Installer Express et connecter à la bdd Postgres

step 1 : Installation dotenv / express / nodemon / pg

dans un nouveau dossier nous allons faire :

```
npm init -y
```

cela va valider l'installation automatique de notre fichier package.json

nous allons ensuite faire cette commande pour installer express :

```
npm i express
```

suivi de :

```
npm i dotenv nodemon pg
```

- dotenv : nous permet de lire le dossier `.env` qu'on va être mis en place
- nodemon : nous permet de relancer automatiquement notre serveur
- pg : créer une connexion entre docker et express

step 2 : création du fichier .env

Pour commencer nous allons créer un fichier `.env` à la racine du projet.

```
touch .env
```

dans le fichier `.env` nous allons mettre en place les informations de connexion à la bdd (si vous avez le moindre doute pensez à relire le projet adatabase sur la partie `docker-compose`)

```
POSTGRES_USER=postgres
POSTGRES_PASSWORD=postgres
POSTGRES_DB=adatabase
POSTGRES_PORT=5432
```

⚠ pensez à bien changer le type-module en `module` dans le `package.json` pour pouvoir utiliser les imports ES6 ⚠

```
{
  "type": "module",
  "scripts": {
    "start": "nodemon src/server.js"
  }
}
```

step 3 : création du serveur express

Nous allons créer un dossier `src` dans lequel nous allons créer deux fichiers : `db.js` et `server.js`

Dans le fichier `db.js` nous allons mettre en place le code suivant :

```
// on importe les modules nécessaires
// on initialise dotenv pour lire le fichier .env
import dotenv from "dotenv";
import { Pool } from "pg";
// on crée une instance d'express
dotenv.config();
// on configure la connexion à la bdd avec les variables d'environnement
const pool = new Pool({
  user: process.env.POSTGRES_USER,
  host: "localhost",
  database: process.env.POSTGRES_DB,
  password: process.env.POSTGRES_PASSWORD,
  port: process.env.POSTGRES_PORT,
});

// on tente de se connecter à la bdd et on affiche un message en fonction
du résultat
pool
  .connect()
  .then(() => {
    console.log("🟢 Connected to the database");
  })
  .catch((err) => {
    console.error("🔴 Error connecting to the database", err);
  });
export default pool;
```

ce fichier va générer la connexion à la bdd via le `pool` de pg

Dans le fichier `server.js` dans lequel nous allons mettre le code suivant :

```
import express from "express";

const app = express();

app.get("/", function (req, res) {
```

```
  res.send("Hello Ada!\n");
});

app.listen(3000, () => {
  console.log("🚀 Serveur lancé : http://localhost:3000");
});
```

nous avons donc un serveur express qui écoute sur le port 3000 et qui affiche "Hello Ada!" à la racine tout en se connectant à la bdd

niveau arborescence nous aurons donc :

```
└── src
  ├── db.js
  └── server.js
├── package.json
├── package-lock.json
├── .env
```

il nous suffit ensuite de lancer le serveur avec la commande :

```
nodemon src/server.js
```

nous allons ensuite faire cette commande pour installer express :

```
npm i express
```

suivi de :

```
npm i dotenv nodemon pg
```

- dotenv : nous permet de lire le dossier `.env` qu'on va être mis en place
- nodemon : nous permet de relancer automatiquement notre serveur
- pg : créer une connexion entre docker et express

step 4 : passons à la connexion à la bdd

```
import express from "express";
import pool from "./db.js";

const app = express();
```

```
app.get("/", async function (req, res) {
  const { rows } = await pool.query("SELECT * FROM resources");
  res.json(rows);
});

app.listen(3000, () => {
  console.log("🚀 Serveur lancé : http://localhost:3000");
});
```

nous avons donc un serveur express qui écoute sur le port 3000 et qui affiche le contenu de la table `resources` tout en se connectant à la bdd

il nous suffit ensuite de lancer le serveur avec la commande :

```
nodemon src/server.js
```