Guide Overview

Prerequisites

Before you start, ensure you have the following installed on your machine:

- **Docker**: Docker Engine is required to build and run containers.
- **Docker Compose**: Docker Compose is used to manage multi-container Docker applications through a YAML file.

Recommendation:
Install docker desktop to fulfill the above prerequisites.

1. Understanding the Compose File

The provided Docker Compose file defines a service named ossqa. Here’s a breakdown of the key components:

- **version**: This specifies the version of the Docker Compose file format.
- **services**: This section defines the services (containers) that make up the application.
- **ossqa**: The name of the service.
- **image**: Specifies the image to use for the service. If the image does not exist locally, Docker will attempt to pull it from the configured registry.
- **build**: Defines configuration options that are applied at build time.
- **context**: The build context path to the directory containing the Dockerfile and any other build files.
- **dockerfile**: The path to the Dockerfile within the build context.

2. Building the Image

To build the Docker image for the ossqa service, navigate to the directory containing your docker-compose.yml file and run:

```
docker-compose build
```

This command tells Docker Compose to build the image using the specified Dockerfile and context.

3. Running the Container

After the image is built, you can run the container with the following command:

```
docker-compose up
```

This command starts the ossqa service. By default, Docker Compose runs in the foreground, displaying the service's log output to the console.

To run the services in the background, add the -d (detached) option:

```
docker-compose up -d
```

4. Viewing Logs and Managing the Service

To view the logs of the running service, use:

```
docker-compose logs
```

You can follow the logs in real-time by adding the -f option:

```
docker-compose logs -f
```

To stop the service, use:

```
docker-compose down
```

This command stops and removes the containers, networks, and volumes associated with the service.