Social Media Website – Gatherly - Milestone 4

By Abel Weldaregay and Neha Niphadkar (fordFanatics)

TASK	**Progress**	**Challenges faced**
1. Allow users to upload files to the message area of a group	complete	The most difficult part about this requirement was differentiating between file posts and image posts. In order to differentiate between the multiple types of posts we added another parameter to our posts table that will tell us what type of post it is before we try to manipulate it. One thing we found helpful was referencing our old code from when we did profile picture uploads to help with moving the uploaded file into the correct directory.
2. If the file is an image, provide a thumbnail preview	complete	This was a very challenging requirement for us because it was hard to figure out what the file was before posting it. What we ended up using was an on change function. When the user clicks choose file and selects a file, we grab the file name and check the extension. If it is a image file then we modify the href of a hidden image tag to preview the chosen image to the user.
3. Allow direct messaging between users	complete	For this requirement, we used some of our old code in where we are handling posting group messages, but instead of group id, we passed user id. We also added a

		new direct messages table that helped keep track of who sent the direct message and to whom it was sent to.
4. Upon a user registering for your web site, utilize the Gravatar API to associate with the user on your system based on the e-mail address the user supplied.	complete	The biggest challenge in this requirement was handling the case where the user does not have a gravatar. After a couple of test cases we found out it defaults to a default gravatar if the user email does not have a gravatar, so we ended up building the url using the users email and using that as an href in the users profile picture.
5.Allow a user to override their Gravatar image by uploading a custom avatar, per Milestone 2.	complete	In order to solve this problem, we added another parameter called display picture that will tell us what type of profile picture the user has. This solved the problem because all we have to do is check the display picture parameter before we show any profile pictures.
6.Allow a user to remove their associated avatar to default to a generic avatar image within your system (for the users in the credentials file) or to default to their Gravatar image if one exists for their associated email address.	complete	As discussed in requirement 5, after adding the display picture parameter it made it easier to check what type of profile picture the user has. In order to meet the requirement, we give the user the ability to switch from the default gatherly profile picture and the default gravatar picture.
7.Integrate your site with the No CAPTCHA ReCAPTCHA API by requiring a user to fulfill the CAPTCHA requirement before proceeding with some functionality of your web site.	complete	This requirement was implemented in our login page. Before a user logs into our website, they will have to complete a CAPTCHA ReCAPTCHA in order to proceed.

8.The HTML on your website's home page, your question display page, and user profile page must be valid according to W3C's HTML validator.	complete	In order to meet this requirement, we had to go through our html code and use the correct tags for displaying the content.
9.BONUS - GitHub	complete	The biggest challenge in solving this problem was making sure that when a user logs into our website using Github credentials, if there is an email that exists under the user table that matches the Github account, then we log that user in, but if there is no such email then the Github credentials are added to the user table.
10.Allow a user to log into your web site using their GitHub credentials. Validate the credentials using the GitHub API.	complete	In our login page, when a user wants to login with their GitHub credentials, we have a button that invokes the GitHub API that will validate the user.
11. When a user is logged in using their GitHub credentials, use their GitHub avatar as the user's avatar on your web site.	complete	As discussed in requirement 5, the display picture parameter also helped with this requirement beca
12. This will need to be dynamic. If I am logged in with my GitHub credentials, post, then change my avatar on GitHub, my avatar on your web site ought to reflect this change retroactively.	complete	This requirement is met because the href url will always return the most recent image from the GitHub API.
13. Posts by a GitHub user should always display their current avatar when the web page is accessed and not the one that existed when the user posted.	complete	As discussed in requirement 12, we are using the same href on all image tag that grabs the users most recent GitHub profile picture.