EG New Developers Working Group (2024-08-21 15:02 GMT-4) – Transcript

Attendees

Christine Morgan, Evergreen Community Development Initiative, Jane Sandberg, Jason Etheridge, Martha Driscoll, Michele Morgan, Mike Rylander, Mike Rylander's Presentation, Stephanie Leary, Steven Mayo, Susan Morrison, Terran McCanna, Tim Means

Transcript

Terran McCanna: Good afternoon, everybody. Welcome to the new developers working group. today we have Mike Rylander with us who has consented to talk about flushing data. This was at the request of Steven who has been struggling to learn it and it has made some progress but it reminded us all It's a topic that all of us would love to know more about the actual details and best practices of

Terran McCanna: with that I will turn it over to you Mike.

Mike Rylander: I didn't prepare much of a presentation. I'm more here hopefully to be good data source than anything else, but I guess the first question I have for the group is When you say data fleshing you mean when you're asking for information through the sea store or Pete Back end Services. How do you get? the object you asked for but also all the things that can hang off of it, right is that I see Terran you're nodding.

Steven Mayo: Hey I suppose so I didn't even realize that was what I was asking, but that's probably the case.

Mike Rylander: Okay, sure, then let's back up because I want to make sure I'm answering the questions. You actually want to ask. So Steven where in some of the code are you looking to flesh data and Not seeing or how to do that. Is there an example that we can walk that? I could try to walk you.

Steven Mayo: All I'll use the stuff that I did recently that I did kind of find it out for I sort of but I wanted to add a new column to a grid that is in angularjs. And it looks like the way to do that was to add a new EG grid field. Elements or…

Mike Rylander: Yes.

Steven Mayo: sub-element or something inside that grid and then you had to give it where it was, I guess fleshing in order for it to show the actual data that you wanted in that column. so somehow there's some process that I'm not fully aware of that is taking the little string of feel the mapper names separated by dots and actually getting stuff. That is what I wanted to see.

Mike Rylander: Yes, so that's actually pretty decent explanation of what happens. There's the magic and that go and the magic happens by looking at the fieldmapper table so I can share screen if it would be useful and the dev is Dev meeting. So why not right? Let's see.

Terran McCanna: Yeah, please do. There should be a box at the bottom of your screen with an arrow on it that allows you to share.

Mike Rylander: All right, so you should see. my

Mike Rylander: To enjoy infinity mirror do not share your entire screen. I don't share that again. And so

Mike Rylander: everybody's favorite oversized under documented l.xml, what grid Were you looking at the time Steven Universe?

Steven Mayo: money billing so I think ID equals MB

Mike Rylander: You are correct. It equals if it was the base money billing table. So was there a field on there that already existed that you wanted to bring into the grid? Or were you adding a new fuse to the technology?

Steven Mayo: No at the time. I was adding a new field that I had put in there. which I don't know how easy that's good be without checking out that Branch or something, but

00:05:00

Mike Rylander: there's a relatively High bar to adding columns to money billing.

Mike Rylander: There are a lot of reasons we went to try to avoid that if we can because it's used, in lots and lots of recording. If it changes aggregation, it changes grouping and aggregation for internal queries that we use. But let's set all that aside and just take a look at.

Mike Rylander: the society the question the use case here we're looking at is more of how do I add more fleshing information to a table? So were you looking to get To you was the colony we're adding. A column that had a foreign key. In it that you were linking off to another object so that you could more information in okay.

Steven Mayo: Yes. Yes, I was adding an extra person a person who created it which meant that it had to link to a user and then get their username and that was the part that was difficult to figure…

Mike Rylander: right

Steven Mayo: how to do.

Mike Rylander: so what I don't want to go through a full example of just yet of adding new column because we started the table then we add the information about the new column to the IDL. And then we, do other things higher up, but we already have a Column that one on that table the voider column. So we can use that as an example of how you might get information from a user that's linked to a billing not necessarily all Billings. But this particular, if the building is voided it should have a v Attached to it if it was voided with since voider existed.

Mike Rylander: So we've got this column here. Let's just go ahead and create one. Why not?

Mike Rylander: So we'll pretend that we Have actually created an upgrade script that does alter table money billing ad column. creator with a foreign key to act your user and we'll use that we'll leave that nuble in our imagination here. We'll leave that knowledgeable because we don't know the truth of who created Billings before that column existed. but for

Mike Rylander: Years. So we've got a new column on the table. we're describing in the IDL when you're creating a column that is meant to link to some other table.

Mike Rylander: You use the reporter data type link that the reporter namespace there?

Mike Rylander: Exists because one of the original purposes of the IDL is in fact, you drive the reporter and the link data type is it's a pseudo data type. It's saying I have data in me, but you don't care about what that data actually is. You care about the thing on the other side of a link. So we set that to reporter data type to link now. The system looks at the actual table definition at startup the sea store and Peak cred and storage applications inspect the table and they say, that's actually an integer or big end value. So I'll interpret it. I'll record internally that the real data type is an integer so that I know how to what the quoting rules are and all those sorts of things.

Mike Rylander: but at the application layer, we just need to know that it is a column that holds data that represents a link to something else. And then down in the links section, so that's in the fields. And links section. We would add a new link element.

Mike Rylander: We say the Creator field is the thing we care about it's a one-to-one relationship maybe the way into one relationships. this billing has a creator. The key on the other end is ID and the class on the other end is or after user? At this point your field mapper specific work is complete.

00:10:00

Terran McCanna: Mike can I ask the question? What is…

Mike Rylander: Sure good.

Terran McCanna: what is the map parameter for?

Mike Rylander: so looking here at the link for field adjustments. the wait, that doesn't have a map. let's find one that has a map.

Terran McCanna: Sorry, this is a tangent. I just never.

Mike Rylander: No, no,…

Mike Rylander: That's because this is all stuff that as you're adding things. It's good to know about.

Terran McCanna: Knew that question.

Mike Rylander: after let's see.

Mike Rylander: All right. I know we have some. stat cat links with a map because we go through.

Mike Rylander: The stat cat map to get to the actual entries.

Mike Rylander: Now it's being different. It's being hard to find. So here we go.

Mike Rylander: All right,

Mike Rylander: we've got

Mike Rylander: I don't do this often enough to remember the correct syntax. I don't think now.

Mike Rylander: Hey, there, we are. All right. So, yes, perfect. So we're on. class user AU and thank That was Stephanie. I think that Spotted that I appreciate you prompting me there so. the way that

Mike Rylander: the way that users are linked to permissions directly is through a user permission map. So you have users on one table permissions on another table then sitting between you have a map that says this user has this permission and it may have a whole bunch of those and we can load up the user permission Maps that's not difficult and it's not difficult to Traverse those necessarily but it's also a step that the computer ought to be able to figure out how to handle on its own and that's what the map attribute does is it says we're going to go through that pum pupm is permission user perm map. That's the table that behind pupm.

Mike Rylander: and the key that we're going to look up there is is users. So we're gonna say our primary key or ID is the same as the key user on the user permission map table. Once we've looked up the list of user permission Maps we can then say get what's behind the perm field on that table? and replace the instance of the user permap object with whatever's on the other side of the perm link on that middle table. So it pulls a list of user permission maps and then Overwrites each of those maps with just the permission object it lets you bounce through a table to a further remote one.

Terran McCanna: Them interesting. Thank you.

Steven Mayo: And so that's how you get something that is across a many many relationship because it Goes across the one to many that is in the map table.

00:15:00

Mike Rylander: Yes, so that's how and it works in addition to many. Anytime and it works on one to one as well. So anytime you need to kind of bounce through and intermediate table you can use the map attribute to describe how to do that.

Steven Mayo: Does it only bounce one? I mean it would probably be a very weird situation if you needed to do more than a couple but

Mike Rylander: Let us I don't remember is the immediate answer.

Steven Mayo: okay.

Mike Rylander: I believe you know what?

Mike Rylander: so the mapping link

Mike Rylander: does

Mike Rylander: You can bounce through multiple ones. So what it's doing is it's a space separated list. of the fields that you bounce through to get from you're starting object to the final object and the way it does that is it knows about all of the other links between all the tables. So as long as you have all of those set up properly it shifts the Context object over each time. and says

Mike Rylander: I'm looking at the user objects. I'll go get the user permit a map table entries that relate to it and shift that over and I'm looking at the bows and I say I'm supposed to map through to the term table. So I'll shift that over and there can be more intermediate steps in between I don't. Recall if there's any instances of that but it works by just looping through those steps. And so it will do it zero or more times as many times as it needs to looking up the interconnected pieces the way the tables connect based on the information in the ideal file.

Steven Mayo: It makes some real spaghetti with this.

Mike Rylander: Yes, but you only have to describe the spaghetti in one place and then everybody can untangle this big Eddie in the same way.

Mike Rylander: and the reason I opened up oils IDL course C this is where for the database backing applications or the ones most used the sea store in Peak credit apps. and reporter store the way it knows what the overall database structure looks like is based on the output of this.

Mike Rylander: This module. So it's a library component that knows how to interpret the IDL. So if you ever need a definitive answer of how does X work internally? This is the ultimate source of Truth.

Mike Rylander: So do we want to go back to?

Mike Rylander: I'm gonna go back and look at this a little bit more. adding a creator so

Terran McCanna: Sure.

Steven Mayo: yeah, there was something else that I had wanted to do that I didn't figure out how which so I like Because I was doing something over a one to many relationship like you you saw one billable transaction with a creator. but there could be more than one person who has Edited that billable transactions. I wanted to in the grid maybe a comma separated list of every person who has done something. but I had no clue how has many relationship in the field map area. It would return a data or

00:20:00

Steven Mayo: if it could in order to get a list of people to deduplicate or something.

Mike Rylander: so there's the user level use case that you're talking about who has touched this transaction and then there is the ways that we have available to us to store that information, so The last editor is what we store in the editor column on a lot of tables, but that's not enough. the

Mike Rylander: the

Mike Rylander: it depends on what you are actually trying to track because a billing gets added and then it can be voided or some other thing can come along and offset it there shouldn't be a case where you're deleting Billings in the normal courses of dealing with Billable transactions you should be offsetting. A billing amount with some other thing either the adjustments which we have a one-to-many link here for adjustments. voiding it and we already have the avoiding staff member.

Mike Rylander: But deleting and billing entry is something we really want to avoid. So. while I could imagine a case where we might want to be able to Market billing as deleted and say this person deleted it and its in an extreme case. There's other ways around it such as offsetting or voiding. Where we retain the fact that it existed. and we do something to the larger billable transaction to negate the effect of that billing line

Steven Mayo: he

Mike Rylander: That's a constraint of trying to. At least get close to having a double entry Ledger where you have debits and credits on a transaction.

Mike Rylander: If…

Steven Mayo:

Mike Rylander: if you're trying to yeah, I guess you need to look at the use case and say is this a use case that we need? once every 10 years because something got really messed up and we've got to go and manually fix it or is this something that we're constantly going to be doing because if we're constantly going to be doing it probably you need to figure out A way to address it. That doesn't upend what already exists.

Steven Mayo: and that was I think what I was hoping for since the information that I needed Was it's already there. So okay, if we have the creating staff member on every building line item and I want to in a grid for billable transactions show every creating staff member in one column as okay here is the set of all people who have made any building light items for any of the line items in this transaction. It's like going from transaction over a to a money billing here and I don't know how. it's

Steven Mayo: Does that do anything…

Mike Rylander: so the

Steven Mayo: but

Mike Rylander: It may not be best to use. This is a very interface specific. This decision that has to be made right?

Steven Mayo: Yeah.

Mike Rylander: So in this specific case, it may not be a great idea to Have all billing line items fleshed on every Transaction what we're trying to show the summary information. And the reason is there maybe hundreds or…

Steven Mayo: a lot

Mike Rylander: thousands of billing line items that come across. And Michele's making a great point. There is a view there that we already aggregate information about the transaction and the subordinate billing and payment line items. the materialized transaction summary table, so that would probably be a good way to do it.

00:25:00

Steven Mayo: and that is the thing that I didn't know existed.

Mike Rylander: and that would be

Mike Rylander: Aggregate the unique set of user IDs into a new column on that I say materialized for you. It's a table that gets updated by triggers. but our, hour in our free materialized views and in Evergreen, And then you would take that. list of IDs and You can get you …

Steven Mayo: And just display it.

Mike Rylander: or get the request the set of usernames for that list of user IDs or Whatever it is you want to show.

Mike Rylander: Or just the count of them perhaps but yeah, that would be a good efficient first step to doing that. However, let's say that this is a situation where we know it's gonna be fine. we're perfectly happy. Waiting for the billing line items to come in because we're going to need them. Anyway, when the

user clicks on a twisty that expands the transaction to show all the billing line items. So we're just gonna go ahead and finish them up front in this hypothetical extension In that case you would.

Mike Rylander: you wouldn't put a colum EEG grid column in there. That's just

Mike Rylander: The path to MB Creator you would probably use a template. that when it's drawn goes and fetches the list of unique users and displays a list of their names and an ally or something like that. So you wouldn't necessarily

Mike Rylander: just pass a path one to many aggregate object like that and a lot of the similar use cases

Mike Rylander: There will be a man or manage or view this subordinate object links in a column and you can do that. with it, you would do that with a grid template that says go to this other interface with this.

Mike Rylander: with this billable transaction as the key and give me all the billing line items so I can and then on that interface you're showing the list of billing line item creators There are a lot of the config interfaces that do that automatically when there's a one to many relationship between sets of objects.

Steven Mayo: so I think implicit in your answer of how that would work was something that makes sense in hindsight to me, but I hadn't considered just because I hadn't thought about how Any of it works in full, which is that when you tell it what column to show it has to go and fetch all of them. Which …

Mike Rylander: Great.

Steven Mayo: that would take time. Wouldn't it?

Mike Rylander: but if you're on an initiative you're sitting at an interface that is showing a list of billing line items based on a transaction ID, which we have the billing detail interface. And you wanted to show creating staff members username or barcode? Then you could do exactly what you were attempting to do on the transaction level, but you do it one step deeper. So on the billing line item grid you would put in as you add new EEG grid column with a of Creator period username, let's say And what would show up in that column the system would take a look at?

00:30:00

Mike Rylander: The path you gave it. Compared and use the IDL to derive what? kind of

Mike Rylander: What kind of query needs to be built? and when the interface gets the data back each of the Creator Fields instead of having an ID in it would have a fleshed user object and then it would pull the username off of that to display.

Terran McCanna: What are you talk a little bit about the differences in retrieving that between angularjs and angular?

Mike Rylander: There's not much of a difference. There's a pretty

Mike Rylander: pretty much the same in terms of

Mike Rylander: basic object grids you tell You can. Tell it which of the fields on the class to show which do ignore. And then you give for each of the columns you give each of the columns that you want to Define explicitly either because you want to be able to adjust the columns parameters whether it should start hidden. Those kinds of things that are easier to do when you have an explicit grid cell or grid colun. element

Mike Rylander: or when you need to do a path, that's not just the data elements directly on the object.

Mike Rylander: Then you give it an easy grid cell and are you Gigi grid column? I can't remember That's what adjacent code is for but you give it any good Define a new column and give it a path. They look very very similar in the template files.

Mike Rylander: if it's just a grid to show a basic object and it's an ideal grid that is one that you give it an ideal class and

Mike Rylander: and the data source is just figured out based on. the IDL it's not a custom or bespoke set of columns from various data. That's kind of mashed together on the client side, then it will be Pretty much the same we can. probably look at a couple me.

Mike Rylander: get rid of those changes and then let's see.

Terran McCanna: So in the grid itself is doing a lot of that work.

Mike Rylander: the grid could kind the behind the template. Yes.

Terran McCanna: but I've run into some cases. I'm trying to think of something up my head and I'm drawing a blank but where I'm not pulling it into a grid, but I'm just pulling a single value into a template.

Mike Rylander: Okay.

Terran McCanna: that I wish I could think of an example right now one of the time so then I usually have had to go into the Pearl to write to flush it out, and I always have to Foam bowl my way through that.

Terran McCanna: To try to get the syntax, right?

Mike Rylander: Figuring out what the Syntax for the peak red call would be anything sure so.

Terran McCanna: Yeah. Yeah. but remember a lot from the page that I just posted in chat, but

Mike Rylander: Let's take a look.

Terran McCanna: I still usually have to struggle with it a bit.

Mike Rylander: So I do and this is a little bit of editorialization.

Mike Rylander: But if we can avoid adjusting if we can avoid especially enhancing the angularjs pages. If there's an angular version.

00:35:00

Mike Rylander: It even experimental version or if it's something that you feel like you can report to angular. I want to make a case for doing it all in angular instead of angularjs if at all possible. But I know that's not always not always possible.

Mike Rylander: so let's take I'm looking at the example

Mike Rylander: the examples on the link that you send and this is all. pretty much angular specific

Mike Rylander: so that is a

Mike Rylander: This is a good reference. So. the

Mike Rylander: the Pearl or almost, the most Direct Services. Are the sea store in Peak Credit Services, you're not going to have access to see Generally speaking. from within the

Mike Rylander: staff client that's pretty much just for the Pearl side stuff, but pcrud is the same thing. but it expects an off token is the first parameter.

Mike Rylander: the peak red service inside of angular and angular Obvious states that away so it looks just like sea store. you can't do Json queries stuff through Peak red because checking permissions on that is It can't be mathematically proved to be correct. But other than Json query you have pretty much the full set of capabilities from C store.

Mike Rylander: that reference page

Mike Rylander: Does a pretty good job of explaining the high level stuff now? It doesn't go into some of the more advanced bits. Let's see. Has searching it doesn't have.

Mike Rylander: doesn't really say anything about Fleshing but that is something that secret can do so.

Mike Rylander: if we look at

Mike Rylander: let's see if I can find an example with some Flushing.

Mike Rylander: All Here we go.

Mike Rylander: here we will look at the patron Bill statement. components which it's a Jason to what we were talking about before.

Mike Rylander: Probably edit that.

Mike Rylander: So here's one where we are. when we are you'll see this peak Credit Service with the product the Property name Peak red but the service behind that is the peak road service.

Mike Rylander: You do have to.

Mike Rylander: You do have to load it from this in angular. But once it's loaded in the file, you can attach it through your Constructor to your object and then to your instance of the class and then make use of it.

Mike Rylander: in the methods of the class. So Here's a request where? We are asking for. a

Mike Rylander: page of results and we want the voider. On the money billing objects to be flesh.

Mike Rylander: This flesh here. The one does not just mean true. It's actually how deep we want to try to flesh objects. So this kind of keeps us from getting into a loop when you have treeish objects, like org units that have parents and children or you have users that share addresses And you can get into a loop this actually lets us set a boundary on the number you pass for flash. So it's a boundary on how many levels deep through the stack of objects that you're going to get back you actually want to continue fleshing out things. you can actually pass negative one and it'll flesh to a maximum. I think of a hundred but it'll go as far as it costs as it can.

00:40:00

Mike Rylander: Generally speaking though. You want to take a look at what you're trying to flesh and stop. Where you don't need anymore?

Mike Rylander: blush Fields takes an object where the keys on this hash are the types of objects and then the values are an array of the fields on that object that you want the system to figure out what the thing behind it is and replace the ID value in the field with the actual object

Steven Mayo: Is there a symbol you can do for get me all of them?

Mike Rylander: that is get me all that. I dont know. I don't know what you mean. let's let me that

Steven Mayo: If I for some reason needed all of the links An object at once and I was lazy.

Mike Rylander: No, there is not and that's intentional. if you are going to be asking for more data to be returned. you need to ask for directly. it's essentially matching. the semantics of a view definition in SQL so when you create a view in a database you can say so yeah, the view is defined as select slap from this table, but when the view is actually recorded it lists out all of the columns that existed when the view was created and if you have more columns to the underlying table The View does not change.

Mike Rylander: And same basic idea here if you need more data to come back to you then you need to ask Asking for a bunch of data that you're not going to use is I mean, it's just a pet. It's just penalizing the server and the user for a little bit of typing on the developers side and we'd rather burden the developer then the user in the server. That's the logic and argument behind that.

Mike Rylander: So you just list the fields that you want filled and it sends them back. There are more type Fields then voider billing line item. But we only need voider here because we already know what the transaction is. We've already got a transaction object that we're using to.

Mike Rylander: To pull the information in and it may not be in the exact same transaction object. if you have action circulation object in hand, and you say give me all the Billings for this object. You can say flesh the transaction, but you're not going to get back You're going to get back a generic billable transaction object. which won't have all the information you need on it anyway, so

Mike Rylander: when you're asking for fleshing. if you think of it purely as a graph where the The type you're asking for is at the top of the graph and everything else is below it. I can see why it makes give me everything under everything attached but in practice the way Evergreen has been built it's more of a tree than a pure graph. It's a directed graph and you're not going up. You're starting from the top and going down. Usually so when you ask for the money billing stuff and what's attached to it? You generally just want the things that are underneath it from the perspective of the directed graph

00:45:00

Mike Rylander: Does that make sense?

Terran McCanna: So for just the sake of argument, let's say at this point you wanted to get information about the voider lake. the name of the Reuters Home Library. How would you alter that?

Mike Rylander: sure, so if So for that so, you're going to have to go one step further than just voider. So you're gonna do two and then you pressing voider but the class that is behind void are this because you're the developer and you 've memorized the FM IDL that XML as we all have. Or you've got a wall that's wallpapered in it. No, I don't have that off camera. I actually wish I did that would be cool. But you would say I also need to get

Mike Rylander: of the actor user object

Mike Rylander: any actor user object that you also flesh that home OU field on it, which will get you the org unit ID.

Mike Rylander: if you wanted to

Mike Rylander: you wanted to know what the Orient type was for all of the units that you fleshed as and bye Windows around

Mike Rylander: then you can do that and you can say and then you would need to do this. You need to set flesh to three because you're gonna have to go three levels deep now.

Mike Rylander: That's so this is not just level two level three because it's not because they're in that order. It's just, you're going to have to That's one level to the voider at home on on the object that got inflated into the voider field. that's two and then you're gonna have to go a third level through the parent OU field on that to get the object underneath it. and then you would be able to say Things like, if you have that object here.

Mike Rylander: You'd be able to say that. so you stick that the return that the object becomes out of this you use your subscription or you two promise it to get the object back and the object you get back is going to be the end of the object. That's boo. You can go through voider, which is the use of the member of the staff member that boy did it to the parent overview of the homo that of the staff remember that boy did it to the short name of that? So that's how you would flesh multiple levels deep

Terran McCanna: So another question I have related to this is I don't understand the difference between Peak cred search and Peak red retrieve. Why?

Mike Rylander: Retreat that let's take a look at the peak red service.

Mike Rylander: And hopefully this will help it all make sense. So

Mike Rylander: so retrieve takes a class string and a prime and a primary key Either number or string. And it calls the peak credit retrieve method. with that primary Value and it's going to return exactly one thing. And it's always going to just give you one thing back. So retrieve is if you have the primary key of a thing in hand, and what type of thing that is you can say, give me the object of type thing with the key of Search you can do the same thing. You can say search for all of the things of this type with a primary key of this value. but

00:50:00

Mike Rylander: Searches meant for looking for things that you don't have the primary key of or where you have a list of the primary keys.

Mike Rylander: And you want to get a bunch of them at once you can say. Search where the ID is this list.

Mike Rylander: Is in this list but the retrieve parameter. is exactly one value and the search parameter is a hash or a Json object of key value pairs where the simpleware Clauses that you want to apply where the name is this or The idea is this or some string contains this value. That kind of thing. does that help?

Terran McCanna: Yeah, that makes a huge amount of sense. Thank you. So since search can be used to retrieve One is using retrieve more performance efficient.

Mike Rylander: It's probably not worth.

Terran McCanna: Because it's more specific to retrieve one.

Mike Rylander: Trying to measure the difference in speed. But if you know that you have an ID in hand and…

Terran McCanna: Okay.

Mike Rylander: that you want to deal with one thing. It's more expressive in the code to say ve. After user with this ID.

Mike Rylander: As opposed to search for actor users where ID is in this list of one thing.

Mike Rylander: and from a

Terran McCanna: Proposing what you're saying?

Mike Rylander: I don't know. I don't have anything worse following up that.

Terran McCanna: We're coming up on our hour just anybody else have any questions related to this that you want to ask Mike while we have them here or anything else you think of Mike that you think we should know.

Terran McCanna: for my sake I want to say I really appreciate this even though I've written some of these in the past and they've worked I think this makes a lot clearer for me. It's going to be a lot easier for me to do next time. And we're really appreciate you coming.

Mike Rylander: Yeah, of course. I always feel like these are super short and we don't get to get through a lot of detail.

Terran McCanna: Yeah, we like to focus on one little thing if possible so at least it's kind of a bite-sized conversation.

Mike Rylander: yeah, and If there's more follow-up to talk about in a later one, I'm happy to come back anytime.

Terran McCanna: Thank you so much. I'm going to go ahead and…

Mike Rylander: Yeah.

Terran McCanna: stop the recording. What's

Meeting ended after 00:54:46 👋