

# Software Requirement Specification Document

## 1.Introduction

### 1.1 Purpose

This SRS describes the scope and system design of “Wazifa” which is created in sake of making it easier to connect with other people works the same as you work or working in the same job Wazifa will minimize the time and efforts in searching for companies or other workers.

### 1.2 Scope

“Wazifa” is a business and employment-oriented social media service that works as web application that allows users to make friends with each other and post anything about jobs or advice to other employees.

Users can sign-up and make their profile as their CV they will add their skills past jobs, current jobs, education .

They can comment, like or share someone's post and they can chat with any user.

Users can add a page about the company they work in with logo, description, position.

Users can search about other users and companies and discover their profile or page.

Users will be able to update their profile info (CV)

an administrator will use web-portal to administrate the system and keep the information accurate. The admin can verify the company page and manage its information.

## **1.3 Out Scope**

There is not job posting  
no bidding in jobs

Not commenting someone's post if he isn't friend.  
you can't login by fake mail or password.

## **1.4 overview**

“Wazifa” is built to make it easier to find other people works with you in the same company or works the same work as you are and make friends with them in order to learn from them and exchange your experiences.

# **2. General Description**

## **2.1 Product Functions**

This system helps you to connect with other people in your career to add more information and experience to you  
You also can search about friends and companies.

## **2.2 Similar System Information**

Our system is more similar to “LinkedIn”. it also can search about jobs in this application and make relation between people in your career like social media

## **2.3 User Characteristics**

We have three actors in our application including admin, visitor and member

Admin is the person who control the whole application by adding and removing posts, users and accounts

Visitor is the person who sign up to the application and can view

someone's profile  
member is the person who add friends, put posts, sending message,  
like, make comment and sharing the posts.

## **2.4 General Constraints**

System can't work without DB working,  
can't work locally,

## **3.Function Requirement**

### 1.visitor

Code		1.00
Name	Sign up	
Type	Functional Requirements	
Criticality	High	
Input	Name, Email, password	
Output	Registered to the system and Logged in	
Description	Sign up in the function that the user will register himself in the system to be able to sign in	
Priority	10/10	
Expected risks	Database registration failed	
Preconditions	He shouldn't have an account	
Post-conditions	The user is registered on the system	

<b>Dependencies</b>	
<b>Code</b>	<b>1.01</b>
<b>Name</b>	View user profile
<b>Type</b>	Functional requirement
<b>Criticality</b>	Low
<b>Input</b>	Profile name
<b>Output</b>	Profile
<b>Description</b>	User wants to see someone's profile and last posts or see his/her skills
<b>Priority</b>	5/10
<b>Expected risks</b>	Database crashes while getting information
<b>Preconditions</b>	User have account already
<b>Post-conditions</b>	The profile is viewed
<b>Dependencies</b>	2.00

<b>Code</b>	<b>1.02</b>
<b>Name</b>	View post
<b>Type</b>	Functional requirement
<b>Criticality</b>	Low
<b>input</b>	
<b>output</b>	Post
<b>Description</b>	User wants to see any post of any profile
<b>Priority</b>	5/10
<b>Expected risks</b>	Database crashes while getting information
<b>preconditions</b>	
<b>Post-conditions</b>	The post is viewed
<b>Dependencies</b>	2.00

2.member

<b>Code</b>	<b>2.00</b>
<b>Name</b>	Sign in
<b>Type</b>	Functional Requirement
<b>Criticality</b>	High
<b>Input</b>	Email, Password
<b>Output</b>	Successfully logged in.
<b>Description</b>	Logging in is the entrance to the site every user should sign in to be able to connect with other
<b>Priority</b>	10/10
<b>Expected risks</b>	Database crashes while getting information
<b>Preconditions</b>	He must have an account
<b>Post-conditions</b>	The user is logged in the system
<b>Dependencies</b>	1.00

<b>Code</b>	<b>2.01</b>
<b>Name</b>	Log out
<b>Type</b>	Functional requirement
<b>Criticality</b>	Low
<b>Input</b>	
<b>Output</b>	Get out of from his profile
<b>Description</b>	This function make prevent user from access to the system by his user unless he logged in again
<b>Priority</b>	5/10
<b>Expected risks</b>	Bug, the logout button not working
<b>Preconditions</b>	He must be logged in
<b>Post-conditions</b>	He will return to system home page
<b>Dependencies</b>	2.00

<b>Code</b>	<b>2.02</b>
<b>Name</b>	Search

<b>Type</b>	Functional requirement
<b>Criticality</b>	Low
<b>Input</b>	Text
<b>Output</b>	Profile has the same text
<b>Description</b>	The user types the text he wants to find and start search and we provide options user can search by the name or the career
<b>Priority</b>	5/10
<b>Expected risks</b>	The system doesn't respond
<b>Preconditions</b>	He must be logged in
<b>Post-conditions</b>	The result of search is given
<b>Dependencies</b>	2.00

<b>Code</b>	<b>2.03</b>
<b>Name</b>	Like post
<b>Type</b>	Functional Requirement
<b>Criticality</b>	Low
<b>input</b>	Post's id
<b>output</b>	You liked someone's post
<b>Description</b>	It is a way of interaction with the post.
<b>Priority</b>	5/10
<b>Expected risks</b>	Bug, the like button not working, database not saving likes on post
<b>preconditions</b>	He must be logged in
<b>Post-conditions</b>	The user will show that he post liked and see users who liked his post
<b>Dependencies</b>	2.00 , 1.02

<b>Code</b>	<b>2.04</b>
<b>Name</b>	Comment
<b>Type</b>	Functional Requirement
<b>Criticality</b>	Low
<b>input</b>	Post id and text
<b>output</b>	Your comment will be shown to you
<b>Description</b>	Is the way of interaction with post by adding your comment about it
<b>Priority</b>	5/10
<b>Expected risks</b>	Bug, the follow button not working
<b>preconditions</b>	He must be logged in and friend or follower with who put the post
<b>Post-conditions</b>	The comment will be shown to anyone to see and will be added to the post
<b>Dependencies</b>	2.00 , 1.02

<b>Code</b>	<b>2.05</b>
<b>Name</b>	Post
<b>Type</b>	Functional Requirement
<b>Criticality</b>	Low
<b>Input</b>	Any text or link or image or video
<b>Output</b>	Showing the input as a post.
<b>Description</b>	Making new post is the way people will type about anything they want to share with other users
<b>Priority</b>	5/10
<b>Expected risks</b>	Crashes on post, crash while saving in DB
<b>Preconditions</b>	He must be logged in
<b>Post-conditions</b>	The post will be published and will be shown to other people
<b>Dependencies</b>	2.00

<b>Code</b>	<b>2.06</b>
<b>Name</b>	Share
<b>Type</b>	Functional requirements
<b>Criticality</b>	Low
<b>input</b>	Post's id
<b>output</b>	Post has been shared
<b>Description</b>	User can share any post he wants to post it in his timeline
<b>Priority</b>	5/10
<b>Expected risks</b>	Bug, the share button not working
<b>preconditions</b>	Must be logged in
<b>Post-conditions</b>	Post has been published in user timeline
<b>Dependencies</b>	2.00 , 1,02



<b>Code</b>	<b>2.07</b>
<b>Name</b>	Message
<b>Type</b>	Functional Requirement
<b>Criticality</b>	Low
<b>Input</b>	Text
<b>Output</b>	Message is sent
<b>Description</b>	User choose the user he wants to contact with him then type the message he wants
<b>Priority</b>	5/10
<b>Expected risks</b>	Database crashes while getting information
<b>Preconditions</b>	He must be logged in
<b>Post-conditions</b>	Message sent successfully
<b>Dependencies</b>	2.00,1.01

<b>Code</b>	<b>2.08</b>
<b>Name</b>	View Messages
<b>Type</b>	Functional Requirement
<b>Criticality</b>	Low
<b>Input</b>	
<b>Output</b>	Messages is viewed
<b>Description</b>	Able user to see his message in his inbox
<b>Priority</b>	5/10
<b>Expected risks</b>	User don't have any messages
<b>Preconditions</b>	He must be logged in
<b>Post-conditions</b>	User showed his messages successfully
<b>Dependencies</b>	2.00

<b>Code</b>	<b>2.09</b>
<b>Name</b>	Follow

<b>Type</b>	Functional Requirement
<b>Criticality</b>	Low
<b>Input</b>	Profile name
<b>Output</b>	Successfully followed a person
<b>Description</b>	Following someone will let you able to view his posts and interact with them by comment or like
<b>Priority</b>	5/10
<b>Expected risks</b>	Bug, the follow button not working
<b>Preconditions</b>	He must be logged in
<b>Post-conditions</b>	The followed user will be added to follow list and his posts will be shown to the follower
<b>Dependencies</b>	2.00 , 1.01

<b>Code</b>	<b>2.10</b>
<b>Name</b>	View followers
<b>Type</b>	Functional requirement
<b>Criticality</b>	Low
<b>Input</b>	
<b>Output</b>	List of user's follower
<b>Description</b>	User can see the people who follow him
<b>Priority</b>	5/10
<b>Expected risks</b>	Database crashes while getting information
<b>Preconditions</b>	He must be logged in
<b>Post-conditions</b>	User will see list of followers
<b>Dependencies</b>	2.00 , 2.22

<b>Code</b>	<b>2.11</b>
<b>Name</b>	View following list
<b>Type</b>	Functional requirement
<b>Criticality</b>	Low

<b>Input</b>	
<b>Output</b>	List of user's following
<b>Description</b>	User can see the people he followed to contact easy with them or see their last posts
<b>Priority</b>	5/10
<b>Expected risks</b>	Database crashes while getting information
<b>Preconditions</b>	He must be logged in
<b>Post-conditions</b>	User will see list of his followed profiles
<b>Dependencies</b>	2.00 , 2.22

<b>Code</b>	<b>2.12</b>
<b>Name</b>	Add Friend
<b>Type</b>	Functional Requirement
<b>Criticality</b>	Low
<b>Input</b>	Profile name
<b>Output</b>	send a request to person
<b>Description</b>	Adding someone make you contact with him easy , see his posts and comment on it .
<b>Priority</b>	5/10
<b>Expected risks</b>	Bug, the add Friend button not working
<b>Preconditions</b>	He must be logged in
<b>Post-conditions</b>	The added user will be receive add request from the user
<b>Dependencies</b>	2.00 , 1.01

<b>Code</b>	<b>2.13</b>
<b>Name</b>	Accept friend
<b>Type</b>	Functional Requirement
<b>Criticality</b>	Low
<b>Input</b>	Profile
<b>Output</b>	Successfully added a person

<b>Description</b>	Accept add request means add someone to his friend list
<b>Priority</b>	5/10
<b>Expected risks</b>	Bug, the accept Friend button not working
<b>Preconditions</b>	He must be logged in
<b>Post-conditions</b>	The added user will be added to friend list and his posts will be shown to the friends
<b>Dependencies</b>	2.00,2.20

<b>Code</b>	<b>2.14</b>
<b>Name</b>	View Friends list
<b>Type</b>	Functional Requirement
<b>Criticality</b>	Low
<b>Input</b>	
<b>Output</b>	List of user's Friends
<b>Description</b>	This function provide you easy way to see the list of your friends
<b>Priority</b>	5/10
<b>Expected risks</b>	Database crashes while getting information
<b>Preconditions</b>	He must be logged in
<b>Post-conditions</b>	You can see all your friends
<b>Dependencies</b>	2.00 , 2.22

<b>Code</b>	<b>2.15</b>
<b>Name</b>	Update profile
<b>Type</b>	Functional requirements
<b>Criticality</b>	Low
<b>input</b>	Name or password o skills
<b>output</b>	Profile has been updated
<b>Description</b>	User can update all of his information like name ,password, age or update his career and experiance
<b>Priority</b>	5/10
<b>Expected risks</b>	Bug, the update button not working
<b>preconditions</b>	Must be logged in
<b>Post-conditions</b>	Profile has been updated
<b>Dependencies</b>	2.00 ,2.22

<b>Code</b>	<b>2.16</b>
<b>Name</b>	View notification
<b>Type</b>	Functional requirements
<b>Criticality</b>	Low
<b>input</b>	
<b>output</b>	List of notification
<b>Description</b>	By this function user can knew how many member accepts his add request, like ,comment in his post or share it
<b>Priority</b>	5/10
<b>Expected risks</b>	Database crashes while getting information
<b>preconditions</b>	Must be logged in
<b>Post-conditions</b>	Notification has been showed
<b>Dependencies</b>	2.00

<b>Code</b>	<b>2.17</b>
<b>Name</b>	Retrieve Password
<b>Type</b>	Performance requirements
<b>Criticality</b>	Low
<b>input</b>	Email
<b>output</b>	Verification code
<b>Description</b>	If user forget his password he can retrieve it again
<b>Priority</b>	5/10
<b>Expected risks</b>	Database crashes while getting information
<b>preconditions</b>	Must have an account
<b>Post-conditions</b>	Password has been retrieved
<b>Dependencies</b>	1.00

<b>Code</b>	<b>2.18</b>
<b>Name</b>	Upload photo
<b>Type</b>	Functional requirements
<b>Criticality</b>	Low
<b>input</b>	Photo
<b>output</b>	Photo
<b>Description</b>	User can upload a photo to his profile easy
<b>Priority</b>	5/10
<b>Expected risks</b>	Bug, the upload button not working
<b>preconditions</b>	Must be logged in and in his profile
<b>Post-conditions</b>	User's photo has been uploaded
<b>Dependencies</b>	2.00 , 2.22

<b>Code</b>	<b>2.19</b>
-------------	-------------

<b>Name</b>	Show likes
<b>Type</b>	Functional requirements
<b>Criticality</b>	Low
<b>input</b>	Post
<b>output</b>	User who like the post
<b>Description</b>	User can see the member who like any post
<b>Priority</b>	5/10
<b>Expected risks</b>	Database crashes while getting information
<b>preconditions</b>	Must be logged in and viewed post
<b>Post-conditions</b>	User knew the member who like the post
<b>Dependencies</b>	2.00 , 1.02

<b>Code</b>	<b>2.20</b>
<b>Name</b>	Show comments
<b>Type</b>	Functional requirements
<b>Criticality</b>	Low
<b>input</b>	Post
<b>output</b>	User who comment in the post
<b>Description</b>	User can see the comments in any post
<b>Priority</b>	5/10
<b>Expected risks</b>	Database crashes while getting information
<b>preconditions</b>	Must be logged in and viewed post
<b>Post-conditions</b>	User see the comments in any post
<b>Dependencies</b>	2.00 , 1.02

<b>Code</b>	<b>2.21</b>
-------------	-------------

<b>Name</b>	Show shares
<b>Type</b>	Functional requirements
<b>Criticality</b>	Low
<b>input</b>	Post
<b>output</b>	User who share the post
<b>Description</b>	User can see the member who share any post
<b>Priority</b>	5/10
<b>Expected risks</b>	Database crashes while getting information
<b>preconditions</b>	Must be logged in and viewed post
<b>Post-conditions</b>	User knew the member who share the post
<b>Dependencies</b>	2.00 , 1.02

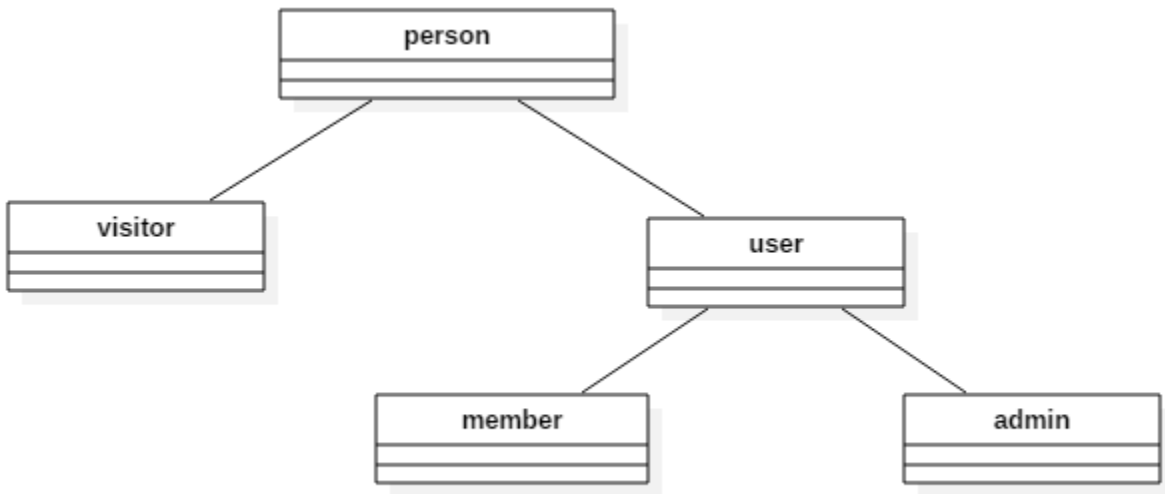
<b>Code</b>	<b>2.22</b>
<b>Name</b>	View own profile
<b>Type</b>	Functional Requirement
<b>Criticality</b>	low
<b>Input</b>	
<b>Output</b>	User profile open
<b>Description</b>	user can view his profile to edit in it ,remove things or add things in it
<b>Priority</b>	5/10
<b>Expected risks</b>	Database crashes while getting information
<b>Preconditions</b>	He must have an account
<b>Post-conditions</b>	The profile page opened
<b>Dependencies</b>	1.00

<b>Code</b>	<b>2.23</b>
-------------	-------------



<b>Name</b>	View friend request
<b>Type</b>	Functional Requirement
<b>Criticality</b>	low
<b>Input</b>	Friend request IDs
<b>Output</b>	Names of friend requests
<b>Description</b>	It's the place where the person is able to view people who send him add request with the ability to add them
<b>Priority</b>	5/10
<b>Expected risks</b>	Database crashes while getting information
<b>Preconditions</b>	He must have an account
<b>Post-conditions</b>	All requests data will be shown
<b>Dependencies</b>	2.00

3.



## Admin

<b>Code</b>	<b>3.00</b>
<b>Name</b>	Remove comment
<b>Type</b>	performance Requirement
<b>Criticality</b>	Low
<b>Input</b>	Comment's id
<b>Output</b>	Your comment successfully deleted
<b>Description</b>	It is an option to delete the comment you typed by mistake or you don't want anyone to see it
<b>Priority</b>	5/10
<b>Expected risks</b>	Bug, the comment not deleted from the database
<b>Preconditions</b>	He must be logged in, the comment must be typed
<b>Post-conditions</b>	The comment will be deleted
<b>Dependencies</b>	2.00 , 2.17

<b>Code</b>	<b>3.01</b>
<b>Name</b>	Remove user
<b>Type</b>	preformance requirement
<b>Criticality</b>	Low

<b>input</b>	Bad user's profile name
<b>output</b>	User removed
<b>Description</b>	Admin can remove bad user
<b>Priority</b>	5/10
<b>Expected risks</b>	Bug, the user not deleted from the database
<b>preconditions</b>	Must be logged in and user exists
<b>Post-conditions</b>	User will be removed
<b>Dependencies</b>	2.00 , 1.01

<b>Code</b>	<b>3.02</b>
<b>Name</b>	Remove post
<b>Type</b>	performance requirement
<b>Criticality</b>	Low
<b>input</b>	Bad Post's id
<b>output</b>	Post removed
<b>Description</b>	User also can remove bad posts which include bad posts
<b>Priority</b>	5/10
<b>Expected risks</b>	Bug, the post not deleted from the database
<b>preconditions</b>	Must be logged in and page exists
<b>Post-conditions</b>	Page will be removed
<b>Dependencies</b>	2.00 , 1.01

<b>Code</b>	<b>3.03</b>
<b>Name</b>	Get Report
<b>Type</b>	Performance Requirements
<b>Criticality</b>	Low
<b>input</b>	Report's id
<b>output</b>	Report
<b>Description</b>	Admin can view a Report about the site user's number

<b>Priority</b>	5/10
<b>Expected risks</b>	Database crashes while getting information
<b>preconditions</b>	Must be logged in
<b>Post-conditions</b>	Report is viewed successfully
<b>Dependencies</b>	2.00

## 4-Other non-functional attributes

code	4.1
name	System dependability
type	performance requirement
description	The fault tolerance of the system ,if the system loses the connection to the internet or the user gets some strange input ,the user should be informed
Priority	10/10

code	4.2
name	System security
type	Security requirement
description	Security of the communication between the system and the server ,message should be encrypted for login communication so other can't see user password
Priority	10/10

code	4.3
name	User create account security
type	Security requirement

description	If the user wants to create an account and the desired name is occupied ,the user should be asked to choose a different user name
Priority	10/10

code	4.4
name	Admin login account security
type	Security requirement
description	If an admin tries to log in to the web portal with a non-existing account then the admin should not be logged in, the admin should be notified of the log in failure
Priority	10/10

code	4.5
name	System reliability
type	Performance requirement
description	The reliability of the system that it gives that it gives the right result on a search
Priority	10/10

code	4.6
name	System availability
type	performance requirement
description	The availability of the system when it is used (not considering network failing)
Priority	10/10

code	4.7
name	Response time
type	usability requirement
description	The fastness of the search the response time of a search
Priority	10/10

## 5-Operational Scenarios

### USE CASE

#### ACTOR

#### PRE-CONDITION

### REGISTER

Visitor

He shouldn't have an account

<b>BASIC FLOW</b>	First the user click on register button then the system will show a register form he must fill in the data then click submit then the system will show him his profile and then the user must enter his skills and jobs and his education and complete registration then submit his changes
<b>ALTERNATIVE FLOW</b>	The system will show a message register not completed
<b>POST CONDITION</b>	The user is registered on the system
<b>USE CASE</b>	<b>LOGIN</b>
<b>ACTOR</b>	Member,admin
<b>PRE-CONDITION</b>	He must have an account
<b>BASIC FLOW</b>	First the user click on login button then the system will show a login form he must fill in the user name and the password click login
<b>ALTERNATIVE FLOW</b>	The system will show a message no account and then will direct him to register form
<b>POST CONDITION</b>	The user is logged in the system
<b>USE CASE</b>	<b>ADD POST</b>
<b>ACTOR</b>	Member
<b>PRE-CONDITION</b>	He must be logged in
<b>BASIC FLOW</b>	The user will click on new post button then he will write his post then click post button
<b>ALTERNATIVE FLOW</b>	The system will show him try again message
<b>POST CONDITION</b>	The post is shown in his timeline
<b>USE CASE</b>	<b>FOLLOW</b>
<b>ACTOR</b>	Member
<b>PRE-CONDITION</b>	He must have an account and be logged in
<b>BASIC FLOW</b>	First the user click on follow button
<b>ALTERNATIVE FLOW</b>	The followed person won't be added on the followed list
<b>POST CONDITION</b>	system will add the followed person on the follower list of the user

USE CASE	ADD FRIEND
ACTOR	Member
PRE-CONDITION	He must have an account and be logged in
BASIC FLOW	First the user click on add friend button
ALTERNATIVE FLOW	The chosen friend won't be added on the friends list of the user and the system will show him try again message
POST CONDITION	The user chosen will be added in the friends list of the user

USE CASE	VIEW FRIENDS
ACTOR	Member
PRE-CONDITION	He must have an account and be logged in
BASIC FLOW	the user click on view friends button and the system will respond
ALTERNATIVE FLOW	The system will show him no friends message
POST CONDITION	system will show the friends list

USE CASE	VIEW FOLLOWING
ACTOR	Member
PRE-CONDITION	He must have an account and be logged in
BASIC FLOW	First the user click on view following button
ALTERNATIVE FLOW	The system will show him no following users
POST CONDITION	system will show the following list

USE CASE	LOGOUT
ACTOR	Member,admin
PRE-CONDITION	He must have an account and be logged in
BASIC FLOW	First the user click on logout button
ALTERNATIVE FLOW	The system will not logout
POST CONDITION	The user must be logged out

USE CASE	VIEW POST
ACTOR	Member, admin, visitor
PRE-CONDITION	
BASIC FLOW	the user click on view profile button
ALTERNATIVE FLOW	The system may crashes and doesn't response to the request so he will stay in the same page
POST CONDITION	system will show the profile of the user

USE CASE	VIEW MESSAGES
ACTOR	member
PRE-CONDITION	He must have an account and be logged in
BASIC FLOW	First the user click on view messages button
ALTERNATIVE FLOW	The system will show him no messages
POST CONDITION	system will show the user his messages

USE CASE	SEND MESSAGE
ACTOR	member
PRE-CONDITION	He must have an account and be logged in
BASIC FLOW	First the user must choose the person he wants to contact with then click on message button then type his message then click send button
ALTERNATIVE FLOW	The system will not send the message
POST CONDITION	The message must be sent to the selected user

USE CASE	SEARCH
ACTOR	Member, admin
PRE-CONDITION	He must have an account and be logged in
BASIC FLOW	First the user type the thing he want to fiend in the search bar then click search button
ALTERNATIVE FLOW	The system will show him not found or will show him similar things
POST CONDITION	system will view result

USE CASE	VIEW PROFILE
ACTOR	Member, admin, visitor
PRE-CONDITION	
BASIC FLOW	the user click on view profile button
ALTERNATIVE FLOW	The system may crashes and doesn't response to the request so he will stay in the same page
POST CONDITION	system will show the profile of the user

USE CASE	LIKE POST
ACTOR	member
PRE-CONDITION	He must have an account and be logged in
BASIC FLOW	First the user click on view like button
ALTERNATIVE FLOW	Like will not be added on the post
POST CONDITION	A like must be added to the post

USE CASE	COMMENT
ACTOR	member



<b>PRE-CONDITION</b>	He must have an account and be logged in
<b>BASIC FLOW</b>	First the user will choose the post then click add comment then type his comment then click ok
<b>ALTERNATIVE FLOW</b>	The system will not add the comment to the post
<b>POST CONDITION</b>	system will show the comment to the post

<b>USE CASE</b>	<b>REMOVE COMMENT</b>
<b>ACTOR</b>	Admin,member
<b>PRE-CONDITION</b>	He must have an admin of the system and be logged in
<b>BASIC FLOW</b>	The admin will click on X button in the top of comment
<b>ALTERNATIVE FLOW</b>	System shows message “comment not deleted”
<b>POST CONDITION</b>	The comment will be removed

<b>USE CASE</b>	<b>REMOVE MEMBER</b>
<b>ACTOR</b>	Admin
<b>PRE-CONDITION</b>	Must be logged in and member exists
<b>BASIC FLOW</b>	First the user click on view friends button
<b>ALTERNATIVE FLOW</b>	The system will show him no friends message
<b>POST CONDITION</b>	User will be removed

<b>USE CASE</b>	<b>SHOW INFO</b>
<b>ACTOR</b>	member
<b>PRE-CONDITION</b>	Must be logged in and open his profile
<b>BASIC FLOW</b>	The user will click on view profile button then he will click on view info
<b>ALTERNATIVE FLOW</b>	His personal info will not viewed
<b>POST CONDITION</b>	The member info will be shown in the current page

<b>USE CASE</b>	<b>REMOVE LIKE</b>
<b>ACTOR</b>	member
<b>PRE-CONDITION</b>	Must be logged
<b>BASIC FLOW</b>	Go to home then click on add page then add page info

<b>ALTERNATIVE FLOW</b>	The system will show him a message “page added successfully ”
<b>POST CONDITION</b>	system will add the page to his account

<b>USE CASE</b>	<b>SHARE</b>
<b>ACTOR</b>	member
<b>PRE- CONDITION</b>	He must have an account and be logged in
<b>BASIC FLOW</b>	First the member click on share button on the post he wants to share
<b>ALTERNATIVE FLOW</b>	The content won’t be shared in the user profile
<b>POST CONDITION</b>	The content will be shared successfully

<b>USE CASE</b>	<b>RETRIEVE PASSWORD</b>
<b>ACTOR</b>	member
<b>PRE -CONDITION</b>	He must have an account
<b>BASIC FLOW</b>	First the user will try to log in if he failed he will click on forget password
<b>ALTERNATIVE FLOW</b>	The system will crashes
<b>POST CONDITION</b>	system will sent him a confirmation message on his mail

<b>USE CASE</b>	<b>UPDATE PROFILE</b>
<b>ACTOR</b>	member
<b>PRE -CONDITION</b>	He must have an account and be logged in
<b>BASIC FLOW</b>	First the user click on view profile button then click on update profile button
<b>ALTERNATIVE FLOW</b>	The system show the user an error message
<b>POST CONDITION</b>	The profile must be updated successfully

<b>USE CASE</b>	<b>ACCEPT FRIEND</b>
<b>ACTOR</b>	member
<b>PRE -CONDITION</b>	He must have an account and be logged in and he must get an accept friend notification
<b>BASIC FLOW</b>	The user will click on accept button
<b>ALTERNATIVE FLOW</b>	The system won’t add the user in his friends list the request won’t be dismissed from the notification list
<b>POST CONDITION</b>	The accepted user will be added in his friends list

USE CASE	VIEW NOTIFICATION
ACTOR	member
PRE -CONDITION	He must have an account and be logged in and he must get a notification alert
BASIC FLOW	First the user click on view notification button then list will be shown
ALTERNATIVE FLOW	The system won't response to his request
POST CONDITION	The notification list will be shown to the user

USE CASE	UPLOAD PHOTO
ACTOR	member
PRE -CONDITION	He must have an account and be logged in
BASIC FLOW	First the user click on view profile button and must view his profile and then update profile then click upload photo button
ALTERNATIVE FLOW	The photo won't be uploaded
POST CONDITION	The photo will be added

USE CASE	SHOW LIKES
ACTOR	Member, admin
PRE -CONDITION	He must have an account and be logged in
BASIC FLOW	First the user click on view post then he will click on view likes button
ALTERNATIVE FLOW	The likes number will not appear
POST CONDITION	The likes number will appear

USE CASE	SHOW COMMENTS
ACTOR	Member, admin
PRE -CONDITION	He must have an account and be logged in
BASIC FLOW	First the user click on view post then he will click on view comments button
ALTERNATIVE FLOW	The comments will not appear
POST CONDITION	The comments will appear

USE CASE	SHOW SHARES
ACTOR	Member, admin
PRE -CONDITION	He must have an account and be logged in
BASIC FLOW	First the user click on view post then he will click on view shares button
ALTERNATIVE FLOW	The shares number will not appear
POST CONDITION	The shares number will appear

USE CASE	SHOW FOLLOWING
ACTOR	Member, admin
PRE -CONDITION	He must have an account and be logged in

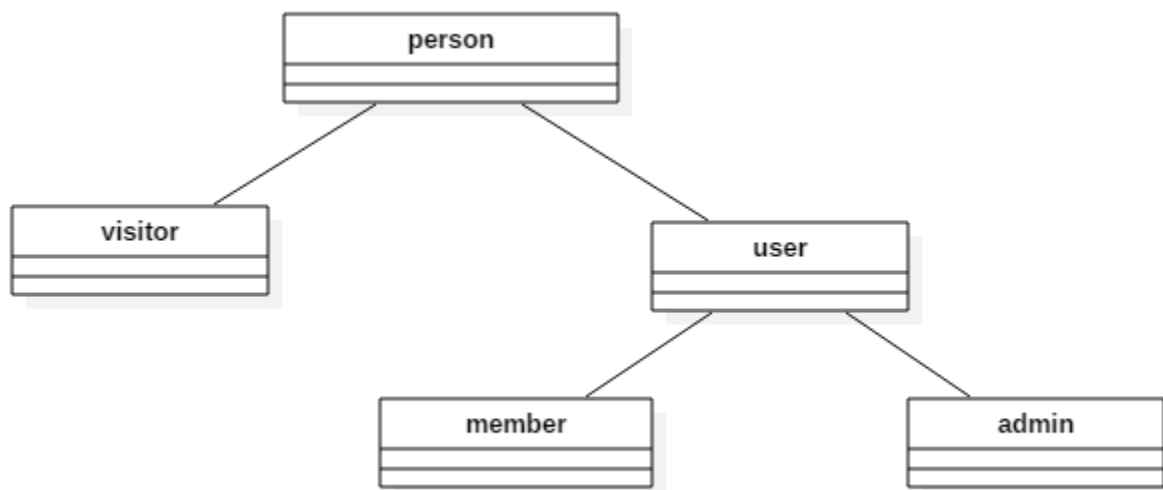
<b>BASIC FLOW</b>	First the user click on view following list
<b>ALTERNATIVE FLOW</b>	
<b>POST CONDITION</b>	The following list will not appear

<b>USE CASE</b>	<b>SHOW OWN PROFILE</b>
<b>ACTOR</b>	Member
<b>PRE -CONDITION</b>	He must have an account and be logged in
<b>BASIC FLOW</b>	the user click on button “profile”
<b>ALTERNATIVE FLOW</b>	
<b>POST CONDITION</b>	The profile page will appear

<b>USE CASE</b>	<b>SHOW FRIEND REQUEST</b>
<b>ACTOR</b>	Member
<b>PRE -CONDITION</b>	He must have an account and be logged in
<b>BASIC FLOW</b>	the user click on friends request button
<b>ALTERNATIVE FLOW</b>	No friend requests.
<b>POST CONDITION</b>	Friend requests will appear

## 6- Preliminary Object-Oriented Domain Analysis

### 6.1 Inheritance Relationships



### 6.2 List of Super classes:

Person class  
user class

## **6.2 List of Sub classes:**

User class  
visitor class  
member class  
admin class

## **6.3 Class name**

1. Class Person
2. Class Post
3. Class User
4. Class Profile
5. Class Member
6. Class Friend
7. Class Admin
8. Class Validation
9. Class Visitor
10. Class Report
11. Class Data Base

## **6.4 Purpose:**

1. Class person  
Include two function view profile and view posts that is help sub class that inherit from it to view their profile and posts
2. Class user  
It help to represent the information of the user like name, email and password  
It can also make the user log in, search, add or remove posts and log out too
3. Class visitor  
To help the new visitor to registration for the first time
4. Class Admin  
It control the application by adding or removing posts and accounts
5. Class member  
It help the members of the application to like, comment, share, adding or removing posts  
And also following, adding friends and make communication with other friends
6. Class message  
To make communication more easy to our members they can also send and received messages from other friends and followers
7. Class friend  
To view friends list and adding or removing friends easily
8. Class post  
To help members to like comment and sharing posts easily

#### 9. Class report

To send reports to the admin about whole the application for more control to the application

#### 10. Class validation

To handle all of errors it can happens in the application to ignore errors and make the application more safety

#### 11. Class Data Base

To storage and save all of data in the application for access and return the data at any time you need

## 6.5 Attributes

Class name	Attribute
User	User name / email / password
Profile	Jobs /interest / education / skills
Member	Age /profile / posts
Post	Like / comment / share
Report	Date / content