

Build, Publish and Run a Docker Image on Digital Ocean from Bitbucket Pipeline on Push

Dockerfile Frontend api

```
# pull official base image
FROM node:13.12.0-alpine
# add a bash shell to the image
RUN apk add --no-cache bash
# set working directory
WORKDIR /app

# add `/app/node_modules/.bin` to $PATH
ENV PATH /app/node_modules/.bin:$PATH
RUN echo "Path: $PATH"
# install app dependencies
COPY package.json ./
COPY package-lock.json ./
RUN npm install --silent
RUN npm install react-scripts@3.4.1 -g --silent

# add All to app
COPY . ./

# start app
# CMD ["npm", "start"]
# Best-Practice with ENTRYPOINT!
ENTRYPOINT ["npm", "start"]
```



Stage 1: build and run

bitbucket-pipelines.yml

```
# Template docker-push
# This template allows you to build and push your docker image to a Docker Hub account.
# The workflow allows running tests, code linting and security scans on feature branches (as well as master).
# The docker image will be validated and pushed to the docker registry after the code is merged to master.
# Prerequisites: $DOCKERHUB_USERNAME, $DOCKERHUB_PASSWORD setup as deployment variables
```

```
image: atlassian/default-image:2

pipelines:
  default:
    - parallel:
      - step:
          name: Build and Test
          script:
            - IMAGE_NAME=$BITBUCKET_REPO_SLUG
            - echo $IMAGE_NAME
            - docker build . --file Dockerfile --tag ${IMAGE_NAME}
          services:
            - docker
          caches:
            - docker
      - step:
          name: Lint the Dockerfile
          image: hadolint/hadolint:latest-debian
          script:
            - hadolint Dockerfile

  branches:
    master:
      - step:
          name: Build and Test
          script:
            - IMAGE_NAME=$BITBUCKET_REPO_SLUG
            - docker build . --file Dockerfile --tag ${IMAGE_NAME}
            - docker save ${IMAGE_NAME} --output "${IMAGE_NAME}.tar"
          services:
            - docker
          caches:
            - docker
          artifacts:
            - "*.tar"
```

Stage 1: build and Test run parallel set
DOCKER_HUB USERNAME and password as variable
IMAGE_NAME: app-cart
IMAGE: /app-cart
VERSION: prod-0.1.X BB Build Number

```
docker build . -file Dockerfile --tag app-cart
docker save app-cart -output app-cart.tar
```

```

- step:
  name: Deploy to Production
  deployment: Production
  script:
    - echo ${DOCKERHUB_PASSWORD} | docker login --username "${DOCKERHUB_USERNAME}" --password-stdin
    - IMAGE_NAME=${BITBUCKET_REPO_SLUG}
    - echo "image name -> " $IMAGE_NAME "image -> " $IMAGE
    - docker load --input "${IMAGE_NAME}.tar"
    - VERSION="prod-0.1.${BITBUCKET_BUILD_NUMBER}"
    - IMAGE=${DOCKERHUB_NAMESPACE}/${IMAGE_NAME}
    - echo "image name -> " $IMAGE_NAME "image -> " $IMAGE "version -> " $VERSION
    - docker tag "${IMAGE_NAME}" "${DOCKERHUB_USERNAME}/${IMAGE}:${VERSION}"
    - docker push "${DOCKERHUB_USERNAME}/${IMAGE}:${VERSION}"
  services:
    - docker

```

echo to console ->

Stage 1: Deploy to Production
docker push to docker - hub



```

docker load --input "${IMAGE_NAME}.tar"
VERSION="prod-0.1.${BITBUCKET_BUILD_NUMBER}"
IMAGE=${DOCKERHUB_NAMESPACE}/${IMAGE_NAME}
echo "image name -> " $IMAGE_NAME "image -> " $IMAGE "version -> " $VERSION
+ echo "image name -> " $IMAGE_NAME "image -> " $IMAGE "version -> " $VERSION
image name -> app-tictactoe image -> /app-tictactoe version -> prod-0.1.9

```

Production environments

Production

Environment name: Production

Branches allowed to deploy to Production: All branches can deploy - to restrict: select branch or type a glob pattern

☐ Only allow admins to deploy to this environment

Control deployment restrictions with our premium plan. [Learn more](#)

Variables

Name	Value	Secured	Add
DOCKERHUB_USERNAME	<input checked="" type="checkbox"/>	
DOCKERHUB_PASSWORD	<input checked="" type="checkbox"/>	

it functions now:

Luigi Cavuoti / proj-tictactoe / app-tictactoe




Pipelines

What's new Run pipeline Schedules Caches Usage ?

All branches Status Trigger type Only mine

OTHER BRANCHES

master MAIN BRANCH

Pipeline	Status	Started	Duration
#10  bitbucket-pipeline adding the DOCKERHUB_USERNAME to th... Luigi Cavuoti  f5f17fa  master	Successful	12 minutes ago	3 min 5 sec


Luigi Cavuoti / ... / Pipelines

✓ #10 Rerun

✚ f5f17fa bitbucket-pipeline adding the DOCKERHUB_USERNAME to the push command

↳ master

[Learn more about reports](#)

🕒 3 min 5 sec 🕒 12 minutes ago 

Pipeline

Build and Test 1m 58s

Deploy to Production 1m 07s Redeploy

```
Build docker

Build setup 11s >

echo ${DOCKERHUB_PASSWORD} | docker login --username "${DOCKERHUB_USERNAME}" --password-stdin 1s >

IMAGE_NAME=${BITBUCKET_REPO_SLUG} <1s >

echo "image name -> " $IMAGE_NAME "image -> " $IMAGE <1s >

docker load --input "${IMAGE_NAME}.tar" 16s >

VERSION="prod-0.1.${BITBUCKET_BUILD_NUMBER}" <1s >

IMAGE=${DOCKERHUB_NAMESPACE}/${IMAGE_NAME} <1s >

echo "image name -> " $IMAGE_NAME "image -> " $IMAGE "version -> " $VERSION <1s >

docker tag "${IMAGE_NAME}" "${DOCKERHUB_USERNAME}${IMAGE}:${VERSION}" <1s >

docker push "${DOCKERHUB_USERNAME}${IMAGE}:${VERSION}" 30s >

Build teardown <1s >
```

docker push
-t tag: lcavuoti
(\$DOCKERHUB_USERNAME)
cart -> name of the app (\$IMAGE)
tag: latest | 1 | etc. (\$VERSION)

Docker-Hub new Image on git – Push (bitbucket)

Advanced Image Management

View all your images and tags in this repository, clean up unused content, recover untagged images. Available for Pro and Team accounts.

View preview

lcavuoti / app-tictactoe

This repository does not have a description

Last pushed: 13 minutes ago

Docker commands

To push a new tag to this repository,

`docker push lcavuoti/app-tictactoe:tagname`

Public View

Tags and Scans

VULNERABILITY SCANNING - DISABLED
Enable

This repository contains 1 tag(s).

TAG	OS	PULLED	PUSHED
prod-0.1.10		13 minutes ago	13 minutes ago

See all

Recent builds

Link a source provider and run a build to see build results here.

Readme

Repository description is empty. Click here to edit.

Docker credentials <https://docs.docker.com/engine/reference/commandline/login/>

```
code $HOME/.docker/config.json
```

```
{
  "auths": {
    "registry.heroku.com": {},
    "https://index.docker.io/v1/": {}
  },
  "credsStore": "desktop",
  "stackOrchestrator": "swarm",
  "experimental": "disabled"
}
```

bitbucket-pipelines.yml-manually

```
# Template docker-push
# This template allows you to build and push your docker image to a Docker Hub account.
# The workflow allows running tests, code linting and security scans on feature branches (as well as master).
# The docker image will be validated and pushed to the docker registry after the code is merged to master.
# Prerequisites: $DOCKERHUB_USERNAME, $DOCKERHUB_PASSWORD setup as deployment variables
```

```
image: atlassian/default-image:2

pipelines:
  default:
    - parallel:
      - step:
          name: Build and Test
          script:
            - IMAGE_NAME=$BITBUCKET_REPO_SLUG
            - echo $IMAGE_NAME
            - docker build . --file Dockerfile --tag ${IMAGE_NAME}
          services:
            - docker
          caches:
            - docker
      - step:
          name: Lint the Dockerfile
          image: hadolint/hadolint:latest-debian
          script:
            - hadolint Dockerfile

  branches:
    master:
      - step:
          name: Build and Test
          script:
            - IMAGE_NAME=$BITBUCKET_REPO_SLUG
            - docker build . --file Dockerfile --tag ${IMAGE_NAME}
            - docker save ${IMAGE_NAME} --output "${IMAGE_NAME}.tar"
          services:
            - docker
          caches:
            - docker
          artifacts:
            - "*.tar"
```

Stage 1: build and Test run parallel set
DOCKER_HUB USERNAME and password as variable
IMAGE_NAME: app-cart
IMAGE: /app-cart
VERSION: prod-0.1.X BB Build Number

-- make an image and than a .tar file
docker build . --file Dockerfile --tag app-cart
docker save app-cart --output app-cart.tar


```
- step:
  name: Deploy to Production
  deployment: Production
  script:
    - echo ${DOCKERHUB_PASSWORD} | docker login --username "$DOCKERHUB_USERNAME" --password-stdin
    - IMAGE_NAME=${BITBUCKET_REPO_SLUG}
    - echo "image name -> " $IMAGE_NAME "image -> " $IMAGE
    - docker load --input "${IMAGE_NAME}.tar". // load the image from a tar file
    - VERSION="prod-0.1.${BITBUCKET_BUILD_NUMBER}"
    - IMAGE=${DOCKERHUB_NAMESPACE}/${IMAGE_NAME}
    - echo "image name -> " $IMAGE_NAME "image -> " $IMAGE "version -> " $VERSION
    - docker tag "${IMAGE_NAME}" "${DOCKERHUB_USERNAME}:${IMAGE}:${VERSION}" // make a new image with the full tag
    - docker push "${DOCKERHUB_USERNAME}:${IMAGE}:${VERSION}". // push to docker-hub
  services:
    - docker
```

echo to console ->

Stage 1: Deploy to Production
docker push to docker - hub

1. build the image

```
docker build . --file Dockerfile --tag app-cart
```

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
app-cart	latest	705f2dbf7711	20 minutes ago	25MB

2. save image to a tar file

```
docker save app-cart --output app-cart.tar
```

```
$ ls *.tar
```

```
app-cart.tar
```

3. Production -> new Context (load the image from the .tar file)

```
$ docker load --input app-cart.tar
```

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
app-cart	latest	705f2dbf7711	24 minutes ago	25MB

4. make a new tag

```
$ docker tag app-cart lcavuoti/app-cart:prod-0.1.10
```

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
lcavuoti/app-cart	prod-0.1.1	705f2dbf7711	26 minutes ago	25MB
lcavuoti/app-cart	prod-0.1.10	705f2dbf7711	26 minutes ago	25MB
app-cart	latest	705f2dbf7711	26 minutes ago	25MB

5. push to docker hub

```
$ docker push lcavuoti/app-cart:prod-0.1.10
```

```
The push refers to repository [docker.io/lcavuoti/app-cart]
```


```
d3c0b545f7e1: Layer already exists
```


```
3810cc0c140f: Layer already exists
```

```
3e207b409db3: Layer already exists
```

```
prod-0.1.10: digest: sha256:36432601d79f3200c8ab83fbfa3b93605405227d54a8cf445ee9f43ce50cc654 size: 950
```

lcavuoti / app-cart

This repository does not have a description 


 Last pushed: a minute ago

Docker comm





To push a new ta

docker push

Tags and Scans

 VULNERABILITY SCANNING - DISABLED
[Enable](#)

This repository contains 6 tag(s).

TAG	OS	PULLED	PUSHED
 prod-0.1.10		17 minutes ago	a minute ago
		17 minutes ago	17 minutes ago

Recent builds

[Link a source pr](#)