Requisites

###### 

###### TODO

* Select 3 of the project description items
* Convert project description to good specification using precise definitions of the requirements
* Sort each into one of the functional or non-functional categories
* Make any relevant UML diagrams

#### Functional Requirements

#### Use Cases

1. Create Account
2. Adding Friends
3. Post Content
4. Create Class Schedule
5. Upload Resume
6. Create Polls

##### Notes

* Convert project description to good specification using precise definitions of the requirements. Currently it's in too much of a narrative form
* Indicate the functional and nonfunctional requirements
  + Functional ex: username must be a mun email
  + Non-functional requirement ex: A time restriction like “email for signup must be sent within 5 minutes”
  + Non-functional ex: Consider scalability,
  + Functional Requirements:
    - business rules
    - transaction corrections, adjustments
    - …
  + Functional notes:
    - verbs
    - mandatory
    - captured in use case
    - product feature
    - easy to capture
  + Non-functional:
    - performance
    - scalability
    - capacity
    - availability
    - reliability
  + Non-functional notes:
    - attributes
    - not mandatory
    - captured in quality attribute scenario
    - product properties
    - difficult to capture
* Bunch of testing methods listed here

# Section X - Use Cases Requirements

## X.1 Defintions

1. *<User>* a person that has a verified MUNbook account and is logged into their account.
2. *<Guest>* a person that does not have a verified MUNbook account or a person who has a MUNbook account but is not currently logged into their account.
3. *<Friend>*
4. *<Notification>* A short message notifying a user of an event that appears on their timeline. Notification events include;
   1. sending or receiving a friend request
   2. accepting or ignoring a friend request
   3. another user accepting or ignoring your friend request
   4. A friend posts new content on their timeline

## X.2 Use Cases

### UC1 - Create an Account

Memorial university students who have not yet made an account can do so.

Functional Requirements

* Username must be a valid mun email, TODO email requirements here
* Password must be alphanumeric and contain at minimum 1 capital letter and 1 number
* TODO supported picture formats, dimensions, file size

**Preconditions**

1. User’s email must not be connected to an existing account.

**Main Flow**

The System provides form fields for email and password as well as the option to upload a profile picture from the User’s computer. When the User submits the form the System sends an email to the address provided by the User. The User reads this email and follows an included link back to a confirmation page on the Website. The User confirms their email by clicking the confirmation button on the page.

**Sub Flows**

None.

**Alternative Flows**

[E1] If the information entered into the forms does not meet the requirements [TODO reference to profile requirements] the submit button will return to the signup page and display the relevant error.

[E2] If the User attempts to upload a profile picture that fails to meet the profile picture requirements [TODO profile pic requirements reference] then the picture will not be uploaded and the User will be shown the relevant error.

### UC2 - Add Friends

Functional Requirements:

* Registered User initially have an empty friends list

**Preconditions**

1. User must be registered [UC1].
2. The User’s friend list begins empty the first time this Use Case is run

**Main Flow**

User navigates to another User’s page using either [TODO person search use case] or [TODO suggested friend list use case]. The User sends a friend request to the target User using the “Send Friend Request” button [S#]. The receiving User has the request appear in their News Feed and may accept or ignore [E#] the request. If accepted both Users are registered as Friends and the request is no longer shown.

**Sub Flow**

[S#] After being used the “Send Request” button is disabled.

**Alternative Flow**

[E#] If the Friend Request is ignored then it ceases to appear in the recipient’s New Feed.

### UC3 - Suggested Friends List

**Functional Requirements**

* List is sorted based off priority. Priority is a weighted score based on number of relations that the listed User has in common with the list’s owner.

**Preconditions**

1. The User is Registered.

**Main Flow**

The User navigates to their Friend List page. The System displays a list of 5 Users that are related to the User [E#] [TODO reference to friend suggestion criteria]. The User may navigate to the Profile page of any of the Users shown in the list.

**Sub Flow**

None.

**Alternative Flow**

[E#] If the User has no Friends, Courses, S or Programs [TODO other?] then the list is a random selection of Users.

**#) Search**

**Preconditions**

1. User is registered.

**Main Flow**

User enters a string into the search bar. The System returns a list of Users whose name matches the string. The User can use the returned list to navigate to the Profile Pages of any of the Users shown.

**Sub Flow**

None.

**Alternative Flows**

[E#] If the string does not match any Users then an error message is returned instead of a list.

### UC4 - User’s Timeline

**Preconditions**

1. *<User>* has their own homepage open and their own timeline is visible.

**Main Flow**

The content within the timeline is sorted chronologically with the most recent at the top. If the timeline has no posts then it will be empty. Content consists of posts and *<Notifications>*. Posts contain images, messages, and video. *<Notifications>* are a short messages notifying a user of an event that appear on their timeline. Events include . Timeline options include Visibility, Posting Rights, and New Post. Options available on an individual post include Delete, Comment, and Visibility. When the *<User>* clicks the timeline option, ‘New Post’, the new post window opens up. It has an area for typing text and the following options available, Upload Video/Image, Publish, and Cancel. When the *<User>* types text, text appears. When the *<User>* Clicks Upload Video/Image a filepicker is shown. When the *<User>* selects a file from the filepicker and hits OK the file picker disappears and the selected media appears on the new post window. When the *<User>* hits publish the new post window disappears and the new post appears at the top of the user’s timeline.

**Sub Flow**

x[S1] *<User>* can respond to notifications by accepting or ignoring friend requests. Accepting or ignoring a friend request is a notification event and it will appear on the User’s timeline. (move to main flow or seperate use case)x

x[S2] *<User>* can reply to messages from other users on their timeline. (move to main flow or seperate use case)x not sure if this is even a requirement.

[S3] When creating a new post the *<User>* hits the cancel button. New Post window disappears and *<User>* is returned to their timeline.

**Alternative Flows**

[E1] <*User>* has attempted to upload an unsupported filetype or has exceeded max size limits. The system notifies the user that the file type is unsupported or that they have exceeded the max file size limit depending on the violation and they will then be given another to upload the file or to cancel the operation.

[E2] <*User>* has exceeded the max character limit for text in a post. User is unable to type anymore text and is alerted.

### UC5 - Timeline Visibility

**Preconditions**

1. There are 3 *<Users>* referred to as User A, User B, and User C.
2. User A is friends with User B and User C

**Main Flow**

The Timeline option Visibility is visible on the timeline menu bar. When a The available options are Only User, Everyone, Only Friends, List of Friends.

**Sub Flow**

None.

**Alternative Flows**

None.

### UC6 - Timeline Posting Rights

**Preconditions**

User has an account and is logged in.

**Main Flow**

**Sub Flow**

**Alternative Flows**

7.)

* Users can post content on other user’s posts depending on the visibility set by the original poster.
* A reply can be made under a comment, with the appropriate visual indentations.
* Comments can be edited and the edit history will always be available for view.

Functional Requirements:

* Only posts visible to a user can be commented on.
* A comments can be characters and images.
* An edit button would be available to make changes to an existing comment.
* A delete button to delete a comment will be available to delete a comment.

Non-Functional Requirements:

* A comment can’t exceed 1000 characters or an image greater than 5MB(approximations)
* A comment made is only visible to the same visibility of the original poster.

**Use Case Name**: Posting comments

**Actors**: Users

**Precondition**: User must have a valid account and a friend’s status to comment on.

**Main Flow**: A user can see a post from a friend or another user due to the visibility setting. User A posts a comment on User B’s status. User A’s comment can only be edited by them but the status can be deleted by User B due to hierarchy of the posting.

**Alternative Flow**: The user changes the privacy of the status while the comment is being produced and then the comment won’t be able to be posted if the privacy setting is changed before the comment is posted.

8.)

A public or a private study group can be created.

* Group Creator becomes the admin.
* The admin can invite other members to join the group and can set different levels of accessibilities for each invited member.
* A public group will allow any member to join instantaneously.
* A private group can only add members by the admin inviting them, accepting them invitation to join the group or by other members inviting them to join, if they are allowed to do so.
* Within the group content can be shared and posted by all the members.

Functional Requirements:

* A group option to create a group
* A private or public group option
* Members must be invited into the group
* Members can be allowed to invite other people into the group or they may not
* An accept to members requested to join group
* Invited members can be approved by the admin or members allowed to invite.
* The group name can be set
* Contents like pictures, links, and documentations can be uploaded to the group.

Non-Functional Requirements:

* A group must be instantaneously created according to the privacy settings chosen by the creator.
* No limits to how many members can be in a group
* Picture formats, links and documentation must all be to the size and file type that has been set throughout the application.

**Use Case:** Create Study Group

**Actors:** Users

**Preconditions:** The user must be a valid member and have friends in the network.

**Main Flow:**

* Group Creator becomes the admin.
* The admin can invite other members to join the group and can set different levels of accessibilities for each invited member.
* A public group will allow any member to join instantaneously.
* A private group can only add members by the admin inviting them, accepting them invitation to join the group or by other members inviting them to join, if they are allowed to do so.
* Within the group content can be shared and posted by all the members.

**Alternative Flow:**

**-**

**-**

**-**

9.

Lost and Found functionality in the application.

* A lost item can be reported by a user by posting a picture of it with a short description and a location tag in order to show the location of the item in the lost and found section.
* A user can contact a poster regarding a lost item by their email, if they are friends the user has access to the poster’s phone number.

Functional Requirements:

* Must be linked to Google maps in order to tag a location

Non Functional Requirements:

* Google maps api
* Email contact

**Use Case:** Create Lost and Found Item

**Actors:** Users

**Preconditions:** User must have found something that is lost and can be posted to help find its owner.

**Main Flow:**

* A lost item can be reported by a user by posting a picture of it with a short description and a location tag in order to show the location of the item in the lost and found section.
* A user can contact a poster regarding a lost item by their email, if they are friends the user has access to the poster’s phone number.

**Alternative Flow:** The lost item is claimed by someone who isn’t the actual owner or the lost item is taken and kept by the person who found it and isn’t posted to be found.

**13) Mystery Feature**

**Preconditions:**

**Main Flow:**

**Sub Flow:**

**Alternative Flow:**