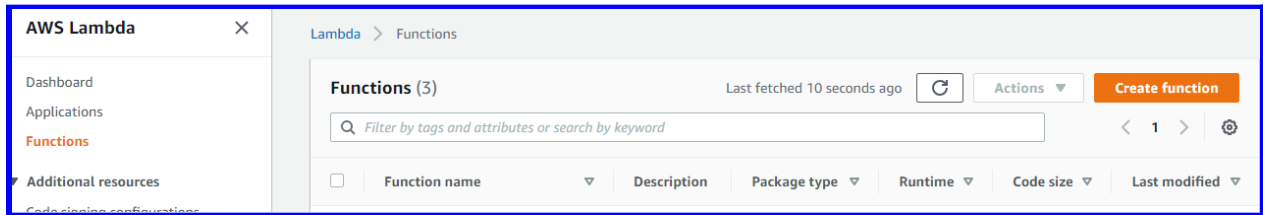


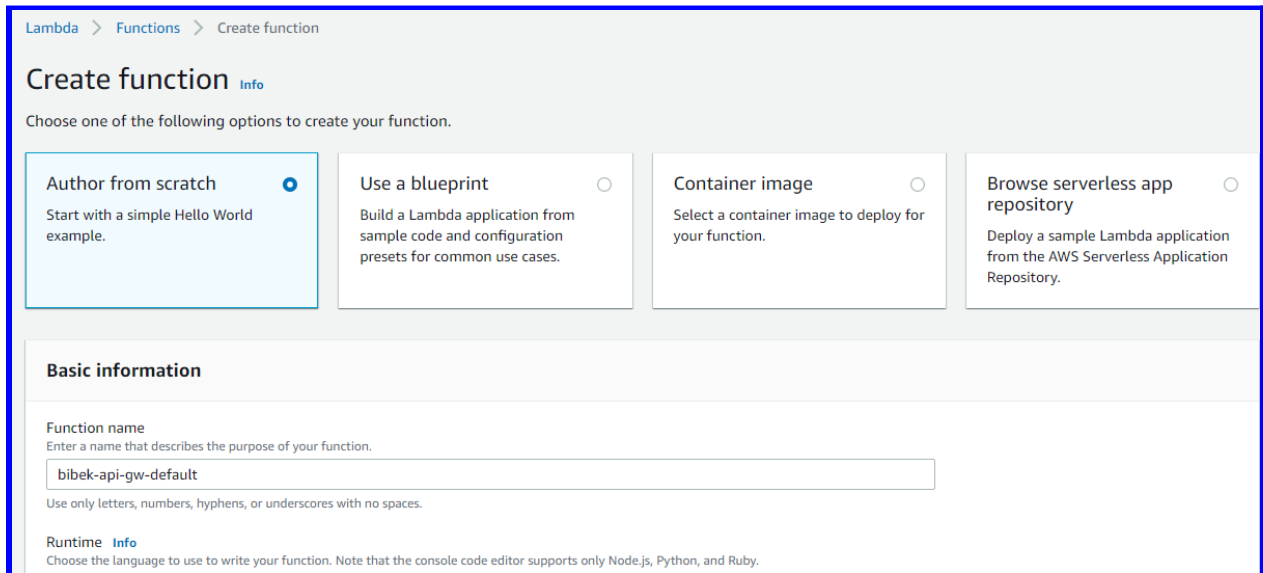
Create two Lambda Functions

First Lambda function returns 200 Response as {"Hello": "Default"}

Home Page of Lambda Function



Creating First Lambda Function- Default



Function name

Enter a name that describes the purpose of your function.

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime

[Info](#)

Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Python 3.9

▼

Architecture

[Info](#)

Choose the instruction set architecture you want for your function code.

☒ x86_64

☐ arm64

Permissions

[Info](#)

By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

► Change default execution role

► Advanced settings

Cancel

Create function

Code of the first Lambda function

Code source

[Info](#)

File Edit Find View Go Tools Window

Test

Deploy

Changes deployed

Go to Anything (Ctrl-P)

Environment

bibek-api-gw-default

lambda_function.py

lambda_function

Execution results

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

Second Lambda function returns 200 Response as {"Hello": "{Dynamic route name}"}

Creating Lambda function 2

Lambda > Functions > Create function

Create function [Info](#)

Choose one of the following options to create your function.

Author from scratch ☒

Start with a simple Hello World example.

Use a blueprint ☐

Build a Lambda application from sample code and configuration presets for common use cases.

Container image ☐

Select a container image to deploy for your function.

Browse serverless app repository ☐

Deploy a sample Lambda application from the AWS Serverless Application Repository.

Basic information

Function name

Enter a name that describes the purpose of your function.

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime [Info](#)

Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Runtime [Info](#)

Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Architecture [Info](#)

Choose the instruction set architecture you want for your function code.

☒ x86_64

☐ arm64

Permissions [Info](#)

By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

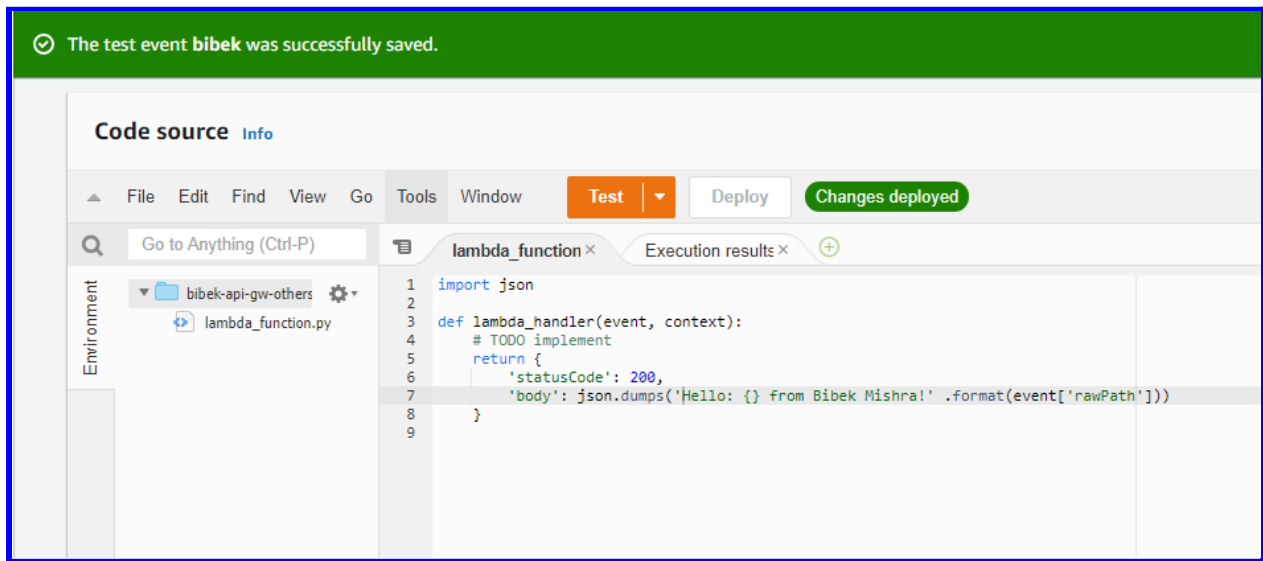
► Change default execution role

► Advanced settings

Cancel

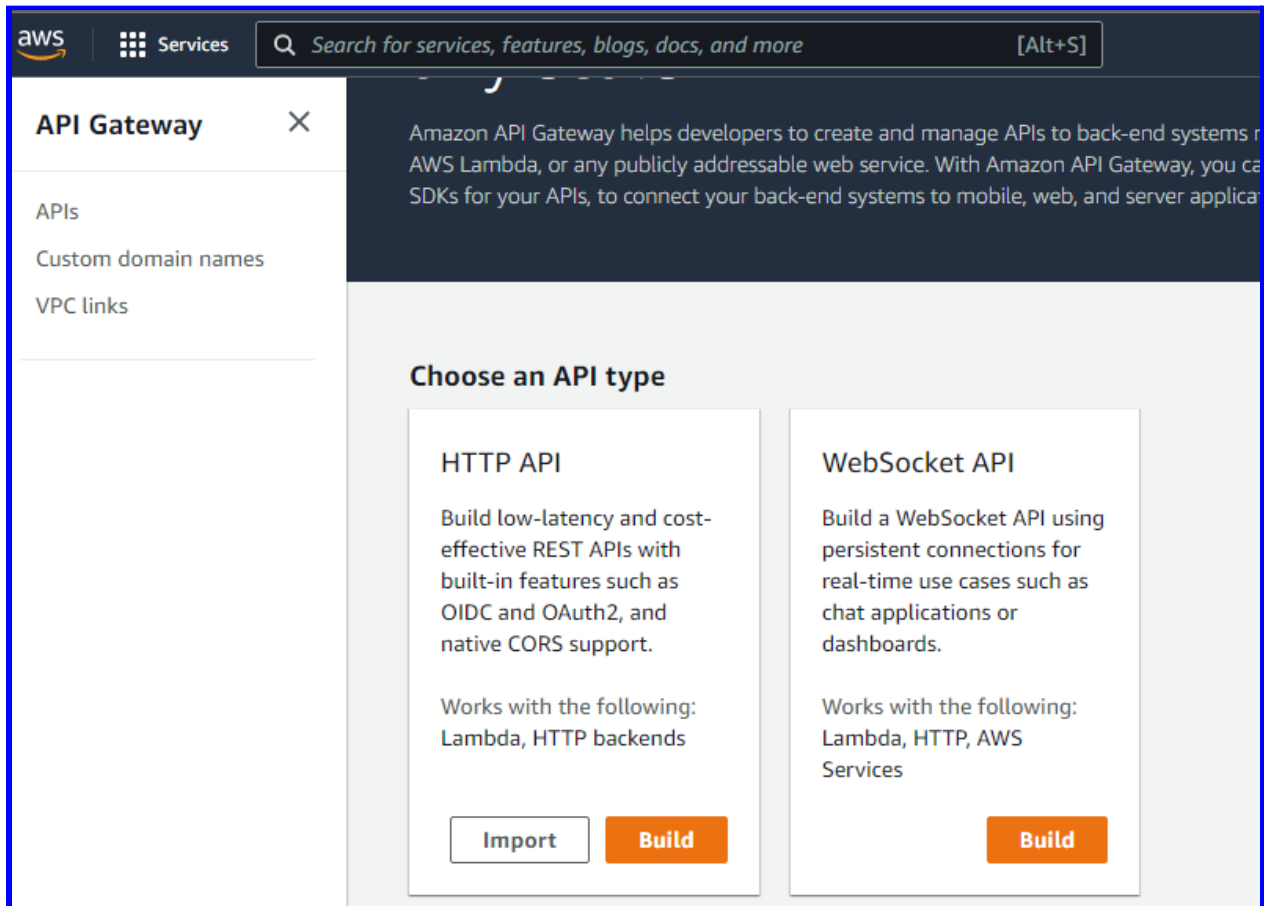
Create function

Code of 2nd Lambda Function



Configure API Gateway with that hits first lambda function on / and the second lambda function on /*

Home Page of API Gateway



The screenshot shows the AWS API Gateway console. At the top, there's a navigation bar with the AWS logo, a 'Services' menu, and a search bar. A left-hand sidebar is open, showing 'API Gateway' with a close button, and a list of links: 'APIs', 'Custom domain names', and 'VPC links'. The main content area has a dark header with a brief description of API Gateway. Below this, a section titled 'Choose an API type' presents two options: 'HTTP API' and 'WebSocket API'. Each option includes a description of its capabilities and the backends it supports. The 'HTTP API' card has 'Import' and 'Build' buttons, while the 'WebSocket API' card has a 'Build' button.

aws Services Search for services, features, blogs, docs, and more [Alt+S]

API Gateway ×

- APIs
- Custom domain names
- VPC links

Amazon API Gateway helps developers to create and manage APIs to back-end systems or AWS Lambda, or any publicly addressable web service. With Amazon API Gateway, you can use SDKs for your APIs, to connect your back-end systems to mobile, web, and server applications.

Choose an API type

HTTP API

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

Works with the following:
Lambda, HTTP backends

WebSocket API

Build a WebSocket API using persistent connections for real-time use cases such as chat applications or dashboards.

Works with the following:
Lambda, HTTP, AWS Services

Building HTTP API

API Gateway ×

APIs
Custom domain names
VPC links

Step 1
Create an API

Step 2
Configure routes

Step 3
Define stages

Step 4
Review and create

Create an API

Create and configure integrations

Specify the backend services that your API will communicate with. These are called integrations. For a Lambda integration, API Gateway invokes the Lambda function and responds with the response from the function. For HTTP integration, API Gateway sends the request to the URL that you specify and returns the response from the URL.

Integrations [Info](#)

Lambda Remove

AWS Region: us-east-2 Lambda function: bibek-api-gw-default Version: 2.0 [Learn more.](#)

Added Two Lambda Function Integration

Lambda Remove

AWS Region: us-east-2 Lambda function: bibek-api-gw-default Version: 2.0 [Learn more.](#)

Lambda Remove

AWS Region: us-east-2 Lambda function: bibek-api-gw-others Version: 2.0 [Learn more.](#)

Add integration

API name

An HTTP API must have a name. This name is cosmetic and does not have to be unique; you will use the API's ID (generated later) to programmatically refer to this API.

bibek-api

Cancel Review and Create Next

Configuring routes.

“/” for default gateway, *to serve request for “/”*

“/{proxy+}” for second Lambda function, *to serve request for all others than “/”*

Configure routes

Configure routes [Info](#)

API Gateway uses routes to expose integrations to consumers of your API. Routes for HTTP APIs consist of two parts: an HTTP method and a resource path (e.g., GET /pets). You can define specific HTTP methods for your integration (GET, POST, PUT, PATCH, HEAD, OPTIONS, and DELETE) or use the ANY method to match all methods that you haven't defined on a given resource.

Method	Resource path		Integration target	
ANY ▼	/	→	bibek-api-gw-default ▼	<button>Remove</button>
ANY ▼	/{proxy+}	→	bibek-api-gw-others ▼	<button>Remove</button>
<button>Add route</button>				

CancelPreviousNext

Define stages

Configure stages [Info](#)

Stages are independently configurable environments that your API can be deployed to. You must deploy to a stage for API configuration changes to take effect, unless that stage is configured to autodeploy. By default, all HTTP APIs created through the console have a default stage named `$default`. All changes that you make to your API are autodeployed to that stage. You can add stages that represent environments such as development or production.

Stage name

`$default`

Auto-deploy



Remove

Add stage

Cancel

Previous

Next

Review and create

API name and integrations

Edit

API name

bibek-api

Integrations

bibek-api-gw-default (Lambda)

bibek-api-gw-others (Lambda)

Routes

Edit

Routes

ANY / → bibek-api-gw-default (Lambda)

ANY /{proxy+} → bibek-api-gw-others (Lambda)

Routes

Edit

Routes

ANY / → bibek-api-gw-default (Lambda)

ANY /{proxy+} → bibek-api-gw-others (Lambda)

Stages

Edit

Stages

\$default (Auto-deploy: enabled)

Cancel

Previous

Create

After successfully creating API gateway with two Lambda function Integrations

✔ Successfully created API bibek-api (ah5clpryld)

✔ Successfully created API bibek-api (agzkfku8sd)

bibek-api

Edit

API details

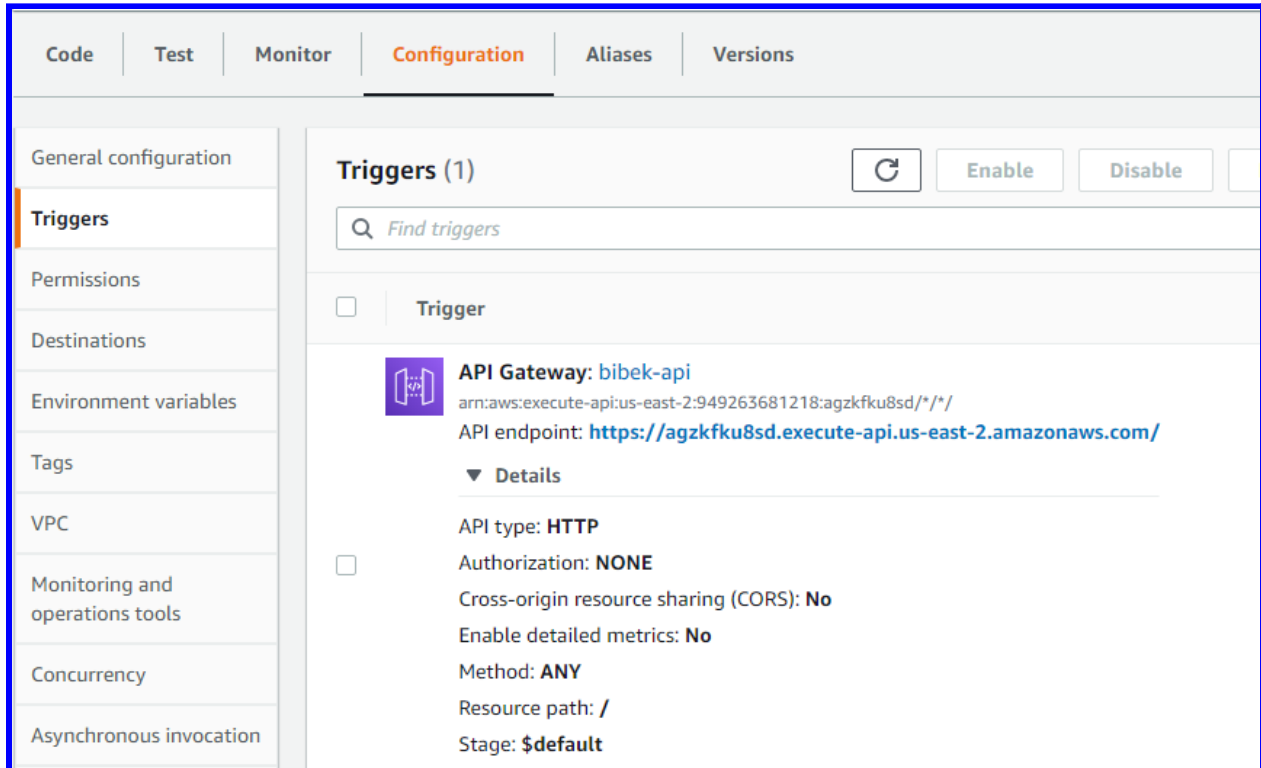
API ID	Protocol	Created
agzkfku8sd	HTTP	2021-12-15
Description	Default endpoint	
No Description	Enabled	

Stages for bibek-api

Stage name	Invoke URL	Attached deployment	Auto deploy	Last updated
\$default	https://agzkfku8sd.execute-api.us-east-2.amazonaws.com	q6gbq6	enabled	2021-12-15

We can see the added triggers for API GW in Respective Lambda Functions

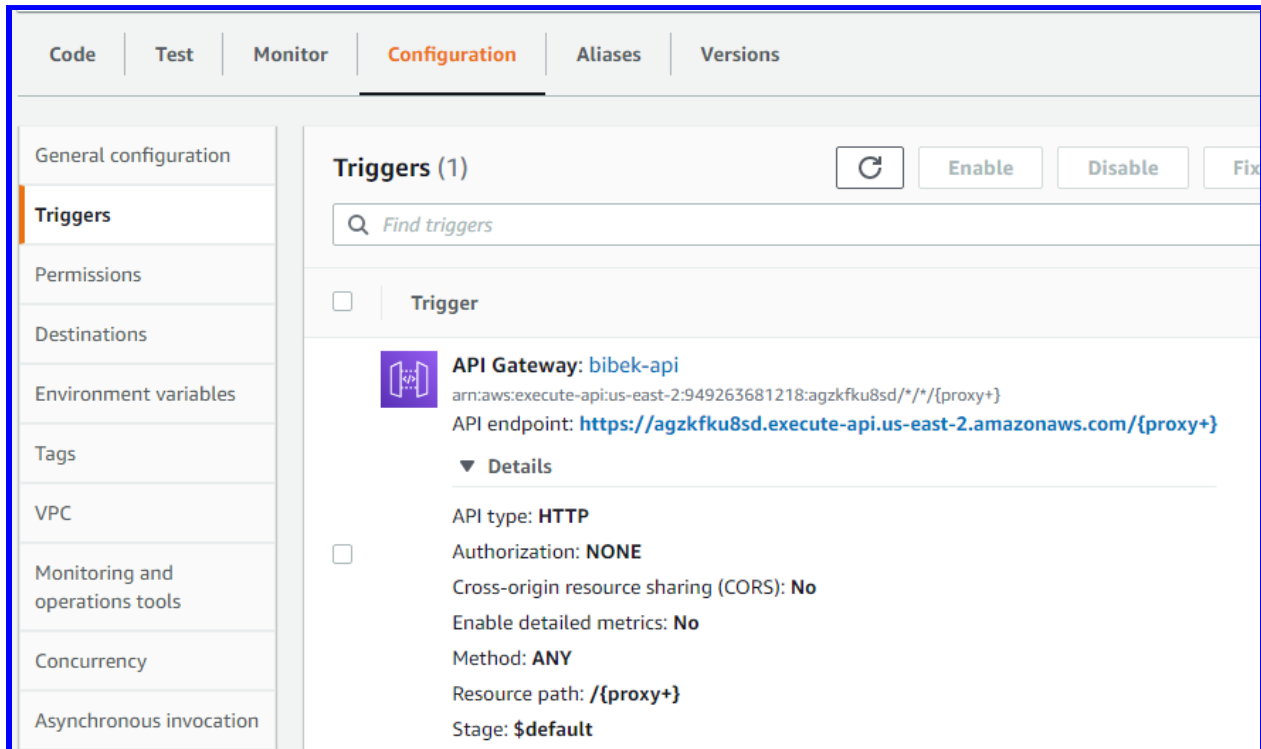
Lambda Function 1 - Triggers



The screenshot displays the AWS Lambda console's Configuration tab for a specific function. The left-hand navigation pane includes options like General configuration, Triggers, Permissions, Destinations, Environment variables, Tags, VPC, Monitoring and operations tools, Concurrency, and Asynchronous invocation. The 'Triggers' section is active, showing a list of triggers. One trigger is listed: 'API Gateway: bibek-api'. The details for this trigger are as follows:

- arn:aws:execute-api:us-east-2:949263681218:agzkfku8sd/*/*
- API endpoint: <https://agzkfku8sd.execute-api.us-east-2.amazonaws.com/>
- Details:
 - API type: HTTP
 - Authorization: NONE
 - Cross-origin resource sharing (CORS): No
 - Enable detailed metrics: No
 - Method: ANY
 - Resource path: /
 - Stage: \$default

Lambda Function 2 - Triggers



The screenshot displays the AWS Lambda console's Configuration tab for a second function. The left-hand navigation pane is identical to the first screenshot, with 'Triggers' selected. The main area shows a single trigger for 'API Gateway: bibek-api'. The details for this trigger are as follows:

- arn:aws:execute-api:us-east-2:949263681218:agzkfku8sd/*/*/{proxy+}
- API endpoint: <https://agzkfku8sd.execute-api.us-east-2.amazonaws.com/{proxy+}>
- Details:
 - API type: HTTP
 - Authorization: NONE
 - Cross-origin resource sharing (CORS): No
 - Enable detailed metrics: No
 - Method: ANY
 - Resource path: /{proxy+}
 - Stage: \$default

While Browsing the domain of API GateWay

