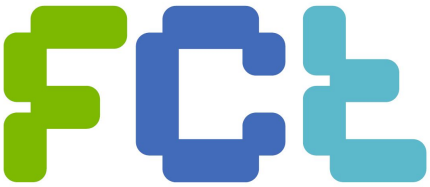


Software Engineering

Part 2

Group:

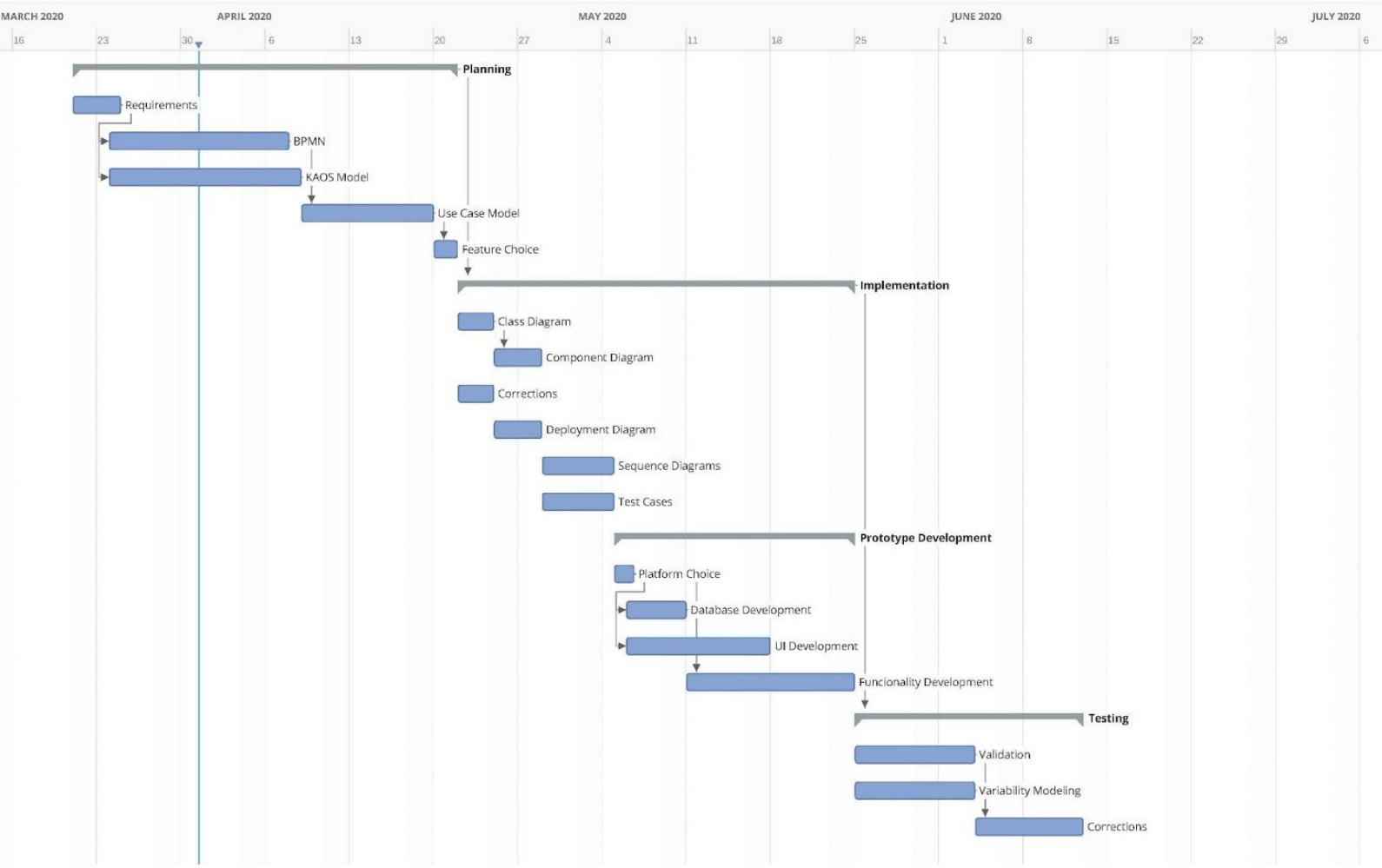
Ana Matos - 49938
António Ferraz - 50110
Hugo Simão - 50266
Rafael Gameiro - 50677

Table of Contents

Introduction

This present report contains part of the modelling of a solution, called Instagram Plus, akin to Facebook's Instagram Social Network.
It contains the Gantt Model chart of the project development, the list of the requirements and the KAOS and BPMN models of all its functionalities.
Finally, it contains a proposal of the features to be developed in the next phase of the project and a use case model of the application.

Gantt Chart


MARCH 2020
APRIL 2020
MAY 2020
JUNE 2020
JULY 2020
16
23
30
6
13
20
27
4
11
18
25
1
8
15
22
29
6
Planning
Requirements
BPMN
KAOS Model
Use Case Model
Feature Choice
Implementation
Class Diagram
Component Diagram
Corrections
Deployment Diagram
Sequence Diagrams
Test Cases
Prototype Development
Platform Choice
Database Development
UI Development
Funcionality Development
Testing
Validation
Variability Modeling
Corrections

Requirements List

Functional Requirements

- Register User
- Login
- Edit Profile
- Follow User
- Block User
- Watch Story
- Make Story
- Delete Story
- Make Post
- React to Post
- Save Post
- Delete Post
- Watch Feed
- Search
- Create Collection of Stories
- Send Message
- Manage Privacy
- Delete Comment

Non-Functional Requirements

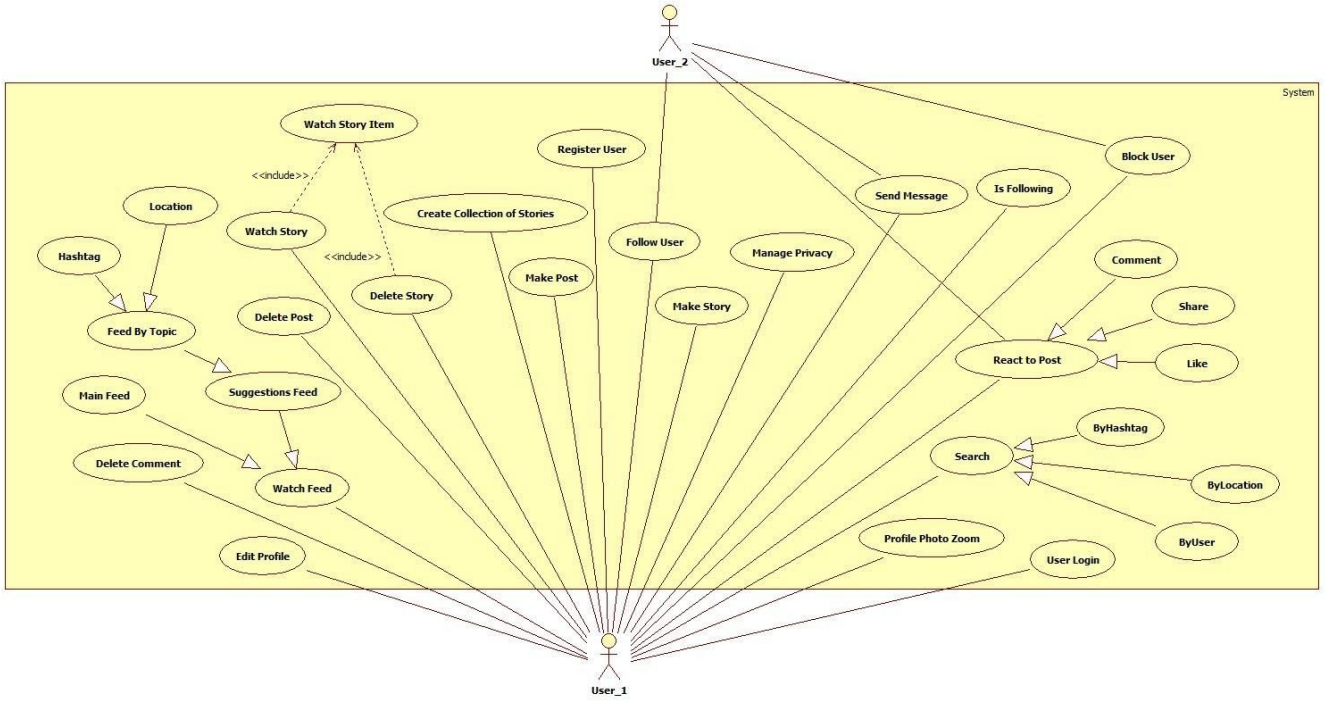
- Availability
- Security
- Data Integrity
- Compatibility
- Usability
- Privacy
- General

New Functionalities

- Zoom in User Photo

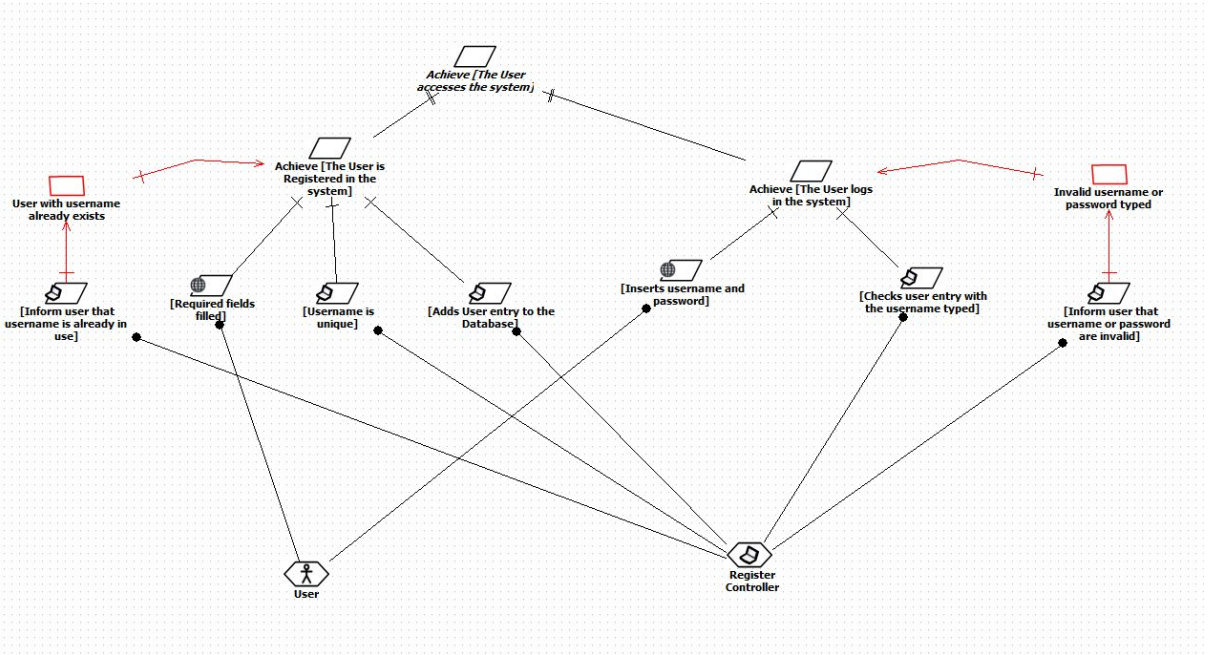
- Check if User is Following

Use Case Model

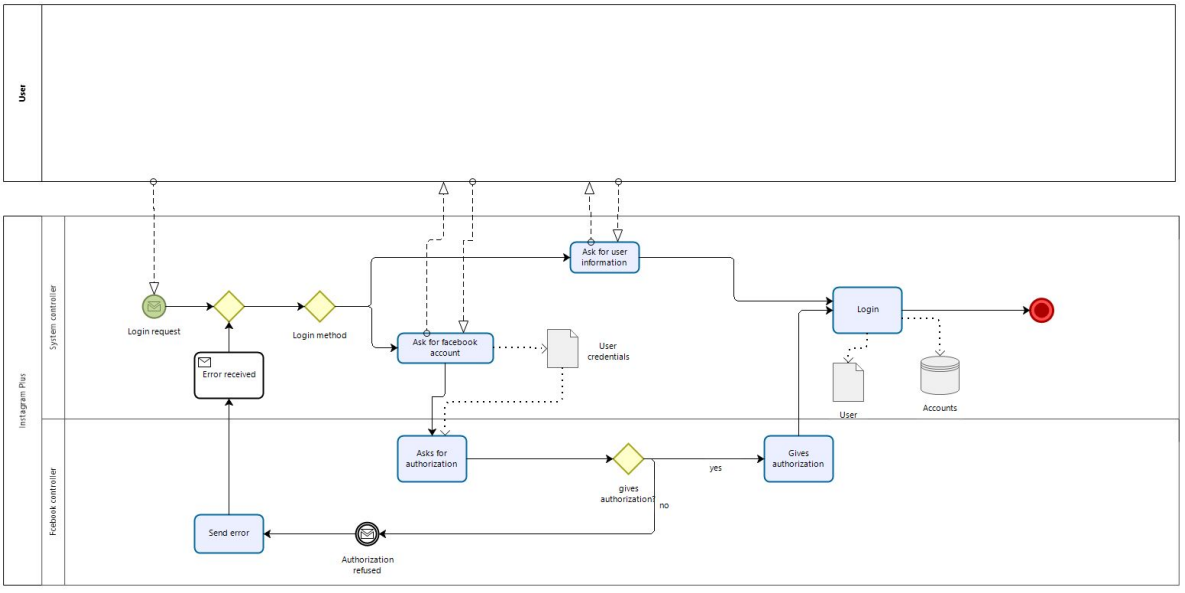


KAOS and BPMN Models

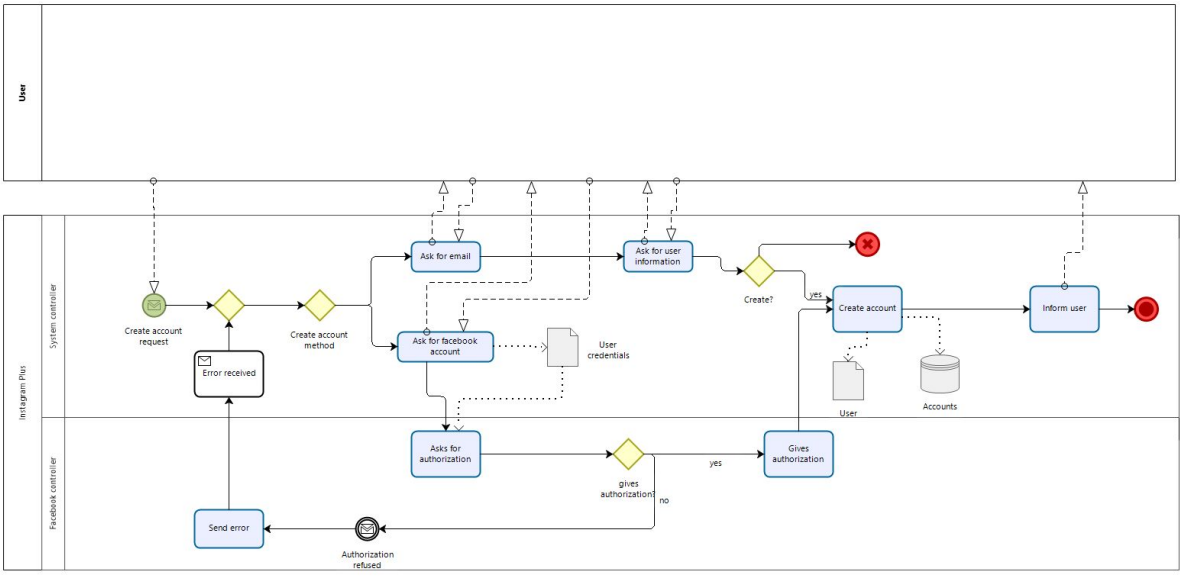
Register User and Login



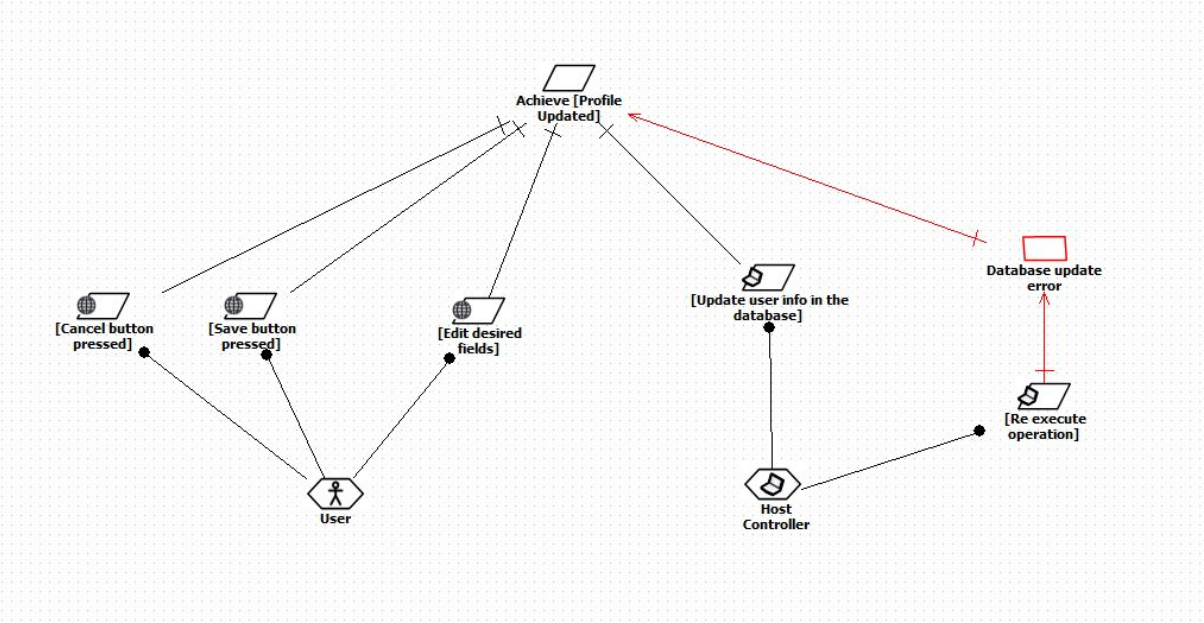
Login

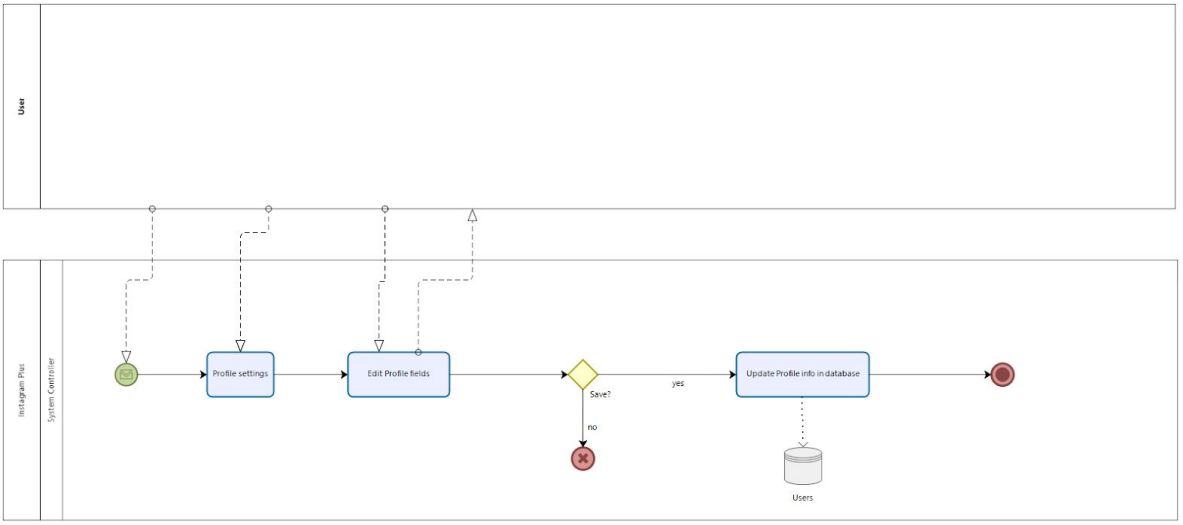


Register

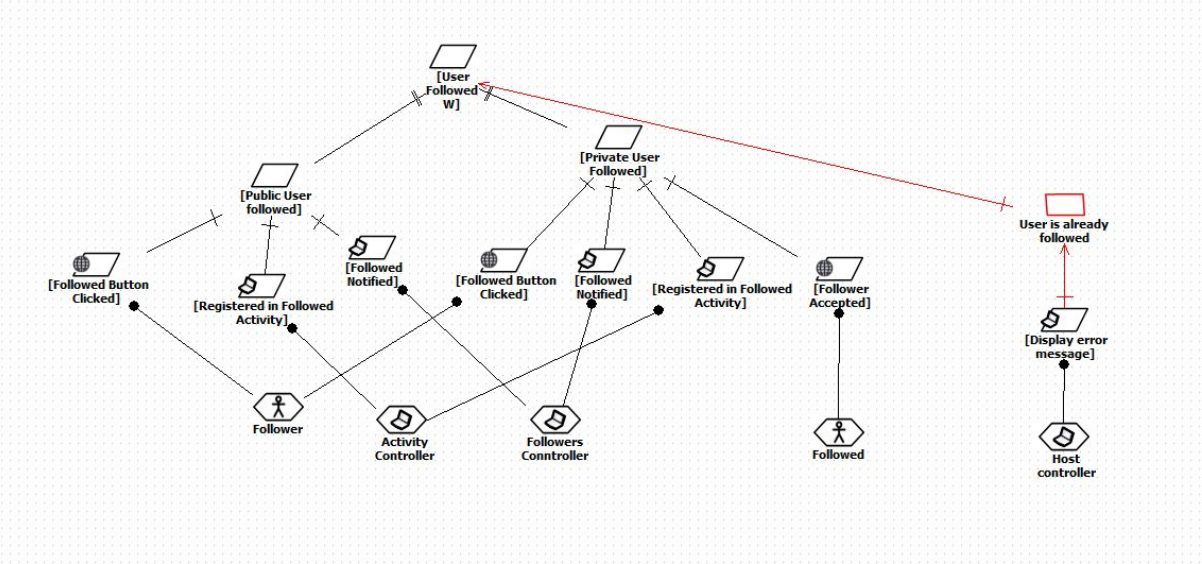

User
Instagram Plus
System controller
Facebook controller
Create account request
Error received
Create account method
Ask for email
Ask for facebook account
User credentials
Ask for user information
Create?
yes
Create account
User
Accounts
Inform user
Asks for authorization
gives authorization?
yes
no
Gives authorization
Send error
Authorization refused

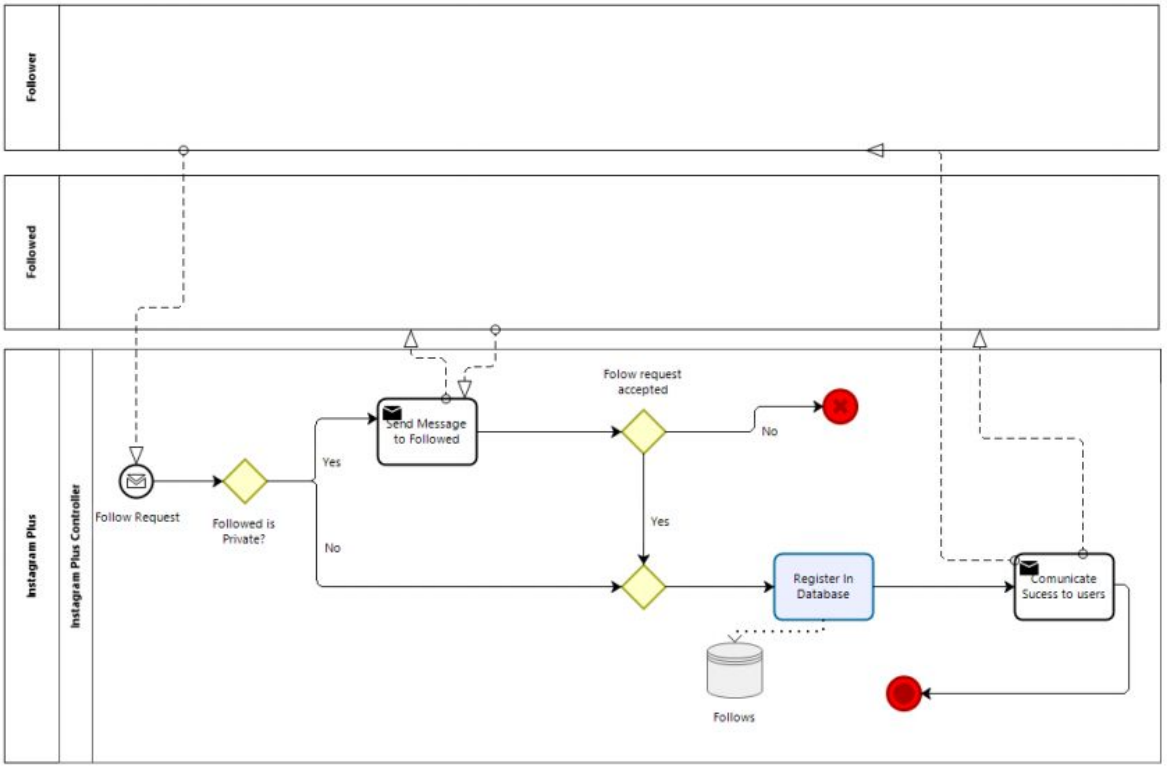
Edit Profile



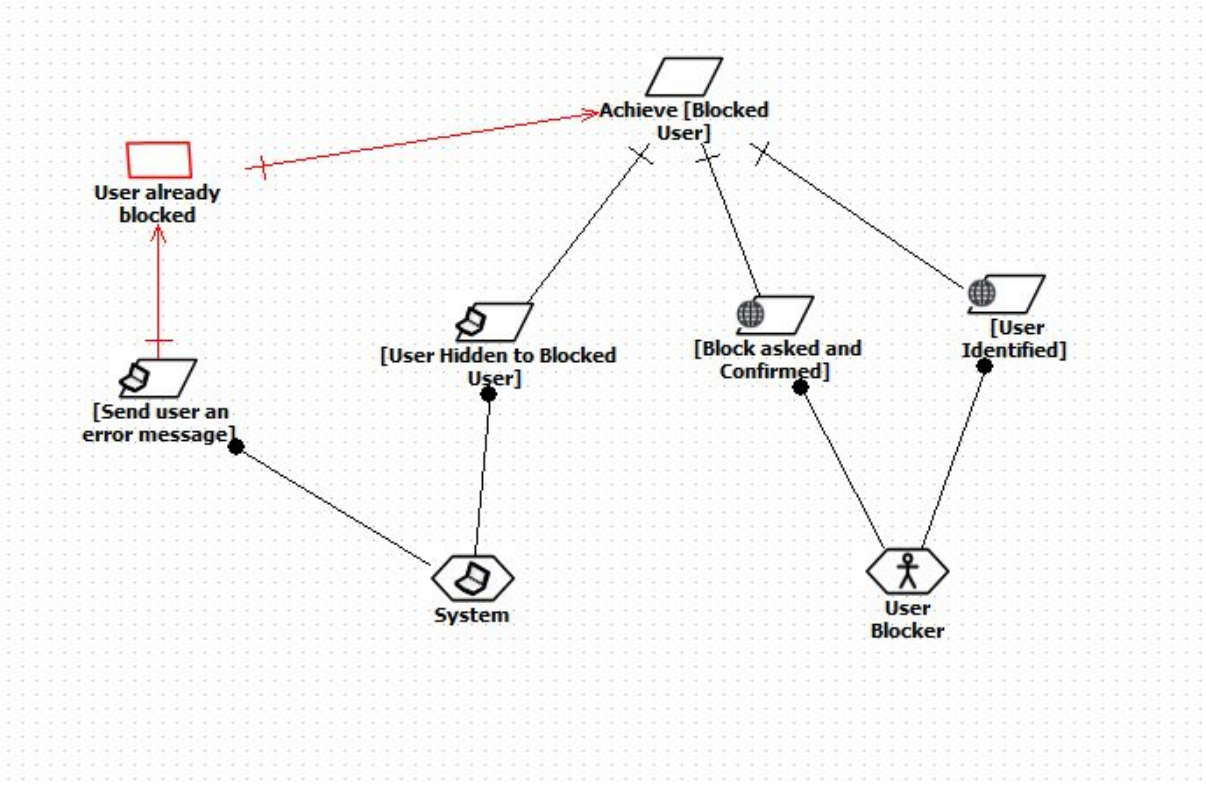


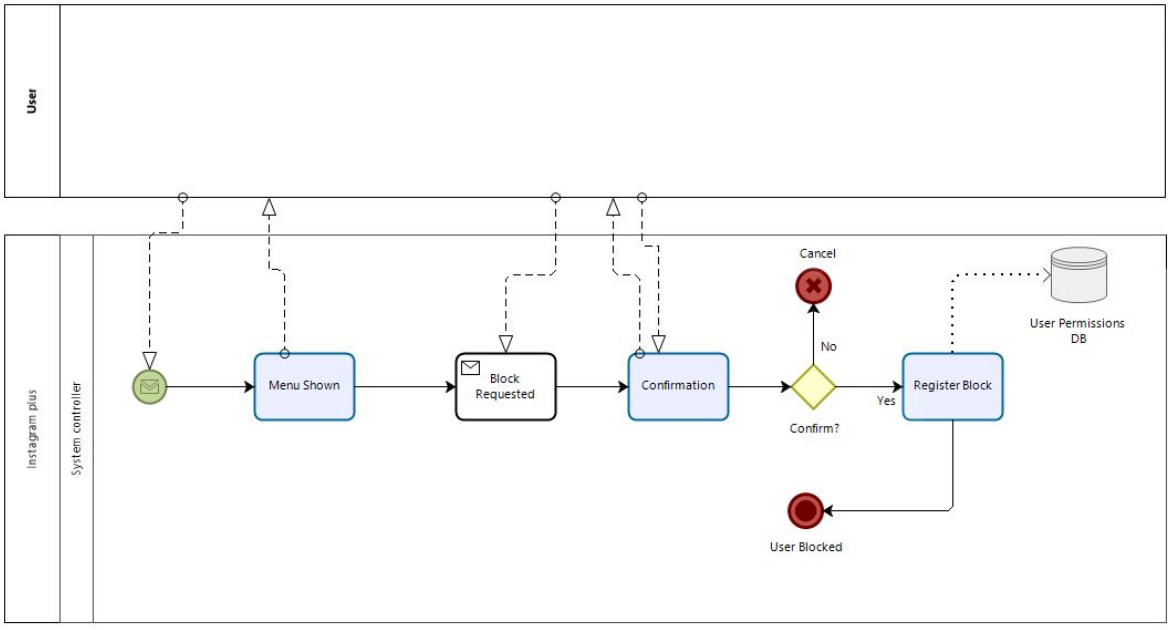
Follow User


[User Followed W]
[Private User Followed]
[Public User followed]
User is already followed
[Followed Button Clicked]
[Registered in Followed Activity]
[Followed Notified]
[Followed Button Clicked]
[Followed Notified]
[Registered in Followed Activity]
[Follower Accepted]
[Display error message]
Follower
Activity Controller
Followers Conntroller
Followed
Host controller

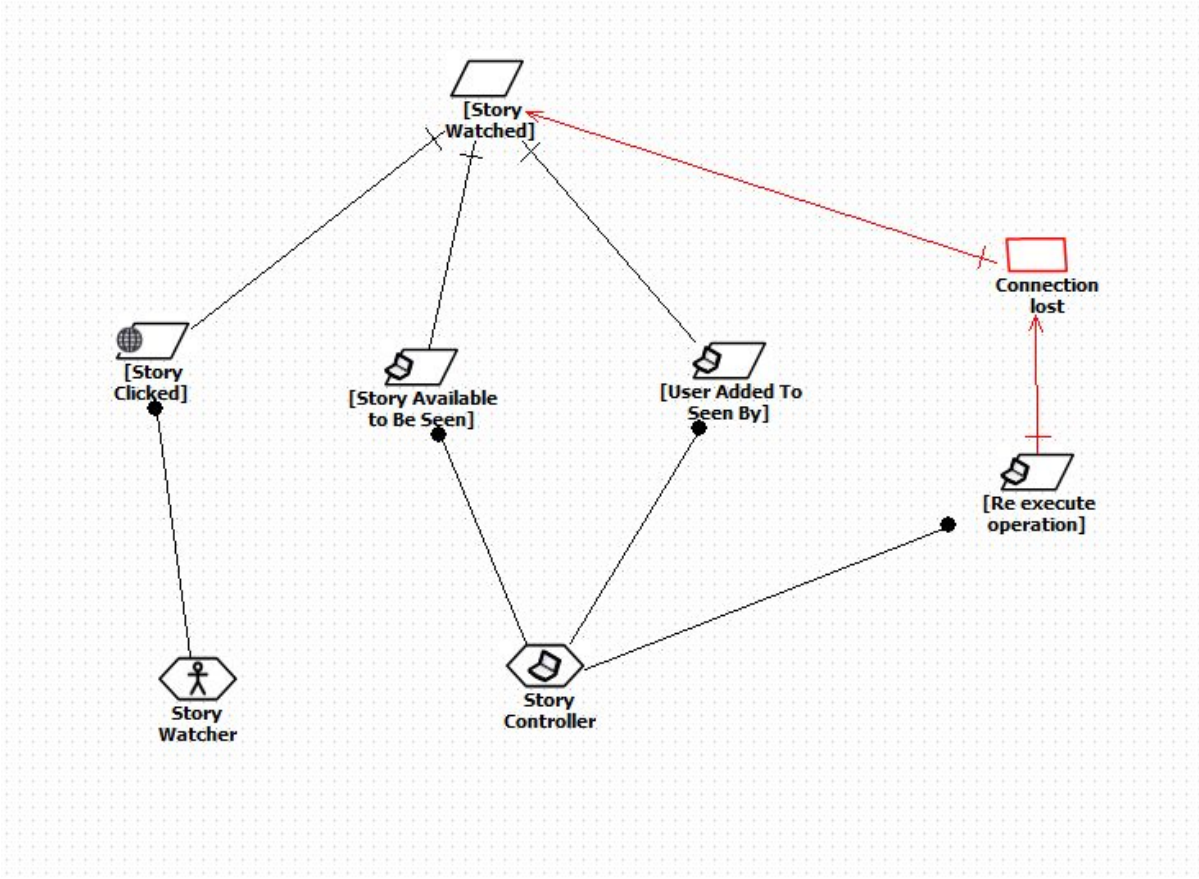

Follower
Followed
Instagram Plus
Instagram Plus Controller
Follow Request
Followed is Private?
Yes
No
Send Message to Followed
Folow request accepted
No
Yes
Register In Database
Comunicate Sucess to users
Follows

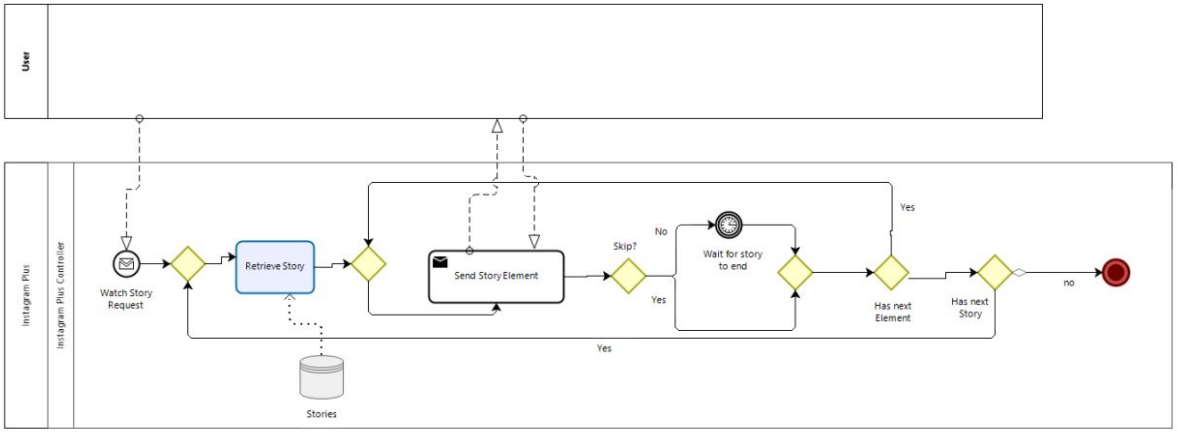
Block User


Achieve [Blocked User]
User already blocked
[Send user an error message]
[User Hidden to Blocked User]
[Block asked and Confirmed]
[User Identified]
System
User Blocker

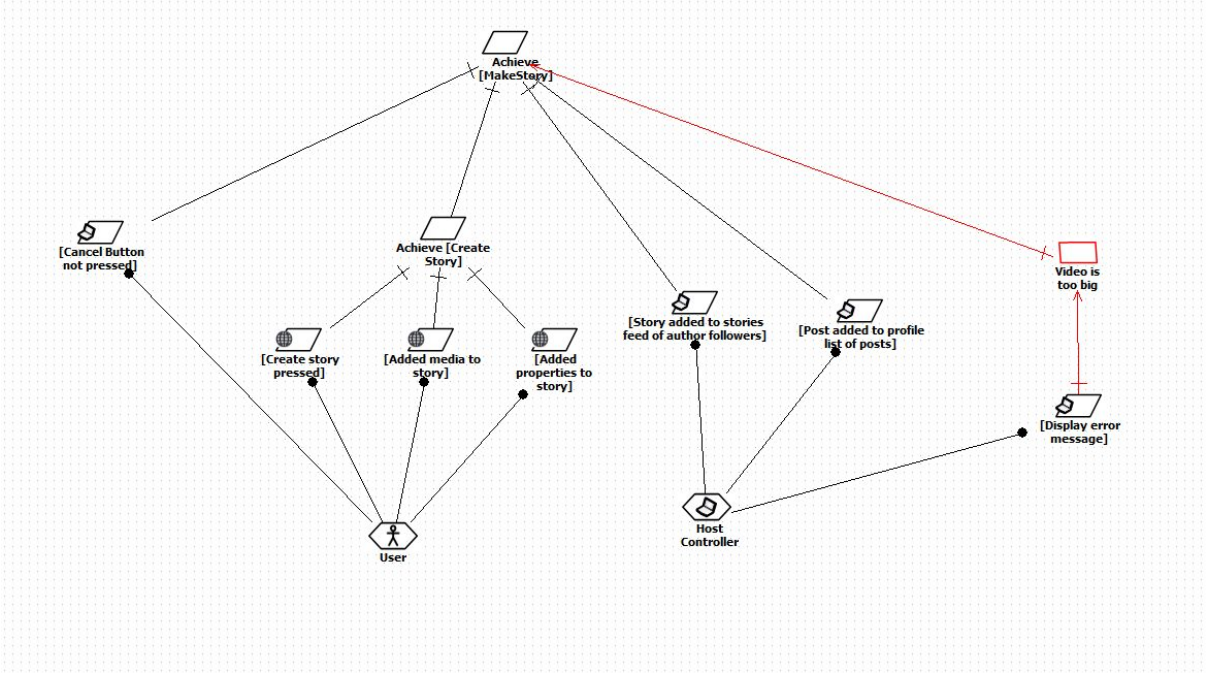

User
Instagram plus
System controller
Menu Shown
Block Requested
Confirmation
Cancel
No
Confirm?
Yes
Register Block
User Permissions DB
User Blocked

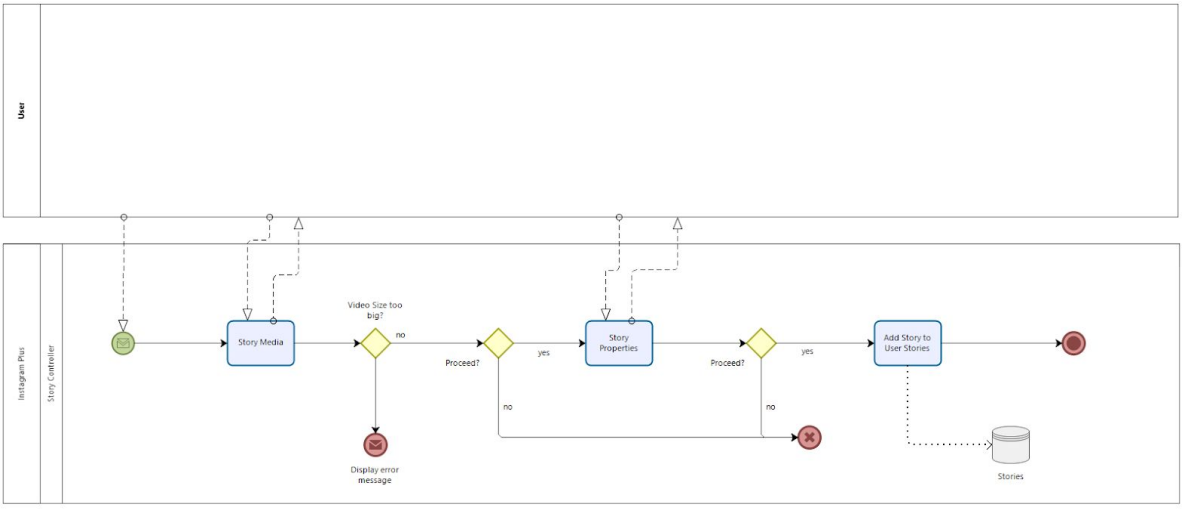
Watch Story


[Story Watched]
Connection lost
[Story Clicked]
[Story Available to Be Seen]
[User Added To Seen By]
[Re execute operation]
Story Watcher
Story Controller

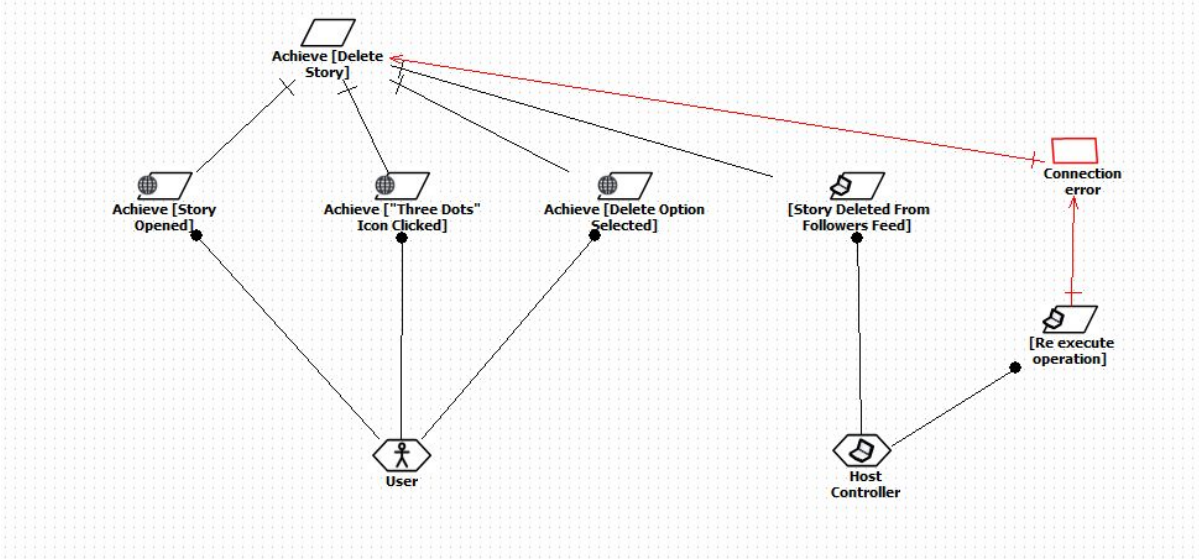

User
Instagram Plus
Instagram Plus Controller
Watch Story Request
Retrieve Story
Send Story Element
Skip?
No
Yes
Wait for story to end
Yes
Has next Element
Has next Story
no
Yes
Stories

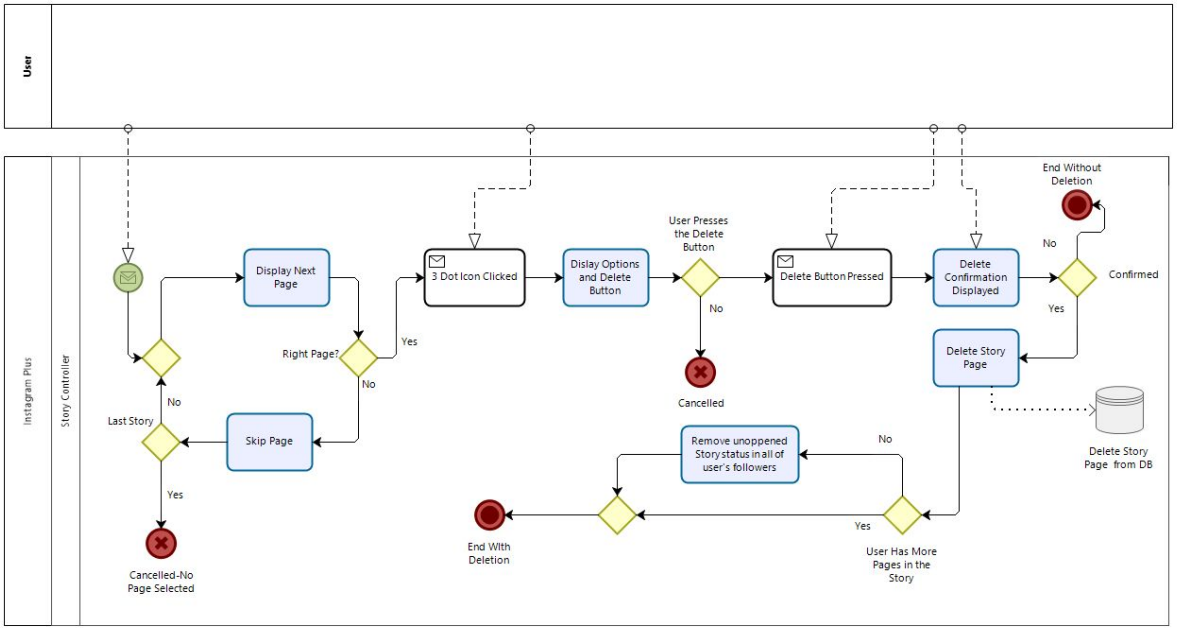
Make Story


Achieve [MakeStory]
[Cancel Button not pressed]
Achieve [Create Story]
[Create story pressed]
[Added media to story]
[Added properties to story]
[Story added to stories feed of author followers]
[Post added to profile list of posts]
Video is too big
[Display error message]
Host Controller
User

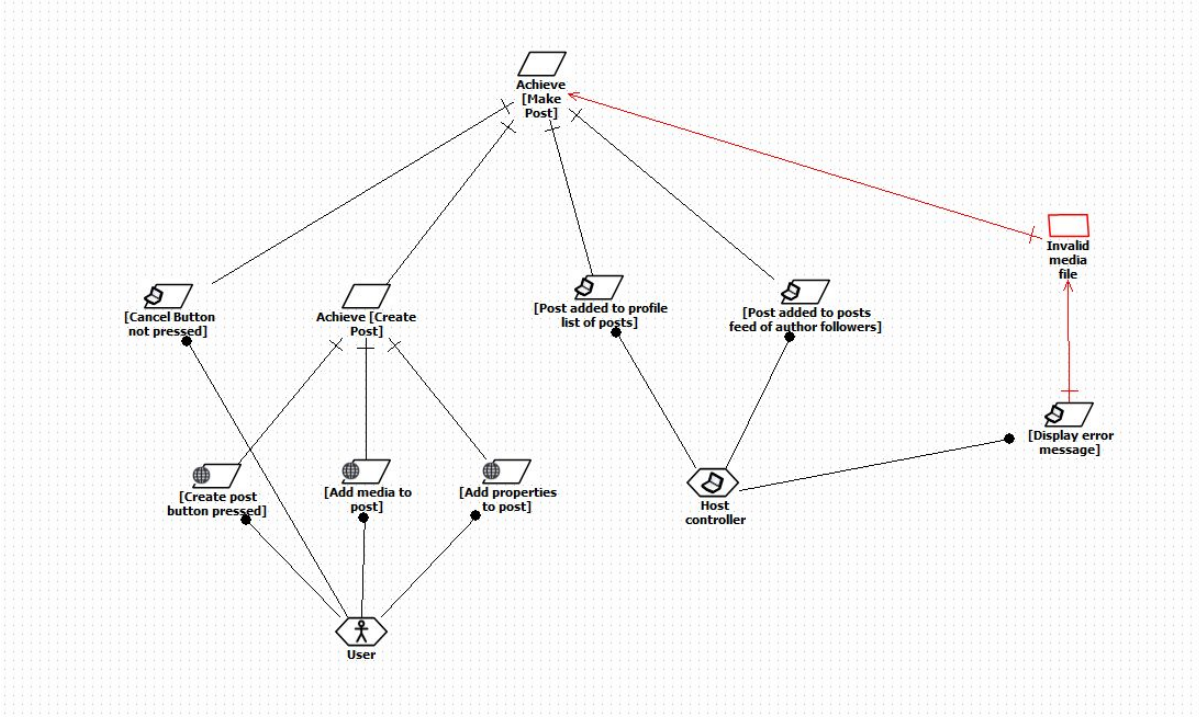

User
Instagram Plus
Story Controller
Story Media
Video Size too big?
no
Display error message
Proceed?
yes
no
Story Properties
Proceed?
yes
no
Add Story to User Stories
Stories

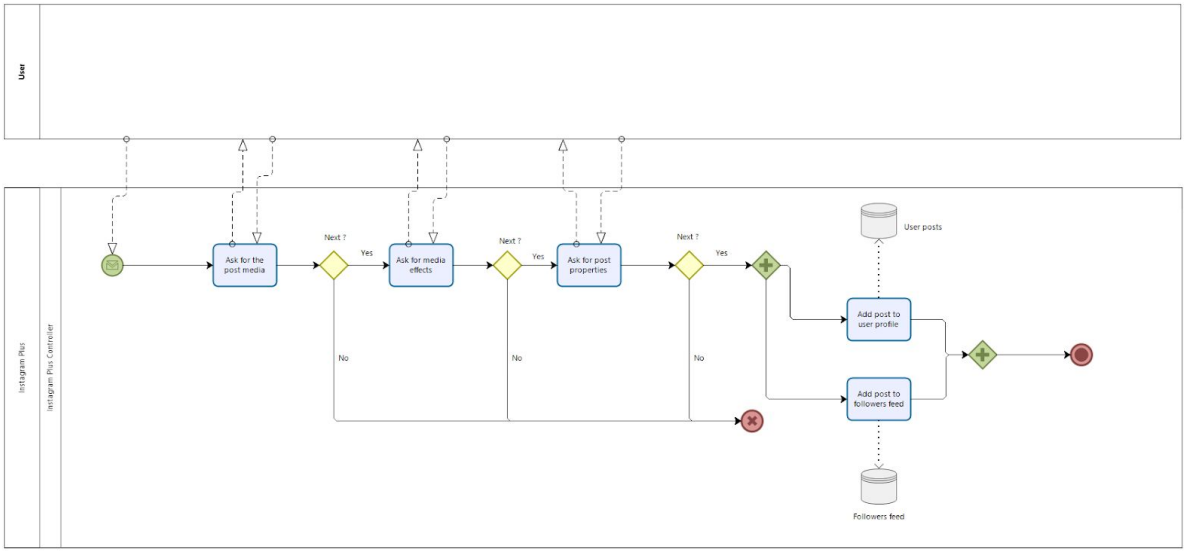
Delete Story



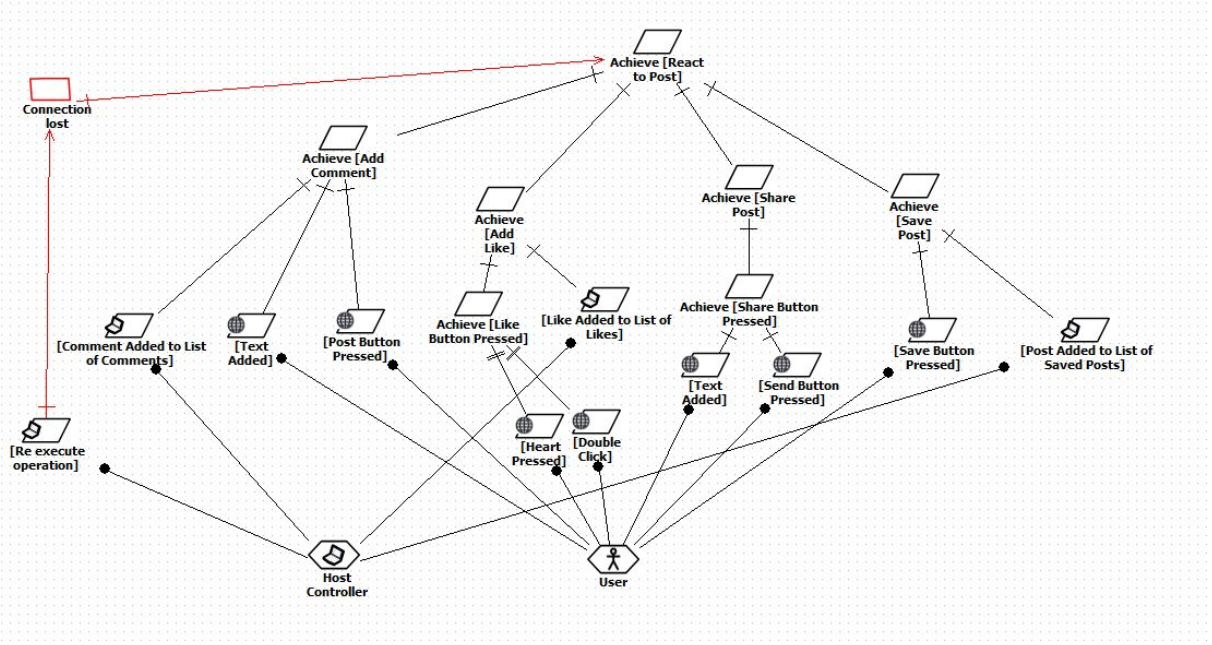


Make Post

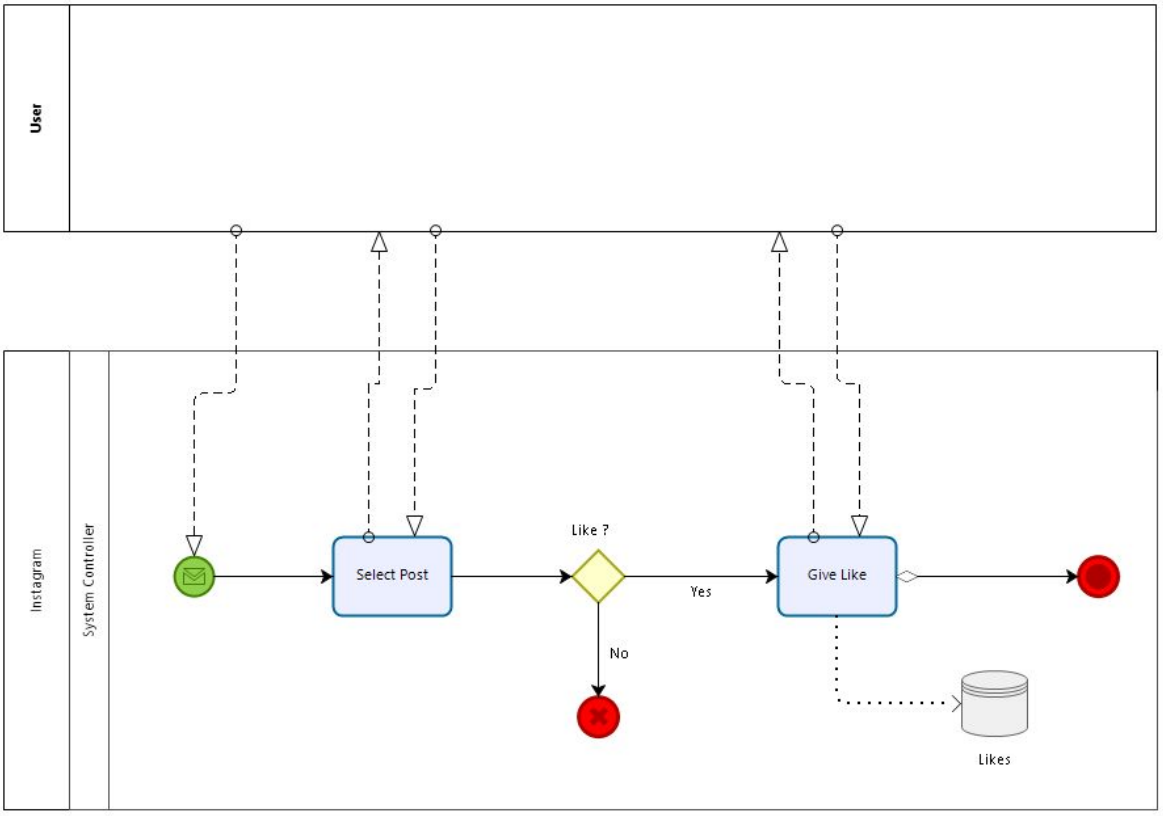

Achieve [Make Post]
Invalid media file
[Cancel Button not pressed]
Achieve [Create Post]
[Post added to profile list of posts]
[Post added to posts feed of author followers]
[Display error message]
[Create post button pressed]
[Add media to post]
[Add properties to post]
Host controller
User


User
Instagram Plus
Instagram Plus Controller
Ask for the post media
Next ?
Yes
No
Ask for media effects
Ask for post properties
User posts
Add post to user profile
Add post to followers feed
Followers feed

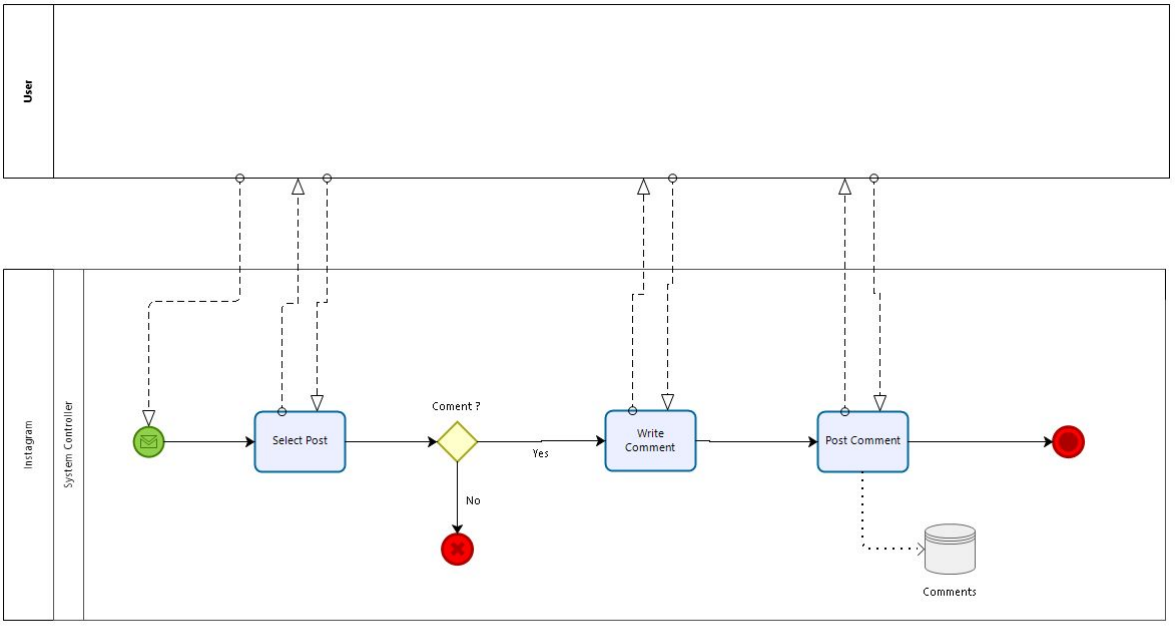
React to Post



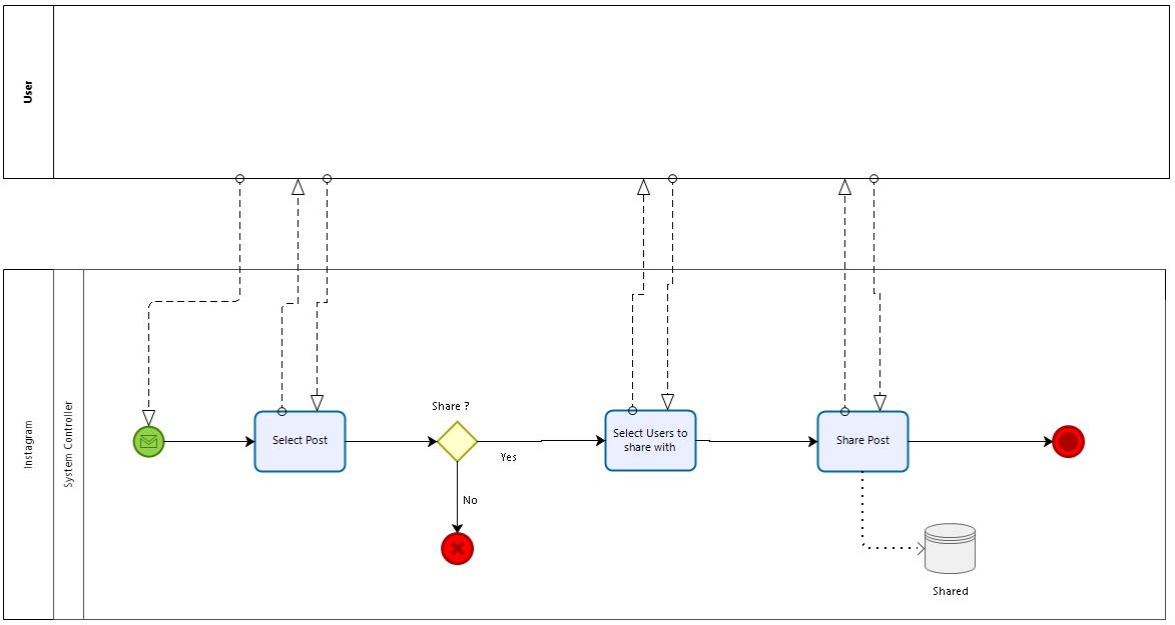
Like



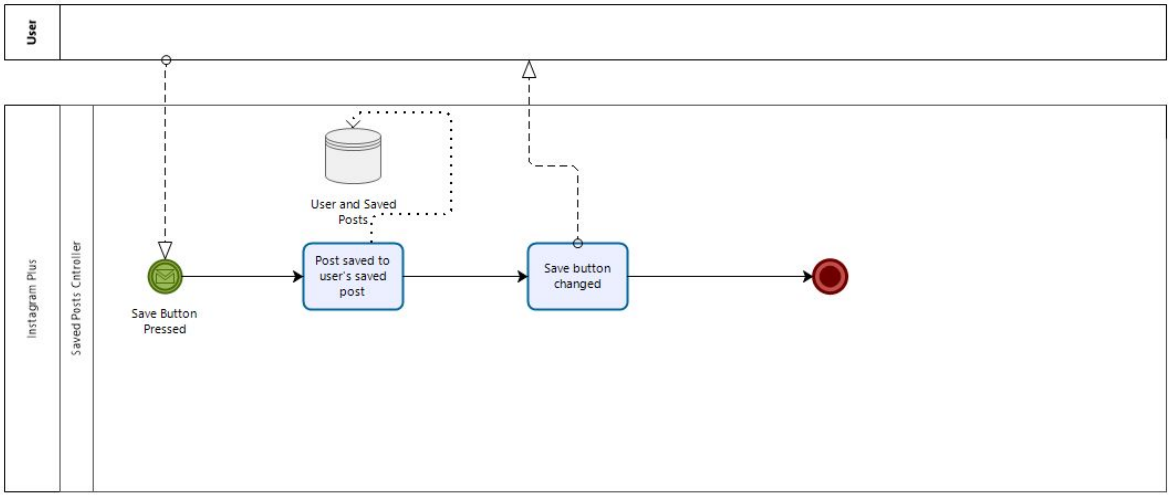
Comment



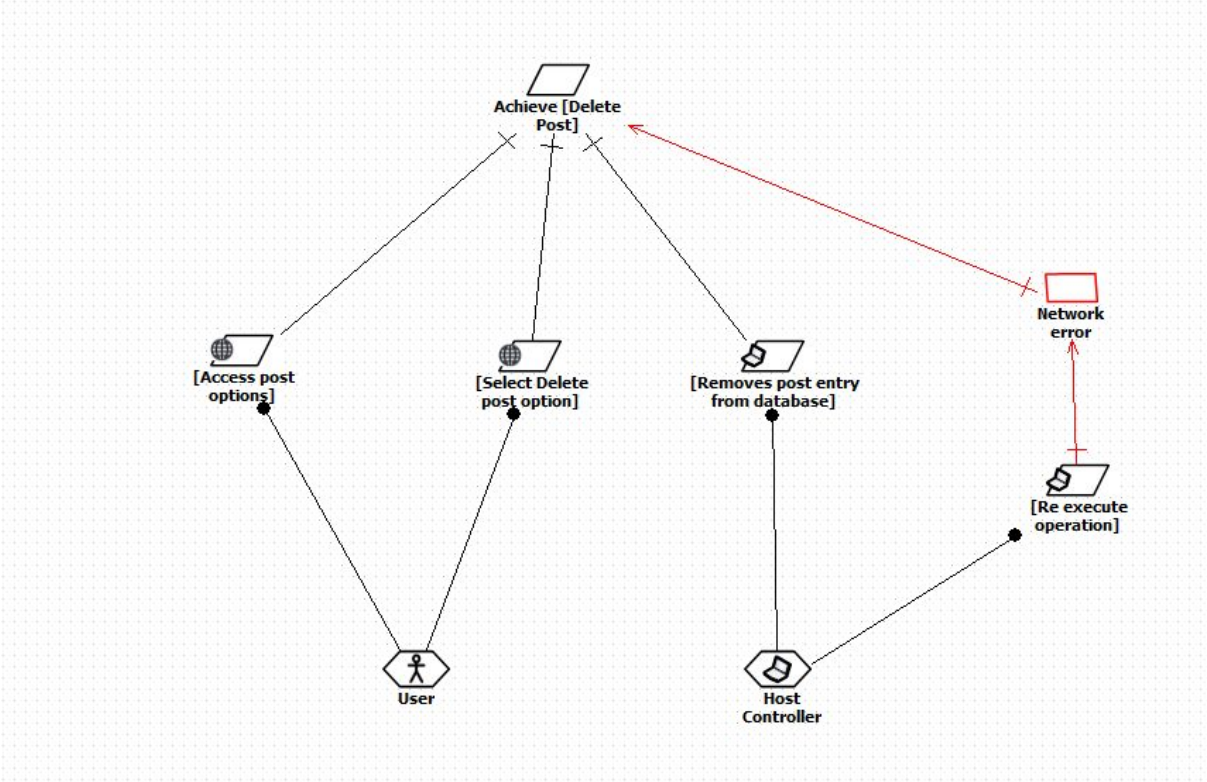
Share

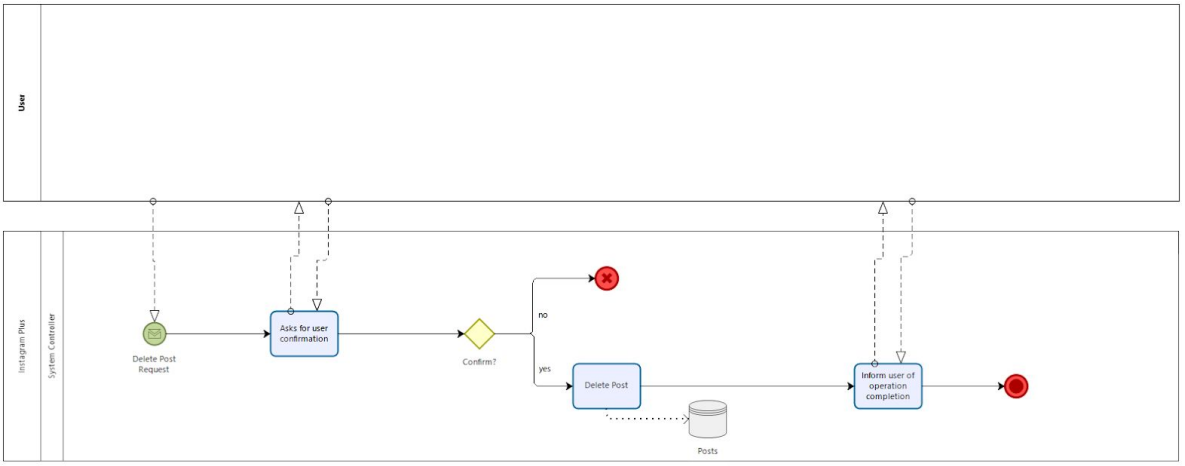

User
Instagram
System Controller
Select Post
Share ?
Yes
No
Select Users to share with
Share Post
Shared

Save

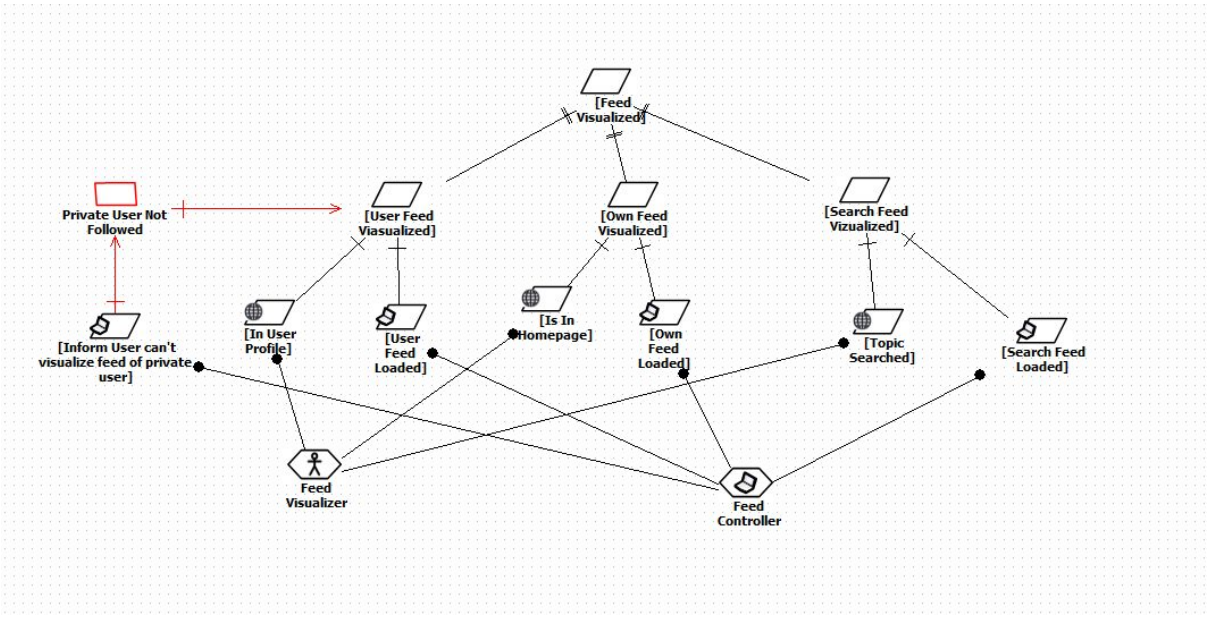

User
Instagram Plus
Saved Posts Cntroller
User and Saved Posts
Save Button Pressed
Post saved to user's saved post
Save button changed

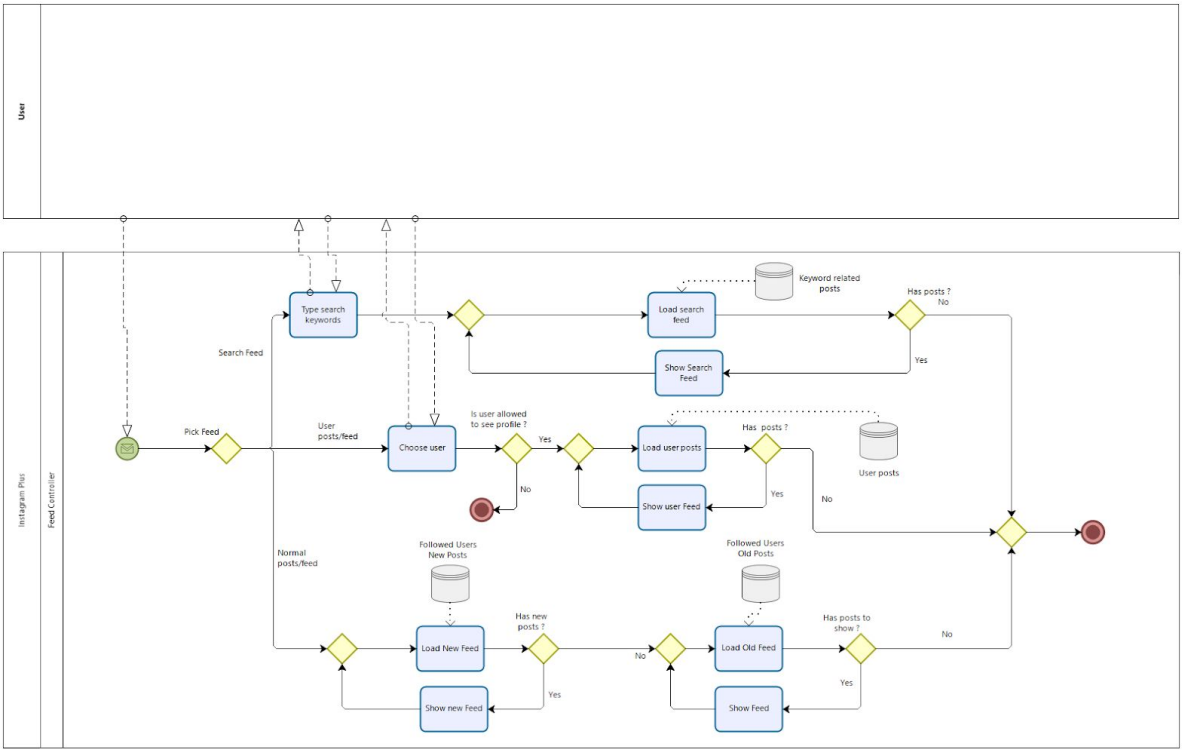
Delete Post


Achieve [Delete Post]
Network error
[Access post options]
[Select Delete post option]
[Removes post entry from database]
[Re execute operation]
User
Host Controller

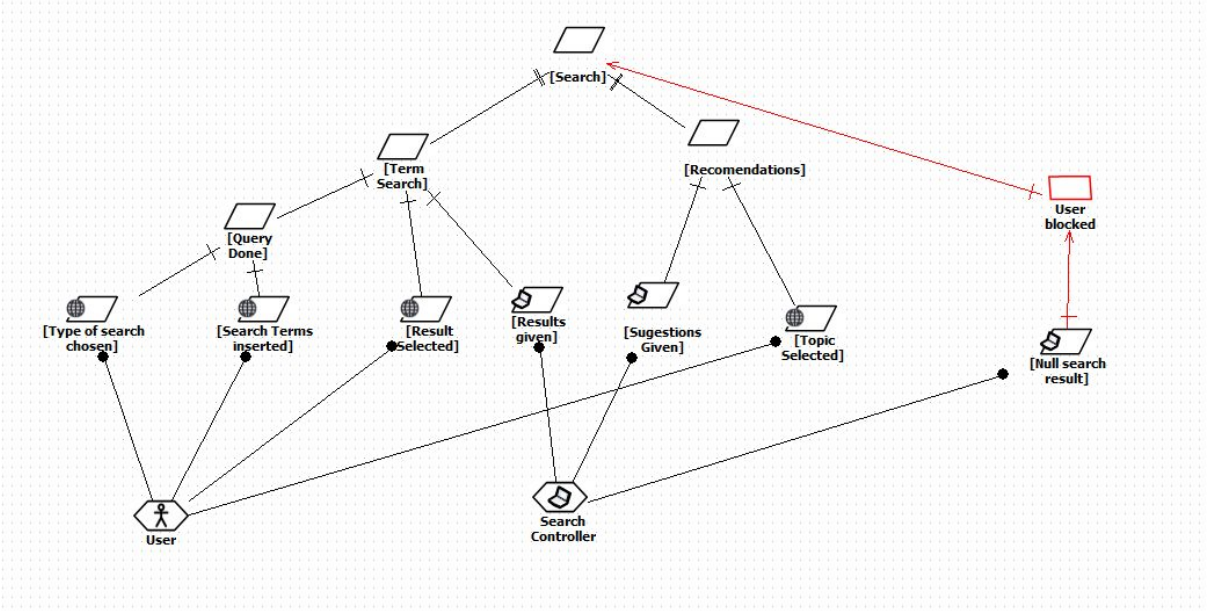

User
Instagram Plus
System Controller
Delete Post Request
Asks for user confirmation
Confirm?
no
yes
Delete Post
Posts
Inform user of operation completion

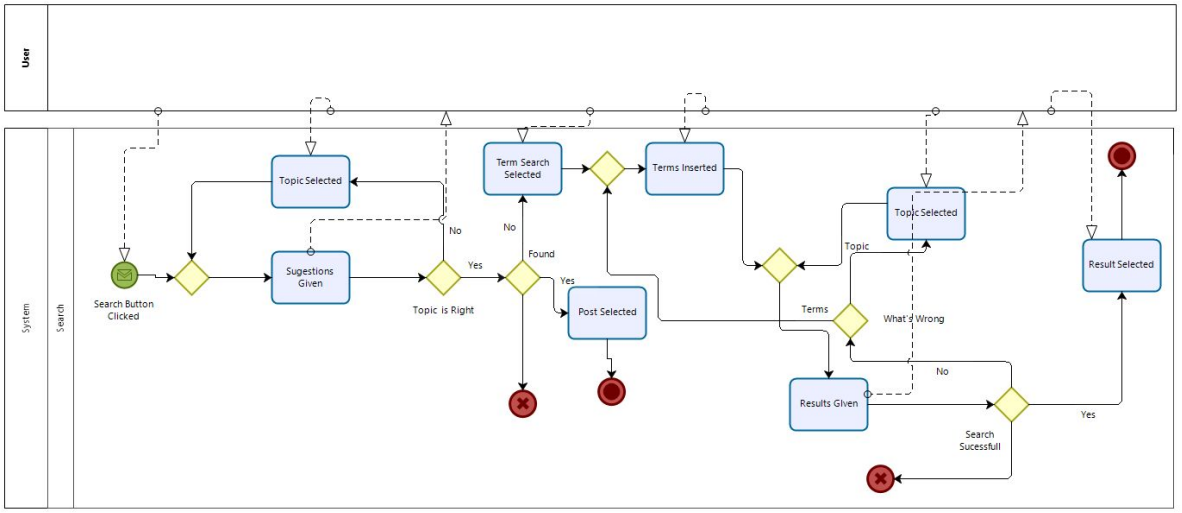
Watch Feed


[Feed Visualized]
Private User Not Followed
[User Feed Viasualized]
[Own Feed Visualized]
[Search Feed Vizualized]
[Inform User can't visualize feed of private user]
[In User Profile]
[User Feed Loaded]
[Is In Homepage]
[Own Feed Loaded]
[Topic Searched]
[Search Feed Loaded]
Feed Visualizer
Feed Controller

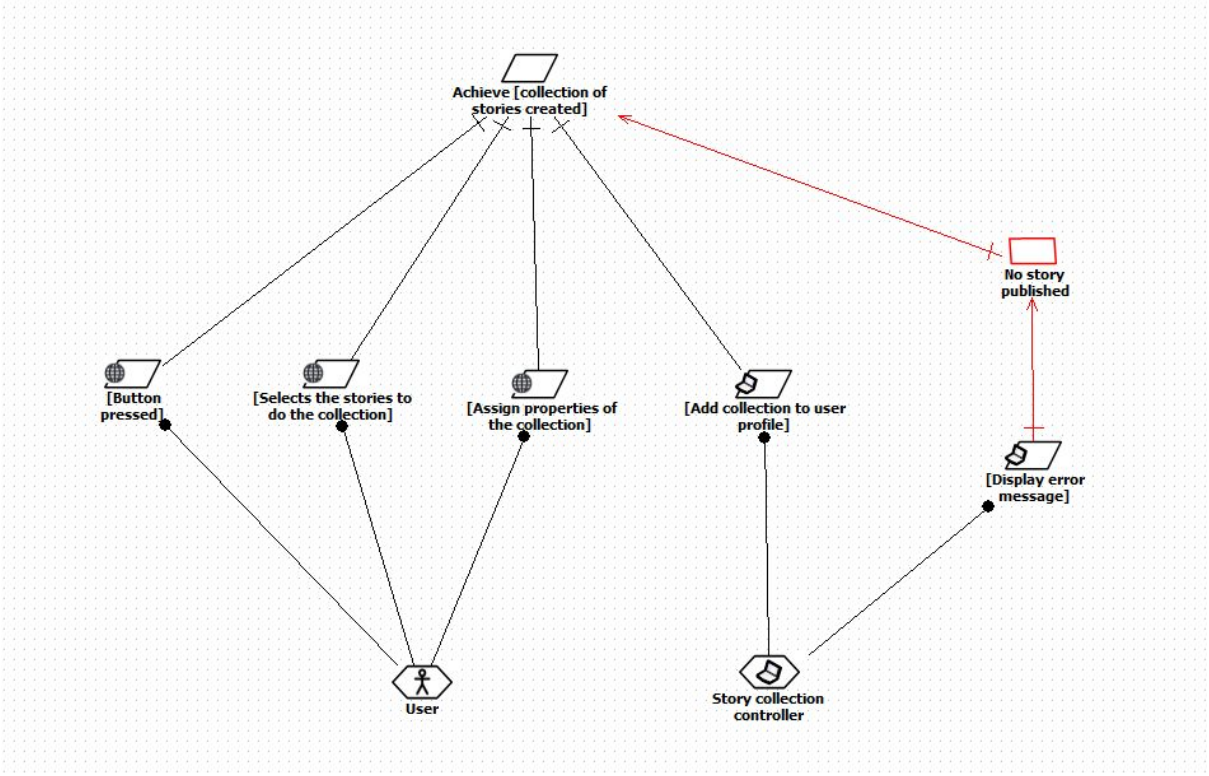

User
Instagram Plus
Feed Controller
Type search keywords
Search Feed
Keyword related posts
Load search feed
Has posts ?
No
Yes
Show Search Feed
Pick Feed
User posts/feed
Choose user
Is user allowed to see profile ?
Yes
No
Load user posts
Has posts ?
User posts
Yes
No
Show user Feed
Normal posts/feed
Followed Users New Posts
Followed Users Old Posts
Load New Feed
Has new posts ?
No
Yes
Show new Feed
Load Old Feed
Has posts to show ?
No
Yes
Show Feed

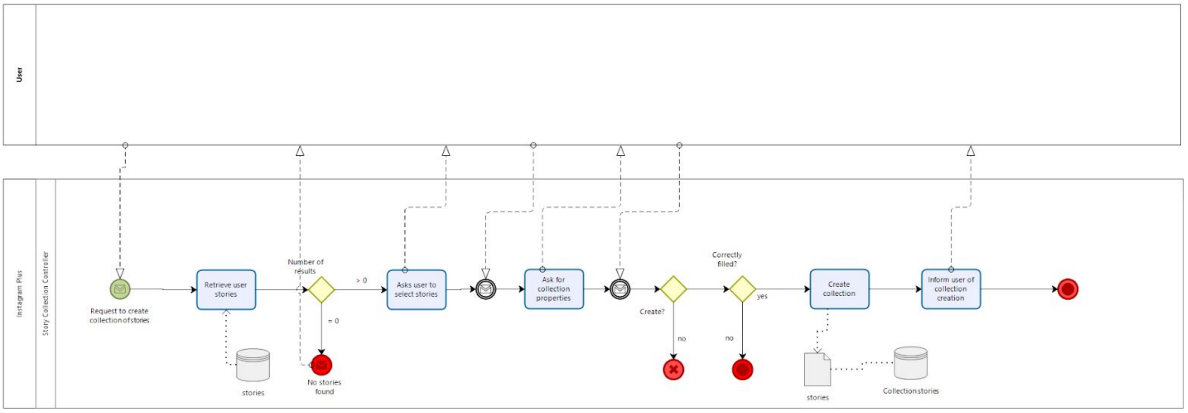
Search


[Search]
[Term Search]
[Recomendations]
[Query Done]
User blocked
[Type of search chosen]
[Search Terms inserted]
[Result Selected]
[Results given]
[Sugestions Given]
[Topic Selected]
[Null search result]
User
Search Controller

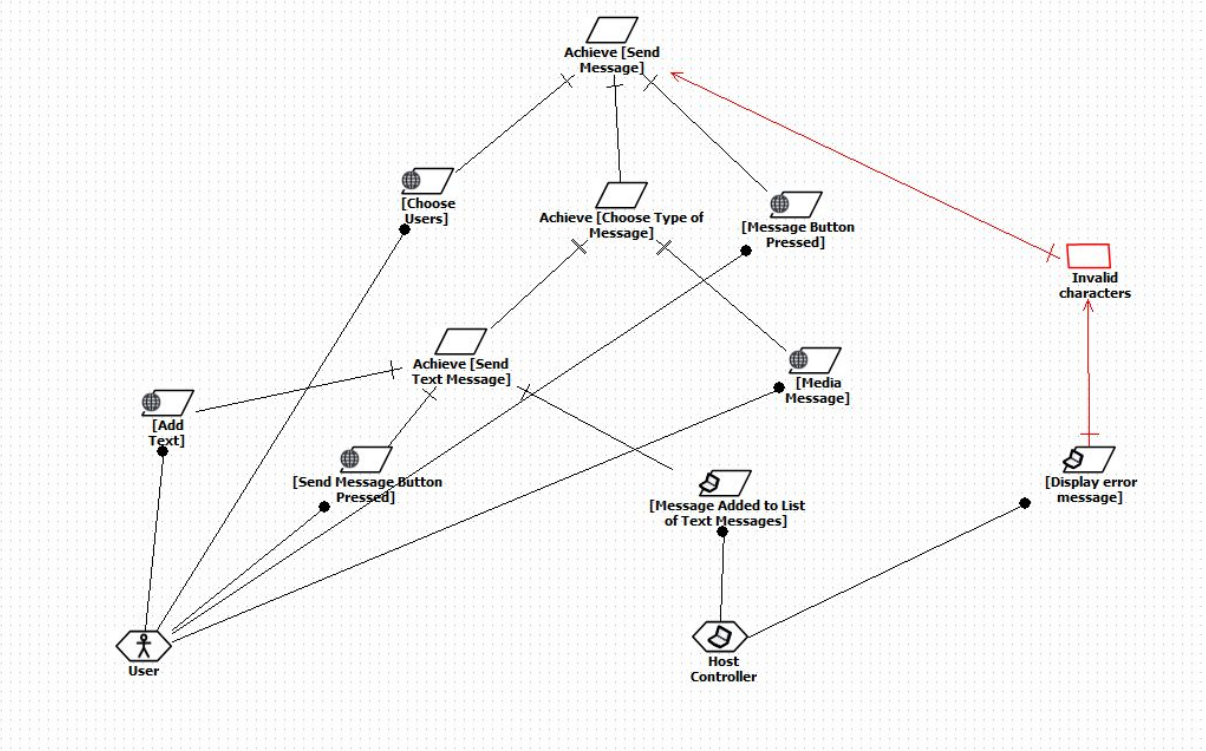

User
System
Search
Search Button Clicked
Topic Selected
Sugestions Given
Topic is Right
No
Yes
Term Search Selected
Found
No
Yes
Post Selected
Terms Inserted
Topic
Terms
Topic Selected
What's Wrong
No
Results Given
Yes
Search Sucessfull
Result Selected

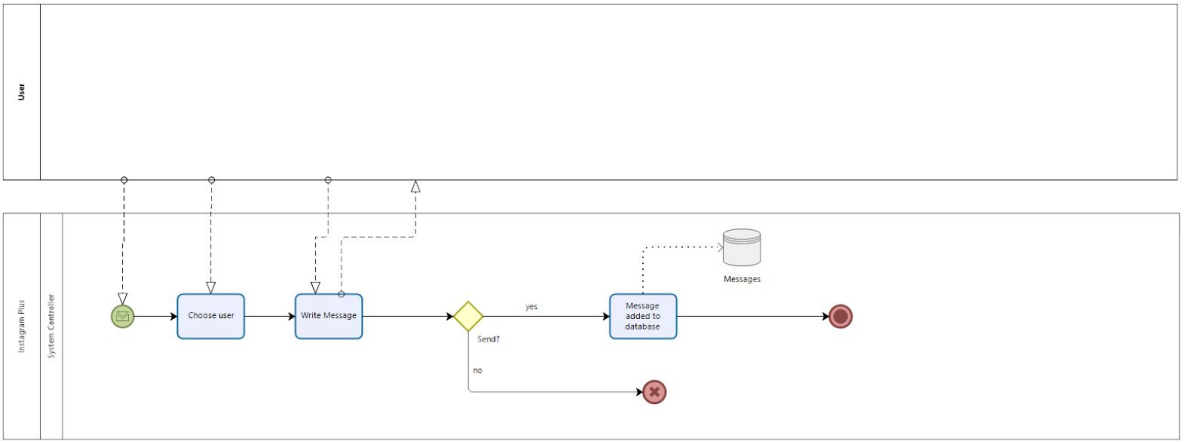
Create Collection of Stories



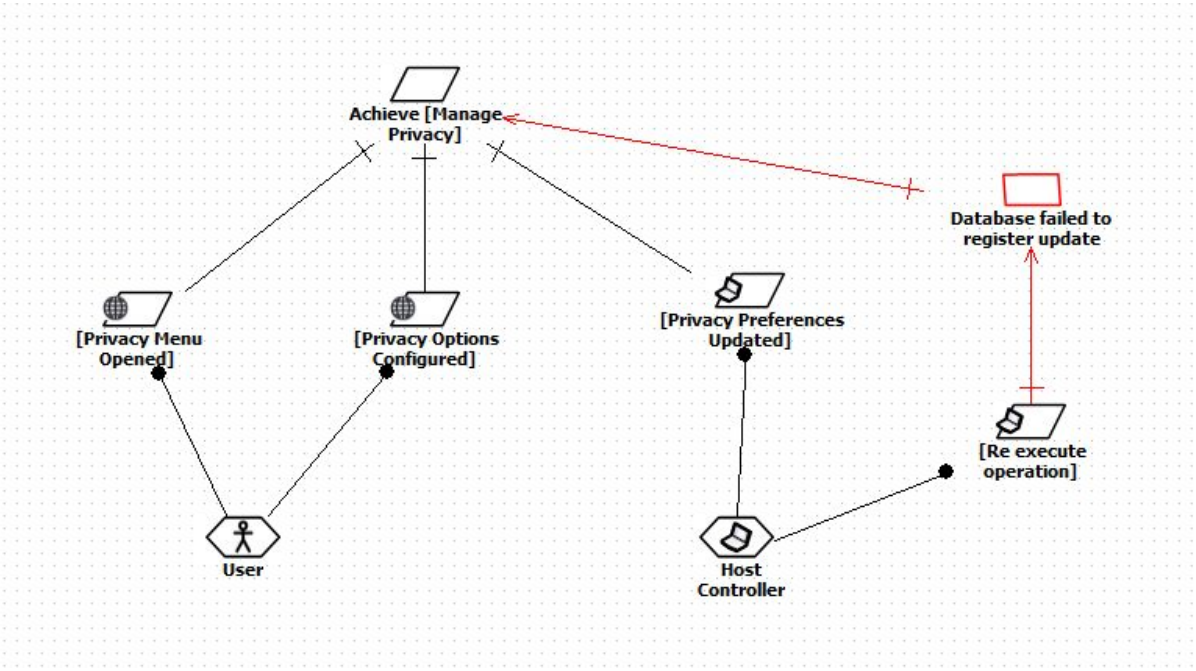


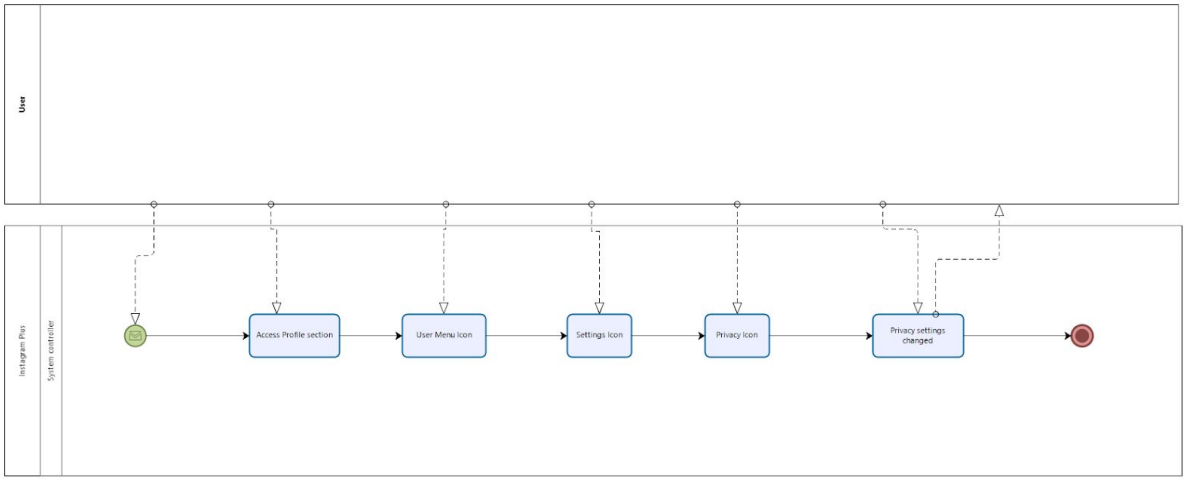
Send Message



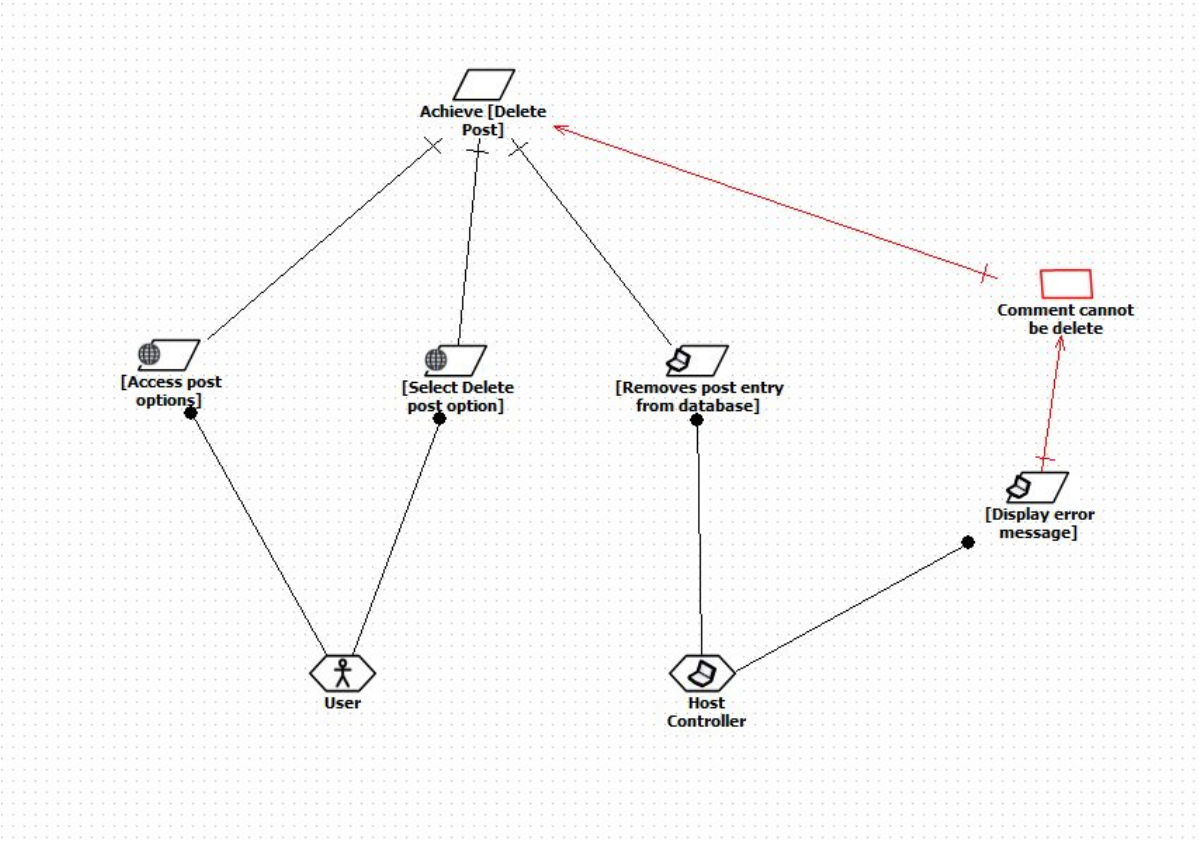


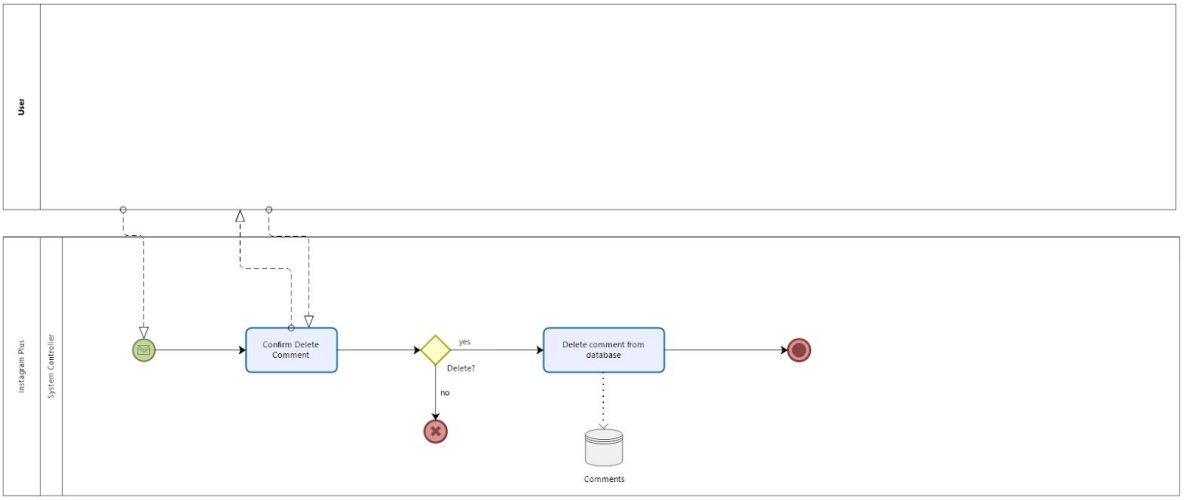
Manage Privacy



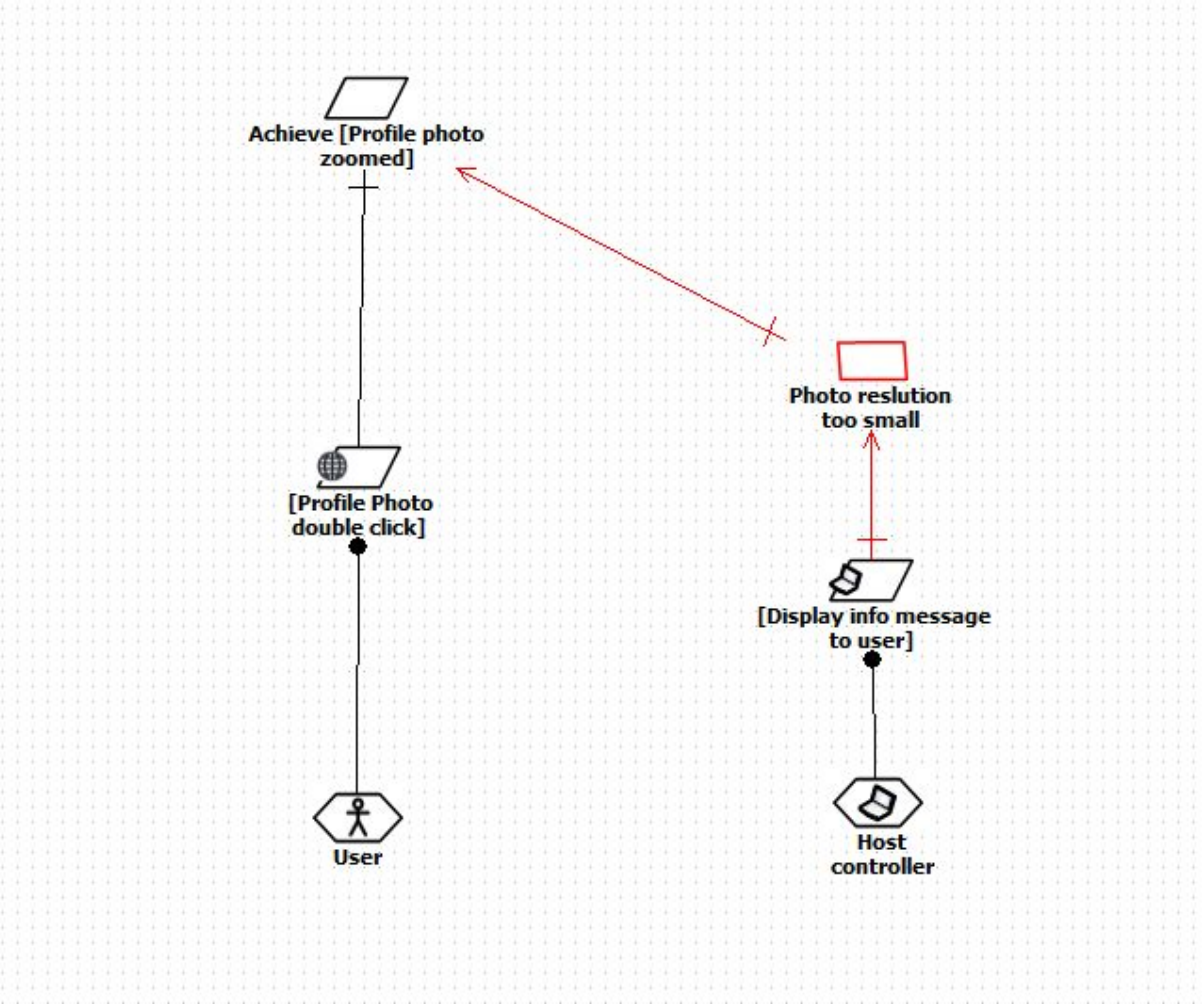


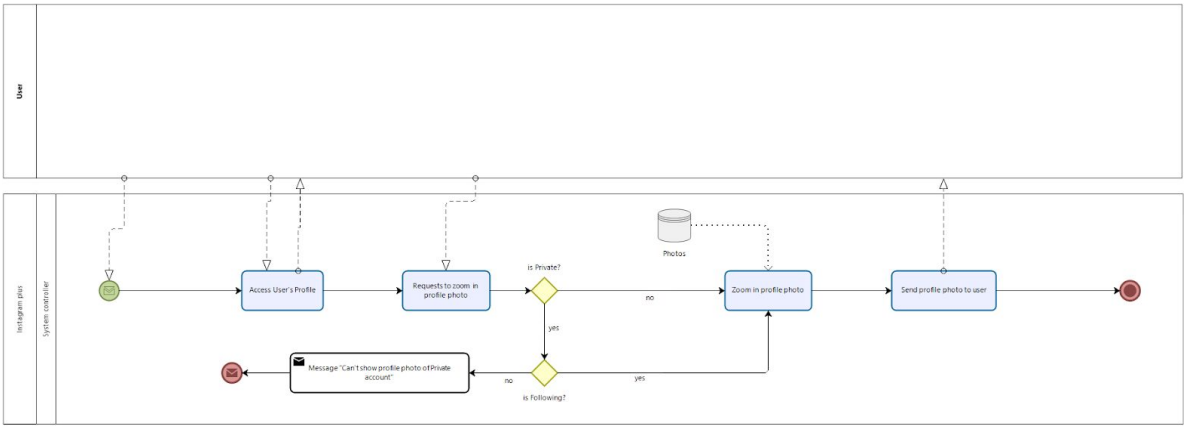
Delete Comment



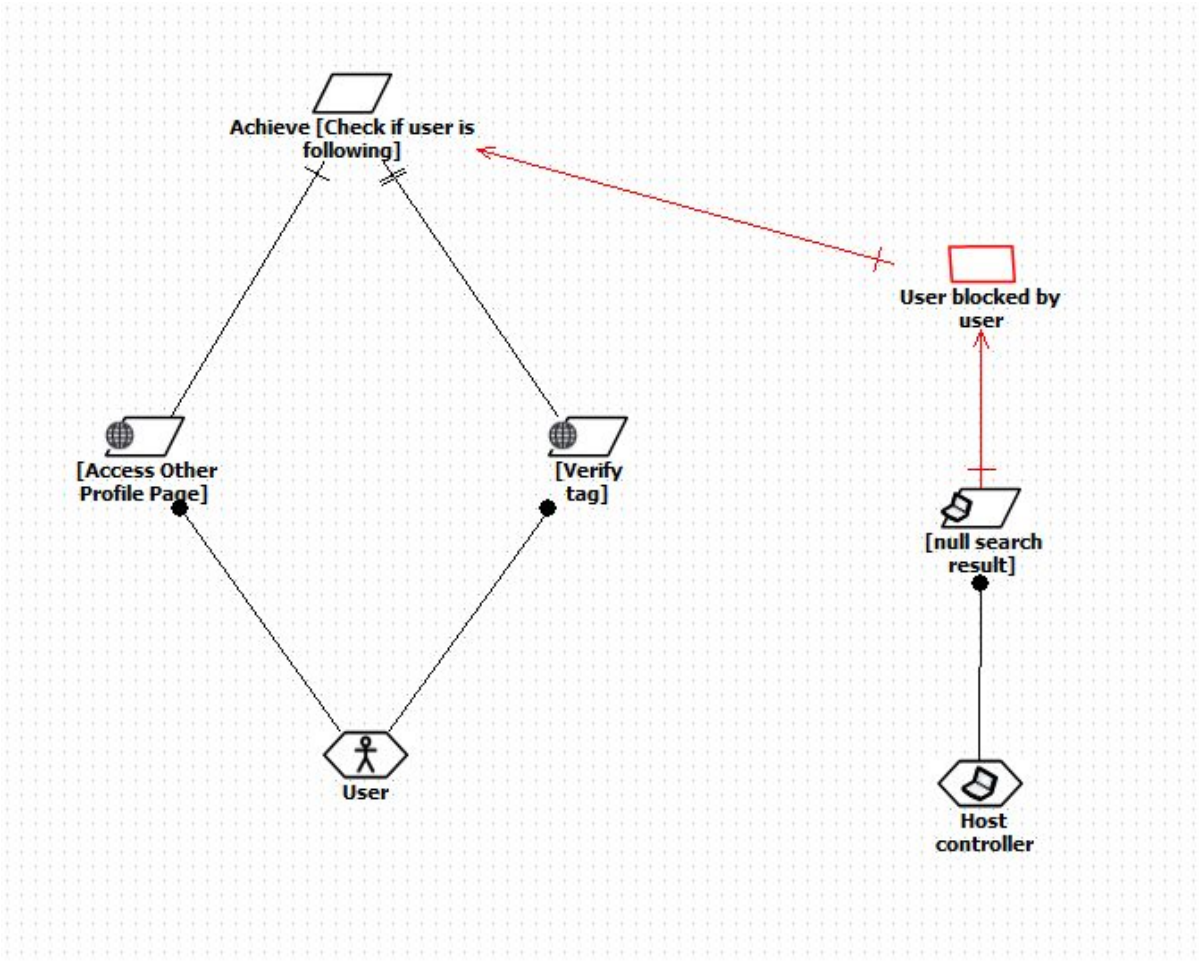


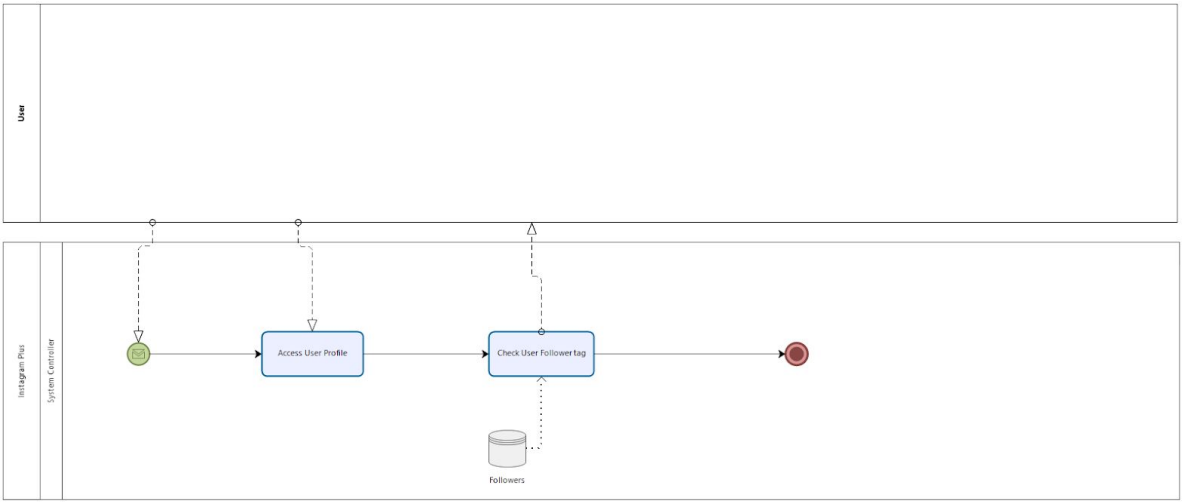
Zoom in User Photo



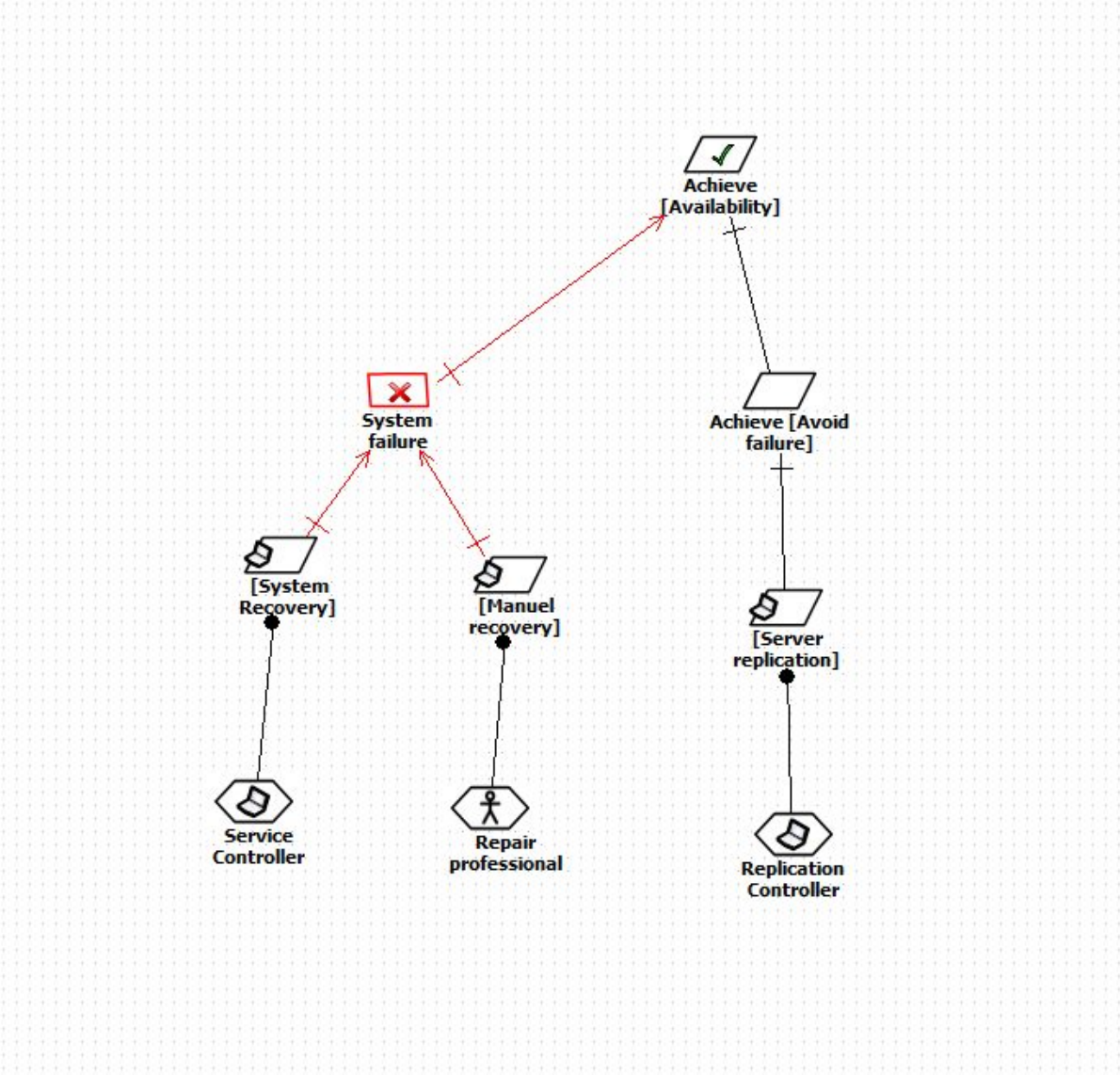


Check if User is Following

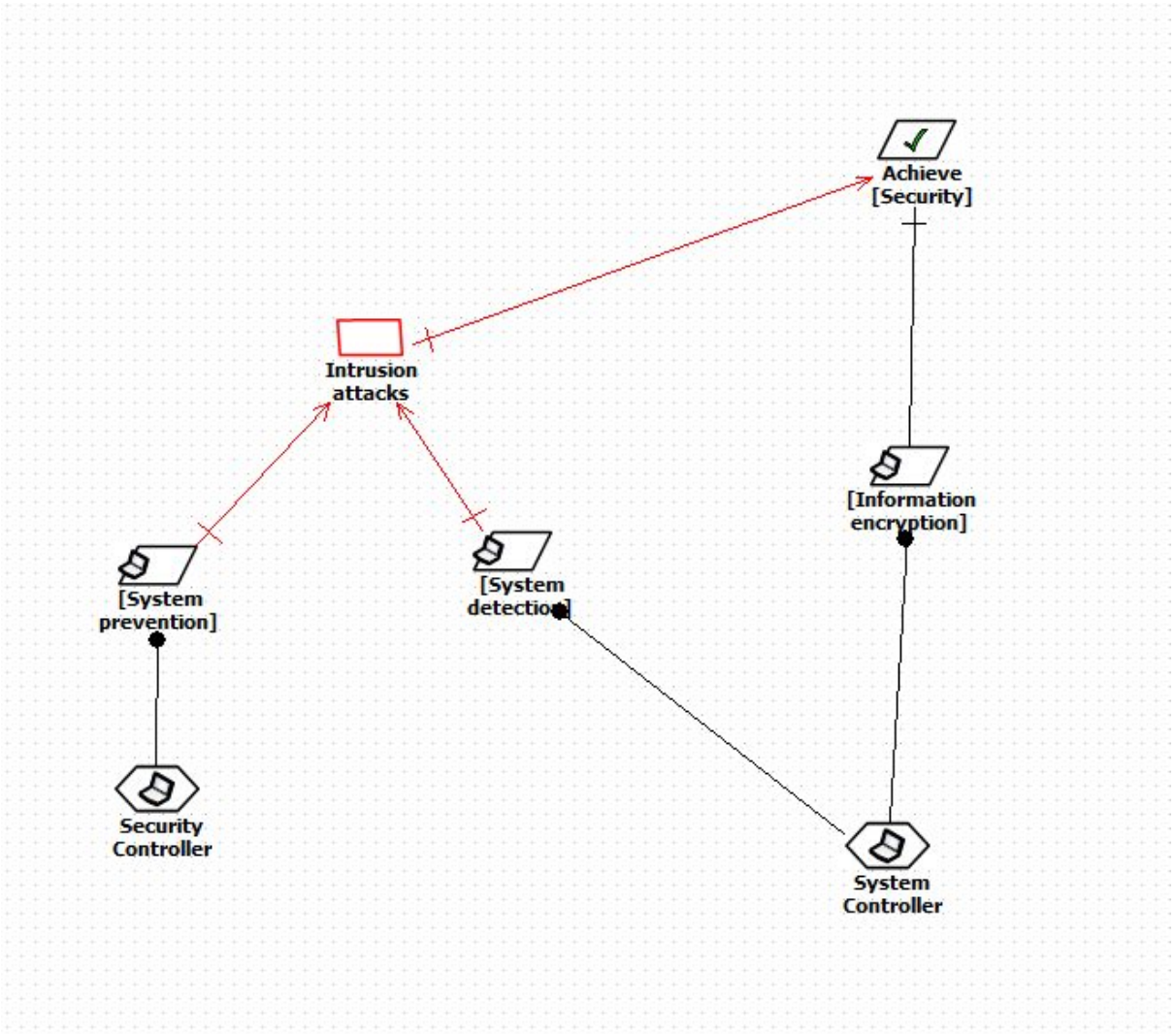




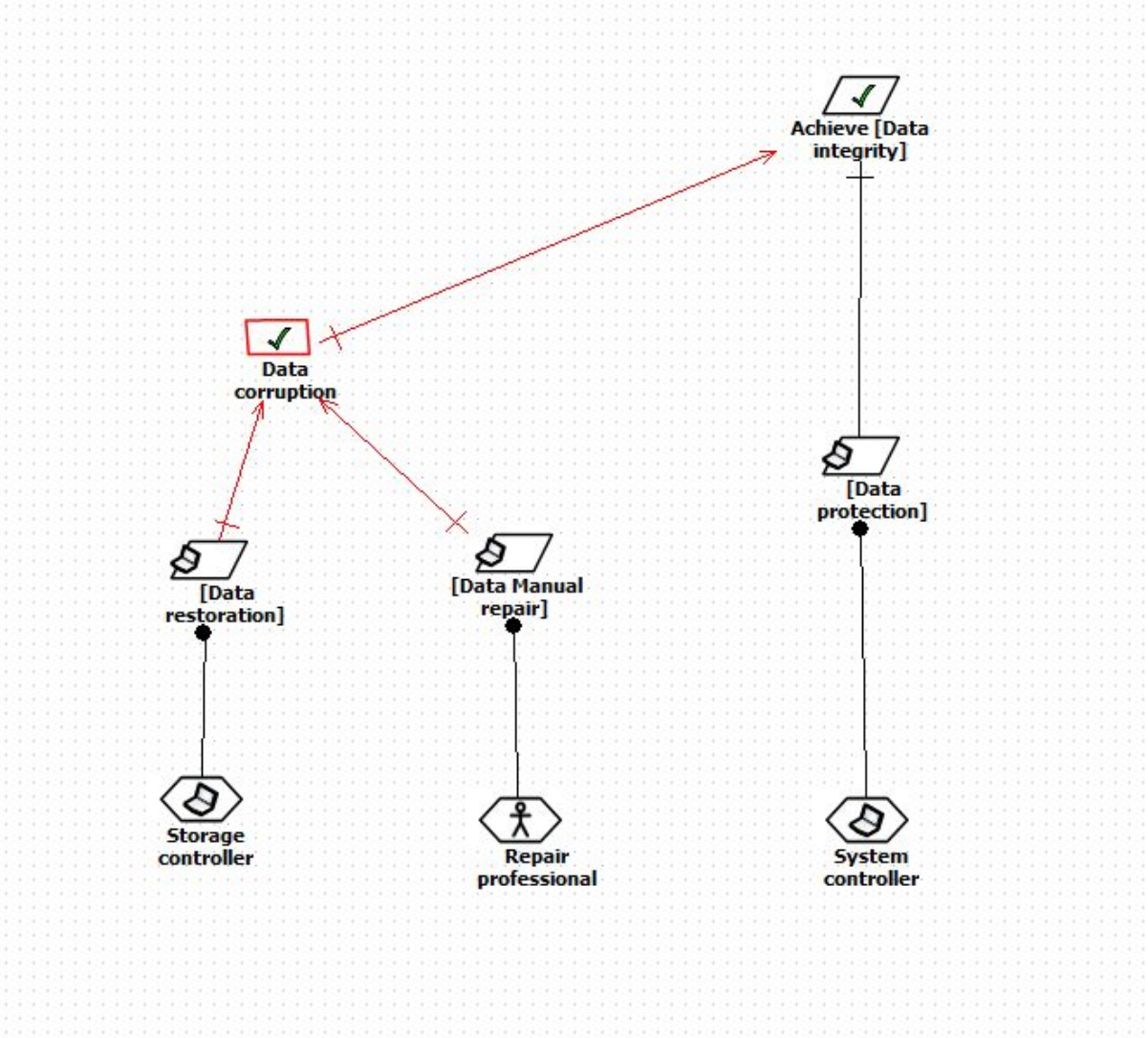
Availability



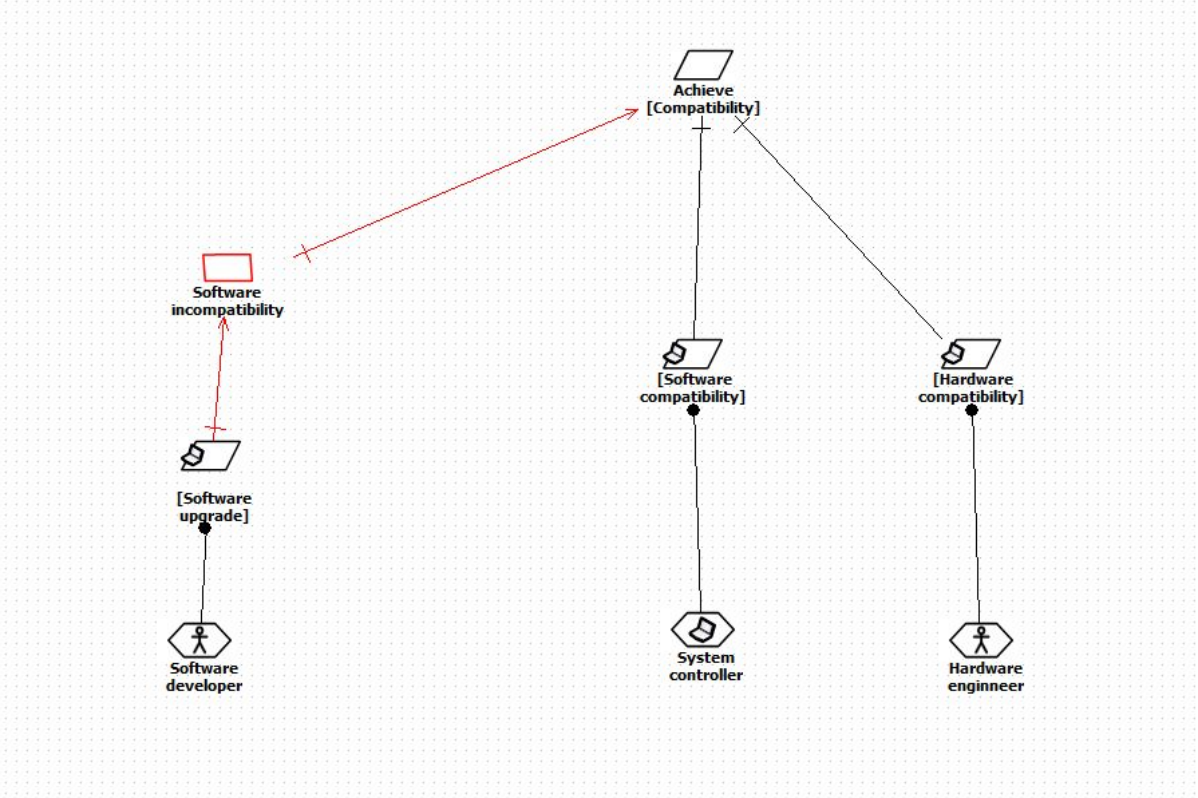
Security


Achieve
[Security]
Intrusion
attacks
[Information
encryption]
[System
prevention]
[System
detection]
Security
Controller
System
Controller

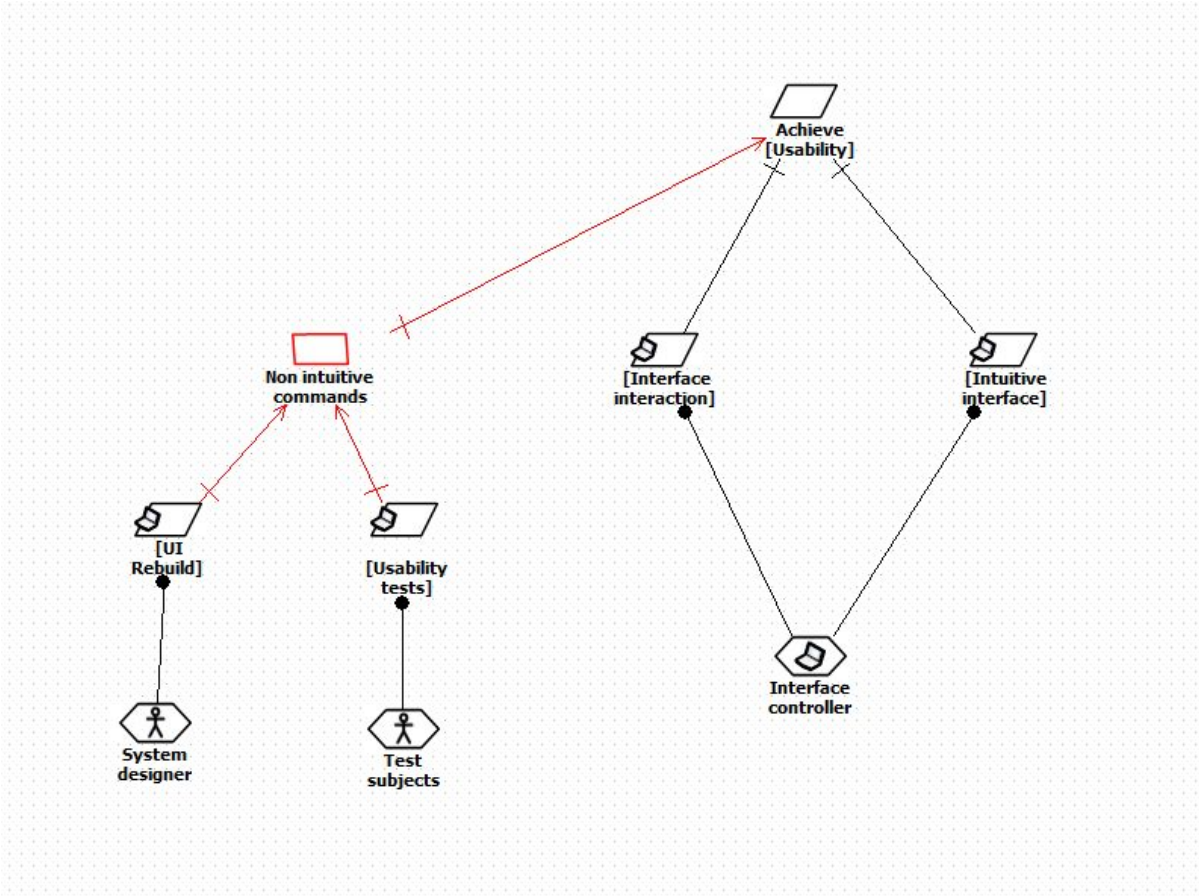
Data Integrity



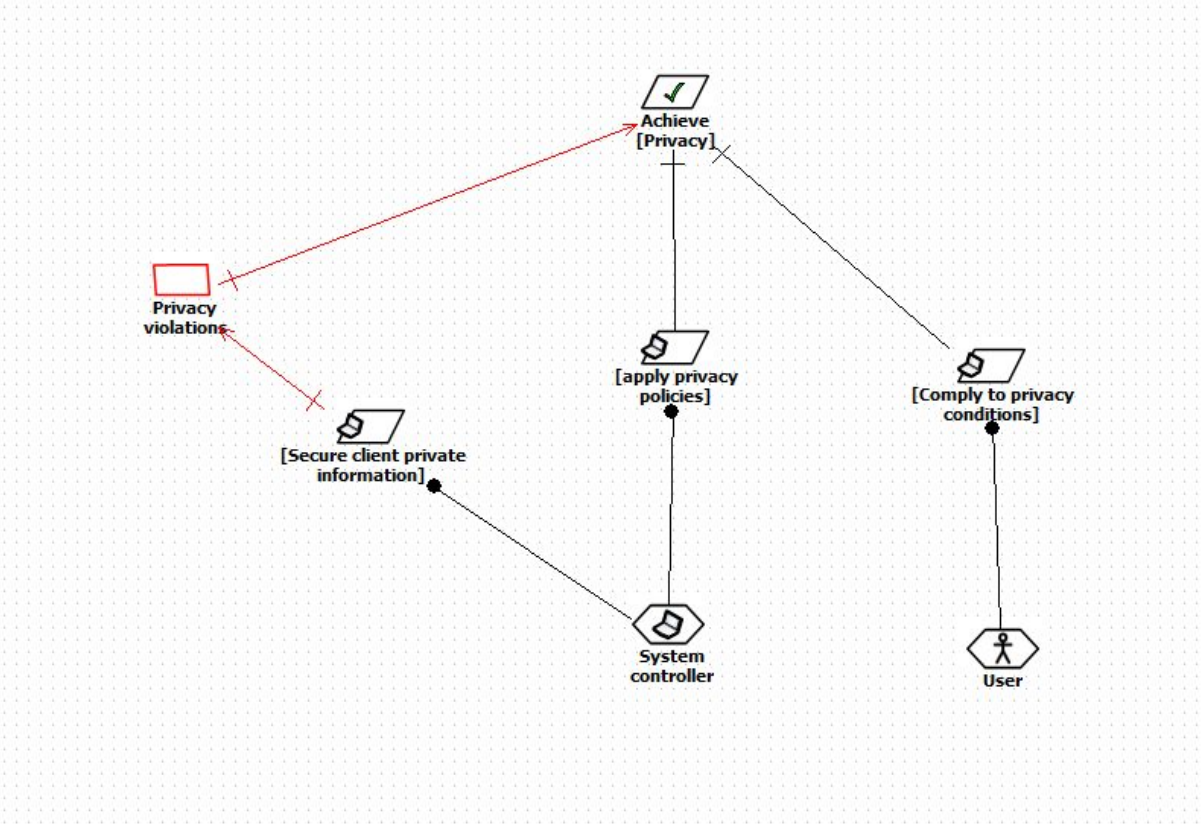
Compatibility



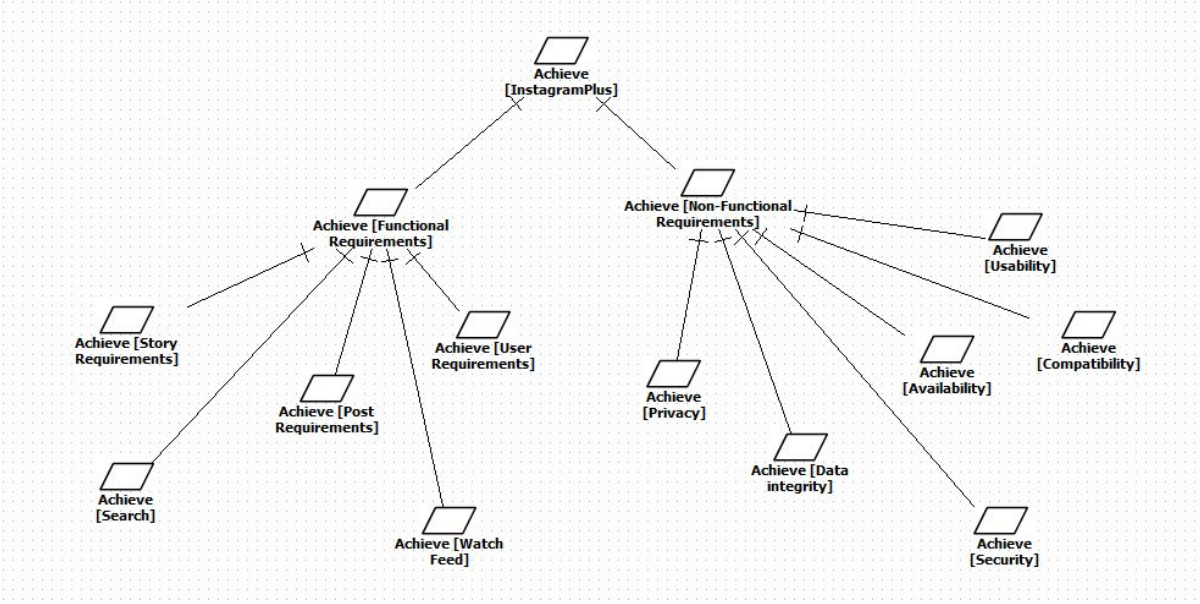
Usability

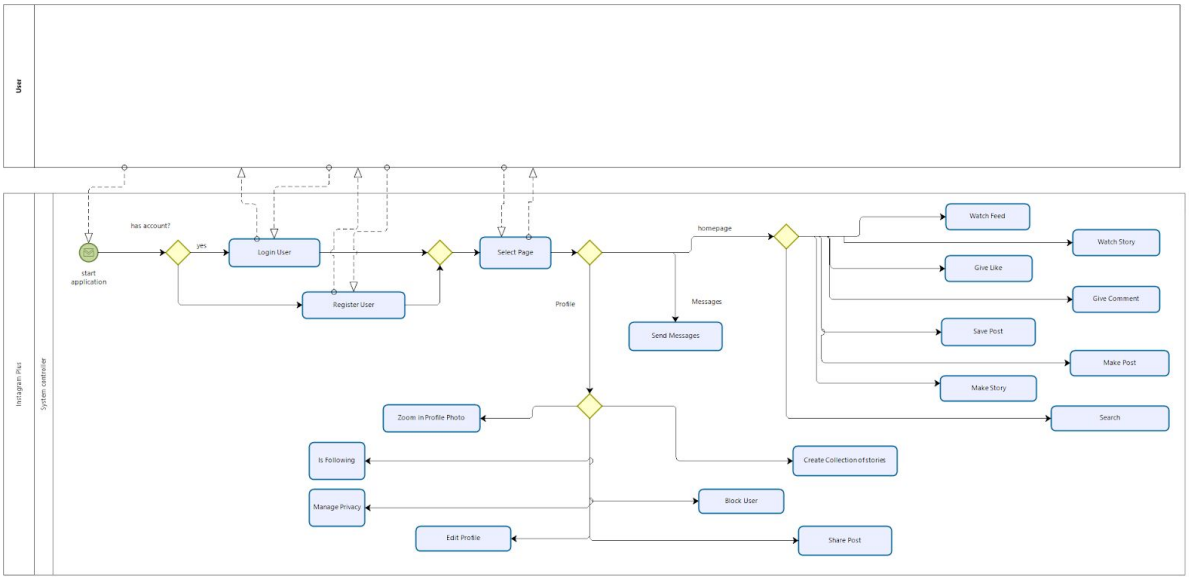

Achieve
[Usability]
Non intuitive
commands
[Interface
interaction]
[Intuitive
interface]
[UI
Rebuild]
[Usability
tests]
Interface
controller
System
designer
Test
subjects

Privacy


Achieve
[Privacy]
Privacy
violations
[apply privacy
policies]
[Comply to privacy
conditions]
[Secure client private
information]
System
controller
User

General


Achieve [InstagramPlus]
Achieve [Functional Requirements]
Achieve [Non-Functional Requirements]
Achieve [Story Requirements]
Achieve [User Requirements]
Achieve [Post Requirements]
Achieve [Search]
Achieve [Watch Feed]
Achieve [Usability]
Achieve [Compatibility]
Achieve [Availability]
Achieve [Privacy]
Achieve [Data integrity]
Achieve [Security]


User
Instagram Plus
System controller
has account?
yes
start application
Login User
Register User
Select Page
Profile
homepage
Messages
Send Messages
Watch Feed
Watch Story
Give Like
Give Comment
Save Post
Make Post
Make Story
Search
Zoom in Profile Photo
Is following
Manage Privacy
Edit Profile
Create Collection of stories
Block User
Share Post

Changes from Phase 1

1. The general BPMN was Added
2. The general KAOS model was added to the report (it was missing in the report but we talked about it during the discussion)
3. The Watch Feed BPMN was changed to list to address the different types of feed, displayed in the KAOS model
4. The BPMN models Create Collection of Stories and Make Story were changed to include possible errors in execution that were in their respective KAOS models.

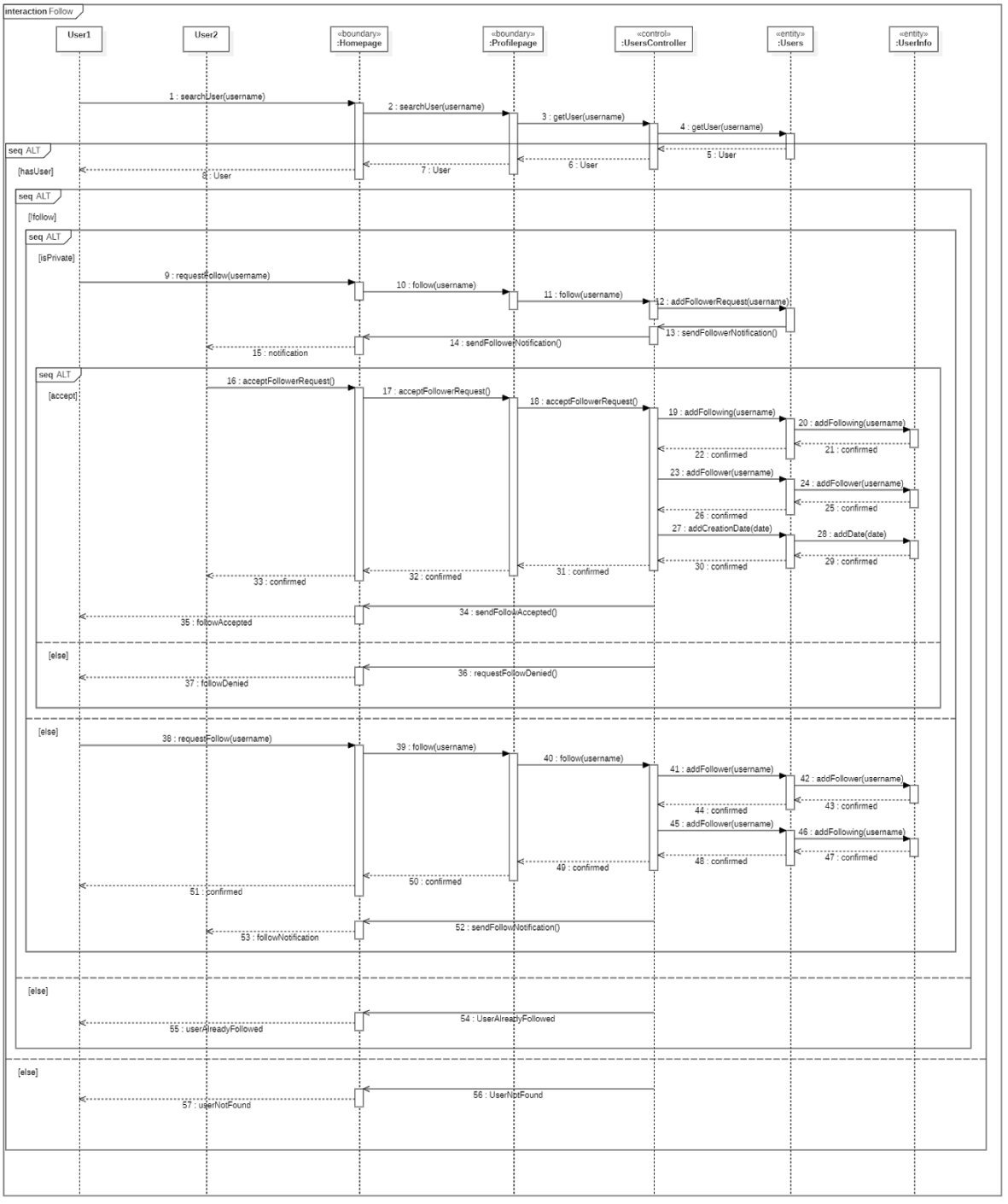
Chosen Features to Implement

For the next phase we will implement a prototype with a focus on the following features (approved by the professor):

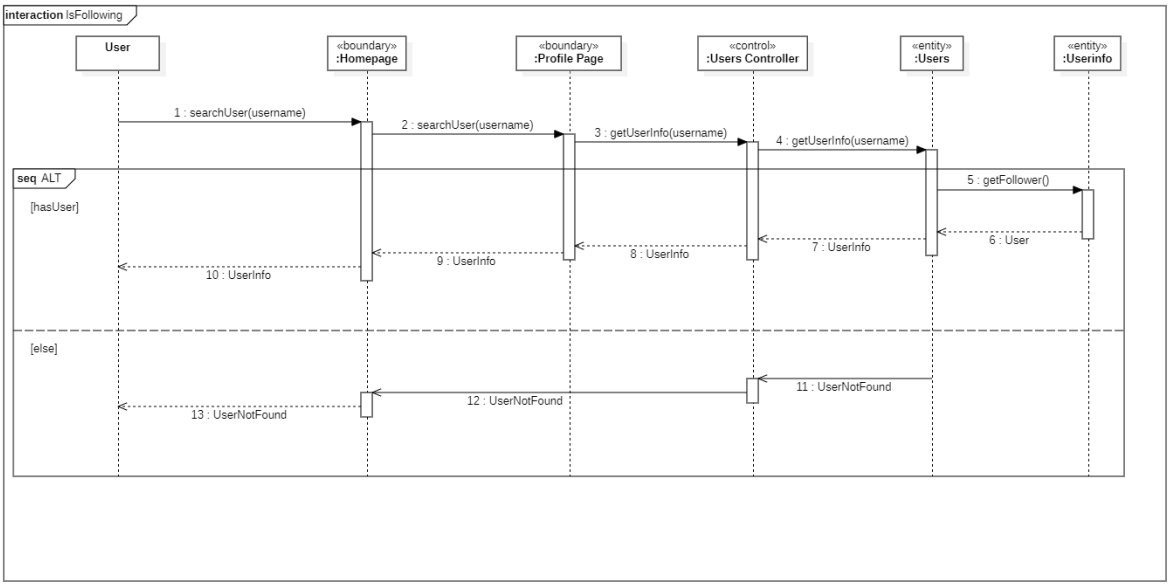
- Follow User
- Check if User is Following
- Send Message
- Make Post
- Delete Post
- Edit Profile

Sequence Diagrams

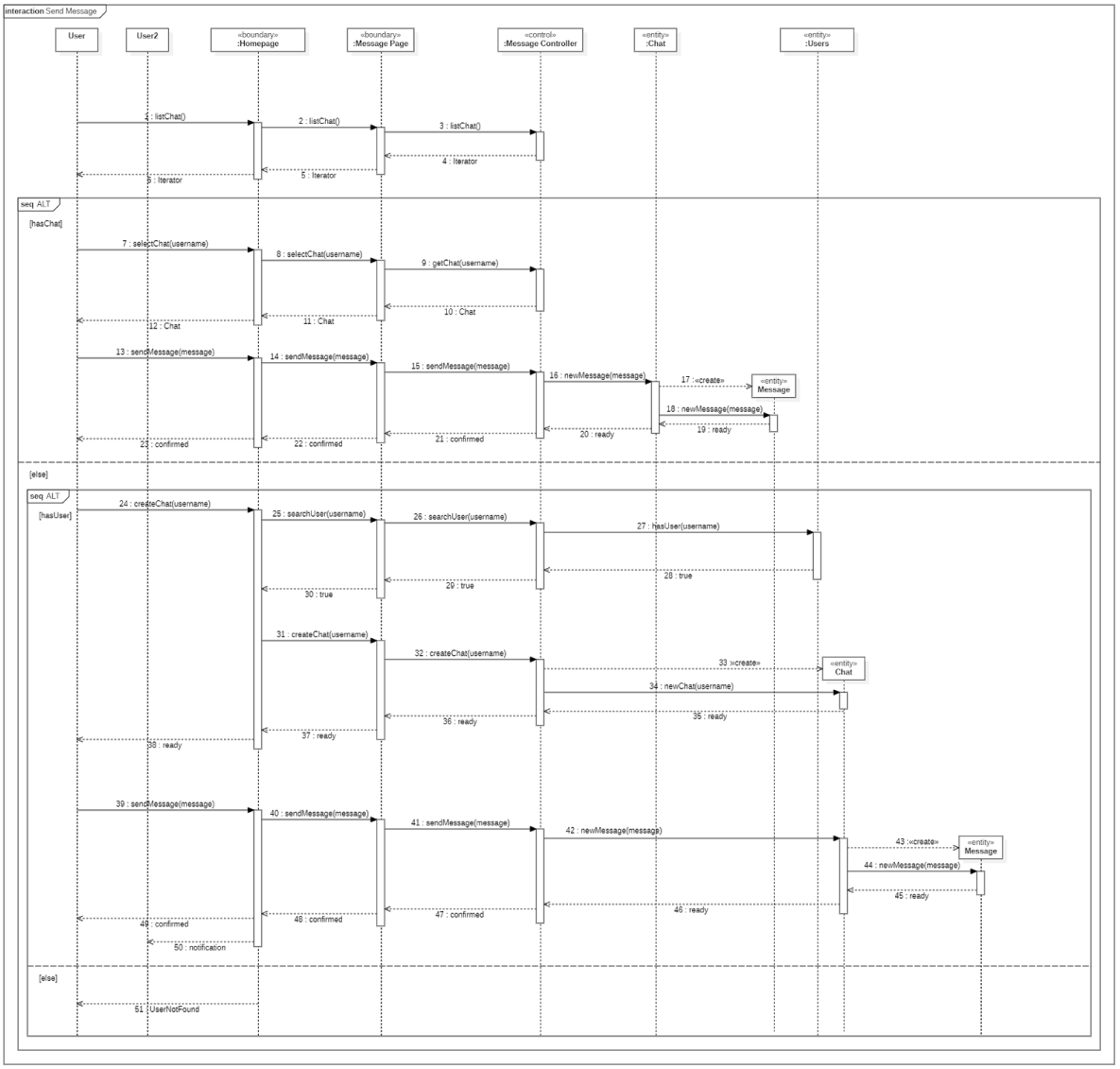
1. Follow User



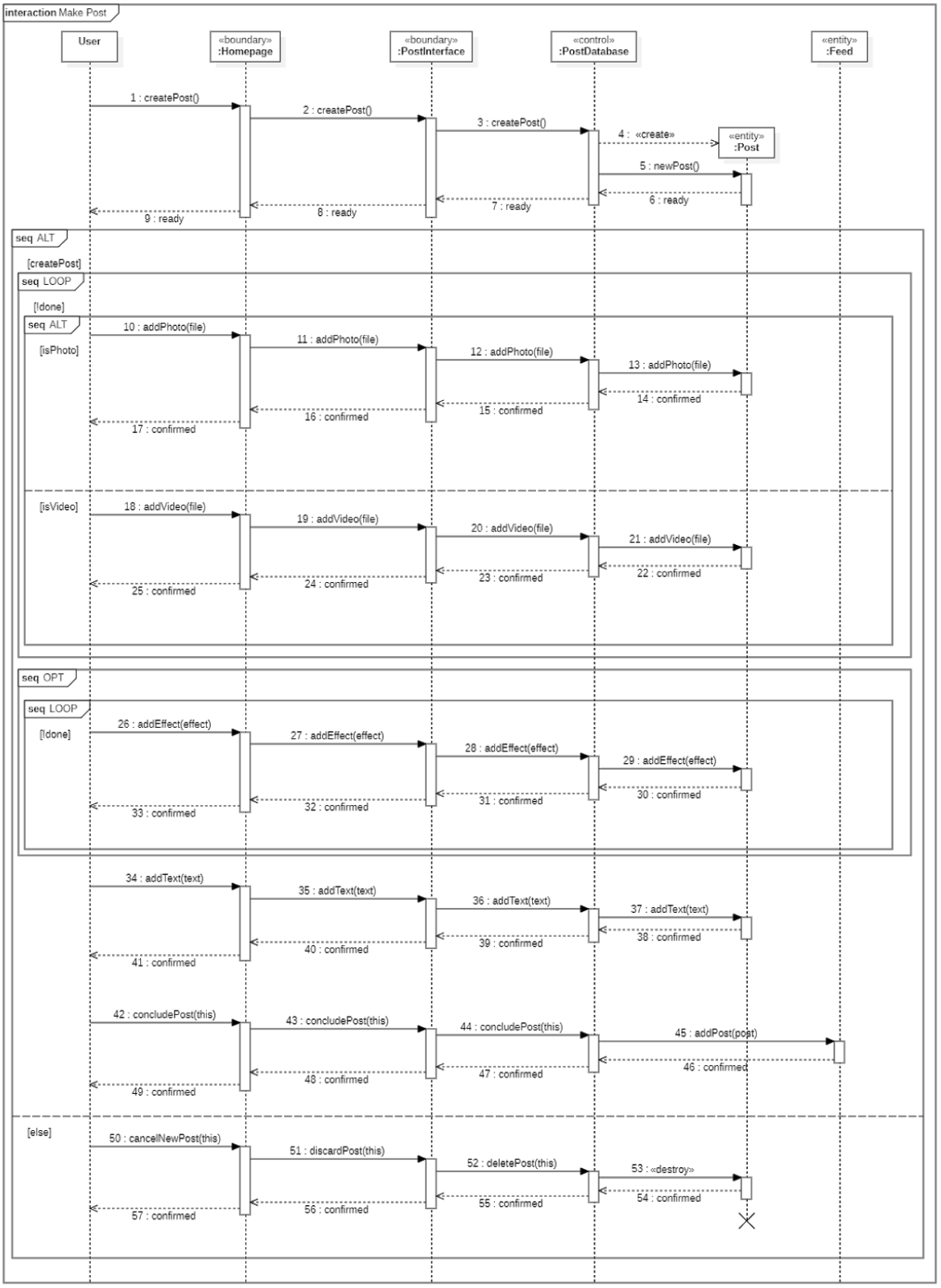
2. Check if User is Following



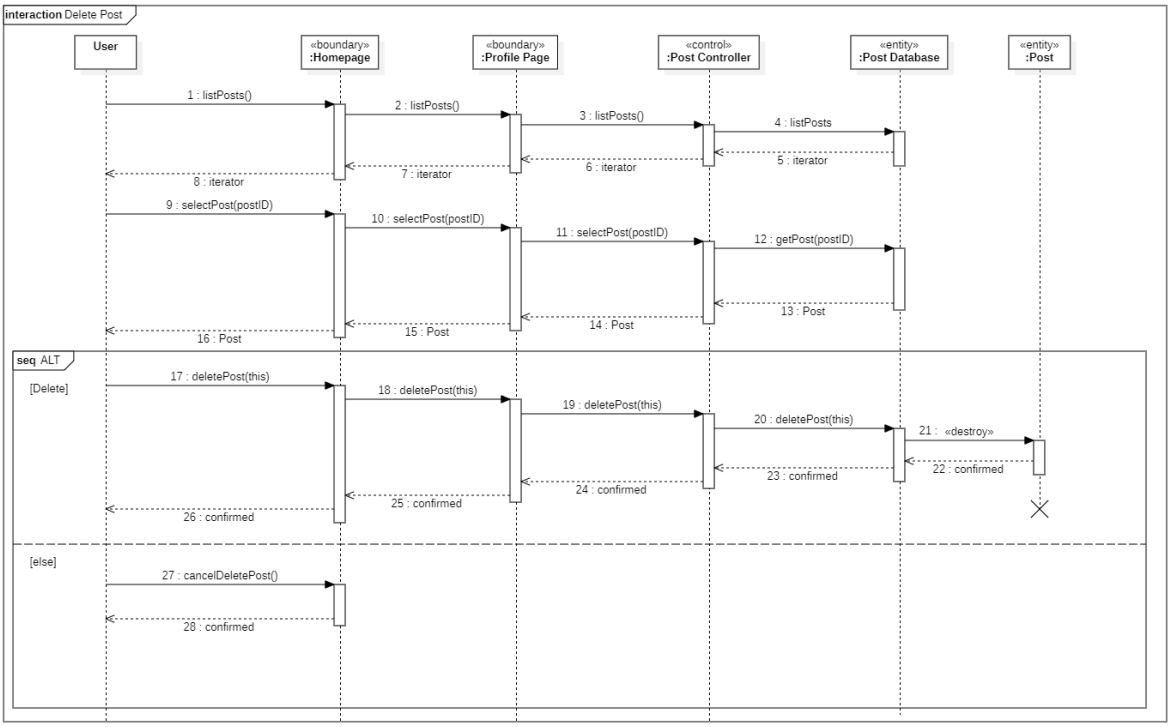
3. Send Message



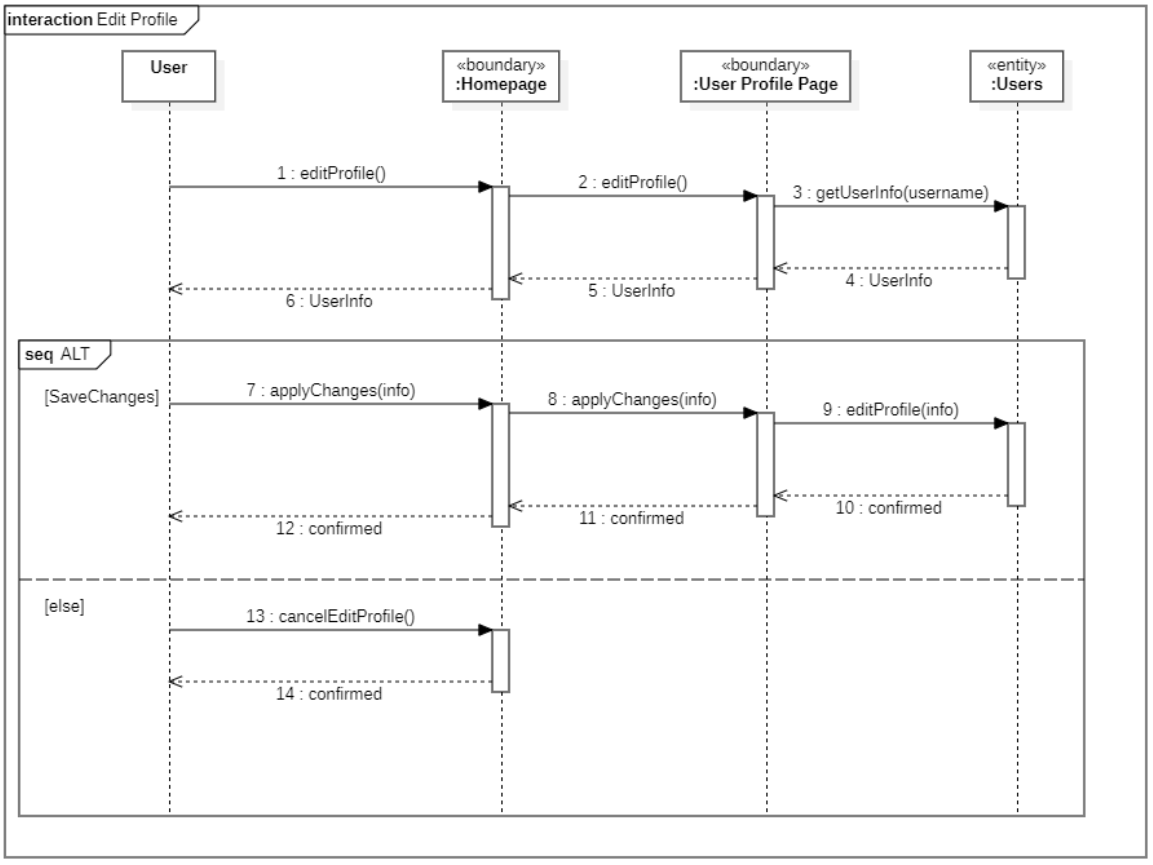
4. Make Post


interaction Make Post
User
«boundary» :Homepage
«boundary» :PostInterface
«control» :PostDatabase
«entity» :Feed
«entity» :Post
1 : createPost()
2 : createPost()
3 : createPost()
4 : «create»
5 : newPost()
6 : ready
7 : ready
8 : ready
9 : ready
seq ALT
[createPost]
seq LOOP
[!done]
seq ALT
[isPhoto]
10 : addPhoto(file)
11 : addPhoto(file)
12 : addPhoto(file)
13 : addPhoto(file)
14 : confirmed
15 : confirmed
16 : confirmed
17 : confirmed
[isVideo]
18 : addVideo(file)
19 : addVideo(file)
20 : addVideo(file)
21 : addVideo(file)
22 : confirmed
23 : confirmed
24 : confirmed
25 : confirmed
seq OPT
seq LOOP
[!done]
26 : addEffect(effect)
27 : addEffect(effect)
28 : addEffect(effect)
29 : addEffect(effect)
30 : confirmed
31 : confirmed
32 : confirmed
33 : confirmed
34 : addText(text)
35 : addText(text)
36 : addText(text)
37 : addText(text)
38 : confirmed
39 : confirmed
40 : confirmed
41 : confirmed
42 : concludePost(this)
43 : concludePost(this)
44 : concludePost(this)
45 : addPost(post)
46 : confirmed
47 : confirmed
48 : confirmed
49 : confirmed
[else]
50 : cancelNewPost(this)
51 : discardPost(this)
52 : deletePost(this)
53 : «destroy»
54 : confirmed
55 : confirmed
56 : confirmed
57 : confirmed

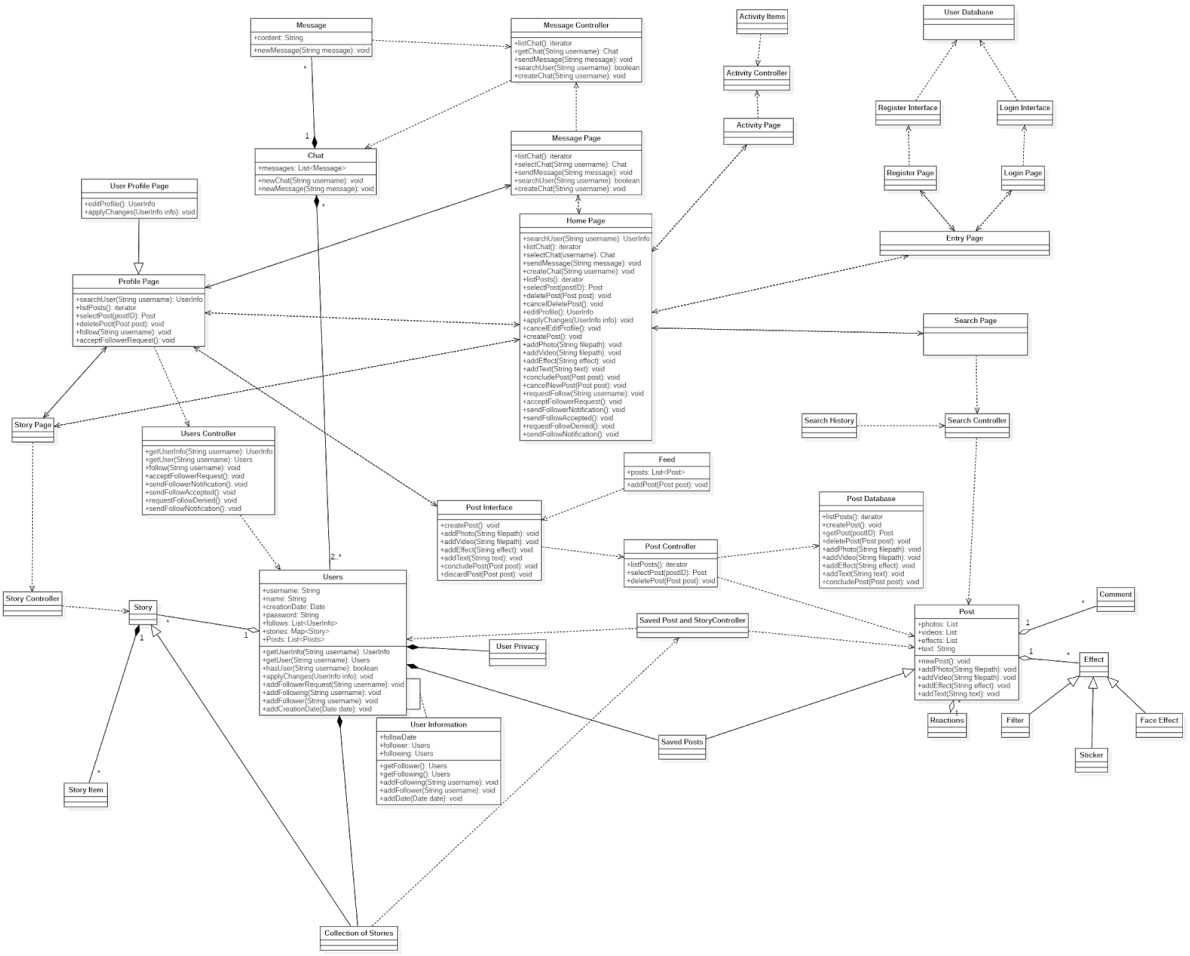
5. Delete Post



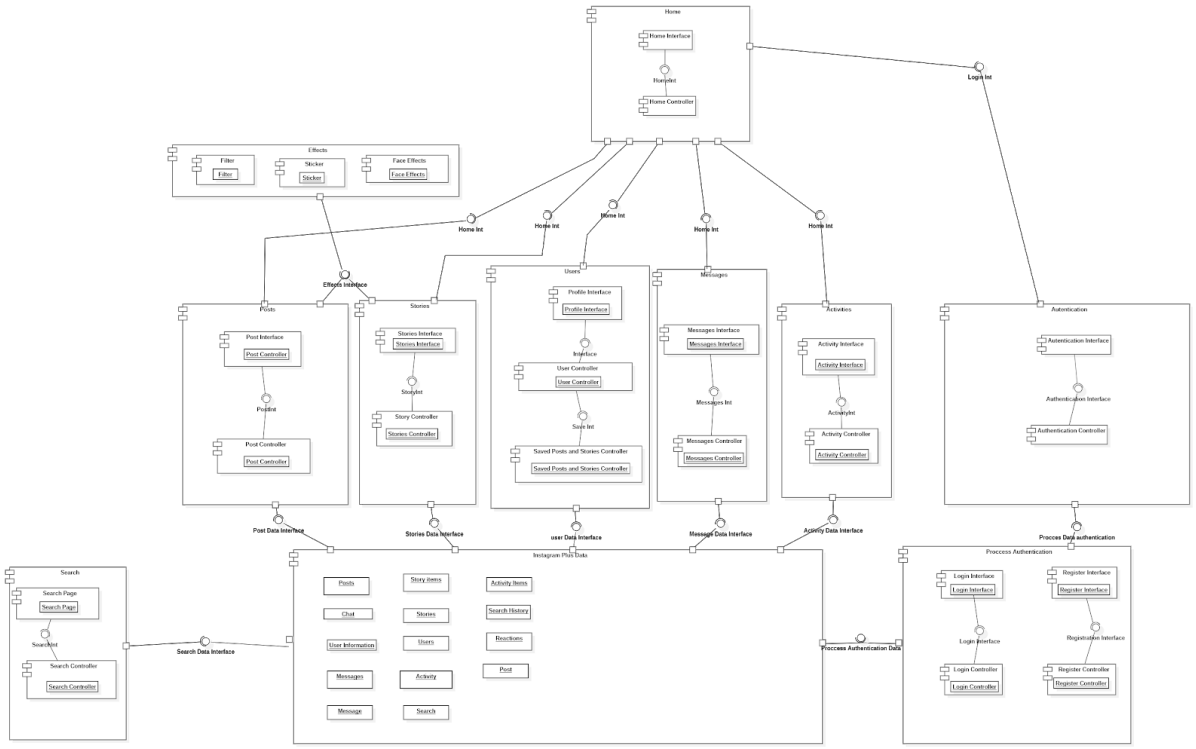
6. Edit Profile



Class Diagram



Component Diagram



The component diagram architecture used provides security, since the presented architecture protects from possible intrusions to the database from the login or register screens; privacy, because personal information, regarding each user is only accessible through login, and users can only consult other users' information which they have permission to see. Regarding the classes that each component implements, each component has the name corresponding to the the respective class. The list of methods each interface provides to the components that request them are displayed below.

Authentication Interface:

Process Data Interface:

Registration Interface:

Login Interface:

Process Authentication Data:

Home Interface:
searchUser(String username)
listChat()

selectChat(username)
sendMessage(String message)
createChat(String username)
listPosts()
selectPost(postID)
deletePost(Post post)
cancelDeletePost()
editProfile()
applyChanges(UserInfo info)
cancelEditProfile()
createPost()
addPhoto(String filepath)
addVideo(String filepath)
addEffect(String effect)
addText(String text)
concludePost(Post post)
cancelNewPost(Post post)
requestFollow(String username)
acceptFollowerRequest()

Profile Interface:
editProfile()
applyChanges(UserInfo info)
listPosts()
selectPost(postID)
deletePost(Post post)
follow(String username)
acceptFollowerRequest()
cancelEditProfile()

Save Interface:

User Data Interface:
getUserInfo(String username)
getUser(String username)
follow(String username)
hasUser(String username)
applyChanges(UserInfo info)
addFollowerRequest(String username)
addFollowing(String username)
addFollower(String username)
getFollower(String username)
cancelEditProfile()

Message Interface:
selectChat(String username)

sendMessage(String message)
searchUser(String username)
createChat(String username)

Message Data Interface:
listChat()
selectChat(String username)
newMessage(String message)
searchUser(String username)
createChat(String username)

Post Interface:
addPhoto(String filepath)
addVideo(String filepath)
addEffect(String effect)
addText(String text)
concludePost(Post post)
discardPost(Post post)

Post Data Interface:
createPost()
addPhoto(String filepath)
addVideo(String filepath)
addEffect(String effect)
addText(String text)
concludePost(Post post)
discardPost(Post post)
listPosts()
getPost(postID)
deletePost(Post post)

Story Interface:

Stories Data Interface:

Effects Interface:

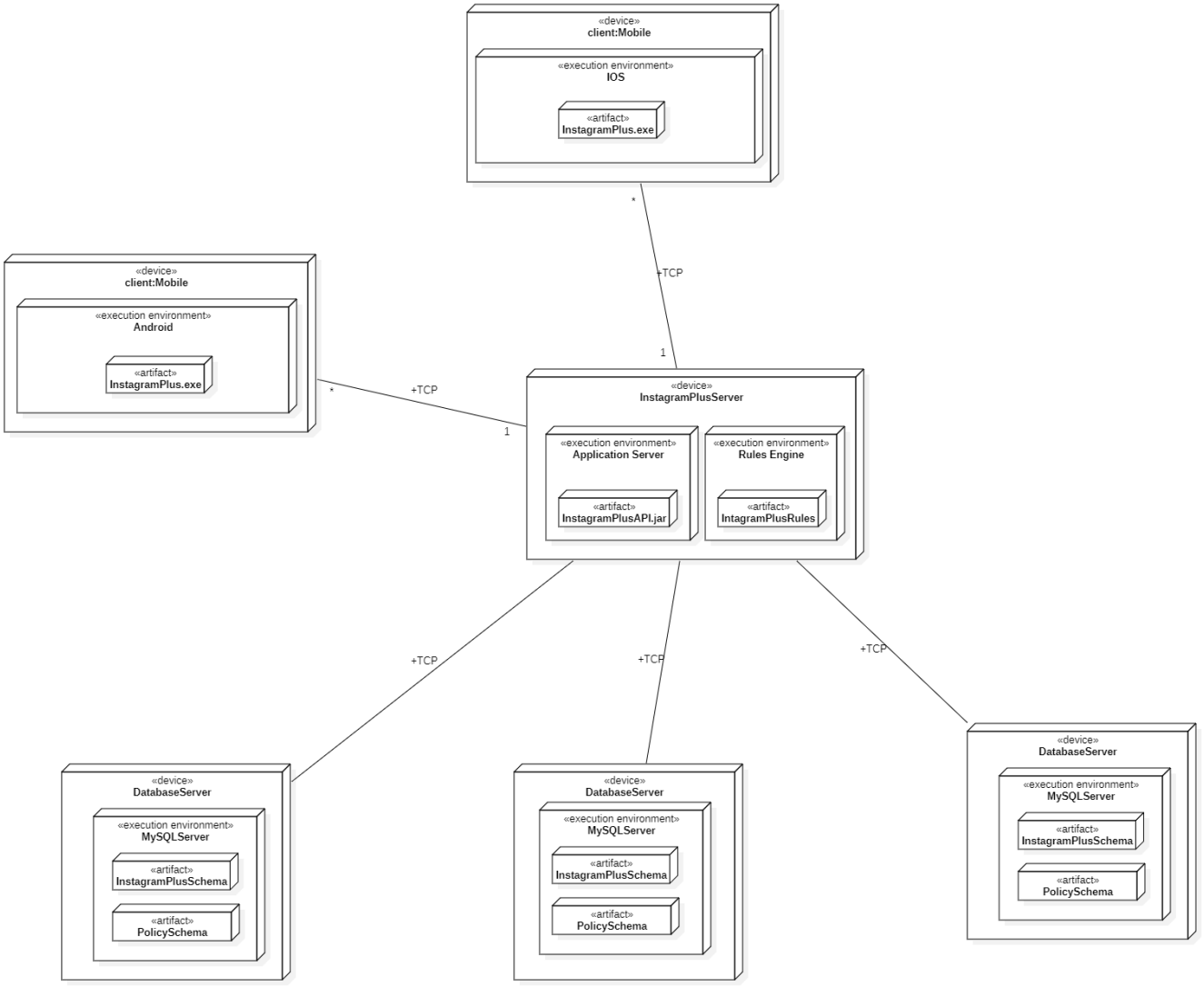
Activity Interface:

Activity Data Interface:

Search Interface:
searchUser(String username)

Search Data Interface:
searchUser(String username)

Deployment Diagram



The deployment diagram architecture used provides the following non-functional requirements: replication and availability, since the architecture provides several database servers, which ensure that data is not lost on server failure, and database service keeps on being active; security, the communication channels between the databases or devices and the InstagramPlusServer are secured, which prevent deviations and steal of information. Also, databases have crucial data encryption granted by the schemas.

Client-side:

Main Interface

- Profile Interface
- Stories Interface
- Message Interface
- Activity Interface
- Authentication Interface
- Post Interface

Instagram plus server:

- User Controller
- Story Controller
- Message Controller
- Activity Controller
- Register Controller
- Login Controller
- Post Controller
- Saved Posts and Stories Controller
- Search Controller

Database server:

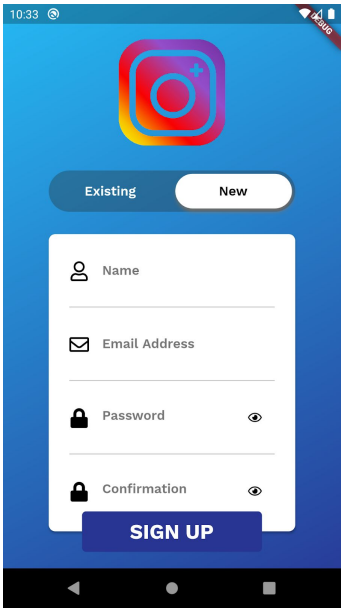
- Saved Stories
- Stories
- Story Items
- Message
- Chat
- Activity Items
- Saved Post
- Posts
- Search Page
- Reactions
- Likes
- Comments
- Save
- Share
- Effects
- Filter
- Sticker
- Face Effect

Prototype

A prototype version of the system described by the diagrams was implemented using Flutter. On a side note, the system previously described in the diagrams was complex to implement, so when building the prototype we aimed to simplify some operations in order to make it usable and more simple. The average amount of time it took to develop and test the prototype was around 1 week.

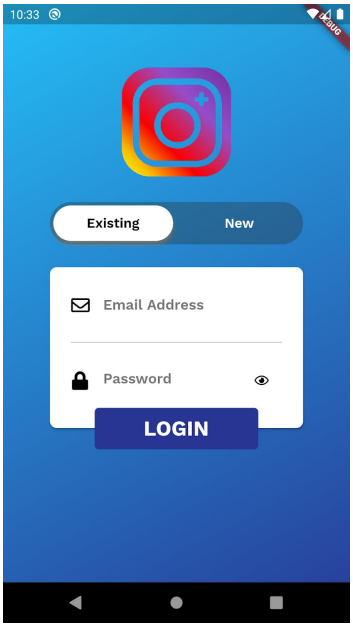
Screenshots

Register

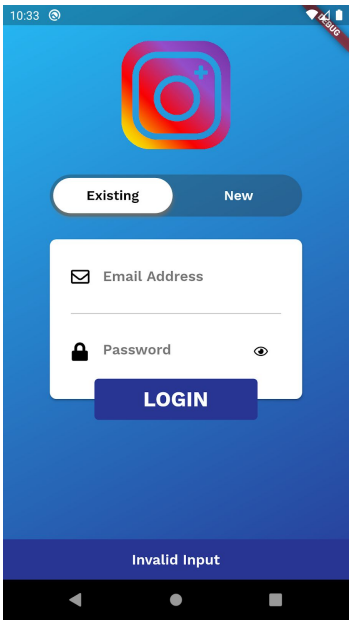


Register Page

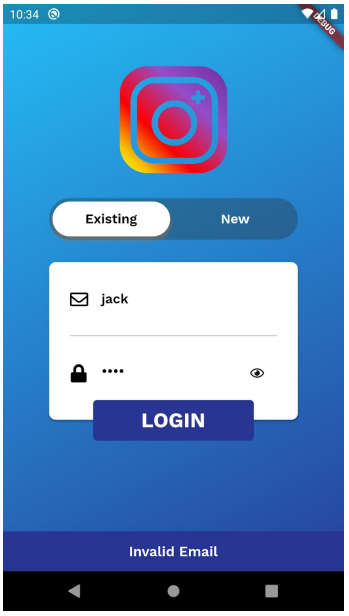
Login



Login Page

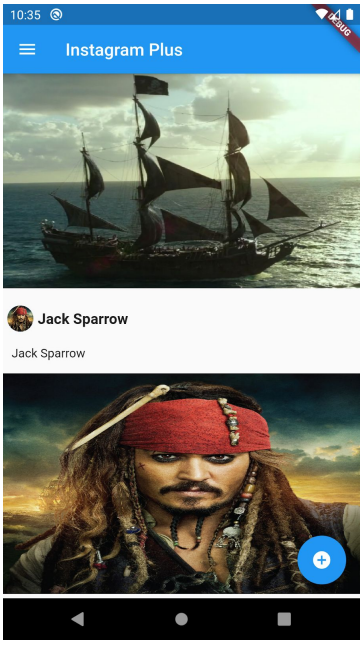


Invalid Input Error

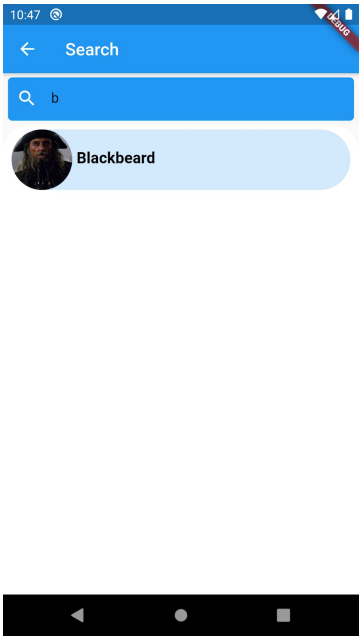


Invalid Email Error

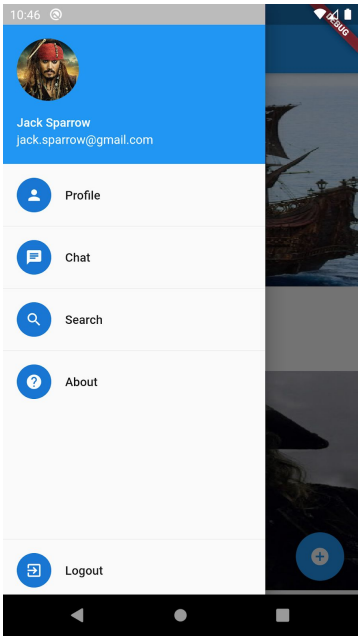
HomePage



Feed

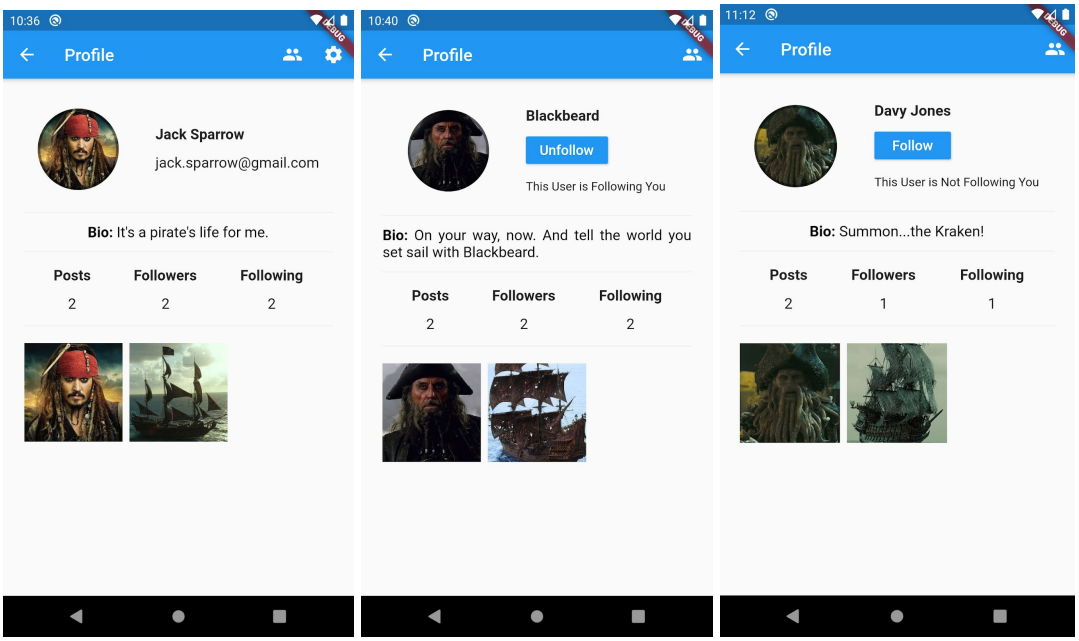


Search



Navigation Sidebar

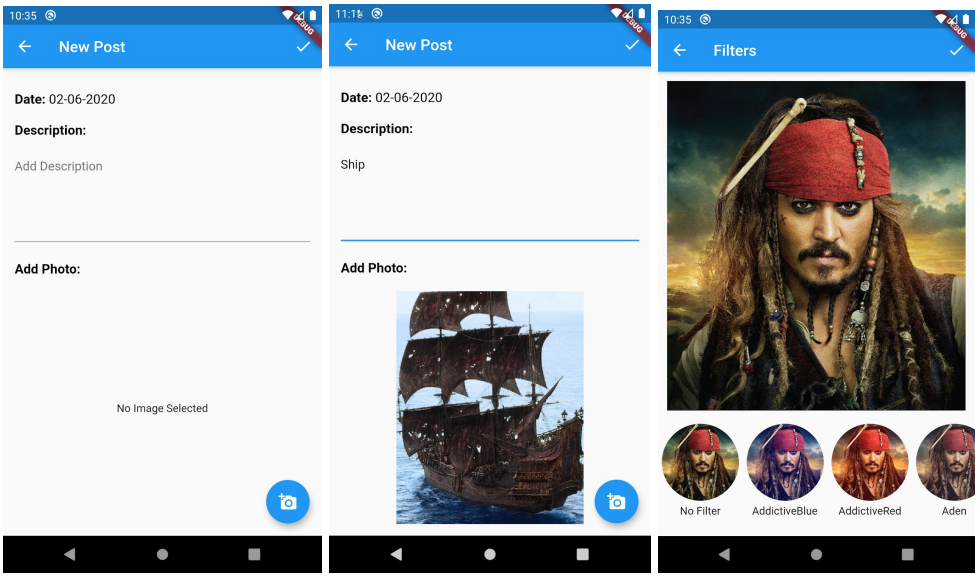
Profile Page



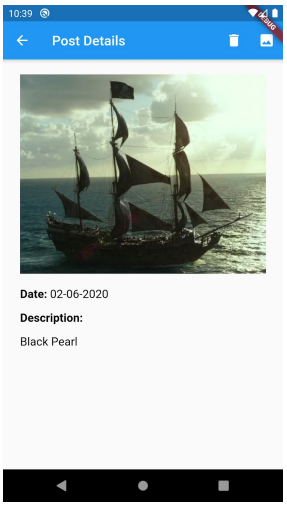
User Own Profile Public Profile (Followed) Public Profile (Unfollowed)

(The extra feature is the isFollowing Tag under the Follow Button)

Add Post

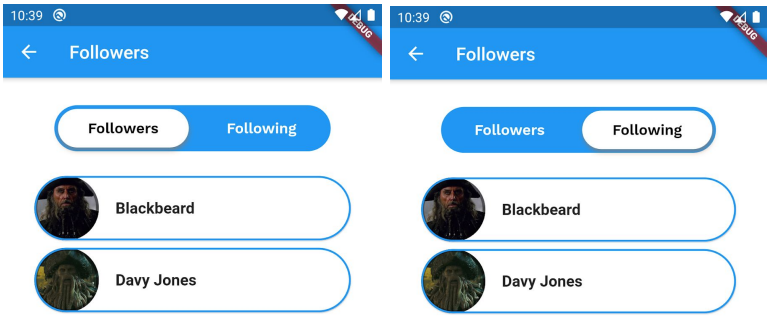


Post Info (without photos) Post Info (with photos) Post Filter



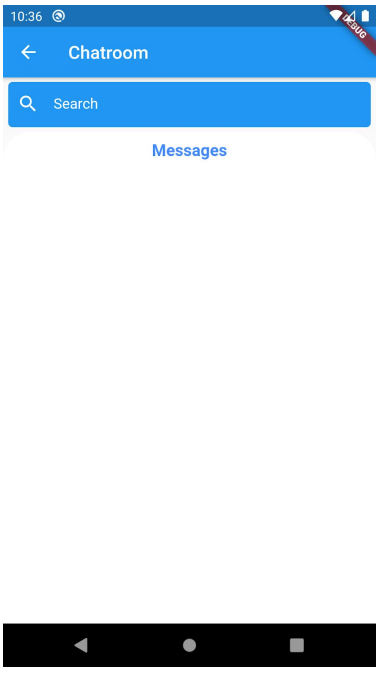
Post Detail

Follow

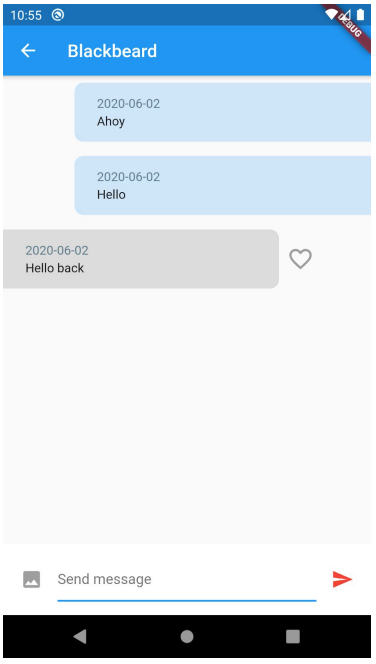


Followers Following

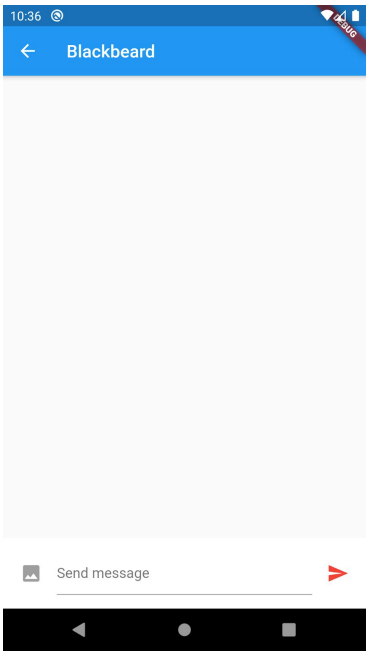
Send Message



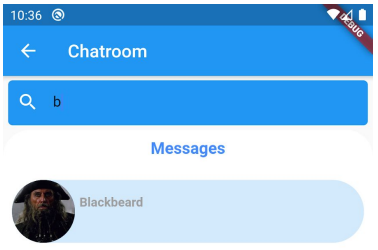
Chatroom without any chats



Chat with messages

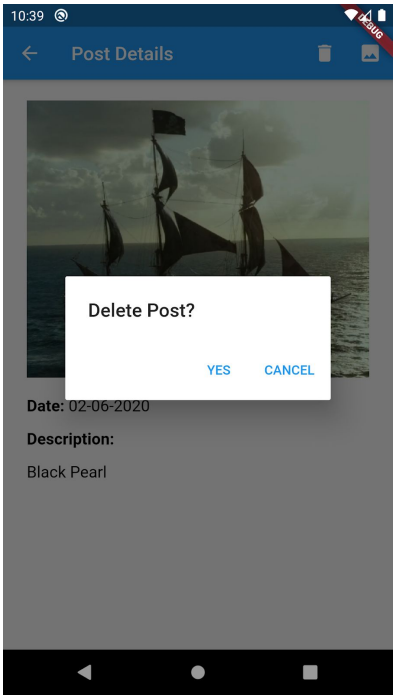


Chat without messages



Chatroom with a chat

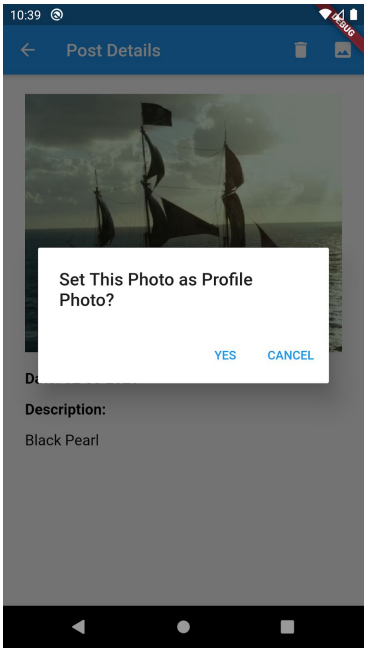
Delete Post



Delete Post

Edit Profile

Edit Profile



Change Profile Pic

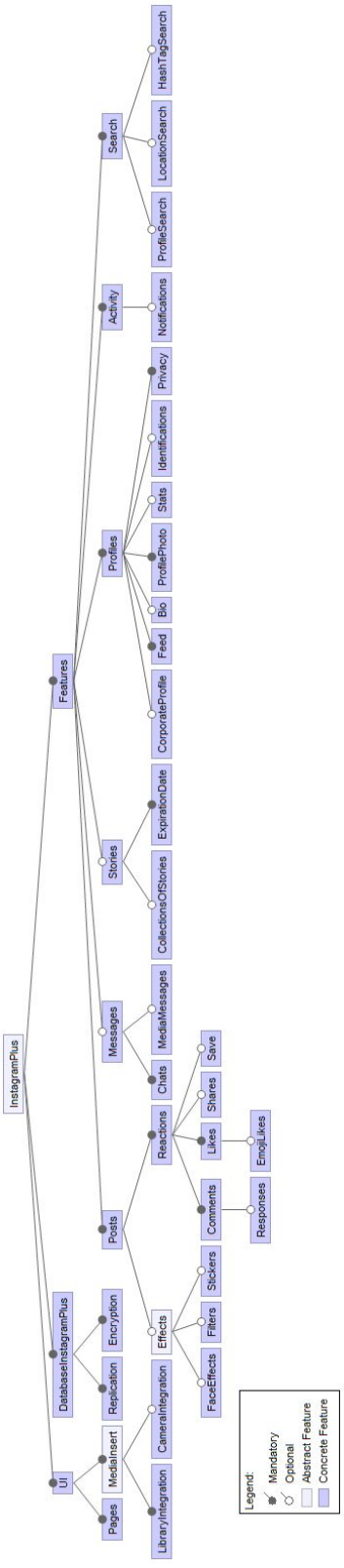
Test Cases

Functionality	Test Scenario	PreRequisites	Test Steps	Expected Results
Follow	Follow Private User Success	User is Logged In Other User account exists	1.Click other User Profile 2. Click Follow Button 3. Wait for the reply	1. Navigate to other user profile page 2. Follow Accepted
Follow	Follow Private User Refused	User is Logged In Other User account exists	1.Click User Profile 2. Click Follow Button 3. Wait for the reply	1. Navigate to other user profile page 2. Follow Denied: Database error
Is Following	Checks if an user is following the current logged in user Success	User is Logged In Other user Exists and is not Following	1.Go to the User Profile 2.Check User tag	1. Navigate to other user profile page. 2. A message saying the user is following you back is displayed
Is Following	Checks if an user is following the current logged in user Refused	User is Logged In Other user is not Following	1.Go to the User Profile 2.Check User tag	1. Navigate to other user profile page. 2. A message saying the user is not following you is displayed
Make Post	Creates a new Post Success	User is Logged In User has media library available.	1.Click “+” 2.Add photo 3.Add Filter 4 Confirm filter 5.Add a text description 6.Click on “create Post”	1. Media Library is displayed. 2. Filter Page is displayed 3. Image changed with the new Effects. 4. Return to creation page and user can enter text description 5. Text description is added 6. Post was created with success.
Make Post	Creates a new Post Refused due to user not having allowed media library to be accessed by the app	User is Logged In Use doesn’t have media library available.	1.Click “+” 2.Add Video(s) or photo(s) 3.Add effects 4.Add a text description 5.Click on “create Post”	1. Media Library Can’t be displayed and an Error messages is displayed “No permissions to access media library” with instructions on how to fix the issue. 1.5 If User fixes the issue the Media Library will be displayed, the operation will be canceled otherwise 2. Filter is displayed 3. Image changed with the new Effects. 4. Return to creation page and user can enter text description

				5. Text description is added 6. Post was created with success.
Send Message	Sends a message Success	User is Logged In Other user exists and is able to be messaged Chat Already Exists	1.User clicks on message icon 2.Selects the chat 3.Writes a message <message> 4.Clicks “send”	1. User Navigates to Message Page 2. User Navigates to Chat Page and is able to insert the message <message> 3. User can send the message 4. Message is sent
Send Message	Sends a message Refused	User is Logged In Other user exists and is able to be messaged	1.Go to the messages page 2.Searches for the chat or creates a new one 3.Writes a message 4.Clicks “send”	1. User Navigates to Message Page 2. User Navigates to Chat Page and is able to insert the message <message> 3. User can send the message 4. Message isn’t sent do to network problems and user is prompted to resend message with the error message “Message was unable to be sent. Please retry”
Delete Post	Deletes an existing post Success	User is Logged In Post Exits	1. User clicks own profile page 2. Users selects a post and clicks the trash can to delete the post 3. User confirms operation	1. User Navigated to Own Profile Page 2. Post displayed and deletion option selected 3. Post deleted and user returns to user page without displaying the post
Delete Post	Deletes an existing post Refused	User is Logged In Post Exists	1. User clicks own profile page 2. Users selects a post and clicks the trash can to delete the post 3. User confirms operation	1. User Navigated to Own Profile Page 2. Post displayed and deletion option selected 3. Post isn’t deleted due to network issues and user receives a warning Message “Post Deleted Couldn’t be deleted try again”.
Edit Profile	Edits the profile information Success	User is Logged In	1. User Click own Profile 2. Edit Profile Button Clicked 3. User writes in Bio <I’m a cool dude> 4. User clicks save button	1. User Navigated to Own Profile Page 2. User Navigated to edit profile page and is able to insert bio. 3. User can save alterations 4. Information saved with success
Edit Profile	Edits the profile	User is Logged In	1. User Click own Profile	1. User Navigated to Own

	information Refused		2. Edit Profile Button Clicked 3. User writes in Bio <I'm a cool dude> 4. User clicks save button	Profile Page 2. User Navigated to edit profile page and is able to insert bio. 3. User can save alterations 4. Information saved failed because of network failure, error message displayed "Failed to edit info, try again" User can try again

Variability Modeling - Feature Model Diagram



Configurations:

Conclusion

In this project, we fully developed a prototype from it's planning phases to the prototype itself and it was very useful to get a clearer idea about the life cycle of a product in its early phases. While we would've liked to develop more functionalities, we are happy with the results and feel that we have learned a lot about software engineering.