DevOps Task

Nasi programiści przygotowali projekt i potrzebują pomocy DevOpsa w zautomatyzowaniu kilku rzeczy, żeby mogli pokazać aplikację klientowi.

Źródła projektu są do pobrania stąd. To prosta aplikacja, która po uruchomieniu udostępnia trzy endpointy REST-owe:

- GET "/status" pokazuje status aplikacji, gdy działa zwraca JSON {"status": "ok"}
- GET "/environment" zwraca nazwę środowiska, na którym jest uruchomiona aplikacja, wziętą ze zmiennej środowiskowej SML_ENV, format odpowiedzi: {"environment": "staging"}
- POST "/dowork?magicnumber=[number]" zwraca rezultat działania logiki aplikacji

Przydatne polecenia:
- sbt compile test - kompiluje i odpala testy
- sbt assembly - buduje wykonywalny plik jar i wypisuje ścieżkę do zbudowanego pliku
- java -jar <scieżka_do_jara> - uruchamia aplikację na porcie 8081

Czego od Ciebie oczekujemy to w skrócie: publikacji źródeł projektu w repozytorium, dockeryzacji aplikacji oraz automatyzacji tworzenia dwóch środowisk, na których zostanie uruchomiony obraz dockera, stworzenia pipeline CI/CD oraz monitoringu czy aplikacja działa. Szczegóły odnośnie każdej z tych rzeczy poniżej.

Task 1: Publikacja źródeł w repozytorium

Umieść źródła aplikacji w repozytorium (na GitHub, Bitbucket, GitLab, itp.). Możesz założyć konto na potrzeby tej rekrutacji i podesłać nam dane dostępowe, możesz udostępnić **prywatne** repozytorium naszemu kontu (klucz SSH poniżej).

ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQC3LiQp4rhANmCrcrOfF0NWZZabWqCOipKgPWUpQ9MOJYB6lPJjSn
2BW7QR49NUOlGse1zBO8EgTs8kjmlAbbxrHQJqmV9WfEBnLBscqrKU9Z7pkp3yaolWbcZSaJPX8xY7Nsn7D9U
g7/9OF0YIII4VMNm76l2Q56st8SB8BW7Ww70FVXDFrCHQ38+mXcRlHXjHG6aCN+arJqNKPZVUKqNU/qhAVb
+Ajpb/CrURkCbJwqz8WI4pRWb8zDFNQDqkQPASNXOenvEwsDQKiKcfbV4KFBV60qLGQSQrPz/3gunP/oyuqCl
gQpYYXkE8nMST2pyn2wDMD0T671n+0hgdHzCP

Task 2: Dockeryzacja aplikacji oraz automatyczne tworzenie dwóch środowisk do deploymentu

Stwórz w tym samym repozytorium kod definiujący obraz dockera i publikujący go w repozytorium oraz opisujący/tworzący infrastrukturę, na której będzie deployowany serwis.

Wymagania szczegółowe:
- Serwis (nasza aplikacja) ma działać jako kontener dockera.
- Obraz dockera ma być publikowany w jakimś repozytorium
- Serwis ma mieć możliwość zdeployowania na środowisko staging (gdzie ustawiona powinna zostać zmienna systemowa SML_ENV = staging) i production (SML_ENV = production).
- Logi z serwisu powinny być zachowywane niezależnie od życia kontenera. Do tego celu możesz użyć gotowego narzędzia as a service lub przygotować self hosted.
- Tworzenie infrastruktury dla obu środowisk powinno być udokumentowane i zautomatyzowane (z dużą przewagą automatyzacji nad dokumentacją).
- Do wyboru jest AWS, Google Cloud Platform i Azure.
- Jeśli zakładasz obecność jakichś zmiennych systemowych, które są potrzebne, aby całość działała (credentiale itp.), proszę dodaj o tym informację w Readme.md.

Task 3: Pipeline CI/CD

Stwórz pipeline (opisany jako kod) do budowania i deployowania aplikacji. Możesz użyć gotowego rozwiązania dostarczanego przez serwis hostujacy repozytorium (GitHub, Bitbucket, GitLab, itp.) albo wykorzystać inne, open-source'owe narzędzie (Jenkins, Hudson, etc.)
Uwaga: jeśli zdecydujesz się użyć jakiegoś open-source'owego rozwiązania, dostarcz wszystko, co jest potrzebne, aby łatwo było nam sprawdzić Twoje rozwiązanie. Może to być jakaś działająca instancja, do której udostępnisz nam adres, login i hasło, może być obraz Docker ze skonfigurowaną instancją lub cokolwiek innego, co pozwoli na łatwe zweryfikowanie, że Twoje rozwiązanie działa.
Chodzi nam o dostarczenie gotowej, działającej instancji tego narzędzia, aby osoba sprawdzająca mogła łatwo zweryfikować czy pipeline działa w oczekiwany sposób.

Wymagania szczegółowe:

- Aplikacja jest testowana i budowana po każdym commicie do dowolnego brancha
- push/merge do mastera wykonuje dodatkowo następujące kroki:
    - buduje obraz dockerowy
    - taguje go w sensowny sposób i pushuje do repozytorium
    - deployuje obraz na stagingu ustawiając odpowiednio SML_ENV i sprawdza, czy aplikacja uruchomiła się poprawnie
    - deployuje obraz na produkcji ustawiając odpowiednio SML_ENV i sprawdza, czy aplikacja uruchomiła się poprawnie

Task 4: Monitoring i notyfikacje

Przygotuj mechanizm, który monitoruje poprawne działanie aplikacji, a w razie problemów wysyła notyfikację (dowolny sposób: e-mail, wiadomość na komunikatorze, sms, coś innego) do konfigurowalnego odbiorcy.

Co podlega ocenie:

1. Czy wszystkie elementy działają zgodnie ze specyfikacją
2. Łatwość odtworzenia całej infrastruktury lokalnie przez programistę z niedużym doświadczeniem DevOpsowym
3. Wysoki poziom automatyzacji rozwiązania
4. Czytelność i zwięzłość dostarczonej dokumentacji
5. Dobór technologii zgodnie z potrzebami projektu oraz własnymi kompetencjami.

Rozwiązanie traktujemy jako zamkniętą całość i rzadko prosimy kandydatów o poprawki rzeczy, do których mamy zastrzeżenia, więc prosimy o dostarczanie kompletnego rozwiązania, nawet kosztem przekroczenia terminu o kilka dni. A wszelkie założenia, które zostały przyjęte przy rozwiązywaniu, a mogą mieć wpływ na ocenę, prosimy umieszczać w pliku README.md.

Pytania do zadania:
Please write your answers in English in README.md

1. What technologies have you chosen to implement the solution? Why?
2. How much time did you spend on this task? Could you please divide this number into a few most important areas?

Informacje końcowe:
Ewentualne pytania (jesteśmy Twoim Klientem) oraz informację, że zadanie jest gotowe do oceny prosimy wysyłać na czlowieki@softwaremill.com z cc do osoby, która przeprowadziła z Tobą pierwszą rozmowę.

Powodzenia! :)


SOFTWAREMILL