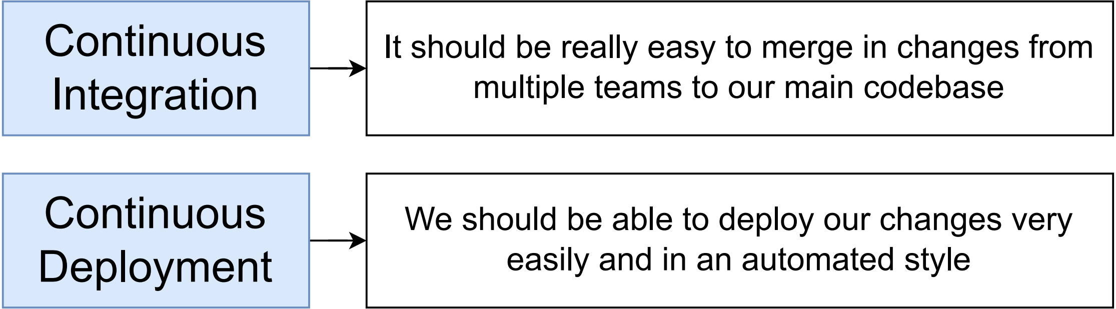
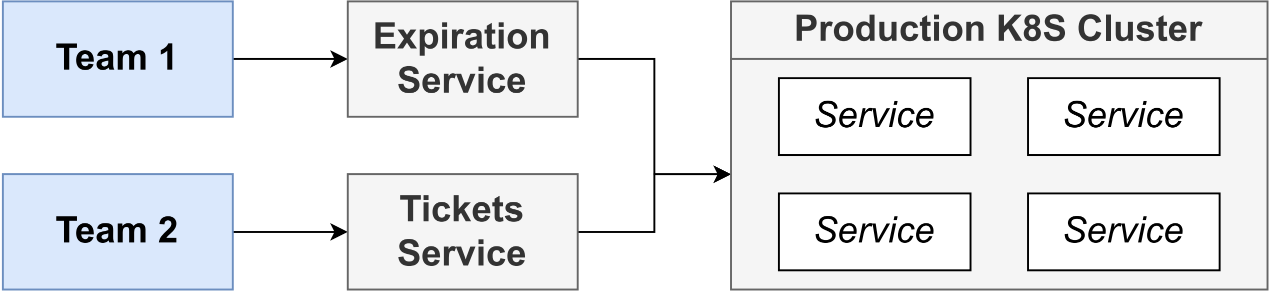
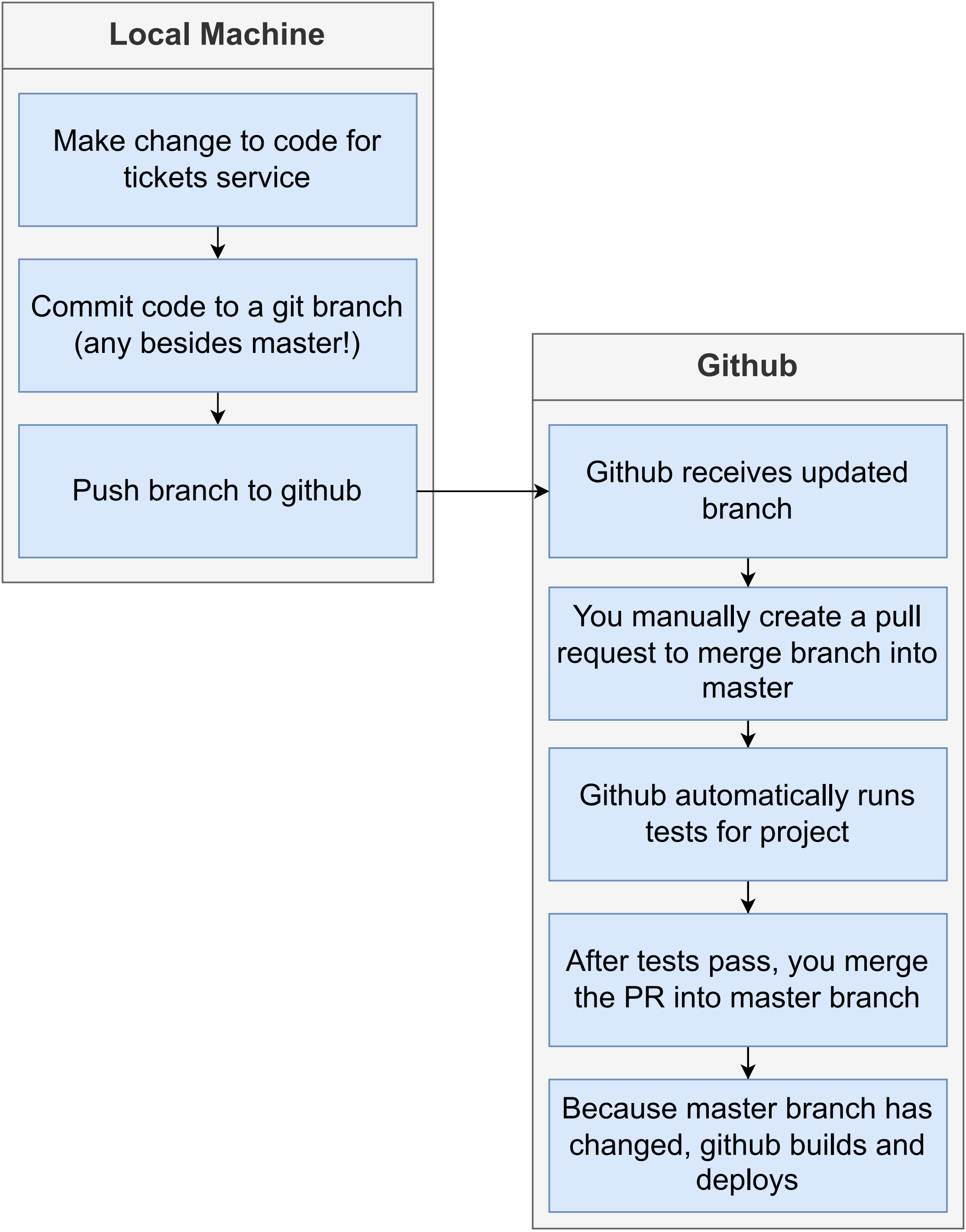
Continuous Integration and Continuous Deployment



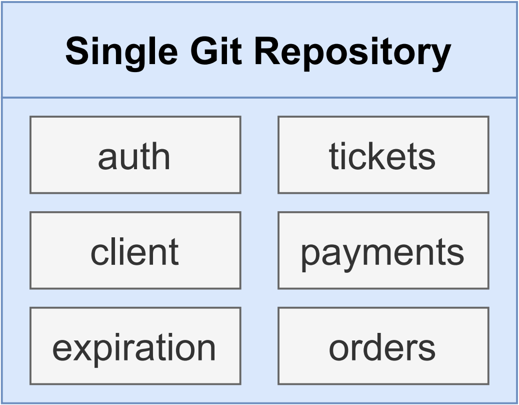
Continuous Integration → It should be really easy to merge in changes from multiple teams to our main codebase

Continuous Deployment → We should be able to deploy our changes very easily and in an automated style


Team 1
Team 2
Expiration Service
Tickets Service
Production K8S Cluster
Service
Service
Service
Service


Local Machine
Make change to code for tickets service
Commit code to a git branch (any besides master!)
Push branch to github
Github
Github receives updated branch
You manually create a pull request to merge branch into master
Github automatically runs tests for project
After tests pass, you merge the PR into master branch
Because master branch has changed, github builds and deploys

Mono Repo Approach



Single Git Repository

auth

tickets

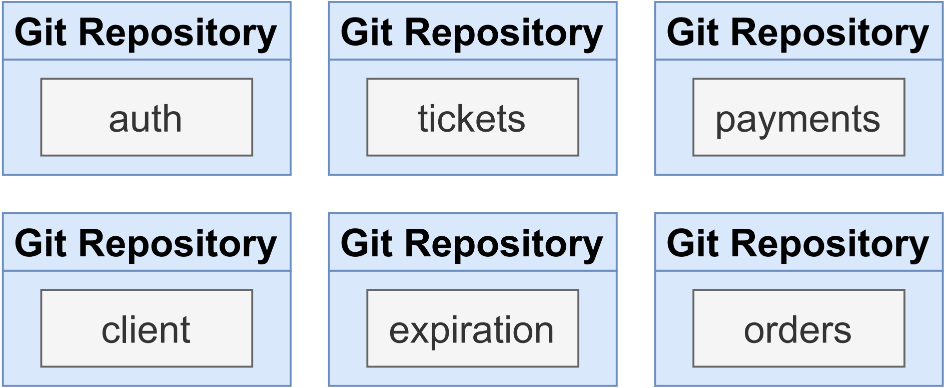
client

payments

expiration

orders

Repo-Per-Service Approach



Git Repository

auth

Git Repository

tickets

Git Repository

payments

Git Repository

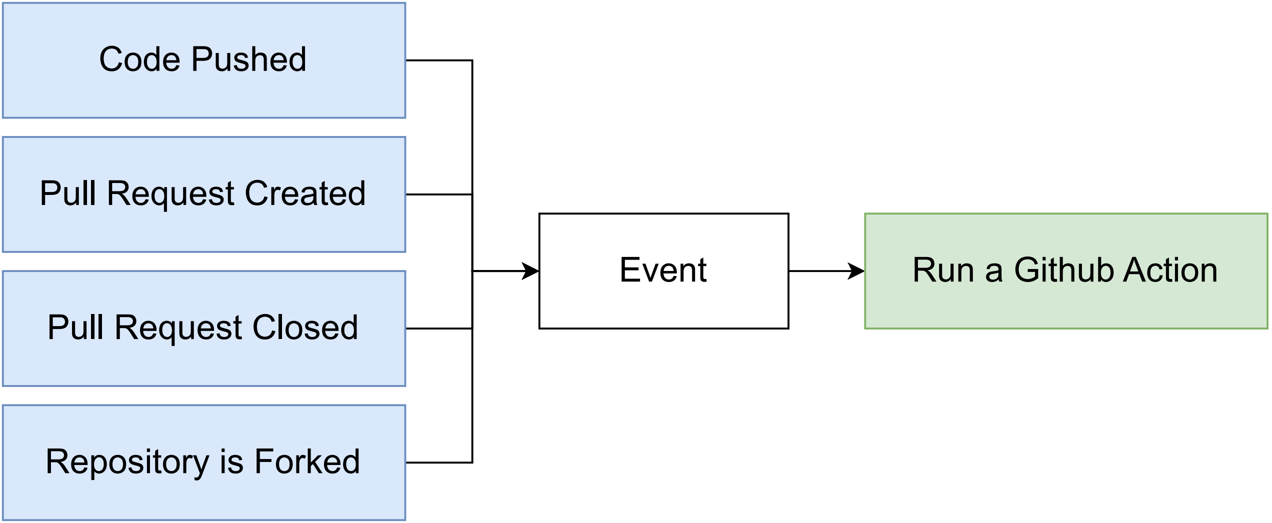
client

Git Repository

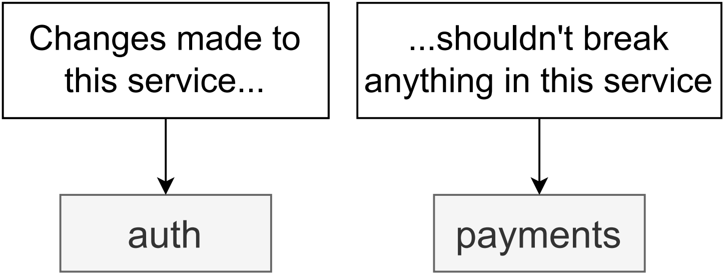
expiration

Git Repository

orders



Code Pushed
Pull Request Created
Pull Request Closed
Repository is Forked
Event
Run a Github Action

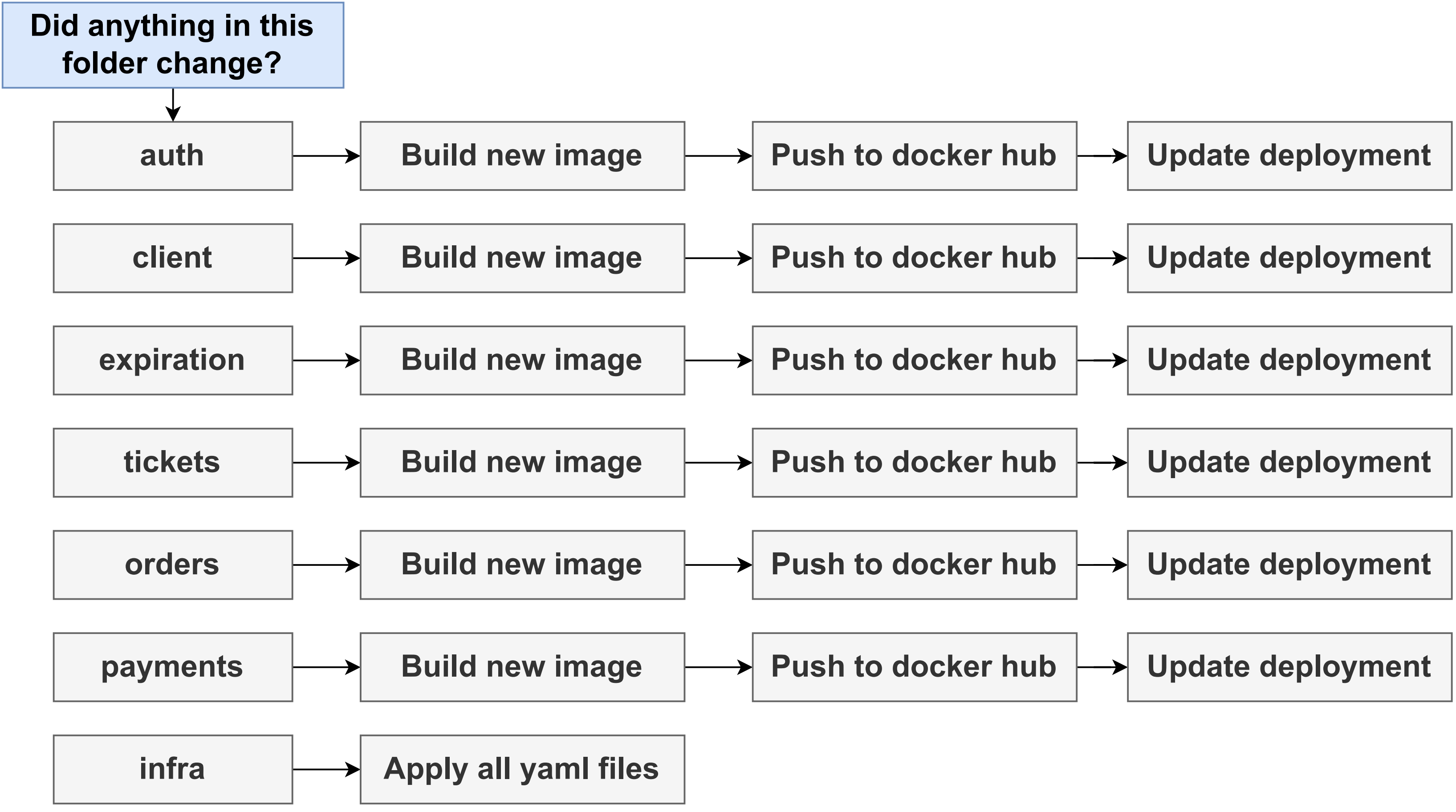


Changes made to this service...

...shouldn't break anything in this service

auth

payments



Did anything in this folder change?

auth → Build new image → Push to docker hub → Update deployment

client → Build new image → Push to docker hub → Update deployment

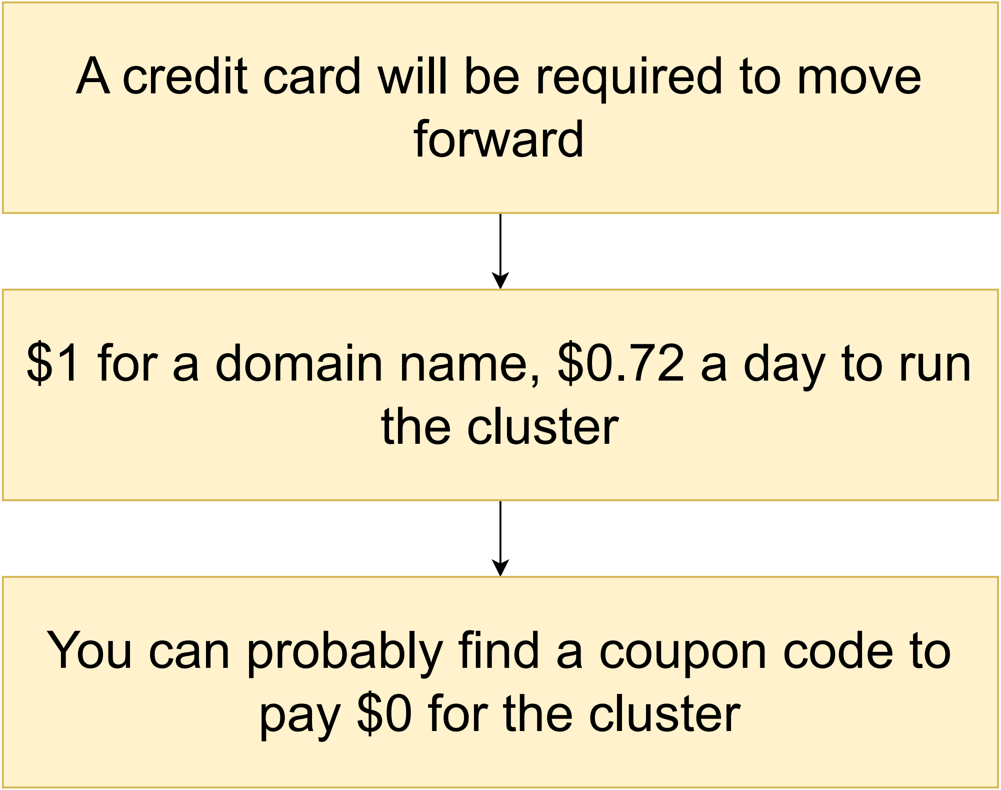
expiration → Build new image → Push to docker hub → Update deployment

tickets → Build new image → Push to docker hub → Update deployment

orders → Build new image → Push to docker hub → Update deployment

payments → Build new image → Push to docker hub → Update deployment

infra → Apply all yaml files



A credit card will be required to move forward

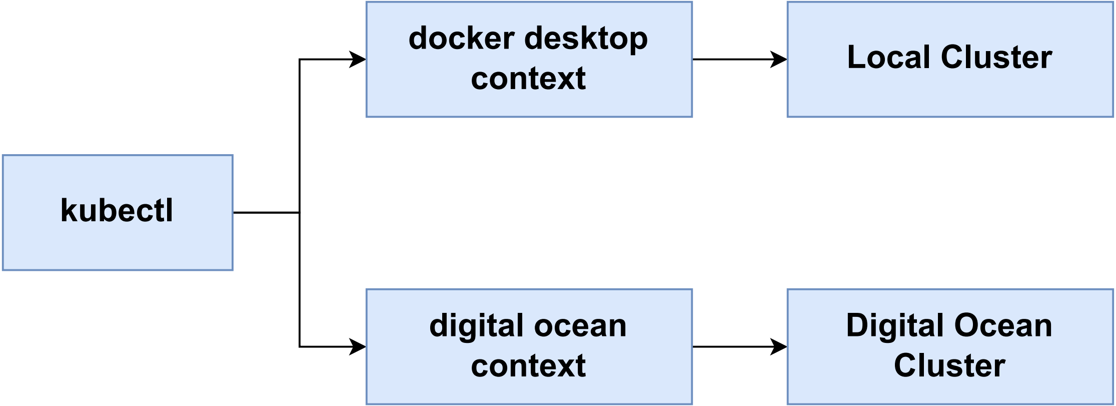
$1 for a domain name, $0.72 a day to run the cluster

You can probably find a coupon code to pay $0 for the cluster

Digital Ocean

$40/month

Really easy to use


kubectl
docker desktop context
Local Cluster
digital ocean context
Digital Ocean Cluster

Authenticating with Doctl

```
doctl auth init
```

Get connection info for our new cluster

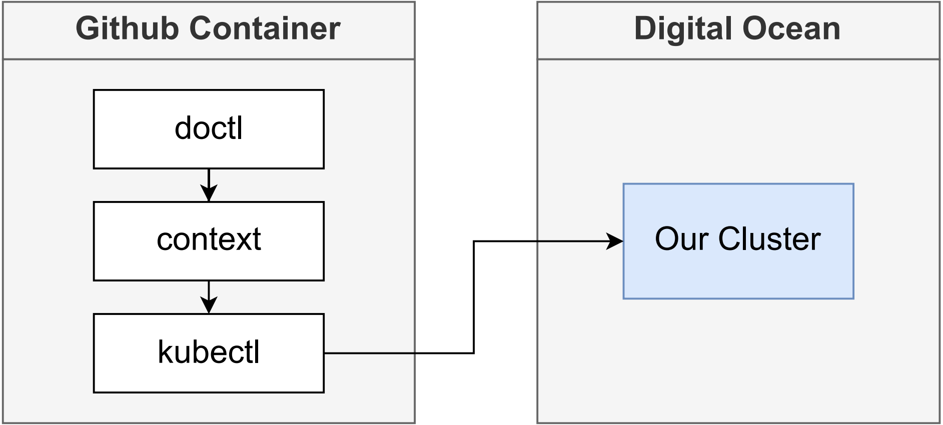
```
doctl kubernetes cluster kubeconfig save <cluster_name>
```

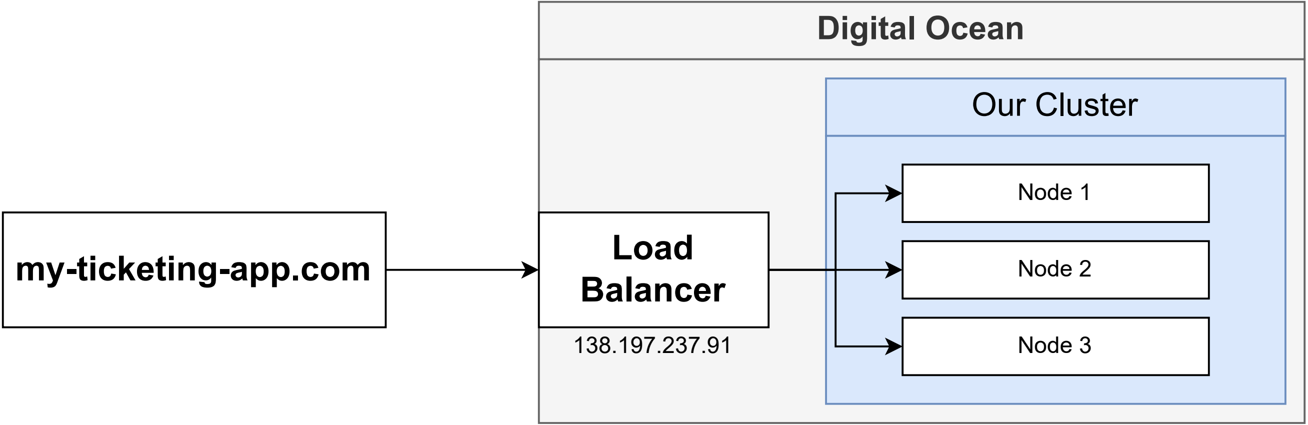
List all contexts

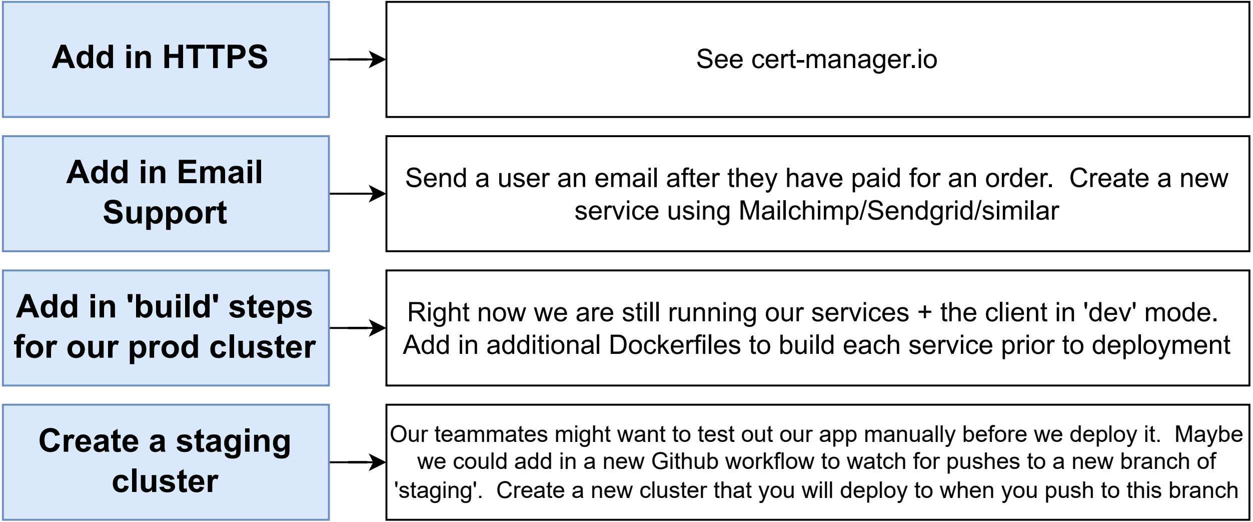
```
kubectl config view
```

Use a different context

```
kubectl config use-context <context_name>
```


Github Container
doctl
context
kubectl
Digital Ocean
Our Cluster


Digital Ocean
Our Cluster
my-ticketing-app.com
Load Balancer
138.197.237.91
Node 1
Node 2
Node 3



Add in HTTPS

See cert-manager.io

Add in Email Support

Send a user an email after they have paid for an order. Create a new service using Mailchimp/Sendgrid/similar

Add in 'build' steps for our prod cluster

Right now we are still running our services + the client in 'dev' mode. Add in additional Dockerfiles to build each service prior to deployment

Create a staging cluster

Our teammates might want to test out our app manually before we deploy it. Maybe we could add in a new Github workflow to watch for pushes to a new branch of 'staging'. Create a new cluster that you will deploy to when you push to this branch