Video Title:

Styles & Theming in React Native - a webinar by Haris Mahmood

Video Duration:

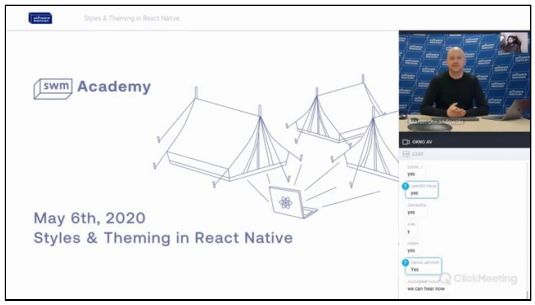
01:23:57

of Extracted Keyframes:

134

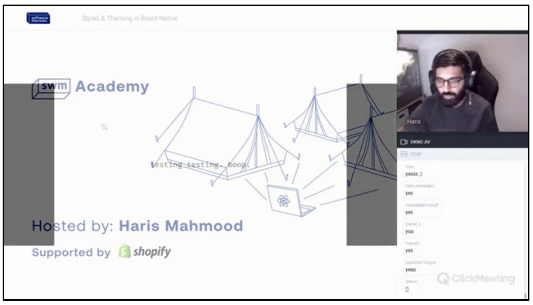
Date of Extraction:

2021/04/16

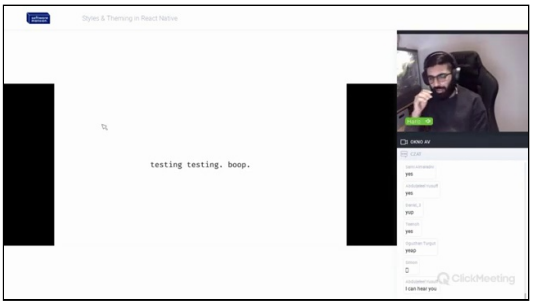


my name is marcin lewandowski
I'm from

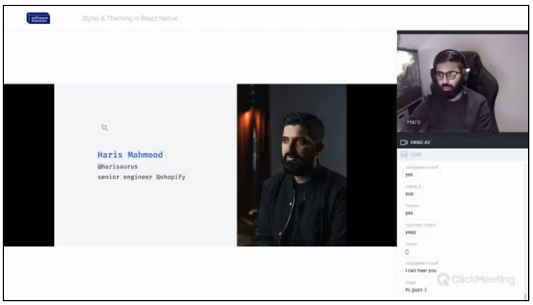
the company software mention we are an agency software agency located in Poland in Krakov music beautiful city of Kharkov and and we are not a large team but I think what makes us distinct is that that we are not only making the apps for our customers or which are mostly located in the United States mostly in New York or Bay Area but we are also actively creating technologies that a lot of you might know mostly in the sphere of react native and expo actually in large large part of the core team of expose us and and we are also actively we are also official partners of Facebook when it comes to react native so in practice that means that we are maintained creating and maintaining like several very popular reef native libraries you might know some of them like react native reanimated or gesture hunter or react native navigation and and they're pretty popular we are we are having a full-time team that is responsible for maintaining edge and recently we started cooperation with Shopify which sponsors us and allows to put more developers to the team so we can fix the Box faster iterate faster and importantly work on the new version of reanimated that is going to be see undergo significantly faster and he way



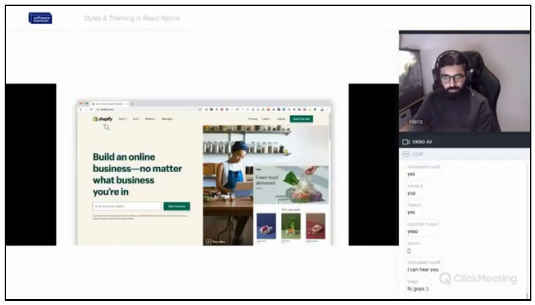
more performant flexible and you know a lot of things that are not possible with the current library and the release should happen pretty soon and the guests the cost today is harris mahmoud who is a senior developer in Shopify he's going to introduce by himself so I'm not going to do this when it comes to the organization mother's the plan is that the webinar will take 45 minutes and the question and answers later will take 15 minutes but the limit is not a treat so if you will have more questions you know feel free to ask Harris and you already received a link and it will be also short again that allows you to sign up for the follow up session with Harris that will ask for some of your questions that may arise later on after the webinar just to put this in context this is the first webinar from the series of web webinar we plan to do we stratify we are also planning to might have workshops more or less one in a month so if like all of you until you didn't click the checkbox you know you will receive information about upcoming events the recording of this will be available for the people who are who attended this webinar or will register for the next one and so nothing is lost



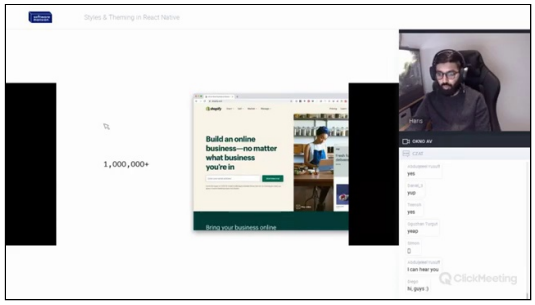
and what else to say



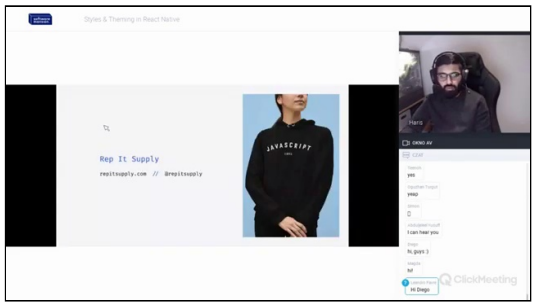
I hope you are going to spend nice evening with Harris and or nice morning



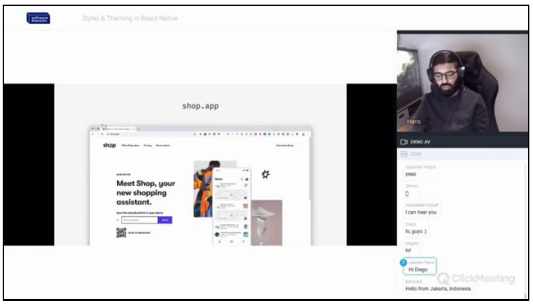
if you're in the States you will learn a



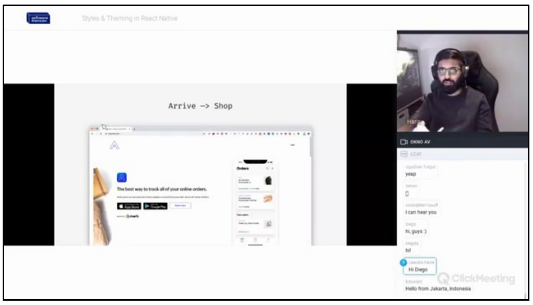
lot and you will come back to us next time so I'm not going to make this any longer I pass the mic to like virtual mic to that final works to do hobbies so thank you very much have a nice day have a



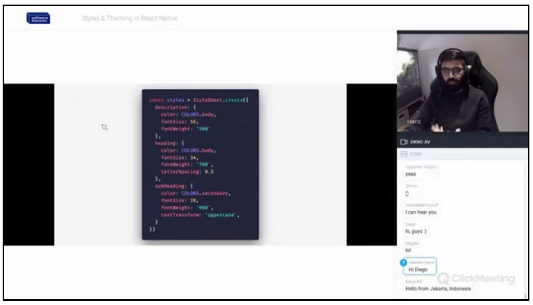
nice time alright hi folks let's just double check can everyone just make sure that they hear me before I continue on excellent perfect so I'm just going to get that started on my end perfect so we're gonna get started here this is just my testing slide don't mind me as mentioned my name is farce mahmoud i'm a senior engineer at Shopify that there's my github and Twitter and Instagram handles so feel free to hit me up so Shopify is a leading global commerce company which means that we provide the tools that you'd need to start grow market managing manager retail business of any size our platform is engineered for performance scale and reliability while delivering a better shopping experience for customers worldwide now a couple of additional neat facts is that we support over a million merchants on our platform which is truly incredible we're also so hiring so we're hiring for engineers we're not slowing down at all we're still hiring for remote roles as well in North America and Europe and in the UK time zones for some teams we're also still making relocation offers as well to other parts of the world where the relocated candidates will be relocating



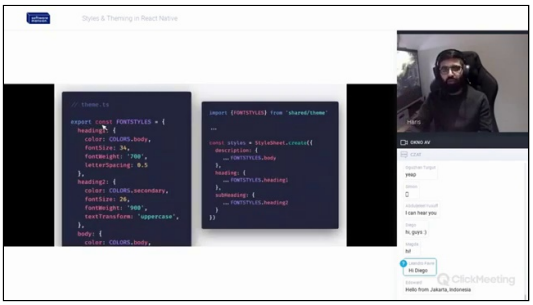
them when it's safe to do so so you may still get it you'll get your offers now but won't be relocated until it's okay for us to do so we also have a pretty awesome three-day onboarding experience our onboarding experience is now fully remote as well so even if you're a joining shop fight today from a remote location you still get to experience in an incredible onboarding experience if you're curious to learn more about some of our openings and some just like careers in general feel free to check out Shopify comm slash careers slash Cove in 19 or feel free to ask any questions as well which I'll be able to get to towards the end of the Q&A session or towards towards the end of the presentation in the Q&A session



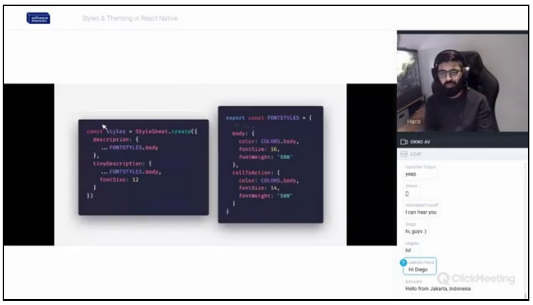
excellent so one of the things that chocolate does also is that it it encourages employees to be merchants themselves that helps us understand the platform better that helps us understand what it feels like to be a merchant what it feels like to use the platform and to like we're like our skin is in the game we feel just as much as our merchants do and that makes us help that makes the product better as well and makes the whole platform better so this is sort of my experiment with running my own store



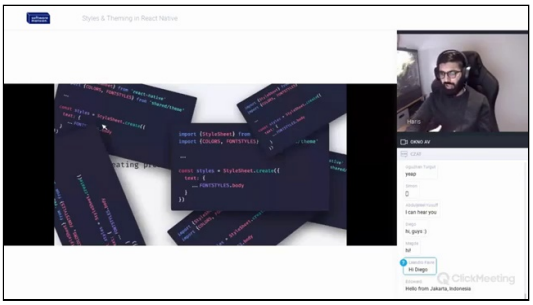
it's called rapid supply it's an online store that sells Apparel's pins stickers and accessories specifically for developers so pretty much all of you folks here today I genuinely love my JIT anywho so today we'll be chatting about styles and theming in react native so there's obviously tons of projects that are currently going on at Shopify ranging from all sorts of stuff I'm specifically part of the clients team that's working on the shop app the shop app is a shopping assistant that helps you keep track of all of your orders it helps you pay better and helps you shop better it helps there's all sorts of rediscovery elements to it so I highly recommend checking it out so the the shop app is 95 percent react native fully cross-platform for both iOS and Android and the remaining small portion relies on a couple of necessary native modules for each of those specific platforms now what I'll be talking about today are some of the problems that we faced when we first started building the app and how some of our best practices and a library that Shopify built called restyle helped solve some of those problems for us so one of little tidbits to know is that just up until a week ago shop was actually the arrive app so we were secretly working on the shop



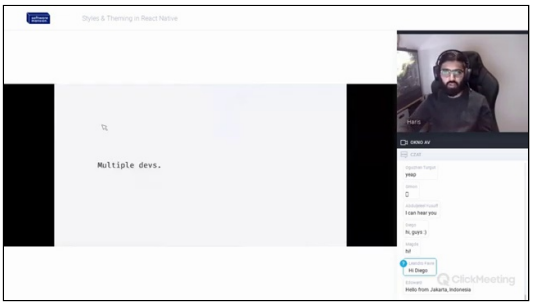
rebrand and everything else that came along with that for over a year so I'll be talking a little bit about that as well and how we were able to get how we



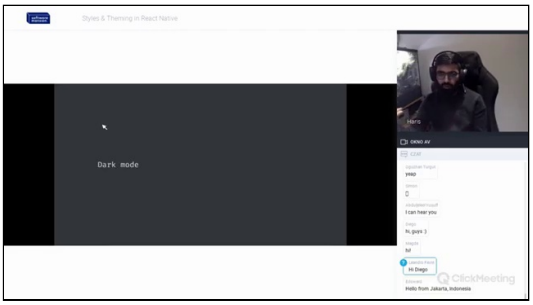
were working on shop while still releasing new functionality and features and polished work on the arrive app as well excellent so the the first thing that comes about with any project is trying to deal with managing colors now colors are notoriously hard to name they're hard to keep track of and they become super hard to maintain so for example like one of the first things someone will do is just create a



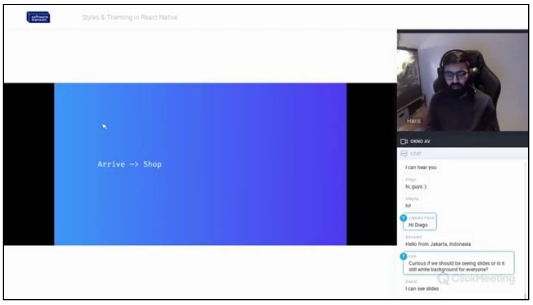
dedicated colors object and like a theme file for example and then you know like define all your colors there which you can use across your components in using the stylesheet API now the color names are not super useful there's no like well-established patterns here like so we had things like primary and secondary and body and then awkward things like background 1 and background 2 and then more like context based colors like error so this was like super frustrating to try and use and try and maintain it just becomes like super super gross the longer you end up needing to use it and as you add and remove or adjust colors already existing now similarly font styles just as painful so we realized that we were needing to write a fair amount of code to style every single text element that exists in our app so we were applying font sizes weights letter spacing transforms obviously colors then a bunch of other stuff and we just realized that like hey look this is just styling for three text components but look at how much code we've had to write already and not only that we realized like we noticed that we were duplicating this over and over again across a wide range of components and screens and that was just becoming increasingly difficult



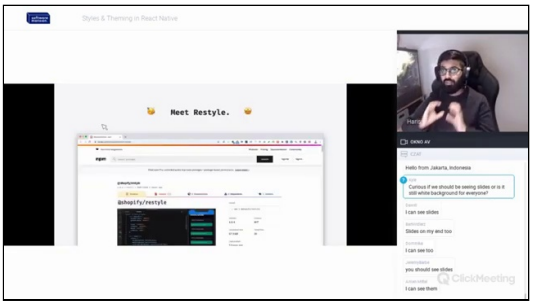
to manage so one of the first things that we did was we said hey let's follow a similar approach that we did to the



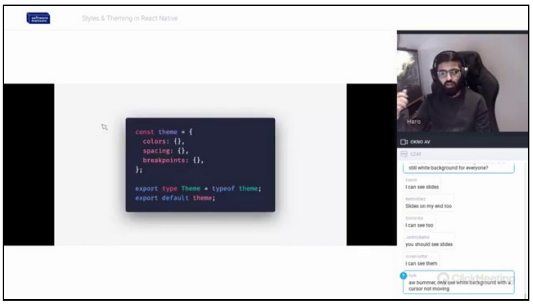
colors let's build a font styles object in our theme file and bring all of our different styles there so we defined our heading 1 heading 2 body yadda



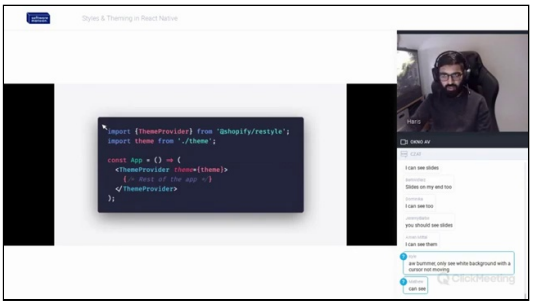
everything else there and then we would import that where necessary and just spread those those font styles where we needed to apply them so this helped alleviate some pressure with regards to like the code duplication so that was kind of nice but it didn't resolve all of our problems when it came to dealing with font styles so we have two main reoccurring issues one was let's say we had a font style called body and we came across an opportunity or like a scenario where we needed to just override one specific style so let's say we had like a small description so we would spread



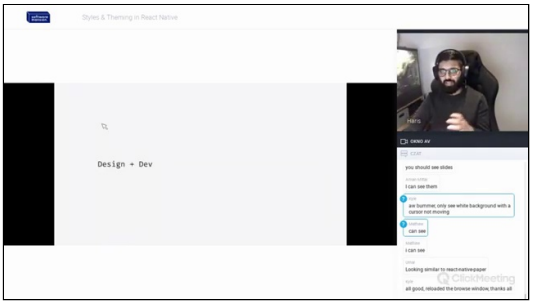
the body styles and then just override one property so we were never super clear as to like what the best approach would be for us here like do we just continue to override this one item here another item elsewhere and all of them are actually still using the body font style or should we create a brand new font style that accommodates this small override as well so that was kind of weird the second thing was determining it was challenging to determine how broad or specific the font style name needed to be was body too broad what's called to action and far too specific what happens if we needed to apply all the same styles that the call-to-action font style uses but the element is not a call to action element do you still use it do you go through a massive renaming process it was just kind of kind of gross and that obviously just like escalates as the app begins to grow the repetition problem like although fixed a little bit was still not great we still had far too much repetition it was just incredibly frustrating to have to bring in all the stuff over and over again and just spread these different font styles across the board so that was



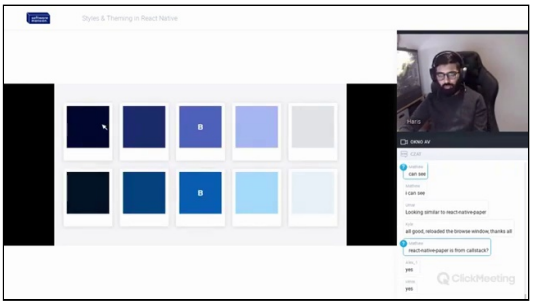
definitely something that we wanted to address things became harder and harder to update harder to maintain keep things consistent especially due to one offs and this like became worse as time went on and seemed to be working on the project for quite a while this became even even more of a problem when more and more devs grew joined the team we started off with just two folks and that's grown quite a bit since then so as new developers joined as developers had to work on various aspects of the app it just became harder



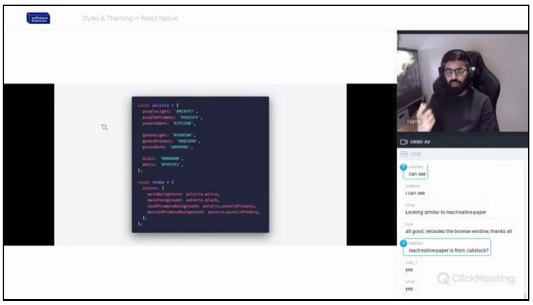
and harder to manage and things just continue to escalate we also had two new features that we wanted to work on or two new aspects that we wanted to work on one was introduced dark mode that was one of the most requested features back when we didn't have it we knew that like



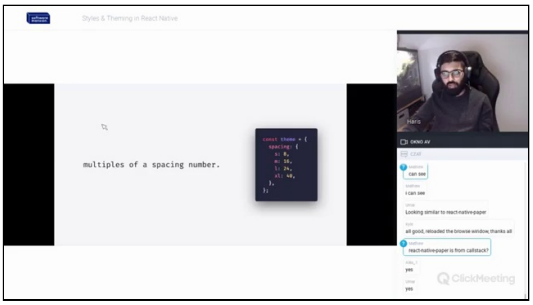
this was something that was necessary but we knew that our current state of already trying to handle our colors managing colors and our font styles and styles in general was just not gonna work well and we ran a couple of experiments to see how introducing dark mode would work and it was a nightmare



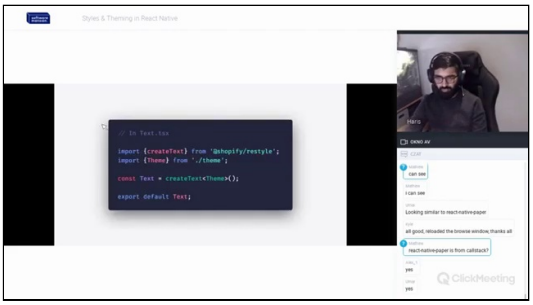
there was just like a ton of spaghetti code all over the place and we knew that that wasn't the approach that we needed to take the second thing as that previously mentioned was that we knew that we were heading towards this massive rebrand from arrived to shop we were gonna be relaunching shop with a new new aesthetic new colors new font styles new typography new UI a whole bunch of new stuff and we needed a way to continue or to begin working on shop without needing to touch the way arrived currently function door without interfering too much with how arrived



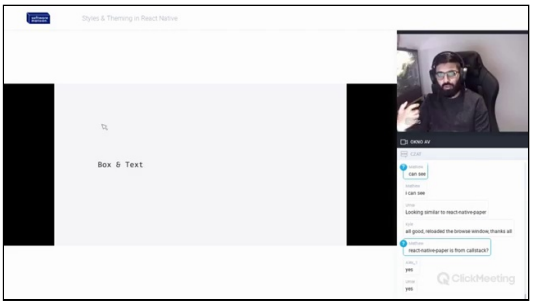
worked today we release new updates super often so we couldn't just stop working on arrive and just put all of our efforts towards shop so that wasn't an option so it needed to be something that would accommodate both of those things so flipping acts like it was like clearly our current Paul was already were plentiful and then needing to introduce dark mode and introduce like this entire new rebrand somewhere down the line really cause a lot of frustration and this was an opportunity where our team took the time to take a step back and figure out a better way of approaching this and and and this is sort of where a solution came to be part of this is just best practices that we've put into place and along with the restyle library that I mentioned early on so restyle is open source it's built in-house by Shopify so it's a but it's



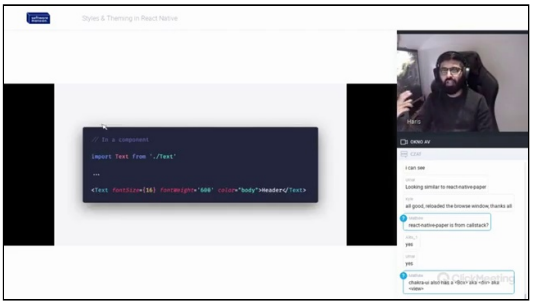
available for all of you folks to use as well it's a type enforced system or sue self provides a type and forced system to help you build UI components now the awesome thing here is that theme ability is baked in from the get-go so if and when you decide to introduce dark mode or an alternate theme or a completely new like visual aesthetic of your app like you there's no new functionality that you needs to add it's there right from the get-go so it and it becomes almost trivial to introduce some of those new things and I'll show you what that looks like as well I'm gonna walk through some of our new best practices some of the concepts that we've put into play and some of the new features that restyle introduces and how all of those things come together to create a really really awesome developer experience and for for shop so the first thing we need to do is we need to define a theme object so this is a global theme object that is going to be used by restyle restyle relies on



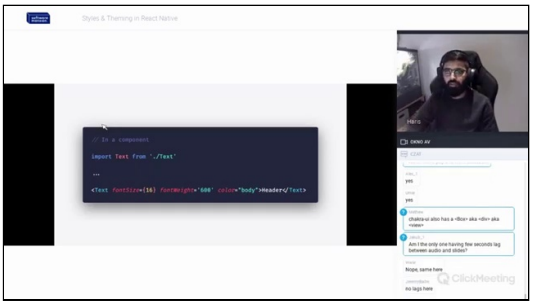
this global theme object that has a set of spacing values color values breakpoints and a whole bunch more we won't be able to get through all of the things that Rhys law provides but I'm going to focus on some of the biggest aspects cool so the way we then utilize that theme that we just created is by using the theme provider this is also provided with restyle you wrap your entire app with a theme provider and you pass in the theme that we created using the theme prop - the theme provider the theme provider uses react context behind the scenes awesome so that's all we need to get a restyle hooked into our app and the crucial point to remember next is that this is like unrelated to restyle but that the relationship and understanding between your designers and developers needs to be rock-solid you have to be we realized that we needed to reference colors the same way sizing the same way components the same way use similar terminology in the naming system that was established for those things to make sure that but we were having the most efficient conversations as possible and future iterations of work were very easy to handle so when it comes to colors quite often any time when a design team or a design person is working on on a new



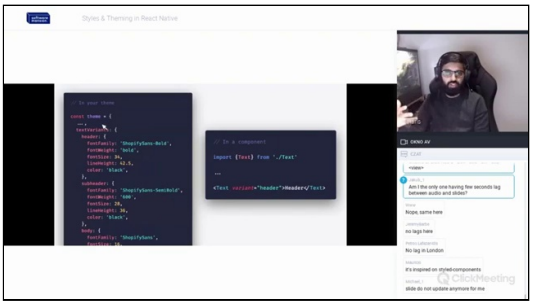
design system they'll quite often start with a color palette a color palette will consist of a set of base colors and then lighter and darker shades of those colors in this case I've got two examples and then obviously you'd have like black and white and some other like body colors or and stuff like that so it's vital that this is established



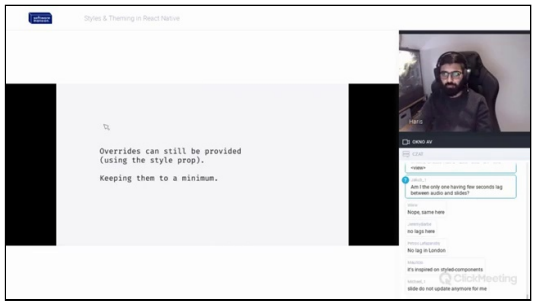
properly if one doesn't exist for the app that you're working on or are going to work on I highly recommend building one out this is the first step to help keep your design a design team and dev team in sync because one of the first steps that you have to do as a developer is to first build out a palette object now this is outside of your theme this is independent from your theme there's no app context there's no UI context all



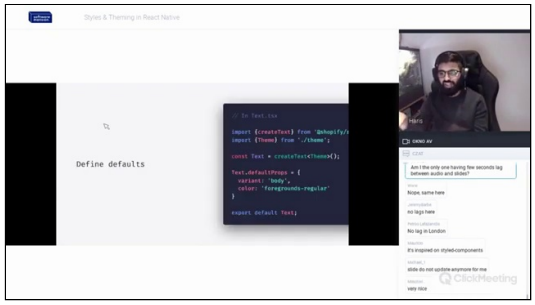
this is is a code representation of that palette so you're taking that color palette that's developed in or put together in a design tool and essentially migrating that over into code into your app what we want to do then is use those colors within our app so within our theme story so now notice that in our colors object we're defining specific keys and then values which are the color values now the keys here so main background card primary background those are very descriptive and those are context based keys that we give our names that we give so you know exactly where something specific is being used so button primary background is extremely obvious as to where that this is being used and the second part of this is that



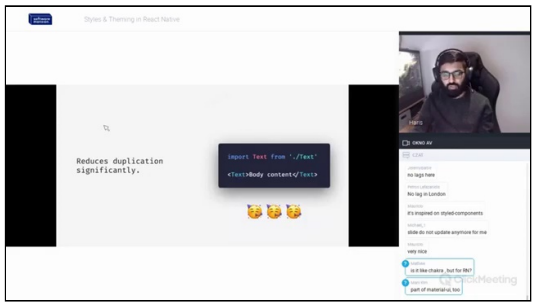
you'll notice that for example two items are using the same color from our palette so card primary background and button primary background both use the purple primary the cool thing here is that if at any point in time the design language meets the change or we need to make some design changes where one of



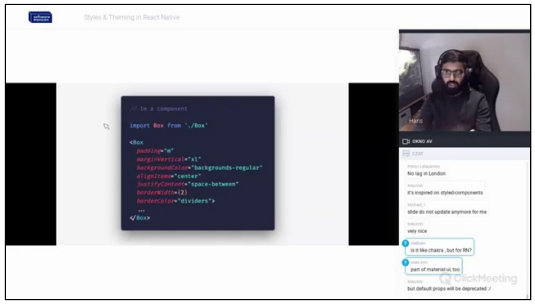
those needs to change it's extremely trivial to come in here and update just one of those to something else you're not worried about oh hey like if I accidentally change this what other crazy ripple effects is it going to have



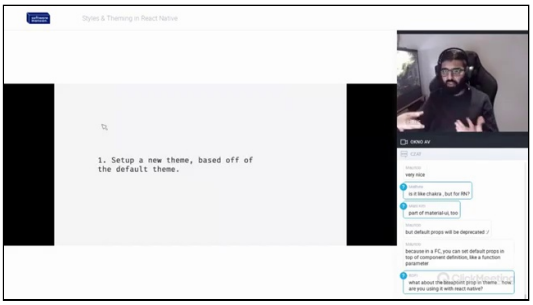
across my entire app so you can feel



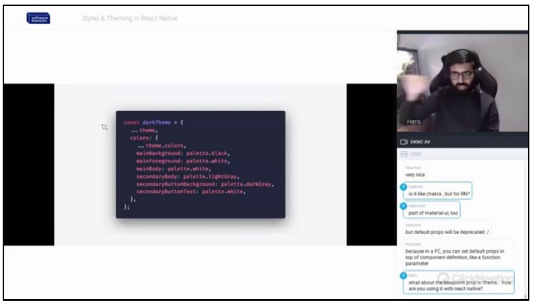
comfortable knowing that hey I can make this one specific change for this specific context and I know that everything else remains the same so this is like one major aspect of how managing colors can become significantly simplified now we also notice that quite often a design system will have a a spacing system as well if not it's pretty nice to include one and because this makes it a lot easier to to define a spacing system within your theme also and then space things out a lot quicker



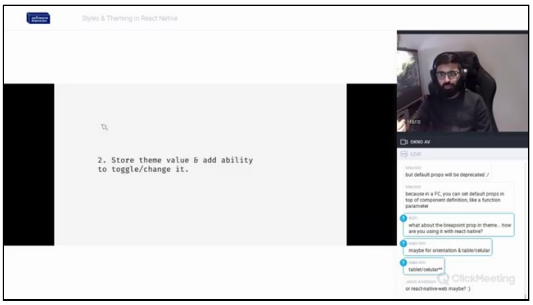
and a lot easier so we used a multiples of a number convention to define our spacing value so over here we've got multiples of four and the naming convention that we utilized was like how P shirt sizes are are established so we've got small medium large extra-large



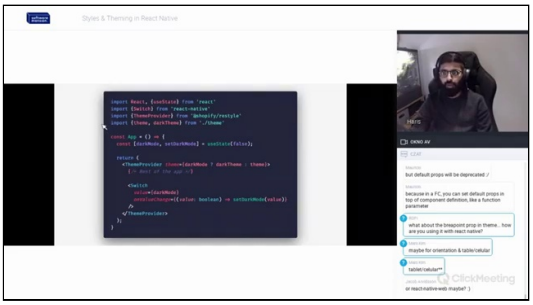
and the cool thing here is that this can easily be extended to larger and smaller values using the X notation so you could introduce double XL triple XL or fight' like in the opposite direction just have X s or a double XS you can even introduce like negative spacing values if that's something that your app needed but we tried to avoid that as much as possible so we really like this approach of spacing values I just made things far more manageable and a lot easier to quickly identify and apply across our app okay so the the view and text component that react native comes along with we realized that they weren't the best starting points we wanted to find a way to create more feature-rich alternatives and so this is where rethought comes into play restyle provides two functions that help you create replacement components for the view component and the text component so the there's the create text function that resaw provides so over here we're



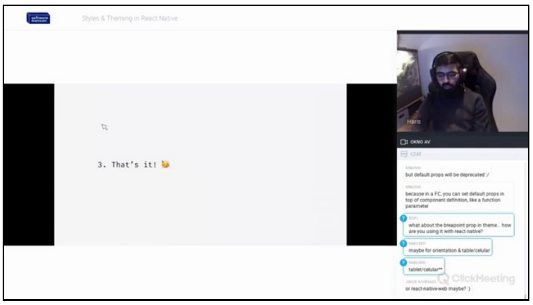
creating a new text component using the create text function and similarly we



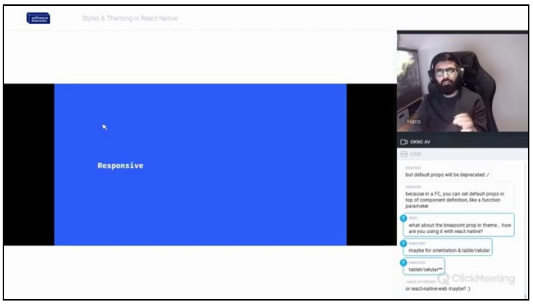
can create a new new view replacement and our terminology is box so you're obviously welcome to call us whatever you like but as long as you use the create box component sorry the create box function to do so now the cool thing here is that the box and text components that we've just created through restyle are now the replacement component or calyx and box sorry for text and view the reason why we want to do this is because these two components when you build them out you're also providing details or information about the theme and what this allows you to do is that both of these components now utilize that theme right out of the box so what does this look like so for example in the component we can import our new text component and then you can pass in



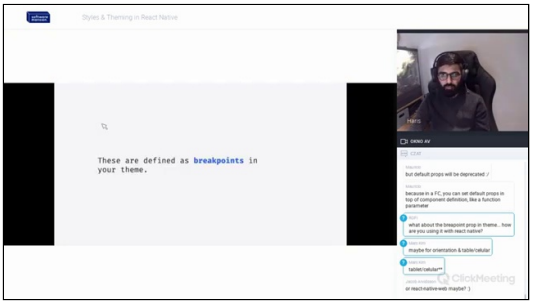
appropriate style properties using props



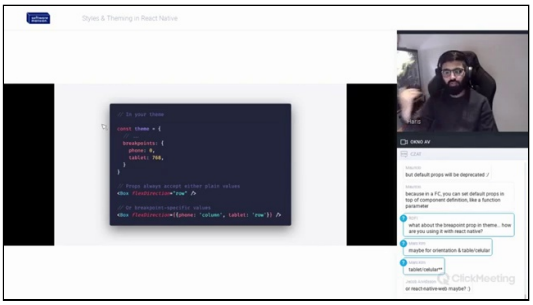
so in this for example here I can pass in font size font weight and color just by using props and specific items that we establish in our theme can be utilized here as well so you'll notice that the color value matches up with the colors object in our theme so all those colors that we defined in our theme can now be utilized directly using our new text component so so color properties measurement properties and a number of others can be are pulled in from the theme and can be utilized here so we pre restyle I mentioned that we had moved all of our font styles into this like



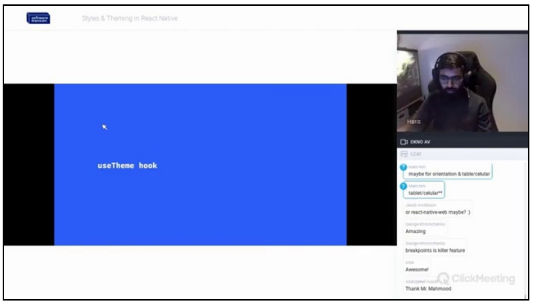
big object that we were in our theme file that we were using all over the place so the text component also accommodates something like that it accommodates the variant prop so you'll notice on the right hand side I'm using the text component with a variant prop and the values for this prop are picked



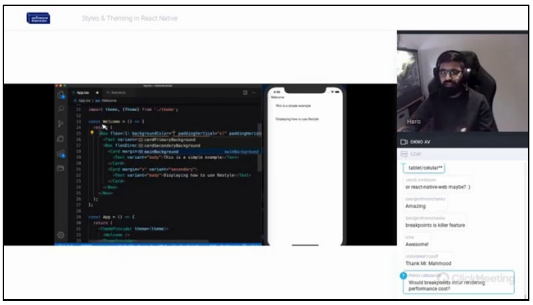
from the text variants key in your theme



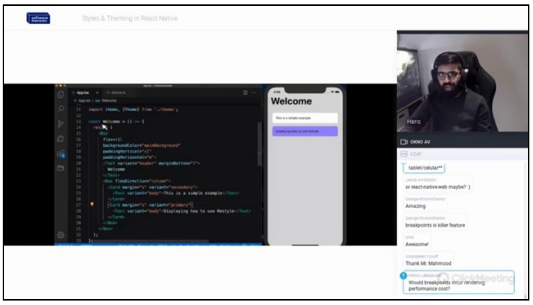
so within your theme you can use an you thing here called text variants and provide all of your different text variant styles here and then easily apply them to the text component as you need to so this eliminates a lot of code duplication and really really streamlines the entire process for defining textiles you can obviously still provide override so the style prop is still something that the text and box



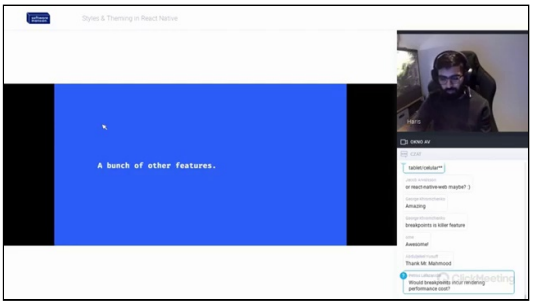
components accommodate but we want to try and keep them to a minimum just because they it that's how like overrides or sorry how like one offs get introduced so you want to try and keep those to an absolute minimum and avoid trying to use them at all this really helps keep things like really nice and clean from a resaw perspective you can also take this a step further so when you define your new text component you can also define the default props for that text component just like you would any other react component so we can define that that the default text variant is body and the color is foregrounds regular for example so what this allows you to do is that anytime you use the text component and let's say this these are like default styles that



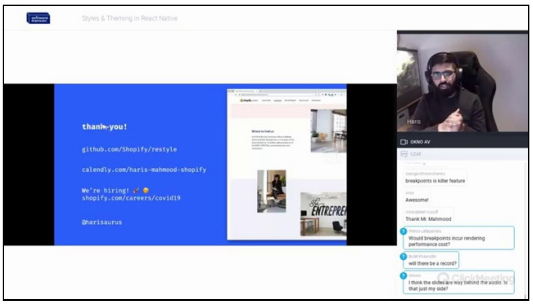
we need to use over and over again you don't need to provide any props at all to the text component because all of those are there by default and any any text component that you that you initialize or that you bring in will by default look like body content which is amazing so this immediately reduces tons of duplication and it creates and like creates a more simple and cleaner app so



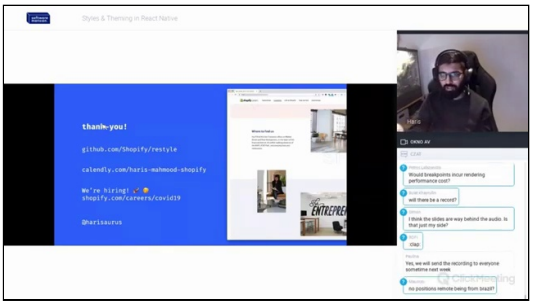
the Box component does something similar to the text component but it adds the



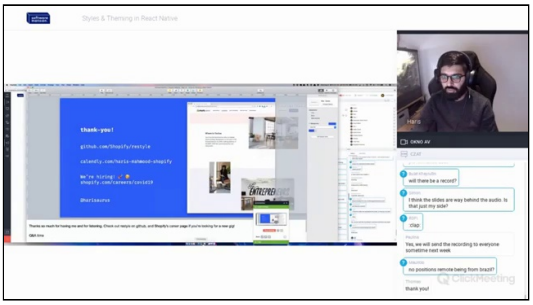
ability to use some additional props these revolve around padding margin alignments border so it's actually like layout related things as well so you'll notice here that my padding values and my margin values here are using values that I've established in the the sizing object within my theme so this allows you to really really quickly define these spacing values and the



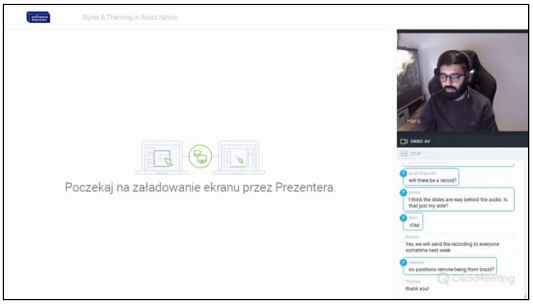
having values and border styles and whatnot for these like layout containers okay so now let's go back to theming and dark mode we know we want to introduce dark mode or we want to introduce a new theme so that sort of how we built the new shop rebrand we essentially created a brand new theme behind the scenes and worked on that so how do we implement that so if for this given scenario I'm going to be implementing like a dark theme for the for a demo app so implementing something like this can be done in a few simple steps the first thing we need to do is we need to define a new theme and base it off of a default theme so in our case I'm defining a dark theme and the first thing I'm doing is I'm spreading the entire default theme here and then just only overriding the values that are going to be different for this specific theme so quite often when you're implementing a dark mode theme not much changes aside from colors so over here by bringing the exact same default theme I bring in all the colors again within the colors object and then just override the things that I needs to for that specific theme so this keeps it very isolated as well from your base theme as well so it keeps things easy



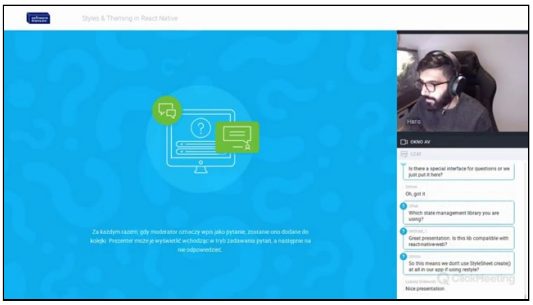
and able and keeps things clean and easy to manage now remember that we still have our pallet so you're still overriding these colors with values from your pallet and like the pallet just comes into use again it makes it even easier to make those necessary changes for your dark sea the next thing you want to do is you want to be able to store the theme value someplace and introduce the ability to be able to toggle those values back and forth so in this super trivial example I'm just using the use date hook from react to set a value for something called dark mode you'll notice that I've got a



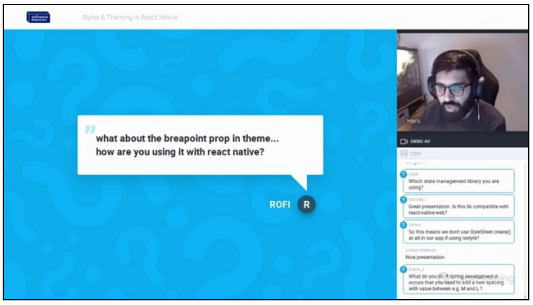
switch within my app that just allows you to toggle the value of dark mode to true and false and then the only other change that we need to make is previously our theme providers theme prop was just taking in the theme that we had created instead now we need to pass in dark theme or theme based on the value of the dark mode variable that we created dark mode state variable and that's literally it so all of your restyle components like the the box component you're using the text



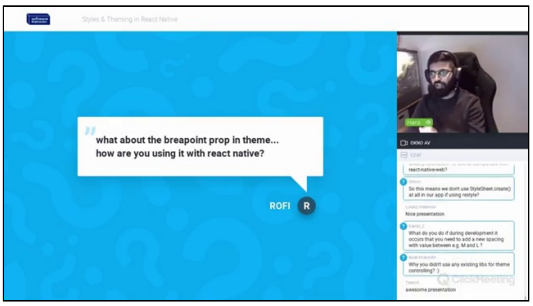
component that you're using will automatically adjust to the the the



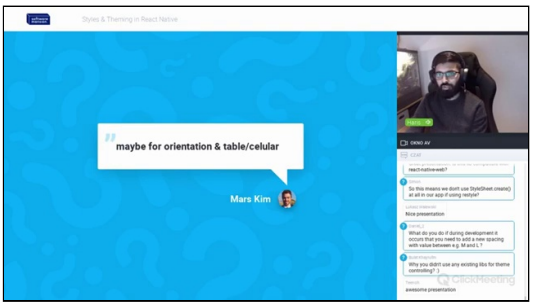
currently selected theme automatically



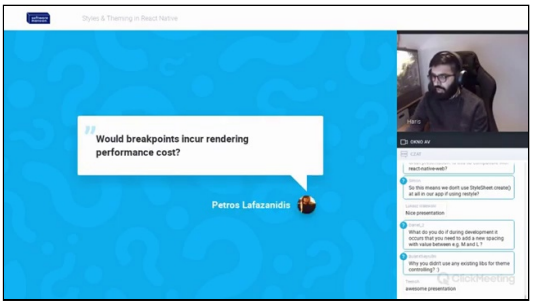
and you now have dark theme or an



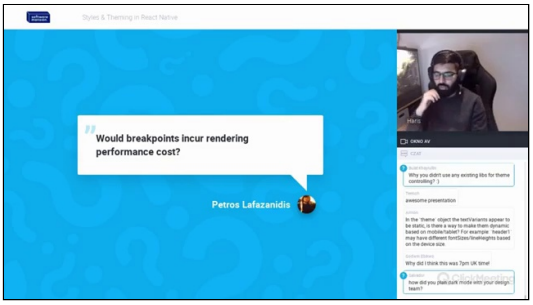
alternate theme baked into your app now what you do with that value of the



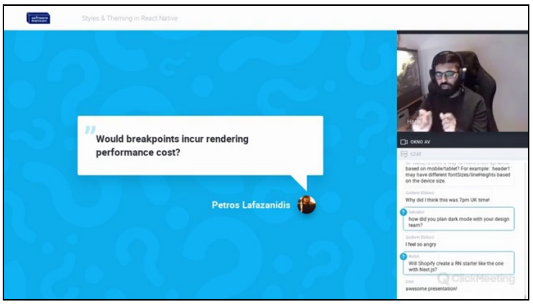
current theme is up to you whether it's whether it's just a state variable or you're putting it into the persisted state or whatever is that you're doing with that that's entirely up to you and very dependent on how you build your your app but how I just wanted to demo how that ties in to actually toggling the theme for the actual app itself so



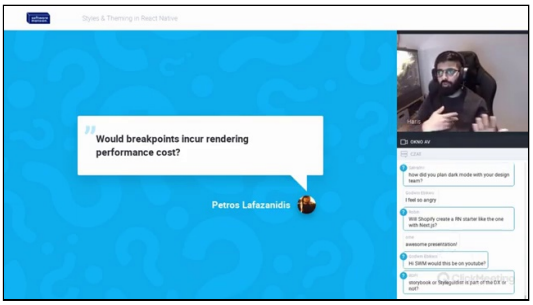
restyle also provides the ability for you to handle



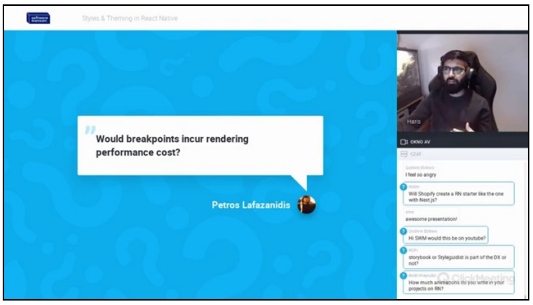
responsive responsiveness so let's say you need to accommodate various phone



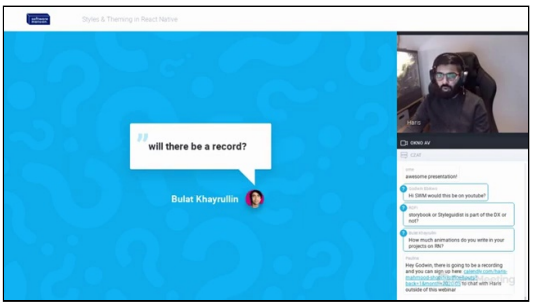
sizes so something that we can do is that any prop that restyle optionally accepts or resell accepts optionally accepts a value for each screen size so you can define first of all define these different screen sizes as breakpoints in your theme so in a new breakpoints object within my theme I can say that hey this is the minimum size for a phone and this is the minimum size for a tablet you can also introduce other ones



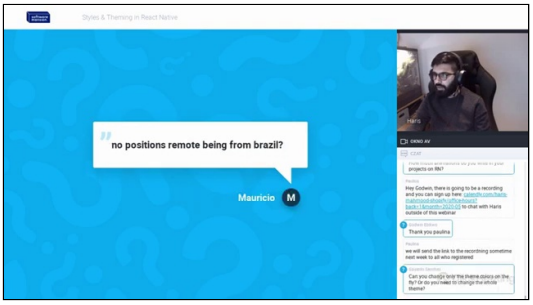
as necessary so you can so below that you can see that you can either just provide a value for a style prop so flex direction row or you can pass it an object with key value pairs with the keys being the different breakpoint values so I can say that on a phone for the phone breakpoint I want the Flex Direction to be a column and on tablet I want it to switch to a row so this works with any prop that you can pass into box or text and you can easily adjust sizes layout margins padding's whatever it is that you might need to to be able to



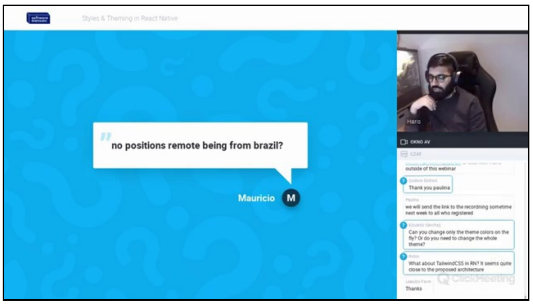
accommodate your your app to those



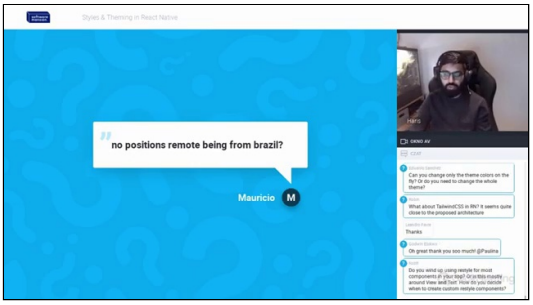
various range of sizes the restyle library also provides a you steam hook there's obviously will be a number of scenarios or use cases where you can't



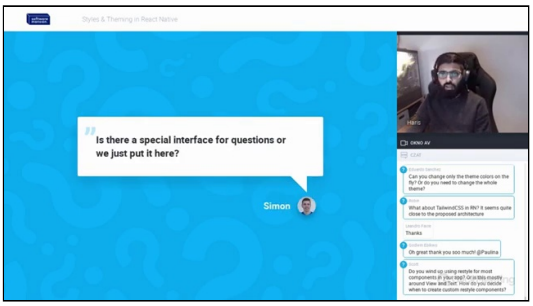
use the text and box components so for example if you're using something like react vacation to implement navigation across your app and you're using the default headers have come about you can use the



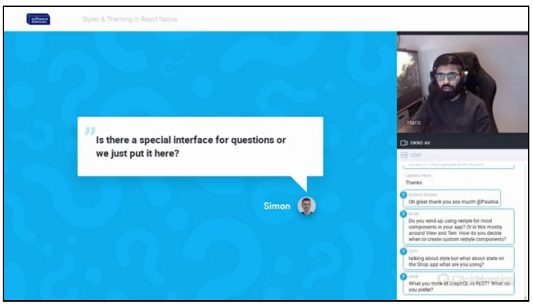
use theme hook to determine or to pull in your theme and pluck out the



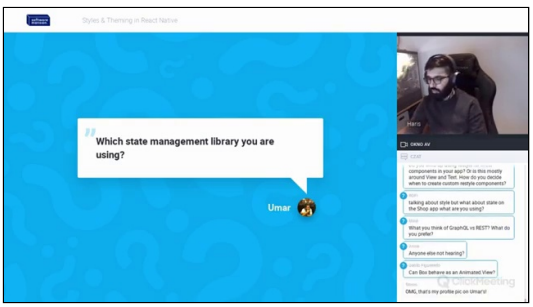
necessary color of values and pass those in to the react to the headers and the idea any other component that doesn't use the text or box components from from restyle and the the one thing to also



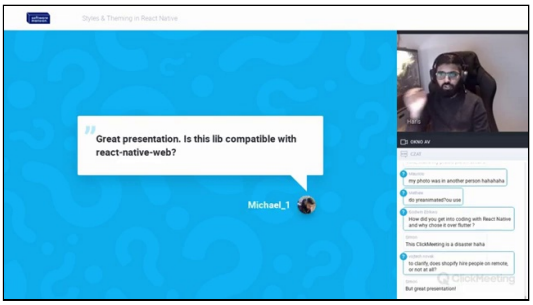
mention is the overall developer experience now given that all of this is



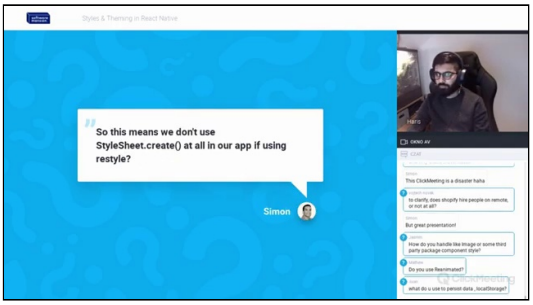
tied to using typescript if you're using



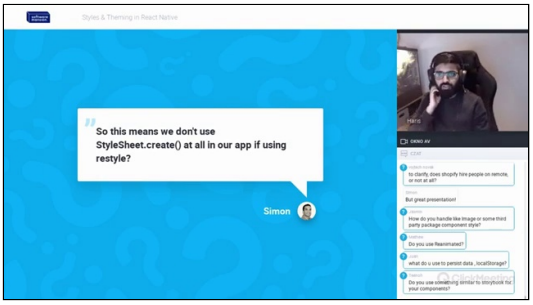
hopefully a a text editor that uses or is powered or utilize this typescript you get all the type enforced benefits and the awesome dev experience that typescript provides so you get all of



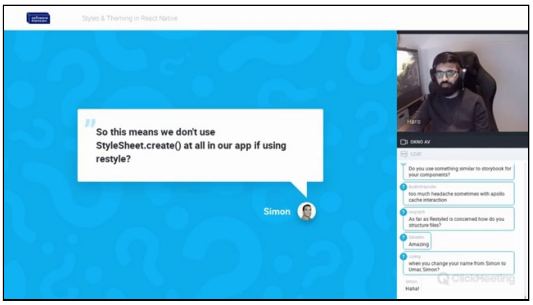
the awesome autocomplete functionality when you're building out a building on an app so you'll notice that as I'm typing and adding in these different properties for different styles it's showing me all the options that I have which immediately like I can't even express to you how the the efficiency in



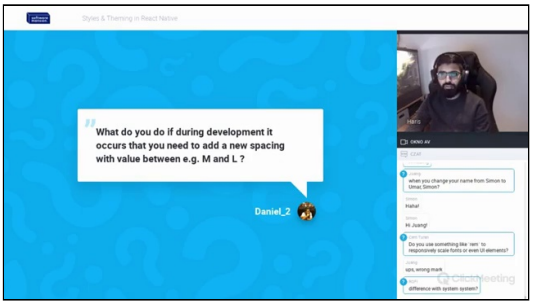
which you'll be able to start building your apps once your theme and the two starting components are defined now there's also a ton of other features that we don't necessarily have a chance to get to today I wanted to keep this talk focused on the absolute core and minimum features that like I think absolutely highlight the power of



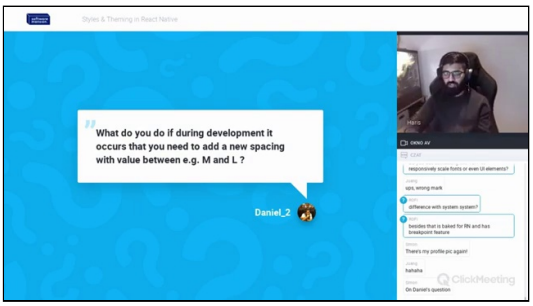
restyle and some of the the new



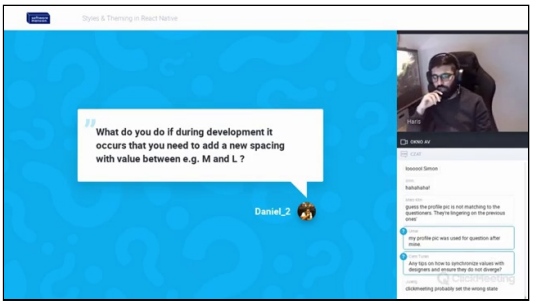
processes that we've established within our team so the two that can quickly mention is that Rhys off for example allows you to create your own custom restyle components so for example let's say that you know a button component isn't similar to a box or a text let's say it's got a lot more nuanced behind-the-scenes you can create a custom button component that is powered by restyle and all the benefits that restyle comes with unique to your app similarly you can create a custom card component that might have like shadows and rounded borders or whatever else you can either build that on top of the box



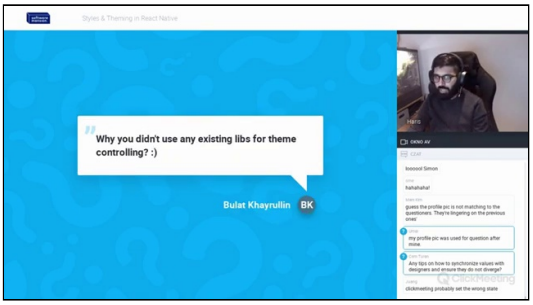
component or create a completely restyle component restyle also provides a bunch of functions that help you



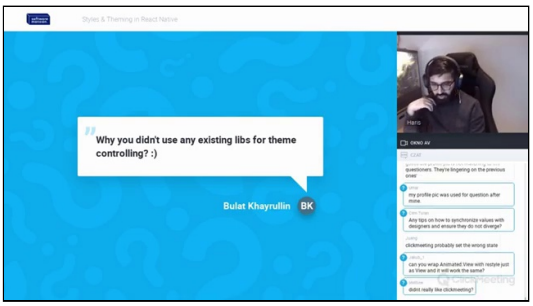
control what some of these custom components are able to do and not do so for example the the text component that you creates through restyle isn't able to take on layout properties isn't able to take on things that are like border



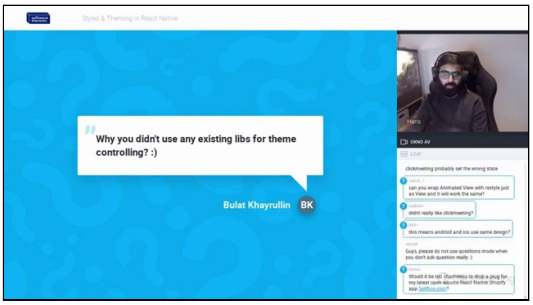
properties for example because those are specific to the box component so but let's say you want to create your own unique component that needed to be you can control which specific properties become exposed to that unique component so those are some of the features that some of the additional features that resaw provides there's a whole bunch of others and i recommend checking out the docs if you're curious to see what those look like awesome so thanks so much for for having me and for listening once again check out restyle on github if you're curious to learn more about some of the functionality and see some more demos and whatnot



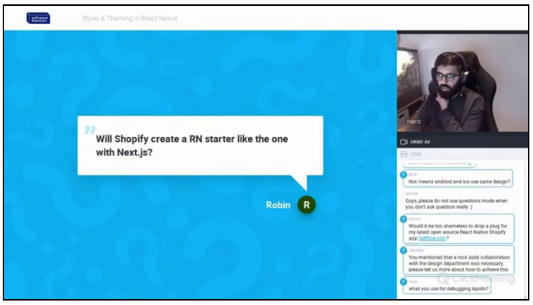
curious about Shopify curse about careers definitely check out the Shopify comm slash careers slash kovat 19 page



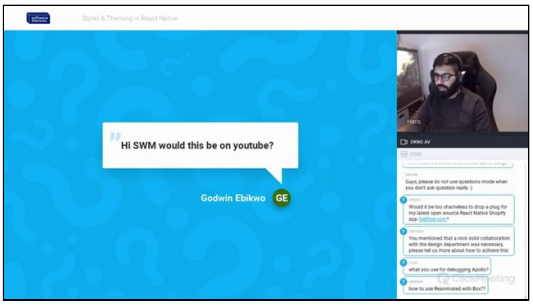
and our general careers page as well to see what current what positions are currently open um there's two more things also there I've set out a number



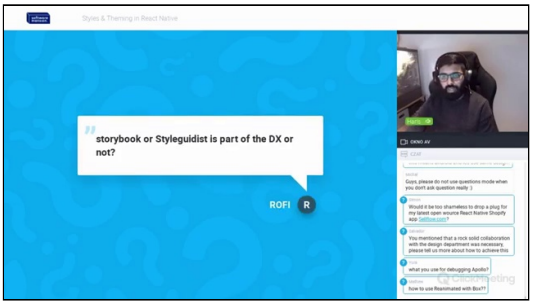
of hours tomorrow for office hours so that's the link there and I'm sure it'll be shared but to all of you as well at some point or it already has been feel free to take it down there as well so there's a number of open spots for you to book if you'd like their child about Shopify if you'd like to chat about career stuff if you'd like to chat about restyle or anything react native particular as well book a timeslot and weekend like have like a lengthier



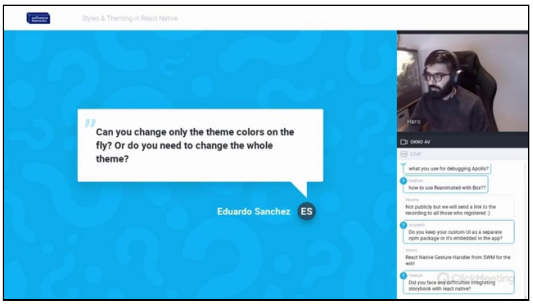
conversation to discuss things there after this I'm going to switch over to Q&A mode we'll have like a good chunk of time to answer some Q&A questions and anything that doesn't get answered here I'll be make sure to continue to send out answers or my replies through Twitter and also feel free to like follow me on Twitter as well to ask any additional questions that you might have there so we can continue the conversation going and we would like absolutely love to hear what your feedback is about restyle and what you can do with it Cheers so I am going to switch back over to to click meaning I'm going to stop sharing awesome so the one of the



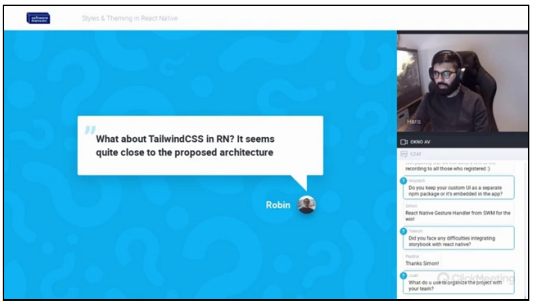
questions was what about the breakpoint propping theme how are you using it with the react native I understand I showed that early on but didn't get a chance to explain it until later on in these slides so yep you just provide an entire object with the keys being the the breakpoint names and then the value that you wants to apply at that specific



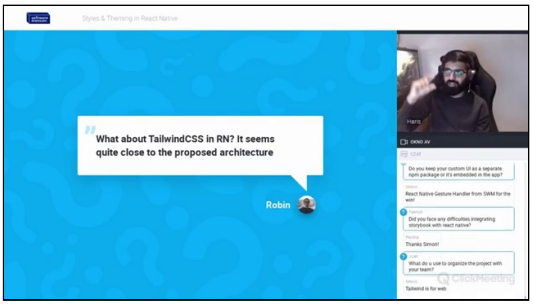
breakpoint so hopefully seeing that allowed you to understand what was going on awesome maybe so next up maybe before orient



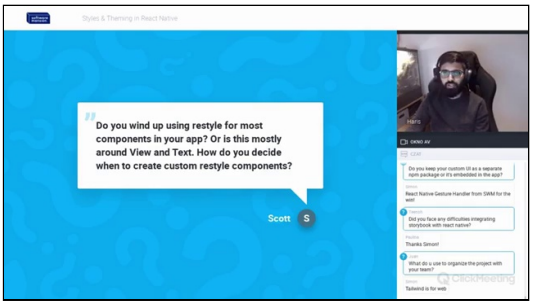
orientation and table-cell I'm assuming that was more so an answer for the previous question would breakpoints incur rendering performance costs so we haven't noticed any issues at all so the main place where we end up using a lot of breakpoints is that we don't really support tablet officially what we do instead is adjust our Styles based on the device size of the mobile phone size we've noticed that you can have like a super small iPhone to a pretty hefty Android device and like we weren't able to have like this magical one-size-fits-all like UI so we were using breakpoints to to adjust those different sizes and adjust those variables depending on what your device size was or what your phone size was we haven't noticed any performance issues at all so that's been pretty neat and we can obviously do a lot more like performance benchmarking to determine if



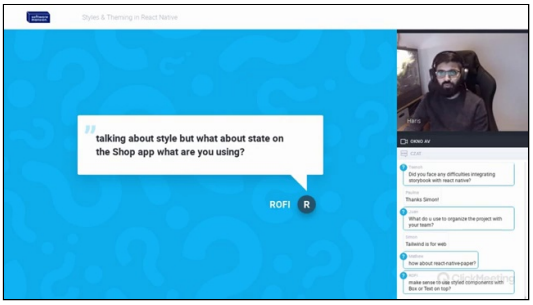
what what costs there might be if there is one but from our initial experiments our initial like usage we haven't noticed anything at all and the one thing to mention here also is that restyle is like fully baked into the shop app relies entirely on restyle and has been for a number of months now so we implemented it a little while ago when we introduced dark mode officially for arrived and have been using it ever since and we were essentially building this hidden theme along with other functionality for shop and we essentially just had to like flip the switch for for all of our users to be able to get access to the awesome new shop shop theme another question will vary will there be a recording yes I believe that all the participants for who attended here and who will be attending future webinars will be receiving a link at some point next week awesome so another question here is no positions remote being from Brazil I'd



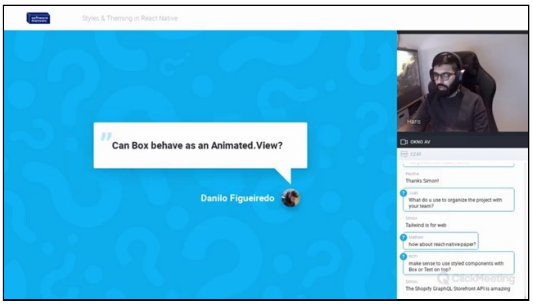
encourage you to continue to check check back we are hiring obviously the listings that we have there will continue to evolve as new positions open up feel free to to send an email out to someone as well or to the main couriers email address to find out a little bit more about if and when



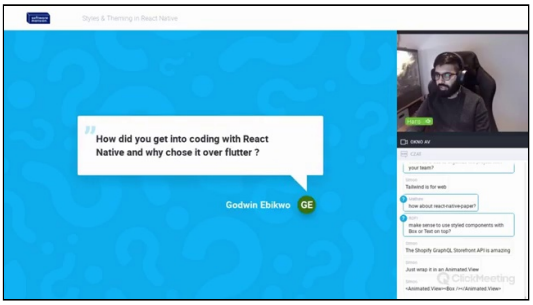
positions might be opening up that would accommodate your particular scenario yeah how you recommend that oh I think yes I mean you're you're killing it just



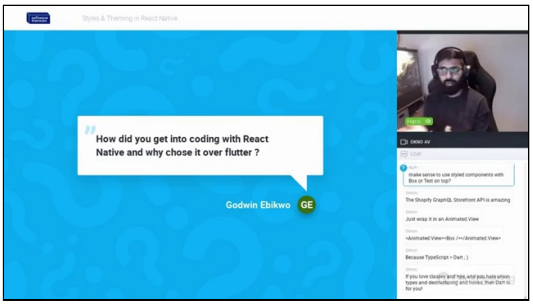
drop questions in there and I'll keep popping them up here as necessary another question from Omar which state management library are you using so we rely on so for all of our graphical needs we rely on Apollo Apollo cash is what we use to to store our state and we use context to handle most of our state changes and just update Apollo cash we don't use Redux we don't use mob X or anything else like that another question this one's from Michael one great presentation thank you so much is this library compatible with react native web that's an excellent question I don't see why it wouldn't be but we haven't personally tried we don't have any projects like the shop team isn't working with react native web so I'm not 100% sure but I don't see why it wouldn't would be very curious to try it out and maybe we can get back to you on that as well next question so this means that from Simon so this means we don't use style sheet duck create at all in our app if using restyle yeah that's correct so unless you absolutely needs to there's no place where you would need to create or use the style sheet API because all of the your styles are being passed in all your style values are being passed in as props to your box and text components along with any other custom components that you create so this was definitely super weird when you first get started it seems kind of strange but I guarantee you you're gonna love it it you'll have significantly less code the styles that you're applying are in line bit like you know are you'll be reading styles where they're used not at the bottom of your component file or



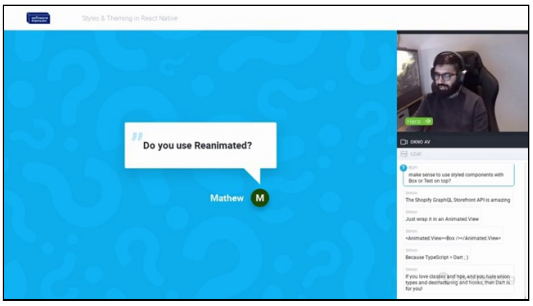
whatever else so just ends up making for a really really nice developer experience by just not having to rely on this style sheet API excellent next question from Daniel - what do you do if during development it occurs that you needs to add a new spacing with values between medium and large yeah that's a great question so I think this comes back to having a really well-defined design system in place ideally if your design stuff is done properly and your design team is utilizing those same spacing values a new value between margin and between medium and large should never magically just occur



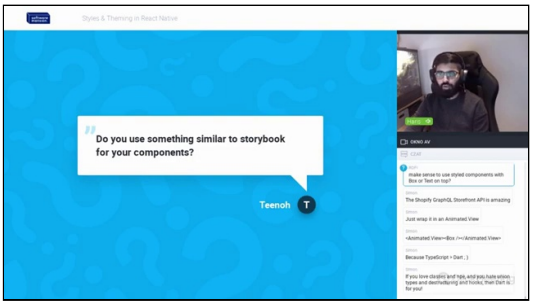
sometimes you know if a design is evolving yes that might happen and that's something to take into account for example the spacing values in the in the shop rebrand might have been different than the arrived default styles as in that case we had like a brand new set of spacing values that we were able to use for that theme but more particularly requestion if you did meets to add new values you would have to go through the exercise for example renaming large 2xl and



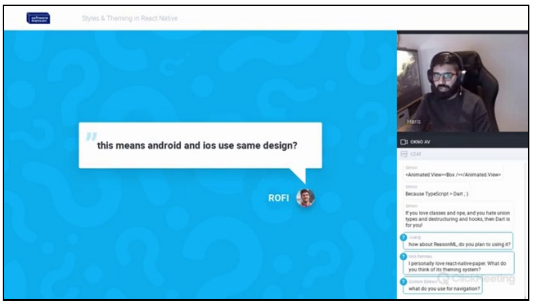
energy and introducing that new value in the middle and making sure that you're like those values get updated across the board that will be definitely a little bit tedious but ideally like if you have your design stuff figured out ahead of time you'll be able to avoid that as much as possible the next question is from balot I hope I'm pronouncing your name right why folks didn't use any existing libraries for a theme controlling a great question so we did a bunch of exploring we looked at a bunch of alternatives and things that existed already but honestly nothing was hitting all of the checkmarks all of the the criteria that we had all the problems that we were experiencing we weren't able to find something that checked off all of those had solutions for all of those things while providing the developer experience that the team would be happy with so we ended up needing to to essentially roll our own system that worked well for us and because we loved it so much we thought it would be an awesome opportunity to like open source it and give it back to the community as well to see if other people would enjoy using it too next question is from Salvador how did you plan dark mode with your design team yeah that's a that's a good question as well fortunately our design team had experience working with stuff like that they definitely like took the time to go through all the different screens all the components to figure out what they needed to look like in dark mode one of the first things that they did was updated the palette the color



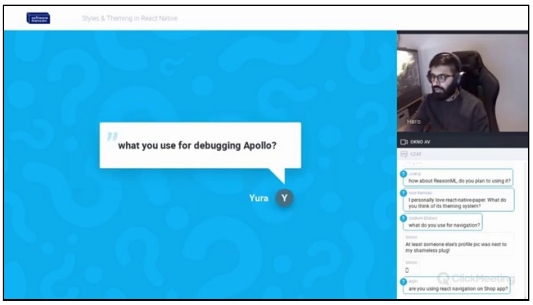
palettes to introduce the alternatives that they would potentially use for dark mode and then created the necessary like duplicated components and screens with the dark darker values or alternative values in place and that's what we used to define our new theme and then it became like almost trivial to switch things out to for the new theme that we created next question is from Robin will shop if I create a react native starter like the one with next Jess that's an awesome question we haven't considered doing that yet but if there is value and if there's interest in having like a starting Korea react native project that has restyled baked right in from the get-go that's nothing something we can look into and I'll be happy to chat with the team to see what we can do in other questions from Godwin high sulfur Manchin would this be on youtube I will let someone from Software mansion answer that in the chat next question is from Rafi storybook or Stalag eidest is part of the developer experience for not a great question we do use storybook



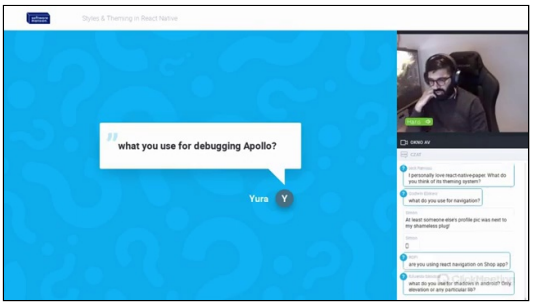
heavily we're allowing the story book entirely that's where we start off where we have a dedicated story that's just all of our different colors and all of our typography elements and then all the



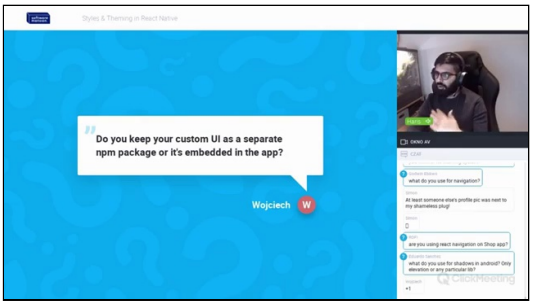
different components that we've created and we've built in the functionality to be able to toggle the theme back and forth in story book as well so as you're developing a new component you can go into settings toggle over to dark mode and then make sure that your component looks okay or test it out immediately there as well next question is from a lot as well how much animations do you write in your projects on react native so the shop app in particular has tons of animations we use the reanimated library extensively which is why we're working so closely with software mansion to further develop it and you know bring new versions of it out into the future yeah we're heavily invested in animations across the board I highly encourage you to check out that entire experience the onboarding flow itself has lots of animations so definitely definitely check that out next question is from Eduardo can you change only the theme colors on the fly or do you need to change the whole theme so you could create an entirely new theme that so you'll notice that when I was creating the dark theme the first thing I did was I used like the the three dot to spread the entire base theme so what that does is immediately duplicates all the values that the main theme has and then I just over like then I just like add in the overrides for the feeb colors and if your app looks like layout wise spacing wise the exact same between light mode and dark mode then that is all you need to do you just need to provide the color overrides in your dark theme now sure it's called a theme but all of your values are identical to your base theme except for the new colors that you've provided as overrides and that's really all we did to introduce dark modes to the arrive app back when Shopp was was arrived only when it came to introducing shop do we get into some of the other new degree details to change a lot of the other styles like the the font fare the text variants and some of the spacing values and some of the typography values and a whole bunch



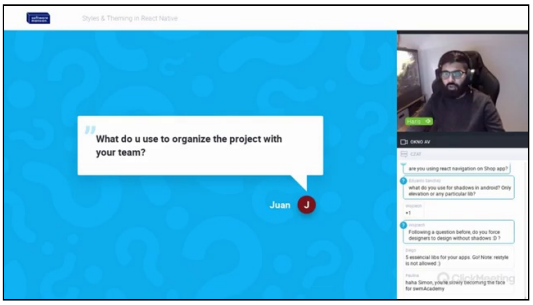
of other things as well hmm another question from Robin what about tailwind CSS in react native it seems quite close to the proposed architecture yeah you're right till wind does something very similar to death similar to this we just sort of had like very specific things that we wanted to implement ourselves there wasn't necessarily something that existed already for for react for react native so this is what our team came up with obviously it adds like a couple unique features that pertain to our specific problems or react native problems that tailwind doesn't necessarily address we felt that this worked pretty pretty well for us Thanks question from Scott do you wind up using Restall for most components in your app or is it them or is it mostly around view and text how do you decide when to create custom Restall components great questions so for the first one do you wind up using reefs off for most sutra components in your app yes we do like practically everything like 98% of our UI is users we sell across the board there might be a couple of small instances where we just have like in line like we just use the style prop provide different values that we need to grab from like animation values for example animated like position values or something like that but when it comes to stylistic things we try and rely on we sell as much as as much as we possibly can so anytime we're not able to use



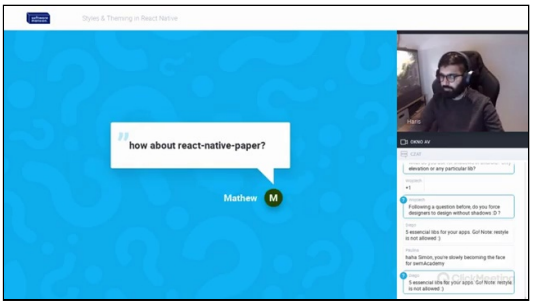
resale values or like let's say we're not able to use like the box and text components that we create through restyle we get to use the use theme hook that resaw provides and so we can pull



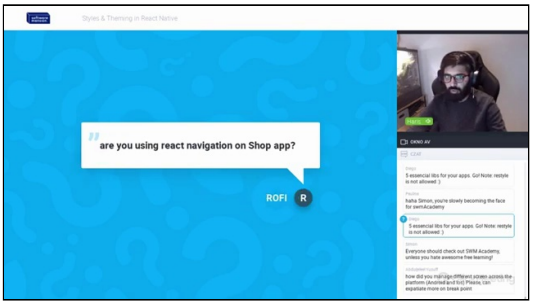
in those theme values from there and apply those as necessary so quite often if you're using like a third-party library or that you know I don't know like builds a component like a slider or something like that that might have its own set of ways that you need to provide colors for example so that might be a place where you can use the use theme



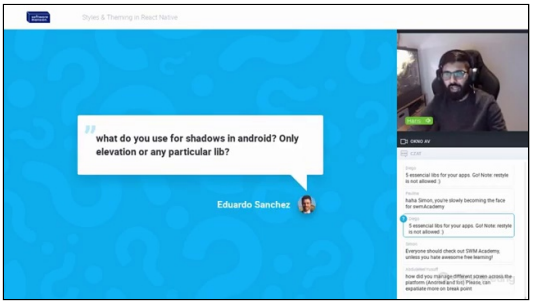
hook pull in the net be the current theme pulling those color values and pass those in to to whatever the API needs are for that component and how do you decide when to create custom Razak components so this is so this is a good question as well so any time where we had a component that required a lot of unique functionality a lot of maybe like dynamic styling needs as well for I think like the button comes into mind because you have to accommodate different states with regards to default state disabled State loading state maybe a couple of others lots of different variants as well so that might be a great opportunity as soon as you realize that hey things are getting a little crazy while trying to do this with just the default box or text component that's an awesome opportunity to create a custom component there's a bunch of



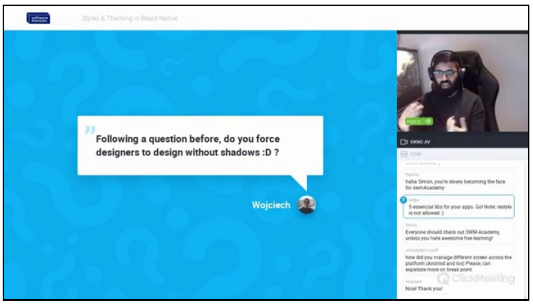
additional details about this in the readme for resale on github so I recommend checking that out as well next question for Murphy talking about style but what about State on the shop app what are you think so already got a chance to mention that we're just managing state using context where we can and our our state sort of lives in



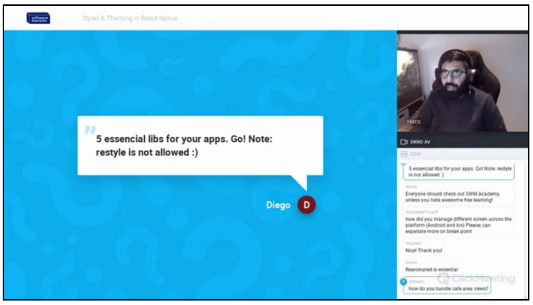
Apollo cash and we just update those values that's necessary we don't use anything like Redux or question from Omar what do you think of graph QL versus rest what do you prefer Shopify has been a long-standing supporter and developer in the graph QL community we're like super gung-ho and very enthusiastic about graph QL we use graph qlex extensively across Shopify



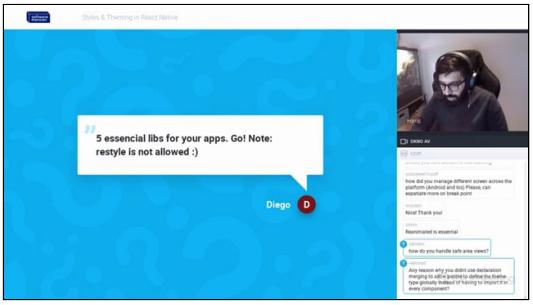
and our app is no exception so we heavily use graph QL and they use Apollo



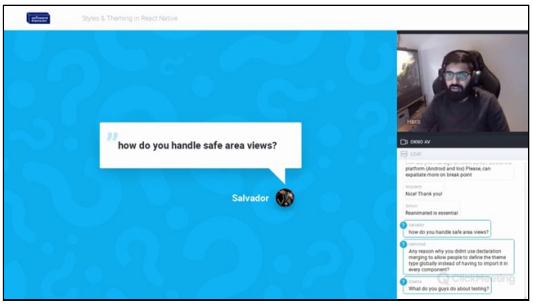
as our graph G library can box behave as an animated view this question is from Danilo I believe it can you'll have to we'll have to double check the docs because I don't fully remember if that's the case or not regardless if it doesn't then that would be an opportunity where you would just use the default view component from from react native to handle any animated related things but if I remember correctly I believe it does question from Matthew do do you use



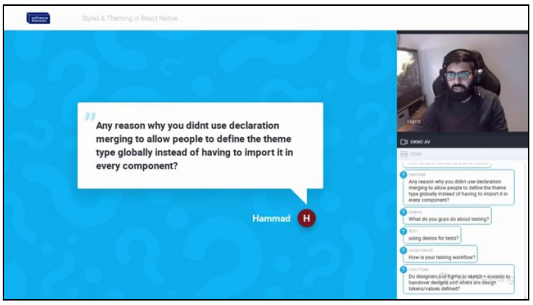
reanimated I'm assuming that's the question and yes we do we use it quite extensively next question from Godwin how did you get into coding with react native and why choose it over flutter a good question so a couple things here firstly Shopify isn't stopping using native like native development when there's a time and a place and it's obvious to use native development that is the route that we will go with but for the most part if react native makes sense we will



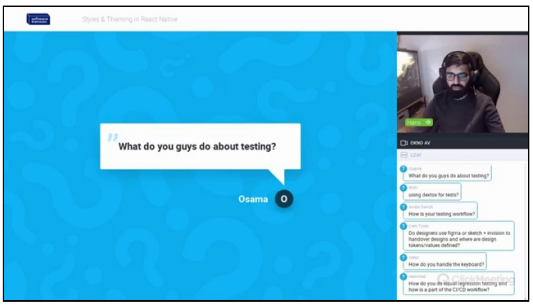
use react native that is like the default approach for new projects that



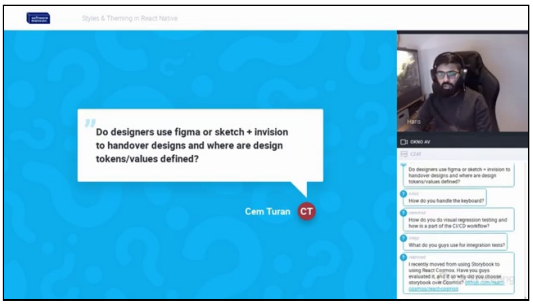
would start to Shopify and now why we chose flood react native versus flutter or anything else Shopify has is a huge we've invested heavily in react and the react ecosystem we rely on react heavily in our front end react as our front end stack so we already have so many developers at the company with the react expertise so to to prove



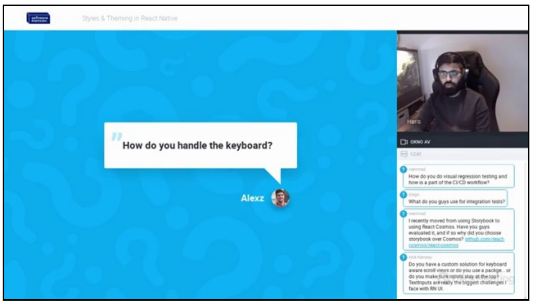
that to illustrate this point even further why react native was the better choice for Shopify in particular was that like I myself am NOT a mobile developer like I my background is a comes from web land like I was a primarily like a UX developer then did a



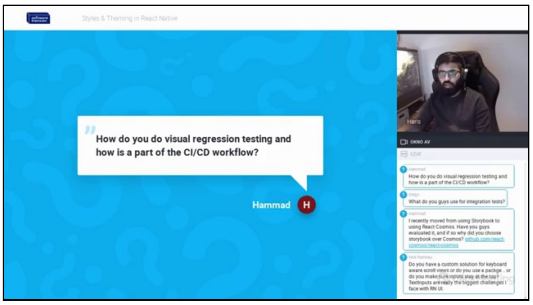
bunch of JavaScript did a bunch of back-end like rail stuff for a while so I had no mobile experience so one of the things that we experimented with or tried early on in the app was like hey let's build this app unless explicitly bling bring on someone who has no mobile experience whatsoever to see what their onboarding experience is like and that sort of how when I came into the react native picture I'd Shopify I knew nothing about iOS I knew nothing about iOS development and nothing about Android development and I was tossed into this to say hey how quickly can someone with react experience with web experience with no mobile experience for



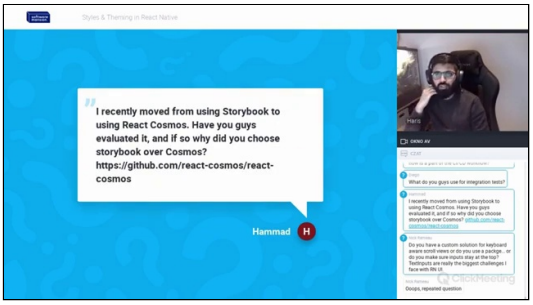
ramp up in a react native context and within like three to four days I was contributing to a mobile app and it was the coolest experience possible and I've been with that ever since next question from Wojtek I hope I pronounced your name right to clarify to Shopify hire people on remote or not at all Shopify does hire remote it's not for all teams there's specific teams that hire for remote for workers so I definitely check out the careers page because it will clearly identify which ones are for remote and which ones are not question from jasmine how do you handle image or some other third-party package component style so image so with regards to images so quite often the only thing that you're applying for images are so I think the the image component is something that you have an option for you can either create a custom resell component that handles images you can then pass in style specific properties to the image component or you can leave the image component as is that's a you don't use a lot of images and you're fine with dealing them in like the old-school wait that's fine too quite a lot of times we'll try and



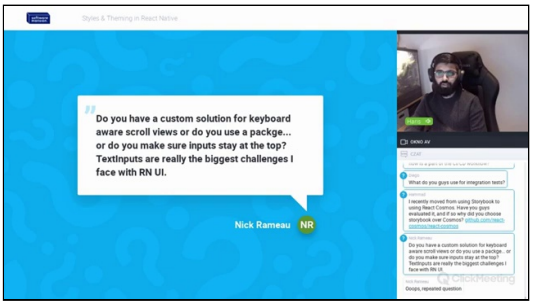
utilize the Box components to sort of fine the layout and styling or slice or like the sizing for the inner image component as well so that's a way that you can handle that or you can just use the use theme hook and bringing the theme values around the color values and sizing values from there and apply them to the image component another question



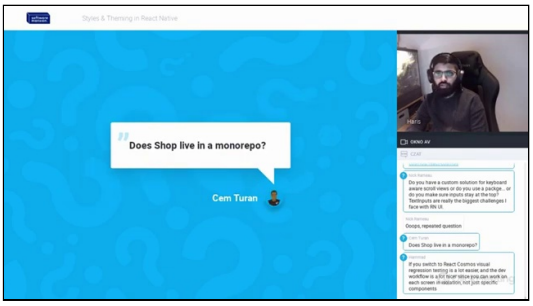
from Matthew used reanimated yes we do we use lots of it what do you use to persist data we use Apollo to and



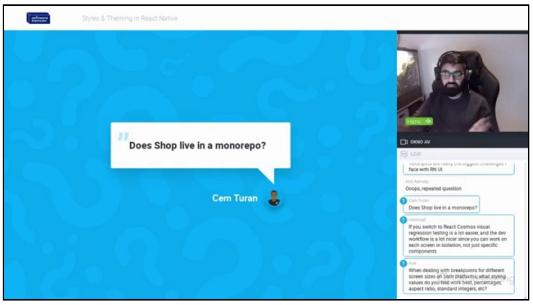
persist at storage to handle our persistent storage needs another question from Tina do you use something similar to storybook for your components yes we definitely do next question is as far as restyled is concerned how do you structure your files so we have a components directory where we have our default box component we have our default text component any other Restall components that we would create and aside from that we have any other standalone components like a card or a button or are like an announcement card or something like that all of those are individual components in like a in a components folder and then we have dedicated I guess

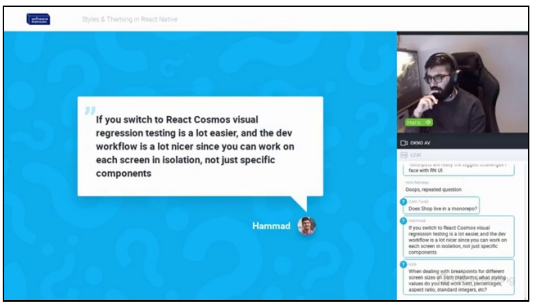


where we have our screens and views where we get to use those components and sort of essentially like stitch those together to create those scrape those screens next from chem or semi I hope I'm pronouncing

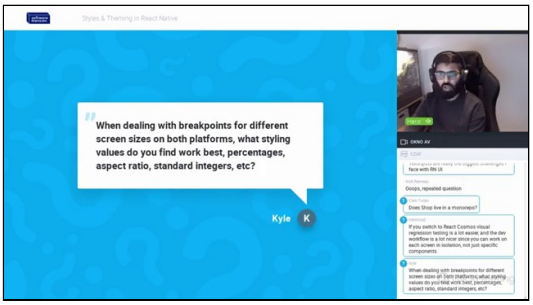


it right do you use something like REM to responsibly scale fonts or even UI elements no we do not we just rely on the breakpoints to define any overrides that we need to create in sizing I've previously experimented with a couple of libraries that will dynamically resize values as like the as the device changes they've worked okay they're a little bit harder to control so we felt that's something that was a bit more explicit in the way we manage sizing seems to work a little bit better with us another

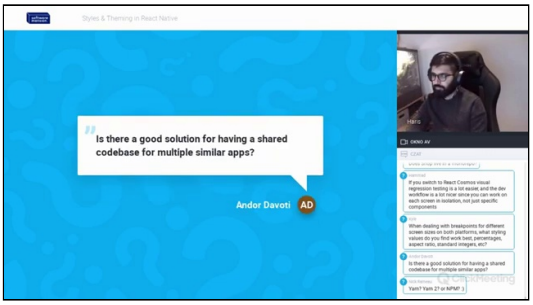




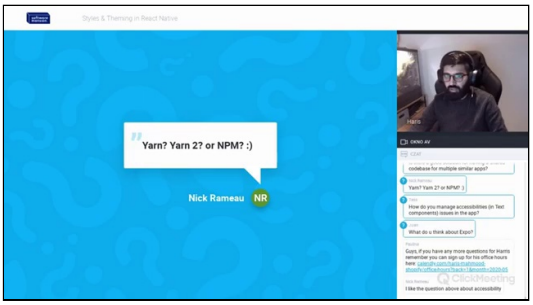
question any tips on how to synchronize values with designers and ensure that they do not diverge yeah this is a great question you needs to have first and foremost you need to have an awesome working relationship with your design team you



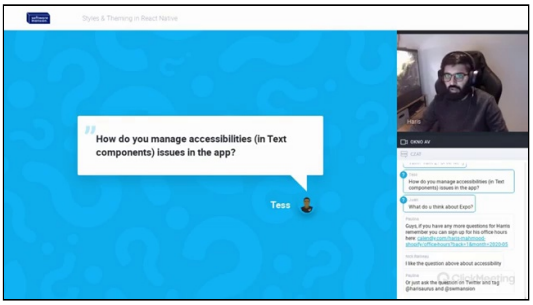
have to work well with them you have to understand them be friends with them and honestly I cannot express to you how valuable that is and how big of a role that plays with your app having less bugs at the end of the day communication needs to be top-notch I work with my designers I talk with our designers every single day to work on what's



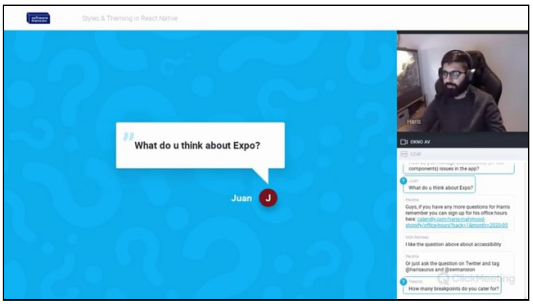
happening any synchronization issues that do emerge I immediately sync up with them try and figure out a way that we can stay within the established patterns if we do need to introduce a new value or a new color we have an actual conversation and it's not just okay I guess we're adding a new one in everything is like very thought through and we don't just like willy-nilly add new things into our themes next question from Rafi this means Android and iOS used the same design they do for the most part there are a couple of instances where we changed the way styles are applied based on the current platform that you're on the header for



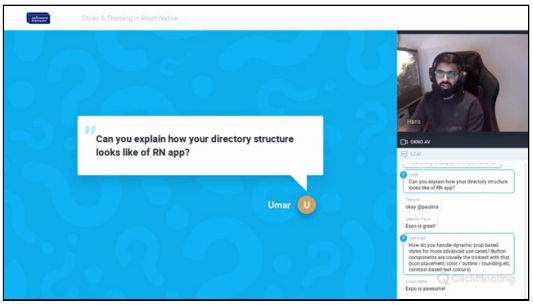
example feel a little bit more native based on the specific platforms and there's like the buttons also like really use the ripple effect versus the opacity feedback effect on iOS so those are handled the normal way that you would are using the platform API and then they just turn to reach X next question from Simon would it be too shameless to drop a plug for my latest open source reactor Shopify app cell flow comm I will leave it as that



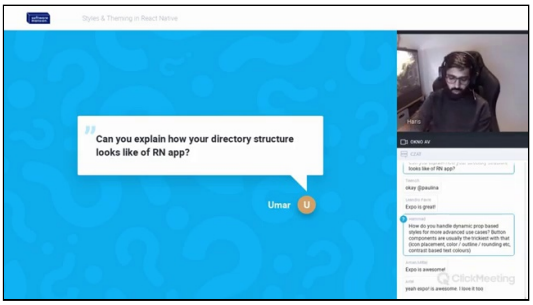
salvadore you mentioned that a rock-solid collaboration with the design department was necessary please tell us more about how to achieve this yes I kind of spoke about this already if you have more specific questions regarding



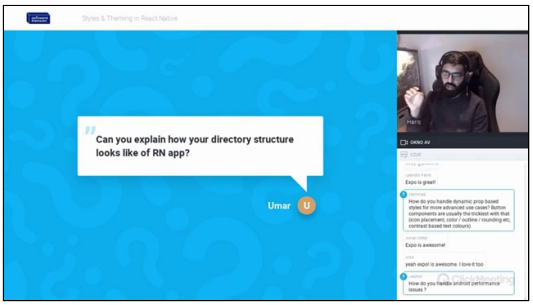
that I'm happy to answer that the one thing I would recommend also I guess is to depending on what tool your design team uses have access to those tools as well like have an opportunity or have space and like you know the the relationship to be able to give feedback early on in the design phase of an app to like you know the the design team takes feedback from developers very seriously they like listen to us we have an awesome working relationship and we work like super collaboratively like and part of that goes into an education aspect too so like instead of just



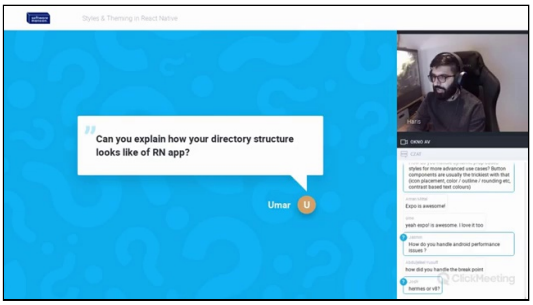
saying hey no we can't do this explain to them why something can't be done help them understand so that they have that knowledge base as they're continuing to evolve and develop and new screens that will go a super long way to make the entire experience of developing way better what do you use for debugging Apollo so we use react native debugger the standalone debugger tool is to like to check all sorts of like debugging related things I believe there's a couple of other standalone apps that just like help you keep track of like the requests that are being made and whatnot we also have like graphical built into our graphical endpoint so that allows us to check for those sorts of things if there's a more specific question of regarding that like feel



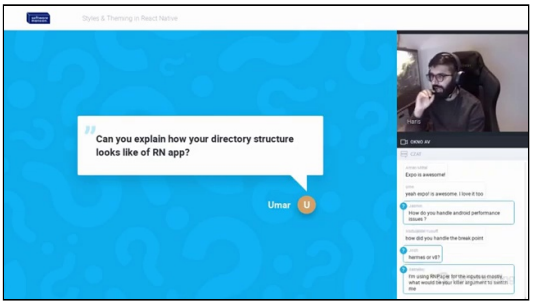
free to ask I'm skipping over a couple questions that were already asked another question



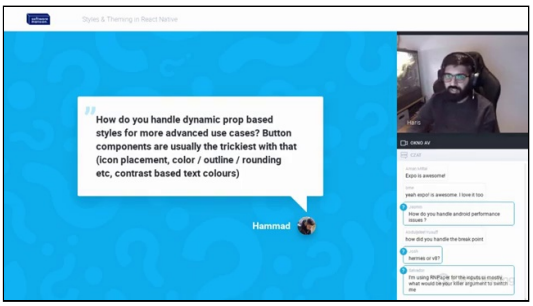
so I'm just gonna butcher your name so I'm not even going to try do you keep your custom UI as they separate NPM package or is it embedded in the app we keep our components embedded in our app so far so the different projects that we



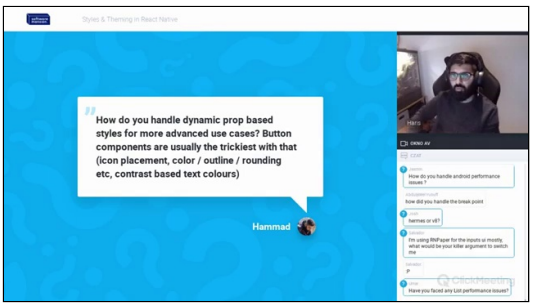
have your projects that are currently going on at Shopify don't have like a unified UI yet like shop has a very standalone aesthetic it's like our buyer focused product that looks very different than the rest of Shopify so there hasn't been a need for it to be packaged up and be reused in other places if and when that comes up and that's something that we would definitely be able to do question 15 oh did you face any difficulties integrating storybook with react native no we did not

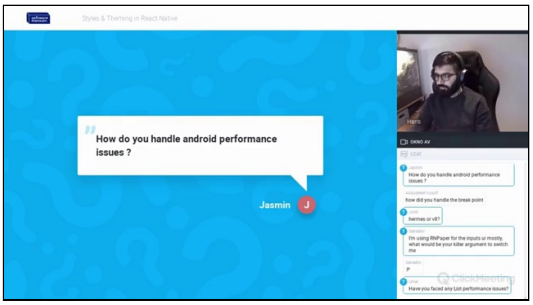


if you have any specific things that are

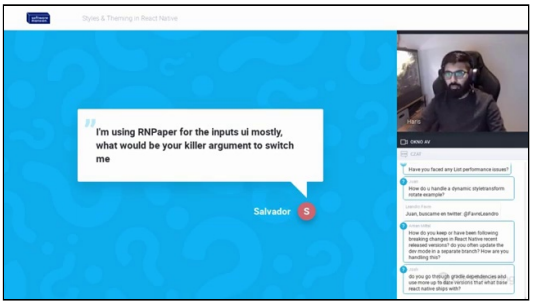


that your any issues that you're facing feel free to book us some time tomorrow or chat or send me a message on on

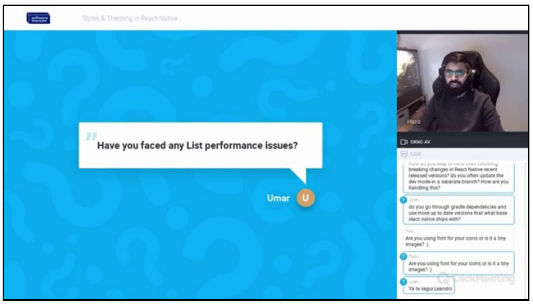




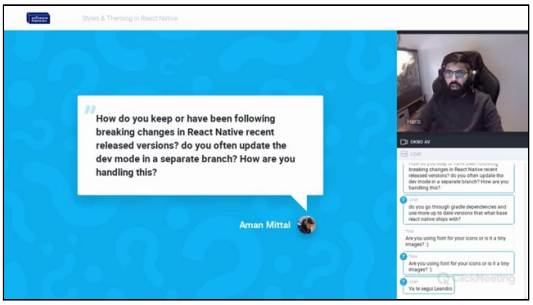
Twitter we can check further question from Kwan or what do you use to organize the project with your team we use github projects we have an add-on called Zen hub that helps manage github projects a little bit more so that's what we use from a development standpoint I'm not entirely sure if the design team uses anything in particular but the design team like we do have access to the design tools that they use how about react native paper a question for Matthew we haven't had a chance to use it I've used it on a couple of personal projects but it just has like a default aesthetic by diva in itself and that didn't match the aesthetic that we were



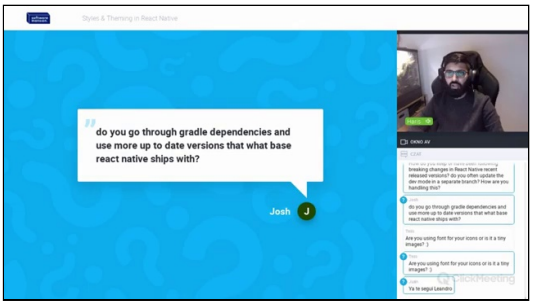
unique UI experience or UX experience for ios and android for shop so that didn't necessarily make sense for us what do you use for react native question sorry would you use for navigation question from godwyn use reactive navigation and follow-up from Rafi are using react navigation yes we are question from Eduardo what are you use for shadows and Android only elevation or any particular libraries we just use allocation next question good question do you force designers to design without shadows no we do not that would be super mean we there's obviously an understood



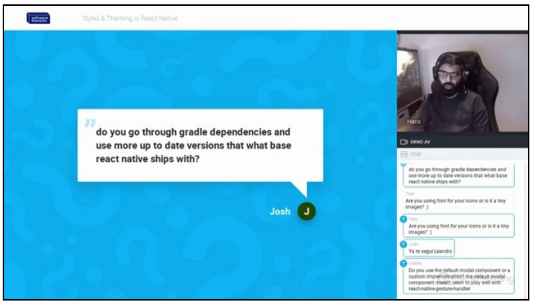
there's a understood notion that shadows that they would like they don't have to redesign the same cover screen or that has shadows or component that has shadows for both iOS and for Android they'll design it predominantly for just one and I'll have the design designer come over and we can just like play around with those specific numbers for the other platform so that so just to make sure that like it feels right and it looks right and that's like pretty much all we do we like sit pretty close to our design team when we're in the office when we're remote we chat with them pretty often we do lots of Q&A and like lots of internal releases so



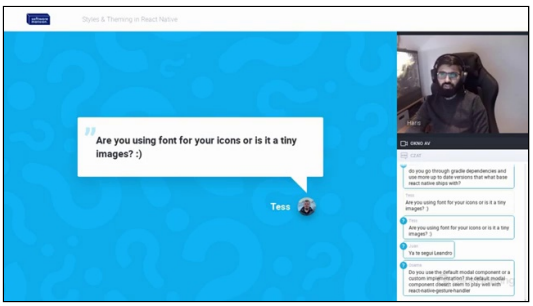
everyone can like get a chance to test things out on our app question from Diego five essential libraries for your apps go no tree style is not allowed well that's definitely not fair but okay react navigation is definitely our go-to for navigation stuff they recently released a few months ago their newest



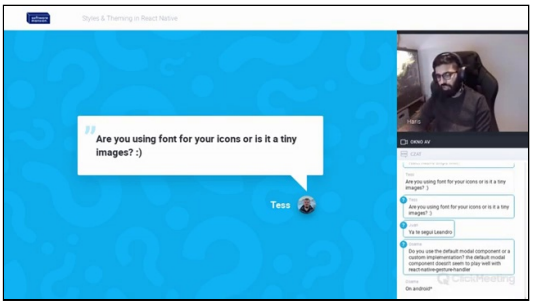
version which looks awesome so like we definitely like that a lot Apollo for all things graph QL related since that's our Shopify like de-facto way of handling handling data api's reanimated for a lot of our animation needs honestly I can't really think of anything else we have a bunch of internal libraries that I can't really mention we use we have like a bunch of



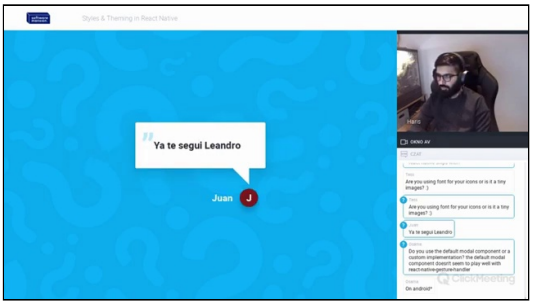
libraries to to deal with like tests and some internal graph QL stuff so I think those are the main ones that come to mind that I like repeatedly use over and over again yeah that's scholars to that question from Salvador how do you handle safe area of use just how you would normally based on the directions that are provided safer reviews just behave the way they normally do there's really like no no difference in the way the



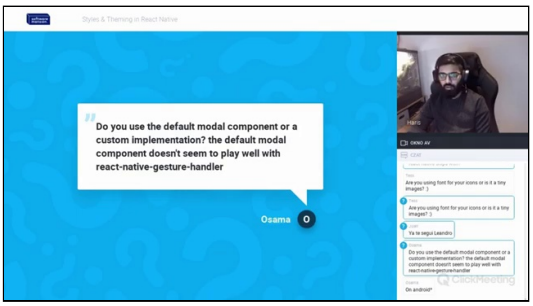
documentation for example would describe



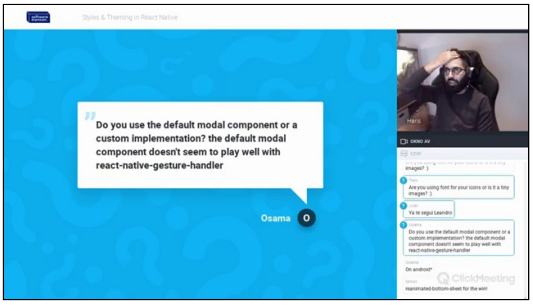
it question from hamad any reason why you didn't use declaration merging to allow people to define the theme type globally instead of having to import it in every component so you don't have to import the theme into every component you only have to import it where you're creating your restyle components so we just import it where you have we're



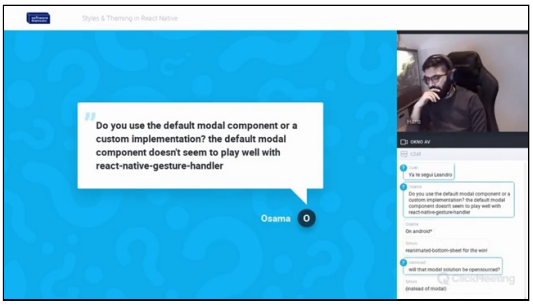
define the Box component or the text component or if you end up yes like that those like the only only places we actually have to do that and as long as



we're using the box or text components or any other real R component you have access to the the theme type to the



theme object what do you guys you would you guys do about testing from Osama we have lots of tests we do lots of integration tests we do lots of unit tests we heavily test our app we also have a pretty big QA process that our team follows so our product team and



design team also play a big role in making sure that our apps look and work the way they do story book plays a big