TP Course

- Créer 3 pages components dans le dossier js/pages :
  CourseAdd.jsx
  CourseList.jsx
  CourseOverview.jsx
- Créer 2 composants dans le dossier js/components
  CourseItem et NavBar

TP Course

- Importer les pages et la barre de navigation a partie de fichier js/AppCourse.jsx
- Utiliser deux buttons pour naviguer entre List des cours et l'ajout d'un cours
- Entrer le code minimum pour lancer l'application

Dans la page blade, utiliser ce code dans un script Element

const fromBackend = <?php echo $fromBackend ?>

Laravel Routes

```
use Illuminate\Support\Facades\Route;


// course
Route::get('add', function () {
    $fromBackend = '20';
    return view('BladeName',compact('fromBackend'));
});
```

Conditionnel rendering

```
<div className="container">
            {
fromBackend == 10 ? <CourseList /> :
fromBackend == 20 ? <CourseAdd /> :
fromBackend == 30 ? <CourseOverview /> : {}
            }
        </div>
```

Ajouter course

- Creer une forme pour recupirer l’entrer d’un cours
- Afficher le cours entrer a partir de submit

```
const [course, setCourse] = useState({
  name: '',
  instructor: '',
  description: '',
  domain: '',
  start_date: '',
  end_date: '',
  location: '',
  status: ''
});
```

Cours overview

- Afficher les attributes d’un cours

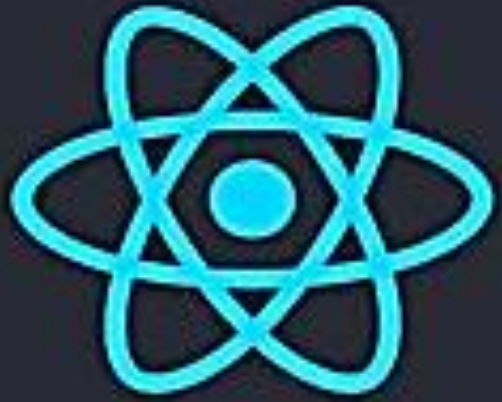
name: 'Introduction to Computer Science',
description: 'An introductory course.',
instructor: 'Dr. John Smith',
domain: 'Computer Science',
start_date: '2024-09-01',
end_date: '2024-12-15',
location: 'Room 101, Computer Science Building',
status: 'Active'

Liste des cours

- Utiliser le meme code de CourseOverview dans le composant CourseItem
- Importer CourseItem pour afficher deux course minimum

Utiliser des données aléatoire

Cycle de vie au montage d'un composant



React

Fonction du cycle de vie

Les fonctions de cycle de vie sont des méthodes spéciales qui sont appelées automatiquement à différents moments du cycle de vie d'un composant. Ces méthodes vous permettent d'effectuer certaines actions lorsque le composant est créé, mis à jour ou détruit. Cependant, avec l'introduction des React Hooks, les méthodes de cycle de vie traditionnelles comme componentDidMount, componentDidUpdate et componentWillUnmount ont été remplacées par des Hooks comme useEffect.

Voici un bref aperçu de certaines méthodes courantes du cycle de vie dans React, ainsi que leurs équivalents à l'aide de Hooks :

componentDidMount :

Cette méthode est appelée immédiatement après qu'un composant est monté (c'est-à-dire inséré dans l'arbre DOM).

Equivalent Hook : useEffect avec un tableau de dépendance vide.

Exemple :

```
useEffect(() => {
  // Perform side effects (e.g., data fetching) after the component is mounted
  fetchData();
}, []);
```

componentDidUpdate :

Cette méthode est appelée immédiatement après la mise à jour. Cette méthode n'est pas appelée pour le rendu initial.

Equivalent Hook : useEffect avec les dépendances.

Exemple :

```
useEffect(() => {
  // Perform side effects (e.g., data fetching) when certain dependencies change
  fetchData();
}, [dependency1, dependency2]);
```

componentWillUnmount :

Cette méthode est appelée immédiatement avant qu'un composant ne soit démonté et détruit.

Hook équivalent : useEffect avec une fonction de nettoyage.

Exemple :

```
useEffect(() => {
  // Perform setup (e.g., event listeners) when the component is mounted
  const subscription = subscribeToData();
  // Return a cleanup function to unsubscribe when the component is unmounted
  return () => {
    unsubscribeFromData(subscription);
  };
}, []);
```

Ces exemples montrent comment répliquer le comportement des méthodes de cycle de vie traditionnelles à l'aide de React Hooks. Il est important de noter que les Hooks offrent un moyen plus flexible et concis de gérer les effets secondaires et le cycle de vie des composants fonctionnels.

TP cours 2

- Envoyer une requete GET, au lien : /api/course une fois la page List Afficher

```
axios.get('api/course')
    .then(response => {
      console.log(response.data);
    })
    .catch(error => {
      console.error(error);
    });
```

TP course

- Creer une endpoint dans le backend pour recupirer les cours

```
// controller
 public function index()
    {
        $courses = Course::all();
        return response()->json($courses);
    }
```