



Patrimoine Informatique MarieTeam

Alouache Yannis

Norme de programmation

- Suivre le modèle MVC (Model-View-Controller) : s'assurer que la logique d'affichage d'une page, la logique métier et la logique d'accès aux données sont bien séparées.
- Écrire un code lisible : utiliser une indentation et une mise en forme appropriées pour faciliter la lecture du code et son maintien.
- Commenter le code : ajouter des commentaires pour expliquer le but et le fonctionnement de chaque fonction ou méthode.
- Utiliser des fonctions et des méthodes modulaires : diviser le code en fonctions et en méthodes réutilisables pour faciliter la maintenance.
- Éviter les noms de variables trop courts ou trop longs : utiliser des noms de variables clairs et concis.
- Éviter la duplication de code : ne copier pas et ne coller pas de code similaire plusieurs fois, créez des fonctions ou des méthodes réutilisables.
- Effectuer des tests unitaires : s'assurer que chaque fonction ou méthode est testée et fonctionne comme prévu.

- Pour les fonction et procedure : première lettre du premier mot en minuscule ensuite première lettre des autres mots en Majuscule

```
function getUtilisateurById($id) {  
    ...  
}
```

- Pour les classes : Première lettre en majuscule

```
class MaSuperClasse {  
    ...  
}
```

- Pour les variables : première lettre du premier mot en minuscule ensuite première lettre des autres mots en Majuscule

```
$idClient = ...
```

Mise en place d'utilisateur BDD MySQL.

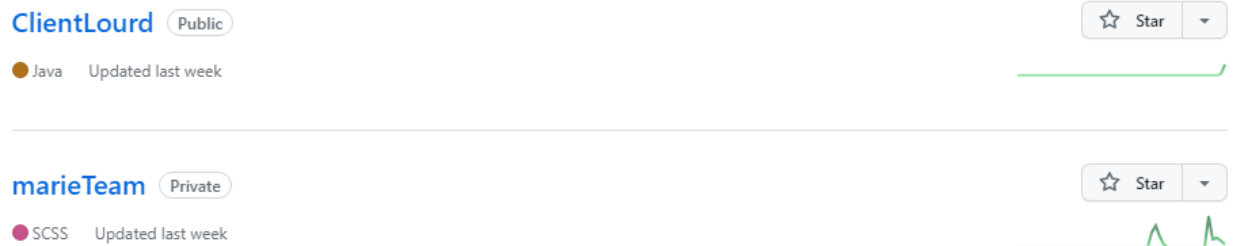
Pour éviter de se connecter à la base de donnée en root ce qui est déconseillé j'ai créé 3 utilisateurs pour les membres du projet et leur ai affecter des droits.

Pour la création de mot de passe fort, j'ai utilisé l'outil fourni par avast.

```
CREATE USER 'yannis'@'localhost' IDENTIFIED BY '^[2$RMg$#S';  
CREATE USER 'redouane'@'localhost' IDENTIFIED BY '7HhXboxT.F';  
CREATE USER 'jawad'@'localhost' IDENTIFIED BY 'ibX]mDv&(8';  
  
GRANT ALL PRIVILEGES ON *.* TO 'yannis'@'localhost' WITH GRANT OPTION;  
GRANT ALL PRIVILEGES ON *.* TO 'redouane'@'localhost' WITH GRANT OPTION;  
GRANT ALL PRIVILEGES ON *.* TO 'jawad'@'localhost' WITH GRANT OPTION;
```

Procédure d'utilisation d'une solution de gestion de version : Github

1) Créer un dépôt pour le projet : une fois connecté à GitHub, cliquer sur le bouton "New" pour créer un nouveau dépôt. Donner un nom au dépôt (Dans notre cas MarieTeam et Client Lourd).



2) Initialiser le dépôt avec les fichiers du projet : ouvrir une invite de commande et naviguer jusqu'au répertoire contenant les fichiers du projet MarieTeam. Puis entrer la commande suivante pour initialiser le dépôt Git :

```
git init
```

3) Ajouter les fichiers au dépôt : utiliser la commande git add pour ajouter les fichiers de votre projet au dépôt. Par exemple, pour ajouter tous les fichiers du projet MarieTeam, entrer la commande suivante :

```
git add .
```


4) Créer un commit : une fois que tous les fichiers sont ajoutés, utiliser la commande git commit pour créer un commit qui représente l'état actuel de votre projet. Assurez-vous d'ajouter un message clair qui décrit les modifications apportées au projet.

```
git commit -m "Ajout du module de Reservation"
```

5) Ajouter le dépôt distant : pour synchroniser le dépôt local avec le dépôt GitHub, utilisez la commande git remote add suivi du lien du dépôt.

```
git remote add origin https://github.com/Yannis-Alouache/marieTeam
```

6) Pousser les modifications vers GitHub : utilisez la commande `git push` pour pousser les modifications vers votre dépôt distant.



```
git push -u origin master
```

7) Mettre à jour le dépôt : pour apporter des modifications au projet MarieTeam, utiliser les commandes `git add` et `git commit` pour ajouter et enregistrer ces modifications dans votre dépôt local. Une fois que je suis satisfait des modifications, j'utilise la commande `git push` pour mettre à jour le dépôt distant sur GitHub.



```
git add .  
git commit -m "Ajout de fonctionnalité à MarieTeam"  
git push
```

Les Normes pour les message de commit

Pour les messages de commit j'ai utilisé une norme qui permet de se repérer facilement et de suivre efficacement les modifications apporté au projet.

Pour les ajouts de code :

```
git add .  
git commit -m "[+] added reservation module"  
git push
```














Pour les suppression de code :

```
git add .  
git commit -m "[-] removed files"  
git push
```

Pour les modification de code :

```
git add .  
git commit -m "[~] modified reservation module"  
git push
```


Respect des normes Github

Commits on Feb 26, 2023		
[+] added selectLiaison vue Yannis-Alouache committed on Feb 26	 9ccea73	 Don
[+] added form to selectLiaison Yannis-Alouache committed on Feb 26	 58dc3ef	 Don
[+] added selectLiaison Yannis-Alouache committed on Feb 26	 cca535d	 Don
[+] added selectLiaison to controleur Yannis-Alouache committed on Feb 26	 e76fe7b	 Don
[-] removed liaison Yannis-Alouache committed on Feb 26	 3b8c24c	 Don
[+] get_liaisons_by_sector JawadBLK committed on Feb 26	 d71bd15	 Don
[~] fixed booking form Yannis-Alouache committed on Feb 26	 d17bd0f	 Don


j'ai réussi à respecter les normes établies pour les commits sur GitHub.

J'ai pris soin d'écrire des messages de commit concis et précis, en fournissant des informations utiles sur les changements que j'ai apportés au code.

En utilisant une syntaxe cohérente et en suivant les conventions établies pour la structure des messages de commit, j'ai pu rendre le processus de collaboration plus facile pour moi-même et pour mes collègues de travail.

J'espère continuer à suivre les normes établies pour les commits de mes projets à l'avenir.

Respect des normes de Programmation


Yannis-Alouache [-] removed all useless file
 3fb3ad1 25 minutes ago
🕒 81 commits

📁 controleur	[-] removed all useless file	25 minutes ago
📁 css	[+]	3 weeks ago
📁 img	[+]	3 weeks ago
📁 js	[+] add header, style.css, folder js and bootstrap and others and resp...	5 months ago
📁 modele	[-] removed all useless file	25 minutes ago
📁 vue	[-] removed all useless file	25 minutes ago
📄 .DS_Store	[+]	5 months ago
📄 getRacine.php	[+]	5 months ago
📄 index.php	[+] add header, style.css, folder js and bootstrap and others and resp...	5 months ago

Help people interested in this repository understand your project by adding a README.
 [Add a README](#)

```
function getBoatByName($name) {
    $boat = null;

    try {
        $connexion = connexionPDO();
        $query = "SELECT * FROM `bateau` where bateau.nom = ?";
        $stmt = $connexion->prepare($query);

        $stmt->execute([$name]);
        $boat = $stmt->fetch(PDO::FETCH_ASSOC);
    } catch (PDOException $e) {
        print "Erreur !: " . $e->getMessage();
        die();
    }

    return $boat;
}
```

J'ai réussi à garder un environnement de code sain en respectant les normes de programmation que j'avais prévu.