1. Store, Slices et components

```
// Project: React-Redux-Stagiaires-App

/*
Instructions:
1) Create a React app (e.g. using create-react-app or Vite).
2) Install dependencies: react-redux @reduxjs/toolkit react-router-dom bootstrap
   npm install react-redux @reduxjs/toolkit react-router-dom bootstrap
3) Replace src/App.jsx and src/index.js with the contents below (or split into multiple files as
indicated).
4) Start the app.

This single file shows:
- Redux store with 2 slices: stagiaireSlice and absenceSlice
- Actions: filterByGroup / setFilterGroup, decrementNoteDiscipline, addAbsence
- Components: Navbar, Stagiaires (stagiaire list + select + checkbox + save), Absences (list by group
sorted by date)
- Routing with react-router-dom
- Uses Bootstrap for styling
*/

// =================== src/store.js ===================
import React from 'react';
import { configureStore, createSlice, current } from '@reduxjs/toolkit';
import { Provider, useDispatch, useSelector } from 'react-redux';
import { BrowserRouter as Router, Routes, Route, Link } from 'react-router-dom';
import 'bootstrap/dist/css/bootstrap.min.css';

// Helper to get distinct groups from stagiaires
const distinctGroups = (stagiaires) => {
  return Array.from(new Set(stagiaires.map(s => s.groupe))).sort();
};

// Initial data for stagiaires
const initialStagiaires = [
  { cef: 'CEF001', nom: 'Dupont', prenom: 'Alice', groupe: 'G1', noteDiscipline: 20 },
  { cef: 'CEF002', nom: 'Martin', prenom: 'Bob', groupe: 'G1', noteDiscipline: 20 },
  { cef: 'CEF003', nom: 'Nguyen', prenom: 'Clara', groupe: 'G2', noteDiscipline: 20 },
  { cef: 'CEF004', nom: 'Khan', prenom: 'David', groupe: 'G2', noteDiscipline: 20 },
  { cef: 'CEF005', nom: 'Smith', prenom: 'Eva', groupe: 'G3', noteDiscipline: 20 }
];

// =================== stagiaireSlice ===================
const stagiaireSlice = createSlice({
  name: 'stagiaires',
  initialState: {
    list: initialStagiaires,
    filterGroupe: 'ALL'
  },
  reducers: {
    setFilterGroupe(state, action) {
      state.filterGroupe = action.payload; // 'ALL' or groupe name
    },
    decrementNoteDiscipline(state, action) {
      // payload: { cef, amount }
      const { cef, amount = 1 } = action.payload;
      const st = state.list.find(s => s.cef === cef);
      if (st) {
        // ensure not below 0
        st.noteDiscipline = Math.max(0, st.noteDiscipline - amount);
      }
    }
  }
});

// =================== absenceSlice ===================
const absenceSlice = createSlice({
  name: 'absences',
  initialState: {
    list: [
      // sample absence
      { id: 'A1', cef: 'CEF002', groupe: 'G1', date: '2025-11-15' }
    ],
    filterGroupe: 'ALL'
  },
  reducers: {
    addAbsence(state, action) {
      // payload: { cef, groupe, date }
      const { cef, groupe, date } = action.payload;
      const id = `${Date.now()}-${Math.random().toString(36).slice(2,7)}`;
      state.list.push({ id, cef, groupe, date });
    },
    setFilterGroupeAbs(state, action) {
      state.filterGroupe = action.payload;
    }
  }
});

const store = configureStore({
  reducer: {
    stagiaires: stagiaireSlice.reducer,
    absences: absenceSlice.reducer
  }
});

// Export actions for local use
const { setFilterGroupe, decrementNoteDiscipline } = stagiaireSlice.actions;
const { addAbsence, setFilterGroupeAbs } = absenceSlice.actions;
```

```
// =================== Components ===================

// Navbar component
function AppNavbar() {
  return (
    <nav className="navbar navbar-expand-lg navbar-dark bg-primary mb-4">
      <div className="container">
        <Link className="navbar-brand" to="/">StagiairesApp</Link>
        <button className="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarNav">
          <span className="navbar-toggler-icon"></span>
        </button>
        <div className="collapse navbar-collapse" id="navbarNav">
          <ul className="navbar-nav">
            <li className="nav-item"><Link className="nav-link"
to="/stagiaires">Stagiaires</Link></li>
            <li className="nav-item"><Link className="nav-link"
to="/absences">Absences</Link></li>
          </ul>
        </div>
      </div>
    </nav>
  );
}

// Stagiaires list component
function Stagiaires() {
  const dispatch = useDispatch();
  const stagiaires = useSelector(state => state.stagiaires.list);
  const filter = useSelector(state => state.stagiaires.filterGroupe);
  const absences = useSelector(state => state.absences.list);

  // local state to store checkboxes per cef
  const [checked, setChecked] = React.useState({});


  // handle select change
  const onSelectGroup = (e) => {
    dispatch(setFilterGroupe(e.target.value));
    dispatch(setFilterGroupeAbs(e.target.value));
  };

  // filtered stagiaires
  const filtered = React.useMemo(() => {
    return filter === 'ALL' ? stagiaires : stagiaires.filter(s => s.groupe === filter);
  }, [stagiaires, filter]);

  const toggle = (cef) => {
    setChecked(prev => ({ ...prev, [cef]: !prev[cef] }));
  };
```

```
 const handleSaveAbsence = (s) => {
  if (!checked[s.cef]) return; // nothing to save
  const today = new Date().toISOString().slice(0,10); // YYYY-MM-DD
  dispatch(addAbsence({ cef: s.cef, groupe: s.groupe, date: today }));
  // optionally decrement discipline note
  dispatch(decrementNoteDiscipline({ cef: s.cef, amount: 1 }));
  // uncheck after saving
  setChecked(prev => ({ ...prev, [s.cef]: false }));
 };

 return (
  <div className="container">
   <h2>Liste des stagiaires</h2>
   <div className="mb-3 row">
    <label className="col-sm-2 col-form-label">Filtrer par groupe</label>
    <div className="col-sm-4">
     <select className="form-select" value={filter} onChange={onSelectGroup}>
      <option value="ALL">Tous</option>
      {groupes.map(g => <option key={g} value={g}>{g}</option>)}
     </select>
    </div>
   </div>

   <table className="table table-striped">
    <thead>
     <tr>
      <th>CEF</th>
      <th>Nom</th>
      <th>Prénom</th>
      <th>Groupe</th>
      <th>Note Disc.</th>
      <th>Absence?</th>
      <th>Action</th>
     </tr>
    </thead>
    <tbody>
     {filtered.map(s => (
      <tr key={s.cef}>
       <td>{s.cef}</td>
       <td>{s.nom}</td>
       <td>{s.prenom}</td>
       <td>{s.groupe}</td>
       <td>{s.noteDiscipline}</td>
       <td>
        <input type="checkbox" checked={!!checked[s.cef]} onChange={() => toggle(s.cef)} />
       </td>
       <td>
        <button className="btn btn-sm btn-success" onClick={() =>
handleSaveAbsence(s)}>Save</button>
       </td>
      </tr>
     ))}
```

```
      </tbody>
    </table>

    <div className="mt-3">
      <h5>Absences récentes (tous groupes)</h5>
      <ul>
        {absences.slice().reverse().slice(0,5).map(a => (
          <li key={a.id}>{a.date} — {a.cef} ({a.groupe})</li>
        ))}
      </ul>
    </div>
  </div>
 );
}

// Absences component: show absences filtered by group and sorted by date
function Absences() {
 const absences = useSelector(state => state.absences.list);
 const filter = useSelector(state => state.absences.filterGroupe);
 const stagiaires = useSelector(state => state.stagiaires.list);
 const dispatch = useDispatch();

 // distinct groupes (from stagiaires)
 const groupes = React.useMemo(() => distinctGroups(stagiaires), [stagiaires]);

 const onSelectGroup = (e) => {
  dispatch(setFilterGroupeAbs(e.target.value));
  dispatch(setFilterGroupe(e.target.value));
 };

 const filtered = React.useMemo(() => {
  const list = filter === 'ALL' ? absences : absences.filter(a => a.groupe === filter);
  // sort by date desc
  return list.slice().sort((a,b) => new Date(b.date) - new Date(a.date));
 }, [absences, filter]);

 return (
  <div className="container">
    <h2>Liste des absences</h2>
    <div className="mb-3 row">
      <label className="col-sm-2 col-form-label">Filtrer par groupe</label>
      <div className="col-sm-4">
        <select className="form-select" value={filter} onChange={onSelectGroup}>
          <option value="ALL">Tous</option>
          {groupes.map(g => <option key={g} value={g}>{g}</option>)}
        </select>
      </div>
    </div>

    <table className="table">
      <thead>
        <tr>
```

```
       <th>Date</th>
       <th>CEF</th>
       <th>Stagiaire</th>
       <th>Groupe</th>
      </tr>
     </thead>
     <tbody>
      {filtered.map(a => {
       const st = stagiaires.find(s => s.cef === a.cef) || {};
       return (
        <tr key={a.id}>
         <td>{a.date}</td>
         <td>{a.cef}</td>
         <td>{st.nom ? `${st.nom} ${st.prenom}` : '—'}</td>
         <td>{a.groupe}</td>
        </tr>
       );
      })}
     </tbody>
    </table>
   </div>
 );
}

// Home page
function Home() {
 return (
   <div className="container">
    <h1>Bienvenue</h1>
    <p>Application de gestion des stagiaires et des absences (démo).</p>
   </div>
 );
}

// Main App
export default function AppRoot() {
 return (
   <Provider store={store}>
    <Router>
     <AppNavbar />
     <Routes>
      <Route path="/" element={<Home />} />
      <Route path="/stagiaires" element={<Stagiaires />} />
      <Route path="/absences" element={<Absences />} />
     </Routes>
    </Router>
   </Provider>
 );
}
```

```
// ================== src/index.js ===================
// In your real project, create src/index.js like this:
// import React from 'react';
// import { createRoot } from 'react-dom/client';
// import AppRoot from './App';
// createRoot(document.getElementById('root')).render(<AppRoot />);

// End of file
```

Analyser les données avec Sonareqube :

1. Installer **SonarQube** avec Docker

```
2. docker run -d --name sonarqube \
    -p 9000:9000 \
   sonarqube:latest
```

3. Connexion et configuration initiale

**Login** : admin

**Mot de passe** : admin

4. Installer et configurer **Sonar Scanner CLI**

```
npm install -g sonarqube-scanner
```

5. Au niveau de votre projet .properties
6. Lancer le scan (chercher la commande)
7. Analyser le dashboard (pour savoir le nombre de : bugs , vulnérabilité , code smells,…..)