Atelier de Professionnalisation 3
Maison des Ligues

Contexte :

Le contexte proposé est celui de la Maison de Ligues de Lorraine (M2L) qui a pour mission de fournir des espaces et des services aux différentes ligues sportives régionales et à d'autres structures hébergées.

SOMMAIRE :

OUTILS UTILISÉS :

Serveur

```
const express = require('express')//la récupération d'express

const app = express() // création d'une instance d'express

require('dotenv').config() // configuration des variables
d'environnement stockées dans un fichier .env

let cors = require('cors') // utilisation de la bibliothèque cors pour
gérer les requêtes cross-origin (CORS)

const mariadb = require('mariadb'); // récupération du module mariadb
pour se connecter à la base de données

app.use(express.json()) // utilisation du middleware express pour
parser les données en JSON

app.use(cors()) // activation de la gestion CORS pour toutes les routes

const pool = mariadb.createPool ({
    host: process.env.DB_HOST, // récupération de l'adresse de l'hôte
de la base de données depuis les variables d'environnement
    user: process.env.DB_USER, // récupération de l'utilisateur de la
base de données depuis les variables d'environnement
    password: process.env.DB_PWD, // récupération du mot de passe de la
base de données depuis les variables d'environnement
    database: process.env.DB_DTB // récupération du nom de la base de
données depuis les variables d'environnement
});

app.get('/Bdd', async(req,res) => { // définition d'une route pour
récupérer les données de la base de données
    let conn;
    try{
        console.log("lancement de la connexion") // affichage d'un
message dans la console pour indiquer le début de la connexion
```

```
        conn = await pool.getConnection(); // établissement de la
connexion avec la base de données
        console.log("lancement de la requete") // affichage d'un
message dans la console pour indiquer le début de la requête
        const rows = await conn.query('SELECT * FROM ap2'); //
exécution d'une requête pour récupérer toutes les données de la table
ap2
        console.log(rows); // affichage des données récupérées dans la
console
        res.status(200).json(rows) // envoi des données récupérées au
client sous forme de JSON
    }
    catch(err){
        console.log(err) // affichage d'un message d'erreur dans la
console en cas de problème lors de l'exécution de la requête
    }
})

// Cette fonction écoute les requêtes GET sur l'URL '/produit'
app.get('/produit', async(req,res) => {
  let conn;
  try{
      // On affiche un message de connexion
      console.log("lancement de la connexion")
      // On récupère une connexion à la base de données
      conn = await pool.getConnection();
      // On affiche un message de requête
      console.log("lancement de la requete")
      // On exécute une requête SELECT pour récupérer tous les produits
de la table 'produit'
      const rows = await conn.query('SELECT * FROM produit');
      // On affiche le résultat de la requête
      console.log(rows);
      // On renvoie les résultats au format JSON
      res.status(200).json(rows)
  }
  catch(err){
      // On affiche les erreurs s'il y en a une
      console.log(err)
  }
})

// Cette fonction écoute les requêtes POST sur l'URL '/inscr'
```

```
app.post('/inscr', async (req, res) => {
  let conn;
  try {
      // On affiche un message de connexion
      console.log("lancement de la connexion")
      // On récupère une connexion à la base de données
      conn = await pool.getConnection();
      // On affiche un message de requête
      console.log("lancement de la requete insert")
      // On affiche le contenu de la requête POST
      console.log(req.body);
      // On exécute une requête INSERT pour insérer une nouvelle ligne
dans la table 'ap2'
      let requete = 'INSERT INTO ap2 (mail, mdp) VALUES (?, ?);'
      let rows = await conn.query(requete, [req.body.mail, req.body.mdp
]);
      // On affiche le nombre de lignes affectées
      console.log(rows);
      // On renvoie le nombre de lignes affectées au format JSON
      res.status(200).json(rows.affectedRows)
  }
  catch (err) {
      // On affiche les erreurs s'il y en a une
      console.log(err);
  }
})

// Endpoint pour vérifier l'authentification d'un utilisateur en
utilisant un objet `pool` fourni.
app.put('/connexion', async (req, res) => {
  const id = parseInt(req.params.id);
  const { mail, mdp } = req.body; // Récupérer les valeurs de mail et
mdp du corps de la requête
  let conn;
  try {
    console.log("Lancement de la connexion");
    conn = await pool.getConnection();
    console.log("Lancement de la requête");
    // Interroger la base de données pour récupérer l'utilisateur
    const rows = await conn.query('SELECT * FROM ap2 WHERE mail = ? AND
mdp = ?', [mail, mdp]);
    console.log(rows);
```

```
    // Si un utilisateur est trouvé, le renvoyer, sinon renvoyer une
erreur d'authentification
    if (rows.length === 1) {
      res.status(200).json(rows[0]);
    } else {
      alert("pas correct")
      res.status(401).json({ message: "Nom d'utilisateur ou mot de
passe incorrect." });
    }
  } catch (err) {
    console.log(err);
    res.status(500).json({ message: "Une erreur s'est produite lors de
la connexion à la base de données." });
  } finally {
    if (conn) {
      conn.release(); // Libérer la connexion à la base de données
    }
  }
});

// Endpoint pour ajouter un nouveau produit
app.post('/Ajt', async(req,res) => {
  let conn;
  try{
      console.log("lancement de la connexion")
      conn = await pool.getConnection();
      console.log("lancement de la requete")
      // Insérer un nouveau produit dans la base de données
      const rows = await conn.query ('INSERT INTO produit (Articles,
Image, Prix) VALUES (?, ?, ?)', [req.body.Articles, req.body.Image,
req.body.Prix]);
      console.log(rows);
      res.status(200).json(rows.affectedRows)
  }
  catch(err){
      console.log(err)
  }
})

// Endpoint pour supprimer un produit en fonction de son ID
app.delete('/Del/:id', async(req,res) => {
const id = parseInt(req.params.id)
let conn;
```

```
try{
    console.log("lancement de la connexion")
    conn = await pool.getConnection();
    console.log("lancement de la requete")
    // Supprimer un produit de la base de données en fonction de son ID
    const rows = await conn.query ('DELETE FROM produit WHERE id = ?',
[id]);
    console.log(rows);
    res.status(200).json(rows.affectedRows)
}
catch(err){
    console.log(err)
}
})

// Démarrer le serveur sur le port 8000
app.listen(8000, ()=>{
  console.log("Serveur à l'écoute");
})
```

APP

```
// On importe les fichiers CSS et les différents composants
import '../style/App.css';
import Banniere from './Banniere';
import Produits from './Produits';
import Panier from './Panier';
import Inscrire from './Inscrire';
import Connecter from './Connecter';
import CoAdmin from './ConnexionAdmin';
import Admin from './Admin';
import Supprimer from './Supprimer'
import Sup from './Suppression';
import Footer from './Footer'

// On importe les composants nécessaires pour utiliser les routes
import { Route, Routes, Link } from "react-router-dom";
```

```
// On importe le hook useState de React
import React, { useState } from 'react';

// On crée la fonction App qui va contenir notre application
function App() {
  return (
    // On crée une div qui contiendra notre application
    <div className="App">

      {/* On utilise le composant Banniere pour afficher la bannière
*/}
      <Banniere />

      {/* On utilise le composant Routes pour gérer les différentes
routes de l'application */}
      <Routes>
        {/* On définit la route pour la page d'inscription */}
        <Route path="/inscr" element={<Inscrire />} />
        {/* On définit la route pour la page de connexion */}
        <Route path="/connect" element={<Connecter />} />
        {/* On définit la route pour la page de connexion à l'interface
d'administration */}
        <Route path='/ConAdmin' element={<CoAdmin/>}/>
        {/* On définit la route pour la page de l'interface
d'administration */}
        <Route path='/Admin' element={<Admin/>}/>
        {/* On définit la route pour la page de suppression d'articles
*/}
        <Route path='/Sup' element={<Supprimer/>}/>
        {/* On définit la route pour la page de suppression d'un
article spécifique */}
        <Route path='/Del/:id' element={<Sup/>}/>
        {/* On définit la route pour la page d'accueil */}
        <Route path="/" element={<Produits />} />
      </Routes>

      {/* On utilise le composant Footer pour afficher le footer */}
      <Footer/>
    </div>
  );
}
```

```
// On exporte la fonction App pour pouvoir l'utiliser dans d'autres
fichiers
export default App;
```

ARTICLES

```
import React from 'react'
import { useEffect, useState } from 'react';  // Importation de useEffect et useState depuis React
import axios from 'axios';  // Importation d'axios pour effectuer des requêtes HTTP
import { Link } from 'react-router-dom'  // Importation de Link depuis react-router-dom pour naviguer entre les pages
import { FaTrash, FaPen } from 'react-icons/fa';  // Importation des icônes FaTrash et FaPen depuis react-icons/fa
import '../style/Produits.css';  // Importation du fichier CSS pour le style
import Balle from '../asset/balle.png'  // Importation de l'image Balle depuis le dossier asset

export default function Quiz() {  // Déclaration du composant Quiz
    const [quiz, setQuiz] = useState([])  // Déclaration d'un état "quiz" initialisé à un tableau vide
    const [affichage, setAffichage] = useState(false)  // Déclaration d'un état "affichage" initialisé à false

    const recup = async () => {  // Déclaration d'une fonction asynchrone "recup"
        await axios.get(`http://localhost:8000/produit`)  // Requête HTTP GET à l'adresse http://localhost:8000/produit
            .then(res => {  // Si la requête est réussie
                console.log(res)  // Afficher la réponse dans la console du navigateur
                setQuiz(res.data)  // Mettre à jour l'état "quiz" avec les données de la réponse
                setAffichage(true)  // Mettre à jour l'état "affichage" à true
            })
    }

    useEffect(() => {  // Utilisation du hook useEffect
        recup()  // Appel de la fonction "recup" pour récupérer les données dès que le composant est monté
```

```
    }, [])

    return (  // Rendu du composant
        <div className='body'>  // Div principale avec une classe
"body"
            <h2> Les produits </h2>  // Titre "Les produits"
            <div className="box">  // Div avec une classe "box"
                {affichage ?  // Si "affichage" est true
                    quiz.map(produit => (  // Boucler sur chaque
produit dans "quiz"
                        <div>  // Div contenant les informations du
produit
                            <div className='box-title' >  // Div avec
une classe "box-title"
                                {produit.Articles}  {produit.Prix}  //
Afficher le nom et le prix du produit
                                <img src={produit.Image}/>  // Afficher
l'image du produit
                            </div>
                            <div className='box-body'>  // Div avec une
classe "box-body"

                            </div>
                        </div>
                    ))
                    : <p>Chargement...</p>  // Sinon, afficher
"Chargement..."
                }
            </div>
        </div>
    )
}
```

INSCRIRE

```
import React from 'react'
import { useState } from 'react';
import { useForm } from "react-hook-form"; // importation de useForm pour gérer les formulaires
import axios from 'axios'; // importation d'axios pour effectuer des appels HTTP
import { useNavigate } from "react-router-dom"; // importation de useNavigate pour la navigation

export default function AjoutQuestion() {
    const { register, handleSubmit, formState: { errors } } = useForm(); // Initialisation des hooks useForm pour le formulaire
    let navigate = useNavigate(); // Initialisation de useNavigate pour la navigation

    const [mail, setmail] = useState("") // Initialisation de l'état mail à une chaîne vide
    const [mdp, setMdp] = useState("") // Initialisation de l'état mdp à une chaîne vide

    const ajoutQuestion = async () => { // Fonction ajoutQuestion qui sera appelée lors de la soumission du formulaire
        await axios.post(`http://localhost:8000/inscr`, { // Appel HTTP POST pour l'inscription
            mail: mail, // Envoi de l'email
            mdp: mdp // Envoi du mot de passe
        })
            .then(res => { // Si la requête est réussie
                console.log(res)
                if (res.status === 200) { // Si le code de statut HTTP est 200 (OK)
                    alert("Ajout réussi") // Affichage d'une alerte avec le message "Ajout réussi"
                    navigate("/"); // Navigation vers la page d'accueil
                }
                else { // Sinon
                    alert("Erreur d'ajout") // Affichage d'une alerte avec le message "Erreur d'ajout"
                }
```

```
            })
    }

    return (
        <div className='container' style={{ marginTop:'200px'}}>
            <h2> Inscrivez-vous </h2>
            <form onSubmit={handleSubmit(ajoutQuestion)} > //
Utilisation de la fonction handleSubmit pour gérer la soumission du
formulaire
                <label>E_mail </label>
                <input {...register("mail", { required: true })}
onChange={(e) => setmail(e.target.value)} /> // Utilisation de la
fonction register pour enregistrer le champ de l'email dans le
formulaire

                <label>Mdp </label>
                <input type='password' {...register("mdp", { required:
true })} onChange={(e) => setMdp(e.target.value)} /> // Utilisation de
la fonction register pour enregistrer le champ du mot de passe dans le
formulaire

                {(errors.mail || errors.mdp ) ? <span>Tous les champs
doivent être remplis</span> : ""} // Affichage d'un message d'erreur si
les champs ne sont pas remplis

                <input type="Submit" /> // Bouton de soumission du
formulaire
            </form>
        </div>
    )
}
```

CONNEXION

```
import React from 'react'
import { useState } from 'react'; // Importation des hooks useState
import { useForm } from "react-hook-form"; // Importation du hook
useForm
```

```
import axios from 'axios'; // Importation de la librairie axios pour effectuer des requêtes HTTP
import '../style/Formulaire.css'; // Importation du fichier CSS
import { useNavigate } from "react-router-dom"; // Importation de la fonction useNavigate pour naviguer entre les pages de l'application

export default function Connexion() {
    const { register, handleSubmit, formState: { errors } } = useForm(); // Déclaration du hook useForm pour gérer le formulaire
    let navigate = useNavigate(); // Initialisation de la fonction useNavigate

    const [mail, setMail] = useState("") // Initialisation de l'état mail à une chaîne vide
    const [mdp, setMdp] = useState("") // Initialisation de l'état mdp à une chaîne vide

    // Fonction pour gérer la connexion de l'utilisateur
    const handleConnexion = async () => {
        await axios.put(`http://localhost:8000/connexion`, { // Envoi d'une requête PUT à l'API
            mail: mail, // Ajout de l'adresse mail de l'utilisateur dans le corps de la requête
            mdp: mdp // Ajout du mot de passe de l'utilisateur dans le corps de la requête
        })
            .then(res => {
                console.log(res)
                if (res.status === 200) { // Si la réponse de l'API est un code de statut 200, la connexion est réussie
                    alert("Connexion réussie") // Affichage d'une alerte pour informer l'utilisateur que la connexion est réussie
                    navigate("/"); // Redirection vers la page d'accueil
                }
                else { // Sinon, la connexion a échoué
                    alert("Erreur de connexion") // Affichage d'une alerte pour informer l'utilisateur que la connexion a échoué
                }
            })
    }

    // Rendu du composant
```

```
    return (
        <div className='container' style={{ marginTop:'200px'}}>
            <h2> Connexion </h2>

            <form className='form'
onSubmit={handleSubmit(handleConnexion)}>
                <label>Adresse email </label>
                <input {...register("mail", { required: true })}
onChange={(e) => setMail(e.target.value)} />

                <label>Mot de passe </label>
                <input type="password" {...register("mdp", { required:
true })} onChange={(e) => setMdp(e.target.value)} />

                {(errors.mail || errors.mdp) ? <span>Tous les champs
doivent être remplis</span> : ""}

                <input type="submit" />
            </form>
        </div>
    )
}
```

ADMIN

```
// Importer les bibliothèques nécessaires
import React from 'react'
import { useState } from 'react';
import { useForm } from "react-hook-form";
import axios from 'axios';
import { Link } from 'react-router-dom';
import { useNavigate } from "react-router-dom";

// Créer et exporter le composant AjoutArticles
export default function AjoutArticles() {
// Initialiser useForm pour la validation de formulaire
const { register, handleSubmit, formState: { errors } } = useForm();
// Initialiser useNavigate pour la navigation de page
let navigate = useNavigate();
```

```
// Initialiser les états Articles, Image et Prix pour stocker les
données entrées dans les champs de saisie
const [Articles, setArticles] = useState("")
const [Image, setImage] = useState("")
const [Prix, setPrix] = useState("")

// Créer une fonction pour envoyer une requête de post au serveur et
ajouter un nouvel article
const AjoutArticles = async () => {
    // Envoyer une requête post à l'API avec les données des champs de
saisie
    await axios.post(`http://localhost:8000/Ajt`, {
        Articles: Articles,
        Image: Image,
        Prix: Prix
    })
        .then(res => {
            console.log(res)
            // Vérifier si la requête a réussi ou non
            if (res.status === 200) {
                alert("Ajout réussi")
                // Naviguer vers la page d'accueil après l'ajout réussi
                navigate("/");
            }
            else {
                alert("Erreur d'ajout")
            }
        })
}


return (
    // Afficher le formulaire d'ajout d'article
    <div className='container' style={{ marginTop:'200px'}}>
        <h2> Ajouter un article</h2>

        <form onSubmit={handleSubmit(AjoutArticles)}>
            <label>Articles </label>
            <input {...register("Articles", { required: true })}
onChange={(e) => setArticles(e.target.value)} />

            <label>Lien de l'images </label>
```

```
            <input {...register("Image", { required: true })}
onChange={(e) => setImage(e.target.value)} />

            <label>Prix </label>
            <input {...register("Prix", { required: true })}
onChange={(e) => setPrix(e.target.value)} />

            // Afficher un message d'erreur si tous les champs ne sont
pas remplis
            {(errors.Articles || errors.Image || errors.Prix) ?
<span>Tous les champs doivent être remplis</span> : ""}

            <input type="submit" />
            // Créer des liens pour naviguer vers la page
d'administration et la page de suppression d'articles
            <Link to="/Admin"><input type='button' value='Ajouter un
article'/></Link>
            <Link to="/Sup"><input type='button' value='Supprimer un
article'/></Link>
        </form>
    </div>
)
}
```

SUPPRIMER

```
import React from 'react'
import { useEffect, useState } from 'react';
import axios from 'axios';
import { Link } from 'react-router-dom'
import { FaTrash, FaPen } from 'react-icons/fa';
import '../style/Produits.css';

export default function Quiz() {
    const [quiz, setQuiz] = useState([]) // état pour stocker les
produits récupérés
    const [affichage, setAffichage] = useState(false) // état pour
indiquer si la récupération des produits est terminée

    // fonction asynchrone pour récupérer les produits depuis le
backend
    const recup = async () => {
```

```
        await axios.get(`http://localhost:8000/produit`) // appel à
l'API GET pour récupérer les produits
            .then(res => {
                console.log(res)
                setQuiz(res.data) // stockage des produits dans l'état
                setAffichage(true) // indique que la récupération des
produits est terminée
            })
    }

    useEffect(() => {
        recup() // appel de la fonction de récupération des produits
une fois que le composant est monté
    }, [])

    return (
        <div className='body'>
            <h2> Les produits </h2>
            <div className="box">
                {affichage ?
                    quiz.map(produit => ( // boucle sur les produits
stockés dans l'état
                        <div>
                            <div className='box-title' >
                                Produit n°{produit.id} // affiche l'ID
du produit
                            </div>
                            <div className='box-body'>
                            {produit.Articles}  {produit.Prix} //
affiche le nom et le prix du produit
                                <img src={produit.Image}/> // affiche
l'image du produit
                            </div>
                            <div className='box-footer'>
                                <Link to={'/Del/'+ produit.id}><FaTrash
/></Link> // lien pour supprimer le produit
                            </div>

                        </div>
                    ))
                    : <p>Chargement...</p> // affiche "Chargement..."
tant que la récupération des produits n'est pas terminée
                }
```

```
                <p> <Link to="/Admin"><input type='button'
value='Ajouter un article' style={{ width:'200px'}}/></Link></p> //
lien pour ajouter un produit
                <Link to="/Sup"><input type='button' value='Supprimer
un article' style={{ width:'200px'}}/></Link> // lien pour supprimer un
produit
            </div>
        </div>
    )
}
```

SUPPRESSION

```
import React from 'react'
import { useForm } from "react-hook-form";
import { Link } from 'react-router-dom';
import { useParams, useNavigate } from "react-router-dom";
import axios from 'axios';

export default function SuppressionQuestion() {
    // Récupération du hook useForm() qui permet de gérer le formulaire
    const { handleSubmit } = useForm();
    // Récupération du paramètre d'URL "id"
    let { id } = useParams();
    // Récupération du hook useNavigate() pour naviguer entre les pages
    let navigate = useNavigate();

    // Fonction pour supprimer la question en envoyant une requête
DELETE au backend
    const suppressionQuestion = async () => {
        await axios.delete(`http://localhost:8000/Del/` + id)
            .then(res => {
                console.log(res);
                if (res.status === 200) {
                    alert("Suppression réussie");
                    navigate("/");
                }
                else {
                    alert("Erreur de suppression");
                }
```

```
            });
    }

    return (
        <div className='container' style={{ marginTop:'200px'}}>
            <form onSubmit={handleSubmit(suppressionQuestion)} >
                <h2> Êtes-vous sûr de vouloir supprimer l'article
?</h2>
                <input type="submit" value="Valider" />
                <Link to="/" className='bouton-annuler'> Annuler
</Link>
            </form>
        </div>
    )
}
```