**Atul Deo:** 0:00

Bedrock is kind of the service that people come to to build their application, but it doesn't offer any UI support or whatnot. It's for educational purposes.

**Craig Smith:** 0:11

But the core problem with large language models being used is certainly in any safety critical application, but in any application for which ground truth is important is hallucinations.

**Atul Deo:** 0:24

Now we make our own chips here, chips which gives us a tremendous advantage these times where you know there is shortage of GPUs.

**Craig Smith:** 0:31

Hi, good tech solves problems you know about. Great tech solves problems you haven't even thought about. What can the commerce platform trusted by millions of merchants do for you? It's time for Shopify, the commerce platform, revolutionizing millions of businesses worldwide. Whether you're a garage entrepreneur or IPO ready, shopify is the only tool you need to start, run and grow your business without the struggle. Shopify puts you in control of every sales channel. So whether you're selling satin sheets from Shopify in person point of sale systems or offering organic olive oil on Shopify's all in one e-commerce platform, you're covered. Shopify powers 10% of all e-commerce in the United States and Shopify's truly a global force powering all birds, rothes and Brooklyn and millions of other entrepreneurs of every size across over 170 countries. Plus, shopify's award-winning help is there to support your success every step of the way. Sign up for a $1 per month trial period at Shopify.com slash Ion AI that's Shopify S-H-O-P-I-F-Y dot com. Slash Ion AI that's E-Y-E-O-N-A-I all run together. Go to Shopify.com slash Ion AI to take your business to the next level today. Excuse me, sir, I couldn't help but overhear. Did you say Shopify, oh, shopify.com. Slash Ion AI? Oh, carry on. Hi, my name's Craig Smith and this is Eye on AI. I just got back from Las Vegas where I was attending Amazon Web Services re-invent conference, a cloud conference for developers. I had the opportunity to speak with Atul Deo, who worked on developing Amazon CodeWhisperer, a generative AI application to help write code, and now heads Bedrock, a service that gives developers a choice of high-performing foundation models to create generative AI applications. He talked about RAG, retrieve, augmented Generation and building agents that execute tasks. Atul gives a clear and clear-eyed overview of today's tech landscape within the context of Amazon's ecosystem and beyond. I hope you find the conversation as interesting as I did.

**Atul Deo:** 3:30

Hey, this is Atul Deo. Here I'm the general manager for Amazon Bedrock. I manage as part of my role. I manage product and engineering for Amazon Bedrock. Prior to AWS, my background was in computer science. I'm an engineer. I wrote code for a few years after undergraduate degree and then I basically realized I was just in a cubicle writing code without understanding the implications of what I was actually writing code for, and I realized I should probably get educated on some of the business aspects, went to business school, went to USC for that and then, after business school, I joined Yahoo. This was back in 0809 when Yahoo was going through some interesting transition with the Microsoft acquisition offer and whatnot. And then I worked there for the next few years and I transitioned from doing finance business operations and then moving into corporate development, where I focused on mergers and acquisitions for Yahoo towards the end. And then I moved to Amazon in 2014 and joined Amazon's corporate development team where I focused on mergers and acquisitions, again for AWS. And as part of that role, I got to work deeply with some of our product leaders, product and engineering leaders in AWS and I really, instead of just being on one side of the table where I am evaluating companies to acquire them. I wanted to kind of actually be a founder myself and I wanted to do that in a very low risk way and I realized that AWS is probably the perfect kind of place to do that. And this was around the 2016-2017 time frame when machine learning the team at AWS was being formed. It was kind of thriving under Swami and I decided to join the team and since then I've been involved in a bunch of different projects. I helped launch Amazon Transcribe. Then I launched a number of AI-powered capabilities for Amazon Connect, which is our contact center software services. Specifically, they're infusing machine learning into customer care, so things like. I launched a service called Contact Lens for Amazon Connect which essentially helps transcribe customer conversations. I'm sure you've called a customer care scenario like you called an airline company or a credit card company with some complaint and these companies want to analyze, kind of what conversations happen between their customers and their customer care professionals and essentially, contact Lens allows you to extract a bunch of different insights from those customer conversations. What is the sentiment? What are the key topics that customers reached out to? What are the trends? Did the agent, the customer care agent today, actually behave properly? Did they follow all the rules that a company had laid out for them? And essentially here we use the power of AI to do this analysis at scale and help the companies understand this better. I think from that point onwards I was like I think I have invested enough in infusing machine learning into other products, within existing products, within AWS. And then generative AI the term was not really coined back then in 2028, but we had started to see the rise of large language models, and these models were materially different from what we had seen in the past. So I embarked upon development of Amazon Code Whisperer, which is our ML-powered code generation service, and I helped launch it in preview, and since then I've been involved in development of Amazon Bedrock. So Code Whisperer is highly specific to code. But at that point I basically realized that this thing is way beyond code. The generalizable nature of this technology could essentially allow people to build a host of different applications just outside of even code and helping developers kind of get productive. So that's basically the short.

**Craig Smith:** 7:14

So Bedrock is a service to help customers build generative AI applications. You were saying earlier that you guys don't really use the term platform. It's not. There isn't necessarily an interface that you're working in. But before we get to Bedrock and Gen AI, I want to. Well, code Whisperer is generative, so I wanted to ask about where you are with the companies with Code Whisperer. I'm not a coder and two or three years ago I was talking to people who were imagining the day when you could use natural language to talk to a computer and it would write code for you. So when Code Pilot and then Code Whisperer and the other code generation tools came out, I got very excited. When AutoGPT came out on GitHub, I got very excited. I thought, great, here it is now. I can talk to the machine and it can code something for me. And there were a plethora of articles in the tech press about. You know I coded this app with Code Whisperer or something, but when I tried to do it, it really is an auto completion tool, it's not a code generation tool. How far are we from having end to end code generation tools that then you could pass the code to review AI that would spot problems and fix them? And yeah, so I would just kind of expand on that right.

**Atul Deo:** 9:10

So code autocomplete has been around for a long, long time. Like you had things like IntelliSense. I mean, I used to write code, as I said, in the early part of my career. I wrote a lot of NET and C-Sharp and Visual Basic code, and obviously the famous kind of IntelliSense recommendation would give you kind of the next one word or kind of. Probably it gave you after a few years to start giving you a little bit more. But that is materially different from what is happening now. So with these large language models, it's not just a tiny bit of autocomplete. At the very least you're getting a quick one, completion of the entire line of code, or you're getting several lines of code, and you can just even get just say in plain English that hey, I want to write a function to do so and so and based on the context of your prior code or rest of your code in your kind of project, you'd get several lines of code. In fact it'd complete the entire function for you. That is materially different from what we had in the past. It's just a simple autocomplete of one or two words. So the key difference though, and kind of where you want it to be, versus where things are today. You still have to have the developer's mind, where you need to decompose your problem into a set of smaller problems and then basically tell, kind of whatever tool you're using, let's say code whisperer. I want to write a function to do so and so. But usually, as you know, real apps don't only have one function, it's usually a combination of many things. So the developer today still has to put together some sort of a plan, as hey, here's kind of my logical flow and here are the various things that I need to do. So that's basically what is happening. That's the current state. Is it?

**Craig Smith:** 10:54

It is possible to use an LLM like GPT-4 to give you the steps to break down a problem into these sub-problems.

**Atul Deo:** 11:07

So that's basically the right way to do it today, where you can use some of the reasoning capabilities of the model to basically say I have a big problem, kind of how do I actually come up with the right architecture? And because the model has seen a lot of code in the past, it'll generally tell you kind of hey, what are the best practices? And if you are building a specific thing in mind, here are the kind of key kind of classes that you may have to create, here are the key functions, key methods, and kind of here's those general structure, and it might even tell you here are the next set of functions, and it might start giving you code. But I think that the key point, though, is ultimately the human in the loop still needs to roughly understand and verify. It's not that it's gonna give you an app that's gonna be perfect. It might do that for simpler apps, but inevitably, if you are a builder, nothing plain vanilla is ever acceptable to you. You wanna put your own twist and turns on the thing that comes out of the model. I mean, that's always been the case. Builders want to be creative.

**Craig Smith:** 12:07

Yeah, and the code review tools? If you generate an app that combines multiple functions that have been written by code whisperer, based on the roadmap given to you by GPT-4, and it doesn't execute, are the code review tools strong enough to go through and say, well, you have an error here, or so?

**Atul Deo:** 12:44

two parts. One is, I'll tell you, the inherent feature of code whisperer is it identifies security vulnerabilities in the generated code, and it is not just in the code that it generated, because usually it's a combination of what a developer wrote and what was generated by the model, and it'll give you kind of a list of security issues in the overall code base as current window, but separately even code reviews. We already launched a service called Codeguru I think in 2019 already, which was even before the advent of generative AI, where it could analyze lines of code and just do code reviews using the power of machine learning.

**Craig Smith:** 13:23

Okay and then I'll play around with that Bedrock. Describe then what Bedrock is.

**Atul Deo:** 13:34

Sure. So Bedrock is a simple API that allows any developer in a company to access some of the best large language models or foundation models, without a developer having to understand nuances of machine learning or having to manage any underlying infrastructure. They don't need to know what kind of GPUs, what instance, none of that. All they need to know is what is the application that they are building, and they just need to be able to interact with the model in plain English. Give them prompts and the model gives you a response.

**Craig Smith:** 14:06

Yeah, I was saying to you earlier at the expo part of a reinvention which is where we are in Las Vegas, there was a booth for something called PartyRock which is built on Bedrock and it's a simple user interface to build GenAI tools, or applications. The guy was showing me that you can pick your LLM, so to me it looked a little bit like an orchestration layer where you're picking your LLM and then you're talking to that LLM. Is that right?

**Atul Deo:** 14:54

That's exactly what happens in Bedrock. So, as I said, the developer has a choice of different LLMs. In Bedrock. We support a number of them from some of the best startups in the world and including Amazon ourselves. So we have developed a set of foundation models which we call as Titan. It's a family of models and the developers can pick any of these models, because different developers have different needs for different models, based on their specific use cases. They may have certain requirements on accuracy, latency, cost and whatnot for their use case and, depending on that, they will basically choose the right model.

**Craig Smith:** 15:28

Right, and then, in Party Rock's case, it has a UI that you put your prompt in and then it builds a very basic app based on that Is one thing that I was asking and maybe you could answer the problem with. And Party Rock is, at this point, kind of a sandbox for people to play in. It's not producing. It's for educational purposes Right but the core problem with large language models being used in any certainly in any safety critical application, but in any application for which ground truth is important, is hallucinations. Is there a companion service within AWS?

**Atul Deo:** 16:36

Yeah, so it's part. Actually, we launched a feature within Bedrock we call Knowledge Basis for Amazon Connect. So essentially, as you may know, there is a very common technique in the generative AI world that has evolved over the last few months, which is called retrieval augmented generation or RAG, right? So essentially, the large language models are pre-trained on a broad set of public data and they don't really understand two things. They don't understand the latest information because they were trained up to a certain point. So the second thing they don't understand is they don't understand the context of your company and your particular documents, right? So even within the company, the documents can sometimes be changed, because maybe the CEO of a company decided to change the return policy. The return policy was certain on November 30th and it is a new return policy on December 1st. So basically, if a customer comes and asks the question about return policy on December 1st, you don't want to be giving it the wrong answer, right? So sometimes you want the latest and the greatest answer, but you also want, at the same time, the capabilities of the generative model to give you the right answer. So, essentially, with retrieval augmented generation, you extract the right snippet of information from your company's data and you include that as part of the context for the model when you pass your request. And the model takes into account two things. It takes your actual specific request. For example, I may summarize this talk. You give a doc to it. But here you may say, hey, here's a bunch of information, here's the question and here's the proportion information the customer is asking about return policy. And here's the exact updated snippet from our internal documentation which talks about this return policy, which was changed yesterday. And here's the thing now the LLM is able to take that into consideration while it crafts its response for the end user.

**Craig Smith:** 18:28

Right, and that maybe you can talk a little bit about RAG, because I haven't delved too deeply in it. Yeah, just describe the architecture. What kind of database you're using.

**Atul Deo:** 18:44

So let me just describe RAG first in more technical detail and then I'll explain our specific solution for that. So essentially, think about a company. So the company has a bunch of documents. Now the first step they have to do is to break those documents into smaller chunks and then pass those individual chunks into what we call an embedding model. Now, this embedding model is not really a generative model. It takes text as input and it generates mathematical representations or vectors for those chunks of text. Now these mathematical representations capture semantic meaning of the text input text in a way that it can understand, like the model can understand better. And then essentially, the company has to. Let's say it has a million documents. It has to. It creates. It takes this million documents and says, creates 10 million chunks and then takes those 10 million chunks, passes them through the embeddings model and now it has corresponding embeddings. Now these embeddings need to go in a specialized store which is called a vector database. Now, this is like a new class of related products that has emerged in the last few months, and this vector store, vector database, is optimized for storing embeddings but also retrieving embeddings. Now let's say the query comes in from our example before I'm trying to ask about a return policy. Now I can't compare text with embeddings, I have to compare embeddings with embeddings. So when a query comes in asking about return policy, I again pass that same query to the same embeddings model and it generates the corresponding embeddings for the input query. And now the vector database does matching between the what it has in terms of its corpus the 10 million, the embeddings corresponding to the 10 million chunks and the input the corpus corresponding to the input query and then it's able to quickly find out hey, based on this end users query, I was able to find that right chunk which corresponds to the return policy, and it didn't return. Be back to the specific text. It says you know, the new return policy as of December 1st is so and so. So now I was able to extract that. Now the next step is I've extracted this information. Now somebody has to do the work, that's particularly a developer has to do the work to then take this in real time, run time and then pass, append it to the prompt. So the original question to the model or the application was what is the return policy? Now the developer has to take this information from the vector database, append that to the before it passes, make that request to the large language model and then the large language model generates the response. Based on that final compiler, it's called basically that's why it's called retrieval augmented, because you're augmenting that output from the vector database into the prompt and that's why you're getting the generated response after augmenting it.

**Craig Smith:** 21:51

Right. I mean I've had on the podcast Ed Enouf, who's from Datastax, which is a vector database company, and I've had a Edo Liberty from Pinecone, but they didn't use the term RAG. Is that because they're only building the database part?

**Atul Deo:** 22:14

So now the interesting thing is vector databases have been around for a while, but they did not have this popularity with the generative aspect. Rag has this retrieval augmented generation. So that generation part only became popular in the last, I would say, 12 months, since they went off for popularity of large language models. So now let me just switch gears. So we've discussed what RAG is, now let me just talk about what Bedrock does for RAG. I think the answer is pretty obvious. There's a lot of undifferentiated heavy lifting here. The developer, as I said, has to develop. Let's say, the developer has a bunch of documents in S3. S3 is our place where people store their data and now, from that point onwards, they have to take these documents, chunk them, pass those chunks to an embedding model, then store them in a vector database. So, which means they have to go sign up for some vector database, create an account, manage permissions, and then they have to manage this runtime workflow of hey, when a query comes in, I have to then create the embeddings for that input query, get the retrieved answer, append it to the prompt and get a final response. There's too much going on. So with knowledge bases, a lot of these aspects are automatically managed but at the same time, while they are managed, we give control to the developer. So the things like they get control on what kind of chunking strategy they had to use. There are different ways of chunking. I use a very simplistic example of saying one million documents, 10 million chunks. But there are different ways in which people can go about it. They can also pick a different embedding model. They may want to pick a different vector database. So we are. We are natively integrated with Pinecone, with Readers, in addition to our own vector database, which is the open search serverless engine vector engine. So in fact, with open search, we don't customers don't even have to go to the open search console. If they say they want to use open search, we actually create the resources on their behalf and they get it automatically.

**Craig Smith:** 24:21

the embeddings get created for them automatically and then they can go and access it and the vectorization of the documents that you want to put into the language model. Does Amazon also have that end of the workflow where you just upload a million documents Exactly?

**Atul Deo:** 24:45

Exactly so. The only thing a customer needs to do here is upload a bunch of documents in S3. After that knowledge basis for Amazon, bedrock does all of this heavy lifting for you and it also has a query API so you can simply pass your end user query which is hey, how do I, what's the latest return policy? And all the sausage making in the back. All they get is the answer to the latest, based on the latest return policy documentation, and they get an answer and then they can do many things with it. They get there's also a generated API which they can discuss this to the specific model and then get the final response. So all that management is automatically taken care of for the developer.

**Craig Smith:** 25:29

Yeah, okay, I have another question that's kind of related. I've been talking to people a lot and hearing people complain a lot about rate limits, about them. You know, the quota allowed to an individual customer in terms of queries per second, per minute or tokens per minute is not big enough to build heavy enterprise level applications. And I've been having kind of a debate with a CEO of a company that I know pretty well, who says, oh, that problem being solved by being able to compute inference on CPUs instead of GPUs, by distillation, by using sparse models and that sort of thing. But when I talk to people, other people, one guy in particular at a company called Nomad data, says no, this problem is fundamental. It comes down to GPU availability, which comes down to Silicon starts, which comes down to, you know, $20 billion faps. You just need more faps. So where's the truth?

**Atul Deo:** 27:01

Yeah. So I would say a lot of what you see in terms of limits. This is a true problem. If you look at almost every service out there. People are bottlenecked on GPU availability, like suddenly we had this massive spike. I mean, few things happened at the same time. We had COVID, supply chain crisis, rise of large language models, massive demand for GPUs and now down, this time for crypto mining, but for for LLMs particularly, an inference where people are seeing the value in what generative AI brings to the table and everybody wants to infuse generative AI, but at the end of the day, there is a compute shortage. So that is basically what is happening. But One of the things that differentiates AWS here is years ago in fact, this was exactly the time and almost 10 years ago when I joined Amazon, we had started investing in our custom silicon chip development efforts and we acquired a company called Pruna Labs. It's Israeli based, I'm sorry. What's the company Under Pruna Labs? So it's an Israeli startup and while the initial focus was mostly around networking because AWS we are kind of that's the kind of core, kind of we are good at infrastructure right so that was the initial focus. But quickly we the team started investing in specialized chips for machine learning, and specifically machine learning training and machine learning inference, and that gave birth to what we call today as tranium, which is chip optimized for training, and inferential, which is a chip optimized for inference. Now we make our own chips here, so of course, we are still a great partner within video, as you saw in Adam's keynote today, but we also have our own chips, which gives us a tremendous advantage in these times where you know there is a shortage of GPUs.

**Craig Smith:** 28:52

So what I've heard from the people who are saying rate limits are a bottleneck, that's been here for a few years, they say that that's you know. Big companies are experimenting, building pilot products, but no one is putting major generative AI products into production on an enterprise scale because they just can't get the throughput, the token throughput. Is that true?

**Atul Deo:** 29:32

I don't think that is the case. I mean, we have plenty of customers who are starting to go into production and do very interesting things. As I said, I think if the company, if they are using certain cloud providers where they are only solely dependent on GPUs, then that is probably going to be the case. But, as I said, in our case we are, we are diversified here. We are in addition to GPUs. We are also using a lot of our tranium and inferential.

**Craig Smith:** 29:57

Yeah, what about GPT-4, which seems to be the preferred model for the time being? Does the constraint exist there?

**Atul Deo:** 30:09

So GPT-4 today, GPT-4 is not available on bedrock. It is only available on Azure Open AI. So I'm assuming that Azure Open AI is probably running on GPUs.

**Craig Smith:** 30:20

Right, right, but so maybe this is a problem that exists primarily with GPT-4, but other foundation models, depending on how their architect or which chips are using.

**Atul Deo:** 30:36

I think in your framing of the question you touched upon a lot of interesting topics. So things like quantization, things like distillation. So I think, instead of going into the specific nuances of that, I do want to say that, hey, people have different problems and you don't need the most expensive, the most powerful tool, your most compute, hungry tool to solve every problem. And that is the approach which we have taken with bedrock, where we have a range of different models of different levels of capabilities. So, for example, we have Titan light, which is a very small, lifty, fast model, which gives customers tremendous cost advantage and it can be fine tuned for their specific use cases. On the higher end, we do have Claude, anthropics Claude, which is a very powerful and capable model, which customers find very comfortable or better in many cases than GPT-4.

**Craig Smith:** 31:35

Right. If you're using RAG in a vector database and you're only using the model to compile the language, the natural language, or to recognize the natural language on the input side, do you not need a powerful model?

**Atul Deo:** 31:57

I think RAG. It seems very simple on paper, but there are different flavors of RAG. If you have a very simple use case where all you're doing is just retrieving the context, sending it to the prompt and generating a response, then the answer is yes. But if it involves multi-step reasoning and thought processes, where do I get this information? Do I need to execute some API calls? Maybe I need to understand the customer's history, or the customer's history stored in the Dynamo DP table. Let me go get a sad history and then, based on that history, I need to do so and so. So if there is a lot of complex reasoning involved in generating that final answer because all of this what happens in this multi-step process is, as I'm retrieving new information, I just keep appending all of it and including it as part of the original request and what started out as a very small, simple request. Obviously some simple sounding questions can be pretty complex in reality, so you may have to do multiple things to get to the final answer. So that is, if the answer, if it's a simple, single step kind of retrieval, augmenting, generating then you should in fact use a simpler, smaller model, because that is going to give you the best bang for your buck, but if you are going to require some complex reasoning orchestration, then you should use a more capable model.

**Craig Smith:** 33:27

Yeah, so Bedrock. How long has Bedrock been around?

**Atul Deo:** 33:32

So, Bedrock, the service became generally available in September. End of September.

**Craig Smith:** 33:38

Yeah, and, as you said, it's a service, it's not a platform. So how does someone use Bedrock?

**Atul Deo:** 33:46

So again, I think about the nuances of what the platform is. We can go and we have each other said we don't usually call anything in AWS a platform, but Bedrock offers a range of different capabilities, right? So, as I said, the most basic capability is giving access to the models via an API. But in addition to the plain models as an API, we also allow companies to customize the model. So there are two ways of customizing. Predominantly, one is called as fine tuning, which is the most common way of customizing and essentially involves use of label data. So essentially, me saying, hey, this is a resume that I like, this is a resume I don't like and I'm a if I'm a recruiting firm, I give 100 examples of that, and I fine tune this model. Next time I give a new resume, it will know exactly what my preference is and how to classify it based on my likes or dislikes. So that's an example of fine tuning. There's another example, an example of a way of customizing the model which we launched this morning is called as continued pre training. Essentially it is domain adaptation, so it doesn't require any label data. But all you're saying is, hey, I'm a company in a particular industry and I. We have a lot of jargon, company specific. Here are a million documents. Just understand more about me and my company and what we do and that way you can better answer my question instead of giving me some generic canned responses. And the advantage of this is you know the rag is good when you are, when you have a question and you need to inject context specific for that question. But sometimes you just want to give it kind of a general, kind of education, the model general education on you. So this is the domain adaptation capability that we've added. So that's just recapping right in terms of what bedrock does. One is plain access to the models via the API. Second is customization. We are fine tuning and continuing pre-training. We also allow the managed rag experience and then we've added a capability called as agents. As I was saying, oftentimes when you ask a question like, for example, I'm just going to give a simple answer like , did you buy the groceries? And? Or I don't know, I'll just let's take an example. The question can be simple but you may actually have to go and buy groceries or do some action as part of that question. So what happens with agents is the companies can provide the context of their data stores. They can give context of their APIs and basically say, hey, based on the end user request, go, take some action, execute some things and make use of the data sources to answer the questions or execute their commands. And agents do all that without, kind of without developers having to manage the context of the session back and forth, without having to write complex prompts, so that seamless experience is provided by bedrock agents for Amazon, bedrock. And the last thing I would add is we've also launched something called guardrails. So today the models themselves have some inherent guardrails which they don't allow certain topics to be discussed or they don't give responses to certain things. But companies also want to create certain rules or guardrails for their specific applications and they may vary based on individual application within their own company and they may want to apply multiple guardrails for their application. So we introduced or launched this new feature for bedrock which allows companies to basically say, hey, here are some denied topics. If a question from the end user comes on this, give this scan response or basically say, hey, there's certain topics like violence, hate, sexual in nature and I want certain filtering. So there is a default filtering. You can set it to low, medium or high. You can change the filters for inputs and outputs. We also have an upcoming capability here called as redaction POUR redaction. So you can redact the inputs going to the model PII inputs and you can redact the PII outputs coming out of the model. And also you know the last but not least, which is like a conventional block list. I don't want any time a certain word is said, basically kind of either mask it or kind of basically say I'm not going to answer questions if a certain word comes up. So those are the various controls. In addition to that, we're continuously doing this. This landscape is evolving so rapidly. So our goal is obviously giving customers access to the best large language models, foundation models, is, is the starting point, but what we want to do is give developers a set of tools so that they can build generative AI apps easily.

**Craig Smith:** 38:33

And to access this all you need is your AWS login Exactly, and there's a tab or something for bedrock. Yeah, bedrock, and it has all the various correctional facilities.

**Atul Deo:** 38:47

And you can. I mean you can. There's even a playground within Amazon, bedrock. You can just select, select a model and start interacting with the model by asking it some questions and you can copy-paste some context and you can see what the model responds to in the playground itself. There are interesting configurations. You can set a temperature, to say one temperature one means the model is going to give a more kind of wider response. Temperature zero means you want a model to be more conservative or kind of have less variations in its response, kind of like sometimes even humans. We tend to have our own implicit temperature on a given day.

**Craig Smith:** 39:26

Right, that's right. Yeah, and how do you charge for this? Is it by throughput or, yeah, it's usual it's per thousand tokens.

**Atul Deo:** 39:37

There's different pricing for the input tokens and different pricing for the output tokens. So typically what we see in production is customers tend to have a lot of input and outputs from the model tend to be smaller.

**Craig Smith:** 39:52

Is that right? Is it for a typical application? And again, I'm thinking personally here, is it prohibitively expensive for someone who just wants to build an app for themselves?

**Atul Deo:** 40:06

It's not prohibitive like that at all. I mean you can do it in tens of dollars. You can experiment and kind of quickly get a lot of value from it.

**Craig Smith:** 40:13

Yeah, and how much support is there? Because AWS is famously self-serve. You know it's. You can configure machines and everything on the website, but is it simple enough that someone can navigate on their own?

**Atul Deo:** 40:32

is simple enough. I think there are a number of things that we have done here. The first thing is we have a lab, so you can basically do zero cost lab. You can just kind of go and do it in a sandbox environment and you can play and you can actually make use of bedrock to implement a few things based on the particular project. We also have a number of training courses that we've launched on generative AI and specifically on bedrock, and we're going to launch some more advanced courses over the next few weeks. And then for a lot of enterprise customers, we also have what we call as Genai Innovation Center, which essentially for companies who have not, who don't have a lot of experience in building Genai apps, we have a set of experts on our end who work closely with the right POCs in the company to build an app and help them get started.

**Craig Smith:** 41:23

Yeah, and we started out talking about Code Whisperer. Is it integrated with bedrock?

**Atul Deo:** 41:30

Code Whisperer. So the way I would think about it is we have a three layer stack, right. So the bottom layer, you have the infrastructure layer, where you have the chips that I talked about. Then we have the middle layer, which is bedrock, which is kind of everything that is built on top of bedrock in the Genai world, and then and then the top layer, you have the application layer. So Code Whisperer is an application, so developers don't have to know about what model, what prompt, they are just doing their job. And what we also launched the top layer this morning was Amazon Q, which is, you know, it's an application for both business users and for developers.

**Craig Smith:** 42:03

Right, where do you think we have a few more minutes? Where do you think this is going? You know, I talk periodically to Yanlacun and he says the day will come soon where everybody has a generative AI, personal generative AI assistant that they're interacting with throughout the day, and certainly companies. There's tremendous promise for companies and operating their businesses, optimizing their businesses. Do you have a vision of where this is going within, say, five years?

**Atul Deo:** 42:48

Yeah, so I think a few things are happening. I think again, for me. Usually I see history repeating itself. It's just a different kind of intensity, of what the repetition is happening with. So, in the past, we've had the industrial revolution, we've had, obviously, arrival of calculators, computers, and this is, for me, this is just the next generation of that disruption, where, today, we can't imagine a world without our phones, without our computers, and this, with the generative AI, with large language models, essentially, we will be able to communicate, get far richer responses. On a personal productivity level, we can also offload a lot of repetitive tasks, multi-step tasks, with things like agents. I think this is all happening very fast and furious, and I think the main thing is, though, that these things are often 98% good, and today and that's the last 2% is where we'll really have to kind of, the space is moving rapidly, so my biggest kind of thing in 2024 is how do we close the gap on the last 2%? And the closing of the gap on the last 2% is happening with is going to happen with better models and also with better tooling around the models, so things like the concept of RAG emerged very rapidly over the last 18 months, so there are going to be the next set of RAGs and other techniques that help us get closer to that 100% mark, and that, basically, will be, you know, dawn of a new era where we will be all more productive because we'll not be doing boring stuff, we'll be doing more interesting things.

**Craig Smith:** 44:27

Yeah, I'm fascinated by the work on world models. I've had Yanlacunan talking about it, a guy named Alex Kendall with a company called Wave AI. They have a model called Gaia 1. Yeah, are you guys working on world models as opposed to language models, models that learn directly from the environment or from video or other forms of unstructured data?

**Atul Deo:** 44:57

So, again, as I said, we are working. We have launched a set of 1P models which we call as Titan, and we are continuing to develop Titan in a host of different ways. Multimodal inputs are going to be critical for us because you know, there are only so many things that can be captured just purely by text or by images or by video. You do need real world understanding because, ultimately, if you look at how, like my, I have a five and seven year old and I see kind of how they learn, they learn basically based on the real world, that is, they don't have to be fed the entire internet to get smarter, so they are learning rapidly from social cues from the real world. So I do expect models to learn a lot from the real world and physical world.

**Craig Smith:** 45:39

Yeah, it's an exciting time. I mean it must be crazy working at one of the big tech companies like Amazon. You know, even RAG. Nobody was talking about RAG that I was aware of six months ago. How do you keep up?

**Atul Deo:** 46:00

For us, one of the best things we have going is an amazing set of customers.

**Craig Smith:** 46:05

Yeah.

**Atul Deo:** 46:06

And the last since yesterday. I already have been in so many customer meetings where customers are relentlessly hungry. They keep coming to us with, with asks and the dissatisfied customers the best customer, because that helps us, keeps us on our toes and helps us in a way. So, as long as we continue to listen and capture that feedback I think that's basically I am anticipating how we will stay up to date and kind of keep innovating.

**Craig Smith:** 46:31

Hi, good tech solves problems you know about. Great tech solves problems you haven't even thought about. What can the commerce platform trusted by millions of merchants do for you? For Shopify, the commerce platform revolutionizing millions of businesses worldwide. Whether you're a garage entrepreneur or IPO ready, shopify is the only tool you need to start, run and grow your business without the struggle. Shopify puts you in control of every sales channel. So whether you're selling satin sheets from Shopify in person point of sale systems or offering organic olive oil on Shopify's, all in one e-commerce platform, you're covered. Shopify powers 10% of all e-commerce in the United States and Shopify's truly a global force powering all birds, rothes and Brooklyn and millions of other entrepreneurs of every size across over 170 countries. Plus, shopify's award winning help is there to support your success every step of the way. Sign up for a $1 per month trial period at Shopifycom. Slash I on AI. That's Shopify S-H-O-P-I-F-Y dot com. Slash I on AI. That's E-Y-E-O-N-A-I all run together. Go to Shopify.com slash I on AI to take your business to the next level today. Excuse me, sir, I couldn't help, but over here Did you say Shopify, shopifycom? Slash I on AI Carry on. That's it for this episode. I want to thank a tool for his time. If you want to read a transcript of today's conversation, you can find one on our website. I am AI, that's E-Y-E-O-N dot AI. And remember, the singularity may not be near, but AI is fast changing your world, so pay attention.