

Dev Containers

CSCI-P 466/566 Software Engineering II

Indiana University Bloomington - Spring 2025

Part I: dev containers in VS Code

Steps

- Complete this short tutorial on creating dev containers in vs code:
- <https://code.visualstudio.com/docs/devcontainers/tutorial>

Part II: Crop Management System

Steps

Project
☐ Gradle - Groovy
☐ Gradle - Kotlin

☒ Maven

Language
☒ Java
☐ Kotlin
☐ Groovy

Spring Boot
☐ 3.5.0 (SNAPSHOT) ☐ 3.5.0 (M3) ☐ 3.4.5 (SNAPSHOT)
☒ 3.4.4 ☐ 3.3.11 (SNAPSHOT) ☐ 3.3.10

Project Metadata

Group

edu.iu.habahram.farm

Artifact

crop-management

Name

crop-management

Description

Crop Management Service

Package name

edu.iu.habahram.farm.crop-management

Packaging

☒ Jar ☐ War

Java

☐ 24 ☐ 21 ☒ 17

Dependencies

ADD DEPENDENCIES... ⌘ + B

Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

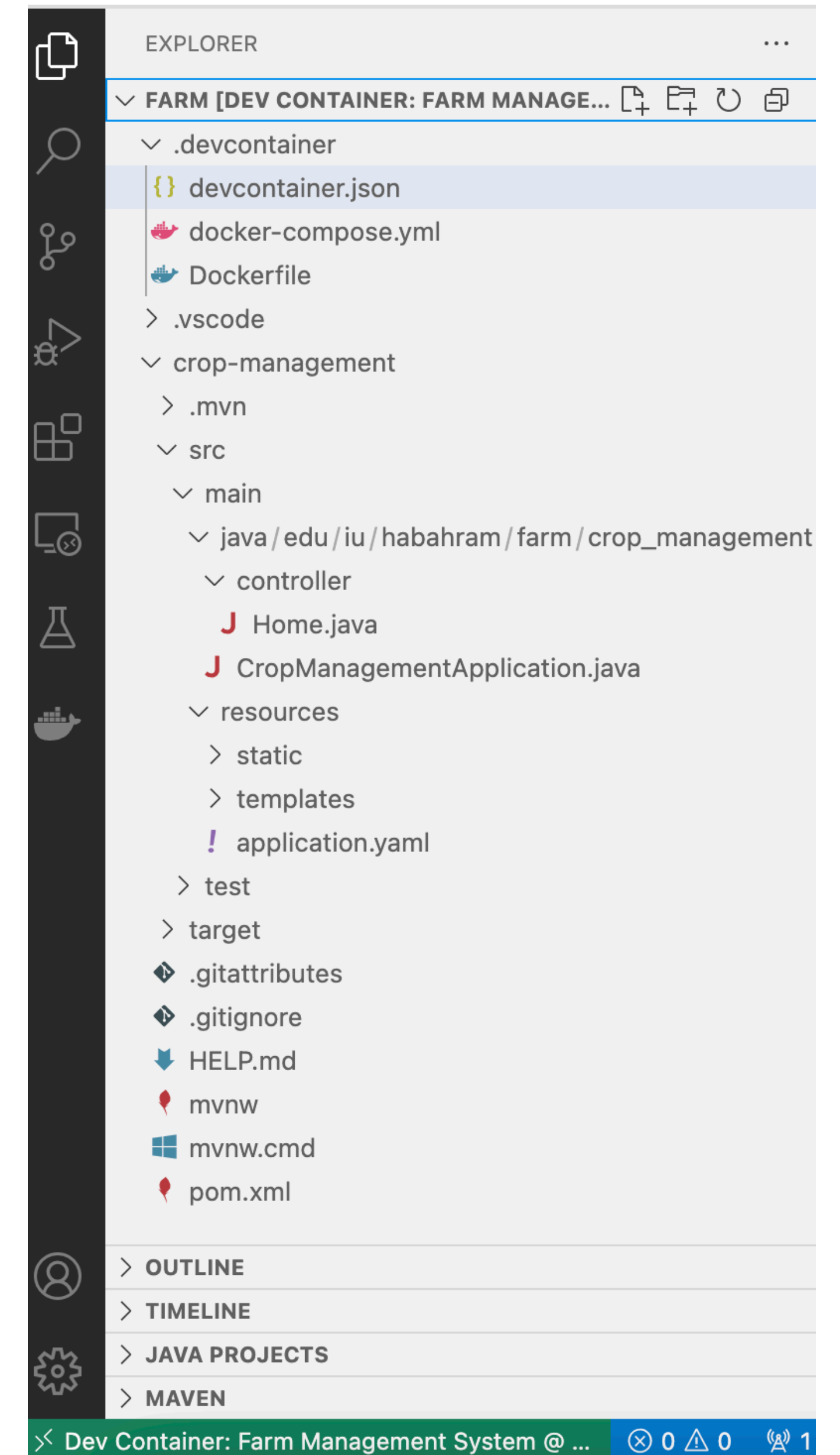
Spring Data JPA SQL
Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

PostgreSQL Driver SQL
A JDBC and R2DBC driver that allows Java programs to connect to a PostgreSQL database using standard, database independent Java code.

Part II: Crop Management System in dev container

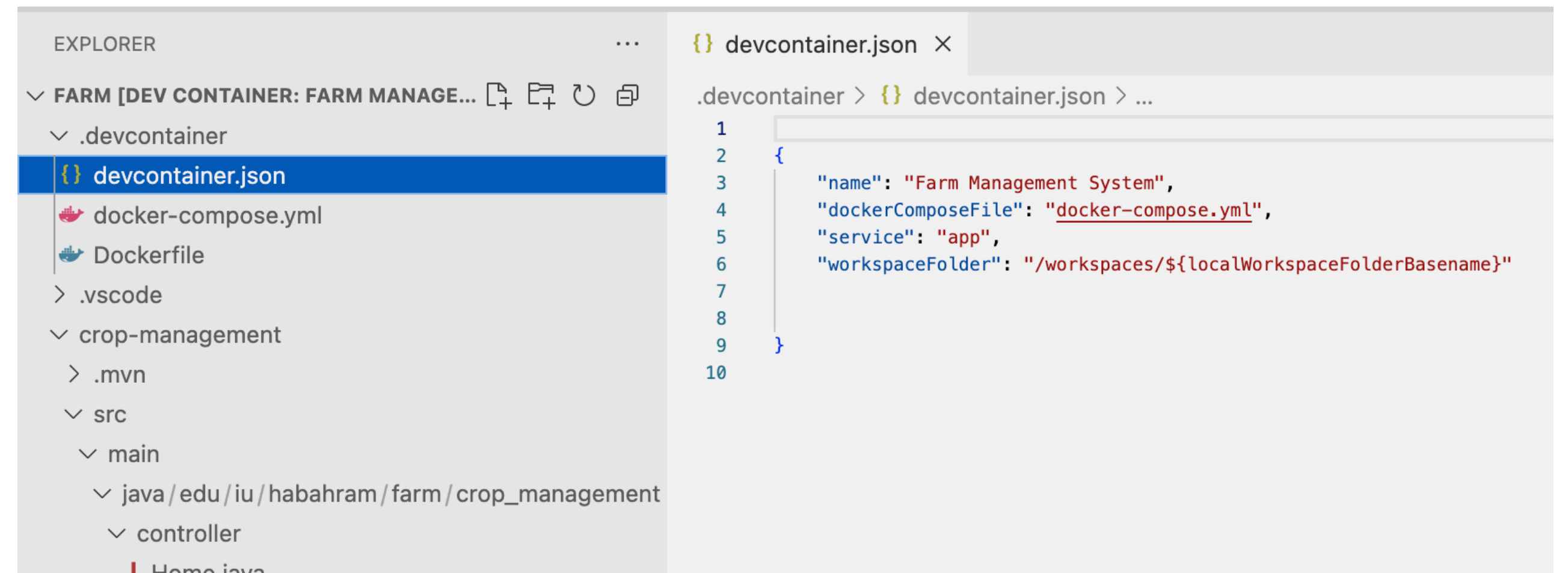
Steps

- Create a folder called “farm” and put your crop management folder inside the farm folder.
- Create a .devcontainer folder at the root of the farm folder with the following files:



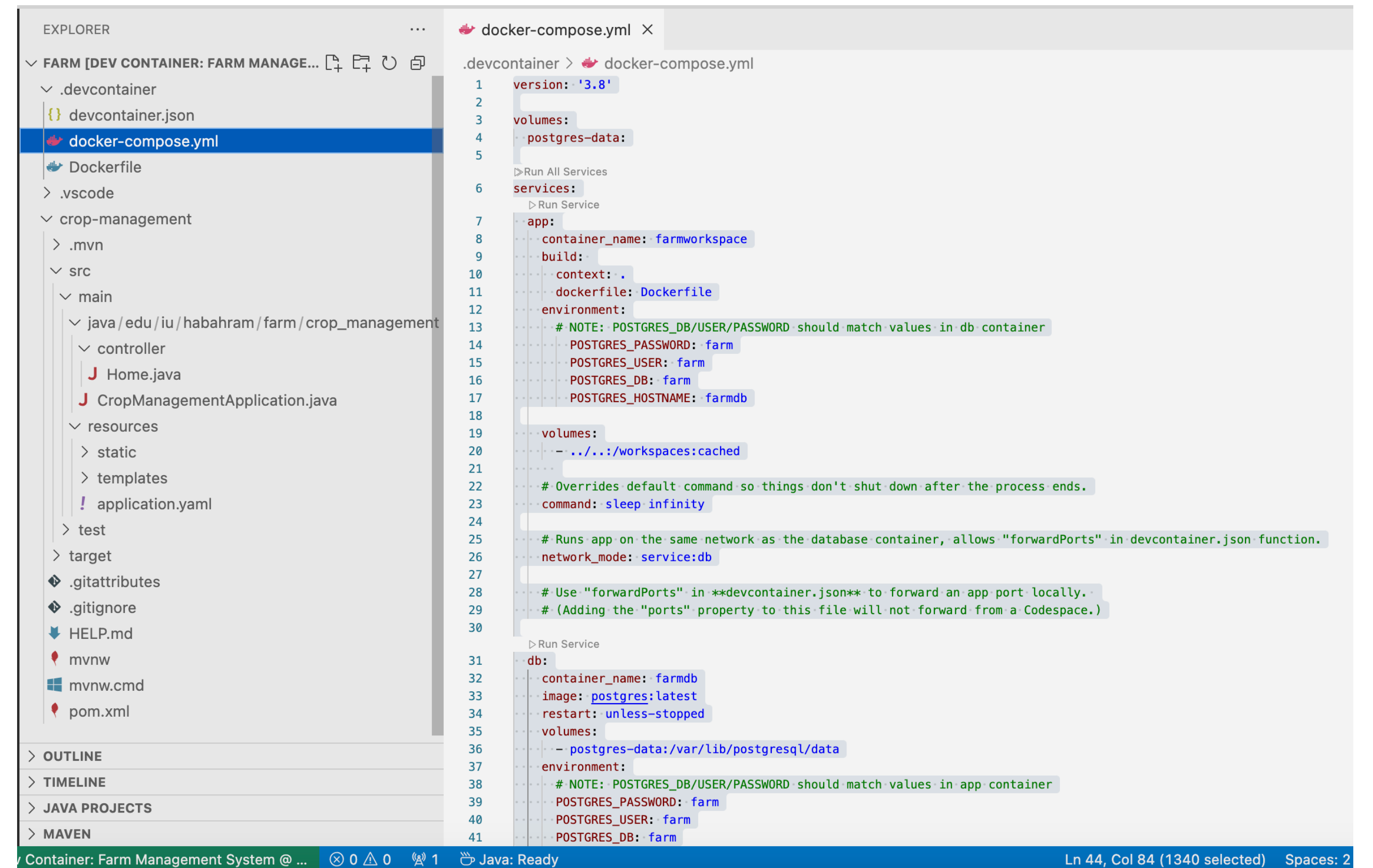
Steps

- devcontainer.json:



Steps

- Docker-compose.yml:
- <https://gist.github.com/hbahramian/3264790dc708ff9eead375854cebcb57>

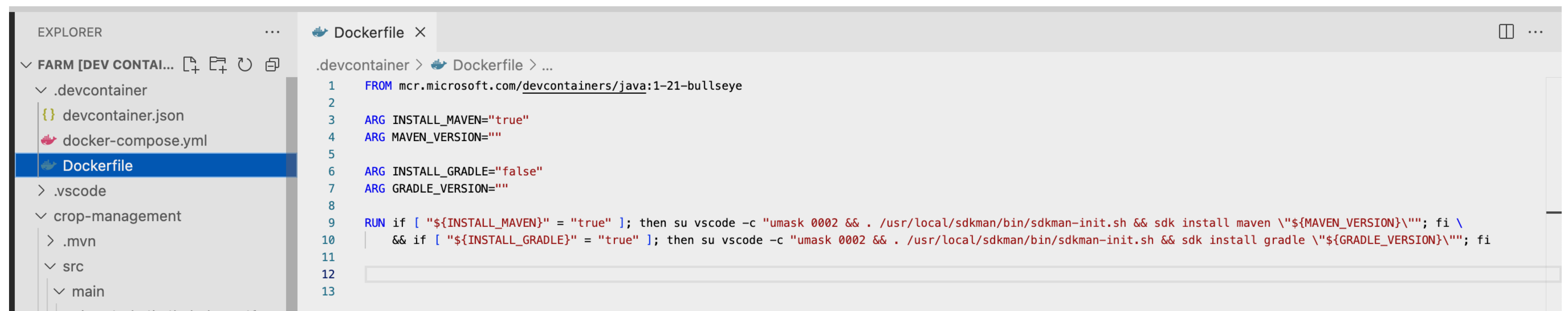


The screenshot shows a VS Code editor with a project named 'FARM [DEV CONTAINER: FARM MANAGE...]' open. The Explorer sidebar on the left shows the project structure, including a 'devcontainer' folder with 'devcontainer.json' and 'docker-compose.yml', and a 'crop-management' folder with 'Dockerfile', '.vscode', and 'src'. The 'docker-compose.yml' file is selected and its content is displayed in the main editor. The configuration defines two services: 'app' and 'db'. The 'app' service uses the 'farmworkspace' container name, builds from the local 'Dockerfile', and sets environment variables for PostgreSQL. The 'db' service uses the 'farmdb' container name, the 'postgres:latest' image, and shares the 'postgres-data' volume. Both services are connected to a common network named 'service:db'.

```
1 version: '3.8'
2
3 volumes:
4   postgres-data:
5
6 > Run All Services
7 services:
8   > Run Service
9   app:
10     container_name: farmworkspace
11     build:
12       context: .
13       dockerfile: Dockerfile
14     environment:
15       # NOTE: POSTGRES_DB/USER/PASSWORD should match values in db container
16       POSTGRES_PASSWORD: farm
17       POSTGRES_USER: farm
18       POSTGRES_DB: farm
19       POSTGRES_HOSTNAME: farmdb
20
21     volumes:
22       - ../../workspaces/ cached
23
24     # Overrides default command so things don't shut down after the process ends.
25     command: sleep infinity
26
27     # Runs app on the same network as the database container, allows "forwardPorts" in devcontainer.json function.
28     network_mode: service:db
29
30     # Use "forwardPorts" in **devcontainer.json** to forward an app port locally.
31     # (Adding the "ports" property to this file will not forward from a Codespace.)
32
33   > Run Service
34   db:
35     container_name: farmdb
36     image: postgres:latest
37     restart: unless-stopped
38     volumes:
39       - postgres-data: /var/lib/postgresql/data
40     environment:
41       # NOTE: POSTGRES_DB/USER/PASSWORD should match values in app container
42       POSTGRES_PASSWORD: farm
43       POSTGRES_USER: farm
44       POSTGRES_DB: farm
```

Steps

- Dockerfile:
- <https://gist.github.com/hbahramian/49525719cda44e7e75d2405fb0021cae>

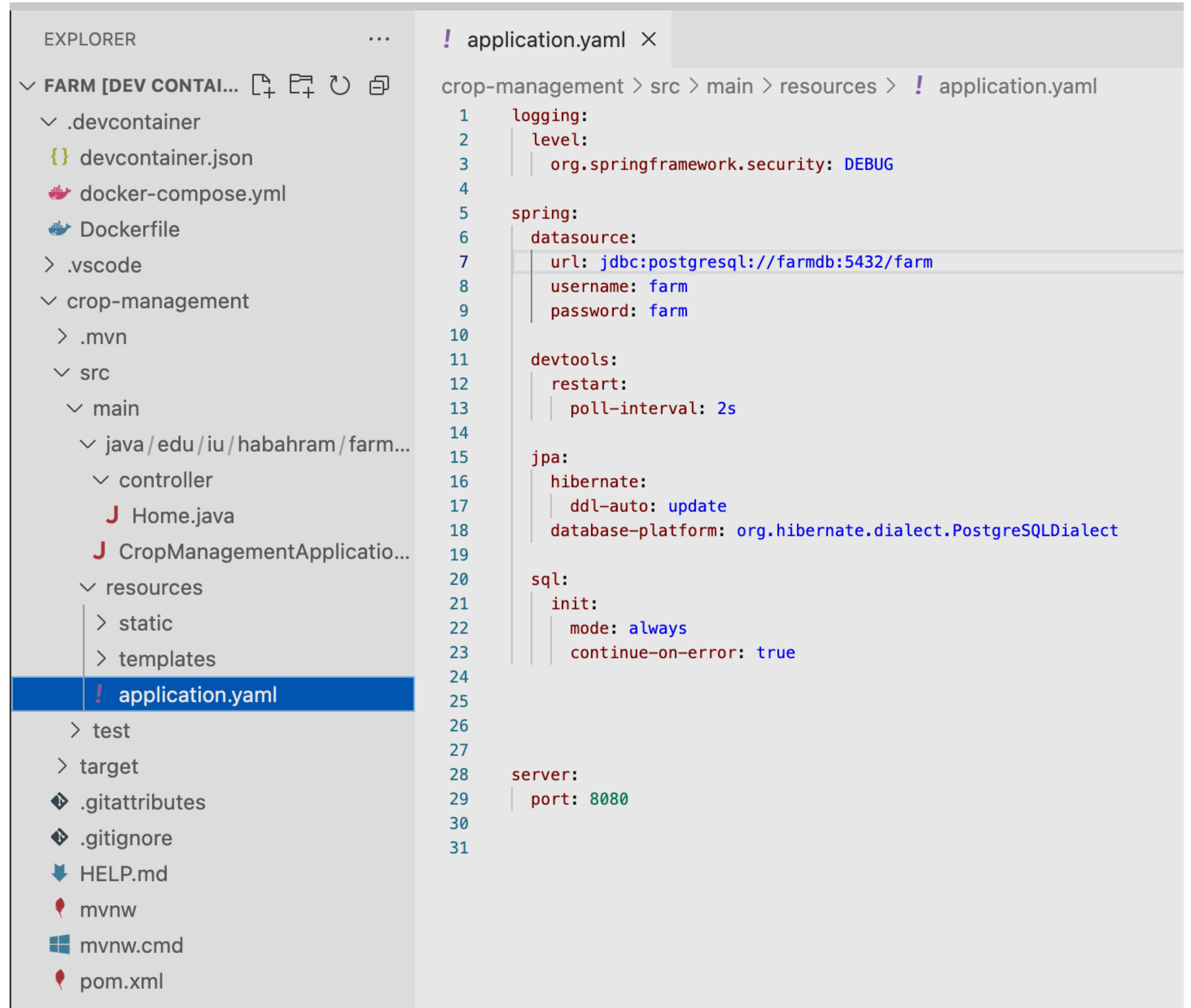


The screenshot shows a VS Code editor interface with a Dockerfile open in a devcontainer. The Explorer sidebar on the left shows the file structure of the devcontainer, including .devcontainer, .vscode, and crop-management. The Dockerfile content is as follows:

```
1 FROM mcr.microsoft.com/devcontainers/java:1-21-bullseye
2
3 ARG INSTALL_MAVEN="true"
4 ARG MAVEN_VERSION=""
5
6 ARG INSTALL_GRADLE="false"
7 ARG GRADLE_VERSION=""
8
9 RUN if [ "${INSTALL_MAVEN}" = "true" ]; then su vscode -c "umask 0002 && . /usr/local/sdkman/bin/sdkman-init.sh && sdk install maven \"${MAVEN_VERSION}\""; fi \
10 | && if [ "${INSTALL_GRADLE}" = "true" ]; then su vscode -c "umask 0002 && . /usr/local/sdkman/bin/sdkman-init.sh && sdk install gradle \"${GRADLE_VERSION}\""; fi
11
12
13
```

Steps

- application.yaml:

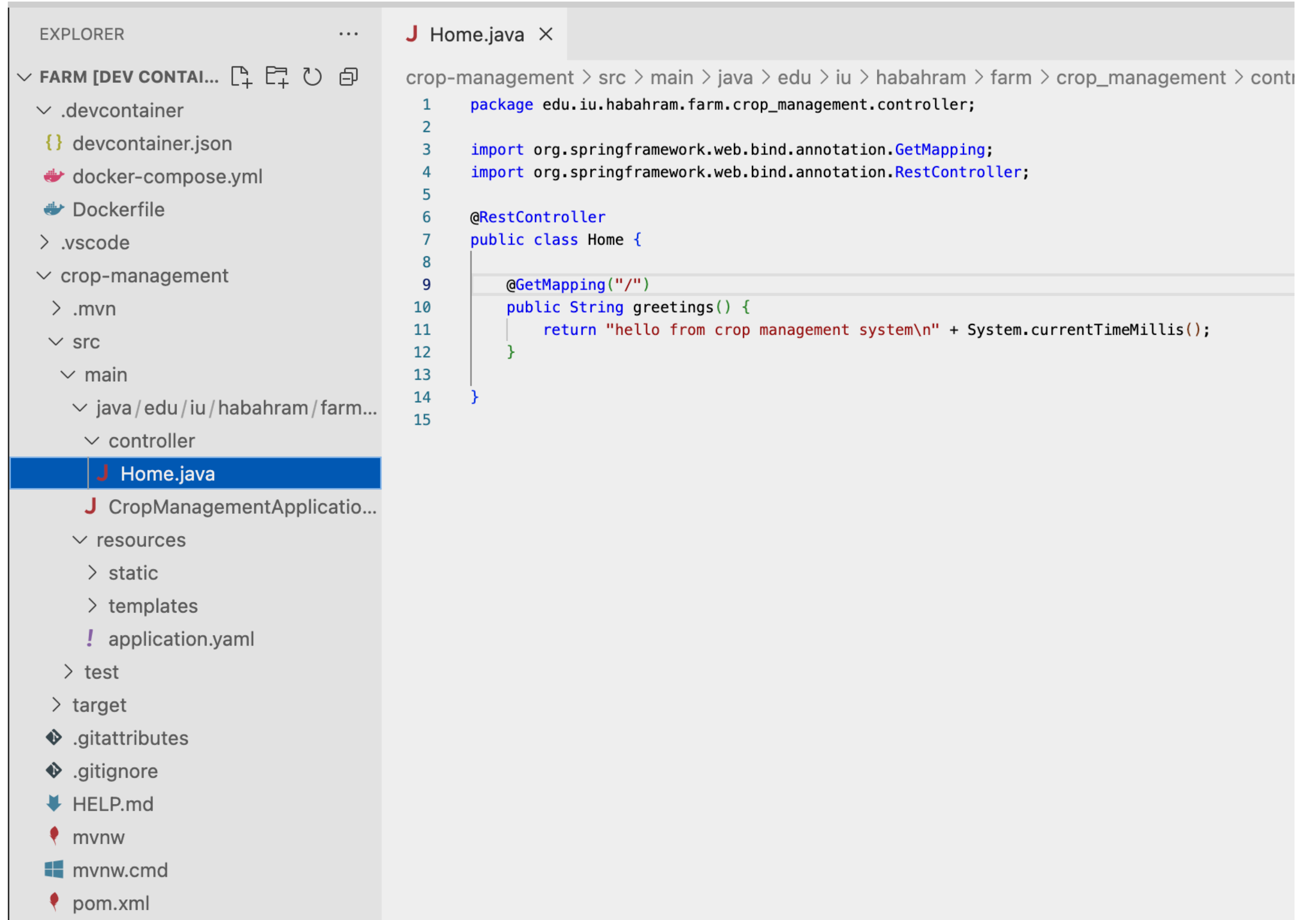


```
EXPLORER
└─ FARM [DEV CONTAI...
  └─ .devcontainer
    ├── devcontainer.json
    ├── docker-compose.yml
    └─ Dockerfile
  └─ .vscode
  └─ crop-management
    └─ .mvn
    └─ src
      └─ main
        └─ java/edu/iu/habahram/farm...
          └─ controller
            ├── Home.java
            └─ CropManagementApplicatio...
          └─ resources
            ├── static
            ├── templates
            └─ ! application.yaml
          └─ test
          └─ target
          └─ .gitattributes
          └─ .gitignore
          └─ HELP.md
          └─ mvnw
          └─ mvnw.cmd
          └─ pom.xml

! application.yaml ×
crop-management > src > main > resources > ! application.yaml
1  logging:
2    level:
3      org.springframework.security: DEBUG
4
5  spring:
6    datasource:
7      url: jdbc:postgresql://farmdb:5432/farm
8      username: farm
9      password: farm
10
11   devtools:
12     restart:
13       poll-interval: 2s
14
15   jpa:
16     hibernate:
17       ddl-auto: update
18     database-platform: org.hibernate.dialect.PostgreSQLDialect
19
20   sql:
21     init:
22       mode: always
23       continue-on-error: true
24
25
26
27
28   server:
29     port: 8080
30
31
```

Steps

- Home controller:



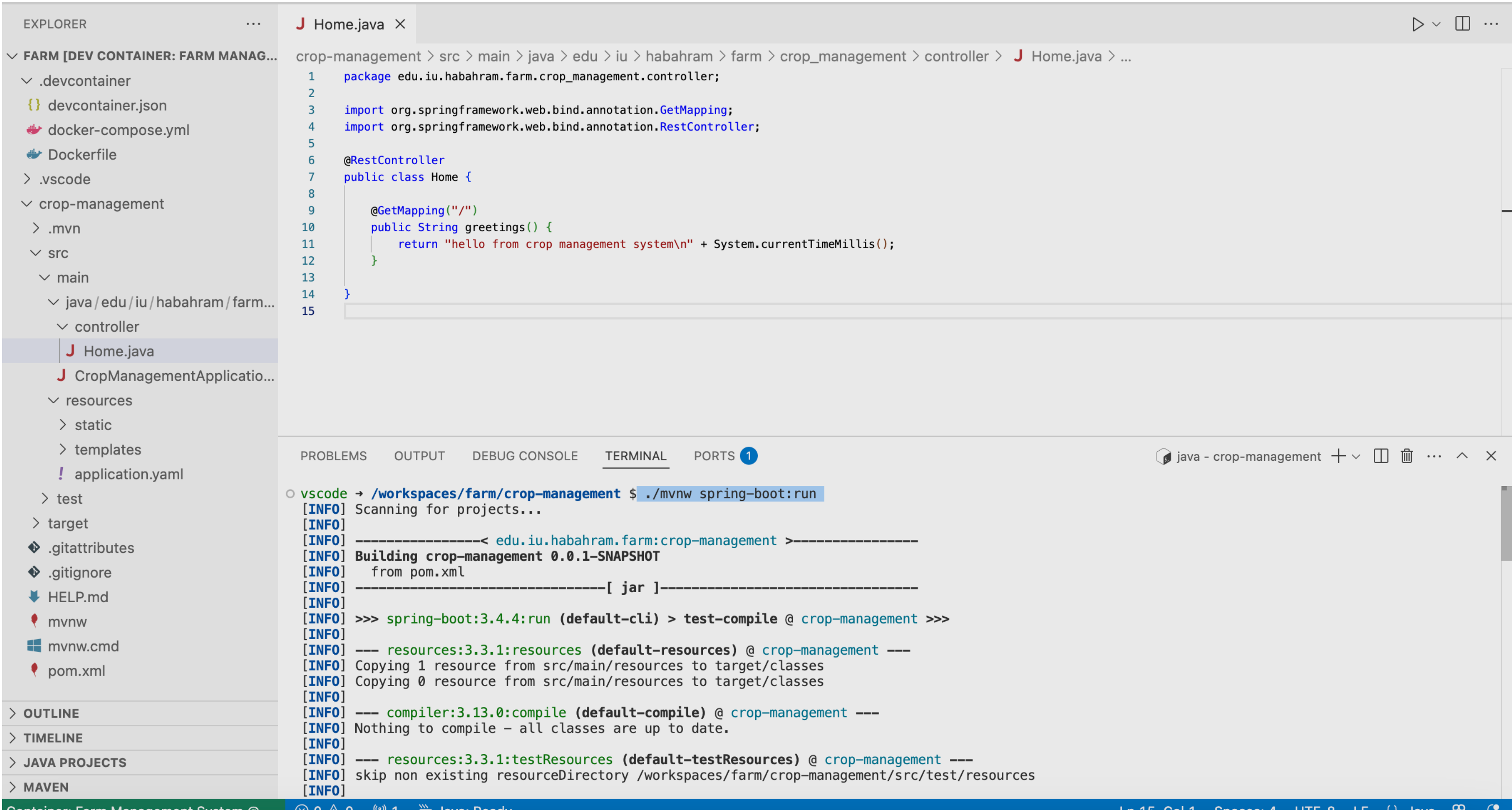
```
EXPLORER
...
FARM [DEV CONTAI...
├── .devcontainer
│   ├── devcontainer.json
│   ├── docker-compose.yml
│   └── Dockerfile
├── .vscode
├── crop-management
│   ├── .mvn
│   ├── src
│   │   └── main
│   │       ├── java/edu/iu/habahram/farm...
│   │       └── controller
│   └── Home.java
├── CropManagementApplicatio...
├── resources
│   ├── static
│   ├── templates
│   └── application.yml
├── test
├── target
├── .gitattributes
├── .gitignore
├── HELP.md
├── mvnw
├── mvnw.cmd
└── pom.xml

J Home.java X
crop-management > src > main > java > edu > iu > habahram > farm > crop_management > controller

1 package edu.iu.habahram.farm.crop_management.controller;
2
3 import org.springframework.web.bind.annotation.GetMapping;
4 import org.springframework.web.bind.annotation.RestController;
5
6 @RestController
7 public class Home {
8
9     @GetMapping("/")
10    public String greetings() {
11        return "hello from crop management system\n" + System.currentTimeMillis();
12    }
13
14 }
15
```

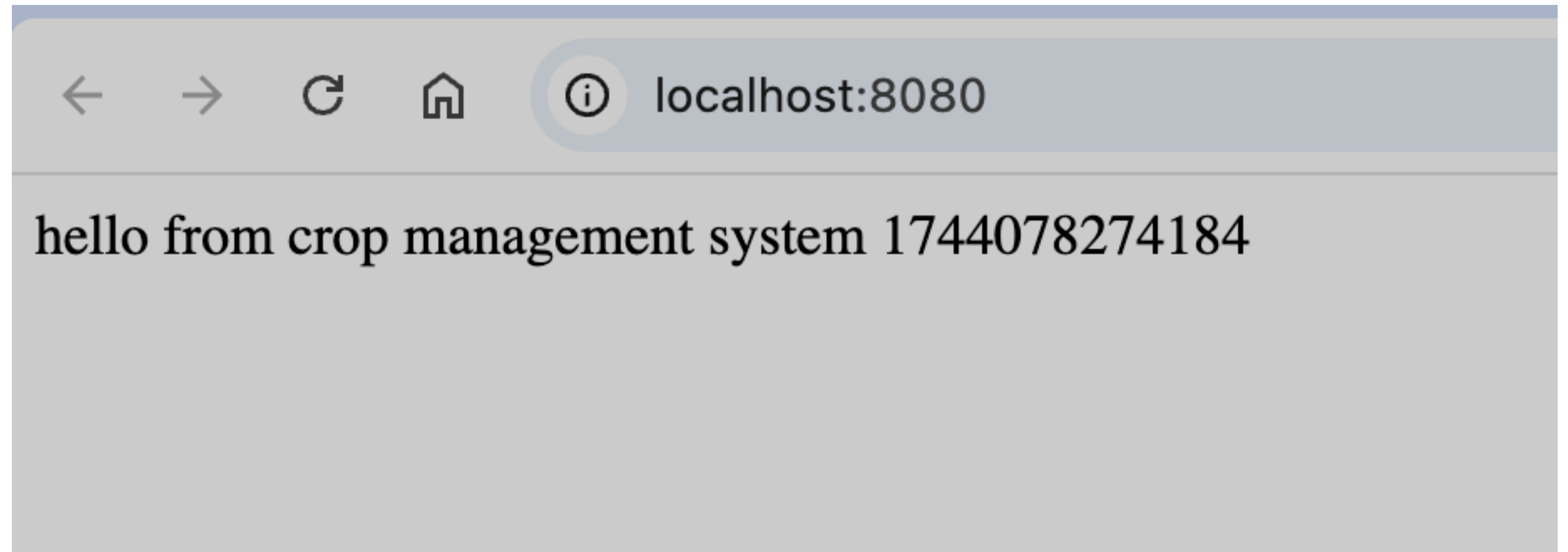
Steps

- Run the application:



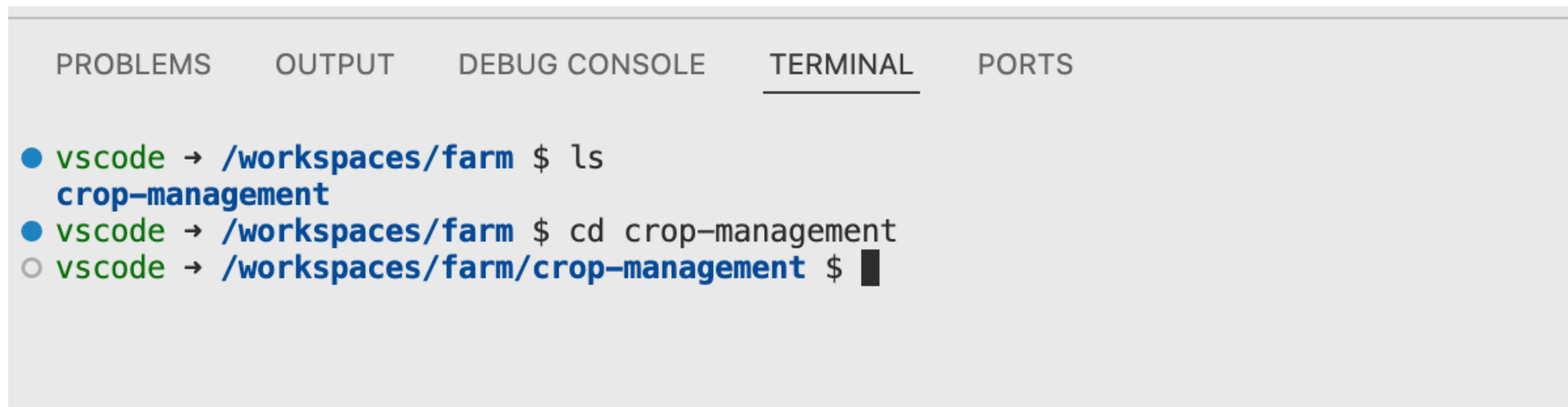
Steps

- Verify:



Steps

- In your GitHub p566/466 organization, create a repository called “crop-management”.
- Then in vs code terminal change directory to crop-management folder:

A screenshot of the Visual Studio Code interface showing the terminal panel. The terminal has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is selected and underlined), and PORTS. The terminal output shows three commands being executed in a shell:

- A blue dot icon followed by `vscode` → `/workspaces/farm` \$ `ls`, with the output `crop-management` displayed below it.
- A blue dot icon followed by `vscode` → `/workspaces/farm` \$ `cd crop-management`.
- A grey dot icon followed by `vscode` → `/workspaces/farm/crop-management` \$, with a black cursor block at the end.

Steps

- Then run the following commands:
- **git init**
- **git remote add origin https://github.com/YOUR-ORGANIZATION/crop-management.git**
- **git add .**
- **git commit -m "greetings endpoint added"**
- **git push --set-upstream origin master**

THE END

- Submit the url of your crop-management to canvas.