A Future Date | (Retrofitting Captions Compliance) - Diamond - Closed Captions at Pluto TV - CVAA case study.mp4

JOE DEVON: Thank you for joining us on this presentation of closed captions at Pluto TV, which is a CVA case study. The CVA is the 21st century Telecommunications and Video Accessibility Act. My name is Joe Devon. And back in 2008, *American Idol* changed my life.

I know what you're thinking, but no, it's not because I'm a singer. You definitely don't want to hear that. But I was actually hired to help rebuild AmericanIdol.com. They used to joke, and it was true, that more people voted for *American Idol* than for the President of the United States.

And that kind of came back because the Iowa caucuses had a bit of a problem with their voting app, which led Seth Meyers to quip, why did *American Idol* have a better system than actual America? Which we took as a source of pride, because I was internal there, so were my co-workers. And we really put a lot of effort into architecting that system.

But then back in 2012, with the team that worked on *American Idol,* I co-founded Diamond, and that has really been our focus is building scalable software, a lot of video work for media companies that are inclusive. When we launched back in 2012, we started working on a lot of the same projects, like *American Idol, X Factor.*

We also did Simpsons World, got some other business from NFL, Disney, and other media companies. Here's an example of some of our clients. So around the same time, I also co-founded Global Accessibility Awareness Day. And this is celebrated every year the third Thursday of May. And my co-founder in that was Jennison Asuncion, which a lot of folks in the accessibility community know.

And it really took off to the point that it achieved a Twitter reach of almost 200 million users last May. And it took a while. As a company, we had accessibility as a core value. However, we didn't feel like we had the chops right away. We really wanted to make sure that we had good enough chops and that we brought something unique to the table that other providers didn't have in order to create an accessibility practice area.

And we were very happy that last year we were able to launch this. And our thesis, really, is that there are the ability to do audits and things like that. But what was really needed is to have developers with chops in accessibility that build product accessibly from day one or, if needed, to remediate it. And one of the first projects we got was actually Pluto, which I will allow my

team to talk about themselves. So with that, let me pass this along to James.

JAMES KENNELLY: Hi, my name's James Kennelly, and I'm a project manager here at Diamond. So we were tasked with the exciting but difficult project to audit and remediate accessibility issues with Pluto's closed captions. Like most projects, we were contacted with a problem statement.

Pluto came to us and said, we would like to have accessible closed captions across all of our platforms that are currently in production. That's a pretty big ask. So first we needed to identify what did that actually mean. What platforms did we have to remediate and audit? Well, at first glance Pluto comes across as a web player, something that you can find on your iPhone or Android mobile.

But it extends far beyond that. In total we found there were 52 platforms that we needed to audit and remediate. And this is a significant amount of work to actually go over and identify what the best pathway would be going forward. So that came to our planning phase.

Planning for this project is a fundamental practice in project management so that we were able to identify a blueprint that allowed us to accomplish our goals within a set time. And being able to accomplish those goals within a set time, we also had to find the right resources. Now, when we think about 52 platforms in a specific time frame, how would we get the resources to be able to build and remediate a lot of those issues? Well, that particular practice was led by the team lead engineer, the partner, and myself, the project manager, to assemble a team that was strong enough and understood the practices of accessibility to achieve the goals that were set in place.

Now, all of this is fantastic to look at. But the real question here is the three main constraints of a project. That is time management, cost, and quality. To compress something in a very short amount of time, in two months, both quality may come down and cost may increase. So we had to find a combination of these three particular components that Pluto were happy with, so that we could deliver a product that was within budget, that was delivered on time, and that had the highest quality achievable within the given frame.

So we came to the conclusion of being able to build a team of around 10 staff that were to audit and remediate all of the work within the two-month given time frame for the project. So we come full circle, and we look retrospectively on this project. And we want to identify, is this project successful?

What are the success criteria that enables us to say, yes, this was a successful project? Well, we can look at the cost and that we're under budget. We can look at the time that we delivered the product within one day of the deadline. And we can look at the quality. We can say that we had identified near to 100% of the closed captions' accessibility issues across the project.

Now, is that a successful project? Yeah. Typically and historically in project management, you can classify those things as successful. But what is successful to us? Successful to us is being able to teach and mentor the clients that we have to be more conscious about accessible practices and accessibility in their product in the future.

So has Pluto adopted accessible in the workplace? Well, yes, they have. With our implementation of accessibility practices, audits, and awareness, they have changed their perception of accessibility and been able to take a step forward to a more successful world, particularly with the products that they are developing.

Now, what is success to us? Success is not only the management of time, cost, and quality, but also a cultural change in a company to ensure that they take the benefits of the lessons that are being learned in the development and audit cycle of these accessibility practices. It's important for us to understand the objective is not just to remediate accessibility issues but to promote accessibility so that we won't have these issues in the future.

Now we can look at Pluto as a fantastic example that shows us a company that is willing to take on an external challenge, a challenge that is unknown to them, to be able to bring a vendor like us to increase that awareness, to solve the problems that they have, and to be able to shift the cultural change in their company for the better.

NICK SCHILLING: Hi, my name is Nick Schilling. I'm an engineering director here at Diamond. I was the one that led the accessibility audit and remediation work for this project. I worked with our dev teams and our QA teams to identify the issues that needed to be resolved and try to bring these apps into the world of accessibility.

So which apps am I talking about? Our client was Pluto TV. Now, if you haven't heard of them, they're a free video streaming platform. You can go into their app, select one of the channels that is streaming a particular piece of content, and you can start watching it. It takes a lot of the guesswork out of what you want to watch because, if you know what the category of the content is you want to see, you can go right there and see it.

Now, this app does fall into FCC guidelines. Even though it's a channel that's broadcasted over the internet, all this content has still been captioned previously. So those captions still need to be available for users so that they can watch the content, listen to the content, they can ingest this content.

So this is amazing. We're so happy to be working on this project. The client Pluto, they have a new app that they've been working on. Now, the FCC's guidelines kick in at a certain date. We need to be working on their legacy apps, which will still be alive by the time that that date rolls around.

So there's a hard deadline from the FCC to be CVAA-compliant by January 1, 2020. Because the new app isn't going to be ready by then, we need to make sure that their legacy apps have been updated so that they're fully compliant with caption requirements from the FCC.

Remember, we're not actually replacing these apps. That's what the client is doing. But we are fixing them up. Now, keep this in mind when you're working with a client. You might not be able to replace an app for every single device. Some of those devices are incompatible with the technology that the client wants to adapt.

You might find yourself in this situation, you might not. It really depends on what your business strategy is. But I think as we get farther and farther into understanding and accepting accessibility in this culture, we're going to need to have people who are changing apps and doing these accessibility audits. So this might not be applicable right away. But this might be applicable down the line.

Also for clients, sometimes they might not be able to afford a whole replacement. Doing a top down fix might actually be a lot more cost-effective than doing a bottom up rebuild. In order to become compliant, you have to alter those legacy apps so that those are all brought into the 21st century.

We want to capture all of the requirements early on so we can be confident what we're delivering so that we know that they need what we're delivering. We had in-person meetings to get these requirements. If you can do these-- right now, we have to do them online, but in the future-- I highly recommend doing these in-person.

It's a lot easier to slow the pace down a bit so that you can get yourself into the mindset of the client. What are they looking for? What's their baseline for their legacy apps? Once you know

what that is, you can start to build a solution, a map forward so you can get to the point where these apps are fully caption compliant.

I'm not sure if you've seen the CVAA guidelines for closed captions and for video stream platforms. Let's just say, they're very dense. We can't just copy-paste these documents right into Jira. No one would really understand and it wouldn't really be actionable. Those requirements are built for lawyers to be able to look at something and say definitively, yes, this is complaint. No, this is not complaint.

But these aren't action items. In order for a developer or a tester to be able to act on these requirements, we need to convert them into a testing strategy so that a tester can look at something that's been built, whether it's the original app or the app that's been fixed by developers, and say yes, this is now compliant.

We needed to start coming up with a testing strategy. So first step is to find out the full matrix of devices and platforms and code bases. So devices would be anything from a computer to a phone to a TV. Platforms might be individual variants on those devices.

And then the code bases are what populate those devices with those applications. So first, we have browsers. This is our wheelhouse. At Diamond, we are very capable of building out websites. So this is something that is a known quantity for us. But we have to test every website with the video playback on every major browser, which I've listed here, including Internet Explorer.

Alongside this, like I said, are also set-top boxes, things that you can plug into your television. So you have the Roku, Apple TV, you have the PlayStation, you have a Amazon Firestick. There's a lot more devices. I didn't want to put them all in my list. But as you can imagine, if you were to go to a tech store and start cruising around, there are a lot of options.

Speaking of a lot of options, we also have smart TVs and phones. There are a lot of different phones. There are a lot of different smart TVs. Thankfully for phones, you really just have to look at iPad, iPhone, some Android phone, and then some Android tablet, because these device makers at least understand, well, there's going to be a lot of variance here, let's just try to make things simple for people because developers aren't able to buy 40 phones.

And then for TVs, we have a Samsung, we have a Roku TV, which has the Roku built into it. There is a lot of complexity that comes with testing and developing on a smart TV because a

lot of testing and development has to happen on the actual physical device. Now, that might not seem like a big deal. But you think about how big TVs are, and you think about how many TVs there are out there, that is quite a bit of work.

In order to start, now that we know what we're doing, we have to build out the right team-- development team who has experience across all these different platforms, or at the very least, who can learn quickly how to work in these platforms. Some of these tech platforms that we had to touch, there was no way that any of us could have worked in them, especially the smart TVs. Our testing team also had to be available locally to build a test on those physical devices. Our devs would be working in the client's code bases to fix issues that were found during that audit by our QA team.

I want to really drive home that if you have a daunting-feeling project, the last thing you want to do is try something new and try to reinvent the wheel. If you have a system that's in place for capturing work for current capturing tests, you should just use that methodology.

There is no reason to throw something away to work faster because if you throw something away that everyone's used to, and then you start something brand new, your team's not going to be used to that. And so there's going to be a learning curve. There's going to be some confusion. So sticking with what you know, sticking with what the team knows is a good way to make sure that you can actually get the work done.

We couldn't be at the Pluto offices every day. They have beautiful offices very far away from my house, very far away from most of our homes. So we did as much as we could at home, and we did as much as we could in our offices.

Some of these devices were very easy to get our hands on. We could test on our phones, we could test on our gaming consoles. But there's some specialized devices that are very hard to get a handle on, like the 2015 Samsung Orsay television. If you hear that, do you know where to get one? We didn't either.

It's critical that all of our testers were on the same page about what the accessibility requirements actually were. The standardized test tickets that we wrote-- which I covered in a previous slide-- went a long way here because we were able to let the testers know what they needed to look out for, what they were actually trying to review. Now, this is different than everybody understanding what accessibility requirements are because you can understand what the CVAA requirements are, but we need to also understand how they apply to the actual

project.

Once testing was completed, we wanted to make sure that we had a pile of actionable tickets. Devs shouldn't be blocked when they're starting out work. So the tickets that are created have to include everything that they need to know in order to start their work. In order to facilitate this, our tickets included screenshots and videos capturing all the test cases, so developers know what they need to fix.

Now that we have a set of actionable tickets, what do you do now? Where do you do this work? Because you can know what apps are available, but how do you know what the code bases are? But the tricky part of the legacy apps is that these are applications that are built by previous development teams.

Some of this knowledge might be lost over time, because these old dev teams might not be around anymore, or some might not remember. I sometimes can't remember what I did last week. I'm not going to expect someone to remember what they typed two years ago. So this stuff sort of required some digging, trying to find, where are these code bases? Where do we make these changes?

Developers who are working across these separate platforms, we had to make sure that they were all still on the same page. Like I said with our testers, someone might understand the goal, but we want to make sure that the process is smooth enough that everybody is working together, even though everyone's sort of working on their own.

We had to make sure the process was clearly communicated to helpers. They had to be sharing tips and successes. We don't want the day-to-day to feel like everyone's in isolation because if you're working on all these separate code bases, chances are you're not going to have a full development team, you're going to have individual developers doing their work.

And finally, to consider accessibility decisions needed to be seen cross-platform. So if we made a decision about how menus should be handled or how a guide should be displayed or hidden, those decisions ought to be cross-platform so that if a user opens up the app on iOS, it should really have the same accessibility considerations it does on Android.

Since we're delivering code to the client developers, we wanted to keep them in the loop when we're committing our code changes. So we included them on all of our code reviews when requested so that we could make sure that their coding practices are followed. Because of the

tight deadline, the last thing we wanted to do is dump a bunch of changes in their laps and then have them come back and say, we actually did it this other way, and we want you guys to change it. So we kept on top of that by keeping in constant communication with them. When you're working with a client like that, who has an external development team, make sure to keep in communication with them so that nothing gets lost.

While we were making these changes, we did hand off code. And some of those changes that we made, we thought, oh, these are going to be good for accessibility. But they broke expected user flows. There was a specific example that I can think of where we wanted to be able to turn on closed captioning right after opening up a video feed off of a TV app. Now, this updated flow that we made seemed like a great idea, but the product team flagged it once we made these changes because the expected user flow was completely different than what we were expecting.

If you're doing this on your projects, what you want to keep in mind is that the client needs to be kept in the loop if you're going to be making breaking changes. A lot of accessibility changes aren't breaking-- adding audio tags, adding closed captions, making sure that buttons are accessible. But if you need to change the way that the user flows through the application, the last thing you want to do is make those changes in a bubble because you're eventually going to be delivering those changes back to the client. So if you bring up those breaking changes early, as early as you can, you will save yourself a big headache in the long run.

I want to tell you guys a story about the strangest case, the case of the disappearing captions. We would have captions disappearing about 10 minutes after playback started-- very, very strange. Definitely didn't want that. It was actually an issue that was blocking an app from being approved by one of these platforms.

What would be causing this? Well, if you thought that only one issue would be causing this, you're close. There was actually at least three different issues causing this problem. The first issue was that legacy code was removing captions after a while-- I believe it was after the ad break ended-- to display a "captions coming soon" block.

Having the problem only appear after commercials makes it very difficult to test, which makes it very difficult to track down. This is a case where I don't think we really could have prepared for something like that because you wouldn't expect for the code to be actively removing captions from the view.

The second issue that we ran into was that sometimes a server might return one of the captions files as a what they call a 404. This just means that the file is not available from the server. If there's a lot of traffic or if the user's connection is spotty on their device, or maybe at their house even, some of these files might not download.

So that's why you see buffering issues on videos. The problem was that when the code was encountering those errors, it thought that, oh, captions are gone. Might as well delete captions forever. So we had to sort of add some structure around this so that there was actually some real logical error handling of these captions. The reason why these errors were in place in the first place was that captions weren't being rendered on all these platforms. And so once we added them in, these problems started to crop up.

The third problem we ran into I think was the most surprising one because a lot of the apps were using what is called the HLS video player. This is a fairly popular web video streaming platform. A lot of devices use it, a lot of websites use it. But it has some strange bugs in it. There were some buffering issues, which means that the data becomes out of sync if too much data is taken in. It was causing captions to be downloaded and then not be displayed because playback had already gotten past that point.

Captions would come in, but they'd come in half a second too late, or a tenth of a second too late, and it wouldn't trigger. Memory issues were causing the player to crash if too many captions had been downloaded. There was no garbage collection, is what they call it, where old files are removed because new files are coming in. This player was just holding onto everything, which doesn't feel like a problem until you're testing on an older device, and you're testing for half an hour.

We investigated these issues, and the client had their dev team check into this. And they provided us a fixed player that resolved all these problems. So we're in the final stretch now. We're preparing for release. Now, these player fixes, because they required, essentially what they call waterfall flow for development, where one team works and does their part of the task, and then they hand off the work to the next part of it to be completed, so on and so forth.

By the time that we receive the actual fixed player, we were about two weeks out from launch. So we were integrating these changes sort of at the last second. And there were some integration issues that we ran into. On one TV, we saw everything working fine until we did a final integration with the player and some of our shared libraries. We ran into some issues.

So that's expected, I would say, because if you have a lot of devices and you have a lot of shared libraries, it's going to be difficult for you to test every single iteration of all of those devices because the shared library, the common code that everyone pulls from, it might be updated for some app flows, but because the developer teams are working independently, not all of them might know that that shared code had been updated.

Another issue we ran into-- this is a very small one. On the Chromecast device, there were old changes that were left up from previous work that had never been launched. And we didn't know that. So we were building off of those old changes. Now, those old changes actually had changes to the player, so they weren't changes that could just be rolled back. But these were never launched up to production. It was completely unexpected, but we had some conversations with the client, and they were fine with that. They gave us the go ahead after we sort of walked them through what the impact was that we were perceiving from these changes.

What I want to underline here is how important it is to have conversations with the client. You want to provide them with enough information to make an informed decision, while also providing your suggestions based on your research. We don't want to just give them the full decision, where they don't know what the consequences are, because you are the one that's just working in the code.

Finally, we packaged up our work. We sent it to the client. That feels great. This is a huge win. We feel very happy with the work that we've done for the client and the work that we put into these apps. And captions are working as expected. So I will hand it off to Michael now.

MICHAEL MISTAK: Hello, my name is Michael Mistak. I'm an accessibility manager here at Diamond. And what we want to talk about today, specifically, is what makes up a logically planned plan of attack for retrofitting in captions. What are some of the tasks that, by knowing the full scope of the project and knowing what tasks we're going to need to challenge our team with, what are those items that, outside of the technical challenge, are also going to require a decent amount of deliberation and it would behoove us to get those types of decisions to the appropriate decision makers ahead of time so that, as we're executing against those tasks that are more clear cut, these types of problems have the opportunity to gestate to get to a state of completion before critical portions of the project will work and I need to start implementing decisions those folks are making?

And some of the things that we learned from this project and others are just some examples

that we came across and I think can serve as learning opportunities for you to, again, take a look at projects ahead of you and being able to identify these tasks and, again, get them to the decision makers before you start hitting critical time crunches.

Examples that we're going to highlight on are looking at remote control interface conventions, looking at screen real estate concerns, and where adding in functionality after the fact might give you some challenges there, but also looking at some of the logistics around standardizing caption standards. And then kind of finally, and I think in conclusion very purposefully, is taking a look at how you can, for your own projects and with your clients, set up and follow a decision-making framework that's demonstrative of maturity of organization, maturity of accessibility knowledge generally, and captioning standards specifically.

Now, in a project such as this and supporting the digital ecosystem that we all see out in the world, we know that it's made up of many different types of devices. Those devices themselves either have their own types of control conventions or have remotes or other types of control devices that, in and of themselves, have their own different types of design and control eccentricities.

It's important to really kind of understand what are the standards that exist within these types of devices ahead of time and how are we going to apply similar controls for functionalities within our application. And again, since we are, in fact, retrofitting, are there going to be times when some of those existing patterns or the patterns, let's say, that we would trend towards naturally in order to do a specific function-- for example, toggling on captions, toggling off captions-- are some of the standard ways in which those are done already taken by other functionality within our application? Do we have to take opportunities to maybe move some of that functionality, or do we have to come up with intuitive ways of accomplishing those same tasks but within the framework of our application?

Again, retrofitting these functions in later, we might find a opportunity where we're going to need to negotiate the different landscape of controls that are available within the devices and, more than likely, we're going to need to engage some of the decision makers for that project near the beginning, if possible. Again, so these decisions can be made while we're doing other critical work. And again, when it comes time to implement these types of functionalities, we'll be ready, we'll have the decisions made, and we'll be able to proceed.

As I stated in my introduction, another area that we'd likely have to make some important

deliberations around when you find yourself in a position where you're going to be retrofitting in captions specifically, but in a lot of instances of functionality in general, are how space is being used on the screen. As much as it's generally true that within the digital realm you're dealing with fairly infinite real estate, when you're dealing with applications where there's not scrolling or whatnot, there actually is a fairly finite amount of real estate that we're dealing with.

Components, such as a video player, designed previously without accommodations in mind, may be, in fact, using some of the space that we would tend to want to use for functionality, such as captioning menus or the captions themselves. So you're going to have content and functionality competing for physical space within your screen.

We're going to need to identify those issues early on. Document and communicate the challenges that that type of competition for screen real estate is going to pose. And again, engage the decision makers who are ultimately going to have to agree on, again, the suite of functionality and how we implement it and how is that going to look, how is that going to lay out, how is that going to be represented on-screen in a way that ultimately can continue to be intuitive as an app holistically, but, again, specifically within the realm of the assistive technology that we're looking to implement in the form of captions.

Another area that is likely going to come up, particularly if you find yourself in a position where you're working within an application space that has a wide variety of content sources, where the captions that come with those different pieces of content, from those different sources, more than likely might come with some of their own variation of competing standards or use of those standards and how captions themselves are provided. Once you've spent a decent amount of time within the captioning space, you invariably come to the understanding that the evolving standards that support captioning and the providing of captioning have, in and of themselves, different abilities and different features available for the user.

The one feature that-- well, one of the features that was particularly important in terms of supporting our project's need to be compliant with CVAA was the standard within 708 that provide the ability to position the caption in different places on the screen so that the captions, when needed, aren't obscuring, say, chyrons at the bottom of news programs, the mouth of the speaker, the face of the speaker, or key action that is happening on-screen.

In order to push the system in general for consistency's sake, but also for the sake of complying with CVAA, we had to make recommendations around what is the path towards

commonization. What's going to be the incentive structure that's going to have all of your content providers who, depending on how their content is being used right now, may not have previously needed to worry, let's say, about the feature set that's available in 708 that isn't available in the 608 standard? So making those recommendations for the business to be able to-- whether it's a incentive structure or some sort of penalty of noncompliance with the 708 standard so that within a reasonable amount of time, both for them as a business and for the SCC, that all captions throughout the system would be available with the positioning that's made possible within the 708 guidelines.

Now we've just covered three decent-sized opportunities to, quite frankly, make our lives easier as a project. We've taken a look at three types of kind of broad decisions where we realize that, let's say, we as a technical team-- or we as a team implementing retrofitting of captions into an application-- won't necessarily be the final authority necessary to make some of the logistical decisions that are going to be important for us to eventually successfully complete our work. But the fourth item that I want to discuss and to highlight are kind of something more holistic.

What are some of the things that we know, having worked on projects-- whether that's within the caption space or within accessibility more broadly-- what are those decisions that we've come to recognize as ones where someone might have a question about this later? Someone might disagree with some of the decision making that we made in good faith, using the best professional judgment that we had available and top notch technical and design acumen necessary to, again, make a system that generally, adheres to good user experience principles, but specifically implements captioning and other types of user accommodations, not only well but in a way that users who need them will believe or understand that the same care and the user experience was taken to provide them with a top notch experience.

Sometimes, even with all of that, there may be disagreements with the decisions that were made order to make those types of accommodations within the system. There might be individuals whose combination of disabilities might not be as well served by what we provided and how we provided it as we hoped. But what really kind of is the hallmark of a mature team looking to achieve accessibility, using captions or outside of that realm, is being able to demonstrate a decision-making framework that is not only sound but is applied consistently and adjusts when necessary.

So within this space generally or specifically, what are some of the decisions that we think

might come across our docket? What is the appropriate way to research the appropriate ways to face those challenges, to make decisions that we think best serve the user. But I think most importantly for us as an organization, looking to achieve these goals, again, consistently and in the user's best interest, being able to not only document what we are trying to do, what we learned in getting there, but then again the criteria that we followed to make those decisions.

So again, if time goes by, someone challenges the decisions that were made or brings some sort of legal challenge to our ascertation that we've done the work to support our users, we'll be able to, again, demonstrate in whatever venue that we've done the work, and we have documentation and a quote, unquote, "digital paper trail" that backs that ascertation up. Again, not only from the perspective that we want to protect ourselves and protect our clients, but that we want to be able to demonstrate when these types of questions do come up that we are working on the behalf of the best interests of our end users.

This type of decision making, this type of framework to achieve that decision making, and the consistency of using that almost like a machine is the hallmark of a mature organization that's using deliberative processes, using them consistently and maturely, looking for the goals that, in general, an accessibility program office would be looking to achieve. Again, so that you can prove that your work is above reproach, and more importantly, for our users be able to examine some of that decision making.

Analyze whether or not maybe, in a certain circumstance, some of the rationale that was used may not in fact be as sound, in retrospect, or whether, particularly, maybe we didn't have all the information that was necessary to craft a solution that we now all understand is appropriate. And being able to look back on this, look back on our decision making, look back on our process, seeing how it may not have served a particular set of users, be able to make the adjustments, make that kind of continuous learning, not only within individual practitioners but within your organization at large.

JOE DEVON: Thank you, team. So in conclusion, succeeding in a project like this would not be possible without proper planning. There's a strong temptation when you have so much work to do, so many devices, such a large team, to just jump in and get started. But if you don't do that with a plan, you are going to plan to fail. So you need to create a solid testing plan, figure out how you are going to be implementing fixes while new product is being developed, figure out how to maintain it.

Essentially, it is like you are fixing the engine of a running train. It is extremely challenging. And it's important to identify and front load tasks that have long lead times, particularly if they require stakeholder deliberation. And last, but certainly not least, accessibility is a journey and not a destination. And we're fortunate that the Pluto executive team from the top down has decided that accessibility is going to be core value for them.

And they have created processes in place in order to make this part of how they are developing their product in the future. So this was a really big win. And we were really happy to be working on this project. Thank you, everyone.