TP REACT-REDAX TOOLKIT

1 : Créer un projet React

2 : Installer Redux Toolkit

Installez Redux Toolkit et les dépendances nécessaires dans votre application React :

npm install @reduxjs/toolkit react-redux

Pour utiliser Redux Toolkit avec React, suivez ces étapes :

### Étape 1 : Créer des slice avec Redux Toolkit

Dans Redux Toolkit, les reducers sont créés à l'aide des "slices". Un slice combine les actions et les reducers pour une partie spécifique de votre état d'application.

1. Créer un slice CompteurSlice.js qui contient état initiale de compteur :
2. Ajouter au CompteurSlice deux réducteurs permettant l’accès aux données de notre état initiale d’incrémenter et décrémenter le compteur

import { createSlice } from '@reduxjs/toolkit'

const initState = { value: 0 }

const CompteurSlice = createSlice({

name: 'compteur',

initialState: { value: 0 },

reducers: {

increment:(state)=> {

state.value++

},

decrement:(state)=> {

state.value--

},

},

})

export const { increment, decrement } = CompteurSlice.actions;

export default CompteurSlice.reducer;

### Étape 2 : Créer un store Redux

Créez un fichier pour votre store Redux, généralement nommé **store.js**. Dans ce fichier, importez **configureStore** depuis Redux Toolkit pour créer le store

1. Importer dans le store redux le slice CompteurSlice

***Store.js :***

import { configureStore } from '@reduxjs/toolkit';

// Importez vos reducers ici

// Exemple :

import CompteurSlice from './CompteurSlice';

const Store = configureStore({

  reducer: {

 Compteur:CompteurSlice

  },

});

export default Store;

### Étape 3 : Utiliser les données du store dans vos composants React

Dans vos composants React, vous pouvez accéder aux données du store et dispatcher des actions pour modifier l'état global :

Créer un composant react qui permet un d’afficher le compteur

* Bouton incrémenter
* Bouton décrémenter

***App.js :***

import './App.css';

import { useDispatch, useSelector } from 'react-redux';

import { increment, decrement} from './CompteurSlice'

function App() {

  const dispatch =useDispatch();

  const counter =useSelector((st) => st.Compteur.value);

  return (

    <div className="App">

     <div>

      <h1>Counter: {counter}</h1>

      <button onClick={() => dispatch(increment())}>Increment</button>

      <button onClick={() => dispatch(decrement())}>Decrement</button>

    </div>

    </div>

  );

}

export default App;

### Étape 4 : Utiliser le store dans votre application React

Dans le composant racine de votre application (généralement src/index.js), enveloppez votre application avec le <Provider> de react-redux pour fournir le store Redux à tous les composants :

Index.js

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import { Provider } from 'react-redux';

import store from './Store';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<Provider store={store}>

    <App />

  </Provider>

);