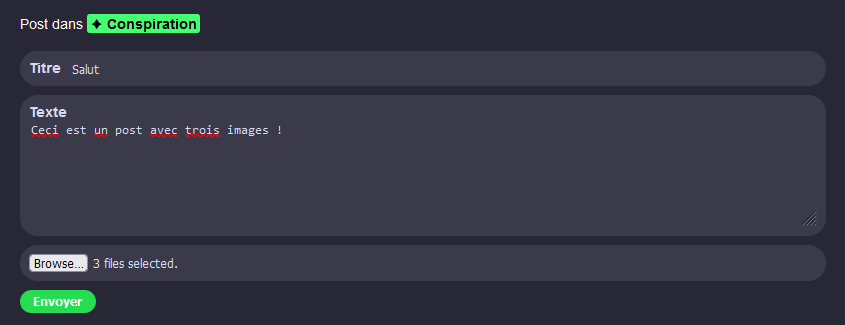
# Fonctionnalités Yang

Ce document donne des indices, mais bien entendu, il n’est ni exhaustif, ni parfait.

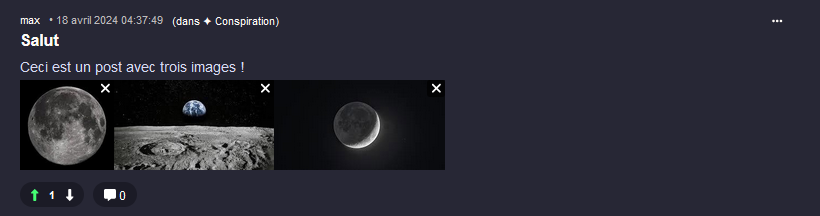
Étape C

« Lorsqu’on crée un post, on doit être capable d’y joindre une ou plusieurs images. »

* Cela se passe surtout dans le composant **EditPostComponent**. (Qui permet de créer un post) Pour créer un post, choisissez un hub spécifique d’abord. (S’il n’y a aucun hub, créez un hub)

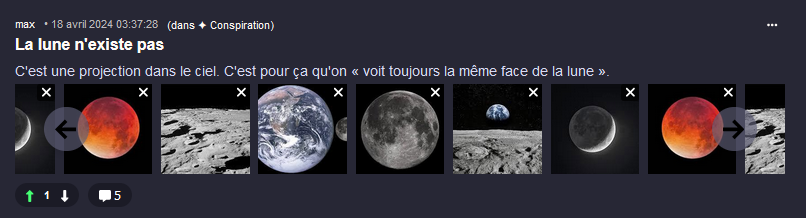


* Dans les notes de cours, on a seulement vu comment joindre une image à un input pour envoyer une seule image au serveur. Dans ce cas, ce sera un peu différent. Ce problème se règle entre autres avec une boucle while. (Vous pourriez donner à chaque image l’étiquette "image0", "image1", "image2", etc. dans le formData)
* Le serveur reçoit un **CommentDTO** dans le corps de la requête actuellement pour créer un nouveau commentaire… ce ne sera plus possible car on va maintenant envoyer un **formData**. Il faudra remplacer le DTO par un formData.
* Gardez bien à l’esprit que le message sous le titre du Post est un Comment. (Ce n’est pas le Post qui contient le texte. Un post, c’est juste un titre !) Ce commentaire est qualifié de « commentaire principal du post » dans le code et dans l’énoncé.
* N’oubliez pas d’afficher les images dans le composant **PostComponent**. Pour y arriver, il faudra que le client Angular reçoivent la liste des ids des Pictures du commentaire.
  + Gardez à l’esprit que le serveur n’envoie pas de **Comment.cs** à Angular, mais bien des **CommentDisplayDTO.cs** !

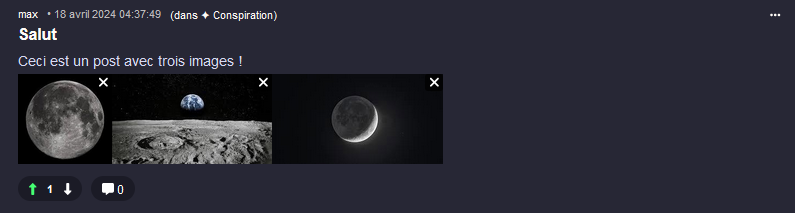


Étape D

« Si un post (commentaire principal d’un post) contient plus de 4 images, les images sont affichées avec un carrousel glidejs plutôt qu’en simple rangée. Les commentaires d’un post n’ont jamais de carrousel. »



* Pour le carrousel, il y aura des ajouts à faire dans le composant **PostComponent**.
* Le seul endroit où il est obligatoire d’afficher un carrousel est dans **PostComponent**, pour le commentaire principal d’un post avec plus de 4 images. Avec 4 images ou moins, on affiche simplement les images normalement :

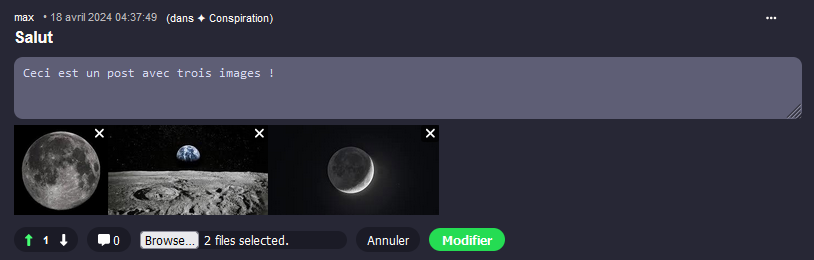


Étape E

« Lorsqu’on modifie un commentaire ou un post, on doit pouvoir ajouter une ou plusieurs images supplémentaires en même temps de modifier le texte. (Les anciennes images du commentaire, s’il y en avait, sont préservées) »

* Vous allez vous inspirer largement des actions PostPost / PostComment pour cette étape. N’oubliez pas qu’il y a deux endroits où on peut modifier un commentaire : dans **PostComponent** et dans **CommentComponent**.
* Encore une fois, le formData va remplacer tout DTO qui était précédemment utilisé.

On voit, ci-dessous, que je modifie mon message et que j’ai ajouté 2 images dans l’input.



Étape F

« Lorsqu’on supprime un commentaire ou un post, toutes ses images doivent être supprimées. »

* Vous remarquerez que l’action **DeleteComment** fait peur. C’est parce qu’elle hard-supprime également tout sous-commentaires ou commentaires parents qui étaient déjà soft-supprimés. Ne vous inquiétez pas trop avec cela : votre objectif est seulement de supprimer les images du commentaire cliqué s’il en avait.
* Les images du commentaire supprimées doivent être cachées immédiatement dans la page Web.

Étape G

« Les utilisateurs doivent pouvoir se connecter en utilisant leur nom d’utilisateur OU leur adresse courriel. (Plutôt que seulement leur nom d’utilisateur) Les utilisateurs doivent pouvoir changer leur mot de passe. »

* Il faudra beaucoup chercher ailleurs que dans les notes de cours pour cette étape, mais ce n’est pas très compliqué. Tout se passe dans le composant **ProfilComponent** et dans **UsersController**.

Étape H

« Un rôle administrateur existe. Les administrateurs peuvent ajouter le rôle modérateur à des utilisateurs. Ils peuvent supprimer les commentaires de leur choix via cette liste. Un utilisateur avec le rôle administrateur est ajouté dans le seed. »



* Ajoutez un mini formulaire supplémentaire seulement visible pour les administrateurs. Ça peut être dans **ProfileCompnent**.