


DESCRIPTION D'UNE MISSION   BTS SIO			
Prénom – Nom	Quentin Coqueran	N° mission	
Option	SLAM		
Situation		Formation	
Lieu de réalisation	Campus Montsouris  2 Rue Lacaze – 75014 PARIS		
Période de réalisation	Novembre 2020	Novembre 2020	
Modalité de réalisation	VÉCUE		

<b>Intitulé de la mission</b>	Titre de la mission
	Création d'un jeu « juste prix »
<b>Description du contexte de la mission</b>	Description en 2 à 3 lignes maxi
	Créer une page où l'utilisateur doit trouver un chiffre, généré aléatoirement et compter son nombre de tentatives.

<b>Ressources et Outils utilisés</b>	Liste des ressources disponibles et outils utilisés (Documentations, Matériels et Logiciels)
	<ul style="list-style-type: none"> <li>- Visual Studio Code</li> <li>- WampServer</li> </ul>
<b>Résultat attendu</b>	Résultat attendu avec la réalisation de cette mission Nombre généré aléatoirement par PHP et afficher en fonction de la réponse de l'utilisateur, « le nombre est trop petit », « le nombre est trop grand » ou « vous avez gagné » et compter son nombre de tentatives.
<b>Contraintes</b>	Contraintes : techniques   budgétaires   temps   O.S. ou outils imposés...
	<ul style="list-style-type: none"> <li>• Générer un chiffre aléatoirement qui ne change pas tant que la partie n'est pas terminée.</li> <li>• Compter le nombre de tentative de l'utilisateur</li> </ul>

<b>Compétences associées</b> (Voir tableau)	<div> <div>Liste des intitulés du tableau de compétences (avec les références)</div> <div>A2.1.1 Accompagnement des utilisateurs dans la prise en main d'un service</div> </div>
<div> <div> Description simplifiée des différentes étapes de réalisation de la mission  en mettant en évidence la démarche suivie, les méthodes et les techniques utilisées </div> <div> <p><b><u>Chapitre 1 : Développement de la partie HTML du projet</u></b></p> <p><b>Etape :</b></p> <p>1.1/ Dans ce projet il y’a très peu de HTML, tout d’abord nous créons le fichier index.php qui sera la seule page de ce projet. Ensuite nous créons le formulaire afin que l’utilisateur puisse saisir un chiffre.</p> <pre>&lt;input type="number" name="nb" placeholder="Tentez votre chance :" required&gt;&lt;br&gt;&lt;br&gt; &lt;input type="submit" value="Envoyer" name="ok"&gt;</pre> <p><b>Résultat :</b></p> <div> <div>Tentez votre chance : <input type="text"/></div> <div>Envoyer</div> </div> <p><b><u>Chapitre 2 : Développement de la partie PHP du projet</u></b></p> <p><b>Etape :</b></p> <p>2.1/ Dans un premier temps nous allons nous occuper du système du comptage de tentative. Pour savoir si l’utilisateur a déjà fait des tentatives nous allons créer un champ caché, qui aura pour valeur un chiffre, pour définir ce chiffre on fait une condition simple, <i>si le champ « compteur » n’existe pas alors crée le, sinon rajoute 1 a ce compteur</i>, du coup lors de notre première visite étant donné que le champ « compteur » n’existe pas il aura pour valeur 0, et quand nous soumettrons notre réponse la condition se relancera et le champ « compteur » cette fois ci existera et s’auto incrémentera.</p> <pre>if(!isset(\$_POST["compteur"])) {     \$compteur = 0;     echo "&lt;input type='hidden' name='compteur' value='". \$compteur. "'&gt;"; } else {     echo "&lt;input type='hidden' name='compteur' value='". ++\$_POST["compteur"]. "'&gt;"; }</pre> </div> </div>	

2.2/ Dans un second temps nous voulons générer un chiffre aléatoire. Générer un chiffre aléatoirement est relativement simple, cependant étant donné qu'à chaque tentative tout le code PHP est relancé, il faut faire le même stratagème que le précédent pour éviter que le chiffre change.

```
if(!isset($_POST["aleat"]))
{
    $aleat = rand(0,100);
    echo "<input type='hidden' name='aleat' value='".$aleat."'>";
}
else
{
    echo "<input type='hidden' name='aleat' value='".$_POST["aleat"]."'>";
}
```

Ici on utilise la fonction rand() pour générer un nombre aléatoire de 0 à 100, qui s'exécutera uniquement si le champ « aleat » n'existe pas, et tout comme dans le cas précédent le champ « aleat » n'existera pas seulement le premier tour, ce qui permettra de garder le chiffre aléatoire toute la partie.

2.3/ Et enfin pour afficher à l'utilisateur si le chiffre indiqué est plus petit, plus grand, ou exact on utilise encore une fois quelque condition

```
<?php
if(isset($_POST["ok"]))
{
    $nb = $_POST["nb"];
    $compteur = $_POST["compteur"];
    $aleat = $_POST["aleat"];
    if($nb < $aleat) echo "<h3>C'est plus grand que $nb</h3>";
    if($nb > $aleat) echo "<h3>C'est plus petit que $nb</h3>";
    echo "Vous avez utilisé $compteur tentative(s)";
    if($nb == $aleat) {
        echo "<h3>Vous avez gagné...</h3>";
        header('Refresh: 5;URL=index.php');
    }
    if($compteur==6 && $nb!=$aleat)
    {
        echo "<h3>Vous avez perdu, il fallait trouvé $aleat...</h3>";
        header('Refresh: 5;URL=index.php');
    }
}
?>
```

Tout d'abord on vérifie si le chiffre est bien envoyé, ensuite on écrit *si le nombre choisit par l'utilisateur est plus petit que le nombre a trouvé, affiché « c'est plus grand »*. *Si le nombre choisit par l'utilisateur est plus grand que le nombre a trouvé, afficher « c'est plus petit »*. Ensuite on affiche le nombre de tentative effectué, puis si le chiffre saisi par l'utilisateur est égal au chiffre a trouvé nous pouvons lui dire qu'il a gagné. Nous avons aussi rajouté une condition qui fait perdre l'utilisateur s'il fait plus de 6 tentatives. Le « header refresh » rafraichie la page au bout de 5 secondes.

## Conclusion

Cette mission m'a été beaucoup utile, afin de bien comprendre le fonctionnement des données du formulaire, et l'impact qu'a un rafraîchissement sur une page. Cela m'a aussi fait plus comprendre l'utilisation des champs « hidden ».