□ Step-by-Step Documentation of Your Commands

1. **Install and Update Packages**

```
sudo apt update
sudo apt install unzip -y
```

- **apt update** → Updates the package list to get the latest versions.
- **apt install unzip -y** → Installs `unzip` utility (used to extract `.zip` files).

---

2. **Install AWS CLI v2**

```
unzip awscliv2.zip
sudo ./aws/install
```

- **unzip awscliv2.zip** → Extract AWS CLI package.
- **sudo ./aws/install** → Installs AWS CLI v2 on your system.

---

3. **Install eksctl**

```
ARCH=amd64
PLATFORM=$(uname -s)_$ARCH
curl -sLO "https://github.com/eksctl-io/eksctl/releases/latest/download/eksctl_$PLATFORM.tar.gz"
tar -xzf eksctl_$PLATFORM.tar.gz -C /tmp && rm eksctl_$PLATFORM.tar.gz
sudo install -m 0755 /tmp/eksctl /usr/local/bin && rm /tmp/eksctl
```

- `eksctl` is a CLI tool for creating and managing **EKS (Elastic Kubernetes Service)** clusters.
- **ARCH** sets architecture (`amd64`, `arm64`, etc.).
- Downloads, extracts, and installs `eksctl`.

---

4. **Install kubectl**

```
curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
sudo chmod +x kubectl
sudo mv kubectl /usr/local/bin
```

- **kubectl** is the command-line tool for interacting with Kubernetes clusters.

---

5. **Configure AWS CLI**

```
aws configure
```

- Enter **Access Key, Secret Key, Region, Output format**.
- Stores credentials in `~/.aws/credentials`.

---

6. **Create EKS Cluster**

```
eksctl create cluster
kubectl get no
```

- Creates a default EKS cluster using `eksctl`.
- **kubectl get no** → Lists worker nodes in the cluster.

---

7. **IAM Policy for AWS Load Balancer Controller**

```
curl -o iam-policy.json https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.9.0/docs/install/iam_policy
aws iam create-policy --policy-name AWSLoadBalancerControllerIAMPolicy --policy-document file://iam-policy.json
```

- Downloads IAM policy JSON file.
- Creates AWS IAM policy for the **Load Balancer Controller**.

---

8. **Create Service Account with IAM Role**

```
eksctl utils associate-iam-oidc-provider --region ap-south-1 --cluster ferocious-outfit-1758549411 --approve
eksctl create iamserviceaccount \
  --cluster ferocious-outfit-1758549411 \
  --namespace kube-system \
  --name aws-load-balancer-controller \
  --attach-policy-arn arn:aws:iam::808932887013:policy/AWSLoadBalancerControllerIAMPolicy \
  --override-existing-serviceaccounts \
  --region ap-south-1 \
  --approve
kubectl get sa aws-load-balancer-controller -n kube-system -o yaml
```

- Associates **OIDC provider** to allow Kubernetes to use IAM roles.
- Creates a service account (`aws-load-balancer-controller`) in the cluster.
- Attaches IAM policy to it.

---

9. **Install Helm**

```
curl https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash
```

- Installs **Helm 3**, the package manager for Kubernetes.

---

10. **Install AWS Load Balancer Controller with Helm**

```
helm repo add eks https://aws.github.io/eks-charts
helm repo update
helm upgrade -i aws-load-balancer-controller eks/aws-load-balancer-controller \
  -n kube-system \
  --set clusterName=ferocious-outfit-1758549411 \
  --set region=ap-south-1 \
  --set vpcId=vpc-0f7cbba0622308ea4 \
  --set serviceAccount.create=false \
  --set serviceAccount.name=aws-load-balancer-controller
```

- Adds the **EKS Helm repo**.
- Installs the AWS Load Balancer Controller in Kubernetes.

---

11. **Deploy Applications**

```
vi dp-glass.yaml
vi dp-villa.yaml
vi svc-glass.yaml
vi svc-villa.yaml
kubectl create ns devnamespace
kubectl config set-context --current --namespace=devnamespace
kubectl apply -f .
```

- Created **Deployment and Service YAML files**.
- Applied them to namespace `devnamespace`.

---

12. **Check Resources**

```
kubectl get all
kubectl get pods -n devnamespace
kubectl get svc -n devnamespace
kubectl get ingress
```

- Lists deployments, services, pods, and ingress.

---

13. **Test Application with Ingress**

```
curl http://<ALB-DNS>/glass
curl http://<ALB-DNS>/villa
```

- Tests ingress routing using **AWS Load Balancer Controller**.

---

14. **Clean Up**

```
kubectl delete ns devnamespace
```

- Deletes namespace and all resources inside.
