VAL'S PROJECT 1

END TO END WEB APP

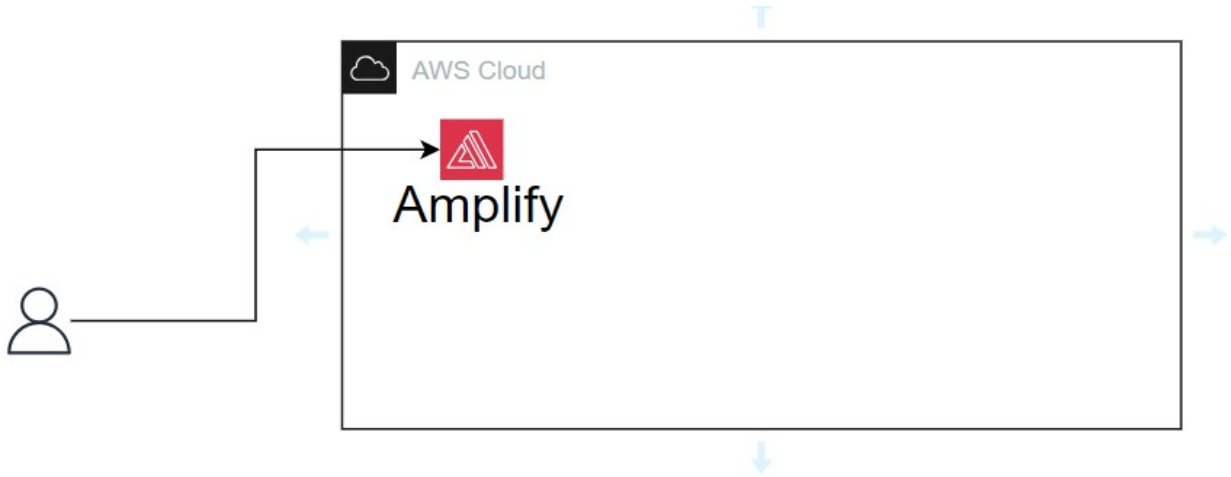
Resources used: (AWS Amplify, API Gateway, AWS lambda, IAM; Identity and Access Management).

The process of building this App(Solving Math problems)

- A way to create and host a webpage.
- A way to invoke the Maths functionality.
- A way to do some Maths.
- Somewhere to store/return the Math results.
- A way to handle permissions.

I will be using this AWS Amplify Architecture below;

AWS Amplify is used to build and host websites but we'll use a simple (index.html) page.

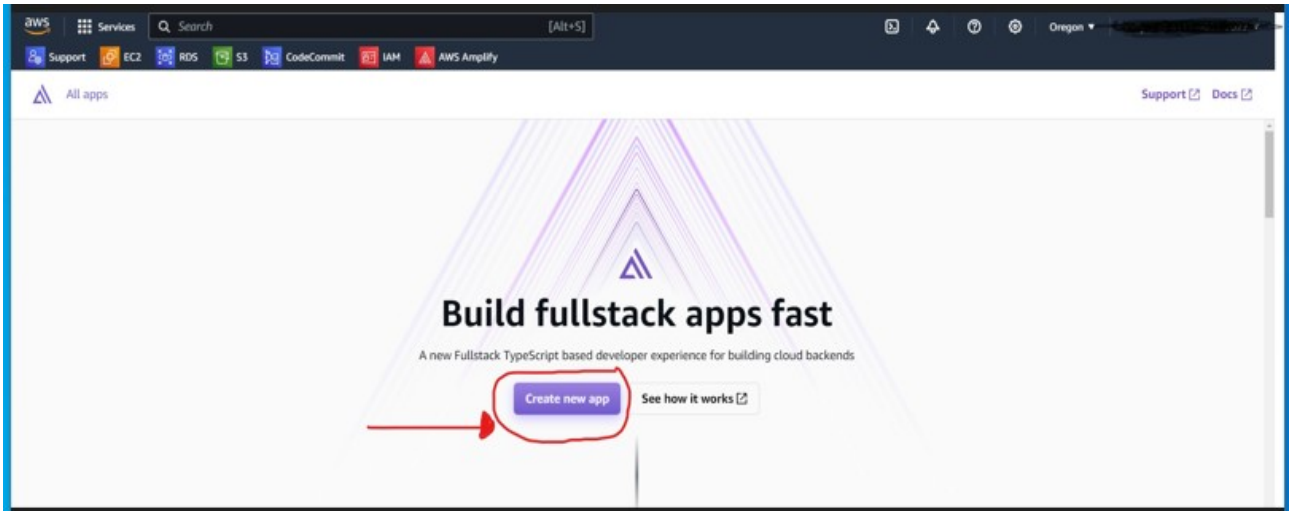


AWS Amplify is also used for frontend Apps. The next step was to create a HTML page. I did it by following these steps:

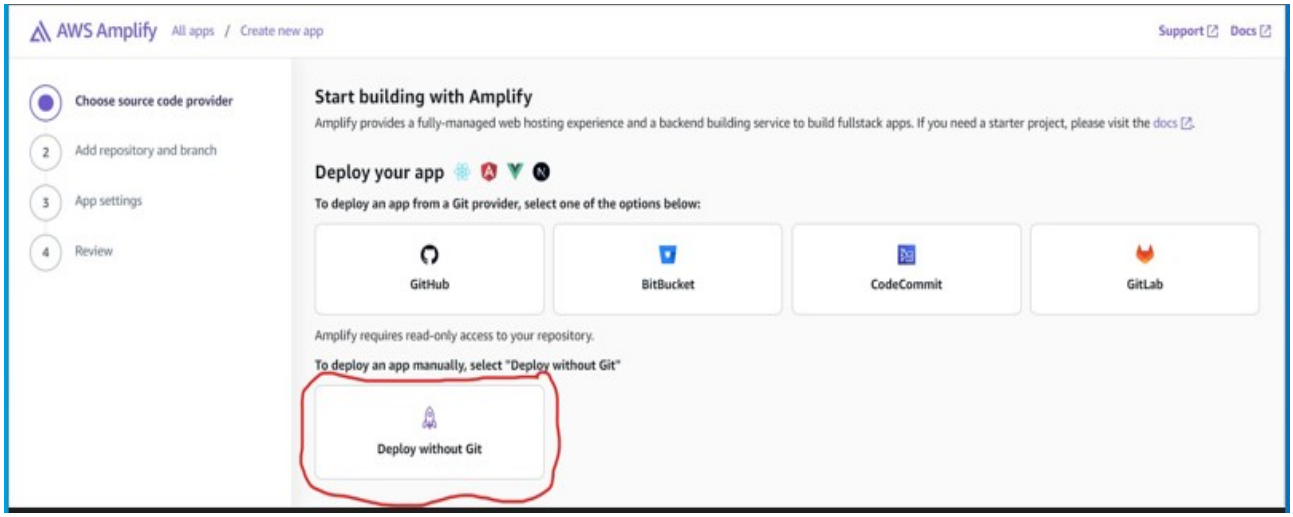
- Right click, open a (new.txt file).
- Create a new folder.
- Open it and (click on view) then (tick file name extensions, then change the (file.txt to html) and save it.
- Getting the codes from the HTML file saved which is (My Projectfile.html) opening it using (Notepad).
- Grabbing the code (index.original.html) and pasting them onto the Notepad, also save the code.

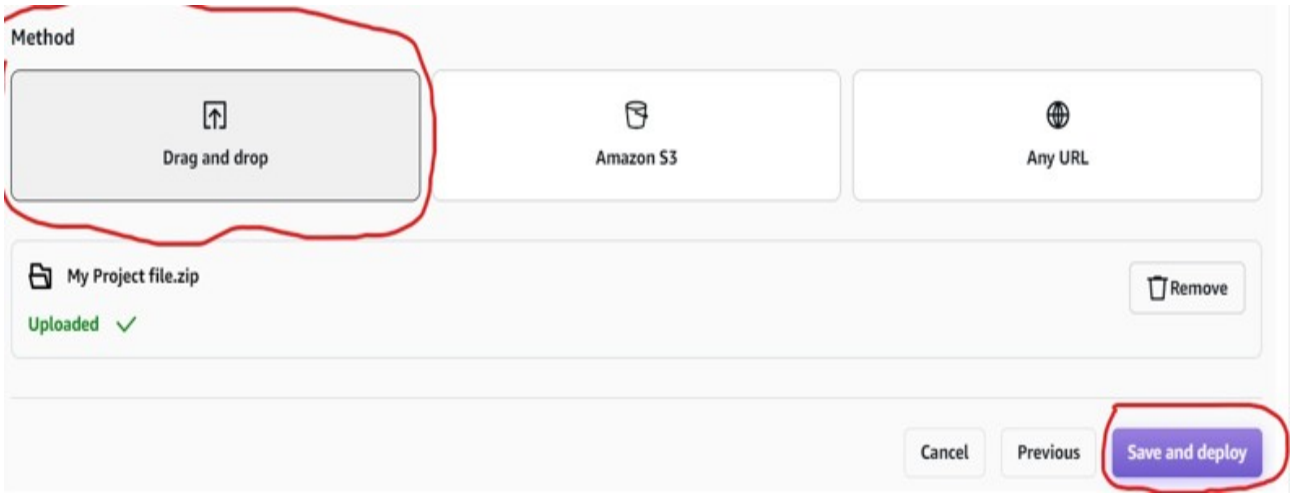
Deploy using Amplify.

- Go to AWS Amplify.
- Click on (Deploy without a Git).
- Give the app a name; I named mine (ValsProject2AmplifyDeployment).
- Branch name (Dev).



- Then drag the (Zip file) I created, and Amplify is deployed.





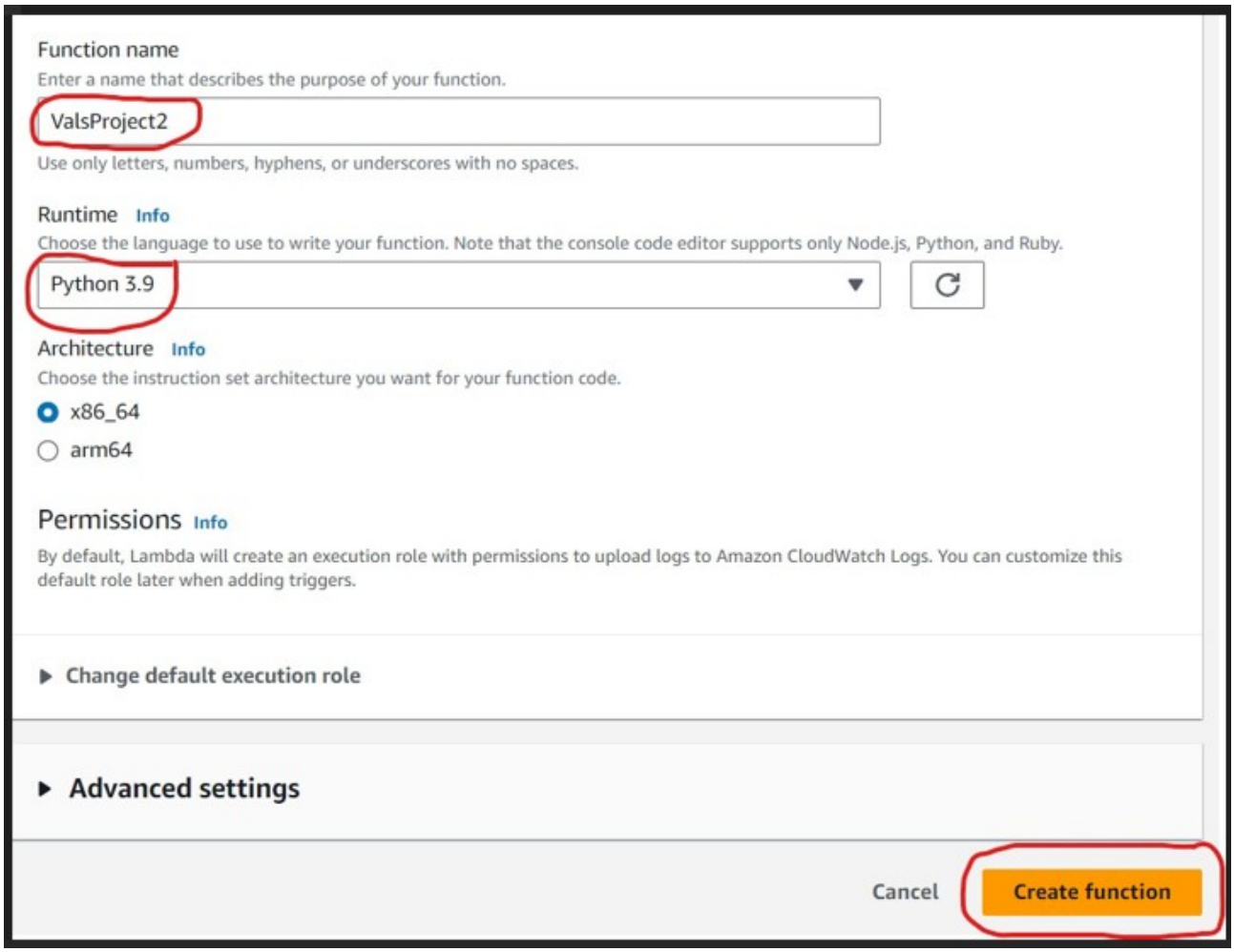
The next step is to introduce a Lambda function to run some code upon some trigger and Lambda is a severless function. I started out by creating a Lambda function with the following steps

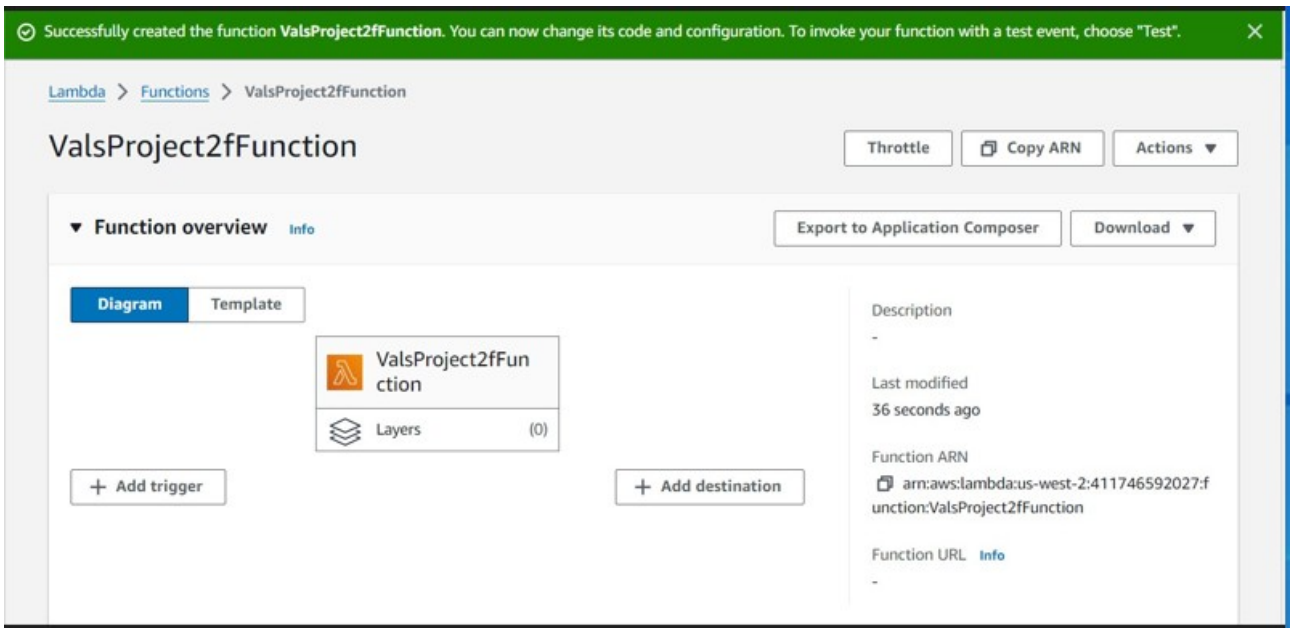
Steps 1; Creating my Lambda function.

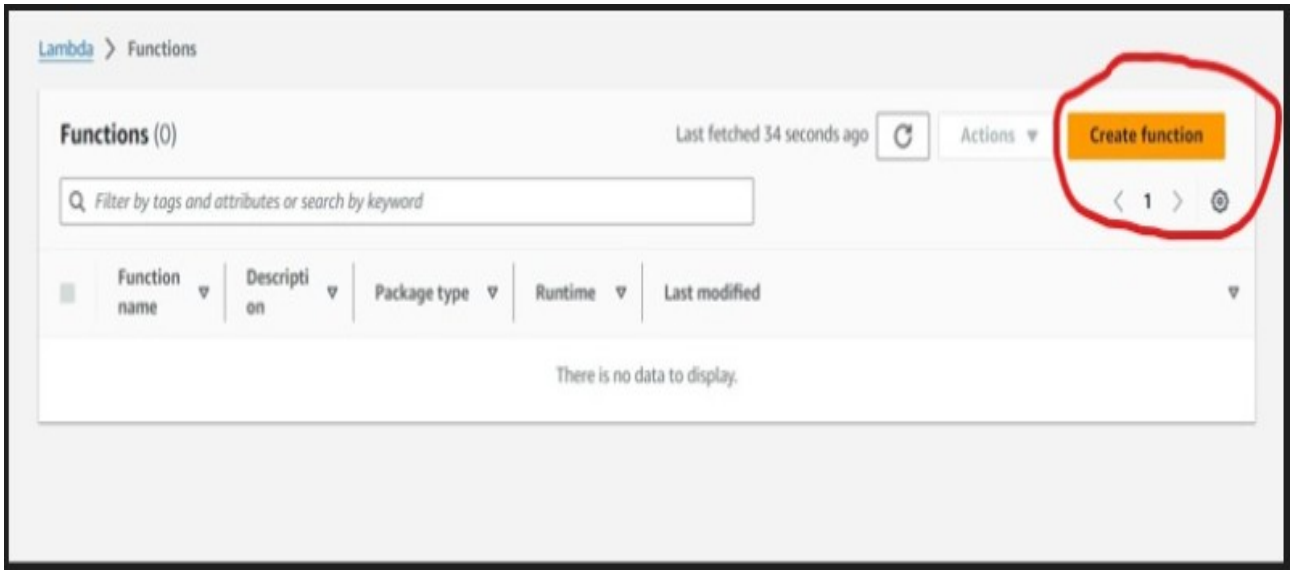
Using the basic Math function as an example (2^3=8).

- Navigate to Lambda.
- Go to (create a function) in the Lambda page.
- Click on (Author from scratch).
- Function name (ValsProject2function).
- Runtime (Python 3.9), the latest version of python.
- Leave everything else the same and click (create function).

These are shown in the photos below;



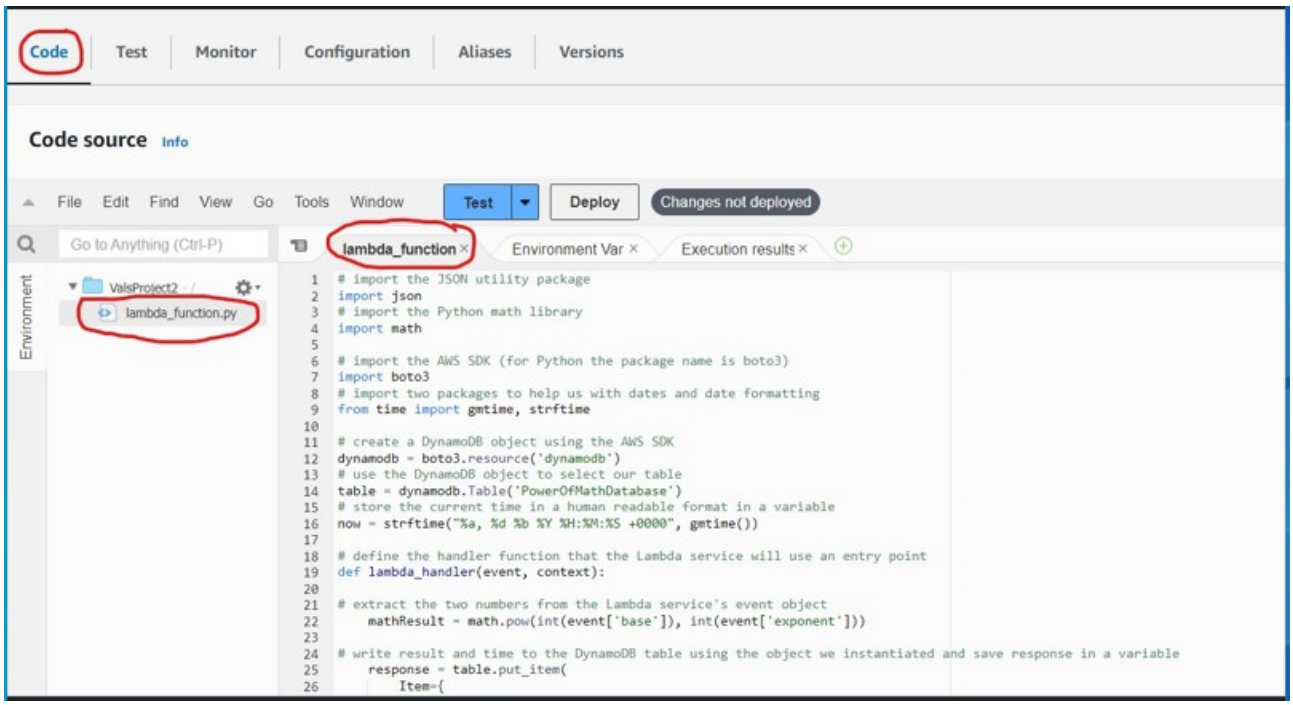


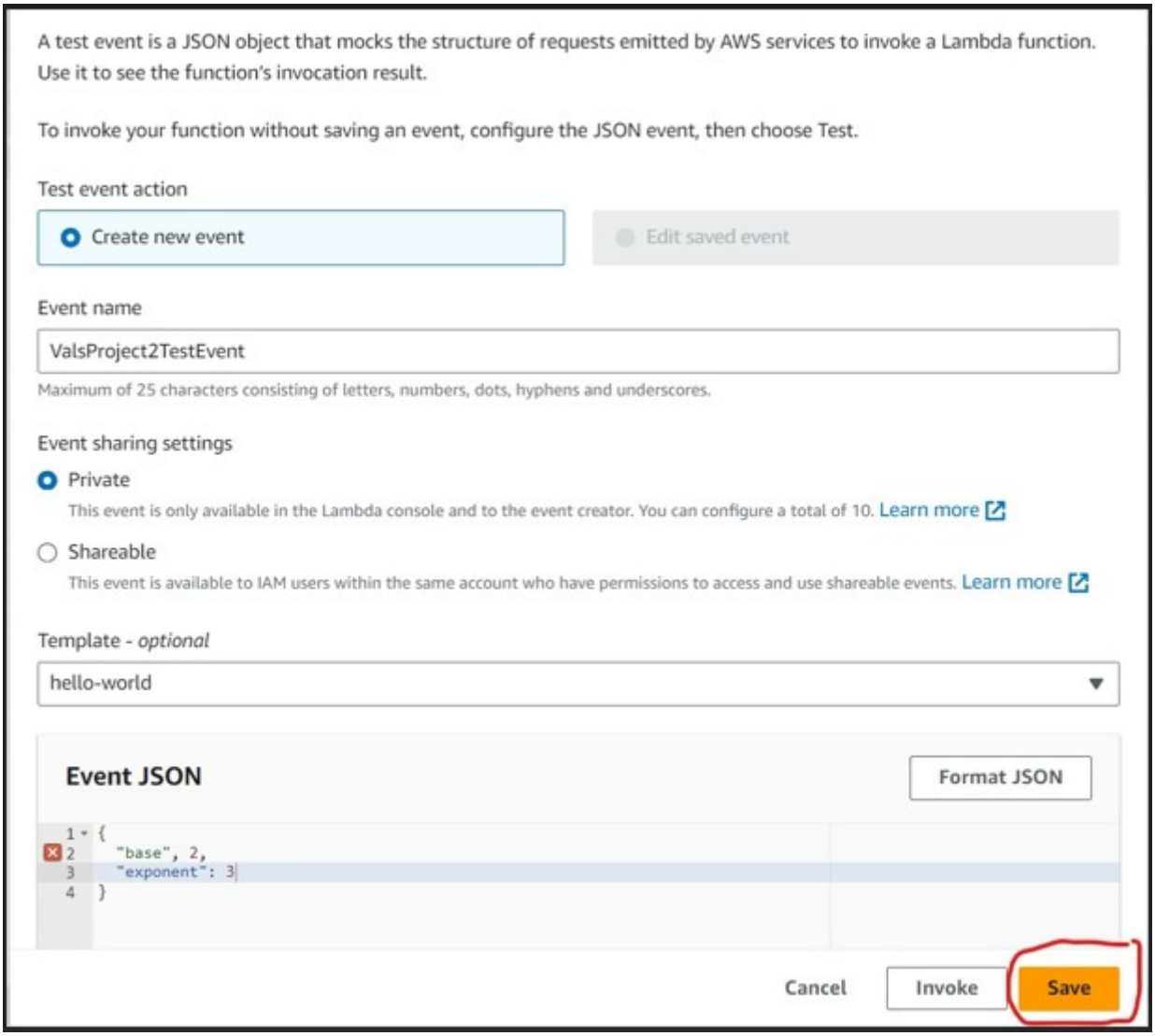


Step 2.

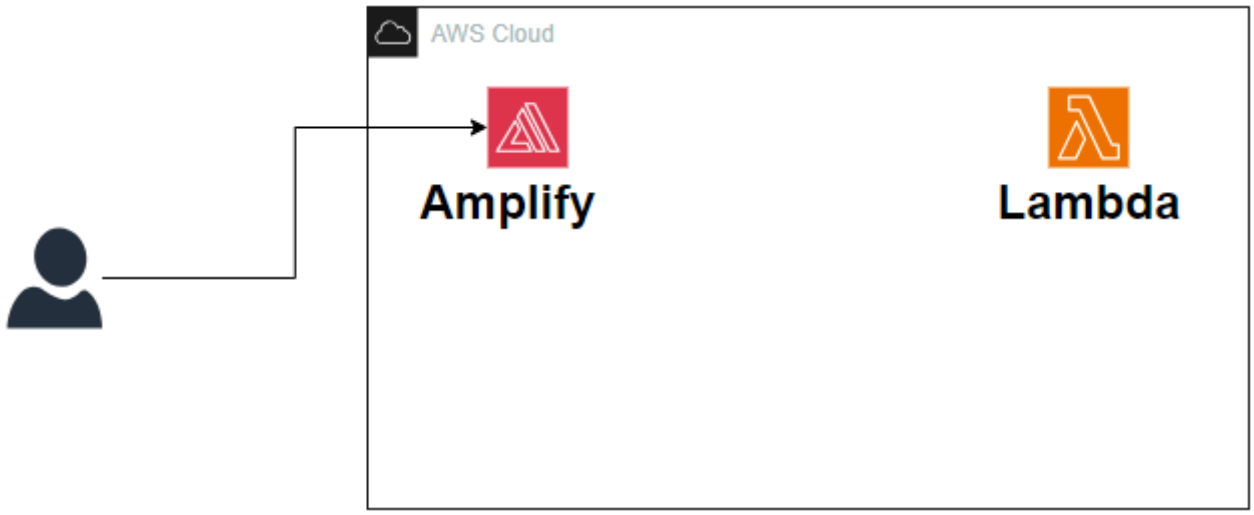
- I have to grab to grab a Lambda original code from a repository.
- Then I had to paste it in the (Lambda function) section under the (Code source).
- Then I clicked on (Deploy) and clicked on (test), then (Open the configure test event) and change line 2 and 3 of the Event JSON document to line 2 (“base”, “2”) and line 3 (“exponent”, “3”).

These are shown in the photos below;


Code
Test
Monitor
Configuration
Aliases
Versions
Code source Info
File Edit Find View Go Tools Window
Test
Deploy
Changes not deployed
Go to Anything (Ctrl-P)
lambda_function
Environment Var
Execution results
Environment
ValsProject2
lambda_function.py
# import the JSON utility package
import json
# import the Python math library
import math
# import the AWS SDK (for Python the package name is boto3)
import boto3
# import two packages to help us with dates and date formatting
from time import gmtime, strftime
# create a DynamoDB object using the AWS SDK
dynamodb = boto3.resource('dynamodb')
# use the DynamoDB object to select our table
table = dynamodb.Table('PowerOfMathDatabase')
# store the current time in a human readable format in a variable
now = strftime("%a, %d %b %Y %H:%M:%S +0000", gmtime())
# define the handler function that the Lambda service will use an entry point
def lambda_handler(event, context):
# extract the two numbers from the Lambda service's event object
mathResult = math.pow(int(event['base']), int(event['exponent']))
# write result and time to the DynamoDB table using the object we instantiated and save response in a variable
response = table.put_item(
Item={


A test event is a JSON object that mocks the structure of requests emitted by AWS services to invoke a Lambda function. Use it to see the function's invocation result.
To invoke your function without saving an event, configure the JSON event, then choose Test.
Test event action
Create new event
Edit saved event
Event name
ValsProject2TestEvent
Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.
Event sharing settings
Private
This event is only available in the Lambda console and to the event creator. You can configure a total of 10. Learn more
Shareable
This event is available to IAM users within the same account who have permissions to access and use shareable events. Learn more
Template - optional
hello-world
Event JSON
Format JSON
{
"base", 2,
"exponent": 3
}
Cancel
Invoke
Save

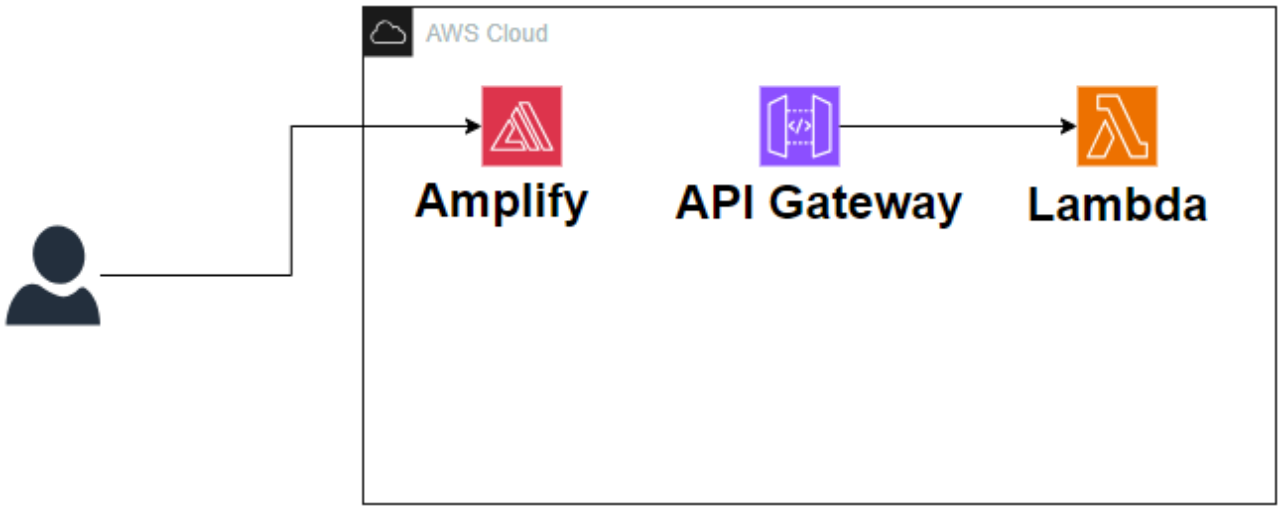
The Current Architecture.



Step 3

- I have to look for a way to invoke the Math functionality or basically invoke the Lambda function.
- I need a (Public URL or Endpoint) to trigger that function and for that, I will introduce to an (API Gateway).

The New Architecture



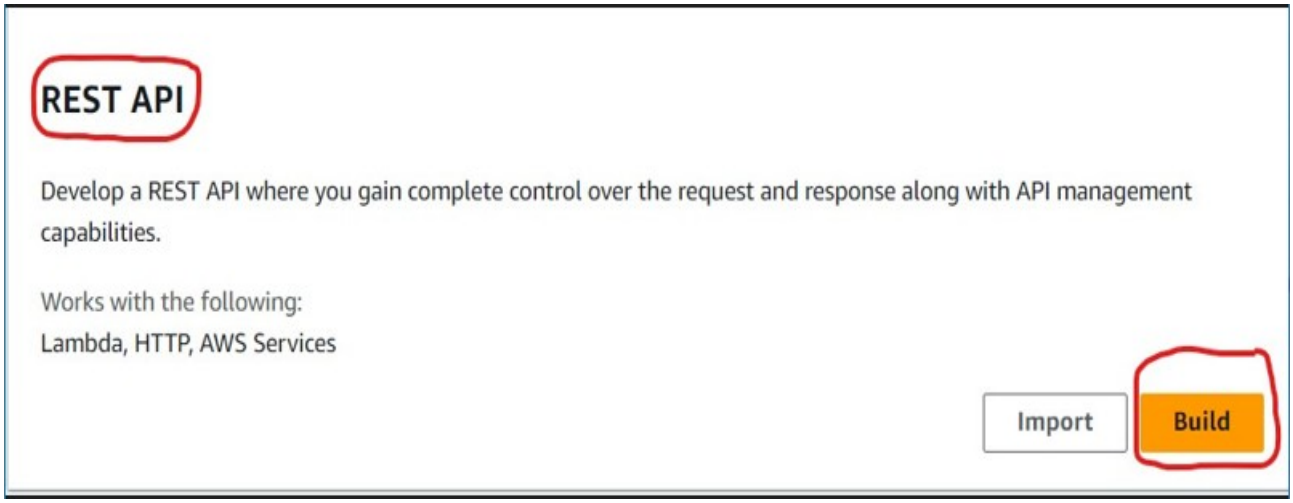
API Gateway is used to build (HTTP, REST and Websocket API'S).

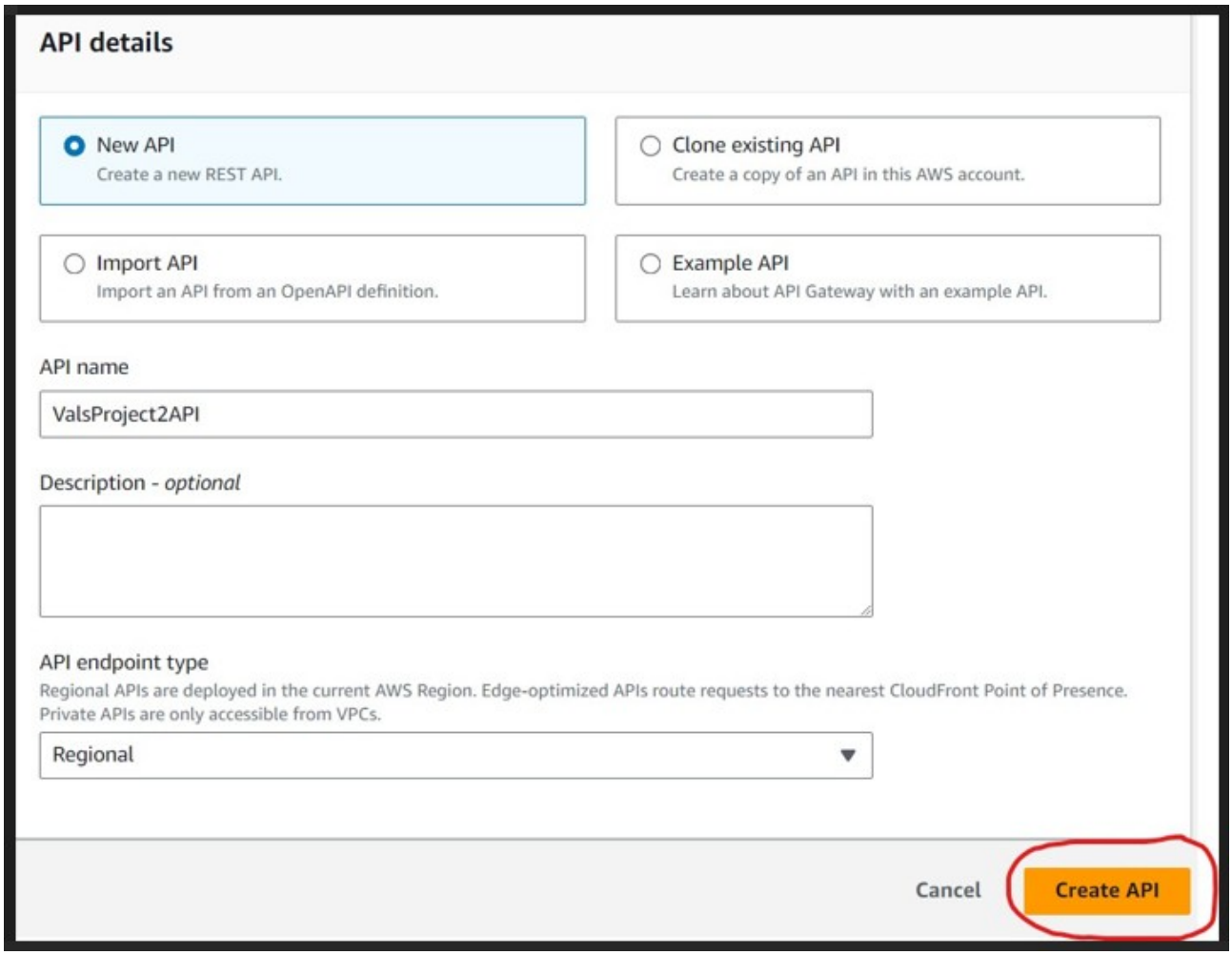
Step 4 (Create an API Gateway).

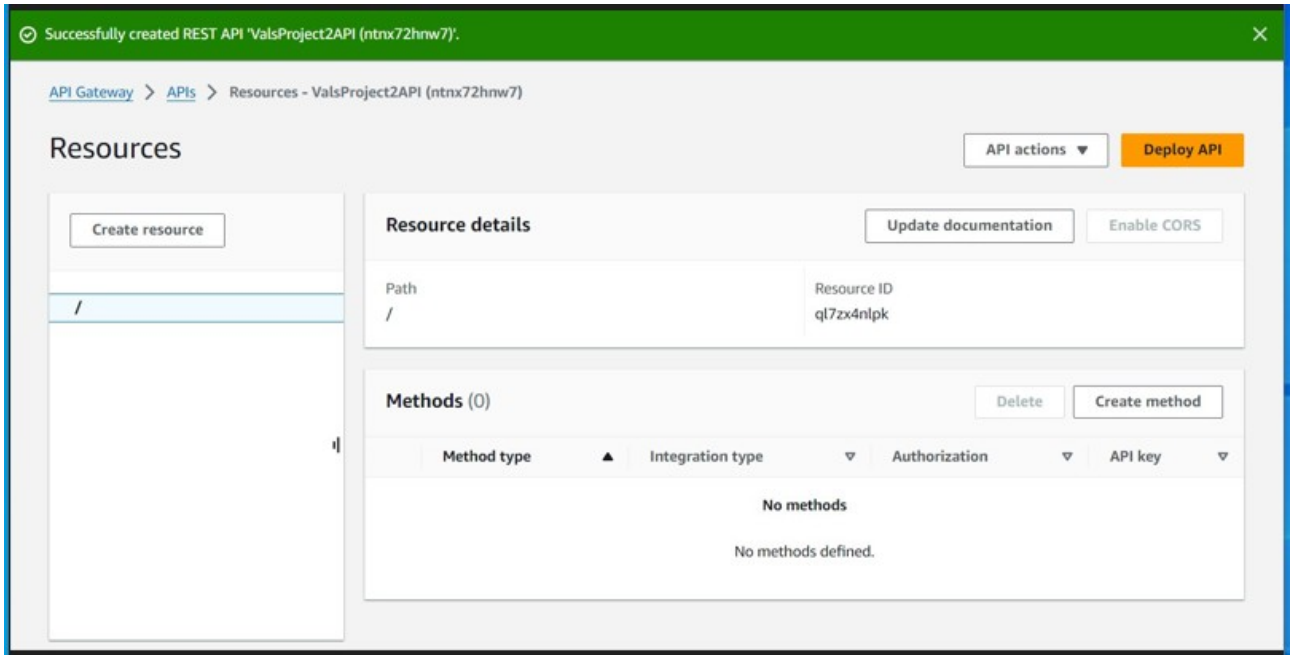
- Open a new tab and navigate to API Gateway.
- I will be using a REST API
- With this, I gained complete control over the request and response along with API management capabilities. This works with the following; Lambda, HTTP and other AWS services.
- It's an empty API.
- Then go to (Resources) and then to (Actions), then click on (Create Method).
- From the (Create Method) tab select the (Method).
- Select a (POST) method as the type of method; I am sending in some data hence I used a (POST) method.
- Next, I had to enable C.O.R.S (Cross Origin Region Sharing).
- Deploy again and save the (invoke URL)
- Go to (Resources) and I clicked on (Post).

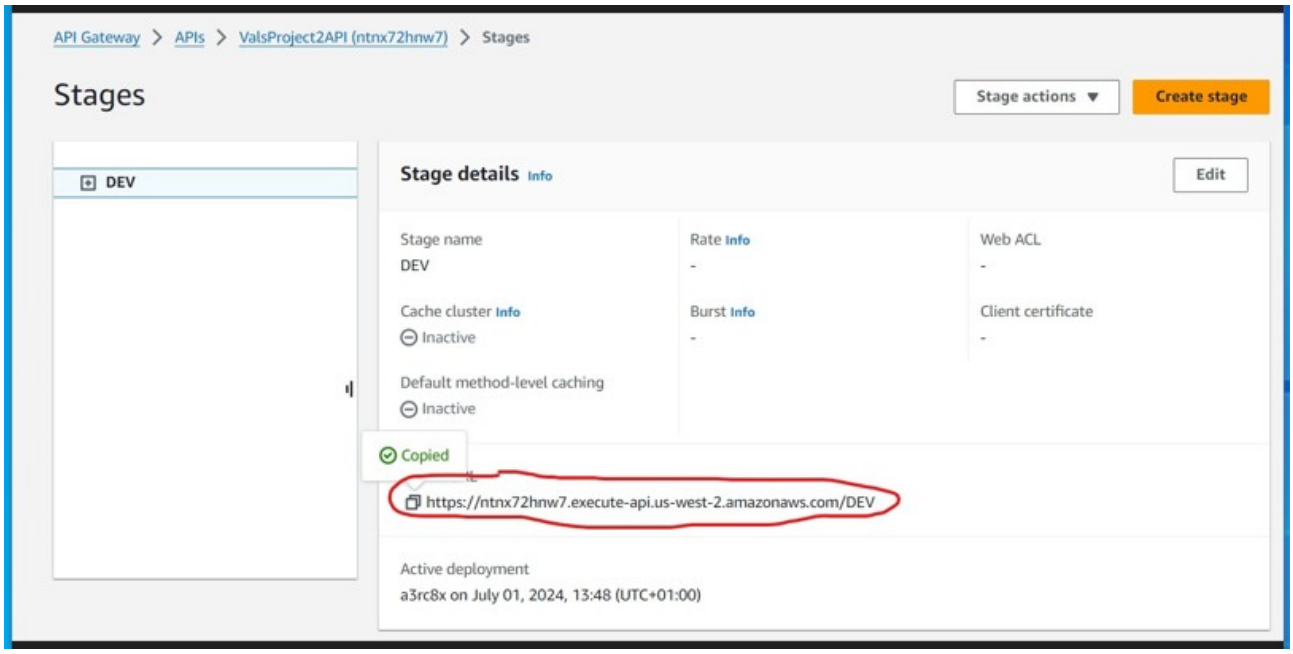
The Test was successful.

These are shown in the photos below;

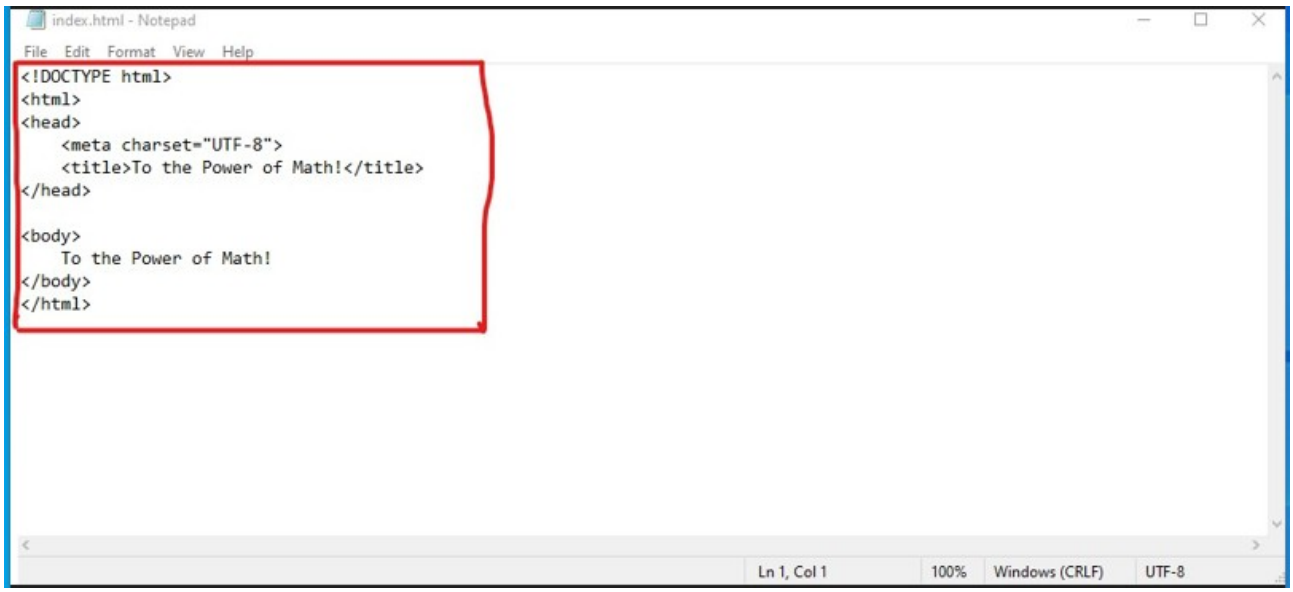



API details
New API
Create a new REST API.
Clone existing API
Create a copy of an API in this AWS account.
Import API
Import an API from an OpenAPI definition.
Example API
Learn about API Gateway with an example API.
API name
ValsProject2API
Description - optional
API endpoint type
Regional APIs are deployed in the current AWS Region. Edge-optimized APIs route requests to the nearest CloudFront Point of Presence. Private APIs are only accessible from VPCs.
Regional
Cancel
Create API


Successfully created REST API 'ValsProject2API (ntnx72hnw7)'.
API Gateway > APIs > Resources - ValsProject2API (ntnx72hnw7)
Resources
API actions
Deploy API
Create resource
/
Resource details
Update documentation
Enable CORS
Path
/
Resource ID
ql7zx4nlpk
Methods (0)
Delete
Create method
Method type
Integration type
Authorization
API key
No methods
No methods defined.



Now I can trigger my Math function in Lambda with an API Call. I haven't attached that to my (index.html page) and (Amplify) yet.



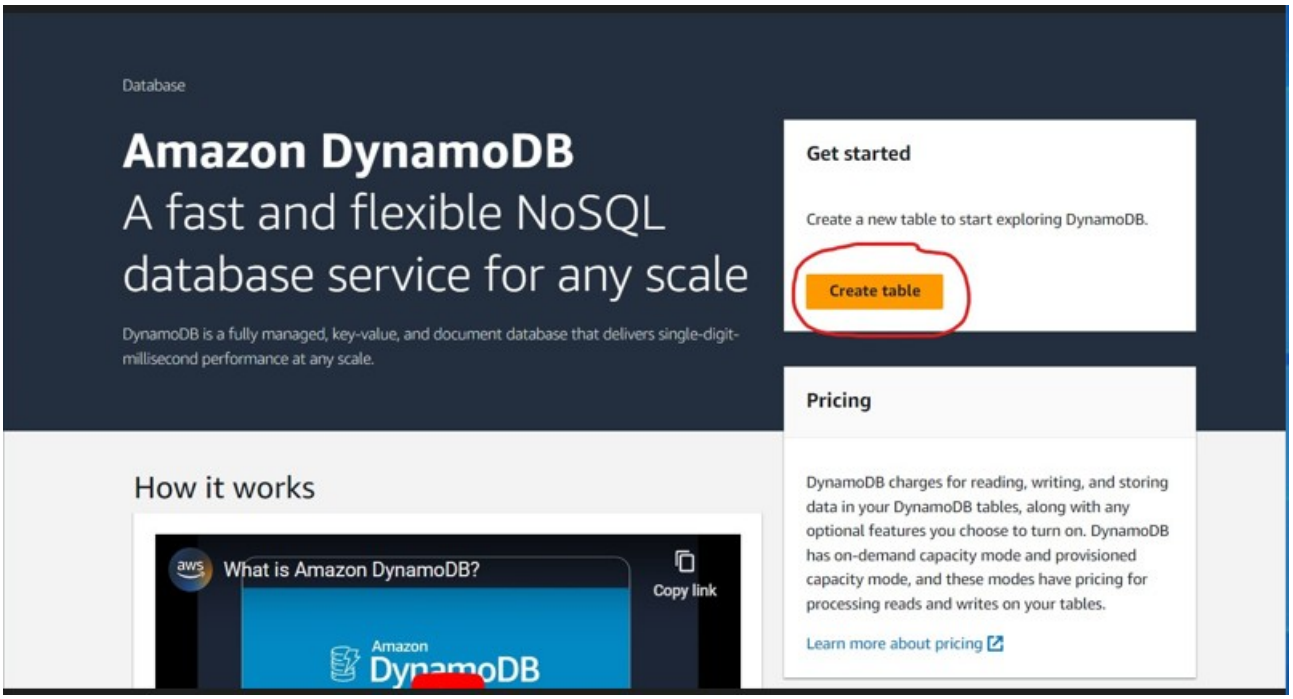
Next, I have to incorporate a Database into all of this to store the Maths results (Data results). I need to handle permissions and I am going to use Dynamo DB which is a key value, No SQL database.

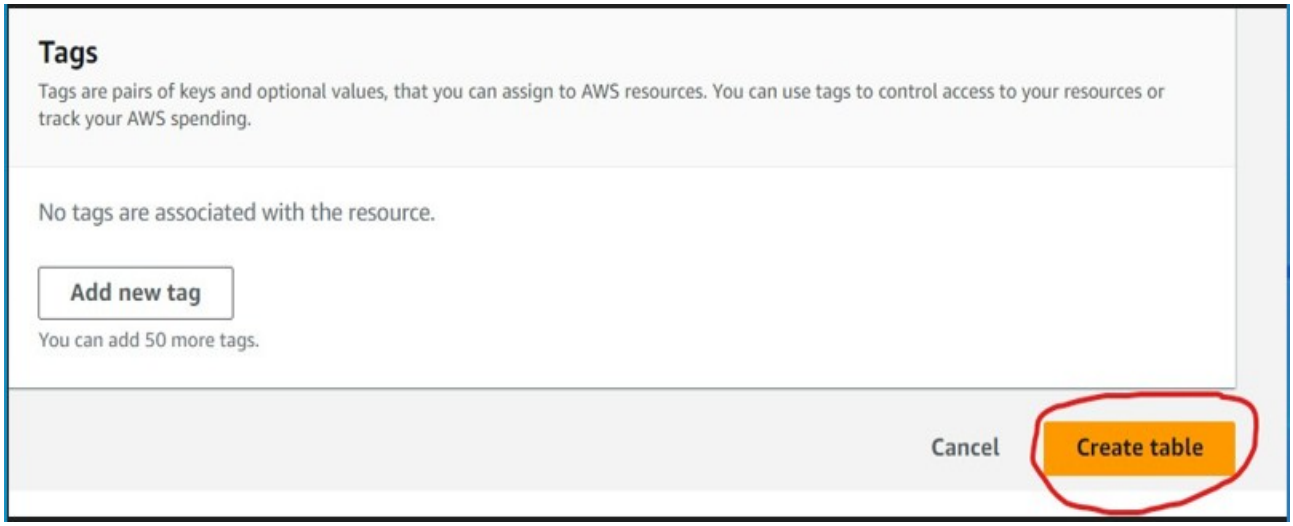
Step 5

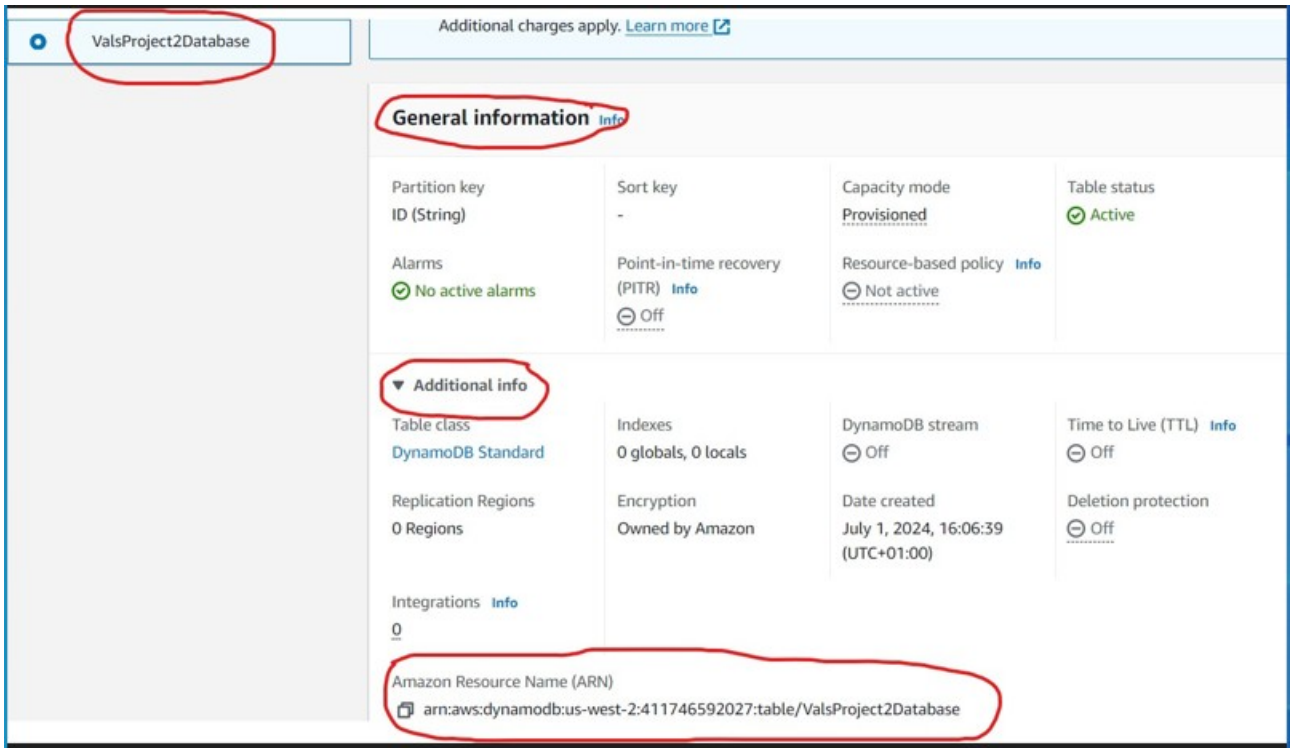
Dynamo DB.

- Go to Dynamo DB on the console.
- Go to (Create table), Click on the (Create table).
- Table Created; I need to save the ARN (Amazon Resource name) for the Dynamo DB table (ValsProject2Database).
- I clicked on (ValsProject2Database), next (General information) and (Add Additional information), then copied the (ARN).

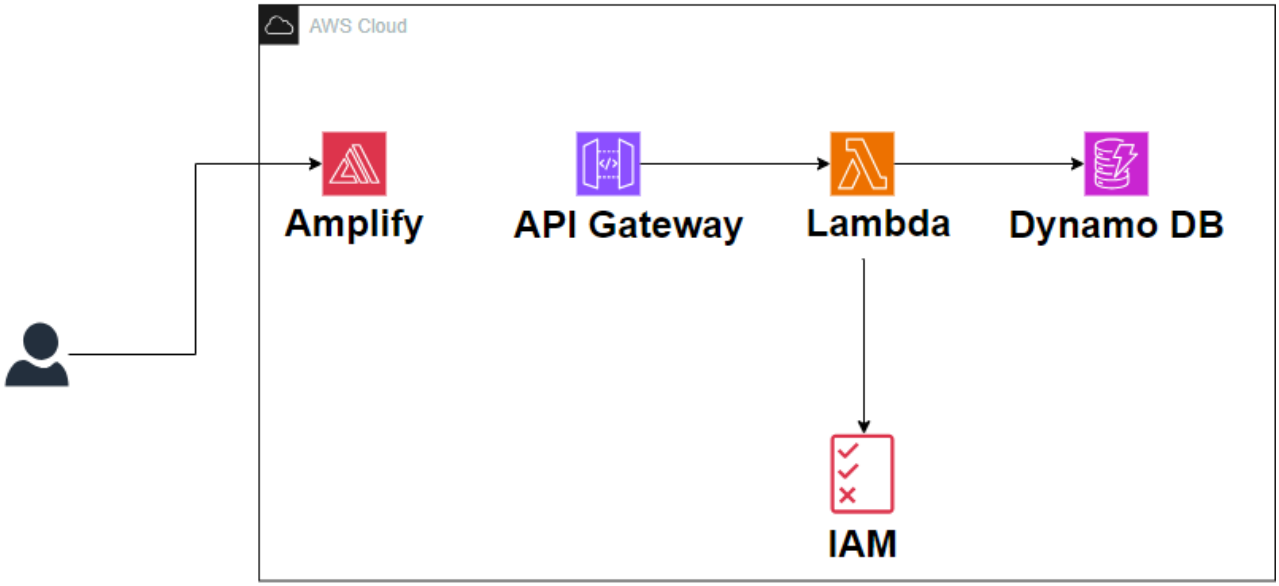
This is shown in the photos below;







Next, I have to link my Lambda function to my Dynamo DB. So going back to the Architecture which is shown below;

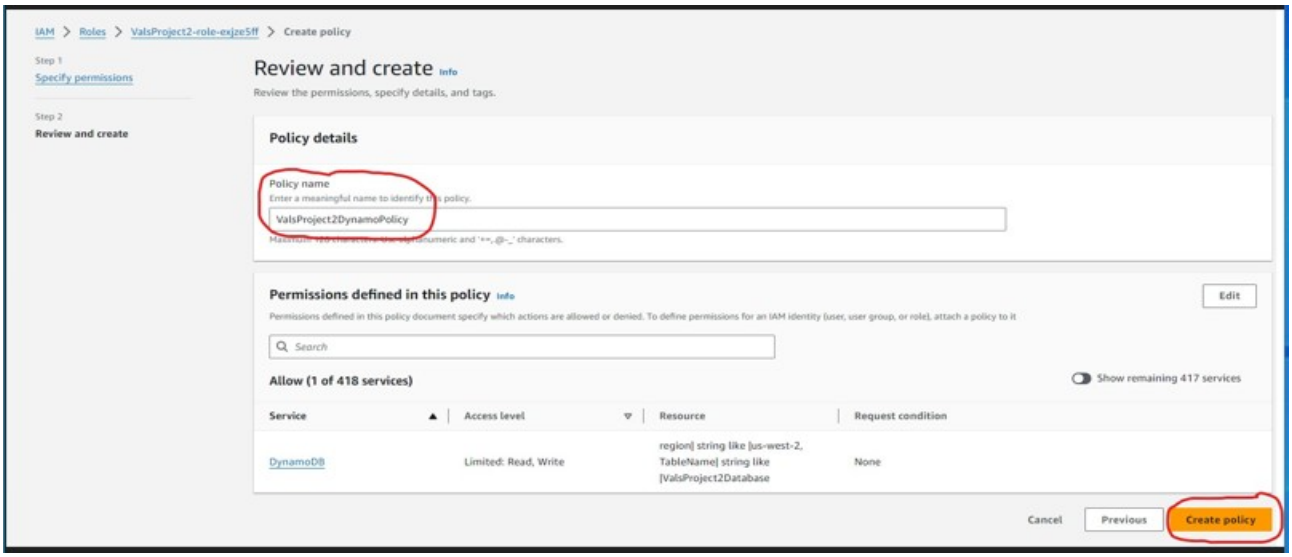


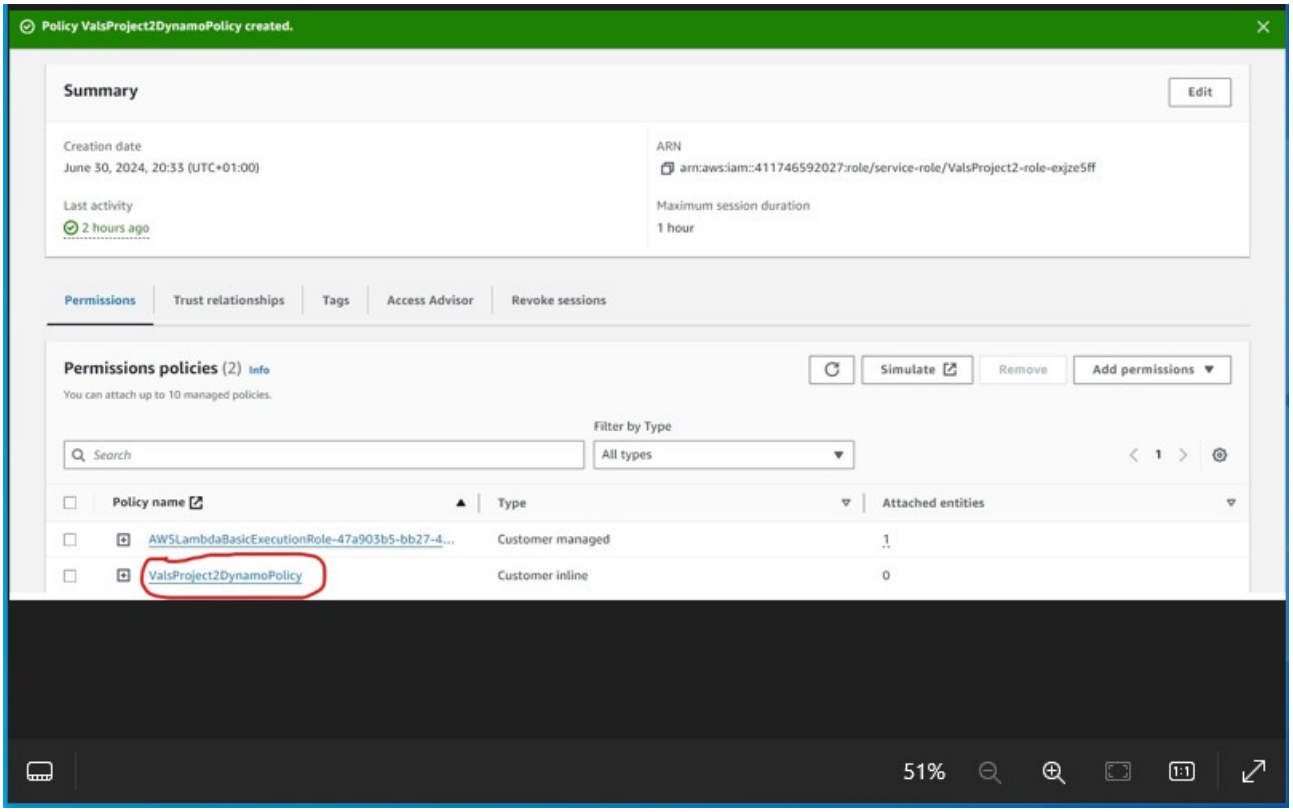
Next, I have to give permissions to my Lambda function.

Setting Up Permissions.

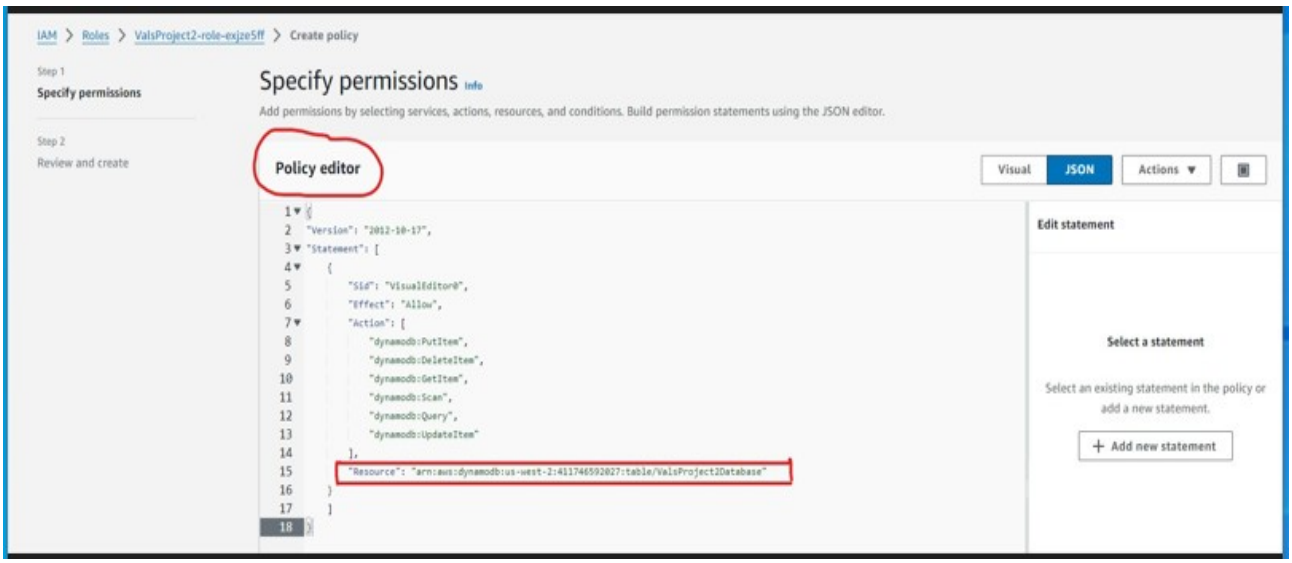
- Go back to my Lambda page.
- Go to the (Execution Role).
- Go to the (Role name), open it and then,
- Go to the (Permissions) page giving Lambda permissions to write to the (Dynamo DB table).
- Go to (Add permissions).
- Then go to (inline Policy).
- Click on the (JSON Policy) and go grab the (Code) needed which is in the repository.
- This (Code) is the (Execution Role Policy JSON.txt Code), grab it and paste the Code in the (Specify permissions) Policy editor.
- Then replace line (15) of the JSON Code which is (“YOUR-TABLE-ARN”) with the ARN copied and click on (Review policy) and (Next).
- Go to the (Review and Create) page, go to the (Policy name), give it a name (ValsProject2DynamoPolicy) and click (Create).

This is shown in the photos below;


IAM > Roles > ValsProject2-role-exjze5ff > Create policy
Step 1
Specify permissions
Step 2
Review and create
Review and create Info
Review the permissions, specify details, and tags.
Policy details
Policy name
Enter a meaningful name to identify this policy.
ValsProject2DynamoPolicy
Permissions defined in this policy Info
Edit
Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it
Search
Allow (1 of 418 services)
Show remaining 417 services
Service
Access level
Resource
Request condition
DynamoDB
Limited: Read, Write
region| string like |us-west-2,
TableName| string like
|ValsProject2Database
None
Cancel
Previous
Create policy


Policy ValsProject2DynamoPolicy created.
Summary
Edit
Creation date
June 30, 2024, 20:33 (UTC+01:00)
ARN
arn:aws:iam::411746592027:role/service-role/ValsProject2-role-exjze5ff
Last activity
2 hours ago
Maximum session duration
1 hour
Permissions
Trust relationships
Tags
Access Advisor
Revoke sessions
Permissions policies (2) Info
You can attach up to 10 managed policies.
Simulate
Remove
Add permissions
Filter by Type
Search
All types
1
Policy name
Type
Attached entities
AWSLambdaBasicExecutionRole-47a903b5-bb27-4...
Customer managed
1
ValsProject2DynamoPolicy
Customer inline
0
51%
1:1

Dealing with the permissions, I went back to the (lambda function), now Lambda function having permissions to write to the table.

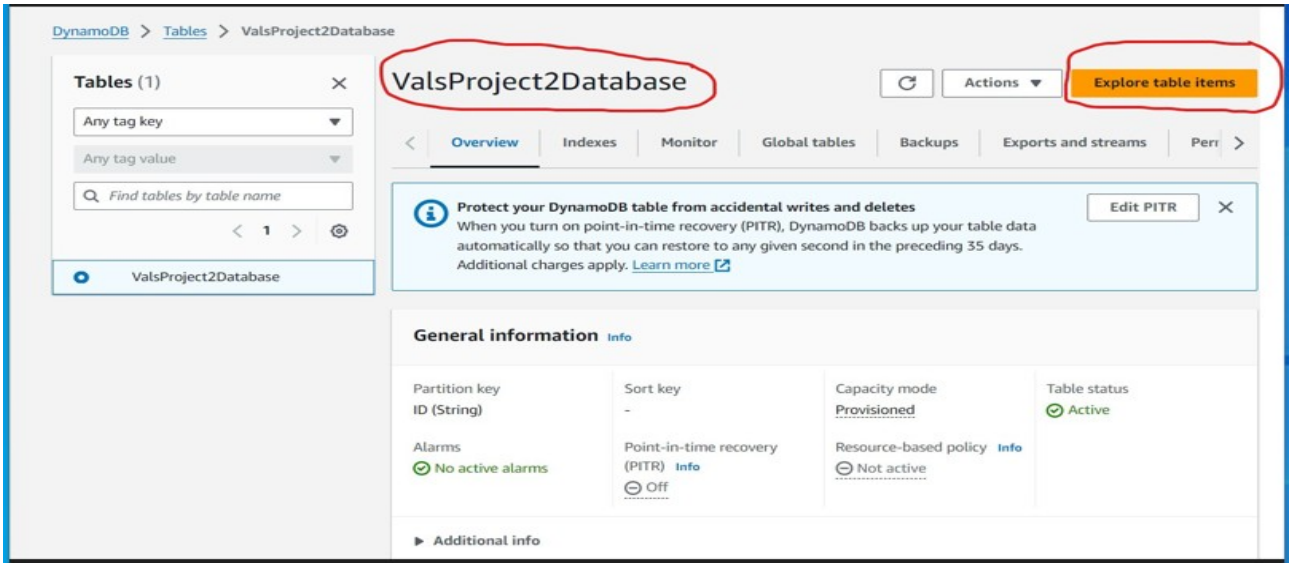


Steps to Testing and Re-Deploying.

- Go to the Lambda function; Grab the Code (Power of the Math function final Code).
- Get the (Code), paste it, (Deploy) it then (test) it.
- Go back to the Dynamo DB table
- Go to the (ValsProject2Database) and open it.
- Go to (Explore table Items) to show what has been stored in there.
- See result in the Dynamo DB table result.

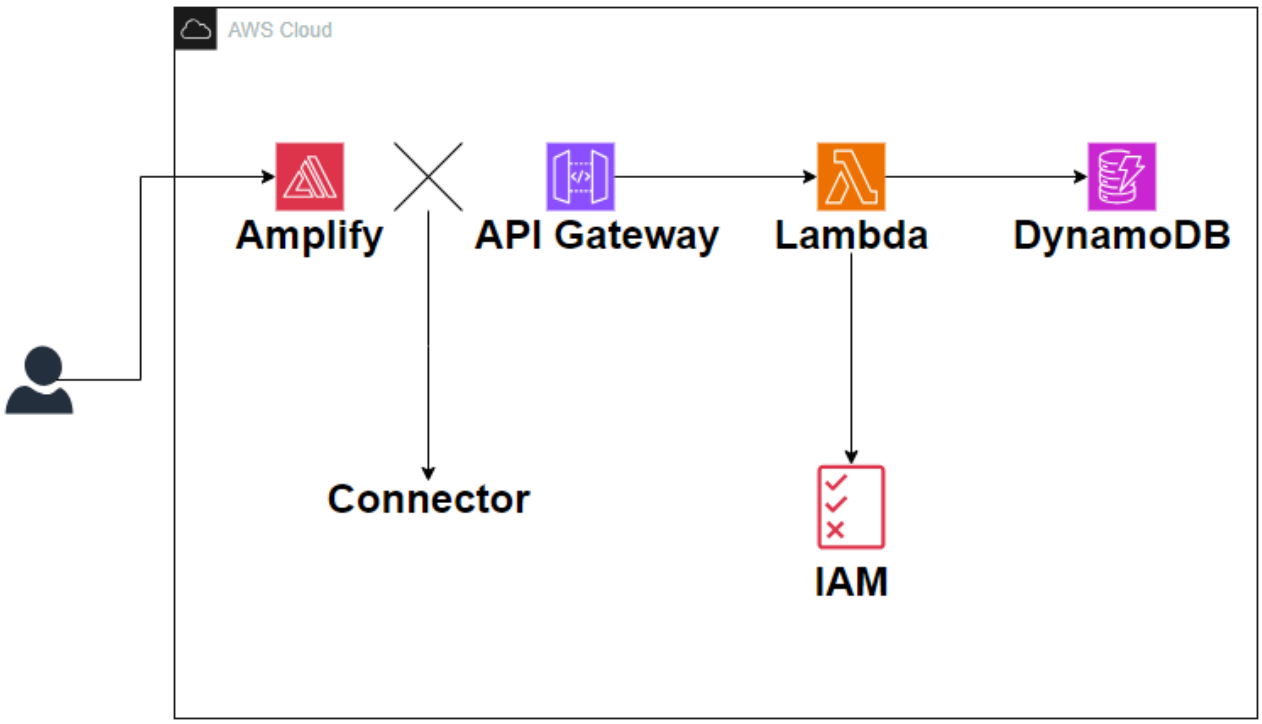
Now I am writing things to the Dynamo DB table and got the permissions of the Lambda function and they are both integrated with each other.

These are shown in the photos below;



Going back to the Architecture, I need to connect the (Amplify) to the (API Gateway) after connecting all the other resources on the right side of the Architecture.

Current Architecture.



For the final piece;

- I am going to update the (index.html) page.
- Open the (index.html folder/file) and get the code.
- Open the (index.html file) with (Notepad).
- Next grab the (final code) from the (repository) copy and paste it onto the (html Notepad).

- Then go under the API section of the code and replace it with your (API Gateway Endpoint).
- Go to (API settings) and copy the (API Gateway Endpoint) and paste it onto the code line of (YOUR API Gateway Endpoint).
- Make a (new zipfile), deleting the (old one).
- Go to the save updated (index.html) file and send it to a (compressed zip folder) to create a new (Zip folder).
- Deploy the new updated code in the (zip folder) with (Amplify).
- Go back to the console and open (Amplify).
- Follow the procedure for creating a new deployed app which I explained in the beginning of this project.

The final Architecture of my Web App.