

# CD with React, Firebase and Cloud Build

## 1. What are we going to do?

- Create a simple React app and deploy it to Firebase Hosting
- Create a CI/CD Pipeline using GCP Cloud Build Service

## 2. Things to know first?

- Having some basic React knowledge
- [What is CI/CD Pipeline](#)
- [What is Git](#)
- [What is Firebase](#)

In this section, I want to show you how to deploy your React application to Firebase. It **should also work** for any JavaScript framework such as Vue or Angular. And you are not limited only in JavaScript framework, you can also have some static HTML/CSS files and it will also work.

## 3. Create React App

Run the command in the terminal:

```
npx create-react-app samplefirebasehosting
```

This will create a react app. Open the folder with `cd` and run `npm start` to start running the app locally.

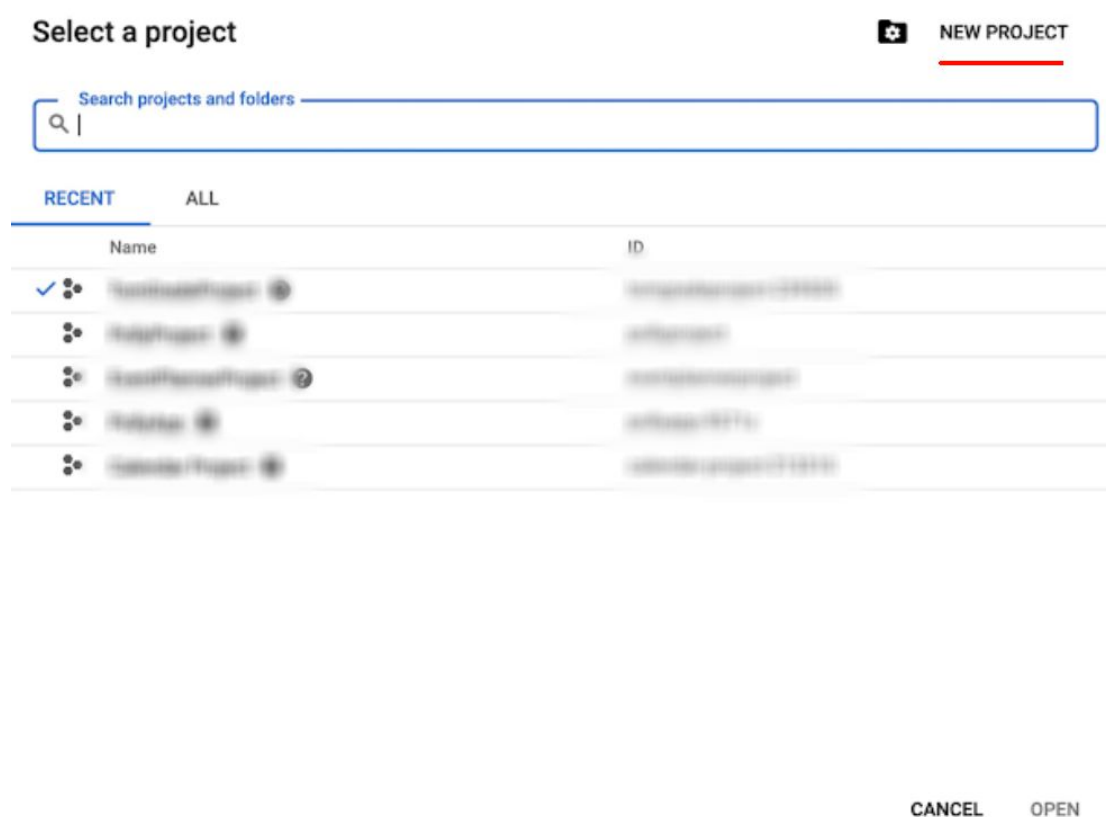
## 4. Setting up Firebase Hosting to our React App:

Before we start we have to install some tools for our local development environment. First, we need to install the Firebase CLI to our global node modules:

```
npm install -g firebase-tools
```

## 5. Setting up Google Cloud Project

After installing the Firebase CLI we need to create a GCP Project in the <https://console.cloud.google.com>. Just click the dropdown in the upper left and you can then create a new GCP Project.



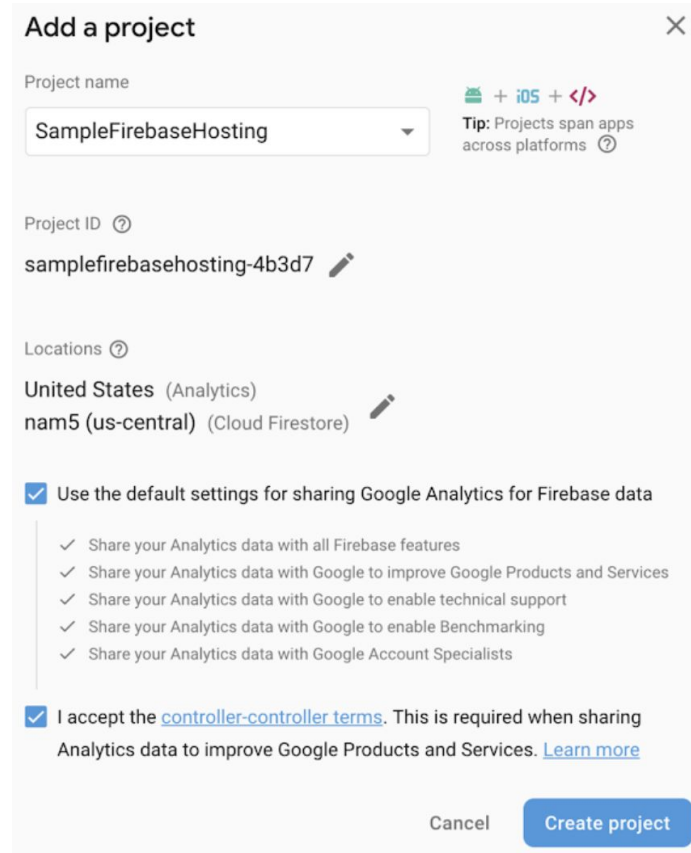
After creating the GCP Project we need to enable billing to your Project so that we can use the GCP Cloud Build Service. GCP has a program in which they will give you free \$300 credit good for a year.

*! For DPS participants we already created the projects and enabled billing. You can directly find your team name and you have admin permission.*

## 6. Setting up Firebase Project

We need to create our Firebase Project in the link below, <https://console.firebase.google.com>. You can see in the dropdown the name of the project (example: SampleFirebaseHosting is the name of my project).

*! For DPS participant you will find automatically your team name, so you do not have to create one*



The screenshot shows the 'Add a project' dialog in the Firebase console. It includes a 'Project name' dropdown set to 'SampleFirebaseHosting', a 'Project ID' field with 'samplefirebasehosting-4b3d7', and 'Locations' set to 'United States (Analytics)' and 'nam5 (us-central) (Cloud Firestore)'. There are checkboxes for sharing Google Analytics data and accepting terms, both of which are checked. At the bottom are 'Cancel' and 'Create project' buttons.

**Add a project**

Project name  
SampleFirebaseHosting

Project ID  
samplefirebasehosting-4b3d7

Locations  
United States (Analytics)  
nam5 (us-central) (Cloud Firestore)

☒ Use the default settings for sharing Google Analytics for Firebase data

- ✓ Share your Analytics data with all Firebase features
- ✓ Share your Analytics data with Google to improve Google Products and Services
- ✓ Share your Analytics data with Google to enable technical support
- ✓ Share your Analytics data with Google to enable Benchmarking
- ✓ Share your Analytics data with Google Account Specialists

☒ I accept the [controller-controller terms](#). This is required when sharing Analytics data to improve Google Products and Services. [Learn more](#)

Cancel Create project

Next, associate the Firebase CLI with your Firebase Account (Google Account) with the command:

*firebase login*

Then when you will be redirected through the browser, you should choose the Google account you used for creating the project (@dpschool.io) and you will get a successful login message. Next, go to the root folder of your React application then type in the command line:

*npm run build*

*firebase init hosting*

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: zsh +
belasinoimeri@U-CM-L112 samplefirebasehosting % npm run build

> samplefirebasehosting@0.1.0 build /Users/belasinoimeri/samplefirebasehosting
> react-scripts build

Creating an optimized production build...
Compiled successfully.

File sizes after gzip:

 39.85 KB build/static/js/2.d47a6ce6.chunk.js
 778 B   build/static/js/runtime-main.ae872105.js
 641 B   build/static/js/main.bef81976.chunk.js
 556 B   build/static/css/main.d1b05096.chunk.css

The project was built assuming it is hosted at /.
You can control this with the homepage field in your package.json.

The build folder is ready to be deployed.
You may serve it with a static server:

  serve -s build

Find out more about deployment here:

  bit.ly/CRA-deploy

belasinoimeri@U-CM-L112 samplefirebasehosting %
```

```
belasinoimeri@U-CM-L112 samplefirebasehosting % firebase init hosting

#####
##
#####
##
#####
##
#####
##

You're about to initialize a Firebase project in this directory:

  /Users/belasinoimeri/samplefirebasehosting

=== Project Setup

First, let's associate this project directory with a Firebase project.
You can create multiple project aliases by running firebase use --add,
but for now we'll just set up a default project.

? Please select an option: Use an existing project
? Select a default Firebase project for this directory: sample-firebase-hosting-bela (samplefirebasehosting)
i Using project sample-firebase-hosting-bela (samplefirebasehosting)

=== Hosting Setup

Your public directory is the folder (relative to your project directory) that
will contain Hosting assets to be uploaded with firebase deploy. If you
have a build process for your assets, use your build's output directory.

? What do you want to use as your public directory? build
? Configure as a single-page app (rewrite all urls to /index.html)? Yes
? File build/index.html already exists. Overwrite? No
i Skipping write of build/index.html

i Writing configuration info to firebase.json...
i Writing project information to .firebaserc...

✓ Firebase initialization complete!
belasinoimeri@U-CM-L112 samplefirebasehosting %
```

Follow the configuration in the image above, so we will use **build** as our public directory because that is the output directory of our app every time we run a build. We will also configure it as a **single-page** app (because it is a React app). And we will **not overwrite** our **build/index.html** file because we already build our application and we don't want to have some Firebase boilerplate content. Next, let's try deploying our app to Firebase. Just type:

*firebase deploy*

```
belasinoimeri@U-CM-L112 samplefirebasehosting % firebase deploy

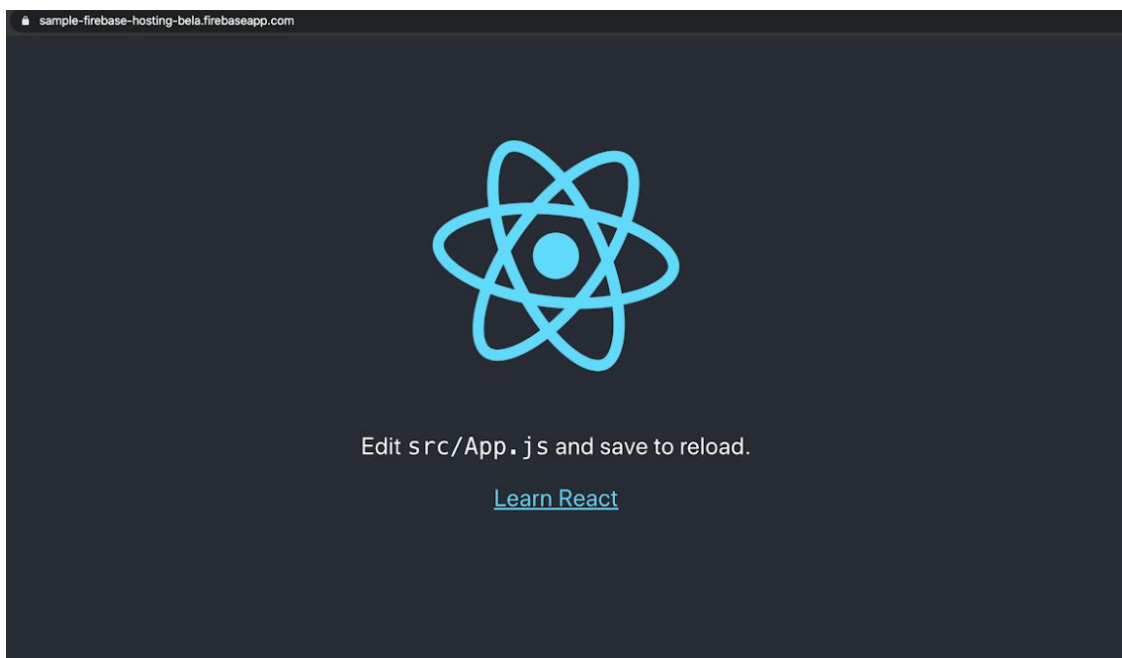
=== Deploying to 'sample-firebase-hosting-bela'...

i deploying hosting
i hosting[sample-firebase-hosting-bela]: beginning deploy...
i hosting[sample-firebase-hosting-bela]: found 19 files in build
✓ hosting[sample-firebase-hosting-bela]: file upload complete
i hosting[sample-firebase-hosting-bela]: finalizing version...
✓ hosting[sample-firebase-hosting-bela]: version finalized
i hosting[sample-firebase-hosting-bela]: releasing new version...
✓ hosting[sample-firebase-hosting-bela]: release complete

✓ Deploy complete!

Project Console: https://console.firebase.google.com/project/sample-firebase-hosting-bela/overview
Hosting URL: https://sample-firebase-hosting-bela.firebaseio.com
```

The URL (<https://sample-firebase-hosting-bela.firebaseio.com/>) will look like this:



## 7. Setting up Remote Git Repository

First, you need to set up your remote repository in Github, Bitbucket or GCP Source Repositories. For this guide we are gonna go with Github because it is easy to set up.


So first you need to login to your Github account or create if you don't have one, then create a new repository for your app.

## Create a new repository


A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner

Repository name \*

 bsinoimeri


 / 

SampleFirebaseHosting 


Great repository names are short and memorable. Need inspiration? How about [laughing-computing-machine?](#)

Description (optional)

A tutorial how to create a pipeline CI/CD with React, Firebase and Google Cloud Build

☒  **Public**

Anyone can see this repository. You choose who can commit.

☐  **Private**


You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☐ **Initialize this repository with a README**

This will let you immediately clone the repository to your computer.

Add .gitignore: **None**

Add a license: **None** 

Create repository

bsinoimeri / SampleFirebaseHosting

Unwatch 1 Star 0 Fork 0

[Code](#) [Issues 0](#) [Pull requests 0](#) [Actions](#) [Projects 0](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

**Quick setup — if you've done this kind of thing before**

Set up in Desktop

 or 

HTTPS

SSH

https://github.com/bsinoimeri/SampleFirebaseHosting.git

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

**...or create a new repository on the command line**

```
echo "# SampleFirebaseHosting" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/bsinoimeri/SampleFirebaseHosting.git
git push -u origin master
```

**...or push an existing repository from the command line**

```
git remote add origin https://github.com/bsinoimeri/SampleFirebaseHosting.git
git push -u origin master
```

**...or import code from another repository**

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

After creating the repository, get the remote URL and go to the root folder of your app and run the commands below:

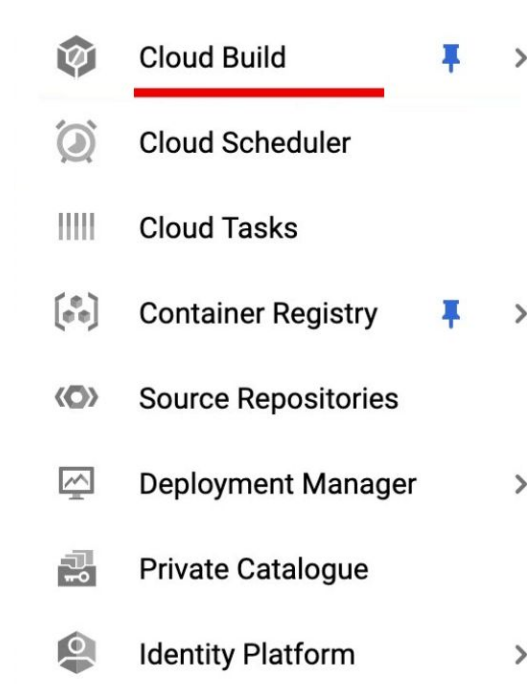
```
git add -A
git commit -m "<YOUR COMMIT MESSAGE>"
git remote add origin <REMOTE URL OF YOUR REPO>
```

*! If you want to add all the files then use git add .*

*! Don't push your changes to the remote repo yet because we will still configure our CI/CD Pipeline to automate our deployment.*

## 8. Setting up CI/CD Pipeline using GCP Cloud Build

Next, go to your GCP console in <https://console.cloud.google.com/> and find the Cloud Build Service in the Tools Section:



Next, enable your Cloud Build API if you have not yet enabled it and go to the Triggers tab, then click the Create trigger button.

Then, we will be choosing a Source for our trigger. So for now, we are gonna go with Github because we set up our remote repo there.



Cloud Build

Build history

Run your container image builds in a fast, consistent and reliable environment on Google Cloud Platform. Build in any language and package your build artefacts into Docker containers for deployment. Use Google Cloud SDK to integrate with your favourite developer tools and any continuous delivery system.

⚠️

Cloud Build API not enabled

Enable Cloud Build API

Cloud Build

Build triggers

Make sure that the container images that you build are up to date by creating a build trigger. A build trigger instructs Cloud Build to automatically build your image whenever there are changes pushed to the build source. You can set a build trigger to rebuild your images on any changes to the source repository, or only changes that match certain criteria. [Learn more](#)

Create trigger

Google Cloud Platform

samplefirebasehosting

🔍

Cloud Build

Dashboard

History

Triggers

Settings

←

Connect repository

1 Select source

2 Authenticate

3 Select repository

4 Create trigger (optional)

Select your source

☐ GitHub (Cloud Build GitHub App)

Pull request triggers are supported. Build status will be posted back to GitHub.

☐ Bitbucket (mirrored) BETA

☐ Cloud Source Repositories

These repositories are already connected. Create triggers for them [here](#).

☒ GitHub (mirrored) BETA

GitHub repositories will be mirrored in Cloud Source Repositories. Pull request triggers are not supported.

⤴️ Hide more options

☒ I consent to Google collecting and storing my authentication token in order to provide the connected repository service

Continue

Cancel



At step 2, you will be redirected to Github to authorize your project, just proceed and give authorization. The 3 step is to choose the repository in Github and directly after that you will be directed to step 4 Create Trigger.

☰

Google Cloud Platform

samplefirebasehosting ▾

Cloud Build

Dashboard

History

Triggers

Settings

←

Create trigger

Repository

bsinoimeri/SampleFirebaseHosting ▾

Name

Must be unique within the project

FirebaseHosting

Description

Push to any branch

Trigger type ?

☒ Branch

☐ Tag

Branch (regex) ?

No branch matches

☐ Invert Regex

.\*

Included files filter (glob) (Optional)

Changes affecting at least one included file will trigger builds

glob pattern example: src/\*\*

Ignored files filter (glob) (Optional)

Changes only affecting ignored files won't trigger builds

glob pattern example: .gitignore

⌵ Hide included and ignored files filters

Build configuration

☐ Dockerfile

Specify the path within the Git repo

☒ Cloud Build configuration file (yaml or json)

Specify the path to a Cloud Build configuration file in the Git repo [Learn more](#)

Cloud Build configuration file location ?

/ cloudbuild.yaml

Substitution variables (Optional)

Substitutions allow to re-use a cloudbuild.yaml file with different variable values [Learn more](#)

+ Add item

Create trigger

Cancel

**Name:** Name of your trigger

**Trigger type:** Which type do you want to listen for changes, For now, we are gonna go with the branch type.

**Branch (regex):** Listen for a regex pattern to trigger your build. I write *master* so that every time we commit a change to our master branch our build will be triggered or */\** for committing in any branch.

**Build configuration:** This is basically the file that holds the steps/scripts for our build config. We are going to create a file named *cloudbuild.yaml* in the root folder of our app that contains the code below:

```
steps:
- name: 'gcr.io/cloud-builders/npm'
  args: ['install']
- name: 'gcr.io/cloud-builders/npm'
  args: ['run', 'build']
- name: 'gcr.io/$PROJECT_ID/firebase'
  args:
    - deploy
    - --only=hosting
```

Do not replace the Project ID !

Cloud Build works by steps, then for each step, it references docker images to run a specific task. In our use case, it uses *npm* to install all dependencies and *run a build* to our React app. Then use the firebase image to deploy to Firebase Hosting. Cloud-builders are currently available images to the Cloud Build Service. And the firebase image is not available there.

So we need to create the firebase image. There is also a community that is building images that are useful like the [Cloud Builders Community](#). They have already created the firebase image so we don't need to do it from scratch. Go check out that link for more details on other docker images that you are interested in.

## Google Cloud Build official builder images

This repository contains source code for official builders used with the [Google Cloud Build API](#).

Pre-built images are available at [gcr.io/cloud-builders/...](https://gcr.io/cloud-builders/) and include:

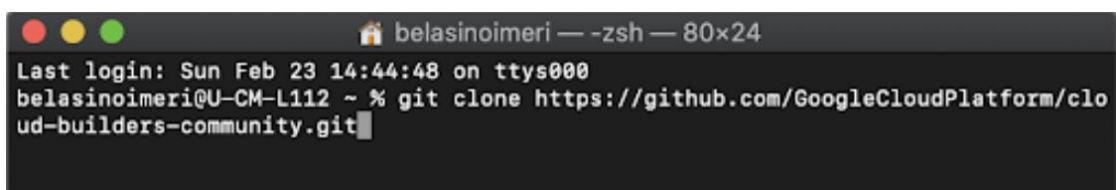
- `bazel` : runs the [bazel](#) tool
- `curl` : runs the [curl](#) tool
- `docker` : runs the [docker](#) tool
- `dotnet` : run the [dotnet](#) tool
- `gcloud` : runs the [gcloud](#) tool
- `git` : runs the [git](#) tool
- `go` : runs the [go](#) tool
- `gradle` : runs the [gradle](#) tool
- `gsutil` : runs the [gsutil](#) tool
- `kubect` : runs the [kubect](#) tool
- `mvn` : runs the [maven](#) tool
- `npm` : runs the [npm](#) tool
- `wget` : runs the [wget](#) tool
- `yarn` : runs the [yarn](#) tool

Builders contributed by the public are available in the [Cloud Builders Community repo](#).

To file issues and feature requests against these builder images, [create an issue in this repo](#). If you are experiencing an issue with the Cloud Build service or have a feature request, e-mail [google-cloud-dev@googlegroups.com](mailto:google-cloud-dev@googlegroups.com) or see our [Getting support](#) documentation.

So let's build the firebase image then save it to our own project. First, clone the Cloud Builders Community Repo with the command:

*`git clone https://github.com/GoogleCloudPlatform/cloud-builders-community.git`*

A terminal window with a dark background. The title bar shows three colored circles (red, yellow, green) and the text 'belasinoimeri — zsh — 80x24'. The terminal content shows: 'Last login: Sun Feb 23 14:44:48 on ttys000', 'belasinoimeri@U-CM-L112 ~ %', and the command 'git clone https://github.com/GoogleCloudPlatform/cloud-builders-community.git' being executed, followed by a cursor.

If you have not installed gcloud CLI before, you have to install it based on the documentation in this link <https://cloud.google.com/sdk/>. After doing this step, check that your email and the project id corresponds to your project in firebase with the command:

*`gcloud auth login`*

```
You are now logged in as [belasinoimeri@gmail.com].
Your current project is [sample-firebase-hosting-bela]. You can change this set
ting by running:
$ gcloud config set project PROJECT_ID
belasinoimeri@U-CM-L112 ~ %
```

Go to the cloud-builders-community/firebase directory that we cloned a while ago, build the firebase image and then save it in our GCP Project.

*cd cloud-builders-community/firebase*  
*gcloud builds submit --config cloudbuild.yaml .*










```
Removing intermediate container f0f6b0893ba5
--> ce6318d8635b
Successfully built ce6318d8635b
Successfully tagged gcr.io/sample-firebase-hosting-bela/firebase:latest
PUSH
Pushing gcr.io/sample-firebase-hosting-bela/firebase
The push refers to repository [gcr.io/sample-firebase-hosting-bela/firebase]
6b24a7c6c256: Preparing
6b24a7c6c256: Preparing
881900c717da: Preparing
e5ed415ee8fa: Preparing
7eddc4bdf1b0: Preparing
a997250354d4: Preparing
03dc1830d2d5: Preparing
1d7382716a27: Preparing
01727b1a72df: Preparing
69dfa7bd7a92: Preparing
4d1ab3827f6b: Preparing
7948c3e5790c: Preparing
03dc1830d2d5: Waiting
1d7382716a27: Waiting
01727b1a72df: Waiting
69dfa7bd7a92: Waiting
4d1ab3827f6b: Waiting
7948c3e5790c: Waiting
e5ed415ee8fa: Layer already exists
7eddc4bdf1b0: Layer already exists
a997250354d4: Layer already exists
1d7382716a27: Layer already exists
01727b1a72df: Layer already exists
03dc1830d2d5: Layer already exists
69dfa7bd7a92: Layer already exists
4d1ab3827f6b: Layer already exists
7948c3e5790c: Layer already exists
6b24a7c6c256: Pushed
881900c717da: Pushed
latest: digest: sha256:6972a63ca8fc30c13863df4703e9b17a964d0d13e66fcf62b8e4f0a2c94aed74 size: 2841
DONE
```

ID	CREATE_TIME	DURATION	SOURCE	STATUS
55d3b484-e0bf-4e12-965c-b4aac3a16544	2020-02-23T16:44:24+00:00	59S	gs://sample-firebase-hosting-bela_cloudbuild/source/1	
582476260.31-2c39a78feb1343c08d5b318c9567e6d7.tgz			gcr.io/sample-firebase-hosting-bela/firebase (+1 more)	SUCCESS

So if you successfully build it you are going to see something like this in the terminal. You can also check if the image was created in your console <https://console.cloud.google.com> in the Tools section, at the Container Registry Service.

As you can see below we successfully built our firebase image then saved it to the Container Registry Service. Container Registry Service is a service that stores, manages and secures your Docker container images in the cloud under the GCP. For more details [here](#).

## TOOLS

-  Cloud Build  
-  Cloud Scheduler
-  Cloud Tasks
-  Container Registry  
-  Source Repositories


## Repositories

 REFRESH

### SampleFirebaseHosting

Filter			All hostnames 
Name ^	Hostname	Visibility 	
 <u>firebase</u>	gcr.io	Private	

Lastly, let's set up some IAM permissions to our Cloud Build service account so that cloud build can deploy to Firebase Hosting. Go to IAM & admin service.

 Support

 IAM & admin

 Getting started







 Security

IAM  ADD  REMOVE

### Permissions for project SampleFirebaseHosting

These permissions affect this project and all of its resources. [Learn more](#)

View By: MEMBERS ROLES

cloud build  Filter table  				
<input type="checkbox"/> Type	Member ↓	Name	Role	Inheritance
<input type="checkbox"/> 	service-523654905389@gcp-sa-cloudbuild.iam.gserviceaccount.com		Cloud Build Service Agent	
<input type="checkbox"/> 	523654905389@cloudbuild.gserviceaccount.com		<u>Cloud Build Service Account</u>	

Select the cloudbuild like the picture above, click the edit icon and add Firebase Admin role, so that Cloud Build can deploy changes to Firebase.

Edit permissions

---

Member	Project
523654905389@cloudbuild.gserviceaccount.com	SampleFirebaseHosting

Role	
Cloud Build Service Accou... ▼ Can perform builds	🗑️
<div>Role Firebase Admin ▼ Full access to Firebase products.</div>	🗑️

[+ ADD ANOTHER ROLE](#)

[SAVE](#) [CANCEL](#)





**Last but not least**, let's change text in App.js to see that our continuous deployment really works. From *Edit src/App.js* and *save to reload* to *Our Continuous Deployment works !*

```
5 function App() {
6   return (
7     <div className="App">
8       <header className="App-header">
9         <img src={logo} className="App-logo" alt="logo" />
10        <p>
11          Our Continuous Deployment works !
12        </p>
13        <a
14          className="App-link"
15
```

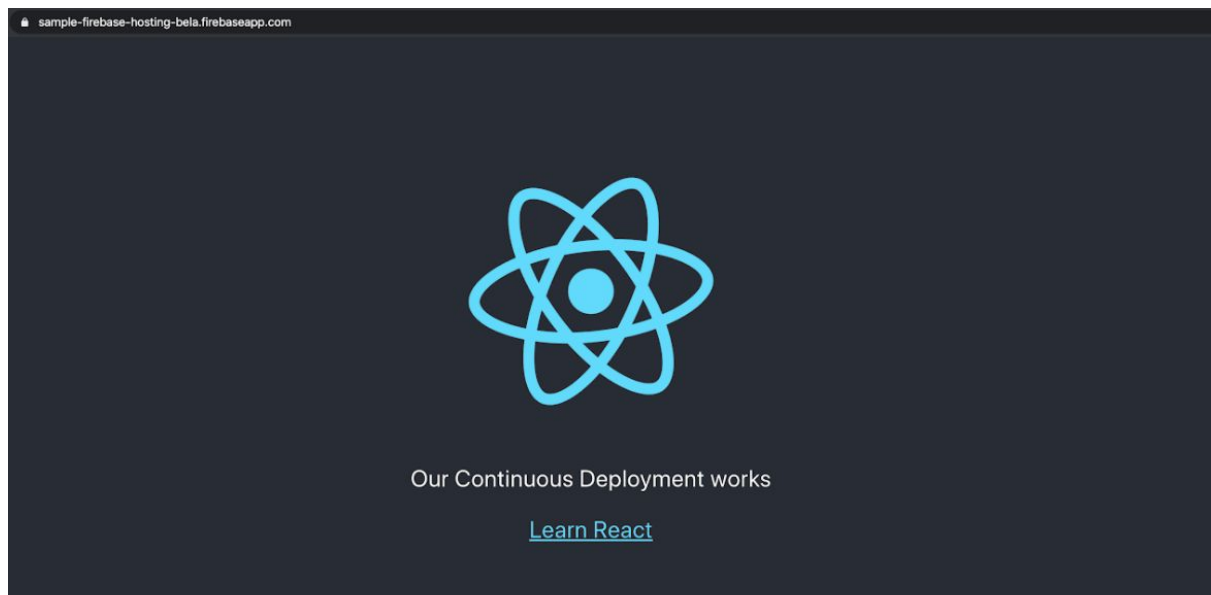
Let's commit all our changes and push to master branch to trigger the build that we set up in Cloud Build. Add all files including the cloudbuild.yaml because it will be needed for our trigger.

```
cd <your react app directory>
git add -A
git commit -m "<YOUR COMMIT MESSAGE>"
git push origin master
```

Go check the History Tab in the Cloud Build Service. And you can see a green build check by our cloud build trigger:

Build history <span>REFRESH</span>								
<input type="text" value="Filter builds"/> ?								
Build	Source	Git commit	Trigger name	Trigger	Started	Duration	Artefacts ?	
 02d80486-3685...	GitHub drys262/sample-firebase-hosting	2ebdc08	Push to master Branch (deploy to Firebase Hosting)	Push to master branch	Just now	—	—	
 7649a9aa-5a1b...	gs://samplefirebasehosting-4b3d7_cloudbuild/source/1557221261.48-7a7ea4bfc5504107aa4db4d10fc32963.tgz	—	—	—	17:27	53 sec	 gcr.io/samplefirebasehosting-4b3d7/firebase  gcr.io/samplefirebasehosting-4b3d7/firebase:latest	

Go browse your app to check if the changes are applied to it.



*So now you do not need to manually deploy every time your app. You just push your changes into the master branch and it is done !*

Github Link: <https://github.com/bsinoimeri/SampleFirebaseHosting>

## 9. Homework

- Try to do it for your own app, no matter what framework you are using :)