You can use AWS CloudWatch Events to schedule triggering of an AWS Lambda function at a specific time of day, and then use Lambda to start or stop your EC2 instances based on the schedule. Below are the steps to achieve this:

**Prerequisites**

An Ec2 instance within an isolated VPC.

1. **IAM Role Setup**

Ensure your Lambda functions have the necessary permissions to start and stop EC2 instances. Create an IAM role with the following policies attached:

* AWSLambdaBasicExecutionRole (basic Lambda execution permissions)
* AmazonEC2FullAccess (full access to EC2, or you can customize the permissions based on your needs)

1. **Create Lambda Functions**

You'll need two Lambda functions: one to start instances and another to stop them.

1. **Start EC2 Instances Lambda Function:** This function starts your EC2 instances.

Lambda function in Python that starts EC2 instances:

python

import boto3

ec2 = boto3.client('ec2')

def lambda\_handler(event, context):

# Define your instance IDs

instance\_ids = ['instance\_id\_1', 'instance\_id\_2', ...]

# Start the EC2 instances

response = ec2.start\_instances(InstanceIds=instance\_ids)

# Print the response

print("Started EC2 instances:", instance\_ids)

print("Response:", response)

This function uses the boto3 library, which is the AWS SDK for Python, to interact with AWS services. Replace 'instance\_id\_1', 'instance\_id\_2', ... with the actual IDs of the EC2 instances you want to start.

Make sure your Lambda function has the necessary IAM permissions to start EC2 instances. You can attach the AmazonEC2FullAccess policy to the IAM role associated with your Lambda function, or create a custom policy granting the ec2:StartInstances permission.

Once you've created and deployed this Lambda function, you can set up a CloudWatch Events rule to trigger it at the desired schedule.

1. **Stop EC2 Instances Lambda Function:** This function stops your EC2 instances.

Lambda function in Python that stops EC2 instances:

python

import boto3

ec2 = boto3.client('ec2')

def lambda\_handler(event, context):

# Define your instance IDs

instance\_ids = ['instance\_id\_1', 'instance\_id\_2', ...]

# Stop the EC2 instances

response = ec2.stop\_instances(InstanceIds=instance\_ids)

# Print the response

print("Stopped EC2 instances:", instance\_ids)

print("Response:", response)

Replace 'instance\_id\_1', 'instance\_id\_2', ... with the actual IDs of the EC2 instances you want to stop.

Make sure your Lambda function has the necessary IAM permissions to stop EC2 instances. You can attach the AmazonEC2FullAccess policy to the IAM role associated with your Lambda function, or create a custom policy granting the ec2:StopInstances permission.

Once you've created and deployed this Lambda function, you can set up a CloudWatch Events rule to trigger it at the desired schedule, just like you did for the Lambda function that starts EC2 instances.

1. **Step 3: CloudWatch Events Setup**

### Create a Rule for Starting EC2 Instances:

#### Start EC2 Instances Rule:

* Click on "Events" in the left sidebar under "Events" section.
* Click on "Create rule" button.
* Select "Schedule" as the event source.
* Choose "Fixed rate of" and specify the frequency at which you want the rule to trigger (e.g., once every day).
* Configure the cron expression to specify the exact time when you want the rule to trigger.
* Click on "Add target" and select "Lambda function".
* Choose the Lambda function you created earlier that starts EC2 instances.
* Click "Configure details".
* Provide a name and description for the rule.
* Click "Create rule" to create the rule.

#### Stop EC2 Instances Rule:

Follow the same steps as above, but create a new rule for stopping EC2 instances:

* Choose the Lambda function you created earlier that stops EC2 instances.
* Provide a name and description for the rule, indicating that it's for stopping EC2 instances.
* Click "Create rule" to create the rule.

1. **Testing**

* Test your Lambda functions manually first to ensure they work as expected.
* Set up test schedules in CloudWatch Events with short intervals for testing purposes.

1. **Monitor and Troubleshoot**

* Monitor CloudWatch Logs for your Lambda functions to troubleshoot any issues.
* Check the execution history in the Lambda console for any errors.

With these steps, you should be able to schedule the start and stop of your EC2 instances at specific times using AWS CloudWatch and Lambda.

**Points to Remember:**

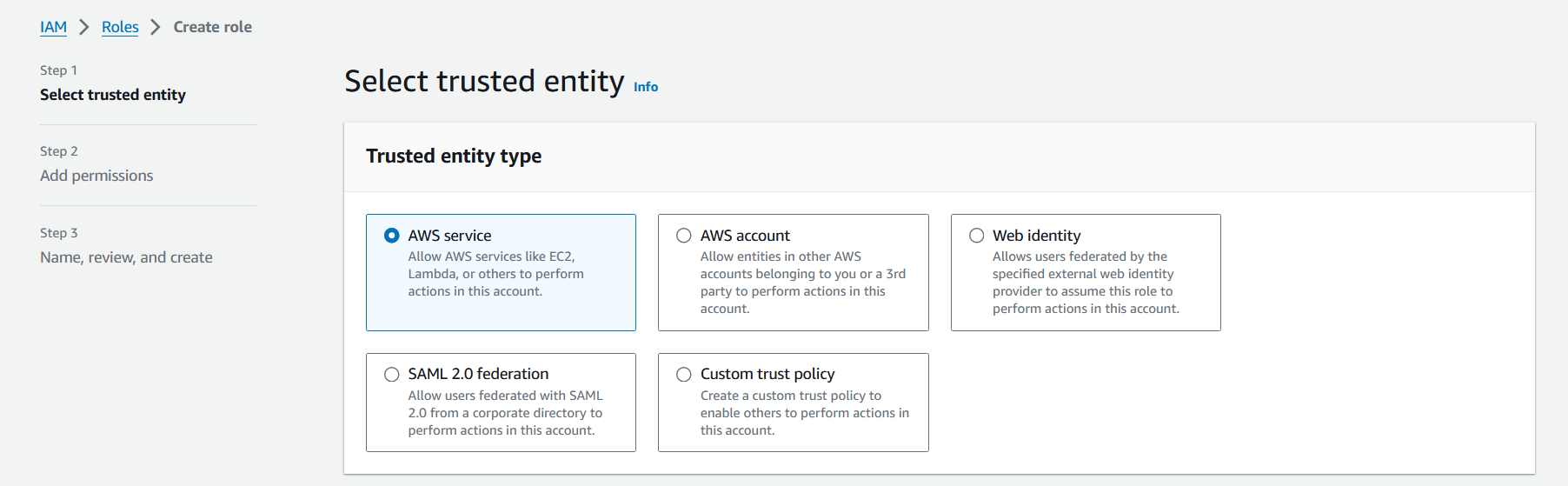
1. **IAM Permissions:**
   * Ensure that the IAM role attached to your Lambda function has the necessary permissions to start and stop EC2 instances.
2. **Error Handling:**
   * Implement proper error handling in your Lambda functions to handle any issues that may arise during execution, such as failures to start or stop instances.
3. **Monitoring:**
   * Monitor CloudWatch Logs for your Lambda functions to troubleshoot any errors or unexpected behavior.
   * Use CloudWatch Alarms to alert you to any issues with the Lambda functions or EC2 instances.
4. **Cost Considerations:**
   * Be mindful of the cost implications of running Lambda functions and EC2 instances. Ensure that your setup is cost-effective and aligns with your budget.
5. **Execution Time:**
   * Keep the execution time of your Lambda functions within the maximum allowed duration (15 minutes). If your operations take longer, consider breaking them into smaller tasks or using other AWS services like Step Functions.

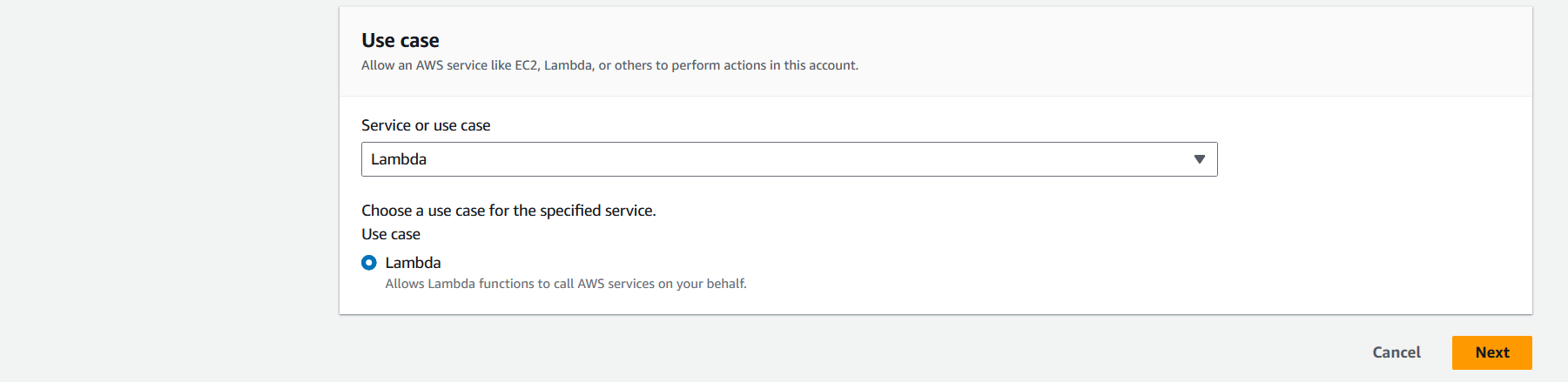
**Best Practices:**

1. **Separation of Concerns:**
   * Consider separating the logic for starting and stopping EC2 instances into separate Lambda functions. This improves modularity and makes it easier to manage and update your code.
2. **Tagging:**
   * Tag your EC2 instances appropriately to facilitate easier management and identification. You can use tags to specify which instances should be started or stopped by your Lambda functions.
3. **Logging and Monitoring:**
   * Use CloudWatch Logs to capture logs from your Lambda functions. Ensure that you log relevant information to help with troubleshooting and auditing.
   * Set up CloudWatch Alarms to monitor Lambda function invocations, errors, and EC2 instance status changes.
4. **Testing:**
   * Test your Lambda functions and CloudWatch Events rules thoroughly before deploying them in a production environment. Consider using staging or development environments for testing.
5. **Security:**
   * Follow AWS security best practices when configuring IAM roles and policies for your Lambda functions and EC2 instances.
   * Implement encryption and access controls as needed to protect sensitive data and resources.
6. **Documentation:**
   * Maintain clear and up-to-date documentation for your Lambda functions, CloudWatch Events rules, and EC2 instance configurations. This helps ensure that your setup is well-documented and understandable by other team members.

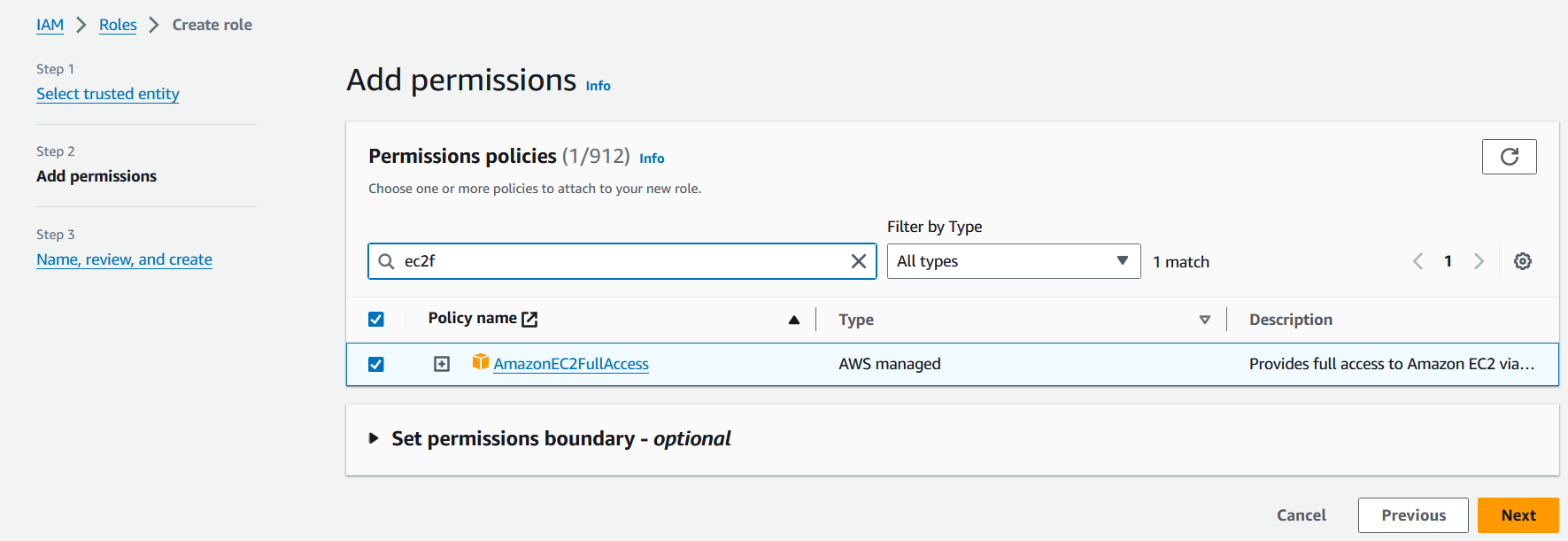
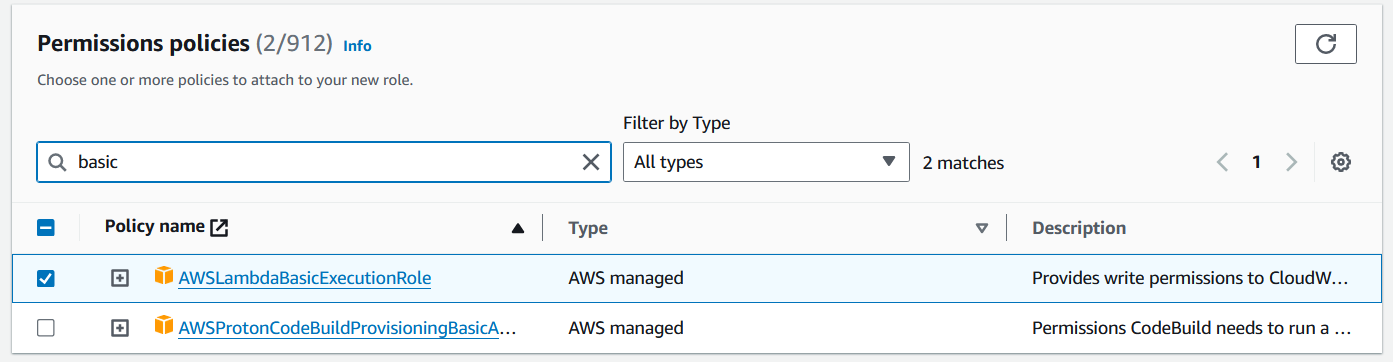
By following these points to remember and best practices, you can set up and manage CloudWatch Events rules and Lambda functions effectively for starting and stopping EC2 instances in your AWS environment.

1. Go to the AWS IAM console.
2. Click on "Roles" in the left sidebar.
3. Click "Create role".

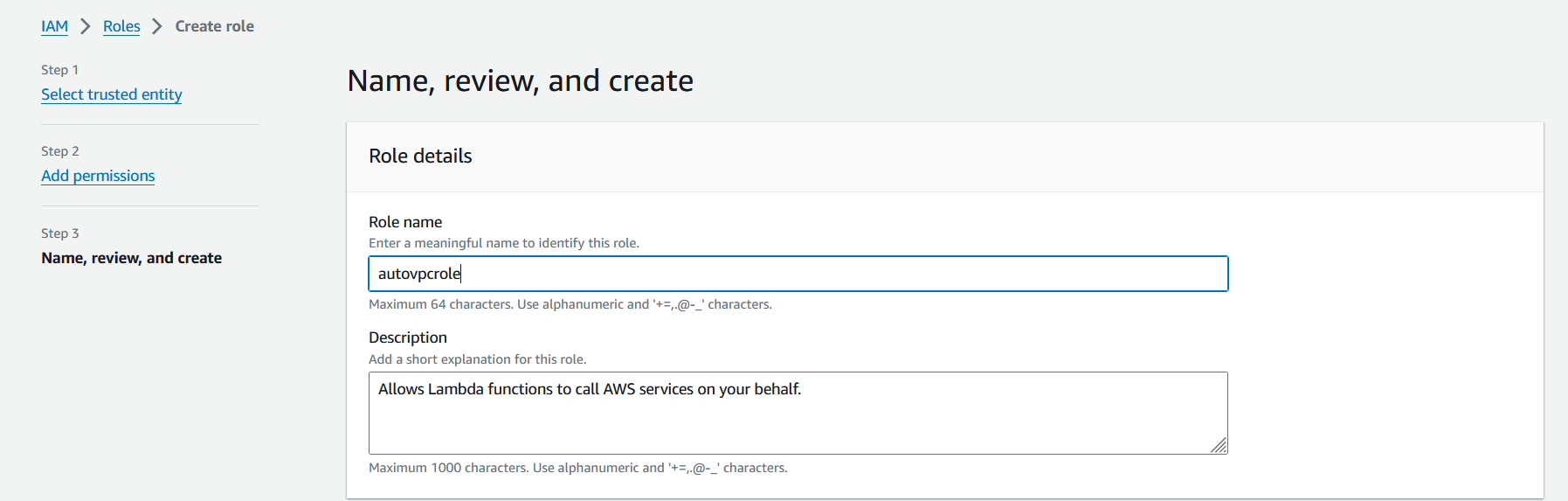


1. Choose "Lambda" as the service that will use this role, then click "Next: Permissions".
2. Ensure your Lambda functions have the necessary permissions to start and stop EC2 instances. Create an IAM role with the following policies attached:

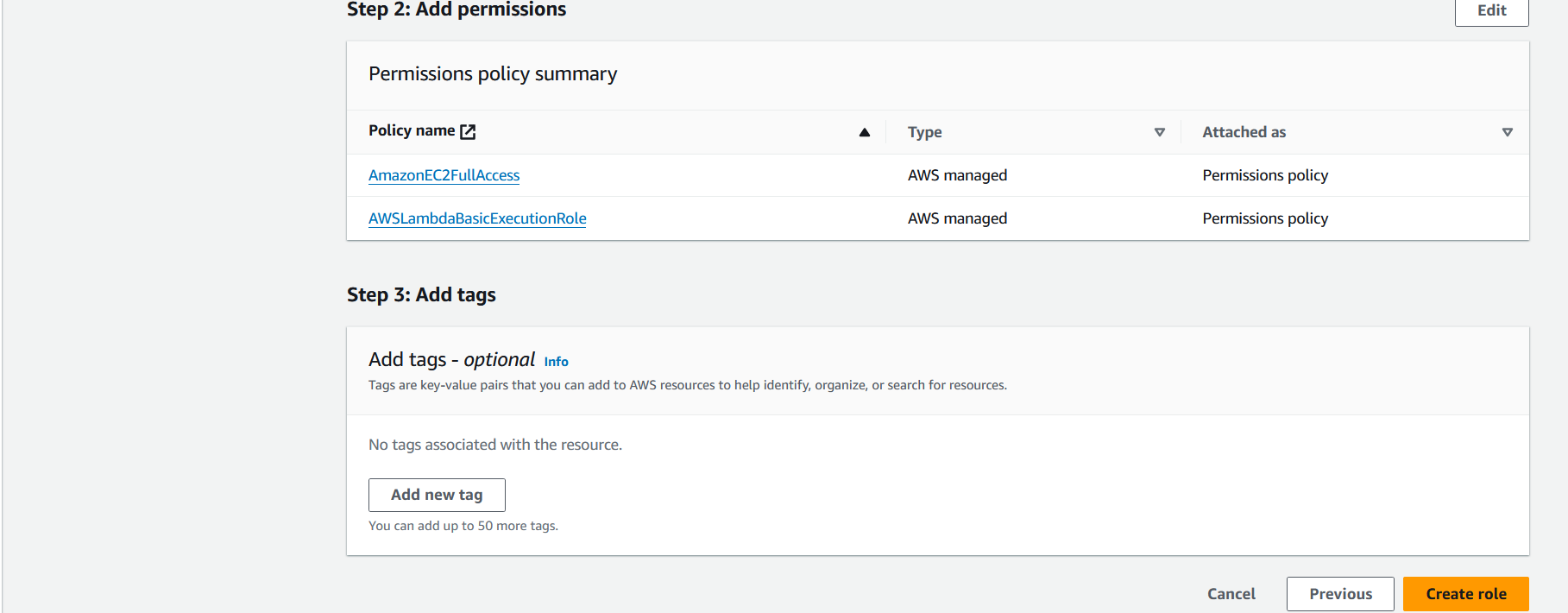
* AWSLambdaBasicExecutionRole (basic Lambda execution permissions)
* AmazonEC2FullAccess (full access to EC2, or you can customize the permissions based on your needs)

1. Click "Next: Tags" (optional), then "Next: Review".



1. Give your role a name and description, then click "Create role".

****