Kochava Project :: Postback Delivery

Description:
You will be building a service to function as a small scale simulation of how we at Kochava distribute data to third parties in real time.

Instructions:
1) Your final delivery should be a private repo in Docker Hub (they offer 1 free private repository). Let us know if you'd like to deliver the docker image via a different method, but please keep it private.
2) We'd prefer your image be build from ubuntu:14.04, but not required. Feel free to go from scratch or a different distro.
3) Build a php application to ingest http requests, and a go application to deliver http responses. Use Redis or Kafka to host a job queue between them.
4) Reach out to (Resources - Contact) once your project is ready to demo, or if you encounter a block during development. Pursue independent troubleshooting prior to escalating questions with contact resource.
5) Maintain development notes, provide support documentation, and commit your project (application code / stack config) to a private Github or Gitlab repo.

Extra Credit:
Clean, descriptive Git commit history.
Clean, easy-to-follow support documentation for an engineer attempting to troubleshoot your system.
All services should be configured to run automatically, and service should remain functional after system restarts.
High availability infrastructure considerations.
Data integrity considerations, including safe shutdown.
Modular code design.
Configurable default value for unmatched url {key}s.
Performance of system under external load.
Minimal bandwidth utilization between ingestion and delivery servers.
Configurable response delivery retry attempts.
Data validation / error handling.
Ability to deliver POST (as well as GET) responses.
Service monitoring / application profiling.
Delivery volume / success / failure visualizations.
Internal benchmarking tool.

Data flow:
1) Web request (see sample request) >
2) "Ingestion Agent" (php) >
3) "Delivery Queue" (redis or kafka)
4) "Delivery Agent" (go) >
5) Web response (see sample response)

App Operation - Ingestion Agent (php):
1) Accept incoming http request
2) Push a "postback" object to Redis/Kafka for each "data" object contained in accepted request.

App Operation - Delivery Agent (go):

1) Continuously pull "postback" objects from Redis/Kafka
2) Deliver each postback object to http endpoint:
   Endpoint method: request.endpoint.method.
   Endpoint url: request.endpoint.url, with {xxx} replaced with values from each request.endpoint.data.xxx element.
3) Log delivery time, response code, response time, and response body.

Sample Request:

```
(POST) http://{server_ip}/ingest.php
(RAW POST DATA)
{
  "endpoint":{
    "method":"GET",
    "url":"http://sample_domain_endpoint.com/data?title={mascot}&image={location}&foo={bar}"
  },
  "data":[
    {
      "mascot":"Gopher",
      "location":"https://blog.golang.org/gopher/gopher.png"
    }
  ]
}
```

Sample Response (Postback):

```
GET
http://sample_domain_endpoint.com/data?title=Gopher&image=https%3A%2F%2Fblog.golang.org%2Fgopher
%2Fgopher.png&foo=
```