



TP 2 : Introduction à Docker Dans un Contexte DevOps

DevOps

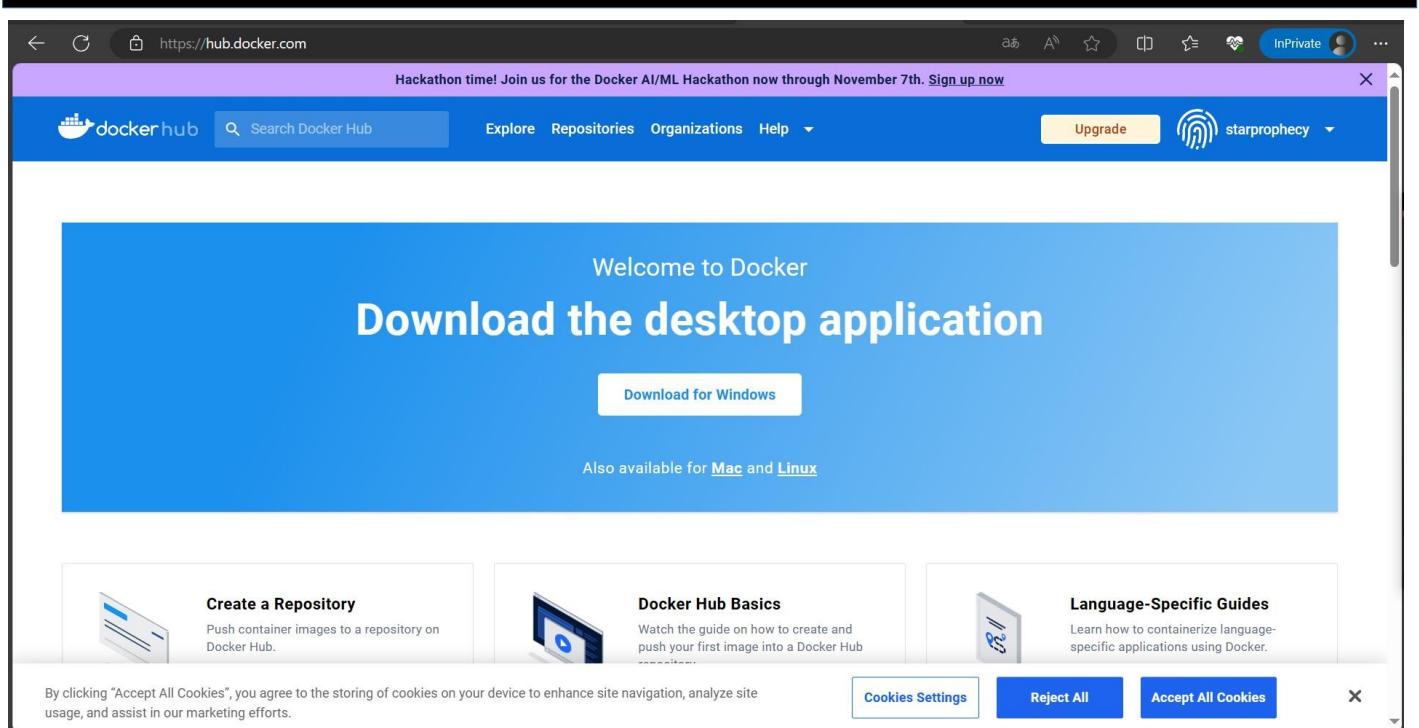
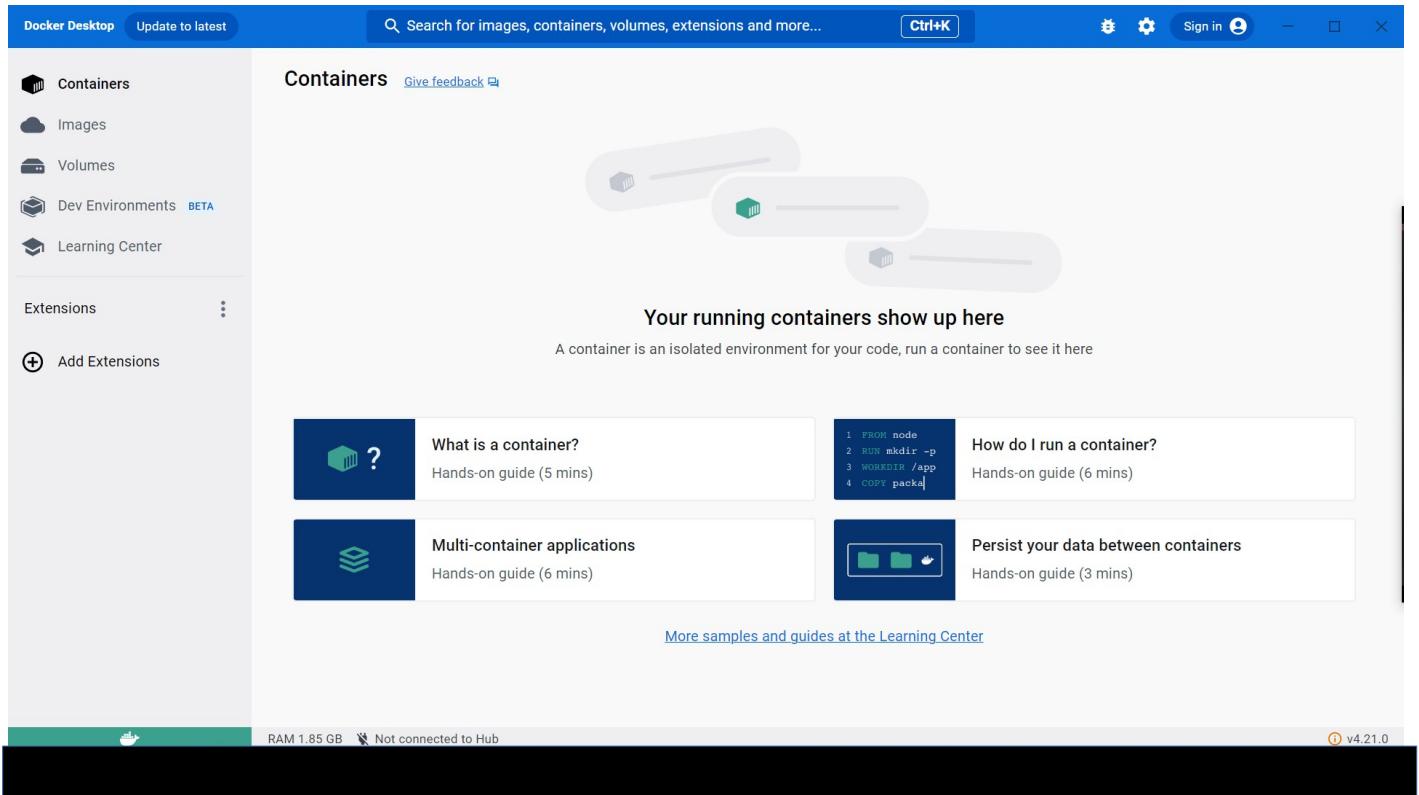
RT5_groupe2



- Réalisé par:

- Gharsallah Ahmed
- Bouchhiwa Hassen
- Ben Jemaa Mouhib

1- Introduction à docker



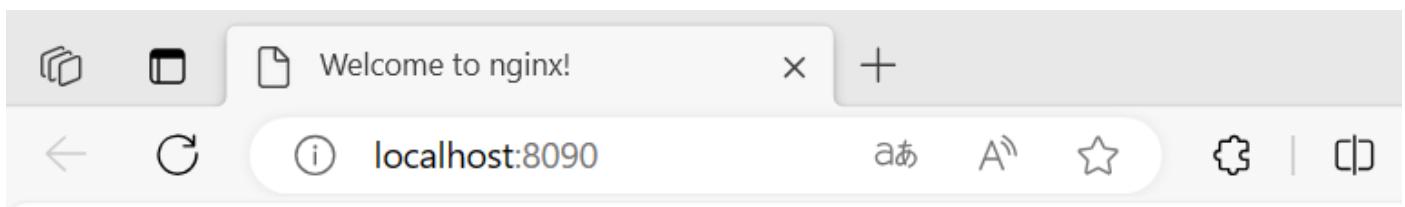
2- Utilisation de docker

- Création d'un conteneur Docker à partir d'une image existante : nginx port 8090

```
Sélection C:\Windows\System32\cmd.exe
Microsoft Windows [version 10.0.19045.3448]
(c) Microsoft Corporation. Tous droits réservés.

[docker run -d -p 8090:80 --name nginx8090 nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
a378f10b3218: Pull complete
4ffff0708538: Pull complete
2135e49ace4b: Pull complete
c843f6b280ce: Pull complete
6f35ab6f1400: Pull complete
6c538b49fa4a: Pull complete
d57731fb9008: Pull complete
Digest: sha256:b4af4f8b6470feb45dc10f564551af682a802eda1743055a7dfc8332dfffa595
Status: Downloaded newer image for nginx:latest
74911f6f05967bb4fcdd5d9cf1f00f8058426b754b03bec399cf56f8c08e76f
```

-Test et validation du fonctionnement du conteneur nginx



- Exploration du contenu du Conteneur :

```
[root@74911f6f0596:/# $>
$> docker exec -it nginx8090 bash
```

```
C:\Windows\System32\cmd.exe - docker exec -it nginx8090 bash
root@74911f6f0596:/home# cd ..
root@74911f6f0596:/# ls
bin dev docker-entrypoint.sh home lib32 libx32 mnt proc run srv tmp var
boot docker-entrypoint.d etc lib lib64 media opt root sbin sys usr
root@74911f6f0596:/# cd home
root@74911f6f0596:/home#
```

- Création d'un deuxième Conteneur nginx sur le port 8091

```
C:\Windows\System32\cmd.exe
root@74911f6f0596:/home# cd ..
root@74911f6f0596:/# ls
bin dev docker-entrypoint.sh home lib32 libx32 mnt proc run srv tmp var
boot docker-entrypoint.d etc lib lib64 media opt root sbin sys usr
root@74911f6f0596:/# cd home
root@74911f6f0596:/home# exit
exit

>docker run -d -p 8091:80 --name nginx8091 nginx
c8c54dcff4d227ba03668dc99df1038c04449052b43bc13f960f90fd8a14634f
>
```

The screenshot shows a web browser window with the following details:

- Address Bar:** localhost:8091
- Page Content:**

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org. Commercial support is available at nginx.com.

Thank you for using nginx.

- Gestion des conteneurs Docker : liste,démarrage, arrêt, suppression.

*- Liste des conteneurs Docker en cours d'exécution :

```
>docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
c8c54dcff4d2 nginx "/docker-entrypoint..." 8 minutes ago Up 8 minutes 0.0.0.0:8091->80/tcp nginx8091
74911f6f0596 nginx "/docker-entrypoint..." 14 minutes ago Up 14 minutes 0.0.0.0:8090->80/tcp nginx8090
```

*- Arrêter un conteneur Docker :

```
>docker stop 74911f6f0596  
74911f6f0596  
  
>docker stop c8c54dcff4d2  
c8c54dcff4d2  
  
>docker ps  
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES  
74911f6f0596 74911f6f0596 /bin/sh -c curl -X POST -d "text=Hello world" http://127.0.0.1:8090/api/greet  
c8c54dcff4d2 c8c54dcff4d2 /bin/sh -c curl -X POST -d "text=Hello world" http://127.0.0.1:8090/api/greet  
>
```

*- Démarrer un conteneur Docker :

```
>docker start c8c54dcff4d2  
c8c54dcff4d2  
  
>docker start 74911f6f0596  
74911f6f0596  
  
>docker ps  
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES  
c8c54dcff4d2 nginx "/docker-entrypoint..." 13 minutes ago Up 7 seconds 0.0.0.0:8091->80/tcp nginx8091  
74911f6f0596 nginx "/docker-entrypoint..." 19 minutes ago Up 1 second 0.0.0.0:8090->80/tcp nginx8090  
>
```

*- suppression d'un conteneur docker :

```
>docker stop c8c54dcff4d2  
c8c54dcff4d2  
  
>docker stop 74911f6f0596  
74911f6f0596  
  
>docker rm c8c54dcff4d2  
c8c54dcff4d2  
  
>docker rm 74911f6f0596  
74911f6f0596  
  
>
```

4. Exécution d'une application SpringBoot dans un conteneur Docker.

- Créer une Application SpringBoot avec un seul Contrôleur qui affiche le message « Bonjour »

Il faut faire « mvn clean package » pour faire le « build » de l'application.

```

C:\Windows\System32\cmd.exe
Downloaded from central: https://repo.maven.apache.org/maven2/org/vafer/jdependency/2.4.0/jdependency-2.4.0.jar (180 kB at 7.7 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/errorprone/error_prone_annotations/2.3.4/error_prone_annotations-2.3.4.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/checkerframework/checker-compat-qual/2.5.5/checker-compat-qual-2.5.5.jar (5.9 kB at 248 B/s)
Downloading from central: https://repo.maven.apache.org/maven2/com/google/j2objc/j2objc-annotations/1.3/j2objc-annotations-1.3.jar
Downloaded from central: https://repo.maven.apache.org/maven2/net/java/dev/jna/jna/5.7.0/jna-5.7.0.jar (1.7 MB at 71 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/commons/commons-lang3/3.7/commons-lang3-3.7.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/jdom/jdom2/2.0.6/jdom2-2.0.6.jar (305 kB at 13 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/errorprone/error_prone_annotations/2.3.4/error_prone_annotations-2.3.4.jar (14 kB at 578 B/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/j2objc/j2objc-annotations/1.3/j2objc-annotations-1.3.jar (8.8 kB at 365 B/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/commons/commons-lang3/3.7/commons-lang3-3.7.jar (500 kB at 17 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/guava/guava/28.2-android/guava-28.2-android.jar (2.6 MB at 86 kB/s)
[INFO] Replacing main artifact with repackaged archive
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 01:39 min
[INFO] Finished at: 2023-10-14T23:17:39+01:00
[INFO] -----

```

- Ecrire le Dockerfile pour créer une image de cette application

Springboot with Docker > target			
Nom	Modifié le	Type	Taille
classes	14/10/2023 23:25	Dossier de fichiers	
generated-sources	14/10/2023 23:16	Dossier de fichiers	
generated-test-sources	14/10/2023 23:16	Dossier de fichiers	
maven-archiver	14/10/2023 23:16	Dossier de fichiers	
maven-status	14/10/2023 23:16	Dossier de fichiers	
surefire-reports	14/10/2023 23:16	Dossier de fichiers	
test-classes	14/10/2023 23:25	Dossier de fichiers	
app	14/10/2023 23:17	Archive WinRAR	17 077 Ko
app.jar.original	14/10/2023 23:16	Fichier ORIGINAL	4 Ko

```
[+] Building 0.7s (2/2) FINISHED docker:default
=> [internal] load build definition from Dockerfile 0.3s
=> => transferring dockerfile: 2B 0.1s
=> [internal] load .dockerignore 0.4s
=> => transferring context: 2B 0.0s
ERROR: failed to solve: failed to read dockerfile: open /var/lib/docker/tmp/buildkit-mount3139265740/Dockerfile: no such file or directory
cd '.\Springboot with Docker\' > docker build -t app .
2023/10/16 11:27:30 http2: server: error reading preface from client //./pipe/docker_engine: file has already been closed
[+] Building 6.6s (7/7) FINISHED docker:default
=> [internal] load build definition from Dockerfile 0.1s
=> => transferring dockerfile: 130B 0.0s
=> [internal] load .dockerignore 0.1s
=> => transferring context: 2B 0.0s
=> [internal] load metadata for docker.io/library/openjdk:11 5.9s
=> [internal] load build context 0.1s
=> => transferring context: 64B 0.0s
=> [1/2] FROM docker.io/library/openjdk:11@sha256:99bac5bf83633e3c7399aed725c8415e7b569b54e03e4599e580fc9cdb7c21ab 0.0s
=> CACHED [2/2] ADD target/app.jar app.jar 0.0s
=> exporting to image 0.1s
=> => exporting layers 0.0s
=> => writing image sha256:803cb4f31de28a1f41f3942866aab426245ffde29ef98d263784b3a5984e465d 0.0s
=> => naming to docker.io/library/app 0.0s
```

What's Next?

- Exécuter cette image dans un Conteneur

- Compréhension des couches d'image et de la réutilisabilité des images.

The screenshot shows the Visual Studio Code interface. The code editor displays `GreetingsController.java` with the following content:

```

1 package com.mycompany.applicationone.controller;
2
3 import org.springframework.web.bind.annotation.GetMapping;
4 import org.springframework.web.bind.annotation.RequestMapping;
5 import org.springframework.web.bind.annotation.RestController;
6
7 @RestController
8 @RequestMapping("/api/v1")
9 public class GreetingsController {
10
11     @GetMapping("/welcome")
12     public String welcome(){
13         return "Welcome to the world of Azure Containers!";
14     }
15 }

```

The terminal window shows log output from a Docker container:

```

2023-10-16 10:30:11.633 INFO 1 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[]      : Initializing Spring embedded WebApplicationC...
2023-10-16 10:30:11.633 INFO 1 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization c...
2023-10-16 10:30:12.351 INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with ...
context path ''
2023-10-16 10:30:12.374 INFO 1 --- [           main] c.m.a.ApplicationOneApplication        : Started ApplicationOneApplication in 3.578 s ...
econds (JVM running for 5.054)
2023-10-16 10:36:35.129 INFO 1 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[]      : Initializing Spring DispatcherServlet 'dispa...
tcherServlet'
2023-10-16 10:36:35.129 INFO 1 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet       : Initializing Servlet 'dispatcherServlet'
2023-10-16 10:36:35.131 INFO 1 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet       : Completed initialization in 1 ms

```

The file explorer on the left shows the project structure, including `Dockerfile`, `src`, and `target` directories.

The screenshot shows the Postman application interface. A GET request is made to `localhost:8080/api/v1/welcome`. The response body is:

```

1 Welcome to the world of Azure Containers!

```

The status bar at the bottom indicates `Status: 200 OK Time: 124 ms Size: 205 B`.

5. Compréhension de l'Application SpringBoot

- Installer MongoDB et Créer les Bases et Collections Nécessaires

The screenshot shows the VS Code interface with the following details:

- EXPLORER:** Shows the project structure with files like Dockerfile, GreetingsController.java, .mvn, mvnw, mvnw.cmd, notes.txt, and pom.xml.
- CODE FILES:** Shows the Dockerfile and GreetingsController.java files.
- TAB BAR:** Shows Dockerfile, GreetingsController.java, PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS, and POSTMAN CONSOLE.
- TERMINAL:** Shows the command `cd ..\Springboot and MongoDB with Docker, Docker Compose\'` followed by `> mongod --version`.
- STATUS BAR:** Shows Ln 16, Col 1, Spaces: 4, UTF-8, LF, Java.

- Tester l'Application SpringBoot Fournie et vérifier qu'elle accède à la Base Correctement en utilisant JUnit et Postman.

The terminal window shows the following command and its output:

```
C:\Windows\System32\cmd.exe - mvn spring-boot:run
```

```
mvn spring-boot:run
```

```
[INFO] [INFO] --- < com.mycompany:ob-item-service >-----
```

```
[INFO] Building ob-item-service 0.0.1-SNAPSHOT
```

```
[INFO] from pom.xml
```

```
[INFO] -----[ jar ]-----
```

```
[INFO] >>> spring-boot:2.4.9:run (default-cli) > test-compile @ ob-item-service >>>
```

```
[INFO] --- resources:3.2.0:resources (default-resources) @ ob-item-service ---
```

```
[INFO] Using 'UTF-8' encoding to copy filtered resources.
```

```
[INFO] Using 'UTF-8' encoding to copy filtered properties files.
```

```
[INFO] Copying 2 resources
```

```
[INFO] Copying 1 resource
```

```
[INFO] --- compiler:3.8.1:compile (default-compile) @ ob-item-service ---
```

```
[INFO] Changes detected - recompiling the module!
```

```
[INFO] Compiling 7 source files to [REDACTED]\target\classes
```

```
[INFO] --- resources:3.2.0:testResources (default-testResources) @ ob-item-service ---
```

```
[INFO] Using 'UTF-8' encoding to copy filtered resources.
```

```
[INFO] Using 'UTF-8' encoding to copy filtered properties files.
```

```
[INFO] skip non existing resourceDirectory [REDACTED]\src\test\resources
```

```
[INFO] --- compiler:3.8.1:testCompile (default-testCompile) @ ob-item-service ---
```

```
[INFO] Changes detected - recompiling the module!
```

```
[INFO] Compiling 1 source file [REDACTED]\target\test-classes
```

```
<<< spring-boot:2.4.9:run (default-cli) < test-compile @ ob-item-service <<<
```

```
[INFO]
```

```
[INFO] --- spring-boot:2.4.9:run (default-cli) @ ob-item-service ---
```

```
Downloading from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-buildpack-platform/2.4.9/spring-boot-buildpack-platform-2.4.9.pom
```

```
Downloaded from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-buildpack-platform/2.4.9/spring-boot-buildpack-platform-2.4.9.pom (3.1 kB at 540 B/s)
```

```
Downloading from central: https://repo.maven.apache.org/maven2/net/java/dev/jna/jna-platform/5.5.0/jna-platform-5.5.0.pom
```

```
Downloaded from central: https://repo.maven.apache.org/maven2/net/java/dev/jna/jna-platform/5.5.0/jna-platform-5.5.0.pom (1.8 kB at 16 kB/s)
```

```
Downloading from central: https://repo.maven.apache.org/maven2/net/java/dev/jna/jna/5.5.0/jna-5.5.0.pom
```

```
Downloaded from central: https://repo.maven.apache.org/maven2/net/java/dev/jna/jna-5.5.0.pom (1.6 kB at 30 kB/s)
```

```
Downloading from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-loader-tools/2.4.9/spring-boot-loader-tools-2.4.9.pom
```

```
Downloaded from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-loader-tools/2.4.9/spring-boot-loader-tools-2.4.9.pom (2.3 kB at 4.0 kB/s)
```

```
Downloading from central: https://repo.maven.apache.org/maven/shared/maven-common-artifact-filters/3.1.1/maven-common-artifact-filters-3.1.1.pom
```

```

C:\Windows\System32\cmd.exe - mvn spring-boot:run
[INFO] Attaching agents: []

[INFO] Spring Boot :: (v2.4.9)

2023-10-16 11:45:11.543 INFO 18240 --- [           main] c.m.o.ObItemServiceApplication        : Starting ObItemServiceApplication using Java 15.0.1 on DESKTOP-JOKLOT4 with PID 18240 (C:\Users\USER\Downloads\9781803236346_Code\Code Files\Springboot and MongoDB with Docker, Docker Compose\target\classes started by Lenovoo in C:\Users\USER\Downloads\9781803236346_Code\Code Files\Springboot and MongoDB with Docker, Docker Compose)
2023-10-16 11:45:11.546 INFO 18240 --- [           main] c.m.o.ObItemServiceApplication        : The following profiles are active: local
2023-10-16 11:45:12.513 INFO 18240 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data MongoDB repositories in DEFAULT mode.
2023-10-16 11:45:12.597 INFO 18240 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 75 ms. Found 2 MongoDB repository interfaces.

2023-10-16 11:45:14.066 INFO 18240 --- [           main] o.s.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8081 (http)
2023-10-16 11:45:14.125 INFO 18240 --- [           main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2023-10-16 11:45:14.127 INFO 18240 --- [           main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.50]
2023-10-16 11:45:14.499 INFO 18240 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2023-10-16 11:45:14.410 INFO 18240 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 2764 ms
2023-10-16 11:45:14.916 INFO 18240 --- [           main] org.mongodb.driver.cluster          : Cluster created with settings {hosts=[{host.docker.internal:28000}], mode=SINGLE, requiredClusterType=UNKNOWN, serverSelectionTimeout='30000 ms'}
2023-10-16 11:45:16.339 INFO 18240 --- [           main] o.s.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8081 (http) with context path ''
2023-10-16 11:45:16.360 INFO 18240 --- [           main] c.m.o.ObItemServiceApplication       : Started ObItemServiceApplication in 6.046 seconds (JVM running for 6.796)
2023-10-16 11:45:27.871 INFO 18240 --- [.internal:28000] org.mongodb.driver.cluster          : Exception in monitor thread while connecting to server host.docker.internal:28000

com.mongodb.MongoSocketOpenException: Exception opening socket
    at com.mongodb.internal.connection.SocketStream.open(SocketStream.java:70) ~[mongodb-driver-core-4.1.2.jar:na]
    at com.mongodb.internal.connection.InternalStreamConnection.open(InternalStreamConnection.java:143) ~[mongodb-driver-core-4.1.2.jar:na]
    at com.mongodb.internal.connection.DefaultServerMonitor$ServerMonitorRunnable.lookupServerDescription(DefaultServerMonitor.java:188) ~[mongodb-driver-core-4.1.2.jar:na]
    at com.mongodb.internal.connection.DefaultServerMonitor$ServerMonitorRunnable.run(DefaultServerMonitor.java:144) ~[mongodb-driver-core-4.1.2.jar:na]
    at java.base/java.lang.Thread.run(Thread.java:832) ~[na:na]
Caused by: java.net.SocketTimeoutException: Connect timed out
    at java.base/sun.nio.ch.NioSocketImpl.timedFinishConnect(NioSocketImpl.java:546) ~[na:na]
    at java.base/sun.nio.ch.NioSocketImpl.connect(NioSocketImpl.java:597) ~[na:na]
    at java.base/java.net.SocksSocketImpl.connect(SocksSocketImpl.java:333) ~[na:na]
    at java.base/java.net.Socket.connect(Socket.java:648) ~[na:na]
    at com.mongodb.internal.connection.SocketStreamHelper.initialize(SocketStreamHelper.java:78) ~[mongodb-driver-core-4.1.2.jar:na]
    at com.mongodb.internal.connection.SocketStream.initializeSocket(SocketStream.java:79) ~[mongodb-driver-core-4.1.2.jar:na]
    at com.mongodb.internal.connection.SocketStream.open(SocketStream.java:65) ~[mongodb-driver-core-4.1.2.jar:na]
    ... 4 common frames omitted

```

En utilisant JUnit :

Ceci est la classe JUnit :



```

Dockerfile | J GreetingsController.java | J ObItemServiceApplicationTests.java ×
Springboot and MongoDB with Docker, Docker Compose > src > test > java > com > mycompany > obitemservice > J ObItemServiceApplicationTests.java > ...
1 package com.mycompany.obitemservice;
2
3 import org.junit.jupiter.api.Test;
4 import org.springframework.boot.test.context.SpringBootTest;
5
6 @SpringBootTest
7 class ObItemServiceApplicationTests {
8
9     @Test
10    void contextLoads() {
11    }
12
13 }
14

```

```
[INFO] Scanning for projects...
[INFO]
[INFO] ----- < com.mycompany:ob-item-service > -----
[INFO] Building ob-item-service 0.0.1-SNAPSHOT
[INFO]   from pom.xml
[INFO]      [ jar ]
[INFO]
[INFO] --- resources:3.2.0:resources (default-resources) @ ob-item-service ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Using 'UTF-8' encoding to copy filtered properties files.
[INFO] Copying 2 resources
[INFO] Copying 1 resource
[INFO]
[INFO] --- compiler:3.8.1:compile (default-compile) @ ob-item-service ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- resources:3.2.0:testResources (default-testResources) @ ob-item-service ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Using 'UTF-8' encoding to copy filtered properties files.
[INFO] skip non existing resourceDirectory [REDACTED]\src\test\resources
[INFO]
[INFO] --- compiler:3.8.1:testCompile (default-testCompile) @ ob-item-service ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- surefire:2.22.2:test (default-test) @ ob-item-service ---
[INFO]
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running com.mycompany.obitemservice.ObItemServiceApplicationTests
11:56:22.507 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantiating CacheAwareContextLoaderDelegate from class [org.springframework.test.context.cache.DefaultCacheAwareContextLoaderDelegate]
11:56:22.524 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantiating BootstrapContext using constructor [public org.springframework.test.context.support.DefaultBootstrapContext(java.lang.Class,org.springframework.test.context.CacheAwareContextLoaderDelegate)]
11:56:22.570 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantiating TestContextBootstrapper for test class [com.mycompany.obitemservice.ObItemServiceApplicationTests] from class [org.springframework.boot.test.context.SpringBootTestTestContextBootstrapper]
11:56:22.587 [main] INFO org.springframework.boot.test.context.SpringBootTestTestContextBootstrapper - Neither @ContextConfiguration nor @ContextHierarchy found for test class [com.mycompany.obitemservice.ObItemServiceApplicationTests], using SpringBootTestContextLoader
11:56:22.587 [main] DEBUG org.springframework.test.context.support.AbstractContextLoader - Did not detect default resource location for test class [com.mycompany.obitemservice.ObItemServiceApplicationTests]; class path resource [com/mycompany/obitemservice/ObItemServiceApplicationTests-context.xml] does not exist
```

```
Selection C:\Windows\System32\cmd.exe - mvn test
[...]
11:56:22.820 [main] INFO org.springframework.boot.test.context.SpringBootTestBootstrapper - Using TestExecutionListeners: [org.springframework.test.context.web.ServletTestExecutionListener@768f02c2, org.springframework.test.context.support.DirtiesContextBeforeModeTestExecutionListener@545d435e, org.springframework.test.context.event.ApplicationEventsTestExecutionListener@20c0a64d, org.springframework.boot.test.mock.mockito.MockitoTestExecutionListener@455b6df1, org.springframework.boot.test.autoconfigure.SpringBootDependencyInjectionTestExecutionListener@4dbbd8f8, org.springframework.test.context.support.DirtiesContextTestExecutionListener@3f67593e, org.springframework.test.context.transaction.TransactionalTestExecutionListener@ab06251, org.springframework.test.context.jdbc.SqlScriptsTestExecutionListener@41ab013, org.springframework.test.context.event.EventPublishingTestExecutionListener@14bee915, org.springframework.boot.test.context.SpringBootTestBootstrapper@1115ec15, org.springframework.boot.test.autoconfigure.restdocs.RestDocsTestExecutionListener@2ea68c, org.springframework.boot.test.autoconfigure.web.client.MockRestServiceServerResetTestExecutionListener@59e505b2, org.springframework.boot.test.autoconfigure.web.servlet.MockMvcPrintOnlyOnFailureTestExecutionListener@3af0a9da, org.springframework.boot.test.autoconfigure.web.servlet.WebDriverTestExecutionListener@43b9fd5, org.springframework.boot.test.autoconfigure.web.services.client.MockWebServiceServerTestExecutionListener@79dc5318]
11:56:22.820 [main] DEBUG org.springframework.test.context.support.AbstractDirtiesContextTestExecutionListener - Before test class: context [DefaultTestContext@1add175 testClass = ObItemServiceApplicationTests, testInstance = [null], testMethod = [null], testException = [null], mergedContextConfiguration = [WebMergedContextConfiguration@aef3540e testClass = ObItemServiceApplicationTests, locations = '{}', classes = {[class com.mycompany.obitemservice.ObItemServiceApplication]}, contextInitializerClasses = '{}', activeProfiles = '{}', propertySourceLocations = '{}', propertySourceProperties = '{org.springframework.boot.test.context.SpringBootTestBootstrapper=true}', contextCustomizers = set[org.springframework.boot.test.context.filter.ExcludeDeflectedContextCustomizer@7ed5b4, org.springframework.boot.test.json.DuplicateJsonObjectContextCustomizerFactory@691dd28], org.springframework.boot.test.mock.mockito.MockitoContextCustomizer@0, org.springframework.boot.test.web.client.TestRestTemplateContextCustomizer@2accdb5, org.springframework.boot.test.autoconfigure.metrics.MetricsExportContextCustomizerFactory@DisableMetricExportContextCustomizer@a2431d0, org.springframework.boot.test.autoconfigure.properties.PropertyMappingContextCustomizer@0, org.springframework.boot.test.context.SpringBootTestTestWebEnvironment@4449d195], resourceBasePath = 'src/main/webapp', contextLoader = 'org.springframework.boot.test.context.SpringBootTestContextLoader', parent = [null]], attributes = {[map@org.springframework.boot.test.context.web.ServletTestExecutionListener.activateListener': true]}], class annotated with @DirtiesContext [false] with mode [null].
11:56:22.853 [main] DEBUG org.springframework.test.context.support.TestPropertySourceUtils - Adding inlined properties to environment: {spring.jmx.enabled=false, org.springframework.boot.test.context.SpringBootTestBootstrapper=true}

```
  ```

  :: Spring Boot ::          (v2.4.9)

2023-10-16 11:56:23.154  INFO 18600 --- [           main] c.m.o.ObItemServiceApplicationTests      : Starting ObItemServiceApplicationTests using Java 15.0.1 on DESKTOP-JOKLOT4 with PID 18600 (started by Lenovoo in C:\Users\USER\Downloads\9781803236346_Code\Code Files\Springboot and MongoDB with Docker, Docker Compose)
2023-10-16 11:56:23.169  INFO 18600 --- [           main] c.m.o.ObItemServiceApplicationTests      : The following profiles are active: local
2023-10-16 11:56:23.834  INFO 18600 --- [           main] s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data MongoDB repositories in DEFAULT mode.
2023-10-16 11:56:23.896  INFO 18600 --- [           main] s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 51 ms. Found 2 MongoDB repository interfaces

2023-10-16 11:56:24.736  INFO 18600 --- [           main] org.mongodb.driver.cluster            : Cluster created with settings {hosts=[host.docker.internal:28000], mode=SINGLE, requiredClusterType=UNKNOWN, serverSelectionTimeout='30000 ms'}
```

7. Docker Compose

Running the docker compose file :

J ObItemServiceApplicationTests.java X J ItemCategoryController.java X docker-compose.yml X

```
springboot and MongoDB with Docker, Docker Compose > src > main > resources > docker-compose.yml > {} services > {} springboot-with-mongo
docker-compose.yml - The Compose specification establishes a standard for the definition of multi-container platform-agnostic applications (compos
1 version: "3"
2 services:
3   mongodb-container-one:
4     image: mongo:latest
5     container_name: "mongodb-container-one"
6     ports:
7       - 28000:27017
8   springboot-with-mongodb-container-one:
9     image: ranjan715/springboot-with-mongodb:v11
10    container_name: springboot-with-mongodb-container-one
11    ports:
12      - 8090:8081
13    links:
14      - mongodb-container-one
```

C:\Windows\System32\cmd.exe - docker-compose up -d

```
[+] Running 4/19
- springboot-with-mongodb-container-one 8 layers [=====] 97.11MB/312.6MB Pulling
- 955615a668ce Extracting [=====] 6.128MB/54.93MB
  2756ef5f69a5 Download complete
  911ea9f2bd51 Download complete
  27b0a22ee906 Downloading [=====] 49.49MB/54.57MB
  785dfbf3c6c Download complete
  3fccb14f0369 Download complete
  4a0c30fed9c Downloading [=====] 41.49MB/203.1MB
  d73d22fb6202 Waiting
- mongodh-container-one 9 layers [=====] 0B/0B Pulling
  43f89b94cd7d Waiting
  54a7480baa9d Waiting
  7f9301fb7df Waiting
  5e4470f2e90f Waiting
  40d046ff8fd3 Waiting
  e062d62b861e Waiting
  72919e34fd8 Waiting
  ab22810dfc64 Waiting
  fb05c29fbdf5 Waiting
```

The container has already started :

C:\Windows\System32\cmd.exe

```
[+] Running 19/19
[+] Running 19/19
- springboot-with-mongodb-container-one 8 layers [=====] 0B/0B Pulled
  955615a668ce Pull complete
  2756ef5f69a5 Pull complete
  911ea9f2bd51 Pull complete
  27b0a22ee906 Pull complete
  785dfbf3c6c Pull complete
  3fccb14f0369 Pull complete
  4a0c30fed9c Pull complete
  d73d22fb6202 Pull complete
- mongodh-container-one 9 layers [=====] 0B/0B Pulled
  43f89b94cd7d Pull complete
  54a7480baa9d Pull complete
  7f9301fb7df Pull complete
  5e4470f2e90f Pull complete
  40d046ff8fd3 Pull complete
  e062d62b861e Pull complete
  72919e34fd8 Pull complete
  ab22810dfc64 Pull complete
  fb05c29fbdf5 Pull complete
[+] Running 3/3
[+] Network resources_default Created
[+] Container mongodb-container-one Started
[+] Container springboot-with-mongodb-container-one Started
```

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to [nginx.org](#).
Commercial support is available at [nginx.com](#).

Thank you for using nginx.

⇒ The application works

```
es>docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
574eaf631307 ranjan715/springboot-with-mongodb:v11 "java -jar app.jar" 3 minutes ago Up 3 minutes 0.0.0.0:8090->8081/tcp
c78f4b127ed9 mongo:latest "docker-entrypoint.s..." 3 minutes ago Up 3 minutes 0.0.0.0:28000->27017/tcp
014d043b30bf app "java -jar app.jar" 43 minutes ago Up 43 minutes 0.0.0.0:8080->8080/tcp
es>
```

Connected on localhost :28000 with mongodb

MongoDB Compass - localhost:28000/admin

Connect Edit View Help

localhost:28000 ...

Collections

+ Create collection Refresh

My Queries

Databases

Search

- admin
- config
- local

⇒ It communicates very well

- Explorer le contenu des deux containers pour vérifier l'absence d'exceptions

```
c:\Windows\System32\cmd.exe - docker exec -it mongodb-container-one bash
>docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
4623a82a86b7 ranjan715/springboot-with-mongodb:v11 "java -jar app.jar" 21 minutes ago Up 2 minutes 0.0.0.0:8090->8081/tcp
24bd1be02126 mongo:latest "docker-entrypoint.s..." 21 minutes ago Up 21 minutes 0.0.0.0:28000->27017/tcp
>docker exec -it mongodb-container-one bash
root@24bd1be02126:/# ls -l
total 164
drwxrwxrwx 1 root root 7 Oct 4 02:08 bin -> usr/bin
drwxr-xr-x 2 root root 4096 Apr 18 2022 boot
drwxr-xr-x 4 root root 4096 Oct 13 04:46 data
drwxr-xr-x 5 root root 340 Oct 16 12:02 dev
drwxr-xr-x 2 root root 4096 Oct 13 04:46 docker-entrypoint-initdb.d
drwxr-xr-x 1 root root 4096 Oct 16 12:01 etc
drwxr-xr-x 2 root root 4096 Apr 18 2022 home
-rw-r--r-- 1 root root 108975 Oct 13 04:46 js-yaml.js
drwxrwxrwx 1 root root 7 Oct 4 02:08 lib -> usr/lib
drwxrwxrwx 1 root root 9 Oct 4 02:08 lib32 -> usr/lib32
drwxrwxrwx 1 root root 9 Oct 4 02:08 lib64 -> usr/lib64
drwxrwxrwx 1 root root 10 Oct 4 02:08 libx32 -> usr/libx32
drwxr-xr-x 2 root root 4096 Oct 4 02:08 media
drwxr-xr-x 2 root root 4096 Oct 4 02:08 mnt
drwxr-xr-x 2 root root 4096 Oct 4 02:08 opt
dr-xr-xr-x 256 root root 0 Oct 16 12:02 proc
drwx----- 1 root root 4096 Oct 13 04:46 root
drwxr-xr-x 5 root root 4096 Oct 4 02:12 run
drwxrwxrwx 1 root root 8 Oct 4 02:08 sbin -> usr/sbin
drwxr-xr-x 2 root root 4096 Oct 4 02:08 srv
dr-xr-xr-x 11 root root 0 Oct 16 12:02 sys
drwxrwxrwt 1 root root 4096 Oct 16 12:02 tmp
drwxr-xr-x 1 root root 4096 Oct 4 02:08 usr
drwxr-xr-x 1 root root 4096 Oct 4 02:12 var
root@24bd1be02126:/#
```

```
C:\Users\USER\Downloads\9781803236346_Code\Code Files\Springboot and MongoDB with Docker, Docker Compose\src\main\resources>docker exec -it springboot-with-mongodb-container-one bash
root@4623a82a86b7:/# ls -l
total 22344
-rwxr-xr-x 1 root root 22809867 Sep 24 2021 app.jar
drwxr-xr-x 1 root root 4096 Sep 3 2021 bin
drwxr-xr-x 2 root root 4096 Apr 10 2021 boot
drwxr-xr-x 5 root root 340 Oct 16 12:21 dev
drwxr-xr-x 1 root root 4096 Oct 16 12:01 etc
drwxr-xr-x 2 root root 4096 Apr 10 2021 home
drwxr-xr-x 1 root root 4096 Sep 2 2021 lib
drwxr-xr-x 2 root root 4096 Sep 2 2021 lib64
drwxr-xr-x 2 root root 4096 Sep 2 2021 media
drwxr-xr-x 2 root root 4096 Sep 2 2021 mnt
drwxr-xr-x 2 root root 4096 Sep 2 2021 opt
dr-xr-xr-x 257 root root 0 Oct 16 12:21 proc
drwxr----- 1 root root 4096 Sep 3 2021 root
drwxr-xr-x 3 root root 4096 Sep 2 2021 run
drwxr-xr-x 1 root root 4096 Sep 3 2021 sbin
drwxr-xr-x 2 root root 4096 Sep 2 2021 srv
dr-xr-xr-x 11 root root 0 Oct 16 12:21 sys
drwxrwxrwt 1 root root 4096 Oct 16 12:21 tmp
drwxr-xr-x 1 root root 4096 Sep 2 2021 usr
drwxr-xr-x 1 root root 4096 Sep 2 2021 var
root@4623a82a86b7:/#
```

Faites communiquer les deux containers et vérifier le bon fonctionnement de l'application avec Postman

The screenshot shows the Postman application interface. At the top, there is a header bar with tabs for 'GET localhost:8090/api/v1' and 'POST localhost:8090/api/v1'. Below the header, the main interface shows a request configuration for a 'GET' request to 'localhost:8090/api/v1/items'. The 'Params' tab is selected, showing a table for 'Query Params' with one row: 'Key' (Value: 'Value') and 'Description' (Value: 'Description'). Below the table, there are tabs for 'Body', 'Cookies', 'Headers (5)', and 'Test Results'. The 'Body' tab is selected, displaying a JSON response. The response is shown in 'Pretty' format, with line numbers 1 through 9. The JSON content is:

```

1 [ {
2   "id": "abc",
3   "name": "test",
4   "description": "test description",
5   "price": 99.99,
6   "categoryId": "abcde"
7 }
8 ]
9 ]

```

At the bottom of the interface, there are various navigation and utility buttons: Postbot, Runner, Start Proxy, Cookies, Trash, and Help.

The screenshot shows the Postman application interface. At the top, there are two tabs: 'GET localhost:8090/api/v1/i' and 'POST localhost:8090/api/v1/i'. Below these is a search bar with 'localhost:8090/api/v1/items'. On the right, it says 'No Environment' with a dropdown arrow. The main area shows a 'POST' request to 'localhost:8090/api/v1/items'. The 'Body' tab is selected, showing the following JSON payload:

```
1 {
2     "id": "abc",
3     "name": "test",
4     "description": "test description",
5     "price": 99.99,
6     "categoryId": "abcde"
7 }
```

Below the body, the 'Body' tab is selected, followed by 'Cookies', 'Headers (8)', and 'Test Results'. The status bar at the bottom indicates 'Status: 201 Created'.

- Quelle est l'utilité de DockerCompose ?

L'utilité de DockerCompose est qu'il est un fichier de configuration YAML utilisé pour définir les services d'une application avec une architecture organisée de façon qu'elle peut être construite en utilisant une seule commande pour la création et le lancement des services.

8. Intégration de Docker dans un environnement DevOps

- Etudier et Expliquer rôle de Docker dans un flux de développement et de déploiement continu.

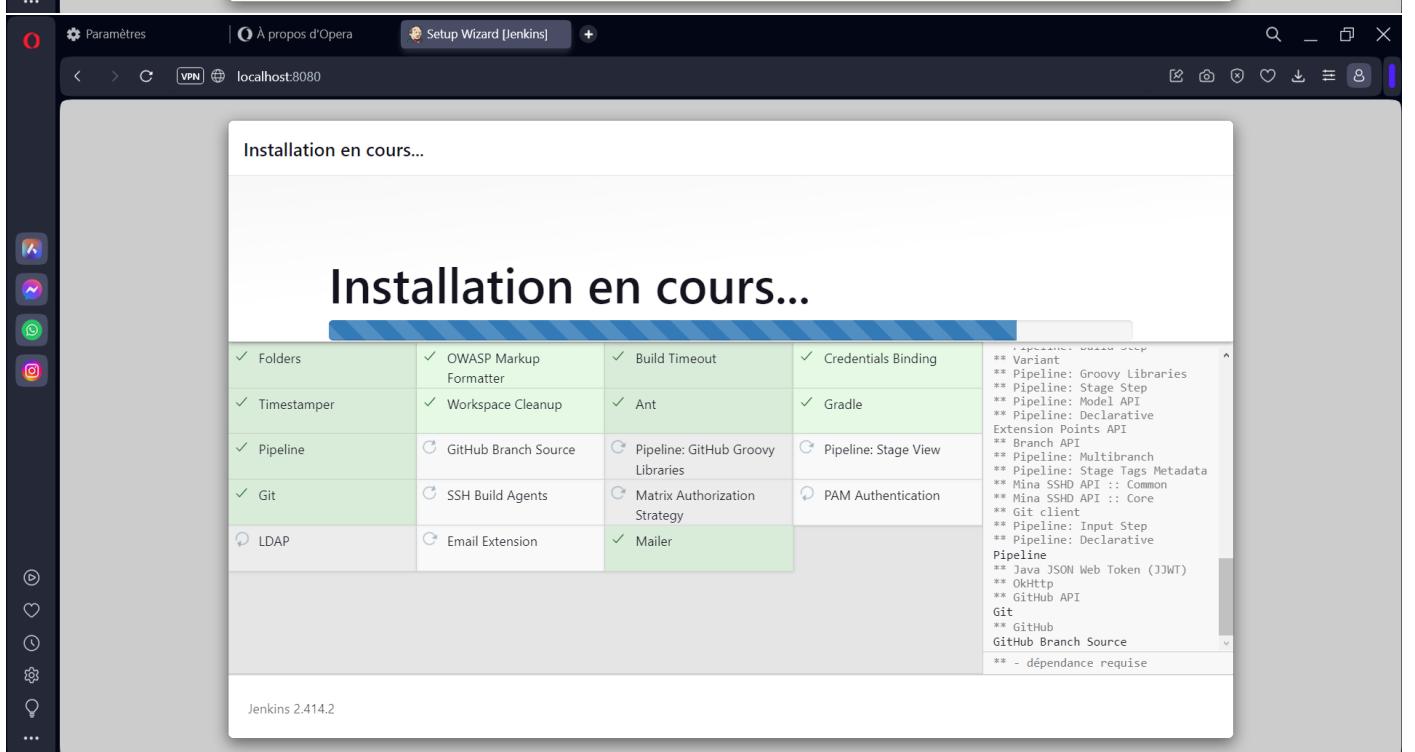
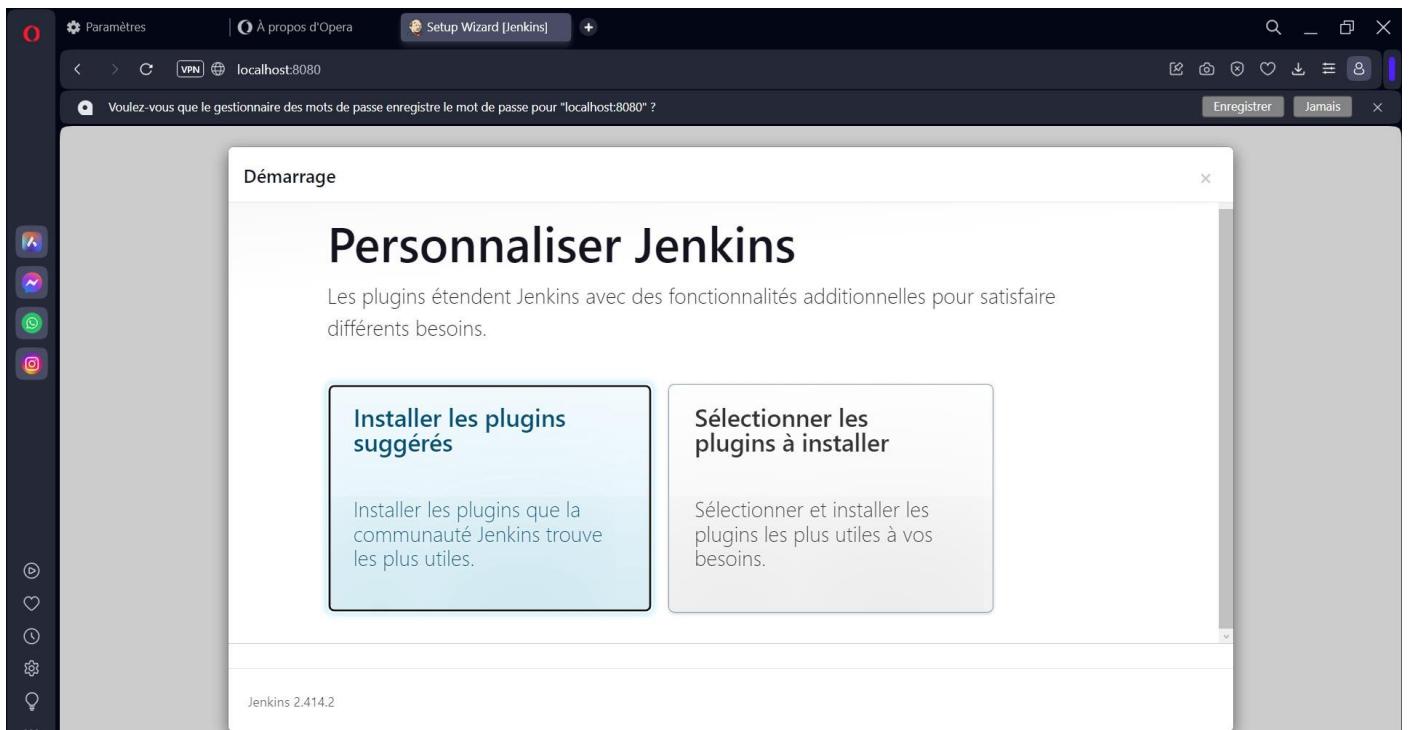
Docker permet de « partager » des applications tout développées sous formes d'images qui peuvent être construites en des containers, en exécutant ces containers, on est sur que l'environnement nécessaire pour que l'application fonctionne correctement existe.

9. Création de premiers Pipeline avec Jenkins

- Installation de Jenkins

```
[+] Building 122.3s (4/9)
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 505B
=> [internal] load metadata for docker.io/jenkins/jenkins:lts
=> [1/6] FROM docker.io/jenkins/jenkins:lts@sha256:b705323eaef70a7da4c1eed8b816f33dff2d5c8c3671170a2c17cf77aa4f15432
=> => resolve docker.io/jenkins/jenkins:lts@sha256:b705323eaef70a7da4c1eed8b816f33dff2d5c8c3671170a2c17cf77aa4f15432
=> => sha256:2a15645303916751e9f781fd403051190d94052d518b6e61b2398ab1f791c 2.77kB / 2.77kB
=> => sha256:d30d48f61941fc2b50307efdf932d060ee6335445f6ff535ac3d1d49e385a7e_12.62kB / 12.62kB
=> => sha256:b705323eaef70a7da4c1eed8b816f33dff2d5c8c3671170a2c17cf77aa4f15432 3.12kB / 3.12kB
=> => sha256:012c0b3e998c1a0c0bedf712eaaffa188580529dd026a04aa1c13fdb39e42b 49.56MB / 49.56MB
=> => sha256:f6154b0007aa47dceee550lebb63ddbf717761a97233c5402c4390d4a873bf 61.32MB / 61.32MB
=> => sha256:343d47f61941fc2b50307efdf932d060ee6335445f6ff535ac3d1d49e385a7e_12.62kB / 12.62kB
=> => sha256:012c0b3e998c1a0c0bedf712eaaffa188580529dd026a04aa1c13fdb39e42b 8.68MB / 8.68MB
=> => sha256:012c0b3e998c1a0c0bedf712eaaffa188580529dd026a04aa1c13fdb39e42b 19.3s
=> => sha256:655ef4e27471c21280d11bc0d0887a3165dzeb9345a44ed75063bed17ae7 198B
=> => sha256:10cad8b7df876c2af6402c97060507d723bdg838daefc82fadab362864f768 183B
=> => sha256:2471c729a3c7a1f2e6a6c1afad134f08fa5c684f7e10816e114d286ce7dc38 89.33MB / 89.33MB
=> => sha256:655ef4e27471c21280d11bc0d0887a3165dzeb9345a44ed75063bed17ae7 189B
=> => extracting sha256:012c0b3e998c1a0c0bedf712eaaffa188580529dd026a04aa1c13fdb39e42b
=> => sha256:3145851d167db7b8dd017f6734a26340476bdf5686808d72b4833947 6.09MB / 6.09MB
=> => sha256:614d8a8c6a95f7a9ab30388c1f16a55f34eb54e0bbc52d4c3e05033ac462000c8 77.15MB / 77.15MB
=> => sha256:995f5dc02b01604b68a93b3009a44dc0f302d5b6e66c478d7c0fb48fb6bf62 1.93KB / 1.93KB
=> => extracting sha256:f6154b0007aa47dceee550lebb63ddbf717761a97233c5402c4390d4a873bf
=> => sha256:c78f869d080beb29a5df2c497c699a86746c448fbba6f29bf4d231ec2a49 1.16KB / 1.16KB
=> => sha256:3a7b865a26287982180264f604f4b596884aa270eds5a0ee3b30711cf4d4c 392B / 392B
=> => sha256:d754f93a72cfe59a3b91340ff01f604be6b30ca1094f9024926d915957cd7fc 265B / 265B
=> => extracting sha256:50e0093fb8d5f36gb93870a544e50bf2f136g88e72f91ada02b9abdc672472
=> => extracting sha256:3d43735955786e6078b0887b6df62afdf7739d047ba49c37c18a5d54b99203c
=> => extracting sha256:10cad8b7df876c2af6402c97060507d723bdg838daefc82fadab362864f768
=> => extracting sha256:2471c729a3c7a1f2e6a6c1afad134f08fa5c684f7e10816e114d286ce7dc38
=> => extracting sha256:247231f280d11bc0d0887a3165d2eb9345a44ed75063bed17ae7
=> => extracting sha256:3145851d167db7b8dd017f6734a26340476b91f6db58668088d72b4833847
=> => extracting sha256:614d8a8c6a95f7a9ab30388c1f16a55f34eb54e0bbc52d4c3e05033ac462000c8
=> => extracting sha256:995f5dc02b01604b68a93b3009a44dc0f302d5b6e66c478d7c0fb48fb6bf62
=> => extracting sha256:c8f869d080beb29a5df2c497c699a86746c448fbba6f29bf4d231ec2a49
=> => extracting sha256:2fe85a26287982180264f604f4b596884aa270eds5a0ee3b30711cf4d4c
=> => extracting sha256:d754f693a72cfe59a3b9340ff01f604be6b30ca1094f9024926d915957cd7fc
[2/6] RUN apt-get update -qq &> apt-get install -qq apt-transport-https ca-certificates curl gnupg2 software-properties-common
=> => # Unpacking liblxml2:amd64 (2.9.4+dfsg-1.3~deb12u1) ...
=> => # Selecting previously unselected package shared-mime-info.
=> => # Preparing to unpack .../08-shared-mime-info_2.2-1_amd64.deb ...
=> => # Unpacking shared-mime-info (2.2-1) ...
=> => # Preparing to unpack .../09-libcurl3-gnutls_7.88.1-10+deb12u4_amd64.deb ...
=> => # Unpacking libcurl3-gnutls:amd64 (7.88.1-10+deb12u4) over (7.88.1-10+deb12u1) ...
77.5s
```

```
** Sélection C:\Windows\System32\cmd.exe - docker run -it -p 8080:8080 -p 50000:50000 -v /var/run/docker.sock:/var/run/docker.sock -v jenkins_home:/var/jenkins_home jenkins
2023-10-16 12:54:20.402+00000 [id=1] INFO org.eclipse.jetty.server.Server#doStart: jetty-10.0.15; built: 2023-04-11T17:25:14.480Z; git: 68017dbd0236bb7e187330d7585a059610f661d; jvm 1
2023-10-16 12:54:20.402+00000 [id=1] INFO o.e.j.w.StandardDescriptorProcessor#visitServlet: NO JSP Support for /, did not find org.eclipse.jetty.jsp.JettyJspServlet
2023-10-16 12:54:20.735+00000 [id=1] INFO o.e.j.s.s.DefaultSessionIdManager#doStart: Session workerName=node0
2023-10-16 12:54:21.222+00000 [id=1] INFO hudson.WebAppMain#contextInitialized: Jenkins home directory: /var/jenkins_home found at: EnvVars.masterEnvVars.get("JENKINS_HOME")
2023-10-16 12:54:21.380+00000 [id=1] INFO o.e.j.s.handler.ContextHandler#doStart: Started w@28cb912@{Jenkins v2.414.2,/,file:///var/jenkins_home/war/,AVAILABLE}{/var/jenkins_home/war/}
}
2023-10-16 12:54:21.419+00000 [id=1] INFO o.e.j.server.AbstractConnector#doStart: Started ServerConnector@37883b97[HTTP/1.1, (http/1.1){0.0.0.0:8080}]
2023-10-16 12:54:21.439+00000 [id=1] INFO org.eclipse.jetty.server.Server#doStart: Started Server@53aa5d5[STARTING][10.0.15,sto=0] @1888ms
2023-10-16 12:54:21.443+00000 [id=25] INFO winstонe.Logger#logInternal: Winstone Servlet Engine running: controlPort=disabled
2023-10-16 12:54:21.662+00000 [id=32] INFO jenkins.InitReactorRunner$1@onAttained: Started initialization
2023-10-16 12:54:21.720+00000 [id=40] INFO jenkins.InitReactorRunner$1@onAttained: Listed all plugins
2023-10-16 12:54:22.462+00000 [id=40] INFO jenkins.InitReactorRunner$1@onAttained: Prepared all plugins
2023-10-16 12:54:22.470+00000 [id=39] INFO jenkins.InitReactorRunner$1@onAttained: Started all plugins
2023-10-16 12:54:22.477+00000 [id=30] INFO jenkins.InitReactorRunner$1@onAttained: Augmented all extensions
2023-10-16 12:54:22.738+00000 [id=30] INFO jenkins.InitReactorRunner$1@onAttained: System config loaded
2023-10-16 12:54:22.739+00000 [id=33] INFO jenkins.InitReactorRunner$1@onAttained: System config adapted
2023-10-16 12:54:22.740+00000 [id=33] INFO jenkins.InitReactorRunner$1@onAttained: Loaded all jobs
2023-10-16 12:54:22.742+00000 [id=33] INFO jenkins.InitReactorRunner$1@onAttained: Configuration for all jobs updated
2023-10-16 12:54:22.808+00000 [id=33] INFO jenkins.install.SetupWizard#init:
*****
***** Jenkins initial setup is required. An admin user has been created and a password generated.
***** Please use the following password to proceed to installation:
*****
This may also be found at: /var/jenkins_home/secrets/initialAdminPassword
*****
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.codehaus.groovy.vmplugin.v7.Java7$1 (file:/var/jenkins_home/war/WEB-INF/lib/groovy-all-2.4.21.jar) to constructor java.lang.invoke.MethodHandles$Lookup.(java.lang.Class,int)
WARNING: Please consider reporting this to the maintainers of org.codehaus.groovy.vmplugin.v7.Java7$1
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
2023-10-16 12:55:00.615+00000 [id=30] INFO jenkins.InitReactorRunner$1@onAttained: Completed initialization
2023-10-16 12:55:00.815+00000 [id=24] INFO hudson.lifecycle.Lifecycle#onReady: Jenkins is fully up and running
```



Démarrage

Créer le 1er utilisateur Administrateur

Nom d'utilisateur

Mot de passe

Confirmation du mot de passe

Nom complet

[Continuer en tant qu'Administrateur](#)[Sauver et continuer](#)

Démarrage

Mot de passe

Confirmation du mot de passe

Nom complet

Adresse courriel

[Continuer en tant qu'Administrateur](#)[Sauver et continuer](#)

On va travailler avec une simple web app développée en python Flask, elle permet de faire un simple login avec des contraintes sur le mot de passe défini et de visualiser les logs des essais de logins faits (les logs sont sous forme de texte ou graphiquement en utilisant une bibliothèque de javascript chartjs) :

Login Page

Username:

Password:

Password must meet the following requirements:

- At least 8 characters long
- Contain at least one uppercase letter
- Contain at least one lowercase letter
- Contain at least one digit
- Contain at least one special character

[View Login Attempt Logs](#)

Login Page

Username:

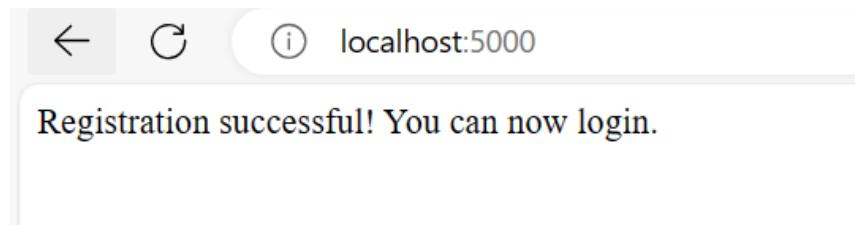
Password:

Password must meet the following requirements:

- At least 8 characters long
- Contain at least one uppercase letter
- Contain at least one lowercase letter
- Contain at least one digit
- Contain at least one special character

[View Login Attempt Logs](#)

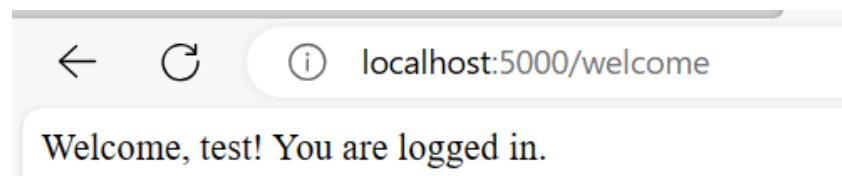
Exemple de « register » :



Exemple de « login » faux :



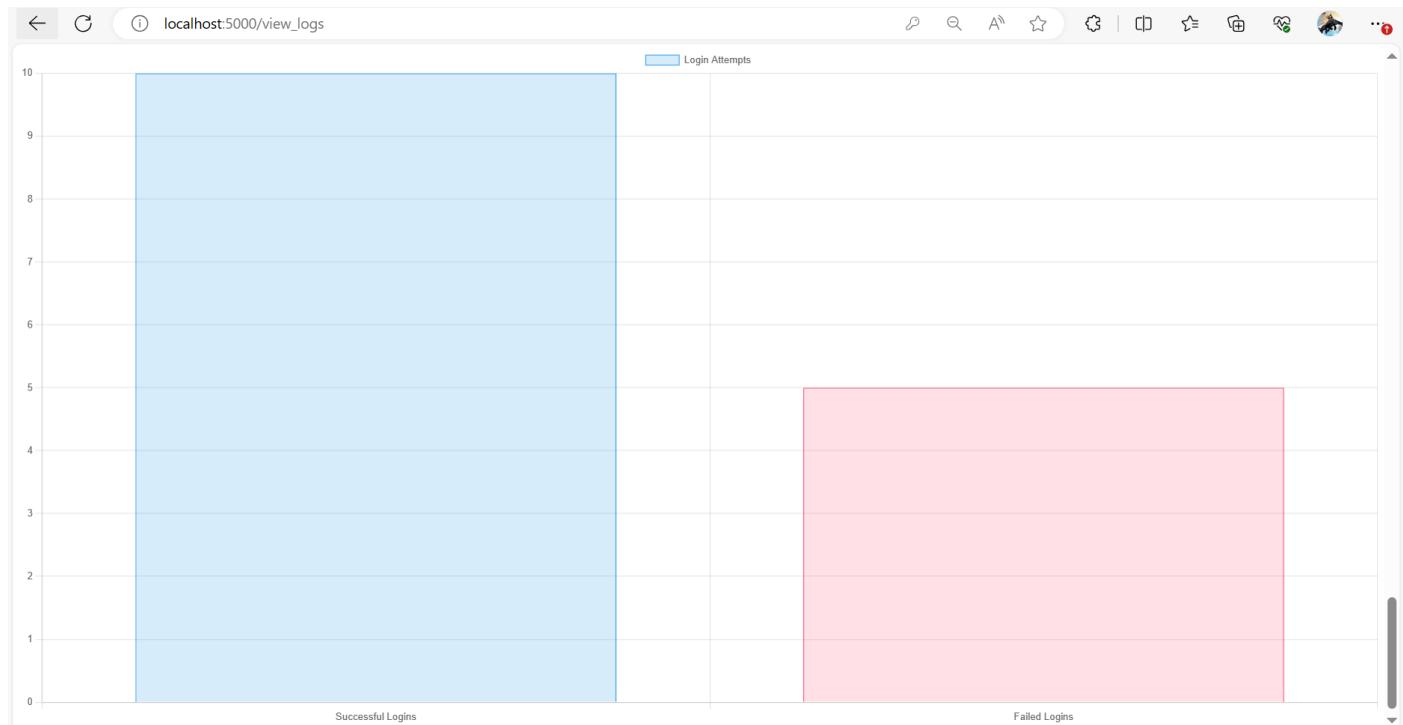
Exemple de « login » correcte :



Visualisation des logs d'essais de login (login attempts) en texte :

Date and Time	Username
WARNING:werkzeug: * Debugger is active!	
INFO:werkzeug: * Debugger PIN: 118-462-061	
INFO:werkzeug: * Detected change in 'F:\Users\Lenovo\Desktop\TPs_RT5\TP_devops\FirstPipeline\app.py', reloading	
WARNING:werkzeug: * Debugger is active!	
INFO:werkzeug: * Debugger PIN: 118-462-061	
INFO:werkzeug:127.0.0.1 - - [16/Oct/2023 17:48:44] "GET / HTTP/1.1" 200 -	
INFO:root:Login attempt for username: abc	
INFO:werkzeug:127.0.0.1 - - [16/Oct/2023 17:48:48] "POST / HTTP/1.1" 200 -	
INFO:werkzeug:127.0.0.1 - - [16/Oct/2023 17:48:50] "GET / HTTP/1.1" 200 -	
INFO:root:Login attempt for username: abc	
INFO:werkzeug:127.0.0.1 - - [16/Oct/2023 17:48:52] "POST / HTTP/1.1" 200 -	
INFO:werkzeug:127.0.0.1 - - [16/Oct/2023 17:48:54] "GET / HTTP/1.1" 200 -	

Visualisation des logs d'essais de login (login attempts) avec chartjs :



```

MINGW64:/c/Users/USER/Downloads/9781803236346_Code/Code Files/Springboot with Docker
Lenovo@DESKTOP-JOKLOT4 MINGW64 ~
$ cd "Springboot with Docker"
Lenovo@DESKTOP-JOKLOT4 MINGW64 ~/Springboot with Docker
$ git init
Initialized empty Git repository in C:\Users\USER\Downloads\9781803236346_Code\Code Files\Springboot with Docker/.git/
Lenovo@DESKTOP-JOKLOT4 MINGW64 ~/Springboot with Docker (master)
$ git add .
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of '.mvn/wrapper/MavenWrapperDownloader.java', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of '.mvn/wrapper/maven-wrapper.properties', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Dockerfile', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'mvnw', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'mvnw.cmd', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'notes.txt', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'pom.xml', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/main/java/com/mycompany/applicationone/ApplicationOneApplication.java', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/main/java/com/mycompany/applicationone/controller/GreetingsController.java', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/main/resources/application.properties', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/test/java/com/mycompany/applicationone/ApplicationOneApplicationTests.java', LF will be replaced by CRLF the next time Git touches it
Lenovo@DESKTOP-JOKLOT4 MINGW64 ~/Springboot with Docker (master)
$ git commit -m "Springboot with docker"
[!master (root-commit) 53e289] Springboot with docker
 13 files changed, 754 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 .mvn/wrapper/MavenWrapperDownloader.java
 create mode 100644 .mvn/wrapper/maven-wrapper.jar
 create mode 100644 .mvn/wrapper/maven-wrapper.properties
 create mode 100644 Dockerfile

```

MINGW64:/c/Users/USER/Downloads/9781803236346_Code/Code Files/Springboot with Docker

```

will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/test/java/com/mycompany/applicationone/ApplicationOneApplicationTests.java', LF will be replaced by CRLF the next time Git touches it
Lenovo@DESKTOP-JOKLOT4 MINGW64 ~
boot with Docker (master)
$ git commit -m "Springboot with docker"
[!master (root-commit) 53e289] Springboot with docker
 13 files changed, 754 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 .mvn/wrapper/MavenWrapperDownloader.java
 create mode 100644 .mvn/wrapper/maven-wrapper.jar
 create mode 100644 .mvn/wrapper/maven-wrapper.properties
 create mode 100644 Dockerfile
 create mode 100644 mvnw
 create mode 100644 mvnw.cmd
 create mode 100644 notes.txt
 create mode 100644 pom.xml
 create mode 100644 src/main/java/com/mycompany/applicationone/ApplicationOneApplication.java
 create mode 100644 src/main/java/com/mycompany/applicationone/controller/GreetingsController.java
 create mode 100644 src/main/resources/application.properties
 create mode 100644 src/test/java/com/mycompany/applicationone/ApplicationOneApplicationTests.java
Lenovo@DESKTOP-JOKLOT4 MINGW64 ~
boot with Docker (master)
$ git branch -M main
Lenovo@DESKTOP-JOKLOT4 MINGW64 ~
boot with Docker (main)
$ git remote add origin https://github.com/Mouhib-hero/Springboot-with-Docker.git

```

```

Lenovo@DESKTOP-JOKLOT4 MINGW64 ~
boot with Docker (main)
$ git push -u origin main
Enumerating objects: 30, done.
Counting objects: 100% (30/30), done.
Delta compression using up to 8 threads
Compressing objects: 100% (21/21), done.
Writing objects: 100% (30/30), 53.38 KiB / 8.90 MiB/s, done.
Total 30 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Mouhib-hero/Springboot-with-Docker.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
Lenovo@DESKTOP-JOKLOT4 MINGW64 ~
boot with Docker (main)
$ 

```

On fait push pour notre simple web application

```
MINGW64:/f/Users/Lenovo/Desktop/TPs_RT5/TP_devops/FirstPipeline
Lenovoo@DESKTOP-JOKLOT4 MINGW64 pipeline (master)
$ git branch -M main

Lenovoo@DESKTOP-JOKLOT4 MINGW64 pipeline (main)
$ git remote add origin https://github.com/Mouhib-hero/First-pipeline.git

Lenovoo@DESKTOP-JOKLOT4 MINGW64 pipeline (main)
$ git push -u origin main
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (9/9), 4.15 KiB | 4.15 MiB/s, done.
Total 9 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Mouhib-hero/First-pipeline.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

Lenovoo@DESKTOP-JOKLOT4 MINGW64 pipeline (main)
$
```

First-pipeline Private Unwatch 1

main 1 branch 0 tags Go to file Add file Code

Mouhib-hero first pipeline commit	a67a498 3 minutes ago	1 commit
.vscode	first pipeline commit	3 minutes ago
templates	first pipeline commit	3 minutes ago
app.py	first pipeline commit	3 minutes ago
login_attempts.log	first pipeline commit	3 minutes ago

Add a README with an overview of your project. Add a README

On va manipuler jenkins pour la création du pipeline :

Enter an item name

First-WebApp-Pipeline

» Required field



Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.



Folder

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

OK

Organization Folder

Pipeline

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?



! Please enter Git repository.

Credentials ?

- none -

Add ▾

Save

Apply

Jenkins

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) > jenkinsUser/***** (This is a credential for github)

Update credentials

Scope ? Global (Jenkins, nodes, items, all child items, etc)

Username ? [REDACTED]

Treat username as secret ?

Password ? Concealed [Change Password](#)

ID ? GitHubCredentials

Description ? This is a credential for github

[Save](#)

Dashboard > First-WebApp-Pipeline > Configuration

Configure

General Advanced Project Options Pipeline

Repositories

Repository URL ? https://github.com/Mouhib-hero/First-pipeline

Credentials ? jenkinsUser/***** (This is a credential for github)

Add Advanced Add Repository

Branches to build

Branch Specifier (blank for 'any') ? */main

Add Branch Save Apply

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

Kind

Username with password

Scope ?
Global (Jenkins, nodes, items, all child items, etc)

Username ?
starprophecy

Treat username as secret ?

Password ?
.....

ID ?
DockerHubCredentials

Description ?
This is a simple credential for DockerHub account!

Create

First-pipeline (Private)

Unwatch 1 | Fork 0 | Star 0

main · 1 branch · 0 tags

Go to file · Add file · Code · About

Mouhib-hero added Dockerfile and Jenkinsfile · 9ae4d07 now · 2 commits

.vscode	first pipeline commit	1 hour ago
templates	first pipeline commit	1 hour ago
Dockerfile	added Dockerfile and Jenkinsfile	now
Jenkinsfile	added Dockerfile and Jenkinsfile	now
app.py	first pipeline commit	1 hour ago
login_attempts.log	first pipeline commit	1 hour ago

Add a README with an overview of your project. · Add a README

About

First pipeline

- Activity
- 0 stars
- 1 watching
- 0 forks

Releases

No releases published · Create a new release

Packages

No packages published · Publish your first package

On clique sur build now ➔ erreur car la méthode d'authentification n'est plus supportée.

Donc on utilisera cette méthode d'authentification : github adresse email avec le token github en tant que mot de passe

The screenshot shows the Jenkinsfile content in VS Code:

```
1 pipeline {
2     agent any
3
4     stages {
5         stage('Pull from GitHub') {
6             steps {
7                 git url: 'https://github.com/Mouhib-hero/First-pipeline', branch: 'main',
8                 credentialsId: 'GitHubCredentials' //jenkins-github-creds
9                 sh "ls -l"
10            }
11        }
12
13        stage('Build Docker Image') {
14            steps {
15                // Build Docker image
16                sh 'docker build -t starprophecy/simplewebapp .'
17            }
18        }
19
20        stage('Push to DockerHub') {
21            steps {
22                withCredentials([usernamePassword(credentialsId: 'DockerHubCredentials', passwordVariable: 'DOCKERHUB_PASSWORD')])
23                sh 'docker login -u $DOCKERHUB_USERNAME -p $DOCKERHUB_PASSWORD'
24                sh 'docker push starprophecy/simplewebapp'
25            }
26        }
27    }
28 }
29
30 }
```

The screenshot shows the Dockerfile content in VS Code:

```
1 # Using the right version of python
2 FROM python:3.8-slim
3
4 # Set the working directory in the container
5 WORKDIR /
6
7 # Copy all files to the container
8 COPY .
9
10 # Expose the port that the application will run on
11 EXPOSE 5000
12
13 # Set the command to run the application using Flask development server
14 CMD ["python", "app.py"]
15 |
```

On fait des builds sur notre pipeline et on remarque que ça a marché sans aucun problème

Jenkins

Dashboard > First-WebApp-Pipeline >

Pipeline First-WebApp-Pipeline

Status Changes Build Now Configure Delete Pipeline Full Stage View Rename Pipeline Syntax Build History trend Filter builds... #15 Oct 17, 2023, 12:27 PM Atom feed for all Atom feed for failures

Add description Disable Project

Stage View

Average stage times: (Average full run time: ~39s)

Declarative: Checkout SCM	Pull from GitHub	Build Docker Image	Push to DockerHub
1s	1s	4s	26s

#15 Oct 17, 13:27 1 commit

1s 1s 4s 26s

Permalinks

- Last build (#15), 1 min 41 sec ago
- Last stable build (#15), 1 min 41 sec ago
- Last successful build (#15), 1 min 41 sec ago
- Last completed build (#15), 1 min 41 sec ago

REST API Jenkins 2.414.2

docker hub Search Docker Hub Explore Repositories Organizations Help Upgrade starprophecy

starprophecy [Edit profile](#)

Community User Joined October 14, 2023

Repositories Starred Contributed

Displaying 1 to 1 repositories

 starprophecy/simplewebapp · 2 · 0
By starprophecy · Updated 7 minutes ago
Image

Création d'un deuxième pipeline en utilisant l'application Springboot avec Docker :

Le pipeline qu'on va créer dans cette partie fait la même chose que le pipeline qui précède, la différence est dans les applications utilisées. Le code de l'application fournie « springboot with docker » sera utilisé pour cette partie. On doit clarifier que le dossier « target » dans lequel on trouve généralement le fichier .jar de l'application ne doit pas être « pushed » sur github puisqu'on doit le générer de nouveau à chaque fois dans notre pipeline. Pour cela, on ajoute la commande qui permet de créer ce type de fichier .jar « mvn clean package » dans le fichier dockerfile donc il correspondra à :

```
1  FROM maven:3.8.4 AS build
2
3  # Set the working directory in the container
4  WORKDIR /app
5
6  # Copy the source code to the container
7  COPY . .
8
9  # Build the JAR
10 RUN mvn package
11 RUN mvn test
12
13 # Use a smaller base image for the final image
14 FROM openjdk:11
15
16 # Set the working directory in the container
17 WORKDIR /app
18
19 # Copy the JAR from the build image
20 COPY --from=build /app/target/app.jar .
21
22 # Expose the port that the application will run on
23 EXPOSE 8080
24
25 # Set the command to run the application using the JAR
26 CMD ["java", "-jar", "app.jar"]
```

The screenshot shows the GitHub Copilot interface with a Jenkinsfile open. The file contains a multi-stage pipeline for building a Docker image and pushing it to DockerHub. The stages include pulling from GitHub, building the Docker image, and pushing it to DockerHub using credentials.

```

1 pipeline {
2     agent any
3
4     stages {
5         stage('Pull from GitHub') {
6             steps {
7                 git url: 'https://github.com/Lquit/tpdevops', branch: 'main',
8                 credentialsId: 'GitHubCredentials' //jenkins-github-creds
9                 sh "ls -ltr"
10            }
11        }
12
13        stage('Build Docker Image') {
14            steps {
15                // Build Docker image
16                sh 'docker build -t lquit/simplewebapp .'
17                //sh './mvnw clean compile'
18            }
19        }
20        stage('Push to DockerHub') {
21            steps {
22                withCredentials([usernamePassword(credentialsId: 'DockerHubCredentials', passwordVariable: 'DOCKERHUB_PASSWORD', usernameVariable: 'DOCKERHUB_USERNAME')])
23                sh 'docker login -u $DOCKERHUB_USERNAME -p $DOCKERHUB_PASSWORD'
24                sh 'docker push lquit/simplewebapp'
25            }
26        }
27    }
28 }
29

```

[Documentation](#) • [Share feedback](#)

Lancement du pipeline :

The screenshot shows the Jenkins job console output for build #13. It displays the log of the Jenkinsfile execution, including the build process, tests, and deployment steps. The log shows the application being built, tests running, and the final success message.

```

[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 3.075 s - in com.mycompany.applicationone.ApplicationOneApplicationTests
#11 7.636 [INFO]
#11 7.636 [INFO] Results:
#11 7.636 [INFO]
#11 7.636 [INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
#11 7.636 [INFO]
#11 7.641 [INFO] -----
#11 7.641 [INFO] BUILD SUCCESS
#11 7.641 [INFO] -----
#11 7.643 [INFO] Total time: 5.653 s
#11 7.644 [INFO] Finished at: 2023-10-18T23:44:46Z
#11 7.644 [INFO] -----
#11 DONE 7.7s

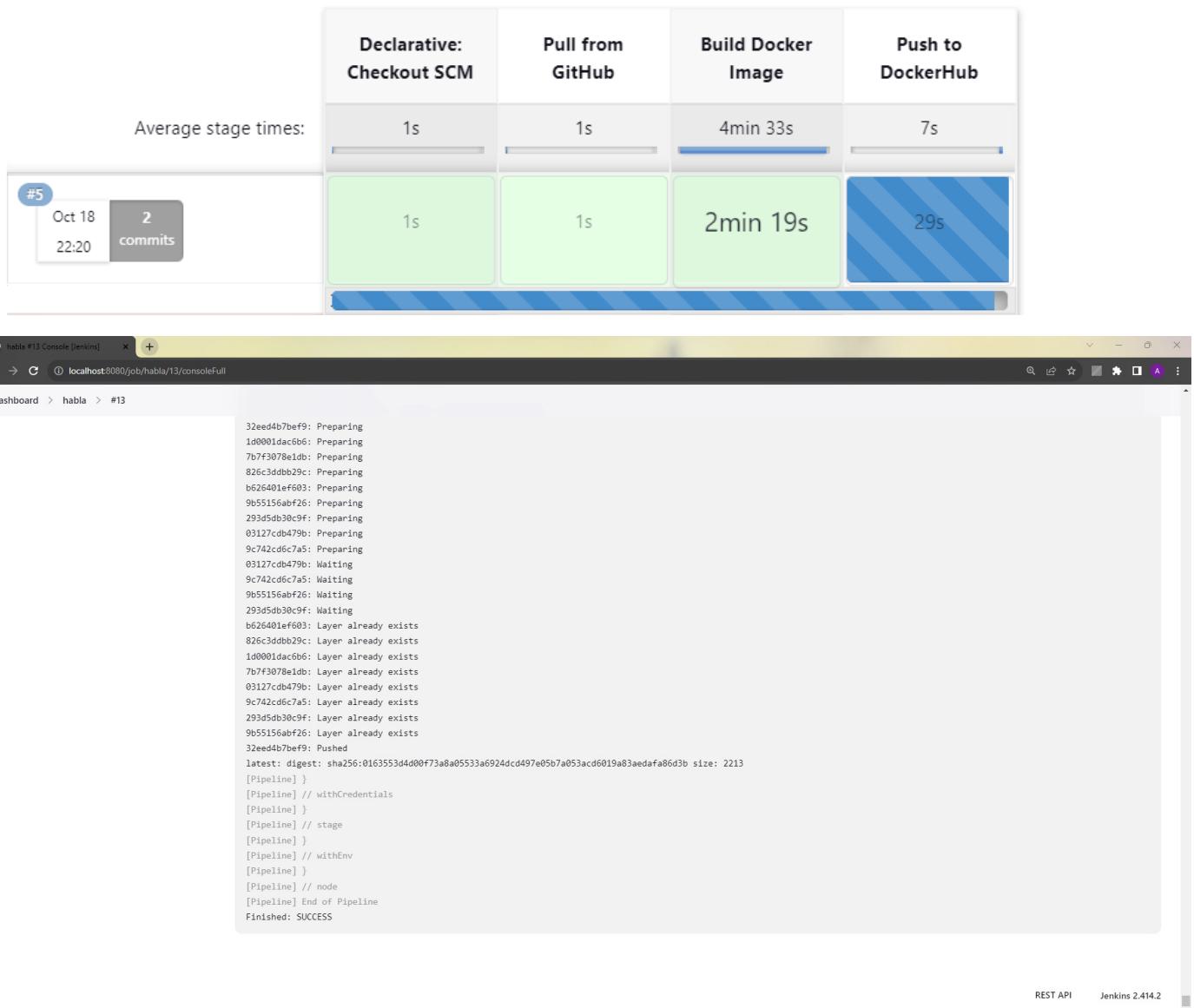
#12 [stage-1 2/3] WORKDIR /app
#12 CACHED

#13 [stage-1 3/3] COPY --from=build /app/target/app.jar .
#13 DONE 0.1s

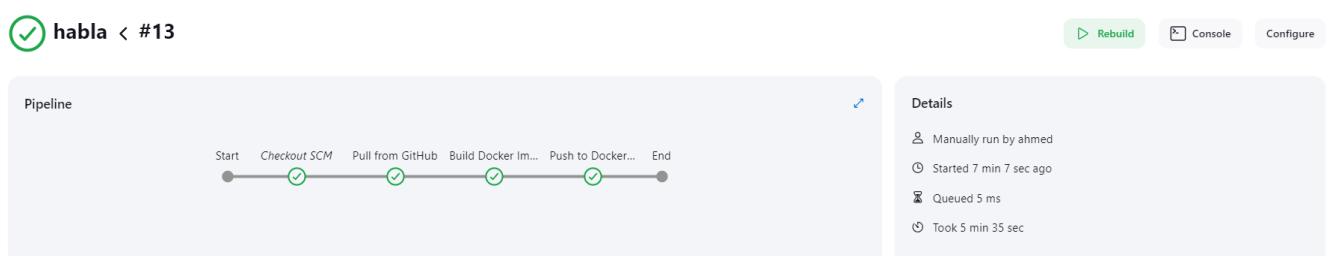
#14 exporting to image

```

Stage View



On génère le schéma du pipeline à l'aide d'un plugin qu'on a installé dans jenkins (« pipeline graph view ») :



- ⇒ Ce schéma est identique pour les deux pipelines puisqu'on a utilisé la même structure du pipeline mais avec des applications différentes (l'une avec Flask et l'autre avec Springboot).

Conclusion :

Dans ce TP nous avons appris à manipuler Docker en utilisant des images nginx dans le début du TP, de manipuler une application springboot avec docker, de voir les communications entre des containers (mongodb et une application springboot) et d'utiliser docker compose pour la mise en place de deux ressources en même temps tout en lançant des tests avec JUnit dans les méthodes de tests définies dans le code.

Dans la dernière partie du TP, nous avons travaillé sur la création de deux pipelines en utilisant l'outil Jenkins. Ces deux pipelines permettent de récupérer du code d'une application web en Flask depuis github ou bien une application springboot depuis github, Construire une image docker à partir d'un docker file qui existe dans la même repository que l'application web (si c'est le cas de l'application springboot, on ajoute aussi qu'il faut créer un fichier .jar à travers la commande « mvn clean package » qui se fait dans le dockerfile) faire un build de l'image de l'application et puis faire un push de cette image dans dockerhub repository de nos propres comptes.