Deploy to Firebase

Test, build, and deploy your applications to Firebase by adding a pipe to your Bitbucket Pipelines configuration.

A full end-to-end example is available in this repository if you prefer hands-on experimentation with deploying to Firebase using Pipelines and Pipes

Step 1: Add your Firebase token and project name as variables

Define the following two variables in your settings:

Name	Value
`FIREBASE_TOKEN`	Firebase CLI token. Use a secured variable for this so that it is masked and encrypted in your logs.
`FIREBASE_PROJECT`	Firebase project name. If you don't have a project yet, you should go to the Firebase console and create one.

You can define these variables at the deployment environment, repository, or team level.

Step 2: Setup your Firebase environment

The next steps presume you have installed the Firebase CLI.

Your project should contain a `firebase.json` file containing your Firebase configuration.

This file is generated automatically after you locally run:

```
firebase init
```

Here's the `firebase.json` file from our example repository:

firebase.json'

```
{
  "hosting": {
    "headers": [
      {"source": "/service-worker.js", "headers": [{"key": "Cache-Control", "value":
    ],
    "public": "build",
    "ignore": [
      "firebase.json",
      "**/.*",
      "**/node_modules/**"
    ],
    "rewrites": [
      {
        "source": "**",
        "destination": "/index.html"
      }
    ]
  }
}
```

Step 3: Use a pipe to deploy to Firebase

Then we add the deployment configuration in your `bitbucket-pipelines.yml` file. We've included the `deployment`: keyword so that Bitbucket Deployments can track your deployment.

Below is a sample Bitbucket Pipelines configuration that deploys a ReactJS application (created with create-react-app), to Firebase. This example also provides insights on some best practices, like having separate steps for building and deploying, and using Bitbucket Deployments to review the deployment.

bitbucket-pipelines.yml

```
image: node

pipelines:
  default:
    - step:
        name: Test
        script:
          - npm install && npm test

    - step:
        name: Build
        script:
          - npm install && npm run build
        artifacts:
          - build/**

    - step:
        name: Deploy to Firebase
        deployment: production
        script:
          - pipe: atlassian/firebase-deploy:0.2.1
            variables:
              FIREBASE_TOKEN: $FIREBASE_TOKEN
              PROJECT_ID: $FIREBASE_PROJECT


```

Now when you commit your code, you can watch your test, build, and deploy steps as they progress. When it reports success, your application will be up and running in Firebase.

Access Pipelines deployment guides

Show more

- Deploy a Lambda function update to AWS
- Deploy to Amazon ECS
- Deploy to Firebase
- Deploy to Google Cloud
- Deploy to Heroku

Show more

On this page

- Step 1: Add your Firebase token and project name as variables
- Step 2: Setup your Firebase environment
- Step 3: Use a pipe to deploy to Firebase

Community

Questions, discussions, and articles