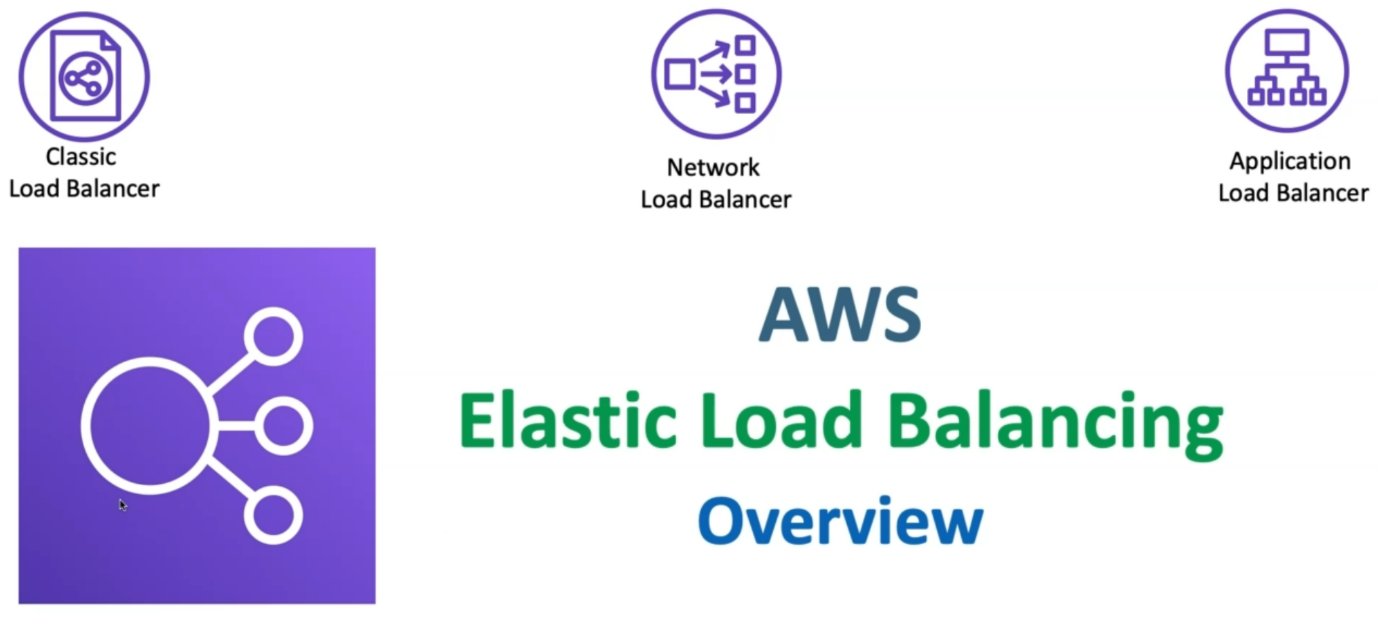
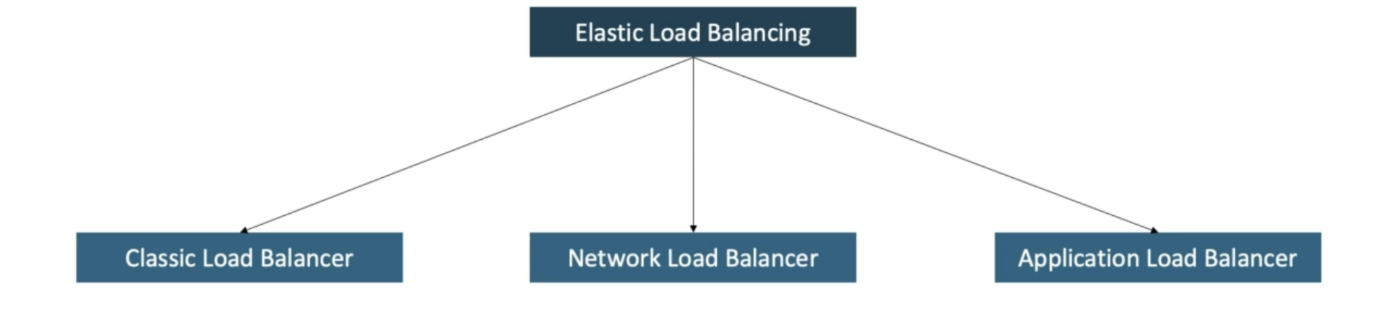
**1. AWS Load Balancers Introduction**

--- in this lecture, we will look into classic load balancer implementation on kubernetes.

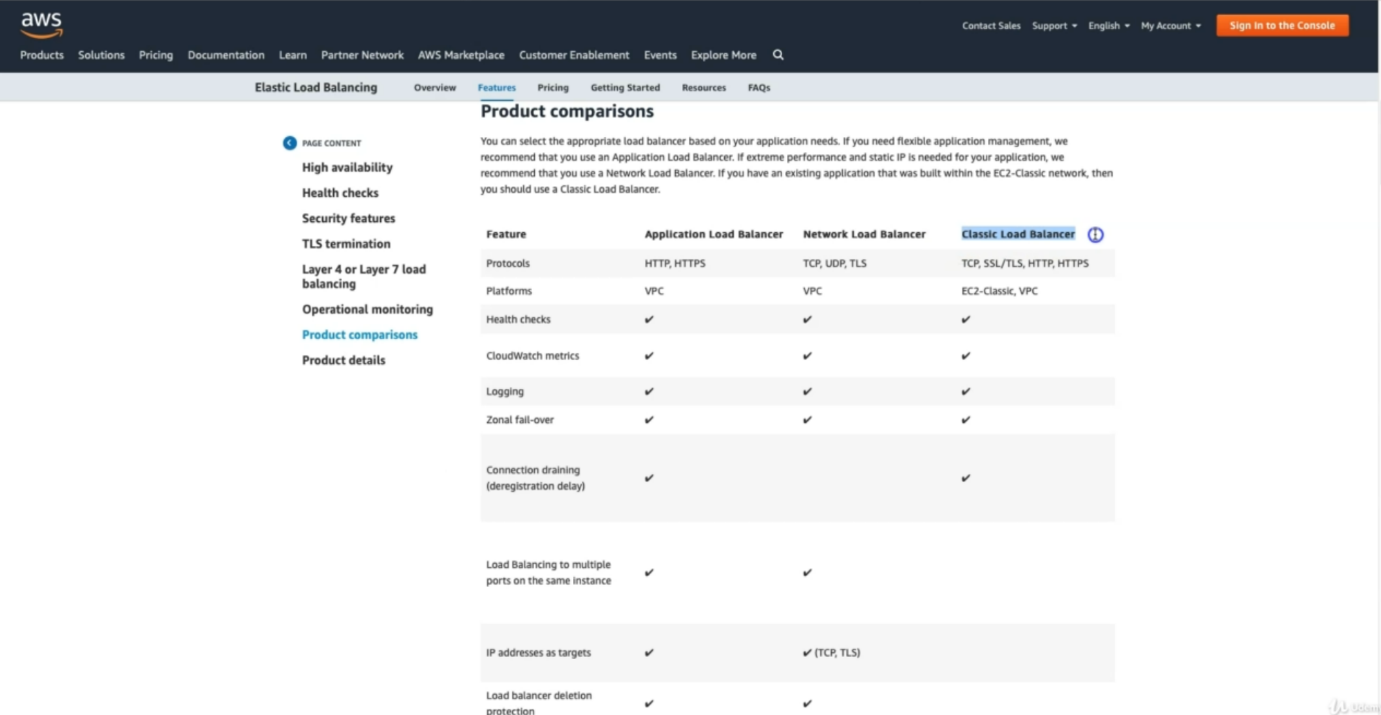


**Elastic load balancing types**

--- **note** – we will see what is elastic load balancing in aws. The elastic load balancers are 3 types.



**Difference between 3 load balancers**



--- **application load balancer** – it supports http and https protocols.

--- **network load balancer** – it supports TCP, UDP and TLS.

--- **Classic Load Balancer** – it supports TCP, SSL/TLS, HTTP, HTTPS. It is a legacy load balancer; it will be deprecated soon.

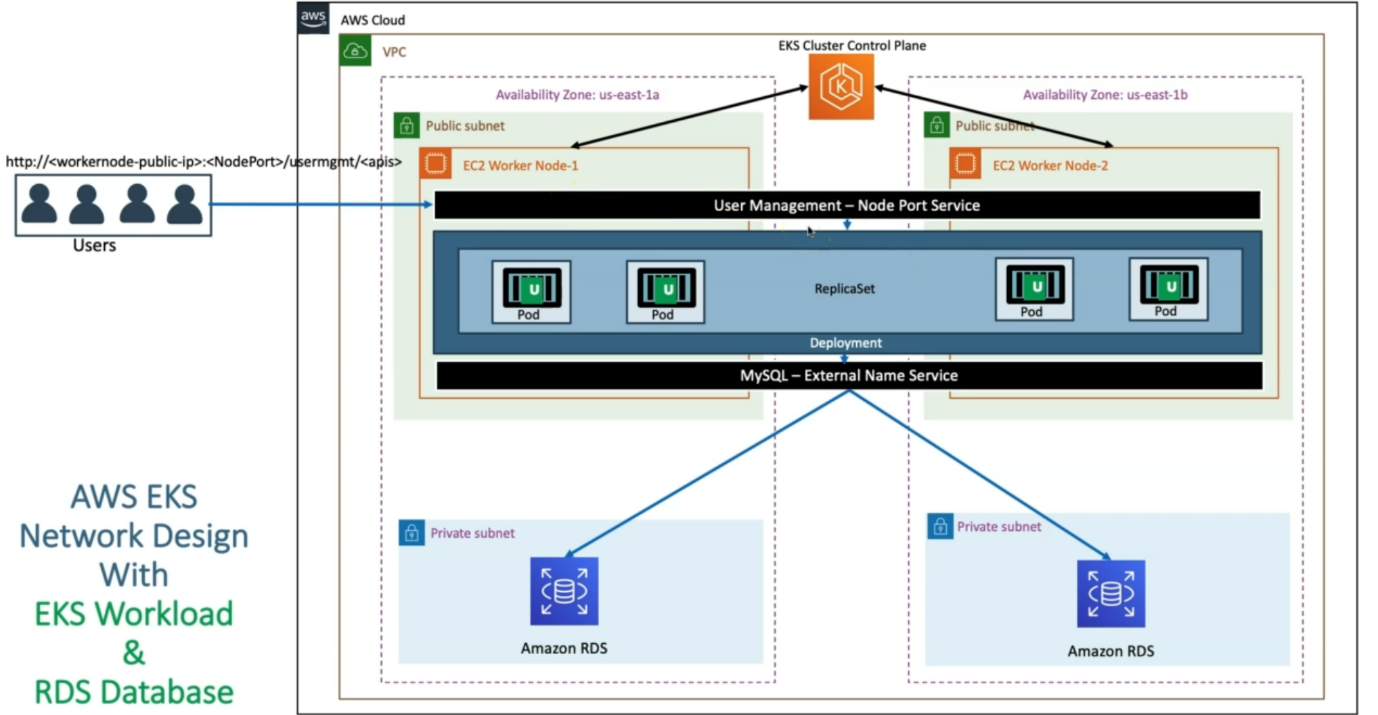
--- **note** – to create a classic or network or application load balancer, we are going too write a kubernetes service manifests.

**Real time concepts**

--- if you see earlier, what we have done is like with ideas, we have implemented the entire solution but what we have done there is like we have used the node port service for testing but now in a real production environment we are not going to use nodeport service. We are going to dns register our url and with our url, we are going to access it. So, if that is the case and we also need to have an 80 or 443. Instead of 31332 or some other ports.

--- We need 80 or 443, which all the default browser ports. For all those things, lets see start with using one of the load balancers for high availability and the start moving on with next steps.

**Previous architecture diagram**



--- **Note** - in our previous architectural diagram we have used nodeport service, we are going to replace with classic load balancer.

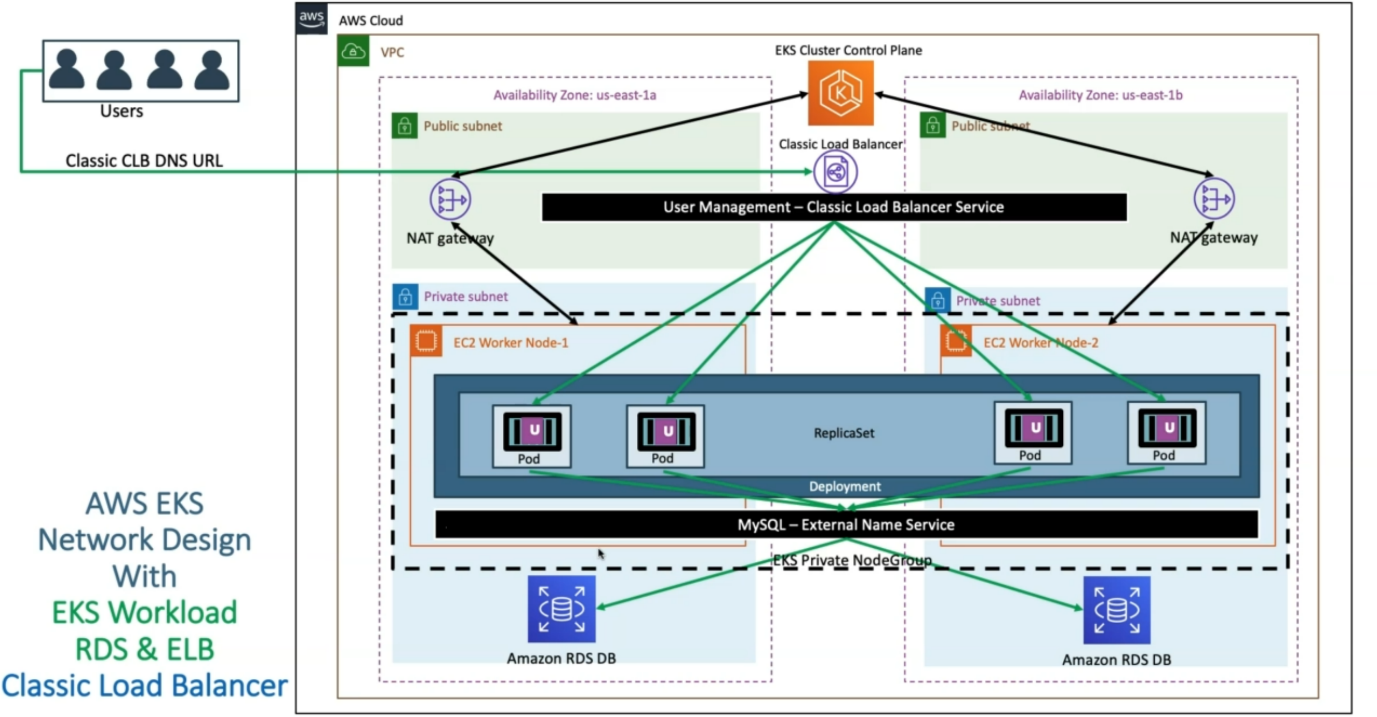
--- So before going and implementing the classic load balancer, we need to understand one important thing here that is EC2 worker node, which is nothing but the node group, EKS node group is created in the public subnets here.

--- So, end that cluster control plan is also in the public subnets and each guest node group is also in

--- So ideally, where in a 3-tier architecture where we deploy our workloads means like our application. In the private subnet. Right.

--- the node group will create in private subnet. Let’s see that now.

**New architectural diagram with classic load balancer**



--- in aws cloud, in a VPC. we have two availability zones. So, we created EKS Cluster Control Plan. So, what happens? it also creates equal public and then private subnets in the availability zones where we have provided to create the cluster.

--- So next, what happens is we are going to create a EKS private node group, which means, earlier the instances worker nodes present in public subnet now they are present in private subnet. So, the EC2 instances will be created in the private subnet.

--- **how that communication is going to happen earlier…?**

Worker nodes present in public subnet then it can directly connect to EKS cluster control plane. but if it is in private subnet, it is going to leverage the nat gateway to communicate.

--- instead of nat gateway for establish a communication between EKS cluster control plane and worker node. We can enable direct communication between EKS cluster control plane and worker node. For now, we will use nat gateway.

--- we will create Amazon RDS in the same private subnet.

--- we will create mysql external service and will configure with Amazon RDS end point.

--- we are going to deploy our user management application and our application will connect to the DB using mysql external service.

--- we will create user management classic LoadBalancer in public subnet.

--- whenever user try to access the application, they will use the Classic ALB DNS URL. From classic load balancer the load will be distribute among the pods. From pods the request goes to mysql external node port service. From mysql node port service the request will goes to the Amazon RDS.