React/Express - Gestion des images

Il y a plusieurs façon de gérer les images dynamiquement. On peut les téléverser manuellement et garder le nom en mémoire. On peut également les téléverser avec un input de type `file` et les stocker dans un dossier sur le serveur ou les stocker dans une base de données en base64 qui est une représentation d'une image en texte.

1. Téléverser manuellement les images

On garde uniquement le nom de l'image dans la base de données. On peut ensuite utiliser ce nom pour afficher l'image dans le front-end. C'est la méthode actuellement en place dans le projet.

Avantages

- Simple à mettre en place
- On garde uniquement le nom de l'image dans la base de données
- Les images sont compilées avec le reste du code React
- Plus rapide à afficher

Inconvénients

- Nécessite de l'espace disque sur l'hebergeur.
- Plus difficile de gérer l'accès aux images (public ou privé)
- Pas de gestion des images (suppression, modification) sans manuellement les supprimer du dossier React

2. Téléverser les images dans un dossier sur le serveur

On stocke les images dans un dossier sur le serveur. On garde le nom de l'image dans la base de données. On peut ensuite utiliser ce nom pour afficher l'image dans le front-end.

On place les images dans le dossier public du serveur Express. On peut ensuite les afficher dans le front-end avec une URL provenant de notre serveur d'API. Par exemple, si on a une image nommée `image.jpg` dans le dossier `public/images`, on peut l'afficher dans le front-end avec l'URL `http://URL DE L'API/images/image.jpg`.

NodeJS possède un module de gestion des fichiers `fs` qui permet de lire, écrire, supprimer et modifier des fichiers et des répertoires. On peut utiliser ce module pour gérer les images dans le serveur Express.

Avantages

- On garde uniquement le nom de l'image dans la base de données
- Gestion des images plus simple avec le module `fs`

Inconvénients

- Requête supplémentaire pour chaque image à afficher
- Plus long à mettre en place

- Nécessite de l'espace disque sur le serveur

3. Téléverser les images dans une base de données en base64

On stocke les images dans une base de données en base64. On peut ensuite utiliser cette représentation pour afficher l'image dans le front-end. On les encode en base64 pour les stocker dans la base de données et on les décode pour les afficher dans le front-end.

Avantages

- Pas de gestion de dossier, ni d'images sur le serveur

Inconvénients

- Nécessite beaucoup d'espace dans une base de données MySQL, moins dans une base de données NoSQL (MongoDB, Firebase)

Envoyer une image avec un formulaire de React à Express

Pour envoyer une image avec un formulaire de React à Express, on peut utiliser un input de type `file` dans le formulaire. On peut ensuite envoyer le formulaire avec une requête HTTP de type `POST` pour envoyer l'image au serveur.

Le type de données envoyé est `multipart/form-data`. On peut utiliser la librairie `multer` pour gérer les fichiers dans Express.

```
<form onSubmit="{handleSubmit}">
    <input type="file" name="image" onChange="{handleFileChange}" />
    <button type="submit">Envoyer</button>
</form>
```

```
const [file, setFile] = useState(null);

const handleFileChange = (e) => {
    setFile(e.target.files[0]);
};

const handleSubmit = (e) => {
    e.preventDefault();

    const formData = new FormData();
    formData.append("image", file);

    fetch("http://URL DE API/images", {
        method: "POST",
```

```
        body: formData,
    })
        .then((response) => response.json())
        .then((data) => console.log(data))
        .catch((error) => console.error(error));
};
```

```
const express = require("express");
const multer = require("multer");
const upload = multer({ dest: "public/images" });

serveur.post("/images", upload.single("image"), (req, res) => {
    res.json({ message: "Image téléversée avec succès" });
});
```

Encoder une image en base64 dans React

Pour encoder une image en base64 dans React, on peut utiliser la méthode `FileReader` pour lire le contenu de l'image. On peut ensuite utiliser la méthode `readAsDataURL` pour lire le contenu de l'image en base64.

```
const [file, setFile] = useState(null);

const handleFileChange = (e) => {
    // On lit le contenu de l'image
    const reader = new FileReader();
    // On met à jour le state avec le contenu de l'image
    reader.onload = () => {
        setFile(reader.result);
    };
    // On démarre la lecture
    reader.readAsDataURL(e.target.files[0]);
};
```

Une image encodée en base64 ressemble à ceci:

```

EAAAABIP6P6MgG
```

Afficher une image en base64 dans React

Pour afficher une image en base64 dans une page Web, on peut utiliser la balise `img` avec l'attribut `src` qui contient le contenu de l'image en base64.

Ex:

```
<img
src="
bXBAQEAAAABIP6P6MgG" />
```