SACRA

EXPERT INTERVIEW

**UPDATED**
06/02/2022

# Jeremy Zhang, CEO of Finch, on building a universal API for employment systems


**TEAM**

Jan-Erik Asplund
Co-Founder
jan@sacra.com

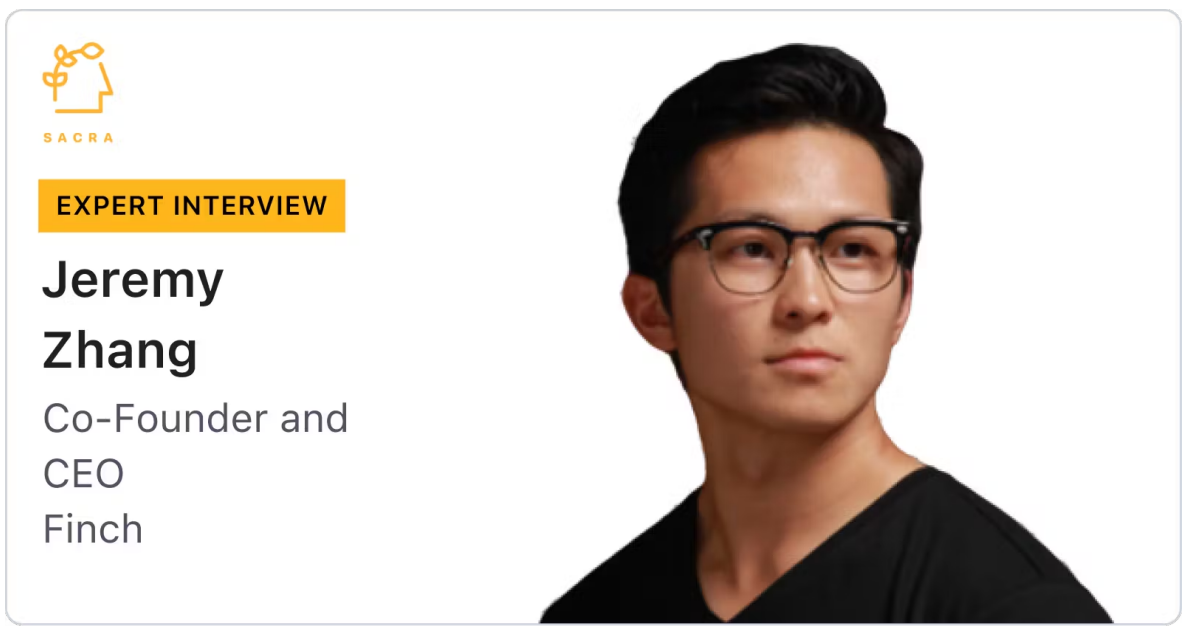

**DISCLAIMERS**

SACRA

Published on **Jun 02nd, 2022**

# Jeremy Zhang, CEO of Finch, on building a universal API for employment systems

By **Jan-Erik Asplund**




## Background

Jeremy Zhang is the CEO and co-founder of Finch. We talked to Jeremy because Finch sits at the middle of two trends: 1) the unbundling of migration of payroll and benefits into vertical SaaS platforms, and 2) the rise of "universal APIs" like Rutter and Plaid.

## Interview

**Can you talk a little about Finch and the core problem you're solving?**

At Finch, we’re building a universal API for employment systems, which we define as any system with access to employee records. The problem we're solving there is really twofold.

One is fragmentation. If you look at the types of systems within the employment ecosystem, there's HR, payroll, admin, recruiting and staffing, time and attendance, and more.

Finch is currently focused on HR and payroll, but our goal is to expand this data model across the other employment systems within the sector.

Now, if you look at payroll or HR alone, there's around 6,000 different systems solely in the U.S. The top 10 of those cover around 55% of the market.

To draw a quick comparison to Plaid, there's around 11,000 banks in the U.S. and the top 10 cover 75% of the market.

To get to that 75% mark in our market, an application would have to build 40 or 50 different connectors, so it’s actually way more fragmented than banking.

The second problem is connectivity.

This whole industry basically operates via files. You’re sending them via SFTP or you’re downloading CSVs and sending them via email to some benefits provider. That report’s going to have all the information, including Social Security number, income, and employment data. As a developer today, I personally do not want to interact with or set up SFTP servers to do the standardizations.

And while there are some parts of the industry that have moved towards an API-driven model, they tend to have closed APIs that take 12 months of partnerships to be able to get into, and they also take a revenue share on top of that—so it isn’t the standard open API ecosystem that we're seeing in other industries.

**Can you give us an example of how Finch gets used?**

Let’s say you’re an employer and use Human Interest for your 401(k) and Gusto for your payroll. You probably don't know this, but you're actually using Finch.

A company like Human Interest sells to thousands of different employers. These employers are on this whole range of systems, such as Gusto, Paychex, Paylocity, Paycor, Paycom,

Payoffice, and others. A company like Human Interest has to have the underlying connectivity to be able to sell their retirement product, and it's a complex connectivity.
First, they need to be able to read back the employee census information all the way down to the paystub.

They also have to control money movement so that they can move money from your payroll to your 401(k) by adding pre-tax deductions along with the company contributions on every single payroll run.

Human Interest was founded long before Finch, and they built around 40 or 50 different connectors internally, but that only covers around 60% of their current customer base.

As for the other half, they have an internal product operations team of 100+ people manually going in, setting data, adding deductions, pulling CSVs, and standardizing it.

Creating this illusion of automation—when there's a lot of manual work that's going on in the background—is just kind of how the whole industry works.

With Finch, we focus on this level of infrastructure, scalability, and reliability, so Human Interest can focus on their go-to-market, channels, and employer experience.

**Is there a more or less consistent buyer customer profile that you've been seeing for Finch?**

Generally, in the startup to mid-market segment, it's the Chief Product Officer—or whoever at a company is primarily focused on product.

On the enterprise side, there are different teams, from an integration team to a separate team focused on workforce products, that end up engaging with Finch. It just depends on the size of the company.

**Could you share any numbers or milestones around product-market fit?**

We have over 10,000 employers and over a million employees connected through our infrastructure. We also have around 5

million daily API calls to our infrastructure to pull employee data, census data, pay statement information, etc.
Those are the high-level numbers. Then we have 150+ connections to the underlying providers, which covers around 88% of the market.

**Why are HR and payroll such fragmented ecosystems?**

Payroll originally grew out of a pen-and-paper model, which means that it's hyper local and also very specialized into specific industries.

Also, it's a very sticky business model. Therefore, you can verticalize it. Instead of having to take over the whole market to be able to build a big business, you can build payroll and get 100 customers, then sell and cross-sell a lot of other products on it and build a very successful business.

Secondly, there is some form of consolidation at the business level—ADP has bought a lot of these companies—but that doesn't mean that there's consolidation at the systems level. Underneath ADP, you have 17 different types of payroll systems and 17 different connections.

**How did you identify HR and payroll as the right wedge here?**

If you're building a bundled, verticalized service like Gusto, what do you start with?

You start with HR and payroll. Then, later, you layer on benefits, time and attendance, and other things.

HR and payroll, though, is the core of the employment system. It's the most sticky part. Therefore, it's the source of truth for employment records. That source of truth does not exist in a benefits system, an ATS system, or a time attendance system. Data flows from the payroll and HR into these other employment modules.

**What does it mean to Finch to be developer-first?**

It's important to have a developer-focused motion and a great developer experience, especially when you're trying to disrupt an old-school sector. But at the same time, if you look at Plaid,

only 15% of their revenue came from their developer-first, bottoms-up product motion. Most of their revenue came from their sales team.
What was important, though, was the marketing part of the developer experience: things like the clean documentation and the whole self-serve dashboard are very important to get someone—even within the sales-driven motion—to be able to trust the product and sign-up.

Twilio and Stripe have been more successful with the bottoms-up motion given that the number of customers they can address is extremely large. With any SaaS product, if your “n” is large, it makes sense to try to drive a really strong marketing motion rather than a sales motion. If you’re a Plaid or another of these specialized API companies, it's really more important to have a strong sales team.

**We're seeing a lot of fintechs expanding across their domains, from Pry, Ramp and Brex going into FP&A, and others. Can you talk about the macro shift that you see happening here and whether it’s a tailwind for Finch?**

When it comes to fintech, what we saw in the past was a period of unbundling and bundling.

When it first started with all the banks, everything was completely bundled. Data was not being transferred between different entities. Plaid changed that—subsequently, there was a period of rebundling.

Now, an app like Robinhood doesn’t just do investments—they also have checking accounts and crypto. We're going to continue to see those cycles across history and into the future.

When we first looked at the employment sector, everything was completely bundled. If you were starting a mom-and-pop shop, you would probably set up your payroll on ADP, and then ADP would be the source of truth for your employment records. Over time, you’d need health insurance, retirement, and worker's compensation insurance, for which you’d just use the whole product suite of ADP. That’s how they cross-sell, even though that might not be the best product set for you.

During the pandemic, when there were a lot of investments and innovations happening in the employment sector,

companies cropped up to tackle specific segments and put out products that were better for employers to use.
Today, you can use Gusto for payroll, and something else for compensation, and something else for retirement, and so on. That wasn't possible until Finch came along and built that level of infrastructure to allow employers and employees to choose which products they want to use.

**Are there explicit lessons you learned from Plaid?**

It’s interesting since, today, people think Plaid is a definitive success story—an obvious product to be built. Back in the day though, it wasn’t obvious.

People were really concerned about competition from Stripe and the problem of revenue concentration. At a point right before Series B, Venmo was 80% of revenue for Plaid. There were also concerns around data security.

One lesson we took away was that revenue concentration is actually not that bad, because it forces you to push your infrastructure and your support and your company to a more mature level.

Secondly, we’ve realized that it’s crucial to focus on how we look at security and put that upfront in how we think about building our infrastructure.

**How do you handle security? How has that been designed into the APIs?**

Security is very important for us, given that we're touching such sensitive employee and employer information. Before we even started, we had to think about our compliance infrastructure and ensure that we built Finch in a compliant, secure manner.

Right now, we have SOC2, CCPA, GDPR, and we're in the process of getting our HIPAA compliance for our health insurance and health benefits companies. Some of our customers are compliance companies like Secureframe or Vanta. We work alongside them to make sure that our system is also as compliant as possible and, to this end, we use their internal security resources to help us to raise the bar.

The second thing we've done is just add a permissions layer to our infrastructure. We know that these systems have a lot of sensitive information, so we built our infrastructure to the point where, if an application doesn't need an SSN, it will never be able to touch an SSN, and that's just one example.

Third is our initial stance of taking a passthrough approach in terms of making a request. Making a request through Finch means making a request to the end infrastructure, so we don't have to store any sensitive information like Social Security or income, and that minimizes our surface area for attacks.

**Can you help us understand Finch's positioning with respect to companies like Check, Pinwheel, Atomic and Argyle?**

We see it as three layers.

The first layer is employees connecting their accounts to payroll systems or employment systems. That's where companies like Atomic, Pinwheel, and Argyle live. Their main focus is income and employment verification. Plaid is also building within that sector and really targeting the consumers.

Then, in the layer beneath Finch, you have companies like Check, which are building embedded payroll systems to allow vertical SaaS companies to build payroll into their product.

There's Gusto with their embedded payroll product, Zeal, and legacy players like PrismHR. Those are all white-labeled solutions. Some of them are customers of Finch, some are partners, and some are investors.

We want to be integrated well within this lower-level embedded infrastructure layer, and we expect it will create more fragmentation within our sector within the next several years—that's good for us because standardizing that fragmentation actually is one of the large value-adds of Finch.

Lastly, Finch connects employers to the end-systems. We sell to companies like benefits providers, B2B fintech companies, and the overall employment HR tool side. Those are completely different use cases than either of the other layers of the infrastructure.

**Where do you see Finch in five years?**

We believe that we will be able to get to $100 million ARR with our current model.

First, there's horizontal expansion in terms of data models—getting into the benefits sector, into staffing and recruiting, time and attendance, and so on.

Second, there's expanding vertically down into more of the rails, doing things like writing contributions and deductions—which we are already doing within the payroll module—processing reimbursements, and potentially more interesting programmatic functionalities.

In the future, we have the opportunity to  expand into the application layer. For example, once there's thousands of applications built on top of Finch, we'll look very similar to a benefits broker. We can actually be a brand to the end employers and become a distribution channel.

These are all things that we can do once there's thousands of applications and benefits providers built on top of Finch.

It's a similar expansion model to most API companies. Start off with a very initial wedge, and then expand to other data models and customers.

A good example is Plaid. Plaid just launched a recurring transaction product, which is built on top of their normal transaction product but with just an added layer that tells you if a transaction is recurring.

Twilio has their MFA functionality, which is just built on top of their normal SMS functionality. They're able to upsell at a different unit price.

These are all just the normal playbooks of API companies.

## Disclaimers