

Тестирование программного обеспечения

Лабораторная работа 7

CI/CD

Дорогин Никита ИП-312

Цель работы

Ознакомиться с принципами CI/CD (Continuous Integration / Continuous Delivery) в контексте разработки и тестирования OpenVMC.

Настройка CI/CD с использованием GitHub и Jenkins в Docker

Развертывание Jenkins в Docker

Уже установив docker для нужд курса Архитектура распределённых приложений используем его команды.

Делаем pull официального образа Jenkins.

```
gastello123@gastello123:~/Desktop/ТестированиеПо/testP0/Lab6$ cd ../Lab7
gastello123@gastello123:~/Desktop/ТестированиеПо/testP0/Lab7$ sudo docker pull jenkins/jenkins:lts
[sudo] пароль для gastello123:
lts: Pulling from jenkins/jenkins
cae3b572364a: Pull complete
11c82e82e8c5: Pull complete
6d8ebc8a18e6: Pull complete
e29665228ac2: Pull complete
cc05fa07d253: Pull complete
7c2b9fc47dae: Pull complete
9e58f885f660: Pull complete
51148860bddf: Pull complete
eba243d676e4: Pull complete
04e220b291b8: Pull complete
b9bcce170b58: Pull complete
606dcc9d6add: Pull complete
Digest: sha256:f2519b99350faeaaef30e3b8695cd6261a5d571c859ec37c7ce47e5a241458d
Status: Downloaded newer image for jenkins/jenkins:lts
docker.io/jenkins/jenkins:lts
gastello123@gastello123:~/Desktop/ТестированиеПо/testP0/Lab7$
```

Небольшое отступление:

В процессе написания файла с инструкциями для CI/CD (Jenkinsfile), я пользовался теми же инструкциями загрузки пакетов, какими пользуюсь в bash-терминале, то есть пакетным менеджером apt. Проблема в том, что Jenkins не знает что такое sudo, а без особых прав не позволяет пользоваться этим менеджером.

Для решения этой проблемы мне пришлось собрать свой образ Jenkins, на основе jenkins:lts, но с указанием root в качестве пользователя.

Для этого пишем Dockerfile следующего содержания:

```
FROM jenkins/jenkins:lts

USER root

RUN jenkins-plugin-cli --plugins workflow-aggregator git docker-workflow

EXPOSE 8080
EXPOSE 50000
```

(строка с RUN, наверное, не обязательна, так как в Web UI нам всё равно при регистрации придётся выбирать установку плагинов)

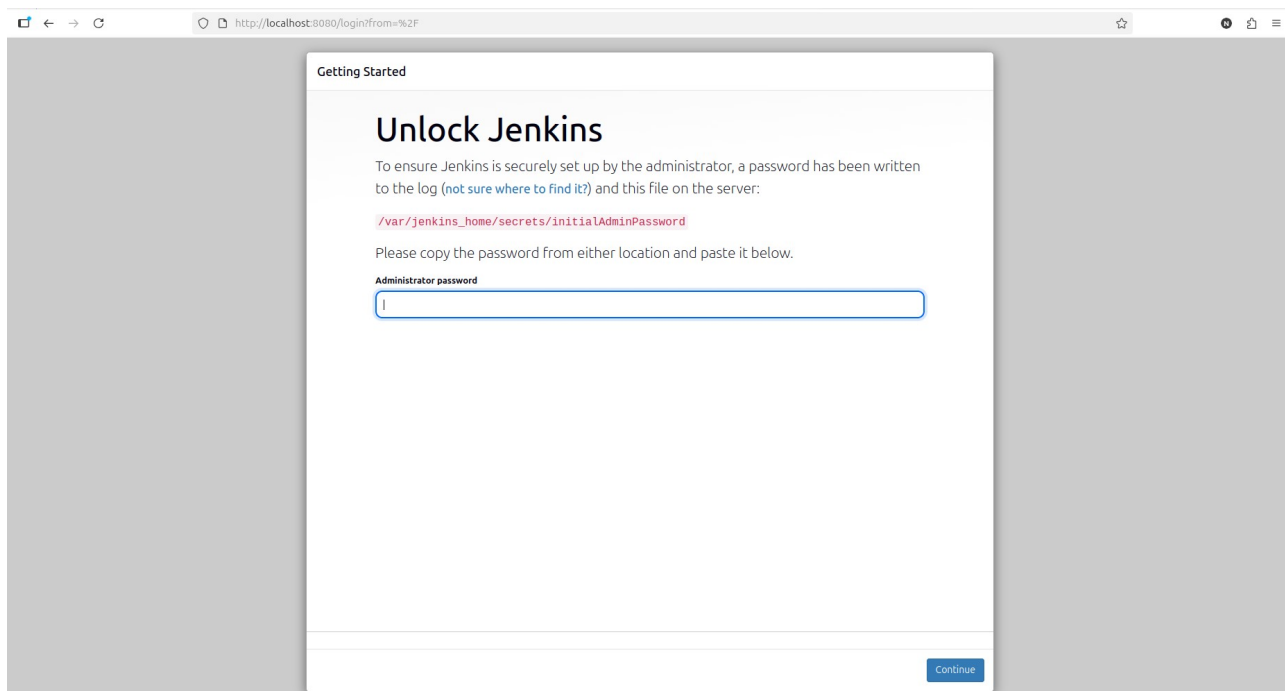
Далее собираем образ командой
`sudo docker build -t <ваше_имя_образа> .`

Запускаем образ (в фоновом режиме -d, с указанием портов -p)

Здесь указан запуск того образа, что мы загружали изначально, поменял его на свой я уже позднее, всё что идёт после указания портов заменяем просто на имя своего образа.

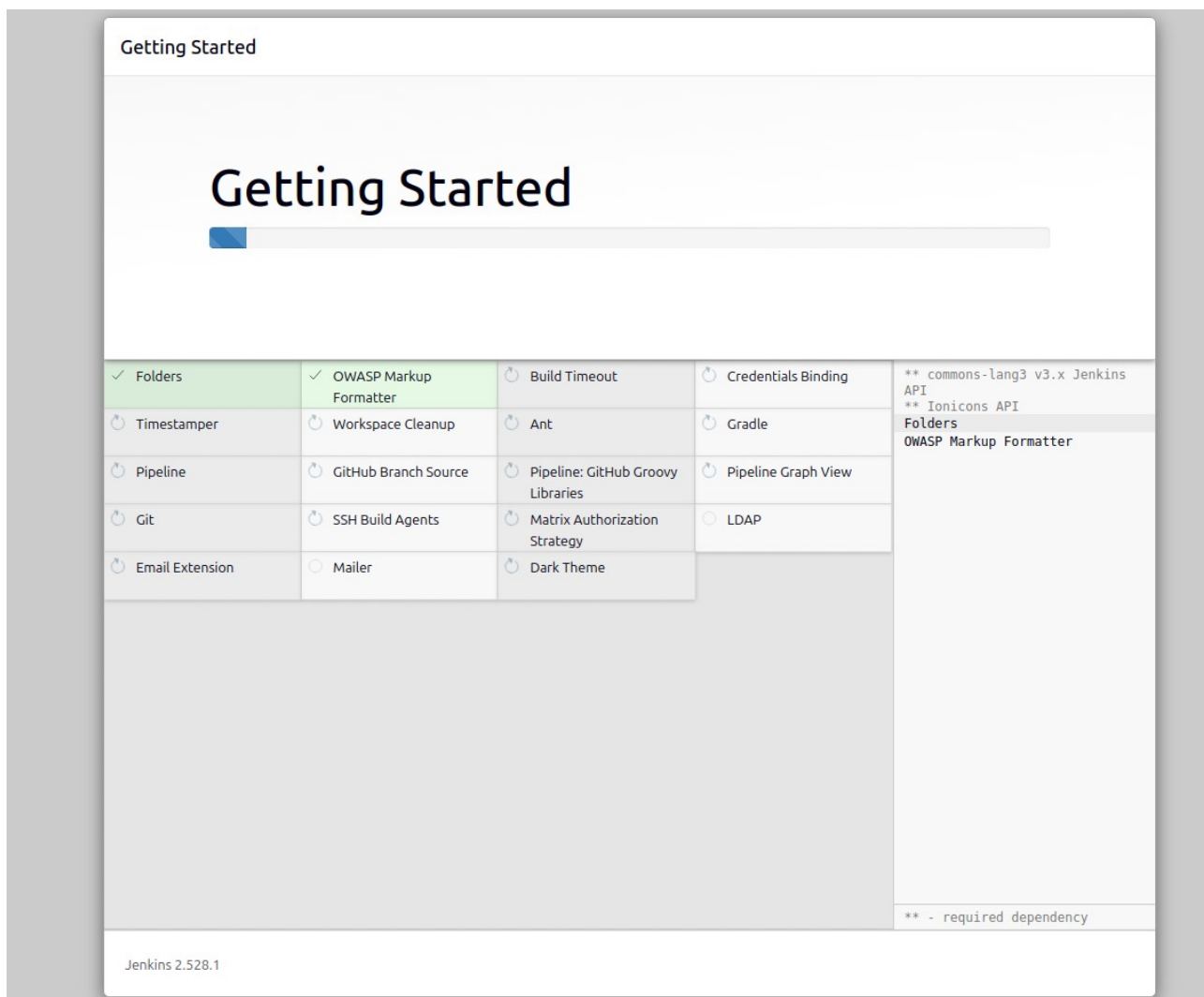
```
gastello123@gastello123:~/Desktop/ТестированиеПо/testP0/Lab7$ sudo docker run -d -p 8080:8080 -p 50000:50000 -v jenkins_home:/var/jenkins_home jenkins/jenkins:lts
156093d1b815a4d35f4f60ee63ec6996bb65331ca8195d422cb1c1bfb6225033
gastello123@gastello123:~/Desktop/ТестированиеПо/testP0/Lab7$ sudo docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS
156093d1b815   jenkins/jenkins:lts   "/usr/bin/tini -- /u..."   13 seconds ago   Up 12 seconds   0.0.0.0:8080->8080/tcp, [::]:8080->8080/tcp, 0.0.0.0:50000->50000/tcp, [::]:50000->50000/tcp
p_hungry_jemison
```

В адресной строке вводим localhost:8080 и оказываемся на следующей странице:



Пароль узнаём из логов docker-контейнера

```
gastello123@gastello123:~/Desktop/ТестированиеПо/testP0/Lab7$ sudo docker logs hungry jemison
2025-10-24 07:48:37.992+0000 [id=90] INFO jenkins.InitReactorRunner$1#onAttained: Configuration for all jobs updated
2025-10-24 07:48:38.004+0000 [id=106] INFO hudson.util.Retrier#start: Attempt #1 to do the action check updates server
2025-10-24 07:48:38.228+0000 [id=80] INFO jenkins.install.SetupWizard#init:
[LF]>
[LF]> *****
[LF]> *****
[LF]> *****
[LF]> *****
[LF]> Jenkins initial setup is required. An admin user has been created and a password generated.
[LF]> Please use the following password to proceed to installation:
[LF]> 8e23b3ccf9ef4c6f80a9438ddb236602
[LF]>
[LF]> This may also be found at: /var/jenkins_home/secrets/initialAdminPassword
[LF]>
[LF]> *****
[LF]> *****
[LF]> *****
2025-10-24 07:48:43.181+0000 [id=80] INFO jenkins.InitReactorRunner$1#onAttained: Completed initialization
2025-10-24 07:48:43.196+0000 [id=53] INFO hudson.lifecycle.Lifecycle#onReady: Jenkins is fully up and running
```



В первый раз загрузка подвисла в самом конце, пришлось перезапустить образ (sudo docker stop <название_контейнера> и sudo docker start <название_контейнера>

Create First Admin User

Имя пользователя

NekitD

Пароль

Повторите пароль

Ф.И.О.

Дорогин Никита Сергеевич

Адрес электронной почты

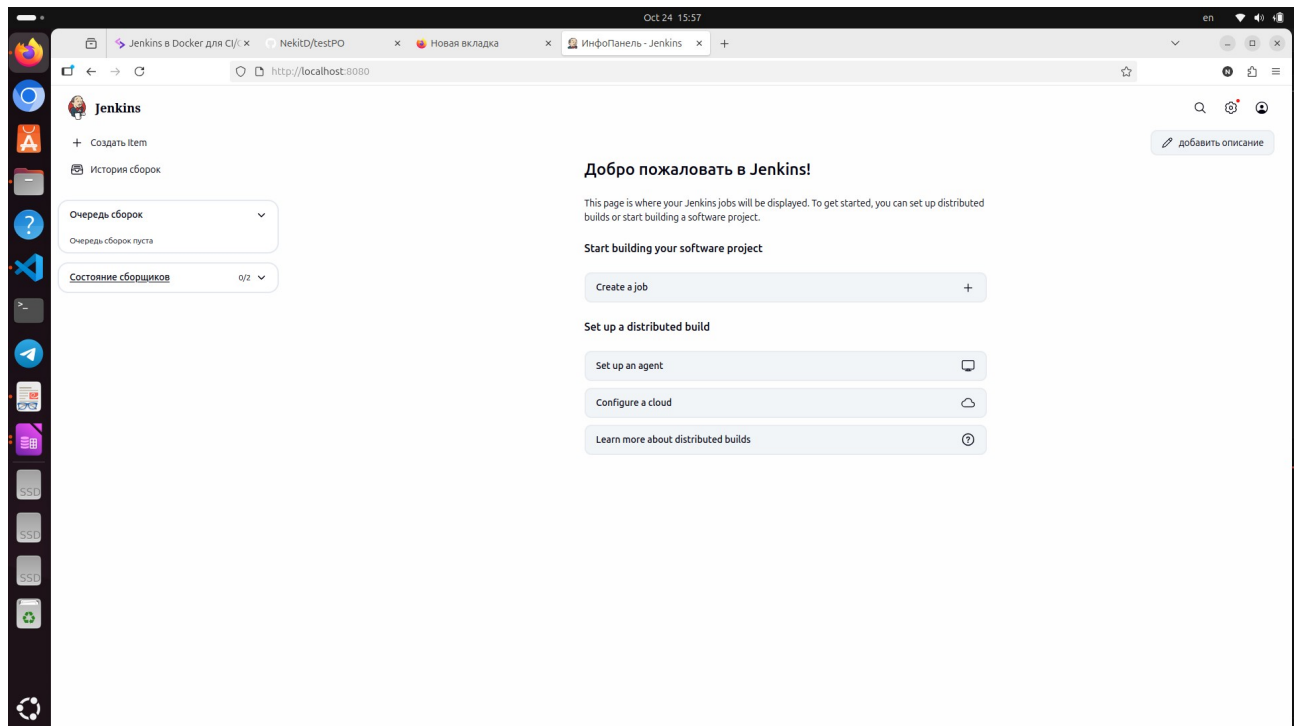
nikitadorogin@yandex.ru

Instance Configuration

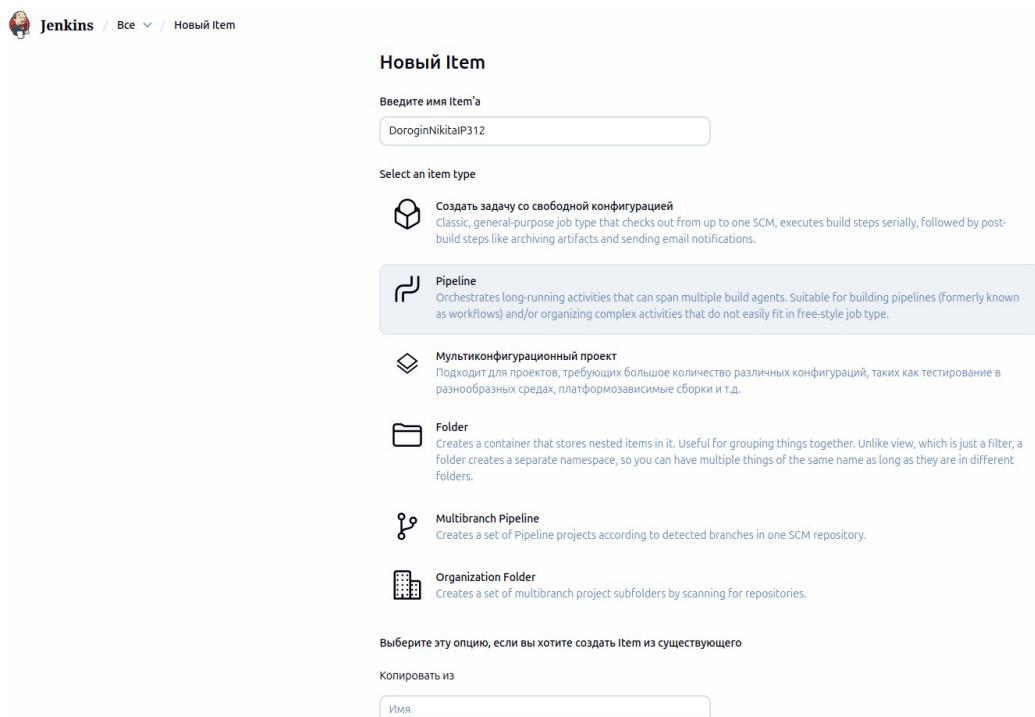
Jenkins URL:

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the `BUILD_URL` environment variable provided to build steps.

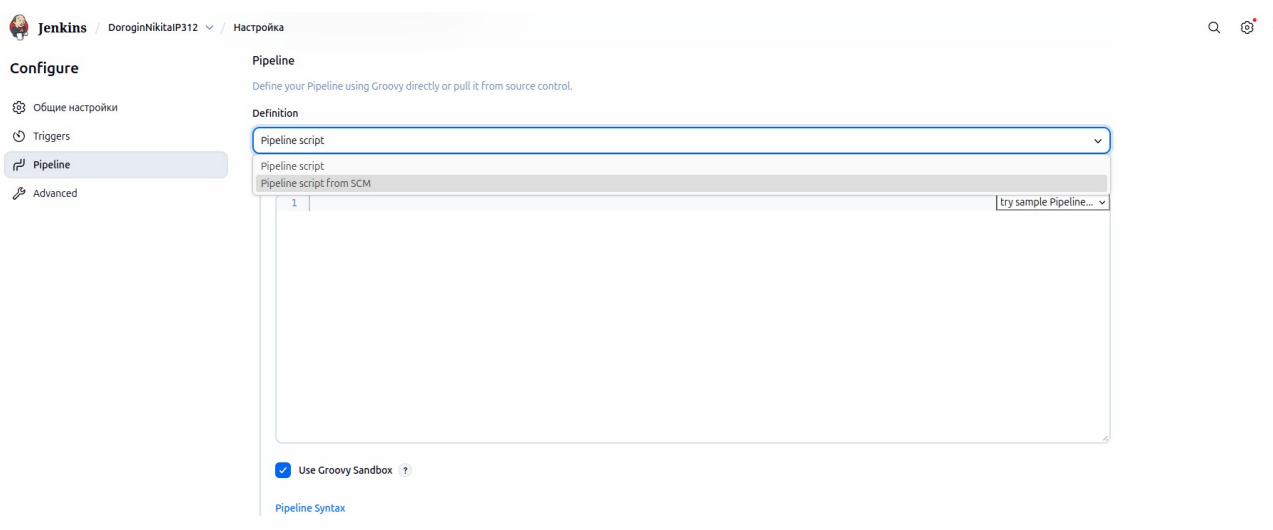
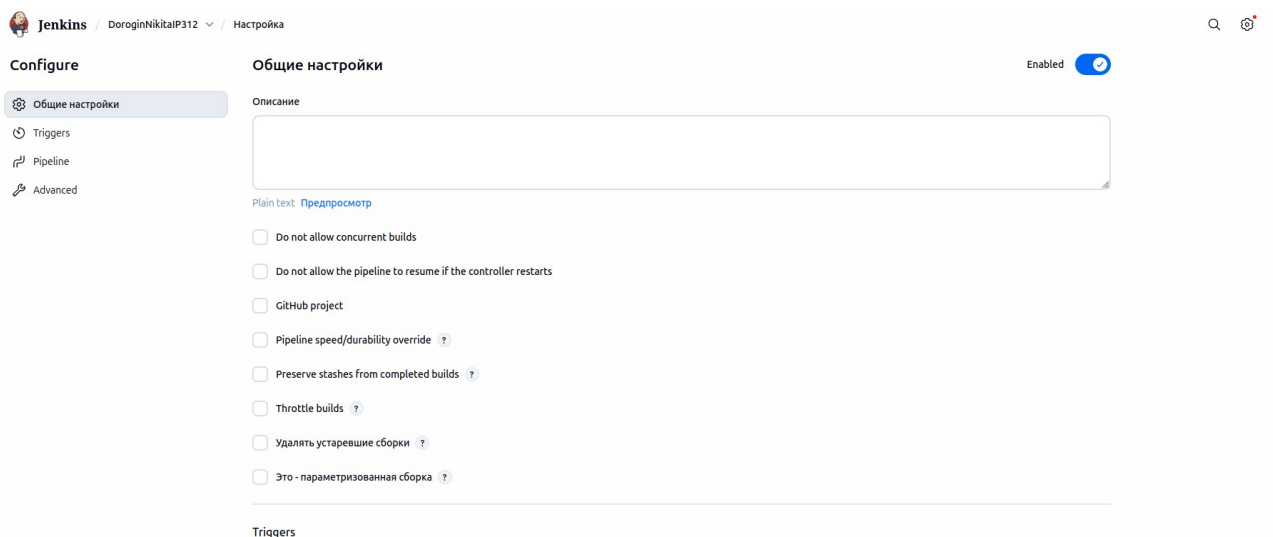
The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.



Создаем и настраиваем нам пайплайн:



Общие настройки, триггеры и продвинутые не трогаем. Нам нужен только Pipeline.



Указываем внешний источник Git, ссылку на наш репозиторий, с которого будет браться исходный код. В Credentials я вписал поля имя пользователя NekitD и ssh-ключ от репозитория, это вполне работает, больше ничего не нужно.

Configure

Общие настройки

Triggers

Pipeline

Advanced

Pipeline

Define your Pipeline using Groovy directly or pull it from source control.

Definition

Pipeline script from SCM

SCM

Git

Repositories

Repository URL

https://github.com/NekitD/testPO

Credentials

NekitD

+ Add

Расширенные

+ Add Repository

Branches to build

Branch Specifier (blank for 'any')

*/main

Указываем путь к Jenkinsfile в нашем репозитории.

Script Path

Lab7/Jenkinsfile

☒ Lightweight checkout

[Pipeline Syntax](#)

Advanced

Расширенные

Save

Применить

Jenkinsfile:

Запуск Qemu:

```
pipeline {
  agent any
  stages {
    stage('Qemu') {
      steps {
        echo 'Start qemu'
        sh 'rm -rf romulus romulus.zip'
        sh 'apt update'
        sh 'apt install -y qemu-system'
        sh 'apt install -y wget'
        sh 'apt install -y unzip'
        sh 'apt install -y expect'
        sh 'wget https://jenkins.openbmc.org/job/ci-openbmc/lastSuccessfulBuild/distro=ubuntu,label=docker-build'
        sh 'unzip romulus.zip'
        sh '''
        LATEST_MTD=$(ls romulus/obmc-phosphor-image-romulus-*.static.mtd | sort -V | tail -1)
        expect -c "
        log_file -a qemu_result.txt
        spawn qemu-system-arm -m 256 -M romulus-bmc -nographic -drive file=$LATEST_MTD,format=raw,if=mtd -net nic
        sleep 120
        send "root\r"
        sleep 4
        send "OpenBmc\r"
        expect \"root@romulus:\"
        interact" &
        QEMU_PID=$!
        echo "QEMU запущен с PID: $QEMU_PID"
        echo $QEMU_PID > qemu.pid
        '''
      }
    }
  }
}
```

Запуск автотестов и тестов для WebUI:

```
stage('Auto') {
  steps {
    echo 'Start autotests and WebUI tests'
    sh 'apt install -y python3'
    sh 'apt install -y python3-pytest'
    sh 'apt install -y python3-selenium'
    catchError(buildResult: 'SUCCESS', stageResult: 'UNSTABLE'){
      sh '''cd Lab4
      ls
      python3 -m pytest -v ob-autotests.py > ob-autotest_results.txt 2>&1'''
    }
    sh 'cd ..'
  }
}
```

(catchError нужны для того чтобы пайплайн не лёг от первого же неудачного теста и дошёл до конца, это важно, учитывая что во всех лабораторных у нас были тесты, которые неизбежно заканчивались неудачно)

Запуск нагрузочного тестирования:

```
stage('Locust') {
    steps {
        echo 'Start Locust tests'
        sh 'apt install -y python3 locust'
        catchError(buildResult: 'SUCCESS', stageResult: 'UNSTABLE'){
            sh '''cd Lab6
                ls
                timeout 180 locust --headless --users 50 --spawn-rate 5 --run-time 3m --host=https://127.0.0.1:2443 >
            }
            sh 'cd ..'
        }
    }
}
```

```
ult: 'UNSTABLE'){
```


```
0 --spawn-rate 5 --run-time 3m --host=https://127.0.0.1:2443 > locust_results.txt 2>&1''
```

Завершение работы, очистка рабочего пространства и сохранения артефактов:

```
stage('Stop Qemu') {
    steps {
        sh '''
            echo "Stop Qemu..."
            if [ -f "qemu.pid" ]; then
                QEMU_PID=$(cat qemu.pid)
                kill $QEMU_PID 2>/dev/null || true
                rm -f qemu.pid
                echo "Qemu shutted down"
            fi
            rm -rf romulus/ romulus.zip || true
            rm -f *.log *.pid *.json *.csv || true
            rm -f qemu.pid image_path.txt config.env || true
            du -sh . | awk '{print $1}'
        '''
    }
}

post {
    always {
        archiveArtifacts artifacts: '**/*.txt', fingerprint: true
    }
}
```

Результаты:

 **Jenkins** / DoroginNikitaIP312

Status

</> Changes

▶ Собрать сейчас

⚙ Настройки

🗑 Удалить Pipeline

📂 Stages

✏ Переименовать

❓ Pipeline Syntax

🔑 Credentials

✓ DoroginNikitaIP312

📁

Last Successful Artifacts

📄 ob-autotest_results.txt

4.55 KiB

🔗 view

📄 locust_results.txt

10.77 MiB

🔗 view

📄 qemu_result.txt

8.67 KiB

🔗 view

Постоянные ссылки

- [Последняя сборка \(#10\)](#), 6 минут 52 секунд назад
- [Последняя стабильная сборка \(#10\)](#), 6 минут 52 секунд назад
- [Последняя успешная сборка \(#10\)](#), 6 минут 52 секунд назад
- [Последняя провалившаяся сборка \(#7\)](#), 41 минут назад
- [Last unsuccessful build \(#7\)](#), 41 минут назад
- [Last completed build \(#10\)](#), 6 минут 52 секунд назад

Builds

⋮ ↗

🔍 Фильтровать сборки...

/

Today

✓ #10 19:24

✓ #9 19:13

✓ #8 19:07

✗ #7 18:49

Ссылка на GitHub:

<https://github.com/NekitD/testPO/tree/main/Lab7>