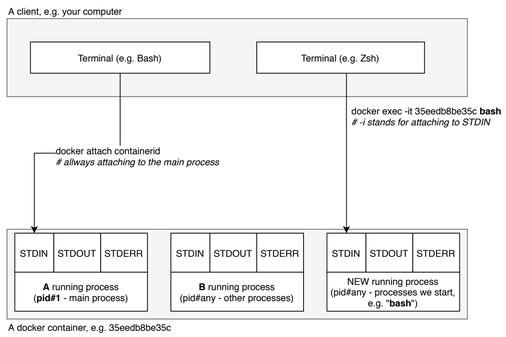
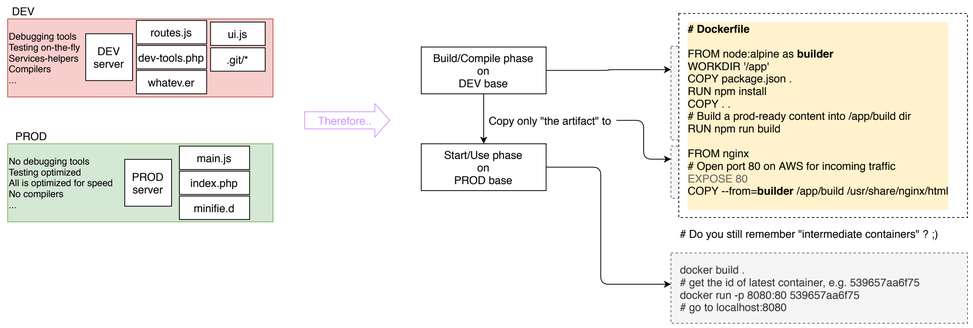
 # Day #3: Docker > Travis CI > AWS

Q: How do I deploy Docker containers with Kubernetes via Travis CI to AWS or other type of hosting platform?
A: Deploying basic app first, including gitflow (features, pull requests) taken into account. Docker > Travis CI > AWS.

Sneaking in process tree of a container



DEV / PROD just differs...



Github

1. Create Github account & repo
   (in github.com)
2. Create local git repo
   git init
   (make working files, merge files in)
   git add .
   git commit -m 'Init'
3. Connect local git to github remote
   git remote add origin git@github...yourrepo.git
4. Push work to github
   git push origin master

   ....~gitflow~...

5. git checkout -b feature
   (...and change the code)
6. git add .
7. git commit -m 'Feature is done.'
8. git push origin feature
9. Open a pull request on github.com repo (feature to master)
   (**This will trigger Travis CI check**; not a build yet)
10. Merge pull request
    (**This will trigger Travis CI build** to AWS)

Travis CI

1. Create Travis CI account
   (in travis-ci.org)
2. Authorize Travis CI to work with Github
   (user data, grant permissions to list repositories, etc.)
3. Activate your GitHub repositories, if not listed any
   (need to be an admin for a repository to setup on Travis CI)
4. Turn on the repository you want
   (switch repo on in the list of repositories)
5. Create and push **.travis.yml** file in git repo telling:
   (p.s. build will start immediately after git push because of step #3 and #4)
   - We need a copy of docker running
   - Build image using Dockerfile.dev
   - Run our test suite
   - Deploy to AWS

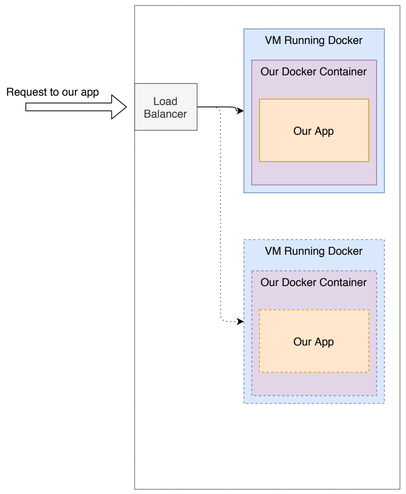
 # .travis.yml

```
# avoiding "rakefile not found" error
language: generic

# we need super user for furter actions
sudo: reuired

# Install docker to Travis CI space
services:
  - docker

# Build a locally tagged docker image
before_install:
  - docker build -t myvendorname/my-repo-name -f Dockerfile.dev .

# https://facebook.github.io/create-react-app/docs/running-tests#linux-macos-bash
# https://docs.docker.com/engine/reference/run/#env-environment-variables
script:
  - docker run -e CI=true myvendorname/my-repo-name npm run test -- --coverage


# edge: true
#  forcing Travis to use the v2 (experimental) version of the dpl script
#  which does not have the bug "Missing bucket_name"
# provider: deploy to one of many preconfigured providers - elasticbeanstalk
#  other options: github pages, github releases, google app engine, etc.
# app: applicationNameFromAwsBeanstalk
# env: environmentCreatedInApplication
# bucket_name: a-value-of-aws-s3-bucket-name-where-travisci-sends-zip-to
#  this bucket was automatically generated after creating beanstalk app
#  so look in S3 service bucket list
# bucket_path: a path inside the bucket of the app
#  if it's the first env in app then it most likely will be same as app name
#  otherwise - something else, browse S3 bucket to find it, or create extra dir
# on: we deploy on git push to this branch of github repo
deploy:
  edge: true
  provider: elasticbeanstalk
  region: "eu-central-1"
  app: "docker-deploy-test"
  env: "Docker-env"
  bucket_name: "elasticbeanstalk-eu-central-1-30647667547"
  bucket_path: "docker-deploy-test"
  on:
    branch: master
  access_key_id: $AWS_ACCESS_KEY
  secret_access_key: $AWS_SECRET_KEY
```

AWS

1. Create AWS account
   (in http://aws.amazon.com/)
2. My Account > AWS Management Console > Sign In

AWS Beanstalk Environment - for auto scaling...



Elastic Beanstalk
(provider, region, app, env)

S3
(bucket_name, bucket_path)

IAM

1. Pick "Elastic Beanstalk" from services
2. Click "Create new application"
   Application name: "Docker whatever"
   Click "Create"
3. Create environment for application
   Pick "Web server environment" now
   (Not "Worker environment" yet)
   Click "Select"
4. Under "Base configuration" - "Platform" - "Preconfigured platform" choose "Docker"
5. Under "Application code" choose "Sample application"
6. Click "Create environment"
7. Find environment **URL**

1. Services > "S3"
   (all what is needed for .travis.yml was created at the time of creating beanstalk application)

1. Services > "IAM"
2. "Users" > "Add user":
   Username: docker-react-travis-ci-whatever
   Access type: "Programmatic access" (check it, the rest may left unchecked)
3. "Next: Permissions":
   "Attach existing policies directly"
   Filter for "beanstalk"
   Select checkbox of the one with "Provides full access to AWS Elastic Beansta.."
4. "Next: Review"
5. "Create user"
6. "Show" (on "Secret Access Key")
   Make sure you copy the value because you are allowed **to see it just once.**
7. You now have values of:
   Access key ID
   Secret access key
   Password

1. Open project in Travis CI
2. More options > Settings > Environment variables
   Add name/value pairs form IAM user
   AWS_ACCESS_KEY , WHATEVERISINAWSUNDERACCESKEYID , click "Add"
   AWS_SECRET_KEY , WHATEVERISINAWUNDERSECRETACCESSKEY , click "Add"
3. Make sure you **un**check "Display value in build log"

WHEN NOT USING APPS -- DO NOT FORGET:

- Go to the Elastic Beanstalk dashboard
- Actions > Delete Application > **Delete**

It may take few minutes to delete.

You have to pay for the existing apps.