Birthday Notification App – Dockerized Setup Guide

🎉 Birthday Notification App – Dockerized Setup Guide

This document outlines the steps to build, run, and troubleshoot your Spring Boot Birthday Notification App with Docker.

🛠️ Step-by-Step Instructions

1. Clean and Package the Application

```
mvn clean package
```

- **Purpose**: Cleans previous builds and compiles the project into a deployable JAR.

2. Build the Docker Image

```
docker build -t birthday-app .
```

- Purpose: Builds a Docker image named birthday-app using the Dockerfile in the current directory.

3. Run the Docker Container (Initial Run)

```
docker run -d --name birthday-app -p 8080:8080 `
-v /mnt/c/data:/app/data `
-e SPRING_PROFILES_ACTIVE=default `
birthday-app
```

- **Explanation**:
  - -d: Run container in detached mode.
  - --name birthday-app: Assigns a name to the container.
  - -p 8080:8080: Maps local port 8080 to container port 8080.
  - -v /mnt/c/data:/app/data: Mounts a volume from host to container.
  - -e SPRING_PROFILES_ACTIVE=default: Sets the active Spring profile.

4. **View Container Logs**

```
docker logs birthday-app
```

- **Use this** to troubleshoot if the app fails to connect to MySQL or access Excel files.

5. **Access the Application**

Open your browser and visit:

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

6. **Manage Your Container**

Check Running Containers:

```
docker ps
```

Stop the Container:

```
docker stop birthday-app
```

Start the Container Again:

```
docker start birthday-app
```

And In your application-local.properties add this one

```
spring.datasource.url=jdbc:mysql://172.17.220.138:3306/birthday_notification?use
spring.datasource.username=root
spring.datasource.password=Root@1234
birthday.excel-file-path=C:/data/birthdays.xlsx
```

🧩 Dealing with MySQL Connection Issues

If you're having trouble with DB connections or Excel file paths, inspect logs:

```
docker logs birthday-app
```

🔧 Recommended: Custom Docker Network (To Avoid Connectivity Issues)

By default, Docker's bridge network may isolate services. Create a user-defined bridge network to resolve networking issues.

1. **Create the Network**

```
docker network create birthday-net
```

- Benefits:
  - Containers can communicate using names.
  - DNS resolution works better.
  - Avoids quirks with Docker Desktop + Windows + WSL.

2. **Rebuild Image (If needed)**

> **"Note: Ensure that when the following command is executed, the MySQL configuration is set to local and not Docker."**

For Local

```
# Change profile from here based on image local or docker
spring.profiles.active=local
```

After that execute this for build docker image

```
docker build -t birthday-app .
```

After that change

```
# Change profile from here based on image local or docker
spring.profiles.active=docker
```

3. **Run Container with Custom Network (PowerShell-friendly)**

```
docker run -d --name birthday-app --network birthday-net -p 8080:8080 `
-e SPRING_PROFILES_ACTIVE=docker `
-e SPRING_DATASOURCE_URL="jdbc:mysql://host.docker.internal:3306/birthday
-e SPRING_DATASOURCE_USERNAME="root" `
-e SPRING_DATASOURCE_PASSWORD="Root@1234" `
-e BIRTHDAY_EXCEL_FILE_PATH="/app/data/birthdays.xlsx" `
-v "C:\data:/app/data" `
birthday-app
```

- **Using** `host.docker.internal` ensures your container can access the host machine (e.g., MySQL running on your PC).

And In your application-docker.properties add this one

For Docker

```
spring.datasource.url=${SPRING_DATASOURCE_URL}
spring.datasource.username=${SPRING_DATASOURCE_USERNAME}
spring.datasource.password=${SPRING_DATASOURCE_PASSWORD}
birthday.excel-file-path=${BIRTHDAY_EXCEL_FILE_PATH}
```