

DOSSIER DE PROJET PRO TITRE PROFESSIONNEL DE DÉVELOPPEUR WEB & WEB MOBILE

—

Alex Lovati



Table des matières

Compétences du référentiel couvertes par le projet	3
Résumé du projet	3
Cahier des charges	4
Spécification du projet	4
Charte graphique	5
Maquettage	5-6
Trello	7
Base de données	8
GitHub / GitKraken	9
Spécifications fonctionnelles	10
Arborescence	10
Description des fonctionnalités utilisateurs	10
1 - Authentification	10
2 - Formulaire de devis	10
3 - Type de prestations	11
4 - Espace client	11
Back Office	11
1 - Gestion des devis	11
2 - Gestion de planning	11
3 - Gestion des utilisateurs	11
4 - Gestion des images	11
Spécifications techniques	12
Technologies utilisées	12
Architecture	12
Réalisations comportant des extraits de code	13
Extrait 1	13
Extrait 2	14
Sécurité	15
Vulnérabilités	15
HTTPS	15
RGPD	15
Injection SQL	16
Failles XML	17
Htmlspecialchars	17
Chiffage (hash)	18
Protection des pages admin	19
SEO	20
Recherches effectuées en anglais	20-22
Conclusion	22

Compétences du référentiel couvertes par le projet

Activité type 1 :

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

- 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

Activité type 2 :

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

- 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

Résumé

L'entreprise pour laquelle j'ai réalisé ce site web est Lovati Paysages, une entreprise familiale de paysagiste géré par M.Lovati depuis 2006. Il souhaitait un site internet étant donné qu'il n'en avait pas pour l'aider à toucher plus de clientèle et avoir plus de visibilité. Mais aussi pour l'aider et faciliter son travail avec la possibilité de faire une demande de devis directement sur le site et une partie admin qui permet de les gérer plus facilement. Il est aussi possible pour lui de gérer son planning de rendez-vous. Ce projet a été réalisé à 2 dans le cadre de ma formation à La Plateforme. On retrouve aussi sur ce site les prestations que l'entreprise propose, il est possible pour l'utilisateur de se connecter afin de suivre ses devis.

Cahier des charges

Spécification du projet

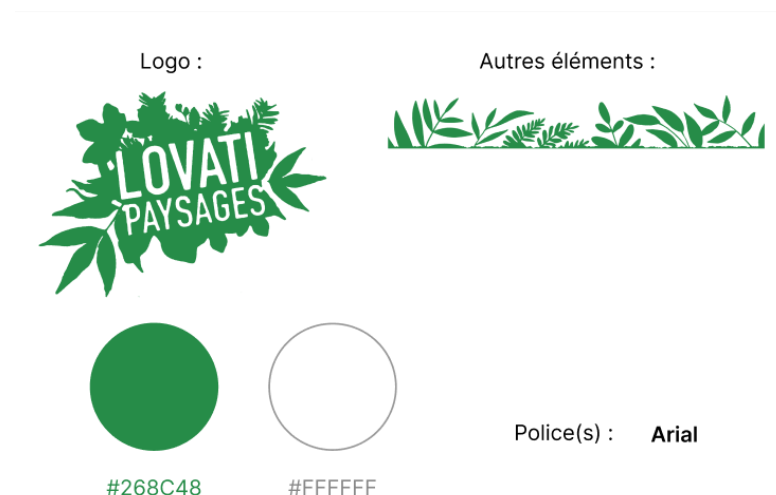
Le client pour qui nous avons travaillé sur ce projet n'avait pas de site au départ et n'avait pas d'idée sur le design de celui-ci, seulement une idée des fonctionnalités qu'il voulait avoir sur son site.

Nous avons dû réfléchir à comment le rendre agréable et attirant tout en étant fonctionnel et responsive pour le confort de l'utilisateur. Tout en suivant les demandes du client.

Nous avons commencé par la page d'accueil en passant par les pages dédiées à l'utilisateur jusqu'à celles réservées à l'administrateur.

Charte graphique

Nous avons réalisé la charte graphique sur Figma, elle nous a permis de visualiser l'ambiance du site, ainsi que d'y regrouper des informations importantes et pratiques telles que la police ou encore les couleurs et leur codes.



Maquettage

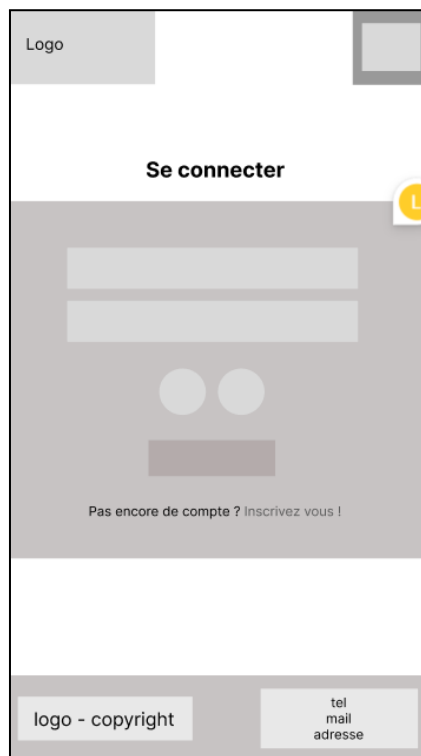
La création d'une maquette nous a permis de visualiser ce à quoi ressemblerait le site, ainsi que de réfléchir au positionnement de chaque élément.

Elle nous a permis également de montrer un premier résultat au client, afin de voir ensemble si le résultat lui convenait ou s' il y avait des modifications à apporter.


Nous avons décidé de faire la maquette en mobile first, qui est le fait de développer le site sur format mobile en premier, puis de l'adapter aux plus grands formats. Nous trouvions ça plus simple d'élargir les éléments du site vers un plus grand format que l'inverse.

Nous avons présenté chaque étape sur Figma, qui permet de montrer le site sous forme de prototype, nous avons également utilisé des composants pour présenter une maquette dynamique avec des éléments cliquables tel que le menu hamburger qui se déroule quand on clique dessus.


Nous avons commencé par la création de la maquette basse fidélité pour ce concentrer sur la position des éléments qui seront sur le site, sans les détails (couleurs, images, ...)





Ensuite après avoir été montrée au client et qu'il ai validé, nous sommes passés à la maquette haute fidélité qui permet de visualiser ce à quoi ressemblerait le site final.




Se connecter





Pas encore de compte ? [Inscrivez vous !](#)



Contact :
06.11.35.34.35
lovatipaysages@gmail.com
33, Boulevard Gavoty
Marseille, France

[Mentions Légales](#)
[Politique de confidentialité](#)
[Données personnelles](#)

Lovati Paysages © 2006 - 2022 - Tous droit réservés

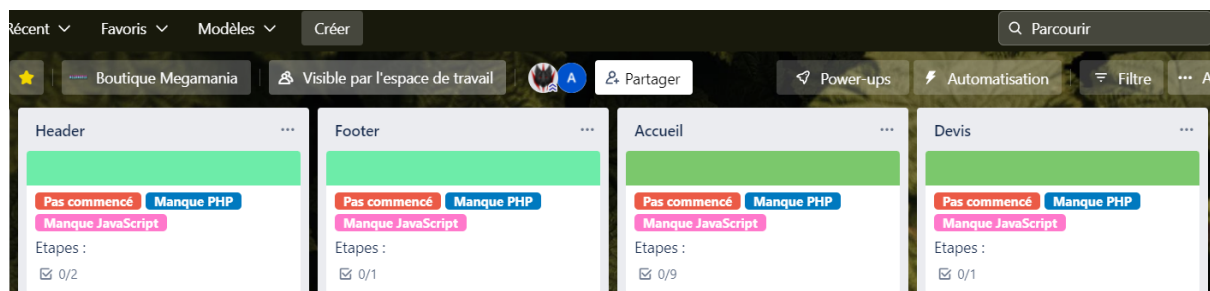
Trello

Trello est un site qui nous a permis d'améliorer l'organisation du travail en équipe et des tâches à réaliser.

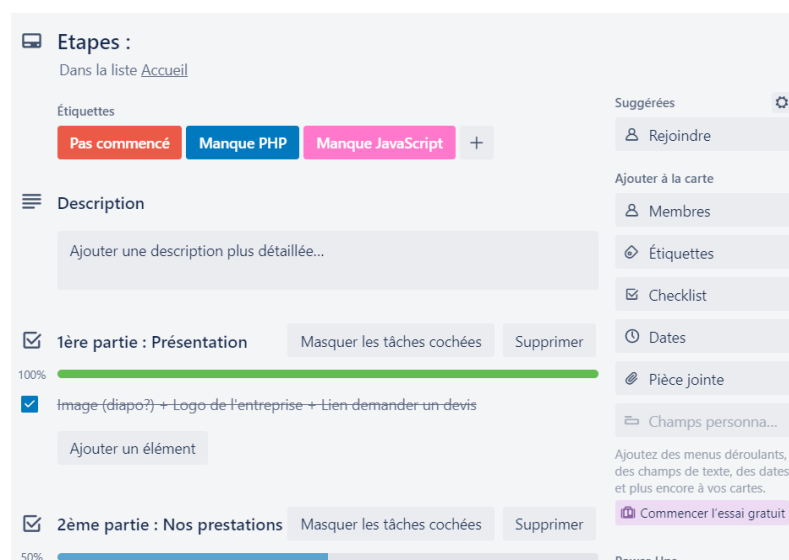
Nous avons pu nous répartir les tâches à faire et noter leur évolution grâce à un système d'étiquettes et de checklist.



Trello fonctionne sous forme de liste comportant des cartes, pour ce projet nous avons décidé qu'une carte sera égale à une page du site.



Exemple de carte avec les différentes étapes :

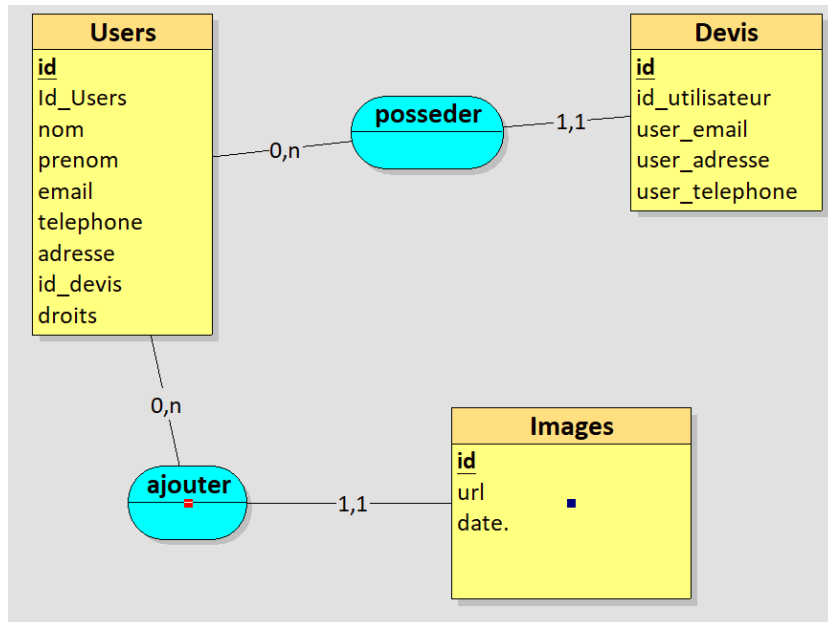


Trello nous a permis d'avoir une première vision du nombre de pages que le site comportera.

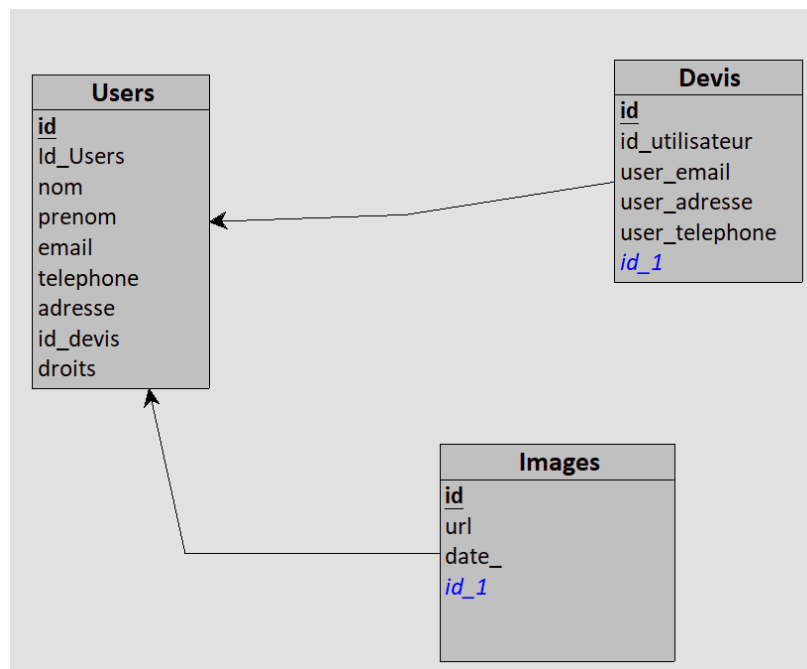
Base de données

Nous avons utiliser le Modèle Conceptuel de Données (MCD) ainsi que le Modèle Logique de Données (MLD)

MCD

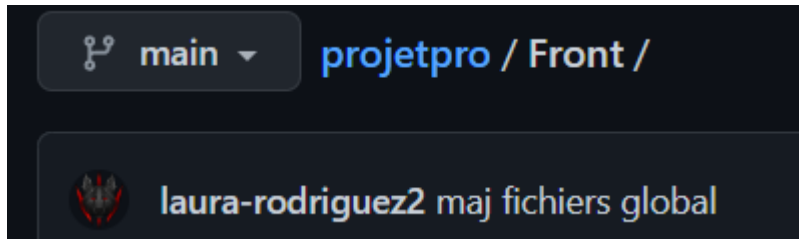


MLD



GitHub et GitKraken

GitHub nous a permis de sauvegarder notre travail et de récupérer l'avancée de chaque personne.



GitKraken nous a servi comme interface graphique d'utilisateur (GUI) en lien avec GitHub, cela nous a facilité le travail.

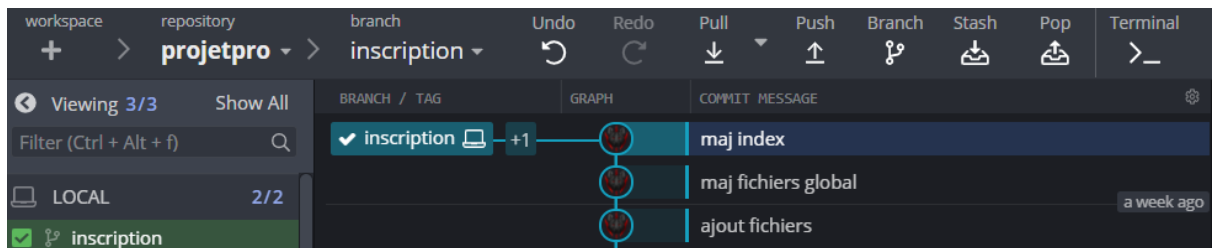
Git Kraken nous permet de de sauvegarder notre travail au fur et a mesure grâce aux 'commits' qui sont des sortes de sauvegardes, nous en faisons régulièrement à chaque avancée afin de garder notre travail à jour et de savoir où nous en sommes.

Ces commits nous les faisons sur nos branches respectives afin de ne pas causer de conflits, ce qui pourrait survenir si nous travaillons sur le 'master' ou sur le même fichier.

Les branches étaient nommées selon la page ou la fonctionnalité que nous étions en train de travailler.

Dès que nous avons fini, nous utilisons la fonctionnalité 'merge' puis 'push' afin d'emmener le travail effectué sur notre branche directement sur le 'master'.

Pour récupérer le travail d'une autre personne sur le 'master' nous utilisons 'pull'.



Spécifications fonctionnelles

Arborescence

Voici l'arborescence du site :

- Page d'Accueil
- Page de Connexion
- Page d'Inscription
- Page de Profil
- Page de Devis

Concernant la partie Back office :

- Page d'Accueil de l'administrateur
- Page Gestion du planning
- Page Gestion des utilisateurs
- Page Gestion des devis

Description des fonctionnalités utilisateurs

1) Authentification

Pour que l'utilisateur puisse accéder à toutes les fonctionnalités, il lui est possible de s'inscrire sur le site, grâce à cela, il pourra avoir accès à son espace client qui lui permet de modifier ses informations personnelles ou encore de suivre ses devis.

Pour faciliter l'authentification, nous avons décidé d'y instaurer une authentification via Facebook Connect.

Pour l'authentification classique, nous devons éviter à l'utilisateur de devoir rentrer trop de champ, il ne rentre seulement que le nécessaire, puis le reste des informations (par exemple l'adresse ou le numéro de téléphone) sera demandé ultérieurement quand cela sera nécessaire.

2) Devis

Un utilisateur peut remplir le formulaire de devis directement sur le site, il se trouve sur une page dédiée et est accessible pour les utilisateurs connectés ou non.

Si un utilisateur n'est pas connecté il devra remplir tous les champs nécessaires et n'aura pas accès à un suivi de son devis. Une fois sa demande de devis envoyée, l'utilisateur la suivra par mail.

Cependant si il décide de s'inscrire, il aura accès à ses devis envoyés dans son espace client, il pourra aussi voir son avancée.

3) Prestations

L'utilisateur pourra notamment voir sur la page d'accueil les différentes prestations que l'entreprise propose, sous forme de bandeaux pour chaque prestation, il sera possible de cliquer dessus ce qui ouvrira une popup grâce à l'utilisation de JavaScript, l'utilisateur y trouvera toutes les informations a savoir sur la prestation choisie.

4) Espace client

L'utilisateur connecté aura accès à un espace client, ou il lui sera possible de modifier ses informations ou encore de suivre ses devis ainsi que leurs statuts.

Concernant le back office :

1) Gestion des devis

L'administrateur pourra gérer les devis des utilisateurs qu'il recevra directement sur une page dédiée. Ils seront triés par statut selon leur avancé, quand l'admin cliquera sur un des devis, il aura la possibilité de répondre au devis.

2) Gestion de planning

L'administrateur pourra gérer son planning, avec la possibilité de rentrer les rendez-vous manuellement s'il le souhaite. Il aura aussi la possibilité de consulter et gérer les rendez-vous inscrits dans le planning, voire aussi de les annuler en cas d'imprévu.

3) Gestion des utilisateurs

L'administrateur aura aussi accès à la liste de tous les utilisateurs, ainsi qu'une barre de recherche permettant de recherchant un utilisateur en particulier. Il pourra accéder aux différentes informations tel que leur nom, prénom, mail, téléphone, adresse... mais pas leur mot de passe. Il pourra également changer leur rang. Ce qui pourra être utile dans le cas ou plusieurs personnes géraient le site.

4 - Gestion des réalisations

Cette page permet simplement à l'admin de gérer les photos de la section 'Nos dernières réalisations' de la page d'accueil.

Spécifications techniques

Technologies utilisées

Les technologies utilisées pour le front-end sont :

- ❖ **HTML/CSS** pour la base du site
- ❖ **JavaScript** pour la dynamisation des pages
- ❖ **Media Queries** pour le responsive

Les technologies utilisées pour le back-end sont :

- ❖ **PHP** en POO (Programmation Orientée Objet)
 - ❖ **SQL** en PDO (PHP Data Objects)
 - ❖ **MySQL** en tant que SGBDR (système de gestion de bases de données relationnelles)
 - ❖ **Github** pour l'organisation et la sauvegarde du travail
 - ❖ **GitKraken** comme interface graphique pour GitHub
 - ❖ **Xampp** en tant que serveur web local
 - ❖ **PhpMyAdmin** en tant qu'interface utilisateur (GUI) de MySql
-
- ❖ **Visual Studio Code** comme éditeur de code
 - ❖ **Figma** pour le maquettage
 - ❖ **Looping** pour la création du MCD/MLD
 - ❖ **Rewrite** pour renommer les url

Architecture

Nous avons décidé de ne pas utiliser le modèle architectural MVC (Model View Controller) car ne le maîtrisant pas assez nous avons décidé de faire sans, de plus nous n'en avons pas ressenti le besoin pour ce site.

Nous avons donc procédé comme suit pour l'architecture :

Un dossier Front contenu un dossier :

- ❖ HTML
- ❖ CSS
- ❖ JAVASCRIPT
- ❖ MEDIAS

Un dossier Back contenant les différents fichiers .php de Classes.

Réalisations comportant des extraits de code

Extrait 1 : Page de connexion

Voici un extrait de la page de connexion :

```
<body>
  <header>
    <?php
      require_once('header_footer/header.php');
    ?>
  </header>

  <main>
    <h1>Connectez-vous :</h1><br>

    <form id="form_connec" method="POST" action="">
      <div class="connec">
        <p>Entrez votre login :</p>
        <input type="text" class="box-input" name="loginconnect">
      </div>

      <div class="connec">
        <p>Entrez votre mot de passe :</p>
        <input type="password" class="text" name="passwordconnect">
      </div>

      <input type="submit" id="bouton_co" name="formconnexion" value="Se connecter !">
      <p class="lr_h2">Vous n'avez pas de compte ? <a class="lien_inscription" href="inscription.php">Inscrivez-vous ici</a></p>
    </form>
  </main>

  <footer>
    <?php
      require_once('header_footer/footer.php');
    ?>
  </footer>
</body>
```

On y retrouve à l'intérieur de la balise <body> tout ce qui sera afficher à l'utilisateur, au début nous avons la balise <header> qui comporte du php qui permet d'afficher le contenu de 'header.php'.

Ça permet d'avoir une seule fois le code du header dans une page à part, que l'on appellera ensuite dans chaque page. Pour la balise <footer> plus bas c'est le même principe.

Ensuite dans la balise <main> nous avons tout le contenu de la page connexion dont le formulaire de connexion dans la balise <form>.

Les informations pourront être rentrées par les utilisateurs grâce aux <input>.

Extrait 2 : Méthode de connexion

```
public function connexion($login, $password){  
    if(isset($_POST['formconnexion']))  
    {  
        $loginconnect = htmlspecialchars($_POST['loginconnect']);  
        $passwordconnect = $_POST['passwordconnect'];  
        $bdd = $bdd = $this->getBdd();  
  
        if(!empty($loginconnect) AND !empty($passwordconnect))  
        {  
            $requeteutilisateur = $bdd->prepare("SELECT * FROM utilisateurs WHERE login  
            $requeteutilisateur->execute(array($loginconnect)); // Execute le prepare  
            $result = $requeteutilisateur->fetchAll(); // Return TOUTE la requete ( t  
            if (count($result) > 0){ // S'il trouve pas de même login, il return ma  
                $sqlPassword = $result[0]['password']; // Récupere le resultat du  
                if(password_verify($passwordconnect, $sqlPassword)) // Si password  
                {  
                    $_SESSION['id'] = $result[0]['id'];  
                    $_SESSION['login'] = $result[0]['login'];  
                    $_SESSION['email'] = $result[0]['email'];  
                    $_SESSION['droits'] = $result[0]['droits'];  
                    header("Location: profil.php");  
                }  
            }  
            else  
            {  
                $erreur = "Mauvais mot de passe !";  
            }  
        }  
    }  
}
```

Voici une partie de la méthode permettant de connecter l'utilisateur.

En premier on sécurise le login récupéré dans l'input login avec `htmlspecialchars` et on le met dans une variable, on fait pareil pour le mot de passe.

Ensuite on se connecte à la base de données avec la fonction `getBdd()`.

Puis si le champ du login et mot de passe est bien rempli on va chercher ces informations dans la base de données avec la requête SQL : INSERT puis on regroupe les informations grâce à `fetchAll()`.

On vérifie avec `count($result)` si il trouve un login dans la base de données, si c'est le cas on peut continuer.

On vérifie le mot de passe si il est valide on crée les variables de `$_SESSION` qui permettent à l'utilisateur d'être connecté, il le sera sur toutes les pages grâce à la fonction `session_start()`.

Ensuite dans le cas ou le mot de passe ne serait pas correcte on affiche un message d'erreur, le code continu avec d'autres messages d'erreurs selon le problème.

Sécurité

Vulnérabilités

Nous avons effectué une veille sur la sécurité afin de sécuriser un maximum le site internet, étant plus qu'un site vitrine, le nombre de failles est encore moins à prendre à la légère.

En effet la sécurité des données utilisateur ainsi que toutes les autres informations stockées dans la base de données doivent être un maximum sécurisées.

Nous allons énumérer quelques types de méthodes et de failles qu'il est impératif de prendre en compte.

HTTPS

Le protocole HTTPS (Hyper Text Transfert Protocol Secure) est la version sécurisée de HTTP, il se trouve au niveau de l'url et permet d'échanger des données cryptées.

C'est un protocole de communication qui permet la liaison entre le client et le serveur Web.

RGPD

Le RGPD (Règlement Général sur la Protection des Données) encadre le traitement des données personnelles sur le territoire de l'Union européenne.

Il concerne toute entreprise traitant des données personnelles et qui est soit établie au sein de l'Union Européenne ou ayant une activité ciblant des résidents de l'Union Européenne.

L'utilisateur a le droit de consentir ou non au partage de ses données personnelles dans un certain but.

Il rentre en compte notamment dans :

- ❖ L'utilisation des cookies
- ❖ Abonnement à une Newsletter
- ❖ Accès à une politique de confidentialité

Injections SQL

L'injection SQL est un type d'attaque sur le web qui permet à un attaquant d'insérer des instructions SQL malveillantes dans un site web, pouvant potentiellement accéder à des données sensibles dans la base de données.

Ce type d'attaque est généralement réalisée au niveau des formulaires, Au niveau des 'inputs' d'un formulaire, l'attaquant rentrera des informations qui seront interprétées par la méthode SQL au lieu de rentrer son nom d'utilisateur ou encore son mot de passe.

Il existe cependant plusieurs manières pour sécuriser son site contre ce type d'attaque, on peut citer par exemple le fait de :

Préparer ses requêtes :

Cette méthode permet de rendre impossible les injections SQL grâce aux déclarations préparées.

En effet, comme on peut le voir dans l'exemple ci-dessous, le fait de spécifier les paramètres de la requête ultérieurement empêche les injections SQL.

```
$query = $bdd->prepare("SELECT * FROM utilisateurs WHERE login = ?");  
$query->execute(array($loginconnect)); // Execute le prepare
```

Vérifier le type de caractères dans les zones de textes :

Par exemple, dans un formulaire d'inscription, l'utilisateur devra entrer un mot de passe comportant un minimum de caractères, avec au minimum un chiffre, une majuscule et un caractère spécial.

De même pour l'e-mail, si l'email entré ne comporte pas d'@, le formulaire renverra une erreur.

Cela n'empêchera pas les attaques d'injections SQL mais ça limitera les attaques de base.

Faibles XML

Le langage XML (eXtensible Markup Language) peut se retrouver dans différent type de fichier (.pdf, .docx, .svg, ...), cependant la trace de données personnelles à l'intérieur de fichier contenant du XML peut être très dangereuse pour la sécurité des données.

Que ce soit un fichier de configuration contenant par exemple des mots de passe, des fichiers uploadés par d'autres utilisateurs, ou encore en copiant le code source d'un site, un utilisateur malveillant pourra se servir de ce type de failles.

htmlspecialchars

htmlspecialchars permet d'éviter le cross-site scripting, aussi appelé 'Failles XSS'.

Ces failles permettent à un utilisateur malveillant d'injecter du code dans une page d'un site et qui sera ensuite vu par les autres utilisateurs de ce même site.

Cette faille permet aux attaquants, entre autres, d'afficher des popups malveillants sur le site, pour par exemple attirer des utilisateurs sur un autre site, en général dangereux.

Pour palier à ce problème, l'utilisation de htmlspecialchars rentre en compte :

```
$login = htmlspecialchars($_POST['login']);  
$email = htmlspecialchars($_POST['email']);  
$password = htmlspecialchars($_POST['password']);
```

Il permet de faire en sorte que les caractères HTML ne soient pas interprétés.

Chiffage (hash)

La sécurisation des mots de passe est aussi très importante, grâce à des vérifications tel que le fait que l'utilisateur doit avoir un mot de passe comportant minimum :

- 1 chiffre
- 1 caractère spécial
- 1 majuscule
- 1 minuscule

Lorsque cette étape est validée, nous passons à la suite.

password_hash()

A cela, nous allons donc rajouter une sécurité supplémentaire et qui est elle aussi très importante : le hachage du mot de passe.

Cela va transformer le mot de passe de l'utilisateur en suite de caractères chiffrés, il sera rentré dans la base de données sous cette forme là.

Cela protégera ainsi la fuite des mots de passe utilisateur dans le cas où il se retrouverait sous la main de personnes malveillantes ayant eu accès à la base de données.

```
$hashage = password_hash($password, PASSWORD_BCRYPT);
```

password_verify()

Ensuite, dans le cas d'un utilisateur voulant se connecter et ayant donc son mot de passe hashé et stocké dans la base de données, nous allons utiliser "password_verify()" afin de vérifier si le mot de passe rentré et celui en base de données sont identiques (en prenant en compte le hachage de celui en base de données)

```
if(password_verify($passwordconnect, $sqlPassword))
```

Protection des pages admin

Lors de la création des différentes pages, une question se pose : comment protéger les pages réservées à l'administrateur ?

Pour cela une vérification est nécessaire, en premier lieu on vérifie que la personne possède une session, c'est-à-dire qu'elle soit connectée et qu'elle possède donc un id.

Ensuite on vérifie encore une fois au niveau des variables de session, cette fois le statut de l'utilisateur grâce à 'droits', 2 étant un admin et 1 un utilisateur classique.

Dans le cas où l'utilisateur ne serait pas connecté ou aurait un statut égale à 1, il serait redirigé vers la page profil.

```
// Vérifications nécessaire pour l'accès à la page
if (!isset($_SESSION['id']) || $_SESSION['droits'] != "2") {
    header("Location: profil.php");
    exit();
}
```

Dans le cas contraire, l'administrateur pourra avoir accès à la page.

Ce système est appliqué sur toutes les pages administrateurs, de ce fait même si un utilisateur saisit l'url de la page, il ne pourra y avoir accès sans les droits nécessaires.

SEO

Le SEO (Search Engine Optimization) définit l'ensemble des techniques permettant d'améliorer la position d'un site web sur les pages de résultats des moteurs de recherche (SERP).

Le SEO est du référencement naturel, contrairement au référencement payant (SEA) qui consiste à utiliser des annonces ciblées et acheter des espaces publicitaires pour apparaître le plus possible dans les résultats des moteurs de recherche.

Concernant le SEO, il est possible d'améliorer le référencement de son site grâce à des techniques dites 'on-site' qui visent à améliorer la qualité du contenu de son site, en passant par l'ajout de certaines balises par exemple, tel que la balise <meta>.

Au niveau de la partie off-site, elle touche plutôt à l'environnement du site, et aux liens redirigeant vers une page de son site.

Situation de travail ayant nécessité une recherche en anglais

Au début du projet j'ai voulu me renseigner pour être sûr de commencer sur de bonnes bases et éviter de faire des erreurs au niveau de la sécurité du site et de la base de données.

Je suis tombé sur ce poste sur le forum populaire Stackoverflow :

What do I need to keep a secure MySQL database? [closed]

Asked 9 years, 1 month ago Modified 6 years, 10 months ago Viewed 2k times



-3



1



As it currently stands, this question is not a good fit for our Q&A format. We expect answers to be supported by facts, references, or expertise, but this question will likely solicit debate, arguments, polling, or extended discussion. If you feel that this question can be improved and possibly reopened, [visit the help center](#) for guidance.

Closed 9 years ago.



I've looked up quite a few tutorials on keeping a secure database, but I still don't know what actions I need to take to protect my database from SQL injections, and hackers.

This is the function I've been using to clean out any user input, but I feel like this isn't all there is to it, what other things am I overlooking?

```
function CleanInput($value) {  
    stripslashes($value);  
    if(!is_numeric($value)) {  
        mysql_real_escape_string($value);  
    }  
    return $value;  
}
```

Cette personne souhaite savoir ce dont elle a besoin pour garder sa base de données MySQL sécurisée.

Elle dit dans son poste qu'elle a déjà regardé quelques tutoriels pour sécuriser sa base de données mais qu'elle ne sait toujours pas quelles actions elle doit prendre pour protéger sa base de données des injections SQL et des hackers.

Elle montre ensuite une capture d'écran et dit que c'est une fonction qu'elle utilise pour nettoyer les éléments rentrés par des utilisateurs dans des inputs.

Voici une des réponses que j'ai trouvé le plus utile :

Best practise is to use [prepared statements](#)

It's also a bad practice to use [mysql_query](#) since It's deprecated, as can be seen if you check for the documentation.

Using [mysql_real_escape_string](#) is also a bad practise since It's also deprecated, and it can't save you from [logical sql injections](#)

I recommend using [PDO](#) or [mysqli](#), both of those implementations support prepared statements.

Prepared statements will avoid hackers to insert commands into your system that can cause injections, but instead it creates a template for constants, which can not affect the database. It's also good for performance.

It's also wise to validate all input, both client side and server side.

Il répond que le meilleur moyen est d'utiliser les 'prepared statements' (donc les requêtes préparées).

Il dit que c'est une mauvaise habitude d'utiliser 'mysql_query' car c'est obsolète et que ça ne réglera pas les injections sql logiques.

Il recommande d'utiliser PDO ou mysqli car cela supporte les requêtes préparées.

Il dit que les requêtes préparées évitera que des hackers insèrent des commandes dans le système qui pourrait causer des injections, et que cela crée plutôt des template pour les constantes, ce qui n'affecte pas la base de données. Et que c'est aussi meilleur pour les performances.

Il finit par dire que c'est sage de valider tous les inputs, aussi bien du côté client que du côté serveur.

Bien que je prenais déjà en compte la majorité de ces éléments, cela m'a permis de consolider mes connaissances, je suis allé ensuite me renseigner plus en détail sur certains de ces éléments.

Conclusion

Ce projet m'a permis d'améliorer mes compétences en développement web que ce soit les langages surtout le PHP ou j'avais quelques difficultés et le JavaScript qui est un peu plus récent encore.

Cela m'a permis donc de découvrir des concepts et de m'améliorer sur certains points.

La relation client aussi à été une nouvelle expérience dans ce domaine là et j'ai beaucoup apprécié, écouter les demandes et essayer de faire au mieux pour y répondre, faire un site internet responsive et agréable, en m'intéressant à l'expérience utilisateur ou encore au SEO à été une expérience très intéressante.