Project Title : Social Network

**Software Requirements Specifications for Social Network**

**Contents**

[Team 3](#_bookmark0)

[Document Purpose and Audience 3](#_bookmark1)

[Introduction 3](#_bookmark2)

[Software Purpose 3](#_bookmark3)

[Software Scope 4](#_bookmark4)

[Definitions, acronyms, and abbreviations 4](#_bookmark5)

[Requirements 4](#_bookmark6)

[Functional Requirements 4](#_bookmark7)

[Non Functional Requirements 6](#_bookmark8)

[System Models 7](#_bookmark9)

[Use Case Model 7](#_bookmark10)

[Use Case Tables 7](#_bookmark11)

[Ownership Report 12](#_bookmark12)

# Document Purpose and Audience

## This document is presenting a software requirements specifications, which is a description of a software system to be developed, laying out functional, non- functional requirements, and system models. It is for a back-end API (Application Programming Interface) for a Social Network, which is a dedicated website which enables users to communicate with each other by posting information, comments, messages, images, like Facebook, etc..

## It targets developers (to be able to maintain or develop functions), software engineers and business developers who are trying to develop their professional network or create a new one.

## It targets also the project manager, who is responsible for the whole project.

# Introduction

**Software Purpose**

* The purpose of the software is to develop a back-end API that will give the audience the basic features for a social network, to be able to develop a new social network and to add to it his own new features.
* It is also able to manage interactions between users, manage pages, groups and posts; with its different types. Adding some other features like hashtags and messages.

**Software Scope**

* This purpose of the software scope is to clarify for the actors what functions could be done within this system, it includes these major components:
  1. Users: Its different types (Normal and Premium).
  2. Groups: Create, manage groups, types of groups with respect to its privacy, and how users manage groups.
  3. Pages: Created and managed by users.
  4. Posts: Manage likes, shares from and its privacy according to different things (from profile/ page/ group) and the different types of posts.
  5. Hashtags: Categorize contents of specific posts and arrange the reviewed posts according to user type's importance.
  6. Messages: Sending messages to people from your friend list, get messages and review group messages.

7-Friends: Relation among social network users.

**Definitions, acronyms, and abbreviations**

|  |  |
| --- | --- |
| **Definitions/**  **Abbreviations** | **Description** |
| **API** | **Stands for: Application Programming Interface, is the way for an**  **application to interact with certain system/ application/ library, etc..** |
| **Hashtag** | **A word prefixed with the hash character #, to form a label. It allows grouping of similarly tagged messages, and also allows an electronic**  **search to return all messages that contain it.** |
| **Premium User** | **A user with advanced feautures.** |
| **Fan Page** | **A page created and maintained by a fan. It could have many**  **different purposes like: Local business, product, art, communities, celebrities, etc...** |
| **Privacy Types** | **Public: Anyone can see the group, its members and their posts. Closed: Anyone can find the group and see who's in it. Only members can see posts**  **Secret: Only members can find the group and see posts.** |

# 

# Requirements

# Functional Requirements

**1. API allows adding functionality for signing up.**

**1.1 a social network user can sign up as premium user or normal**

**1.2 it takes email(mandatory) , password(mandatory) , username(mandatory) , profile picture and gender**

**3. API allows adding functionality for for adding post.**

**3.1 social network user can add posts on his profile page**

**3.2 social network user can add posts in groups that he is a member in**

**3.3 social network user can add posts on pages he created**

**4. API allows adding functionality for sending text messages to social network user’s friends only.**

**5. API allows adding functionality for viewing messages from friends**

**6. API allows adding functionality for signing in by email and password for social network users who has accounts**

**7. API allows adding functionality for signing out**

**8. API allows adding functionality for liking posts that are added by social network user friend’s or posts in groups he is a member in**

**9. API allows adding functionality for sharing posts (that are added by social network user friend’s or posts in groups he is a member in ) on his profile**

**10. API allows adding functionality for creating group**

**10.1 the groups could be public that any social network user can join it or private that users must be approved by**

**group creator user to join it**

**11. API allows adding functionality for creating pages**

**12. API allows adding functionality for assigning hashtags to posts**

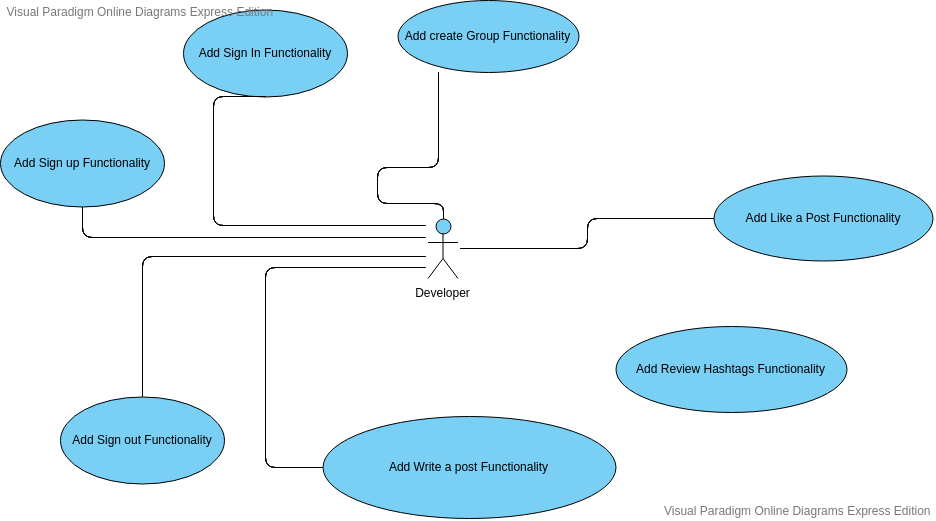
**13. API allows adding functionality for viewing posts that are assigned to a specific hashtag , and sorting them based on maximum number of likes**

|  |  |
| --- | --- |
| **Details** | **URPS** |
| → API's functionalities won't need more than four access steps  through to be reached in the API  → The API provides tips & guidelines for features allowed to add in social  network during it's creation guide for the developer | **Usability** |
| * Robustness:   → the API developer sends error message if not defined functionality was requested to be added in the social network   * Safety:   →The probability that API Servers for social networks created by the API  to work always without failing is 90% | **Reliability** |
| * Throughput:   → The API servers can respond to hundred request per min   * Availability:   → The API servers are available with probability 90% to provide requests'  response   * Response time:   → 90% from requests to API servers are processed in 10s  → max time for processing request won’t take more than 20s   * Scalability:   →in case requests exceeds max number of requests (100) the exceeding  requests are saved in a queue & processed after 50 of old requests are  processed at least | **Performance** |
| * Adaptability:   → the API can be changed to allow adding other features for the social  network that will be created by it   * Maintainability:   → the API can be modified to fix found errors occur in it's features of adding  functionalities to desired social network | **Supportability** |

**Non Functional Requirements**

# System Models

**Use Case Model**

****

**Use Case Tables**

|  |  |  |
| --- | --- | --- |
| Use Case ID: | #1 | |
| Use Case Name: | Add Sign Up functionality | |
| Actors: | the interface system that will use this API | |
| Pre-conditions: |  | |
| Post-conditions: | An account is created | |
| Flow of events: | **User Action** | **System Action** |
| 1- User enters his data |  |
|  | 2- System Verify user data |
| 3- User agree to the social network terms and conditions. |  |
|  | 4- System creates a new account to the user |
| Exceptions: | -The user enters an email that already exist.  -The API will send error message to the interface system | |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | #2 | |
| Use Case Name: | Add Sign in functionality | |
| Actors: | the interface system that will use this API | |
| Pre-conditions: | -User has an account | |
| Post-conditions: | User is logged in | |
| Flow of events: | **User Action** | **System Action** |
| 1- User enters his data |  |
|  | 2- System Verify user data |
| 3- User view his account |  |
| Exceptions: | -The user enters invalid data (wrong email/password)  -The API will send error message to the interface system | |
| Includes: | - | |
| Notes and Issues: | - | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | #3 | |
| Use Case Name: | Add Write a Post fucnctionality | |
| Actors: | the interface system that will use this API | |
| Pre-conditions: | - The user is logged in | |
| Post-conditions: |  | |
| Flow of events: | **User Action** | **System Action** |
| 1- User writes his post, and click on "post". |  |
|  | 2- System views the post on (his profile/ group he owns/ page he owns). |
| Exceptions: | -The user exceeds the number of letters of the post.  -The system won't allow posting it, and asks from the user to decrease the number of letters. | |
| Includes: | - | |
| Notes and Issues: | - | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | #4 | |
| Use Case Name: | Add Review Hashtags functionality | |
| Actors: | the interface system that will use this API | |
| Pre-conditions: | User is logged in | |
| Post-conditions: |  | |
| Flow of events: | **User Action** | **System Action** |
| 1- User enters the hashtag |  |

|  |  |  |
| --- | --- | --- |
|  |  | 2- System view all the posts with this hashtag, arranged with its importance. |
| Exceptions: | - | |
| Includes: | - | |
| Notes and Issues: | - | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | #5 | |
| Use Case Name: | Add Create a group functionality | |
| Actors: | the interface system that will use this API | |
| Pre-conditions: | -User logged in | |
| Post-conditions: | - A group is created | |
| Flow of events: | **User Action** | **System Action** |
| 1- User chooses the name of the group, type of privacy, and choose admins. |  |
|  | 2- System creates the new group with the chosen data. |
| Exceptions: | -The user doesn't add any members, when creation.  -The system asks the user to add at least one member, when creation. | |
| Includes: | - | |
| Notes and Issues: | - | |
| Use Case ID: | #6 | |
| Use Case Name: | Add Like a post Functionality | |
| Actors: | the interface system that will use this API | |
| Pre-conditions: | User is logged in / Like a post, is available to his friends' posts only | |
| Post-conditions: | Post is liked | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Flow of events: | **User Action** | **System Action** |  |
| 1- User likes some post |  |
|  | 2-Increases the number of likes by one. |
| Exceptions: | - | |  |
| Includes: | - | |  |
| Notes and Issues: | - | |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Use Case ID: | #7 | |  |
| Use Case Name: | Add Sign out Fuctionality | |
| Actors: | the interface system that will use this API | |
| Pre-conditions: | - The user is logged in | |
| Post-conditions: |  | |
| Flow of events: | **User Action** | **System Action** |
| 1- User signs out. |  |
|  | 2- System logs out the user from his profile. |
| Exceptions: |  | |  |
| Includes: | - | |  |
| Notes and Issues: | - | |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Use Case ID: | #8 | |  |
| Use Case Name: | Add Share post Functionality | |
| Actors: | the interface system that will use this API | |
| Pre-conditions: | -User is logged in / share a post, is available to his friends' posts only | |
| Post-conditions: | Post is shared | |
| Flow of events: | **User Action** | **System Action** |
| 1- User shares some post |  |
|  | 2-Increases the number of shares by one. |
| Exceptions: |  | |  |
| Includes: | - | |  |
| Notes and Issues: | - | |  |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | #9 | |
| Use Case Name: | Add Create a Page functionality | |
| Actors: | the interface system that will use this API | |
| Pre-conditions: | -User logged in | |
| Post-conditions: | - A page is created | |
| Flow of events: | **User Action** | **System Action** |
| 1- User chooses the name of the page, type of privacy, an |  |
|  | 2- System creates the new page with the chosen data. |
| Exceptions: | - | |
| Includes: | - | |
| Notes and Issues: | - | |

## **Class Diagram**

## 

# Traceability Matrix

# 

|  |  |  |
| --- | --- | --- |
| **BR#**  **( business/functional requirement number )** | **Use Case ID** | **Class(esID(s)** |
| # 1 | # 1 | #23,#24 |
| # 6 | # 2 | #23,#24 |
| # 3 | # 3 | #30,#31,#32 |
| # 13 | # 4 | #6 |
| # 10 | # 5 | #16,#17 |
| # 8 | # 6 | #23 |
| # 7 | # 7 | #23 |
| # 9 | # 8 | #23 |
| # 11 | # 9 | #2 |