

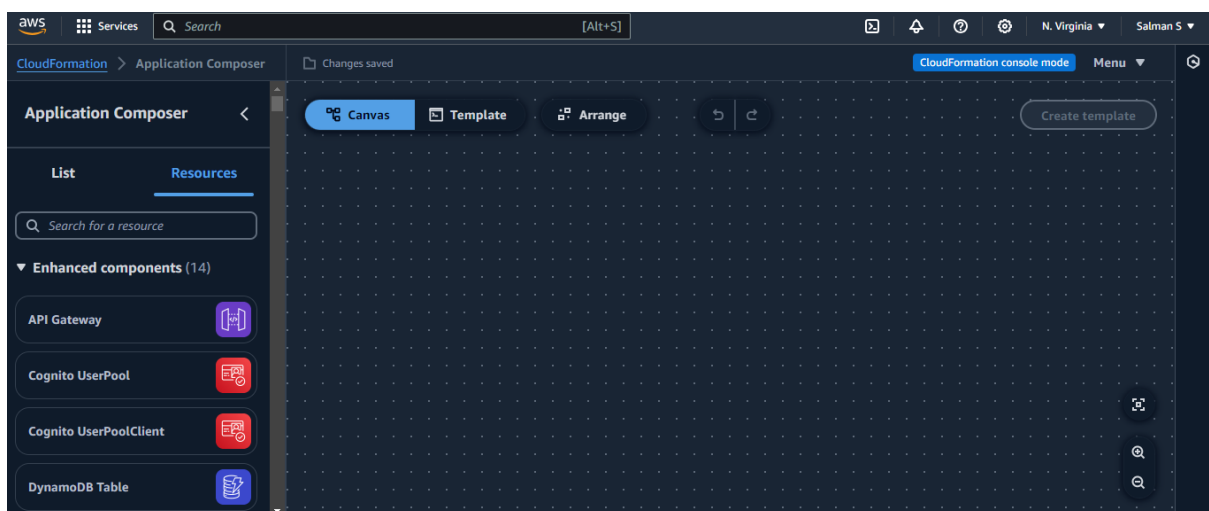
Module 8: CloudFormation Assignment - 1

Problem Statement:

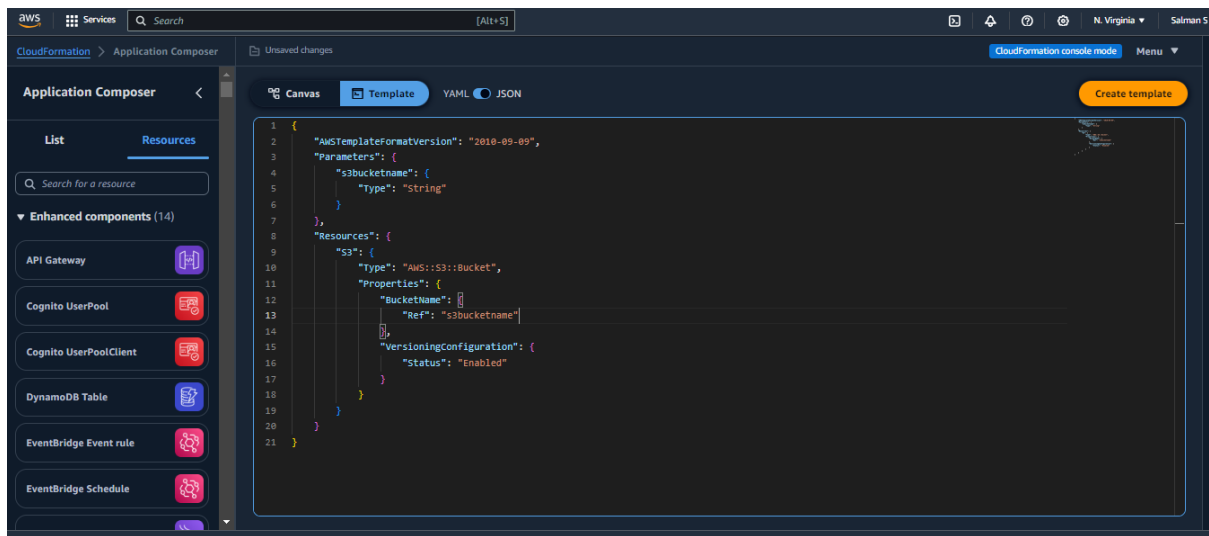
You work for XYZ Corporation. Your team is asked to deploy similar architecture multiple times for testing, development, and production purposes. Implement CloudFormation for the tasks assigned to you below.

Tasks To Be Performed:

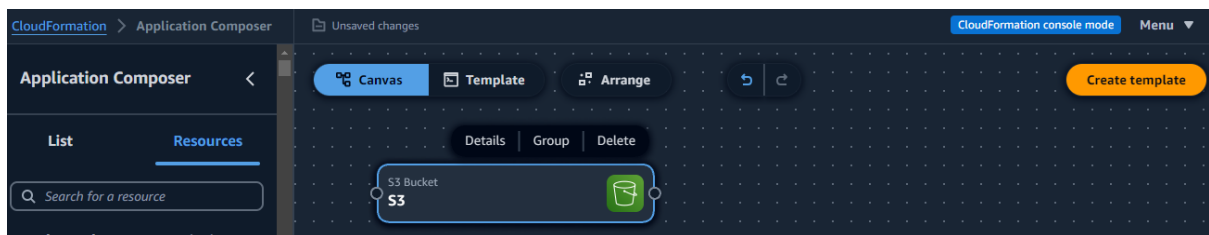
1. Create a template which can create an S3 bucket named "Intellipaat-<yourname >"
 2. The template should be able to enable versioning for the bucket created.
1. Go to **Cloudformation Composer** and **write the code as per the assignment task** and save



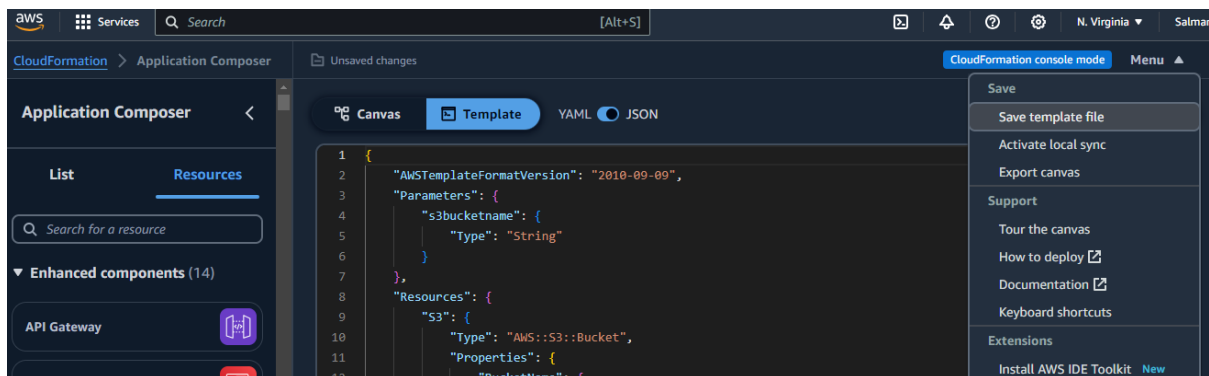
2. **Creating a Parameter** and Set **type as string** and **Created the S3 Resources** and **Enabling Versioning** in Cloudformation



3. Canva vision



4. Go to Template and **Save Template file**



5. Go To **Cloud Stack** and **Choose existing template** and **Upload a Template file** which we Already Created

The screenshot shows the 'Create stack' wizard in AWS CloudFormation. The left sidebar indicates the current step is 'Step 1: Create stack'. The main panel is titled 'Prerequisite - Prepare template'. It contains three radio buttons: 'Choose an existing template' (selected), 'Use a sample template', and 'Build from Application Composer'. Below this, the 'Specify template' section explains that a template is a JSON or YAML file. It offers three options for the template source: 'Amazon S3 URL', 'Upload a template file' (selected), and 'Sync from Git - new'. Under 'Upload a template file', there is a 'Choose file' button and a text input field containing 'Assignment-1.json'.

6. Don't give spaces and we set **parameter as a string** so we can give a **bucketname** here

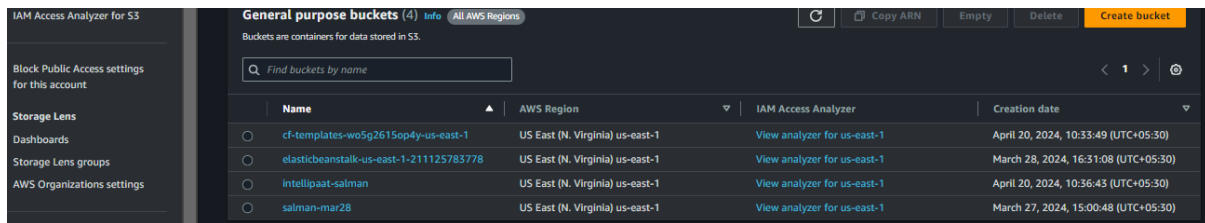
The screenshot shows the 'Specify stack details' step of the 'Create stack' wizard. The left sidebar indicates the current step is 'Step 2: Specify stack details'. The main panel is titled 'Specify stack details'. It has two main sections: 'Provide a stack name' and 'Parameters'. In the 'Provide a stack name' section, the 'Stack name' input field contains 'salmans-first-stack'. Below it, a red warning message states: 'Stack name must contain only letters, numbers, dashes. Must start with a letter. Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-)'. In the 'Parameters' section, the 's3bucketname' parameter is defined, and its value is set to 'intellipaat-salman'. At the bottom right, there are 'Cancel', 'Previous', and 'Next' buttons.

7. Now the Stacks **Created Successfully**

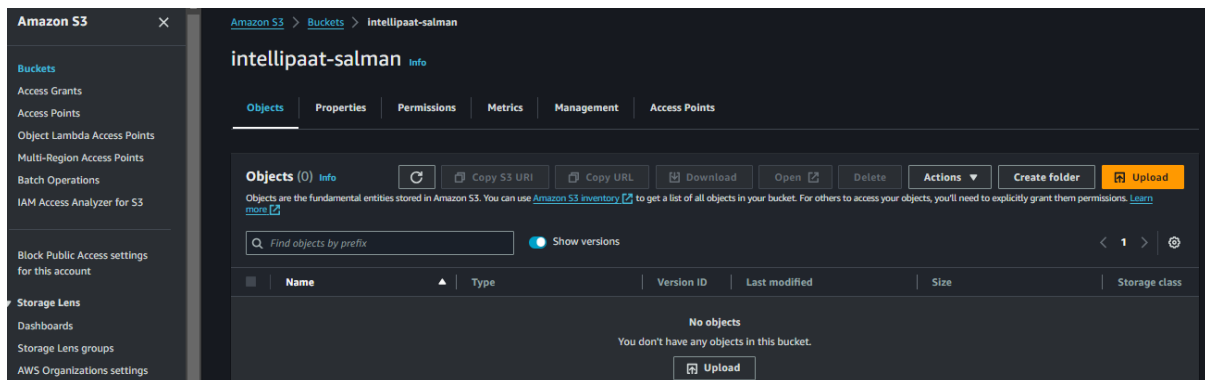
The screenshot shows the AWS CloudFormation console. On the left, the 'Stacks (1)' list shows the 'salmans-first-stack' with a status of 'CREATE_COMPLETE'. On the right, the 'Events' tab for the 'salmans-first-stack' is selected, showing a list of events. The events table has columns for 'Timestamp', 'Logical ID', 'Status', and 'Details'. The events show the stack creation process, including the creation of the stack itself and the creation of S3 buckets for the template and outputs.

Timestamp	Logical ID	Status	Details
2024-04-20 10:37:04 UTC+0530	salmans-first-stack	CREATE_COMPLETE	-
2024-04-20 10:37:03 UTC+0530	S3	CREATE_COMPLETE	-
2024-04-20 10:36:41 UTC+0530	S3	CREATE_IN_PROGRESS	-
2024-04-20 10:36:39 UTC+0530	S3	CREATE_IN_PROGRESS	-
2024-04-20 10:36:36 UTC+0530	salmans-first-stack	CREATE_IN_PROGRESS	-

8. We can able to see the **intellipaat-salman** bucket



9. We also see the **versioning** and this is Assignment task which we **completed**



Module 8: CloudFormation Assignment - 2

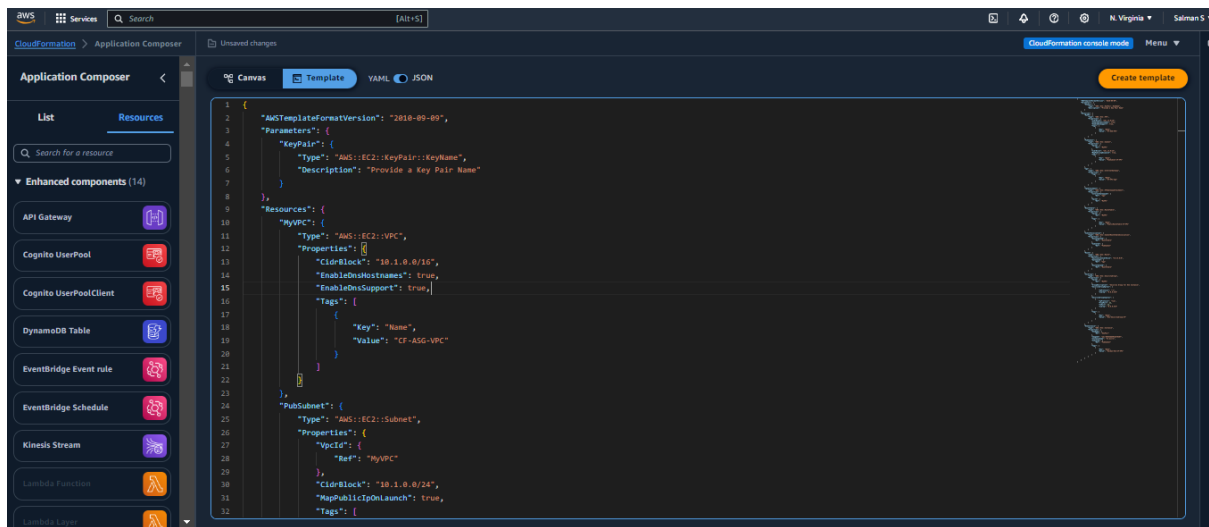
Problem Statement:

You work for XYZ Corporation. Your team is asked to deploy similar architecture multiple times for testing, development, and production purposes. Implement CloudFormation for the tasks assigned to you below.

Tasks To Be Performed:

1. Create a template with 1 VPC and 1 public subnet.
2. Launch an Amazon Linux EC2 instance in the public subnet and tag the instance as "CFInstance"

1. **Develop the Code** as per the Assignment in Cloudformation Created **parameters,Resources**



2. Code of 2nd Assignment and Creating a **public sub and IGW, IGW Attachment to VPC** Creating **RT**

```
33     {
34         "Key": "Name",
35         "Value": "PubSubnet-CF-VPC"
36     }
37 }
38 },
39 ],
40 "Igw": {
41     "Type": "AWS::EC2::InternetGateway",
42     "Properties": {
43         "Tags": [
44             {
45                 "Key": "Name",
46                 "Value": "CF-VPC-Igw"
47             }
48         ]
49     }
50 },
51 "IgwAttachment": {
52     "Type": "AWS::EC2::VPCGatewayAttachment",
53     "Properties": {
54         "InternetGatewayId": {
55             "Ref": "Igw"
56         },
57         "VpcId": {
58             "Ref": "MyVPC"
59         }
60     }
61 },
62 "RouteTable": {
63     "Type": "AWS::EC2::RouteTable",
```

3. Code for RT and Adding Tags and Value giving name and Creating Subnetasso through code in Cloudformation and Route

```
63     "Type": "AWS::EC2::RouteTable",
64     "Properties": {
65         "VpcId": {
66             "Ref": "MyVPC"
67         },
68         "Tags": [
69             {
70                 "Key": "Name",
71                 "Value": "PublicRouteTable-CF-VPC"
72             }
73         ]
74     },
75 },
76 "SubnetAssociation": {
77     "Type": "AWS::EC2::SubnetRouteTableAssociation",
78     "Properties": {
79         "RouteTableId": {
80             "Ref": "RouteTable"
81         },
82         "SubnetId": {
83             "Ref": "PubSubnet"
84         }
85     }
86 },
87 "Route": {
88     "Type": "AWS::EC2::Route",
89     "Properties": {
90         "DestinationCidrBlock": "0.0.0.0/0",
91         "GatewayId": {
92             "Ref": "Igw"
93         },
94         "RouteTableId": {
```

4. SecGroup, SGEgress and SGIingress

```
94     "RouteTableId": {
95         "Ref": "RouteTable"
96     }
97 }
98 },
99 "SecGroup": {
100     "Type": "AWS::EC2::SecurityGroup",
101     "Properties": {
102         "VpcId": {
103             "Ref": "MyVPC"
104         },
105         "GroupDescription": "Security Group for EC2 Instance",
106         "SecurityGroupEgress": [
107             {
108                 "IpProtocol": "-1",
109                 "CidrIp": "0.0.0.0/0"
110             }
111         ],
112         "SecurityGroupIngress": [
113             {
114                 "IpProtocol": "tcp",
115                 "FromPort": 22,
116                 "ToPort": 22,
117                 "CidrIp": "0.0.0.0/0"
118             }
119         ],
120         "Tags": [
121             {
122                 "Key": "Name",
123                 "Value": "EC2-SecurityGroup-CF"
124             }
125         ]
126     }
127 }
```

5. Finally Creating EC2 Instances and ref to pub subnet and Giving Tag I mean Name for EC2 Instances

```
124     },
125     ],
126   },
127   },
128   "EC2Instance": {
129     "Type": "AWS::EC2::Instance",
130     "Properties": {
131       "KeyName": {
132         "Ref": "KeyPair"
133       },
134       "ImageId": "ami-080e1f13689e07408",
135       "InstanceType": "t2.micro",
136       "SubnetId": {
137         "Ref": "PubSubnet"
138       },
139       "Tags": [
140         {
141           "Key": "Name",
142           "Value": "Salman-EC2-CF-VPC"
143         }
144       ]
145     }
146   }
147 }
```

6. Canva Vision(HLD)



7. Creating Stack and existing template and Upload it

The screenshot shows the 'Create stack' wizard in the AWS CloudFormation console. The 'Specify template' step is active, showing options to choose an existing template, use a sample template, or build from Application Composer. The 'Upload a template file' option is selected, and a file named 'Assignment-2.json' is shown as uploaded.

CloudFormation > Stacks > Create stack

Step 1: **Create stack**

Step 2: Specify stack details

Step 3: Configure stack options

Step 4: Review and create

Create stack

Prerequisite - Prepare template

Prepare template
Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you want to include in the stack.

☒ **Choose an existing template**
Upload or choose an existing template.

☐ **Use a sample template**
Choose from our sample template library.

☐ **Build from Application Composer**
Create a template using a visual builder.

Specify template

A template is a JSON or YAML file that describes your stack's resources and properties.

Template source
Selecting a template generates an Amazon S3 URL where it will be stored.

☐ **Amazon S3 URL**
Provide an Amazon S3 URL to your template.

☒ **Upload a template file**
Upload your template directly to the console.

☐ **Sync from Git - new**
Sync a template from your Git repository.

Upload a template file

Assignment-2.json
JSON or YAML formatted file

Shell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy

8. Stack Name and select Key Pair and Next

The screenshot shows the 'Specify stack details' step of the 'Create stack' wizard. The 'Stack name' field is filled with 'salmons-Second-stack'. The 'Parameters' section shows a 'KeyPair' parameter with the value 'Salman' selected from a dropdown menu.

CloudFormation > Stacks > Create stack

Step 1: [Create stack](#)

Step 2: **Specify stack details**

Step 3: Configure stack options

Step 4: Review and create

Specify stack details

Provide a stack name

Stack name

salmons-Second-stack

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

KeyPair
Provide a Key Pair Name

Salman

9. Stack Created Successfully EC2, IGW, IGWAttachment, MYVPC, PubSubnet, Route, RouteTable, SecGroup, SubnetAssociation

The screenshot shows the 'salmons-Second-stack' stack in the AWS CloudFormation console. The stack is in the 'CREATE_COMPLETE' state. The 'Resources' section lists 9 resources: EC2Instance, Igw, IgwAttachment, MyVPC, PubSubnet, Route, RouteTable, SecGroup, and SubnetAssociation, all of which are in the 'CREATE_COMPLETE' state.

AWS CloudFormation > Stacks > salmons-Second-stack

Stacks (1)

Filter by stack name:

Filter status: **Active** ☒ View nested

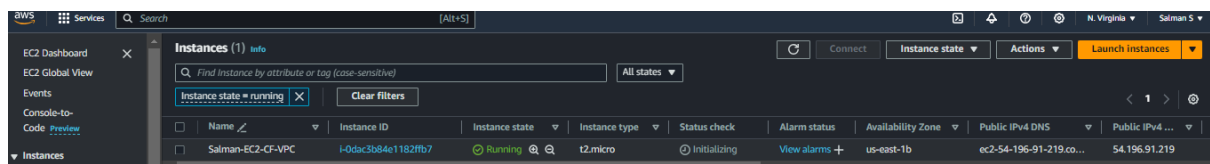
Stacks

salmons-Second-stack
2024-04-20 11:39:20 UTC+0530
CREATE_COMPLETE

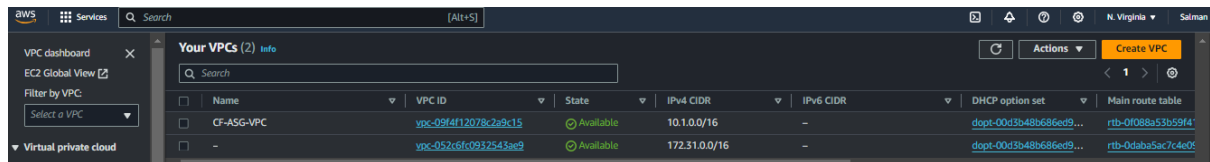
Resources (9)

Logical ID	Physical ID	Type	Status	Module
EC2Instance	i-0d4c3b84e1182ff57	AWS::EC2::Instance	CREATE_COMPLETE	-
Igw	igw-0d8a2e0f9346c95bd	AWS::EC2::InternetGateway	CREATE_COMPLETE	-
IgwAttachment	IGW/vpc-09f4f12078c2a9c15	AWS::EC2::VPCGatewayAttachment	CREATE_COMPLETE	-
MyVPC	vpc-09f4f12078c2a9c15	AWS::EC2::VPC	CREATE_COMPLETE	-
PubSubnet	subnet-09525a2af0faa654	AWS::EC2::Subnet	CREATE_COMPLETE	-
Route	rtb-0481b4fc74d264d070.0.0/0	AWS::EC2::Route	CREATE_COMPLETE	-
RouteTable	rtb-0481b4fc74d264d07	AWS::EC2::RouteTable	CREATE_COMPLETE	-
SecGroup	sg-00bb1eed5ed8b9872	AWS::EC2::SecurityGroup	CREATE_COMPLETE	-
SubnetAssociation	rtbassoc-05d251a8bf9e9d49	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE	-

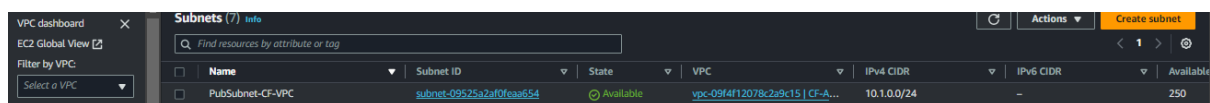
10. Successfully Created EC2 in Console



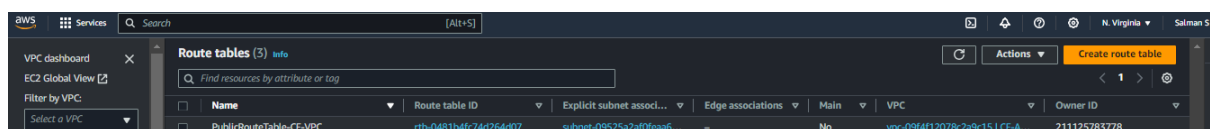
11. Successfully Created VPC in Console



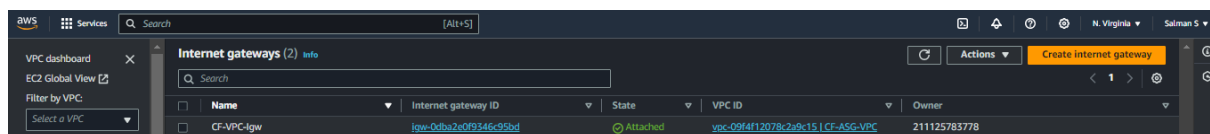
12. Successfully Created Public Subnet in Console



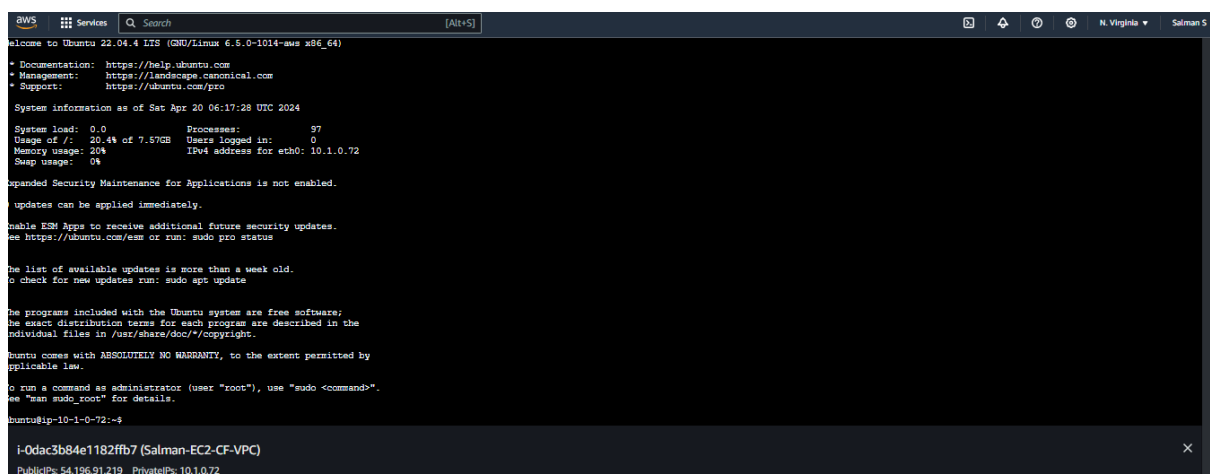
13. Successfully Created Public Route Table in Console



14. Successfully Created VPC-IGW in Console



15. Launch Instances Successfully and Once we delete the Stack All Resources will be deleted Automatically



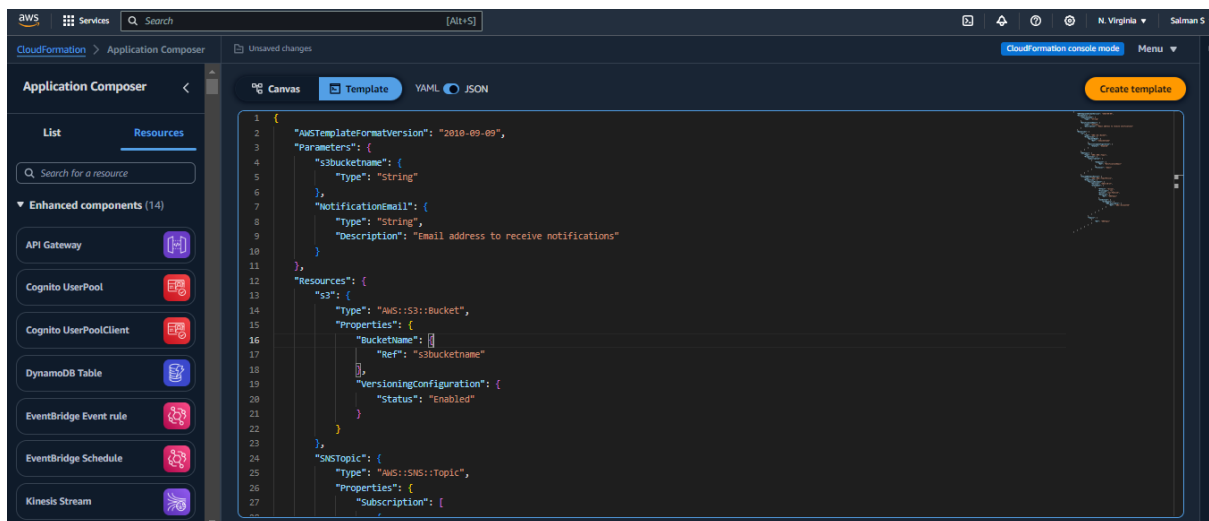
Module 8: CloudFormation Assignment - 3

Problem Statement:

You work for XYZ Corporation. Your team is asked to deploy similar architecture multiple times for testing, development, and production purposes. Implement CloudFormation for the tasks assigned to you below.

Tasks To Be Performed:

1. Use the template from CloudFormation task 1.
 2. Add Notification to the CloudFormation stack using SNS so that you get a notification via mail for every step of the stack creation process.
1. Using Cloudformation Task 1 and Template and Add Notification to the Cloudformation Stack using SNS you get notification via mail for every step of the stack creation process and **Creating SNSTopic** and Email Subscription



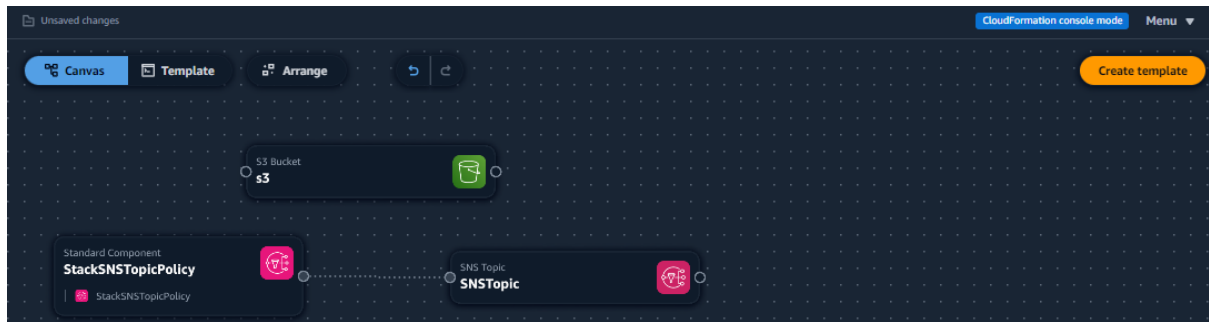
2. In the Subscription The Endpoint is to **NotificationEmail and Protocol Email** and Adding **StackSNSTopicPolicy, Properties**, PolicyDocument, Version, Statement, Effect and Principal, Action, Resource is Res SNSTopic and Condition and Ref AWS AccountID

```
27     "Subscription": [  
28         {  
29             "Endpoint": {  
30                 "Ref": "NotificationEmail"  
31             },  
32             "Protocol": "email"  
33         }  
34     ],  
35 },  
36 },  
37 "StackSNSTopicPolicy": {  
38     "Type": "AWS::SNS::TopicPolicy",  
39     "Properties": {  
40         "PolicyDocument": {  
41             "Version": "2012-10-17",  
42             "Statement": [  
43                 {  
44                     "Effect": "Allow",  
45                     "Principal": "*",  
46                     "Action": "sns:Publish",  
47                     "Resource": {  
48                         "Ref": "SNSTopic"  
49                     },  
50                     "Condition": {  
51                         "ArnEquals": {  
52                             "AWS:SourceArn": {  
53                                 "Ref": "AWS::AccountId"
```

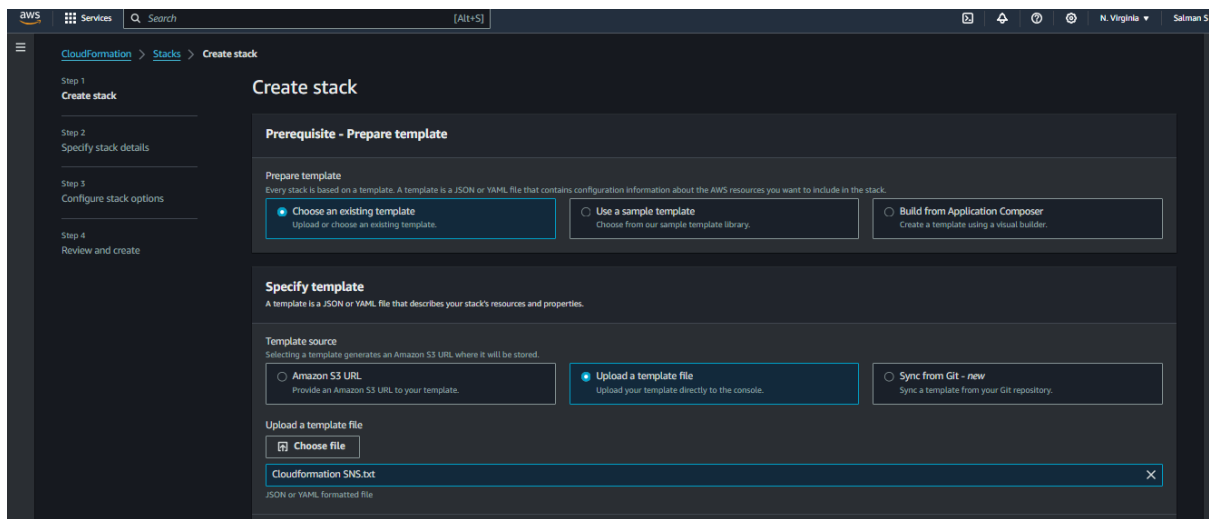
3. Topics and **Ref to SNSTopic**

```
53         "Ref": "AWS::AccountId"  
54     },  
55     "Topics": [  
56         {  
57             "Ref": "SNSTopic"  
58         }  
59     ],  
60 },  
61 },  
62 },  
63 },  
64 },  
65 },  
66 },  
67 },  
68 }
```

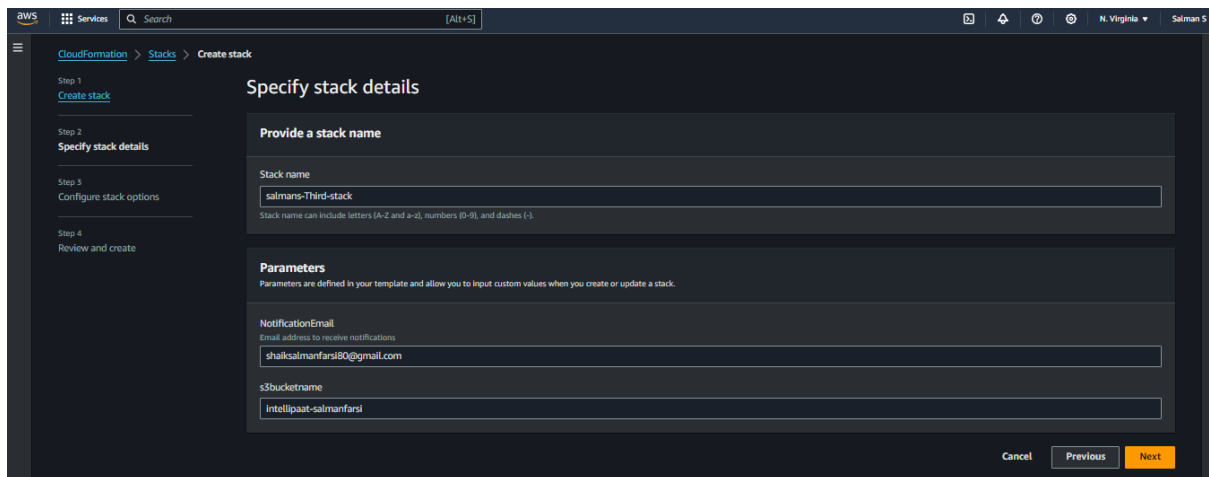
4. Canva Vision



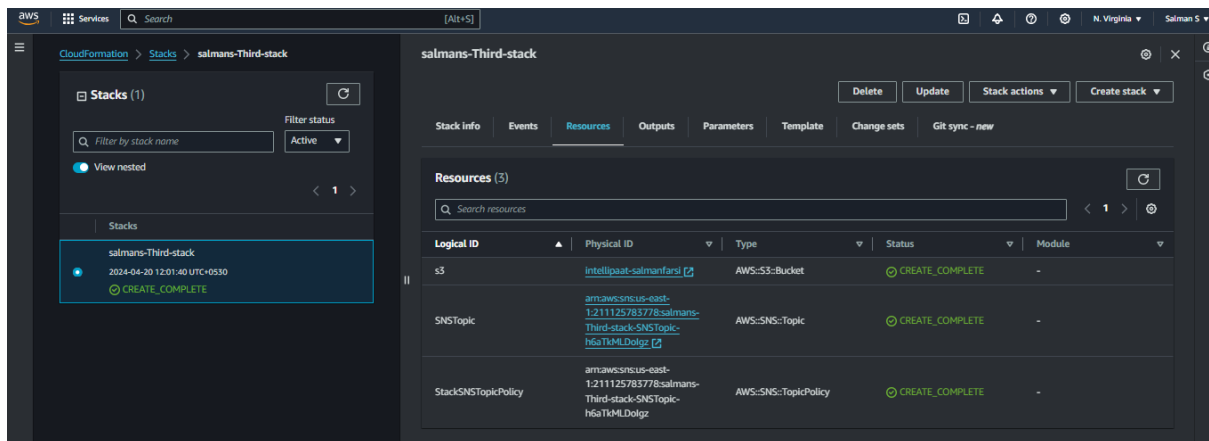
5. Create Stack and Existing one and Upload it



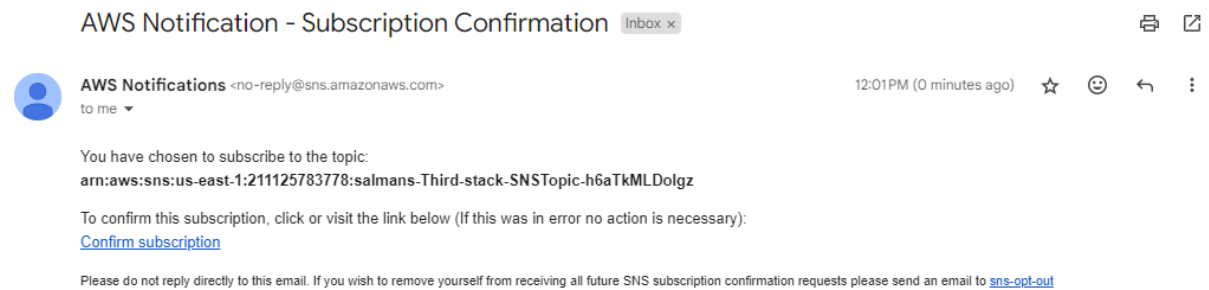
6. Giving Email and Bucket Name



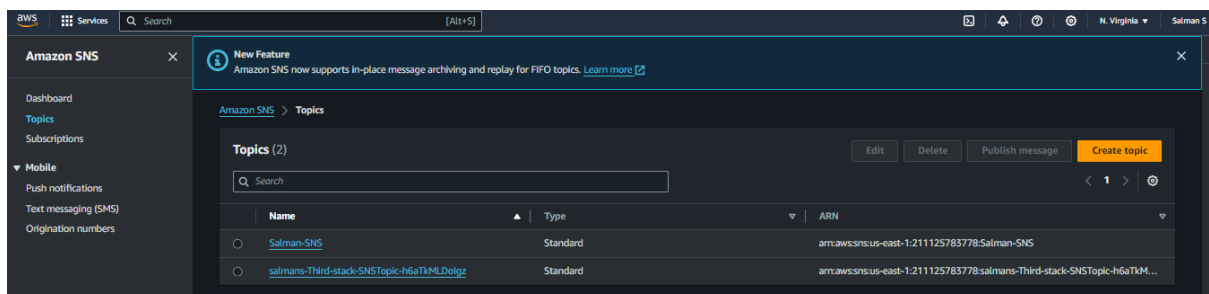
7. Stack Created Successfully and Resources too



8. I got a Subscription Mail From AWS Notification



9. Topics Created for Every Step of Stack Creation and Now the Assignment is Completed



Module 8: SQS and SES Assignment

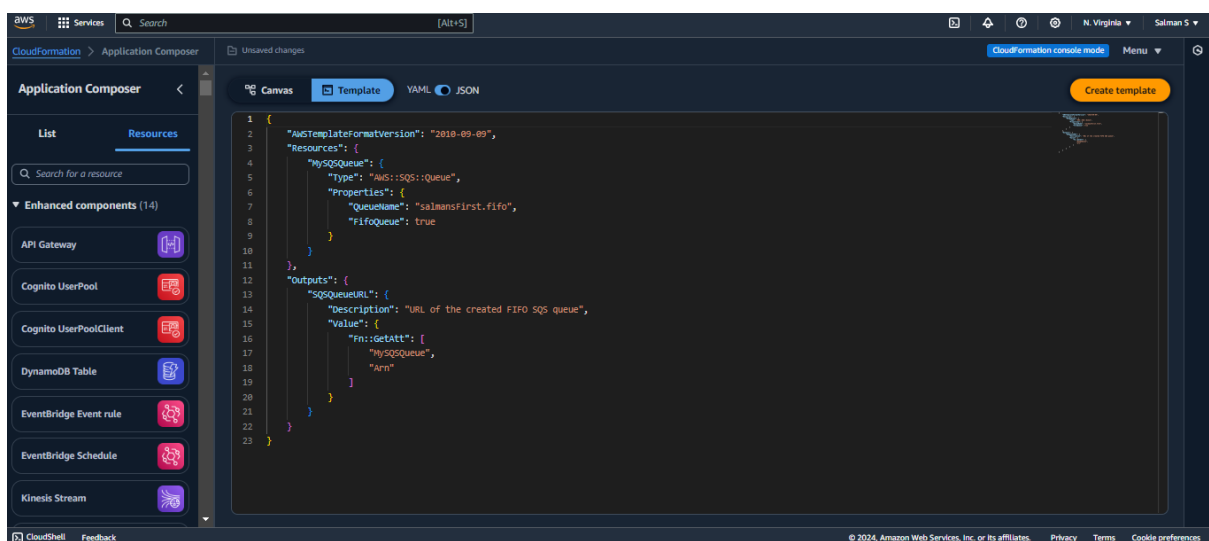
Problem Statement:

You work for XYZ Corporation. Your team is asked to deploy similar architecture multiple times for testing, development, and production purposes. Implement CloudFormation for the tasks assigned to you below.

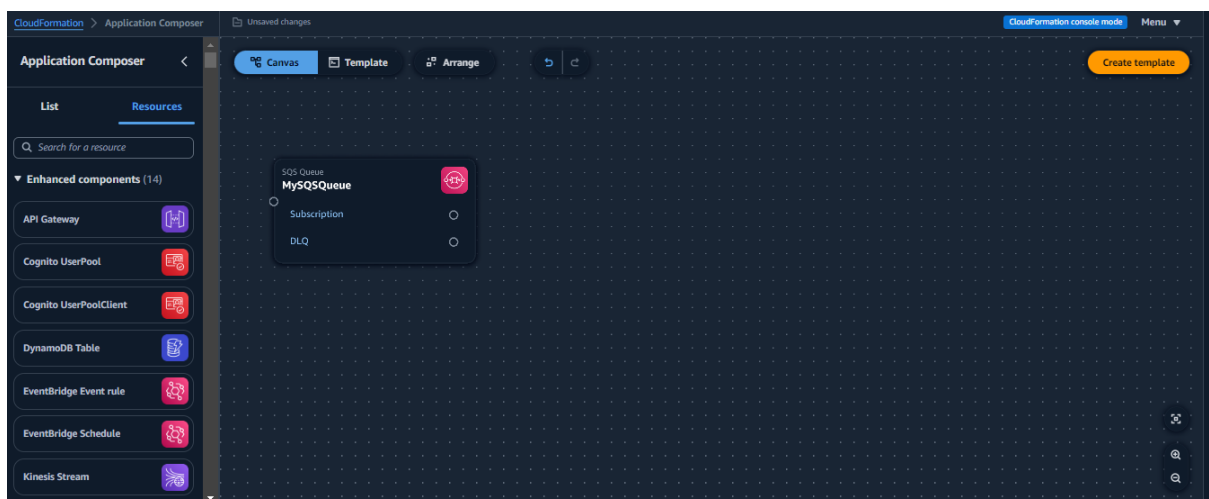
Tasks To Be Performed:

1. Create a FIFO SQS queue and test by sending messages.
2. Register your mail in SES and send a test mail to yourself.

1. Creating FIFO SQS Queue only through json code and AWS::SES::Identity is showing Unrecognized



2. Canva Vison(HLD)



3. **Create Fourth Stack** Means 4th Assignment and **Choose existing** one and upload it and next

Create stack

Step 1: Create stack

Step 2: Specify stack details

Step 3: Configure stack options

Step 4: Review and create

Prerequisite - Prepare template

Prepare template
Every stack is based on a template. A template is a .JSON or .YAML file that contains configuration information about the AWS resources you want to include in the stack.

☒ Choose an existing template
Upload or choose an existing template.

☐ Use a sample template
Choose from our sample template library.

☐ Build from Application Composer
Create a template using a visual builder.

Specify template
A template is a .JSON or .YAML file that describes your stack's resources and properties.

Template source
Selecting a template generates an Amazon S3 URL where it will be stored.

☐ Amazon S3 URL
Provide an Amazon S3 URL to your template.

☒ Upload a template file
Upload your template directly to the console.

☐ Sync from Git - new
Sync a template from your Git repository.

Upload a template file

Assignment-4.json

JSON or YAML formatted file

S3 URL: <https://s3.us-east-1.amazonaws.com/cf-templates-wo5g2615op4y-us-east-1/2024-04-20T083524.292Ztv5-Assignment-4.json>

4. **Stack Name** and Next

Specify stack details

Step 1: Create stack

Step 2: Specify stack details

Step 3: Configure stack options

Step 4: Review and create

Provide a stack name

Stack name

salmans-Fourth-stack

Stack name can include letters [A-Z and a-z], numbers [0-9], and dashes [-].

Parameters
Parameters are defined in your template and allow you to input custom values when you create or update a stack.

No parameters
There are no parameters defined in your template.

5. Created **Successfully**

salmans-Fourth-stack

Stack info | Events | **Resources** | Outputs | Parameters | Template | Change sets | Git sync - new

Resources (1)

Logical ID	Physical ID	Type	Status	Module
MySQSQueue	https://sqs.us-east-1.amazonaws.com/2111257/83778/salmansFirst.fifo	AWS::SQS::Queue	CREATE_COMPLETE	-

6. See the **SalmanFirst.fifo** is Created According to our **given code**

Amazon SQS > Queues

Queues (1)

Name	Type	Created	Messages available	Messages in flight	Encryption	Content-based deduplication
salmansFirst.fifo	FIFO	2024-04-20T14:06:05:30	0	0	Amazon SQS key (SSE-SQS)	Disabled

7. Create a Message in SQS

Message body
Enter the message to send to the queue.
Hi I am Salman, I'm here to Test Messages to check sqs

Message group ID
The tag that identifies that a message belongs to a specific message group.
sqsmessage

Message deduplication ID
The token used for deduplication of messages within the deduplication interval.
sqsmessagedup

► Message attributes - Optional info

Receive messages info

Messages available: 1 | Polling duration: 30 | Maximum message count: 10 | Polling progress: 0 receives/second 0%

Messages (0)

Search messages

View details | Delete

◀ 1 ▶

ID	Sent	Size	Receive count
----	------	------	---------------

No messages. To view messages in the queue, poll for messages.

Poll for messages

8. And Creating Multiple Messages and Poll for Messaging

✓ Your message has been sent and is ready to be received.

Message body
Enter the message to send to the queue.
fsfsfsfs

Message group ID
The tag that specifies that a message belongs to a specific message group.
fsfsfs

Message deduplication ID
The token used for deduplication of messages within the deduplication interval.
fsfsfwsf

► Message attributes - Optional info

Receive messages info

Messages available: 2 | Polling duration: 30 | Maximum message count: 10 | Polling progress: 2 receives/second 73%

Messages (2)

Search messages

View details | Delete

◀ 1 ▶

ID	Sent	Size	Receive count
bccr0aa4-29ef-4b61-b8f1-6d4cc8a87efd	2024-04-20T14:36+05:30	54 bytes	4
d1c04810-4ee4-4ae2-898c-7935e1b25d45	2024-04-20T14:39+05:30	10 bytes	1

9. Go To Amazon SES and Click Identities and Create Identity

Amazon SES

Get set up New
Account dashboard
Reputation metrics
SMTP settings

▼ Configuration
Identities
Configuration sets
Dedicated IPs
Email templates
Suppression list
Cross-account notifications
Email receiving

Amazon SES > Configuration: Identities

Identities

The Identities pane lists your domains, subdomains, and email address identities. All identities must be verified before you use them to send email in Amazon SES. Learn more. The Recommendations pane lists high-impact email authentication issues found for the identities you select and check for recommendations. Learn more.

Identities (0) info

Search all identities

Check for recommendations | Send test email | Delete | Create identity

◀ 1 ▶

Identity	Identity type	Identity status
----------	---------------	-----------------

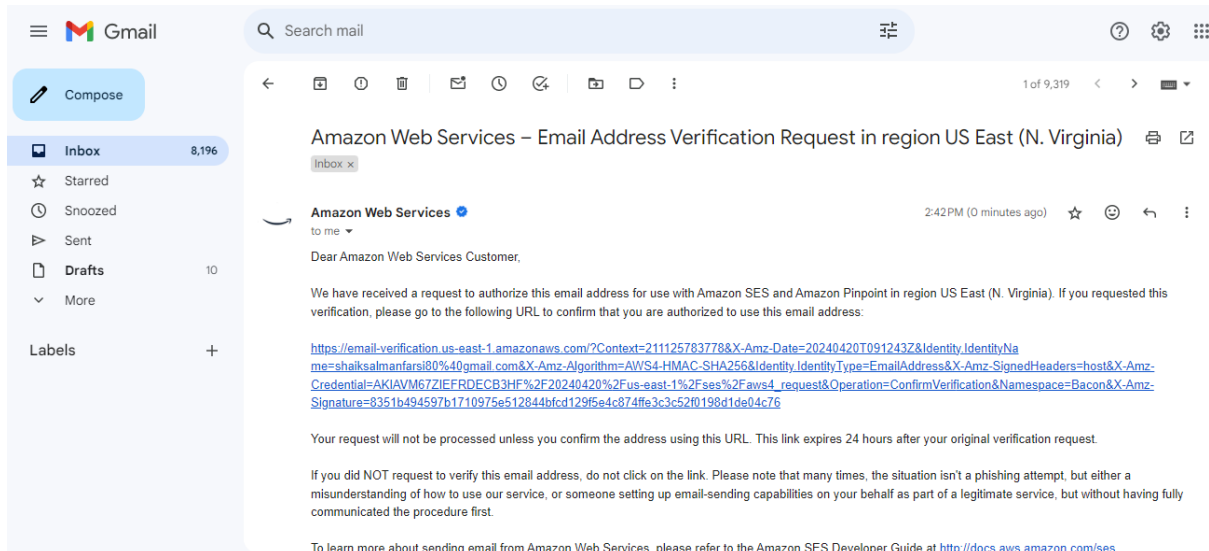
No identities
No identities to display.

Create identity

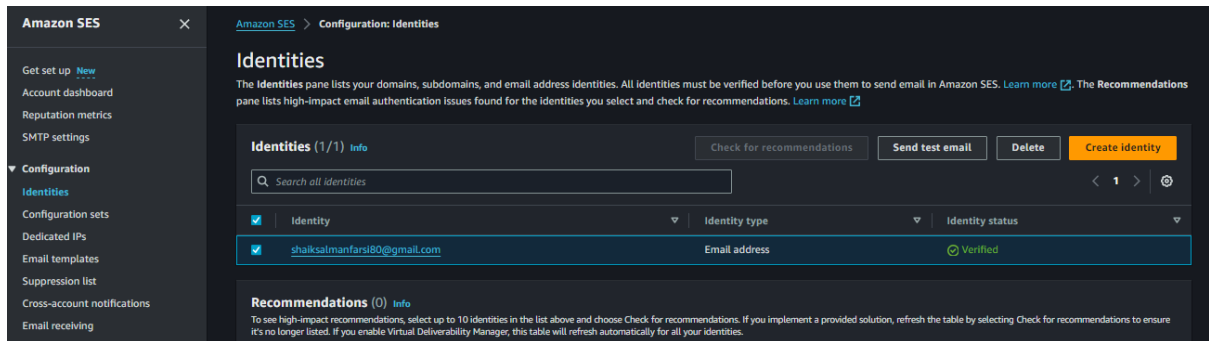
10. Click **Email Address** and **Give Your Mail Id**

The screenshot shows the AWS Management Console interface for the 'Create identity' page under 'Amazon SES > Configuration: Identities > Create identity'. The page title is 'Create identity'. Below the title, a description states: 'An identity is a domain, subdomain, or email address you use to send email through Amazon SES. Identity verification at the domain level extends to all email addresses under one verified domain identity.' The 'Identity details' section has an 'Info' tab. Under 'Identity type', there are two options: 'Domain' (unselected) and 'Email address' (selected). The 'Email address' option includes a note: 'To verify ownership of an email address, you must have access to its inbox to open the verification email.' A warning box states: 'Sending email from an email address identity without having the domain identity verified will result in your message being quarantined or rejected depending on the domain's DMARC policy. Learn more about DMARC and how to look up a domain's DMARC policy.' Below this, the 'Email address' field contains 'shaiksalmanfarsi80@gmail.com'. A note at the bottom says: 'Email address can contain up to 320 characters, including plus signs (+), equals signs (=) and underscores (_).'

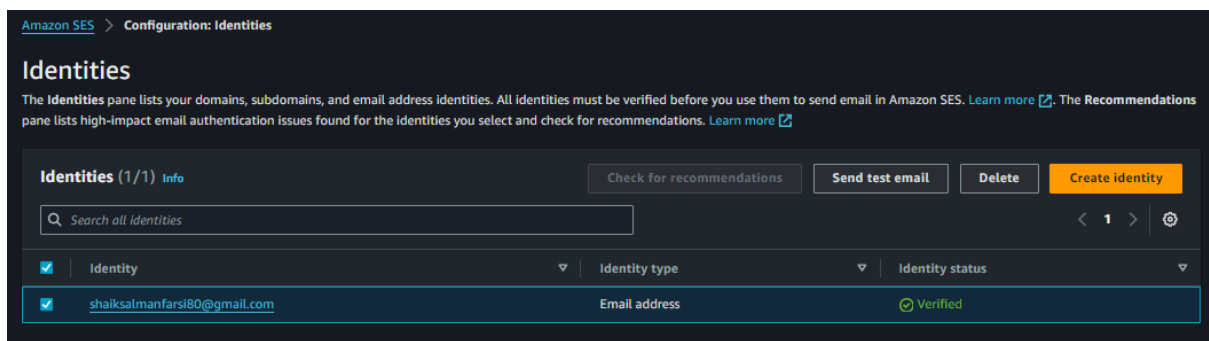
11. After u have to **verify it**



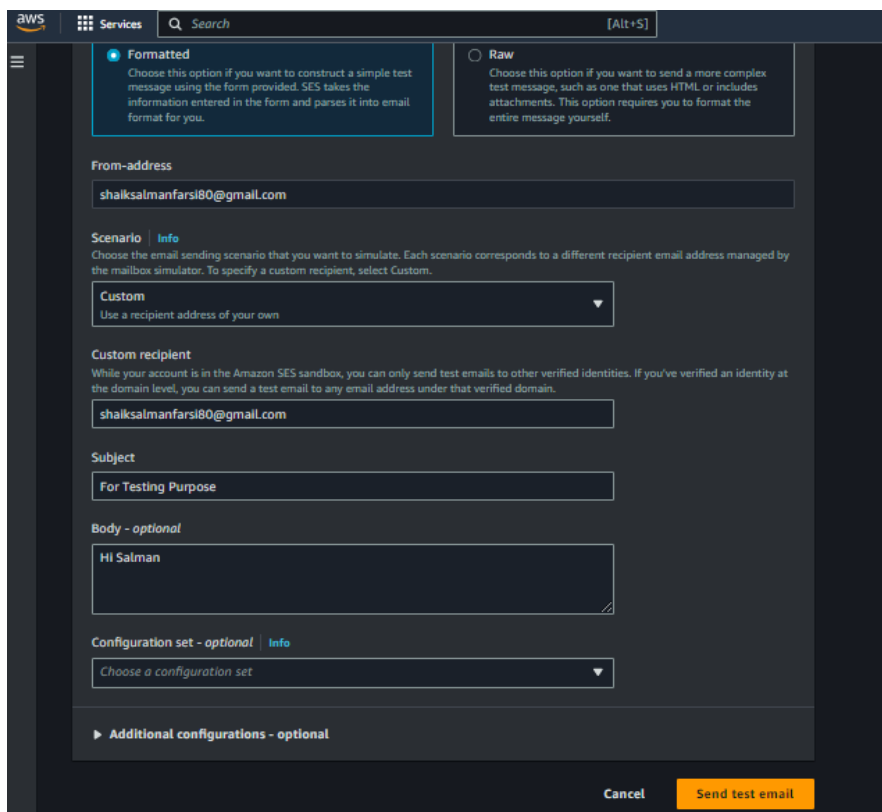
12. Select Email and Click Send Test Email



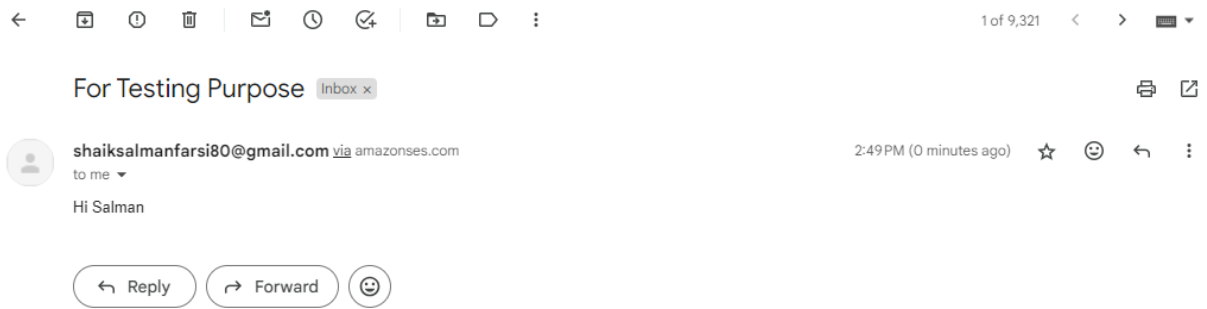
13. Same As it is Above the step



14. Simple Test Message then Select Formatted and From and To I selected Same And Subject is Testing Purpose and Body is Hi Salman



15. We Got Mail Via Amazonses.com and the Task Completed



Thank You