Samuel Anctil
Programmeur analyste

# Sprint retrospective
(Itération 1 - 30 mars 2017 au 30 avril 2017) pour
Gestion Licence, DLL et Service web

Document présenté au
Département de développement

MicroAge
27 avril 2017

# Sprint rétrospective (Itération 1 – 30 mars 2017 au 30 Avril 2017)

Devis de conception (30 mars 2017):
https://docs.google.com/document/d/1kj7row0W_ZRVD_W1NJu0nk2DcpZy12Q2u-HyB1qx1F0/edit?usp=sharing

Documents d'analyse (100% à jour avec le projet):
https://www.lucidchart.com/invitations/accept/df383138-0a25-46e2-bce0-b2bb29018194

**Quelques modifications par rapport au devis** :

- Ajout d'une colonne isActive dans les table contenant une clef primaire lié à une clef étrangère (nécessaire pour la suppression d'élément)
- La gestion des modules se fait via une page à part et non sur la page des logiciels dû à une contrainte de la bibliothèque editableTable (permet une seule grille par vue - pourra être modifié avec du temps de développement)
- Les options de filtrage et de recherches dans les grilles sont plus complexes que dans le devis pour offrir plus de flexibilité

***Remarques*** :

- Tous les données de la base de données sont conservées, il n'y a pas de delete en tant que tel sauf pour les licences car il a sa propre table pour l'historique (principe de base de la gestion de base de données)
- Un utilisateur qui a comme rôle '**Utilisateur'** n'a pas accès à la section '**Administration'**

***Serait intéressant à développer (pas dans le devis)*** :

- Utiliser l'architecture MVVC (Model - ViewModel - View - Controller) au lieux de MVC (Model - View - Controller) pour séparer complètement la couche logique de la couche UI et avoir plus de flexibilité (serait aussi plus cohérent avec le DatabaseManager mais n'empêche pas son fonctionnement)
- S'assurer que la couleur bleu claire reste lors du clic gauche de la souris dans une grille
- Intégrer plus d'option de recherche à certain endroit comme pour la gestion des modules, s'assurer également de pouvoir faire 'enter' pour confirmer la recherche

# Élément à développer

**Ce contenu devra être ajouté dans une itération ultérieur:**

1. Possibilité de cacher des colonnes dans les grilles
2. Hacher les mots de passe dans la base de données (question de renforcer la sécurité)
3. La validation des informations lors de l'update et de l'insert de données est implémenté mais les règles de validation n'y sont pas tous dans les modèles
4. Les barres de recherche ne sont pas implémenté partout
5. Faire la gestion des modules lors de la restauration de licence (ceci n'a pas encore été implémenté), une restauration de licence ne prend pas en compte les modules présentement
6. Permettre la modification de liste déroulante dans les grilles, nécessitera une condition dans le jQuery de editableTable.js pour vérifier si l'input est un select ou un input de type text.
7. Ajouter un système de pagination dans les grilles pour des raison de performances et d'affichage
8. L'affichage des date dans les grilles est incorrect

# Élément en cours de modification à terminer

1. Terminer la gestion des modules (tester les méthodes existante car elle n'ont pas été testé)
2. Peaufiner l'affichage des grille lors de la modification d'une cellule afin que les cellules reste toujours de la même largeur

# Bugs connus et non résolu (chemin **:** description**)**

1. **View/Licences/index** : Erreur d'affichage de la version lors de l'update d'un élément d'une ligne, serait supposé se mettre à jour seulement si la valeur change mais ça le fait même lorsque la valeur reste la même (n'affecte pas la base de données)
2. ***partout*** : La session se ferme après 20 minutes qu'il y est activité ou non, peut causer des erreurs car lorsque la session n'est plus active il est impossible de communiquer à la BD. Il suffit de se déconnecter et de se reconnecter manuellement via le menu de gauche pour régler le problème. Voir la variable HttpContext.Session (reste valide 20 minutes par défaut) dans **/helpers/sessionHelpers**
3. **Views/Licences/_Edit**: Gestion de la sélection des modules non fonctionnel dû aux valeurs des checkbox qui ne sont pas envoyé correctement au contrôleur. Voir ligne 70 à 80 dans la vue _Edit du dossier Licences.

```
var sessionName = new Byte[20];
bool isValid = ctrl.HttpContext.Session.TryGetValue("session", out sessionName);
```

# Notes sur les outils créés dans cette itération (au même titre que Bootstrap et JQuery)

Ceci est à titre informatif pour faciliter la compréhension de celui qui prendra en main le projet par la suite.

- DatabaseManager conçu en ASP.NET (méthodes statiques seulement : open, execute, close) permet de communiquer avec la BD (100% autonome, info vers la BD à changer)
    - **En lien directe avec le nom des propriétés des modèles, un modèle peut avoir des noms de propriété qui ne sont pas les mêmes que les noms de colonne dans la table liée si set { } n'est pas lié à la propriété (doit être readonly)**
    - DatabaseManager prend une classe modèle et utilise les propriétés de la classe pour communiquer avec la base de données, il crée d'A à Z la requête SQL et la renvoie à la BD puis créer et renvoi une List<T>, T étant le modèle si c'est un select. Renvoi le nombre d'élément affecté si c'est une autre requête que select comme (update, delete, etc)
    - **CONTRAINTES** : Gère les propriétés des modèles de type Guid, DateTime, Byte, SByte, Int16, UInt16, Int32, Uint32, Single, Double, Decimal, Boolean, String, Char et rien d'autre pour l'instant
    - Liste des définitions de fonction:
        - **public partial class DatabaseManager<T>**
        - Select
            - List<T> **select**(string[] columns, string filter = null)
            - List<T> **select**(string column, string filter = null)
            - List<T> **select**(string filter = null)
            - IEnumerable<SelectListItem> **selectList**(string idProperty, string textProperty, Guid selectedId, string filter = null)
            - IEnumerable<SelectListItem> **selectList**(string idProperty, string textProperty, string filter = null)
        - Update
            - void **update**(Dictionary<string, dynamic> columns, string filter = null)
            - void **updateAttribute**(string key, string value, string filter = null)
        - Insert
            - void **insert**(Dictionary<string, dynamic> newColumns)
        - Delete
            - void **delete**(string filter = null)
        - Autres
            - bool **checkIfExists**(string filters)
        - Spécifique à Gestion Licence! (**extention du DatabaseManager**)
            - T **selectSpecific**(Guid id)

                - void **updateSpecific**(Dictionary<string, dynamic> columns, Guid id)
                - void **updateSpecificAttribute**(string key, string value, Guid id)
                - void **updateSpecificAttributeIfNotExists**(string key, string value, Guid id)
                - void **deleteSpecific**(Guid id)
- Bibliothèque - UI (editableTable) conçu en JQuery / CSS (n'est pas autonome)
    - Permet la gestion de grille dynamique éditable à partir des view modèles
        - Gestion d'un (context menu) partiellement intégré (clique droit de la souris)
        - Gestion de la modification des valeurs dans les cellules intégrées
        - Inclus une fonction pour récupérer le model ASP.NET à partir d'une cellule
        - Inclus une fonction pour gérer les suppressions de ligne en AJAX
        - Inclus le CSS de base pour la grille
- Bibliothèque - UI (popup) conçu en JQuery / CSS / Bootstrap (autonome, copier-coller dans un autre projet et c'est prêt à fonctionner)
    - Permet la gestion de fenêtres modulaires
        - Fenêtre de dialogue Annuler/Ok
        - Fenêtre de dialogue Ok
        - Fenêtre modulaire avec vue personnalisable
        - Inclus le css de base pour les popup