Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE CHARGÉ
DE L'EMPLOI

DOSSIER PROFESSIONNEL (DP)

Nom de naissance - *Vigneron*
Nom d'usage -
Prénom - *Tristan*
Adresse - *44 rue Antoine Gardette bat B appt 8 63120 Courpière*

Titre professionnel visé

Graduate Développeur Flutter

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

Sommaire

Exemples de pratique professionnelle

EXEMPLES DE PRATIQUE PROFESSIONNELLE

DOSSIER PROFESSIONNEL (DP)

Activité-type 1

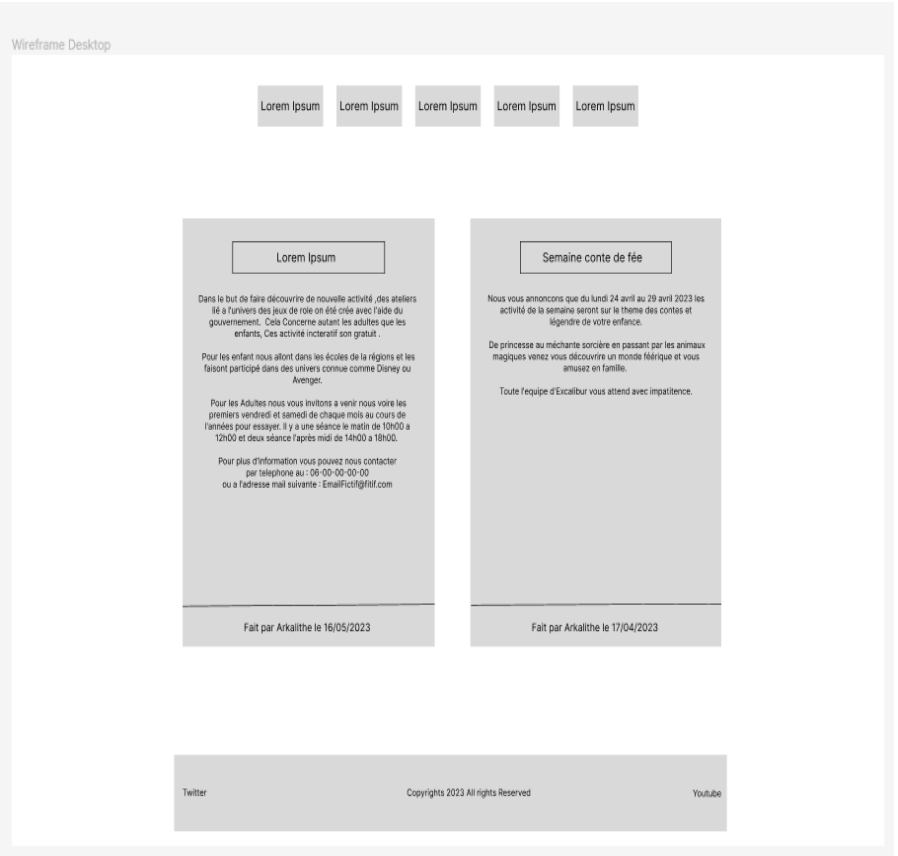
Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple n°1 ▸ *Réaliser une interface utilisateur web Responsive pour un site vitrine.*

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

Dans le contexte de notre formation, il nous a été demandé de mettre en pratique nos connaissances en langages HTML, CSS, ainsi que dans le Framework Bootstrap, à travers le développement d'un site vitrine.

Pour commencer, j'ai effectué des recherches sur d'autres sites vitrines afin de comprendre les principes d'une interface utilisateur bien conçue, ainsi que les meilleures pratiques en la matière.



Ensuite, j'ai créé le zoning et les wireframes. En utilisant les wireframes, nous avons commencé l'intégration du site vitrine.

Chaque page a été structurée de manière sémantique pour améliorer l'accessibilité et le référencement.

Voici un exemple :

```
<body class="bg-image d-flex flex-column min-vh-100 " id="bodyBg">

<nav class="navbar navbar-expand-lg navbar-dark  " id="headerNav">
    <!-- ... -->
    <ul class="navbar-nav mx-auto text-end">
        <!-- ... -->
    </ul>
</nav>

<article class="d-flex justify-content-center align-items-center pt-4">
    <section class="container display-10 row mx-5 my-5 ">
        <!-- ... -->
    </section>
</article>

<footer class="container p-2 mt-auto">
    <!-- ... -->
</footer>

</body>
```

En se basant sur l'exemple précédent, il est également visible que la conception Responsive des pages a été soigneusement prise en compte, grâce à Bootstrap.Cela nous a également facilité la création d'un site réactif grâce à son système de grille et de mise en page adaptative

2. Précisez les moyens utilisés :

Pour réaliser mon site j'ai eu besoin :
-Des langages Html et Css.
-De l'outil de prototypage Figma .
-Du Framework Bootstrap.
-De l'IDE Visual Studio Code.

Les sites Visité :
https://developer.mozilla.org/fr/docs/Web/HTML
https://getbootstrap.com/docs/5.3/getting-started/introduction/
https://99designs.fr/blog/creative-inspiration/idees-de-design-web/

3. Avec qui avez-vous travaillé ?

Je suis seul sur ce projet.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ *Studi*

Chantier, atelier, service ▸ *Projet d'évaluation d'entrainement..*

Période d'exercice ▸ Du : *Cliquez ici* au : *Cliquez ici*

5. Informations complémentaires *(facultatif)*

Activité-type 1

Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple n° 2 – *Utilisation du modèle objet de document (DOM) avec un jeu de dés.*

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

J'ai entrepris la création d'un jeu de dés interactif en utilisant le Modèle Objet de Document (DOM) pour mettre en œuvre une expérience de jeu conviviale et dynamique. Dans ce projet, j'ai exploité les fonctionnalités du DOM pour gérer l'affichage des scores, la manipulation des éléments HTML, la mise à jour de l'interface utilisateur et la réaction aux actions des joueurs.

J'ai créé des éléments HTML pour afficher le dé, les résultats, les boutons destinés à effectuer les lancers et les joueurs.

```
const playerOne = document.querySelector(".player0")
const playerTwo = document.querySelector(".player1")
const playerOneScore = document.getElementById("score0");
const playerTwoScore = document.getElementById("score1");
const playerOneCurrent = document.getElementById("current0");
const playerTwoCurrent = document.getElementById("current1");
const dice = document.querySelector('.dice');
const btnNew = document.querySelector('.btnNew');
const btnRoll = document.querySelector('.btnRoll');
const btnHold = document.querySelector('.btnHold');
```

J'ai mis en place des écouteurs d'événements pour les boutons "Roll", "Hold" et "New", qui sont les éléments clés de l'interaction avec le jeu. Lorsqu'un joueur clique sur un bouton, le dé est lancé, le résultat est affiché et le score courant du joueur actif est mis à jour en temps réel. Un exemple en cas de résultat de dé égal à 1, la fonction "changePlayer()" est appelée pour passer au joueur suivant .

```
btnRoll.addEventListener('click', function () {
    if (playing) {
        if (diceResult !== 1) {
            // Mettre à jour le score courant et l'affichage
        } else {
            changePlayer();
        }
    }
});
```

J'ai employé la modification des classes CSS pour refléter visuellement l'état du jeu. Par exemple, j'ai ajouté la classe "activePlayer" à l'élément du joueur en cours et "playerWinner" au joueur gagnant. De plus, j'ai utilisé la classe "hidden" pour masquer l'affichage du dé lorsque ce dernier n'est pas nécessaire.

En utilisant JavaScript, j'ai exploité les fonctionnalités du Modèle Objet de Document (DOM) pour créer une expérience interactive et dynamique dans mon jeu de dés. Grâce au DOM, j'ai pu accéder et manipuler les éléments HTML de la page, ce qui m'a permis de mettre à jour le contenu, de modifier les styles et de réagir aux interactions de l'utilisateur de manière fluide et en temps réel.

L'une des fonctionnalités clés du DOM est sa capacité à mettre à jour le contenu des éléments HTML. Par exemple, pour réinitialiser les scores des joueurs au début du jeu, j'ai utilisé les lignes suivantes de code :
Exemple mise à jour du contenu :

```
playerOneScore.textContent = 0;
playerTwoScore.textContent = 0;
```

De même, pour afficher le score courant d'un joueur actif, j'ai utilisé le code suivant :

```
document.getElementById(`current${activePlayer}`).textContent = currentScore;
```

Ces lignes de code ont permis de mettre à jour dynamiquement les valeurs affichées à l'écran en fonction de l'état actuel du jeu.

Une autre utilisation puissante du DOM est la possibilité de modifier les styles CSS des éléments HTML. Dans mon jeu, j'ai utilisé les classes CSS pour définir les styles des joueurs actifs et du joueur gagnant. Par exemple, pour mettre en évidence visuellement le joueur actif, j'ai ajouté la classe "activePlayer" à l'élément correspondant :

```
playerOne.classList.toggle('activePlayer');
```

Lorsqu'un joueur gagne, j'ai appliqué la classe "playerWinner" pour mettre en avant visuellement ce joueur :

```
document.querySelector(`.player${activePlayer}`).classList.add(`playerWinner`);
```

En utilisant ces classes, j'ai pu créer une expérience visuelle engageante qui reflète l'état actuel du jeu et guide les joueurs tout au long de leur expérience.

Pour rendre le jeu interactif, j'ai ajouté des écouteurs d'événements aux boutons "Hold" et aux autres actions importantes. Par exemple, lorsque le joueur appuie sur le bouton "Hold", le code réagit en conséquence pour mettre à jour les scores, vérifier les conditions de victoire et effectuer des changements de joueur si nécessaire :

```
btnHold.addEventListener('click', function () {
    // La logique pour mettre à jour les scores, vérifier la victoire, etc.
});
```

Ces écouteurs d'événements permettent aux joueurs de prendre des décisions et d'interagir avec le jeu en cliquant sur les boutons, rendant ainsi l'expérience de jeu immersive et engageante.

En combinant ces techniques, j'ai pu créer un jeu de dés interactif qui tire pleinement parti des fonctionnalités offertes par le DOM.

2. Précisez les moyens utilisés :

Pour réaliser mon site j'ai eu besoin :
-Des langages Html et Css.
-Du Javascript pour la gestion du dom .
-Du Framework Bootstrap pour le responsive.
-De l'IDE Visual Studio Code et ces extensions.

Les sites Visité :
https://developer.mozilla.org/fr/docs/Web/HTML
https://getbootstrap.com/docs/5.3/getting-started/introduction/

3. Avec qui avez-vous travaillé ?

Je suis seul sur ce projet.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ *Studi*

Chantier, atelier, service ▸ *Projet d'évaluation d'entrainement.*

Période d'exercice ▸ Du : *Cliquez ici* au : *Cliquez ici*

5. Informations complémentaires *(facultatif)*

DOSSIER PROFESSIONNEL (DP)

Activité-type 1

Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple n° 3 - *Gestion des requête Http avec axios*

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

Dans le développement d'applications web modernes, la gestion des requêtes HTTP est une compétence fondamentale. Dans le cadre de mon projet de garage automobile, j'ai dû mettre en place un système de gestion de requêtes HTTP pour assurer une communication fluide entre le front-end et le back-end de l'application.

J'ai choisi d'utiliser la bibliothèque Axios pour simplifier la gestion des requêtes HTTP. Axios offre une interface conviviale, prend en charge les promesses et fournit des mécanismes intégrés de gestion des erreurs. Pour commencer, j'ai installé Axios à l'aide de la commande suivante :

```
npm install axios
```

Pour éviter les répétitions et maintenir la flexibilité entre les environnements de développement et de production, j'ai élaboré un fichier de configuration. Ce fichier contient les URLs de base pour le serveur local et Heroku, ainsi que les en-têtes couramment utilisés. Voici comment j'ai configuré cela :

```
import axios from "axios";
const localhost = axios.create({
    baseURL:"http://localhost",
    headers : {
        'Content-Type': 'application/json'
    },
    withCredentials: true
});
const herokuUrl = axios.create({
    baseURL:"https://garagevparrotstudi-15b74863d868.herokuapp.com",
    headers : {
        'Content-Type': 'application/json'
    },
    withCredentials: true
});
const config = {
    localTestingUrl: localhost
    herokuTesting: herokuUrl
```

```
}
export default config
```

Avant d'envoyer des données au serveur, il est essentiel de valider les données côté client. Dans mon cas, j'ai effectué des vérifications à l'aide d'expressions régulières pour les adresses e-mail et les mots de passe. Si les données ne respectent pas les critères, un message d'erreur est affiché :

```
    const v1 = email_regex.test(email);
    const v2 = password_regex.test(password);
    if (!v1 || !v2) {
      setErr("Mauvaise entrée")
      return;
    }
```

Pour garantir une expérience utilisateur fluide et réactive, j'ai utilisé le hook "useEffect" dans ma mise en œuvre de la gestion des requêtes HTTP. Le hook "useEffect" m'a permis d'exécuter des effets secondaires de manière contrôlée, notamment lors du rendu initial ou lorsque certaines dépendances changent.

Un exemple lors de la construction de mon formulaire d'inscription pour mettre en évidence le champ de saisie de l'e-mail dès que la page est chargée :

```
useEffect(() => {
    emailRef.current.focus();
}, []);
```

Ou pour valider automatiquement l'adresse e-mail et le mot de passe lors de chaque changement de saisie.

```
useEffect(() => {
    const result = email_regex.test(email);
    setValidEmail(result);
}, [email]);

useEffect(() => {
    const result = password_regex.test(password);
    setValidPassword(result);
    const match = password === matchPassword;
    setValidMatchPassword(match);
}, [password, matchPassword]);
```

Un autre aspect clé de la gestion des requêtes HTTP est la soumission de données.
En prenant l'exemple précédant j'ai créé la fonction "handleSubmit" pour gérer l'envoi des données au serveur après la validation côté client. Cette fonction capture les erreurs potentielles et ajuste l'état de l'application en conséquence :

```
const handleSubmit = async (e) => {
    e.preventDefault();
    const v1 = email_regex.test(email);
    const v2 = password_regex.test(password);
    if (!v1 || !v2) {
        setErr("Mauvaise entrée");
        return;
    }
    try {
        await config.localTestingUrl.post(register_url, JSON.stringify({ email,
password }));
        setSuccess(true);
        setEmail('');
        setPassword('');
        setMatchPassword('');
    } catch (err) {
        if (!err?.response) {
            setErr('Pas de réponse serveur');
        } else if (err.response?.status === 422) {
            setErr("Email déjà utilisé");
        } else {
            setErr('Problème lors de l\'ajout de l\'employé');
        }
        errReff.current.focus();
    }
};
```

En utilisant "useEffect" et en créant "HandleSubmit", j'ai réussi à améliorer la gestion globale des requêtes HTTP dans mon application de gestion de garage automobile. Ces approches offrent une expérience utilisateur plus fluide, des retours en temps réel et une gestion efficace des erreurs lors des interactions entre le front-end et le back-end.

La gestion des erreurs est justement une partie cruciale de la gestion des requêtes HTTP. J'ai mis en place une logique pour gérer différents scénarios d'erreur lors de l'envoi de requêtes. Cela inclut l'absence de réponse du serveur, une réponse avec un code d'erreur 422 (non autorisé) et d'autres erreurs génériques. Voici comment j'ai géré ces situations :

```
try {
    await config.localTestingUrl.post(register_url, JSON.stringify({ email, password
}));
```

```
    setSuccess(true);
    setEmail('');
    setPassword('');
    setMatchPassword('');
} catch (err) {
    if (!err?.response) {
        setErr('Pas de réponse serveur');
    } else if (err.response?.status === 422) {
        setErr("Email déjà utilisé");
    } else {
        setErr('Problème Ajout employé');
    }
    errReff.current.focus();
}
```

2. Précisez les moyens utilisés :

Pour réaliser mon site j'ai eu besoin :
- Des langages Html, Css et Javascript.
- Les bibliothèques React et Axios .
- Du Framework Bootstrap pour le responsive.
- De l'IDE Visual Studio Code et ces extensions.
- Git

Les sites Visité :
- https://developer.mozilla.org/fr/docs/Web
- https://axios-http.com/docs/intro
- https://app.studi.fr
- https://git-scm.com/doc
- https://github.com/

3. Avec qui avez-vous travaillé ?

Je suis seul sur ce projet.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ *Studi.*

Chantier, atelier, service ▸ *ECF*

Période d'exercice ▸ Du : *Cliquez ici* au : *Cliquez ici*

5. Informations complémentaires *(facultatif)*

Activité-type 2

Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple n° 1 – *Conception d'une base de données*

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

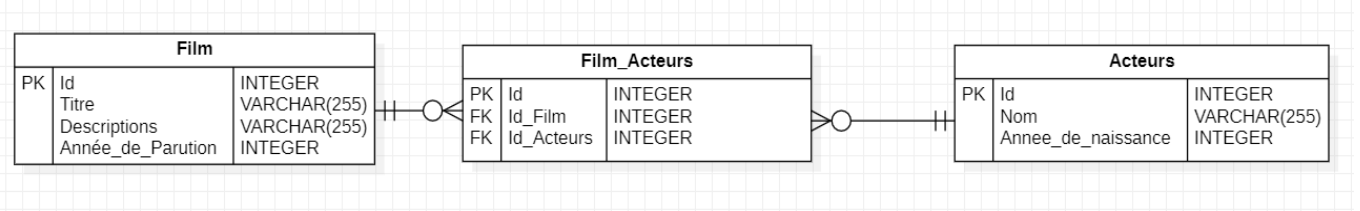
Dans le cadre de la mise en pratique de nos connaissances en Base de données, j'ai été chargé de créer une base de données pour des films.

Ma première étape a consisté à identifier les entités principales : les Films et les Acteurs, ainsi que leurs attributs. Ensuite, j'ai défini la relation entre ces entités. Un film peut avoir plusieurs acteurs, tandis qu'un acteur peut participer à plusieurs films, ce qui constitue une relation de type Many-to-Many.

En suivant une approche similaire à un schéma de classe UML, j'ai défini les entités principales avec leurs attributs et les relations entre elles. Ma conception comprend trois tables :

Pour représenter cette relation, j'ai compris que j'aurais besoin d'un attribut unique que je pourrais ré-utiliser pour identifier les Films et les Acteurs.
J'aurais pu utiliser les noms des films et des acteurs, mais il m'a semblé plus approprié d'utiliser un identifiant unique (ID) pour faciliter la lecture des tables

Une fois les Entités, les Attributs et les relations identifiés, il ne me restait plus qu'à créer la base de données.
Dans cette base de donnée j'ai créé trois table



-Une première appelée "Movie", qui contient le titre, l'année de sortie, une description/synopsis, le chemin vers l'image et un identifiant unique (ID).

```sql
CREATE TABLE Movie (
    ID INT AUTO_INCREMENT PRIMARY KEY,
    titre VARCHAR(255) NOT NULL,
    annee_sortie INT NOT NULL,
    description VARCHAR(255) NOT NULL,
    chemin_image VARCHAR(255) NOT NULL,
    CONSTRAINT unique_movie_id UNIQUE (ID)
);
```

-Une deuxième appelée "Actor", qui contient le nom de l'acteur, son année de naissance et un identifiant unique (ID).

```sql
CREATE TABLE Actor (
    ID INT AUTO_INCREMENT PRIMARY KEY,
    nom VARCHAR(255) NOT NULL,
    annee_naissance INT NOT NULL,
    CONSTRAINT unique_actor_id UNIQUE (ID)
);
```

-Enfin, j'ai créé une dernière table appelée "Movie_Actor" pour représenter la relation Many-to-Many entre les deux autres tables.

```sql
CREATE TABLE Movie_Actor (
    movie_id INT NOT NULL,
    actor_id INT  NOT NULL,
    FOREIGN KEY (movie_id) REFERENCES Movie(ID),
    FOREIGN KEY (actor_id) REFERENCES Actor(ID),
    PRIMARY KEY (movie_id, actor_id)
);
```

Pour cela, j'ai utilisé des clés étrangères faisant référence aux IDs correspondants afin d'établir les relations entre les entités.
j'ai définit la clé primaire de la table "Movie_Actor" comme une combinaison de "movie_id" et "actor_id", garantissant ainsi qu'il n'y aura pas de doublons dans les relations entre films et acteurs.

2. Précisez les moyens utilisés :

Pour réaliser ma base de donné j'ai eu besoin :
-StarUml pour les schéma de relation.
-PhpMyAddmin pour la création de la base de donnée
-serveur web Apach XAMPP

Les sites Visité :
https://docs.staruml.io/

https://www.phpmyadmin.net/docs/

3. Avec qui avez-vous travaillé ?

j'ai travaillé seul

4. Contexte

Nom de l'entreprise, organisme ou association ▸ *Studi*

Chantier, atelier, service ▸ *Exercice d'entraînement.*

Période d'exercice ▸ Du : *Cliquez ici* au : *Cliquez ici*

5. Informations complémentaires *(facultatif)*

Activité-type 2

Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple n° 2 – *Création d'une API CRUD (php)*

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

Dans le cadre d'un site de vente de voitures d'occasion, il était nécessaire de mettre en place une méthode permettant d'interagir avec les données en fonction du rôle de l'utilisateur.

J'ai créé une classe qui avec des fonction qui englobe les différentes requête SQL requises, notamment :
- Lecture des annonces de voitures.

```
public function getVoiture()
    {
        <!-- ... -->
    }
```

- Création de nouvelles annonces de voitures .

```
public function createVoiture()
    {
        <!-- ... -->
    }
```

- Modification des annonces de voitures existantes.

```
public function singleVoiture()
    {
        <!-- ... -->
    }
```

- Suppression d'annonces spécifiques.

```
public function updateVoiture()
    {
        <!-- ... -->
    }
```

Pour la Sécurité j'ai fait les actions suivantes .
- Utilisation de requêtes préparées avec PDO pour éviter les attaques par injection SQL.

Exemple :

```
    $sql = "UPDATE VOITURES
        SET
            prix = :prix,
            kilometrage = :kilometrage,
```

```
            annee_circulation = :annee_circulation,
            modele = :modele,
            nom = :nom,
            prenom = :prenom,
            numero = :numero
        WHERE
            id = :id";
```

- Nettoyage des données pour prévenir les attaques XSS.
Exemple :

```
$this->prix = htmlspecialchars(strip_tags($this->prix));
$this->kilometrage = htmlspecialchars(strip_tags($this->kilometrage));
$this->annee_circulation =
htmlspecialchars(strip_tags($this->annee_circulation));
$this->modele = htmlspecialchars(strip_tags($this->modele));
$this->nom = htmlspecialchars(strip_tags($this->nom));
$this->prenom = htmlspecialchars(strip_tags($this->prenom));
$this->numero = htmlspecialchars(strip_tags($this->numero));
$this->id = htmlspecialchars(strip_tags($this->id));
```

Ensuite, il était crucial de créer les différents points d'accès pour recevoir les requêtes du front-end.
- Exemple définition d'en-têtes pour gérer les contrôles d'accès (CORS).

```
header("Access-Control-Allow-Origin:
https://imaginative-lollipop-cdaa75.netlify.app");
header("Access-Control-Allow-Methods: GET,POST,PUT");
header('Access-Control-Allow-Credentials: true');
header("Access-Control-Allow-Headers: Content-Type,
Access-Control-Allow-Methods, Access-Control-Allow-Origin,
Access-Control-Allow-Credentials, Authorization, X-Requested-With");
```

-Exemple inclusion des différentes classes nécessaires dans le script.

```
include_once '../../Database/Connect.php';
include_once '../../Class/Voiture.php';
include_once '../../Class/Image.php';
include_once '../../Class/Equipement.php';
include_once '../../Class/Caracterstique.php';
```

- Exemple d'établissement de la connexion à la base de données.

```
$database = new DatabaseConnect();
$db = $database->dbConnectionNamed();
```

-Exemple d'instanciation des objets de différentes classes en utilisant la connexion à la base de données

($db)

```
$voiture = new Voiture($db);
$image = new Image($db);
$equipement = new Equipement($db);
$caracteristique = new Caracteristique($db);
```

-Exemple d'appel de la méthode

```
$voiture->updateVoiture()
```

2. Précisez les moyens utilisés :

Pour réaliser mon API CRUD j'ai eu besoin :
-Php
-Serveur web Apach XAMPP
-Visual Studio Code et ces extention
-Git
-Postman

Les sites Visité :
- https://www.php.net/docs.php
- https://app.studi.fr
- https://git-scm.com/doc
- https://github.com/
- https://www.w3schools.com/php/

3. Avec qui avez-vous travaillé ?

J'ai travaillé seul

4. Contexte

Nom de l'entreprise, organisme ou association ▸ *Studi*

Chantier, atelier, service ▸ *Exercise*

Période d'exercice ▸ Du : *Cliquez ici* au : *Cliquez ici*

5. Informations complémentaires *(facultatif)*

Activité-type 2

Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple n° 3 *Sécurité avec JWT (php).*

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

Je devais créer un site pour un Garage, et pour empêcher les utilisateurs non autorisés d' accéder à certaines parties du site j'ai utilisé le JWT Token.

La première chose que j'ai fait c'est d'installer la bibliothèque Firebase/php-jwt avec composer

```
composer require firebase/php-jwt
```

j'ai créé une classe pour la gestion du JWT et la première chose à faire c'est d'importer les dependance

```
require '../vendor/autoload.php';
```

Cette première ligne permet d'importer automatiquement les classes nécessaires du package Firebase\JWT

```
use Firebase\JWT\JWT;
use Firebase\JWT\ExpiredException;
use Firebase\JWT\Key;
```

J'ai définie les différentes propriétés nécessaires :

```
    protected $jwt_secrect;
    protected $token;
    protected $issuedAt;
    protected $expire;
    protected $jwt;
```

J'ai ensuite initialiser les propriétés de base de l'objet :

```
public function __construct()
    {
        date_default_timezone_set('Europe/Paris');
        $this->issuedAt = time();
        $this->expire = $this->issuedAt + 3600;
        $this->jwt_secrect = "clef_secret";
```

```
    }
```

J'ai créé la méthode qui permet de créer un token JWT encodé.

Elle crée un tableau associatif contenant les informations essentielles et les données spécifiques, puis encode ce tableau en utilisant la clé secrète fournie. Le token encodé est ensuite renvoyé.

```
public function jwtEncodeData($iss, $data)
    {
        $this->token = array(
            "iss" => $iss,
            "aud" => $iss,
            "iat" => $this->issuedAt,
            "exp" => $this->expire,
            "data" => $data
        );
        $this->jwt = JWT::encode($this->token, $this->jwt_secrect, 'HS256');
        return $this->jwt;
    }
```

j'ai fait la méthode qui prend un token JWT en entrée, tente de le décoder en utilisant la clé secrète.
- Elle vérifie si le token a expiré en comparant le moment actuel avec le moment d'expiration.
- Si le token a expiré, il supprime également le token stocké dans le stockage local du navigateur.
- Si tout est en ordre, elle renvoie les données incluses dans le token décodé.

```
public function jwtDecodeData($jwt_token)
    {
        try {
            $decode = JWT::decode($jwt_token, new Key($this->jwt_secrect ,
'HS256'));
            $currentTimestamp = time();
            if ($currentTimestamp > $decode->exp) {
                echo "<script>localStorage.removeItem('accessToken');</script>";
                throw new ExpiredException('Token has expired');
            }
            return [
                "data" => $decode->data
            ];
        } catch (ExpiredException $e) {
            return [
                "message" => $e->getMessage()
            ];
```

```
        } catch (Exception $e) {
            return [
                "message" => $e->getMessage()
            ];
        }
    }
```

Une fois cette partie la faite j'ai créé le middleware qui étend la classe précédente

```
require __DIR__.'/Class/JwtHandler.php';
class Auth extends JwtHandler
```

Il a pour but de vérifier si le token JWT est présent dans les en-têtes de la requête HTTP et s'il est valide.

Je définie les différentes propriétés nécessaires et j'initialise les propriétés de base de l'objet : :

```
    protected $db;
    protected $headers;
    protected $token;

    public function __construct($db, $headers)
    {
        parent::__construct();
        $this->db = $db;
        $this->headers = $headers;
    }
```

Ce constructeur prend en paramètres l'objet de connexion à la base de données ($db) et les en-têtes de la requête HTTP ($headers)

Je crée ensuite ma methode de verification :

```
  public function isValid()
  {
    if (array_key_exists('Authorization', $this->headers) &&
preg_match('/Bearer\s(\S+)/', $this->headers['Authorization'], $matches)) {
            $data = $this->jwtDecodeData($matches[1]);
            if (
                isset($data['data']->id) &&
                $email = $this->fetchEmail($data['data']->id)
            ) :
                return [
                    "success" => 1,
                    "email" => $email
                ];
            else :
```

```
                return [
                    "success" => 0,
                    "message" => $data['message'],
                ];
            endif;
        } else {
            return [
                "success" => 0,
                "message" => "Token not found in request"
            ];
        }
    }
```

Cette méthode vérifie si le token JWT est présent dans les en-têtes de la requête HTTP et s'il est valide. Si le token est valide et contient un identifiant utilisateur (`id`), la méthode appelle la fonction `fetchEmail()` pour récupérer l'e-mail associé à cet identifiant dans la base de données. Ensuite, elle renvoie un tableau indiquant le succès avec l'e-mail récupéré.
Si le token n'est pas valide ou s'il ne contient pas l'identifiant, elle renvoie un message d'erreur correspondant.

2. Précisez les moyens utilisés :

Pour réaliser mon API CRUD j'ai eu besoin :
-Php
-Composer
-Firebase/php-jwt
-Serveur web Apach XAMPP
-Visual Studio Code et ces extension

3. Avec qui avez-vous travaillé ?

J'ai travaillé seul

4. Contexte

Nom de l'entreprise, organisme ou association ▸ *Studi.*

Chantier, atelier, service ▸ *ECF*

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
Attestation de Formation	Studi	26/09/2022 au 01/01/2024

Déclaration sur l'honneur

Je soussigné(e),Vigneron Tristan

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 Courpière le 21/08/2023

pour faire valoir ce que de droit.

Signature :

Vigneron Tristan

Documents illustrant la pratique professionnelle

(facultatif)

Intitulé
Cliquez ici pour taper du texte.

ANNEXES

(Si le RC le prévoit)