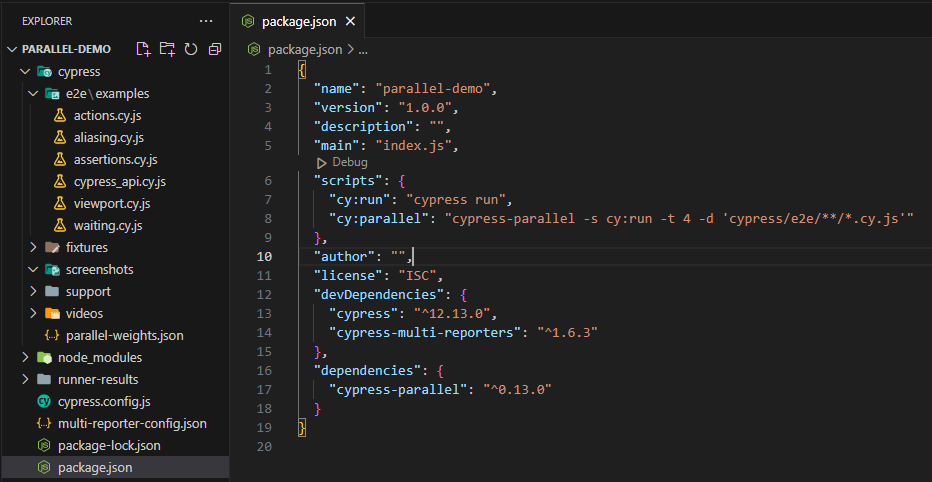
# Run Cypress Test parallel without Cypress Cloud

Cypress is an excellent tool for end-to-end testing, and parallel testing is a feature that Cypress supports in order to save execution time. When we want to run cypress tests in parallel, we need a Cypress Cloud account, which is not free; you can view the price of Cypress Cloud [here](https://www.cypress.io/pricing/#monthly).

If we don't have a Cypress Cloud account, the rest of this post will show you how to run Cypress tests in parallel by utilizing the matrix feature from GitHub Actions.

## Install Cypress and create some example tests

npm i –D cypress

The Cypress version at the time of writing is 12.13.0. I kept six test specs from the default Cypress examples for this demonstration. 

## Run parallel test in single machine:

From Cypress: *“While parallel tests can also technically run on a single machine, we do not recommend it since this machine would require significant resources to run your tests efficiently.”*

Continue reading to see why executing Cypress parallel tests on a single machine is not recommended.

Install package ‘Cypress-parallel’ <https://github.com/tnicola/cypress-parallel>

npm i cypress-parallel -D

Install ‘cypress-multi-reporters’ <https://www.npmjs.com/package/cypress-multi-reporters>

npm install cypress-multi-reporters --save-dev

Add two scripts below to package.json

 "scripts": {

    "cy:run": "cypress run",

    "cy:parallel": "cypress-parallel -s cy:run -t 2 -d 'cypress/e2e/\*\*/\*.cy.js'"

  }

-s: Script to run

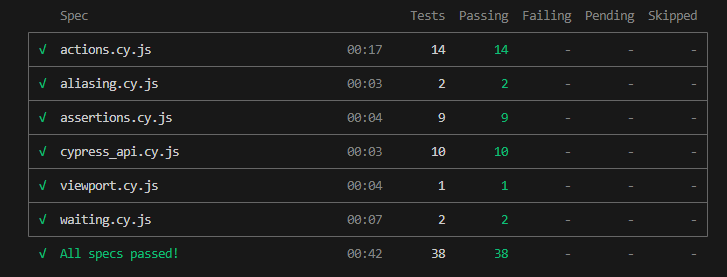
-t: number of threads

-d: path of test specs

After we have completed the setup, we will attempt to run tests sequentially and parallel using cypress-parallel and evaluate the timing and resources used.

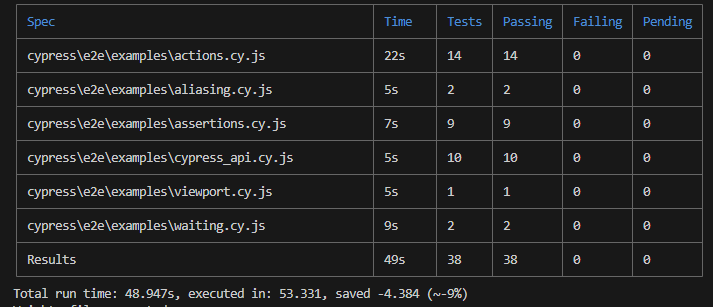
First, try running tests sequentially:

npm run cy:run



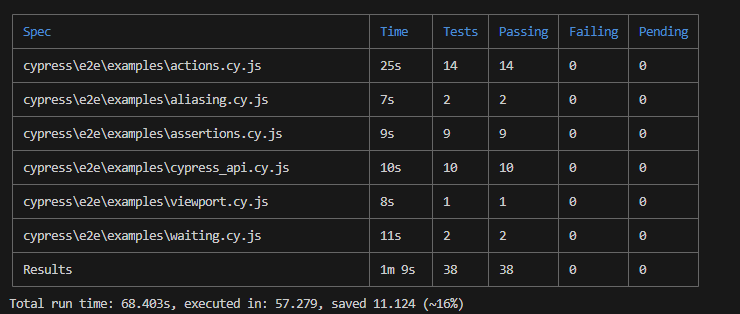
As seen in the picture above, it takes 42 seconds to complete 38 tests, let us now use cypress-parallel in two threads.

npm run cy:parallel



The test with two threads took around 54 seconds to complete. Now we repeat the test with four threads.

npx cypress-parallel -s cy:run -t 4 -d 'cypress/e2e/\*\*/\*.cy.js'



This time, the execution duration was 57 seconds, indicating that running cypress-parallel in a single machine did not assist to lower the execution time while running sequentially. Parallel execution took 54 seconds in two threads and 57 seconds in four threads, whereas sequential execution took just 42 seconds. Furthermore, during simultaneous testing, CPU utilization got up from 2% to 80-100%.

|  |  |  |
| --- | --- | --- |
| Running 6 test in single machine | Execution time (seconds) | CPU utilization |
| sequentially | 42 | Increase from 2% to 50% |
| Cypress-parallel (2 threads) | 53.331 | 80% |
| Cypress-parallel (4 threads) | 57.279 | 90-100% |

## Run parallel tests by utilizing GitHub Action:

On GitHub Actions, we will take advantage of the matrix features. Create a workflow file (.yml) with two jobs install and tests in it: .github\workflows\parallel-test.yml

name: cypress parallel

on:

  push:

    branches: master

jobs:

  install:

    runs-on: ubuntu-latest

  tests:

    runs-on: ubuntu-latest

    needs: install

You can adjust the event [push] and branch [master] to trigger the workflow on the on section.

Add the necessary steps into install and tests jobs.

name: cypress parallel

on:

  push:

    branches: master

jobs:

  install:

    runs-on: ubuntu-latest

    steps:

      - name: Checkout

        uses: actions/checkout@v3

      - name: Install Cypress

        uses: cypress-io/github-action@v5

        with:

          runTests: false

  tests:

    runs-on: ubuntu-latest

    needs: install

    steps:

      - name: Checkout

        uses: actions/checkout@v3

The actions/checkout@v3 to checkout code from repository, cypress-io/github-action@v5 sets up Cypress, installs dependencies and caches them between runs. In the install step it uses runTests: false since we only want to set up the Cypress, not run tests just yet.

We now leverage Github Action's matrix feature to run tests in parallel; instead of using the flag --parallel from cypress-io/github-action@v5 as we usually do with Cypress Cloud, we specify the spec using the matrix feature. The spec files will be placed in matrix variable containers.

In the first run, I will only put two spec files actions and aliasing to matrix containers.

name: cypress parallel

on:

  push:

    branches: master

jobs:

  install:

    runs-on: ubuntu-latest

    steps:

      - name: Checkout

        uses: actions/checkout@v3

      - name: Install Cypress

        uses: cypress-io/github-action@v5

        with:

          runTests: false

  tests:

    runs-on: ubuntu-latest

    needs: install

    strategy:

      fail-fast: false

      matrix:

        containers: [actions, aliasing]

    steps:

      - name: Checkout

        uses: actions/checkout@v3

      - name: Run Test Parallel

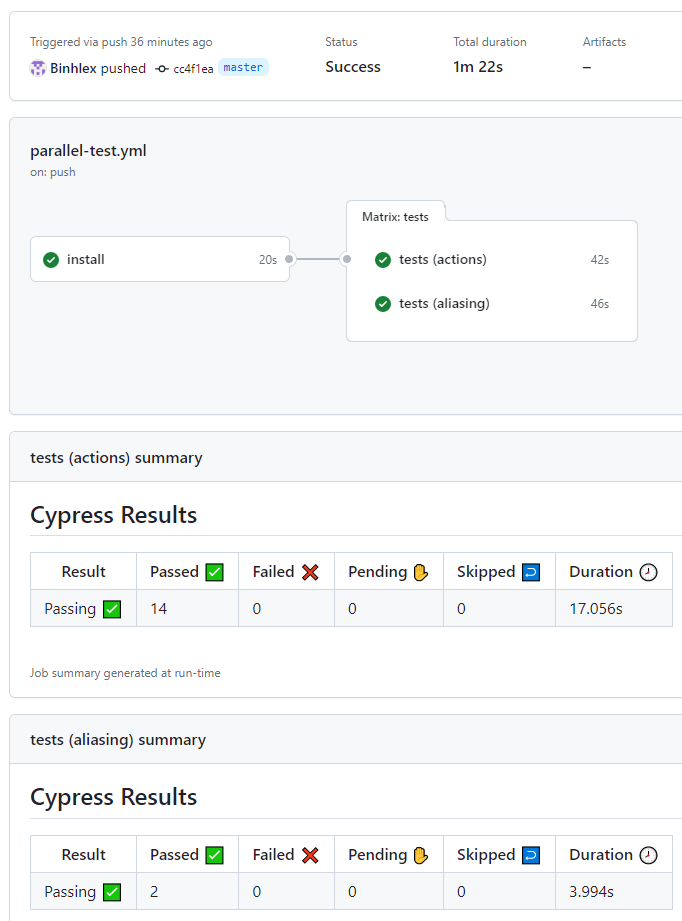
        uses: cypress-io/github-action@v5

        with:

          spec: cypress/e2e/examples/${{matrix.containers}}.cy.js

          browser: chrome

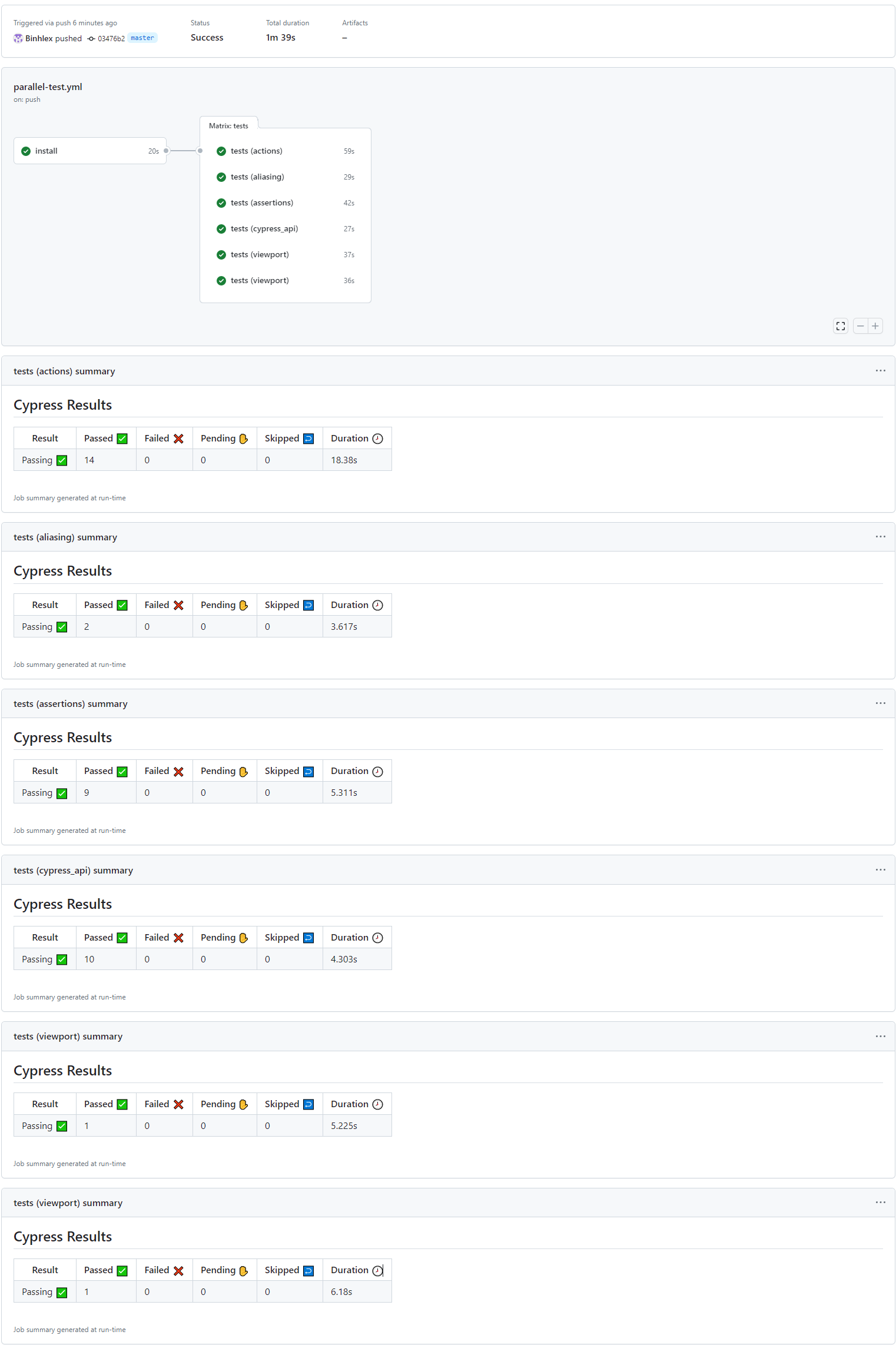
Commit and Push the workflow to a repository on Github, the Github Actions will trigger the jobs, two instances of Ubuntu will be created and run a test in each instance.



Now we added all 6 tests to the workflow and push them to GitHub.

matrix:

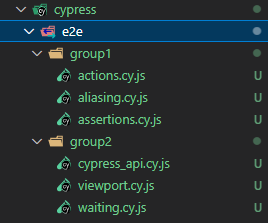
     containers: [actions, aliasing, assertions, cypress\_api, viewport, viewport]



The total execution time of 6 tests is around 43 seconds, which is comparable to the time spent when executing tests sequentially. The longest test took 18.38 seconds, which means it only took 18 seconds to execute all 6 tests in parallel.

For more information, Github allows running Github Actions for 2000 minutes and 20 concurrent jobs monthly for a free account. A job matrix can generate a maximum of 256 jobs per workflow run. This limit applies to both GitHub-hosted and self-hosted runners.

In the real world, we have a large number of features and test spec files in our test project. How can we organize and run them together? I'll utilize the same six tests as previously, dividing them into two folders labeled 'group1' and 'group2'.



And edit the workflow:

name: cypress parallel

on:

  push:

    branches: master

jobs:

  install:

    runs-on: ubuntu-latest

    steps:

      - name: Checkout

        uses: actions/checkout@v3

      - name: Install Cypress

        uses: cypress-io/github-action@v5

        with:

          runTests: false

  tests:

    runs-on: ubuntu-latest

    needs: install

    strategy:

      fail-fast: false

      matrix:

        groups: [group1, group2]

    steps:

      - name: Checkout

        uses: actions/checkout@v3

      - name: Run Test Parallel

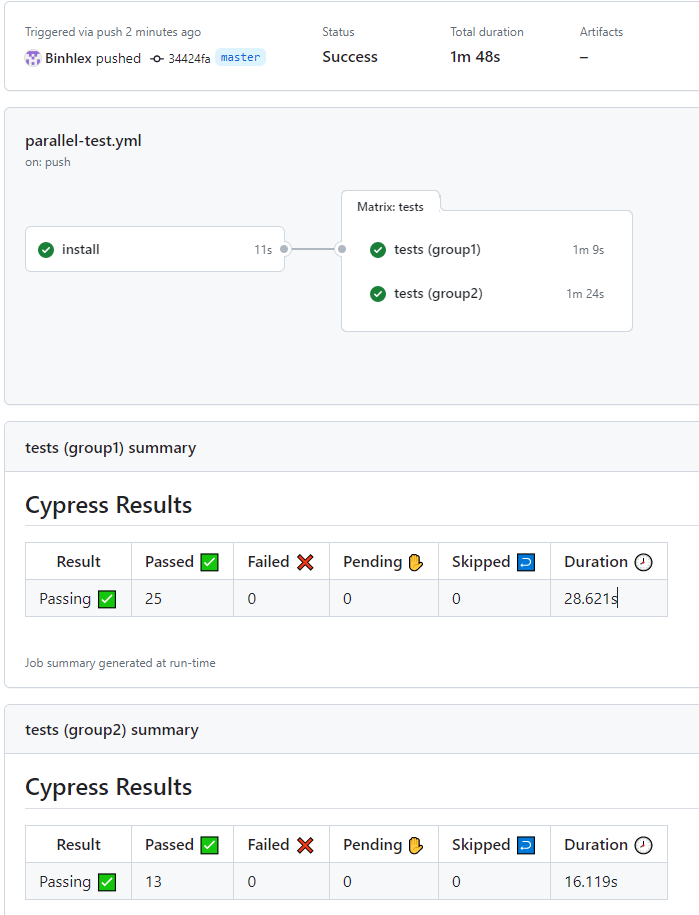
        uses: cypress-io/github-action@v5

        with:

          spec: cypress/e2e/${{matrix.groups}}/\*.cy.js

          browser: chrome

We defined a variable groups in the stratety.matrix and pass the test folders as an array. At the spec section, we passed the variable matrix.groups to specify the test spec folders. When pushing this workflow to Github, it will create two Ubuntu instances, each instance will run all tests in each folder.



# Conclusion:

We may run Cypress tests in parallel without using Cypress Cloud by leveraging the matrix feature on Github Actions, saving money for your project. The time spent in each instance depends on the number of test spec files placed in the folders (groups). Therefore, you should organize the number of test specifications among directories such that their execution durations are equal.