## Discussions

## Upload and store photo

We attempted to store profile photo of user user in the backend by using “user id + file extension” as the filename, assuming each user will has at most one profile photo. In order to do so, we need to get the extension of the image file. However, we found the we the backend play framework receive image on the HTTP multibody part, the filename and the extension of that file will be lost, and changed to something like “multipartBody1975347307109665184asTemporaryFile”, that means we cannot get the type of the image form the filename.

Solution:

We found that we can get the type of the image directly using the method getContentType, and the return value, for example, is “image/png”, which can be used to display the image on the response. Therefore problem solved.

## Stack overflow when converting object to JSON

We encountered stack overflow error when we tried to convert post and user object to JSON object. The error message shown in the terminal seems to be a infinite loop. We found that because of the many-to-many relation between user-user and user-post, every time a user object is being converted to JSON object, the users he/she following, and the users who following him/her, and the posts he/she created and shared will be included in the JSON object. Therefore, in some cases, for example, two users, user 1 and user 2, follow each other, if we want to convert user 1 to JSON object, user 2 will also be included, and because user 2 need to be converted, user 1 will be converted again … and thus, an infinite loop is formed.

Solution:

We add some annotation to the variables of the models and use customized exclusion strategy during conversion , to ensure that there would not be any infinite during conversion process.

**Upload profile image from front-end directly**

We tried to show the image on the front-end directly as soon as user upload a image, so we have to use a tricky way instead of following MVC pattern that sending a request to the front-end controller and then communicate with backend. Instead, we try to send the ajax request from client-side JavaScript directly skipping the controllers.

In order to do this, we are actually sending a cross-domain request, which is not supported by default in the Play framework. So I have to find a way to enable to CORS filter by setting Globals configuration to the back-end server.

**Image filter and sync state at both front-end and back-end**

Borrowing an idea from facebook rainbow filter and France flag filter, We tried to implement a similar feature. In order to display it correctly, we utilized css absolute layout to take the element out of the normal flow to put the filter image on top of the original image to accomplish this feature.

In addition, we also need to sync the state of filter between front-end and back-end, so we design an extra field at User model to store the state and as soon as user choose add the filter, front-end will display the filter layer and sending the request to backend to update the field at the same time.