Openfaas using AWS-EKS

1. Pre-requisites
    a. Install eksctl CLI
        i. curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.gz" | tar xz -C /tmp
        ii. mv /tmp/eksctl /usr/local/bin
        iii. eksctl version
    b. Install AWS CLI
        i. curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
        ii. sudo apt-get install unzip (Optional)
        iii. unzip awscliv2.zip
        iv. sudo ./aws/install
    c. Add your AWS credentials in the .aws folder
    d. Install Kubernetes CLI
        i. curl -LO https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl
        ii. curl -LO https://dl.k8s.io/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl.sha256
        iii. echo "$(<kubectl.sha256) kubectl" | sha256sum --check
    e. Install Helm CLI
        i. curl -fsSL -o get_helm.sh
        ii. chmod 700 get_helm.sh
        iii. ./get_helm.sh
        iv. helm version
    f. Install OpenFaaS CLI
        i. curl -sSL https://cli.openfaas.com | sudo -E sh
2. Create the cluster
    a. eksctl create cluster --name=openfaas-eks --nodes=2 --auto-kubeconfig --region=eu-central-1
    b. export KUBECONFIG=~/.kube/eksctl/clusters/openfaas-eks
    c. kubectl get nodes
3. Create Kubernetes Namespaces for OpenFaaS
    a. kubectl apply -f https://raw.githubusercontent.com/openfaas/faas-netes/master/namespaces.yml
    b. PASSWORD=$(head -c 12 /dev/urandom | shasum | cut -d' ' -f1)
    c. Echo $ PASSWORD
    d. kubectl -n openfaas create secret generic basic-auth \
       --from-literal=basic-auth-user=admin \
       --from-literal=basic-auth-password=$PASSWORD
4. Add OpenFaaS helm repository
    a. helm repo add openfaas https://openfaas.github.io/faas-netes/
5. Install OpenFaaS
    a. helm upgrade openfaas --install openfaas/openfaas \
       --namespace openfaas \
       --set functionNamespace=openfaas-fn \
       --set serviceType=LoadBalancer \
       --set basic_auth=true \
       --set operator.create=true \
       --set gateway.replicas=2 \
       --set queueWorker.replicas=2
6. Set OpenFaaS URL
    a. export OPENFAAS_URL=$(kubectl get svc -n openfaas gateway-external -o jsonpath='{.status.loadBalancer.ingress[*].hostname}'):8080 \
       && echo Your gateway URL is: $OPENFAAS_URL
7. Login
    a. echo $PASSWORD | faas-cli login --username admin --password-stdin
8. Application
    a. kubectl create deployment faasml --image="docker.io/sdgamer007/faasml:latest" -n openfaas
    b. kubectl expose deployment/faasml --type=LoadBalancer --port=5000 -n openfaas
    c. `kubectl get service faasml -n openfaas`
9. `Deploy ML Functions`
    a. Face Blur
        i. docker pull esimov/pigo-openfaas-faceblur:0.1
        ii. faas-cli deploy --image=esimov/pigo-openfaas-faceblur --name faceblur
        iii. Image Link: https://cnet4.cbsistatic.com/img/j7SdHs9Ac8coHkwTOcJG1DYcQI4=/940x0/2019/04/19/f20d0d6a-1781-49a4-90ab-e285109b65b2/avengers-endgame-imax-poster-crop.png
10. Change namespace
    a. kubectl config current-context
    b. kubectl config set-context iam-root-account@openfaas-eks-cc.eu-central-1.eksctl.io --namespace openfaas
11. Prometheus
    a. kubectl expose deployment prometheus --type=NodePort --name=prometheus-ui
    b. kubectl get svc prometheus-ui
    c. kubectl port-forward svc/prometheus-ui 9090:9090 &

12. Grafana
    a. kubectl run grafana --image=stefanprodan/faas-grafana:4.6.3 --port=3000
    b. kubectl expose deployment grafana --type=NodePort --name=grafana
    c. kubectl expose pod grafana --type=NodePort --name=grafana
    d. kubectl get service grafana
    e. kubectl port-forward svc/grafana 3000:3000 &