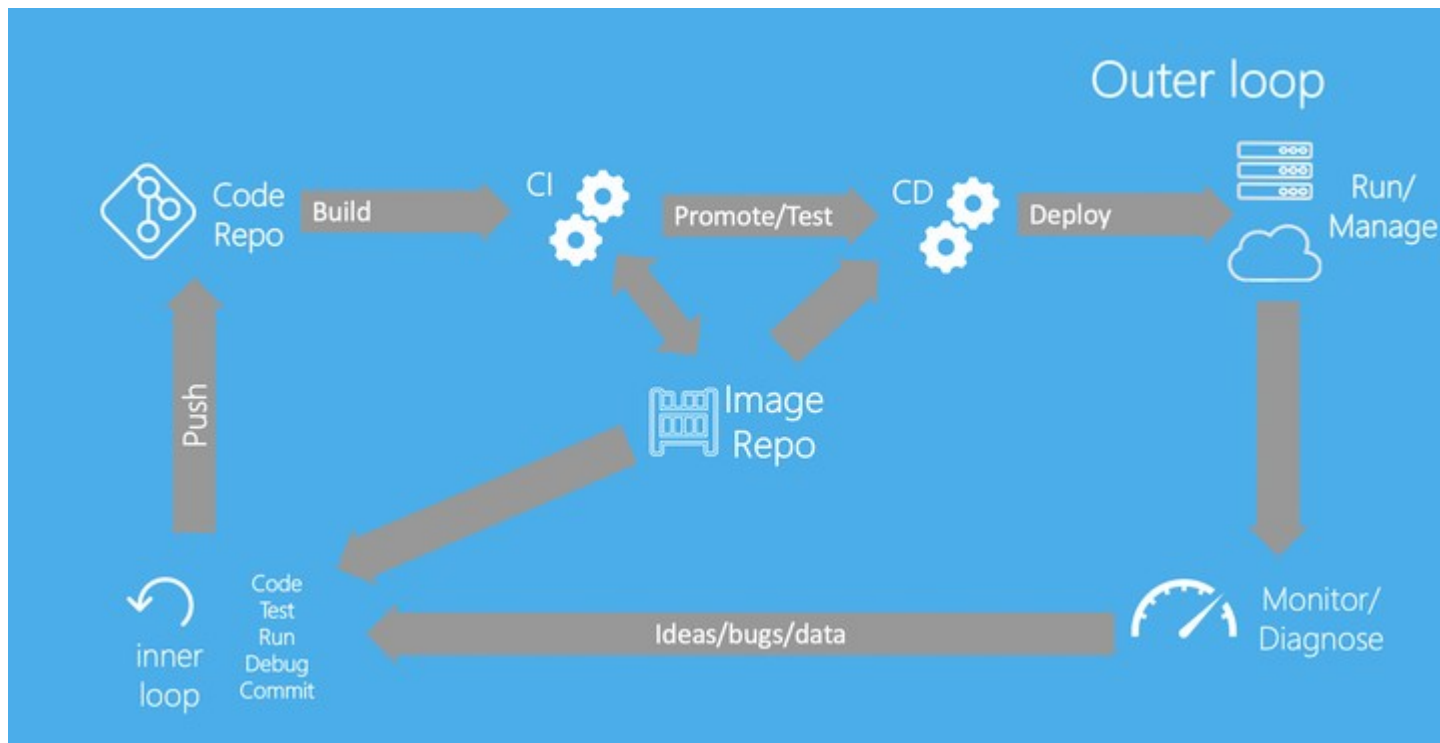
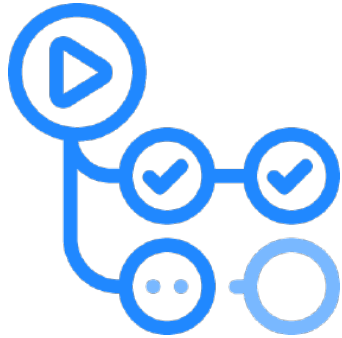


## Krótkie spojrzenie na zagadnienie CI/CD

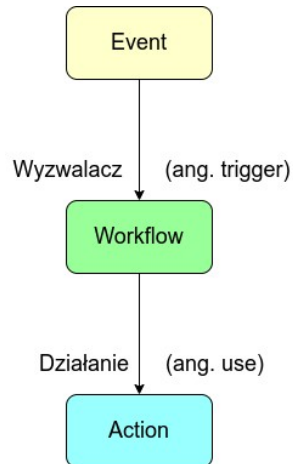


**WAŻNE:**

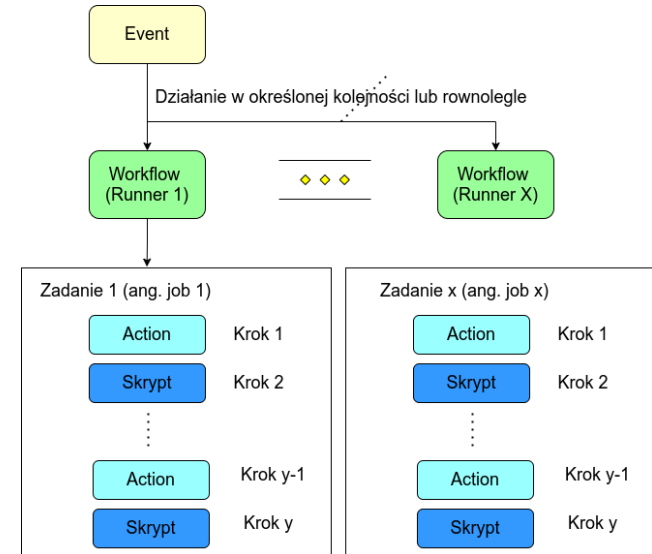
<https://docs.github.com/en/actions/learn-github-actions/usage-limits-billing-and-administration#about-billing-for-github-actions>



- całkowicie zintegrowany z GitHub (np. można działać bez Travis CI),
- Działanie może być powiązane z dowolnym zdarzeniem w ramach posiadanego konta GitHub,
- zbiór definicji “Workflow” rozwijana przez społeczność,
- bogaty (i wciąż rosnący zbiór opisów “Actions”,
- brak ograniczeń na wymagany typ platformy (realizacji działań w oparciu o VM Linux, Windows lub MacOS), język programowania czy też docelowe środowisko chmurowe.



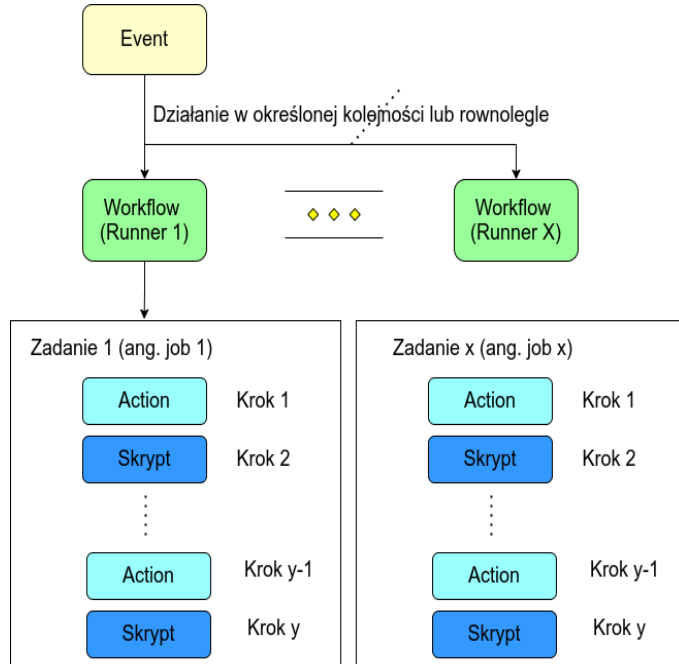
## Realizacja



# GitHub Action – podstawowe pojęcia (1)

**Event** – określone zdarzenie w ramach danego repozytorium, które zostało zdefiniowane jako wyzwalacz w danym workflow.

**Workflow** – zautomatyzowany ciąg działań, realizowany w wyniku wystąpienia określonego zdarzenia (ang. event) w danym repozytorium, uruchamiany manualnie bądź wywołany przez zewnętrzne działania planowe (ang. scheduler).



**Job** – zbiór kroków (ang. Steps) do wykonania w ramach danego workflow. Każdy krok może mieć formę skryptu (ang. script execution) lub predefiniowanej akcji (ang. action run). Poszczególne steps są wykonywane w kolejności wystąpienia i są wzajemnie zależne. Dane w obrębie jednego runnera mogą być współdzielone przez poszczególne kroki.

**UWAGA1:** Dany workflow wykonywany jest w pojedynczym środowisku wykonawczym (ang. Runner) – zazwyczaj VM lub kontener.

**UWAGA2:** Domyślnie poszczególne jobs są niezależne (wykonywane są równoległe). Można jednak zdefiniować pomiędzy nimi zależności a tym samym zdefiniować wymaganą kolejność realizacji.

## GitHub Action – podstawowe pojęcia (2)

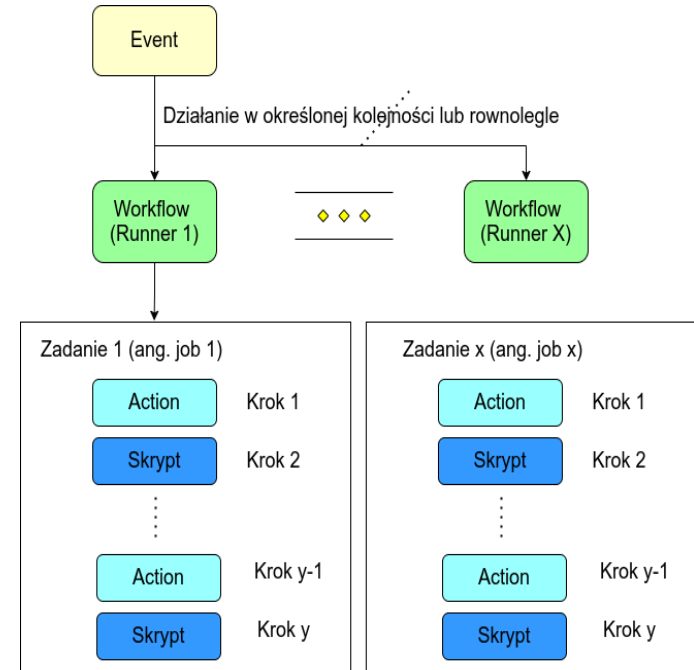
**Runner** – serwer, na którym wykonywane są zadania opisane w danym workflow (po pojawieniu się właściwego event-u). Każdy runner wykonuje jedno zadanie (job). Obecnie dostępne są: Ubuntu Linux, Microsoft Windows oraz macOS.

**Action** – aplikacja dedykowana dla platformy GitHub Actions, która wykonuje zestaw często powtarzanych zadań (np. zbudowanie i przesłanie obrazów do wskazanego repozytorium). Można opracować własne Action lub (zdecydowanie częstsze rozwiązanie), wykorzystać actions zgromadzone na GitHub Marketplace.

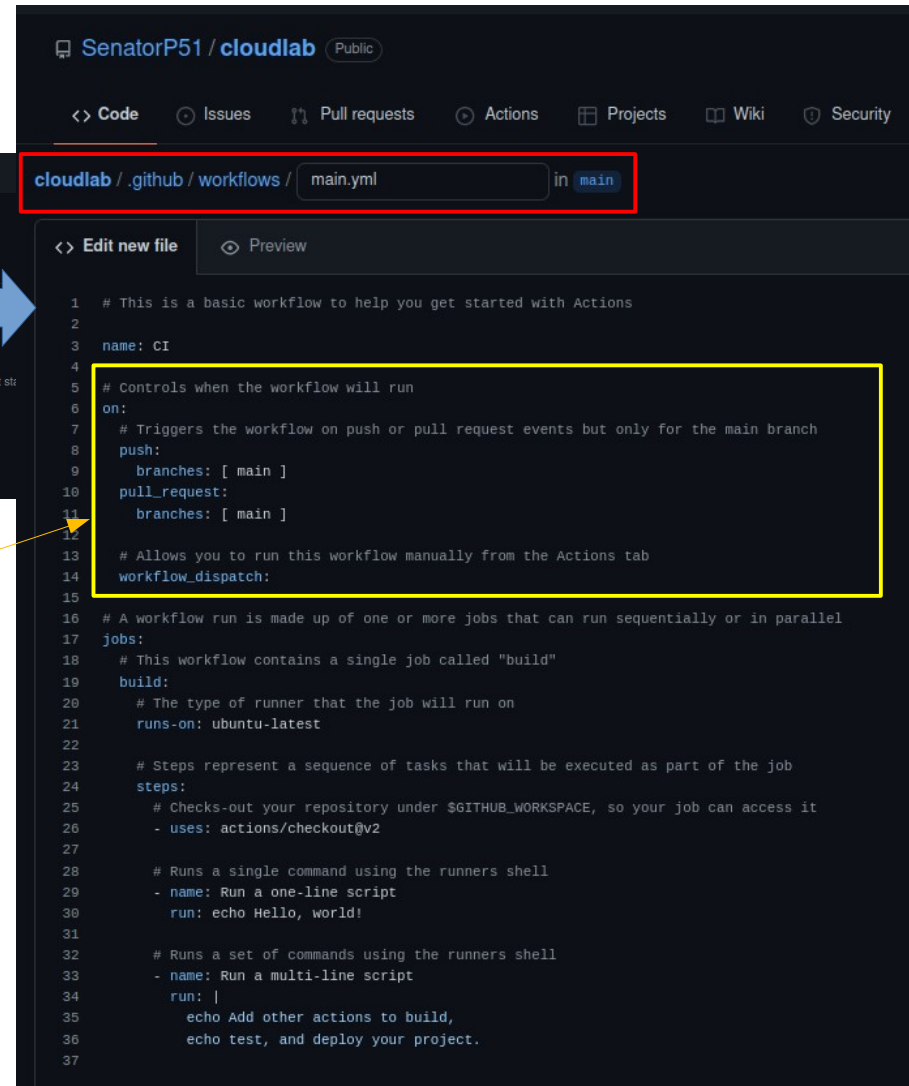
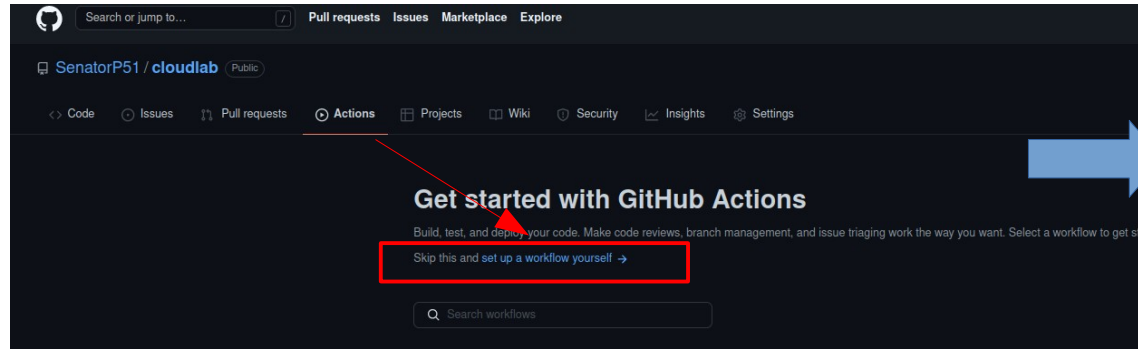
<https://github.com/marketplace>

**UWAGA:** Jeżeli niezbędne jest skorzystanie z innego runner-a niż dostępne domyślnie, można skonfigurować własne środowisko uruchomieniowe.

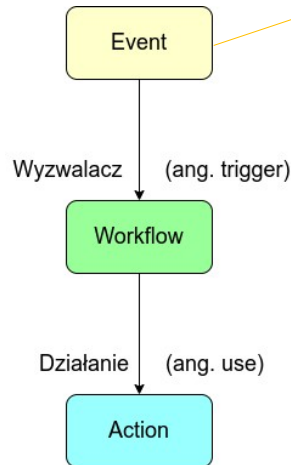
<https://docs.github.com/en/actions/hosting-your-own-runners>



# GitHub Actions – podstawy konfiguracji (1)



- GitHub triggered events:  
push, pull\_request,  
public
- Scheduled events: `schedule`
- Manually triggered:  
`workflow_dispatch`  
(external systems)



# GitHub Actions – podstawy konfiguracji (2)

## Inny przykład: matrix runner

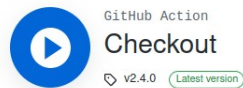
```
runs-on: ${ matrix.os }

strategy:
  matrix:
    node-version: [8.x, 10.x, 12.x]
    os: [macos-latest, windows-latest, ubuntu-18.04]

steps:
  - uses: actions/checkout@v1
  - name: Use Node.js ${ matrix.node-version }
    uses: actions/setup-node@v1
    with:
      node-version: ${ matrix.node-version }
  - name: npm install, build, and test
    run: |
      npm ci
      npm run build --if-present
      npm test
```

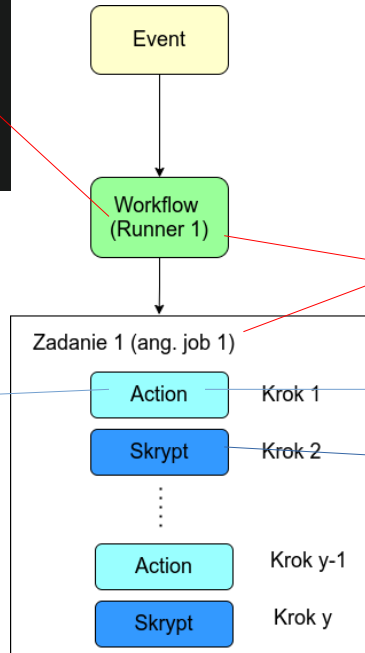
## Przykład z GitHub Marketplace

Marketplace / Actions / Checkout



Checkout V2

<https://github.com/marketplace/actions/checkout>



```
1 # This is a basic workflow to help you get started with Actions
2
3 name: CI
4
5 # Controls when the workflow will run
6 on:
7   # Triggers the workflow on push or pull request events but only for the main branch
8   push:
9     branches: [ main ]
10  pull_request:
11    branches: [ main ]
12
13 # Allows you to run this workflow manually from the Actions tab
14 workflow_dispatch:
15
16 # A workflow run is made up of one or more jobs that can run sequentially or in parallel
17 jobs:
18   # This workflow contains a single job called "build"
19   build:
20     # The type of runner that the job will run on
21     runs-on: ubuntu-latest
22
23     # Steps represent a sequence of tasks that will be executed as part of the job
24     steps:
25       # Checks-out your repository under $GITHUB_WORKSPACE, so your job can access it
26       - uses: actions/checkout@v2
27
28       # Runs a single command using the runners shell
29       - name: Run a one-line script
30         run: echo Hello, world!
31
32       # Runs a set of commands using the runners shell
33       - name: Run a multi-line script
34         run: |
35           echo Add other actions to build,
36           echo test, and deploy your project.
```

## GitHub Actions – przykład – przygotowanie (1)

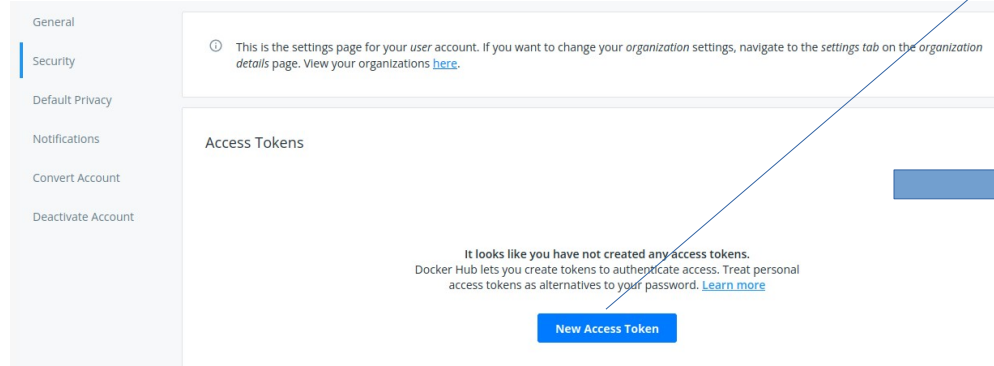
Przykład z omówieniem wykorzystywanych poleceń można znaleźć pod adresem:

<https://docs.github.com/en/actions/learn-github-actions/understanding-github-actions#understanding-the-workflow-file>

GitHub Actions wykorzystuje składnię YAML do definiowania danego workflow. Każdy workflow jest opisany w oddzielnym pliku YAML, który należy w repozytorium, w katalogu o nazwie **.github/workflows**.

# GitHub Actions – przykład – dostęp do DockerHUB

<https://hub.docker.com/settings/security>



## New Access Token

A personal access token is similar to a password except you can have many tokens and revoke access to each one at any time. [Learn more](#)

Access Token Description \*

Access permissions

Read, Write, Delete tokens allow you to manage your repositories.

Cancel

Generate

**UWAGA:**

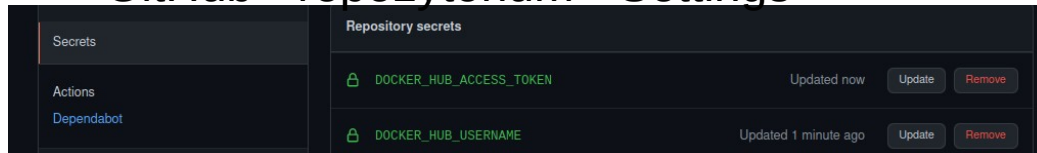


**WARNING:** This access token will only be displayed once. It will not be stored and cannot be retrieved. Please be sure to save it now.

Copy and Close

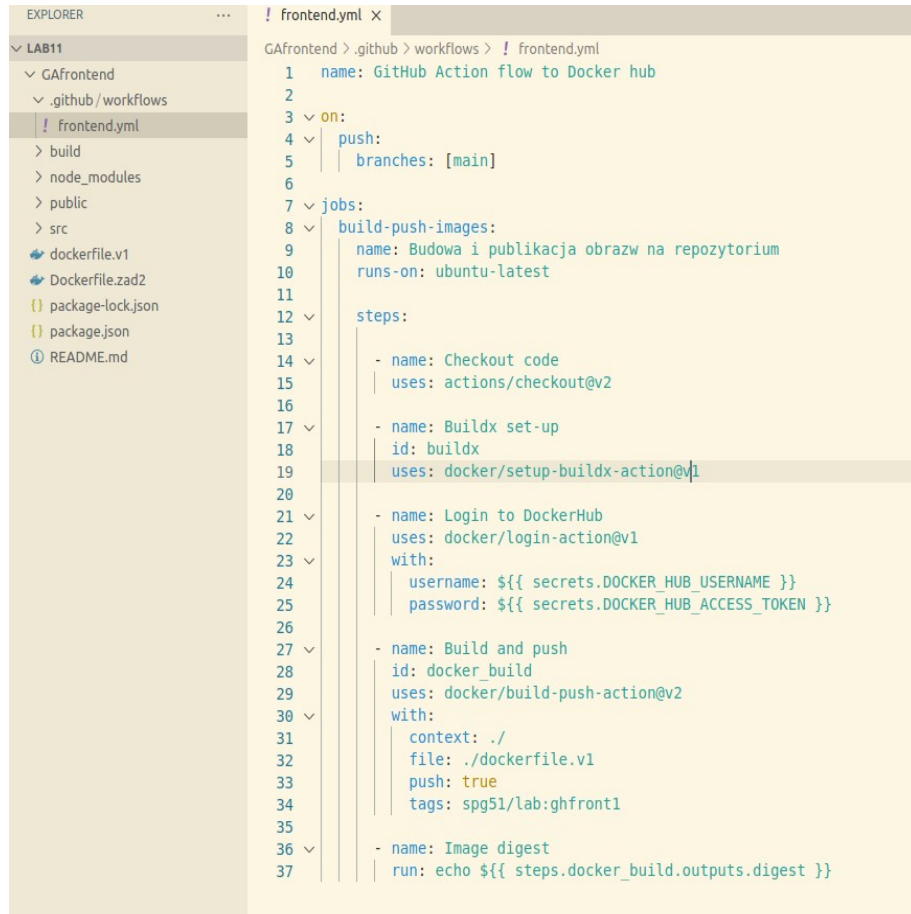
DOCKER\_HUB\_USERNAME  
DOCKER\_HUB\_ACCESS\_TOKEN

GitHub->repozytorium->Settings



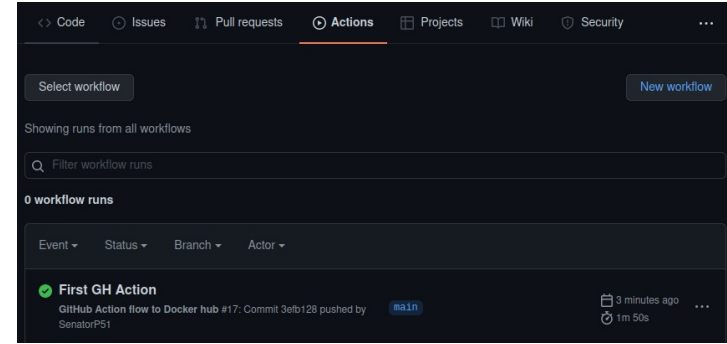


# GitHub Actions – przykład - działanie

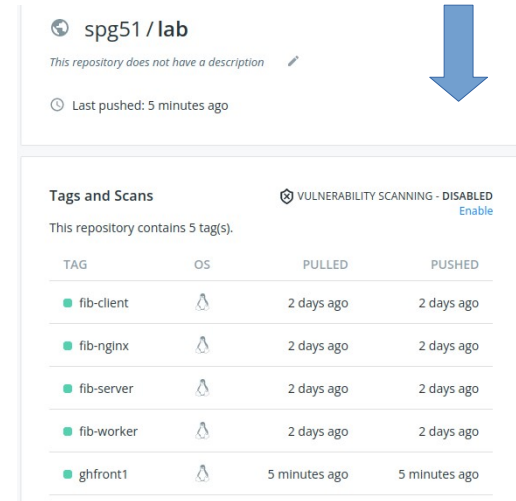


The image shows a VS Code Explorer on the left with a file tree for a project named LAB11. The file `frontend.yml` is selected. The main editor displays the content of `frontend.yml`, which is a GitHub Actions workflow. The workflow is triggered on a push to the `main` branch and consists of a single job named `build-push-images`. This job uses `ubuntu-latest` and contains several steps: checking out the code, setting up buildx, logging into Docker Hub, building and pushing Docker images, and finally echoing the image digest.

```
1 name: GitHub Action flow to Docker hub
2
3 on:
4   push:
5     branches: [main]
6
7 jobs:
8   build-push-images:
9     name: Budowa i publikacja obrazw na repozytorium
10    runs-on: ubuntu-latest
11
12    steps:
13
14      - name: Checkout code
15        uses: actions/checkout@v2
16
17      - name: Buildx set-up
18        id: buildx
19        uses: docker/setup-buildx-action@v1
20
21      - name: Login to DockerHub
22        uses: docker/login-action@v1
23        with:
24          username: ${ secrets.DOCKER_HUB_USERNAME }
25          password: ${ secrets.DOCKER_HUB_ACCESS_TOKEN }
26
27      - name: Build and push
28        id: docker_build
29        uses: docker/build-push-action@v2
30        with:
31          context: ./
32          file: ./dockerfile.v1
33          push: true
34          tags: spg51/lab:ghfront1
35
36      - name: Image digest
37        run: echo ${ steps.docker_build.outputs.digest }
```



The image shows the GitHub Actions interface. At the top, there are tabs for Code, Issues, Pull requests, Actions, Projects, Wiki, and Security. The 'Actions' tab is selected. Below the tabs, there is a 'Select workflow' button and a 'New workflow' button. A search bar for 'Filter workflow runs' is present. Below this, it says '0 workflow runs'. A dropdown menu shows 'Event', 'Status', 'Branch', and 'Actor'. A workflow run is listed with a green checkmark, labeled 'First GH Action', and a status of 'Completed'. It was triggered by a push to the `main` branch by user 'SenatorP51' 3 minutes ago, with a duration of 1m 50s.



The image shows the GitHub repository page for `spg51/lab`. The repository does not have a description. The last push was 5 minutes ago. Below this, there is a section for 'Tags and Scans' with a 'VULNERABILITY SCANNING - DISABLED' warning. The repository contains 5 tags. A table lists the tags, their OS, and the time they were pulled and pushed.

TAG	OS	PULLED	PUSHED
fib-client	linux	2 days ago	2 days ago
fib-nginx	linux	2 days ago	2 days ago
fib-server	linux	2 days ago	2 days ago
fib-worker	linux	2 days ago	2 days ago
ghfront1	linux	5 minutes ago	5 minutes ago

Kody źródłowe na moodle: **lab\_gh.zip**