

# DOSSIER PROFESSIONNEL (DP)

*Nom de naissance*

► KERACHI

*Nom d'usage*

► KERACHI

*Prénom*

► KENZO

*Adresse*

► 1 rue de la ferme Goffaert – Res. La Roseaie  
Bat. Carla – Appt.19 - 59192 Beuvrages

## Titre professionnel visé

Développeur web et web mobile

### MODALITE D'ACCES :

- ☒ Parcours de formation
- ☐ Validation des Acquis de l'Expérience (VAE)

## Présentation du dossier

Le dossier professionnel (DP) constitue un élément du système de validation du titre professionnel.  
**Ce titre est délivré par le Ministère chargé de l'emploi.**

Le DP appartient au candidat. Il le conserve, l'actualise durant son parcours et le présente **obligatoirement à chaque session d'examen.**

Pour rédiger le DP, le candidat peut être aidé par un formateur ou par un accompagnateur VAE.

Il est consulté par le jury au moment de la session d'examen.

### Pour prendre sa décision, le jury dispose :

1. des résultats de la mise en situation professionnelle complétés, éventuellement, du questionnaire professionnel ou de l'entretien professionnel ou de l'entretien technique ou du questionnement à partir de productions.
2. du **Dossier Professionnel** (DP) dans lequel le candidat a consigné les preuves de sa pratique professionnelle.
3. des résultats des évaluations passées en cours de formation lorsque le candidat évalué est issu d'un parcours de formation
4. de l'entretien final (dans le cadre de la session titre).

*[Arrêté du 22 décembre 2015, relatif aux conditions de délivrance des titres professionnels du ministère chargé de l'Emploi]*

### Ce dossier comporte :

- ▶ pour chaque activité-type du titre visé, un à trois exemples de pratique professionnelle ;
- ▶ un tableau à renseigner si le candidat souhaite porter à la connaissance du jury la détention d'un titre, d'un diplôme, d'un certificat de qualification professionnelle (CQP) ou des attestations de formation ;
- ▶ une déclaration sur l'honneur à compléter et à signer ;
- ▶ des documents illustrant la pratique professionnelle du candidat (facultatif)
- ▶ des annexes, si nécessaire.

*Pour compléter ce dossier, le candidat dispose d'un site web en accès libre sur le site.*



<http://travail-emploi.gouv.fr/titres-professionnels>

## Sommaire

# DOSSIER PROFESSIONNEL (DP)

## Exemples de pratique professionnelle

<b>Intitulé de l'activité-type n° 1 : Développer la partie front-end d'une application web ou web mobile sécurisée</b>	<b>p.</b>	<b>5</b>
▶ Intitulé de l'exemple n° 1 : Installer et configurer son environnement de travail en fonction du projet web ou web mobile.....	p.	5
▶ Intitulé de l'exemple n° 2 : Maquetter des interfaces utilisateur web ou web mobile .....	p.	7
▶ Intitulé de l'exemple n° 3 : Réaliser des interfaces utilisateurs statiques web ou web mobile .....	p.	9
▶ Intitulé de l'exemple n° 4 : Développer la partie dynamique des interfaces utilisateur web ou web mobile	p	11
<b>Intitulé de l'activité-type n° 2 : Développer la partie back-end d'une application web ou web mobile sécurisée</b>	<b>p.</b>	<b>13</b>
▶ Intitulé de l'exemple n° 1 : Mettre en place une base de données relationnelle .....	p.	13
▶ Intitulé de l'exemple n° 2 : Développer des composants d'accès aux données SQL et NoSQL .....	p.	15
▶ Intitulé de l'exemple n° 3 : Développer des composants métier coté serveur .....	p.	19
▶ Intitulé de l'exemple n° 4 : Réaliser des interfaces utilisateurs statiques web ou web mobile	p.	24
<b>Titres, diplômes, CQP, attestations de formation</b> <i>(facultatif)</i>	<b>p.</b>	<b>26</b>
<b>Déclaration sur l'honneur</b>	<b>p.</b>	<b>27</b>
<b>Documents illustrant la pratique professionnelle</b> <i>(facultatif)</i>	<b>p.</b>	<b>28</b>
<b>Annexes</b> <i>(Si le RC le prévoit)</i>	<b>p.</b>	<b>29</b>

# **EXEMPLES DE PRATIQUE PROFESSIONNELLE**

ADAPTES

## Activité-type 1

Développer la partie front-end d'une application web ou web mobile sécurisée

**Exemple n°1** ► Installer et configurer son environnement de travail en fonction du projet web ou web mobile

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

- Installation des outils nécessaires :
  - Installation de l'IDE Visual Studio Code et configurations des extensions (Prettier, ESLint...)
  - Installation de Node.js pour gérer les dépendances et exécuter les scripts
  - Installation de MongoDB pour la base de données
- Configuration des outils :
  - Initialisation d'un dépôt Git pour le versionnement et création d'un dépôt GitHub pour la sauvegarde et la collaboration
  - Configuration des variables d'environnement via un fichier .env pour sécuriser les données sensibles (ports, clé MongoDB, etc..)
  - Configuration des dépendances nécessaires via npm (npm install) en fonction du projet
- Test de l'environnement :
  - Vérification que le serveur Node.js fonctionne correctement avec un simple script de test
  - Test de connexion à MongoDB pour s'assurer que la base de données est accessible

### 2. Précisez les moyens utilisés :

- Logiciels et outils
  - Visual Studio Code : Développement du code
  - MongoDB : Gestion de la base de données
  - Git/Github : Gestion du versionnement
  - Postman : Test des API back-end
- Technologies :
  - Node.js, npm : Gestion des dépendances et exécution des scripts
  - Fichier .env : Sécurisation des données sensibles

### 3. Avec qui avez-vous travaillé ?

J'ai travaillé seul.

#### 4. Contexte

Nom de l'entreprise, organisme ou association ► *Projet réalisé durant la formation*

Chantier, atelier, service ► Cliquez ici pour taper du texte.

Période d'exercice ► Du : *18/06/2024* au : *19/06/2024*

#### 5. Informations complémentaires (facultatif)

ADAPTECO

# DOSSIER PROFESSIONNEL (DP)

## Activité-type 1

Développer la partie front-end d'une application web ou web mobile sécurisée

Exemple n°2 ► Maquetter des interfaces utilisateur web ou web mobile

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

- Dans le cadre de mon projet personnel, j'ai conçu une maquette complète d'une interface utilisateur pour mon application web. J'ai conçu les maquettes en faisant ces étapes :
- Analyse des besoins utilisateurs :
  - Définition des fonctionnalités principales et des parcours utilisateurs (UX) pour répondre aux besoins identifiés.
- Création de la maquette interactive avec Figma :
  - Utilisation des outils de design pour élaborer une interface visuelle attrayante (choix des couleurs, typographie, mise en page).
  - Mise en place de liens interactifs entre les différentes pages pour simuler la navigation.
- Optimisation de l'expérience utilisateur (UX) :
  - Validation de la simplicité et de l'intuitivité du design en prenant en compte les standards de responsive design pour qu'il soit adapté aux écrans desktop et mobile.

### 2. Précisez les moyens utilisés :

- Figma
- Behance et Dribbble pour l'inspiration

### 3. Avec qui avez-vous travaillé ?

- J'ai travaillé seul.

### 4. Contexte

Nom de l'entreprise, organisme ou association ► *Projet réalisé durant la formation*

Chantier, atelier, service ► Cliquez ici pour taper du texte.

Période d'exercice ► Du : *02/10/2024* au : *10/10/2024*

## 5. Informations complémentaires *(facultatif)*

ADAPECO



## Activité-type 1

Développer la partie front-end d'une application web ou web mobile sécurisée

**Exemple n°3** ► Réaliser des interfaces utilisateurs statiques web ou web mobile

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

- Pour cet exemple, j'ai développé des cards responsives destinées à être utilisées dans une interface utilisateur d'une application web.
- Conception et développement des cards :
  - J'ai réalisé ces deux cards en utilisant HTML et CSS, tout en m'assurant qu'elles étaient responsives pour s'adapter à toutes les tailles d'écran (ordinateurs, tablettes, mobiles)
- Mise en page dynamique :
  - J'ai appliqué une approche CSS simple pour garantir l'adaptation des cards telles que "box-sizing: border-box" pour faciliter la gestion des marges et des espacements. J'ai également utilisé des variables CSS (:root) pour définir les couleurs, les tailles, et d'autres paramètres, permettant de gérer la consistance du design.
- Effet de style et animation :
  - J'ai ajouté des effets de survol (hover) pour améliorer l'interactivité des cards, notamment des ombrages (box-shadow) pour les rendre plus dynamiques visuellement.
- Test de compatibilité :
  - J'ai testé les cards sur plusieurs navigateurs (Chrome, Brave, Firefox Developer Edition, Edge) et sur différentes tailles d'écran pour m'assurer de leur bon fonctionnement.

### 2. Précisez les moyens utilisés :

Pour réaliser ces cards

- J'ai eu besoin :
  - HTML5
  - CSS3
  - Visual Studio Code
  - Les extensions Prettier et Live Server de Visual Studio Code pour faciliter l'écriture du code et visualiser les changements en temps réel

### 3. Avec qui avez-vous travaillé ?

J'ai travaillé seul.

### 4. Contexte

Nom de l'entreprise, organisme ou association ► *Projet réalisé durant la formation*

Chantier, atelier, service ► Cliquez ici pour taper du texte.

Période d'exercice ► Du : 09/07/2024 au : 10/07/2024

### 5. Informations complémentaires (facultatif)

## Activité-type 1

Développer la partie front-end d'une application web ou web mobile sécurisée

**Exemple n°4** ► Développer la partie dynamique des interfaces utilisateur web ou web mobile

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Pour cet exemple, j'ai développé une application pour changer la couleur de l'arrière-plan lors d'un clic sur un bouton. J'ai créé deux variantes : une pour générer des couleurs prédéfinies et une autre pour générer des couleurs hexadécimales.

- Structure HTML :
  - J'ai utilisé HTML pour structurer les pages de l'application (index.html et hex.html). Ces pages contiennent un en-tête de navigation, un bouton pour changer la couleur de fond, ainsi qu'un espace d'affichage du code couleur actuel.
- JavaScript pour la logique de changement de couleur :
  - Dans app.js, j'ai développé la logique qui permet de changer dynamiquement la couleur de l'arrière-plan en utilisant une liste de couleurs prédéfinies. Un bouton déclenche la génération aléatoire d'une couleur différente à chaque clic, tout en évitant de répéter deux fois la même couleur consécutive.
  - Dans hex.js, j'ai créé une logique pour générer un code hexadécimal aléatoire à chaque clic sur le bouton. Le code hexadécimal est généré de manière dynamique en combinant des caractères alphanumériques choisis aléatoirement.
- Mise en page CSS :
  - J'ai utilisé CSS pour styliser les différentes pages de l'application (style.css). J'ai stylisé la navigation, les sections, les boutons, ainsi que les couleurs de texte. J'ai aussi intégré des effets de survol pour améliorer l'interactivité et l'expérience utilisateur.

## 2. Précisez les moyens utilisés :

Pour réaliser ces cards

- J'ai eu besoin :
  - HTML5
  - CSS3
  - JavaScript (ES6)
  - Visual Studio Code
  - Les extensions Prettier et Live Server de Visual Studio Code pour faciliter l'écriture du code et visualiser les changements en temps réel.
  - Google Fonts pour les polices personnalisées.

## 3. Avec qui avez-vous travaillé ?

J'ai travaillé seul.

## 4. Contexte

Nom de l'entreprise, organisme ou association ► *Projet réalisé durant la formation*

Chantier, atelier, service ► Cliquez ici pour taper du texte.

Période d'exercice ► Du : *30/07/2024* au : *31/07/2024*

## 5. Informations complémentaires (facultatif)

## Activité-type 2

Développer la partie back-end d'une application web ou web mobile sécurisée

**Exemple n°1** ► Mettre en place une base de données relationnelle

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Pour cet exemple, j'ai créé une base de données relationnelle nommée `owners_pets` pour gérer les informations sur les propriétaires et leurs animaux de compagnie.

- Conception de la base de données :
  - J'ai conçu deux tables principales : "owners" pour les informations des propriétaires (prénom, nom, ville, email) et "pets" pour les animaux (espèce, nom, âge, propriétaire associé via une clé étrangère)
  - J'ai défini des relations entre les tables pour garantir l'intégrité des données (relation 1 to many entre owners et pets)
- Création et modification des structures :
  - J'ai utilisé des commandes SQL pour créer les tables et leurs champs, et pour ajouter des contraintes comme UNIQUE sur le champ email dans owners
  - J'ai effectué des modifications (ex : changement de type de colonnes) pour répondre aux exigences du projet.
- Insertion de données :
  - J'ai inséré des données fictives dans les deux tables pour tester les relations et valider la cohérence des informations.
- Test de fonctionnement :
  - J'ai testé les relations entre les tables en réalisant des requêtes SQL pour :
    - Ajouter des propriétaires et leurs animaux
    - Vérifier que chaque animal est correctement relié à son propriétaire
    - S'assurer qu'aucun e-mail dupliqué ne peut être ajouté

## 2. Précisez les moyens utilisés :

Pour cet exemple j'ai eu besoin de :

- PostgreSQL pour la gestion de la base de données
- SQL pour la création des tables, insertion des données
- Utilisation de ALTER TABLE pour des modifications après la création
- Interface graphique (tel que pgAdmin) et aussi du terminal PostgreSQL pour manipuler les données

## 3. Avec qui avez-vous travaillé ?

- J'ai travaillé seul sur ce projet mais j'ai pu échanger avec mes collègues de formation concernant la veille technologique que nous avons effectués et sur la gestion des risques de sécurité.

## 4. Contexte

Nom de l'entreprise, organisme ou association ► *Projet réalisé durant la formation*

Chantier, atelier, service ► Cliquez ici pour taper du texte.

Période d'exercice ► Du : 09/10/2024 au : 10/10/2024

## 5. Informations complémentaires (facultatif)

## Activité-type 2

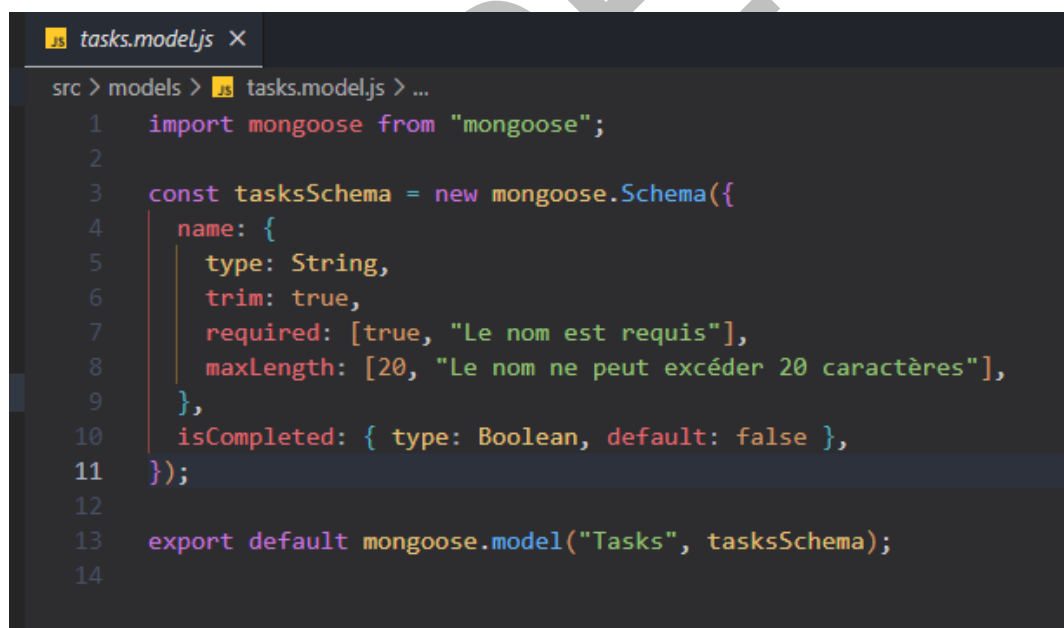
Développer la partie back-end d'une application web ou web mobile sécurisée

**Exemple n°2** ► Développer des composants d'accès aux données SQL et NoSQL

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Dans le cadre de cet exemple de projet, j'ai développé une API RESTful permettant de gérer des tâches en utilisant Node.js, Express et MongoDB. L'objectif était de proposer un backend performant, structuré et sécurisé pour effectuer des opérations CRUD (Créer, Lire, Mettre à jour, Supprimer).

- **Configuration de la base de données avec MongoDB :**
  - Utilisation de Mongoose pour définir un modèle de données (`tasks.model.js`) avec un schéma comprenant :
    - Un champ `name` (obligatoire, avec une validation sur la longueur).
    - Un champ `isCompleted` (valeur par défaut : `false`).



```
src > models > tasks.model.js > ...
1  import mongoose from "mongoose";
2
3  const tasksSchema = new mongoose.Schema({
4    name: {
5      type: String,
6      trim: true,
7      required: [true, "Le nom est requis"],
8      maxLength: [20, "Le nom ne peut excéder 20 caractères"],
9    },
10   isCompleted: { type: Boolean, default: false },
11 });
12
13 export default mongoose.model("Tasks", tasksSchema);
14
```

- **Développement des routes et des contrôleurs :**
  - Mise en place des routes dans `tasks.route.js` pour les différentes opérations CRUD.
    - Exemple : La route POST `/api/v1/tasks` permet d'ajouter une nouvelle tâche en appelant un contrôleur.
  - Création des contrôleurs dans `tasks.controller.js`, qui utilisent des services pour interagir avec la base de données :
    - `create` : Ajoute une tâche après validation.
    - `getAll` : Récupère toutes les tâches existantes.
    - `get` : Récupère une tâche par ID, avec vérification de la validité de l'ID MongoDB.

- `update` : Met à jour une tâche existante après validation des données.
- `remove` : Supprime une tâche en fonction de son ID.

- **Mise en place des services d'accès aux données :**

- Les services (`tasks.services.js`) gèrent directement l'interaction avec la base de données via Mongoose :
  - Création de nouvelles tâches (`create`).
  - Requêtes pour récupérer ou mettre à jour les données (`find`, `findByIdAndUpdate`).

- **Gestion des erreurs :**

- Création de middlewares personnalisés pour gérer les erreurs (`error-handler.middleware.js`) :
  - Retour d'un message d'erreur adapté en cas de problème (ex. : ID invalide, tâche introuvable).
  - Exemple de gestionnaire d'erreurs global :

```

src > middlewares > error-handler.middleware.js > ...
1  import { StatusCodes } from "http-status-codes";
2
3  const errorHandler = (err, req, res, next) => {
4    const customError = {
5      statusCode: err.statusCode || StatusCodes.INTERNAL_SERVER_ERROR,
6      msg:
7        err.message || "Une erreur s'est produite, veuillez réessayer plus tard",
8    };
9    res.status(customError.statusCode).json({ msg: customError.msg });
10 };
11
12 export default errorHandler;
13

```

- Middleware pour les routes inexistantes (`not-found.middleware.js`) : Retourne une erreur 404 si une route n'est pas trouvée.

- **Connexion à la base de données :**

- Gestion de la connexion MongoDB via `db.config.js` avec affichage d'un message en cas de réussite ou d'échec.

- **Test et débogage :**

- Tests des endpoints avec Postman pour valider toutes les fonctionnalités CRUD.
- Ajout de logs pour identifier et corriger les erreurs rapidement.

- **Conditions :**

- Environnement de développement local avec une configuration sécurisée (variables sensibles dans un fichier `.env`).



## 2. Précisez les moyens utilisés :

- Visual Studio Code : Écriture et organisation du code
- Postman : Test des API
- MongoDB
- Node.js
- Mongoose

## 3. Avec qui avez-vous travaillé ?

- Sur ce projet, j'ai travaillé en collaboration avec le graphiste qui s'est occupé de la réalisation des wireframes, de l'établissement de la charte graphique, de la création des éléments graphiques, et de la création des pages statiques.
- Nous avons travaillé ensemble sur l'analyse des besoins clients, sur la création de l'arborescence du site durant les différentes phases, ainsi que sur la définition des priorités et l'ordre des tâches à effectuer.

#### 4. Contexte

Nom de l'entreprise, organisme ou association ► *Projet réalisé durant la formation*

Chantier, atelier, service ► Cliquez ici pour taper du texte.

Période d'exercice ► Du : 17/10/2024 au : 18/10/2024

#### 5. Informations complémentaires (facultatif)

ADAPTECO

ADAPTECO

## Activité-type 2

Développer la partie back-end  
d'une application web ou web  
mobile sécurisée

Exemple n°3 ► Développer des composants métier  
coté serveur

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Dans mon projet personnel, j'ai développé les composants backend pour la gestion des utilisateurs, en mettant en œuvre la logique métier et les fonctionnalités nécessaires.

- **Modèle utilisateur (user.model.js) :**

- Création d'un schéma Mongoose structurant les données utilisateur avec des validations intégrées :
  - Champs obligatoires : firstName, lastName, email, password, role, etc.
  - Validation du format de l'email et des limites sur la longueur des champs.
  - Exemple de hachage de mot de passe avant sauvegarde :

```
// Middleware pour hacher le mot de passe avant de sauvegarder
UserSchema.pre("save", async function () {
  if (this.isModified("password")) {
    const salt = await bcrypt.genSalt();
    this.password = await bcrypt.hash(this.password, salt);
  }
});
```

- **Mise en place de méthodes utilisateur :**

- Création de jetons JWT pour l'authentification (createAccessToken).
- Comparaison des mots de passe lors de la connexion (comparePasswords).

- **Contrôleurs utilisateur (user.controller.js) :**

- Gestion des opérations métier principales :
  - Inscription (register) : Validation des données avec Zod (RegisterUserSchema) et vérification de l'unicité de l'email
  - Connexion (login) : Validation des identifiants et création de jetons JWT.
  - Récupération de profil (getMe) : Retourne les informations utilisateur pour les sessions authentifiées.
  - Déconnexion (logout) : Suppression du cookie de session.

- Gestion des erreurs spécifiques via des codes HTTP appropriés (400, 401, 404).

- **Services utilisateur (user.service.js) :**

- Création d'un service pour centraliser les interactions avec la base de données via Mongoose :
  - Fonctions principales : create, get, getAll, update, remove.

- Exemple :

```
const create = async (data) => {  
  return await carsModel(data).save();  
};
```

- **Routes utilisateur (user.route.js) :**
  - Définition des endpoints pour l'API utilisateur :
    - POST /register : Inscription d'un utilisateur.
    - POST /login : Connexion.
    - GET /me : Récupération du profil utilisateur (protégé par un middleware d'authentification).
- **Middlewares personnalisés :**
  - auth.middleware.js : Vérification du JWT pour protéger les routes.

```
auth.middleware.js X  
src > middlewares > auth.middleware.js > ...  
1 import { StatusCodes } from "http-status-codes";  
2 import { verifyJWT } from "../utils/token.utils.js";  
3  
4 const authenticateUser = async (req, res, next) => {  
5   const token = req.cookies.token;  
6  
7   if (!token) {  
8     return res  
9       .status(StatusCodes.UNAUTHORIZED)  
10      .json({ message: "Authentification invalide" });  
11   }  
12  
13   try {  
14     const decoded = verifyJWT(token);  
15  
16     req.car = {  
17       car: decoded._id,  
18       userID: decoded.userID,  
19     };  
20  
21     req.user = {  
22       userID: decoded.userID,  
23       role: decoded.role,  
24     };  
25  
26     req.isLoggedIn = true;  
27     next();  
28   } catch (error) {  
29     console.error("Erreur de vérification du token :", error);  
30     return res  
31       .status(StatusCodes.UNAUTHORIZED)  
32       .json({ message: "Authentification invalide" });  
33   }  
34 };  
35  
36 export default authenticateUser;  
37
```

- validation.middleware.js : Validation des données avec Zod.

```
validation.middleware.js X
src > middlewares > validation.middleware.js > ...
1  import { z } from "zod";
2  import { StatusCodes } from "http-status-codes";
3
4  const validate = (schema) => (req, res, next) => {
5    try {
6      const ParsedBody = schema.parse(req.body);
7      req.body = ParsedBody;
8      next();
9    } catch (error) {
10     if (error instanceof z.ZodError) {
11       return res.status(StatusCodes.BAD_REQUEST).json({ errors: error.errors });
12     }
13
14     next(error);
15   }
16 };
17
18 export default validate;
19
```

## 2. Précisez les moyens utilisés :

- Node.js et Express : Gestion des contrôleurs, services et middlewares
- MongoDB avec Mongoose : Base de données NoSQL pour les utilisateurs
- Zod : Validation des données au niveau des contrôleurs
- Postman : Test des endpoints API

## DOSSIER PROFESSIONNEL (DP)

### 3. Avec qui avez-vous travaillé ?

J'ai travaillé seul

### 4. Contexte

Nom de l'entreprise, organisme ou association ► *Projet réalisé durant la formation*

Chantier, atelier, service ► Cliquez ici pour taper du texte.

Période d'exercice ► Du : 22/10/2024 au : 23/12/2024

### 5. Informations complémentaires (facultatif)

## Activité-type 2

Développer la partie back-end d'une application web ou web mobile sécurisée

**Exemple n°4** ► Documenter le déploiement d'une application dynamique web ou web mobile

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Pour cet exemple, j'ai conçu et documenté une API RESTful pour la gestion des utilisateurs et des offres d'emploi. Cette API inclut une documentation interactive pour les développeurs et a été testée en vue de son déploiement.

- **Documentation avec Swagger :**
  - Création d'un fichier swagger.yaml détaillant tous les endpoints de l'API, leurs paramètres, et les réponses attendues.
  - Intégration de Swagger UI dans l'application pour permettre aux utilisateurs de tester et comprendre les fonctionnalités de l'API depuis une interface web interactive.
- **Déploiement et sécurité :**
  - Mise en place des bonnes pratiques de sécurité pour l'API :
    - Protection contre les injections MongoDB avec express-mongo-sanitize
    - Ajout de headers de sécurité avec helmet pour réduire les vulnérabilités.
    - Mise en oeuvre d'un système de limitation de requêtes (rateLimit) pour protéger contre les attaques.
    - Déploiement sur un environnement cloud (Render).

### 2. Précisez les moyens utilisés :

- JavaScript (ES6)
- Swagger UI et yamls pour la documentation interactive
- JWT pour sécuriser les accès
- MongoDB avec Mongoose
- Visual Studio Code

### 3. Avec qui avez-vous travaillé ?

J'ai travaillé seul



## DOSSIER PROFESSIONNEL (DP)

### 4. Contexte

Nom de l'entreprise, organisme ou association ►

*Projet réalisé durant la formation*

Chantier, atelier, service

► Cliquez ici pour taper du texte.

Période d'exercice ►

Du : 16/09/2024 au : 17/09/2024

### 5. Informations complémentaires (facultatif)

## Titres, diplômes, CQP, attestations de formation

*(facultatif)*

Intitulé	Autorité ou organisme	Date
Bac STI2D	Lycée du Hainaut	01/07/2018
BTS Électrotechnique	Lycée du Hainaut	01/07/2021

### Déclaration sur l'honneur

Je soussigné(e) [prénom et nom] Kenzo KERACHI,  
déclare sur l'honneur que les renseignements fournis dans ce dossier sont exacts et que je suis  
l'auteur(e) des réalisations jointes.

Fait à Beuvrages le 10/12/2024

pour faire valoir ce que de droit.

Signature :



## Documents illustrant la pratique professionnelle

*(facultatif)*

### Intitulé

Voici quelques exemples de documents que vous pourrez ajouter à votre dossier professionnel afin d'illustrer votre pratique professionnelle. Veillez à les classer et à les numérotés afin de faciliter la lecture pour les membres du jury.

ADAPTECO

### ANNEXES

*(Si le RC le prévoit)*

Le référentiel du Titre Professionnel Développeur web et web mobile ne prévoit pas d'annexes obligatoires à ce jour.

ADAPTECO