Звіт
Лабораторна робота 5
ТЕХНОЛОГІЇ ПРОГРАМУВАННЯ
Большаков Андрій МІТ-31

Тема: The Application Delivery pipeline
Мета: навчитися створювати елементарний ковеєр автоматизованого випуску програмного забезпечення (application delivery pipeline).

https://github.com/Utilka/univ_programming_technologies_proj

Установили плагины
Вставили дженкинсфайл
Вставили генератор xml репортов

В дженкинсе создали новый проект типа пайплайн
вставили туда дженкинсфайл

```
13:04:18  Running tests...
13:04:18  ----------------------------------------------------------------------
13:04:18  ......
13:04:18  ----------------------------------------------------------------------
13:04:18  Ran 6 tests in 0.004s
13:04:18
13:04:18  OK
13:04:18
13:04:18  Generating XML reports...
Post stage
[Pipeline] junit
13:04:18  Recording test results
13:04:18  [Checks API] No suitable checks publisher found.
[Pipeline] echo
13:04:18  Application testing successfully completed
[Pipeline] }
13:04:18  $ docker stop --time=1 3529869d10f418b5afc82295bc5e62ebde6cba9112c1cfd673fd9467862f17e6
13:04:19  $ docker rm -f 3529869d10f418b5afc82295bc5e62ebde6cba9112c1cfd673fd9467862f17e6
[Pipeline] // withDockerContainer
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Docker Publish)
[Pipeline] echo
13:04:20  Application Publishing
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // timestamps
[Pipeline] End of Pipeline
Finished: SUCCESS
```

работает
теперь добавляем часть пайплайна для того чтобы он выкладывал на докер хаб

```
pipeline
{
   environment {
      registryCredential = 'dockerhub'
      dockerImage = ''
   }
        options
        {
                timestamps()
        }
        agent none
        stages
        {
                stage('Check scm')
                {
                        agent any
                        steps
                        {
                                checkout scm
                        }
                } // stage Build
                stage('Build')
                {
                        steps
                        {
                                echo "Building ...${BUILD_NUMBER}"
                                echo "Build completed"
                        }
                } // stage Build
                stage('Test')
                {
                        agent
                        {
                                docker
                                {
                                        image 'python:3.8.6-slim'
                                        args '-u=|"root|"'
                                }
                        }
                        steps
                        {
                                sh 'pip install --no-cache-dir -r ./requirements.txt'
                                sh 'python3 unitTest.py'
                        }
                        post
                        {
                                always
                                {
                                        junit 'test-reports/*.xml'
                                }
                                success
                                {
                                        echo "Application testing successfully completed "
                                }
                                failure
                                {
                                        echo "Oooppss!!! Tests failed!"
```

```
                                }
                        } // post
                } // stage Test
                stage('Docker Publish')
                {
        agent any
        when {
            expression {
                currentBuild.result == null || currentBuild.result == 'SUCCESS'
            }
        }
        steps {
            echo "Application Publishing"
            checkout scm
            script {
                def customImage = docker.build("docker-test:${env.BUILD_ID}")
                docker.withRegistry('',registryCredential )
                {
                    customImage.push()}
                }

        }
                } // stage Docker Publish
        } // stages
}
```

результат соответствует ожиданиям:

```
14:56:26  114ca5b7280f: Preparing
14:56:26  a90a4e55d4e6: Waiting
14:56:26  cc343e3cd1d7: Waiting
14:56:26  c4a6d8ca5d2c: Waiting
14:56:26  475b4eb79695: Waiting
14:56:26  059ed1793a98: Waiting
14:56:26  f3be340a54b9: Waiting
14:56:26  712264374d24: Waiting
14:56:26  114ca5b7280f: Waiting
14:56:26  7ca07421f35c: Waiting
14:56:27  denied: requested access to the resource is denied
[Pipeline] }
[Pipeline] // withDockerRegistry
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // timestamps
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] End of Pipeline
ERROR: script returned exit code 1
Finished: FAILURE
```

Выводы: навчилися створювати елементарний ковеєр автоматизованого випуску програмного забезпечення (application delivery pipeline).