KUMAR GAURAV 20122065

LAB 4, CLOUD COMPUTING

1. **List the different types of SQL and NoSQL database services available in AWS.**

SQL databases are relational, NoSQL databases are non-relational. ... SQL databases are table-based, while NoSQL databases are document, key-value, graph, or wide-column stores. SQL databases are better for multi-row transactions, while NoSQL is better for unstructured data like documents or JSON

Amazon Aurora. ...

Amazon Relational Database Service (RDS) ...

Amazon Redshift. ...

Key-value Database. ...

Amazon DynamoDB. ...

In-memory Database. ...

Amazon ElastiCache for Memcached.

RDS is an SQL database service and DynamoDB, Neptune, Amazon QLDB,

Amazon DocumentDB, Amazon Keyspaces and Amazon Timestream are NoSQL database services in AWS. The database services available in AWS are RDS (Relational Database Servies) , DynamoDB,

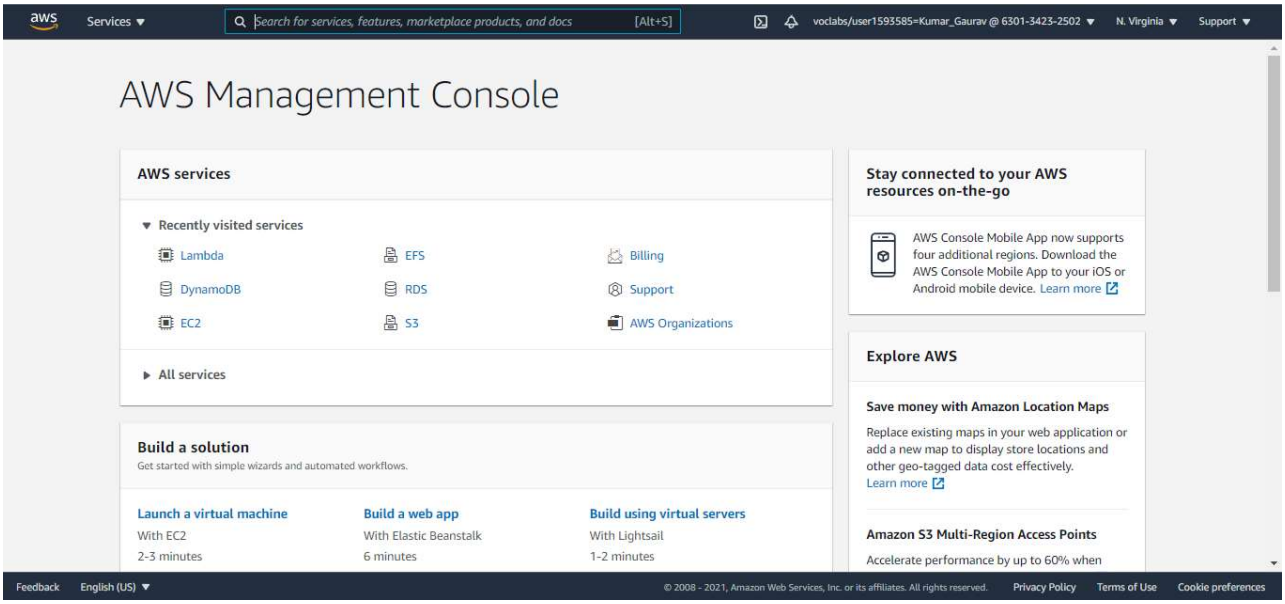
ElastiCache, Neptune, Amazon QLDB (Quantum Ledger Database), Amazon DocumentDB, Amazon

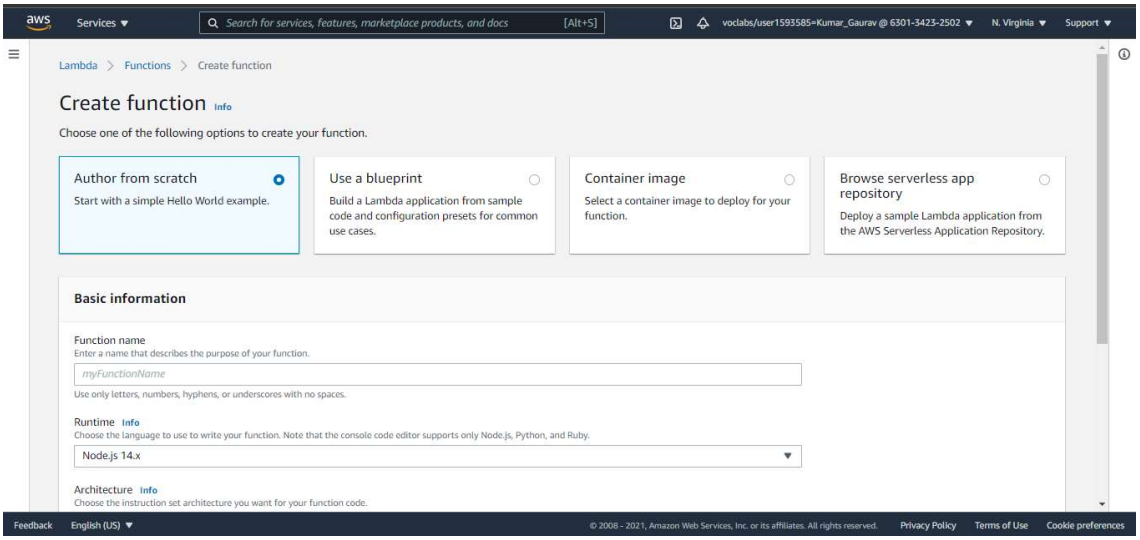
Keyspaces, Amazon Timestream and Amazon MemoryDB for Redis

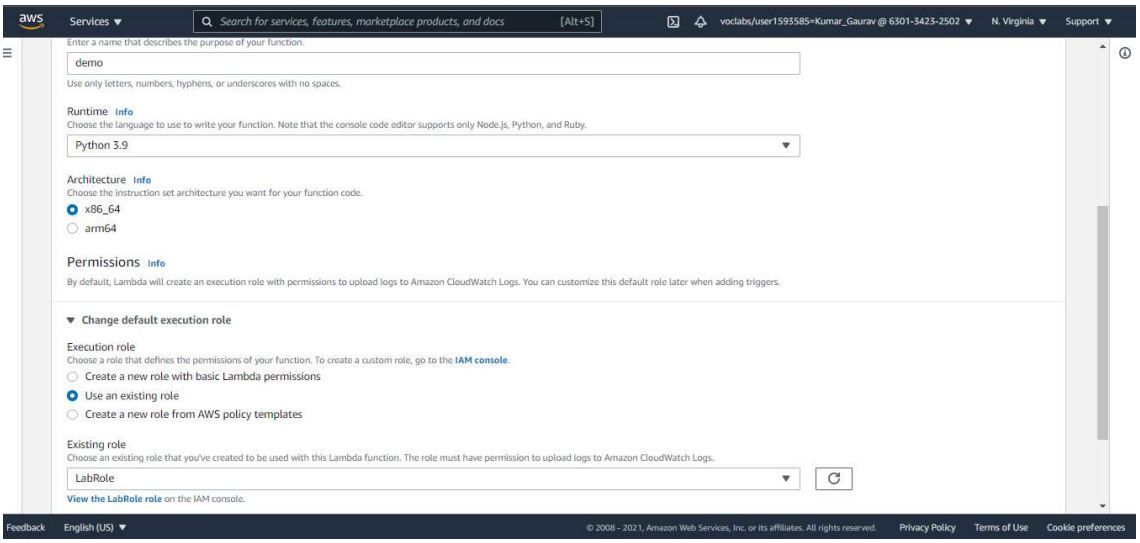
SQL databases provide great benefits for transactional data whose structure doesn't change frequently (or at all) and where data integrity is paramount. It's also best for fast analytical queries. NoSQL databases provide much more flexibility and scalability, which lends itself to rapid development and iteration.

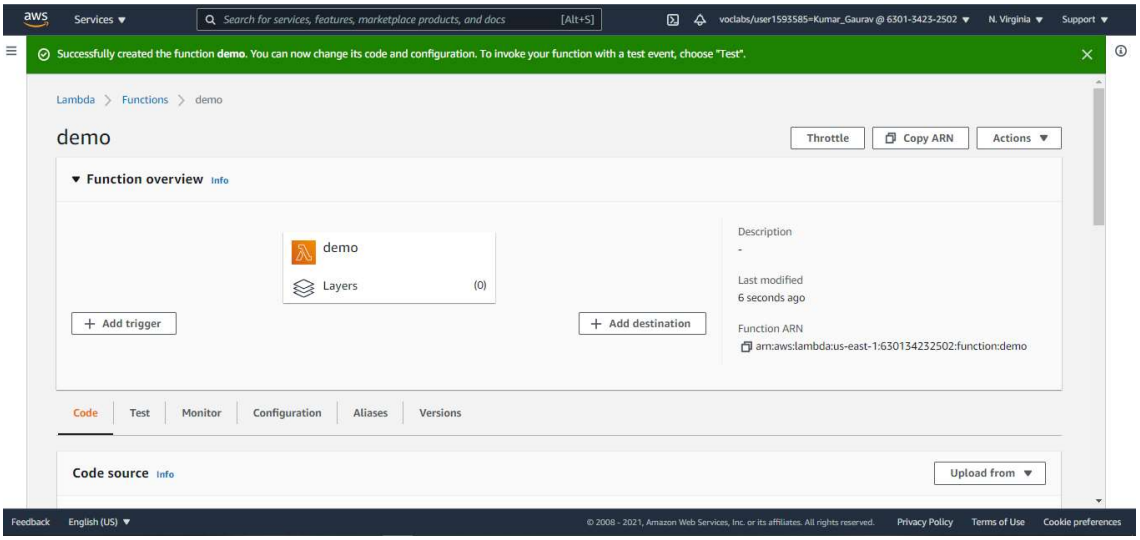
2. **Consider a weather forecast dataset stored in S3 (JSON or CSV) which has**

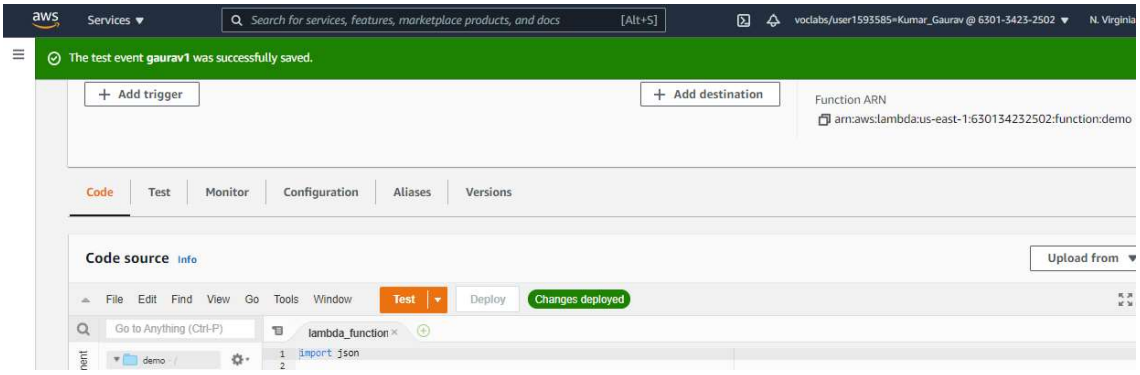
3. **temperature as one of the feature. Extract at-least 5 most important feature from the**
4. **dataset and check if the temperature value of the day is > 20 degree then only store**
5. **the data in the DynamoDB for Analysis.**

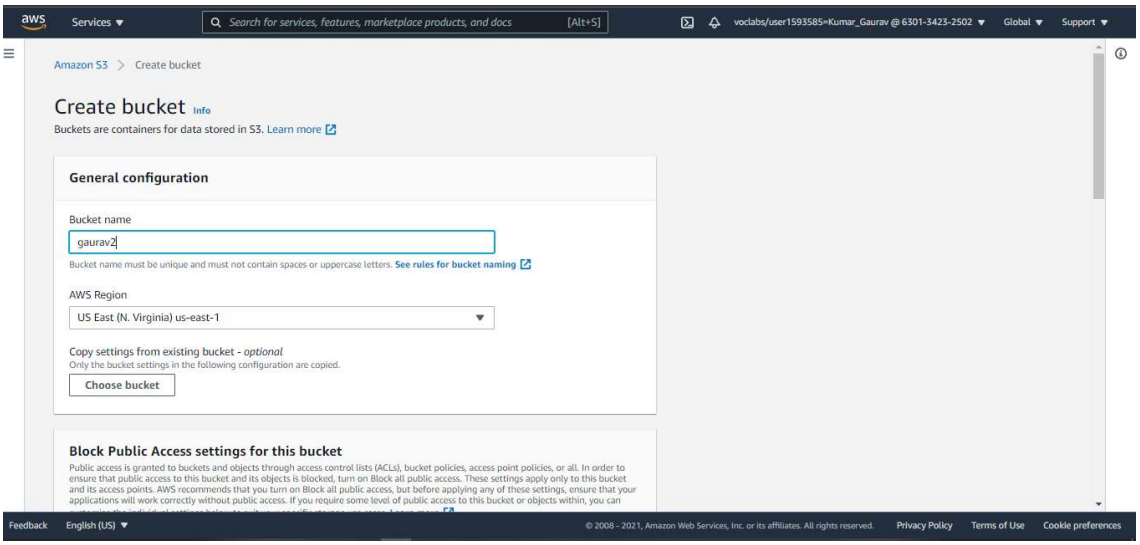


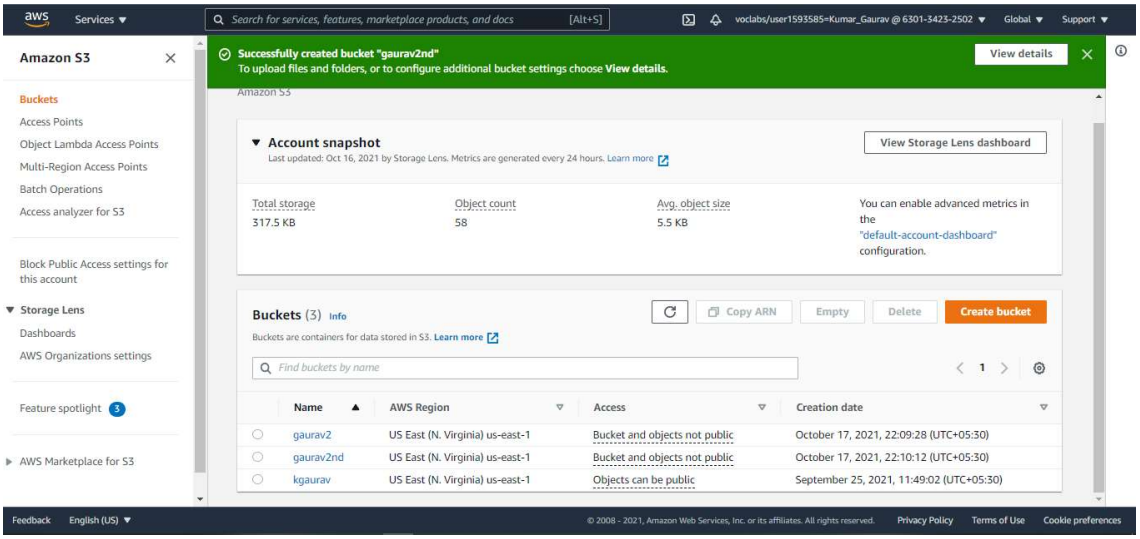


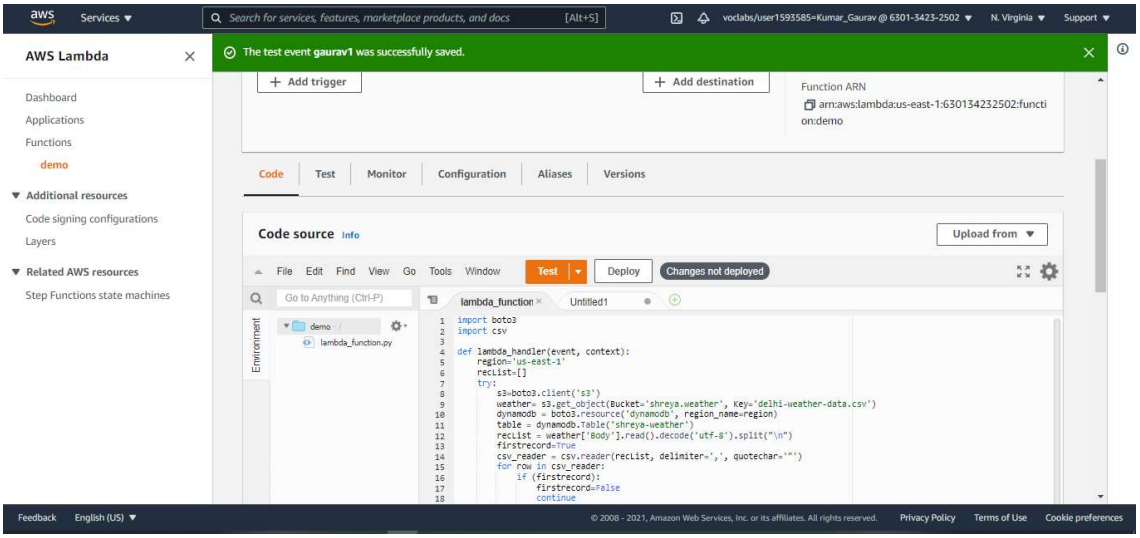

Services
Search for services, features, marketplace products, and docs [Alt+S]
voclabs/user1593585=Kumar_Gaurav @ 6301-3423-2502
N. Virginia
Support
Enter a name that describes the purpose of your function.
demo
Use only letters, numbers, hyphens, or underscores with no spaces.
Runtime Info
Choose the 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
Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.
Create a new role with basic Lambda permissions
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.
LabRole
View the LabRole role on the IAM console.
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences

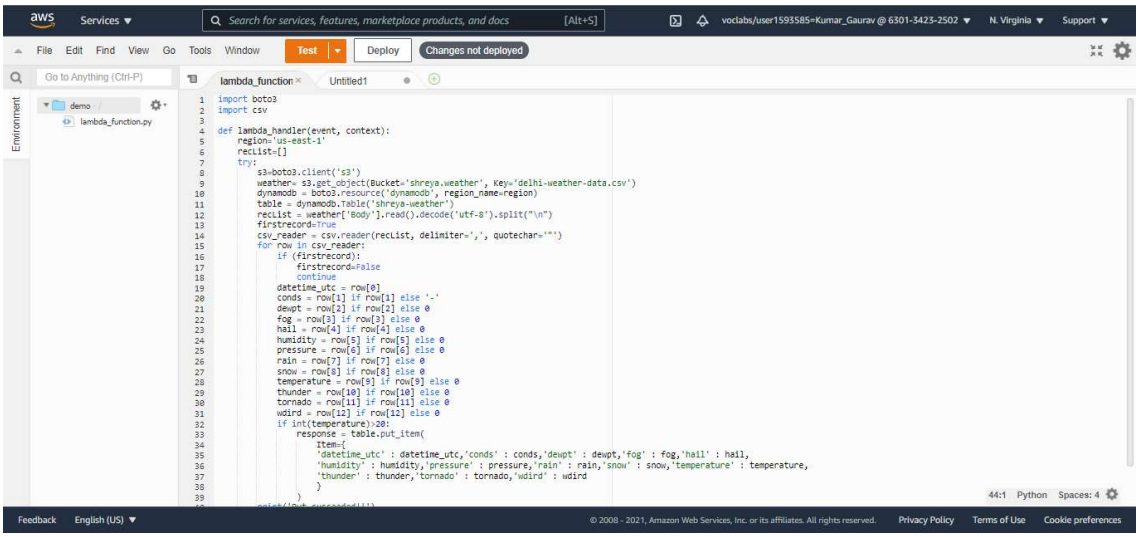

Services
Search for services, features, marketplace products, and docs [Alt+S]
voclabs/user1593585=Kumar_Gaurav @ 6301-3423-2502
N. Virginia
Support
Successfully created the function demo. You can now change its code and configuration. To invoke your function with a test event, choose "Test".
Lambda > Functions > demo
demo
Throttle
Copy ARN
Actions
Function overview Info
demo
Layers (0)
Add trigger
Add destination
Description
-
Last modified
6 seconds ago
Function ARN
arn:aws:lambda:us-east-1:630134232502:function:demo
Code
Test
Monitor
Configuration
Aliases
Versions
Code source Info
Upload from
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences


Services
Search for services, features, marketplace products, and docs [Alt+S]
voclabs/user1593585=Kumar_Gaurav @ 6301-3423-2502
N. Virginia
The test event gaurav1 was successfully saved.
Add trigger
Add destination
Function ARN
arn:aws:lambda:us-east-1:630134232502:function:demo
Code
Test
Monitor
Configuration
Aliases
Versions
Code source Info
Upload from
File Edit Find View Go Tools Window
Test
Deploy
Changes deployed
Go to Anything (Ctrl-P)
lambda_function
demo
import json

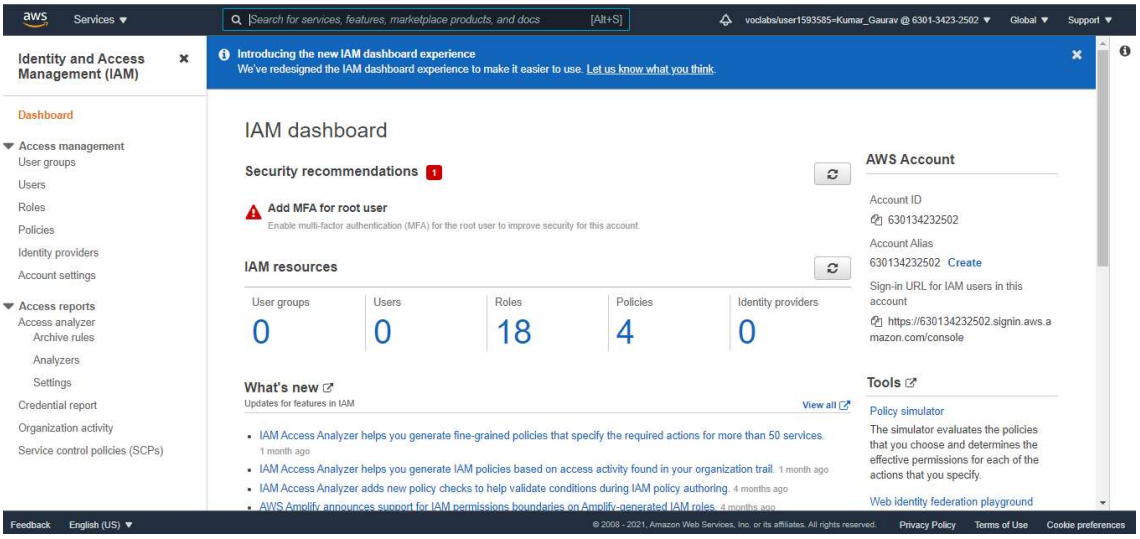

Services
Search for services, features, marketplace products, and docs [Alt+S]
voclabs/user1593585=Kumar_Gaurav @ 6301-3423-2502
Global
Support
Amazon S3 > Create bucket
Create bucket Info
Buckets are containers for data stored in S3. Learn more
General configuration
Bucket name
gaurav2
Bucket name must be unique and must not contain spaces or uppercase letters. See rules for bucket naming
AWS Region
US East (N. Virginia) us-east-1
Copy settings from existing bucket - optional
Only the bucket settings in the following configuration are copied.
Choose bucket
Block Public Access settings for this bucket
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences

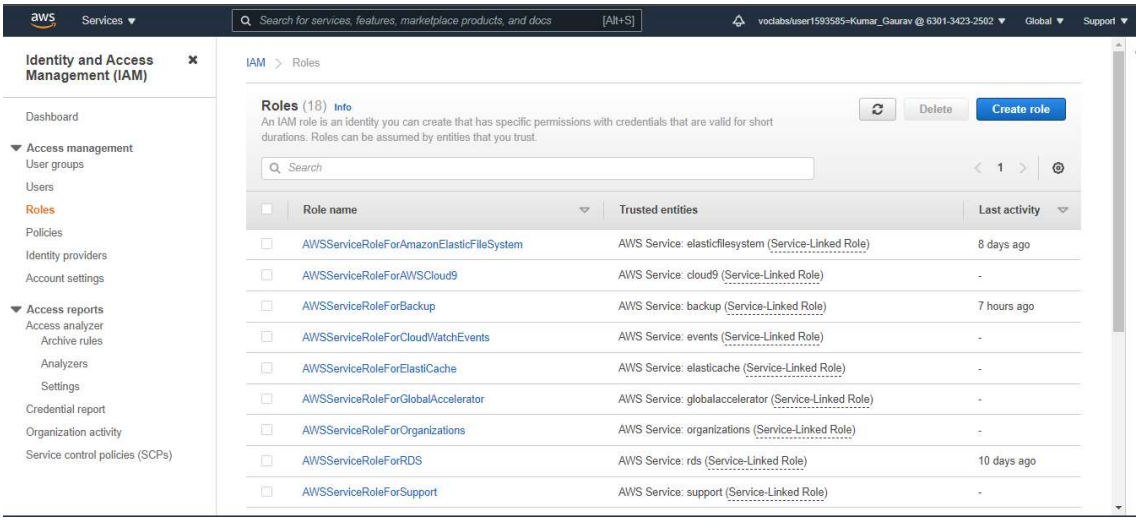

Services
Search for services, features, marketplace products, and docs [Alt+S]
voclabs/user1593585=Kumar_Gaurav @ 6301-3423-2502
Global
Support
Amazon S3
Buckets
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
Access analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
AWS Organizations settings
Feature spotlight 3
AWS Marketplace for S3
Successfully created bucket "gaurav2nd"
To upload files and folders, or to configure additional bucket settings choose View details.
View details
Amazon S3
Account snapshot
Last updated: Oct 16, 2021 by Storage Lens. Metrics are generated every 24 hours. Learn more
View Storage Lens dashboard
Total storage
317.5 KB
Object count
58
Avg. object size
5.5 KB
You can enable advanced metrics in the "default-account-dashboard" configuration.
Buckets (3) Info
Buckets are containers for data stored in S3. Learn more
Copy ARN
Empty
Delete
Create bucket
Find buckets by name
Name
AWS Region
Access
Creation date
gaurav2
US East (N. Virginia) us-east-1
Bucket and objects not public
October 17, 2021, 22:09:28 (UTC+05:30)
gaurav2nd
US East (N. Virginia) us-east-1
Bucket and objects not public
October 17, 2021, 22:10:12 (UTC+05:30)
kgaurav
US East (N. Virginia) us-east-1
Objects can be public
September 25, 2021, 11:49:02 (UTC+05:30)
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences

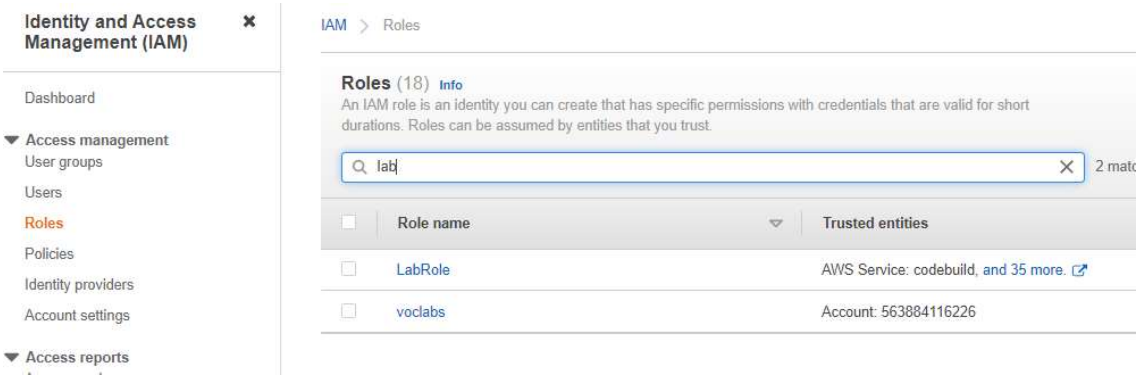

Services
Search for services, features, marketplace products, and docs [Alt+S]
voclabs/user1593585=Kumar_Gaurav @ 6301-3423-2502
N. Virginia
Support
AWS Lambda
Dashboard
Applications
Functions
demo
Additional resources
Code signing configurations
Layers
Related AWS resources
Step Functions state machines
The test event gaurav1 was successfully saved.
Add trigger
Add destination
Function ARN
arn:aws:lambda:us-east-1:630134232502:function:demo
Code
Test
Monitor
Configuration
Aliases
Versions
Code source Info
Upload from
File Edit Find View Go Tools Window
Test
Deploy
Changes not deployed
Go to Anything (Ctrl-P)
lambda_function
Untitled1
Environment
demo
lambda_function.py
import boto3
import csv
def lambda_handler(event, context):
region='us-east-1'
recList=[]
try:
s3=boto3.client('s3')
weather= s3.get_object(Bucket='shreya.weather', Key='delhi-weather-data.csv')
dynamodb = boto3.resource('dynamodb', region_name=region)
table = dynamodb.Table('shreya-weather')
recList = weather['Body'].read().decode('utf-8').split("\n")
firstrecord=True
csv_reader = csv.reader(recList, delimiter=',', quotechar='"')
for row in csv_reader:
if (firstrecord):
firstrecord=False
continue
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences

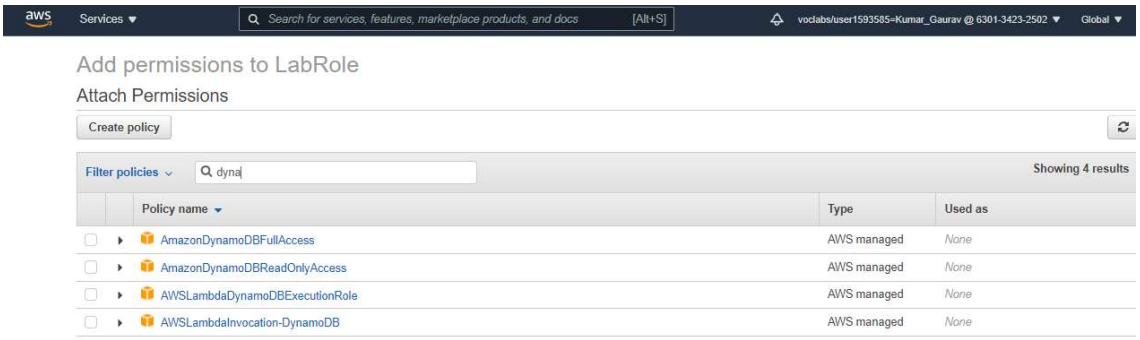


IAM role dashboard

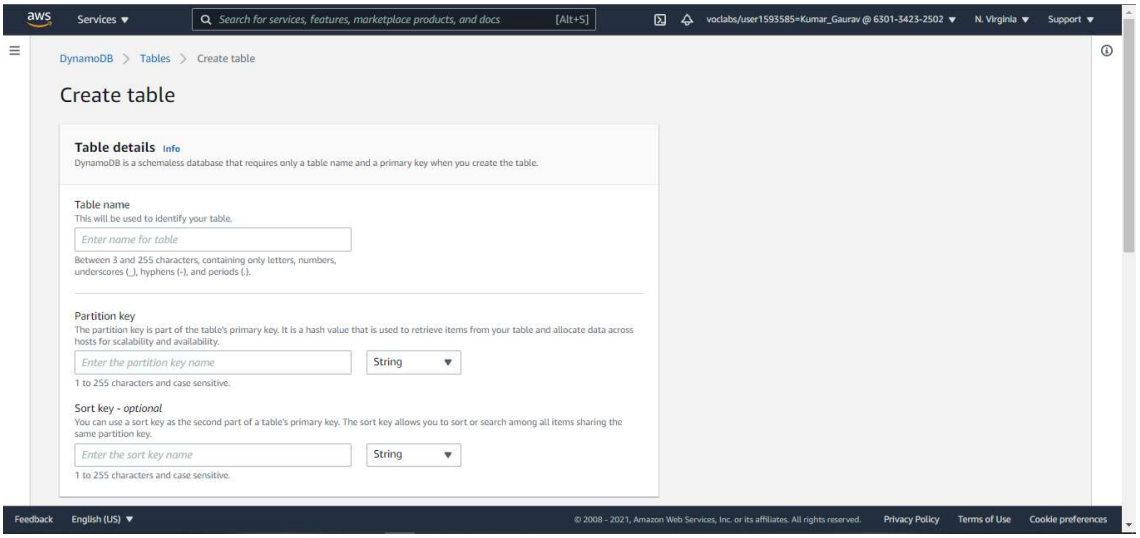


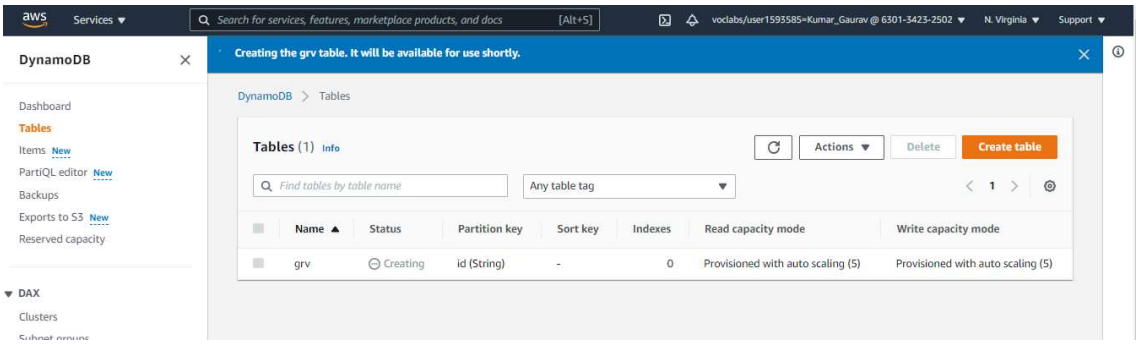




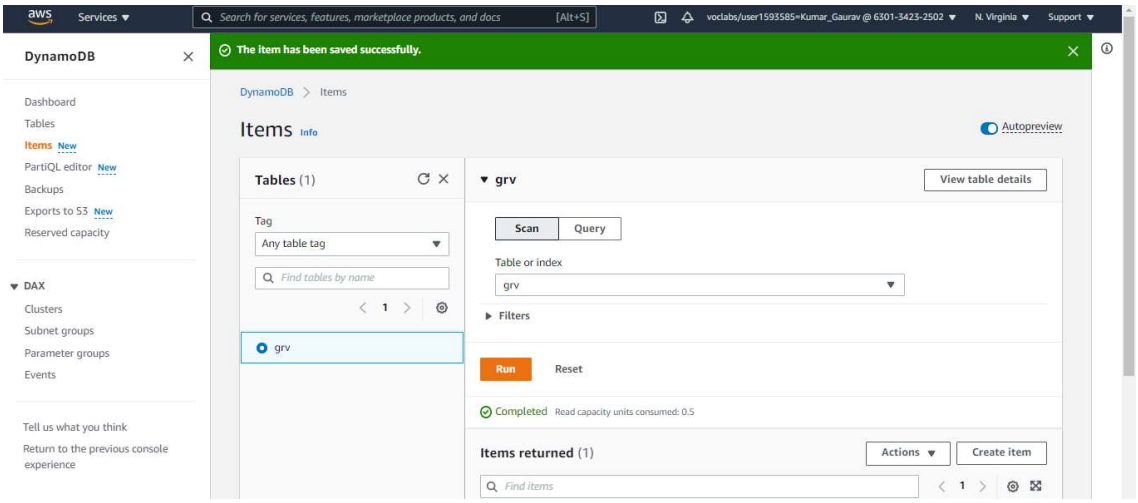


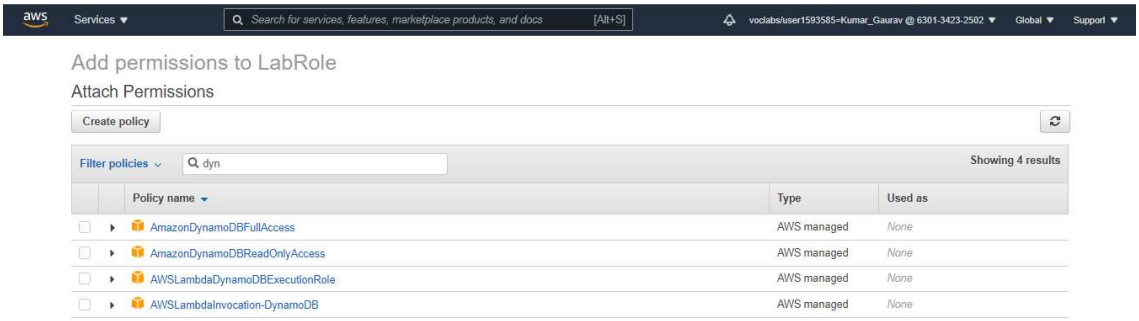
Create table in DynamoDb



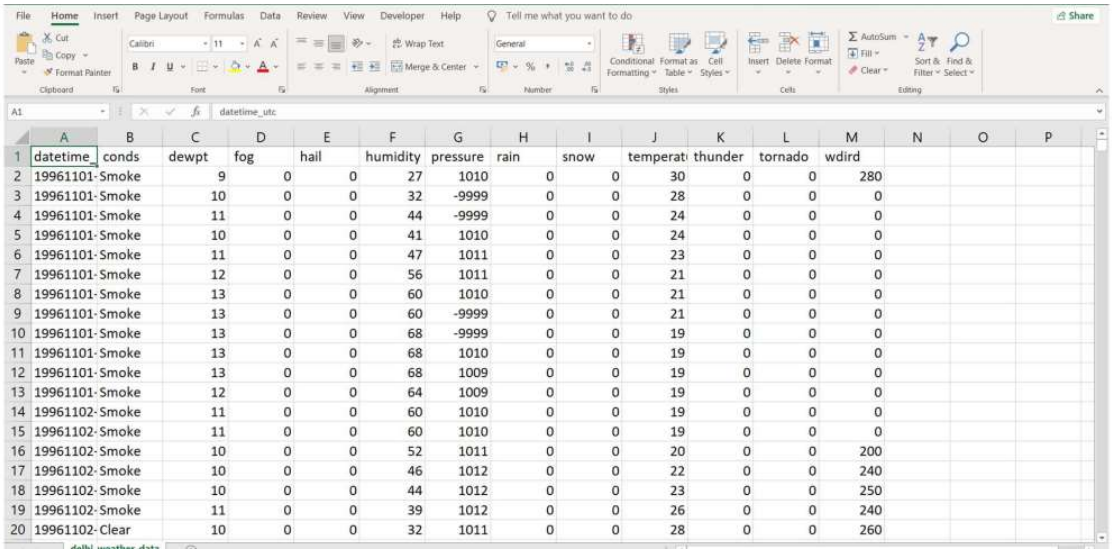


Create DynamoDb

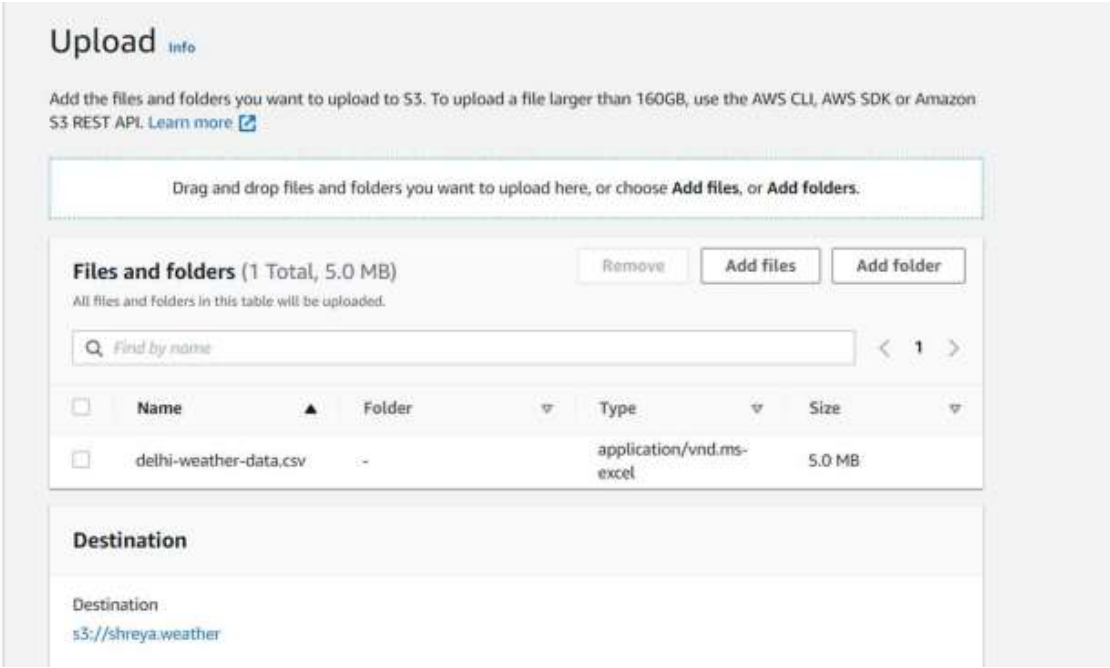




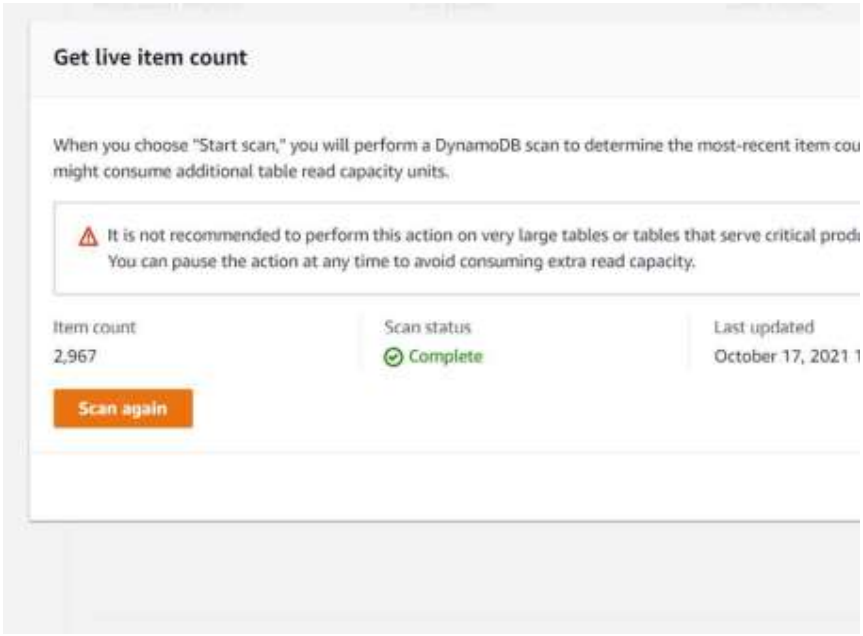
Weather report dataset



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	datetime_	conds	dewpt	fog	hail	humidity	pressure	rain	snow	temperat	thunder	tornado	wdird
2	19961101-	Smoke	9	0	0	27	1010	0	0	30	0	0	280
3	19961101-	Smoke	10	0	0	32	-9999	0	0	28	0	0	0
4	19961101-	Smoke	11	0	0	44	-9999	0	0	24	0	0	0
5	19961101-	Smoke	10	0	0	41	1010	0	0	24	0	0	0
6	19961101-	Smoke	11	0	0	47	1011	0	0	23	0	0	0
7	19961101-	Smoke	12	0	0	56	1011	0	0	21	0	0	0
8	19961101-	Smoke	13	0	0	60	1010	0	0	21	0	0	0
9	19961101-	Smoke	13	0	0	60	-9999	0	0	21	0	0	0
10	19961101-	Smoke	13	0	0	68	-9999	0	0	19	0	0	0
11	19961101-	Smoke	13	0	0	68	1010	0	0	19	0	0	0
12	19961101-	Smoke	13	0	0	68	1009	0	0	19	0	0	0
13	19961101-	Smoke	12	0	0	64	1009	0	0	19	0	0	0
14	19961102-	Smoke	11	0	0	60	1010	0	0	19	0	0	0
15	19961102-	Smoke	11	0	0	60	1010	0	0	19	0	0	0
16	19961102-	Smoke	10	0	0	52	1011	0	0	20	0	0	200
17	19961102-	Smoke	10	0	0	46	1012	0	0	22	0	0	240
18	19961102-	Smoke	10	0	0	44	1012	0	0	23	0	0	250
19	19961102-	Smoke	11	0	0	39	1012	0	0	26	0	0	240
20	19961102-	Clear	10	0	0	32	1011	0	0	28	0	0	260



Records successfully inserted into dynamoDb



Successfully, show top 20 weather report.