Documentation de la partie front-end

Documentation du Composant Application

Introduction

Cette partie décrit le composant Application, qui représente la page d'affichage des applications dans une application web. Le composant est chargé de récupérer et d'afficher la liste des applications, ainsi que de permettre la modification et la suppression des applications existantes.

Fonctionnalités

- Affichage de la liste des applications.
- Recherche d'applications par nom ou version.
- Ajout, modification et suppression d'applications.
- Affichage des détails d'une application.
- Gestion des alertes d'expiration de licence.

Structure du Code

Le composant Application est structuré comme suit :
- Importations des dépendances.
- Définition du composant Application.
- Définition des états avec useState.
- Fonctions auxiliaires.
- Effets secondaires avec useEffect.
- Fonctions pour la gestion des applications (ajout, suppression).
- Rendu JSX pour l'affichage des applications.

Référence API

États

- `listApplication`: Liste des applications récupérées du serveur.
- `searchTerm`: Terme de recherche pour filtrer les applications.
- `itemsPerPage`: Nombre d'éléments par page.
- `currentPage`: Page actuelle de la pagination.
- `openAlert`: État pour contrôler l'ouverture de l'alerte d'expiration de licence.
- `listAppArt`: Liste des alertes d'expiration de licence.

Fonctions

- `fetchApplication()`: Récupère la liste des applications depuis le serveur.
- `onDelClick(id)`: Supprime une application du serveur. Paramètre :

```
  - `id`: ID de l'application à supprimer.
- `extractDateOnly(dateTimeString)`: Extrait la date uniquement à partir d'une chaîne de date et heure.
- `checkLicenseExpiry()`: Vérifie l'expiration des licences des applications.
- `handleSearchChange(libelle)`: Met à jour le terme de recherche. Paramètre :
  - `libelle`: Terme de recherche saisi par l'utilisateur.
- `handleChangePaginate(value)`: Gère le changement de page de la pagination. Paramètre :
  - `value`: Valeur pour déterminer la nouvelle page (peut être -100 pour la page suivante, -200 pour la page précédente ou un nombre de page).
- `closeAlert()`: Ferme l'alerte d'expiration de licence.
- `openAlertFunction()`: Ouvre l'alerte d'expiration de licence.
```

Documentation du Composant Deploiement

Introduction

```
Cette partie décrit le composant Deploiement, qui est une partie d'une application web. Le composant est responsable de la gestion des déploiements des différentes applications.
```

Fonctionnalités

```
- Affichage de la liste des déploiements.
- Ajout d’un nouveau déploiement.
- Modification d’un déploiement existante.
- Suppression d’un déploiement.
```

Structure du Code

```
Le composant Licence est structuré comme suit :
- Importations des dépendances.
- Définition du composant Licence.
- Définition des états avec useState.
- Fonctions auxiliaires.
- Effets secondaires avec useEffect.
- Fonctions pour la gestion des licences (ajout, modification, suppression).
- Rendu JSX pour l'affichage des licences.
```

Référence API

États

- `listLicence`: Liste des licences récupérées du serveur.
- `listDeploiement`: Liste des déploiements.
- `listApplication`: Liste des applications.
- `searchTerm`: Terme de recherche pour filtrer les déploiements.
- `itemsPerPage`: Nombre d'éléments par page pour la pagination.
- `currentPage`: Numéro de la page courante.

- `open` et `open2`: États pour ouvrir/fermer les boîtes de dialogue d'ajout et de modification respectivement.
- `serveur`: État pour stocker le nom du serveur.
- `dateDeploiement`: État pour stocker la date de déploiement.
- `chargerDeploiement`: État pour stocker les détails du déploiement à charger pour la modification.
- `selectedOption`: Option sélectionnée dans la liste déroulante d'application.

Méthodes Principales

- `fetchDeploiement()`: Récupère les déploiements depuis le serveur.
- `fetchApplication()`: Récupère les applications depuis le serveur.
- `ajouterDeploiement()`: Ajoute un déploiement en envoyant une requête POST au serveur.
- `modifierDeploiement(id)`: Modifie un déploiement en envoyant une requête PATCH au serveur.
- `onDelClick(id)`: Supprime un déploiement en envoyant une requête DELETE au serveur.
- `openDialog()`: Ouvre la boîte de dialogue d'ajout.
- `closeDialog()`: Ferme la boîte de dialogue d'ajout.
- `openDialog2(id)`: Ouvre la boîte de dialogue de modification avec les détails du déploiement sélectionné.
- `closeDialog2()`: Ferme la boîte de dialogue de modification.

Autres Méthodes et Fonctions Utilitaires

- `handleSearchChange(event)`: Gère le changement dans le champ de recherche.
- `handleChangePaginate(value)`: Gère le changement de page dans la pagination.
- `handleChange(event)`: Gère le changement de sélection dans la liste déroulante d'application.
- `toDateFr(dateISO)`: Formate une date au format français.
- `extractDateOnly(dateTimeString)`: Extrait la date uniquement à partir d'une chaîne de date/heure.

Documentation du Composant Application

Introduction

```
Cette partie décrit le composant Application, qui représente la page d'affichage
des applications dans une application web. Le composant est chargé de récupérer
et d'afficher la liste des applications, ainsi que de permettre la modification
et la suppression des applications existantes.
```

Fonctionnalités

```
- Affichage de la liste des applications.
```

- Recherche d'applications par nom ou version.
- Ajout, modification et suppression d'applications.
- Affichage des détails d'une application.
- Gestion des alertes d'expiration de licence.

Structure du Code

Le composant Application est structuré comme suit :
- Importations des dépendances.
- Définition du composant Application.
- Définition des états avec useState.
- Fonctions auxiliaires.
- Effets secondaires avec useEffect.
- Fonctions pour la gestion des applications (ajout, suppression).
- Rendu JSX pour l'affichage des applications.

Référence API

États

- `listApplication`: Liste des applications récupérées du serveur.
- `searchTerm`: Terme de recherche pour filtrer les applications.
- `itemsPerPage`: Nombre d'éléments par page.
- `currentPage`: Page actuelle de la pagination.
- `openAlert`: État pour contrôler l'ouverture de l'alerte d'expiration de licence.
- `listAppArt`: Liste des alertes d'expiration de licence.

Fonctions

- `fetchApplication()`: Récupère la liste des applications depuis le serveur.
- `onDelClick(id)`: Supprime une application du serveur. Paramètre :
  - `id`: ID de l'application à supprimer.
- `extractDateOnly(dateTimeString)`: Extrait la date uniquement à partir d'une chaîne de date et heure.
- `checkLicenseExpiry()`: Vérifie l'expiration des licences des applications.
- `handleSearchChange(libelle)`: Met à jour le terme de recherche. Paramètre :
  - `libelle`: Terme de recherche saisi par l'utilisateur.
- `handleChangePaginate(value)`: Gère le changement de page de la pagination. Paramètre :
  - `value`: Valeur pour déterminer la nouvelle page (peut être -100 pour la page suivante, -200 pour la page précédente ou un nombre de page).
- `closeAlert()`: Ferme l'alerte d'expiration de licence.
- `openAlertFunction()`: Ouvre l'alerte d'expiration de licence.

Exemple d'Intégration

Pour utiliser un composant de nom nom_composant dans une application React, suivez ces étapes :
1. Importez le composant Licence dans le fichier où vous souhaitez l'utiliser.
2. Intégrez le composant Licence dans la structure JSX de votre application.
3. Le composant Licence prend en charge l'affichage et la gestion des licences.

Voici un exemple d'intégration du composant Licence dans une application React :

```jsx
import React from 'react';
import Licence from './Licence';

function App() {
  return (
    <div>
      <h1>Application de Gestion des Licences</h1>
      < nom_composant />
    </div>
  );
}

export default App;
```