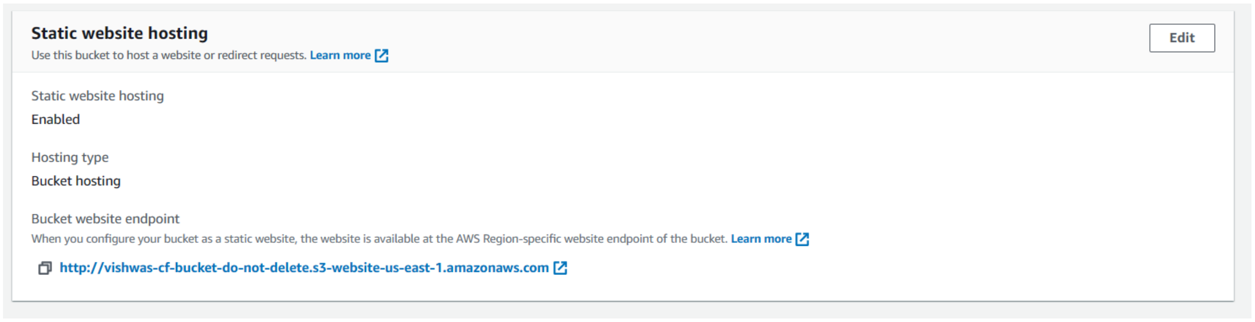
Hands-On: Serverless Website Hosting Using S3, DynamoDB, Lambda & API Gateway

To start with, Create an S3 Bucket and Make it available for S3 website hosting.



Static website hosting

Use this bucket to host a website or redirect requests. Learn more

Static website hosting
Enabled

Hosting type
Bucket hosting

Bucket website endpoint
When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. Learn more

http://vishwas-cf-bucket-do-not-delete.s3-website-us-east-1.amazonaws.com

You can create a sample html file using the following code:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Website Counter</title>
  <body>
<h1> This is Website Counter Sample page</h1>
<p>You are visitor number: <span id="visitors"></span></p>

<script>
// GET API REQUEST
async function get_visitors() {
    // call post api request function
    //await post_visitor();
    try {
        let response = await
fetch('https://j8lobmiw54.execute-api.us-east-1.amazonaws.com/default/VisitorCounter', {
            method: 'GET',
        });
        let data = await response.json()
        document.getElementById("visitors").innerHTML = data['count'];
        console.log(data);
        return data;
    } catch (err) {
        console.error(err);
    }
}
```

```
get_visitors();

</script>
  </body>
</html>
```

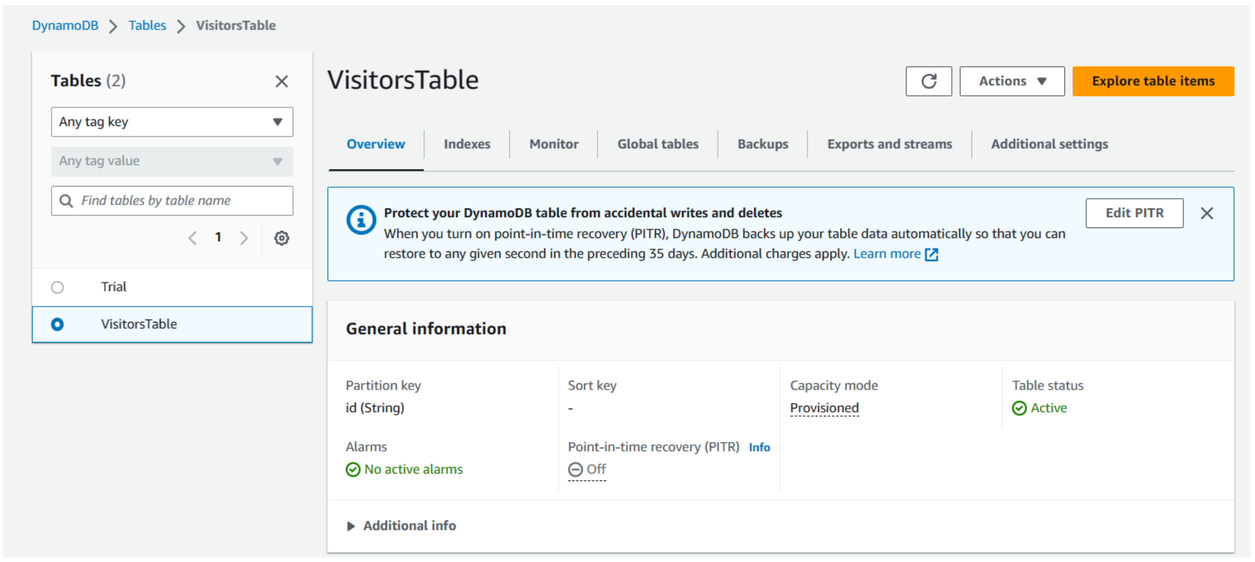
Now, As our website also has a functionality which allows us to monitor user traffic coming to the website and maintain the Visit Count in real-time. To store and update the visit counts in real time, we would need a DynamoDB Database.

Create a DynamoDB Table with the Configurations Below:

TableName: VisitorsTable
PrimaryKey : id (string)

Once created, add an item with id as **visitor_count.**

Now add a new attribute: **visitor_count** (number) and initialize it as 0 or 1.



Protect your DynamoDB table from accidental writes and deletes
When you turn on point-in-time recovery (PITR), DynamoDB backs up your table data automatically so that you can restore to any given second in the preceding 35 days. Additional charges apply. Learn more

General information

Partition key: id (String)	Sort key: -	Capacity mode: Provisioned	Table status: Active
Alarms: No active alarms	Point-in-time recovery (PITR) Info: Off		

Done with DynamoDB Table.

Now, let's create a Lambda Function which will be triggered whenever a user visits our website (this will happen as an API will be called on every user request which will trigger the lambda function). Once lambda is triggered, it will update the user count in the database and return that same to API as a response which in the end will be delivered to our website and updated user count will be shown to the users.

You can use the code below for Lambda:

```
import json
import boto3
import os

# Initialize dynamodb boto3 object
dynamodb = boto3.resource('dynamodb')
# Set dynamodb table name variable from env
ddbTableName = os.environ['databaseName']
table = dynamodb.Table(ddbTableName)


def lambda_handler(event, context):
    try:
        # Try to update the item
        ddbResponse = table.update_item(
            Key={
                'id': 'visitor_count'
            },
            UpdateExpression='SET visitor_count = visitor_count + :value',
            ExpressionAttributeValues={
                ':value':1
            },
            ReturnValues="UPDATED_NEW"
        )
    except:
        # If the item doesn't exist, create it
        table.put_item(
            Item={
                'id': 'visitor_count',
                'visitor_count': 1
            }
        )
        ddbResponse = table.get_item(
            Key={
                'id': 'visitor_count'
            }
        )
```

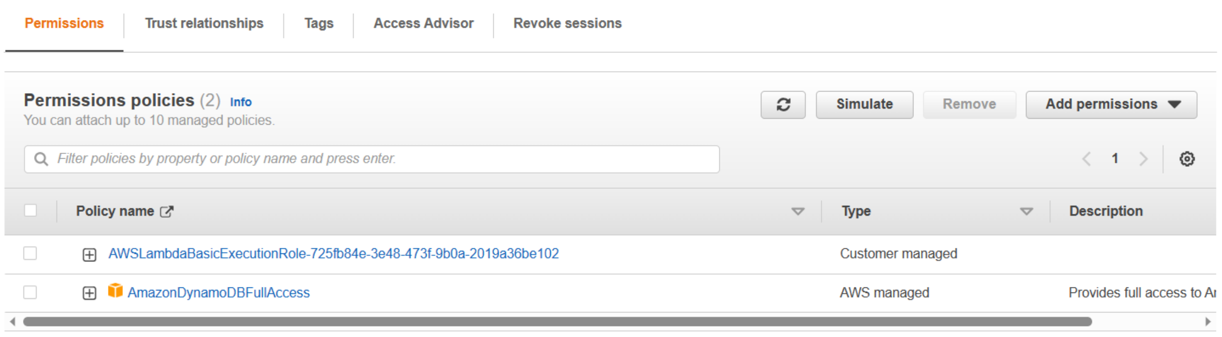
```
# Format dynamodb response into variable
responseBody = json.dumps({"count": int(ddbResponse["Attributes"]["visitor_count"])})

# Create api response object
apiResponse = {
    "isBase64Encoded": False,
    "statusCode": 200,
    'headers': {
        'Access-Control-Allow-Headers': 'Content-Type',
        'Access-Control-Allow-Origin': '*',
        'Access-Control-Allow-Methods': 'OPTIONS,POST,GET'
    },
    "body": responseBody
}


# Return api response object

return apiResponse
```

Now, as this Lambda Function needs access to the DynamoDB Database, lets edit the default execution role associated with the lambda function and provide it **DynamoDBFullAccess**.

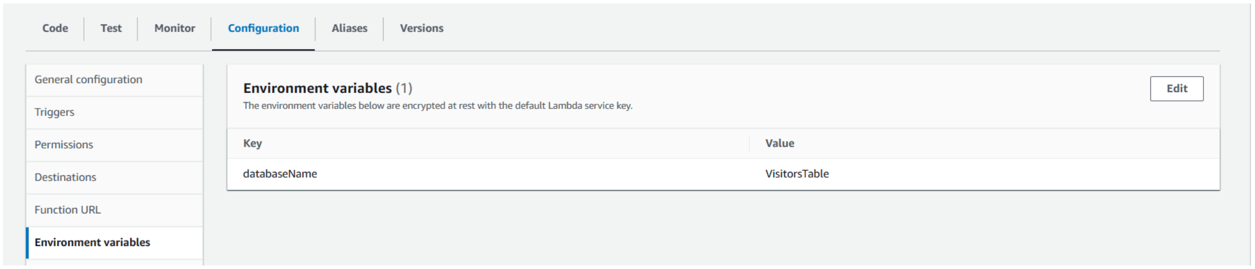


Permissions policies (2) Info
You can attach up to 10 managed policies.

Filter policies by property or policy name and press enter.

Policy name	Type	Description
AWSLambdaBasicExecutionRole-725fb84e-3e48-473f-9b0a-2019a36be102	Customer managed	
AmazonDynamoDBFullAccess	AWS managed	Provides full access to A

Now, as per the lambda code we have, we need to provide Environment Variables to the Lambda Function.
Key: databaseName Value: VisitorsTable

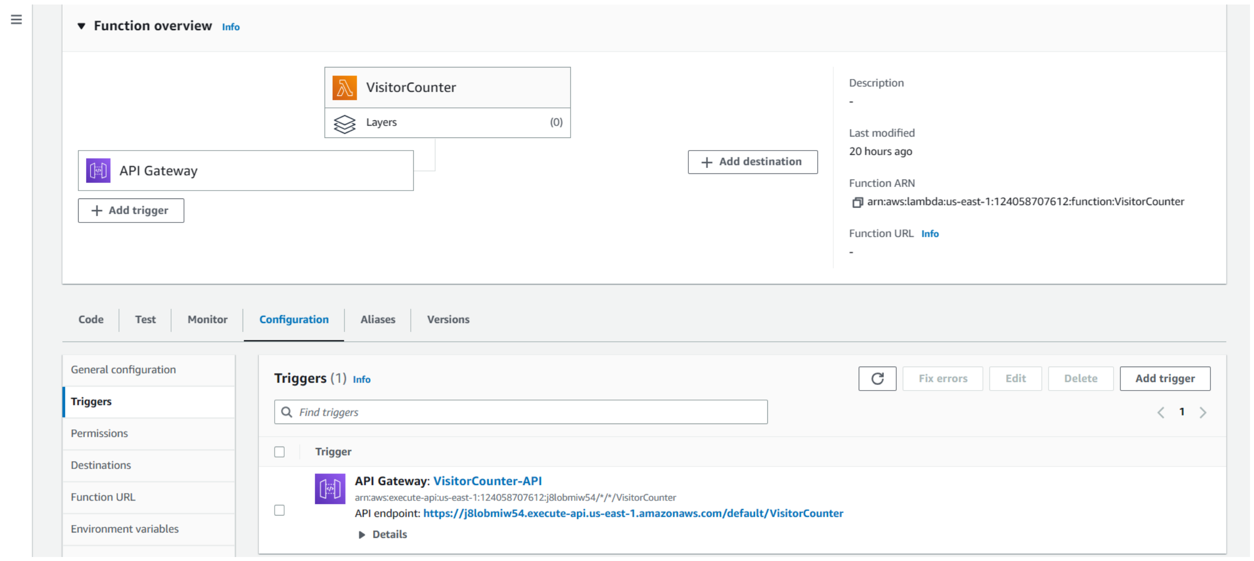


Environment variables (1)

The environment variables below are encrypted at rest with the default Lambda service key.

Key	Value
databaseName	VisitorsTable

Now, we are ready to add API Gateway as a trigger to this lambda function.
We create an HTTP API, with CORS on.



Description
-

Last modified
20 hours ago

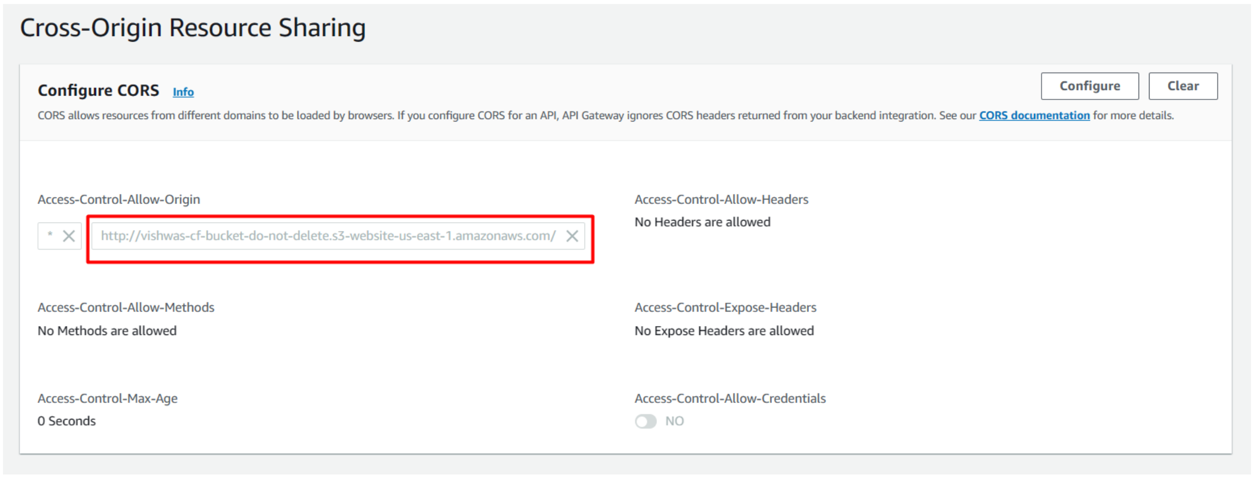
Function ARN
arn:aws:lambda:us-east-1:124058707612:function:VisitorCounter

Function URL Info
-

Triggers (1) Info

Trigger
API Gateway: VisitorCounter-API arn:aws:execute-api:us-east-1:124058707612:j8lobmiw54/*/*/VisitorCounter API endpoint: https://j8lobmiw54.execute-api.us-east-1.amazonaws.com/default/VisitorCounter

Go to API and provide the S3 website url under Access-Control-Allow-Origin.



Cross-Origin Resource Sharing

Configure CORS Info

CORS allows resources from different domains to be loaded by browsers. If you configure CORS for an API, API Gateway ignores CORS headers returned from your backend integration. See our CORS documentation for more details.

Access-Control-Allow-Origin
* http://vishwas-cf-bucket-do-not-delete.s3-website-us-east-1.amazonaws.com/

Access-Control-Allow-Headers
No Headers are allowed

Access-Control-Allow-Methods
No Methods are allowed

Access-Control-Expose-Headers
No Expose Headers are allowed

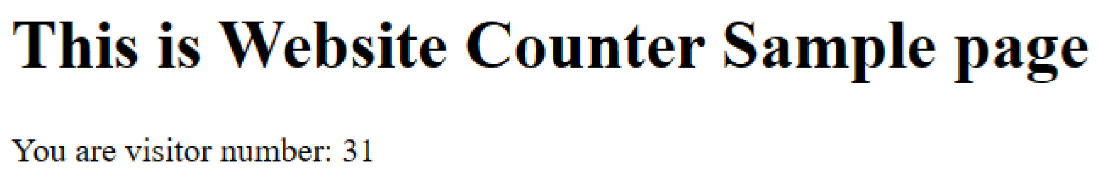
Access-Control-Max-Age
0 Seconds

Access-Control-Allow-Credentials
NO

Now, provide the API Endpoint in this part of the website code to integrate the API with the website.

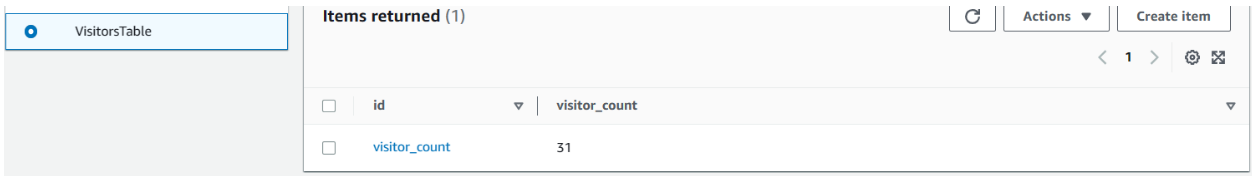
```
 try {
      let response = await
fetch('https://j8lobmiw54.execute-api.us-east-1.amazonaws.com/default/VisitorCounter', {
          method: 'GET',
      });
```

Now, make a request to the website, you should be able to see Visitor Count getting updated in real time.

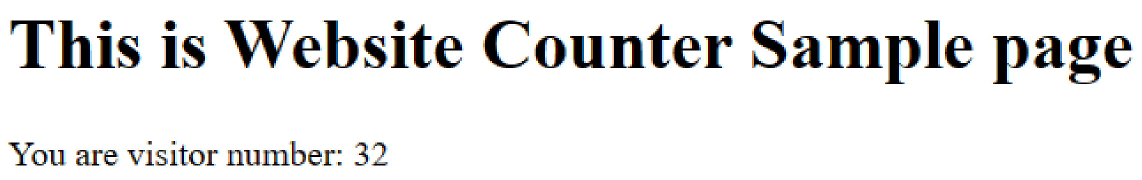


This is Website Counter Sample page

You are visitor number: 31

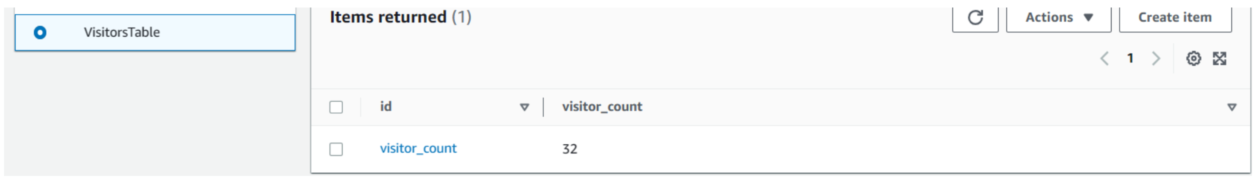


	id	visitor_count
☐	visitor_count	31



This is Website Counter Sample page

You are visitor number: 32



	id	visitor_count
☐	visitor_count	32