(Team Name) Team IdeoLogic

**(Project Title)** Scribble (Social Media Platform)

Software Design Document

Name (s):

Muhammad Hammad Asif(161109)

Hamza Aslam(161112)

Bilal Fareed(161135)

Ahmad Razi(161117)

Date: (03/05/2019)

**TABLE OF CONTENTS**

[1. INTRODUCTION 1](#_Toc4987)

[1.1 Purpose 1](#_Toc4988)

[1.2 Scope 1](#_Toc4989)

[1.3 Overview 2](#_Toc4990)

[1.4 Reference Material 2](#_Toc4991)

[1.5 Definitions and Acronyms 2](#_Toc4992)

[2. SYSTEM OVERVIEW 2](#_Toc4993)

[3. SYSTEM ARCHITECTURE 3](#_Toc4994)

[3.1 Architectural Design 3](#_Toc4995)

[3.2 Decomposition Description 3](#_Toc4996)

[3.3 Design Rationale 4](#_Toc4997)

[4. DATA DESIGN 4](#_Toc4998)

[4.1 Data Description 4](#_Toc4999)

[4.2 Data Dictionary 4](#_Toc5000)

[5. COMPONENT DESIGN 5](#_Toc5001)

[6. HUMAN INTERFACE DESIGN 12](#_Toc5002)

[6.1 Overview of User Interface 15](#_Toc5003)

[6.2 Screen Images 15](#_Toc5004)

[6.3 Screen Objects and Actions 15](#_Toc5005)

[7. REQUIREMENTS MATRIX 16](#_Toc5006)

[8. APPENDICES 16](#_Toc5007)

# INTRODUCTION

## Purpose

This design document is intended to give the overview of implementation of Life scribble at a high level. It also identifies the framework and technologies used for the development and tries to define the system architecture. This document will also be used for identifying contradictions, if any, prior to coding phase. The main goal of this document is to make design level information easily understandable

## Scope

This design document is meant to provide an overview of the structure of the system. This document also includes the database architecture of Life scribble along with database diagram for reference. This document also serves as a mandate for the design standards, data structures and design patterns to be implemented. UML diagrams are included to show how they different components interact with each other.

## Overview

The remaining document has 5 more sections. The second section gives general description of the project. Section 3 gives architectural view of the system. Section 4 consists of the data model. Section 5 gives the consideration of any risks if occurred and their mitigations. Section 6 is a list of appendix that helps better understand the document.

## Reference Material

## Definitions and Acronyms

The following table explains terms and acronyms specific to this SRS.

|  |  |
| --- | --- |
| ***Term/Acronym*** | ***Description/Definition*** |
| ***Life scribble*** | *Life scribble is a social networking website.* |
| ***UC*** | *Use Case* |
| ***Scribble*** | *Scribble is a post shared by the user on his slate.* |
| ***ChitChat*** | *Chatting activity between two users registered on Life scribble.* |

# System Overview

1) Social Investment Rewards Platform

2) There’s a feed page / User Profile page with profile picture and cash wallet/option to donate and invest

3) Every scribble will appear on the feed page / Every Re-Scribble

4) There’s a Favorite / Re-Scribble Button / Share / Tip button

5) Tip button is used to tip other user’s scribble using real cash

6) Favorite button is to actually keep the scribbles that are favorited

7) Re-Scribble button is used to re-scribble the other user’s scribble

8) Share button is to share the scribble to other platforms (Facebook/Whatsapp)

9) Each time a user tips he will earn a certain amount of GOLD 2.0 / G2.0 a crypto coin which then the user can accumulate to exchange for physical goods ( Air Jordan / Iphone XR)

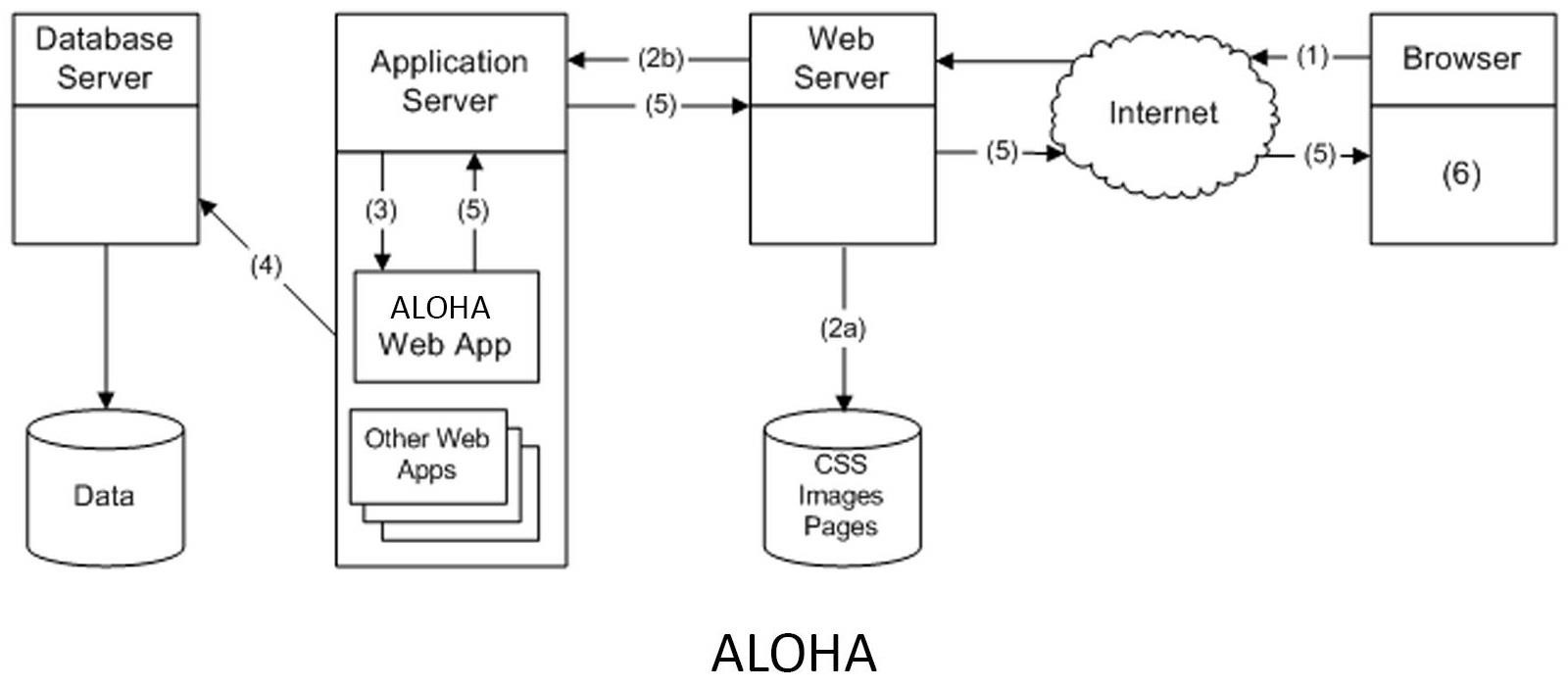
10) GOLD 2.0 serves as a rewarding point to exchange goods

11) Invest button will be fintech app creating a customized portfolio of shares/properties/currency/blue chip stocks

# SYSTEM ARCHITECTURE

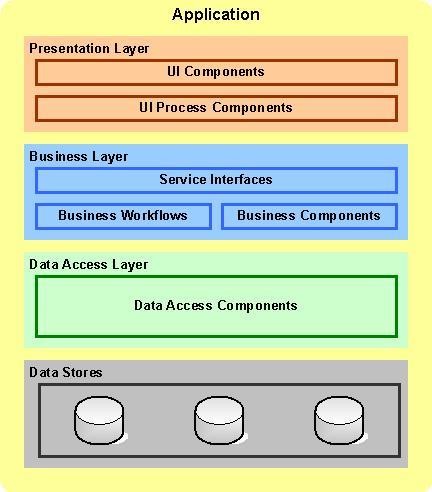
## Decomposition Description

## Top level architecture

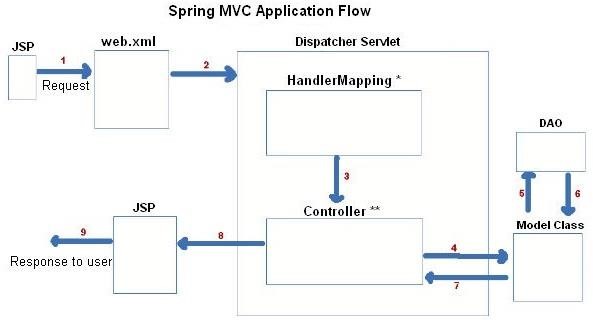


The above diagram gives a block view of Life scribble system showing the interaction of different modules with each other. The numbered arrows show the flow of control between these modules.

## Application Architecture



### **Web Application Architecture**



## 3.2 Design Rationale

Spring MVC framework will be used for developing the Life scribble web application. It divides the application into three interconnected parts. Thus the internal representations of the information are kept separate from the information that is presented to the end-user or accepted from end-user.

1. Presentation Layer

The presentation layer of the application will be JSP pages displayed to the user. The JSP pages will be both to display information and to take inputs from user.

1. Controller Layer

The controller initiates the commands to the model to make changes to database. It also sends commands to the presentation layer views associated to the models.

1. Data Access Layer

The data access layer is used to access the database and make changes to the data.

1. **Data Design**

**4.1 Data Description**

MySQL database and JDBC to communicate with the database that is installed locally on the server.

**4.1 Data Dictionary**

**Table 1.** Data Dictionary

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | **Field** | | **Type** | | **Null** | | **Default** | |
| **approved\_by** | | Request\_IDi | | int(11) | | Yes | | *NULL* | |
|  | | Person\_ID | | int(11) | | Yes | | *NULL* | |
|  | | Approval\_level | | enum('Level1','Level2', | | No | | Level1 | |
|  | |  | | 'Level3') | |  | |  | |
|  | | Approved\_Date | | timestamp | | Yes | | CURRENT\_TIMESTAMP | |
|  | |  | |  | |  | |  | |
| **assets** | | Asset\_ID | | int(11) | | No | |  | |
|  | | BarcodeNum | | varchar(50) | | Yes | | *NULL* | |
|  | | SerialNum | | varchar(50) | | Yes | | *NULL* | |
|  | | Location\_ID | | int(11) | | No | |  | |
|  | | Type | | enum('Desk','Computer','Acad | | No | |  | |
|  | |  | | emic\_stuff','Other','Mouse','K | |  | |  | |
|  | |  | | eyboard','Printer','Monitor', | |  | |  | |
|  | |  | | 'Table','Chair','Projector','Soft | |  | |  | |
|  | |  | | ware') | |  | |  | |
|  | | Description | |  | | Yes | | *NULL* | |
|  | |  | |  | | s | |  | |
|  | | Status | | enum('available','broken','not | | No | | available | |
|  | |  | | \_available') | |  | |  | |
|  | | Color | | varchar(250) | | Yes | | *NULL* | |
|  | | Material | | varchar(250) | | Yes | | *NULL* | |
|  | | Brand | | varchar(250) | | Yes | | *NULL* | |
|  | | Host | | varchar(250) | | Yes | | *NULL* | |
|  | | Created\_date | | timestamp | | No | | CURRENT\_TIMESTAMP | |
|  | | PurchaseNo | | varchar(250) | | Yes | | not set | |
|  | | RequestNo | | varchar(250) | | Yes | | not set | |
|  | |  | |  | |  | |  | |
| **assets\_group** | | Asset\_master\_ID | | int(11) | | Yes | | *NULL* | |
|  | | Asset\_child\_ID | | int(11) | | Yes | | *NULL* | |
|  | | Type | | enum('Group','Work\_station',' | | No | | group | |
|  | |  | | Office\_equipment') | |  | |  | |
|  | |  | |  | |  | |  | |
| **assets\_history** | | Asset\_ID | | int(11) | | No | | *NULL* | |
|  | | BarcodeNum | | varchar(50) | | Yes | | *NULL* | |
|  | | SerialNum | | varchar(50) | | Yes | | *NULL* | |
|  | | Location\_ID | | int(11) | | Yes | | *NULL* | |
|  | | Type | | varchar(50) | | Yes | | *NULL* | |
|  | | Description | | varchar(2000) | | Yes | | *NULL* | |
|  | | Status | | varchar(50) | | Yes | | *NULL* | |
|  | | Color | | varchar(250) | | Yes | | *NULL* | |
|  | | Material | | varchar(250) | | Yes | | *NULL* | |
|  | | Brand | | varchar(250) | | Yes | | *NULL* | |
|  | | Host | | varchar(250) | | Yes | | *NULL* | |
|  | | Modified\_by | | varchar(250) | | Yes | | *NULL* | |
|  | | PurchaseNo | | varchar(250) | | Yes | | Not set | |
|  | | RequestNo | | varchar(250) | | Yes | | Not set | |
|  | |  | |  | |  | |  | |
| **batch\_request** | | Request\_ID | | int(11) | | Yes | | *NULL* | |
|  | | Asset\_ID | | int(11) | | Yes | | *NULL* | |
|  | | Type | | enum('Move','Buy','Repair','D | | No | | Repair | |
|  | |  | | elete') | |  | |  | |
|  | |  | |  | |  | |  | |
| **building** | | Building\_ID | | int(11) | | No | | *NULL* | |
|  | | Address | | varchar(250) | | Yes | | *NULL* | |
|  | | Name | | varchar(250) | | Yes | |  | |
|  | | Type | | enum('Big','Medium', 'Small') | | No | | Medium | |
|  | | FloorNum | | int(3) | | Yes | | 1 | |
|  | |  | |  | |  | |  | |
| **fac\_dep** | | Fac\_dep\_ID | | int(11) | | No | |  | |
|  | | Building | | int(11) | | Yes | | *NULL* | |
|  | | Name | | varchar(250) | | No | |  | |
|  | | Type | | enum('Faculty','Department') | | No | | Department | |
|  | | Belong\_to | | int(11) | | Yes | | *NULL* | |
|  | |  | |  | |  | |  | |
| **full\_person\_info** | | Person\_ID | | int(11) | | Yes | | 0 | |
|  | | FirstName | | varchar(250) | | Yes | | *NULL* | |
|  | | LastName | | varchar(250) | | Yes | | *NULL* | |
|  | | UserName | | varchar(50) | | Yes | | *NULL* | |
|  | | Password | | varchar(50) | | Yes | | *NULL* | |
|  | | Address | | varchar(250) | | Yes | | *NULL* | |
|  | | EmailAddress | | varchar(50) | | Yes | | *NULL* | |
|  | | MobileNumber | | varchar(50) | | Yes | | *NULL* | |
|  | | PersonCode | | varchar(50) | | Yes | | *NULL* | |
|  | | Status | | enum('available','blocked','te | | Yes | | available | |
|  | |  | | mporary','deleted') | |  | |  | |
|  | | Type | | enum('full\_worker','temp\_wor | | Yes | | undefined | |
|  | |  | | ker','grad\_student','undergrad | |  | |  | |
|  | |  | | \_student','temporary', | |  | |  | |
|  | |  | | 'visitor','undefined') | |  | |  | |
|  | | Check\_Biometric | | tinyint(4) | | Yes | | 0 | |
|  | | Created\_date | | timestamp | | Yes | | 0000-00-00 00:00:00 | |
|  | | Delete\_date | | date | | Yes | | *NULL* | |
|  | | Name | | varchar(250) | | Yes | | *NULL* | |
|  | | LocationNum | | varchar(10) | | Yes | | 0 | |
|  | |  | |  | |  | |  | |
| **licenses** | | License\_ID | | int(11) | | No | |  | |
|  | | Asset\_ID | | int(11) | | Yes | | *NULL* | |
|  | | Name | | varchar(250) | | Yes | | *NULL* | |
|  | | Type | | enum('BSD','Open | | Yes | | Quantity | |
|  | |  | | Source','Dual', 'Quantity') | |  | |  | |
|  | | Licence\_counter | | int(11) | | Yes | | *NULL* | |
|  | | Price | | decimal(10,0) | | Yes | | *NULL* | |
|  | | Term | | varchar(250) | | Yes | | *NULL* | |
|  | | Licence\_compan | | varchar(250) | | Yes | | *NULL* | |
|  | | y | |  | |  | |  | |
|  | | Creation\_date | | timestamp | | Yes | | CURRENT\_TIMESTAMP | |
|  | | Deleted\_date | | date | | Yes | | *NULL* | |
|  | | PurchaseNo | | varchar(250) | | Yes | | not set | |
|  | | RequestNo | | varchar(250) | | Yes | | not set | |
|  | |  | |  | |  | |  | |
| **location\_location** | | Location\_master | | int(11) | | No | |  | |
|  | | \_ID | |  | |  | |  | |
|  | | Location\_child\_I | | int(11) | | No | |  | |
|  | | D | |  | |  | |  | |
|  | | Relation\_type | | enum('contain','unit') | | No | | contain | |
|  | |  | |  | |  | |  | |
| **location\_plan** | | Plan\_ID | | int(11) | | No | |  | |
|  | | Plan\_of\_Location | | int(11) | | Yes | | *NULL* | |
|  | | \_ID | |  | |  | |  | |
|  | | Plan | | varchar(550) | | Yes | | *NULL* | |
|  | |  | |  | |  | |  | |
| **locations** | | Location\_ID | | int(11) | | No | |  | |
|  | | Capacity | | int(3) | | Yes | | 0 | |
|  | | Type | | enum('storage','home', 'floor', | | No | | drawer | |
|  | |  | | 'holl','admin\_office','suite', | |  | |  | |
|  | |  | | 'cubicle','atrium','teaching\_lab | |  | |  | |
|  | |  | | ','research\_lab','grad\_seat','te | |  | |  | |
|  | |  | | acher\_office','drawer','printer | |  | |  | |
|  | |  | | \_room') | |  | |  | |
|  | | Belong\_to | | int(11) | | Yes | | *NULL* | |
|  | | Description | | varchar(1000) | | Yes | | No Description | |
|  | | LocationNum | | varchar(10) | | No | | 0 | |
|  | | KeyNum | | int(11) | | Yes | | 0 | |
|  | | CodeNum | | int(11) | | Yes | | 0 | |
|  | | Width | | varchar(250) | | Yes | | *NULL* | |
|  | | Length | | varchar(250) | | Yes | | *NULL* | |
|  | |  | |  | |  | |  | |
| **person** | | Person\_ID | | int(11) | | No | |  | |
|  | | FirstName | | varchar(250) | | Yes | | *NULL* | |
|  | | LastName | | varchar(250) | | Yes | | *NULL* | |
|  | | UserName | | varchar(50) | | No | |  | |
|  | | Password | | varchar(50) | | Yes | | *NULL* | |
|  | | Address | | varchar(250) | | Yes | | *NULL* | |
|  | | EmailAddress | | varchar(50) | | No | |  | |
|  | | MobileNumber | | varchar(50) | | Yes | | *NULL* | |
|  | | PersonCode | | varchar(50) | | Yes | | *NULL* | |
|  | | Status | | enum('available','blocked', | | No | | available | |
|  | |  | | 'temporary', 'deleted') | |  | |  | |
|  | | Type | | enum('full\_worker','temp\_wor | | No | | undefined | |
|  | |  | | ker','grad\_student','undergrad | |  | |  | |
|  | |  | | \_student','temporary', | |  | |  | |
|  | |  | | 'visitor','undefined') | |  | |  | |
|  | | Check\_Biometric | | tinyint(4) | | Yes | | 0 | |
|  | | Created\_date | | timestamp | | No | | CURRENT\_TIMESTAMP | |
|  | | Delete\_date | | date | | Yes | | *NULL* | |
|  | |  | |  | |  | |  | |
| **person\_assets** | | Asset\_ID | | int(11) | | Yes | | *NULL* | |
|  | | Person\_ID | | int(11) | | Yes | | *NULL* | |
|  | | Type | | varchar(250) | | Yes | | *NULL* | |
|  | | Check\_in\_Date | | timestamp | | Yes | | CURRENT\_TIMESTAMP | |
|  | |  | |  | |  | |  | |
| **Person\_depart** | | Fac\_Dep\_ID | | int(11) | | No | |  | |
|  | | Person\_ID | | int(11) | | No | |  | |
|  | | Type | | enum('works\_for','study\_in') | | No | | study\_in | |
|  | |  | |  | |  | |  | |
| **person\_location** | | Location\_ID | | int(11) | | No | |  | |
|  | | Person\_ID | | int(11) | | No | |  | |
|  | | Type | | enum('grad\_seat','research\_se | | No | | belong | |
|  | |  | | at','responsible','works\_place', | |  | |  | |
|  | |  | | 'belong') | |  | |  | |
|  | |  | |  | |  | |  | |
| **person\_roles** | | Role\_ID | | varchar(250) | | No | |  | |
|  | | Person\_ID | | int(11) | | No | |  | |
|  | |  | |  | |  | |  | |
| **request** | | Request\_ID | | int(11) | | No | |  | |
|  | | Requested\_by\_ID | | int(11) | | Yes | | *NULL* | |
|  | | Location\_to\_ID | | int(11) | | Yes | | *NULL* | |
|  | | Type | | enum('Movement','Acquisitio | | No | | Movement | |
|  | |  | | n', | |  | |  | |
|  | |  | | 'Reparation', | |  | |  | |
|  | | Description | | varchar(2000) | | Yes | | *NULL* | |
|  | | Status | | enum('Not\_Approved','Approv | | No | | Not\_Approved | |
|  | |  | | ed', 'Rejected','Compleated') | |  | |  | |
|  | | Creation\_Date | | timestamp | | No | | CURRENT\_TIMESTAMP | |
|  | | Delete\_date | | date | | Yes | | *NULL* | |
|  | | Period | | int(11) | | Yes | | *NULL* | |
|  | |  | |  | |  | |  | |
| **roles** | | Role\_ID | | varchar(250) | | No | |  | |
|  | | Description | | varchar(2000) | | Yes | | *NULL* | |
|  | |  | |  | |  | |  | |
| **roles\_set** | | Role\_ID | | varchar(250) | | No | |  | |
|  | | Level\_Name | | varchar(250) | | No | |  | |
|  | |  | |  | |  | |  | |
| **sub\_group** | | Asset\_ID | | int(11) | | No | |  | |
|  | | sub\_type | | varchar(250 | | Yes | | *NULL* | |
|  | |  | |  | |  | |  | |
| **voice** | | Voice\_ID | | int(11) | | No | |  | |
|  | | Person\_ID | | int(11) | | Yes | | *NULL* | |
|  | | Voice | | varchar(250) | | Yes | | *NULL* | |

# COMPONENT DESIGN

**5.1 Feed Page**

Feed page will be used for posts with profile picture. There is also cash wallet/option to donate and invest.

**5.2 Favorite**

There’s a Favorite page where Re-Scribble Button, Share Button and Tip button are there for the post to make it favorite for future use.

**6. Human Interface Design**

**6.1 Overview of User Interface**

The following is how user interact with the system.

1. User opens the platform.
2. User signup if he is using first time.
3. User login to the system.
4. Now there are different modules that user can access to.
5. User can post scribble on timeline.
6. User can view and update his profile
7. User can manage blue coins in his profile
8. User can buy blue coins.
9. User can sale blue coins
10. User can view some other user’s profile
11. User can Like others post and gift blue coins.
12. User can post on others timeline if allowed.
13. User can logout the system
14. User can share others post to own timeline
15. User can live chat with other users when he/she is online or send an offline message.

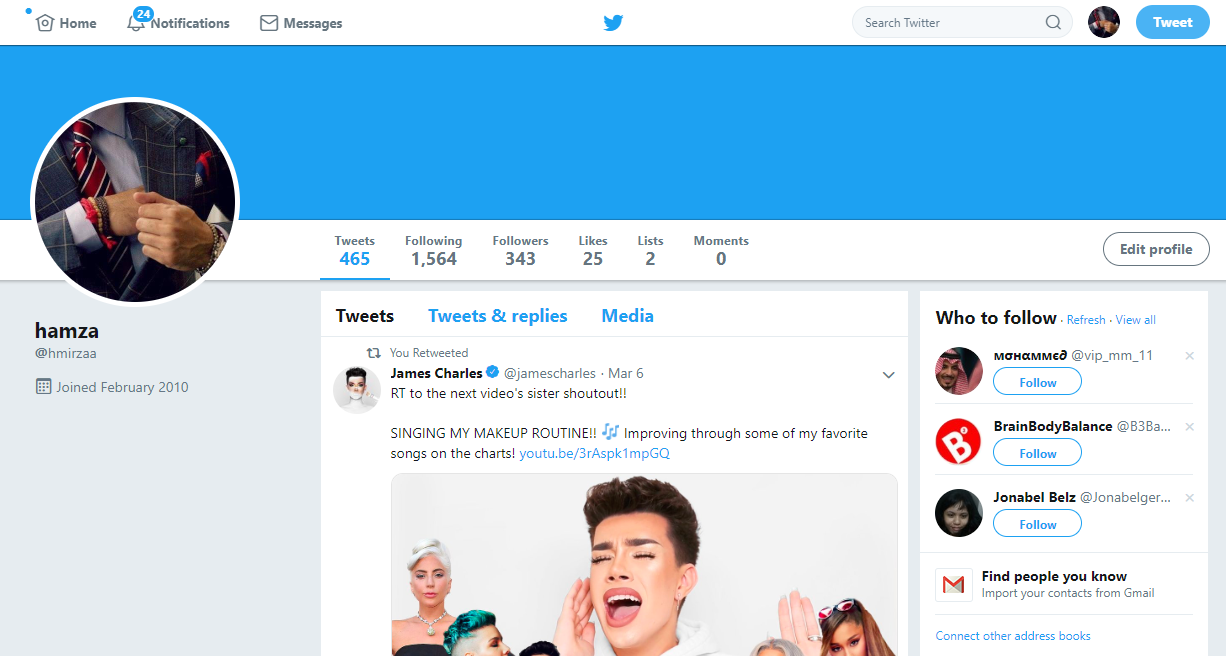
**6.2 Screen Images**

The frontend will be like twitter. So, instead of using hand drawn images. I am using twitter images directly to give the idea about the system.

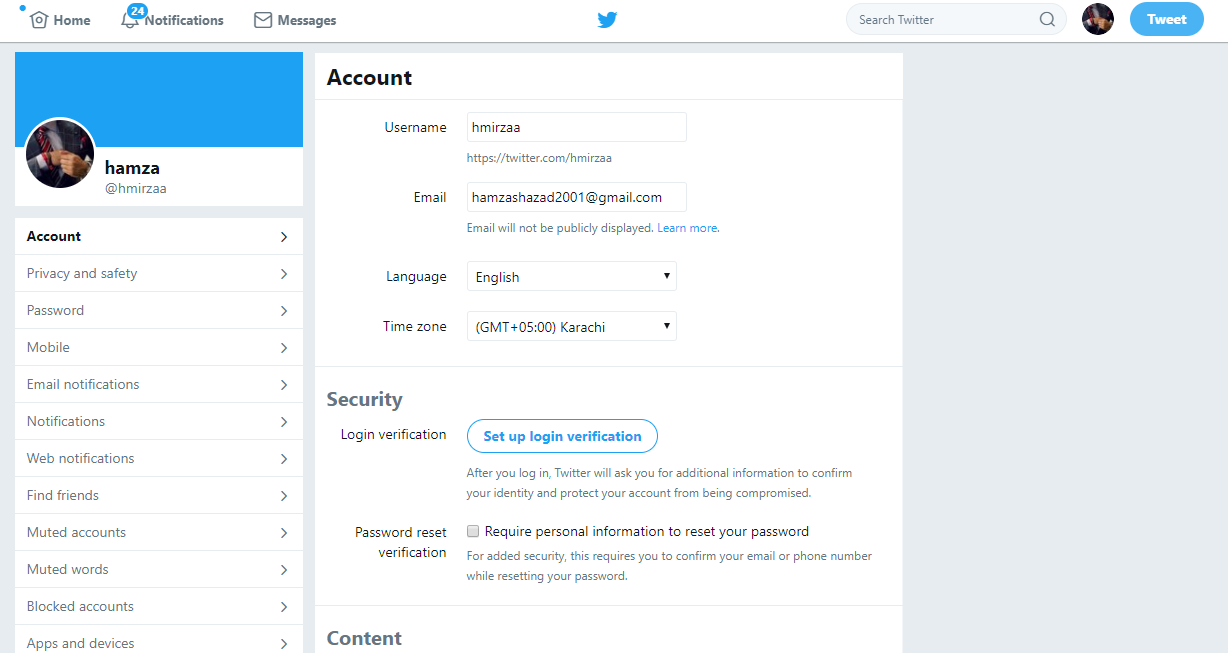
**Main Page**

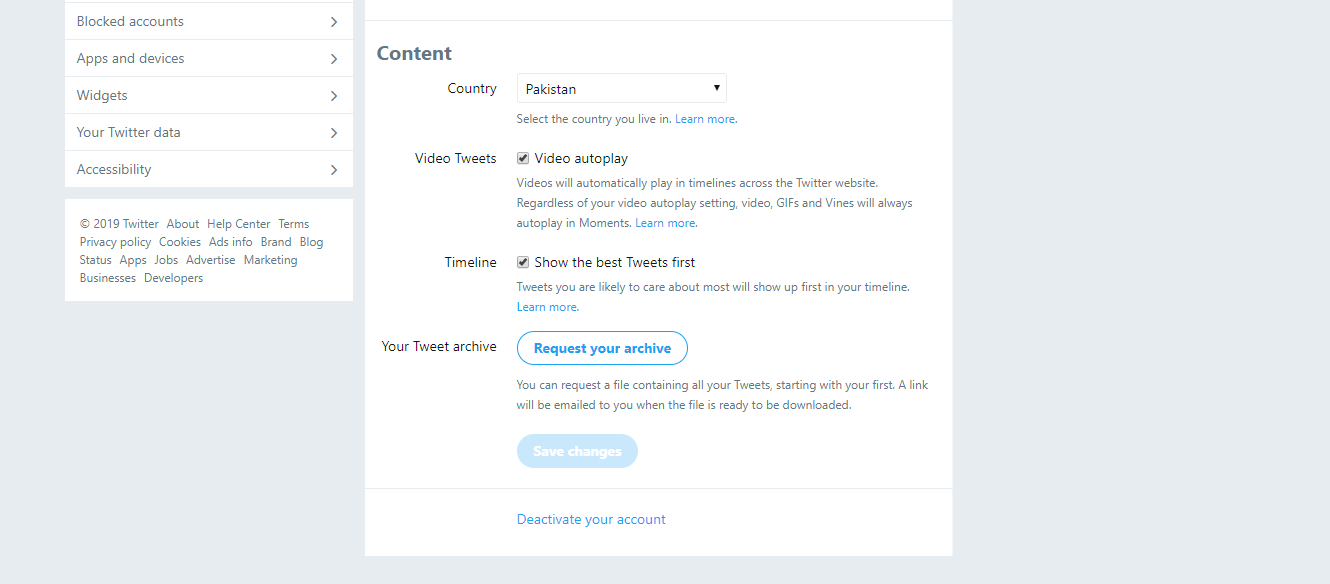


**Profile Page**



**Settings Page**





Rest will be same as twiiter just with an additional feature of managing blue coins.

**6.3 Screen Objects and Actions**

N/A

# HUMAN INTERFACE DESIGN

## Overview of User Interface

Describe the functionality of the system from the user’s perspective. Explain how the user will be able to use your system to complete all the expected features and the feedback information that will be displayed for the user.

## Screen Images

Display screenshots showing the interface from the user’s perspective. These can be hand­ drawn or you can use an automated drawing tool. Just make them as accurate as possible. (Graph paper works well.)

## Screen Objects and Actions

A discussion of screen objects and actions associated with those objects.

# REQUIREMENTS MATRIX

# System Features

A functional requirement defines a function of a system or its component.

## System Feature 1

Create Account DESCRIPTION- If user is new and does not have Life Scribble account. INPUT-Request for first name,last name , email id/phone no. and password. PROCESSING-Retrieves the provided information and makes a new account for user. OUTPUT- Displays created account.

## System Feature 2

Searching Friends DESCRIPTION- User searches for friends to add in friend list. INPUT- Click on search,enter name of friend. PROCESSING- User clicks on search , enters name and then list Of persons displayed of that name. OUTPUT- Name of Friend displayed.

## System Feature 3

Sending Request DESCRIPTION- User can send request to a person he/she wants to add in his/her friend list. INPUT- Tap on ‘add friend’. PROCESSING- When user tap on ‘add friend’,friend request sent displayed. OUTPUT-Friend request sent.

## System Feature 4

Accepting Friend Request DESCRIPTION- When someone sends friend request to user , user gets notification whether he wants to ‘accept’ or ‘delete’. INPUT-Click on ‘accept’ button. PROCESSING- User clicks on ‘accept’ and friend added to his/her friend list. OUTPUT-Friend added in friend list.

## System Feature 5

Creating Groups DESCRIPTION- One can create groups on Life Scribble and add some peoples from his/her friend list. INPUT- Click ‘profile’ at the top, click ‘groups’ ,click ‘see all’, tap ‘create group’,name group, add some people,add description , select privacy and click ‘save’. PROCESSING- When someone click on ‘groups’ transferred to ‘see all’ and adds the name,people,description and selects privacy transferred to ‘save’. OUTPUT-Group created.

## System Feature 6

Uploading Photos DESCRIPTION- User can add photos on Life Scribble to update his activities. INPUT- Tap ‘photo’,select photo to upload ,tap ‘done’. PROCESSING- user clicks on ‘photo’,transferred to ‘select’ , photo selected by user then transferred to tap ‘done’. OUTPUT-Photo uploaded successfully.

## System Feature 7

Creating Albums DESCRIPTION- User can add Life Scribble photos and create album INPUT-Go to ‘update status’,create ‘photo album’,choose order of photos,choose album cover,choose privacy and post. PROCESSING- OUTPUT-Album Created.

## System Feature 8

Sharing Status DESCRIPTION- When someone clicks Share below a post, they are able to share your photos, videos or status updates through Life Scribble. INPUT-Search post to share, tap ‘share’ , tap ‘write post’ and tap ‘share now’. PROCESSING- User clicks on share , transferred to write post and then clicks on ‘share now’ to share. OUTPUT- Post shared by the user.

## System Feature 9

Create New Page DESCRIPTION- One can create a new page on Life Scribble to add his/her activities and connect to friends. INPUT- Tap on ‘pages’,tap ‘create page’,tap ‘get started’,select name,select categories ,add cover photo,profile picture for page,tap ‘visit page’. PROCESSING- User tap on ‘pages’ ,transferred to ‘create page’ and then clicks ‘get started’ ,transferred to add name ,cover photo ,profile photo and then tap on ‘visit page’ to see the created page. OUTPUT- A new page created .

## System Feature 10

Sending Message DESCRIPTION- User should be able to send instant message to any contact on his/her contact list. User should be notified when message is successfully delivered to recipient by displaying a tick sign next to message sent. INPUT- Message typed. PROCESSING- Message send to other user. OUTPUT- Tick on Message .

## System Feature 11

Send Attachments DESCRIPTION- User should be able to send audio,video and images as attachments. INPUT- File attached. PROCESSING-Send to other side user. OUTPUT-Tick on file.

## System Feature 12

Commenting DESCRIPTION- Life Scribble comments are key to understanding how users engage with one’s content. INPUT- Click the comment link, type comment,press enter to publish. PROCESSING- Post the comment on user’s attachment. OUTPUT- Comment attached.

## System Feature 13

Uploading Videos DESCRIPTION- User can upload video in his/her account . INPUT- Click ‘add video’ button,choose file,add description and post. PROCESSING- User clicks on ‘add video’, transferred to choose files and then clicks on ‘post’ to upload and then video uploaded. OUTPUT-Video uploaded successfully.

## System Feature 14

Notes DESCRIPTION- User can add notes INPUT- Select ‘more’ at right of profile picture, click ‘notes’, click ‘add notes’ , drag file,create note and ‘publish’. PROCESSING- User clicks on ‘more’ , choose notes and clicks to add , and then attaches file then click to create and ‘publish’. OUTPUT- Note published.

## System Feature 15

Videos DESCRIPTION- User can watch uploaded videos on Life Scribble. INPUT-Search videos,click to watch. PROCESSING- User searches for a video in ‘search’ and a list of videos related to search displayed and then click to watch. OUTPUT- video watched by user.

## System Feature 16

Notifications DESCRIPTION- Notifications are updates about activity on Life Scribble. INPUT-Tap the globe icon,click see all. PROCESSING- user click on globe icon then list of notifications displayed. OUTPUT-List of notifications displayed.

# APPENDICES

* 1. References:

1. [www.facebook.com](http://www.facebook.com)
2. [www.wikipidia.com](http://www.wikipidia.com)
3. [www.twitter.com](http://www.twitter.com)