EPISODE 1422

[INTRODUCTION]

[00:00:00] KP: In contrast to other IDs and programming environments, the notebook interface offers software developers a unique environment idealized for data professionals. Despite the growth and popularity, a surprising learning curve exists just to get set up and configuration going. After that, your siloed notebook offers no native collaboration tools. And while I can connect to a SQL database programmatically to get the data I need, if I'm looking for an ideal ergonomic environment for some heavy-duty SQL queries, I have historically sought an external tool for that job.

In this episode I interview Barry McArdle and Caitlin Colgrove from Hex. Hex is a collaborative data workspace that makes it easy to go from idea, to analysis, to sharing. We talk about Hex's offering in the evolving space of notebook solutions for going beyond some of the issues noted above.

[INTERVIEW]

[00:00:55] KP: So, Barry and Caitlin, welcome to Software Engineering Daily.

[00:01:00] CC: Great to be here.

[00:01:02] BM: Thanks for having us.

[00:01:03] KP: And can both of you tell me a little bit about how you got your start in software and data related topics?

[00:01:09] CC: Yeah, totally. So I started as a software engineer, first, at Palantir, where I worked on a variety of different full stack data products. There, I actually also moved into more engineering management side of things. And then I spent a year or so building out the data team at Remix, which is sort of a small transportation planning software startup. And that was really interesting for me, because I got to sort of emerge and see the beginnings of the

development of this modern data stack. And that was where actually where I started talking about Barry about the idea for Hex, even though we've sort of been working together for a few years at that point. And here I am.

[00:01:46] BM: Yeah, and I've spent my career more on the like user of data product side. So I've been either an analyst or sort of more product roles around using and building data tools. And Caitlin and I actually met when we were both working at Palantir. We wound up working closely together to build some data products for customers I was working with and her and her engineering team. And so that was five or six years ago now. We've pretty much been working together and partnering on building data products ever since.

[00:02:13] KP: So both of you have had enough experience that you saw when the idea of notebook starting with the Jupyter Notebook or maybe, I guess, it was Mathematica that had something similar before. Could you talk a little bit about that moment when you saw this paradigm first hit the market, so to speak?

[00:02:29] BM: Yeah. For me, it was probably almost 10 years ago. I, like a lot of people, started using notebooks when I think there were so called – Like, the predominant was IPython notebooks. And that project eventually evolved into Jupyter. As you mentioned, there were other iterations of that. I think most people think of Mathematica is the first one. And like a lot of people, notebooks to me like immediately caught on and clicked. They're very interactive. They're great for exploratory work. It's that sort of REPL, like, literate programming paradigm where I can write some code, see the output, edit the code, see the output. And they're really great for that. But I think, like a lot of people, I kind of always had in the back of my mind a bunch of the problems and limitations with them.

And so I was a user of notebooks for many, many years before the idea for Hex really congealed. And in some ways, what we're doing with Hex is taking the best of notebooks, I think the potential of notebooks and trying to really lean into that while thinking just in a much bigger picture on what these analytics workflows really should look like.

There's another thing about notebooks, too, it's like they've traditionally only really been accessible to a small population of people. It was like, “Can you get a local Python environment

set up? And can you manage all of your environments and dependencies and run Jupyter locally and figure out all the version control stuff yourself?" That can be a pretty high barrier for entry. I struggled with that actually when I was first using them. And so another sort of core, I think, preceptive Hex, is like how do we take this amazing literate programming paradigm and help just a ton more people be able to benefit and access it?

[00:03:59] KP: And can you talk a little bit about what Hex is as a company and what the products or services are?

[00:04:05] BM: So hex is a collaborative data workspace. People use hex to analyze data, collaborate with others, and then share their work as interactive data apps that anyone else can use. And this solves some really core problems around data analytics and data science projects, especially on how teams collaborate and share. This is a problem that I had years of experience with as noted. And then we see all the time today when we're talking to potential users and customers where people are doing really great work. But it's super siloed. Like, they're running a local notebook, or they're running something in a SQL scratchpad or whatever. It's really hard to work together. And a lot of times you'll see collaboration is really suboptimal. And then sharing is done through like taking a screenshot of a chart and pasting it in a Google Doc, or doing a CSV export, or all these sorts of really horrible hacks.

And the sort of workflows and products really just haven't been built around modern collaboration. And so at Hex, we've built a really forward-looking platform that takes the best of these things, and then pushes these workflows forward. And the core of Hex is this notebook like UI. Like if you're coming from Jupyter, it's all very familiar and powerful. But I think part of the magic of Hex is that it then layers on a bunch of awesome new superpowers. And it's also really approachable for folks coming from other backgrounds.

As an example, it has a full-blown sequel IDE, and then a bunch of UI-driven tools, like, visual charts. So you don't even need to know any Python at all, or be able to run anything locally, to be able to do amazing things in Hex. And this has just completely opened up the population of people who can do these types of workflows. This is really amazing as an individual. I can run a SQL query. I can then go do something in Python. Go back to SQL. Build a chart using UI all in one flow. It's really great. But then it's also great for teams. Like, we can work together. I can

version my work. We can do code reviews. I can quickly invite you and ask you to do a review. And you can comment on, I mean, what you're familiar with, if you've ever used like Google Docs. We basically brought that to these analytics workflows.

And then you can take that project and build an app. And this is really great, because you can basically wire in a UI really quickly. You can add like a dropdown, or a slider, format the whole thing in a beautiful way, and then share with others. And instead of them getting like a static export or a code notebook file that they don't understand how to run, other people you're sharing with just see a gorgeous web app that they can use to ask and answer questions.

And this really starts to get into this bigger picture of what we're trying to accomplish at Hex, which is being a platform for knowledge. Data work isn't just about one-off insights. I actually think that's really part of the problem. Like you see a lot of data scientists or data analysts will do some really interesting work and then they'll share it out as a one-off export, or a CSV, or Google sponsored screenshots in a Google doc and it kind of just float some ether.

And so another thing we've introduced into Hex is this idea of a knowledge library. It allows you to categorize and organize your work. You can assign a status like draft, or approved, or archive. So now there's one place you can go in an organization to find all the knowledge that has been created by the data team. There's no more hunting down random links in Slack or whatever. And we think this really fundamentally changes the way data teams and data practitioners interact with the rest of their organization. And so from a mission perspective, I think at the highest level, that's really what we're trying to accomplish in effect. And I'm really excited about what we've done and what's coming on that front.

[00:07:29] KP: Is Hex – Or could we describe it a Jupyter as a service plus all these extra features? Or have you reinvented the notebook in a more fundamental way?

[00:07:38] BM: We owe a debt to Jupyter. We owe a debt to the folks that – Even back to Mathematica, and folks who have iterated on this format over time. But I wouldn't personally describe it that way. I think that we've really taken and added a lot of other things. And we've really taken it in some other directions as well. So, again, if folks are coming from Jupyter, I think if you're like, “Hey, I'm looking for like hosted notebook as a service.” Don't get me wrong. I think

we're a great solution for that. But we've really gone deep on some other areas that I'm excited about. And I think one of them is how we've reimagined the compute model for notebooks, which was traditionally one of the most problematic things.

[00:08:17] CC: Yeah, this is actually one of the things that's been one of the most interesting projects that we've worked on. And fundamentally, architecturally, we did start very similar to Jupyter, where you sort of have a stateful memory bound kernel. But there are a lot of problems with that. Some of the most famous ones come from the talk I Hate Notebooks, which I think is pretty famous at this point. And there's a couple problems with the way notebooks work under the hood. And we sort of think about these as kind of the state problem and the scale problem.

And the state problem is that it's great to be able to rerun individual cells. But you can also do this out of order. You can get into a state where you have no idea what's running under the hood. It's very hard to reproduce. I think everybody who's worked with a notebook has had this experience where they are working in their notebook, they're running different cells, trying to get it to work. And then they run it top to bottom, and it's totally broken. And so this was something that we saw really early on with our users. One of the most common answers to support requests was, "Well, have you tried running at top to bottom?" and to try to see what's going on there.

And while being able to run things out of order and sort of in isolation, it does have some uses. And we actually still support that. Often, it ends up not being the workflow that most people are targeting. And so what we actually did is we actually sort of rethought and reimagined what this state problem should look like. And we ended up building what we refer to as reactivity on top of our notebooks. And what this does is, essentially, what we do is we look at the different cells in the Python that you've written and we automatically determine the dependencies between cells.

And so what this means is that when you change a cell and you rerun it, we automatically know all of the cells that are downstream of that and we run those for you. And so the end result is that your state is always consistent. It's always predictable. And when you run things top to bottom, you're always going to get the same result. And this has just been hugely popular among all of our users, because all of a sudden, they don't have to think about like what are the variables containing in-memory? Or because it's all just what you see is what you get.

It also have some really awesome sort of knock on benefits of performance, for example, because now we can partially recompute notebooks in a really predictable and accurate way. And this was a super fun project, just from an engineering side, because we had to go into the Python and actually, like, look at the abstract syntax tree and start to pull out the different variables and how they're being used and things like that. So in addition to be being a great feature, it was just a really fun project.

[00:10:42] KP: Yeah, that is a really exciting feature for anyone like myself who's had that exact problem of going back and making edits. And now it doesn't run top to bottom. But I also fear that somewhere lurking in there is the halting problem, or some reason why this isn't necessarily going to work in every single case. Can you put my fears to rest in that regard?

[00:11:02] CC: Well, technically, you're correct. It's actually very, very difficult. And I believe that you're right, provably impossible to do this 100% accurately. And I think that dissuaded us for a long time from even trying. But what we found is, in practice, you can do it really, really well. And we've actually not yet encountered a situation where this was actually the bug where we'd sort of inaccurately calculated these dependencies.

[00:11:24] BM: Yeah. I think what's interesting is you can focus a lot on like the edge cases or the theoretics of like these problems. But in practice, like, what we find is the benefit of this feature is actually, such that, even if there's edge cases, or whatever, like users are pretty tolerant of it, because we're giving them all of this other magic. And so every product contains its own tradeoffs. And we just felt like this was just the really obvious thing to do.

[00:11:49] KP: So to date, my experience collaborating with notebooks is either I export an HTML or a PDF of it, or like you say, take a screenshot of the figure. Or I guess I could say I could screen share with somebody. And that's a form of collaboration. On top of these very baseline things, how does Hex helped me have a more sophisticated collaboration?

[00:12:10] BM: Yeah, everyone's had that experience, right? So I think when you think about the sort of software space more broadly, in the last few years, you've seen this really renaissance of collaborative first tools and a lot of other spaces. Figma has done this for design.

You have products like Notion that's done this for sort of doc editing knowledge. There're even things like frame that have done this for video editing. And we're doing this for data and analytics workflows.

And so a pretty early project that we worked on that's also very interesting from an engineering perspective is a real time multiplayer. So just like any of these other products I just mentioned, you and I can be in a project together. I can see what you're doing. We can full multiplayer state comment. So I can ask you to do a review on it. And we can collaborate in that way. And that's useful sometimes synchronously. Like you and I are on maybe a chat together and want to debug something. It's also really useful asynchronously. We see a lot of people use this for code review workflows, basically.

Then we push well beyond that, though, because that's a great feature. But it's important then we've introduced like real version control, like the ability to checkpoint, save, revert, have different versions you're working on, and have all of that be really thorough. And then even sort of what I would call a basic branching model of being able to publish your work and say, “Hey, this is a version that's published,” is actually sort of less. It's almost like deploying it. And so you can have like a draft version and then something that's published. And then beyond that, then syncing to GitHub so you can manage your analytics, like code. Like having a diffable file format. These are all things that sort of – You may even reach for different things in different contexts of collaborating. Like you and I may want to work synchronously in a project to debug something. Or I may want to check my code into GitHub for another reason. So there's just a lot of ways we've sort of enabled these different modalities of collaboration. And then I mentioned earlier, the sharing bit, I think is equally important of like how can I take my work and make it useful for others without like doing a static HTML export or whatever, which is not great. And so that's where like the data apps bit comes in, and like being able to categorize and share those and have people who've never even seen a line of Python before be able to interactively rerun your project and ask answer questions of it. That's crucial if you want your data work to be impactful. And that's been a big focus of ours.

[00:14:32] KP: When I run a Jupyter Notebook, the state that you've been mentioning has things like what the variables are that I've set, and also is something about the particular library instances I have running locally, because of course, that Jupyter Notebook is really from a

server on my local machine. That's not going to work when it comes to collaborating. So in the case of Hex, where does my memory and the choice of libraries and all these configurations, where does that live?

[00:14:57] CC: Actually, this is one of the huge benefits of Hex because of the shared library problem and this shared sort of local state issue. And what libraries you have installed? At what versions? And things like that is a really, really common pain point of making Jupyter Notebooks portable. And so the way we operate is actually we have kernels that we're hosting, and those have a set of packages installed on them. And you can change that if they're not what you want. But this particular configuration is associated with the notebook. So every time I come in, I'm going to spin up a version of this kernel that has that configuration. And so I, as the end user, don't even have to think about that. That version or that set of libraries, all of that state is portable between different users, because we've actually sort of preserved that as part of the application state.

[00:15:45] BM: What's interesting about this, similarly to the reactivity, like the compute model, is this is really useful for advanced users who, like you or me, have that shared experience of like trying to manage environments and deal with all that. But it's also really useful for more novice users who are like, “What's a Python environment?” And that's a lot of our users. These are SQL folks. These are folks who are maybe coming up in data analytics and data science.

And whether it's the compute model, or its environment management, or collaboration, like, we have this philosophy as a product of like a low floor and a high ceiling. Like the entry point to participate in data science shouldn't be like go learn how environment management works, or go learn how to run. It's not right. And this is just like then the legacy of a lot of these tools so far. Like, step one can't be like figure out how to build a Docker image. We can take on a lot of that for you, while also giving those advanced users flexibility and control. Like that's, I think, the real magic of a lot of what we've done and something we've really spent a lot of time focused on, whether it's from the engineering or design perspective.

[00:16:53] KP: I appreciate that you have a Git integration. I like source control for notebooks. But one of the most painful experiences that – Well, not most painful, but a very painful

experience I had in life is when there's a merge conflict on a notebook, because of course, Jupyter won't even recognize the format anymore. Can Hex help me in that situation?

[00:17:12] BM: Yeah. I mean, this is one of those yes earlier about sort of our relationship with Jupyter. And again, I can't say enough about how much we feel indebted to that project and that community. But it's one of those two where you kind of look and you say, “Hey, there's some assumptions there that maybe won't work as well for what we're doing, whether it's the compute model, or files.” As we approach the question of being able to have like different file formats, we had a real – You really had to tackle this, like, are we going to embrace the IPYNB file format? There're some reasons to do it. Like, it's a standard. There's sort of ecosystem around it. You could potentially export from Hex and import directly into a Jupyter Notebook. But there were also some really strong reasons recommending against it. Like, even if you could export from HEX as an IPYNB, there's a bunch of features we built that like don't exist in Jupyter. So that that's not going to work necessarily.

And then more broadly, and I think what you're referencing, there's some real issues with the format. Like it serializes outputs into it. It is actually really difficult to diff. You have a lot of merge conflicts. I don't think there's anyone that I've ever met who's like really enjoyed the idea of doing like Git-based workflows around IPYND, despite there being a little cottage industry of like various projects to try to make it less painful.

So we've developed our own file format. It's not like some super-secret proprietary format. It's basically just YAML. But it's, like, cleaner. We think it's more concise. It's easier to diff. It's something that if someone did want to take it and you push it to another tool, feel free to do it. It's not meant to try and lock-in or anything like that. But we think it's just a better, saner format than what I think a lot of folks wind up grappling with today. And like I mentioned earlier, every product has its own tradeoffs. We just felt like this was the right set of tradeoffs for us to embrace.

[00:18:58] KP: Well, sharing a notebook experience, I have a pretty good vision of that. You'd also touched on some like UI elements and ways you could construct something interactive, which isn't traditionally a notebook idea. Could you describe more of that experience in Hex?

[00:19:12] BM: Yeah, sure. So I can fire up a Hex project. I can write a SQL query. I can write another SQL query on top of the results of that SQL query. A cool feature of data frame SQL. I can write Python to manipulate these data frames. I can go back to SQL. So I can do fully code-based workflow. But then there's a bunch of other cells that we've introduced, like charts cell as an example, where I can build a chart visually, instead of writing a bunch of lines of Matplotlib, or Altair, or whatever my weapon of choice is. I can build a visual style chart cell, which is really appealing to some users.

And then I can also wire in a UI using our input parameter cells. And so it could be something like a text cell, which, as you can imagine, like a little text field. I can enter in some text, and that cell returns a variable with that string. It could be a slider, like a numeric slider, like 1 through 10. And it will return variable from that. There's a whole library of these that we've built that effectively lets you – It's like a UI Builder, really. Like I can go through and wire up in my project these UI-based cells. And that's cool. I use that all the time when I'm doing work just myself of like I'll build, instead of a bunch of hard coded parameters, I'll build these input cells. It's easier to slide them around.

And Caitlin was talking about the reactive model. It'll only recompute downstream dependencies. That's pretty slick. But then these are really great in the interactive apps that Hex lets you publish. When you think about publishing an app in Hex, when you go to the app builder, by default, it kind of takes all of the input parameters and then all of the cell outputs, whether it's a chart, or a table, or whatever, or markdown, formats them really, really nicely, hides all the code, and then lets you publish that. So the end result could be something that is really delightfully interactive for end users. Like I can move a slider and it'll rerun all the logic behind the scenes and serve new results backup to me without me even needing to know what a notebook is, or what Hex is even necessarily, if I'm just someone using this app that's been published.

And so people build all sorts of things with this, whether it's a simple report, or a really complex app, or something that's more like a dashboard. Like, we're effectively allowing people to publish full blown web apps without needing to go learn a bunch of frontend code. It's pretty slick.

[00:21:18] KP: Do you see that being something that people ultimately published? Like maybe someone will build their whole business your platform? Or is this more targeted as the way I share things to executives who don't know SQL, but still need controls?

[00:21:32] BM: Yeah, those mythical low-tech executives. That is how a lot of people use it, right? They'll publish something to stakeholders internally, whether it's a report that's like, “Hey, here's the results of this experiment.” And they want to send out some product managers and marketers. Or something more complex. We have people have build full, like, CRUD workflow apps in Hex, which is pretty cool. And then, yeah, we've seen people build all sorts of stuff and publish them publicly. Like we have folks who have published some really neat – We had someone do this really cool New York Times crossword app that they publish publicly. It's just like all sorts of neat stuff that people build and share. And it's really cool to be a platform for people's creativity. So yeah, most of those use cases of those internal ones. But we've started to see a lot of people just start sharing things publicly. And it's really fun to see that.

[00:22:17] KP: Well, I'm not sure how much access you have to data like this. But I'm curious if you see job titles of your typical users. And if there's any patterns of the type of person that spending some of their working day in Hex?

[00:22:28] BM: Yeah, it's a hard thing to get exact data on, like job titles of users or whatever. It's more anec data just from talking to users. But it's a really cool span that we see. I think, like, you can imagine this sort of core persona that's like a data scientist who has used Jupyter in the past who’s like, immediately, Hex is just sort of a new upgrade and superset of superpowers. But there's this really wide span, we call it this analytically technical population of people that are – They're data-driven. They're curious. They want to do more with data. And this can span from a data scientist, to a data analyst, to machine learning engineer, to a business analyst.

Like, we actually think that one of the big problems with a lot of traditional tools is that they wind up being really siloed, because they're only targeting very thin slices of this. Like, traditionally, if I'm using a notebook, by necessity, I have this like higher degree of technicality. And if I'm someone who's mostly working in SQL, I'm off using something completely different. And if I'm more of a business analyst, I might be in Excel or a BI tool. And if I'm a machine learning

engineer, I'm often some other thing. And it's like you can't be everything to everyone. But we think that fragmentation is actually a bug.

And so I mentioned this earlier, but that sort of low floor, high ceiling philosophy is a way for us to bring all these people together and sort of say, like, “Hey, these are actually one consistent set of workflows that people should be able to collaborate and share on together.” So it's an interesting question, like, what is the title data scientist even mean? And who are these personas? But we think of it like, “Hey, there's this big population of people who are trying to do more with data.” And we want to be something amazing for all of them.

[00:23:59] KP: Correct me if I'm wrong on this, but I'm pretty sure Hex is not a storage solution. That is to say, I'm not going to bring my data to you. I already have my data somewhere. And I bring Hex to my data. Are there any prerequisites for companies who want to onboard?

[00:24:12] CC: So we actually made the product decision very early on that we wanted to integrate really tightly with existing storage solutions? And there's a lot of really good reasons for this. I mean, number one is that Snowflake, Databricks, all of these folks, are basically building supercomputers. And we think that our expertise is really in building this really compelling analytics experience that's an amazing user experience for this population of users. And we want to be able to take advantage of what people have invested years and years in cloud data warehouses.

And I think the other thing that we saw as we were getting started is that, increasingly, there was just a lot of adoption at the storage layer of cloud data warehouses, or things like DVT to transform your data and clean up your data that way. And that we could come in and provide this really strong sort of added value on top of that without needing to kind of reinvent the wheel that's already being done really, really well by these other products.

[00:25:13] KP: And then, what are the support systems? If I have Snowflake, or I'm on AWS, what kinds of things can you connect to?

[00:25:20] CC: I mean, you name it, Snowflake, Databricks. We just did Dremio, and Trino, MySQL, Postgres, S3. I mean, at the end of the day, we’re also – It’s a Python notebook. We’ve

built a lot of data native integrations. But this was pretty core, again, going back to kind of Barry's low floor, high ceiling comment. Ultimately, with Python, we want to make the experience great for the really common path. But with Python, you have access to all of this stuff. Many, many of these systems have great Python bindings, and it makes it a really great choice for sort of a base level technology to build on.

[00:25:55] KP: One of the things I might be doing is building some machine learning models, which at least in my experience, when I'm going to do them locally, not with some distributed tool, I'm going to need a lot of memory, maybe more than I need on the days I'm just doing some SQL queries and things like that. Do I have options to configure the compute that's behind my notebook?

[00:26:14] CC: Short answer is yes. This is something that we've seen a lot of. And I think there was some interesting considerations here, quite frankly, because we are running the compute for you. And so there are implications around just giving you more and more memory that's pretty expensive on the underlying cloud infrastructure. But yeah, this was something that we heard a lot from users. And so as we've been sort of thinking through, we were talking about customizable libraries and environments earlier in the show. We also have been able to do this with resources. So with increased memory, if that's something that you need, increased CPU, even GPUs, things like that. That's something that we've been able to provide for the users who want to do more advanced things.

[00:26:55] BM: I think there's also cool opportunities to integrate with a lot of things in the ecosystem here. There're people building some really interesting distributed systems. So you were asking earlier about storage. And it's like, "Well, we're not going to build our own data warehouse. There's people doing a very good job of that." We'll let you scale up resources that we can provide, but also, like, integrate closely with – If you're running other types of distributed infrastructure, making that a first-class experience as well.

[00:27:20] KP: You'd mentioned SQL ID as being one aspect of Hex. That's not traditionally something that's in the notebook experience. Although, I mean, it's Python, of course. So I can figure out my connector and connect to my database. What enhancements or easier paths to success are available in Hex for me?

[00:27:38] BM: Yeah. I mean, I think this is an interesting thing, just to back up for a moment. Like, if you think of the world as like Python notebook, then like, yeah, there's all sorts of things that aren't in there. And as we think of the bigger opportunity around these workflows, we've always thought of Hex as the sort of workspace of like, “Hey, it's not just about being a Python notebook. It's about this whole range of these analytics workflows.”

And so first-class SQL support is always a very obvious thing to do for us. And we've built something, I think, that's pretty compelling. So I can establish a data connection, whether I'm using Trino, or Snowflake, or Redshift, or whatever I'm using. I can browse my schemas so I can see all the underlying data structures. If I'm using something like DBT, or another data catalog, I can enrich that schema with information about metadata, descriptions, or tests. And then I can write SQL queries in a SQL cell in Hex that returns a data frame.

And the SQL experience is great. We have schema type ahead, code formatting, previews, all the things you'd expect from like a modern SQL editor. And I can actually just go wall to wall building a sequel notebook. I don't have to write a single line of Python. And so that's, again, why it's like is it a Python notebook? Well, no, it's this bigger thing. And so we introduced this feature called Data Frame SQL, which is pretty sweet, which is like I can write an initial query against whatever database I'm using. And then I can write another query on that data frame result. We kind of treat every data frame in Hex as part of this sort of meta schema, like meta database that I can query. And that's true even for a data frame that might be returned by a Python cell.

So if I'm running a Python cell, it’s transforming a data frame in Pandas, or maybe it's hitting an API and pulling a data frame back, I can query that data frame and transform it using SQL using our data frame SQL feature. And so we have a lot of users, actually a ton of users that just use Hex like an end-to-end SQL experience, because it's bringing that literate programming notebook style experience to this bigger population of users. That is really, really cool. And again, I think it's very core to our mission. We don't see that as like a bolt-on. It's like, “Yes, this is the point. Bringing more people in and allowing them to engage in data analytics workflows and data science workflows. Like, that's the mission.”

[00:29:54] KP: Another feature that is sort of set outside the traditional scope of what a notebook is, is scheduling. I've often been in this case where I have a great notebook and I'd like to have it just run every day or on some frequency like that. Can Hex help me with scheduling?

[00:30:09] BM: Yeah, we have built-in scheduling. You can set it up to run hourly, daily, weekly. And if I've published my work as a data app, the outputs, the cached outputs of that app actually will refresh on that schedule. So we see a lot of people do this, like they'll have a project in Hex, and maybe it takes like 15 minutes to run end-to-end, because it's doing a bunch of data processing. Or if the query is expensive, they'll set that up to run like daily, or hourly, or whatever. And then the end users who are going in to look at the results, they're seeing the sort of cached outputs from that last scheduled run versus having to run it all like ad hoc, which is grateful both for you don't want to sit and stare at the thing while it's running. It's also great for costs. Like, you're not shredding a bunch of queries against your underlying database all the time.

So, yeah, the scheduling features are pretty popular. And we also have the ability to automatically send emails. If your scheduled run fails, like it'll email you, like, "Hey, there's an error." But also, if the run succeeds, you can also set it up to email groups of people with a link to the project. So this is something we'll see people do. Like every week, on Monday morning, we run the project and email out to a group of stakeholders, like to the exact sort of the product manager for this thing, a link to the project with the results. And that's great just in terms of that shareability and being able to consume the work.

[00:31:28] KP: So I think, earlier, you described Hex is a collaborative data space. And we've talked through mostly kind of the analytical environment, or what I would see data scientist doing. I'm curious now on the data engineering side, do you see this as an ETL tool as well?

[00:31:43] BM: When you build something that has a lot of flexibility, you'll see people use it for all sorts of stuff. We did not set out to build an ETL tool. There's a lot of great ETL tools I can recommend people. Our friends at DBT Labs build a superlative experience around transforming data in your warehouse. Although it's funny, we'll look at some things that people built in Hex, and they're like, "Yeah, I'm scheduling this to run every hour." And you're looking at

what it's doing. And it's like basically an ETL pipeline. And it's cool to build something that gives people that degree of flexibility. I think, if people want to build that in Hex, like, go for it. But I don't think that's the main focus.

What we do see people building, actually like the more like on pieced workflow would be if I'm a data engineer, I'm an analytics engineer, that a lot of my work is developing ETL pipelines, I might use Hex to iterate on the logic for it. Like that notebook style literate, REPL style thing is actually great. I can say, “Oh, I can write the SQL, look at the result, iterate, iterate, iterate. Great. This looks good.” And then maybe I'm taking that logic and then pushing it down to a DBT model or to some scripts I'm scheduling through Airflow or something like that. Like, that's a totally valid workflow. And that's great. And if you want to schedule it to run from Hex, like, go for it. I'm not going to stop you. But it's just not a primary focus of what we're trying to offer in terms of like being an ETL platform or whatever.

[00:33:02] KP: Well, from startups and small enterprise, all the way through medium and large corporate enterprise situations, there's different needs within companies for how they want to deploy or use different technologies. Do you see any patterns in adoption? Are startups being especially successful because they're nimble? Or do some of your compliance and features like that make you really more enterprise-ready?

[00:33:24] BM: Yeah. So I think, like, in tech-first companies that are built around data, like this sort of if you've been founded in the last 10 years, I think you've kind of been born into a time as a company of like understanding that data is really important. And these are also companies that are typically adopting tools like cloud data warehouses and all the other sort of modern data stack things that go with that. So those companies like immediately get a great lift out of Hex, and they typically have data teams.

But we have a lot of experience in our past lives in bigger enterprise as well that a lot of those are or are trying to be much more data-driven. And as we've matured the product, we've seen a bunch of uptick of market. And I think that's where a lot of security investments have really mattered.

[00:34:07] CC: Yeah. I think this was something that we knew we were going to have to do from the very beginning. We actually started building out our security infrastructure as some of our earliest set of features. And that was actually really interesting for us, because we've found some early customers who were actually very security conscious. And we went and we iterated directly with them on all of our infrastructure and our Terraform and making sure that were in a place that was just having a really good strong security stance from nearly day one. It was like nearly one of like our earliest features. We actually got SOC 2 way earlier than anybody recommended to me getting SOC 2. And the reason we did this is because we knew that companies that have a lot of data, they care about their data, and they care about what you're doing with it and how you treat it. And this has actually been enormously successful.

We've actually worked with a fair amount of companies with really quite sensitive data, and they've all been really, really impressed, I guess, with our maturity compared to the age of our company. I do think a lot of that comes from our experience working with this type of data earlier in our careers and just knowing immediately upfront that that was going to be critical.

[00:35:16] BM: I think most companies, in my experience, will view security stuff as like a chore or a box checking exercise. And one of the things Caitlin and the team have done since very early on is, I think, we've seen it as an opportunity. Like, being front-footed about security. Having an awesome security stance. Taking it seriously. When a customer asks to have a security conversation, instead of being like, "Oh, gosh." It'd be like, "Great. Let's lean into that," because we think we do a great job on that." That's been super meaningful for our business. And that will just be more true over time.

[00:35:47] KP: This space of tools that data scientists and analysts and the spectrum of people who work in this area have access to is evolving. I think that's probably going to evolve the way people do their jobs and the way teams operate. When you think a few years into the future, do you have a vision for how that team and the professional data person evolves?

[00:36:07] BM: Yeah. I think, historically, you've seen this dichotomy between like people who are data people and people who aren't. Like, I've got data in my job title. I'm a data scientist. I'm a data analyst. I'm a data engineer. I do data things. And other people do not, or they have less access to these things. And I think that's changing. And I think one thing we see this bigger

population of people that are sort of data literate and technically literate. And that doesn't mean everyone's going to go learn Python tomorrow. But you do see this bigger population of people who know SQL, or they are doing really complex things in Excel, which is basically code. Or maybe they have learned some Python. You see a lot of people coming out of undergraduate or graduate programs who have taken some data science classes. And these are all people who are data literate, data curious. They're they want to ask and answer questions with data. And we want to be a platform that allows this bigger population of people to do this. And that's where I really think this is going.

There's going to be a change over the next few years of these workflows not just being limited to the people with data and their job title. Like, everyone's going to become a data person. And so I think you'll always have data teams. There're always things that you're going to want real specialists to do. But allowing more people to ask and answer these questions is something we're really focused on and that we want to help advance. And I think this opportunity is also really coming about because of this sort of “modern data stack” that's emerged.

You have technologies like Snowflake, and Fivetran, and DBT that have made it really easy to get a bunch of data in your data warehouse, clean it up into usable, reliable tables, and make it accessible to the organization. That was not true a few years ago. Like a few years ago, it was actually a real ordeal to try to get like your hands on, “Hey, give me all the orders data for our business or whatever.” That was that was a big – You'd go ask someone for that. And now, like, “Hey, it lives in Snowflake.” And I think Hex can be a wonderful front end to allowing people to go do more with that data. And we're really excited about that.

[00:38:05] KP: Can you describe your release process or how you go from ideation to something getting into the final product?

[00:38:11] CC: Yeah, this has been really important to us from the very beginning. And we sort of designed our engineering team around two things that are often seen as intention with each other. One is shipping a high-quality product. And the other is doing it quickly. And I think we've built out a really great system for this that doesn't require us to work 80-hour weeks or whatever, but also sort of maintain this high-cadence of release, and also just build things that are a really great user experience.

And the way that I sort of think about how you'll accomplish this is by having and enabling a really high-degree of parallelism on your engineering team. And the big thing that we've done here is that our ICs, so our engineers and our designers, they really own not just sort of implementation and projects, but problems end-to-end. We're not going to give you a whole list of JIRA tickets or really long spec and then spend weeks iterating on that and then have you go just like implement it line by line. What we do is we give our folks just an important problem that we know we need to solve. And their responsibility is to come up with a solution and validate and deliver it.

And, as a manager, as an engineering manager, I'm here to help, provide feedback, get you resources if you need them. But ultimately, we leave it up to the ICs to kind of drive those things end-to-end. So what this does is, because we have a bunch of these things kind of running in parallel at any given time, we're constantly shipping, but we don't necessarily need to sacrifice that extra bit of quality or that like one or two days trying to get something out the door. We can actually take that time to invest in that.

And I think there's also another nuance here, which is that in order to enable this, the other thing that we've done really well is to make some pretty smart technical investments along the way that are not just based around features. So I think, for an example, one of the very first things that we built was actually a complete CI/CD pipeline. And that might seem like overkill before you even have users. But our ability to ship new updates multiple times a day without a lot of process or ceremony has dramatically accelerated our ability to learn and iterate on features. And not to mention, just like the sheer number of engineering hours that we've saved over that period.

And we also sort of think pretty critically upfront about technical debt. And I actually don't even like to call it technical debt. I sort of think about this as investments in future velocity, and really framing them around what is the return on that investment going to be. And I found that that's actually a really powerful framework for helping to prioritize them against feature work, which sometimes you can kind of get in this cycle where the feature work always feels more important, and then you get stuck on sort of the technical debt. And doing that has allowed us to stay

ahead of this on both the sort of on the developer experience and a lot of the architecture. And that's also, I think, really contributed to our ability to deliver quickly and at a high-quality bar.

[00:41:02] KP: And when you're looking on the roadmap, are there any upcoming features, or maybe recent releases you want to highlight?

[00:41:08] BM: Yeah, we've built a lot. We recently released the Knowledge Library, which I mentioned a little bit earlier. I think it's sort of how do the things that data teams are building and publishing get widely disseminated and discovered by an organization? How do you go from building one-off analyses to building true knowledge? That's a theme that we're really excited about.

We recently released the reactive compute model. That was last year. That was something that's really, really changed the way people think about these workflows. And it's been pretty amazing to see that take hold. And then the last one, we actually just released yesterday, I think, was a new app builder. This is kind of a fun one. But, like, we had looked around at when you're building a UI, the app UI in Hex, we've previously offered a couple different modes, story in canvas mode. We looked around at a bunch of different library – These weren't ideal. So we looked around at a bunch of different libraries for sort of drag and drop elements arrangement UIs. And none of them quite fit what we're looking for. So this is something we love doing, which is sort of been, "All right, let's go build something ourselves," and dug in and built a really amazing new app builder UI where I can drag input parameters, or outputs, or headers, or whatever and arrange them. It's really, really slick. The result of like two months of really intensive work, but it came out amazingly. And I think, huge kudos to our engineering and design team who just really built an amazing experience there.

[00:42:36] CC: Yeah, totally. I think the thing that – I mentioned this a little bit earlier. But one of the things that I'm also really excited about, when we talked about sort of notebooks and how they work and what we kind of see some of the technological barriers there, we talked a lot about state with reactivity. But we didn't talk so much about the scale side of things. And you mentioned, "Hey, can I scale up my resources and get more memory and things being very memory intensive?" And like, "Yes, we can do that." And there's times where that's appropriate. And then there's times where that's just impractical. And we're moving into this era where

everyone has a cloud data warehouse, and things are operating at cloud data warehouse scale. So one thing that I'm really excited about is kind of exploring that and exploring how we can really empower users to use the same kind of Hex experience that that we've built, but it gets these like massive scales of data.

[00:43:24] KP: And where can listeners learn more about Hex online?

[00:43:28] BM: Go to our website, hex.tech. We also have a Twitter handle that's hard to spell over audio. So you can just go to our website and check things out there. And then if you're the type of person who loves working on interesting technical problems or interesting technologies like this, we'd love to hear from you. Check out our jobs page. We're hiring for engineers, and designers, and data nerds up and down the stack. And we have a very ambitious roadmap full of big ideas and little details. And we love bringing people aboard who get excited about that. So we'd love to hear from you if you fall into that category.

[00:44:06] KP: Well, Barry and Caitlin, thank you both so much for coming on Software Engineering Daily.

[00:44:11] CC: Thanks. This has been great.

[00:44:12] BM: It was our pleasure. Thanks for having us.

[END]