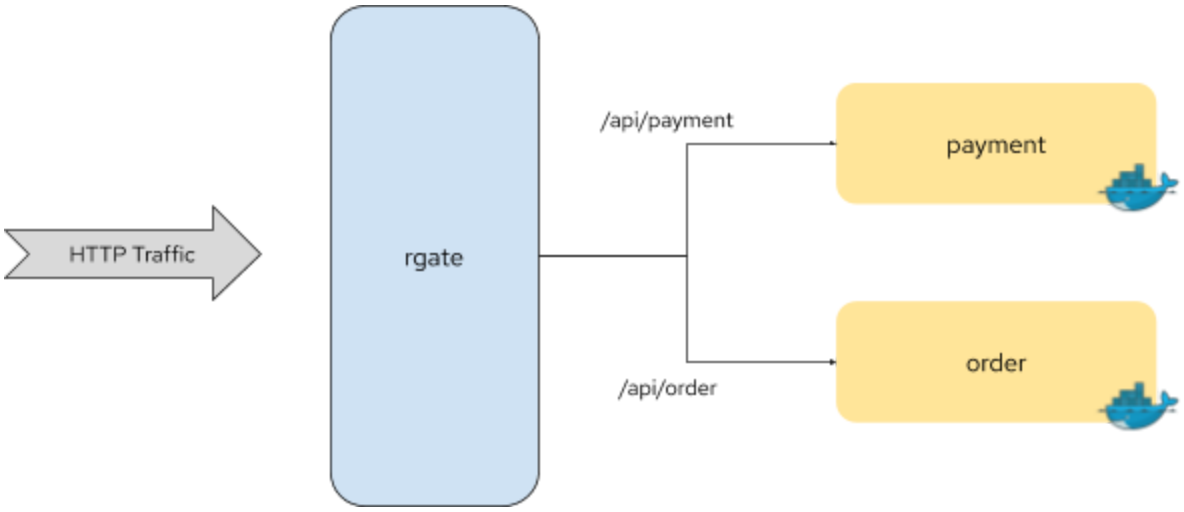
API Gateway for Docker Containers

Problem statement

Design an API Gateway CLI tool, *rgate*



rgate can be run with a command like below to receive traffic on localhost:8080

```
rgate --port 8080 --config config.yml
```

With the contents of config.yml looks like,

```
routes:
   -  path_prefix: /api/payment
      backend: payment
   -  path_prefix: /api/orders
      backend: orders
default_response:
      body: "This is not reachable"
      status_code: 403
```

```
backends:
   -  name: payment
      match_labels:
         -  app_name=payment
         -  env=production
   -  name: orders
      match_labels:
         -  app_name=orders
         -  env=production
```

Features

- Each backend is a container.
- A container is selected as a backend if the container has all the *match_labels* present as Docker labels. If there are multiple containers matching, select a random container as a backend.
- Incoming http requests are routed to the corresponding backend if the path starts with the given *path_prefix*.
- If there are no routes matching the request, it should respond with the given body and status_code in the `default_response`
- If the backend is down, respond with 503 code
- Accessing http://localhost:8080/stats should give us information about the traffic it has received so far. An example JSON response is

```
{
   “requests_count” : {
     “success”: 100, // status codes 200-399
     “error”: 110,   // status codes >399
   },
   “latency_ms”: {
     “average”: 2,
     “p95”: 5,
     “p99”: 10
   }
}
```

Instructions

- Use any language that you are comfortable with
- You are NOT expected to implement any backend yourself. You could use an nginx container in your local to test
- Aim to achieve good unit test coverage
- Ensure README has instructions for installing the tool, testing, usage, etc.. Feel free to add your thoughts on the design choices and assumptions