David, hello everybody. My name is David hope, and I'm going to give you a quick overview of the elastic observability platform, including all the latest and greatest things we've recently introduced in the product. When you log into Elastic observability, the first thing you'll see is this overview page. You can see how it's already brought quite a lot of data into view. Here. We've got alerts and log events and hosts, but the question remains, how do you get the data in here in the first place?

Well, that's great because we've recently introduced some really nice new onboarding workflows for bringing your data into Elastic observability. Now, one of the most common places that people like to start with elastic observability is bringing in their Kubernetes data. Now, elastic has recently moved it over to using open telemetry as our standard data collection methodology. So we have the elastic distributions for open telemetry, which uses both the collectors and the agents, the zero instrumentation agents, to bring logs, metrics and traces into Elastic observability in open telemetry, native format.

So I'm going to go ahead and just click on here, and we'll see how easy it is to bring data in. First of all, you just add the helm repo, then you install the open telemetry operator, which will manage your open telemetry collectors and your instrumentation. And then it's that easy to bring all your logs, metrics and traces into Elastic observability. Just three simple steps, and your data will start flowing.

Now to instrument, all you need to do is add annotations, either to your Kubernetes namespace or directly to any deployments that you want to instrument.

So that's how you can very quickly get all your logs, metrics and traces into Elastic

now there are a couple of other ways you can get data in if you're in the cloud. We have these new Quick Starts for the cloud providers, for example, with AWS firehose, which now very simple to create a firehose stream that brings logs and metrics directly into Elastic using a cloud formation template, and once it's done that, you'll automatically get access to all of these dashboards. So we can have a look at, for example, this AWS usage dashboard, and you'll see that we're bringing all this data in from AWS, very easy, very quick to bring that data in.

Now I showed you earlier that we brought the open telemetry data in for Kubernetes that gets logs, metrics and traces in from anything that's deployed on a Kubernetes environment, and that also has an out of the box dashboard that you can see here. This is brand new, and you can see things like Node statuses, CPU usage, memory usage, any problems that might be occurring. You can see stateful sets and replica sets, lots of very useful information, and it's all being brought into Elastic in open telemetry native format. That's with the open telemetry semantic conventions. Now if you're moving away from dashboards that a lot of people are now because you know, or at least reducing and consolidating dashboards, perhaps you don't even want to use dashboards. Well, we have the elastic AI assistant over here, and what that can do is it can very quickly analyze your logs, metrics and traces. So for example, we can ask it what Kubernetes metrics are available that can tell us what is the average memory over the last 15 minutes. And it can do this because it has access to all of that observability data inside elastic, and can very quickly show us that there are actually some pods here with some quite high average memory usage.

Now let's take a look at another dashboard. Let's go back here, and we can see here, if we click on the business health dashboard that I built over here, we actually have brought together all of our logs, metrics and traces data into a single dashboard. We can even see things like revenue over time. We can see nginx log message and error codes coming in. Oh, it looks like we had a plethora of 500 errors hitting the system here, that is, that's not great, so maybe there was some downtime here. But we can also see anomaly detection. In fact, if we go up past the last 24 hours, we can see there have been a number of anomalies over the past 24 hours, and our service level objectives are being. Impacted. Now one of the things that's really interesting about elastic observability is that we have had really great machine learning anomaly detection to help give you that early warning indicator of something that might be happening or something that might be building up in your system.

Now what I like to say is, is that, you know, why do you spend hours in a lot of observability tools? I mean, I've been in this industry for quite a long time, just curating the perfect URL to monitor when we have machine learning available that can honestly monitor 1000s of high cardinality data points. And here you can see we've got machine learning set up on our in Gen X logs, and we're analyzing things like anomalies in status code, anomalies in IP address, anomalies in the amount of visitors that there are. So for example, if we look at the IP address one over here, we can see there are 1000s of different anomaly detection algorithms for all of the different IP addresses that have hit our Engine server here, and this honestly can handle a very high cardinality data, hundreds of 1000s of IP addresses, and it can search for anomalies throughout them.

So elastics machine learning anomaly detection can save you a lot of time and give you a lot of that early warning indication that something's happening without you having to spend hours looking at dashboards.

Now, going back to our business health dashboard, there is one slo that I really want to dig into this cart service slo seems to be hurting, right? I mean, 91% when we actually have a target of 99.9% is very low for

service levels.

So let's take a little look at this particular SLO. Now, the great thing about SLOs is that the way they work is they have, like, an error budget. So what you usually, typically do is you say, I want you know this APM availability for the cart service. I need the cart service to be up 99% of the time over a period of seven days. And so what that means is, is I may be able to tolerate one in 1000 customers having a problem, but I certainly can't tolerate two in 1003 in 1000 and every time that builds up, we burn down what's called error budget. So our error budget becomes exhausted, and that's great, because it means that if there's a an infrequent one off event, I perhaps don't need to alert people. I don't need to wake everybody up at night. But if all of a sudden it's trending in a direction where there's 2345678, and 10 in 1000 in a very, in very quick succession, I can then set an alert on that particular threshold being breached.

So the cart service here looks like it's in a lot of trouble. So what we're going to do is we're going to dig into this and have a little look at what's happening now. We're going to have a look at the last hour in particular, and we can see here that our latency for this particular service is a little variable. And what's happening here is that the blue shading that you can see here is our machine learning algorithm. You can see it's actually comparing it to the week before, and if it's very anomalous compared to the week before, it will actually trigger an anomaly, which you can, of course, alert on.

Now this service also has a number of errors. So we can see here that there's some kind of error cannot access car storage. Now I'm not a Redis expert, so I'm not really sure what this means or what's happening here. Now what I can do is I can use our AI assistant to help us out here. So if I click here at what is this error, it can analyze the error for us and give us its recommendation of what we should do.

So as you can see here, it tells me what the error means, gives probable causes, and because it's actually plugged into our GitHub repository, the AI assistant is able to tell us about things that are happening in real time, things that our team are working on, because it's actually connected to our GitHub repository.

And I'll show you a little bit about how that works. So how do we bring that additional context into the conversation here? Well, first of all, what we're doing is we're using a search connector that you can see over here. We've got some search connectors set up that are actually pulling information in from GitHub into Elastic Search. But it doesn't stop there. When it pulls that data in, it actually goes through a pipeline here and a machine learning pipeline. That vectorizes the data. And by vectorizing the data, that means we can actually search it very, very easily using just semantic search.

And so when you're having conversations with the AI assistant, like I did earlier, you actually can bring that data in to the conversation, because it is able to use the, you know, plain English, natural language, to search for any relevant data that's related to the error or the problem that's happening. And so with that connection to GitHub, the ability to search that data bring back incredibly relevant data. It's not just bringing back any old data, it's bringing bringing back the most relevant data we can actually then bring that into the conversation.

Now there's another thing I just wanted to do here. I want to see if there is a runbook.

So this is also going to search our GitHub repository. And you can see here that there is a runbook in our GitHub repository, and in fact, it was actually recently updated. So you can actually see the link to the original runbook and to the pull request where there's an update happening. So somebody has recently updated this runbook. It seems like it's currently grab a cup of coffee while Luca is working on a fix. So Luca is the person to call. Luca is going to fix the problem for us now, then

the next thing I wanted to show you is

our LLM observability now, LLM observability is all the rage right now. If you're not building a chat bot, then I'm very surprised. Almost everybody now is building some kind of integration into an LLM to either create nginix workflows or improve customer experience by integrating, you know, chat features into your app, into their applications. The first thing I want to show you is that our new LLM integration. So the first integration that we have available here is the bedrock integration.

So if we look here, we've actually got Amazon bedrock. And if we go back past the last 24 hours, we're actually pulling all sorts of information from Amazon bedrock, including the models that are being used, the invocations, the tokens, the errors. We can see the invocation rate by LLM. We can see the token usage. We can also see the logs, so we can see a great deal of information here that can help us manage our costs. We can actually use the information that is coming back, because we can see the prompts and the responses to see if our llms are hallucinating. For example, we can do a great deal with this information to help us diagnose and resolve any LLM issues. And we don't just stop there. We also with the elastic distributions of open telemetry. We also are able to provide tracing. So if you look through here, you can actually see, if you click on here, for example, this post, API chat, which is a part of this LLM application, you can see that we get full tracing, which gives you the ability to see when something is taking too long, which is very common with these applications that have To integrate with llms. And in addition to that, you can see that we bring all of this data in, like, for example, the prompt and the responses for that specific trace. So that gives you a little overview of where we are with elastic observability. I showed you how to bring data in. I showed you how to analyze your data. I showed you powerful machine learning that can help you spot problems before they build up in your system. I also showed you how you can actually use the AI assistant to integrate with your own knowledge and your own knowledge bases. I also showed you a little bit about how you can use SLOs as the de facto way to do health monitoring, which is now the most popular way to do health monitoring. And finally, I showed you that LLM observability that we now have available in the product. I think will you'd all agree that it is definitely the time to start to think about all of this LLM observability stuff, since I'm just sure everybody will be investing in creating these types of applications in the near future. Thank you very much. Applause.