МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»

Институт информатики и кибернетики

Дисциплина: «Инженерия программного обеспечения: DevOps»

Отчет по лабораторной работе №2

«Git and CI +ArgoCD»

Выполнил:

студент группы 6412-100503D

Правдин И.Д.

Самара 2023

# Цель работы

1. Запустить локальный инстанс gitlab.
2. Создать gitlab runner и подключить его.
3. Запустить gitlab pipeline для сборки приложения.
4. Подключить ArgoCD

# Создание виртуальной машины и её настройка

1. Скачан и установлен в VirtualBox ovf image Debian Gitlab-Bitnami (64bit) c следующего сайта <https://bitnami.com/stack/gitlab/virtual-machine>
2. После запуска машины произведен логин под следующим логином/паролем: *bitnami/bitnami.*
3. В настройках домашнего роутера машине назначен статический айпи адрес 192.168.1.39 (сетевая конфигурация созданной виртуальной машины – bridge adapter).
4. В файле /etc/gitlab/gitlab.rb заменены следующие настройки:

|  |
| --- |
| external\_url 'http://192.168.1.39  nginx['redirect\_http\_to\_https'] = false  nginx['ssl\_verify\_client'] = "off" |

1. Gitlab перезагружен с помощью следующей команды.

|  |
| --- |
| sudo gitlab-ctl reconfigure |

1. Получен root пароль с помощью следующей команды:

|  |
| --- |
| sudo cat /home/bitnami/bitnami\_credentials |

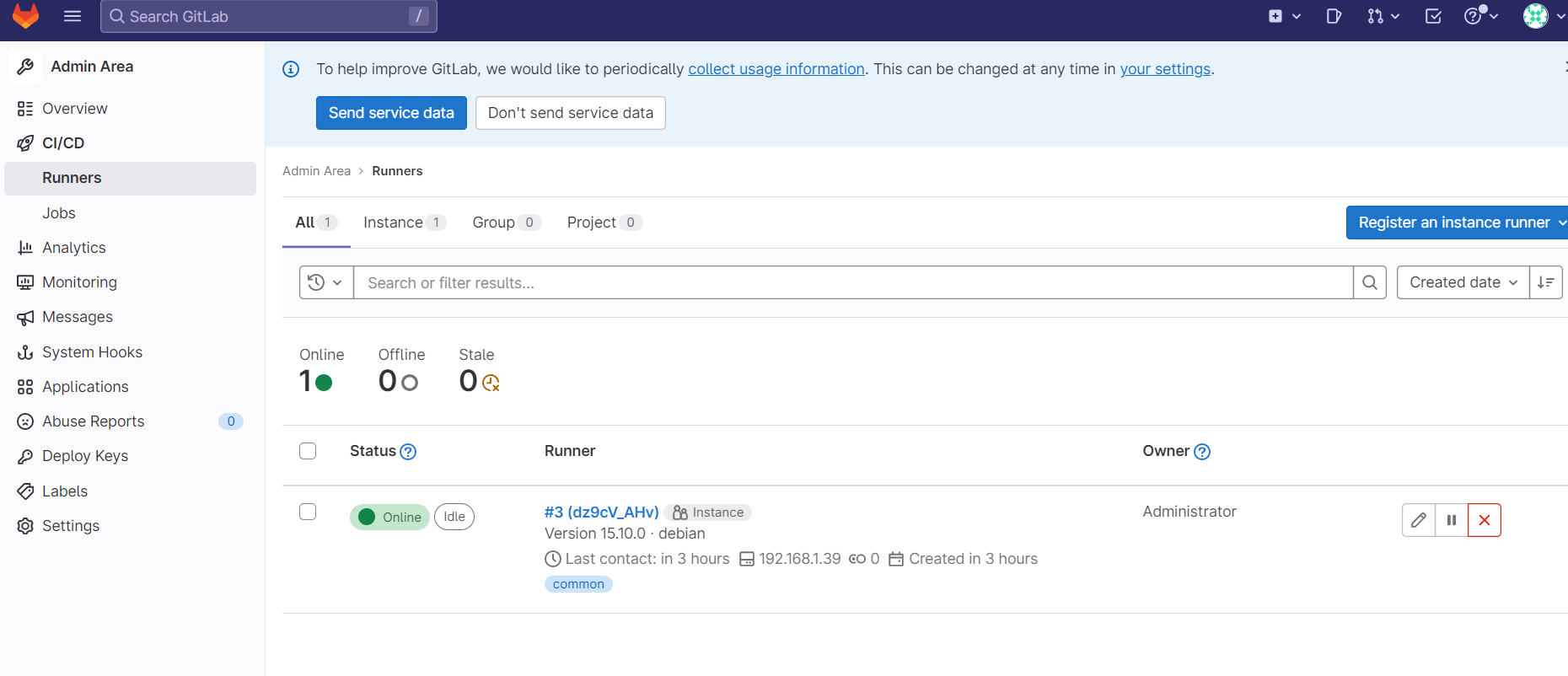
1. Произведен успешный логин под полученными credentials по адресу <https://192.168.1.39>
2. В файле /etc/ssh/sshd\_config на виртуальной машине изменено значение PasswordAuthentication и ChallengeResponseAuthentication на yes.
3. Запущен ssh сервер.

|  |
| --- |
| sudo rm -f /etc/ssh/sshd\_not\_to\_be\_run  sudo systemctl enable ssh  sudo systemctl start ssh |

# Создать gitlab runner и подключить его.

1. Установлен gitlab runner на виртуальной машине, где запущен инстанс gitlab по следующей инструкции:

|  |
| --- |
| # Download the binary for your system  sudo curl -L --output /usr/local/bin/gitlab-runner https://gitlab-runner-downloads.s3.amazonaws.com/latest/binaries/gitlab-runner-linux-amd64  # Give it permission to execute  sudo chmod +x /usr/local/bin/gitlab-runner  # Create a GitLab Runner user  sudo useradd --comment 'GitLab Runner' --create-home gitlab-runner --shell /bin/bash  # Install and run as a service  sudo gitlab-runner install --user=gitlab-runner --working-directory=/home/gitlab-runner  sudo gitlab-runner start  sudo gitlab-runner register --url http://192.168.1.39/ --registration-token fBD1TMnDV4W597c7YL63 |

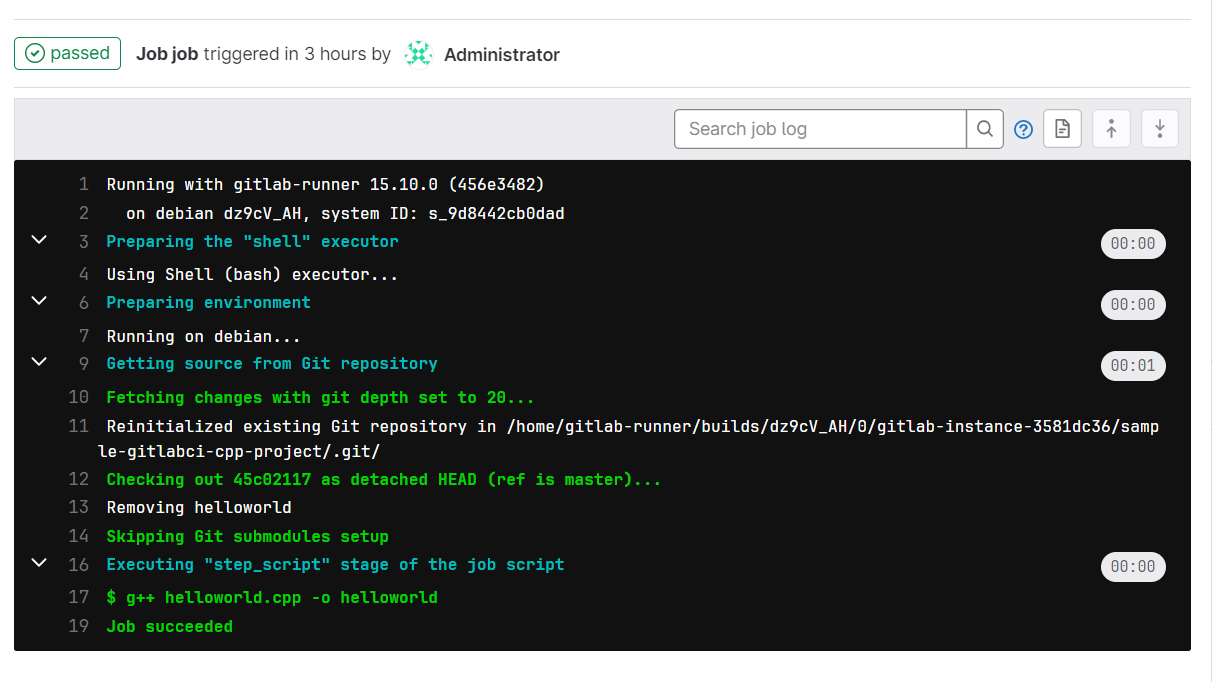
1. Runner появился в списке доступных:  
   
2. В качестве дополнительных настроен был установлен g++ и git.

|  |
| --- |
| sudo apt-get install –y git g++ |

# Запустить gitlab pipeline для сборки приложения.

1. Склонирован и загружен sample проект из следующего репозитория <https://github.com/olindata/sample-gitlabci-cpp-project>
2. Содержимое файла .gitlab-ci.yml заменено на следующее:

|  |
| --- |
| job:    script:          - g++ helloworld.cpp -o helloworld    tags:      - common # Где common это тег, присвоенный runner. |

1. Запуск пайплайна прошёл успешно.  
   

# Запуск ArgoCD

1. ArgoCD будем устанавливать в кластер из лабораторной работы №1.   
   Выполним следующие команды на машине, где запущен кластер:

|  |
| --- |
| kubectl create namespace argocd  kubectl apply -n argocd -f https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml |

1. После того как все deployments будут в ready статусе сделаем порт-форвард к сервису, для этого выполним следующую команду:

|  |
| --- |
| kubectl port-forward svc/argocd-server -n argocd 8080:443 --address 0.0.0.0 |

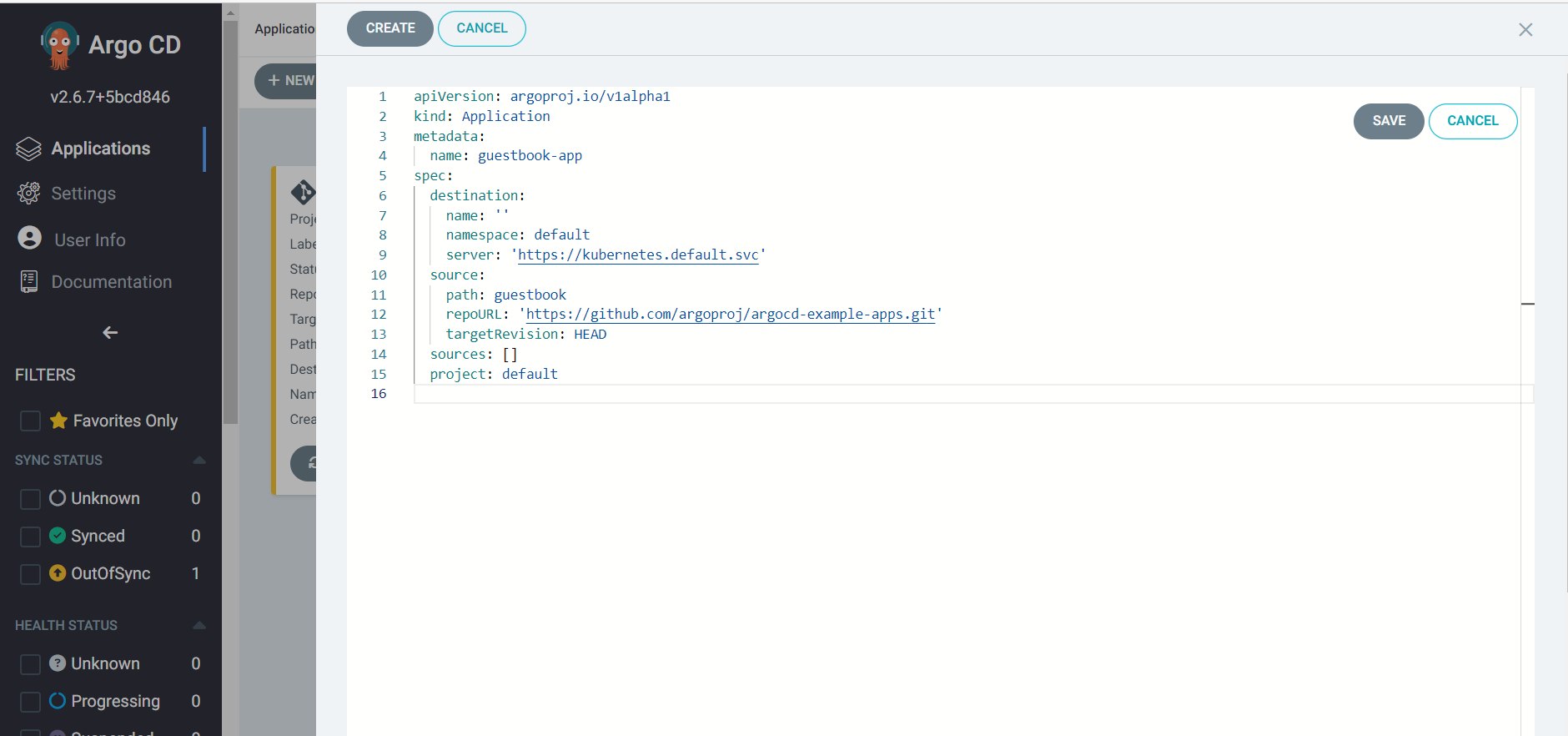
После этого UI ArgoCD доступен по адрессу 192.168.1.46:8080

1. Получим пароль для пользователя admin, для этого выполним следующую команду:

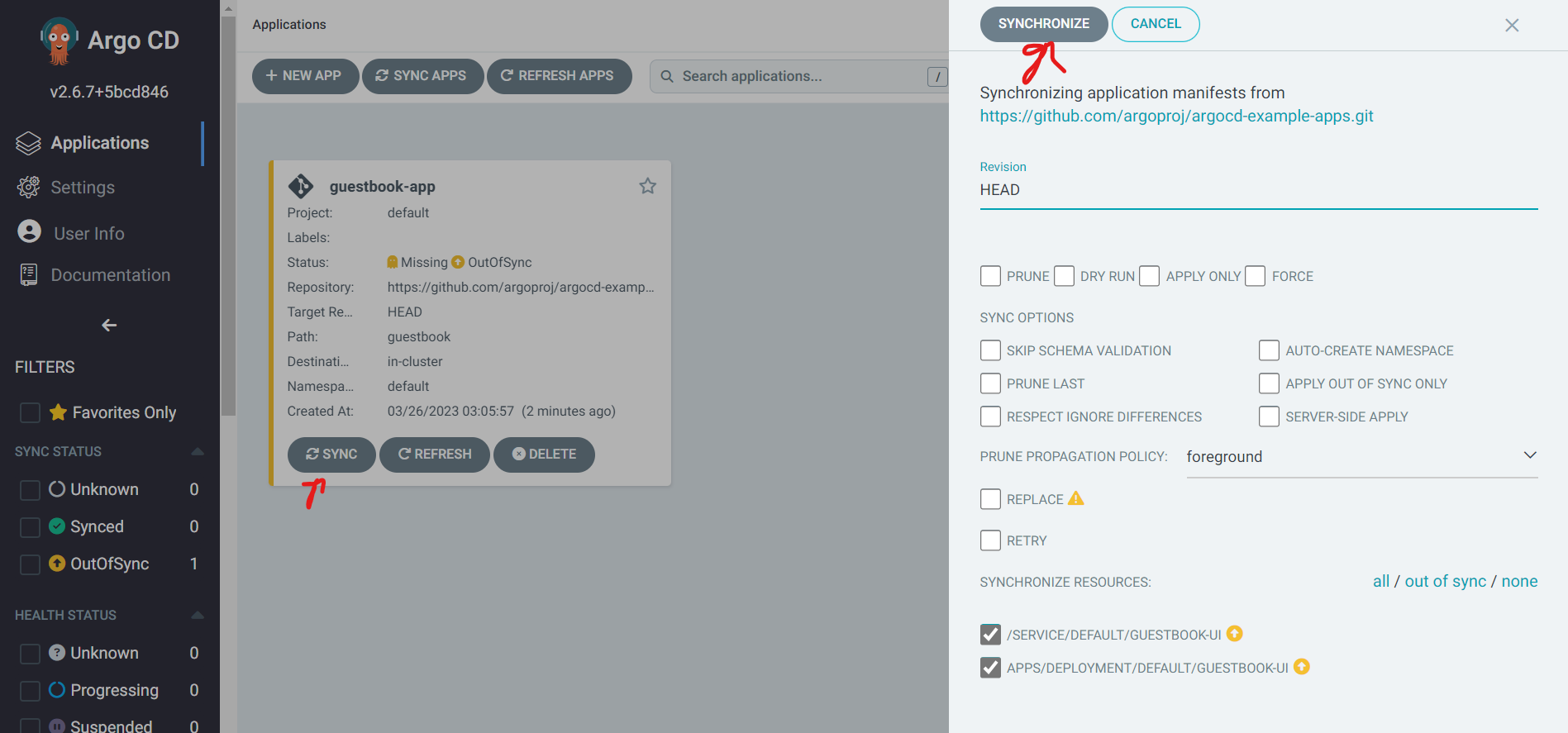
|  |
| --- |
| kubectl get secret argocd-initial-admin-secret -o jsonpath="{.data['password']}" -n argocd | base64 --decode |

1. Далее создадим приложение в ArgoCD через UI, используя следующий манифест yaml файл:

|  |
| --- |
| apiVersion: argoproj.io/v1alpha1  kind: Application  metadata:    name: guestbook-app  spec:    destination:      name: ''      namespace: default      server: 'https://kubernetes.default.svc'    source:      path: guestbook      repoURL: 'https://github.com/argoproj/argocd-example-apps.git'      targetRevision: HEAD    sources: []    project: default |



1. Задеплоим приложение в кластер, нажав на SYNC в UI



1. Сделаем port-forward приложения следующей командой

|  |
| --- |
| kubectl port-forward svc/guestbook-ui -n default 8888:80 --address 0.0.0.0 |

Откроем приложение в браузере  
