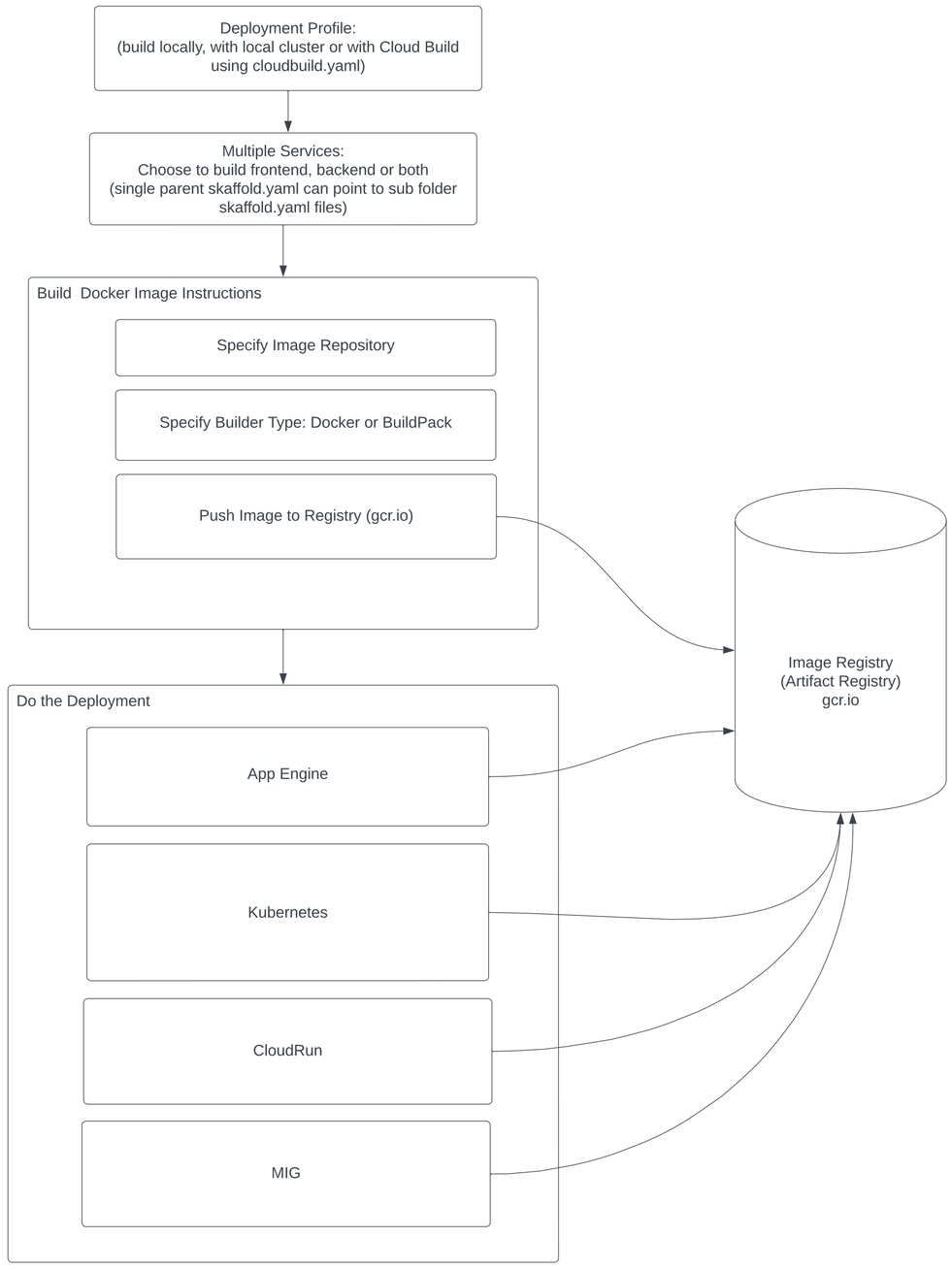
SKAFFOLD (skaffold.yaml) - Sean Corzo 8/14/2024
Used for Application Building and Deployment



DOCKERFILE VS. DOCKER IMAGE VS. DOCKER CONTAINER

In containerization, a **container registry** primarily holds container images.

A **Dockerfile** (or similar format) **defines how the container image should be constructed**. These instructions include steps like copying files into the image, installing packages, and setting environment variables.

A **container image** is a lightweight, standalone, and executable software package that includes everything needed to run a piece of software, including the code, runtime, system tools, libraries, and settings. It's essentially a snapshot of a specific application along with its dependencies and environment settings.

A container image is not just a definition file; it's **a complete package that includes the application code, its runtime environment, and any required dependencies**.

A container, on the other hand, **is an instance of a running container image**. When you launch a container from an image, you are essentially running a process isolated from the host system and other containers, but with access to the resources and libraries included in the container image. Containers provide a consistent and reproducible environment for running applications, and they are designed to be portable across different computing environments.

BUILDING A DOCKER CONTAINER

- Building a Docker image using a Dockerfile may require access to additional commands. Docker will look for those additional commands within the context of the image being built, not on your local filesystem. This means that Docker will use the base image specified in the Dockerfile (such as a Node.js base image) and install npm packages inside the image itself. Docker will download and install npm packages as specified in the Dockerfile's instructions.

- Here are some example commands that are defined in the Dockerfile and that get run when doing a docker build in order to create a docker container image

- Example command "docker build -t "gcr.io/project/my-image""

    # go code
    go mod download // download everything in go.mod file - can be run in Dockerfile
    go build mainfile.go // compile to executable binary with same name in same location - can be run in Dockerfile

    # create react app code
    npm install (installs package.json packages) // can be run in Dockerfile
    npm run build (runs "build" command from package.json) // can be run in Dockerfile

- Here is an example Dockerfile that would install the Node.js base image in order to use npm:

```
# Use a Node.js base image
FROM node:14

# Set the working directory
WORKDIR /app

# Copy package.json and package-lock.json to the container
COPY package*.json ./

# Install npm packages
RUN npm install

# Copy the rest of the application code
COPY . .

# Specify the command to run when the container starts
CMD ["npm", "start"]
```

And here is an example for a "go build" command:

```
# Use an official Go image as the base image
FROM golang:1.17

# Set the working directory within the container
WORKDIR /app

# Copy the Go application source code into the container
COPY . .

# Build the Go application
RUN go build -o myapp

# Set the command to run the application when the container starts
CMD ["./myapp"]
```