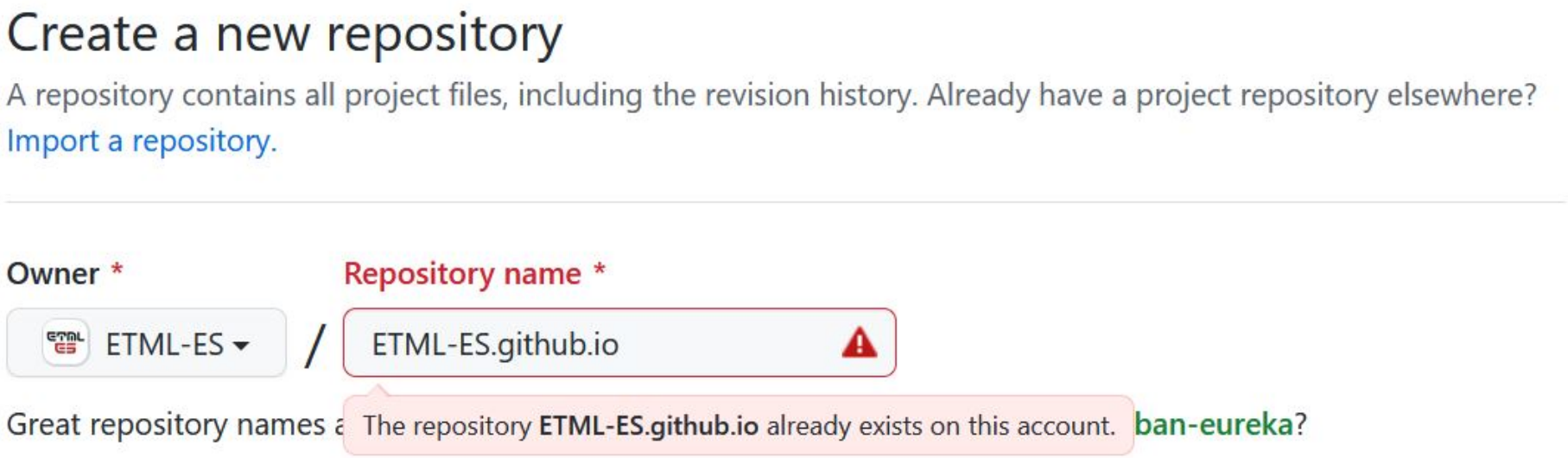
Partie 3: Mise en place de l'environnement d'exercice

Dans cette partie, nous allons commencer à mettre en place notre exercice qui consistera à créer une PWA.

Pour débuter

1. Créer un nouveau repository Github
   ATTENTION nommez le: *votrePseudoGithub.github.io* (nous verrons pourquoi par la suite, déjà une petite idée ?)



2. Clonez votre repository sur votre machine
3. Ouvrez-le dans votre éditeur/IDE favori
4. Créez une architecture de dossier suivant cette proposition (*italique* = dossier / **gras** = fichier)
    - *dist*
        - *assets*
        - *style*
        - **index.html**
        - **main.js**
5. Créez un squelette HTML de base (raccourcis clavier dans VSCode => *!* + *enter*).
6. Ajoutez y le titre de la PWA dans la balise *title* => Ciné★thèque.
7. Ajoutez dans la balise *head* la ligne

```
<script defer scr="main.js"><script>
```

Point théorique: le mot clé *defer* rend possible la déclaration de la balise *script* dans la balise *head* même si celle-ci se trouve au tout début du document HTML. Le chargement du script se fera quand même après le HTML. *defer* agit de la même façon que si le script avait été déclaré à la fin du document HTML.

8. Rajoutez cette ligne dans le fichier *main.js*

```
console.log(42);
```

9. Pour voir si tout ce que nous avons fait marche, il faut mettre en place un serveur web de développement. Vous avez le choix ! Soit vous êtes dans VSCode ou tout autre IDE qui implémente un live-server, vous lancez donc juste votre serveur. Soit vous installez un live-server sur votre machine qui marchera partout, peu importe votre IDE. Plus besoin d'être tributaire de votre éditeur de code.

    - **OPTIONEL**: pour faire cela, un peu de mise en place.
        - Premièrement, il faut avoir NodeJS d'installé sur votre machine, car nous allons installer ce qui s'appelle un *package npm*.
        - Il faut savoir qu'en installant *NodeJS* le package manager (*npm* pour Node Package Manager) de celui-ci est aussi installé. Vous avez donc la possibilité d'utiliser l'entièreté des package mis à disposition par la communauté sur le site de npm.
        - Nous allons pour notre part nous intéresser au package live-server. Ce package que nous allons installer **globalement** pour qu'il puisse être indépendant de notre projet, nous permettra, après configuration, de lancer un live-server pour notre projet en une seule petite commande.
        - Pour l'installation, insérez cette ligne dans votre console bash

```
$> npm install -g live-server
```

10. SUITE OPTIONEL vous avez donc *live-server* d'installer globalement sur votre machine. Pour information complémentaire sur une machine Windows, le fichier se trouve sous:

```
C:\Users\votreNomDeUsers\AppData\Roaming\npm
```

Maintenant dernière configuration, nous allons créer un fichier *package.json* dans notre repository que nous venons de cloner et où nous avons déjà amené une architecture de dossier (ce fichier viendra se positionner à la racine). Copier-coller le code suivant dans votre fichier nommé *package.json*.

```
{
  "name": "cinemathequepwa",
  "version": "1.0.0",
  "main": "index.html",
  "scripts": {
    "dev": "live-server dist/"
  },
  "author": "NF01",
  "license": "MIT"
}
```

La partie *script* crée juste un alias pour utiliser la commande live server. Au lieu d'écrire *live-server dist/* pour démarrer le live-server, nous aurons juste besoin d'écrire *npm run dev* dans la console.
Voilà ! maintenant que vous voudrez utiliser un live server dans un de vos projets il vous suffira simplement de mettre le fichier *package.json* dans votre repository et hop c'est fini !

11. Vous devriez donc avoir votre browser d'ouvert avec une page blanche (puisqu'il n'y a rien dans le HTML pour l'instant) et si vous ouvrez la console de votre navigateur vous devriez apercevoir la sortie *42* (le console.log(42) de notre fichier *main.js*).

Ressources

Introduction à NodeJS

Introduction à npm

Exercice pratique n°1

DONNÉE: le but de ce premier exercice pratique est de construire le squelette HTML de la page d'accueil de notre PWA. Comme dit précédemment, nous allons créer une cinémathèque. Il faut donc créer:

- Un logotype (= le nom de la PWA => Ciné★thèque)
  - Ce logotype doit être codé en HTML (donc pas de svg, ni image quelconque)
  - Voir la PWA finale pour avoir un aperçu

- Une section *globalPage* qui comporte:
  - Des filtres
    - Un champ de filtre textuel (pour pouvoir trier les films par nom)
    - Un bouton de filtre par étoiles (pour pouvoir trier le film par notes)
    - Un bouton de filtre par dates (pour pouvoir trier le film par dates de visionnage)
    - Un bouton de remise à zéro des filtres
  - Un bouton d'ajout (pour ajouter les films)
  - Une partie qui contiendra tous les objets films. Un film est composé de:
    - Une image
    - Un titre
    - Une date
    - Des étoiles
    - Un bouton de suppression de l'objet film