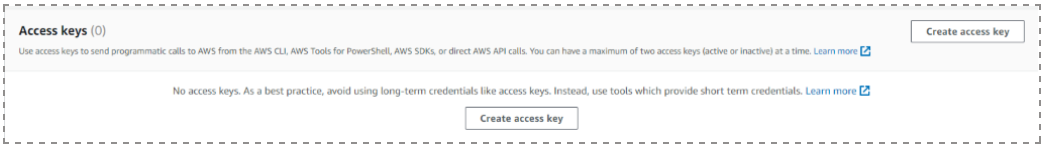
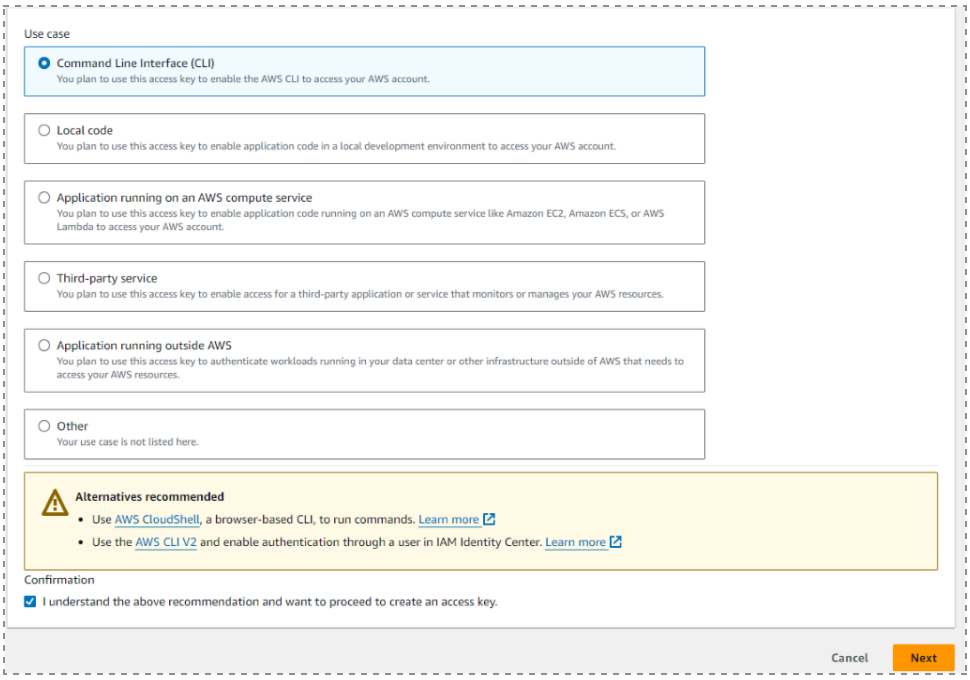
Kubernetes and Kubeflow - Setup

Prerequisites:

1. Get IAM access keys from AWS account
    - You can access this by clicking on the top right your AWS account name and selecting **Security Credentials**
    - Scroll to access keys and select **Create Access Key**
    - 

    - A new screen will appear where the **Command Line Interface(CLI)** will be selected and check the confirmation box at the bottom and select next
    - 

    - Add your description of the key and select **Create Access Key**
    - It will then show your Access key and Secret access key. **STORE/SAVE THESE FOR LATER USE**. (Recommend saving to home directory)

2. Connect to instance shell through ssh
    - Deactivate conda if it has not been already using: `deactivate conda`
    - Note: if conda ia activate (base) will show to the left of the user, you do not want that

Preparing the EC2 instance (Create an EC2 instance if not already created)

1. Run this command to install packages: `sudo apt update && sudo install -y git curl unzip tar make sudo vim wget`
2. Run these commands to install Kubeflow:
   - `export KUBEFLOW_RELEASE_VERSION=v1.6.1`
   - `export AWS_RELEASE_VERSION=v1.6.1-aws-b1.0.2`
   - `git clone https://github.com/awslabs/kubeflow-manifests.git && cd kubeflow-manifests`
   - `git checkout ${AWS_RELEASE_VERSION}`
   - `git clone --branch ${KUBEFLOW_RELEASE_VERSION} https://github.com/kubeflow/manifests.git upstream`
3. Edit the Configuration
   - Run the command: `vim Makefile` (or nano Makefile, text editor doesn't matter, we just need to edit the Makefile
   - Scroll down to install-jq and comment out the two lines listed
     and Add the line: `sudo apt-get install -y jq as shown below`
   - `***type "i" to insert text in vim***`
   -
     ```
     install-jq:
             #$(eval JQ_VERSION:=1.5+dfsg-2)
             #sudo apt-get install jq=$(JQ_VERSION) -y
             sudo apt-get install -y jq
     ```
   - Save + Exit the Makefile (esc, then ":wq")
   - Open the bashrc file using: `vim ~/.bashrc`
   - Set the default version of python: `alias python=python3.8`
   - Set the path variable..
   - It should look like this:
   -
     ```
     alias python=python3.8
     export PATH=/home/ubuntu/.local/bin:$PATH
     ```
   - Exit + save the ~/.bashrc file
   - Run the command to reload the changes: `source ~/.bashrc`
   - Deactivate conda if it is not already (command below): `conda deactivate`
   - Run the automatic tool install process: `make install-tools`
   - *If this command doesn't finish or gives dependency errors, anaconda needs to be removed using: `rm -r /path/to/anaconda`
4. Configure AWS CLI

- Run this to configure: `aws configure --profile=kubeflow`
- `AWS Access Key ID [None]: <enter your access key id>`
  `AWS Secret Access Key [None]: <enter your secret access key>`
  `Default region name [None]: us-east-1`
  `Default output format [None]: json`
- Fill out the blanks (or stuff with <>) with the access key ID and secret access key obtained during prerequisite
- Edit the ~./bashrc file again to include this on empty line: `export AWS_PROFILE=kubeflow`
- ```
  export AWS_PROFILE=kubeflow
  ```
- Save + Exit the file
- Run the command to reload the changes: `source ~/.bashrc`
- Run this command to check if AWS CLI was configured correctly: `aws sts get-caller-identity`
- It should display an output that looks like this:

```
{
    "UserId": "<redacted>",

   "Account": "<redacted>",

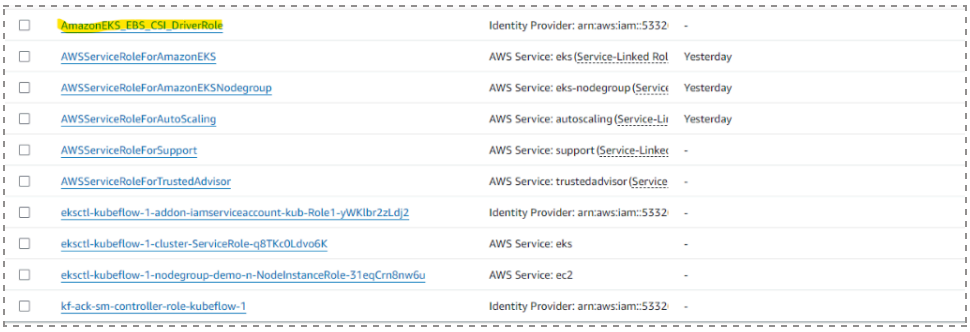
"Arn": "arn:aws:iam::<redacted>:user/<redacted>"

}
```

5. Create the cluster
   - Run this to export the cluster name (Note: this name can be changed, this works for our use case):
     ```
     export CLUSTER_NAME=kubeflow-2
     export CLUSTER_REGION=us-east-1
     ```
   - Run this to create and cluster and wait for it to be successfully created (20-30 minutes)
     ```
     eksctl create cluster \
         --name ${CLUSTER_NAME} \
         --version 1.23 \
         --region ${CLUSTER_REGION} \
         --nodegroup-name demo-nodes \
         --node-type m5.xlarge \
         --nodes 5 \
         --nodes-min 1 \
         --nodes-max 10 \
         --managed \
     ```
     ```

```
--with-oidc
```

6. Install the Amazon EBS CSI Driver
   - Create role by running: `eksctl create iamserviceaccount --name ebs-csi-controller-sa --namespace kube-system --cluster my-cluster --role-name AmazonEKS_EBS_CSI_DriverRole2 --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy --approve`
   - Rule should show up in AWS console (not necessary in this case, we already create a rule that is added to the instances)
   - For example, it would look like this:



☐	AmazonEKS_EBS_CSI_DriverRole	Identity Provider: arn:aws:iam::5332	-
☐	AWSServiceRoleForAmazonEKS	AWS Service: eks (Service-Linked Rol	Yesterday
☐	AWSServiceRoleForAmazonEKSNodegroup	AWS Service: eks-nodegroup (Servic	Yesterday
☐	AWSServiceRoleForAutoScaling	AWS Service: autoscaling (Service-Li	Yesterday
☐	AWSServiceRoleForSupport	AWS Service: support (Service-Linke	-
☐	AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service	-
☐	eksctl-kubeflow-1-addon-iamserviceaccount-kub-Role1-yWKIbr2zLdj2	Identity Provider: arn:aws:iam::5332	-
☐	eksctl-kubeflow-1-cluster-ServiceRole-q8TKc0Ldvo6K	AWS Service: eks	-
☐	eksctl-kubeflow-1-nodegroup-demo-n-NodeInstanceRole-31eqCrn8nw6u	AWS Service: ec2	-
☐	kf-ack-sm-controller-role-kubeflow-1	Identity Provider: arn:aws:iam::5332	-

   - Create addon by replacing <accountid> with the account id of aws account as shown in photo (remove dashes in between):

```
eksctl create addon --name aws-ebs-csi-driver --cluster kubeflow-2 --service-account-role-arn arn:aws:iam::<accountid>:role/AmazonEKS_EBS_CSI_DriverRole2 -force
```

   - (change kubeflow-2 to your cluster name if changed)
   - (change AmazonEKS_EBS_CSI_DriverRole to the name of new IAM role if added)
   - Get the update needed (change kubeflow-2 to your cluster name if changed):

```
eksctl get addon --name aws-ebs-csi-driver --cluster kubeflow-2
```

     (should list details, if nothing under update driver skip next step)
   - Update addon to the version listed in the command output

     (if necessary change kubeflow-2 to cluster name)
     (change <v1.31.0-eksbuild.1> to the correct updated driver without the <>)
     (replace <accountid> with the account id as done above)
     (if necessary change AmazonEKS_EBS_CSI_DriverRole to correct role created)

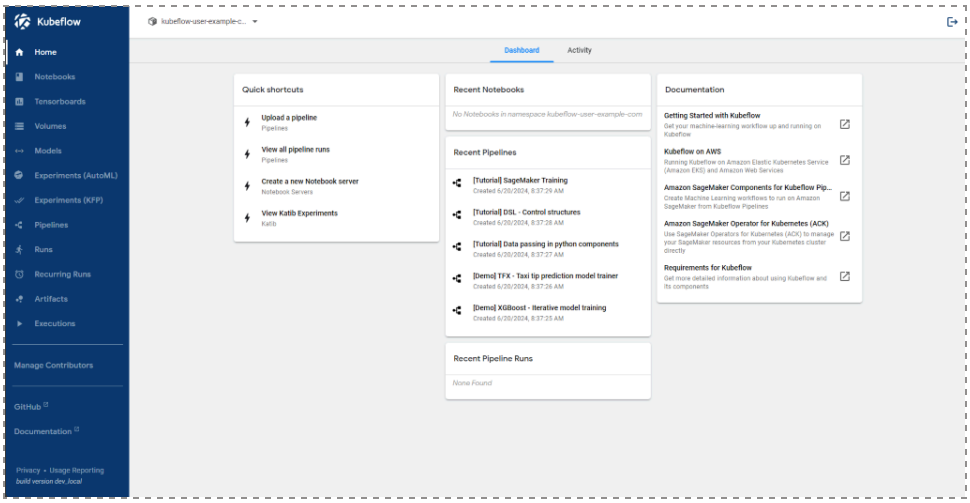
```
eksctl update addon --name aws-ebs-csi-driver --version <v1.31.0-eksbuild.1> --cluster kubeflow-2 \
```

```
  --service-account-role-arn
arn:aws:iam::<accountid>:role/AmazonEKS_EBS_CSI_DriverRole
2 -force
```

7. Add Kubeflow to the Cluster
   - Deploy Kubeflow: `make deploy-kubeflow INSTALLATION_OPTION=kustomize DEPLOYMENT_OPTION=vanilla`
   - Verify all of the services are running in the pods:
     ```
     kubectl get pods -n cert-manager
     kubectl get pods -n cert-manager
     kubectl get pods -n auth
     kubectl get pods -n knative-eventing
     kubectl get pods -n knative-serving
     kubectl get pods -n kubeflow
     kubectl get pods -n kubeflow-user-example-com
     ```
   - If everything returns running, Kubeflow has been installed. If there any issues with specific services, try installing those services again
   - Open up other terminal to allow dashboard connection (<desired_local_port> is the port you want to run locally):
     ```
     ssh -i /path/to/identity_key.pem -L
     <desired_local_port>:127.0.0.1:8080
     ubuntu@<EC2_PUBLIC_IPV4_ADDRESS>
     ```
   - Check your directory with running: `pwd`
   - If you are not already in kubeflowmanifest, run `cd kubeflowmanifest`
   - Once you are in the correct directory, run: `make port-forward`
   - Access Kubeflow from a browser using `127.0.0.1:<desired_local_port>`

     Kubeflow's default username is: `user@example.com`

     Kubeflow's default password is: `12341234`



Scaling Nodes in Cluster:

Open a new terminal and run: `eksctl scale nodegroup --cluster=clusterName --nodes=desiredCount --name=nodegroupName`

The Cluster must be scaled down to zero nodes after use, and then hibernated. In order for Kubeflow to run properly, it needs to be scaled up to five nodes.