

# Cours MERN - Semaine 7

## Maîtriser les Hooks Essentiels React

`useState, useEffect et useReducer`

Abdelweheb GUEDDES & Mohamed Ben Jazia / Ecole Polytechnique Sousse

8 novembre 2025

## Table des matières

<b>1 Objectifs Pédagogiques</b>	<b>2</b>
<b>2 Partie 1 : Concepts Essentiels (30 min)</b>	<b>2</b>
2.1 Hook 1 : <code>useState</code> - Ajouter de la mémoire . . . . .	2
2.1.1 L'Immutabilité : La Règle d'Or . . . . .	2
2.2 Hook 2 : <code>useEffect</code> - Les effets de bord . . . . .	3
2.3 Hook 3 : <code>useReducer</code> - État complexe . . . . .	3
<b>3 Partie 2 : Projets Pratiques (2h30)</b>	<b>5</b>
3.1 Projet 1 : Gestionnaire de Tâches avec <code>useReducer</code> (45 min) . . . . .	5
3.1.1 Étape 1 : Créer le Reducer . . . . .	5
3.1.2 Étape 2 : Le Composant Principal . . . . .	6
3.2 Projet 2 : Liste de Courses avec Timer (45 min) . . . . .	11
3.2.1 Code Complet . . . . .	11
3.3 Projet 3 : Blog Interactif avec Tri et Recherche (60 min) . . . . .	18
3.3.1 Code Complet de l'Application . . . . .	18
<b>4 Récapitatif et Concepts Clés</b>	<b>26</b>
<b>5 Travail à Rendre</b>	<b>27</b>
<b>6 Ressources</b>	<b>27</b>
6.1 Documentation Officielle . . . . .	27
6.2 Outils . . . . .	28

# 1 Objectifs Pédagogiques

**L'objectif central de cette séance** est de maîtriser les trois hooks fondamentaux de React à travers des projets pratiques concrets.

À la fin de cette séance, vous serez capable de :

- Utiliser useState pour gérer l'état local simple et complexe
- Maîtriser useEffect pour les effets de bord (API, timers, localStorage)
- Implémenter useReducer pour organiser la logique d'état complexe
- Appliquer l'immutabilité correctement dans toutes les mises à jour d'état
- Créer des applications interactives complètes avec persistance de données

## 2 Partie 1 : Concepts Essentiels (30 min)

### 2.1 Hook 1 : useState - Ajouter de la mémoire

useState permet d'ajouter de l'état à un composant fonctionnel. React conserve cet état entre les rendus.

**Syntaxe de base :**

```
1 const [ state , setState ] = useState ( initialValue ) ;  
2  
3 // Exemples  
4 const [ count , setCount ] = useState ( 0 ) ;  
5 const [ name , setName ] = useState ( '' ) ;  
6 const [ items , setItems ] = useState ( [ ] ) ;  
7 const [ user , setUser ] = useState ( { name : '' , age : 0 } ) ;
```

#### 2.1.1 L'Immutabilité : La Règle d'Or

En React, l'état ne doit **jamais** être modifié directement.

**INCORRECT - Mutation directe :**

```
1 // Objets  
2 user . name = " Bob " ;  
3 setUser ( user ) ; // React ne détecte pas le changement  
4  
5 // Tableaux  
6 items . push ( newItem ) ;  
7 setItems ( items ) ; // React ne détecte pas le changement
```

**CORRECT - Immutabilité :**

```
1 // Objets  
2 setUser ( { ... user , name : " Bob " } ) ; // Nouvel objet  
3  
4 // Tableaux  
5 setItems ( [ ... items , newItem ] ) ; // Nouveau tableau
```

### Opérations immutables sur les tableaux :

Opération	Méthode immutable
Ajouter	[...arr, newItem]
Supprimer	arr.filter((_,i) => i !== index)
Modifier	arr.map((item,i) => i === index ? newItem : item)

## 2.2 Hook 2 : useEffect - Les effets de bord

`useEffect` permet d'exécuter du code après le rendu pour synchroniser avec des systèmes externes.

### Syntaxe :

```

1  useEffect(() => {
2      // Code à exécuter
3
4      return () => {
5          // Nettoyage (optionnel)
6      };
7 }, [ dependencies ]);
```

### Les trois cas d'usage :

1. Une seule fois au montage : `useEffect(() => {...}, [])`
2. Quand une valeur change : `useEffect(() => {...}, [value])`
3. À chaque rendu : `useEffect(() => {...})` (rarement utilisé)

### Exemples courants :

- Charger des données depuis une API
- Créer un timer ou intervalle
- Sauvegarder dans localStorage
- S'abonner à des événements

## 2.3 Hook 3 : useReducer - État complexe

Quand l'état devient complexe (plusieurs sous-valeurs liées), `useReducer` offre une meilleure organisation.

### Syntaxe :

```

1  const [state, dispatch] = useReducer(reducer, initialState);
2
3 // Fonction reducer
4 function reducer(state, action) {
5     switch (action.type) {
6         case 'ACTION_NAME':
7             return {...state, /* modifications */};
```

```
8     default :
9         return state ;
10    }
11 }
12
13 // Utilisation
14 dispatch({ type: 'ACTION_NAME', payload: value });
```

### Quand utiliser useReducer ?

- Plusieurs sous-valeurs d'état liées
- Logique de mise à jour complexe
- Transitions d'état multiples

### 3 Partie 2 : Projets Pratiques (2h30)

#### 3.1 Projet 1 : Gestionnaire de Tâches avec useReducer (45 min)

**Objectif :** Créer une Todo List complète utilisant `useReducer` pour gérer l'état.

**Fonctionnalités :**

- Ajouter des tâches
- Marquer comme complétées
- Supprimer des tâches
- Filtrer (toutes / actives / complétées)
- Persistance avec localStorage

##### 3.1.1 Étape 1 : Crée le Reducer

```

1  export const initialState = {
2      todos: [] ,
3      filter: 'all' // 'all', 'active', 'completed'
4  };
5
6  export function todoReducer(state, action) {
7      switch (action.type) {
8          case 'ADD_TODO':
9              return {
10                  ...state,
11                  todos: [
12                      {
13                          id: Date.now(),
14                          text: action.payload,
15                          completed: false,
16                          createdAt: new Date().toISOString()
17                      },
18                      ...state.todos
19                  ]
20              };
21
22          case 'TOGGLE_TODO':
23              return {
24                  ...state,
25                  todos: state.todos.map(todo =>
26                      todo.id === action.payload
27                          ? {...todo, completed: !todo.completed}
28                          : todo
29                  )
30              };
31
32          case 'DELETE_TODO':
33              return {
34                  ...state,
35                  todos: state.todos.filter(todo =>

```

```

36             todo.id !== action.payload
37         )
38     };
39
40     case 'SET_FILTER':
41         return {
42             ...state,
43             filter: action.payload
44         };
45
46     case 'LOAD_TODOS':
47         return {
48             ...state,
49             todos: action.payload
50         };
51
52     default:
53         return state;
54     }
55 }
```

Listing 1 – src/reducers/todoReducer.js

### 3.1.2 Étape 2 : Le Composant Principal

```

1 import { useReducer, useState, useEffect } from 'react';
2 import { todoReducer, initialState } from './reducers/todoReducer';
3
4 function App() {
5     const [state, dispatch] = useReducer(todoReducer, initialState);
6     const [input, setInput] = useState('');
7
8     // Charger depuis localStorage au montage
9     useEffect(() => {
10         const saved = localStorage.getItem('todos');
11         if (saved) {
12             dispatch({
13                 type: 'LOAD_TODOS',
14                 payload: JSON.parse(saved)
15             });
16         }
17     }, []);
18
19     // Sauvegarder dans localStorage à chaque modification
20     useEffect(() => {
21         if (state.todos.length > 0) {
22             localStorage.setItem('todos', JSON.stringify(state.todos))
23         }
24     }, [state.todos]);
25 }
```

```
26 const addTodo = () => {
27   if (input.trim()) {
28     dispatch({ type: 'ADD_TODO', payload: input });
29     setInput('');
30   }
31 };
32
33 // Filtrer les todos
34 const filteredTodos = state.todos.filter(todo => {
35   if (state.filter === 'active') return !todo.completed;
36   if (state.filter === 'completed') return todo.completed;
37   return true;
38 }) ;
39
40 return (
41   <div style={{
42     maxWidth: '600px',
43     margin: '50px auto',
44     padding: '20px',
45     fontFamily: 'Arial, sans-serif'
46   }>
47     <h1 style={{ textAlign: 'center', color: '#2c3e50' }}>
48       Ma Todo List
49     </h1>
50
51   /* Formulaire d'ajout */
52   <div style={{
53     display: 'flex',
54     gap: '10px',
55     marginBottom: '20px'
56   }>
57     <input
58       type="text"
59       value={input}
60       onChange={(e) => setInput(e.target.value)}
61       onKeyPress={(e) => e.key === 'Enter' && addTodo()}
62     }
63     placeholder="Nouvelle tâche..."
64     style={{
65       flex: 1,
66       padding: '12px',
67       fontSize: '16px',
68       border: '2px solid #ddd',
69       borderRadius: '5px'
70     }}
71   />
72   <button
73     onClick={addTodo}
74     style={{
75       padding: '12px 24px',
76       backgroundColor: '#27ae60',
```

```

76         color: 'white',
77         border: 'none',
78         borderRadius: '5px',
79         cursor: 'pointer',
80         fontSize: '16px',
81         fontWeight: 'bold'
82     } }
83     >
84     Ajouter
85     </button>
86 </div>

87     {/* Filtres */}
88     <div style={{
89         display: 'flex',
90         gap: '10px',
91         marginBottom: '20px',
92         justifyContent: 'center'
93     }}>
94     {[ 'all', 'active', 'completed' ].map( filter => (
95         <button
96             key={filter}
97             onClick={() => dispatch({
98                 type: 'SET_FILTER',
99                 payload: filter
100            })}
101            style={{
102                padding: '8px 16px',
103                backgroundColor: state.filter === filter
104                    ? '#3498db'
105                    : '#ecf0f1',
106                color: state.filter === filter
107                    ? 'white'
108                    : '#2c3e50',
109                border: 'none',
110                borderRadius: '5px',
111                cursor: 'pointer',
112                fontWeight: state.filter === filter
113                    ? 'bold'
114                    : 'normal'
115            }}
116        >
117        { filter === 'all' ? 'Toutes' :
118            filter === 'active' ? 'Actives' :
119                'Complété
120        es' }
121        </button>
122    ))}
123 </div>

124     {/* Liste des tâches */}
125 <div>
```

```
126     { filteredTodos.length === 0 ? (
127         <p style={{{
128             textAlign: 'center',
129             color: '#95a5a6',
130             padding: '40px'
131         }}>
132             Aucune tâche à afficher
133         </p>
134     ) : (
135         filteredTodos.map(todo => (
136             <div
137                 key={todo.id}
138                 style={{
139                     display: 'flex',
140                     alignItems: 'center',
141                     padding: '15px',
142                     marginBottom: '10px',
143                     backgroundColor: todo.completed
144                     ? '#d5f4e6'
145                     : 'white',
146                     border: '1px solid #ddd',
147                     borderRadius: '5px'
148                 }}
149             >
150                 <input
151                     type="checkbox"
152                     checked={todo.completed}
153                     onChange={() => dispatch({
154                         type: 'TOGGLE_TODO',
155                         payload: todo.id
156                     })}
157                     style={{
158                         marginRight: '15px',
159                         width: '20px',
160                         height: '20px',
161                         cursor: 'pointer'
162                     }}
163                 />
164                 <span style={{
165                     flex: 1,
166                     fontSize: '16px',
167                     textDecoration: todo.completed
168                     ? 'line-through'
169                     : 'none',
170                     color: todo.completed
171                     ? '#95a5a6'
172                     : '#2c3e50'
173                 }}>
174                     {todo.text}
175                 </span>
176                 <button>
```

```

177         onClick={() => dispatch({
178             type: 'DELETE_TODO',
179             payload: todo.id
180         })}
181         style={{
182             padding: '6px 12px',
183             backgroundColor: '#e74c3c',
184             color: 'white',
185             border: 'none',
186             borderRadius: '3px',
187             cursor: 'pointer'
188         }}
189     >
190         Supprimer
191     </button>
192 </div>
193 )
194 )
195 </div>
196
197 /* Statistiques */
198 <div style={{
199     marginTop: '20px',
200     textAlign: 'center',
201     padding: '15px',
202     backgroundColor: '#ecf0f1',
203     borderRadius: '5px'
204 }}>
205     <strong>Total :</strong> {state.todos.length} | 
206     <strong> Complétées :</strong> {
207         state.todos.filter(t => t.completed).length
208     }
209 </div>
210 </div>
211 );
212 }
213
214 export default App;

```

Listing 2 – src/App.jsx

**Points clés de ce projet :**

- **useReducer** centralise toute la logique d'état
- **useEffect** gère la persistance localStorage
- **useState** pour l'input du formulaire
- **Immutabilité** respectée dans tous les cas du reducer

## 3.2 Projet 2 : Liste de Courses avec Timer (45 min)

**Objectif :** Créer une liste de courses avec un timer Pomodoro intégré.

**Compétences :**

- useState avec différents types
- useEffect avec nettoyage (timer)
- Persistance localStorage
- Gestion de formulaire

### 3.2.1 Code Complet

```

1 import { useState, useEffect } from 'react';
2
3 function ShoppingListApp() {
4     // Etats pour la liste
5     const [items, setItems] = useState([]);
6     const [inputValue, setInputValue] = useState('');
7     const [inputPrice, setInputPrice] = useState('');
8
9     // Etats pour le timer Pomodoro
10    const [minutes, setMinutes] = useState(25);
11    const [seconds, setSeconds] = useState(0);
12    const [isActive, setIsActive] = useState(false);
13
14    // Charger la liste depuis localStorage
15    useEffect(() => {
16        const saved = localStorage.getItem('shopping-list');
17        if (saved) {
18            setItems(JSON.parse(saved));
19        }
20    }, []);
21
22    // Sauvegarder la liste
23    useEffect(() => {
24        localStorage.setItem('shopping-list', JSON.stringify(items));
25    }, [items]);
26
27    // Timer Pomodoro
28    useEffect(() => {
29        let interval = null;
30
31        if (isActive) {
32            interval = setInterval(() => {
33                if (seconds === 0) {
34                    if (minutes === 0) {
35                        setIsActive(false);
36                        alert('Temps de course écouté !');
37                    } else {
38                        setMinutes(minutes - 1);
39                        setSeconds(59);
40                    }
41                } else {
42                    setSeconds(seconds - 1);
43                }
44            }, 1000);
45        }
46    }, [isActive]);
47
48    // Gestion du formulaire
49    const handleInputChange = (e) => {
50        setInputValue(e.target.value);
51    };
52
53    const handleInputPriceChange = (e) => {
54        setInputPrice(e.target.value);
55    };
56
57    const handleAddItem = () => {
58        if (inputValue === '') return;
59
60        const newItem = {
61            name: inputValue,
62            price: inputPrice,
63            quantity: 1
64        };
65
66        setItems([newItem, ...items]);
67        setInputValue('');
68        setInputPrice('');
69    };
70
71    const handleDeleteItem = (index) => {
72        const newItems = [...items];
73        newItems.splice(index, 1);
74        setItems(newItems);
75    };
76
77    const handleEditItem = (index, newName) => {
78        const newItems = [...items];
79        newItems[index].name = newName;
80        setItems(newItems);
81    };
82
83    const handleEditPrice = (index, newPrice) => {
84        const newItems = [...items];
85        newItems[index].price = newPrice;
86        setItems(newItems);
87    };
88
89    const handleEditQuantity = (index, newQuantity) => {
90        const newItems = [...items];
91        newItems[index].quantity = newQuantity;
92        setItems(newItems);
93    };
94
95    const handleReset = () => {
96        setItems([]);
97    };
98
99    const handlePomodoroStart = () => {
100        setIsActive(true);
101    };
102
103    const handlePomodoroStop = () => {
104        setIsActive(false);
105    };
106
107    const handlePomodoroReset = () => {
108        setMinutes(25);
109        setSeconds(0);
110    };
111
112    const handlePomodoroPause = () => {
113        clearInterval(interval);
114    };
115
116    const handlePomodoroResume = () => {
117        if (interval) {
118            return;
119        }
120        setIsActive(true);
121        interval = setInterval(() => {
122            if (seconds === 0) {
123                if (minutes === 0) {
124                    setIsActive(false);
125                    alert('Temps de course écouté !');
126                } else {
127                    setMinutes(minutes - 1);
128                    setSeconds(59);
129                }
130            } else {
131                setSeconds(seconds - 1);
132            }
133        }, 1000);
134    };
135
136    const handlePomodoroNext = () => {
137        if (minutes === 0) {
138            if (seconds === 0) {
139                setIsActive(false);
140                alert('Temps de course écouté !');
141            } else {
142                setMinutes(25);
143                setSeconds(seconds - 1);
144            }
145        } else {
146            setMinutes(minutes - 1);
147            setSeconds(59);
148        }
149    };
150
151    const handlePomodoroPrevious = () => {
152        if (minutes === 0) {
153            if (seconds === 0) {
154                setIsActive(false);
155                alert('Temps de course écouté !');
156            } else {
157                setMinutes(25);
158                setSeconds(seconds + 1);
159            }
160        } else {
161            setMinutes(minutes + 1);
162            setSeconds(59);
163        }
164    };
165
166    const handlePomodoroReset = () => {
167        setMinutes(25);
168        setSeconds(0);
169    };
170
171    const handlePomodoroPause = () => {
172        clearInterval(interval);
173    };
174
175    const handlePomodoroResume = () => {
176        if (interval) {
177            return;
178        }
179        setIsActive(true);
180        interval = setInterval(() => {
181            if (seconds === 0) {
182                if (minutes === 0) {
183                    setIsActive(false);
184                    alert('Temps de course écouté !');
185                } else {
186                    setMinutes(minutes - 1);
187                    setSeconds(59);
188                }
189            } else {
190                setSeconds(seconds - 1);
191            }
192        }, 1000);
193    };
194
195    const handlePomodoroNext = () => {
196        if (minutes === 0) {
197            if (seconds === 0) {
198                setIsActive(false);
199                alert('Temps de course écouté !');
200            } else {
201                setMinutes(25);
202                setSeconds(seconds - 1);
203            }
204        } else {
205            setMinutes(minutes - 1);
206            setSeconds(59);
207        }
208    };
209
210    const handlePomodoroPrevious = () => {
211        if (minutes === 0) {
212            if (seconds === 0) {
213                setIsActive(false);
214                alert('Temps de course écouté !');
215            } else {
216                setMinutes(25);
217                setSeconds(seconds + 1);
218            }
219        } else {
220            setMinutes(minutes + 1);
221            setSeconds(59);
222        }
223    };
224
225    const handlePomodoroReset = () => {
226        setMinutes(25);
227        setSeconds(0);
228    };
229
230    const handlePomodoroPause = () => {
231        clearInterval(interval);
232    };
233
234    const handlePomodoroResume = () => {
235        if (interval) {
236            return;
237        }
238        setIsActive(true);
239        interval = setInterval(() => {
240            if (seconds === 0) {
241                if (minutes === 0) {
242                    setIsActive(false);
243                    alert('Temps de course écouté !');
244                } else {
245                    setMinutes(minutes - 1);
246                    setSeconds(59);
247                }
248            } else {
249                setSeconds(seconds - 1);
250            }
251        }, 1000);
252    };
253
254    const handlePomodoroNext = () => {
255        if (minutes === 0) {
256            if (seconds === 0) {
257                setIsActive(false);
258                alert('Temps de course écouté !');
259            } else {
260                setMinutes(25);
261                setSeconds(seconds - 1);
262            }
263        } else {
264            setMinutes(minutes - 1);
265            setSeconds(59);
266        }
267    };
268
269    const handlePomodoroPrevious = () => {
270        if (minutes === 0) {
271            if (seconds === 0) {
272                setIsActive(false);
273                alert('Temps de course écouté !');
274            } else {
275                setMinutes(25);
276                setSeconds(seconds + 1);
277            }
278        } else {
279            setMinutes(minutes + 1);
280            setSeconds(59);
281        }
282    };
283
284    const handlePomodoroReset = () => {
285        setMinutes(25);
286        setSeconds(0);
287    };
288
289    const handlePomodoroPause = () => {
290        clearInterval(interval);
291    };
292
293    const handlePomodoroResume = () => {
294        if (interval) {
295            return;
296        }
297        setIsActive(true);
298        interval = setInterval(() => {
299            if (seconds === 0) {
300                if (minutes === 0) {
301                    setIsActive(false);
302                    alert('Temps de course écouté !');
303                } else {
304                    setMinutes(minutes - 1);
305                    setSeconds(59);
306                }
307            } else {
308                setSeconds(seconds - 1);
309            }
310        }, 1000);
311    };
312
313    const handlePomodoroNext = () => {
314        if (minutes === 0) {
315            if (seconds === 0) {
316                setIsActive(false);
317                alert('Temps de course écouté !');
318            } else {
319                setMinutes(25);
320                setSeconds(seconds - 1);
321            }
322        } else {
323            setMinutes(minutes - 1);
324            setSeconds(59);
325        }
326    };
327
328    const handlePomodoroPrevious = () => {
329        if (minutes === 0) {
330            if (seconds === 0) {
331                setIsActive(false);
332                alert('Temps de course écouté !');
333            } else {
334                setMinutes(25);
335                setSeconds(seconds + 1);
336            }
337        } else {
338            setMinutes(minutes + 1);
339            setSeconds(59);
340        }
341    };
342
343    const handlePomodoroReset = () => {
344        setMinutes(25);
345        setSeconds(0);
346    };
347
348    const handlePomodoroPause = () => {
349        clearInterval(interval);
350    };
351
352    const handlePomodoroResume = () => {
353        if (interval) {
354            return;
355        }
356        setIsActive(true);
357        interval = setInterval(() => {
358            if (seconds === 0) {
359                if (minutes === 0) {
360                    setIsActive(false);
361                    alert('Temps de course écouté !');
362                } else {
363                    setMinutes(minutes - 1);
364                    setSeconds(59);
365                }
366            } else {
367                setSeconds(seconds - 1);
368            }
369        }, 1000);
370    };
371
372    const handlePomodoroNext = () => {
373        if (minutes === 0) {
374            if (seconds === 0) {
375                setIsActive(false);
376                alert('Temps de course écouté !');
377            } else {
378                setMinutes(25);
379                setSeconds(seconds - 1);
380            }
381        } else {
382            setMinutes(minutes - 1);
383            setSeconds(59);
384        }
385    };
386
387    const handlePomodoroPrevious = () => {
388        if (minutes === 0) {
389            if (seconds === 0) {
390                setIsActive(false);
391                alert('Temps de course écouté !');
392            } else {
393                setMinutes(25);
394                setSeconds(seconds + 1);
395            }
396        } else {
397            setMinutes(minutes + 1);
398            setSeconds(59);
399        }
400    };
401
402    const handlePomodoroReset = () => {
403        setMinutes(25);
404        setSeconds(0);
405    };
406
407    const handlePomodoroPause = () => {
408        clearInterval(interval);
409    };
410
411    const handlePomodoroResume = () => {
412        if (interval) {
413            return;
414        }
415        setIsActive(true);
416        interval = setInterval(() => {
417            if (seconds === 0) {
418                if (minutes === 0) {
419                    setIsActive(false);
420                    alert('Temps de course écouté !');
421                } else {
422                    setMinutes(minutes - 1);
423                    setSeconds(59);
424                }
425            } else {
426                setSeconds(seconds - 1);
427            }
428        }, 1000);
429    };
430
431    const handlePomodoroNext = () => {
432        if (minutes === 0) {
433            if (seconds === 0) {
434                setIsActive(false);
435                alert('Temps de course écouté !');
436            } else {
437                setMinutes(25);
438                setSeconds(seconds - 1);
439            }
440        } else {
441            setMinutes(minutes - 1);
442            setSeconds(59);
443        }
444    };
445
446    const handlePomodoroPrevious = () => {
447        if (minutes === 0) {
448            if (seconds === 0) {
449                setIsActive(false);
450                alert('Temps de course écouté !');
451            } else {
452                setMinutes(25);
453                setSeconds(seconds + 1);
454            }
455        } else {
456            setMinutes(minutes + 1);
457            setSeconds(59);
458        }
459    };
460
461    const handlePomodoroReset = () => {
462        setMinutes(25);
463        setSeconds(0);
464    };
465
466    const handlePomodoroPause = () => {
467        clearInterval(interval);
468    };
469
470    const handlePomodoroResume = () => {
471        if (interval) {
472            return;
473        }
474        setIsActive(true);
475        interval = setInterval(() => {
476            if (seconds === 0) {
477                if (minutes === 0) {
478                    setIsActive(false);
479                    alert('Temps de course écouté !');
480                } else {
481                    setMinutes(minutes - 1);
482                    setSeconds(59);
483                }
484            } else {
485                setSeconds(seconds - 1);
486            }
487        }, 1000);
488    };
489
490    const handlePomodoroNext = () => {
491        if (minutes === 0) {
492            if (seconds === 0) {
493                setIsActive(false);
494                alert('Temps de course écouté !');
495            } else {
496                setMinutes(25);
497                setSeconds(seconds - 1);
498            }
499        } else {
500            setMinutes(minutes - 1);
501            setSeconds(59);
502        }
503    };
504
505    const handlePomodoroPrevious = () => {
506        if (minutes === 0) {
507            if (seconds === 0) {
508                setIsActive(false);
509                alert('Temps de course écouté !');
510            } else {
511                setMinutes(25);
512                setSeconds(seconds + 1);
513            }
514        } else {
515            setMinutes(minutes + 1);
516            setSeconds(59);
517        }
518    };
519
520    const handlePomodoroReset = () => {
521        setMinutes(25);
522        setSeconds(0);
523    };
524
525    const handlePomodoroPause = () => {
526        clearInterval(interval);
527    };
528
529    const handlePomodoroResume = () => {
530        if (interval) {
531            return;
532        }
533        setIsActive(true);
534        interval = setInterval(() => {
535            if (seconds === 0) {
536                if (minutes === 0) {
537                    setIsActive(false);
538                    alert('Temps de course écouté !');
539                } else {
540                    setMinutes(minutes - 1);
541                    setSeconds(59);
542                }
543            } else {
544                setSeconds(seconds - 1);
545            }
546        }, 1000);
547    };
548
549    const handlePomodoroNext = () => {
550        if (minutes === 0) {
551            if (seconds === 0) {
552                setIsActive(false);
553                alert('Temps de course écouté !');
554            } else {
555                setMinutes(25);
556                setSeconds(seconds - 1);
557            }
558        } else {
559            setMinutes(minutes - 1);
560            setSeconds(59);
561        }
562    };
563
564    const handlePomodoroPrevious = () => {
565        if (minutes === 0) {
566            if (seconds === 0) {
567                setIsActive(false);
568                alert('Temps de course écouté !');
569            } else {
570                setMinutes(25);
571                setSeconds(seconds + 1);
572            }
573        } else {
574            setMinutes(minutes + 1);
575            setSeconds(59);
576        }
577    };
578
579    const handlePomodoroReset = () => {
580        setMinutes(25);
581        setSeconds(0);
582    };
583
584    const handlePomodoroPause = () => {
585        clearInterval(interval);
586    };
587
588    const handlePomodoroResume = () => {
589        if (interval) {
590            return;
591        }
592        setIsActive(true);
593        interval = setInterval(() => {
594            if (seconds === 0) {
595                if (minutes === 0) {
596                    setIsActive(false);
597                    alert('Temps de course écouté !');
598                } else {
599                    setMinutes(minutes - 1);
600                    setSeconds(59);
601                }
602            } else {
603                setSeconds(seconds - 1);
604            }
605        }, 1000);
606    };
607
608    const handlePomodoroNext = () => {
609        if (minutes === 0) {
610            if (seconds === 0) {
611                setIsActive(false);
612                alert('Temps de course écouté !');
613            } else {
614                setMinutes(25);
615                setSeconds(seconds - 1);
616            }
617        } else {
618            setMinutes(minutes - 1);
619            setSeconds(59);
620        }
621    };
622
623    const handlePomodoroPrevious = () => {
624        if (minutes === 0) {
625            if (seconds === 0) {
626                setIsActive(false);
627                alert('Temps de course écouté !');
628            } else {
629                setMinutes(25);
630                setSeconds(seconds + 1);
631            }
632        } else {
633            setMinutes(minutes + 1);
634            setSeconds(59);
635        }
636    };
637
638    const handlePomodoroReset = () => {
639        setMinutes(25);
640        setSeconds(0);
641    };
642
643    const handlePomodoroPause = () => {
644        clearInterval(interval);
645    };
646
647    const handlePomodoroResume = () => {
648        if (interval) {
649            return;
650        }
651        setIsActive(true);
652        interval = setInterval(() => {
653            if (seconds === 0) {
654                if (minutes === 0) {
655                    setIsActive(false);
656                    alert('Temps de course écouté !');
657                } else {
658                    setMinutes(minutes - 1);
659                    setSeconds(59);
660                }
661            } else {
662                setSeconds(seconds - 1);
663            }
664        }, 1000);
665    };
666
667    const handlePomodoroNext = () => {
668        if (minutes === 0) {
669            if (seconds === 0) {
670                setIsActive(false);
671                alert('Temps de course écouté !');
672            } else {
673                setMinutes(25);
674                setSeconds(seconds - 1);
675            }
676        } else {
677            setMinutes(minutes - 1);
678            setSeconds(59);
679        }
680    };
681
682    const handlePomodoroPrevious = () => {
683        if (minutes === 0) {
684            if (seconds === 0) {
685                setIsActive(false);
686                alert('Temps de course écouté !');
687            } else {
688                setMinutes(25);
689                setSeconds(seconds + 1);
690            }
691        } else {
692            setMinutes(minutes + 1);
693            setSeconds(59);
694        }
695    };
696
697    const handlePomodoroReset = () => {
698        setMinutes(25);
699        setSeconds(0);
700    };
701
702    const handlePomodoroPause = () => {
703        clearInterval(interval);
704    };
705
706    const handlePomodoroResume = () => {
707        if (interval) {
708            return;
709        }
710        setIsActive(true);
711        interval = setInterval(() => {
712            if (seconds === 0) {
713                if (minutes === 0) {
714                    setIsActive(false);
715                    alert('Temps de course écouté !');
716                } else {
717                    setMinutes(minutes - 1);
718                    setSeconds(59);
719                }
720            } else {
721                setSeconds(seconds - 1);
722            }
723        }, 1000);
724    };
725
726    const handlePomodoroNext = () => {
727        if (minutes === 0) {
728            if (seconds === 0) {
729                setIsActive(false);
730                alert('Temps de course écouté !');
731            } else {
732                setMinutes(25);
733                setSeconds(seconds - 1);
734            }
735        } else {
736            setMinutes(minutes - 1);
737            setSeconds(59);
738        }
739    };
740
741    const handlePomodoroPrevious = () => {
742        if (minutes === 0) {
743            if (seconds === 0) {
744                setIsActive(false);
745                alert('Temps de course écouté !');
746            } else {
747                setMinutes(25);
748                setSeconds(seconds + 1);
749            }
750        } else {
751            setMinutes(minutes + 1);
752            setSeconds(59);
753        }
754    };
755
756    const handlePomodoroReset = () => {
757        setMinutes(25);
758        setSeconds(0);
759    };
760
761    const handlePomodoroPause = () => {
762        clearInterval(interval);
763    };
764
765    const handlePomodoroResume = () => {
766        if (interval) {
767            return;
768        }
769        setIsActive(true);
770        interval = setInterval(() => {
771            if (seconds === 0) {
772                if (minutes === 0) {
773                    setIsActive(false);
774                    alert('Temps de course écouté !');
775                } else {
776                    setMinutes(minutes - 1);
777                    setSeconds(59);
778                }
779            } else {
780                setSeconds(seconds - 1);
781            }
782        }, 1000);
783    };
784
785    const handlePomodoroNext = () => {
786        if (minutes === 0) {
787            if (seconds === 0) {
788                setIsActive(false);
789                alert('Temps de course écouté !');
790            } else {
791                setMinutes(25);
792                setSeconds(seconds - 1);
793            }
794        } else {
795            setMinutes(minutes - 1);
796            setSeconds(59);
797        }
798    };
799
800    const handlePomodoroPrevious = () => {
801        if (minutes === 0) {
802            if (seconds === 0) {
803                setIsActive(false);
804                alert('Temps de course écouté !');
805            } else {
806                setMinutes(25);
807                setSeconds(seconds + 1);
808            }
809        } else {
810            setMinutes(minutes + 1);
811            setSeconds(59);
812        }
813    };
814
815    const handlePomodoroReset = () => {
816        setMinutes(25);
817        setSeconds(0);
818    };
819
820    const handlePomodoroPause = () => {
821        clearInterval(interval);
822    };
823
824    const handlePomodoroResume = () => {
825        if (interval) {
826            return;
827        }
828        setIsActive(true);
829        interval = setInterval(() => {
830            if (seconds === 0) {
831                if (minutes === 0) {
832                    setIsActive(false);
833                    alert('Temps de course écouté !');
834                } else {
835                    setMinutes(minutes - 1);
836                    setSeconds(59);
837                }
838            } else {
839                setSeconds(seconds - 1);
840            }
841        }, 1000);
842    };
843
844    const handlePomodoroNext = () => {
845        if (minutes === 0) {
846            if (seconds === 0) {
847                setIsActive(false);
848                alert('Temps de course écouté !');
849            } else {
850                setMinutes(25);
851                setSeconds(seconds - 1);
852            }
853        } else {
854            setMinutes(minutes - 1);
855            setSeconds(59);
856        }
857    };
858
859    const handlePomodoroPrevious = () => {
860        if (minutes === 0) {
861            if (seconds === 0) {
862                setIsActive(false);
863                alert('Temps de course écouté !');
864            } else {
865                setMinutes(25);
866                setSeconds(seconds + 1);
867            }
868        } else {
869            setMinutes(minutes + 1);
870            setSeconds(59);
871        }
872    };
873
874    const handlePomodoroReset = () => {
875        setMinutes(25);
876        setSeconds(0);
877    };
878
879    const handlePomodoroPause = () => {
880        clearInterval(interval);
881    };
882
883    const handlePomodoroResume = () => {
884        if (interval) {
885            return;
886        }
887        setIsActive(true);
888        interval = setInterval(() => {
889            if (seconds === 0) {
890                if (minutes === 0) {
891                    setIsActive(false);
892                    alert('Temps de course écouté !');
893                } else {
894                    setMinutes(minutes - 1);
895                    setSeconds(59);
896                }
897            } else {
898                setSeconds(seconds - 1);
899            }
900        }, 1000);
901    };
902
903    const handlePomodoroNext = () => {
904        if (minutes === 0) {
905            if (seconds === 0) {
906                setIsActive(false);
907                alert('Temps de course écouté !');
908            } else {
909                setMinutes(25);
910                setSeconds(seconds - 1);
911            }
912        } else {
913            setMinutes(minutes - 1);
914            setSeconds(59);
915        }
916    };
917
918    const handlePomodoroPrevious = () => {
919        if (minutes === 0) {
920            if (seconds === 0) {
921                setIsActive(false);
922                alert('Temps de course écouté !');
923            } else {
924                setMinutes(25);
925                setSeconds(seconds + 1);
926            }
927        } else {
928            setMinutes(minutes + 1);
929            setSeconds(59);
930        }
931    };
932
933    const handlePomodoroReset = () => {
934        setMinutes(25);
935        setSeconds(0);
936    };
937
938    const handlePomodoroPause = () => {
939        clearInterval(interval);
940    };
941
942    const handlePomodoroResume = () => {
943        if (interval) {
944            return;
945        }
946        setIsActive(true);
947        interval = setInterval(() => {
948            if (seconds === 0) {
949                if (minutes === 0) {
950                    setIsActive(false);
951                    alert('Temps de course écouté !');
952                } else {
953                    setMinutes(minutes - 1);
954                    setSeconds(59);
955                }
956            } else {
957                setSeconds(seconds - 1);
958            }
959        }, 1000);
960    };
961
962    const handlePomodoroNext = () => {
963        if (minutes === 0) {
964            if (seconds === 0) {
965                setIsActive(false);
966                alert('Temps de course écouté !');
967            } else {
968                setMinutes(25);
969                setSeconds(seconds - 1);
970            }
971        } else {
972            setMinutes(minutes - 1);
973            setSeconds(59);
974        }
975    };
976
977    const handlePomodoroPrevious = () => {
978        if (minutes === 0) {
979            if (seconds === 0) {
980                setIsActive(false);
981                alert('Temps de course écouté !');
982            } else {
983                setMinutes(25);
984                setSeconds(seconds + 1);
985            }
986        } else {
987            setMinutes(minutes + 1);
988            setSeconds(59);
989        }
990    };
991
992    const handlePomodoroReset = () => {
993        setMinutes(25);
994        setSeconds(0);
995    };
996
997    const handlePomodoroPause = () => {
998        clearInterval(interval);
999    };
1000
1001    const handlePomodoroResume = () => {
1002        if (interval) {
1003            return;
1004        }
1005        setIsActive(true);
1006        interval = setInterval(() => {
1007            if (seconds === 0) {
1008                if (minutes === 0) {
1009                    setIsActive(false);
1010                    alert('Temps de course écouté !');
1011                } else {
1012                    setMinutes(minutes - 1);
1013                    setSeconds(59);
1014                }
1015            } else {
1016                setSeconds(seconds - 1);
1017            }
1018        }, 1000);
1019    };
1020
1021    const handlePomodoroNext = () => {
1022        if (minutes === 0) {
1023            if (seconds === 0) {
1024                setIsActive(false);
1025                alert('Temps de course écouté !');
1026            } else {
1027                setMinutes(25);
1028                setSeconds(seconds - 1);
1029            }
1030        } else {
1031            setMinutes(minutes - 1);
1032            setSeconds(59);
1033        }
1034    };
1035
1036    const handlePomodoroPrevious = () => {
1037        if (minutes === 0) {
1038            if (seconds === 0) {
1039                setIsActive(false);
1040                alert('Temps de course écouté !');
1041            } else {
1042                setMinutes(25);
1043                setSeconds(seconds + 1);
1044            }
1045        } else {
1046            setMinutes(minutes + 1);
1047            setSeconds(59);
1048        }
1049    };
1050
1051    const handlePomodoroReset = () => {
1052        setMinutes(25);
1053        setSeconds(0);
1054    };
1055
1056    const handlePomodoroPause = () => {
1057        clearInterval(interval);
1058    };
1059
1060    const handlePomodoroResume = () => {
1061        if (interval) {
1062            return;
1063        }
1064        setIsActive(true);
1065        interval = setInterval(() => {
1066            if (seconds === 0) {
1067                if (minutes === 0) {
1068                    setIsActive(false);
1069                    alert('Temps de course écouté !');
1070                } else {
1071                    setMinutes(minutes - 1);
1072                    setSeconds(59);
1073                }
1074            } else {
1075                setSeconds(seconds - 1);
1076            }
1077        }, 1000);
1078    };
1079
1080    const handlePomodoroNext = () => {
1081        if (minutes === 0) {
1082            if (seconds === 0) {
1083                setIsActive(false);
1084                alert('Temps de course écouté !');
1085            } else {
1086                setMinutes(25);
1087                setSeconds(seconds - 1);
1088            }
1089        } else {
1090            setMinutes(minutes - 1);
1091            setSeconds(59);
1092        }
1093    };
1094
1095    const handlePomodoroPrevious = () => {
1096        if (minutes === 0) {
1097            if (seconds === 0) {
1098                setIsActive(false);
1099                alert('Temps de course écouté !');
1100            } else {
1101                setMinutes(25);
1102                setSeconds(seconds + 1);
1103            }
1104        } else {
1105            setMinutes(minutes + 1);
1106            setSeconds(59);
1107        }
1108    };
1109
1110    const handlePomodoroReset = () => {
1111        setMinutes(25);
1112        setSeconds(0);
1113    };
1114
1115    const handlePomodoroPause = () => {
1116        clearInterval(interval);
1117    };
1118
1119    const handlePomodoroResume = () => {
1120        if (interval) {
1121            return;
1122        }
1123        setIsActive(true);
1124        interval = setInterval(() => {
1125            if (seconds === 0) {
1126                if (minutes === 0) {
1127                    setIsActive(false);
1128                    alert('Temps de course écouté !');
1129                } else {
1130                    setMinutes(minutes - 1);
1131                    setSeconds(59);
1132                }
1133            } else {
1134                setSeconds(seconds - 1);
1135            }
1136        }, 1000);
1137    };
1138
1139    const handlePomodoroNext = () => {
1140        if (minutes === 0) {
1141            if (seconds === 0) {
1142                setIsActive(false);
1143                alert('Temps de course écouté !');
1144            } else {
1145                setMinutes(25);
1146                setSeconds(seconds - 1);
1147            }
1148        } else {
1149            setMinutes(minutes - 1);
1150            setSeconds(59);
1151        }
1152    };
1153
1154    const handlePomodoroPrevious = () => {
1155        if (minutes === 0) {
1156            if (seconds === 0) {
1157                setIsActive(false);
1158                alert('Temps de course écouté !');
1159            } else {
1160                setMinutes(25);
1161                setSeconds(seconds + 1);
1162            }
1163        } else {
1164            setMinutes(minutes + 1);
1165            setSeconds(59);
1166        }
1167    };
1168
1169    const handlePomodoroReset = () => {
1170        setMinutes(25);
1171        setSeconds(0);
1172    };
1173
1174    const handlePomodoroPause = () => {
1175        clearInterval(interval);
1176    };
1177
1178    const handlePomodoroResume = () => {
1179        if (interval) {
1180            return;
1181        }
1182        setIsActive(true);
1183       
```

```

40         }
41     } else {
42         setSeconds(seconds - 1);
43     }
44 }, 1000);
45 }

46 // Nettoyage : arrêter l'intervalle
47 return () => {
48     if (interval) clearInterval(interval);
49 };
50 , [isActive, minutes, seconds]);

52 const addItem = () => {
53     if (inputValue.trim()) {
54         setItems([
55             ...items,
56             {
57                 id: Date.now(),
58                 name: inputValue,
59                 price: inputPrice ? parseFloat(inputPrice) : 0,
60                 bought: false
61             }
62         ]);
63         setInputValue('');
64         setInputPrice('');
65     }
66 };
67 }

68 const toggleBought = (id) => {
69     setItems(items.map(item =>
70         item.id === id
71             ? {...item, bought: !item.bought}
72             : item
73     )));
74 };
75 }

76 const deleteItem = (id) => {
77     setItems(items.filter(item => item.id !== id));
78 };

79 const resetTimer = () => {
80     setIsActive(false);
81     setMinutes(25);
82     setSeconds(0);
83 };
84

85 const totalPrice = items.reduce((sum, item) => sum + item.price, 0);
86 const boughtCount = items.filter(item => item.bought).length;
87
88
89

```

```

90     return (
91       <div style={{
92         maxWidth: '700px',
93         margin: '50px auto',
94         padding: '20px'
95       }>
96         <h1 style={{ textAlign: 'center' }}>
97           Liste de Courses
98         </h1>
99
100        {/* Timer Pomodoro */}
101        <div style={{
102          backgroundColor: '#34495e',
103          color: 'white',
104          padding: '20px',
105          borderRadius: '10px',
106          marginBottom: '30px',
107          textAlign: 'center'
108        }>
109          <h3>Timer de Course</h3>
110          <div style={{
111            fontSize: '48px',
112            fontWeight: 'bold',
113            margin: '20px 0',
114            color: isActive ? '#e74c3c' : 'white'
115          }>
116            {String(minutes).padStart(2, '0')}: {String(seconds).padStart(2, '0')}
117          </div>
118          <div style={{ display: 'flex', gap: '10px',
119            justifyContent: 'center' }}>
120            <button
121              onClick={() => setIsActive(!isActive)}
122              style={{
123                padding: '10px 20px',
124                backgroundColor: isActive ? '#f39c12' : '#27ae60',
125                color: 'white',
126                border: 'none',
127                borderRadius: '5px',
128                cursor: 'pointer',
129                fontSize: '16px'
130              }}
131            >
132              {isActive ? 'Pause' : 'Démarrer'}
133            </button>
134            <button
135              onClick={resetTimer}
136              style={{
137                padding: '10px 20px',
138                backgroundColor: '#95a5a6',

```

```

139             color: 'white',
140             border: 'none',
141             borderRadius: '5px',
142             cursor: 'pointer',
143             fontSize: '16px'
144         )}
145     >
146         Reset
147         </button>
148     </div>
149 </div>
150
151     /* Formulaire d'ajout */
152 <div style={{{
153     backgroundColor: '#ecf0f1',
154     padding: '20px',
155     borderRadius: '10px',
156     marginBottom: '20px'
157 }}>
158     <div style={{ display: 'flex', gap: '10px',
159     marginBottom: '10px' }}>
160         <input
161             type="text"
162             value={inputValue}
163             onChange={(e) => setInputValue(e.target.value)
164             }
165             onKeyPress={(e) => e.key === 'Enter' &&
166             addItem() }
167             placeholder="Nom de l'article..."
168             style={{
169                 flex: 2,
170                 padding: '10px',
171                 fontSize: '16px',
172                 border: '1px solid #bdc3c7',
173                 borderRadius: '5px'
174             }}
175         />
176         <input
177             type="number"
178             value={inputPrice}
179             onChange={(e) => setInputPrice(e.target.value)
180             }
181             onKeyPress={(e) => e.key === 'Enter' &&
182             addItem() }
183             placeholder="Prix..."
184             step="0.01"
185             style={{
186                 flex: 1,
187                 padding: '10px',
188                 fontSize: '16px',
189                 border: '1px solid #bdc3c7',
190             }}
191     </div>
192 
```

```

185             borderRadius: '5px'
186         )}
187     />
188     <button
189         onClick={addItem}
190         style={{
191             padding: '10px 20px',
192             backgroundColor: '#3498db',
193             color: 'white',
194             border: 'none',
195             borderRadius: '5px',
196             cursor: 'pointer',
197             fontSize: '16px'
198         }}
199     >
200         Ajouter
201     </button>
202     </div>
203 </div>
204
205     /* Liste des articles */
206 <div>
207     {items.map(item => (
208         <div
209             key={item.id}
210             style={{
211                 display: 'flex',
212                 alignItems: 'center',
213                 padding: '15px',
214                 marginBottom: '10px',
215                 backgroundColor: item.bought ? '#d5f4e6'
216                 : 'white',
217                 border: '1px solid #ddd',
218                 borderRadius: '5px'
219             })
220         >
221             <input
222                 type="checkbox"
223                 checked={item.bought}
224                 onChange={() => toggleBought(item.id)}
225                 style={{
226                     marginRight: '15px',
227                     width: '20px',
228                     height: '20px'
229                 }}
230             />
231             <span style={{
232                 flex: 1,
233                 textDecoration: item.bought ? 'line-
234 through' : 'none',
235                 color: item.bought ? '#95a5a6' : '#2c3e50

```

```

    ,
    fontSize: '16px'
  }}>
  {item.name}
</span>
<span style={{
  fontWeight: 'bold',
  marginRight: '15px',
  color: '#27ae60'
}}>
  {item.price.toFixed(2)}DT
</span>
<button
  onClick={() => deleteItem(item.id)}
  style={{
    padding: '6px 12px',
    backgroundColor: '#e74c3c',
    color: 'white',
    border: 'none',
    borderRadius: '3px',
    cursor: 'pointer'
  }}
>
  Supprimer
</button>
</div>
))}

</div>

/* Statistiques */
<div style={{
  marginTop: '20px',
  padding: '15px',
  backgroundColor: '#3498db',
  color: 'white',
  borderRadius: '5px',
  display: 'flex',
  justifyContent: 'space-between'
}}>
<div>
  <strong>Total :</strong> {items.length} articles
| 
  <strong> Achetés :</strong> {boughtCount}
</div>
<div>
  <strong>Prix total :</strong> {totalPrice.toFixed
(2)}DT
</div>
</div>
</div>
);

```

```
282 }  
283  
284 export default ShoppingListApp;
```

Listing 3 – src/ShoppingListApp.jsx

**Concepts utilisés dans ce projet :**

1. **useState multiple** : Pour gérer plusieurs états indépendants
2. **useEffect avec cleanup** : Pour le timer (important !)
3. **useEffect avec dépendances** : Pour localStorage
4. **Calculs dérivés** : totalPrice et boughtCount calculés à la volée
5. **Inputs contrôlés** : text et number

### 3.3 Projet 3 : Blog Interactif avec Tri et Recherche (60 min)

**Objectif :** Reprendre le blog de la semaine précédente et le rendre complètement interactif.

**Nouvelles fonctionnalités :**

- Ajouter des articles via formulaire
- Liker / Disliker
- Recherche par titre et contenu
- Tri par likes ou date
- Filtrer par auteur
- Supprimer des articles
- Persistance complète

#### 3.3.1 Code Complet de l'Application

```

1 import { useState, useEffect } from 'react';
2
3 function BlogApp() {
4     // Etats principaux
5     const [articles, setArticles] = useState(() => {
6         const saved = localStorage.getItem('blog-articles');
7         if (saved) return JSON.parse(saved);
8         return [
9             {
10                 id: 1,
11                 title: "Introduction à React",
12                 author: "Alice",
13                 content: "React est une bibliothèque JavaScript...",
14                 likes: 5,
15                 createdAt: new Date().toISOString()
16             },
17             {
18                 id: 2,
19                 title: "Les Hooks en détail",
20                 author: "Bob",
21                 content: "Les hooks révolutionnent React...",
22                 likes: 3,
23                 createdAt: new Date().toISOString()
24             }
25         ];
26     });
27
28     // Etats de filtrage et tri
29     const [searchTerm, setSearchTerm] = useState('');
30     const [filterAuthor, setFilterAuthor] = useState('');
31     const [sortBy, setSortBy] = useState('date'); // 'date' ou 'likes'
32
33     // Etats du formulaire

```

```
34  const [showForm, setShowForm] = useState(false);
35  const [title, setTitle] = useState('');
36  const [author, setAuthor] = useState('');
37  const [content, setContent] = useState('');

38
39 // Sauvegarder dans localStorage
40 useEffect(() => {
41   localStorage.setItem('blog-articles', JSON.stringify(articles));
42 }, [articles]);

43
44 // Fonctions de gestion
45 const addArticle = () => {
46   if (!title.trim() || !author.trim() || !content.trim()) {
47     alert('Tous les champs sont obligatoires');
48     return;
49   }

50   const newArticle = {
51     id: Date.now(),
52     title,
53     author,
54     content,
55     likes: 0,
56     createdAt: new Date().toISOString()
57   };

58   setArticles([newArticle, ...articles]);
59   setTitle('');
60   setAuthor('');
61   setContent('');
62   setShowForm(false);
63 };

64
65 const handleLike = (id) => {
66   setArticles(articles.map(article =>
67     article.id === id
68       ? {...article, likes: article.likes + 1}
69       : article
70   ));
71 };
72

73
74 const handleDelete = (id) => {
75   if (window.confirm('Supprimer cet article ?')) {
76     setArticles(articles.filter(article => article.id !== id));
77   }
78 };
79

80
81 // Filtrage et tri
82 let filteredArticles = articles;
```

```

83
84 // Recherche
85 if (searchTerm) {
86     filteredArticles = filteredArticles.filter(article =>
87         article.title.toLowerCase().includes(searchTerm.
88         toLowerCase()) ||
89         article.content.toLowerCase().includes(searchTerm.
90         toLowerCase()))
91     );
92
93 // Filtre par auteur
94 if (filterAuthor) {
95     filteredArticles = filteredArticles.filter(article =>
96         article.author === filterAuthor
97     );
98
99 // Tri
100 filteredArticles = [...filteredArticles].sort((a, b) => {
101     if (sortBy === 'likes') {
102         return b.likes - a.likes;
103     }
104     return new Date(b.createdAt) - new Date(a.createdAt);
105 });
106
107 // Liste unique des auteurs
108 const authors = [...new Set(articles.map(a => a.author))];
109
110 return (
111     <div style={{
112         maxWidth: '900px',
113         margin: '0 auto',
114         padding: '20px'
115     }>
116         <h1 style={{ textAlign: 'center', color: '#2c3e50' }}>
117             Mon Blog React Interactif
118         </h1>
119
120         {/* Barre d'actions */}
121         <div style={{
122             backgroundColor: '#ecf0f1',
123             padding: '20px',
124             borderRadius: '10px',
125             marginBottom: '20px'
126         }>
127             <div style={{
128                 display: 'grid',
129                 gridTemplateColumns: '1fr 1fr 1fr auto',
130                 gap: '10px',
131                 marginBottom: '15px'
132             }>
133
134             <div style={{
135                 display: 'grid',
136                 gridTemplateColumns: '1fr 1fr 1fr auto',
137                 gap: '10px',
138                 marginBottom: '15px'
139             }>
140
141             <div style={{
142                 display: 'grid',
143                 gridTemplateColumns: '1fr 1fr 1fr auto',
144                 gap: '10px',
145                 marginBottom: '15px'
146             }>
147
148             <div style={{
149                 display: 'grid',
150                 gridTemplateColumns: '1fr 1fr 1fr auto',
151                 gap: '10px',
152                 marginBottom: '15px'
153             }>
154
155             <div style={{
156                 display: 'grid',
157                 gridTemplateColumns: '1fr 1fr 1fr auto',
158                 gap: '10px',
159                 marginBottom: '15px'
160             }>
161
162             <div style={{
163                 display: 'grid',
164                 gridTemplateColumns: '1fr 1fr 1fr auto',
165                 gap: '10px',
166                 marginBottom: '15px'
167             }>
168
169             <div style={{
170                 display: 'grid',
171                 gridTemplateColumns: '1fr 1fr 1fr auto',
172                 gap: '10px',
173                 marginBottom: '15px'
174             }>
175
176             <div style={{
177                 display: 'grid',
178                 gridTemplateColumns: '1fr 1fr 1fr auto',
179                 gap: '10px',
180                 marginBottom: '15px'
181             }>
182
183             <div style={{
184                 display: 'grid',
185                 gridTemplateColumns: '1fr 1fr 1fr auto',
186                 gap: '10px',
187                 marginBottom: '15px'
188             }>
189
190             <div style={{
191                 display: 'grid',
192                 gridTemplateColumns: '1fr 1fr 1fr auto',
193                 gap: '10px',
194                 marginBottom: '15px'
195             }>
196
197             <div style={{
198                 display: 'grid',
199                 gridTemplateColumns: '1fr 1fr 1fr auto',
200                 gap: '10px',
201                 marginBottom: '15px'
202             }>
203
204             <div style={{
205                 display: 'grid',
206                 gridTemplateColumns: '1fr 1fr 1fr auto',
207                 gap: '10px',
208                 marginBottom: '15px'
209             }>
210
211             <div style={{
212                 display: 'grid',
213                 gridTemplateColumns: '1fr 1fr 1fr auto',
214                 gap: '10px',
215                 marginBottom: '15px'
216             }>
217
218             <div style={{
219                 display: 'grid',
220                 gridTemplateColumns: '1fr 1fr 1fr auto',
221                 gap: '10px',
222                 marginBottom: '15px'
223             }>
224
225             <div style={{
226                 display: 'grid',
227                 gridTemplateColumns: '1fr 1fr 1fr auto',
228                 gap: '10px',
229                 marginBottom: '15px'
230             }>
231
232             <div style={{
233                 display: 'grid',
234                 gridTemplateColumns: '1fr 1fr 1fr auto',
235                 gap: '10px',
236                 marginBottom: '15px'
237             }>
238
239             <div style={{
240                 display: 'grid',
241                 gridTemplateColumns: '1fr 1fr 1fr auto',
242                 gap: '10px',
243                 marginBottom: '15px'
244             }>
245
246             <div style={{
247                 display: 'grid',
248                 gridTemplateColumns: '1fr 1fr 1fr auto',
249                 gap: '10px',
250                 marginBottom: '15px'
251             }>
252
253             <div style={{
254                 display: 'grid',
255                 gridTemplateColumns: '1fr 1fr 1fr auto',
256                 gap: '10px',
257                 marginBottom: '15px'
258             }>
259
260             <div style={{
261                 display: 'grid',
262                 gridTemplateColumns: '1fr 1fr 1fr auto',
263                 gap: '10px',
264                 marginBottom: '15px'
265             }>
266
267             <div style={{
268                 display: 'grid',
269                 gridTemplateColumns: '1fr 1fr 1fr auto',
270                 gap: '10px',
271                 marginBottom: '15px'
272             }>
273
274             <div style={{
275                 display: 'grid',
276                 gridTemplateColumns: '1fr 1fr 1fr auto',
277                 gap: '10px',
278                 marginBottom: '15px'
279             }>
280
281             <div style={{
282                 display: 'grid',
283                 gridTemplateColumns: '1fr 1fr 1fr auto',
284                 gap: '10px',
285                 marginBottom: '15px'
286             }>
287
288             <div style={{
289                 display: 'grid',
290                 gridTemplateColumns: '1fr 1fr 1fr auto',
291                 gap: '10px',
292                 marginBottom: '15px'
293             }>
294
295             <div style={{
296                 display: 'grid',
297                 gridTemplateColumns: '1fr 1fr 1fr auto',
298                 gap: '10px',
299                 marginBottom: '15px'
300             }>
301
302             <div style={{
303                 display: 'grid',
304                 gridTemplateColumns: '1fr 1fr 1fr auto',
305                 gap: '10px',
306                 marginBottom: '15px'
307             }>
308
309             <div style={{
310                 display: 'grid',
311                 gridTemplateColumns: '1fr 1fr 1fr auto',
312                 gap: '10px',
313                 marginBottom: '15px'
314             }>
315
316             <div style={{
317                 display: 'grid',
318                 gridTemplateColumns: '1fr 1fr 1fr auto',
319                 gap: '10px',
320                 marginBottom: '15px'
321             }>
322
323             <div style={{
324                 display: 'grid',
325                 gridTemplateColumns: '1fr 1fr 1fr auto',
326                 gap: '10px',
327                 marginBottom: '15px'
328             }>
329
330             <div style={{
331                 display: 'grid',
332                 gridTemplateColumns: '1fr 1fr 1fr auto',
333                 gap: '10px',
334                 marginBottom: '15px'
335             }>
336
337             <div style={{
338                 display: 'grid',
339                 gridTemplateColumns: '1fr 1fr 1fr auto',
340                 gap: '10px',
341                 marginBottom: '15px'
342             }>
343
344             <div style={{
345                 display: 'grid',
346                 gridTemplateColumns: '1fr 1fr 1fr auto',
347                 gap: '10px',
348                 marginBottom: '15px'
349             }>
350
351             <div style={{
352                 display: 'grid',
353                 gridTemplateColumns: '1fr 1fr 1fr auto',
354                 gap: '10px',
355                 marginBottom: '15px'
356             }>
357
358             <div style={{
359                 display: 'grid',
360                 gridTemplateColumns: '1fr 1fr 1fr auto',
361                 gap: '10px',
362                 marginBottom: '15px'
363             }>
364
365             <div style={{
366                 display: 'grid',
367                 gridTemplateColumns: '1fr 1fr 1fr auto',
368                 gap: '10px',
369                 marginBottom: '15px'
370             }>
371
372             <div style={{
373                 display: 'grid',
374                 gridTemplateColumns: '1fr 1fr 1fr auto',
375                 gap: '10px',
376                 marginBottom: '15px'
377             }>
378
379             <div style={{
380                 display: 'grid',
381                 gridTemplateColumns: '1fr 1fr 1fr auto',
382                 gap: '10px',
383                 marginBottom: '15px'
384             }>
385
386             <div style={{
387                 display: 'grid',
388                 gridTemplateColumns: '1fr 1fr 1fr auto',
389                 gap: '10px',
390                 marginBottom: '15px'
391             }>
392
393             <div style={{
394                 display: 'grid',
395                 gridTemplateColumns: '1fr 1fr 1fr auto',
396                 gap: '10px',
397                 marginBottom: '15px'
398             }>
399
399
400             <div style={{
401                 display: 'grid',
402                 gridTemplateColumns: '1fr 1fr 1fr auto',
403                 gap: '10px',
404                 marginBottom: '15px'
405             }>
406
407             <div style={{
408                 display: 'grid',
409                 gridTemplateColumns: '1fr 1fr 1fr auto',
410                 gap: '10px',
411                 marginBottom: '15px'
412             }>
413
414             <div style={{
415                 display: 'grid',
416                 gridTemplateColumns: '1fr 1fr 1fr auto',
417                 gap: '10px',
418                 marginBottom: '15px'
419             }>
420
421             <div style={{
422                 display: 'grid',
423                 gridTemplateColumns: '1fr 1fr 1fr auto',
424                 gap: '10px',
425                 marginBottom: '15px'
426             }>
427
428             <div style={{
429                 display: 'grid',
430                 gridTemplateColumns: '1fr 1fr 1fr auto',
431                 gap: '10px',
432                 marginBottom: '15px'
433             }>
434
435             <div style={{
436                 display: 'grid',
437                 gridTemplateColumns: '1fr 1fr 1fr auto',
438                 gap: '10px',
439                 marginBottom: '15px'
440             }>
441
442             <div style={{
443                 display: 'grid',
444                 gridTemplateColumns: '1fr 1fr 1fr auto',
445                 gap: '10px',
446                 marginBottom: '15px'
447             }>
448
449             <div style={{
450                 display: 'grid',
451                 gridTemplateColumns: '1fr 1fr 1fr auto',
452                 gap: '10px',
453                 marginBottom: '15px'
454             }>
455
456             <div style={{
457                 display: 'grid',
458                 gridTemplateColumns: '1fr 1fr 1fr auto',
459                 gap: '10px',
460                 marginBottom: '15px'
461             }>
462
463             <div style={{
464                 display: 'grid',
465                 gridTemplateColumns: '1fr 1fr 1fr auto',
466                 gap: '10px',
467                 marginBottom: '15px'
468             }>
469
470             <div style={{
471                 display: 'grid',
472                 gridTemplateColumns: '1fr 1fr 1fr auto',
473                 gap: '10px',
474                 marginBottom: '15px'
475             }>
476
477             <div style={{
478                 display: 'grid',
479                 gridTemplateColumns: '1fr 1fr 1fr auto',
480                 gap: '10px',
481                 marginBottom: '15px'
482             }>
483
484             <div style={{
485                 display: 'grid',
486                 gridTemplateColumns: '1fr 1fr 1fr auto',
487                 gap: '10px',
488                 marginBottom: '15px'
489             }>
490
491             <div style={{
492                 display: 'grid',
493                 gridTemplateColumns: '1fr 1fr 1fr auto',
494                 gap: '10px',
495                 marginBottom: '15px'
496             }>
497
498             <div style={{
499                 display: 'grid',
500                 gridTemplateColumns: '1fr 1fr 1fr auto',
501                 gap: '10px',
502                 marginBottom: '15px'
503             }>
504
505             <div style={{
506                 display: 'grid',
507                 gridTemplateColumns: '1fr 1fr 1fr auto',
508                 gap: '10px',
509                 marginBottom: '15px'
510             }>
511
512             <div style={{
513                 display: 'grid',
514                 gridTemplateColumns: '1fr 1fr 1fr auto',
515                 gap: '10px',
516                 marginBottom: '15px'
517             }>
518
519             <div style={{
520                 display: 'grid',
521                 gridTemplateColumns: '1fr 1fr 1fr auto',
522                 gap: '10px',
523                 marginBottom: '15px'
524             }>
525
526             <div style={{
527                 display: 'grid',
528                 gridTemplateColumns: '1fr 1fr 1fr auto',
529                 gap: '10px',
530                 marginBottom: '15px'
531             }>
532
533             <div style={{
534                 display: 'grid',
535                 gridTemplateColumns: '1fr 1fr 1fr auto',
536                 gap: '10px',
537                 marginBottom: '15px'
538             }>
539
540             <div style={{
541                 display: 'grid',
542                 gridTemplateColumns: '1fr 1fr 1fr auto',
543                 gap: '10px',
544                 marginBottom: '15px'
545             }>
546
547             <div style={{
548                 display: 'grid',
549                 gridTemplateColumns: '1fr 1fr 1fr auto',
550                 gap: '10px',
551                 marginBottom: '15px'
552             }>
553
554             <div style={{
555                 display: 'grid',
556                 gridTemplateColumns: '1fr 1fr 1fr auto',
557                 gap: '10px',
558                 marginBottom: '15px'
559             }>
560
561             <div style={{
562                 display: 'grid',
563                 gridTemplateColumns: '1fr 1fr 1fr auto',
564                 gap: '10px',
565                 marginBottom: '15px'
566             }>
567
568             <div style={{
569                 display: 'grid',
570                 gridTemplateColumns: '1fr 1fr 1fr auto',
571                 gap: '10px',
572                 marginBottom: '15px'
573             }>
574
575             <div style={{
576                 display: 'grid',
577                 gridTemplateColumns: '1fr 1fr 1fr auto',
578                 gap: '10px',
579                 marginBottom: '15px'
580             }>
581
582             <div style={{
583                 display: 'grid',
584                 gridTemplateColumns: '1fr 1fr 1fr auto',
585                 gap: '10px',
586                 marginBottom: '15px'
587             }>
588
589             <div style={{
590                 display: 'grid',
591                 gridTemplateColumns: '1fr 1fr 1fr auto',
592                 gap: '10px',
593                 marginBottom: '15px'
594             }>
595
596             <div style={{
597                 display: 'grid',
598                 gridTemplateColumns: '1fr 1fr 1fr auto',
599                 gap: '10px',
600                 marginBottom: '15px'
601             }>
602
603             <div style={{
604                 display: 'grid',
605                 gridTemplateColumns: '1fr 1fr 1fr auto',
606                 gap: '10px',
607                 marginBottom: '15px'
608             }>
609
610             <div style={{
611                 display: 'grid',
612                 gridTemplateColumns: '1fr 1fr 1fr auto',
613                 gap: '10px',
614                 marginBottom: '15px'
615             }>
616
617             <div style={{
618                 display: 'grid',
619                 gridTemplateColumns: '1fr 1fr 1fr auto',
620                 gap: '10px',
621                 marginBottom: '15px'
622             }>
623
624             <div style={{
625                 display: 'grid',
626                 gridTemplateColumns: '1fr 1fr 1fr auto',
627                 gap: '10px',
628                 marginBottom: '15px'
629             }>
630
631             <div style={{
632                 display: 'grid',
633                 gridTemplateColumns: '1fr 1fr 1fr auto',
634                 gap: '10px',
635                 marginBottom: '15px'
636             }>
637
638             <div style={{
639                 display: 'grid',
640                 gridTemplateColumns: '1fr 1fr 1fr auto',
641                 gap: '10px',
642                 marginBottom: '15px'
643             }>
644
645             <div style={{
646                 display: 'grid',
647                 gridTemplateColumns: '1fr 1fr 1fr auto',
648                 gap: '10px',
649                 marginBottom: '15px'
650             }>
651
652             <div style={{
653                 display: 'grid',
654                 gridTemplateColumns: '1fr 1fr 1fr auto',
655                 gap: '10px',
656                 marginBottom: '15px'
657             }>
658
659             <div style={{
660                 display: 'grid',
661                 gridTemplateColumns: '1fr 1fr 1fr auto',
662                 gap: '10px',
663                 marginBottom: '15px'
664             }>
665
666             <div style={{
667                 display: 'grid',
668                 gridTemplateColumns: '1fr 1fr 1fr auto',
669                 gap: '10px',
670                 marginBottom: '15px'
671             }>
672
673             <div style={{
674                 display: 'grid',
675                 gridTemplateColumns: '1fr 1fr 1fr auto',
676                 gap: '10px',
677                 marginBottom: '15px'
678             }>
679
680             <div style={{
681                 display: 'grid',
682                 gridTemplateColumns: '1fr 1fr 1fr auto',
683                 gap: '10px',
684                 marginBottom: '15px'
685             }>
686
687             <div style={{
688                 display: 'grid',
689                 gridTemplateColumns: '1fr 1fr 1fr auto',
690                 gap: '10px',
691                 marginBottom: '15px'
692             }>
693
694             <div style={{
695                 display: 'grid',
696                 gridTemplateColumns: '1fr 1fr 1fr auto',
697                 gap: '10px',
698                 marginBottom: '15px'
699             }>
700
701             <div style={{
702                 display: 'grid',
703                 gridTemplateColumns: '1fr 1fr 1fr auto',
704                 gap: '10px',
705                 marginBottom: '15px'
706             }>
707
708             <div style={{
709                 display: 'grid',
710                 gridTemplateColumns: '1fr 1fr 1fr auto',
711                 gap: '10px',
712                 marginBottom: '15px'
713             }>
714
715             <div style={{
716                 display: 'grid',
717                 gridTemplateColumns: '1fr 1fr 1fr auto',
718                 gap: '10px',
719                 marginBottom: '15px'
720             }>
721
722             <div style={{
723                 display: 'grid',
724                 gridTemplateColumns: '1fr 1fr 1fr auto',
725                 gap: '10px',
726                 marginBottom: '15px'
727             }>
728
729             <div style={{
730                 display: 'grid',
731                 gridTemplateColumns: '1fr 1fr 1fr auto',
732                 gap: '10px',
733                 marginBottom: '15px'
734             }>
735
736             <div style={{
737                 display: 'grid',
738                 gridTemplateColumns: '1fr 1fr 1fr auto',
739                 gap: '10px',
740                 marginBottom: '15px'
741             }>
742
743             <div style={{
744                 display: 'grid',
745                 gridTemplateColumns: '1fr 1fr 1fr auto',
746                 gap: '10px',
747                 marginBottom: '15px'
748             }>
749
750             <div style={{
751                 display: 'grid',
752                 gridTemplateColumns: '1fr 1fr 1fr auto',
753                 gap: '10px',
754                 marginBottom: '15px'
755             }>
756
757             <div style={{
758                 display: 'grid',
759                 gridTemplateColumns: '1fr 1fr 1fr auto',
760                 gap: '10px',
761                 marginBottom: '15px'
762             }>
763
764             <div style={{
765                 display: 'grid',
766                 gridTemplateColumns: '1fr 1fr 1fr auto',
767                 gap: '10px',
768                 marginBottom: '15px'
769             }>
770
771             <div style={{
772                 display: 'grid',
773                 gridTemplateColumns: '1fr 1fr 1fr auto',
774                 gap: '10px',
775                 marginBottom: '15px'
776             }>
777
778             <div style={{
779                 display: 'grid',
780                 gridTemplateColumns: '1fr 1fr 1fr auto',
781                 gap: '10px',
782                 marginBottom: '15px'
783             }>
784
785             <div style={{
786                 display: 'grid',
787                 gridTemplateColumns: '1fr 1fr 1fr auto',
788                 gap: '10px',
789                 marginBottom: '15px'
790             }>
791
792             <div style={{
793                 display: 'grid',
794                 gridTemplateColumns: '1fr 1fr 1fr auto',
795                 gap: '10px',
796                 marginBottom: '15px'
797             }>
798
799             <div style={{
800                 display: 'grid',
801                 gridTemplateColumns: '1fr 1fr 1fr auto',
802                 gap: '10px',
803                 marginBottom: '15px'
804             }>
805
806             <div style={{
807                 display: 'grid',
808                 gridTemplateColumns: '1fr 1fr 1fr auto',
809                 gap: '10px',
810                 marginBottom: '15px'
811             }>
812
813             <div style={{
814                 display: 'grid',
815                 gridTemplateColumns: '1fr 1fr 1fr auto',
816                 gap: '10px',
817                 marginBottom: '15px'
818             }>
819
820             <div style={{
821                 display: 'grid',
822                 gridTemplateColumns: '1fr 1fr 1fr auto',
823                 gap: '10px',
824                 marginBottom: '15px'
825             }>
826
827             <div style={{
828                 display: 'grid',
829                 gridTemplateColumns: '1fr 1fr 1fr auto',
830                 gap: '10px',
831                 marginBottom: '15px'
832             }>
833
834             <div style={{
835                 display: 'grid',
836                 gridTemplateColumns: '1fr 1fr 1fr auto',
837                 gap: '10px',
838                 marginBottom: '15px'
839             }>
840
841             <div style={{
842                 display: 'grid',
843                 gridTemplateColumns: '1fr 1fr 1fr auto',
844                 gap: '10px',
845                 marginBottom: '15px'
846             }>
847
848             <div style={{
849                 display: 'grid',
850                 gridTemplateColumns: '1fr 1fr 1fr auto',
851                 gap: '10px',
852                 marginBottom: '15px'
853             }>
854
855             <div style={{
856                 display: 'grid',
857                 gridTemplateColumns: '1fr 1fr 1fr auto',
858                 gap: '10px',
859                 marginBottom: '15px'
860             }>
861
862             <div style={{
863                 display: 'grid',
864                 gridTemplateColumns: '1fr 1fr 1fr auto',
865                 gap: '10px',
866                 marginBottom: '15px'
867             }>
868
869             <div style={{
870                 display: 'grid',
871                 gridTemplateColumns: '1fr 1fr 1fr auto',
872                 gap: '10px',
873                 marginBottom: '15px'
874             }>
875
876             <div style={{
877                 display: 'grid',
878                 gridTemplateColumns: '1fr 1fr 1fr auto',
879                 gap: '10px',
880                 marginBottom: '15px'
881             }>
882
883             <div style={{
884                 display: 'grid',
885                 gridTemplateColumns: '1fr 1fr 1fr auto',
886                 gap: '10px',
887                 marginBottom: '15px'
888             }>
889
890             <div style={{
891                 display: 'grid',
892                 gridTemplateColumns: '1fr 1fr 1fr auto',
893                 gap: '10px',
894                 marginBottom: '15px'
895             }>
896
897             <div style={{
898                 display: 'grid',
899                 gridTemplateColumns: '1fr 1fr 1fr auto',
900                 gap: '10px',
901                 marginBottom: '15px'
902             }>
903
904             <div style={{
905                 display: 'grid',
906                 gridTemplateColumns: '1fr 1fr 1fr auto',
907                 gap: '10px',
908                 marginBottom: '15px'
909             }>
910
911             <div style={{
912                 display: 'grid',
913                 gridTemplateColumns: '1fr 1fr 1fr auto',
914                 gap: '10px',
915                 marginBottom: '15px'
916             }>
917
918             <div style={{
919                 display: 'grid',
920                 gridTemplateColumns: '1fr 1fr 1fr auto',
921                 gap: '10px',
922                 marginBottom: '15px'
923             }>
924
925             <div style={{
926                 display: 'grid',
927                 gridTemplateColumns: '1fr 1fr 1fr auto',
928                 gap: '10px',
929                 marginBottom: '15px'
930             }>
931
932             <div style={{
933                 display: 'grid',
934                 gridTemplateColumns: '1fr 1fr 1fr auto',
935                 gap: '10px',
936                 marginBottom: '15px'
937             }>
938
939             <div style={{
940                 display: 'grid',
941                 gridTemplateColumns: '1fr 1fr 1fr auto',
942                 gap: '10px',
943                 marginBottom: '15px'
944             }>
945
946             <div style={{
947                 display: 'grid',
948                 gridTemplateColumns: '1fr 1fr 1fr auto',
949                 gap: '10px',
950                 marginBottom: '15px'
951             }>
952
953             <div style={{
954                 display: 'grid',
955                 gridTemplateColumns: '1fr 1fr 1fr auto',
956                 gap: '10px',
957                 marginBottom: '15px'
958             }>
959
960             <div style={{
961                 display: 'grid',
962                 gridTemplateColumns: '1fr 1fr 1fr auto',
963                 gap: '10px',
964                 marginBottom: '15px'
965             }>
966
967             <div style={{
968                 display: 'grid',
969                 gridTemplateColumns: '1fr 1fr 1fr auto',
970                 gap: '10px',
971                 marginBottom: '15px'
972             }>
973
974             <div style={{
975                 display: 'grid',
976                 gridTemplateColumns: '1fr 1fr 1fr auto',
977                 gap: '10px',
978                 marginBottom: '15px'
979             }>
980
981             <div style={{
982                 display: 'grid',
983                 gridTemplateColumns: '1fr 1fr 1fr auto',
984                 gap: '10px',
985                 marginBottom: '15px'
986             }>
987
988             <div style={{
989                 display: 'grid',
990                 gridTemplateColumns: '1fr 1fr 1fr auto',
991                 gap: '10px',
992                 marginBottom: '15px'
993             }>
994
995             <div style={{
996                 display: 'grid',
997                 gridTemplateColumns: '1fr 1fr 1fr auto',
998                 gap: '10px',
999                 marginBottom: '15px'
1000            }>
1001
1002             <div style={{
1003                 display: 'grid',
1004                 gridTemplateColumns: '1fr 1fr 1fr auto',
1005                 gap: '10px',
1006                 marginBottom: '15px'
1007            }>
1008
1009             <div style={{
1010                 display: 'grid',
1011                 gridTemplateColumns: '1fr 1fr 1fr auto',
1012                 gap: '10px',
1013                 marginBottom: '15px'
1014            }>
1015
1016             <div style={{
1017                 display: 'grid',
1018                 gridTemplateColumns: '1fr 1fr 1fr auto',
1019                 gap: '10px',
1020                 marginBottom: '15px'
1021            }>
1022
1023             <div style={{
1024                 display: 'grid',
1025                 gridTemplateColumns: '1fr 1fr 1fr auto',
1026                 gap: '10px',
1027                 marginBottom: '15px'
1028            }>
1029
1030             <div style={{
1031                 display: 'grid',
1032                 gridTemplateColumns: '1fr 1fr 1fr auto',
1033                 gap: '10px',
1034                 marginBottom: '15px'
1035            }>
1036
1037             <div style={{
1038                 display: 'grid',
1039                 gridTemplateColumns: '1fr 1fr 1fr auto',
1040                 gap: '10px',
1041                 marginBottom: '15px'
1042            }>
1043
1044             <div style={{
1045                 display: 'grid',
1046                 gridTemplateColumns: '1fr 1fr 1fr auto',
1047                 gap: '10px',
1048                 marginBottom: '15px'
1049            }>
1050
1051             <div style={{
1052                 display: 'grid',
1053                 gridTemplateColumns: '1fr 1fr 1fr auto',
1054                 gap: '10px',
1055                 marginBottom: '15px'
1056            }>
1057
1058             <div style={{
1059                 display: 'grid',
1060                 gridTemplateColumns: '1fr 1fr 1fr auto',
1061                 gap: '10px',
1062                 marginBottom: '15px'
1063            }>
1064
1065             <div style={{
1066                 display: 'grid',
1067                 gridTemplateColumns: '1fr 1fr 1fr auto',
1068                 gap: '10px',
1069                 marginBottom: '15px'
1070            }>
1071
1072             <div style={{
1073                 display: 'grid',
1074                 gridTemplateColumns: '1fr 1fr 1fr auto',
1075                 gap: '10px',
1076                 marginBottom: '15px'
1077            }>
1078
1079             <div style={{
1080                 display: 'grid',
1081                 gridTemplateColumns: '1fr 1fr 1fr auto',
1082                 gap: '10px',
1083                 marginBottom: '15px'
1084            }>
1085
1086             <div style={{
1087                 display: 'grid',
1088                 gridTemplateColumns: '1fr 1fr 1fr auto',
1089                 gap: '10px',
1090                 marginBottom: '15px'
1091            }>
1092
1093             <div style={{
1094                 display: 'grid',
1095                 gridTemplateColumns: '1fr 1fr 1fr auto',
1096                 gap: '10px',
1097                 marginBottom: '15px'
1098            }>
1099
1100             <div style={{
1101                 display: 'grid',
1102                 gridTemplateColumns: '1fr 1fr 1fr auto',
1103                 gap: '10px',
1104                 marginBottom: '15px'
1105            }>
1106
1107             <div style={{
1108                 display: 'grid',
1109                 gridTemplateColumns: '1fr 1fr 1fr auto',
1110                 gap: '10px',
1111                 marginBottom: '15px'
1112            }>
1113
1114             <div style={{
1115                 display: 'grid',
1116                 gridTemplateColumns: '1fr 1fr 1fr auto',
1117                 gap: '10px',
1118                 marginBottom: '15px'
1119            }>
1120
1121             <div style={{
1122                 display: 'grid',
1123                 gridTemplateColumns: '1fr 1fr 1fr auto',
1124                 gap: '10px',
1125                 marginBottom: '15px'
1126            }>
1127
1128             <div style={{
1129                 display: 'grid',
1130                 gridTemplateColumns: '1fr 1fr 1fr auto',
1131                 gap: '10px',
1132                 marginBottom: '15px'
1133            }>
1134
1135             <div style={{
1136                 display: 'grid',
1137                 gridTemplateColumns: '1fr 1fr 1fr auto',
1138                 gap: '10px',
1139                 marginBottom: '15px'
1140            }>
1141
1142             <div style={{
1143                 display: 'grid',
1144                 gridTemplateColumns: '1fr 1fr 1fr auto',
1145                 gap: '10px',
1146                 marginBottom: '15px'
1147            }>
1148
1149             <div style={{
1150                 display: 'grid',
1
```

```

132     })>
133         {/* Recherche */}
134         <input
135             type="text"
136             value={searchTerm}
137             onChange={(e) => setSearchTerm(e.target.value)}
138         )}
139         placeholder="Rechercher..."
140         style={{
141             padding: '10px',
142             border: '1px solid #bdc3c7',
143             borderRadius: '5px',
144             fontSize: '14px'
145         }}
146     />
147     {/* Filtre auteur */}
148     <select
149         value={filterAuthor}
150         onChange={(e) => setFilterAuthor(e.target.value)}
151     )}
152         style={{
153             padding: '10px',
154             border: '1px solid #bdc3c7',
155             borderRadius: '5px',
156             fontSize: '14px'
157         }}
158         >
159             <option value="">Tous les auteurs</option>
160             {authors.map(author => (
161                 <option key={author} value={author}>
162                     {author}
163                 </option>
164             )))
165             </select>
166             {/* Tri */}
167             <select
168                 value={sortBy}
169                 onChange={(e) => setSortBy(e.target.value)}
170                 style={{
171                     padding: '10px',
172                     border: '1px solid #bdc3c7',
173                     borderRadius: '5px',
174                     fontSize: '14px'
175                 }}
176             >
177                 <option value="date">Plus récents</option>
178                 <option value="likes">Plus likés</option>
179             </select>
180

```

```

181     {/* Bouton ajouter */}
182     <button
183         onClick={() => setShowForm (!showForm)}
184         style={{
185             padding: '10px 20px',
186             backgroundColor: '#27ae60',
187             color: 'white',
188             border: 'none',
189             borderRadius: '5px',
190             cursor: 'pointer',
191             fontWeight: 'bold'
192         }}
193     >
194         {showForm ? 'Annuler' : '+ Nouvel Article'}
195     </button>
196 </div>
197
198     <div style={{ fontSize: '14px', color: '#7f8c8d' }}>
199         {filteredArticles.length} article(s) affiché(s)
200     sur {articles.length}
201     </div>
202 </div>
203
204     {/* Formulaire d'ajout */}
205     {showForm && (
206         <div style={{
207             backgroundColor: '#fff',
208             padding: '20px',
209             borderRadius: '10px',
210             marginBottom: '20px',
211             border: '2px solid #27ae60'
212         }>
213             <h3>Nouvel Article</h3>
214             <input
215                 type="text"
216                 value={title}
217                 onChange={(e) => setTitle(e.target.value)}
218                 placeholder="Titre..."
219                 style={{
220                     width: '100%',
221                     padding: '10px',
222                     marginBottom: '10px',
223                     border: '1px solid #ddd',
224                     borderRadius: '5px',
225                     fontSize: '16px'
226                 }}
227             />
228             <input
229                 type="text"
230                 value={author}
231                 onChange={(e) => setAuthor(e.target.value)}>

```

```
231         placeholder="Auteur..."  
232         style={{  
233             width: '100%',  
234             padding: '10px',  
235             marginBottom: '10px',  
236             border: '1px solid #ddd',  
237             borderRadius: '5px',  
238             fontSize: '16px'  
239         }}  
240     />  
241     <textarea  
242         value={content}  
243         onChange={(e) => setContent(e.target.value)}  
244         placeholder="Contenu..."  
245         rows="5"  
246         style={{  
247             width: '100%',  
248             padding: '10px',  
249             marginBottom: '10px',  
250             border: '1px solid #ddd',  
251             borderRadius: '5px',  
252             fontSize: '14px',  
253             fontFamily: 'inherit'  
254         }}  
255     />  
256     <button  
257         onClick={addArticle}  
258         style={{  
259             padding: '12px 24px',  
260             backgroundColor: '#27ae60',  
261             color: 'white',  
262             border: 'none',  
263             borderRadius: '5px',  
264             cursor: 'pointer',  
265             fontSize: '16px',  
266             fontWeight: 'bold'  
267         }}  
268     >  
269         Publier  
270     </button>  
271 </div>  
272 )}  
273  
274 /* Liste des articles */  
275 <div>  
276     {filteredArticles.length === 0 ? (  
277         <p style={{ textAlign: 'center', color: '#95a5a6'  
278         , padding: '40px' }}>  
279             Aucun article trouvé  
280         </p>  
281     ) : (
```

```

281         filteredArticles.map(article =>
282             <article
283                 key={article.id}
284                 style={{
285                     backgroundColor: 'white',
286                     padding: '20px',
287                     marginBottom: '20px',
288                     borderRadius: '10px',
289                     border: '1px solid #ddd'
290                 }}>
291             >
292                 <h2 style={{ color: '#2c3e50', marginTop: '0' }}>
293                     {article.title}
294                 </h2>
295                 <p style={{ color: '#7f8c8d', fontSize: '14px', marginBottom: '15px' }}>
296                     Par {article.author} | {new Date(article.createdAt).toLocaleDateString('fr-FR')}
297                     </p>
298                     <p style={{ lineHeight: '1.6', color: '#34495e' }}>
299                         {article.content}
300                     </p>
301                     <div style={{
302                         display: 'flex',
303                         alignItems: 'center',
304                         gap: '10px',
305                         marginTop: '15px',
306                         paddingTop: '15px',
307                         borderTop: '1px solid #ecf0f1'
308                     }}>
309                         <button
310                             onClick={() => handleLike(article.id)}
311                             style={{
312                                 padding: '8px 16px',
313                                 backgroundColor: '#3498db',
314                                 color: 'white',
315                                 border: 'none',
316                                 borderRadius: '5px',
317                                 cursor: 'pointer',
318                                 display: 'flex',
319                                 alignItems: 'center',
320                                 gap: '5px'
321                             }}>
322                         </button>
323                     </div>
324                 </p>
325             </article>
326         </div>
327     )

```

```

328 >           { article.likes }
329 </button>
330 <button
331     onClick={() => handleDelete(
332         article.id)}
333     style={{
334         padding: '8px 16px',
335         backgroundColor: '#e74c3c',
336         color: 'white',
337         border: 'none',
338         borderRadius: '5px',
339         cursor: 'pointer',
340         marginLeft: 'auto'
341     }}>
342     > Supprimer
343     </button>
344     </div>
345     </article>
346   ))
347   )
348 </div>
349 </div>
350 ) ;
351 }
352 }
353
354 export default BlogApp;

```

Listing 4 – src/BlogApp.jsx - Version complète

#### Récapitulatif des techniques utilisées :

- **useState avec initialisation lazy** : Charge depuis localStorage au montage
- **useEffect pour persistance** : Sauvegarde automatique
- **Filtrage multiple** : Recherche + auteur + tri
- **Immutabilité stricte** : map(), filter(), spread operator
- **Affichage conditionnel** : Formulaire et message vide
- **Calculs dérivés** : Liste d'auteurs unique avec Set

## 4 Récapitulatif et Concepts Clés

### Les 10 Règles d'Or des Hooks

1. **Toujours au top level** : Jamais dans des conditions, boucles ou fonctions
2. **Immutabilité stricte** : Toujours créer de nouveaux objets/tableaux
3. **Forme fonctionnelle** : `useState(prev => ...)` pour updates multiples
4. **Tableau de dépendances** : Inclure toutes les valeurs utilisées
5. **Cleanup obligatoire** : Pour timers, événements, abonnements
6. **Initialisation lazy** : `useState(() => ...)` pour calculs coûteux
7. **useReducer pour complexité** : Quand plusieurs états liés
8. **localStorage** : `useEffect` pour persistance
9. **Inputs contrôlés** : `value + onChange` toujours liés
10. **Clés stables** : Utiliser des IDs uniques, pas les index

### Patterns à retenir :

```

1 // Ajouter à un tableau
2 setItems ([... items , newItem]) ;

3
4 // Supprimer d'un tableau
5 setItems(items . filter(item => item . id !== id)) ;

6
7 // Modifier dans un tableau
8 setItems(items . map(item =>
9     item . id === id ? {... item , prop: value} : item
10));
11
12 // Modifier un objet
13 setUser ({... user , name: "Bob"});

14
15 // Timer avec cleanup
16 useEffect(() => {
17     const timer = setInterval(() => {...} , 1000);
18     return () => clearInterval(timer);
19 } , [deps]) ;

20
21 // localStorage
22 useEffect(() => {
23     localStorage .setItem('key' , JSON .stringify(data));
24 } , [data]);

```

## 5 Travail à Rendre

### Projet Final

Créer une application complète "Gestionnaire de Projets" combinant les trois hooks.

Fonctionnalités obligatoires :

1. useReducer : Gérer les projets (ajouter, modifier statut, supprimer)
2. useEffect : Persistance localStorage + timer par projet
3. useState : Formulaires, filtres, recherche
4. Chaque projet a : titre, description, statut (todo/doing/-done), deadline
5. Filtrer par statut
6. Trier par deadline
7. Recherche dans titre/description
8. Timer Pomodoro par projet
9. Statistiques visuelles (nombre par statut)

Livrables :

- Code source complet
- Captures d'écran de toutes les fonctionnalités
- Explication du reducer (tous les cas)
- Explication de chaque useEffect
- README avec instructions

Échéance : La veille de la prochaine séance à 23h59

Ce travail compte pour 20% de la note finale.

## 6 Ressources

### 6.1 Documentation Officielle

- React Documentation : <https://react.dev>
- useState : <https://react.dev/reference/react/useState>
- useEffect : <https://react.dev/reference/react/useEffect>
- useReducer : <https://react.dev/reference/react/useReducer>
- Rules of Hooks : <https://react.dev/reference/rules-of-hooks>

## 6.2 Outils

- ESLint Plugin React Hooks (détection des erreurs)
- React DevTools (inspection des états)