Round Robin API

Take home assessment

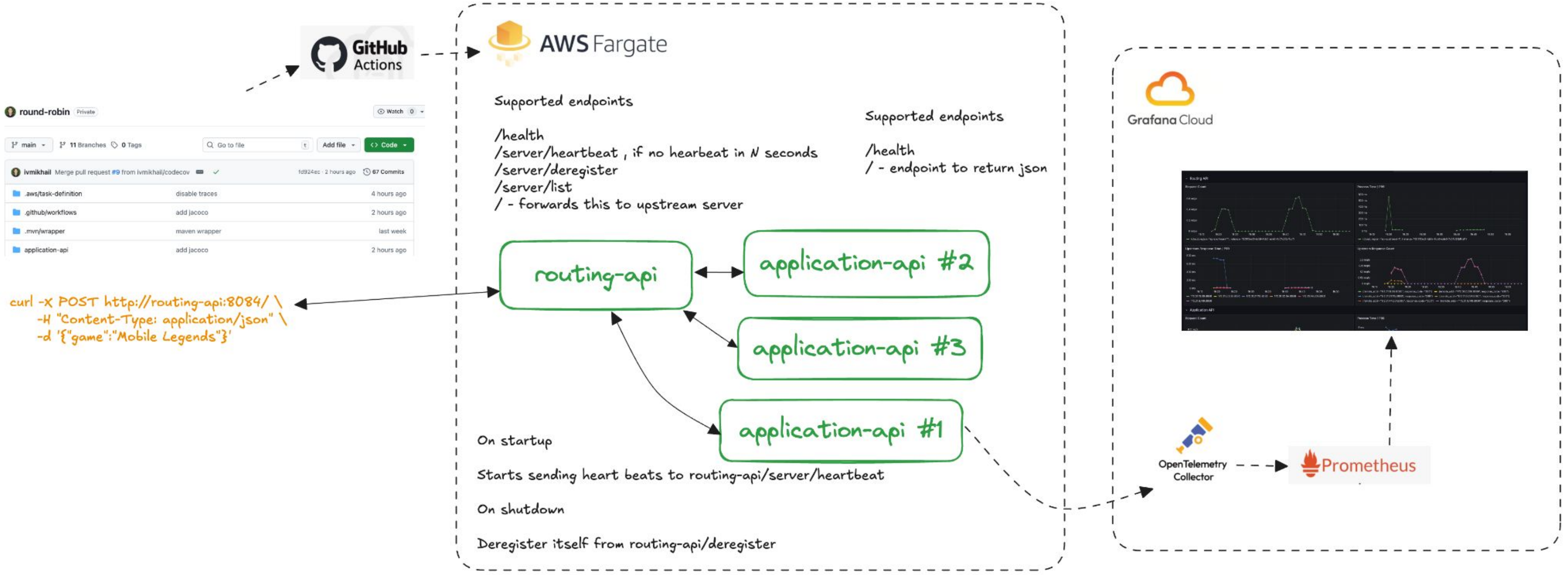
Agenda

- Project overview
- Round Robin logic
- Q&A

Project overview

- Github repository
  - Login: roundrobin1
  - Password: roundrobin99
- Java application, Maven as build tool
- CI/CD jobs
  - Unit tests + code coverage report
  - Deploy
- Deployment to AWS Fargate
  - 1 replica of routing-api and 3 application-api
- Business metrics on Grafana
  - Login: roundrobin1
  - Password: roundrobin99

Diagram



round-robin Private

Watch 0

main · 11 Branches · 0 Tags · Go to file · Add file · Code

ivmikhail Merge pull request #9 from ivmikhail/codecov · fd924ec · 2 hours ago · 67 Commits

.aws/task-definition	disable traces	4 hours ago
.github/workflows	add jacoco	2 hours ago
.mvn/wrapper	maven wrapper	last week
application-api	add jacoco	2 hours ago

```
curl -X POST http://routing-api:8084/ \
   -H "Content-Type: application/json" \
   -d '{"game":"Mobile Legends"}'
```

Supported endpoints

/health
/server/heartbeat , if no hearbeat in N seconds
/server/deregister
/server/list
/ - forwards this to upstream server

Supported endpoints

/health
/ - endpoint to return json

On startup

Starts sending heart beats to routing-api/server/heartbeat

On shutdown

Deregister itself from routing-api/deregister

Round Robin Logic

Routing API

- Maintain application-api servers with CopyOnWriteArrayList
- Each request forwarded to servers[AtomicCounter % size]
- Cleanup job every 20s to remove dead servers (no heartbeat in last 10s)
- Endpoints
  - POST /server/deregister → remove from the pool
  - POST /server/heartbeat → update heartbeat with current timestamp
  - GET /server/list → list all servers

Application API

- Discovers Routing API via DNS (config)
- After startup sends heartbeat every 3s to Routing API
- Calls /server/deregister on shutdown

Summary

- Simple
- Thread-safe with AtomicInteger and CopyOnWriteArrayList
- Observable with metrics and logging

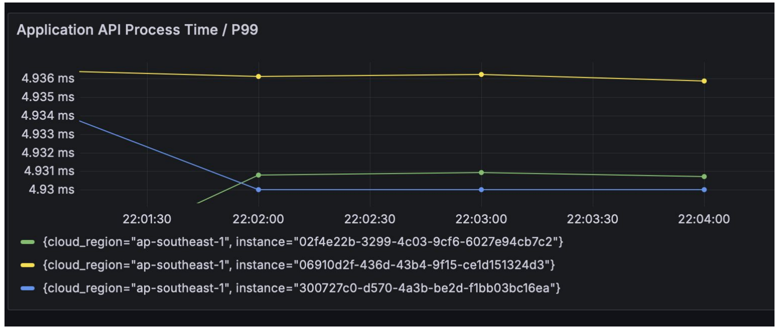
Questions & Answers

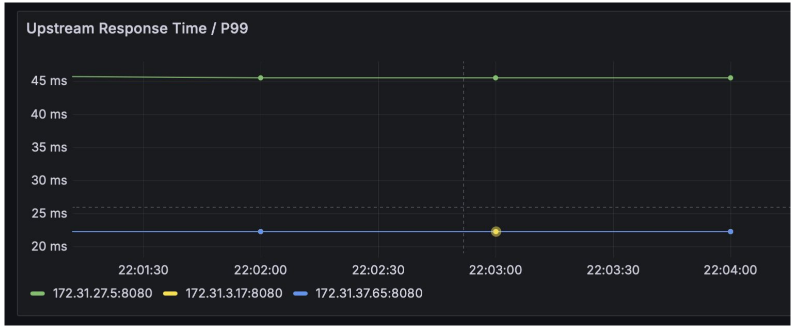
Q: How does the round-robin API handle a situation where one of the application APIs goes down?
A: If a server stops sending heartbeats, the routing API automatically evicts it from the pool

Q: How does it handle a situation where one of the application APIs becomes slow?
A: It doesn't adjust for slowness since all servers are treated equally, but performance is monitored in Grafana

Q: How would you test this application?
A: Through a combination of unit tests and functional tests

Q: Does this implementation support sticky sessions?
A: No, sticky sessions are not supported. To achieve that, we would need to implement consistent hashing or session affinity based on client identifiers


Application API Process Time / P99
4.936 ms
4.935 ms
4.934 ms
4.933 ms
4.932 ms
4.931 ms
4.93 ms
22:01:30
22:02:00
22:02:30
22:03:00
22:03:30
22:04:00
{cloud_region="ap-southeast-1", instance="02f4e22b-3299-4c03-9cf6-6027e94cb7c2"}
{cloud_region="ap-southeast-1", instance="06910d2f-436d-43b4-9f15-ce1d151324d3"}
{cloud_region="ap-southeast-1", instance="300727c0-d570-4a3b-be2d-f1bb03bc16ea"}


Upstream Response Time / P99
45 ms
40 ms
35 ms
30 ms
25 ms
20 ms
22:01:30
22:02:00
22:02:30
22:03:00
22:03:30
22:04:00
172.31.27.5:8080
172.31.3.17:8080
172.31.37.65:8080