Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования «Тихоокеанский государственный университет»

Высшая школа кибернетики и цифровых технологий

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

ПО КУРСУ «ОСНОВЫ ПРОГРАММНОЙ ИНЖЕНЕРИИ»

«АВТОМАТИЗАЦИЯ СБОРКИ И РАЗВЕРТЫВАНИЯ (CI/CD)»

Выполнил: студент 2 курса группы ПО(б)-31

Рожин Кирилл Сергеевич

Проверила: ассистент ВШ КЦТ

Балабасова Алиса Дмитриевна

Хабаровск 2025 г.

Цель: освоить настройку автоматической сборки и развертывания кода с использованием GitHub Actions и Jenkins.

Задания:

1. Автоматизировать любой проект с помощью GitHub Actions.

2. Автоматизировать любой проект с помощью Jenkins (или другого CI/CD на выбор).

3. Проверить работу при внесении изменений (скриншоты до/после).

1. Автоматизировать любой проект с помощью GitHub Actions.
2. Создание файла workflow и ci.yml для GitHub Actions (рисунок 1, рисунок 2)

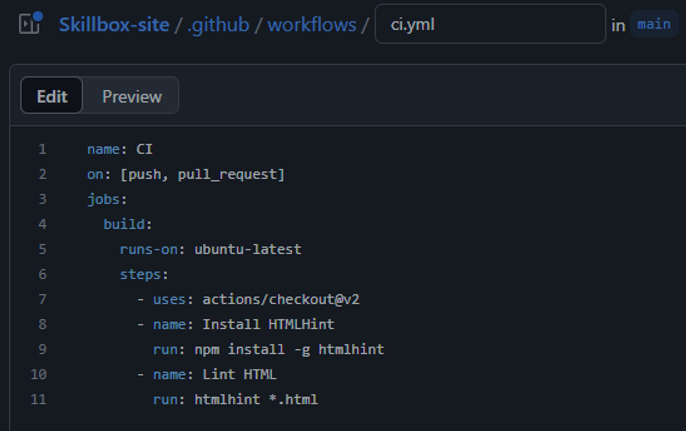


Рисунок 1 – создание workflow, ci.yml

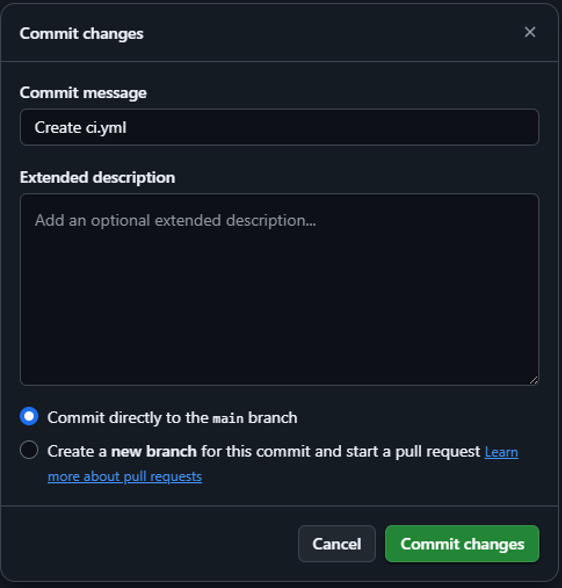


Рисунок 2 – комит ci.yml

1. Проверка выполнения workflow/ci.yml (рисунок 3, рисунок 4).

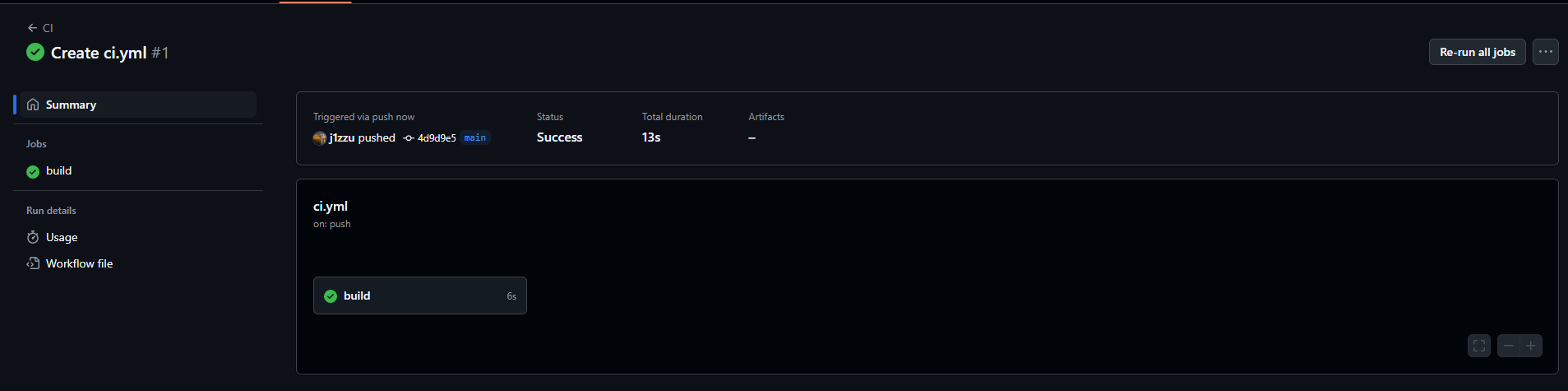


Рисунок 3 – проверка выполнения GitHub Actions

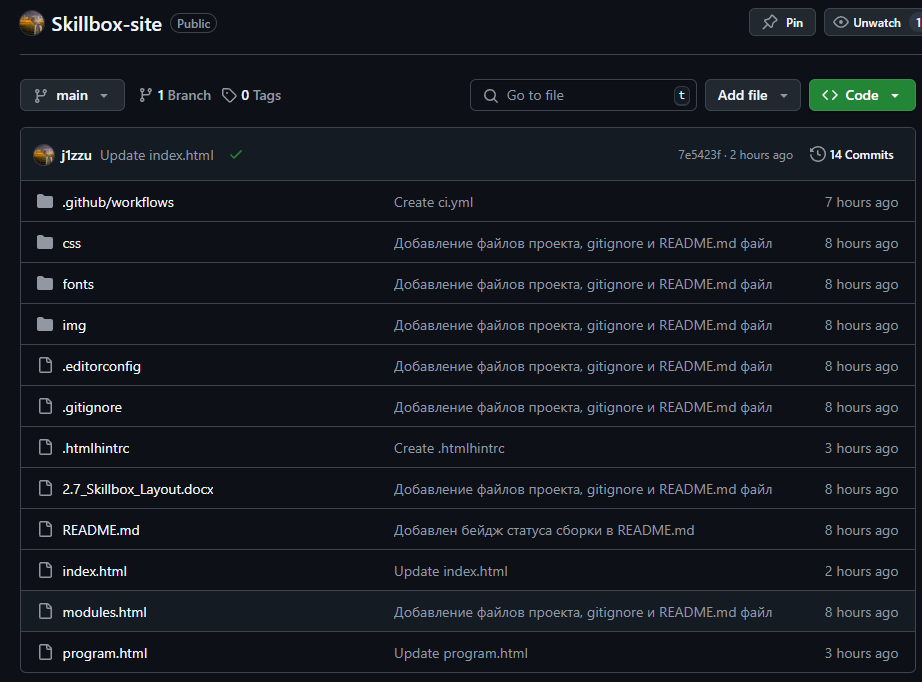


Рисунок 4 – проверка выполнения GitHub Actions

1. Автоматизировать любой проект с помощью Jenkins
2. Добавление конфигурации (рисунок 5).

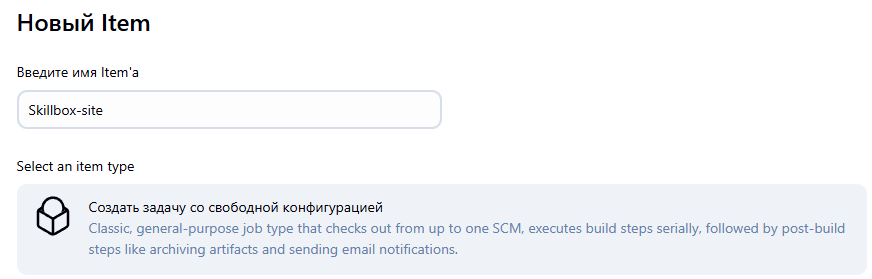


Рисунок 5 – добавление свободной конфигурации

1. Установление связи с репозиторием (рисунок 6)

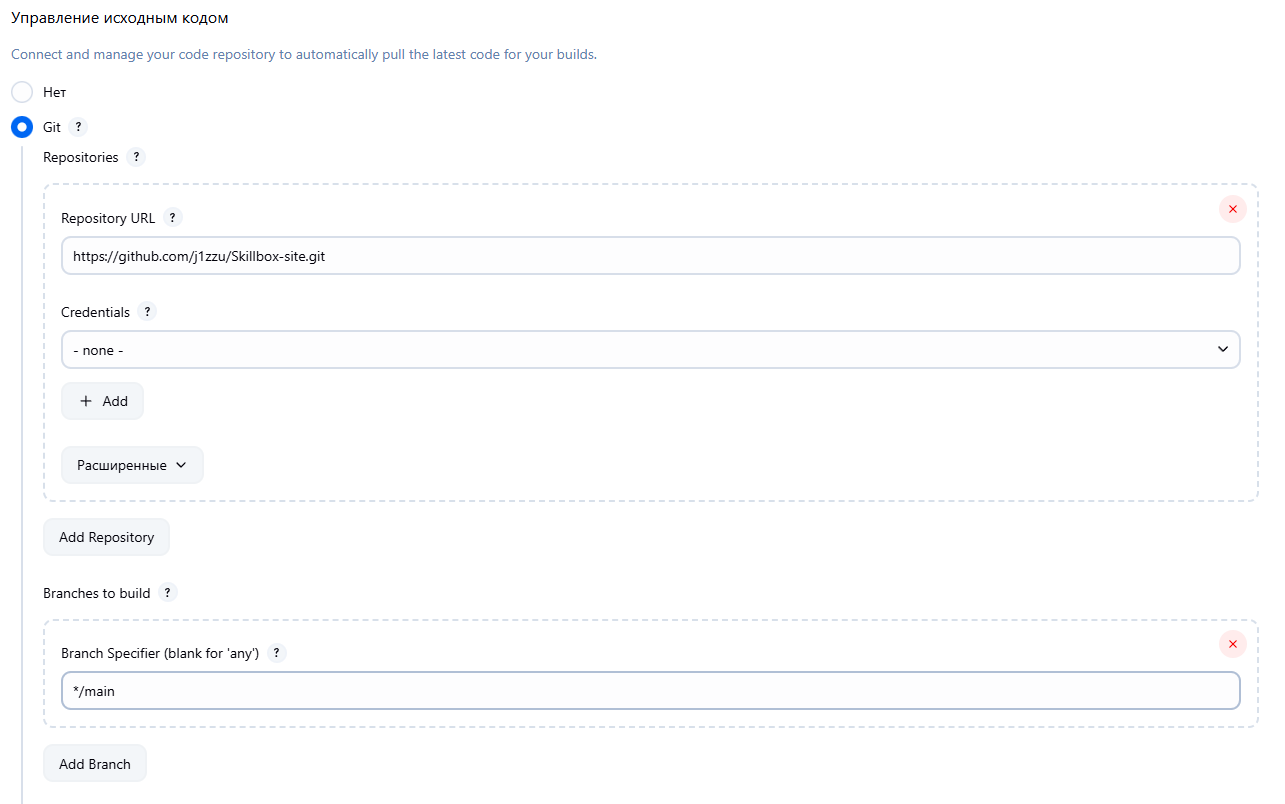


Рисунок 6 – настройка связи с репозиторием

1. Автоматизация запуска сборки (рисунок 7, рисунок 8).

Делаем так, чтобы сборка запускалась, когда Jenkins получит уведомление через вебхук от GitHub

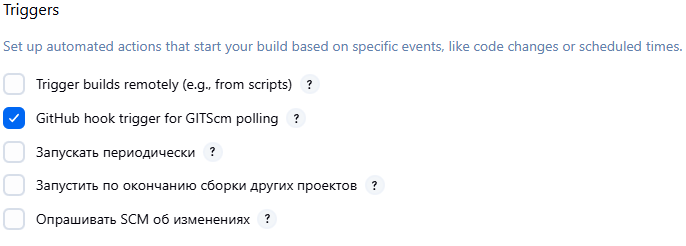


Рисунок 7 – уведомления через вебхук

Написание кода для Windows для установки зависимостей, Node.js и HTMLHint для того, чтобы находились ошибки в HTMl-коде.



Рисунок 8 – код для Windows

1. Проверка работы при внесении изменений.
2. До изменений кода (рисунок 9, рисунок 10):

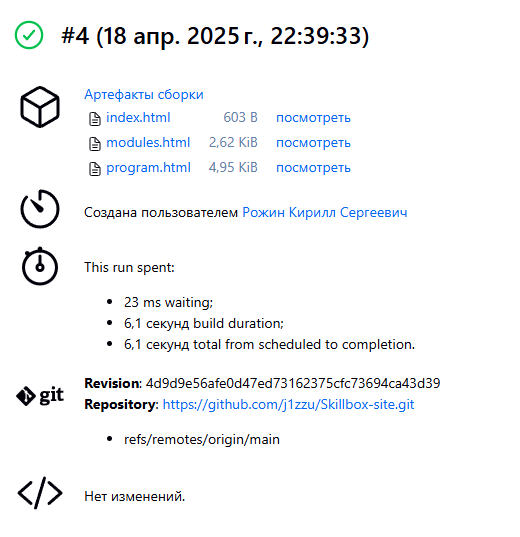


Рисунок 9 – до изменений кода (Jenkins)

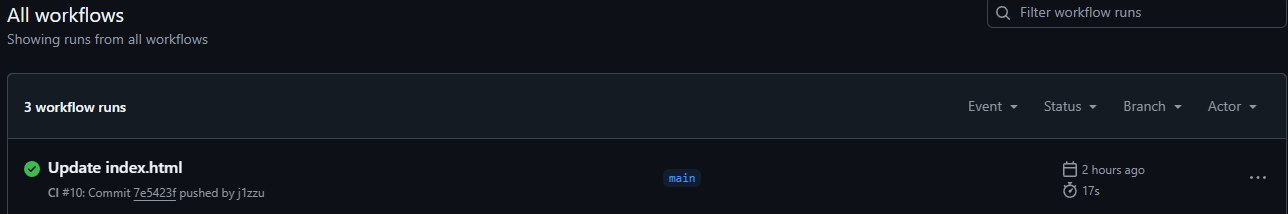


Рисунок 10 – до изменений (GitHub Actions)

1. После изменений кода (рисунок 11, рисунок 12):

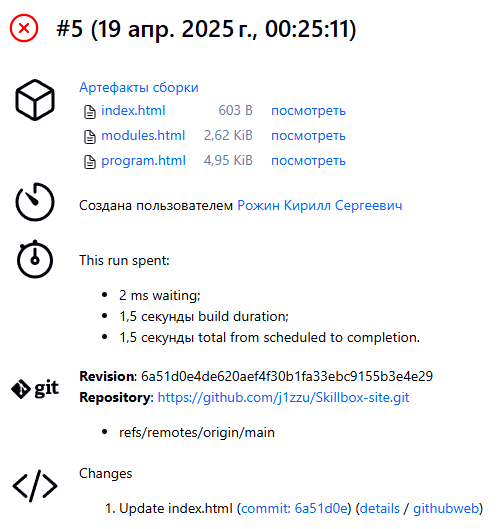


Рисунок 11 – после изменений кода (Jenkins)

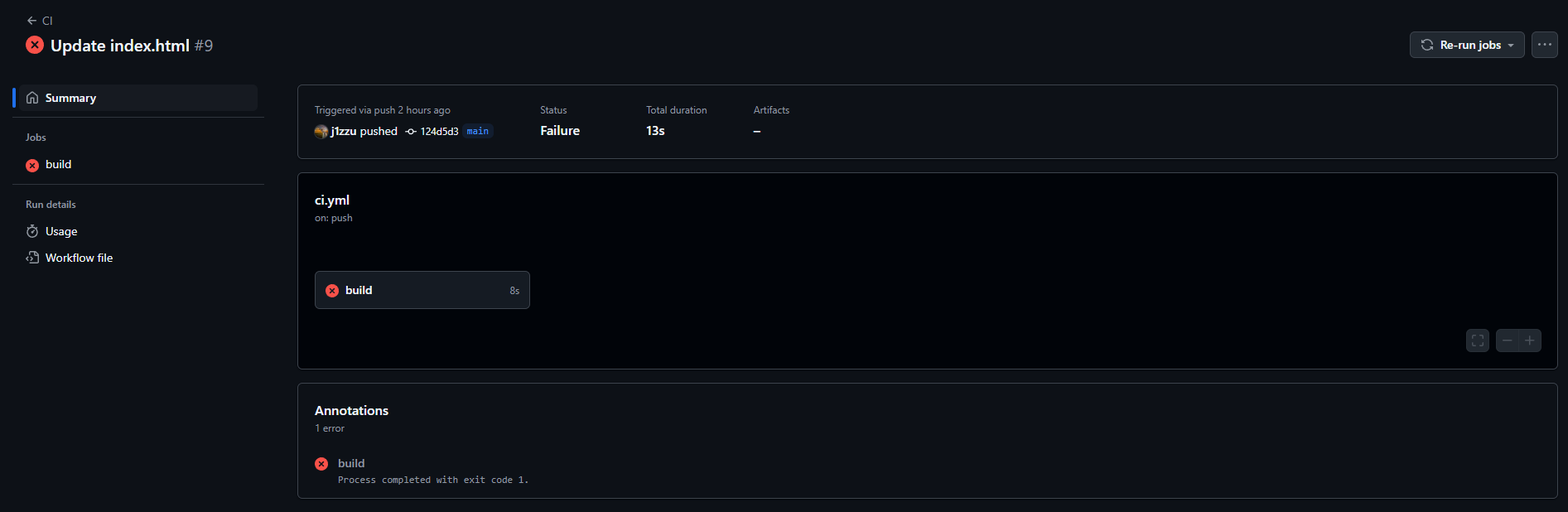


Рисунок 12 – после изменений кода (GitHub Actions)

Вывод: в рамках лабораторной работы была реализована автоматизация процесса проверки HTML-кода для обеспечения его качества с использованием инструментов непрерывной интеграции (CI): GitHub Actions и Jenkins. Проект, выбранный для автоматизации, представляет собой репозиторий на GitHub, содержащий HTML-файлы, которые проверяются на наличие ошибок (например, незакрытых тегов) с помощью инструмента htmlhint. Целью было настроить автоматический запуск проверки при каждом изменении кода (push) и убедиться, что процесс CI/CD корректно реагирует на валидный и невалидный код.