Déploiement d'une application spring-boot avec Docker vers le cloud Heroku

Heroku est une plateforme cloud qui offre des services d'hébergement et de déploiement d'applications. Voici quelques caractéristiques et concepts clés de la plateforme Heroku :

Facilité de Déploiement : Heroku simplifie le processus de déploiement d'applications en permettant aux développeurs de se concentrer sur leur code sans se soucier de la gestion de l'infrastructure sous-jacente.

1. **Prise en Charge de Divers Langages :** Heroku prend en charge plusieurs langages de programmation, y compris Java, Ruby, Python, Node.js, PHP, et d'autres. Cela permet aux développeurs de choisir le langage qui convient le mieux à leurs besoins.
2. **Gestion des Conteneurs :** Heroku utilise des conteneurs pour isoler les applications et simplifier le déploiement. Les développeurs peuvent utiliser des conteneurs Docker ou déployer directement à partir de leur référentiel de code.
3. **Automatisation du Scaling :** Heroku propose des options de scaling automatique pour gérer la charge de l'application. Les applications peuvent évoluer horizontalement en ajoutant des instances en fonction de la demande.
4. **Add-ons :** Heroku propose une variété d'add-ons qui permettent d'étendre les fonctionnalités de l'application, notamment des bases de données, des services de cache, des systèmes de gestion de logs, etc.
5. **Gestion des Dépendances :** Heroku permet aux développeurs de spécifier les dépendances de leur application, ce qui facilite la gestion des environnements d'exécution.
6. **Intégration Continue :** Heroku s'intègre facilement avec des outils d'intégration continue tels que Jenkins, Travis CI, etc., facilitant ainsi le déploiement continu.
7. **Évolutivité :** La plateforme Heroku est conçue pour être évolutive, permettant aux applications de gérer une croissance significative du trafic.
8. **Environnements de Développement et de Production :** Heroku propose des environnements distincts pour le développement et la production, facilitant le test avant le déploiement.
9. **Command Line Interface (CLI) :** Heroku fournit une interface en ligne de commande (CLI) qui permet aux développeurs de gérer leurs applications, de déployer du code, et d'effectuer d'autres tâches directement depuis le terminal.

En résumé, Heroku offre une solution cloud flexible et facile à utiliser pour le déploiement et la gestion d'applications, ce qui permet aux développeurs de se concentrer sur le développement de leur code sans se soucier des détails d'infrastructure.

Installer Docker : Voir doc correspondante.

Installer Heroku : Voir Doc correspondante.

Ouvrir Docker Desktop et connectez vous

Ajouter le fichier Dockerfile à la racine du projet spring-boot :

```
FROM adoptopenjdk/openjdk15:alpine-jre

ARG JAR_FILE=target/*.jar
COPY ${JAR_FILE} app.jar

ENTRYPOINT ["java","-jar","/app.jar"]
```

Remarque : Utiliser la même version du jdk installé localement que celui du fichier Dockerfile.
Pour chercher les jdk compatibles docker, lancer la commande :
docker search adoptopenjdk
Il faut se connecter à docker d'abord.

Lancer un « mvn clean install » afin de préparer la dockerisation de l'application.
Admettons qu'elle s'appelle : test-spring-boot
Remarque : N'utilisez pas les caractères « _ » (underscore) dans le nom. Ce n'est pas accépté.
Lancer les commandes suivantes dans la fenetre cmd windows :
set app=test-spring-boot
set img=%app%

#Se connecter à Heroku :
heroku login
#Se connecter au Contenaire de Heroku :
heroku container:login

Ajouter l'application cible dans Heroku :
heroku create %app%

construction de l'image docker :
docker buildx build . -t %img%

Envoyer l'image docker vers la registry du contenaire de Heroku :
docker tag %img% registry.heroku.com/%app%/web
docker push registry.heroku.com/%app%/web

#Libérer l'image docker vers l'application Heroku :
heroku container:release web -a %app%

Ouvrir l'application :

heroku open -a %app%

Voir les logs

heroku logs --tail -a %app%

Historique de la plateforme cloud Heroku :

Heroku a une histoire intéressante en tant que plateforme de cloud computing qui a évolué au fil des ans. Voici un bref historique de Heroku :

1. **Fondation (2007) :**
    - Heroku a été fondé en 2007 par James Lindenbaum, Adam Wiggins, and Orion Henry.
    - Le but initial était de simplifier le processus de déploiement des applications web en permettant aux développeurs de se concentrer sur le code sans se préoccuper de la gestion de l'infrastructure.
2. **Introduction de Ruby on Rails (2007) :**
    - Heroku a d'abord été lancé avec un support fort pour Ruby on Rails, un framework de développement web en Ruby.
3. **Acquisition par Salesforce (2010) :**
    - En 2010, Salesforce.com a acquis Heroku pour environ 212 millions de dollars.
    - Heroku est resté relativement indépendant au sein de Salesforce, continuant à opérer en tant qu'entité distincte.
4. **Expansion des Langages de Programmation (2011) :**
    - Heroku a élargi sa prise en charge pour inclure plusieurs autres langages de programmation, dont Python, Java, Node.js, et plus encore.
    - Cette expansion a permis à Heroku de devenir une plateforme plus polyvalente et d'attirer une base d'utilisateurs plus large.
5. **Introduction des Conteneurs Docker (2015) :**
    - En 2015, Heroku a introduit la prise en charge des conteneurs Docker, permettant aux développeurs de déployer des applications dans des conteneurs standardisés.
6. **Passage à l'AdoptOpenJDK (2019) :**
    - En 2019, Heroku a annoncé son passage à AdoptOpenJDK pour les images Java, en remplacement des distributions d'OpenJDK précédemment utilisées.
7. **Transition vers Eclipse Adoptium (2021) :**
    - L'AdoptOpenJDK project a été renommé Eclipse Adoptium en 2021.
    - Heroku continue de fournir des images OpenJDK via Eclipse Adoptium pour les applications Java déployées sur la plateforme.

Heroku est devenu une solution populaire pour les développeurs cherchant une plateforme de déploiement rapide et facile, en particulier pour les applications web et mobiles. Son intégration transparente avec des technologies telles que Git, sa facilité d'utilisation et son modèle de facturation basé sur la consommation ont contribué à sa popularité au fil des ans.