

Site Web de QCM

- Rapport de peojet -

Mohamed MIRI | WEB | 28 juillet 2020

Table de matière :

1. Introduction …………………………………………………………………………………………………………. 2

2. Description du projet …………………………………………………………………………………………… 3  
 2.1 Exigences fonctionnelles …………………………………………………………………………………. 3  
 2.2 Exigences non fonctionnelles …………………………………………………………………………. 3  
 2.3 Exigence de performance ……………………………………………………………………………….. 4

3. Analyse et conception …………………………………………………………………………………………… 5  
 3.1 Diagramme de cas d’utilisation ………………………………………………………………………. 5  
 3.2 Diagramme de classe ………………………………………………………………………………………. 7

4. Réalisation ………………………….………………………….………………………….………………………….. 8  
 4.1 Les langages utilisés ………………………….………………………….………………………………….. 8  
 4.2 Implémentation ………………………….………………………….……………………………………….. 9  
 4.3 Exemple ………………………….………………………….………………………….………………………… 9

5. Conclusion …………………………………………………………………………………………………………….. 28

1. Introduction

Le but de ce projet est d’appliquer ce qu’on a appris dans le module du Développement web, et d’implémenter le HTML, le CSS et le PHP.

On va réaliser un site web qui facilite la tâche des qcm, un professeur qui créer un qcm et l’étudiant qui répond, et le site web qui est l’intermédiaire et qui correcte le qcm.

1. Description du projet

2.1 Exigences fonctionnelles :

* Inscription :

Tout visiteur du site il peut créer un compte, il saisit ses informations personnelles, afin de devenir membre.

* Authentification :

Afin d’accéder au site et pouvoir accéder aux différentes fonctionnalités tout membre doit s’authentifier.

* QCM :

Un membre de type professeur peut créer un qcm avec sa correction et le publier.

Un membre de type étudiant, il peut sélectionner un QCM disponible et répondre à ses questions.

* Les réponses :

Un membre de type professeur peut visualiser les notes des étudiant qui ont participé au QCM après sa fin.

Un membre de type étudiant il peut visualiser sa note et son classement et la moyenne des participants du QCM qu’il vient juste de le passer.

* Modifier :

Un membre de type professeur, il peut modifier un QCM après l’avoir validé.

Le nombre de champs à modifier varie selon l’état du QCM (En cours, Pas encore).

2.2 Exigences non fonctionnelles :

* Exigence de qualité :

Afin que ce site donne envie aux étudiants et aux professeurs de rester dessus et bien passer leur QCM, il est important de répondre aux exigences de qualité suivantes :

* La facilité d’utilisation :

La mise en page du site facilite la navigation et la compréhension du site et ses fonctionnalités.

* L’interface graphique :

Les différentes couleurs et choix typographiques doivent permettent à un utilisateur de repérer les différentes fonctionnalités qui s’offrent à lui.

* Exigence de performance :

Le site est codé d’une façon simple, ce qui facilite le flux d’information, ce que veut dire une navigation rapide.

1. Analyse et conception

Dans cette partie, nous utilisons la modélisation UML pour représenter les spécifications des exigences grâce au diagramme de cas d’utilisation, mais aussi pour analyser le domaine avec le diagramme de classe.

3.1 Diagramme de cas d’utilisation :

Nous allons répondre aux questions suivantes : Quels sont les utilisateurs du système ? Quelles sont leurs interactions avec celui-ci ? Il faut donc identifier les différents acteurs ainsi que les cas d’utilisation c’est-à-dire les différentes fonctionnalités du système.

Les acteurs sont :

-**Le visiteur** : personne qui n’est pas membre du site et qui souhaite le devenir.

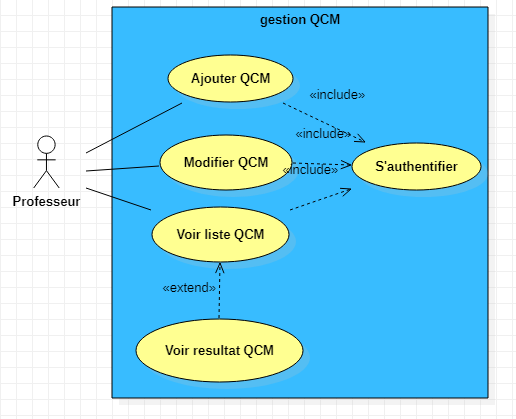
-**Professeur/Etudiant** : personne qui possède un compte, et chacun à des fonctionnalités à pouvoir faire.

Figure 1 : usecase Professeur

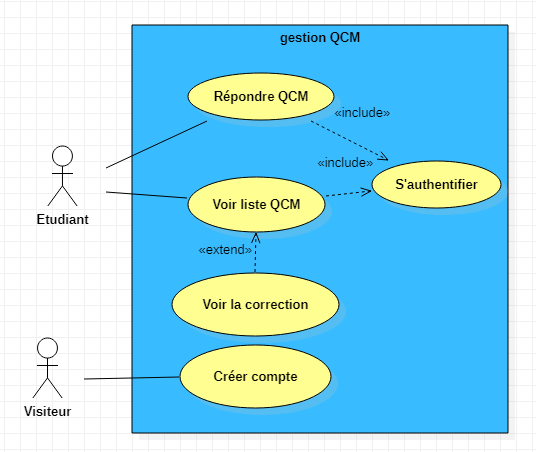
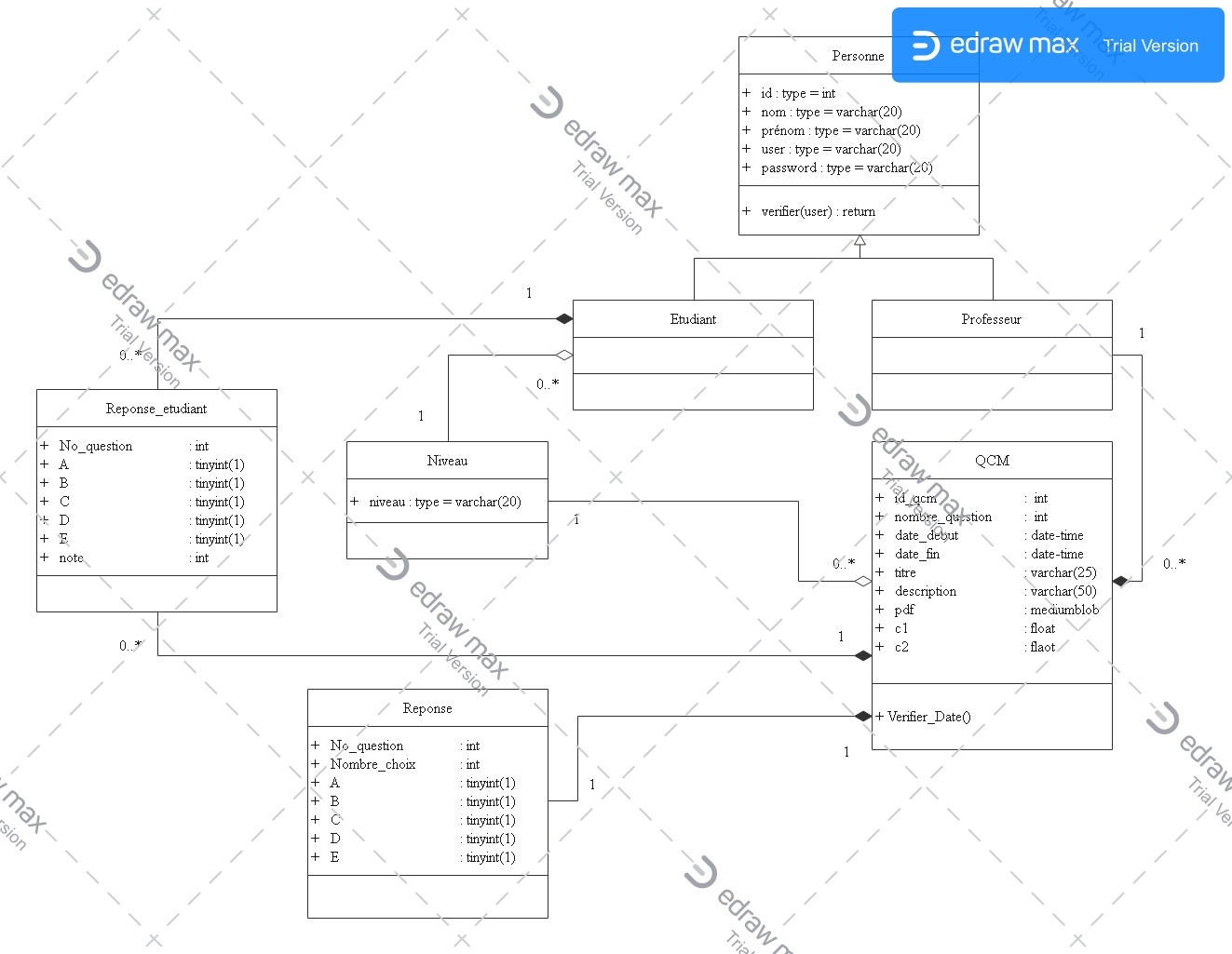


Figure 2 : usecase Etudiant et Visiteur

3.2 Diagramme de classe :



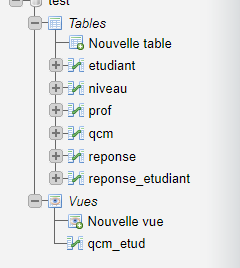
La méthode verifier(user) dans la classe personne, vérifie si un certain user est déjà utiliser.

La méthode Verifier\_date() dans la classe QCM, compare la date fin et la date début à la date du jour.

Apres l’implémentation de la base de données, j’ai ajouté une VUE :

CREATE VIEW qcm\_etud AS  
SELECT id\_qcm, id\_etudiant, sum(note) as notes  
FROM reponse\_etudiant  
GROUP BY id\_qcm, id\_etudiant

Qu’elle va m’aider à calculer la moyenne générale d’un étudiant et son classement.

La base de données ressemble à ça :

1. Réalisation :
   1. Les langages utilisés :

* HTML 5 : Pour la structure et la mise en page du Site Web.
* CSS :

Pour bien décrire les documents HTML et pour alléger le code source écrit en   
 HTML, puisque tout ce qui est relatif à la présentation est géré dans un fichier séparé

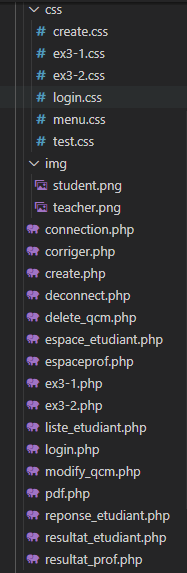
* PHP :

Pour le coté serveur, un des langages coté serveur le plus utilisé.

* JAVASCRIPT:

Pour un petit passage de l’implémentation de l’horloge.

* 1. Implémentation :

L’implémentation est répartie comme suivant.

Tous les fichiers PHP son visualiser, appart :

* Connection.php : qui assure la connexion avec la base de données, et il est appelée à chaque fichier PHP.
* Deconnect.php : assure la déconnexion de la base de données, ce fichier est exécuté quand l’utilisateur se déconnecte.
* Delete\_qcm.php : quand ce fichier est exécuté, il exécute une requête puis il redirige vers espaceprof.php.
  1. Exemple :

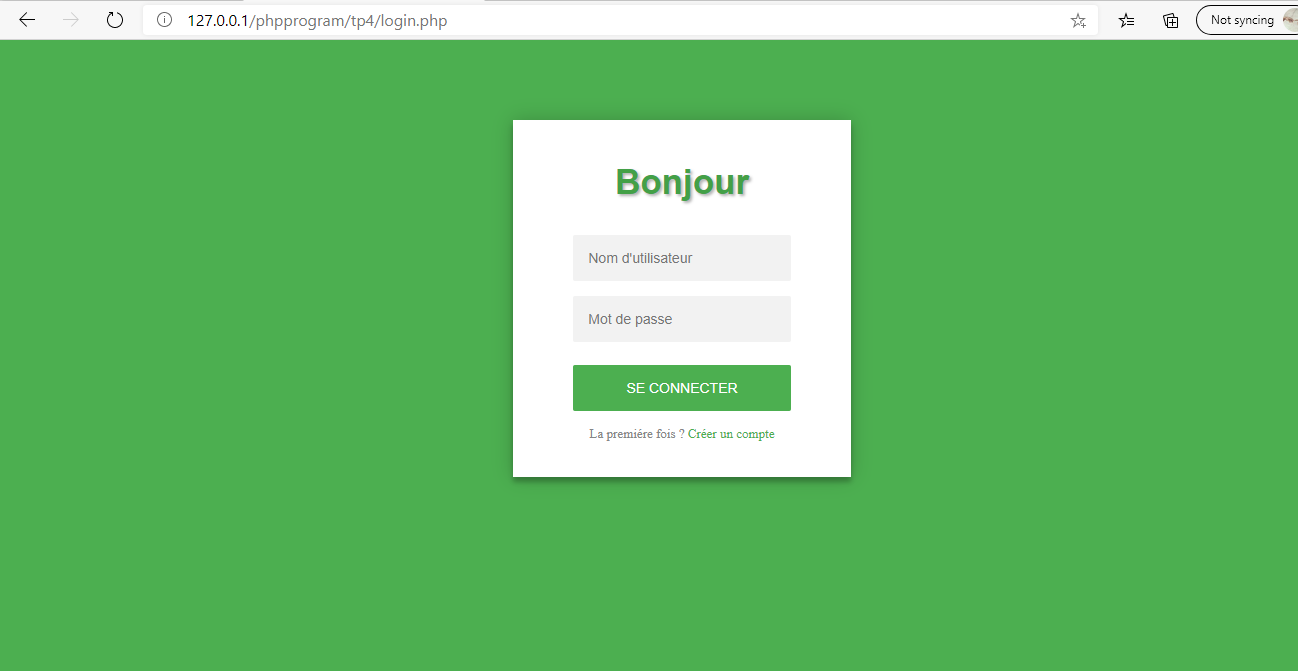
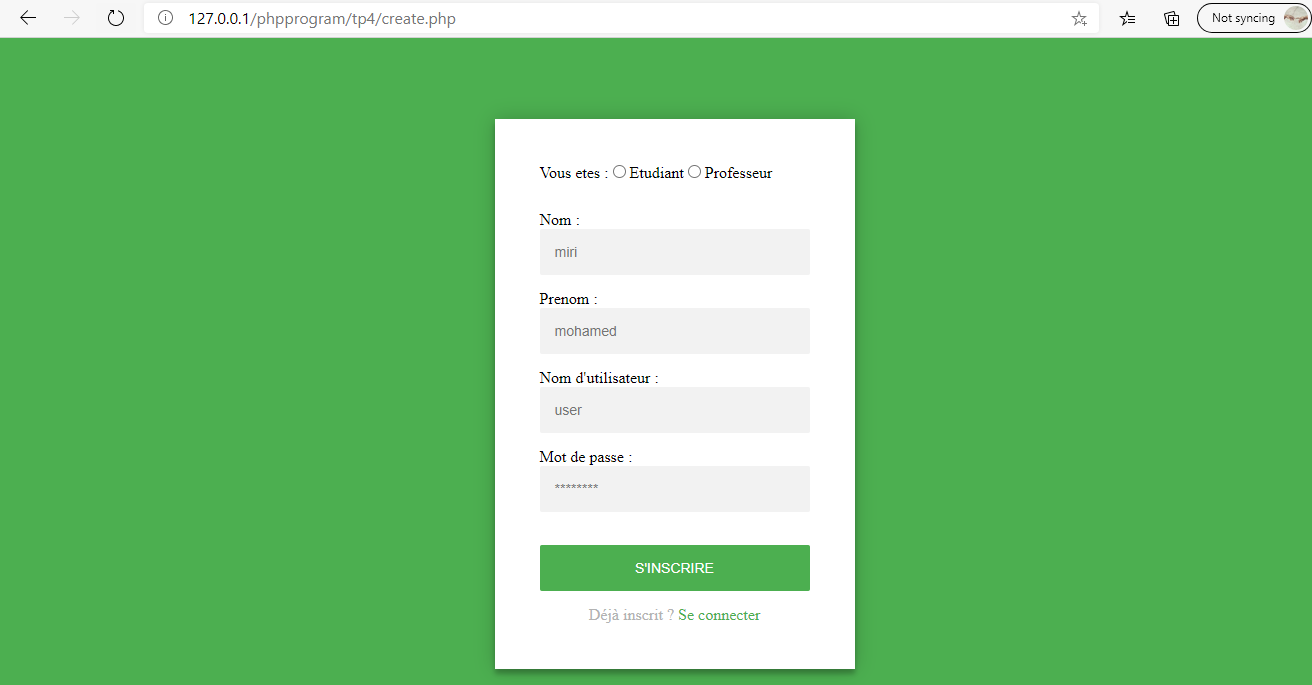
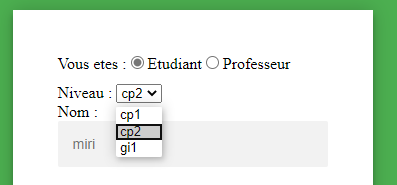
Visiteur :

Figure 3 : page login

On clique sur Créer un Compte :

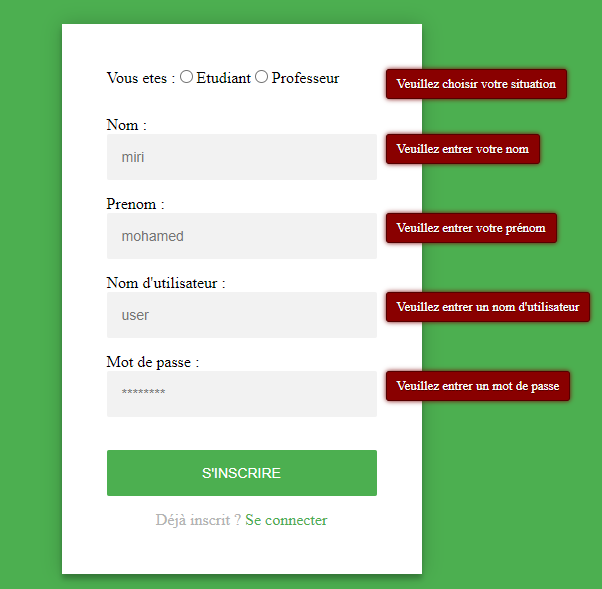


Pour le radio, si on choisit Etudiant, un combo box s’affichera :

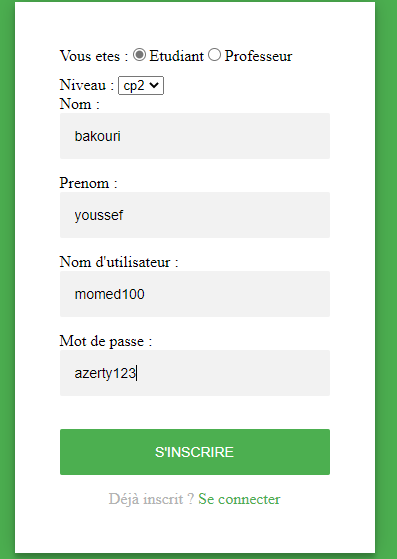
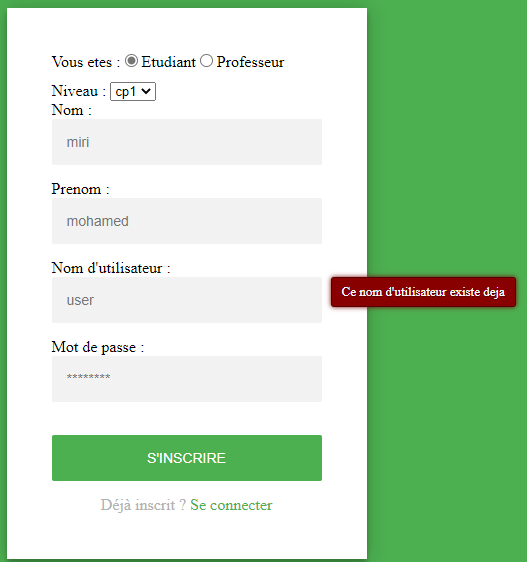


Pour ajouter d’autre niveau et filière il suffit de les ajouter manuellement à la base de données, et ils seront automatiquement ajouter dans cette combo box.

On remplit les champs, si on laisse un champ vide, une erreur s’affiche :

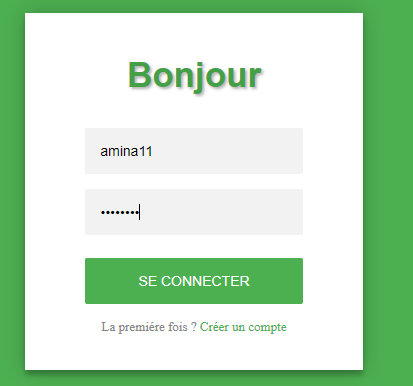


La même chose si on saisit un nom d’utilisateur déjà utiliser : (momed100)



Quand on arrive à créer un compte, on sera redirigé vers la page login.

**Professeur :**



Quand on saisit le login et le mot de passe, le site détecte

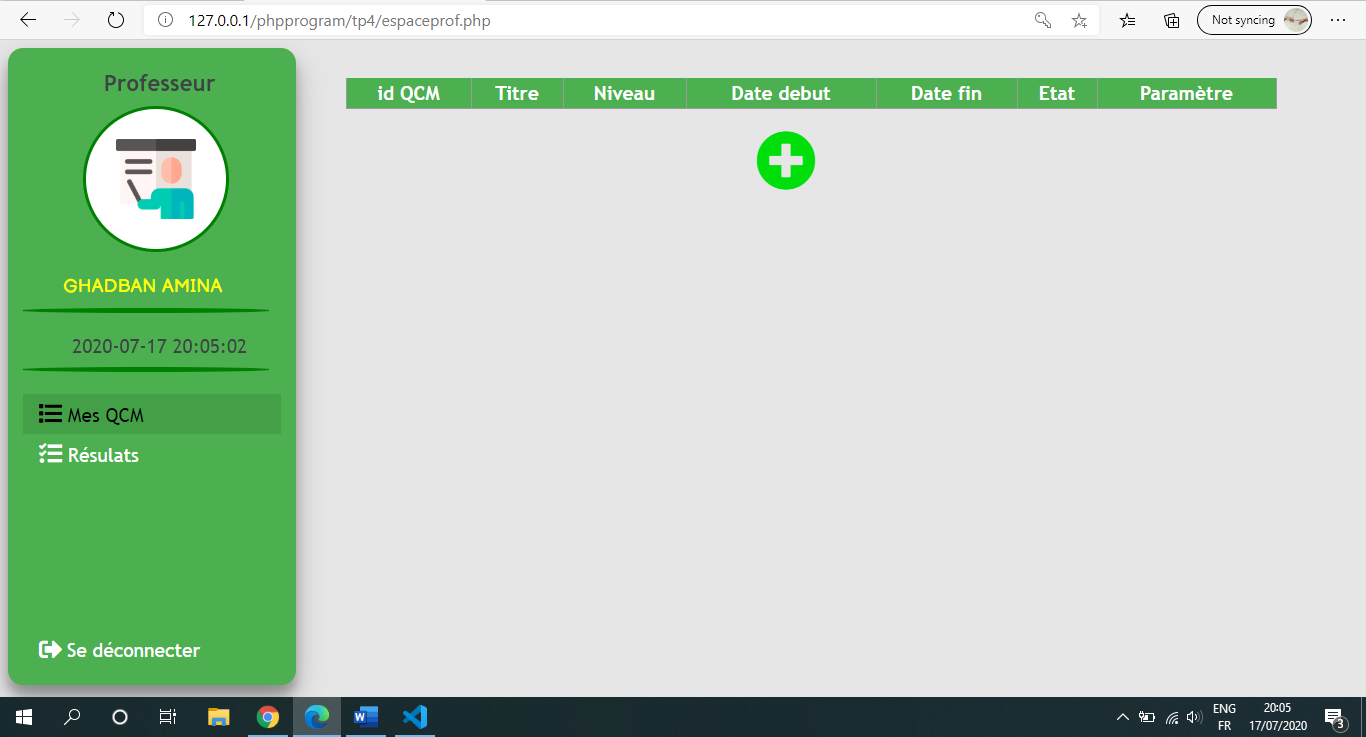
automatiquement le type du compte, professeur ou étudiant.

(User : amina11 , password : useruser . Et de type professeur)

Cette interface détecte si un champ est vide ou bien si les informations entrer sont incorrecte :



Si toutes les informations sont correctes, l’interface suivante sera comme suivant :

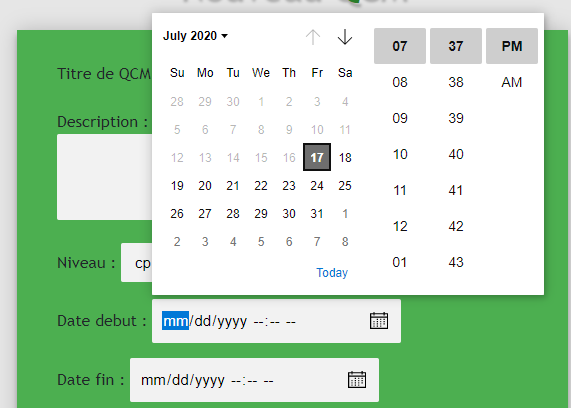


La partie gauche se répète dans presque tous les pages.

On clique sur (+) pour ajouter un QCM :



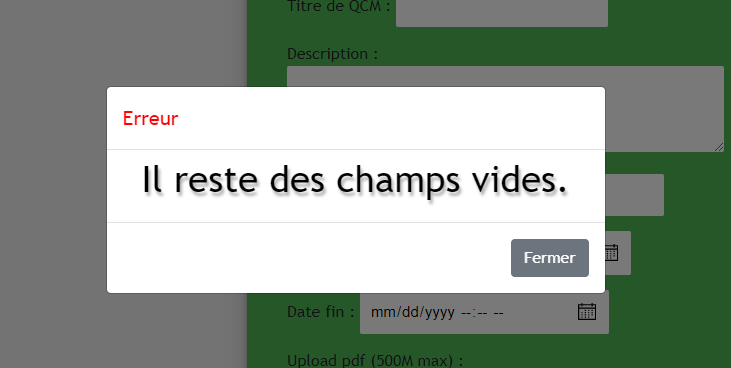
On remplit les champs, la date min des dates est celle du même jour, par exemple nous sommes le 17/07 :



Et on peut avoir des erreurs dans deux cas :

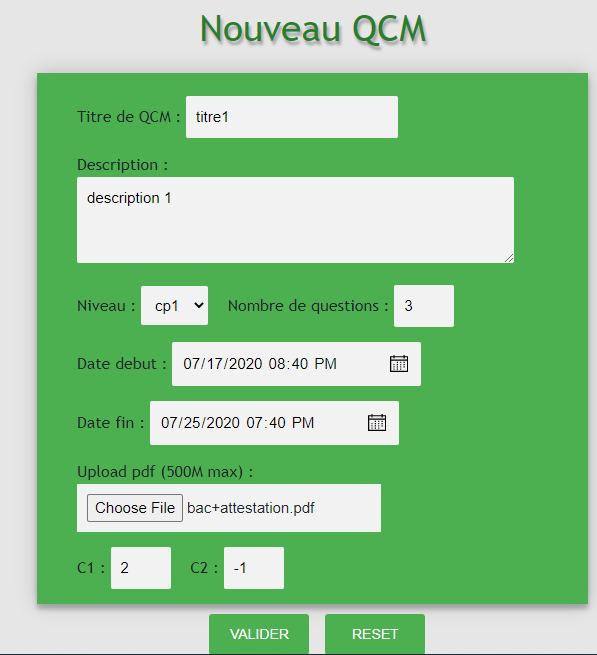
* Si on laisse un champ vide.
* Si la date fin et avant la date de début.

Exemple si on valide sans remplir tous les champs :

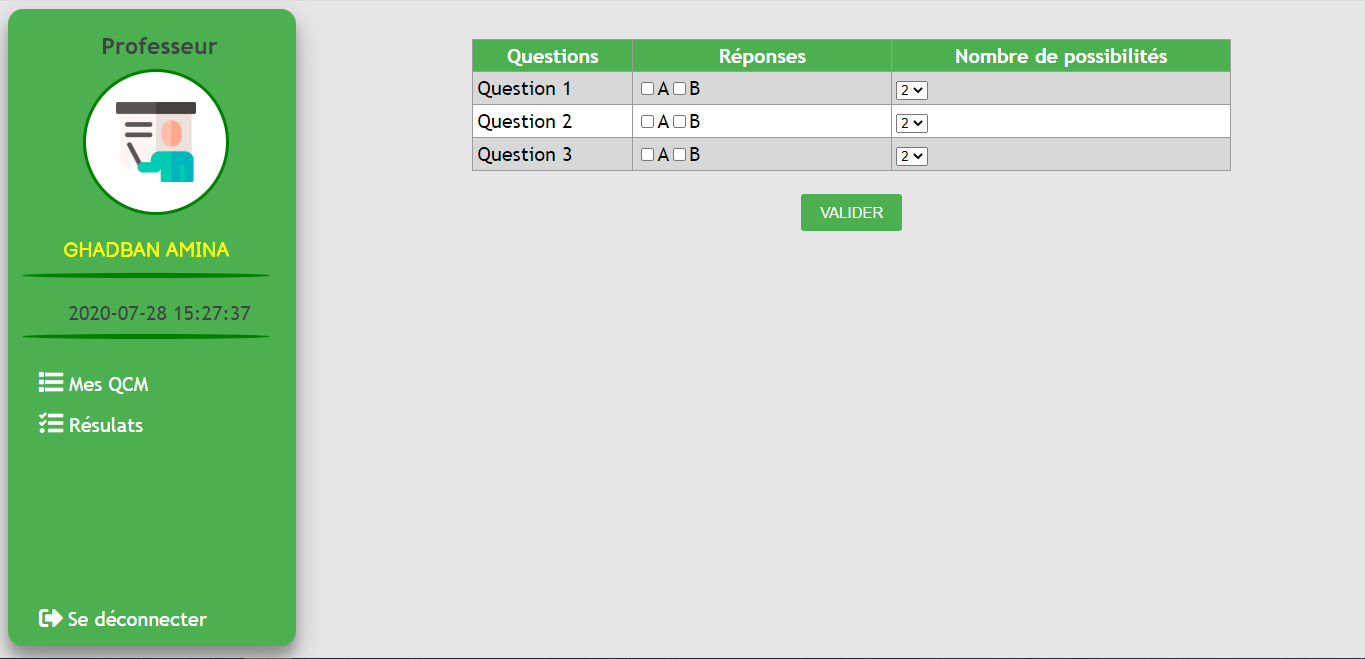


Ce modal est réaliser avec Bootstrap.

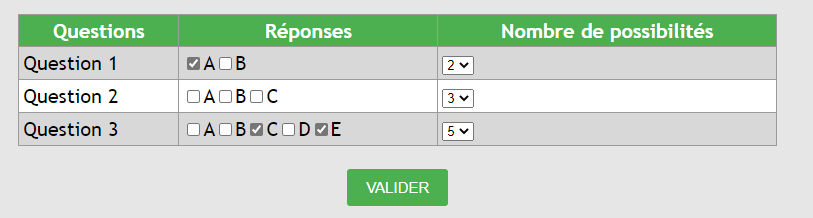
On crée un QCM de trois questions :



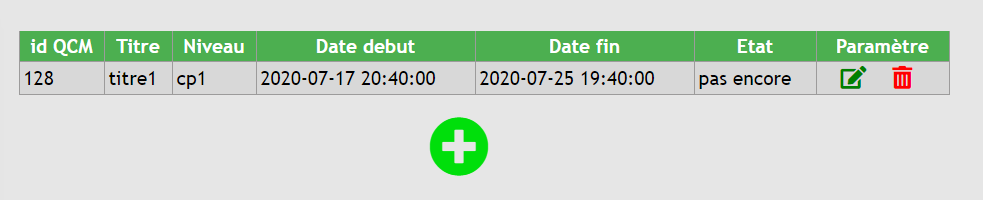
Puis on valide, on se retrouve sur la page suivante :



Avec 3 questions comme prévue, et la possibilité de choisir entre 2 à 5 réponses par question.



Quand on valide, on revient automatiquement à la page « mon espace » :

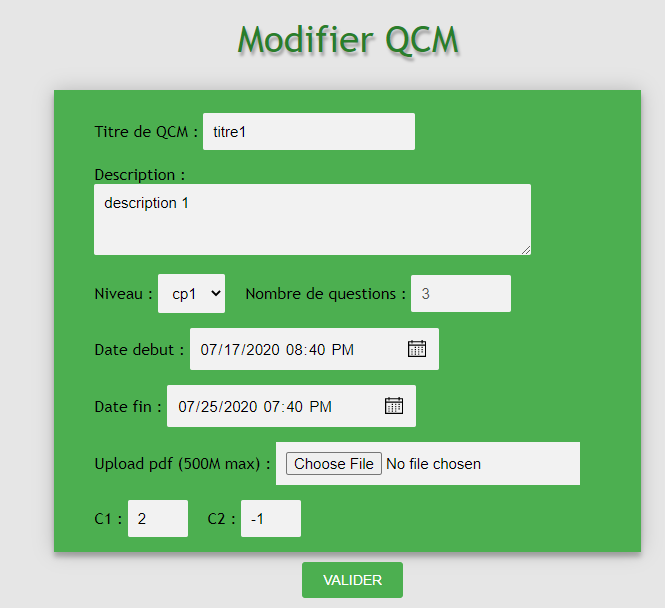


Cette fois le qcm ajouter se voit, avec un état de pas encore, càd pas encore commencer.

Les lignes de cette table seront triées par la date de création du QCM.

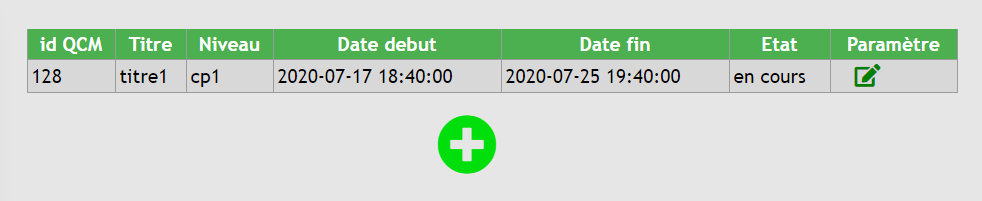
L’ID s’incrémente automatiquement, dans ce cas égal à 128 car j’ai déjà des exemples au paravent.

Et comme on voit à droit, on peut supprimer le QCM ou bien le modifier.

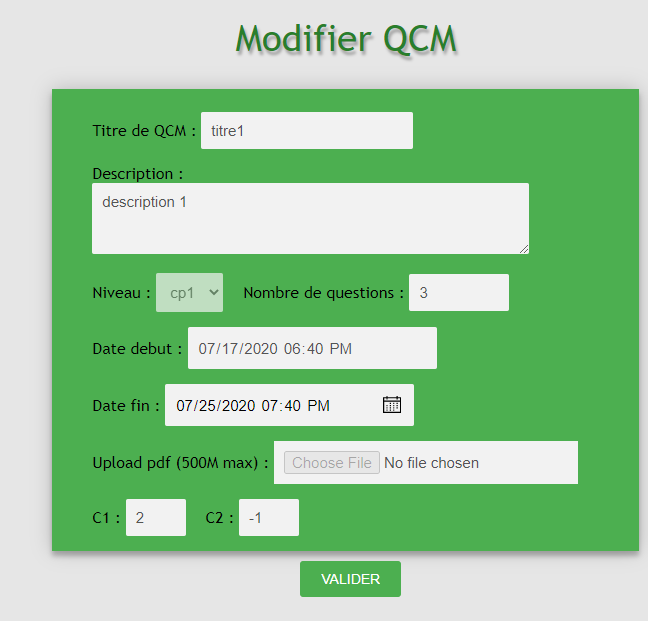
On peut tous modifier appart le nombre de question et leurs réponses :

* Quand on accès à cette page, on trouve les informations du QCM avec le champ « nombre de question » inaccessible (disabled).

Quand la date début arrive, l’interface change :



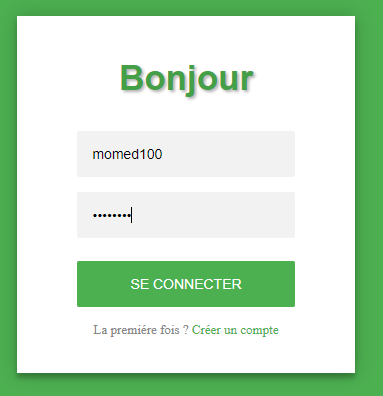
On remarque que le QCM est en cours et on ne peut plus le supprimer. Pour la modification, on peut juste modifier la date fin :



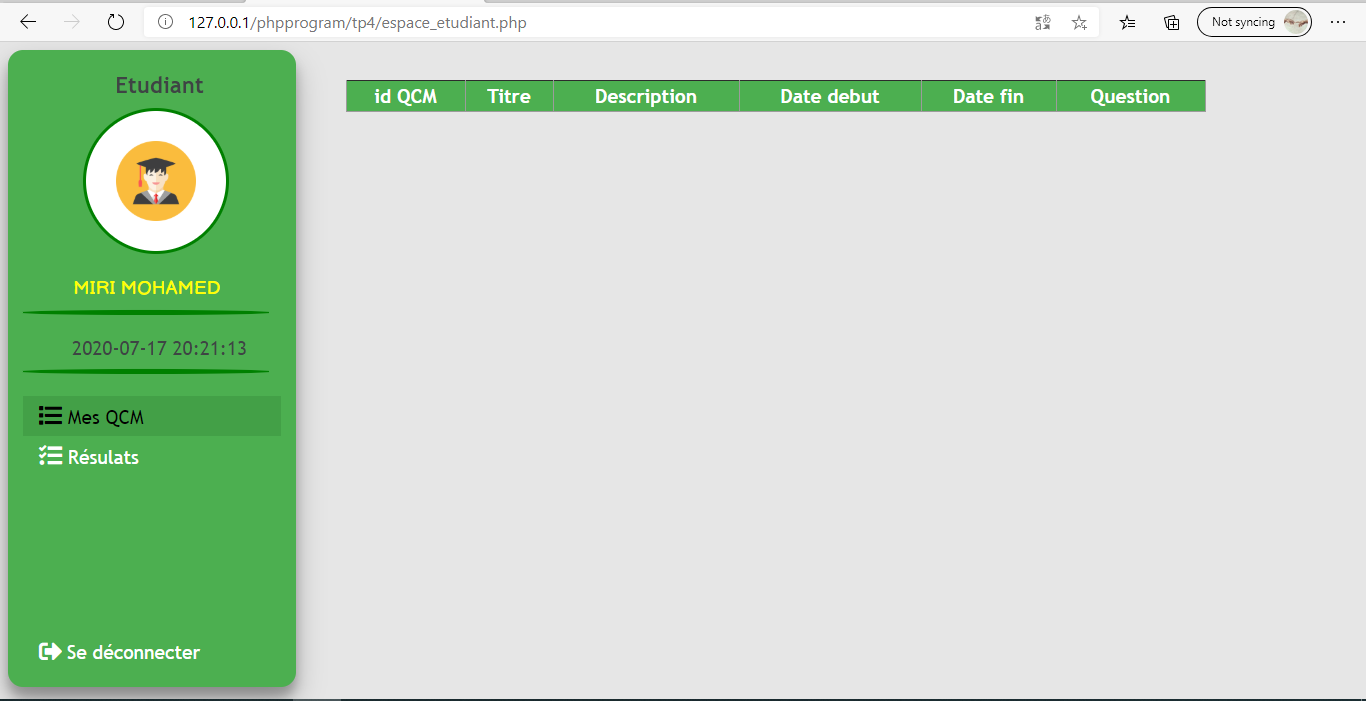
* On voit que tous les champs sont inaccessible appart date fin.

Maintenant on bascule au côté étudiant.

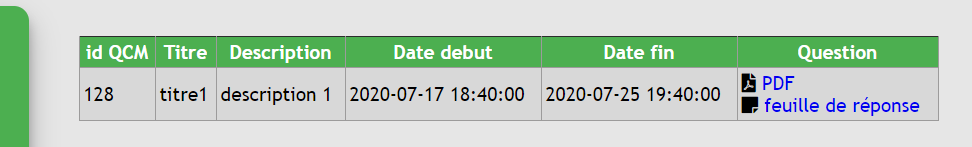
**Etudiant :**

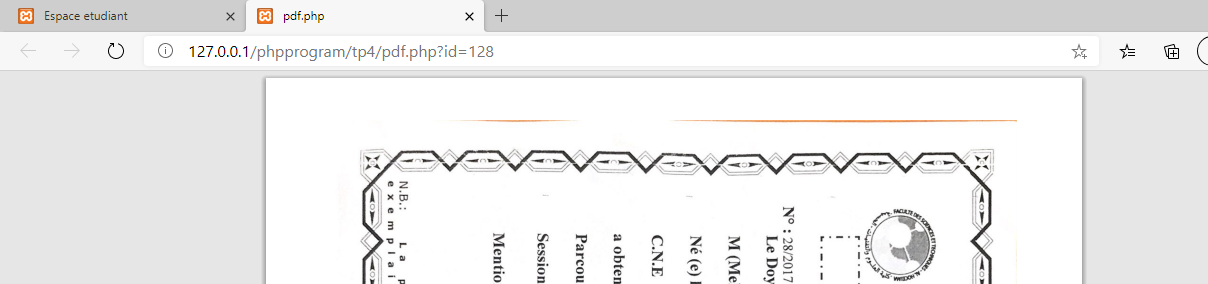
* Après la création d’un compte de type étudiant de CP1, on se connecte.

L’espace étudiant :

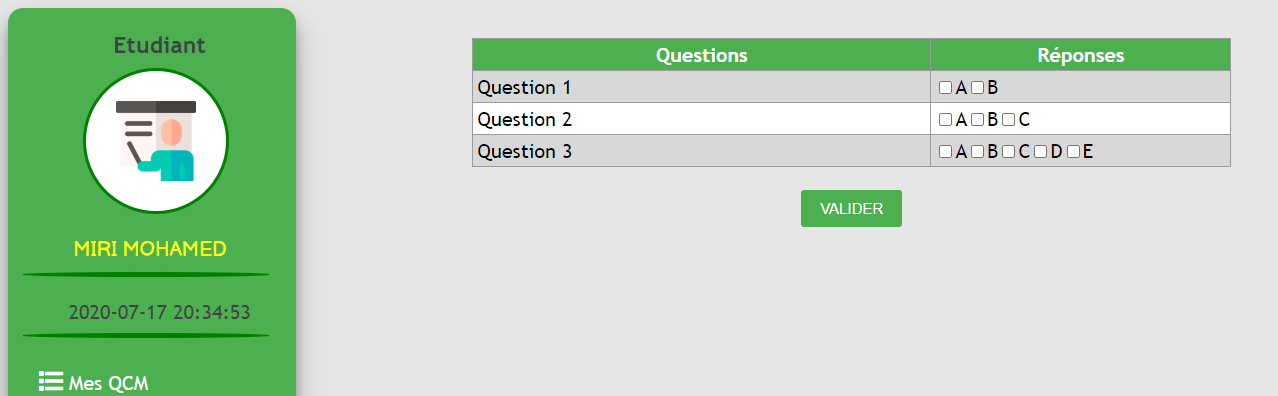


Quand un QCM commence, on le verra : (seul les étudiant du CP1)



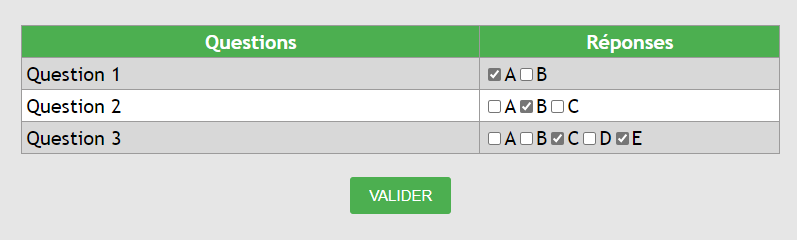
* Quand on clique dur PDF, une page s’ajoute (target : blank), avec le PDF dedans.

* Quand on clique sur feuille de réponse la page change (target : self)

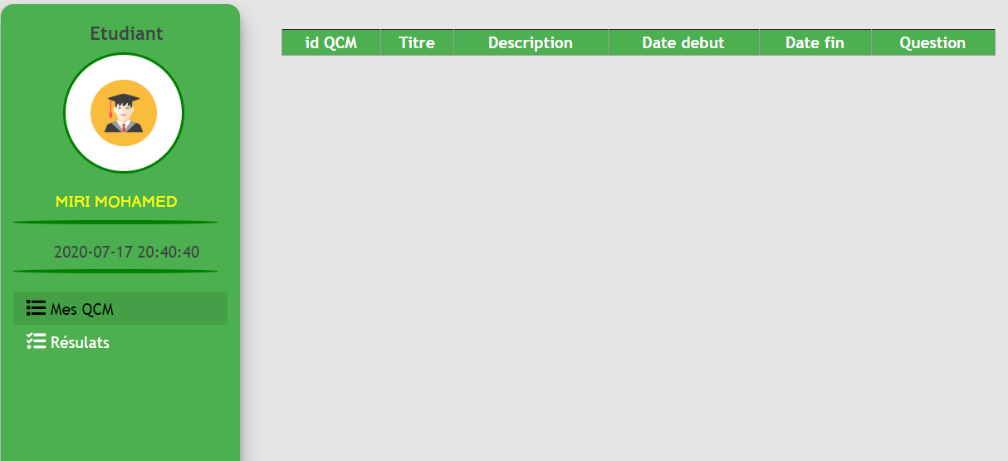


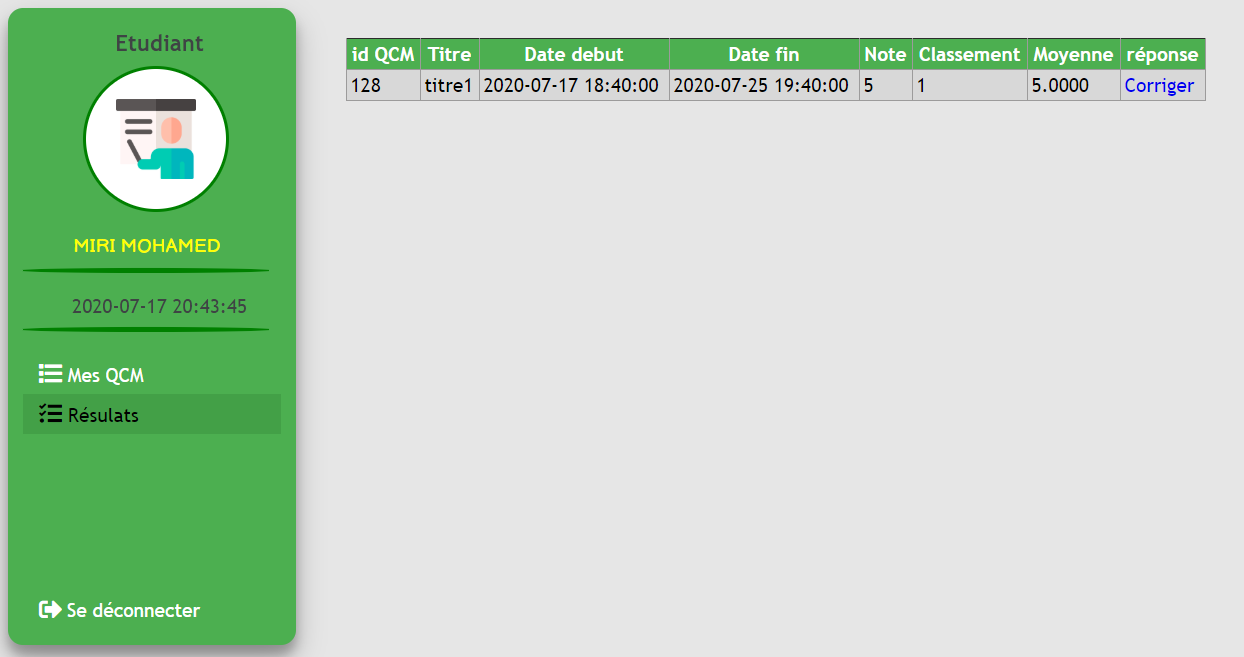
La feuille est comme le professeur à programmer.

On coche les réponses voulues et on valide :



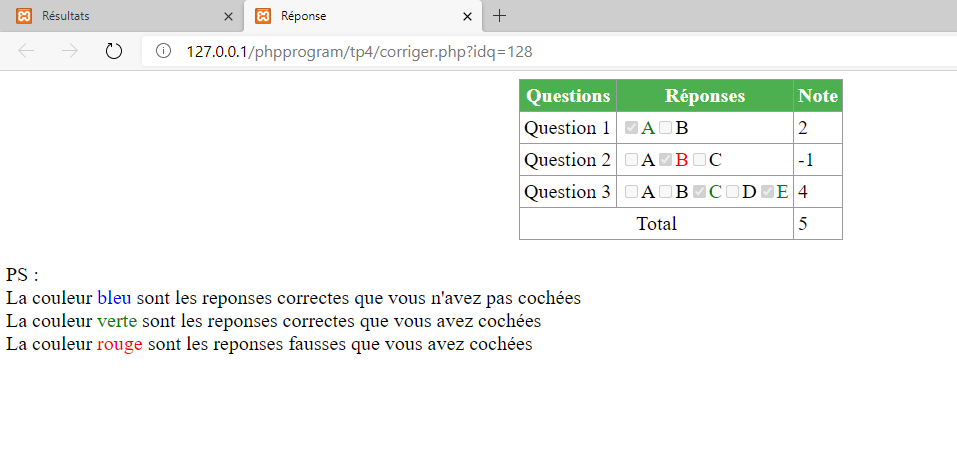
On sera redirigé vers la page « mon espace » automatiquement.



Maintenant le QCM est disparu, pour voire le résultat on clique sur « Résultats » :

Tous les QCM passer on les trouvèrent dans cette page.

On remarque que le QCM qu’on vient de le passer est là, avec la note, le classement et la moyenne, avec la possibilité de voir la correction.

Quand on clique sur corriger une page va s’ajouter :

On va ajouter un deuxième étudiant de niveau CP2 :

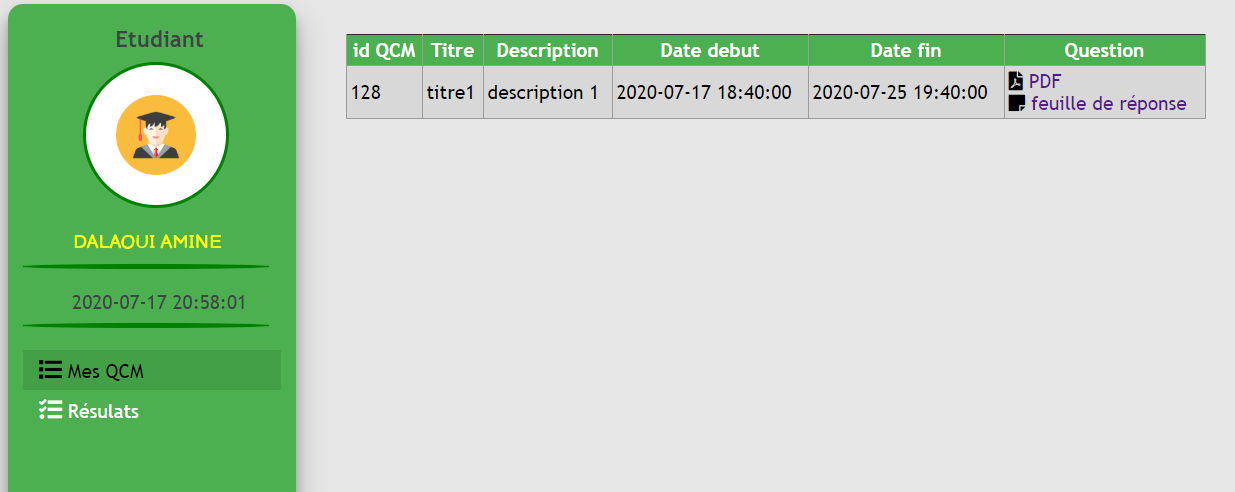




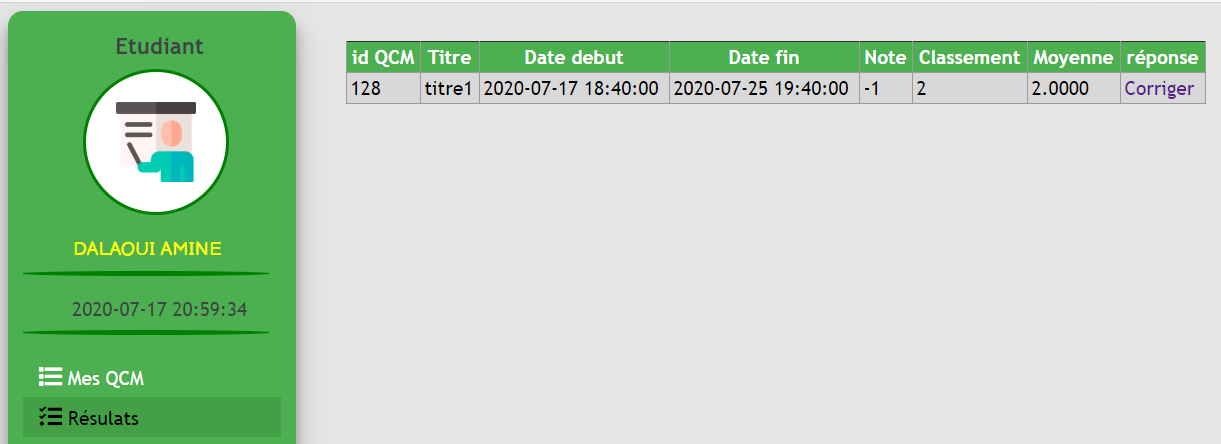
Même si le qcm est en cours mais cet étudiant ne peut pas le voir, car le QCM est pour le niveau CP1.

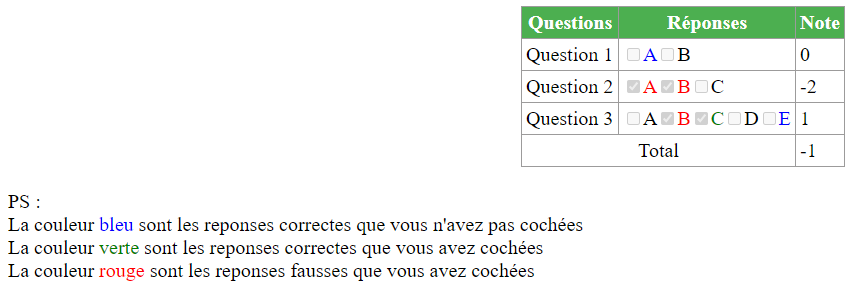
On va ajouter un deuxième étudiant de niveau CP1 :



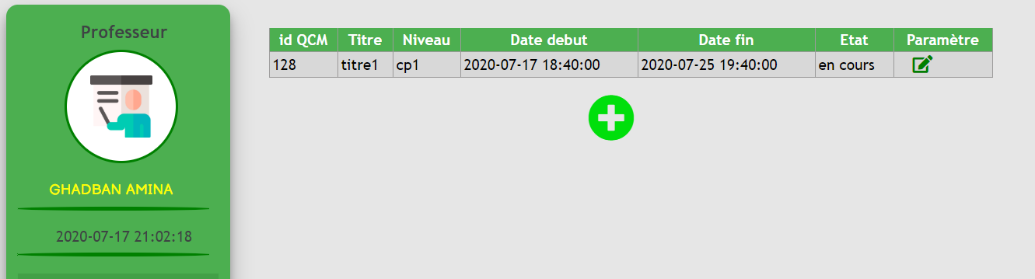


Et voici de nouveau le QCM apparait, on va répondre et visualiser le résultat.

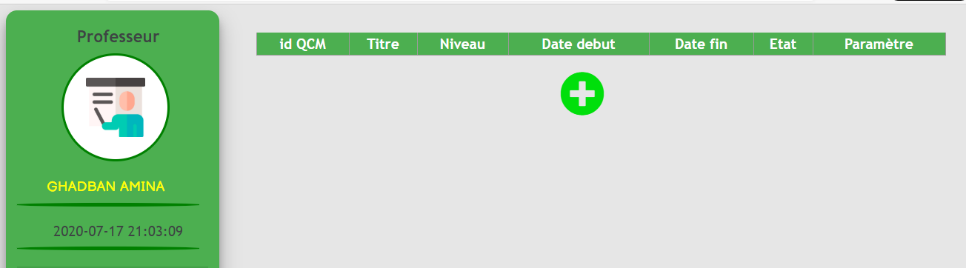


 La correction :

On revient au professeur :

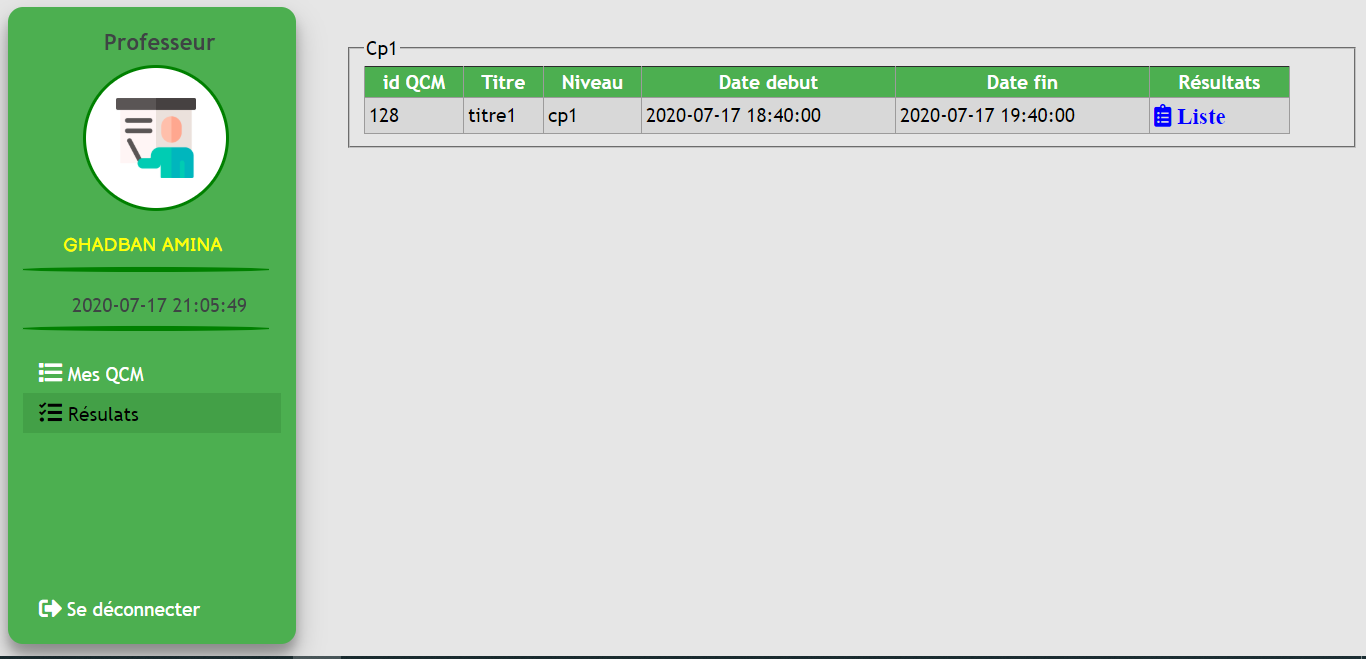


Le QCM est toujours en cours.



Une fois la date fin arrive, le QCM disparait.

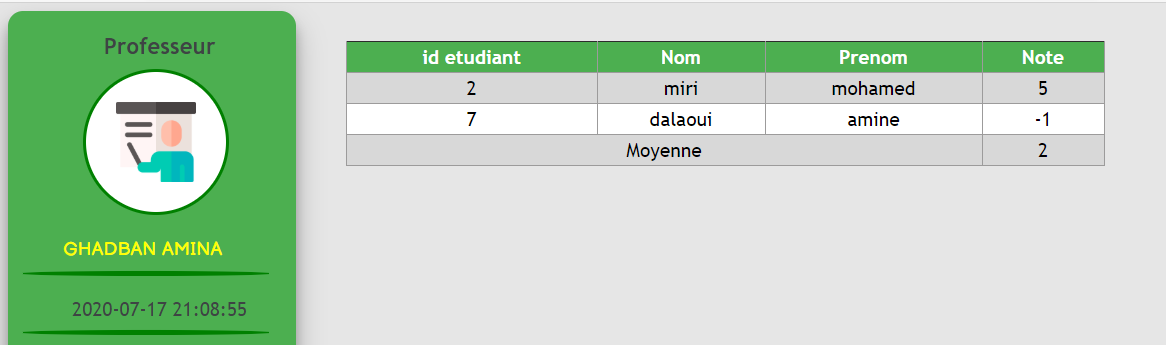
On passe à la page « Résultats » :



Tous les QCM créer par ce professeur et qui on passer leurs délais, seraient afficher dans cette page.

Tous les QCM seront regroupés par niveau (cp1, cp2, gi1…).

Pour voir le résultat d’un QCM, on clique sur liste :



La liste s’affichera dans l’ordre décroisant de la note comme demander dans l’énoncé du projet, cela est réaliser par la requête : (order by note DESC)

select e.id,e.nom,e.prenom,sum(r.note) as note

                    from etudiant e,reponse\_etudiant r

                    where r.id\_etudiant=e.id

                    and r.id\_qcm=?

                    group by e.id

                    order by note DESC

* 1. Les contraintes :

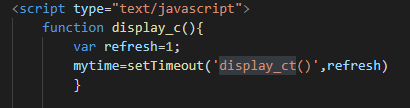
La programmation de ce site web est généralement facile, appart quelque difficulté que je vais mentionner :

* L’horloge :



J’ai voulu implémenter une horloge qui s’actualise à chaque seconde, cela n’est pas possible sous PHP, c’est pour ca j’ai passé au JS

Le code JS :

Cette méthode donne la possibilité d’exécuté la méthode display\_ct() chaque « refresh ».

La méthode display\_ct qui affiche la date et l’heure, on n’est pas intéressé par son contenue, avec un appelle à la méthode display\_c() à sa fin.

* Les icones :

   ETC..

Ces icones sont importés du site « https://fontawesome.com »

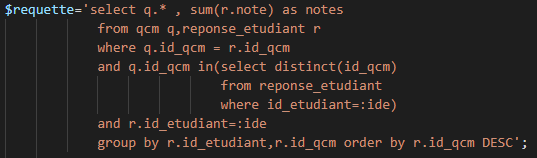
On le relie dans notre fichier dans le Head avec : 

Et on ajoute un petit code, par exemple l’icône de « Mes QCM » :

<i class="fas fa-list"></i>

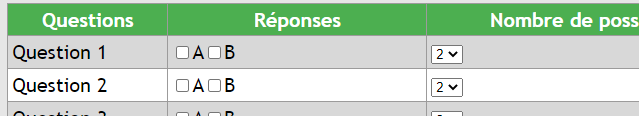
* Requêtte SQL:

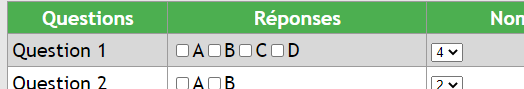
Dans certain cas l’importation des informations de la base de données devient compliquée, Par exemple la page resultat\_etudiant, la page qui affiche à un étudiant les QCM qu’il a passé :



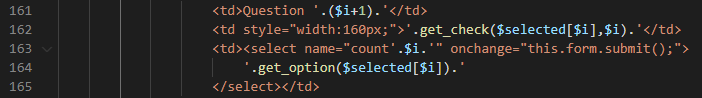
* Le changement du nombre de checkbox :

Quand le professeur est entrain de saisir la correction il peut modifier le nombre de réponse par question, cela doit être tous de suite changer :





Le code du combobox :

Ce code est inclue dans une boucle, $i représente le numéro de la question.

Si on change le nombre de checkbox, ONCHANGE s’exécute (this.form.submit() )

Le code qui s’exécute après celui-là est :



$selected est un tableau qui represente le nombre de reponse de chaque question, par exemple :

$selected = ( [0] -> 2 , [1] ->3) veut dire, la question 0 a 2 checkbox , la question 1 a 3 checkbox.

Quand le tableau est mis à jour, on va repasser par la méthode get\_check à la ligne 162 avec le nouveau $selected alors la méthode va nous générer différent nombre de checkbox.

Car le code PHP est avant le code HTML, car quand ONCHANGE s’exécute dans la partie HTML il nous fait revenir au code PHP qu’il est avant HTML du coup tous le code va être réexécuter.

1. Conclusion :

Tout d’abord, ce projet m’a permis d'appliquer les connaissances que j’ai acquises durant notre formation, telles que la modélisation, les différents langages de programmation, la gestion de projet.

Grâce à celui-ci j’ai pu renforcer mes connaissances. A travers des méthodes de travail et des outils, ce projet m’a permis de nous immerger dans un univers professionnel.

Il est vrai que de créer une application et respecter un cahier des charges rendent un projet tutoré intéressant et professionnel mais il y a aussi toutes les démarches qui ne sont pas visibles et qui rendent enrichissante une telle expérience. Le projet m’apporte une idée sur l’organisation dans le monde professionnel et qui permettra de m’adapter plus facilement lors de notre stage.