

Tarea 1

- you should use the 'node' official image, with the alpine 6.x branch
- this app listens on port 3000, but the container should launch on port 80
so it will respond to <http://localhost:80> on your computer
- then it should use alpine package manager to install tini
- then it should create directory /usr/src/app for app files
- Node uses a "package manager", so it needs to copy in package.json file
- then it needs to run 'npm install' to install dependencies from that file
- to keep it clean and small, run 'npm cache clean --force' after above
- then it needs to copy in all files from current directory
- then it needs to start container with command '/sbin/tini -- node ./bin/www'

Solución

Creación archivo Dockerfile

```
FROM node:6-alpine
WORKDIR /usr/src/app
RUN apk add --no-cache tini
COPY package*.json ./
RUN npm install
RUN npm cache clean --force
COPY . .
EXPOSE 3000
ENTRYPOINT ["/sbin/tini", "--"]
CMD [ "node", "./bin/www" ]
```

Construir imagen:

Estando en la raíz del proyecto, se ejecuta el comando **docker build -t nodetarea .**

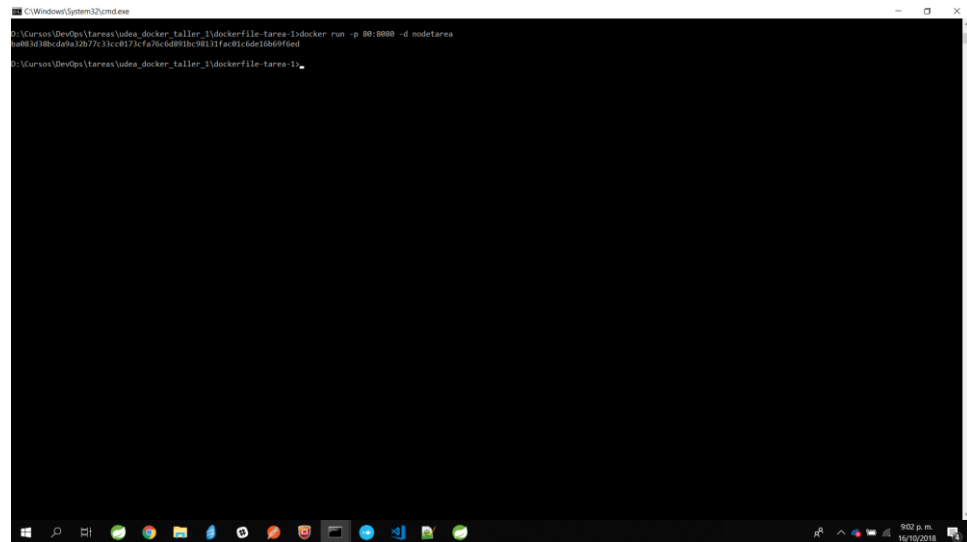
Ejecución contenedor:

Creada la imagen se ejecuta la instrucción: **docker run -p 80:3000 -d nodetarea**

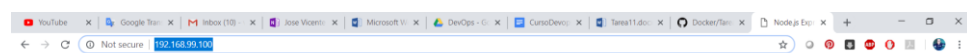
Imágenes demostrativas:

Localmente se ejecuta con la ip del docker o con localhost, para el caso se ejecuta

<http://192.168.99.100/>



```
C:\Windows\System32\cmd.exe
C:\Users\DevOps\Documents\tarea_docker_taller_1\dockerfile\tarea-1>docker run -p 80:3000 -d nodetarea
5a8b3d8d6a2b7f33cc077cfa7ac6d891bc9b131f6c8f6e16d6d1f6d
C:\Users\DevOps\Documents\tarea_docker_taller_1\dockerfile\tarea-1>
```



Node.js Express App

It Worked! You Deserve The Captain's Applause

