Hi! We hope you're doing well!

I am delighted to inform you that we have decided to proceed with your application.

The next phase of the interview process is a test, which we will use to estimate your skill level. The deadline for the assignment is 8 days from today. Your application is solely going to be used to analyse your skills as a software developer, and will not be used for any commercial purposes.

We have a few technical requirements:

- The solution must be implemented in Python. We leave the choice of a framework to you. Here at Sendcloud we use *Django+Django REST Framework*, *Flask* and *FastAPI*.
- Tests are mandatory. We believe in sensible testing rather than achieving 100% coverage, so make sure the important parts of your solution are tested. Our team uses *pytest*, but you may use whichever framework you prefer.
- Your code should be reasonably documented, save for the blocks that are self-explanatory. Usage of docstrings is highly encouraged.
- Provide a specification of your API that will allow us to interact with your application. In Sendcloud we use *OpenAPI* for this purpose, but feel free to choose your own format and tools.
- Wrap your application and all its dependencies in Docker container(s). This requirement will showcase your understanding of the modern way of deploying projects. Describe clearly the build steps in your README, so our developers may review hassle-free.
- Submit your code through the link of the e-mail that we've sent (with this doc).

Here is some advice to help you get started:

- The requirements are simple, but keep in mind that we're looking for a good developer with solid software architecture knowledge. Try to avoid over-engineering.
- Develop as if what you're delivering would be *deployed to production*. This means your solution should be correct, secure and devoid of major bugs.
- Please write clean & tidy code. Within our backend team, we enforce strict PEP8 compliance.
- We will evaluate the *quality* of your final product more than the *speed* of delivery. We have estimated that, on average, this project takes around ~8 hours of pure development time, so take your time and do things nicely.

The test is as follows:

Create a simple **RSS scraper** application which saves RSS feeds to a database and lets a user view and manage feeds they've added to the system **through an API**. Think of Google Feedburner as an example.

Functional requirements (your test will not be accepted unless ALL the following points are fulfilled):

- Support *at least* the following feeds:
  - http://www.nu.nl/rss/Algemeen
  - https://feeds.feedburner.com/tweakers/mixed
- A user of this API should be able to:
  - Follow and unfollow multiple feeds
  - List all feeds registered by them
  - List feed items belonging to one feed
  - Mark items as read
  - Filter read/unread feed items per feed and globally (e.g. get all unread items from all feeds or one feed in particular). Order the items by the date of the last update
  - Force a feed update
- Feeds (and feed items) should be updated in a background task, asynchronously, periodically and in an unattended manner. We use Dramatiq for Sendcloud tasks, but you may use whichever solution you're comfortable with (e.g. asyncio, Celery).
- Implement a back-off mechanism for feed fetching
  - If a feed fails to be updated, the system should fall back for a while.
  - After a certain amount of failed tries, the system should stop updating the feed automatically.
  - Users should be notified and able to retry the feed update if it is stalled after these failed tries.
  - Simulate this behaviour in a test case.

We're looking forward to seeing the results. If you have any questions, feel free to drop us a message.

Good luck!