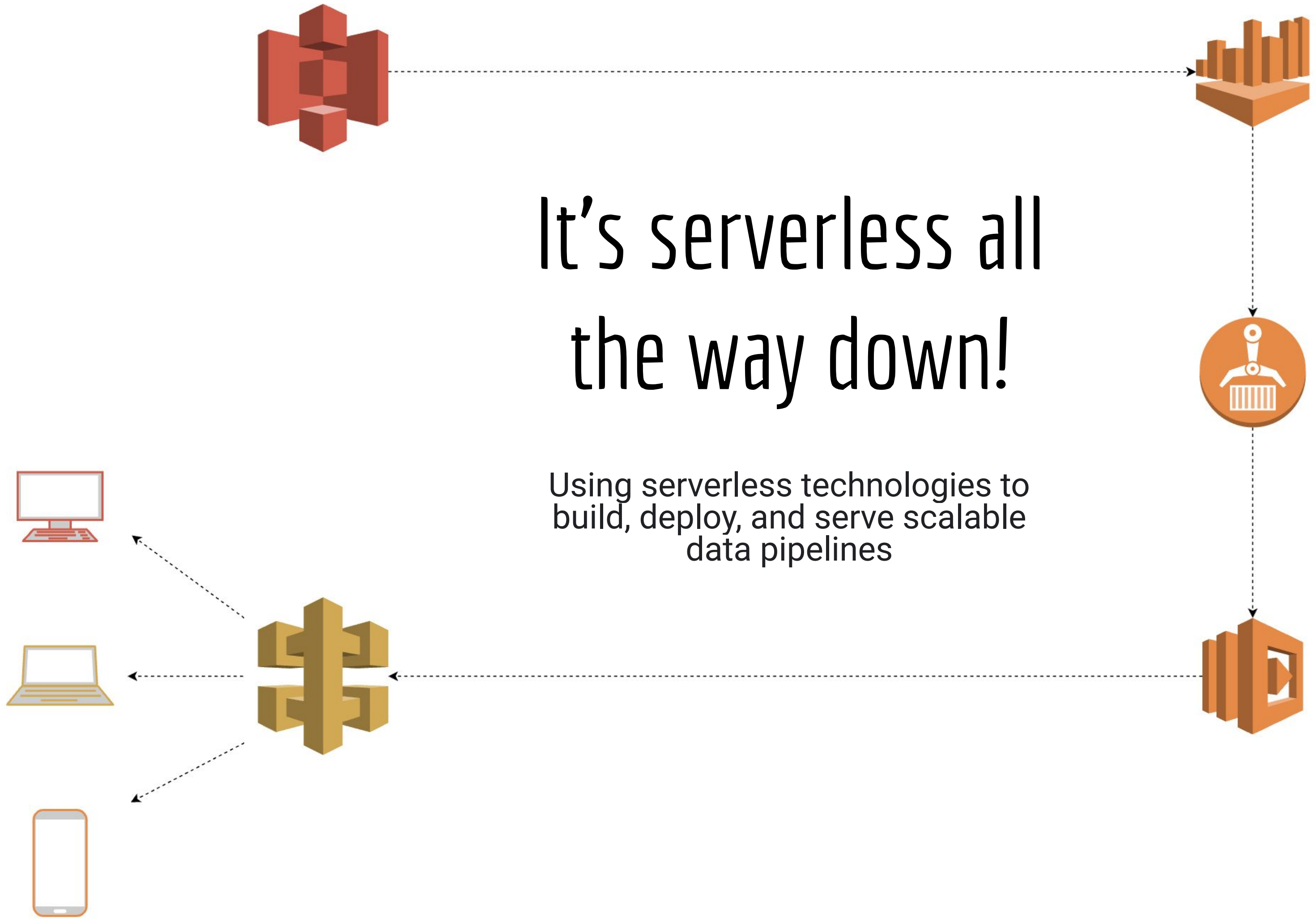
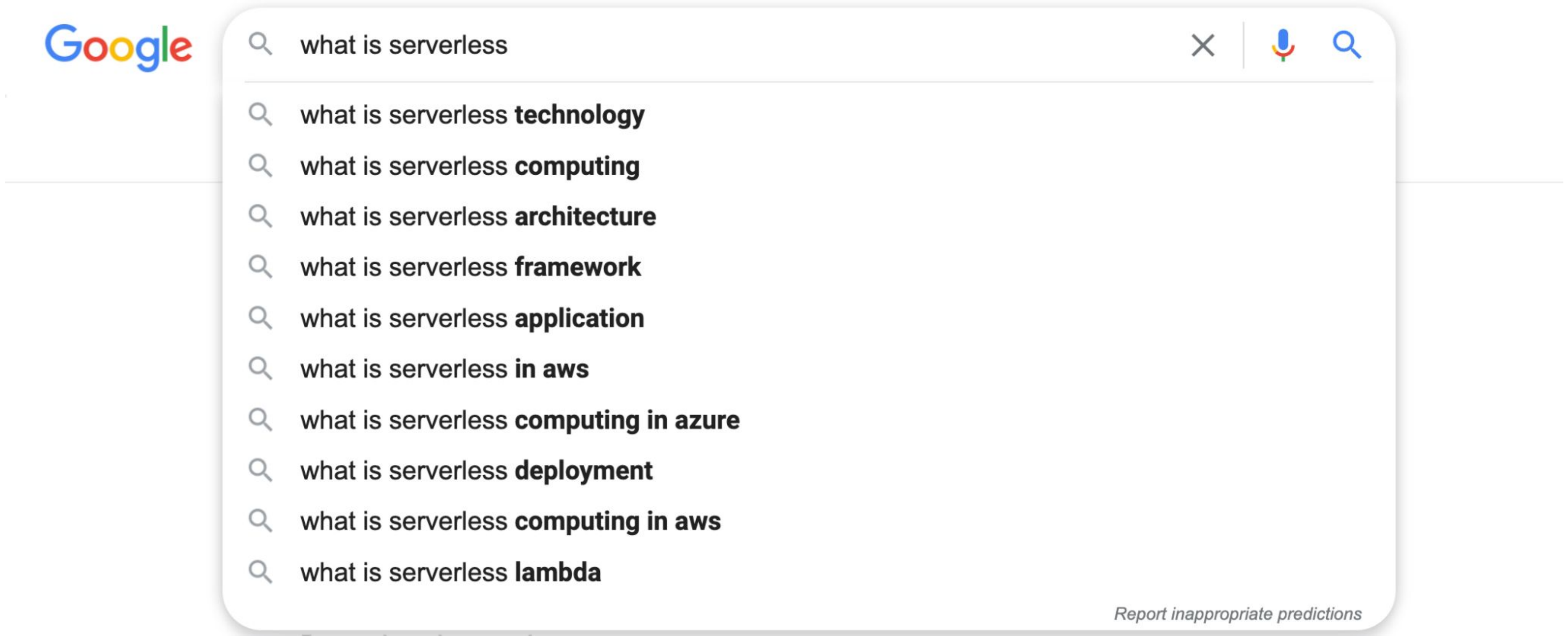
It's serverless all the way down!

Using serverless technologies to build, deploy, and serve scalable data pipelines



Who has used serverless technologies?



what is serverless

what is serverless **technology**

what is serverless **computing**

what is serverless **architecture**

what is serverless **framework**

what is serverless **application**

what is serverless **in aws**

what is serverless **computing in azure**

what is serverless **deployment**

what is serverless **computing in aws**

what is serverless **lambda**

Report inappropriate predictions

Why go serverless?

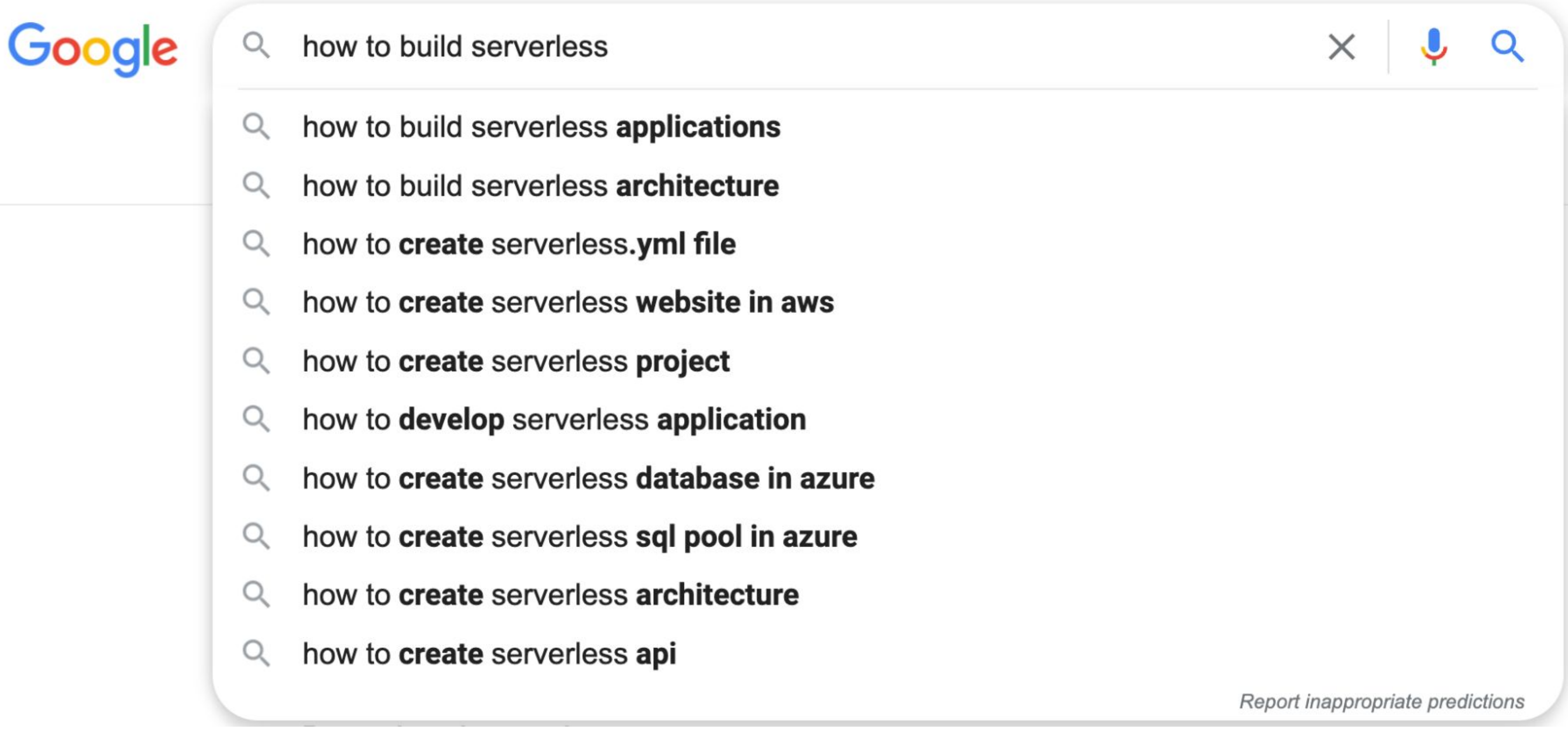
- Eliminates server maintenance
- Only pay for what you use
- Inherently and automatically scalable
- Easy to deploy

When to go serverless?

- Non-constant workload
- Volume-varying tasks
- Non-low latency requirements
- If you’ve asked “how can i meet these requirements with a service that’s always on and available at a low cost?”

When to **NOT** go serverless?

- Low latency responses
- High memory computations
- Long running tasks
- Many tasks interacting with each other and require statefulness



how to build serverless

how to build serverless **applications**

how to build serverless **architecture**

how to **create** serverless**.yml file**

how to **create** serverless **website in aws**

how to **create** serverless **project**

how to **develop** serverless **application**

how to **create** serverless **database in azure**

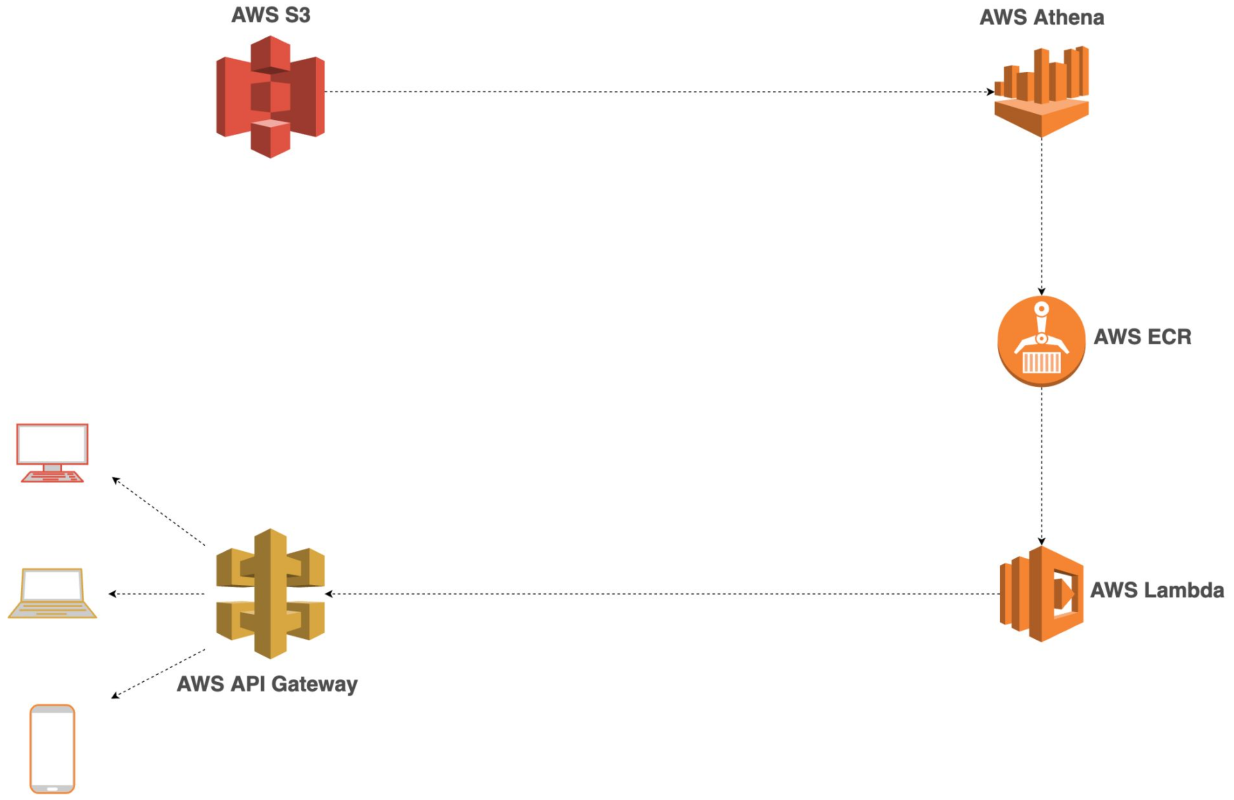
how to **create** serverless **sql pool in azure**

how to **create** serverless **architecture**

how to **create** serverless **api**

Report inappropriate predictions

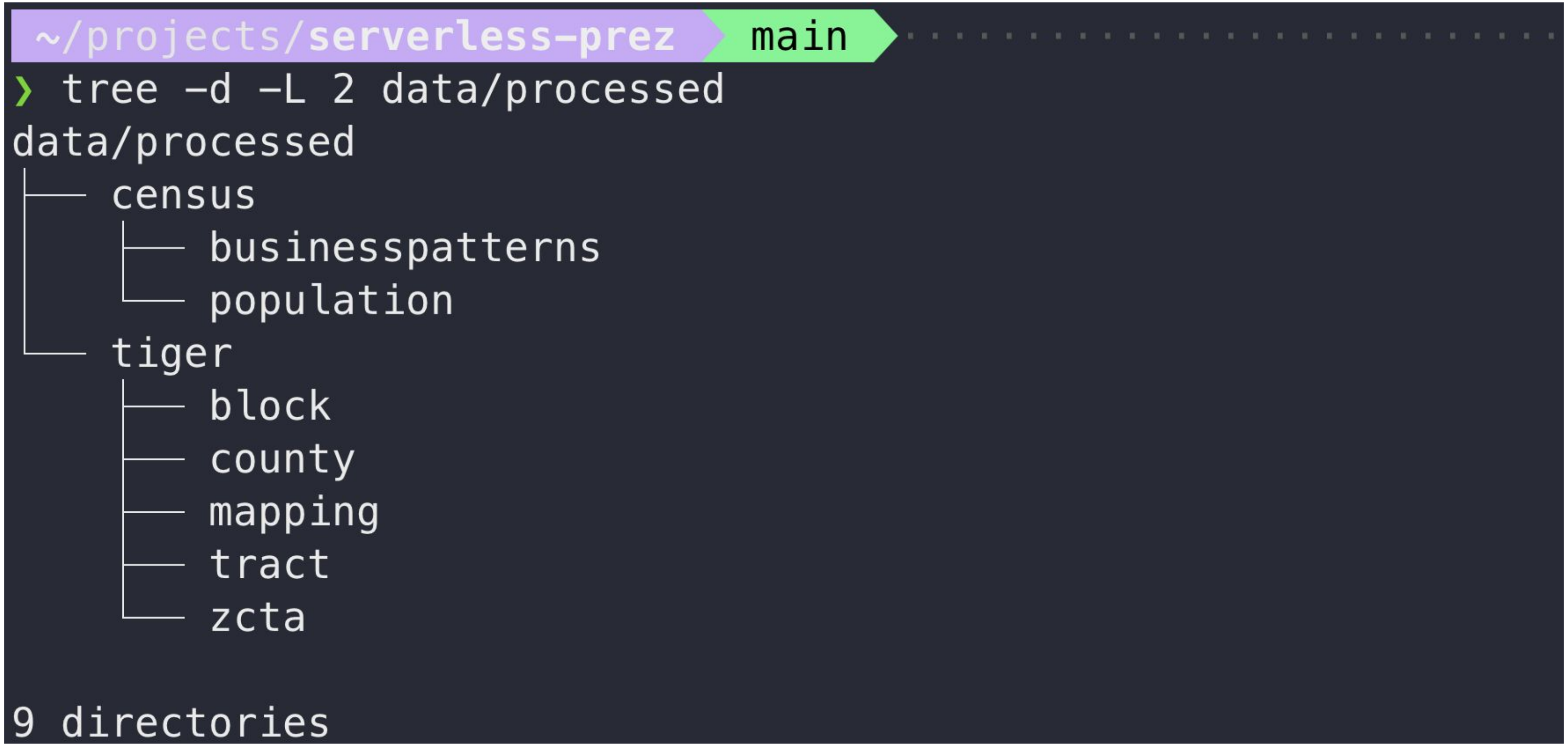
Before giving an example, let's define some things



For the sake of time...

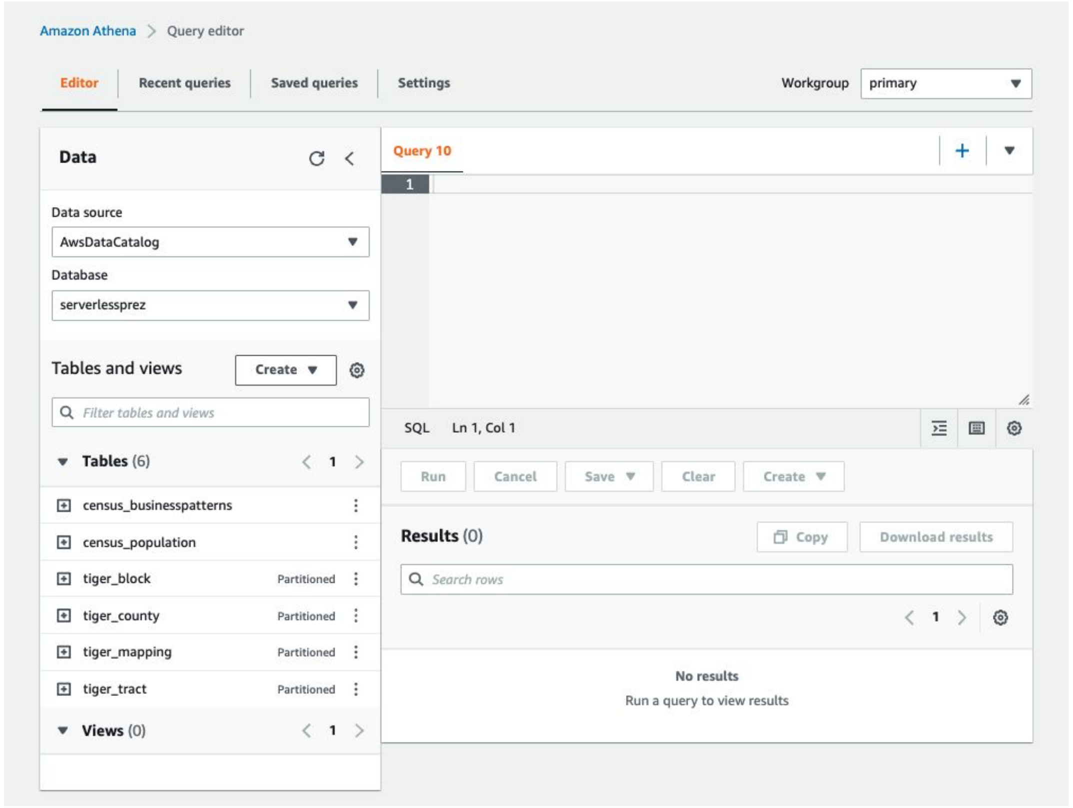
- I’ve already populated an **S3** bucket with data that we’ll retrieve here
- I’ve already created our **Athena** query engine that we’ll use to query our data in S3

Here's what our data in S3 looks like



```
~/projects/serverless-prez  main
❯ tree -d -L 2 data/processed
data/processed
├── census
│   ├── businesspatterns
│   └── population
└── tiger
    ├── block
    ├── county
    ├── mapping
    ├── tract
    └── zcta

9 directories
```

Here's what our Athena environment looks like



Workgroup: primary

Data source: AwsDataCatalog

Database: serverlessprez

Tables and views

Tables (6)

census_businesspatterns	
census_population	
tiger_block	Partitioned
tiger_county	Partitioned
tiger_mapping	Partitioned
tiger_tract	Partitioned

Views (0)

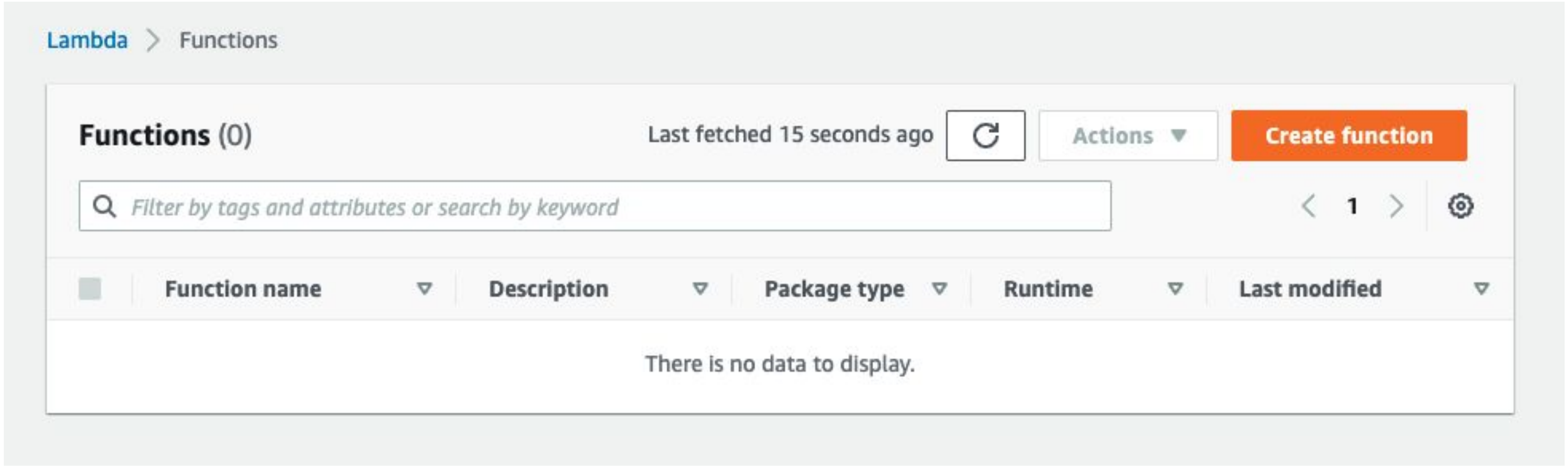
Results (0)

No results

Run a query to view results

Now let’s build the rest of it!

We'll start with creating our Lambda function

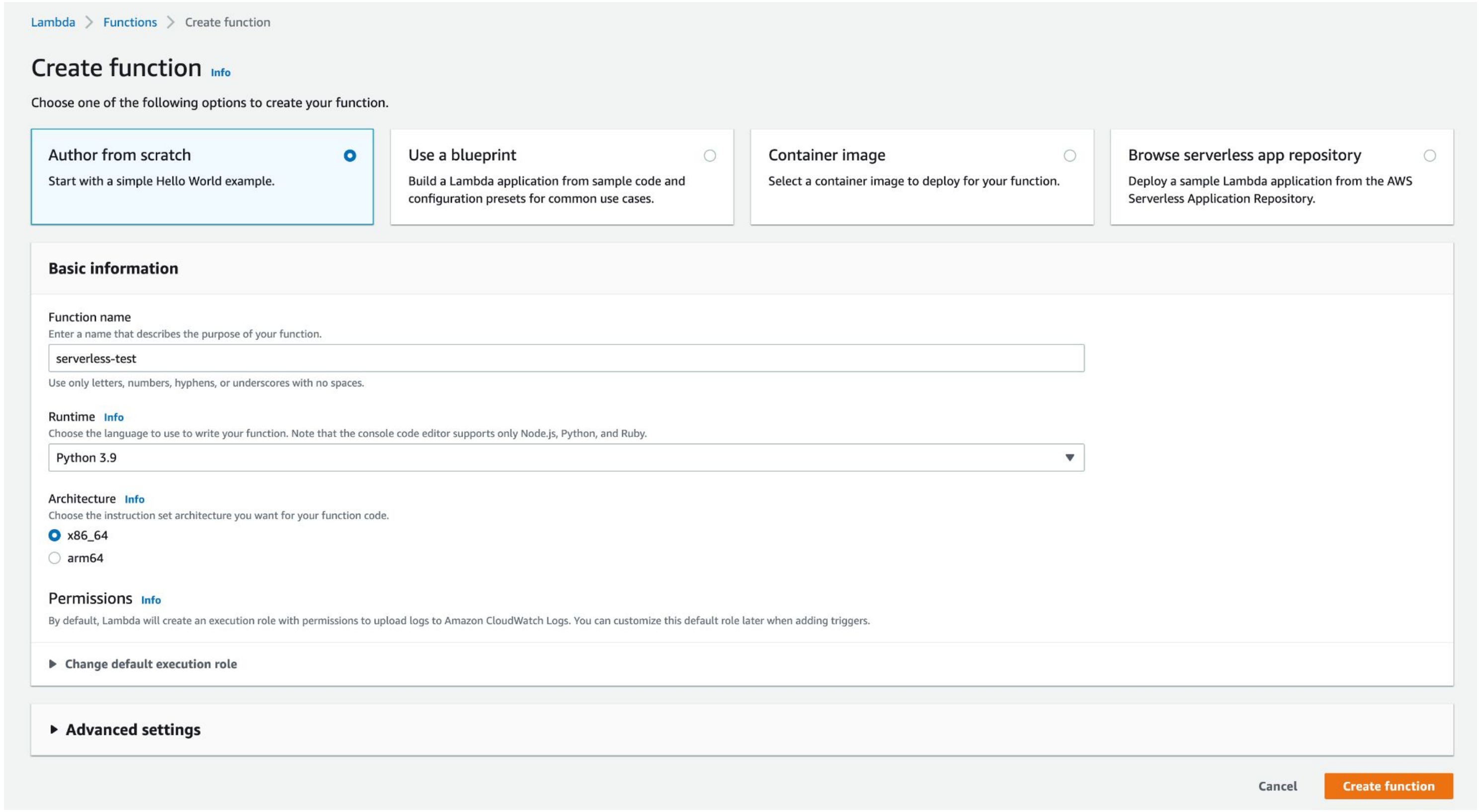

Lambda > Functions

Functions (0)

Last fetched 15 seconds ago

	Function name	Description	Package type	Runtime	Last modified
There is no data to display.					

Configure and create



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.
serverless-test
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

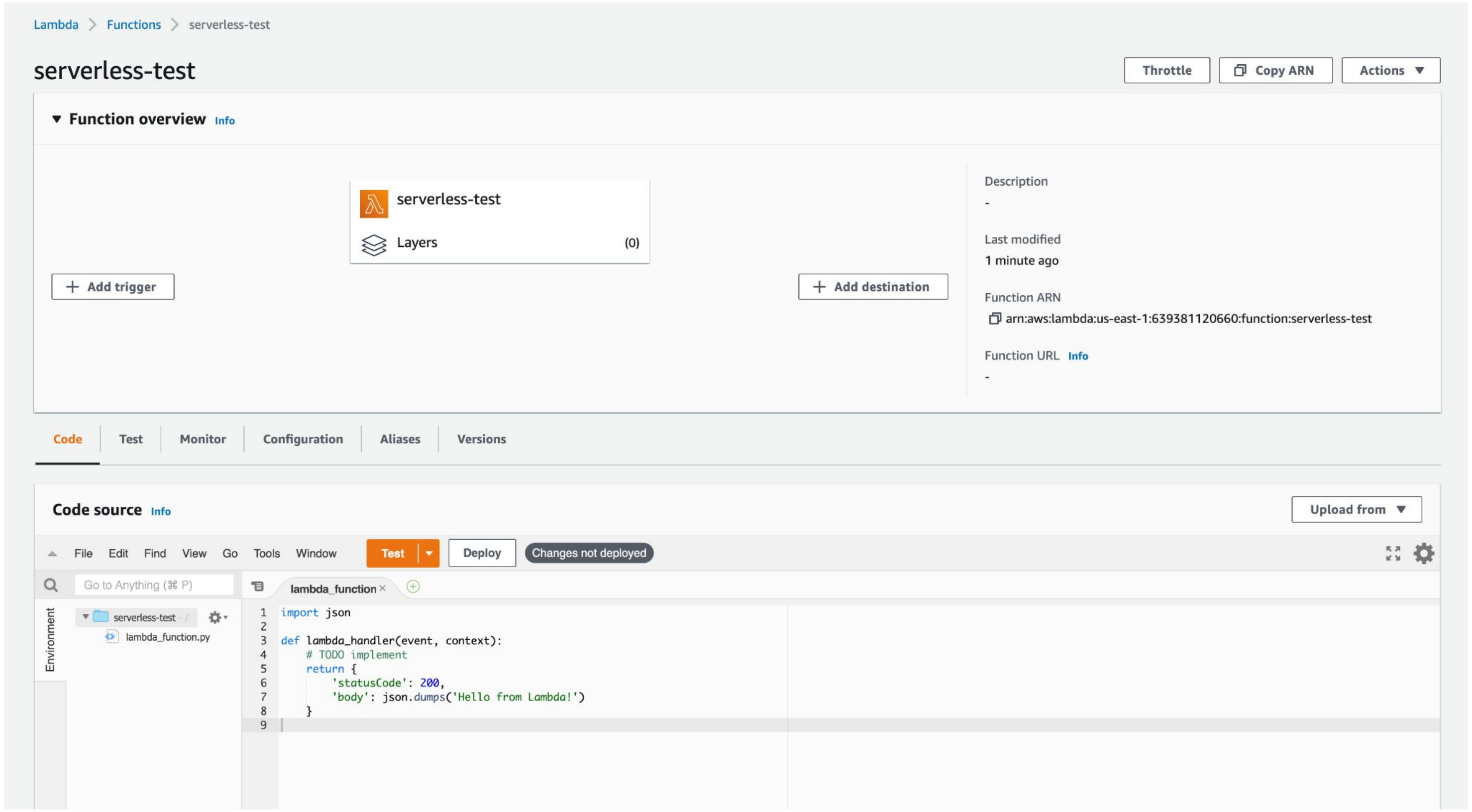
Architecture Info
Choose the instruction set architecture you want for your function code.
[✓ x86_64]
arm64

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**

Edit function code


Lambda > Functions > serverless-test

serverless-test

Function overview

Description
-

Last modified
1 minute ago

Function ARN
arn:aws:lambda:us-east-1:639381120660:function:serverless-test

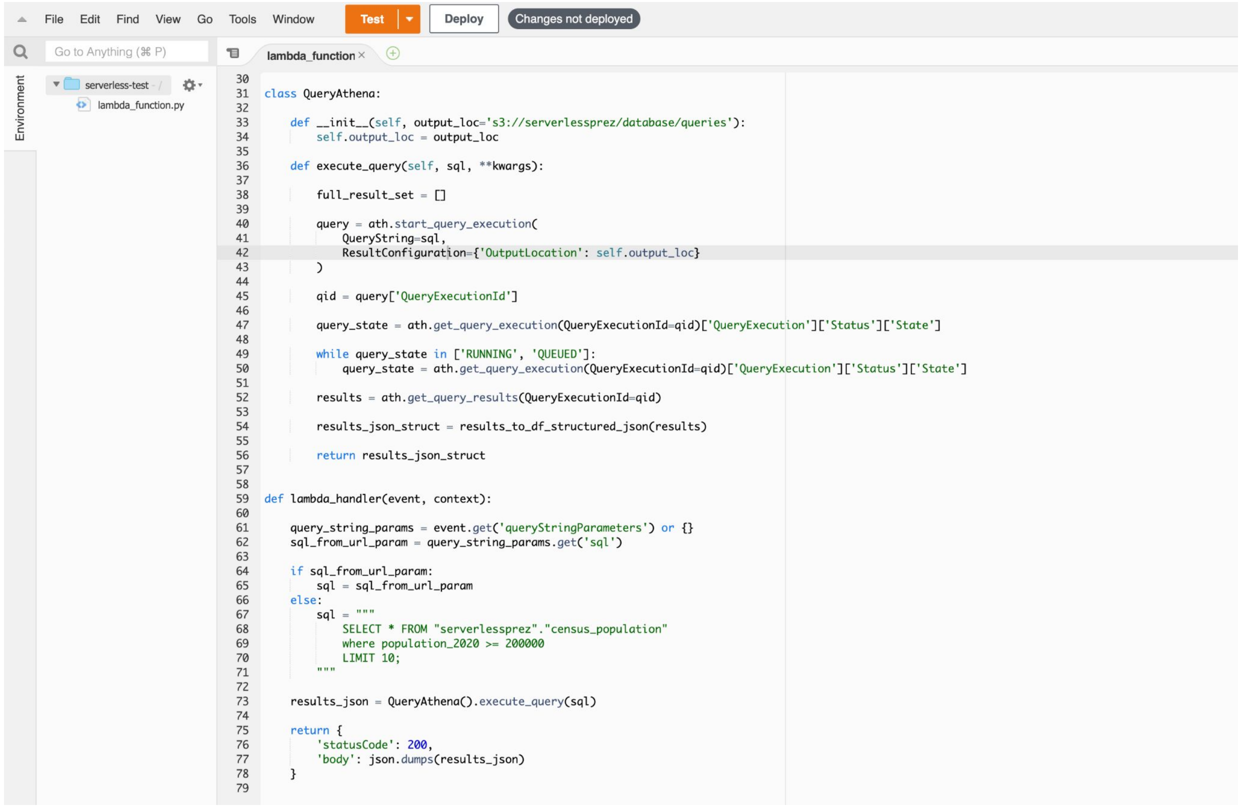
Function URL
-

Code source

Changes not deployed

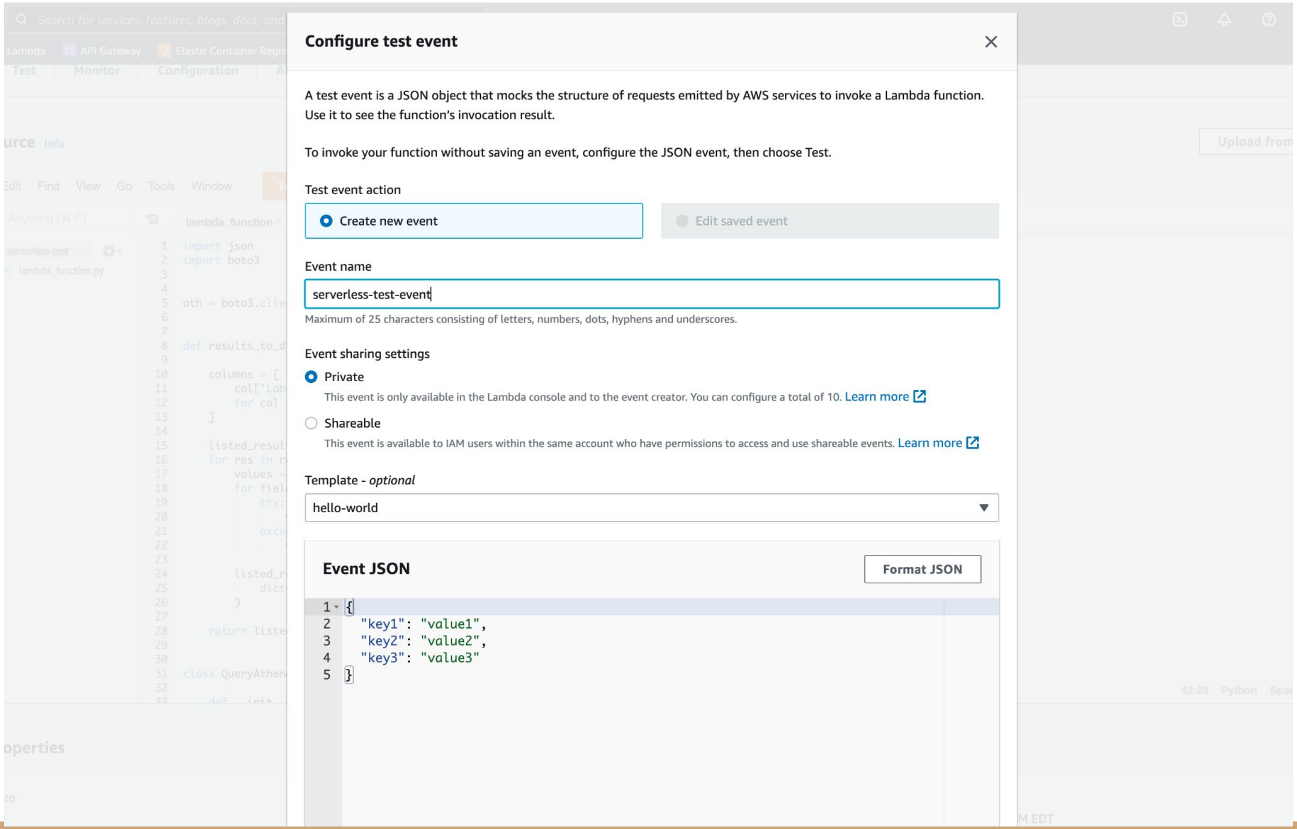
```
import json

def lambda_handler(event, context):
    # TODO implement
    return {
        'statusCode': 200,
        'body': json.dumps('Hello from Lambda!')
    }

```

I'll use code I've already written for this



```python

class QueryAthena:

    def __init__(self, output_loc='s3://serverlessprez/database/queries'):
        self.output_loc = output_loc

    def execute_query(self, sql, **kwargs):

        full_result_set = []

        query = ath.start_query_execution(
            QueryString=sql,
            ResultConfiguration={'OutputLocation': self.output_loc}
        )

        qid = query['QueryExecutionId']

        query_state = ath.get_query_execution(QueryExecutionId=qid)['QueryExecution']['Status']['State']

        while query_state in ['RUNNING', 'QUEUED']:
            query_state = ath.get_query_execution(QueryExecutionId=qid)['QueryExecution']['Status']['State']

        results = ath.get_query_results(QueryExecutionId=qid)

        results_json_struct = results_to_df_structured_json(results)

        return results_json_struct


def lambda_handler(event, context):

    query_string_params = event.get('queryStringParameters') or {}
    sql_from_url_param = query_string_params.get('sql')

    if sql_from_url_param:
        sql = sql_from_url_param
    else:
        sql = """
            SELECT * FROM "serverlessprez"."census_population"
            where population_2020 >= 200000
            LIMIT 10;
        """

    results_json = QueryAthena().execute_query(sql)

    return {
        'statusCode': 200,
        'body': json.dumps(results_json)
    }

```

Test our Lambda function before deploying



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

serverless-test-event

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

Event sharing settings

[✓ Private]
This event is only available in the Lambda console and to the event creator. You can configure a total of 10. Learn more

Shareable
This event is available to IAM users within the same account who have permissions to access and use shareable events. Learn more

Template - *optional*

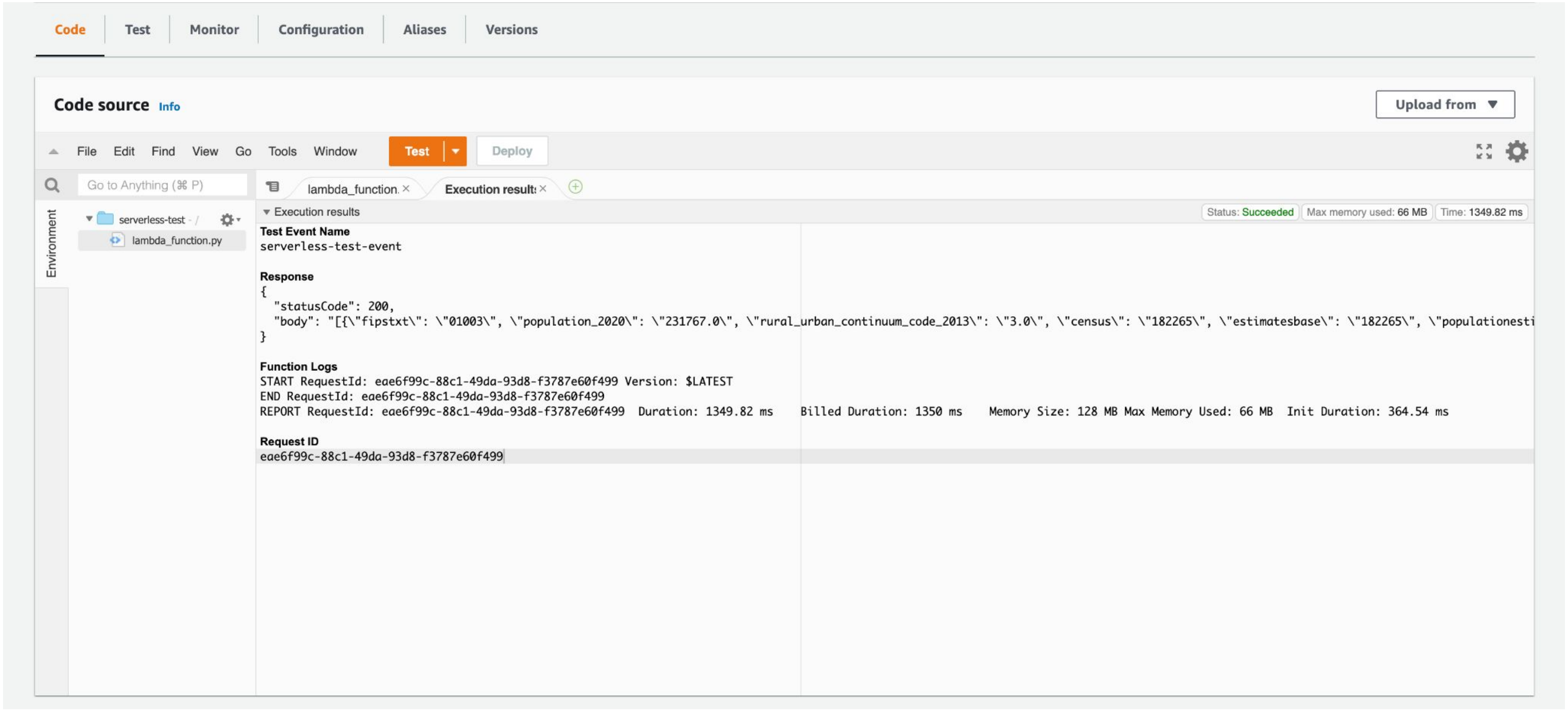
hello-world

Event JSON

Format JSON

```
{
  "key1": "value1",
  "key2": "value2",
  "key3": "value3"
}
```

It works!



Execution results

Status: Succeeded | Max memory used: 66 MB | Time: 1349.82 ms

Test Event Name

```
serverless-test-event
```

Response

```
{
  "statusCode": 200,
  "body": "[{\"fipstxt\": \"01003\", \"population_2020\": \"231767.0\", \"rural_urban_continuum_code_2013\": \"3.0\", \"census\": \"182265\", \"estimatesbase\": \"182265\", \"populationesti
}
```

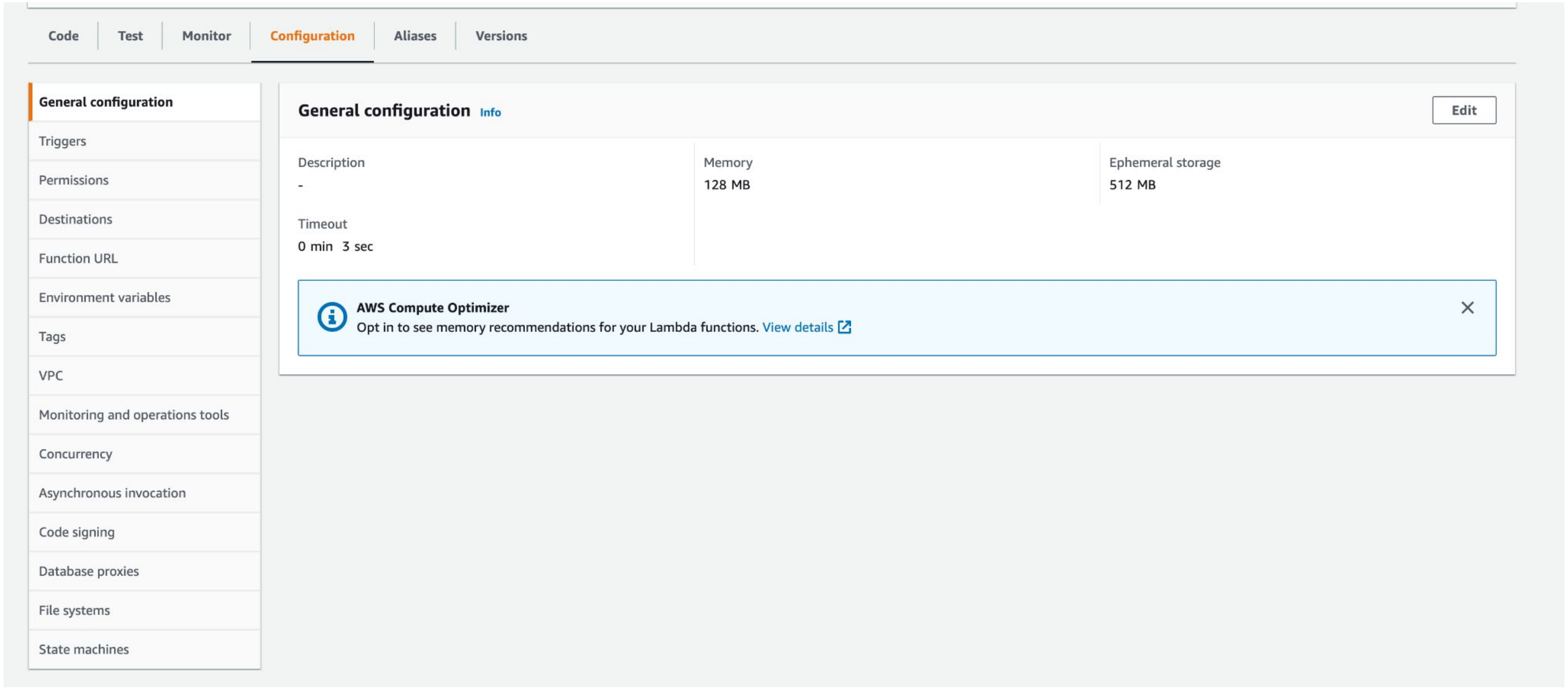
Function Logs

```
START RequestId: eae6f99c-88c1-49da-93d8-f3787e60f499 Version: $LATEST
END RequestId: eae6f99c-88c1-49da-93d8-f3787e60f499
REPORT RequestId: eae6f99c-88c1-49da-93d8-f3787e60f499  Duration: 1349.82 ms    Billed Duration: 1350 ms    Memory Size: 128 MB Max Memory Used: 66 MB  Init Duration: 364.54 ms
```

Request ID

```
eae6f99c-88c1-49da-93d8-f3787e60f499
```

Now let's extend the timeout and memory



General configuration

Description
-

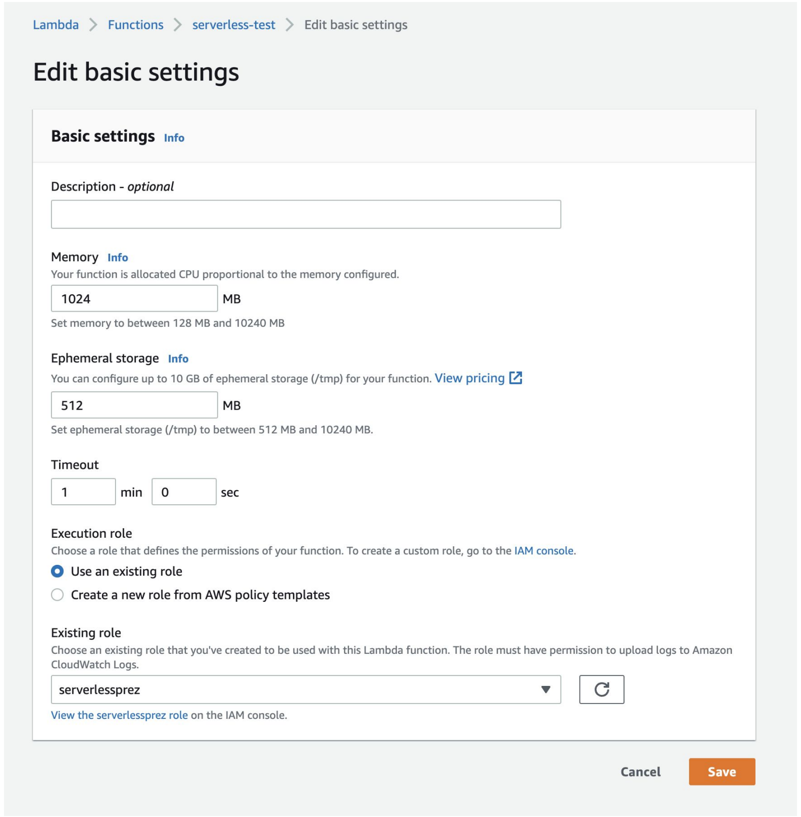
Memory
128 MB

Ephemeral storage
512 MB

Timeout
0 min 3 sec

AWS Compute Optimizer
Opt in to see memory recommendations for your Lambda functions. View details

Now let's extend the timeout and memory



Edit basic settings

Basic settings Info

Description - *optional*

Memory Info
Your function is allocated CPU proportional to the memory configured.
1024 MB
Set memory to between 128 MB and 10240 MB

Ephemeral storage Info
You can configure up to 10 GB of ephemeral storage (/tmp) for your function. View pricing
512 MB
Set ephemeral storage (/tmp) to between 512 MB and 10240 MB.

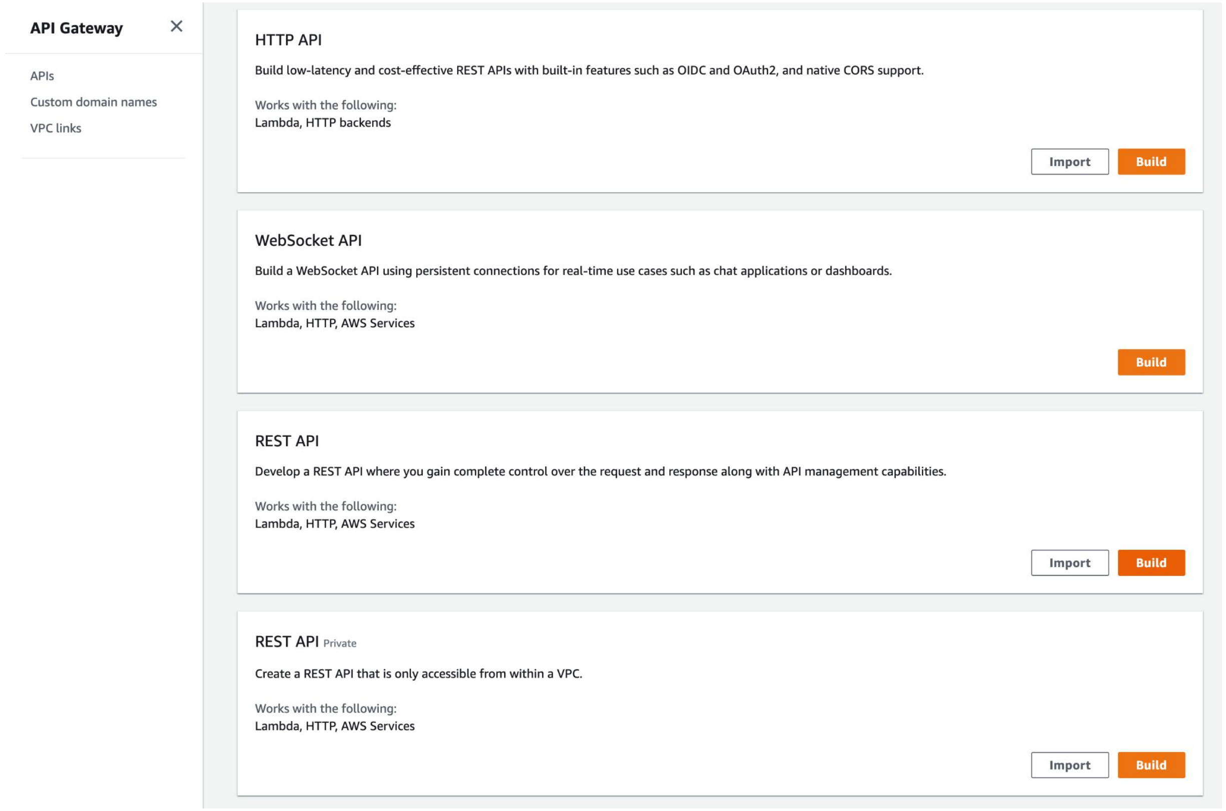
Timeout
1 min 0 sec

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.
[✓ Use an existing role]
Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
serverlessprez
View the serverlessprez role on the IAM console.

Now let's make our data accessible over HTTPS

Create an API on AWS API Gateway



HTTP API

Build low-latency and cost-effective REST APIs with built-in features such as OIDC 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

REST API

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

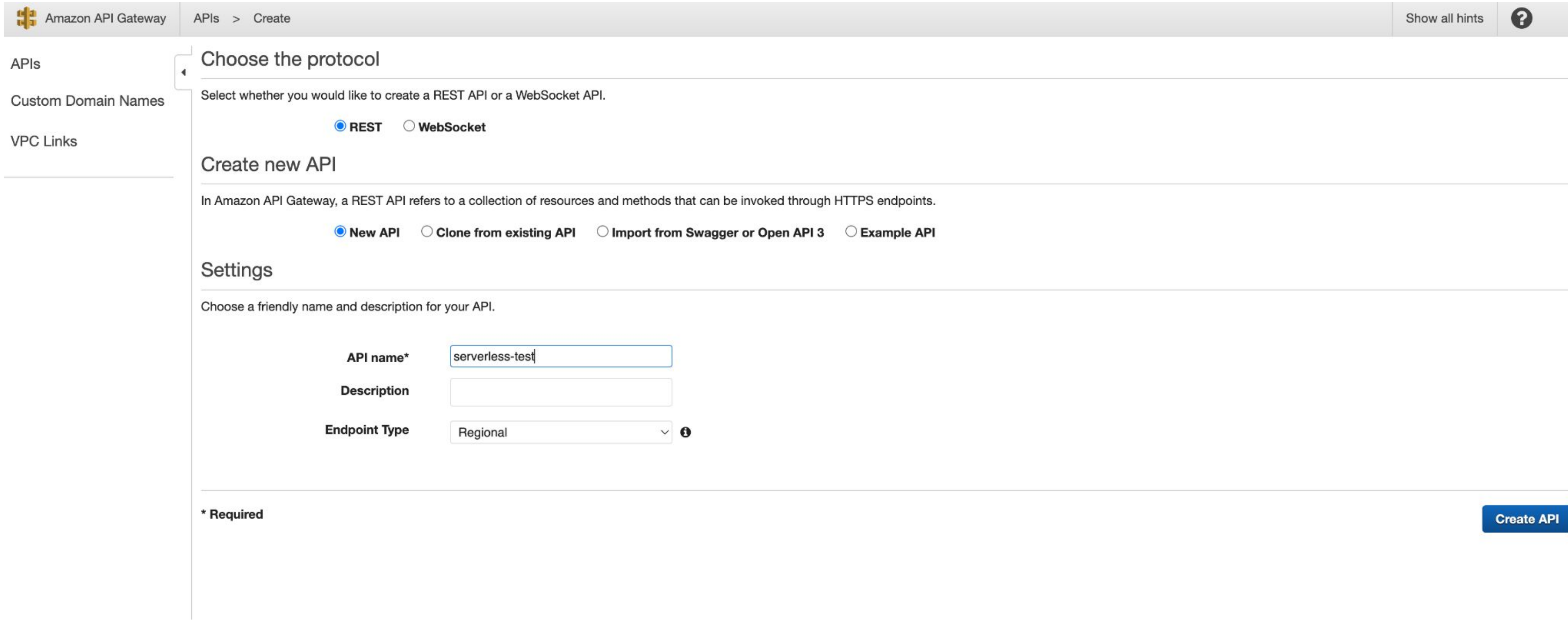
Works with the following:
Lambda, HTTP, AWS Services

REST API Private

Create a REST API that is only accessible from within a VPC.

Works with the following:
Lambda, HTTP, AWS Services

Name the API



Choose the protocol

Select whether you would like to create a REST API or a WebSocket API.

[✓ REST] WebSocket

Create new API

In Amazon API Gateway, a REST API refers to a collection of resources and methods that can be invoked through HTTPS endpoints.

[✓ New API] Clone from existing API Import from Swagger or Open API 3 Example API

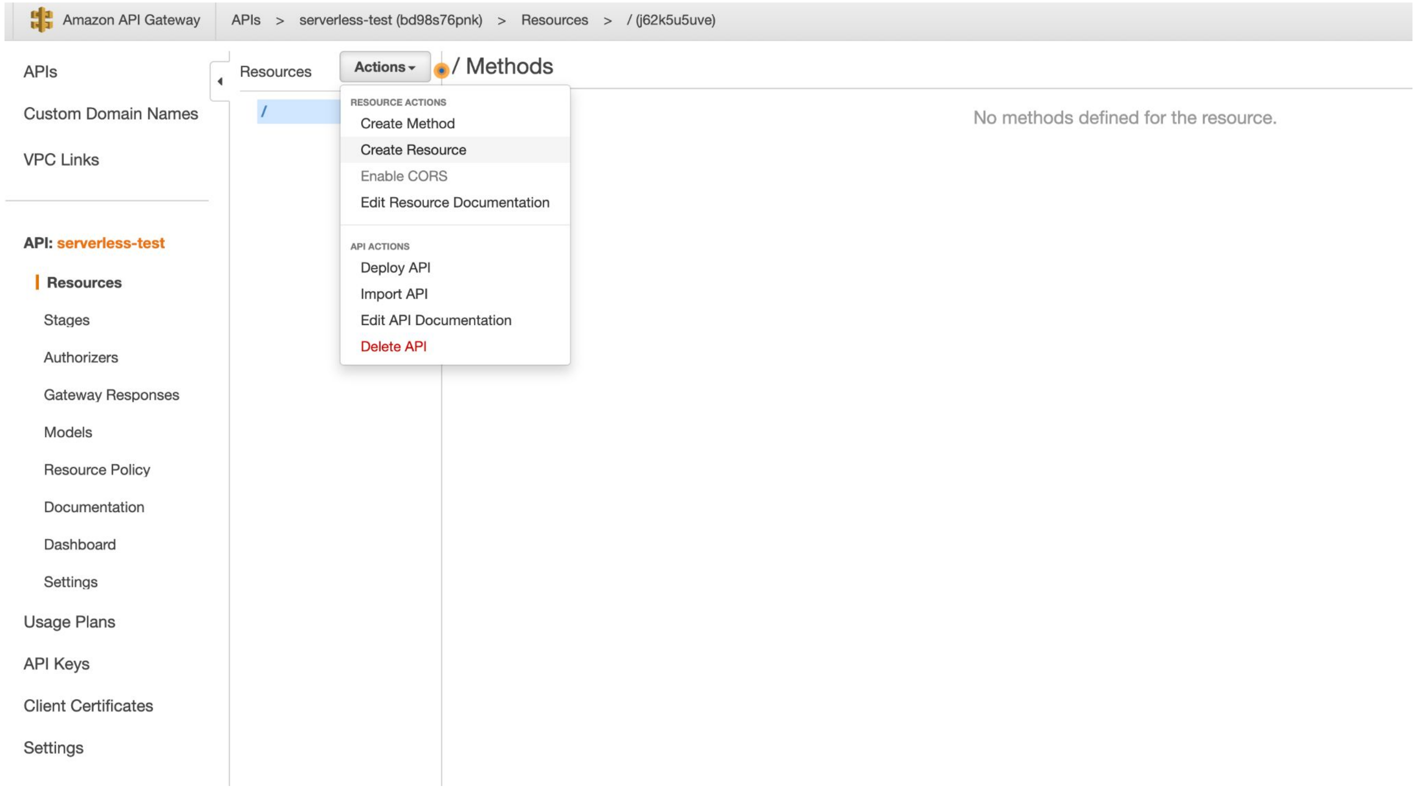
Settings

Choose a friendly name and description for your API.

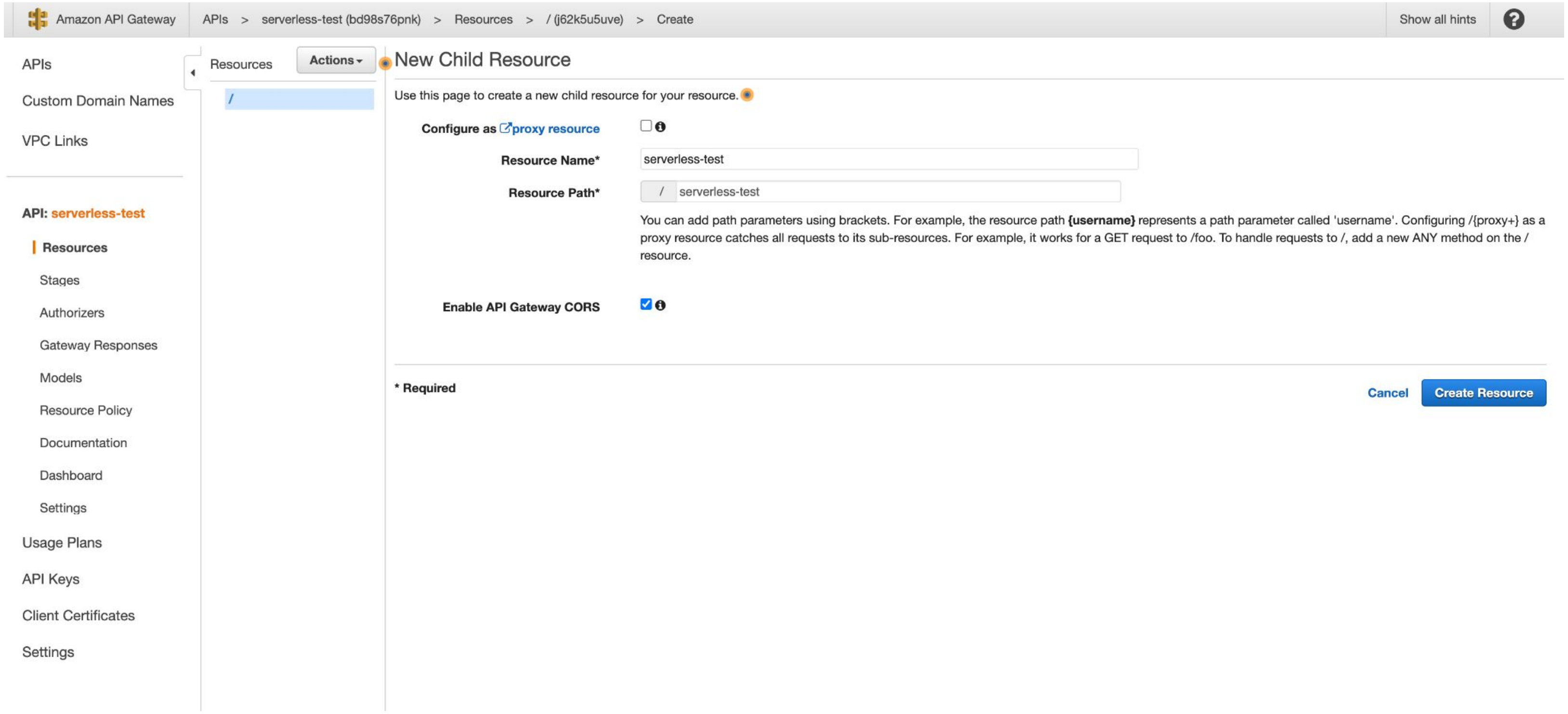
API name*	serverless-test
Description	
Endpoint Type	Regional

*** Required**

Create a resource on the API



Create a resource on the API



New Child Resource

Use this page to create a new child resource for your resource.

Configure as proxy resource ☐

Resource Name* serverless-test

Resource Path* / serverless-test

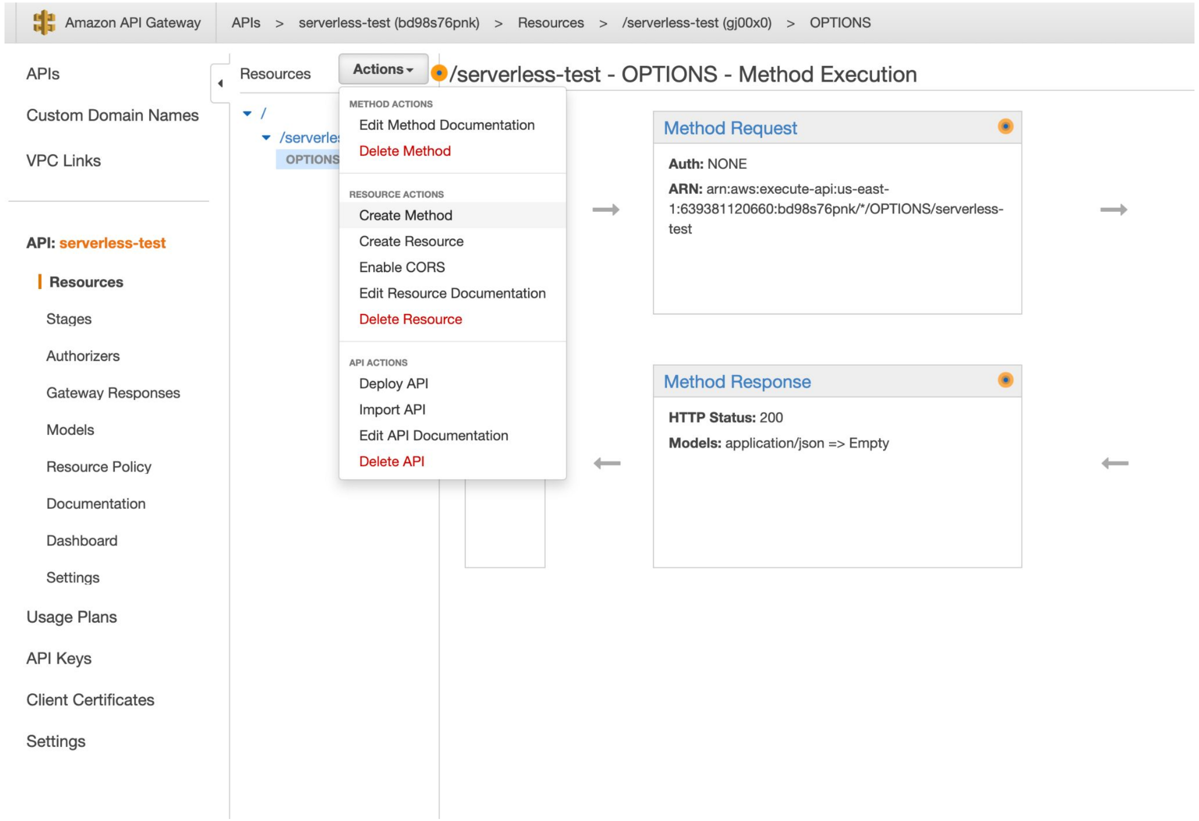
You can add path parameters using brackets. For example, the resource path **{username}** represents a path parameter called 'username'. Configuring /{proxy+} as a proxy resource catches all requests to its sub-resources. For example, it works for a GET request to /foo. To handle requests to /, add a new ANY method on the / resource.

Enable API Gateway CORS ☑

* Required

Cancel | Create Resource

Create a method on the resource



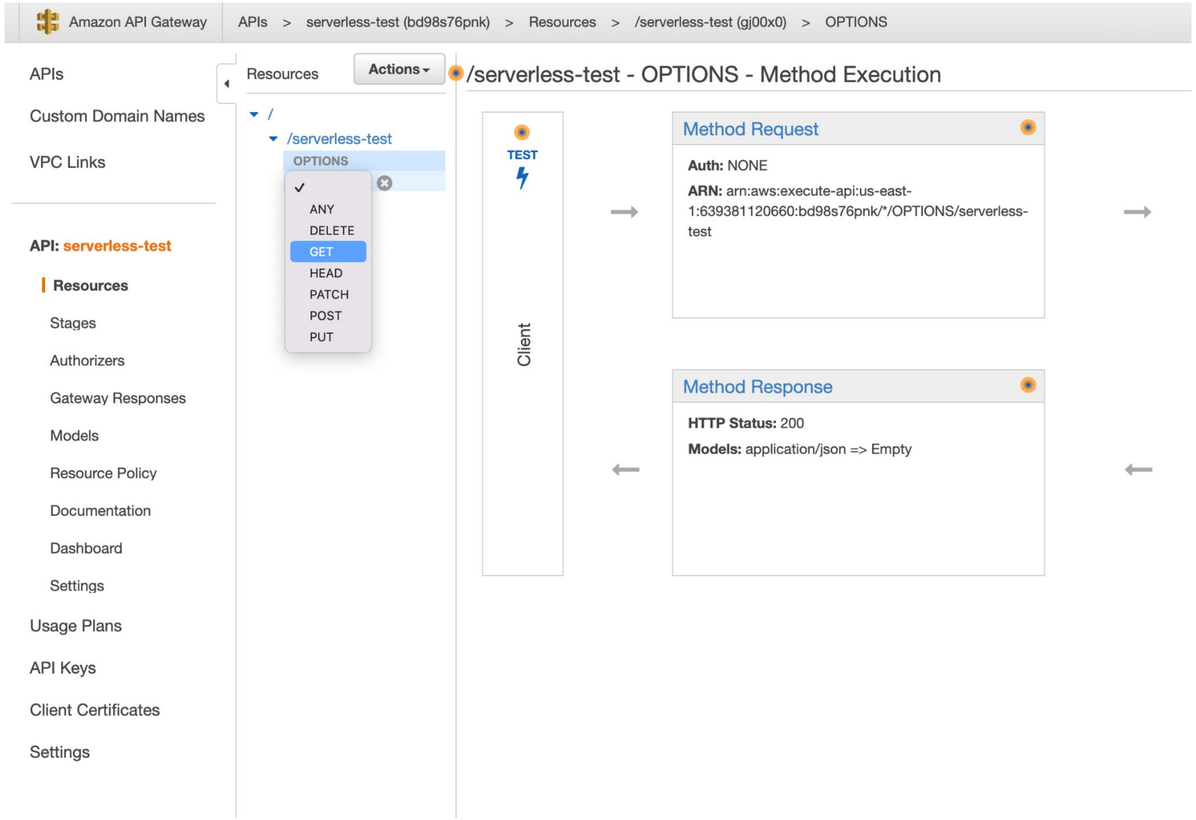
Auth: NONE

ARN: arn:aws:execute-api:us-east-1:639381120660:bd98s76pnk/*/OPTIONS/serverless-test

HTTP Status: 200

Models: application/json => Empty

Create a method on the resource



Amazon API Gateway | APIs > serverless-test (bd98s76pnk) > Resources > /serverless-test (gj00x0) > OPTIONS

/serverless-test - OPTIONS - Method Execution

Method Request

Auth: NONE

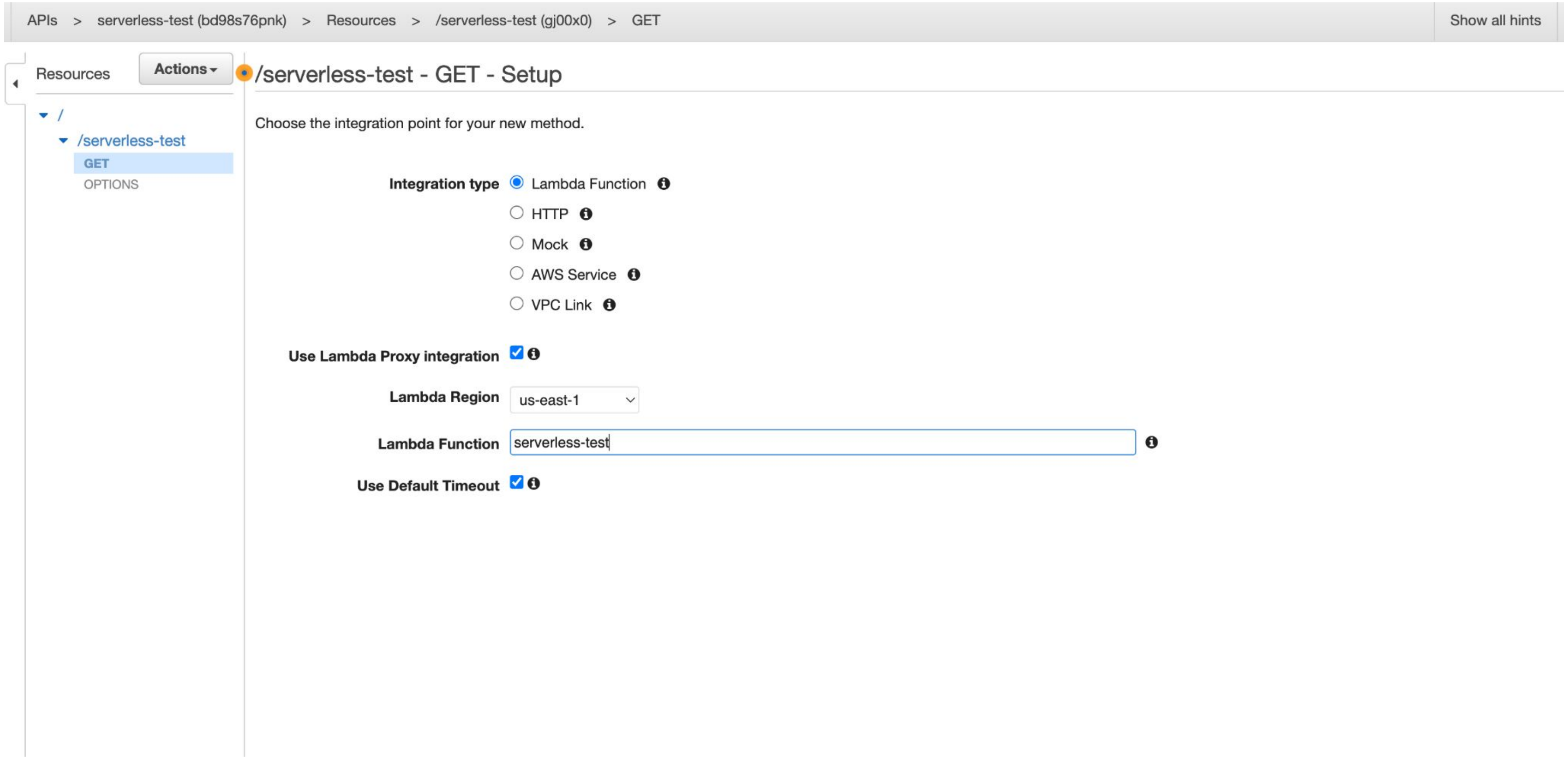
ARN: arn:aws:execute-api:us-east-1:639381120660:bd98s76pnk/*/OPTIONS/serverless-test

Method Response

HTTP Status: 200

Models: application/json => Empty

Use the Lambda function as the integration point



APIs > serverless-test (bd98s76pnk) > Resources > /serverless-test (gj00x0) > GET
Show all hints
Resources
Actions
/
/serverless-test
GET
OPTIONS

/serverless-test - GET - Setup

Choose the integration point for your new method.

Integration type [✓ Lambda Function] HTTP Mock AWS Service VPC Link

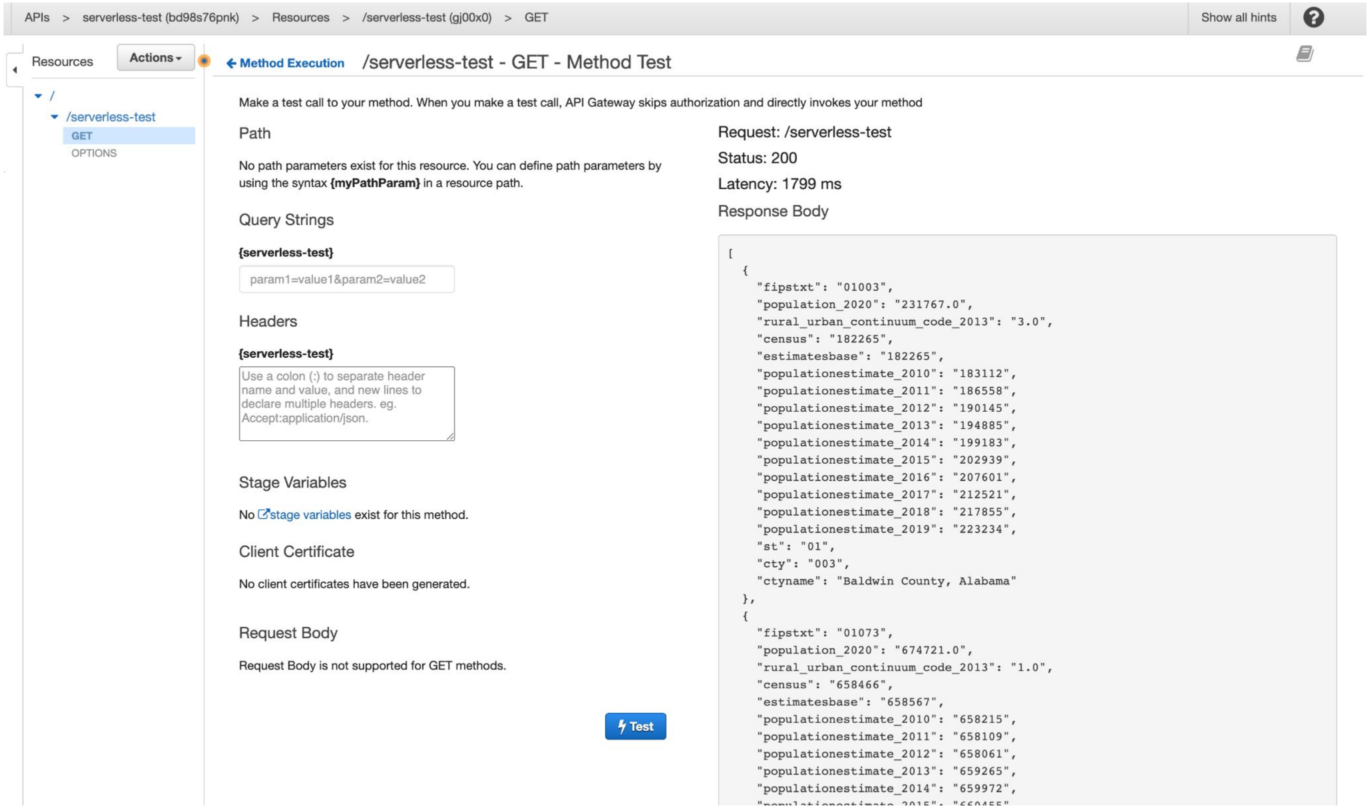
Use Lambda Proxy integration ✓

Lambda Region us-east-1

Lambda Function serverless-test

Use Default Timeout ✓

Test it out


APIs > serverless-test (bd98s76pnk) > Resources > /serverless-test (gj00x0) > GET
Show all hints
Resources
Actions
/
/serverless-test
GET
OPTIONS
Method Execution
Test

/serverless-test - GET - Method Test

Make a test call to your method. When you make a test call, API Gateway skips authorization and directly invokes your method

Path

No path parameters exist for this resource. You can define path parameters by using the syntax **{myPathParam}** in a resource path.

Query Strings

{serverless-test}

param1=value1&param2=value2

Headers

{serverless-test}

Use a colon (:) to separate header name and value, and new lines to declare multiple headers. eg. Accept:application/json.

Stage Variables

No stage variables exist for this method.

Client Certificate

No client certificates have been generated.

Request Body

Request Body is not supported for GET methods.

Request: /serverless-test

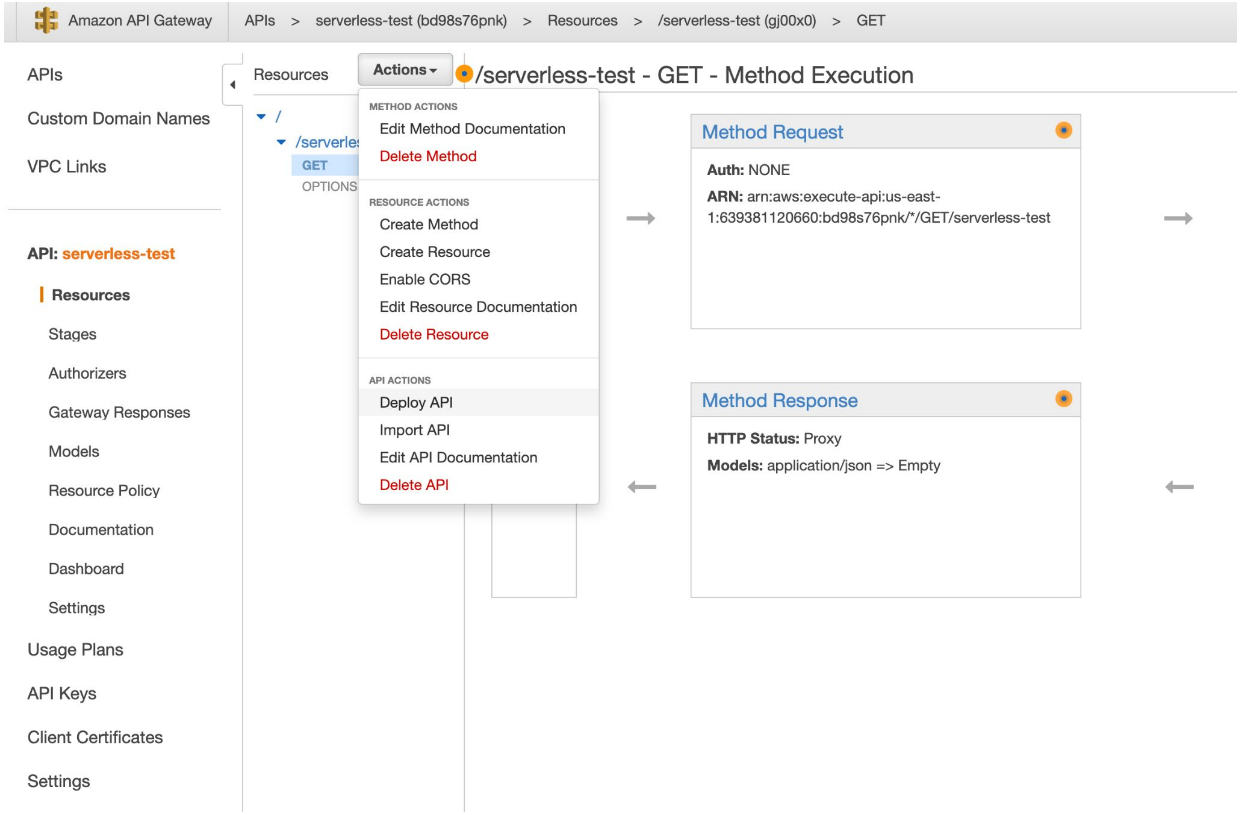
Status: 200

Latency: 1799 ms

Response Body

```
[
  {
    "fipstxt": "01003",
    "population_2020": "231767.0",
    "rural_urban_continuum_code_2013": "3.0",
    "census": "182265",
    "estimatesbase": "182265",
    "populationestimate_2010": "183112",
    "populationestimate_2011": "186558",
    "populationestimate_2012": "190145",
    "populationestimate_2013": "194885",
    "populationestimate_2014": "199183",
    "populationestimate_2015": "202939",
    "populationestimate_2016": "207601",
    "populationestimate_2017": "212521",
    "populationestimate_2018": "217855",
    "populationestimate_2019": "223234",
    "st": "01",
    "cty": "003",
    "ctyname": "Baldwin County, Alabama"
  },
  {
    "fipstxt": "01073",
    "population_2020": "674721.0",
    "rural_urban_continuum_code_2013": "1.0",
    "census": "658466",
    "estimatesbase": "658567",
    "populationestimate_2010": "658215",
    "populationestimate_2011": "658109",
    "populationestimate_2012": "658061",
    "populationestimate_2013": "659265",
    "populationestimate_2014": "659972",
```

Deploy the API (so we can access it)



Amazon API Gateway | APIs > serverless-test (bd98s76pnk) > Resources > /serverless-test (gj00x0) > GET

APIs
Custom Domain Names
VPC Links

API: serverless-test
Resources
Stages
Authorizers
Gateway Responses
Models
Resource Policy
Documentation
Dashboard
Settings
Usage Plans
API Keys
Client Certificates
Settings

Resources | Actions

/
/serverles
GET
OPTIONS

METHOD ACTIONS
Edit Method Documentation
Delete Method

RESOURCE ACTIONS
Create Method
Create Resource
Enable CORS
Edit Resource Documentation
Delete Resource

API ACTIONS
Deploy API
Import API
Edit API Documentation
Delete API

/serverless-test - GET - Method Execution

Method Request

Auth: NONE

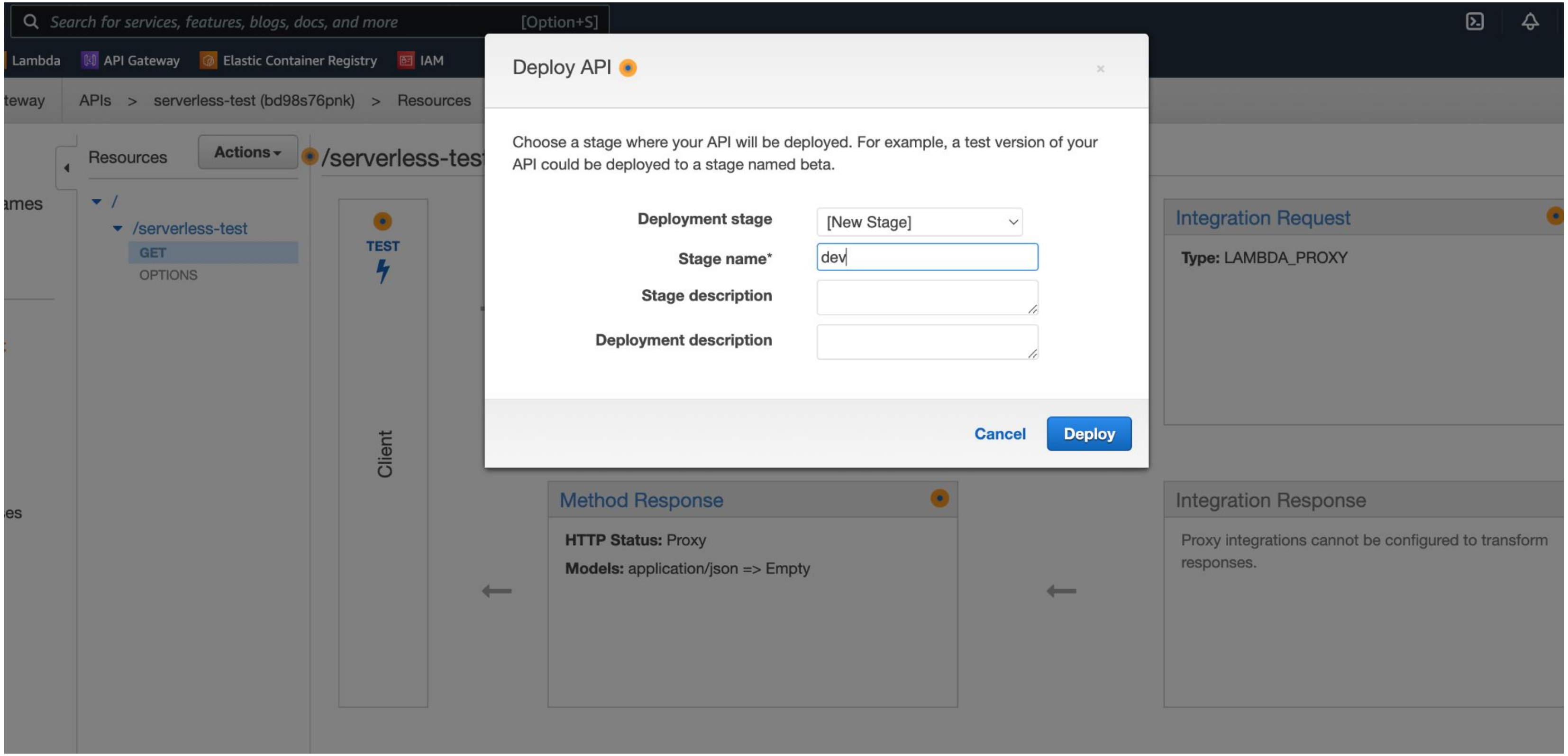
ARN: arn:aws:execute-api:us-east-1:639381120660:bd98s76pnk/*/GET/serverless-test

Method Response

HTTP Status: Proxy

Models: application/json => Empty

Name the deployment



Deploy API

Choose a stage where your API will be deployed. For example, a test version of your API could be deployed to a stage named beta.

Deployment stage	[New Stage]
Stage name*	dev
Stage description	
Deployment description	

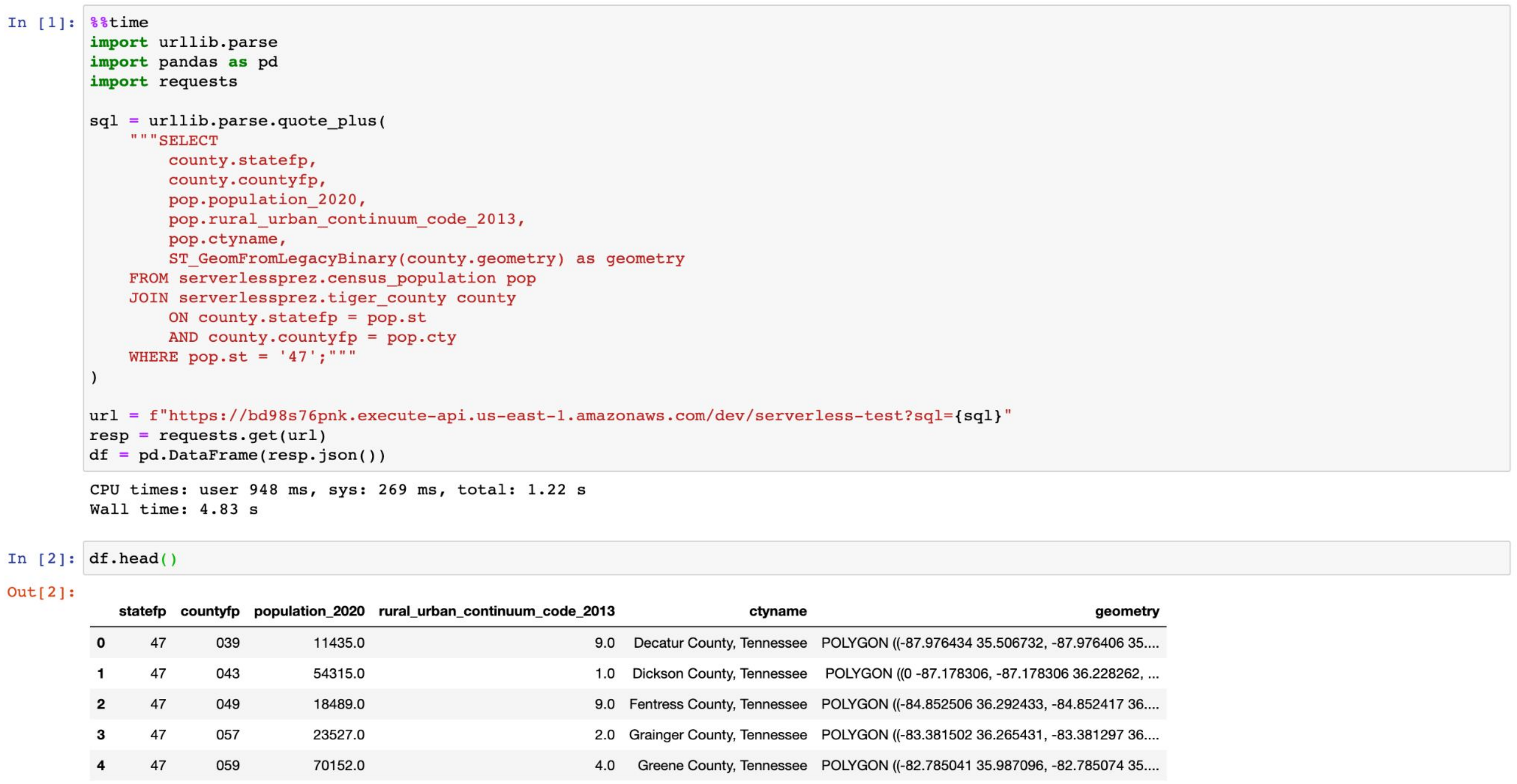
Type: LAMBDA_PROXY

HTTP Status: Proxy

Models: application/json => Empty

Proxy integrations cannot be configured to transform responses.

Now let’s call it from Jupyter



In [1]:
```
%%time
import urllib.parse
import pandas as pd
import requests

sql = urllib.parse.quote_plus(
    """SELECT
        county.statefp,
        county.countyfp,
        pop.population_2020,
        pop.rural_urban_continuum_code_2013,
        pop.ctyname,
        ST_GeomFromLegacyBinary(county.geometry) as geometry
    FROM serverlessprez.census_population pop
    JOIN serverlessprez.tiger_county county
        ON county.statefp = pop.st
        AND county.countyfp = pop.cty
    WHERE pop.st = '47';"""
)

url = f"https://bd98s76pnk.execute-api.us-east-1.amazonaws.com/dev/serverless-test?sql={sql}"
resp = requests.get(url)
df = pd.DataFrame(resp.json())
```

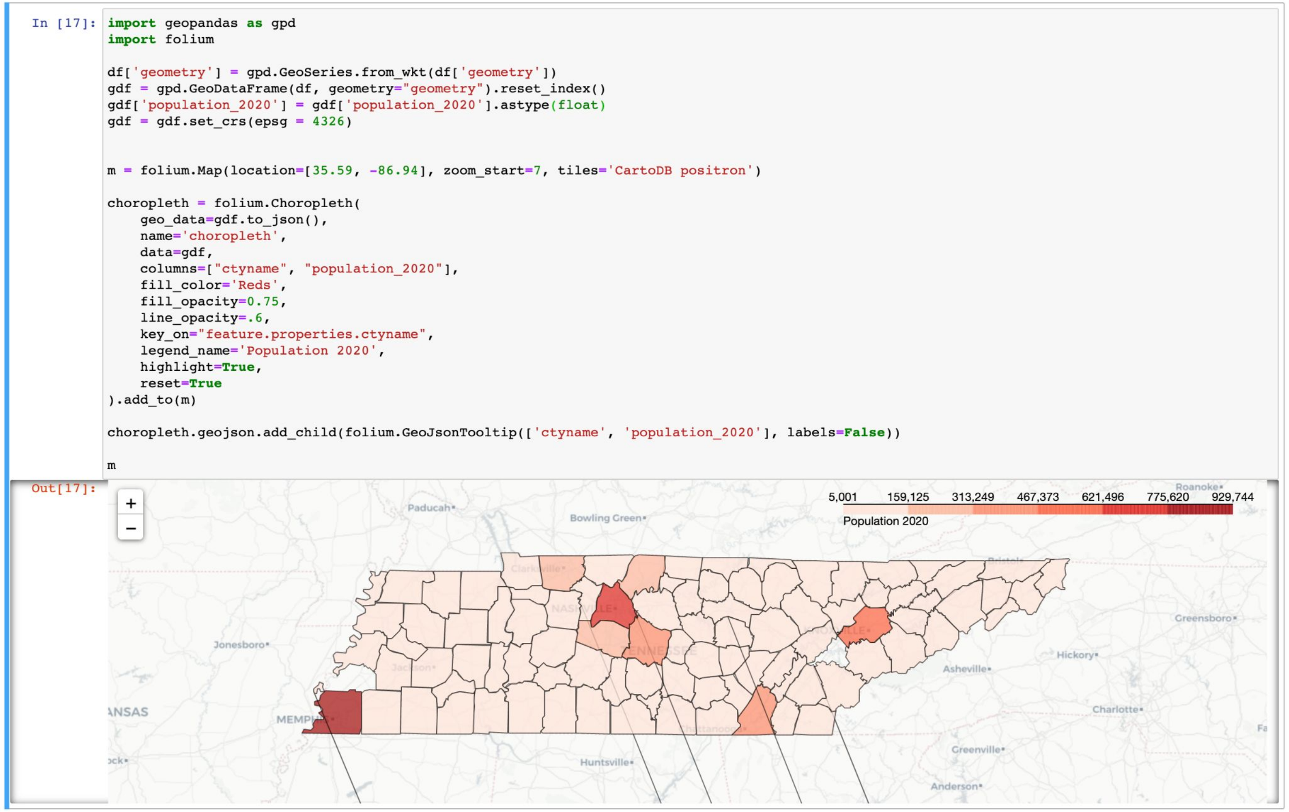
```
CPU times: user 948 ms, sys: 269 ms, total: 1.22 s
Wall time: 4.83 s
```

In [2]:
```
df.head()
```

Out[2]:

	statefp	countyfp	population_2020	rural_urban_continuum_code_2013	ctyname	geometry
0	47	039	11435.0	9.0	Decatur County, Tennessee	POLYGON ((-87.976434 35.506732, -87.976406 35....
1	47	043	54315.0	1.0	Dickson County, Tennessee	POLYGON ((0 -87.178306, -87.178306 36.228262, ...
2	47	049	18489.0	9.0	Fentress County, Tennessee	POLYGON ((-84.852506 36.292433, -84.852417 36....
3	47	057	23527.0	2.0	Grainger County, Tennessee	POLYGON ((-83.381502 36.265431, -83.381297 36....
4	47	059	70152.0	4.0	Greene County, Tennessee	POLYGON ((-82.785041 35.987096, -82.785074 35....

Doing more stuff with that data in Jupyter



In [17]:
```
import geopandas as gpd
import folium

df['geometry'] = gpd.GeoSeries.from_wkt(df['geometry'])
gdf = gpd.GeoDataFrame(df, geometry="geometry").reset_index()
gdf['population_2020'] = gdf['population_2020'].astype(float)
gdf = gdf.set_crs(epsg = 4326)


m = folium.Map(location=[35.59, -86.94], zoom_start=7, tiles='CartoDB positron')

choropleth = folium.Choropleth(
    geo_data=gdf.to_json(),
    name='choropleth',
    data=gdf,
    columns=["ctyname", "population_2020"],
    fill_color='Reds',
    fill_opacity=0.75,
    line_opacity=.6,
    key_on="feature.properties.ctyname",
    legend_name='Population 2020',
    highlight=True,
    reset=True
).add_to(m)

choropleth.geojson.add_child(folium.GeoJsonTooltip(['ctyname', 'population_2020'], labels=False))

m
```

Out[17]:

Need help with a project or have questions?

Timothy Dobbins
Data Scientist|Data Engineer|Consultant

Github: github.com/tmthyjames
Linkedin: linkedin.com/in/tim-dobbins
Email: tmthyjames@gmail.com

Or scan this →

Presentation Github link: github.com/tmthyjames/serverless-prez