DEVOPS

Desarrollo -> Testing Desarrollador->Generar Artifacto -> Pruebas en QA->Pruebas de Aceptación->Produccion

Crear proyecto

npm init -> inicia el package.json del proyecto

npm install express supertest --save -> instalando librerias en el proyecto –save para que se guarde en el package.json

npm install -g mocha -> intalando mocha

mkdir test -> creando directorio donde se alojarán las pruebas

touch app.js server.js Dockerfile .dockerignore -> creando archivos para armar la imagen y para programar el servicio

si no se instaló mocha puede correr la prueba con npm test

O

npm run test

comando docker

**Dockerfile** este file es un manifiesto de comandos que se deben correr para construir nuestro environment correctamente cada vez.

**-El FROM.** Designa una imagen que queremos usar como base.  
**-COPY.** Mueve files desde nuestra máquina local al container. Con esto podemos mover nuestro código al container.  
**-RUN.** Permite correr comandos arbitrarios.  
**-CMD.** Le indica a **Docker** que éste es el default command con el que se va a correr el container.  
**-Add.** Copia un archivo desde el host en el contenedor.  
**-CMD.** Configura comandos por defecto para ser ejecutados o se pasa al punto de entrada.  
**-ENTRYPOINT.** Ajusta el punto de entrada por defecto de la aplicación desde el contenedor.  
**-ENV.** Inicializa variables de entorno (por ejemplo, “clave=valor”)  
**-EXPOSE.** Expone un puerto al exterior.  
**-MAINTAINER.** Establece los datos de autor/propietario del archivo **Dockerfile**  
**-USER.** Establece el usuario para ejecutar los contenedores de la imagen.  
**-VOLUMEN.** Monta un directorio desde el host al contenedor.  
**-WORKDIR.** Establece el directorio para las directivas de **CMD** que se ejecutarán.

Se creo .dockerignore para que no traslades los archivos que no se necesitan

node-modules

.git

Se creo dockerfile para crear la imagen

FROM -> de que imagen va a crear nuestra imagen

Expose -> el puerto de nuestro aplicativo

Workdir -> la dirección donde va a estar nuestro contenedor

Add -> añade archivos de nuestro aplicativo al contenedor

Run -> para correr algún comando

CMD -> le indica a nuestro contenedor que siempre corra un comando

FROM node:12.16.3

EXPOSE 3000

WORKDIR /devops-node-app

ADD package.json /devops-node-app

RUN npm install

ADD . /devops-node-app

CMD ["node", "server"]

Ejecutando comandos docker del aplicativo

Este comando toma el dockerfile y construye nuestra imagen

docker build -t docker-devops-app .

ponemos a correr nuestro servicio o aplicativo en nuestra imagen

docker run -**p** 3000:3000 docker-devops-app

para correr las pruebas en docker

docker run docker-devops-app npm test

se trabajar con la herramienta circle ci para la integracion continua