МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА № 43

ОТЧЁТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Старший преподаватель | |  |  | | | |  | | М. Д. Поляк |
| должность, уч. степень, звание | |  | подпись, дата | | | |  | | инициалы, фамилия |
| ОТЧЁТ О ЛАБОРАТОРНОЙ РАБОТЕ №7 | | | | | | | | | | |
| Непрерывная интеграция в облаке с использованием GitHub Actions | | | | | | | | | | |
| по дисциплине: Операционные системы | | | | | | | | | | |
|  | | | | | | | | | | |
| РАБОТУ ВЫПОЛНИЛА | | | | | | | | | | |
| СТУДЕНТКА ГР. | 4931 | | |  | 20.05.2022 |  | | Е.Ю. Ильченко | | |
|  |  | | |  | подпись, дата |  | | инициалы, фамилия | | |
|  |  | | |  |  |  | |  | | |

Санкт-Петербург 2022

## Вариант 11

## Цель:

Изучение принципов организации непрерывной интеграции приложений с использованием облачных технологий.

#### Задание:

1. Добавить в репозиторий предыдущей лабораторной работы файл конфигурации .github/workflows/tests.yml, управляющий созданием виртуальной машины в облаке, на которой автоматически будет запускаться сборка и тестирование проекта. Примеры оформления файла конфигурации можно найти в репозиториях лабораторных работ №№1-6, в [официальной документации](https://docs.github.com/en/actions), а также с помощью поиска в Интернете. Задачу (job) в файле конфигурации необходимо назвать test, использование других имен приведет к потере баллов. Команды в разделе steps задачи должны запускать сборку той части предыдущей лабораторной работы, которая в соответствии с вариантом была выполнена в ОС Linux (или Mac OS). Например, если по варианту задания на предыдущую лабораторную работу необходимо было разработать серверную часть приложения под ОС Linux, а клиентскую часть приложения под ОС Windows, то в данной лабораторной работе необходимо осуществить автоматическую сборку серверной части приложения. Сборка должна осуществляться упешно (без ошибок).
2. (**NB**! Этот пункт задания необязателен к выполнению) Разработать как минимум один тест для проверки правильности функционирования приложения, автоматическая сборка которого была обеспечена на предыдущем шаге. Добавить сборку и запуск теста (-ов) в файл конфигурации *tests.yml*.
3. По аналогии с предыдущей работой процесс выполнения задания должен быть разбит на задачи, которые должны быть описаны в разделе Issues репозитория. Решению каждой задачи должен соответствовать как минимум один коммит. Название задач должно начинаться с префикса Lab6:, такой же префикс должен быть в названии коммита (-ов). Коммиты и задачи без префикса засчитаны не будут.

**Описание структуры конфигурационного файла:**

//список событий, на который экшен должен запускаться

|  |
| --- |
| on: [push] |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
| **// список того, что нужно делать. каждый job** |

|  |
| --- |
| jobs: |
| //задача –тест |
|  |

|  |
| --- |
| test: |
| //запуск вирутальной машины |
|  |

|  |
| --- |
| runs-on: ubuntu-latest |
| //ожидание выполнения теста |
|  |

|  |
| --- |
| timeout-minutes: 2 |
|  |

|  |
| --- |
|  |
| //шаги |

|  |
| --- |
| steps: |
| //клонирование репозитория на новую виртуальную машину |
|  |

|  |
| --- |
| - uses: actions/checkout@v2 |
|  |

|  |
| --- |
| - name: Testing client.cpp |
|  |

|  |
| --- |
| shell: bash |
|  |

run: bash tests.sh

**Содержимое написанного конфигурационного файла:**

|  |
| --- |
| on: [push] |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| jobs: |
|  |

|  |
| --- |
| test: |
|  |

|  |
| --- |
| runs-on: ubuntu-latest |
|  |

|  |
| --- |
| timeout-minutes: 2 |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| steps: |
|  |

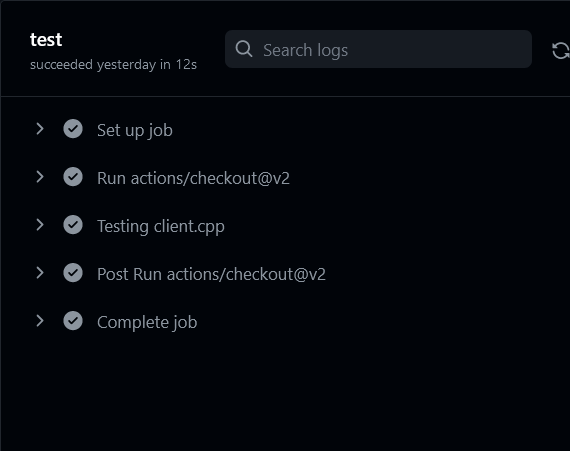
|  |
| --- |
| - uses: actions/checkout@v2 |
|  |

|  |
| --- |
| - name: Testing client.cpp |
|  |

|  |
| --- |
| shell: bash |
|  |

run: bash tests.sh

**Логи сборки проекта в облаке:**

****

**Исходный код тестов**

|  |
| --- |
| #!/usr/bin/env bash |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| g++ client.cpp -o client |
|  |

|  |
| --- |
| if [ $? -ne 0 ] |
|  |

|  |
| --- |
| then |
|  |

|  |
| --- |
| echo "Test 1 failed!" |
|  |

|  |
| --- |
| exit 1 |
|  |

|  |
| --- |
| fi |
|  |

|  |
| --- |
| echo "Test 1 passed" |
|  |

|  |
| --- |
| ./client build fhff5r65645t |
|  |

|  |
| --- |
| if [ $? -eq 0 ] |
|  |

|  |
| --- |
| then |
|  |

|  |
| --- |
| echo "Test 2 failed!" |
|  |

|  |
| --- |
| exit 1 |
|  |

|  |
| --- |
| fi |
|  |

|  |
| --- |
| echo "Test 2 passed" |
|  |

|  |
| --- |
| ./client 1.1.1.1 |
|  |

|  |
| --- |
| if [ $? -eq 0 ] |
|  |

|  |
| --- |
| then |
|  |

|  |
| --- |
| echo "Test 3 failed!" |
|  |

|  |
| --- |
| exit 1 |
|  |

|  |
| --- |
| fi |
|  |

|  |
| --- |
| echo "Test 3 passed" |
|  |

|  |
| --- |
| ./client 127.0.0.1 2701551 |
|  |

|  |
| --- |
| if [ $? -eq 0 ] |
|  |

|  |
| --- |
| then |
|  |

|  |
| --- |
| echo "Test 4 failed!" |
|  |

|  |
| --- |
| exit 1 |
|  |

|  |
| --- |
| fi |
|  |

|  |
| --- |
| echo "Test 4 passed" |
|  |

|  |
| --- |
| g++ server\_test.cpp -o server -std=c++11 |
|  |

|  |
| --- |
| ./server 3425 & sleep 2 |
|  |

|  |
| --- |
| echo "Hello|100" | ./client 127.0.0.1 3425 >> output2.txt |
|  |

|  |
| --- |
| if [[ $(diff output1.txt output2.txt | wc -l) -eq 0 ]] |
|  |

|  |
| --- |
| then |
|  |

|  |
| --- |
| echo "Test 5 passed" |
|  |

|  |
| --- |
| else |
|  |

|  |
| --- |
| echo "Test 5 failed!" |
|  |

|  |
| --- |
| exit 1 |
|  |

fi

**Выводы:**

Изучили принципы организации непрерывной интеграции приложений с использованием облачных технологий.