be before a content type is created.

So the developer is going to ask you this specific hook, and only if someone ask you this, you have to build it. Then you have to document it first, I think. So you can do documentation first and development. And then you are going to develop this specific hook.

[00:31:34] JM: And what are some examples of plugins and the architecture of the plugins?

[00:31:40] PB: Yeah. If you want to build, for example, a plugin for Algolia, you need to know that every time a content is created, updated or deleted. You need to get the type of this content. So you need to know that post has been updated. Then you need the content of this post. So you need to title the content and so on. At this specific moment, you are going to synchronize with Algolia. So that's some hooks you would need in this example. So that's a very

backend-oriented plugin, because you only manage content. But then if you want, for example, to add specific things in the dashboard of Strapi, that's another thing.

One thing we are going to focus on in the next few month, we have the custom fields. So that's where you don't only have access to the default fields, such as text, image and so on, but you can also add a custom field such as Shopify. In that way, in Strapi, you can select a product, which is actually from the Shopify API and you just don't realize it's coming from someone else. You just select the product, and in Strapi, it's going to save the Shopify product ID, and that's it. That's something we really want to focus on. Another example for this is the [inaudible 00:33:06] on field to have some kind of maps. So in a content, you can select a specific location. That will be from a custom field.

[SPONSOR MESSAGE]

[00:33:24] JM: You probably do not enjoy searching for a job. Engineers don't like sacrificing their time to do phone screens, and we don't like doing whiteboard problems and working on tedious take home projects. Everyone knows the software hiring process is not perfect. But what's the alternative? Triplebyte is the alternative.

Triplebyte is a platform for finding a great software job faster. Triplebyte works with 400+ tech companies, including Dropbox, Adobe, Coursera and Cruise Automation. Triplebyte improves the hiring process by saving you time and fast-tracking you to final interviews. At triplebyte.com/sedaily, you can start your process by taking a quiz, and after the quiz you get interviewed by Triplebyte if you pass that quiz. If you pass that interview, you make it straight to multiple onsite interviews. If you take a job, you get an additional $1,000 signing bonus from Triplebyte because you use the link triplebyte.com/sedaily.

That $1,000 is nice, but you might be making much more since those multiple onsite interviews would put you in a great position to potentially get multiple offers, and then you could figure out what your salary actually should be. Triplebyte does not look at candidate's backgrounds, like resumes and where they've worked and where they went to school. Triplebyte only cares about whether someone can code. So I'm a huge fan of that aspect of their model. This means that they work with lots of people from nontraditional and unusual backgrounds.

To get started, just go to triplebyte.com/sedaily and take a quiz to get started. There's very little risk and you might find yourself in a great position getting multiple onsite interviews from just one quiz and a Triplebyte interview. Go to triplebyte.com/sedaily to try it out.

Thank you to Triplebyte.

[INTERVIEW CONTINUED]

[00:35:41] JM: You mentioned this use case of IBM earlier, and I think of IBM as quite a large company. Probably has a lot of legacy CMS systems internally. It probably has a lot of legacy data sources internally. What's the process for such a legacy company setting up Strapi and migrating all their existing resources to being visible to Strapi?

[00:36:11] PB: Yeah. Most of the time, big companies start with very small projects and all sections of their website. So they never start with a very, very big project. We really want to battle test Strapi in production and then to decide to go to a bigger project. If they want to migrate, most of the time they have to build their own scripts, because they use very old CMSs. So we are not going to refer these kinds of scripts for the old CMSs. Probably in the future we will provide importers, which means you can import content for medium, for example, but definitely not from very old services.

[00:36:55] JM: Got it. Coming back to the plugin ecosystem actually, you mentioned Shopify. You've mentioned Algolia. One can imagine building a pretty richly featured website out of these different kinds of components. It kind of brings to mind, again, the low code platforms where you can have this kind of drag and drop interaction to construct websites. How do you compare the burgeoning Jamstack CMS ecosystem to the low code ecosystem?

[00:37:29] PB: I think the headless and Jamstack ecosystem is going low code, because we see more and more easy to use projects. And I think the headless, the Jamstack – actually, I did a lot of complexity especially because it's only for custom projects. It's only for developers. But the reality that in a few years, you would use headless CMS and you want to know it. Because I think you will have the possibility to pick a design. So a website design, which can be a

[inaudible 00:38:02] website or a blog post or anything else. And then design would be automatically connected to a headless CMS and maybe the headless CMS of your choice. And that way you will have a completely open running website connected to the Atlas CMS, maybe also connected to Algolia or anything else, and it will be deployed on Netlify or Vercel, and you will just have to select some component and you will use the internal Jamstack and without any code. And maybe you want to know that you will use a headless CMS.

If you take a look at the [inaudible 00:38:42], typically you have very no code solutions such as WebPro. And at the opposite, you have headless CMSs. I think that the headless CMSs will connect more and more to things like Webflow. And also Webflow, we go more and more to headless CMSs, because headless CMSs have tons of possibilities. And it's really complex to build a headless CMS. So maybe in some use cases, it would make sense to use native headless CMS instead of the CMS from Webflow or no solutions. So I really look forward to seeing what will be the evolution in this ecosystem.

[00:39:26] JM: Regarding the static side deployment, obviously you said it was dynamic. Strapi is dynamic and so it's not deployable to Netlify, for example. Can you see a path towards getting to static deployments? Would that be desirable?

[00:39:45] PB: For Strapi?

[00:39:45] JM: Yes.

[00:39:47] PB: I don't think so. Because it would be desirable if you have a project which is very simple. But as soon you have a project which is complex, you have lots of parameters when someone request an API. So it's never the same content. I definitely think it's good to [inaudible 00:40:06] the frontend and host it on a static hosting provider. But I think the CMS has to stay dynamic. It's really easy to host something like Strapi. It scales very well both horizontally and vertically. And if you want to cache the content, you can just use a CDN on top of the CMS, and that way you have kind of a static CMS.

[00:40:35] JM: Got it. In the users of the ecosystem, do you see any bias towards React of Vue?

[00:40:43] PB: No. We have users from every community. So I think the Jamstack in general is not focused on a specific technology, and that's good.

[00:40:56] JM: Tell me a little bit about the Journey to creating Strapi, the early days and the – I think you've been at it for longer than 6 years, right?

[00:41:06] PB: Yeah, 5 years actually. So we started Strapi in 2015. We actually were students. We are three cofounders. We worked as freelance developers. We used a lot of traditional CMSs. But yeah, we know we had to do a mobile application or a website using all of these [inaudible 00:41:22] technologies. It was Angular at the time. It was not a great feat.

So we decided to build something for our customers' projects. We started using it. And at the end of 2015, we just say, "Well, probably not a lot in this situation. So let's push the code in GitHub and let's see what happens." We started [inaudible 00:41:45] lots of users. We are still students, so it was so very exciting. We spent our nights and weekends on Strapi to maintain it as much as we could. At the end of university, we decided to create the company behind the open source project and to switch fulltime in it.

We started at the web agency, because we didn't have cash. We had to make money. So we did a lot of websites using Strapi. Then we actually did our first [inaudible 00:42:16] one in 2018. Then there's one in 2019 and there's one in 2020. It allowed us to stay fulltime on the project, which I think very important, because when you work as a freelancer or as a web agency, the clients are kind of a trap, because you have to build very custom things and you actually lose focus on the product. So I think it's really important to grab the possibility to switch more time on it.

[00:42:47] JM: And how did that process of building customer-facing websites with your own framework, how did that treat you as a way of bootstrapping the product development?

[00:43:00] PB: Yeah. Actually, I think we just didn't know we are using Strapi at the beginning. Just at the end of the project, we give access to these clients. So this is a CMS. And it was fine. So that's how we got started.

[00:43:16] JM: The customer didn't really know the difference.

[00:43:18] PB: Not really, and even some customers wanted to stop using WordPress. So it was actually a good opportunity for us. Sometimes we just said at the beginning that our only constraint is that we want to use Strapi and because we were the maintainers. They were okay with this.

[00:43:35] JM: How much inspiration did you take from WordPress's user model and WordPress's permission model and the whole UI and the workflow of WordPress?

[00:43:46] PB: Yeah. WordPress is definitely inspiring. We are very impressed by what they did, especially being completely open source, building this marketplace and having such a big community and creating a [inaudible 00:44:00] system of thousands of companies building things on top of WordPress, such as template, plugins. That's really impressive and it's really inspiring, because it's very complex to do. I think a marketplace, you never know what is going to be published into the marketplace. But at the same time, you have to open it. So that's something which is I'm very interested.

I think there's quite a big gap between the community edition of WordPress and the .com website, the wordpress.com. So it's definitely not the same target. It's probably a good idea for them, but I think it's also a risk because they don't get a lot of value from their open source community. That's something we'll try to do [inaudible 00:44:46].

[00:44:49] JM: So the journey at this point looks like you just raised $10 million. You got 17 people. You're building access management tooling and you are hopefully, at some point, going to get to hosting. Give me a little bit more on the tactical positioning where you're at right now and how you've aligned the company towards the goals that you're trying to accomplish.

[00:45:18] PB: Yeah, sure. Our first priority is really to where the usage, because this is what will grow revenue and our impact and everything else. So to include the size of Kubernetes, so we really take care of them. We provide great support and documentation. We also try to be

very active on our GitHub repository. So we answer to every issue. We have pull requests. That's sort of the front, but it's really a good way to build such a big community.

Another thing is that we publish a lot of tutorials. So we are not the first to do this, but it definitely works well. So we provide tutorials such as how to build a blog with Strapi and Gatsby. So that way we do some kind of partnerships with you documenting this. Kubernetes is our first focus. Then the other thing, so in the enterprise side, that's also a very [inaudible 00:46:08] thing that we have to do in the next few months. It's really to make sure that we can convert, not everyone of course, but a sufficient number of companies from the community edition to the enterprise edition. So we are going to add a pricing page on our website, and I think we'll build stage pipeline and so people can subscribe and [inaudible 00:46:31] of these features. That's going to be quite a big challenge, but that's going to be a very interesting to see lots of companies switching to the enterprise edition and to make the project sustainable.

Another thing that we are also going to increase or present in the US, because as you may know, we started Strapi in France. We already have an employee in San Francisco, who is our VP of marketing. And we are going to increase the number of people in the team in the US.

[00:47:04] JM: Just to wrap up, I was thinking about WordPress, and one of the things that has made WordPress WordPress is the vast landscape of plugins. I understand you've built this plugin system. There are hooks into the Strapi system. WordPress really has the whole like sales model. They have this whole marketplace. Is the vision for the Strapi system similar to that marketplace model?

[00:47:33] PB: Yes. It's part of our business model, and the first features in the enterprise edition will be developed from us. Then we really would like to go as soon as possible to the marketplace plugin. Not only for the revenue, but really because we'd encourage people to create plugins with high quality and it will really make Strapi much more powerful. So that's something we really would look forward to do. Magenta is maybe a better example for the marketplace. So, yeah, we have different [inaudible 00:48:08].

[00:48:10] JM: Understood. Well, Pierre, thank you so much for coming on the show. It's been really great talking to you.

[00:48:14] PB: Thanks to you Jeff. Thanks a lot for the time.

[END OF INTERVIEW]

[00:48:25] JM: JFrog Container Registry is a comprehensive registry that supports Docker containers and Helm chart repositories for your Kubernetes deployments. It supports not only local storage for your artifacts, but also proxying remote registries and repositories and virtual repositories to simplify configuration.

Use any number of Docker registries over the same backend providing quality gates and promotion between those different environments. Use JFrog Container Registry to search the custom metadata of the repositories. You can find out more about JFrog Container Registry by visiting softwareengineeringdaily.com/jfrog. That's softwareengineering.com/jfrog.

[END]