SACRA

EXPERT INTERVIEW

UPDATED
02/16/2024

Samiur Rahman, CEO of Heyday, on building a production-grade AI stack

TEAM

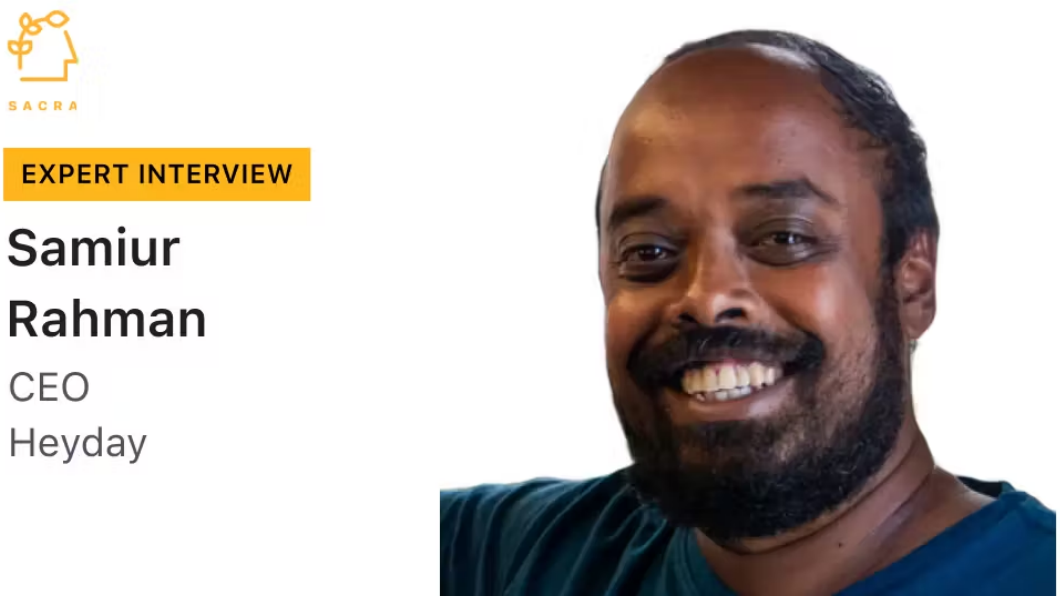
Jan-Erik Asplund
Co-Founder
jan@sacra.com

DISCLAIMERS

Published on **Feb 16th, 2024**

Samiur Rahman, CEO of Heyday, on building a production-grade AI stack

By **Jan-Erik Asplund**



Background

Samiur Rahman is the co-founder and CEO of Heyday. We talked to Samiur about training and deploying AI models using CoreWeave, the differences between GPU cluster companies like Lambda Labs and CoreWeave, and the workflow for a company building generative AI features today.

Interview

You've been building web products powered by machine learning for 10+ years. Can you give a history of what that has meant leading up to the present?

Yeah, I'll give the brief spiel. So I went to college for electrical engineering, signal processing and machine learning. I didn't even really know about machine learning until my junior year of college. I was super fascinated by it. I've always been a bimodal person where I can be lazy and procrastinate or be

super driven and ambitious, and it depends on the thing that I need to do. If it's a mundane, boring task that isn't fun or challenging, then it'll just sit forever.
I had that brief glimmer of like, "Whoa, this could be the way that we can automate so much of the annoying bullshit." And that was a long time ago. I was building AI for games like chess and stuff like that, and that was my first foray into it.

I started my career off at Amazon. I was on the retail electronic side, so the pages for TVs, cameras, et cetera. And one of the first things that I did machine learning wise was try to convince our team all the way up to our VP that we shouldn't be using a team of humans to decide which accessories to recommend for each of our products.

At the time, Amazon had negative margins on things like the main products like TVs and cameras and would have had huge margins on accessories. A lot of the bottom line was based on selling accessories. So HDMI cables with TVs, camera bags with cameras or whatever. And so we had humans who would pick those things out, and I was like, that's crazy. We should be using machine learning. We have a lot of data to do this.

And at the time, basically no one understood what I was talking about, and so I had to do three months of convincing that we could do this. Let's at least do an experiment. And immediately, once we launched that, not only did we have people not having to do this, but also quarter over quarter revenue went up 70%.

Turns out there are things that algorithms are better than humans are doing. Shocker. After that, I was doing embedded machine learning with a company working with Nike to do algorithms to predict step count from sensors. Then I worked at Mattermark where I was the head of machine learning, and a lot of what we did was extract structured information from the unstructured news or websites. As we got bigger and bigger, we had millions of companies in our database and our users wanted to be able to search for things like gig economy companies or dev tools companies, but the problem is that something like a GitHub wouldn't call themselves a dev tools company in their description, so to solve this problem, we basically made our document embeddings and created our own vector database to do search long before Pinecone and all these people.

We solved search in a really awesome way and I thought that that could be generally applicable, but instead of trying to build a dev tool, I was more excited about what that could enable that didn't exist before in terms of end-user experiences.

We started a company called Journal, which we thought of as Google search for your own stuff. It would integrate with your Google Docs, Gmail, Slack, Dropbox, all these places where you have stuff all over the place, so you can search for everything in one place.

It was popular, but we didn't end up making a business model that actually worked in making revenue. so we shut that down and we started Heyday.

We were very far ahead AI-wise when we started the company, but then within a year of that, OpenAI came out with APIs that anyone could use, and suddenly that introduced all kinds of people who could do—at least at the 80th percentile from the baseline—what we were able to do with Heyday. So that's been the evolution I've seen, from no one knowing about machine learning to literally everyone trying to do something with it.

Can you share some background about Heyday, what you're building and what are the key AI-powered features?

Yeah, so the mission of Heyday is to be kind of an AI co-pilot for knowledge workers who do research heavy work—honestly, folks like you and folks like me.

We started with a pretty general product—think like Rewind, and I'm biased, but better. We had a learning which is that when you build super general products, either you somehow figure it out and you are ChatGPT, or you're just a magic toy. That was a big learning for us.

What we wanted to do was focus on one persona at a time. We ended up focusing on coaches and ancillary consultants, folks who have long-term relationships with 10 to 20 clients that they are helping in a knowledge oriented way.

We basically relaunched the product with coaches in October. It's been amazing to see how when you focus in on the problems of a single use case, how much easier and how

much faster you can iterate the product and how much better the product gets. And commensurately, coaches have been loving the product.
We have a 45% trial to paid user conversion rate with a $40 a month product, so it's been pretty gratifying in that way.

For us, what we're trying to be is a unified platform on top of which we can learn the workflow needs and the specific fine tuning AI needs of different personas, and then easily iterate to more and more personas and build. We're basically building atoms along the way. Every N persona should be n plus one persona should be easier for us to, we've got more and more atoms and put together the molecule that's perfect for them.

Can you talk a bit about how coaches emerged as a good vertical to go after first? What is it about their needs that makes it a good fit?

It's a good vertical because they have the problem, and a lot of them are well enough off that basically saving them 10 hours a month or accelerating their workflow is worth way more than $40 a month. It becomes an easy decision in terms of paying for a product. And then they're also not part of a large IT org, so they're really good for initial startup sales cycles. We need to iterate, learn about the product, iterate, iterate, iterate as much as possible In that way. They were great in terms of total market size. We went into that knowing that it's a small market because our hypothesis was we're going to move over to a second market really quickly.

Could we have done more work to find a market that was bigger that also had the same properties of coaches? Maybe that's a thing that we've internally talked about. I don't know, not worth dwelling on too much, still working fairly well, and well ideally we'll figure out the next market really quickly in terms of why we could solve their problems.

The superpower coaches have is basically to have insightful conversations with their clients, and then they have a lot of other work around that which isn't part of that superpower, but they have to do it to maintain their business and provide value to their clients. You can imagine a coach who's amazing at having conversations and getting insights out, but a lot of clients need to have things actionable and have accountability.

Maybe that's not the thing that the coach likes doing, or they're a great coach at having a conversation, but then they have 15 to 20 clients, so their memory isn't great. And so the next session they're coming in, there's a 15 minute reintroduction of, Hey, what are the challenges that client is facing?

We thought we could be really helpful in understanding everything about the client for a coach and then getting them to 80th-90th percentile prepped for every session because we can summarize the key takeaways and the action items that were discussed.

You mentioned a few times this potential threat that OpenAI posed, but that it didn't really pan out in that way. What other tools do you have in your AI stack that you're using?

We do use OpenAI, by the way, for certain things. What our stack looks like is we run a lot of our own models.

Before the wave of GPU-enabled clusters, we were still using AWS for some of their GPU clusters. It was annoying. They had the wrong video cards. Their GPUs were optimized for people doing graphics or video modeling and stuff like that, not for machine learning, and so it was really annoying.

We were paying 10x as much as we should, but we're like, well, what else could we do? I guess we are not going to set up our own GPU clusters in the office.

We were super excited when things like CoreWeave and Lambda Labs and stuff like that started showing up where we could run ML specific GPUs in a production capable cloud where we can scale up and down and things like that.

We currently use CoreWeave for all of our ML compute, where we run our own models that can range from small embedding models, our own small embedding models to large LLMs that are fine tuned on top of things like Llama or Mixtral to have specific specializations. So yeah, we're running a whole bunch of ML models on top of GPUs on CoreWeave right now.

Did you evaluate CoreWeave against Lambda Labs?

Lambda Labs is at least 50% cheaper per GPU, so we use it for certain things, but we don't use it for production because CoreWeave specializes as a Kubernetes cluster of GPU based machines. It's very similar to just running our normal production stack in AWS, like the CPU based stack, except we're running it in CoreWeave. So it was super easy. We were Docker heavy, we run on things like Kubernetes already. So we didn't have to adjust the code that was already running in AWS at all just to start running it. We just had to adjust where APIs were being hit. CoreWeave was production ready. You can expose APIs to the public web or have VPCs with AWS. It has a lot of actual production features that we want, like autoscaling, and CoreWeave just takes care of that which for a startup is worth the extra 50% of costs.

We could start renting our stuff from Lambda Labs and then build that Kubernetes cluster ourselves in our own hosts, but the difference between Lambda Labs and CoreWeave, I would say, is similar to running your own instances on Digital Ocean. Digital Ocean basically lets you rent an instance and do whatever the hell you want with it. But most people aren't running production scale stuff on Digital Ocean just because it's just annoying.

That’s the main reason why we only do things like experimentation on Lambda Labs. When we're doing initial training and things like that where it doesn't have to have production capacity, we'll just run a Jupyter notebook, download our training data from S3 and Amazon, and then start to play around with it in Lambda Labs just because it's cheaper.

What do you use for fine-tuning?

Yeah, again, the tooling has become a lot better than it used to be.

When we first started, we basically had to build our own. There was an algorithm called LoRa, which is basically used by everyone now, but back then it was a paper that a bunch of us knew about, and there was a little library that made it easy to do it, but then you still had to do your own little adjustments, so we were doing LoRa on top of some of our models. Now all that stuff is built into the Hugging Face trainer library. You don't have to do any weird customization other than just parameter

optimization, typical stuff. So it's built into all the standard ways that people do fine tuning.
A lot of people may not know the difference between what fine tuning is and reinforcement learning with human feedback. They're two different stages in some ways.

Fine tuning is typically, “Hey, we're going to give you more representative information related to the problem.” Not exactly, “Oh, hey, given an input, do this output.” I mean, that can help. But it's typically like, “Okay, let's say we want our LLM to be really good at understanding transcripts and doing summaries based on that.” We don't even need to have summaries. We could literally just start to fine tune the LLM on transcripts so that it understands more in an unsupervised way about what transcripts look like. We'd usually start with that stage.

The reinforcement learning with human feedback stage is where we'll say, “Okay, here's the transcript. Here's your five takeaways. Is this good or bad?” Or, “Give a rating between one to five.” We could either do it directly with humans, or with users, we can leverage the edits that they're doing.

Either way, we have a corpus of data that's marked good or bad, and then with reinforcement learning, we'll take that and start to adjust the output of the LLM.

How do you think about the customer-facing advantage that CoreWeave confers? Is it something that you measure in terms of responsiveness or more qualitatively?

No, I wouldn't say there's any real difference from speed. If I compare using a Nvidia H100 instance based in CoreWeave or based in Lambda Labs, it's more about speed of development.

Lambda Labs would be really annoying to run production stuff on just because we'd have to do our own auto scaling. We'd have to run our own Kubernetes cluster. And yeah, we could, but that's not the thing that I'd rather spend time on.

At the point where that’s worth it, maybe we're to the point where we just run our own servers. We can run our own racks.

So Lambda Labs feels great for things where we don't have to care about all the production level things—API access or public web access or scalability, automatic scalability based on

resource load, guaranteed servers, whatever, all this stuff. Whereas CoreWeave have built it up basically to where they're like AWS, but they specialize in GPU instances.

Have you been looking at AWS as a potential tool to use for this?

So they've recognized how important this is. And so they do have more of the ML focused Nvidia GPUs. Now you can get H100 instances, but they cost two to three times as much as CoreWeave. That's the main reason we're not using them. We put in the effort to be able to have a dual cluster environment, but unless AWS makes it super cheap or at least brings the cost back down to the same level as CoreWeave, there's no reason for us to switch back.

On the other hand, I have a friend who runs a company called Groq. He used to basically work on Google's TP stuff and left to create Groq. He's been working on this for a long time and they're finally coming up with a Groq Cloud offering over hopefully the next six months.

If that works, that would be a big reason for us to switch from CoreWeave because a lot of the things that I've seen internally about what they can do is can be 5 to 10X faster at the same cost compared to Nvidia GPUs.

Can you talk a bit about what makes CoreWeave a compelling infrastructure product for Heyday?

CoreWeave have been fantastic partners and they've made things stable in a space where most people haven't been able to provide production-grade infrastructure. I don't think they've built much of a moat other than just being way ahead of people, but everyone's catching up. They know that this is important.

It's almost like the thing that made it really easy for us to adopt them will make it really easy for us to leave them—because all they are is just a Kubernetes cluster of GPU capable instances. That just means we can move to any Kubernetes cluster, which is, if we move back to AWS, super easy. It will take us maybe five days. It's just like running Docker images.

If the price on AWS went down, you would be inclined to switch over so you could centralize everything?

Exactly. Right. And just for reliability. CoreWeave's definitely had more outages than AWS. I would actually say that given the size and maturity of the company they are, it's actually impressive how few outages they've had. But it's not AWS.

If AWS goes down, the internet goes down, and so no one can blame you. Everyone knows everything's broken. Once in a while, we'll have a CoreWeave outage and we're like, "Oh, fuck, half our models aren't running. We can't serve search properly, we can't serve our assistant properly, and we have to put up the banner saying an upstream provider is having issues."

We probably would have to do that 1/10th of the time if we were on AWS, and if it was happening on AWS, most likely people are already freaking out about so many things that they would hardly even notice.

Disclaimers