Bitrise + Sonarqube

How we use it to save time and maintain a good code quality

Carebook

We'll talk about

01 - Build and distribution problematic on iOS

02 - Bitrise CI

- Workflow, Steps
- Trigger and Automation
- Use case : how we use it

03 - Sonarqube

- A static code analyzer
- The Rules and Issues
- Use case: How we use it

Intro

01

Build & distribution on iOS

Why it's ... painful

Carebook

Use case - **Publish on AppStore**

Publish an update on the **AppStore**

- Build the app with the correct **version** and **build number**
- Execute tests and UI tests, and ensure it's all green
- Sign the build with the correct **provisioning profile**
- Archive & Upload the build to AppStore Connect
- Submit to Apple for review (~ 2 days)
- Fix possible Apple's feedback
- Publish on the Store

Use case - **TestFlight Internal Tester**

You want to deploy a build on TestFlight for your QA team (TestFlight “Internal Tester”)

- Build the app with the correct **version** and **build number**
- Execute tests and UI tests, and ensure it’s all green
- Sign the build with the correct **provisioning profile**
- Upload the build to AppStore connect
- Wait for Build process on TestFlight

@ Carebook

Our case

- 4 mobile dev working on 2 Android and 2 iOS apps
  - built on 1 iOS code base and 1 Android codebase
- 2 QA who need to test completed User Stories, or do regression on release candidates

Mobile dev should be able to :

- Build and Test the app in Dev after each feature merge
- Deploy a dev build for the QA team *(multiples times during each sprint)*
- Deploy a Release Candidate Build for the QA team *(at least once each Sprint)*
- Depoy the correct Production build for the release *(once each sprint)*

It's **time consuming** and **error prone**

02

Bitrise

A flexible CI/CD solution

Bitrise

- Continuous Integration and Delivery (**CI/CD**)
- Platform as a Service (**PaaS**)
- Main focus on mobile app development (so a lot of mobile oriented tools)
- Easy to use webapp, to manage app, builds and work in team

Bitrise - **Get Started with iOS**

To setup a Bitrise Application:

- Link your iOS app repository
- Upload your Provisioning Profiles and Certificates

Then you can start creating **Workflows** usings **Steps**.

When a Build is triggered, Bitrise assigns a Virtual Machine to execute your **Workflow**.

The Virtual Machine is destroyed after the build finish.

The only things left are the generated log files and other artefacts.

Workflow & Steps

Bitrise - Workflow & **Steps**

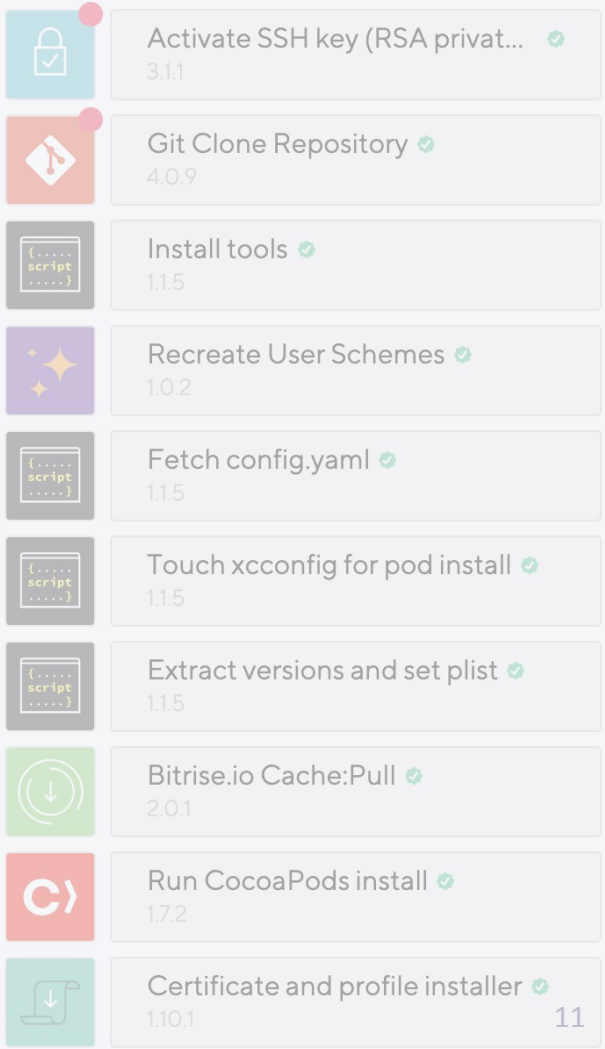
Steps are open source command line scripts based programs that can be executed by the **Bitrise CLI**

---> it's a **build task**

It can be simple or complex useful tools, like for example :

- Execute unit or UI tests
- Send a **Slack** message
- Create an **Xcode Archive**
- **Publish** build to **TestFlight** ... !

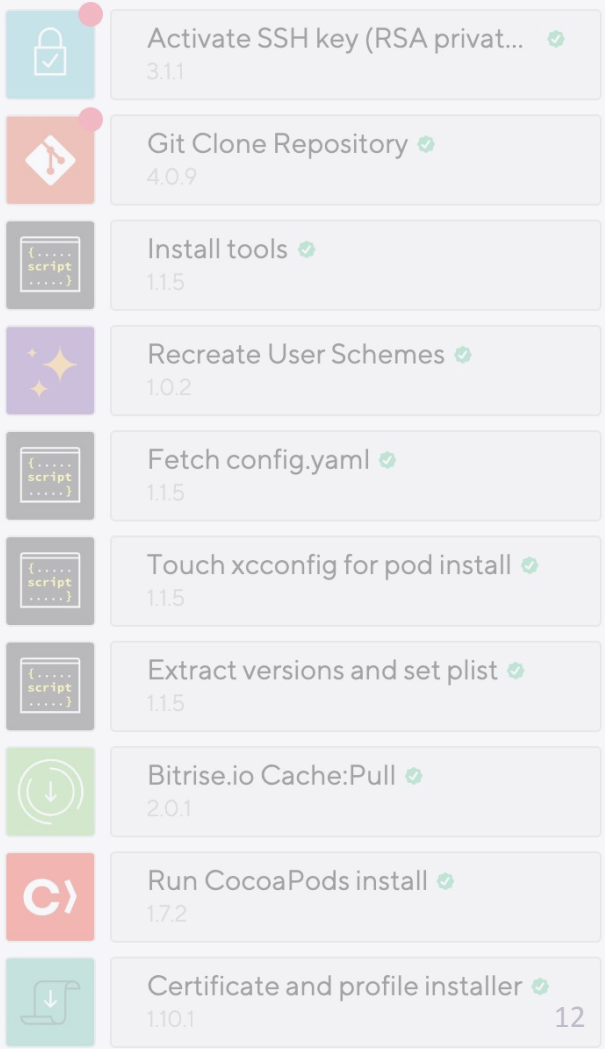
Steps lives in a **Workflow**



Bitrise – Workflow & **Steps**

Steps can

- Get **input variables** from the previous step,
- Return **output variable** to the next step
- Deal with **Workflow environment variables**
- Deal with **App** (in a bitrise context) **environment variables**
- Deal with **Secrets** (encrypted environment variables that you can chose to hide)



Bitrise – **Workflow** & Steps

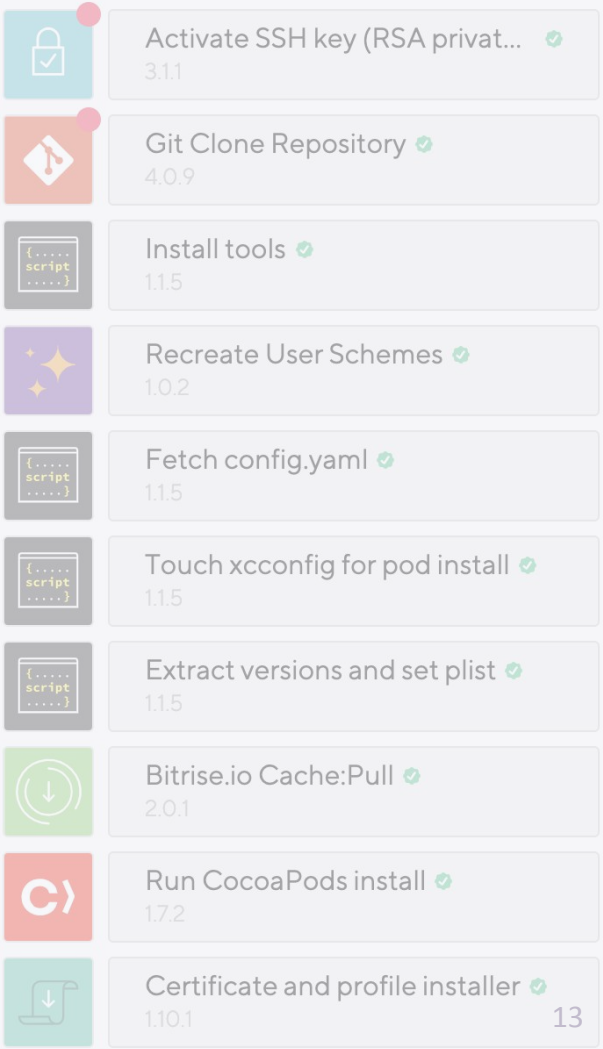
A **Workflow** is a set of **Steps** … and/or **Workflow**

You can create Workflow based on an existing Workflow

Expose environment variables

Triggerable:

- On specific events
- Manually
- Periodically



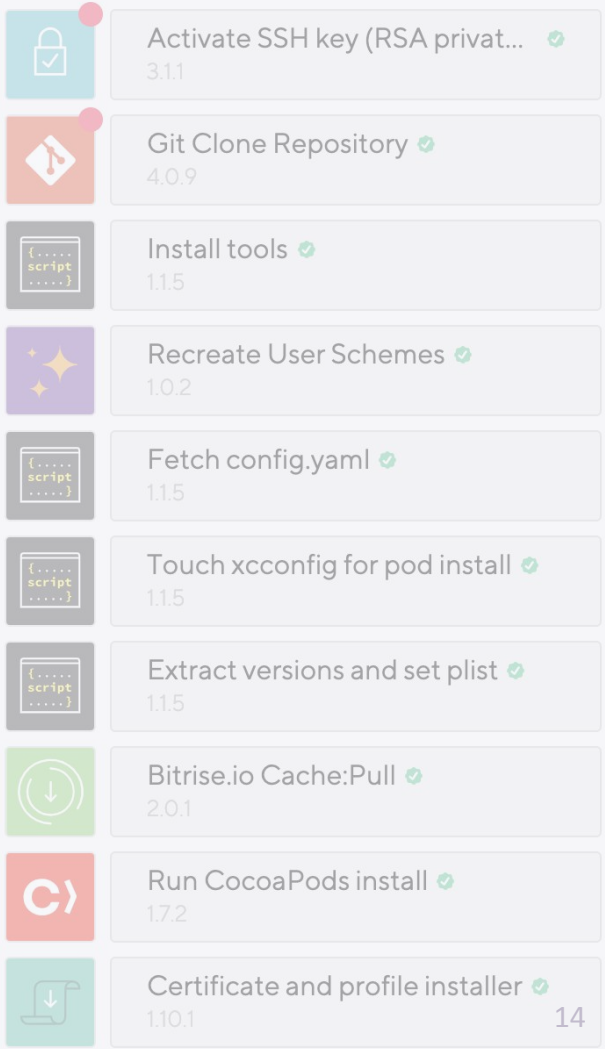
Bitrise – **Triggers**

Triggers are the bitrise representation of the **webhooks** you register on your Bitrise App.

You can then map **Trigger** to **Workflows**

For example, with a BitBucket webhook :

- **Push** event on **Master branch** will start your "deploy_to_store" **Workflow**
- **Tag** event on **Dev branch** will start your "release_candidate" **Workflow**

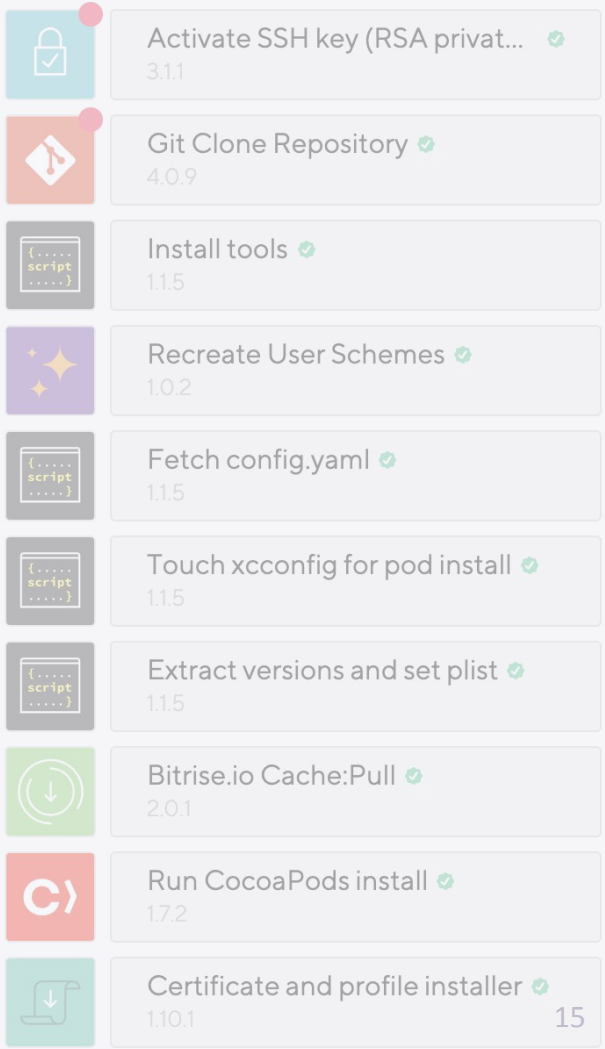


Bitrise – **Triggers**

Triggers are not the only way to start Workflows.

You can start a workflow :

- manually from the web app
- with an **API call**
- by scheduling it

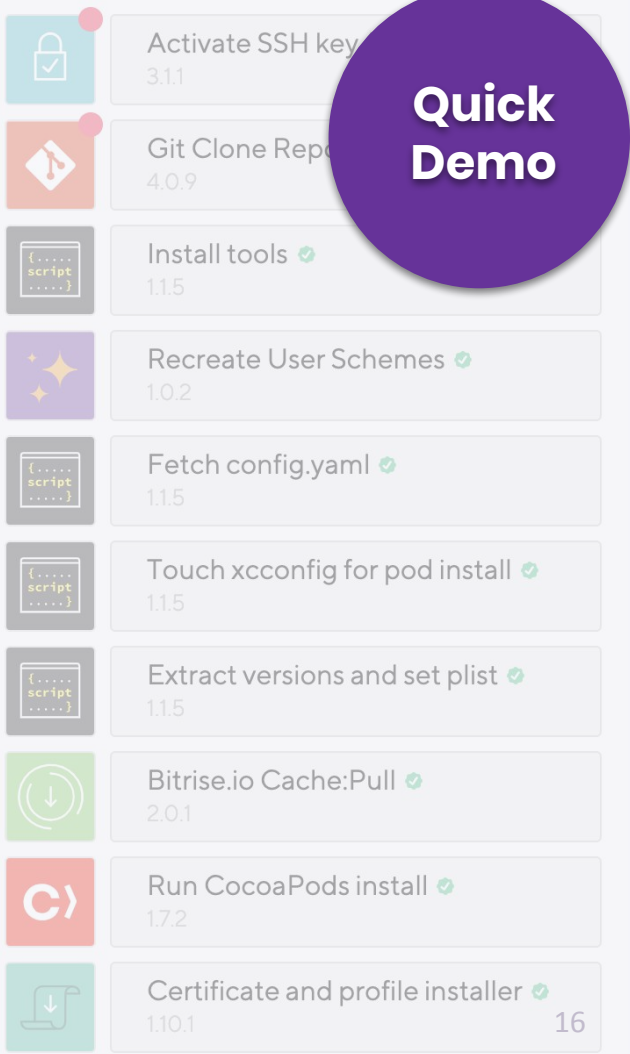


Build Workflow & Steps

Use graphic interface Demo

- Create Workflow
- Create Steps
- Environment Variables, Secrets
- Triggers

All your workflows steps and triggers will be declared in one file, the **bitrise.yaml**



Examples

- Automatically or manually deploy to TestFlight
- Run unit & UI Tests
- Run code analysis tools **#Sonarqube**

Carebook Context – iOS

- **1** Code base
- **2** Flavored Apps
- **1** tools submodule to deal with
  - Translation
  - Generate code
  - Apply configuration and settings for a given app flavor and version
- Each app is **split in 3 apps**, matching build configurations (Dev, QA, Production)

Carebook Context

Some solutions we created using Bitrise

- Automatic **sync of Dev branch**, with our external translation tool (OneSky)
- Automatic **nightly Dev build** deployed to TestFlight
- Automatic **Build + Tests** on every merge in Dev
- Automatic **Build + Tests** on every Pull Request status appears in Bitbucket

- Ability to trigger manually a **Dev build deployment** from the dev machine
- Ability to trigger manually a **QA build deployment** from the dev machine
- Ability to trigger manually a **Production** build deployment from the dev machine

Automated builds - **Copy Sync**

Nightly build to sync our Translation from our external tools

- Get new translations from our external tool (OneSky)
- Apply change generate updates Strings file
- Execute tests
- Commit + push to Dev
- Report error/success on Slack

Every morning the Dev branch is up to date with the latest copy values.

Automated builds - **Dev build for daily QA**

Nightly builds of the current Dev branch

- Configure the app (apply the given flavor, generate configs, build numbers)
- Build and Execute tests
- Create an Archive
- Upload to TestFlight
- Report error/success on slack

Every morning, the latest Dev version of the app is available for all our TestFlight **internal testers.**

Manual builds - **Release Candidate / Production**

Locally execute a Fastlane lane to trigger the sonarqube **Workflow** on Bitrise, with the appropriate parameters.

Fastlane lane wrap the logic to **manually trigger** a build :

- Check if the current branch/tag is correct for that type of build
- Map some configuration param (Flavor etc)
- Figure the workflow ID to trigger
- Build a JSON with all configuration params
- Call the **Bitrise API** with this params to start the build

sonarqube

02

Sonarqube

Code analysis

Carebook

Sonarqube - **Analyzer Rules**

Static **source code** analysis tool. Open source but with options

Support 25+ languages : *Swift, Objective-C, Java, Kotlin, C/C++, C#, TypeScript, Python, Ruby, HTML, PHP, XML ….*

Support Multi Language Project

Analyzers execute **Rules**, regrouped into **Quality Profiles** to generate **issues.**

Sonarqube - **Issues Types**

Classify discovered issues into different **Types**

- **Code Smell**
- **Bug**
- **Vulnerability**
- **Security Hotspot**

Sonarqube - **Issues Severity**

Quick Demo

Then Sonarqube assign a severity to issues. (*More info here*)

Severity

- Blocker
- Critical
- Major
- Minor
- Info

How we use it

- Run it manually on your branch
- Automatically run on **PR**

Sonarqube - **Trigger a Scan locally**

Fastlane lane triggers the Sonar Scanner Locally :

- You need to build your app and execute tests before
- Sonar Scanner analyse the (***test reports***)
- Parse source code applying profile rules
- Generate a report
- Upload this report to the platform to compute the results

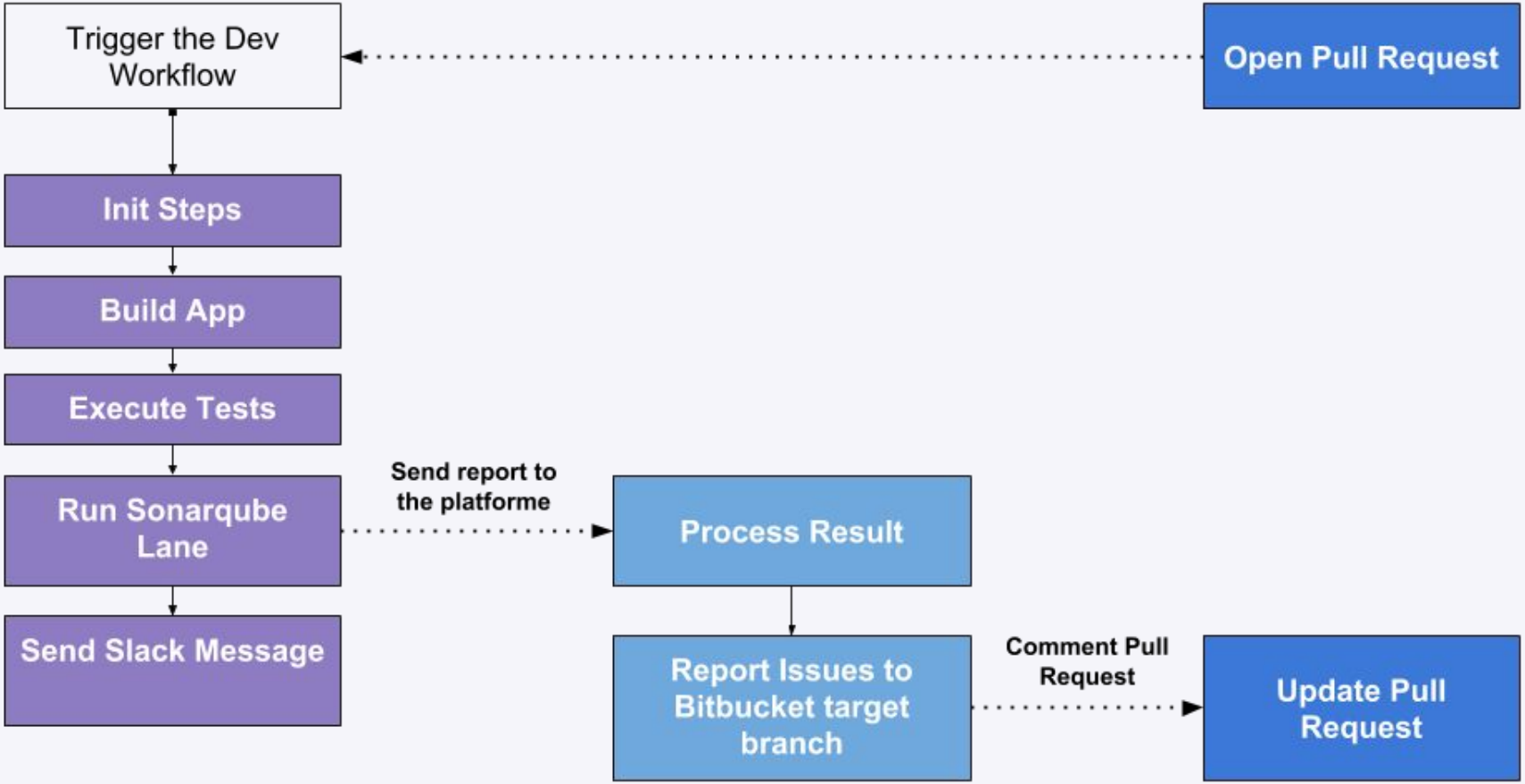
Sonarqube - **Trigger a Scan on PR**

Link Bitbucket and Sonarqube!

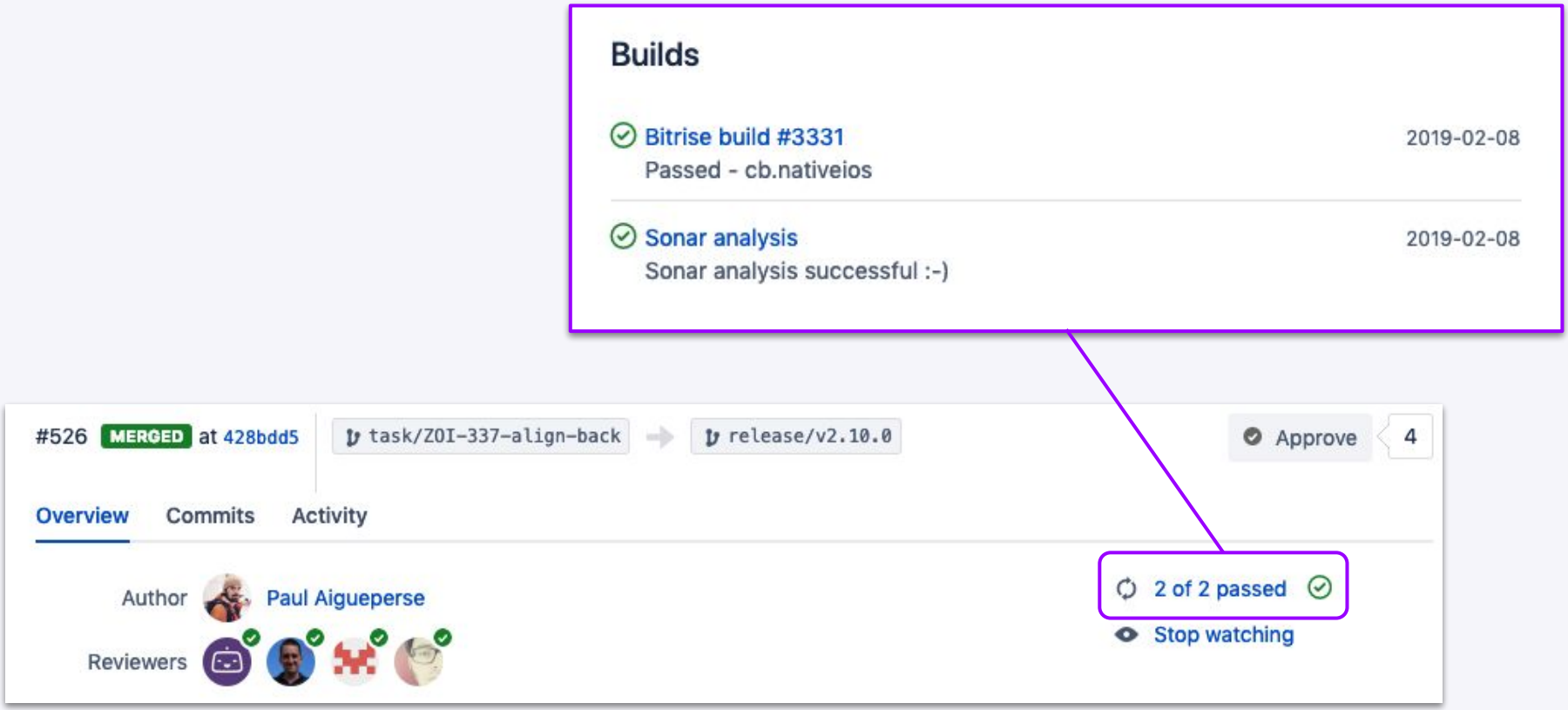
The same **Fastlane** lane is triggered in a **Bitrise** Workflow

- Build and test process are done in Bitrise
- Reports are uploaded to the Sonarqube platform
- Sonarqube report issue to Bitbucket target branch

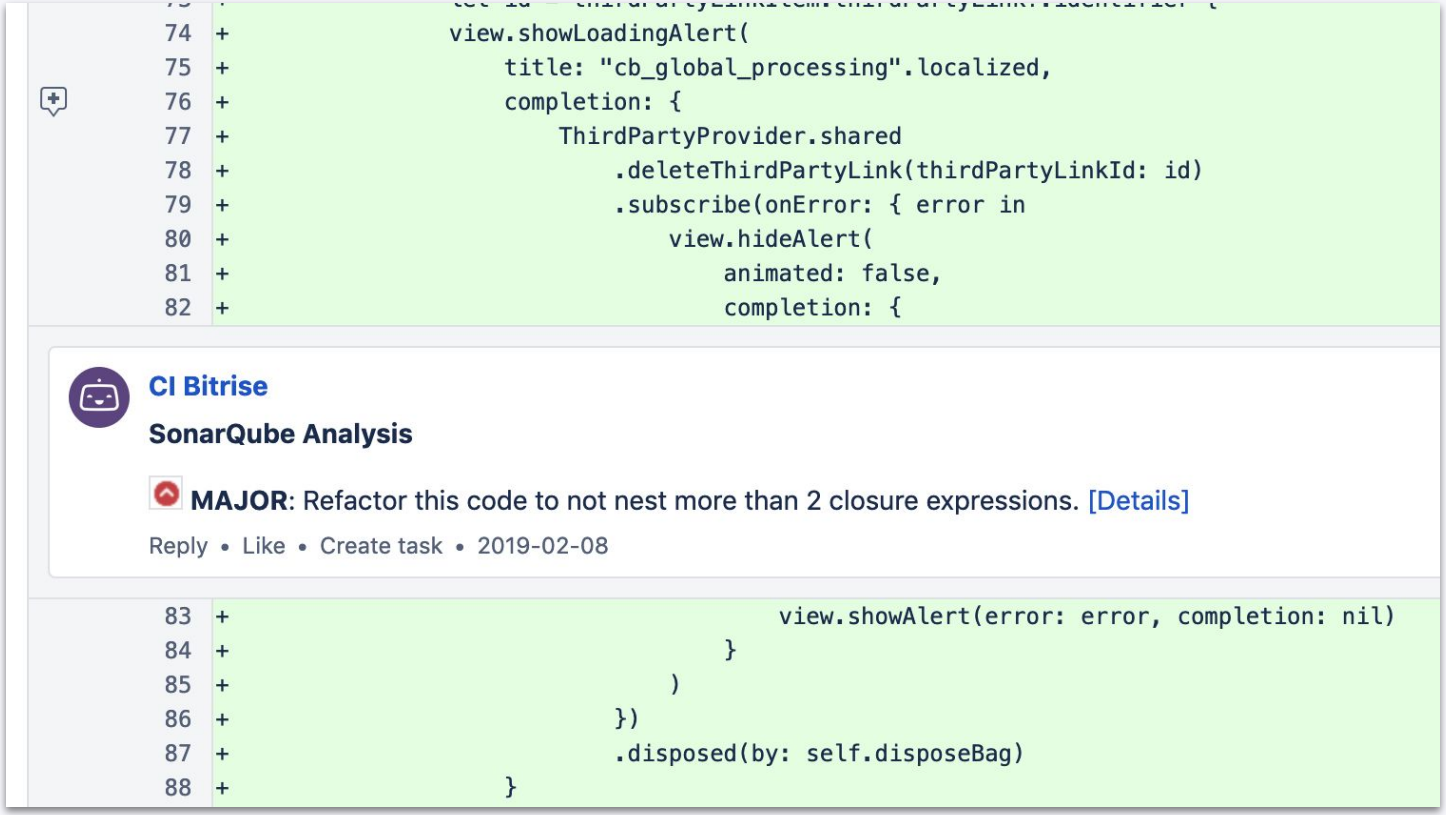
bitrise sonarqube Bitbucket



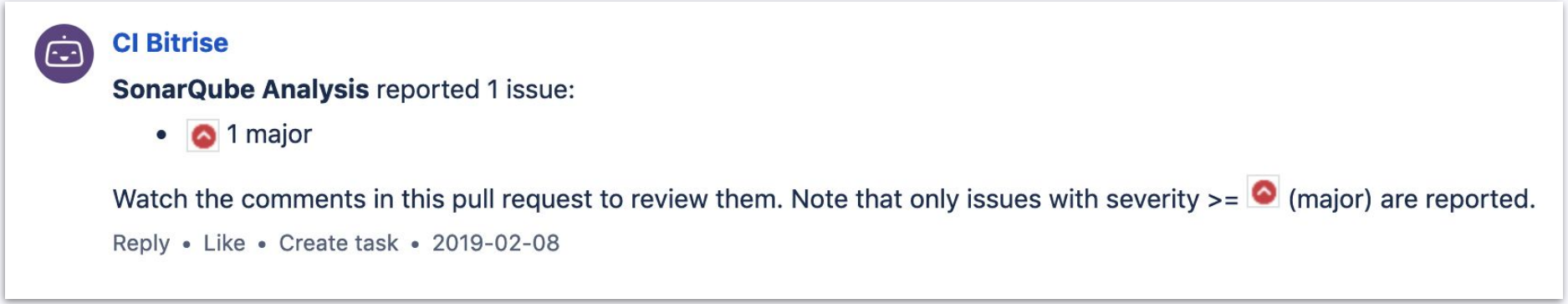
Sonarqube - **Trigger a Scan on PR**



Sonarqube - **Trigger a Scan on PR**



Sonarqube - **Trigger a Scan on PR**



Conclusion

- **Bitrise** is very flexible and scalable, and make it easy to
  - Design workflow for all your use cases（production build, QA build, dev build, test build）
  - Maintain your code quality over builds with tests and tools
  - Deploy build automatically
- **Sonarqube** is a powerful static code analyzer, it helps you
  - evaluate your code health with Quality Gate
  - highlight important issue even in your Pull Request

Questions ?

Merci