Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа Группа 21.Б11-мм

Настройка CI/CD для сайта кафедры системного программирования

Калашников Матвей Михайлович

Отчёт по учебной практике
в форме «Производственное задание»

Научный руководитель:
старший преподаватель каф. СП Я.А. Кириленко

Консультант:
старший преподаватель каф. ИАС К.К. Смирнов

Санкт-Петербург
2023

Оглавление

Введение

В современном мире веб-сайты являются неотъемлемой частью образовательного процесса, поскольку они предоставляют студентам и абитуриентам полезную информацию. Один из таких ресурсов — сайт кафедры. На нем можно найти новости, архив курсовых и дипломных работ, темы учебных практик и другую полезную информацию. Однако чтобы ресурс был актуальным и удобным, необходимо своевременно обновлять его функциональность и содержание. Без автоматизации, развертывание может быть неудобным процессом. Разработчики должны вручную переносить код на сервер и перезапускать приложение. В таком случае развертывание зависит от определенных людей, отнимает время и уменьшает количество релизов. Для того чтобы упростить этот процесс, можно использовать технологии, которые позволяют быстро вносить изменения в код и автоматизируют развертывание новых версий сайта.

Одной из таких технологий является CI/CD (Continuous Integration and Continuous Delivery/Deployment), которая представляет собой последовательность шагов, автоматизирующих процесс разработки, тестирования и развертывания программного продукта. CI/CD помогает улучшать качество кода, а также автоматически развертывать готовый продукт на сервере.

В данной работе будет выбран подход к реализации CI/CD, а также рассмотрен процесс его настройки, для сайта кафедры системного программирования, с использованием существующих инструментов и сервисов. Также будут проанализированы преимущества и недостатки выбранного подхода.

1. Постановка задачи

Целью работы является перенос кода в репозиторий сайта кафедры системного программирования и настройка CI/CD для него. Для её выполнения были поставлены следующие задачи:

1. Перенести код в основной репозиторий сайта кафедры.
2. Провести обзор существующих решений по CI/CD выбрать набор инструментов и сервисов.
3. Настроить процесс CI/CD для сайта кафедры.

2. Обзор

CI/CD — это ключевые практики DevOps, которые помогают повысить качество кода, увеличить количество релизов и их частоту. Важность и эффективность CI/CD можно оценить по частоте его использования. Более 70% организаций используют CI/CD в своей работе [7]. В соответствии с другим исследованием 44% разработчиков регулярно используют CI/CD, причем 22% из них внедрили этот инструмент в течение последнего года [2]. Такая потребность привела к существованию большого количества инструментов на рынке CI/CD.

2.1. Обзор инструментов

Используя данные исследования JetBrains [2], можно выделить несколько наиболее популярных инструментов, таких как GitHub Actions, Jenkins и GitLab. У каждого из них есть свои преимущества и недостатки, а также различные способы интеграции с другими сервисами и платформами. Выбор оптимального инструмента зависит от многих факторов, таких как размер и сложность проекта, используемые технологии и пр.

GitHub Actions наиболее часто используется для личных проектов или небольших продуктов, из-за своей простоты в настройке и использования. Нет необходимости самостоятельно управлять сервером и агентами сборки, так как GitHub предоставляет виртуальные машины для запуска workflow. Благодаря интеграции с GitHub, можно удобно добавлять и редактировать workflow с помощью YAML-файлов, которые хранятся в репозитории. Также можно использовать готовые действия из GitHub Marketplace.

Jenkins [6] — это бесплатный инструмент с открытым исходным кодом, поддерживаемый сообществом. Чаще всего он используется для организации CI/CD в компаниях [2], так как можно гибко настроить конвейер. Как плюсом, так и минусом является необходимость поддержания собственной инфраструктуры, что способствует обеспечению безопасности, но усложняет настройку. Также существует множество плагинов, облегчающих настройку CI/CD.

GitLab CI [5] — облачный инструмент CI/CD, который встроен в платформу GitLab. Зачастую используется компаниями. Предлагает опыт использования, аналогичный GitHub Actions.

Для реализации CI/CD был выбран GitHub Actions, так как все исходники сайта находятся на GitHub, а также он достаточно легок в настройке.

Существует несколько подходов к реализации развертывания. Одним из них является использование Docker, для упаковки приложения в контейнер. Преимуществом является переносимость приложения на все системы, поддерживающие Docker. На тестовом сервере было протестировано развертывание сайта с помощью Docker, но после этого выяснилось, что Docker не поддерживает Ubuntu 16.04 [3]. Кроме этого нет явной необходимости во многих преимуществах Docker, а настраивать его сложнее. Поэтому для развертывания был выбран другой подход, который рассмотрен в следующем разделе.

2.2. Используемые технологии

GitHub Actions [4] — это облачная платформа CI/CD, позволяющая автоматизировать конвейер сборки, тестирования и развертывания. Есть возможность запускать необходимые workflows на различных операционных системах (Linux, Windows, mac OS), используя виртуальные машины, предоставляемые GitHub. Кроме того GitHub Actions можно интегрировать с другими функциями GitHub, такими как Issues, Pull Requests, Push, GitHub Pages и др. workflows определяются с помощью YAML-файлов, которые содержат набор задач. В публичных репозиториях сервис можно использовать бесплатно.

3. Описание решения

Код был перенесен из личного репозитория в основной репозиторий[1].

Реализованный CI/CD представляет собой несколько YAML файлов, которые автоматизируют процесс поддержания стиля кода, развертывания новой версии на сервере, и мониторинга состояния работы сайта.

3.1. Линтер

Для форматирования кода на Python используется линтер Black [1]. Black — это инструмент, который автоматически приводит код к единому стилю, основанному на стандарте PEP 8. Это помогает улучшить качество и читаемость кода. Процесс проверки кода запускается после каждого push в репозиторий.

3.2. Мониторинг

Для периодической проверки доступности сайта используется workflow, который при помощью curl проверяет работоспособность приложения. Каждые 15 минут он отправляет GET запрос на главную страницу se.math.spbu.ru, извлекает код ответа и проверяет его. Если код ответа отличается от 200, то это означает, что сайт недоступен или работает с ошибками. В этом случае проверка считается неудачной и в беседу в телеграм отправляется сообщение, где указывается код ответа.

3.3. Развертывание

Изначально для развертывания планировалось использовать Docker, так как это позволяет обеспечить унифицированное и изолированное окружение для приложения. Был настроен и протестирован сценарий развертывания, с использованием GitHub Actions, GitHub Container registry,

[1]https://github.com/spbu-se/spbu_se_site

и контейнера Watchtower или appleboy/ssh-action action. После пуша тега запускался workflow, который собирал образ контейнера и загружал его в GitHub Container registry. После этого контейнер Watchtower загружал новый образ и запускал его.

Однако от такой схемы развертывания было решено отказаться, так как у нее есть несколько недостатков. Использование Docker избыточно, так как он потребляет дополнительные ресурсы, сложнее в настройке и может являться дополнительным риском для безопасности, при этом его преимущества, такие как масштабируемость и переносимость не используются.

Организация развертывания в репозитории

После того как код в репозитории обновился, необходимо развернуть его на целевом сервере. Для этого используется соответствующий workflow, который запускается после пуша тега, соответствующего формату 'v*.*.*'. Используя сторонний action appleboy/ssh-action процесс подключается к целевому серверу по ssh, но не выполняет никаких команд. Важно упомянуть, что чувствительные данные, такие как ssh ключи, адрес хоста, имя пользователя и пр., для безопасности хранятся в GitHub secrets.

Организация развертывания на сервере

Данная часть работы делалась совместно со Смирновым Кириллом Константиновичем. Мне принадлежит Bash скрипт развертывания новой версии сайта на сервер и конфигурационный ini файл.

Для обеспечения безопасности и изоляции кода скрипта развертывания от репозитория, сервер сконфигурирован таким образом, чтобы при подключении по ssh автоматически выполнялся, хранимый локально на сервере скрипт, который запрашивает изменения из репозитория и перезапускает приложение.

На сервере специально для развертывания создан пользователь deploy, в домашней папке которого находятся файлы сайта, а также deploy bash

скрипт.

Листинг 1: Bash скрипт развертывания новой версии сайта на сервер

```
#!/bin/sh

cd /home/deploy/spbu_se_site
git fetch
git rebase origin/current
.venv/bin/pip install -r requirements.txt
cd src
../.venv/bin/flask db upgrade
sudo /usr/bin/sv restart flask_se
```

В супервизоре процессов веб-сервер запускается следующей командой:

```
exec chpst -u deploy:deploy ../.venv/bin/uwsgi --ini flask_se.ini
```

Вместо runit можно использовать другой супервизор процессов. В качестве сервера используется uwsgi, принимающий на вход следующий ini Файл:

```
[uwsgi]
module = wsgi:app
logto = /var/log/uwsgi/%n.log

master = true
processes = 5
enable-threads = true

socket = 127.0.0.1:4545
chmod-socket = 660
vacuum = true

die-on-term = true
buffer-size=8192
```

Чтобы скрипт развертывания запускался после входа пользователя по ssh необходимо отредактировать файл /etc/ssh/sshd_config и добавить в него строки:

```
Match User deploy
  ForceCommand /home/deploy/deploy
```

В целях безопасности необходимо предоставить пользователю deploy, права суперпользователя только на перезапуск веб-сервера. Для этого нужно записать в /etc/sudoers.d/deploy строку, разрешающую перезапуск сервиса веб сервера, без ввода пароля:

```
deploy ALL=(root) NOPASSWD: /usr/bin/sv restart flask_se
```

Заключение

В результате работы над учебной практикой в течение весеннего семестра были выполнены следующие задачи:

- Код перенесен в основной репозиторий сайта кафедры.
- Проведен обзор подходов и инструментов к организации CI/CD и выбраны подходящие инструменты и сервисы.
- Настроен и опробован процесс CI/CD.

Список литературы

[1] Black documentation. — URL: https://black.readthedocs.io/en/stable/ (дата обращения: 10 сентября 2023 г.).

[2] Continuous Integration Tools for 2023 – Survey Results. — URL: https://www.jetbrains.com/lp/devecosystem-2022/team-tools/ (дата обращения: 10 сентября 2023 г.).

[3] Docker documentation. — URL: https://docs.docker.com/engine/install/ubuntu/ (дата обращения: 10 сентября 2023 г.).

[4] GitHub actions documentation. — URL: https://docs.github.com/actions (дата обращения: 10 сентября 2023 г.).

[5] GitLab documentation. — URL: https://docs.gitlab.com/ (дата обращения: 10 сентября 2023 г.).

[6] Jenkins documentation. — URL: https://www.jenkins.io/sigs/docs/ (дата обращения: 10 сентября 2023 г.).

[7] Stackoverflow developer survey 2023. — URL: https://survey.stackoverflow.co/2023/ (дата обращения: 10 сентября 2023 г.).