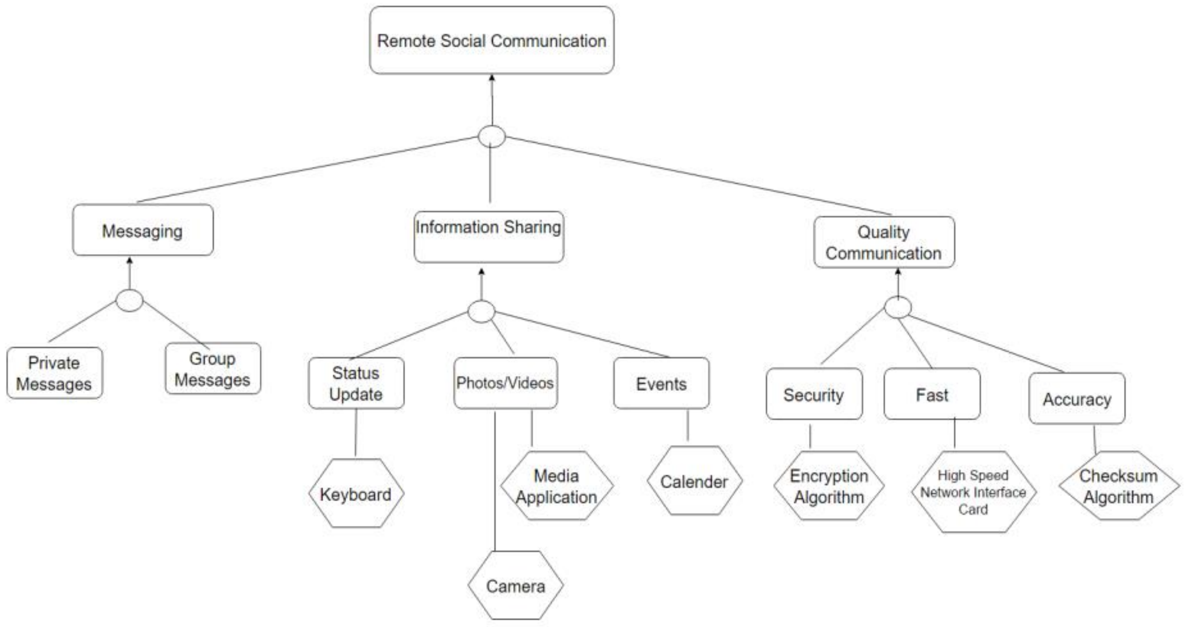
Facebook

Chandrashekar, Jeevan

The application I choose for exercise is Facebook. Facebook is an online social networking service. The website makes it easy for you to connect and share with your family and friends. Our top level goal is; remote social communication. The goal diagram below depicts our top level goal with several sub level goals.



Data Dictionary

Term	**Description**
Status update	A statement, pictures or videos that describe anything you want to say that can be posted to Facebook page, visible by others.
Fast	When a status, a sharing or a message has been sent, it should not take long to be seen by the recipient.
Accuracy	Information is sent to intended address
Security	Information be well protected
Encryption Algorithm	Algorithm used to increase the security
Checksum Algorithm	Algorithm used to ensure the accuracy
Media Application	Application used to upload media files
Events	Description of the time and location of a planned public or social occasion
Private Messages	Individual messaging intended for one specific person
Group Messages	Message intended for a specific group of people
High Speed Network Interface Card	This environmental agent is responsible for fast communication

Goal Diagram Evaluation

Quality

The diagram we generated for Facebook application covered its main purpose that we put it as “remote social communication”. The quality of the top level goal completely meets the purpose of Facebook application. The sub-goals we derived from the top-level goal are covered its significant parts of the application which will enable user: 1 Send messages between two people or within a group. 2 Share information like pictures, videos, status, events or links and so on. 3 All the communication should meet the quality as fast, accuracy, and security. For each level of the goals are described with appropriate abstraction to cover its own functionality. Sub-goals are separated well to be more prepared for further derivation. They are relatively independent from each other to fulfill their own purposes. The diagram is not as complete as to cover all the sub-goals to serve Facebook’s top-level goal. Each level of sub-goals could be derived to more sub-goals to achieve its full functionality.

Completeness

The completeness of the diagram reaches a level that covers the essence of the Facebook application. The sub goals could definitely be expanded to fulfill all aspects of the top level goal. There were several changes made during our reviews. For example during our initial review we changed the sub goal “efficient communication” to “quality communication”. We thought the word quality better defined the goal and allowed us to move one of our other sub goals, security underneath it. This was found to be more appropriate as security, in general, is a quality goal anyway. During a second review we removed the sub goal of “links” from “information sharing” as we found other sub goal from information sharing covered this goal. We also found that the information integrity could be put as a sub-goal for “quality communication”.

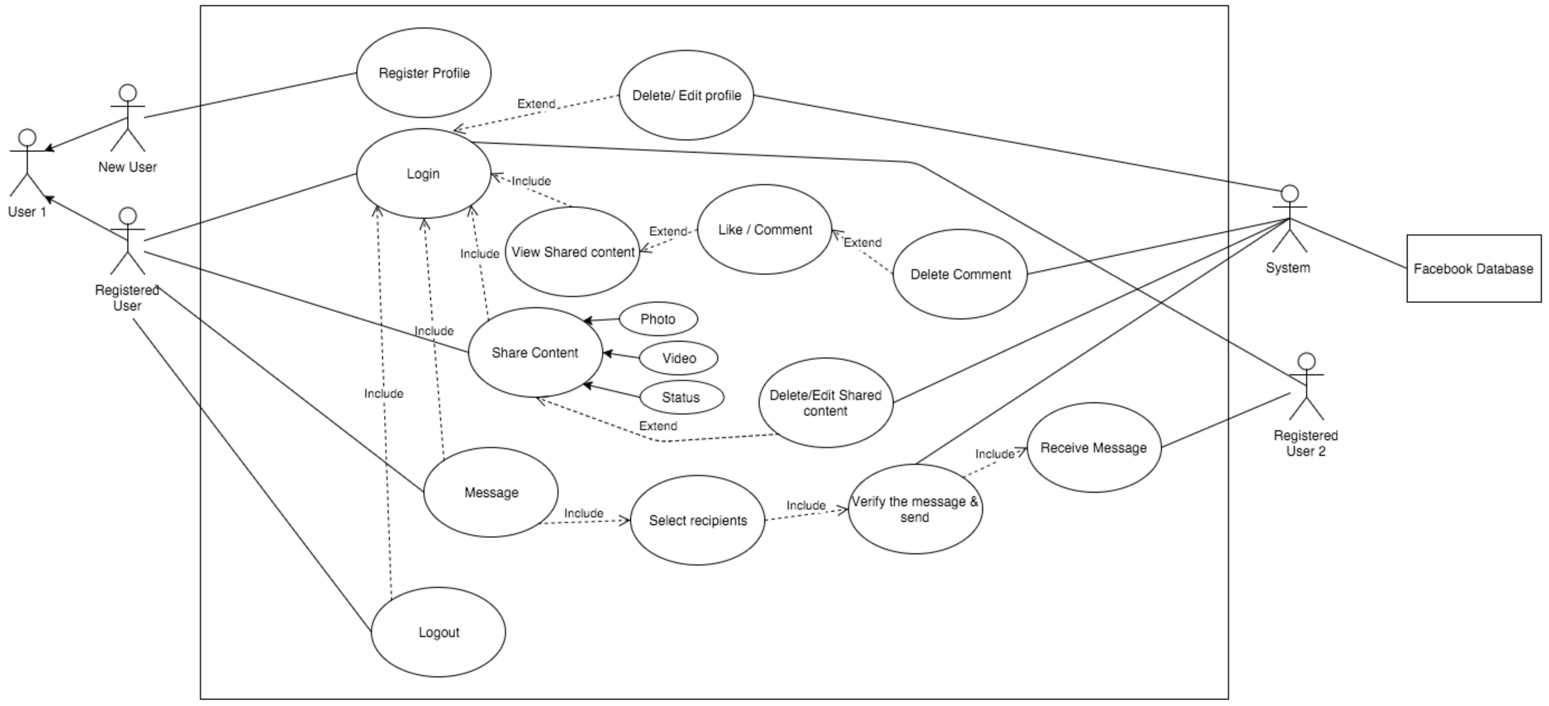
Usefulness

We feel the goal diagram’s overall general usefulness is good. We feel the sub goals and agents listed could help a developer in the design and build of the software modules. As the sub goals we generated in this diagram are separated well in that each branch of the sub-goal is relatively independent. This could be used as guidance to indicate some direction for module development. However, the diagram is still general and not refined enough to make each goal have clear and precise requirements and that each requirement could be assigned to a single agent.

Lessons Learned

This section contains lessons learned from each team member. Overall, as a team we learned that it is not so easy to distinguish between a goal and a feature. We continually discussed this on the creation of our sub goals. The following sections contain each team members own lessons learned from this assignment.

Use Case Diagram:



Use Case Specifications:

Use case 1: Login

Primary Actors: User 1, System.

Goal in context: To gain access to his profile by logging in.

Preconditions: The user should be registered on Facebook and should have valid credentials for the login.

Trigger: When User 1 tries login into his Facebook profile.

Course of events:

User 1	**System Response**	**Facebook Database**
1) Select the login option		
	2) Display the login page	
3) Enter the username and password		
	4) Takes the input from the user and requests the database to forward the users login information.	
		5)Forwards the users login information to the server
	6) Verifies the user entered credentials with the login information and authorizes a login if the information is correct or gives message “incorrect password”.	
7) Login to the profile if the Username and password are correct		

Exceptions:

1. Network: The login will not be possible in case of network failure.
2. Login: The user cannot login into his account without valid proper credentials.

Post condition:
The profile page of the user is displayed.

Priority: High priority, must be implemented.

Use case 2: Send a message

Actors: User 1, User 2, System.

Goal in context: To communicate with other registered users on the network.

Preconditions: Both the actors should be registered users on Facebook and logged in and the message should not contain any potential malware.

Trigger: When User 1 tries to initiate a communication with User 2.

Course of events:

User 1		**System Response**	**Facebook database**	**User 2**
1) Select the message option				
		2) Redirect the user to messaging page and ask him to select the list of recipients		
3) Search for the recipients by typing their names				
		4) Ask the Database to forward the profiles based on names		
			5) Forward the profiles based on names	
		6) Display the profiles forwarded by the database.		
7) Select the recipients and type the message.				
		8) Check the message for any possible malware and forward it to recipients.		
				9) Receive the message.

Exceptions:

1. **Network:** The message will not be delivered in case of network failure.
2. **Login:** The message cannot be sent if the user fails to login to his account.
3. **Malware:** The message will not be delivered if it has any potential malware.

Post Condition:
New message notification is displayed to user

Priority: High priority, must be implemented.

Use case 3: Share Content

Actors: User1, System.

Goal in context: Share content with other registered users on the network.

Preconditions: User should be registered users on Facebook and logged in.

Trigger: When User 1 tries to share a content.

Course of events:

User 1	**System Response**
1) User1 selects the share option	
	2) System will display a prompt to the user to enter text and browse for a file to share (photo/video).
3) User enters the text and browses and selects the photo/video files to share.	
	4) Verify the Content for any inappropriate material and post it.

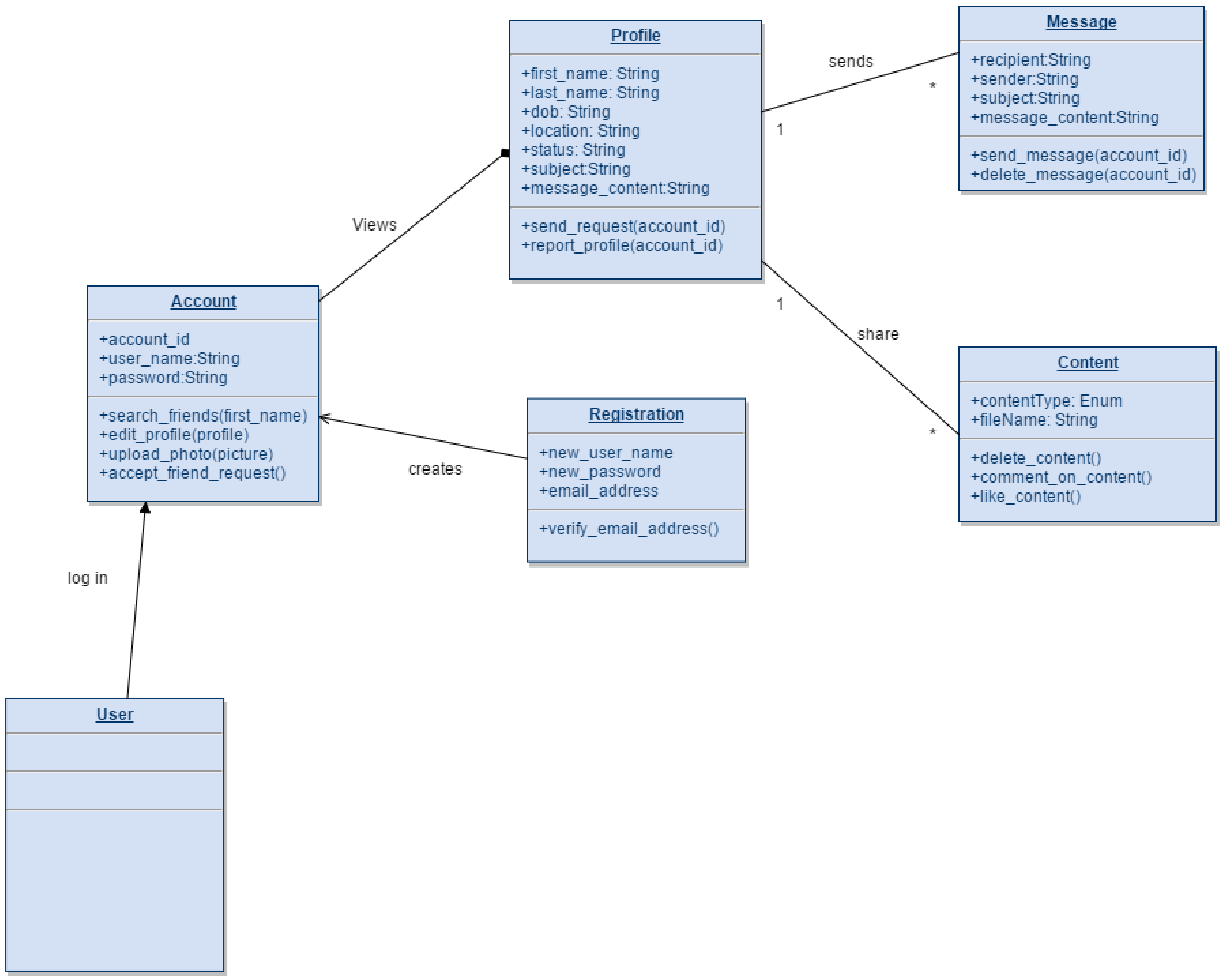
Exceptions:
1. **Network**: The content will not be delivered in case of network failure.
2. **File Format**: The content will not be shared if an unsupported format file is selected.
3. **File Size**: The content will not be shared if the file selected is larger than allowed size.

Post Condition:
Profile page of the user is updated with the content.

Priority: High priority, must be implemented.

Information Model

UML Class diagram



Data Dictionary

Term	Description
Delete/edit profile	To remove/change user data in the database
Delete/edit shared content	To remove/change a piece of shared content
Like/comment	To leave feedback specific to a single shared item
Login	To provide credentials and gain access to the application
Logout	To end a session that was initiated by a login
Message	Text communication shared between two or more users
Recipients	Other users who will receive a message communication
Register profile	To provide new username and password data to the database
Shared content	Uploaded text, photos, or videos for other users to access
Status	A piece of shared content most current for a certain user
User	A person who accesses the application (a new user or registered user)
User, new	A user who has not registered on Facebook
User, registered	A user who already registered on Facebook
Verify/send message	To preview a message and choose to make it available to recipients
View shared content	To access the shared content of another user
Share content	The user can post text/photo/video to share with other users
Creates	Registration of a new User creates a new account
Views	Registered users can view their/other user profiles
Sends	A user sends a message to other

Evaluation

Quality: The use-case model we proposed integrates all the primary functions in the goal diagram and provides a good example of how the application meets its goals. For example, messaging and sharing information are monitored by the system, with other users participating in the process by creating and sending messages. The Facebook application integrates the activities of these human actors with software, thus meeting the goal of messaging identified in the goal diagram. Good use cases should model the actors, software, and their relationships in a realistic use situation, which is true of this diagram for Facebook. An additional check for quality would be to present the diagram to other developers for review, checking that the terms and drawings are understandable from another viewpoint.

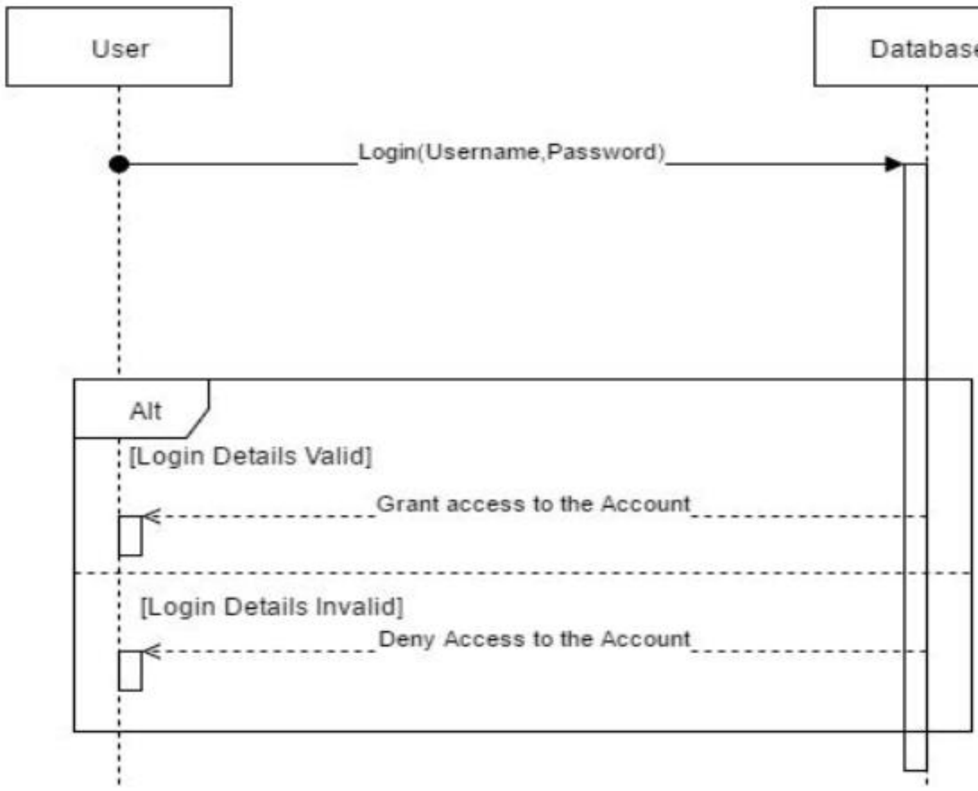
Completeness: The use case specifications presented here can be considered complete, but more details of the entire use case diagram would be necessary to ensure the model is thorough. Each specification can be realized with the actors, triggers, preconditions and steps given in this document. This shows an advantage to identifying individual specification "bubbles" within the diagram: each is a lower level model that plays a role in the overall use case model. While completeness of these low-level models are more easily recognized, it can be seen that the same level of detail should be applied to specify the rest of the "bubbles." In addition, the original goal diagram was not meant to cover the Facebook application in its entirety. More goals, and therefore more interactions within the use case, should be considered when working toward completeness.

Usefulness: The use cases and information model should present a helpful representation of the system that avoids ambiguity and assists developers in future work. This use case diagram is a step in the development that describes the goals in the goal diagram in terms of the agents needed, leading to the class diagram. The use case diagram is useful since it allowed the class diagram to be constructed. The class diagram, then, is a further representation of the system that is one step closer to software implementation. The classes presented here are useful be-cause they are those that would be implemented, in an object oriented design for instance. Attributes and methods could be defined next, referring to the UML class diagram because it accurately models the system and its interactions.

Design review:

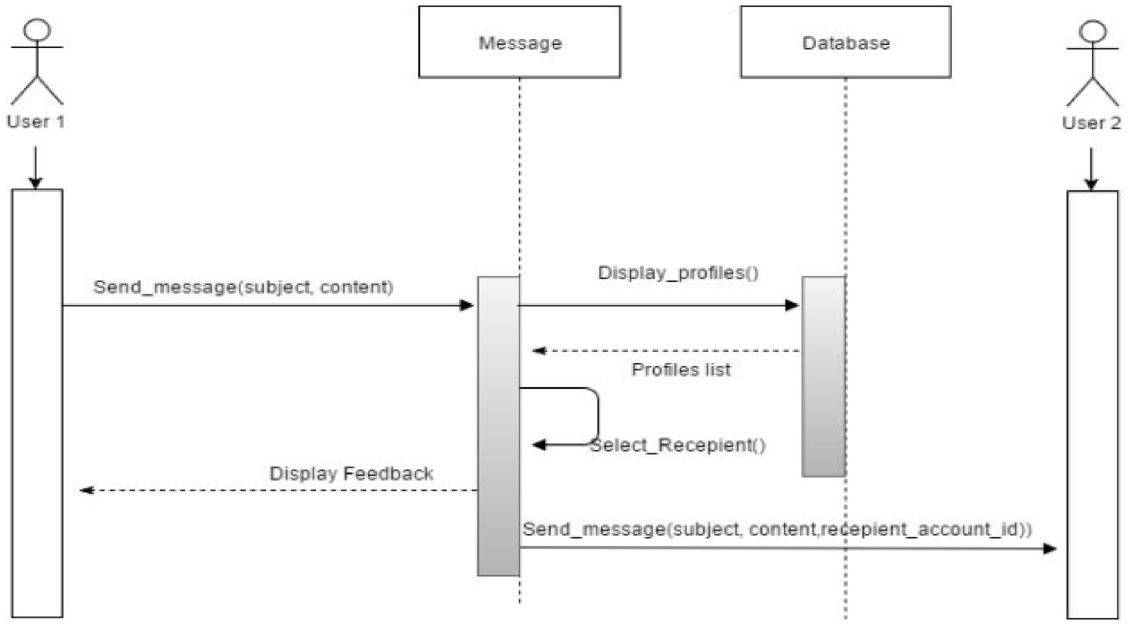
Since this document would be an intermediate step in the modeling process, the team learned the importance of making one's work usable to other developers. Given the goal diagram from a different team, our group needed to further specify the Facebook system into use cases and an information model. The usefulness of a data dictionary quickly became clear, as it was necessary to understand the terms and structure of the goal diagram. After reviewing the entire document from the previous team, the model was clear and ready to be expounded by use cases. The team had to take care in specifying the use cases, as a class diagram and further modeling would be the next development step. This team's addition to the data dictionary, as well as the discussion given in this document, were provided to assist other readers on future development steps. The review process also proved to be of importance, as work done by individual members was often improved by collaboration. Review within the team and critique from external sources were essential to refining the models in this deliverable.

Login Sequence Diagram:



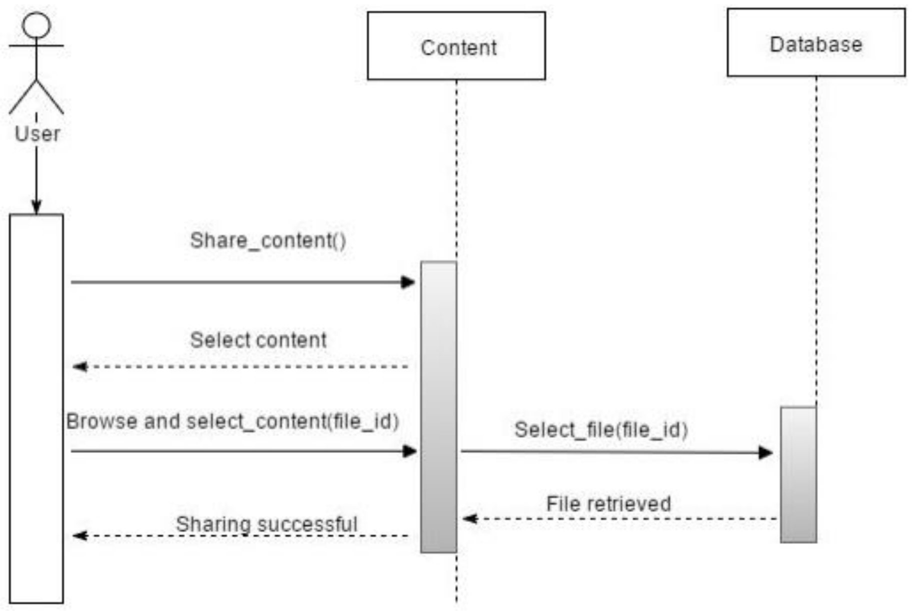
Changes made to the Originating Document:
The user class doesn't have any attributes but now we have added username, password as attributes and login as a method (username,password).

Send Message Sequence Diagram:



Changes made to the Originating Document:
We have added display_profiles as a method in database class and select_receipient, Send_message(subject, content, recepient_account_id) as methods in the message class.

Share Content Sequence Diagram:



Changes made to the Originating Document:
We have added Share_content and Browse and select_content(file_id) as methods in content class and select_file(file_id) as a method in the Database class.

Traceability Table:

Traceable Item	Kind of Item	Item Traced From	Item Traced to	Description
Login	Use Case	Requirements	Class diagram, Sequence Diagram	Authenticatio n of user
Register Profile	Use Case	Requirements	Class diagram	User creating a new account
Share content	Use Case	Requirements	Class diagram, Sequence Diagram	Sharing content (pictures/stat us) with other users
Message	Use Case	Requirements	Class diagram, Sequence Diagram	Used to communicate with the recipient.
Logout	Use Case	Requirements	Class diagram	Process of concluding user application
View Shared	Use Case	Requirements	Class diagram,	View content

Content			Sequence Diagram	shared by other users
Delete/Edit Profile	Use Case	Requirements	Class diagram	Make changes or delete an account
Like/Comment	Use Case	Requirements	Class diagram	Like and make a comment on other user's post.
Delete Comment	Use Case	Requirements	Class diagram	Remove a previously made comment
Select Recipient	Use Case	Requirements	Class diagram, Sequence Requirements Diagram	Select a recipient to send a message
Receive Message	Use Case	Requirements	Class diagram, Sequence Diagram	To be able to receive a message sent by other users
Verify the message and send	Use Case	Requirements	Class diagram, Sequence Diagram	Check for malware before sending the message
Delete/Edit Shared Content	Use Case	Requirements	Class diagram	Make changes or remove any shared content
Photo	Use Case	Requirements	Class diagram	Type of a file
Video	Use Case	Requirements	Class diagram	Type of a file
Account	Class	Requirements and Use Case Diagram	A public profile of an user's interest and activity.	
Account_id	Attribute	Class Diagram	Unique value associated with each account	
User_name	Attribute	Class Diagram	Sequence diagram, Class diagram	An identification used by users to access the

				account
Password	Attribute	Class Diagram	Sequence diagram, Class diagram	String of characters user for user authenticatio n
Search_friends (first_name)	Method	Class Diagram		Search other users using their name
Edit_profile(profile)	Method	Use Case Diagram		Make changes to the user profile
Upload_photo (picture)	Method	Class Diagram		Add a photo to the account
Accept_friend_ request()	Method	Class Diagram		Accept the connection request from other user
Profile	Class	Requirements and Use Case Diagram		Denotes the details of the particular user account
First_Name	Attributes	Class Diagram		User's first name
Last_Name	Attributes	Class Diagram		User's last name
Dob	Attributes	Class Diagram		User's date of birth
Location	Attributes	Class Diagram		User's physical location
Subject	Attributes	Class Diagram		Message subject
Message_content	Attributes	Class Diagram	Sequence diagram.	Actual message
Send_Request (Account_id)	Method	Class Diagram		Request other users to connect
Report_Profile (Account_id)	Method	Class Diagram		Report any suspicious user account
Registration	Class	Requirements and Use Case Diagram		Process of enrolling a new user
New_User_Name	Attributes	Class Diagram		Creating a new user identification
New_Password	Attributes	Class Diagram		Creating a new user authenticatio

				n string
Email _Address	Attributes	Class Diagram		User's email address
Verify_Email_ Address()	Method	Class Diagram		Verifying authenticity of email address
Message	Class	Requirements and Use Case Diagram	Sequence diagram, Class diagram	Gives the operations and attributes related to the message.
Recipient	Attributes	Class Diagram	User that receives the message	
Sender	Attributes	Class Diagram	User that sends the message	
Subject	Attributes	Class Diagram	Sequence diagram.	Subject of the message
Message _Content	Attributes	Class Diagram	Sequence diagram.	Actual message
Send_Message (subject,content)	Method	Class Diagram	Sequence diagram, Class diagram	Send the message with subject and content
Delete_Message (account_id)	Method	Class Diagram		Delete a message
Content	Class	Requirements and Use Case Diagram	Sequence diagram, Class diagram	Specifies the attributes and operations related to content of the message
content _Type	Attributes	Class Diagram		Type of the message content
File_Name	Attributes	Class Diagram		Name of the file to be shared shared
Delete_Content()	Method	Class Diagram		Delete some content content
Comment_on_ Content()	Method	Class Diagram		Make some comments on the shared content content
Like_Content()	Method	Class Diagram		Like some content shared by another user another user
User	Class	Requirements and Use Case Diagram	Sequence diagram, Class	Person that user the

			diagram	application
Display_profile()	Method	Class Diagram	Sequence diagram, Class diagram	Show the profile of the user
Select_ Recipient ()	Method	Use Case Diagram	Sequence diagram, Class diagram	Select a recipient to send the message
Send_Message(subject, content, recipient_account id)	Method	Class Diagram	Sequence diagram, Class diagram	Send message to the particular user
Login(Username, Password)	Method	Use Case Diagram	Sequence diagram, Class diagram	Accessing the account using username and password
Share_content	Method	Use Case Diagram	Sequence diagram, Class diagram	Share some content (picture, video etc)
Browse and select_content(file _id)	Method	Class Diagram	Sequence diagram, Class diagram	Browse through the database and select a file to share
select_file(file_id)	Method	Class Diagram	Sequence diagram, Class diagram	Select a particular file from the database
Database	Class	Requirements and Use Case Diagram	Sequence diagram, Class diagram	Collection of all the data related to the application

Evaluation:
In the given use case specification they have mentioned database in several places but their class diagram doesn't contain the database class hence we decided and added a new class named database

Test Case Specification:

1. Login:
Test Case ID: TC_1
Test Case Name: Login
Preconditions:

1. Application must be installed on device.
2. Alex Tanvi should be registered on Facebook with the email alex@test.com.
3. Alex Tanvi has a valid password, 12345, for login.

Steps:

Step No.	Test Steps	Expected Output
1	Click on Facebook application icon.	Screen with email address or phone number and password text boxes with login button will appear.
2	Enter “alex@test.com” and “12345” in respective text boxes on screen	“alex@test.com” will appear in proper text whereas password will appear as asterisk
3	Click on log in button	Alex Tanvi’s newsfeed screen will be displayed along with options like status, photo, check In, and menu bar.

Note: In originating document login button directs to profile page which we changed to newsfeed page according to actuality.

2. Send a Message:
Test Case ID: TC_2
Preconditions:

1. Alex Tanvi and Robert Douglas should be registered users on Facebook.
2. Alex Tanvi must be logged in to Facebook.

Steps:

Step No.	Test Steps	Expected Output
1	Alex Tanvi selects message option.	Screen containing a search bar for searching people on Facebook.
2	Type “Robert Douglas” in search bar text box	Displays list of users with name “Robert Douglas”
3	Select desired recipient “Robert Douglas”.	Chat window with a text entry box for a conversation with “Robert Douglas” is displayed
4	Type text “Hi Robert, this is a test message.”	“Hi Robert, this is a test message.” will be displayed in text box on screen.

5	Alex Tanvi selects the send button.	"Alex Tanvi: Hi Robert, this is a test message." is displayed in chat window and "Robert Douglas" receives a new message notification.

In addition to this test case, other test cases can be written for the use case to test alternate and exceptional flows. These include instances where the user searches a name for which an account cannot be found, the user search for a blank name, the user attempts to send an empty message or a very long message, the user attaches pictures or videos to the message, and the user includes malware in the message (e.g. URL to a malicious website, malware embedded images, etc.). Creating test cases for all of the variations in user actions will result in a more robust test plan.

3. Share Content:
Test Case ID: TC_3
Test Case Name: Share content
Preconditions:

1. Alex Tanvi should be registered on Facebook.
2. Alex Tanvi should be logged in to Facebook.

Steps:

Step No.	Test Steps	Expected Output
1	Alex Tanvi clicks on share option	Screen containing a share options are displayed
2	Alex Tanvi writes on the post "Feeling great"	"Feeling great" is displayed in the text entry box
3	Alex Tanvi clicks on the post button.	"Feeling great" is displayed on the news feed.

Traceability Table:

Traceable Item	Kind of Item	Item traced From	Item traced to	Description
Login	Use Case	Requirements	Class diagram, Sequence diagram, TC_1	Authentication of User
Share content	Use Case	Requirements	Class diagram, Sequence diagram, TC_3	Select a recipient to send a message
Message	Use Case	Requirements	Class diagram, Sequence diagram, TC_2	Used to communicate with recipient
Select Recipient	Use Case	Requirements	Class diagram, Sequence diagram, TC_3 step 3	Select a recipient to send a message

User_name	Attribute	Class Diagram	Class diagram, Sequence diagram, TC_1 step 2	An identification used by users to access their account.
Password	Attribute	Class Diagram	Class diagram, Sequence diagram, TC_1 step 2	String of characters used for user authentication
Search_friends(first name)	Method	Class Diagram	TC_2 step 2	Search other using their name
Message_content	Attribute	Class Diagram	Sequence diagram, TC_2 step 4	Actual message
Message	Class	Requirements, Use case diagram	Class diagram, Sequence diagram, TC_2	Gives the operations and attributes related to the message.
Send Message(subject, content)	Method	Class Diagram	Class diagram, Sequence diagram, TC_2 step 5	Send the message with subject and content
Content	Class	Requirements, Use case diagram	Class diagram, Sequence diagram, TC_2	Specifies the attributes and operations related to content of the message
Content_type	Attribute	Class Diagram	TC_2	Type of message content
User	Class	Requirements, Use case diagram	Class diagram, Sequence diagram, TC_2	Person who uses the application
Select Recipient()	Method	Use Case Diagram	Class diagram, Sequence diagram, TC_2	Select a recipient to send the message
Login(Username, Password)	Method	Use Case Diagram	Class diagram, Sequence diagram, TC_1	Accessing the account using username and password
Share content	Method	Use Case Diagram	Class diagram, Sequence diagram, TC_2	Share some form of content(photo, status, video, etc)

Login	Test Case TC_1	Use Case Specification: Login, sequence diagram: Login		Steps to be performed to test login operation
Share content	Test Case TC_3	Use Case Specification: Send a Message, sequence diagram: Send a Message		Steps to be performed to test share content operation
Send a message	Test Case TC_2	Use Case Specification: Share conent, sequence diagram: Share conent		Steps to be performed to test send a message
Test Case TC_2, Step 1	Test Case Interaction	Use Case Specification: Send a Message, Steps 1,2	Test Case TC_2	Selection of the messaging option/menu
Test Case TC_2, Step 2	Test Case Interaction	Use Case Specification: Send a Message, Steps 3,4,5,6; Operation: search_friends(first_name)	Test Case TC_2	Searching for recipient
Test Case TC_2, Step 3	Test Case Interaction	Use Case Specification: Send a Message, Step 7;	Test Case TC_2	Selecting recipient
Test Case TC_2, Step 4	Test Case Interaction	Use Case Specification: Send a Message, Step 7; Attribute: message_content	Test Case TC_2	Composing message to be sent
Test Case TC_2, Step 5	Test Case Interaction	Use Case Specification: Send a Message, Steps 7,8,9; Operation: send_message(subject, content)	Test Case TC_2	Sending message and notifying recipient

Note: The only modifications made to the traceability table are the new entries for traceable items from our test cases and additions to the “Trace to” column for corresponding, preexisting entries.

Function Point Evaluation:

Measurement parameter	Count	Weighting factor			Subtotal
Number of user inputs	14	3	**4**	6	56
Number of user outputs	20	4	**5**	7	100
Number of user inquiries	22	**3**	4	6	66
Number files	2	**7**	10	15	14
Number of external interfaces	9	5	7	**10**	90
				Count total	326
				Complexity multiplier	1.07
				Total	348

Note: The complexity multiplier was calculated using an online questionnaire which is cited at the end of this section.

Lines of Code Estimation:
Java (53 loc/fp): 18,444
C++ (50 loc/fp): 17,400
Assembler (119 loc/fp): 41,412
Perl (24 loc/fp): 8,352

Function Point Breakdown:

We considered user inputs as average because while they are mostly text inputs, they are being sent over a network. User outputs were also considered average because they mainly consist of menus and posts being displayed. User inquiries were considered simple because they consist of button and menu selections. Files were considered simple they already exist outside of the application. External interfaces were categorized as complex because of the large variety of hardware devices with which the application interacts and the frequent communication with web servers.

User Input:
3 – Send Message: search query, message body, subject
2 – Login: username, password
2 – Share Content: post/text body, file attachment
1 – Edit: edit/changes text
1 – Comment: comment text body
5 - Registration of User: first name, last name, email address/phone number, confirm email/phone number, password

User Output:
6 – Send Message: search bar, list of search results, chat window, text entry feedback, message displayed in chat window, notification received by recipient
3 – Login: search bar, newsfeed, menu/notification bar
3 – Share Content: share options, text entry feedback, message posted on wall
1 – Edit Content: edited post
1 – Comment: comment
1 – Like: like icon
1 – View Content: content displayed

1 – Logout: home screen displayed after log out
3 - Registered user: search bar, newsfeed, menu/notification bar

User Inquiries:
3 – Send Message: select message option, select recipient, send button
1 – Login: log in button
2 – Comment: click comment button, post button
2 – Share Content: share button, post button
2 – Delete Content: select content, delete button
2 – Edit Content: select edit option, post button
1 – View Content: click on content
2 – Delete Comment: select comment, delete button
1 – Like: click like button
1 – Logout: click logout button
5 – Registration: month menu, day menu, year menu, gender button, sign up button

Note: The use case diagram and three use case specifications were used as guides for estimating functions points for inputs, outputs, and inquiries. Use cases without detailed specifications required more guesswork.

Files:
1 – Photos
1 – Videos
Note: These refer to types of files that the application would access.

External Interfaces:
1 – GPS
1 – Camera
1 – Wifi
1 – SMS
1 – Local files
1 – Contact list
1 – Calendar
1 – Microphone
1 – Facebook database / web server

Sources:
http://groups.engin.umd.umich.edu/CIS/course.des/cis375/projects/fp99/help/help.html
http://groups.engin.umd.umich.edu/CIS/course.des/cis525/js/f00/artan/functionpoints.htm
http://www.qsm.com/resources/function-point-languages-table

Evaluation of Previous Exercises:

The goal diagram and data dictionary provided in the Exercise A clearly present the core functionality of the Facebook mobile application. They state the intent and desired functionality of the application at a high level, but could use further derivation elaboration or expansion to expedite the development of requirements and analysis models. The main fault of the goal diagram is the manner in which the goals are written. Formal statements of the goals are not given outside of the goal diagram, which results in a certain level of ambiguity. However, this is not the fault of the team that designed the goal diagram considering the short-term nature of the assignment.

The use case diagram from Exercise B covers all of the primary interactions between the user and the software-to-be implied by the goal diagram and also contains some use cases which were not directly implied by the goal diagram (e.g. Deleting, Editing, and Liking content). In this respect, the use case diagram would be useful for further specifying these scenarios and interactions found in the software-to-be. Some use cases present in the diagram are confusing in the sense that they seem to merge two sets of interactions. For example, the use cases “Like/Comment” and “Delete/Edit Shared content” would have benefited from being split into two use cases each for the sake of clarity. Another point of confusion stemming from the use case diagram is the actor named “System”. This actor could be interpreted in two ways: an incorrect assignment of the software-to-be as a use case actor or an ill-defined external system interacting with the software-to-be.

The three test case specifications are straightforward and understandable for the most part, providing an idea of what the software and actors are responsible for in different scenarios. The interactions between System and Facebook Database found in the Login use case are somewhat confusing though. It is unclear whether the software-to-be or the Facebook database is authenticating the user. Step 5 of the Login use case also mentions a server, which has not been mentioned elsewhere in the deliverable. The Login use case also specifies that the user is directed to the profile page instead of the newsfeed. It is unclear whether this server is another actor or if the step has just been ambiguously worded. In the Send a Message use case, there is a precondition requiring that no malware can be contained or attached to a message, and there is also a step in which the software-to-be scans the message for malware. Since it is stated that the presence malware would cause an exceptional flow of the use case, the corresponding precondition may not be necessary.

The class diagram lacks completeness, and hence is not as useful as it should be. The User class is presented as a subclass of Account, but does not contain any attributes or operations. Because of this, its purpose is not made clear and there is little distinction what the difference between Account and User is. Furthermore, the data dictionary does not provide descriptions for all of the attributes and operations found in the class diagram. This results in more ambiguity in the class diagram as more interpretation will be required when relying on the class diagram in developing future analysis or design documents. It can also be noted that send_message has a subject parameter, when in actuality messages do not have subjects.

The Exercise C is initially difficult to understand because changes were made to the Exercise B deliverable but were not explained in a cohesive manner. Instead the changes are mentioned sporadically throughout the deliverable, which can cause some confusion. In the Login sequence diagram, the User interacts directly with Database. This seems incorrect at first glance, because Database was a use case actor. Later in the deliverable, it is mentioned that a Database class was created to serve as an interface with the Database actor. Providing a clear and cohesive description of changes made to previous deliverables would have mitigated this initial, incorrect interpretation. There is also ambiguity and/or incorrectness in the Send Message sequence diagram. The same operation, send_message, is used in two different contexts with two different sets of parameters. Otherwise, the sequence diagrams are understandable and provide insight to the purposes and contexts of use for operations found in the class diagram.

The traceability table is incomplete, but this partly due to the incompleteness of the data dictionary. Since the data dictionary did not contain entries for attributes and operations, the team for Exercise C was unable to completely trace them. The traceability table also refers to requirements, which are not present in any of the deliverables. Since there is no mention of goals in the traceability table, it can be interpreted that the team uses goals and requirements interchangeably. The traceability table also lacks enough specificity to make it useful. For example, attributes are traced from the class diagram as whole rather than specific classes. This makes it more difficult and time consuming to see where and how various traceable objects are connected.

This set of deliverables has some overall usefulness as a rough framework on which to base more formal modeling documents. As pointed out, it could benefit from revision and consistency checks. Many of the inconsistencies could arguably be attributed to the short amount of time in which they were prepared as well as assumptions made about preexisting software. While there are indeed weak aspects of some the models and documents, they could be revised fairly easily to be more understandable and useful for future phases of development.

Evaluation:

Our test cases provide a useful basis from which to create a test plan for the three specified use cases. Variations could be developed from our test cases to cover many scenarios related to the corresponding use cases. We feel that our test cases are clear, specific, and understandable to be used during actual testing. When estimating function points, we tried to be as clear as possible in stating from where our estimated function points came. Since neither our use case specifications nor our class diagram were complete, we had to make some rough estimates regarding some input and output values. While it is not a completely accurate estimation of the application's complexity and length, it does serve to provide a reasonable idea of the effort required to implement it.