Scale from 0: Saving nickels to feed my AWS addiction

Keeping my bad ideas from bankrupting me

AARON BATILO
JAN 31, 2023

2 1 Sh

So, I'm going to keep it real honest with ya'll this week. This post won't have quite t same *refinement* that I usually try to stick to, but I kind of wanted to try writing in a more natural tone for me. Trying to sound all proper and professional is sucking so of the fun out of my experiments. Let me know what you think. I'm not sure if this tone fits for a newsletter. Maybe it's better for a blog, but you know what, does it matter? I'm writing for me and I just get a kick out of things when people find the content useful or interesting.

Anyways…

I've come to learn that most of my ideas are really bad. In my mind, I keep hoping t some project of mine is going to go kind of viral and like, maybe I become famous f it or something. In reality, I know that banking on any one particular site to do well isn't great. On top of that, my projects are always pretty small. An old co-worker of mine said that I have "one weekend syndrome". I'm only interested in a project for about a weekend before I move on, which is… it's pretty true. It's hard to build an

Mexico where we went out on a boat that had like 12 fishing rods sticking out of it, even though there was only 4 humans on the boat. How does that relate to my side projects? I know most of my ideas are bad, so instead of worrying about how bad th are, I ignore it completely and think that if I put out as many as I can, then maybe o of them will actually stick. Angry Birds was the 51st game made by Rovio. That's a l of games. And now they have a whole ass movie and stuff. BUUUUUUUT, I don't h any funding. I don't have angel investors or VCs funding anything. So I need to pay everything and that can get pretty expensive.

A lot of hosting websites will give you a free tier of some kind and that's great, but free tier tends to only be useful for one project worth of compute unless you do something that's a little cheeky.

Heroku's basic tier costs about $5 a month. Fly.io gives you about $6 a month worth compute on the free tier. The smallest DigitalOcean VM is still going to run you $4 month. Vercel is free but only for personal use and not commercial use. AWS has a pretty cool sounding free tier. For the first 12 months you get things like 750 hours month worth of compute for free if you run the smallest possible instances. But it's pretty well understood these days that the AWS free tier is delightfully deceptive. A you only get one of those EC2 instances for free. If you run two for the month, you'r going to pay $7.60 a month for that second t3.micro. If you're comfortable with AW you can run it as a spot instance and that's still $2.27 a month.

If I had a bunch of side projects, this all starts to add up really quickly. But let's be r My AWS bill is something like $210 a month. $75 a month for my EKS control plane About $30 a month for 3 t3.medium EC2 instances. About $55 a month for my Auro Postgres cluster, and about $40 a month on CloudWatch logs. The rest is in small storage costs like ECR or S3. Yes, yes. I know about things like serverless Aurora. I know that I could reduce how much logging I'm doing.

So scaling to 0 isn't exactly going to save me that much money. That's not the point though, I like messing with the infra that I have and hey, it's my experiments. I get do what I want. I consider it an investment in myself and my growth because I have amazing time just messing around with all of the things in my cluster. I like the Kubernetes ecosystem. Tons of software is available these days as a docker containe and sometimes even a helm chart, which means it takes literally minutes for me to stand up a new project to play with in some cases. Fortunately, that amount of mon is probably going to stay pretty flat for as long as I keep making projects. So if I cor up with like 40 ideas, I might actually break even. Scaling to 0 helps with that so we going to talk about it.

"What do you even mean when you say scale to 0, Aaron?"

Yeah, good point. So when I'm talking about scaling to 0, I'm talking about making sure that if a project or website that I'm running isn't actually being visited by anyo that the code isn't actually running anywhere at all. Scaling on demand, serverless, functions as a service. These are some of the other names that I've heard used for tl It's a really interesting concept. But SOMETHING has to run, right? Yep, so what a running? How am I accepting requests to my EKS cluster and how is it knowing to schedule pods?

OpenFaaS

OpenFaaS is a functions as a service framework that you can deploy to a Kubernete cluster. Also maybe worth trying one day but it basically reinvents how you do thin like deployments. You use their CLI for building and deploying functions and I war to keep things more Kubernetes native, which actually leads me to…

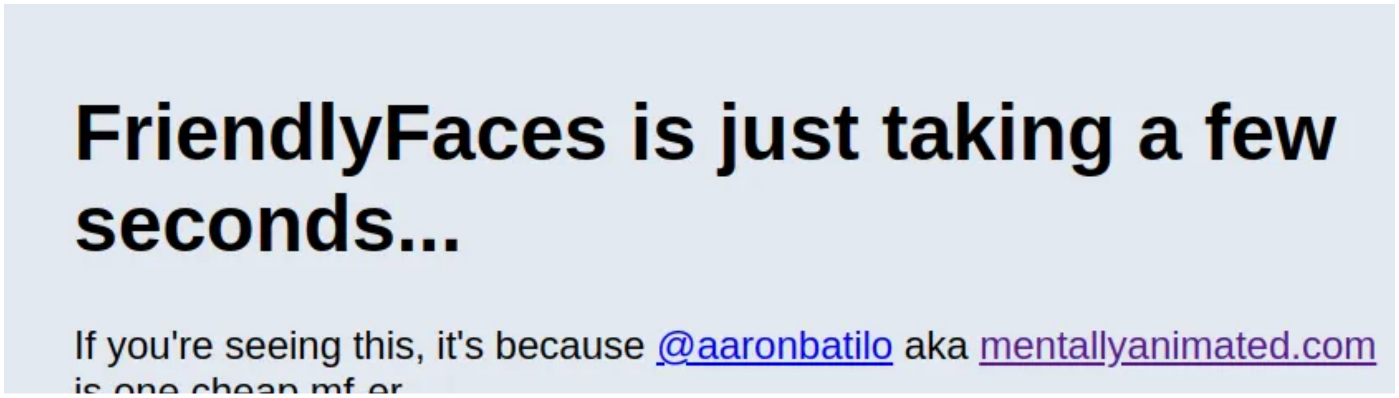
KNative

KNative is probably the default that most people end up coming across. I haven't actually tried it but thought I should mention it. I've purposefully decided that I wanted to try to avoid KNative entirely if I could, because in order to use it, you hav to install a separate networking layer. That's a little more complexity than I care to actually try right now, but maybe one day.

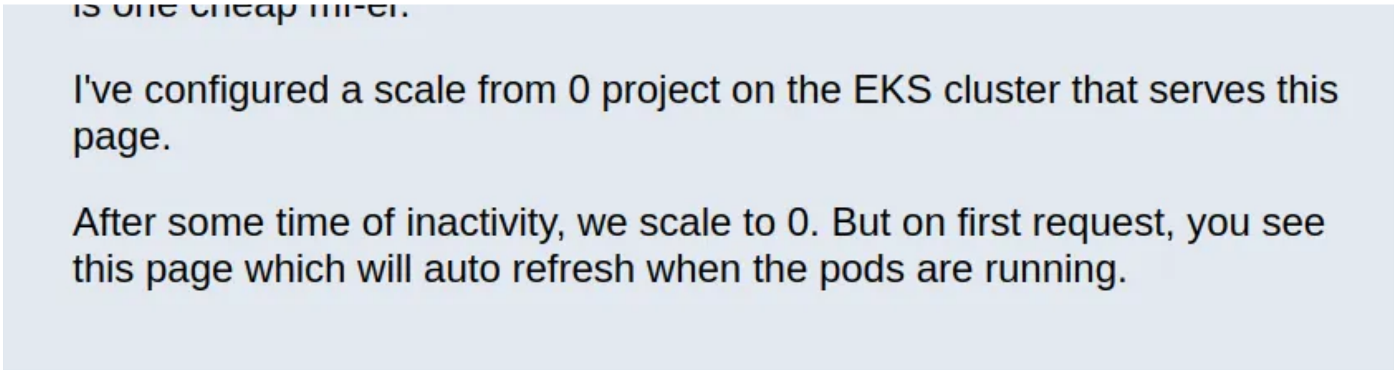
Sablier

Sablier is actually the first project that I tried. And conceptually, I actually REALL like how it works. When I set it up, I configured it as a plugin for Traefik which is r ingress controller of choice these days. I got it set up in about 25 minutes and starte to play with it. The way it works is that you set it up as Traefik middleware, which means when Traefik receives a request, before it makes it to your service, it goes through sablier first. The way sablier itself works is that when the middleware is triggered, it sends a blocking request to the sablier service that you install into you cluster. When the middleware sends a request to sablier, sablier checks to see if it needs to change the number of replicas available.

Sablier has a few different modes. It has a "dynamic" mode which will immediately serve a temporary web page that you can customize like so:


FriendlyFaces is just taking a few seconds...

If you're seeing this, it's because @aaronbatilo aka mentallyanimated.com is one cheap mf er


is one cheap mf-er.

I've configured a scale from 0 project on the EKS cluster that serves this page.

After some time of inactivity, we scale to 0. But on first request, you see this page which will auto refresh when the pods are running.

The temporary page is setup to auto-refresh itself until the actual application is up running. I really like this feature. It's great for any website that has static content. T other mode is called "blocking" and unfortunately, this is where I ran into a big problem. Blocking mode waits before forwarding the request. The first request for a "blocking" mode request will actually fail, which I've reported but I can't seem to fi a good fix for. There are some other interesting oddities about sablier. Namely, you configure it by having a specially formatted name. Like… there's already other configuration in this project, why am I configuring things like the Kubernetes namespace of the scale target by joining a string with underscores?

keda-http-addon

The project I ended up sticking with is the keda-http-addon. It's also an experimen project but I've been wanting to play with its core KEDA component for a while. KEDA itself is a set of abstractions for managing event based autoscaling and it's a more flexible than the build in horizontal pod autoscalers. You can use things like S queue attributes as your scale metric, or use database queries to trigger your scale c events. The HTTP addon works as a proxy that queues up HTTP requests. So you configure your ingress controller to send all requests to the "interceptor" and the interceptor holds the requests while your applications scale out. This has a differen problem from sablier though which I still need to figure out. The first request to th HTTP addon takes like 25 seconds to respond, even though the application has bee running after 3-4 seconds.

Edit: On February 7th, my PR for why the first request was taking so long was merged, but time of writing this edit, a new version of the add-on hasn't been released yet. You can ch out the fix here.

BUT, the request doesn't fail completely which is a big deal. For applications like CatStories.ai, it receives webhooks from Twilio when you send text messages. Havi the first request fail like with sablier's "blocking" mode is not going to be great for that use case.

Closing

Another totally random thing to consider. EC2 instances have a max number of IP addresses that it can have. That's actually why I don't use smaller EC2 instances for cluster. So if I have too many applications running, even if they don't use much actu compute, I may be forced to spin up another EC2 instances due to IP address exhaustion. There are ways around this too by using separate CIDRs and what not I don't have those configured… yet.

That's all I've got for this week. Maybe a less exciting experiment than some of the other things I've written about but that's what I played with. Now some of my really low traffic projects like https://terragen.sh and https://catstories.ai might take 25 seconds to load but the sites are up when people want to check them out but they'r not eating up resources when no one is checking them out. And I can keep the link alive on my various internet bios and stuff.

Thanks for reading A slice of experiments!

Type your email... Subscribe

Previous Next

Discussion about this post

Comments Restacks

Write a comment...

Tim Myers Feb 1, 2023

BRB, creating a bot to query all your web pages once a minute.

LIKE (1) REPLY