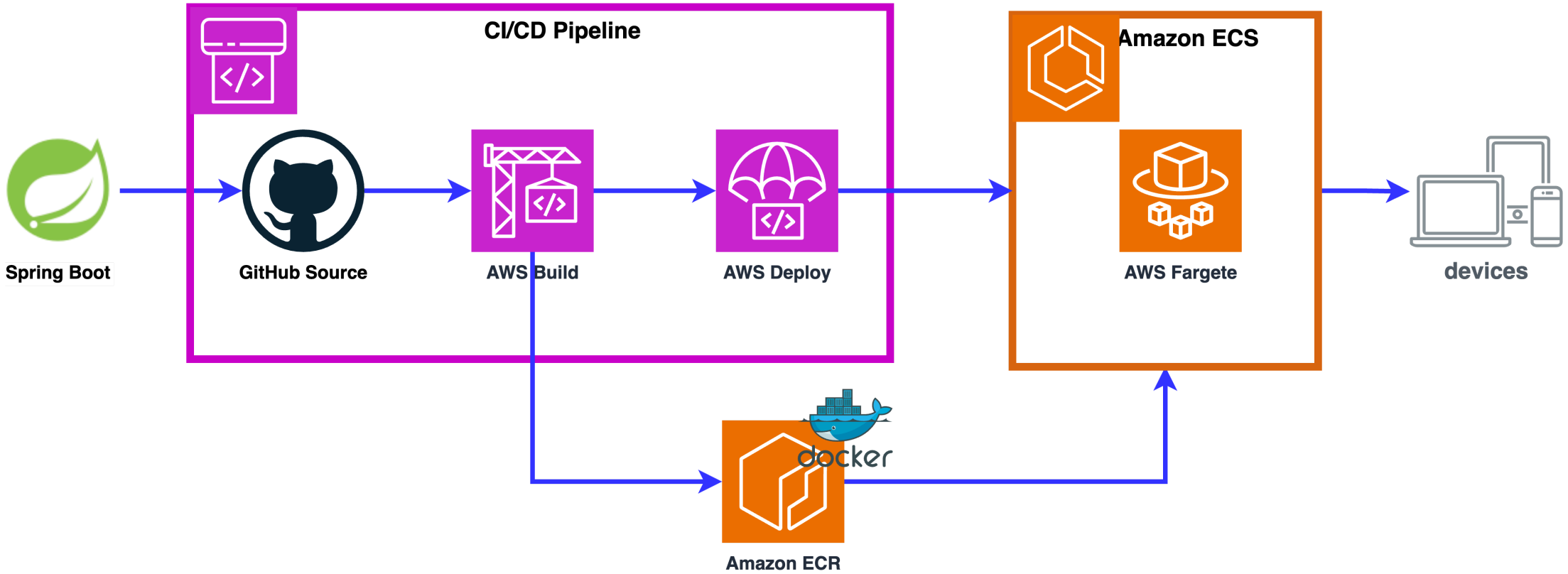
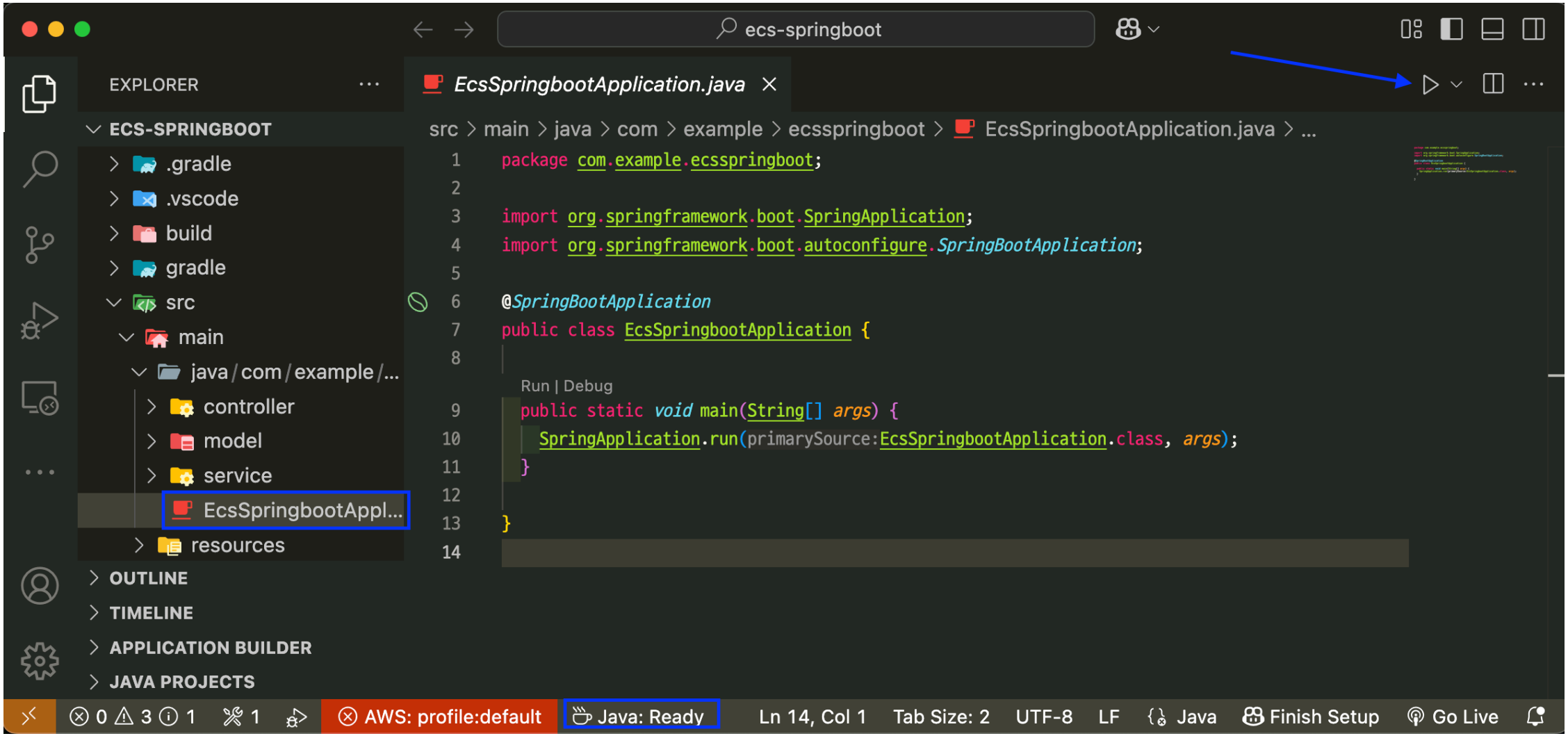
전체 아키텍처

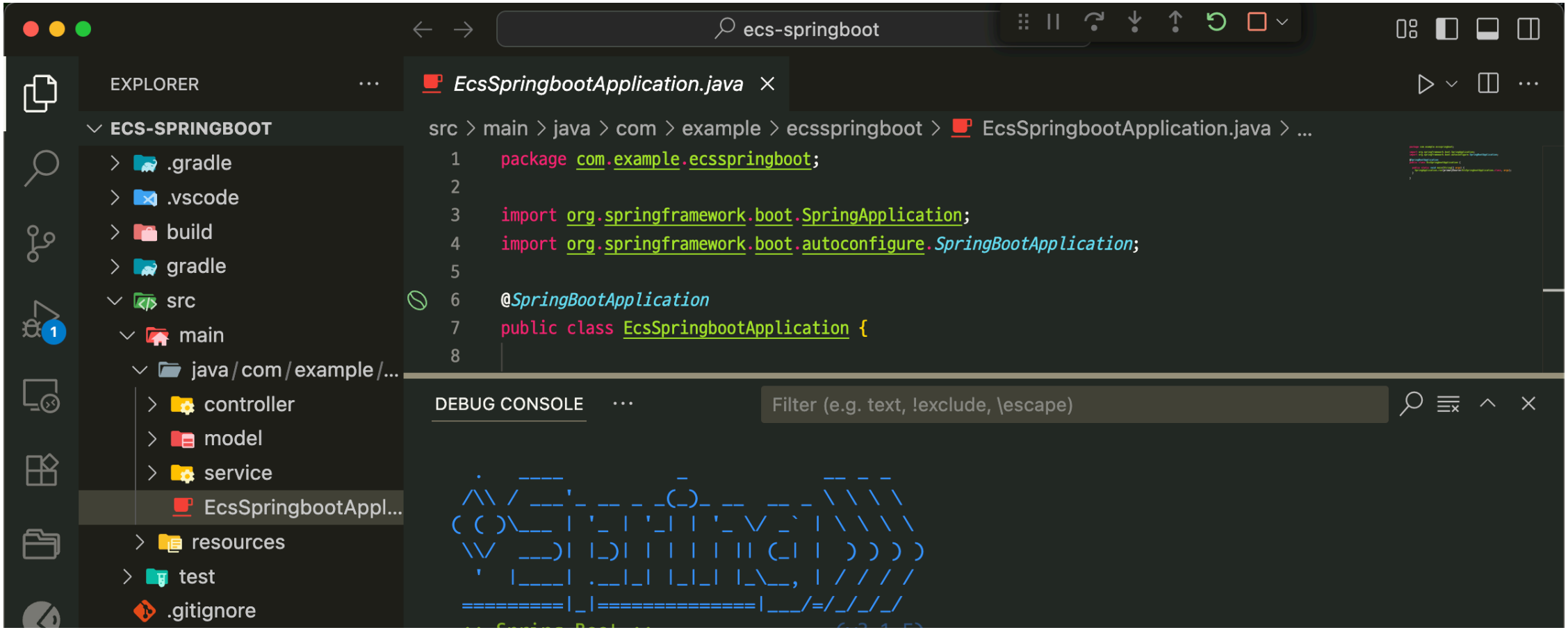

CI/CD Pipeline
Amazon ECS
Spring Boot
GitHub Source
AWS Build
AWS Deploy
AWS Fargete
devices
docker
Amazon ECR

Spring Boot

- Spring Boot는 Java 기반의 웹 애플리케이션을 빠르고 쉽게 개발할 수 있도록 도와주는 프레임워크입니다.

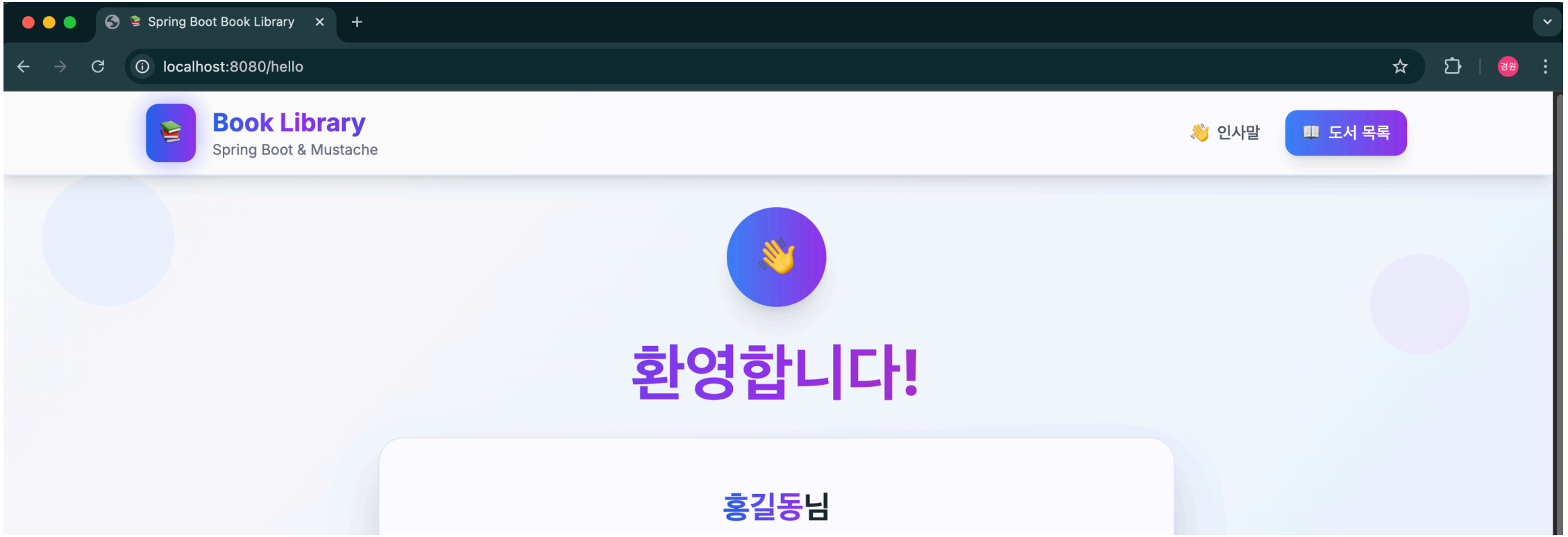
단계1: 프로젝트 실행


ecs-springboot
EXPLORER
EcsSpringbootApplication.java
ECS-SPRINGBOOT
.gradle
.vscode
build
gradle
src
main
java / com / example / ...
controller
model
service
EcsSpringbootAppl...
resources
OUTLINE
TIMELINE
APPLICATION BUILDER
JAVA PROJECTS
src > main > java > com > example > ecsspringboot > EcsSpringbootApplication.java > ...
1 package com.example.ecsspringboot;
2
3 import org.springframework.boot.SpringApplication;
4 import org.springframework.boot.autoconfigure.SpringBootApplication;
5
6 @SpringBootApplication
7 public class EcsSpringbootApplication {
8
Run | Debug
9 public static void main(String[] args) {
10 SpringApplication.run(primarySource:EcsSpringbootApplication.class, args);
11 }
12
13 }
14
0 3 1 1
AWS: profile:default
Java: Ready
Ln 14, Col 1
Tab Size: 2
UTF-8
LF
Java
Finish Setup
Go Live


ecs-springboot
EXPLORER
ECS-SPRINGBOOT
.gradle
.vscode
build
gradle
src
main
java / com / example / ...
controller
model
service
EcsSpringbootAppl...
resources
test
.gitignore
EcsSpringbootApplication.java
src > main > java > com > example > ecsspringboot > EcsSpringbootApplication.java > ...
1 package com.example.ecsspringboot;
2
3 import org.springframework.boot.SpringApplication;
4 import org.springframework.boot.autoconfigure.SpringBootApplication;
5
6 @SpringBootApplication
7 public class EcsSpringbootApplication {
8
DEBUG CONSOLE
Filter (e.g. text, !exclude, \escape)

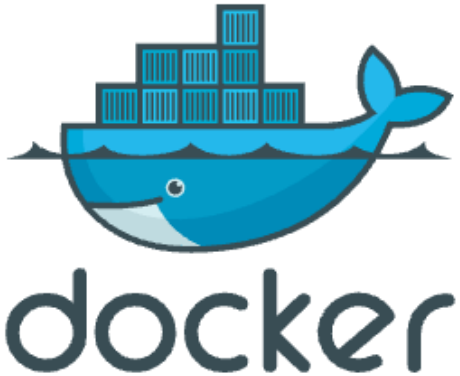
단계2: 프로젝트 테스트

```
http://localhost:8080/hello
```

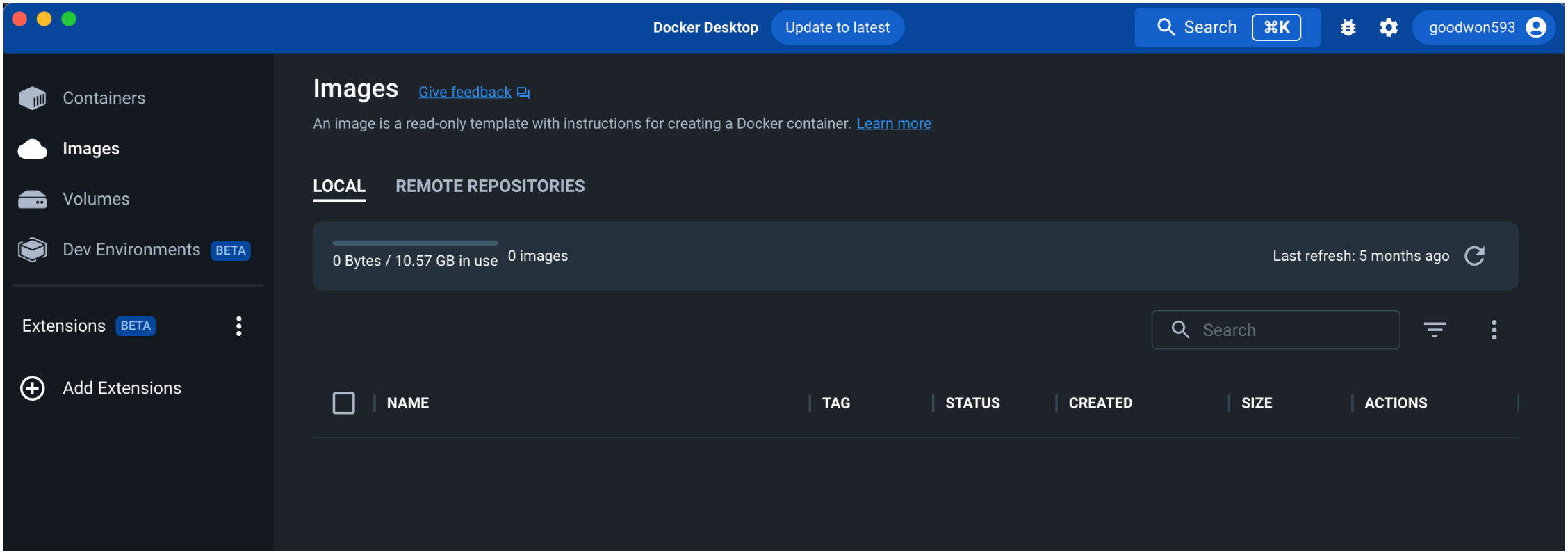


Docker

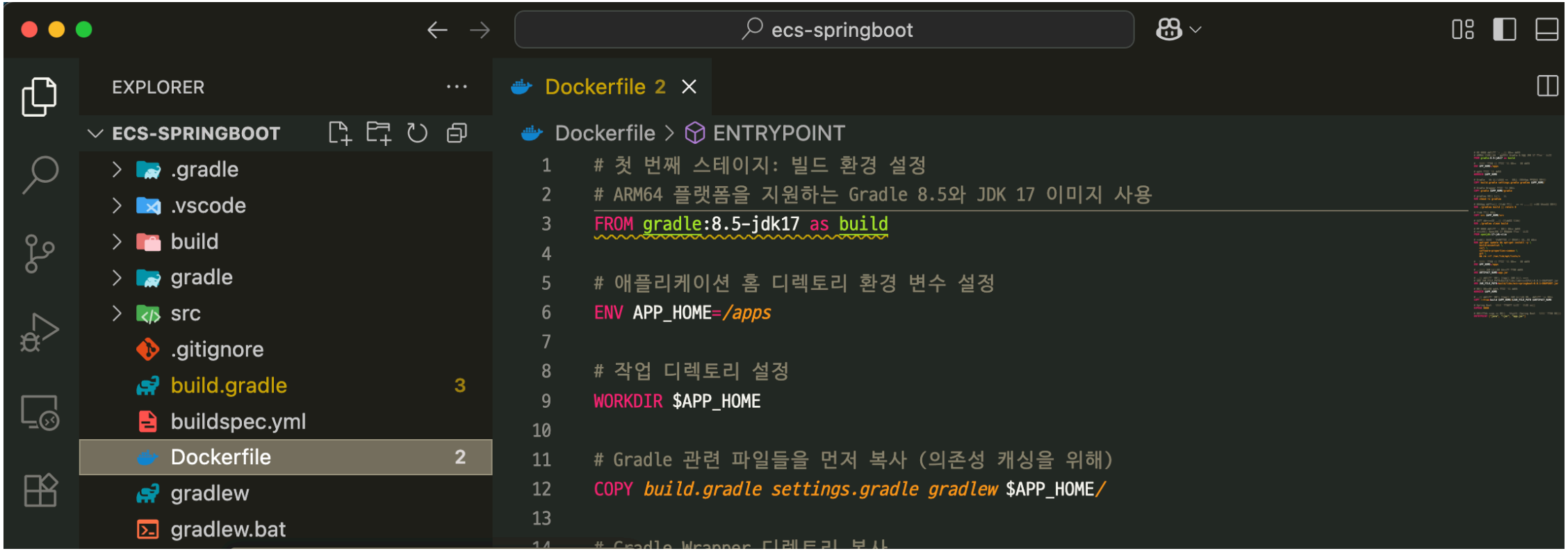
- Docker는 애플리케이션을 컨테이너(container)라는 가상화된 환경에서 실행할 수 있도록 해주는 오픈소스 플랫폼입니다.



단계1: Docker 실행



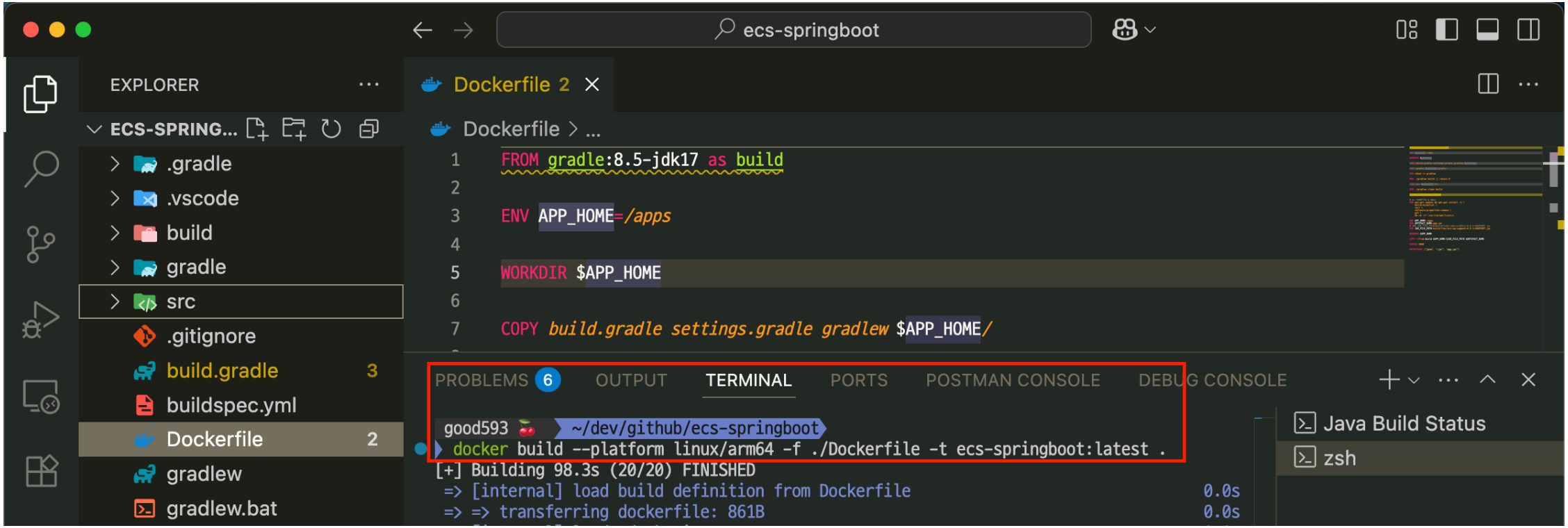
단계2: Dockerfile 확인



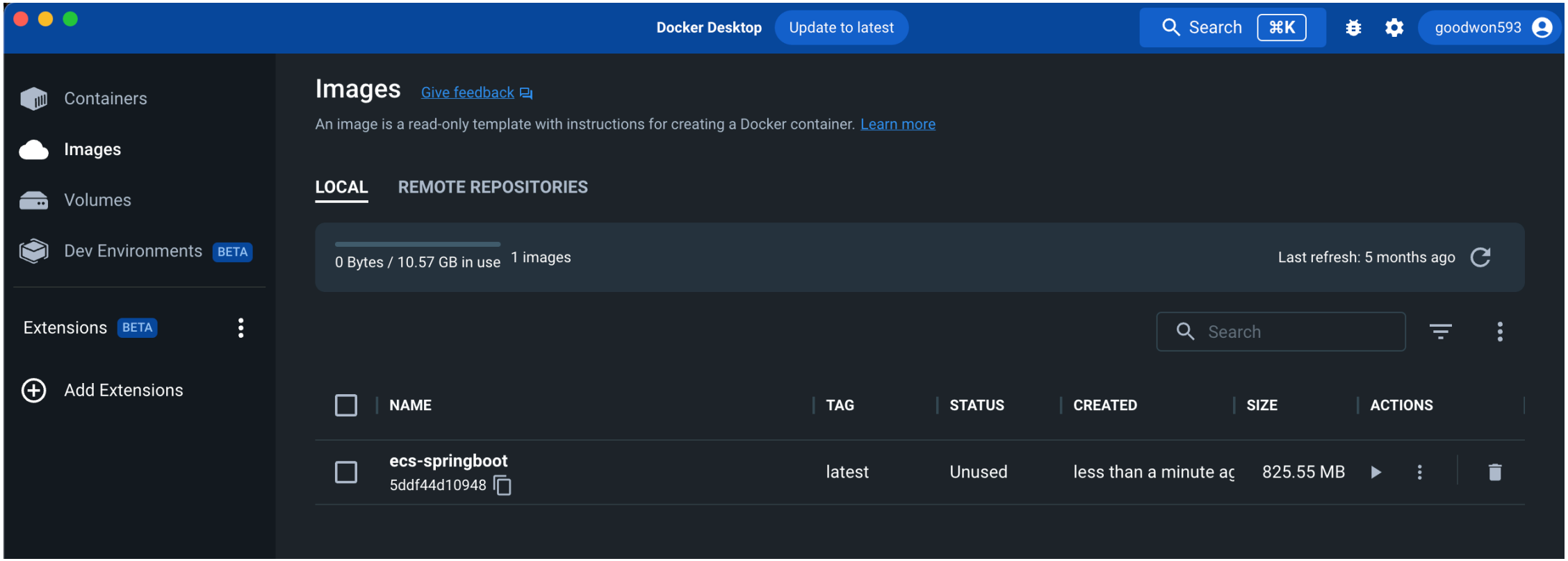
단계3: Docker Iamge 생성

```
# docker image 생성
docker build -f ./Dockerfile -t ecs-springboot:latest .

# ARM 기반 CPU인 경우
docker build --platform linux/arm64 -f ./Dockerfile -t ecs-springboot:latest .
```



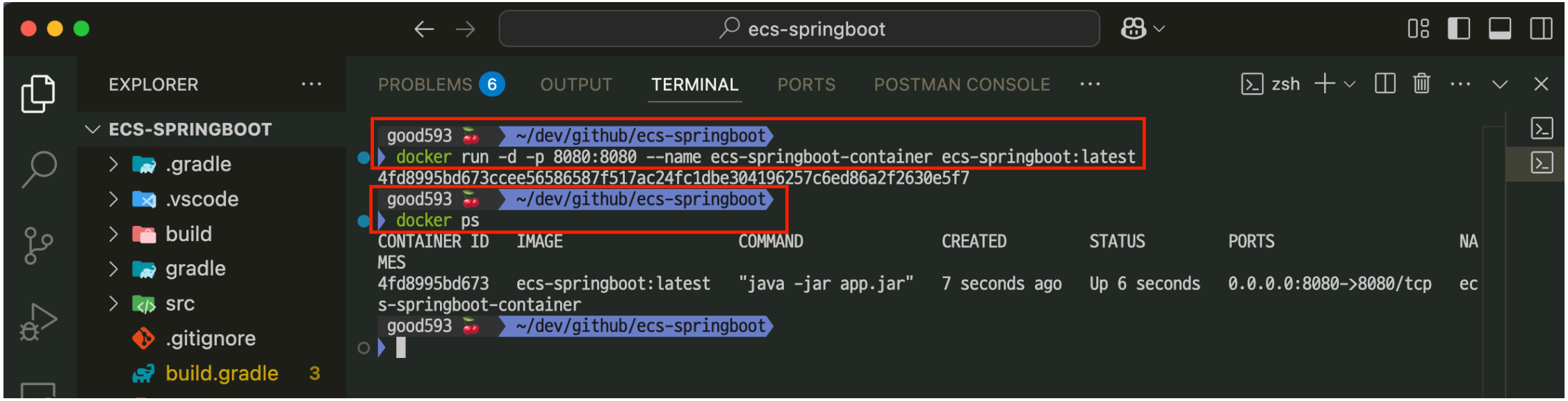
단계4: Docker Iamge 생성 확인



단계5: Docker Container 생성 및 실행

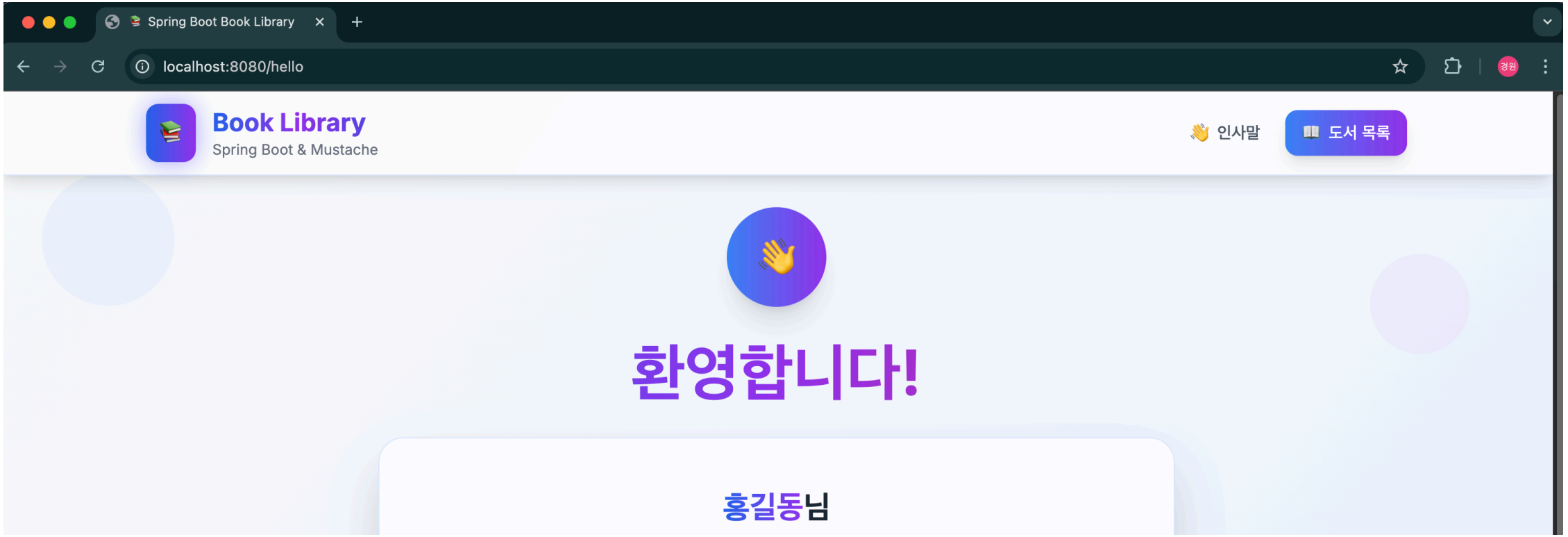
```
# Container 생성 및 실행
docker run -d -p 8080:8080 --name ecs-springboot-container ecs-springboot:latest

# 확인
docker ps
```

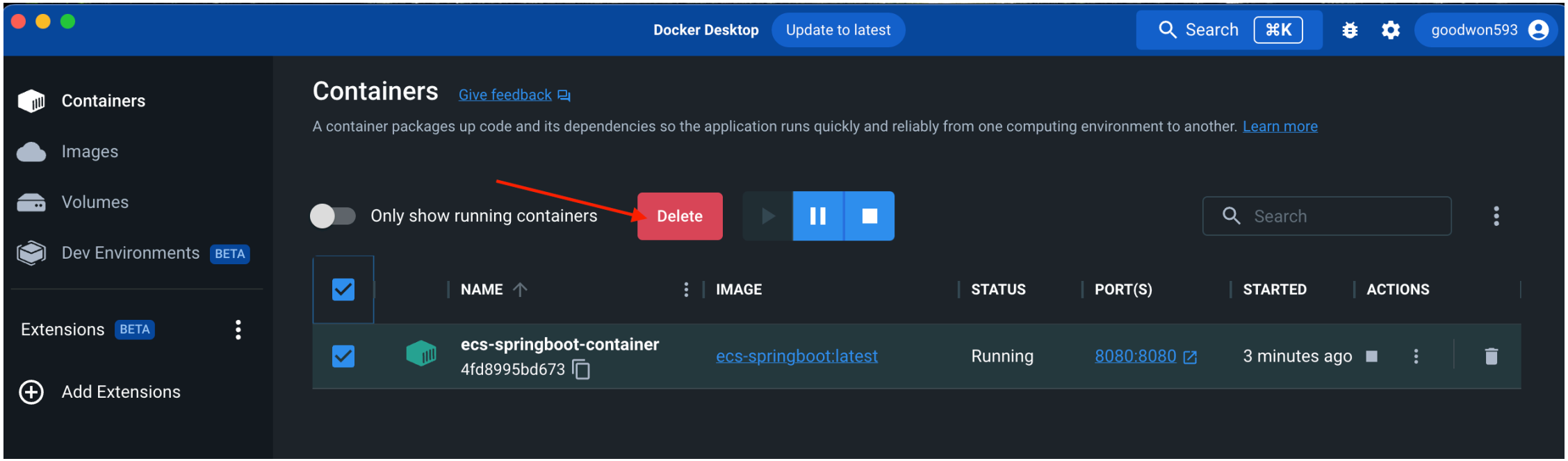


단계6: 테스트

```
http://localhost:8080/hello
```



단계5: Docker Container 삭제

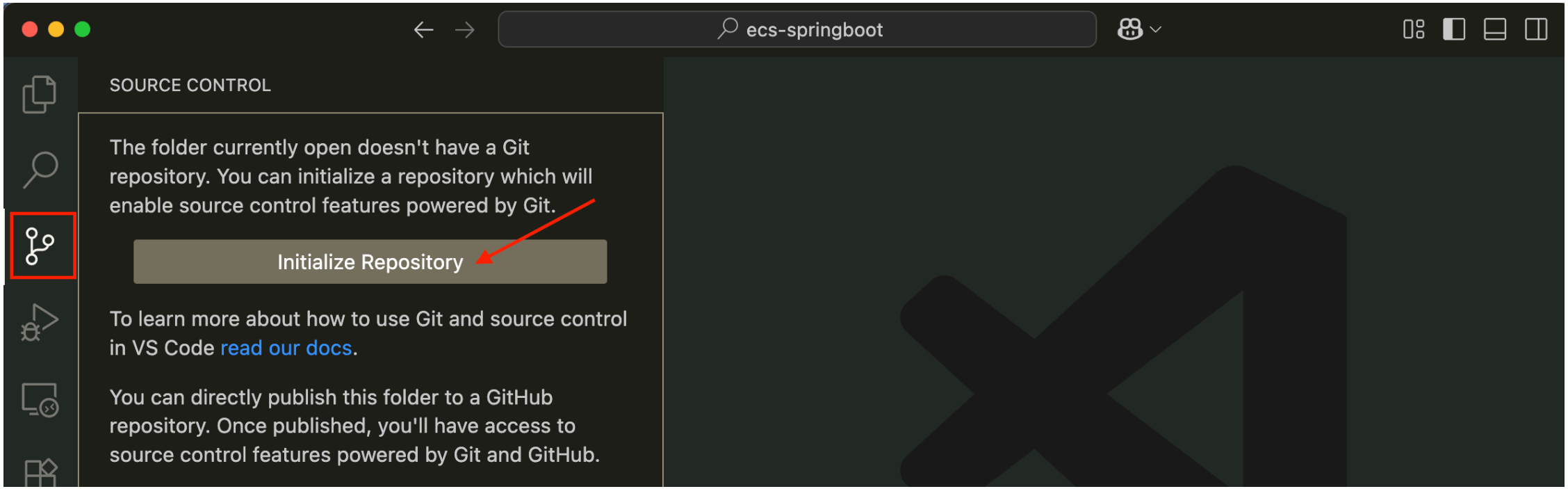


Github

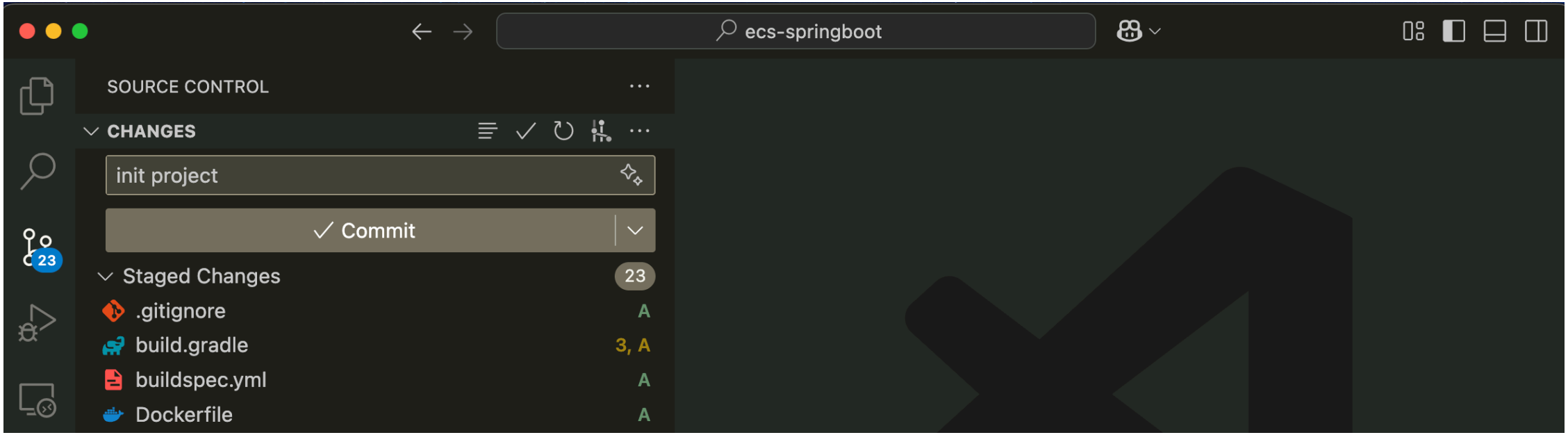
- GitHub는 개발자들이 코드를 저장하고, 협업하며, 버전 관리를 할 수 있는 웹 기반 플랫폼입니다.

GitHub Source

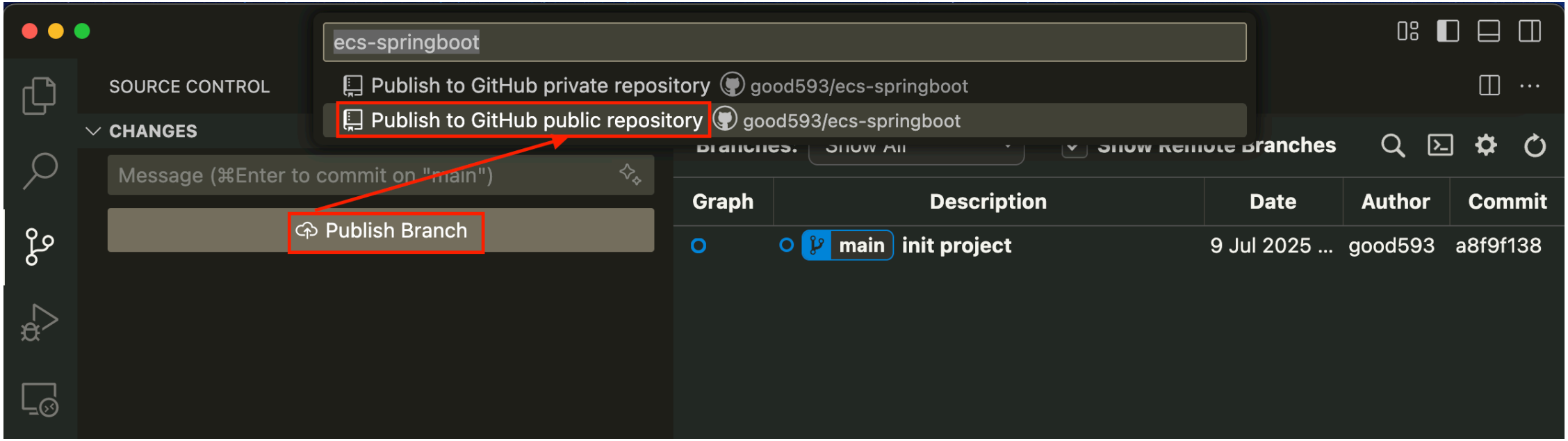
단계1: Initialize Repository



단계2: Commit


ecs-springboot
SOURCE CONTROL
CHANGES
init project
Commit
23
Staged Changes
23
.gitignore
A
build.gradle
3, A
buildspec.yml
A
Dockerfile
A

단계3: Publish Branch



단계4: Github에서 확인