

GRUPO:
FECHA:

Techforge
06 / 06 / 2024

TEMA:

Manual de Configuración

Manual de Configuración

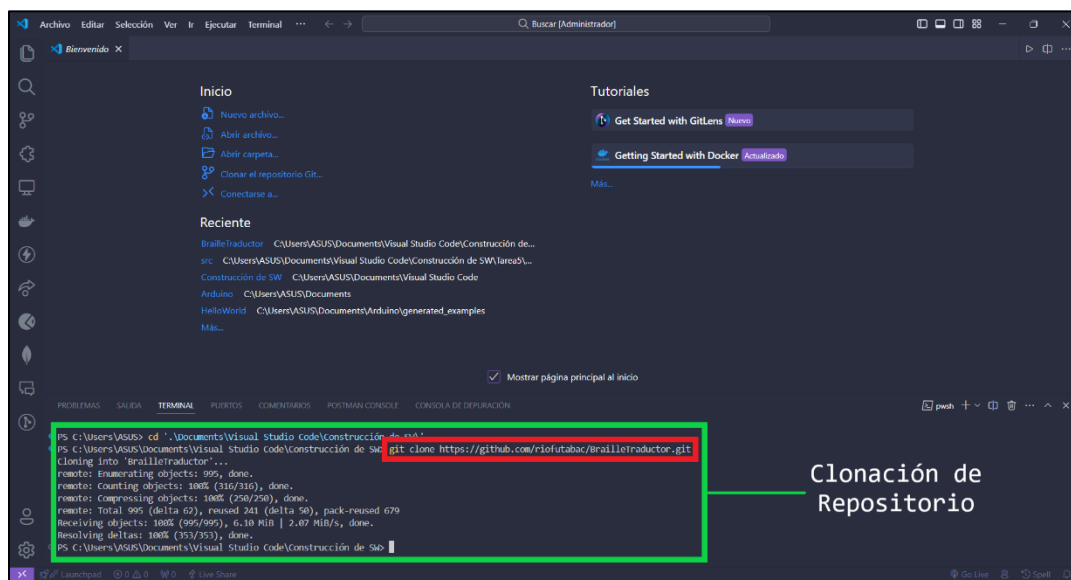
Requisitos Previos

- Tener instalado Visual Studio Code.
- Poseer una cuenta en Docker.
- Tener instalado Docker Desktop en su ordenador e iniciado sesión con su cuenta.



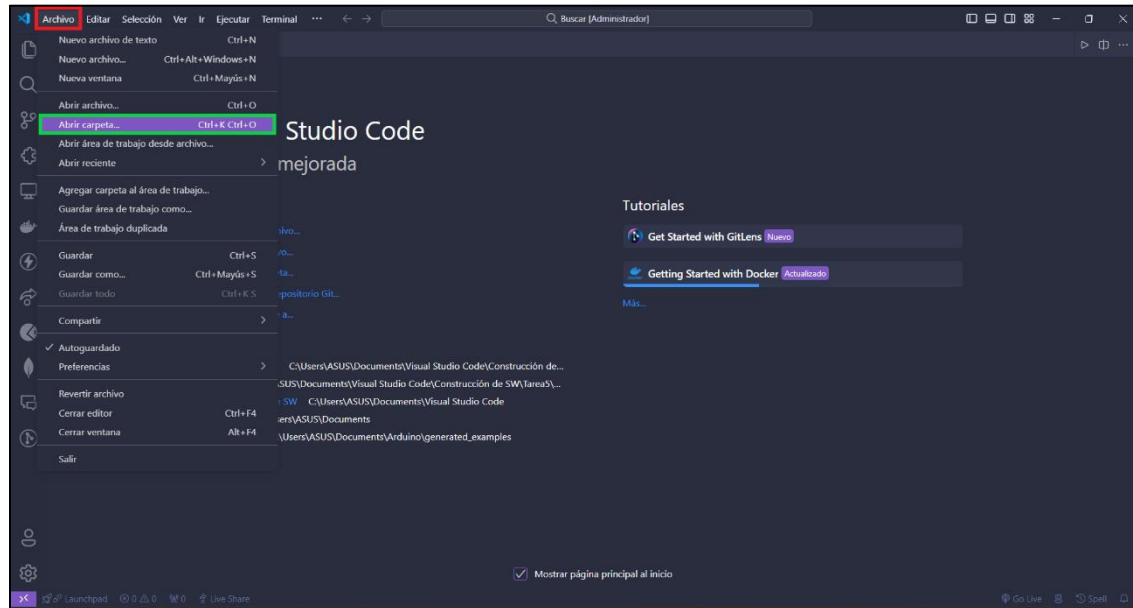
Configuración

Para utilizar el aplicativo WEB, se debe abrir Visual Studio Code y clonar el repositorio desde el terminal con el comando `git clone https://github.com/riofutabac/BrailleTraductor.git` en un directorio de su elección.

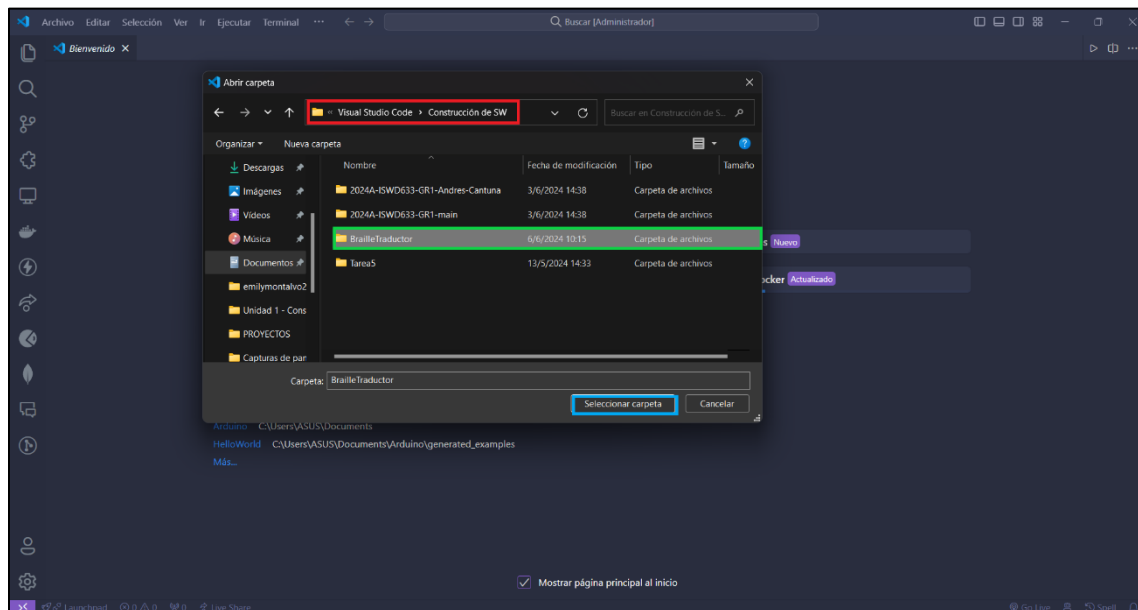


ISWD633 CONSTRUCCIÓN Y EVOLUCIÓN DE SOFTWARE

Clic en Archivo, abrir Carpeta.

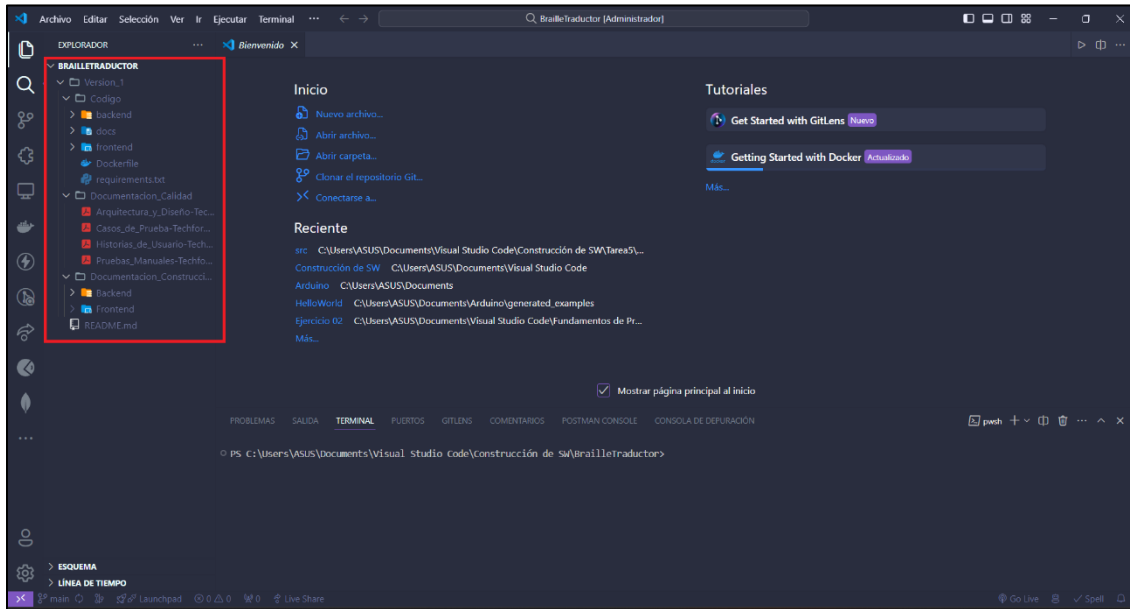


Dirígete al directorio donde clonaste el repositorio y selecciona la carpeta BrailleTraductor.



ISWD633 CONSTRUCCIÓN Y EVOLUCIÓN DE SOFTWARE

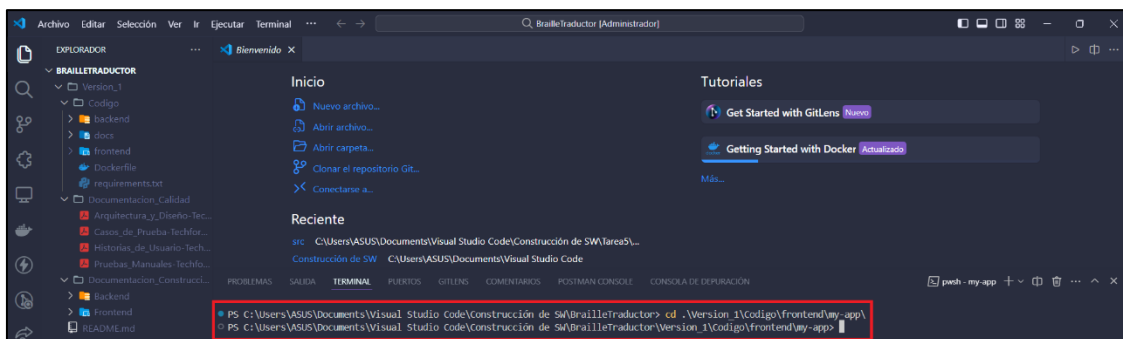
En el explorador aparecerá todo el contenido relacionado a la aplicación (archivos de código y documentación).



Ahora se procederá a levantarlos servicios para backend o frontend. Para ello, inicia Docker Desktop.

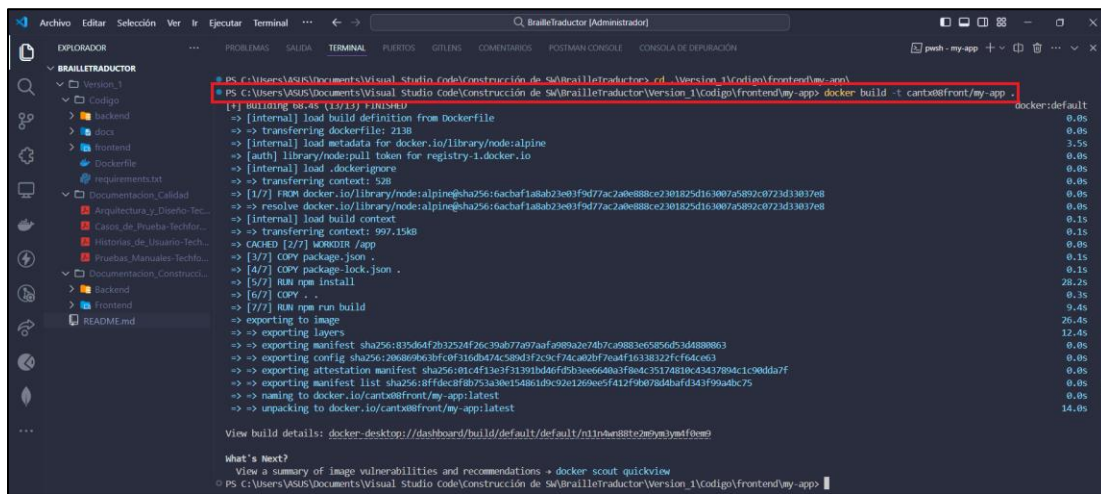
Frontend

En Visual Studio Code, abre el terminal e ingresa a la carpeta frontend/my-app mediante el comando `cd .\Version_1\Codigo\frontend\my-app\`



Ahora, con el comando `docker build -t <tu-usuario>front/my-app .` construye una imagen de Docker identificado con `<tu-usuario>front/my-app`. Asegúrate de escribir el usuario con el que has iniciado sesión en Docker.

ISWD633 CONSTRUCCIÓN Y EVOLUCIÓN DE SOFTWARE

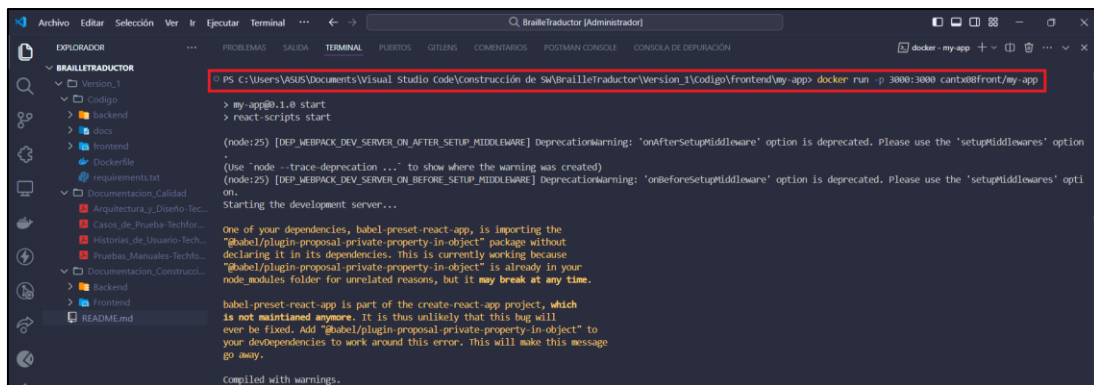


```
PS C:\Users\ASUS\Documents\Visual Studio Code\Construcción de SW\BrailleTraductor\Version_1\Codigo\frontend\my-app> docker build -t cantx08front/my-app .
[+] Building 0.0s (12/13) FINISHED
=> [internal] load build definition from Dockerfile
=> transferring Dockerfile: 213B
=> [internal] load metadata for docker.io/library/node:alpine
=> [auth] library/node:pull token for registry-1.docker.io
=> [internal] load .dockerignore
=> transferring context: 52B
=> [1/7] FROM docker.io/library/node:alpine@sha256:6acba1a8ab23e03f9d77ac2a0e888ce2301825d163007a5892c0723d33037e8
=> resolve docker.io/library/node:alpine@sha256:6acba1a8ab23e03f9d77ac2a0e888ce2301825d163007a5892c0723d33037e8
=> [internal] load build context
=> transferring context: 997.15kB
=> CACHED [2/7] WORKDIR /app
=> [3/7] COPY package.json .
=> [4/7] COPY package-lock.json .
=> [5/7] RUN npm install
=> [6/7] COPY . .
=> [7/7] RUN npm run build
=> exporting to image
=> exporting layers
=> exporting manifest sha256:1835d64f2b32524f26c39ab77a07aaf9080a2e74b7ca0883e65856d53d4880863
=> exporting config sha256:206889b03fc0f116b0474c509d3f2c9c74ca02b770a4f16338322fc64ce03
=> exporting attestation manifest sha256:81c4f13e3f31391bd46f5b3ee6640a3f8e4c35174810c43437894c1c90dda7f
=> exporting manifest list sha256:8ffdec8f8b75a30e154861d9c92e1269e5f412f9b078d4baf343f99a0bc75
=> naming to docker.io/cantx08front/my-app:latest
=> unpacking to docker.io/cantx08front/my-app:latest

View build details: docker-desktop://dashboard/build/default/default/n11n8w88te2mymymf0wep

What's Next?
View a summary of image vulnerabilities and recommendations + docker scout quickview
PS C:\Users\ASUS\Documents\Visual Studio Code\Construcción de SW\BrailleTraductor\Version_1\Codigo\frontend\my-app>
```

Luego, inicia el contenedor basado en la imagen <tu-usuario>front/my-app a través del comando `docker run -p 3000:3000 <tu-usuario>front/my-app`



```
PS C:\Users\ASUS\Documents\Visual Studio Code\Construcción de SW\BrailleTraductor\Version_1\Codigo\frontend\my-app> docker run -p 3000:3000 cantx08front/my-app
> my-app@0.1.0 start
> react-scripts start

(node:25) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' option
(Use 'node --trace-deprecation ...' to show where the warning was created)
(node:25) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' option.
Starting the development server...

One of your dependencies, babel-preset-react-app, is importing the
"babel/plugin-proposal-private-property-in-object" package without
declaring it in its dependencies. This is currently working because
"babel/plugin-proposal-private-property-in-object" is already in your
node_modules folder for unrelated reasons, but it may break at any time.

babel-preset-react-app is part of the create-react-app project, which
is not maintained anymore. It is thus unlikely that this bug will
ever be fixed. Add "babel/plugin-proposal-private-property-in-object" to
your devDependencies to work around this error. This will make this message
go away.

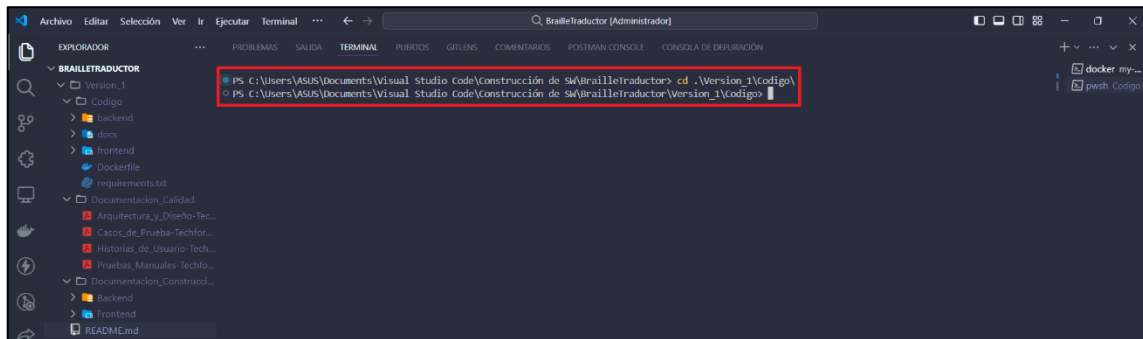
compiled with warnings.
```

Con eso se ha levantado el servicio frontend del traductor braille.

Backend

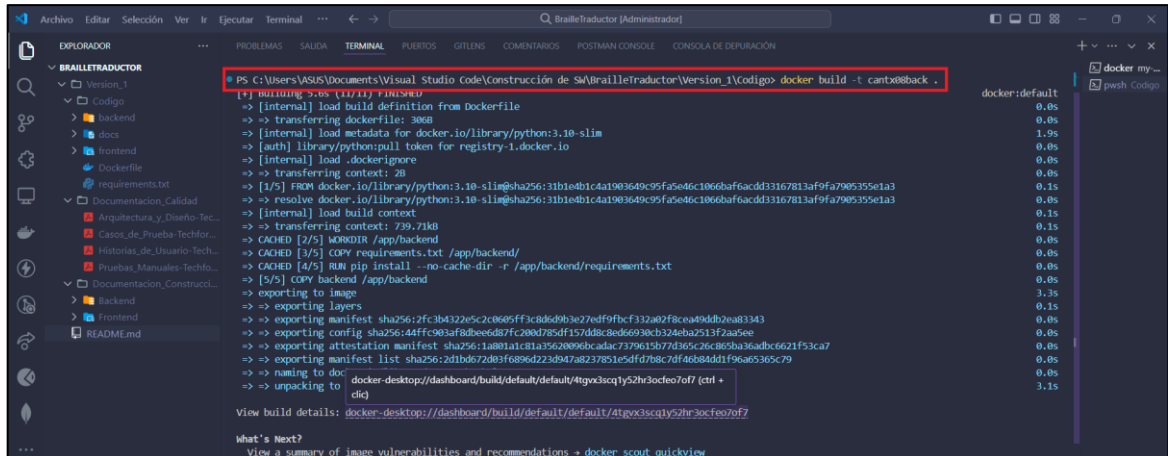
Para el servicio backend, abre otro terminal e ingresa a la carpeta Codigo mediante el comando `cd .\Version_1\Codigo\`

ISWD633 CONSTRUCCIÓN Y EVOLUCIÓN DE SOFTWARE



```
PS C:\Users\ASUS\Documents\Visual Studio Code\Construcción de SW\BrailleTraductor> cd .\Version_1\Codigo\
PS C:\Users\ASUS\Documents\Visual Studio Code\Construcción de SW\BrailleTraductor\Version_1\Codigo>
```

Construye otra imagen identificada como <tu-usuario>back mediante el comando `docker build -t <tu-usuario>back`.

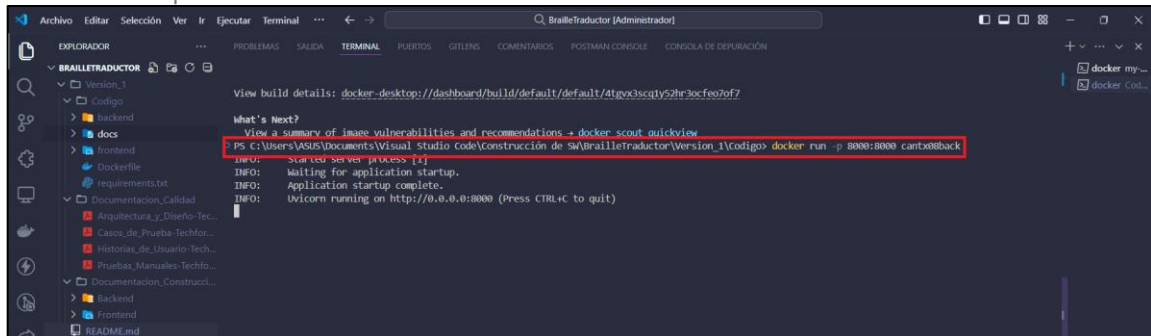


```
PS C:\Users\ASUS\Documents\Visual Studio Code\Construcción de SW\BrailleTraductor\Version_1\Codigo> docker build -t cantx08back .
```

Build output details:

```
[+] Building 5.5s (10/11) FINISHED
=> [internal] load build definition from Dockerfile
=> transferring dockerfile: 306B
=> [internal] load metadata for docker.io/library/python:3.10-slim
=> [auth] library/python:pull token for registry-1.docker.io
=> [internal] load dockerignore
=> transferring context: 2B
=> [1/5] FROM docker.io/library/python:3.10-slim@sha256:31b0e4b1c4a1903649c95fa5e46c1066baf6acd33167813af9fa7905355e1a3
=> resolve docker.io/library/python:3.10-slim@sha256:31b0e4b1c4a1903649c95fa5e46c1066baf6acd33167813af9fa7905355e1a3
=> [internal] load build context
=> transferring context: 739.71kB
=> CACHED [2/5] WORKDIR /app/backend
=> CACHED [3/5] COPY requirements.txt /app/backend/
=> CACHED [4/5] RUN pip install --no-cache-dir -r /app/backend/requirements.txt
=> [5/5] COPY backend /app/backend
=> exporting to image
=> exporting layers
=> exporting manifest sha256:2fc3b4322ef5c2c0665ff3cdd69b3e2edf9bfcf332a02f8ca40ddba01343
=> exporting config sha256:44ffc903af8d8e0ed87fc200d785df157dd8c8ed6930cb324eba2513f2aa5ee
=> exporting attestation manifest sha256:1a001a1c81a35e20096bcadac7379615b77d365c26c805ba36adbc6621f53ca7
=> exporting manifest list sha256:2d1bd672d03f6896d223d947a8237851e5df7b8c7df46b84dd1f96a65365c79
=> naming to docker-desktop/dashboards/build/default/default/atgpx3scaty5zhr3ocfe07of7 (ctrl + clic)
=> unpacking to docker-desktop/dashboards/build/default/default/atgpx3scaty5zhr3ocfe07of7
```

Luego, iniciaremos un contenedor con la imagen creada anteriormente utilizando el comando `docker run -p 8000:8000 <tu-usuario>back`



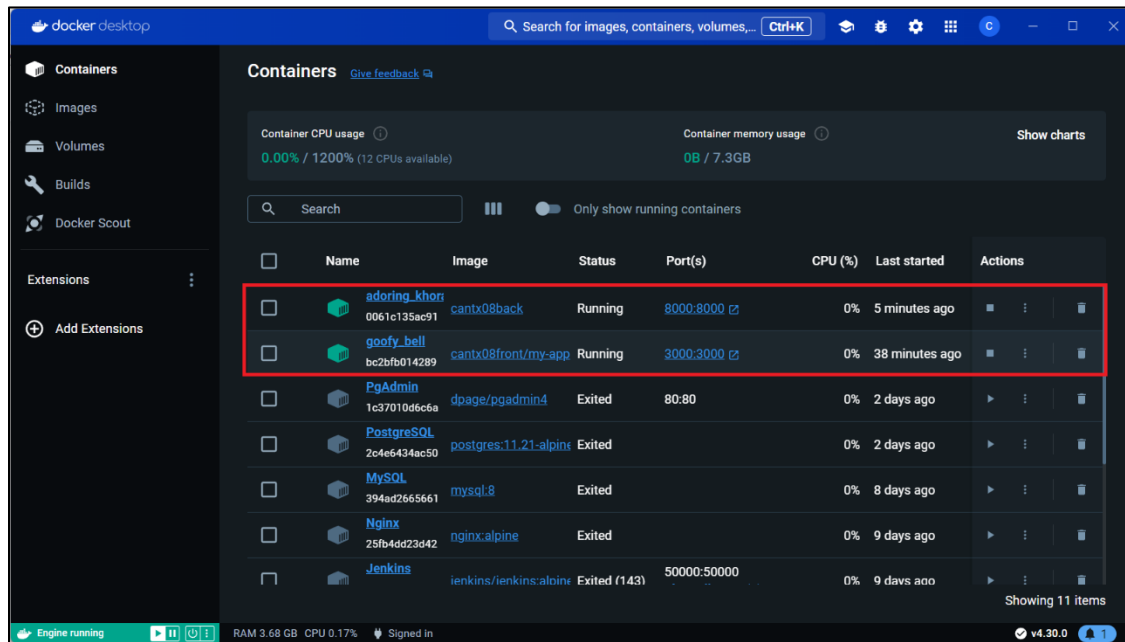
```
PS C:\Users\ASUS\Documents\Visual Studio Code\Construcción de SW\BrailleTraductor\Version_1\Codigo> docker run -p 8000:8000 cantx08back
```

Build output details:

```
INFO: started server process [1]
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)
```

ISWD633 CONSTRUCCIÓN Y EVOLUCIÓN DE SOFTWARE

Ahora se ha levantado el servicio de backend. Asegúrate de que estén ejecutándose los contenedores en la aplicación de Docker.



Para comprobar su funcionamiento ingresa a la dirección: <http://localhost:3000> y se te abrirá la siguiente página.

