Blog

Written words that are often helpful

Google App Engine Made Easy

Posted by chathaway on March 28, 2020, 7:39 a.m.

Google App Engine made easy

Turns out, GAE can run a Docker container, which makes life super easy.Some basic rules:

- EXPOSE 8080 # GAE will only talk to port TCP/8080 to serve web traffic
- All build files (i.e., everything in the folder app.yaml is in) gets uploaded and built

So its very limited in what it can do, but basically with the structure

```
.■■■ app.yaml■■■ Dockerfile
```

You can easily make an app. You might need to add persistence (i.e., databases) through their service,but you could at least bring small apps to play with for very low budgets ($4/month, for the smallest contained).

An example; minimal Rust webserver

Disclaimer; I like it simple, so we're not going to integrate the Docker packaging with the binary creation.Just play along.

First, make our Rust app:

```
cargo new rust-gaecd rust-gae
```

Now, quickly setup a basic web server; we'll use Hyper and tokio.

```
cat <<EOF > src/main.rs#![deny(warnings)]use std::convert::Infallible;use hyper::service::{make_service_fn, service_fn};use hy
```

Make sure to setup the Cargo.toml file! Default for everything, but add 3 libraries.

```
# This attempts to append the depends to the bottom of the Cargo.toml; if the base# template changes, you may have to do this
```

If you need to, add the muslc component. This lets us make the Docker file really small,and really simple.

```
rustup target add x86_64-unknown-linux-musl
```

Then, build our application!

```
cargo build --target x86_64-unknown-linux-musl --release
```

Finally, let's make a Docker image. First, make a directory and copy the binary we just build in.

```
mkdir distcd distcp ../target/x86_64-unknown-linux-musl/release/rust-gae .
```

Then make our Dockerfile

```
cat <<EOF > DockerfileFROM scratchCOPY ./rust-gae .EXPOSE 8080ENTRYPOINT ["./rust-gae"]EOF
```

Verify that it builds with `docker build .`, and we're in business!.

Last thing to do is setup app.yaml, and do the work of setting up a GCP project. Basically,you want to be at a point where the command `gcloud app deploy` works. Follow the Google documentation for that bit,since it's not my specialty.

Setup the app.yaml file

```
cat <<EOF > app.yamlenv: flex# This bit here is optional, but makes it so only a single instance is brought up.# This keeps co
```

And finally, use `gcloud app deploy` to deploy it. You should be provided with a URL, and see magic when you go to it.

Good luck out there!

Still here?

Try to bring down that app. I started polling at 120 requests per second, at which point I think the limiting factor wasactually my machine/internet. A more complex application might be a bit slower, but still, for a dirt cheap machine 120RPS isn't bad. That equates to roughly 10_368_000 a day. Need more? Scale up.