Microsoft 365

Hands-on Autoscaling and Load-Balancing Using AWS

Experience the scalability and elasticity of cloud computing

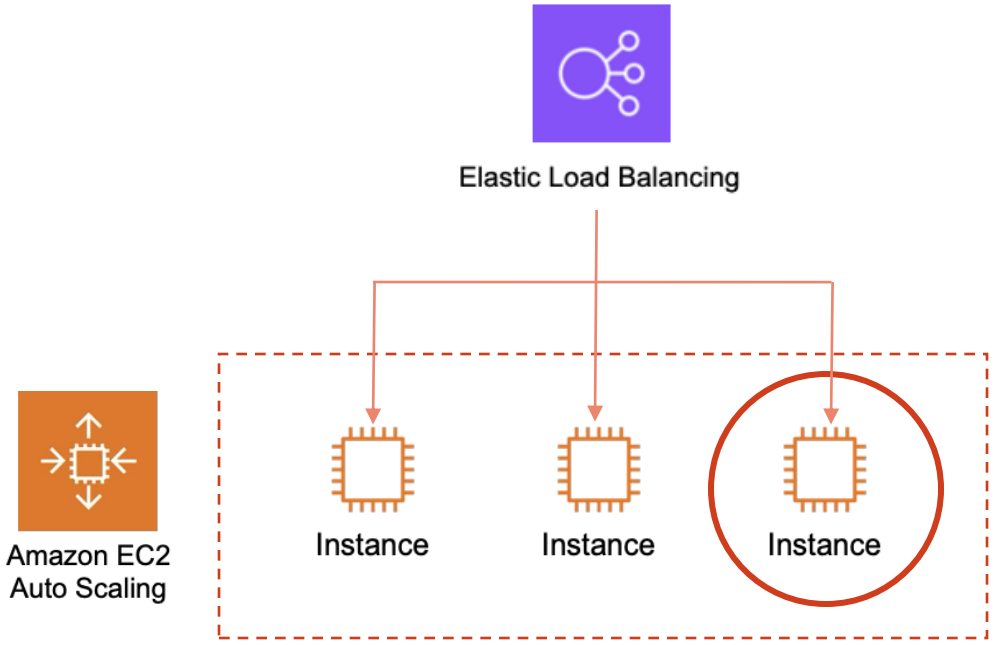
AY25/26 Semester 1 CS5224 Tutorial - Deng Siyang

Auto-scaling and Load-balancing

High availability, scalability, fault tolerance

- **Load Balancer**: distributes incoming client requests (e.g., web traffic) across multiple backend targets (like EC2 instances).

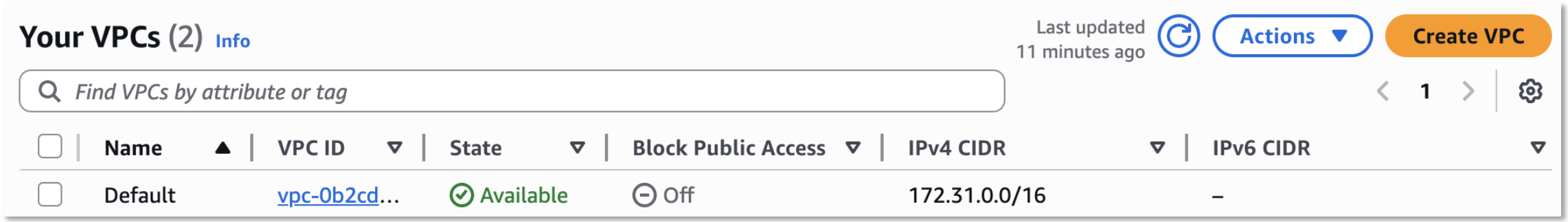
- **Autoscaling**: automatically adds or removes compute resources (e.g., EC2 instances) based on real-time demand.



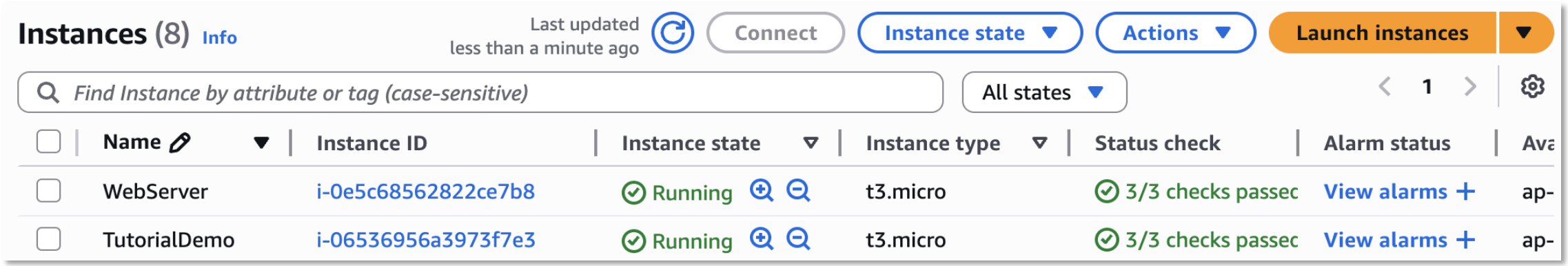
Note: LB focuses on distributing traffic across **existing** servers, while **AS** adjusts the **number** of servers based on demand.

Step-by-step on load-balancing

- Create **VPC**

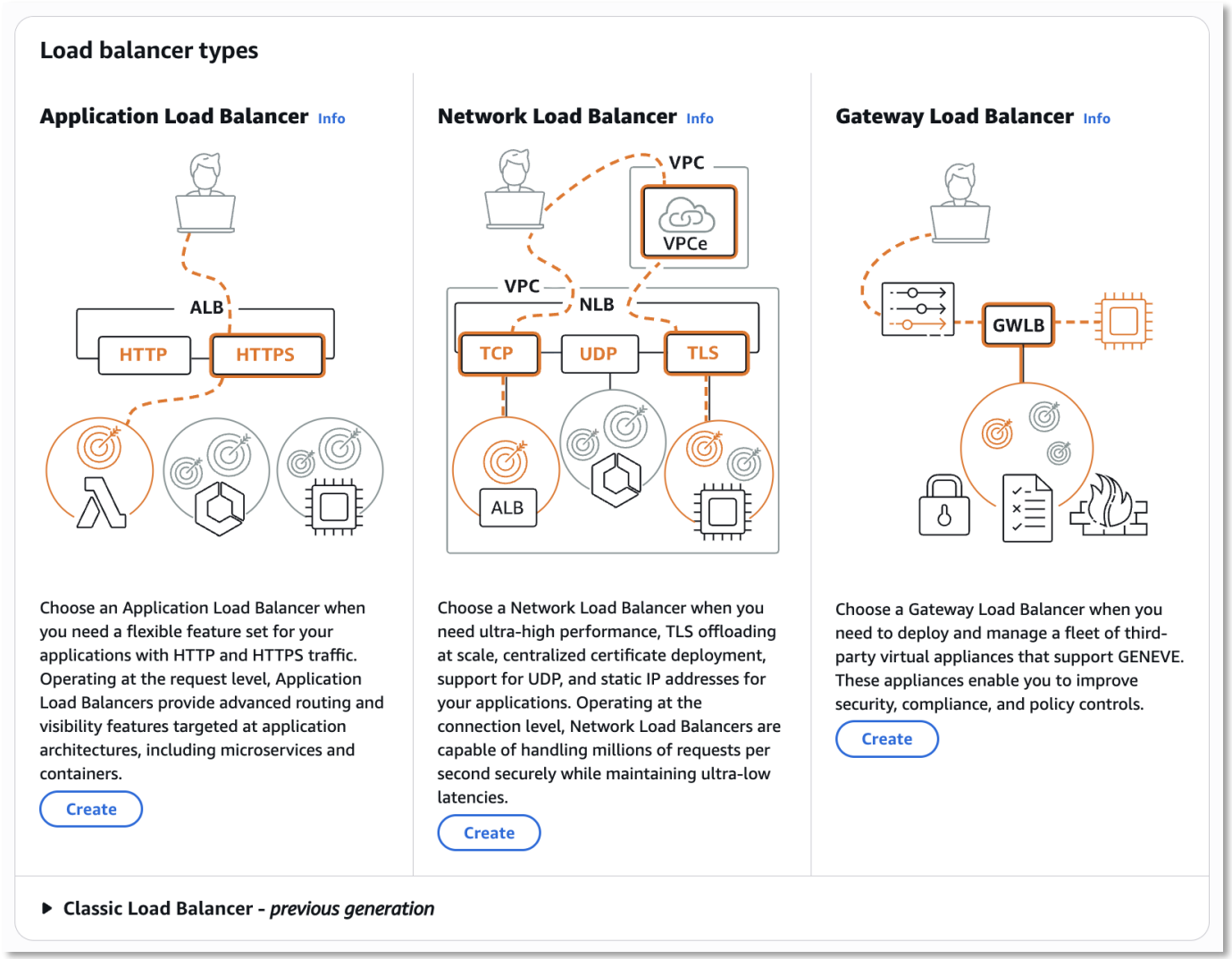


- Create at least 2 **EC2 instances**
  - Ensure you have web server for your machines



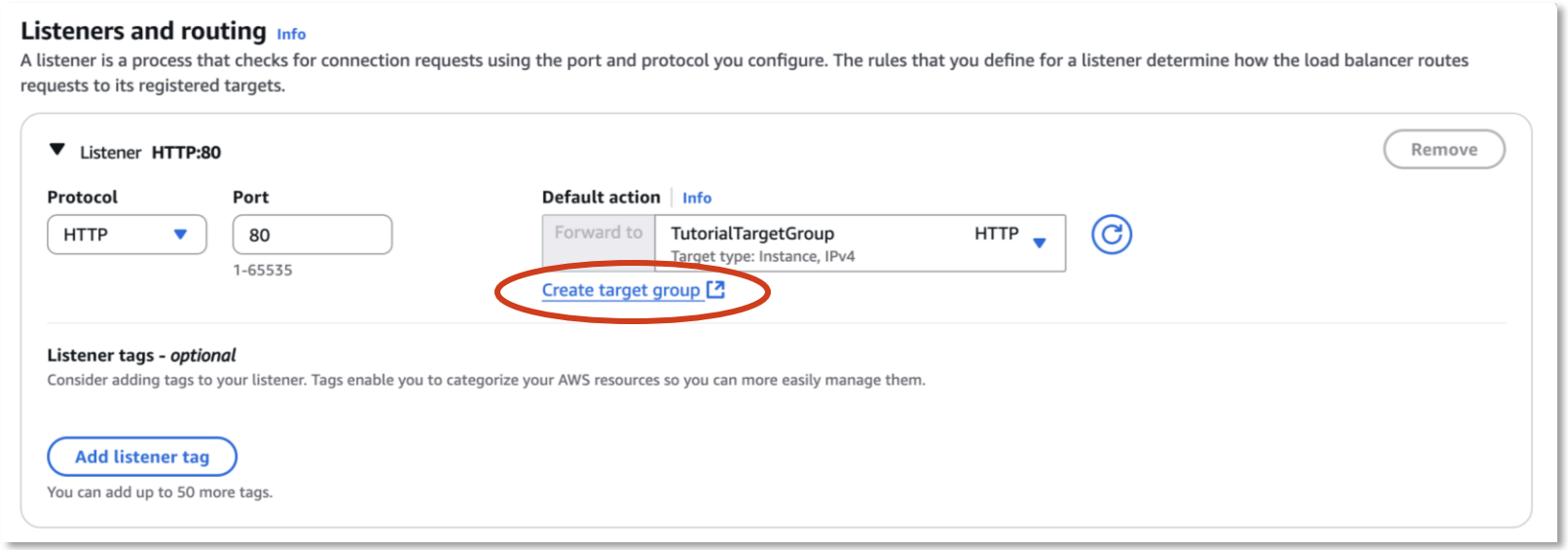
Step-by-step on load-balancing

- Setup **load balancer**
  - LoadBalancer type
  - Security group
  - Listener on target group

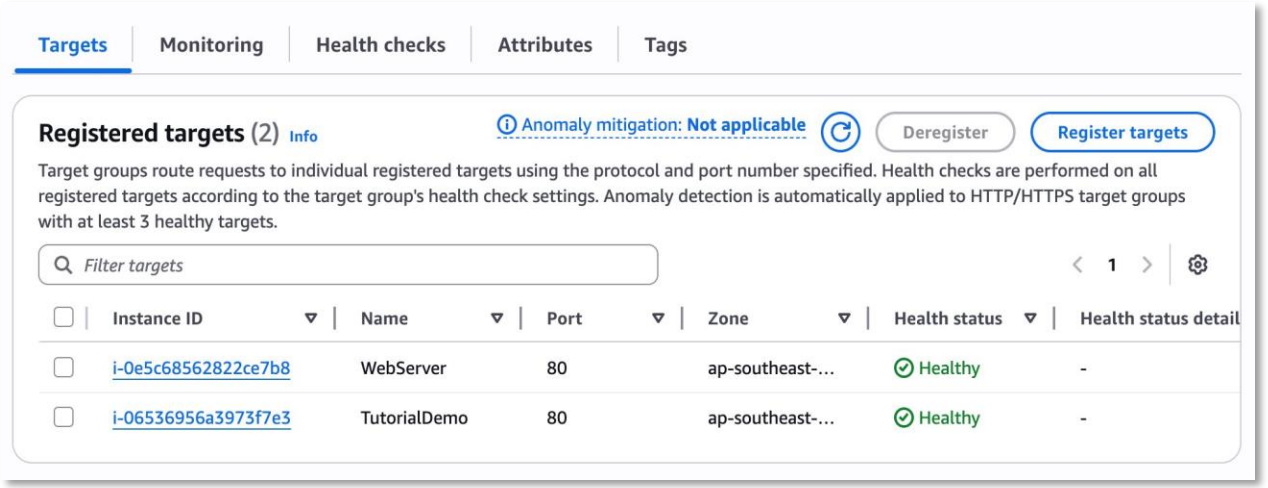


Step-by-step on load-balancing

- Setup **load balancer**
  - LoadBalancer type
  - Security group
  - Listener on target group

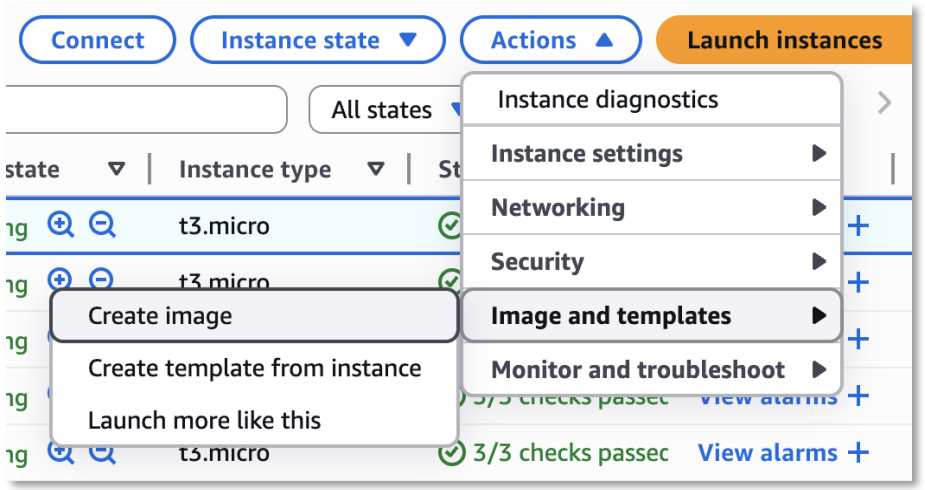


Target type: instances
Target name: TutorialTargetGroup
Health check protocol: HTTP
Health check path: /
Register targets: available instances



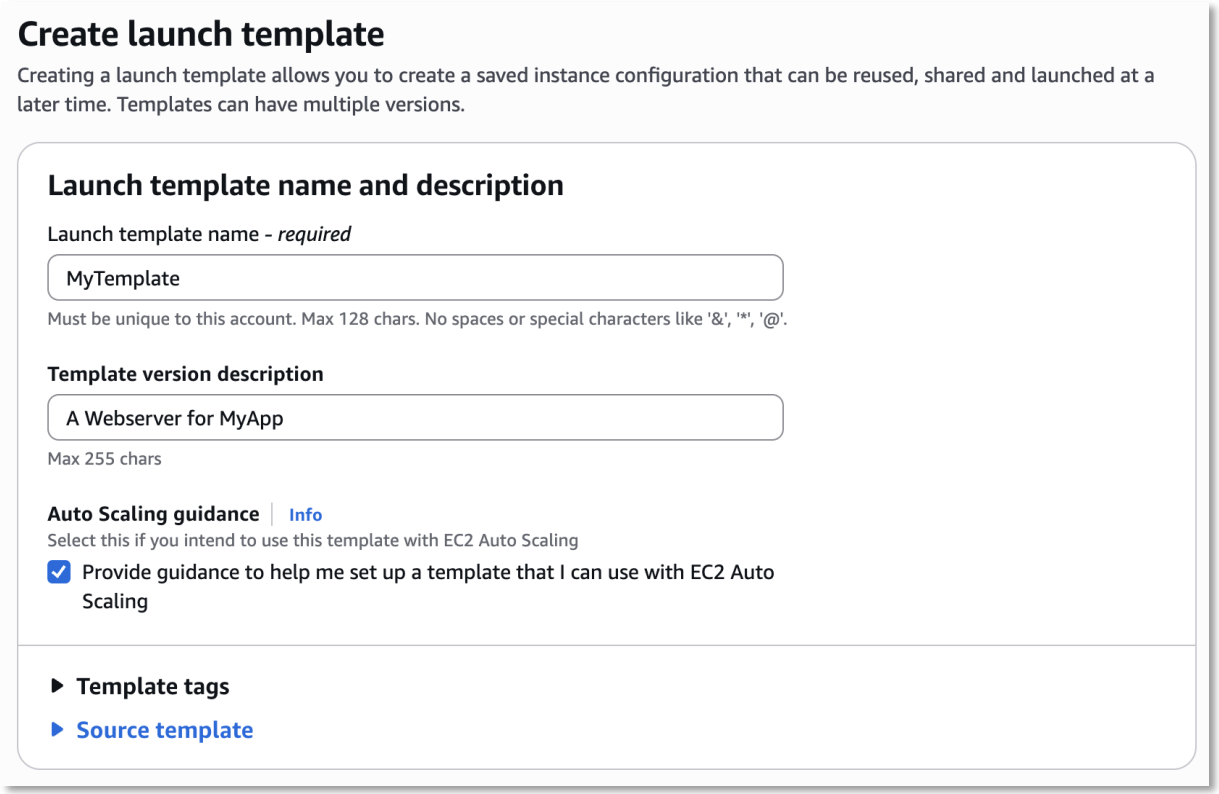
Step-by-step on autoscaling

- Create my **image**
  - With application, configured settings and dependencies



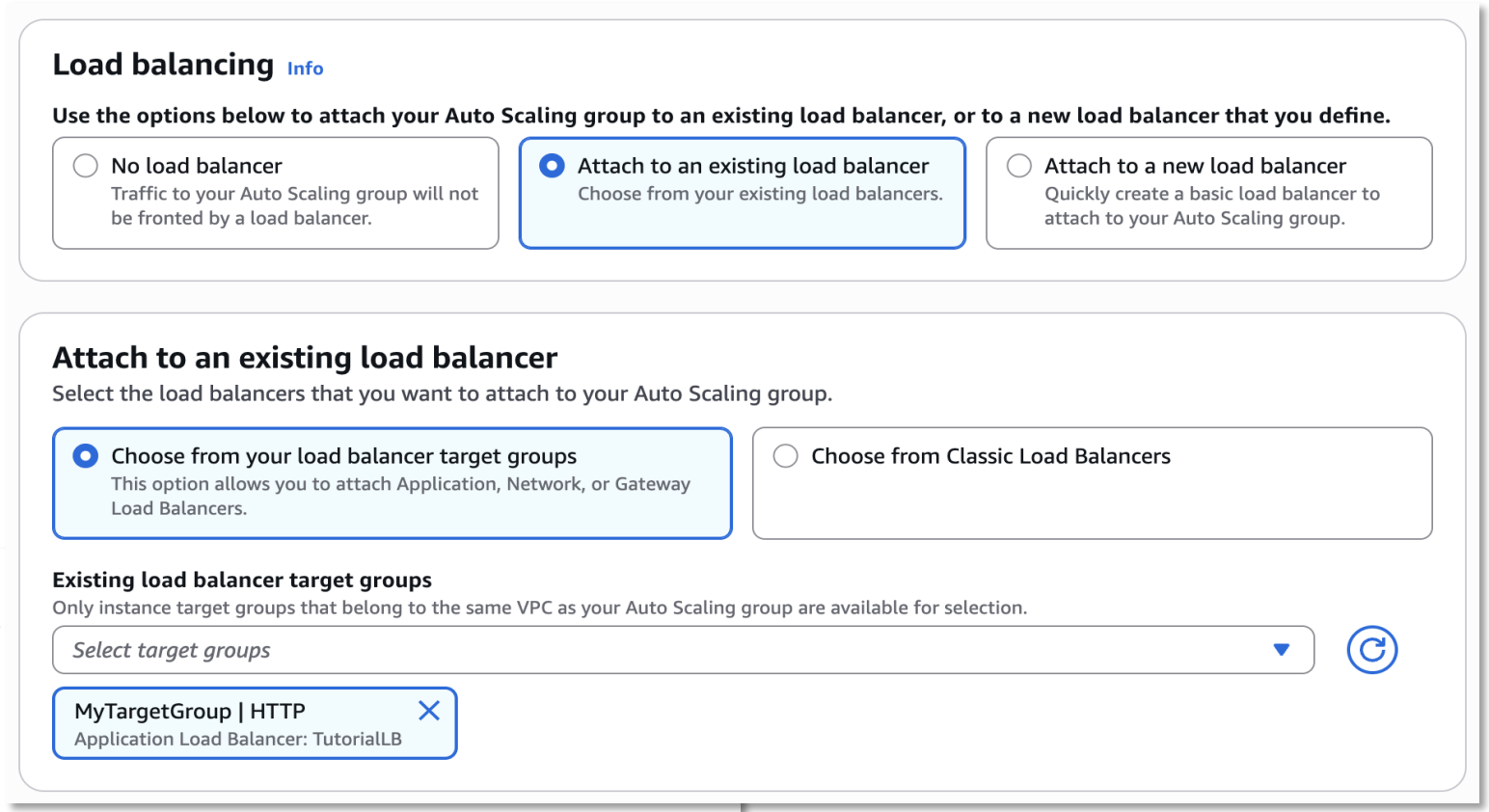
Note: so, every instance launched by Autoscaling Group starts with the exact same environment.

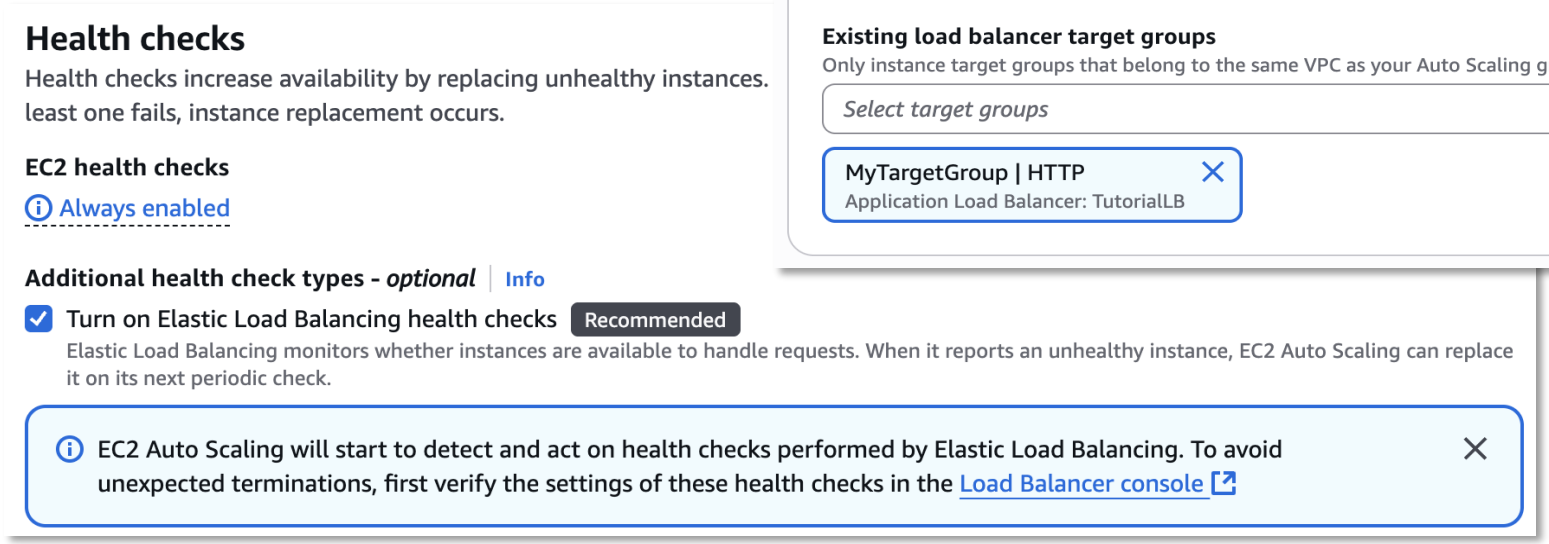
- Create a **template**
  - Image, instance type…



Step-by-step on autoscaling

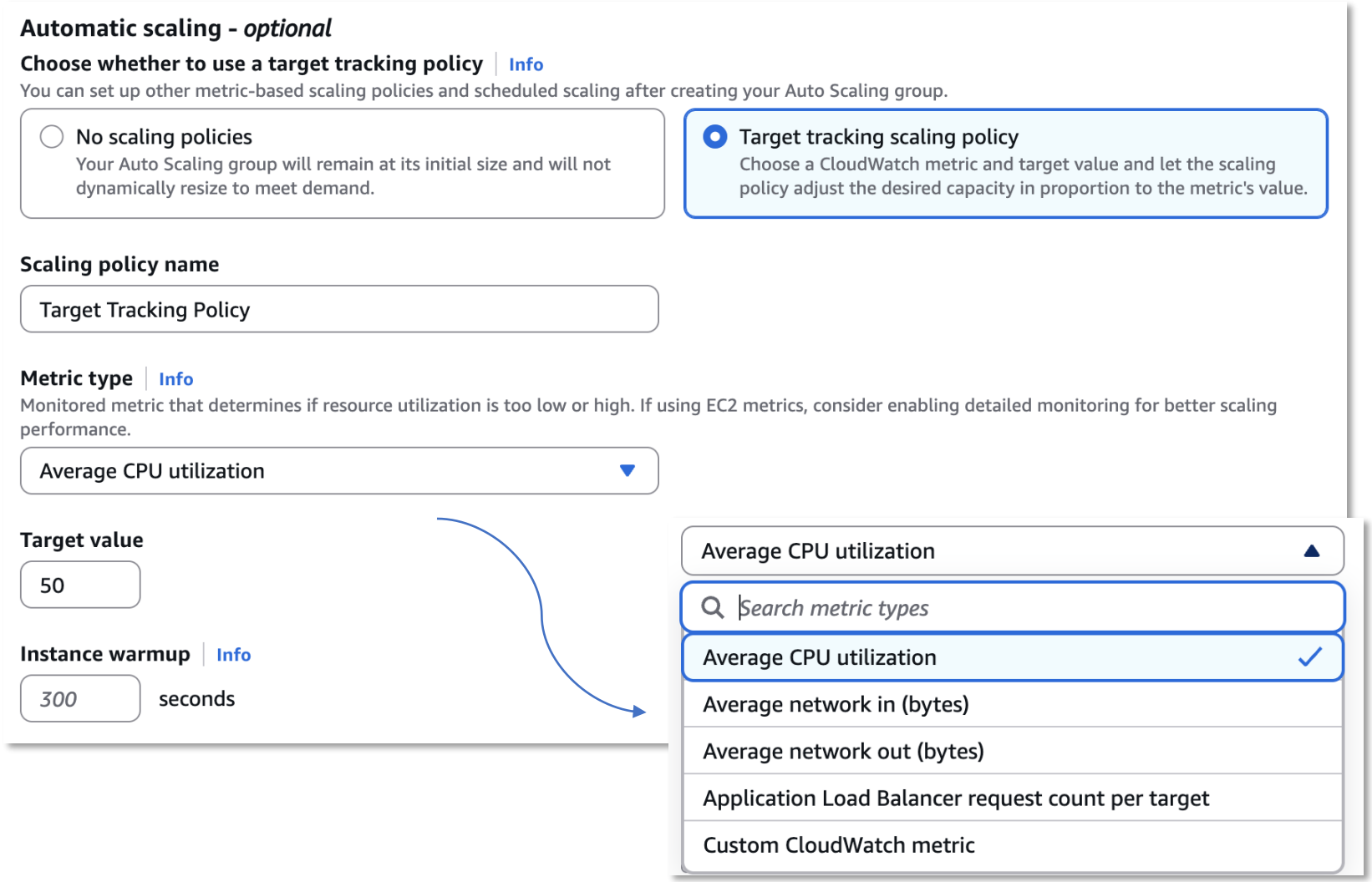
- Setup **autoscaling group**
  - Attach to a load balancer
  - Health checks
  - Group size
  - Scaling policies: metric, value





Step-by-step on autoscaling

- Setup **autoscaling group**
  - Attach to a load balancer
  - Health checks
  - Group size
  - Scaling policies

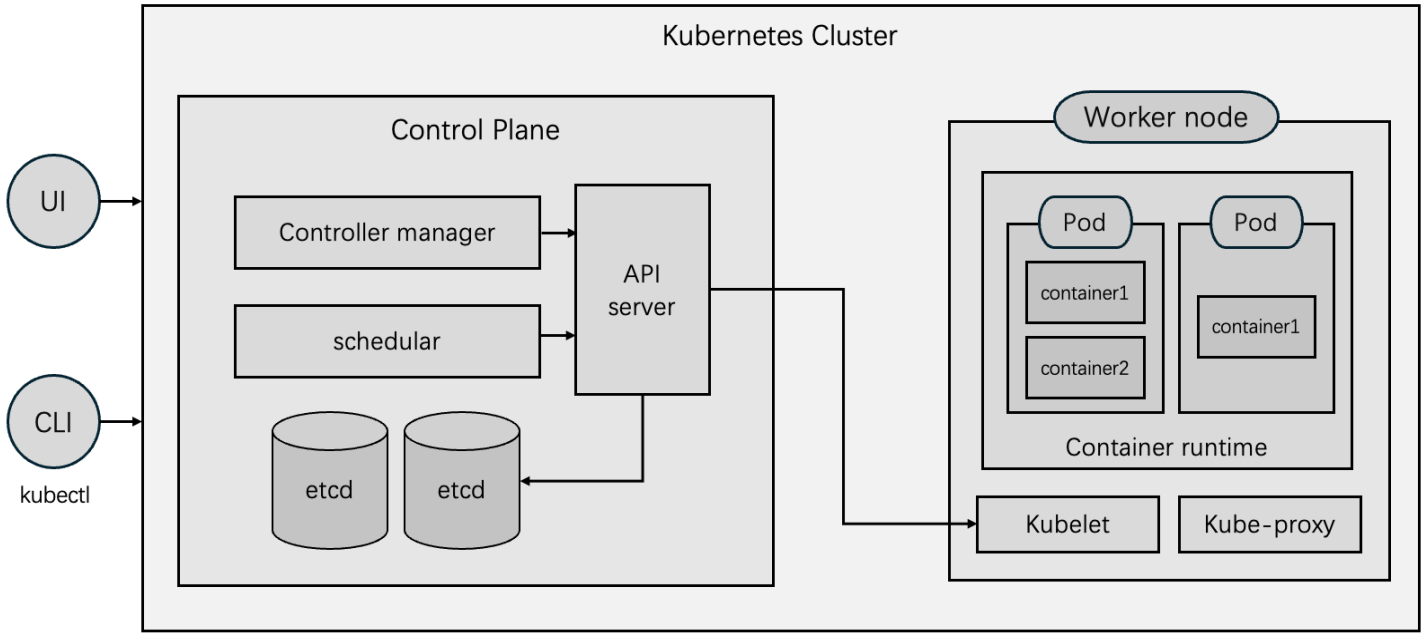


Autoscaling and Load-balancing of Kubernetes

- **Pod**: the minimum unit represents a single instance of a running process in your cluster
- **Workload**: Kubernetes control plane automatically manages Pod objects based on the specification for the workload object you defined
- **Service**: exposes a network application that is running as one or more Pods in your cluster

The Service (of type LoadBalancer) provides the load balancing
- distributes incoming traffic to all healthy Pods
- Canary release, A/B testing

The Horizontal Pod Autoscaler provides the autoscaling
- tells the workload to adjust the number of Pod replicas
- E-commerce sale event, online game rush hours



https://kubernetes.io/docs/concepts/