05232023 Build Kevin Scott Keynote

Build 2023
Kevin Scott
Greg Brockman
Tuesday, May 23, 2023

KEVIN SCOTT: That was an amazing video and thank you Satya for sharing it. It really is inspiring to see this technology getting diffused so quickly and having a real positive impact across the globe, not just in the global – the urban innovation centers here in the United States and in the capitals of the industrialized world.

So I'm so excited and happy to be with you all here today, in person, at Build after a four-year hiatus, and I guess it goes without saying that a lot has changed in the world of technology over these past four years.

One of the biggest changes, and the theme of this conference is what has happened in the world of AI just even in the past year, and what that means for you all as developers. I'll go to my first program, as a developer, in the early '80s, when I was 11 years old, I think, and I remember what a thrilling moment that was, like being able to do something for the first time that I didn't even realize was possible.

And I've been chasing that feeling my entire career, like trying to find those moments where something impossible became possible, and then as a developer, figuring out how I could participate in that change.

And the thing that I can say is, like the most exciting thing in the world, and maybe the most exciting time that I've experienced in my career, is what that power of AI is doing right now to help all of us have that moment, that ability to take something in our hands, to look at what was possible, what was impossible and becoming possible now, and then going and doing something great with it.

So I'm going to spend my next half-an-hour or so chatting with you all about some of those technological themes that are driving all of this great progress in AI that we're seeing. So we're going to start with, you know, maybe the obvious thing.

So there's an incredible amount of attention being paid right now to what's happening with the rapid progress with these AI models, these foundation models that we're calling them now. And in particular, like the rapid pace of innovation that's being driven by OpenAI in their partnership with Microsoft.

We really are setting the pace of innovation in the field of AI right now. And I think even for us, it's been surprising to see how much of the zeitgeist is being captured by things like ChatGPT and applications that people are building on top of these large foundation models.

The reason that this partnership between OpenAI and Microsoft has been so successful is that we really do have an end-to-end platform for building AI applications. We built the world's most powerful supercomputers, we have the world's most capable foundation models, either hosted that we built ourselves and make available to you all via API, or open source, which run great on Azure.

We also have the world's best AI developer infrastructure. So whether that is using these super powerful computers to train your models from scratch or to build these applications that we're going to be talking about at Build this year, on top of that infrastructure, like, we have that end-to-end platform, and you're going to hear a ton about it today and tomorrow. Scott's keynote is right after mine. Like, he's going to dive into detail on a bunch of this stuff. And then the breakout sessions are going to be amazing, and like equip you all with the information that you need to go do some pretty awesome stuff.

So you know, this end-to-end platform starts with Azure. We really believe that Azure is the cloud for AI, and it's not just the amazing technically complicated and brilliant work that our partners at OpenAI have done on top of all of this infrastructure, but it's things that the teams at Microsoft are doing to build our Copilot applications and our own advanced AI models.

And it's also the things that our partners, some of you here in this room, are building on top of Azure, making Azure this amazing platform for doing the most ambitious flavors of AI in the world. But it's not just Azure. Windows, we believe is the best client for AI development. And you're going to see a bunch of that today, and Panos is going to dive into it pretty deeply tomorrow.

So Satya showed the Windows Copilot, which is going to be an amazing part of your productivity story, like GitHub Copilot works great on Windows. But increasingly what you're going to see is the ability to run these powerful AI models on your Windows PC so that you can develop these true hybrid AI applications that span the edge all the way to the cloud. And it's just a really, really exciting thing.

But what I'm going to spend most of my talk discussing with you all is this idea of the copilot. Satya has already referenced a whole bunch of the copilots we've launched, like you've – you know, as he said, you know, it's almost as if we woke up on January the 1st and decided to do a whole bunch of press releases.

But it's really been years of work where we have built a platform for building copilots that has enabled us to do these amazing releases that we've been doing, and we are sharing with you all today some of those patterns that have helped us build copilots and showing you and opening up our platform so that you can build copilots of your own.

And so just to start with, like a copilot, simply said, is an application that uses modern AI, that has a conversational interface that assists you with cognitive tasks. And we're going to talk a lot about what that means later.

We believe that it must be an open ecosystem. So like one of the most important things that we believe is, even though, like there are a whole bunch of copilots that Microsoft has built, that maybe the most interesting copilots that get built are by you all, using these powerful tools that you have, both on Azure, on Windows and in the open-source community.

So you know, as we start talking about this, I would love to bring to stage Greg Brockman, the president and co-founder of OpenAI, to talk about his experiences building GPT-4, like this powerful model that's powering a bunch of these copilots, and about ChatGPT, which maybe is the most interesting copilot in the world right now.

So please join me on stage Greg.

(Applause.)

KEVIN SCOTT: Awesome. Fantastic. So thank you so much for joining us today here at Build.

So I wanted to start with the ChatGPT experience. So like, I believe it's caught us all by surprise, like just how crazy the adoption of ChatGPT has been and how much interest there is, but – like, it's a really big engineering challenge to build something like ChatGPT. So maybe you could talk a little bit with us about that.

GREG BROCKMAN: Yeah, you know, ChatGPT was a really interesting process, both from a infrastructure perspective and ML perspective. We'd actually been working on kind of the idea of having a chat system for a number of years. We'd even demoed at Build an early version called Web GPT, and it was cool. It was a fun demo. We had a couple hundred contractors, literally people we had to pay to use this system, and they were like, "Eh, it's kind of useful, it can kind of help with coding tasks."

But for me, the moment that really clicked was when we had GPT-4, and that we had a traditional process GPT-3, where we'd just deployed the base model, so we had just been pre-trained it, but we hadn't really tuned it in any direction, and that was in the API. For 3.5, we'd actually gotten to the point where we're doing instruction following where we had contractors who were given, "Here's an instruction and here's how you're supposed to complete it." And we did that training on GPT-4.

And the thing that was so interesting was I just – as a little experiment I was like, "Well, what happens if you follow up with a second instruction after it already generated something?" And the model has replied with a perfectly good response that incorporated everything from before then.

And so you realize that this model was capable enough. It had really generalized this idea of, "Well, if you really want me to follow instructions, and you give me a new instruction, maybe you really want me to have a conversation."

And so for me, that was the moment that it kind of clicked that, "OK, we have this infrastructure that's already in place with the earlier model," and this new model, even just using this sort of

technology that wasn't meant for chat, it wants to chat, like it's going to work. And so this was a real "aha" moment. And from there, we just were just like, "We've got to get this thing out. It's going to work."

KEVIN SCOTT: Yeah. And I think it was really surprising to me. I remember when Sam called me up and said, "Hey, like, you know, we want to release this ChatGPT thing, and like – you know, we think it's going to be a few weeks' worth of work to condition one of these models," and I was like, "Sure, why not?" And I had no idea that it was going to work, technically, as well as it did, and that it was going to be such a crazy success.

And so, you know, maybe related to that, I know that you are one of the principal architects on all of the infrastructure that was used to train GPT-4, and so GPT-4 powers parts ChatGPT, and it has just really been a revelation for everyone who's been working in the field of AI.

So I wonder if you could share a little bit of, you know, what are some of the interesting things that you found about the development of GPT-4?

GREG BROCKMAN: Yeah, GPT-4 was very much a labor of love. As a company, we had actually, after GPT-3, multiple attempts to surpass the performance of that model. It's not an easy thing. And what we ended up doing was going back to the drawing board, rebuilding our entire infrastructure.

And a lot of the approach we took was get every detail right. I'm sure that there are still bugs, and I'm sure there are still more details to be found, but you know, an analogy from Yaka (ph), who was one of the leads on the project, was that it's almost like building a rocket, where you want all the engineering tolerances to be incredibly tiny.

And so lots of little details – like for example, it used to be – it turns out that if we had a data bug in our checkpointing, where if you killed the job at exactly the wrong moment, you could end up with a blend which means, you know, new weights and old weights when the job restarted. Machine learning mostly doesn't care. It's happy to, to recover from that.

But it's one of those things where, every time you see a weird wiggle in your graph, you're like, "Huh? I wonder if this was that particular issue or if it's a real, real other one?" And so if you go back and you really pay attention to every single detail and just do the boring engineering work, like that is the main thing that I do.

KEVIN SCOTT: Yeah, well. I mean, the boring engineering work you do is like just at an unbelievable, phenomenal scale, but – like, I do think that that's a good, you know, parable for everyone in the room. It's sometimes the boring engineering work that like really leads to success.

So Satya talked a little bit in his talk about this shared approach that we're developing for plugins, this idea that we're going to empower all of these folks in the room to write software that can extend the capability of things like ChatGPT and like all of these copilots that we're building. And I know that that also has been an interesting technical challenge. And like, we still

don’t yet have all of the technical, you know, issues sorted out, and like there’s a lot of work left to do to – like, get it into the state that we ultimately want it to be in.

So I wonder if you, like, have some thoughts you wanted to share on that.

GREG BROCKMAN: I love plugins, yeah. I think it’s been a really amazing opportunity, both for sort of every developer to leverage this technology in a way that just makes the system better for everyone, right? And that’s what I think is so exciting, and part of the reason we designed it as an open standard was because, that way, as a developer, you build this thing once and then any AI can use.

And it’s such a beautiful idea, right? I think that the web part of what really drove it was anyone can build a website, and now everyone gets access to that, and then you build an API, and suddenly anyone can leverage it.

And I think that this kind of core design principle of really having any developer who wants to be able to plug in and get the power of the system and be able to bring all the power of any domain into ChatGPT is really, really amazing.

KEVIN SCOTT: Yeah, and the thing that I really love about plugins is conceptually it’s so simple. It reminds me a little bit about the first HTTP server that I ever wrote. Like if you understand the core concepts, you can stand up something very quickly that can do something very powerful, and like, I think that is an awesome, awesome thing as an engineer.

So you know, in your role at OpenAI, like you are constantly thinking about how to push the limits of the technology. And you know, I think one of the really amazing things about our partnership is working with you all. It feels like we get to see a little bit further into the future than we otherwise would be able to.

So I wonder if you could say a few things about what’s exciting to you about what’s over the horizon, like either with applications or with the models?

GREG BROCKMAN: Yeah. The thing to me that is interesting is we’re almost on a bit of a TikTok cycle, like you know, in Tel Aviv yore, where you kind of come up with an innovation and then you really push it.

And I think that with GPT-4, we’re kind of in that early stage of really pushing it, right, that we still have vision capabilities that have been announced, but we’re still productionizing. And I think that will just kind of change how these systems work and how they feel, and kind of the kinds of applications that can be built on top of them.

So I’m really excited to – if you also look back at kind of the history of it, over the past couple of years, I think we did like a 70% price reduction, two years ago, where basically this past year we did a 90% cost reduction, like a 10x cost drop, and like that’s crazy, right? And I think we’re going to be able to do the same thing repeatedly with new models.

And so GPT-4 right now, it's expensive, you know, and it's not fully available, but that's one of the things that I think will change.

KEVIN SCOTT: Yeah. And I think that is, you know, a thing that I would want to leave everyone here in the room with is that, you know – and it's what we say to all of the developers inside of Microsoft, building on top of these things, like what's expensive today won't be tomorrow because the progress there is so fantastic.

So I think we've got time to squeeze one last thing in. So you've already dispensed a bunch of like really great advice for developers here in the room, but like maybe one more thing that you would leave the audience with.

GREG BROCKMAN: I think that in this field there's – the technology is clearly getting better and better. But the thing that I think every developer can do that is hard for us, and even Microsoft at Microsoft's scale to do, is to really go into specific domains and figure out how to make this technology work there.

So I really love companies that are in the legal domain and really getting expertise and talking to lots of lawyers and understanding what their pain points are with this technology. And so I think that there's a huge amount of value to be added by the efforts of everyone.

KEVIN SCOTT: I think that's awesome, like you heard it from Greg. You all are the ones who are going to make AI great, but thank you so much, Greg, for being with us here today, and for all that you're doing. Thanks very much.

(Applause.)

KEVIN SCOTT: So one more interesting OpenAI thing that we're going to do. So we have Andrej Karpathy here. I think I see Andrej in the front row. So Andrej is going to be here on this stage later today doing a "State of GPT." So he's going to walk through the technology from beginning to end. Like, it's going to be an awesome session, like probably going to be tight on seating, so like, try to get your spot here. You are not going to want to miss that.

So let's talk about copilots. So Satya mentioned a bunch of the copilots that Microsoft has launched, you know, and that our partners have launched. So you know, we sort of think of ChatGPT as fitting this copilot pattern, Bing Chat, certainly GitHub Copilot, Microsoft Security Copilot, Microsoft 365 Copilot, and Designer, and many, many more.

So the thing that we noticed as we were building these copilots, starting with GitHub Copilot several years ago, is that the idea of a copilot is actually pretty general. So this notion that you're going to have a multi-turn conversational agent-like interface on your software that helps you do cognitively complex things applies to more than just helping someone do software development.

And that's sort of what you see. Like we have search copilots now, we're going to have security copilots, we have productivity copilots, and we're going to have all of the copilots that you all build.

And the thing that we noticed, for us at Microsoft, is that we needed to look at what is common across all of these things so that we can understand how to design great user experiences and what the technology stack is that is going to empower us to deliver these things safely, responsibly, cost effectively at scale.

So the only reason that we have been able to do this sort of blitz of copilot announcements, and delivering these products to users so quickly, is because we stopped and took the time and energy to go build a copilot technology stack that would allow us to move quickly with safety.

So one of the things that I want to talk with you about today is what that technology stack looks like. But before we dive into the details, like, you know, I think Satya's reminder to us all, like, why do we do what we do?

So one of the important reasons that we have taken the time to sort of think about this copilot stack as one coherent thing is platforms are important. It gives us the opportunity to build things that are more ambitious than you otherwise would be able to build. And it gives you, the developers, a chance to build things that wouldn't be possible if the platform didn't exist.

So I love this quote from Bill Gates. It may or may not be apocryphal, but it's still just been attributed to Bill for many, many years. And you know, what Bill is saying here is that the true value of a platform only materializes when the value created on that platform accrues to the people who are building on top of the platform, not the platform builder itself. And so, like, if that's not true of a platform, then it's really not a platform.

And you know, the thing that makes platforms even greater than all of that value that they can potentially produce is it prevents folks from having to bear the burden of building very complicated things from the ground up just to build the application that they want to go build.

It's great if you want to build all of this stuff, if you want to be a platform company or an infrastructure company, but if what you want to do is build a legal copilot like Greg was talking about, or you want to make a copilot for medicine, or a copilot for helping people get through their insurance claims, you are not going to want to build all of this stuff from the ground up. It will be economically infeasible.

The amount of compute that we are investing in and just sort of the scale of all of that infrastructure is absolutely astronomical. And the fact that the things that come out of the other end of the compute, these foundation models and this entire platform, that they are reusable and generalizable is like really a fantastic thing, and like, one of the things that we've been betting on for five years now, that this was going to be a durable property of these systems.

So one of the things that you're going to hear a lot about at Build is this idea that the foundation models are powerful and they're getting more powerful, but they can't do everything. And you shouldn't have to wait around until we train a model that can do the thing that you want to do. You should have ways to accommodate your application, build your application on top of this technology, even when the model itself isn't complete or perfect.

And so we're going to talk about a ton of ways that you can do that. Satya has already referenced plugins. Greg and I chatted about plugins. Like, plugins are going to be one of those powerful mechanisms that you use to augment a copilot or an AI application so that it can do more than what the base platform allows you to do.

And like, what a plugin may do is, it helps augment your AI system so that it can access APIs, and via an API, it can do anything, like change state in a digital system or retrieve information. Like for sure, people will use plugins to retrieve useful information. Like, you've already seen some video demos of that happening already and you're going to hear a lot more about that.

It allows you to perform arbitrary computations and to safely act on the user's behalf. And you know, really the way that we think about these plugins is they're almost like actuators of the digital world. So anything that you can imagine doing digitally, like you can connect a copilot to those things via plugins.

But what I'm going to spend the rest of this talk focusing on is the anatomy of a copilot. So what does a copilot look like? What is shared? What's common among all of these things that we built and like, what are the platform components that we are building to help you all build copilots of your own?

So this starts from the user experience. There are some things that are the same and there are some things that are different about building copilot user experiences. There is an application architecture, and there will be some familiar things about it, but like a bunch of new stuff to learn.

And then it is so important for all of us to think about safety and security. You'll inherit a lot of that by using the tools that we've built for you all. But like, you know, it's a thing that you need to think about from the very first steps of building your copilot applications.

I just want to start with the thing that doesn't change when you're thinking about building a copilot. You have to build a great product. It is something that we sometimes forget, but you have to understand what that unmet user need is, like what it is that you are trying to make better, where you have a unique understanding of that thing that maybe no one else has. And then you need to apply the technology. Sure, the tech is great. It's making a whole bunch of things that were impossible or infeasible or expensive, possible, easier and cheaper. But it does not absolve any of us of the responsibility of sort of thinking about what good product making looks like.

And one thing in particular that you have to really bear in mind is the model is not your product. Unless you are an infrastructure company, the model itself is just infrastructure that is enabling your product. It isn't the thing in and of itself.

And so, one of the mistakes that I've seen, just being in the tech industry over 20 years, is having people sort of fixate on infrastructure versus fixating on product. It's just the thing that we even have to remind our teams here inside of Microsoft over and over and over again, is use the

infrastructure that you have at hand, that is best going to enable you to solve your problem. Don't build infrastructure that you don't have to build.

And again, it's just up to you all; it's up to us. Like, we have to build great experiences, things that delight users. We've got to get things out into the hands of users as quickly as possible, see what works, see what doesn't work, iterate, make them better.

Let's dive into the copilot stack. Satya already showed this, and we're going to blow it up a little bit now.

This is how our copilots at Microsoft are structured. And these are some of the things that we're going to be diving into greater detail in subsequent talks for you all to have a look at, to pick up, to use, to learn about and to make things.

Some of this may look familiar. There are three boxes. You can think of these as roughly corresponding to the three tiers of a normal application. You've got a front end, you've got a mid-tier, you've got a backend.

The front end, the things that we've already talked about, is you start with understanding what your amazing product idea is. The thing that's a little bit different about the user experience design with copilots is we have more or less been building user experiences the same way for 180+ years, since Ada Lovelace wrote the first program. Like, we have had to understand what the machine is capable of, and then we are fiddling around with how we express the connection between the human and the machine in very explicit ways.

What that means for you all is fiddling around with user interface elements, menus, binding code to actions, trying to fully anticipate the needs of the user, and architecting your applications in a particular and familiar way, so that people know how to get at all the functionality, the capability that you really built into your code.

The thing that's a little bit different in a copilot is you're going to spend less time thinking about what your user interface widgets are and trying to second guess the user about what it is they want, because they have this really natural mechanism to express what it is they want: natural language.

And so, what you have to think about in the design of these copilots is what it is you want the copilot to actually be capable of. What are the things the model can't do that you need to augment with a bunch of the stuff that I'm about to show you in the orchestration layer, with plugins, and maybe even with finetuning models or using portfolio models to accomplish. But it's going to be way less of that fiddling around mapping user interface elements to little chunks of code than you're accustomed to.

And you also, on the flip side of that, have to think about what you want the copilot not to do. And so, this is important in how you're thinking about safety, but also because the thing at the bottom of the stack, these foundation models are sort of like a big bucket of unrestrained capability. You're the one who oftentimes has to restrain it to your particular domain.

For instance, with GitHub Copilot, a bunch of the work that we did is to keep the model on task, which is helping you solve your development problems. Like, you're not trying to figure out what the best menu item is on Taco Bell when you're sitting in GitHub Copilot, trying to write a piece of code.

That's the user interface, just broad brush, what is different there. Now, let's talk about orchestration.

Orchestration is the business logic of your copilot. And as I mentioned, when we started building our own copilot, every team inside of the company was building their own orchestration layer, all of that logic to figure out how to get a thing to sequence through all of the models, do all of the filtering, do all of the prompt augmentation that you have to do to build a really great app. And we just sort of noticed that there was commonality across all of those things.

One of the things that we did that greatly affected our ability to get these copilots out to market at scale and to do more ambitious things was to decide that inside of Microsoft, we are going to have one orchestration mechanism that we will use to help build our apps. That is called Semantic Kernel, which we've open sourced, and there's a session on Semantic Kernel later at Build, which I would encourage you all to attend.

But we also know that we're not the only ones who've seen that there's all of this commonality across orchestration. And there's some really great open source orchestration tools that work super well inside of the Azure ecosystem that we're building.

Harrison from LangChain – shout out to Harrison, who's here with us in the front row. Yeah, give Harrison a round of applause, please. (Applause.) LangChain is one of the most popular open source orchestration mechanisms. And Harrison, with a very small team, has built a thing that is useful to an extraordinary number of developers.

And orchestration isn't a solved problem. Like, we're going to see a lot of new ideas there. And the thing that I want to assure everyone here in the room is that you'll be able to use whatever orchestration mechanism you want. We'll give you some options that we think are great for us. Like, we'll point you to some of our open source favorites, but if you want to roll your own thing, that's your choice. I'm a developer. I like rolling my own stuff sometimes, too.

One of the things that you'll see in Scott Guthrie's talk that's coming up next is Prompt Flow, which is another orchestration mechanism that actually unifies LangChain and Semantic Kernel. And so, I encourage you all to go dive a little bit deeper there.

Inside of the orchestration layer, the fundamental thing that you're going to be manipulating is a prompt. And so, a prompt is just a bucket of tokens that is generated by the user experience layer of your application. It could be, in something like Bing Chat or ChatGPT, a question or a thing that a user is asking the model to do, or it could be something that your application constructs, like where it's not a direct natural language thing from the user, but a natural language thing that you are conveying to the model from your application.

And a big part of handling those prompts at the beginning stages of orchestration is prompt response filtering. Basically saying I'm not going to allow these prompts through because maybe they will cause the model to respond in a way that doesn't meet the needs of your application or do something unsafe. And you also filter responses on the way back up. After the model has produced a response to the prompt, you may decide that you want to filter some or all of the prompt out.

A natural thing to where this happens is with the safety infrastructure that you're going to see Sara Bird talk about in her talk later. But there are other reasons that you may want to do some filtering on the responses.

You also have this unit of prompt code called the meta prompt. And the meta prompt is the standing set of instructions that you give to your copilot that get passed down to the model on every turn of conversation, that tells it how to accommodate itself to the copilot that you're trying to build. It's where a bunch of your safety tuning is going to happen. It's where you sort of tell the model what personality you want it to have. For instance, we use the meta prompt to do things telling Bing Chat to be more balanced versus more precise.

It is also how you sort of teach the model new capabilities. You can even think of meta prompt design as a form of finetuning. And so, it's just far easier to do things in the meta prompt than to have to go down to the lower layers of the infrastructure and start rolling your own things.

Once you get past the meta prompt and the prompt filtering stages, you start to think about grounding. And grounding is all about adding additional context to the prompt that may be useful for helping the model respond to the prompt that's flowing down.

In the case of Bing Chat, which I think is the first place that was really doing retrieval augmented generation before retrieval augmented generation had a name, we basically look at the prompt, the user query, and issue a query to the search index to find relevant documents for the prompt. We add those documents to the prompt and send it to the model so that it has that extra context to provide a good answer.

Increasingly, people are using vector databases to do retrieval augmented generation. You may take the prompt, compute a set of embeddings for them, and then do a lookup in a vector database that is indexed by those embeddings to get relevant documents for the prompt, and give that extra context for the model to give you a better answer. But you may also augment the prompts and do grounding with arbitrary Web APIs. And you can even think about using plugins for doing grounding.

And so, the next step here is this is where plugin execution happens. At this stage, again, what I just mentioned in grounding, you may use the plugin to add some extra context to the prompt before it goes down to the model, or you may execute something, do a plugin execution on the way back up from the model, so that you can take an action on a system.

Once you get through all of the stuff in the orchestration layer, and I should say, also, you may be doing multiple turns through this whole system, calling multiple models, making multiple passes through this whole pipeline in order to get what you need from the system.

But at the very bottom of the stack are foundation models and infrastructure, and we give you a bunch of choices for how to use foundation models in this copilot platform on Azure and on Windows. You can choose to use one of the hosted foundation models, like the ChatGPT model or the GPT-4 model that are now available on the Azure OpenAI API Service. You can finetune one of these hosted foundation models. The ChatGPT 3.5 finetuning APIs are live now, and you'll be able to finetune GPT-4 soon.

But if neither of those options work for you, like you have sort of exhausted all of the things that you can do in the orchestration layer to get your copilot to do what you need, and neither of these things will work for you, you can't wholly solve your problem with a hosted API because of whatever reason, you can't use the finetuning APIs to accomplish what you want to accomplish, you can bring your own model.

And we are incredibly excited about what's happening in the open source community right now. Like, there's a bunch of brilliant, brilliant work happening with open source models. And one of the things that you will see in the next talk is we have the Azure AI catalog, model catalog that is going to be a place you can go inside of Azure to find the most popular models on Hugging Face and in GitHub, where you'll be able to push button provision and deploy those models to Azure to use in your copilots.

And also you can train your own model from scratch, as we mentioned several times, from the most ambitious models in the world, the ones that OpenAI are training, all the way down to smaller things, like Azure AI supercomputing infrastructure in our environment give you a great, great way to train your model from scratch, if that's what you need to do.

This is the copilot stack, top to bottom. And what I want to do now is make this maybe a little bit less abstract by talking about a copilot that I wrote.

I host a podcast called Behind The Tech. And every month when the podcast airs, my team comes and bugs me to write a social media post to advertise the podcast. And I suck at this. Like, I forget to read my emails. They have to bug me over and over again, and they really, really want a Kevin social media copilot so they don't have to go through the irritation of dealing with me.

And so, I had the honor recently of interviewing Neil deGrasse Tyson on the podcast. And so, I'm just going to walk you through this copilot that we built, that we actually just ran. And it did the social media post for the Neil deGrasse Tyson podcast that just went live.

Here's what it looks like. Just end-to-end picture, the copilot runs on a Windows PC. It uses a mixture of open source models and hosted models. It does retrieval augmented generation, and it calls a plugin to finish doing its work. Let's walk through these step by step.

The first step of this process is we have an audio file, and we need a transcript. On our Windows PC, we take the OpenAI open source Whisper model and run the audio through the model to get a transcript. It does a really amazing job.

Once we have the transcript, the next stage in the orchestration is we have the Databricks DALL-E 2.0, 12 billion parameter, large language model, running on our Windows PC, and we ask it some things about the transcript, for instance, who was the guest this episode, because again, we want to do this lights out, not have to have Kevin answering a bunch of questions because he's slow and annoying.

The next thing we do, once we have the transcript and we have all of this information that we've extracted from the transcript, we want to send a chunk of that to the Bing API, or we want to send Neil's name to the Bing API to get a bio. And then we're going to combine all of this stuff together into a single packet of information, a big prompt that has some stuff about the transcript, and some stuff about Neil. And we get our social media blurb.

This is a pretty good blurb, so we're going to go to the next step here, which is we need a thumbnail. We call our hosted OpenAI API to get an image from the DALL-E model. This looks pretty good. It's cosmic, it's sort of podcast-y, plenty good enough for this post.

And so, the last step is we want to invoke a plugin for LinkedIn that will take the thumbnail, and the post, and the link to the podcast and just post it to my LinkedIn feed. One of the super-important things about this is before we take an action on the users' behalf, we want to present to them what it is that's going to happen, because if for some reason or another, the model went haywire and produced something that we didn't want to post, once I hit yes, this is going to 800,000 people on LinkedIn.

We review, we click yes, and we post, and this is the live post that's on LinkedIn right now. You should go check out this episode with Neil. It's awesome. (Applause.)

This is really just an illustration for you all. Like, I'm not claiming that this is the most interesting copilot in the world, but it was really pretty easy to do. We posted all of the code on the GitHub repo. Like, I encourage all of you to check it out. It's a good template for thinking about how to build your first copilot.

The thing that we want to talk about last before we jump into Scott's keynote is AI Safety. It's the first thing that we think about when we're building copilots, and we think about it at every step of the process. You're going to hear a ton about this great AI Safety work from my colleague, Sarah Boyd, who runs our Responsible AI Infrastructure team inside of the AI Platform group. It's really, really super good stuff. Like, we're giving you all some amazing tools to go build really safe, responsible AI applications.

Just very quickly, I want to mention one of the things that you're hearing here first, Satya mentioned it, is we're giving you a bunch of like, amazing media provenance tools that will help users understand when they're seeing generated content or not. Like, we're going to be watermarking all of the content that we are producing, and we're giving you tools where if your

AI applications, your copilots are generating synthetic content, you'll be able to call our APIs and add these cryptographic provenance watermarks to your tools. It's super, super exciting stuff.

Copilots, you have heard from us that we have this amazing new software development pattern. You've heard about how we think about architecting copilots, and you have heard our enthusiasm that not only are there going to be a bunch of copilots from Microsoft and from our partners, but we really think that you all are going to be the ones who build the most interesting copilots in the world. It's just like any other major platform. The thing that makes your PC great, the thing that makes the Internet great, the thing that makes a smartphone great aren't the things that launched when those platforms launched. It's what you all will create on top of them.

I want to share one anecdote before we go. I was an intern at Microsoft Research in 2001. I came to MSR with my Ph.D. advisor when he went on sabbatical. And we would go out with our research group every Thursday to this burrito joint in Bellevue that I think is closed now called Acapulco Fresh. And occasionally, this gentleman would join us. His name is Murray Sargent. And Murray, I was a 30-year-old Ph.D. student, seemed like a legend to me because Murray was the guy who had broken the 64k limit on the Intel microprocessors.

Many of you may be too young to even remember this, but at one point in time, the computers that we shipped could only use 64kb of memory for doing the work that they had to do. And Murray was the guy, when the 286 came out, that figured out protected mode, and got Microsoft software to work beyond that 64k memory barrier. And it's unbelievable to sort of think about what impacts small things that had on the trajectory of the industry.

I was in awe of Murray, and I wondered every time we had lunch with him, what am I ever going to do in my career that would allow someone like me, a younger version of myself, to look at me and think, wow, this guy did some legendary stuff. And so, this is the moment for all of us now. We have capabilities in our hands with these new tools in the early days of this new platform to absolutely do amazing things, where literally, the challenge for you all is to go do some legendary shit that someone will be in awe of you for one day. (Applause.)

And so, with that, I would to bring to stage my colleague, Executive Vice President of Cloud and AI, the legend himself, Scott Guthrie. (Applause.)

END