TITLE:

SHOPPING LIST APPLICATION – BACKEND

SUBTITLE:

SPRING BOOT REST API WITH DOCKER & RENDER DEPLOYMENT

III-BCA
23SUCA50

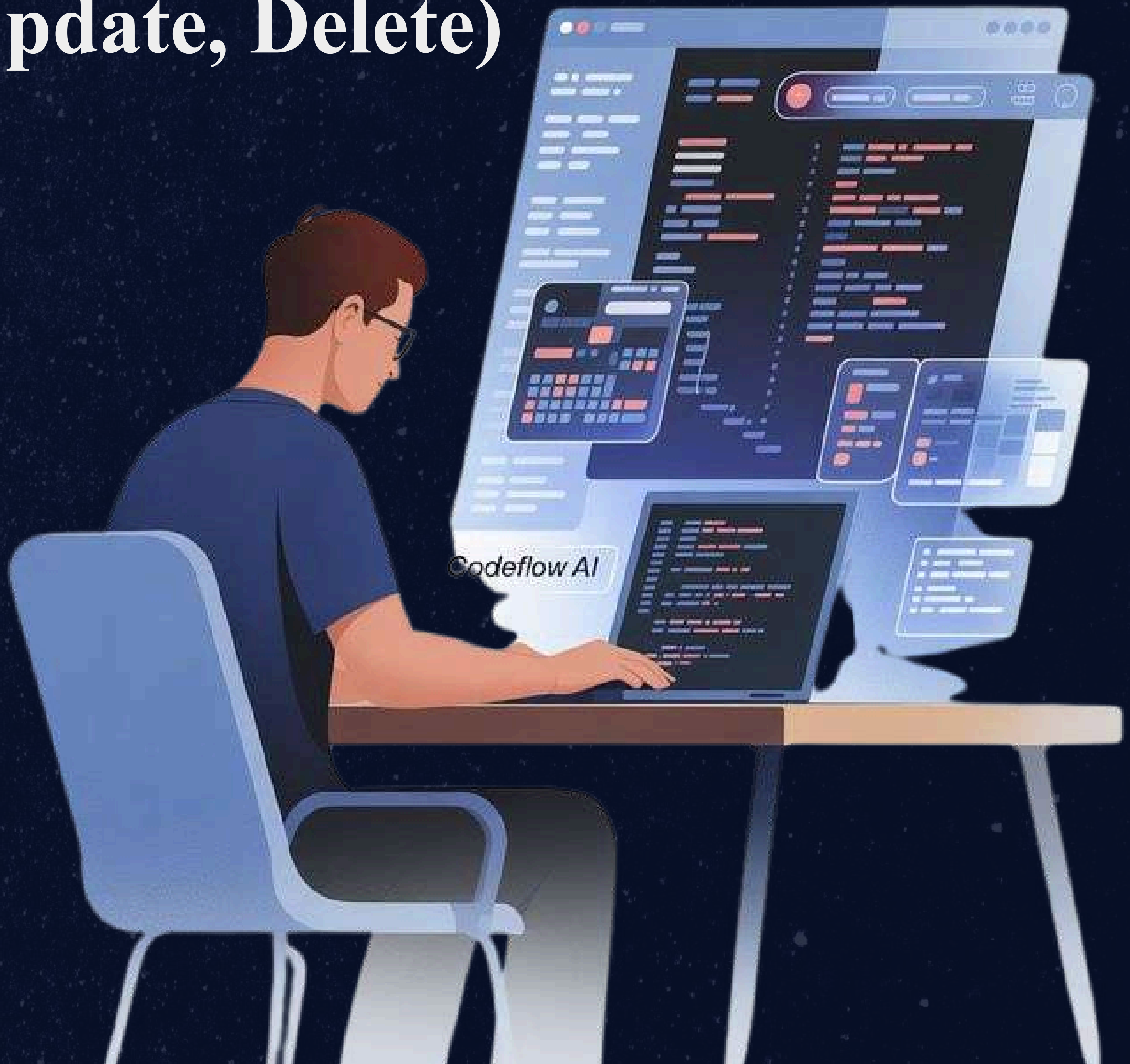
Backend:

The project follows modern DevOps practices, including:

- **Frontend and backend separation**
- **REST API communication**
- **Dockerization**
- **CI/CD using GitHub Actions**
- **Code quality analysis using Sonar**
- **Cloud deployment using Vercel (frontend) and Render (backend)**

How the Project Works (Flow):

- Backend (Spring Boot)
- Exposes REST APIs (/api/items)
- Stores shopping tems in H2 in-memory database
- Handles CRUD operations (Create, Read, Update, Delete)

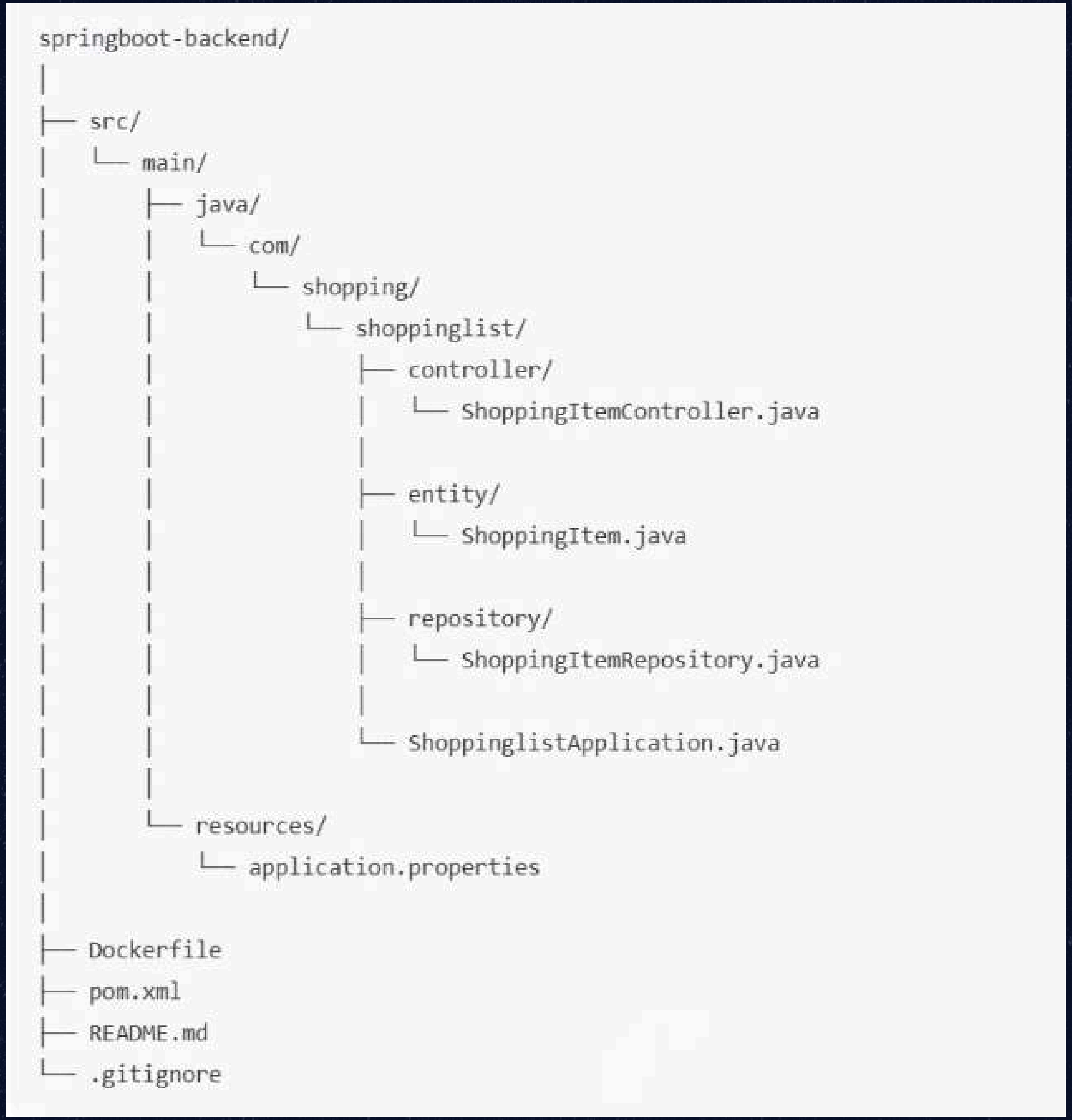


Backend Description (Spring Boot)

- Built using Spring Boot
- RESTful API design
- Uses Spring Data JPA
- H2 in-memory database
- Dockerized for cloud deployment
- Backend Features
- POST /api/items → Add item
- GET /api/items → Get all items
- PUT /api/items/{id} → Update item
- DELETE /api/items/{id} → Delete item
- DELETE /api/items → Delete all items

FOLDER STRUCTUTRE:

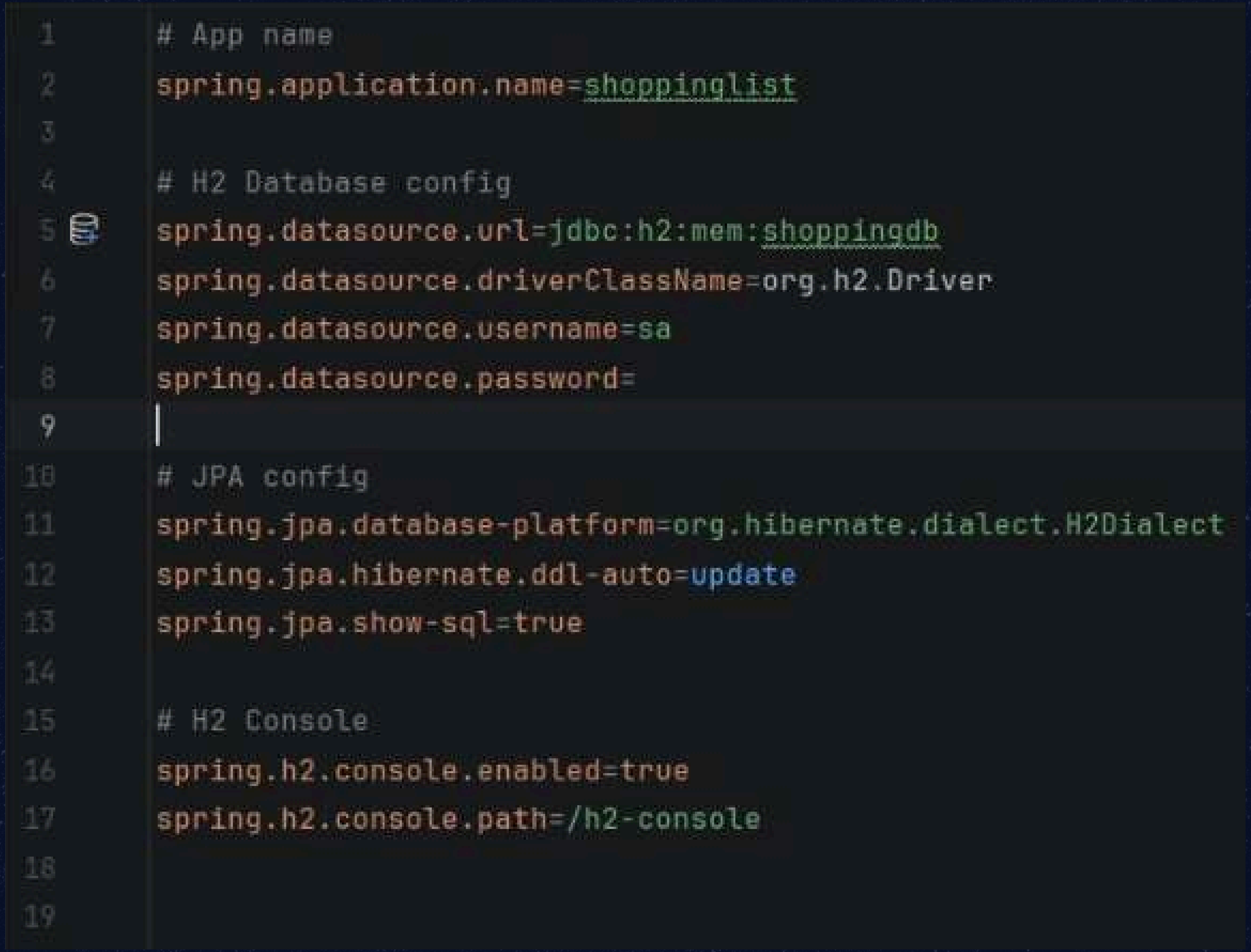
The backend of the Shopping List Application is developed using Spring Boot. It exposes REST APIs to perform CRUD operations and manages application data.



```
springboot-backend/
│
├── src/
│   └── main/
│       ├── java/
│       │   └── com/
│       │       └── shopping/
│       │           └── shoppinglist/
│       │               ├── controller/
│       │               │   └── ShoppingItemController.java
│       │               │
│       │               ├── entity/
│       │               │   └── ShoppingItem.java
│       │               │
│       │               ├── repository/
│       │               │   └── ShoppingItemRepository.java
│       │               │
│       │               └── ShoppinglistApplication.java
│       │
│       └── resources/
│           └── application.properties
│
├── Dockerfile
├── pom.xml
├── README.md
└── .gitignore
```

Database Configuration:

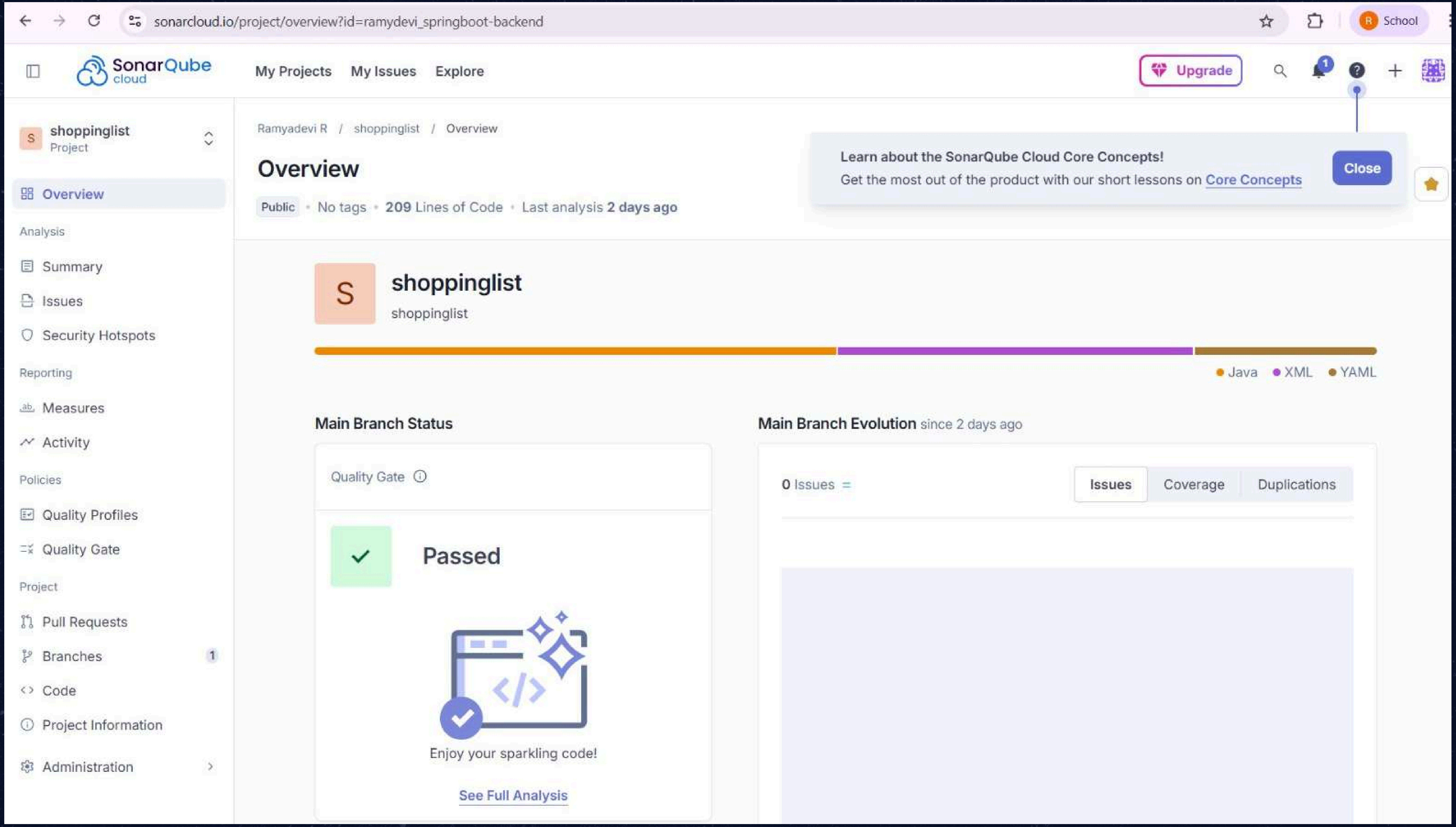
An H2 in-memory database is used for data persistence. JPA automatically manages table creation and updates.



```
# App name
spring.application.name=shoppinglist

# H2 Database config
spring.datasource.url=jdbc:h2:mem:shoppingdb
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=

# JPA config
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true

# H2 Console
spring.h2.console.enabled=true
spring.h2.console.path=/h2-console
```

Sonar Analysis – Backend:

SonarQube analysis was performed on backend code to measure code quality, reliability, and maintainability.



Ramyadevi R / shoppinglist / Overview

Overview

Public • No tags • 209 Lines of Code • Last analysis 2 days ago

Learn about the SonarQube Cloud Core Concepts!
Get the most out of the product with our short lessons on Core Concepts

shoppinglist
shoppinglist

Main Branch Status

Quality Gate

Passed

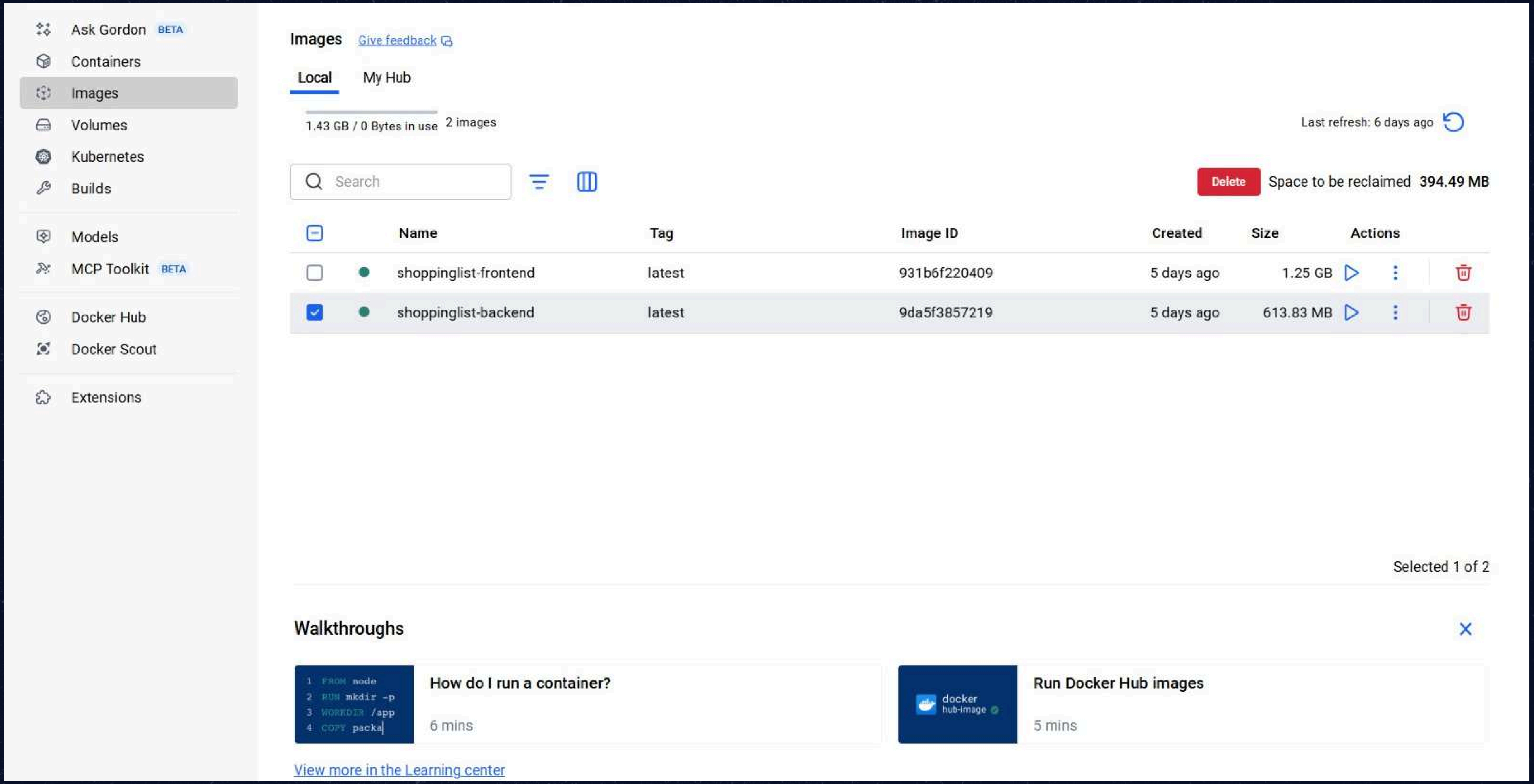
Enjoy your sparkling code!

Main Branch Evolution since 2 days ago

0 Issues =

Docker Image Build:

The backend application was containerized using Docker. A Dockerfile was used to package the application into an image.



1.43 GB / 0 Bytes in use 2 images

Last refresh: 6 days ago

Space to be reclaimed 394.49 MB

	Name	Tag	Image ID	Created	Size	Actions
☐ ●	shoppinglist-frontend	latest	931b6f220409	5 days ago	1.25 GB	
☑ ●	shoppinglist-backend	latest	9da5f3857219	5 days ago	613.83 MB	

Selected 1 of 2

Walkthroughs

```
FROM node
RUN mkdir -p
WORKDIR /app
COPY packa
```

How do I run a container?

6 mins

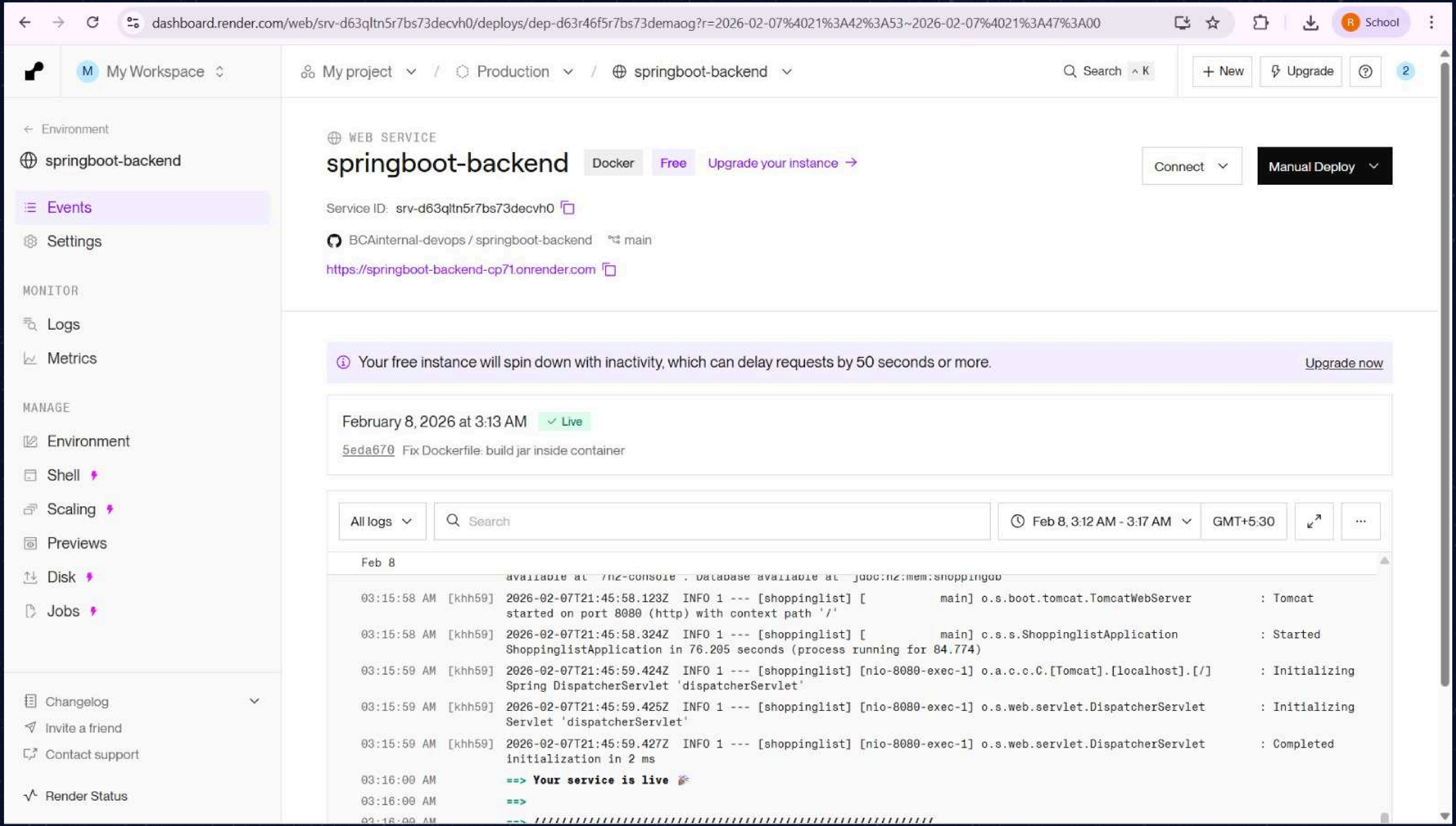
Run Docker Hub images

5 mins

View more in the Learning center

Backend Deployment – Render:

The Dockerized backend was deployed on Render as a Web Service.



springboot-backend

Service ID: srv-d63qltn5r7bs73decvh0

BCAinternal-devops / springboot-backend main

https://springboot-backend-cp71.onrender.com

Your free instance will spin down with inactivity, which can delay requests by 50 seconds or more.

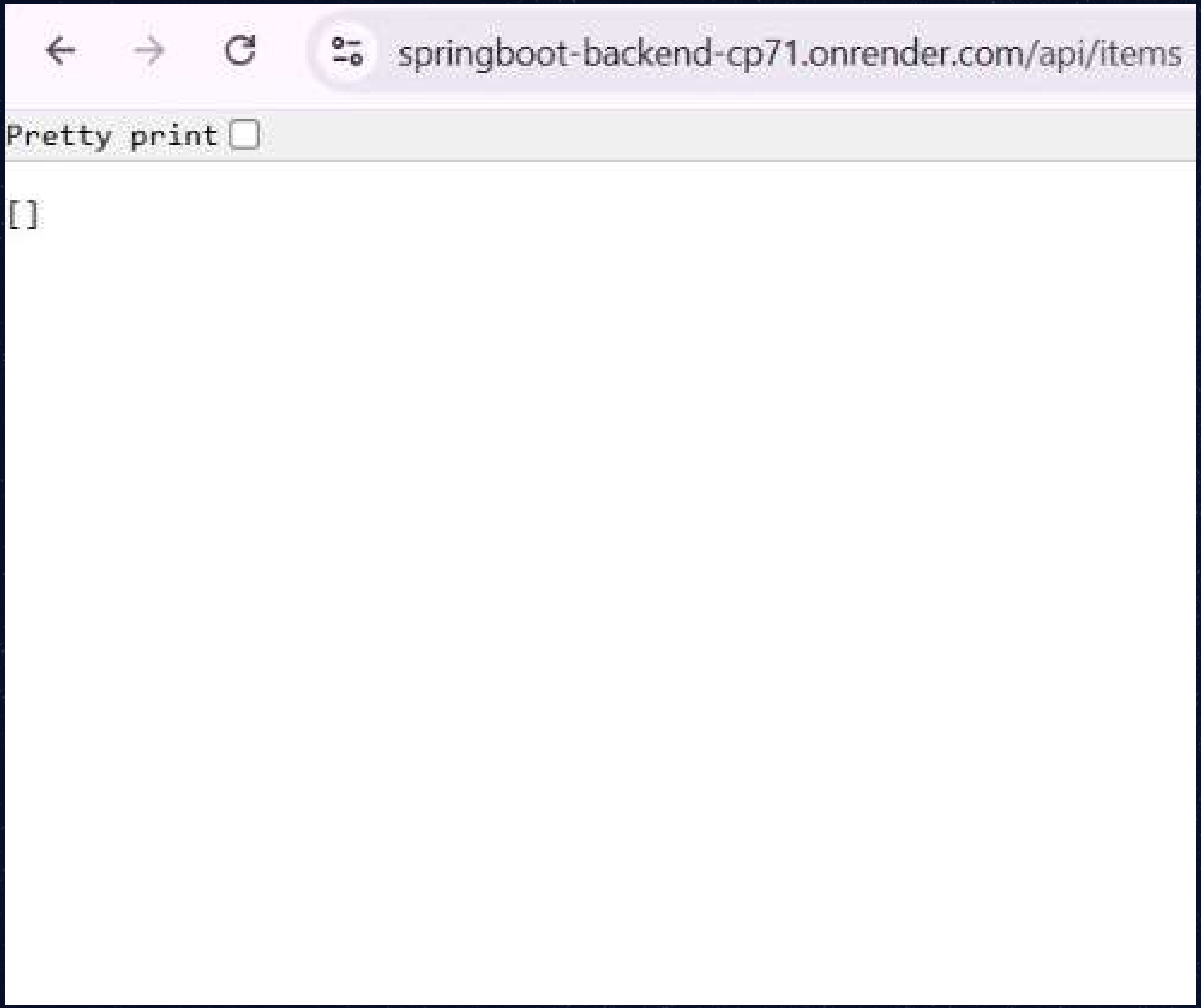
February 8, 2026 at 3:13 AM

5eda670 Fix Dockerfile: build jar inside container

```
Feb 8
                            available at '/h2-console'. Database available at 'jdbc:h2:mem:shoppingdb'
03:15:58 AM  [khh59]  2026-02-07T21:45:58.123Z  INFO 1 --- [shoppinglist] [           main] o.s.boot.tomcat.TomcatWebServer          : Tomcat
                      started on port 8080 (http) with context path '/'
03:15:58 AM  [khh59]  2026-02-07T21:45:58.324Z  INFO 1 --- [shoppinglist] [           main] c.s.s.ShoppinglistApplication            : Started
                      ShoppinglistApplication in 76.205 seconds (process running for 84.774)
03:15:59 AM  [khh59]  2026-02-07T21:45:59.424Z  INFO 1 --- [shoppinglist] [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing
                      Spring DispatcherServlet 'dispatcherServlet'
03:15:59 AM  [khh59]  2026-02-07T21:45:59.425Z  INFO 1 --- [shoppinglist] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Initializing
                      Servlet 'dispatcherServlet'
03:15:59 AM  [khh59]  2026-02-07T21:45:59.427Z  INFO 1 --- [shoppinglist] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Completed
                      initialization in 2 ms
03:16:00 AM           ==> Your service is live 🎉
03:16:00 AM           ==>
03:16:00 AM           ==> ////////////////////////////////////////////////////////////
```

Backend API Output:

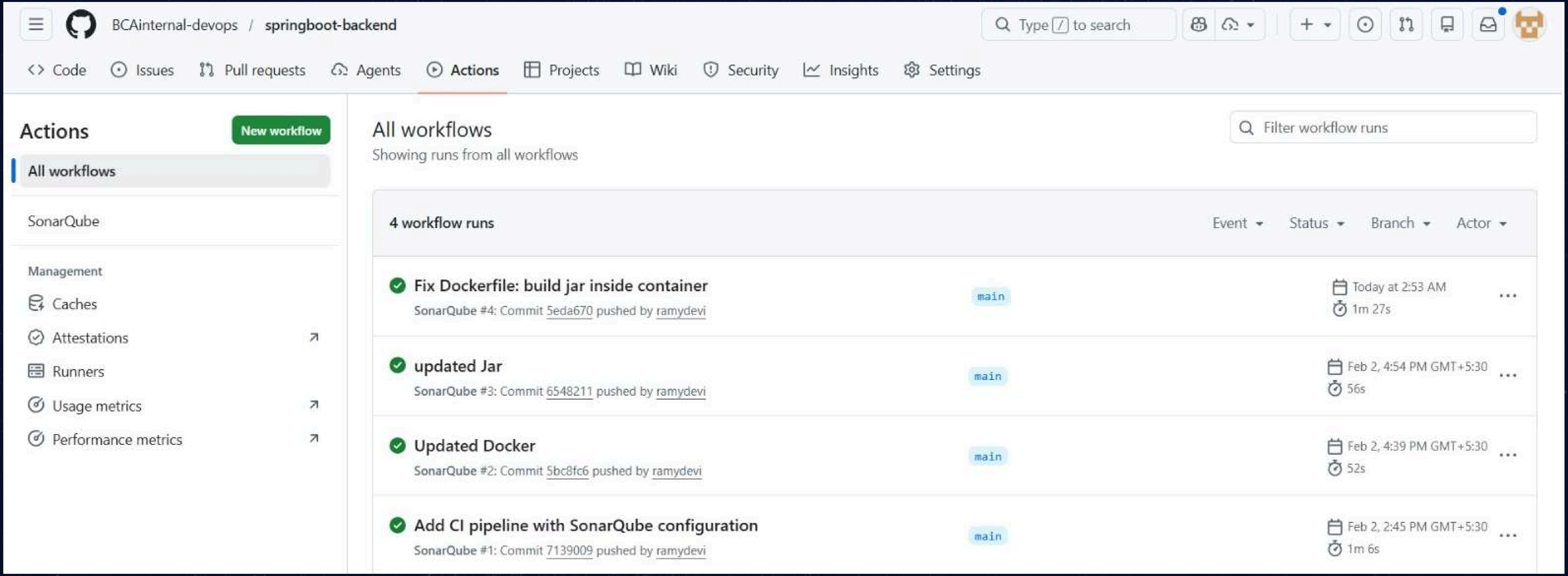
Backend APIs were tested using browser/Postman and returned JSON responses.



```
[]
```

CI & GitHub Integration:

GitHub Actions was used to automate backend builds and Sonar analysis.



All workflows

Showing runs from all workflows

4 workflow runs

Fix Dockerfile: build jar inside container SonarQube #4: Commit 5eda670 pushed by ramydevi	main	Today at 2:53 AM 1m 27s
updated Jar SonarQube #3: Commit 6548211 pushed by ramydevi	main	Feb 2, 4:54 PM GMT+5:30 56s
Updated Docker SonarQube #2: Commit 5bc8fc6 pushed by ramydevi	main	Feb 2, 4:39 PM GMT+5:30 52s
Add CI pipeline with SonarQube configuration SonarQube #1: Commit 7139009 pushed by ramydevi	main	Feb 2, 2:45 PM GMT+5:30 1m 6s

Challenges Faced:

- Docker build failures
- Missing JAR during deployment
- Render redeploy issues
- CORS configuration