

간단한 어플을 실제로 배포해보기(테스트 & 배포 부분)

이번 섹션 설명

간단하게

개발환경에서
개발

Development

Test

Production

개발 된 것들을
테스트

배포

세세하게

개발환경에서
개발

Github에
소스를 Push

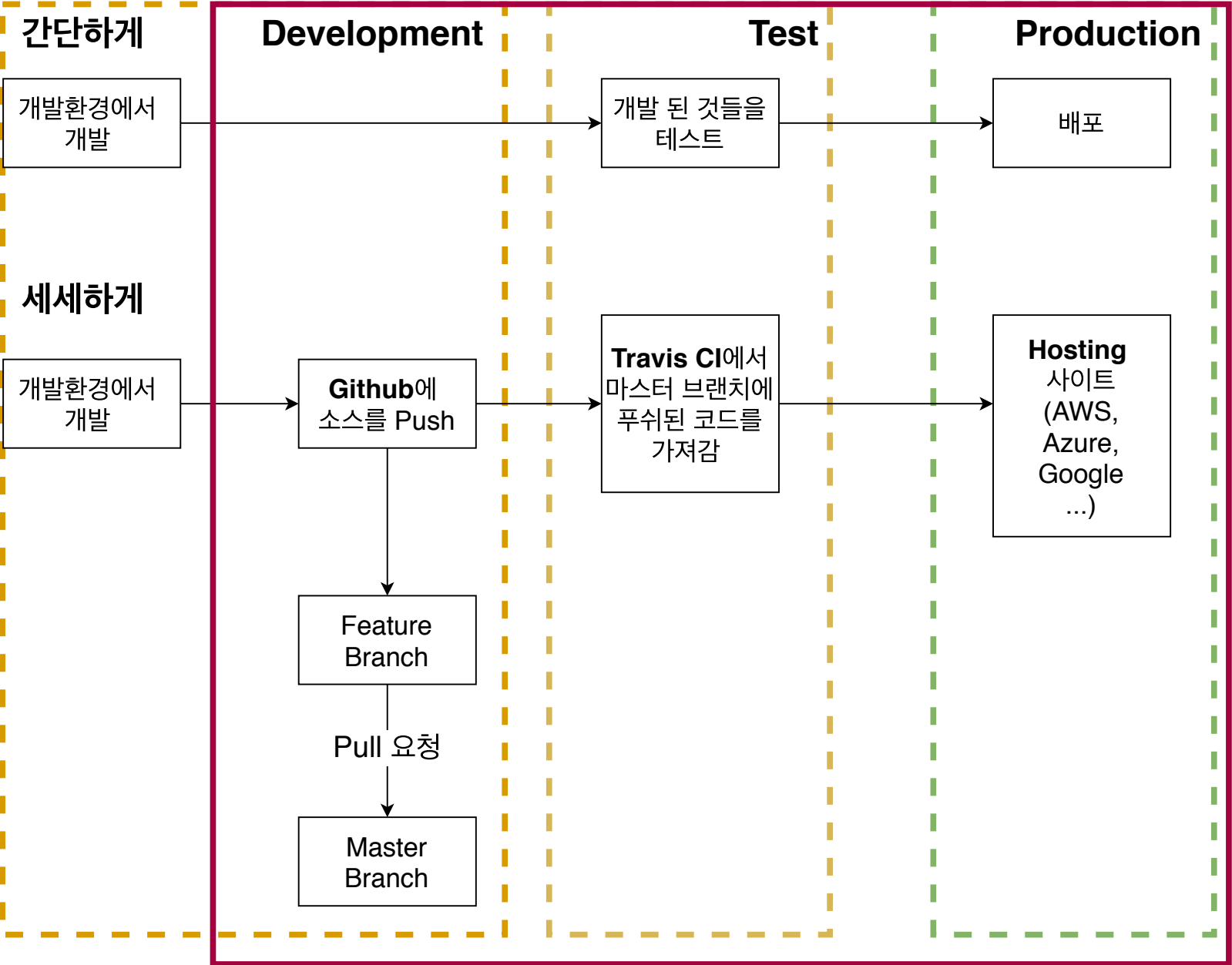
Feature
Branch

Pull 요청

Master
Branch

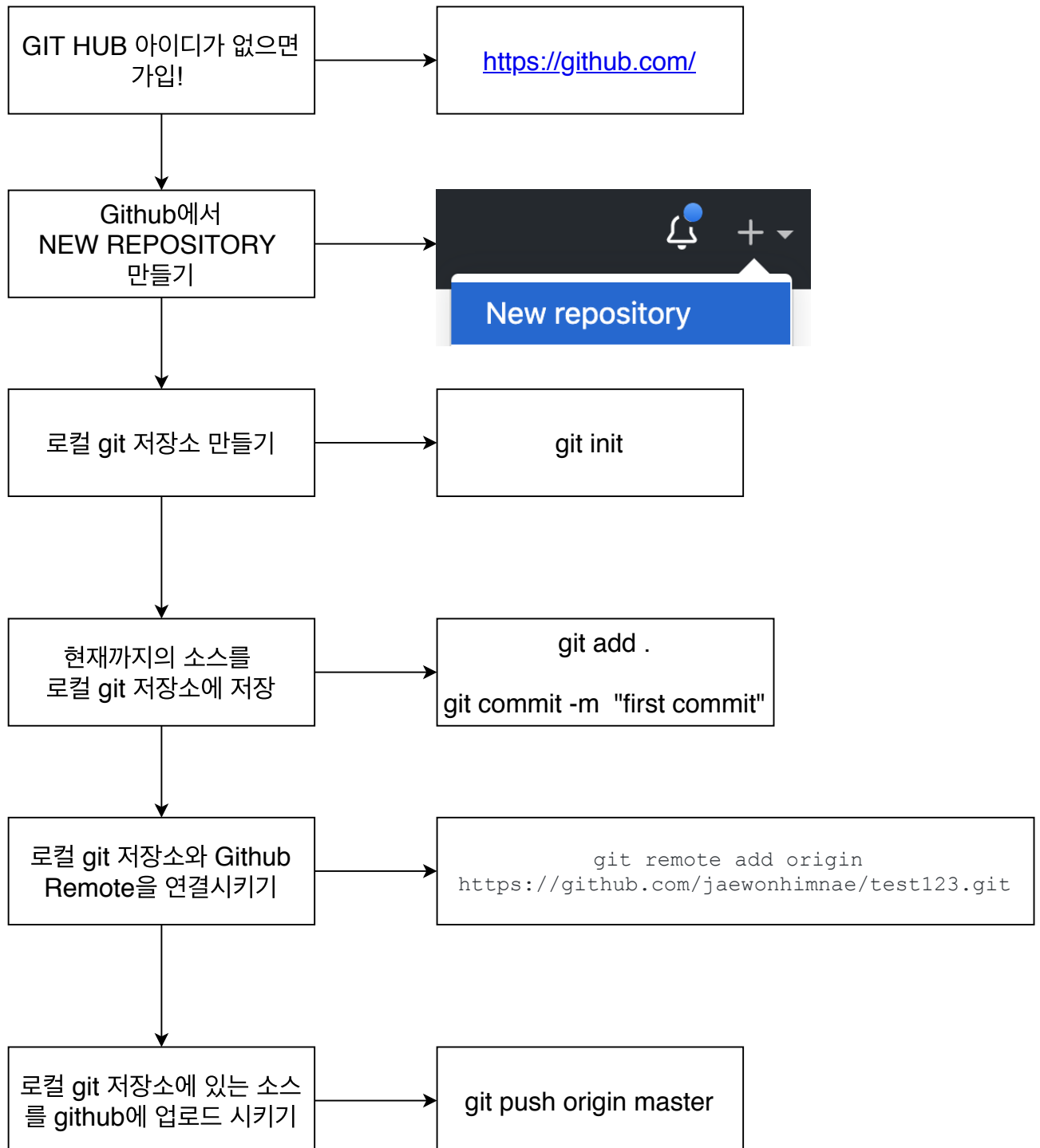
Travis CI에서
마스터 브랜치에
푸쉬된 코드를
가져감

Hosting
사이트
(AWS,
Azure,
Google
...)



Github에 소스 코드 올리기

Steps

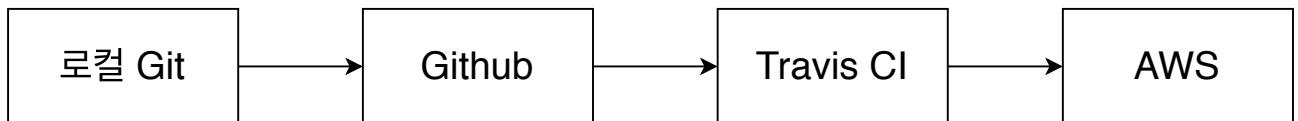


Travis CI 설명

Travis CI 란 ?

Travis CI는 Github에서 진행되는 오픈소스 프로젝트를 위한 지속적인 통합(Continuous Integration) 서비스이다. 2011년에 설립되어 2012년에 급성장하였으며 Ruby언어만 지원하였지만 현재 대부분의 개발언어를 지원하고 있다. Travis CI를 이용하면 Github repository에 있는 프로젝트를 특정 이벤트에 따라 자동으로 **테스트**, 빌드하거나 **배포**할 수 있다. Private repository는 유료로 일정 금액을 지불하고 사용할 수 있다.

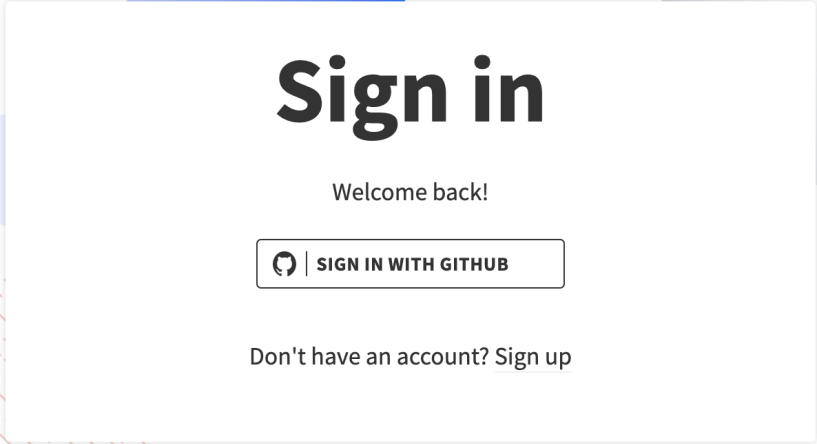
Travis CI의 흐름



Travis CI 이용 순서

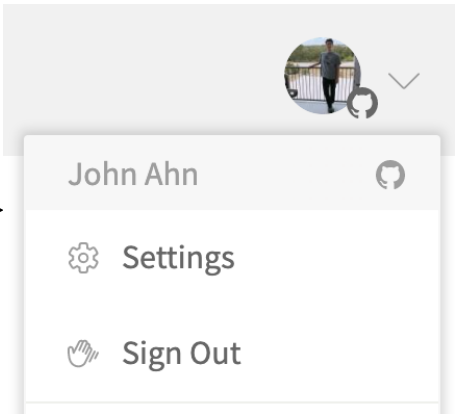
Travis CI 사이트로 이동 → <https://travis-ci.org/>

Travis CI에 로그인 할때
깃헙 아이디로 로그인



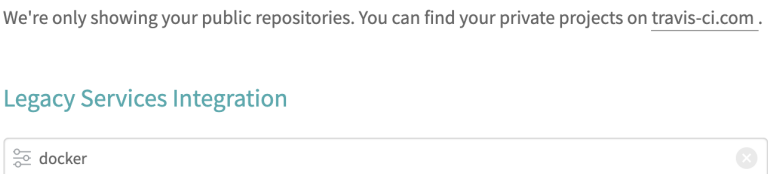
The image shows the Travis CI 'Sign in' page. It features a large 'Sign in' heading, a 'Welcome back!' message, and a 'SIGN IN WITH GITHUB' button. Below the button, it says 'Don't have an account? Sign up'.

Settings 페이지로 이동



The image shows a user profile dropdown menu for 'John Ahn'. It includes a profile picture, a 'Settings' option with a gear icon, and a 'Sign Out' option with a hand icon.

깃헙에 올린 도커 리액트앱
저장소를
Travis CI에서 찾기



The image shows the 'Repositories' page on Travis CI. It has tabs for 'Repositories' and 'Settings'. Below the tabs, it says 'We're only showing your public repositories. You can find your private projects on travis-ci.com.' There is a search bar with 'docker' entered and a 'Legacy Services Integration' link.

해당 저장소의 Setting 버튼을 눌러서 Travis CI에게 Github 저장소의 소스가 변경될때 마다 소스를 가져와서 **테스트**하고 **배포**하라고 알려줌.

 docker-react-app

Dashboard로 이동.

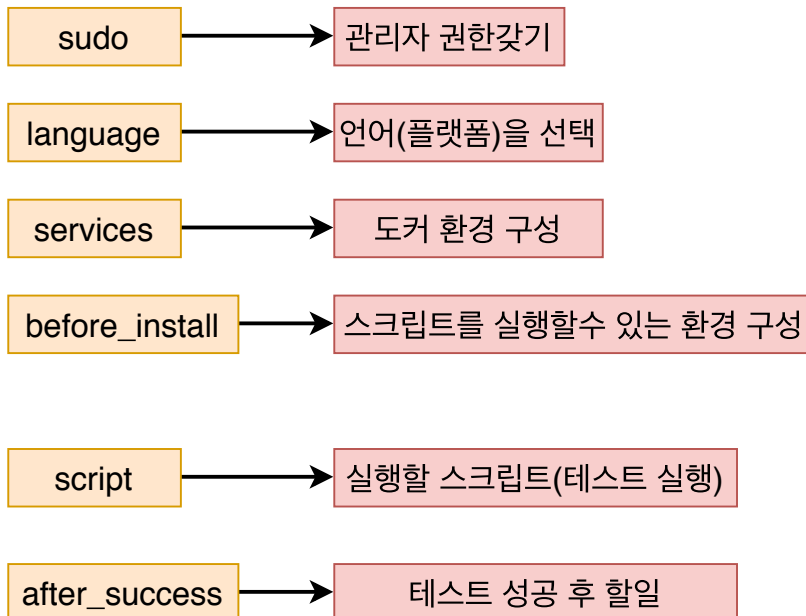
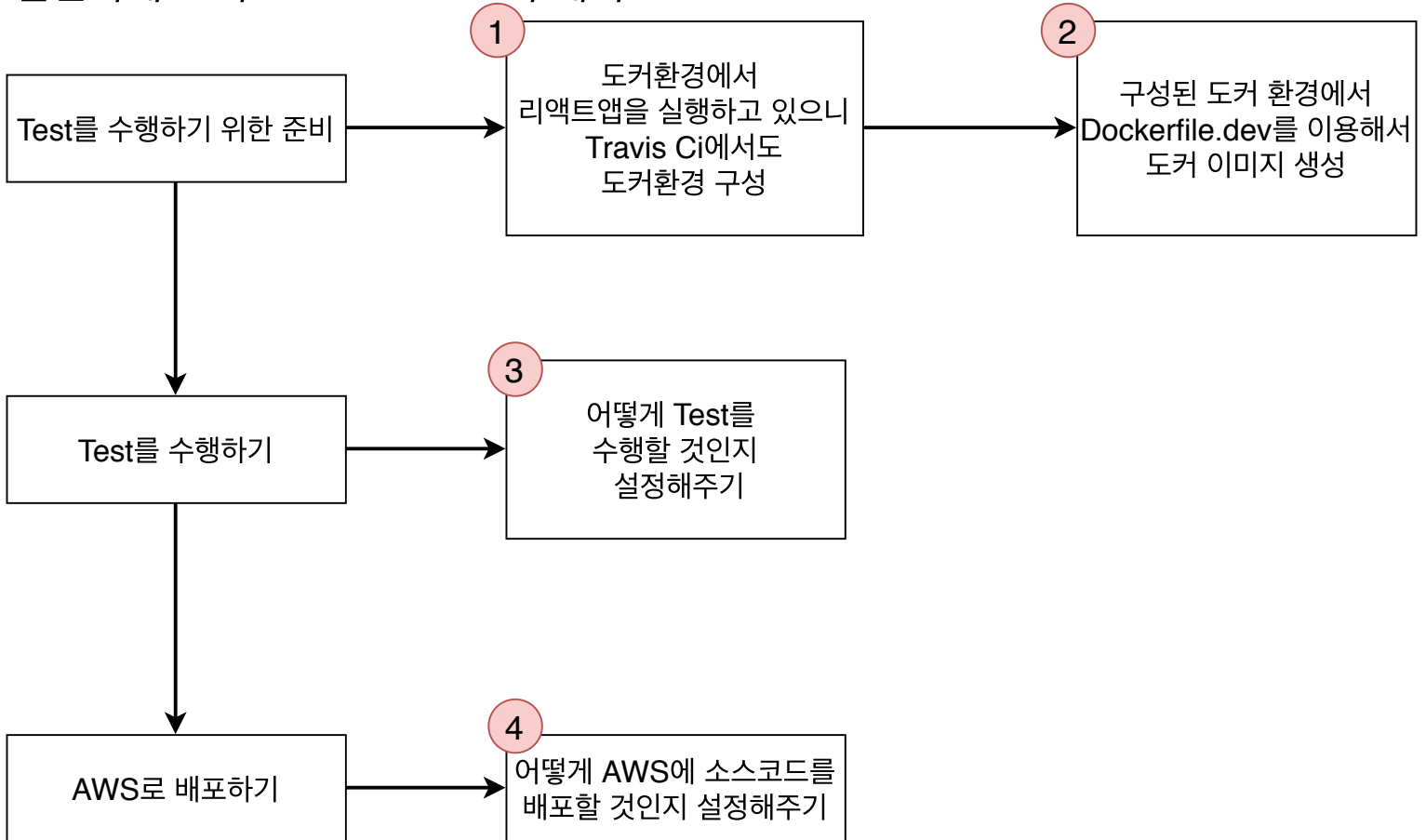
Dashboard

.travis.yml 파일 작성하기 (테스트까지)

어떻게 작성하나요?

간단하게 보기

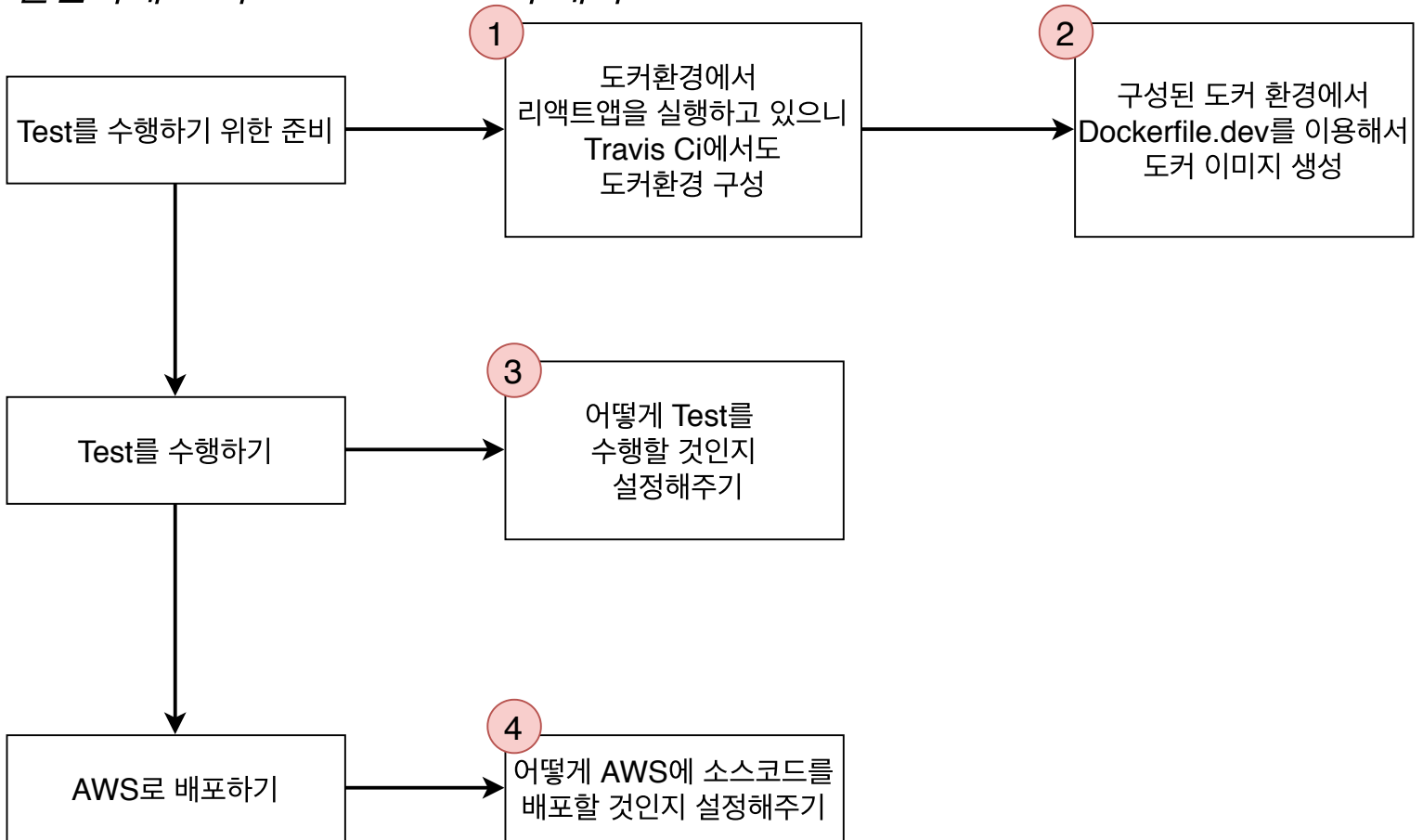
구체적으로



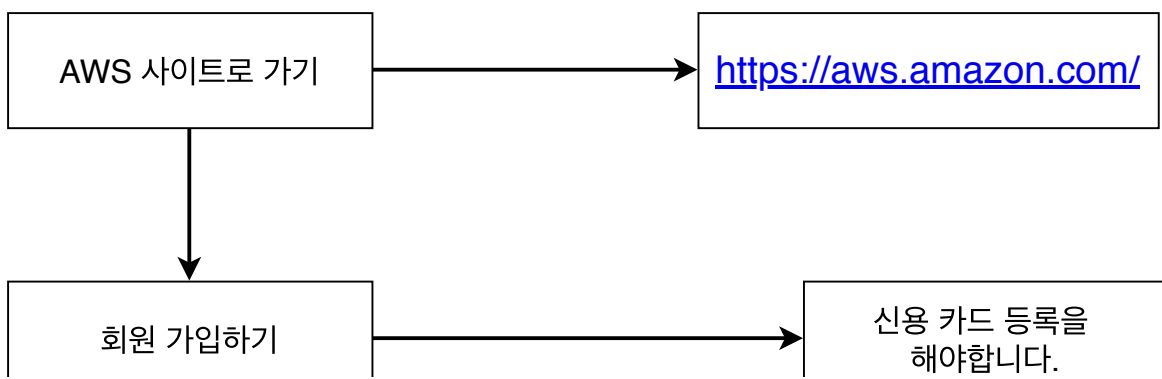
AWS 알아보기

간단하게 보기

구체적으로



AWS로 배포 하는 순서

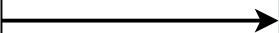




AWS Dashboard로 오기



Elastic BeanStalk 검색



AWS services

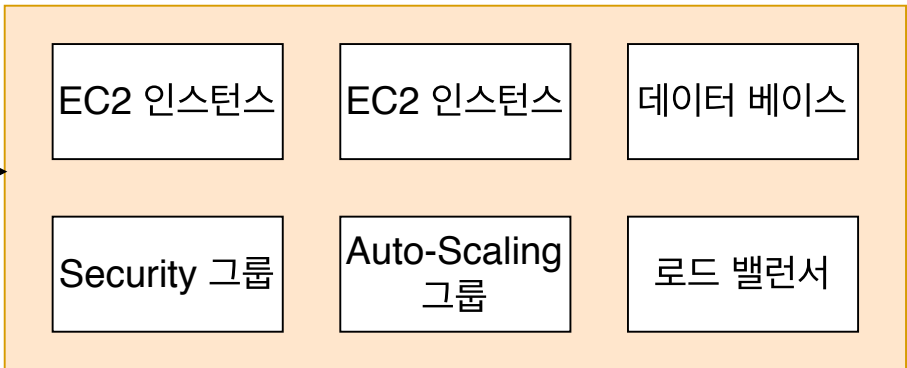
Find Services

You can enter names, keywords or acronyms.

🔍 Elastic Beanstalk



컨트롤



Elastic Beanstalk에서 어플리케이션 만들기

어플리케이션 만드는 순서

Get started

Easily deploy your web application in minutes.

Create Application

Create Application
버튼 클릭

어플리케이션 이름 정하기

Application information

Application name

docker-react-app

Application name

docker-react-app

Application name

docker-react-app

어플리케이션 플랫폼 선택

Platform	
----------	--

Platform

Docker

Platform

Docker

Platform branch
Docker running on 64bit Amazon Linux 2

Platform branch
Docker running on 64bit Amazon Linux 2

Platform version


3.0.2 (Recommended) ▼


Platform version


3.0.2 (Recommended) ▼

Create Application
버튼 눌러서 생성하기

Create application

 **Creating DockerReactApp-env**
This will take a few minutes. ..

 **Creating DockerReactApp-env**
This will take a few minutes. ..

 **Creating DockerReactApp-env**
This will take a few minutes. ..

```
9:41pm Successfully launched environment: DockerReactApp-env
9:41pm Application available at DockerReactApp-env.eba-x5uamned.ap-northeast-2.elasticbeanstalk.com.
9:42pm Added instance i-06764704100624e01 to your environment.
```

```
9:41pm Successfully launched environment: DockerReactApp-env
9:41pm Application available at DockerReactApp-env.eba-x5uamned.ap-northeast-2.elasticbeanstalk.com.
9:42pm Added instance i-06764704100624e01 to your environment.
```

```
9:41pm Successfully launched environment: DockerReactApp-env
9:41pm Application available at DockerReactApp-env.eba-x5uamned.ap-northeast-2.elasticbeanstalk.com.
9:42pm Added instance i-06764704100624e01 to your environment.
```

앱 생성중

9:40pm Added instance [i-0676d70118a13a4ee] to your environment.
9:40pm Waiting for EC2 instances to launch. This may take a few minutes.
9:39pm Environment health has transitioned to Pending. Initialization in progress (running for 31 seconds). There are no instances.
9:39pm Created EIP: 3.34.54.242
9:39pm Created security group named: awseb-e-fqe526adqa-stack-AWSEBSecurityGroup-O94N8QRWV82G
9:39pm Using elasticbeanstalk-ap-northeast-2-972153559337 as Amazon S3 storage bucket for environment data.
9:39pm createEnvironment is starting.

다 생성 후

DockerFullstackApp-env
DockerFullstackApp-env:eb-p-afk2an8s.ap-northeast-2.amazonaws.com [p-afk2an8s]
Application name: docker-fullstack-app

Refresh

Actions ▼

Health



OK

Console

Running version

Sample Application

Upload and deploy

Platform

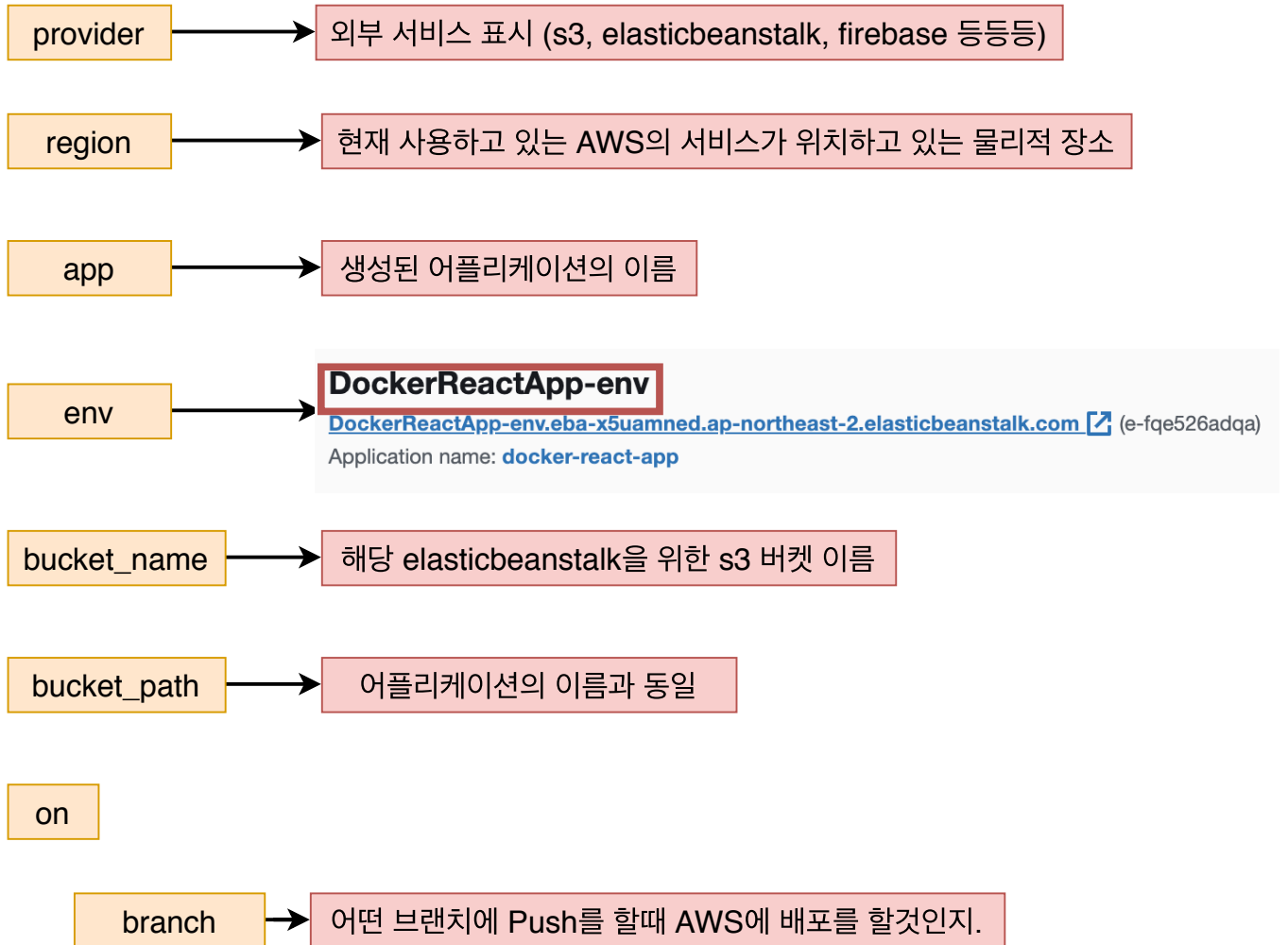


Multi container Docker running on
64bit Amazon Linux/2.86.4

Change

.travis.yml 파일 작성하기 (배포 부분)

deploy 새롭게 추가된 부분 설명

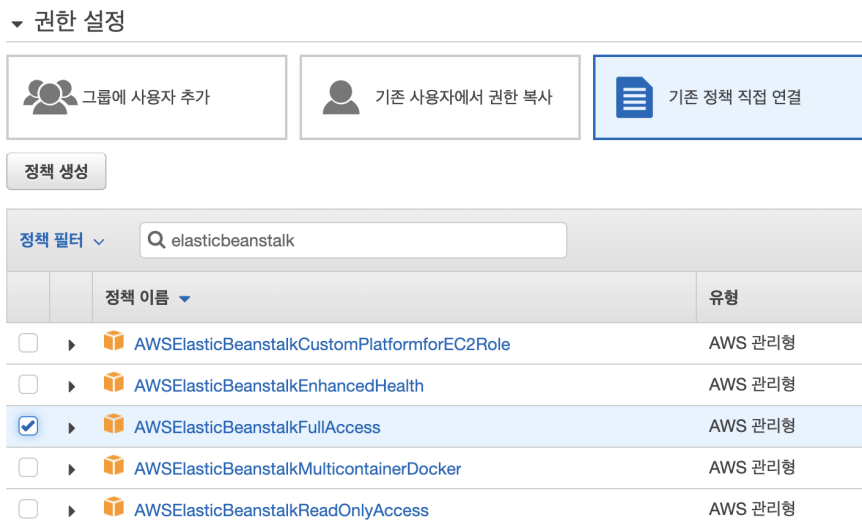
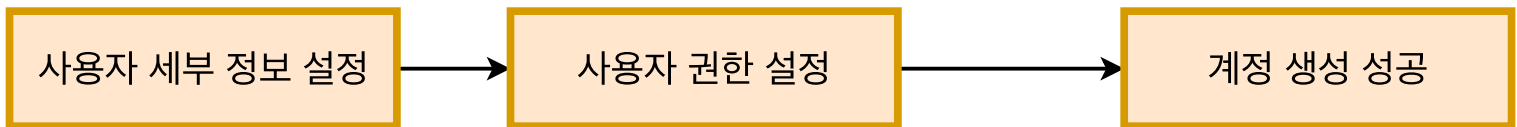
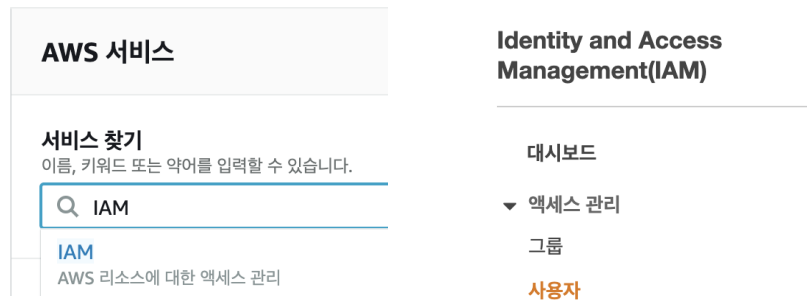
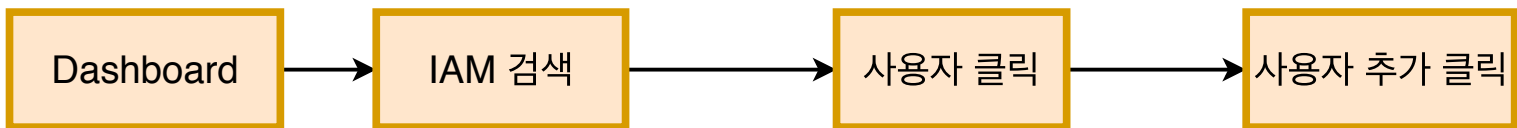


Travis CI의 AWS접근을 위한 API 생성



Travis CI 아이디 로그인시 Github 연동으로 인증을 해줍니다.

AWS에서 제공해주는 **Secret Key**를 Travis yml 파일에다가 적어주면 됩니다.



직접 API키를 Travis yml 파일에
적어 주면 노출이 되기 때문에
다른곳에 적고 그것을 가져와줘야한다.

Travis 웹사이트
해당 저장소
대쉬보드에 오기

설정 클릭

사진과 같이 AWS에서 받은 API 키들을
NAME과 VALUE에 적어서 넣어줍니다.
이곳에 넣어 주면 외부에서 접근을 할 수
없어 더욱 안전합니다.

Travis CI 웹사이트에서 보관중인 Key를
로컬 환경에서 가지고 올수 있게 travis yml 파일
에서
설정을 해줍니다.

```
FROM node:alpine as builder
WORKDIR '/usr/src/app'
COPY package.json .
RUN npm install
COPY ./ ./
RUN npm run build

FROM nginx
EXPOSE 80
COPY --from=builder /usr/src/app/build /usr/share/nginx/html
```