

# Experiment No: 03

## Working with Amazon DynamoDB

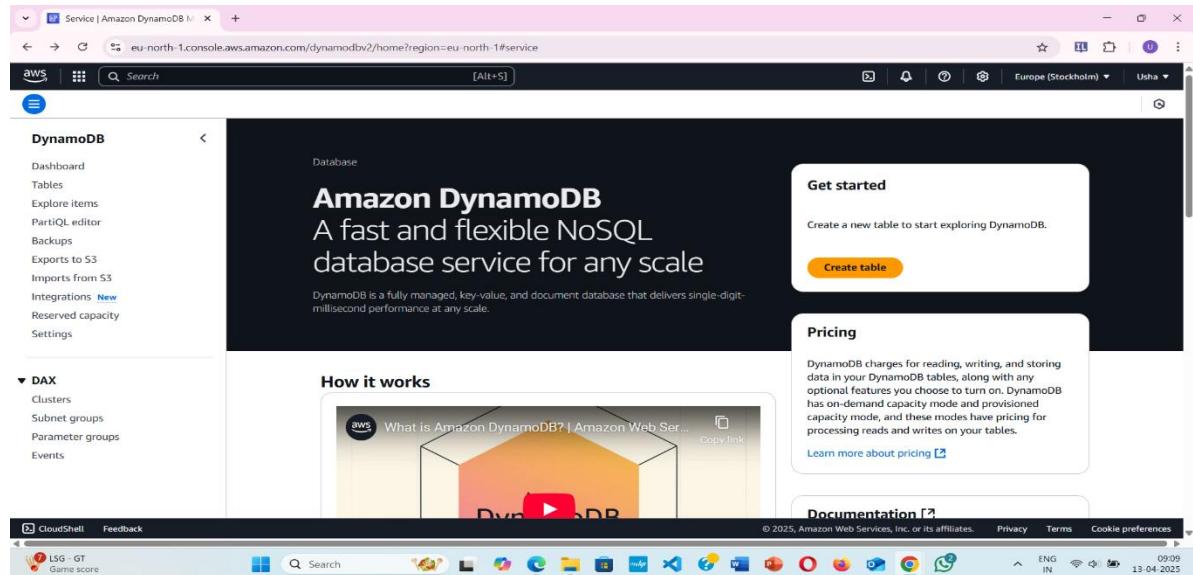
**Apparatus Required:** AWS Account, AWS Dynamo DB

**Pre-Requisite:** Aws basics, DBMS basics

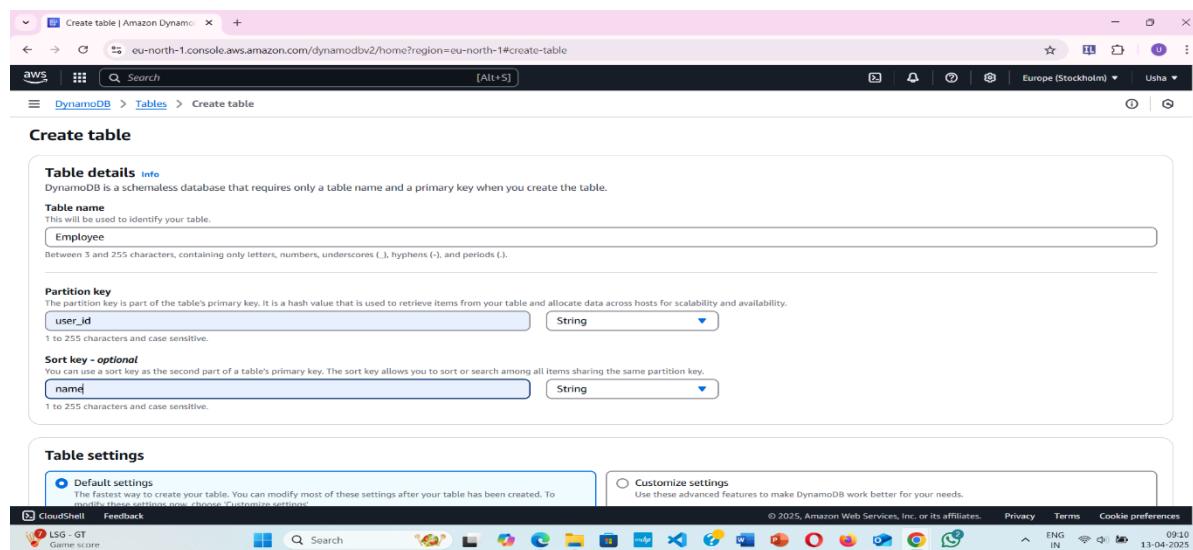
**Introduction:** AWS DynamoDB is a fast and flexible NoSQL database service for all applications that need consistent, single-digit-millisecond latency at any scale. Its flexible data model and reliable performance make DynamoDB a great fit for mobile, web, gaming, advertising technology, Internet of Things, and other applications.

### Procedure

**Navigate to Amazon DynamoDB → click on create table**



**Give table name, Partition Key (is a primary key)**



## Successfully created the table

The screenshot shows the Amazon DynamoDB console. A green success message at the top right says "The Employee table was created successfully." Below it, the "Tables (1) Info" section displays the "Employee" table. The table has one item: user\_id (S) and name (S). The table status is Active, and its provisioned capacity is 0. The "Actions" dropdown menu is visible, showing options like "Update settings", "Explore items", and "Create table". The left sidebar shows navigation links for Dashboard, Tables, Explore items, PartiQL editor, Backups, Exports to S3, Imports from S3, Integrations, Reserved capacity, and Settings. The bottom of the screen shows the Windows taskbar with various pinned icons.

Table created → select your table → explore table items → create items

This screenshot is similar to the previous one, but the "Actions" dropdown menu is now open over the table list. It contains options such as "Update settings", "Explore items" (which is highlighted in blue), "Add tag to selection", "Remove tags from selection", and "Turn on deletion protection". The rest of the interface, including the table list and sidebar, remains the same.

Screenshot of the Amazon DynamoDB console showing the 'Employee' table exploration page.

The left sidebar shows navigation links for DynamoDB (Dashboard, Tables, Explore items, PartiQL editor, Backups, Exports to S3, Imports from S3, Integrations, Reserved capacity, Settings) and DAX (Clusters, Subnet groups, Parameter groups, Events).

The main area displays the 'Employee' table selection screen. It includes fields for 'Select a table or index' (Table - Employee) and 'Select attribute projection' (All attributes). A 'Run' button is present, and a green status bar indicates 'Completed - Items returned: 0 - Items scanned: 0 - Efficiency: 100% - RCU consumed: 2'.

The 'Table: Employee - Items returned (0)' section shows a message 'Scan started on April 13, 2025, 09:11:07'. It displays a table with two columns: 'Actions' and 'Create item'. Below the table, it says 'No items' and 'No items to display.'

The browser address bar shows the URL: <https://eu-north-1.console.aws.amazon.com/dynamodbv2/home?region=eu-north-1#item-explorer?table=Employee>.

Screenshot of the Amazon DynamoDB console showing the 'Create item' dialog for the 'Employee' table.

The top navigation bar shows the URL: [https://eu-north-1.console.aws.amazon.com/dynamodbv2/home?region=eu-north-1#edit-item?itemMode=1&route=ROUTE\\_ITEM\\_EXPLORER&table=Employee](https://eu-north-1.console.aws.amazon.com/dynamodbv2/home?region=eu-north-1#edit-item?itemMode=1&route=ROUTE_ITEM_EXPLORER&table=Employee).

The dialog title is 'Create item' under 'DynamoDB > Explore items: Employee > Create item'. It has tabs for 'Form' (selected) and 'JSON view'.

The 'Attributes' section contains two entries:

Attribute name	Value	Type
user_id - Partition key	CS100	String
name - Sort key	Usha	String

Buttons at the bottom include 'Cancel' and 'Create item'.

The browser address bar shows the URL: [https://eu-north-1.console.aws.amazon.com/dynamodbv2/home?region=eu-north-1#edit-item?itemMode=1&route=ROUTE\\_ITEM\\_EXPLORER&table=Employee](https://eu-north-1.console.aws.amazon.com/dynamodbv2/home?region=eu-north-1#edit-item?itemMode=1&route=ROUTE_ITEM_EXPLORER&table=Employee).

The item has been saved successfully.

**Employee**

Scan or query items

Tables (1)

Employee

Select a table or index

Table - Employee

Autopreview View table details

Completed Items returned: 0 - Items scanned: 0 Efficiency: 100% RCU's consumed: 2

Table: Employee - Items returned (1)

Scan started on April 13, 2025, 09:11:07

Actions Create Item

user\_id (String) name (String)

CS100 Usha

CloudShell Feedback

Air: Moderate Today

Search

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

09:12 13-04-2025

## Using JSON to create items

Create item

Form JSON view

You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep. Learn more [»](#)

Attributes View DynamoDB JSON

```
1 ↴ {  
2   "user_id": {  
3     "S": "CS100"  
4   },  
5   "name": {  
6     "S": "Usha"  
7   }  
8 }
```

Cancel Create item

CloudShell Feedback

Finance headline US Fed eyes tarif...

Search

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

09:15 13-04-2025

The item has been saved successfully.

**Employee**

Scan or query items

Select a table or index: Table - Employee

Completed - Items returned: 0 - Items scanned: 0 - Efficiency: 100% - RCU consumed: 2

	user_id (String)	name (String)
CS99	Usha	
CS100	Usha	

## DynamoDB → PartiQL editor → write the queries

### Table view

Query 1

```
1 SELECT * FROM "Employee"
```

Run Clear

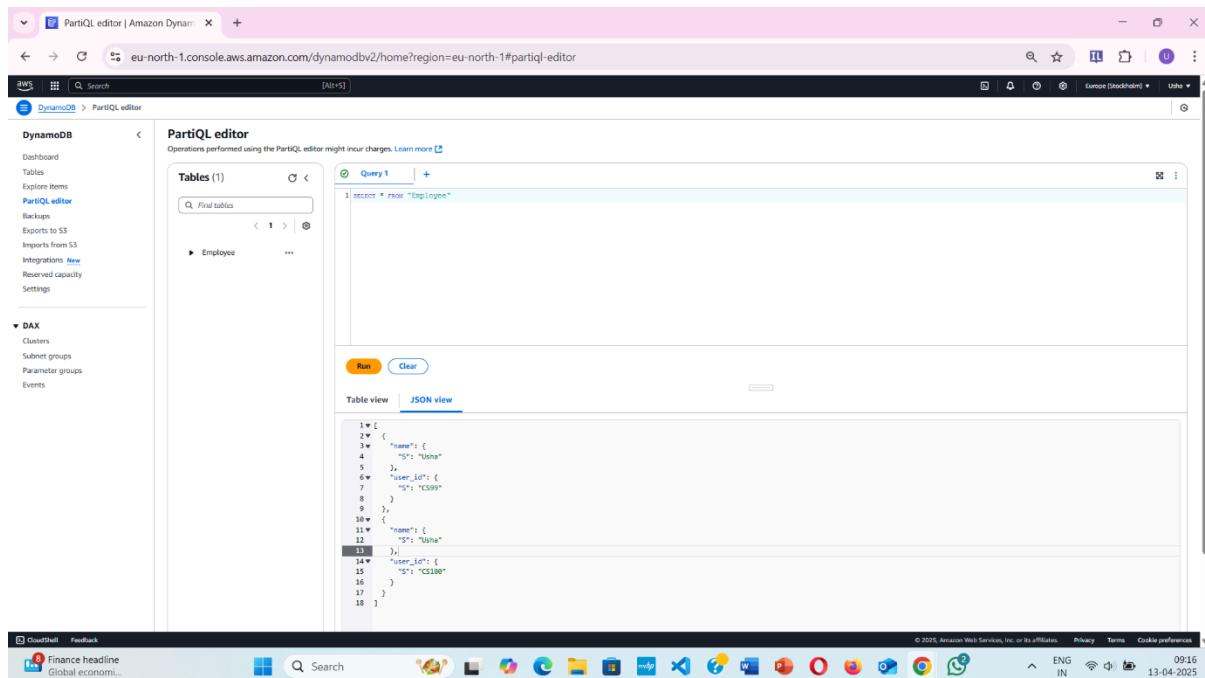
Table view JSON view

Completed  
Started on 4/13/2025, 9:15:50 AM  
Elapsed time 226ms

Items returned (2)

	user_id (String)	name (String)
CS99	Usha	
CS100	Usha	

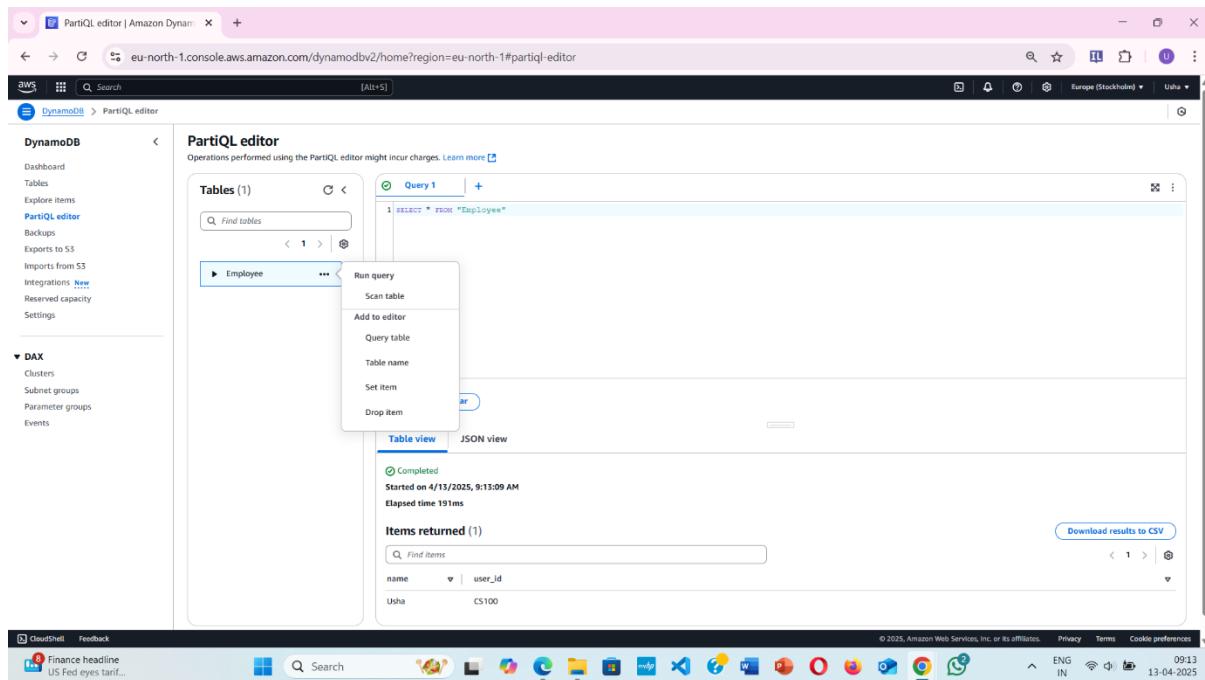
## JSON view



The screenshot shows the AWS PartiQL editor interface. On the left, the navigation sidebar includes 'DynamoDB' and 'DAX' sections. In the main area, a 'Tables (1)' section lists 'Employee'. Below it, a 'Query 1' tab contains the SQL query: 'SELECT \* FROM "Employee"'. The results are displayed in 'JSON view', showing two items:

```
1 | {
2 |   "name": {
3 |     "$": "Usha"
4 |   },
5 |   "user_id": {
6 |     "$": "CS099"
7 |   }
8 | },
9 | {
10|   "name": {
11|     "$": "Usha"
12|   },
13|   "user_id": {
14|     "$": "CS100"
15|   }
16| }
17 |
18 }
```

## Update Command



The screenshot shows the AWS PartiQL editor interface. The 'Employee' table is selected. A context menu is open over the table name, with 'Run query' highlighted. The 'Table view' tab is active, showing the results of the previous query. The results table has one item:

name	user_id
Usha	CS100

## Delete table

The screenshot shows the AWS DynamoDB console with the URL [eu-north-1.console.aws.amazon.com/dynamodbv2/home?region=eu-north-1#tables](https://eu-north-1.console.aws.amazon.com/dynamodbv2/home?region=eu-north-1#tables). The left sidebar shows the navigation menu for DynamoDB, including 'Dashboard', 'Tables', 'Explore items', 'PartiQL editor', 'Backups', 'Exports to S3', 'Imports from S3', 'Integrations', 'Reserved capacity', and 'Settings'. Under 'Tables', there is a sub-menu for 'DAX' with options 'Clusters', 'Subnet groups', 'Parameter groups', and 'Events'. The main content area displays a table titled 'Tables (1/1)'. It lists a single table named 'Employee' with the following details: Status: Active, Partition key: user\_id (\$), Sort key: name (\$), Indexes: 0, Replication Regions: 0, Deletion protection: Off, Favorite: Not favorited, Read capacity mode: On-demand. Below the table, there is a message: 'CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 09:38 13-04-2025'. The message bar also includes a link to 'Finance headline US Fed eyes tariff...'.

The screenshot shows the AWS DynamoDB console with the URL [eu-north-1.console.aws.amazon.com/dynamodbv2/home?region=eu-north-1#tables](https://eu-north-1.console.aws.amazon.com/dynamodbv2/home?region=eu-north-1#tables). The left sidebar shows the navigation menu for DynamoDB, including 'Dashboard', 'Tables', 'Explore items', 'PartiQL editor', 'Backups', 'Exports to S3', 'Imports from S3', 'Integrations', 'Reserved capacity', and 'Settings'. Under 'Tables', there is a sub-menu for 'DAX' with options 'Clusters', 'Subnet groups', 'Parameter groups', and 'Events'. The main content area displays a table titled 'Tables (0)'. A green message bar at the top says 'The request to delete the "Employee" table has been submitted successfully.' Below the table, there is a message: 'CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 09:39 13-04-2025'. The message bar also includes a link to 'Finance headline Global econom...'. A cartoon robot icon is displayed above the message 'You have no tables in this account in this AWS Region.' The 'Create table' button is visible at the bottom of the table list.

## Observations:

Tables and items are created, and we can run the commands to manipulate the data in the tables.

# Experiment No – 04

## Developing REST APIs with Amazon API Gateway

**Objectives:** Developing REST APIs with Amazon API Gateway.

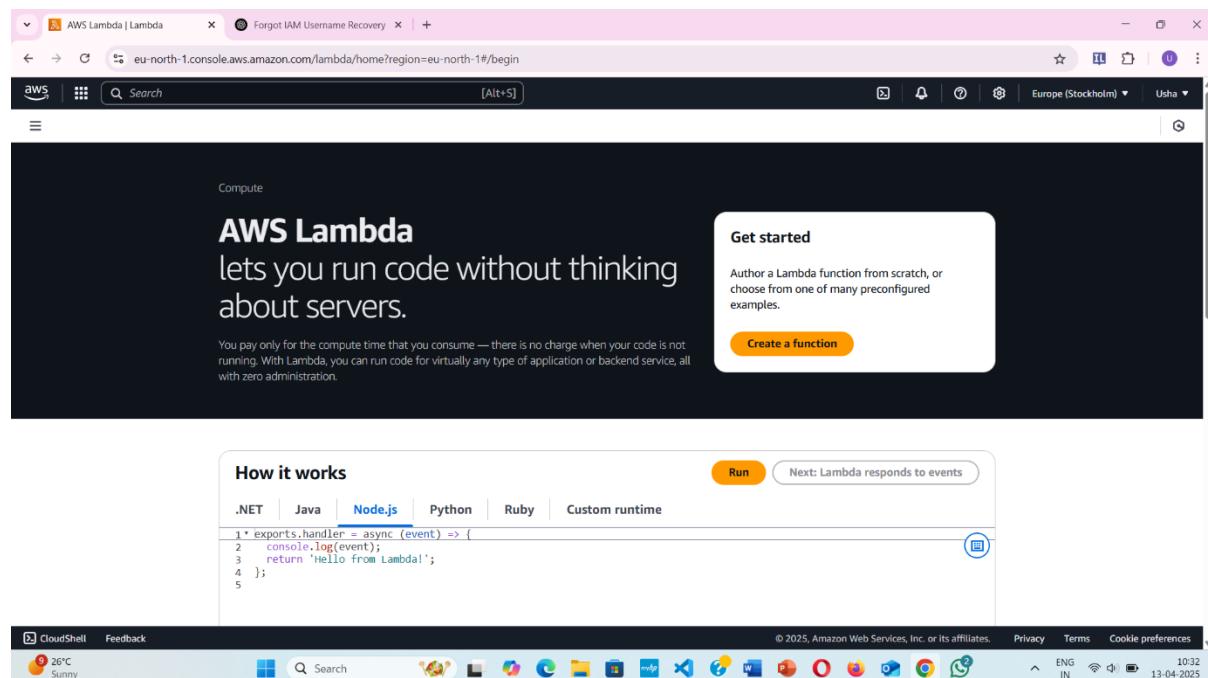
**Apparatus Required:** AWS Account

**Pre-Requisite:** Pre - requisite is creation of Lambda Function

**Introduction:** API Gateway enables you to connect and access data, business logic, and functionality from backend services such as workloads running on Amazon Elastic Compute Cloud (Amazon EC2), code running on AWS Lambda, any web application, or real-time communication applications.

## Procedure

### Open AWS Lambda



## Create function

The screenshot shows the 'Create function' wizard on the AWS Lambda console. The 'Basic information' step is active. It includes fields for 'Function name' (set to 'Rit'), 'Runtime' (set to 'Python 3.15'), 'Architecture' (set to 'x86\_64'), and 'Permissions'. A note indicates that Lambda will create an execution role with permissions to upload logs to CloudWatch Logs. Below these, the 'Change default execution role' section shows the 'Create a new role with basic Lambda permissions' option selected. A note states that role creation might take a few minutes. On the right, a sidebar titled 'Create a simple web app' provides a tutorial.



**Create function | Functions | Lambda**

eu-north-1.console.aws.amazon.com/lambda/home?region=eu-north-1#/create/function

Lambda > Functions > Create function

View policy statement

```
1+ {
2+ "Version": "2012-10-17"
3+ "StatementId": [
4+   {
5+     "StatementId": "FunctionRLAllowPublicAccess",
6+     "Effect": "Allow",
7+     "Principal": "*",
8+     "Action": "lambda:InvokeFunction",
9+     "Resource": "arn:aws:lambda:eu-north-1:985539760735:function:Rit",
10+    "Condition": {
11+      "StringEqual": {
12+        "lambda:FunctionArn": "arn:aws:lambda:eu-north-1:985539760735:function:Rit"
13+      }
14+    }
15+  }
16+ ]
17+ }
```

2:27 JSON Spaces: 2

Invoke mode **Info**

Choose how your function returns responses. [Learn more](#)

**BUFFERED (default)**

The invocation results are available when the payload is complete. Response payload max size: 6 MB

**RESPONSE\_STREAM**

Stream the invocation results. Streaming responses incur additional costs. Refer to the documentation for payload size limitations. [Learn more](#)

**Configure cross-origin resource sharing (CORS)**

Use CORS to allow access to your Function URL from any origin. You can also use CORS to control access for specific HTTP headers and methods in requests to your function URL. By default, all origins are allowed. You can edit this after creating the function. [Learn more](#)

**Enable tags [Info](#)**

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources, track your AWS costs, and enforce attribute-based access control.

**Enable VPC [Info](#)**

Connect your function to a VPC to access private resources during invocation.

[Cancel](#) [Create function](#)

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 10:52 13-04-2025

## Successfully created the function

**Rit | Functions | Lambda**

eu-north-1.console.aws.amazon.com/lambda/home?region=eu-north-1#/functions/Rit?newFunction=true&tab=code

Lambda > Functions > Rit

Successfully created the function Rit. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

**Diagram** **Template**

Rit

Layers (0)

+ Add trigger + Add destination

**Description**

Last modified 52 seconds ago

**Function ARN**

arn:aws:lambda:eu-north-1:985539760735:function:Rit

**Function URL** [Info](https://t5v7tispc6olg6zgaia2i6bby0gxth.lambda-url.eu-north-1.on.aws/)

[Learn more](#) [Start tutorial](#)

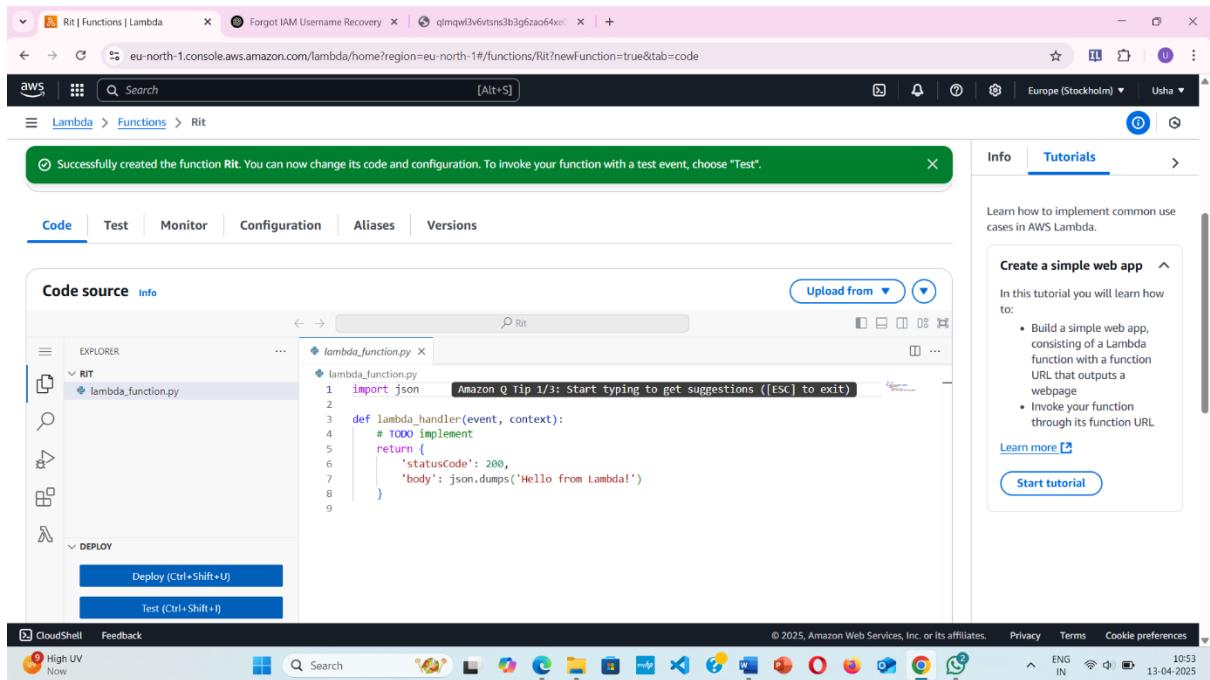
**Code** **Test** **Monitor** **Configuration** **Aliases** **Versions**

**Code source** [Info](#)

Upload from

EXPLORER lambda\_function.py

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 10:53 13-04-2025



Code source Info

```
lambda_function.py
1 import json
2
3 def lambda_handler(event, context):
4     # TODO implement
5     return {
6         'statusCode': 200,
7         'body': json.dumps('Hello from Lambda!')
8     }
9
```

Upload from ...

EXPLORER

RIT

lambda\_function.py

Deploy

Deploy (Ctrl+Shift+U)

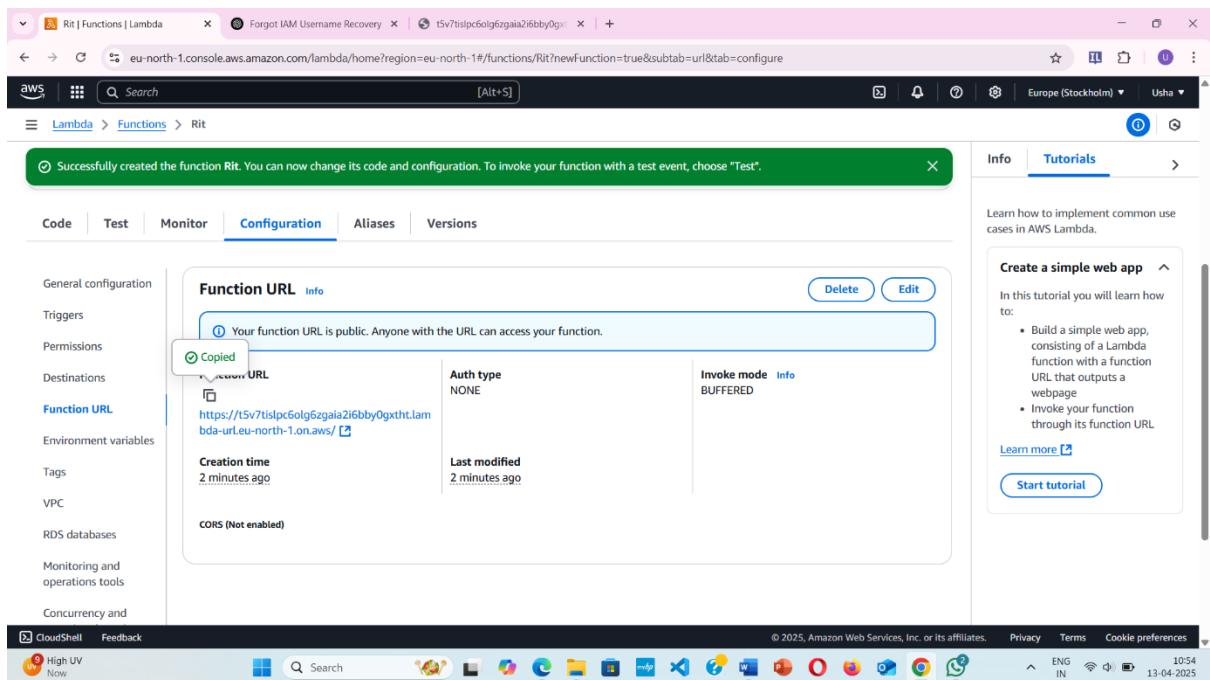
Test (Ctrl+Shift+I)

CloudShell Feedback

High UV Now

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 10:53 13-04-2025

## Configuration → function URL → check URL runs



Code Test Monitor Configuration Aliases Versions

General configuration

Triggers

Permissions

Destinations

**Function URL**

Environment variables

Tags

VPC

RDS databases

Monitoring and operations tools

Concurrency and

Function URL Info

Your function URL is public. Anyone with the URL can access your function.

Copied

Recent URL

https://t5v7islp6olg6gzaia2i6bby0gxth.lambdashurl.eu-north-1.on.aws/

Auth type

NONE

Invoke mode

BUFFERED

Creation time

2 minutes ago

Last modified

2 minutes ago

CORS (Not enabled)

CloudShell Feedback

High UV Now

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 10:54 13-04-2025

## Lambda Code Samples

```
import json

print(json.dumps({"name": "John", "age": 30}))

print(json.dumps(["apple", "banana"]))

print(json.dumps(("apple", "bananas")))

print(json.dumps("hello"))

print(json.dumps(42))

print(json.dumps(31.76))

print(json.dumps(True))

print(json.dumps(False))

print(json.dumps(None))
```

## Go to API Gateway in Console And Select Rest API

The screenshot shows the AWS API Gateway - Create API page. The URL is eu-north-1.console.aws.amazon.com/apigateway/main/apis?region=eu-north-1. The page title is "API Gateway - Create API". The main content area is titled "REST API" and contains the following text:  
"Develop a REST API where you gain complete control over the request and response along with API management capabilities."  
"Works with the following:"  
Lambda, HTTP, AWS Services  
At the bottom right of this section are "Import" and "Build" buttons.  
Below this section is another titled "REST API Private" with similar content and "Import" and "Build" buttons.

## Build Rest API

The screenshot shows the 'Create REST API' wizard in the AWS Management Console. The 'API details' step is selected, showing options for 'New API' (selected), 'Clone existing API', 'Import API', and 'Example API'. The 'API name' is set to 'myfirstapi'. The 'Description - optional' field is empty. Under 'API endpoint type', it says '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.' The 'Regional' option is selected. Under 'IP address type', 'IPv4' is selected. The status bar at the bottom shows 'CloudShell Feedback' and the system tray with weather information.

## Successfully created the REST API

The screenshot shows the 'Resources' page for the 'myfirstapi' API. A green banner at the top says 'Successfully created REST API 'myfirstapi' (ejinnwsp96).'. The left sidebar shows the API gateway navigation menu. The main area displays the 'Resources' section with a single resource entry: 'Path /' under 'Resource details'. The 'Methods (0)' section is empty. The status bar at the bottom shows 'CloudShell Feedback' and the system tray with weather information.

## Create method And Select Lambda Integration

### Select GET method

The screenshot shows the 'Create method' page in the AWS API Gateway console. The 'Method type' dropdown is set to 'GET'. Under 'Integration type', the 'Lambda function' option is selected, highlighted with a blue border. The 'Lambda function' input field contains the ARN: arn:aws:lambda:eu-north-1:985539760735:function:Rit. A green success message at the top right states: 'Successfully created REST API myfirstapi (ejinwsp96)'. The browser address bar shows: eu-north-1.console.aws.amazon.com/apigateway/main/apis/ejinwsp96/resources/3pg9lbrts//create-method?api=ejinwsp96&experience=rest&region=eu-north-1.

The screenshot shows the 'Create method' page in the AWS API Gateway console. The 'Lambda proxy integration' option is selected, highlighted with a blue border. The 'Lambda function' input field contains the ARN: arn:aws:lambda:eu-north-1:985539760735:function:Rit. A blue button at the bottom right says: 'Grant API Gateway permission to invoke your Lambda function. To turn off, update the function's resource policy yourself, or provide an invoke role that API Gateway uses to invoke your function.' The browser address bar shows: eu-north-1.console.aws.amazon.com/apigateway/main/apis/ejinwsp96/resources/3pg9lbrts//create-method?api=ejinwsp96&experience=rest&region=eu-north-1.

## Successfully created method “GET”

The screenshot shows the AWS API Gateway Resources page. A green notification bar at the top says "Successfully created method 'GET' in '/'. Redeploy your API for the update to take effect." On the left, the navigation menu includes "API Gateway", "APIs", "Custom domain names", "Domain name access associations", "VPC links", "AP: myfirstapi", "Resources", "Stages", "Authorizers", "Gateway responses", "Models", "Resource policy", "Documentation", "Dashboard", "API settings", "Usage plans", "API keys", "Client certificates", and "Settings". The main content area shows a "Resources" section with a "Create resource" button. Below it is a "Method execution" diagram for a GET request. The diagram shows a "Client" sending a "Method request" to an "Integration request", which then triggers a "Lambda integration". The "Integration response" is sent back to the "Method response". Below the diagram are tabs for "Method request", "Integration request", "Integration response", "Method response", and "Test". Under "Method request settings", there are fields for "Authorization" (set to "NONE"), "Request validator" (set to "None"), "API key required" (set to "False"), and "SDK operation name" (set to "Generated based on method and path"). There is also a "Request paths (0)" section with a "Name" field and a "Caching" dropdown. The bottom of the screen shows a Windows taskbar with various icons and system status.

## To Test API

### Deploy API

The screenshot shows the AWS API Gateway Resources page with a "Deploy API" dialog box overlaid. The dialog asks to "Create or select a stage where your API will be deployed. You can use the deployment history to revert or change the active deployment for a stage." It has a dropdown for "Stage" set to "New stage\*", a "Stage name" input field containing "CS", and a "Deployment description" input field. At the bottom are "Cancel" and "Deploy" buttons. The background shows the same API Gateway interface as the previous screenshot, including the "Resources" section and the "Method execution" diagram. The bottom of the screen shows a Windows taskbar with various icons and system status.

The screenshot shows the AWS API Gateway Stages console. On the left, a sidebar lists the API (myfirstapi) and its stages (CS). The main area displays the 'Stage details' for stage CS, which is active. It includes configuration for Cache cluster (inactive), Default method-level caching (inactive), and Invoke URL (https://a2eqq9v4hc.execute-api.eu-north-1.amazonaws.com/CS). Below this, the 'Logs and tracing' section shows CloudWatch logs (inactive), X-Ray tracing (inactive), and Data tracing (inactive). A green banner at the top indicates a successful deployment. The bottom of the screen shows the Windows taskbar with various pinned icons and system status.

## Use URL and copy in new tab

The screenshot shows the AWS Lambda function output. The response body is a JSON object: {"statusCode": 200, "body": "\"Hello from Lambda!\""}.

# Experiment No – 5

## Creating Lambda Functions Using the AWS SDK for Python

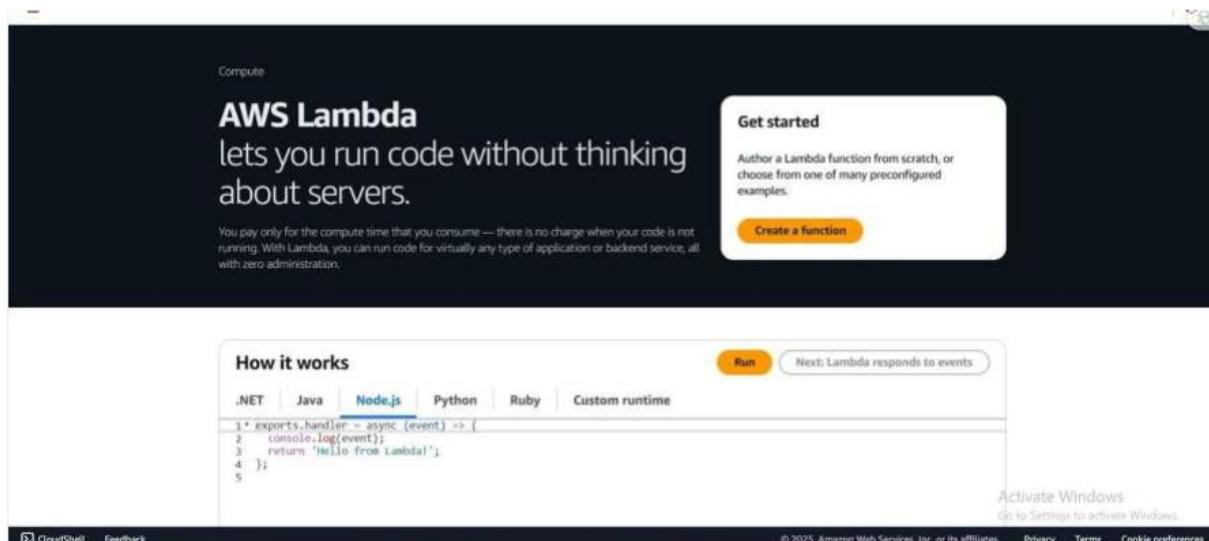
**Objectives:** Creating Lambda Functions Using the AWS SDK for Python

**Apparatus Required:** AWS Account

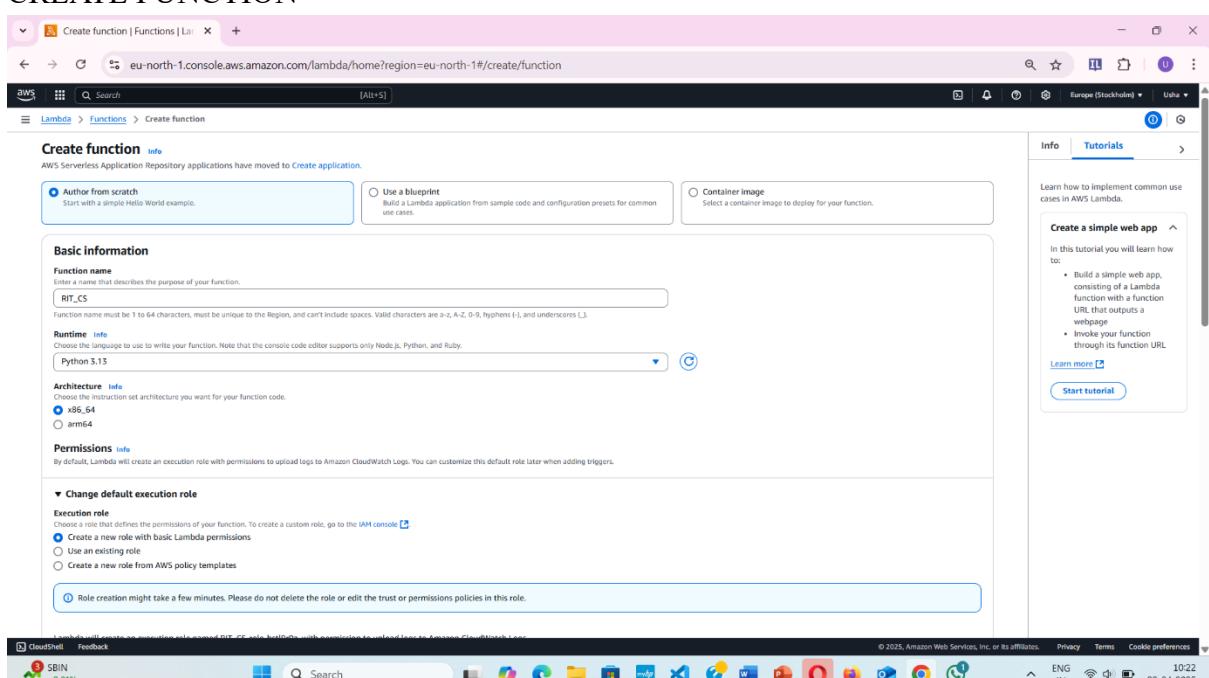
**Pre-Requisite:** AWS basic knowledge, basics of lambda functions and Python

**Introduction:** An AWS Lambda function is a serverless function that runs code in response to events, such as user actions or system updates. You can use Lambda functions to create custom backend services, extend other AWS services, or trigger actions in other services.

### Procedure:



### CREATE FUNCTION



**Create function | Functions | Lambda**

eu-north-1.console.aws.amazon.com/lambda/home?region=eu-north-1#/create/function

Lambda > Functions > Create function

**Additional Configurations**

- Enable Code signing** Info  
Use code signing configurations to ensure that the code has been signed by an approved source and has not been altered since signing.
- Enable encryption with an AWS KMS customer managed key** Info  
By default, Lambda encrypts the .zip file archive using an AWS owned key.
- Enable function URL** Info  
Use function URLs to expose HTTPS endpoints to your Lambda function.

**Auth type**  
Choose the auth type for your function URL. [Learn more](#)

- AWS\_IAM**  
Only authenticated IAM users and roles can make requests to your function URL.
- NONE**  
Lambda won't perform IAM authentication on requests to your function URL. The URL endpoint will be public unless you implement your own authorization logic in your function.

**Function URL permissions**

When you choose auth type **NONE**, Lambda automatically creates the following resource-based policy and attaches it to your function. This policy makes your function public to anyone with the function URL. You can edit the policy later. To limit access to authenticated IAM users and roles, choose auth type **AWS\_IAM**.

```
1 "Version": "2012-10-17",
2 "Statement": [
3     {
4         "StatementId": "FunctionURLAllowPublicAccess",
5         "Effect": "Allow",
6         "Principal": "*",
7         "Action": "lambda:InvokeFunction",
8         "Resource": "arn:aws:lambda:eu-north-1:985539760735:function:RIT_CS",
9         "Condition": {
10             "StringEquals": {
11                 "lambda:FunctionUrlAuthType": "NONE"
12             }
13         }
14     }
15 ]
16
17 ]
```

**View policy statement**

```
1 "Version": "2012-10-17",
2 "Statement": [
3     {
4         "StatementId": "FunctionURLAllowPublicAccess",
5         "Effect": "Allow",
6         "Principal": "*",
7         "Action": "lambda:InvokeFunction",
8         "Resource": "arn:aws:lambda:eu-north-1:985539760735:function:RIT_CS",
9         "Condition": {
10             "StringEquals": {
11                 "lambda:FunctionUrlAuthType": "NONE"
12             }
13         }
14     }
15 ]
16
17 ]
```

**CloudShell Feedback**

SBIN +2.9%

CloudShell Feedback

eu-north-1.console.aws.amazon.com/lambda/home?region=eu-north-1#/create/function

Lambda > Functions > Create function

**When you choose auth type **NONE**, Lambda automatically creates the following resource-based policy and attaches it to your function. This policy makes your function public to anyone with the function URL. You can edit the policy later. To limit access to authenticated IAM users and roles, choose auth type **AWS\_IAM**.**

**View policy statement**

```
1 "Version": "2012-10-17",
2 "Statement": [
3     {
4         "StatementId": "FunctionURLAllowPublicAccess",
5         "Effect": "Allow",
6         "Principal": "*",
7         "Action": "lambda:InvokeFunction",
8         "Resource": "arn:aws:lambda:eu-north-1:985539760735:function:RIT_CS",
9         "Condition": {
10             "StringEquals": {
11                 "lambda:FunctionUrlAuthType": "NONE"
12             }
13         }
14     }
15 ]
16
17 ]
```

**Invoke mode** Info

Choose how your function returns responses. [Learn more](#)

- BUFFERED (default)**  
The invocation results are available when the payload is complete. Response payload max size: 6 MB
- RESPONSE\_STREAM**  
Stream the invocation results. Streaming responses incur additional costs. Refer to the documentation for payload size limitations. [Learn more](#)

**Configure cross-origin resource sharing (CORS)**  
Use CORS to allow access to your function URL from any origin. You can also use CORS to control access for specific HTTP headers and methods in requests to your function URL. By default, all origins are allowed. You can edit this after creating the function. [Learn more](#)

**Enable tags** Info  
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources, track your AWS costs, and enforce attribute-based access control.

**Enable VPC** Info  
Connect your function to a VPC to access private resources during invocation.

**Tutorials**

Learn how to implement common use cases in AWS Lambda.

**Create a simple web app**

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#) [Start tutorial](#)

**Cancel** **Create function**

CloudShell Feedback

SBIN +2.9%

CloudShell Feedback

eu-north-1.console.aws.amazon.com/lambda/home?region=eu-north-1#/create/function

Lambda > Functions > Create function

**Learn how to implement common use cases in AWS Lambda.**

**Create a simple web app**

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#) [Start tutorial](#)

Screenshot of the AWS Lambda console showing the creation of a new function named RIT\_CS.

**Function Overview:**

- Description:** Last modified 55 seconds ago.
- Function ARN:** arn:aws:lambda:eu-north-1:985539760735:function:RIT\_CS
- Function URL:** https://mofyiptcw7xekdz3gdxv54vcu0rcwed.lambda-url.eu-north-1.on.aws/

**Tutorials:** Create a simple web app

**Code Source:** lambda\_function.py

```

1 import json
2
3 def lambda_handler(event, context):
4     # TODO implement
5     return {
6         'statusCode': 200,
7         'body': json.dumps('Hello from Lambda!')
8     }
9

```

**Actions:** Throttle, Copy ARN, Actions ▾

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

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

**Code source**

**EXPLORER**: RIT\_CS, lambda\_function.py

**DEPLOY**: Deploy (Ctrl+Shift+U), Test (Ctrl+Shift+I)

**TEST EVENTS [NONE SELECTED]**

**CloudShell | Feedback**

**Search**

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 10:23 22-04-2025

## CONFIGURATIONS → FUNCTION URL → CHECK URL RUNS

The screenshot shows the AWS Lambda Functions configuration page for a function named RIT\_CS. The 'Configuration' tab is selected. In the 'Function URL' section, it displays the public URL: <https://mofyiptcw7ekdz3ggxkv54vcu0rcwed.lambda-url.eu-north-1.on.aws/>. A tooltip indicates that the URL is copied. Other details shown include 'Auth type: NONE' and 'Invoke mode: BUFFERED'. The creation time was '1 minute ago' and the last modified time was also '1 minute ago'. CORS is noted as 'Not enabled'. On the right side, there is a 'Tutorials' sidebar with a link to 'Create a simple web app'.

Successfully created the function RIT\_CS. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

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

General configuration  
Triggers  
Permissions  
Destinations  
**FUNCTION URL**  
Environment variables  
Tags  
VPC  
RDS databases  
Monitoring and operations tools

Function URL Info

Your function URL is public. Anyone with the URL can access your function.

Copied

Function URL https://mofyiptcw7ekdz3ggxkv54vcu0rcwed.lambda-url.eu-north-1.on.aws/

Auth type: NONE

Invoke mode: BUFFERED

Creation time: 1 minute ago

Last modified: 1 minute ago

CORS (Not enabled)

Info | Tutorials

Create a simple web app

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

Learn more ↗

Start tutorial

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 10:23 22-04-2025

mofyiptcw7ekdz3ggxkv54vcu0rcwed.lambda-url.eu-north-1.on.aws

Pretty-print □

"Hello from Lambda!"



# Experiment No – 06

## Migrating a Web Application to Docker Containers

**Objectives:** Migrating a Web Application to Docker Containers

**Apparatus Required:** AWS Account

**Pre-Requisite:** AWS basics knowledge, Basics of docker

**Introduction:** Docker is a set of platform as a service (PaaS) products that use OS-level virtualization to deliver software in packages called containers.

Docker is a software platform that packages software into containers.

Docker images are read-only templates that contain instructions for creating a Container.

A Docker image is a snapshot or blueprint of the libraries and dependencies required inside a container for an application to run.

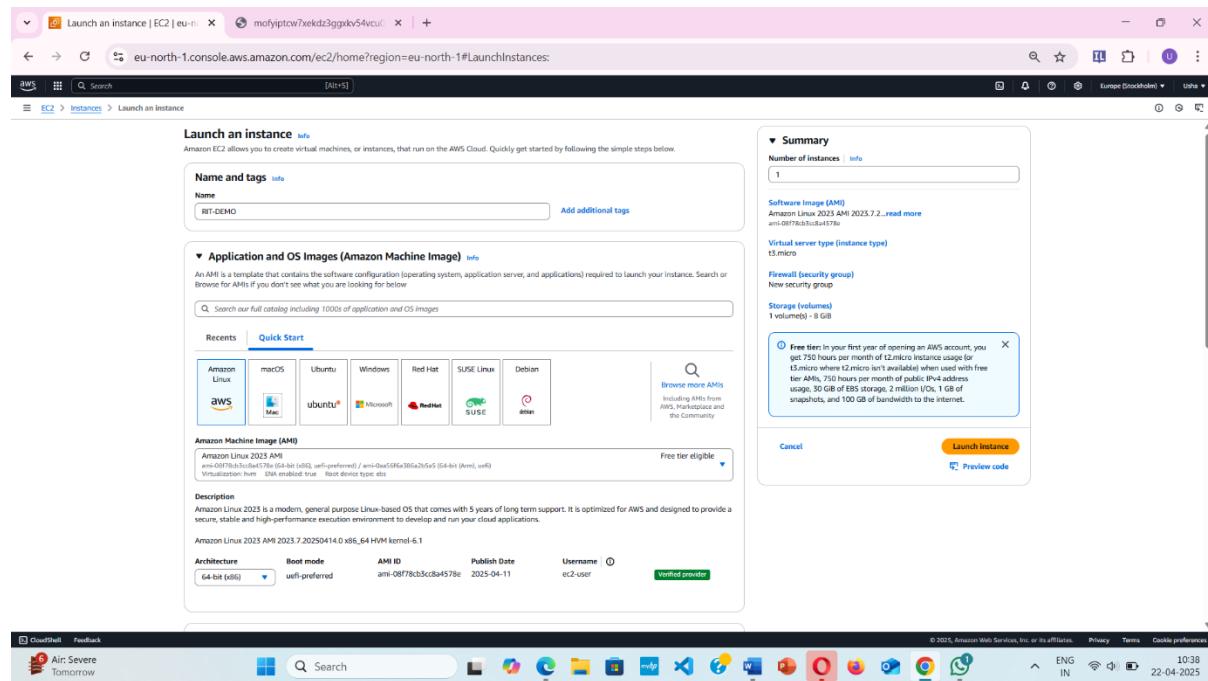
A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime,

### Procedure:

Create EC2 Instance

Linux Machine

Enable HTTPS and H TTP



Screenshot of the AWS EC2 Launch Instance wizard.

**Instance type:** t3.micro (2 vCPU, 1 GB Memory, Current generation, free tier eligible)

**Key pair (login):** Proceed without a key pair (Not recommended)

**Network settings:**

- Network: vpc-0b8efabebcc0ef593
- Subnet: No preference (Default subnet in any availability zone)
- Auto-assign public IP: Enabled
- Firewall (security groups): Create security group (selected)

**Summary:**

- Number of instances: 1
- Software Image (AMI): Amazon Linux 2023 AMI 2023.7.2... (selected)
- Virtual server type (instance type): t3.micro
- Storage (volumes): 1 volume(s) - 8 GiB

**Free tier information:** In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs. This includes 100 hours of t2.micro instance usage, 30 GiB of EBS storage, 2 million IOPS, 1 GiB of snapshots, and 100 GiB of bandwidth to the internet.

**Buttons:** Cancel, Launch instance, Preview code.

**Success message:** Successfully initiated launch of instance (i-00318b1e6db1f6b0a)

**Next Steps:**

- Create billing and free tier usage alerts
- Connect to your instance
- Connect an RDS database
- Create EBS snapshot policy
- Manage detailed monitoring
- Create Load Balancer
- Create AWS budget
- Manage CloudWatch alarms

## Go To Instance

The screenshot shows the AWS EC2 Instances page. The left sidebar navigation includes: Dashboard, EC2 Global View, Events, Instances (selected), Instances Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images (AMIS, AMI Catalog), Elastic Block Store (Volumes, Snapshots, Lifecycle Manager), Network & Security (Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces), and Load Balancing (Load Balancers). The main content area displays 'Instances (1/2) info' for two instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 IP	Elastic IP	IPv6 IP
RIT-DEMO	i-00318b1e6db1f6b0a	Running	t3.micro	Initializing	<a href="#">View alarms</a>	eu-north-1a	ec2-51-20-138-216.eu...	51.20.138.216	-	-
firstinstance	i-0946e732b05c78ae2	Stopped	t3.micro	-	<a href="#">View alarms</a>	eu-north-1b	-	-	-	-

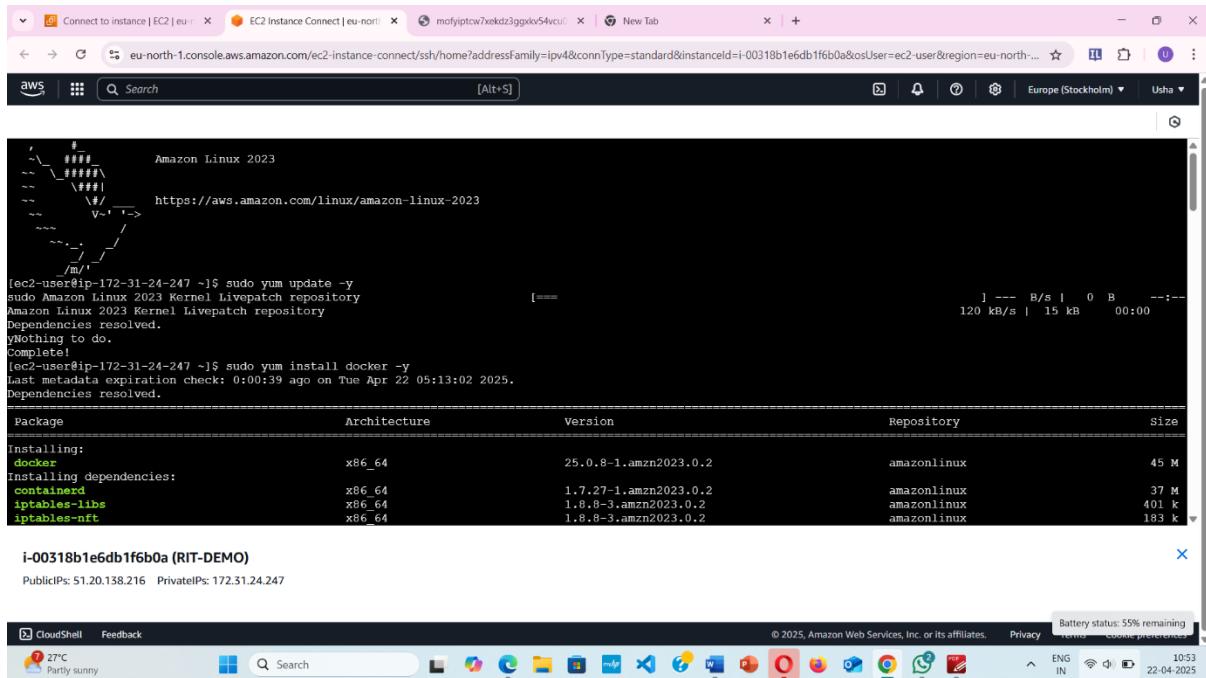
Below the table, the details for instance i-00318b1e6db1f6b0a (RIT-DEMO) are shown:

- Instance summary:**
  - Instance ID: i-00318b1e6db1f6b0a
  - IPV6 address: -
  - Hostname type: IP name: ip-172-31-24-247.eu-north-1.compute.internal
  - Answer private resource DNS name: IPv4 (A)
  - Auto-assigned IP address: 51.20.138.216 [Public IP]
  - IAM Role: -
  - HDDv2: Required
  - Operator: -
- Public IPv4 address:** 51.20.138.216 [open address]
- Private IPv4 address:** 172.31.24.247
- Public IPv4 DNS:** ec2-51-20-138-216.eu-north-1.compute.amazonaws.com [open address]
- Private IPv4 addresses:** 172.31.24.247
- VPC ID:** vpc-06bfbafbe0ef593
- Subnet ID:** subnet-0d46995e@faef5f9
- Instance ARN:** arn:aws:ec2:eu-north-1:985539760735:instance/i-00318b1e6db1f6b0a
- Elastic IP addresses:** -
- AWS Compute Optimizer finding:** No recommendations available for this instance.
- Auto Scaling Group name:** -
- Managed:** false

Press connect

The screenshot shows the 'Connect to instance' page for instance i-00318b1e6db1f6b0a (RIT-DEMO). The top navigation bar includes: EC2 > Instances > i-00318b1e6db1f6b0a > Connect to instance. The main content area has tabs: EC2 Instance Connect, Session Manager, SSH client, and EC2 serial console. The EC2 Instance Connect tab is selected. It shows the instance ID: i-00318b1e6db1f6b0a (RIT-DEMO). The Connection Type section has two options: 'Connect using EC2 Instance Connect' (selected) and 'Connect using EC2 Instance Connect Endpoint'. Below this, there are fields for 'Public IPv4 address' (51.20.138.216) and 'IPv6 address' (-). The 'Username' field contains 'ec2-user'. A note at the bottom states: 'Note: In most cases, the default username, ec2-user, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.' At the bottom right are 'Cancel' and 'Connect' buttons.

## It will take you to Linux Machine



## EXECUTE BELOW COMMANDS IN LINUX ENGINE

sudo yum update -y

sudo yum install docker -y

sudo service docker start

sudo service docker status

(ctrl Z)

sudo su

docker version

docker pull nginx

docker images

docker run -d -p 80:80 nginx

docker ps

after this go to ec2 instance → connect → copy public ip

The screenshot shows the AWS EC2 Instance Connect interface. At the top, it displays the instance ID: **i-00318b1e6db1f6b0a (RIT-DEMO)**. Below this, under the **Connection Type** section, the **Public IPv4 address** is selected, showing the IP **51.20.138.216**. The **Username** field contains **ec2-user**. A note at the bottom states: **Note: In most cases, the default username, ec2-user, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.** At the bottom right are **Cancel** and **Connect** buttons.

Paste it in browser

The screenshot shows a web browser displaying the **Welcome to nginx!** page. The URL in the address bar is **51.20.138.216**. The page content includes: **If you see this page, the nginx web server is successfully installed and working. Further configuration is required.**, **For online documentation and support please refer to [nginx.org](http://nginx.org).**, and **Commercial support is available at [nginx.com](http://nginx.com).**. Below the content, it says **Thank you for using nginx.**



