MARC DESLANDES (1360023),   
FRANCIS VERREAULT (1159578),  
 MICHAËL LACROIX-ROY (0585584),   
KEVIN BEAUDOIN (1213807)   
  
Projet d’intégration I  
420-B63-JO, gr. 00001

Projet d’intégration :   
Patrimoine

Travail présenté à   
Monsieur Denis Drolet

Département de l’informatique  
Cégep Régional de Lanaudière à Joliette  
Le 11 mars 2016

Table des matières

[1. Mise en contexte - 4 -](#_Toc445499700)

[1.1 But de l’application : - 4 -](#_Toc445499701)

[1.2 Demandes du client : - 4 -](#_Toc445499702)

[1.3 Vision de l’équipe : - 5 -](#_Toc445499703)

[1.4 Liste des tâches : - 6 -](#_Toc445499704)

[2. Exposition du système implémenté - 7 -](#_Toc445499705)

[2.1 -Départ - 7 -](#_Toc445499706)

[2.2 -Nouvelles fonctionnalités : - 7 -](#_Toc445499707)

[2.3 Fonctionnalités retirées - 8 -](#_Toc445499708)

[2.4 Langage : - 9 -](#_Toc445499709)

[2.5 Librairie : - 9 -](#_Toc445499710)

[2.5.1 -JQuery - 9 -](#_Toc445499711)

[2.5.2 -Mustache - 9 -](#_Toc445499712)

[2.5.3 -Dropzone - 10 -](#_Toc445499713)

[2.5.4 -Qr Code - 10 -](#_Toc445499714)

[2.5.5 -Foundation - 10 -](#_Toc445499715)

[3. Rapport d’analyse - 11 -](#_Toc445499716)

[3.1 [Bug #1] – Les photos n’apparaissent pas toujours au bon endroit. - 11 -](#_Toc445499717)

[3.2 [Bug #2] – Rendre le texte cliquable - 13 -](#_Toc445499718)

[3.3 [Bug #3] – Redirection Ajouter Modifier Image - 14 -](#_Toc445499719)

[3.4 [Bug #4] – Multiplication Image - 15 -](#_Toc445499720)

[3.5 [Bug #6] – Corriger date de fin de garantie - 16 -](#_Toc445499721)

[3.6 [Bug #7] – Corriger l’affichage sur appareil mobile - 17 -](#_Toc445499722)

[3.6.1 Page de connexion - 17 -](#_Toc445499723)

[3.6.2 Fenêtre pour téléverser des images - 17 -](#_Toc445499724)

[3.7 [Bug #8] – Page de login : S’inscrire enregistre les infos mais ne les affichent pas dans Comptes pour validation. - 18 -](#_Toc445499725)

[3.8 [Bug #10] – Vérifier le contenu des rapports - 19 -](#_Toc445499726)

[3.9 [Bug #12] – No image (quand on cancel l’ajout d’image) - 20 -](#_Toc445499727)

[3.10 [Bug #13] – Ajouter le nom du conteneur sur l’étiquette du code QR. - 21 -](#_Toc445499728)

[3.11 [Bug #15] – Ajout contenant principal - 22 -](#_Toc445499729)

[3.12 [Bug #16] – Indentation Arborescence - 23 -](#_Toc445499730)

[3.13 [Bug #17] – Augmenter le nombre d’objet visible maximum - 25 -](#_Toc445499731)

[3.14 [Bug #19] – Upload Image ne fonctionne pas toujours - 26 -](#_Toc445499732)

[3.15 [Bug #20] – Corriger l’emplacement du nom de fichier - 27 -](#_Toc445499733)

[4. Points forts de l’équipe - 28 -](#_Toc445499734)

[5. Conclusion - 29 -](#_Toc445499735)

[6. Annexes - 30 -](#_Toc445499736)

[6.1 Utilisation de l’activation : - 30 -](#_Toc445499737)

[6.1.1 Premièrement l’utilisation s’inscrit - 30 -](#_Toc445499738)

[6.1.2 L’utilisateur soumet son inscription - 30 -](#_Toc445499739)

[6.1.3 Cependant l’utilisateur n’est pas encore activé lorsqu’il essaie de se connecter - 31 -](#_Toc445499740)

[6.1.4 L’admin doit donc se connecter - 31 -](#_Toc445499741)

[6.1.5 Il a alors accès au panneau d’utilisateur et doit rechercher l’utilisateur - 31 -](#_Toc445499742)

[6.1.6 Il active l’utilisateur (l’utilisateur peut se connecté) - 32 -](#_Toc445499743)

[7. IMPORTANT (À LIRE) - 33 -](#_Toc445499744)

# Mise en contexte

## But de l’application :

Au début de la session, notre équipe a été formé afin de participer à un projet dans le cadre de notre cours projet intégration proposé par notre professeur Denis Drolet. Ce projet est en fait un site web qui a pour but de permettre à une famille ou un individu de recenser et d’enregistrer leurs biens et leurs possessions sur un site web. Le principe est assez simple et vise la simplicité d’utilisation !!! Tout d’abord, il suffit à une personne de s’inscrire et de se connecter sous son compte afin de pouvoir commencer son patrimoine. Ensuite, en suivant le principe des poupées russes, la personne commence par créer des contenants qui contiendront ensuite d’autres contenants qui contiendront par la suite d’autres contenants. Par exemple, une maison contient un salon qui contient une armoire qui contient d’autres items. Pour chacun des contenants, on peut spécifier un nom, un prix, une description et ajouter une ou plusieurs images. En tout temps, l’utilisateur peut générer des rapports afin d’obtenir la valeur totale de ses biens ou d’un contenant en particulier. De plus, il est possible de générer et d’imprimer des codes QR pour un contenant afin de savoir rapidement les objets qui s’y trouvent. Finalement, un utilisateur peut ajouter des membres dans son patrimoine et ajouter des contacts dans le cas où certain de leurs objets seraient en possession d’un ami.

## Demandes du client :

Le site web du Patrimoine est un projet déjà implémenté par une équipe des années passées, mais le site web n’est pas fonctionnel dans sa totalité. Donc, notre mandat est de finaliser celui-ci en testant toutes les fonctionnalités, en corrigeant les bugs rencontrés et en modifiant les fonctionnalités problématiques, afin que notre client puisse le mettre sur le marché. Les points marquants en ce qui a trait aux demandes de monsieur Denis Drolet, sont la gestion d’images et d’objets dans la section Patrimoine, la gestion des comptes utilisateurs au niveau de la hiérarchie, ainsi que l’ajustement et l’indexation de l’arborescence d’objets.

## Vision de l’équipe :

Notre équipe est formée de 4 membres, soit Kevin Beaudoin, Marc Deslandes, Michael Lacroix-Roy et Francis Verreault. Notre but consiste à rendre le site web fonctionnel, de modifier le moins de codes possibles, et de respecter la vision de l’équipe précédente et du client. Notre mandat n’est pas d’optimiser le code, mais bien de comprendre le système actuel et de modifier le code nécessaire pour corriger les erreurs. L’idée est que nous devons livrer le produit dans un délai restreint et nous voulons éviter de s’embourber dans des modifications en cascade, donc on se limite au problème directement. Finalement, les seuls changements majeurs que nous proposons sont d’enlever l’option d’ajout d’une zone réactive, de changer le principe de l’objet racine en le créant par défaut de manière à pouvoir créer plusieurs contenants principaux.

## Liste des tâches :

* Les photos n’apparaissent pas toujours au bon endroit.
* Glisser-déposer une image ici ne fonctionne pas bien.
* Standardiser la dimension des photos.
* Indenter le contenu des contenants dans l’arborescence.
* Utiliser une barre de défilement verticale et horizontale plutôt que le «Plus… ».
* Corriger la fonction ajout de zone (les zones se multiplient) sur la photo d’un conteneur.
* Ajouter le nom du conteneur sur l’étiquette du code QC.
* Vérifier le contenu des rapports.
* Page de login : S’inscrire enregistre les infos mais ne les affichent pas dans Comptes pour validation.
* La validation userinfostatus dans la table userinfo demeure donc à zéro.
* Ajouter un conteneur principal manuellement suite à la validation d’un nouvel utilisateur et se reconnecter pour que ça fonctionne.
* Dans modifier, corriger date de fin de garantie : enlever aucune garantie du champ.
* Corriger toutes autres erreurs trouvées.

# Exposition du système implémenté

## -Départ

Projet patrimoine est un site web permettant de gérer des objets vous appartenant. L’expérience débute par la création d’un compte et l’attente de l’activation de celui-ci. Une fois le compte activer, l’utilisateur peut se connecter et ajouter/supprimer/modifier des objets qui lui appartiennent. Ceux-ci fonctionne par hiérarchie et permet d’ajouter des items dans chaque objets dans une idée de conteneur-contenant permettant une navigation facile dans l’inventaire. Des images peuvent être reliées à chaque objet pour bien reconnaitre chaque objet. Il est aussi possible de faire des prêts à des contacts et de suivre leur progression et retour. Cependant, L’utilisation d’un inventaire n’est pas restreinte à un utilisateur. Celui-ci peux rajouter des utilisateurs qui seront considérer comme des membres de sa famille et pourront gérer ou voir (selon ce que l’administrateur de famille leur aura permis) les objets de la famille. De plus il est possible de créer des rapports de chaque contenant et d’analyser en profondeur le patrimoine de l’individu. Le rapport restreint permet d’accéder en détail à un item et son contenue, tandis que le rapport d’architecture permet d’accéder à l’architecture complète en dessous d’un conteneur sélectionner mais possède moins de détaille que le rapport simple. Finalement l’utilisateur peut créer un code QR qui représente un conteneur et son contenues, qu’il peut par après imprimer et mettre sur l’objet qu’il représente pour facilement localiser le contenue présent dans le conteneur.

## -Nouvelles fonctionnalités :

Se connecter sur un compte administrateur du site web permet maintenant d’activer un compte utilisateur et la gestion générale des comptes utilisateurs (Suppression, modification etc.). En ce moment le seul administrateur du site web est [etudiant@b63.com](mailto:etudiant@b63.com). Pour ajouter un administrateur, il faut manuellement aller dans la base de donner et modifier le champ UserInfoIsMod à « 2 ».

Un nouveau rapport voie le jour : Rapport complet. Il y avait déjà un rapport complet, certes, mais celui-ci n’étais pas réellement complet puisqu’il affichait en réalité que l’architecture du conteneur et non toutes les informations des objets. Donc maintenant le rapport complet afficheras en détaille les informations d’un conteneur, leur contenue et à leur tour leur contenue, ainsi de suite. Ce sera un rapport très détaillé mais très utile lorsque l’utilisateur veux analyser ces acquis en profondeur. Les autres rapports préexistant sont toujours accessibles, celui-ci n’est qu’un ajout pour faciliter l’utilisation de l’utilisateur.

## Fonctionnalités retirées

La fonctionnalité des zones réactives était une fonction déjà présente à la réception du projet. Elle permettait d’ajouter une ou plusieurs zones sur une image d’un item et de donner à celles-ci des références vers d’autres contenus à l’intérieur de l’item en cours. Par exemple, dans une image qui représente un bureau, on pourrait mettre une zone réactive sur le premier tiroir ce qui veut dire qu’en cliquant sur ce tiroir (dans l’image) le site nous redirige vers ce tiroir qui peut contenir ou non d’autres items. Malheureusement, la fonction n’était pas bien implémentée et, donc il y avait de multiples bogues qui rendaient la fonction difficile à utiliser surtout sur mobile où la sélection était pratiquement impossible. Par conséquent, nous avons décidé de retirer complètement cette fonction qui aurait dû être refaite complètement puisque la librairie utilisé « DropZone » est pratiquement non-modifiable et que le temps nous manquait pour refaire complètement la fonction en « Javascript ».

## Langage :

* Javascript : C’est le Javascript qui est responsable d’effectuer des opérations du côté client.
* PHP : Le PHP est le contrôleur de l’application, il s’occupe de décider quelle information, page et action est accessible à l’utilisateur
* HTML : Affichage direct à l’utilisateur. Structure générale du programme. Utilise Mustache pour l’affichage des données.
* CSS : Permet de changer l’apparence du code HTML. La majorité de l’apparence des contrôles sont prédéfinie par Foundation
* SQL : Permet de faire les requêtes à la base de données via une connexion PDO pour obtenir les données de l’utilisateur.

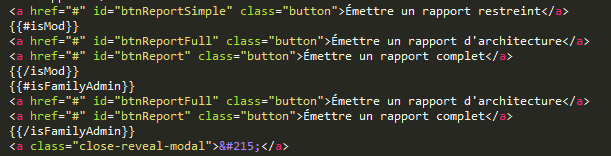
## Librairie :

### -JQuery

Librairie classique de JavaScript, contient des raccourcis de JavaScript pour faciliter le codage. La librairie n’apporte aucune fonctionnalité particulière au site, elle est simplement utile pour le programmeur

### -Mustache

Mustache est un gabarit sans logique permettant de transférer des données depuis le PHP vers le html (À préciser qu’il y a plusieurs versions qui s’appliquent à différents langages, mais ici il est utilisée PHP). On le dit sans logique puisque qu’il n’y a pas de « if », « else » ou « for ». Ils sont remplacés par des « tag » ce qui rend l’usage beaucoup plus simple. Ici Mustache est utilisé pour afficher en HTML les différentes requêtes SQL à la BD et détermine quel bouton et contrôle est visuellement accessible sur la page à l’utilisateur.



Exemple : mustache (représenté par l’utilisation des {{…}}) permet de cacher les rapports si l’utilisateur n’y a pas accès

### -Dropzone

Dropzone est une page de chargement de fichier pré-créé pour permettre à l’utilisateur de télécharger ces fichiers de façon simple et très visuel. Ici Dropzone est l’interface de téléchargement des images pour les objets du patrimoine de l’utilisateur.

### -Qr Code

QrCode est une librairie très simple qui permet de changer une chaine de caractère vers un Code QR. Ici le code QR permet de sauvegarder un conteneur et son contenue dans un code QR et l’afficher à l’utilisateur.

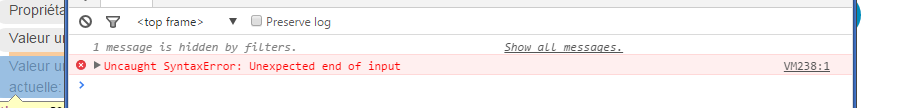
### -Foundation

Foundation est un  « framework » se basant principalement sur le design et apparence des pages. I permet avec très peu d’effort de rendre le système élégant et plus simple d’utilisation.

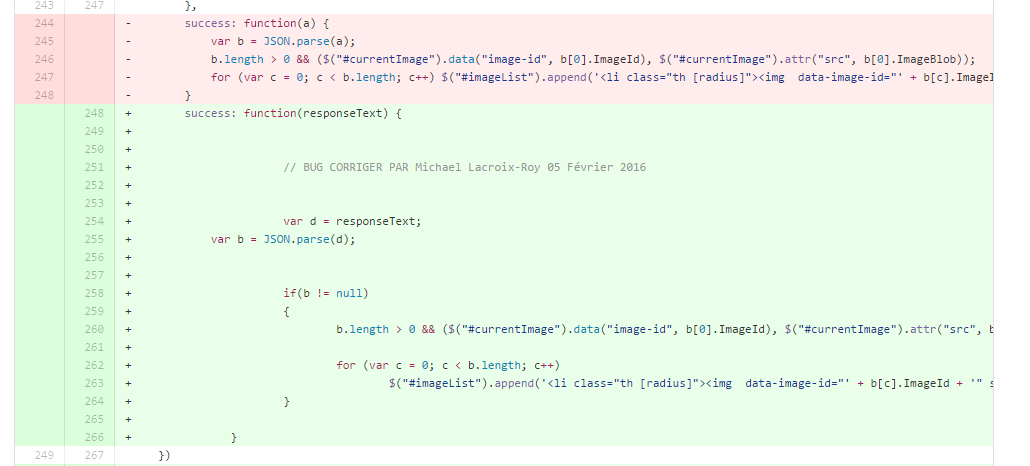
# Rapport d’analyse

## [Bug #1] – Les photos n’apparaissent pas toujours au bon endroit.

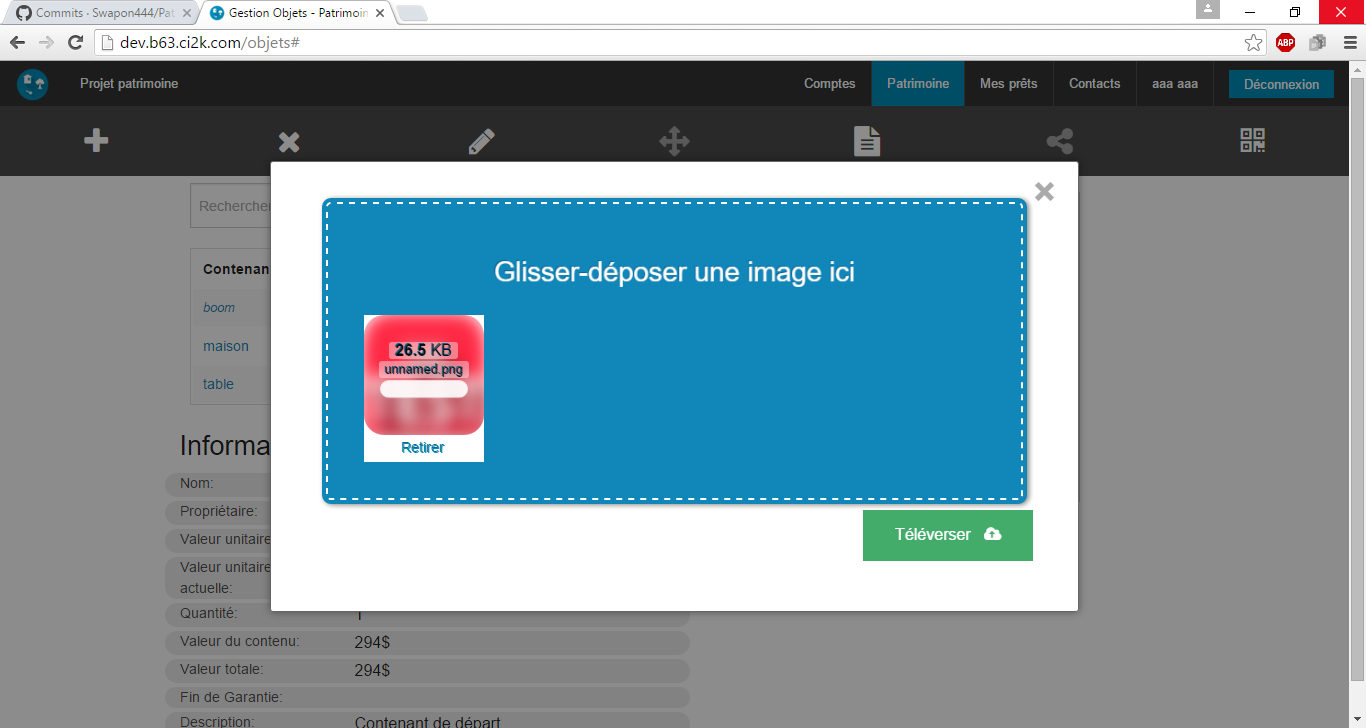
**Description** : Le problème suivant concerne la mise à jour des images d’un objet au chargement de la section Patrimoine et suite à la sélection d’un autre objet. Les images ne sont pas chargées et cela empêche la suite du déroulement dans la page, surtout au niveau de l’ajout et de la suppression d’image. De plus, une erreur dans la console survient au niveau d’une méthode en JavaScript.

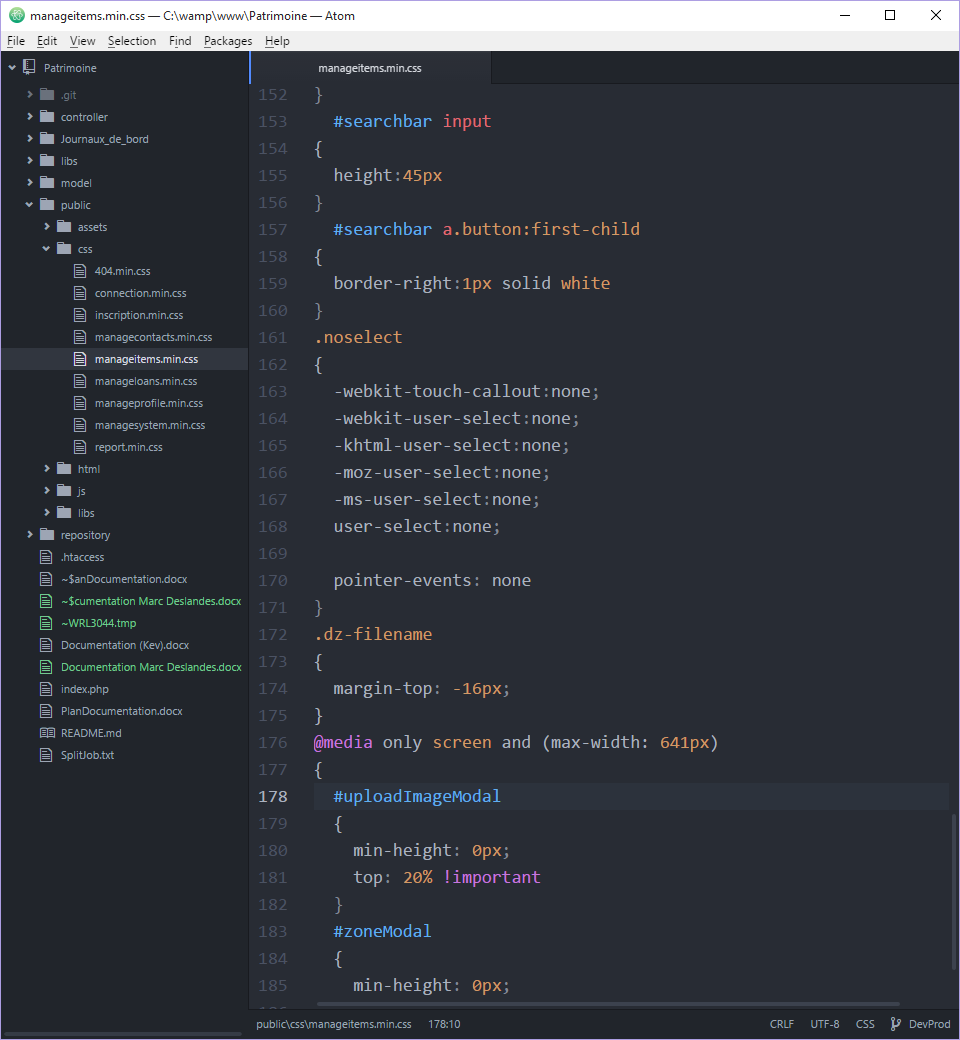


**Solution** : J’ai passé beaucoup de temps à côté du problème, sans savoir pourquoi le script avait une erreur car dans cette méthode, une requête Ajax est envoyé au server et attend une réponse sous forme de JSON pour poursuivre. Ceci dit, je pensais que le problème était du côté server et qu’aucune réponse n’était envoyée au client. J’ai debuggé en regardant les données envoyées par le server, et tout était correct, sans toutefois fonctionner. Donc, je me suis attardé à la requête Ajax. Une fois de plus, les données envoyées au server étaient correct, donc l’erreur était ailleurs. En effet, l’erreur ciblée venait vraiment de la réception de la réponse, et plus précisément de la ligne qui s’occupe d’encoder le résultat en JSON. Le problème est que la réponse est bel bien reçu correctement, mais elle est en même temps « parser » en JSON, ce qui supposément n’est pas gérer. Donc, la solution a été de premièrement affecter le résultat à une variable, et d’ensuite « parser » la variable dans une autre variable. Une autre erreur s’est glissée après la modification, si le résultat était null, la condition « b.length » occasionnait une erreur car on ne peut pas obtenir la longueur d’un objet null. Donc, j’ai rajouté une condition si b est différent de null.

**Correction** : 

## [Bug #2] – Rendre le texte cliquable

**Explication**

Dans la fenêtre pour téléverser des images on peut soit glisser-déposer une image sur la zone bleu ou cliquer sur cette zone permettant d’ouvrir une boîte de dialogue où l’on peut choisir notre fichier. Le problème était que le texte était sélectionnable, donc pour ouvrir la boîte de dialogue il fallait cliquer ailleurs que sur le texte (sur le bleu extérieur) sinon cela sélectionnait le texte et donc la boîte de dialogue ne s’ouvrait pas.

**Correction**

Ce problème peut être corrigé tout simplement avec du CSS. J’ai vu qu’il y avait déjà une classe CSS « noselect », mais elle ne fonctionnait pas à 100%. J’ai seulement ajouté la ligne « pointer-events : none » dans la classe déjà existante pour régler le problème.

## [Bug #3] – Redirection Ajouter Modifier Image

**Description** : Dans la section Patrimoine, lorsqu’une image est ajoutée ou supprimer, la page ne se met pas à jour et les images ne sont pas chargés.

**Solution** : Ce bug se situe en même temps que le bug précède, qui était que la redirection sur la sélection d’objet ne fonctionnait pas. Une des solutions que j’avais envisagées était de tout simplement recharger la page. Par contre, en rechargeant la page, l’objet par défaut est sélectionné et la redirection ne fonctionne pas, donc les images ne sont pas plus chargées. De plus, le but d’une requête Ajax est de permettre de rendre la page dynamique, donc cette option perdait tout son sens. Ceci, je me suis concentre sur l’autre problème et après l’avoir corrigé, les problèmes d’ajouts et de suppressions d’images ont disparues. Alors, il y avait aucun problème à ce niveau.

**Correction**: Aucune correction n’a été nécessaire.

## [Bug #4] – Multiplication Image

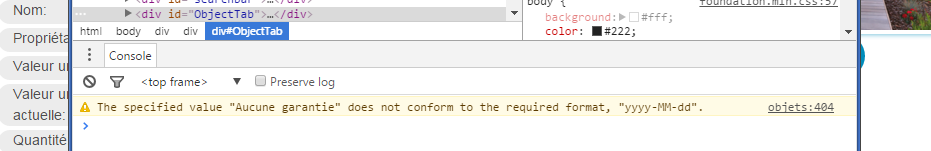
**Description** : Ce problème est rencontré sur la version qui est sur le web, et non en local, due à la différence de vitesse entre la communication du serveur et de l’application. Lorsqu’on sélectionne des objets très rapidement en changeant d’un à l’autre, dans la section Patrimoine, les images se multiplient en chargeant des images qui ne correspondent pas avec l’objet sélectionné.

**Solution** : Lors de la sélection d’un objet dans la liste, une requête est envoyé au server par Ajax afin d’obtenir la liste de tous les images de cet objet. Or, en local, le problème n’est pas apparent car le délai de temps pour charger les images est presque instantané. Par contre, en ligne, les images prennent un certain temps à charger, et par le fait même, prennent du temps à envoyer le résultat au script apax qui est en mode écoute. Alors, en changeant d’une image à l’autre, les scripts s’accumulent effectuent leur code en simultanée dépendamment du délai de chacune des requêtes. Donc, en bref, la solution était d’abandonner la requête si l’objet n’est plus celui en cours.

**Correction** :

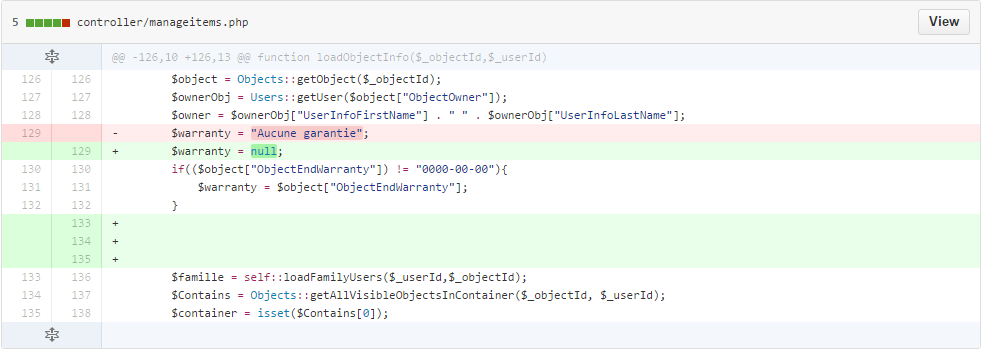
## [Bug #6] – Corriger date de fin de garantie

**Description** : Erreur dans la console lorsqu’un objet du patrimoine est sélectionner dans la section patrimoine car le champ associé à la garantie de l’objet doit contenir une date et contient dans certains cas aucune date, exprimé par la chaine de caractère suivante : “Aucune Garantie”.



**Solution** : La solution envisagée est assez simple, au lieu d’affecter une chaine de caractère, on affecte null à la variable qui sera affiché dans le champ de la date de garantie.

**Correction** :

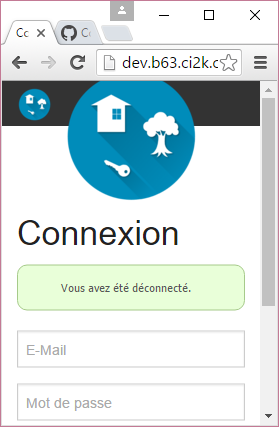


## [Bug #7] – Corriger l’affichage sur appareil mobile

**Explication**

Lorsqu’on allait sur le site du patrimoine avec un appareil mobile (cellulaire), certains éléments n’étaient pas affichés à la bonne position due à la résolution réduite.

### Page de connexion

Sur la page de connexion lorsque que l’écran était vraiment petit, le logo embarquait sur le menu et/ou dépassait de l’écran. Encore ici, cela peut être accomplit avec du CSS avec les « media queries ». Lorsque le site est en résolution suffisamment petite on n’a qu’à changer la position du logo. Dans les deux images ci-contre, on peut voir l’avant-après.

### Fenêtre pour téléverser des images

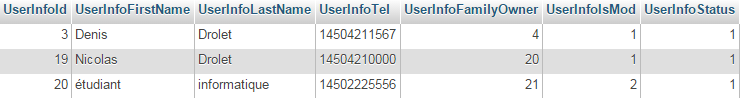
Lorsque le site set en affichage avec une résolution mobile l’utilisateur est obligé de défiler vers le bas pour cliquer sur les boutons pour ajouter une image à ces items. Le problème était que le bouton nous ramène est haut de la page alors que la fenêtre apparaissait. Pour régler le problème j’ai utilisé du CSS et j’ai mis à la fenêtre une hauteur minimale de zéro pixel ainsi qu'une marge supérieure négative.

## [Bug #8] – Page de login : S’inscrire enregistre les infos mais ne les affichent pas dans Comptes pour validation.

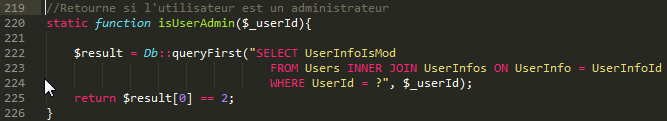
Le problème ici est qu’il n’y a aucun moyen d’activer un compte qui tente de s’inscrire pour l’administrateur du site web.

La correction de ce problème n’étais pas compliquer, cependant plusieurs concept appliquer dans le code du site web étais très mal démontrer dans les vues de celui-ci. Par exemple tout le système de «Famille » était inexistant sur le site web et la gestion de la famille a été confondue avec la gestion des chefs de familles. Pour être plus précis, il n’y avait simplement pas de compte administrateur dans la base de donnée, seulement des chefs de familles qui avait la gestion de leur propre famille, mais personne pour gérer les chefs de famille et activer leur comptes. Un autre problème rencontré fut l’utilisation du Template « mustache » qui permet de transférer des informations depuis PHP vers le html. Puisque la gestion de l’information étais très dépendante de se Template, ce qui inclus quel page est affiché/accessible à quel utilisateur, il était vital de comprendre son fonctionnement.

Pour régler le problème il a fallu créer un compte admin, qui est identifier par un l’integer « 2 » dans la colonne « UserInfoIsMod » de la table « userinfos »



Ensuite il a simplement fallu changer la fonction « isUserAdmin » dans le fichier « users.php », qui détermine si un utilisateur est administrateur, pour qu’elle reconnaisse la colonne changée plus tôt et détermine que si la colonne est égale à 2, l’utilisateur est un administrateur.



La page d’administration était déjà présente.

## [Bug #10] – Vérifier le contenu des rapports

Le problème ici était supposément que les rapports n’affichaient pas correctement l’information.

Le problème était plutôt dans le concept des rapports. Ceux-ci affichaient belle et bien les informations supposé, mais étais mal nommé. Par exemple le rapport simplifié affichait beaucoup plus d’informations sur les items que le rapport complet, ce qui laissait supposé que celui-ci n’étais pas réellement simplifié. En réalité le rapport simplifié déterminait que seul le conteneur et ces enfants directes étaient démontrés, le reste étais laissé de côté, mais affichait en détail ceux-ci. Alors que le rapport complet donne seulement le nom des items, mais affiche toute l’arbre en dessous du conteneur sélectionné. Les noms ne représentent simplement pas leur fonctionnalité, mais celle-ci est bien présente.

J’ai donc changé le nom de rapport simplifié pour restreint, car il est restreint a un groupe d’item au lieu de toute l’arbre, et le rapport complet est devenue le rapport d’architecture car il donne très peu de détails à l’exception de l’architecture complète du patrimoine.

## [Bug #12] – No image (quand on cancel l’ajout d’image)

**Description** : Lorsqu’on sélectionne le bouton pour ajouter une image, un dropzone s’affiche afin de pouvoir sélectionner une image dans notre répertoire. Si l’usager décide de fermer le dropzone au lieu d’ajouter une image, alors aucune image ne se charge à l’écran pour l’objet en cours.

**Solution** : Après l’analyse du processus d’ajout d’images, avant d’afficher le dropzone, la liste d’images est vider, donc s’il y a un ajout d’images, les méthodes qui suivront l’ajout feront leur rôle de chargement d’images et de la mise à jour de la page. Par contre, si l’usager n’ajoute pas d’image, aucun code ne se poursuit, étant donné que rien n’a changé dans le contexte. Ceci dit, au lieu de vider la liste d’images avant le « dropzone », elle sera vidé seulement si un ajout d’image est effectué.

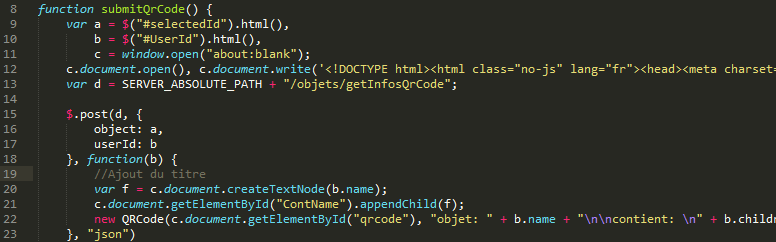
**Correction** :

## [Bug #13] – Ajouter le nom du conteneur sur l’étiquette du code QR.

Le problème est qu’il n’y a pas de titre au code QR donc il peut ne pas être très claire sur quoi se base le code QR, surtout une fois imprimer et que l’utilisateur veut apposer l’étiquette sur un objet. Ainsi en écrivant le titre il est facile de savoir de quel conteneur le QR représente.

La plus grande difficulté de ce problème fut de comprendre le fonctionnement des requêtes SQL à la BD puisque celle-ci ont plusieurs redirection dans les méthodes et l’utilisation du Ajax en JQuery.

Une fois le code analysé, il a simplement fallu changer la fonction « submitQrCode » dans le fichier « manageitems.min.js » pour que la fonction affiche aussi le nom du conteneur (qui était déjà renvoyer par la requête SQL) en plus du code QR.



Exemple : j’ai ajouté une balise <h1> sur l’affichage à la ligne 12, et affecter un « text node » à la ligne 20/21

## [Bug #15] – Ajout contenant principal

Pour ma part j’ai reçues la tâche de modifier grandement l’affichage des objets. De plus je devais ajouter la possibilité de faire des contenants principaux. J’expliquerai premièrement l’ajout des contenants principaux et deuxièmement l’indexation des objets qui a été plus laborieux.

L’ajout d’un contenant principal a impliqué beaucoup plus de changements que je le croyais. En effet l’ensemble du système de stockage des objets fonctionnait de façon à ce qu’un seul contenant principal existe. La manière de trouver les objets d’un propriétaire consistait à trouver son unique contenant avec le type contenant principal et de prendre ses enfants. La recherche et l’affichage des objets impliquaient des méthodes PHP, du JavaScript, du HTML, CSS et du SQL. Toutes ces méthodes fonctionnaient en cherchant l’unique contenant principal. Je tenterai de plus expliquer mes changements que de montrer du code puisqu’il y a eu des changements à une dizaine d’endroits. La solution aux contenants principaux est venue de Marc qui a suggéré d’enlever l’affichage de l’unique contenant principal déjà existant. Ainsi nous pouvions avoir plusieurs contenants principaux soit les premiers objets du contenant initial. La façon d’identifier les contenants principaux devenait très simple; regarder si son parent est le contenant initial. Ceci a donc impliqué l’ajout de quelques méthodes PHP effectuant de simples « queries » à la bd. Voici donc de quoi à l’air la nouvelle fonctionnalité d’ajout de contenant racine:

## [Bug #16] – Indentation Arborescence

Cette partie était de loin la plus compliquée puisque le code existant n’était pas commenté et il avait été optimisé par un programme qui enlève les sauts de lignes donc l’indentation était à refaire. Cela n’aurait pas été très difficile mais ce code était en jQuery une extension de JavaScript que je connaissais très peu. J’ai beaucoup travaillé avec Marc sur cette fonctionnalité et le principal problème était de changer les fonctions PHP qui allaient chercher les objets du user. Dans l’ancienne version les objets parents n’étaient affichés alors que maintenant ils le sont lorsqu’on va plus loin dans l’arborescence des objets. Voici un exemple de la nouvelle arborescence.



En fin de compte la fonction qui était déjà présente fût complètement remplacée. Nous n’avons gardé que les styles CSS qu’elle appliquait au tableau et à ses éléments.



L’ajout de ce bouton permet donc d’ajouter des contenants racines, s’il n’est pas sélectionné le nouvel objet sera ajouté dans l’objet présentement sélectionné sauf si il n’y aucun contenant principal le nouvel objet sera un contenant principal. De plus il nous a obligé à changer les données de quelques clients qui désiraient conserver leur ancien contenant principal.

## [Bug #17] – Augmenter le nombre d’objet visible maximum

Ce problème c’est réglé par lui-même en changeant complètement la méthode PHP qui obtenait les objets à afficher dans l’arborescence. L’ancien code ne permettait que de récupérer 8 objets. Le nouveau code récupère absolument tous les objets et ils sont ensuite affichés par le JavaScript.

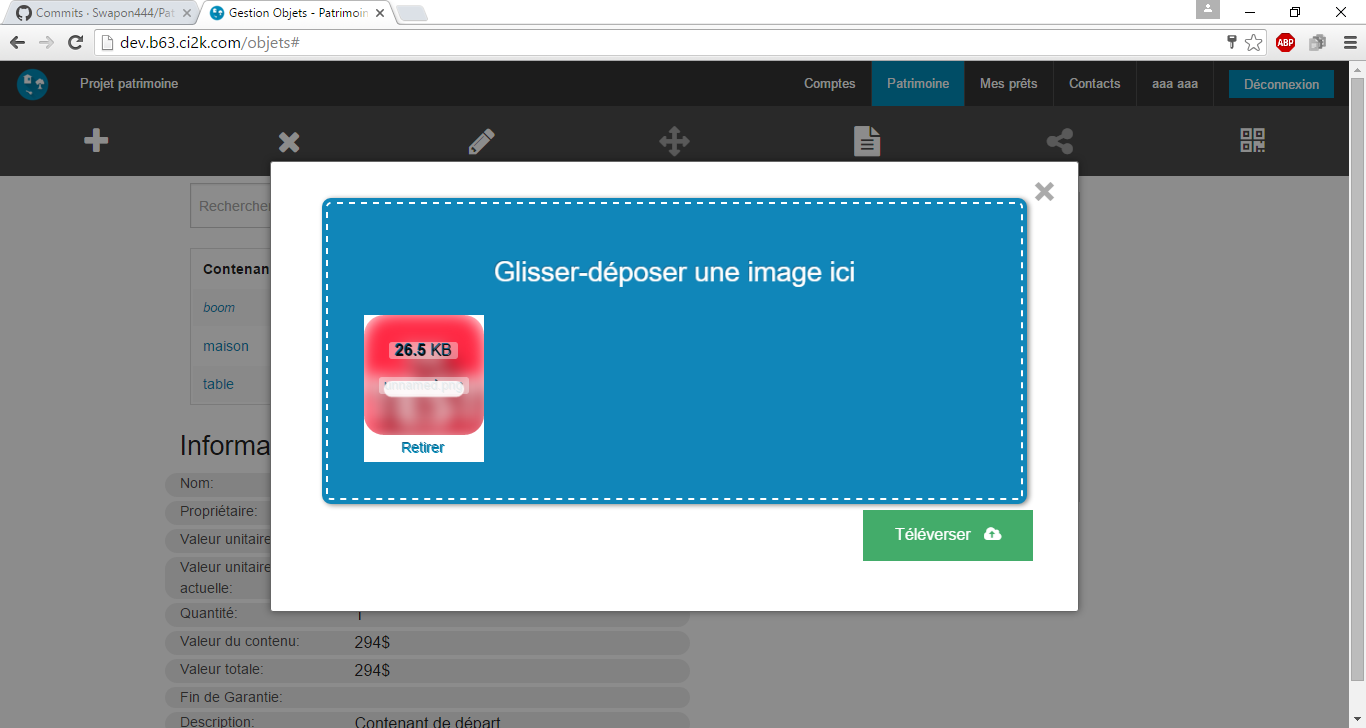
## [Bug #19] – Upload Image ne fonctionne pas toujours

**Description** : D’une façon assez imprévisible, lors d’ajout d’images, le chargement des images échoue dans le sens où la dernière image ajoutée ne se charge pas. Cette situation n’est pas fréquente, mais occasionne un problème au niveau de la mise à jour des éléments à l’écran.

**Solution** : Tout d’abord, je croyais que l’image n’avait pas le temps de s’enregistrer dans la base de données, étant donné que la situation est plus fréquente sur la version en ligne. Donc, j’ai rajouté un « sleep » de 3 secondes lors de l’ajout de l’image afin de laisser du temps pour la sauvegarde. Cependant, malgré le taux de réussite au niveau du local, la version web n’était pas convaincante. J’ai donc regardé au niveau des méthodes prêt de la requête SQL, soit la méthode « addImage » dans le fichier Références et le « loadImage » dans le « manageItems.php ». Croyant que le « sleep » n’agissait pas, j’ai pensé mettre une boucle pour m’assurer que l’image soit ajoutée avant de continuer le code restant. Après beaucoup de test, la situation continuais de se reproduire, mais toujours dans des contextes différents, rendant très laborieux la détection de la source du problème. Alors, j’ai commencé à mettre des alertes un peu partout pour observer les étapes à partir du bouton « submit » du dropzone jusqu’au chargement des éléments dans la page. Après plusieurs tests, j’ai pu identifier où est ce que le problème partait, soit dans la méthode « SubmitImage ». Dans cette méthode, d’autres méthodes sont appelés, une à la suite de l’autre afin d’insérer l’image, de les charger et de les afficher. La première méthode appelée est « processQueue », de l’objet dropzone, ensuite la méthode « loadimages ». L’erreur se produit entre ces deux méthodes. Quand on est dans la situation non fonctionnelle, la méthode « loadimage » n’attend pas que la méthode « processQueue » termine. Donc, à partir de ce moment, j’ai cherché des solutions au niveau du dropzone.

## [Bug #20] – Corriger l’emplacement du nom de fichier

**Explication**

Dans la fenêtre pour téléverser des images, lorsqu’on choisit un fichier soit en le glissant-déposant ou en choisissant le fichier dans la fenêtre de dialogue une petite icone apparait représentant notre image. Lorsque l’on met notre souris sur cette miniature, la taille et le nom du fichier apparaissent, mais le nom de celui-ci était directement sous la barre de progression ce qui le rendait presque impossible à lire comme démontrer dans l’image ci-contre. 

**Correction**

Il s’agit d’un simple problème de positionnement d’un élément HTML, c’est-à-dire qu’il va falloir le repositionner grâce au CSS. Ceci aurait pu être accompli de plusieurs façons, mais j’ai décidé de simplement modifier une classe déjà existante pour mettre une marge supérieure négative. Dans les deux images ci-contre, on peut voir le code CSS nécessaire et le résultat suite à la correction.

# Points forts de l’équipe

Tout d’abord, de mon point de vue, la chimie de l’équipe a nettement aidé au développement de ce projet. J’ai fait plusieurs travaux d’équipes au cours de la technique avec Marc, Francis et Kevin, ce qui rendait la communication facile. Chacun s’est adapté rapidement à son rôle et l’atmosphère était détendu étant donné qu’on se connaissait tous très bien, donc pas de malaise. De plus, chacun semblait se soucier des autres membres de l’équipe au niveau des interrogations ou des problèmes rencontrés.

Ensuite, l’implication des membres de l’équipe était équilibrée et présente en tout temps. Tout le monde s’est occupé de leur partie ainsi que coopérer au partie des autres membres de l’équipe.

Finalement, j’ai trouvé que les membres de l’équipe étaient très compétents et cela a nettement contribué au succès du projet. Même si nous avons des compétences différentes, chacun semble avoir acquiert de bonne connaissance au cours de la technique.

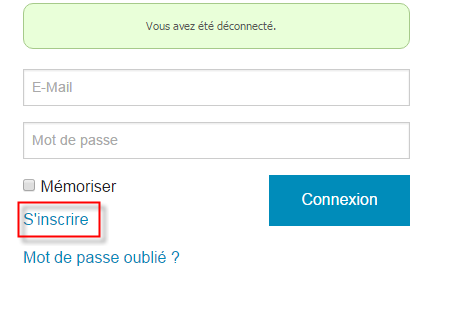
# Conclusion

En conclusion, notre équipe de quatre programmeurs a reçu le mandat de mettre à jour un site web de gestion de patrimoine. Cette tâche nous demandait tout d’abord de bien comprendre le projet dans son ensemble puis de trouver les fonctionnalités que le client voulait améliorer ou supprimer. Nous avons appris que travailler sur un projet déjà codé demande beaucoup de préparation. Lorsque je parle de préparation, je veux dire apprendre les technologies du programme et sa fonctionnalité générale. Sans cette préparation il aurait été extrêmement difficile et pénible de se retrouver dans les milliers de lignes de codes pour chercher la cause d’un bug. L’implémentation d’une version locale était une partie cruciale de la préparation. Cette partie était absolument nécessaire puisque le projet se trouvait sur un serveur distant et que nous devions pouvoir tester rapidement les modifications. Cette importation fût difficile dû au fait de notre faible expérience en matière de serveur. De plus le projet Patrimoine possédait une base de donnée qui heureusement c’est facilement installé dans MySQL. En bref, travailler sur le code d’autres personnes a été difficile et d’autant plus lorsque le code n’était pas commenté. Côté coéquipier je n’ai pour ma part que du positif à dire sur mes camarades. La division des tâches au début du projet c’est très bien déroulée et tout le monde a accompli ce qui était attendu de leur part. De plus Mike a été un très bon leader pour Marc, Kevin et moi, il était celui qui avait la meilleure vue d’ensemble sur le projet et qui savait si le travail avançait suffisamment selon les buts que nous nous étions fixés.

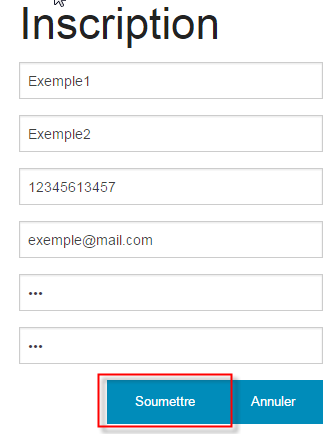
# Annexes

## Utilisation de l’activation :

### Premièrement l’utilisation s’inscrit



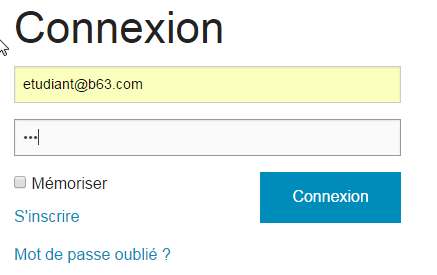
### L’utilisateur soumet son inscription

****

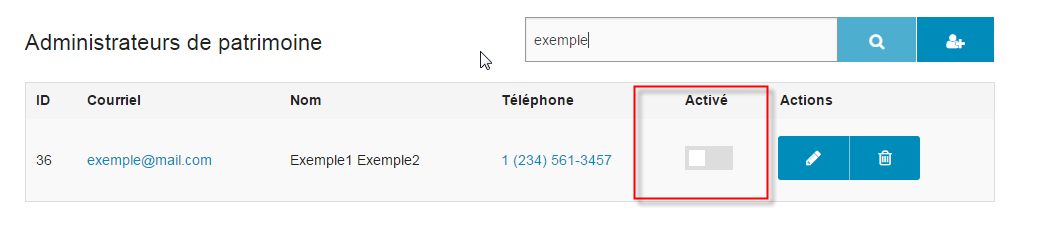
### Cependant l’utilisateur n’est pas encore activé lorsqu’il essaie de se connecter

**C:\Users\1213807\Desktop\exempleActivation\PasActiver.png**

### L’admin doit donc se connecter

****

### Il a alors accès au panneau d’utilisateur et doit rechercher l’utilisateur

****

### Il active l’utilisateur (l’utilisateur peut se connecté)

****

# IMPORTANT (À LIRE)

Dû à l’implémentation de la nouvelle indentation des contenants, nous avons changé un peu la façon dont l’affichage fonctionne ce qui fait que pour le compte [denis@b63.com](mailto:denis@b63.com) on ne voit pas le terrain. J’ai été modifié légèrement la base de données pour que le compte [denis@b63.com](mailto:denis@b63.com) fonctionne comme avant avec tous les objets qui y étaient présents. La nouvelle base données est à la racine du projet et se nomme « dcrlb63.sql ».