Final Project

Building a CI/CD for Spring Framework PetClinic

- Graduated from Kharkiv National University of Radio Electronics
- I started my career as a engineer with Spezvuzavtomatika
- Now I have been working as a network engineer for over 20 years
- I have worked with different equipment of different brands

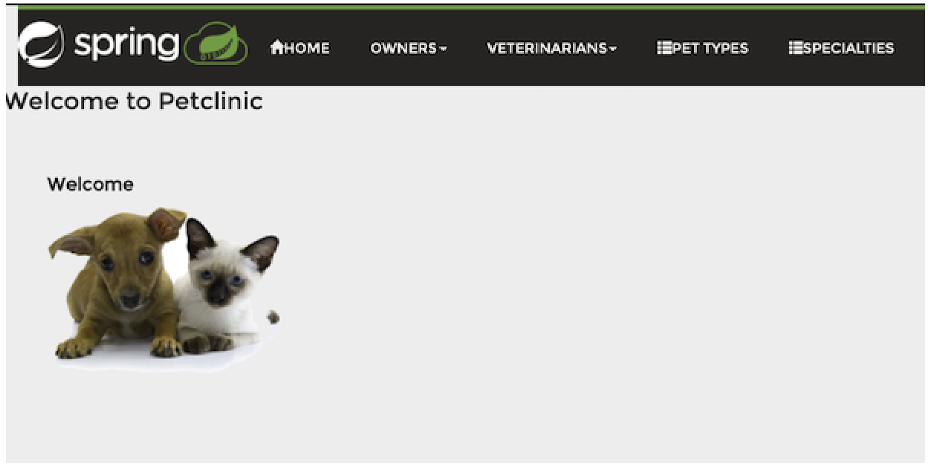
Vyacheslav Chudnov

Spring Framework PetClinic

Spring Framework is a Java platform that provides comprehensive infrastructure support for developing Java applications. Spring handles the infrastructure so you can focus on your application.

The Spring PetClinic is a sample application designed to show how the Spring stack can be used to build simple, but powerful database-oriented applications.

Maven is a popular open-source build tool developed by the Apache Group to build, publish, and deploy several projects at once for better project management. The tool provides allows developers to build and document the lifecycle framework.



Welcome to Petclinic

Welcome

Official version of PetClinic: https://github.com/spring-projects/spring-petclinic

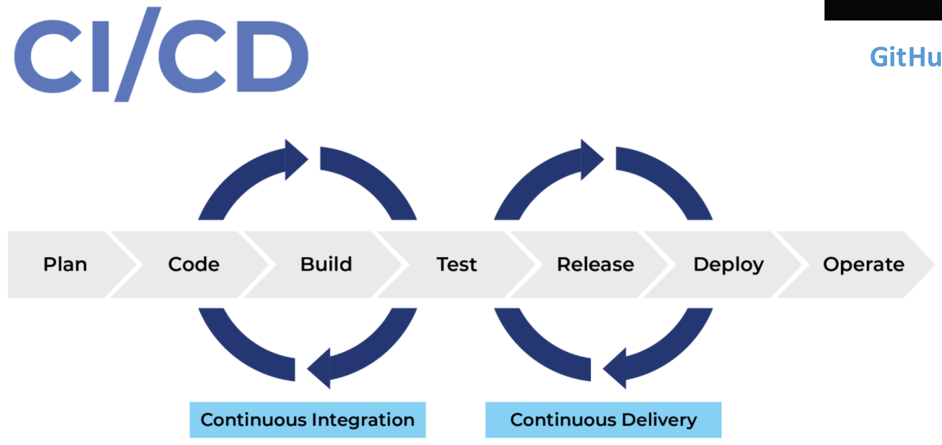
List of used DevOps tools:

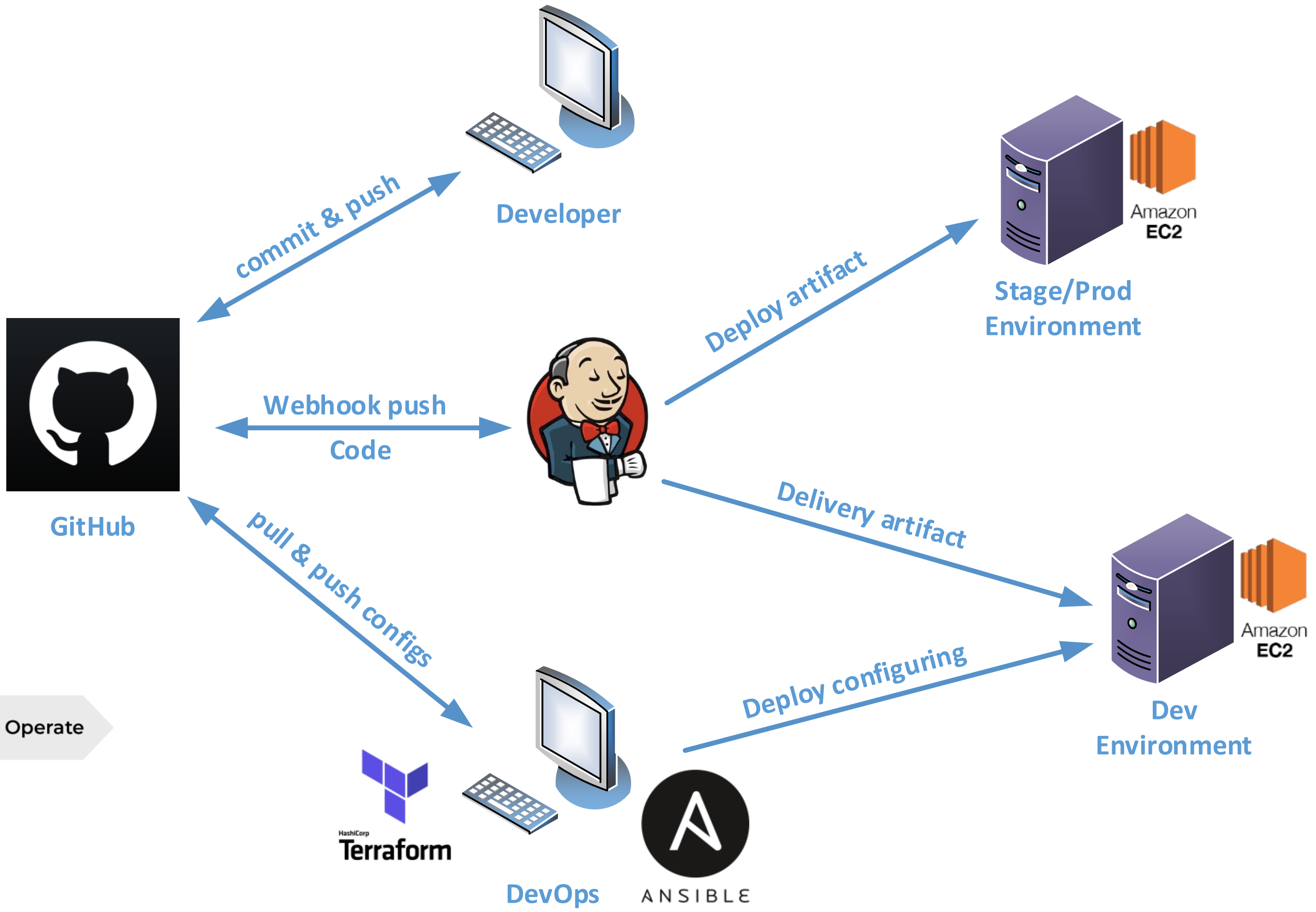
- VirtualBox
- GitHub
- AWS
- Terraform
- Ansible
- Docker
- Jenkins
- Linux
- Windows
- Visual Studio Code
- Atom

CI/CD process

Main goals:

1. Understand and run the whole process because I am new to DevOps
2. Achieve maximum automation
3. Reduce the time between commit and getting the result





AWS & Terraform

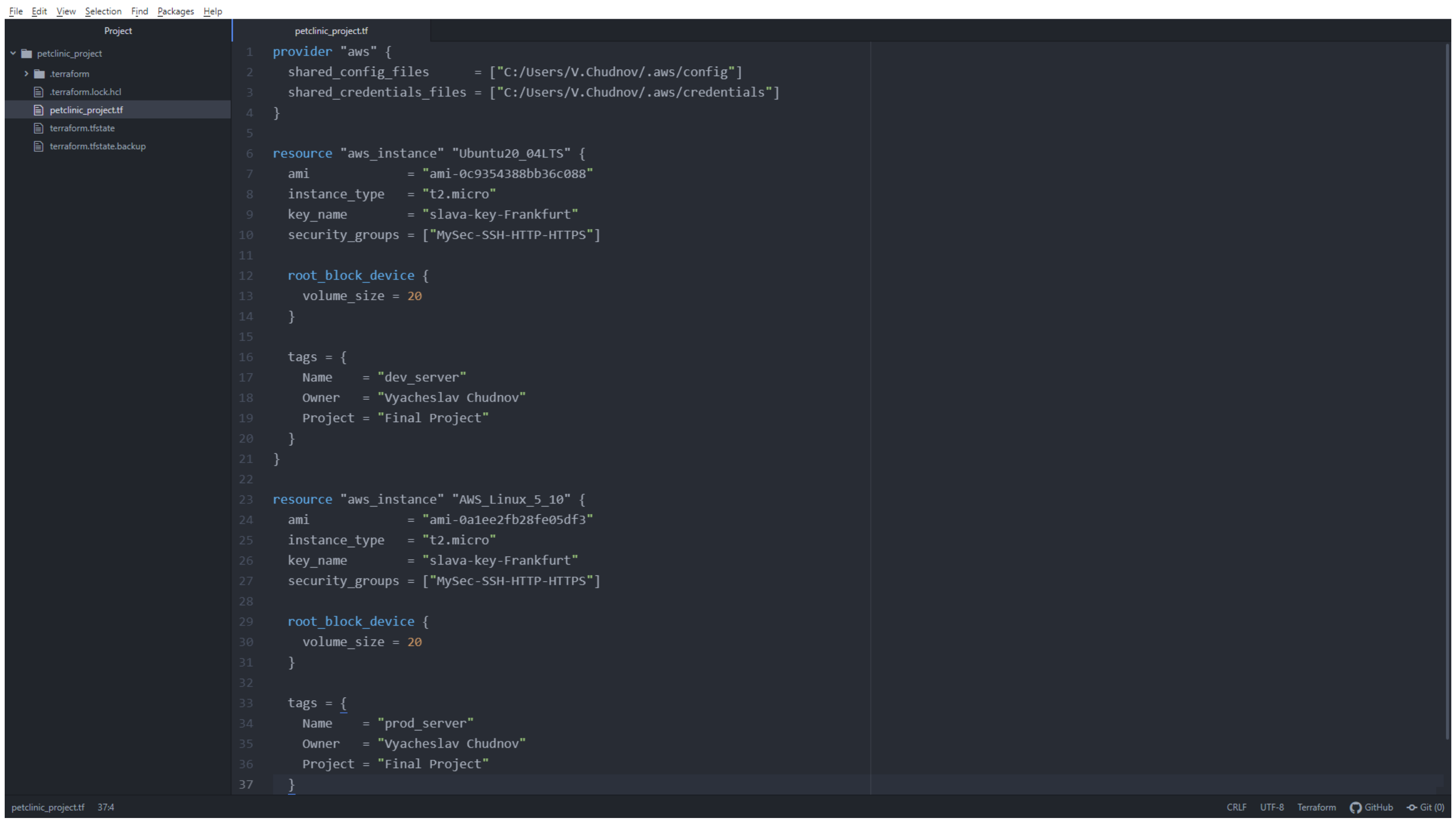
What should we think about for our project?

- OS (Linux, Windows, version)
- Number of servers
- Best Region
- Configure IAM, VPC
- Security Groups
- Key Pairs
- Size of Volumes

This is not all but enough for our project

AWS & Terraform

Based on the collected data we must create a terraform file



```
provider "aws" {
  shared_config_files      = ["C:/Users/V.Chudnov/.aws/config"]
  shared_credentials_files = ["C:/Users/V.Chudnov/.aws/credentials"]
}

resource "aws_instance" "Ubuntu20_04LTS" {
  ami             = "ami-0c9354388bb36c088"
  instance_type   = "t2.micro"
  key_name        = "slava-key-Frankfurt"
  security_groups = ["MySec-SSH-HTTP-HTTPS"]

  root_block_device {
    volume_size = 20
  }

  tags = {
    Name    = "dev_server"
    Owner   = "Vyacheslav Chudnov"
    Project = "Final Project"
  }
}

resource "aws_instance" "AWS_Linux_5_10" {
  ami             = "ami-0a1ee2fb28fe05df3"
  instance_type   = "t2.micro"
  key_name        = "slava-key-Frankfurt"
  security_groups = ["MySec-SSH-HTTP-HTTPS"]

  root_block_device {
    volume_size = 20
  }

  tags = {
    Name    = "prod_server"
    Owner   = "Vyacheslav Chudnov"
    Project = "Final Project"
  }
```

HashiCorp Terraform

Specify

- AWS AMI
- Instance Type
- Region
- Security Groups
- Key Pairs
- Size of Volumes
- Tags

No credentials should be in plain text!

ANSIBLE

Using Ansible we install the necessary packages on the servers

Dev and Prod servers:

- OpenJDK
- Unit file pet.service

pet.service is used to run the PetClinic in the background so that the Jenkins job completes without error (UNSTABLE) and also to restart the service each time Jenkins pushes a new version of the project.

Jenkins servers (master and agent):

- Install required system packages
- Install Docker
- Run docker container Jenkins latest LTS

All credentials (keys, passwords, users) are stored in separate files for security

Jenkins

Jenkins start:

- Install suggested plugins
- SSH Agent plugin
- Copy Artifact plugin
- Add GitHub key
- Add keys for each servers

Jenkins pipeline:

- Create pipeline
- Create Stage ‘Pull’
- Create Stage ‘Test’
- Create Stage ‘Build’
- Create Stage ‘Delivery’

Jenkins job:

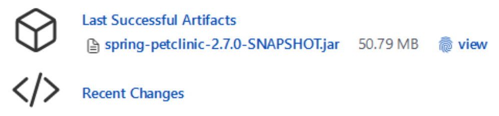
- Create a PetClinic-Project pipeline and add it to the Jenkins GitHub repository
- Create a commit tracking Job
- Create freestyle Job for Deploy to Production

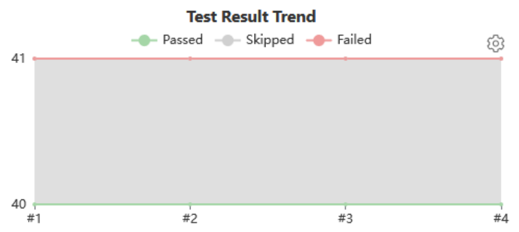
Summary

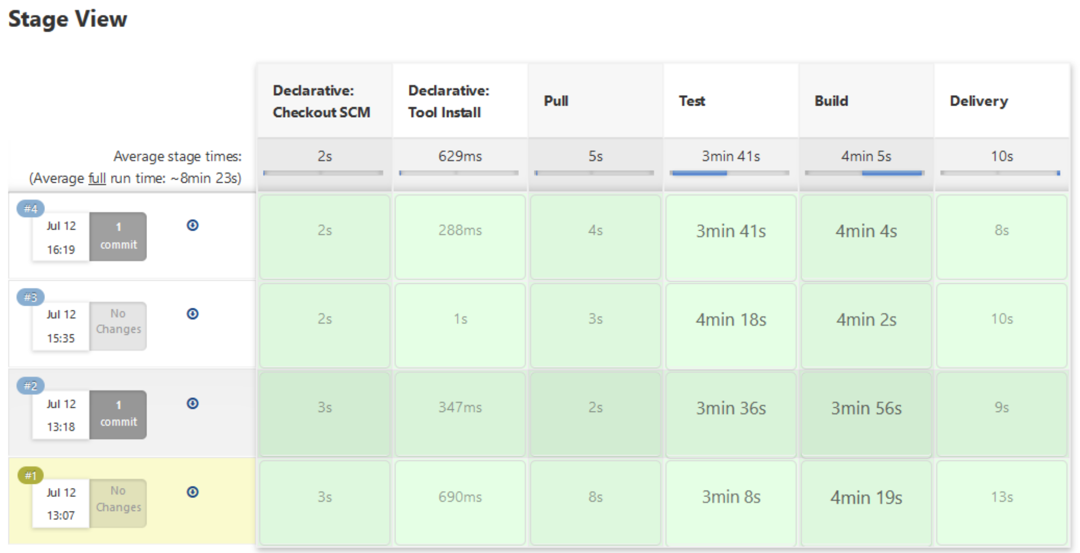
Average stage times:

- Deploying Servers in the AWS Cloud: 33 sec
- Configuring Servers with Ansible: 1 min 48 sec
- Launch pipeline on Jenkins server and delivery to Dev server: ~7min 55s
- Deployment to Prod server: 10 sec

Total times: 10 min 30 sec


Last Successful Artifacts
spring-petclinic-2.7.0-SNAPSHOT.jar 50.79 MB view
Recent Changes




Stage View

	Declarative: Checkout SCM	Declarative: Tool Install	Pull	Test	Build	Delivery
Average stage times: (Average full run time: ~8min 23s)	2s	629ms	5s	3min 41s	4min 5s	10s
#4 Jul 12 16:19 1 commit	2s	288ms	4s	3min 41s	4min 4s	8s
#3 Jul 12 15:35 No Changes	2s	1s	3s	4min 18s	4min 2s	10s
#2 Jul 12 13:18 1 commit	3s	347ms	2s	3min 36s	3min 56s	9s
#1 Jul 12 13:07 No Changes	3s	690ms	8s	3min 8s	4min 19s	13s