Hey guys, Soloman here again, this is part 3 of our video.

To recap, in part 1, I kind of walked you through the API Management highlights from the different portals that come as part of the API Management.

Part 2, I covered the REST APIs.

I walked through an example that I wrote that used REST APIs and I kind of tested it.

I did deploy that application in the WebSphere Application Server Liberty profile within Eclipse and we’ve tested it to make sure it does work.

And of course part 3 of this tutorial of end to end solution that takes you from Worklight to WebSphere Application Server APIs that are exposed through API Management, we’re going to basically take those APIs that we defined – which is one of them, really, the GetRates – I’m going to go ahead and create that API and expose it in API Management.

What I’m going to do is create multiple plans, one with limited access and the other one with unlimited access.

I’m going to call the unlimited one a gold plan and the limited one a silver plan.

We will then test it out when we get to that point to show you how you can see the APIs and how they are being used within the system through the analytics side.

If we look, for example, at the analytics part, where we can display how the APIs are being used, and we also can show you how you can put limits on their access.
For example, I will define them as five APIs per minute, and that means on the sixth call within a minute it's basically going to get rejected, and that's what we will basically show as part of this particular demo.

So let's get started.

The very first thing I would like to do – now remember I did create an organization that's called Soloman World Bank which is in here, and of course the owner of it is this barg0018@gmail.com and that is my email address.

So I'm going to go ahead in the environment, click on it, and I'm going to define a new environment for us.

The new environment I'm going to call it SolomanProd environment.

I'm going to make it restricted, such that not everybody can use it without approval.

And then for the URL path, which is basically like you see in here the SB is where things show, SWB is the URL path for the Soloman World Bank, and SB is the short name that you can add, and this one I'm just going to say SolPD.

Now I would like to bring something to your attention which you might run into.

Notice if I put upper case D in here and click add, I would get an error message that tells me that the short name field SLD used to prepare URL for the API is not valid.

Should be a valid DNS name.

What this tells you when you do see this error message, basically it all has to be lower case so this should be spd.

Beyond that I think we should be good.

We are not defining the gateways because we only have one, but if you happen to have different kind of gateways, for example, for the cluster that you want to use, one for production, one for development, this is where you would define them.
Let's go ahead and add this SolPD environment and now we have it.

This is the URL to access this particular environment.

So from a user registry perspective, like I mentioned earlier, I would continue to use the one sandbox.

If you want to use a different one just like we defined in the user registry in here, there is only one, but you can actually go ahead and create a new one if you want, a new user registry or a new local user registry that you want to just simply point to your own environment and use.

For the purpose of this, we are not going to create anything new, I'm just going to continue using the one and only, because like I said I have limited number of users which is myself only with email addresses, so I'm not going to make this too complex by creating too many email addresses, as there's no benefits really from showing you this.

One email address should be good enough.

So from a user's perspective right now, we have that user and, as I mentioned in part 1, when I defined it that Soloman or barg0018 is the organization owner, it's the administrator, the product manager, and the developer, so it's all in one you can separate them, but for the purpose of this I am not going to do the separation.

This is for real application in a real organization.

For now I'm just trying to get things going.

So the next step for us to do is under the consumers, as you see I don't have any consumers and I would like to define them now, because the consumers is the one you're going to basically go ahead and specify who is going to be using your application.

So I am just going to say this is Soloman B and I'm going to say the owner of this particular consumer, and I'm just going to use my IBM email address.

Just to make sure that I specify it, SolomanIBM, so I know it's an IBM account that I'm trying to use and I'm just going to go ahead and please add it.
And now I have this particular one created.

So next after this let’s go ahead and go to the plan section, because once you define an API to be able to test it you need to make sure you have a plan.

So these are the things you can prepare your system with from a plans perspective, from a consumers perspective, and users, this is kind of done once for the most part, and beyond that it’s just a matter of defining APIs and to start using what you have from a plans and consumers perspective.

So let’s go to the plans and as I mentioned earlier, I would like to define two plans, one that I’m going to refer to as the gold plan and the other one is going to be the silver plan.

This one I’m going to say this is my gold plan and I want access for this one, so I don’t want anybody to use it without actual approval, and this is going to say “Gold plan to access the Soloman World Bank API to get interest rate. No restriction exists for this plan.”

So add it, and also I would like to add another plan, and that plan would be called the silver plan, and there would be no accesses required and approvals and I’m just going to do “Silver plan at Soloman World Bank API to get interest rate,” so we got to say “Restrictions do exist for this plan: only five API calls a minute are allowed.”

So we’ve done that and as you see it’s straightforward and simple.

These are the two plans that I’ve created so I’m just going to go ahead with these two plans.

Now let’s go into the APIs.

If you notice under the plans we have the default plans that we created and obviously there’s the two environments that I have, I have the sandbox environment and the SolomanProd environment, and you’ll know what those mean in a minute when we get to those.

And of course as you see you can define multiple versions of it as well, so if you want to make some changes, I’m clicking on this API, I want to add the restriction, this is where you add it at this point I have no restriction.
But anyways we’ll get to this so you don’t have to worry about it just yet.

Let’s go to the APIs now.

So in the API case you have the option of defining either a REST API or a SOAP API.

And what I’m going to do right now is define a REST API because that’s what I specified within the WebSphere Application Server Liberty profile, that’s when I deployed my application for getting the rates.

I’m adding a new API, so click on add and this is going to be interestRatesByType, that’s my API, and I’m just going to say get rates.

And please make sure again it’s all lowercase and I’m going to say, “This is an API to return the interest rates based on type, the type of this API is going to be short, long or jumbo and the API also takes the first name of the user” and the reason I made it do the first name because when I deploy my Worklight application I would like to say Soloman requested this specific information and here is the rate that’s being returned to Soloman, or John, or whoever.

Click add.

Now we have the API.

I can go ahead and click on this particular API that I just created, and notice that from identifying the application using, you have the client ID, you have the client ID and secret, or none.

I’m going to define this one with a client ID and secret, and that basically means that when you define that API you will need the client ID and the secret that would be generated when we do the application and would be part of the URL that you actually issue.

Part of the test we’re going to do initially in part 3 of the video before we get it to Worklight is basically the ability to issue that request from a browser where you can see that I am talking to the gateway – I don’t even know WebSphere is located or the APIs are located, all that’s done behind the scene away from you, and I’m passing the information that would basically get me what I want, and it will pass that call on and it will keep track of what I’m actually trying to access.
It's going to be all apparent to you once we get that done.

So let's go ahead and specify one with the client ID and client secret and I'm going to go ahead and add the resource at this point.

Now go back to the URL so I can show you how you can do this.

So this is the URL when we call the method, and as you see we got it with type equals short or long or jumbo and first name equals whatever you want to add.

So I'm just going to go ahead and copy it from the place just before you do the question marks, so that's the records in this case, and I'm going to add this into the path and this is the path that I'm looking for.

As you see this is kind of also an example of how you can do this.

In this particular example we're having the other path – if you remember I showed you different ways of doing the JAX-RS or REST service, and this is kind of one example either as a parameter with a question mark, and then you do the type equals value, or you can actually do it as path parameters instead of query parameters, or both in this case, it's showing you the path parameter and here is the query parameter.

I'm just going to copy the record and it is irrelevant really what you specify in here, you can do anything, just put some default values, which is first name equals Soloman and short name, and just do the records.

For the display name I'm just going to say, "Get interest rate by type" and description, "Return the interest rate based on type."

We have done that at this point, and this isn't going to be a post, this is only a get, of course you have the option – you can have get, post, put, or delete, of course this is what REST allows you to do – so in my case I'm just going to get rid of the post and just do a get.

And I'm going to go ahead and do an add.

Now notice that we did do the add and now this is the method that I have.
I would like to, after you do this, to click on the edit button or this pencil here, and that will open up another tab for you, another section.

There is the overview section and as you see it will recognize that I have two parameters to type in, first name, and which you can enter some values, and it tells you if they are required or not – of course in my case they are required, so I'm going to say type is short and here I'm going to say this is Soloman for the first name.

Click on the implementation tab and that you will have a proxy or an assemble.

An assemble is when you start that in the request and response and whatever you want with it.

I will defer this to a different video that I will do where I will explain more of how you can assemble your request.

I would like to do it in a simple way right now – both are simple, but I'm just going to illustrate the whole end to end solution so it doesn't really matter to assemble in this.

Go ahead and click on the proxy and what I would like you to do, again go back to the URL where you defined it, and take everything up to an including the records.

Notice that what we're doing right now is we're basically including the records twice; the records is included in the URL and it is also included in here.

This is kind of the way to do this thing.

Go ahead and save it.

This thread start here indicates there are changes that need to be saved.

So once you save the changes -- now at this point you can also add some documentation, and the documentation you could either basically upload a file that's going to be displayed, or you can just simply point to a URL that somebody could click on that would point to your environment, and I think what I'm going to do right now is I will add a URL that goes into a read-me, HTML, or some welcome page that I have in my WebSphere Application Server environment into my application.
that basically talks about these particular APIs, and that's what I'm going to do right now.

Like I said, I would like to add that API, so what I just did I actually want to define it into my environment and in this case WebSphere, so I went ahead inside of WebSphere in my application this REST API demo and I created a new HTML file so I just right clicked on what content it should be, do new and you do an HTML file, or a web page for that matter, it gives you a blank web page.

And what I did after just literally add “Welcome to Soloman World Bank” and here is the API name and the parameters it takes, which is type and first name, and as you see those are the same exact APIs or I use the name that I just defined in here.

By the way, I did make a quick change to it just to put uppercase for interest and uppercase for R and B and T.

Now I would like to define it, so here is how you can actually access it just literally from the host port, the application name, and then this is what I have, the Soloman World Bank.

So I'm just going to copy this and put the URL in here.

And again from here the display name, and this going to be the “Soloman World Bank APIs” is what I'm going to have.

Type a description, it's optional, I'm just going to keep it at that.

I'm just going to go ahead and do add.

Now as you see I added the path to the documentation and you can obviously add more attachments and documentations as well.

One thing I ran into in here, if you do add it you will get prompted by IE if you want to open the file or save it that's coming from this particular host, feel free to just say save or open it doesn't really matter.

Sometimes you might run into an issue that seems as if it's hung, it's actually not hung, it did upload it, all that I recommend that you do at this point is just literally open a new tab window and copy the URL into it and just re-open it, and just close the other window that you had open, the browser window that is.
That's really all that you need to get that working.

Let's go ahead and at this point I think we're ready to test it out so what I'm going to do right now is, I would like to kind of show you both, again we've done the API with the client ID and secret, and we've added the actual API, the path which starts with the entry before the question mark like I mentioned earlier, and then the second one, which is really what we used in the URL when we defined or when we made the call as you see in here, everything up until the records are included, the record that we also start here with.

Let's go ahead and save this and as soon as that's done saving we should just go ahead and test it.

When I want to test it – this is the cool thing about the API Management – it will allow me to test it out.

So let me go ahead and open the other window, which is where the WebSphere Application Server is running at this point, because I want to make sure that as we make a call to this application server, we're going to find out – that's why I added the system out entries – so basically we can see where that call is happening.

So let's go ahead and do that.

Watch out for the system out at this point which is really what I care about.

I will use some different entries so this one we did short and name equals high, so what I'm going to do in here and then invoke, you'll see in here when we get into the test area the resource is not contained within a plan, so – this is a very important step – after you define the API that you want to use with the resource, and you define everything in it, now we need to make it part of a plan.

And the plan that we're going to make it part of is the basically the one that's going to allow us to, whether we're going to put any limitations on it or not, so let's go ahead and click on the plan at this point.

You will see under the draft plans, you will have the gold and silver and for now I would like to go ahead and deploy it into the silver one, because I want to test the limits also so click on the silver, and you will
see that right now we did not put any limitation – I told you that we're going to do this later.

So let's go ahead and change the rate limit from unlimited to make it five calls per one minute, this is the interval that I want, and after this I have reject calls when limit is reached.

The options you have is you could not reject the calls, and that's what is called a soft reject, which basically or soft limit is that rather than rejecting it we would simply receive an email – we as the owner, so Soloman would see an email saying this API exceeded the number.

So it's up to you how you want to define your business, but for now I would like to reject it just to be kind of illustrate how that works.

So I go ahead and applied it.

Now we add a resource, of course as you know you could change the limit per API, you could do it at the resource or entire plan level, it's up to you how you want to do this.

So let's go ahead and add the resource.

You will see that I have this resource that's already available, because I did that, and I'm just going to go ahead and add that particular resource.

Now what happened when we add the resource, so step number one, I defined the APIs, so after defining your environment and users, which is the thing that you do only once, now I go ahead and define my API or the resource, and as you see it's there right now.

Now the next step is I would like to deploy it.

If you look at the deploy button in here you see that it's not deployed anywhere, so we need to make sure that we deploy it.

Let's go ahead and do the deployment.

You can decide which version that you want to deploy.

In this case I only have one.
If you want to make different changes to your API, you could make them on the next version and when you're ready, you can make that one the production one, so you're not impacting any of your existing ones as you're doing your development and test on your newer one that you're working on.

This is very very nice to have actually.

So let's click on the deploy, and remember when you do deploy it's going to tell me which environment are you trying to deploy it to.

If you recall when we created the environments, there were two of them, there was the sandbox, then I created SolomonProd.

I'm just going to go ahead and deploy it to my sandbox for now – the reason I defined multiple of them is I want to make sure that you see it – and there is a disconnected cloud.

Now what this does – this is another important feature -- this allows you to export this particular package such that you can later import it for that particular plan or put it into your existing environment.

So if you're saying, well I want to do it to my sandbox but also, just in case if I crash my system, I want to make sure I have a backup of it, you can actually export it to what's called a disconnected cloud, and then you will basically be prompted to save it on your local disk.

So just go ahead and deploy it to the sandbox.

It says it has unsaved changes, so go ahead and save it, now let's go ahead and deploy it, we're going to do it to the sandbox.

OK success.

Now notice one thing, as soon as it got deployed, it went from undeployed into a sandbox, and look there's sort of a donut shape circle, what we have in here, which is basically green with a little bit of white in the middle.

What that means is that this particular API is now deployed but it's not published.
Now nobody has access to it until you publish it, so the next step for us is we are going to go ahead and publish that API.

So I'm going to go back to the plans where we had it, and instead of going to the draft plans, I'm going to go ahead to the deployed two, so I'm going to click on the sandbox in this case, because that's where I deployed it, and you will see that again it tells me there is an API that's deployed but not published, by the empty circle that you see in here.

So I click on silver and notice that now I have the ability to publish it.

Now before I publish it, I need to define the consumers because, I deployed it, but now I'm saying I have an API that's deployed, but that API, I haven't specified who is going to use it, whether it's available to the public or not.

So let's go ahead and click on the consumers.

Now notice that on the consumers I did define the member IBM API or IBM ID, and that IBM ID that I defined – when I did define it, this guy got sent an email, basically a welcome email to my IBM email account, with a link that allows me to click it; once you click that link it will actually bring you up where into the first name, the last name, and the password and I did already all that.

When you specify any new users make sure that you check your email address, because that's when you specify the actual password for that particular user and you can specify the name, which I did here it's called Soloman as first name and last name IBM, and I did that because I want to make sure I differentiate between my gmail and my IBM account.

In this case we have two options, public users or select groups.

If you do it as public users, what that means that, as soon as I have access to the application that I'm deploying, I can see the APIs that are available and I could call them if I want to, but I have to logon first before I am able to use them – versus if you make it select group, is if people logon to your website, they wouldn't even see the APIs until they actually logon.

And this is useful in cases where you're saying, I would like to expose some APIs to certain users but I have public APIs.
For example, if I have some gold service that's available only to my internal customers or some special users, obviously I don't want to expose that API outside, because this is something kind of a top secret for me that only you have to logon, and once you logon with your ID I can see, yup I trust you, now I can show you those special APIs, but for the rest of the world, I'm just going to expose the generic ones.

I hope that's very clear in how you can define public versus specifying select group.

So in this case, I'm going to go ahead and make it public.

So anyone can see this plan but I need to sign up to actually consume it.

So I'm just going to go ahead and publish it and as you see in here, as it is publishing, it's going to go into silver and then publish to sandbox, and now this has been published already, so I am just waiting for the resources, for everything to get loaded up, but at this point I am very much ready.

Now notice that this green now becomes solid green rather than what it used to be before.

And now I have the ability to make the calls.

Now let me go back, remember I've done all that because I want to make sure that I test my API.

I did make a mistake in this video, but hopefully you did catch it, which is, I just needed to deploy to test it, but in my case I also published it, so I did not have to publish it to be able to do the test of the API, because the whole point is you can deploy it and go back and test it, and then when you're happy with it, that's when you can go ahead and actually publish it.

But that's OK right now because I've already published it so I'm just going to go ahead and do the invoke.

Now notice the URL is what I have in here and I have the test and notice that I can specify the type, it's going to be long and for the first name I'm going to say Solo, just to make sure that I use a different one that what I have used in the past.
Now I want to go back to my WebSphere Application Server, in this case the output of the console.

Notice the last thing is the name equals high and it gives me this file not found and I got this error when I was trying to create the previous files, you can ignore that, it's irrelevant here.

So let's go ahead and invoke it.

So as you see, it did invoke it and it came back saying OK and it told me the first name is Solo and it says that this is a great kind of loan that I have, it's not the best, it's great because it's not jumbo, but it's not short either, and the type is long, and it told me that the rate is 4.375.

This is perfect, now let's look at WebSphere and see what WebSphere did.

Look what it did, it says, by the way here is the API, and it's says the type is long, it is indeed, and the 4.375, and the name is Solo.

So here you go, success.

So now I just tested my API, I exposed the API that existed on my WebSphere Application Server machine, which by the way, in this case, would be a completely different machine.

In my environment in here I actually have a virtual machine that is the API Management and I have my existing laptop which is my WebSphere Application Server environment.

And hopefully that illustrates what we're doing in here, now how I can test it out and everything works just great.

Let's go back right now – this is really most of what I wanted to show you as part of this particular part three, but I'm still not done just yet because now that I defined the API, I tested it, I have the plans created, now I would like to go ahead at this point and actually define the application that will be exposed to the public, where people can logon and take a look at the list of APIs and decide which APIs they want to use.

So let's do that.

The best way now to go to the third portal that we're talking about is if you go into your environment, and you will see that up here where it
says developer portal, it gives you the address which is what I would like you to capture.

Remember we deployed it to the sandbox, so this will give me swb/sv and that's exactly what we want to go to, and once we do that and continue, obviously when you run in your own production environment you will have the correct certificate defined.

Now look at this.

This is my website.

This is exactly what I make available to these developers who are interested in using my APIs.

It says Welcome to the Soloman World Bank developer center and it tells you, I haven't logged on yet obviously, it tells you you can change the logo, this is the logo that you see in here.

If I go into and do customize portal you will see that you can change that into whatever logo you want for your own corporations, you don't have to do anything with mine in here, you can just add your own, and again you can define it as easy or as much as you want.

And here is the link to it if you want to as well.

So right now I go in here and I see the API that I want.

Of course I can't use it until I log on, so I make it public such that anybody can see the APIs that I have, but if you want to use it you have to sign in.

So I'm just going to go ahead and sign in and I'm going to sign in with my IBM account, because remember I added IBM account as the consumer, and I'm going to go ahead and sign in as Soloman.

Now again, I come back to the portal, but this is the third one that I talked about now I'm logged in as Soloman, there's the home, there's the applications, the APIs, which is what you see in here, and the list of users.

Now let's go to the actual API – you might have a lot of them so there's an option to basically order by and look at them – this used a REST API and it was created October first 2014, that's the date when it got created.
As you see, today it’s October second, but when I set up my DataPower machine and API Management the date wasn’t set up accurately there – it was a different time zone and hence it is actually correct, but it’s a different time zone that is still October first there.

I’m going to go ahead and click on that particular API that I created and if you click on the plus you will see that this is the actual API that I’m interested in.

Notice the URL that I’m using, so if I’m going to call the API from a browser that goes through the API Management because, notice this is 192 and 68153.5 and if you look at the host name for my WebSphere Application Server machines 192 1682.9 so it’s a different host.

This is going through DataPower and going through the API Management appliance that I have in here or virtual machine in my case.

And of course it has the type equals short, the first name equals Soloman, again this is just a sample in terms of the parameters, but then look it has the client ID equals, and the client secret equals.

Now would I get client ID and client secret you ask.

Let’s go ahead and create the application first, because that’s what I’m going to do if I’m going to be using this particular API.

We’ll come back to this, in fact maybe we could just copy it and kind of leave it on the clipboard for now.

Notice that the plan is silver, that’s what I defined it with.

I never actually deployed the API into the gold plan, and hence when I do the dropdown I only see the silver.

So let’s go ahead and go to applications.

Now the last thing I’m going to do is I have to, remember, they have defined the APIs, defined the consumers, deployed it.
I could do the testing when I deploy, when I'm done testing and I'm happy with it I can publish it into a specific plan that makes it available, so if I come in here you can start seeing it.

Of course you specify the consumers at that point.

Now we will go back and create an application.

Remember I'm a developer – I am Soloman now, I am no longer part of your organization, if you will – I am the person who is trying to take advantage of that particular API and expose it to developers.

So I'm going to go ahead and create an application.

The application name is going to be SolomanWorldBankLoanApp.

And this one application that exposes the interest rate API, to be exact, get interest rate by type API.

And I'm going to register now this application which I just did.

Now look what happened.

When I registered the application it told me, by the way, here's the secret and this is the one that you need to save.

Now even though you need to save it and remember it, if you do lose it, you could easily come back and do reset, in which it will give you a new one, but for now let's just go ahead and save it, and what I'm going to do in here when I save it is, I'm going to go ahead and find a place to put it.

So I copied it like I mentioned earlier and I put it in a place here, I'm going to copy it also, so I got the secret and also I would like to get the client ID as well.

And remember the client ID that I have is the one that I will be using alongside the entry here.

So here's the client ID and as you sign in there is the client secret, and if I go back to my portal here – you will see, by the way, I could validate that, so let's say if I have the secret but I'm not sure whether it applies here or not, you can just do verify into that secret, which will tell you if it's successful or not.
Now with that, I just have the application ready and what I need to do right now with that application, if you click on it you can do any changes on it you want to that application.

We have the application, now let's go to the API.

Remember we created the API right here.

Now we said that the plans that are available to me are silver or gold, but I did not really define a plan for it just yet, so let's go ahead and select a plan, because that's what I am going to allow it to do, and this is obviously the account that's going to be used for that plan, and then I'm going to say go ahead and use this plan.

And when I do go ahead and use this plan, it's going to give me basically, we're going to use this plan, the applications that are available for you to use for this plan is the SolomanWorldBankLoanApp – remember, the one I just defined earlier, so I'm going to go ahead and select it.

And now it is successful, it is associated with it.

I went ahead with the third one, I created my application – this is the one that's going to give me the secret, because remember you can have multiple applications with multiple secrets and use whatever plans for the same exact API, that's the purpose of it, if you have silver, gold, pink, black, white, whatever, whatever plans you have, you can create an application for it with a secret, and that's the application that you will be exposing out to the public and selling it out.

So if Soloman or person X who is developer looking to use that API for their own purpose, that you want to come in and start charging money for them to use, they'll come to your portal right here and they'll see the API, because they're going to go to the home portal like I mentioned earlier, and they're going to see, here's the APIs that do exist for this particular environment, and then you can basically say, from within these APIs, basically associated with this particular application as you see in here, SolomanWorldBankLoanApp, and it tells you these are the APIs, and by the way it has a silver to it, this is the API that exists with the silver plan.
I give them the client ID, I give them the secret after I sign him up with my service, and from that point on, now they can just simply start making the call and taking advantage of my API.

The last illustration before we end this video or part 3, now remember, I said I have five calls per minute.

So let's go ahead and see if we can reach the five calls per minute.

What do I want?

I want the API first of all, so let me go back to the API.

I want to make sure that I have the URL for the API, remember I wanted you to save it to the clipboard, so let's go ahead and paste it here, and instead of client ID I'm just going to put the client ID, which is this number right here.

That's what I'm going to replace this one with.

And of course we have the and, and the client secret equals, and that's the one that I'm going to replace with the client secret that I have in here.

And that should be all that I want that I need to actually get this working.

Now at this point, I would like you to look at – let's go back to the portal with the application and look at the analytics, which basically shows me the actual calls.

So far I've only made one call, remember when I did the testing.

So I'm going to go ahead right now and through this browser right here I'm going to go ahead and just paste it and I'm going to call it a couple of times, here's once, remember 3375, remember I added the rate, Soloman, and first name, and the type is short.

If I go back to WebSphere, here you go, it's 3375 and short.

Let's see what happens if I change it into jumbo instead of short and the first name being SolomanIBM, just to change it to kind of show you, and click that again it's the jumbo rate 3.075.
I go back to WebSphere, WebSphere tells me – remember I’m making the calls through the API Management – now WebSphere is automatically being updated, as you see, just to illustrate that I am calling the APIs that are hosted on the IBM WebSphere Application Server, which is a completely different layer obviously than the API Management layer.

Let me do this a couple of more times.

Now if you go back in here and let’s refresh this website quickly.

And if I go to the analytics, now you will see that I just called it nine times which is really what happened in here.

This is how you can keep track of how many times you are calling a particular API and whether you want more details on it.

I mean this comes down, gives you even more information, look at this, it’s telling you, by the way this is SolomanI and it’s called a couple of times, and it’s got SolomanIBM, and I’ve got Soloman by itself, I got Solo, and response time, 17 milliseconds the first time, 518 milliseconds.

All that gives you a lot of information that is very useful to you, even sometimes the request body and request headers and response, this is something that only returns to me the response, which is OK, that’s because I used a proxy.

If I actually end up using the assemble one, you end up seeing a lot more than what you saw in here.

Hopefully this illustrates to you what I was talking about when it comes to the API Management and how useful it is.

The next video that I’m going to be showing you is basically the Worklight backend, where I’m going to use Worklight to develop an application that’s going to be using the APIs that are defined in the API Management to make a call, to get the interest rate, and use it actually in my mobile application, and I will be using the API Management solution for that.

Thank you and I will talk to you in the next video.