HTTP Notification Provider

Simple, with a tiny codebase service, on top of the most reliable, Redis-based queue, that is easy to maintain and scale.

Main features

- Messages are batched by URL for high throughput
- Easy to apply different backoff strategies
- Rate limiting

This service consists of several microservices, connected throw a message queue

Technology stack

- Redis - Cache and message queue
- Kafka-connectors - node.js microservice(s)
- HTTP-senders - node.js microservices
- MongoDb - Store user configuration and statistics

Scaling

Depending on the volume of the load, you can choose the following scaling methods: - Add more HTTP-senders, it is quite possible that you will need this in the first place - Add more Kafka connectors, assuming that the data source has enough partitions.
- Place another Redis server next to the existing one, completely separate, share nothing. In this case, you need to split messages by URL between Redis servers to control rate limit, but this is definitely easy to do - You hardly ever need to scale MongoDb. It is only used for statistics and it would be nice not to store it for a long time.

User Interaction

- Step 1 The user selects `RSS` provider from the menu

- Step 2 User submits their URL for notifications. In response, he receives HTML-code, which must be saved on his site. This is the easiest and most familiar way to verify ownership of a website.

  ```
   Please follow the instructions:
  1. Create a file 6361-41ed-427b-98d4-578e.html in the webroot with the following content:
  <html>
   <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    </head>
    <body>Verification: 63361-41ed-427b-98d4-52578e </body>
  </html>
  Call DeBot again and send us just a blank line, after that we confirm your endpoint.
  ```

- Step 3 The user selects `RSS` provider again from the menu and this time just sends him an empty string. This is a signal that he has completed step 2.

- Step 4. The service informs the user about the result of the check and gives him a unique link by which the user can perform some actions that require authorization, for example, get statistics

  ```
   https://your.site.com/webhooks
  URL confirmed, use this link to get your statistics:
  https://toncontest.tmweb.ru/stats?token=5024fa-486e-df83-02a822
  ```