Global Knowledge®

ilionx

Module – Introduction to CI/CD

What is CI/CD?

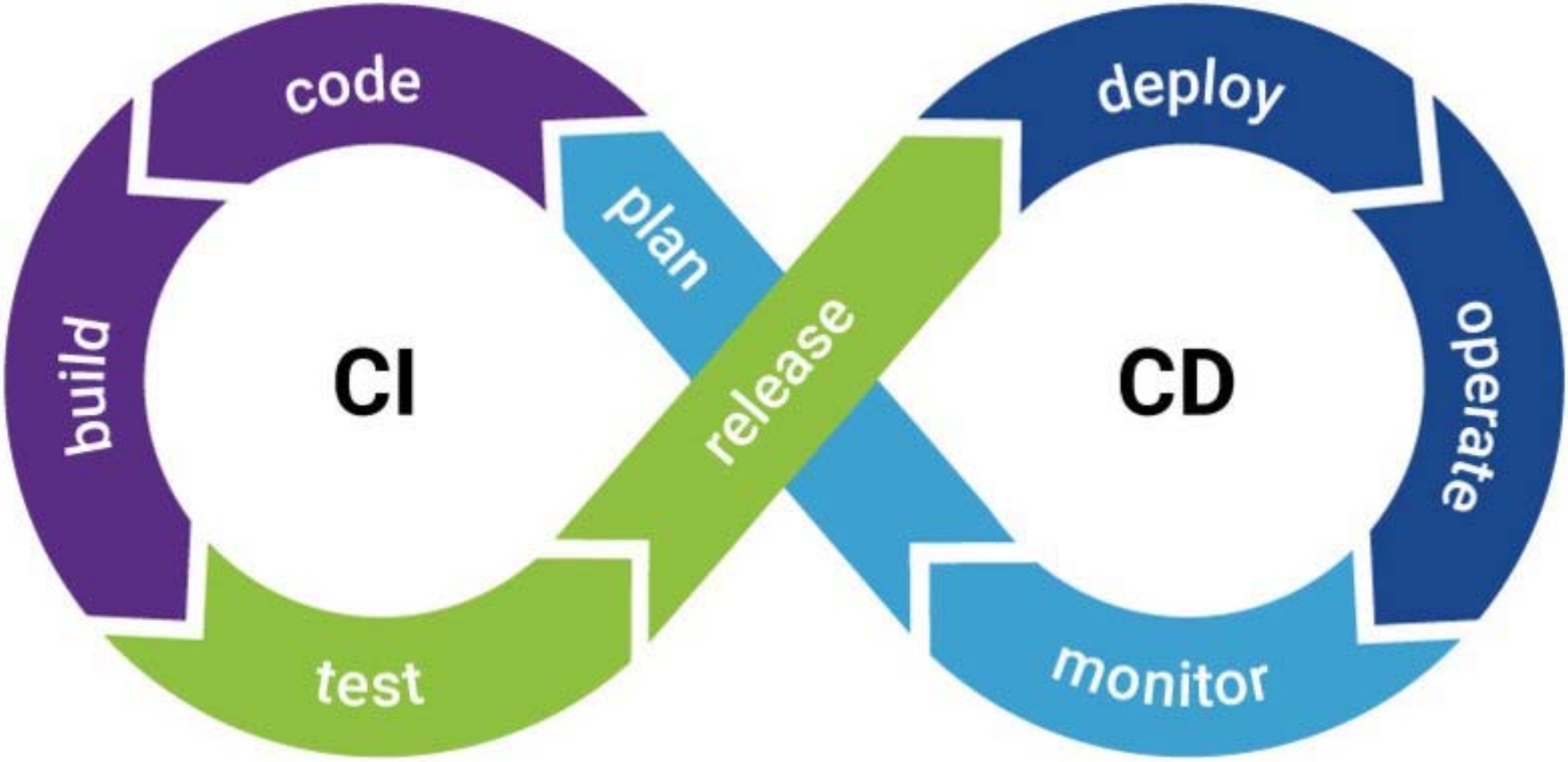
Acronym: *Continuous Integration* & *Continuous Delivery*

(or *Continuous Deployment*)

"CI/CD is a method to frequently deliver apps to customers by introducing automation into the stages of app development"

https://www.redhat.com/en/topics/devops/what-is-ci-cd

CI/CD Visualisation



Phases

- Ongoing automation and continuous monitoring
- Respond to the lifecycle (events) in apps
- Phases
  - Designing & Coding
  - Testing & merging into branch
  - Release
  - Deploy to production
  - Monitoring
  - Planning new features, solving bugs, etc.

Continuous integration

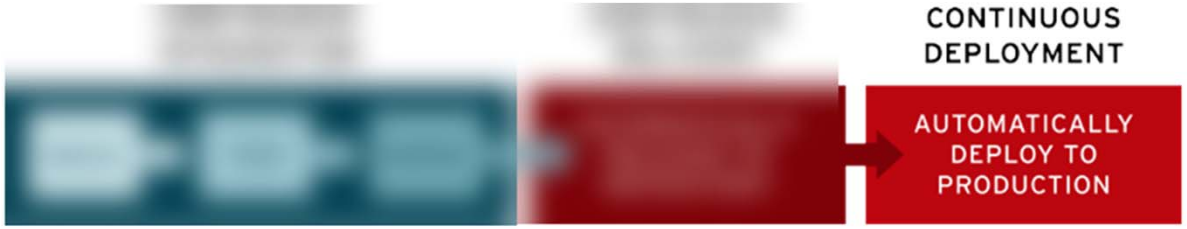
- In bigger apps: multiple developers working simultaneaously on different features of the same app
- Chances of conflicts if there is one big ‘merge day’
- CI: merge changes back to a shared branch or *trunk*
- Changes are validated and tested automatically during the process
- Conflicts? Easier to solve b/c of relatively small changes

Continuous Delivery

- CI OK?
  - Continuous Delivery automates the release of that validated code to a repository
- So, No CD without CI in the pipeline!
- The main branch or *trunk* should always be 'production ready'
- Again: automation!

Continuous Deployment

- Continuous Deployment automates releasing an app to production
- Because there is no manual gate at the stage of the pipeline before production, continuous deployment relies heavily on well-designed test automation.
- In practice: changes to a cloud application go live within minutes of writing it and checking in!
  - Assuming all tests are passed ;-).
- [cloud] Tools are often used for all three phases



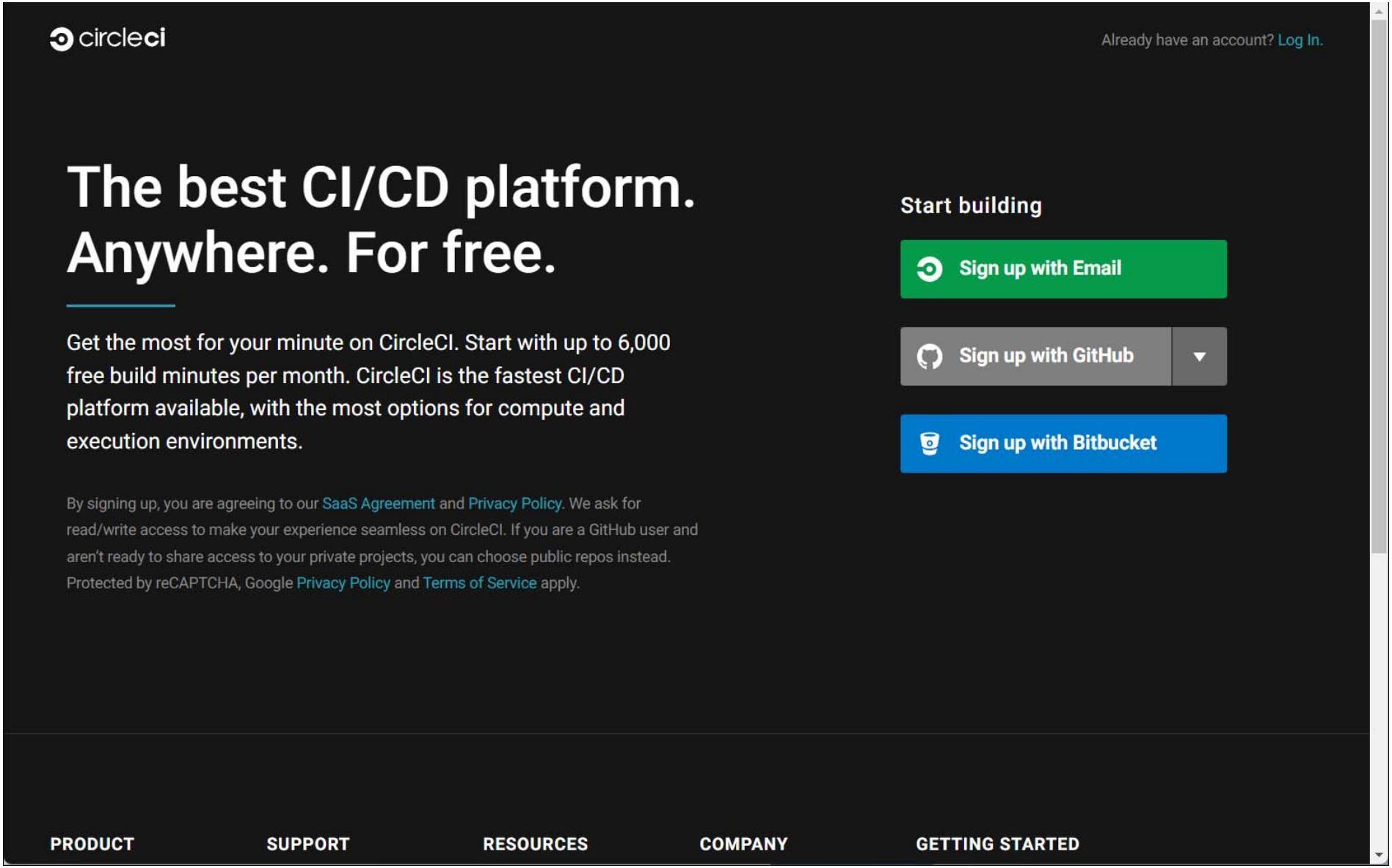
Popular CI/CD Tooling options

tekton.dev/

www.jenkins.io/

- Open source
- On Premise
- Custom Configuration

But there is (a lot) more... for instance:



https://circleci.com/

Also: *managed* CI/CD tooling

- Provided by virtual all public cloud providers:
  - *Alibaba Cloud*
  - *Amazon Web Services (AWS)*
  - *Google Cloud Platform (GCP)*
  - *IBM Cloud*
  - *Microsoft Azure*
  - ...

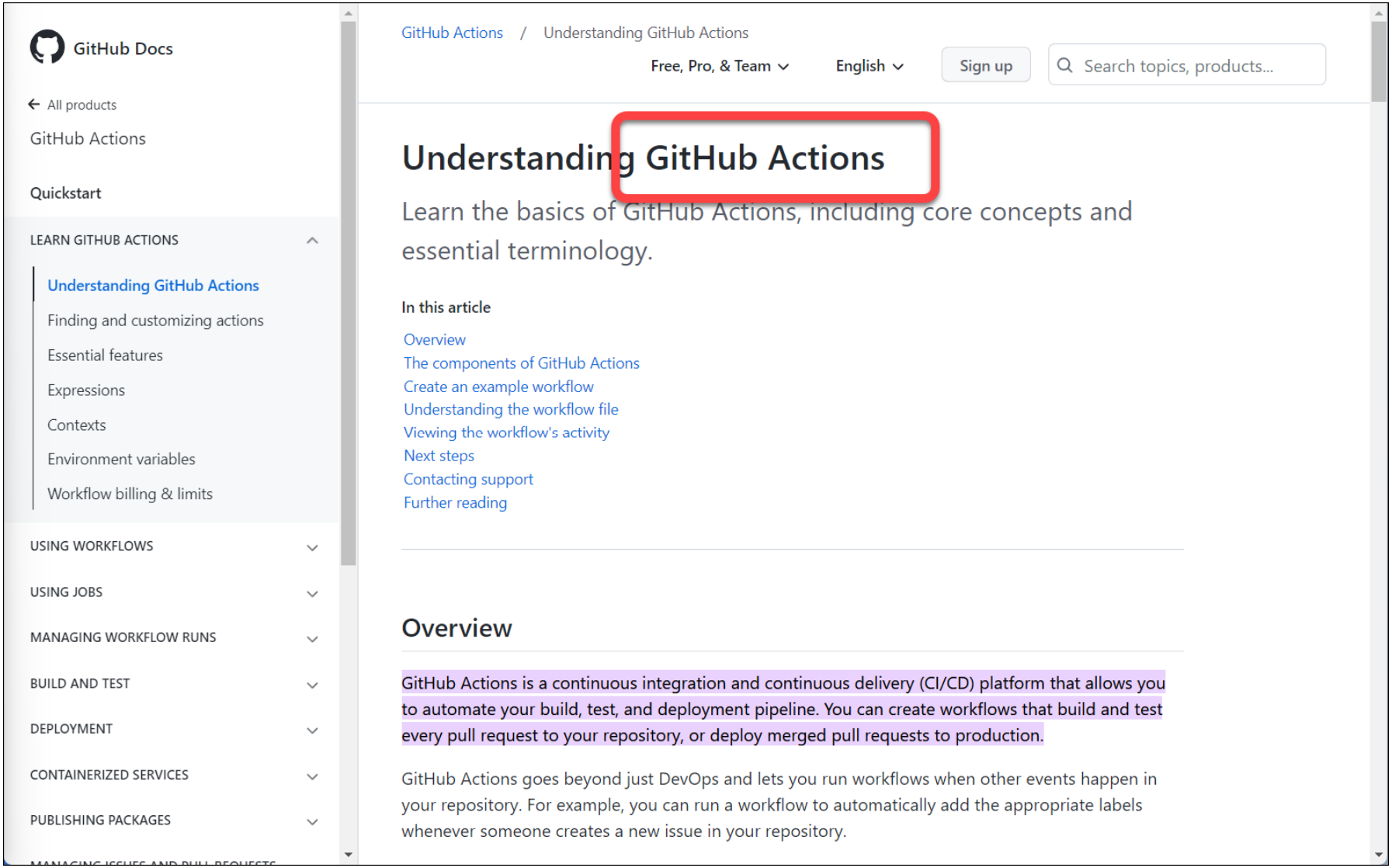
- Cloud provider manages the infrastructure, availability, data storage, virtualization, etc.
- Customer writes configuration scripts, pays bill
  - $$$

Other options

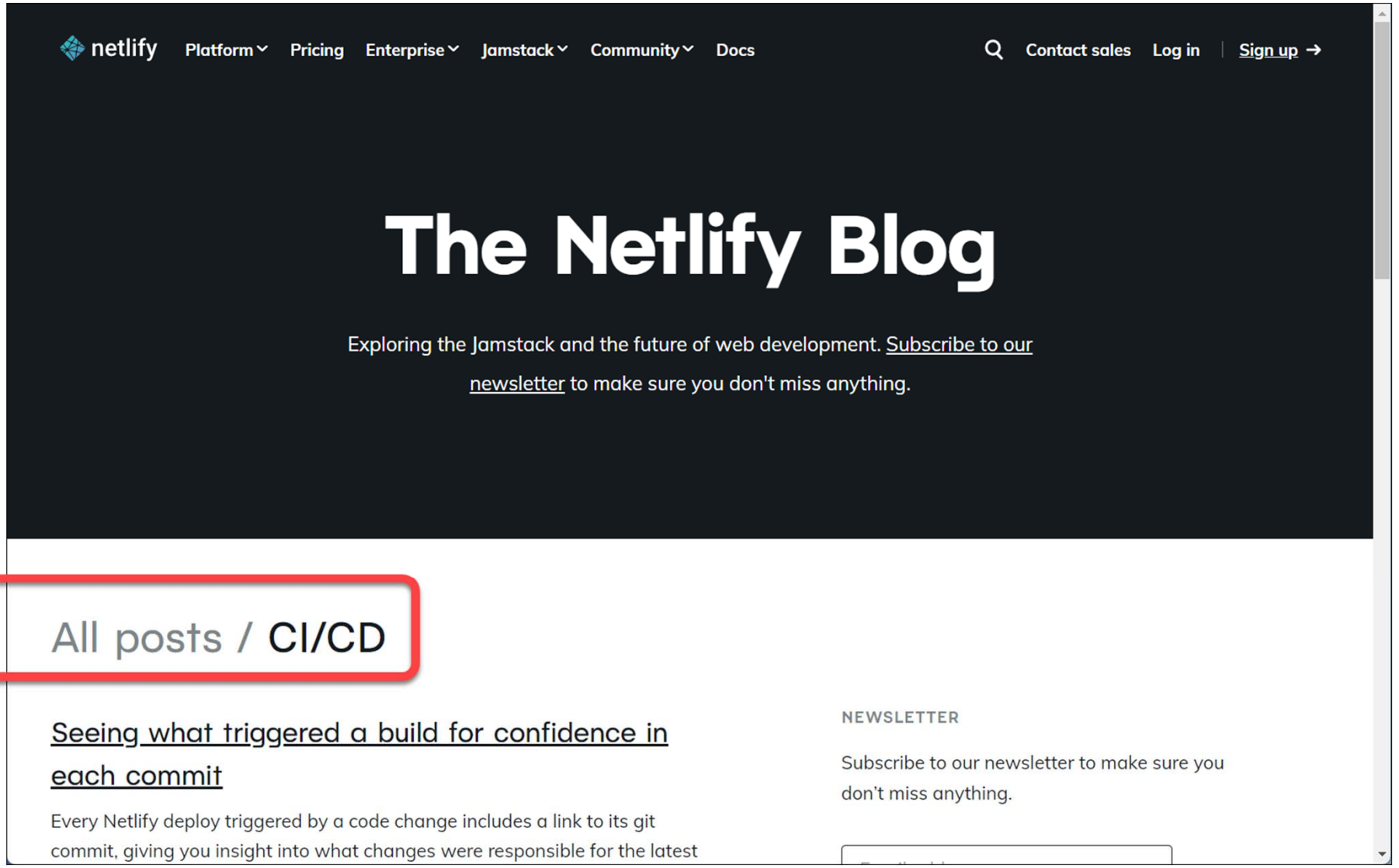
Use SAAS options, for instance provided by (but not limited to!):

How?

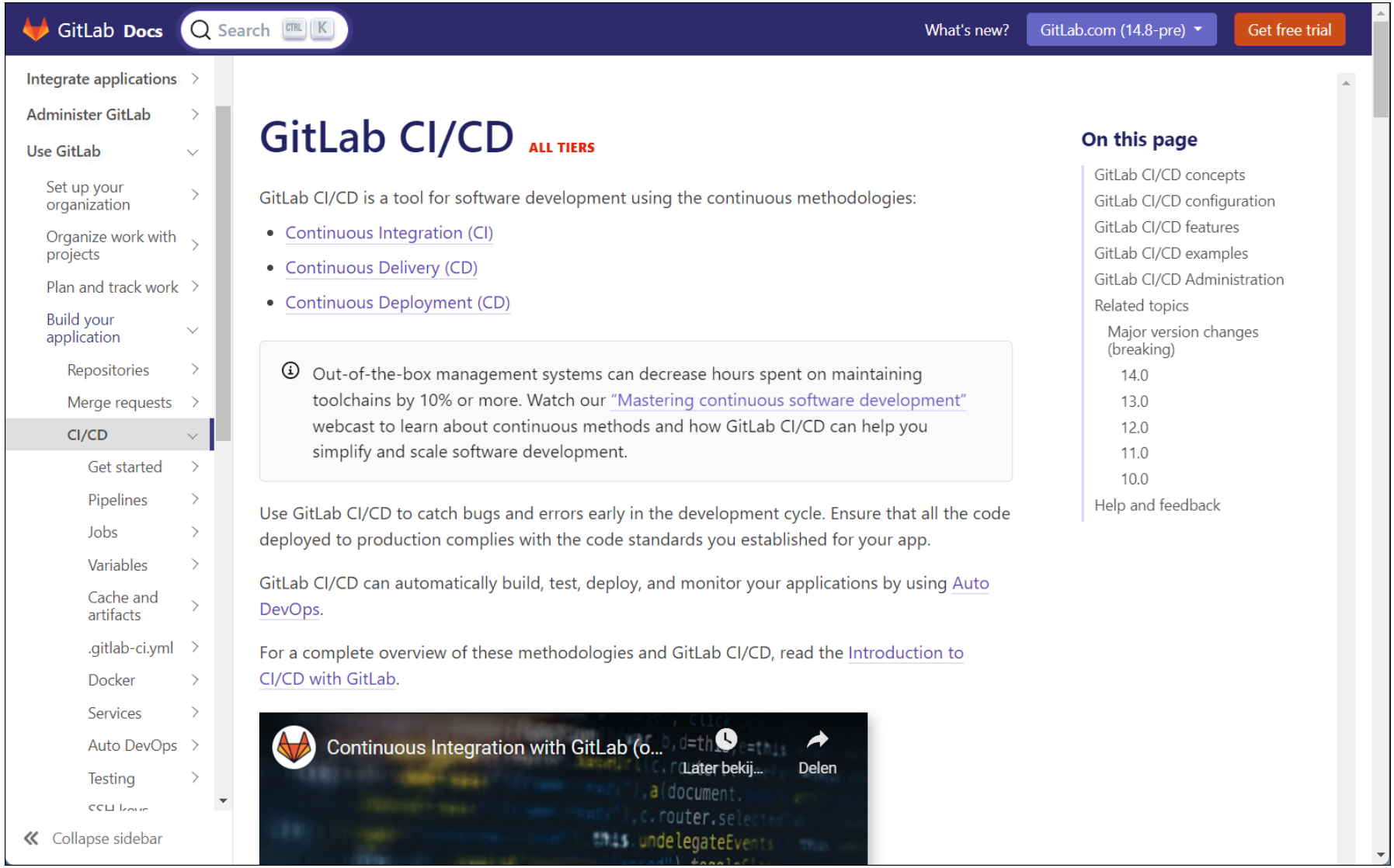
"RTFM"



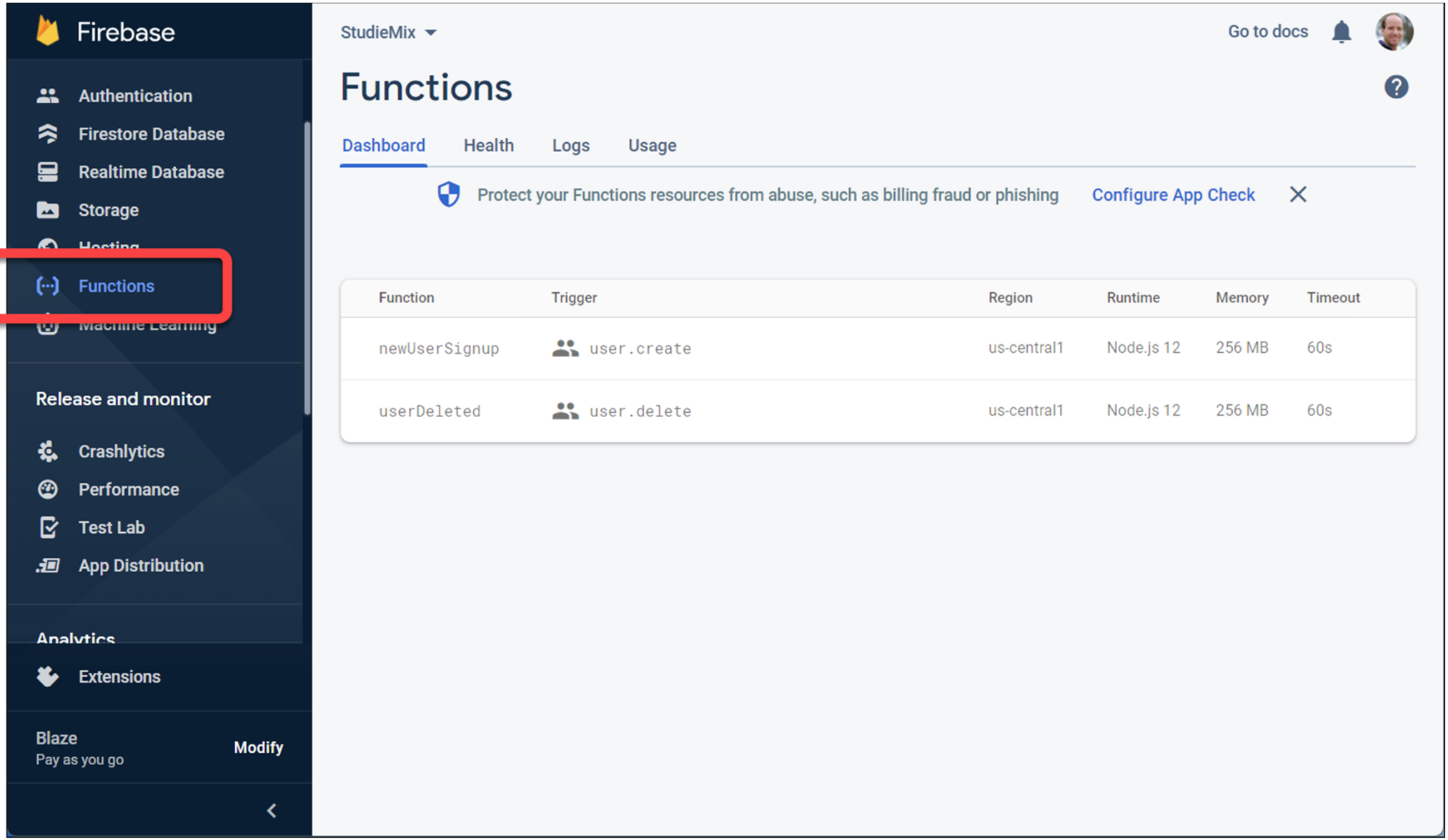
https://docs.github.com/en/actions/learn-github-actions/understanding-github-actions

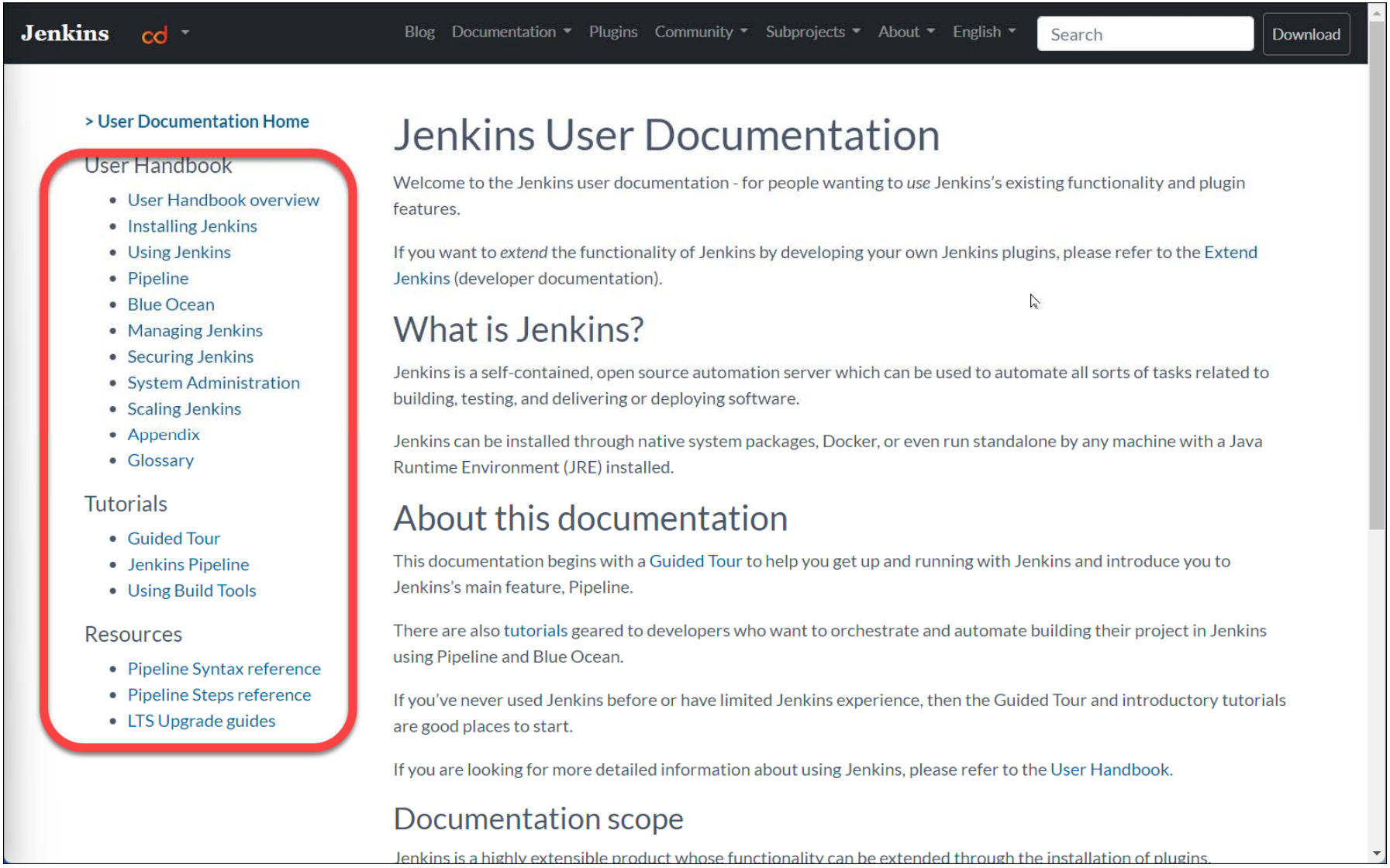


https://www.netlify.com/tags/ci/cd/



https://docs.gitlab.com/ee/ci/


Firebase
Authentication
Firestore Database
Realtime Database
Storage
Functions
Release and monitor
Crashlytics
Performance
Test Lab
App Distribution
Extensions
Blaze
Pay as you go
Modify
StudieMix
Go to docs
Functions
Dashboard
Health
Logs
Usage
Protect your Functions resources from abuse, such as billing fraud or phishing
Configure App Check
Function
Trigger
Region
Runtime
Memory
Timeout
newUserSignup
user.create
us-central1
Node.js 12
256 MB
60s
userDeleted
user.delete
us-central1
Node.js 12
256 MB
60s

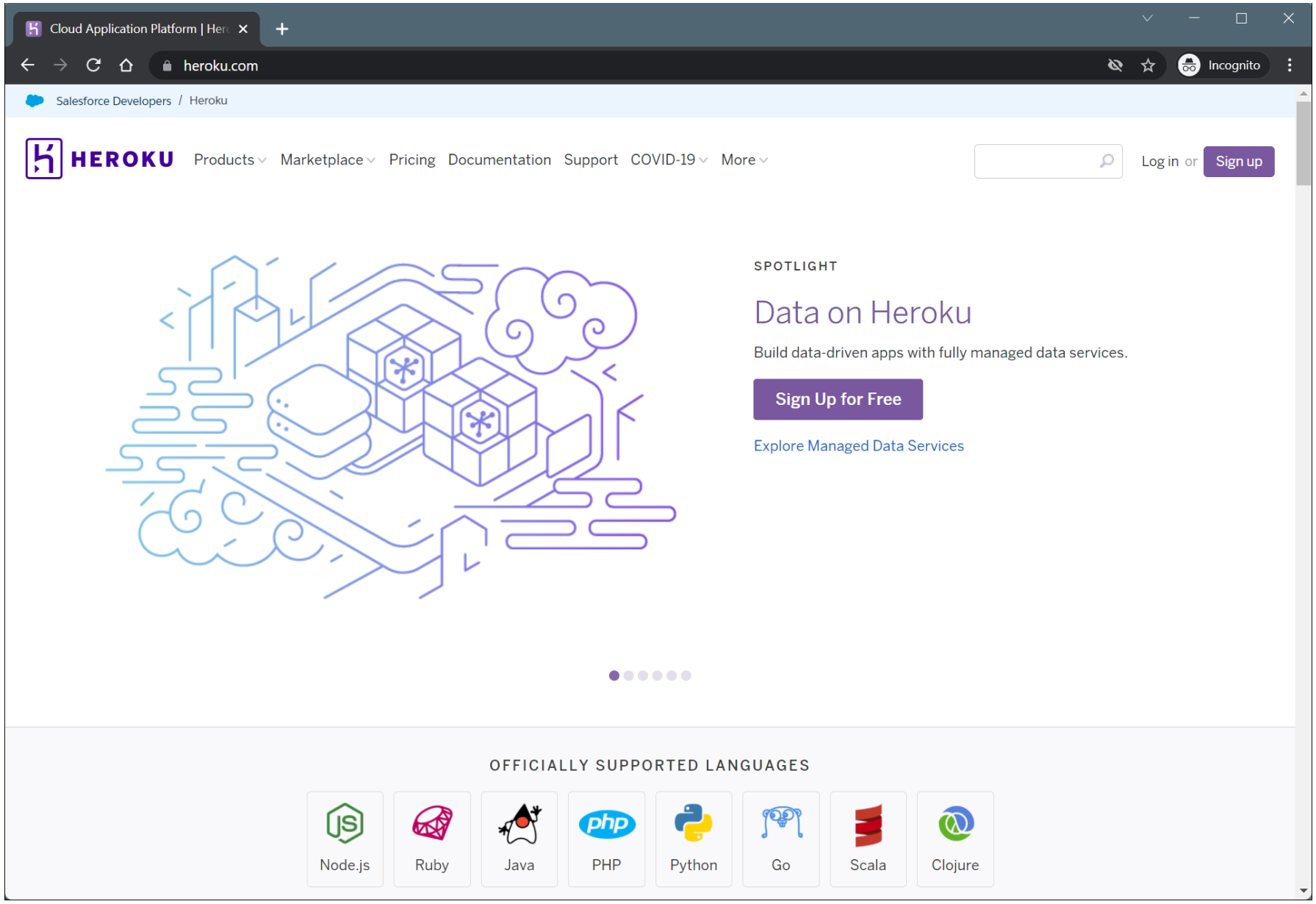


https://www.jenkins.io/doc/

CI/CD Example

Our choice: Netlify + Heroku

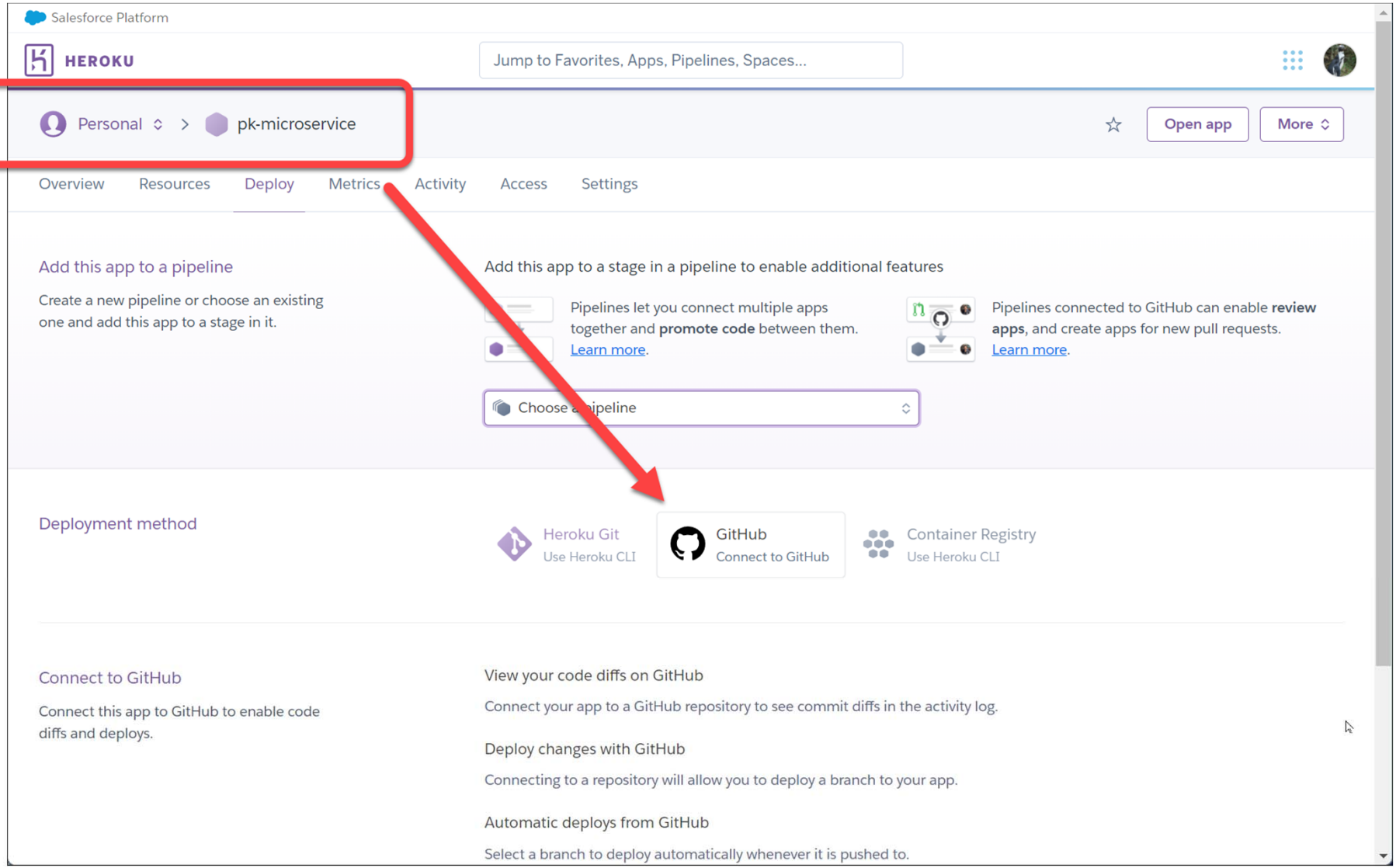
Backend hosting – a node.js hosting provider



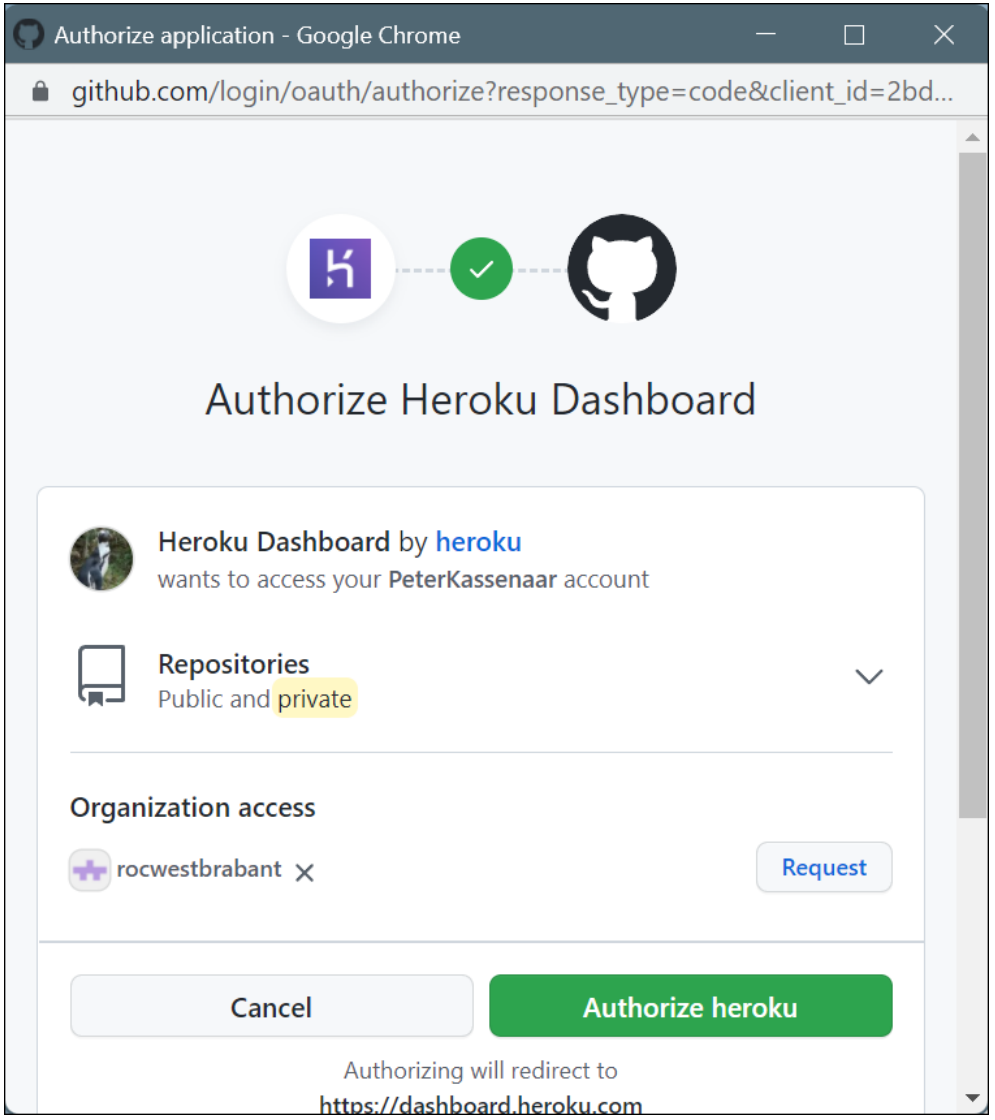
https://www.heroku.com/

Step 1. Deploy Backend to SAAS

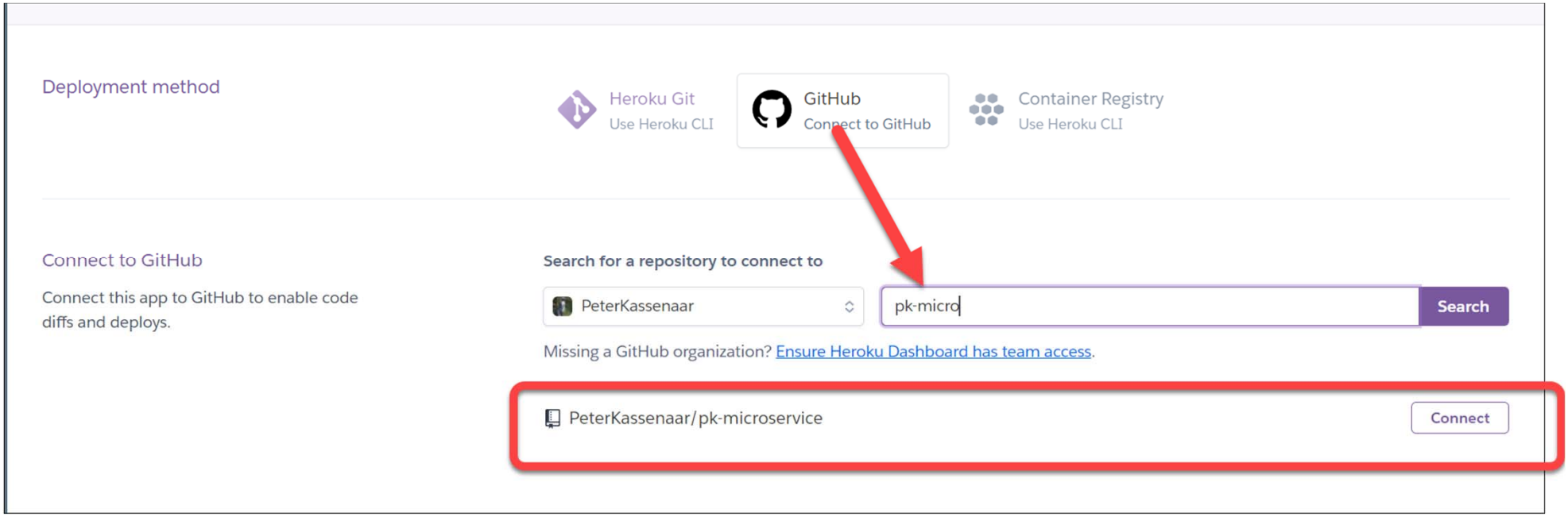
We’re using Heroku here with Github as VCS – sign up for an account!



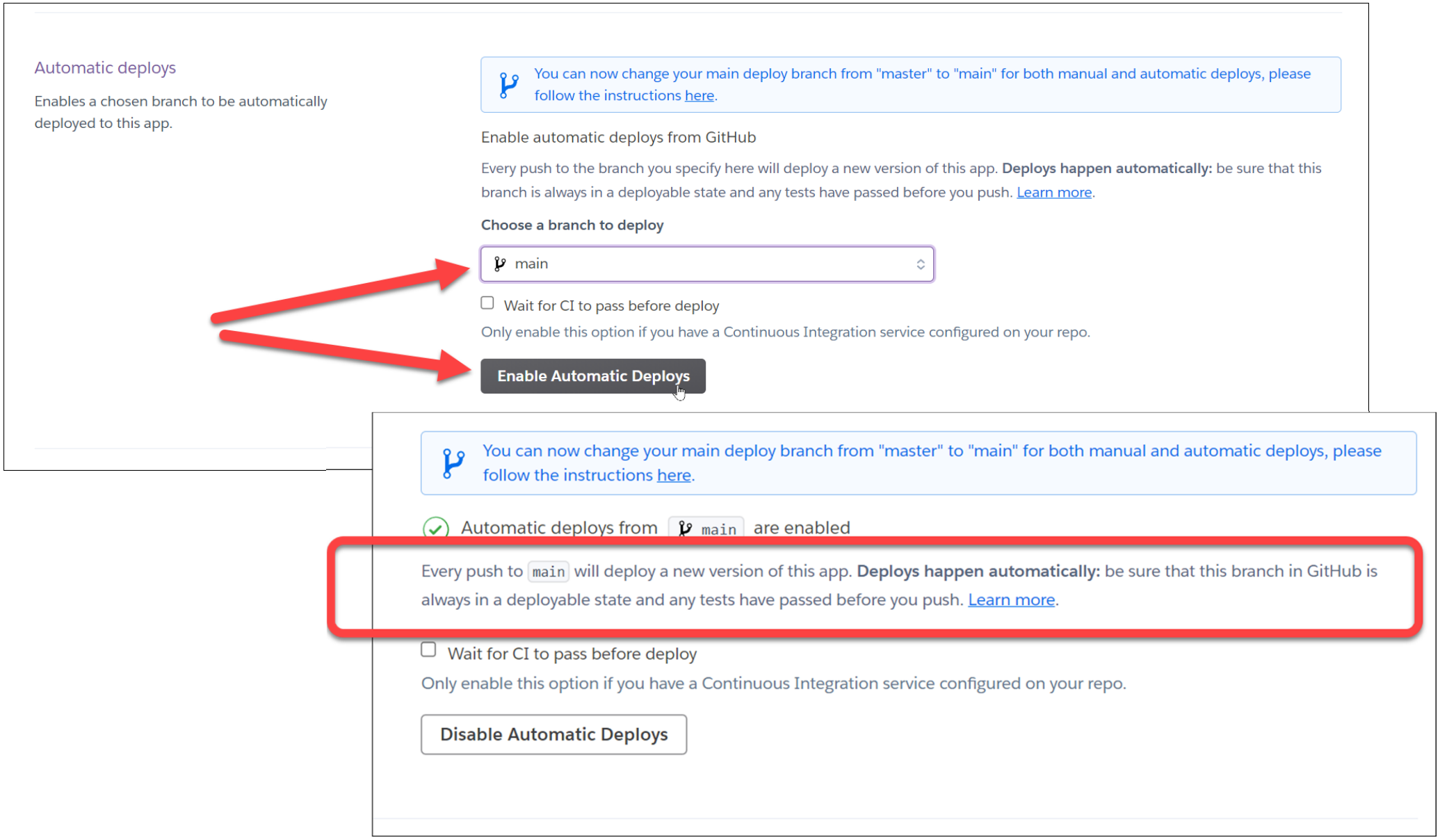
1a. Authorize external apps



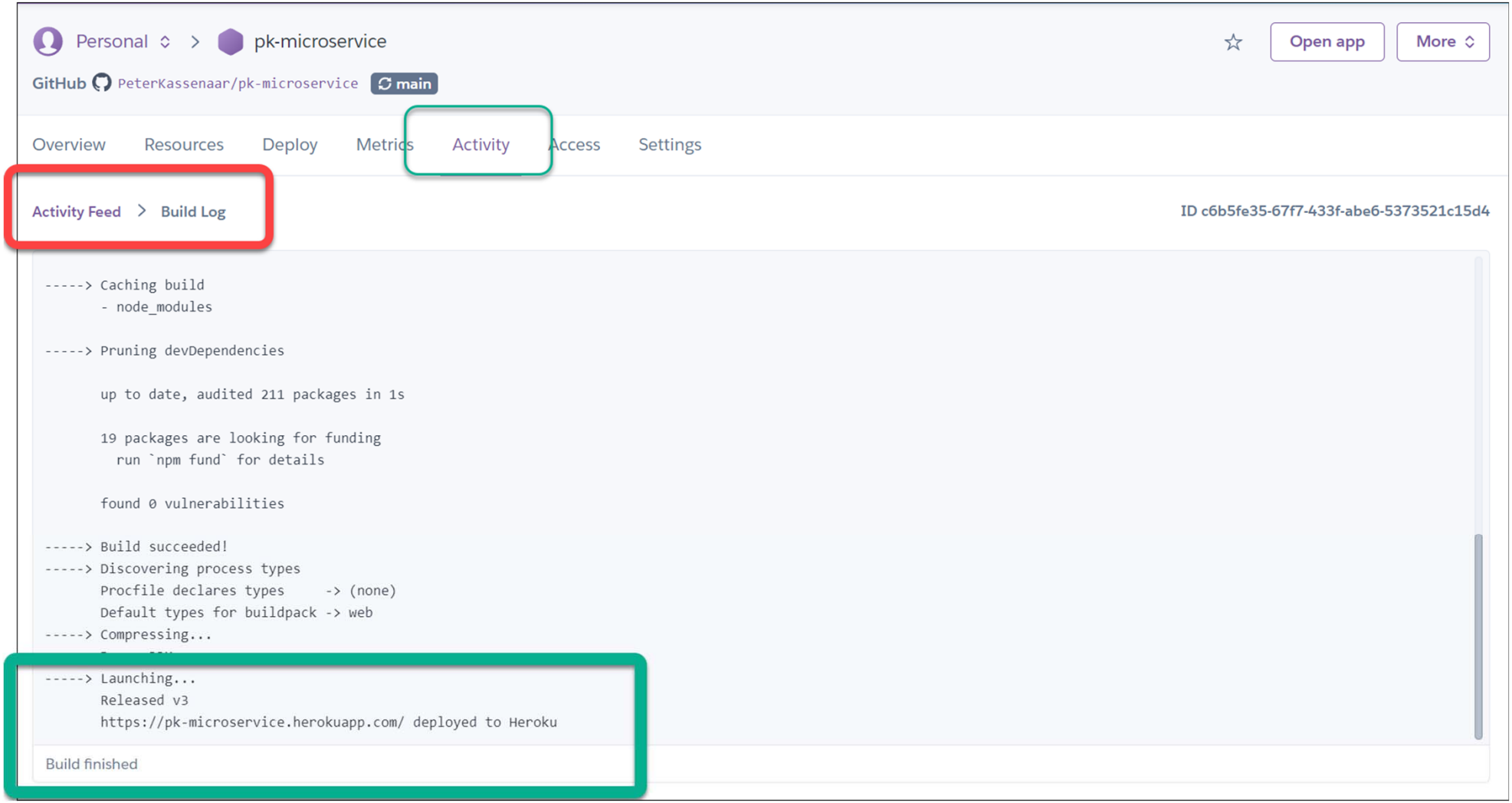
1b. Connect correct repo



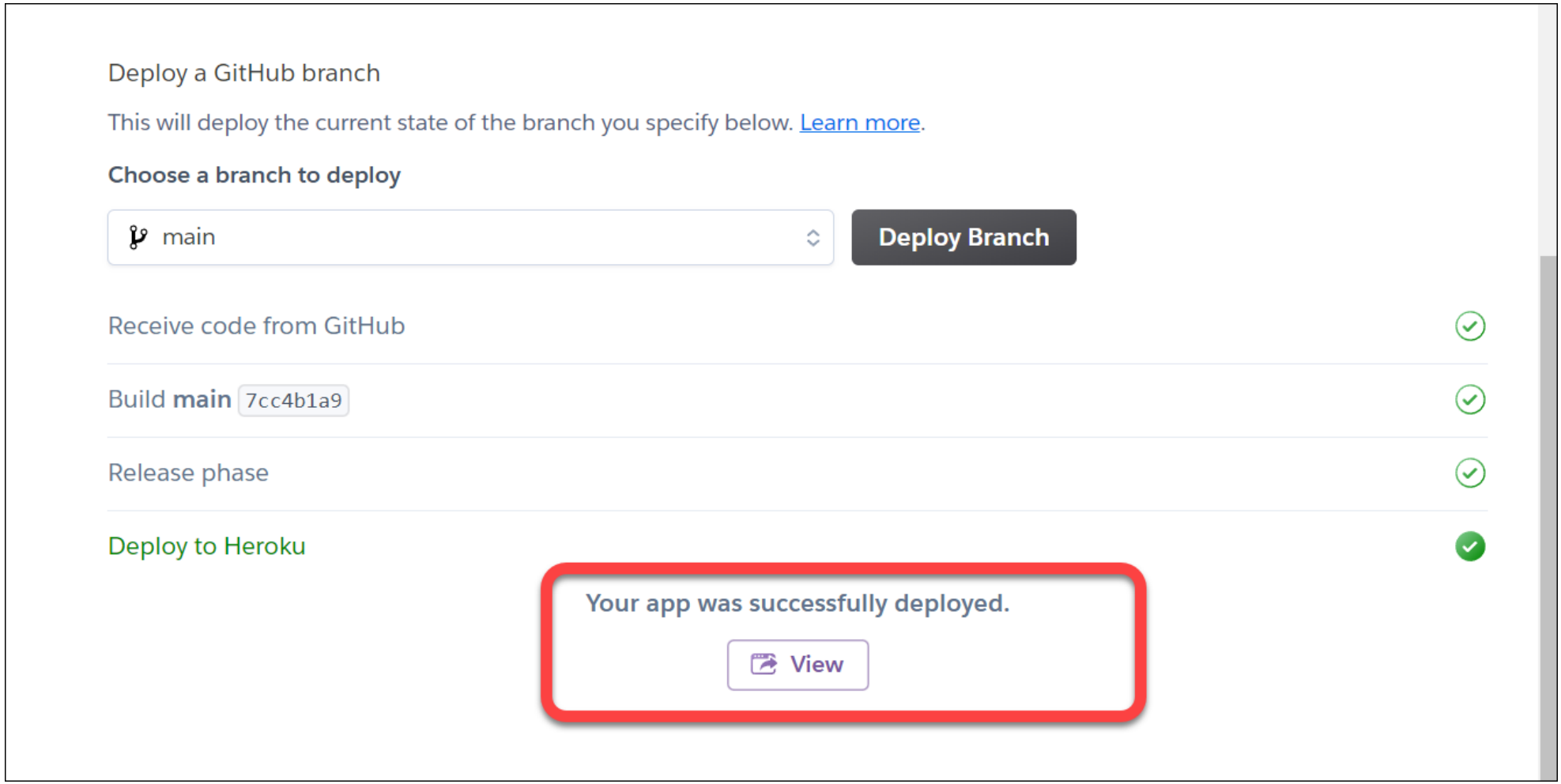
1c. Automatic deploys



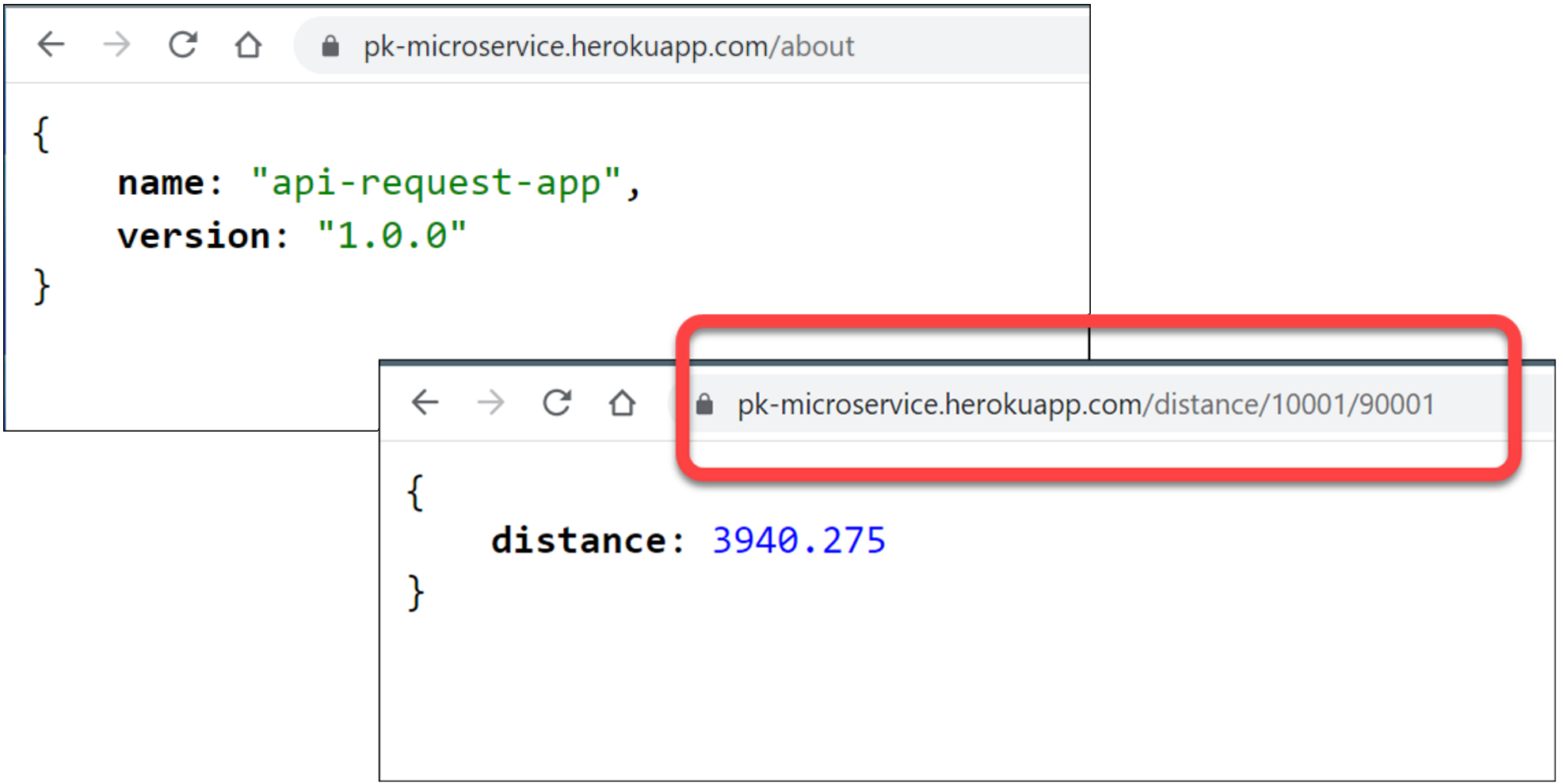
Optional – check result



1d. First Deploy

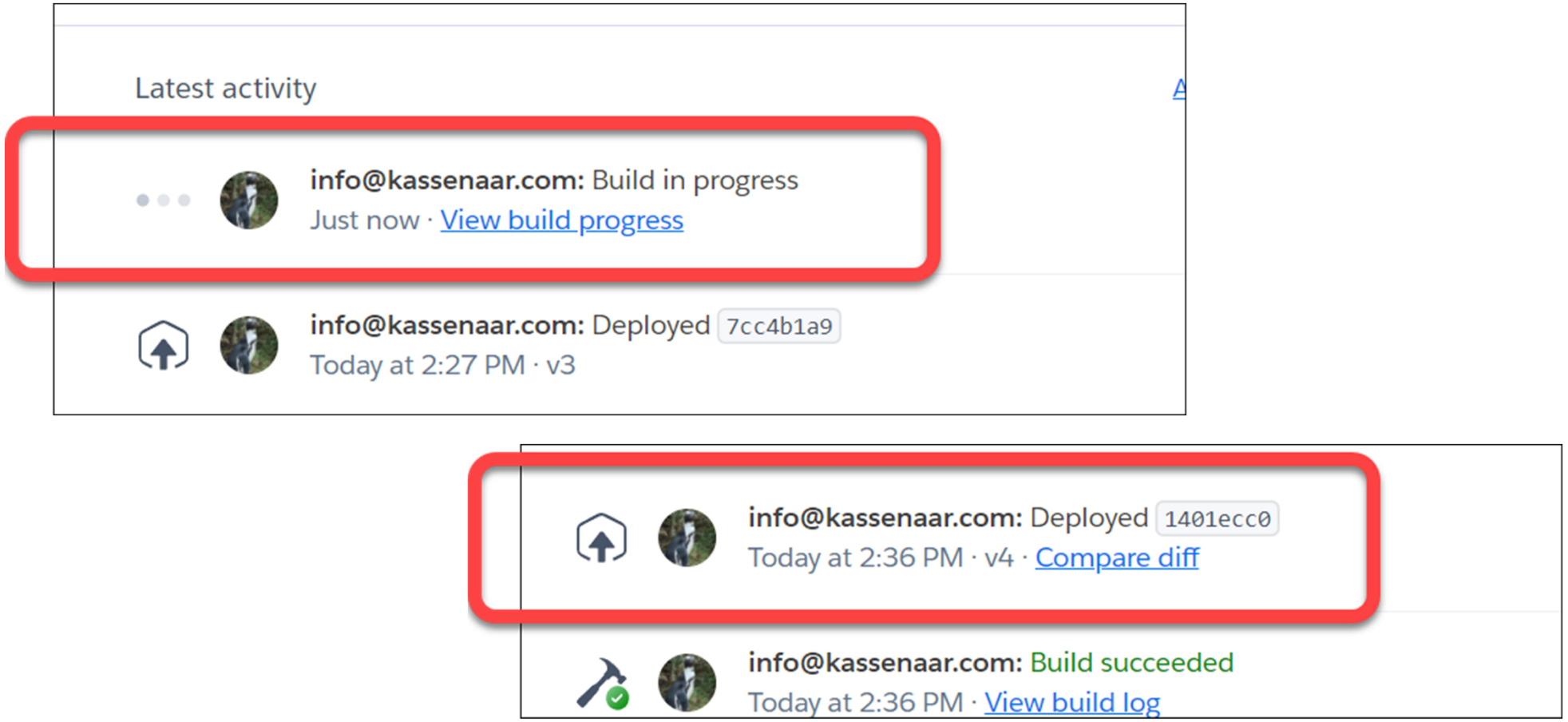


1e. Run the app



1f. Update the app – push to github

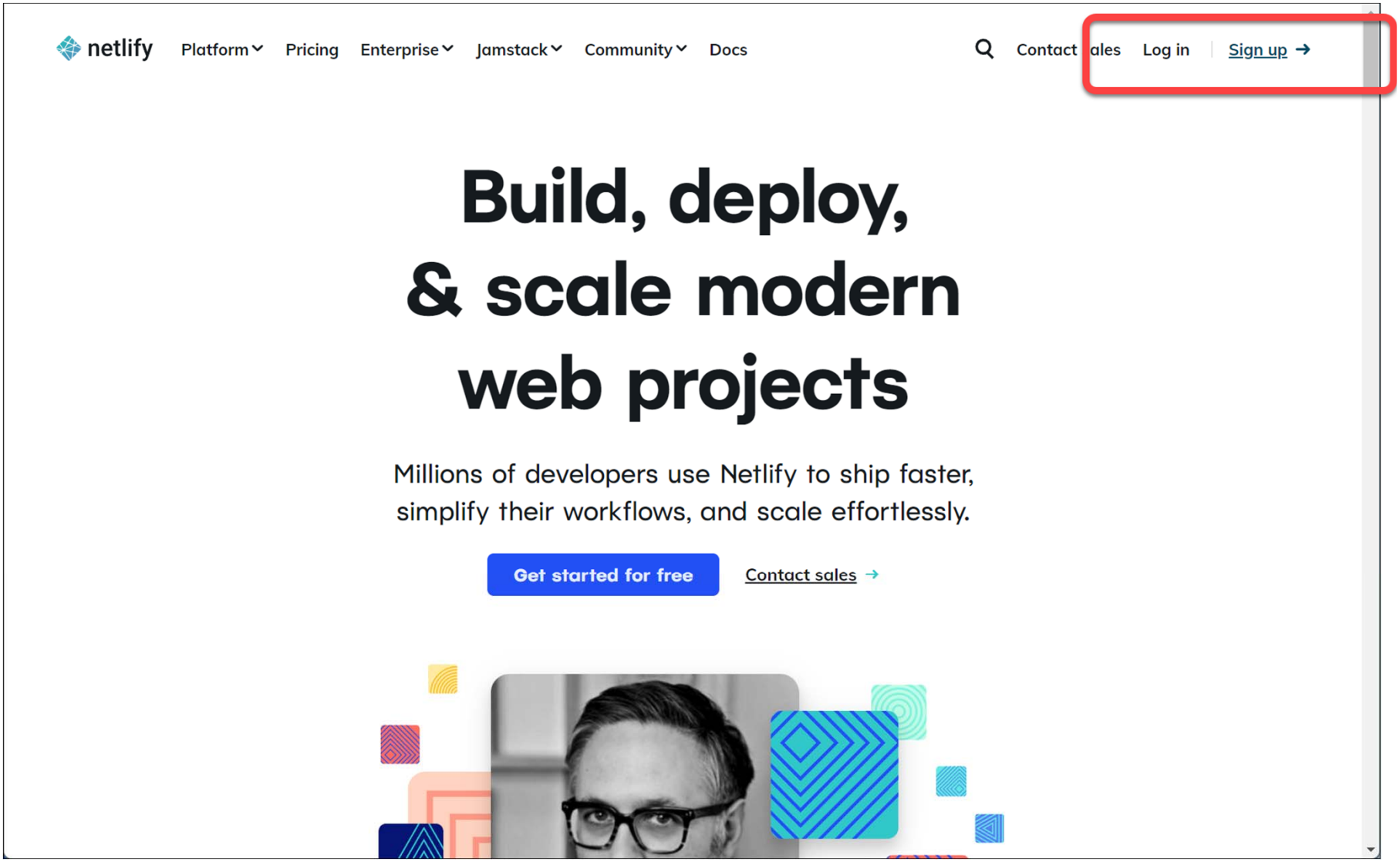
Every commit to github is picked up by Heroky and Continuously Deployed



Frontend Hosting

Update your frontend to use the CI/CD’d backend

Step 2. Frontend hosting - Netlify

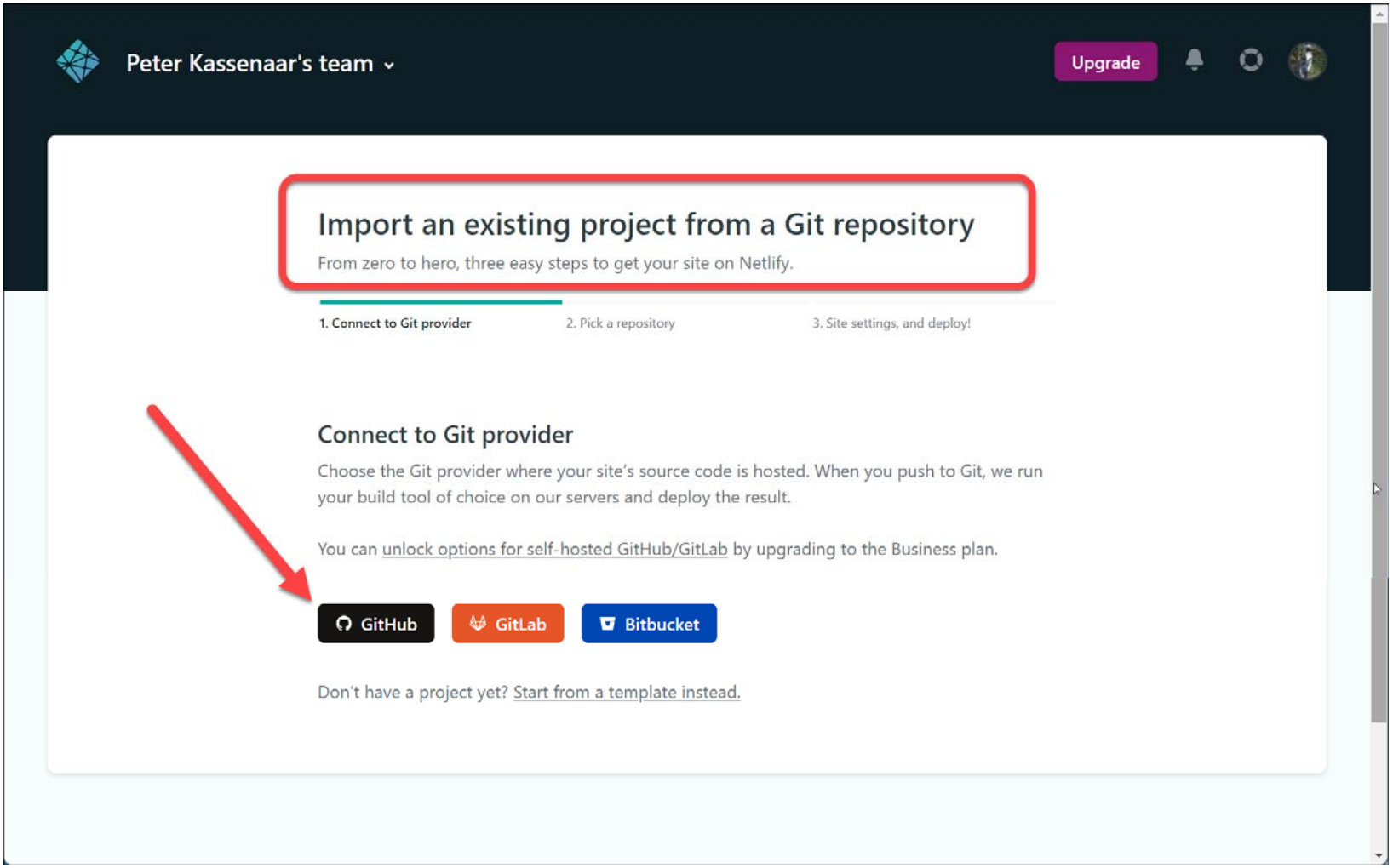


https://www.netlify.com/

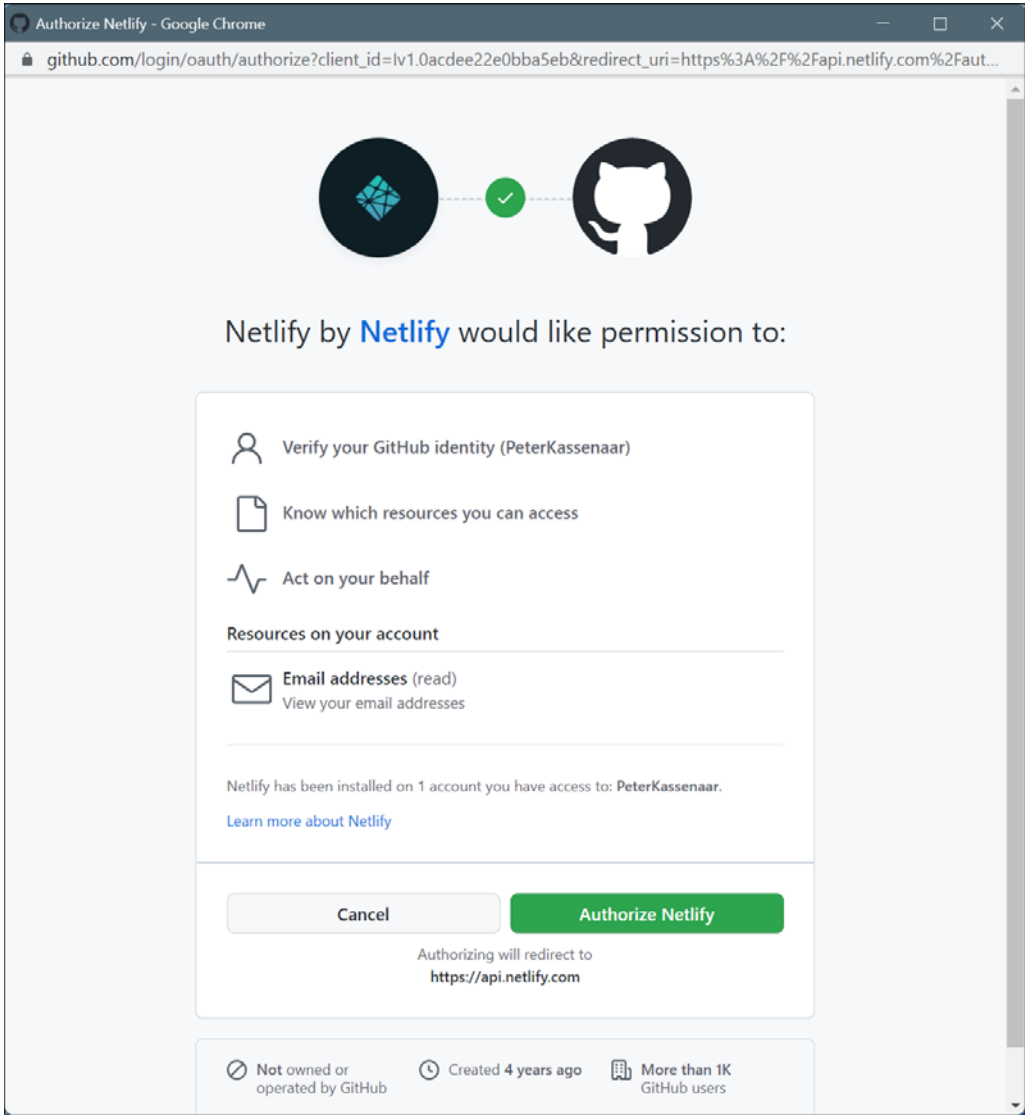
Our choice – again, connect to Github

We’re connecting to a different repo.

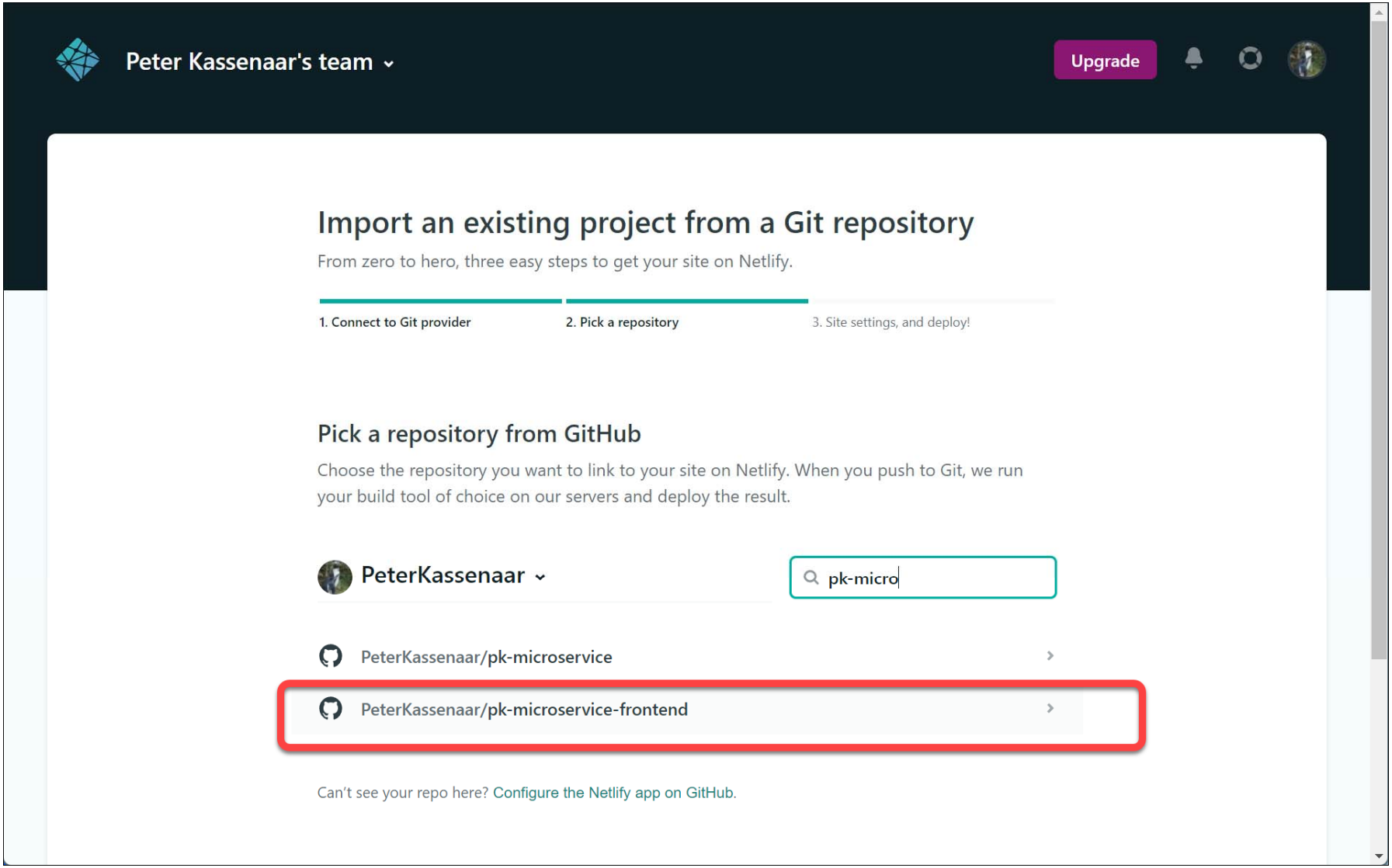
You can also connect to different branches in the same repository



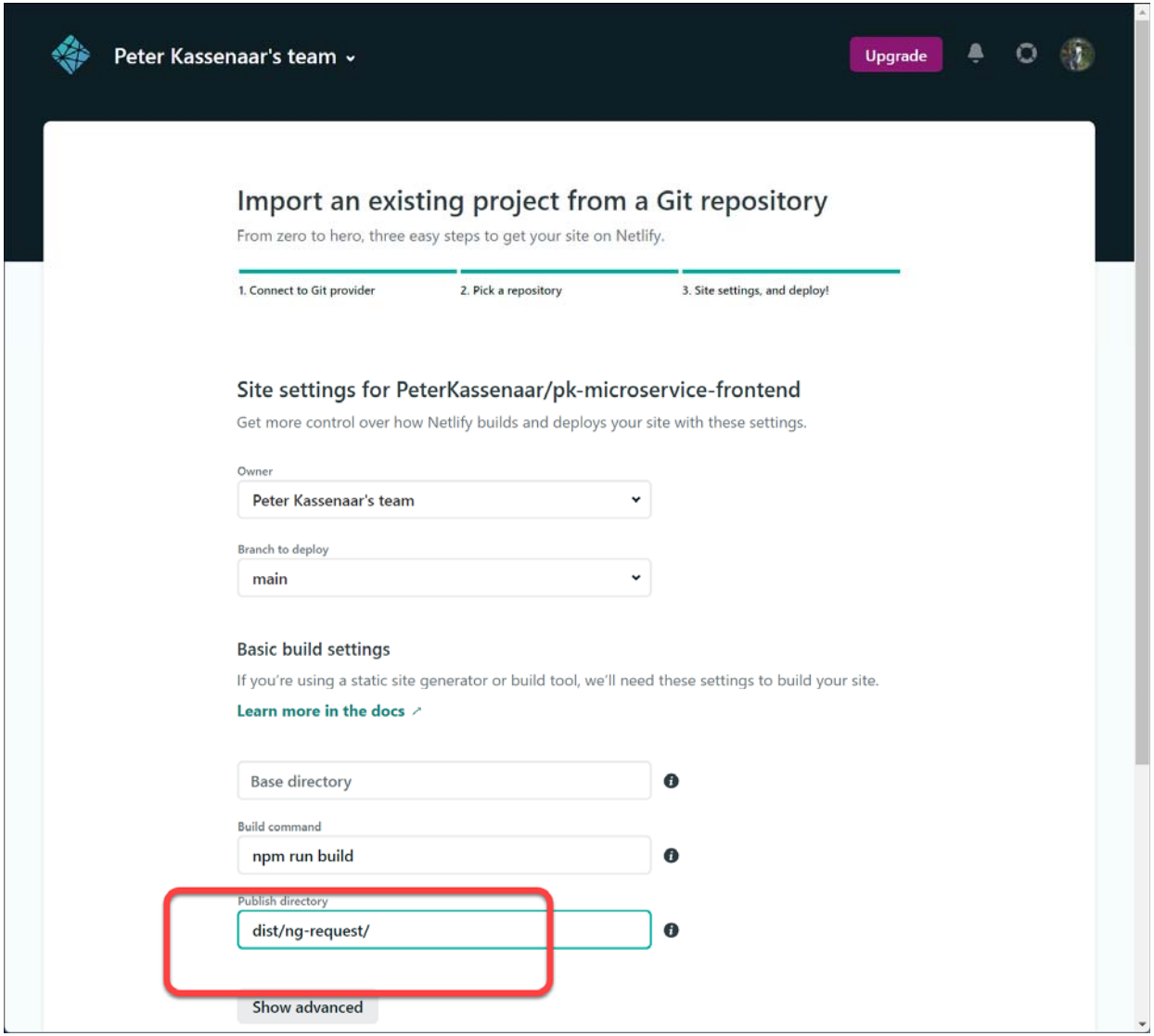
2a. Authorize platform



2b. Import from github

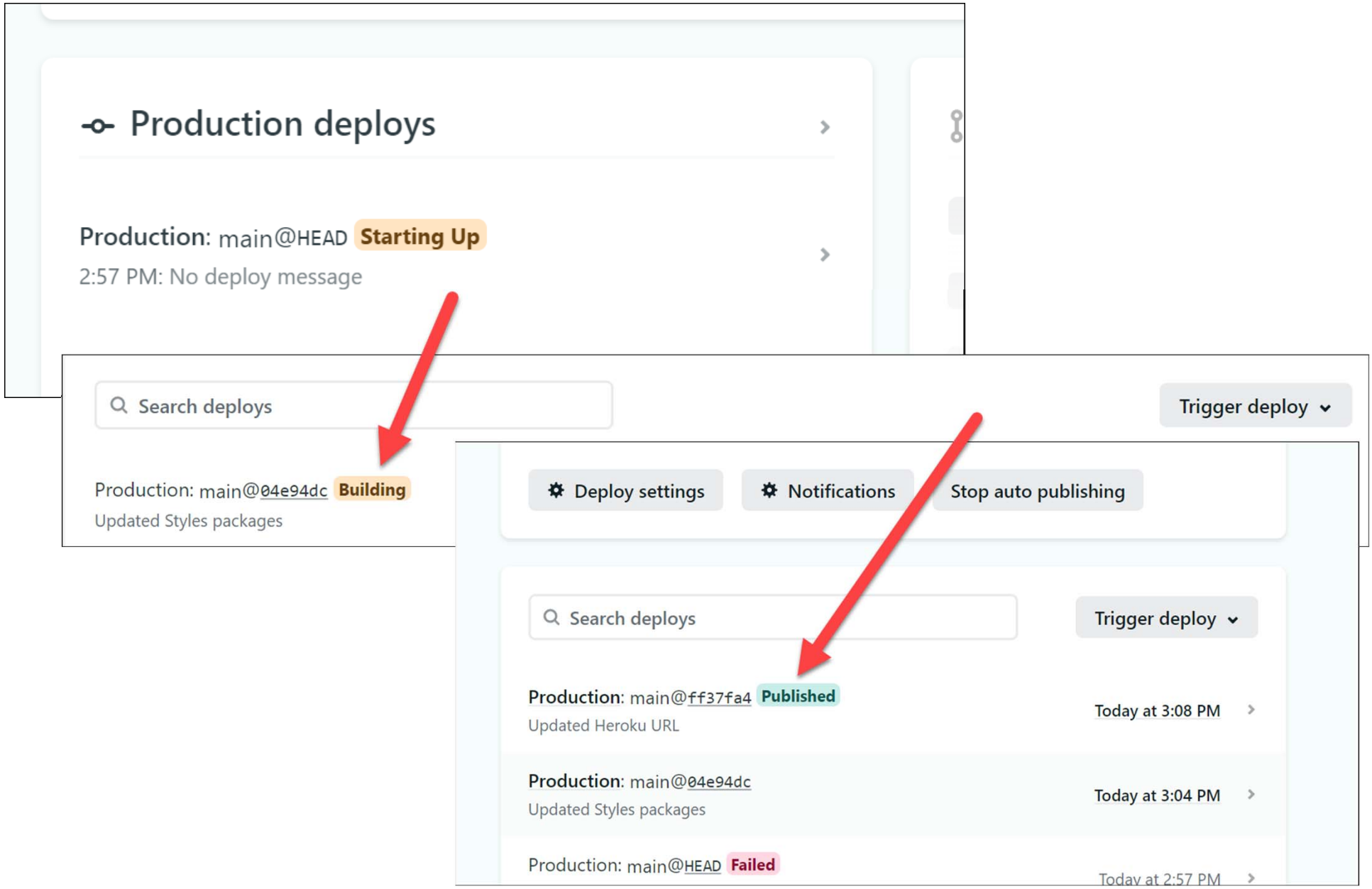


2c. Enter deployment details

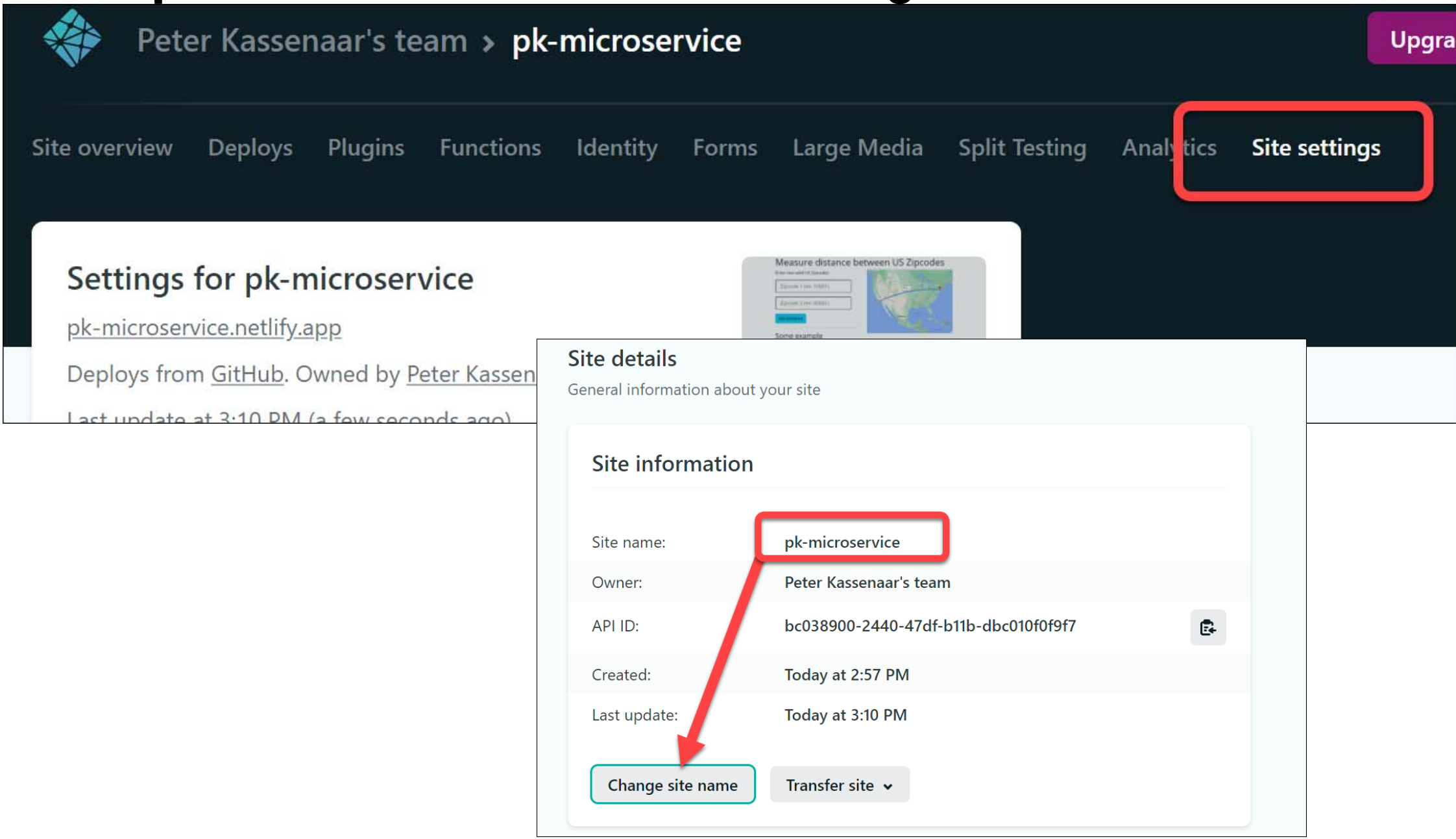


Make sure to use the correct settings – which of course depends on your technology of choice

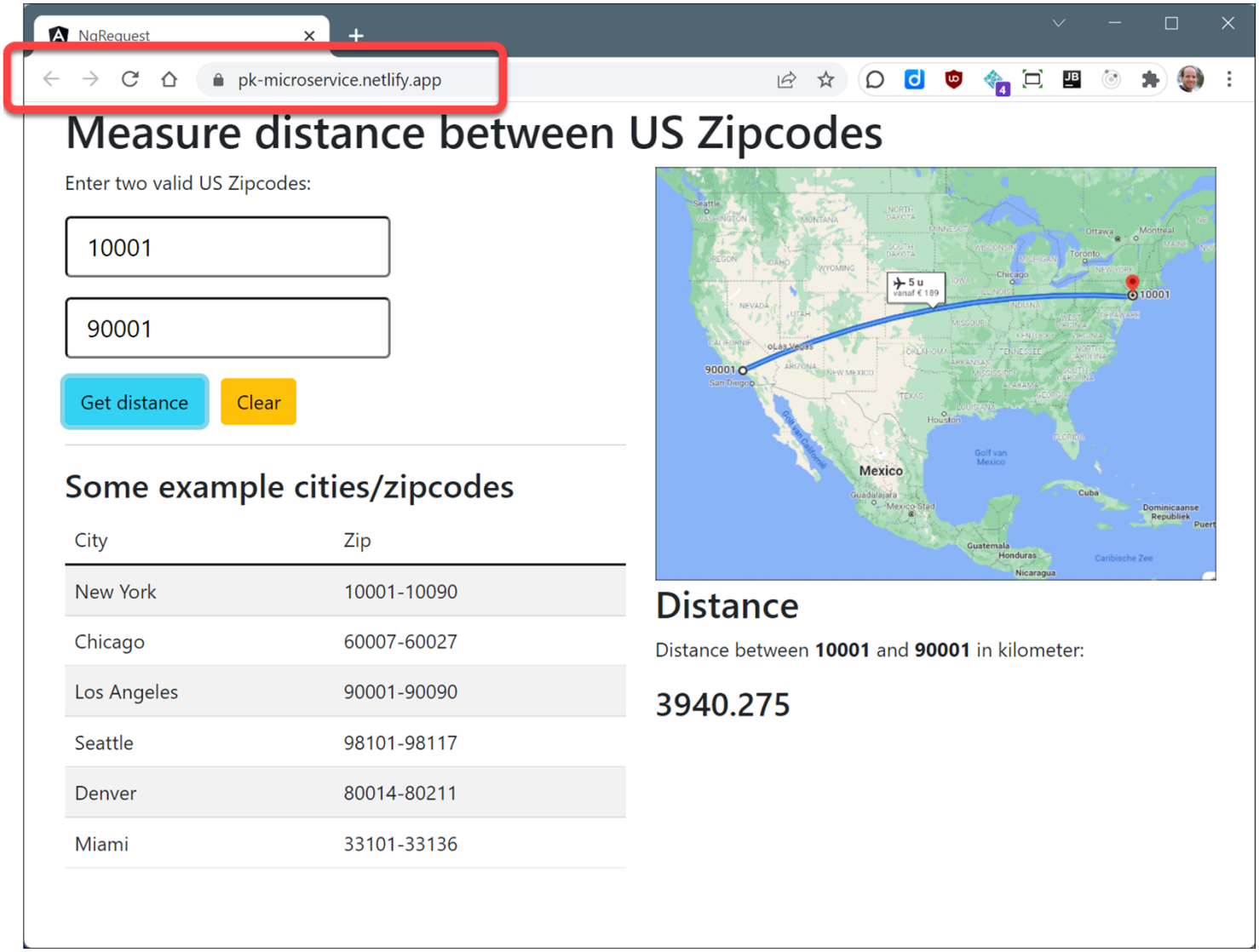
2d. Deploy


Production deploys
Production: main@HEAD Starting Up
2:57 PM: No deploy message
Search deploys
Trigger deploy
Production: main@04e94dc Building
Updated Styles packages
Deploy settings
Notifications
Stop auto publishing
Search deploys
Trigger deploy
Production: main@ff37fa4 Published
Updated Heroku URL
Today at 3:08 PM
Production: main@04e94dc
Updated Styles packages
Today at 3:04 PM
Production: main@HEAD Failed
Today at 2:57 PM

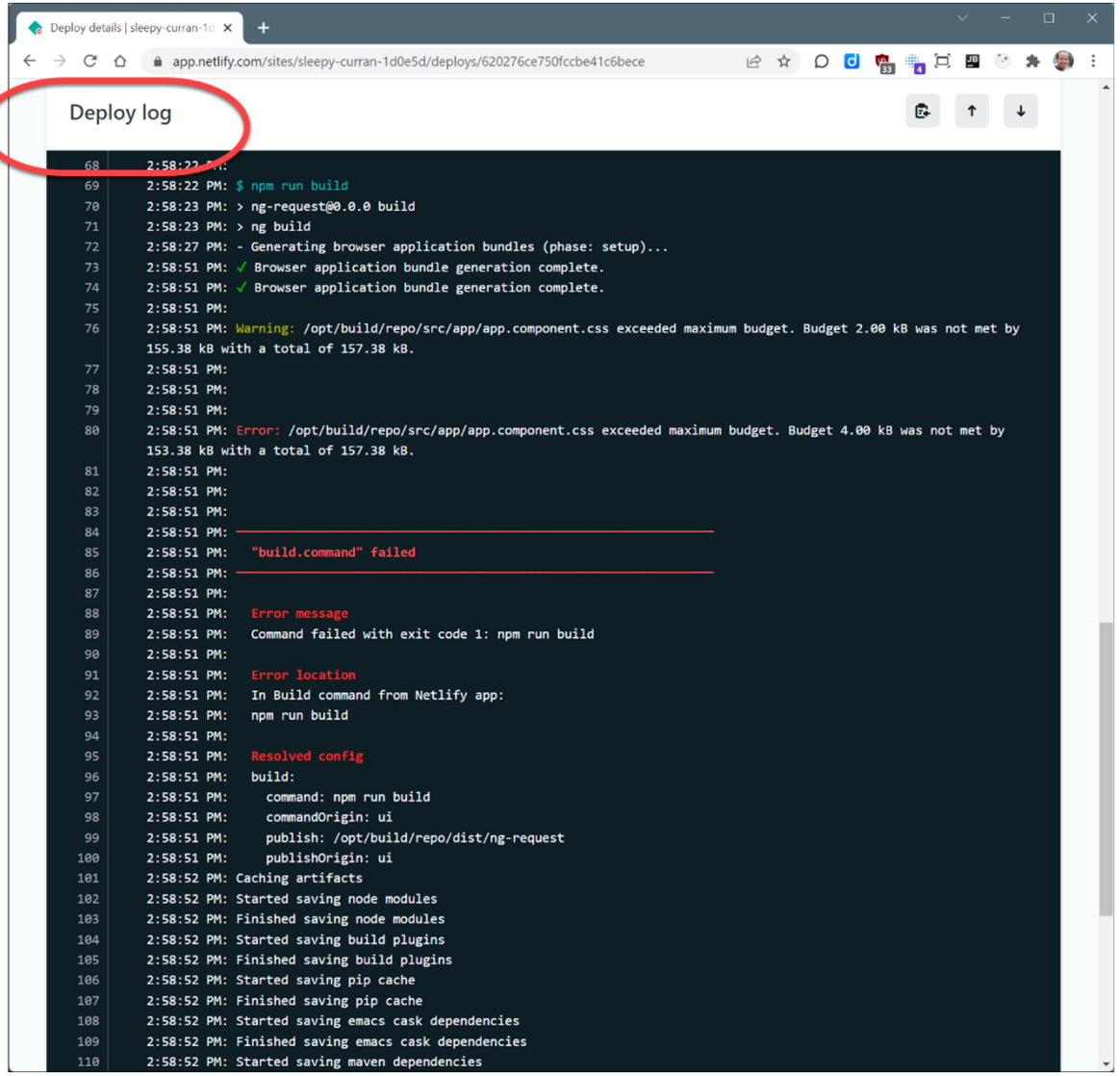
Optional – chance site settings



2e. Run your app

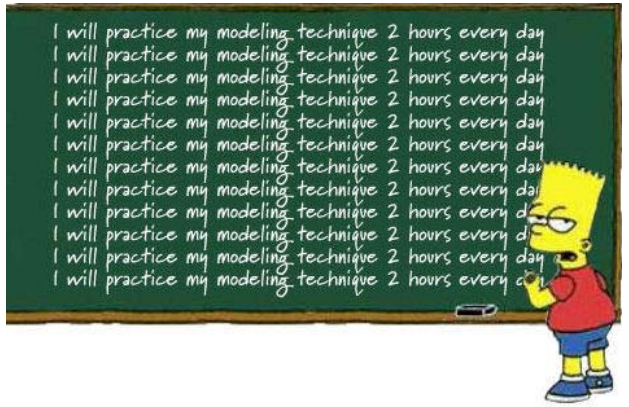


Error? Read the logs!



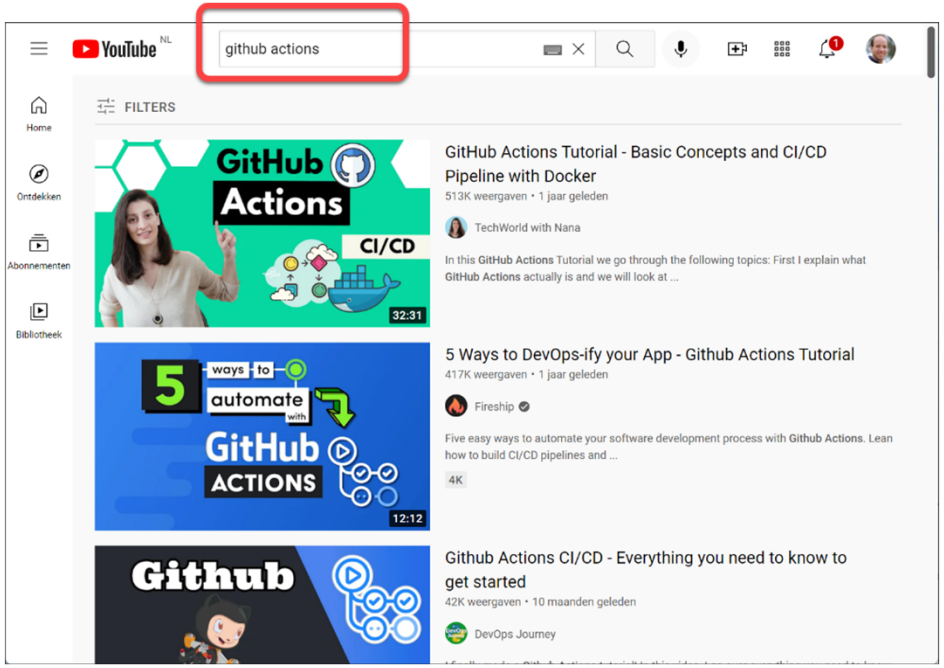
Mini workshop

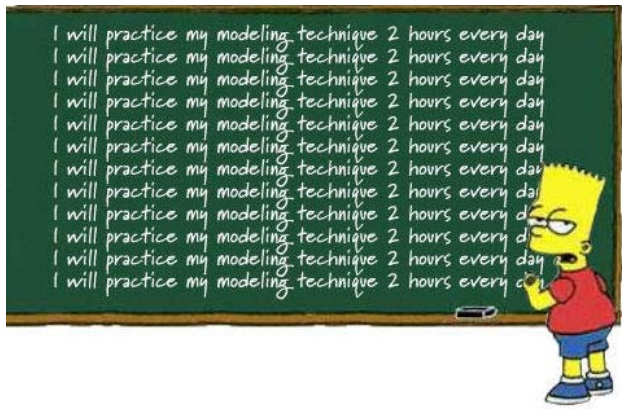
- Go ahead and publish your own application(s) to an online CI/CD provider
- You can use Heroky/Netlify if you built a web app, or choose a platform to your liking.
- Show/send the URL of your deployed app to your colleagues / or chatbox



Mini workshop

- Pick a platform of your choice
  - (for instance: Github, Azure, AWS, ...)
- Go to YouTube and find some CI/CD-videos on the matter
- OR: go to platform and read CI/CD-documentation





Questions?