

Cours MERN - Semaine 9

React Router & Authentification JWT

Navigation et Sécurité dans une Application React

Abdelweheb GUEDDDES & Mohamed Ben Jazia / Ecole Polytechnique Sousse

8 novembre 2025

Table des matières

1	Objectifs Pédagogiques	3
2	Partie 1 : Concepts Fondamentaux (40 min)	3
2.1	React Router : Navigation dans React	3
2.1.1	Pourquoi React Router ?	3
2.1.2	Installation et Configuration	3
2.1.3	Les Composants Essentiels	4
2.2	JWT : Authentification avec JSON Web Token	5
2.2.1	Qu'est-ce qu'un JWT ?	5
2.2.2	Flux d'Authentification	5
2.2.3	Stockage du Token	5
2.3	Routes Protégées	6
3	Partie 2 : Ajout de l'Authentification au Backend (15 min)	7
3.1	Installation des Dépendances Backend	7
3.2	Mise à Jour du Modèle User	7
3.3	Routes d'Authentification	7
3.4	Middleware de Protection	9
3.5	Utilisation du Middleware	10
3.6	Fichier .env	10
3.7	Enregistrement des Routes	11
4	Partie 3 : Application React Complète (2h)	12
4.1	Structure du Projet	12
4.2	Configuration Axios	12
4.3	Context d'Authentification	13
4.4	Wrapper l'Application	14
4.5	Configuration des Routes	15
4.6	Les Composants	16
4.6.1	Navbar	16

4.6.2	ProtectedRoute	17
4.7	Les Pages	18
4.7.1	Page de Connexion	18
4.7.2	Page d'Inscription	20
4.7.3	Page des Cours	23
4.7.4	Page Détails d'un Cours	24
4.7.5	Page Profil	27
5	Points Importants à Retenir	29
6	Travail à Rendre	29
7	Ressources	30

1 Objectifs Pédagogiques

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

- **Utiliser** React Router pour créer une navigation multi-pages
- **Comprendre** le fonctionnement de JWT (JSON Web Token)
- **Implémenter** un système d'authentification complet
- **Créer** des routes protégées accessibles uniquement aux utilisateurs connectés
- **Gérer** le token JWT (stockage, envoi, validation)
- **Connecter** le frontend React avec l'API EduPlatform

Prérequis : L'API EduPlatform de mi-formation doit être fonctionnelle (modèles User, Profile, Course, Review avec les routes correspondantes).

2 Partie 1 : Concepts Fondamentaux (40 min)

2.1 React Router : Navigation dans React

2.1.1 Pourquoi React Router ?

React est une bibliothèque pour créer des **Single Page Applications (SPA)**. Par défaut, il n'y a qu'une seule page HTML. React Router permet de simuler plusieurs pages avec différentes URLs.

Comparaison :

Sans React Router :

- Une seule URL : `http://localhost:5173/`
- Tout le contenu dans `App.jsx`

Avec React Router :

- `http://localhost:5173/` → Page d'accueil
- `http://localhost:5173/courses` → Liste des cours
- `http://localhost:5173/login` → Page de connexion
- `http://localhost:5173/profile` → Profil utilisateur

2.1.2 Installation et Configuration

```
1 npm install react-router-dom
```

Configuration de base :

```
1 import React from 'react';
2 import ReactDOM from 'react-dom/client';
3 import { BrowserRouter } from 'react-router-dom';
4 import App from './App';
5
6 ReactDOM.createRoot(document.getElementById('root')).render(
7   <React.StrictMode>
8     <BrowserRouter>
9       <App />
10    </BrowserRouter>
  )
```

```
11 </React.StrictMode>
12 );
```

Listing 1 – src/main.jsx

2.1.3 Les Composants Essentiels

1. Routes et Route

Définissent les chemins et les composants à afficher.

```
1 import { Routes, Route } from 'react-router-dom';
2
3 <Routes>
4   <Route path="/" element={<Home />} />
5   <Route path="/courses" element={<Courses />} />
6   <Route path="/login" element={<Login />} />
7 </Routes>
```

2. Link

Crée des liens de navigation (remplace <a>).

```
1 import { Link } from 'react-router-dom';
2
3 <Link to="/courses">Voir les cours</Link>
```

3. useNavigate

Navigation programmatique (après une action).

```
1 import { useNavigate } from 'react-router-dom';
2
3 const navigate = useNavigate();
4
5 const handleLogin = () => {
6   // Après connexion réussie
7   navigate('/profile');
8 };
```

4. useParams

Récupère les paramètres dynamiques de l'URL.

```
1 // Route : /courses/:id
2 import { useParams } from 'react-router-dom';
3
4 function CourseDetails() {
5   const { id } = useParams();
6   // id contient la valeur de l'URL
7 }
```

2.2 JWT : Authentification avec JSON Web Token

2.2.1 Qu'est-ce qu'un JWT ?

Un JWT (JSON Web Token) est un moyen sécurisé de transmettre des informations entre le client et le serveur sous forme de token.

Structure d'un JWT :

Un JWT se compose de 3 parties séparées par des points :

xxxxx.yyyyy.zzzzz

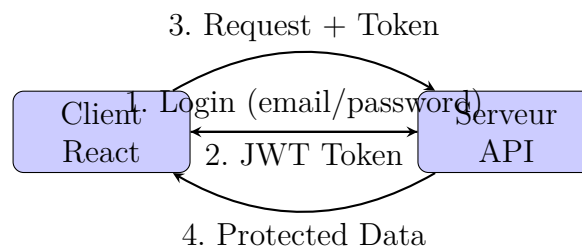
1. **Header** : Type de token et algorithme de signature
2. **Payload** : Données (ID utilisateur, email, etc.)
3. **Signature** : Vérification de l'authenticité

Exemple de JWT :

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.

eyJ1c2VySWQiOiI2NzJhYmMxMjM0NTY3ODkiLCJlbWFPbCI6ImFsaWNlQGV4YW1wbGUuY29tIn0.
SflKxwRJSMeKKF2QT4fwpMeJf36P0k6yJV_adQssw5c

2.2.2 Flux d'Authentification



Étapes :

1. L'utilisateur envoie email + password
2. Le serveur vérifie et génère un JWT
3. Le client stocke le token (localStorage ou cookies)
4. Pour chaque requête protégée, le client envoie le token dans les headers
5. Le serveur vérifie le token et autorise ou refuse

2.2.3 Stockage du Token

Option 1 : localStorage (le plus simple)

```
1 // Sauvegarder
2 localStorage.setItem('token', jwt_token);
3
4 // Récupérer
5 const token = localStorage.getItem('token');
6
7 // Supprimer (logout)
8 localStorage.removeItem('token');
```

Option 2 : sessionStorage

Même API que localStorage, mais expire à la fermeture du navigateur.

Option 3 : Cookies httpOnly

Plus sécurisé mais nécessite configuration serveur.

2.3 Routes Protégées

Une route protégée redirige vers la page de connexion si l'utilisateur n'est pas authentifié.

```
1 import { Navigate } from 'react-router-dom';
2
3 function ProtectedRoute({ children }) {
4   const token = localStorage.getItem('token');
5
6   // Si pas de token, rediriger vers login
7   if (!token) {
8     return <Navigate to="/login" replace />;
9   }
10
11   // Sinon, afficher le composant demandé
12   return children;
13 }
14
15 export default ProtectedRoute;
```

Listing 2 – src/components/ProtectedRoute.jsx

Utilisation :

```
1 <Routes>
2   <Route path="/login" element={<Login />} />
3
4   <Route
5     path="/profile"
6     element={
7       <ProtectedRoute>
8         <Profile />
9       </ProtectedRoute>
10    }
11  />
12 </Routes>
```

3 Partie 2 : Ajout de l'Authentification au Backend (15 min)

Important : L'API EduPlatform n'a pas encore de système d'authentification. Il faut l'ajouter avant de connecter le frontend.

3.1 Installation des Dépendances Backend

```
1 npm install bcryptjs jsonwebtoken
```

3.2 Mise à Jour du Modèle User

```
1 const userSchema = new mongoose.Schema({
2   username: {
3     type: String,
4     required: true,
5     unique: true
6   },
7   email: {
8     type: String,
9     required: true,
10    unique: true
11  },
12  password: {
13    type: String,
14    required: true,
15    minlength: 6
16  },
17  courses: [{
18    type: mongoose.Schema.Types.ObjectId,
19    ref: 'Course'
20  }]
21 }, { timestamps: true });
22
23 module.exports = mongoose.model('User', userSchema);
```

Listing 3 – models/User.js - Ajout du password

3.3 Routes d'Authentification

```
1 const express = require('express');
2 const router = express.Router();
3 const bcrypt = require('bcryptjs');
4 const jwt = require('jsonwebtoken');
5 const User = require('../models/User');
6
7 // @desc    Register new user
```

```
8 // @route    POST /api/auth/register
9 router.post('/register', async (req, res) => {
10   try {
11     const { username, email, password } = req.body;
12
13     // Vérifier si l'utilisateur existe
14     const userExists = await User.findOne({ email });
15     if (userExists) {
16       return res.status(400).json({ message: 'Email déjà utilis
17         é' });
18     }
19
20     // Hasher le mot de passe
21     const hashedPassword = await bcrypt.hash(password, 10);
22
23     // Créer l'utilisateur
24     const user = await User.create({
25       username,
26       email,
27       password: hashedPassword
28     });
29
30     // Générer le token
31     const token = jwt.sign(
32       { userId: user._id, email: user.email },
33       process.env.JWT_SECRET,
34       { expiresIn: '7d' }
35     );
36
37     res.status(201).json({
38       token,
39       user: {
40         id: user._id,
41         username: user.username,
42         email: user.email
43       }
44     });
45   } catch (error) {
46     res.status(500).json({ message: error.message });
47   }
48 });
49 // @desc    Login user
50 // @route    POST /api/auth/login
51 router.post('/login', async (req, res) => {
52   try {
53     const { email, password } = req.body;
54
55     // Trouver l'utilisateur
56     const user = await User.findOne({ email });
57     if (!user) {
```



```
58         return res.status(401).json({ message: 'Email ou mot de
passee incorrect' });
59     }
60
61     // Vérifier le mot de passe
62     const isMatch = await bcrypt.compare(password, user.password)
;
63     if (!isMatch) {
64         return res.status(401).json({ message: 'Email ou mot de
passee incorrect' });
65     }
66
67     // Générer le token
68     const token = jwt.sign(
69         { userId: user._id, email: user.email },
70         process.env.JWT_SECRET,
71         { expiresIn: '7d' }
72     );
73
74     res.json({
75         token,
76         user: {
77             id: user._id,
78             username: user.username,
79             email: user.email
80         }
81     });
82 } catch (error) {
83     res.status(500).json({ message: error.message });
84 }
85 });
86
87 module.exports = router;
```

Listing 4 – routes/authRoutes.js

3.4 Middleware de Protection

```
1 const jwt = require('jsonwebtoken');
2
3 const protect = (req, res, next) => {
4     let token;
5
6     // Récupérer le token du header Authorization
7     if (req.headers.authorization &&
8         req.headers.authorization.startsWith('Bearer')) {
9         try {
10             // Extraire le token
11             token = req.headers.authorization.split(' ')[1];
12
13             // Vérifier le token
```

```
14         const decoded = jwt.verify(token, process.env.JWT_SECRET)
15         ;
16         // Ajouter l'ID utilisateur à la requête
17         req.userId = decoded.userId;
18
19         next();
20     } catch (error) {
21         res.status(401).json({ message: 'Token invalide' });
22     }
23 }
24
25 if (!token) {
26     res.status(401).json({ message: 'Pas de token, accès refusé'
27 });
28 };
29
30 module.exports = { protect };
```

Listing 5 – middleware/authMiddleware.js

3.5 Utilisation du Middleware

```
1 const express = require('express');
2 const router = express.Router();
3 const { protect } = require('../middleware/authMiddleware');
4
5 // Route publique
6 router.get('/', async (req, res) => {
7     // Accessible sans authentification
8 });
9
10 // Route protégée
11 router.get('/profile', protect, async (req, res) => {
12     // req.userId contient l'ID de l'utilisateur connecté
13     const user = await User.findById(req.userId).select('-password');
14     res.json(user);
15 });
16
17 module.exports = router;
```

Listing 6 – routes/userRoutes.js - Protéger les routes

3.6 Fichier .env

```
1 PORT=5000
2 MONGO_URI=mongodb://localhost:27017/eduplatform
3 JWT_SECRET=votre_secret_super_securise_ici_changez_le
```

Listing 7 – .env

3.7 Enregistrement des Routes

```
1 const authRoutes = require('./routes/authRoutes');  
2  
3 app.use('/api/auth', authRoutes);
```

Listing 8 – server.js - Ajouter les routes auth

4 Partie 3 : Application React Complète (2h)

Objectif : Créer une application React connectée à l'API EduPlatform avec :

- Système d'authentification (Register/Login/Logout)
- Navigation avec React Router
- Routes protégées
- Liste des cours
- Page de détails d'un cours
- Page profil utilisateur

4.1 Structure du Projet

```
1 src /
2     components /
3         Navbar.jsx
4         ProtectedRoute.jsx
5         CourseCard.jsx
6     pages /
7         Home.jsx
8         Login.jsx
9         Register.jsx
10        Courses.jsx
11        CourseDetails.jsx
12        Profile.jsx
13    context /
14        AuthContext.jsx
15    api /
16        axios.js
17    App.jsx
18    main.jsx
```

4.2 Configuration Axios

```
1 import axios from 'axios';
2
3 const api = axios.create({
4     baseURL: 'http://localhost:5000/api'
5 });
6
7 // Intercepteur pour ajouter automatiquement le token
8 api.interceptors.request.use(
9     (config) => {
10         const token = localStorage.getItem('token');
11         if (token) {
12             config.headers.Authorization = `Bearer ${token}`;
13         }
14         return config;
15     },
```

```
16     (error) => {
17         return Promise.reject(error);
18     }
19 );
20
21 export default api;
```

Listing 9 – src/api/axios.js

4.3 Context d'Authentication

```
1 import { createContext, useContext, useState, useEffect } from 'react'
2   ;
3
4 const AuthContext = createContext();
5
6 export const useAuth = () => {
7     const context = useContext(AuthContext);
8     if (!context) {
9         throw new Error('useAuth must be used within AuthProvider');
10    }
11    return context;
12 };
13
14 export const AuthProvider = ({ children }) => {
15     const [user, setUser] = useState(null);
16     const [loading, setLoading] = useState(true);
17
18     // Vérifier si token existe au chargement
19     useEffect(() => {
20         const token = localStorage.getItem('token');
21         if (token) {
22             // Optionnel : Vérifier le token avec l'API
23             api.get('/auth/me')
24                 .then(res => setUser(res.data))
25                 .catch(() => localStorage.removeItem('token'))
26                 .finally(() => setLoading(false));
27         } else {
28             setLoading(false);
29         }
30     }, []);
31
32     const register = async (username, email, password) => {
33         const response = await api.post('/auth/register', {
34             username,
35             email,
36             password
37         });
38
39         localStorage.setItem('token', response.data.token);
```

```
40     setUser(response.data.user);
41     return response.data;
42   };
43
44   const login = async (email, password) => {
45     const response = await api.post('/auth/login', {
46       email,
47       password
48     });
49
50     localStorage.setItem('token', response.data.token);
51     setUser(response.data.user);
52     return response.data;
53   };
54
55   const logout = () => {
56     localStorage.removeItem('token');
57     setUser(null);
58   };
59
60   const value = {
61     user,
62     loading,
63     register,
64     login,
65     logout,
66     isAuthenticated: !!user
67   };
68
69   return (
70     <AuthContext.Provider value={value}>
71       {children}
72     </AuthContext.Provider>
73   );
74 };
```

Listing 10 – src/context/AuthContext.jsx

4.4 Wrapper l'Application

```
1 import React from 'react';
2 import ReactDOM from 'react-dom/client';
3 import { BrowserRouter } from 'react-router-dom';
4 import { AuthProvider } from '../context/AuthContext';
5 import App from './App';
6 import './index.css';
7
8 ReactDOM.createRoot(document.getElementById('root')).render(
9   <React.StrictMode>
10     <BrowserRouter>
11       <AuthProvider>
```

```
12     <App />
13     </AuthProvider>
14     </BrowserRouter>
15   </React.StrictMode>
16 );
```

Listing 11 – src/main.jsx

4.5 Configuration des Routes

```
1 import { Routes, Route } from 'react-router-dom';
2 import Navbar from '../components/Navbar';
3 import ProtectedRoute from '../components/ProtectedRoute';
4 import Home from '../pages/Home';
5 import Login from '../pages/Login';
6 import Register from '../pages/Register';
7 import Courses from '../pages/Courses';
8 import CourseDetails from '../pages/CourseDetails';
9 import Profile from '../pages/Profile';
10
11 function App() {
12   return (
13     <div>
14       <Navbar />
15       <Routes>
16         <Route path="/" element={<Home />} />
17         <Route path="/login" element={<Login />} />
18         <Route path="/register" element={<Register />} />
19         <Route path="/courses" element={<Courses />} />
20         <Route path="/courses/:id" element={<CourseDetails
21           />} />
22
23         <Route
24           path="/profile"
25           element={
26             <ProtectedRoute>
27               <Profile />
28             </ProtectedRoute>
29           }
30         />
31       </Routes>
32     </div>
33   );
34 }
35 export default App;
```

Listing 12 – src/App.jsx

4.6 Les Composants

4.6.1 Navbar

```
1 import { Link } from 'react-router-dom';
2 import { useAuth } from '../context/AuthContext';
3
4 function Navbar() {
5   const { user, logout, isAuthenticated } = useAuth();
6
7   return (
8     <nav style={{
9       backgroundColor: '#2c3e50',
10      padding: '15px 30px',
11      display: 'flex',
12      justifyContent: 'space-between',
13      alignItems: 'center'
14    }}>
15       <div style={{ display: 'flex', gap: '20px' }}>
16         <Link to="/" style={{ color: 'white', textDecoration:
17           'none' }}>
18           Accueil
19         </Link>
20         <Link to="/courses" style={{ color: 'white',
21           textDecoration: 'none' }}>
22           Cours
23         </Link>
24       </div>
25
26       <div style={{ display: 'flex', gap: '15px', alignItems: '
27         center' }}>
28         {isAuthenticated ? (
29           <div>
30             <Link to="/profile" style={{ color: 'white',
31               textDecoration: 'none' }}>
32               Profil ({user.username})
33             </Link>
34             <button
35               onClick={logout}
36               style={{
37                 padding: '8px 16px',
38                 backgroundColor: '#e74c3c',
39                 color: 'white',
40                 border: 'none',
41                 borderRadius: '5px',
42                 cursor: 'pointer'
43               }}
44             >
45               Déconnexion
46             </button>
47           </div>
48         ) : (
```



```
45         <Link to="/login" style={{ color: 'white',
46         textDecoration: 'none' }}>
47             Connexion
48         </Link>
49         <Link to="/register" style={{ color: 'white',
50         textDecoration: 'none' }}>
51             Inscription
52         </Link>
53     </>
54 </div>
55 </nav>
56 );
57 }
58
59 export default Navbar;
```

Listing 13 – src/components/Navbar.jsx

4.6.2 ProtectedRoute

```
1 import { Navigate } from 'react-router-dom';
2 import { useAuth } from '../context/AuthContext';
3
4 function ProtectedRoute({ children }) {
5     const { isAuthenticated, loading } = useAuth();
6
7     if (loading) {
8         return <div>Chargement... </div>;
9     }
10
11     if (!isAuthenticated) {
12         return <Navigate to="/login" replace />;
13     }
14
15     return children;
16 }
17
18 export default ProtectedRoute;
```

Listing 14 – src/components/ProtectedRoute.jsx

4.7 Les Pages

4.7.1 Page de Connexion

```
1 import { useState } from 'react';
2 import { useNavigate, Link } from 'react-router-dom';
3 import { useAuth } from '../context/AuthContext';
4
5 function Login() {
6   const [email, setEmail] = useState('');
7   const [password, setPassword] = useState('');
8   const [error, setError] = useState('');
9   const { login } = useAuth();
10  const navigate = useNavigate();
11
12  const handleSubmit = async (e) => {
13    e.preventDefault();
14    setError('');
15
16    try {
17      await login(email, password);
18      navigate('/profile');
19    } catch (err) {
20      setError(err.response?.data?.message || 'Erreur de
connexion');
21    }
22  };
23
24  return (
25    <div style={{
26      maxWidth: '400px',
27      margin: '100px auto',
28      padding: '30px',
29      backgroundColor: 'white',
30      borderRadius: '10px',
31      boxShadow: '0 2px 10px rgba(0,0,0,0.1)'
32    }}>
33      <h2>Connexion</h2>
34
35      {error && (
36        <div style={{
37          padding: '10px',
38          marginBottom: '15px',
39          backgroundColor: '#fee',
40          color: '#c33',
41          borderRadius: '5px'
42        }}>
43          {error}
44        </div>
45      )}
46
47      <form onSubmit={handleSubmit}>
```

```
48     <div style={{ marginBottom: '15px' }}>
49       <label>Email :</label>
50       <input
51         type="email"
52         value={email}
53         onChange={(e) => setEmail(e.target.value)}
54         required
55         style={{
56           width: '100%',
57           padding: '10px',
58           marginTop: '5px',
59           border: '1px solid #ddd',
60           borderRadius: '5px'
61         }}
62       />
63     </div>
64
65     <div style={{ marginBottom: '15px' }}>
66       <label>Mot de passe :</label>
67       <input
68         type="password"
69         value={password}
70         onChange={(e) => setPassword(e.target.value)}
71         required
72         style={{
73           width: '100%',
74           padding: '10px',
75           marginTop: '5px',
76           border: '1px solid #ddd',
77           borderRadius: '5px'
78         }}
79       />
80     </div>
81
82     <button
83       type="submit"
84       style={{
85         width: '100%',
86         padding: '12px',
87         backgroundColor: '#3498db',
88         color: 'white',
89         border: 'none',
90         borderRadius: '5px',
91         cursor: 'pointer',
92         fontSize: '16px'
93       }}
94     >
95       Se connecter
96     </button>
97   </form>
98
```

```
99         <p style={{ marginTop: '15px', textAlign: 'center' }}>
100             Pas encore de compte ? <Link to="/register">S'
inscrire</Link>
101         </p>
102     </div>
103 );
104 }
105
106 export default Login;
```

Listing 15 – src/pages/Login.jsx

4.7.2 Page d'Inscription

```
1 import { useState } from 'react';
2 import { useNavigate, Link } from 'react-router-dom';
3 import { useAuth } from '../context/AuthContext';
4
5 function Register() {
6     const [username, setUsername] = useState('');
7     const [email, setEmail] = useState('');
8     const [password, setPassword] = useState('');
9     const [error, setError] = useState('');
10    const { register } = useAuth();
11    const navigate = useNavigate();
12
13    const handleSubmit = async (e) => {
14        e.preventDefault();
15        setError('');
16
17        try {
18            await register(username, email, password);
19            navigate('/profile');
20        } catch (err) {
21            setError(err.response?.data?.message || 'Erreur d\'
inscription');
22        }
23    };
24
25    return (
26        <div style={{
27            maxWidth: '400px',
28            margin: '100px auto',
29            padding: '30px',
30            backgroundColor: 'white',
31            borderRadius: '10px',
32            boxShadow: '0 2px 10px rgba(0,0,0,0.1)'
33        }}>
34            <h2>Inscription</h2>
35
36            {error && (
```

```
37     <div style={{
38         padding: '10px',
39         marginBottom: '15px',
40         backgroundColor: '#fee',
41         color: '#c33',
42         borderRadius: '5px'
43     }}>
44         {error}
45     </div>
46 )}
47
48 <form onSubmit={handleSubmit}>
49     <div style={{ marginBottom: '15px' }}>
50         <label>Nom d'utilisateur :</label>
51         <input
52             type="text"
53             value={username}
54             onChange={(e) => setUsername(e.target.value)}
55             required
56             style={{
57                 width: '100%',
58                 padding: '10px',
59                 marginTop: '5px',
60                 border: '1px solid #ddd',
61                 borderRadius: '5px'
62             }}
63         />
64     </div>
65
66     <div style={{ marginBottom: '15px' }}>
67         <label>Email :</label>
68         <input
69             type="email"
70             value={email}
71             onChange={(e) => setEmail(e.target.value)}
72             required
73             style={{
74                 width: '100%',
75                 padding: '10px',
76                 marginTop: '5px',
77                 border: '1px solid #ddd',
78                 borderRadius: '5px'
79             }}
80         />
81     </div>
82
83     <div style={{ marginBottom: '15px' }}>
84         <label>Mot de passe :</label>
85         <input
86             type="password"
87             value={password}
```

```
88         onChange={e => setPassword(e.target.value)}
89         required
90         minLength={6}
91         style={{
92             width: '100%',
93             padding: '10px',
94             marginTop: '5px',
95             border: '1px solid #ddd',
96             borderRadius: '5px'
97         }}
98     />
99 </div>
100
101     <button
102         type="submit"
103         style={{
104             width: '100%',
105             padding: '12px',
106             backgroundColor: '#27ae60',
107             color: 'white',
108             border: 'none',
109             borderRadius: '5px',
110             cursor: 'pointer',
111             fontSize: '16px'
112         }}
113     >
114         S'inscrire
115     </button>
116 </form>
117
118     <p style={{ marginTop: '15px', textAlign: 'center' }}>
119         Déjà un compte ? <Link to="/login">Se connecter</Link>
120     </p>
121 </div>
122 );
123 }
124
125 export default Register;
```

Listing 16 – src/pages/Register.jsx

4.7.3 Page des Cours

```
1 import { useState, useEffect } from 'react';
2 import { Link } from 'react-router-dom';
3 import api from '../api/axios';
4
5 function Courses() {
6   const [courses, setCourses] = useState([]);
7   const [loading, setLoading] = useState(true);
8
9   useEffect(() => {
10     api.get('/courses')
11       .then(res => {
12         setCourses(res.data);
13         setLoading(false);
14       })
15       .catch(err => {
16         console.error(err);
17         setLoading(false);
18       });
19   }, []);
20
21   if (loading) {
22     return <div style={{ textAlign: 'center', padding: '50px' }}>
Chargement... </div>;
23   }
24
25   return (
26     <div style={{ padding: '30px', maxWidth: '1200px', margin: '0
auto' }}>
27       <h1>Liste des Cours</h1>
28
29       <div style={{
30         display: 'grid',
31         gridTemplateColumns: 'repeat(auto-fill, minmax(300px,
1fr))',
32         gap: '20px',
33         marginTop: '30px'
34       }}>
35         {courses.map(course => (
36           <div key={course._id} style={{
37             padding: '20px',
38             backgroundColor: 'white',
39             borderRadius: '10px',
40             boxShadow: '0 2px 5px rgba(0,0,0,0.1)'
41           }}>
42             <h3>{course.title}</h3>
43             <p style={{ color: '#666', marginTop: '10px' }}
44               {course.description.substring(0, 100)}...
45             </p>
```

```
46         <p style={{ marginTop: '10px', fontWeight: '
    bold' }}>
47             Instructeur : {course.instructor}
48         </p>
49         <p style={{ color: '#3498db' }}>
50             {course.students.length} étudiants
    inscrits
51         </p>
52
53         <Link
54             to={`/${courses}/${course._id}`}
55             style={{
56                 display: 'inline-block',
57                 marginTop: '15px',
58                 padding: '10px 20px',
59                 backgroundColor: '#3498db',
60                 color: 'white',
61                 textDecoration: 'none',
62                 borderRadius: '5px'
63             }}
64         >
65             Voir détails
66         </Link>
67     </div>
68     )))
69 </div>
70 </div>
71 );
72 }
73
74 export default Courses;
```

Listing 17 – src/pages/Courses.jsx

4.7.4 Page Détails d'un Cours

```
1 import { useState, useEffect } from 'react';
2 import { useParams, useNavigate } from 'react-router-dom';
3 import { useAuth } from '../context/AuthContext';
4 import api from '../api/axios';
5
6 function CourseDetails() {
7     const { id } = useParams();
8     const [course, setCourse] = useState(null);
9     const [reviews, setReviews] = useState([]);
10    const [loading, setLoading] = useState(true);
11    const { isAuthenticated } = useAuth();
12    const navigate = useNavigate();
13
14    useEffect(() => {
15        // Charger le cours
```



```
16     api.get(`/courses/${id}`)
17       .then(res => setCourse(res.data))
18       .catch(err => console.error(err));
19
20     // Charger les reviews
21     api.get(`/courses/${id}/reviews`)
22       .then(res => {
23         setReviews(res.data);
24         setLoading(false);
25       })
26       .catch(err => {
27         console.error(err);
28         setLoading(false);
29       });
30   }, [id]);
31
32   const handleEnroll = async () => {
33     if (!isAuthenticated) {
34       navigate('/login');
35       return;
36     }
37
38     try {
39       await api.post(`/courses/${id}/enroll`);
40       alert('Inscription réussie !');
41     } catch (err) {
42       alert('Erreur lors de l\'inscription');
43     }
44   };
45
46   if (loading || !course) {
47     return <div style={{ textAlign: 'center', padding: '50px' }}>
Chargement... </div>;
48   }
49
50   return (
51     <div style={{ padding: '30px', maxWidth: '900px', margin: '0
auto' }}>
52       <h1>{course.title}</h1>
53       <p style={{ fontSize: '18px', color: '#666', marginTop: '
15px' }}>
54         {course.description}
55       </p>
56
57       <div style={{
58         marginTop: '20px',
59         padding: '20px',
60         backgroundColor: '#f8f9fa',
61         borderRadius: '10px'
62       }}>
63         <p><strong>Instructeur :</strong> {course.instructor
```

```

    }</p>
64         <p><strong> étudiants inscrits :</strong> {course.
students.length}</p>
65     </div>
66
67     <button
68         onClick={handleEnroll}
69         style={{
70             marginTop: '20px',
71             padding: '15px 30px',
72             backgroundColor: '#27ae60',
73             color: 'white',
74             border: 'none',
75             borderRadius: '5px',
76             cursor: 'pointer',
77             fontSize: '16px'
78         }}
79     >
80         S'inscrire au cours
81     </button>
82
83     <h2 style={{ marginTop: '40px' }}>Avis des étudiants</h2>
84
85     {reviews.length === 0 ? (
86         <p style={{ color: '#999' }}>Aucun avis pour le
moment</p>
87     ) : (
88         reviews.map(review => (
89             <div key={review._id} style={{
90                 padding: '15px',
91                 marginTop: '15px',
92                 backgroundColor: 'white',
93                 borderRadius: '5px',
94                 border: '1px solid #ddd'
95             }}>
96                 <div style={{ color: '#f39c12', fontSize: '20
px' }}>
97                     {review.rating}.repeat(review.rating)}
98                 </div>
99                 <p style={{ marginTop: '10px' }}>{review.
comment}</p>
100             </div>
101         ))
102     )}
103 </div>
104 );
105 }
106
107 export default CourseDetails;

```

Listing 18 – src/pages/CourseDetails.jsx

4.7.5 Page Profil

```
1 import { useState, useEffect } from 'react';
2 import { useAuth } from '../context/AuthContext';
3 import api from '../api/axios';
4
5 function Profile() {
6   const { user } = useAuth();
7   const [enrolledCourses, setEnrolledCourses] = useState([]);
8   const [loading, setLoading] = useState(true);
9
10  useEffect(() => {
11    api.get(`/users/${user.id}/courses`)
12      .then(res => {
13        setEnrolledCourses(res.data);
14        setLoading(false);
15      })
16      .catch(err => {
17        console.error(err);
18        setLoading(false);
19      });
20  }, [user.id]);
21
22  return (
23    <div style={{ padding: '30px', maxWidth: '900px', margin: '0
24    auto' }}>
25      <h1>Mon Profil</h1>
26
27      <div style={{
28        marginTop: '20px',
29        padding: '20px',
30        backgroundColor: 'white',
31        borderRadius: '10px',
32        boxShadow: '0 2px 5px rgba(0,0,0,0.1)'
33      }}>
34        <p><strong>Nom d'utilisateur :</strong> {user.
35        username}</p>
36        <p><strong>Email :</strong> {user.email}</p>
37      </div>
38
39      <h2 style={{ marginTop: '40px' }}>Mes Cours</h2>
40
41      {loading ? (
42        <div>Chargement...</div>
43      ) : enrolledCourses.length === 0 ? (
44        <p style={{ color: '#999' }}>Vous n'êtes inscrit à
45        aucun cours</p>
46      ) : (
47        <div style={{
48          display: 'grid',
49          gridTemplateColumns: 'repeat(auto-fill, minmax
```

```
47 (250px, 1fr))',
48     gap: '20px',
49     marginTop: '20px'
50   }}>
51   {enrolledCourses.map(course => (
52     <div key={course._id} style={{
53       padding: '15px',
54       backgroundColor: 'white',
55       borderRadius: '5px',
56       border: '1px solid #ddd'
57     }}>
58       <h4>{course.title}</h4>
59       <p style={{ fontSize: '14px', color: '
60         #666' }}>
61         {course.instructor}
62       </p>
63     </div>
64   ))}
65 </div>
66 );
67 }
68
69 export default Profile;
```

Listing 19 – src/pages/Profile.jsx

5 Points Importants à Retenir

React Router :

- `<BrowserRouter>` entoure toute l'application
- `<Routes>` et `<Route>` définissent les chemins
- `<Link>` pour la navigation (remplace `<a>`)
- `useNavigate()` pour redirection programmatique
- `useParams()` pour récupérer les paramètres d'URL

JWT et Authentification :

- Token stocké dans `localStorage`
- Envoyé dans le header `Authorization: Bearer TOKEN`
- Middleware backend pour vérifier le token
- Context React pour partager l'état d'authentification
- Routes protégées redirigent vers `/login` si non connecté

Sécurité :

- Ne JAMAIS stocker de données sensibles dans le token
- Hasher les mots de passe avec `bcrypt`
- Utiliser HTTPS en production
- Définir une date d'expiration au token
- Utiliser des secrets forts pour `JWT_SECRET`

6 Travail à Rendre

Projet Final

Compléter l'application avec ces fonctionnalités supplémentaires.

Fonctionnalités à ajouter :

1. Page "Mes Reviews" listant toutes les critiques de l'utilisateur
2. Formulaire pour ajouter une review sur `CourseDetails`
3. Page d'édition du profil (bio, website)
4. Pagination sur la liste des cours (10 par page)
5. Barre de recherche pour filtrer les cours par titre
6. Page 404 (Not Found) pour les routes inexistantes

Livrables :

- Code frontend React complet

- Code backend mis à jour avec auth
 - README avec instructions de démarrage
 - Captures d'écran de toutes les pages
 - Fichier .env.example
- Échéance : La veille de l'examen final à 23h59
- Ce travail compte pour 30% de la note finale.

7 Ressources

- React Router : <https://reactrouter.com/>
- JWT Documentation : <https://jwt.io/>
- Axios : <https://axios-http.com/>
- Bcrypt : <https://www.npmjs.com/package/bcryptjs>