

# Desplegando una Aplicación Node Dockerizada para Portales de Clientes

**Tiempo estimado necesario:** 30 minutos

En este laboratorio, configurarás y desplegarás una aplicación de Node.js acoplada con MongoDB en Docker. Tu objetivo es replicar el repositorio del portal de clientes utilizado en ejercicios anteriores y comenzar el despliegue de la aplicación del portal de clientes usando Docker.

## Objetivo de Aprendizaje:

Después de completar este laboratorio, podrás:

- Clonar una aplicación Node.js del portal de clientes desde un repositorio de GitHub
- Completar la configuración de Dockerfile y docker-compose.yml
- Desplegar la aplicación Node.js en Docker

## Acerca de Skills Network Cloud IDE

Skills Network Cloud IDE (basado en Theia y Docker) proporciona un entorno para laboratorios prácticos relacionados con cursos y proyectos. Theia es un IDE (Entorno de Desarrollo Integrado) de código abierto.

## Aviso Importante sobre este entorno de laboratorio

Ten en cuenta que las sesiones para este entorno de laboratorio no se persisten. Cada vez que te conectes a este laboratorio, se creará un nuevo entorno para ti. Cualquier dato que hayas guardado en la sesión anterior se perderá. Planea completar estos laboratorios en una sola sesión, para evitar perder tus datos.

## Tarea 1: Clonar una aplicación de portal de clientes Node.js desde un repositorio de GitHub

1. Abre una ventana de terminal utilizando el menú en el editor: Terminal > Nueva Terminal.
2. Cambia a tu carpeta de proyecto, si aún no estás en la carpeta del proyecto.

```
cd /home/project
```

3. En la terminal, clona el repositorio de git que contiene el código inicial.

```
git clone https://github.com/ibm-developer-skills-network/abltc-backend-nodejs-customerportal.git
```

4. Cambia al nuevo directorio que acabas de clonar.

```
cd abltc-backend-nodejs-customerportal
```

5. En la terminal, ejecuta el siguiente comando para instalar los paquetes de nodo requeridos.

```
npm install
```

## Tarea 2: Configuración de Dockerfile y docker-compose.yml

1. Crea un archivo Dockerfile utilizando el comando a continuación.

```
touch Dockerfile
```

## 2. Pega el código a continuación en Dockerfile.

```
FROM node:18.12.1-bullseye-slim
# Set the working directory in the container
WORKDIR /app
# Copy package.json and package-lock.json to the working directory
COPY package*.json ./
# Install app dependencies
RUN npm install
# Copy necessary files
COPY . .
# Expose port 3000 (adjust the port based on your application)
EXPOSE 3000
# CMD to run the application
CMD [ "node", "customer_app.js" ]
```

## 3. Crea un archivo docker-compose.yml usando el siguiente comando.

```
touch docker-compose.yml
```

## 4. Pega el siguiente código en docker-compose.yml

```
version: '3'
services:
  mongodb:
    image: mongo
    ports:
      - "27017:27017"
    networks:
      - app-network

  app:
    image: customerapp
    ports:
      - "3000:3000"
    depends_on:
      - mongodb
    networks:
      - app-network

networks:
  app-network:
    driver: bridge
```

# Tarea 3: Desplegar la aplicación Node.js en Docker

1. Ejecuta el siguiente comando para construir la aplicación Docker

```
docker build . -t customerapp
```

2. Se ha creado el archivo docker-compose.yml para ejecutar dos contenedores, uno para Mongo y el otro para la aplicación Node. Ejecuta el siguiente comando para iniciar el servidor:

```
docker-compose up
```

3. Haga clic en el botón de abajo para iniciar la aplicación.

Portal del Cliente

► Haga clic aquí si el botón anterior no funciona

## Authors

Pallavi Rai

© IBM Corporation. Todos los derechos reservados.