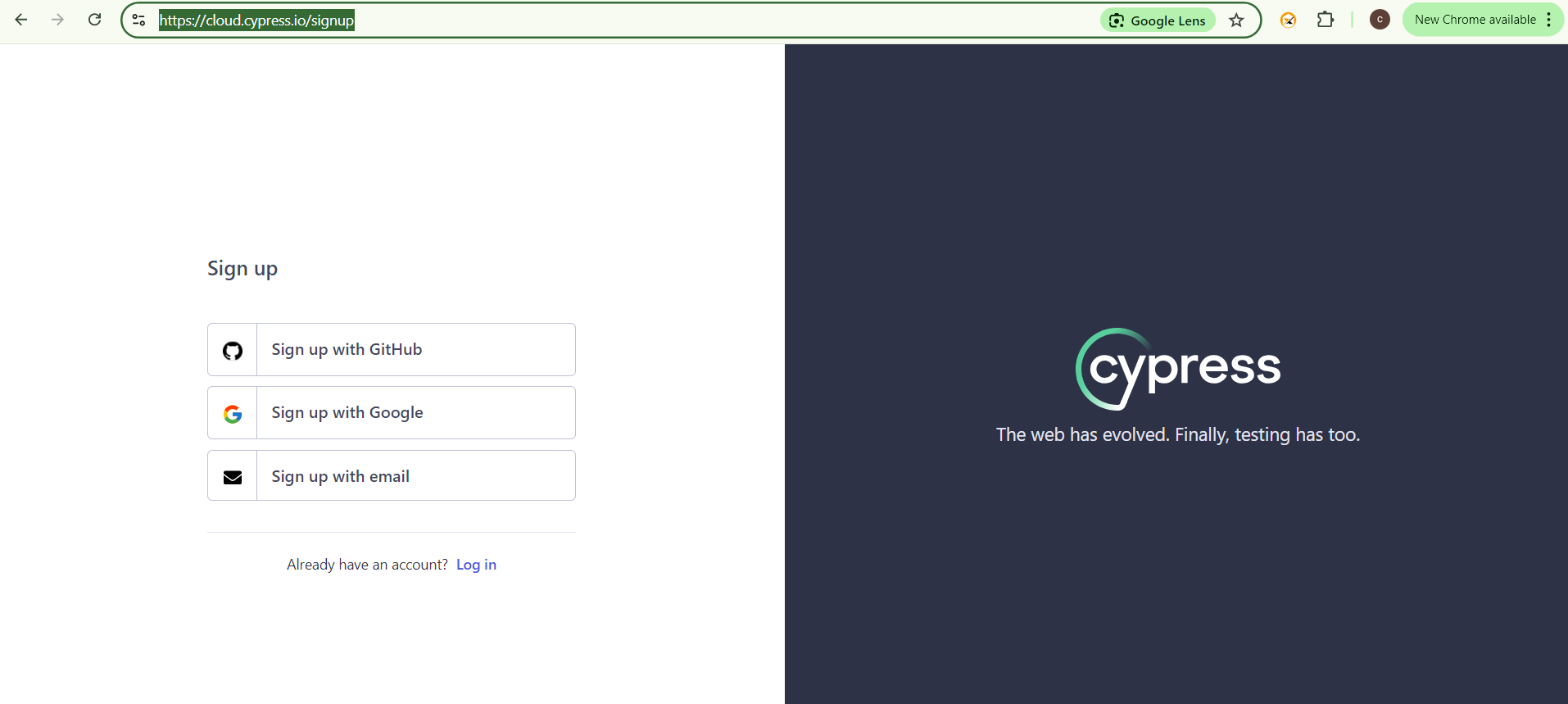
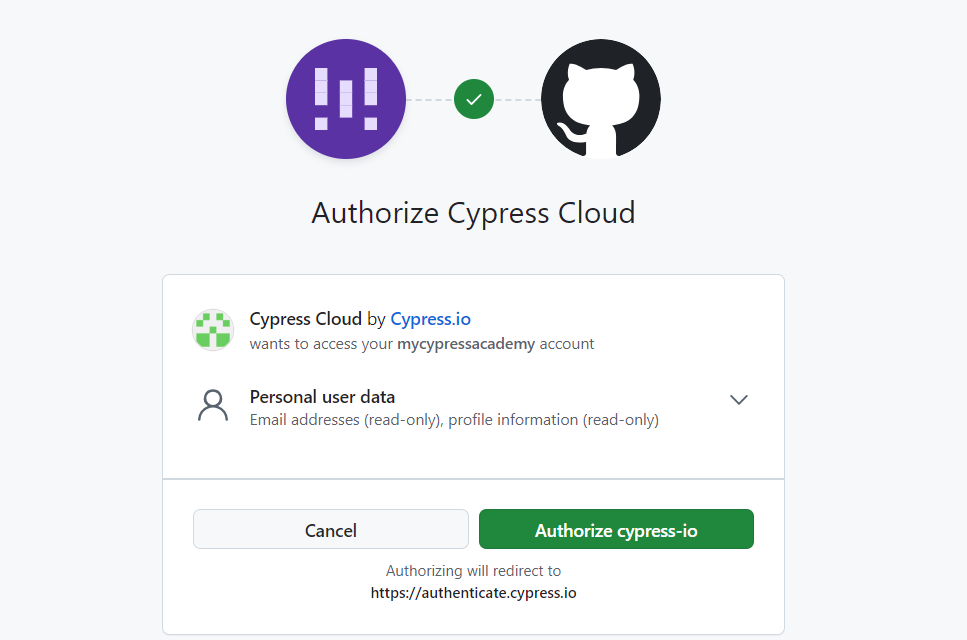
Create a cypress cloud account with github account:

1. Browse <https://cloud.cypress.io/signup>

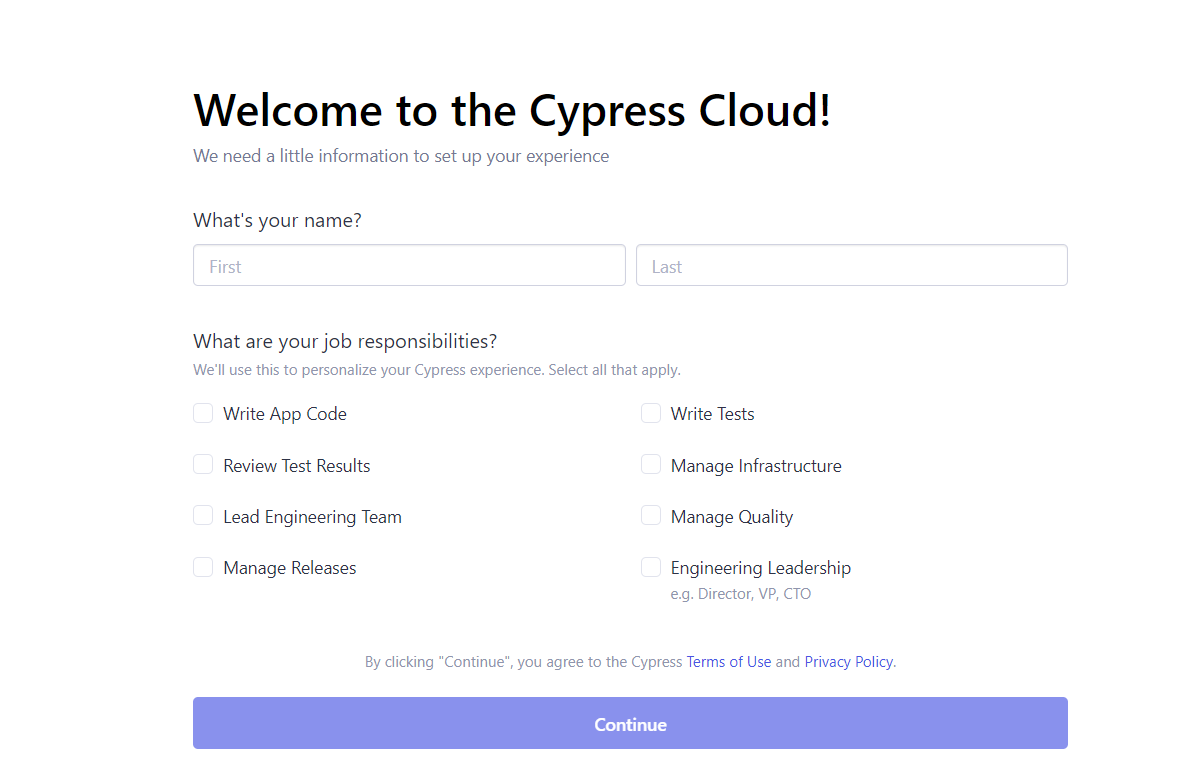


1. Select Signup with Github option

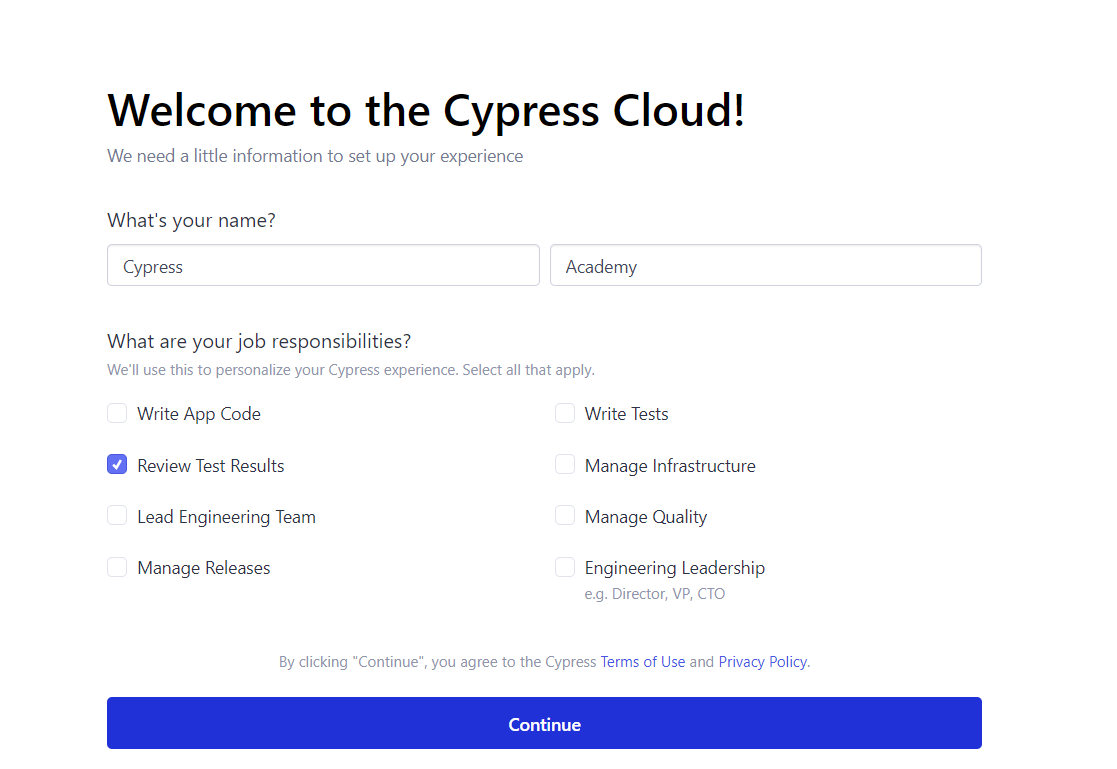


If you already Logged to Github account it will request the Authorization from your side . Click on Authorize Cypress-io button

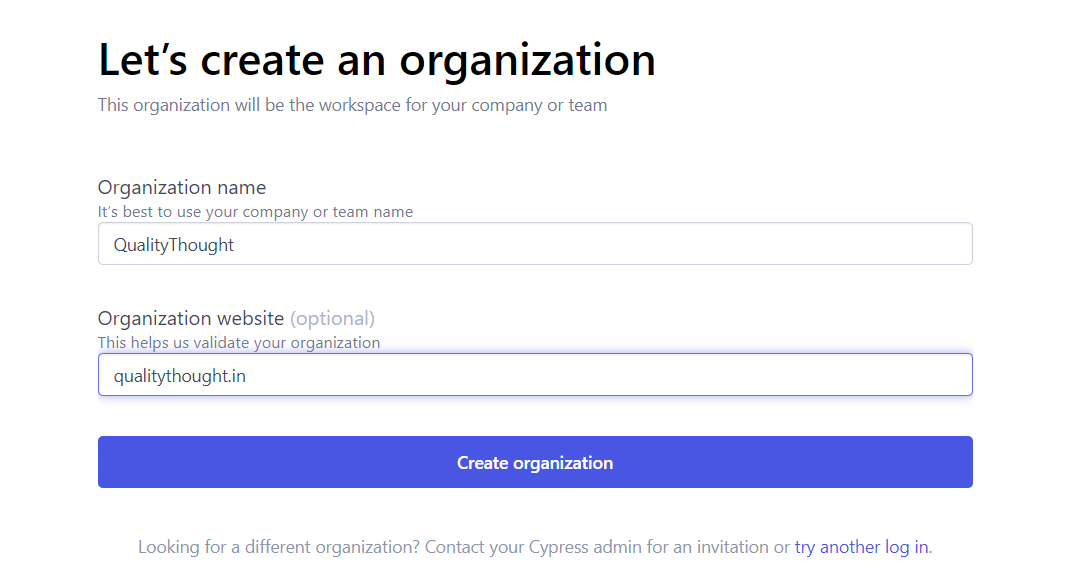
1. Welcome to cypress cloud screen should be displayed



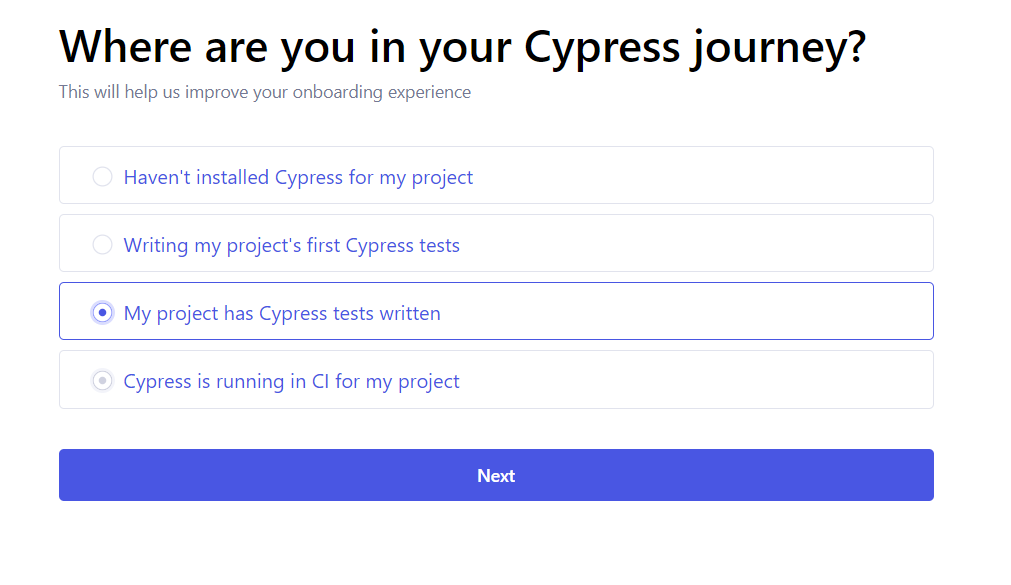
1. Fill up the form and click on continue



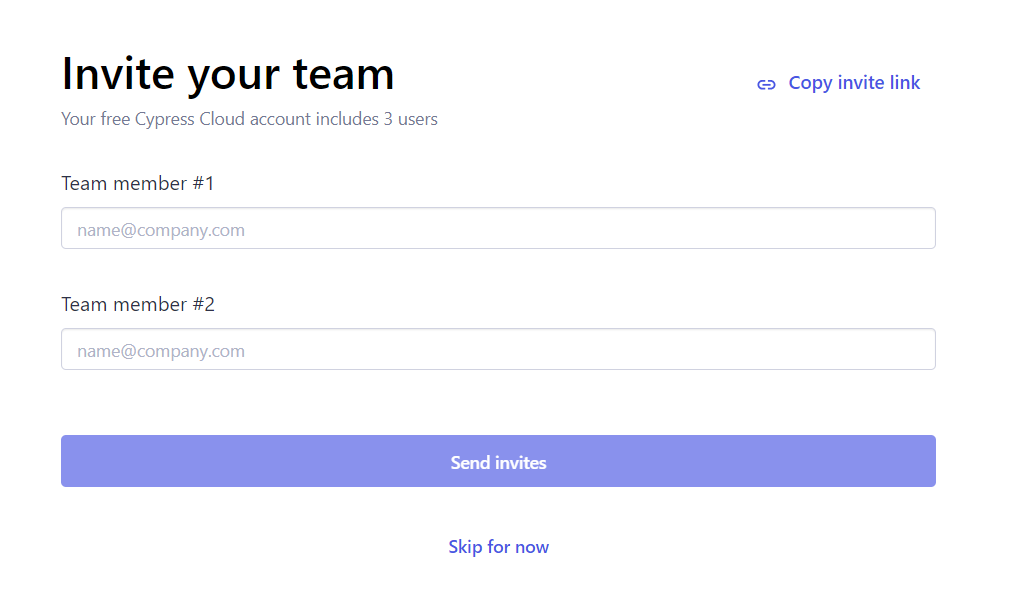
1. Fill the organization details as follows and click on Create organization button.



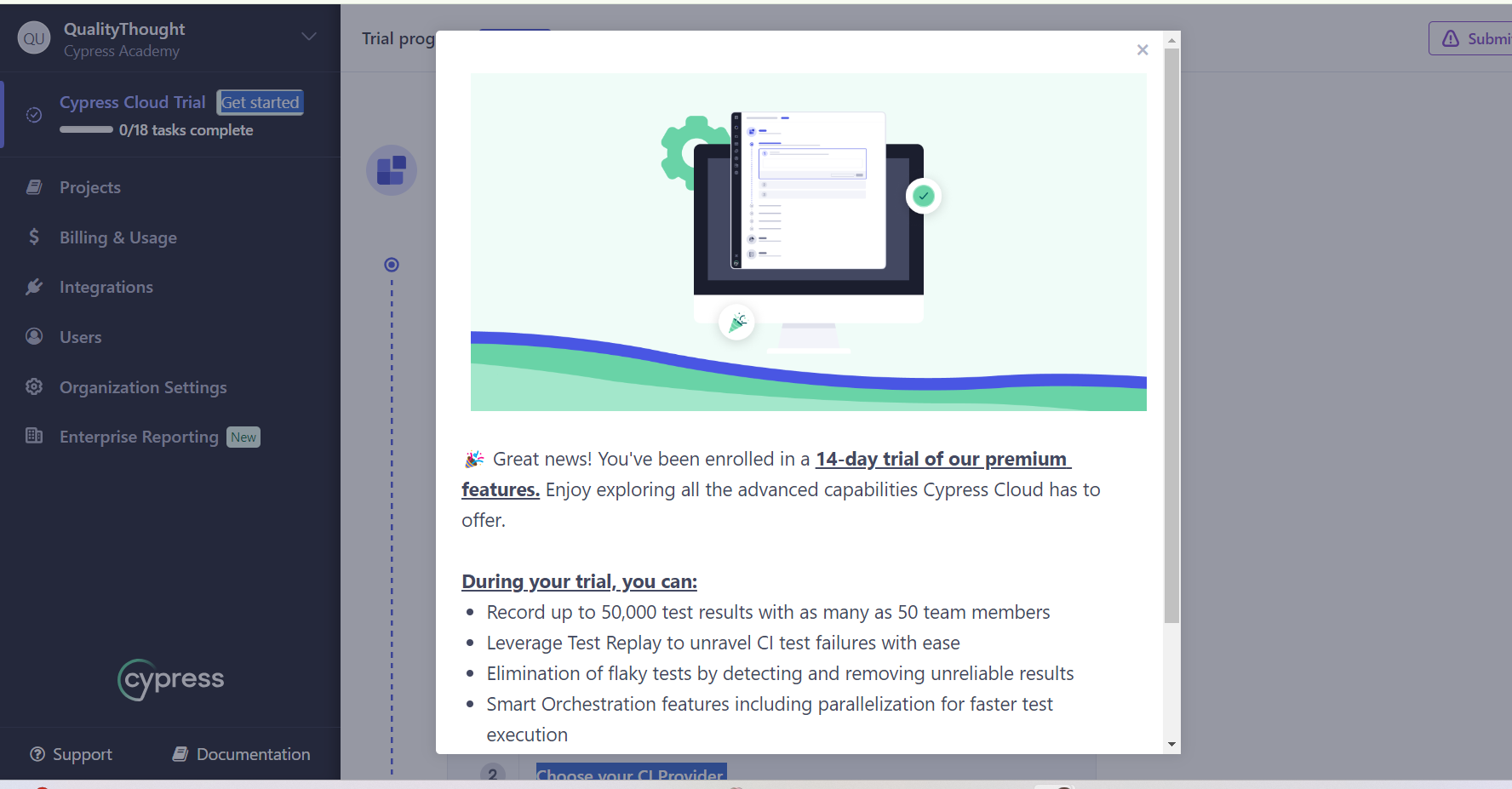
1. Input where are you in cypress journey to the Cypress cloud



1. Invite the team or share the link to join your team or Skip

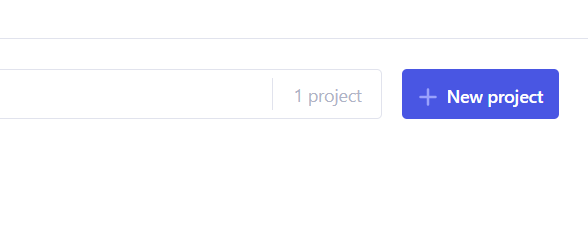


1. On successful Registration you can see the below screen

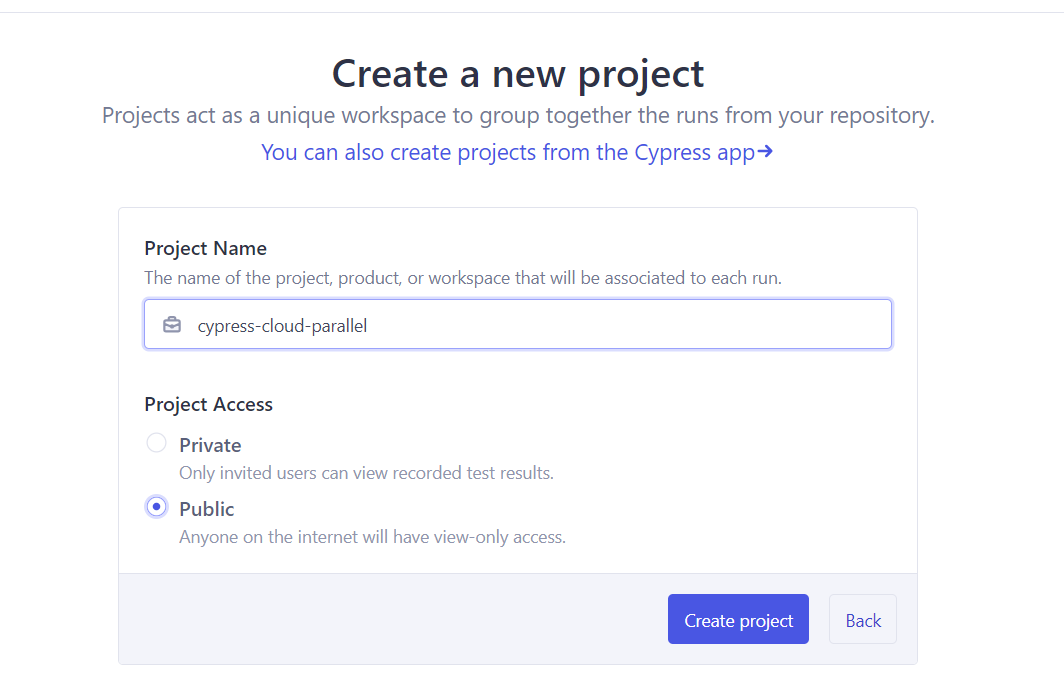


All set , you can play with cypress cloud.

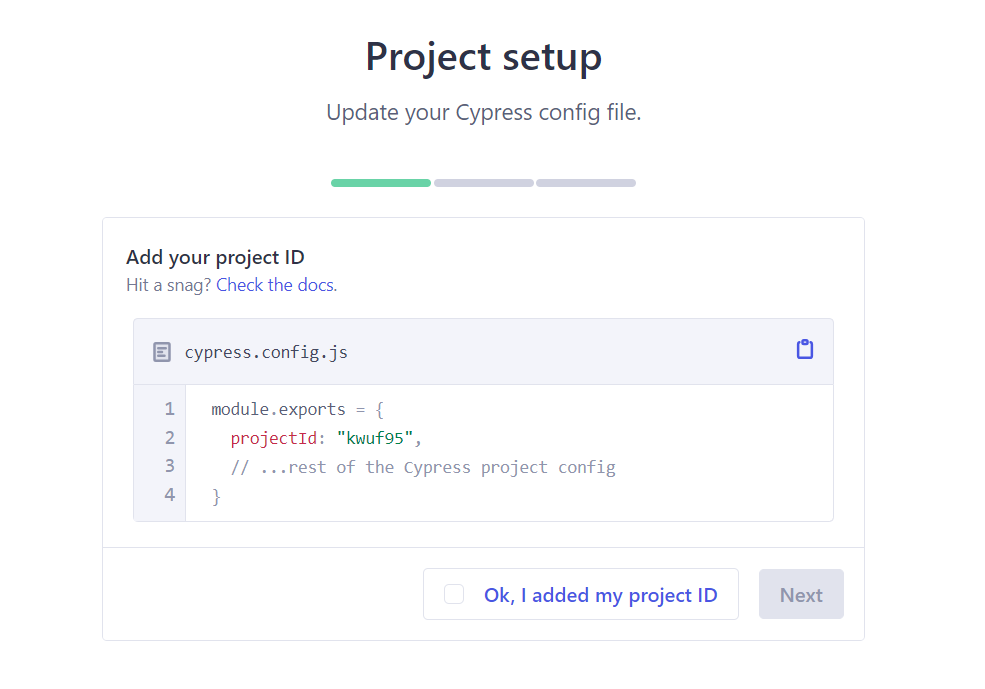
1. In projects section create a Project by using **New project** button

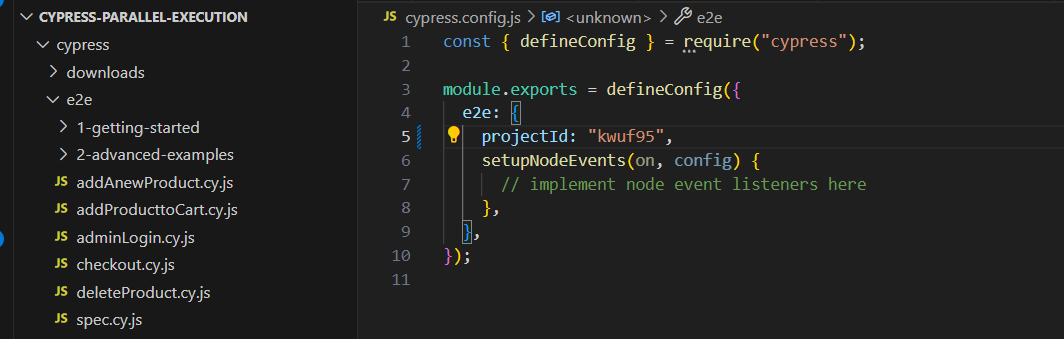


1. Supply the details to create Project in cypress cloud



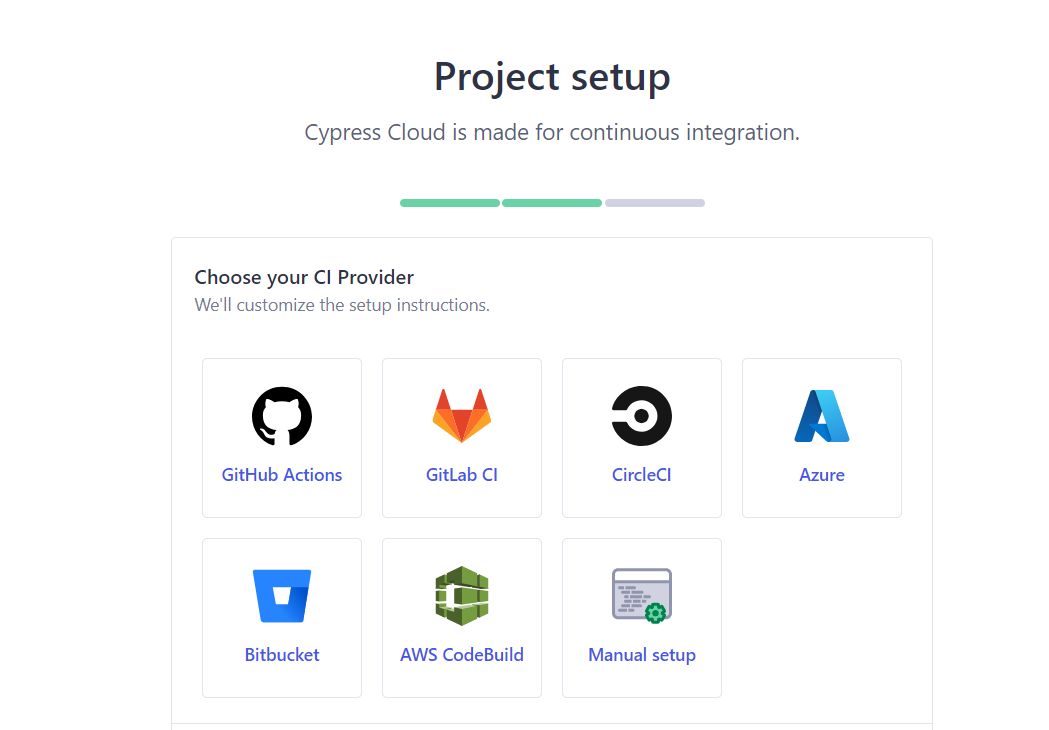
1. Capture the Project Id and place it in cypress.config.js



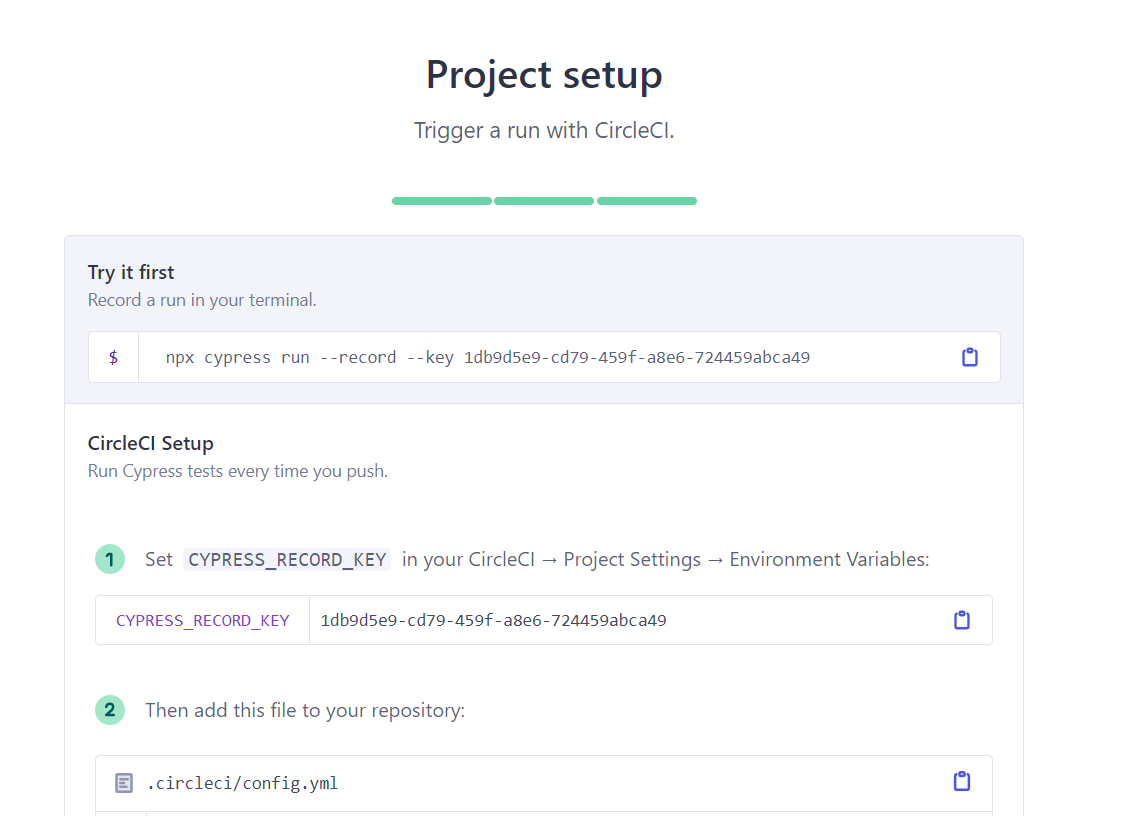


Save the cypress.config.js file

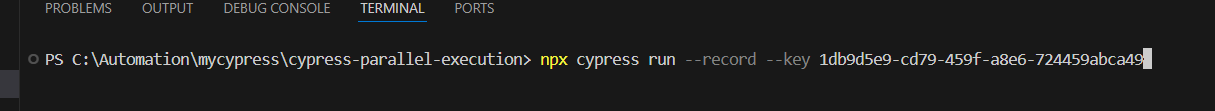
1. In project setup screen, select suitable CI provider



1. Steps to trigger CI – CircleCI

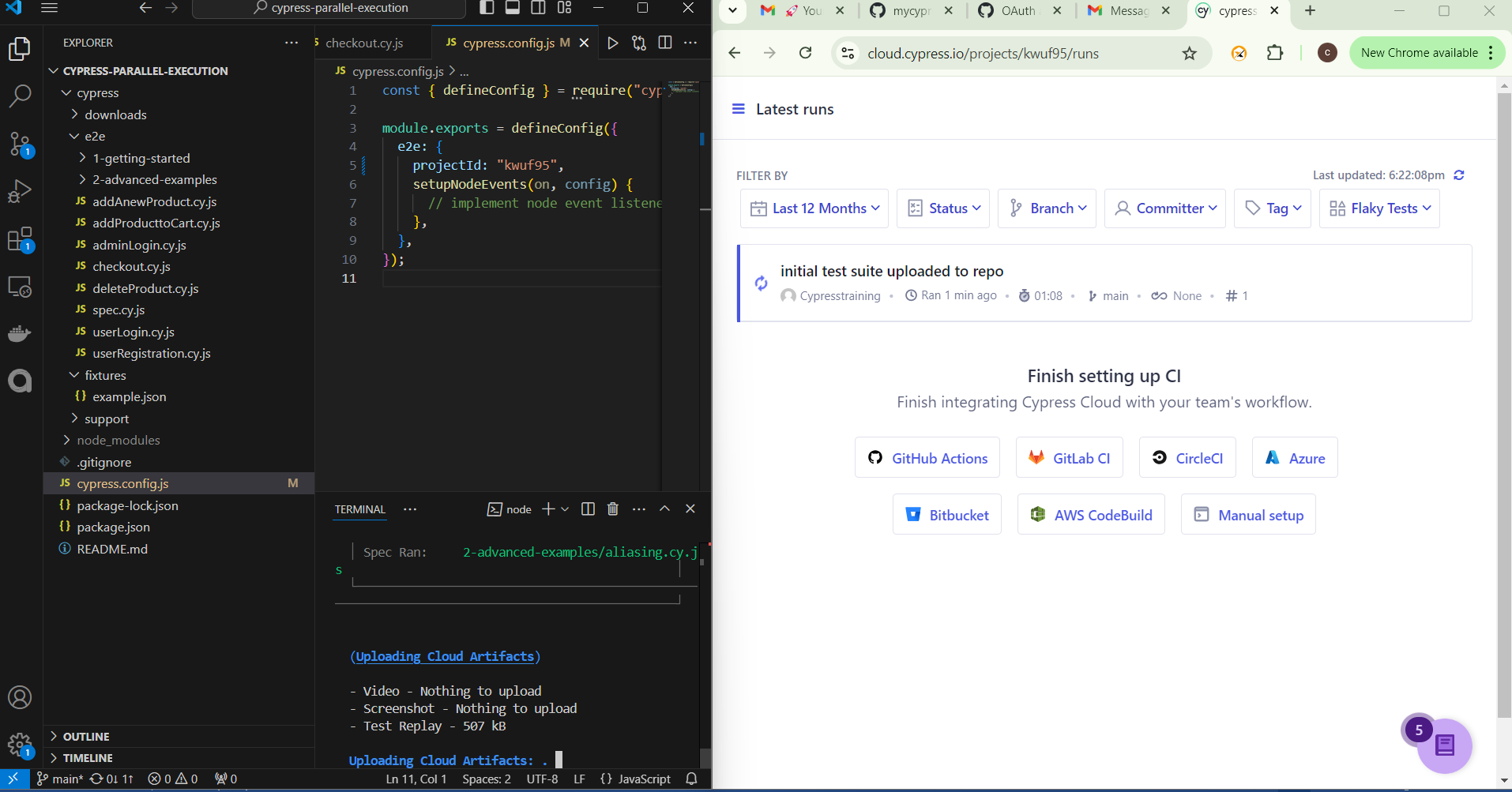


1. Copy the npx statement and run in the VSCode



1. Based on test run results will uploaded to cypress cloud

Latest run section in the cypress cloud



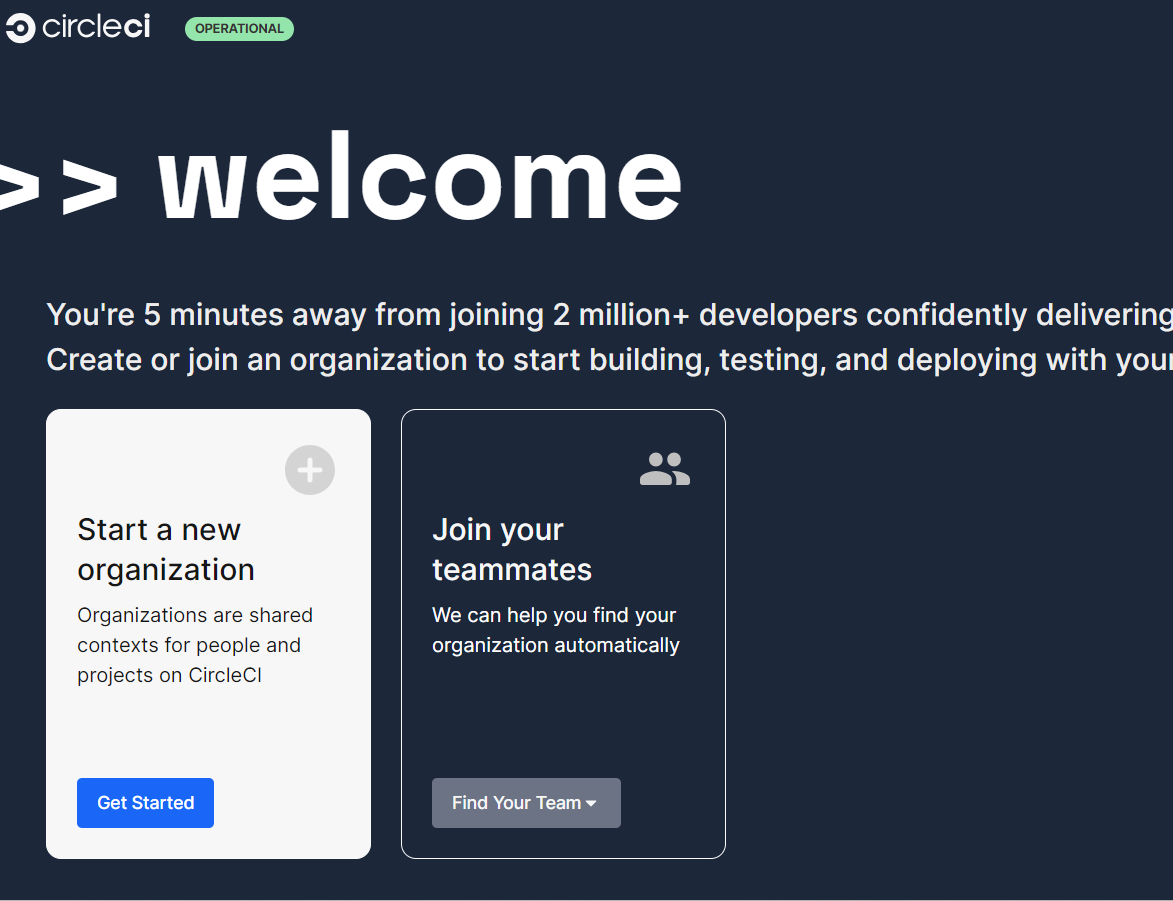
1. Setup circle as per below guide lines



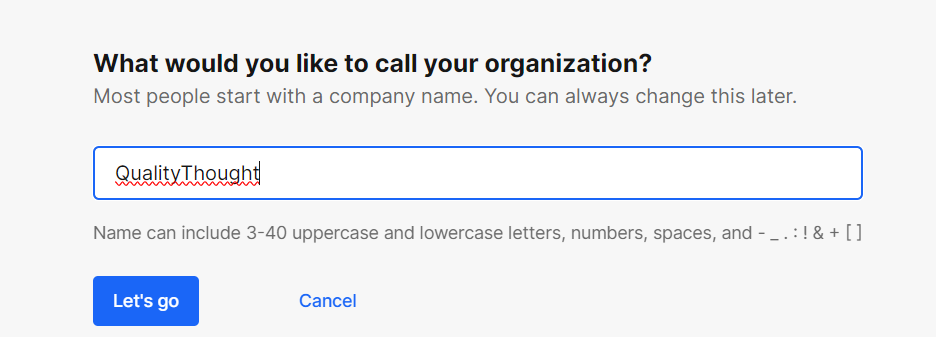
1. Create a circle AI account – to integrate with cypress cloud

Signup to the circle CI - <https://circleci.com/signup/>

Once you are done with signup and Login actions you can able to see the Welcome page as follows



1. Lets create a new Organization to create a project

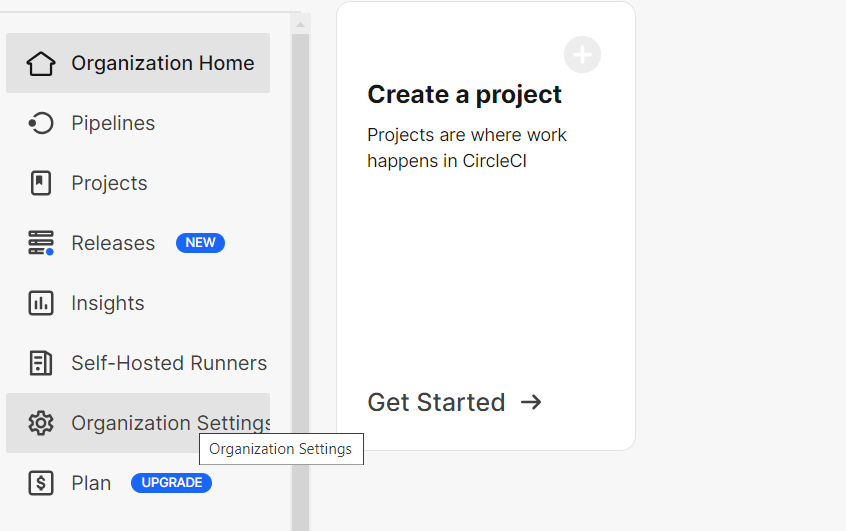


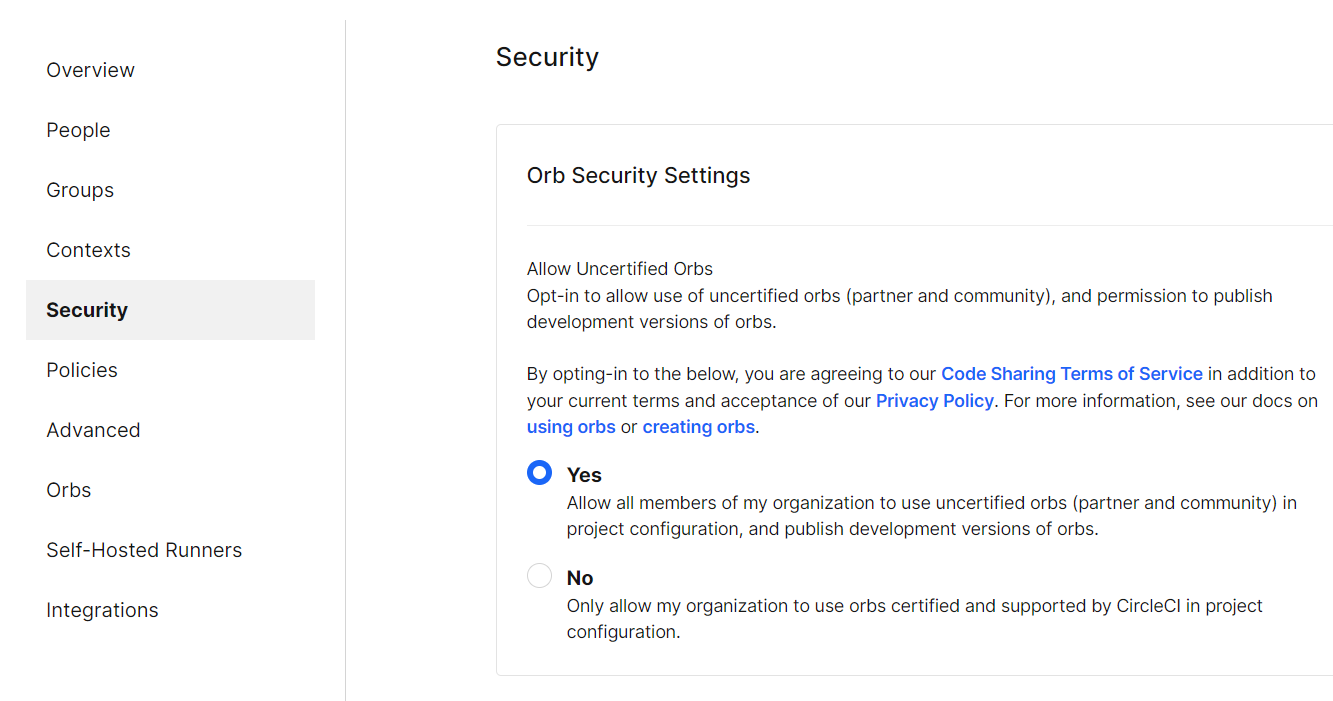
1. In the organization lets select the security settings

“Allow all members of my organization to use uncertified orbs (partner and community) in project configuration, and publish development versions of orbs.”

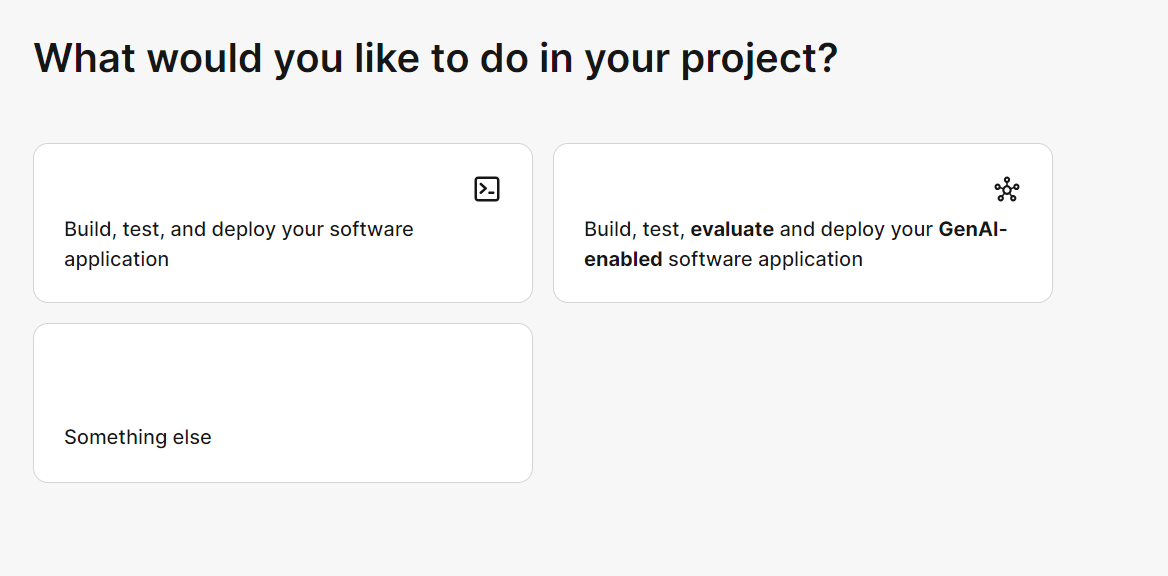
This settings is required to use cypress orb to run parallel execution

1. Select the Organization settings section in the below screen



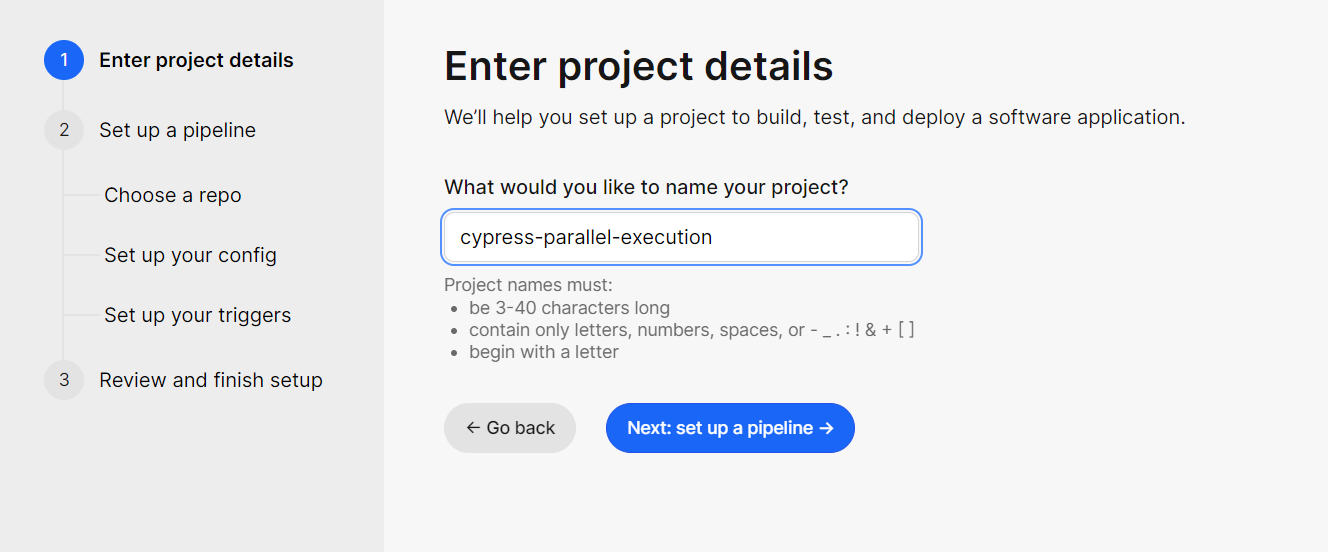


1. Return to Organization home and create a project by following below screens

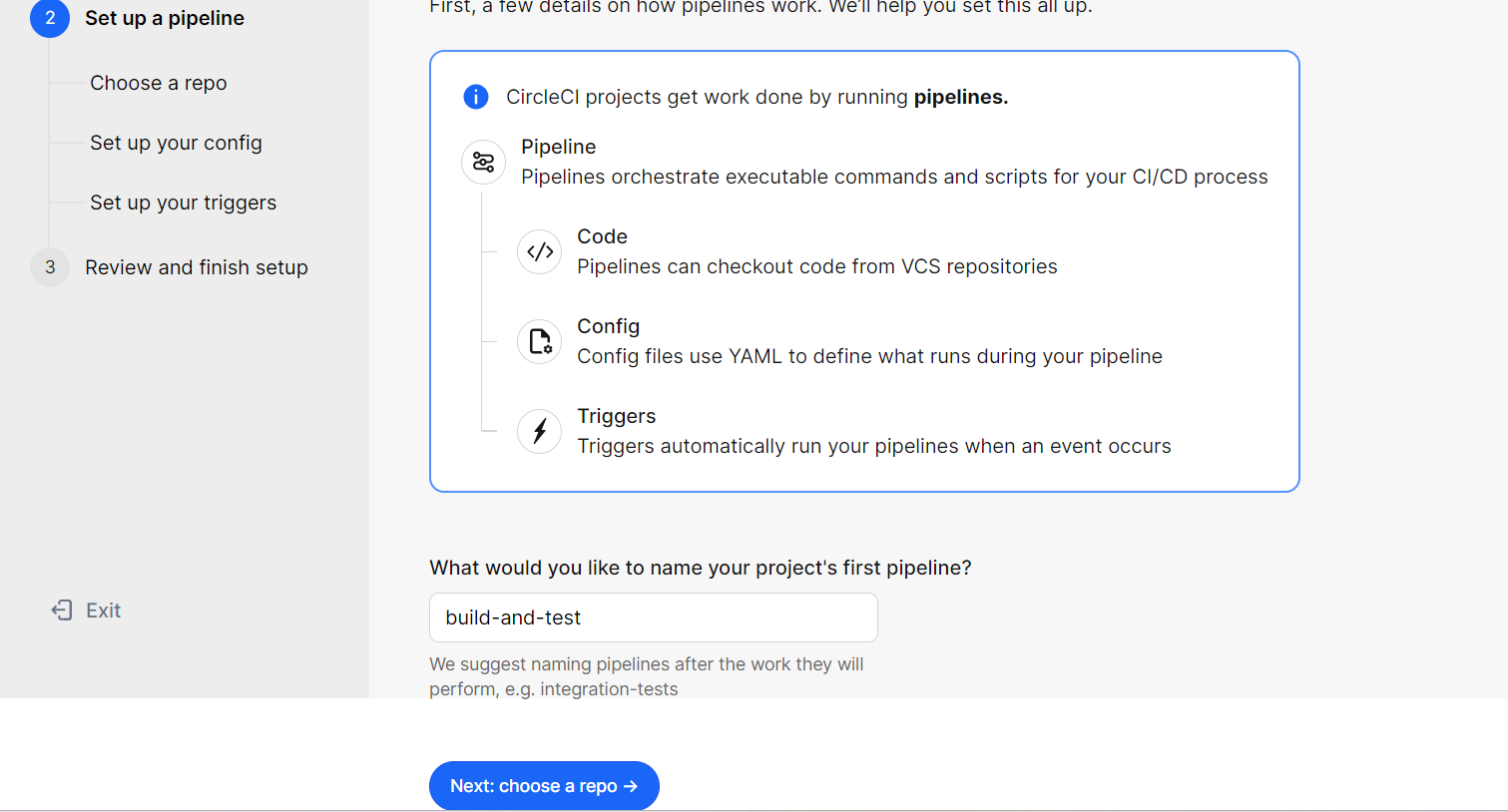


Select build, test and deploy your software application in above screen .

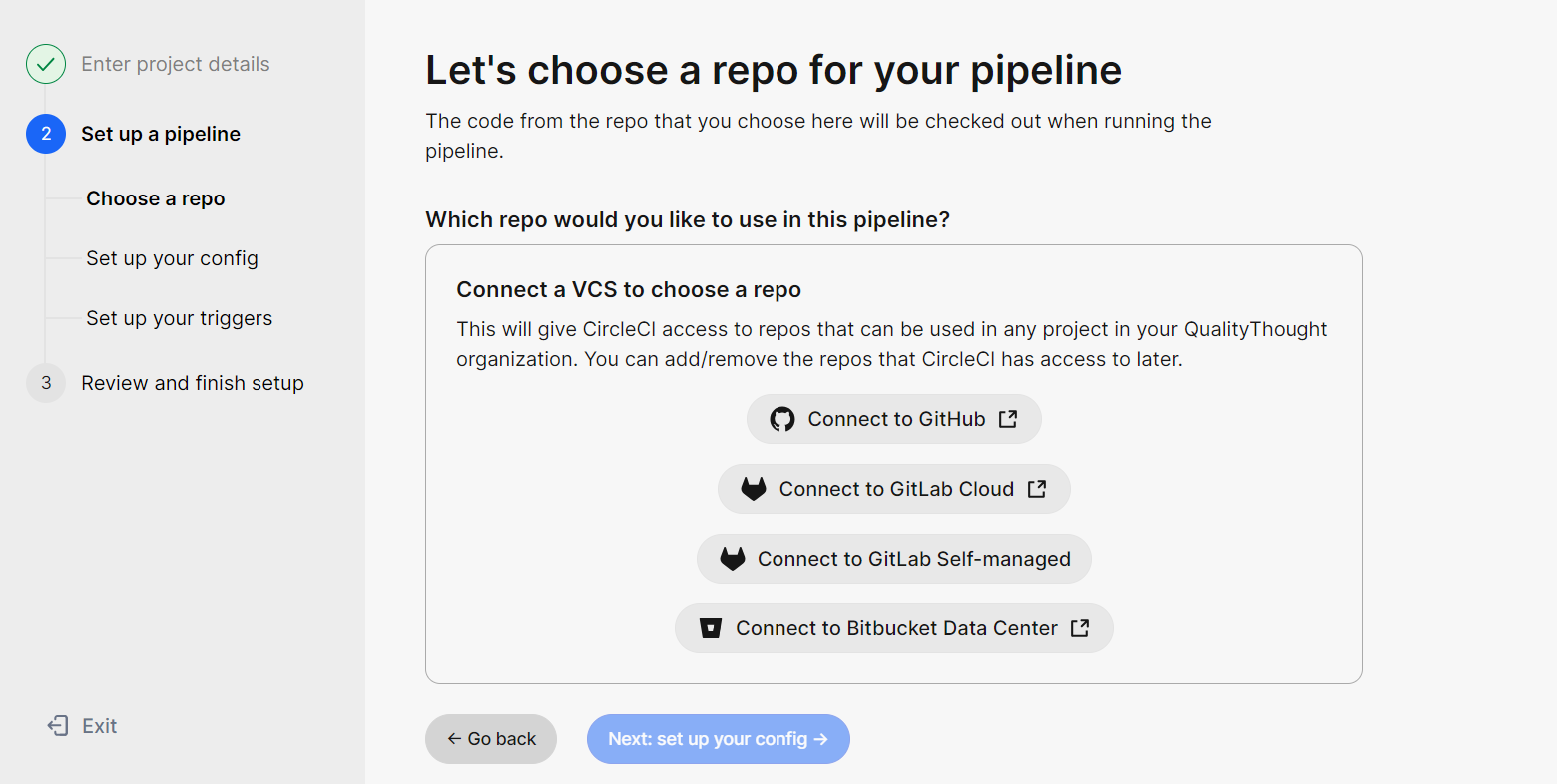
Enter the project name :



Setup a pipe line

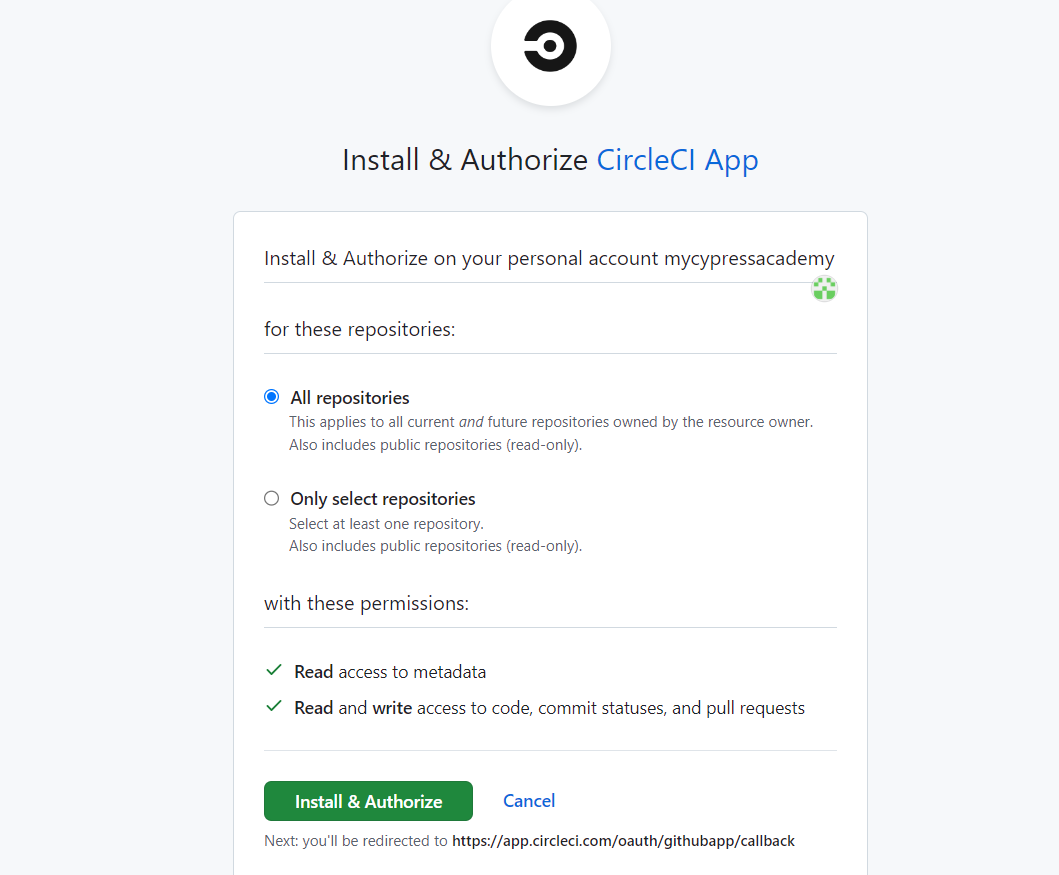


Choose a repository to build and run .

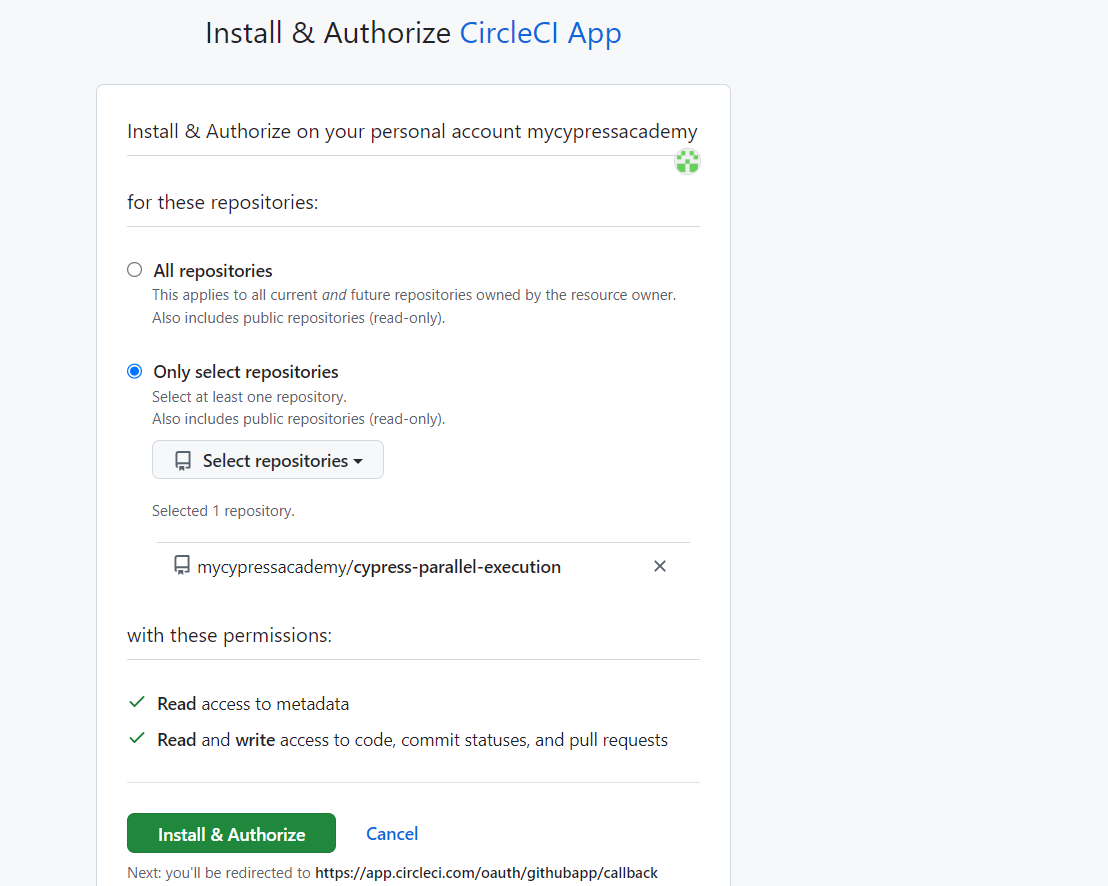


Select Github

Circle and github integration is happening through below screen authorization

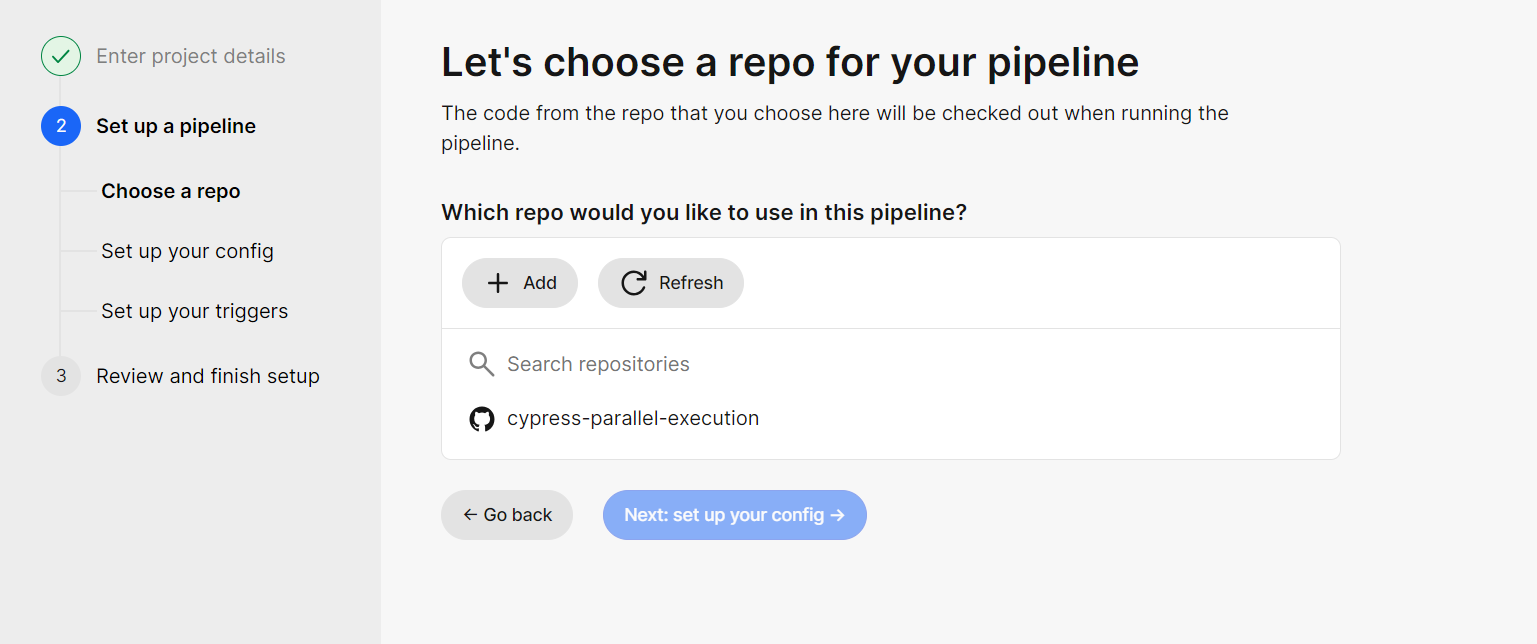


Selected only one repository to conduct parallel execution.

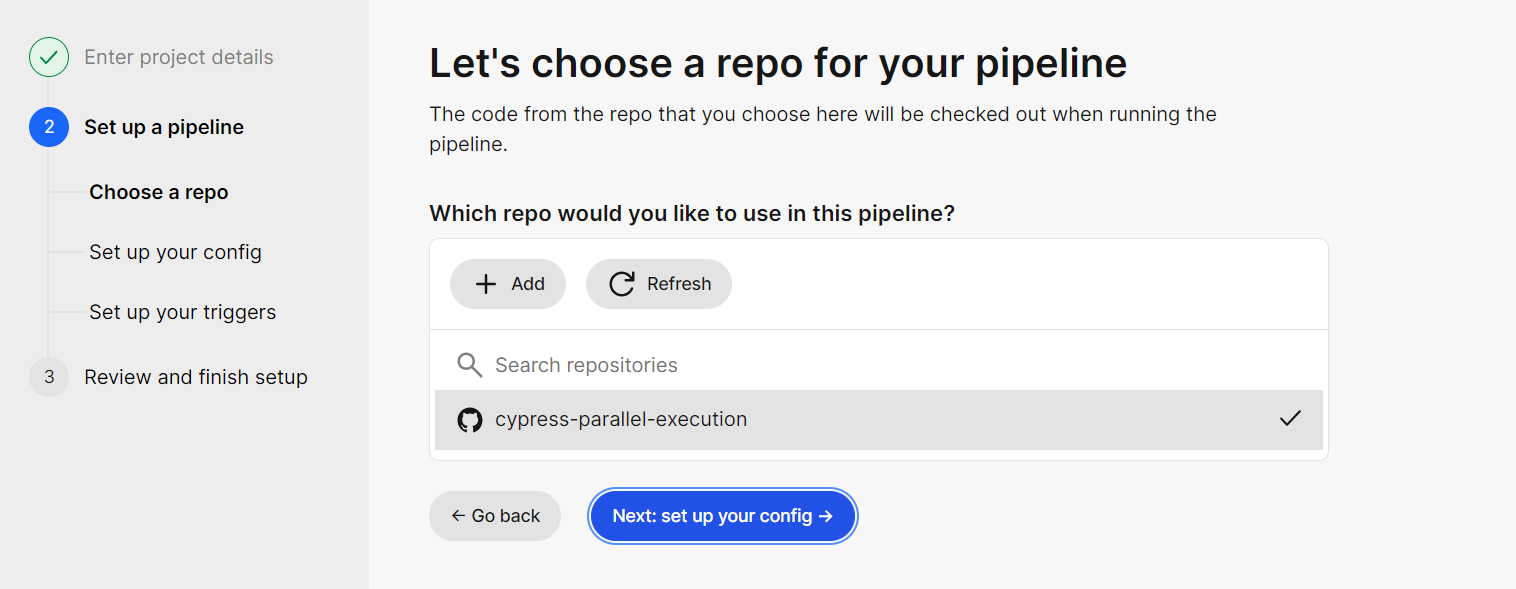


Install the necessary circleCI application at your repository to connect circleCI piple line with github repository checkins and commits.

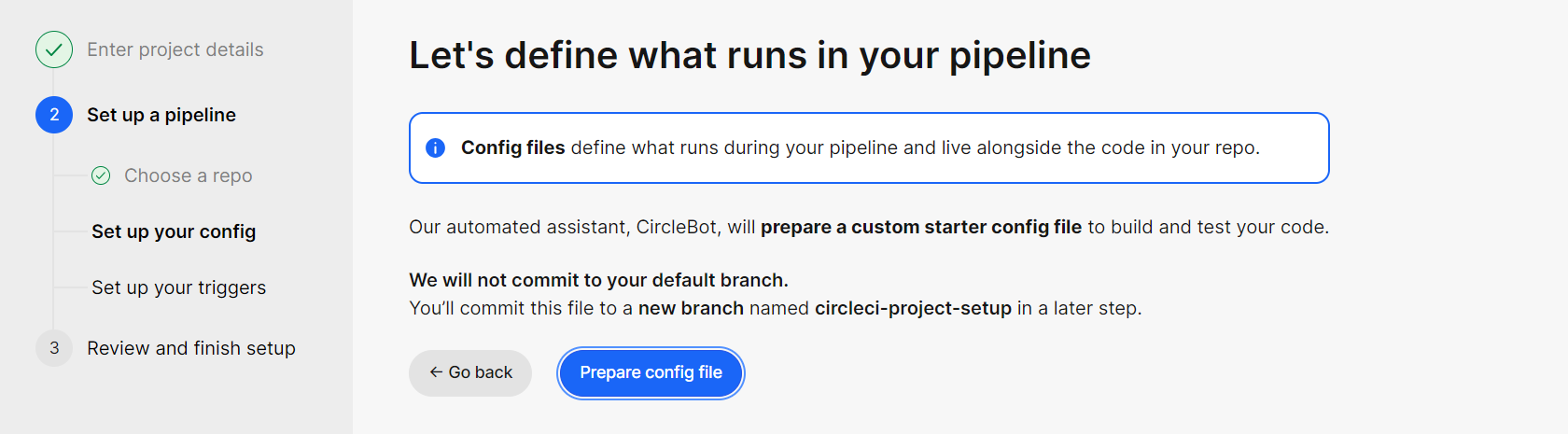
Selected repo should be displayed after connection



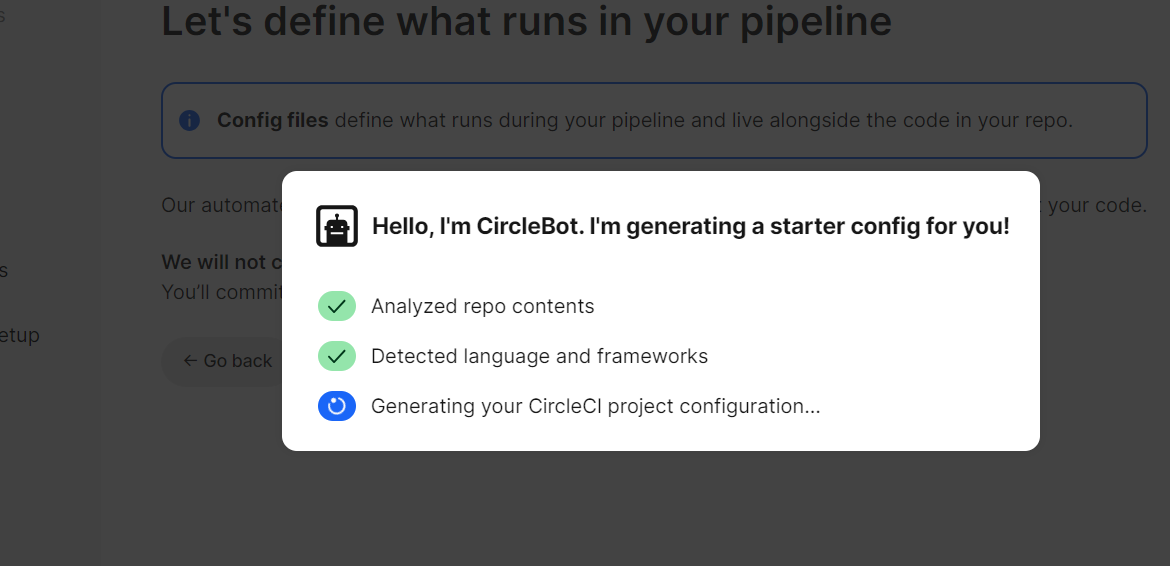
Select repo to enable the Next step.



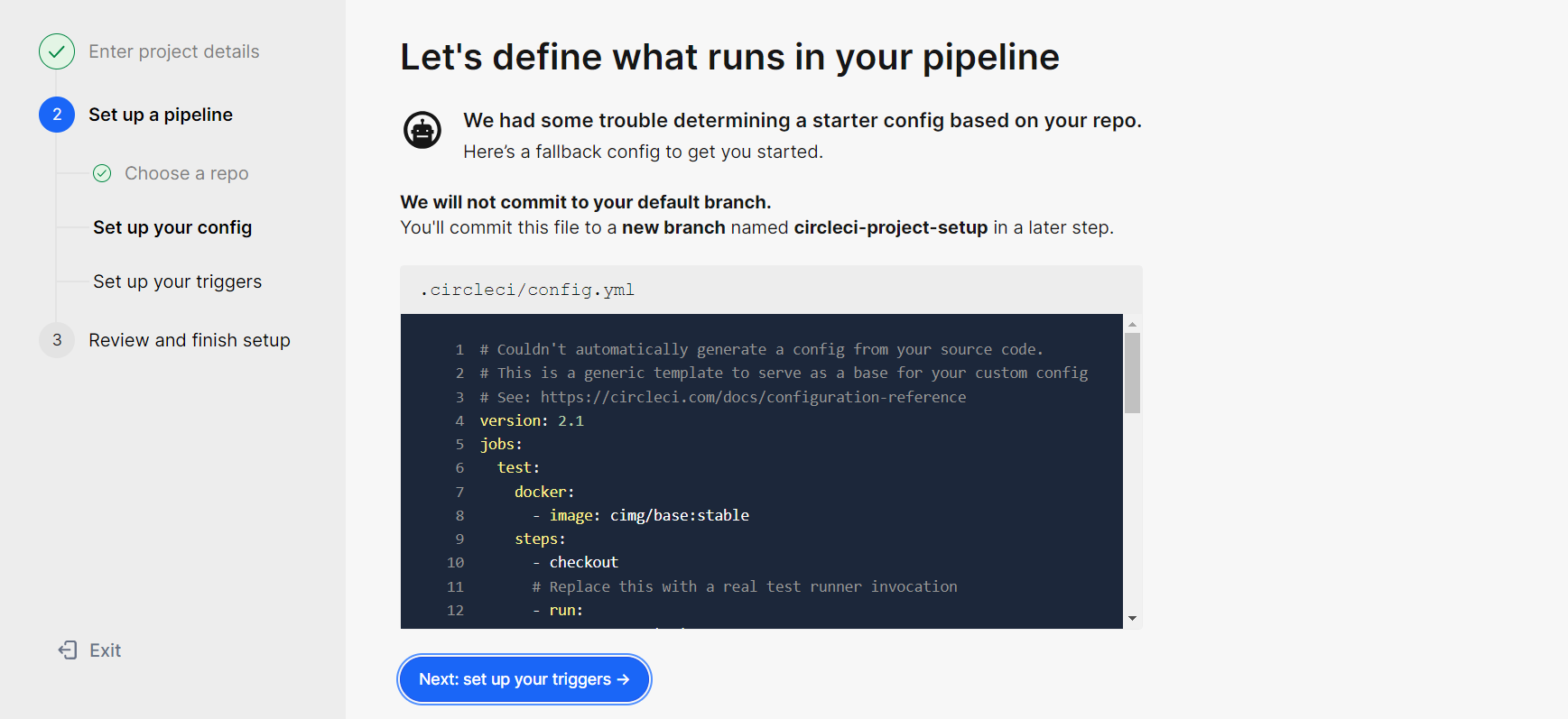
Next step is to configure the pipeline with config.yml file



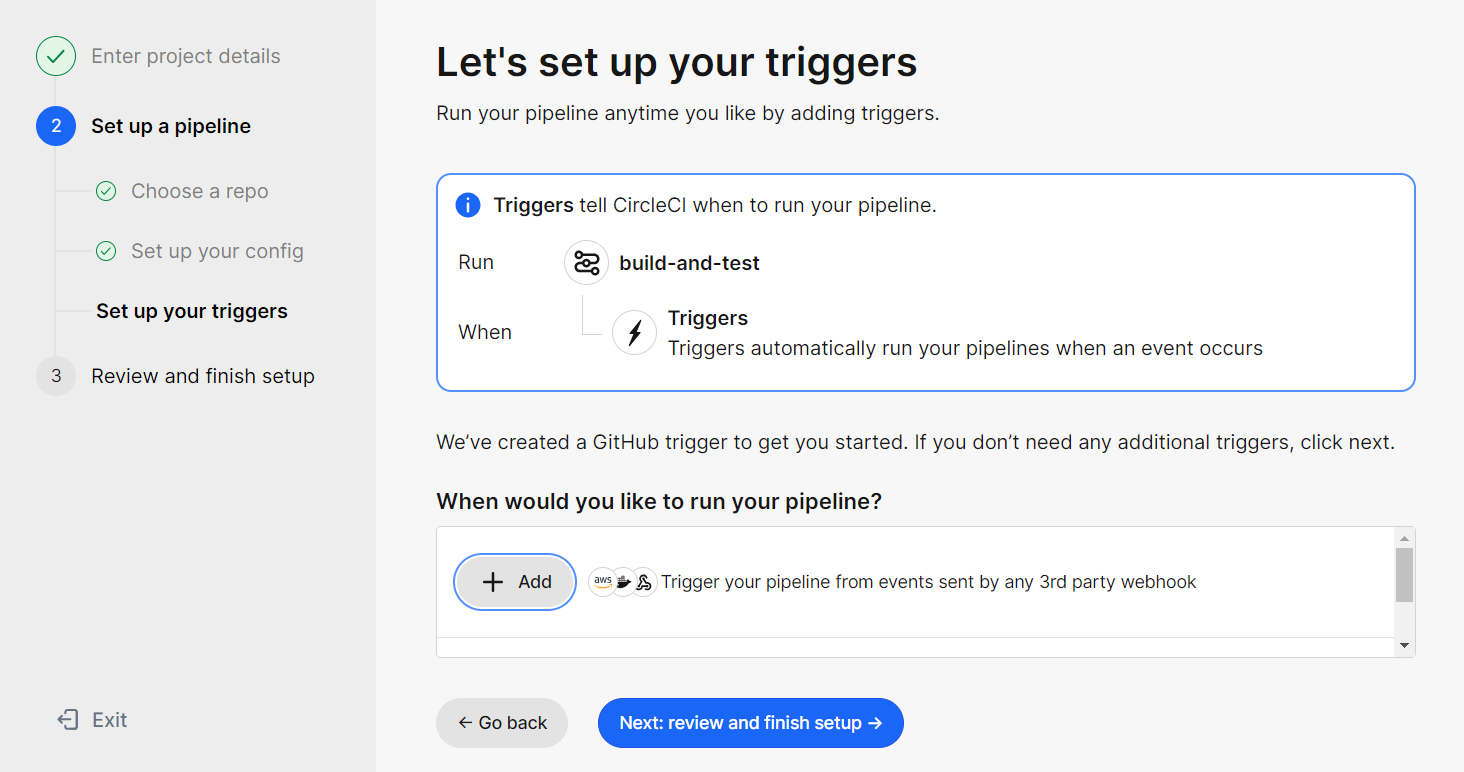
CircleCI bot will help you to create config file in the repository



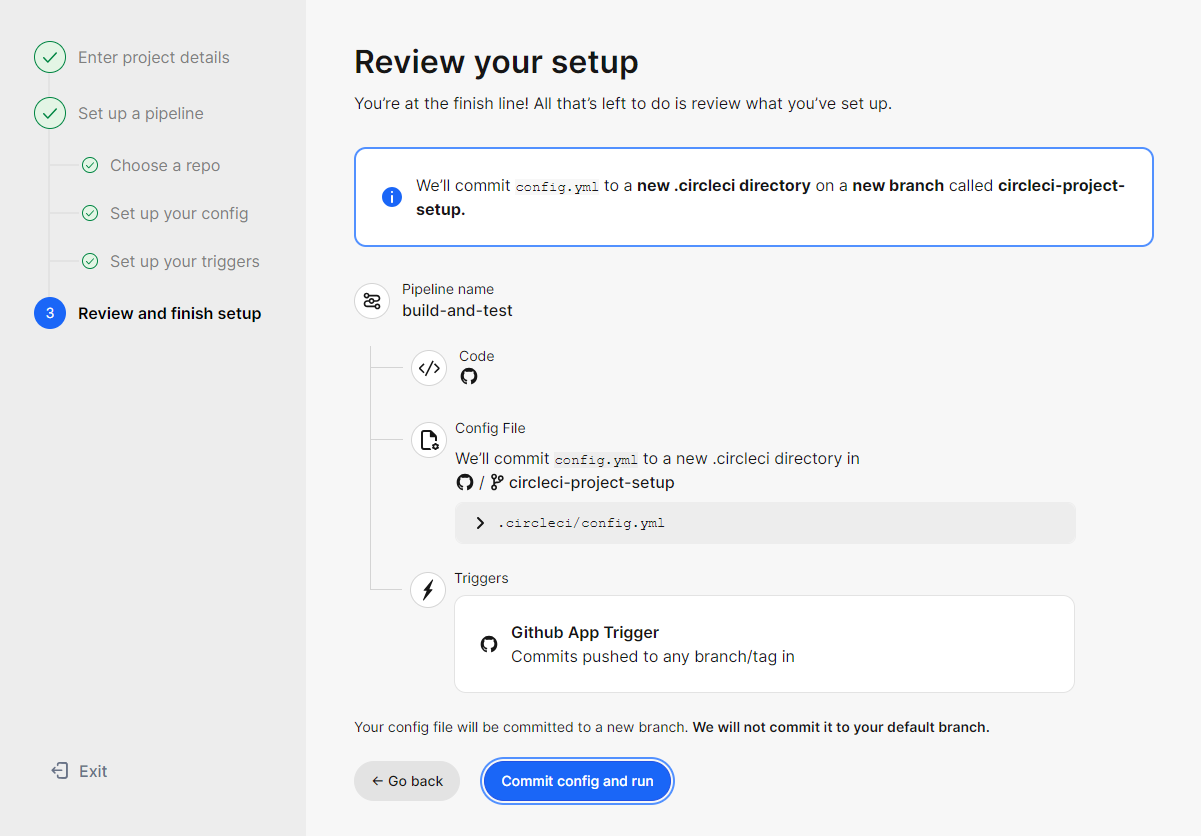
Sample config.yml is added to your repo



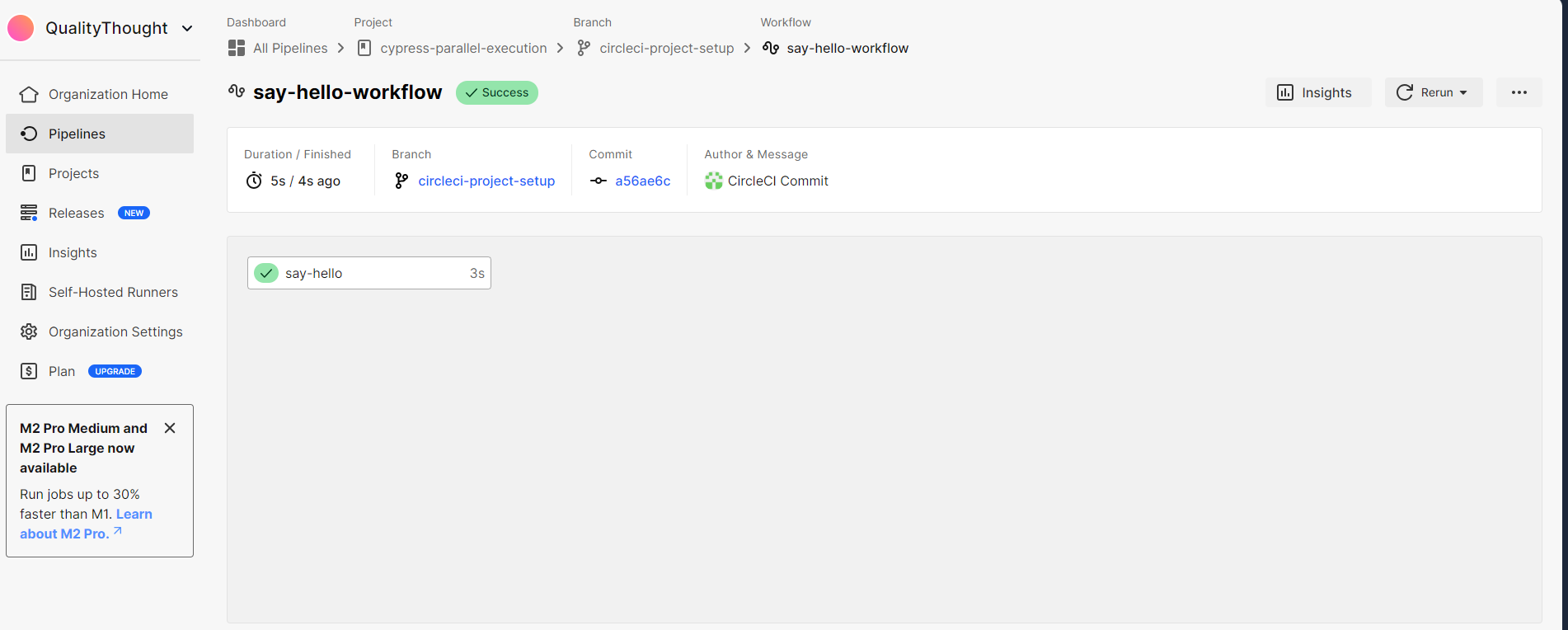
Review and finish the setup



Review the outline settings in project-pipeline



Your run for the above settings is initiated and done with success.



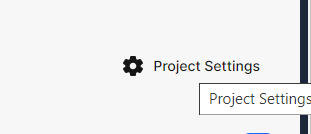
Once Pipeline is created and successful, do the below simple changes to run the cypress tests with the help of circleCI parallel execution

1. projectId: "jk7faz", in the cypress.config.js file
2. Add environmental variable :

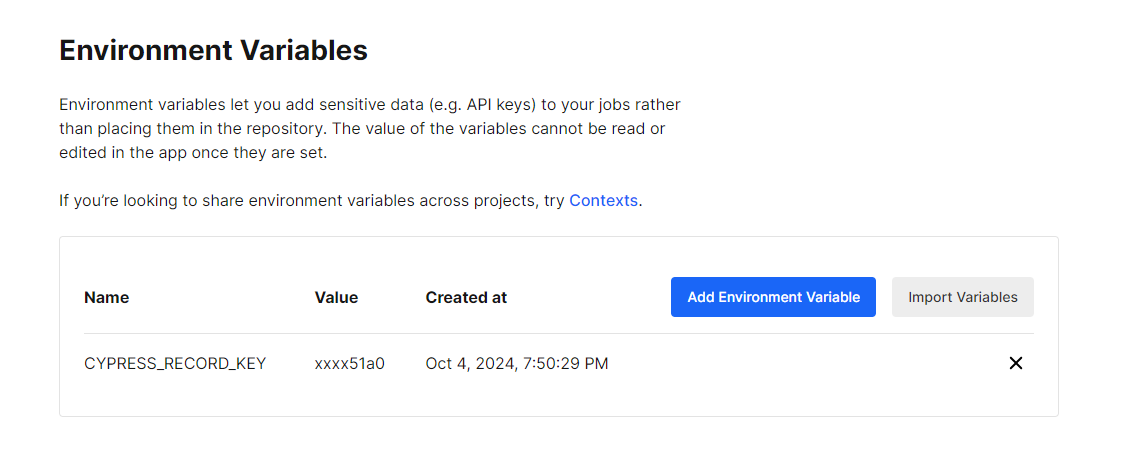
Example record key :

CYPRESS\_RECORD\_KEY

2f47ca88-c5d0-4d93-98db-d2051dfde4b1

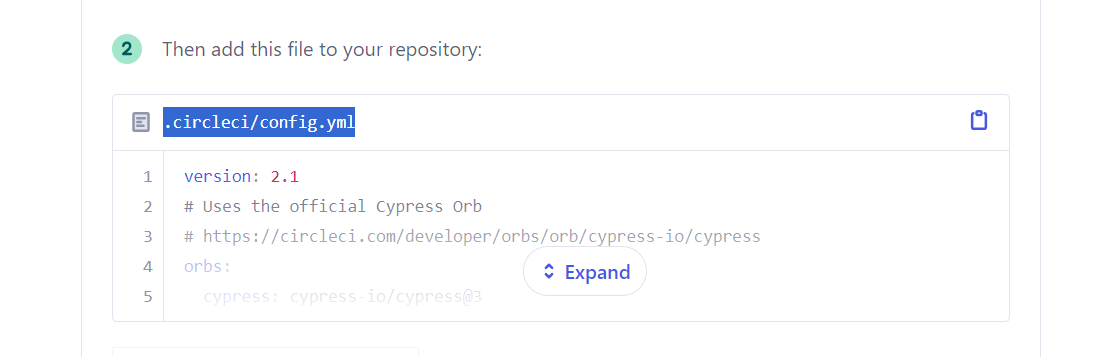


In the project settings navigate to Environmental Variables section and create a new environmental variable



Then create a folder with the name “.circleci “ with in the folder create a file config.yml

Copy the code from the cypress cloud



Copy the data from cypress cloud and paste in project folder >>.circleci/config.yml

Working config.yml file

version: 2.1

# Uses the official Cypress Orb

# https://circleci.com/developer/orbs/orb/cypress-io/cypress

orbs:

  cypress: cypress-io/cypress@3

workflows:

  build:

    jobs:

      - cypress/run:

          # For recording and parallelization to work you must set your CYPRESS\_RECORD\_KEY

          # in CircleCI → Project Settings → Environment Variables

          # Records in parallel to Cypress Cloud

          # https://docs.cypress.io/guides/guides/parallelization

          install-browsers: true

          parallelism: 2 # Uses 2 parallel instances

          # Starts web server for E2E tests - replace with your own server invocation

          # https://docs.cypress.io/guides/continuous-integration/introduction#Boot-your-server

          start-command: npm run start

          cypress-command: 'npx cypress run --record --parallel --browser chrome'

if you want to run scripts other than electron browser lets add this step

          install-browsers: true

if you want to increase or decrease the parallel machines update the for parallesim

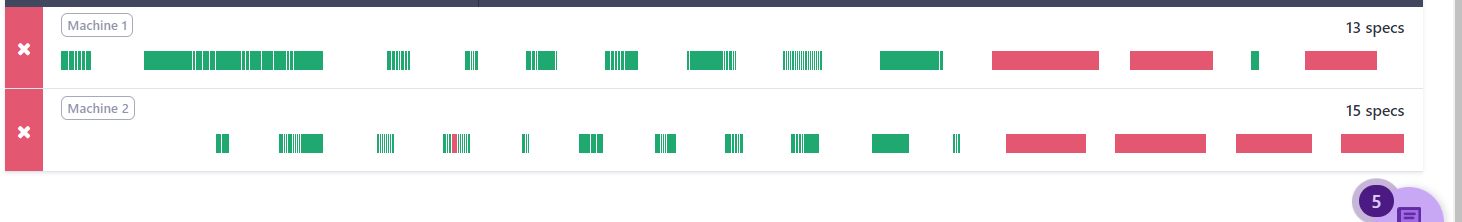
parallelism: 2 # Uses 2 parallel instances

parallelism: 4 # Uses 4 parallel instances

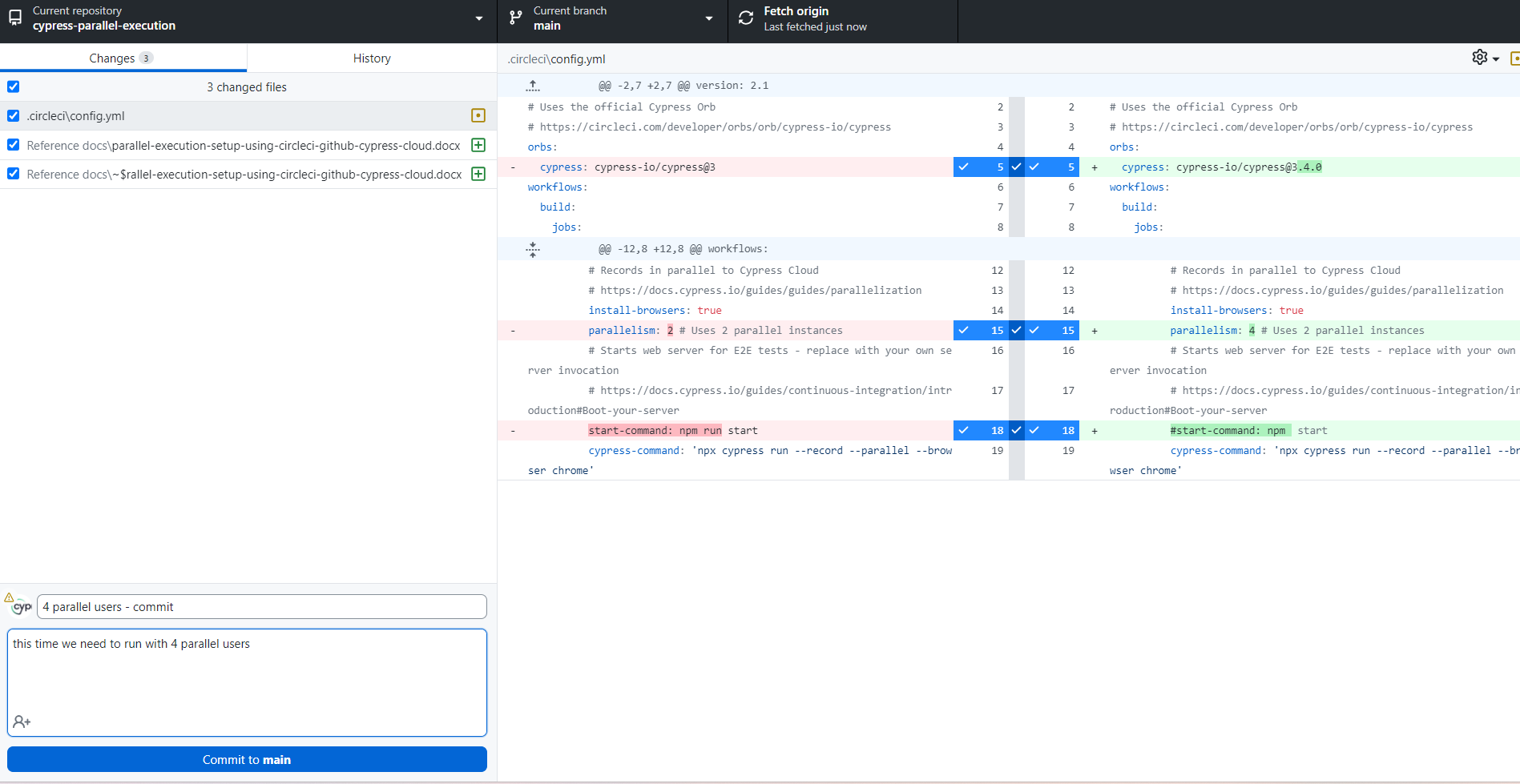
Once all setup is done as per above conditions, lets create a scaffold examples in the e2e folder and commit those tests to the repository

Automatotically your tests run through the circleci and update those results in the cypress cloud – your project

As you are using 2 machines - result is two machines

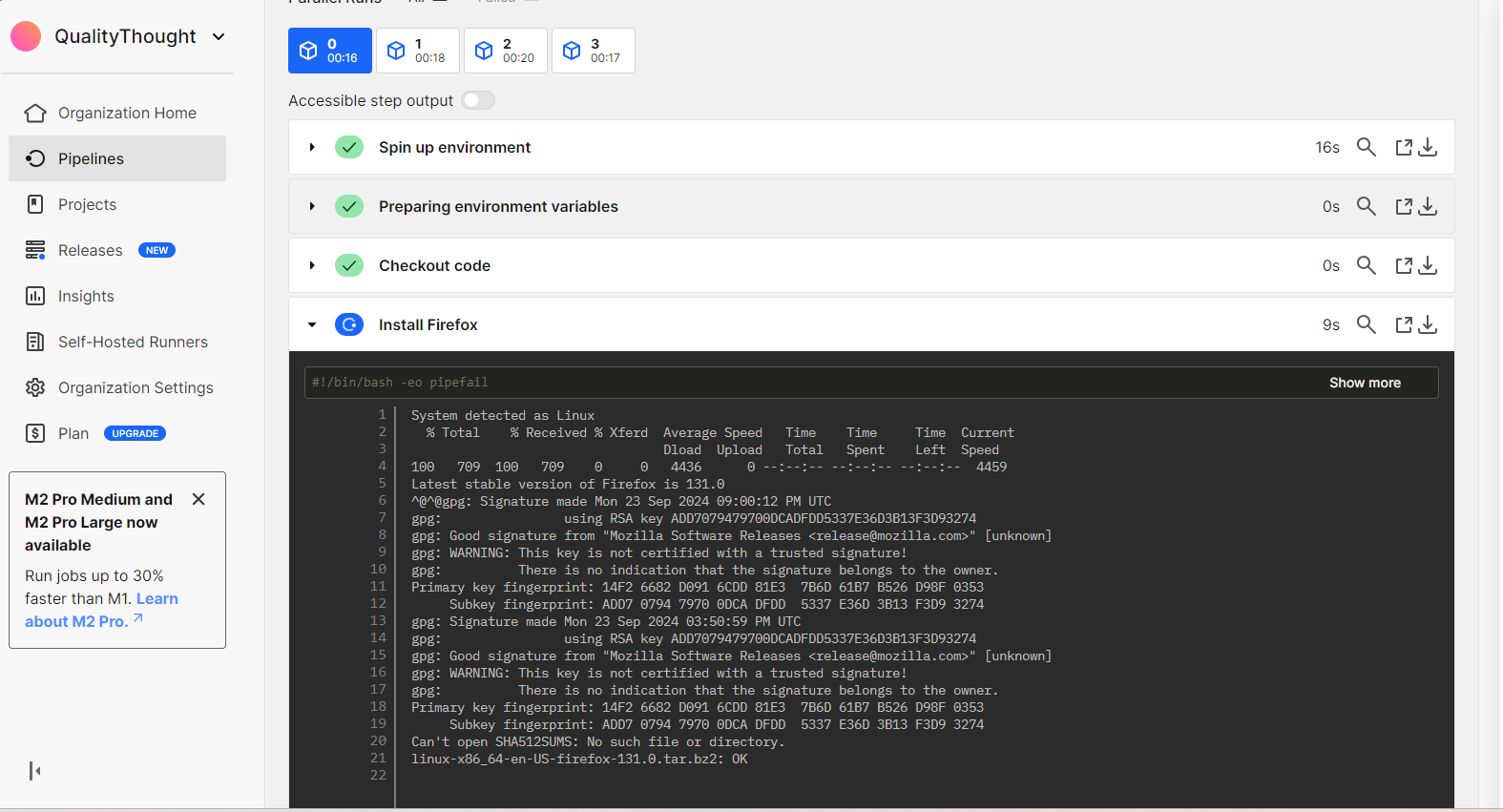


Commit the 4 user changes into the repo using github desktop

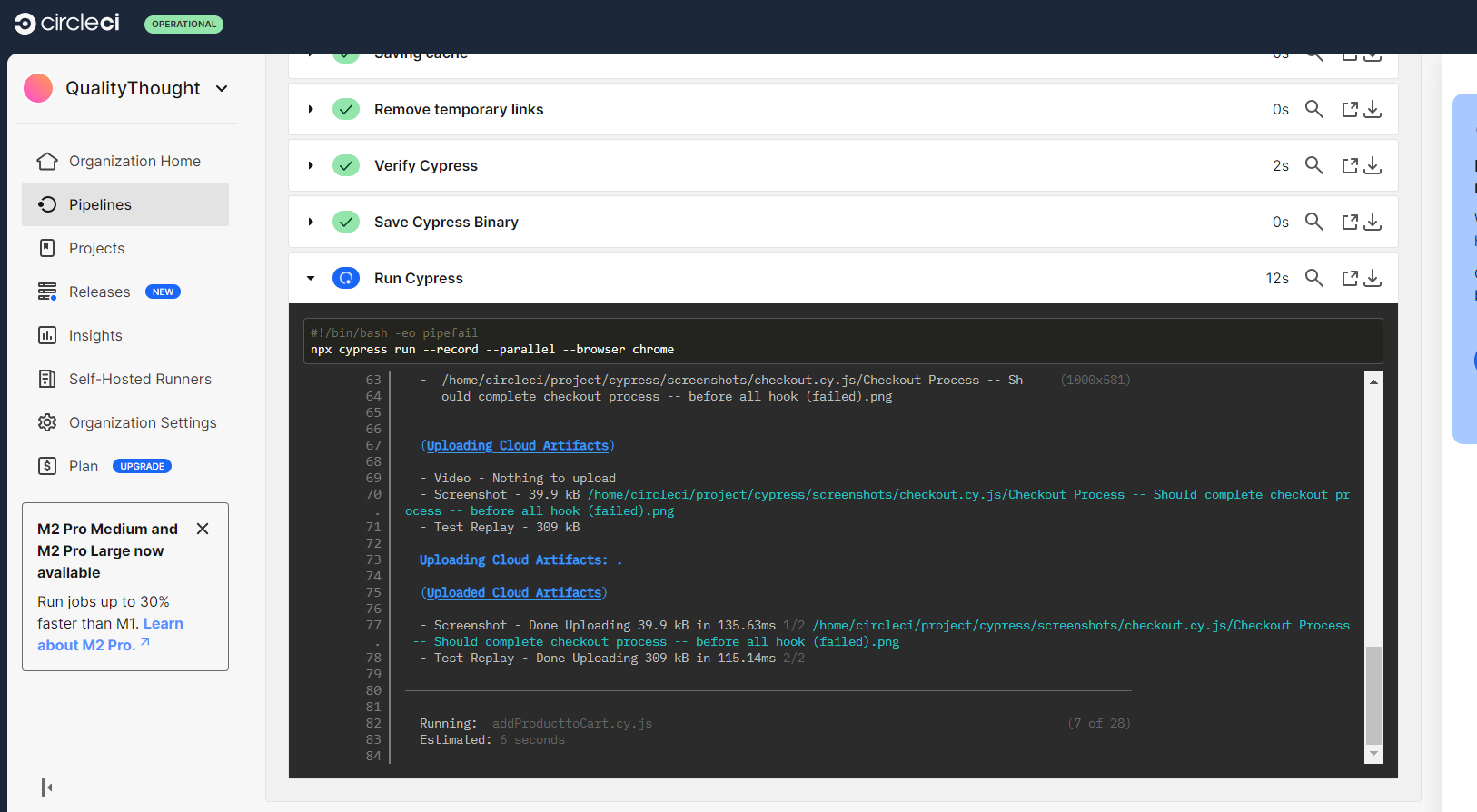


Then push the changes to origin

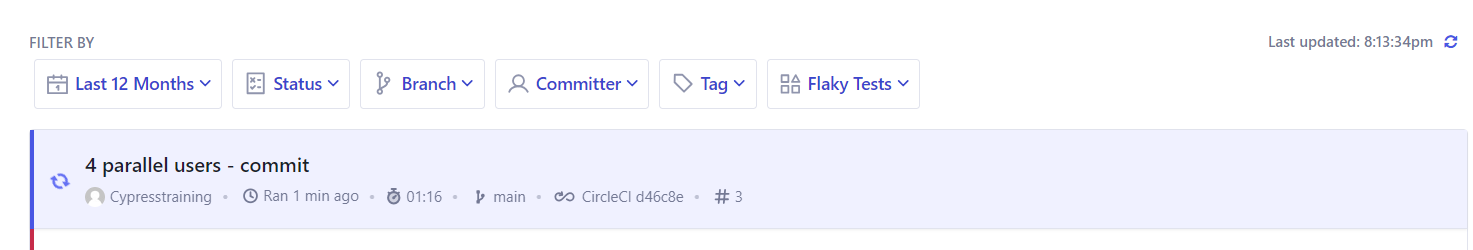
CircleCI initiate the auto run based on your commit –



Your tests are running exactly similar to the CLI statements

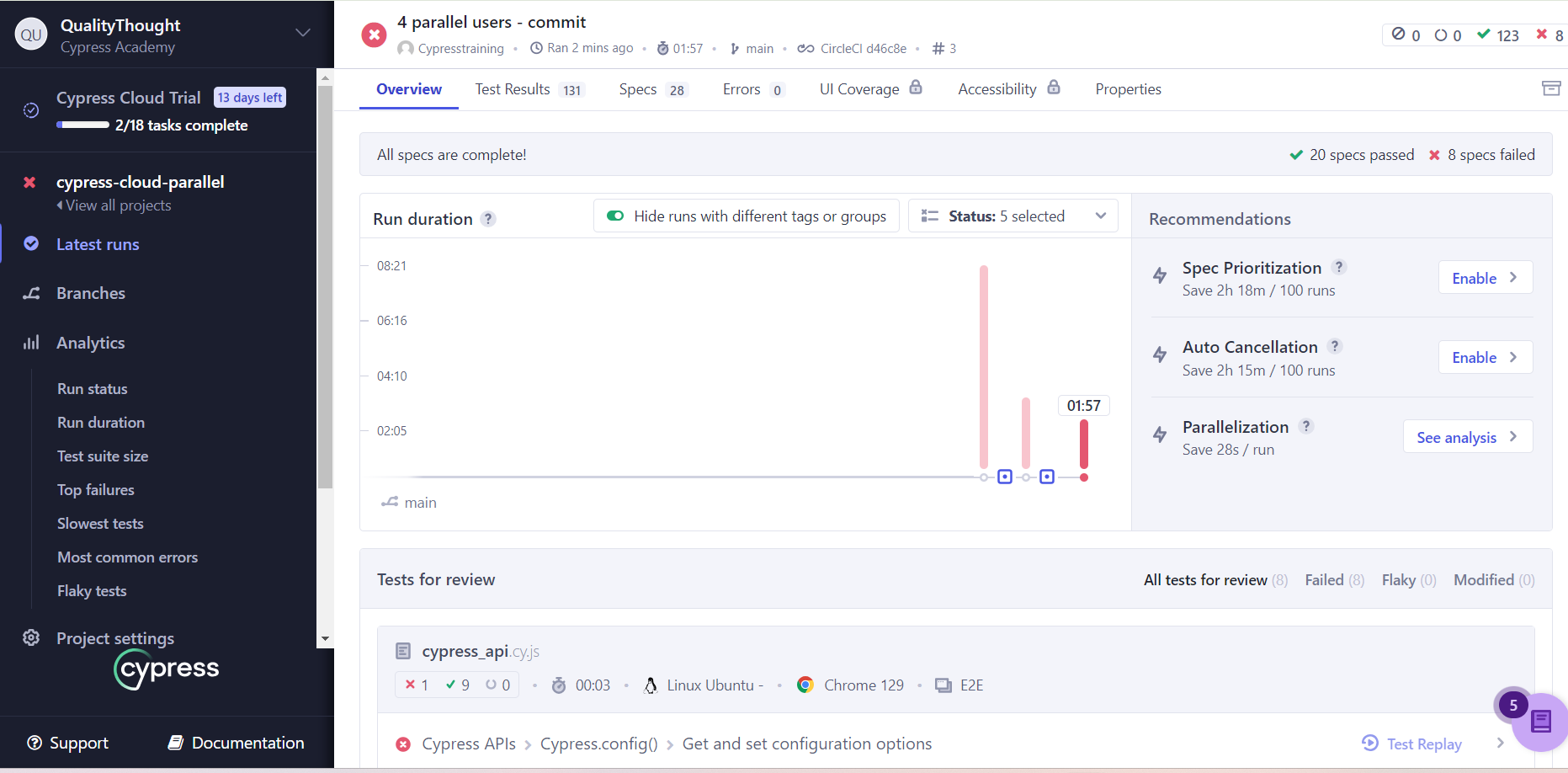


In cypress cloud we can observe test runs –

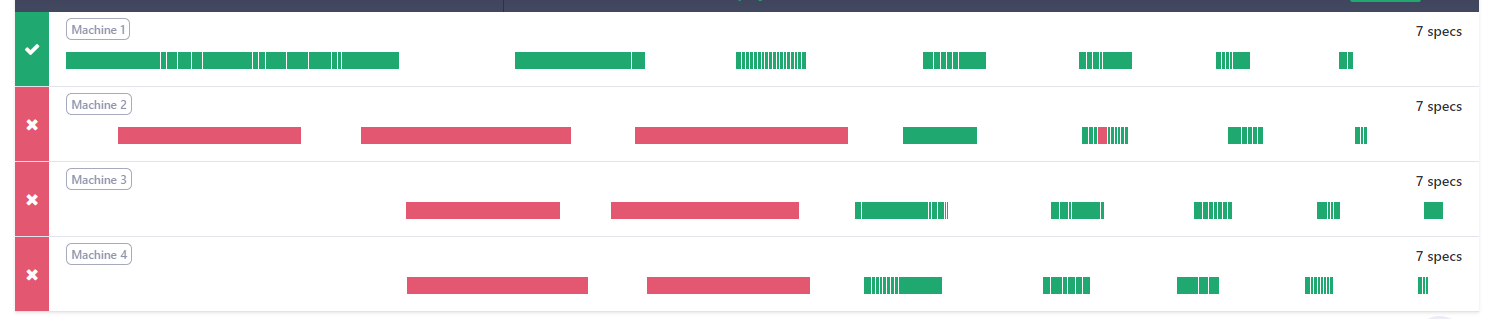


Expand the results

In the resulting screen you can see the test results



Click on see analysis for – Parallelization



We can analyse the machine wise results

Spec wise results to understand the failures.