Team: Ruby in the Rough

Team members:

Casey Kelly

Josh Lorenz

Marc McCarty

Jason Pelgen

Use Case UC1: Create_Post

Scope: Forum posting System
Level: User goal
Primary Actor: Student and Faculty

Stakes/Interests:
Faculty – Wants to make topics that are relevant to the course on the forum.
Student – Wants to discuss topics with others in the class on the forum.

Preconditions:
Faculty or Student is logged in.
Each class forum is separated from one another.

Success Guarantees: Post has title and content and ready to be posted on the forum.

Main Success Scenario:

1. Student or Faculty give the topic a title.
2. Student or Faculty create the body with the topic information.
3. Student or Faculty submit the topic.

Extensions:
3a. Topic is missing title, body, or both.

1. Topic does not get posted.
2. Goes back to the create topic page and system notifies the user.

Special Requirements: n/a

Tech & Data: Image compatibility, and accept special characters (e.g. other languages/symbols).

Frequency Of Occurrence: Very frequent, as many users should be participating.

Misc: User must be able to comment on posts.

Use Case UC2: Flag_Post

Scope: Class Registration System
Level: User goal
Primary Actor: Faculty

Stakes/Interests:
Faculty – wants to keep the discussion in the forum on-topic, clean, and friendly.
Student – wants to have a peaceful discussion/debate with his/her classmates.

Preconditions:
A Student has posted a comment that is off-topic and/or vulgar.
Faculty is logged in.

Success Guarantees: The post made by the Student will be placed in a quarantine.

Main Success Scenario:

1. Faculty encounters a post that is off-topic and/or vulgar.
2. Faculty indicates that the post should be removed by flagging it.
3. After flagging it, Faculty writes the reason as to why the post should be removed.
4. The post is then temporarily removed and placed into a quarantined state.

Extensions:

3a: Reason is empty

1. The post is not flagged.
2. System notifies the user to enter the reason why the post should be removed.

Special Requirements: n/a

Tech & Data: Flagged posts are to be stored in a location that a moderator is able to access.

Frequency Of Occurrence: Low frequency.

Misc: n/a

Use Case UC3: Delete_Post

Scope: Forum posting System
Level: User goal
Primary Actor: Moderator

Stakes/Interests:
Moderator – Wants to make sure that topics are kept on the subject matter, and nothing explicit is being posted.

Preconditions:
Moderator is logged in and has access to all posts.
Designated post has been flagged by a user.

Success Guarantees: Moderator deems post unfit for topic and removes from the forum.

Main Success Scenario:

1. Moderator thoroughly reviews post.
2. Moderator decides to either keep or remove the post.
3. Post gets removed by moderator.
4. Notifies all faculty within the class, with moderator's reason.

Extensions:
3a. Post is not removed by moderator.

1. Post gets un-flagged.
2. Notifies all faculty within the class that a flagged post has been un-flagged.

Special Requirements: n/a

Tech & Data: n/a

Frequency Of Occurrence: Low frequency, most users aren't purposely being malicious.

Misc: n/a

Use Case UC4: Upvote_Post

Scope: Forum posting System
Level: User goal
Primary Actor: Student and Faculty

Stakes/Interests:
Faculty – Wants to reward students that spread good information.
Student – Wants to inform other students that their content was useful.

Preconditions:
Faculty or student is logged in.
Faculty or student can see the post.
Faculty or student hasn't upvoted the post previously.

Success Guarantees: User clicks the upvote button and the post is still present.

Main Success Scenario:

1. Student or faculty navigate to a post and find it informative.
2. Student or faculty presses the upvote button.

Extensions:
2a. The post was deleted between viewing it and pressing "upvote".

1. Upvote does not occur.
2. User is notified that the post was deleted.

Special Requirements: n/a

Tech & Data: Upvote data member for post class. The posts can/will be sorted by upvotes.

Frequency Of Occurrence: Very frequent, as many users should be participating.

Misc: Highly upvoted comments or posts will be more visible on the site.

Use Case UC5: Add_Friend

Scope: Forum posting System
Level: User goal
Primary Actor: Student and Faculty

Stakes/Interests:
Student/Faculty – Wants to be able to use extra features with certain users, like posting to their status page, or searching for posts made by friends, or having friends' posts show up before others.

Preconditions:
Student/Faculty is logged in and can navigate to any user's status page.
[The other] Student/Faculty allows adding friends.

Success Guarantees: Student/Faculty requests friendship from another and they accept.

Main Success Scenario:

1. Student/Faculty navigates to status page of other Student/Faculty.
2. Student/Faculty presses "Add Friend" button.
3. [The other] Student/Faculty receives a friend notification and accepts.
4. Student/Faculty receives a friendship confirmation notification.

Extensions:
3a. [The other] Student/Faculty does not accept the invitation.

1. The users do not become friends.
2. The original sender's request is retracted.
3. The original sender can no longer send requests, but the decliner can send one later, if they wish.

Special Requirements: n/a

Tech & Data: Friend status is by default disabled, friend status can only be enabled when both users agree (involves a relationship object that's created when adding/blocking another user), extra mutual features become available upon becoming friends.

Frequency Of Occurrence: Medium frequency, since most users won't become friends with most other users, but rather a moderate amount of colleagues that wish to use some extra features with each other.

Misc: n/a

Use Case UC6: Send_Private_Messages

Scope: Forum posting System
Level: User goal
Primary Actor: Student, Faculty and Moderator

Stakes/Interests:
Student/Faculty/Moderator – Having the ability to discuss private issues and messages in a secure environment not visible to everyone.

Preconditions:
Student/Faculty/Moderator is logged in and can navigate to any user's status page.

Success Guarantees: Student/Faculty/Moderator sends a private message and the recipient Student/Faculty/Moderator receives it, can view it and respond if desired.

Main Success Scenario:

1. Student/Faculty/Moderator navigates to status page of other Student/Faculty/Moderator.
2. Student/Faculty/Moderator presses the "Send a Private Message" button.
3. Student/Faculty/Moderator enters a title, writes the message in the body and clicks send.
4. After hitting send, a message pops up notifying the user that the message was sent successfully.
5. The Student/Faculty/Moderator is redirected back into the status page of the recipient Student/Faculty/Moderator.
6. The recipient Student/Faculty/Moderator, when logged in, sees a notification in their inbox denoting they have an unread private message.
7. The recipient Student/Faculty/Moderator can click on the notification, read the message and respond.

Extensions:

- 3a. The message does not send
  1. Check and make sure the private message has a title
  2. Check and make sure the message has a body
  3. Check and make sure the forum is not down.
- 6a. Recipient does not click or view the message
  1. The message will remain in inbox
  2. It will be marked as unread
  3. It will delete only when the user chooses to delete it

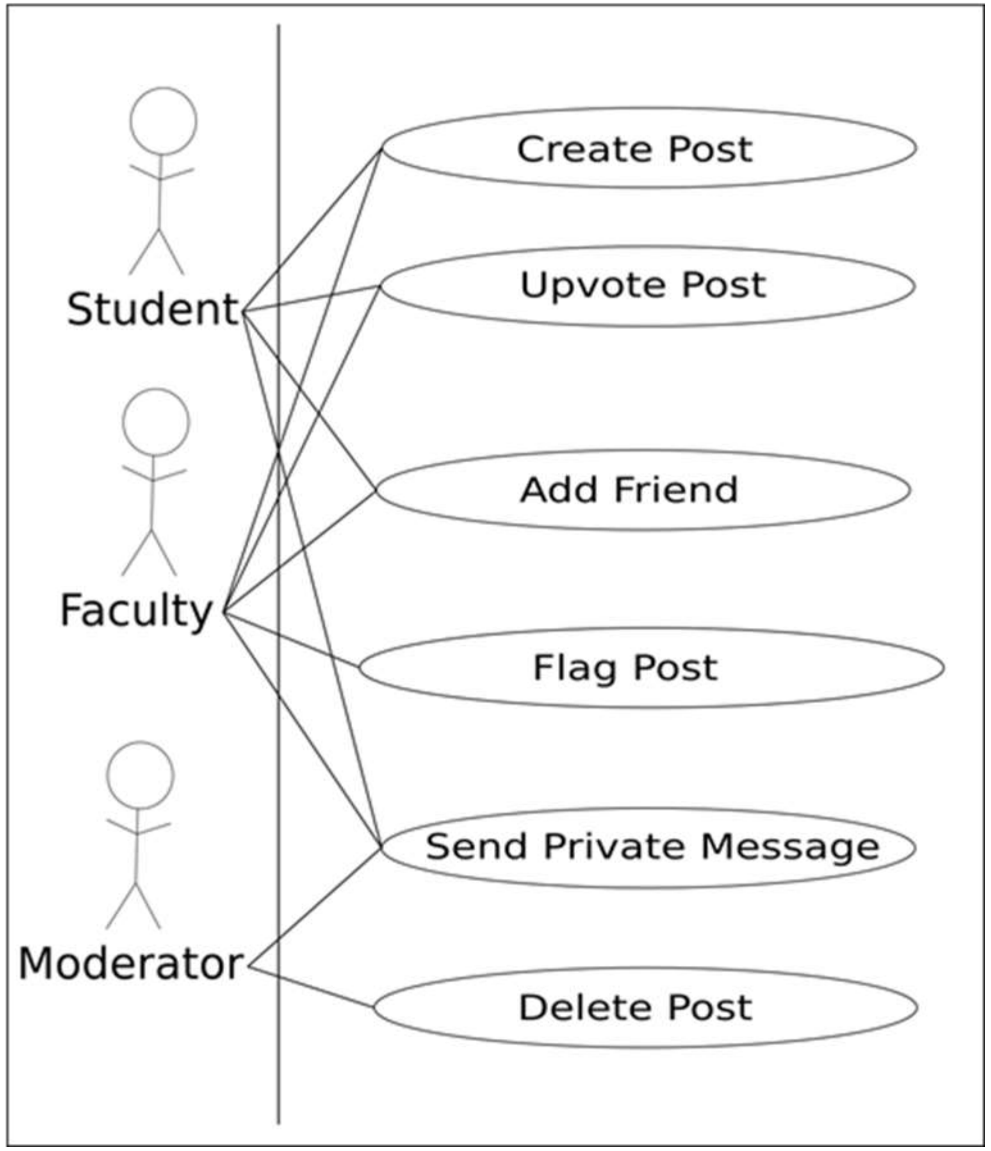
Special Requirements: n/a

Tech & Data: Private Messages, by default, are enabled for everyone with no ability to disable them. This will create a private thread for individuals that only the specified users can see.

Frequency Of Occurrence: Medium frequency. Not all users will access or choose to use private messages, however they are likely to be used very commonly among colleagues, advisors, moderators and close friends.

Misc: User must be able to view and respond in private messages.

Use Case Diagram


Student
Faculty
Moderator
Create Post
Upvote Post
Add Friend
Flag Post
Send Private Message
Delete Post