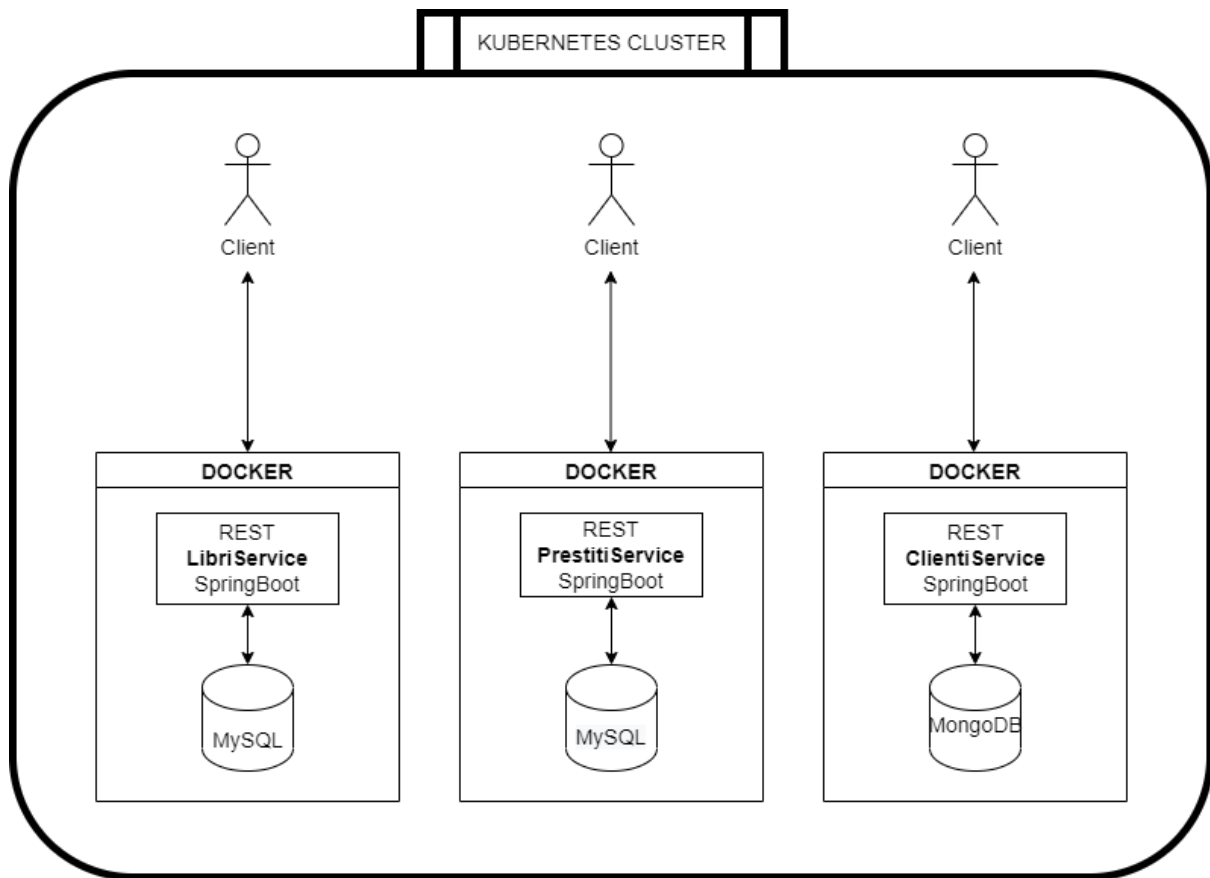


Documentazione_LabMicroservizi



Per questo progetto, ho iniziato disegnando uno schema di ciò che avrei voluto fare.

Sono partita con la creazione dei 3 microservizi utilizzando eclipse:

- il linguaggio di programmazione è **Java**
- ho utilizzato il **framework SpringBoot**(utile per creare microservizi)
- Database utilizzati:

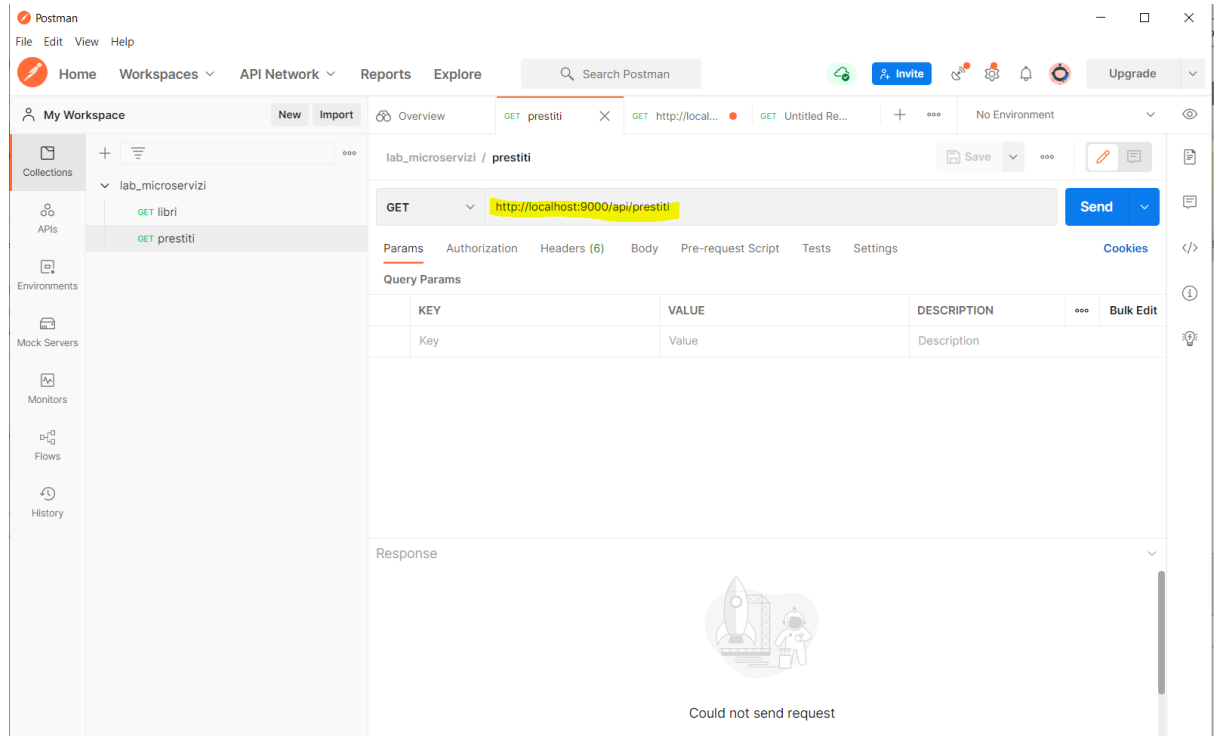
Mysql	MongoDB
microservizi Libri e Prestiti	microservizio Clienti
jpa repository	Mongo Repository

POSTMAN

Per testare i verbi HTTP presenti nei Microservizi, ho utilizzato Postman, le chiamate effettuate sono state:

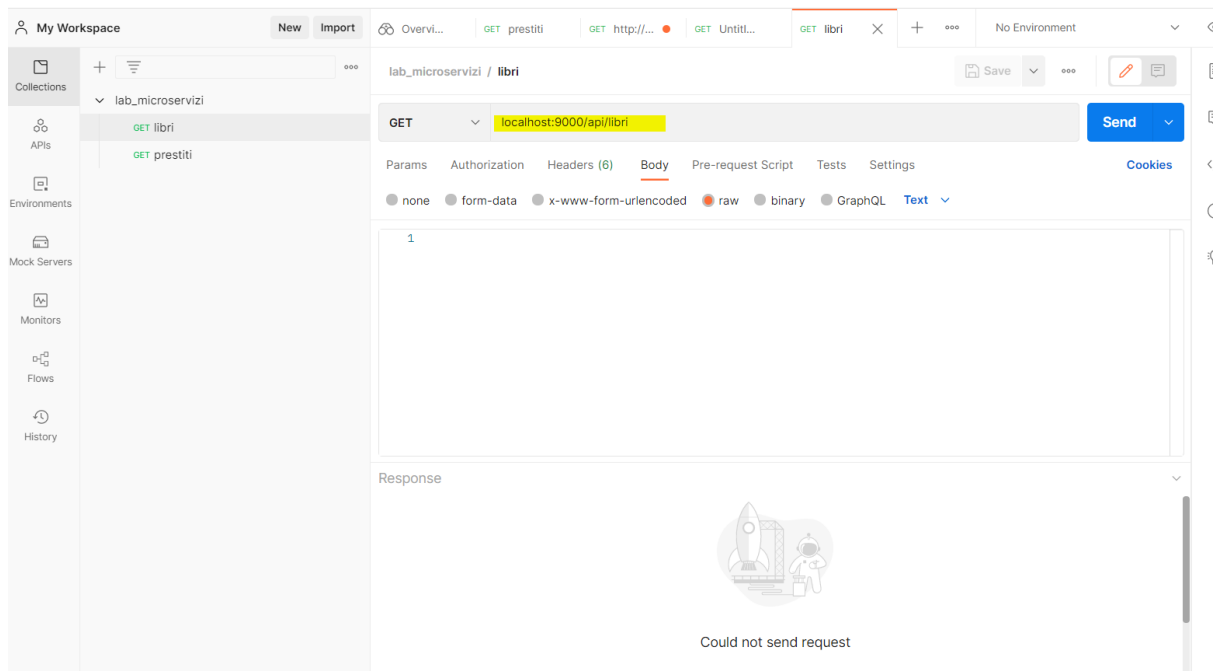
1. <http://localhost:9000/api/prestiti>

2.



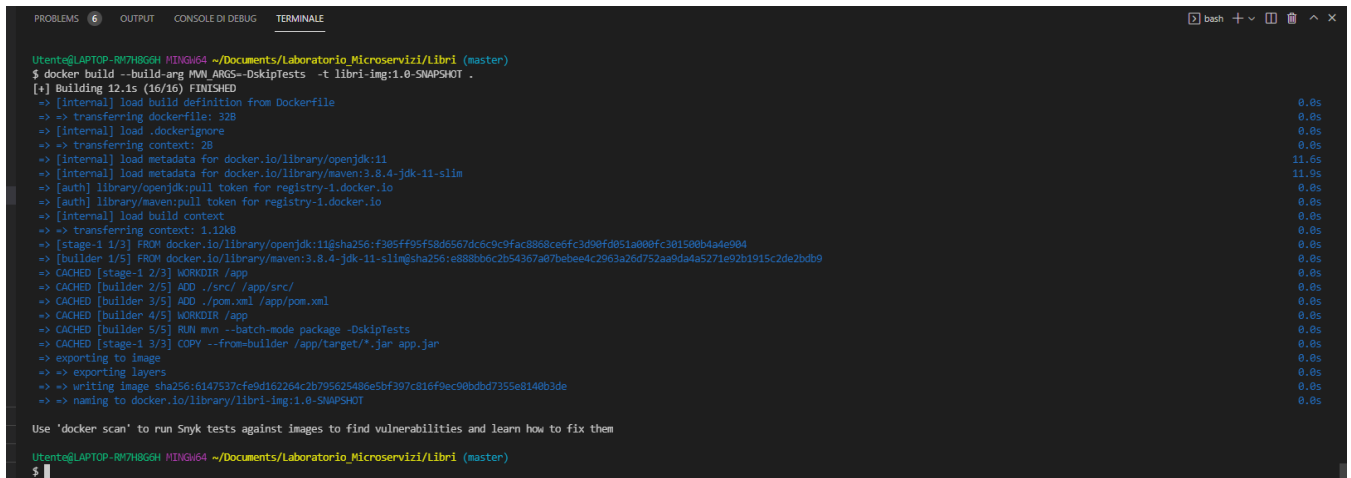
2.

2. <http://localhost:9000/api/libri>



DOCKER

Ho creato le immagini dei 3 microservizi su docker

A screenshot of a terminal window with a dark background. The terminal shows the command `$ docker build --build-arg MVN_ARGS=-DskipTests -t libri-1.0-SNAPSHOT .` and its output. The output indicates that the build is finished, context is transferred, and the image is successfully built and named. The terminal window has tabs for 'PROBLEMS', 'OUTPUT', 'CONSOLE DI DEBUG', and 'TERMINALE'. The prompt shows the user is in a directory `~/Documents/Laboratorio/Microservizi/Libri`.

```
Utente@LAPTOP-RW7H8GGH MINGW64 ~/Documents/Laboratorio/Microservizi/Libri (master)
$ docker build --build-arg MVN_ARGS=-DskipTests -t libri-1.0-SNAPSHOT .
[*] Building 12.1s (16/16) FINISHED
-> [internal] load build definition from Dockerfile
-> => transferring dockerfile: 32B
-> [internal] load .dockerignore
-> => transferring context: 2B
-> [internal] load metadata for docker.io/library/openjdk:11
-> [internal] load metadata for docker.io/library/maven:3.8.4-jdk-11-slim
-> [auth] library/openjdk:pull token for registry-1.docker.io
-> [auth] library/maven:pull token for registry-1.docker.io
-> [internal] load build context
-> => transferring context: 1.12kB
-> [stage-1 1/3] FROM docker.io/library/openjdk:11@sha256:f305ff95f58d6567dc6c9cfac8868ce6fc3d90fd051a000fc301500b4ade904
-> [builder 1/5] FROM docker.io/library/maven:3.8.4-jdk-11-slim@sha256:e888b06c2b54367a07bebee4c2963a26d752aa9da4a5271e92b1915c2de2bde9
-> CACHED [stage-1 2/3] WORKDIR /app
-> CACHED [builder 2/5] ADD ./src/ /app/src/
-> CACHED [builder 3/5] ADD ./pom.xml /app/pom.xml
-> CACHED [builder 4/5] WORKDIR /app
-> CACHED [builder 5/5] RUN mvn --batch-mode package -DskipTests
-> CACHED [stage-1 3/3] COPY --from=builder /app/target/*.jar app.jar
-> exporting to image
-> => exporting layers
-> => writing image sha256:6147537cfe9d162264c2b795625486e5bf397c816f9ec980bd0d7355e8146b3de
-> => naming to docker.io/library/libri-1.0-SNAPSHOT

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them

Utente@LAPTOP-RW7H8GGH MINGW64 ~/Documents/Laboratorio/Microservizi/Libri (master)
$
```

immagine di Libri:

`docker build --build-arg MVN_ARGS=-DskipTests -t russog0/libri-1:1.0-SNAPSHOT .`

immagine di Prestiti:

`docker build --build-arg MVN_ARGS=-DskipTests -t russog0/prestiti-1:1.0-SNAPSHOT .`

immagine di Clienti:

`docker build --build-arg MVN_ARGS=-DskipTests -t russog0/clienti-1:1.0-SNAPSHOT .`

push delle immagini su dockerhub

`docker push russog0/libri-1:1.0-SNAPSHOT`

`docker push russog0/prestiti-1:1.0-SNAPSHOT`

`docker push russog0/clienti-1:1.0-SNAPSHOT`

Installazione dei componenti su kubernetes

1. kubectl (minikube start)
2. kafka
`helm install --values kafka-values.yaml kafka bitnami/kafka`
3. elasticsearch

```
helm install --values elasticsearch-values.yaml elasticsearch
elastic/elasticsearch
```

4. Logstash

```
helm install --values logstash-values.yaml logstash elastic/logstash
```

5. Filebeat

```
helm install --values filebeat-values.yaml filebeat elastic/filebeat
```

6. kibana

```
helm install --values kibana-values.yaml kibana elastic/kibana
```

Logging and tracing

- Monitoraggio dei pod su Kubernetes con Kubectl:

1. per visualizzare i **Pods** con il comando: **kubectl get pods**

2. per visualizzare i **logs**, specificando il nome del pod: **kubectl logs pod-name**
es. **kubectl logs kafka-0**

Failed to pull image "docker.elastic.co/kibana/kibana:7.16.3":

non avevo fatto il pull dell'immagine su docker desktop di kibana e di conseguenza non riuscivo a vedere i log sull'interfaccia

```
$ kubectl get pods
NAME                                READY   STATUS              RESTARTS   AGE
customer-mongodb-7698b66f4c-vqvsz  0/1     ImagePullBackOff    0           4h7m
elasticsearch-0                     0/1     Running             9           20h
elasticsearch-1                     0/1     Running             9 (10m ago)  20h
elasticsearch-2                     0/1     Running             9           20h
filebeat-nntmx                     0/1     Running             35 (23s ago) 13h
kafka-0                             0/1     Running             75 (24s ago) 21h
kafka-zookeeper-0                  0/1     Running             35 (2m16s ago) 21h
kibana-6c6959c9c9-rwkhv            0/1     ImagePullBackOff    0           12h
logstash-0                          0/1     Running             10 (2m16s ago) 17h
```

(non funziona comunque)