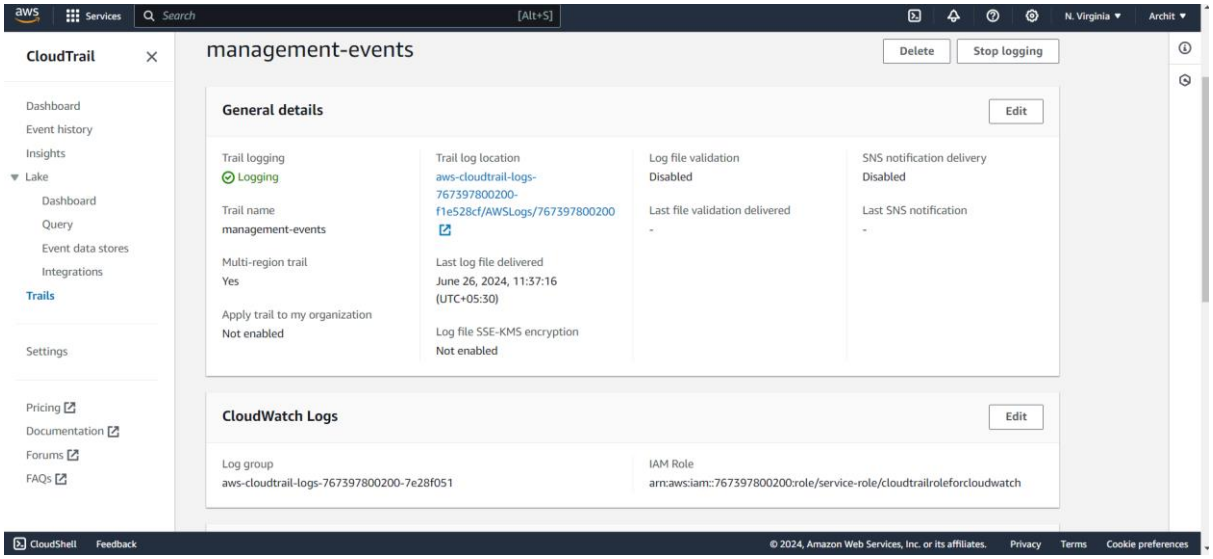
Auto tagging Resources Using Lambda

Step 1-Create A Cloud Trail with CloudWatch logs enabled(or we can use an existing one)here we only need event management trail



Step 2-Create A Role For Lambda Function(eg.autotag-role)

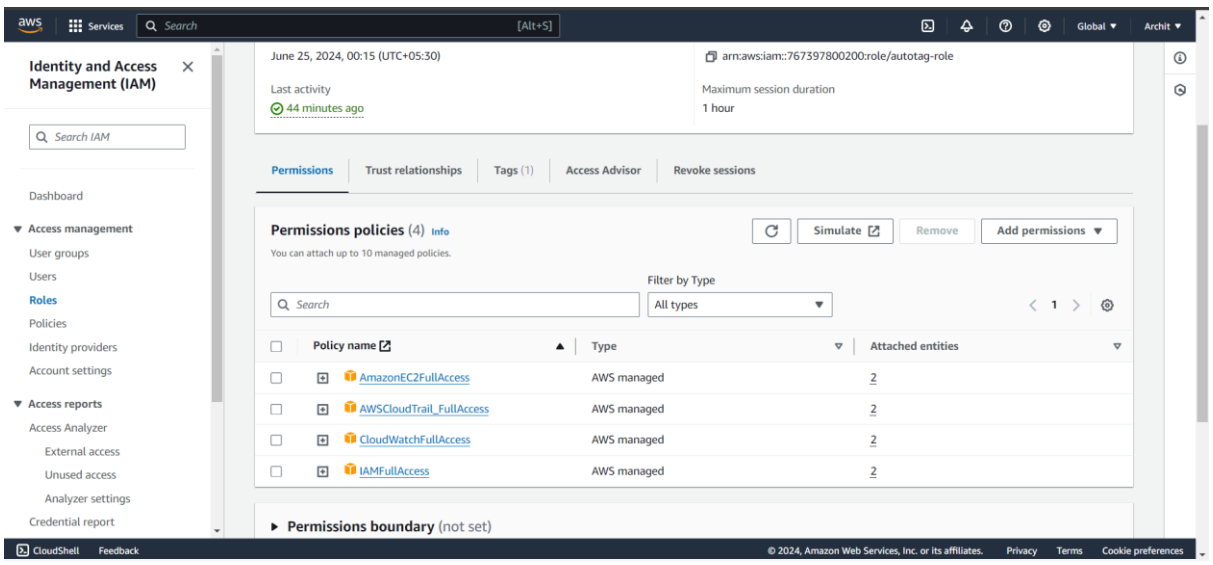
Give permissions:

AmazonEC2FullAccess

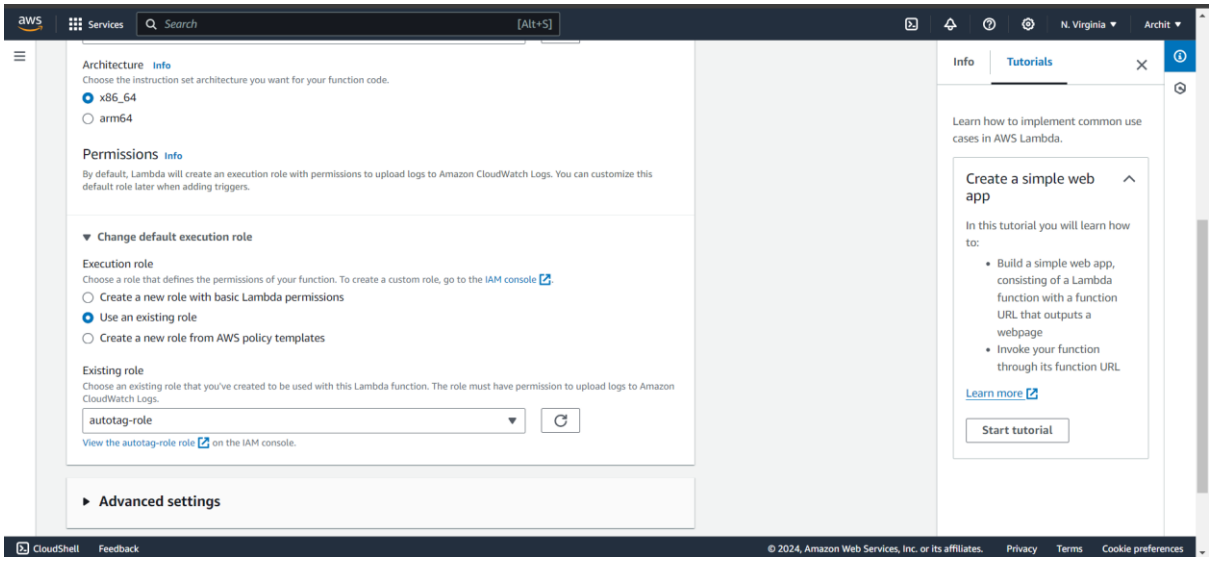
AWSCloudTrail_FullAccess

CloudWatchFullAccess

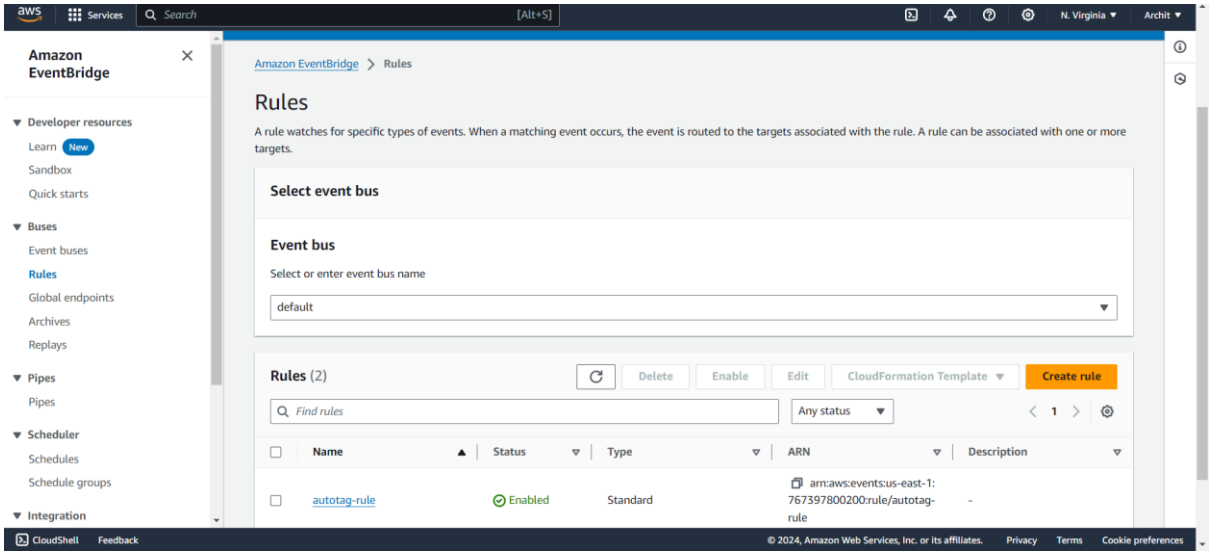
IAMFullAccess



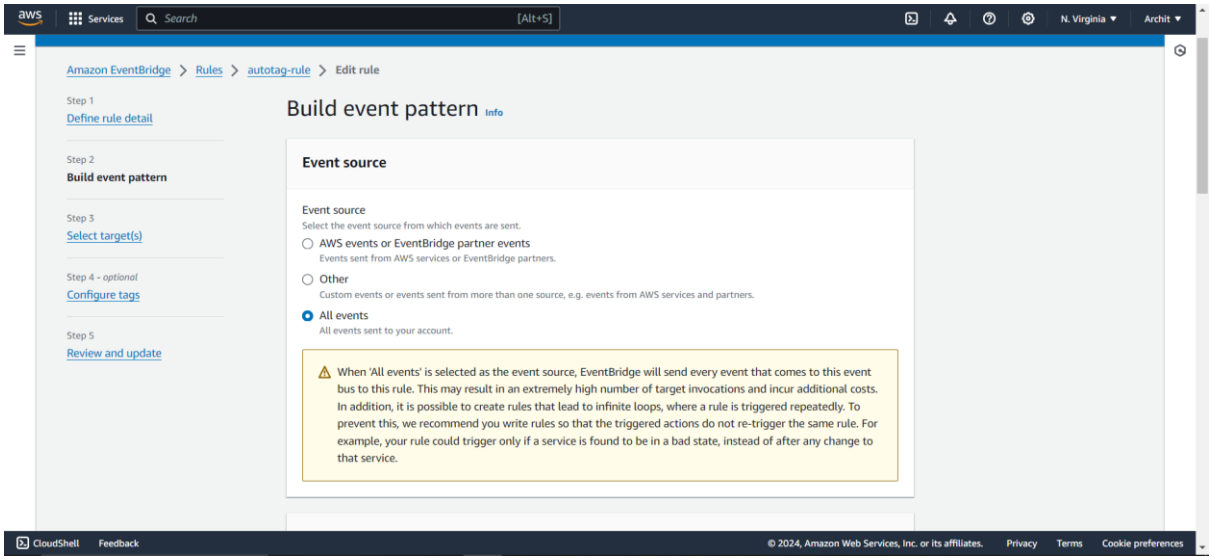
Step 3-Create A Lambda Function(Python 3.11), choose the role above which we have created



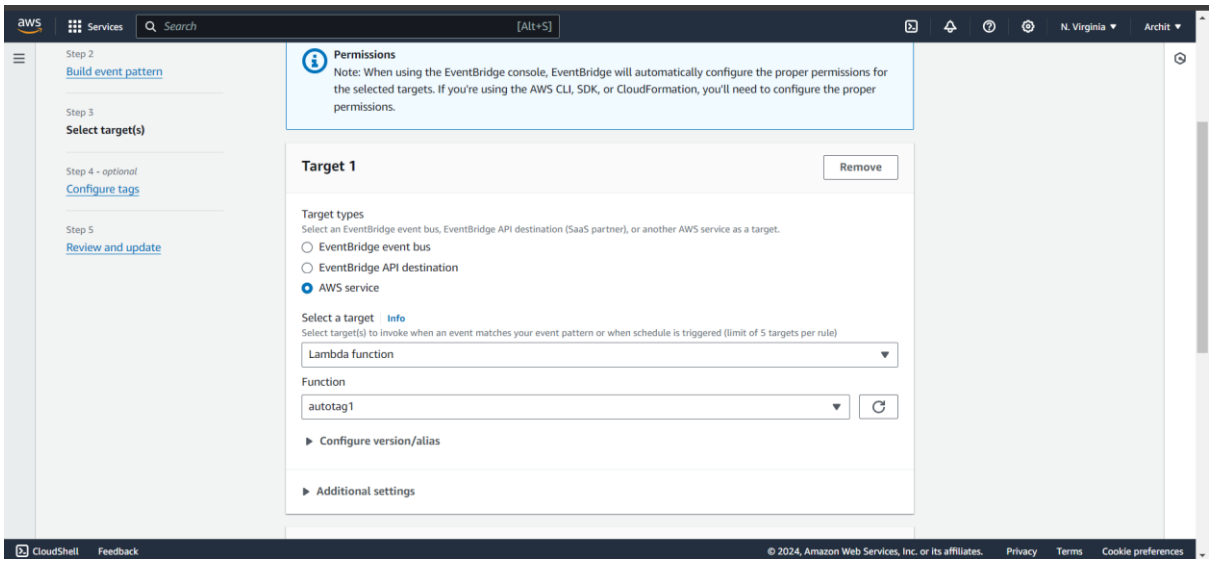
Step 4-Create A Rule in Cloud Watch



In event pattern choose all events

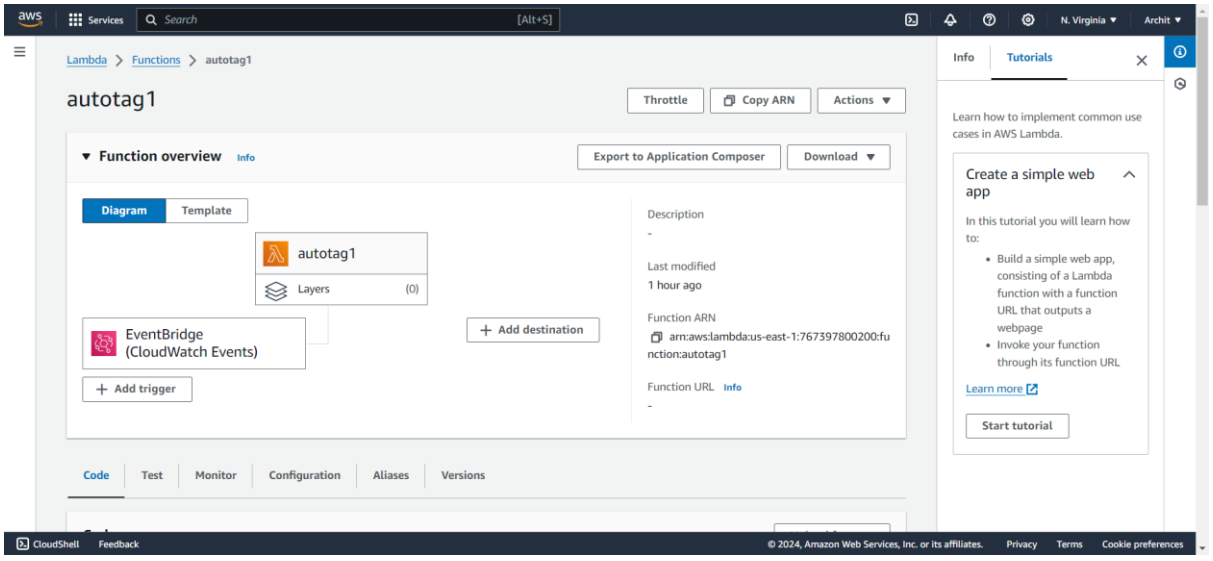


Select Targets

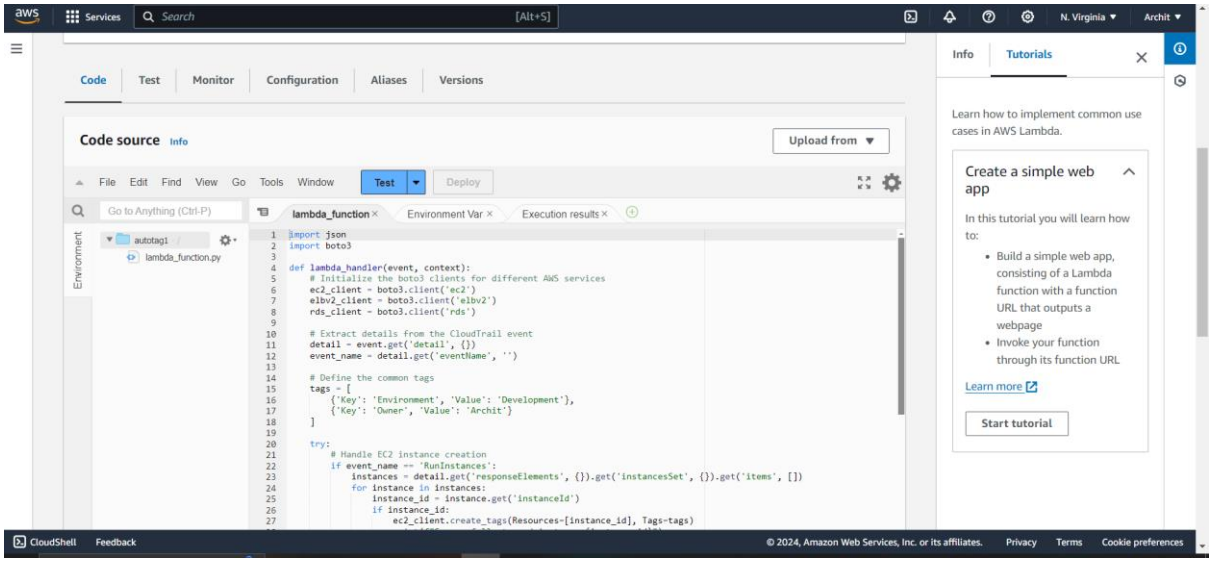


Finally create

Step5- check triggers will be automatically attached to event bridge



Step 6- write code in function below




```
import json
import boto3

def lambda_handler(event, context):
    # Initialize the boto3 clients for different AWS services
    ec2_client = boto3.client('ec2')
    elbv2_client = boto3.client('elbv2')
    rds_client = boto3.client('rds')

    # Extract details from the CloudTrail event
    detail = event.get('detail', {})
    event_name = detail.get('eventName', '')

    # Define the common tags
    tags = [
        {'Key': 'Environment', 'Value': 'Development'},
        {'Key': 'Owner', 'Value': 'Archit'}
    ]

    try:
```

```
    # Handle EC2 instance creation
    if event_name == 'RunInstances':
        instances = detail.get('responseElements', {}).get('instancesSet', {}).get('items', [])
        for instance in instances:
            instance_id = instance.get('instanceId')
            if instance_id:
                ec2_client.create_tags(Resources=[instance_id], Tags=tags)
                print(f"Successfully tagged instance {instance_id}")

    # Handle Elastic IP allocation
    elif event_name == 'AllocateAddress':
        allocation_id = detail.get('responseElements', {}).get('allocationId')
        if allocation_id:
            ec2_client.create_tags(Resources=[allocation_id], Tags=tags)
            print(f"Successfully tagged Elastic IP {allocation_id}")

    # Handle Security Group creation
    elif event_name == 'CreateSecurityGroup':
        group_id = detail.get('responseElements', {}).get('groupId')
        if group_id:
            ec2_client.create_tags(Resources=[group_id], Tags=tags)
            print(f"Successfully tagged Security Group {group_id}")

    # Handle Volume creation
    elif event_name == 'CreateVolume':
        volume_id = detail.get('responseElements', {}).get('volumeId')
        if volume_id:
            ec2_client.create_tags(Resources=[volume_id], Tags=tags)
```

```
            print(f"Successfully tagged Volume {volume_id}")

        # Handle Image creation
        elif event_name == 'CreateImage':
            image_id = detail.get('responseElements', {}).get('imageId')
            if image_id:
                ec2_client.create_tags(Resources=[image_id], Tags=tags)
                print(f"Successfully tagged Image {image_id}")

        # Handle Load Balancer creation
        elif event_name == 'CreateLoadBalancer':
            load_balancer_arn = detail.get('responseElements', {}).get('loadBalancers',
[{}])[0].get('loadBalancerArn')
            if load_balancer_arn:
                elbv2_client.add_tags(ResourceArns=[load_balancer_arn], Tags=tags)
                print(f"Successfully tagged Load Balancer {load_balancer_arn}")

        # Handle RDS instance creation
        elif event_name == 'CreateDBInstance':
            db_instance_arn = detail.get('responseElements', {}).get('dBInstanceArn')
            if db_instance_arn:
                rds_client.add_tags_to_resource(ResourceName=db_instance_arn,
Tags=tags)
                print(f"Successfully tagged RDS Instance {db_instance_arn}")

    except Exception as e:
        print(f"Error tagging resource: {e}")

    return {
        'statusCode': 200,
```

```
        'body': json.dumps('Auto-tagging completed.')
    }
```

Step 7- Testing Create any resource and check tags will be printed automatically