Découverte Express.js

🕶 Installation

- Créer un dossier pour y héberger nos fichiers.
- Dans le dossier, runner la commande pour installer Express.js :

```
npm i express
```

- Y copier le fichier app.js.
- Pour tester les requêtes API, utiliser Insomnia (gratuit) : **https://insomnia.rest/**

🤔 Pages intéressantes de doc Express

https://expressjs.com/fr/starter/basic-routing.html

https://expressjs.com/fr/starter/faq.html

Et surtout : **https://scotch.io/tutorials/use-expressjs-to-get-url-and-post-parameters**

🧐 Fonctionnalités à implémenter

- Pour l'instant, l'app propose une route toute simple en GET permettant de récupérer une liste de recettes située plus haut dans le fichier.
- On aimerait par la suite :
  - une route qui retourne une seule recette d'après son ID (/recettes/1, /recettes/2)
  - une route qui retourne les recettes pour un maximum de X personnes (/recettes?max_people=2)
  - une route qui retourne les recettes dont le temps de préparation (en minutes) est compris entre deux bornes (/recettes?preparation_min=5&preparation_max=20)
- Pour les routes avec des paramètres après le 2, on les récupère facilement avec req.query suivi du nom du paramètre (exemple: req.query.max_people).
- Pour la route avec un paramètre directement dans l'url (/recettes/2), la route doit être écrite au format /recettes/:id, et l'on récupère l'ID avec req.params.id.
- Par la suite, on peut aussi imaginer une route au format POST, au travers de laquelle on passerait une nouvelle recette au format JSON comme suit :

```
{
        title: 'Burger',
        preparation: 5,
        people: 1
    }
   // Pas d'id, il sera créé par l'application
```

Il suffira ensuite de l'ajouter au tableau recettes. Evidemment, pas de persistance, dès que l'app est fermée dans le terminal, les ajouts disparaissent ! Si tout fonctionne, lorsqu'on rappelle la route par défaut, on aura la recette dans la liste.