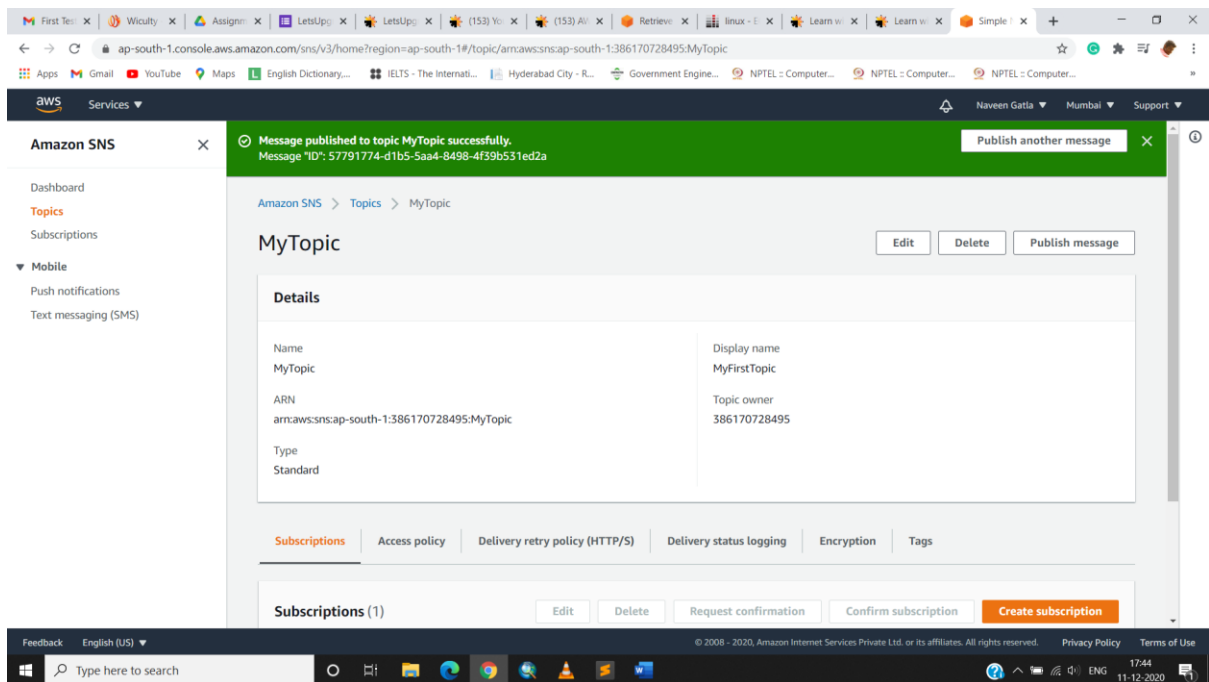


Assignment Day 11 & 12 | 7th November 2020

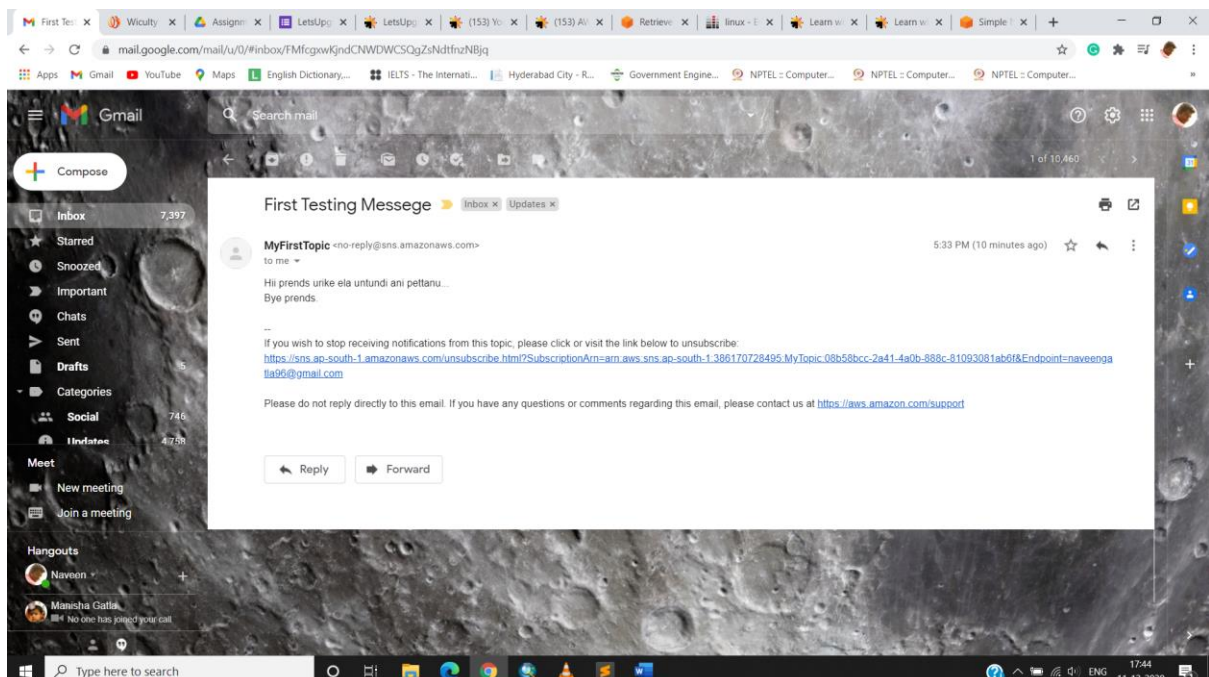
Question 1:

Task 1: Working with SNS

ss1:sns console

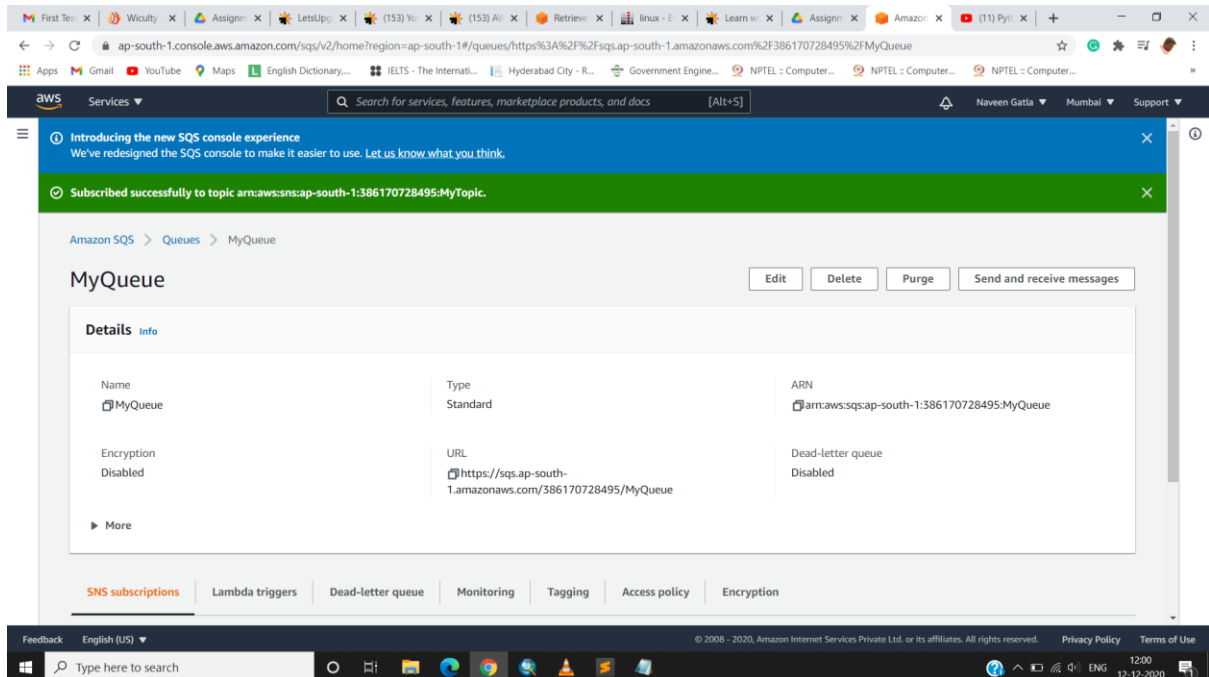


ss2:your inbox with the published message

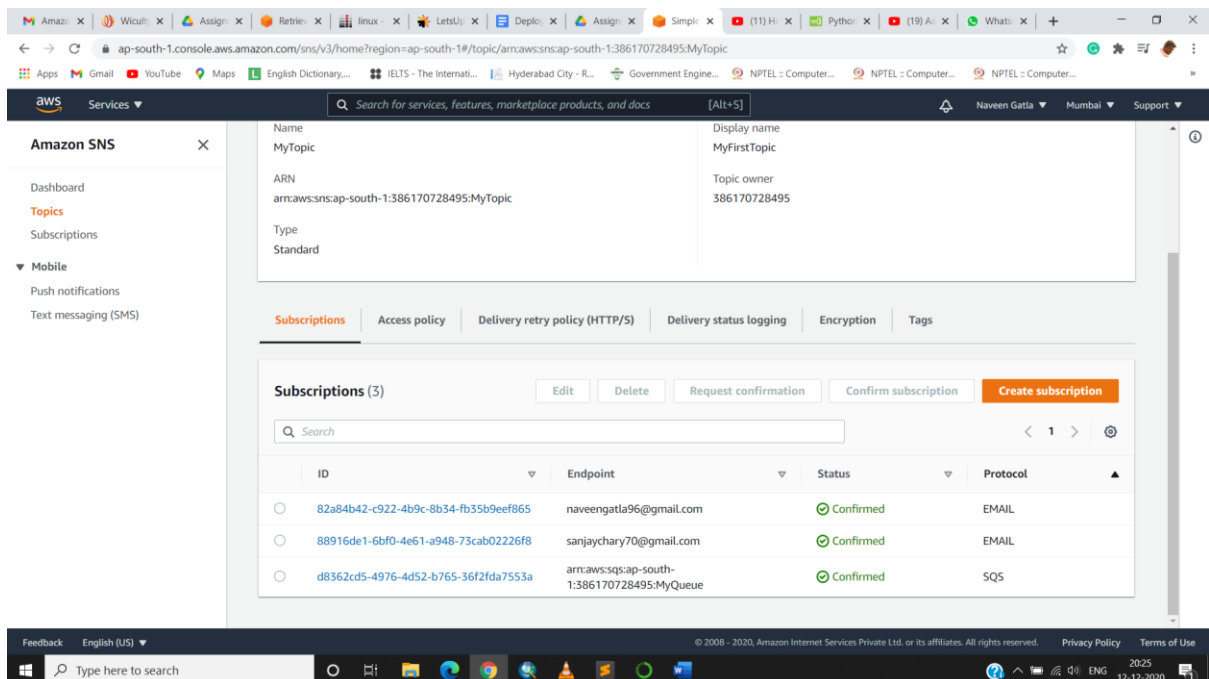


Task 2 :Working with SQS

ss1:sqs console

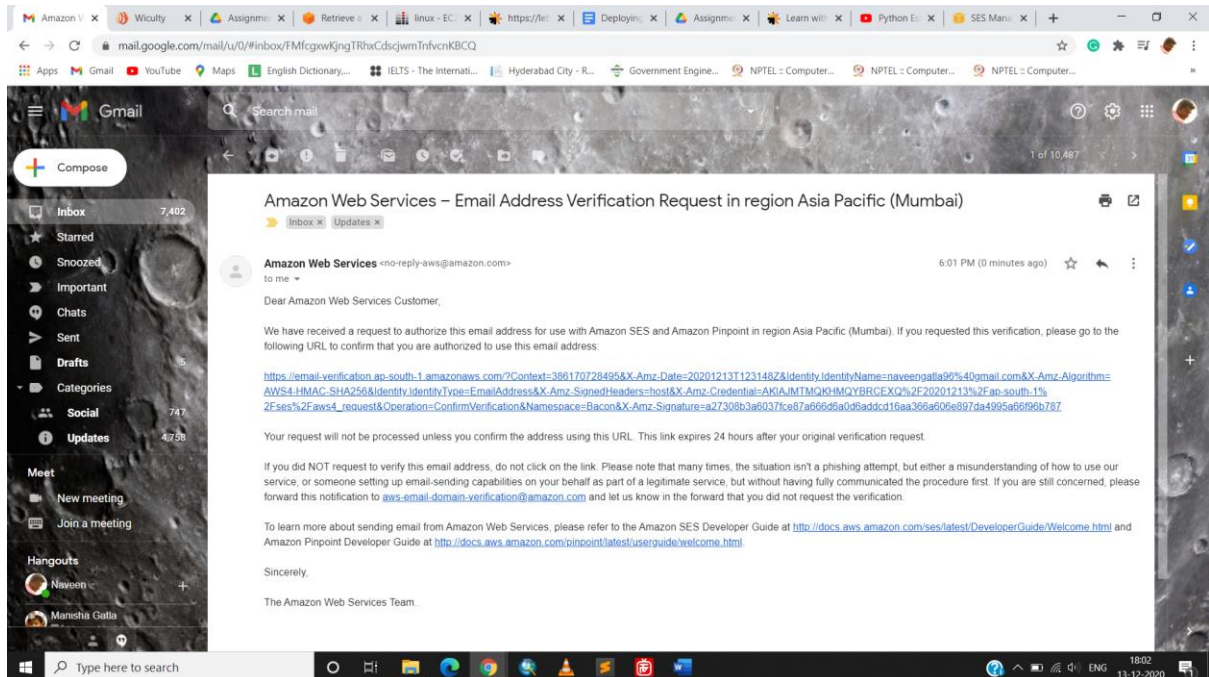


ss2: poll for messages and display message window



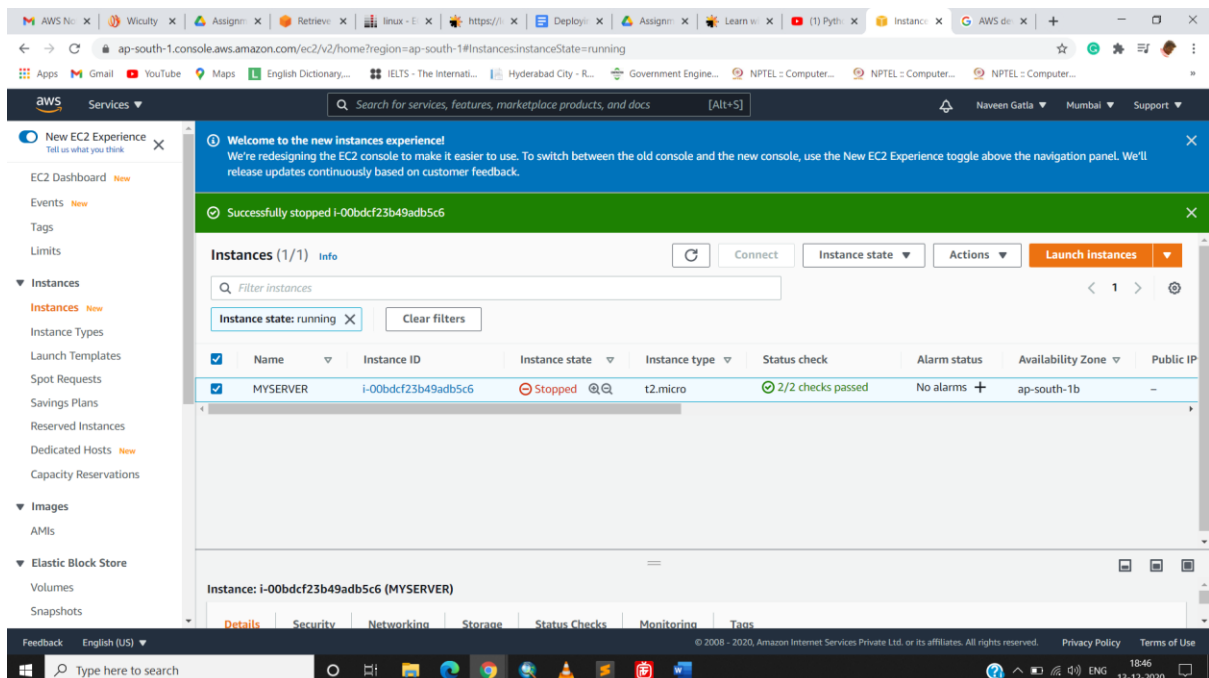
Task 3: Working with SES

ss1: inbox mail verification



Task 4: TRIGGERING CLOUDWATCH EVENT SNS NOTIFICATION

ss1: ec2 console



ss2: rule configuration

The screenshot shows the AWS CloudWatch console with the 'Rules' section selected. The rule 'MyEC2StateNotification' is configured with the following details:

- ARN:** arn:aws:events:ap-south-1:386170728495:rule/MyEC2StateNotification
- Event pattern:**

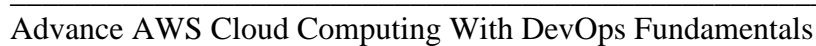
```
{  "source": [    "aws.ec2"  ],  "detail-type": [    "EC2 Instance State-change Notification"  ]}
```
- Status:** Enabled
- Description:** Notification for the change in EC2 state
- Monitoring:** [Show metrics for the rule](#)
- Targets:** One target is configured: SNS topic 'MyEC2Status' with the role 'Matched event'.

ss3: sns topic

The screenshot shows the Amazon SNS console with the 'Topics' section selected. The topic 'MyEC2Status' is configured with the following details:

- Name:** MyEC2Status
- Display name:** For Ec2 Instance State
- ARN:** arn:aws:sns:ap-south-1:386170728495:MyEC2Status
- Topic owner:** 386170728495
- Type:** Standard

Below the details, there are tabs for 'Subscriptions', 'Access policy', 'Delivery retry policy (HTTP/S)', 'Delivery status logging', 'Encryption', and 'Tags'. The 'Subscriptions' tab is active, showing a list of subscriptions (2) with buttons for 'Edit', 'Delete', 'Request confirmation', 'Confirm subscription', and 'Create subscription'.

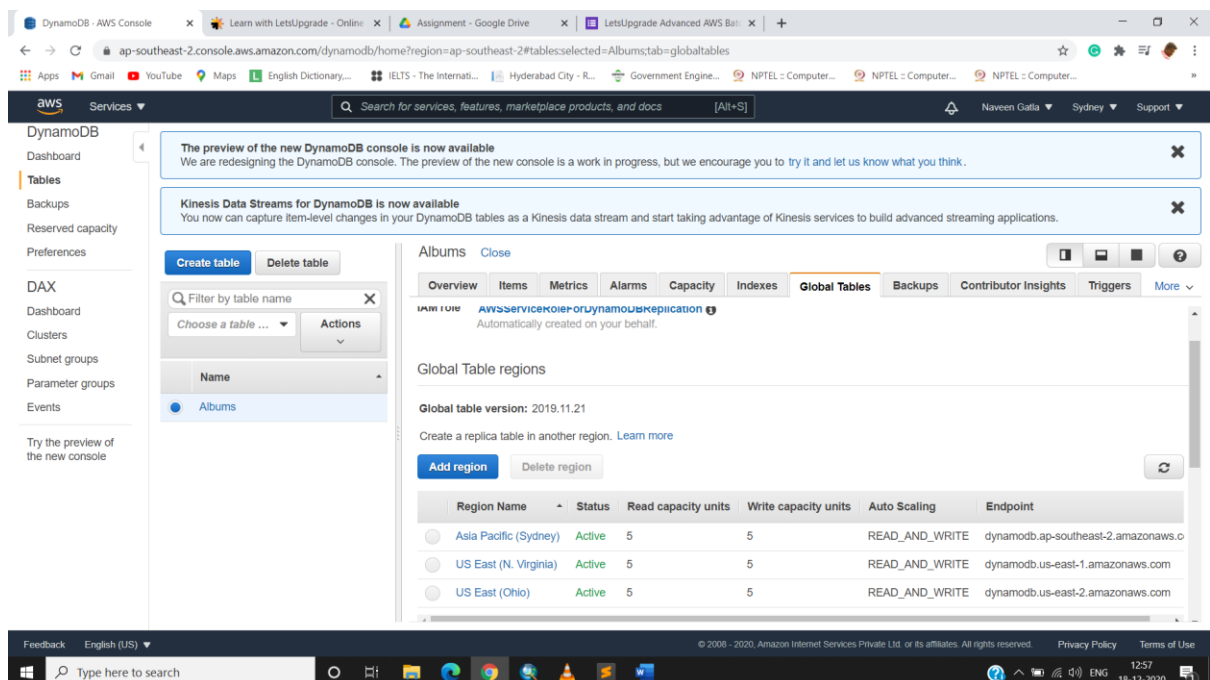
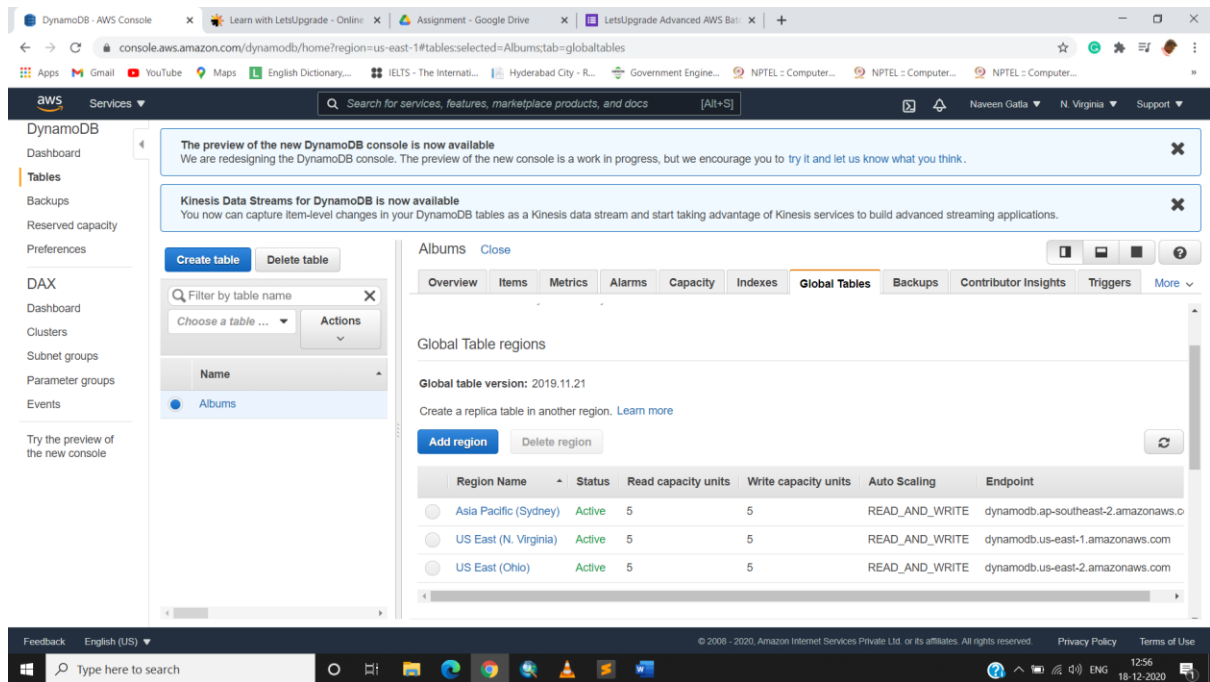


DAY 12 Assignment

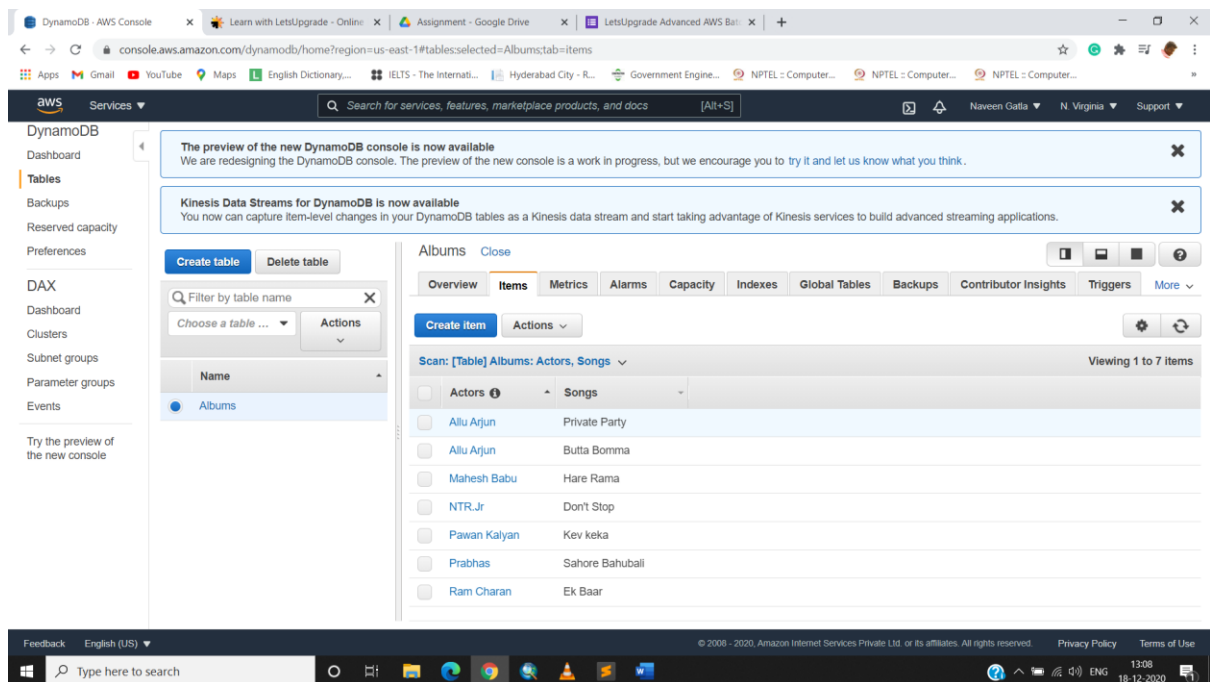
Question 1:

Task 1: Create a dynamo db table with minimum two disaster recovery zones and verify replication.

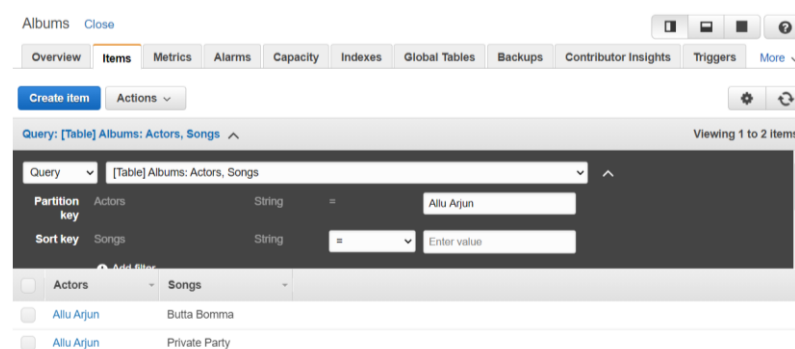
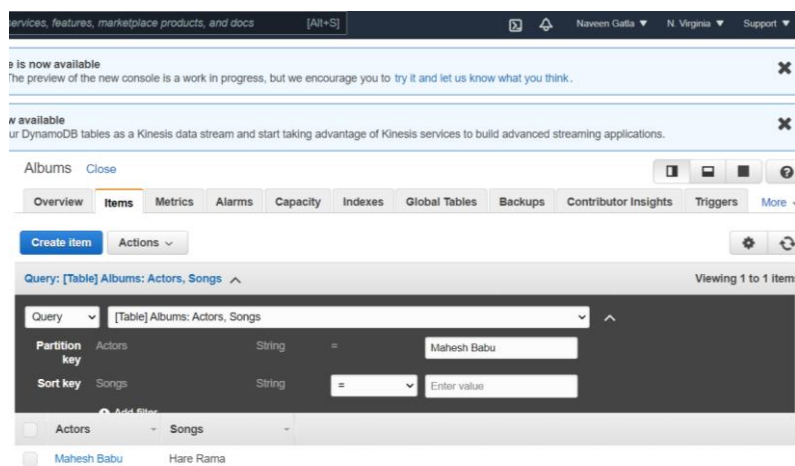
ss1: Disaster recovery regions with the table



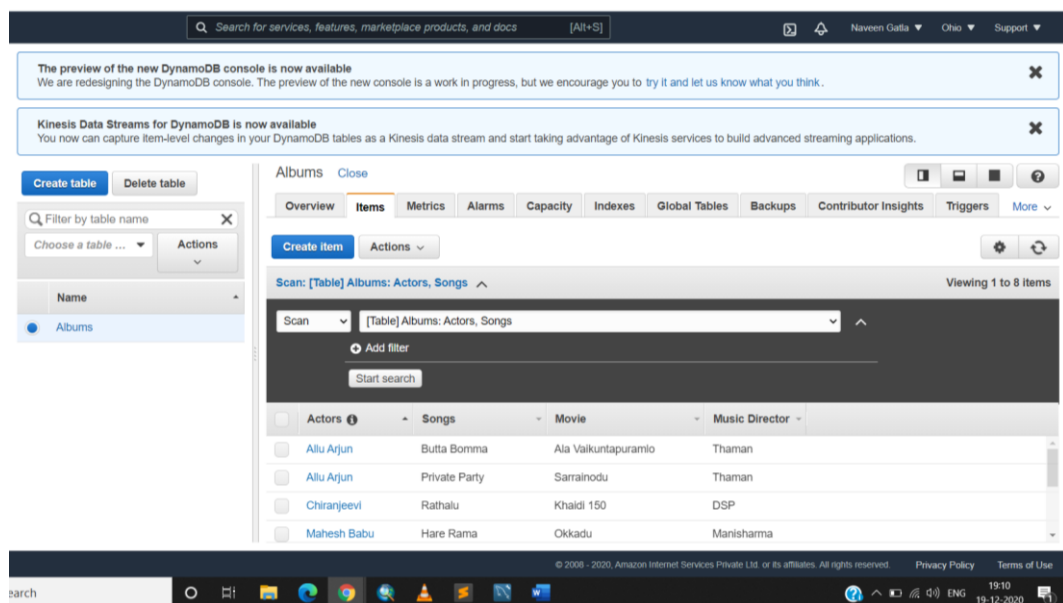
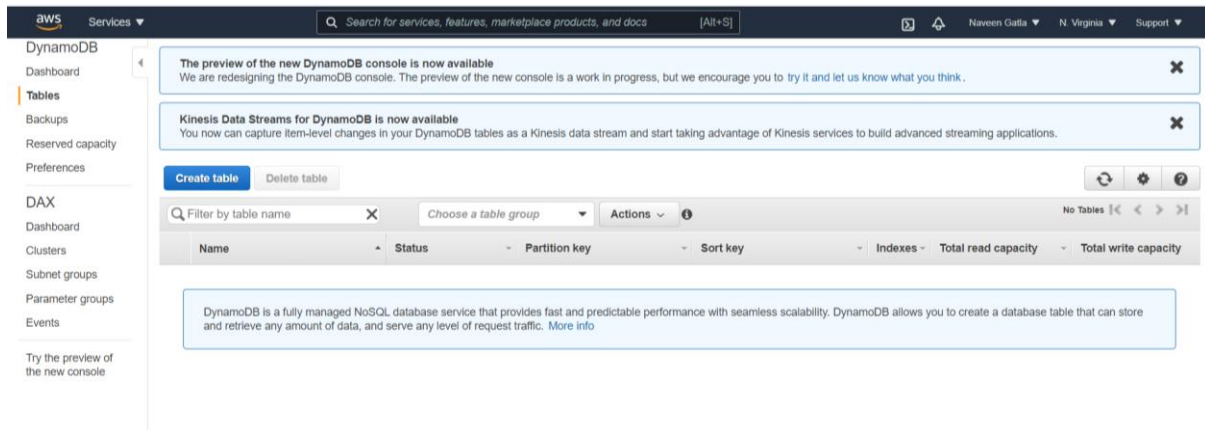
ss2:Home region with all items displayed



ss3:Use query to fetch few items

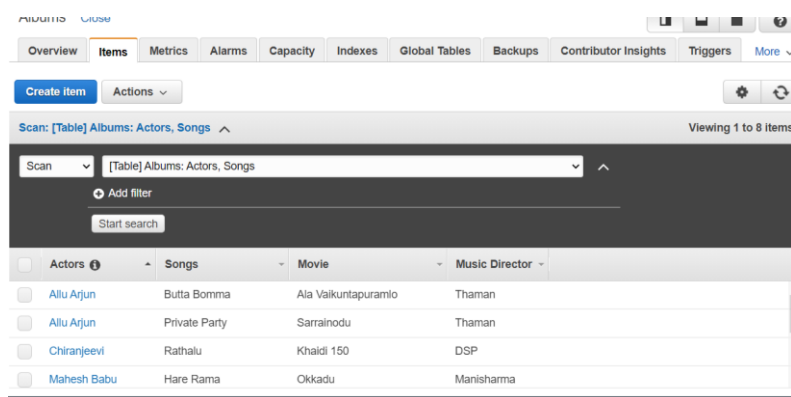


ss4:deletion and verification



Task 2: Creating a dynamo DB table with global secondary indexes and fetching data using global secondary indexes.

ss1: Table with its items displayed



ss2:creating global secondary index

The screenshot shows the AWS DynamoDB console interface. At the top, there's a search bar and navigation links. Below that, there are two informational banners. The main content area shows the 'Albums' table with tabs for Overview, Items, Metrics, Alarms, Capacity, **Indexes**, Global Tables, Backups, Contributor Insights, Triggers, and More. The 'Indexes' tab is active, displaying a table of indexes. A 'Create index' button is visible. The table lists one index: 'Movie-Music-Director-Ind' with status 'Active', type 'GSI', partition key 'Movie (String)', sort key 'Music Director (String)', attributes 'ALL', read capacity '5', and write capacity '5'.

| Name | Status | Type | Partition key | Sort key | Attributes | Read capacity | Write capacity |
|--------------------------|--------|------|----------------|-------------------------|------------|---------------|----------------|
| Movie-Music-Director-Ind | Active | GSI | Movie (String) | Music Director (String) | ALL | 5 | 5 |

ss3:scan with global secondary index

The screenshot shows the AWS DynamoDB console with the 'Items' tab selected for the 'Albums' table. A 'Scan' operation is in progress using the 'Movie-Music-Director-Index'. The scan filter is set to 'Music Director' with a 'String' type and an '=' operator, resulting in 'DSP'. The results are displayed in a table with columns: Actors, Songs, Movie, and Music Director. Two items are shown: one for Chiranjeevi and one for NTR Jr.

| Actors | Songs | Movie | Music Director |
|-------------|------------|-----------------|----------------|
| Chiranjeevi | Rathalu | Khaidi 150 | DSP |
| NTR Jr | Don't Stop | NannakuPrematho | DSP |

Task 3: Deploying a python application in elastic beanstalk

ss1: Application page

The screenshot shows the AWS Elastic Beanstalk console for the application 'MyPythonWebApp'. The left sidebar shows the 'Environments' tab selected. The main content area shows a table of environments for the application.

| Environment name | Health | Date created | Last modified | URL | Running versions | Platform | Platform state | Tier name |
|--------------------|--------|------------------------------|------------------------------|--|--------------------|--|----------------|-----------|
| Mypythonwebapp-env | Ok | 2020-12-18 19:27:07 UTC+0530 | 2020-12-18 19:30:28 UTC+0530 | Mypythonwebapp-env.eba-h8kpidi6.us-east-1.elasticbeanstalk.com | Sample Application | Python 3.7 running on 64bit Amazon Linux 2 | Supported | WebServer |

ss2: env list page

The screenshot shows the AWS Elastic Beanstalk console for the 'All environments' page. The left sidebar shows the 'Environments' tab selected. The main content area shows a table of all environments.

| Environment name | Health | Application name | Date created | Last modified | URL | Running versions | Platform |
|-------------------------------|--------|------------------|------------------------------|------------------------------|--|----------------------|--|
| Mypythonwebapp-env | Ok | MyPythonWebApp | 2020-12-18 19:27:07 UTC+0530 | 2020-12-18 19:30:28 UTC+0530 | Mypythonwebapp-env.eba-h8kpidi6.us-east-1.elasticbeanstalk.com | Sample Application | Python 3.7 running on 64bit Amazon Linux 2 |
| Pythonapp-env (terminated) | - | PythonApp | 2020-12-18 19:18:42 UTC+0530 | 2020-12-18 19:28:29 UTC+0530 | Pythonapp-env.eba-i9accara.us-east-1.elasticbeanstalk.com | Sample Application-1 | Python 3.7 running on 64bit Amazon Linux 2 |
| Pythonwebapp-env (terminated) | - | PythonWebApp | 2020-12-18 19:02:06 UTC+0530 | 2020-12-18 19:13:28 UTC+0530 | Pythonwebapp-env.eba-xaivv5c2.us-east-1.elasticbeanstalk.com | pythonwebapp-source | Python 3.7 running on 64bit Amazon Linux 2 |

ss3:env health status page

The screenshot shows the AWS Elastic Beanstalk console. The left sidebar lists the environment 'Mypythonwebapp-env' under 'Environments'. The main content area displays the health status of this environment. The 'Health' section shows a green checkmark and the status 'Ok'. The 'Running version' section shows 'Sample Application' and a button to 'Upload and deploy'. The 'Platform' section shows the Python logo and 'Python 3.7 running on 64bit Amazon Linux 2/3.1.3'. A notification banner at the top mentions 'Enhanced health authorization'.

ss4:Web page launched using the elastic beanstalk env

The screenshot shows a web browser displaying a 'Congratulations' message. The message states: 'Your first AWS Elastic Beanstalk Python Application is now running on your own dedicated environment in the AWS Cloud. This environment is launched with Elastic Beanstalk Python Platform.' To the right, under 'What's Next?', there are links to 'AWS Elastic Beanstalk overview', 'AWS Elastic Beanstalk concepts', 'Deploy a Django Application to AWS Elastic Beanstalk', 'Deploy a Flask Application to AWS Elastic Beanstalk', 'Customizing and Configuring a Python Container', and 'Working with Logs'. The browser's address bar shows the URL 'pythonapp-env.eba-9accara.us-east-1.elasticbeanstalk.com'.