Then next I'm going to need to make some small changes so that the flow of information is correct between the consumer of the API and the Lambda function, and then back again. I'm just trying to use it as a demonstration to indicate exactly what each component of API gateway does and this is probably the least important part of the lesson, but I do need to do it to make sure that this end to end flow of information works. So first, I want to configure some validation of the imputs to this API. So I'm going to click on method request and then tick the pencil next to request validator and I want to change this dropdown to validate query string parameters and headers, so that just allows us to perform some validation. So I'll select that and then click on the tick to save that setting. Once I've done that, I need to define exactly what information the API is going to receive and so I'm going to expand URL query string parameters because later on, I'll be configuring how that information is mapped through to the Lambda function. Now, in this case, because it's a calculator function, it's going to accept three individual bits of data the two values so the two numbers that it's going to perform an operand on and we would call those operand one and operand two and it's also going to take an operator, which will be the type of operation that's going to be performed. So I'm going to add all of those three query strings. So the first one is operand1 and then operand2 and then lastly operator and then I need to set all of these three to require values, because in order for the API to function correctly, it will need all of these three to be provided. Next, I'll need to instruct API gateway to send those values received by the API to the Lambda function. So to do that, I'm going to go back to this method execution screen. I'm going to select the integration request component. I'm going to expand mapping templates and the mapping template defines exactly what the Lambda function will receive, so it maps the information that the API is given to the information that it gives to the underlying Lambda function. So I just need to provide a little mapping template, so that's what I'll do next. So I need to make sure that when no template matches the request content type header is selected, it should be the default, but I need to make sure it's selected, and then I'll click on add mapping template and for content type, it's going to be application for /json because this is the type of information that's going to be received by this API. So I'll select that I do want to secure this integration so I select yes, secure this integration. Don't worry about exactly what that does. It is beyond the scope of what you need for the solutions architect associate certification. Just keep in mind what I'm doing is I'm creating a mapping between what the API gets and what Lambda gets. Now, inside the lesson files for this lesson, I've created a mapping template that I'm going to use. So this essentially just maps the values of operand1 and two and operators and these are the values at the API receives and it maps those onto three variables A, B, and OP and A, B, and OP or what the Lambda function will expect. So this mapping essentially converts from these names. So these are what the API gets into these names which are what the Lambda function gets. So I copy those into my clipboard, move back to the console, paste this into the mapping template box, and click on save. Okay, so that's all we need to configure for the API. So at this point, we should be able to test it, so I'm going to go back to method execution on I'm going to select test. This allows us to test the API before we publish it. So we're able to simulate the data that the API will receive before we actually make it public and use real services or applications. What I'm going to do here is paste in this, which is essentially the query string that this API would receive in the real world. So I'm going to specify an operand1 value of two. So that's the first value to be manipulated by the calculator. Operand2 is three, so that's the second value and then operator is plus and if this works, what should happen is these values will get passed through to the Lambda function by using those mappings, the Lambda function will receive three variables A, B, and OP so A will be equal to two, B will be equal to three, and OP will be equal to the plus sign and then the Lambda function will take those, perform the operation, return it back to the API, and we should get that result back. So there you go. It took 418 milliseconds to perform this operation. It used the Lambda function for the logic part. It passed in A, B, and the operator and it got back the value of five. So this proves that the API is working. It's accepting these values, its passing those through to the Lambda function. We can see from the logs that this Lambda function is getting executed. It's providing values back and the API would have returned this body if it was used in production.

So now this API is fully functional, but it's not yet published. We don't have an endpoint and that's the next step we need to do in order for an API to be active and available, we need to create the API endpoint. We need to publish it. So I'm going to do that. I'll click on actions and **deploy the API**. Now API Gateway allows each individual API to have multiple stages. Now you can think of these as prod, dev, test, or version one, two, and three. You can use the stages however you want. I'm going to create a single new stage. I'm going to call the stage V1. For the description, I'm going to put version one and for the deployment description, I'm just going to something random so Hi Linux Academy. Once I'm happy with those, I'll go ahead and click on deploy and that will deploy out the API to the V1 stage so the version one stage and this is how I can make this API available to the public internet because now I have an API endpoint. I can use this endpoint to interact with the API across the public internet. Now, to demonstrate this works, I'm going to use third party utility called Post Man. It's a free download, and I'll make sure I include a link in the lesson description where you can download it for whatever operating system you're using. So copy that into my clipboard, I'll move across the Postman. Don't worry about the details of how this works. It just is an application that allows it to interact with APIs. So I'm going to do that. I'm going to open a new session and I'm going to interact with the API using the API endpoint URL but on the end of this, I'm going to add /calc which is the resource inside this API. Now the API is expecting three different query parameters. It's expecting operand one and two. So I'll take those and then it's expecting an operator. Now it needs to have values for all three of those. I'm going to run the same test. I'm going to use the value of two, a value of three, and then for the operator as well as the symbols, I can also add text descriptions so I could use add instead of the plus sign to add these two values together and I'm going to do with that. So I'm going to perform a call against the public endpoint for this API and it's going to attempt to add two and three together. So I'll click on send and then we get the result back and we can see that the result is five. It is what we expected, and that proves that the API is deployed and fully functional.

Now, this has been a pretty simple demonstration but what I've essentially done with very little effort is **create an API using API gateway and to support this API, the logic is provided by a Lambda function. This API could be provisioned and while it's not being used, it's costing next to nothing. The only time significant resources are used are when the API is consumed by another service or another application and when it does get consumed, I'm only billed for the level of that consumption. So when the Lambda function is executing as well as any of the resources that are involved in the API call. Now API gateway does have a number of other features, which might come in handy to be aware of so you can utilize the API cache. So instead of having to constantly run against these backend logic functions, if the results returned are similar you can enable the API cache and it will store results in the API gateway and return those rather than spinning up these underlying resources. You can also use the web application firewall, which we haven't discussed yet and that provides a layer seven application Firewall that can sit in between your customers or consumers of your API and the API gateway.** So don't worry about that. We'll be talking about it extensively later on in the course but remember, you do have the option to use the **web application firewall, and you can do that by associating a web ACL with this API** and a web ACL is the entity provided by the web application firewall. It's essentially and application firewall that you can associate with AWS services.