|  |  |  |
| --- | --- | --- |
|  |  |  |
| *Nom de naissance* |  | *Entrez votre nom de naissance ici.* |
| *Nom d’usage* |  | *Entrez votre nom d’usage ici.* |
| *Prénom* |  | *Entrez votre prénom ici.* |
| *Adresse* |  | *Entrez votre adresse ici.* |
|  |  |

|  |  |
| --- | --- |
| **Titre professionnel visé** | |
|  | |
| *Cliquez ici pour entrer l’intitulé du titre professionnel visé.* | |
|  | |
| **Modalité d’accès :** | |
|  | |
|  | 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**](http://travail-emploi.gouv.fr/titres-professionnels) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Sommaire** | | | |
| **Exemples de pratique professionnelle** | | |
| **Intitulé de l’activité-type n° 1** | | **p.** | **5** |
|  |  Intitulé de l’exemple n° 1  Créative 🡪 frontend p. | p. |  |
|  |  Intitulé de l’exemple n° 2  Backend p. | p. | 11 |
|  |  Intitulé de l’exemple n° 3  fusion du back et du front p | p. | 13 |
|  |  |  |  |
| **Intitulé de l’activité-type n° 2** | | **p.** |  |
|  |  Intitulé de l’exemple n° 1  p. | p. |  |
|  |  Intitulé de l’exemple n° 2  p. | p. |  |
|  |  Intitulé de l’exemple n° 3  p | p. |  |
|  |  |  |  |
| **Intitulé de l’activité-type n° 3** | | **p.** |  |
|  |  Intitulé de l’exemple n° 1  p. | p. |  |
|  |  Intitulé de l’exemple n° 2  p. | p. |  |
|  |  Intitulé de l’exemple n° 3  p | p. |  |
|  |  |  |  |
| **Intitulé de l’activité-type n° 4** | | **p.** |  |
|  |  Intitulé de l’exemple n° 1  p. | p. |  |
|  |  Intitulé de l’exemple n° 2  p. | p. |  |
|  |  Intitulé de l’exemple n° 3  p | p. |  |
|  |  |  |  |
| **Titres, diplômes, CQP, attestations de formation** *(facultatif)* | | **p.** |  |
| **Déclaration sur l’honneur** | | **p.** |  |
| **Documents illustrant la pratique professionnelle** *(facultatif)* | | **p.** |  |
| **Annexes** *(Si le RC le prévoit)* | | **p.** |  |

**Exemples de pratique**

**professionnelle**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Activité-type** | **1** | **Cliquez ici pour entrer l’intitulé de l’activité** | | | |
| ***Exemple n°1***  | | *Cliquez ici pour entrer l’intitulé de l’exemple* | | | |
|  | | | | | |
|  | | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | |
|  | | | | | |
| Le but de ce projet était de créer un site qui permet de générer des images en ce servant d’une intelligence artificielle.  J’ai commencé par démarrer un projet react avec vite. Ensuite j’ai importé les librairies dont j’allais me servir pour ce projet, react-router-dom, file-saver, tailwind, firebase et bien d’autres.  J’allais me servir de tailwind ainsi que de bootstrap afin de faciliter grandement ma manière de faire le front.  -J’ai commencé par créer les pages du site et j’y ai implémenté les routes. Ensuite j’ai décidé de commencer par la navbar et le système d’authentification. Pour cela j’allais me servir de firebase et du servir d’authentification par email et password qu’il met à disposition.  Je m’occupe de créer les composants de bouton de connexion dans un header de l’app.jsx(la navbar)  -Je créer 2 fenêtres modales, bien plus simples et performantes que des pages, puis j’en fais une pour la connexion et une pour l’enregistrement.  Pour la modale de connexion par exemple je me sers de usestate, useref, usecontect que react met à disposition    const { modalState, toggleModals, signIn } = useContext(UserContext);    const navigate = useNavigate();    const [validation, setValidation] = useState("");    const inputs = useRef([]);    const addInputs = (el) => {      if (el && !inputs.current.includes(el)) {        inputs.current.push(el);      }    };    const formRef = useRef();    const handleForm = async (e) => {      e.preventDefault();      try {        const cred = await signIn(          inputs.current[0].value,          inputs.current[1].value        );        setValidation("");        console.log(cred);        toggleModals("close");        navigate("/create-post");      } catch {        setValidation("Wopsy, email and/or password incorrect")      }    };    const closeModal = () => {      setValidation("");      toggleModals("close");    };  Comme vous avez pu le constater je me sers d’un usercontext dans le usecontext, je vais donc le créer dans la foulée. L’idée sera de déterminer si l’utilisateur est connecté et de pouvoir agir sur le dom en fonction de sa présence ou non grâce à l’affichage conditionnel de react.  export function UserContextProvider(props) {    const signUp = (email, pwd) => createUserWithEmailAndPassword(auth, email, pwd)    const signIn = (email, pwd) => signInWithEmailAndPassword(auth, email, pwd)    const [currentUser, setCurrentUser] = useState();    const [loadingData, setLoadingData] = useState(true);    useEffect(() => {      const unsubscribe = onAuthStateChanged(auth, (currentUser) => {        setCurrentUser(currentUser)        setLoadingData(false)      })      return unsubscribe;    }, [])    // modal    const [modalState, setModalState] = useState({      signUpModal: false,      signInModal: false    })    const toggleModals = modal => {      if(modal === "signIn") {        setModalState({          signUpModal: false,          signInModal: true        })      }      if(modal === "signUp") {        setModalState({          signUpModal: true,          signInModal: false        })      }      if(modal === "close") {        setModalState({          signUpModal: false,          signInModal: false        })      }    }    return (      <UserContext.Provider value={{modalState, toggleModals, signUp, currentUser, signIn}}>        {!loadingData && props.children}      </UserContext.Provider>    )  }  Dans ce usercontext.provider je set :  modalState, toggleModals, signUp, currentUser, signIn    je créer des fonctions, des constantes et je mets ces valeurs en tant que value dans le usercontextProvider que j’exporte jusqu’en haut de l’arborescence dans le main.jsx.  Ensuite je créer le bouton de déconnexion qui aura une fonction async onclick qui utilisera la méthode signout que met à disposition firebase puis je me sers de usenavigate pour retourner dans le menu principal à la déconnexion. Le tout dans une try and catch function.  -Il ne me reste plus qu’à aller dans l’app.jsx pour tester tout ça et mettre tous ces composants.  -Pour la sécurité je n’ai pas omis de préciser que si quelqu’un se redirigeait vers la page createPost(l’url createpost) et qu’il n’était pas connecté, le contenu ne s’afficherait pas.  Ainsi, voilà à quoi ressemblait mon app :  const App = () => {    const {currentUser} = useContext(UserContext)    return (        <BrowserRouter>          <SignInModal />          <SignUpModal />          <header className='w-full flex justify-between items-center bg-white sm:px-8 px-3 py-2 border-b border-b-[#e6ebf4]'>            <Link to={"/"}>              <img src={logo} alt="logo" className='sm:w-28 w-20 pic object-contain' />            </Link>            { currentUser &&            <Link to={"/create-post"} className="font-inter sm:text-[20px] main-btn text-sm font-medium bg-[#6469ff] hover:bg-[#474ce7] mx-auto text-white px-4 py-2.5 rounded-md">                Génerer Image            </Link>            }            {            !currentUser  ?            <LoginButton /> :            <LogoutButton />            }          </header>          <main className='sm:p-8 pb-2 px-2 py-8 w-full bg-[#f9fafe] min-h-[calc(100vh-73px)]'>            <Routes>                <Route path='/' element={<Home/>} />                { currentUser &&                  <Route path='/create-post' element={<CreatePost/>} />                }            </Routes>            <Footer />          </main>        </BrowserRouter>    )  }  -Ensuite après avoir fait ceci il me restait plus qu’à m’occuper de l’apparence du site. Je me suis servi des paramètres basiques de tailwind en paramétrant la couleur, les width, color, bg color et aussi du responsif en m’aidant de sm ou m sur certains composants parce que tailwind mets à disposition des média queries mobile first en min-width et est donc une bonne pratique de nos jours.    -Pour simplifier l’ux, je vais créer un loader dont je pourrais me servir dans les différentes fonctions du site pour la génération d’images par exemple.  Dans la page de création de post, l’espace réservé à ceux qui se connectent, je vais mettre en place un système qui va aller chercher des idées de phrases à rentrer si l’utilisateur n’a pas d’idées. Pour cela je vais créer un dossier constants puis un fichier index.js et stocker une cinquantaine de phrases dans une fonction surpriseMePrompts que j’exporte dans un dossier utils puis dans un autre index.js.  import { surpriseMePrompts } from '../constants'  export function getRandomPrompt(prompt) {    const randomIndex = Math.floor(Math.random() \* surpriseMePrompts.length)    const randomPrompt = surpriseMePrompts[randomIndex]    if(randomPrompt === prompt) return getRandomPrompt(prompt)    return randomPrompt;  }  Puis je m’en sers dans le fichier createpost dans une fonction handlesurpiseme qui retournera un element du tableau de la fonction.    <FormField              labelName="Génerer texte aléatoirement"              type='text'              name="prompt"              placeholder="Un robot qui mange une pomme"              value={form.prompt}              handleChange={handleChange}              handleSurpriseMe = {handleSurpriseMe}              isSurpriseMe            />  Formfield est un composant que j’ai créé qui sert de modèle pour les inputs de mon app.  const FormField = ({labelName, type, name, placeholder, value, handleChange, isSurpriseMe, handleSurpriseMe}) => {    return (      <div>        <div className="flex items-center gap-2 mb-2">          <label htmlFor={name}            className="block text-sm font-medium text-gray-900 my-3"          >            {labelName}          </label>          {isSurpriseMe && (            <button              type='button'              onClick={handleSurpriseMe}              className="font-semibold text-xs bg-[#b4b4b6] hover:bg-sky-700 hover:text-white p-2 rounded-[50%] text-black"            >              <svg xmlns="http://www.w3.org/2000/svg" fill="none" viewBox="0 0 24 24" strokeWidth={1.5} stroke="currentColor" className="w-6 h-6">                <path strokeLinecap="round" strokeLinejoin="round" d="M16.023 9.348h4.992v-.001M2.985 19.644v-4.992m0 0h4.992m-4.993 0l3.181 3.183a8.25 8.25 0 0013.803-3.7M4.031 9.865a8.25 8.25 0 0113.803-3.7l3.181 3.182m0-4.991v4.99" />              </svg>            </button>          )}        </div>        <input          type={type}          id={name}          name={name}          placeholder={placeholder}          value={value}          onChange={handleChange}          required          className='bg-gray-50 border border-gray-300 text-gray-900 text-sm rounded-lg          focus:ring-[#4649ff] focus:border-[#4649ff] outline-none block w-full p-3'        />      </div>    )  }  Pour l’affichage des images dans la page principale je me suis servi de display grid.  Ensuite je suis allé m’occuper du backend ou j’ai créé une api fromscratch.  Mes librairies :  -dotenv pour la sécurité  -cloudinary pour stocker mes photos de manière éco résponsable et sécurisée  -cors  -express  -mongoose comme sgbd  -nodemon  -openai comme générateur d’image par IA  J’ai dû me connecter et créer un compte sur mongodb atlas, cloudinary, openai puis j’ai dû mettre les variables d’environnement dans le fichier dotenv dédié    -Ensuite je suis allé faire l’index.js qui allait initialiser les librairies puis les routes que j’allais définir plus tard.  -Ainsi, mon index.js ressemblait à ça :  import express from "express";  import \* as dotenv from "dotenv";  import cors from "cors";  import connectDB from "./mongodb/connect.js";  import postRoutes from './routes/postRoutes.js';  import dalleRoutes from './routes/dalleRoutes.js';  dotenv.config();  const app = express();  app.use(cors());  app.use(express.json({ limit: '50mb' }));  app.use('/api/v1/post', postRoutes);  app.use('/api/v1/dalle', dalleRoutes);  app.get('/', async (req, res) => {    res.send('Hello there')  })  const startServer = () => {    try {      connectDB(process.env.MONGODB\_URL);      app.listen(8080, () => console.log('Server has started on port http://localhost:8080'))    }    catch (error) {      console.log(error)    }  }  startServer();  J’ai ensuite créé mon model mongoose.schema de post  import mongoose from "mongoose";  const Post = new mongoose.Schema({    name: {type: String, required: true},    prompt: {type: String, required: true},    photo: {type: String, required: true},  });  const PostSchema = mongoose.model('Post', Post);  export default PostSchema;  Puis mes routes, par exemple ici, voici le fichier postRoute.js  import express from "express";  import \* as dotenv from 'dotenv';  import {v2 as cloudinary} from 'cloudinary'  import Post from '../mongodb/models/post.js'  dotenv.config();  const router = express.Router();  cloudinary.config({    cloud\_name: process.env.CLOUDINARY\_CLOUD\_NAME,    api\_key: process.env.CLOUDINARY\_API\_KEY,    api\_secret: process.env.CLOUDINARY\_API\_SECRET,  })  //GET ALL POSTS  router.route('/').get(async (req, res) => {    try {      const posts = await Post.find({});      res.status(200).json({ success: true, data: posts });    } catch (err) {      res.status(500).json({success: false, message: 'fetching has failed'})    }  })  //CREATE A POST  router.route('/').post(async (req, res) => {    try {      const { name, prompt, photo } = req.body;      const photoUrl = await cloudinary.uploader.upload(photo);      const newPost = await Post.create({        name,        prompt,        photo: photoUrl.url,      });      res.status(200).json({ success: true, data: newPost });    } catch (err) {      res.status(500).json({success: false, message: "l'érreur est humaine"})    }  })  export default router;  -Mon backend était terminé il ne me restait plus qu’à introduire tout ça dans le front end.  Pour commencer je m’occupe de la fonction de génération d’image au clic. A noter que si le headers est possède des doubles quotes le code est mal compilé.    const generateImage = async () => {      if(form.prompt) {        try {          setGeneratingImg(true)          const response = await fetch('http://localhost:5000/api/v1/dalle', {          method: 'POST',          headers: {            'Content-type': 'application/json'          },          body: JSON.stringify({ prompt: form.prompt }),          })          const data = await response.json();          setForm({...form, photo: `data:image/jpeg;base64,${data.photo}`})        } catch (error) {          alert(error);        } finally {          setGeneratingImg(false);        }      } else {        alert('veuillez remplir le champ !!')      }    }  Puis je me suis occupé de la fonction handlesubmit qui partagerait l’image à tout le monde.    const handleSubmit = async (e) => {      e.preventDefault();      if(form.prompt && form.photo) {        setLoading(true);        try {          const response = await fetch('https://aipicgenerator.onrender.com/api/v1/post', {            method: 'POST',            headers: {              'Content-Type': 'application/json',            },            body: JSON.stringify(form)          })          await response.json();          navigate('/');        } catch (error) {          alert(error)        } finally {          setLoading(false);        }      } else {        alert('Svp rentrez une image à générer avant de vouloir la partager')      }    }  -Je créer une fonction qui va get les posts dans la page principale home.    useEffect(() => {      const fetchPosts = async () => {          setLoading(true);          try {            const response = await fetch('https://aipicgenerator.onrender.com/api/v1/post', {              method: 'GET',              headers: {                'Content-Type': 'application/json'              }            })            if(response.ok) {              const result = await response.json();              setAllPosts(result.data.reverse());            }          } catch (error) {            alert(error)          } finally {            setLoading(false)          }      }      fetchPosts()    }, []);  -Puis je m’occupe de l’affichage de l’image qui sera affichée pour tout le monde avec les informations de l’utilisateur qui l’a partagé ainsi que la fonction de téléchargement de la photo.    Il ne restait plus qu’à m’occuper de la fonction de recherche. J’ai donc créé une nouvelle fonction handlesearchchange    const [searchText, setSearchText] = useState('');    const [searchedResults, setSearchedResults] = useState(null);    const [searchTimeout, setSearchTimeOut] = useState(null);    const handleSearchChange = (e) => {      clearTimeout(searchTimeout)      setSearchText(e.target.value);      setSearchTimeOut(      setTimeout(() => {        const searchedResults =        allPosts.filter((item) =>        item.name.toLowerCase().includes(searchText.toLowerCase())          ||        item.prompt.toLowerCase().includes(searchText.toLowerCase()));        setSearchedResults(searchedResults);      }, 500)      )    }  On distingue clairement que je set un timeout pour que l’affichage dynamique ne soit pas instantané mais qu’il charge au bout d’un certain temps pour laisser le temps à l’utilisateur d’écrire. Dans cette fonction filtre les posts par le nom et le prompt en précisant bien tolowercase car les recherches sont sensibles à la casse.  -Voilà comment j’implémente le tout dans le front :    return (      <section className='max-w-6xl mx-auto'>        <div className="">          <h1 className='rounded-md font-extrabold text-[35px] text-center p-5 mt-6 mb-14 bg-slate-400 text-white'>CréaTive</h1>          <p className='mt-2 text-[#666e75] text-[16px] max-w[500px]'>Parcourez une collection d'images visuellement époustouflantes générées automatiquements par une AI.</p>        </div>        <div className="mt-10">          <FormField            labelName="Rechercher un post :"            type="text"            name="text"            placeholder="Rechercher une image"            value={searchText}            handleChange={handleSearchChange}          />        </div>        <div className="mt-10">          {loading ? (            <div className="flex justify-center items-center">                <Loader />            </div>          ) : (            <>              {searchText && (                <h2 className='font-medium text-[#666e75] text-xl mb-3'>                   Résultats pour : <span className='text-[#222328]'>"{searchText}"</span>                </h2>              )}              <div className="grid lg:grid-cols-4 sm:grid-cols-3 xs:grid-cols-2 grid-cols-1 gap-3">                {searchText ? (                  <RenderCards                    data={searchedResults}                    title="Aucun résultat trouvé .. Réessayez"                  />                ) : (                  <RenderCards                    data={allPosts}                    title="Aucun post créé encore"                  />                )}              </div>            </>          )}        </div>      </section>    )  }  export default Home  Et le rendu en version mobile par exemple est le suivant    -Il ne me restait plus qu’à déployer l’application sur le web. La première étape fût de mettre à jour mon projet puis de push le projet sur github. Puis je me suis inscris sur render qui se met en lien automatiquement avec mes projets git pour mettre mon api en ligne.  -Je n’ai rencontré aucun souci, tout s’est bien passé du premier coup c’était simple et bien expliqué. J’ai pu mettre les variables d’environnement une par une dans une séction dédiée et sécurisée puisque non présentes dans le git à cause du gitignore.  -Ensuite J’ai récupéré l’url de l’api pour la mettre sur mes fonctions ou je me servais du localhost :8080. Une fois que cela a été fait il ne me restait plus qu’à créer un nom de domaine avec un hébergeur sécurisé que j’ai fait à l’aide d’hostinger. J’ai build mon app puis j’ai pu mettre mon projet en ligne en drag & droppant les fichiers dist. | | | | | |
|  | | | | | |
|  | | | | |
|  | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | |
|  | | | | | |
| Je me suis aidé de la doc technique de javascript. De stack overflow sur des problèmes techniques. J’ai utilisé des bouts de code de projets que j’avais fais précédemment.  J’ai utilisé le devtools pour comprendre pourquoi certaines requêtes marchaient ou non. | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | |
|  | | | | | |
| J’ai travaillé seul sur ce projet | | | | | |
|  | | | | | |
|  | | | | |
|  | | | | | |
| **4. Contexte** | | | | | |
|  | | | |  | |
| Nom de l’entreprise, organisme ou association | | | | *studi* | |
| Chantier, atelier, service | | | *Cliquez ici pour taper du texte.* | | |
| Période d’exerciceDu : *01/01/2023*au :*30/01/2023* | | | | | |
|  | | | | |
|  | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | |
|  | | | | | |
| Mon site est disponible sur :  totywebsite.fr  Le git du projet est :  https://github.com/Toty-01/randompics | | | | | |
|  | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Activité-type** | **2** | **Cliquez ici pour entrer l’intitulé de l’activité** | | | |
| ***Exemple n° 1***  | | *Cliquez ici pour entrer l’intitulé de l’exemple* | | | |
|  | | | | | |
|  | | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | |
|  | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | |
|  | | | | | |
| **4. Contexte** | | | | | |
|  | | | |  | |
| Nom de l’entreprise, organisme ou association | | | | *Cliquez ici pour taper du texte.* | |
| Chantier, atelier, service | | | *Cliquez ici pour taper du texte.* | | |
| Période d’exerciceDu : *Cliquez ici*au :*Cliquez ici* | | | | | |
|  | | | | |
|  | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Activité-type** | **3** | **Cliquez ici pour entrer l’intitulé de l’activité** | | | |
| ***Exemple n° 1***  | | *Cliquez ici pour entrer l’intitulé de l’exemple* | | | |
|  | | | | | |
|  | | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | |
|  | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | |
|  | | | | | |
| **4. Contexte** | | | | | |
|  | | | |  | |
| Nom de l’entreprise, organisme ou association | | | | *Cliquez ici pour taper du texte.* | |
| Chantier, atelier, service | | | *Cliquez ici pour taper du texte.* | | |
| Période d’exerciceDu : *Cliquez ici*au :*Cliquez ici* | | | | | |
|  | | | | |
|  | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |

|  |  |  |
| --- | --- | --- |
| **Titres, diplômes, CQP, attestations de formation** | | |
|  | | |
| *(facultatif)* | | |
| **Intitulé** | **Autorité ou organisme** | **Date** |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |
| --- |
| **Déclaration sur l’honneur** |
|  |
|  |

Cliquez ici pour taper du texte.

Je soussigné(e) [prénom et nom] ,

déclare sur l’honneur que les renseignements fournis dans ce dossier sont exacts et que je suis l’auteur(e) des réalisations jointes.

Cliquez ici pour taper du texte.

Cliquez ici pour choisir une date

Fait à le

pour faire valoir ce que de droit.

Signature :

|  |
| --- |
| **Documents illustrant la pratique professionnelle** |
|  |
| *(facultatif)* |
| **Intitulé** |
| Cliquez ici pour taper du texte. |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

|  |
| --- |
| **Annexes** |
|  |
| *(Si le RC le prévoit)* |