Criando o Projeto VITE

npm create vite@latest meu-app -- --template react-ts
cd meu-app
npm install
npm install @reduxjs/toolkit react-redux

Criando Store

```
// src/store/store.ts
import { configureStore } from '@reduxjs/toolkit'

export const store = configureStore({
  reducer: {
    // Reducers vão aqui
  },
})
```

Configurando o Provider

```
// src/main.tsx
import React from 'react'
import ReactDOM from 'react-dom'
import App from './App'
import { store } from './store/store'
import { Provider } from 'react-redux'

ReactDOM.render(
  <React.StrictMode>
    <Provider store={store}>
      <App />
    </Provider>
  </React.StrictMode>,
  document.getElementById('root')
)
```

Criando o CaixaReducer

```
// src/store/caixaReducer.ts
import { createSlice, PayloadAction } from '@reduxjs/toolkit'

interface CaixaState {
  saldo: number
}

const initialState: CaixaState = {
  saldo: 0,
}

const caixaSlice = createSlice({
  name: 'caixa',
  initialState,
  reducers: {
    depositar: (state, action: PayloadAction<number>) => {
      state.saldo += action.payload
    },
    sacar: (state, action: PayloadAction<number>) => {
      state.saldo -= action.payload
    },
  },
})

export const { depositar, sacar } = caixaSlice.actions
export default caixaSlice.reducer
```

Integrar o caixaReducer à Store

```tsx
// src/store/store.ts
import { configureStore } from '@reduxjs/toolkit'
import caixaReducer from './caixaReducer'

export const store = configureStore({
  reducer: {
    caixa: caixaReducer,
  },
})
```

Criando Componente Caixa

```tsx
// src/components/Caixa.tsx
import React, { useState } from 'react'
import { useAppDispatch, useAppSelector } from '../store/hooks'
import { depositar, sacar } from '../store/caixaReducer'

function Caixa() {
  const [quantia, setQuantia] = useState(0)
  const saldo = useAppSelector((state) => state.caixa.saldo)
  const dispatch = useAppDispatch()

  const handleDeposito = () => {
    dispatch(depositar(quantia))
  }

  const handleSaque = () => {
    dispatch(sacar(quantia))
  }

  return (
    <div>
      <h1>Saldo: {saldo}</h1>
      <input type="number" value={quantia} onChange={(e) => setQuantia(Number(e.target.value))} />
      <button onClick={handleDeposito}>Depositar</button>
      <button onClick={handleSaque}>Sacar</button>
    </div>
  )

export default Caixa;
```

Uso do Componente Caixa

```tsx
// src/App.tsx
import React from 'react'
import Caixa from './components/Caixa'

function App() {
  return (
    <div>
      <Caixa />
    </div>
  )
}

export default App;
```

Persistir Dados

npm install redux-persist

```tsx
// src/store/store.ts
import { configureStore } from '@reduxjs/toolkit'
import storageSession from 'redux-persist/lib/storage/session'
import { combineReducers } from 'redux'
import { persistReducer, persistStore } from 'redux-persist'

import caixaReducer from './caixaReducer'

const rootReducer = combineReducers({
  // Adicione seus reducers aqui
  caixa: caixaReducer,
})

const persistConfig = {
  key: 'root',
  storage: storageSession,

  //blacklist:["caixa"] caso queira que os dados desse Reducer não seja salvos no LocalStorage
}

const persistedReducer = persistReducer(persistConfig, rootReducer)

export const store = configureStore({
  reducer: persistedReducer,
})

export const persistor = persistStore(store);
```

Utilizando este método, é possível armazenar os dados no SessionStorage. Dessa forma, mesmo que a página seja recarregada, as informações do estado mantidas pelo Reducer permanecerão intactas.

Utilizando Arrays

Modificar a Estrutura do Estado

```tsx
// src/store/caixaReducer.ts
import { createSlice, PayloadAction } from '@reduxjs/toolkit';

interface Transacao {
  tipo: 'deposito' | 'saque';
  valor: number;
}

interface CaixaState {
  saldo: number;
  transacoes: Transacao[];
}

const initialState: CaixaState = {
  saldo: 0,
  transacoes: [],
};
```

Atualizar as Actions para Registrar Transações

```tsx
const caixaSlice = createSlice({
  name: 'caixa',
  initialState,
  reducers: {
    depositar: (state, action: PayloadAction<number>) => {
      state.saldo += action.payload;
      state.transacoes.push({ tipo: 'deposito', valor: action.payload });
    },
    sacar: (state, action: PayloadAction<number>) => {
      state.saldo -= action.payload;
      state.transacoes.push({ tipo: 'saque', valor: action.payload });
    },
  },
});

export const { depositar, sacar } = caixaSlice.actions;
export default caixaSlice.reducer;
```

Exibir Transações no Componente React

```tsx
// src/components/Caixa.tsx
import React, { useState } from 'react';
import { useDispatch, useSelector } from 'react-redux';
import { depositar, sacar } from '../store/caixaReducer';

function Caixa(){
  const [quantia, setQuantia] = useState(0);
  const { saldo, transacoes } = useSelector((state: any) => state.caixa);
  const dispatch = useDispatch();

  // Resto do código...

  return (
    <div>
      <h1>Saldo: {saldo}</h1>
      <input
        type="number"
        value={quantia}
        onChange={(e) => setQuantia(Number(e.target.value))}
      />
      <button onClick={() => dispatch(depositar(quantia))}>Depositar</button>
      <button onClick={() => dispatch(sacar(quantia))}>Sacar</button>

      <h2>Transações:</h2>
      <ul>
        {transacoes.map((transacao, index) => (
          <li key={index}>
            {transacao.tipo} de R${transacao.valor}
          </li>
        ))}
      </ul>
    </div>
  );
};

export default Caixa;
```