GUILLERMO RAUCH

LENNY'S PODCAST

BILINGUAL TRANSCRIPT

ORIGINAL BY

Lenny Rachitsky

@lennysan • x.com/lennysan

ANALYSIS BY

@Penny777 • x.com/penny777

Generated: 2026-01-16

Guillermo Rauch - 双语对照

This is the complete bilingual transcript for **Lenny's Podcast** featuring **Guillermo Rauch**, CEO of Vercel.

[00:00:00] Guillermo Rauch

English:

One of our users yesterday submitted feedback. They were saying, "v0 is like a super genius five-year-old PhD with ADHD." I'm not going to oversell this. It knows everything about everything, but it has these sparks of brilliance.

中文翻译:

昨天我们的一位用户提交了反馈。他们说：“v0 就像一个患有多动症（ADHD）、拥有博士学位的五岁超级天才。” 我不想夸大其词。它确实无所不知，而且总能迸发出天才般的火花。

[00:00:00] MUSIC

(instrumental music)

（器乐乐曲）

[00:00:14] Lenny Rachitsky

English:

How do you think things are going to change for product managers, for product teams?

中文翻译:

你认为产品经理和产品团队的情况会发生怎样的变化?

[00:00:18] Guillermo Rauch

English:

People could be more full stack. Imagine a designer that can ship a fully baked product, a product manager that can prototype and ship to production. We shouldn't put limits on ourselves and what we can build, and what we can ship, and what we can dream about making possible on these web surfaces.

中文翻译:

人们可以变得更加“全栈”。想象一下，一个设计师可以交付一个完整的产品，一个产品经理可以制作原型并将其发布到生产环境。我们不应该给自己设限，不应该限制我们能构建什么、能交付什么，以及我们梦想在这些网页界面上实现什么。

[00:00:34] Lenny Rachitsky

English:

A lot of people are wondering, "What happens to engineers? Should I learn how to code?"

中文翻译:

很多人都在想：“工程师会怎么样？我还需要学习编程吗？”

[00:00:37] Guillermo Rauch

English:

A lot of the programming jobs to be done that used to be specializations, I think, are going away, in a way. They're translation tasks, but knowing how things work under the hood is going to be very important for you because you're going to be able to influence the model and make it follow your intention a lot better.

中文翻译:

我认为，很多以前属于专业分工的编程任务在某种程度上正在消失。它们本质上是翻译任务，但了解底层运作原理（under the hood）对你来说仍然非常重要，因为这样你才能影响模型，让它更好地遵循你的意图。

[00:00:52] Lenny Rachitsky

English:

We hear this word taste all the time, in terms of building taste, people are always like, "How the hell do I do that?"

中文翻译:

我们经常听到“品味”（taste）这个词，在培养品味方面，人们总是会问：“我到底该怎么做？”

[00:00:57] Guillermo Rauch

English:

Taste, sometimes I think we think of as this inaccessible thing that, "Oh, that person was born with taste." I see it as a skill that it can develop. I think is extremely important to try lots of products. We have one of our internal operating principles as increasing exposure hours. Try to quantify how much time you expose yourself to watching how people use your products and you'll develop that muscle.

中文翻译:

品味，有时我们认为它是某种遥不可及的东西，觉得“噢，那个人天生就有品味”。但我认为它是一种可以培养的技能。我觉得尝试大量产品极其重要。我们的内部运营原则之一是增加“暴露时长”（exposure hours）。试着量化你花在观察人们如何使用你产品上的时间，你就会练就这种肌肉记忆。

[00:01:25] Lenny Rachitsky

English:

Where do you think the biggest change is going to happen?

中文翻译:

你认为最大的变化会发生在什么地方?

[00:01:26] Guillermo Rauch

English:

We need to stop talking about AI at some point. I just see a future where AI becomes synonymous with software. We build software and we use software to build software.

中文翻译:

在未来的某个时刻，我们需要停止谈论 AI。我预见 AI 将成为软件的代名词。我们构建软件，并使用软件来构建软件。

[00:01:38] Lenny Rachitsky

English:

Today my guest is Guillermo Rauch. Guillermo is the founder and CEO of Vercel, which, amongst other things, makes a product called v0, which has become one of the most popular AI website building tools in the world. He's also a legendary engineer and contributor to open source. He's created some of those popular JavaScript frameworks in the world like Next.js and Socket.IO. He's both a builder and is building a product that's going to change the way we all build products in the future. This episode is incredible. If you want to really understand how product development is going to change with the rise of AI and what skills you should be focusing on right now, I highly recommend you keep listening.

中文翻译:

今天的嘉宾是 Guillermo Rauch。Guillermo 是 Vercel 的创始人兼 CEO，Vercel 旗下的产品 v0 已成为全球最受欢迎的 AI 网站构建工具之一。他也是一位传奇工程师和开源贡献者，创建了 Next.js 和 Socket.IO 等全球流行的 JavaScript 框架。他既是一位构建者，也在构建一个将改变我们未来产品开发方式的产品。这一集内容非常精彩。如果你想真正了解产品开发将如何随 AI 的兴起而改变，以及你现在应该关注哪些技能，我强烈建议你听下去。

[00:02:14] Lenny Rachitsky

English:

If you enjoy this podcast, don't forget to subscribe and follow it in your favorite podcasting app or YouTube. Also, if you become a yearly subscriber of my newsletter, you get a year free of Linear, Notion, Superhuman, Perplexity Pro, and Granola. Check it out at lennysnewsletter.com. With that, I bring you Guillermo Rauch.

中文翻译:

如果你喜欢这个播客，别忘了在常用的播客应用或 YouTube 上订阅和关注。此外，如果你成为我时事通讯（newsletter）的年度订阅者，你可以免费获得一年份的 Linear、Notion、Superhuman、Perplexity Pro 和 Granola。请访问 lennysnewsletter.com 查看。下面，让我们欢迎 Guillermo Rauch。

[00:02:34] Lenny Rachitsky (Sponsor: WorkOS)

English:

This episode is brought to you by WorkOS. If you're building a SaaS app, at some point your customers will start asking for enterprise features like SAML authentication and SCIM provisioning. That's where WorkOS comes in, making it fast and painless to add enterprise features to your app. Their APIs are easy to understand so that you can ship quickly and get back to building other features. Today, hundreds of companies are already by WorkOS, including ones you probably know like Vercel, Webflow and Loom. WorkOS also recently acquired Warrant, the fine-grained authorization service. If you're currently looking to build role-based access control or other enterprise features like single sign-on, SCIM, or user management, you should consider WorkOS. It's a drop-in replacement for Auth0 and supports up to one million monthly active users for free. Check it out at WorkOS.com to learn more. That's workos.com.

中文翻译:

本集节目由 WorkOS 赞助。如果你正在构建 SaaS 应用，客户迟早会要求 SAML 身份验证和 SCIM 配置等企业级功能。这就是 WorkOS 的用武之地，它能让你快速、无痛地为应用添加企业功能。他们的 API 易于理解，让你能快速交付并回到核心功能的开发中。如今，数百家公司已在使用 WorkOS，包括你可能熟悉的 Vercel、Webflow 和 Loom。WorkOS 最近还收购了细粒度授权服务 Warrant。如果你正打算构建基于角色的访问控制（RBAC）或其他企业功能，如单点登录（SSO）、SCIM 或用户管理，你应该考虑 WorkOS。它是 Auth0 的即插即用替代方案，并免费支持多达 100 万月活跃用户。访问 WorkOS.com 了解更多信息。

[00:03:52] Lenny Rachitsky (Sponsor: Vanta)

English:

This episode is brought to you by Vanta. When it comes to ensuring your company has top-notch security practices, things get complicated fast. Now you can assess risk, secure the trust of your customers, and automate compliance for SOC 2, ISO 27001, HIPAA, and more with a single platform, Vanta. Vanta's market-leading trust management platform helps you continuously monitor compliance alongside reporting and tracking risk. Plus, you can save hours by completing security questionnaires with Vanta AI. Join thousands of global companies that use Vanta to automate evidence collection, unify risk management, and streamline security reviews. Get $1,000 off Vanta when you go to vanta.com/lenny. That's V-A-N-T-A.com/lenny. Guillermo, thank you so much for being here. Welcome to the podcast.

中文翻译:

本集节目由 Vanta 赞助。要确保公司拥有顶级的安全实践，事情很快就会变得复杂。现在，你可以通过 Vanta 这一个平台来评估风险、赢得客户信任，并自动完成 SOC 2、ISO 27001、HIPAA 等合规认证。Vanta 市场领先的信任管理平台可帮助你持续监控合规性，同时报告和跟踪风险。此外，你还可以通过 Vanta AI 完成安全问卷，节省数小时的时间。加入全球数千家使用 Vanta 自动收集证据、统一风险管理并简化安全审查的公司吧。访问 vanta.com/lenny 即可获得 1,000 美元的优惠。Guillermo，非常感谢你能来，欢迎来到播客。

[00:04:50] Guillermo Rauch

English:

Thanks for having me. Longtime listener, first time, I guess participating in the podcast, and love being here.

中文翻译:

谢谢邀请。我是老听众了，这应该是第一次正式参与播客，很高兴来到这里。

[00:04:57] Lenny Rachitsky

English:

Oh, I appreciate that. Okay, I know you saw this, I did this survey recently where I asked my readers, "What tools do you use most in your day-to-day work as a product builder, as or product manager?" And in the category of engineering tools, v0 came in right below Cursor and GitHub for people's most used AI building tools. So clearly people love what you're doing.

中文翻译:

噢，非常感谢。我知道你肯定看到了，我最近做了一个调查，问读者：“作为产品构建者或产品经理，你在日常工作中最常使用哪些工具？” 在工程工具类别中，v0 在人们最常用的 AI 构建工具中排名仅次于 Cursor 和 GitHub。显然，大家非常喜欢你们的产品。

[00:05:18] Guillermo Rauch

English:

Yeah, we're very happy to see that. And for us, we're at the very beginning of the journey in some ways, because v0 is a relatively new tool, but for Vercel, our company has been around for a while. The way that I explain to people is, "Anytime you're using the internet, if there's a website or web application that's really fast, innovative, hopefully it's running on our platform." We're out there. We are running a lot of websites at scale. If you watched the Super Bowl recently, three different companies were promoting digital products that were built and delivered on Vercel. So not only can you deploy your ideas and build them on Vercel, they can scale to huge volumes of traffic and huge audiences.

(00:05:58):

So a lot of people know us because of a framework called Next.js. It's an open source framework based on the React technology, open source by Meta, and it powers some of the most innovative products on the internet. So when you use Claude, or Grok, or Midjourney, you're using Next.js. You're using Vercel's technologies. So with v0, what we're trying to do is, and it's funny, because you put us rightfully, I think, in the building or development category in that survey, but what we're trying to do with v0 is help more people participate in building software, increase the total addressable market of people that are actually shipping things, shipping real products. And at the same time, just like you would with ChatGPT, we want v0 to be just extremely, extremely easy, and the outputs that it generates, make them as refined and realistic as possible. The things that you created with v0 hopefully live up to that standard set by some of the best and largest websites on the internet.

中文翻译:

是的，我们很高兴看到这一点。对我们来说，在某种程度上我们才刚刚起步，因为 v0 是一个相对较新的工具，但 Vercel 这家公司已经存在一段时间了。我向人们解释的方式是：“每当你使用互联网时，如果有一个网站或 Web 应用非常快、非常有创意，希望它是在我们的平台上运行的。” 我们无处不在，承载着大量大规模的网站。如果你最近看了超级碗，有三家不同的公司在推广基于 Vercel 构建和交付的数字产品。所以，你不仅可以在 Vercel 上部署和构建你的想法，它们还可以扩展到巨大的流量和受众规模。

(00:05:58):

很多人知道我们是因为一个叫 Next.js 的框架。它是一个基于 React 技术（由 Meta 开源）的开源框架，驱动着互联网上一些最具创新性的产品。当你使用 Claude、Grok 或 Midjourney 时，你就在使用 Next.js，就在使用 Vercel 的技术。对于 v0，我们想做的是——很有趣，因为你在调查中正确地将我们归类为构建或开发类别——但我们通过 v0 真正想做的是帮助更多人参与到软件构建中，增加那些真正交付产品的人的市场规模（TAM）。同时，就像你使用 ChatGPT 一样，我们希望 v0 极其简单易用，并且它生成的输出尽可能精致和真实。希望你用 v0 创建的东西能达到互联网上那些最好、最大的网站所设定的标准。

[00:07:04] Lenny Rachitsky

English:

I was going to ask you how v0 came out of Vercel, and my theory was it was like you guys are sitting around being like, "How do we get more people building websites?" And it's like, "Okay, let's just help them do it really easily." It's like TAM expansion for Vercel. Is that?

中文翻译:

我正想问你 v0 是如何从 Vercel 诞生的，我的理论是，你们当时坐在一起想：“我们如何让更多人构建网站？”然后觉得：“好吧，那就帮他们极其轻松地完成这件事。” 这就像是 Vercel 的市场规模（TAM）扩张。是这样吗？

[00:07:19] Guillermo Rauch

English:

In some ways what I've been doing for not only 10 years that I've been almost working on Vercel, but maybe my entire life because my strength as a developer is kind of meta. It's been to create developer tools. So I've created a bunch of open source frameworks that are really popular. So Next.js is one, but before that in a previous life, I created another tool called Socket.IO, which is a real-time communication mechanism that powers, for example, every time you use Notion, I think you interviewed Ivan, when Notion is to broadcast messages in real time to other collaborators, they use a real-time engine that I built for Socket.IO.

(00:07:59):

So the reason that startups and companies have used my products in the past is because I took something that was very difficult to do, but very compelling. It was with real-time in the past. It's building cutting-edge applications on the web with Next.js. And I try to make it as easy as possible. But you still needed to know development skills. For us and the opportunity was if there is maybe five million React developers, which is the the library engine that we use, and there's maybe 20 million JavaScript developers, how many product builders are people with aspirations of building products exist? My back of the napkin, minimum calculation is a hundred million.

(00:08:45):

And I'll tell you, it's funny where I get that number from. Slack has about a hundred million monthly active users. And what you do on Slack is you go in IT and you talk to people. A lot of those people are building digital products. And they talk to one another about what they would want to see in the world. They talk to customers through shared channels. I love that feature. We talk to a lot of the Vercel customers and they tell us, like, "I want to build this, I want to see that. I want this feature, I want that thing." So the opportunity with v0 was, it's not that you're going to stop talking to other people, but what if you could yap into the computer and see something happen, build a prototype, build your first version of a product, build a demo, build a full stack product, build it and ship it?

(00:09:30):

And so the inspiration for it was very natural to the mission of Vercel. But concretely, the genesis, the story was when ChatGPT came out, we noticed that it was very good at writing the code that our tools used. So ChatGPT, right out of the bat, was good at JavaScript, was good at Tailwind, which is a CSS styling technology, was good at Next.js, and again, the power of open source. Our tools were already in the training data of the internet. And so that long-term bet and vision in open source really paid off. So

because the models were so good at writing this kind of code, the idea for v0 came naturally from, "What if we could build a ChatGPT for building web products?"

中文翻译:

在某种程度上，这不仅是我在 Vercel 工作的近 10 年里一直在做的事，甚至可能是我这辈子都在做的事，因为我作为开发者的强项在于“元层面”（meta），即创建开发者工具。我创建过很多非常流行的开源框架。Next.js 是其中之一，但在那之前，我创建了另一个工具叫 Socket.IO，这是一种实时通信机制。例如，每当你使用 Notion 时（我想你采访过 Ivan），当 Notion 需要向其他协作者实时广播消息时，它们使用的就是我为 Socket.IO 构建的实时引擎。

(00:07:59):

初创公司和大型企业过去使用我的产品，是因为我把一些非常困难但又非常有吸引力的事情变得简单了。过去是实时通信，现在是用 Next.js 构建前沿的 Web 应用。我努力让它变得尽可能简单。但你仍然需要具备开发技能。对我们来说，机会在于：如果大约有 500 万 React 开发者（这是我们使用的库引擎），大约有 2000 万 JavaScript 开发者，那么有多少有志于构建产品的人存在呢？我粗略估算的最小值是 1 亿。

(00:08:45):

我告诉你这个数字是怎么来的。Slack 大约有 1 亿月活跃用户。你在 Slack 上做的是在 IT 领域与人交流。这些人中有很多都在构建数字产品。他们互相讨论想在世界上看到什么，通过共享频道与客户交流。我非常喜欢这个功能。我们经常与 Vercel 的客户交流，他们会告诉我们：“我想建这个，我想看那个，我想要这个功能。”所以 v0 的机会在于，不是让你停止与人交流，而是如果你能对着电脑“唠叨”几句，就能看到事情发生——构建原型、构建产品的第一个版本、构建演示、构建全栈产品，然后直接交付，那会怎样？

(00:09:30):

所以它的灵感与 Vercel 的使命非常契合。但具体来说，它的起源故事是：当 ChatGPT 问世时，我们注意到它非常擅长编写我们工具所使用的代码。ChatGPT 一经推出就精通 JavaScript、Tailwind（一种 CSS 样式技术）和 Next.js。这再次证明了开源的力量，我们的工具已经在互联网的训练数据中了。所以我们在开源上的长期赌注和愿景得到了回报。既然模型如此擅长编写这类代码，v0 的想法就自然而然地产生了：“如果我们能为构建 Web 产品做一个 ChatGPT 呢？”

[00:10:14] Lenny Rachitsky

English:

Speaking of that, I didn't actually know. So I had Bolt's CEO on the podcast and he talked about how Claude kind of unlocked what they're doing and do you guys sit on ChatGPT and OpenAI's stuff?

中文翻译:

说到这个，我其实不太清楚。我之前请过 Bolt 的 CEO 上播客，他谈到 Claude 如何开启了他们的业务。你们是基于 ChatGPT 和 OpenAI 的技术吗？

[00:10:25] Guillermo Rauch

English:

We started out on OpenAI. And we've always used a combination of models. It's funny, right now on Twitter there's thread with a million views of people trying to reverse engineer the prompt and the models they used. And they're all finding that there is all these kinds of different models that are specialists in different tasks. And there's a pipeline of models where a model could hand off work to another model. And so OpenAI, Gemini, Claude, but we predate Anthropic because I'll give credit to

ChatGPT that the utility of it was so general purpose, but from the very first release, it was very good. In fact, by the way, if I'm not mistaken, the first prototype of v0 might have even predated ChatGPT, or at the very least I think we were running on GPT 3.5. So we've always had this vision of unlocking more power for the web through LLMs, and there's a lot of very interesting technical details of why, by the way, LMs happen to be so good at the task of web design and web development that we could get into. But it was the perfect timing for us.

中文翻译:

我们是从 OpenAI 开始的。我们一直使用多种模型的组合。很有趣，现在 Twitter 上有一个百万浏览量的帖子，人们在尝试反向工程我们的提示词和使用的模型。他们发现有各种不同的模型专门负责不同的任务。这是一个模型流水线，一个模型可以将工作移交给另一个模型。所以我们使用了 OpenAI、Gemini、Claude，但我们比 Anthropic 更早开始，因为我要归功于 ChatGPT，它的通用性非常强，而且从第一个版本开始就非常好。事实上，如果我没记错的话，v0 的第一个原型甚至可能早于 ChatGPT，或者至少我们当时是在 GPT 3.5 上运行的。所以我们一直有通过大语言模型（LLM）为 Web 释放更多能力的愿景。顺便说一下，LLM 为什么如此擅长网页设计和开发，这其中有很多非常有趣的技术细节，我们可以深入探讨。但对我们来说，这确实是完美的时机。

[00:11:45] Lenny Rachitsky

English:

I want to come back to that. That's actually a really good question. But let me ask a couple other questions here. In terms of v0, what's the scale at this point? We hear all these numbers about all the folks in the space. What can you share about what's happening with v0?

中文翻译:

我想回头再聊那个话题，那确实是个好问题。但先让我问几个别的问题。关于 v0，目前的规模如何？我们听到了很多关于这个领域各种玩家的数据。关于 v0 的现状，你能分享些什么吗？

[00:11:45] Guillermo Rauch

English:

I can share that it's growing exponentially, and that over 1.3 million users have interacted with v0 so far. We had our largest day ever yesterday and today, again, we're one of the largest customers of most of cloud providers at this point. We're hitting the limits of every GPU, LLM infrastructure out there in the planet. And the most exciting thing for me is what I'm seeing people build with v0. So we launched a feature about a month ago, maybe even less than a month ago, called v0 Community. It already has 20,000 submissions. I am sure people in your audience have used Figma, one of the things that I love about Figma is Figma files, that I can go and grab a starting point for something. It could be a logo, could be a menu, and you can start with something that someone has already contributed, like that spirit of open source.

(00:12:44):

And so in less than a month, I think we've done over 20,000 community submissions. So we've learned so much about building AI products with this and we continue to open source and share our best practices. But one of the things that I've definitely learned is prompting it seems like the easiest interface in the world because it's just an input and you put text in it. But there's a little bit of a writer's block sometimes. So one of my favorite things that I've seen, and I'm even looking at the home page right now, and you can see a random assortment of community submissions. And they have 1,200 forks, and 1,500 forks, and

6,000 forks, and this is every time people saying like, "Oh, instead of starting from scratch, I'll start from this application that someone else has built and I'm going to prompt it to modify it and make it my own."

中文翻译:

我可以分享的是，它正在呈指数级增长，到目前为止已有超过 130 万用户与 v0 进行了交互。昨天和今天我们都刷新了单日最高记录，目前我们是大多数云服务商最大的客户之一。我们正在触及全球几乎所有 GPU 和 LLM 基础设施的极限。对我来说最兴奋的是看到人们用 v0 构建的东西。大约一个月前（甚至不到一个月），我们推出了一个名为“v0 社区”（v0 Community）的功能。目前已经有 2 万份提交。我相信你的听众中有人用过 Figma，我喜欢 Figma 的一点就是 Figma 文件，我可以去获取某个东西的起点。它可以是一个 Logo，可以是一个菜单，你可以从别人已经贡献的东西开始，就像开源精神一样。

(00:12:44):

所以在不到一个月的时间里，我们已经有了超过 2 万份社区提交。我们从中学习到了很多关于构建 AI 产品的知识，并继续开源和分享我们的最佳实践。但我学到的一点是，提示词（prompting）看起来是世界上最简单的界面，因为它只是一个输入框，你把文字放进去。但有时也会出现“写作障碍”。所以我最喜欢看到的事情之一——我现在正看着主页，你可以看到随机排列的社区提交。它们有的有 1200 个派生（fork），有的有 1500 个，有的有 6000 个。这意味着人们在说：“噢，与其从零开始，不如从别人构建的这个应用开始，然后我通过提示词来修改它，把它变成我自己的东西。”

[00:13:34] Lenny Rachitsky

English:

So the community submissions are people building apps on v0 and sharing what they built?

中文翻译:

所以社区提交是指人们在 v0 上构建应用并分享他们的成果？

[00:13:39] Guillermo Rauch

English:

Correct.

中文翻译:

没错。

[00:13:39] Lenny Rachitsky

English:

You can look at the code and fork it?

中文翻译:

你可以查看代码并派生（fork）它？

[00:13:40] Guillermo Rauch

English:

It is becoming like a compounding investment. People share something, someone else grabs it, makes it better. Maybe you used it at that point. In many ways, I see this as the next evolution of GitHub, whereas GitHub, it was a marvel for software development because I don't know if you remember this, but the initial, little tagline underneath the GitHub logo was social coding. And it had this democratization effect of building software. But you still needed to know how to code. And so what we're after is social product building in many ways, everybody should be able to cook and share what they're building.

中文翻译:

这正变成一种复利投资。有人分享了东西，另一个人拿去把它做得更好。在很多方面，我将其视为 GitHub 的下一次进化。GitHub 曾是软件开发的奇迹，不知道你是否记得，GitHub Logo 下方最初的小标语是“社交化编程”（social coding）。它对软件构建起到了民主化的作用。但你仍然需要知道如何写代码。而我们追求的是多方面的“社交化产品构建”，每个人都应该能够“下厨”并分享他们正在构建的东西。

[00:14:25] Lenny Rachitsky

English:

I hadn't thought of it this way, but I love that it connects so much to your open source roots, where people are building on v0, and then sharing what they're building and then people can build off those things. It's kind like an open source AI building experience.

中文翻译:

我之前没这么想过，但我很喜欢它与你的开源背景联系在一起：人们在 v0 上构建，分享成果，然后其他人再基于这些成果继续构建。这有点像一种开源的 AI 构建体验。

[00:15:26] Guillermo Rauch

English:

It's fascinating, right? In many ways, if you think about the Git commit, the Git commit is super interesting. If you watch how an engineer works, they look at a problem, they spend a lot of time in their code editor, and at the end they say, "I think I got it. I think I've fixed it." And then they produce a Git commit. They summarize their intent and what they try to do after they've done the work. v0 inverts that. The Git commit is you go into the chat and say, "Please change the color of this button. And when I click it, save this form to a database." And so you're starting with the intent and the output is the code.

(00:15:26):

And as a side effect, we can also produce a Git commit for you. That feature's not online yet, but it's coming in the next couple of days. Spoiler alert for the group. And so I like this idea of we can create this super set of all software building with this platform. And that is true to my initial intention with Vercel. Our mission is to enable the world to build and ship the best products. And so enabling that for the largest possible group of people is very exciting to me.

中文翻译:

这很迷人，对吧？在很多方面，如果你思考一下 Git 提交（Git commit），它非常有趣。如果你观察工程师如何工作，他们看到一个问题，在代码编辑器里花很长时间，最后说：“我想我搞定了，我想我修复了它。” 然后他们生成一个 Git 提交。他们在完成工作后总结自己的意图和所做的事情。v0 反转了这一过程。Git 提交变成了你进入聊天框说：“请更改这个按钮的颜色。当我点击它时，将此表单保存到数据库。” 所以你是从意图开始，输出结果是代码。

(00:15:26):

作为副产品，我们也可以为你生成 Git 提交。这个功能还没上线，但未来几天就会推出。给听众们剧透一下。所以我喜欢这个想法：我们可以通过这个平台创建所有软件构建的“超集”。这符合我创立 Vercel 的初衷。我们的使命是让世界能够构建并交付最好的产品。因此，让尽可能多的人群具备这种能力，对我来说非常令人兴奋。

[00:16:19] Lenny Rachitsky

English:

So let's go to this question of just the elephant in the room for a lot of people seeing these things happening, product builders that have been doing things a certain way for a long time with apps like this coming around, whether you could just type a thing in, and build it for you, and it's beautiful. How do you think things are going to change for product managers, for product teams? Where do you think the biggest change is going to happen? How do you think product will be built in the next few years?

中文翻译:

让我们来谈谈那个“房间里的大象”——对于很多看到这些变化发生的人来说，那些长期以来以特定方式工作的开发者，面对这类只需输入文字就能为你构建出精美产品的应用。你认为产品经理和产品团队的情况会发生怎样的变化？你认为最大的变化会发生在什么地方？未来几年产品将如何构建？

[00:16:19] Guillermo Rauch

English:

The most profound one that I alluded to is that conversations between product builders and their customers will be mediated by these zero links, these artifacts. I think when Claude came up with the name artifacts, I found it phenomenal, because we're all in this world, especially in this group of people, we're here to build awesome things and share them with the world. Steve Jobs said this awesome speech about, "It's like our form of giving back to the world is to try and do the best possible job we can and share it with the world." And so the idea that when we talk, we would not have the power to make those ideas a reality, it seems like an L to me. I would love to see people constantly live in the product, be in the design, spend time tuning and trying out new ideas. And that's what the ideal work of the future should look like, and less about again, that abstraction, that being removed from the product or even sometimes I can feel powerless to not be able to change something.

(00:17:47):

This happens a lot when departments collaborate within an organization. Marketing wants design to do something, marketing wants engineering, engineering needs a design. It cuts always. One of the things that people got excited about that we published on the Vercel blog was about design engineering, because a lot of the people that we were noticing were being very successful at Vercel were people that had both the design and engineering skills. And that was actually another huge motivator and inspiration for v0, because we realized that people could be more full stack. We shouldn't put limits on ourselves, and what we can build, and what we can ship, and what we can dream about making possible on these web surfaces.

(00:18:36):

And so you could imagine removing all those limitations, a designer that can ship a fully-baked product, a product manager that can prototype and shift to production. A lot of people that use v0 are back-end

engineers that never had the ability to, they could ship an API, they could build a great low-level infrastructure system, but to actually bring their end-to-end vision to life, v0's completing that for them.

中文翻译:

我提到的最深刻的一点是，产品构建者与客户之间的对话将由这些 v0 链接（即“artifacts”，作品/产物）来媒介。我觉得 Claude 想到“artifacts”这个名字非常了不起。因为我们都在这个领域，尤其是我们这群人，我们的目标是构建出色的东西并与世界分享。史蒂夫·乔布斯曾有过一段精彩的演讲，他说：“我们回馈世界的方式，就是尽我们所能做到最好，并将其分享给世界。” 所以，如果我们只是空谈，却没有能力将这些想法变成现实，对我来说就像是一种失败（an L）。我希望看到人们不断地沉浸在产品中，沉浸在设计中，花时间调整和尝试新想法。这就是未来理想工作的样子，而不是那种抽象的、脱离产品的状态，甚至有时会因为无法改变某些东西而感到无能为力。

(00:17:47):

这种情况在组织内部跨部门协作时经常发生。营销部门想要设计做点什么，营销想要工程做点什么，工程又需要设计。这种隔阂一直存在。我们在 Vercel 博客上发表的一篇关于“设计工程”（design engineering）的文章让人们非常兴奋，因为我们注意到在 Vercel 取得巨大成功的人，往往是那些同时具备设计和工程技能的人。这实际上也是 v0 的另一个巨大动力和灵感来源，因为我们意识到人们可以变得更加“全栈”。我们不应该给自己设限，不应该限制我们能构建什么、能交付什么，以及我们梦想在这些网页界面上实现什么。

(00:18:36):

所以你可以想象，消除所有这些限制后，一个设计师可以交付一个完整的产品，一个产品经理可以制作原型并将其发布到生产环境。很多使用 v0 的人是后端工程师，他们以前没有能力做前端——他们可以交付 API，可以构建出色的底层基础设施系统，但要真正将他们的端到端愿景变为现实，v0 为他们补全了这一环。

[00:19:21] Lenny Rachitsky

English:

Let me follow the thread on engineers. A lot of people are wondering, "Do we need engineers in the future? What happens to engineers? Should I learn how to code?" Your long-time engineer thoughts for folks that are trying to decide the career for themselves?

中文翻译:

让我顺着工程师的话题问下去。很多人都在想：“未来我们还需要工程师吗？工程师会怎么样？我还需要学习编程吗？” 作为一名资深工程师，你对那些正在规划职业生涯的人有什么看法？

[00:19:21] Guillermo Rauch

English:

Yeah, I think knowing how things work is the most important skill in the world. I foresee a lot of people becoming incredibly impactful in building and shipping amazing products, and building gigantic companies, and everything you could imagine, where a single person can do the job of a hundred different people in a hundred different specializations. Take the example of one skill set that's really important to build a front-end product is you need to know how to use CSS or Tailwind to style it. And once upon a time, I would hire people that were truly specialists in this task, the task of there's a Figma design or there is some kind of sketch, and translating that into reality because they knew really well how to manipulate layouts, layout code, box model code, we call it, and borders, paddings, margins, flex box, all these technologies for styling.

(00:20:32):

And notice, I actually use the word translation very intentionally, because the origin of the LLM or the transform architecture at least, goes as far back as the architecture for systems like Google Translate. They were generative LLM techniques, basically. That's how they cross that chasm of, remember when translating tools were horrible and then one day the problem was just solved? And I look at a lot of the programming jobs to be done that used to be specializations, that I think are going away, in a way, or the tasks to be done, they're translation tasks. We were translating from a screenshot, or intent, or a design into a React, and Tailwind, and CSS implementation.

(00:21:32):

And right now, v0 is incredibly good at doing that. It's so good that every time we put a new generation of the model out, I run this test of converting my own website and try to generate it with v0. Last time I did it, it had taken me like 10 prompts to replicate it. Keep in mind I'm an expert front-end engineer that's been in the arena since I'm like 10 years old and I'm 35 now. And so I do that test because it's almost like a test of self-imposed humility of, like, "I remember exactly how long it took me to build my website with Next.js, the framework that I created, and ship it." And so with the last model, it took me maybe 10, 15 prompts? With the most recent model, it took me two prompts.

(00:22:22):

And so that translation from the design intent into working implementation, another anecdote that I like to share with people is the model, because v0 tries to embed all of the best practices of the web, the model output more accessible code than what I wrote. It follows the accessibility guidelines that the web standards consortiums put out better than I did, because it just knows everything. And so those tasks where you can almost model it to a translation task, definitely going away. But knowing how things work under the hood, notice all the ... I'm using specific tokens in this conversation. I'm saying, "CSS," I'm saying, "Layout." I'm naming styles. Knowing those tokens is going to be very important for you because you're going to be able to influence the model and make it follow your intention a lot better.

(00:23:22):

And so the TLDR would be knowing how things work, the symbolic systems, and that will mean that you have to probably go into each subject with less depth. I have engineers at Vercel that know every single CSS property by heart. They know when they became available in a certain web browser, they've been tracking this specification. It's almost like you're an encyclopedia of knowledge of each CSS property. You probably won't need that in the future, and probably that's good, because you'll free up your mind for more ambitious things.

中文翻译:

是的，我认为“了解事物如何运作”是世界上最重要的技能。我预见到很多人将在构建和交付惊人产品、建立巨型公司等方面变得极具影响力，一个人就能完成 100 个不同专业领域的人的工作。举个例子，构建前端产品的一项重要技能是需要知道如何使用 CSS 或 Tailwind 来设置样式。曾几何时，我会雇佣专门负责这项任务的人——将 Figma 设计或草图转化为现实，因为他们非常擅长操作布局、布局代码、盒模型代码（box model code），以及边框、内边距、外边距、Flexbox 等所有样式技术。

(00:20:32):

请注意，我非常刻意地使用了“翻译”（translation）这个词。因为 LLM 或至少 Transformer 架构的起源，可以追溯到像 Google 翻译这样的系统架构。它们基本上都是生成式 LLM 技术。这就是它们如何跨越鸿沟的——还记得翻译工具曾经很糟糕，然后有一天问题突然就解决了？我看到很多以前属于专业分工的编程任务，我认为在某种程度上正在消失，或者说那些任务本质上是“翻译任务”。我们是在将截图、意图或设计“翻译”成 React、Tailwind 和 CSS 的实现。

(00:21:32):

现在，v0 非常擅长做这件事。它做得太好了，以至于每次我们发布新一代模型时，我都会做一个测试：尝试用 v0 重新生成我自己的网站。上次做的时候，我用了大约 10 个提示词才复刻出来。记住，我是一名资深前端工程师，从 10 岁起就在这个领域摸爬滚打，现在 35 岁了。我做这个测试几乎是一种“自我谦逊”的测试，就像在说：“我清楚地记得当初用我自己创建的框架 Next.js 构建并交付网站花了多长时间。” 用上一个模型，我可能需要 10 到 15 个提示词；而用最新的模型，只用了两个提示词。

(00:22:22):

这种从设计意图到实际实现的“翻译”——我还喜欢分享另一个轶事：因为 v0 试图嵌入 Web 的所有最佳实践，模型输出的代码比我写的更具“可访问性”（accessible）。它比我更好地遵循了 Web 标准联盟发布的辅助功能指南，因为它无所不知。所以，那些几乎可以建模为翻译任务的工作，肯定会消失。但是，了解底层的运作原理——注意我在对话中使用的特定词汇（tokens），我说“CSS”，说“布局”，命名各种样式。了解这些词汇对你来说非常重要，因为这样你才能影响模型，让它更好地遵循你的意图。

(00:23:22):

所以，简而言之（TLDR）就是：了解事物如何运作，了解符号系统。这意味着你可能不需要在每个学科上钻研得那么深。Vercel 有些工程师能背出每一个 CSS 属性，知道它们什么时候在哪个浏览器上线，一直追踪规范。你就像是一个 CSS 属性的百科全书。未来你可能不再需要这些了，这也许是件好事，因为你可以腾出大脑去思考更有野心的事情。

[00:23:59] Lenny Rachitsky

English:

No, that's fascinating. So what I'm hearing is a skill that will continue to be valuable in the future, but I want to push on this a little bit, no matter how far AI gets, is understanding conceptually how software works, end-to-end systems, databases, CSS is a thing. So I don't know if you have kids, whether you have kids or not, just say they were trying to decide, "What should I learn to be, to thrive in this future?" Well, how would you summarize it? How far? Should they get into software engineering?

中文翻译:

不，这太迷人了。所以我听到的是，无论 AI 发展到什么程度，未来仍然有价值的技能是：从概念上理解软件如何运作，理解端到端系统、数据库、CSS 是什么。我不知道你有没有孩子，假设他们正在决定“为了在未来茁壮成长，我应该学习什么？”，你会如何总结？他们还应该进入软件工程领域吗？

[00:24:31] Guillermo Rauch

English:

Great question, because I have five kids, and I've already enrolled them in this school of G, myself, in the sense that I'm already guiding them towards the things I think are going to be very useful to them. So understanding how things work needs, I think the ability to understand the fundamental logic behind things, incredibly valuable. So I push them really hard on math. "If you don't know math really well, you're out of my house." Just kidding. But it's a fundamental skill that I want them to know. Eloquence. I joke sometimes. Have you heard a meme of word cells versus shape rotators?

(00:25:14):

So a shape rotator is someone that only has a math brain. You could argue the kings and queens of Silicon Valley have been the shape rotators, because those have been the jobs that have historically commanded

the most status, respect, net worth, whatever. And then there's the word cells, which is communicating, more of the liberal arts. There's also the funny and awesome slide of Apple saying that they're at the intersection of liberal arts and technology. I've always had immense amounts of respect for both sides of the brain, so to speak. But I think developing great eloquence, and knowing and memorizing those tokens that I talked about, knowing how to refer to things in that global mental map of symbolic systems will be highly valuable. And we have some tools to help people prompt better, but prompt enhancement and embellishment cannot replace thinking and cannot replace your own creativity that you want to infuse into the world.

(00:26:18):

So one of the things that v0 does is it tries and it succeeds, I think, at creating very nice designs out of the box. We try to infuse what we've learned about what do people think is typically good web design? We've influenced the model in that direction. But still we also don't want the whole internet to look the same way. So your ability to steer the model with your words into those references, into those inspirations, is going to be very important.

(00:26:49):

I actually have an amazing anecdote. We hosted a design demo night at the Vercel HQ in San Francisco last night. And we were showing off how Vercel uses v0 to build v0 and to build Vercel. And one of our designers showed this amazing animation that he built, actually two amazing animations that he built. And in one of them it was this amazing triangle that had an animation that I didn't think was possible to make, in that it was all built with v0. And he used the word turbulence to describe the effect that he wanted.

(00:27:30):

So I just want to call out that to people because the difference between knowing that word and not knowing it is getting that style into that beautiful triangle that he created that was interactive, and it's probably going to end up in some landing page soon that you're going to visit on vercel.com. And so developing eloquence and your linguistic ability I think is going to be very important. So I love my kids to know that. And I think that idea of sharing things, and putting yourself out there, and broadcasting to the world, so another thing that I do is I take my kids to hackathons, which just went to an awesome hackathon at University of San Francisco, USF. It was called the BLOOM Hackathon. And I took two of my kids and I wanted them to watch how people presented their ideas and we had a lot of fun. We also ate waffles and grilled sandwiches, which is a bonus.

(00:28:29):

So presenting and putting yourself out there. I mentioned in the beginning of the podcast when we were chatting, I've learned so much from you and your guests because you put out all these awesome little posts on X in these videos and these snippets of your interviews. And so the ability to present what you've built and put yourself out there, incredibly important skill in the future. Especially in a world where the marginal cost of producing software and new things are going down, you need to build an audience, you need to know how to talk to people, you need to build your own signature brand and style. And so maybe they're a little too young for that one, but I guess taking them into hackathons probably back, is influencing their neural networks or pre-training data for the future.

中文翻译:

问得好，因为我有五个孩子，我已经把他们招进了我自己的“G 学校”，也就是说我已经在引导他们学习我认为对他们非常有用的东西。理解事物如何运作需要理解背后的基本逻辑，这极其宝贵。所以我非常努力地推他们学数学。“如果你数学学不好，就给我搬出去。”开玩笑的。但数学是我希望他们掌握的一项基本技能。还有

“口才”（Eloquence）。我有时会开玩笑，你听过“文字细胞”（word cells）对阵“形状旋转者”（shape rotators）的梗吗？

(00:25:14):

“形状旋转者”是指只有数学头脑的人。你可以说硅谷的国王和王后们一直都是形状旋转者，因为这些工作在历史上占据了最高的地位、尊重和净值。然后是“文字细胞”，即擅长沟通、更偏向文科的人。苹果公司也有一张很有名且很棒的幻灯片，说他们处于文科与技术的交汇点。我一直对大脑的这两个方面都充满敬意。但我认为，培养出色的口才，了解并记住我提到的那些词汇（tokens），知道如何在符号系统的全球心理地图中引用事物，将具有极高的价值。虽然我们有一些工具可以帮助人们更好地写提示词，但提示词的增强和润色无法取代思考，也无法取代你想注入世界的个人创意。

(00:26:18):

v0 所做的一件事是，它尝试并成功地开箱即用地创建非常漂亮的设计。我们试图注入我们学到的知识：人们通常认为什么是好的网页设计？我们在这个方向上影响了模型。但我们也不希望整个互联网看起来都一样。所以，你用语言引导模型走向那些参考、那些灵感的能力将非常重要。

(00:26:49):

我其实有一个很棒的轶事。昨晚我们在旧金山的 Vercel 总部举办了一个设计演示之夜。我们展示了 Vercel 如何使用 v0 来构建 v0 以及构建 Vercel。我们的一位设计师展示了他构建的一个惊人动画，实际上是两个。其中一个是这个神奇的三角形，它的动画效果我以前认为是不可能做出来的，而它完全是用 v0 构建的。他使用了“湍流”（turbulence）这个词来描述他想要的效果。

(00:27:30):

我特别想向大家强调这一点，因为知道这个词和不知道这个词的区别，决定了能否在他创建的那个漂亮的交互式三角形中实现那种风格。这个三角形很快就会出现在你访问 vercel.com 的某个落地页上。因此，培养口才和语言能力我认为将非常重要。我希望我的孩子们掌握这些。还有分享、展示自我、向世界广播的想法。我做的另一件事是带孩子们去参加黑客松（hackathons），最近刚去了旧金山大学（USF）的一个很棒的黑客松，叫 BLOOM Hackathon。我带了两个孩子去，想让他们看看人们是如何展示自己的想法的，我们玩得很开心，还吃了华夫饼和烤三明治，这是额外奖励。

(00:28:29):

所以，展示和推销自己。我在播客开始聊天时提到过，我从你和你的嘉宾身上学到了很多，因为你在 X 上发布了这些精彩的小帖子、视频和采访片段。因此，展示你所构建的东西并推销自己的能力，是未来极其重要的技能。尤其是在生产软件和新事物的边际成本不断下降的世界里，你需要建立受众，需要知道如何与人交谈，需要建立自己的签名品牌和风格。也许他们现在还太小，理解不了这一点，但带他们去黑客松可能正在潜移默化地影响他们的神经网络或未来的预训练数据。

[00:29:19] Lenny Rachitsky

English:

I love it. They're going to tell their friends, "My dad took me to a hackathon." "What's that?" So are you encouraging to learn to code? Because it's interesting you mentioned math, eloquence, presenting, and then, okay, so also learn to code.

中文翻译:

太棒了。他们会告诉朋友：“我爸带我去参加了黑客松。” “那是什么？” 所以你还是鼓励学习编程吗？因为很有趣，你提到了数学、口才、演讲，然后，好吧，还要学习编程。

[00:29:32] Guillermo Rauch

English:

Yeah, I think again, learning how to prompt, learning how to code. With v0, we show you the code when we build things. So if you can build that mapping of maybe not learning how to code necessarily as an abstraction, if you do have a knack for it, I'm a big believer also that my five kids have super diverse personalities and inclinations, and I don't want to be pushing for something that they wouldn't want to do or whatever. And so learning to code in the abstract might be good for some people, but it may not be the fun thing to do for other people. And so what I would recommend is try to understand how things work. So if you prompt v0 or any other tool and it generates some code, try to build an understanding of what that does at a high level. It's like actually maybe an extension even of eloquence.

(00:30:32):

One of the bets that I made early on with Vercel that really paid off is Vercel, maybe as a metaphor is like AWS in easy mode for a lot of people. We have a very large user base of people that would have otherwise not have been able to configure all of the ins and outs of the cloud, but do want the scale, flexibility, speed, et cetera. They want to create very high quality products and services. So I like to give the Super Bowl example because one of our customers, Ramp, had a 43X increase in traffic when their ad went live. The engineer that worked on that only needed to learn Next.js. Then they pushed their code to Vercel and now they can reach an audience of a hundred million people without a blip, a hundred percent uptime.

(00:31:24):

That superpower comes from, we made it as easy as possible to get started, and the language that we choose is actually very relevant in this story. JavaScript, in my mind, has always been almost like the English of programming languages. It's a language that, if you learn it, you reach billions of devices. So it's not a coincidence that when you ask ChatGPT, or Anthropic, or Gemini to build you web app, it uses these tools. It uses JavaScript, it uses React. It's become the lingua franca of building products on the web. So I would say to my kids, "Look, if you do want to go deeper into programming, start learning there." You can reach huge numbers of people. If you have a passion, I would say there's going to be a fundamental engineering skill that's going to be useful for decades or centuries to come, which is creating foundational infrastructure.

(00:32:26):

Think about LLMs in terms of, they're like Oracles that can go and write software for you, but there's a limit to how much software they can write. There's context windows, there is time and computational constraints. So it's very hard for an agent today to go and say, "I'm going to write a cloud from scratch. I'm going to write all the foundational services. I'm going to write the framework from scratch. I'm going to write the compiler." No, the LM is orchestrating those tools and infrastructure. It's not writing the compiler from scratch. Otherwise, you get into the Newton thing, in order to create an Apple, you have to create the entire underlying universe. No, the elements are interoperating with the universe as it exists. And so the engineers that learn foundational infrastructure are probably going to be extremely empowered still, for years to come.

中文翻译:

是的，我认为学习如何写提示词，学习如何编程。在 v0 中，我们在构建时会向你展示代码。所以，如果你能建立这种映射——也许不一定要把学习编程当作一种抽象的任务，如果你确实有这方面的天赋。我也坚信我的五个孩子有着非常多样化的个性和倾向，我不想强迫他们做他们不想做的事。所以，抽象地学习编程对某些人可能很好，但对另一些人来说可能并不有趣。因此，我的建议是尝试理解事物如何运作。如果你给 v0 或任何其他工具一个提示词，它生成了一些代码，试着在宏观层面理解这些代码的作用。这实际上甚至可能是“口才”的一种延伸。

(00:30:32):

我早期在 Vercel 身上下的一个赌注得到了回报，那就是 Vercel 对很多人来说就像是“简易模式的 AWS”。我们拥有庞大的用户群，这些人原本可能无法配置云服务的所有细节，但他们确实需要规模、灵活性、速度等等。他们想要创建高质量的产品和服务。我喜欢举超级碗的例子，因为我们的一个客户 Ramp 在广告上线时流量增加了 43 倍。负责该项目的工程师只需要学习 Next.js，然后将代码推送到 Vercel，现在他们就可以在没有任何故障的情况下触达 1 亿受众，实现 100% 的在线率。

(00:31:24):

这种超能力源于我们让入门变得尽可能简单，而我们选择的语言在这个故事中也非常关键。在我心目中，JavaScript 一直几乎就是编程语言中的“英语”。它是一种只要你学会了，就能触达数十亿设备的语言。所以，当你要求 ChatGPT、Anthropic 或 Gemini 为你构建 Web 应用时，它使用这些工具并非巧合。它使用 JavaScript，使用 React。它已成为 Web 产品构建的通用语言（lingua franca）。所以我会对我的孩子们说：“看，如果你确实想深入学习编程，从那里开始。” 你可以触达巨量的人群。如果你有热情，我会说有一种基础工程技能在未来几十年甚至几个世纪都将非常有用，那就是创建基础架构（foundational infrastructure）。

(00:32:26):

把 LLM 想象成先知（Oracles），它们可以为你编写软件，但它们能编写的软件量是有限的。存在上下文窗口、时间和计算约束。所以，现在的智能体（agent）很难说：“我要从头开始写一个云平台，写所有基础服务，写框架，写编译器。” 不，LLM 是在编排这些工具和基础设施，而不是从头写编译器。否则，你就会陷入牛顿说的那个困境：为了创造一个苹果，你必须先创造整个宇宙。不，LLM 是在与现有的宇宙进行互操作。因此，学习基础架构的工程师在未来几年仍将拥有极大的权力。

[00:33:30] Lenny Rachitsky

English:

There's a world where you could argue ChatGPT will build the next version of ChatGPT. What I'm hearing from you is that's a long ways away, if ever.

中文翻译:

有一种观点认为 ChatGPT 会构建下一个版本的 ChatGPT。但我从你这里听到的是，那还很遥远，甚至可能永远不会发生。

[00:33:30] Guillermo Rauch

English:

Absolutely. This is why the common, the running joke is that all of these companies have, you go to their careers page. It's like "Engineer, engineer, engineer." The counterpoint of that is that at Vercel had, we have 150 engineers that can write code and 600 total headcount. Now we have 600 engineers. Some of the best things that I've seen created with v0 have not come from our engineering team. They've come from the marketing team, they've come from the sales team, they've come from the product management team. The product management team is fascinating, because now they're actually building the product. So last night I saw how we've specced out in v0, think of it as like a live PRD, we've specced out how the new functionality for deploying a v0 to Vercel is going to work.

(00:34:26):

The amount of detail that was contained in that v0, I mean, we're all just saying, "Well, just ship it. There's nothing else to discuss." It was animated, it was interactive. We were demonstrating the error state, the

success state, the slow stream state. So it really empowers product builders not only with technical skills, I think that does a disservice to the tool. It empowers them to explore and augment their thinking with a lot of things that perhaps they wouldn't have considered otherwise, a lot of states of the product they wouldn't have considered otherwise.

中文翻译:

绝对如此。这就是为什么那个流传甚广的笑话是：你去所有这些公司的招聘页面看，全是“工程师，工程师，工程师”。与此相对的是，在 Vercel，我们有 150 名能写代码的工程师，总人数是 600 人。但现在，我们有 600 名“工程师”。我看到的用 v0 创建的一些最好的东西并非出自我们的工程团队，而是来自营销团队、销售团队和产品管理团队。产品管理团队非常迷人，因为现在他们实际上是在构建产品。昨晚我看到我们在 v0 中如何定义规格（你可以把它看作是一个活生生的 PRD），我们定义了将 v0 部署到 Vercel 的新功能将如何运作。

(00:34:26):

那个 v0 原型中包含的细节量，让我们所有人都说：“好吧，直接发布吧，没什么好讨论的了。” 它是带动画的，是交互式的。我们演示了错误状态、成功状态、慢速流状态。所以它真正赋予了产品构建者力量，不仅是技术技能——我觉得说它只提供技术技能是贬低了这个工具。它赋予了他们探索和增强思维的能力，让他们考虑到许多原本可能不会考虑的事情，以及许多原本不会考虑的产品状态。

[00:35:05] Lenny Rachitsky

English:

The name v0 implies the product is for prototypes for the first attempt at stuff. And that's definitely where all these tools are finding product market fit prototypes, PMs showing a thing working versus just design. Do you expect v0 and other tools to get to a place where you can build salesforce.com and scale it to billions of dollars? Do you-

中文翻译:

v0 这个名字暗示该产品是用于原型的，用于事物的第一次尝试。这确实是所有这些工具找到产品市场契合点（PMF）的地方——原型制作，PM 展示一个可以运行的东西而不仅仅是设计。你预见 v0 和其他工具能达到构建像 salesforce.com 这样规模并扩展到数十亿美元业务的程度吗？你——

[00:35:27] Guillermo Rauch

English:

Absolutely. We already have an enterprise customer of v0 that only works with v0. All of their products, all of their features, all of their client communications are v0 native. Two days ago, I just heard anecdotally on X, someone tells me, "My brother just sold his first website to a client completely built in v0." Yesterday at an investor conference, an investor walks up to me and says, "Two of my friends just got engaged on v0." I was like, "Okay, v0 is a dating app now." So the engagement website, the proposal, the wedding, it's all v0 native.

(00:36:07):

So because we've integrated v0, the Vercel infrastructure, we can do that whole story that I just told you of like, "I have a website to build and I can get it in front of a hundred million people." We can enable that for everybody now. And so the end-to-end full stack, v0 native, and built on this awesome fluid serverless infrastructure that scales to billions of people, all just from prompts, or screenshots, or just copying and pasting your PRDs into the tool.

中文翻译:

绝对可以。我们已经有一个企业客户完全基于 v0 运作。他们所有的产品、所有的功能、所有的客户沟通都是 v0 原生的。两天前，我在 X 上听到了一个轶事，有人告诉我：“我兄弟刚把他的第一个网站卖给了一个客户，完全是用 v0 构建的。” 昨天在一个投资者会议上，一位投资者走过来对我说：“我的两个朋友刚在 v0 上订婚了。” 我当时想：“好吧，v0 现在成相亲应用了。” 订婚网站、求婚、婚礼，全都是 v0 原生的。

(00:36:07):

因为我们将 v0 与 Vercel 基础设施集成在了一起，我们可以实现我刚才讲的那个完整故事，比如：“我有一个网站要建，我可以让它触达 1 亿人。” 我们现在可以让每个人都具备这种能力。这种端到端全栈、v0 原生、构建在可扩展至数十亿人的出色流式 Serverless 基础设施上的体验，只需通过提示词、截图，或者直接把你的 PRD 复制粘贴到工具里就能实现。

[00:36:59] Guillermo Rauch

English:

Number one is you can be as ambitious as you want in terms of what you ask the tool. If you can steer the tool towards some kind of inspiration that you have, you're always going to get better results. If you don't have ideas on what to build or what to prompt, I would recommend using the v0 community so that you can find something to fork to get started. I would say in some ways, if you have technical skills, this one is interesting, have some suspension of disbelief. It humbled me, I was saying about accessibility. So be open-minded about whether the tool actually knows some things that you might not know, and so focus more on the product description, focus more on what do you want the end user to experience? What do you want the product to do? And try to be open-minded about how well the tool can implement it. Those would be my main wants.

(00:38:04):

You also have to have a sense of iteration, I guess. Think of it this way, if you were working with a design firm or an agency that you've hired, you will go back and forth and say, "Try something else." If you were coaching an engineer that's getting stuck in something, you would say, "Try something else." It's amazing how many times I've gotten unstuck in v0 by just saying, "Just try something else."

中文翻译:

第一点是，在向工具提出要求时，你可以尽可能大胆。如果你能引导工具走向你拥有的某种灵感，你总能得到更好的结果。如果你不知道该构建什么或该写什么提示词，我建议使用 v0 社区，这样你可以找个东西派生（fork）来开始。我想说，在某种程度上，如果你有技术背景，这一点很有趣：请保持一种“暂缓怀疑”的态度。就像我说的辅助功能那样，它让我感到谦卑。所以，对于工具是否真的知道一些你可能不知道的事情，请保持开放的心态。更多地关注产品描述，更多地关注你希望最终用户体验到什么，你希望产品做什么，并对工具能实现到什么程度保持开放心态。这些是我的主要建议。

(00:38:04):

我想你还需要有一种迭代意识。这样想：如果你在和一家设计公司或你雇佣的代理机构合作，你会来回沟通说：“试试别的方案。” 如果你在指导一个卡在某个问题上的工程师，你会说：“试试别的办法。” 令人惊讶的是，有多少次我仅仅通过说“试试别的办法”就在 v0 中解决了卡壳问题。

[00:38:35] Lenny Rachitsky

English:

Just saying that as the prompt, not even giving direct-

中文翻译:

直接把这句话当作提示词？甚至不给具体的指令？

[00:38:37] Guillermo Rauch

English:

Just saying that. I mean, the chat is like "v0, we need to have ... " It's like, "Yeah." Like you have a one-on-one performance review with a tool. "Hey, way to talk, try something else. What you're doing so far is not working. And it's amazing." One fitness function that I'm keeping in my head is I really want to find the thing that it cannot build with v0. So as part of the v0 community, I have my own profile. We'll share the link with people. You can see six or seven things that I've built that I consider to be pretty impressive. So for example, I was flying from Tokyo to San Francisco. The internet was horrible. What I like to do during flights is I like to monitor our own flight while I'm on the flight. So I open Flightradar or whatever, and I was extremely bored as well.

(00:39:32):

And I noticed that Flightradar, I don't know which one it was, Flightradar, there's like four or five of them. They were very bloated. They had ads. They were not what I wanted the flight radar to look like. So I built my own during the flight with the worst internet connection that you could imagine in the world, integrated into a flight data API called Edge Aviation. So this is what I told v0, "You're going to build the best flight radar on the planet." I wasn't prescriptive at how, so it used a tool called Mapbox and a JavaScript library called Leaflet. I didn't tell him that, or her, I don't know, v0, what it is. And subsequently, once we cooked on the design, which looks, I would say beautiful, I then got more ambitious and I said, "All right, let's make it real now."

(00:40:30):

And by the way, that's actually how I would work. So it's how I like to work. I like to work experience first, and that's also how Vercel was built. "Let's start with the front end. Let's start with the planes on the screen." And by the way, there's a lot of subtleties, here. For example, there's so many flights going on at any given time that there's just too many. So I had to work with v0 on improving performance. And once again, I wasn't prescriptive. I just said, "We have a lot of flights, chief. Let's- "

中文翻译:

就这么说。我的意思是，聊天框就像是：“v0，我们需要……” 就像你在和工具进行一对一绩效评估。“嘿，说话注意点，试试别的。你目前做的行不通。” 这太神奇了。我脑子里一直有一个“适应度函数”（fitness function），就是我真的很想找到 v0 无法构建的东西。作为 v0 社区的一部分，我有自己的个人资料，我们会把链接分享给大家。你可以看到我构建的六七个我认为相当令人印象深刻的东西。例如，我当时正从东京飞往旧金山，网速极差。在飞行过程中，我喜欢监控我们自己的航班，所以我打开了 Flightradar 之类的应用，当时我也非常无聊。

(00:39:32):

我注意到 Flightradar（我不确定是哪一个，这类应用有四五个）非常臃肿，有广告，不是我想要的飞行雷达的样子。于是我在飞行过程中，用你能想象到的世界上最差的网速，构建了我自己的飞行雷达，并集成了一个名为 Edge Aviation 的飞行数据 API。我是这样告诉 v0 的：“你要构建地球上最好的飞行雷达。” 我没有规定具体怎么做，所以它使用了一个叫 Mapbox 的工具和一个叫 Leaflet 的 JavaScript 库。我没告诉它（或者“她”，我不知道 v0 是什么性别）这些。随后，当我们完成了设计（我觉得非常漂亮）后，我变得更有野心了，我说：“好了，现在让它变成真的吧。”

(00:40:30):

顺便说一下，这实际上就是我的工作方式。我喜欢“体验优先”，这也是 Vercel 的构建方式。“让我们从前端开始，从屏幕上的飞机开始。” 顺便说一下，这里有很多微妙之处。例如，在任何给定时间都有太多的航班在飞行，数量太庞大了。所以我不得不和 v0 一起优化性能。我再次没有规定具体做法，我只是说：“头儿（chief），我们有很多飞机，让我们——”

[00:41:02] Lenny Rachitsky

English:

Did you say, "Chief?"

中文翻译:

你叫它“头儿”？

[00:41:03] Guillermo Rauch

English:

Yeah, I do say that a lot. And this is, I think when I shared it on X, it blew a lot of engineers' minds, because it created a canvas-based, canvas is the sort of underlying rendering surface that very sophisticated products use like Figma. And it created this awesome overlay on top of the map that can render tens of thousands of flights at any given time. And then I told it, "Let's make it a full stack application. Okay, plug into the flights' API." So that's an example of we cooked and there was no limit. And so I'm always in the lookout. The service that I'm providing to the v0 community is I'm part of the team that is really trying to break this and say, "Can it not build something?" And even when it does build it, we're very obsessed with quality and performance. It has to be real. That's our commitment to our users.

中文翻译:

是的，我经常这么叫。我想当我把这个分享到 X 上时，它让很多工程师大吃一惊，因为它创建了一个基于 Canvas 的渲染层（Canvas 是像 Figma 这样非常复杂的产品使用的底层渲染表面）。它在地图上创建了这个很棒的覆盖层，可以同时渲染数万个航班。然后我告诉它：“让它成为一个全栈应用吧。好了，接入航班 API。”这是一个我们不断尝试且没有限制的例子。所以我一直在寻找突破口。我为 v0 社区提供的服务是，我是那个真正试图“玩坏”它并问“它有什么是不能构建的吗？”的团队成员。即使它构建出来了，我们也对质量和性能非常痴迷。它必须是真实的，这是我们对用户的承诺。

[00:42:00] Lenny Rachitsky

English:

And how much did this cost, how much time does this take to make something like this?

中文翻译:

这花了多少钱？做这样一个东西需要多长时间？

[00:42:06] Guillermo Rauch

English:

So the flight radar example or v0?

中文翻译:

你是说飞行雷达的例子还是 v0 本身?

[00:42:08] Lenny Rachitsky

English:

Yeah, the flight radar example specifically just like very-

中文翻译:

是的，具体指飞行雷达那个例子。

[00:42:11] Guillermo Rauch

English:

I mean, that one probably took less than two hours with the worst internet.

中文翻译:

我的意思是，那个在网速极差的情况下可能花了不到两个小时。

[00:42:14] Lenny Rachitsky

English:

Yeah, what-

中文翻译:

天哪，那——

[00:42:15] Guillermo Rauch

English:

Sorry, Japan Airlines, I love you, but you give me a hard time.

中文翻译:

抱歉，日本航空，我爱你们，但你们的网速确实让我很头疼。

[00:42:18] Lenny Rachitsky

English:

And what did that cost? Like 10 bucks? What would you estimate?

中文翻译:

那花了多少钱？大约 10 美元？你估算一下。

[00:42:24] Guillermo Rauch

English:

I mean, I pay for the $20 v0 subscription.

中文翻译:

我的意思是，我付的是每月 20 美元的 v0 订阅费。

[00:42:25] Lenny Rachitsky

English:

20 bucks, okay, for a month. So it's like a month, but you used it for two hours, 20 bucks.

中文翻译:

20 美元，好吧，是一个月的费用。所以相当于一个月 20 美元，但你只用了两个小时。

[00:42:31] Guillermo Rauch

English:

Yeah.

中文翻译:

是的。

[00:42:31] Lenny Rachitsky

English:

If you had engineers building this, how much do you think that would cost? How long do you think that would take?

中文翻译:

如果你让工程师来构建这个，你认为会花多少钱？需要多长时间？

[00:42:36] Guillermo Rauch

English:

I mean, weeks, easily. Easily.

中文翻译:

我的意思是，轻轻松松也要几周时间。

[00:42:40] Lenny Rachitsky

English:

And that's like tens of thousands of dollars.

中文翻译:

那相当于数万美元的成本。

[00:42:42] Guillermo Rauch

English:

Maybe the most cracked engineer at Vercel could knock it out in ... without using any AI, could knock it out in a couple days. But then what about the design? What about me? Because I'm the bottleneck, not the engineer. And this is what's amazing about this collaboration because I'm providing the product guidance. I'm saying, "Draw a dashed line between the ... " And by the way, v0 just blew my mind so hard. I said, "Draw a dashed line between the two destination airports." And v0 said, "Well, I have to account for the spherical, or what is it, it's a pseudosphere, for the curvature of the earth." It's like, "Okay, v0, super genius, whatever." And so that's what I mentioned about how you can go back and forth. It's like a product copilot, it's like an all-knowing being.

(00:43:40):

One of our users yesterday submitted feedback to the tool and it was positive feedback. They were very happy, what they were saying, "v0 is a super genius five-year old PhD with ADHD." So you still have to, I'm not going to oversell this like, "It knows everything about everything. It gives everything perfect," of course. But it has these sparks of brilliance. Really, truly, I think, I've been a big believer that AGI undersells what we are collectively building because we already have, all of this sparks of super intelligence. I don't believe that v0 is an AGI if it knows everything about how to draw a dashed line according to the curvature of the earth and this high-performance map of airplanes. That's just superhuman. And yeah, it's a joy to use.

中文翻译:

也许 Vercel 最顶尖的工程师在不使用任何 AI 的情况下，可以在几天内搞定。但设计怎么办？我怎么办？因为我是瓶颈，而不是工程师。这就是这种协作的神奇之处，因为我提供产品指导。我说：“在两个目的地机场之间画一条虚线……” 顺便说一下，v0 让我大受震撼。我说：“在两个机场之间画一条虚线。” v0 说：“好吧，我必须考虑到球体，或者叫伪球体，考虑到地球的曲率。” 我当时想：“好吧，v0，你真是个超级天才。” 这就是我提到的如何来回沟通。它就像一个产品副驾驶，一个无所不知的存在。

(00:43:40):

昨天我们的一位用户提交了反馈，是正向反馈。他们非常开心，他们说：“v0 就像一个患有多动症的五岁天才博士。” 当然，我不会过度推销它，说它“无所不知，给出的东西都完美无缺”。但它确实有这些天才的火花。真的，我一直认为 AGI（通用人工智能）这个词低估了我们正在共同构建的东西，因为我们已经拥有了所有这些超智能的火花。如果 v0 知道如何根据地球曲率画虚线，并能处理高性能的飞机地图，我不认为它只是 AGI，那是超人类的表现。是的，使用它是一种乐趣。

[00:44:39] MUSIC

(instrumental music)

（器乐乐曲）

[00:44:40] Lenny Rachitsky (Sponsor: LinkedIn Ads)

English:

Today's episode is brought to you by LinkedIn ads. One of the hardest and also most important parts of B2B marketing is reaching the right people. I'm constantly getting ads for products that I will never buy. When you're ready to reach the right professionals use LinkedIn ads. LinkedIn has grown to a network of over one billion professionals, including 130 million decision makers. You can target your ad buyers by job title, industry, company, role, seniority, skills, even company revenue. Stop wasting budget on the wrong audience and start targeting the right professionals only on LinkedIn ads. LinkedIn will even give you $100 credit on your next campaign so that you can try it for yourself. Just go to linkedin.com/podlenny.

中文翻译:

今天的节目由 LinkedIn Ads 赞助。B2B 营销中最难也最重要的部分之一就是触达正确的人群。我经常收到一些我永远不会购买的产品广告。当你准备好触达正确的专业人士时，请使用 LinkedIn Ads。LinkedIn 已发展成为拥有超过 10 亿专业人士的网络，其中包括 1.3 亿决策者。你可以根据职位、行业、公司、角色、资历、技能甚至公司收入来定位你的广告受众。停止在错误的受众身上浪费预算，开始在 LinkedIn Ads 上针对正确的专业人士进行投放。LinkedIn 甚至会为你的下一次营销活动提供 100 美元的抵用金，让你亲自尝试。只需访问 linkedin.com/podlenny 即可。

[00:45:48] Lenny Rachitsky

English:

As you talk, it's interesting, the way I'm thinking about this now, there's almost like three core skills in building apps with AI. There's figuring out what to build, there's making it look good, like design, and then there's getting it unstuck. And coaching it. Yeah. Or just like, "Oh, here's the database error. I don't know. It's not figuring it out."

中文翻译:

听你这么说很有趣，我现在思考的方式是，用 AI 构建应用几乎有三项核心技能：确定构建什么、让它看起来漂亮（即设计）、以及解决卡壳问题。还有指导它。是的，或者就像：“噢，这里有个数据库错误，我不知道怎么回事，它没解决掉。”

[00:46:15] Guillermo Rauch

English:

Oh, absolutely. In fact, I'll tell you a little bit of a story of something. So even going way back in time, Next.js builds on React. React was this UI component library that Facebook created, actually with very similar goals. They had so many cracked engineers, and they had to help them collaborate on an enormous product surface. So they invented or at least pioneered, I would say the concept of this component as a unit of reusability, as a building block, as a Lego brick of how you build software. It's no coincidence that LLMs love to work with React components, by the way. And one of the things that always has stood out to me about that model is it basically enables people to scale in how they work together. And one of the key design principles that they embedded into this thing, is they called it escape hatch.

(00:47:54):

The API, when when React doesn't perfectly model your problem with its component system, they give you escape hatch. They say, "Okay, engineer. You are on your own now. There's no guardrails." And in fact, one of these escape hatches is called dangerously set inner HTML. They want the developer to know uncharted territory. But they did give people the API. That is a profound systems design engineering principle. And throughout my life, I've always thought about escape hatches.

(00:47:54):

One amazing escape hatch that v0 has is that you're looking at the code that we're generating with Next.js. You can edit it, you can even have other experts look at it. One thing that one of our demos last night came from this awesome company, Lumalabs. They're creating one of the most amazing video models in the world, and they use v0 and Vercel extensively to build their application, their websites, et cetera. And the design engineer was talking about how he was on a v0 that had 120 or so iterations. So he was knee-deep into the latent space. He was in the matrix. And at one point he got stuck. But you know what he did? He copied and pasted the code that we generated and he gave it to ChatGPT o1 and ChatGPT o1 thought about the solution.

(00:48:51):

Honestly, I'd never even thought about this myself. I was so blown away. And it does speak to, I love that your third point of, "You need to learn a skill of how to get unstuck." It's like a profound life lesson. It's just more a generic life advice you need to get. Facebook actually had a principle, "Don't get blocked. Seek to get unblocked, seek help from other people." What's fascinating is that you can seek help from other AIs to get unstuck. And those escape hatches of actually understanding and seeing the code underneath, and even being able to say, "Okay, now let's use Git. Let's turn this into more of a hybrid project, not just prompts, but also traditional software engineering." The fact that that door is open to you is extremely valuable.

中文翻译:

噢，绝对如此。事实上，我给你讲个故事。追溯到很久以前，Next.js 是建立在 React 之上的。React 是 Facebook 创建的一个 UI 组件库，其实目标非常相似。他们有那么多顶尖工程师，必须帮助他们在巨大的产品表面上进行协作。所以他们发明（或者说开创）了“组件”这个概念，将其作为可重用的单元、构建块，就像构建软件的乐高积木。顺便说一下，LLM 喜欢处理 React 组件并非巧合。那个模型一直让我印象深刻的一点是，它基本上让人们在协作方式上实现了规模化。他们嵌入其中的一个关键设计原则叫做“逃生舱”（escape hatch）。

(00:47:54):

当 React 的组件系统无法完美建模你的问题时，API 会给你一个逃生舱。他们会说：“好了，工程师，现在靠你自己了，没有护栏了。” 事实上，其中一个逃生舱就叫 `dangerouslySetInnerHTML`。他们想让开发者知道这是进入了未知领域。但他们确实给了人们这个 API。这是一个深刻的系统设计工程原则。在我的一生中，我一直在思考逃生舱。

(00:47:54):

v0 拥有的一个神奇逃生舱是：你可以看到我们用 Next.js 生成的代码。你可以编辑它，甚至可以让其他专家查看它。昨晚我们的一个演示来自一家很棒的公司 Lumalabs。他们正在创建世界上最惊人的视频模型之一，他们广泛使用 v0 和 Vercel 来构建他们的应用、网站等。那位设计工程师谈到他如何在一个经历了 120 次左右迭代的 v0 项目上工作。他深陷于潜空间（latent space）之中，就像在《黑客帝国》里一样。在某个时刻他卡住了。但你知道他做了什么吗？他复制粘贴了我们生成的代码，把它交给了 ChatGPT o1，然后 ChatGPT o1 思考出了解决方案。

(00:48:51):

老实说，我自己都没想过这一点，我被震撼到了。这确实印证了你提到的第三点：“你需要学习如何解决卡壳问题的技能。” 这就像是一个深刻的人生教训，也是一种通用的生活建议。Facebook 实际上有一个原则：“不要被堵住。寻求解除阻塞，寻求他人的帮助。” 迷人之处在于，你可以寻求其他 AI 的帮助来解决卡壳问题。而那些能够让你真正理解并看到底层代码的“逃生舱”，甚至能让你说：“好了，现在让我们使用 Git，把这变成一个混合项目，不仅有提示词，还有传统的软件工程。” 扇门向你敞开的事实极其宝贵。

[00:49:59] Lenny Rachitsky

English:

Let's actually make the super concrete and show people what this actually looks like in v0. So pull up, we'll share screen, and then we'll do a little live demo. We'll keep it brief. I find people are like, "Okay, I get it." But we'll make it fun and brief at the same time. There it is. I see it.

中文翻译:

让我们把它变得超级具体，向大家展示 v0 到底是什么样子的。来吧，我们共享屏幕，做一个简短的现场演示。我们会保持简短，我发现人们通常会说：“好吧，我懂了。” 但我们会让它既有趣又简短。看到了，我看到了。

[00:50:02] Guillermo Rauch

English:

Beautiful. Okay. So as I mentioned, you write in English, you yap into the tool. I'll say for a demo, let's create a contact sales form in the style of ... By the way, I had a typo. I don't care. Let's get it. It's Supreme, the clothing company for an online store. Now, I mentioned that sometimes people get blocked, there is a writer paralysis at this step. So we added enhanced prompt. So now you're tapping into the latent space of the model, which has a random component to it.

(00:50:48):

And by the way, this is still not a substitute. It doesn't contradict what I said earlier, knowing the meaningful tokens, knowing what the right style is, and what it's called, and whatever is still highly valuable. So the first thing you're going to notice is that as the model thinks, you can introspect its thinking. So we added this recently. It's been mostly inspired actually by the Deepseek revolution. I would say.

(00:51:17):

So the fact that when you tell it, "Develop a contact sales form," what is it going to do? We talked about escape hatches. Okay, it's going to use shadcn/ui, it's going to use Tailwind CSS, it's going to use React. And this is your opportunity that if v0 is not doing exactly what you wanted, this is your opportunity to actually go and correct, or influence, or give feedback and so on. So you're going to notice it spits out a bunch of files, and it gives me the thing that I wanted. I'm going to zoom out a little bit, here. A couple of things that stand out here that again, as an experienced engineer, I can point out. The underlying component system that it uses is the same component system that the best tools on the planet are built with. This is called shadcn. If you go to grok.com today, they're using shadcn to build their UI. They're using Next.js. You're getting that caliber of code.

(00:52:15):

The other thing that it did is people on social media talk about this a lot. When you use a global shared component system with the world, you don't want everything to look the same. So the fact that he was able to apply the style and he kind of knew what supreme looked like was kind of cool. But now I'm going to say, "Actually, because I am building a financial institution, make it more serious, make it in the style of let's say Charles Schwab. Change typefaces." So this is the iteration process of like, "I'm going to go and give feedback to the model. I'm going to make it try different things." So once that initial generation was already created, now the model is actually acting more as an editor. It's going and making tweaks to what's already been built.

中文翻译:

太棒了。好的。正如我提到的，你用英语写，对着工具“唠叨”。我演示一下，让我们创建一个销售联系表单，风格模仿……顺便说一下，我打错字了，但我不在乎。让我们开始吧。模仿 Supreme，那家服装公司，为一个在线商店制作。现在，我提到过有时人们会卡住，在这一步会有“写作瘫痪”。所以我们添加了“增强提示词”（enhanced prompt）。现在你正在接入模型的潜空间，它带有一种随机成分。

(00:50:48):

顺便说一下，这仍然不是替代品。它并不矛盾于我之前说的：了解有意义的词汇（tokens）、了解正确的风格叫什么等等，仍然非常有价值。所以你首先会注意到，当模型思考时，你可以内省它的思考过程。这是我们最近添加的功能，实际上主要是受 Deepseek 革命的启发。

(00:51:17):

当你告诉它“开发一个销售联系表单”时，它会做什么？我们谈到了逃生舱。好的，它将使用 shadcn/ui，使用 Tailwind CSS，使用 React。如果 v0 做的不是你完全想要的，这就是你去纠正、影响或提供反馈的机会。你会注意到它吐出了一堆文件，并给了我想要的东西。我稍微缩小一点。这里有几点作为一个资深工程师可以指出的：它使用的底层组件系统与地球上最好的工具所使用的系统是一样的。这叫 shadcn。如果你今天去 grok.com，他们就在用 shadcn 构建 UI，用的是 Next.js。你得到的是这种水准的代码。

(00:52:15):

另一件事是——社交媒体上的人经常讨论这个——当你使用全球共享的组件系统时，你不希望所有东西看起来都一样。所以它能够应用这种风格，并且它大概知道 Supreme 长什么样，这很酷。但现在我要说：“实际上，因为我正在建立一家金融机构，让它更严肃一点，模仿查尔斯·施瓦布（Charles Schwab）的风格。更换字体。” 这就是迭代过程：“我要去给模型反馈，让它尝试不同的东西。” 一旦初始生成完成，现在模型实际上更多地充当编辑器的角色，对已经构建好的东西进行微调。

[00:53:08] Guillermo Rauch (Demo continued)

English:

And this actually scales to very large projects. You could have started with something much bigger. So in the meantime, I'm going to show you what Lumalabs created with v0, which is absolutely phenomenal. I learned about this last night. It already has 2,000 forks. I was telling you about the power of our community. So by the way, you can just click community here on the v0 sidebar. I'm going to fork it, because they generously shared it with the world. Notice all the incredible animations here? By the way, they shipped this to hire and attract talent to their company. Notice that it's an interactive, everything is AI generated, they used their own AI image generation tool to create these beautiful frames. These are all AI generated as well.

中文翻译:

这实际上可以扩展到非常大的项目。你可以从更大的东西开始。与此同时，我向你展示一下 Lumalabs 用 v0 创建的东西，绝对是现象级的。我昨晚才了解到这个，它已经有 2000 个派生（fork）了。我刚才跟你说过我们社区的力量。顺便说一下，你只需点击 v0 侧边栏的“社区”即可。我要派生它，因为他们慷慨地与世界分享了它。注意这里所有不可思议的动画了吗？顺便说一下，他们发布这个是为了招聘和吸引人才。注意它是交互式的，所有东西都是 AI 生成的，他们使用自己的 AI 图像生成工具创建了这些漂亮的帧。这些也全都是 AI 生成的。

[00:53:59] Lenny Rachitsky

English:

Wow.

中文翻译:

哇。

[00:54:00] Guillermo Rauch

English:

And it's interactive. So there is the autoplaying functionality. This is actually a complex layout in animation system that they built entirely in v0. I was telling you that at one point they even got some advice from O-Wan (o1), so shout out to OpenAI. I'm going to say, "Make it sepia style colors." So this is an example of like, "Okay, I forked something. I already have a starting point." My bank grade contact form is ready. In the meantime, another fun thing to do is you can start with a screenshot. So I'll use another Next.js user as an example, which is fortune.com. Shout out to them. They built a slick website.

(00:54:48):

But let's say that I'm actually wanting to break into the news business, so I'm just going to paste a screenshot. I could have also attached the Figma file. And I'm going to have, v0 already knows, v0 can answer questions as well about the engineering design product world. So I can ask v0, "What is a newsletter? Explain with a diagram. Use Lenny as an example." So v0 is also a knowledge seeking tool. But we do strongly like, "Steer the tool to create things." So if I paste a screenshot, as you can see, it's cooking on creating a hopefully awesome news website. I specifically asked, because I think it's funny, to explain a newsletter with a diagram, so v0 can create again, explanations, content, knowledge. The creator is Lenny, you were a former Airbnb product lead. I guess I should have used some examples from Airbnb, by the way. But let's look at here what it created with Fortune.

中文翻译:

它是交互式的，有自动播放功能。这实际上是他们完全在 v0 中构建的一个复杂的布局和动画系统。我刚才说过，他们在某个阶段甚至得到了 o1 的建议，所以向 OpenAI 致敬。我要说：“把它改成深褐色（sepia）风格。” 这是一个例子：“好吧，我派生了一个东西，我已经有了一个起点。” 我的银行级联系表单已经准备好了。与此同时，另一件有趣的事是你可以从截图开始。我以另一个 Next.js 用户为例，即 fortune.com（财富杂志官网）。向他们致敬，他们建了一个很漂亮的网站。

(00:54:48):

但假设我想进军新闻业务，所以我直接粘贴一张截图。我也可以附上 Figma 文件。v0 已经知道了，它也可以回答关于工程、设计、产品领域的问题。所以我可以问 v0：“什么是时事通讯（newsletter）？用图表解释一下，以 Lenny 为例。” 所以 v0 也是一个知识获取工具。但我们非常强调“引导工具去创造东西”。所以如果我粘贴一张截图，如你所见，它正在努力创建一个（希望是）很棒的新闻网站。我特意问了一个我觉得很有趣的问题，用图表解释 newsletter，所以 v0 可以再次创建解释、内容和知识。创建者是 Lenny，你曾是 Airbnb 的产品主管。顺便说一下，我想我应该用一些 Airbnb 的例子。但让我们看看它根据《财富》杂志生成的网页。

[00:56:01] Lenny Rachitsky

English:

Wow.

中文翻译:

哇。

[00:56:03] Guillermo Rauch

English:

So notice that, I'm just noticing now the sidebar should have been on the center. I'm going to zoom out a little bit. Let's use the refinement tool to center this. I call this, by the way, one of the hardest problems in computer science is actually centering things.

中文翻译:

注意看，我现在才注意到侧边栏应该在中间。我缩小一点。让我们使用微调工具把它居中。顺便说一下，我把这称为计算机科学中最难的问题之一：实际上就是把东西居中。

[00:56:26] Lenny Rachitsky

English:

With CSS.

中文翻译:

用 CSS 居中。

[00:56:28] Guillermo Rauch

English:

That's right, centering a div. And in fact, look at it. It was a div. So notice that I did a precise inline prompt? And the difference between v0 and a lot of other tools is that yes, you do have the code and code is very important, but I call it code last rather than code first. You're living in the product. So center that. Another website that I love also built with Next.js is Semaphore. So I really like their sepia style. So I'm going to say, "Apply this style instead, including- "

中文翻译:

没错，居中一个 `div`。事实上，看，它就是一个 `div`。注意到我做了一个精确的行内提示词吗？v0 与许多其他工具的区别在于，虽然你确实拥有代码且代码很重要，但我称之为“代码最后”（code last）而不是“代码优先”（code first）。你沉浸在产品中。所以，把它居中。另一个我也很喜欢的用 Next.js 构建的网站是 Semaphore。我非常喜欢他们的深褐色风格。所以我要说：“改为应用这种风格，包括——”

[00:57:09] Lenny Rachitsky

English:

So you're sharing a screen. So you used a screenshot to design, to build a site, and now you're using a different screenshot to tell it, "Make it look like this." Very cool.

中文翻译:

所以你正在共享屏幕。你先用一张截图来设计和构建一个网站，现在你又用另一张不同的截图告诉它：“让它看起来像这样。” 非常酷。

[00:57:21] Guillermo Rauch

English:

Yes. And so the idea is that v0 can Grok different aspects of what it needs to build. It can be functional aspects, it can be layout aspects. And one of the things that's also very important to know is we influence the model. So a lot of the things that you would have had to prompt you might get for free. One that's important to call out is responsiveness. So as an example, if I notice that if I do this, it's going to make it work quite well on mobile, it's going to give me that hamburger menu. I can now tell it like, "Apply that style to everything."

(00:58:00):

In the meantime, I'll show you, this is actually to me very, very impressive. And I don't know why today I'm so fixated on the theme of sepia. But notice that not only did it change the background, I hope people can notice this. It applied it to the checkboxes and it applied a CSS. I'm assuming this is a CSS filter. Yeah, it applied a CSS filter. Just for the sake of it, because I'm a nerd, I'm going to look at it. But yes, it applied a CSS filter. Confession time, I actually didn't know that there was a sepia function in the filter property of CSS. There were many ways to accomplish this. You could have also written the images or the videos to a canvas, and apply all kinds of algorithms, and whatever.

中文翻译:

是的。所以核心想法是 v0 可以理解（Grok）它需要构建的不同方面。可以是功能方面，也可以是布局方面。还有一点非常重要：我们会影响模型。所以很多你原本需要写提示词的东西，现在可能免费就能得到。其中一个值得强调的是响应式设计。例如，如果我注意到这样做能让它在移动端运行良好，它会给我那个汉堡菜单。我现在可以告诉它：“把这种风格应用到所有东西上。”

(00:58:00):

与此同时，我向你展示一下，这对我来说真的非常非常令人印象深刻。我不知道为什么今天我如此痴迷于深褐色主题。但请注意，它不仅改变了背景（我希望大家能注意到这一点），它还把它应用到了复选框上，并应用了一个 CSS。我猜这是一个 CSS 滤镜（filter）。是的，它应用了一个 CSS 滤镜。为了严谨起见，因为我是个技术宅，我要看一下代码。没错，它应用了一个 CSS 滤镜。坦白说，我其实不知道 CSS 的 `filter` 属性里有一个 `sepia` 函数。实现这个效果有很多方法，你也可以把图像或视频写入 Canvas，然后应用各种算法等等。

[00:58:48] Lenny Rachitsky

English:

I like that it did more elegantly than you would have.

中文翻译:

我喜欢它做得比你亲自动手还要优雅这一点。

[00:58:51] Guillermo Rauch

English:

Yeah, exactly. So that's why you can't be too opinionated with the tool. So another cool thing is I do like showing screenshots, but I do want to remind people that the idea is not to clone other people's websites, necessarily. Right? It's just a cool demo. It's a simple way to show off what it can do. Take screenshots of your own things. Take screenshots of your art boards, take screenshots of things that people post in Slack, and also don't hesitate add functionality.

中文翻译:

是的，没错。所以这就是为什么你不能对工具太有成见。另一件酷的事是，我确实喜欢展示截图，但我确实想提醒大家，核心想法不一定非要克隆别人的网站，对吧？这只是一个很酷的演示，是展示它能做什么的一种简单方式。拍下你自己的东西的截图，拍下你的画板截图，拍下人们在 Slack 里发布的东西的截图，并且不要犹豫去添加功能。

[00:59:22] Lenny Rachitsky

English:

Incredible. Thank you for doing the demo. I'm just trying to imagine having an engineer I'm working with, asking them to do these things, and not only just how annoying that would be, like, "Make it sepia." But just how much time it would take from, "Okay, do this thing, copy fortune.com." It'd be like days, weeks. Here, it's just months. If ever. Maybe it never ships.

中文翻译:

不可思议。谢谢你做的演示。我只是在想象如果我正和一个工程师合作，要求他们做这些事，不仅是那有多烦人（比如“把它改成深褐色”），还有那会花多少时间。从“好吧，做这件事，复制 fortune.com”开始，那可能需要几天、几周。在这里，（如果用传统方式）可能要几个月。甚至永远无法交付，也许永远发布不了。

[00:59:44] Lenny Rachitsky

English:

That's right. Well, something that I noticed that I loved at the beginning when you were doing the prompting and that prompt improvement feature is it basically is best practices to make it look good and look better. Which I think is one of the more interesting, I don't know, levers to working with AI is it just has best practices to help you build things that are beautiful and also feels like there's this opportunity of just helping you figure out if what you're building is at all a good idea. "What is the problem you're trying to solve?" It feels like there's a PM 1-pager step that should exist. Like, "How do you know this is a problem? What have users told you? How many people have told you this?" Things like that.

中文翻译:

没错。我在开始看到你写提示词和那个提示词改进功能时，我非常喜欢的一点是，它基本上是利用最佳实践来让它看起来更好。我认为这是与 AI 合作中更有趣的杠杆之一：它拥有最佳实践来帮助你构建美观的东西。而且感觉这里还有一个机会，就是帮助你弄清楚你正在构建的东西是否真的是个好主意。“你试图解决什么问题？” 感觉应该存在一个 PM 的单页文档（1-pager）步骤。比如：“你怎么知道这是一个问题？用户告诉了你什么？有多少人告诉过你这个？” 诸如此类。

[01:00:24] Guillermo Rauch

English:

Yeah. There's something to be said about the fact that over time we're more and more peeking into the mind of the AI. That in itself is becoming a killer feature. So the Deepseek stream, the thinking tokens moment was a very big moment for our industry, I think. Because OpenAI did have the technology, but they decided that for competitive reasons, which, it's a reasonable think to think, no pun intended, they were going to withhold it. And also it wasn't clear that there was going to be product end user and product utility. But when Deepseek hit, it was very obvious that people really liked the idea of understanding how the AI thinks and influencing where it should go. We've gotten actually amazing feedback and bug reports where people actually specifically point out, "Look, this is where the AI went

wrong. Please fix it." So the more people we get on this product, the more thumbs up, thumbs down, the more user feedback we get.

(01:01:30):

And by the way, I'll tell you for people out there building products, my number one guidance or piece of advice I would give to any startup founder was, "Create a lot of opportunities for people to give you feedback inside the product." I drew inspiration from Stripe. And this was amazing for the early days of Vercel, there was a feedback button with a very slick inline form, with four emojis that would allow you to decide how you were feeling about the feature, the product at that very moment. And that would go straight into Slack. And we were building day in and day out, just streaming users' thoughts right into our consciousness. And maybe we would get, I don't know, tens, hundreds a day, especially the early days, maybe a couple a day and whatever. When you're building AI products, it's a constant stream of user feedback. So for people that are thinking about not building AI products, it's going to be hard to compete with something that has such a tight feedback loop with users.

(01:02:40):

The whole idea is to capture users' feedback so the next iteration of the model, the prompt, the fine-tuning, the examples, the rag is better. And one of the things that Vercel has done as a result of this insight is we've open-sourced a lot of what makes v0 work. So let's say that you wanted to create the v0 for doctors as an example. You can go to vercel.com/templates, and you can clone a ChatGPT template that basically follows all of the best practices in the world for really high-performance, awesome UIs, and now you can go out and build your own AI products. We've also open-sourced the AI SDK, which is the foundational plumbing of v0. It allows you to connect any model and generate UI from its responses, not just output text, but actually generate UI. So maybe because I love showing stuff, I'll just really quickly show you this.

中文翻译:

是的。值得一提的是，随着时间的推移，我们越来越多地窥探到 AI 的内心。这本身正在成为一个杀手级功能。我认为 Deepseek 的流式输出、那些“思考标记”（thinking tokens）的时刻，对我们行业来说是一个非常重大的时刻。因为 OpenAI 确实拥有这项技术，但他们出于竞争原因（这是一个合理的想法，无意双关）决定保留它。而且当时也不清楚这是否会对最终用户和产品效用产生影响。但当 Deepseek 冲击市场时，很明显人们非常喜欢了解 AI 是如何思考的，并影响它的走向。我们实际上收到了惊人的反馈和错误报告，人们会具体指出：“看，这就是 AI 出错的地方，请修复它。” 所以，使用这个产品的人越多，我们得到的点赞、踩和用户反馈就越多。

(01:01:30):

顺便说一下，对于那些正在构建产品的人，我要给任何初创公司创始人的头号指导或建议是：“在产品内部创造大量让人们给你反馈的机会。” 我从 Stripe 那里汲取了灵感。这对 Vercel 的早期阶段非常有帮助：有一个反馈按钮，配有一个非常漂亮的行内表单，有四个表情符号，让你决定那一刻对该功能或产品的感受。这些反馈会直接进入 Slack。我们日复一日地构建，用户的想法就像流一样直接进入我们的意识。早期可能每天只有几个，后来可能有几十个、几百个。当你构建 AI 产品时，用户反馈是源源不断的。所以对于那些考虑不构建 AI 产品的人来说，很难与拥有如此紧密用户反馈闭环的产品竞争。

(01:02:40):

核心思想是捕捉用户反馈，以便模型、提示词、微调、示例和 RAG（检索增强生成）的下一次迭代变得更好。基于这一洞察，Vercel 做的一件事就是开源了许多让 v0 运作的技术。假设你想为医生创建一个 v0。你可以去 vercel.com/templates，克隆一个 ChatGPT 模板，它基本上遵循了世界上所有关于高性能、出色 UI 的最佳实践，现在你就可以去构建自己的 AI 产品了。我们还开源了 AI SDK，这是 v0 的基础架构。它允许你连接任何模型，并根据其响应生成 UI，而不仅仅是输出文本。因为我喜欢展示东西，所以我快速给你看下这个。

[01:03:45] Guillermo Rauch (Demo: AI SDK)

English:

So if you go to chat.vercel.ai super quick, you're going to see this is the open-source ChatGPT demo that we've built. You can ask questions like old-school LLM. But also, you can ask, let's actually finish this, let's ask, "What is the weather in San Francisco?" We call this generative UI. It's responding not with just plain text, it's creating components as a result. Last but not least, and this is a v0 style opportunity, let's ask it to help me write an essay about Silicon Valley. It's going to create a canvas or artifacts style experience, and everything is generative, but also users can edit, refine, et cetera, et cetera, et cetera.

中文翻译:

如果你快速访问 chat.vercel.ai，你会看到这是我们构建的开源 ChatGPT 演示。你可以像使用传统 LLM 一样提问。但你也可以问——让我们完成这个——问：“旧金山的天气怎么样？” 我们称之为“生成式 UI”（Generative UI）。它不仅用纯文本回答，还因此创建了组件。最后但同样重要的是，这是一个 v0 风格的机会，让我们要求它帮我写一篇关于硅谷的文章。它将创建一个 Canvas 或 Artifacts 风格的体验，所有东西都是生成的，但用户也可以编辑、微调等等。

[01:04:36] Lenny Rachitsky

English:

This actually reminds me of something I've been thinking about. There's all these startups that are building vertical AI tools. This is a little bit of a tangent, and there's always this AI stuff for lawyers, AI stuff for doctors, nurses, and the pitch there is that these are going to be founders that know a lot about the specific problem in this useless market, and so they'll build the tools that are very specific to them.

中文翻译:

这实际上让我想起了一些我一直在思考的事情。现在有很多初创公司在构建垂直领域的 AI 工具。这有点跑题了，总是有针对律师的 AI、针对医生和护士的 AI，他们的卖点是：创始人非常了解这个特定市场的具体问题，因此他们会构建非常针对性的工具。

[01:04:58] Guillermo Rauch

English:

Yeah, I'm absolutely convinced that expert AI tools are the future. There's an amazing product being built on Vercel called chatprd.com. It's the v0 for writing PRDs and it's going to get a v0 integration soon so that you can write your PRD with AI and then you can create it with AI. That's just an example of a vertical that you can go after. There's also OpenEvidence. It's like the ChatGPT for doctors, actually. There is an amazing startup building x-ray AI tooling. So the ideas I think are infinite, and what I've seen from users of AI at Vercel, for example, our legal team loves this tool called GetGC.AI. They could in theory go to ChatGPT to ask legal questions, but someone out there decided, "I'm going to build the best legal AI tool in the world. It's going to be up to date. I'm going to obsess about this problem." The CEO herself is a lawyer, so it's going to be hard to compete with that, I think.

中文翻译:

是的，我绝对相信专业的 AI 工具是未来。在 Vercel 上有一个正在构建的惊人产品叫 chatprd.com。它是用于编写 PRD 的 v0，很快就会有 v0 集成，这样你就可以用 AI 编写 PRD，然后用 AI 创建它。这只是你可以追求的一个垂直领域的例子。还有 OpenEvidence，它实际上就像是医生的 ChatGPT。还有一家很棒的初创公司在构建 X 光 AI 工具。所以我认为创意是无限的。我从 Vercel 的 AI 用户那里看到的，例如我们的法务团队非常喜欢一个

叫 GetGC.AI 的工具。理论上他们可以去 ChatGPT 问法律问题，但有人决定：“我要构建世界上最好的法律 AI 工具，它将保持最新，我会痴迷于解决这个问题。” CEO 本人就是一名律师，所以我认为很难与之竞争。

[01:06:03] Lenny Rachitsky

English:

But here's what I'm thinking. This is almost the opposite and I'm curious to get your take, but let's not spend too much time on this, because this is a complete change in- So you showed me the weather widget that you just built, basically it's like a little mini app that the AI built as you're talking to it. Is there a world where when AI, when AGI is far enough and approaching super intelligence? Can it just build you a Harvey, for example, in real time? "Here's the best experience for a lawyer. Here we go. We got it for you."

中文翻译:

但这是我的想法，这几乎是相反的，我很想听听你的看法，但我们不要在这上面花太多时间，因为这是一个完全的转变——你刚才向我展示了你构建的天气组件，基本上就像是 AI 在你说话时构建的一个小程序。是否存在这样一个世界：当 AI、当 AGI 足够发达并接近超智能时，它能实时为你构建一个像 Harvey（法律 AI）那样的东西？ “这是给律师的最佳体验，拿去吧，我们为你做好了。”

[01:06:31] Guillermo Rauch

English:

Totally, totally. I believe that eventually, yes, but humans will always want to have some guardrails. The reality is that GetGC is taking a double job. One is making the best tools for lawyers possible, but also putting their weight behind it, saying, "We've actually used this and we believe that this is what the future should look like." There is a sense of direction and opinion about things and I think left to its own devices, AI, I don't know, this is the double-edged like prompt embellishment. AI doesn't always know exactly what we want or what we need. It's still very much a copilot, a partner, an assistant. It's not really running our lives, and I don't know that we even would want that, ultimately.

中文翻译:

完全可能。我相信最终会实现的，但人类总会想要一些护栏。现实情况是，像 GetGC 这样的公司承担了双重任务：一是尽可能为律师制作最好的工具，二是为其背书，说：“我们确实用过这个，我们相信这就是未来的样子。” 这其中包含了一种方向感和对事物的见解。我认为如果任由 AI 自由发挥——我不知道，这就是提示词润色这种双刃剑——AI 并不总是确切知道我们想要什么或需要什么。它在很大程度上仍然是一个副驾驶、一个伙伴、一个助手。它并没有真正掌控我们的生活，我不知道我们最终是否真的想要那样。

[01:07:23] Lenny Rachitsky

English:

Okay, I'm going to go in a whole different direction, which is taste. We hear this word taste all the time. It feels like a thing that people are always suggesting. This will continue to be an important skill, to know what is good, basically to know what people are likely to find valuable and good. And I know clearly you have great taste. You're building incredibly beautiful products, v0's clearly, it's like the most beautiful by default builder out there, as we've seen. So in terms of building taste, people are always like, "How the hell do I do that? I have great taste. I know I do. I don't need to." How have you built taste? How do you think you build taste and any advice for folks that are trying to improve their taste?

中文翻译:

好的，我要转向一个完全不同的方向，那就是“品味”（taste）。我们一直听到这个词。感觉人们总是在建议：这将继续是一项重要的技能——知道什么是好的，基本上就是知道人们可能觉得什么是有价值和好的。我知道你显然很有品味，你正在构建极其漂亮的产品，正如我们所见，v0 显然是目前默认最漂亮的构建工具。那么在培养品味方面，人们总是问：“我到底该怎么做？我有很好的品味，我知道我有，我不需要培养。” 你是如何培养品味的？你认为该如何培养品味？对那些试图提高品味的人有什么建议吗？

[01:08:03] Guillermo Rauch

English:

Yes, I think it's extremely important to try lots of products. You need to get yourself out there. I think it's very important to go back to that, get into the world, ship things. Don't be hesitant of self-promotion in a way. So being very honest with yourself, building something, getting it out there, see how people react. Go back to the drawing board. I think it's about exposure. At Vercel we have one of our internal operating principles as increasing exposure hours. Try to quantify how much time you expose yourself to watching how people use your products, even to watch how people use other products, and you'll develop that muscle. Taste, sometimes I think we think of as this inaccessible thing that, "Oh, that person was born with taste." I see it as a skill that it can develop.

(01:09:07):

And again, the AI will help you a lot here because we try and capture some of the universal principles of it. But there's also trends in the world. I'm not a super couture guy, but you can see that every year Paris Fashion Week has a theme to it and there is some innovations, there have some breakthroughs, whatever. And so trying to stay at the frontier or even try and define the frontier as well is certainly very exciting.

中文翻译:

是的，我认为尝试大量产品极其重要。你需要让自己走出去。我认为回到那一点非常重要：进入真实世界，交付东西。在某种程度上，不要犹豫去自我推销。对自己保持诚实，构建一些东西，把它发布出去，看看人们的反应。然后回到绘图板前。我认为这关乎“暴露”（exposure）。在 Vercel，我们的内部运营原则之一就是增加“暴露时长”（exposure hours）。试着量化你花在观察人们如何使用你的产品，甚至观察人们如何使用其他产品上的时间，你就会练就这种肌肉记忆。品味，有时我们认为它是某种不可企及的东西，觉得“噢，那个人天生就有品味”。但我认为它是一种可以培养的技能。

(01:09:07):

此外，AI 在这方面会给你很大帮助，因为我们试图捕捉一些通用的设计原则。但世界上也有趋势。我不是一个超级时尚达人，但你可以看到每年巴黎时装周都有一个主题，会有一些创新、一些突破。因此，努力站在前沿，甚至尝试去定义前沿，当然是非常令人兴奋的。

[01:09:42] Lenny Rachitsky

English:

I love how doable this is, increasing your exposure hours. Basically what I'm hearing is, "Use the best apps." There's a feedback cycle component to it. Just like, "Show people the thing."

中文翻译:

我喜欢这种可操作性，增加你的暴露时长。基本上我听到的是：“使用最好的应用。” 还有一个反馈循环的组成部分，就像：“把东西展示给人们看。”

[01:09:54] Guillermo Rauch

English:

And understand these nuances. Right? So I actually recently created, I published it to my community, v0. I created a ChatGPT style interface inspired by Grok. And I captured a few things that Grok does that are just so smart. So on mobile web, when you press enter on their input, they default to creating a new line. Because they know that the way that people are used to submitting things on mobile is not by hitting enter, like we would do on a desktop computer. You can tap the little icon and your message goes out. On desktop, they inverted it. When you press enter, you're expected to submit. And I think if you got a new line, I think a lot of people would get frustrated that most people don't know that they can press command, enter to submit and whatever, and it slows everything down. And you can basically prompt for those things.

(01:10:48):

But you have to pay attention to the details and you have to decide what you want to see in the world. And sometimes that means either defining best practices, or seeking the best practices, and learning from others. Another aspect of exposure hours is that you tend to overrate how well your products work. It's very important to give your product to another person and watch them interact with it, expose yourself to the pain of reality. And the more you submerge yourself in the real deal, nitty-gritty of what happens when people use your interfaces and whatnot, I think you you'll come out stronger, more grounded, hopefully more humbled.

中文翻译:

还要理解这些细微差别。对吧？所以我最近创建并发布到了我的 v0 社区一个受 Grok 启发的 ChatGPT 风格界面。我捕捉到了 Grok 做的几件非常聪明的事情。在移动端网页上，当你在输入框按回车时，他们默认是换行。因为他们知道人们在手机上提交内容的习惯不是按回车（不像我们在电脑上那样）。你可以点击那个小图标发送消息。在桌面端，他们反转了过来：当你按回车时，默认是提交。我想如果你在桌面端按回车是换行，很多人会感到沮丧，因为大多数人不知道可以按 Command+Enter 来提交，这会减慢所有操作。你基本上可以通过提示词要求实现这些细节。

(01:10:48):

但你必须关注细节，必须决定你想在世界上看到什么。有时这意味着要么定义最佳实践，要么寻找最佳实践并向他人学习。暴露时长的另一个方面是，你往往会高估自己产品的好用程度。把你的产品交给另一个人，观察他们如何与之交互，让自己暴露在现实的痛苦中，这非常重要。你越是沉浸在人们使用你的界面时发生的真实、琐碎的细节中，我认为你就会变得更强大、更务实，也希望变得更谦逊。

[01:11:34] Lenny Rachitsky

English:

We don't like pain, though, and I like that this is a push, "Create some more pain in your life. Show people the thing you're building." Do you have a heuristic or number of how many exposure hours per week, per month you want your team to have or is it just more is always better?

中文翻译:

虽然我们不喜欢痛苦，但我喜欢这种推动：“在你的生活中创造更多痛苦，把你在构建的东西展示给人们看。”对于你希望团队每周或每月有多少暴露时长，你有一个启发式的方法或具体的数字吗？还是说越多越好？

[01:11:48] Guillermo Rauch

English:

Yeah, I'm more is always better. I mean, because the inertia is to get inside your head, and the inertia is to think that you know everything, and assume that everything is going good, and, "There are no errors. Of course it's fast. It worked on my machine." I think it's always a push for more. I do sometimes little things like I asked my team to color my calendar. So I say I have to have a certain amount of one-on-ones with my team represented on my calendar, kind of like meetings so that I can sync with people and see how the company's doing. Then I want to have customer meetings. And during those customer meetings I push myself to use the products. In fact, with our enterprise customers, something that I do is I try to forget how things are built, what feature of Vercel they use and whatever. I just frequently use their products. And I want the product to be great, that's all. And then I could try to work backwards.

(01:12:49):

So a form of exposure hours for me is seeing what kind of success our customers are having in the real world. But again, it's just heuristic. Maybe one third of my meetings this week where customer meetings I tried and watched them do. Another really quick one is we invite people frequently to demo how they use the product live, sometimes to the executive team, sometimes to the whole company. And we always inevitably discover something interesting from the customer about maybe there is something that they're in pain about that we didn't know about, or maybe something was not as intuitive as we thought.

中文翻译:

是的，我认为越多越好。因为惯性会让你陷入自己的思维定式，惯性会让你认为自己无所不知，并假设一切进展顺利，“没有错误，当然很快，在我的机器上运行正常”。我认为必须不断推动更多。我有时会做一些小事，比如让团队给我的日历上色。我会说，我的日历上必须有一定比例的与团队的一对一沟通，就像会议一样，这样我可以与人同步，了解公司的情况。然后我想要有客户会议。在这些客户会议期间，我强迫自己使用产品。事实上，对于我们的企业客户，我会做的一件事是尝试忘记东西是怎么构建的，忘记他们使用了 Vercel 的什么功能。我只是频繁地使用他们的产品。我只想要产品好用，仅此而已。然后我可以尝试倒推。

(01:12:49):

所以对我来说，暴露时长的一种形式是观察我们的客户在现实世界中取得了什么样的成功。但这只是启发式的。也许我本周三分之一的会议是客户会议，我尝试观察他们操作。另一个很快的方法是，我们经常邀请人们现场演示他们如何使用产品，有时是向高管团队演示，有时是向全公司演示。我们总是不可避免地从客户那里发现一些有趣的事情，也许是他们感到痛苦但我们不知道的事情，或者也许某些东西并不像我们想象的那么直观。

[01:13:31] Lenny Rachitsky

English:

And I find with these sorts of things, when you do them, when you talk to customers, you have them show how they use the product. You always like, "Why have I not done this more often? What am I thinking?" It's just so mind-blowing usually.

中文翻译:

我发现这类事情，当你真正去做时，当你与客户交谈，让他们展示如何使用产品时，你总是会想：“为什么我没早点多做几次？我当时在想什么？” 这通常非常令人震撼。

[01:13:44] Guillermo Rauch

English:

Yes.

中文翻译:

是的。

[01:13:44] Lenny Rachitsky

English:

I want to talk about limitations of v0 at this point. So what should people know about just what v0 can't do? If you have an existing code base, can you plug it in and start doing stuff? Or is that coming? What else should people know? Just like, "Okay, it's not going to do this yet. But- "

中文翻译:

我想谈谈目前 v0 的局限性。人们应该了解 v0 有哪些做不到的地方？如果你有一个现有的代码库，你能把它接入并开始操作吗？还是说这个功能即将推出？人们还应该知道什么？比如：“好吧，它现在还做不到这个，但是——”

[01:13:59] Guillermo Rauch

English:

Yeah, you can import code bases through zip files and Git is coming very soon. It can do full stack development, it can connect to APIs. In the next couple of days, maybe even before this podcast is out, we'll have these very tight integrations so that if you need a database, or if you need an AI model, or if the AI decides it needs that, it'll just seamlessly install it from the Vercel marketplace. And the Vercel marketplace has already curated some of the best infrastructure products in the world to store data, to search data, et cetera. So it's going to make the product even more powerful. I'll say again, I did that exercise, and I do that exercise every day of I have a wild idea and try to see if it can come to life. It's very powerful so far. AIs are still very much a work in progress. They can make mistakes. We have it as a little disclaimer underneath the input. You will find errors, our fitness function. And we've seen such a strong correlation between user love and retention.

(01:15:05):

v0's actually their retentive product compared to other AI products that I've built in the past, or little demos that we've done, or whatever. People subscribe and use it every single day, and are very, if they notice a bug, they're very, very jittery about it because they're depending on it day in and day out. But I'll say errors are still possible. Every once in a while you might get a runtime error or whatever, but a lot of the technology that we've added is so that v0 is very agentic. It has a lot of agency in how to act. So you're going to see very frequently that if it runs into errors, v0 tries to solve them itself.

(01:15:48):

And then last I will say, when products get really big, AI today is just not as good at dealing with massive code bases. But going back to that idea of the React component, because we break down things into files and components, we tend to do quite well in that dimension. In fact, one thing that Next.js was known for is that in order to start a project, you just create a file, and Next.js will route to that page. If anyone is familiar with PHP, it's like how PHP worked. And so it's so good that LLMs are good at working with files now, because it fits very naturally into our world. And if you can scope down when things get really big, if you can give it a smaller task, to work on a specific component or a specific file, you decrease that likelihood of the LLM not being able to reason over very, very, very long context windows.

中文翻译:

是的，你可以通过 Zip 文件导入代码库，Git 集成也即将推出。它可以进行全栈开发，可以连接 API。在接下来的几天里，甚至可能在这个播客发布之前，我们将实现非常紧密的集成，这样如果你需要数据库，或者需要 AI 模型，或者 AI 决定它需要这些，它就会从 Vercel 市场（Marketplace）无缝安装。Vercel 市场已经策划了一些世界上最好的基础设施产品，用于存储数据、搜索数据等。所以这将使产品变得更加强大。我再说一遍，我每天都会做那个练习：我有一个疯狂的想法，看看它是否能实现。到目前为止它非常强大。AI 仍然处于不断完善的过程中，它们会犯错。我们在输入框下方有一个小免责声明。你会发现错误，这是我们的“适应度函数”。我们看到用户喜爱度和留存率之间有很强的相关性。

(01:15:05):

与我过去构建的其他 AI 产品或我们做的小演示相比，v0 实际上是留存率最高的产品。人们订阅并每天使用它，如果他们注意到一个 Bug，他们会非常紧张，因为他们日复一日地依赖它。但我得说，错误仍然是可能的。偶尔你可能会遇到运行时错误之类的，但我们添加的很多技术是为了让 v0 具有很强的“智能体属性”（agentic），它在如何行动方面有很大的自主权。所以你会经常看到，如果遇到错误，v0 会尝试自己解决。

(01:15:48):

最后我要说，当产品变得非常庞大时，现在的 AI 在处理海量代码库方面还不够出色。但回到 React 组件的想法，因为我们将事物分解为文件和组件，我们在那个维度上往往做得很好。事实上，Next.js 闻名的一点是，为了开始一个项目，你只需创建一个文件，Next.js 就会路由到该页面。如果有人熟悉 PHP，这就像 PHP 的工作方式。LLM 现在非常擅长处理文件，这非常自然地契合了我们的世界。当项目变得非常大时，如果你能缩小范围，给它一个较小的任务，让它处理特定的组件或特定的文件，你就能降低 LLM 无法在极长上下文窗口中进行推理的可能性。

[01:16:54] Lenny Rachitsky

English:

I want to go back to design. We talked about how v0 is really good at just great design by default. To lean into that more, if someone wants to improve the design of their product, most people are not designers, they don't really know how to make it look good. They don't know what to ask for. Any just tips and best practices for making their app even better, look even nicer?

中文翻译:

我想回到设计的话题。我们谈到 v0 默认就非常擅长出色的设计。为了进一步探讨这一点，如果有人想改进他们产品的设计，但大多数人不是设计师，他们真的不知道如何让它看起来漂亮，不知道该要求什么。对于让他们的应用变得更好、看起来更美观，有什么建议和最佳实践吗?

[01:17:15] Guillermo Rauch

English:

Yeah, it was really interesting. The other day I met with a CIO of a large bank who, on the side does a lot of coding, or tries out new technologies and whatnot. And I showed him v0. And he immediately became a v0 addict. He texts me every day with feedback. He moved two websites of his own from another website builder type provider to v0 and Vercel, deployed them, gave them a domain name, they're live in production. And then he said, "Look, I have this challenge. I have this music festival that I organize with a couple of friends and this is what the designer gave us." And he had this brochure. It looked very much like a print style design. And so he gave that to v0 and the first result, he was dinging me for it. He's like, "Look, this doesn't look good."

(01:18:09):

And then, because I have experience with the tool, I said, "Why don't I just give it the feedback?" Literally you were asking me yesterday, earlier, some of the things that I've learned with the product or the best practice, what would I recommend if it were sitting next to someone? Not only, you should not hesitate to give the AI feedback, it's so interesting, dude. Sometimes people will press a feedback button to tell us what they wanted v0 to do, and literally all we had to do, in many cases is just, "Can you just tell v0 that?" And so he sent me this message saying, "Yeah, I just don't like the design." And I gave him back a prompt that I would've given. I don't know what I said specifically, but it's like, "Make it more jazzy, make it more, make it pop."

(01:18:56):

And so trying, and again, it goes back to try to draw inspiration from variety that the AI already knows about. So in a couple of prompts, we ended up something that was in his mind, better than the original print design of that brochure, that concert lineup. And at that time, and again, I'm even learning about what v0 is capable of and the best ways to use it. But with design, I think unleashing its creativity, and seeing things, and playing with it is definitely super helpful.

中文翻译:

是的，这非常有趣。前几天我遇到了一家大银行的 CIO，他在业余时间做很多编程工作，或者尝试新技术之类的。我向他展示了 v0，他立刻成了 v0 的拥趸。他每天都给我发短信反馈。他把自己原来的两个网站从另一个网站构建平台迁移到了 v0 和 Vercel，部署了它们，绑定了域名，现在已经在生产环境中运行了。然后他说：“看，我有个挑战。我和几个朋友组织了一个音乐节，这是设计师给我们的东西。” 他有一份手册，看起来非常像印刷风格的设计。他把它交给了 v0，第一个结果出来后，他向我抱怨：“看，这看起来并不好。”

(01:18:09):

然后，因为我有使用这个工具的经验，我说：“为什么不直接给它反馈呢？” 就像你刚才问我的，如果我坐在某人旁边会推荐什么最佳实践？你不仅不应该犹豫给 AI 反馈，而且这真的很有趣。有时人们会按反馈按钮告诉我们他们想让 v0 做什么，而在很多情况下，我们唯一需要做的就是说：“你能直接告诉 v0 吗？” 于是他给我发消息说：“是的，我就是不喜欢这个设计。” 我回给了他一个我会用的提示词。我不记得具体说了什么，但大概是：“让它更活泼一点（jazzy），让它更出彩（make it pop）。”

(01:18:56):

所以尝试——再次回到这一点——尝试从 AI 已经知道的多样性中汲取灵感。在几个提示词之后，我们最终得到了一个在他看来比手册原始印刷设计更好的方案。在那段时间里，我也在学习 v0 的能力和最佳使用方法。但在设计方面，我认为释放它的创造力、去观察、去尝试，绝对是非常有帮助的。

[01:19:35] Lenny Rachitsky

English:

So one thing I'm hearing here is just tell it, "Make this look better." Or, "I don't like- "

中文翻译:

所以我听到的一点是，直接告诉它：“让这个看起来更好。” 或者“我不喜欢——”

[01:19:40] Guillermo Rauch

English:

"Make it pop."

中文翻译:

“让它更出彩。”

[01:19:41] Lenny Rachitsky

English:

"Make it pop."

中文翻译:

“让它更出彩。”

[01:19:42] Guillermo Rauch

English:

You can, totally. And if you can use tokens that are relevant, so, "Neobrutalist, minimalist, newspaper-like, vintage, make it look like a telegram." You can try and reach for things that maybe would not naturally come to mind and you'll be surprised about how well it can transfer those ideas into reality.

中文翻译:

完全可以。如果你能使用相关的词汇（tokens），比如：“新丑风（Neobrutalist）、极简主义、报纸风格、复古、让它看起来像电报。” 你可以尝试寻找那些可能不会自然浮现在脑海中的东西，你会惊讶于它能如此出色地将这些想法转化为现实。

[01:20:09] Lenny Rachitsky

English:

Incredible. Too easy. Maybe to close out our conversation, we'll see where this topic goes. I had this tweet that I loved, that I super resonate with, "The secrets of product quality is blood, sweat, and tears." I completely agree. I think that's why I think my newsletter's been successful. I spend so much time on every newsletter post, more than I think anyone spends on a newsletter post, like 10, 20, 30 hours. And that's why I think it works. Is there anything more behind that tweet, anything you've learned in just the importance of working hard, I guess to great, great stuff?

中文翻译:

不可思议，太简单了。也许作为我们谈话的结尾，看看这个话题会引向何方。我发过一条我很喜欢的推文，我也非常有共鸣：“产品质量的秘密是鲜血、汗水和泪水。” 我完全同意。我认为这就是为什么我的 newsletter 能成功的原因。我在每一篇帖子上面花的时间比任何人都多，大约 10、20、30 个小时。这就是我认为它奏效的原因。那条推文背后还有什么深意吗？关于努力工作对于创造伟大作品的重要性，你学到了什么？

[01:20:42] Guillermo Rauch

English:

Yeah, I mentioned exposure hours is a good example of like, "Look, it can be painful. It can be painful to see your baby break in front of everyone and noticing all the ... " The other thing is that a great product is made up of a thousand little details and so you're never really done. There's a humility that comes from the process also of why the best product builders will say nine nos for every yes. Because when you say

yes, it's like adopting a puppy. A feature is like adopting a puppy. It grows into a beast that you have to take care of, and it's very demanding and loving. But also it's a lot, and poops everywhere. So you have to have a creative restraint. And while you also have to have a give, you have to withhold, sometimes with the respect of the real world complexity that emerges.

(01:21:42):

A little thing that I kind of obsess about. I'll give kudos to the Midjourney team. I really love how Midjourney works on mobile web. I don't know if they have an app yet, like a native app, but their mobile website is phenomenal. And to get it to be that good, by the way, it's possible. It's actually possible to make great things on mobile web. But it needs that sense of love, and restraint, and obsession, and testing a lot, and using your own products a lot. Dogfooding is a great mechanism, obviously. So we use the heck out of Vercel and v0 to make Vercel and v0, and hopefully that helps us do better. But there is a lot of blood, sweat, and tears in the process.

中文翻译:

是的，我提到的暴露时长就是一个很好的例子：“看，这可能是痛苦的。看着你的‘孩子’在所有人面前出丑，注意到所有的瑕疵，这很痛苦。” 另一件事是，一个伟大的产品是由一千个小细节组成的，所以你永远没有真正完成的时候。这个过程也会带来一种谦卑感，这也是为什么最好的产品构建者每说一个“是”之前会说九个“不”。因为当你答应一个功能时，就像领养了一只小狗。一个功能就像领养小狗，它会成长为一个你必须照顾的庞然大物，它既需要爱，又非常磨人。而且它很麻烦，还会到处乱拉屎。所以你必须有创造性的克制。在给予的同时，你必须有所保留，有时这是出于对现实世界复杂性的尊重。

(01:21:42):

我有点痴迷的一件小事：我要向 Midjourney 团队致敬。我非常喜欢 Midjourney 在移动端网页上的运作方式。我不知道他们是否有原生 App，但他们的移动端网站做得太棒了。要做到那么好——顺便说一下，这是可能的，在移动端网页上做出伟大的东西是完全可能的。但这需要那种爱、克制、痴迷、大量的测试，以及大量使用自己的产品。显然，“吃自家狗粮”（Dogfooding）是一个伟大的机制。所以我们拼命使用 Vercel 和 v0 来构建 Vercel 和 v0，希望这能帮助我们做得更好。但这个过程中确实充满了鲜血、汗水和泪水。

[01:22:30] Lenny Rachitsky

English:

Yeah. You can tell how much you use the product. It comes through in everything you say. Let me actually ask about this. You talked about how you said you have 600 engineers? No, 600 people, total and a hundred- 150. How is AI changing the way they work? Is there anything there? Because I feel like you guys are the cutting edge of how products are built. What's happening? Is it just everyone's on Cursor and v0 to build stuff?

中文翻译:

是的。能听出来你有多频繁地使用这个产品，这体现在你说的每一句话中。让我问问这个：你刚才说你有 600 名工程师？不，是总共 600 人，其中 150 名工程师。AI 是如何改变他们的工作方式的？有什么可以分享的吗？因为我觉得你们处于产品构建的最前沿。发生了什么？是每个人都在用 Cursor 和 v0 来构建东西吗？

[01:22:55] Guillermo Rauch

English:

Yeah. Yes, but actually it's more profound. I think it's the, everybody can ship, it's the, we build with AI principles in mind. I actually give a shout-out to the Lumalabs engineer who said, "Well, I'll use AI for

everything. I'll use AI also to generate the images for the website." And I'm seeing, for example, our designers that are working on our next conference generate all of the animations with video models. I'm looking at, our marketing team are creating demos of how the infrastructure works with v0 that are better than any static diagram or landing page that I've ever seen. One of my most viral xeets or X posts is something that one of our designers created, which explains how our compute infrastructure works with an interactive demo. And until he created that, by the way he designed, it and created, and we shipped it all in the tool, first of all, it wasn't part of his day-to-day job to do that.

(01:24:06):

v0 is making you such a powerful generalist that you can step out of your comfort zone of like, "Well, my job was to do only this." You can just create. We have a ritual every Friday, we had it this morning, called Demo Fridays. And so it's very important to create the space for people to step out of that comfort zone and use AI. So us giving permission to people to build and ship things is part of that cultural backdrop that makes these things possible.

(01:24:42):

We had a demo today as part of the Demo Friday of our VP of sales engineering also creating an amazing tool that he's going to use to help prospects understand Vercel with v0. So I've heard from DevOps and infrastructure engineers how much they use tools like Cursor to work on the low levels of the Vercel infrastructure. So I think very quickly we're seeing AI being embedded everywhere. I just heard a product request from a customer that was saying, "Okay, Vercel, you sell domain names. Let me come up with new domain ideas with AI." So I just see a future where AI becomes synonymous with software. I do look forward to it because we need to stop talking about AI at some point. I foresee, it's probably not going to happen, but it is useful to remind people that AI equals software now, and we are a software company. We build software, and we use software to build software.

中文翻译:

是的。是的，但实际上影响更深远。我认为是“人人皆可交付”，是我们带着 AI 原则去构建。我实际上要向那位 Lumalabs 工程师致敬，他说：“好吧，我会用 AI 做一切，我也会用 AI 生成网站的图像。” 我看到，例如我们负责下一次会议的设计师正在用视频模型生成所有的动画。我看到我们的营销团队正在用 v0 创建基础设施如何运作的演示，这些演示比我见过的任何静态图表或落地页都要好。我转发量最高的推文之一就是我们的一位设计师创作的，他用一个交互式演示解释了我们的计算基础设施是如何运作的。在他创作那个之前——顺便说一下，他设计、创作并交付全是在工具里完成的——首先，做那件事并不是他日常工作的一部分。

(01:24:06):

v0 让你成为一个如此强大的全能选手，以至于你可以走出“我的工作只是做这个”的舒适区。你可以直接去创造。我们每周五都有一个仪式，今天早上刚举行过，叫“演示周五”（Demo Fridays）。为人们创造走出舒适区并使用 AI 的空间非常重要。因此，我们允许人们去构建和交付东西，这是让这些事情成为可能的文化背景的一部分。

(01:24:42):

在今天的“演示周五”中，我们的销售工程副总裁也展示了一个惊人的工具，他将使用这个工具通过 v0 帮助潜在客户理解 Vercel。我也从 DevOps 和基础设施工程师那里听说，他们非常频繁地使用 Cursor 之类的工具来处理 Vercel 基础设施的底层工作。所以我认为很快我们就会看到 AI 被嵌入到每一个角落。我刚听到一个客户的产品需求，他说：“好吧，Vercel，你们卖域名，让我用 AI 来想出新的域名创意吧。” 所以我预见 AI 将成为软件的代名词。我非常期待那一天的到来，因为在某个时刻我们需要停止谈论 AI。我预见——虽然可能不会很快发生——但提醒人们“AI 现在等于软件”是有用的。我们是一家软件公司，我们构建软件，并使用软件来构建软件。

[01:25:41] Lenny Rachitsky

English:

And AI is just a part of that.

中文翻译:

而 AI 只是其中的一部分。

[01:25:42] Guillermo Rauch

English:

Yeah.

中文翻译:

是的。

[01:25:43] Lenny Rachitsky

English:

Guillermo, what a beautiful way to end it. Is there anything else you wanted to mention? Anything else that you want to leave listeners with before I let you go?

中文翻译:

Guillermo，这是一个非常完美的结尾。在结束之前，还有什么你想提到的吗？有什么想留给听众的话吗？

[01:25:53] Guillermo Rauch

English:

I'll leave you with my vision of the future, which is we have this billboard in San Francisco, which is, "Everybody Can Cook." It is also part of the Ratatouille film, one of my favorite movies. I look forward to a future where everybody can get their ideas out there. If you can dream it, you can ship it. And also that when you use products and when you see the creations of other people and the things that they put out into the world, that we are collectively making the world better. So anything you experience hopefully gets faster, higher quality, fewer bugs as we go along. And I think we're all contributing to that. And I look forward to that and I look forward to everyone's feedback on how Vercel can play a part in that future.

中文翻译:

我想留给你们我对未来的愿景。我们在旧金山有一个广告牌，上面写着“人人皆可烹饪”（Everybody Can Cook）。这也是电影《料理鼠王》里的一句话，那是我最喜欢的电影之一。我期待一个每个人都能把自己的想法付诸实践的未来。如果你能梦想到，你就能交付它。而且，当你使用产品、看到别人的创作以及他们带给世界的东西时，我们正在共同让世界变得更好。希望随着我们的进步，你体验到的任何东西都能变得更快、质量更高、Bug 更少。我认为我们都在为此做出贡献。我期待那一天的到来，也期待大家对 Vercel 如何在那个未来中发挥作用提供反馈。

[01:26:45] Lenny Rachitsky

English:

So to build on that, where can folks find you online? Should they just go to vercel.com, visit v0.dev?

中文翻译:

顺着这个话题，大家可以在哪里找到你？是直接去 vercel.com，还是访问 v0.dev？

[01:26:53] Guillermo Rauch

English:

Yeah. And go to v0.dev to get started. I did mention if you want to build your own v0, this is more advanced, but check out our templates on vercel.com/templates. And also I'm @rauchg on X, so you can DM me or tweet at me at any time.

中文翻译:

是的，去 v0.dev 开始尝试。我提到过，如果你想构建自己的 v0（这是更高级的操作），请查看 vercel.com/templates 上的模板。另外，我的 X 账号是 @rauchg，你可以随时私信我或艾特我。

[01:27:11] Lenny Rachitsky

English:

Amazing. Guillermo, thank you so much for being here.

中文翻译:

太棒了。Guillermo，非常感谢你能来。

[01:27:13] Guillermo Rauch

English:

Thank you, Lenny. It was so fun.

中文翻译:

谢谢你，Lenny。非常有意思。

[01:27:15] Lenny Rachitsky

English:

Bye, everyone.

中文翻译:

大家再见。

[01:27:18] MUSIC

(instrumental music)

（器乐乐曲）

[01:27:19] Lenny Rachitsky

English:

Thank you so much for listening. If you found this valuable, you can subscribe to the show on Apple Podcasts, Spotify, or your favorite podcast app. Also, please consider giving us a rating or leaving a review as that really helps other listeners find the podcast. You can find all past episodes or learn more about the show at lennyspodcasts.com. See you in the next episode.

中文翻译:

非常感谢您的收听。如果您觉得本期节目有价值，可以在 Apple Podcasts、Spotify 或您喜欢的播客应用中订阅本节目。此外，请考虑给我们评分或留下评论，这能极大地帮助其他听众发现这个播客。您可以在 lennyspodcasts.com 找到所有往期节目或了解更多信息。下期节目再见。