**Cairo University  
Faculty of Computers and Information** 

**CS251**

**Software Engineering I**

Social Network

Software Requirements Specifications

Team Names

Month & Year

**Contents**

[Team 3](#_30j0zll)

[Document Purpose and Audience 3](#_1fob9te)

[Introduction 3](#_3znysh7)

[Software Purpose 3](#_2et92p0)

[Software Scope 3](#_tyjcwt)

[Definitions, acronyms, and abbreviations 4](#_3dy6vkm)

[Requirements 4](#_1t3h5sf)

[Functional Requirements 4](#_4d34og8)

[Non Functional Requirements 6](#_2s8eyo1)

[System Models 8](#_17dp8vu)

[Use Case Model 8](#_3rdcrjn)

[Use Case Tables 9](#_26in1rg)

Class Diagram 44

Traceability Matrix 45

[Ownership Report 46](#_35nkun2)

# Team

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Email** | **Mobile** |
| 20170022 | Ahmed Sayed Mansour | ahmed.mans20719@gmail.com | 01117908189 |
| 20170002 | Ibrahim Ramadan Abdou | ibrahemramadan130@gmail.com | 01152395815 |
| 20170257 | Mohamed Mostafa Awad | mmostafaouda@gmail.com | 01122331195 |
| 20170136 | Atef Magdy Metwally | atefmagdy12@gmail.com | 01120797140 |

# Document Purpose and Audience

This Software Requirements Specification (SRS) provides an overview of the purpose, scope, functional requirements, non-functional requirements and overview of the project.

The aim of this document analyze and give an in-depth insight of the complete API by defining the problem statement in detail.

The audience to read this document client, project manager, system analysis, system designer, developer.

# Introduction

## Software Purpose

## The purpose of this project is to develop back-end API for social network.

## Something like Facebook API.

## In graph API you can use some functions to perform specific actions or get information about any component in Facebook, for example you can get any user profile picture by this call

## You should develop back-end API that is able to manage interactions between users, create and manage groups and pages, manage different type of posts and retrieve current posts in specific hashtag and sort these posts according to the most important post

## Software Scope

Our project will be a back-end API for a social network .this API will be a simple social network which allow users to create or join groups, sending messages to each other, sharing posts, using hashtags, creating pages.

## 

## Definitions, acronyms, and abbreviations

|  |  |
| --- | --- |
| **Definition** | **Description** |
| **API** | **Application programmable interface** |
| **User** | **Anyone can be a user by signup by writing his personal information or singing by writing his username and password, type of user can be normal or premium, Premium user has advanced features, user can add friends, write posts, like posts , share posts, create or join groups, create or like pages, send messages to user(s) , create message group or be a member in message group and premium user has 2 functions post ad and post premium post** |
| **Premium User** | **User with additional features like posting AD and Posting premium post** |
| **Group** | **Group is a set of users collected together can write posts in it, and using hashtags.**  **Group can be public or private, public group is open for anyone to join , private group need the account to be accepted to join it and write post in groups to be seen with others and only admin can change group info and accept members if the group is private.**  **Group is managed by a set of users (including the user who created the group) and you can search for group and leave or remove member from group.** |
| **Post** | **: user can write what he/she wants and share it with his/her friends, friends can see it, comment on it , “like” it and share it , user can neither use nor using hashtags.**  **User can write posts in his/her personal pages and joined groups.**  **Posts can be written by normal or premium user, posts should maintain number of likes and shares.** |
| **Hashtag** | **Categorize the post to specific phrase comes after “#” in a post written by user.**  **User can pull all posts contains specific phrase and read them, posts should be sorted according to post importance.** |
| **Page** | **User can create a fan page for specific entity or product.**  **User can like the page, and the page should maintain number of likes, only admin can post in this page and change page info and add admins.**  **Page contains posts written by admins.** |
| **Message** | **Message is a private text from user to another user(s) no one can see it except them, user can create group message in specific group.**  **User can get messages from any user(s).**  **User can send or receive messages in specific message group (contains more than 2 users).** |

# Requirements

## Functional Requirements

**Sign Up**

**Description**: User can sign up using this component to and enter his personal information like name, gender, profile picture and email. User can choose whether he would like to be premium which has advanced features or normal user who has basic features of the system.

**Log In**

**Description**: After user has an active account in the database he can easily sign in to this account, by entering his email and password if the system accepted relayed information, the system will let the user enter directly to his account.

**Send Message**

**Description**: After user has friend(s) in his/her friend list he/she can send message to them using this function of the system and the receiving side will see this message and who send it, the receiving user can reply to this message in the same thread by message also, message is private text content no one can see it but the sender and receiver.

**Create Group**

**Description**: With this component you can create your own group and choose it’s type (private – public), each group creator is called admin and every group can has an many admins and members, users can write posts in this group and only members in this group can see it.

**Join Group**

**Description**: With this component you can join a group and become member if the group privacy is public or sent request if the group is private.

**Search**

**Description**: With this component you can search for group, user or page. This component will get you results after you enter keyword you would like to search for.

**Write Post**

**Description**: With this component you can write post in your profile for your friends or write post in page that you are admin in or write post in group you either admin or member in the group, you can write hashtag in the post, system will automatically identify the hashtag and pull it when searched for.

**Manage Page**

**Description**: With this component you can manage your page like add admins to your page or remove them and change info about the page only admin of page can access and use this function of the system.

**Manage Group**

**Description**: With this component user can manage the group, user can accept member to his/her group, change group info or add admins to the group, only admins has access to this function.

**Like Page**

**Description**: User can like any created page(s) he/she like and after user liked it user can see posts and updates from this page.

**Share Post**

**Description**: User can share post to people in his/her friend list, after users can see, like also share the shared post to their friend list.

**Like Post**

**Description**: User can like any post to mark for post owner amount of people liked his/her post.

**Create Group Message**

**Description**: User can create group of list of users to send and receive private messages between each other, any user can create group message.

**Add Friend**

**Description**: User can add any user to their friend list to be friend and send message to each other.

**Add Advertisement**

**Description**: Premium users can add an advertisement to the system so other users on the system can see it.

## 

## Non Functional Requirements

|  |  |
| --- | --- |
|  | **Details** |
| **Performance** | ● Accessing API request will be done within 2 second.  ● System allows up to 30,000 request per minute.  ● The System is available all the time.  ● It should not be down more than 1 hour in 3  ● Years and available to use for all users. |
| **Reliability** | * Crash safe, The API software should be crash safe in 85 % of its runtime. * The System should be able to avoid invalid input and stressful environment conditions. |
| **Usability** | * Simple to use,the Software must be simple as possible that make user treat with it so easy even included a tool guide to help users. |
| **Supportability** | * Maintainability: System must be maintained by support latest technologies on client’s decision. * Adaptability: it will be only web based. |
| **Implementation** | * Programming language, Bumpers must be implemented in Java. * Hardware Platforms, Personal computers (browsers). |
| **Interface** | * Simple user interface, the user interface of API should be understandable to interface. The user interface is based on a main window, which includes a text which write E\_mailor username and button for login and other for sign up. |
| **Legal** | * This software is developed for international use under the government‘s protocols, requiring that government information systems must be accessible to people with disabilities. |
| **Packaging** | * Must be available over the web within one download. All needed files. |

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# System Models

## Use Case Model

## 

## Use Case Tables

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 1 | |
| Use Case Name: | Sign Up | |
| Actors: | User | |
| Pre-conditions: | Having email | |
| Post-conditions: | Account information saved in database | |
| Flow of events: | User Action | System Action |
| 1- User Enter name , gender , email , password , profile picture. |  |
|  | 2- System Verify user data |
|  | 3- System save information to database |
|  | 4 - System alert user |
| Exceptions: | User Action | System Action |
| 1-User Enter name , gender , email , password , profile picture. |  |
|  | 2- User information is invalid and system return to enter data again. |
| Includes: |  | |
| Notes and Issues: | user information is real | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 2 | |
| Use Case Name: | Log In | |
| Actors: | User | |
| Pre-conditions: | User signed up before | |
| Post-conditions: | API return user profile | |
| Flow of events: | User Action | System Action |
| 1-User write email & password |  |
|  | 2- System verify user data  3- system Sign in the user |
|  |  |
|  |  |
|  |  |
| Exceptions: | User Action | System Action |
| 1- User Enter Email and Password. |  |
|  | 2-Email isn’t valid or password.  3- System refuse sign in. |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 3 | |
| Use Case Name: | Create group | |
| Actors: | User | |
| Pre-conditions: | User logged in | |
| Post-conditions: | group created and saved in database | |
| Flow of events: | User Action | System Action |
| 1- User enter the name of group and type of the group. |  |
|  | 2- API Verify user data |
|  |  |
|  | 3- API alert the user |
| Exceptions: | User Action | System Action |
| 1- User Enter invalid data |  |
|  | 2- System return user to create group again |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 4 | |
| Use Case Name: | Search group | |
| Actors: | User | |
| Pre-conditions: | User logged in | |
| Post-conditions: | group will appear if found | |
| Flow of events: | User  Action | System Action |
| 1- User enter the name of group |  |
|  | 2-System will return group info |
|  |  |
| Exceptions: | User Action | System Action |
| 1- User Enter invalid group |  |
|  | 2- System return not found |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 5 | |
| Use Case Name: | write a post in group | |
| Actors: | User | |
| Pre-conditions: | User logged in and a member in a group | |
| Post-conditions: | post added to group | |
| Flow of events: | User Action | System Action |
| 1- User adds a post to a group. |  |
|  | 2-System adds the post to the group |
|  |  |
|  |  |
| Exceptions: | User Action | System Action |
|  |  |
|  |  |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 6 | |
| Use Case Name: | Add group admin | |
| Actors: | Admin | |
| Pre-conditions: | Admin logged in . | |
| Post-conditions: | Admin is added. | |
| Flow of events: | User Action | System Action |
| 1- Admin added a new admin |  |
|  | 2- system approve user to be admin. |
|  |  |  |
| Exceptions: | User Action  1-Admin adds a non-existent member | System Action  2-System refuse to make him admin |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 7 | |
| Use Case Name: | Admin adds new members to group | |
| Actors: | Admin | |
| Pre-conditions: | Admin logged in . | |
| Post-conditions: | new members are added. | |
| Flow of events: | User Action | System Action |
| 1- Admin check pending list. |  |
|  | 2- system respond with pending list. |
|  | 3-Admin approves new members | 4-system adds this members to the group. |
| Exceptions: | User Action | System Action |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 8 | |
| Use Case Name: | Admin changes group info. | |
| Actors: | Admin | |
| Pre-conditions: | Admin logged in. | |
| Post-conditions: | Group info is changed. | |
| Flow of events: | User Action | System Action |
| 1- Admin choose a group attribute to change. |  |
|  | 2- system respond with old attribute . |
|  | 3-Admin changes this attribute | 4-System changes old attribute by the new one. |
| Exceptions: | User Action | System Action |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 9 | |
| Use Case Name: | Admin remove a member from the group. | |
| Actors: | Admin | |
| Pre-conditions: | Admin logged in. | |
| Post-conditions: | member is removed. | |
| Flow of events: | User Action | System Action |
| 1- Admin choose a member from group. |  |
|  | 2- system respond with this member. |
|  | 3-Admin removes this member. | 4-System remove this member from the group. |
| Exceptions: | User Action | System Action |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 10 | |
| Use Case Name: | User leave a group. | |
| Actors: | user | |
| Pre-conditions: | user logged in. | |
| Post-conditions: | user is removed. | |
| Flow of events: | User Action | System Action |
| 1- user choose a group. |  |
|  | 2- system respond with this group. |
|  | 3- user leaves this group. | 4-System remove this user from the group. |
| Exceptions: | User Action | System Action |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 11 | |
| Use Case Name: | Join Group | |
| Actors: | User | |
| Pre-conditions: | User logged in | |
| Post-conditions: | user joined group | |
| Flow of events: | User Action | System Action |
| 1- User request to join group |  |
|  | 2- system approve user to group if it is public or waits for Admin approve if is private. |
|  |  | 3- system alert user when it approved. |
| Exceptions: | User Action | System Action |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 12 | |
| Use Case Name: | Create Page | |
| Actors: | User | |
| Pre-conditions: | User logged in | |
| Post-conditions: | User has new page created | |
| Flow of events: | User Action | System Action |
| 1- User enter page’s name , type and enter choose page picture |  |
|  | 2-System verify data |
|  | 3- System save page in database |
|  | 4- System alert user |
| Exceptions: | User Action | System Action |
| 1- User enter wrong information |  |
|  | 2- System alert user and return him to Create page |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 13 | |
| Use Case Name: | Add page admin | |
| Actors: | Admin | |
| Pre-conditions: | Admin logged in | |
| Post-conditions: | A new Admin is added. | |
| Flow of events: | User Action | System Action |
| 1- Admin added a new admin |  |
|  | 2- system approve user to be admin. |
|  |  |  |
| Exceptions: | User Action | System Action |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 14 | |
| Use Case Name: | write a post in page | |
| Actors: | admin | |
| Pre-conditions: | admin logged in | |
| Post-conditions: | post added to page | |
| Flow of events: | User Action | System Action |
| 1- admin adds a post to a page. |  |
|  | 2-System adds the post to the page |
|  |  |
|  |  |
| Exceptions: | User Action | System Action |
|  |  |
|  |  |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 15 | |
| Use Case Name: | Like Page | |
| Actors: | User | |
| Pre-conditions: | User logged in | |
| Post-conditions: | system increase the number of page likes | |
| Flow of events: | User Action | System Action |
| 1-user search for a page. |  |
|  | 2-system return page info. |
| 3-user likes the page. | 4- system increase the number of page likes. |
| Exceptions: | User Action | System Action |
| 1- User search for non\_existing page |  |
|  | 2- system doesn’t return anything. |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 16 | |
| Use Case Name: | Unlike Page | |
| Actors: | User | |
| Pre-conditions: | User logged in , and like page | |
| Post-conditions: | system decrease the number of page likes | |
| Flow of events: | User Action | System Action |
| 1-user search for a page. |  |
|  | 2-system return page info. |
| 3-user Unlikes the page. | 4- system decrease the number of page likes. |
| Exceptions: | User Action | System Action |
| 1- User search for non\_existing page |  |
|  | 2- system doesn’t return anything. |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 17 | |
| Use Case Name: | Search Page | |
| Actors: | User | |
| Pre-conditions: | User logged in | |
| Post-conditions: | system show page if found | |
| Flow of events: | User Action | System Action |
| 1-user search for a page. |  |
|  | 2-system return page info. |
| Exceptions: | User Action | System Action |
| 1- User search for non\_existing page |  |
|  | 2- system doesn’t return anything. |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 18 | |
| Use Case Name: | share post | |
| Actors: | User | |
| Pre-conditions: | User logged in and friend with post writer | |
| Post-conditions: | Friends with user who shared post will be able to see the post | |
| Flow of events: | User Action | System Action |
| 1- User share post |  |
|  | 2- System make post available to users friends to see |
|  | 3- system alert user |
| Exceptions: | User Action | System Action |
|  |  |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 19 | |
| Use Case Name: | Write Post in profile | |
| Actors: | User | |
| Pre-conditions: | User logged in | |
| Post-conditions: | System show post where user writes | |
| Flow of events: | User Action | System Action |
| 1- User write block of text |  |
|  | 2- System save post in database |
|  | 3- System show post to inner circle where post is written |
| Exceptions: | User Action | System Action |
|  |  |  |
| Includes: |  | |
| Notes and Issues: | to post in group you should be a member in this group. | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 20 | |
| Use Case Name: | Like post | |
| Actors: | User | |
| Pre-conditions: | User logged in and friend with post writer | |
| Post-conditions: | The number of likes in this post will increase. | |
| Flow of events: | User Action | System Action |
| 1- User like post |  |
|  | 2- System increase the number of likes in this post |
| Exceptions: | User Action | System Action |
|  |  |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 21 | |
| Use Case Name: | Unlike post | |
| Actors: | User | |
| Pre-conditions: | User logged in and friend with post writer | |
| Post-conditions: | The number of likes in this post will decrease. | |
| Flow of events: | User Action | System Action |
| 1- User Unlike post |  |
|  | 2- System decrease the number of likes in this post |
| Exceptions: | User Action | System Action |
|  |  |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 22 | |
| Use Case Name: | Remove post | |
| Actors: | User | |
| Pre-conditions: | User logged in and friend with post writer | |
| Post-conditions: | Post is removed. | |
| Flow of events: | User Action | System Action |
| 1- User remove post |  |
|  | 2- System will remove post |
| Exceptions: | User Action | System Action |
|  |  |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 23 | |
| Use Case Name: | Send message to user(s) | |
| Actors: | User. | |
| Pre-conditions: | User Logged in and friend with user(s) trying to send message to. | |
| Post-conditions: | Receiver user(s) will have message from sender user and able to read it | |
| Flow of events: | User Action | System Action |
| 1- User write message and send it |  |
|  | 2- System verify user message |
|  | 3- System send message to the receiver user(s) |
|  |  | 4- System save message to database |
| Exceptions: | User Action | System Action |
| 1- User write message |  |
|  |  | 2- System alert user of empty message |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 24 | |
| Use Case Name: | Get message from user. | |
| Actors: | User. | |
| Pre-conditions: | User Logged in and friend with user trying to get message from. | |
| Post-conditions: | Use will have message from sender user and able to read it. | |
| Flow of events: | User Action | System Action |
|  | 1- Get message from sender user. |
| 2- User sees the message. |  |
|  |  |
|  |  |  |
| Exceptions: | User Action | System Action |
|  |  |
|  |  |  |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 25 | |
| Use Case Name: | Get message from message group. | |
| Actors: | User. | |
| Pre-conditions: | User Logged in and be a member in this group. | |
| Post-conditions: | Use will have message from message group and able to read it. | |
| Flow of events: | User Action | System Action |
|  | 1- Get message from message group. |
| 2- User gets the message and able to read it. |  |
| Exceptions: | User Action | System Action |
|  |  |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 26 | |
| Use Case Name: | Create message group. | |
| Actors: | User. | |
| Pre-conditions: | User Logged in and friend with users trying to send message to. | |
| Post-conditions: | System sends the message to all friends he chosen. | |
| Flow of events: | User Action | System Action |
| 1- User create message group. |  |
|  | 2- System add all users in specific group. |
|  |  | 3- System save message group in database. |
| Exceptions: | User Action | System Action |
|  |  |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 27 | |
| Use Case Name: | Write Hashtag | |
| Actors: | User | |
| Pre-conditions: | User writing post | |
| Post-conditions: | Hashtag added to this post | |
| Flow of events: | User Action | System Action |
| 1- User write post and write hashtag. |  |
|  | 2- System categorize this post depending on this hashtag. |
|  |  | 3- System save post in database. |
| Exceptions: | User Action | System Action |
|  |  |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 28 | |
| Use Case Name: | search for Hashtag | |
| Actors: | User | |
| Pre-conditions: | User write hashtag | |
| Post-conditions: | All posts with this hashtag is shown. | |
| Flow of events: | User Action | System Action |
| 1- User write hashtag. |  |
|  | 2- System verify hashtag. |
|  | 3- System show all posts with hashtags. |
| Exceptions: | User Action | System Action |
| 1- User write hashtag. |  |
|  |  | 2- System verify hashtag |
|  |  | 3- System alert user if the hashtag not available or not used before. |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 29 | |
| Use Case Name: | Add friend | |
| Actors: | User | |
| Pre-conditions: | User logged in | |
| Post-conditions: | Friend request sent to the receiver user | |
| Flow of events: | User Action | System Action |
| 1- User Send friend request |  |
|  | 2- System verify request |
|  | 3- System send friend request to user |
|  |  | 4- System save information in database |
|  |  | 5- System alert sender user |
| Exceptions: | User Action | System Action |
|  |  |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 30 | |
| Use Case Name: | Search Profile | |
| Actors: | User | |
| Pre-conditions: | User logged in | |
| Post-conditions: | profile info will appear if found | |
| Flow of events: | User Action | System Action |
| 1- Enter profile name |  |
|  | 2- System search the name |
|  | 3- System return profile info |
| Exceptions: | User Action | System Action |
| 1- Enter wrong profile name |  |
|  |  | 2- System return not found |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 31 | |
| Use Case Name: | Remove friend | |
| Actors: | User | |
| Pre-conditions: | User logged in and friend with the user | |
| Post-conditions: | Friend removed from the friend list of the two users | |
| Flow of events: | User Action | System Action |
| 1- User remove friend |  |
|  | 2- System verify action |
|  | 3- System remove friend from the friend list from the two users |
|  |  |  |
|  |  |  |
| Exceptions: | User Action | System Action |
|  |  |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 32 | |
| Use Case Name: | Send message to group | |
| Actors: | User. | |
| Pre-conditions: | User Logged in and member in message group trying to send message to. | |
| Post-conditions: | Receiver user(s) will have message from sender user and able to read it | |
| Flow of events: | User Action | System Action |
| 1- User write message and send it |  |
|  | 2- System verify user message |
|  | 3- System send a message to the group |
|  |  | 4- System save message to database |
| Exceptions: | User Action | System Action |
| 1- User write message |  |
|  |  | 2- System alert user of empty message |
| Includes: |  | |
| Notes and Issues: |  | |

## 

## 

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 33 | |
| Use Case Name: | Change page info | |
| Actors: | Page admin | |
| Pre-conditions: | User logged in and admin in this page | |
| Post-conditions: | Friend request sent to the receiver user | |
| Flow of events: | User Action | System Action |
| 1- User write new page info. |  |
|  | 2- System verify info. |
|  | 3- System apply new info. |
|  |  | 4- System save information in database |
|  |  | 5- System alert sender user |
| Exceptions: | User Action | System Action |
| 1- User enters new info |  |
|  |  | 2- System alert user of wrong information |
| Includes: |  | |
| Notes and Issues: |  | |

## 

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 34 | |
| Use Case Name: | add premium post | |
| Actors: | premium user | |
| Pre-conditions: | premium user logged in | |
| Post-conditions: | premium post | |
| Flow of events: | User Action | System Action |
| 1- User write new premium post. |  |
|  | 2- System adds new premium post. |
|  |  |
|  |  |  |
|  |  |  |
| Exceptions: | User Action | System Action |
|  |  |
|  |  |  |
| Includes: |  | |
| Notes and Issues: |  | |

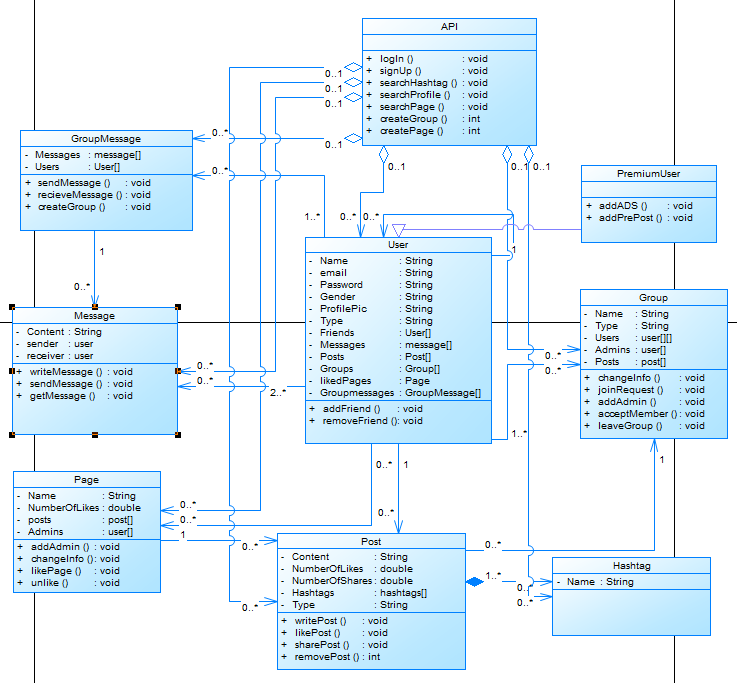
## 

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 35 | |
| Use Case Name: | add an ad | |
| Actors: | premium user | |
| Pre-conditions: | premium User logged in | |
| Post-conditions: | ad is added | |
| Flow of events: | User Action | System Action |
| 1- User write new ad. |  |
|  | 2- System verify ad. |
|  | 3- System adds this new ad. |
|  |  |  |
|  |  |  |
| Exceptions: | User Action | System Action |
|  |  |
|  |  |  |
| Includes: |  | |
| Notes and Issues: |  | |

## 

## 

## Class Diagram



# Traceability Matrix

|  |  |  |
| --- | --- | --- |
| **BR#**  **( business/functional requirement number )** | **Use Case ID** | **Class(es) Name(s)** |
| User | 29,31,34,35 | User , premium user |
| Group | 5,6,7,8,9,10,11 | Group |
| Message | 23,24,25,26,32 | Message , Message Group |
| Post | 5,14,18,19,20,21,22,27 | Post |
| Hashtag | 27 | Hashtag |
| Page | 13,15,16,33 | Page |
| API | 1,2,3,4,12,17,28,30 | API , Page |

# 

# Ownership Report

|  |  |
| --- | --- |
| **Item** | **Owners** |
| Use Case diagram | *Ahmed Sayed Mansour* |
| Class Diagram | *Mohamed Mostafa / Atef Magdi* |
| Non-functional Requirements | *Ahmed Sayed Mansour / Ibrahim Ramadan* |
| Functional Requirements | *Mohamed Mostafa / Atef Magdy* |
| Use table | *Ibrahim Ramadan / Atef Magdy* |