MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR DE LA RECHERCHE SCIENTIFIQUE ET DE L'INNOVATION

UNIVERSITÉ SULTAN MOULAY SLIMANE

ÉCOLE NATIONALE DES SCIENCES APPLIQUÉES DE KHOURIBGA

Rapport TP

Express

Réalisé par :

Achir Chaimaa

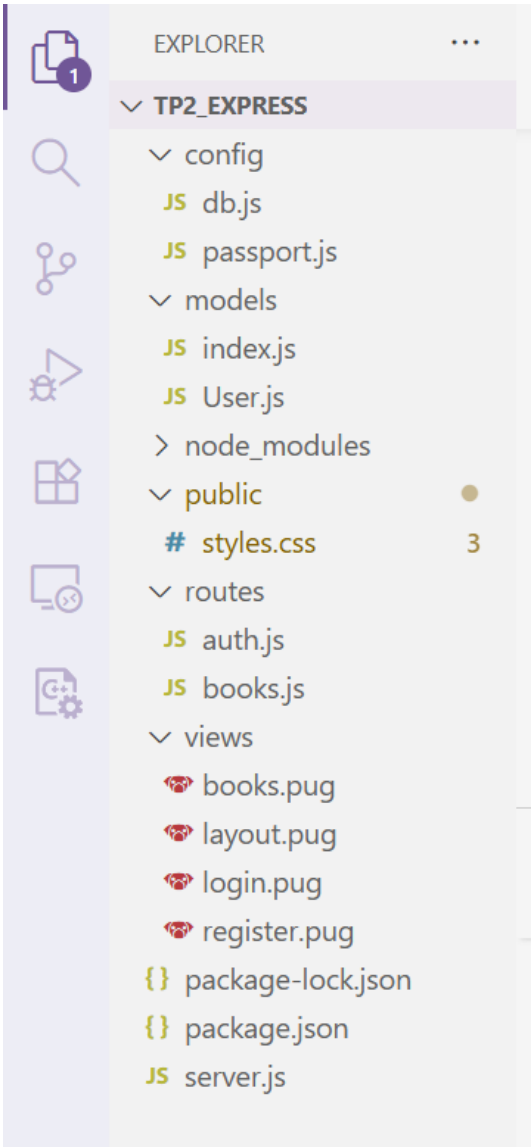
Encadré par : Pr. ourdo Amal

Année Universitaire : 2025 / 2026

Objectif du TP

L'objectif de ce TP est de créer une application web complète qui gère l'inscription et l'authentification des utilisateurs à l'aide de Passport.js et, avec une interface construite en Pug et stylisée avec Tailwind CSS. Après authentification, l'utilisateur est redirigé vers une page affichant une liste de livres, qui n'est accessible qu'aux utilisateurs connectés.

Architecture du projet



1 - Créez une page web d'inscription et d'authentification en utilisant Pug.

Code :

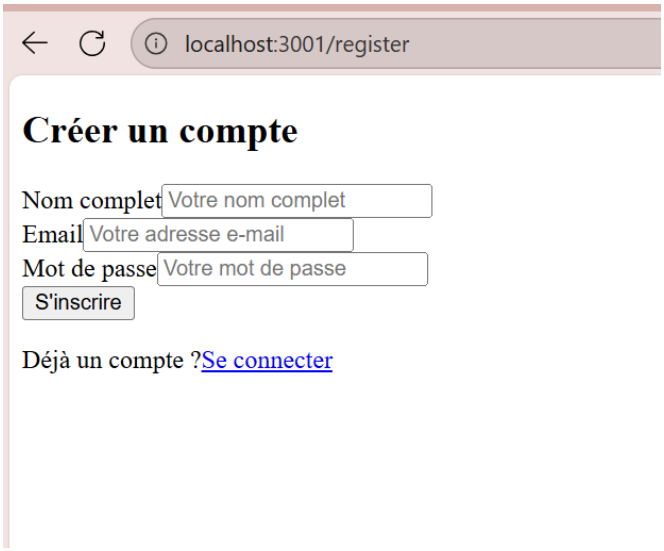
```
block content
  .bg-white.p-8.rounded.shadow-md.w-96
    h2.text-2xl.mb-4.text-center Connexion
    form(action="/login" method="POST")
      input(type="email" name="email" placeholder="Email" class="w-full border p-2 rounded mb-3")
      input(type="password" name="password" placeholder="Mot de passe" class="w-full border p-2 rounded mb-3")
      button(type="submit" class="bg-green-500 text-white w-full py-2 rounded") Se connecter
    a.text-blue-500(href="/register") Pas encore de compte ? S'inscrire
```

Exécution :

```
extends layout

block content
  .bg-white.p-8.rounded.shadow-md.w-96
    h2.text-2xl.font-bold.mb-6.text-center Créer un compte

    form(action="/register" method="POST")
      div.mb-4
        label.block.text-gray-700.text-sm.font-bold.mb-2(for="name") Nom complet
        input#name(type="text" name="name" placeholder="Votre nom complet" required class="shadow appearance-none bor

      div.mb-4
        label.block.text-gray-700.text-sm.font-bold.mb-2(for="email") Email
        input#email(type="email" name="email" placeholder="Votre adresse e-mail" required class="shadow appearance-no

      div.mb-6
        label.block.text-gray-700.text-sm.font-bold.mb-2(for="password") Mot de passe
        input#password(type="password" name="password" placeholder="Votre mot de passe" required class="shadow appear

      button(type="submit" class="bg-blue-500 hover:bg-blue-700 text-white font-bold py-2 px-4 rounded w-full") S'ins

    p.text-center.text-sm.text-gray-600.mt-4
      | Déjà un compte ?
      a(class="text-blue-500 hover:underline" href="/login") Se connecter
```

Code :

Execution :



Utilisez passport.js pour gérer l'authentification.

```
const { User } = require("../models");
const LocalStrategy = require("passport-local").Strategy;
const bcrypt = require("bcryptjs");

module.exports = function (passport) {
  passport.use(
    new LocalStrategy({ usernameField: "email" }, async (email, password, done) => {
      try {
        const user = await User.findOne({ where: { email } });
        if (!user) return done(null, false, { message: "Utilisateur non trouvé" });

        const isMatch = await bcrypt.compare(password, user.password);
        if (!isMatch) return done(null, false, { message: "Mot de passe incorrect" });

        return done(null, user);
      } catch (err) {
        return done(err);
      }
    })
  );

  passport.serializeUser((user, done) => done(null, user.id));
  passport.deserializeUser(async (id, done) => {
    try {
      const user = await User.findByPk(id);
```

Après une authentification réussie, redirigez l'utilisateur vers la page des livres.

```
app.post(
  "/login",
  passport.authenticate("local", {
    successRedirect: "/books",
    failureRedirect: "/login",
  })
);
```

Les livres sont stockés dans une variable locale.s

```
const books = [
  { title: "Le Petit Prince", author: "Antoine de Saint-Exupéry" },
  { title: "L'Étranger", author: "Albert Camus" },
  { title: "Les Misérables", author: "Victor Hugo" },
  { title: "Candide", author: "Voltaire" },
];

app.get("/books", ensureAuthenticated, (req, res) => {
  res.render("books", { user: req.user, books });
});
```

Resultat

La page des livres ne doit pas être accessible si l'utilisateur n'est pas authentifié.

```
function ensureAuthenticated(req, res, next) {
  if (req.isAuthenticated()) return next();
  res.redirect("/login");
}
```

```
app.get("/books", ensureAuthenticated, (req, res) => {
  res.render("books", { user: req.user, books });
});
```