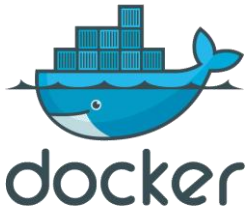
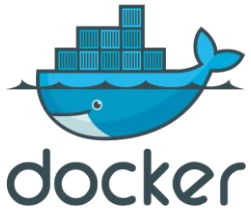
Environnement de développement

Docker, Git, .env



DOCKER & GIT



- POURQUOI UTILISER DOCKER ?
  - Indépendance des environnements : Les conteneurs Docker embarquent tout ce qu'il faut pour exécuter l'application (backend, frontend, base de données).
  - Facilité de configuration : Pas besoin de configurer manuellement les dépendances ou de résoudre des conflits de versions.
  - Cohérence des environnements : Peu importe la machine utilisée, le projet fonctionnera de la même manière grâce à Docker.
- PC PERSO vs PC LABO
  - Continuer sur PC perso en installer Docker (accès au BIOS nécessaire)
  - Basculer sur PC labo (quand vous voulez) et faire un git clone / pull.

Lancer mon appli avec Docker

- **Installer WSL**
  - Installer Docker
  - Installer Docker compose
- **Installer Git**
  - Paramétrer sa clé SSH avec son compte Github
- **Cloner son repos**
  - Installer les paquets nécessaires *(.gitignore)*
- **Paramétrer son docker-compose et Dockerfile**
- **Fichier .env**
  - `MYSQL_ROOT_PASSWORD=root`
  - `MYSQL_DATABASE=hackathon`
- **Build & Stop**
  - `docker-compose up`
  - `docker-compose down`
- **Connection à la base de données**

Dockerfile (backend)

- Le **Dockerfile** est un script qui définit **comment construire une image Docker** pour une application. Chaque instruction dans le fichier indique une étape pour configurer l'environnement.
- Etapes :

- FROM node:14 → *On commence par une image officielle de Node.js (version 14). C'est comme une machine prête à exécuter du code JavaScript.*
- WORKDIR /app → On configure /app comme répertoire de travail à l'intérieur du conteneur.
- COPY package.json ./* → *On copie les fichiers package.json et package-lock.json dans le conteneur pour préparer l'installation des dépendances.*
- RUN npm install -g nodemon → *On installe globalement nodemon, un outil qui redémarre automatiquement le serveur à chaque modification du code.*
- COPY . . → *On copie tous les fichiers du projet dans le conteneur.EXPOSE 5000 : On informe Docker que l'application utilisera le port 5000.*
- CMD ["nodemon", "app.js"] → *C'est la commande qui démarre l'application avec nodemon.*

docker-compose.yml (global projet)

- Le fichier **docker-compose.yml** est un outil qui permet de **définir et gérer plusieurs conteneurs en une seule commande**. C'est ici que tu décris comment tes services (frontend, backend, base de données) interagissent.
- Etapes :

- **frontend (Service Frontend) :**
  - image: nginx:latest → On utilise l'image officielle nginx, qui est un serveur web pour servir des fichiers HTML/CSS/JS.
  - ports: "8080:80" → Le port 80 du conteneur est accessible via le port 8080 sur la machine hôte.
  - volumes :
    - ./frontend:/usr/share/nginx/html → Le dossier frontend de la machine hôte est monté dans le conteneur à l'emplacement où Nginx cherche les fichiers web.
    - ./logs/nginx:/var/log/nginx → Les journaux Nginx sont sauvegardés sur la machine hôte.
  - networks :hackathon_network → Le conteneur communique avec les autres services via le réseau hackathon_network.
- **backend (Service Backend) :**
  - build: ./backend → Docker construit une image personnalisée pour le backend à partir du Dockerfile dans le dossier backend.
  - ports: "5000:5000" → Le port 5000 du conteneur est exposé sur la machine hôte.
  - depends_on: db → Ce service attend que le service db démarre avant de se lancer.
  - volumes :./backend:/app → Le code source du backend est partagé entre la machine hôte et le conteneur.
  - networks :hackathon_network → Le backend est relié au réseau hackathon_network.
- **db (Service Base de Données) :**
  - image: mariadb → On utilise l'image officielle de MariaDB pour la base de données.
  - environment :
    - MYSQL_ROOT_PASSWORD: root → Définit le mot de passe pour l'utilisateur root.
    - MYSQL_DATABASE: hackathon → *La base de données hackathon est créée au démarrage.*
  - ports: "3306:3306" → *Le port 3306 (par défaut pour MariaDB) est exposé sur la machine hôte.*
  - volumes :db_data:/var/lib/mysql → *Les données de la base sont persistantes et stockées dans le volume nommé db_data.*
  - volumes :db_data → *Un volume Docker persistant pour stocker les données de la base de données.*
  - networks :hackathon_network → Un réseau Docker pour permettre aux conteneurs de communiquer entre eux.