Практическая работа 5

«Создание Jenkins pipeline для сборки и публикации Docker-образа в Docker Hub»

Цели задания:

Научиться создавать CI/CD pipeline с использованием Jenkins для автоматизации процесса сборки Docker-образа из исходного кода и его последующую публикацию в Docker Hub.

Обратите внимание: успешность выполнения данного задания зависит от предварительной подготовки и завершения предыдущих этапов лабораторных работ, включая развертывание Jenkins с использованием Vagrant и развертывание веб-приложения с помощью Docker. Убедитесь, что вы завершили все перечисленные выше этапы, прежде чем приступать к выполнению данного задания.

Шаги:

1. **Проверка доступности Jenkins сервера**

   Прежде всего, убедитесь, что Jenkins сервер запущен и доступен. Вы можете открыть браузер и перейти по URL-адресу Jenkins сервера (обычно это http://localhost:8080 или IP-адрес сервера), тем самым вы удостоверитесь, что вы можете получить доступ к его веб-интерфейсу. Кроме того, проверьте, что у вас установлены необходимые плагины для работы с Docker в Jenkins. Для этого перейдите в раздел "Manage Jenkins" -> "Manage Plugins" -> "Installed" и убедитесь, что установлены плагины, которые могут понадобится для работы с Docker. В случае их отсутствия, установите недостающие.

2. **Проверка доступности Docker Engine**

   Для корректной сборки Docker контейнера потребуется наличие предустановленного Docker Engine на виртуальной машине, где запущен Jenkins. Вы можете выполнить команду *docker ps* в терминале на сервере, чтобы убедиться, что Docker Engine запущен и доступен для использования. Если проверка выявила отсутствие данного компонента, руководствуясь инструкцией ниже выполните установку и настройку: https://docs.docker.com/engine/install/

3. **Создание аккаунта на Docker Hub**

   Еще одним обязательным этапом перед созданием Jenkins pipeline для сборки и публикации Docker-образа в Docker Hub является регистрация аккаунта на https://hub.docker.com. Вот пошаговая инструкция по регистрации:

- Откройте браузер и перейдите по адресу https://hub.docker.com.
- На главной странице Docker Hub найдите и нажмите на кнопку «Sign up».
- Введите запрашиваемую информацию в форму регистрации или выберите один из вариант регистрации с помощью сторонних сервисов. После заполнения формы регистрации вам могут потребоваться дополнительные действия по подтверждению вашего аккаунта. Это может включать в себя подтверждение по электронной почте или решение капчи.
- После завершения регистрации выполните вход в свой новый аккаунт на Docker Hub. После входа в аккаунт убедитесь, что ваш профиль заполнен корректно.
- По окончанию регистрации вам необходимо создать личный репозиторий. Обратите внимание, что при создании репозитория вам необходимо придумать уникальное наименование. Вы также можете создавать дополнительные организации и репозитории в соответствии с вашими потребностями.
- Создайте уникальный токен в вашем профиле для последующего доступа к вашему репозиторию указав соответствующий уровень доступа. Значение токена сохраните, так как оно пригодится в дальнейшем.

После завершения этих шагов ваш аккаунт на Docker Hub будет создан, и вы будете готовы использовать его для публикации и управления вашими Docker-образами.

Примечание: шаги 1-3 являются важными предварительными настройки вашего рабочего окружения которые понадобятся далее. Не приступайте к выполнению следующих шагов если они не выполнены в полной мере.

Примечание: шаги 1-2 можно автоматизировать с помощью дополнения соответствующих строк в файл provision.sh. Эта часть является не обязательной к выполнению, но в качестве дополнительного задания может вам дать возможность заработать дополнительный бал при оценивании работы.

4. **Предоставьте Jenkins доступ к проекту GitLab**

- Создайте личный токен доступа, чтобы использовать его для всех интеграций Jenkins этого пользователя.
- Установите область действия токена доступа **API**.
- Скопируйте значение токена доступа, чтобы настроить сервер Jenkins.

5. **Настройте сервер Jenkins**

Установите и настройте плагин Jenkins для авторизации подключения к GitLab.

1. На сервере Jenkins выберите **Manage Jenkins > Manage Plugins**.

2. Выберите вкладку **Available**. Найдите **gitlab-plugin** и выберите его для установки.
3. Выберите **Manage Jenkins > Configure System**.
4. В разделе **GitLab** выберите **Enable authentication for '/project' end-point**.
5. Выберите **Add**, затем выберите **Jenkins Credential Provider**.
6. Выберите **GitLab API token** в качестве типа токена.
7. В **API Token** вставьте значение токена доступа, скопированное из GitLab, и выберите **Add**.
8. Введите URL-адрес сервера GitLab в **GitLab host URL**.
9. Чтобы проверить соединение, выберите **Test Connection**.

Итог выполнения данных пунктов приведен на рисунке 1.

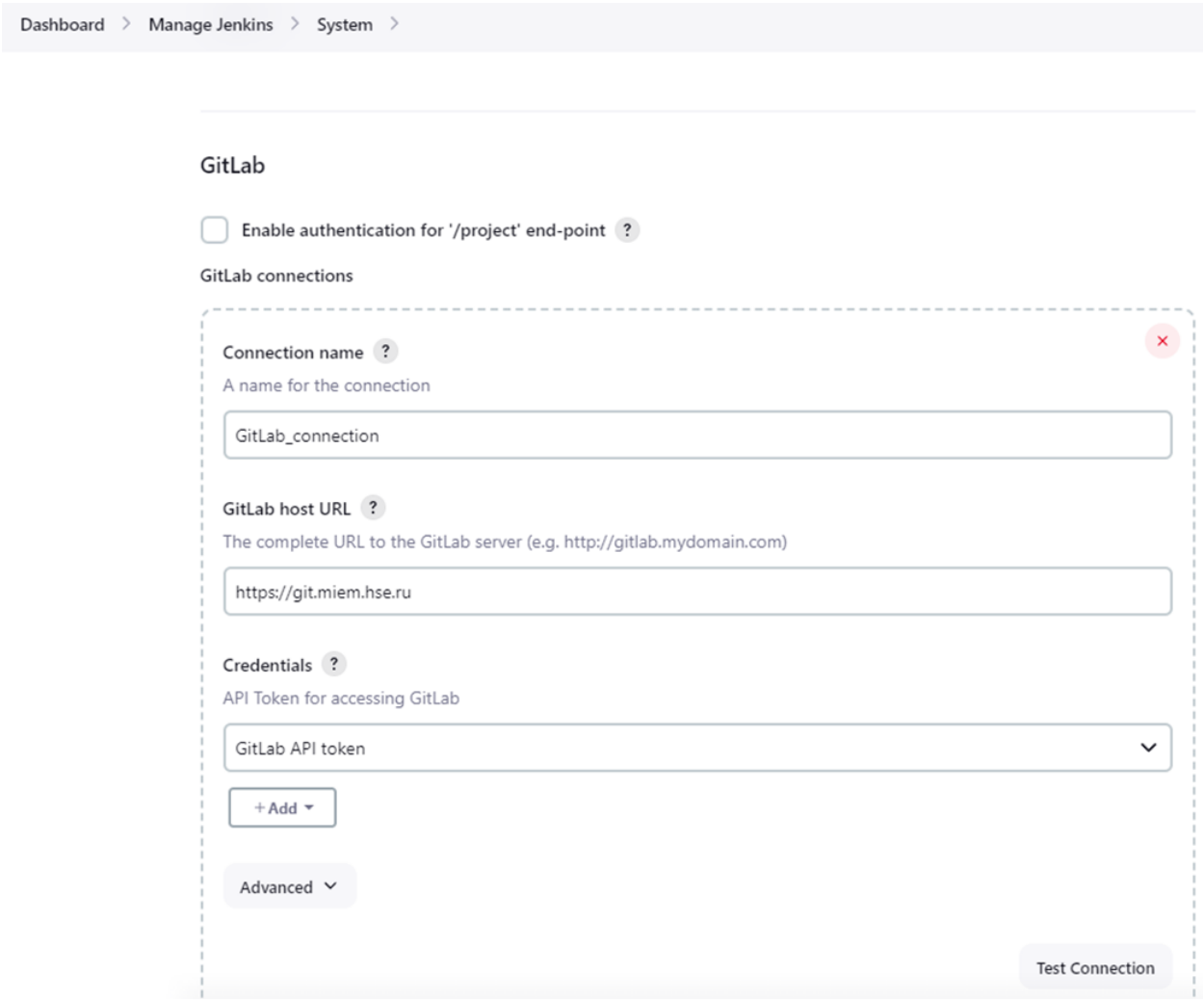

Рисунок 1. Скриншот примера конфигурации.

6. Настройте проект Jenkins (Job + Pipeline)

Настройте проект Jenkins, на котором вы собираетесь запустить свою сборку.

1. В своем экземпляре Jenkins выберите **New Item**.
2. Введите название проекта.
3. Выберите **Freestyle** или **Pipeline** и нажмите **OK** . Вам следует выбрать фристайл-проект, поскольку плагин Jenkins обновляет статус сборки на GitLab. В проекте конвейера необходимо настроить скрипт для обновления статуса в GitLab.
4. Выберите подключение к GitLab из раскрывающегося списка.
5. Выберите **Build when a change is pushed to GitLab.**
6. Установите следующие флажки:
   - **Accepted Merge Request Events**
   - **Closed Merge Request Events**
7. Укажите, как статус сборки передается в GitLab:
   - Если вы создали **freestyle project**, в разделе **Post-build Actions** выберите **Publish build status to GitLab**.
   - Если вы создали **pipeline project**, вам необходимо использовать сценарий Jenkins Pipeline для обновления статуса в GitLab.

Сценарий **pipeline**, должен содержать следующие этапы:

- Checkout - получите исходный код из репозитория для последующих шагов.
- Сборка Docker образа (Docker Build) - используя Docker, соберите Docker-образ из исходного кода вашего приложения.
- Публикация в Docker Hub (Docker Push) - опубликуйте собранный Docker-образ в вашем аккаунте Docker Hub.
- Проверка выполнения - запустите созданный pipeline в Jenkins и убедитесь, что он успешно собирает Docker-образ и публикует его в Docker Hub.

Ниже представлен пример **pipeline** для сборки и публикации Docker-образа в Docker Hub:

```
pipeline {
  agent any

  environment {
    DOCKER_HUB_CREDENTIALS = credentials('docker-hub-credentials')
    DOCKER_IMAGE_NAME = 'your-docker-username/your-image-name'
    DOCKERFILE_PATH = 'path/to/your/Dockerfile'
  }

  stages {
    stage('Checkout') {
```

```
            steps {
                git 'https://github.com/your/repo.git'
            }
        }

        stage('Docker Build') {
            steps {
                script {
                    docker.build(DOCKER_IMAGE_NAME, "-f ${DOCKERFILE_PATH} .")
                }
            }
        }

        stage('Docker Push') {
            steps {
                script {
                    docker.withRegistry('https://index.docker.io/v1/', DOCKER_HUB_CREDENTIALS)
{
                        docker.image(DOCKER_IMAGE_NAME).push('latest')
                    }
                }
            }
        }
    }

    post {
        success {
            echo 'Docker image build and push successful!'
        }
        failure {
            echo 'Docker image build or push failed!'
        }
    }
}
```

Примечание: для корректной работы примера представленного выше **pipeline** необходимо, создать **credentials** для **DokerHub** с соответствующим наименованием **docker-hub-credentials**, а также указать правильные имена и пути в **environment** детективе.

7. Проверка работы pipeline

С помощью команды **Build Now** из веб-интерфейса Jenkins выполните сборку и публикацию пакета, как на рисунке 2. В случае если сборка прошла не успешно проанализируйте логи вашей сборки и устраните ошибки, как на рисунке 3.

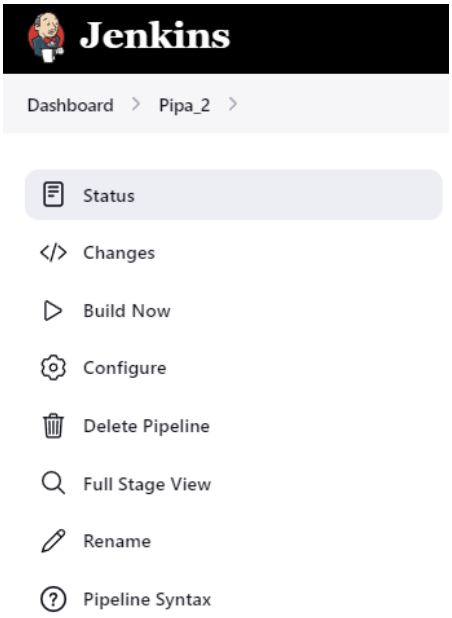

Рисунок 2. Меню запуска pipeline.

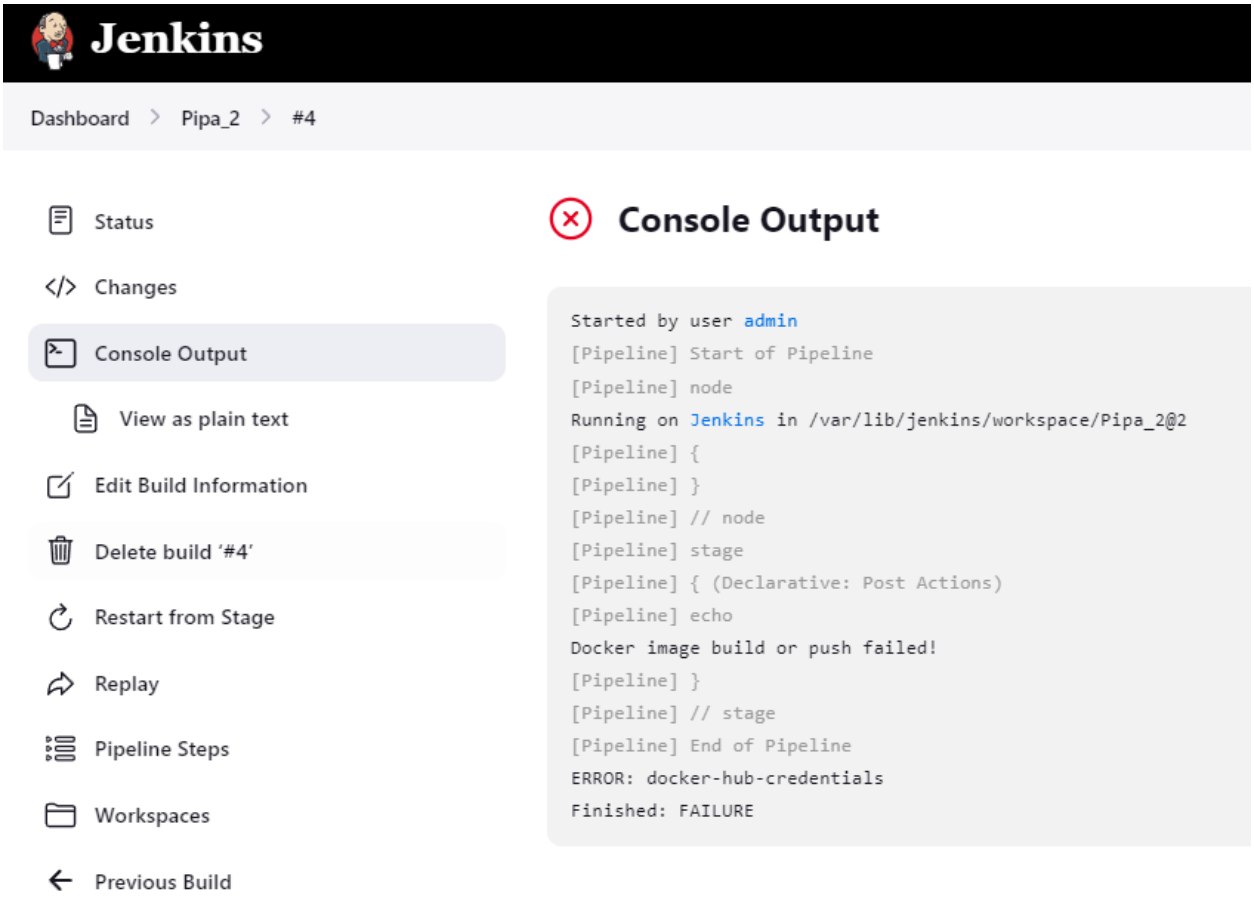

Рисунок 3. Логи сборки(смотрите примечание выше).

8. **Фиксация результатов работы**

   В GitLab необходимо будет сохранить рабочий pipeline в файле Jenkinsfile, а также отчет со скриншотами успешного запуска pipeline и опубликованого Docker image в DockerHub. Выполните фиксацию своих изменений с помощью git add и git commit. Отправьте изменения в удаленный GitLab репозиторий с использованием git push.

9. **Создание Pull Request**

   Создайте Pull Request на слияние вашей ветки в master ветку. В запросе укажите пул выполненных работ по заданию. В качестве Rewiewer укажите преподавателя.