COS301: MINI PROJECT PHASE 1
GROUP 5 A

Khathutshelo Matidza 11072157
Renaldo van Dyk 12204359
Andreas du Preez 12207871
Sean Hill 12221458
Kgomotso Sito 12243273
Hlavutelo Maluleke 12318109
Siboniso Masilela 10416260
Semaka Malapane 13081129

GitHub link

Contents

1 User Management **5**
- 1.1 Use case prioritization . . . 5
  - 1.1.1 Critical . . . 5
  - 1.1.2 Important . . . 5
  - 1.1.3 Nice-To-Have . . . 5
- 1.2 Use case/Services contracts . . . 5
  - 1.2.1 Pre-Conditions . . . 5
  - 1.2.2 Post-Conditions . . . 5
- 1.3 Required functionality . . . 6
- 1.4 Process specifications . . . 7
  - 1.4.1 User Management . . . 7

2 Admin Management **7**
- 2.1 Use case prioritization . . . 7
  - 2.1.1 Critical . . . 7
  - 2.1.2 Important . . . 8
  - 2.1.3 Nice-To-Have . . . 8
  - 2.1.4 Pre-Conditions . . . 8
  - 2.1.5 Post-Conditions . . . 8
- 2.2 Required functionality . . . 9
- 2.3 Process specifications . . . 10
  - 2.3.1 User Management . . . 10

3 Threads **10**
- 3.1 Use case prioritization . . . 10
  - 3.1.1 Critical . . . 10
  - 3.1.2 Important . . . 11
  - 3.1.3 Nice-To-Have . . . 11
- 3.2 Use case/Services contracts . . . 11
  - 3.2.1 Pre-Conditions . . . 11
  - 3.2.2 Post-Conditions . . . 11
- 3.3 Required functionality . . . 12
- 3.4 Process specifications . . . 13
  - 3.4.1 Create Threads . . . 13
  - 3.4.2 Delete Threads . . . 14
  - 3.4.3 Read Threads . . . 15

3.4.4 Read Tracking 16

4 User Ranking **16**
4.1 Use case prioritization 16
4.1.1 Critical 16
4.2 Important 16
4.3 Use case/Services contracts 16
4.3.1 Pre-Conditions 16
4.3.2 Post-Conditions 17
4.4 Required functionality 17
4.5 Process specifications 18
4.5.1 Rating Post 18
4.5.2 Mark Allocation 19
4.5.3 Manage Ranking 20

5 Social Tagging **20**
5.1 Use case prioritization 20
5.1.1 Critical 20
5.1.2 Important 20
5.1.3 Nice-To-Have 21
5.2 Use case/Services contracts 21
5.2.1 Pre-Conditions 21
5.2.2 Post-Conditions 21
5.3 Request and Results Data Structures 22
5.4 Process specifications 23
5.4.1 Social Tagging 23

6 Reporting **23**
6.1 Use case prioritization 23
6.1.1 Critical 23
6.1.2 Important 24
6.1.3 Nice-To-Have 24
6.2 Use case/Services contracts 24
6.2.1 Pre-Conditions 24
6.2.2 Post-Conditions 24
6.3 Request and Results Data Structures 25
6.4 Process specifications 26

7 Domain Model **27**

8 High Level Use Case Diagram **28**

1 User Management

1.1 Use case prioritization

1.1.1 Critical

- 1.1 updateAccountInformation
- 1.2 viewProfile
- 1.3 logIn
- 1.4 logOut

1.1.2 Important

- 1.5 directMessage

1.1.3 Nice-To-Have

1.2 Use case/Services contracts

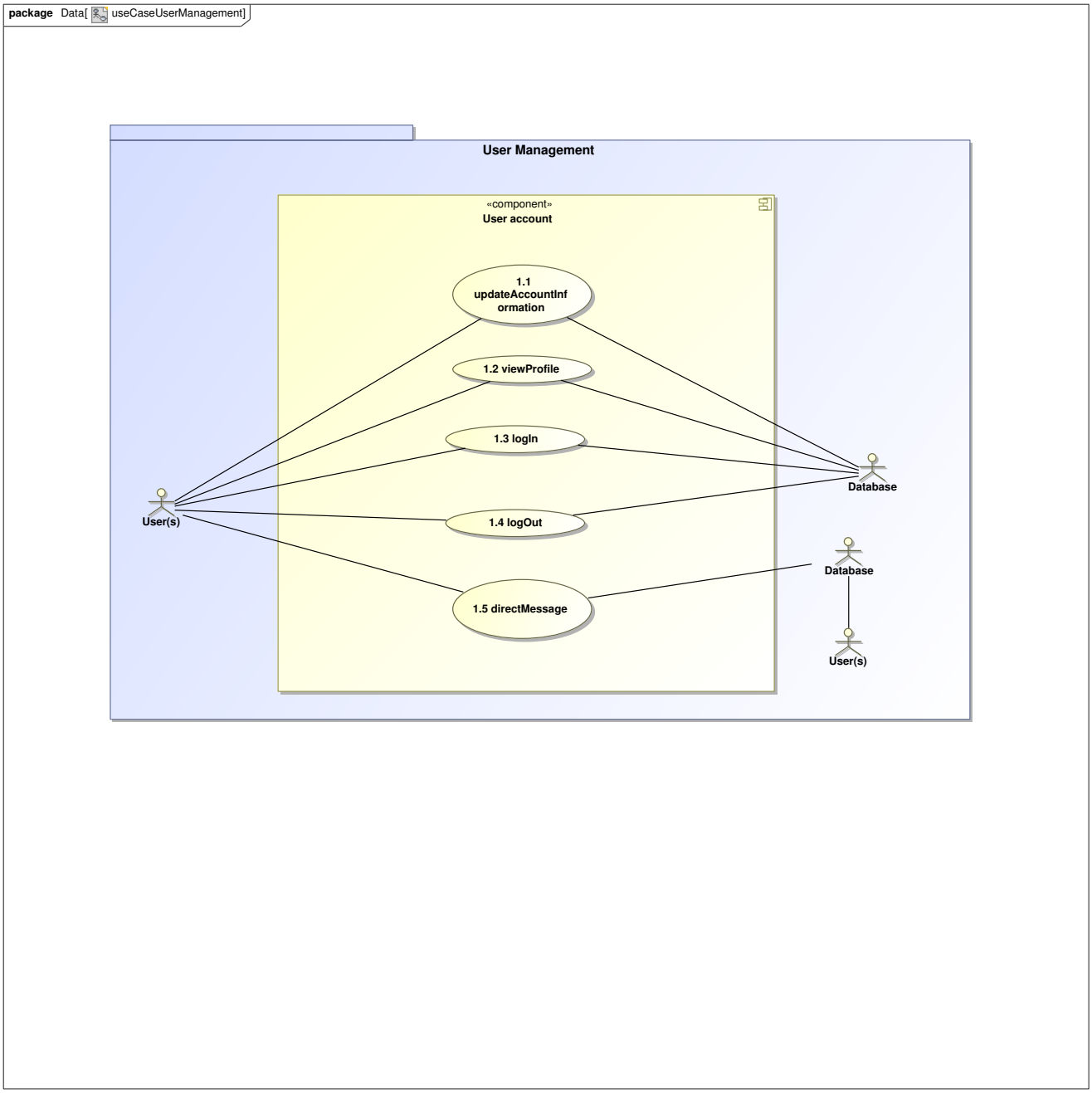
1.2.1 Pre-Conditions

- 1.1 New information must be valid.
- 1.4 User must be logged in.
- 1.5a User must have a message.
- 1.5b User must have (a) valid recipient(s).

1.2.2 Post-Conditions

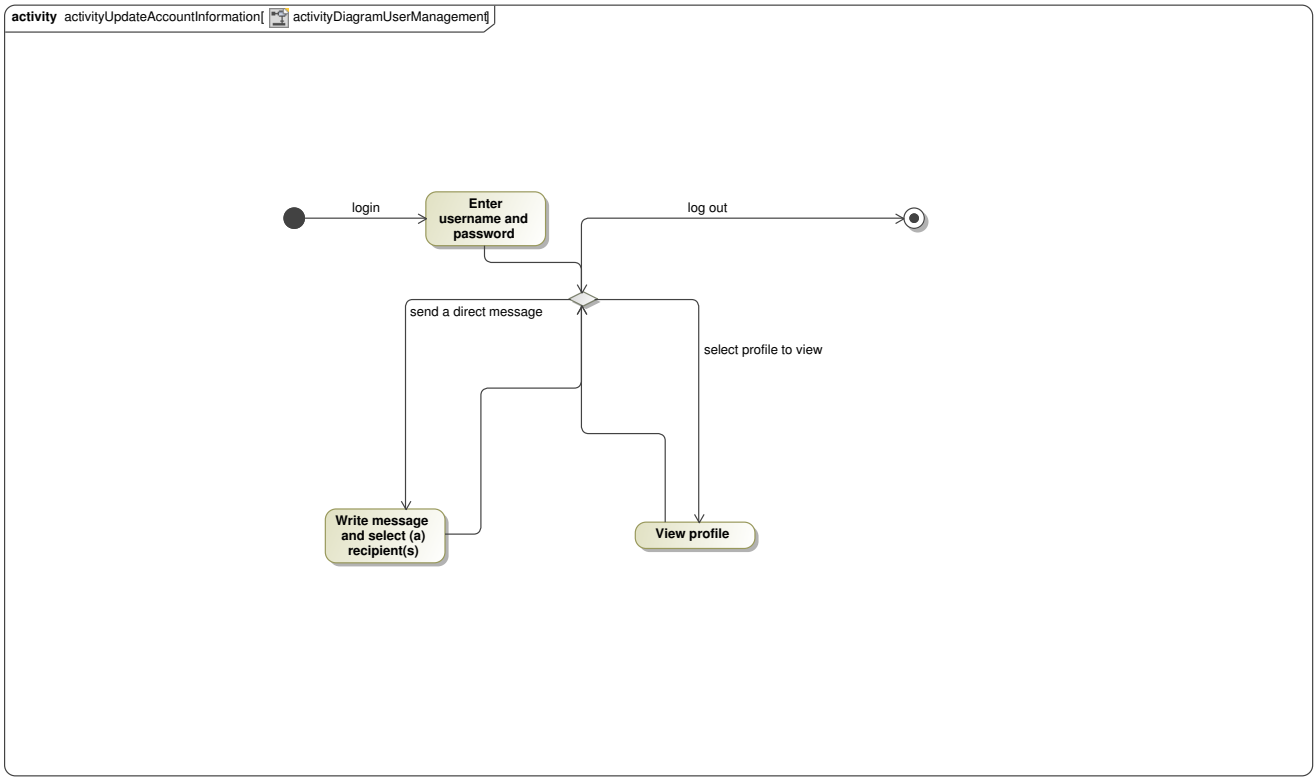
- 1.1 Information is updated on database.
- 1.4 User is logged out.
- 1.5 (The) recipient(s) recieve message.

1.3 Required functionality



1.4 Process specifications

1.4.1 User Management



2 Admin Management

2.1 Use case prioritization

2.1.1 Critical

- 2.1 createAccount
- 2.2 deleteAccount

2.1.2 Important

2.1.3 Nice-To-Have

- 2.3 customizeInterface
- 2.2 summarizeThreads

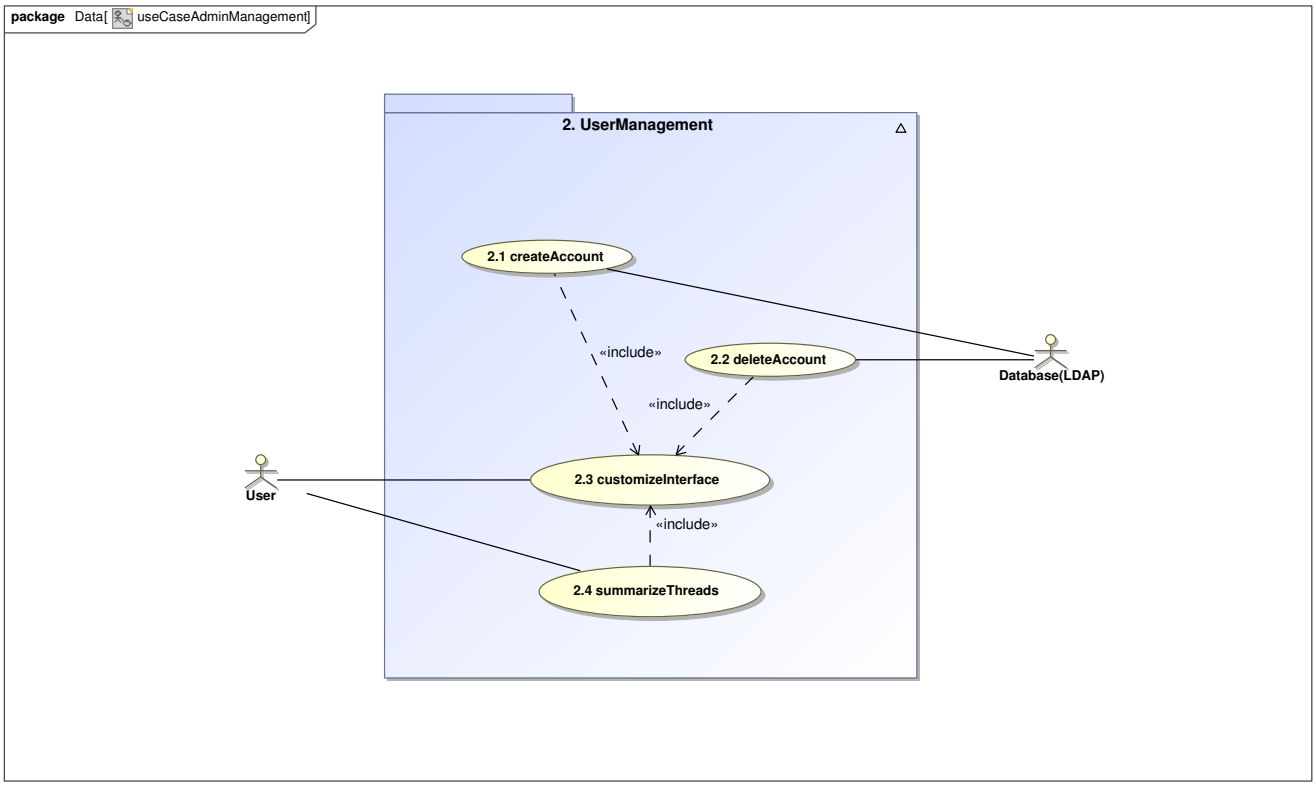
2.1.4 Pre-Conditions

- 2.1 User must be a registered with the university
- 2.2 User must no longer be a student at the university
- 2.3 User must be of a specific status level(authority)
- 2.4 User must be of a specific status level(authority)

2.1.5 Post-Conditions

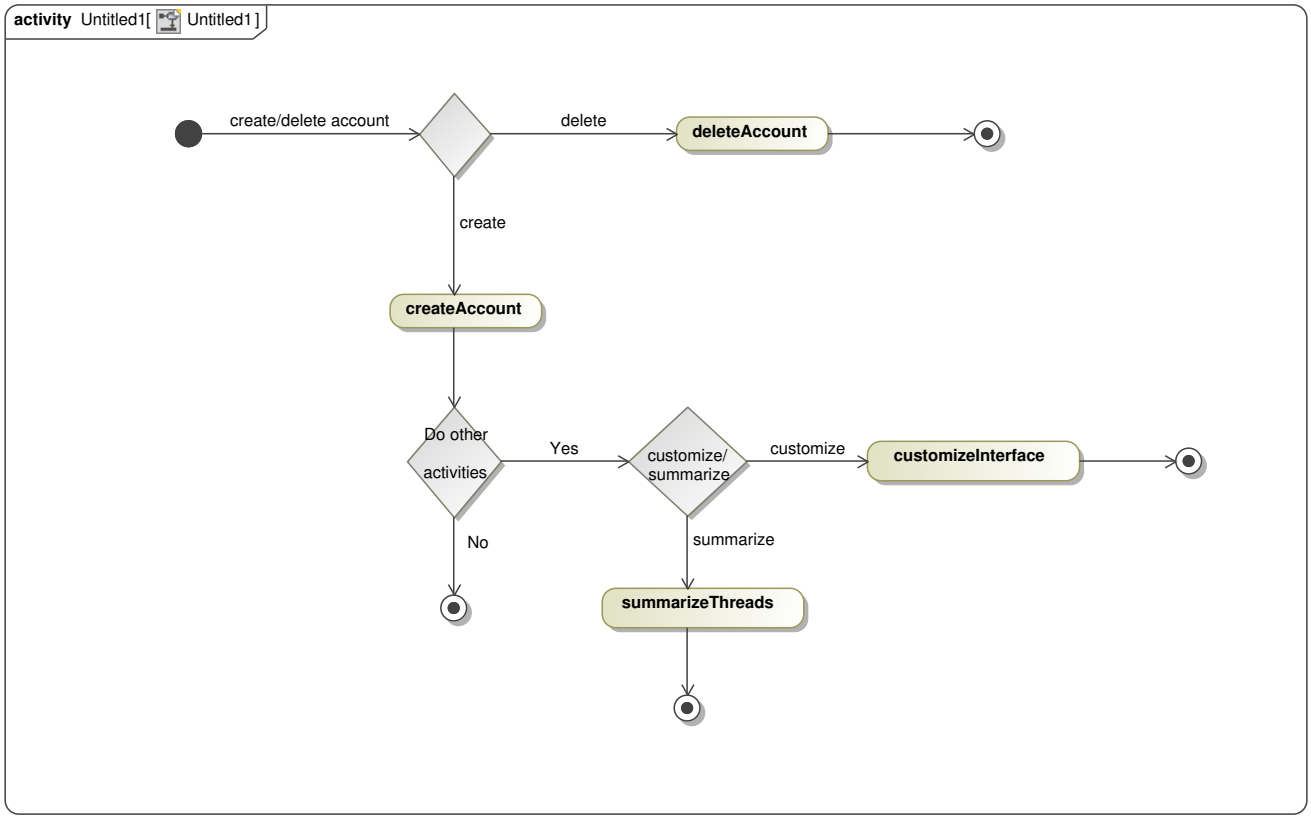
- 2.1 Account must exist
- 2.2 User account should be deleted along with all activities associated with it
- 2.3 Posts should be moved around and changes be visible
- 2.4 Summaries of threads should be created

2.2 Required functionality



2.3 Process specifications

2.3.1 User Management



3 Threads

3.1 Use case prioritization

3.1.1 Critical

- 3.1 createRootThread
- 3.1.1 createSubThread
- 3.2 deleteThreads

- 3.2.1 archiveThreads
- 3.3 updateThreads
- 3.4 readThreads

3.1.2 Important

- 3.5 Read Tracking

3.1.3 Nice-To-Have

3.2 Use case/Services contracts

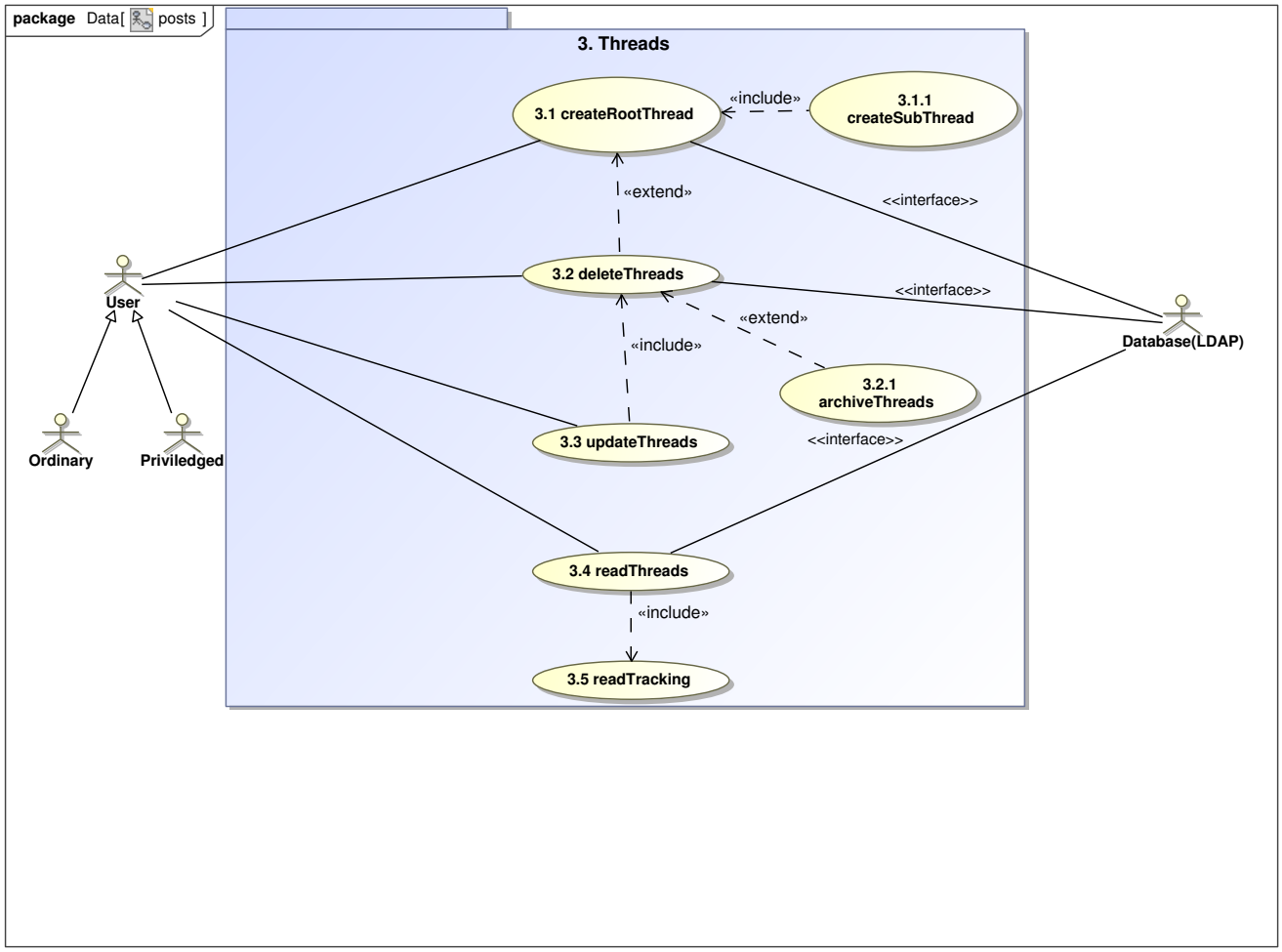
3.2.1 Pre-Conditions

- 3.1 a) User must be logged into the system to create root threads
- 3.2 a) Thread must be read from database before it can be deleted
- 3.2.1 a) Thread must be deleted before it can be archived
- 3.3 a) User must be logged into the system to update the database
- b) Before the database can be updated values must be changed
- 3.4 a) User must be logged into the system to read values from database
- 3.5 a) User must be logged into the system to see unread mesassages in bold

3.2.2 Post-Conditions

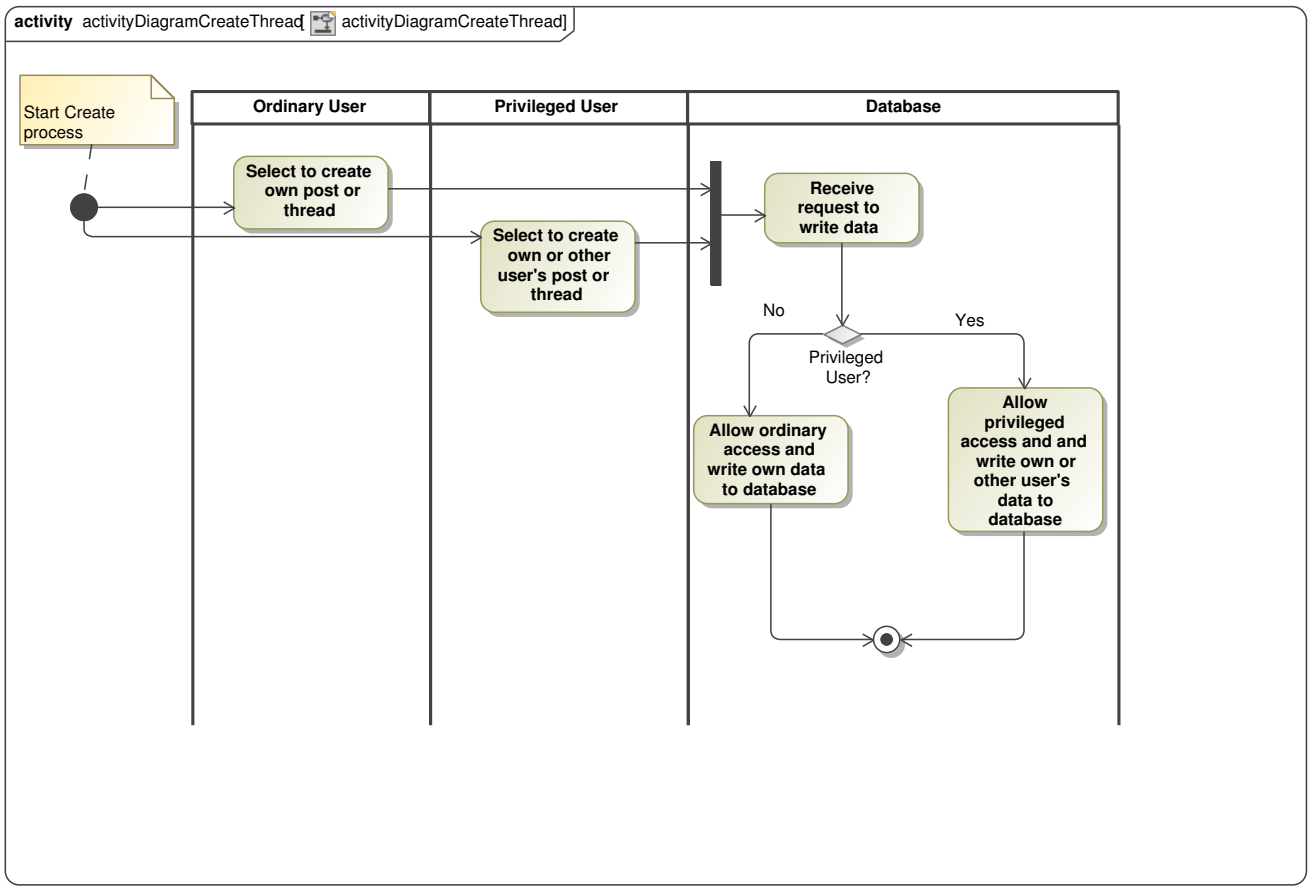
- 3.1 a) All threads will be visible to users who are logged into the system
- b) Threads will exist in database
- 3.2 a) Threads will be acrhived.
- b) Threads will not be visible to users
- 3.3 a) Updated threads will be visible
- 3.5 a) Thread is marked as unread

3.3 Required functionality

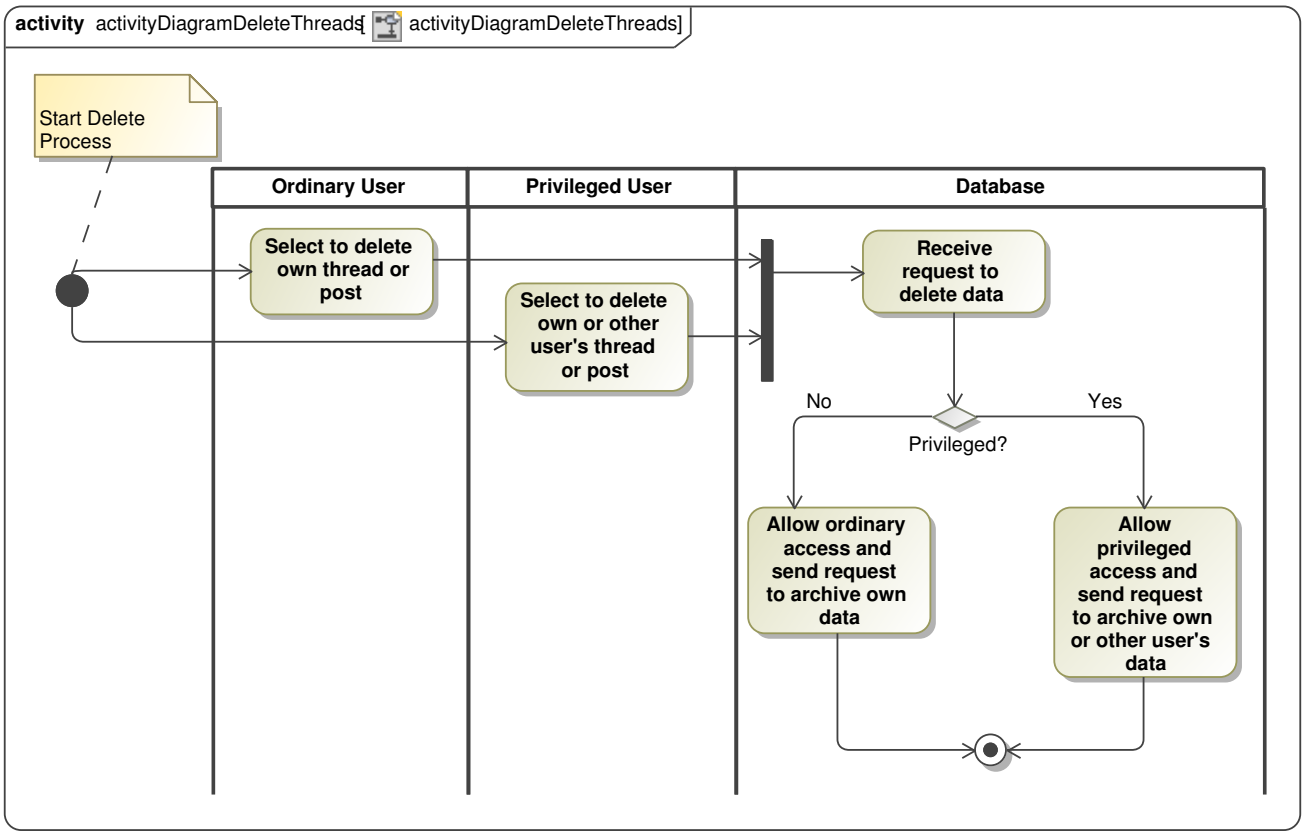


3.4 Process specifications

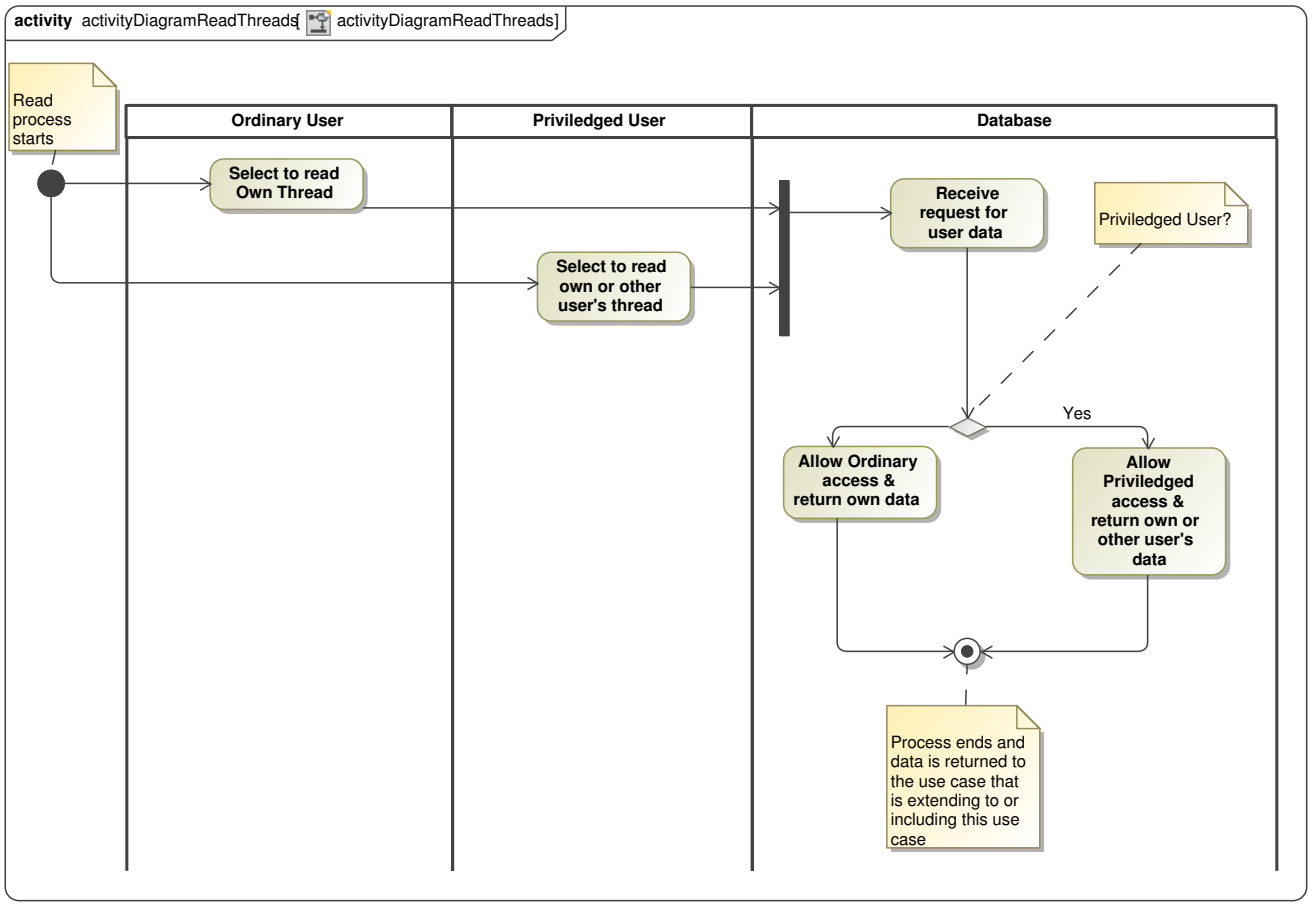
3.4.1 Create Threads



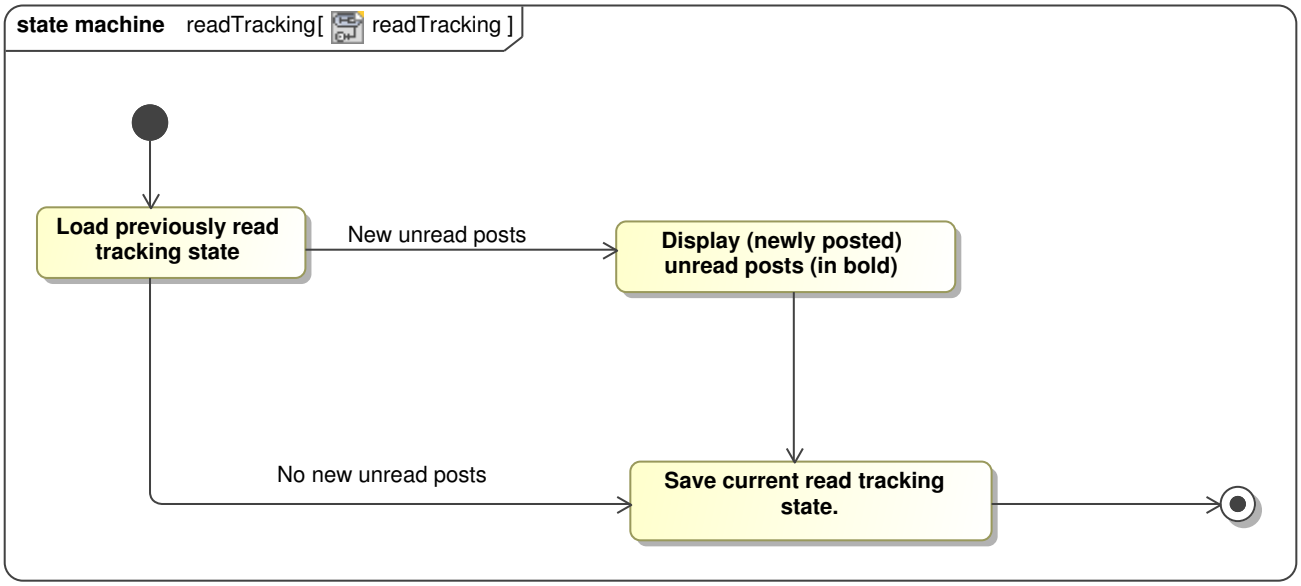
3.4.2 Delete Threads



3.4.3 Read Threads



3.4.4 Read Tracking



4 User Ranking

4.1 Use case prioritization

4.1.1 Critical

- 4.1 ratePost
- 4.3 manageUserRanking

4.2 Important

- 4.2 markAllocation

4.3 Use case/Services contracts

4.3.1 Pre-Conditions

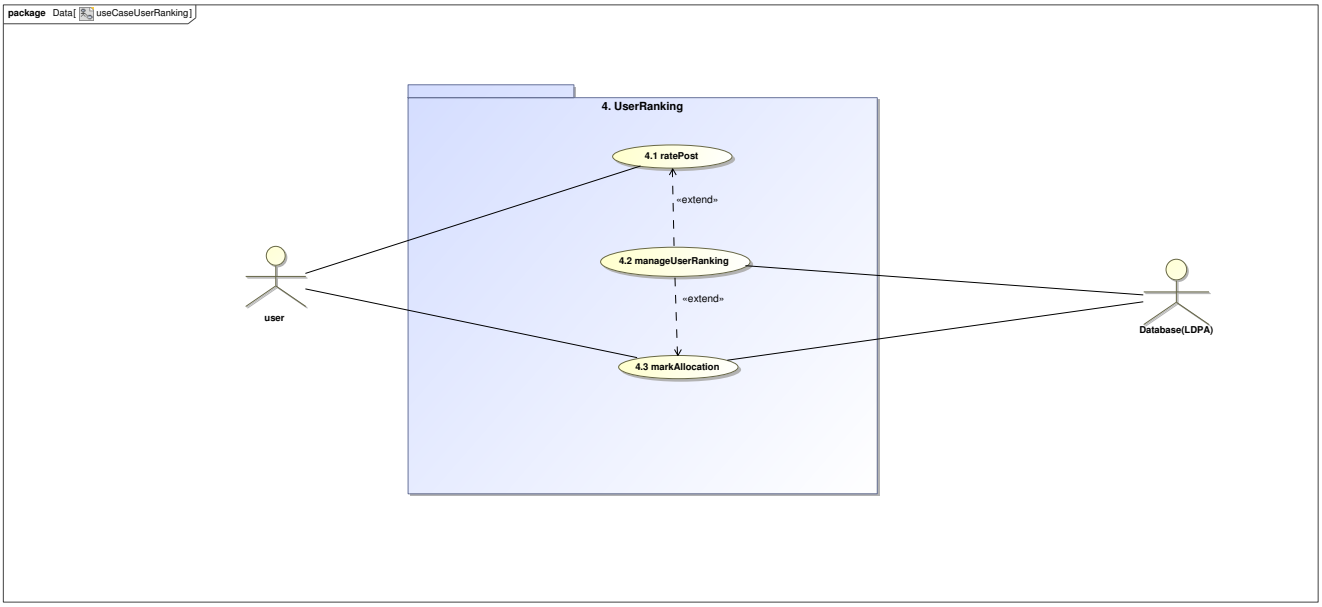
- 4.1 Must be logged in

- 4.2 Must be logged in and also a lecturer or TA
- 4.3 Must be logged in and also a lecturer or TA

4.3.2 Post-Conditions

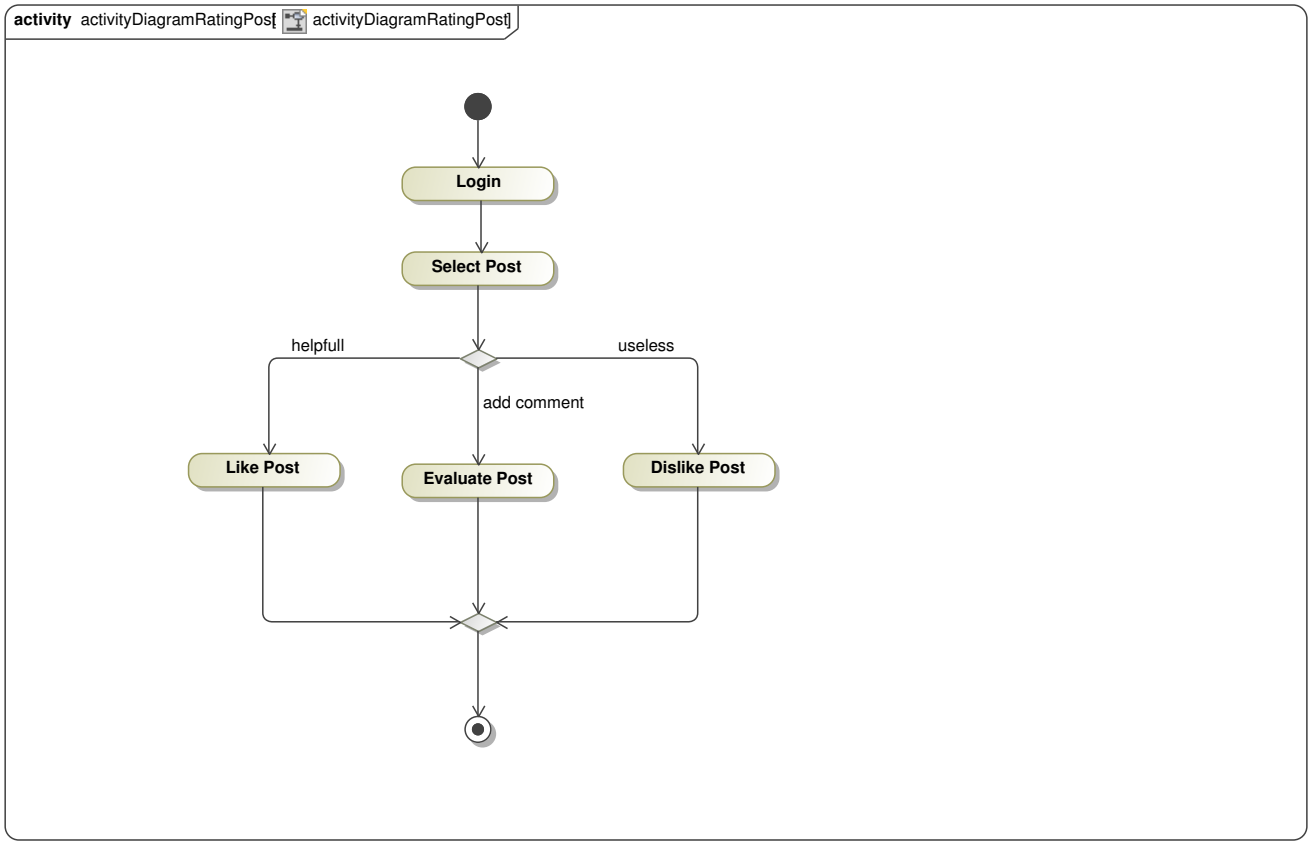
- 4.1 Post is rated
- 4.2 Student marks are allocated and status level changes accordingly
- 4.3 Students are ranked accordingly

4.4 Required functionality

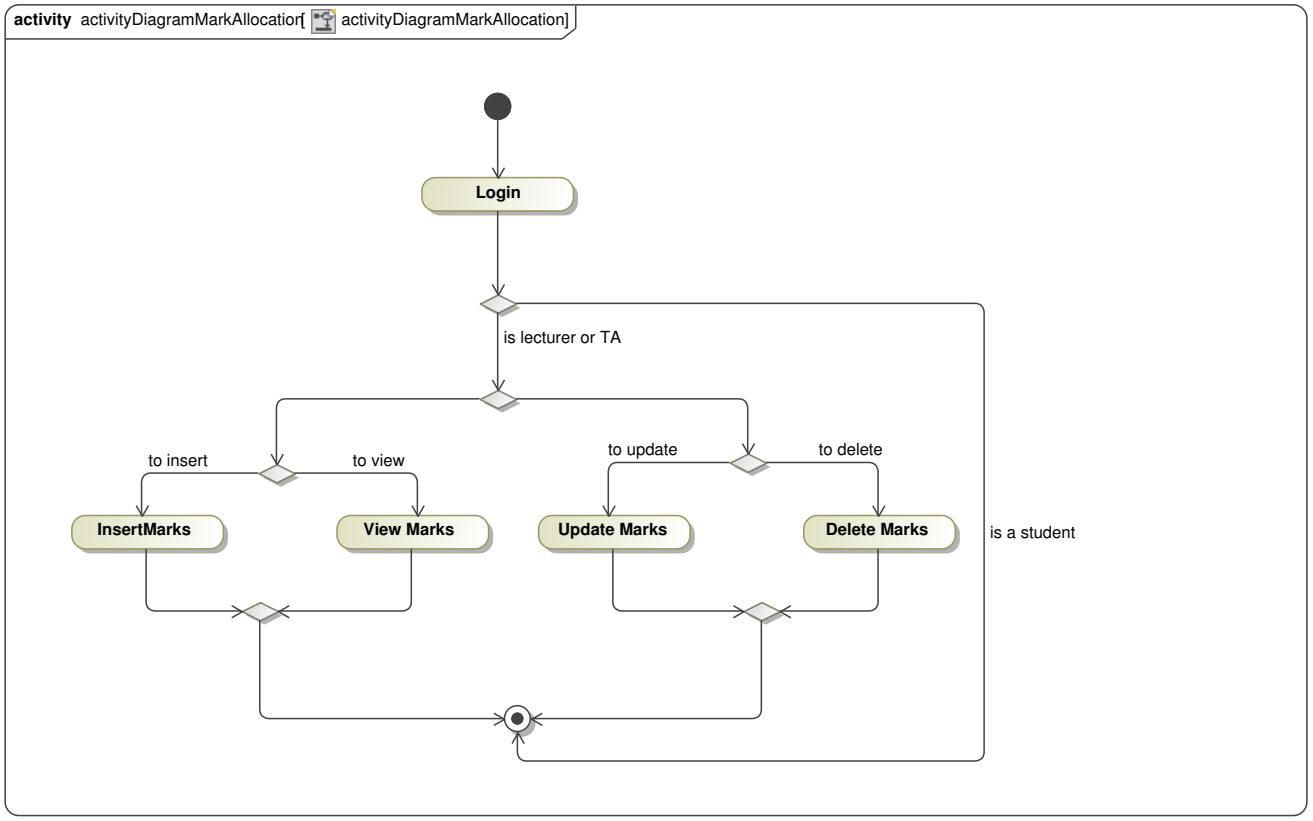


4.5 Process specifications

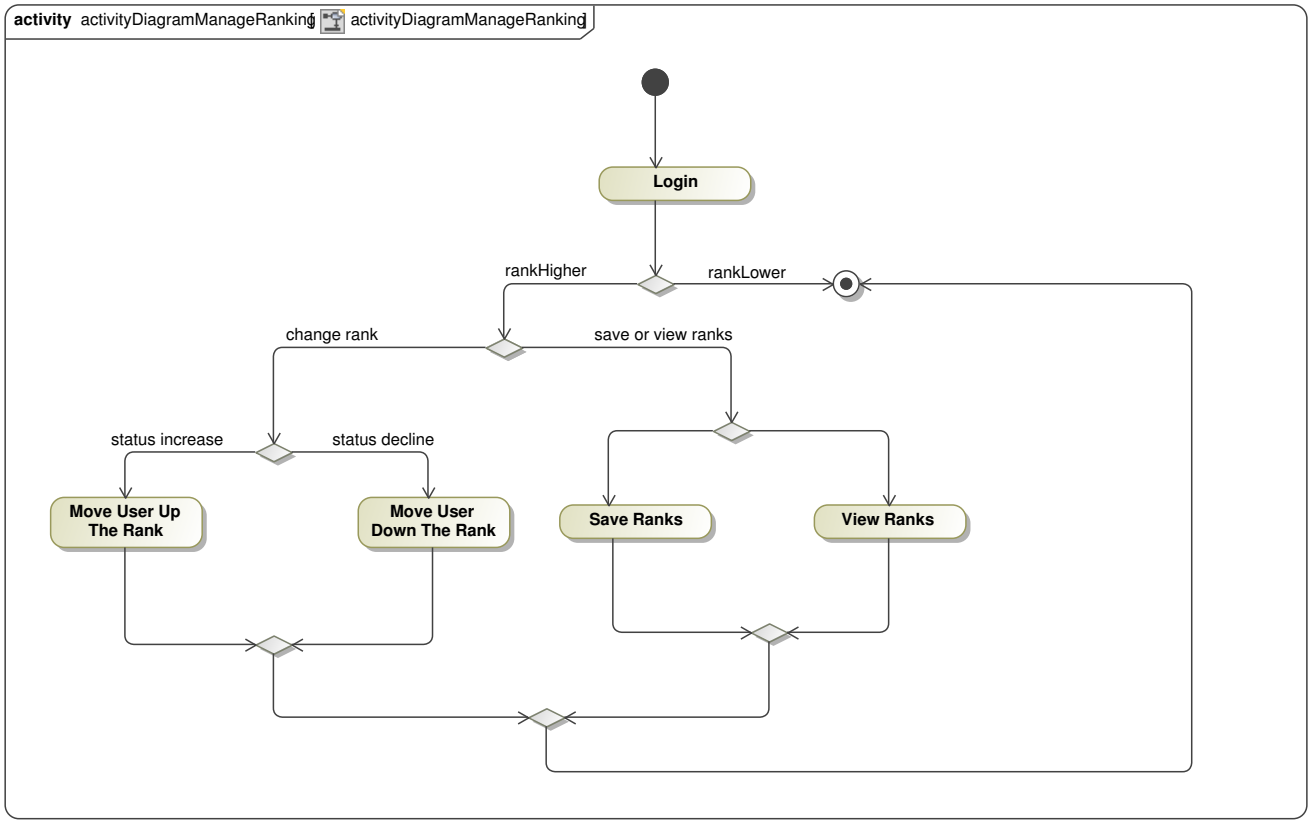
4.5.1 Rating Post



4.5.2 Mark Allocation



4.5.3 Manage Ranking



5 Social Tagging

5.1 Use case prioritization

5.1.1 Critical

- 5.1 createATtag
- 5.3 deleteTag

5.1.2 Important

- 5.2 searchTag

- 5.4 updateTag
- 5.5 viewTag

5.1.3 Nice-To-Have

- notify

5.2 Use case/Services contracts

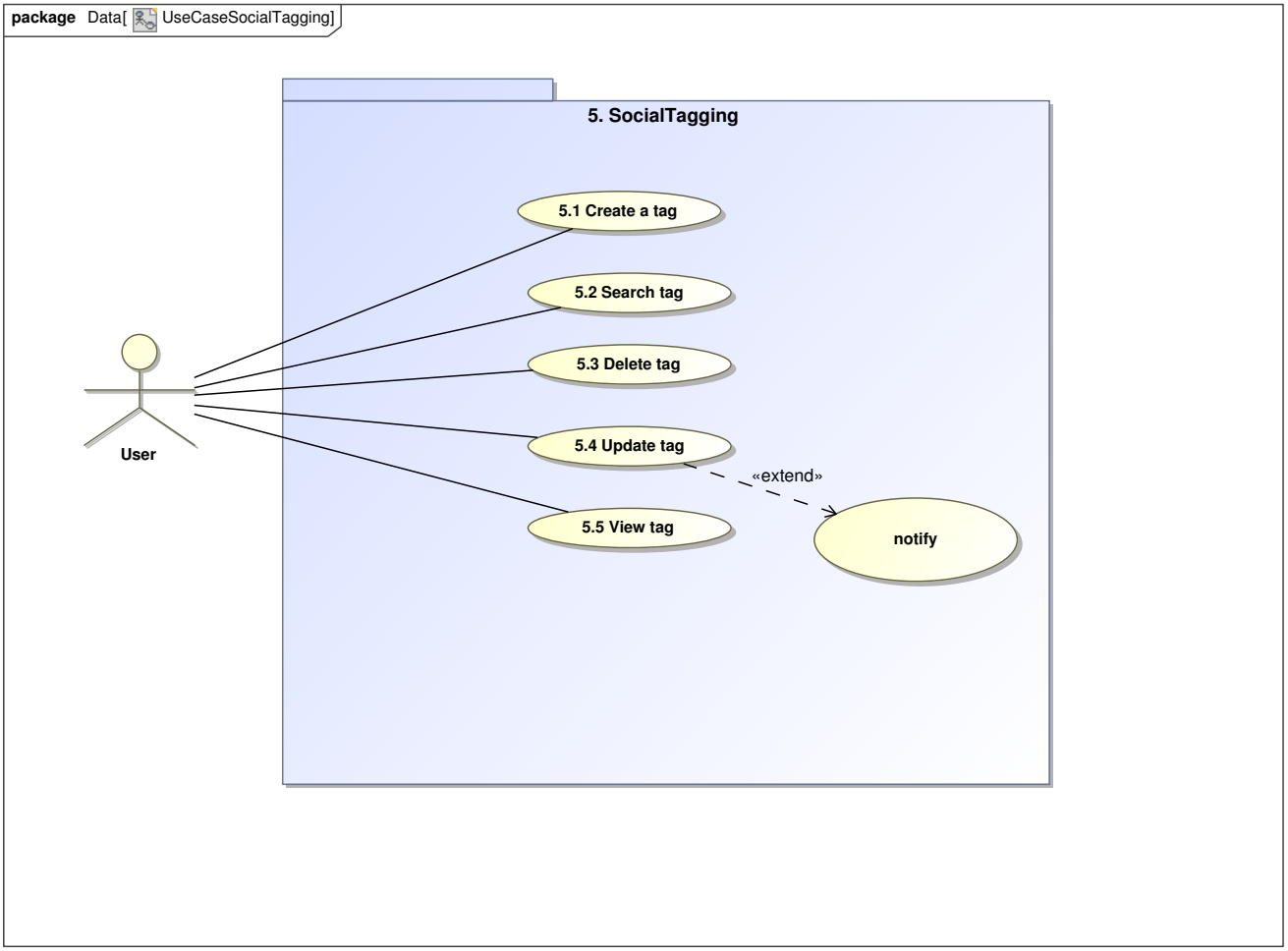
5.2.1 Pre-Conditions

- 5.1 User needs to be registered for the buzz space to be able to create tags

5.2.2 Post-Conditions

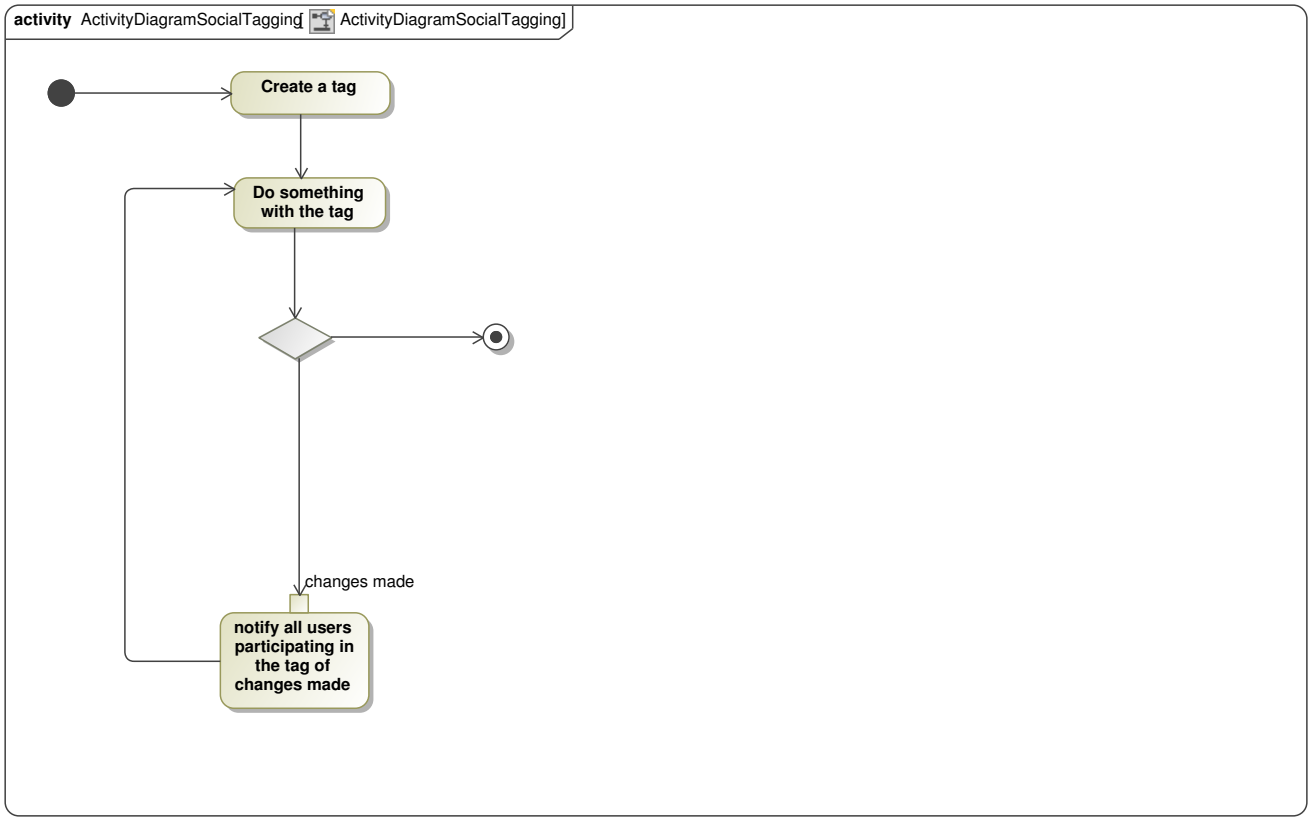
- 5.1 User should be able to access and or search for posts

5.3 Request and Results Data Structures



5.4 Process specifications

5.4.1 Social Tagging



6 Reporting

6.1 Use case prioritization

6.1.1 Critical

- 6.1 viewReports
- 6.2 notifications

6.1.2 Important

- 6.3 plagiarism
- 6.4 netiquette
- 6.5 systemFailure

6.1.3 Nice-To-Have

-

6.2 Use case/Services contracts

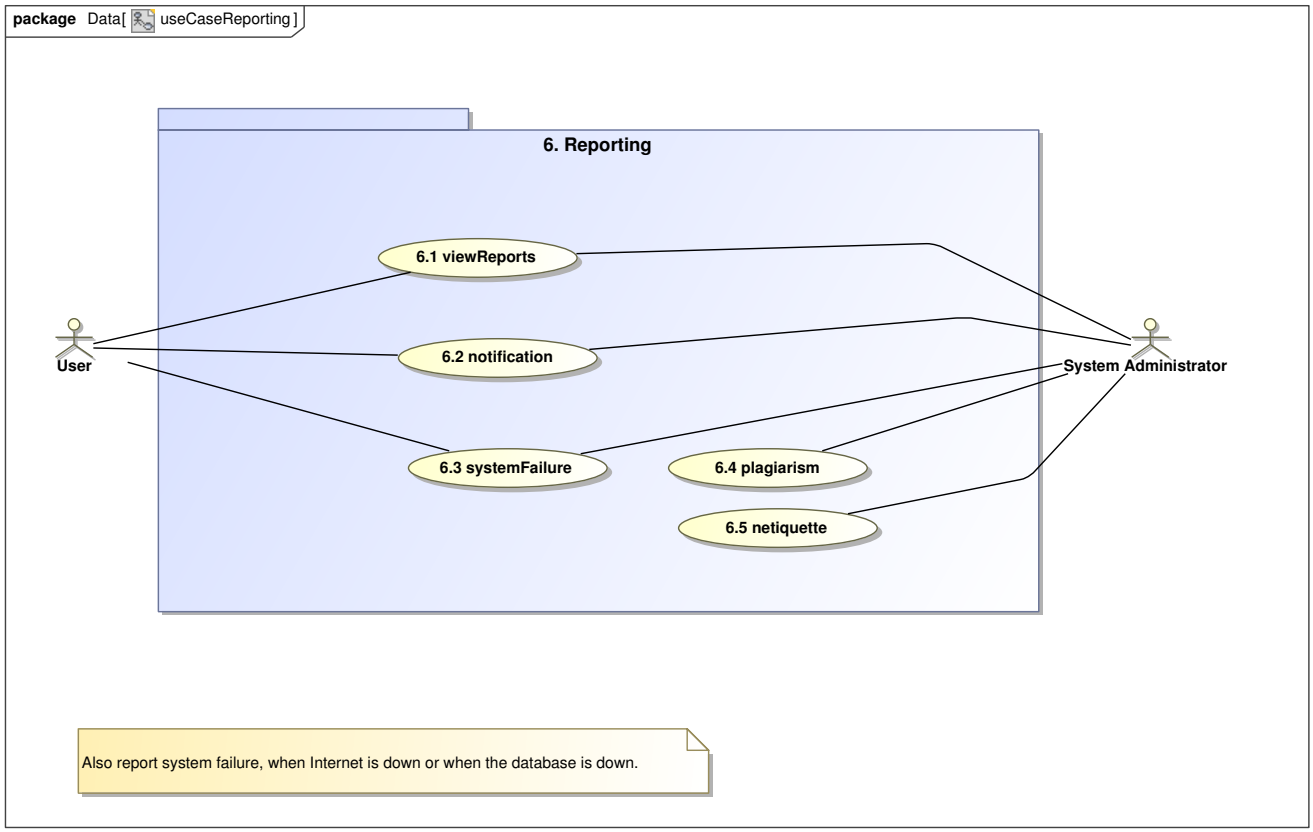
6.2.1 Pre-Conditions

- 6.1 Internet connection and a database required.
- 6.2 Must have logged in to be able to access your notifications.
- 6.3 System must be running, so to generated updated reports.

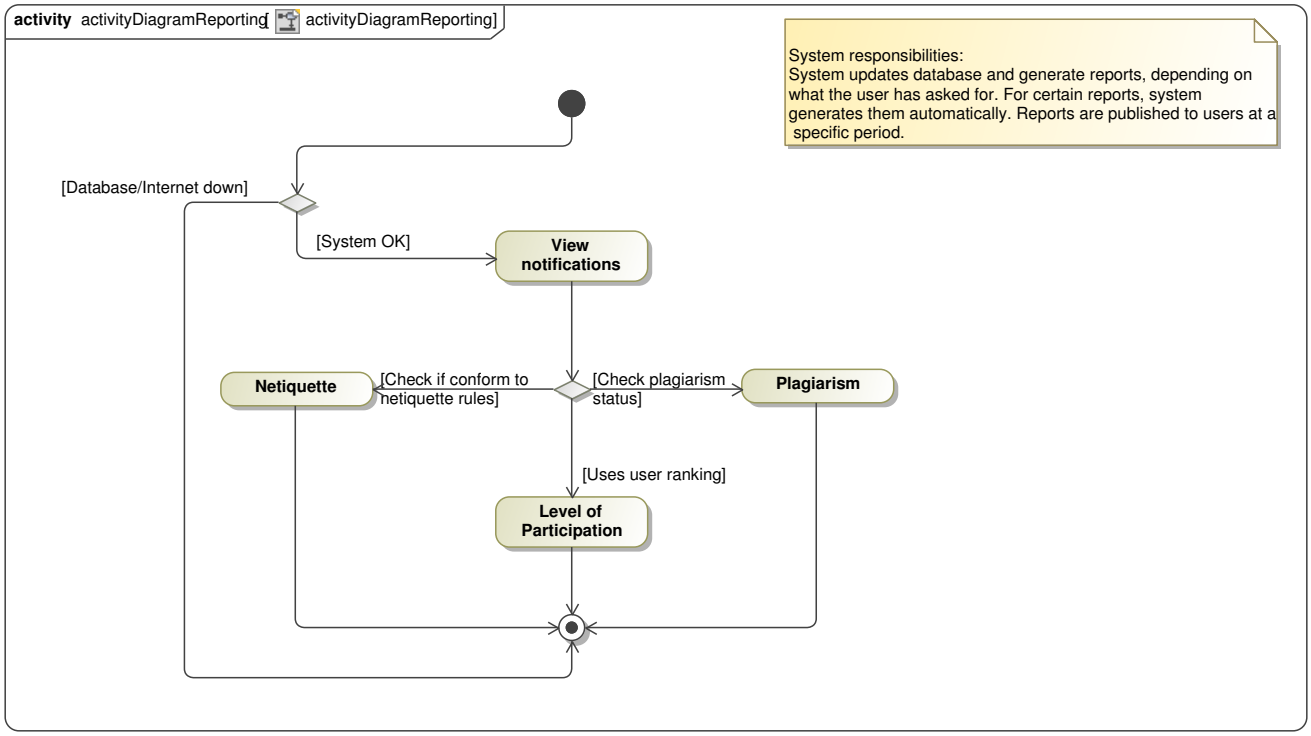
6.2.2 Post-Conditions

- 6.1 System publishes reports to users, on request or when necessary.
- 6.2 Activity blog, see what got updated and by who and so forth.
- 6.3 Logged in users will be able to see their PLAGIARISM STATUS and if their post comfrom with NETIQUETTE rules

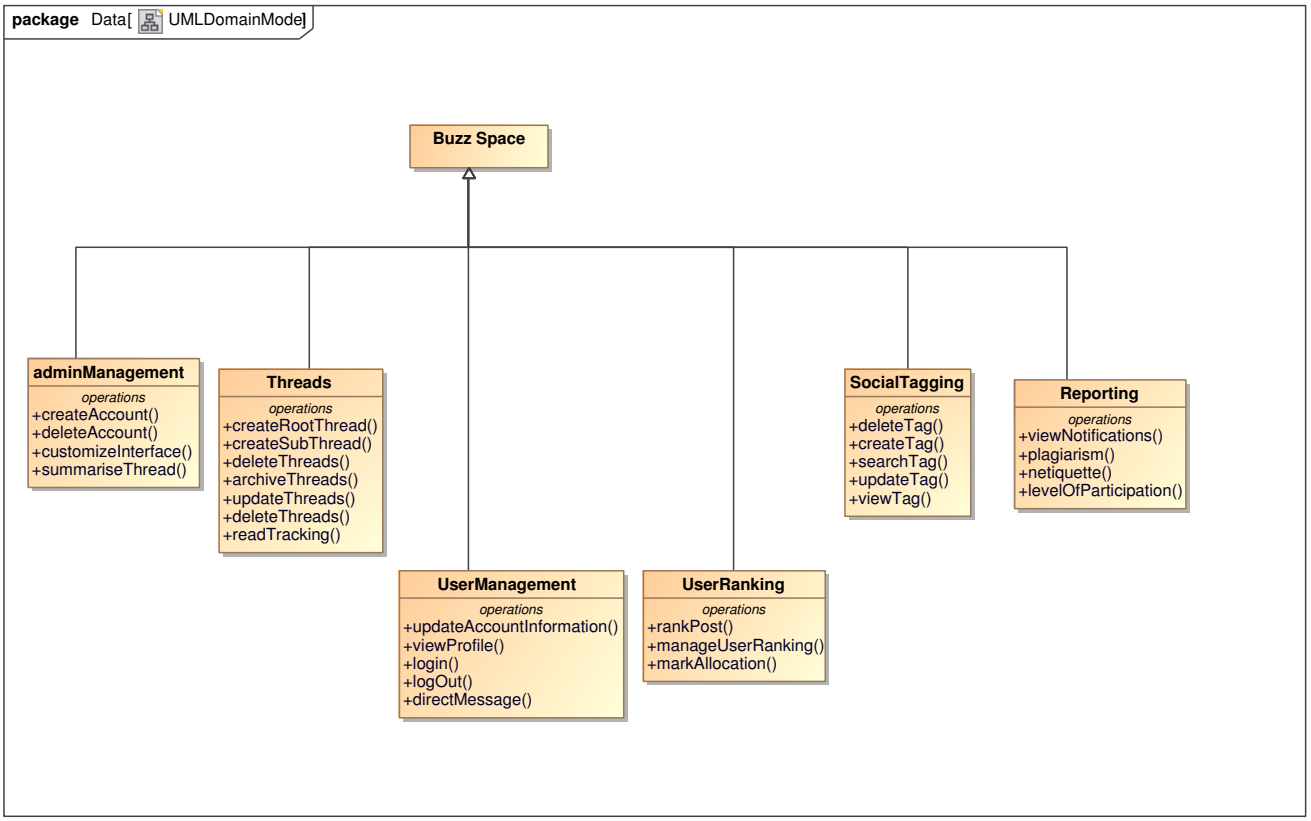
6.3 Request and Results Data Structures



6.4 Process specifications



7 Domain Model



8 High Level Use Case Diagram