Cairo University  
Faculty of Computers and AI



**CS251**

**Software Engineering I**

Social Network

Software Design

(Phase 2a)

December & 2019

Contents

[Instructions [To be removed] 3](#_Toc25570060)

[Team 3](#_Toc25570061)

[Document Purpose and Audience 3](#_Toc25570062)

[System Models 3](#_Toc25570063)

[I. System Decomposition 3](#_Toc25570064)

[II. Class diagrams 6](#_Toc25570065)

[Important Algorithm 7](#_Toc25570066)

[III. Sequence diagrams 7](#_Toc25570067)

[Class - Sequence Usage Table 9](#_Toc25570068)

[Ownership Report 10](#_Toc25570069)

[Policy Regarding Plagiarism: 10](#_Toc25570070)

[References 10](#_Toc25570071)

[Authors 10](#_Toc25570072)

# Team

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Email** | **Mobile** |
| 20160198 | Mohamed Hussein Desouki | m.h.d.a.2030@gmail.com | 01142029729 |
| 20160003 | Ibrahim Elsayed Ibrahim |  |  |
| 20160025 | Ahmed Maher Ali Zahran | [Ahmedmaher072@gmail.com](mailto:Ahmedmaher072@gmail.com) | 01201001906 |
| 20160036 | Ahmed Mohamed Yassein | Ahmedyassein.97@gmail.com | 01151265044 |

# 

# Document Purpose and Audience

## This document specifies the Software Requirements Specification (SRS) for the Project Management System (PMS). It describes scope of the system, both functional and non-functional requirements for the software, design constraints and system interfaces.

# The target audiences of the best practices group are:

* end users (all those who use social networks)
* developers of social networking platforms or technologies (e.g., software publishers)
* operators of social networks or social networking technologies (e.g., existing community operators, handset manufacturers)
* the business own , marketer employees

Note: in the other SRS the list not organized and not complete

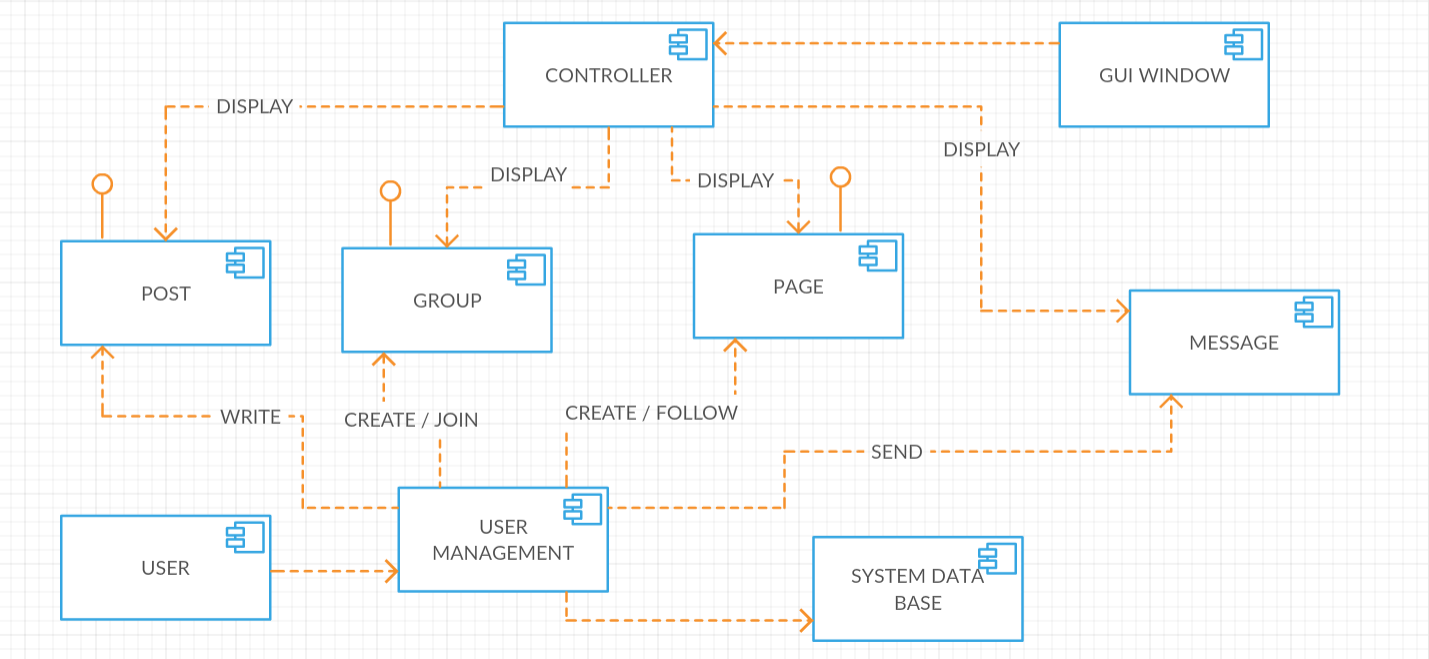
## Software Purpose

* 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 (https://graph.facebook.com/[USER\_ID]/picture)
* 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
* Note: this is software purpose of SRS that is develop back-end API for social network not like the other version.

# System Models

## System Decomposition

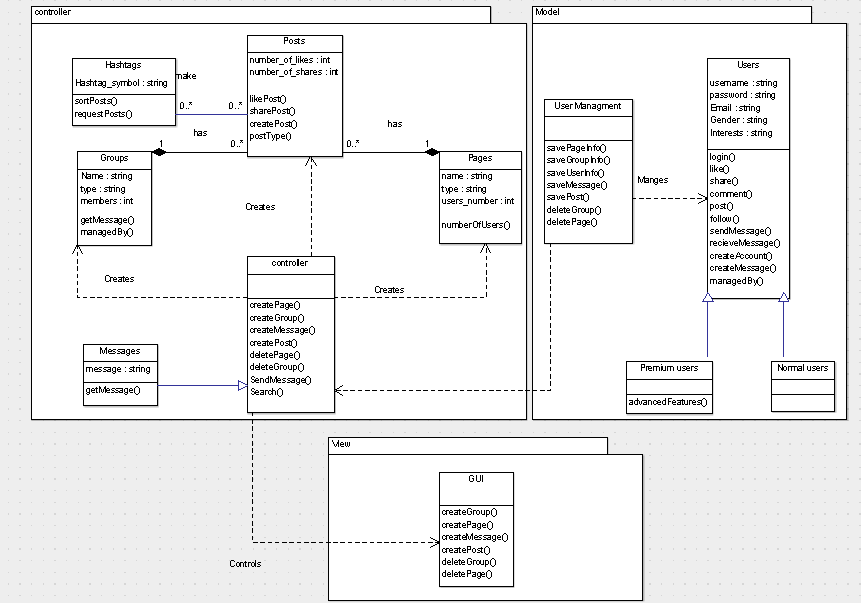
## Component Diagram

****

## System Architecture

## 

## Class diagrams



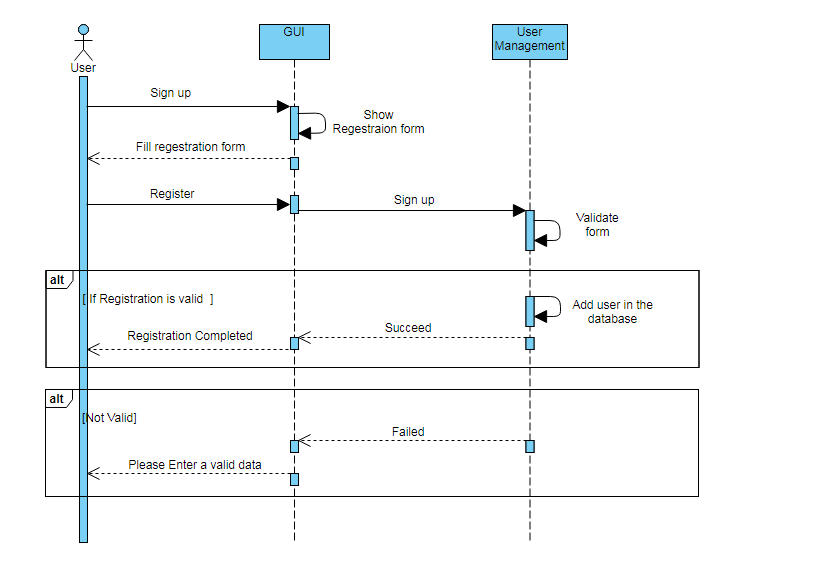
| **Class ID** | **Class Name** | **Subsystem ID** | **Description & Responsibility** |
| --- | --- | --- | --- |
| 1 | User | Model | first user sign up by entering some info then login to the system and he can interact with posts and create group or delete it or join to existing one and create page or delete it or follow an existing one and he can write a message with friends or into group , Done by user. |
| 2 | Controller | Controller | Control everything in the system once the user click on it in the GUI, It do the actually work into the system. |
| 3 | Post | Controller | Contains all data of the post (number and name who like it and comments and number of shares and hashtags). |
| 4 | Group | Controller | Contains all data of the group (numbers of members and names of each member and posts of each member). |
| 5 | Page | Controller | Contains all data of the page (numbers of followers and names of each follower and posts of each page admin). |
| 6 | Message | Controller | Chat box contain all messages between two or more user. |
| 7 | Hashtag | Controller | A part of the post used to identify the post when the user wishes to search for similar posts or contains the same topic. |
| 8 | User management | Model | Responsible for keeping all user actions on the system and sending them to the controller for actual implementation on the system. |
| 9 | GUI | View | Used to display all system contents to the user in a beautiful way. |

### 

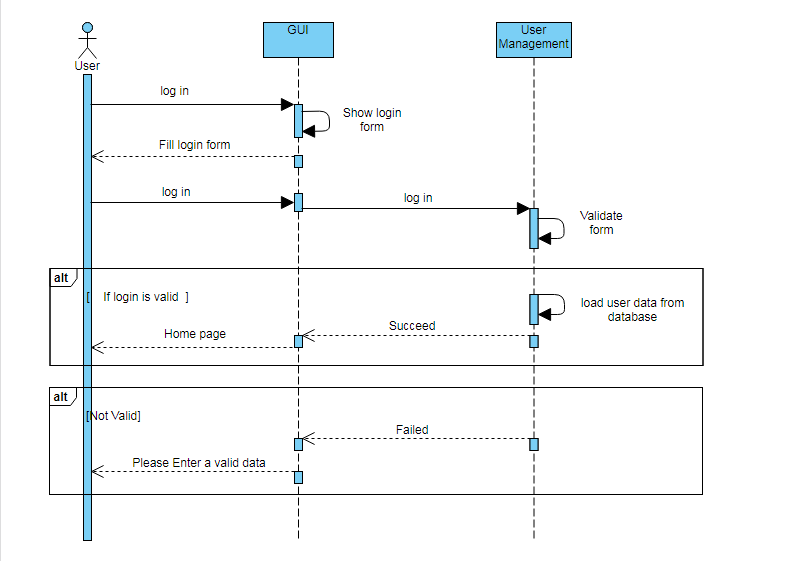
## 

## III. Sequence diagrams

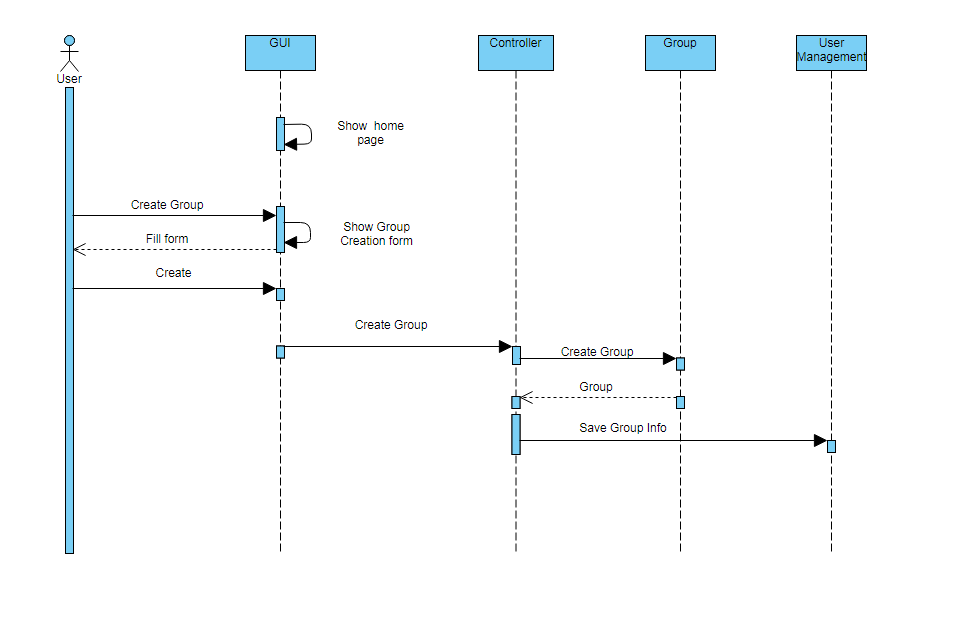
**Sequence ID:** 1 **Sequence Description:** This diagram describes how Users register in the system.

****

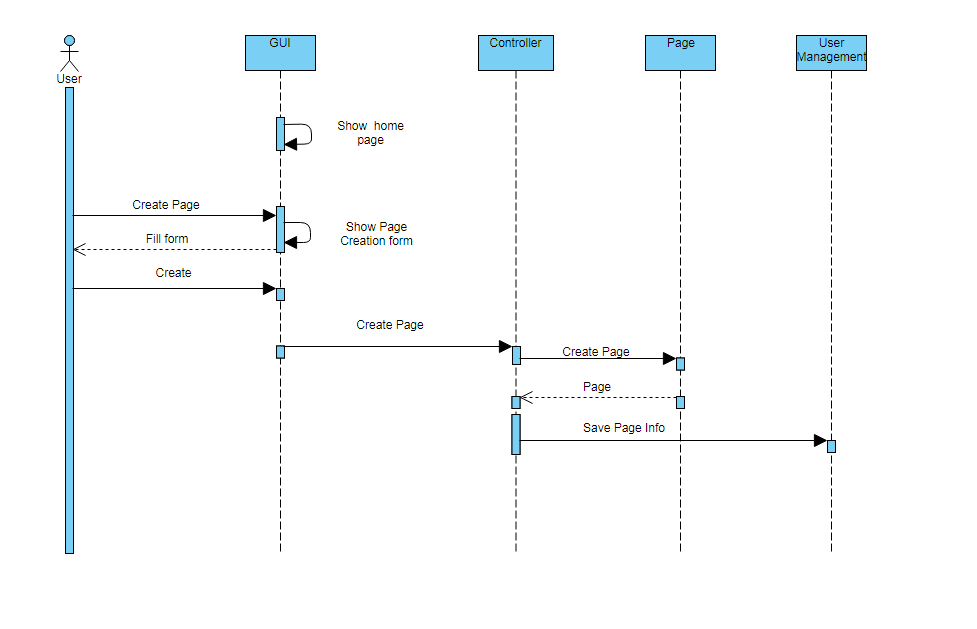
**Sequence ID:** 2 **Sequence Description:** This diagram describes how Users login the system.



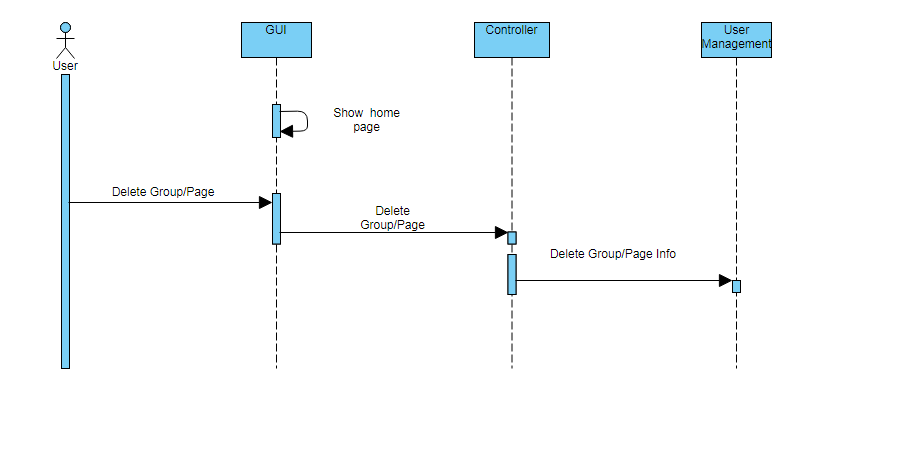
**Sequence ID:** 3 **Sequence Description:** This diagram describes how Users Creates groups.



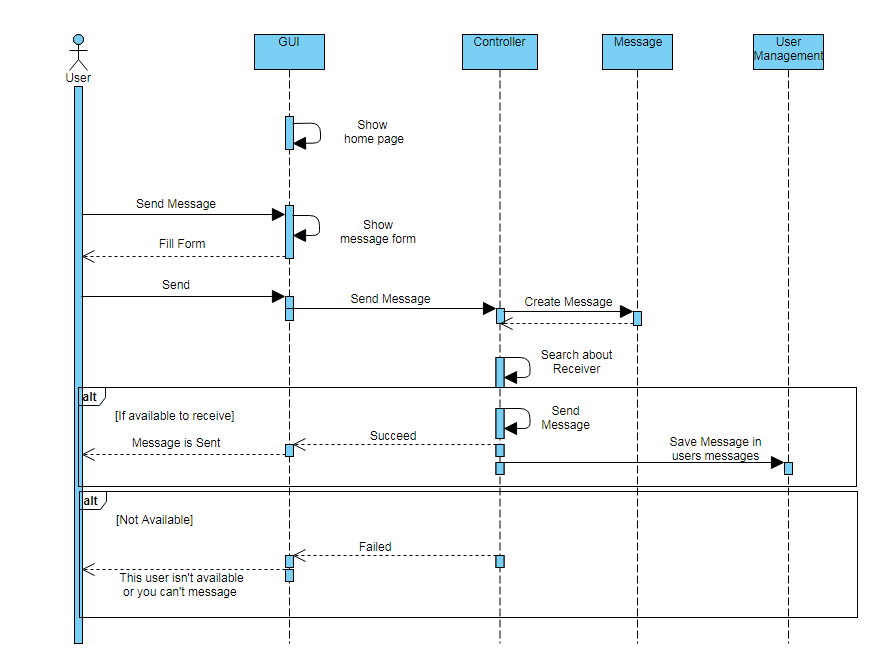
**Sequence ID:** 4 **Sequence Description:** This diagram describes how Users Creates Pages.



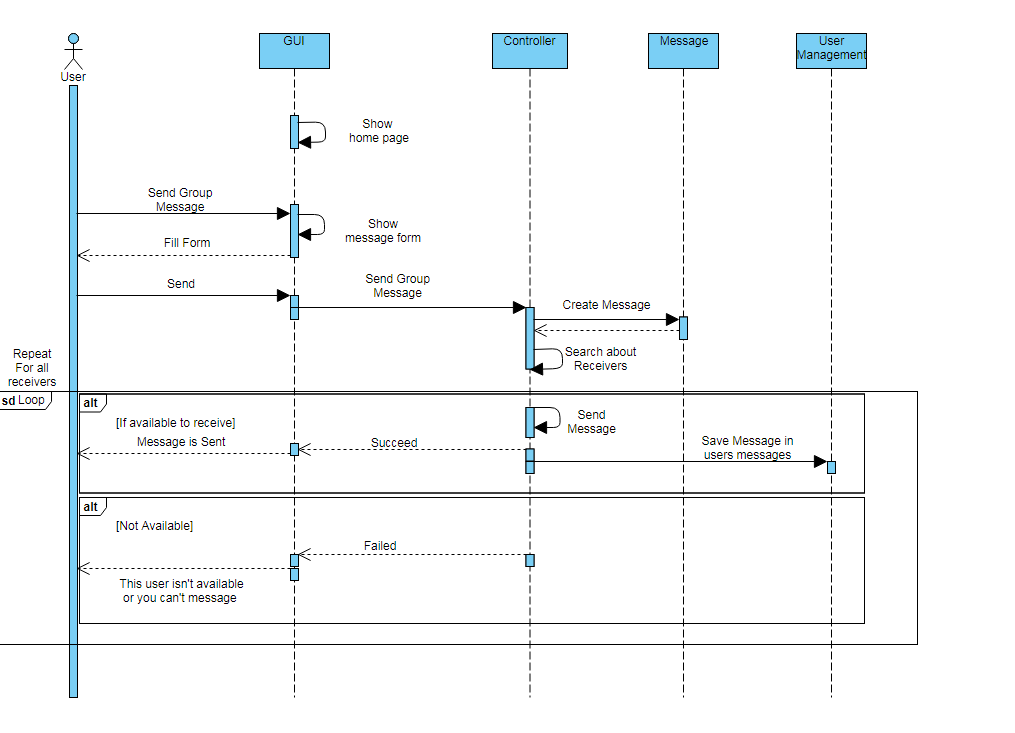
**Sequence ID:** 5 **Sequence Description:** This diagram describes how Users Delete Group/Page.



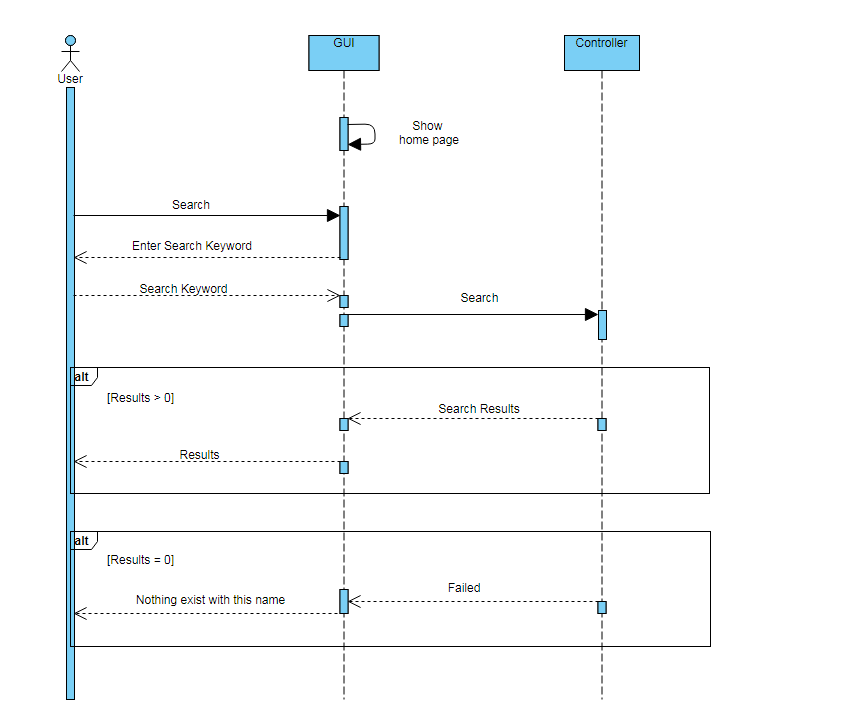
**Sequence ID:** 6 **Sequence Description:** This diagram describes the process of messaging.



**Sequence ID:** 7 **Sequence Description:** This diagram describes the process of group messaging.



**Sequence ID:** S1 **Sequence Description:** This diagram describes the process of searching.



### Sequence Usage Table

| **Class Name** | **Sequence Diagrams** | **Overall used methods** |
| --- | --- | --- |
| Controller | 3,4,5,6,7,8 | Create Page, Create Post, Create Group, Create Message |
| Message | 7,8 | Get Message |
| Page | 4 | Get Page |
| Group | 3 | Get Group |
| User Management | 1,2,3,4,5,6,7 | Save page info, save group info, save user info, Save Message, Save Post, Delete group, Delete page |

## Non Functional Requirements

* Every Post must show only 5 comments
* System able to load the page in under 3 seconds
* Hash tag Page must contain only 7 Posts
* you able to create message from user to any other user(s), or get messages between 2 users
* Every Group must be more than 2 members
* Group Page can contain more than 10 Posts
* System allows up to 22,000 Transaction per minutes

|  |  |
| --- | --- |
|  | **Details** |
| **Scalability** | * API should be able handle a large number of users, for handling around 22,000 at the same time |
| **Speed** | * The API should be fast not slow down with increase the number of users, and the API should be fast to response to the user actions within a short period of time when the user need to create (pages , posts , …) |
| **Security** | * During the user register the given email address is validated * The password should be at least 8 character and stores as a hash function in database * No one can see the private messages of other users |
| **Usability** | * The API provide easy way to create or manage (post, group…). |
| **Maintainable** | * The API be developed in such a way that is extensible.it easy to incorporate new feature requirements or apply changes |
| **Availability** | * The API can reach to use any time |
| **Reliability** | * The API agrees on one up-to-date copy of data * The API returns a response, whether it succeeds or fails * The API continues to function in the presence of network failures |
| **supportability** | * API adapts with changes in case have changes * Can changes if use new technology of development of API |

# Ownership Report

|  |  |
| --- | --- |
| **Item** | **Owners** |
| Component diagram , List class and describe them | Ahmed Yassein |
| Sequence diagram , Sequence diagram and usage table | Ahmed Maher |
| system architecture , correct the missing parts in phase one which is in the feedback | Mohamed Hussein |
| Sequence diagram , class diagram | Ibrahim Elsayed |