**Serverless Order Management REST API Documentation**

This document explains the serverless Order Management REST API implementation using Serverless Framework and deployed in AWS. It uses AWS Lambda and API Gateway functionality and saves the orders in the AWS dynamodb at the backend. Once it is deployed in AWS cloud, it can be accessed from anywhere using the REST endpoints provided to manage the orders.

**Prerequisites for Serverless Framework:**

1. Install node, npm
2. Install Serverless Framework
3. Oracle JDK 10
4. Apache Maven

**Used Serveless boilerplate template to create project**

serverless create --template aws-java-maven --name orders-api -p aws-java-orders-api

**Once we add the Orders POJO and implement the GET, POST, DELETE REST API end points, we can deploy the project using serverless into AWS**

**Deploying the service in AWS**

1. **Build the project using “mvn clean install”**
2. **Deploy the service to the cloud using** “**sls deploy”**

**Once the service is deployed in the cloud, it will disply the rest endpoints as shown below**

REST endpoints:

  GET - <https://15k7wa6k5h.execute-api.us-east-1.amazonaws.com/dev/orders>

  GET - [https://15k7wa6k5h.execute-api.us-east-1.amazonaws.com/dev/orders/{id}](https://15k7wa6k5h.execute-api.us-east-1.amazonaws.com/dev/orders/%7bid%7d)

  POST - <https://15k7wa6k5h.execute-api.us-east-1.amazonaws.com/dev/orders>

  DELETE - [https://15k7wa6k5h.execute-api.us-east-1.amazonaws.com/dev/orders/{id}](https://15k7wa6k5h.execute-api.us-east-1.amazonaws.com/dev/orders/%7bid%7d)

We can use these rest endpoings to add order, get all orders, get order based on order id and delete order based on order id as shown below:

**REST API to add/ show and delete orders**

**Adding an order:**

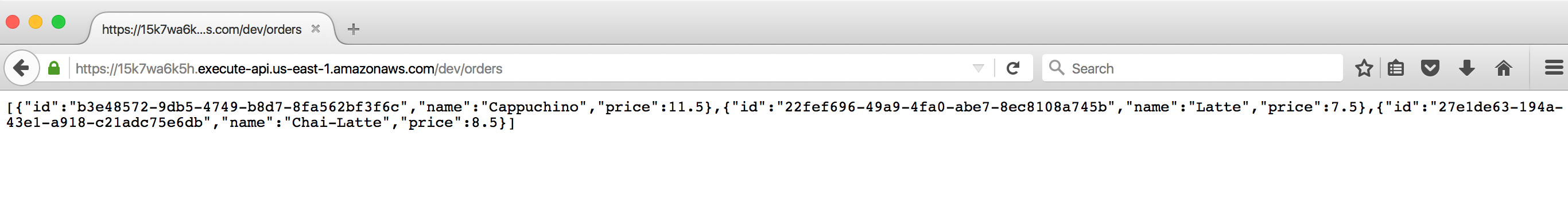
curl -X POST https://15k7wa6k5h.execute-api.us-east-1.amazonaws.com/dev/orders -d '{"name": "Tea", "price": 6.00}'

{"id":"1785f486-479b-45b0-88aa-c0ae01a97969","name":"Tea","price":6.0}

**Listing all the orders:**

abhinaya-mbp:su18-202-py abhinaya$ curl https://15k7wa6k5h.execute-api.us-east-1.amazonaws.com/dev/orders

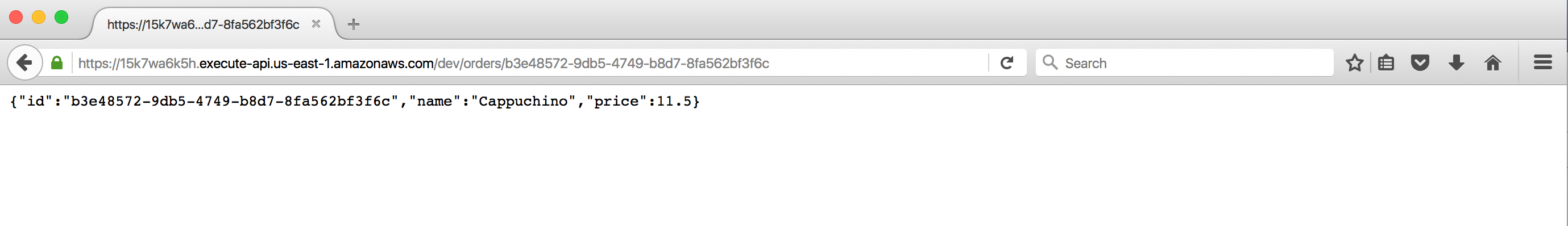
[{"id":"b3e48572-9db5-4749-b8d7-8fa562bf3f6c","name":"Cappuchino","price":11.5},{"id":"1785f486-479b-45b0-88aa-c0ae01a97969","name":"Tea","price":6.0},{"id":"22fef696-49a9-4fa0-abe7-8ec8108a745b","name":"Latte","price":7.5},{"id":"27e1de63-194a-43e1-a918-c21adc75e6db","name":"Chai-Latte","price":8.5}]



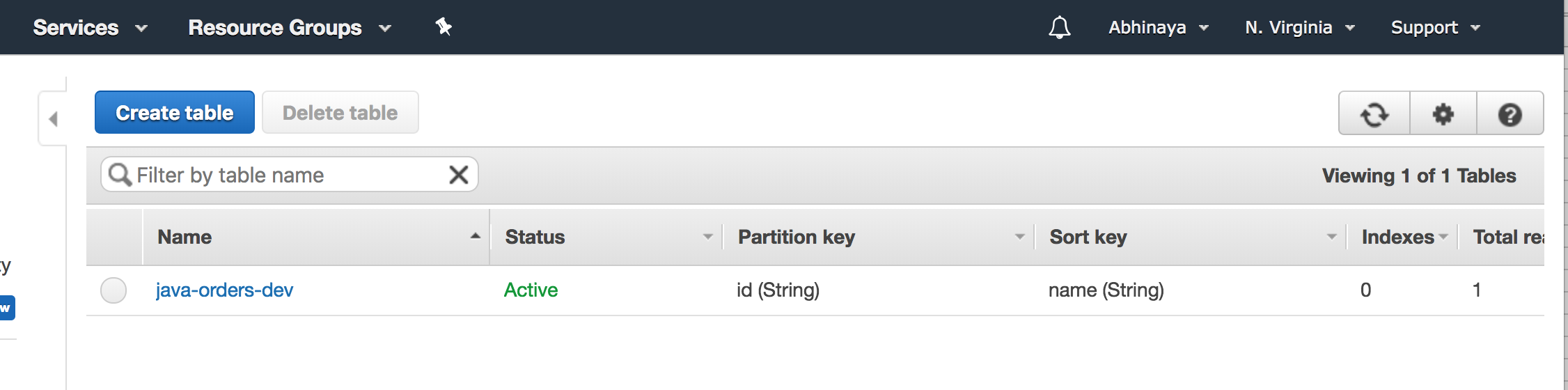
**Deleting an order**

curl -X DELETE <https://15k7wa6k5h.execute-api.us-east-1.amazonaws.com/dev/orders/1785f486-479b-45b0-88aa-c0ae01a97969>

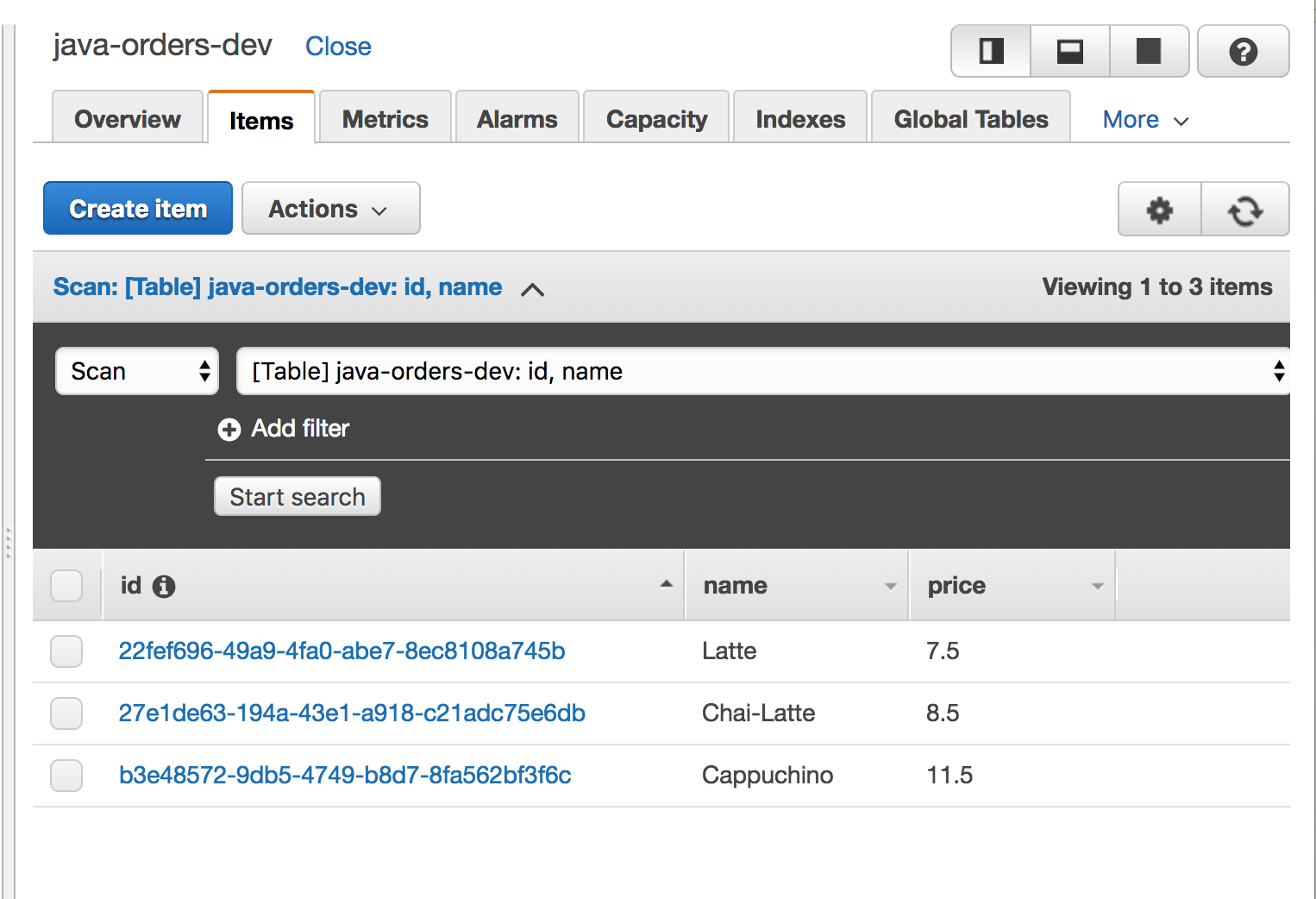
**To list an order based on order id**



**Table created in the AWS dynamodb table:**



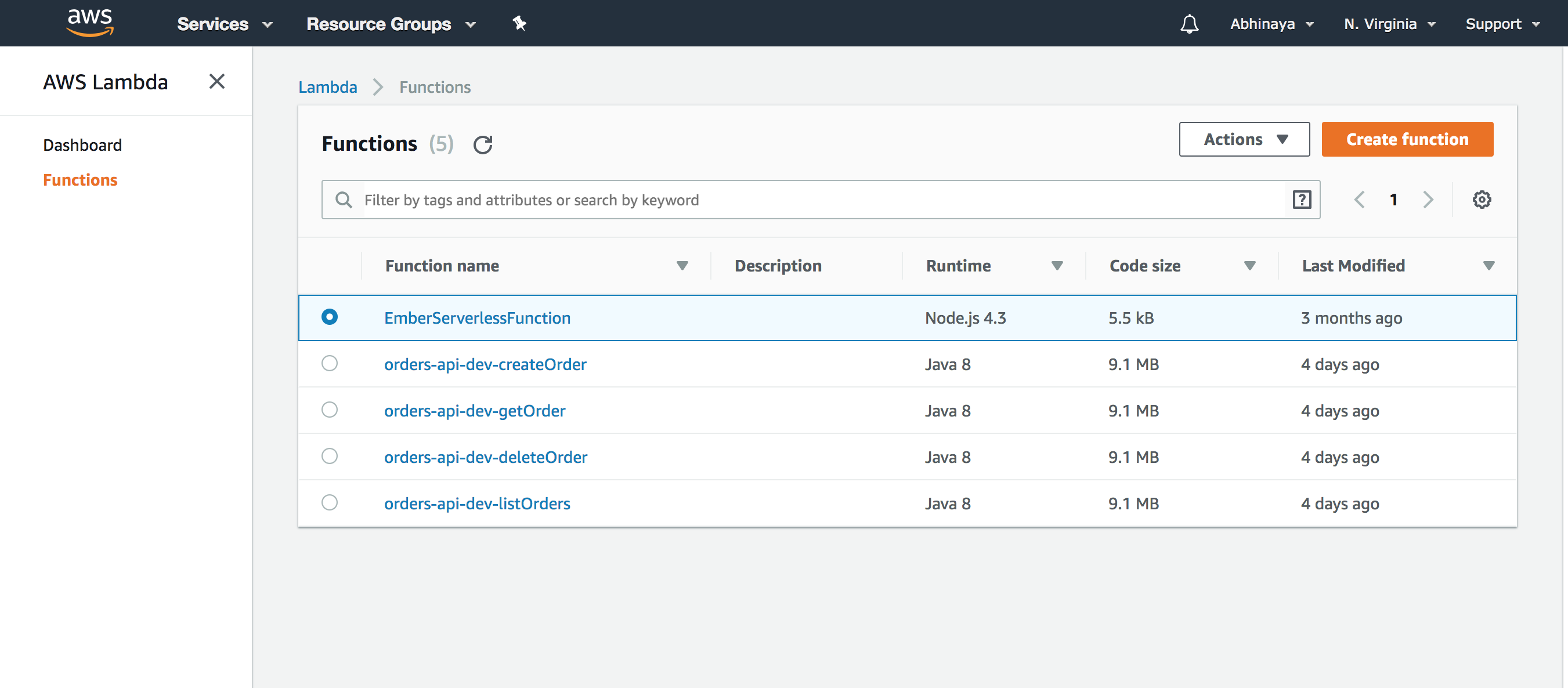
**Orders saved in the AWS dynamo db table:**



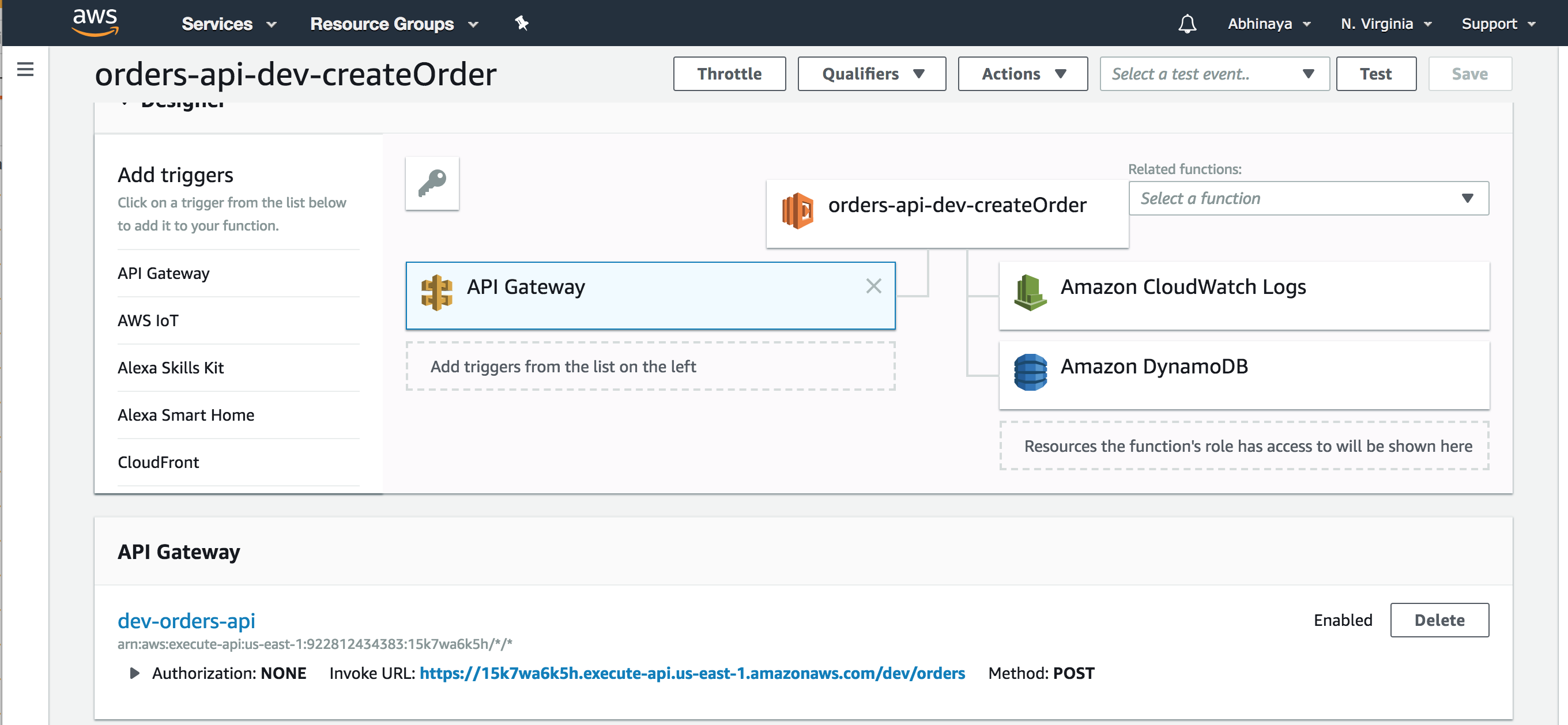
**AWS Lamba APIs:**

Please ignore the “emberServerlessFunction in the screenshot below. That was for a different project.

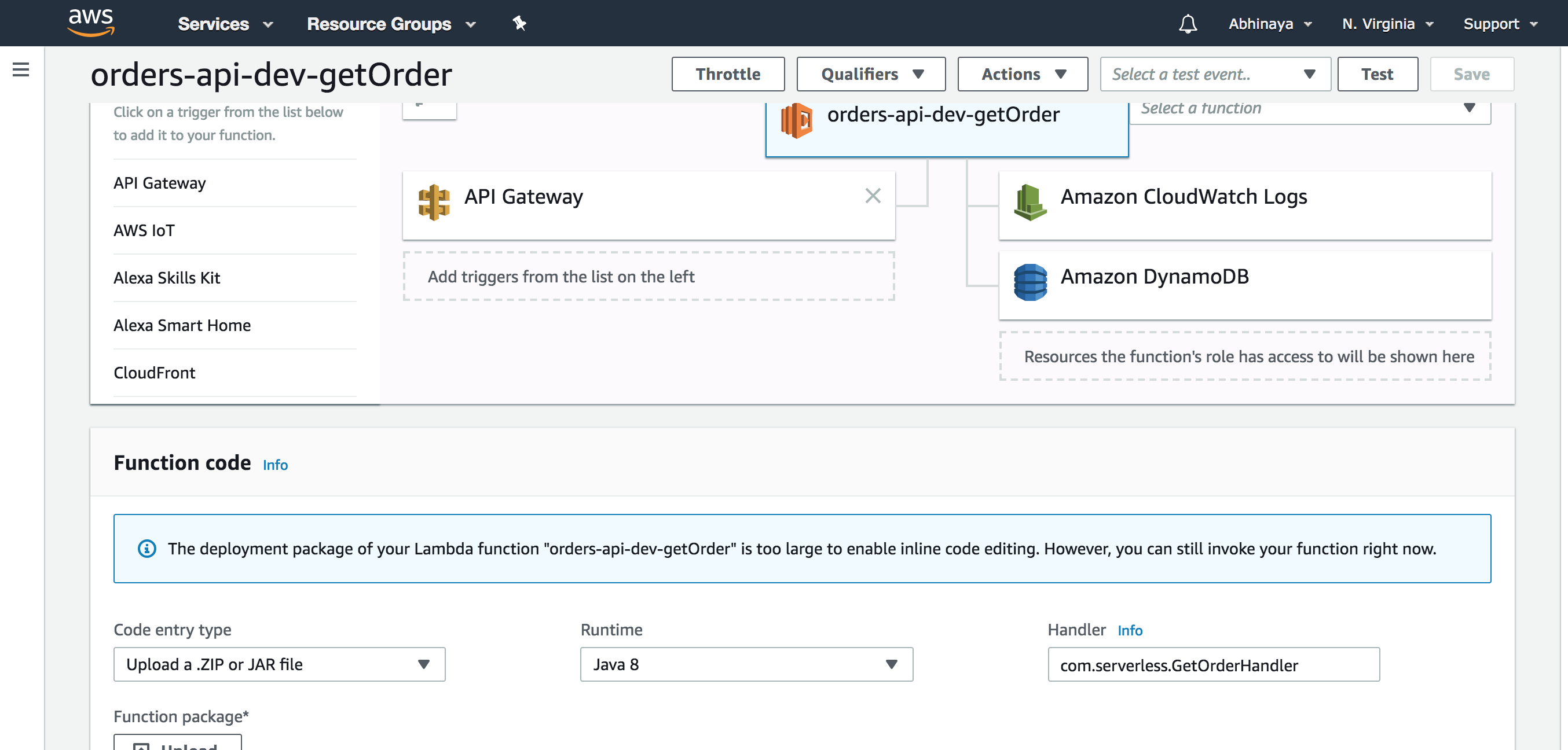
The orders-api-dev-CreateOrder, orders-api-dev-getOrder, orders-api-dev-deleteOrder and orders-api-dev-listOrders are the Lamba functions for this serverless project.



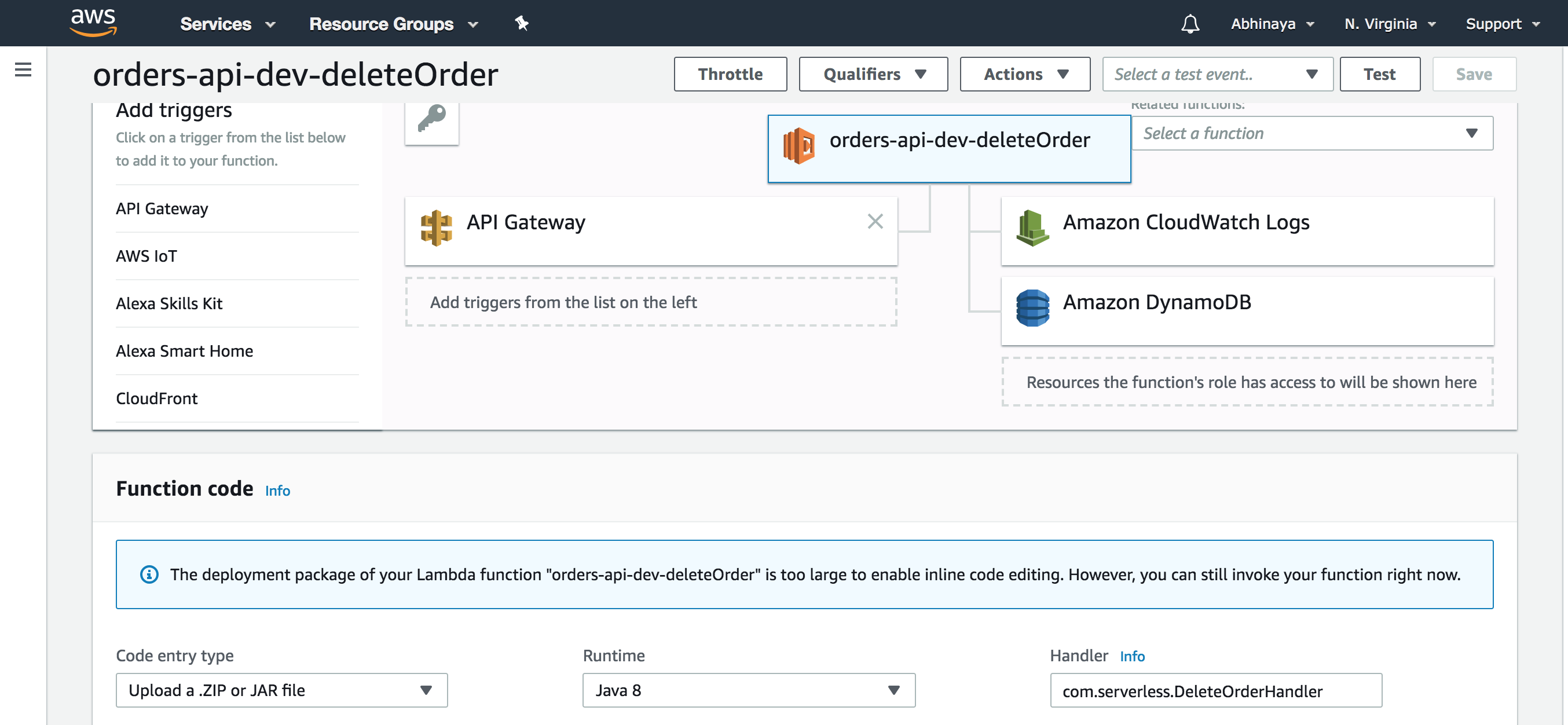
**Create Order Lambda function:**



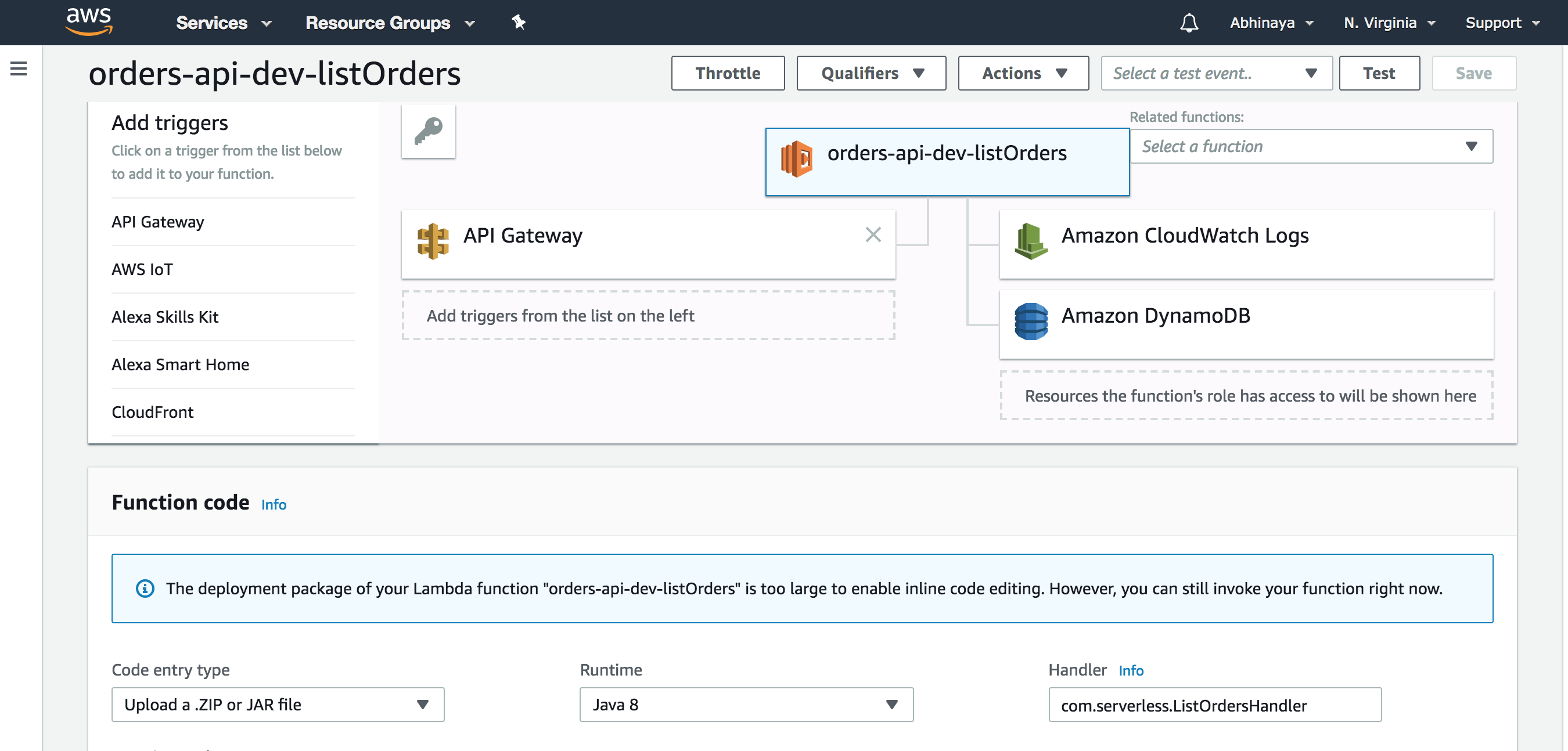
**Get a specific order based on id Lambda function:**



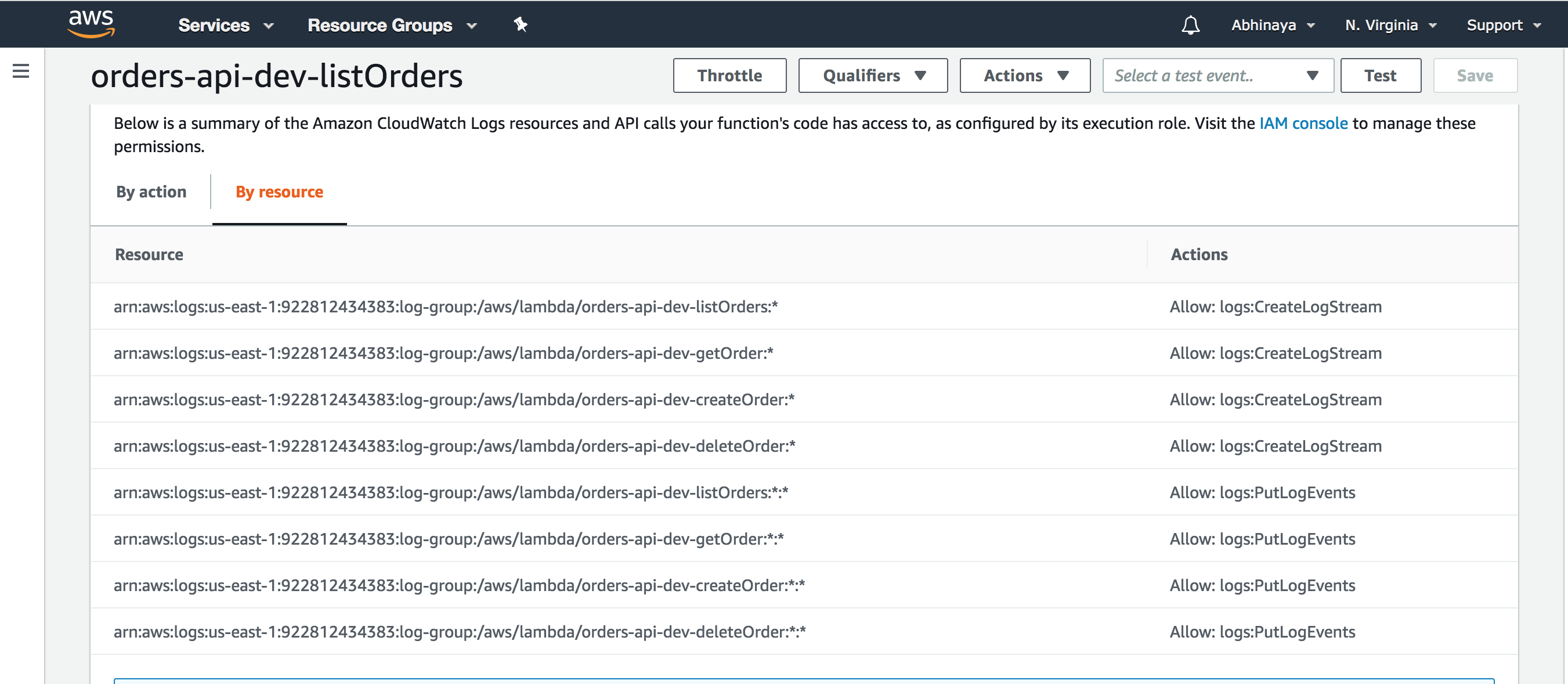
**Delete order based on id Lambda function:**



Show all orders Lambda function:



**Cloud formation logs:**



**References:**

<https://blog.sourcerer.io/full-guide-to-developing-rest-apis-with-aws-api-gateway-and-aws-lambda-d254729d6992>

<https://medium.freecodecamp.org/quickly-create-a-serverless-restful-api-with-nodejs-and-aws-lambda-api-gateway-and-a6be891cc16a>

<https://serverless.com/blog/how-to-create-a-rest-api-in-java-using-dynamodb-and-serverless/>

<https://medium.freecodecamp.org/a-crash-course-on-securing-serverless-apis-with-json-web-tokens-ff657ab2f5a5>

<https://github.com/serverless/examples>