


## IAM Policies for the lambda function which grants it minimal permissions


This is how the IAM Role looks like:


Role name  
serverless-form-role [↗](#)

### Resource summary

To view the resources and actions that your function has permission to access, choose a service.

 Amazon CloudWatch Logs  
3 actions, 1 resource

 Amazon CloudWatch Logs  
3 actions, 1 resource

 Amazon DynamoDB  
3 actions, 1 resource

**All resources**

Allow: logs:CreateLogGroup  
Allow: logs:CreateLogStream  
Allow: logs:PutLogEvents

### Permissions policies (2) [Info](#)

You can attach up to 10 managed policies.

Filter by Type

All types

☐

**Policy name** [↗](#)

☐

**Type**

☐

**Attached entities**

<input type="checkbox"/>	<a href="#">AWSLambdaBasicExecutionRole-1cd7e6...</a>	Customer managed	1
<input type="checkbox"/>	<a href="#">dyanamo-perm</a>	Customer inline	0

**dyanamo-perm**

[Copy JSON](#) [Edit](#) [↗](#)

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Sid": "VisualEditor0",  
6       "Effect": "Allow",  
7       "Action": [  
8         "dynamodb:PutItem",  
9         "dynamodb:GetItem",  
10        "dynamodb:ListTables"  
11      ],  
12      "Resource": "*"   
13    }  
14  ]  
15 }
```

I have created an inline custom policy for the lambda function to access the DynamoDB Table,

- List tables is being used to fetch the table that has been created by me
- Put Item is being used to put the values received from the form into the table created, i.e email, name, phone no., intro.

Inline Policy Code:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "dynamodb:PutItem",
        "dynamodb:ListTables",
        "dynamodb:GetItem"
      ],
      "Resource": "*"
    }
  ]
}
```