

1)

FROM: Especifica la imagen base a partir de la cual se creará la nueva imagen. Todas las demás instrucciones se basarán en esta imagen. Ejemplo: Uso FROM ubuntu para construir una imagen basada en el sistema operativo Ubuntu.

RUN: Se utiliza para ejecutar comandos en una nueva capa sobre la imagen actual. Ejemplo: Uso RUN apt-get update para actualizar los paquetes del sistema en una imagen basada en Ubuntu.

ADD: Copia archivos o directorios desde el sistema de archivos local al sistema de archivos de la imagen. Puede realizar copias de archivos locales o incluso descargar archivos desde una URL y copiarlos a la imagen.

COPY: Similar a ADD, pero se utiliza específicamente para copiar archivos y directorios desde el sistema de archivos local al sistema de archivos de la imagen.

EXPOSE: Indica los puertos en los que la aplicación en ejecución escuchará en el contenedor. Esto no publica el puerto ni lo hace accesible desde fuera del contenedor, pero es útil para documentar qué puertos debe exponer la aplicación.

CMD: Define el comando predeterminado que se ejecutará cuando se inicie un contenedor basado en la imagen.

ENTRYPOINT: También define un comando predeterminado, pero a diferencia de CMD, los argumentos proporcionados en la línea de comandos no reemplazarán el comando especificado en ENTRYPOINT. Se pueden proporcionar argumentos adicionales.

2)

```

=> [internal] load build definition from dockerfile                                0.1s
=> => transferring dockerfile: 508B                                              0.0s
=> [internal] load .dockerignore                                                 0.1s
=> => transferring context: 2B                                                  0.0s
=> [internal] load metadata for mcr.microsoft.com/dotnet/sdk:7.0              2.4s
=> [build 1/8] FROM mcr.microsoft.com/dotnet/sdk:7.0@sha256:d0755fbf1ed34aecdd79c127fd1dd01b80587 23.3s
=> => resolve mcr.microsoft.com/dotnet/sdk:7.0@sha256:d0755fbf1ed34aecdd79c127fd1dd01b80587acdb8ff 0.1s
=> => sha256:d0755fbf1ed34aecdd79c127fd1dd01b80587acdb8ff50e5a68d1adf8b076922b 1.79kB / 1.79kB 0.0s
=> => sha256:fbf7eed1efcc4cd2571e40f0c34949a09474f4a96594b0501b4e24b7f0767bcb 2.81kB / 2.81kB 0.0s
=> => sha256:bd36eac352dbb2d27a63f4714b9063fc8c7e9c615e430006dbeb7e3448cd5780 5.28kB / 5.28kB 0.0s
=> => sha256:e67fdae3559346105027c63e7fb032bba57e62b1fe9f2da23e6fd9b56384e00b 31.42MB / 31.42MB 1.3s
=> => sha256:0ab66724116f089079aab4d1403ce98586595f13ca4b4163005ec3733a5dab94 14.97MB / 14.97MB 2.7s
=> => sha256:14cdddebb1bc93dd113768cf39770236e666888c5d08fc5a554d2e47a0fe930e 32.46MB / 32.46MB 3.8s
=> => sha256:5e265b51b431d80b30bfff2e2f867af78d559418b418ea7b3edffaead2221b244 156B / 156B 1.6s
=> => extracting sha256:e67fdae3559346105027c63e7fb032bba57e62b1fe9f2da23e6fd9b56384e00b 7.8s
=> => sha256:3bd8a6efdfc5b92b1a9760092efa9eb1a4f50a0374789af843c6842acaf848b8f 10.12MB / 10.12MB 2.6s
=> => sha256:9a6d473c86f69f795e0ff2b6014ed32f2c9b35e743068b430a4adf1c1448176b 25.38MB / 25.38MB 8.8s
=> => sha256:5464f27bea88b4d6d19d0c0e15a62a967f65d74b9753611d0a174526bc08b1da 180.99MB / 180.99MB 8.7s
=> => sha256:3cc0e9253374a2ce8dd07c45550ebb8a207cc9898aa397219369e970962f7ab6 13.98MB / 13.98MB 7.1s
=> => extracting sha256:0ab66724116f089079aab4d1403ce98586595f13ca4b4163005ec3733a5dab94 1.5s
=> => extracting sha256:14cdddebb1bc93dd113768cf39770236e666888c5d08fc5a554d2e47a0fe930e 2.1s
=> => extracting sha256:5e265b51b431d80b30bfff2e2f867af78d559418b418ea7b3edffaead2221b244 0.0s
=> => extracting sha256:3bd8a6efdfc5b92b1a9760092efa9eb1a4f50a0374789af843c6842acaf848b8f 0.7s
=> => extracting sha256:9a6d473c86f69f795e0ff2b6014ed32f2c9b35e743068b430a4adf1c1448176b 3.3s
=> => extracting sha256:5464f27bea88b4d6d19d0c0e15a62a967f65d74b9753611d0a174526bc08b1da 4.6s
=> => extracting sha256:3cc0e9253374a2ce8dd07c45550ebb8a207cc9898aa397219369e970962f7ab6 0.4s
=> [internal] load build context                                                0.2s
=> => transferring context: 4.78MB                                              0.2s
=> [build 2/8] WORKDIR /src                                                    0.1s
=> [build 3/8] COPY [MiProyectoWebAPI.csproj, .]                             0.0s
=> [build 4/8] RUN dotnet restore "./MiProyectoWebAPI.csproj"                 11.8s
=> [build 5/8] COPY . .                                                        0.1s
=> [build 6/8] WORKDIR /src/.                                                  0.0s
=> [build 7/8] RUN dotnet build "MiProyectoWebAPI.csproj" -c Release -o /app/build 9.4s
=> [build 8/8] RUN dotnet publish "MiProyectoWebAPI.csproj" -c Release -o /app/publish /p:UseAppH 2.9s
=> exporting to image                                                         0.4s
=> => exporting layers                                                         0.4s
=> => writing image sha256:e790483ad63c23427718f2a0c4d4ba0f354489f9dc1e07750581e2b4bda175d93 0.0s
=> => naming to docker.io/library/miprojectowebapi                           0.0s

```

```

80 -lt --rm miprojectowebapi
[info: Microsoft.Hosting.Lifetime[14]
Now listening on: http://localhost:5000
[info: Microsoft.Hosting.Lifetime[0]
Application started. Press Ctrl+C to shut down.
[info: Microsoft.Hosting.Lifetime[0]
Hosting environment: Production
[info: Microsoft.Hosting.Lifetime[0]
Content root path: /src

```

```

root@23cf3b59a2ff:/# cd app
root@23cf3b59a2ff:/app# ls
Microsoft.AspNetCore.OpenApi.dll  Swashbuckle.AspNetCore.Swagger.dll
Microsoft.OpenApi.dll             Swashbuckle.AspNetCore.SwaggerGen.dll
SimpleWebAPI.deps.json            Swashbuckle.AspNetCore.SwaggerUI.dll
SimpleWebAPI.dll                  appsettings.Development.json
SimpleWebAPI.pdb                  appsettings.json
SimpleWebAPI.runtimeconfig.json   web.config
root@23cf3b59a2ff:/app# pwd
/app
root@23cf3b59a2ff:/app#

```

Las diferencias entre el Dockerfile anterior y el nuevo son principalmente organizativas y de claridad. El nuevo Dockerfile mantiene las mismas etapas básicas pero las separa de manera más clara y legible. Las etapas base, build, publish y final tienen nombres más descriptivos, lo que hace que el Dockerfile sea más fácil de entender y mantener.

```

root@20c77a72293d:/# cd app/
root@20c77a72293d:/app# ls
MiProyectoWebAPI.deps.json  Microsoft.OpenApi.dll  appsettings.Development.json
MiProyectoWebAPI.dll        Newtonsoft.Json.dll    appsettings.json
MiProyectoWebAPI.pdb        Swashbuckle.AspNetCore.Swagger.dll  web.config
MiProyectoWebAPI.runtimeconfig.json  Swashbuckle.AspNetCore.SwaggerGen.dll
Microsoft.AspNetCore.OpenApi.dll      Swashbuckle.AspNetCore.SwaggerUI.dll
root@20c77a72293d:/app#

```

```

PS C:\Users\gusta\OneDrive\Escritorio\UCC\Cuarto Año\IngenieriaSoftware 3\trabajo-practico-06nodejs-docker> npx create-react-app my-app

Creating a new React app in C:\Users\gusta\OneDrive\Escritorio\UCC\Cuarto Año\IngenieriaSoftware 3\trabajo-practico-06nodejs-docker\my-app.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...

added 1459 packages in 1m

242 packages are looking for funding
  run 'npm fund' for details

Initialized a git repository.

Installing template dependencies using npm...

added 69 packages, and changed 1 package in 13s

246 packages are looking for funding
  run 'npm fund' for details
Removing template package using npm...

removed 1 package, and audited 1528 packages in 7s

246 packages are looking for funding
  run 'npm fund' for details

8 vulnerabilities (2 moderate, 6 high)

To address all issues (including breaking changes), run:
  npm audit fix --force

Run 'npm audit' for details.

Created git commit.

Success! Created my-app at C:\Users\gusta\OneDrive\Escritorio\UCC\Cuarto Año\IngenieriaSoftware 3\trabajo-practico-06nodejs-docker\my-app
Inside that directory, you can run several commands:

  npm start
    Starts the development server.

  npm run build
    Bundles the app into static files for production.

  npm test
    Starts the test runner.

  npm run eject
    Removes this tool and copies build dependencies, configuration files
    and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

  cd my-app
  npm start

Happy hacking!
PS C:\Users\gusta\OneDrive\Escritorio\UCC\Cuarto Año\IngenieriaSoftware 3\trabajo-practico-06nodejs-docker>

```

```

Dockerfile X
C: > Users > gusta > OneDrive > Escritorio > UCC > Cuarto Año
1  # Etapa 1: Imagen de construcción
2  FROM node:13.12.0-alpine as build
3  WORKDIR /app
4  COPY package*.json ./
5  RUN npm install
6  COPY . .
7  RUN npm run build
8
9  # Etapa 2: Imagen de producción
10 FROM node:13.12.0-alpine
11 WORKDIR /app
12 COPY --from=build /app .
13 EXPOSE 3000
14 CMD ["npm", "start"]
15
16

```


```

PS C:\Users\gusta\OneDrive\Escritorio\UCC\Cuarto Año\IngenieriaSoftware 3\trabajo-practico-06nodejs-docker\my-app> docker build -t prueba .
[*] Building 449.5s (13/13) FINISHED
=> [internal] load .dockerignore
=> transferring context: 2B
=> [internal] load build definition from Dockerfile
=> transferring dockerfile: 317B
=> [internal] load metadata for docker.io/library/node:13.12.0-alpine
=> [auth] library/node:pull token for registry-1.docker.io
=> [build 1/6] FROM docker.io/library/node:13.12.0-alpine@sha256:cc85e728fab3827ada20a181ba280cae1f8b625f256e2c86b9094d9bfe834766
=> resolve docker.io/library/node:13.12.0-alpine@sha256:cc85e728fab3827ada20a181ba280cae1f8b625f256e2c86b9094d9bfe834766
=> sha256:cc85e728fab3827ada20a181ba280cae1f8b625f256e2c86b9094d9bfe834766 1.19kB / 1.19kB
=> sha256:ed06828d8fb6f4711e0a6f58c9f147fb2596399866319e1bb3b0a52393c5615f 1.16kB / 1.16kB
=> sha256:4033a3d6c5f5d658963b7c16c4d992d7f7f68b6f402581245f9e195401e1a1fc1 6.77kB / 6.77kB
=> sha256:c57f2c59b93778192e60e0e213949630f9ad30324736bbb8a71e12adf8a16d46 2.24MB / 2.24MB
=> sha256:aad63a9339440e7c3e1fff2b988991b9bfb81280042fa7f39a5e327023856819 2.80MB / 2.80MB
=> sha256:a08bd932288e2de29cd2b7e0bab954325913d146401effb482ffff3d8775aaab 35.29MB / 35.29MB
=> sha256:f3446470f297e51c1e27f90f7ae9a94flee7b6c8e529aec83aa1a04ad70531c0 284B / 284B
=> extracting sha256:aad63a9339440e7c3e1fff2b988991b9bfb81280042fa7f39a5e327023856819
=> extracting sha256:a08bd932288e2de29cd2b7e0bab954325913d146401effb482ffff3d8775aaab
=> extracting sha256:c57f2c59b93778192e60e0e213949630f9ad30324736bbb8a71e12adf8a16d46
=> extracting sha256:f3446470f297e51c1e27f90f7ae9a94flee7b6c8e529aec83aa1a04ad70531c0
=> [internal] load build context
=> transferring context: 248.35MB
=> [build 2/6] WORKDIR /app
=> [build 3/6] COPY package*.json ./
=> [build 4/6] RUN npm install
=> [build 5/6] COPY . .
=> [build 6/6] RUN npm run build
=> [stage-1 3/3] COPY --from=build /app .
=> exporting to image
=> exporting layers
=> writing image sha256:3dd0ad2157410a81a0109d94a337c7b99d987a32ae9df0326d4870ec638b05b5
=> naming to docker.io/library/prueba

What's Next?
  View summary of image vulnerabilities and recommendations → docker scout quickview

PS C:\Users\gusta\OneDrive\Escritorio\UCC\Cuarto Año\IngenieriaSoftware 3\trabajo-practico-06nodejs-docker\my-app> docker run -d -p 3000:3000 --name test-no
de prueba
db6b5fe51d348e88be4eb41d5435ec1b7a3573095eb01f1975b00cb158f12c78

```



Edit `src/App.js` and save to reload.

[Learn React](#)

```

PS C:\Users\gusta\OneDrive\Escritorio\UCC\Cuarto Año\IngenieriaSoftware 3\trabajo-practico-06nodejs-docker\my-app> docker tag prueba gustavoschonfeld14/prueba:latest
PS C:\Users\gusta\OneDrive\Escritorio\UCC\Cuarto Año\IngenieriaSoftware 3\trabajo-practico-06nodejs-docker\my-app> docker push gustavoschonfeld/prueba:latest
The push refers to repository [docker.io/gustavoschonfeld/prueba]
An image does not exist locally with the tag: gustavoschonfeld/prueba
PS C:\Users\gusta\OneDrive\Escritorio\UCC\Cuarto Año\IngenieriaSoftware 3\trabajo-practico-06nodejs-docker\my-app> docker push gustavoschonfeld14/prueba:latest
The push refers to repository [docker.io/gustavoschonfeld14/prueba]
5c75db2d97a0: Pushed
627a2dc81a49: Pushed
65d358b7de11: Pushed
f97384e8ccbc: Pushed
d56e5e720148: Pushed
beee9f30bc1f: Pushed
latest: digest: sha256:46d51af4715a80fb5371736429d0e64751017ede1752b2e37a07d9a2dad0065c size: 1577
PS C:\Users\gusta\OneDrive\Escritorio\UCC\Cuarto Año\IngenieriaSoftware 3\trabajo-practico-06nodejs-docker\my-app>

```