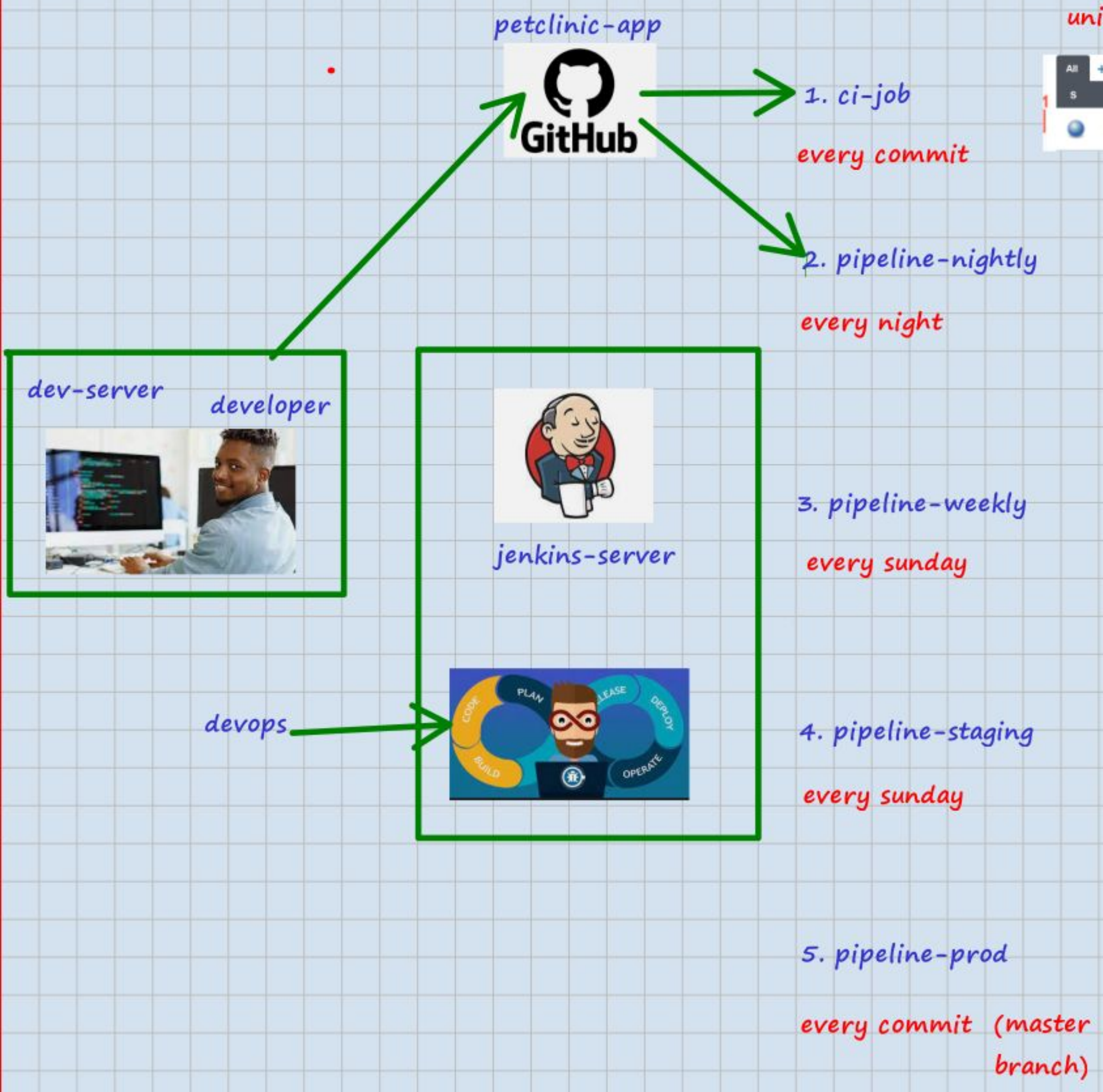
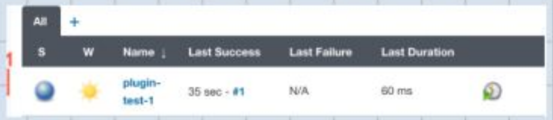


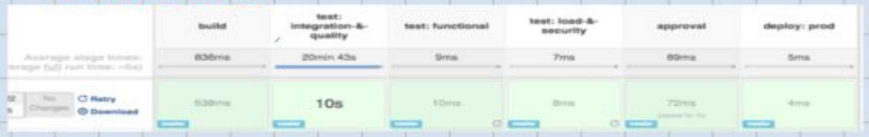
unit tests ve jacoco report



S	W	Name ↓	Last Success	Last Failure	Last Duration	
●	☀	plugin-test-1	35 sec - #1	N/A	60 ms	

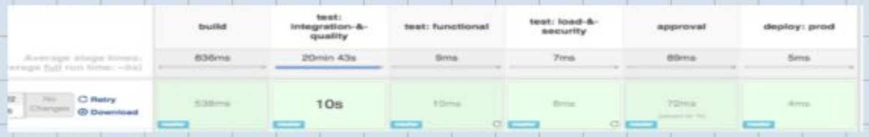
mvn clean package

functional tests



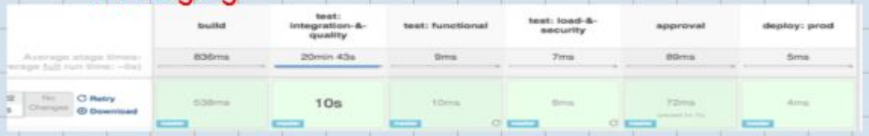
	build	test: integration-&-quality	test: functional	test: load-&-security	approval	deploy: prod
Average stage times: (Average full run time: ~5s)	836ms	20min 43s	9ms	7ms	89ms	5ms
No Changes · Retry · Download	538ms	10s	10ms	8ms	72ms	4ms

for QA testers



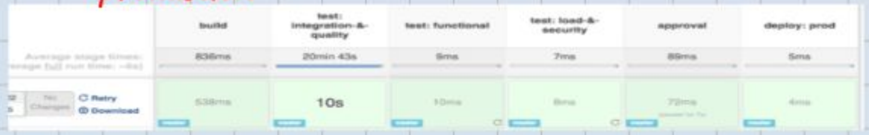
	build	test: integration-&-quality	test: functional	test: load-&-security	approval	deploy: prod
Average stage times: (Average full run time: ~5s)	836ms	20min 43s	9ms	7ms	89ms	5ms
No Changes · Retry · Download	538ms	10s	10ms	8ms	72ms	4ms

for staging



	build	test: integration-&-quality	test: functional	test: load-&-security	approval	deploy: prod
Average stage times: (Average full run time: ~5s)	836ms	20min 43s	9ms	7ms	89ms	5ms
No Changes · Retry · Download	538ms	10s	10ms	8ms	72ms	4ms

production



	build	test: integration-&-quality	test: functional	test: load-&-security	approval	deploy: prod
Average stage times: (Average full run time: ~5s)	836ms	20min 43s	9ms	7ms	89ms	5ms
No Changes · Retry · Download	538ms	10s	10ms	8ms	72ms	4ms

Pipelines to be Configured

Name	Branch	Trigger	Environment / Test Type	Tools
petclinic-ci-job	dev feature** bugfix**	**Webhook** on each commit	Unit Test	jenkins, maven, git, github, jacoco
petclinic-nightly	dev	**Cronjob** every night 11.59pm	Functional IT	jenkins, git, github, docker, docker-compose, kubernetes, ansible, maven, selenium with python, bash scripting, aws cli / ecr / cloudformation
petclinic-weekly	release	**Cronjob** every sunday 11.59pm	Manual QA	jenkins, git, github, docker, docker-compose, kubernetes, ansible, maven, bash scripting, aws cli / ecr / terraform
petclinic-staging	release	**Cronjob** every sunday 11.59pm	Staging Env.	jenkins, git, github, docker, rancher, kubernetes, maven, bash scripting, aws cli / ecr / terraform, rancher
petclinic-prod	master	**Webhook** on each commit	Production Env.	jenkins, git, github, docker, rancher, kubernetes, maven, bash scripting, aws cli / ecr / terraform, rancher

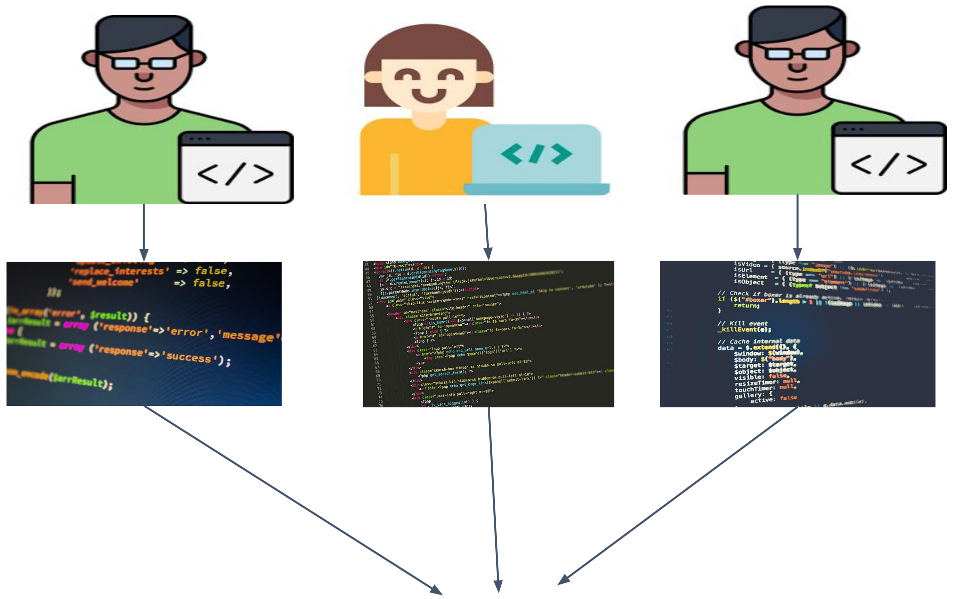
MAVEN WRAPPER

- **Maven Wrapper** is a tool that allows you to use Maven in your projects without having to install Maven itself.
- Instead of requiring users to install Maven manually, the Maven Wrapper provides a way to bootstrap Maven automatically.
- When you execute mvnw, it automatically downloads the necessary version of Maven specified for the project, along with its dependencies, if it's not already present.
- This approach ensures that everyone working on the project uses the same version of Maven, reducing potential inconsistencies and ensuring reproducibility across different development environments.

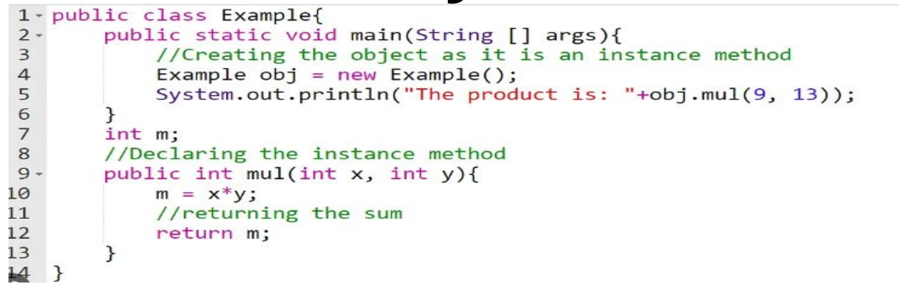
Unit Testing

- **Unit testing** is a type of software testing where individual units or components are tested.
- **Unit testing** is performed by the developer during the development cycle.
- The purpose is to validate each unit of the software code and check whether they are performing as expected.

Continuous Integration

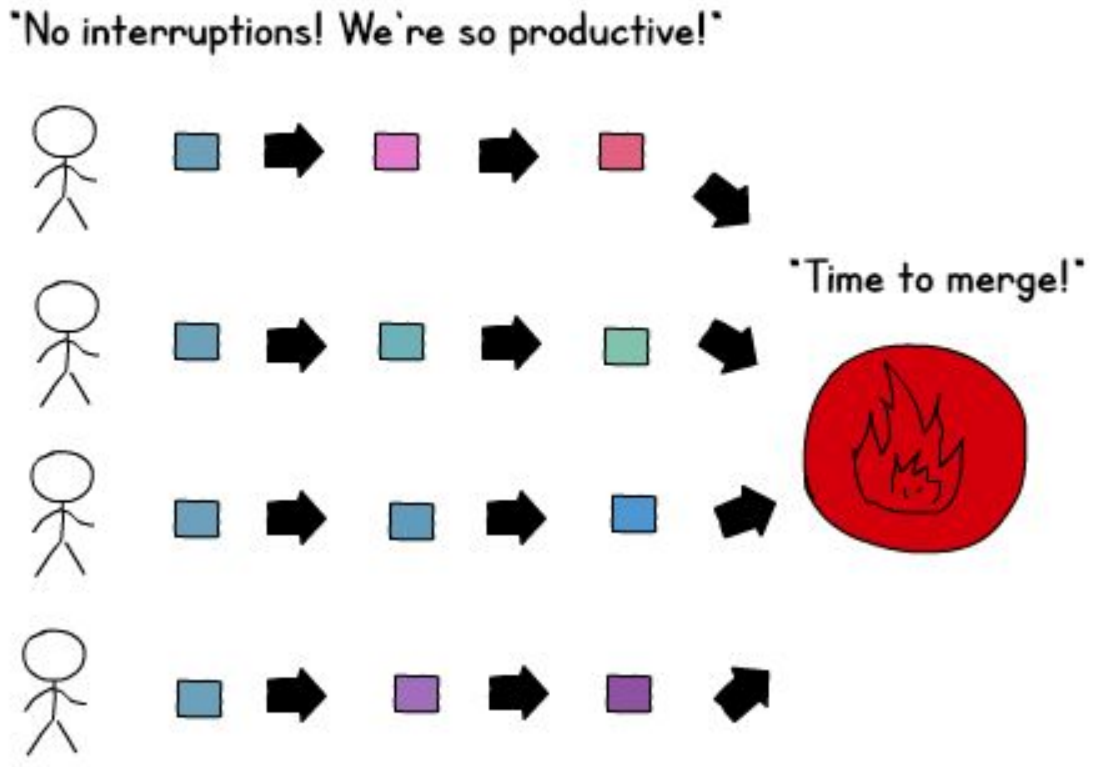


integration

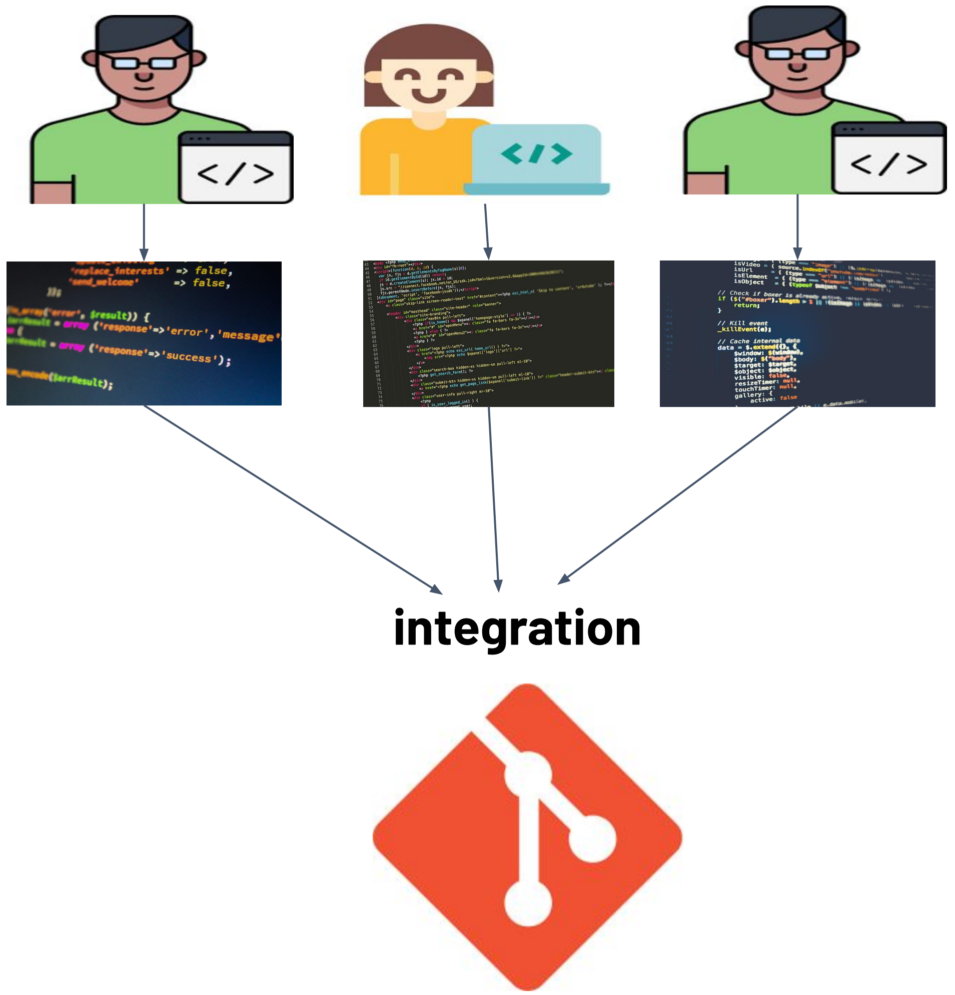


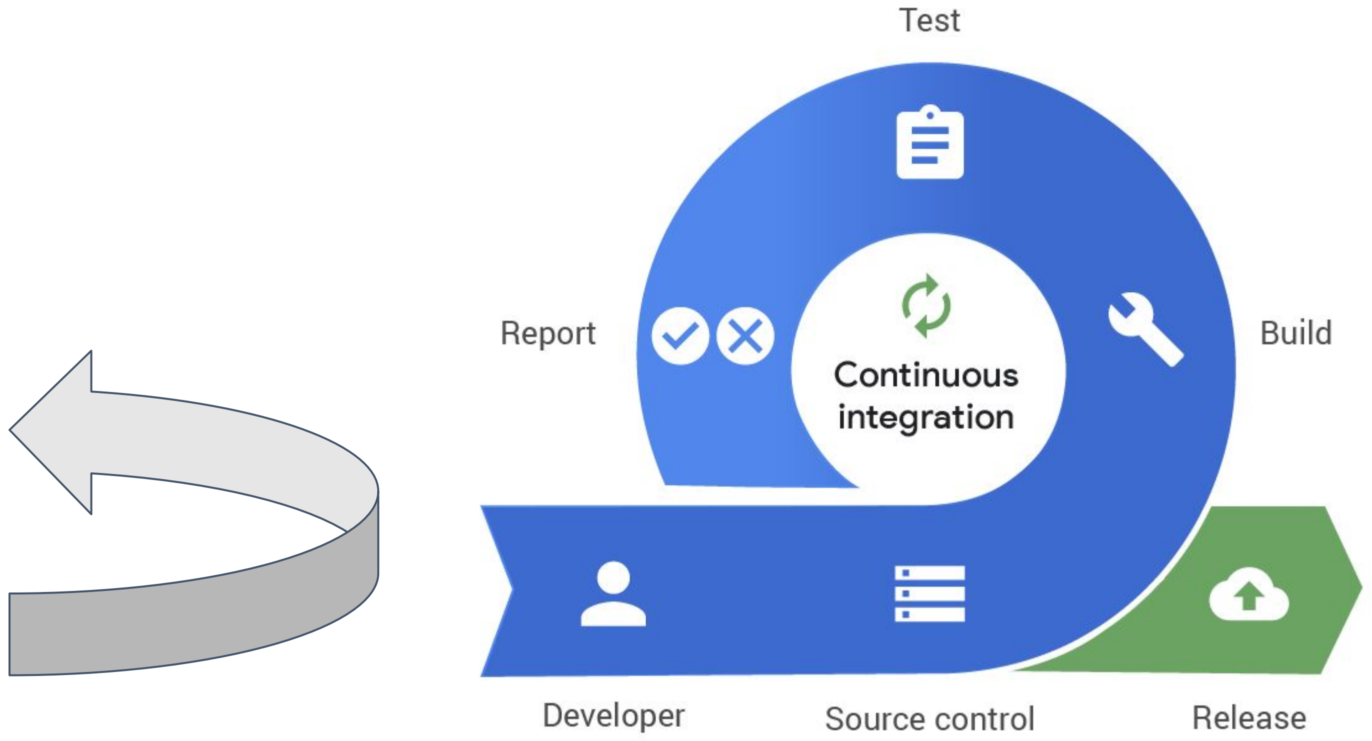
```java
public class Example{
    public static void main(String [] args){
        //Creating the object as it is an instance method
        Example obj = new Example();
        System.out.println("The product is: "+obj.mul(9, 13));
    }
    int m;
    //Declaring the instance method
    public int mul(int x, int y){
        m = x*y;
        //returning the sum
        return m;
    }
}
```

source code



Continuous Integration





CI/CD Server

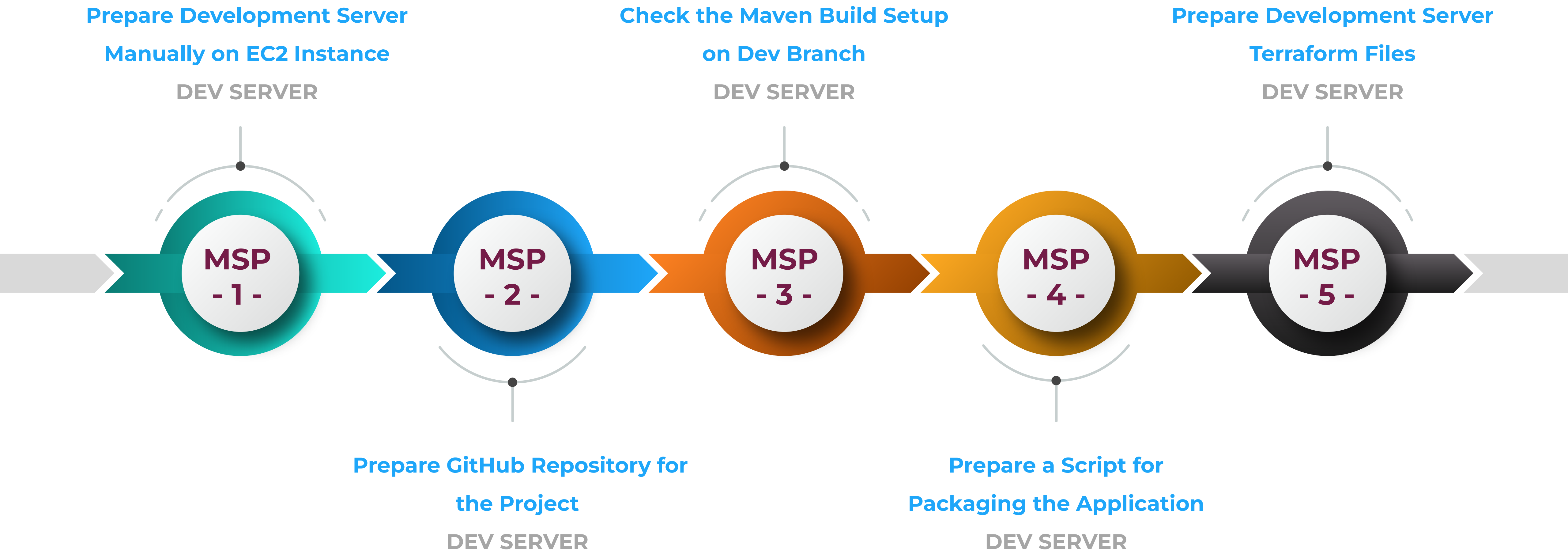
Unit Testing Vs Functional Testing

The goal of any software or application testing is to build a quality product. **Unit testing** and **Functional testing** are the foundation of the testing process.

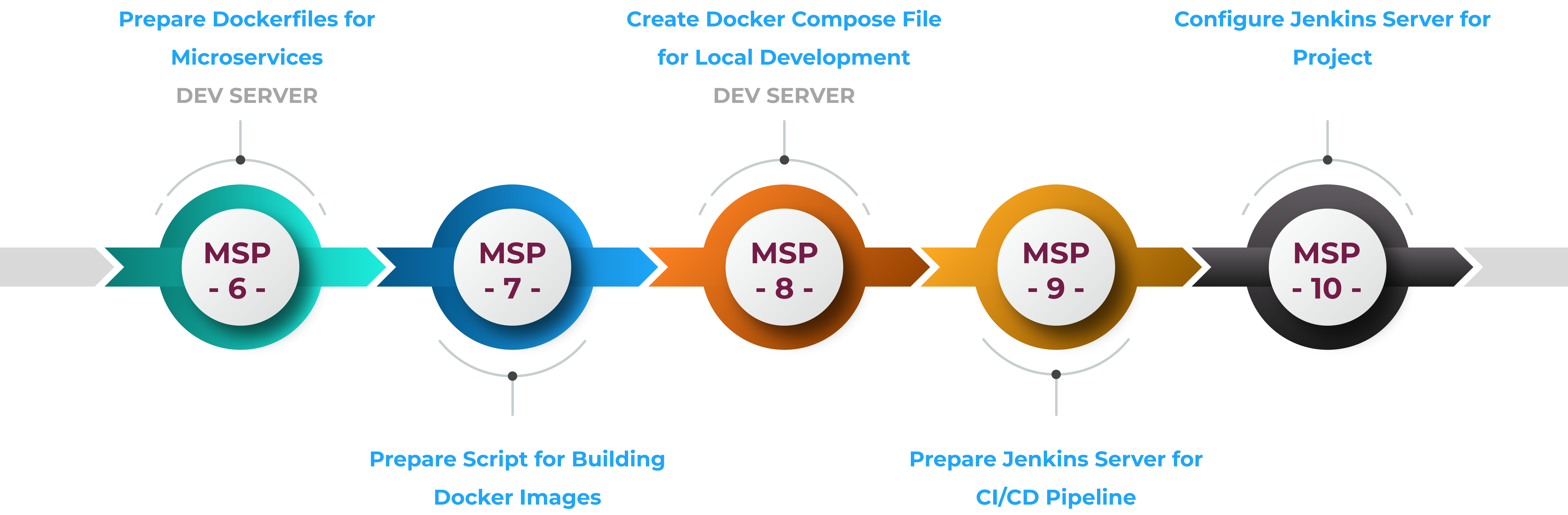
- **Unit testing** is a type of software testing where individual units or components are tested.
- **Unit testing** is performed by the developer during the development cycle.
- The purpose is to validate each unit of the software code and check whether they are performing as expected.

Unit Testing Vs Functional Testing

- **Functional Testing** tests the basic functionality of the application.
- It checks if the application runs as per the functional requirements.
- **Functional testing** is performed by the tester during the level of system testing.
- In functional testing, a tester is not worried about the core code, instead they need to verify the output based on the user requirements with the expected output.



DEVOPS CAPSTONE PROJECT


Prepare Dockerfiles for Microservices
DEV SERVER
MSP - 6 -
Prepare Script for Building Docker Images
MSP - 7 -
Create Docker Compose File for Local Development
DEV SERVER
MSP - 8 -
Prepare Jenkins Server for CI/CD Pipeline
MSP - 9 -
Configure Jenkins Server for Project
MSP - 10 -

DEVOPS CAPSTONE PROJECT

Setup Unit Tests and Configure Code Coverage Report

UNIT TESTS

- POM.XML Jacoco Plugin

Prepare and Implement Selenium Tests

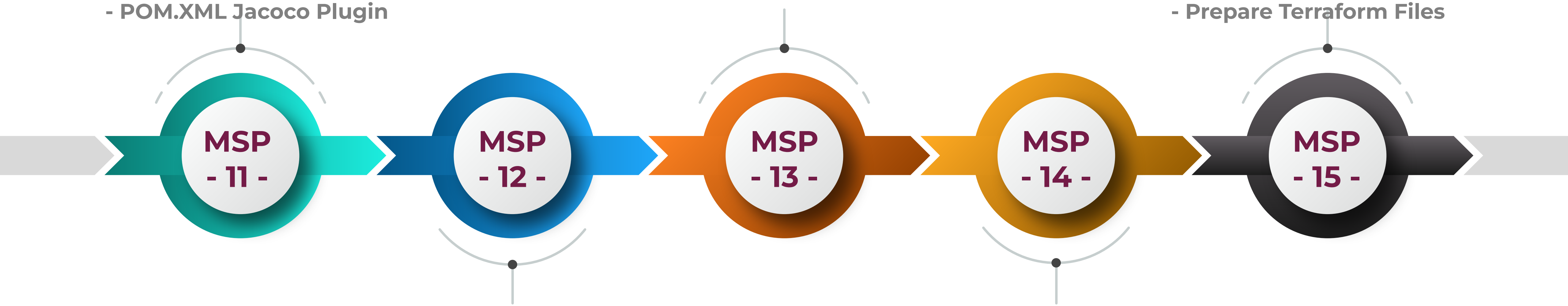
NIGHTLY - FUNCTIONAL TESTS

Create a QA Automation Environment with Kubernetes

NIGHTLY - FUNCTIONAL TESTS

- Prepare Policies

- Prepare Terraform Files



Prepare Continuous Integration (CI) Pipeline

UNIT TESTS

- CI Job for Jacoco

Create ECR Registry for Dev Manually

NIGHTLY - FUNCTIONAL TESTS

- CI Job: Create ECR Registry

DEVOPS CAPSTONE PROJECT

Prepare a QA Automation Pipeline for Nightly Builds

QA TESTS

- Shell Scripts: Prepare .sh files for Pipeline
- CI Job: Test scripts on Jenkins (Freestyle)
- Selenium Tests: Make dummy tests
- CI Pipeline: Jenkinsfile + QA Test Pipeline

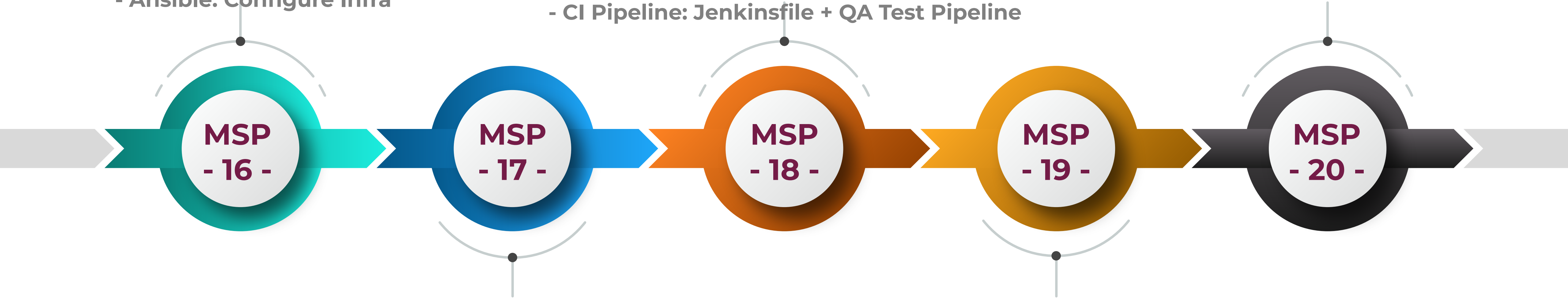
Create a QA Automation Environment with Kubernetes

NIGHTLY - FUNCTIONAL TESTS

- Terraform: Create Infra
- Ansible: Configure Infra

Prepare Build Scripts for QA Environment

QA TESTS with EKS



Prepare Petlinic Kubernetes YAML Files

NIGHTLY - FUNCTIONAL TESTS

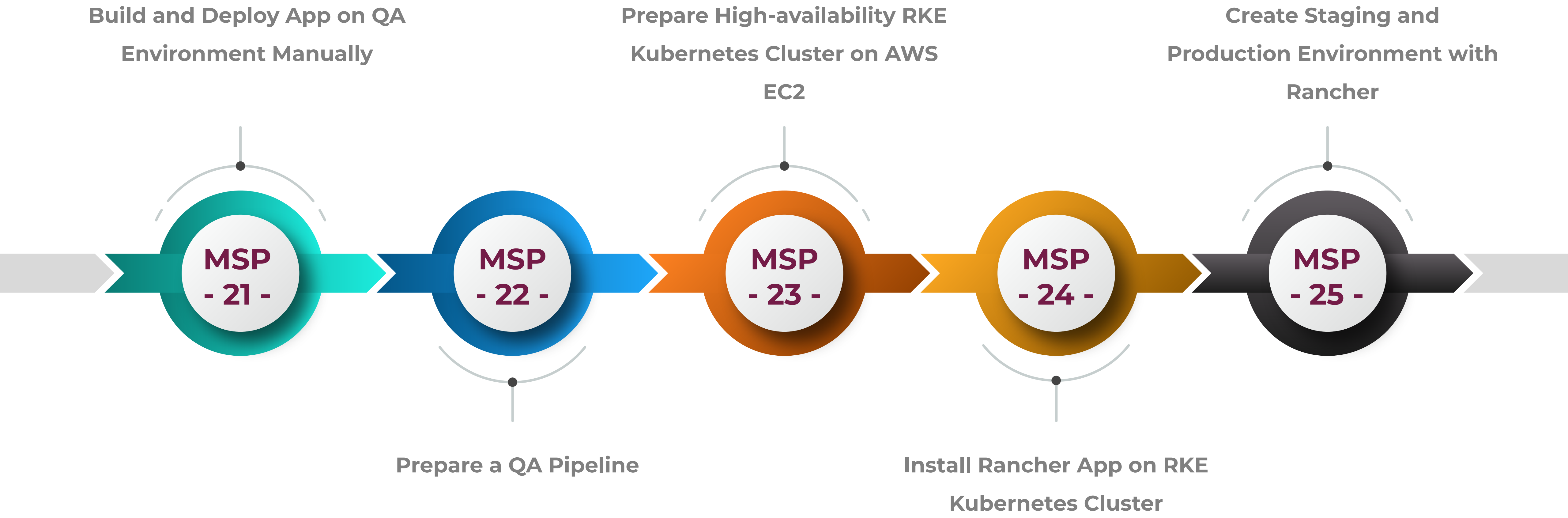
- Kompose: YAML Transformation
- Helm: Prepare Chart + Send Repo

Create a QA Environment on EKS Cluster

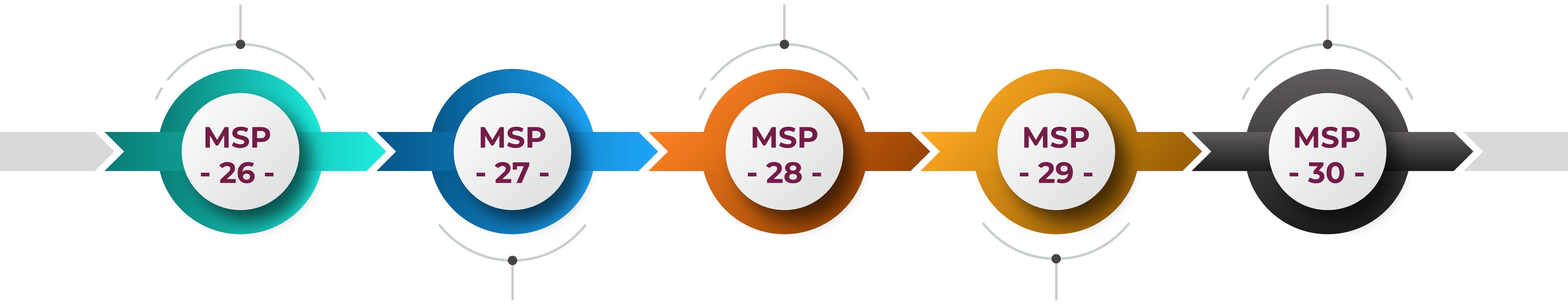
NIGHTLY - FUNCTIONAL TESTS

- CI Pipeline on EKS: Jenkinsfile + QA Test Pipeline

DEVOPS CAPSTONE PROJECT


Build and Deploy App on QA Environment Manually
Prepare High-availability RKE Kubernetes Cluster on AWS EC2
Create Staging and Production Environment with Rancher
MSP - 21 -
MSP - 22 -
MSP - 23 -
MSP - 24 -
MSP - 25 -
Prepare a QA Pipeline
Install Rancher App on RKE Kubernetes Cluster

DEVOPS CAPSTONE PROJECT



Prepare Nexus Server

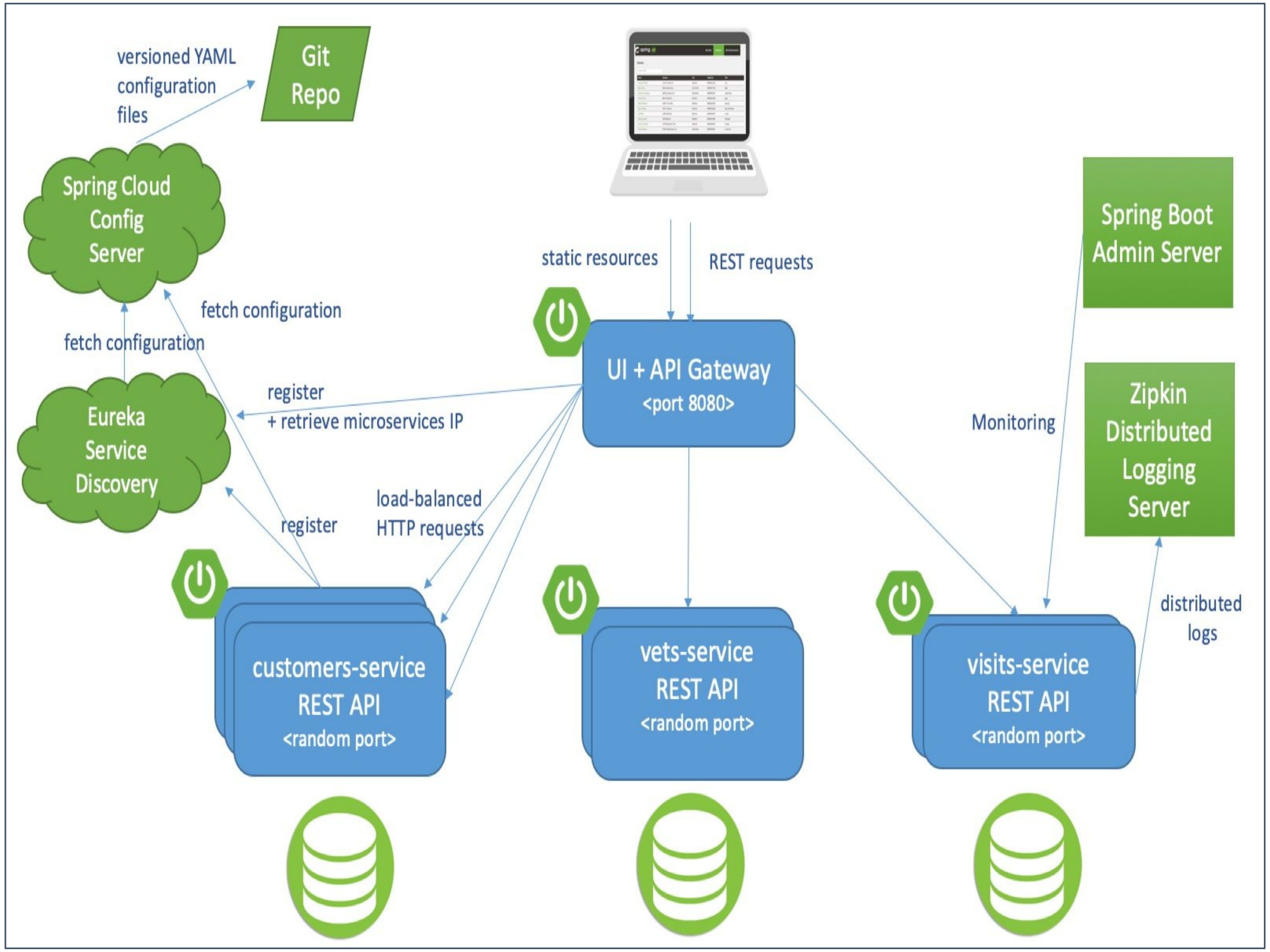
Prepare a Staging Pipeline

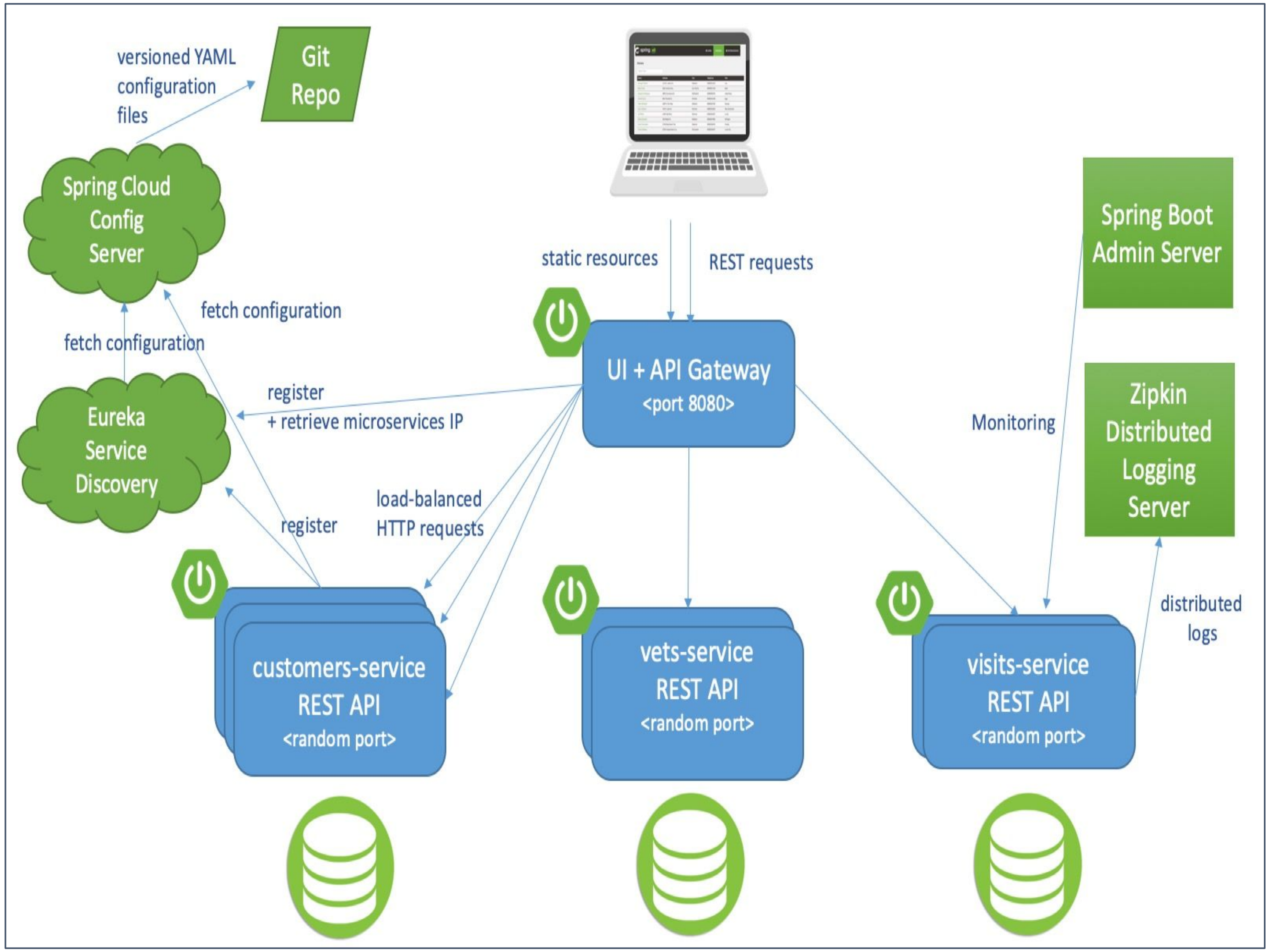
Prepare a Production Pipeline

Setting Domain Name and TLS for Production Pipeline with Route 53

Monitoring with Prometheus and Grafana

DEVOPS CAPSTONE PROJECT

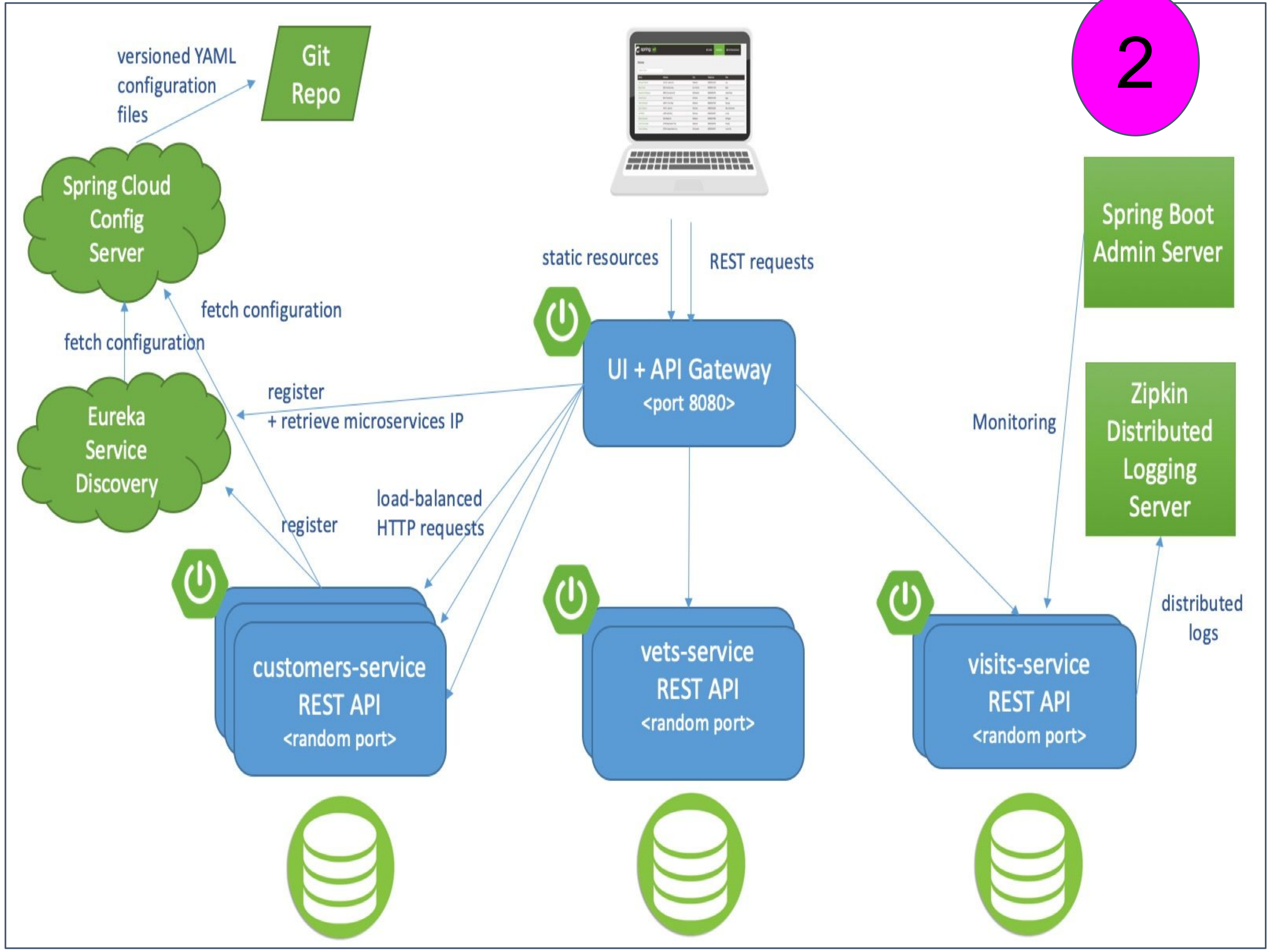

versioned YAML configuration files
Git Repo
Spring Cloud Config Server
fetch configuration
fetch configuration
Eureka Service Discovery
register + retrieve microservices IP
register
load-balanced HTTP requests
static resources
REST requests
UI + API Gateway <port 8080>
Spring Boot Admin Server
Monitoring
Zipkin Distributed Logging Server
distributed logs
customers-service REST API <random port>
vets-service REST API <random port>
visits-service REST API <random port>


versioned YAML configuration files
Git Repo
Spring Cloud Config Server
fetch configuration
fetch configuration
Eureka Service Discovery
register + retrieve microservices IP
register
load-balanced HTTP requests
static resources
REST requests
UI + API Gateway <port 8080>
Spring Boot Admin Server
Monitoring
Zipkin Distributed Logging Server
distributed logs
customers-service REST API <random port>
vets-service REST API <random port>
visits-service REST API <random port>

1

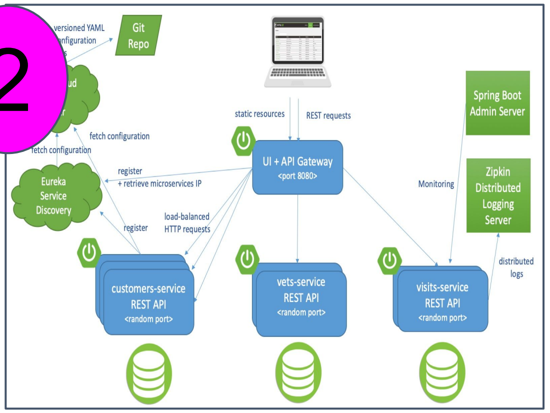
3

2


versioned YAML configuration files
Git Repo
Spring Cloud Config Server
fetch configuration
fetch configuration
Eureka Service Discovery
register + retrieve microservices IP
register
load-balanced HTTP requests
static resources
REST requests
UI + API Gateway <port 8080>
Spring Boot Admin Server
Monitoring
Zipkin Distributed Logging Server
distributed logs
customers-service REST API <random port>
vets-service REST API <random port>
visits-service REST API <random port>

1

2



3

Create infrastructure

- Launch instances with terraform ***
- Setup Kubernetes cluster with ansible ***

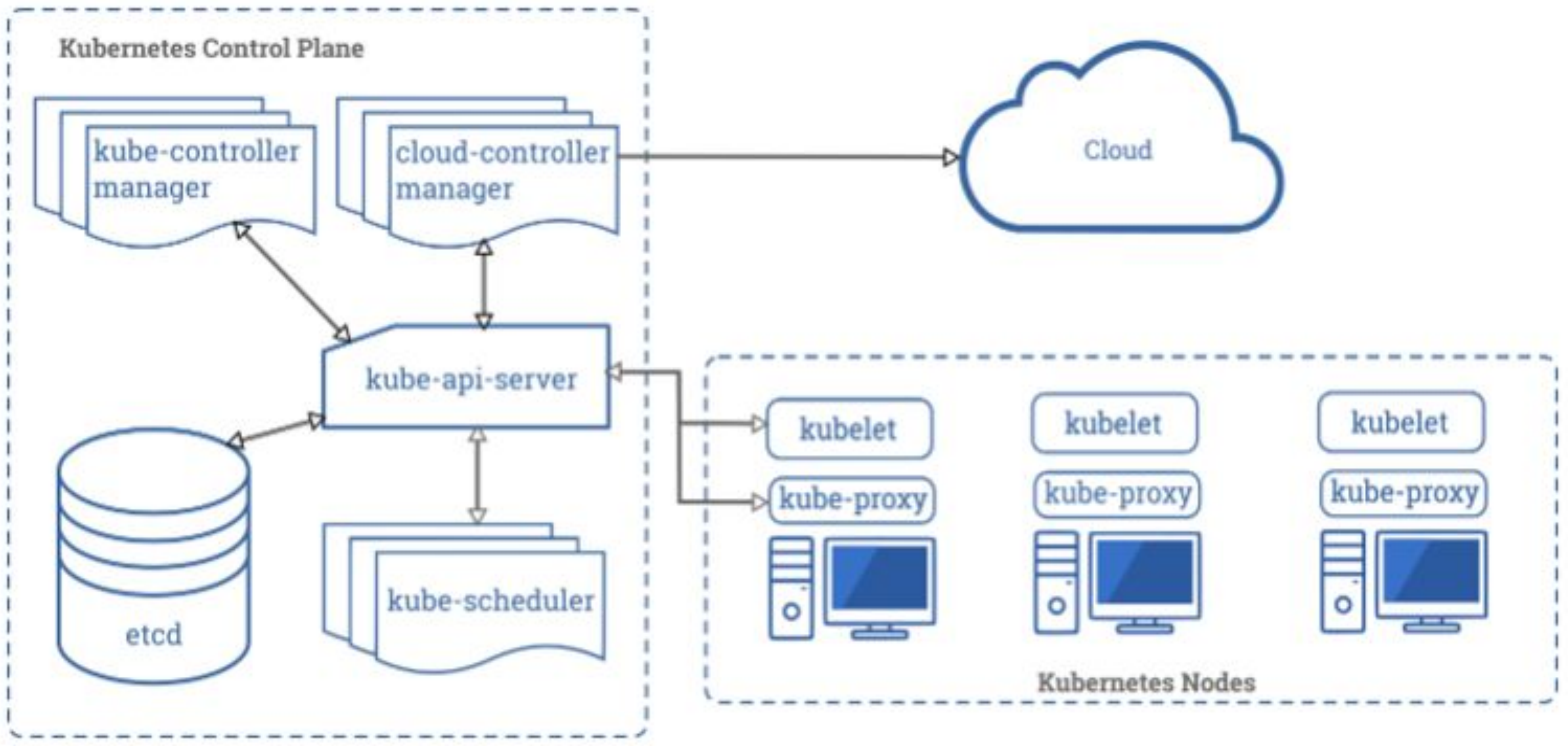
Create application and deploy to kubernetes cluster

- Create ECR repo ***
- Prepare Docker Images ***
- Push Images to ECR Repo ***
- Create Kubernetes manifest files ***
- Create helm charts ***
- Deploy application on Kubernetes cluster with helm ***

Run Functional test with selenium ***

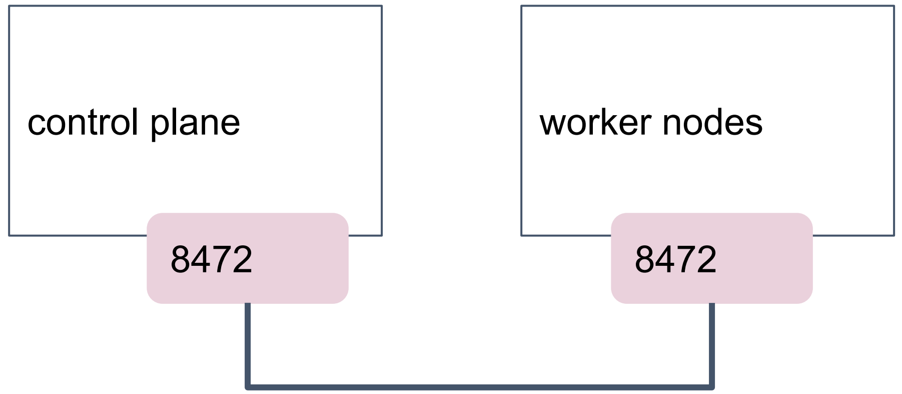
Name	Project	environment	role
kube-master	tera-kube-ans	dev	master
worker-1	tera-kube-ans	dev	worker
worker-2	tera-kube-ans	dev	worker

Control Plane Components



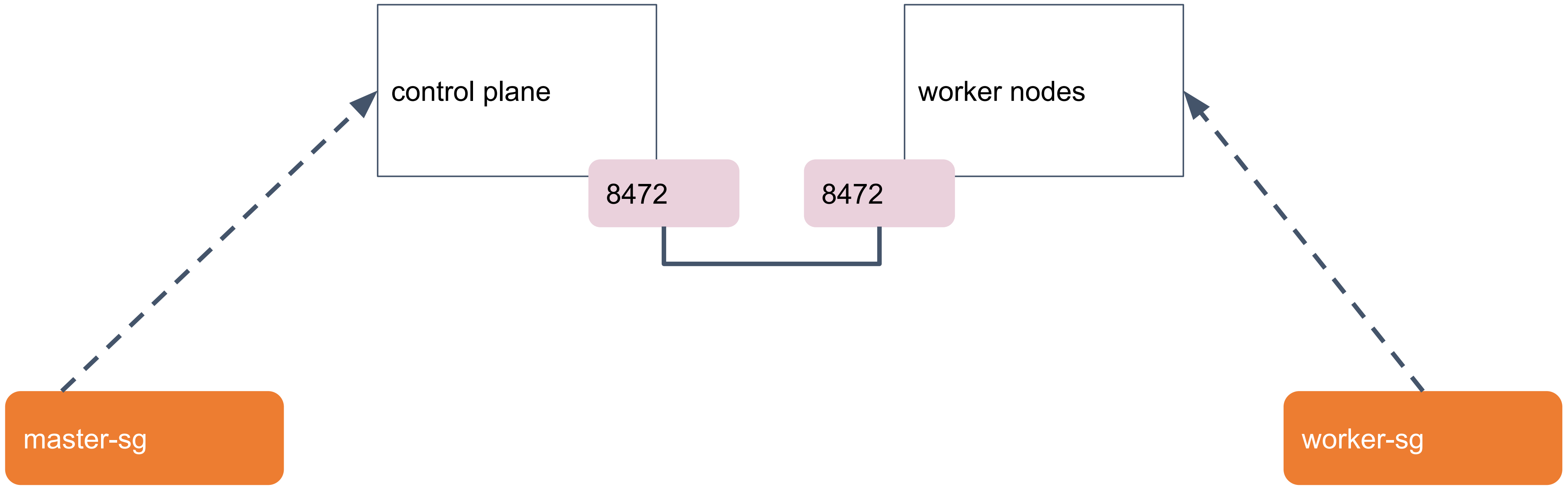
Control plane

Protocol	Direction	Port Range	Purpose	Used By
TCP	Inbound	6443	Kubernetes API server	All
TCP	Inbound	2379-2380	etcd server client API	kube-apiserver, etcd
TCP	Inbound	10250	Kubelet API	Self, Control plane
TCP	Inbound	10259	kube-scheduler	Self
TCP	Inbound	10257	kube-controller-manager	Self



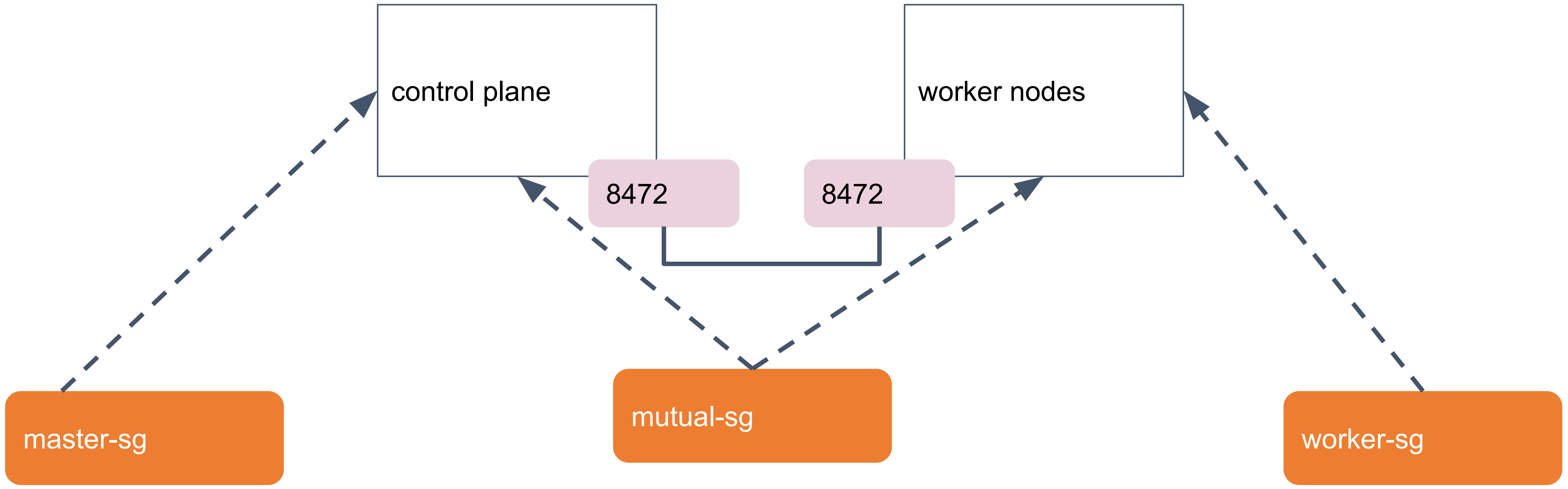
Worker node(s)

Protocol	Direction	Port Range	Purpose	Used By
TCP	Inbound	10250	Kubelet API	Self, Control plane
TCP	Inbound	30000-32767	NodePort Services†	All



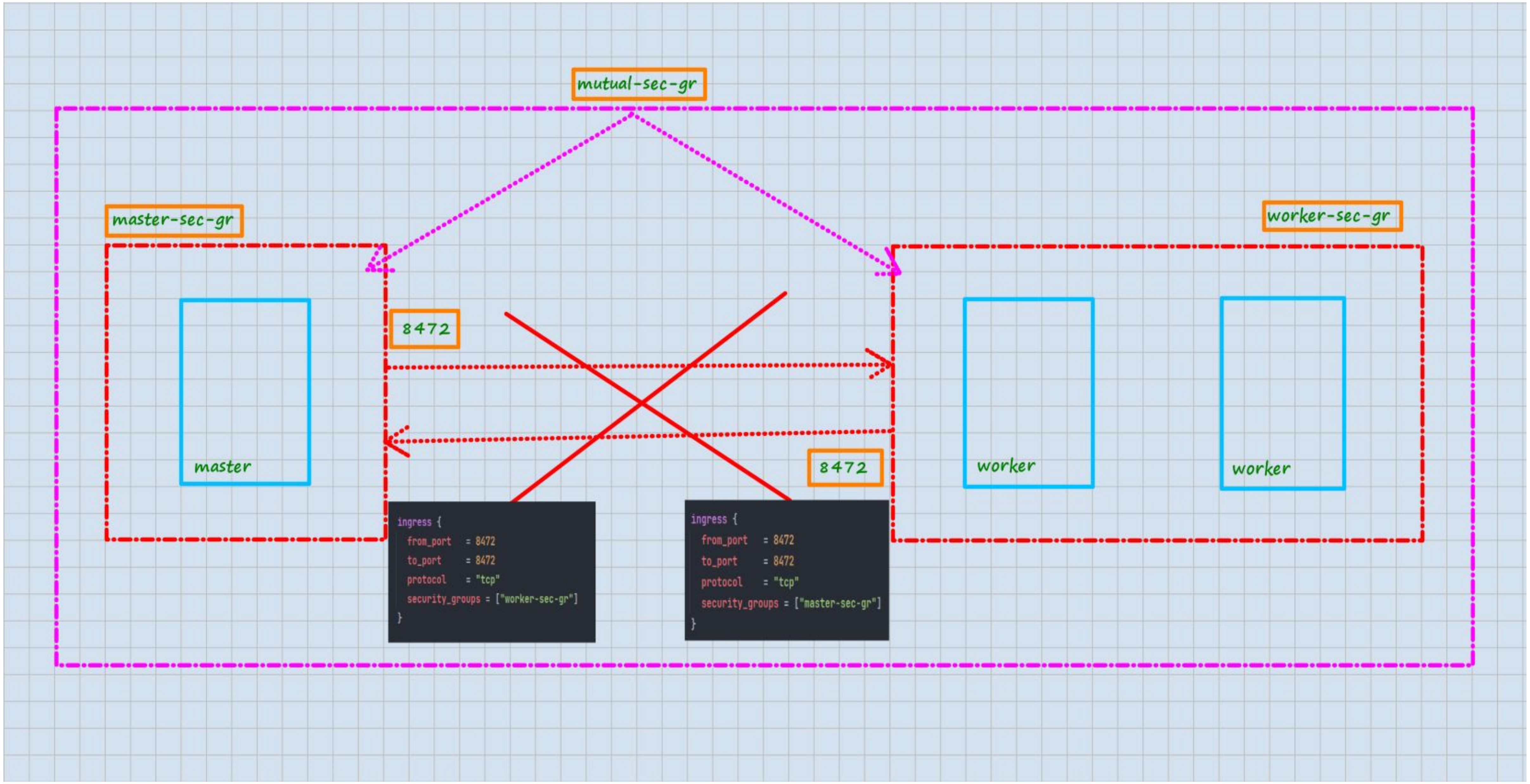
```
ingress {
    protocol = "udp"
    from_port = 8472
    to_port = 8472
      security_groups = [aws_security_group.worker-sg.id]
```

```
ingress {
    protocol = "udp"
    from_port = 8472
    to_port = 8472
      security_groups = [aws_security_group.master-sg.id]
```



```
ingress {
    protocol = "tcp"
    from_port = 8472
    to_port = 8472
      security_groups = [aws_security_group.mutual-sg.id]
```

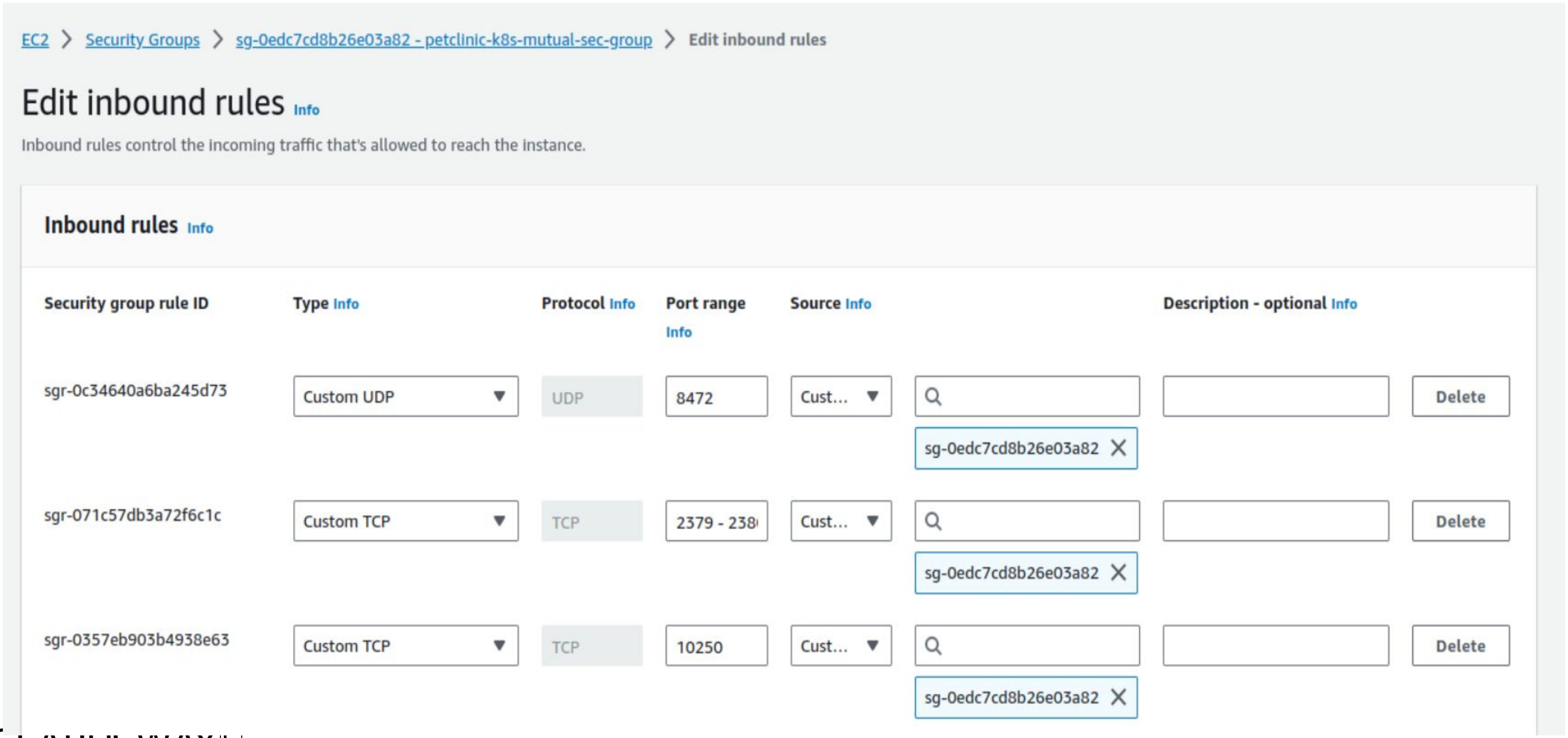
```
ingress {
    protocol = "tcp"
    from_port = 8472
    to_port = 8472
      security_groups = [aws_security_group.mutual-sg.id]
```



```
ingress {
  from_port       = 8472
  to_port         = 8472
  protocol        = "tcp"
  security_groups = ["worker-sec-gr"]
}
```

```
ingress {
  from_port       = 8472
  to_port         = 8472
  protocol        = "tcp"
  security_groups = ["master-sec-gr"]
}
```

mutual-sg



EC2 > Security Groups > sg-0edc7cd8b26e03a82 - petclinic-k8s-mutual-sec-group > Edit inbound rules

Edit inbound rules Info

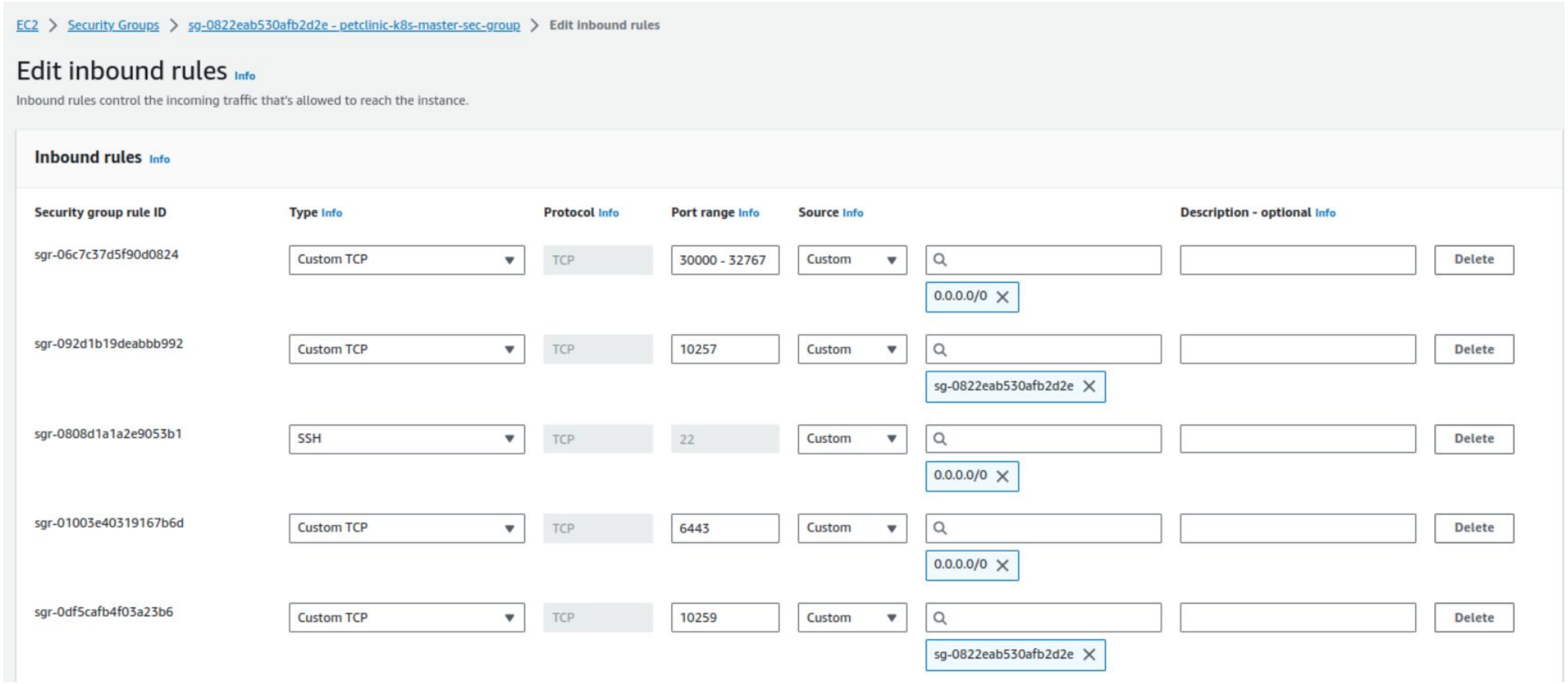
Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-0c34640a6ba245d73	Custom UDP	UDP	8472	Cust...	sg-0edc7cd8b26e03a82		Delete
sgr-071c57db3a72f6c1c	Custom TCP	TCP	2379 - 238	Cust...	sg-0edc7cd8b26e03a82		Delete
sgr-0357eb903b4938e63	Custom TCP	TCP	10250	Cust...	sg-0edc7cd8b26e03a82		Delete

CLARUSWAY©
WAY TO REINVENT YOURSELF

master-sg


EC2 > Security Groups > sg-0822eab530afb2d2e - petclinic-k8s-master-sec-group > Edit inbound rules

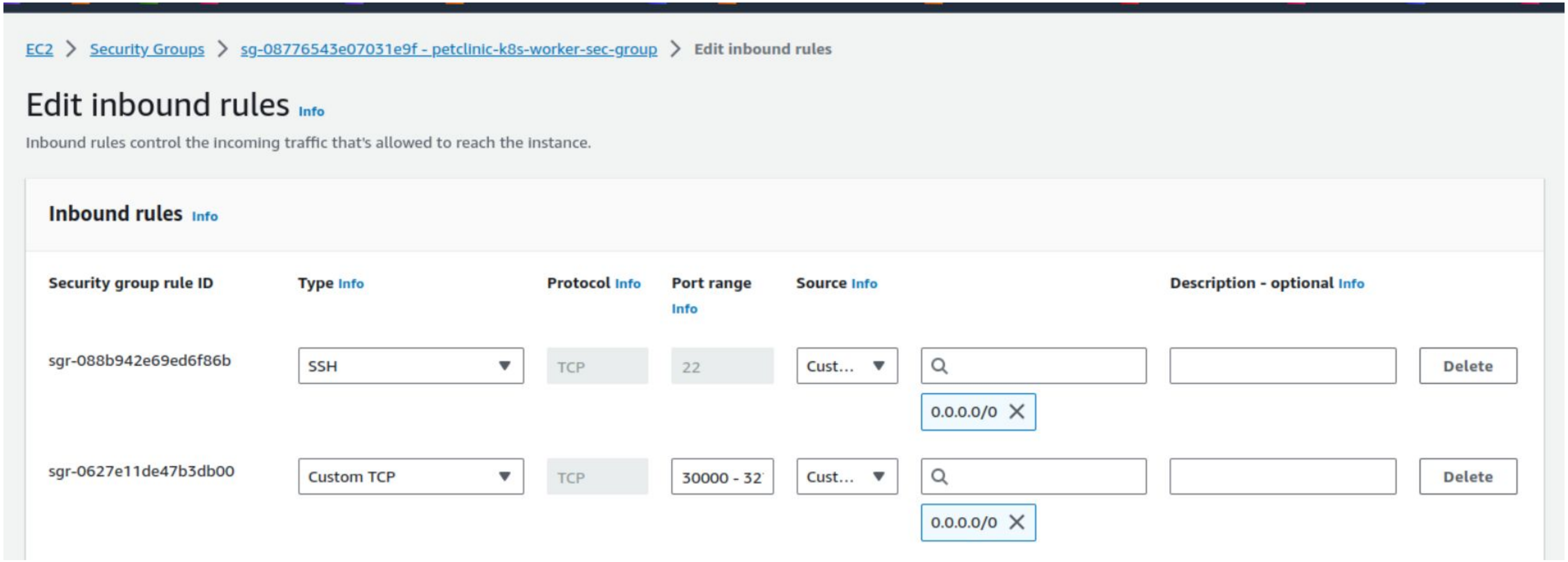
Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-06c7c37d5f90d0824	Custom TCP	TCP	30000 - 32767	Custom	0.0.0.0/0		Delete
sgr-092d1b19deabbb992	Custom TCP	TCP	10257	Custom	sg-0822eab530afb2d2e		Delete
sgr-0808d1a1a2e9053b1	SSH	TCP	22	Custom	0.0.0.0/0		Delete
sgr-01003e40319167b6d	Custom TCP	TCP	6443	Custom	0.0.0.0/0		Delete
sgr-0df5cafb4f03a23b6	Custom TCP	TCP	10259	Custom	sg-0822eab530afb2d2e		Delete

worker-sg

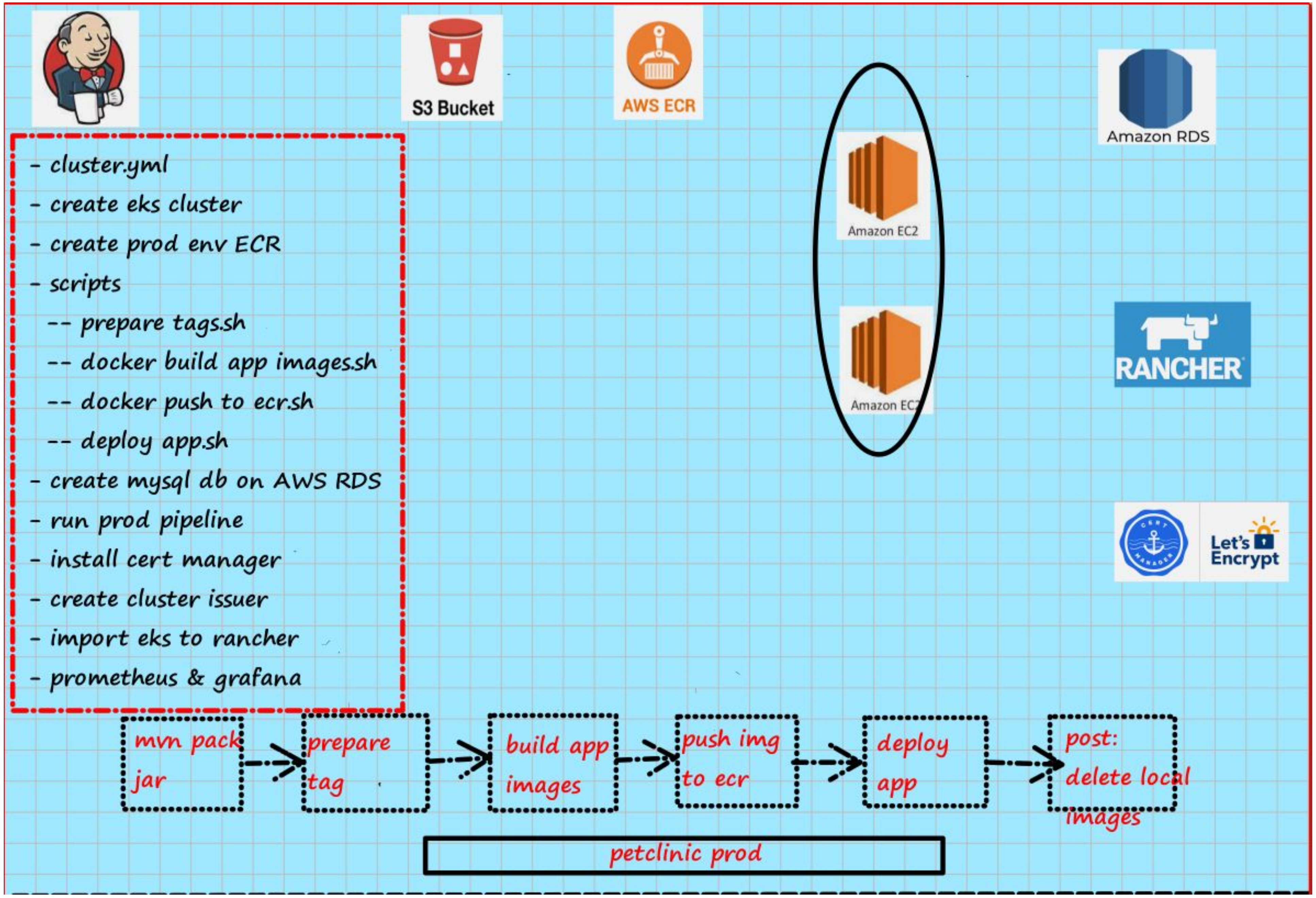

EC2 > Security Groups > sg-08776543e07031e9f - petclinic-k8s-worker-sec-group > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

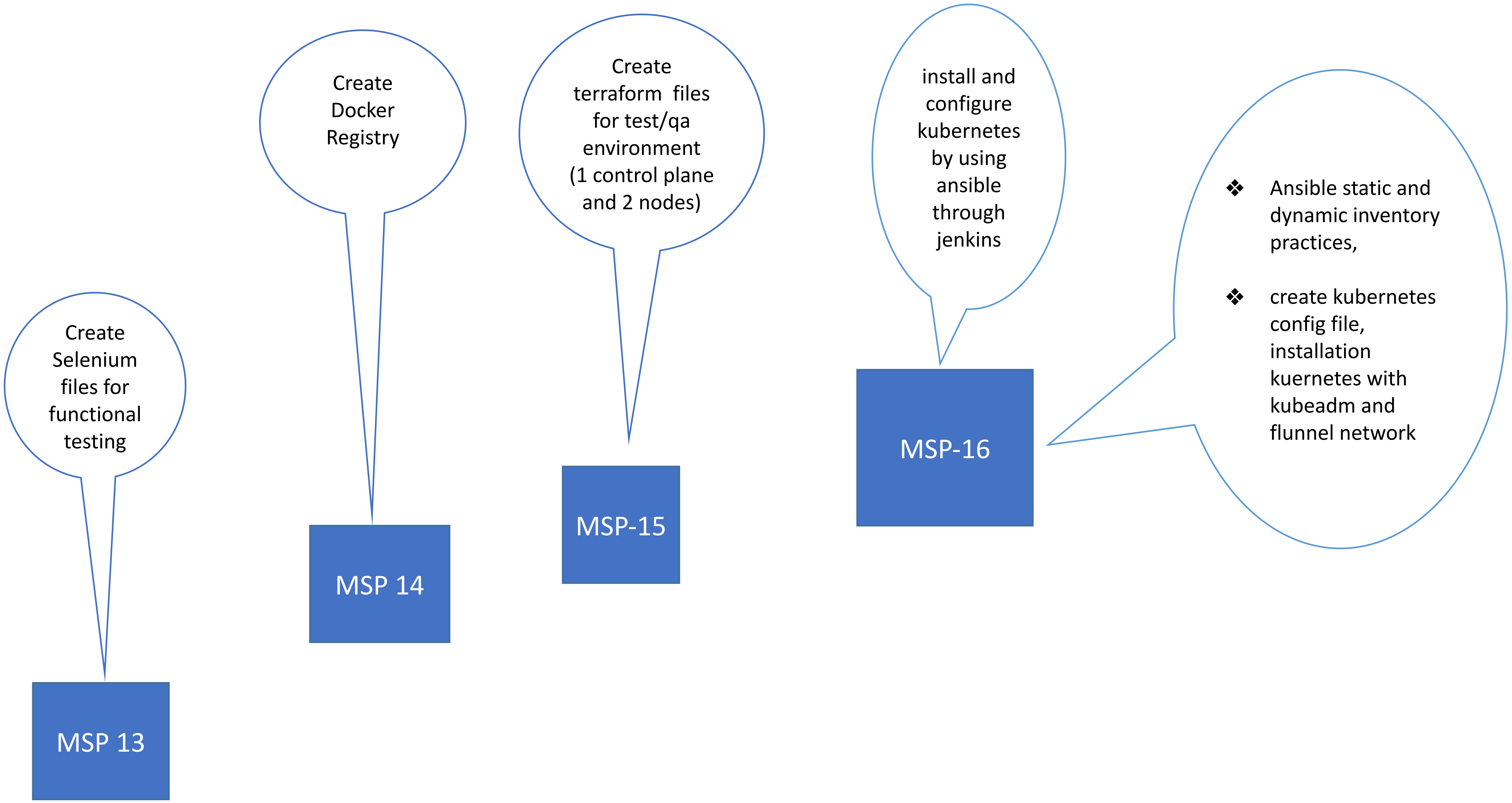
Inbound rules Info

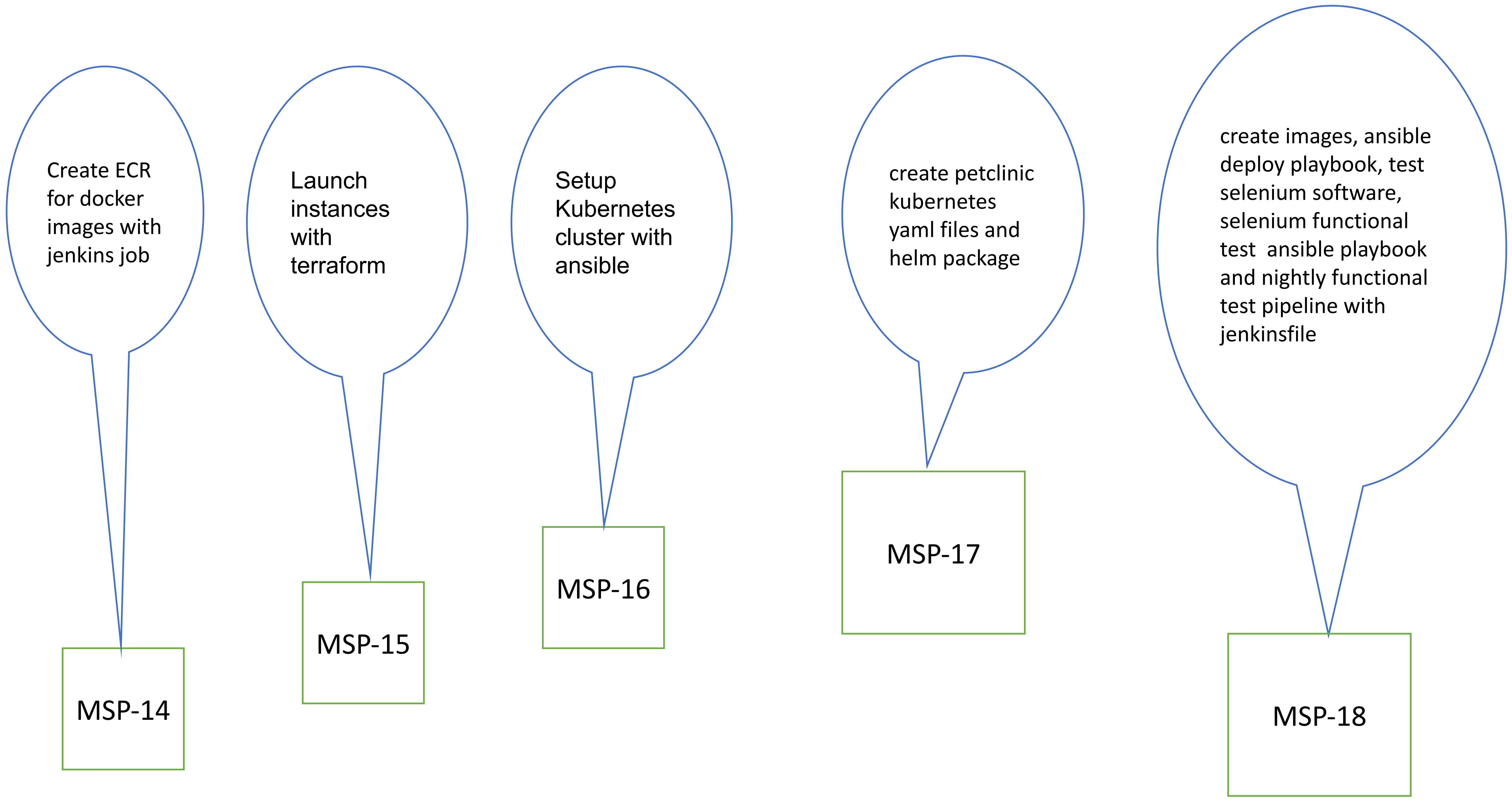
Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-088b942e69ed6f86b	SSH	TCP	22	Cust...	0.0.0.0/0		Delete
sgr-0627e11de47b3db00	Custom TCP	TCP	30000 - 32	Cust...	0.0.0.0/0		Delete


- cluster.yml
- create eks cluster
- create prod env ECR
- scripts
  -- prepare tags.sh
  -- docker build app images.sh
  -- docker push to ecr.sh
  -- deploy app.sh
- create mysql db on AWS RDS
- run prod pipeline
- install cert manager
- create cluster issuer
- import eks to rancher
- prometheus & grafana

THANKS!

Any questions?


Create Selenium files for functional testing
MSP 13
Create Docker Registry
MSP 14
Create terraform files for test/qa environment (1 control plane and 2 nodes)
MSP-15
install and configure kubernetes by using ansible through jenkins
MSP-16
❖ Ansible static and dynamic inventory practices,
❖ create kubernetes config file, installation kuernetes with kubeadm and flunnel network

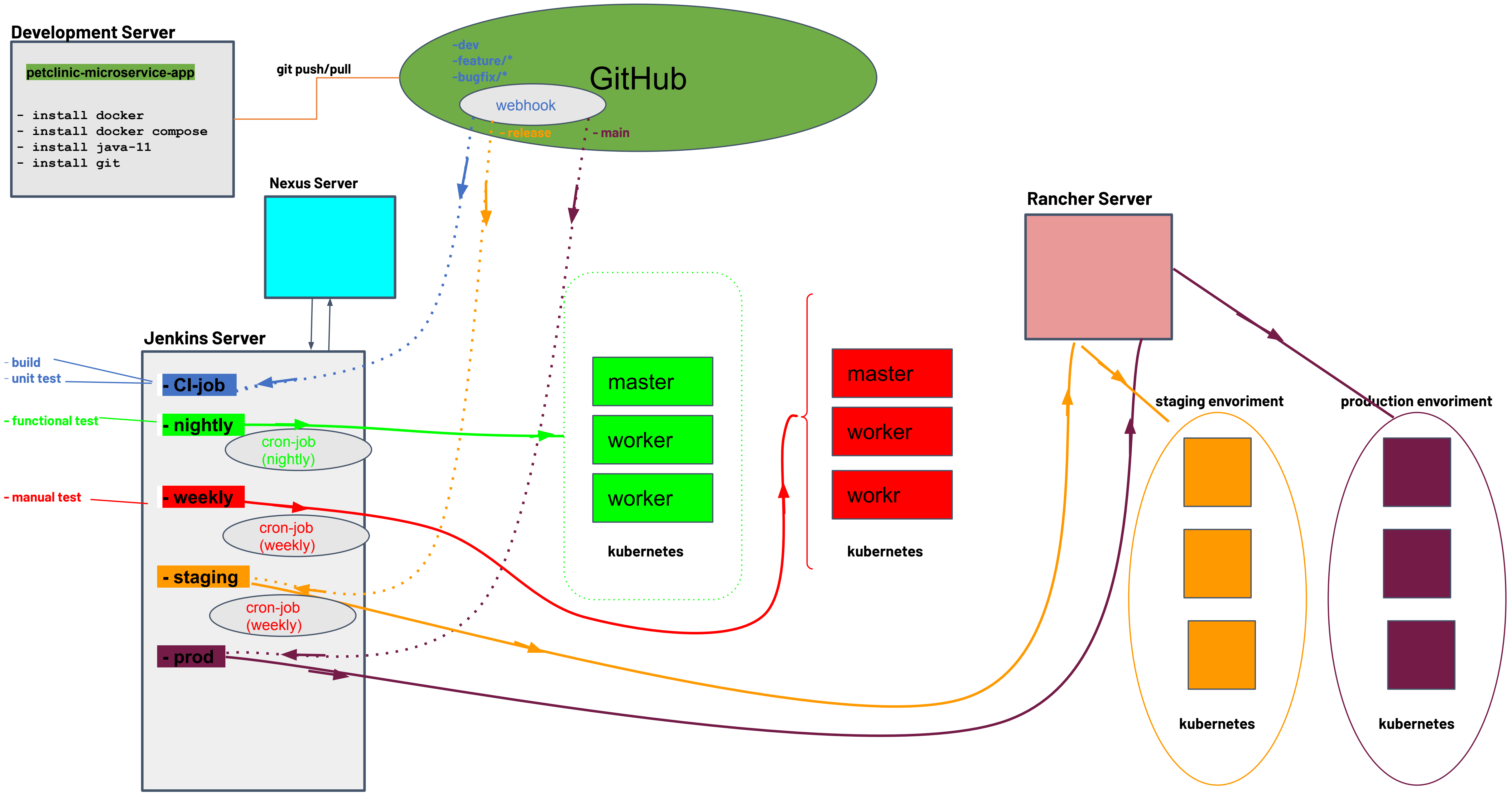

Create ECR for docker images with jenkins job
MSP-14
Launch instances with terraform
MSP-15
Setup Kubernetes cluster with ansible
MSP-16
create petclinic kubernetes yaml files and helm package
MSP-17
create images, ansible deploy playbook, test selenium software, selenium functional test ansible playbook and nightly functional test pipeline with jenkinsfile
MSP-18

PETCLINIC NIGHTLY PIPELINE

1. Create infrastructure
    a. Launch instances with terraform
    b. Setup Kubernetes cluster with ansible

2. Create application and deploy to kubernetes cluster
    a. Create ECR repo
    b. Prepare Docker Images
    c. Push Images to ECR Repo
    d. Create Kubernetes manifest files
    e. Create helm charts
    f. Deploy application on Kubernetes cluster with helm

3. Run Functional test with selenium


Development Server
petclinic-microservice-app
- install docker
- install docker compose
- install java-11
- install git
git push/pull
GitHub
-dev
-feature/*
-bugfix/*
webhook
- release
- main
Nexus Server
Jenkins Server
- build
- unit test
- functional test
- manual test
- CI-job
- nightly
cron-job (nightly)
- weekly
cron-job (weekly)
- staging
cron-job (weekly)
- prod
master
worker
worker
kubernetes
master
worker
workr
kubernetes
Rancher Server
staging enviroment
kubernetes
production enviroment
kubernetes

PETCLINIC NIGHTLY PIPELINE

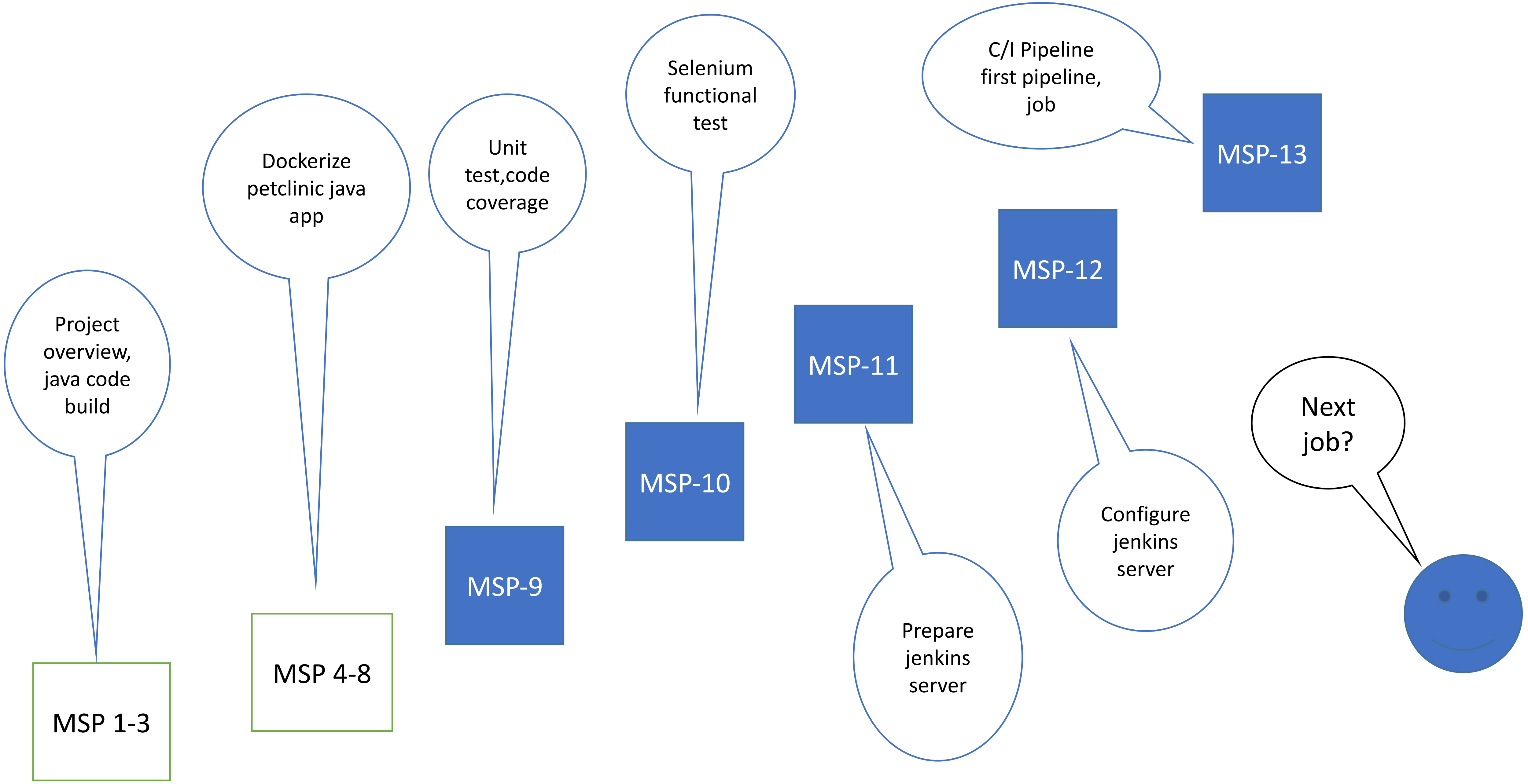
- Create ECR Repo
- Package Application
- Prepare Tags for Docker Images
- Build App Docker Images
- Push Images to ECR Repo
- Create Key Pair for Ansible
- Create QA Automation Infrastructure
- Create Kubernetes Cluster for QA Automation Build
- Deploy App on Kubernetes cluster
- Test the Application Deployment
- Run QA Automation Tests

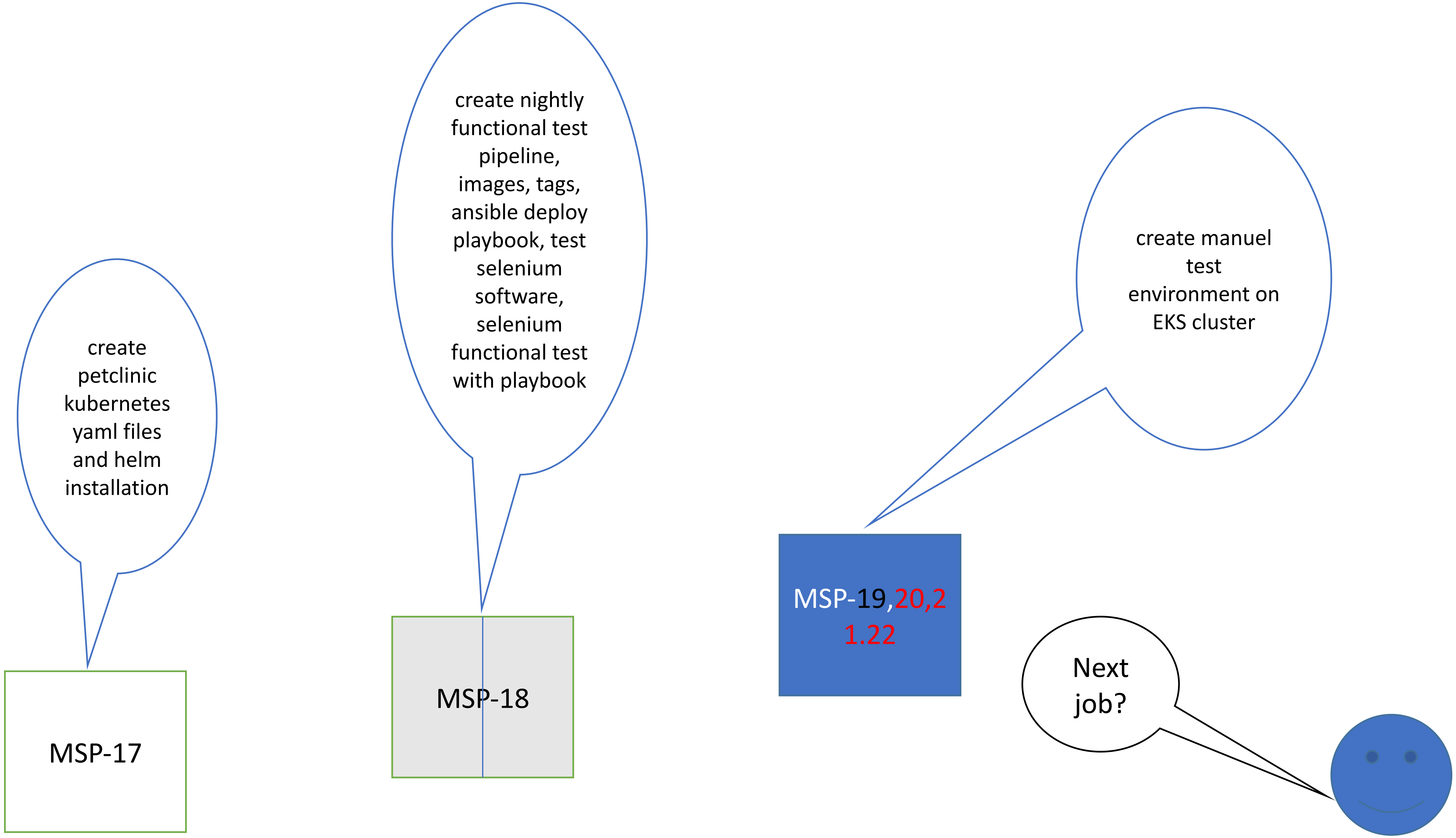
PETCLINIC NIGHTLY PIPELINE

- Create infrastructure with Terraform
- Launch Kubernetes Cluster with Ansible
- Create and push the helm charts to AWS S3
- Create images of services
- Deploy application on Kubernetes cluster with helm as helm release
- Run QA Automation Tests

PETCLINIC NIGHTLY PIPELINE

- Create infrastructure with terraform
  - Create Key Pair for Ansible
  - Create QA Automation Infrastructure
- Launch Kubernetes Cluster with ansible
  - Create Kubernetes Cluster for QA Automation Build
- Create image of services
  - Create ECR Repo
  - Package Application
  - Prepare Tags for Docker Images
  - Build App Docker Images
  - Push Images to ECR Repo
- Deploy App on Kubernetes cluster
  - Create and push the helm charts to AWS S3
  - Deploy application on kubernetes cluster with helm as helm release
- Run QA Automation Tests


Project overview, java code build
MSP 1-3
Dockerize petclinic java app
MSP 4-8
Unit test,code coverage
MSP-9
Selenium functional test
MSP-10
MSP-11
Prepare jenkins server
MSP-12
Configure jenkins server
C/I Pipeline first pipeline, job
MSP-13
Next job?


create petclinic kubernetes yaml files and helm installation
MSP-17
create nightly functional test pipeline, images, tags, ansible deploy playbook, test selenium software, selenium functional test with playbook
MSP-18
create manuel test environment on EKS cluster
MSP-19,20,21.22
Next job?

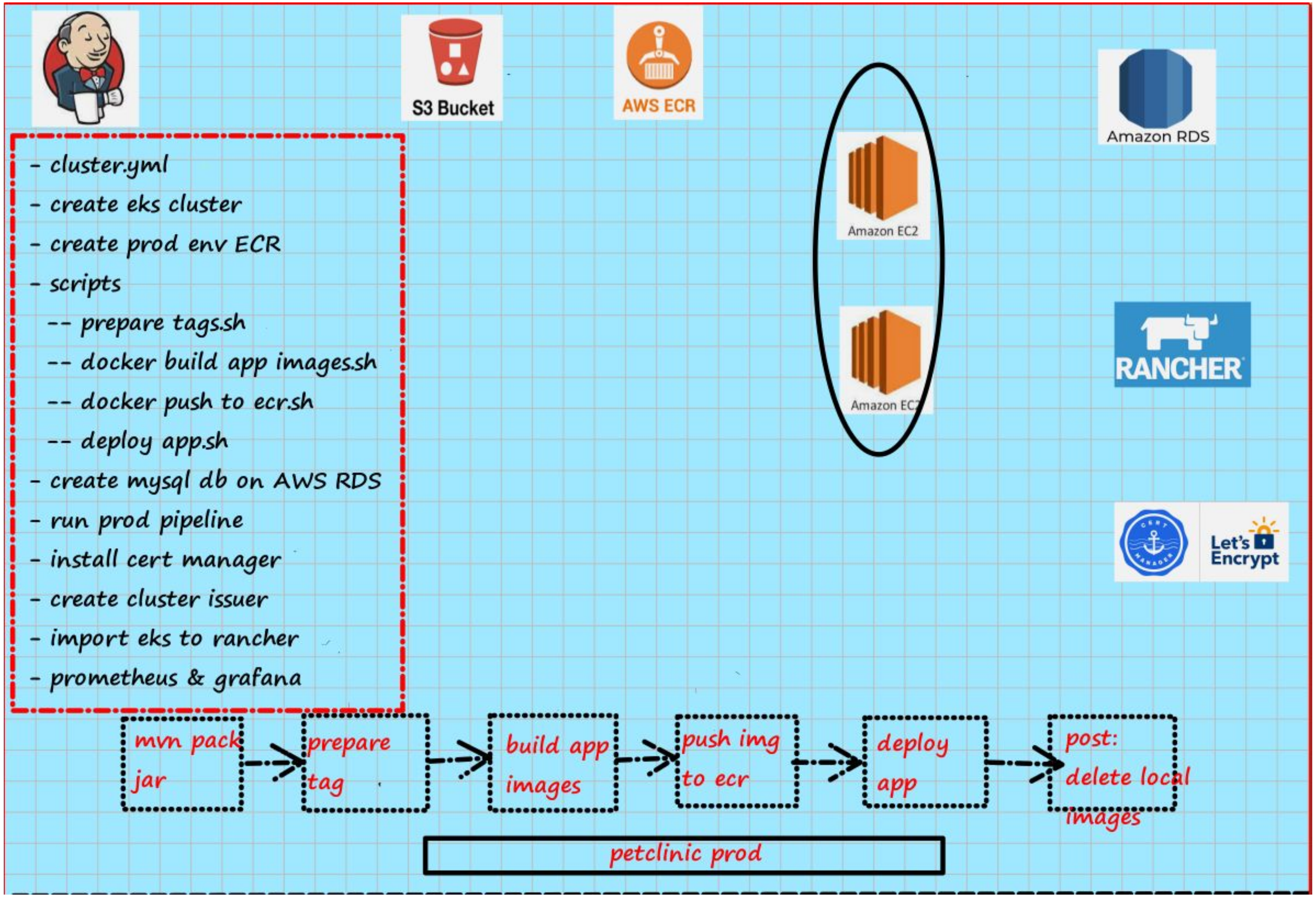
create nightly functional test pipeline, images, tags, ansible deploy playbook, test selenium software, selenium functional test with playbook

MSP-18

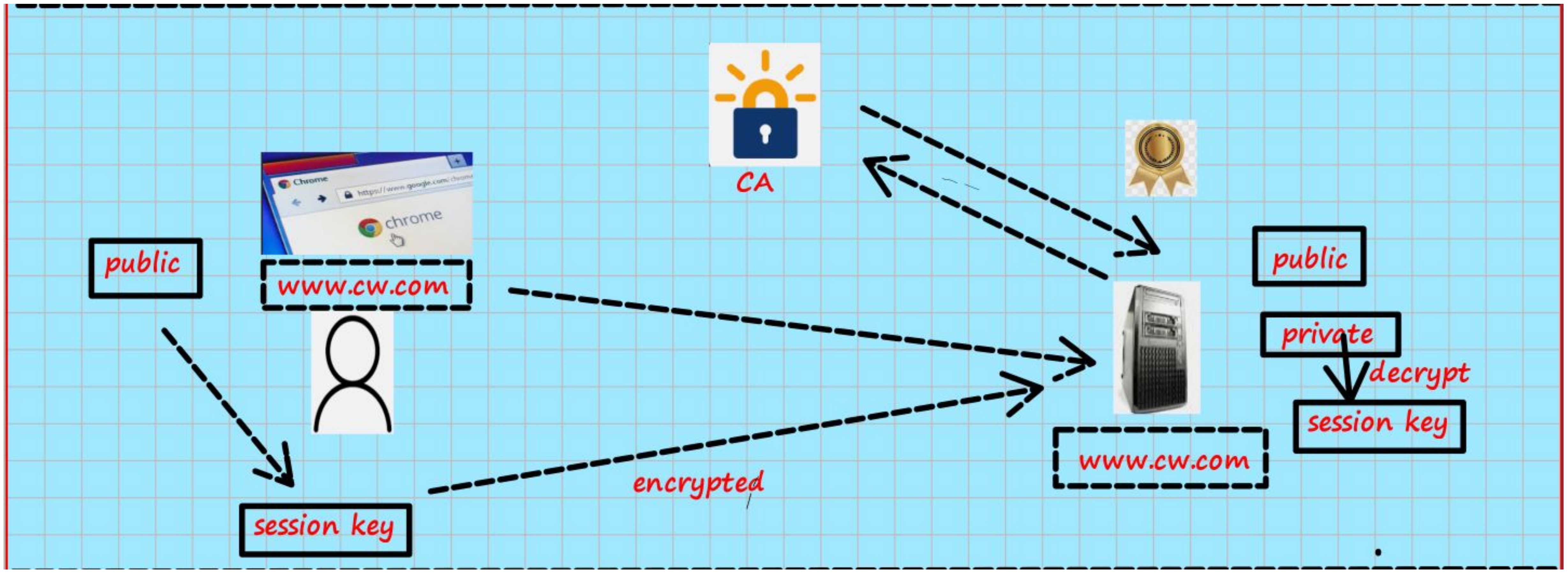
MSP-19,20,21.22

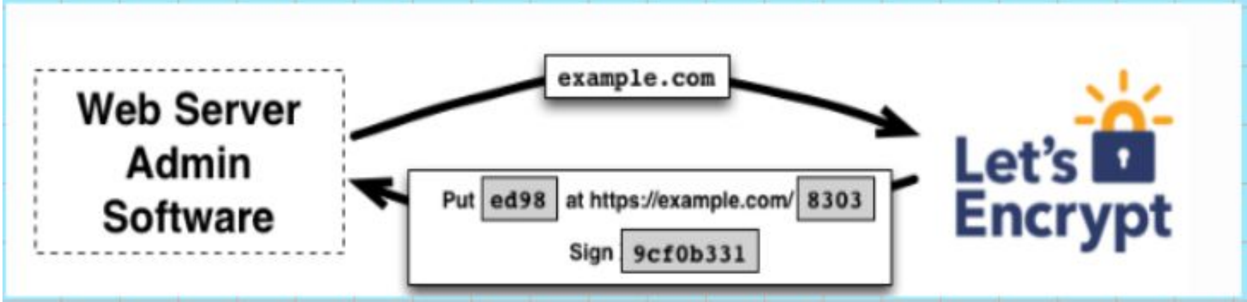
create manuel test environment on EKS cluster, prepare qa images and tags, deploy app manually to q/a inf., deploy with pipeline to q/a inf.

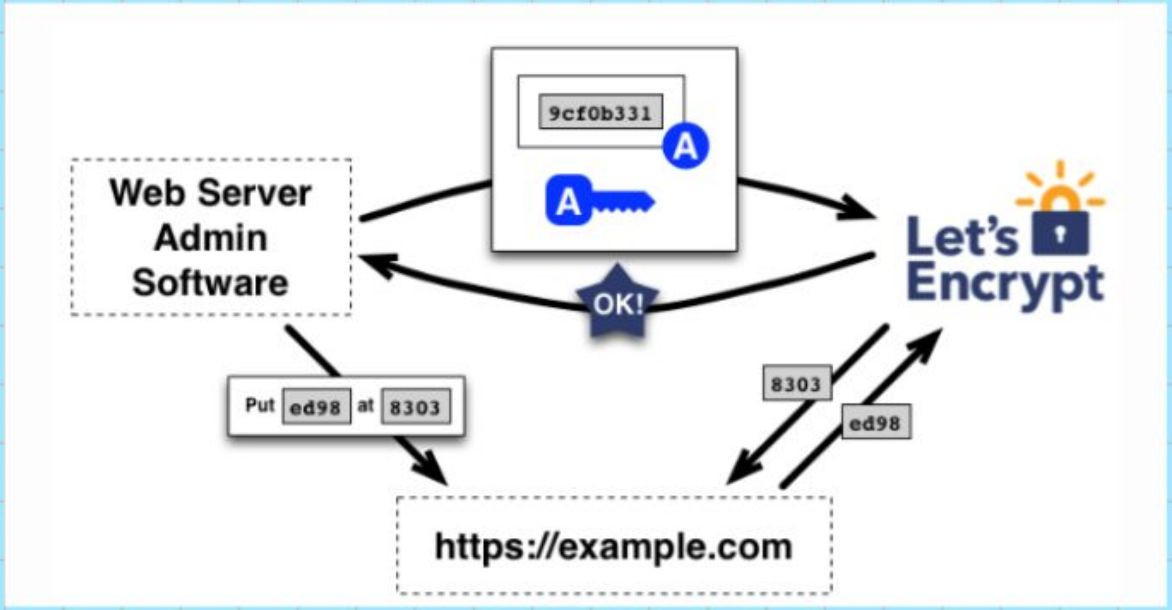
Next job?



- cluster.yml
- create eks cluster
- create prod env ECR
- scripts
  - -- prepare tags.sh
  - -- docker build app images.sh
  - -- docker push to ecr.sh
  - -- deploy app.sh
- create mysql db on AWS RDS
- run prod pipeline
- install cert manager
- create cluster issuer
- import eks to rancher
- prometheus & grafana


CA
public
www.cw.com
session key
encrypted
public
private
decrypt
session key
www.cw.com


Web Server
Admin
Software
example.com
Put ed98 at https://example.com/ 8303
Sign 9cf0b331
Let's Encrypt


9cf0b331
A
A
Web Server
Admin
Software
OK!
Let's Encrypt
Put ed98 at 8303
8303
ed98
https://example.com