Preparation

- Please do coding and prepare design docs before the pair session.
- We love Go and Postgres. But sure you are free to use any language and database.

Pair session

- We will first discuss your solution.
- A code review session comes after that.
- And then let's talk about further issues and improvements.

Problem brief

As a part of "3rd parties integration", we will implement the **Webhook** feature. It will allow the integration partner to send an HTTP POST to the URL(s) of certain events occuring in Flodesk customer's account. All Webhook events contain data serialized as JSON. See the Developer Documentation for more information on how to format the data.

(To learn more about webhooks, visit Zapier's article)

The APIs to manage webhooks are already implemented as documented here. We still need to design and implement the **Webhook Notifier** component which listens to the events and notifies (i.e. POST) the associated webhooks via the configured URL(s).

Below are the challenges we will need to tackle when building this component:

- ***Reliability***. A retry mechanism should be in place in case the system fails or there are networking issues from integration partners.
- ***Scalability***. The workers should be horizontally scalable depending on events' volume and traffic.
- ***Fairness***. Flodesk's customers have subscribers ranging from a few hundred to hundreds of thousands (for "whale" accounts). Imagine a situation where whale accounts and accounts with a small list are performing subscriber creation at the same time:
    - subscriber.created events from whale accounts can flood the queue and result in long waiting times for other small list accounts.
    - We want to prevent that from happening, to bring the best experience to all our customers.

Tasks

System design

Please provide proper design documents including:

- A high-level diagram showing how components wiring together.
- A document describes your solution and the technologies to be used.

Scope: The system needs to be able to address the *Reliability* and *Scalability* challenges outlined above.

Implementation

The implementation should be able to provide:

- A proof that the concept is feasible, including the core functionalities of the Webhook Notifier component to listen for events, process them, and send HTTP POST requests to configured URLs.
- A way to test and benchmark.

For further discussion during pair session

- **Monitoring.** We need proper monitoring metrics for the performance and latency of the system.
- **Fairness.** Propose a solution to ensure balanced processing of events from both small and large accounts, preventing delays for smaller accounts.

If you have any questions please don't hesitate to reach out!

***Attention:* please ensure that your submitted document and code have no references to our branding.**