PORTFOLIO

Évaluation de compétences - Back-end

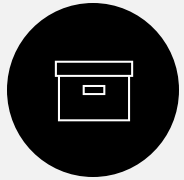
TP – DWWM AFEC Bayonne

2024/2025

EYMAS Ryan

SOMMAIRE

PRESENTATION



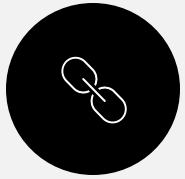
PROJET

TECHNOLOGIES UTILISÉES

DÉVELOPPEMENT

DÉPLOIEMENT

CONCLUSION



RESSOURCES

PRÉSENTATION

- Ryan EYMAS
- Parcours scolaire paysagisme & travail
- Formation préparatoires aux métiers du numérque
- TP – développeur web et web mobile
- Stage de 10 semaines à partir du 17 mars

PROJET

Portfolio **dynamique**

Projet fullstack commun proposé par le formateur

Application **MERN** (MongoDB, Express, React, Node.js)

Gérer et **afficher** ses compétences

Sécurisée & **responsive** (mobile-first)

Conforme au **RGPD** avec gestion des **cookies** et un **reCaptcha** pour **sécuriser** l'authentification

FONCTIONNALITÉS

BASE DE DONNÉES MONGODB

STOCKAGE DES UTILISATEURS, COMPÉTENCES ET PRÉFÉRENCES

BACKEND AVEC EXPRESS

API REST, GESTION DES IMAGES (CLOUDINARY), JWT & BCRYPT POUR L'AUTHENTIFICATION

SÉCURITÉ

HELMET, CORS, MIDDLEWARE DE RÔLES, GESTION DES COOKIES, RECAPTCHA

DÉPLOIEMENT

BACKEND (RENDER), FRONTEND (VERCEL)

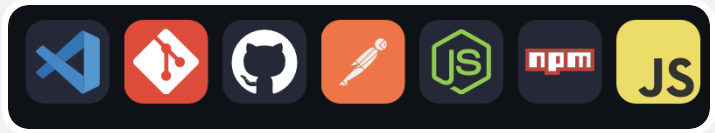
FRONTEND AVEC REACT

PORTFOLIO INTERACTIF, DASHBOARD SÉCURISÉ, REACT ROUTER, BOOTSTRAP

COMMENT

CRUD (create, read, update, delete) pour les utilisateurs et les compétences

Technologies utilisées



ORGANISATION

Back - Entry
- server
- + Ajouter une carte

Back - Config
- db
- logg
- + Ajouter une carte

Back - Models
- User
- Skills
- Settings
- + Ajouter une carte

Back - Controllers
- authController (add, getAll, update, delete, login)
- skillController (add, getAll, update, delete)
- Token (JWT)
- Hash (bcrypt)
- + Ajouter une carte

Back - Middleware
- authMiddleware (JWT)
- isAdmin
- morganMiddleware (logs)
- recaptchaMiddleware (google reCaptcha)
- + Ajouter une carte

Back - Routes
- authRoutes
- skillsRoutes
- protect, isAdmin & verifyCaptcha
- + Ajouter une carte

DÉVELOPPEMENT

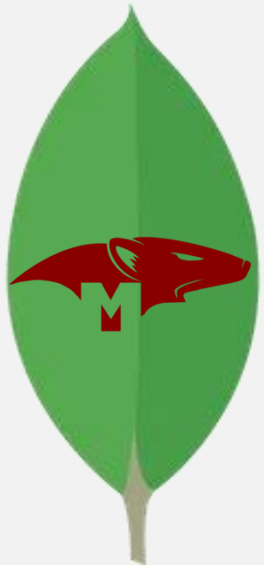
Back-end

- Express pour la création de l'API et la gestion des requêtes
- MongoDB pour la base de données
- Architecture MVC (Model View Controller)
- Sécurité (JWT, hash, helmet, rôles, reCaptcha)
- Testing avec Postman (avec collection)

DÉVELOPPEMENT

Mise en place de la base de données

- MongoDB : Stockage des données
- Mongoose : Lien entre la base de données et Javascript Node.js
- Mongo Compass pour la visualisation des données



```
const mongoose = require('mongoose');

const connectDB = async () => {
  try {
    await mongoose.connect(process.env.MONGO_URI);
    console.log(`Base de donnée connectée`);
  } catch (error) {
    console.error(`Erreur lors de la connexion à la base de donnée`, error);
  }
}

module.exports = connectDB      You, last week • commit
```

```
const mongoose = require('mongoose');      You, last

const userSchema = new mongoose.Schema({
  name: {
    type: String,
    required: true,
  },
  email: {
    type: String,
    required: true,
    unique: true
  },
  password: {
    type: String,
    required: true,
  },
  role: {
    type: String,
    enum: ["user", "admin"],
    default: "user",
    required: true
  }
}, {
  timestamps: true
})

module.exports = mongoose.model('User', userSchema);
```

DÉVELOPPEMENT

Les contrôleurs

- Logique métier et fonctionnalité
- CRUD (création, lecture, mise à jour et suppression)

```
exports.registerUser = async (req, res) => {
  const { name, email, password, role } = req.body;
  if (!name || !email || !password) {
    return res.status(400).json({ message: `Tous les champs doivent être remplis` });
  }
  try {
    const hashedPassword = await bcrypt.hash(password, saltRounds);
    const user = await User.create({ name, email, password: hashedPassword, role });
    res.status(201).json({ message: `L'utilisateur a bien été crée: `, user });
  } catch (error) {
    console.error(error);
    res.status(500).json({ message: `Erreur lors de la création de l'utilisateur` });
  }
};

exports.deleteUser = async (req, res) => {
  try {
    const { id } = req.params;
    const user = await User.findByIdAndDelete(id);
    res.status(200).json({ message: `Utilisateur supprimé avec succès`, user });
  } catch (error) {
    res
      .status(500)
      .json({
        message: `Erreur lors de la suppression de l'utilisateur`,
        error,
      });      You, 1 second ago • Uncommitted changes
  }
};
```

DÉVELOPPEMENT

La sécurité

- Hash du mot de passe à la création d'un utilisateur
- Création d'un token lors de la connexion
- Gestion des rôles et routes restreintes
- reCaptcha Google pour prévenir des bots
- Helmet pour les en-têtes http

```
exports.registerUser = async (req, res) => {
  const { name, email, password, role } = req.body;
  if (!name || !email || !password) {
    return res.status(400).json({ message: `Tous les champs doivent être remplis` });
  }
  try {
    const hashedPassword = await bcrypt.hash(password, saltRounds);
    const user = await User.create({ name, email, password: hashedPassword, role });
    res.status(201).json({ message: `L'utilisateur a bien été crée: `, user });
  } catch (error) {
    console.error(error);
    res.status(500).json({ message: `Erreur lors de la création de l'utilisateur` });
  }
};
```


```
_id: ObjectId('67b73e534aea368517ce62b6')
name : "Ryan"
email : "ryanadmin@mail.com"
password : "$2b$10$2JItu1EO8PZKPkzMIf6lauK3Fn0aoyuAgwM4aTaeFyqxD0xvfhBpa"
role : "admin"
createdAt : 2025-02-20T14:38:11.776+00:00
updatedAt : 2025-02-20T14:38:11.776+00:00
__v : 0
```

DÉVELOPPEMENT

Testing

J'ai effectué tests de routes sur Postman

Lien vers la collection

Method	Path	Desc
POST	/api/auth/register	Créer un utilisateur
GET	/api/auth/getallusers	Afficher tous les utilisateurs
PUT	/api/auth/updateuser/:id	Modifier un utilisateur
DEL	/api/auth/deleteuser/:id	Supprimer un utilisateur
POST	/api/auth/login	Connecter un utilisateur
POST	/api/skills/addskill	Ajouter une compétence
GET	/api/skills/getallskills	Afficher toutes les compétences
PUT	/api/skills/updateskill/:id	Modifier une compétence
DEL	/api/skills/deleteskill/:id	Supprimer une compétence

DÉVELOPPEMENT

Liaison back-front

- Utilisation de CORS dans le back-end
- Utilisation d'axios dans le front

```
import axios from "axios";
```

```
const cors = require('cors')
app.use(cors())
```

```
const fetchSkills = async () => {
  try {
    const response = await axios.get(`${API_URL}/api/skills/getallskills`, {
      headers: getAuthHeaders(),
    });
```

DÉVELOPPEMENT

Front-end

- Configuration avec Vite
- React Router pour la navigation entre les pages
- Utilisation d'axios pour récupérer et afficher les données dynamiquement
- Utilisation de Bootstrap React pour l'interface utilisateur
- Mobile first & responsive

DÉVELOPPEMENT

Gestion et protection des routes

```jsx
import { Navigate } from "react-router-dom";

const ProtectedRoute = ({ children }) => {
  const token = localStorage.getItem("token");

  return token ? children : <Navigate to="/login" />;
};

export default ProtectedRoute;
```

```jsx
function App() {
  return (
    <>
      <Router>
        <Navbar />

        <Routes>
          <Route path="/" element={<Home />} />
          <Route path="/login" element={<Login />} />
          <Route path="/register" element={<Register />} />
          <Route
            path="/dashboard"
            element={
              <ProtectedRoute>
                <Dashboard />
                <ProtectedRoute />
              </ProtectedRoute>
            }
          />      You, last week • commit
        </Routes>

        <Footer />
      </Router>
    </>
  );
}
```

DÉVELOPPEMENT

Affichage dynamique des compétences récupérées de la base de données

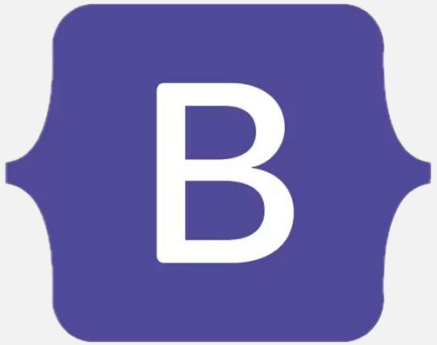
Il est également possible de les modifier et de les supprimer

```
useEffect(() => {
  document.title = "Portfolio - Dashbord";

  const fetchSkills = async () => {
    try {
      const response = await axios.get(`${API_URL}/api/skills/getallskills`, {
        headers: getAuthHeaders(),
      });

      if (response.data && response.data.skills) {
        setSkills(response.data.skills);
      } else {
        console.error("Format de réponse invalide:", response.data);
        setSkills([]);
      }
    } catch (error) {
      console.error("Erreur détaillée:", {
        message: error.message,
        response: error.response?.data,
        status: error.response?.status,
      });
      setSkills([]);
    }
  };
  fetchSkills();
}, []);
```

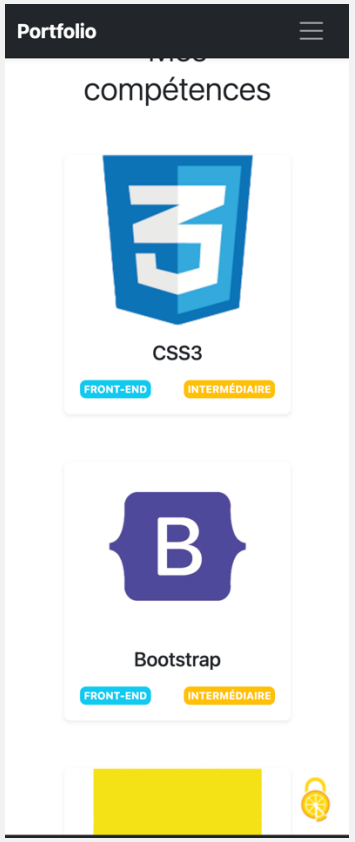
DÉVELOPPEMENT

Interface utilisateur

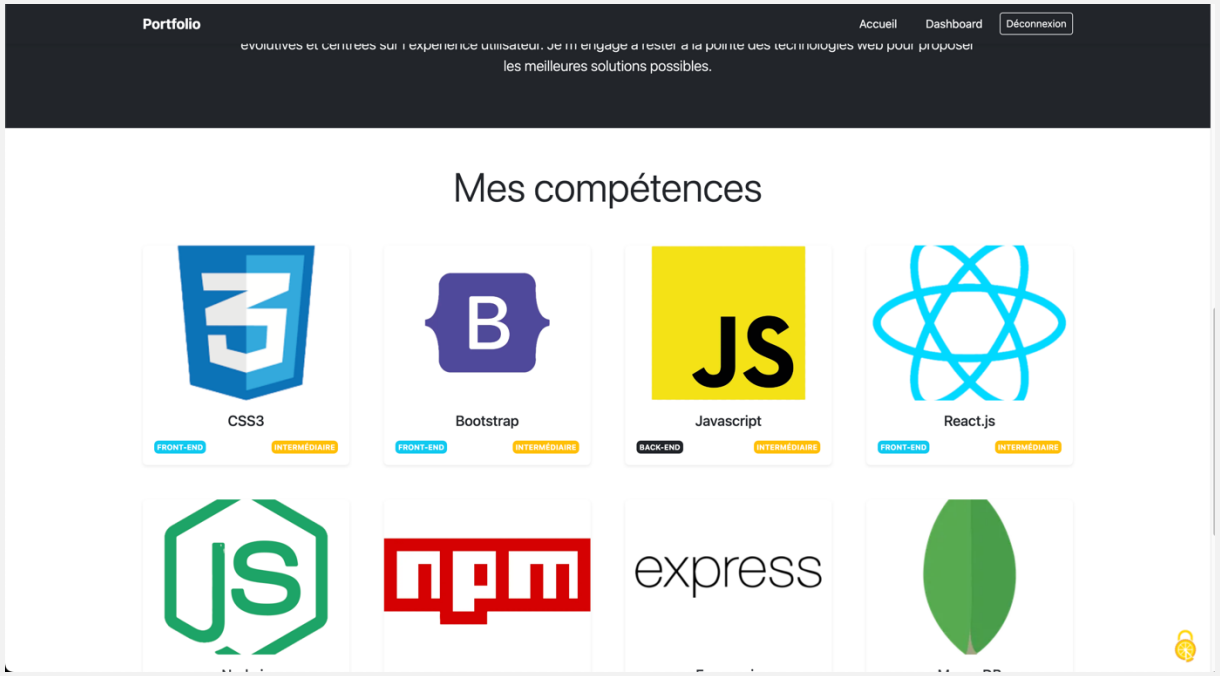
```
<ToastContainer />
{isLoading ? (
  <Container fluid className="py-5 bg-white">
    <Container className="text-center">
      <h1 className="text-center display-4 mb-5">Mes compétences</h1>
      <div
        className="d-flex justify-content-center align-items-center"
        style={{ minHeight: "200px" }}
      >
        <div className="spinner-border text-primary" role="status">
          <span className="visually-hidden">Chargement...</span>
        </div>
      </div>
    </Container>
  </Container>
) : (
  <Container fluid className="py-5 px-5 bg-white">
    <Container>
      <h1 className="text-center display-4 mb-5">Mes compétences</h1>
      <Row xs={1} sm={2} md={3} lg={4} className="g-5">
        {skills.map((skill) => (
          <Col key={skill._id}>
            <SkillCard skill={skill} />
          </Col>
        ))}
      </Row>
    </Container>
  </Container>
```



DÉVELOPPEMENT

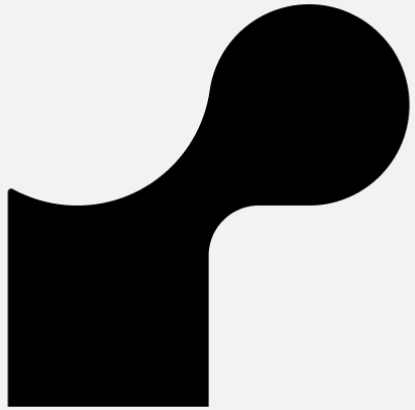


Mobile first & responsive



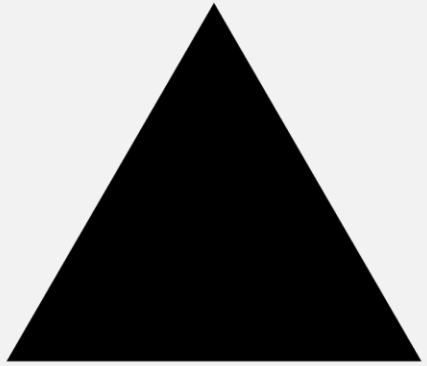
DÉPLOIEMENT

Back-end



Lien ici

Front-end

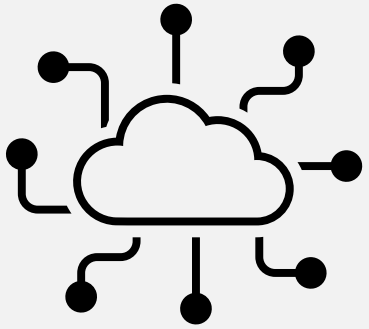


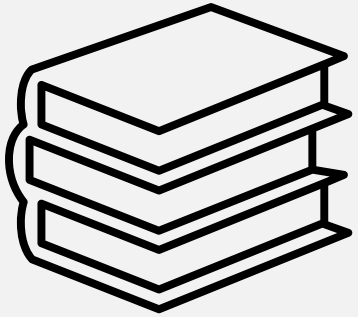
Lien ici

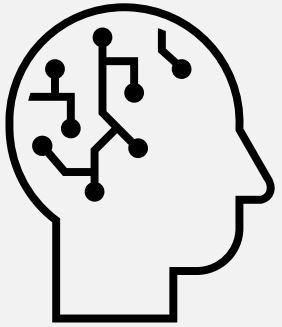
CONCLUSION

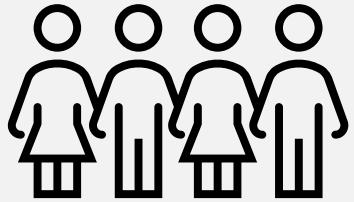
- Ce projet m'a permis de mettre en application une grande partie de ce que j'ai appris durant cette formation.
- Ce projet va rester et je vais y apporter plusieurs améliorations dans le futur pour en faire mon portfolio personnel
- Je souhaite continuer vers un CDA qui est une suite logique à ce Titre Pro

RESSOURCES









MERCI POUR VOTRE ATTENTION