1-2\_Making Resources

Sunday, August 22, 2021

9:36 AM

Two Checkboxes we have, What do they mean?

Services V Q Search forsen.'ices, features, marketplace products, and docs Ohio Support 
Amazon API Gateway APIs > compare-yourself (2kw15a2kg3) > Resources > / (Iq9sxzoOml) > Create 
Show all hints 
APIS 
Custom Domain Names 
vpc Links 
API: compare-yourself 
I Resources 
Stages 
Authorizers 
Gateway Responses 
Models 
Resource Policy 
Documentation 
Settings 
Usage Plans 
API Keys 
Client Certificates 
Settings 
Actions - 
Resources 
New Child Resource 
use this page to create a new child resource for your resource.' 
Configure as C?proxy resource 
Resource Name' 
Resource path* 
Enable API Gateway CORS 
• Required 
compare-yourself 
/ compare-yourself 
You can add path parameters using brackets. For example. the 
resource path {username) represents a path parameter called 
'usernarne'. Configuring '(proxp} as a proxy resource catches all 
requests to its sub-resources. For example. it works for a GET 
request to ,'foo. To handle requests to l, add a new ANY method 
on the / resource. 
Cancel 
Create Resource 

』 『



**Configure as proxy resource**

Configure as proxy resource means that this will be a catch-all resource, catching all other paths and methods. So it will be a flexible path and your API may only have this single resource because it catches all requests.

Why do we do this?

Serverless Approach is great for creating single page applications with APIs, especially APIs which you can also use with mobile apps but limitation relies on limited usage or use case where you can use for full stack apps you and it's this use case.

If you catch all incoming requests and methods and forward them to whichever action you're executing, this action can be a lambda function, this code on-demand construct.

Lambda supports NodeJS and since it does so, you can simply run your Node, Express or whatever app on a lambda function, forward all requests to it, therefore do the routing in that lambda function and return HTML, whatever you want, so you can basically create a fullstack app with the serverless approach you could say.

**Enable API Gateway CORS**

CORS stands for Cross Origin Resource Sharing and it is about a security model where we're in general not allowed to access resources on a server from another server.

So let's say, our single page application written with Angular or React or anything like that runs on example.com. On example.com is our app, this app of course has no server-side code, our API runs on api.com.

Now we have two different servers, two different domains here. Now our single page application wants to access data on our API and that's a common use case, you have it all over the web, for example also if you use the Google API. Now by default this is prohibited.

How can we solve this?

The server api.com has returned the right headers to the client. The client needs to know that it is OK to access this data because otherwise the client, the browser especially to be precise, will prevent this request from being handled in the client in the browser. So the server needs to send headers telling the browser that this cross site access is OK, that there is nothing fishy about it and therefore theses right headers have to be sent.

Another Check that need to be done - Preflight Request

If you are using modern browser (e.g. Chrome) they typically send preflight requests to the server when you were sending post, put, delete, basically anything but get or header requests. Preflight requests simply are there to check if the resource the post request is about to get sent to is actually available and if it's allowed to send a post request or whichever request you are about to send.

So Chrome sends this extra request without you asking for it but it still does it and therefore this extra request will hit your API. Now you need to be able to handle this extra request and actually it will be a request of type options, that's a HTTP verb just like put or post or get. So you need to provide an options endpoint so that Chrome can send its preflight request and this options endpoint also needs to return the right CORS headers to inform Chrome that it is OK to send the post request and that it is OK to do so even though we're using different resources, different servers.

And if we check this box, it will create us such an options method which is automatically configured to send the write headers back to the client, to the browser so that the browser get no issues accessing our API.