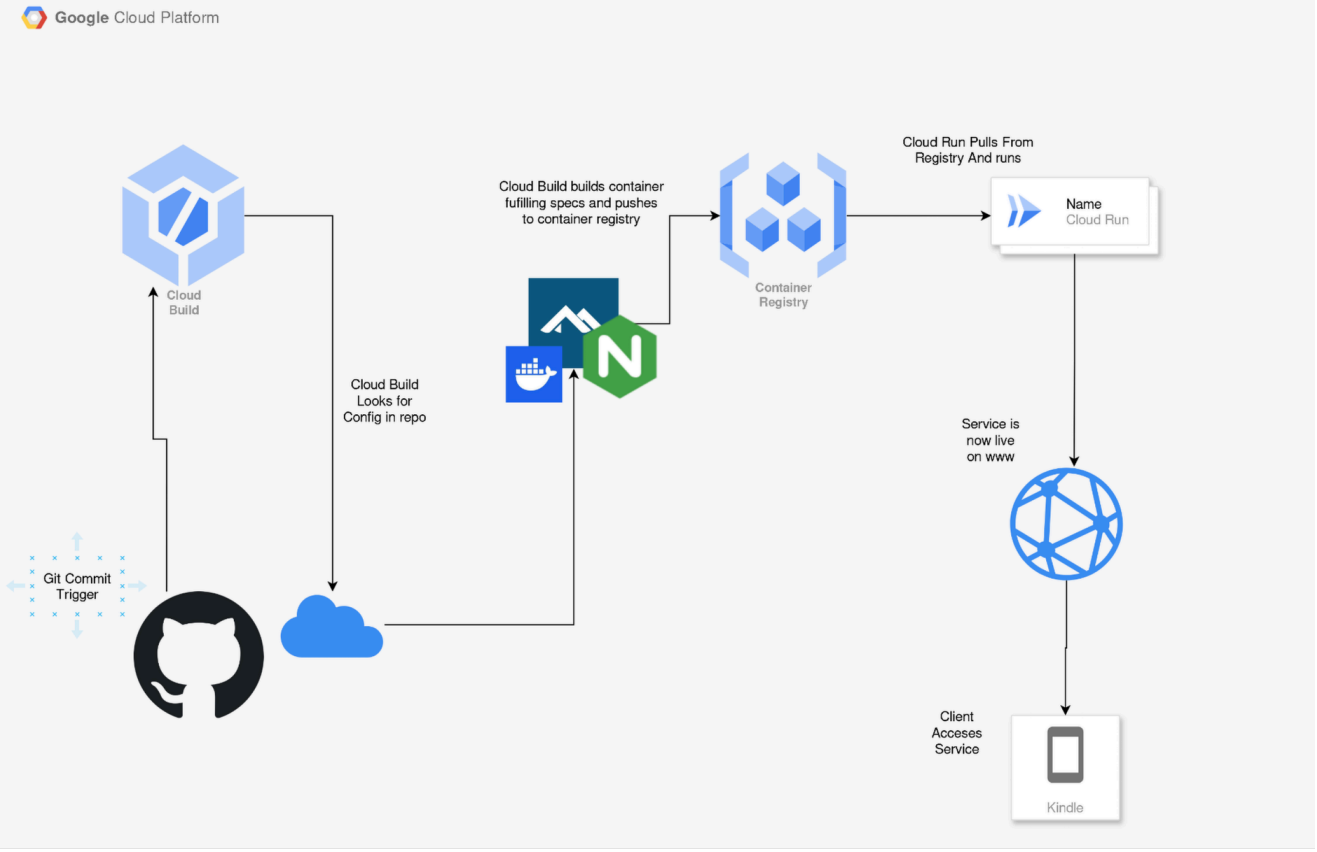
[Link To Github Repo]

Application Overview

I am serving a web based application with a few games and a markdown editor for use om kindle and other e-paper reader devices.
Since the kindle is a locked down system, using web based applications to extend its functionality can greatly expand its potential.
The target user base is people seeking "digital minimalism" in a world full of bright screen devices constantly seeking your attention. it can be a nice break to do some work or play on a device without notifications or sound.

Architecture Diagram



GCP Services

Cloud Run:

Cloud Run is used to Serve a containerised web application, managing scaling and https traffic.

The deployment process consists of three stages

Build

```
-  name: gcr.io/cloud-builders/docker
args:
[
"build",
"-t", "gcr.io/$PROJECT_ID/markdown-editor:latest",
"."
]
```

This **builds** the image with docker, and assigns a name and a tag to it
in this instance, the name markdown-editor and the tag latest.

Push

```
- name: gcr.io/cloud-builders/docker
args:
[
"push",
"gcr.io/$PROJECT_ID/markdown-editor:latest"
]
```

Now this **pushes** the image, referenced with our name and tag to the container registry, this is the holding place for our images after build.

Deploy

```
- name: gcr.io/cloud-builders/gcloud
args:
[
"run", "deploy", "markdown-editor",
"--image", "gcr.io/$PROJECT_ID/markdown-editor:latest",
"--platform", "managed",
"--region", "us-central1",
"--allow-unauthenticated",
"--port", "8080"

]
```

This orchestrates cloud run to **pull** the identified image from the container registry and **deploy** it.
we have additional options.
Most importantly for base function, cloud run listens on port 8080 which is aligned with our nginx config so we can serve our content.

Cost Calculation.

Considering the lightweight nature of this application as well as its niche use-case, i would imagine expect to remain within the limitations of the free tier with a single instance and 30GiB of outgoing traffic. of course this will need to be reviewed post release if the application gains unexpected traction.

My Goal

To deploy a website by cloning the github repo during container build to keep my files seperate.

What happened in reality.
This function works in docker using
`RUN git clone /nginx/www
If you directly push this to cloud run it also works
But
if you try and orchestrate this behaviour in your cloud builder file to be run on a trigger, it doesnt work.

fixes that helped:
re checking permissions for service accounts.
Give as much permissions required to fulfil the deployment

Watch out for what ports you are serving to,
Make sure cloud run is alligned with your nginx config

My Soloution:

Add Web Files to cloud deploy repository,
If any of these files are edited and pushed to this repository, it wil successfully trigger the CID pipeline and redeploy my service.

What i learned about DevOps from this experience.

Just because something works in docker doesn't mean you can implement it the same way on a full ci/cd cloud deployment.
Always double and triple check your service account setup and permissions.
Make sure to git pull before you push, don't let your local repository desync from your online repo.
If you have problems, change test unchage test, don't make multiple fixes at the same time.
If all else fails, go back to something that's guaranteed to work and slowly add in what you need.