

TITLE:

SHOPPING LIST APPLICATION – BACKEND

SUBTITLE:

**SPRING BOOT REST API WITH DOCKER &
RENDER DEPLOYMENT**

III-BCA
23SUCAS0

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 items 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 STRUCTURE:

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



Database Configuration:

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

```
1  # App name
2  spring.application.name=shoppinglist
3
4  # H2 Database config
5  spring.datasource.url=jdbc:h2:mem:shoppingdb
6  spring.datasource.driverClassName=org.h2.Driver
7  spring.datasource.username=sa
8  spring.datasource.password=
9
10 # JPA config
11 spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
12 spring.jpa.hibernate.ddl-auto=update
13 spring.jpa.show-sql=true
14
15 # H2 Console
16 spring.h2.console.enabled=true
17 spring.h2.console.path=/h2-console
18
19
```

Sonar Analysis – Backend:

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

The screenshot shows the SonarCloud project overview for the 'shoppinglist' project. The left sidebar contains navigation links for My Projects, My Issues, Explore, Overview, Summary, Issues, Security Hotspots, Reporting, Measures, Activity, Policies, Quality Profiles, Quality Gate, Pull Requests, Branches, Code, Project Information, and Administration. The main content area displays the project name 'shoppinglist' and its status as 'Passed' under the Quality Gate. It also shows the Main Branch Evolution with 0 issues found since the last analysis 2 days ago. A callout box in the top right corner encourages learning about SonarQube Core Concepts. The URL in the browser bar is sonarcloud.io/project/overview?id=ramydevi_springboot-backend.

Docker Image Build:

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

The screenshot shows the Docker desktop application interface. On the left is a sidebar with various options: Ask Gordon (BETA), Containers, Images (selected), Volumes, Kubernetes, Builds, Models, MCP Toolkit (BETA), Docker Hub, Docker Scout, and Extensions. The main area is titled 'Images' with a 'Give feedback' link. It shows two images under the 'Local' tab: 'shoppinglist-frontend' and 'shoppinglist-backend'. Both images are tagged 'latest'. The 'shoppinglist-frontend' image has an ID of 931b6f220409 and was created 5 days ago, taking up 1.25 GB. The 'shoppinglist-backend' image has an ID of 9da5f3857219 and was created 5 days ago, taking up 613.83 MB. There is a 'Delete' button and a note about reclaiming 394.49 MB of space. Below the table, it says 'Selected 1 of 2'. At the bottom, there are 'Walkthroughs' sections for 'How do I run a container?' and 'Run Docker Hub images', each with a duration of 6 mins. A link at the bottom says 'View more in the Learning center'.

| Name | Tag | Image ID | Created | Size | Actions |
|-----------------------|--------|--------------|------------|-----------|--|
| shoppinglist-frontend | latest | 931b6f220409 | 5 days ago | 1.25 GB | ... Delete |
| shoppinglist-backend | latest | 9da5f3857219 | 5 days ago | 613.83 MB | ... Delete |

Selected 1 of 2

Walkthroughs

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.

The screenshot shows the Render dashboard interface for a project named "springboot-backend".

Header: The URL is `dashboard.render.com/web/srv-d63qltn5r7bs73decvh0/deploys/dep-d63r46f5r7bs73demaog?r=2026-02-07%4021%3A42%3A53~2026-02-07%4021%3A47%3A00`. The top navigation bar includes "My Workspace", "My project / Production / springboot-backend", "Search", "+ New", "Upgrade", and a user icon.

Left Sidebar: Includes sections for "Environment", "Events" (selected), "Settings", "MONITOR" (Logs and Metrics), "MANAGE" (Environment, Shell, Scaling, Previews, Disk, Jobs, Changelog, Invite a friend, Contact support), and "Render Status".

Service Overview: Shows the "springboot-backend" service as a "WEB SERVICE". It is running on "Docker" and is "Free". A link to "Upgrade your instance" is available. Buttons for "Connect" and "Manual Deploy" are present.

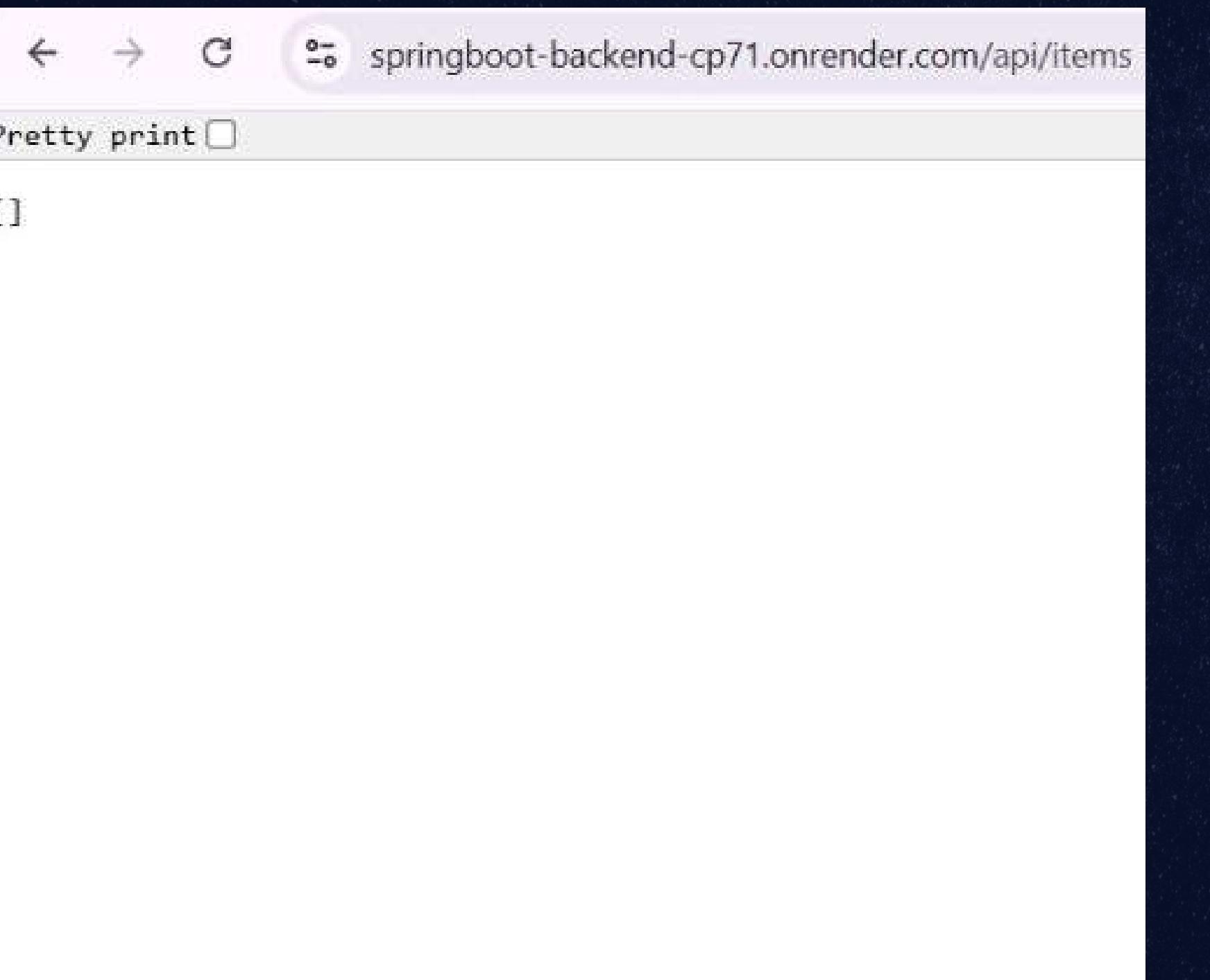
Logs: The logs section displays a log entry from February 8, 2026, at 3:13 AM, labeled "Live". The log message is: "Seda670 Fix Dockerfile: build jar inside container".

Log Viewer: A detailed log viewer for February 8, 2026, from 3:12 AM to 3:17 AM, in GMT+5:30. The log output shows the application starting up, including the database connection and Tomcat initialization. The final message is "Your service is live".

| Date | Time | Message | Details |
|-------|-------------|--|--|
| Feb 8 | 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 '/' |
| Feb 8 | 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) |
| Feb 8 | 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' |
| Feb 8 | 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' |
| Feb 8 | 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 |
| Feb 8 | 03:16:00 AM | ==> Your service is live !* | |
| Feb 8 | 03:16:00 AM | ==> | |
| Feb 8 | 03:16:00 AM | --- | |

Backend API Output:

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



A screenshot of a web browser window. The address bar shows the URL: `springboot-backend-cp71.onrender.com/api/items`. Below the address bar, there is a checkbox labeled "Pretty print". The main content area of the browser displays a single character "[". This represents the JSON response from the API endpoint, indicating an empty array.



CI & GitHub Integration:

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

The screenshot shows the GitHub Actions interface for the repository 'BCAinternal-devops / springboot-backend'. The left sidebar is collapsed, and the top navigation bar includes 'Code', 'Issues', 'Pull requests', 'Agents', 'Actions' (which is highlighted), 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. A search bar at the top right says 'Type / to search'.

The main area is titled 'All workflows' and shows '4 workflow runs'. The runs are listed as follows:

| Workflow Run | Event | Status | Branch | Actor | Timestamp | Duration | More Options |
|--|---|---------|--------|----------|-------------------------|----------|--------------|
| Fix Dockerfile: build jar inside container | SonarQube #4: Commit Seda670 pushed by ramydevi | Success | main | ramydevi | Today at 2:53 AM | 1m 27s | ... |
| updated Jar | SonarQube #3: Commit 6548211 pushed by ramydevi | Success | main | ramydevi | Feb 2, 4:54 PM GMT+5:30 | 56s | ... |
| Updated Docker | SonarQube #2: Commit 5bc8fc6 pushed by ramydevi | Success | main | ramydevi | Feb 2, 4:39 PM GMT+5:30 | 52s | ... |
| Add CI pipeline with SonarQube configuration | SonarQube #1: Commit 7139009 pushed by ramydevi | Success | main | ramydevi | Feb 2, 2:45 PM GMT+5:30 | 1m 6s | ... |

Challenges Faced:

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

