



Universidad de las Fuerzas Armadas - ESPE.

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN.

Taller Docker.

Aplicaciones Distribuidas.

Estudiante:
Shirley Stefania Otuna Rojano.

Docente:
Ing. Darío Javier Morales Caiza.

Agosto 2024.

TALLER DOCKER - PRUEBAS Y DESPLIEGUE DE MICROSERVICIOS Y APLICACIÓN FRONTEND UTILIZANDO DOCKER

19/08/2024

INFORMACIÓN GENERAL

1. Introducción

i El presente informe documenta el proceso de despliegue y pruebas de un sistema compuesto por tres microservicios desarrollados en Spring Boot y una aplicación frontend construida con React. Para gestionar el despliegue de estos componentes, se ha utilizado Docker, una plataforma de contenedorización que facilita la creación, implementación y ejecución de aplicaciones mediante contenedores.

El sistema desarrollado se centra en la gestión de cursos, estudiantes y matrículas, donde cada funcionalidad está aislada en su propio microservicio. Docker se utilizó no solo para contenedizar los microservicios, sino también para gestionar sus bases de datos PostgreSQL y desplegar la interfaz de usuario. El uso de Docker Compose permitió la orquestación de todos estos componentes, asegurando que se comunicaran correctamente entre sí en un entorno aislado.

2. Objetivo General

i Implementar un sistema basado en microservicios y una aplicación frontend utilizando Docker, mediante la creación, configuración y despliegue de contenedores Docker para cada componente del sistema, con el propósito de asegurar una correcta conectividad y funcionalidad integrada entre todos los servicios, así como verificar el correcto funcionamiento del sistema a través de pruebas automatizadas y manuales.

METODOLOGÍA

1. Preparación del Entorno

i Paso 1: Crear Dockerfiles para los Microservicios

Se crearon Dockerfiles para cada microservicio (Cursos, Estudiantes, Matrículas). Cada Dockerfile define cómo construir y ejecutar el microservicio en un contenedor Docker.

a. Dockerfile para cursos-service

Ubicación: cursos/Dockerfile

```
cursos\ Dockerfile x estudiantes\ Dockerfile matriculas\ Dockerfile frontend\ Dockerfile
1 # Usar la imagen base de OpenJDK 17
2 FROM openjdk:17-jdk-slim
3
4 # Crear un directorio de trabajo en el contenedor
5 WORKDIR /app
6
7 # Copiar el JAR de la aplicación al contenedor
8 COPY target/cursos-0.0.1-SNAPSHOT.jar /app/cursos.jar
9
10 # Exponer el puerto 4100
11 EXPOSE 4100
12
13 # Comando de inicio para ejecutar el JAR
14 ENTRYPOINT ["java", "-jar", "/app/cursos.jar"]
15
```

b. Dockerfile para estudiantes-service

Ubicación: estudiantes/Dockerfile

```
cursos\ Dockerfile estudiantes\ Dockerfile x matriculas\ Dockerfile frontend\ Dockerfile
1 # Usar la imagen base de OpenJDK 17
2 FROM openjdk:17-jdk-slim
3
4 # Crear un directorio de trabajo en el contenedor
5 WORKDIR /app
6
7 # Copiar el JAR de la aplicación al contenedor
8 COPY target/estudiantes-0.0.1-SNAPSHOT.jar /app/estudiantes.jar
9
10 # Exponer el puerto 4101
11 EXPOSE 4101
12
13 # Comando de inicio para ejecutar el JAR
14 ENTRYPOINT ["java", "-jar", "/app/estudiantes.jar"]
15
```

c. Dockerfile para matriculas-service

Ubicación: matriculas/Dockerfile

```
cursos\ Dockerfile estudiantes\ Dockerfile matriculas\ Dockerfile x frontend\ Dockerfile
1 # Usar la imagen base de OpenJDK 17
2 FROM openjdk:17-jdk-slim
3
4 # Crear un directorio de trabajo en el contenedor
5 WORKDIR /app
6
7 # Copiar el JAR de la aplicación al contenedor
8 COPY target/matriculas-0.0.1-SNAPSHOT.jar /app/matriculas.jar
9
10 # Exponer el puerto 4102
11 EXPOSE 4102
12
13 # Comando de inicio para ejecutar el JAR
14 ENTRYPOINT ["java", "-jar", "/app/matriculas.jar"]
15
```



Paso 2: Crear Dockerfile para la Aplicación React

Se creó un Dockerfile para la aplicación frontend React, que maneja tanto la construcción de la aplicación como su despliegue utilizando Nginx.

a. Dockerfile para la aplicación React

Ubicación: frontend/Dockerfile

```
1 # Fase 1: Construcción de la aplicación con Node.js
2 FROM node:18 as build
3
4 # Establecer el directorio de trabajo
5 WORKDIR /app
6
7 # Copiar los archivos del proyecto al contenedor
8 COPY package*.json ./
9
10 # Instalar las dependencias
11 RUN npm install
12
13 # Copiar el resto de los archivos del proyecto
14 COPY . .
15
16 # Construir la aplicación para producción
17 RUN npm run build
18
19 # Fase 2: Configuración de Nginx para servir la aplicación
20 FROM nginx:alpine
21
22 # Copiar los archivos construidos desde la fase anterior
23 COPY --from=build /app/build /usr/share/nginx/html
24
25 # Exponer el puerto en el que Nginx está sirviendo
26 EXPOSE 80
27
28 # Comando para iniciar Nginx
29 CMD ["nginx", "-g", "daemon off;"]
```



Paso 3: Crear Archivos de Configuración de PostgreSQL

Se crearon scripts SQL de inicialización para configurar las bases de datos PostgreSQL utilizadas por cada microservicio.

a. Configuración de la base de datos para cursos

Ubicación: cursos/init-cursos.sql

```
1 CREATE DATABASE cursos_db;
2 CREATE USER cursos_user WITH ENCRYPTED PASSWORD 'cursos_password';
3 GRANT ALL PRIVILEGES ON DATABASE cursos_db TO cursos_user;
```

b. Configuración de la base de datos para estudiantes

Ubicación: estudiantes/init-estudiantes.sql

```
1 CREATE DATABASE estudiantes_db;
2 CREATE USER estudiantes_user WITH ENCRYPTED PASSWORD 'estudiantes_password';
3 GRANT ALL PRIVILEGES ON DATABASE estudiantes_db TO estudiantes_user;
```

c. Configuración de la base de datos para matriculas

Ubicación: matriculas/init-matriculas.sql

```
estudiantes\Dockerrfile x EstudianteController.java matriculas\Dockerrfile init-estudiantes.sql init-cursos.sql init-ma
*.sql files are supported by IntelliJ IDEA Ultimate Try IntelliJ IDEA Ultimate
1 CREATE DATABASE matriculas_db;
2 CREATE USER matriculas_user WITH ENCRYPTED PASSWORD 'matriculas_password';
3 GRANT ALL PRIVILEGES ON DATABASE matriculas_db TO matriculas_user;
```

2. Construcción y Publicación de Imágenes Docker



Se construyeron las imágenes Docker para cada microservicio y la aplicación frontend, y se publicaron en Docker Hub.



Paso 1: Construir las Imágenes Docker

Se utilizaron los siguientes comandos para construir las imágenes.

a. Construir la imagen para cursos-service

cd cursos

docker build -t shirley24/cursos-service .

```
Terminal Local x Local (2) x Local (3) x Local (4) x
PS C:\Users\shirley\Documents\Universidad\Bvo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app\cursos
os> docker build -t shirley24/cursos-service .
[+] Building 56.7s (9/9) FINISHED docker:desktop-linux
=> [internal] load build definition from Dockerfile 0.0s
=> transferring dockerfile: 408B 0.0s
=> [internal] load metadata for docker.io/library/openjdk:17-jdk-slim 4.0s
=> [auth] library/openjdk:pull token for registry-1.docker.io 0.0s
=> [internal] load .dockerignore 0.0s
=> transferring context: 2B 0.0s
=> [1/3] FROM docker.io/library/openjdk:17-jdk-slim@sha256:aaa3b3cb27a3e520b8f16863d0580c438e 47.6s
=> resolve docker.io/library/openjdk:17-jdk-slim@sha256:aaa3b3cb27a3e520b8f16863d0580c438e 0.1s
=> sha256:aaa3b3cb27a3e520b8f16863d0580c438e55ecf4b0c126b41f48c3f62f9774 547B / 547B 0.0s
=> sha256:779635c0c3d23cc8dbab28c1ee4cf2a9202e198dfc8f4c0b279824d9b8e0f22 953B / 953B 0.0s
=> sha256:37cb44321d0423bc5726aa3bff658daf00478e4cdf2a3b8091f78531053425ad 4.80kB / 4.80kB 0.0s
=> sha256:44d3aa8d076675d49d85180b0ced9dae7210fe4dff4b0bb422b9cf384e591d0 1.58MB / 1.58MB 0.4s
=> sha256:6ca999fd16e8bd02faad6a60e1334878528b5a4b5487850a76e0c08a7a27 187.90MB / 187.90MB 38.5s
=> sha256:1fe172e4850f03bb45d41a20174112bc119fbfec42a650edbb8491ae32e3c3 31.38MB / 31.38MB 8.2s
=> extracting sha256:1fe172e4850f03bb45d41a20174112bc119fbfec42a650edbb8491ae32e3c3 6.1s
=> extracting sha256:44d3aa8d076675d49d85180b0ced9dae7210fe4dff4b0bb422b9cf384e591d0 0.4s
=> extracting sha256:6ca999fd16e8bd02faad6a60e1334878528b5a4b5487850a76e0c08a7a27d56 8.7s
=> [internal] load build context 3.1s
```

b. Construir la imagen para estudiantes-service

cd estudiantes

docker build -t shirley24/estudiantes-service .

```
Terminal Local x Local (2) x Local (3) x Local (4) x
=> [3/3] COPY target/estudiantes-0.0.1-SNAPSHOT.jar /app/estudiantes.jar 0.4s
=> exporting to image 0.5s
=> exporting layers 0.4s
=> writing image sha256:3255305cf36341bd2ad98b475092a521a1eb0b2453062e669cfc1652b5c18fac 0.0s
=> naming to docker.io/shirley24/estudiantes-service 0.0s

What's next:
View a summary of image vulnerabilities and recommendations + docker scout quickview
PS C:\Users\shirley\Documents\Universidad\Bvo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app\estudiantes>
PS C:\Users\shirley\Documents\Universidad\Bvo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app\estudiantes> |
```

c. Construir la imagen para matriculas-service

cd matriculas

docker build -t shirley24/matriculas-service .

```
Terminal Local (2) Local (3) Local (4)
=> [5/5] COPY target/matriculas-0.0.1-SNAPSHOT.jar /app/matriculas.jar 1.0s
=> exporting to image 0.5s
=> exporting layers 0.4s
=> writing image sha256:43ce8444b8bf2c5270ab393837fdcae2fde0ec187b07589275a4e119f16bd6bd 0.0s
=> naming to docker.io/shirley24/matriculas-service 0.0s

What's next:
View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\shirl\Documents\Universidad\8vo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app\matriculas>
```

d. Construir la imagen para la aplicación React

cd frontend

docker build -t shirley24/react-app .

```
Terminal Local (2) Local (3) Local (4)
PS C:\Users\shirl\Documents\Universidad\8vo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app\frontend> docker build -t shirley24/react-app .
[+] Building 132.2s (9/15) docker:desktop-linux
[+] Building 132.3s (9/15) docker:desktop-linux
[+] Building 132.4s (9/15) docker:desktop-linux
=> WARN: FromAsCasing: 'as' and 'FROM' keywords' casing do not match (Line 2) 0.0s
=> [internal] load build context 129.1s
[+] Building 132.5s (9/15) docker:desktop-linux
=> WARN: FromAsCasing: 'as' and 'FROM' keywords' casing do not match (Line 2) 0.0s
=> [internal] load build context 129.2s
[+] Building 132.6s (9/15) docker:desktop-linux
=> WARN: FromAsCasing: 'as' and 'FROM' keywords' casing do not match (Line 2) 0.0s
=> [internal] load build context 129.3s
=> ==> transferring context: 160.63MB 129.2s
=> ==> extracting sha256:532b9a30583c1bf82204f3c0c8054882bace1669cc85fdeb45b8f88b4db82833 0.0s
[+] Building 132.7s (9/15) docker:desktop-linux
=> WARN: FromAsCasing: 'as' and 'FROM' keywords' casing do not match (Line 2) 0.0s
=> [internal] load build context 129.4s
=> ==> transferring context: 160.69MB 129.3s
=> ==> extracting sha256:532b9a30583c1bf82204f3c0c8054882bace1669cc85fdeb45b8f88b4db82833 0.0s
=> ==> extracting sha256:41c49c0b06a69c286164443a90e47a59e906186088b706d32abe1071d9f262b0 0.0s
[+] Building 132.8s (9/15) docker:desktop-linux
```



Paso 2: Publicar las Imágenes en Docker Hub

Después de iniciar sesión en Docker Hub, se publicaron las imágenes:

a. Iniciar sesión en Docker Hub

docker login

b. Publicar las imágenes

docker push shirley24 /cursos-service

```
Terminal Local (2) Local (3) Local (4)
PS C:\Users\shirl\Documents\Universidad\8vo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app\cursos> docker login
Authenticating with existing credentials...
Login Succeeded

PS C:\Users\shirl\Documents\Universidad\8vo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app\cursos> docker push shirley24/cursos-service
Using default tag: latest
The push refers to repository [docker.io/shirley24/cursos-service]
433faac5f1c6: Pushing [====] 2.95MB/46.45MB
ae71eef449bd: Pushing 1.536kB
0be49b267e47: Mounted from Library/openjdk
13a34b6fff78: Preparing
9c1b6d6c1e6: Mounted from Library/openjdk
```

docker push shirley24/estudiantes-service

```
Terminal Local x Local (2) x Local (3) x Local (4) x
PS C:\Users\shirl\Documents\Universidad\8vo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app\estudiantes> docker login
Authenticating with existing credentials...
Login Succeeded
PS C:\Users\shirl\Documents\Universidad\8vo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app\estudiantes> docker push shirley24/estudiantes-service
Using default tag: latest
The push refers to repository [docker.io/shirley24/estudiantes-service]
991555f70648: Pushing [=====] 33.42MB/46.45MB
aa71eef4498d: Mounted from shirley24/cursos-service
6be690267e47: Mounted from shirley24/cursos-service
13a34b6fff78: Mounted from shirley24/cursos-service
9c1b6dddc1e6: Mounted from shirley24/cursos-service
[ ]
```

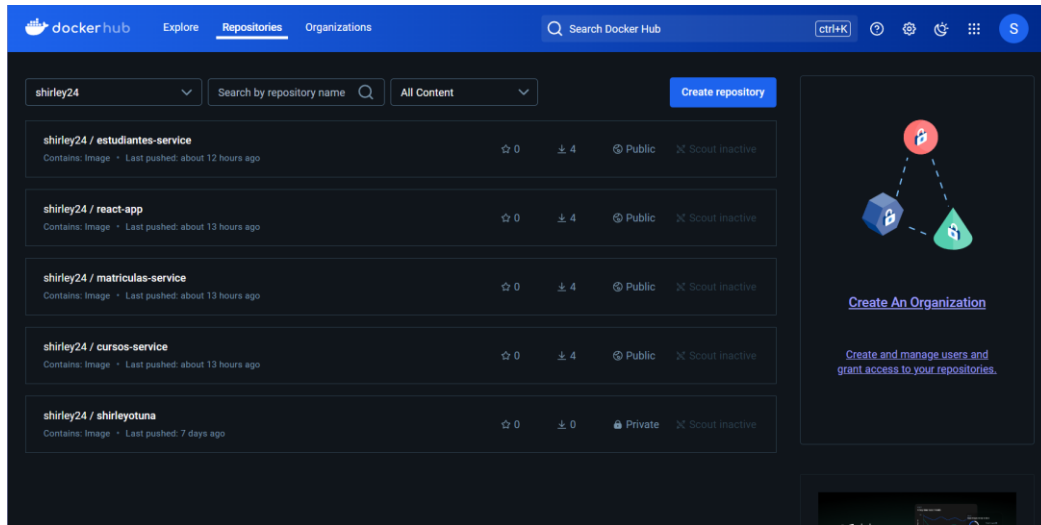
docker push shirley24/matriculas-service

```
Terminal Local x Local (2) x Local (3) x Local (4) x
PS C:\Users\shirl\Documents\Universidad\8vo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app\matriculas> docker login
Authenticating with existing credentials...
Login Succeeded
PS C:\Users\shirl\Documents\Universidad\8vo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app\matriculas> docker push shirley24/matriculas-service
Using default tag: latest
The push refers to repository [docker.io/shirley24/matriculas-service]
5ea7e6c37ba: Pushing [=====] 33.92MB/46.45MB
aa71eef4498d: Mounted from shirley24/estudiantes-service
6be690267e47: Mounted from shirley24/estudiantes-service
13a34b6fff78: Mounted from shirley24/estudiantes-service
9c1b6dddc1e6: Mounted from shirley24/estudiantes-service
[ ]
```

docker push shirley24/react-app

```
Terminal Local x Local (2) x Local (3) x Local (4) x
PS C:\Users\shirl\Documents\Universidad\8vo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app\frontend> docker login
Authenticating with existing credentials...
Login Succeeded
PS C:\Users\shirl\Documents\Universidad\8vo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app\frontend> docker push shirley24/react-app
Using default tag: latest
The push refers to repository [docker.io/shirley24/react-app]
752b01510575: Pushing [=====] 584.7kB
26d9c9583797: Mounted from library/nginx
2cd44bacf827: Pushed
f3719eb0da5e: Pushing [=====] 5.12kB
9a2d14b22cbe: Pushed
16f2939def51: Waiting
b65aff77ec426: Waiting
c028c01f43bc: Waiting
78561cef0761: Waiting
[ ]
```

c. Docker Hub



3. Despliegue del Sistema con Docker Compose



Se configuró un archivo docker-compose.yml para coordinar el despliegue de todos los servicios y sus bases de datos.

a. Archivo docker-compose.yml

b. Ubicación: appdocker-compose.yml

```
1  version: '3'
2
3  services:
4    cursos-db:
5      image: postgres:15
6      container_name: cursos-db
7      environment:
8        POSTGRES_DB: cursos_db
9        POSTGRES_USER: cursos_user
10       POSTGRES_PASSWORD: cursos_password
11      volumes:
12        - ./cursos/init-cursos.sql:/docker-entrypoint-initdb.d/init-cursos.sql
13      networks:
14        - microservices-network
15      ports:
16        - "5432:5432"
17
18  estudiantes-db:
19    image: postgres:15
20    container_name: estudiantes-db
21    environment:
22      POSTGRES_DB: estudiantes_db
23      POSTGRES_USER: estudiantes_user
24      POSTGRES_PASSWORD: estudiantes_password
25    volumes:
26      - ./estudiantes/init-estudiantes.sql:/docker-entrypoint-initdb.d/init-estudiantes.sql
27    networks:
28      - microservices-network
```



```

3   services:
18  estudiantes-db:
29      ports:
30      - "5433:5432"
31
32  ▶ matriculas-db:
33      image: postgres:15
34      container_name: matriculas-db
35      environment:
36          POSTGRES_DB: matriculas_db
37          POSTGRES_USER: matriculas_user
38          POSTGRES_PASSWORD: matriculas_password
39      volumes:
40      - ./matriculas/init-matriculas.sql:/docker-entrypoint-initdb.d/init-matriculas.sql
41      networks:
42      - microservices-network
43      ports:
44      - "5434:5432"
45
46  ▶ corsos-service:
47      image: shirley24/corsos-service
48      environment:
49          SPRING_DATASOURCE_URL: jdbc:postgresql://corsos-db:5432/corsos_db
50          SPRING_DATASOURCE_USERNAME: corsos_user
51          SPRING_DATASOURCE_PASSWORD: corsos_password
52      ports:
53      - "4100:4100"
54      networks:
55      - microservices-network

```

```

3   services:
46  corsos-service:
56      depends_on:
57      - corsos-db
58
59  ▶ estudiantes-service:
60      image: shirley24/estudiantes-service
61      environment:
62          SPRING_DATASOURCE_URL: jdbc:postgresql://estudiantes-db:5432/estudiantes_db
63          SPRING_DATASOURCE_USERNAME: estudiantes_user
64          SPRING_DATASOURCE_PASSWORD: estudiantes_password
65      ports:
66      - "4101:4101"
67      networks:
68      - microservices-network
69      depends_on:
70      - estudiantes-db
71
72  ▶ matriculas-service:
73      image: shirley24/matriculas-service
74      environment:
75          SPRING_DATASOURCE_URL: jdbc:postgresql://matriculas-db:5432/matriculas_db
76          SPRING_DATASOURCE_USERNAME: matriculas_user
77          SPRING_DATASOURCE_PASSWORD: matriculas_password
78      ports:
79      - "4102:4102"
80      networks:
81      - microservices-network

```

```

82      depends_on:
83      - matriculas-db
84
85  ▶ react-app:
86      image: shirley24/react-app
87      ports:
88      - "80:80"
89      networks:
90      - microservices-network
91
92  networks:
93      microservices-network:
94      driver: bridge
95

```



Paso 1: Desplegar los Servicios

Utilizando Docker Compose, se levantaron todos los servicios:

- c. Navegar a la raíz del proyecto donde se encuentra el archivo docker-compose.yml.
- d. Ejecutar el siguiente comando para los servicios

docker-compose up -d

```
Terminal Local x Local (2) x Local (3) x Local (4) x Local (5) x + -
PS C:\Users\shirl\Documents\Universidad\8vo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app> docker-compose up -d
time="2024-08-20T22:55:40-05:00" level=warning msg="C:\\Users\\shirl\\Documents\\Universidad\\8vo Semestre\\Aplicaciones Distribuidas\\Segundo Parcial\\app\\docker-compose.yml: 'version' is obsolete"
[+] Running 5/5
 ✓ Network app_microservices-network Created 0.2s
 ✓ Container app-react-app-1 Started 2.2s
 ✓ Container app-cursos-service-1 Started 2.2s
 ✓ Container app-estudiantes-service-1 Started 2.2s
 ✓ Container app-matriculas-service-1 Started 2.2s
PS C:\Users\shirl\Documents\Universidad\8vo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app>
```

- e. Ejecutar el siguiente comando para las bases de datos

docker-compose up -d

```
Terminal Local x Local (2) x Local (3) x Local (4) x Local (5) x + -
time="2024-08-20T23:06:55-05:00" level=warning msg="C:\\Users\\shirl\\Documents\\Universidad\\8vo Semestre\\Aplicaciones Distribuidas\\Segundo Parcial\\app\\docker-compose.yml: 'version' is obsolete"
[+] Running 9/27
  matriculas-db Pulling 14.2s
  estudiantes-db Pulling 14.2s
  cursos-db [##### ] Pulling 14.2s
    - e6fff07796ad Extracting [=====] 10.91MB/29.13MB 9.7s
    - ac2c2f5ebbf Download complete 1.5s
    - 6423808c0870 Download complete 2.8s
    - 9fc1a9a46316 Download complete 4.0s
    - f486755b3b27 Download complete 7.1s
    - e3f131c9ffee Download complete 6.9s
    - bb37c85245b8 Download complete 7.5s
    - af85de36d2ba Download complete 7.7s
    - 3fb6cac099e8 Downloading [====] 6.974MB/106.9MB 9.7s
    - 31efa1bde9c4 Download complete 8.4s
    - 15bf20fe9460 Download complete 8.9s
    - 3216df30c32c Waiting 9.7s
    - cecbd5a08338 Waiting 9.7s
    - a733d89613cc Waiting 9.7s

[+] Running 7/7
 ✓ 6423808c0870 Pull complete 14.6s
 ✓ 9fc1a9a46316 Pull complete 14.7s
 ✓ f486755b3b27 Pull complete 15.9s
 ✓ e3f131c9ffee Pull complete 16.1s
 ✓ bb37c85245b8 Pull complete 16.2s
 ✓ af85de36d2ba Pull complete 16.3s
 ✓ 3fb6cac099e8 Pull complete 36.5s
 ✓ 31efa1bde9c4 Pull complete 36.6s
 ✓ 15bf20fe9460 Pull complete 36.7s
 ✓ 3216df30c32c Pull complete 36.7s
 ✓ cecbd5a08338 Pull complete 36.8s
 ✓ a733d89613cc Pull complete 36.9s
[+] Running 7/7
 ✓ Container app-react-app-1 Running 0.0s
 ✓ Container estudiantes-db Started 5.5s
 ✓ Container cursos-db Started 5.5s
 ✓ Container matriculas-db Started 5.5s
 ✓ Container app-estudiantes-service-1 Started 2.9s
 ✓ Container app-matriculas-service-1 Started 2.8s
 ✓ Container app-cursos-service-1 Started 2.8s
PS C:\Users\shirl\Documents\Universidad\8vo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app>
```

i Paso 2: Verificar que los Contenedores Estén Corriendo

Se utilizó el comando docker ps para verificar que todos los contenedores estaban en ejecución correctamente:

a. Ejecutar el siguiente comando:

docker ps

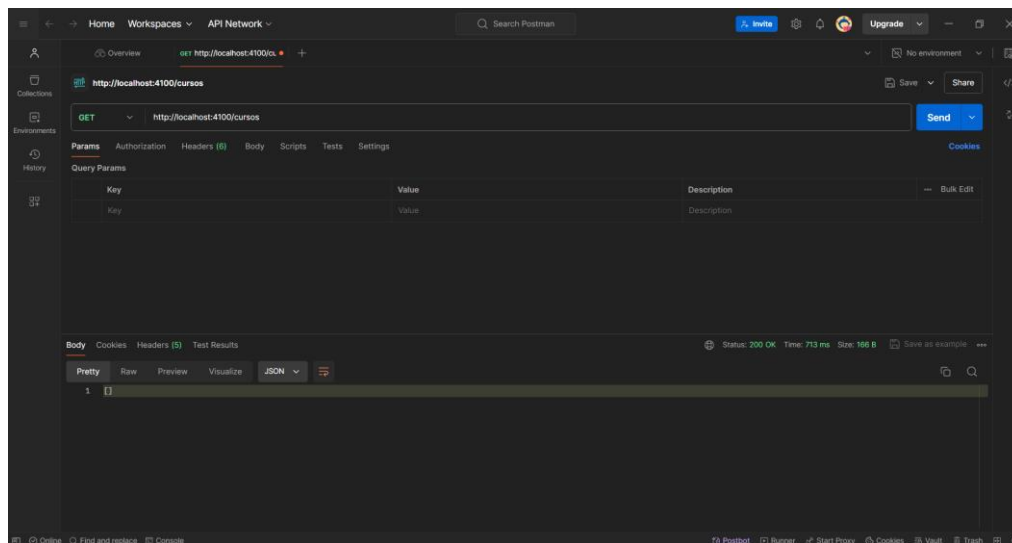
```
Terminal Local (2) Local (3) Local (4) Local (5)
PS C:\Users\shirl\Documents\Universidad\8vo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app> docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                               NAMES
82c8b13e71e6   shirley24/matriculas-service       "java -jar /app/matr..." About a minute ago Up 23 seconds  0.0.0.0:4102->4102/tcp              app-matriculas-service-1
81ea7ef995c5   shirley24/cursos-service           "java -jar /app/curs..." About a minute ago Up 25 seconds  0.0.0.0:4100->4100/tcp              app-cursos-service-1
37431927b26    shirley24/estudiantes-service      "java -jar /app/estu..." About a minute ago Up 21 seconds  0.0.0.0:4101->4101/tcp              app-estudiantes-service-1
32e6d2e99fdd   postgres:15                        "docker-entrypoint.s..." About a minute ago Up 29 seconds  0.0.0.0:5434->5432/tcp              matriculas-db
26ee754640e5   postgres:15                        "docker-entrypoint.s..." About a minute ago Up 27 seconds  0.0.0.0:5432->5432/tcp              cursos-db
b4497817bb22   postgres:15                        "docker-entrypoint.s..." About a minute ago Up 26 seconds  0.0.0.0:5433->5432/tcp              estudiantes-db
2f7e110946ba   shirley24/react-app               "/docker-entrypoint..." 13 minutes ago  Up 13 minutes  0.0.0.0:80->80/tcp                  app-react-app-1
PS C:\Users\shirl\Documents\Universidad\8vo Semestre\Aplicaciones Distribuidas\Segundo Parcial\app>
```

4. Pruebas del Sistema

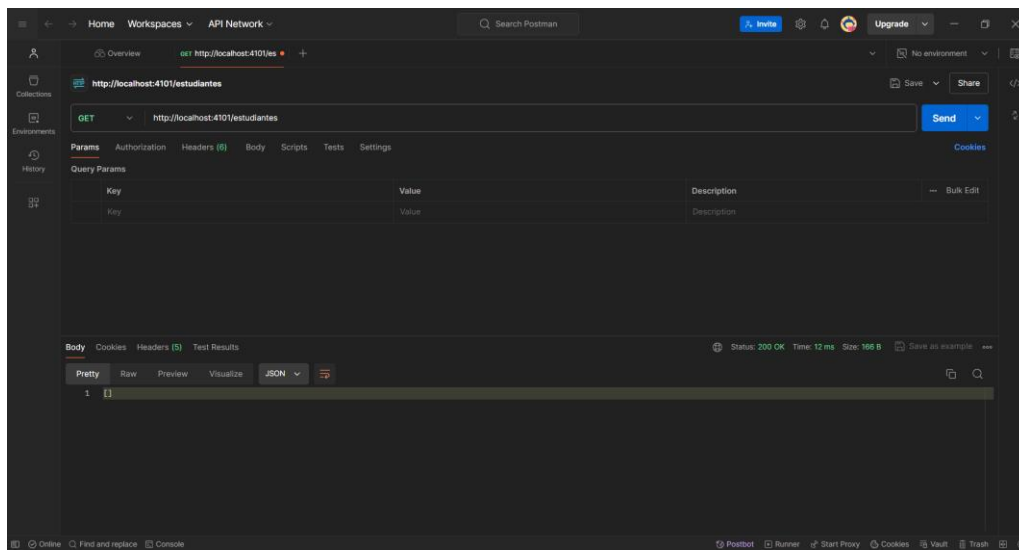
i Paso 1: Verificación de Conectividad entre Microservicios

Las pruebas con Postman verificaron que los microservicios respondieron correctamente a las solicitudes API. Se probaron las operaciones CRUD para cada microservicio utilizando las siguientes rutas:

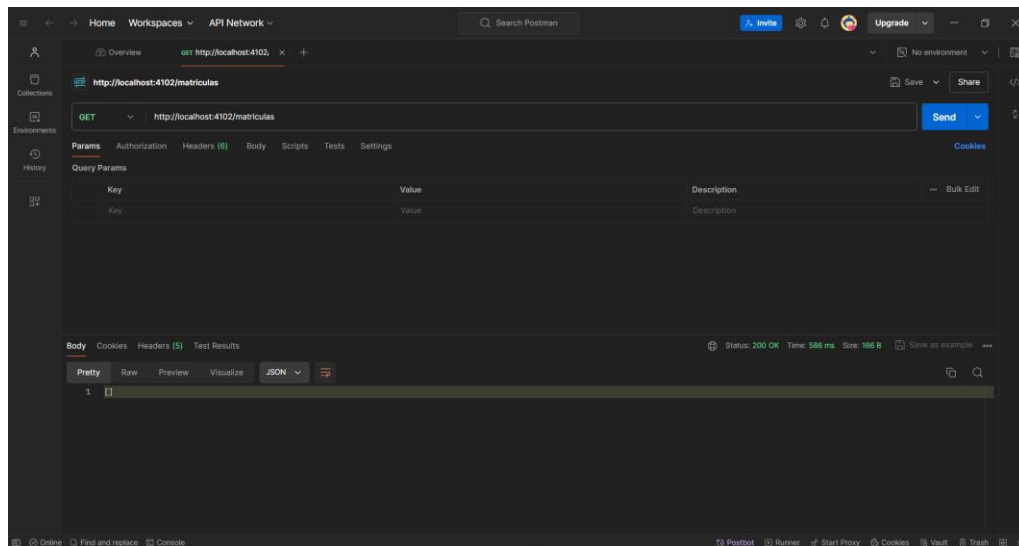
a. **Cursos:** <http://localhost:4100/cursos>



b. **Estudiantes:** <http://localhost:4101/estudiantes>

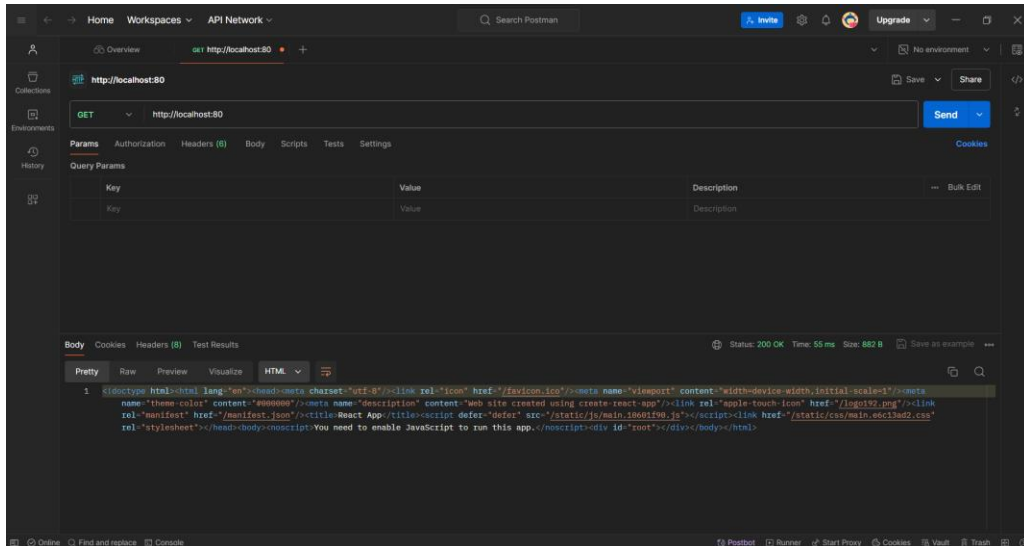


c. **Matrículas:** <http://localhost:4102/matriculas>



i Paso 2: Pruebas de la Aplicación React

Se accedió a la aplicación frontend en `http://localhost`, donde se realizaron pruebas de funcionalidad utilizando la interfaz gráfica. Se verificó que las operaciones CRUD se reflejaran correctamente en la base de datos.



i Paso 3: Pruebas de Integración

Se realizaron pruebas de integración para asegurarse de que los microservicios y la aplicación frontend funcionaran de manera coherente. Se probó un flujo completo donde un estudiante se matricula en un curso, y se verificó que todos los datos se gestionaran correctamente.

RESULTADOS

1. Verificación de Conectividad entre Microservicios

i Las pruebas realizadas con Postman verificaron que los microservicios están expuestos en los puertos correctos y responden adecuadamente a las solicitudes API. Se probaron las operaciones CRUD básicas (crear, leer, actualizar, eliminar) en cada microservicio.

2. Pruebas de la Aplicación React

i Se verificó que la aplicación React interactúa correctamente con los microservicios. Las operaciones CRUD fueron probadas a través de la interfaz gráfica, y los datos reflejados en la base de datos correspondieron correctamente a las acciones realizadas en la interfaz.

3. Pruebas de Integración

i Las pruebas de integración confirmaron que los microservicios y la aplicación React funcionan de manera coherente e integrada. Se realizaron operaciones de matrícula de estudiantes en cursos y se verificó que los datos fueran almacenados y recuperados correctamente.

PROBLEMAS ENCONTRADOS Y SOLUCIONES

i Durante las pruebas, se encontró un problema relacionado con la configuración incorrecta de las rutas en los microservicios, lo que causó errores 404. Esto se resolvió revisando y corrigiendo las rutas en los controladores de los microservicios.

i Otro problema detectado fue la necesidad de reiniciar los contenedores después de aplicar configuraciones en el archivo docker-compose.yml para asegurar que los cambios surtieran efecto.

REFLEXIÓN: VENTAJAS Y DESAFÍOS DE DOCKER

1. Ventajas

i Docker permite aislar y escalar cada microservicio de manera independiente, lo que facilita el mantenimiento y la actualización del sistema.

i Las imágenes Docker pueden ser ejecutadas en cualquier entorno que soporte Docker, lo que simplifica el proceso de despliegue.

i Docker Compose permite configurar y desplegar todos los servicios del sistema de manera coherente y repetible.

2. Desafíos

i La configuración inicial de Dockerfiles y Docker Compose puede ser compleja, especialmente para sistemas con múltiples servicios interdependientes.

i La gestión de volúmenes para la persistencia de datos en bases de datos puede requerir una planificación cuidadosa para evitar pérdida de datos.

CONCLUSIONES

i En base al objetivo planteado, se **implementó** un sistema basado en microservicios y una aplicación frontend utilizando Docker, **mediante** la creación, configuración y despliegue de contenedores Docker para cada componente del sistema. Durante este proceso, se logró asegurar una correcta conectividad y funcionalidad integrada entre los servicios de cursos, estudiantes y matrículas, así como con la aplicación frontend.

i Las pruebas realizadas, tanto automatizadas como manuales, permitieron verificar que todos los componentes del sistema funcionaran de manera coordinada, cumpliendo con los requerimientos de operación y comunicación esperados. Aunque se presentaron algunos desafíos, como errores en la configuración inicial de rutas y la necesidad de reiniciar los contenedores para aplicar correctamente ciertas configuraciones, estos problemas fueron solucionados eficazmente, demostrando la flexibilidad y robustez que Docker ofrece en el despliegue de aplicaciones complejas.

i En conclusión, el uso de Docker para desplegar y gestionar un sistema basado en microservicios y una aplicación frontend no solo facilita la portabilidad y escalabilidad del sistema, sino que también permite mantener una consistencia en la configuración y operación de los servicios, cumpliendo así con el objetivo de asegurar una integración exitosa y el correcto funcionamiento del sistema en su conjunto.

ENLACE A IMÁGENES EN DOCKER HUB

- a. **Cursos-service:** <https://hub.docker.com/repository/docker/shirley24/cursos-service/general>
- b. **Estudiantes-service:** <https://hub.docker.com/repository/docker/shirley24/estudiantes-service/general>
- c. **Matriculas-service:** <https://hub.docker.com/repository/docker/shirley24/matriculas-service/general>
- d. **React-app:** <https://hub.docker.com/repository/docker/shirley24/react-app/general>