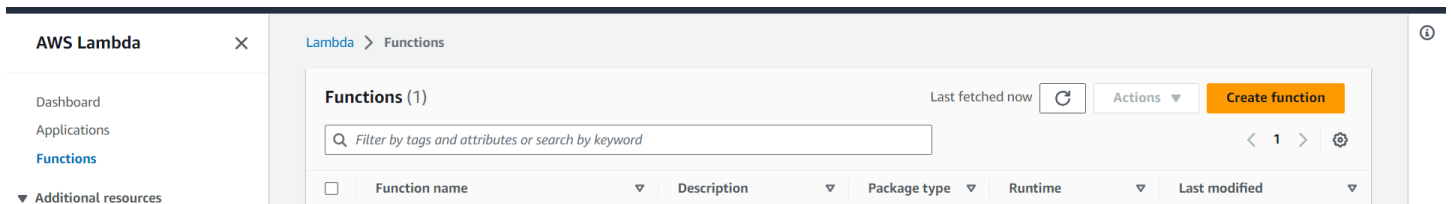# Serverless Function on AWS

Aim: Create a serverless function using AWS Lambda, which allows you to run code without provisioning or managing servers. Develop a simple function (e.g., a function that generates random numbers) and trigger it through API Gateway.

Steps 1: Set Up AWS Lambda Function

a) Navigate to the Lambda service.
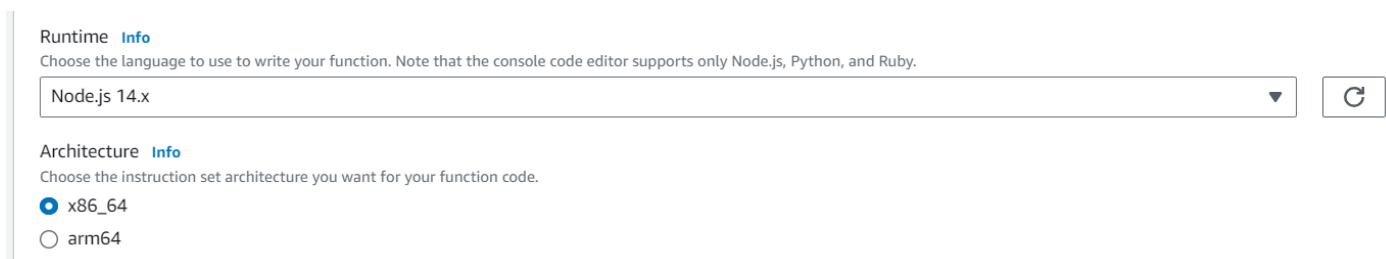b) Click the "Create function" button.

c) Choose "Author from scratch" and fill in the following details:
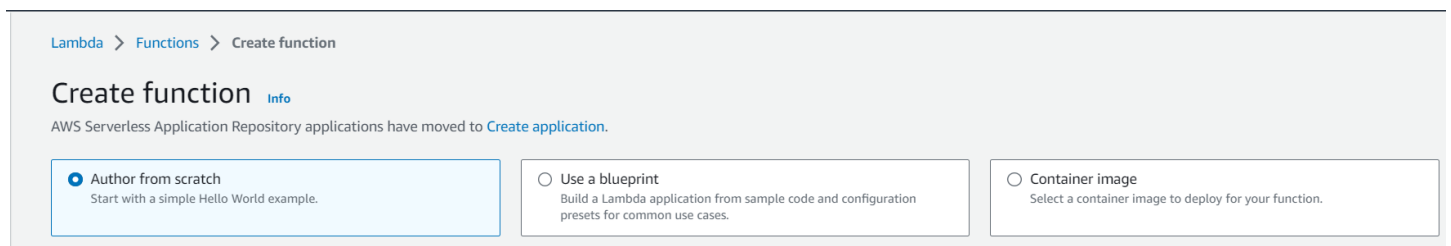
d) Function name: "RandomNoGenerator".

e) Runtime: Choose the runtime you prefer (e.g., Node.js 14.x)

f) Click the "Create function" button.

g) In the "Function code" section, you can paste the following Node.js code that generates a random number:

```javascript
exports.handler = async (event) => {
    const randomNumber = Math.floor(Math.random() * 100);

    const response = {
        statusCode: 200,
        body: JSON.stringify({
            randomNumber: randomNumber
        }),
    };
    return response;
};
```
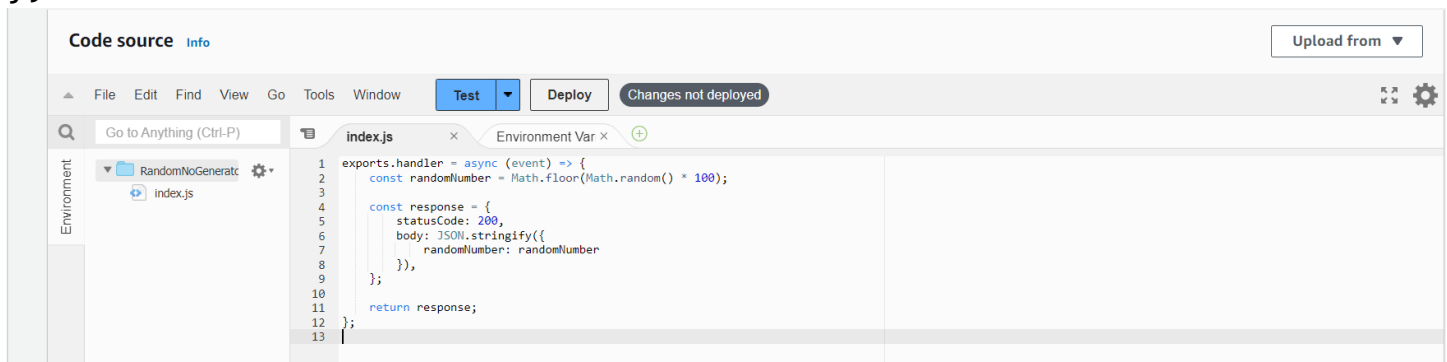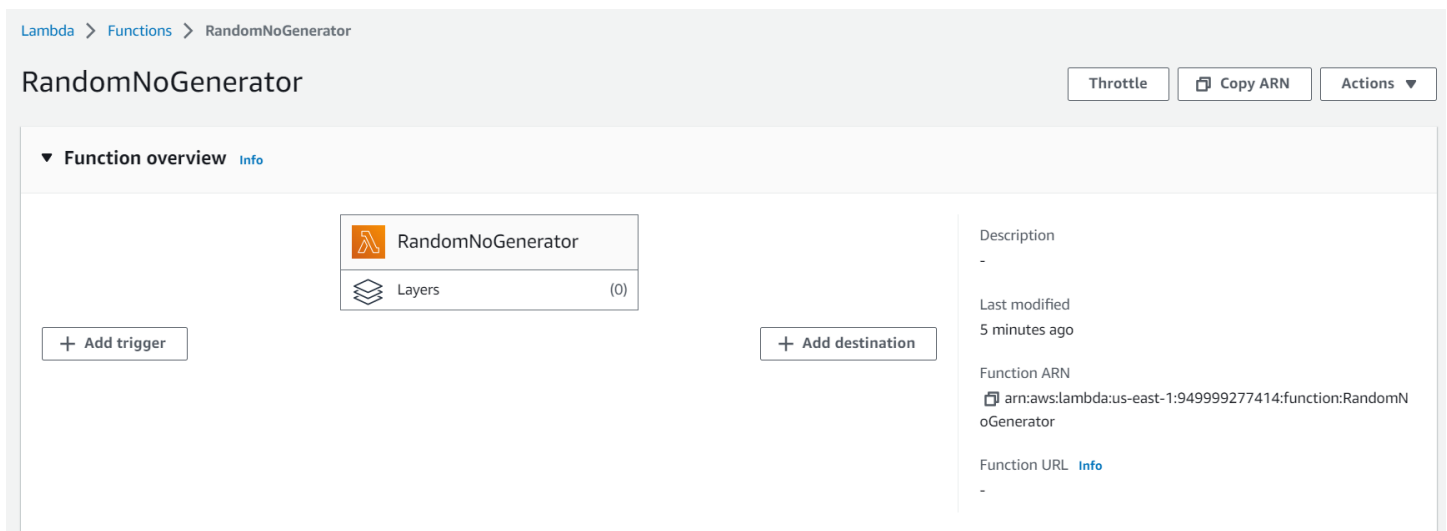

Step 2: Set Up API Gateway

a) Once the Lambda function is created, click on the "Add trigger" button.

b) Choose "API Gateway" as the trigger type.

Lambda > Add trigger

Add trigger

Trigger configuration Info

> API Gateway
> aws   api   application-services   backend   HTTP   REST   serverless   ▼

Add an API to your Lambda function to create an HTTP endpoint that invokes your function. API Gateway supports two types of RESTful APIs: HTTP APIs and REST APIs. Learn more

c) In the "Configure triggers" section, choose "Create an API" and fill in the following details:

Intent
Use an existing api or have us create one for you.
● Create a new API
○ Use existing API

API type

● HTTP API
Build low-latency and cost-effective REST APIs with built-in features such as OIDC and OAuth2, and native CORS support.

○ REST API
Develop a REST API where you gain complete control over the request and response along with API management capabilities.

Security
Configure the security mechanism for your API endpoint.

Open   ▼

d) API name: RandomNoGenerator-API

▼ **Additional settings**

API name
Choose a name for your API. API names don't need to be unique.

RandomNoGenerator-API

e) Deployment stage: [Create new stage]

Deployment stage
The name of your API's deployment stage.

RNGS

f) Click the "Add" button to create the API Gateway.

Lambda will add the necessary permissions for Amazon API Gateway to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model.

Cancel   Add

Step 3: Test and Deploy the code.

a) Select Test event action as "Create new event" and Event name: "TestEvent" and Click on Save Button.

Configure test event   ✕

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

TestEvent

Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

b) Then test the code and deploy it.

| index.js ✕ | Environment Var ✕ | Execution result: ✕ | ⊕ |

▼ Execution results   Status: Succeeded | Max memory used: 58 MB | Time: 76.56 ms

Test Event Name
TestEvent

Response
```
{
  "statusCode": 200,
  "body": "{\"randomNumber\":59}"
}
```

Function Logs
START RequestId: ca171cc5-26fe-4a1b-983d-66a2ec810f77 Version: $LATEST
END RequestId: ca171cc5-26fe-4a1b-983d-66a2ec810f77
REPORT RequestId: ca171cc5-26fe-4a1b-983d-66a2ec810f77 Duration: 76.56 ms Billed Duration: 77 ms Memory Size: 128 MB Max Memory Used: 58 MB

Request ID
ca171cc5-26fe-4a1b-983d-66a2ec810f77

c) Go to Configuration and in trigger click on API endpoint " https://v8h0pg08b1.execute-api.us-east-1.amazonaws.com/RNGS/RandomNoGenerator"

| Code | Test | Monitor | **Configuration** | Aliases | Versions |

General configuration

Triggers   **Triggers (1)** Info   ⟳ Fix errors Edit Delete Add trigger

Permissions   🔍 Find triggers   ‹ 1 ›

Destinations   ☐ **Trigger**
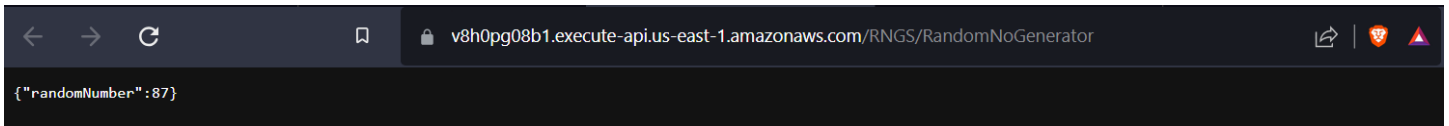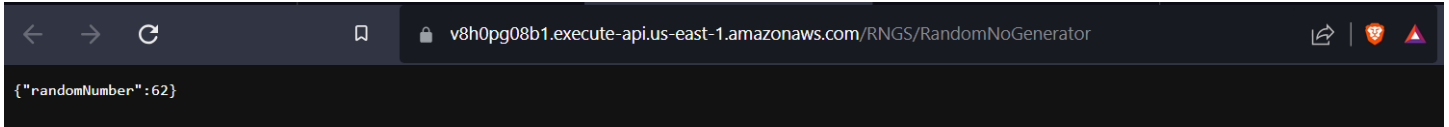
Function URL   ☐ 🔲 **API Gateway: RandomNoGenerator-API**
                                            arn:aws:execute-api:us-east-1:949999277414:v8h0pg08b1/*/*/RandomNoGenerator
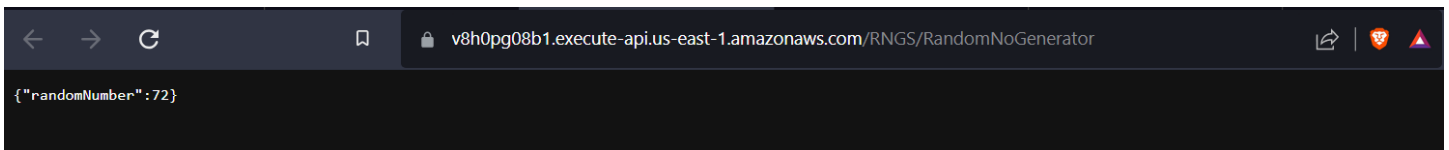Environment variables   API endpoint: **https://v8h0pg08b1.execute-api.us-east-1.amazonaws.com/RNGS/RandomNoGenerator**
                                            ▶ Details
Tags

d) The result


```
v8h0pg08b1.execute-api.us-east-1.amazonaws.com/RNGS/RandomNoGenerator
{"randomNumber":87}
```



```
v8h0pg08b1.execute-api.us-east-1.amazonaws.com/RNGS/RandomNoGenerator
{"randomNumber":62}
```



```
v8h0pg08b1.execute-api.us-east-1.amazonaws.com/RNGS/RandomNoGenerator
{"randomNumber":72}
```

Conclusion: We have successfully created a serverless function using AWS Lambda and triggered it through API Gateway. This simple example demonstrates the power of serverless computing and event-driven architectures. we can further enhance this project by adding more complex logic to your Lambda function and implementing different types of triggers and event sources.