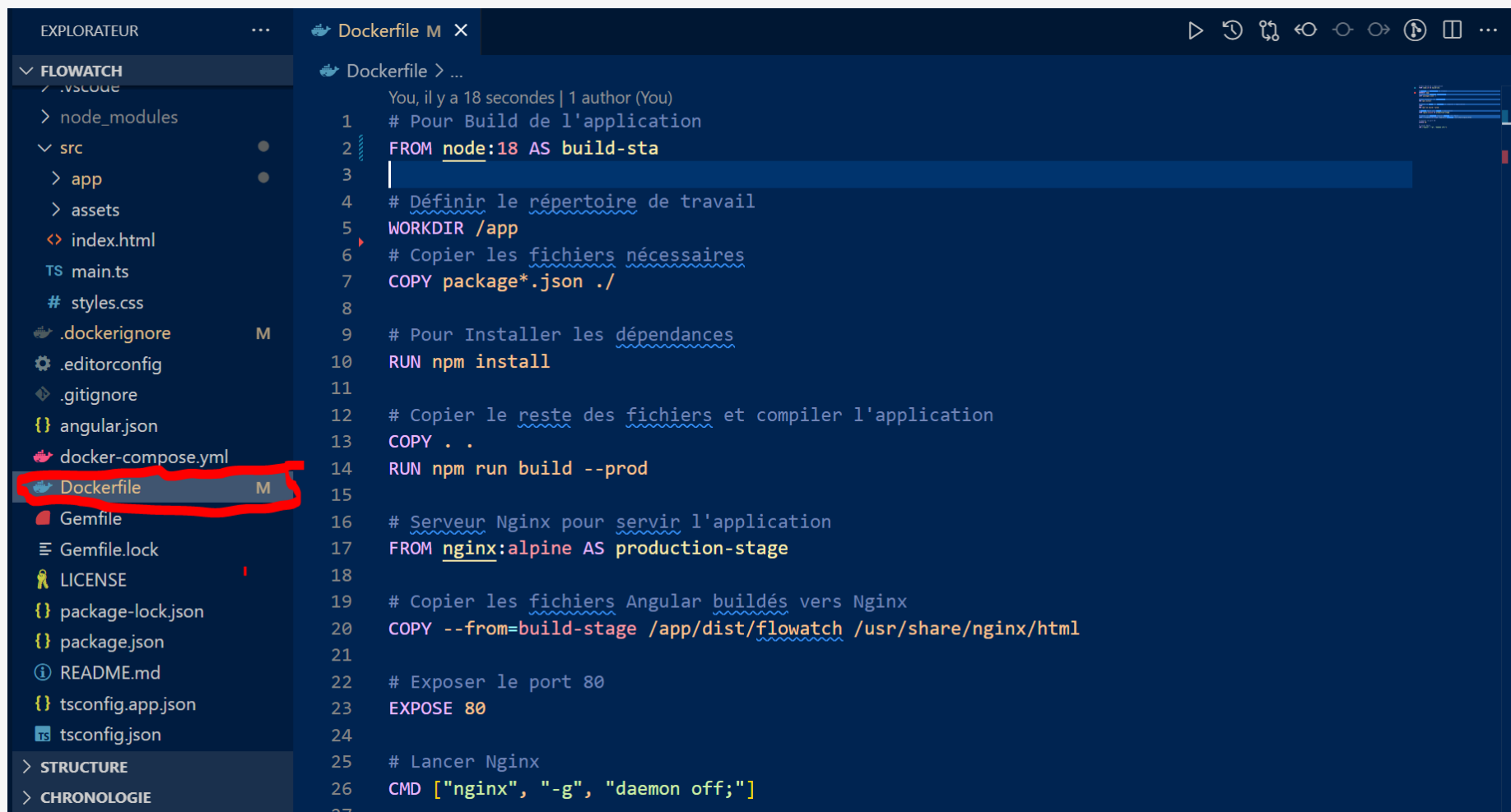


Pour déployer l'application j'ai utilisé Docker et permettre à d'autres de la déployer facilement sur leurs machines.
Voici les étapes mise en place :

1. Création d'un fichier Dockerfile

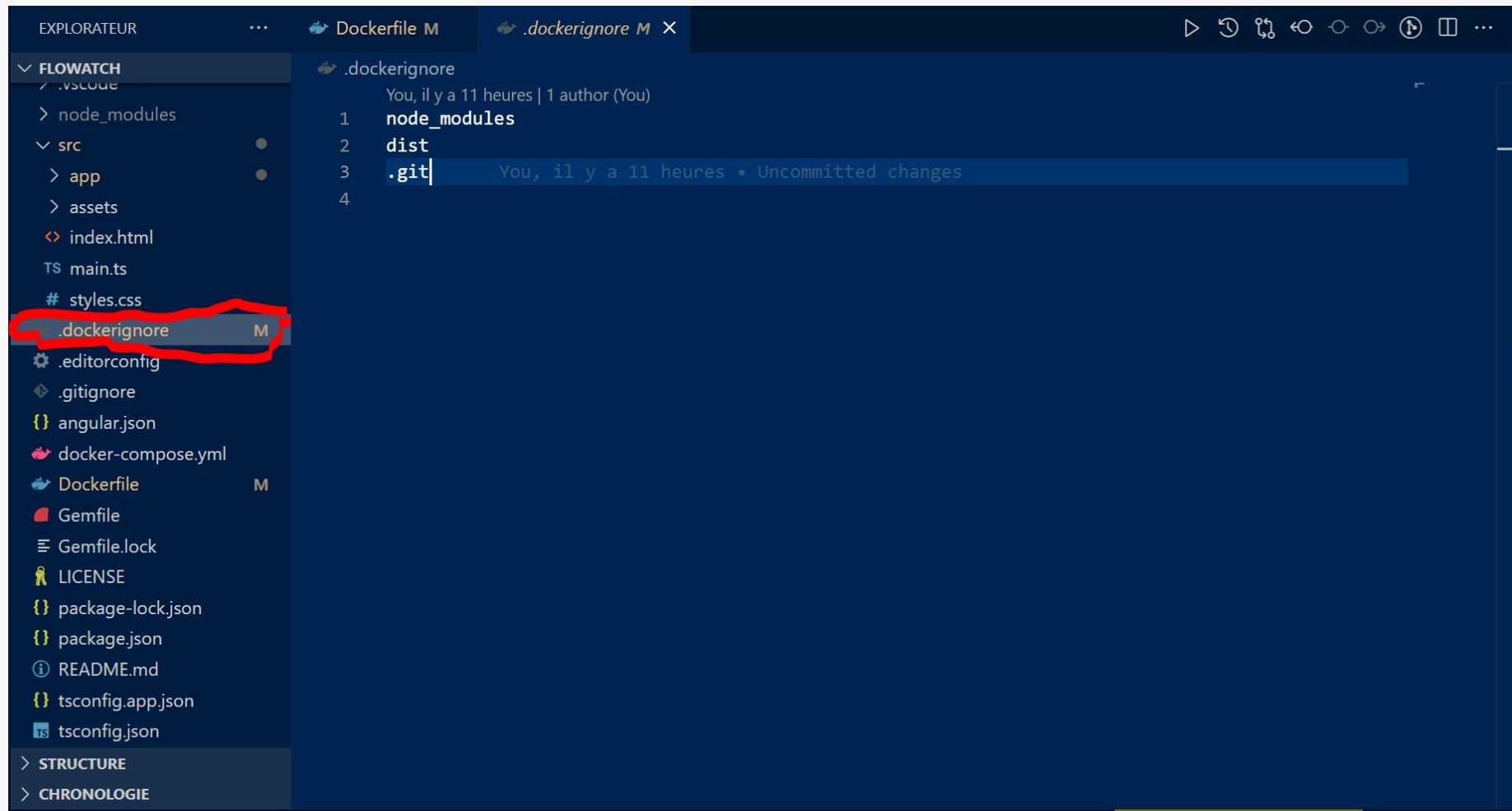
Ce fichier décrit comment l'application Angular doit être construite et exécutée dans un conteneur.
J'ai créé un fichier **Dockerfile** à la racine du projet et ajoute ceci :



```
1  # Pour Build de l'application
2  FROM node:18 AS build-stage
3
4  # Définir le répertoire de travail
5  WORKDIR /app
6  # Copier les fichiers nécessaires
7  COPY package*.json ./
8
9  # Pour Installer les dépendances
10 RUN npm install
11
12 # Copier le reste des fichiers et compiler l'application
13 COPY . .
14 RUN npm run build --prod
15
16 # Serveur Nginx pour servir l'application
17 FROM nginx:alpine AS production-stage
18
19 # Copier les fichiers Angular buildés vers Nginx
20 COPY --from=build-stage /app/dist/flowwatch /usr/share/nginx/html
21
22 # Exposer le port 80
23 EXPOSE 80
24
25 # Lancer Nginx
26 CMD ["nginx", "-g", "daemon off;"]
```

2. Créer un fichier `.dockerignore`

Puis j'ai ajouté un fichier `.dockerignore` à la racine pour éviter d'inclure des fichiers inutiles :



- Construire l'image Docker

```
docker build -t flowatch .
```

```

dell@IMANA-47 MINGW64 /d/ME/Flowatch (main)
$ docker build -t flowatch .
[+] Building 584.3s (15/15) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 659B
=> [internal] load metadata for docker.io/library/node:18
=> [internal] load metadata for docker.io/library/nginx:alpine
=> [auth] library/nginx:pull token for registry-1.docker.io
=> [internal] load .dockerignore
=> => transferring context: 66B
=> CACHED [production-stage 1/2] FROM docker.io/library/nginx:alpine@sha256:814a8e88df978ade80e584cc5b333144b9372a8e3c98872d07137dbf3b44d0e4
=> [build-stage 1/6] FROM docker.io/library/node:18
=> [internal] load build context
=> => transferring context: 297.29kB
=> CACHED [build-stage 2/6] WORKDIR /app
=> CACHED [build-stage 3/6] COPY package*.json ./
=> CACHED [build-stage 4/6] RUN npm install
=> [build-stage 5/6] COPY . .
=> [build-stage 6/6] RUN npm run build --prod
=> [production-stage 2/2] COPY --from=build-stage /app/dist/flowatch /usr/share/nginx/html
=> exporting to image
=> => exporting layers
=> => writing image sha256:34f136572fa0014c3387a3bfcdf30b2e7a1a760a71bbb2c23ca81a3676bf6342
=> => naming to docker.io/library/flowatch

```

docker:desktop-linux

	1.0s
	0.2s
	0.0s
	6.3s
	0.0s
	1.3s
	0.0s
	0.0s
	3.1s
	1.8s
	0.0s
	0.0s
	0.0s
	340.8s
	201.1s
	10.1s
	7.5s
	3.5s
	0.4s
	1.1s

- Exécuter le conteneur

```
docker run -p 8080:80 flowatch
```

```
dell@IMANA-47 MINGW64 /d/ME/Flowatch (main)
$ docker run -p 8080:80 flowatch
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2025/02/01 11:09:07 [notice] 1#1: using the "epoll" event method
2025/02/01 11:09:07 [notice] 1#1: nginx/1.27.3
2025/02/01 11:09:07 [notice] 1#1: built by gcc 13.2.1 20240309 (Alpine 13.2.1_git20240309)
2025/02/01 11:09:07 [notice] 1#1: OS: Linux 5.15.167.4-microsoft-standard-WSL2
2025/02/01 11:09:07 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2025/02/01 11:09:07 [notice] 1#1: start worker processes
2025/02/01 11:09:07 [notice] 1#1: start worker process 30
2025/02/01 11:09:07 [notice] 1#1: start worker process 31
2025/02/01 11:09:07 [notice] 1#1: start worker process 32
2025/02/01 11:09:07 [notice] 1#1: start worker process 33
172.17.0.1 - - [01/Feb/2025:11:18:56 +0000] "GET / HTTP/1.1" 200 5265 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/132.0.0.0 Safari/537.36 Edg/132.0.0.0" "-"
172.17.0.1 - - [01/Feb/2025:11:18:58 +0000] "GET /runtime.1e192dfb7a425f10.js HTTP/1.1" 200 894 "http://localhost:8080/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/132.0.0.0 Safari/537.36 Edg/132.0.0.0" "-"
172.17.0.1 - - [01/Feb/2025:11:18:59 +0000] "GET /polyfills.1c7aaa5620f12014.js HTTP/1.1" 200 33815 "http://localhost:8080/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/132.0.0.0 Safari/537.36 Edg/132.0.0.0" "-"
172.17.0.1 - - [01/Feb/2025:11:18:59 +0000] "GET /main.3a19c9d817dc20c6.js HTTP/1.1" 200 203718 "http://localhost:8080/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/132.0.0.0 Safari/537.36 Edg/132.0.0.0" "-"
172.17.0.1 - - [01/Feb/2025:11:18:59 +0000] "GET /scripts.43d8f451dbbb79b8.js HTTP/1.1" 200 159129 "http://localhost:8080/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/132.0.0.0 Safari/537.36 Edg/132.0.0.0" "-"
172.17.0.1 - - [01/Feb/2025:11:18:59 +0000] "GET /styles.cfbc08b6b99c5e7.css HTTP/1.1" 200 319487 "http://localhost:8080/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/132.0.0.0 Safari/537.36 Edg/132.0.0.0" "-"
172.17.0.1 - - [01/Feb/2025:11:18:59 +0000] "GET /assets/Flowatch.png HTTP/1.1" 200 19376 "http://localhost:8080/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/132.0.0.0 Safari/537.36 Edg/132.0.0.0" "-"
172.17.0.1 - - [01/Feb/2025:11:19:00 +0000] "GET /bootstrap-icons.bfa90bda92a84a6a.woff2?dd67030699838ea613ee6dbda90effa6 HTTP/1.1" 200 130396 "http://localhost:8080/styles.cfbc08b6b99c5e7.css" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/132.0.0.0 Safari/537.36 Edg/132.0.0.0" "-"
172.17.0.1 - - [01/Feb/2025:11:26:10 +0000] "GET /assets/Flowatch.png HTTP/1.1" 304 0 "http://localhost:8080/signup" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/132.0.0.0 Safari/537.36 Edg/132.0.0.0" "-"
```

Accéder à l'application :

Ouvre ton navigateur et va sur <http://localhost:8080>

4. Partager l'image Docker sur Docker Hub

Pour permettre aux autres d'utiliser sans devoir compiler eux-mêmes

- **Se connecter à Docker Hub**

```
docker login
```

```
de11@IMANA-47 MINGW64 /d/ME/Flowatch (main)
$ docker login
Authenticating with existing credentials...
Login Succeeded
```

- **Taguer l'image**

```
Docker tag flowatch nsengimana47/flowatch:latest
```

```
de11@IMANA-47 MINGW64 /d/ME/Flowatch (main)
$ docker tag flowatch nsengimana47/flowatch:latest
```

- **Pousser l'image sur Docker Hub**

```
docker push nsengimana47/flowatch :latest
```

```
de11@IMANA-47 MINGW64 /d/ME/Flowatch (main)
$ docker push nsengimana47/flowatch:latest
The push refers to repository [docker.io/nsengimana47/flowatch]
faf512bedd96: Pushed
5a760029e979: Mounted from library/nginx
23625999797d: Mounted from library/nginx
9aa22afcf27f: Mounted from library/nginx
59a5cb944b91: Mounted from library/nginx
598e577f3a23: Mounted from library/nginx
fd5f65a144ef: Mounted from library/nginx
a8903c9578e9: Mounted from library/nginx
ce5a8cde9eee: Mounted from library/nginx
latest: digest: sha256:cf51c8af7a4f3e0ddea23a953822eaadc767b9a72f5aeac2d7a7765c59fb0428 size: 2200
```

```
docker pull nsengimana47/flowatch :latest
```

```
docker run -p 8080:80 nsengimana47/flowatch :latest
```

```
dell@IMANA-47 MINGW64 /d/ME/Flowatch (main)
$ docker pull nsengimana47/flowatch:latest
docker run -p 8080:80 nsengimana47/flowatch
latest: Pulling from nsengimana47/flowatch
Digest: sha256:cf51c8af7a4f3e0ddea23a953822eaadc767b9a72f5aeac2d7a7765c59fb0428
Status: Image is up to date for nsengimana47/flowatch:latest
docker.io/nsengimana47/flowatch:latest
docker: Error response from daemon: driver failed programming external connectivity on endpoint pedantic_pascal (2d80cd7517f8aed658eaf4b1d451427679e278388255d6b40af845e1e76c41d9): Bind for 0.0.0.0:8080 failed: port is already allocated.
```

Comment tester l'application ? (Guide pour le déploiement sur un autre pc)

1. Prérequis

- ✓ Installer **Docker** et **Docker Compose**
- ✓ Avoir une connexion internet

2. Télécharger le projet

```
dell@IMANA-47 MINGW64 ~  
$ git clone git@github.com:IMANA47/Flowatch.git  
| cd Flowatch
```

3. Lancer l'application avec Docker Compose

```
dell@IMANA-47 MINGW64 /d/ME/Flowatch (main)  
$ docker-compose up -d
```

4. Accéder à l'application

Ouvrir un navigateur et aller sur : <http://localhost:4200>

5. Arrêter l'application

```
dell@IMANA-47 MINGW64 /d/ME/Flowatch (main)  
$ docker-compose down
```