|  |  |  |
| --- | --- | --- |
| *Nom de naissance* | ► | Sigaud |
| *Nom d’usage* | ► |  |
| *Prénom* | ► | Raphaël |
| *Adresse* | ► | 105 allée de Roucoulès 12200 Toulonjac |
|  |  |

| **Titre professionnel visé** | |
| --- | --- |
|  | |
| Développeur web et web mobile | |
|  | |
| **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** | | | |
| **Développer la partie front-end d’une application web ou web mobile en intégrant les recommandations de sécurité** | | **p.** | **8** |
|  | |  |  |
| **Compétences liées à l’activité :**   * Maquetter une application * Réaliser une interface utilisateur web statique et adaptable * Développer une interface utilisateur web dynamique * Réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce | |  |  |
|  | ► Intitulé de l’exemple n° 1 | p. | 8 |
|  | ► Intitulé de l’exemple n° 2 | p. | 12 |
|  | ► Intitulé de l’exemple n° 3 | p. | 18 |
|  |  |  |  |
| **Développer la partie back-end d’une application web ou web mobile en intégrant les recommandations de sécurité** | | **p.** | **25** |
|  | |  |  |
| **Compétences liées à l’activité :**   * Créer une base de données * Développer les composants d’accès aux données * Développer la partie back-end d’une application web ou web mobile * Elaborer et mettre en œuvre des composants dans une application de gestion de contenu ou e-commerce | |  |  |
|  | ► Intitulé de l’exemple n° 1 | p. | 25 |
|  | ► Intitulé de l’exemple n° 2 | p. | 27 |
|  | ► Intitulé de l’exemple n° 3 | p. | 30 |
|  |  |  |  |
|  |  |  |  |
| **Titres, diplômes, CQP, attestations de formation** *(facultatif)* | | p. | 34 |
| **Déclaration sur l’honneur** | | p. | 35 |
| **Documents illustrant la pratique professionnelle** *(facultatif)* | | p. | 36 |
| **Annexes** *(Si le RC le prévoit)* | | p. |  |

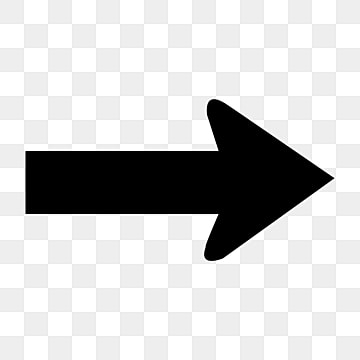
****

**Exemples de pratique**

**professionnelle**

****

|  |  | | |  | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Activité-type** | **1** | | | **Développer la partie front-end d’une application web ou web mobile en intégrant les recommandations de sécurité** | | | | |
| ***Exemple n°1*** ► | | | | Maquetter une application | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| Pour cet exercice en groupe, il nous a fallu concevoir la maquette d’un site fictif d’e-commerce pour une animalerie portant le nom d’Animalin. Nous avons donc fait l’arborescence du site, le zoning et le wireframe.  L’arborescence était pensée pour le site complet, en partant de l’accueil nous avions accès aux articles par catégorie ou par page individuelle, mais aussi au panier regroupant les articles choisis par l’acheteur, panier menant bien sûr au paiement. Une page d'inscription/connexion était aussi accessible depuis l’accueil, menant à la page du client, son profil donc, comportant une page des commandes (historique des commandes et suivi de commande en cours) et une page regroupant les différentes informations du client (adresse, informations de paiement, préférences, etc.). Enfin, les classiques pages de contact et d’information (mentions légales, etc.) sont présentes.  Sur les pages suivantes, nous avons réalisé le zoning de la page d’accueil et le wireframe. Ceux-ci sont composés d’un header, d’un corps de page et d’un footer.  Le header est composé d’un logo à gauche, du nom du magasin accompagné d’un slogan au centre et d’une partie regroupant l’inscription, la connexion et le panier à droite. En dessous sont présentes un menu en barre de navigation et une barre de recherche.  Le corps de page est lui constitué de cinq élément principaux : un bloc d'accroche pour présenter l’entreprise, d’un bloc constitué d’un carrousel d’images et en dessous d’une liste des articles. Le quatrième bloc est un bloc vertical à gauche de la liste des articles, il permet de choisir rapidement la catégorie d’article souhaitée, enfin le cinquième bloc est lui aussi vertical, mais à droite de la liste d’articles, il s’agit d’une publicité.  Le footer n’est pas détaillé sur le zoning, il reste classique : donc en trois parties. La première à gauche est composé des différentes informations concernant les ventes, d’une F.A.Q et d’une page concernant les livraisons.  Au centre, nous avons accès aux mentions légales et au service client.  La dernière partie permet de renvoyer vers la page de contact et vers le compte facebook de la boutique.  **Zoning :**    **Wireframe :** | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
| Les différents outils utilisés furent :   * **gloomaps** (<https://www.gloomaps.com/>)      * **Pencil**      * **Balsamiq Wireframes** | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| Travail en groupe lors d’un cours de maquettage avec :  Yasmeen, Quentin et Yann. | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association** ► | | | | | | **ADRAR** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | | ► | Cours | | | | | |
| **Période d’exercice** | | ► | **Du** | | 20/12/21 | | **au** | 21/12/21 |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| Cliquez ici pour taper du texte. | | | | | | | | |



| **Activité-type** | **1** | | | **Développer la partie front-end d’une application web ou web mobile en intégrant les recommandations de sécurité** | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Exemple n°2*** ► | | | | Réaliser une interface utilisateur web statique et adaptable | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| Au cours de mon stage, j’ai dû mettre en place un site vitrine responsive, statique via wordpress. Si le site a été réalisé via Wordpress, l’aspect responsive du site a lui été fait uniquement via CSS car il s’agissait pour moi de la solution la plus efficace et de l’outil que je connaissais le mieux. Je vais donc présenter ici la page d’accueil sous ses trois principales formes, une version pour écran d’ordinateur classique, une version tablette et une version mobile.    Ici les informations importantes sont : le menu qui va changer dès la version tablette, de même pour la section des chiffres clés, le carrousel sera réduit et le footer commencera à changer lui aussi.  **Voici la version tablette, à 912 pixel de large :**      Si le menu change grâce aux réglages qu’offre Wordpress, le reste du site va se transformer via la requête css : **@media**. Les chiffres clés passent de trois blocs indépendants à un seul bloc - sur la demande du client.    Ici on note donc qu’à partir de 1050 pixels de large, cet extrait de css influe la page. Les trois blocs précédents passent en **display : none** et on fait apparaître le bloc seul comprenant toutes les informations en adaptant le tout à la taille. On centre le bloc, et on lui attribue les différentes caractéristiques qu’avaient les trois blocs auparavant, couleurs, ombres, etc.  Si dans sa globalité, le texte ne change pas, le carousel lui va être réduit.  Ici on note la présence de 3 carrousels différents, chacun adapté à la taille de l’écran.  On empêche donc l’affichage du carrousel classique et du carrousel mobile, laissant ainsi le seul carousel adapté aux tablettes.  Ci-dessous un extrait CSS correspondant à la principale différence entre les versions classique et tablette :    On agrandit donc la taille du paragraphe pour profiter de la place laissée par le passage de trois avis à deux avis.  **Voici la version mobile à 390 pixels de large :**        En premier lieu, revenons sur le bloc des “chiffres clés”, nous avons là le même bloc que pour la version tablette en taille réduite, passant de 25em à 20em.  Par ailleurs, la taille de la vidéo est ici grandement réduite, passant de 850px de largeur à 300px. :    Enfin le carrousel subit lui aussi une dernière modification puisqu’à gauche nous avons l’affichage du carouseul **#slide\_1023** qui correspond à l’affichage tablette et à droite il disparaît pour laisser place au carrousel **#slide\_1025** carrousel avec un unique avis par slide : | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
| * **Wordpress**      * **CSS** | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| Seul | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association** ► | | | | | | **Avec Mention** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | | ► | Développement Web | | | | | |
| **Période d’exercice** | | ► | **Du** | | 07/03/22 | | **au** | 29/04/22 |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| Cliquez ici pour taper du texte. | | | | | | | | |

| **Activité-type** | **1** | | | **Développer la partie front-end d’une application web ou web mobile en intégrant les recommandations de sécurité** | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Exemple n°3*** ► | | | | Développer une interface utilisateur web dynamique | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| Il s’agissait ici de récupérer les données d’une API météo en JSON pour créer une interface présentant la météo par ville, puis par jour et enfin par heure.  Voici dans un premier temps le fichier html, comportant le formulaire permettant d’entrer le nom de la ville dont on veut connaître la météo. La div **#display** permet quant à elle d’afficher les résultats. Enfin la balise script permet de lancer le script JS.  Voici le fichier css :  Présentons maintenant le code Javascript, ici nous commençons par déclarer la constante **url** qui contiendra l’url de l’API. Ensuite on crée la fonction **displayData(data)**, celle-ci va permettre l’affichage des données requises dans la div **#display**.  Pour se faire, on utilise **document.getElementById()** qui nous permet de cibler via l’ID (ici **display**) l’élément HTML que l’on souhaite récupérer en JS. Pour s’assurer que la div que l’on souhaite remplir soit vide, on utilise **div.innerHTML =””;** qui permet donc de remplacer le contenu de la div, ici de la vider.  Ensuite nous procéderons par étapes, il nous faut respecter l’ordre “classique” d’apparition des balises HTML pour insérer chaque partie des informations de l’API.  Nous utiliserons donc **document.createElement()** déclaré dans une constante, pour ajouter un nouvel élément dans la div **display**. Dans un premier temps un H2 permettra d’afficher la ville et le pays (**data.city\_info.name** et **data.city\_info.country**).  Dans un H3, ce seront la date et l’heure (**data.city\_info.date** et **data.city\_info.hour**), le H4 servira de titre sans récupération de données via l’API. Une balise **img** permettra de récupérer l'icône correspondant au temps qu’il fait à l’heure de la requête. Enfin la balise **p** va contenir les températures, la vitesse du vent, sa direction et enfin la condition.  Précisons par ailleurs, qu’en utilisant **innerHTML**, on ajoute du contenu dans la balise créée au-dessus (via **document.createElement**), balise contenue dans une constante en JS. Chaque occurrence de **div.appendChild()** permet d’ajouter le HTML créé en JS dans la div **display** du fichier HTML. Donc, on déclare une constante contenant une balise, on insère le contenu dans la balise et on ajoute ensuite **dynamiquement** la balise dans le fichier HTML.  Dans la suite du code, on commence par déclarer une tableau **days** au sein duquel on insère les quatre jours suivant le jour de la requête (le jour de la requête étant **data.fcst\_day\_0** les suivants commencerons par 1) pour afficher des prévisions.  Ensuite, nous utiliserons ce tableau dans une **boucle for in**, ainsi à chaque **day** contenu dans le tableau **days** on va afficher les informations récupérées en dessous. La boucle va continuer jusqu’à la fin du tableau.  Le même processus qu'au-dessus prend forme, ici on affiche dans un élément H5 le jour de la semaine et la date (**days[day].day\_long** et **days[day].date**). L’image permettra d’afficher l'icône liée au temps (ensoleillé, pluvieux, nuit claire, etc.) de chaque jour (**days[day].icon\_big**). Le paragraphe créé contiendra la condition, les températures maximale et minimale (**days[day].condition** et **days[day].tmax** et **days[day].tmin**).  Ici aussi nous utilisons **document.createElement** pour créer un élément HTML, **innerHTML** ou **setAttribute** pour ajouter un contenu à la balise HTML et enfin **appendChild** pour injecter la balise et son contenu dans le fichier HTML **dynamiquement**.    La deuxième **boucle for** servira à récupérer les heures et la météo de chaque heure en évitant de se répéter. Donc nous déclarons **i** que nous mettons à zéro, la boucle s’arrêtera lorsque **i** sera égal à 23 (donc 24h vu que l’on part de 0) et **i** s’incrémentera à chaque tour de boucle. Au sein de cette boucle, nous déclarons **hour** qui contiendra chaque heure (**days[day].hourly\_data[`${i}H00`]** en ré-utilisant **i** pour incrémenter automatiquement cette heure à chaque tour de boucle. Nous déclarons ensuite une constante **ol** dans laquelle on crée un élément de liste ordonnée. Dans cette liste nous insérons l’heure et la météo précise de l’heure (**${i}H00` : ${hour.CONDITION}**), on récupère la météo en réutilisant la variable **hour** précédemment créée. Enfin tout ceci est inséré **dynamiquement** dans l’HTML à chaque tour de boucle via **appendChild**.  Dans cette partie, on crée la fonction qui va nous permettre, grâce à l’écoute sur le click ( ligne 89), de récupérer les informations rentrées par l’utilisateur (le nom de la ville) et de lancer la requête auprès de l’API.  Dans la constante **city**, on récupère l’**input** du formulaire, contenant donc le nom de la ville (**form input[name=’city’]**). Grâce à la méthode **fetch()**, on récupère la constante **url** et **city.value** (la concaténation **.value** permet de cibler la valeur contenue dans l’input de l’HTML).  Ensuite, lorsque l’on reçoit une réponse, elle est traitée en JSON et les données sont traitées par la fonction **displayData(data)** expliquée en premier lieu.  Ci-dessous, les captures d’écran du résultat | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
|  | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| En groupe, avec Yasmeen et Nausicaa. | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association** ► | | | | | | **ADRAR** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | | ► | Cours | | | | | |
| **Période d’exercice** | | ► | **Du** | | 16/12/2021 | | **au** | 19/12/2021 |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| Cliquez ici pour taper du texte. | | | | | | | | |

****

|  |  | | |  | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Activité-type** | **2** | | | **Développer la partie back-end d’une application web ou web mobile en intégrant les recommandations de sécurité** | | | | |
| ***Exemple n°1*** ► | | | | Créer une base de données  Développer les composants d’accès aux données  Développer la partie back-end d’une application web ou web mobile | | | | |
|  | | | |  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | | | | |
| Il va s’agir ici de **créer une base de données** dans laquelle on insérera des données de base.  Pour commencer voici les **MCD/MLD** qui servira à construire la BDD :  Nous aurons trois tables : musicien, instrument et groupe, et une table d’association : appartenir (les cardinalités en **0,n / 0,n** impliquent la création de cette table puisque les deux tables vont avoir besoin de la clé étrangère de l’autre).  La table instrument contiendra la clé étrangère **id\_musicien** car un instrument appartient à un musicien (cardinalité en **0,1**).  Passons donc à la création, **create database** permet de créer la base de donnée, **character set** et **collate** vont permettre de définir le format précis de celle-ci. **Use** indique que l’on va se servir de la bdd que l’on vient de créer.  Ensuite la création des trois tables va être semblable. **Create table** pour commencer, on lui associe le nom de la table (musicien, instrument et groupe) et entre parenthèses on va définir les différents **champs** contenus dans les lignes ou **enregistrements.** Et il faut préciser le format de chaque champ, **int** (nombre entier)**, varchar** (chaîne de caractères).    Précisons tout de même ce qu’il se passe pour l’identifiant, il sera en format **int,** mais plus précisément en incrémentation automatique puisque ce n’est pas nous qui allons l’ajouter, il s’ajoute seul à chaque nouvel enregistrement. **Primary key** indique qu’il s’agira de la clé primaire, la clé permettant l’identification de l’enregistrement et **not null** est obligatoire puisque cette clé ne peut pas contenir de valeur **null**.    Ici nous allons voir les deux façon d’ajouter des clés étrangères et donc de relier les tables. Tout d’abord, grâce à **alter table** nous allons pouvoir modifier la table **instrument** en lui ajoutant **id\_musicien** en **int**. Nous voulons donc pouvoir ajouter l’identifiant du musicien à qui appartient l’instrument.  **add constraint** permet l’ajout d’une contrainte, ceci même après la création d’une table, on ajoute donc une clé étrangère **fk\_instrument\_musicien** fk (convention d’affectation, fk pour foreign key) est suivi en premier du nom de la table de clé étrangère et en second de la table de clé primaire. On indique la **foreign key** entre parenthèses puis la **references** c’est-à-dire de quelle table vient la clé étrangère.  Dans la deuxième partie de l’extrait de code, on crée une table d’association **appartenir** qui va donc associer la table **musicien** et la table **groupe**. On ajoute les champs **id\_musicien** et **id\_groupe** et **primary key** sert donc à désigner les clés primaires de chaque tables d’origine et ensuite de la même manière que précédemment, on va les désigner comme **foreign key** et donner leurs **references**.    Voici quelques exemples d’insertions d’informations dans les différents champs. En sommes ce sont des enregistrements que l’on crée. **Insert into** est suivi du nom de table et des champs à remplir entre parenthèses, **values** est suivi des valeurs insérées dans ces champs. Les trois derniers **insert into** permettent de relier les différentes clés étrangères des tables **musicien** et **groupe** entre elles, la première valeur ici correspond au musicien (1,2,3) et la deuxième correspond au groupe (1,1,1).  Partie 2 : **Développer les composants d’accès aux données**  Procédons à un affichage pour voir où nous en sommes :        Nous utilisons donc **select \* from** (\* pour tous) afin d’afficher la totalité du contenu des tables.  Voici d’autres méthodes de sélection :  Affiche les noms de groupe de la table **groupe** :  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_    Affiche le nombre d’**enregistrement** dans la table **musicien** :  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_    Cette méthode diffère un peu, ici nous voulons afficher depuis la table **musicien**, à la fois deux champs spécifiques à cette table : **prenom\_musicien** et **nom\_musicien** et ensuite un champ venu de la table **groupe** : **nom\_groupe**.  Plus exactement nous allons afficher des enregistrements avec ces champs spécifiques uniquement.  Un **inner join** est donc nécessaire pour récupérer via **id\_musicien** les champs voulus (**prenom\_musicien** et **nom\_musicien**) venu de la table **musicien** mais présent dans la table **appartenir.**  En gros, on sélectionne les musiciens faisant partie d’un groupe, on laisse de côté ceux qui ne sont pas associés à la table **groupe** via **appartenir.**  Enfin le second **inner join** permet de sélectionner les groupes associés à des musiciens, on récupère donc **nom\_groupe** dans la table **groupe** grâce à son id.    Ici, **update** va servir à mettre les données à jour, en les modifiant donc en donnant l’information à modifier via **set** et en désignant l’enregistrement exact à modifier grâce à **where**.  **Delete from** va supprimer un enregistrement en utilisant **where** pour cibler l’enregistrement.          Pour la suite du TP il nous faut attribuer un musicien aux instruments, on modifie donc l’**id\_musicien.**  Partie 2 : **Développer la partie back-end d’une application web ou web mobile**  Nous allons maintenant mettre en place un script PHP pour récupérer la base de données et afficher des informations en fonction de celle-ci. Pour se faire nous utiliserons le modèle MVC (model, vue, controller) qui permet d’organiser le code en le divisant en fonction de son usage. Le model concentre le script php principal, la vue comporte la page HTML sur laquelle on intégrera les données de façon dynamique et le controller permet de relier tous les fichiers. Nous ajouterons un fichier bdd.php qui nous permettra d’établir la connexion avec la base de données.  **Require** va donc permettre d’associer tous les fichiers en donnant leur chemin d’accès.    Ensuite, le fichier bdd.php va se connecter à la base de données **groupe\_musique**, l’identifiant ‘root’ et l’absence de mot de passe sont laissés tels quels ici pour l’exemple, en situation réelle ils seraient changés dès la création.    Le HTML créé restera vide puisque l’affichage se fera via model.php sur l’écran suivant.    Le try va nous permettre d’effectuer la requête SQL là où le catch gèrera l’exception, donc une éventuelle erreur, en fin de fichier.  la première requête va afficher le nom du groupe (il n’y en a qu’un en bdd par simplicité), on fait la requête **select … from** auprès de la bdd récupérée dans **$database**, tout ceci est stocké dans **$query** pour ensuite être traité via **fecth()** qui va permettre de récupérer les informations.  Avec un **echo** on affiche les balises HTML qui suivent et leur contenu dans la **vue**. **$groupe[‘nom\_groupe’]** est donc la variable qui va permettre d’afficher le nom du groupe après sa recherche dans la bdd.    On fait ensuite une deuxième requête pour afficher les prénoms et noms des musiciens, associés à leurs instruments via un **select from** et un **inner join** sur la table d’association **appartenir**. l’inner join permet la jointure entre la table de départ **musicien** et la table d’association, pour ne choisir que les musiciens associés aux deux tables. Le **left join** permet lui de récupérer les instruments associés aux **id\_musicien**, cette donnée étant stockée dans la table **instrument** via la clé étrangère.  Nous faisons ensuite un **echo** dans une boucle **while** pour afficher le prénom d’un musicien, son nom et son instrument et boucler pour afficher le musicien suivant, cela jusqu’à ce que la boucle arrive au dernier musicien lié à la table **appartenir** et donc au groupe associé à cette table**.**  Le résultat sur navigateur est le suivant : | | | | | | | | |
|  | | | | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | | | | |
|  | | | | | | | | |
|  | | | | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | | | | |
| Seul | | | | | | | | |
|  | | | | | | | | |
| **4. Contexte** | | | | | | | | |
|  | | | | | |  | | |
| **Nom de l’entreprise, organisme ou association** ► | | | | | | **ADRAR** | | |
|  | |  | | | | | | |
| **Chantier, atelier, service** | | ► | Cours | | | | | |
| **Période d’exercice** | | ► | **Du** | | 16/02/22i | | **au** | 18/02/22 |
|  | | | | | | | | |
|  | | | | | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | | | | | |
| Cliquez ici pour taper du texte. | | | | | | | | |

****

****

| **Titres, diplômes, CQP, attestations de formation** | | |
| --- | --- | --- |
|  | | |
| *(facultatif)* | | |
| **Intitulé** | **Autorité ou organisme** | **Date** |
| Baccalauréat série L | Éducation nationale | 2012 |
| Licence de Lettres Modernes | Université Toulouse II Jean Jaurès | 2017 |
| Formation Compétence Plus PAO | GRETA Nord Midi Pyrénées | 2021 |
| Projet Pro Numérique | Adrar Formation - Montauban | 2021 |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |
| Cliquez ici. | Cliquez ici pour taper du texte. | Cliquez ici pour sélectionner une date. |

****

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

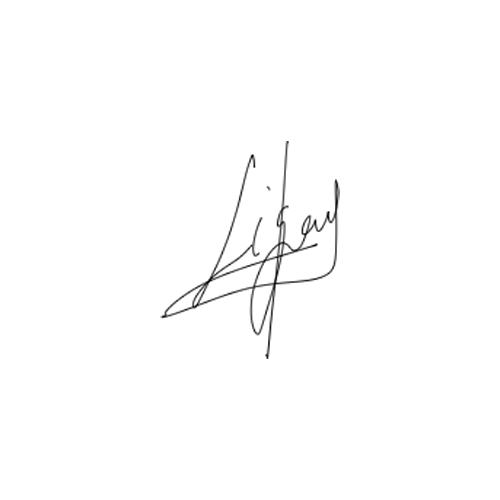


Je soussigné(e) [prénom et nom] ,

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



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)* |
|  |

****