Twitter Thread by Patrick McKenzie



Patrick McKenzie
@patio11

One of the big promises of software is composability. You can build rich, powerful experiences out of basic building blocks.

APIs add new things to the toolbox. For example: Treasury, which lets an app/platform store, move, and track a business'

I've been a small business owner and can talk at length about SMB banking, and will later, but let's put on the software developer hat right now.

Lots of software talks about money, keeps records about money, does calculations about money, but can't *touch* money.

This is extremely frustrating when you're building SaaS apps for businesses, because you have total control over your UX right until your app needs to touch money... at which point all data about it lives in a silo you can't access.

So you generally push work to the operator.

For example, suppose you're writing a business-in-a-box system for electricians, including an invoicing feature.

You need to be able to read bank transactions to reconcile. You probably can't. The owner can. So you ask the owner to do mind-numbing work a computer does better.

It sure would be great if your business customers had bank accounts you could actually introspect and operate on their behalves! You could just get the list of incoming payments and match against the invoices.

There is some software to write but it is not rocket science.

Stripe Billing goes to some crazy lengths to do this (virtual bank accounts, bespoke contracts with banking partners, etc); most SaaS companies offering invoicing as a feature probably shouldn't. You would want to spend your developer time elsewhere.

Stripe Treasury, now in limited beta, will allow software companies to provision money management accounts for their users, and then operate on those accounts like they expect to operate on data, servers, Stripe charges, or anything else you

have an API for.

What’s the difference between a money management account and e.g. a standard banking checking account? As a user, not that much. You get a card to buy things with and get cash, and the ability to send ACH payments, bill payments, and the like.

The distinction is mostly implementation-layer concerns in banking-land.

It's difficult to envision what you can create given powerful primitives. Almost everything you see on a screen sits on top of NAND gates; looking at a NAND gate doesn't show you Twitter.

But let me speculate a bit:

B2B SaaS which deals with money, which is most of it, just got a whole lot more interesting, because it can directly operate on the money.

Take Shopify, for example. Shopify gives users an online storefront, and naturally needs to keep track of the total in the virtual register.

But with Stripe Treasury API, Shopify launched Shopify Balance, which lets them build a much better register. They always know the real total inside of it, not just what "should" be there.

They can automate its operations.

https://t.co/fMOqHh2Tlv

And because of Stripe's global payments and treasury network, we didn't just make these capabilities *accessible* to developers.

We made it *better* for your users than many business bank accounts.

What does every small business care about? Cash flow. What's a top 10 customer support request for every business which sees funds flows, like Shopify?

"When do I get my money?"

Can you imagine needing to email Google to ask when that email that you know was sent will arrive?

If the money just arrived faster, there would be less angst for users whether it would arrive at all. Less tickets opened. Less questions you can’t usefully answer.

Why is money *so slow* and *so opaque*?

sigh That is a complicated topic, but at the end of the day, you're generally rate limited by the slowest rail between where the money comes in and where it goes out.

In the U.S., you'll often be blocked on the ACH network. Fast compared to stagecoach; slow compared to email.

But with Stripe Treasury, money doesn't have to lag its way through the ACH network between a customer paying it and a business receiving it.

Stripe has an arrangement with the banks that ultimately hold the business users' funds.

So when your software says "OK, disburse the money they've earned from the platform", it's in their account (and spendable!) on the same day.

(We are working on making that even faster by default. It's not called HTTP 200 Check Back In An Hour.)

SaaS platforms are one use case for Stripe Treasury, but you can envision another one: novel fintechs, with different UXes, brands, and target business users, built on a core set of primitives.

Putting together fintech products is historically a pain in the keister.

Every significant feature you add requires an integration with a banking partner, often a *new* banking partner, often *several* banking partners.

That integration requires lengthy negotiations, bespoke legal work, approvals, etc just to unblock your engineering team.

Then your engineering team receives the spec, and the *real* fun begins.

We did away with most of that for Stripe Treasury, just like we did for Payments before it.

No negotiation required. No bespoke legal work. More of the necessary levels of complexity in touching money businesses depend on; less of the overhead.

And, of course, the care and attention to the developer experience that you'd expect from us.

A day may come when Stripe forgets what is was like to build software, when we ship a 400 page PDF file as the sole documentation for an API, but it is not this day.

So if you want to build a vertical fintech targeting carpenters? Offering payments in and out, a place to hold insured funds, and a card to spend them with?

You can build all of that on Stripe APIs now.

And if your users need financing? Snap in Stripe Capital for Platforms and we can facilitate access to loans based on their predicted future revenue through your platform. https://t.co/Ye28gyMPRh (This also came out this week.)

And now, back to being an erstwhile small businessman who has had bank accounts for many years.

AAAAAAAARGH.

Small business banking has a poor UX, because of structural issues.

It is managed at most banks as an offshoot of personal banking, because the userbase is basically the same people who show up at the branch.

But the needs are quite different.

Banks specialize in one-size-fits-most products. The basic checking account you have is designed to cover more than 60% of the total population. It functions effectively the same for almost all of its users.

This kinda makes sense for individuals. Financial circumstances differ, but tend to rhyme: money in from a job or pension; money out to housing, utilities, debit card payments; some cash management.

(Narrator: This is not, in fact, an adequate spec for a checking account.)

But the needs of businesses are _wildly different_ from each other!

A small scale landlord cares quite a bit about mortgages and timely rent collection (and super-timely notice of non-payment), but rather little about payroll or expenses for consumables, and probably almost nothing for cash management.

A pizzeria, on the other hand, thinks it is inconceivable to make almost all the money on one day a month, spends most of revenue on payroll and consumables, and cares quite a bit about cash management.

These businesses deserve fundamentally different application-layer experiences from their banking. And they don't get them, because banks are incapable of offering hundreds of vertical software solutions.

(Just making the one-size-fits-most personal banking and mobile apps *actually good* is hard enough! It seems like we're just getting there in the last 5 years!)

Who does offer hundreds of vertically-specialized application experiences? The software industry. Shopify specializes in e-commerce merchants and understands them, thoroughly. They don't sell to dentists.

Dentists have thriving ecosystems of firms that live and breathe every aspect of their practices.

So do graveyards. And hotels. And landlords. And spas. And tutors. And yoga teachers. And...

Why is software so varied but banking so… not? One reason is that businesses pay an awful lot for software, but pay relatively little for banking. Banking is a scale game.

Not enough dentists pay not enough dollars for banks to put software teams against dental practice UX.

But because a software company can address dentists by the nation, not by the square-mile-around-a-branch, and because they charge more, a software company serving dentists can reliably generate enough money to pay an engineering team.

Here Stripe acts as an aggregator: we can gather up *many* businesses attached to *many* software companies. These product teams understand their customers' businesses *thoroughly*.

We can then take that package to leading banks. That reach is *very interesting* to them.

That is why Goldman Sachs, Citi, Barclays, and Evolve Bank and Trust partner with us.

A pizzeria can't walk into Goldman Sachs and walk out with a bank account.

A software company serving pizzerias could if they had, uh, a lot of dough.

But a material chunk of the software industry, that's a different story altogether.

And so our partner banks have made great products available, at pricing and terms that small businesses just don't usually get in direct banking relationships.

Take account maintenance fees. Please.

I know exactly how many times I paid the $14 account maintenance fee for my software businesses, ten years later. That's how much I hated them.

I had gone below $X,000 in deposits (I was bootstrapped! Rent was due!) and hadn't managed to spend at least $250 on the debit card because, whoops, mis-timed a billing cycle by two days.

(Being a bit older and wiser I probably should have just considered that $14 a cost-of-doing-business and spent my time on sales, not timing debit card transactions, but my emotional state at the time was closer to "Gah I feel like a chump.")

We have agreed, with our partners, on a pricing structure where a SaaS company or fintech doesn't need to charge an account maintenance fee.

(As the software company, you *could*, because you control your pricing, but presumably you will make better choices, and your users will not say, ten years later, "I trusted you, and you made me feel like a chump. Six times.")

How is this possible, and why didn't the banking system do it years ago?

One way to think of it is that banks have huge expenses to attract SMB deposits, including marketing campaigns and branch networks, and those drive the pricing of SMB banking.

The cost of *servicing* SMB accounts has gone down over the years, now that account statements are not calculated by highly-trained artisans, but the cost of *acquiring* them is increasing over time.

Nationwide advertising, branches, and sales reps aren’t as cheap as cron jobs.

Since the software industry is doing their own customer acquisition, and there is sharply less need for the branch network (itself more a sales tool than a servicing tool), some banks are happy to pass the savings back to Main Street.

There's obviously a lot more detail here (hosted onboarding flows, KYC and AML compliance workflows built for you, all accounts have deposit insurance, etc), but I have to end this tweetstorm somewhere.

I'm very excited to see what software people do with the Stripe Treasury.

Anticipating a FAQ: "Is this the product you were referring to here?"

No. This was my number two. You're welcome to your guess at the number one. (Though, who knows, I heard of a new project last week and might steal the zeroth spot for it.)

https://t.co/osjjx8gNM0

> Part of the needle threading is making sure that one is still developing some things which are uniquely exciting to developers and smaller shops.
>
> Which: I can\u2019t spoil it, but one thing in the pipeline is maybe my favorite Stripe product since Stripe Atlas if we do it right.
>
> — Patrick McKenzie (@patio11) September 23, 2020