DigiCloud Programming Challenge

We are delighted to inform you that we have decided to proceed further with your application!

The next step is to write a small app in **Python** with whatever library or framework you prefer such as Django or Flask that showcases your skills as a developer. You have **one week** to do it - take your time and do things nicely. Your app will not be used for any commercial purposes and is solely going to be used to analyze your skills as a software developer. We estimate that this project should take around ~20 hours of pure development time, so you can do it in a few days.

Bonus: If you feel adventurous, you can develop the app in a microservice style with the Nameko framework. Developing your app with Nameko is a **big plus**, but please note that if you get things wrong, it will have a negative impact on our assessment of the project. So if you're not confident about microservices and their challenges, you're better off going the normal way.

We can give you some advice beforehand:

- Be tidy with your code. We like clean code over here.
- The build system of the app matters. We use Docker extensively, so please Dockerize your application and its stack.
- The requirements are **really** simple but remember we're looking to figure out how good of a developer you are. Avoid over-engineering things but make sure to show solid software engineering knowledge.

Please host your app in a Git repository somewhere and give us access once it is done.

Description

The idea is to create a simple RSS scraper which saves RSS feeds to a database and lets a user view and manage his feeds via a simple API. Suppose that you're developing a backend service for a feed aggregator such as Feedly. The API should expose all the

necessary functionality to view all feeds registered by the user, the amount of unread entries in his feeds and the ability to comment on a feed item.

Some functionality which should be there:

- User account creation and login
- Support for following multiple feeds
- Ability to add a bookmark/favourite to a feed item. Those bookmarks/favorites should be private for a user.
- Feeds must be updated asynchronously. You can use Celery, asyncio or any other async solutions you know.
- Implement a back-off mechanism in case the system fails to update the feed. Simulate this behaviour in your testcase.
- Proper testing is mandatory. We mostly use pytest, but other test frameworks are fine too (nose, Django TestCase, etc.)
- Feel free to design the architecture of the app as you prefer.

Looking forward to seeing the results. We wish you the best of luck!