Projet Web

**M1 MIAGE - 2014**

Pavinich Christian & Siddi Steven



Table des matières

[1. Introduction 2](#_Toc389682269)

[2. Thème et template 2](#_Toc389682270)

[3. Base de données 3](#_Toc389682271)

[4. Servlets 4](#_Toc389682272)

[5. Inscription, connexion et profil 4](#_Toc389682273)

[6. Album et upload de photo 4](#_Toc389682274)

[7. Photo 6](#_Toc389682275)

[8. Commentaires 7](#_Toc389682276)

[9. Notifications et Actualité 7](#_Toc389682277)

[10. Barre de recherche 8](#_Toc389682278)

[11. Traduction & AngularJS 9](#_Toc389682279)

[12. Page d’erreur 404 9](#_Toc389682280)

[13. Points forts, points faibles et problèmes rencontrés 10](#_Toc389682281)

[14. Gestionnaire de versions 11](#_Toc389682282)

[15. Conclusion 11](#_Toc389682283)

# 1. Introduction

Dans le cadre du module de programmation Web du semestre 8 de Master 1 MIAGE, nous avions pour but la réalisation d’un site social de gestion de photo en Java EE. Les utilisateurs de ce site devaient pouvoir créer des albums contenant des photos. Les albums sont publics ou privés, et peuvent être partagés avec les autres utilisateurs de la plateforme.

Un système de commentaires est en place, permettant aux utilisateurs de donner leurs avis sur une photo, ou plus globalement, sur un album. Des notifications sont aussi envoyées aux utilisateurs pour les tenir informés.

Nous avons décidé de partir sur un site parodiant Instagram, appelé Instamiam, et dont le principal but est le partage de photos de nourriture. En effet, Instagram est bien connu pour abriter un nombre incroyable de photos des repas de ses utilisateurs, détournant quelque peu le principe du site.

# 2. Thème et template

En partant de cette idée, le thème visuel tourne autour de couleurs vives et sucrées comme le violet et le vert rappelant viennoiseries et pâtisseries. On remarquera aussi une police de type manuscrite cursive, donnant un effet “artistique” ou fait main.



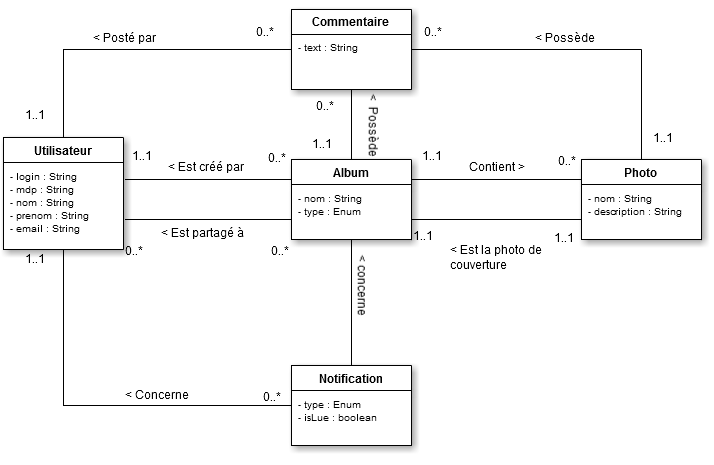
Le thème a était réalisé en récupérant 3 images d’un thème préexistant : l’image de fond et les images des bordures du header et du footer, depuis le thème Cupcakery trouvable sur le site Themeforest.net. Malgré cela, aucun fichier CSS n’a était gardé du thème de base. Nous avons, en nous servant de bootsrap, écrit le CSS de notre site à la main afin de le faire correspondre pleinement à nos besoins.

Le template du site est décomposé en 3 parties :

* Le header, contenant le logo, la barre de recherche et le menu.
* Le content, variant suivant le servlet qui traite l’information.
* Le footer, contenant nos noms, des liens vers nos sites personnels et les boutons permettant de changer la langue du site vers le français ou l’anglais.

# 3. Base de données

Nous avons utilisé les technologies EJB et JPA afin d’utiliser et d’organiser notre base de données. Elle est composée des entités suivantes :



Un utilisateur possède ainsi des albums, contenant les photos, et une liste de notifications qui lui est destinée (ajout d’une nouvelle photo dans un album partagé par exemple). Un album est lié à une liste de commentaires, mais aussi la liste des utilisateurs à qui celui-ci est partagé, ainsi qu’un type (album public ou privé). Un album possède des photos, qui sont elles aussi liées à des commentaires postés par des utilisateurs.

# 4. Servlets

Une classe abstraite, appelée SuperServletVerification, a était créé afin de factoriser le code de vérification des autres servlets du projet. Ainsi, tout servlet héritant de SuperServletVerification pourra invoquer le processRequest de sa classe mère et ainsi directement gérer le cas d’un utilisateur souhaitant accéder à une page nécessitant une connexion préalable.

C’est aussi dans ce servlet qu’on récupère le nombre de notifications à afficher comme une petite vignette à côté de l’onglet Actualités du menu du site.

# 5. Inscription, connexion et profil

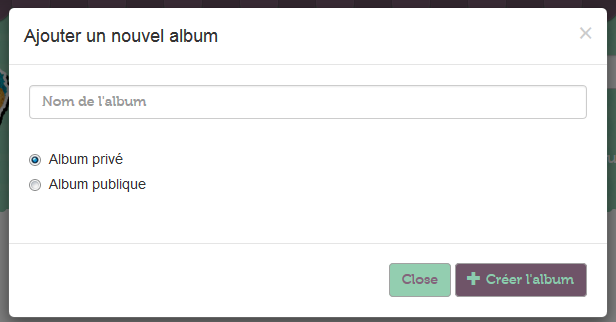
Pour pouvoir se connecter au site, il faut bien évidemment un compte. Dans la partie inscription, il est ainsi possible à un utilisateur de créer son compte en choisissant login, mot de passe, et en indiquant des informations personnelles (nom, prénom, email).

Une fois connecté, l’internaute peut aller dans sa page de profil et gérer ses informations. Il peut aussi changer l’image de profil par défaut.

# 

# 6. Album et upload de photo

Avant de pouvoir partager ses photos de paella avec ses amis, l’utilisateur va devoir créer un album dans la page “Mes albums”, en cliquant sur le bouton “Nouvel album”.



*Fenêtre modale de création d’album*

Une fois l’album crée, on peut se rendre sur ce dernier et ajouter des photos en les déposant dans la zone prévue à cet effet. Cette zone permettant le glissé/déposé de fichier a été mise en place grâce à la librairie DropzoneJs, permettant de définir les callback sur les différentes actions de la zone.

À l’aide d’un id de transaction généré aléatoirement, nous avons pu mettre en place un upload image par image en ajax des fichiers, à mesure que ceux-ci sont déposés. Les différents fichiers se retrouvent ainsi au fur et à mesure dans un répertoire dont le nom est l’id de la transaction. Les différents id ont été générés avec la classe java UUID. De plus, de la même façon, chaque image voit son nom de fichier remplacé par un ID unique généré avec la classe UUID. Ceci permet notamment d’empêcher les utilisateurs indésirables d’accéder aux images trop facilement en ayant une url quasiment impossible à deviner.

Une fois tous les fichiers voulus présents dans la zone, on va valider l’ajout en cliquant sur le bouton “Envoyer”. Côté serveur, les images étant déjà uploadées avant la confirmation, elles sont tout simplement déplacées dans le répertoire de l’album. Les entrées dans la base de données sont créées pour lier le fichier à une entité photo, et la placer dans l’album.

Sur la page de l’album, on peut supprimer l’album ou le modifier. Dans ce dernier cas, on peut redéfinir le nom de l’album, tout en ajouter/supprimer les personnes à qui ont a partagé notre album.



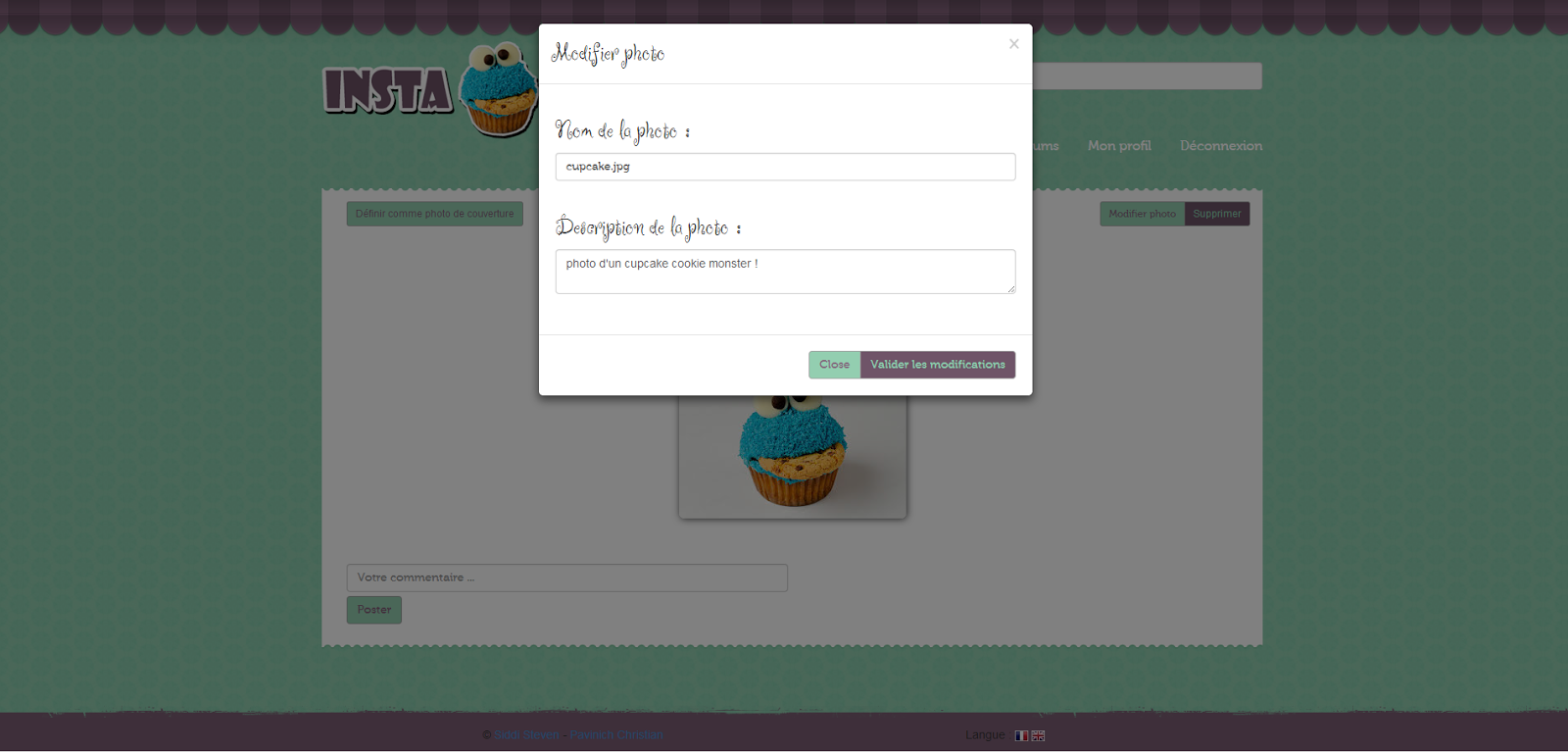
*Fenêtre modale de modification d’album*

Dans l’encadré “Partager l’album”, une autocomplétion a été réalisée : il suffit ainsi de saisir quelques lettres du nom ou du prénom d’un utilisateur pour afficher la liste des correspondances, et simplement sélectionner la personne désirée. La croix de suppression à côté des noms des utilisateurs permet, en ajax, de retirer le partage de l’album à la personne.

Enfin, sur cette page se trouvent les commentaires de l’album, ainsi que les différentes photos. Au survol de celles-ci, une fenêtre d’information apparait afin de spécifier le nom de la photo, sa description, et le nombre de commentaires.

# 7. Photo

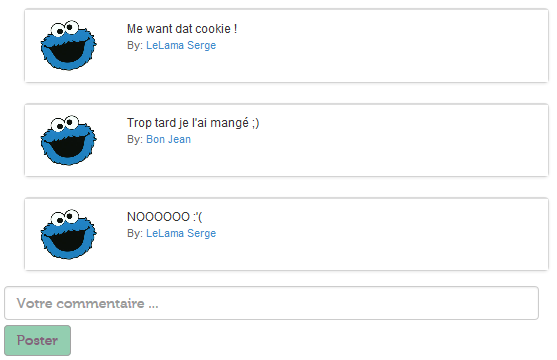
Depuis un album, en cliquant sur la vignette d’une photo, on arrive sur la page de celle-ci, composée de l’image en grand, de la partie des commentaires, d’un bouton pour la supprimer, d’un bouton pour modifier le nom de l’image et sa description, et enfin d’un bouton pour définir cette photo comme étant la photo de couverture de l’album.



*Fenêtre modal de modification de photo*

# 8. Commentaires

La zone de commentaires (d’un album ou d’une photo) permet d’afficher la liste des commentaires liée. Chaque commentaire affiche ainsi la photo de profil de l’auteur, son commentaire et son nom cliquable pour aller directement sur la liste de ses albums.



# 9. Notifications et Actualité

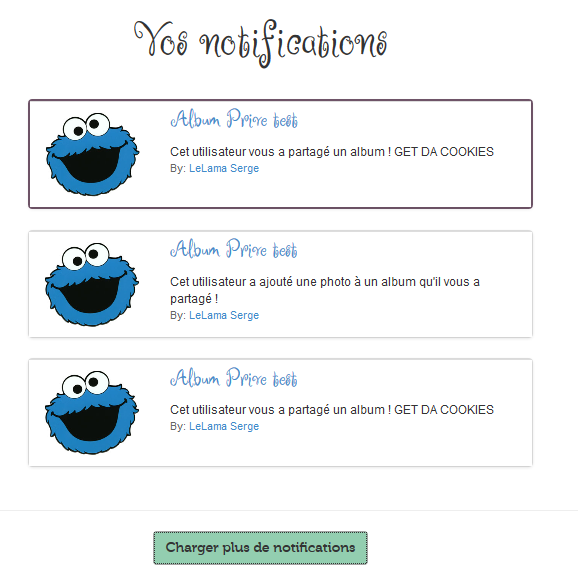
Un système de notification a été mis en place pour prévenir les utilisateurs de certaines actions : Partage d’un album, ou ajout d’une photo dans un album partagé précédemment, voire encore l’ajout d’un commentaire sur une photo ou un album.

Le nombre de nouvelles notifications apparait sous forme de puce dans la barre de menu, sur l’onglet actualité.



*Onglet “Actualités” avec 5 nouvelles notifications*

Sur la page actualité, les dernières notifications de l’utilisateur apparaissent. Les notifications non lues possèdent un cadre plus foncé pour les mettre en avant. Un bouton permet de charger en ajax les notifications les plus anciennes. Une fois une notification lue, celle-ci change d’état et ne possédera plus un cadre plus foncé la prochaine fois que l’utilisateur la regardera.



*Page d’actualités avec les notifications de l’utilisateur*

# 10. Barre de recherche

Une barre de recherche est également présente et permet à un utilisateur de rechercher un autre utilisateur afin d’accéder aux albums publics et partagés de ce dernier. Bien évidemment, les fonctionnalités de modification et de suppression sont désactivées pour l’utilisateur qui “visite”.

On peut également noter que la barre de recherche est de type autocomplétion, identique à celle présente pour partager un album à d’autres utilisateurs.

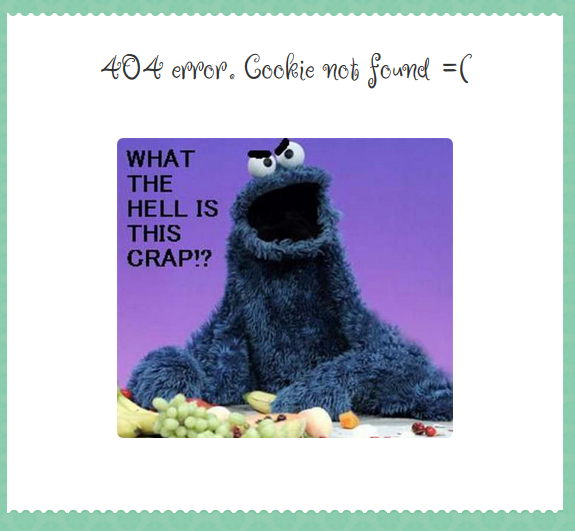
# 11. Traduction & AngularJS

Enfin, en exploitant le module ngSanitize d’AngularJS, nous avons mis en place un système de changement de langue de tous les libellés sur chaque page et ce, sans rechargement de la page.

Le fonctionnement est simple : chaque libellé correspond à un index d’un tableau JSon crée dans le contrôleur principal Angular, et selon la langue choisie on injecte dans ce tableau un autre tableau correspondant à la langue où les noms d’index sont identiques mais le contenu adapté. Une fois ceci fait, angular se charge du rechargement des phrases affichées.  
 Seulement le français et l’anglais sont disponibles pour le moment.

# 12. Page d’erreur 404

Nous avons configuré le fichier web.xml afin de transmettre toutes les erreurs 404 vers une page personnalisée.



# 13. Points forts, points faibles et problèmes rencontrés

Nous pouvons dire que nous sommes heureux de certaines fonctionnalités de notre projet que nous ne pensions pas implémenter aussi bien dans ce projet.

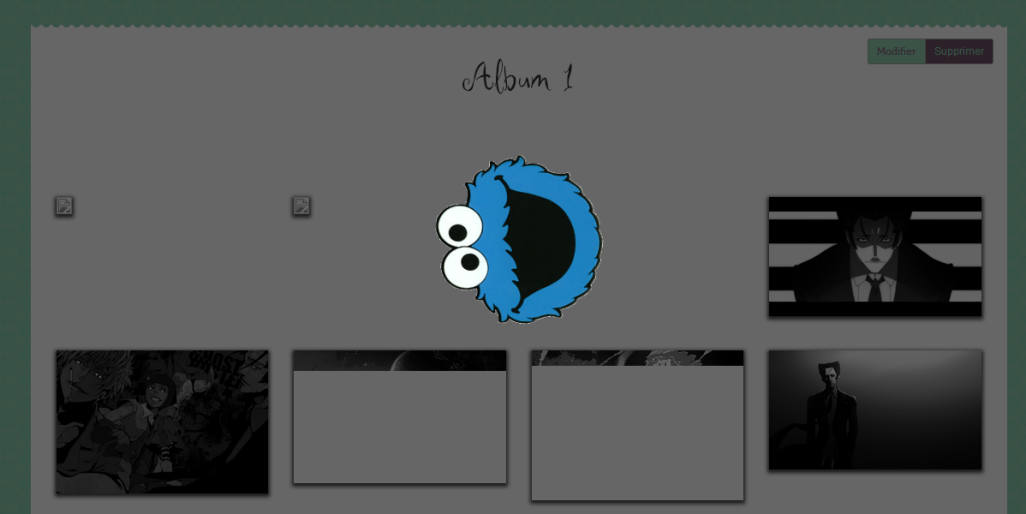
Tout d’abord, en démarrant le projet, notre niveau en CSS3 ne nous permettait pas d’avoir toute la flexibilité nécessaire pour la réalisation de ce que nous imaginions, surtout sans acheter un thème déjà existant. De ce fait, nous avons appris à mieux utiliser Boostrap mais aussi suivi des tutoriels sur le CSS3 pour réussir à construire un thème par nous-mêmes. Nous avons ainsi fait d’énormes progrès en CSS et sommes très satisfaits du design global de notre site, entièrement responsive.

Ensuite, avec l’utilisation d’Ajax et d’AngularJs, nous avons pu réaliser un projet globalement ergonomique avec le minimum de chargement de pages possible (partage d’album, changement de langue, etc.).

Nous avons bien sûr rencontré des difficultés lors de la réalisation de ce projet, outre le manque de niveau en CSS à rattraper : avec une expérience limitée aux TP avec les bases de données JPQL, nous avons bloqué pendant de longues périodes sur des problèmes de base de données où l’on n’arrivait pas à mettre en place les liens entre les entités. Ensuite, le serveur Glassfish s’est révélé très complexe à débugger : les indications utiles sur les erreurs sont systématiquement noyées dans un torrent d’exceptions serveur et souvent le projet refusait de se lancer tout court.

Enfin, un autre problème de taille était le traitement côté serveur de l’upload de multiples photos. Un upload de moins de 5 photos est instantané, mais lors de l’upload de plus de 5 photos, le chargement de la page pour afficher les nouvelles images devient extrêmement long (30 secondes, parfois plus) alors que toutes les images étaient bien placées dans le répertoire de l’album, et que la base de données était bien à jour. La JSP était bien renvoyée mais, au milieu de son chargement, le serveur semble être en surcharge et ne répond plus pendant une trentaine de secondes.

Une solution a été mise en place pour “forcer” l’utilisateur à patienter pendant le chargement avec un dispositif Javascript permettant de bloquer la page en insérant une image de chargement. Nous avons finalement identifié le problème comme venant de la configuration de glassfish qui ne dispose pas d’assez d’espace en mémoire vive pour traiter le déplacer de plus de 5 images depuis le dossier temporaire vers le dossier de l’album, ce qui entraine les différents threads java à se bloquer et à rendre le serveur incapable de traiter des demandes pendant quelques secondes.



*Image d’attente lors d’un upload de plus de 5 photos.*

# 14. Gestionnaire de versions

Pour la réalisation de ce projet, nous nous sommes servis d’un Git public hébergé sur la plateforme GitHub, et des logiciels GitHub et SourceTree.

Notre repository est trouvable à l’adresse suivante <https://github.com/jokho26/InstaMiam>

Comme il est visible sur le site, nous avons effectué des commits réguliers durant deux mois, sauf deux semaines à la fin du mois d’avril.

À titre d’information, le pseudo Synthr4x correspond à Christian Pavinich et jokho26 à Steven Siddi.

# 15. Conclusion

Pour conclure, nous pouvons dire que toutes les fonctionnalités demandées par le sujet ont été réalisées. Les nombreuses notions apprises lors de ce développement nous serviront sans aucun doute lors de projets futurs de Web, que ce soit avec JEE ou non.