

DOSSIER TECHNIQUE DE PROJET INVIVIDUEL

26 MAI

Télélocation médicale

Créé par : Mattéo ELEOUET

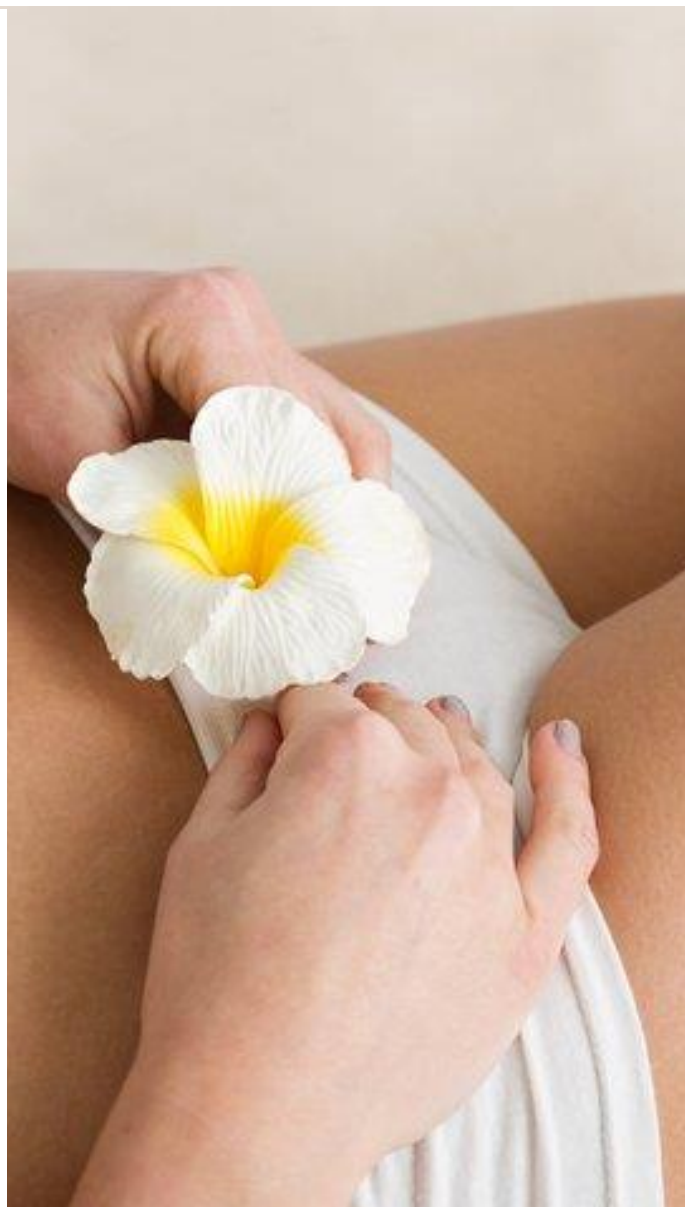


Table des matières

| | |
|---|----|
| 1 - Présentation générale du système | 3 |
| 1.1 – Présentation de l’entreprise GEMEB et son intention | 3 |
| 1.2 – Analyse de l’existant | 3 |
| 1.3 – Expression du besoin | 3 |
| 1.4 – Schéma général | 4 |
| 1.5 – Répartition des tâches | 5 |
| 2 – Description de la partie personnelle..... | 6 |
| 2.1 – Ma responsabilité..... | 6 |
| 2.2 – Les étapes de réalisation | 6 |
| 2.3 – Planification | 6 |
| 3 – Diagramme de cas d’utilisation et d’exigences | 7 |
| 4 - Réalisation du projet..... | 8 |
| 4.1 – L’espace membre | 8 |
| 4.2 – Consommation | 15 |
| 4.3 - Changement de mot de passe | 20 |
| 4.4 – Facturation..... | 22 |
| 4.5 – Tableau d’utilisateur | 24 |
| 4.6 – Modification des clients | 26 |
| 4.7 – Désabonnement des clients..... | 26 |
| 4.8 - Déconnexion | 27 |
| 5 – Test unitaire..... | 27 |
| 5.1 – Test espace membre/client/administrateur | 27 |
| 5.2 – Test de l’affichage des données de la BDD | 28 |
| 6 – Conclusion | 29 |
| 7 - Annexe | 30 |

1 - Présentation générale du système

1.1 – Présentation de l'entreprise GEMEB et son intention

La société GEMEB commercialise des appareils laser à visée médicale : dermatologie, médecine esthétique. Elle souhaite développer son activité sur la location d'appareil médical, par le nombre de flashes effectué avec la machine.

1.2 – Analyse de l'existant

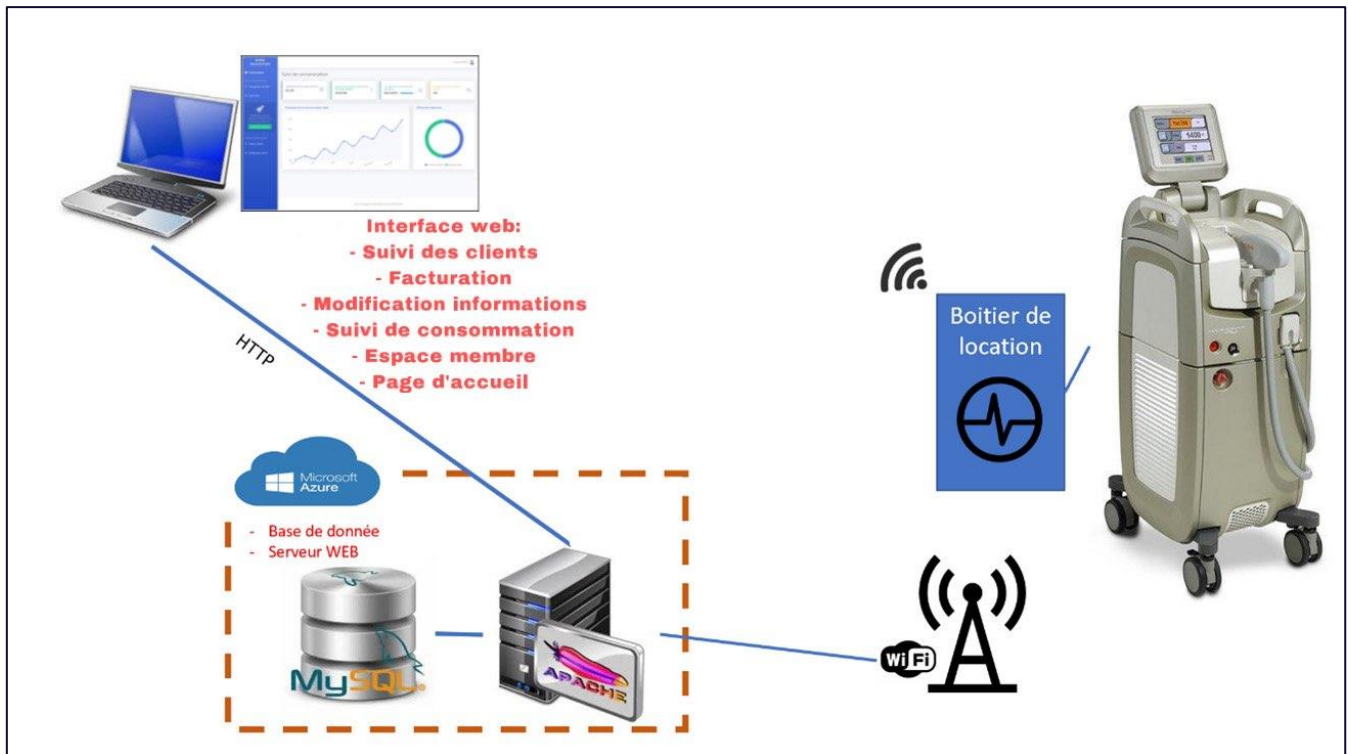
Les appareils lasers commercialisé par GEMEB permette de traiter des problèmes de peau, de faire de l'épilation définitive ainsi que de l'amaigrissement. Ces appareils ont un coût d'achat élevé. Afin d'augmenter son marché, GEMEB souhaite proposer aux médecins de louer ces appareils. La location par mois n'est pas attractive pour les praticiens. Une location en fonction de l'utilisation est plus attrayante puisqu'elle propose une facturation seulement si des actes médicaux ont été effectuées avec l'appareil en location.

1.3 – Expression du besoin

Le besoin de GEMEB et d'avoir une solution pour connaître l'utilisation de la machine et de facturer le médecin en fonction de l'utilisation de l'appareil. Il n'est pas question de modifier les appareils, car ceux-ci fabriqués par des sociétés qui ont obtenu un CE médical dans la configuration actuelle. Pour promouvoir leur développement, GEMEB souhaite disposer d'un boîtier permettant de s'adapter à la majorité des appareils du marché.

Ces appareils ont une augmentation de la consommation électrique lors du flash lumineux ou de l'alimentation de la diode laser. L'idée est de détecter cette surconsommation afin de facturer les clients au nombre de flashes lumineux ou durée d'utilisation du laser. Le client ne doit être facturé lorsqu'il laisse la machine en veille. Ainsi le montant de la location sera indexé sur les actes médicaux effectués. Les consommations du client doivent alors être transmises à un serveur permettant d'adresser la facturation au praticien chaque mois. GEMEB souhaite aussi fournir des statistiques de rentabilité au client afin qu'il réfléchisse à la possibilité d'acheter la machine.

1.4 – Schéma général



1.5 – Répartition des tâches

Tache IR 1 – Architecture réseau – Guylian FLOC'H

L'étudiant doit mettre en place sur un cloud l'architecture réseau du système

Détails de la tâche :

- Installation et configuration d'un serveur MySQL
- Installation et configuration d'un serveur Apache
- Création de la BDD
- Envoie des emails automatiques aux clients
- Communication des données

Tache IR 2 – IHM WEB – Mattéo ELEOUET

L'étudiant doit développer l'IHM web permettant à GEMEB de gérer ses locations.

Détails des fonctionnalités de l'IHM :

- Gestion des clients (ajout, suppression, édition)
- Gestion des contrats de location (ajout, suppression, édition)
- Visualisation des consommations des clients par machine
- Visualisation de la facturation
- Visualisation d'une page d'accueil pour les clients

Tache EC 1 – Détection le nombre de flashes consommés – Thai hong LE

L'étudiant doit développer l'électronique permettant d'effectuer la détection du nombre de flash utilisé par

Détails de la tâche :

- Choix des composants
- Routage et réalisation de la carte électronique.
- Tests et validation avec le terminal Windows en RS232
- Mise en place de la communication Wifi vers le serveur Web.

2 – Description de la partie personnelle

2.1 – Ma responsabilité

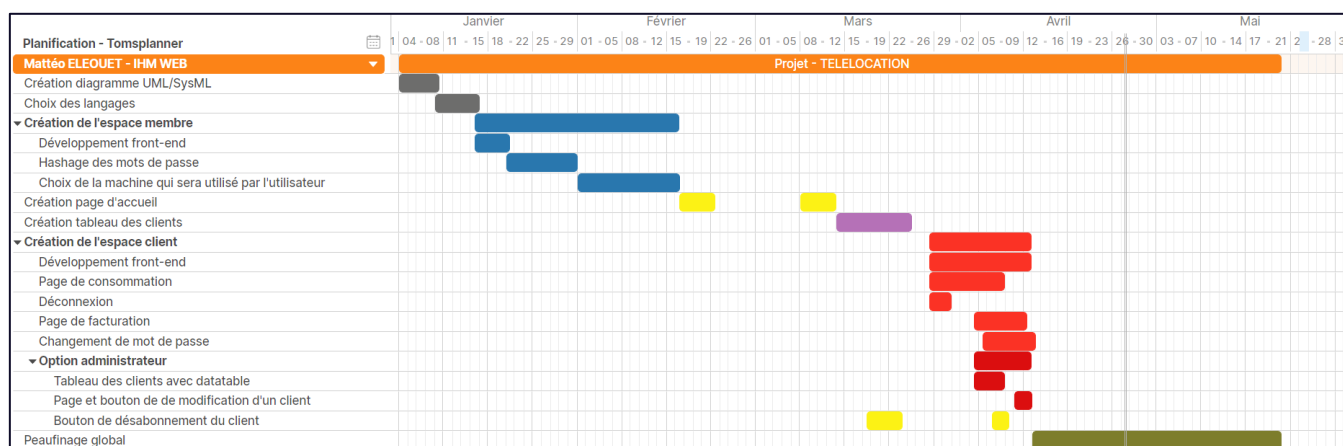
Durant ce projet mon rôle a été de faire une IHM WEB, cette IHM doit permettre d'accueillir le client sur une page sobre et le conduire vers son suivi de consommation de flashs, sa prochaine facture, une analyse de rentabilité d'achat de d'une machine d'épilation laser. Un administrateur de GEMEB doit pouvoir consulter la consommation de flash de chaque client, modifier ses informations en cas de perte de ses données d'identification, désabonné un client, consulter les machines disponibles à la location et celle qui sont utilisé et par qui elles sont utilisé.

2.2 – Les étapes de réalisation

La première étape a été l'analyse du cahier des charges pour comprendre le projet, pour mieux comprendre nous avons créé des diagrammes UML/SysML, cette étape occupe une semaine, c'est assez court.

Deuxième étape et pas des moindres le choix des langages de programmation qui vont être utilisé pour ce projet, pour une page WEB le HTML5 & CSS3 s'imposent comme des évidences, ce sont des standards. Le SQL pour manipuler la base de données MySQL ainsi que le langage qui va interpréter MySQL, au début j'étais parti sur du Django puis finalement sur du C++ pour finir sur le choix du PHP que je ne connaissais pas, mais qui était le plus adapté au projet à réaliser, car il est très facile avec de récupérer les informations entrées par l'utilisateur sur un formulaire et afficher des informations de la base de données pour créer un site WEB avec des pages dynamiques.

2.3 – Planification



3 – Diagramme de cas d'utilisation et d'exigences

Diagramme de cas d'utilisation

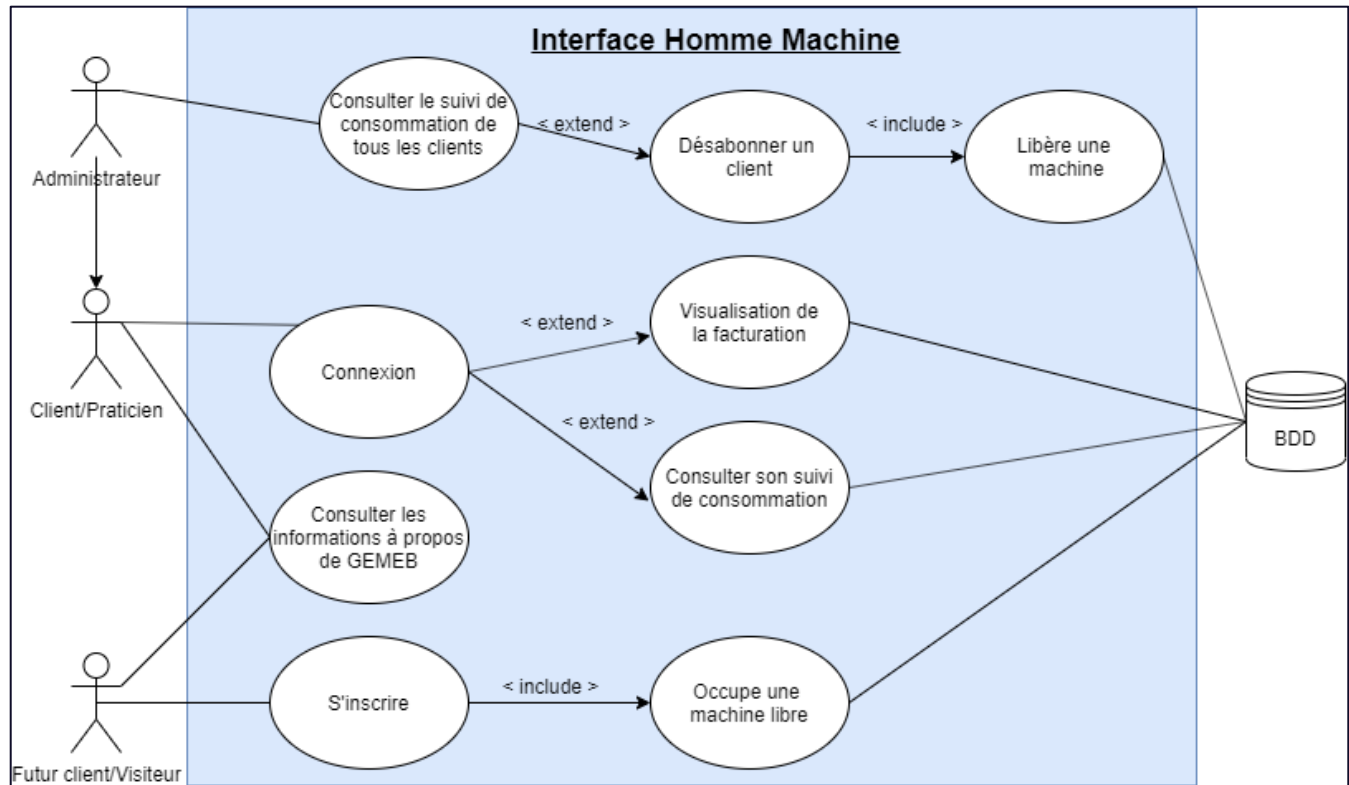
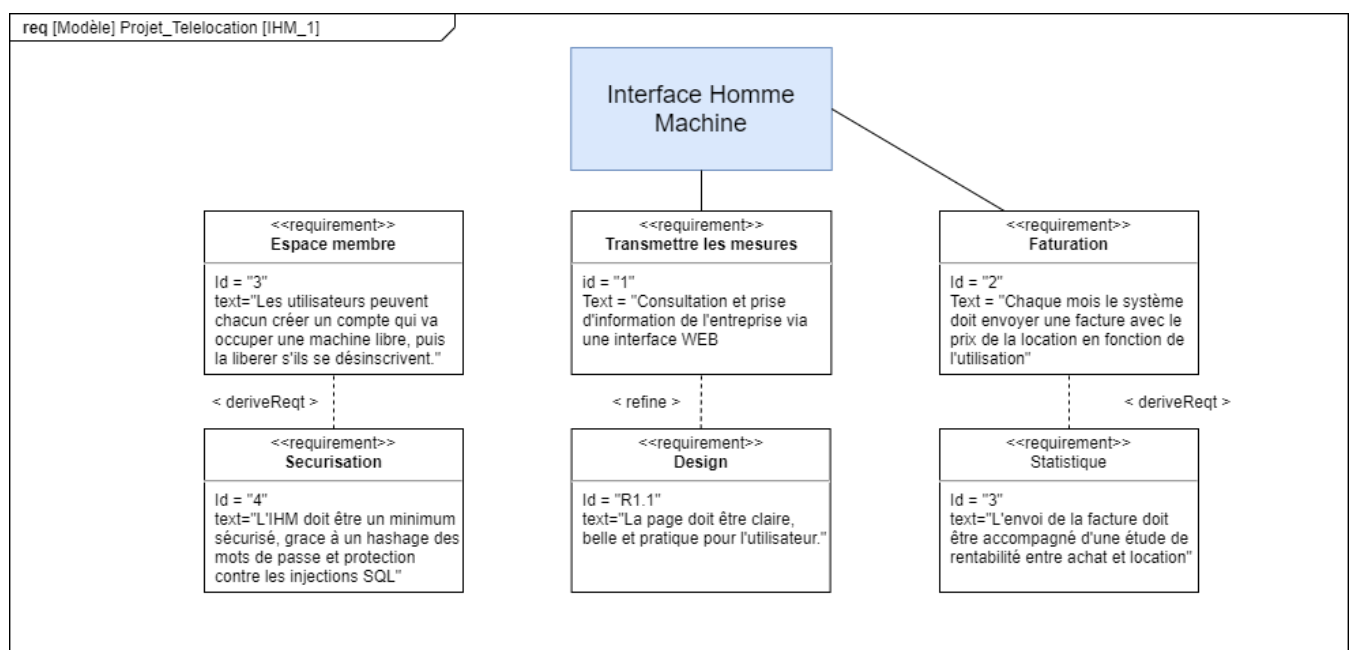


Diagramme d'exigence



4 - Réalisation du projet

4.1 – L'espace membre

Le début du projet débute par la création d'un espace membre, c'est-à-dire les comptes des futurs clients, et du choix des informations que les futurs clients vont rentrer. J'ai décidé de choisir de demander un numéro de téléphone, une adresse, un prénom, un nom, un mot de passe et évidemment un email qui sera très important pour répondre au cahier des charges, d'envoyer la facture par email. Je demande à mon camarade architecte réseau du projet, Guylian Floc'h de créer une table dans la base de données avec les champs que vous voyez ci-dessous :

| Nom | Prenom | MotDePasse | Email | Telephone | Adresse |
|-----|--------|------------|-------|-----------|---------|
|-----|--------|------------|-------|-----------|---------|

Après savoir quelles informations je vais demander aux futurs clients, je peux déjà commencer à créer l'IHM, une page d'inscription très simple, pour un rendu élégant je vais utiliser un Bootstrap. Bootstrap c'est un framework pour les développeurs frontaux, cela permet à un développeur de gagner du temps et de créer un site optimal sur toutes les tailles d'écran, que ce soit sur des téléphones à petit écran ou des ordinateurs de bureau à grand écran, cela donne un cadre à mon projet, la conception de ma page d'inscription reposera sur des fichiers CSS personnalisable et déjà personnalisé pour le formulaire que je vais créer. Le rendu donne ce que vous voyez ci-contre avec un champ pour chaque information que je souhaite recevoir, si l'utilisateur ne rend pas une information un message apparaîtra à l'utilisateur pour lui signifier que le formulaire doit être rempli entièrement.

Devenir client ?

!

!

Merci d'entrer votre adresse !

S'INSCRIRE !

Vous avez déjà un compte ?

CONNECTER VOUS !

A chaque saisi de donnée de l'utilisateur on doit donner à un nom à cet saisi de donnée. Pour le cas du formulaire d'inscription j'ai donné des noms très simple comme 'Email' ou 'MotDePasse' des noms très claire qui me permette de plus facilement m'y retrouver quand il faudra le stocker dans une variable en PHP.

```
<div class="wrap-input100 validate-input m-b-16" data-validate="Merci d'entrer votre e-mail">
  <input class="input100" type="email" name="Email" placeholder="Adresse mail">
  <span class="focus-input100"></span>
</div>
<div class="wrap-input100 validate-input m-b-16" data-validate = "Merci d'entrer votre mot de pas">
  <input class="input100" type="password" name="MotDePasse" placeholder="Mot de passe">
  <span class="focus-input100"></span>
</div>
```

Après que le client ait cliqué sur le bouton *S'INSCRIRE* cela enclenche le fichier *insert5.php* avec la méthode *POST* :

```
<form class="login100-form validate-form p-l-55 p-r-55 p-t-178" method="POST" action="scripts/insert5.php">
  <!-- formulaire... -->
</form>
```

La méthode *POST* me permet d'utiliser la variable superglobale `$_POST`, grâce à elle

```
$Email = htmlspecialchars($_POST['Email']);
$MotDePasse = $_POST['MotDePasse'];
$MotDePasseh = password_hash($MotDePasse, PASSWORD_BCRYPT);
$MotDePasse = $MotDePasseh;
$adresse = htmlspecialchars($_POST['Adresse']);
```

je peux récupérer les différentes saisis de donnée et les stocker dans une variable PHP.

Je vais utiliser la fonction `htmlspecialchars()` pour éviter que des informations malicieuses soient enregistrées dans la base de données j'applique cette fonction sur les données avant qu'elles soient enregistré dans la base de données auquel cas une cyber-attaque pourrait avoir lieux.

Après avoir récupéré le mot de passe je le *hash* de manière à sécuriser les mots de passe des clients, le hachage, c'est crypter un mot de passe de façon qu'il ne soit pas décryptable en cas de piratage de la base de données et de protéger les mots de passe d'une attaque de brute force (tester toutes les combinaisons possibles) ou tout simplement qu'un administrateur lui aussi ne devrait pas avoir accès aux mots de passe des utilisateurs. Le mot de passe est haché grâce à la fonction de PHP `password_hash()`, je donne comme paramètre le mot de passe saisi par l'utilisateur et le choix de l'algorithme de cryptage :

```
$MotDePasse = password_hash($MotDePasse, PASSWORD_BCRYPT);
```

Le choix de cryptage s'est fait sur BCRYPT, c'est un algorithme qui datant de 1999, il a montré son efficacité dans le temps, l'avantage de BCRYPT comparé à un hachage comme SHA2 ou SH512 est sa protection face aux attaques par brute force, pas forcément utile dans ce cas d'utilisation, un brute force sur un tel serveur ressemblerait vite à une attaque DDOS. En termes de sécurité il est lent à calculer,

demande au pirate une bonne configuration informatique notamment en VRAM, et possède un grand nombre itération de fonction cryptographique qui rend le calcul plus long, même s'il est moins efficace qu'un SHA512 mais pas la même consommation de performance pour le serveur et il est toujours estimé parmi les hachages les plus sécurisés du marché.

Maintenant que les mots de passe des utilisateurs sont hachés, il faut aussi penser qu'il est techniquement possible que deux clients puissent techniquement avoir entré la même adresse mail, comme il est possible que deux clients aient le même prénom ou nom de famille, pour un prénom/nom de famille ce n'est pas dérangeant, pour un email ça l'est. Selon le cahier des charges une facture sera envoyée par email tous les mois, cette facture indiquera la somme que le praticien doit payer à GEMEB. Dans la situation qu'un client entre la même adresse email qu'un autre client cela créerait une mauvaise situation puisse qu'un client recevrait les factures à payer d'un autre praticien. Pour éviter cette mauvaise situation on interdit à la base de données d'insérer une adresse email si elle est déjà renseignée dans la base de données. Pour résoudre cet éventuel problème, je crée un script qui sera exécuté à chaque nouveau compte.

```
2  $SELECT = "SELECT Email FROM registers WHERE Email = ? LIMIT 1";
3  $stmt = $connection->prepare($SELECT);
4  $stmt->bind_param("s", $Email);
5  $stmt->execute();
6  $stmt->bind_result($Email);
7  $stmt->store_result();
8  $rnum = $stmt->num_rows;
9  if($rnum == 0 && $comptePasDispo == 0)
10 {
11     # Execution...
12 }
13 else
14 {
15     echo 'Cette email est déjà utilisé';
16 }
```

La ligne numéro 2 sélectionne une adresse un email de la base de données qui sera pareil à l'email rentré par l'utilisateur, ensuite on prépare la requête PHP, on indique le type, c'est un string donc on met un 's' pour la variable `$Email`, on exécute et on retourne combien de fois qu'il trouve l'adresse mail entré par l'utilisateur dans sa base de données. Dans le cas que l'adresse email entré par l'utilisateur existe déjà dans la base de données `$rnum` sera égale à 1 puisque que MySQL n'aille pas chercher plus loin grâce l'instruction 'LIMIT 1', donc il ne sera pas possible de rentrer dans la condition `if($rnum == 0 && $comptePasDispo == 0)`, donc un message s'affichera pour indiquer que l'adresse email est déjà utilisé. Dans le cas contraire que l'adresse

email n'est pas utilisé `$rnum` sera égale à 0 et on aura une des deux conditions requises pour rentrer dans la condition du `if($rnum == 0 && $comptePasDispo==0)`. Avant de vous parler de la seconde condition pour rentrer dans cet `if($rnum == 0 && $comptePasDispo == 0)` je vais vous présenter comment fonctionne l'inscription d'un compte dans la base de données. Très souvent quand on crée un compte sur un site internet le site ajoute une nouvelle ligne et un nouvel identifiant, c'est la méthode utilisée avec un *INSERT INTO*. Pour le projet nous avons réfléchi et pensé à la possibilité que le nombre de machines à louer n'était pas en illimité, il est possible qu'il ait 20 ou 50 machines à louer et aussi qu'aucune machine ne soit disponible à la location, dans ce cas il n'est pas possible d'attribuer une machine à un client s'il n'y en a pas. Pour résoudre cet éventuel problème, j'ai utilisé la requête SQL UPDATE, nous établissons à l'avance un certain nombre de machines disponible à la location.

```
if($rnum == 0 && $comptePasDispo == 0)
{
    $sql1 = 'SELECT idClient FROM registers WHERE idClient = (SELECT MIN(idClient) WHERE registre=0)';
    $results = $bdd->query($sql1);
    $dataMin = $results->fetch(PDO::FETCH_ASSOC);
    $data2 = $dataMin['idClient'];
    $query = "UPDATE registers SET Adresse = '$adresse', Nom = '$_POST[Nom]', Prenom = '$_POST[Prenom]',
        MotDePasse='$MotDePasse', Email='$Email', Telephone='$_POST[Telephone]', registre=1, date=CURDATE(),
        datefact=date + INTERVAL 1 MONTH WHERE idClient = '$data2' and registre=0";
    $query2= "UPDATE Prestation INNER JOIN registers ON registers.idClient=Prestation.Client SET EtatMachine=1
        WHERE registers.registre=1";
    $query_run = mysqli_query($connection, $query);
    mysqli_query($connection, $query2);
    $stmt->close();
    if($query_run)
    {
        header("location: ../connexion.php");
        exit();
        echo 'Good';
    }
    else
    {
        echo 'Erreur';
    }
}
```

Je vais expliquer déjà la première ligne, quand on fait un *UPDATE* en SQL cela change les informations de toutes les lignes de la table, pour que cela change qu'une seule ligne je vais mettre une condition avec un *WHERE* mais pour savoir la quelle ligne je modifie j'ai choisi de modifier la ligne où la valeur de l'identifiant (*idClient*) est la plus faible avec l'instruction SQL *MIN()* que je stockerai dans la variable `$data2` après on fait la requête UPDATE avec un *WHERE* qui identifiera la machine avec la valeur la plus faible et qui n'est pas utilisé grâce au champ *registre*. La difficulté est que je ne savais pas qu'on n'avait pas le droit d'utiliser l'instruction *MIN()* dans un UPDATE d'où la raison du pourquoi je n'ai pas directement rentré l'instruction *MIN()* directement et que j'ai stocké l'information dans une variable.

On met à jour toutes les informations rentrer par le client, et on ajoute la date de création du compte du client dans le champ *date*, et on indique que la prochaine facture sera dans un mois après la création du compte dans le champ *datefact*. On exécute la requête SQL avec la fonction `mysqli_query`, si la requête est bonne elle mènera le client vers la page de connexion, si elle est mauvaise, elle indiquera au client qu'il y a eu une erreur, ce qui n'est pas sensé arrivé, c'est plutôt un message réservé au développeur.

```
$sqlRegistre = "SELECT registre FROM registers ORDER BY idClient DESC LIMIT 1";
$resultsRegistre = $bdd->query($sqlRegistre); // 8
$registreMax = $resultsRegistre->fetch(PDO::FETCH_ASSOC);
$comptePasDispo = $registreMax['registre'];

if($rnum == 0 && $comptePasDispo == 0)
{
    # Execution...
    {
        header("location: ../connexion.php");
        exit();
        echo 'Good';
    }
}
```

Avant d'autoriser l'utilisateur de faire tout ça il faut aussi il vérifie qu'il y a une machine disponible

l'instruction SQL `ORDER BY idClient DESC LIMIT 1` permet de sélectionner le registre de l'idClient ayant la plus grande valeur, si le registre est égal à 1 cela signifie que la machine est occupée, si le registre renvoie 0, c'est qu'elle est libre donc qu'il y a au minimum une machine disponible à la location.

Après la création du compte on redirige l'utilisateur vers la page de connexion avec un `header("location: ../connexion.php");` la fonction `header` permet rediriger l'utilisateur vers la page de connexion. La fonction `exit();` Permet de sortir du programme et de faire en sorte que le serveur ne le continue pas à lire inutilement le reste du script, c'est une bonne pratique à prendre en PHP, mettre un `exit();` permet d'interrompre l'exécution du script et d'optimiser un petit peu le programme.

S'il n'y a plus de machine disponible ou que l'email est déjà utilisé on affiche à l'utilisateur un message pour lui indiquer l'une des deux raisons de l'impossibilité de la création de son compte GEMEB.

```
else
{
    echo "Cette email est déjà utilisé ou il n'y pas plus de machine disponible à la location";
}
```

Maintenant les utilisateurs ayant un créé un compte GEMEB TELELOCATION peuvent se connecter.

```
<form class="login100-form validate-form p-l-55 p-r-55 p-t-178" method="POST" action="scripts/connexion1.php">  
<span class="login100-form-title">
```

Ci-dessus un extrait du code HTML, on fait un comme à la page d'inscription, un formulaire qui prendra la méthode POST et une action qui mène vers un script qui se nomme *connexion1.php* qui va exécuter des fonctions PHP et requête SQL.

```
3  if(isset($_POST['Email']) && isset($_POST['MotDePasse']))  
4  {  
5      // connexion à la base de données  
6      $db_Email = 'root';  
7      $db_MotDePasse = 'willy9105';  
8      $db_name      = 'test2';  
9      $db_host       = 'localhost';  
10     $db = mysqli_connect($db_host, $db_Email, $db_MotDePasse,$db_name)  
11     or die('La connexion à la base de données a échoué');
```

La première ligne du code est un `isset`, cette fonction détermine si une variable possède un contenu, dans le cas ici il faut que l'utilisateur ait rempli son adresse mail et un mot de passe pour rentrer dans la condition if qui se connecte à la base de données. Il est possible de se connecter de deux manières à une base de données avec PHP soit PDO, soit avec MySQLi, pour le fichier de connexion, j'utilise l'extension MySQLi pour les requêtes préparées et donc éviter les injections SQL ce qui est important pour une page de connexion. Si la connexion à la base de données échoue j'utilise un `die` (qui équivaut à un `exit`) qui affiche un message le message suivant : `La connexion à la base de données a échoué` qui indique que les identifiants de connexion à la base de données sont faux ou que le service MySQL n'est pas activé.

```
$Email = mysqli_real_escape_string($db,$_POST['Email']);
$MotDePasse = $_POST['MotDePasse'];

$requete = "SELECT MotDePasse FROM registers WHERE Email = '". $Email."' ";
$exec_requete = mysqli_query($db,$requete);
$reponse      = mysqli_fetch_array($exec_requete);
$Mdp_h = $reponse['MotDePasse'];
if(password_verify($MotDePasse, $Mdp_h))
{
    $_SESSION['Email'] = $Email;
    $_SESSION['idClient'] = $idClient;
    header('Location: ../consommation.php');
    exit();
}
else
{
    header('Location: ../connexion.php?erreur=1');
    exit(); // utilisateur ou mot de passe incorrect
}
```

Premièrement pour protéger le site des attaques par injection SQL je vais utiliser la fonction `mysqli_real_escape_string()` pour être sûr de faire une requête SQL valide qui ne comporte pas un caractère comme ' \n \r \0 " caractère nécessaire à l'injection SQL. Je sélectionne le mot de passe de l'utilisateur connecter en préparant puis en exécutant l'instruction SQL et je compare le mot passe haché dans la base donnée et le mot de passe entré par l'utilisateur avec la fonction `password_verify()`, si les mots de passe se correspondes on redirige l'utilisateur vers sa page de consommation qui se nomme *consommation.php* avec toujours un `exit()`. Si les mots de passe se ne correspondent pas on redirige le client vers la page de connexion avec comme message lui indiquant que le mot de passe ou l'email est correct. L'affichage de ce message d'erreur est fait grâce au nouveau lien

```
$etatErreur = $_GET["erreur"];
if($etatErreur == 1)
{
    echo "
    <div class='flex-col-c p-t-20 p-b-0'>
        <span class='txt1 p-b-9'>
            Mot de passe ou email incorrect
        </span>
    </div>";
}
```

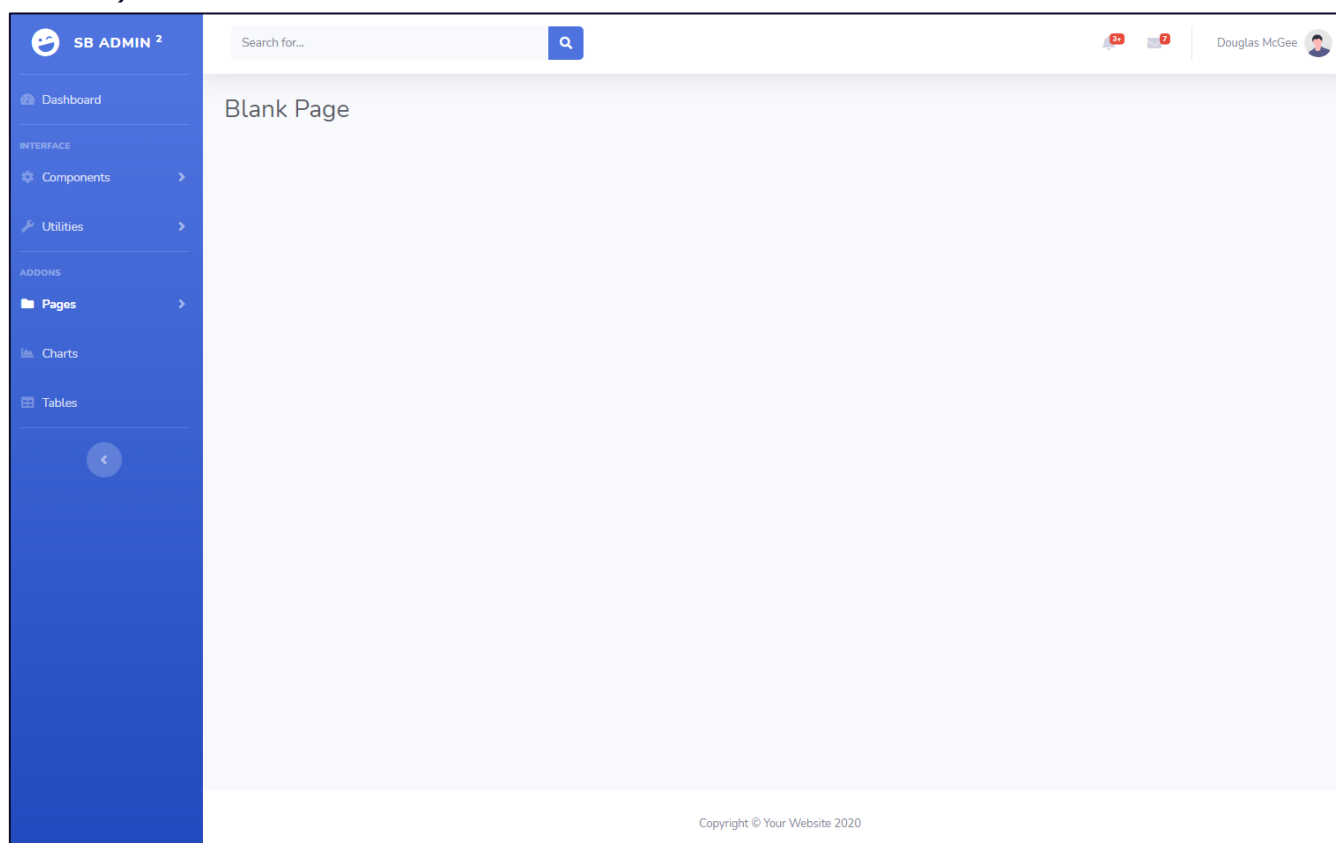
où sera rediriger le client, le lien comportera un paramètre `?erreur=1` qui permet de faire une condition if de l'affichage de code HTML en plus, cette valeur qui sera capturé avec un `$_GET`.

4.2 – Consommation



Pour faire la page de consommation, je vais encore utiliser le framework CSS « Bootstrap ». L'utilisation de ce framework va permet de donner un cadre de travail et de ne pas partir dans tous les sens, gagner en temps et en efficacité grâce aux outils fournis (CF annexe). Développer mon site web sans me soucier du rendu par les différents navigateurs, du périphérique ordinateur, téléphone ou tablette et pouvoir faire un beau site sans être un héros du CSS. Pour faire la page de consommation j'ai choisi une blank page de Bootstrap avec un menu à gauche qui

me permettra de placer mes fonctionnalités (Gestions des clients : ajout, suppression et édition, visualisation des consommations des clients par machine, visualisation de la facture, gestion des contrats de location : ajout, suppression et édition)

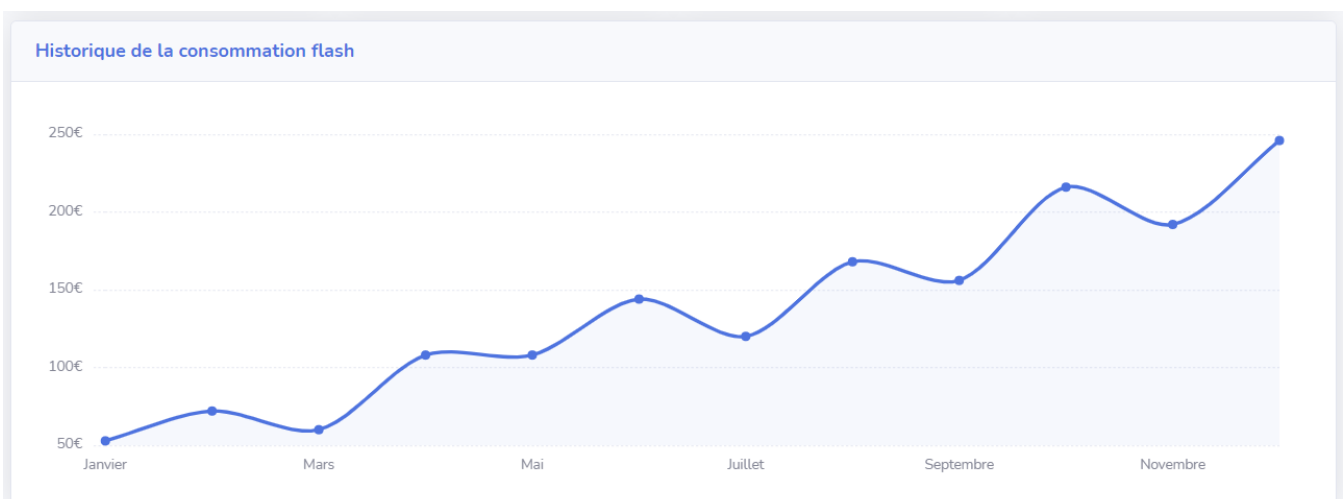


J'ai finalement enlevé la flèche inférieure qui permettait de réduire le menu, enlever les notifications et barre de recherche qui ne me serviront pas afin d'épurer le site et ne pas laisser des fonctionnalités je n'utiliserai pas.

Ci-contre une capture d'écran de la barre latéral gauche du site avec les fonctionnalités du cahier des charges. J'ai décidé de laisser trois fonctionnalités au client, celle de consulter son suivi de consommation, changer son mot de passe et consulter sa facture. Si l'utilisateur est un administrateur, il aura accès à deux fonctionnalités en plus : *Tableau clients* / *Modification clients*, à savoir que les autres utilisateurs ne voient pas les options administrateurs à leur écran. Pour la page de consommation, j'ai décidé d'intégrer quatre encadrés, pour afficher les quatre informations suivantes au client : son compteur mensuel du mois en cours, sa dépense totale depuis son abonnement chez GEMEB, combien représente sa dépense totale en pourcentage par rapport à l'achat de la machine d'épilation directement avec l'affichage d'une barre de progression faite en CSS qui évolue selon le pourcentage, le nombre de flashes utilisé dans le mois en cours.



Et plus bas l'intégration d'un graphique de l'historique de la consommation flash :



Pour intégrer ce graphique à mon site, j'utilise la bibliothèque open source de JavaScript « ChartJS », l'avantage est une intégration facile et compatible à tous les navigateurs modernes, très léger par rapport aux autres solutions existantes, une documentation très bien faite et un large choix de style de graphique disponible. Toute l'implémentation de ces modules pour ce rendu :



La page *consommation.php* regorge d'information de la base de données MySQL. Pour lire une information de cette base de données on exécute une instruction SQL comme `SELECT * FROM registers WHERE Email = '$email';` qui sélectionnera tous champs de la table registers pour l'utilisateur ayant le même email que celui

```
$sqlUser = "SELECT * FROM registers WHERE Email = '$email'";
$results = $PDO->query($sqlUser);
while ($data = $results->fetch(PDO::FETCH_ASSOC)) {
    $nomUser = $data['Nom'];
    $prenomUser = $data['Prenom'];
    $idClient = $data['idClient'];
    $etatAdmin = $data['EtatAdmin'];
}
```

connecté à la page. Ensuite je vais stocker la valeur de chacun de ces champs dans une variable PHP avec la fonction de la PDO de PHP: `fetch(PDO::FETCH_ASSOC)`.

Maintenant je vais répéter ce processus à chaque fois que j'ai besoin de capturer une information de la base de données, comme pour afficher le prénom, le nom ou sa consommation de flashes avec cette méthode.

La page *consommation.php* est une page réservée aux clients, il est donc nécessaire que seul les clients/utilisateurs connectés aient accès à un suivi de consommation,

```
1  <?php session_start();
2  if ($_SESSION['Email'])
3  {
4      # Execution...
5  }
6  else {header("Location: connexion.php");}
```

alors je vais rediriger tous les utilisateurs qui ne sont pas connecté vers la page de connexion. Le programme démarre par la fonction `session_start()` qui sert à

démarrer une nouvelle session ou reprendre une session existante. Maintenant que nous avons une session via les cookies, je demande si la il possède une adresse mail dans ses cookies, nous lisons les cookies avec la variable superglobal `$_SESSION`, si c'est la variable renvoie 1 c'est qu'il est connecté alors l'utilisateur rentre dans la condition et se connecte à la base de données et pourra charger la page *consommation.php*, sinon on va rediriger cet utilisateur vers la page de connexion et l'inviter à se connecter afin de pouvoir l'identifier.

La page affiche les options classiques pour les clients, mais il existe aussi les *options administrateurs*, mais seuls les administrateurs doivent pouvoir voir ces options

```
254 <?php
255 if ($etatAdmin == 1) {
256     // Divider
257     echo "<hr class='sidebar-divider d-none d-md-block'>
258
259     <div class='sidebar-heading'>
260         Option administrateur
261     </div>
262
263     <li class='nav-item'>
264         <a class='nav-link' href='tableau-user.php'>
265             <i class='fas fa-fw fa-table'></i>
266             <span>Tableau clients</span></a>
267         </li>
268
269     <li class='nav-item'>
270         <a class='nav-link' href='modification-client.php'>
271             <i class='fas fa-fw fa-folder'></i>
272             <span>Modification clients</span></a>
273         </li>";
274 }
```

s'afficher, pour cela je récupère dans une variable l'information si l'utilisateur est un administrateur ou non avec un `fetch`, ensuite j'affiche le contenu administrateur seulement pour les *options administrateurs* avec une condition `if($etatAdmin==1)` cette condition est vraie alors on rentre dans la condition et on affiche les options administrateur avec les liens vers ces fonctionnalités.

Avant de vous montrer les informations que j'affiche, je vais vous présenter rapidement à quoi ressemble la table *date1* de la base de données faite par mon camarade Guylian Floc'h qui comporte un champ pour chaque mois de l'année.

| idClient | Client | Janvier | Fevrier | Mars | Avril | Mai | Juin | Juillet | Aout | Septembre | Octobre | Novembre | Decembre |
|----------|--------|---------|---------|------|-------|-----|------|---------|------|-----------|---------|----------|----------|
| 1 | 1 | 0 | 0 | 0 | 0 | 214 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 4 | 44 | 60 | 50 | 90 | 90 | 120 | 100 | 140 | 130 | 180 | 160 | 205 |

Pour sélectionner le bon mois je vais utiliser la fonction trouver par mon camarade de projet Guylian Floc'h qui est `date('m')`; de PHP qui va renvoyer une valeur entre 01 et 12, exemple si nous sommes en juin donc `date('m')`; retournera la valeur 06 avec cette valeur tout ce que j'ai à faire c'est de stocker le nombre de flashes de juin dans la variable `$nbFlash` et afficher le nombre de flashes du moins en cours. Je stocke la valeur retourner de date dans la variable `$month` `$month = date('m');`

```
elseif($month=="06")
{
    $sqlSelectMonth = "SELECT Juin FROM date1 WHERE Client = '$idClient'";
    $resultsMois = $PDO->query($sqlSelectMonth);
    $dataSelectMois = $resultsMois->fetch(PDO::FETCH_ASSOC);
    $nbFlash = $dataSelectMois['Juin'];
}
```

Je fais un `elseif/if` à 12 propositions qui entrera l'instruction SQL qui sélectionnera le nombre de flashes enregistré pour le mois en cours, je vais pouvoir stocker le nombre de flashes en consommation dans une variable que je vais nommer `$nbFlash`.



Pour afficher le nombre de flashes, je récupère l'information dans la base de données MySQL avec la méthode expliquée précédemment et j'utilise un `echo` pour afficher le contenu de ma variable `$nbFlash`.



Un module où j'additionne tous les mois * le prix d'un flash que j'ai donné arbitrairement à 1,2 €. Même méthode que précédemment avec un `echo` pour afficher le contenu de la variable.



Ce module est résultat du compteur total donné précédemment diviser par le prix de la machine qui est 20'000€ que j'ai aussi donné arbitrairement le tout divisé par 100 pour donner un pourcentage. Ce module possède une jauge qui évolue selon le pourcentage ceci est fait en CSS je définis la largeur de la barre en % avec un `echo` pour coller les informations de ma variable.

```
style="width: <?php echo $machine; ?>%"
```



Ce module ressemble beaucoup au premier module présenter, il prend en compte le nombre de flashes, mais en plus il les multiplie par le prix du flash le prix d'un flash, soit $90 * 1,2$.

Pour afficher les informations de mes variables PHP je vais inclure le code JavaScript



du graphique dans du PHP comme on le fait avec le HTML avec les balises `<script>`. J'implémente avec un echo les données de chaque variable de chaque mois dans un tableau en JavaScript.

4.3 - Changement de mot de passe

Le cahier des charges demande la possibilité que l'utilisateur puisse changer ses informations, une option du site propose à l'utilisateur de modifier son mot de

A screenshot of a password reset form. The title is 'Réinitialisation du mot de passe'. Below the title, there is a message: 'Veuillez remplir ce formulaire afin de réinitialiser votre mot de passe'. Another message states: 'Votre mot de passe devra contenir au mini 6 caractères'. There are two input fields: 'Nouveau mot de passe' and 'Confirmez le mot de passe'. At the bottom, there are two buttons: 'Valider' (in blue) and 'Annuler'.

passe. Cette page fonctionne avec un formulaire couplé de la méthode POST pour communiquer avec le script du mot de passe. Je demande à l'utilisateur d'entrer un nouveau mot de passe six caractères ou plus. Comme c'est une page réserver aux clients vérifie si l'utilisateur est un client et je vérifie si l'utilisateur est connecté comme il a été fait pour la page *consommation.php*

```
if(empty(trim($_POST["new_password"])))
{
    $new_password_err = "Veuillez entrer votre nouveau mot de passe."; }
elseif(strlen(trim($_POST["new_password"])) < 6)
{
    $new_password_err = "Votre mot de passe doit contenir au moins 6 caractères."; }
else
{
    $new_password = trim($_POST["new_password"]); }
// Valider la confirmation du mot de passe
if(empty(trim($_POST["confirm_password"])))
{
    $confirm_password_err = "Veuillez confirmer votre mot de passe."; }
else
{
    $confirm_password = trim($_POST["confirm_password"]);
    if(empty($new_password_err) && ($new_password != $confirm_password))
    {
        $confirm_password_err = "Les mots de passe ne correspondent pas."; }
    }
}
```

La première ligne de vérifier si l'utilisateur a au moins renter un mot de passe avec la fonction `empty()` qui va déterminer si une variable est vide ou non. Si l'utilisateur a rentré un mot de passe le programme ne rentre pas dans la première condition, il passe à la seconde condition qui vérifiera que le mot de passe possède au minimum six caractères je vais utiliser la fonction PHP `strlen()` qui calcule la taille d'une chaîne de caractère, si la valeur retournée est inférieure à 6 j'affiche un

Nouveau mot de passe

Votre mot de passe doit contenir au moins 6 caractères.

message demandant à l'utilisateur d'entrer un mot de passe d'au moins six caractères comme montrer ci-contre.

Si le mot de passe réussit les deux premières vérifications, le programme entre dans la dernière condition, le programme traite le mot de passe avec la fonction `trim()`, qui supprime les espaces/tabulation ou autres caractères invisibles que l'utilisateur n'aurait pas vus. Pour vérifier que les mots de passe se correspondent j'utilise l'opérateur logique `!=` qui signifie « *différent de* », si les deux les variables sont

Confirmez le mot de passe

Les mots de passe ne correspondent pas.

différentes le programme affiche un message signifiant à l'utilisateur que les mots de passe ne se correspondent pas.


Si le mot de passe réussit à passer tous les tests alors j'utilise l'instruction SQL `UPDATE` avec un mot de passe haché et déconnecte l'utilisateur avec la fonction `session_destroy()` qui détruit une session et invitait l'utilisateur à se reconnecter

```
if(mysqli_stmt_execute($stmt)){
    // Mot de passe mis à jour avec succès.
    session_destroy();
    header("location: ../connexion.php");
    exit();
}
```

avec son nouveau mot de passe en le redirigeant vers la page de connexion et un `exit()`.

4.4 – Facturation

La facturation est un document très important dans le suivi d'une comptabilité sérieuse. En effet, ce document en plus d'assurer le suivi comptable et d'éviter les problèmes entre acheteur et vendeur. La facturation comporte les informations légales comme le nom de l'entreprise, l'adresse le numéro, ainsi que les informations du client comme son adresse email, son nom et prénom. La facture indique la date de facturation qui est calculé grâce à un programme backend fait par mon camarade de projet Guylian Floc'h, tous les mois le programme incrémente la date de facturation et le 12 il passe à 01 qui signifie le mois de janvier.

| | | | |
|---|--|---|--|
|  | | GEMEB 230, rue de Faubourg-Saint-Honoré 01 45 63 55 31 admin@gemeb.site | |
| FACTURE DE: ROBINO Matteo admin@gemeb.site | | FACTURE TELELOCATION Date de facturation: 2021-06-10 | |

Le programme sélectionne aussi le mois en cours exactement de la même manière qu'il a été fait pour la page *consommation.php*. On calcule le coût total avec une

```
$prixDuFlash = 1.2;
$totalFlash= $nbFlash * $prixDuFlash;
$sousTotal = $totalFlash;
$taxe = $sousTotal * 0.20;
$grandTotal = $sousTotal + $taxe;
```

simple multiplication, le prix du flash est le même que sur le page *consommation.php* et la taxe est de 20% sans vraiment savoir combien de taxe un praticien payerai vraiment pour ce genre de service.

| # | DESCRIPTION | PRIX A L'UNITE | QUANTITE | TOTAL |
|-------------|---|----------------|----------|--------|
| 1 | Consommation de flash Utilisation du nombre de flash de la machine d'épilation | 1.2€ | 90 | 108€ |
| 2 | Frais supplémentaire | 0€ | 0 | 0€ |
| SOUS-TOTAL | | | | 108€ |
| TAXE 20% | | | | 21.6€ |
| GRAND TOTAL | | | | 129.6€ |

Le cahier des charges demande aussi une étude de rentabilité entre achat et location. N'ayant pas connaissance du prix d'une telle machine, j'ai spéculé sur un prix rond de 20'000 €. Pour faire cette étude d'achat de rentabilité, j'ai décidé de stocker dans une variable la valeur de tous les mois, que je divise par le nombre de mois actif détecter par une condition `if`.

```
$tousLesMois = $fevrier + $janvier + $fevrier + $mars + $avril + $mai + $juillet +  
    $juin + $aout + $septembre + $octobre + $decembre + $novembre;  
if ($janvier > 0)      {$moisActif++;}  
if ($fevrier > 0)      {$moisActif++;}  
if ($mars > 0)         {$moisActif++;}  
if ($avril > 0)        {$moisActif++;}  
if ($mai > 0)          {$moisActif++;}  
if ($juin > 0)         {$moisActif++;}  
if ($juillet > 0)      {$moisActif++;}  
if ($aout > 0)         {$moisActif++;}  
if ($septembre > 0)    {$moisActif++;}  
if ($octobre > 0)      {$moisActif++;}  
if ($decembre > 0)     {$moisActif++;}  
if ($novembre > 0)     {$moisActif++;}  
$moyenneMoisActif = $tousLesMois / $moisActif;
```

Je porte l'étude sur une location de 3 ans donc je fais `$bilanAchat = $moyenneMoisActif * 36`; et si j'obtiens une valeur supérieure au prix d'achat d'une machine d'épilation j'affiche à l'utiliser dans sa facture un message lui conseillant d'acheter une machine d'épilation par flash.

Etude d'achat de machine:

Une machine GEMEB d'épilation coûte 20000€ sur une location de 3 ans vous payerai 36000€, nous vous invitons à vous intéressé à l'achat d'une machine d'épilation

Si la valeur obtenue est inférieure au prix d'achat d'une machine d'épilation, j'affiche un message dans la case de l'étude de rentabilité lui conseillant au client de continuer de louer la machine d'un point de vue économique sur trois ans.

Etude d'achat de machine:

Une machine GEMEB d'épilation coûte 20000€ sur une location de 3 ans vous payerai 4287€, nous vous conseillons de continuer à louer notre machine.

4.5 – Tableau d'utilisateur

Cette page est une page administrateur, réserver aux administrateurs, tout utilisateur qui n'est pas administrateur sera redirigé vers la page *consommation.php*. Pour vérifier si l'utilisateur est un administrateur, c'est très simple, une condition *if*.

```
if ($etatAdmin == 0) {
    header("Location: consommation.php");
}
```

Pour faire le tableau, j'ai encore décidé de pas réinventer la roue et utiliser un plug-in de tableau déjà existant. J'ai utilisé le plug-in DataTables qui est open source, c'est facile à intégrer, beaucoup d'options sont proposées et cela me permet de dynamiser un tableau HTML et de ne pas à devoir le faire moi-même non plus. Énormément d'entreprise utilise ce plug-in pour leur site web comme : Amazon / la NASA / CISCO / TESLA et beaucoup d'autres multinationales, c'est une référence pour l'utilisation de tableau de dans un site Web.

Table de données des utilisateurs inscrits.

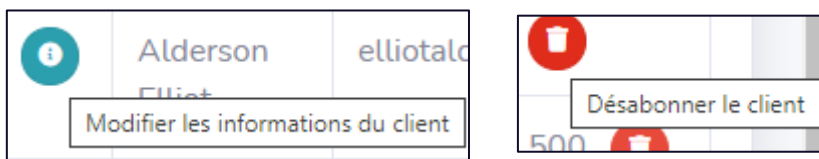
Montrer 10 clients Chercher :

| Num client | Nom | Email | Telephone | Adresse | date de création | Nombre flash |
|------------|-----------------|----------------------------|----------------|---|------------------|--------------|
| 1 | Alderson Elliot | elliotalderson@hack.com | 03.45.76.23.45 | Allsafe Security 34 New York | 2021-05-08 | 1494 |
| 10 | Noel Pere | perenoel@yahoo.fr | 3630 | Rue des montagnes enneigé | 2021-04-11 | 500 |
| 11 | Odelette Leduc | OdeletteLeduc@armyspyf.com | 01.55.41.79.26 | 5 boulevard de Europe Brest 29200 | 2021-04-11 | 0 |
| 12 | Bizier Benoît | BenoitBizier@dayrep.com | 04.76.03.93.22 | 5, Avenue De Marlioz 74100 ANNEMASSE | 2021-04-11 | 0 |
| 13 | Macron Emmanuel | emmanuelmacron@wanadoo.fr | 0667438362 | Rue des rois | 2021-05-23 | 0 |
| 14 | Ancelote Blais | AnceloteBlais@teleworm.us | 02.17.86.34.37 | 47, boulevard Aristide Briand 76120 LE GRAND-QUEVILLY | 2021-04-11 | 100 |

Pour afficher de manière dynamique la liste des clients ainsi que leurs informations je vais utiliser une boucle *while* et faire un *echo* de chaque valeur que je veux afficher. J'ai décidé d'afficher le numéro du client, le prénom, le nom, l'adresse email, le numéro de téléphone du client, son adresse physique et la date de création de son compte.


```
<?php while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) : ?>
<tr>
<td><?php echo htmlspecialchars($row['idClient']);
echo "&nbsp; &nbsp; &nbsp; ";
echo "<a href='modification-client.php?idClient=".$row["idClient"].
'' title ='Modifier les informations du client' class='btn btn-info btn-circle btn-sm'>
<i class='fas fa-info-circle'></i>"; ?>
</td>
<td><?php echo htmlspecialchars($row['Nom']);
echo " ";
echo htmlspecialchars($row['Prenom']); ?></td>
<td><?php echo htmlspecialchars($row['Email']); ?></td>
<td><?php echo htmlspecialchars($row['Telephone']); ?></td>
<td><?php echo htmlspecialchars($row['Adresse']); ?></td>
<td><?php echo htmlspecialchars($row['date']); ?></td>
<td><?php echo htmlspecialchars($row['NbFlash']);
echo "&nbsp; &nbsp; &nbsp; &nbsp; ";
echo "<a href='scripts/sup2.php?idClient="
.$row["idClient"]."'' title='Désabonner le client' class='btn btn-danger btn-circle btn-sm'>
<i class='fas fa-trash'></i></a>";
```

Grâce à cette boucle le tableau va charger tous les utilisateurs abonnés, j'affiche chaque information de chaque client avec un `echo` et deux boutons fait avec le framework CSS Bootstrap pour faire deux petites icônes, un bouton d'information qui comporte chacun une infobulle HTML qui indique à quoi sert chacun de ces petits boutons.



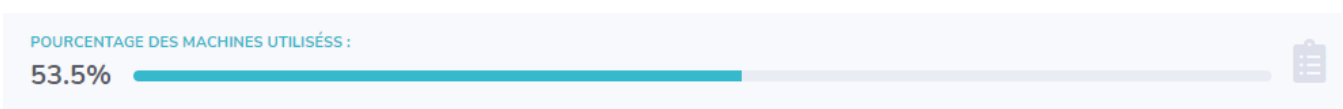
Ces boutons possèdent un lien qui ramène vers de la page de modification des informations du client ou alors vers une page qui va désabonner un client.

J'ai aussi intégré un bouton qui insère une machine en plus dans la base de données, au cas où l'entreprise GEMEB souhaiterait investir dans plus de machine de location.



C'est un court programme qui utilise l'instruction SQL `INSERT INTO` le code de ce programme est en annexe si vous voulez y jeter un coup d'œil.


J'utilise une carte Bootstrap comme je l'ai fait pour la page *consommation.php*, je l'utilise pour afficher une barre de progression afin d'indiquer le pourcentage de machine en cours de location.



4.6 – Modification des clients

La page de modifications des clients est aussi une page administrative au même titre que la page *Tableau-clients.php*, seuls les administrateurs peuvent accéder à cette page.

La page est très simple comporte un formulaire avec sept entrées textuel pour insérer les informations du client.

| | | |
|---|----------|-----------|
|  | Alderson | elliotald |
| | Elliot | |
| Modifier les informations du client | | |

Si on clique sur le bouton de modification cela indique dans l'url le numéro du client à qui on veut changer les informations `/modification-client.php?idClient=1`

Dans l'exemple précédent on indique que `idClient=1` `$idClient = $_GET['idClient'];` je capture la valeur avec la super variable `$_GET` .et j'indique

Numéro d'identification du client :

par défaut la valeur du champ *Numéro d'identification du client* avec l'attribut HTML `value` `value="<?php echo $idClient;?>"` qui sera transmise par méthode POST qui actionne le

programme *update-client.php* qui fonctionne de la même manière que le programme d'inscription avec l'instruction *UPDATE* présenter précédemment.

4.7 – Désabonnement des clients

Le programme de désabonnement consiste à changer le statut *registre* de *1* qui signifie que la machine est en cours de location à *0* qui signifie que la machine est disponible à la location. Le programme va remplacer les informations du client par d'autres informations par défaut.

```
$sqlUpdate="UPDATE registers, Prestation SET
    registers.Nom = 'Le nom', registers.Prenom = 'Le prenom',
    registers.Adresse = 'Adresse', Prestation.NbFlash = 0 registers.Telephone = 667 ,
    registers.MotDePasse = 'LeMDP', registers.Email = 'adresse@ma.il',
    registers.registre = 0 , Prestation.EtatMachine = 0
WHERE registers.idClient = $ids AND Prestation.Client = $ids";
```

Pour ne perdre les informations des anciens clients, nous les gardons quand même et les stockons dans une autre table qui se nomme *OldClient* qui rangera dedans toutes les informations du client comme son email, son prénom, son nom et d'autre information afin de pourquoi pas proposer à l'entreprise d'envoyer des emails d'offre promotionnelle pour faire rapatrier le client vers GEMEB.

4.8 - Déconnexion

Le programme de déconnexion est le programme le plus simple de tous les programmes que je peux vous présenter, il est composé de 4/5 lignes de codes.

```
// Initialiser la session
session_start();
// Vider toutes les variables de session
$_SESSION = array();
// Détruire la session.
session_destroy();
// Rediriger vers la page de login
header("location: connexion.php");
exit;
```

Le programme démarre premièrement la session, je vide toutes les variables de session avec un tableau vide, je détruis la session avec la `session_destroy()`, et je redirige l'utilisateur vers la page de

déconnexion.

5 – Test unitaire

5.1 – Test espace membre/client/administrateur

| | | | | |
|---|---|--|--|---|
| Élément testé : | Espace membre et espace client et espace administrateur | | | |
| Objectif du test : | Création, connexion et modification d'un compte TELELOCATION GEMEB | | | |
| Nom du testeur : | ELEOUET Mattéo | | Date : | 25/05/2021 |
| Moyens mis en œuvre : | Logiciel : Visual Studio Code / FileZilla / Terminal / Navigateur Web | | Matériel : Serveur Ubuntu | Outil de développement : HTML / PHP / SQL |
| Procédure du test : Simuler le chemin que prendra l'utilisateur lors des inscriptions, la connexion ou un administrateur lors de la modification d'un compte TELELOCATION GEMEB | | | | |
| Id | Description du vecteur de test | Résultat attendu | Résultat obtenu | Validation (O/N) |
| 0 | Créer un compte depuis l'IHM d'inscription | Création d'un compte GEMEB effectuer par une instruction SQL : UPDATE et sélection automatique de la machine qui sera loué et hachage du mot de passe | Création de compte possible qui occupe une machine sélectionnée automatiquement | O |
| 1 | Se connecter depuis la page de connexion. | Connexion à partir d'un compte GEMEB avec l'utilisation de la fonction PHP password_verify | Connexion réussit avec un mot de passe haché | O |
| 2 | Modifier son mot d passe Modification du mot de passe pour pour les utilisateurs | Modification du mot de passe utilisateur depuis l'espace client | Modification possible depuis l'espace client | O |
| 3 | Modification des informations d'un compte | Modifier les informations d'un client depuis un espace administrateur. | Modification depuis un formulaire accessible seulement pour | O |
| 4 | Désabonner un client depuis le tableau administrateur. | Le client souhaiter est désabonner, rend disponible sa machine à la location et transfère ses informations (Nom prénom, email etc) dans la table OldClient | Le client est bien désabonné et ses anciennes informations sont stocker dans une table avec les anciens clients. | O |
| Conclusion du test : | | Toutes les fonctionnalités sont fonctionnelles | | |

5.2 – Test de l’affichage des données de la BDD

| | | | | |
|---|--|---|--|--|
| Élément testé : | | Graphique ChartJS, cartes de consommation, tableau des clients. | | |
| Objectif du test : | | Afficher les différentes informations de la base de données | | |
| Nom du testeur : | | ELEOUET Mattéo | | Date : 25/05/2021 |
| Moyens mis en œuvre : | | Logiciel : Visual Studio Code / FileZilla / Terminal / Navigateur Web | | Matériel : Serveur Ubuntu |
| | | | | Outil de développement : HTML / PHP / SQL / JavaScript |
| Procédure du test : Utiliser la PDO pour récupérer les données et les stocker dans une variable PHP pour les afficher avec echo | | | | |
| Id | Description du vecteur de test | Résultat attendu | Résultat obtenu | Validation (O/N) |
| 0 | Utilisation d'un fetch pour parcourir les données d'une table SQL | Stocker une donnée de la base de données dans une variable PHP | Les variables PHP stocke une donnée de la base de données | O |
| 1 | Intégration du code JavaScript dans le code HTML du fichier php pour utiliser le mot-clé echo de PHP | Afficher les informations de la base de données dans du code JavaScript | L'intégration du PHP dans le code JavaScript marche très bien nous voyons la valeur de la variable | O |
| 2 | Intégrer du code PHP dans le code PHP avec < ?php ?> avec les balises d'ouvertures PHP | Afficher les informations de la base de données dans du code HTML | L'intégration du PHP dans le code HTML marche très bien nous voyons la valeur de la variable | O |
| 3 | Utilisation d'une boucle while pour afficher en boucle toutes les informations de tous les clients | Afficher les informations de tous les clients dans un tableau | Le tableau affiche bien toutes les informations des clients. | O |
| Conclusion du test : | | Tous les éléments qui devaient afficher des informations de la base de données affiches ces informations. | | |

6 – Conclusion

Ce projet m'a apporté beaucoup d'un point de vue des connaissances acquise et d'organisation, apprendre à travailler en équipe et communiquer avec mes camarades de projets. J'ai pu mettre en pratique mes connaissances théoriques acquises durant ma formation, de plus, je me suis confronté aux difficultés réelles d'un projet. J'ai dû apprendre que lire cinq minutes de document peut sauver de six heures de débogage (oui oui), j'ai appris à connaître des sites comme GitHub pour visualiser comment les autres ont construit un projet, aller sur les forums comme OpenClassRooms et Stack Overflow pour aller chercher de l'aide, apprendre à demander de l'aide quand je bloque ce que je ne faisais jamais auparavant.

Après une semaine à comprendre comment aller se faire le projet, l'organiser, comprendre les langages de programmation que je pouvais utiliser pour le projet, les frameworks, faire la répartition des tâches avec mes camarades de projet pour le bon fonctionnement du projet.

Le PHP et le SQL ont été les langages que j'ai le plus appris lors du projet, c'est un langage très agréable à utiliser. J'ai appris à utiliser le framework CSS Bootstrap, apprendre à intégrer des bibliothèques à mon projet comme DataTables et ChartJS.

Pour finir je suis content du projet que j'ai obtenu, le projet était bien, il a été enrichissant, il constitue une bonne expérience valorisante et encourageante et me permet de découvrir la partie informatique du développement web, en frontend et backend.

7 - Annexe

Se connecter

Adresse mail

Mot de passe


SE CONNECTER

Mot de passe ou email incorrect


Vous n'êtes pas enregistré ?


INSCRIVEZ-VOUS


GEMEB
TELELOCATION

 Consommation

OPTION SUPPLÉMENTAIRE

 **Changement de MDP**


 Facturation





GEMEB Vous propose
d'acheter la machine et non
de la louer tous les mois !

[Acheter la machine !](#)

OPTION ADMINISTRATEUR

 Tableau clients

 Modification clients

Matteo ROBINO 

Réinitialisation du mot de passe

Veuillez remplir ce formulaire afin de réinitialiser votre mot de passe
Votre mot de passe devra contenir au mini 6 caractères

Nouveau mot de passe

Confirmez le mot de passe

[Valider](#) [Annuler](#)

La Croix Rouge La Salle Projet TELELOCATION 2021

Color Utilities

Bootstrap's default utility classes can be found on the official [Bootstrap Documentation](#) page. The custom utilities below were created to extend this theme past the default utility classes built into Bootstrap's framework.

Custom Text Color Utilities

.text-gray-100

.text-gray-200

.text-gray-300

.text-gray-400

.text-gray-500

.text-gray-600

.text-gray-700

.text-gray-800

.text-gray-900

Custom Font Size Utilities

.text-xs

.text-lg

Custom Background Gradient Utilities

.bg-gradient-primary

.bg-gradient-secondary

.bg-gradient-success

.bg-gradient-info

.bg-gradient-warning

.bg-gradient-danger

.bg-gradient-light

.bg-gradient-dark

Custom Grayscale Background Utilities

.bg-gray-100

.bg-gray-200

.bg-gray-300

.bg-gray-400

.bg-gray-500

.bg-gray-600

.bg-gray-700

.bg-gray-800

.bg-gray-900

Border Utilities

Bootstrap's default utility classes can be found on the official [Bootstrap Documentation](#) page. The custom utilities below were created to extend this theme past the default utility classes built into Bootstrap's framework.

.border-left-primary

.border-bottom-primary

.border-left-secondary

.border-bottom-secondary

.border-left-success

.border-bottom-success

.border-left-info

.border-bottom-info

.border-left-warning

.border-bottom-warning

.border-left-danger

.border-bottom-danger

.border-left-dark

.border-bottom-dark

Other Utilities

Bootstrap's default utility classes can be found on the official [Bootstrap Documentation](#) page. The custom utilities below were created to extend this theme past the default utility classes built into Bootstrap's framework.

Overflow Hidden Utility

Use `.o-hidden` to set the overflow property of any element to hidden.

Progress Small Utility

Normal Progress Bar



Small Progress Bar



Use the `.progress-sm` class along with `.progress`

Dropdown - No Arrow

Dropdown (no arrow)

Add the `.no-arrow` class alongside the `.dropdown`

Rotation Utilities



Buttons

Circle Buttons

Use Font Awesome Icons (included with this theme package) along with the circle buttons as shown in the examples below!

`.btn-circle`



`.btn-circle .btn-sm`



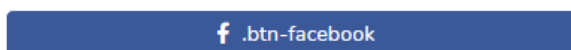
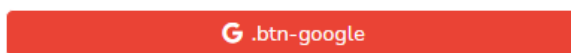
`.btn-circle .btn-lg`



Brand Buttons

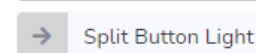
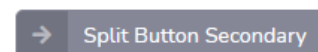
Google and Facebook buttons are available featuring each company's respective brand color. They are used on the user login and registration pages.

You can create more custom buttons by adding a new color variable in the `_variables.scss` file and then using the Bootstrap button variant mixin to create a new style, as demonstrated in the `_buttons.scss` file.

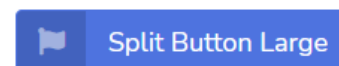
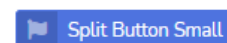


Split Buttons with Icon

Works with any button colors, just use the `.btn-icon-split` class and the markup in the examples below. The examples below also use the `.text-white-50` helper class on the icons for additional styling, but it is not required.



Also works with small and large button classes!



Cards

EARNINGS (MONTHLY)

\$40,000



EARNINGS (ANNUAL)

\$215,000



TASKS

50%



PENDING REQUESTS

18



Default Card Example

This card uses Bootstrap's default styling with no utility classes added. Global styles are the only things modifying the look and feel of this default card example.

Dropdown Card Example



Dropdown menus can be placed in the card header in order to extend the functionality of a basic card. In this dropdown card example, the Font Awesome vertical ellipsis icon in the card header can be clicked on in order to toggle a dropdown menu.

Basic Card Example

The styling for this basic card example is created by using default Bootstrap utility classes. By using utility classes, the style of the card component can be easily modified with no need for any custom CSS!

Collapsible Card Example



This is a collapsible card example using Bootstrap's built in collapse functionality. **Click on the card header** to see the card body collapse and expand!



GEMEB
230, rue de Faubourg-Saint-Honoré
01 45 63 55 31
admin@gemeb.site

FACTURE DE:
ROBINO Matteo
admin@gemeb.site

FACTURE LOCATION

Date de facturation: 2021-06-10

| # | DESCRIPTION | PRIX A L'UNITE | QUANTITE | TOTAL |
|-------------|---|----------------|----------|--------|
| 1 | Consommation de flash Utilisation du nombre de flash de la machine d'épilation | 1.2€ | 180 | 216€ |
| 2 | Frais supplémentaire | 0€ | 0 | 0€ |
| SOUS-TOTAL | | | | 216€ |
| TAXE 20% | | | | 43.2€ |
| GRAND TOTAL | | | | 259.2€ |

Etude d'achat de machine:

Une machine GEMEB d'épilation coûte 20000€ sur une location de 3 ans vous payerai 9331.2€, nous vous conseillons de continuer à louer notre machine.

Merci !

NOTICE:

Des frais financiers de 1,5 % seront appliqués aux soldes impayés après 30 jours.

La facture a été créée sur un ordinateur et est valable sans la signature et le cachet.

Tableau des clients

Table de données des utilisateurs inscrits.

Montrer 10 clients

Chercher :

| Num client | Nom | Email | Telephone | Adresse | date de création | Nombre flash |
|------------|-----------------|----------------------------|----------------|---|------------------|--------------|
| 1 | hack hack | hack@hack.hack | 667 | 130 | 2021-05-08 | 1494 |
| 10 | Renaud Agate | AgateRenaud@teleworm.us | 01.82.03.41.19 | | 2021-04-11 | 500 |
| 11 | Leduc Odelette | OdeletteLeduc@armyspyf.com | 01.55.41.79.26 | 5 boulevard | 2021-04-11 | 0 |
| 12 | Bizier Benoit | BenoitBizier@dayrep.com | 04.76.03.93.22 | 5, Avenue De Marlioz 74100 ANNEMASSE | 2021-04-11 | 0 |
| 13 | Macron Emmanuel | emmanuel.macron@wanadoo.fr | 0667438362 | Rue des rois | 2021-05-23 | 0 |
| 14 | Ancelote Blais | AnceloteBlais@teleworm.us | 02.17.86.34.37 | 47, boulevard Aristide Briand 76120 LE GRAND-QUEVILLY | 2021-04-11 | 100 |
| 15 | Blanquer Michel | micheLblanquer@gouv.fr | 0765453412 | 2 rue de Rivolie Paris 75000 | 2021-04-13 | 0 |
| 16 | Acer Crhstian | acer.sony@wanadoo.fr | 0304058354 | 34 rue des Camarades brules | 2021-04-14 | 0 |
| 17 | Evra Patrice | f@f.f | 0604653465 | 52 rues des bons amis | 2021-04-14 | 0 |
| 18 | Robino Pierre | pierreroBino@gmail.com | 0632456599 | 33 rue des forces Rennes 35000 | 2021-05-16 | 0 |
| Num client | Nom | Email | Telephone | Adresse | date de création | Nombre flash |

Affichage de 1 à 10 sur 21 clients

Précédent 1 2 3 Suivant

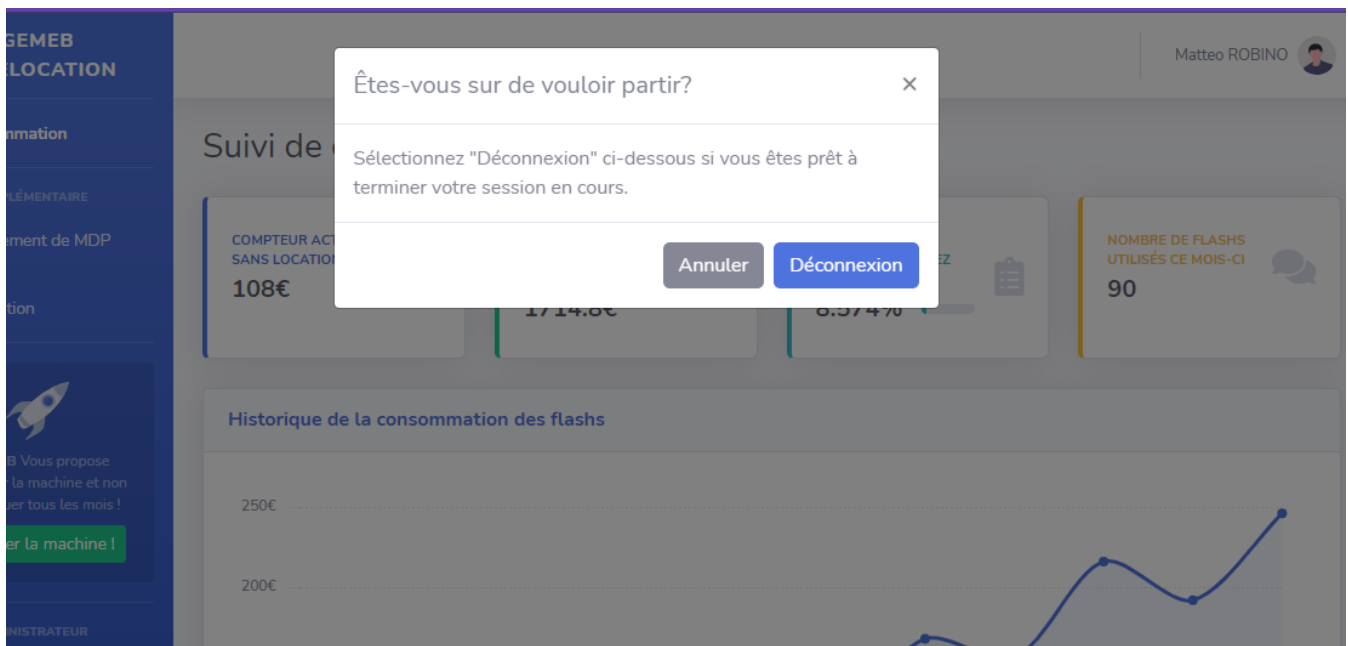
+ Ajouter une machine

POURCENTAGE DES MACHINE UTILISER :

50%



```
1  <?php // Programme ajout-machine
2
3  $host = 'mysql:host=localhost;dbname=test2';
4  $dbUser = 'root';
5  $dbMdp = 'willy9105';
6
7  $PDO = new PDO($host, $dbUser, $dbMdp);
8
9  $sql = 'SELECT Client FROM Prestation ORDER BY Client DESC LIMIT 1';
10 $results = $PDO->query($sql); // 8
11 $dataMAX = $results->fetch(PDO::FETCH_ASSOC);
12 $clientMAX = $dataMAX['Client'];
13 $numClient = $clientMAX + 1;
14 $nomMachine = "ENS$numClient";
15
16 $sqlRegister = "INSERT INTO registers (Nom) VALUES ('DISPO')";
17 $sqlPrestation = "INSERT INTO Prestation (NomMachine, Client)
18 | VALUES ('$nomMachine', '$numClient')";
19 $sqlDate1 = "INSERT INTO date1 (idClient, Client)
20 VALUES ('$numClient', '$numClient')";
21
22 $PDO->query($sqlRegister);
23 $PDO->query($sqlPrestation);
24 $PDO->query($sqlDate1);
25 header("location: ../tableau-user.php");
```





Philosophie

Le centre GEMEB est situé au cœur du 8ème arrondissement de Paris. Dédié au mieux-être et à la prévention, il réunit des compétences médicales, techniques et esthétiques permettant de répondre à vos besoins.

GEMEB tient compte de l'aspect humain de la prise en charge, le corps étant conçu comme un tout et non une succession d'organes. C'est pourquoi, avant tout acte, nous avons mis au point un protocole d'expertises (questionnaires, entretiens, analyses sanguines afin d'identifier les carences spécifiques ou les

