Cégep Ste-Foy – Hiver 2014

Énoncé du Travail pratique 3

Préparé par Benjamin Lemelin

17 mars 2014

Résumé

Ce travail pratique consiste à réaliser en équipe de 2 ou de 3 un forum qui sera utilisé par des pirates informatiques.

1 Introduction au contexte du travail

Dernièrement, le groupe de pirates informatiques « Weak End », experts en instruisons de toutes sortes, vous a contacté afin de concevoir un forum à l'usage exclusif de leurs membres. N'ayant aucun autre contrat en vue, vous décidez donc d'accepter de les rencontrer.

Le groupe « Weak End » vous explique alors leurs besoins : ces derniers désirent un forum où des utilisateurs pourront s'y inscrire et s'y connecter afin de créer des sujets, de les consulter et d'y répondre. Pour l'instant, rien d'anormal, jusqu'à ce que l'un d'eux vous demande de porter une attention seulement minimale à la sécurité du site. Complètement ahuris, vous en demandez donc la raison : en fait, ces pirates *white hat*[1] désirent se procurer un terrain de jeu afin d'effectuer des démonstrations et d'entrainer leurs membres. Un peu hébété, mais content d'aider leur cause, vous acceptez donc immédiatement le contrat.

Maintenant, reste plus qu'à prendre de mauvais plis...en restant professionnel bien entendu.

2 Conditions de réalisation du travail

Ce travail d'une valeur de 12% de la note finale s'effectue en équipe de deux ou de trois. Chaque participant de l'équipe devra s'assurer d'effectuer une quantité de travail équitable. Il est bien entendu déconseillé de diviser les tâches directement : tout membre de l'équipe devrait normalement travailler sur tous les aspects du projet.

Contrairement au second travail pratique, les informations entrées sur le site web créé dans ce travail seront enregistrées. Vous devrez utiliser une base de données Microsoft Access.

Les équipes de trois auront plus de travail à effectuer. Veuillez consulter la section des fonctionnalités à ce sujet.

3 Objectifs pédagogiques

Ce travail permet de vérifier:

- la compréhension des formulaires, des contrôles de base et des validations
- la compréhension de l'utilisation des bases de données
- la compréhension de l'utilisation des pages maitresses
- l'application des standards de programmation.
- l'appréciation de l'ergonomie des sites web.

[1] Un *white hat* (en français : « chapeau blanc ») est un pirate éthique ou un expert en sécurité informatique qui réalise des tests d'intrusion et d'autres méthodes de test afin d'assurer la sécurité des systèmes d'information d'une organisation.

4 Fonctionnalités à implémenter

Tout comme dans le second travail pratique, les fonctionnalités de ce travail vous seront décrites sous la forme de « User Stories ». Notez que la « user story » 27 n'est à faire que si vous êtes en équipe de trois; une équipe de deux effectuant cette « user story » n'aura pas de bonus.

Voici donc, dans l'ordre de priorité, les « user stories » à compléter pour ce travail :

No	Description
1	En tant qu'administrateur, je veux que les utilisateurs puissent s'inscrire sur le forum à l'aide d'un pseudonyme, d'une adresse courriel, d'un mot de passe et d'un avatar.
2	En tant qu'administrateur, je considère que le pseudonyme, l'adresse courriel et le mot de passe de l'usager sont obligatoires.
3	En tant qu'administrateur, je veux que les images servant d'avatar pour les utilisateurs ne dépassent pas 180 pixels de haut par 180 pixels de large, lorsque stockés sur le serveur.
4	En tant qu'utilisateur, si je ne fournis pas d'image en guise d'avatar, je veux que l'on m'en assigne une par défaut.
5	En tant qu'utilisateur, je veux que l'on me confirme que mon inscription sur le site s'est bien effectuée.
6	En tant qu'administrateur, je veux que les utilisateurs puissent se connecter sur le forum à l'aide de leur adresse courriel et de leur mot de passe.
7	En tant qu'administrateur, je veux que les utilisateurs connectés puissent se déconnecter.
8	En tant qu'utilisateur, je veux pouvoir changer d'avatar.
9	En tant qu'utilisateur, je veux que l'on me confirme que mon avatar a bien été modifié.
10	En tant que DBA, je veux que tous les usagers soient regroupés dans une table et que leurs pseudonymes servent de clé primaire.
11	En tant qu'utilisateur, je veux pouvoir créer un sujet sur le forum auquel je définirai un titre et un premier message.
12	En tant qu'administrateur, je considère que les titres des sujets ainsi que le contenu du premier message sont obligatoires.
13	En tant qu'utilisateur, je veux être en mesure de consulter la liste des sujets, présentés à l'aide : • du titre du sujet • du pseudonyme de l'auteur • de la date de création
14	En tant qu'administrateur, je veux que seuls les usagers connectés sur le forum puissent créer un sujet.
15	En tant qu'utilisateur, je veux que l'on me redirige vers la page du sujet une fois ce dernier créé.
16	En tant que DBA, je veux que tous les sujets soient regroupés dans une table. Je veux qu'un numéro séquentiel serve de clé primaire. Je veux que cette table contienne une clé étrangère vers la table des usagers afin de savoir qui a créé le sujet.
17	En tant qu'utilisateur, je veux pouvoir ajouter un message dans un sujet existant.
18	En tant qu'utilisateur, je veux que le formulaire d'ajout de message soit situé tout en bas de la page contenant les messages du sujet.

19	En tant qu'administrateur, je considère que tous les messages ajoutés dans un sujet ne doivent pas être vides.
20	En tant qu'administrateur, je veux que seuls les usagers connectés sur le forum puissent ajouter un message dans un sujet existant.
21	En tant qu'utilisateur, je veux que l'on me confirme que mon message a bel et bien été ajouté en affichant un message en haut de la page du sujet.
22	En tant qu'utilisateur, je veux être en mesure de consulter un sujet.
23	En tant qu'utilisateur, je veux que les messages s'affichent dans l'ordre où ils ont été créés.
24	En tant qu'utilisateur, lorsque je consulte un sujet, je veux être en mesure de voir, à côté de chaque message : • la date d'écriture du message, sous forme de délai depuis sa création tel que « il y a 3 minutes » • le pseudonyme de l'auteur • l'avatar de l'auteur
25	En tant que DBA, je veux que tous les messages soient regroupés dans une table. Je veux qu'un numéro séquentiel serve de clé primaire. Je veux aussi que cette table contienne une clé étrangère vers la table des usagers et vers la table des sujets afin de savoir qui a écrit le message et dans quel sujet.
26	En tant qu'utilisateur, je veux que l'on me confirme que je suis connecté en affichant mon pseudonyme et mon avatar en haut de la page, à côté du bouton de déconnexion.
- Équipes de trois uniquement -	
27	En tant que membre du « Weak End », je ne veux pas trop simplifier la tâche à mes membres et je veux donc que les mots de passe sur la base de données soient cryptés et salés.

5 Contraintes supplémentaires

Voici la liste des contraintes supplémentaire pour ce travail :

- Vous devez utiliser obligatoirement des « RequiredValidator » pour les validations nécessitant de vérifier que des données sont entrées.
- Vous devez utiliser obligatoirement une « MasterPage » pour toutes les pages de votre site web.
- L'utilisation du contrôle « DataGridView » est interdite.

6 Évaluation

Le travail sera évalué sur les critères suivants :

Éléments	Pondération
Respect des User Stories • Tous implémentés (selon la taille de l'équipe) • Respect de ce qui est demandé • Ne bogue pas	60%
Qualité de la base de données • Utilisation de l'intégrité référentielle • Nom des champs et des tables significatif	10%
Qualité de l'interface de la confirmation • Charge visuelle minimale • Couleurs agréables • Professionnalisme (droit et ajusté)	10%
Qualité et clarté du code • Indentation impeccable • Nom de variables significatives • Facilité de lecture • Pas de commentaire inutile	15%
Professionnalisme de la remise : • À la date, à l'heure et à l'endroit mentionnés • Sous la forme demandée (fichier « zip » avec un nom particulier)	5%

Les points seront enlevés en fonction de la gravité de la faute. Par exemple, une équipe ayant remis un travail dont l'indentation est incorrecte dans tous les fichiers pourrait se voir imputer automatiquement 15% de la note. Au contraire, une équipe ayant remis un travail dont l'indentation est incorrecte pour certaines lignes seulement pourrait se voir imputer uniquement 5% de la note.

Bien entendu, il est impossible de descendre en dessous de 0% pour une catégorie donnée, sauf pour cas extrême. Un de ces cas pourrait être un travail dont seule l'interface a été remise et qu'aucun code côté serveur n'existe. Un tel travail pourrait se voir remettre la note de 25%, car seuls 25% du travail aurait été fait. Enfin, notez que tout travail plagié se mérite la note 0%.

7 Documents à réaliser et à remettre

Veuillez remettre un fichier « zip » contenant tous les fichiers nécessaires à l'utilisation de votre site web sous le nom de « ***NomPrénom1-NomPrénom2-NomPrénom3.zip*** ». Veuillez remettre votre fichier sur « LÉA » et assurez-vous de ne rien oublier.

8 Remarques et guide de conception

8.1 Comment vérifier la taille d'une image?

La 3e user story demande à ce que la taille des images utilisées en guise d'avatar soit validée. Pour ce faire, les librairies « .NET » offrent quelques solutions, mais la plus simple reste l'utilisation de la classe « `Image` » dans l'espace de nom « `System.Drawing` ».

Vous devez cependant faire attention à ne pas utiliser la classe « `Image` » de l'espace de nom « `System.Web.UI.WebControls` », qui est une autre classe avec le même nom. Pour utiliser la bonne classe, assurez-vous de préciser l'espace de nom avant le nom de la classe, ce qui nous donne « `System.Drawing.Image` ».

Bien que la classe « `System.Drawing.Image` » soit directement instanciable, il est fortement recommandé d'utiliser les méthodes statiques de la classe pour créer facilement des instances. Parmi ces méthodes, il existe la méthode « `System.Drawing.Image.FromStream(…)` » qui permet de charger une image à partir d'un flux de données tel qu'un fichier.

Cependant, l'image que nous désirons charger ne se trouve pas sur le disque dur, mais bien dans un contrôle « `FileUpload` ». Qu'à cela ne tienne : il est possible de charger une image à partir de n'importe quel flux de données, et fort heureusement, le contrôle « `FileUpload` » nous fournit une propriété nommée « `FileContent` » qui est en fait un flux de données brutes que nous pouvons utiliser.

Une fois votre image chargée, il ne vous reste plus qu'à utiliser les propriétés « `Height` » et « `Width` » pour effectuer vos validations.

Dernier conseil : entourez votre validation d'un bloc « try-catch », car il se pourrait que le flux fourni par le contrôle « `FileUpload` » ne soit pas une image, mais un autre type de fichier.

8.2 Comment calculer une durée facilement?

Les librairies « .NET » contiennent une classe dénommée « `TimeSpan` », représentant un délai. Cette classe peut donc fortement vous simplifier la vie pour la 24e user story. En tout premier lieu, il faut savoir que si vous soustrayez un objet de type « `DateTime` » à un autre, vous obtenez non pas un autre objet « `DateTime` », mais bien un « `TimeSpan` ». Cette technique vous permet donc facilement de calculer des durées.

Quant à l'affichage de ces objets « `TimeSpan` », c'est plus complexe. Par défaut, il vous est possible d'utiliser un format « `hh:mm:ss` » avec la méthode « `toString()` ». Cependant, ce format n'est pas très intéressant pour un affichage sur un site web. Fort heureusement encore, il est possible de personnaliser ce format en utilisant la méthode « `toString()` » prenant en paramètre un format sous forme de « `string` ».

Créer un format n'est pas réellement compliqué. Il faut tout d'abord comprendre qu'un format est en fait un modèle que l'objet « `TimeSpan` » remplira avec les données demandées. Dans notre cas, en utilisant un format tel que « `"'Il y a '%d' jours'"` », nous demandons à l'objet « `TimeSpan` » d'inscrire les

jours dans la chaine de caractère de retour en plus des mots qu'il y a autour. Notez d'ailleurs les guillemets simples entourant les mots. Utilisez « `%d` » pour les jours, « `%h` » pour les heures, « `%m` » pour les minutes et « `%s` » pour les secondes.

8.3 Comment faire l'architecture de la base de données?

Faire une base de données est beaucoup plus complexe qu'il ne parait à première vue : le choix des types de données, des liens entre les tables et le choix des clés primaires et étrangères nécessitent à eux seuls un cours complet pour les maitriser entièrement. C'est donc dans ce contexte qu'a été écrit ce petit guide afin de vous simplifier la tâche.

Votre application contient trois groupes de données :

- les utilisateurs
- les sujets
- les messages

Déjà, vous devriez vous rendre compte que vous avez besoin de trois tables pour contenir ces données. Vous aurez donc une table pour les utilisateurs, une table pour les sujets et une table pour les messages. Vous pouvez déjà les créer dans Access avec leurs principaux champs. Par convention, nommez vos tables et vos champs en utilisant le « snake_case ».

Une fois fait, il faut réfléchir aux liens entre ces tables. Intuitivement, la question à se poser serait « qui possède qui ? ». Par exemple, vous pourriez dire qu'un sujet possède plusieurs messages. Or, ceci n'est pas directement transposable dans Access, car étant sous une base de données dite « relationnelle », il n'existe pas de champs de type liste. Il faut donc inverser la relation en se demandant plutôt « qui appartient à qui ? ». En suivant le même exemple, on dira donc qu'un message appartient à un sujet, indiquant donc que la table des messages contiendra un champ référençant le sujet auquel chaque message appartient.[2]

Les tables et les relations étant complétées, votre base de données est déjà prête à fonctionner. Pour les plus entrepreneurs d'entre vous, demandez-vous à présent quels sont les champs qui ne peuvent pas être nuls et appliquez des règles en conséquence. Dans certains cas, vous pouvez même vous demander si une valeur par défaut ne serait pas possible afin de simplifier les ajouts dans la base de données. Les valeurs par défaut sont particulièrement utiles pour les champs contenant des dates de création. Dans Access, utilisez « `=Maintenant()` » en guise de valeur par défaut pour des champs de ce type.

8.4 Comment saler et crypter des mots de passe?

Les équipes de trois doivent crypter les mots de passe enregistrés dans leur base de données afin de donner plus de fil à retordre aux membres du « Weak End ». En fait, la méthode présentée ici est réputée comme étant inviolable et c'est d'ailleurs celle qui est utilisée par les plus grandes organisations telles que « Google », « Microsoft » et désormais « Sony ».

---

[2] Lorsque vous créerez dans Access vos relations entre les tables, assurez-vous d'activer l'intégrité référentielle.

Pour tout comprendre, il s'impose tout d'abord de présenter les méthodes possibles de sauvegarde des mots de passe, la première étant, bien entendu, l'absence complète de cryptage. Nul besoin de mentionner que cette méthode n'est pas sécuritaire si quelqu'un s'empare de la base de données.

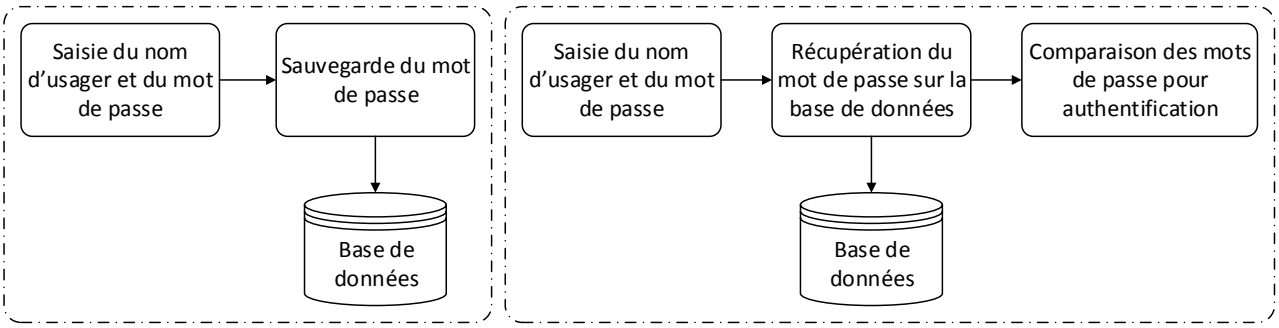

Figure 1 - Séquence lors de l'inscription et de l'authentification pour la première méthode

La seconde méthode est déjà beaucoup mieux : il s'agit de crypter le mot de passe à l'aide d'un algorithme de cryptage bidirectionnel. Cet algorithme utilise une clé pour fonctionner et cette clé est stockée sur le serveur. Un tel algorithme rend alors le mot de passe sur la base de données illisible. Cependant, si la clé utilisée est volée, alors tous les mots de passe peuvent être décryptés très facilement.

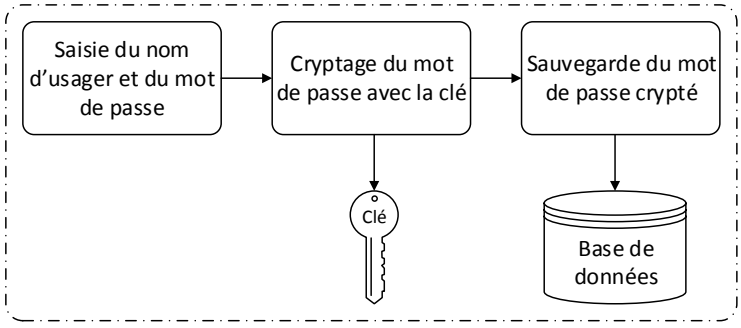

Figure 2 - Séquence lors de l'inscription pour la seconde méthode

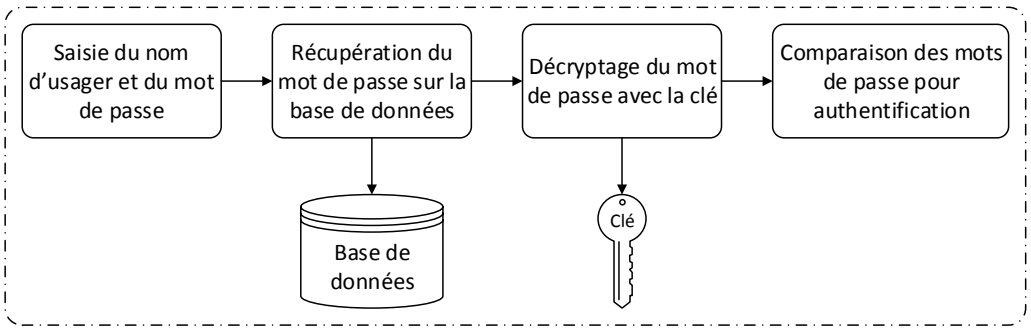

Figure 3 - Séquence lors de l'authentification pour la seconde méthode

La troisième méthode consiste à utiliser un algorithme de cryptage unidirectionnel, aussi appelé algorithme de hachage. Avec ces derniers, il est impossible de retrouver le mot d'origine à l'aide du hash produit étant donné que les éléments à crypter sont leurs propres clés. Ces algorithmes ont aussi la particularité de donner le même hash pour le même mot de passe.

Pour cette troisième méthode, les hash des mots de passe sont stockés sur la base de données en guise de mot de passe et donc, afin de savoir si un mot de passe est bon, il suffit de comparer le hash du mot de passe entré par l'utilisateur avec le hash conservé sur la base de données.

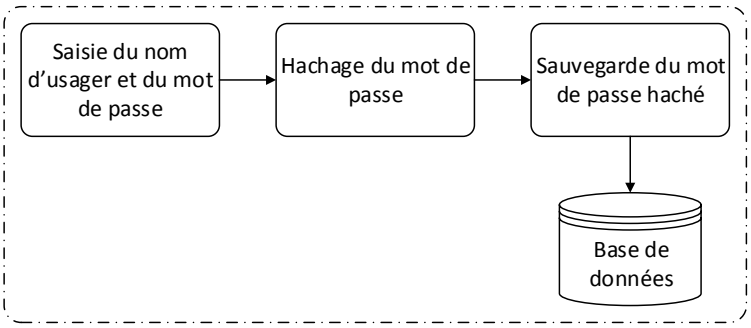

Figure 4 - Séquence lors de l'inscription pour la troisième méthode

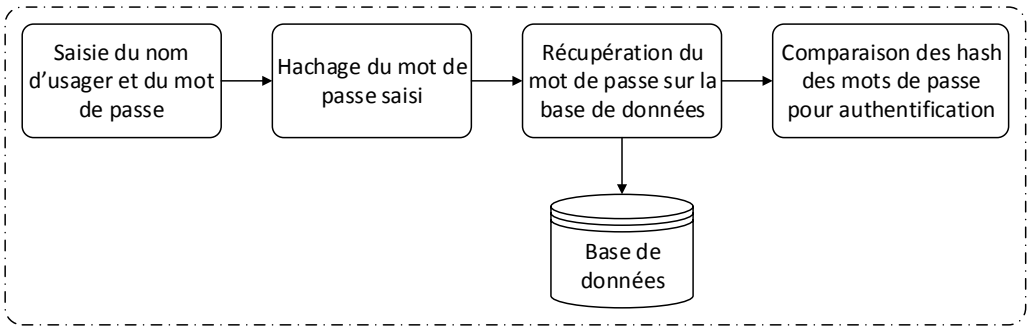

Figure 5 - Séquence lors de l'authentification pour la troisième méthode

Nombre d'experts en sécurité pensaient alors avoir trouvé une solution inviolable. Cependant, les pirates ont trouvé une méthode de contournement : nombre d'ordinateurs infectés par des bots produisent aujourd'hui ce qui est appelé des « Lookup Tables », permettant de retrouver un mot de passe à partir de son « hash ». L'internet a donc servir littéralement de supercalculateur à hash et ces tables permettent maintenant de retrouver pratiquement n'importe quel mot de passe à partir de son hash, et ce, pour nombre d'algorithmes existants.

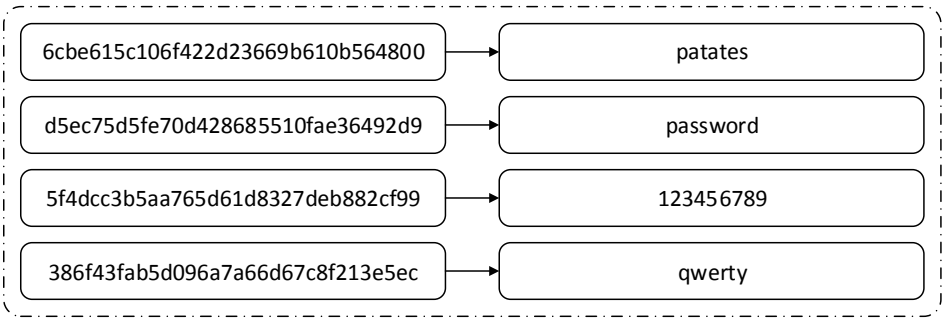

Figure 6 - Exemple de « Lookup Table »

Que faire alors? En fait, le gros problème vient du fait que le même mot de passe produira nécessairement le même hash. Il faut donc trouver un moyen pour que ces mots de passe ne produisent jamais le même hash, rendant ainsi les « Lookup Tables » inefficaces.

C'est à ce moment-là que le salage des mots de passe entre en jeu. En ajoutant une série de chiffres et de lettres complètement aléatoires au mot de passe, ce que l'on appellera le sel, nous obtenons alors un nouveau mot de passe que nous pouvons hacher et stocker dans la base de données avec le sel. Ce faisant, les pirates sont donc obligés de générer une « Lookup Table » de mots de passe pour chaque sel possible, ce qui est alors impensable compte tenu du nombre de possibilités.

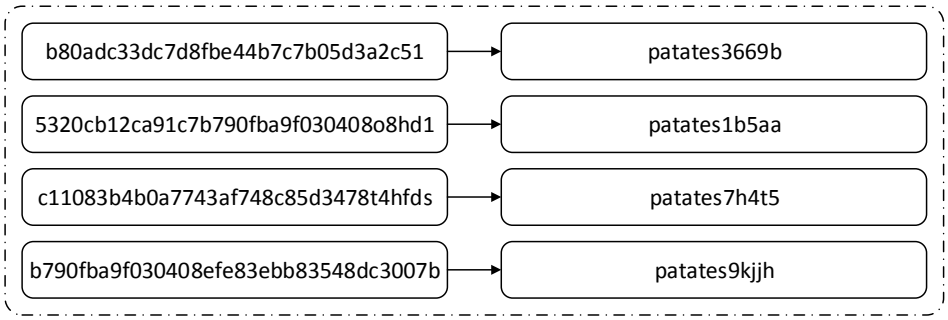

Figure 7 - Exemple montrant l'inefficacité des « Lookup Table » après salage des mots de passe

Pour cette dernière méthode, les mots de passe salés et hachés sont stockés dans la base de données avec le sel. Lors d'une demande de connexion, le mot de passe fourni est salé avec le sel stocké dans la base de données. Ce nouveau mot de passe est en suite haché pour être comparé au hash stocké en guise de mot de passe dans la base de données. Si les deux hash sont égaux, cela veut donc dire que les mots de passe sont égaux.

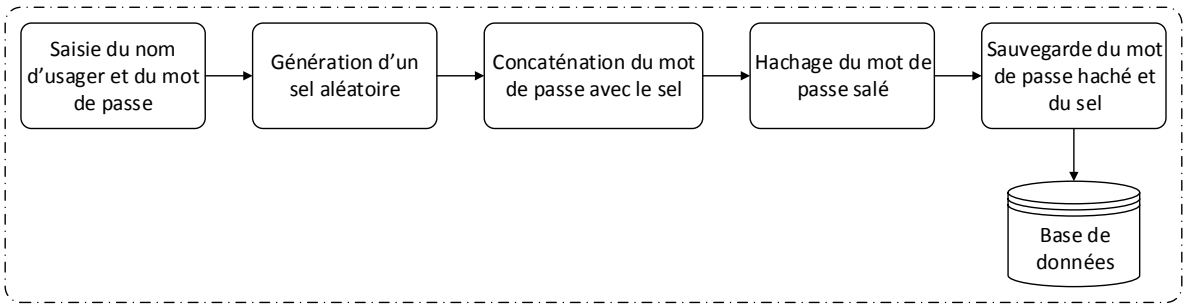

Figure 8 - Séquence lors de l'inscription pour la dernière méthode

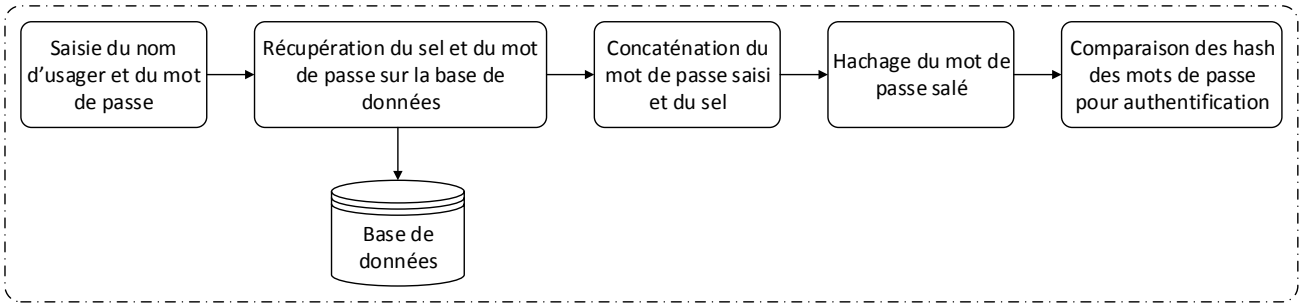

Figure 9 - Séquence lors de l'authentification pour la dernière méthode

C'est cette solution qui vous est demandée d'implémenter dans ce travail. Pour ce faire, le cadriciel .NET offre plusieurs outils. En premier lieu, la classe « `RNGCryptoServiceProvider` » dans l'espace de noms « `System.Security.Cryptography` » permet d'obtenir des sels complètement aléatoires. Vous devez obtenir de cette classe un tableau de « `byte` » à l'aide de la méthode « `GetBytes` » que vous

enregistrerez dans votre base de données avec les données de l'utilisateur. Dans Access, le type de données à utiliser est le « OLE Object ». Pour cette implémentation, assurez-vous que le sel est d'une longueur de 24 éléments.

La classe « `Rfc2898DeriveBytes` » dans l'espace de noms « `System.Security.Cryptography` » permet quant à elle de hacher un mot de passe et son sel. Cette classe demande, lors de sa construction, le mot de passe, le sel et le nombre d'itérations pour le hachage. Le nombre d'itérations doit être d'au moins 1000. Pour obtenir le hash à placer dans la base de données en guise de mot de passe, utilisez la méthode « `GetBytes` » en fournissant 24 comme longueur en paramètre.

Ce sont les deux seules classes nécessaires à l'implémentation de cette technique sur un serveur ASP.NET. Pour plus d'informations, prière de me contacter.