Tech Challenge for Consulting

Build a Camunda Animal Picture App

Description

Build a simple Camunda Client App for Camunda 8, that gets a random picture of a cat, a dog, or a bear. A list of free APIs can be found here: https://github.com/public-apis/public-apis. We suggest:

- Cats: https://placekitten.com/
- Dogs: https://place.dog/
- Bears: https://placebear.com/

Main Tasks

1. Create a process model (BPMN) that reflects the following
    a. Based on a user's selection the process fetches a picture of either a cat, a dog or a bear
2. Build a client app that
    a. Deploys the process model to a Camunda installation
    b. Has a custom REST API to
        i. Start a process instance of the Camunda Process
        ii. Retrieve the picture
    c. Implements a Job worker that
        i. Fetches the picture from an API
        ii. Stores the picture in a DB of your choice
    d. Add automated tests
3. Containerize your application
4. Add a setup to make the app run locally on every machine
    a. Think of providing a way to configure the access to different Camunda clusters so that your app is runnable from anywhere with a provided configuration
5. Add README that includes
    a. All relevant documentation about your app
    b. A how-to part on how to run the app locally
    c. A small architectural diagram that shows how these components interact with each other

Bonus points:

- Add a HELM chart for a Kubernetes Deployment
- Simple UI to request and then show the picture (no fancy CSS needed)

You are free to choose languages and frameworks as you wish.

Share your solution by inviting << the hiring manager >> into your Camunda Organization in Camunda SaaS and by creating a GitHub Repository for your code.

Hints

- You can use the Camunda Saas Offering to get started with our Product, there is a 30-day free trial without any cost - camunda.io/signup
- Check out our guides: https://docs.camunda.io/docs/guides/
    - See also: https://github.com/camunda/camunda-platform-get-started
- A “Job-Worker” is a piece of software that interacts with the Process Engine (= Zeebe) directly, more details on the Job-Workers: https://docs.camunda.io/docs/components/concepts/job-workers/
- Interacting with the Process Engine (=Zeebe) is done via a gRPC API. For more details on that API please see: https://docs.camunda.io/docs/apis-tools/grpc/
    - A full overview of APIs and Clients: https://docs.camunda.io/docs/apis-tools/working-with-apis-tools/
    - Postman collection: https://www.postman.com/camundateam
    - Community clients for other languages than Java, Go, and the CLI client: : https://docs.camunda.io/docs/apis-tools/community-clients/
- For testing purposes, there are some supporting libraries available for example
    - Java: https://github.com/camunda/zeebe-process-test
    - *Hint*: it is completely fine if you choose a language you are comfortable with even if there is no test support for Camunda for that specific language. Just add some tests that test your code without testing the process directly.

We at Camunda are perfectly aware that you might not be familiar with the product and that it takes time to get started. We advise you to keep things as simple as possible and not invest too much time. We know that people have jobs and therefore advise that the amount of time invested should not exceed more than 6 hours in total. If you struggle to finish all the tasks in time feel free to submit the open tasks in written form (let’s say you are missing tests then write a short explanation on how you would approach testing, what libraries you would use, etc.).