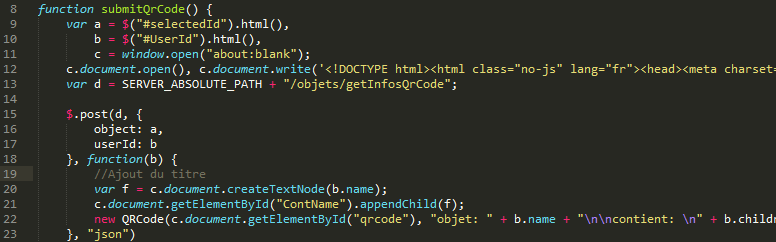
• **Ajouter le nom du conteneur sur l’étiquette du code QR. (Kevin)**

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 J-Query.

Une fois le code analyser, 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 étais déjà renvoyer par la requête SQL) en plus du code QR.



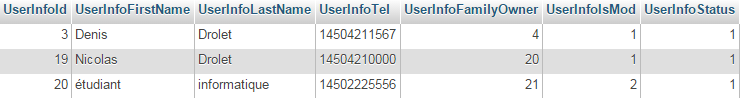
Exemple : j’ai ajouté une balise <h1> sur l’affichage à la ligne 12, et affecter une text node à la ligne 20/21

• **Page de login : S’inscrire enregistre les infos mais ne les affichent pas dans Comptes pour validation. (Kevin)**

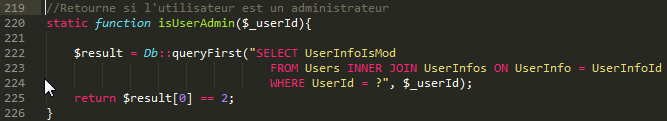
Le problème ici est qu’il n’y a aucun moyen d’activer un compte qui tente de s’inscrire pour l’administrateur du site web.

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.

• **Vérifier le contenu des rapports (Kevin)**

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

**Explication du système :**

**Fonctionnalité :**

**-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.

-**Nouvelle fonctionnalité**

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.

**Langage :**

Javascript :

PHP : Le PHP est le contrôleur de l’application, il s’occupe de décider quel 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 prèdéfinie par Foundation

SQL : Permet de faire les requêtes à la base de données via une connexion PDO pour obtenir les données de l’utilisateur.

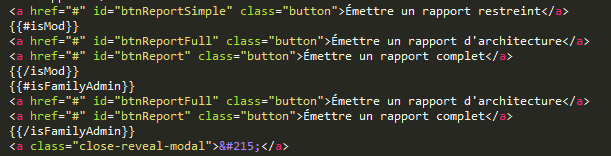
**Librairie :**

**-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 utilisé en 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**

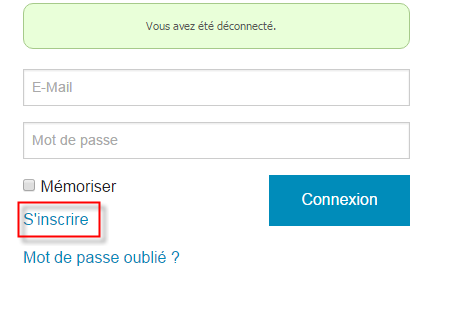
QrCode est une librairie très simple qui permet de changer une chaine de caractère vers un Code QR. Ici le code QR permet de sauvegarder un conteneur et son contenue dans un code QR et l’afficher à l’utilisateur.

-**Foundation**

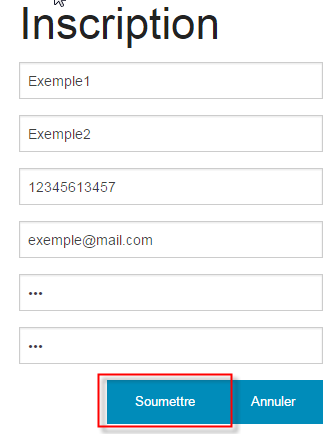
Foundation est un  « framework » se basant principalement sur le design et apparence des pages. I permet avec très peu d’effort de rendre le système élégant et plus simple d’utilisation.

**Utilisation de l’activation :**

1. **Premièrement l’utilisation s’inscrit**



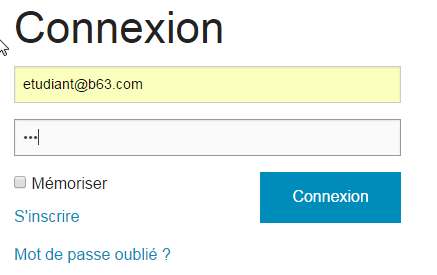
1. **L’utilisateur soumet son inscription**

****

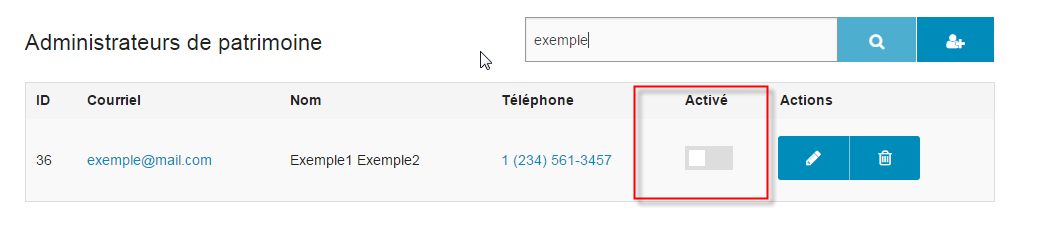
1. **Cependant l’utilisateur n’est pas encore activé lorsqu’il essaie de se connecter**

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

1. **L’admin doit donc se connecter**

****

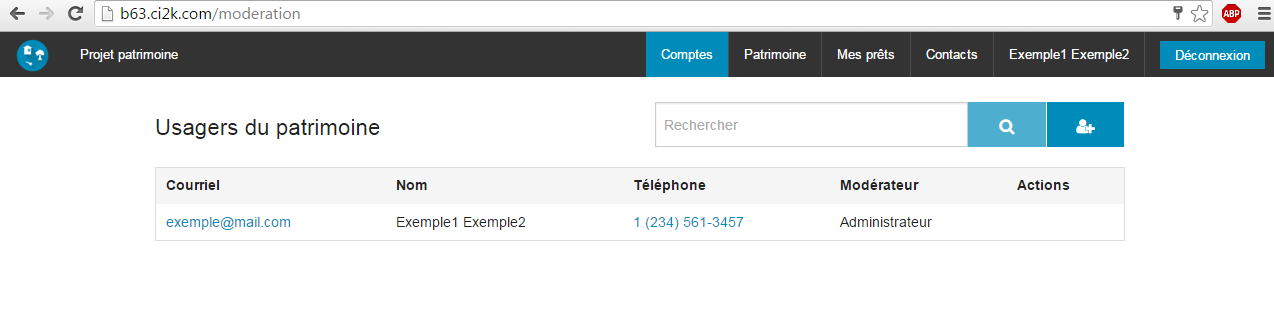
1. **Il a alors accès au panneau d’utilisateur et doit rechercher l’utilisateur**

****

1. **Il active l’utilisateur**

****

1. **L’utilisateur peut se connecté**

****