SOFIA LEIVA: Thanks for joining this webinar entitled "How Slack Approaches Accessibility Testing." I'm Sofia Leiva from 3Play Media, and I'll be moderating today. And today, I'm joined by Sharanya, Test Infrastructure Staff Engineer at Slack; Kirstyn, Quality Staff Engineer at Slack; and Kristina, Quality Staff Engineer at Slack. And with that, I'll hand it off to Sharanya, Kirstyn, and Kristina, who have a wonderful presentation prepared for you all.

KRISTINA RUDZINSKAYA: So hello, and thank you for taking time out of your busy schedules to attend our webinar. My name is Kristina. Oh. So sorry, technical difficulties. So my name is Kristina, and I'm here in the middle. And it's a tremendous pleasure for me to introduce my colleagues, fellow quality engineers, Kirstyn to the left and Sharanya on the right. Our goal is to share how we incorporated accessibility at Slack with the main focus on testing.

Slack is a communication platform that brings all of your tools together in one place. It's where work happens. We'll introduce you to Slack company values, which helped us build a strong cultural foundation for accessibility; talk about our journey, along with the challenges, and ways we solved them through testing.

Slack's mission is to make people's working life simpler, more pleasant, and more productive, and is driven by company values such as empathy, courtesy, and solidarity. Empathy is deeply embedded in Slack culture. Historically, it has driven most of our accessibility efforts too. Slack [INAUDIBLE] community that wanted, but couldn't use Slack. They create accessibility channel because everything starts with a channel at Slack, and that's where the first accessibility discussion started.

Next value was solidarity. We were on this together. We were driven by a goal to deliver the most delightful, inclusive experience to our users. So we switched to making decisions with accessibility in mind. We kept each other accountable by introducing checklists for all software development lifecycle functions so nothing could slip through.

Courtesy. We realized that the earlier we have accessibility built in, the more people would be able to use Slack. But for that, we used an unconventional approach. We focused not on just fixing bugs, but on improving human experience overall.

But how did we start? In the early beginning, we didn't have an expertise in what it takes to build an accessible product. Just values were not enough to kick things off. We needed

expertise the most, and so accessibility team officially formed three years ago. The team started to think proactively forward, build foundations and accessibility-specific features. Accessibility team offered training to enable everyone to do accessibility.

But with all that expertise, we still had a problem. We couldn't scale accessibility efforts. Why? Because everyone should own accessibility. Slack values became foundation for building our accessibility strategy, which was a joint effort of both accessibility advocates and accessibility team.

So how did we empower to build accessible features? That's a tricky question, but we did find a solution. We came up with a strategy which was called Everyone does Accessibility. It helped us answer the following questions. How do we increase accessibility awareness and knowledge? How do we fit accessibility into the existing process? And what's close and dear to us quality engineers, how do we provide a thorough coverage? And later on in this webinar, Kirstyn and Sharanya will help us illustrate answers to those questions through the lens of quality. Tactically, the strategy helps each role involved in software development with a simplified answer on "what's in it for me" question.

And as quality engineers, we will focus on quality now. Quality engineers are always closer than anyone else to end users and can feel their pain. As the embodiment of empathy, courtesy, and solidarity, QE were the first adopters of the strategy. And this wonderful lady on this slide, and next to me, will explain how we changed quality culture and involved quality engineers into accessibility testing.

KIRSTYN TORIO: Hi, my name is Kirstyn. I'm a quality engineer at Slack. Thanks for the lovely introduction, Kristina. As Kristina mentioned, we wanted to fulfill this mission of "everybody does accessibility," but we needed to address a few problems, such as the limited understanding, fitting accessibility into the existing process, and providing thorough coverage for all accessibility features.

So I'll take you through the manual approach, and I'll illustrate this by using a pyramid. So we'll be using this pyramid as a guide to help illustrate our process for incorporating accessibility into our manual quality-testing process. I'll start by talking about the bottom layer, foundational testing; to the middle layer, Slack-wide testing; and then to the top layer, external testing.

Despite being a young company that thought about accessibility early on, Slack hadn't built

accessibility into the product, so we needed to retrofit it into the existing code. This was our reactive approach to accessibility. We focused on building features that increased our customers' ability to use Slack, and I became the person that manually tested these features and built the foundation for Slack quality.

So what goes into this foundational layer? The features that I'll be talking about were the highest requested by our customers with disabilities, so we wanted to build our empathy by listening and making sure that these crucial needs were met. And on this slide is an emoji representation of what those features are. So the magnifying glass represents our zoom, and then the stop sign represents stopping animated images. Next is keyboard navigation, and lastly, screen-reader support.

So when these features were first being developed, only one person was needed to test. But soon enough, the product became more complex, and we still needed to deliver on accessibility for all of the new features being built. So our next question was, how are we going to make sure that we would scale our product and still support these features?

That brings us to the next layer of our pyramid, which is Slack-wide testing. Many quality engineers hadn't tested for accessibility before, and a lot of times people don't know where to start when they want to make a product or feature accessible. And that's OK because that question alone can be pretty daunting. But for our testers, I wanted to demystify this question, and to do this, I needed to evaluate the current process we had. So this brings us to Slack-wide testing.

This layer is where we update any existing processes to ensure that both existing and new quality engineers are supported and empowered. If I could explain this slice of the pyramid in one message, it would be this message shown here. This message from Kristina highlights our strategy for scaling accessibility across the quality team at Slack. She talks about internal quality responsibilities, training, and the discussion of outsourcing, which I'll cover later.

We needed to strategize the internal effort in the way that targeted the most critical features for Slack, but also in a way that didn't require too much expertise. So we decided, for Slack, it would be color contrast and keyboard navigation. For our attendees in the webinar, this approach may vary depending on your team's knowledge of accessibility and the level of commitment that you can promise to your audience.

So now that we identified two critical parts of accessibility testing, we needed to make sure that

we were accounting for this strategy in each feature. We decided to make a checklist of the internal strategy to make sure people were considering these two facets of accessibility testing. This checklist would be present for each quality engineer to use in their test plans. And once we identified the checklist criteria, it was time to make sure people had the knowledge to proceed.

So we needed to educate people, and for Slack, we have an internal learning library that folks can use to educate themselves about all parts of accessibility, and what features we have in Slack currently. On this slide is the screenshot of the actual course itself that any Slack employee can enroll in to learn about accessibility testing.

And so in this course, we also have quiz questions that helped supplement the information in those courses. For example, on this slide, I have a quiz question that asks what the contrast ratio for WCAG AA guidelines is. We also provide tools for them to find out how to test for color contrast.

And on this slide is an example of what our keyboard-navigation course looks like. And the example shown here is a quiz question that asks, if you're testing a new dialogue component, what are the behaviors that you should expect when testing with the keyboard? Anyone can always retake this course as a refresher if they need to, and it's not just limited to quality engineers. Anyone can take this. So let's move on.

So now that we've defined the strategy and we've educated the team, we realized that we still needed that expertise to cover more robust and complex scenarios, such as screen-reader usage, and we solved for this by turning to our external partner, ULTRA Testing. For those who don't know, ULTRA provides highly flexible, high-quality software-testing services through exceptional onshore teams that include individuals on the autism spectrum. We have had a partnership with them since late 2016 for functional UI testing, but we recognized that there was a lot that they had to offer in terms of accessibility testing.

ULTRA accessibility testers have been extensively trained on how to use assistive technology and understand the accessibility guidelines. They find exquisite bugs and write detailed reports, which, for quality testers, is always a major thumbs-up. Their bug reports also provide remediation steps for any bugs and explain why a certain scenario is broken with regards to a guideline.

I'd now like to take you through the process of how Slack and ULTRA work together. So the

first step is that an incoming project is sent to the coordination channel between Slack and ULTRA. Next, the quality engineer provides a feature demo via a conference call and also sends over the test scenarios. Next, the ULTRA tester reviews the material and creates accessibility test cases based on that.

Next, the ULTRA tester runs the cycle. During this time, communication between Slack and ULTRA will stay open in case any questions or changes come up. And as the cycles are completed, the defects that were reported are assigned to a Jira project for Slack bugs.

We then review the bugs ourselves and send them to the feature team for them to fix. And once those bugs are fixed on Slack's end, we can resend the bug reports back to ULTRA for verification. And the entire process is handled through Slack with our Shared Channels feature, and we're able to collaborate together in the same channel.

And ULTRA helps decrease the load on manual testing, but we still needed to scale our efforts from an automation perspective. The person that will be talking about this next is Sharanya.

SHARANYA VISWANATH: Thanks, Kirstyn, for walking us through our testing philosophies and how we leverage experts in the industry to test for accessibility at Slack. I'm Sharanya, and I'm a staff engineer on the test infrastructure team at Slack. I hope to share some of my experiences with accessibility automation and how my team helps scale, as well as reliably run, those automated tests at Slack.

As Kristina had mentioned, the accessibility ambassadors and the team focus on preservation since we deploy code to production between 5 and 10 times a day onto the map, which is basically frontend, and backend services.

And for mobile and desktop, we merge call daily to several variants of our app, be it internal alpha, external beta, before releasing it to the app store every two to four weeks. So that's quite a lot of times, right? How do we both develop, as well as test, and manage quality around the schedule? Well, the saving graces are automated tests.

Like Kirstyn, I also have a pyramid for you. You may have heard about the testing-pyramid principle, which is to consciously validate for data at the right data layer. By following this principle the best we can, it helps us scale test code efficiently so that the heavy tests don't end up validating what the lightweight tests can do, and still can provide a very effective impact. This philosophy helps with the faster feedback loop and can also reduce the load on

test instances.

At Slack, we strive to adhere to the test pyramid and have modeled it for testing for accessibility to rely on linters, unit tests, and UI tests. We have compartmentalized these checks before and after code gets shipped to production. We heavily invest in pre-merge checks because we want to minimize regressions. We want a faster feedback loop.

So let's go over the types of pre-merge checks in the forthcoming slides. Pre-merge checks, linters-- they are the first and foremost checks in the pipeline. Linter job serves as the first line of defense here. Linters are used to analyze source code, to flag for any sort of programming errors, bugs, or even stylistic errors.

So at Slack, we use a React-based plugin, which is the JSX accessibility plugin, to help lint source code for both the browser and the desktop app. So as an example, this plug-in actually makes sure whether you have an alternate text for an image because that's how our visually-impaired users can actually understand what this image is about or have more context about it.

Next up is unit tests. We use JavaScript unit framework for the development and same for testing too. We try our best to write clear and concise, testable code, and we write tests to test every possible component in our code.

For example, let's say if I want to validate a function should be returning a black or white text based on the background color, or even false, if an invalid hex color is provided. So the validation here is to make sure that the original function returns expected values for the conditions. So this is an example for a unit test that we use at Slack.

Next step, UI tests. So this is the last part of the pre-merge check, which is the end-to-end or functional UI tests. These tests mimic the user's actions via the UI, be it browser or app, and validate if the control flow works as expected. So we've been using Ruby WebDriver for UI tests, and we have made a shift recently to take advantage of Cypress, which is another JavaScript-based tooling. We specifically chose this tool for several reasons, and specifically around development experience of not having to switch contexts between development, as well writing tests.

Putting all of this together-- so let's keep in mind that the UI tests are heavy, so we only run critical UI tests as part of our pre-merge check and run extensive tests post-merge every hour.

With that being said, I want to help piece all of this together on how they look on our continuous-integration system.

In the screenshot, you might notice that we have automation-smoke-ui, jsunit, as well as lint. automation-smoke-ui is the end-to-end test. jsunit is the unit test, and lint is the linter test. Unless and until all of these pass, the person who is opening the PR will not be able to merge their change and ship it to production.

So with the pre-merge and post-merge checks, I also want to take the time to illustrate how early do we think about automation, and how it flows along with our process to assure for quality. We tackle this in three different stages. One is before development, one is during development, and one is post development. Let's take a sneak peak at each of them.

Before development, there are a few steps that we do. First, we meet. I mean, who doesn't like meeting? Second, product and engineering start thinking about a project, and appropriate representation from quality and automation engineering are involved in the kickoff meetings when the ideas get fully baked and approved by our internal council. That's when the key blockers, edge cases, are identified. They are also discussed in the meeting. And post-kickoff, quality engineering comes up with a test plan, partners with dev to identify the pieces that can be automated via unit or end-to-end tests. This is where the test-pyramid principle also starts getting involved or couple [INAUDIBLE]. Next is [INAUDIBLE] and prioritized work is tracked on our project-management tool and have made sure that they are done before the feature gets shipped to our users.

Next step is during development. Both development and quality engineering write and maintain tests, and sometimes these tests may not run properly and be flaky. A flaky test is a test when it fails when there are no changes detected in the source code. It sends the wrong signal to everyone that consumes the test report about performance failures, and it's critical to maintain a healthy test suite.

So when someone develops tests, they have the opportunity to run a test several times in a stability test [? job. ?] Once the test has 100% stability, it is introduced in the automation test suite as a pre-merge or pre-blocking test.

There are times when a test, despite having 100% stability, be flaky, say, when a page genuinely loads slow and test waits for the element to appear. The test can fail temporally.

Test infrastructure team has put forth appropriate tooling in place to surface flaky tests, and there are also [INAUDIBLE] defined for test owners to fix those flaky tests depending on test priority, severity, and also the stability rate.

Next step, after development. The regression automated tests that catch bugs post-merge, and the triage captain triages the test failure to follow up with appropriate engineers in a fix forward, revert, or even having the logic behind a feature flag to allow for further investigation and thorough testing. This is such an example where our triage captain has posted saying that, hey, my tests are failing. This is the report, and I suspect that there is a change that was actually made to the source code that's reflecting our test to fail.

But for further usage of feature flags, Kristina will show us an example of how we use feature flags, but also utilizing manual as well as automated testing in the forthcoming section. That was all for me, folks. Thank you very much.

KRISTINA RUDZINSKAYA: Thank you, Sharanya, and we do have some great examples to show. For instance, this case shows a fantastic collaboration of quality engineer outside accessibility team, accessibility quality engineer, and an automation engineer.

We usually feature flag features or bug fixes until they're thoroughly tested. So what happened here? Quality engineer outside of accessibility team surfaced an issue, which ended up to be a system bug with all dialects across the app. This issue was quickly prioritized, fixed, and feature flagged for verification. QE from accessibility and outside of accessibility team verified the fix for keyboard, screen reader, and collaborated with automation team to add tests to the automation test suite.

Or this example, displaying our calibration with ULTRA. While testing custom status expiration, ULTRA found a lot of bugs for the date-picker component, which was also used to filter search results. Fixing those bugs made the component accessible and helped us release two more accessible features.

Isn't it amazing? We believe that we will be able to scale this strategy through our values and have more success stories to celebrate and way, way more delighted users. Thank you for giving us the opportunity to share our knowledge and for listening in. We now look forward to answering any questions you might have for us.

SOFIA LEIVA: Thank you so much for such a great presentation. We have a couple questions already on

queue. So the first question we have is, can you talk about how slash if the quality team works with UX, design, and research teams?

KIRSTYN TORIO: So the question was if the quality team works with UX and research?

SOFIA LEIVA: Yeah, UX, design, and research teams.

KRISTINA RUDZINSKAYA: Yeah, maybe I can take this. This is Kristina. So as Sharanya mentioned, we do have kickoff meetings, during which all the people involved in the feature development get together. And usually what happens, everyone already had accessibility training for their roles. So they know what they need to implement, from accessibility standpoint, in their designs. And quality engineers can get together with them and just clarify if everything was addressed in their designs.

KIRSTYN TORIO: And to add to Kristina's point, we do have a designer on the accessibility team. So if there's a team that we work with, then the quality engineer on the other team will partner with the designer as well.

SOFIA LEIVA: Great. Thank you. The next question we have is, how do you ensure that engineers are taking the quiz and implementing what they learn?

KRISTINA RUDZINSKAYA: I think it goes back to our values as a company. So we actually encourage everyone, during the onboarding, to get to know all the processes, and accessibility is a part of the process. And there is a training for each role, as we mentioned before. And for engineers specifically, they cannot ship their code if the code is not accessible, the front ends. Because that will be caught during the testing and they have to fix a lot of bugs. So it's in their interest to ensure their code meets accessibility criteria.

KIRSTYN TORIO: Yeah, front-end developers also have their own separate accessibility training that is led by our friend and engineer on accessibility, Todd. So while this specific training is more geared towards a general audience and maybe with specifically quality testers in mind, there's still additional training that happens outside of this course.

SOFIA LEIVA: Thank you. The next question we have is, how do you ensure that accessibility is happening further up the funnel with designers and product managers as well?

KRISTINA RUDZINSKAYA: Yeah, that's a great question. So there are simplified checklists for each role, and our project managers are doing a fantastic job in ensuring that everyone just thinks of their checklists, and

accessibility is obviously in the checklist.

SHARANYA VISWANATH: The software development lifecycle process at Slack-- this is Sharanya here. So that has accessibility embedded into that checklist. So to just add onto Kristina's point, this is incorporated as early as possible, and with all roles here at Slack. Unless and until we meet certain accessibility requirements, we can't ship it. So it's a duty of each and every person to actually have a fair understanding of how accessibility works at Slack and how to meet at least the bare-minimum requirements. So training, education, and also that reflects their own quality of how they're delivering their work in their own roles.

So to also touch a little bit about career ladder, that also has a technical quality as one of the spheres that we evaluate each person. So it's sort of culturally embedded in all spheres.

SOFIA LEIVA: Thank you. The next question we have is, which automation library are you using?

SHARANYA VISWANATH: Hi. So for Cypress, we've been using in house. Some of them are in house, and we've also been trying out axe accessibility library. And for Ruby, we've already had a accessibility library. That's the only library that we are using from the third party.

But for other cases-- say, if I want to test keyboard navigation works as expected-- then within the automation, we have special tags that our automated tests look for. And they validate whether the navigation happened as expected. So some of them are in house, and we do use one external library, which is axe accessibility library.

SOFIA LEIVA: Thank you. The next question we have is, when testing Slack, you're obviously testing against WCAG 2.1. But since Slack also generates content, have you looked at ATAG, Authority Tool Accessibility Guidelines?

KIRSTYN TORIO: So for us specifically, we do have guidelines that we test against. I think WCAG is probably the more widely known one here at Slack. We also try to use experiences themselves as what we test accessibility against. So it's based on how someone would be interacting with the product, not necessarily via a guideline. We could probably look into that more, the ATAG.

SOFIA LEIVA: Thank you. The next question we have is, do you have blind participants in your user testing? Are they native screen-reader users?

KIRSTYN TORIO: So recently, I think we've reached out with Fable. They're an accessibility-research company based in Toronto. And our project manager, George, has been helping conduct that research.

And so the participants are blind users.

KRISTINA RUDZINSKAYA: To add onto that, we also have some blind testers from ULTRA testing.

KIRSTYN TORIO: No, we don't.

KRISTINA RUDZINSKAYA: Oh.

[LAUGHTER]

It's just the Fable.

KRISTINA RUDZINSKAYA: OK.

SOFIA LEIVA: Great. Thank you. The next question we have is, do you have any tests for consumer data? For instance, uploading emojis with adequate color contrast or providing alternative text on images?

KIRSTYN TORIO: At the moment, as far as emojis go, I'm not too sure. I know that when someone uploads an image to Slack, we do encourage people to use the description tag because it automatically adds it as an alt tag to the image.

SOFIA LEIVA: Thank you. The next question we have is, do you rely on manual testing more or automated testing more, or are both equally important to you?

KIRSTYN TORIO: That is a good question. So I'd say that manual testing is always going to be very important, especially because accessibility is varied across different OSes, different types of screen readers, and sometimes automated tests can't catch that. But we do try and prioritize automated testing. How do you feel about it, Sharanya?

SHARANYA VISWANATH: Yes, I think Kirstyn summed it up pretty much very well. I think some of them are looked at very early on. We invest heavily on linters, unit tests, and then finally, we come to end-to-end tests. And when we look at things that can be automated, I would say about 30% can be automated, but most of the validation is relied on manual testing.

SOFIA LEIVA: Thank you. We have a couple more questions. The next one is, where can someone file an

accessibility issue? Is there someplace we can see what issues are outstanding accessibility issues, and when they might be fixed?

KIRSTYN TORIO: So if you'd like to report an accessibility bug within Slack, we do have several ways. One is if you're using Slack, you can type in forward slash feedback, and then that will trigger a dialog that gets looked at by our customer experience team, and they're amazing. We have two very dedicated accessibility ambassadors who filter out accessibility tickets, and then we can take a look at it directly.

There's also a Help menu, which you can access through the triple ellipses at the top-right of Slack. And then that also leads you to the Help page to report feedback. And then, if you prefer email, we have feedback@slack.com.

KRISTINA RUDZINSKAYA: Yeah. Unfortunately, there is no place where we can list all of the accessibility issues. But if there's anything that definitely blocks you from doing your work or there is a really significant bug that you reported, we will definitely follow up with you when we fix it. Our customer experience team is just absolutely fantastic, and they're delightful to work with. Yeah. They will keep you posted on the update on the issue.

SOFIA LEIVA: Thanks. The next question we have is, what resources do you recommend for engineers to learn more about accessibility?

KIRSTYN TORIO: One that comes to mind right away-- this is something that I was looking into, but Google has this great set of videos, if that's a preferred way of learning. They have a lot of YouTube videos about accessibility. Off the top of my head, I can't remember. I'll see if I can-- I think it's called like A11ycat or A11y-- something.

SHARANYA VISWANATH: We can follow up on that.

SOFIA LEIVA: I can include that link in the email tomorrow as well. So two more questions. The first one is, does Slack have a central accessibility team? And if yes, then how many people are there in your central accessibility team, and what departments are they from?

KIRSTYN TORIO: Yeah, I can answer that. This is Kirstyn. So we do have a central accessibility team. We do have one product manager, one designer, two front-end engineers, one quality tester, and one program manager. And we also have the two specialists on our customer experience team. So it's a pretty small team, but that's everyone on our team.

We also have people that are very ingrained in accessibility that are on different teams. So we have iOS developers and Android developers, but they're not necessarily part of the accessibility team. They hold those parts of the product accountable.

SOFIA LEIVA: Thank you. And the last question we have is, what advice do you have for a company that is just starting to think about accessibility?

KIRSTYN TORIO: I feel like Kristina has thoughts here.

KRISTINA RUDZINSKAYA: That's a really good one. It depends on the company size. It depends on the company processes, but I would say start small. Don't try to achieve big goals from the beginning.

And what helped us as a company, it's our values. Slack deeply values empathy, and we did have a lot of empathetic humans here who advocated for accessibility efforts. So I would say if you're passionate, try to find some like-minded people within your company and start slow.

SHARANYA VISWANATH: Yeah, and at Slack, we also hired a few-- to Kristina's point, we hired a few strong accessibility advocates who have also been in that industry for a while. For example, George, our product manager, as well as Todd. They have been doing this for 10-plus years, and so they were able to also guide us, to educate us more, and to do this right. In that sense, we were lucky. We have those people, and they are here at Slack, and still continuing to educate engineers that onboard every week.

KRISTINA RUDZINSKAYA: Yeah. So if your company has an opportunity and headcount for accessibility hires, as Sharanya said, that will definitely speed up the process and definitely improve your accessibility efforts and the overall quality of your app.

KIRSTYN TORIO: Even at the beginning, the advocates that we're talking about, they targeted the low-hanging fruit. So quick accessibility wins, like if they could add a label to a button, if they could fix the contrast on something. Starting somewhere small, like Kristina said, is a good way to open the doors and have discussions about what else you want to make more accessible.

SOFIA LEIVA: Thank you so much. You've given us amazing advice, and this was a great presentation. I'm so happy to hear how much work Slack's doing. It's really awesome.

KRISTINA RUDZINSKAYA: Thank you so very much.

KIRSTYN TORIO: Thank you so much.

SHARANYA VISWANATH: Thank you.