Specification Document

For

Orchestrating an angular web application on Docker

Version 0.0.1

Prepared by Soumyajit Basu

11th July 2020

Table of contents

Introduction

This is a brief documentation for understanding how the docker ecosystem works and how container orchestration can be done using the docker compose.

Problem statement

- Setting up an image for code development
- Build the application in Docker and host it in Docker Hub
- Run the Angular application successfully in the Docker container.
- Use Docker Compose to manage the Angular application running inside the Docker container.
- Host the application on Docker Hub.

Project Specification

Installation

- To install docker please follow this link.
- Select the option from `Mac with Intel chip` or `Mac with Apple chip`
- Download the software.
- Drag the `Docker` application to the `/applications` directory.

Technology

- **Angular js** for developing the application.
- **Docker** to orchestrate the angular application.
- **Docker compose** to manage containers.
- **DockerHub** where the application will be pushed once the image is built inside the container.
- **GitHub** to maintain the version control

Application

The angular application is a collaborative blogging platform where a user can essentially pen down their thoughts on technology or any article related to technology. So as a user I can register myself to the platform. Once registered I can sign in to the platform, write and submit an article that can be visible on the dashboard. The workflow for the application is in this link.

Orchestration

To orchestrate the application the project is built using docker. To execute the container with respective configuration we are going to use the docker-compose to drive the container where we are going to declare a web service that would build, version and run the container.

Docker Build

The docker build command builds Docker images from a Dockerfile and a “context”. A build’s context is the set of files located in the specified PATH or URL. The build process can refer to any of the files in the context. For example, your build can use a COPY instruction to reference a file in the context.
The URL parameter can refer to three kinds of resources: Git repositories, pre-packaged tarball contexts and plain text files.

The architecture involves two aspects in the context of build.

- A docker build that sets up a basic image with all the system dependencies with the ubuntu vanilla operating system as the base image.
- A second docker build that would refer to the above image as a reference point. This would reduce subsequently the build time as it would already contain the cached packages within the image already. This build would handle the node dependencies, angular dependencies.

Docker Compose

Compose is a tool for defining and running multi-container Docker applications. With Compose, you use a YAML file to configure your application's services. Then, with a single command, you create and start all the services from your configuration.

Compose works in all environments: production, staging, development, testing, as well as CI workflows.

Docker Hub

Docker Hub is a platform to create, manage, and deliver your teams' container applications.

Workflow

The workflow will define the thought process in developing the orchestration to manage and control the deployment of the angular application within docker. Before going ahead with the explanation of the orchestration process we should understand how the application is built and deployed using angular.

Building and running an angular application

To run an angular build and execute the application we need some basic installations to go ahead.

Requirement

- Installing the latest version of node. If not, the latest version updates to the latest version of node.
- Installing the latest version of npm.
- Install angular cli.

Installation

Note: The documentation considers debian based operating system (Ubuntu) as the reference to installation of the required packages as mentioned.

- To install node run the command `apt-get install -y nodejs`. This would install the stable version of node which would be v10.0. But to run angular cli build it requires v12.0 or v14.0. So we would need to update the version of node.
- To install npm run the command `apt-get install -y npm`. This would install the node package manager to install packages and dependencies.
- To update the version of node we require a package called n. To install n we can run the command `npm install -g n`.
- To install angular-cli we need to run the command `npm install --global @angular/cli`.

- To install yarn we need to run the command `npm install --global yarn`.

As a best practice we have installed yarn as a package manager for the javascript code.

Build

To run the build execute the command `ng build`.

Serve

`ng serve` command builds and **serve** the **application**. It rebuilds the **application** if changes occur. Here project is the name of the **application** as defined in **angular**. Json.

Docker Build

To write configuration using docker we need to create a `Dockerfile`. This should copy the respective project to the container. Make the required installations required to drive the application as mentioned above and run the command `docker build -t <image_tag> <dockerfile_path>`. The docker configurations are written here.

Docker Hub

The docker image once executed successfully can be made available to the rest of the team using **DockerHub**. To push the application image to the registry follow the required steps.
Make sure you have docker hub login done. You can do this from the docker application.
To push an image to the docker registry first you will need to create a `tag` that would mention the username as `<username>/<required_tag_name>`.

- To create a tag run the command `docker tag <image_name> <hub_user_name>/<tag_name>`.

Docker Compose

The docker compose will be used to configure the container based on the service defined. In the docker composer we have mentioned:

- The service `web` and the build operation.
- The corresponding `port` to which the application will be published.
- Define the `context` and the dockerfile `path`.

To run the container we just need to run the command `docker-compose up`. The docker-compose yml configuration is in this link.

To test if the application is running, open a browser and hit 127.0.0.1:4200. If the application is successfully running then the deployment is successful.

- To push the image to the registry run the command `docker push <hub_user_name>/<tag_name>` .