Documentation

PROJET : QUESTIONNAIRES
AUTEUR : FOY ORIANE

Sommaire

1. Introduction

1.1. Contexte.

Dans le cadre de mon stage de première année de BTS SIO au sein de l'entreprise Dolfi Formation, j'ai participé à l'amélioration d'un site de questionnaires en ligne initialement statique, c'est-à-dire sans base de données. Ce fonctionnement limitait fortement la récupération des résultats, la maintenance des questionnaires et l'évolution de l'application. Le projet consistait à transformer ce site en une application web dynamique, capable de gérer les questionnaires, les participants et leurs réponses, tout en modernisant l'interface existante et en corrigeant les dysfonctionnements présents.

1.2. Objectifs.

Ce projet m'a permit d'appliquer concrètement les connaissances acquises en formation, notamment en PHP, MySQL et développement web. Il m'a permis de comprendre la conception et l'utilisation d'une base de données à travers la réalisation d'un MCD, ainsi que la logique de traitement et de

stockage des données. J'ai également développé mes compétences en amélioration de l'expérience utilisateur, en dynamisation d'un site web et en mise en place de fonctionnalités concrètes telles que le calcul automatique des scores, la génération de résultats et l'envoi de résultats par e-mail.

1.3. Technologies utilisées.

Langages : PHP , HTML, CSS, JavaScript.

IDE: VisualStudioCode.

2. Cahier des charges

L'application de questionnaire en ligne existante présentait plusieurs limitations, tant sur le plan technique qu'ergonomique. Le projet avait pour objectif d'améliorer l'application tout en conservant le design initial, en le modernisant afin de répondre aux standards actuels et d'améliorer l'expérience utilisateur.

Objectifs principaux :

Modernisation de l'interface existante

- Conservation de l'identité visuelle et du design de base de l'application.
- Modernisation de l'interface (mise en page, lisibilité, interactions).
- Adaptation de l'application à tous les types d'écrans (ordinateur, tablette, mobile) grâce à un design responsive, absent dans la version précédente.

Dynamisation de l'application

- Mise en place d'une base de données, inexistante auparavant, afin de stocker :
    - Les questionnaires
    - Les questions
    - Les réponses
    - Les résultats des utilisateurs
- Résolution des problèmes liés à :
    - La récupération et l'exploitation des résultats
    - La maintenance et l'évolution des questionnaires
- Mise en place d'un fonctionnement centralisé avec un seul code commun pour l'ensemble des questionnaires, reposant sur un identifiant transmis via un paramètre GET dans l'URL.

Correction des dysfonctionnements

- Correction des différents bugs présents dans l'ancienne version.

- Amélioration de la stabilité et de la fiabilité de l'application.

Amélioration du parcours utilisateur

- Transformation du questionnaire en un système à question unique par écran afin d'améliorer la clarté et la compréhension des questions.
- Navigation fluide entre les questions.

Conservation des fonctionnalités existantes

- Maintien de toutes les fonctionnalités essentielles, notamment :
    - L'envoi automatique des résultats par e-mail
    - Le calcul des scores
    - Les règles de calcul déjà en place

2.1. Besoins fonctionnels.

L'application de questionnaire en ligne doit répondre aux besoins fonctionnels suivants :

- Permettre l'accès à différents questionnaires à partir d'une URL unique, grâce à un identifiant passé en paramètre (GET).
- Afficher les questionnaires de manière dynamique à partir des données stockées en base de données.
- Proposer un fonctionnement du questionnaire avec une question affichée à la fois, afin d'améliorer la lisibilité et l'expérience utilisateur.
- Enregistrer les informations des participants ainsi que leurs réponses de manière sécurisée.
- Calculer automatiquement les scores en fonction des réponses fournies par l'utilisateur.
- Générer des résultats exploitables à la fin du questionnaire (affichage à l'écran et/ou génération de document PDF).
- Envoyer automatiquement les résultats du questionnaire par e-mail au participant ou à un destinataire défini.
- Permettre la gestion et la maintenance des questionnaires (questions, réponses, scores) sans modification du code principal.
- Garantir la conservation des fonctionnalités existantes tout en corrigeant les bugs présents dans l'ancienne version.
- Assurer une navigation fluide et intuitive sur l'ensemble de l'application.

2.2. Contraintes techniques

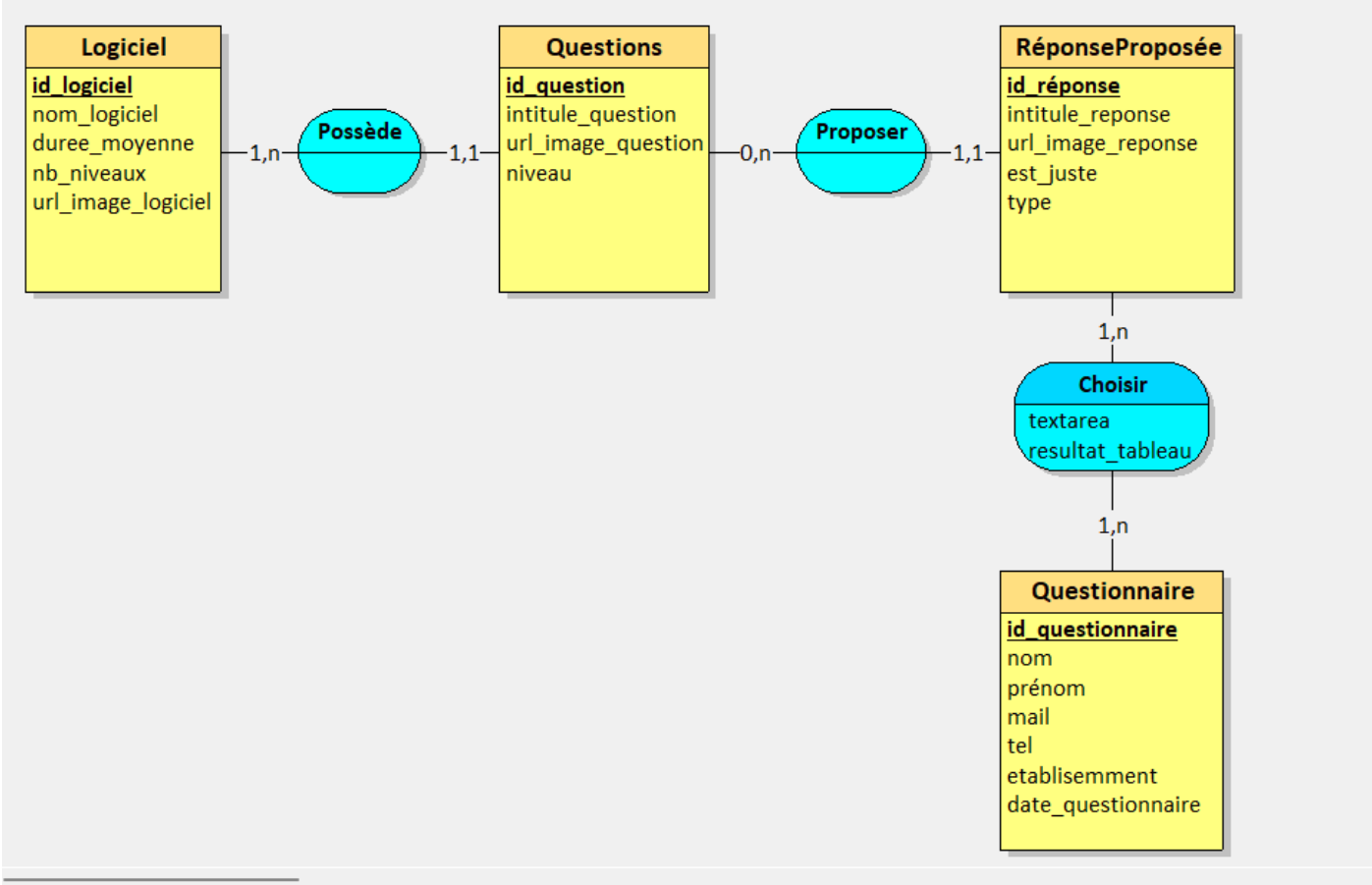
Le développement de l'application doit respecter les contraintes techniques suivantes :

- L'utilisation d'une base de données **MySQL** est requise pour la gestion des questionnaires, des questions, des réponses, des participants et des résultats.

- L’application doit fonctionner à partir d’un **code unique** pour l’ensemble des questionnaires, avec une gestion dynamique via des paramètres transmis dans l’URL (GET).
- Le design existant doit être conservé et modernisé, sans modification de l’identité visuelle de l’application.
- L’interface doit être **responsive**, afin de garantir une utilisation correcte sur ordinateur, tablette et smartphone.
- Le site doit être compatible avec les navigateurs web courants.
- Le système doit permettre l’envoi automatique d’e-mails contenant les résultats du questionnaire.
- La solution doit être facilement maintenable et évolutive, notamment grâce à une structure de code claire et à l’utilisation d’une base de données.

3. Conception du site

3.1. Modèle conceptuel de la base de données



MCD de l'application questionnaire

Entité

Logiciel : Liste des logiciels évalués.

Questionnaire : Informations sur les personnes ayant répondu aux questionnaires.

Questions : Questions pour chaque logiciel.

Reponses_proposees : Réponses possibles pour chaque question.

Associations

Possède : Un logiciel possède une ou plusieurs questions.

Proposer : Une question propose une ou plusieurs réponses possibles.

Choisir : Un participant choisit une ou plusieurs réponses pour compléter le questionnaire.

Attributs

RéponseProposée.type : textarea : Pour les réponses libres. resultat_tableau : Pour stocker un tableau sous forme de chaine de caractères (voir documentation technique).

Choisir.textarea : Pour les réponses libres, la réponse du candidat (questionnaire) est stockée sous forme de texte.

Choisir.resultat_tableau : Pour les réponses à l'origine d'un tableau les réponses du candidat sont stockées sous la forme d'une chaine de caractères qui associe sa réponse à une case du tableau

Description générale

Le MCD décrit un système de questionnaires pour différents logiciels : Chaque logiciel possède plusieurs questions, chacune de difficulté spécifique. Chaque question peut proposer plusieurs réponses, dont certaines sont correctes. Les participants remplissent un questionnaire et choisissent des réponses, éventuellement avec des commentaires libres ou en cochant les cases d'un tableau. Les données recueillies incluent les informations personnelles du participant et les résultats pour analyses ultérieures.

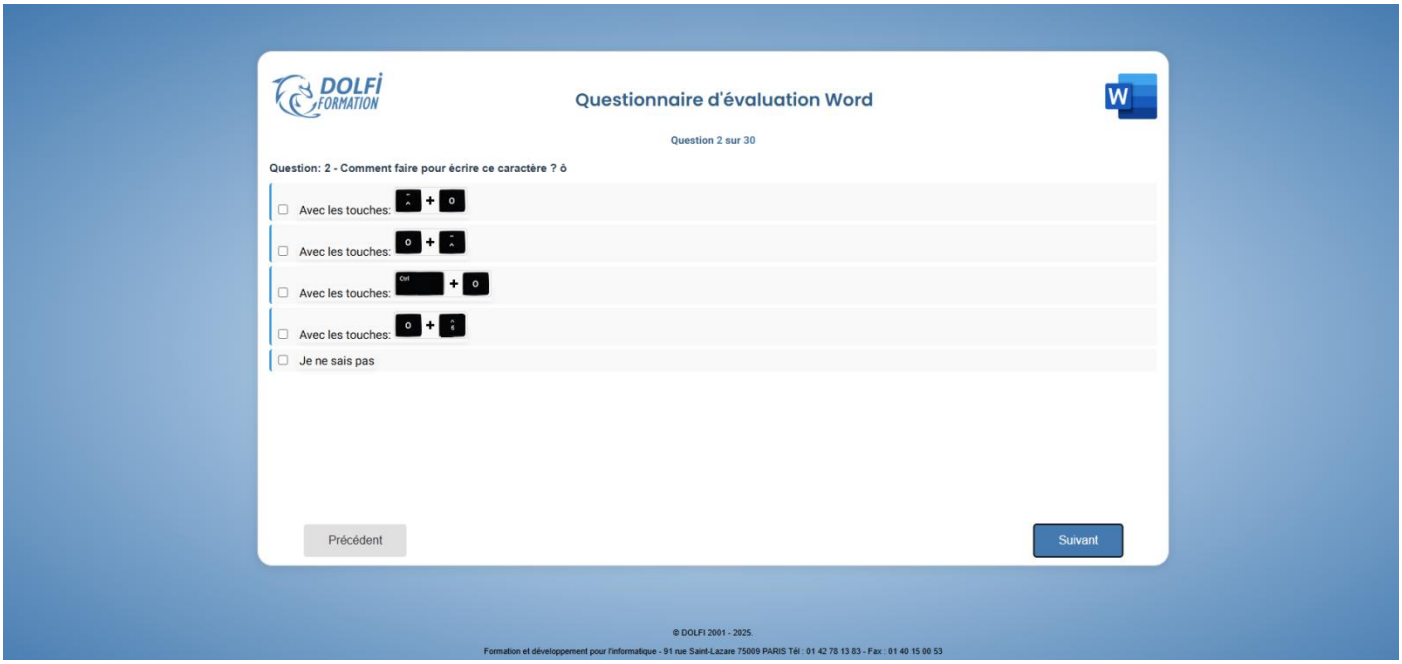
3.2. Charte graphique

Modernisation du design déjà existant

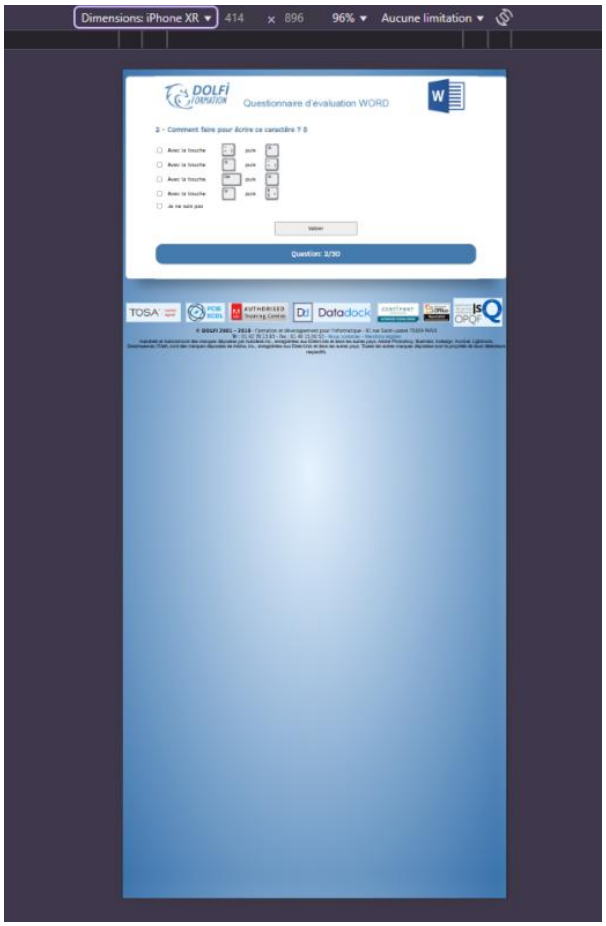
Cahier des charges relatif au design :

- Couleurs de l'entreprise : bleu
- Rectangle blanc contenant le questionnaire
- Logo de l'entreprise et logo de l'application relatif au questionnaire effectué
- Intitule de l'application relatif au questionnaire effectué
- Compteur de question affiché
- Crédits de l'entreprise dans le footer
- Créer un design pour la responsive

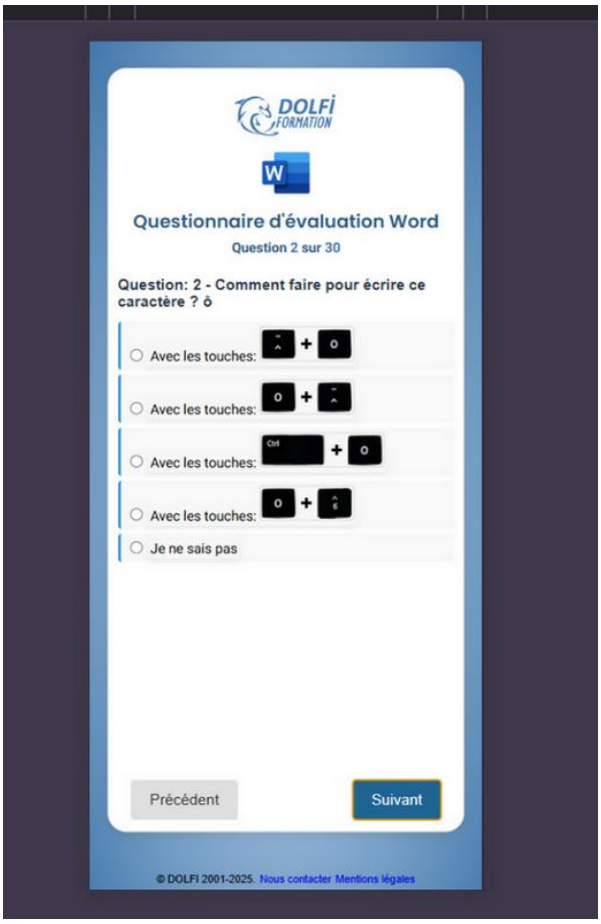
Ancien design de l'application



Nouveau design de l'application

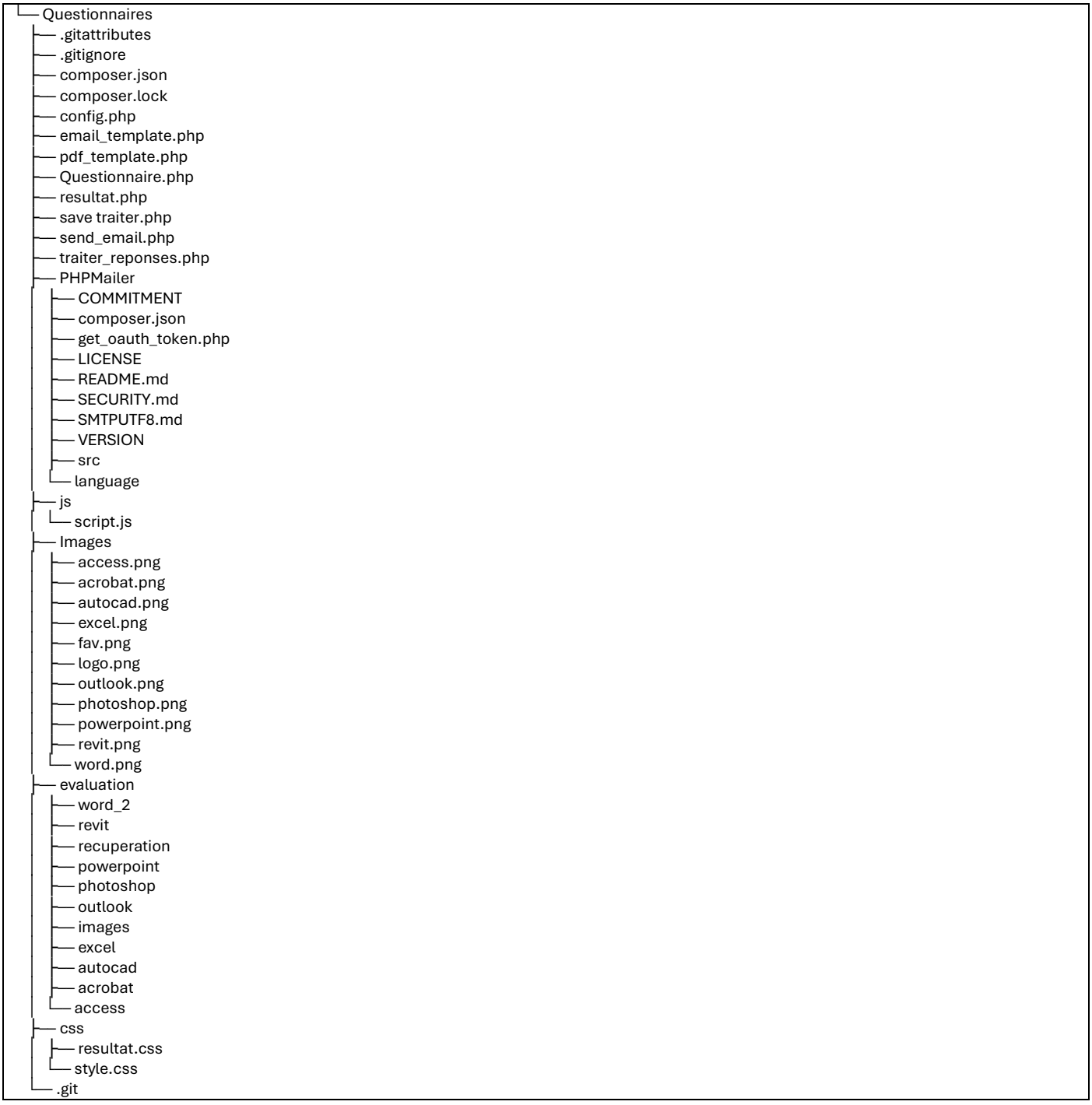


Ancien design dans le format telephone



Nouveau design dans le format telephone

3.3. Arborescence du site.



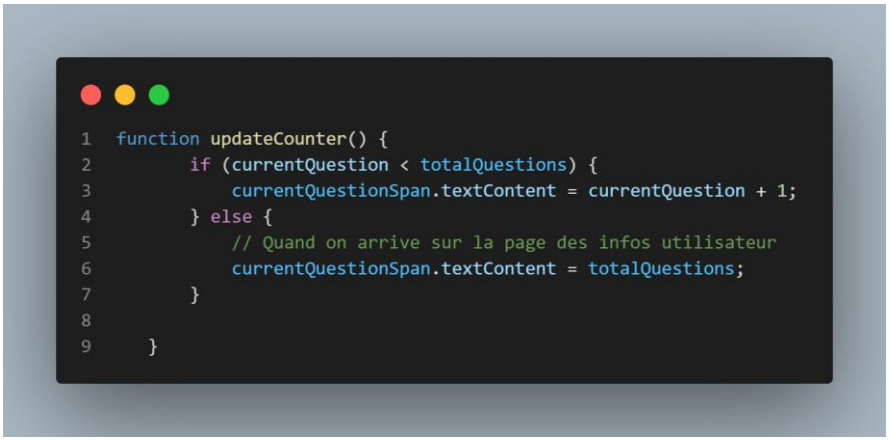
4. Documentation technique

4.1. Fonctions JavaScript

script.js

Ce fichier gère la navigation entre les questions d'un formulaire dynamique. Il permet à l'utilisateur de répondre à une question à la fois, d'avancer ou revenir en arrière, et de valider le formulaire final avec vérification des champs d'identité.

updateCounter() :



```
function updateCounter() {
        if (currentQuestion < totalQuestions) {
            currentQuestionSpan.textContent = currentQuestion + 1;
        } else {
            // Quand on arrive sur la page des infos utilisateur
            currentQuestionSpan.textContent = totalQuestions;
        }

    }
```

Fonction updateCounter

Rôle : Met à jour l'affichage du compteur de questions (ex. "Question 2 sur 5").

Entrée : Aucune

Sortie : Met à jour le texte du dans Questionnaire.php.

updateButtons() :

```
function updateButtons() {
        const isFirstQuestion = currentQuestion === 0;
        const isLastQuestion = currentQuestion === questions.length - 1;
        const isUserInfoPage = currentQuestion === questions.length;

        prevBtn.classList.toggle('hidden', isFirstQuestion && !isUserInfoPage);
        nextBtn.classList.toggle('hidden', isUserInfoPage);
        submitBtn.classList.toggle('hidden', !isUserInfoPage);

        if (isUserInfoPage) {
            userInfo.classList.remove('hidden');
            nom.classList.remove("IsNotValid");
            nomlabel.classList.remove("IsNotValidLabel");
            prenom.classList.remove("IsNotValid");
            prenomlabel.classList.remove("IsNotValidLabel");
            mail.classList.remove("IsNotValid");
            maillabel.classList.remove("IsNotValidLabel");
            compteur.classList.add('hidden');
        } else {
            userInfo.classList.add('hidden');
            compteur.classList.remove('hidden');
        }
    }
```

Fonction updateButtons

Rôle : Gère l'affichage des boutons (Précédent, Suivant, Envoyer) et du formulaire d'identification.
Entrée:/
Sortie : mise à jour du DOM (classes hidden, IsNotValid, etc.)

EventListener pour nextBtn

```
nextBtn.addEventListener('click', () => {
        questions[currentQuestion].classList.add('hidden');
        currentQuestion++;

        if (currentQuestion < questions.length) {
            questions[currentQuestion].classList.remove('hidden');
        }

        window.scrollTo(0, 0);
        updateButtons();
        updateCounter();
    });

```

EventListener pour nextBtn pour la question suivante

Rôle : Passe à la question suivante.
Entrée : clic utilisateur
Sortie : question suivante affichée, compteur et boutons mis à jour

EventListener pour PrevtBtn

```
prevBtn.addEventListener('click', () => {
        if (currentQuestion === questions.length) {
            userInfo.classList.add('hidden');
            currentQuestion--;
            questions[currentQuestion].classList.remove('hidden');
        } else {
            questions[currentQuestion].classList.add('hidden');
            currentQuestion--;
            questions[currentQuestion].classList.remove('hidden');
        }

        window.scrollTo(0, 0);
        updateButtons();
        updateCounter();

    });
```

EventListener pour PrevtBtn pour revenir à la question précédente

Rôle : Revient à la question précédente.
Entrée : clic utilisateur
Sortie : question précédente affichée, compteur et boutons mis à jour

EventListener pour submitBtn

```
submitBtn.addEventListener('click', (event) => {

    submitBtn.disabled = true;
    const checkedRadios = document.querySelectorAll('input[type="radio"]:checked');
    if (checkedRadios.length === 0 ) {
        toastr.options.timeOut = 1000;
        toastr.error("Veuillez répondre à au moins une question.", "Erreur");
        submitBtn.disabled = false;
    }else if (nom.value === "" || mail.value === "" || prenom.value === "") {
        toastr.options.timeOut = 1000;
        toastr.error("Veuillez remplir tous les champs d'identités *.", "Erreur");
        submitBtn.disabled = false;
        if (nom.value === "") {
            nom.classList.add("IsNotValid");
            nomlabel.classList.add("IsNotValidLabel");
        }
        if (prenom.value === "") {
            prenom.classList.add("IsNotValid");
            prenomlabel.classList.add("IsNotValidLabel");
        }
        if (mail.value === "") {
            mail.classList.add("IsNotValid");
            maillabel.classList.add("IsNotValidLabel");
        }
    }else{



        formulaire.submit();
    }
     });
```

EventListener pour submitBtn pour vérifier si tous les champs obligatoires sont remplis avant de submit

Rôle : Vérifie que toutes les informations sont remplies avant d'envoyer le formulaire.
Entrées : valeurs des champs nom, prénom, mail + cases cochées
Sortie : affichage d'erreurs via toastr ou soumission du formulaire.

Resultat.php

AddEventListenerPogressBar

```php
// Animation de la barre de progression
document.addEventListener('DOMContentLoaded', () => {
    const progressBar = document.getElementById('progressBar');
    const pourcentage = <?= $pourcentage ?>;

    setTimeout(() => {
        progressBar.style.width = pourcentage + '%';
        progressBar.textContent = pourcentage + '%';
    }, 100);
});
```

AddEventListenerPogressBar

Rôle : Animer la barre de progression pour qu'elle reflète le score.

Entrée : $pourcentage → pourcentage de bonnes réponses (PHP injecté dans JS).

Sortie :La largeur de la barre augmente progressivement jusqu'au pourcentage calculé.

Le texte de la barre affiche le pourcentage réel.

4.2. Fonctions php par page

Role :Créer le template du corp du mail envoyé.

email_template.php

Le template sert à afficher les résultats d'un questionnaire pour un participant, dans un corps d'email en HTML envoyé automatiquement à un utilisateur choisi.
Il affiche :

- Les informations du participant,
- Le score par niveau (Initiation, Intermédiaire, Perfectionnement, Expertise),
- Une barre de progression visuelle pour chaque niveau,
- Un message automatique et la signature DOLFI Formation.

```php
<?php if (isset($nbreponseInit)): ?>
            <div class="score">
                Score niveau Initiation : <?= $nbreponseInit ?> / <?= $nbquestionInit ?>
            </div>
            <div class="progress-container">
                <div class="progress-bar bar-init" style="width: <?= $pourcentageInit ?>%;">
                    <?= $pourcentageInit ?>%
                </div>
            </div>
        <?php endif; ?>

        <?php if (isset($nbreponseInter)): ?>
            <div class="score">
                Score niveau Intermédiaire : <?= $nbreponseInter ?> / <?= $nbquestionInter ?>
            </div>
            <div class="progress-container">
                <div class="progress-bar bar-inter" style="width: <?= $pourcentageInter ?>%;">
                    <?= $pourcentageInter ?>%
                </div>
            </div>
        <?php endif; ?>

        <?php if (isset($nbreponsePerf)): ?>
            <div class="score">
                Score niveau Perfectionnement : <?= $nbreponsePerf ?> / <?= $nbquestionPerf ?>
            </div>
            <div class="progress-container">
                <div class="progress-bar bar-perf" style="width: <?= $pourcentagePerf ?>%;">
                    <?= $pourcentagePerf ?>%
                </div>
            </div>
        <?php endif; ?>

        <?php if (isset($nbreponseExp)): ?>
            <div class="score">
                Score niveau Expertise : <?= $nbreponseExp ?> / <?= $nbquestionExp ?>
            </div>
            <div class="progress-container">
                <div class="progress-bar bar-exp" style="width: <?= $pourcentageExp ?>%;">
                    <?= $pourcentageExp ?>%
                </div>
            </div>
        <?php endif; ?>
```

Code php pour gérer les différentes barres de progression dans le corp du mail

Rôle : Afficher le score du participant pour chaque niveau (Initiation, Intermédiaire, Perfectionnement, Expertise) et représenter visuellement sa performance avec une barre de progression.

Entrées :

- $nbreponseInit / $nbquestionInit : nombre de réponses correctes et nombre total de questions pour le niveau Initiation.

- $nbreponseInter / $nbquestionInter : idem pour le niveau Intermédiaire.
- $nbreponsePerf / $nbquestionPerf : idem pour le niveau Perfectionnement.
- $nbreponseExp / $nbquestionExp : idem pour le niveau Expertise.
- $pourcentageInit, $pourcentageInter, $pourcentagePerf, $pourcentageExp : pourcentage de réussite calculé pour chaque niveau.

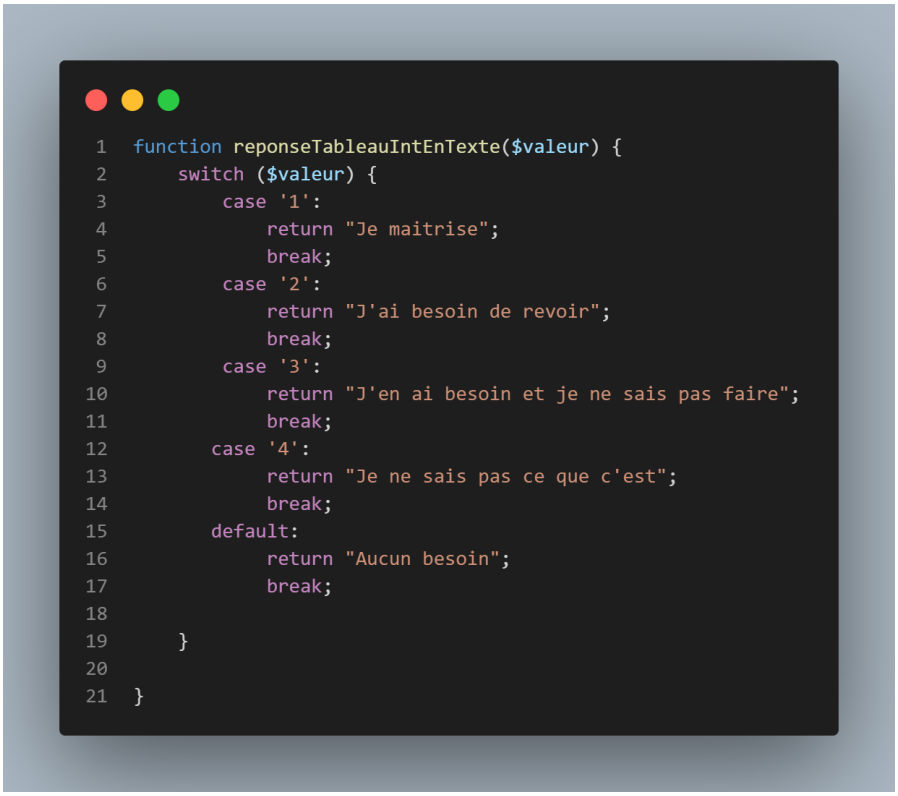
Sortie :

- Affichage HTML d'un bloc pour chaque niveau existant :
- Le score sous la forme « X / Y ».
- Une barre de progression remplie proportionnellement au pourcentage de réussite, avec une couleur différente selon le niveau.

pdf_template.php

Role : Créer le template du pdf envoyé par mail.

responseTableauIntEnTerxte($valeur)



```php
function reponseTableauIntEnTexte($valeur) {
    switch ($valeur) {
        case '1':
            return "Je maitrise";
            break;
        case '2':
            return "J'ai besoin de revoir";
            break;
        case '3':
            return "J'en ai besoin et je ne sais pas faire";
            break;
       case '4':
            return "Je ne sais pas ce que c'est";
            break;
       default:
            return "Aucun besoin";
            break;

    }

}
```

Fonction reponseTableauIntEnTexte

Rôle : Cette fonction permet de convertir une valeur numérique issue d'une réponse de type *tableau* en une réponse textuelle compréhensible par l'utilisateur. Elle facilite l'affichage des résultats dans le PDF généré.

Entrée : $valeur *(int)* : valeur numérique correspondant au choix de l'utilisateur dans un tableau (1 à 4).

Sortie *(string)* : texte correspondant au niveau de maîtrise :

1 → *Je maîtrise*

2 → *J'ai besoin de revoir*

3 → *J'en ai besoin et je ne sais pas faire*

4 → *Je ne sais pas ce que c'est*

Valeur par défaut → *Aucun besoin*

getPdfHtmlContent()

Rôle :

Cette fonction génère le contenu HTML complet du PDF de résultats d'un questionnaire. Elle récupère les informations du candidat, les questions, les réponses fournies, calcule l'affichage adapté selon le type de question et retourne un code HTML prêt à être converti en PDF.

Entrées :

$pdo *(PDO)* : connexion à la base de données permettant l'exécution des requêtes SQL.

$id_questionnaire *(int)* : identifiant unique du questionnaire rempli par l'utilisateur.

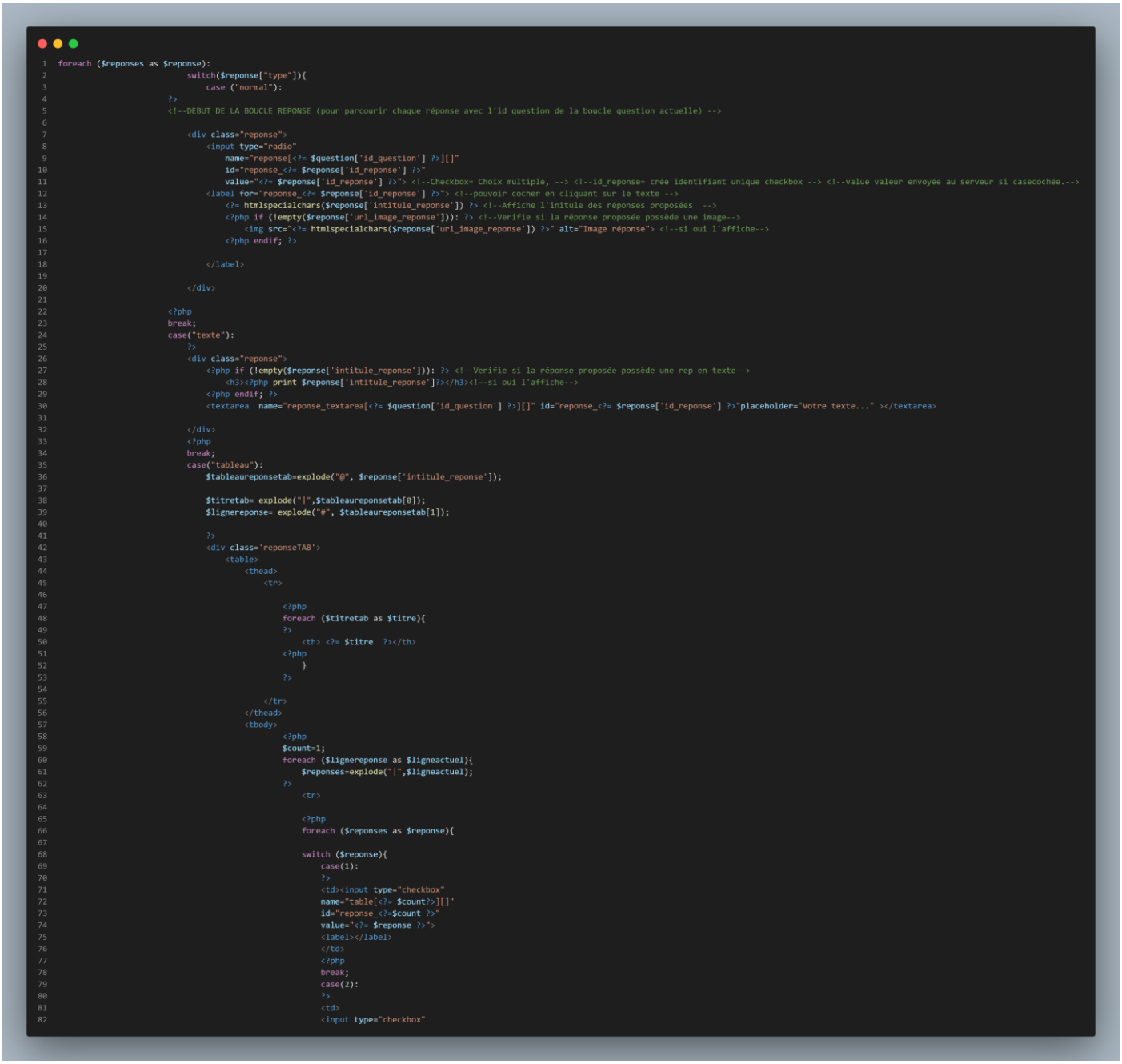
Sortie :

(string) : contenu HTML complet du document de résultats, généré à l'aide d'un tampon de sortie (ob_start()), destiné à être utilisé par une librairie de génération de PDF (ex : DomPDF).

Questionnaire.php

Role : Afficher le questionnaire.

Réponses vers tableau ou texte



```php
foreach ($reponses as $reponse):
                        switch($reponse["type"]){
                            case ("normal"):
                    ?>
                    <!--DEBUT DE LA BOUCLE REPONSE (pour parcourir chaque réponse avec l'id question de la boucle question actuelle) -->

                        <div class="reponse">
                            <input type="radio"
                                name="reponse[<?= $question['id_question'] ?>][]"
                                id="reponse_<?= $reponse['id_reponse'] ?>"
                                value="<?= $reponse['id_reponse'] ?>"> <!--Checkbox= Choix multiple, --> <!--id_reponse= crée identifiant unique checkbox --> <!--value valeur envoyée au serveur si casecochée.-->
                            <label for="reponse_<?= $reponse['id_reponse'] ?>"> <!--pouvoir cocher en cliquant sur le texte -->
                                <?= htmlspecialchars($reponse['intitule_reponse']) ?> <!--Affiche l'initule des réponses proposées  -->
                                <?php if (!empty($reponse['url_image_reponse'])): ?> <!--Verifie si la réponse proposée possède une image-->
                                    <img src="<?= htmlspecialchars($reponse['url_image_reponse']) ?>" alt="Image réponse"> <!--si oui l'affiche-->
                                <?php endif; ?>

                            </label>

                        </div>

                    <?php
                    break;
                    case("texte"):
                        ?>
                        <div class="reponse">
                            <?php if (!empty($reponse['intitule_reponse'])): ?> <!--Verifie si la réponse proposée possède une rep en texte-->
                                <h3><?php print $reponse['intitule_reponse']?></h3><!--si oui l'affiche-->
                            <?php endif; ?>
                            <textarea  name="reponse_textarea[<?= $question['id_question'] ?>][]" id="reponse_<?= $reponse['id_reponse'] ?>"placeholder="Votre texte..." ></textarea>

                        </div>
                        <?php
                        break;
                        case("tableau"):
                            $tableaureponsetab=explode("@", $reponse['intitule_reponse']);

                            $titretab= explode("|",$tableaureponsetab[0]);
                            $lignereponse= explode("#", $tableaureponsetab[1]);

                            ?>
                            <div class='reponseTAB'>
                                <table>
                                    <thead>
                                        <tr>

                                            <?php
                                            foreach ($titretab as $titre){
                                            ?>
                                                <th> <?= $titre  ?></th>
                                            <?php
                                                }
                                            ?>

                                        </tr>
                                    </thead>
                                    <tbody>
                                            <?php
                                            $count=1;
                                            foreach ($lignereponse as $ligneactuel){
                                                $reponses=explode("|",$ligneactuel);
                                            ?>
                                                <tr>

                                                <?php
                                                foreach ($reponses as $reponse){

                                                switch ($reponse){
                                                    case(1):
                                                    ?>
                                                    <td><input type="checkbox"
                                                    name="table[<?= $count?>][]"
                                                    id="reponse_<?=$count ?>"
                                                    value="<?= $reponse ?>">
                                                    <label></label>
                                                    </td>
                                                    <?php
                                                    break;
                                                    case(2):
                                                    ?>
                                                    <td>
                                                    <input type="checkbox"
```

Code pour afficher les questions de type normal texte ou tableau

Pour chaque question :

- Récupérer toutes les réponses associées.

Pour chaque réponse, déterminer le type :

- normal → afficher des boutons radio
- texte → afficher une zone de texte (textarea)
- tableau → générer un tableau HTML interactif avec checkbox et texte :

L'algorithme prend une chaîne formatée contenant les titres et les lignes d'un tableau. Il découpe d'abord la chaîne avec @ pour séparer titres et lignes, puis utilise | pour obtenir chaque **titre et cellule** et # pour séparer les lignes. Ensuite, il génère le tableau HTML : les titres deviennent <rh>, chaque cellule devient <td>, et les valeurs numériques prédéfinies sont transformées en checkbox, le reste reste du texte. Ainsi, une chaîne plate devient un tableau interactif complet.

Resultat.php

Rôle : Afficher les résultats du questionnaire à l'utilisateur, avec score, barre de progression et message personnalisé.

SaveTraiter.php

default_value()

```php

        function default_value($idLogiciel,$bdd)
        {
            $defaultValue= array();
            $stmt = $bdd->prepare("SELECT id_question,id_reponse FROM reponses_proposees
             Join questions using(id_question) WHERE id_logiciel = ? AND intitule_reponse =
              'Je ne sais pas' AND  `type`= 'normal' ORDER BY id_question"); //contient la question,
              //  id logiciel a ? pour eviter les injections sql
            $stmt->execute([$idLogiciel]);
            $defaultValue = $stmt->fetchAll(PDO::FETCH_KEY_PAIR);
            return($defaultValue);
        }
```

Fonction default_value

- **Rôle** : La valeur par default d'une question sans réponse est « je sais pas », la fonction cherche à quel numéro elle est associée.
- **Entrée** : $idLogiciel, $bdd.
- **Sortie** : Tableau associatif $defaultValue[$id_question] = $id_reponse.

SendEmail.php

LoadEnv()

```php
function loadEnv($path) {
    $lines = file($path, FILE_IGNORE_NEW_LINES | FILE_SKIP_EMPTY_LINES);
    foreach ($lines as $line) {
        if (strpos(trim($line), '#') === 0) continue; // Ignore les commentaires
        list($name, $value) = explode('=', $line, 2);
        $_ENV[$name] = trim($value);
    }
}

// Charger le .env
loadEnv(__DIR__ . '/.env');
```

FonctionLoadEnv

Rôle : Lire les variables d'environnement depuis un fichier .env et les stocker dans $_ENV.

Entrée : Chemin du fichier .env.

Sortie : Variables accessibles via $_ENV['NOM_VARIABLE'].

sendQuestionnaireResults()

```php
function sendQuestionnaireResults($pdo, $id_questionnaire, $user_data) {
    $mail = new PHPMailer(true);

    try {
        // SMTP Configuration
        $mail->isSMTP();
        $mail->Host = 'smtp.gmail.com';
        $mail->SMTPAuth = true;
        $mail->Username = $_ENV['SMTP_USER'];
        $mail->Password = $_ENV['SMTP_PASS'];
        $mail->SMTPSecure = 'tls';
        $mail->Port = 587;

        $mail->setFrom('noreply@dolfi-formation.com', 'DOLFI Formation');
        $mail->addAddress('ori.cazou@gmail.com');

        $mail->isHTML(true);
        $mail->Subject = "Resultat questionnaire - {$user_data['nom']}";
        $mail->Body = generateEmailBody($user_data);

        // Générer le fichier des réponses
        $attachmentContent = generateResponsePdf($pdo, $id_questionnaire);
        $attachmentName = "reponses_candidat_{$user_data['nom']}_{$user_data['prenom']}.pdf";
        // Ajouter la pièce jointe
        $mail->addStringAttachment($attachmentContent, $attachmentName);

        $mail->send();
        return true;
    } catch (Exception $e) {
        error_log("PHPMailer Error: {$mail->ErrorInfo}");
        return false;
    }
}
```

FonctionSendQuestionnaireResults

Rôle : Envoyer les résultats du questionnaire par email, avec pièce jointe PDF.

Entrée :

$pdo → connexion PDO à la base.

$id_questionnaire → identifiant du questionnaire du candidat.

$user_data → tableau contenant les informations utilisateur et score.

Sortie :

true si l'email est envoyé avec succès.

false en cas d'erreur (loguée).

generateEmailBody()

```php
function generateEmailBody($user_data) {
    if (!isset($user_data['pourcentage'])) {
        error_log("pourcentage non trouvé " . print_r($user_data, true));
        $user_data['pourcentage'] = 0; // Default value
    }

    extract($user_data);
    ob_start();
    include __DIR__ . '/email_template.php';
    $content = ob_get_clean();

    if (empty($content)) {
        error_log("Template vide");
    }

    return $content;
}
```

Fonction generateEmailBody

Rôle : Générer le contenu HTML de l'email à partir d'un template.

Entrée : $user_data → informations du candidat, score, etc.

Sortie : Chaîne HTML complète pour le corps de l'email.

generateResponsePdf()

```php
function generateResponsePdf($pdo, $id_questionnaire) {
    require_once __DIR__ . '/pdf_template.php';
    $html = getPdfHtmlContent($pdo, $id_questionnaire);

    // 1. Créer et configurer UNE SEULE instance d'Options
    $options = new Options();
    $options->set('isRemoteEnabled', true);
    $options->set('defaultFont', 'Helvetica');

    // 2. Créer Dompdf AVEC ces options
    $dompdf = new Dompdf($options);

    // 3. Charger et rendre le HTML
    $dompdf->loadHtml($html);
    $dompdf->setPaper('A4', 'portrait');
    $dompdf->render();

    // 4. Retourner le PDF généré
    return $dompdf->output();
}
```

Fonction generateReponsePdf

Rôle : Générer un PDF des réponses du questionnaire pour le candidat.

Entrée :

$pdo → connexion à la BDD.

$id_questionnaire → identifiant du questionnaire.

Sortie : Contenu binaire du PDF (string) prêt à être attaché à un email.

5.Tests et validation

5.1. Tests de compatibilité.

Navigateurs testés :

- Google Chrome
- Mozilla Firefox
- Microsoft Edge
- Brave

Résultats :

- L'ensemble des pages s'affiche correctement et les mises en page restent cohérentes.
- Les animations et les transitions fonctionnent sur tous les navigateurs testés.
- Les liens de navigation sont opérationnels
- La responsive se comporte correctement sur petits écrans

5.2 Tests de responsive design.

Ecrans testés :

- iPhone SE
- iPhone XR
- iPhone 12 Pro
- iPhone 14 Pro Max
- Pixel 7
- Samsung Galaxy S8+
- Samsung Galaxy S20 Ultra
- iPad Mini
- iPad Air
- iPad Pro
- Surface Pro 7
- Surface Duo
- Galaxy Z Fold 5
- Asus Zenbook Fold
- Samsung Galaxy A51/71
- Nest Hub
- Nest Hub Max
- MSI 23.6" LED - MAG 242C

Résultats :

- L'ensemble des pages s'affiche correctement et les mises en page restent cohérentes.
- Les liens de navigation sont opérationnels

6.Documentation utilisateur

- Chaque question apparaît une à la fois.
- Ne pas répondre à la question revient à choisir la réponse « je ne sais pas ».

Types de questions :

Choix unique : sélectionner une réponse parmi plusieurs.

- Texte libre : écrire votre réponse dans un champ texte.
- Tableau / grille : cocher les cases appropriées selon la question.

Utilisez les boutons :

- Suivant → passer à la question suivante.
- Précédent → revenir à la question précédente.
- Envoyer → valider et soumettre le questionnaire (apparaît à la dernière question).

Avant de soumettre le questionnaire, renseignez :

- Nom (*obligatoire*)
- Prénom (*obligatoire*)
- Email (*obligatoire*)
- Téléphone (optionnel)
- Établissement (optionnel)

Après avoir cliqué sur Envoyer, vos réponses sont :

1. Enregistrées dans la base de données.
2. Évaluées pour calculer le score.
3. Un email est envoyé à l'administrateur et vous même avec :
    - Votre nom et prénom.
    - Votre score et pourcentage de réussite.
    - Un PDF récapitulatif de vos réponses.

7.Conclusion

7.1 Bilan.

Ce projet de refonte d'une application de questionnaires en ligne a permis d'atteindre les objectifs fixés au début du stage. Le site, initialement statique et difficile à maintenir, a été transformé en une application dynamique reposant sur une base de données. La mise en place d'un modèle de données structuré a facilité la gestion des questionnaires, des questions, des réponses et des résultats. La modernisation de l'interface, tout en conservant le design existant, a amélioré l'ergonomie et rendu l'application responsive. Les fonctionnalités essentielles, telles que le calcul automatique des scores et l'envoi des résultats par e-mail, ont été conservées et fiabilisées.
Ce projet m'a permis de consolider mes compétences techniques en PHP et MySQL, ainsi que de mieux comprendre la logique de fonctionnement d'une application web complète, de la conception de la base de données à la mise en production des fonctionnalités.

7.2 Améliorations possibles.

Plusieurs améliorations pourraient être envisagées afin de faire évoluer l'application. Il serait possible de développer une interface d'administration permettant de créer, modifier et supprimer les questionnaires directement depuis le site, sans intervention dans le code.
L'ajout d'un système d'authentification permettrait de sécuriser l'accès aux résultats et de proposer un suivi personnalisé des participants. Des améliorations en matière de sécurité, telles que la protection renforcée contre les injections SQL ou la gestion avancée des sessions, pourraient également être mises en place.
Enfin, l'optimisation des performances, l'ajout de statistiques détaillées sur les résultats ou encore l'amélioration du design graphique pourraient contribuer à rendre l'application plus complète et plus professionnelle.