Group Circle

Submitted By

Jahid Ahmed
ID: 2012-3-60-033

Md. Moniruzzaman Khondaker
ID: 2012-2-60-056

Abir Mohammad Rafi Al Sami
ID: 2012-3-60-014

Supervised By

Md.Shamsujjoha
Senior Lecturer
Department of Computer Science & Engineering
East West University

A Project Submitted in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING EAST WEST UNIVERSITY

December, 2017

ABSTRACT

This project is about online social media service with some features and interesting user experience. In recent year the online social media services are getting popular. Our system is for everybody of age limit over 14 years. The user can share their thought and can update and delete the post with the feature of uploading photos. User can share their post anonymously and can chat with them as a group. They can send individual message to their friends. The user can login directly or via Facebook, Google, Twitter, Linkdin, Github if he/she has been register. The user will be notifying when his/her friend is online or if anybody has commented on their post. The user can search their friend. Our project can be popular among the groups and with their friends and they can create a new group or circle with our site.

DECLARATION

I hereby declare that, this project has been done by us under the supervision of **Md. Shamsujjoha, Senior Lecturer & Assistant Proctor, Department of Computer Science and Engineering, East West University**. I also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma. Any material reproduced in this project has been properly acknowledged.

Submitted by:

_____

Jahid Ahmed
ID: 2012-3-60-033
Department of Computer Science and Engineering
East West University

_____

Md. Moniruzzaman Khondaker
ID: 2012-2-60-056
Department of Computer Science and Engineering
East West University

_____

Abir Mohammad Rafi Al Sami
ID: 2012-3-60-014
Department of Computer Science and Engineering
East West University

Supervised by:

_____

Md. Shamsujjoha
Senior Lecturer
Department of Computer Science and Engineering
East West University
Bangladesh

LETTER OF ACCEPTANCE

This Project entitled **"Group Circle"** submitted by Jahid Ahmed (2012-3-60-033), Md. Moniruzzaman Khondaker (2012-2-60-056) and Abir Mohammad Rafi Al Sami (2012-3-60-014) to the Department of Computer Science and Engineering, East West University, has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on 07th December, 2017.

Supervisor

_____

Md. Shamsujjoha
Senior Lecturer
Department of Computer Science and Engineering,
East West University, Dhaka, Bangladesh

Chairperson

_____

Dr. Ahmed Wasif Reza
Associate Professor and Chairperson
Department of Computer Science and Engineering,
East West University, Dhaka, Bangladesh

ACKNOWLEDGEMENT

Any accomplishment requires the effort of many people and there are no exceptions. The report being submitted today as a result of collective effort and we would like to extend our sincere thanks to all of them.

We would like to express our deepest gratitude to the almighty for His blessings on us. Next, our special thanks go to our supervisor, Md. Shamsujjoha, who gave us this opportunity, initiated us into the field of “Group Circle”, and without whom this work would not have been possible. Whenever, we came up with complicated issues, he guided us the simple way to resolve the issues.

And last but not the least we place a deep sense of gratitude to our family members and my friends who inspires us during the preparation of this project work. We will remember them in our heart and hope to find a more appropriate place to acknowledge them in the future.

TABLE OF CONETNTS

Chapter 1

Introduction

1.1 Introduction

Now internet is popular and for that web application is getting popular day by day. Most of the internet services like Google, YouTube, Facebook are most popular among the users. Revolutions in internet are advanced so far that the communication system from one end of the world to another end of the world becomes easier. More than hundreds of million people worldwide using World Wide Web(www) to fulfill their demand and jobs.

Almost every website supports a poplar formatted markup language called Hyper Text Markup Language(HTML). Some other language such as Cascading Style Sheets(CSS) and Java Script(JS) are popular in web designing and styling the web pages. For backend of the website Hypertext Preprocessor(PHP) and MySQL are used to function the site dynamically.

"Group Circle" is an online social media system which is built to communicate. Anybody above 14 years old can be registered on this system and can make new friends and can communicate with them. Admin has privilege upon editor and the normal users. Admin can give access to the editor to manipulate the user list by adding and deleting the users.

1.2 Motivation

For our project we have considered a real time chatting system without load and for status share the user can add new category and the user can see this post. Our system makes comfortable to the user by implementing a customizable option

1.3 Objectives

- To create new post
- To create same mentality base community
- To share knowledge between the friends in chatting
- To have recreation moment with the friends

1.4 Scopes

For Admin:

- Have privilege to the editor and the users
- Can add or delete the editor
- Can delete any users
- Update profiles of editor

For Editor:

- Have privilege to the user only
- Can delete any unwanted or fake users

For User:

- Update/edit/delete status and upload/delete photos
- Can Search Friends
- Can have group chat and individual messaging options
- Update personal information and details
- Can post anonymous status and anonymous comment

1.5 Technology Stack

For building our project we have used Laravel, a PHP based framework, as our main programming language. For the database system we have used MYSQL and for designing the view in both front-end and back-end, we have used HTML, CSS, JavaScript and Bootstrap.

Markup Language: Hyper Text Markup Language(HTML), Cascading Style Sheets(CSS),

Java Script(JS),Bootstrap

Programming Language: PHP

Framework: Laravel (PHP Base)

Database System: MySQL

Server: Apache Web Server(Xampserver)

Operating System: Windows

Web Browser: Mozilla Firefox, Google Chrome, Opera, Microsoft Edge

1.6 Limitation

Need high-speed internet connections and as a social site, need to improve design and more facility or option is needed.

Chapter 2

Analysis and Design

Development of web based systems requires analysis of the process to be digitized in order to enable a error free system, a system that functions as required and helps to understand the general functionality of the system. The analysis specifies the system's objectives and constraints to which designers have to maintain. The purpose of doing analysis is to transform the system's major inputs into structured specification. This has mainly three requirement section- Admin section, Editor section and User section.

2.1 Requirement Analysis

- ✓ The user needs to register to the system giving personal information such as name, email, phone and other info which are required. To verify the account a confirmation link will be sent to the email for verification.
- ✓ After verification the user can login with email and password, which has been used while registration, then can update other information such as upload profile picture, upload some other detail information, post status and chat with friend or message them individually.
- ✓ The editor will be chosen by admin and allow them some special features to delete any unwanted users for this system
- ✓ The admin who has special functionality to add or remove the editor and can maintain the unwanted users as well.
- ✓ The system has search option to find friends or family(if they are registered to system), can share anonymous post to friends and can add custom category to post.

2.2 Use case Diagram

Use Case Diagram for Admin Panel

Use case diagram for admin panel is given in the following figure 2.2

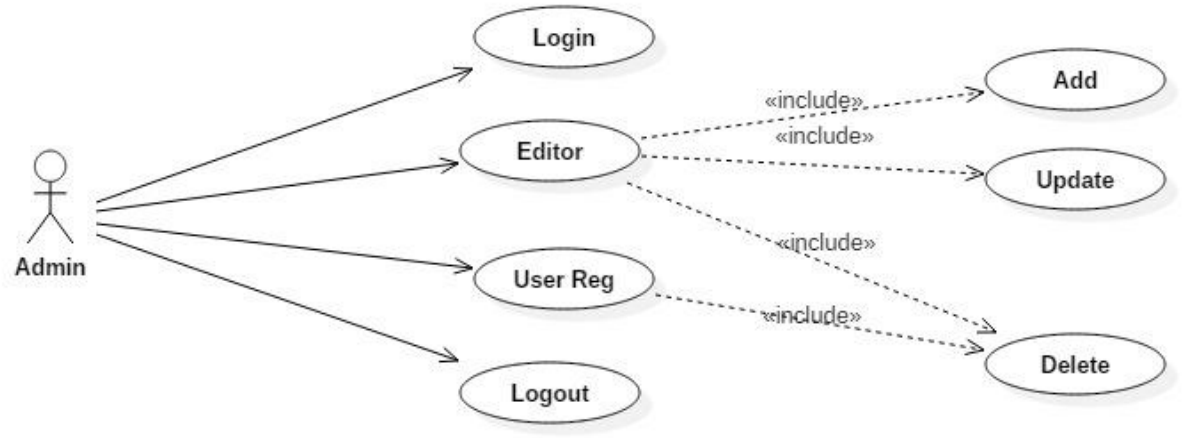


Fig 2.2: Use case diagram for Admin Panel

Functionalities provided:

- Admin need to login first to perform operation
- Admin can add editors
- Admin can update editors
- Admin can delete editors
- Admin can delete registered users

2.3 Use case Diagram

Use Case Diagram for Editor Panel

Use case diagram for editor panel is given in the following figure 2.3

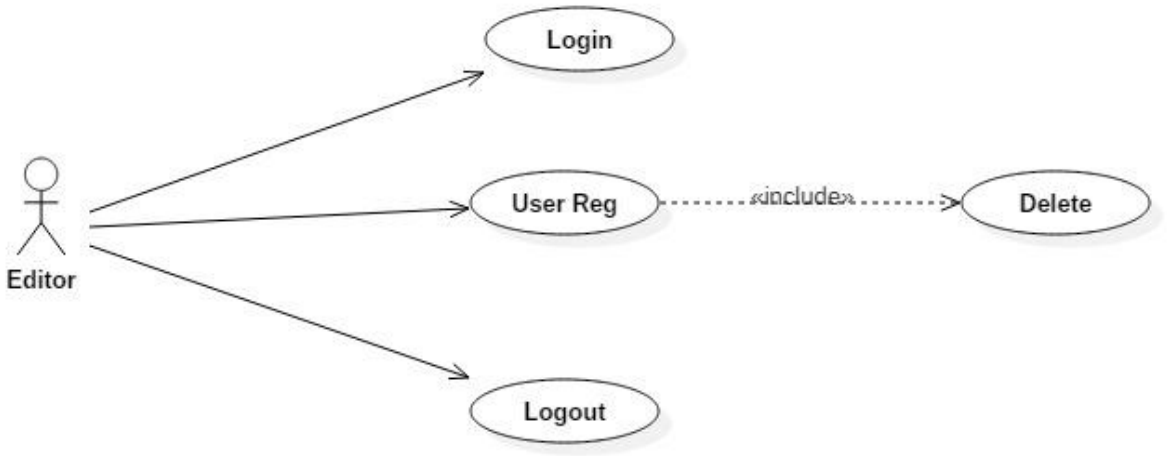


Fig 2.3: Use case diagram for Editor Panel

Functionalities provided:

- Editor need to login first to perform operation
- Editor can delete registered users

2.4 Use case Diagram

Use Case Diagram for User Panel

Use case diagram for user panel is given in the following figure 2.4

Fig 2.4: Use case diagram for User Panel

Functionalities provided:

- User need to login first to perform operation
- User can update and remove profile
- User can Search other Users
- User can post, update and remove photos
- User can post, update and remove Status
- User can chat with others in private and group
- User can add, update and remove comment
- User can view others feed

2.5 Activity Diagram

Activity Diagram for Admin panel

Activity diagram for admin panel is given in the following figure 2.5

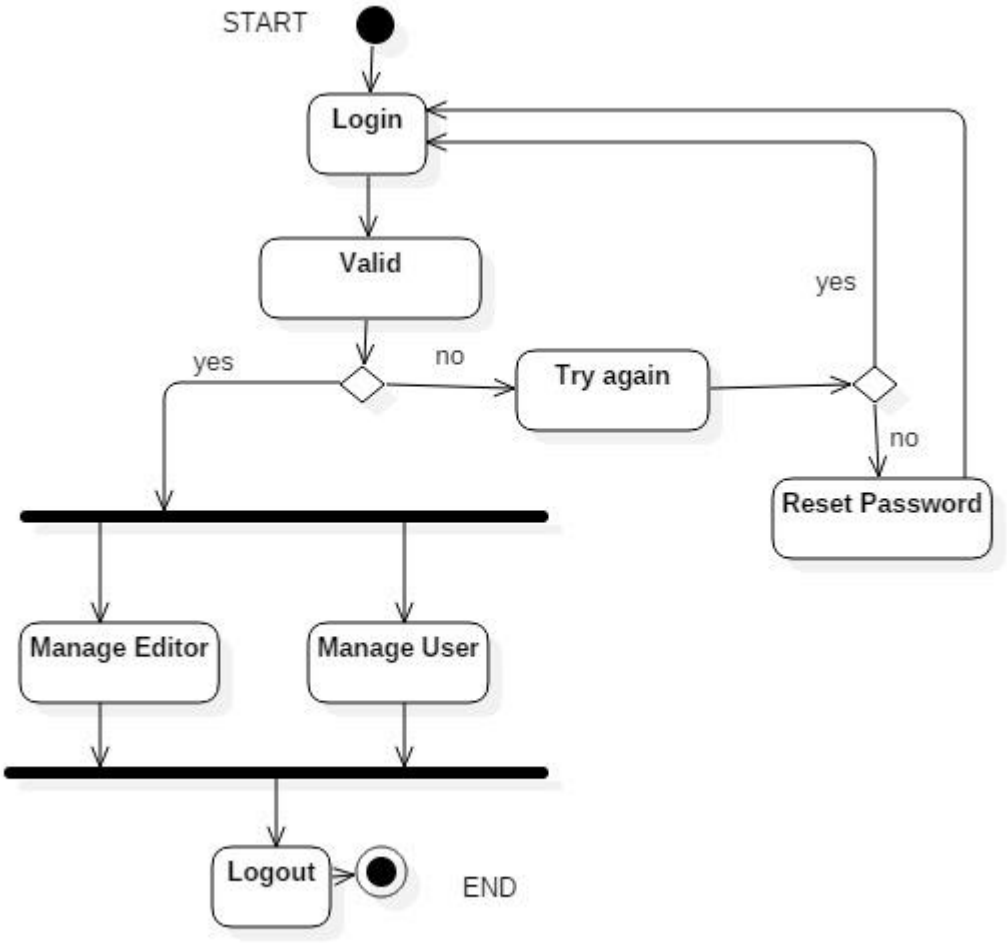


Fig 2.5: Activity diagram for Admin Panel

2.6 Activity Diagram

Activity Diagram for Editor Panel

Activity diagram for editor panel is given in the following figure 2.6

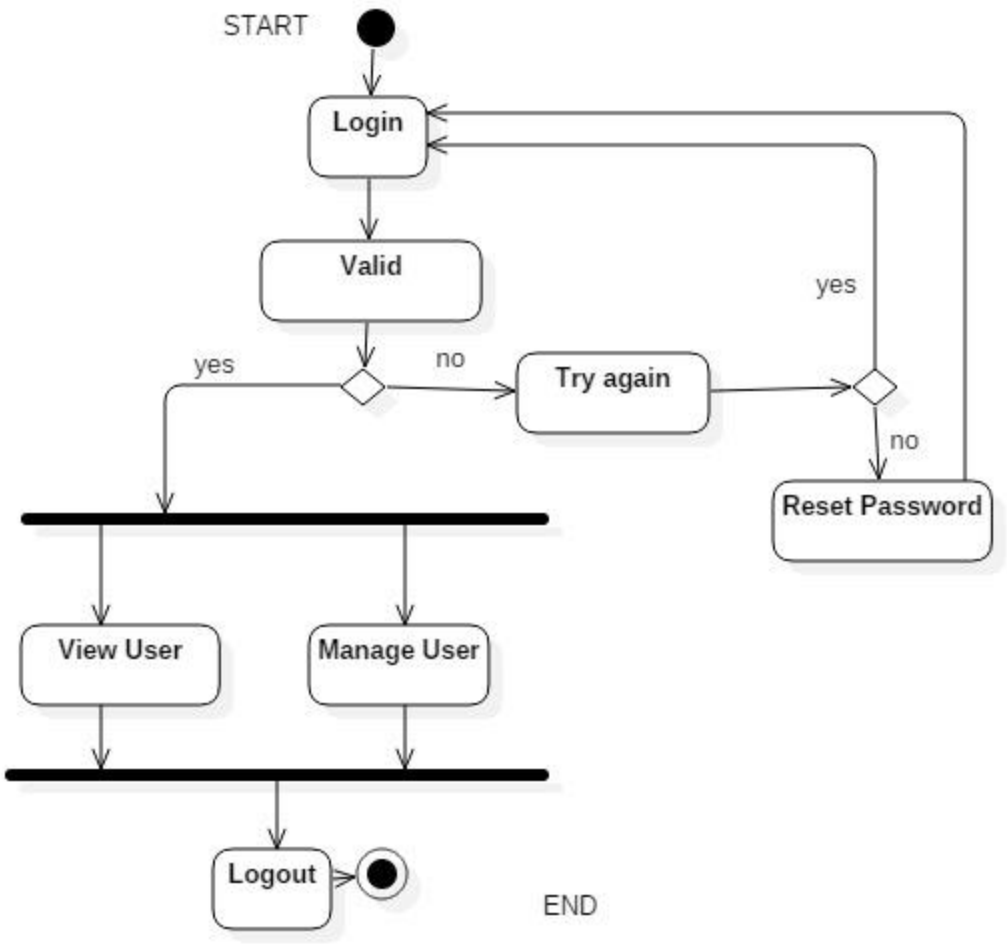


Fig 2.6: Activity diagram for Editor Panel

2.7 Activity Diagram

Activity Diagram for User panel

Activity diagram for user panel is given in the following figure 2.7

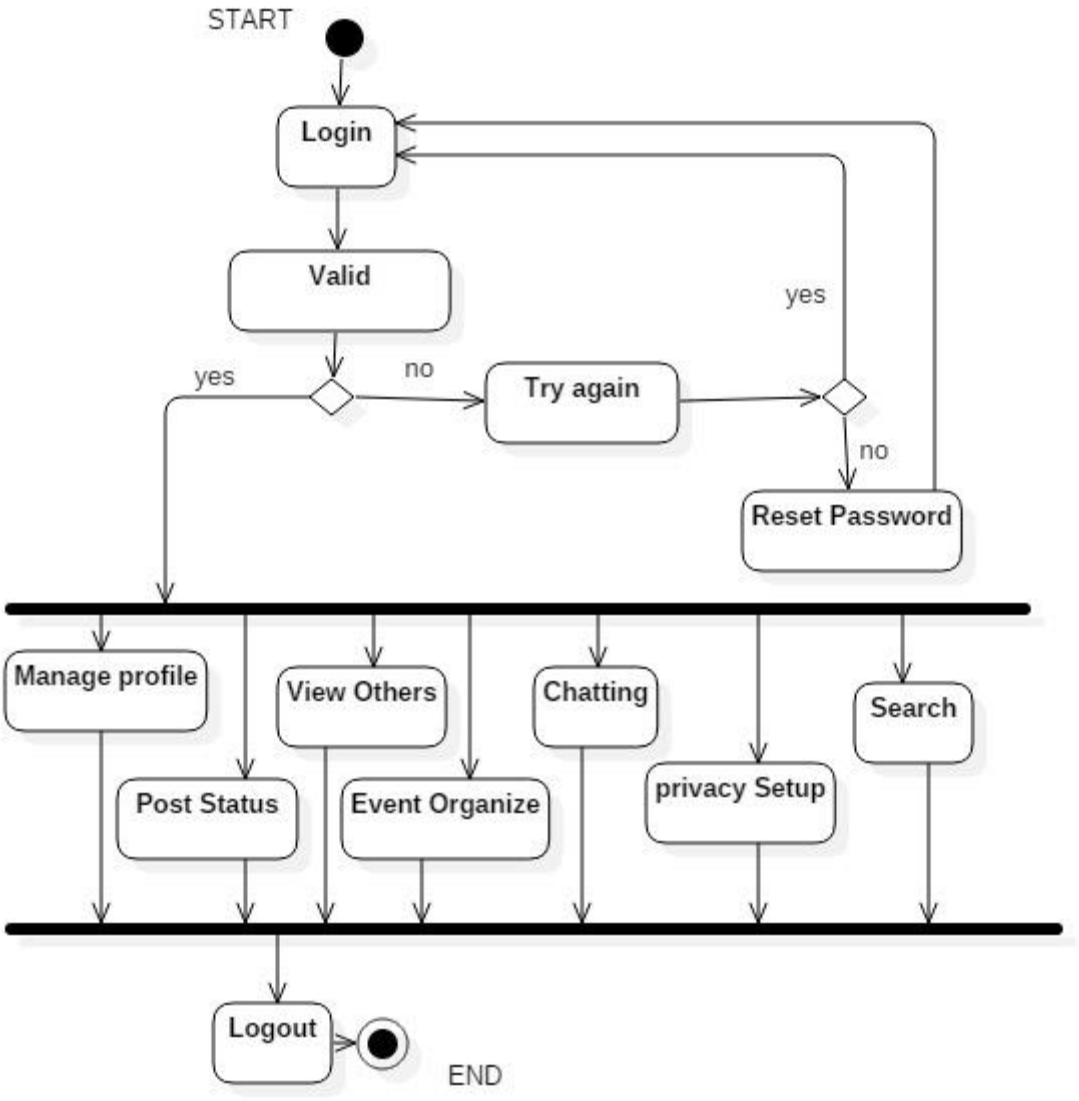


Fig 2.7: Activity diagram for User Panel

Chapter 3

Implementation of the project

3.1 Software and Language

A web application or web app is any software that runs in a web browser. It is created in a browser-supported programming language (such as the combination of JavaScript, HTML and CSS) and relies on a web browser to render the application.
Web programming is different from just programming, which requires interdisciplinary knowledge on the application area, client and server scripting, and database technology. Web programming can be briefly categorized into client and server coding. The client side needs programming related to accessing data from users and providing information. It also needs to ensure there is enough plug-ins to enrich user experience in a graphic user interface, including security measures.

- HTML
- CSS
- Bootstrap
- Java Script
- PHP 7
- MySQL
- Laravel5.4
- Veu.js
- jQuery
- W3CSS Framework

3.1.1 HTML

HTML stands for Hyper Text Markup language which is used to create web pages and also create user interfaces for mobile and web applications. A markup language is a set of markup tags where the markup tells the web browser how to display a web pages words and images for the user. Web pages are simply strings of words put in a special format that web browsers are able to display.

3.1.2 CSS

A cascading style sheet (CSS) is a Web page derived from multiple sources with a defined order of precedence where the definitions of any style element conflict. The Cascading Style Sheet, level 1 recommendation from the World Wide Web Consortium, which is implemented in the latest versions of the Netscape and Microsoft Web browsers, specifies the possible style sheets or statements that may determine how a given element is presented in a Web page. CSS gives more control over the appearance of a Web page to the page creator than to the browser designer or the viewer. With CSS, the sources of style definition for a given document element are in this order of precedence.

3.1.3 Bootstrap

Bootstrap is a framework is used for web customization. Some the most popular features of bootstrap used for ‘speed of development’, for ‘responsiveness’, for ‘consistency’ and for ‘customizable’ the websites.

3.1.4 JavaScript

JavaScript is a high-level, dynamic, weakly typed, prototype-based, multi-paradigm, and interpreted programming language. Alongside HTML and CSS, JavaScript is one of the three core technologies of World Wide Web content production. It is used to make webpages interactive and provide online programs, including video games. As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative (including object-oriented and prototype-based) programming styles and initially only implemented client-side in web browsers.

3.1.5 PHP 7

PHP is an alternative to Microsoft's Active Server Page (ASP) technology. As with ASP, the PHP script is embedded within a Web page along with its HTML. Before the page is sent to a user that has requested it, the Web server calls PHP to interpret and perform the operations called for in the PHP script. An HTML page that includes a PHP script is typically given a file name suffix of php .php3," or ".phtml". Like ASP, PHP can be thought of as dynamic HTML pages, since content will vary based on the results of interpreting the script.

3.1.6 MySQL

MySQL is an open source Relational Database Management System (RDBMS) based on Structured Query Language (SQL). It is a database system used on the web and that is run on a server. It is very fast, reliable, easy to use and ideal for both small and large applications.

3.1.7 Laravel 5.4

Laravel is a web application framework with expressive, elegant syntax.Laravel attempts to take the pain out of development by easing common tasks used in the majority of web projects, such as authentication, routing, sessions, and caching.Some of the features of Laravel are :Bundles provide a modular packaging system for easy addition to applications. Eloquent ORM (object-relational mapping) is an advanced PHP implementation of the active record pattern, providing at the same time internal methods for enforcing constraints on the relationships between database objects. Query builder provides a more direct database access alternative to the Eloquent ORM. Application logic is an integral part of developed applications, implemented either by using controllers or as part of the route declarations.

3.1.8 Veu.js

Veu.js is a popular framework of JavaScript is used for building user interfaces to organize and simplify web development. Integration of veu.js into a project, that uses other JavaScript libraries, makes easier to develop because it is designed with an incrementally adoptable architecture.The core library focuses on declarative rendering and component composition, and can be embedded into existing pages. Some of the advanced features are used for complex applications.

3.1.9 jQuery

jQuery is a JavaScript library is used for client-side scripting of a website. The syntax of jQuery is designed such a way for navigating a document, selecting DOM(Document Object Model) elements, creating animation, event handling and developing AJAX application.

3.1.10 W3.CSS

W3.CSS is a cascading style sheet developed by w2schools.com. Inspired from Google Material Design it helps to creates website faster, beautiful and responsive as well. It is a modern CSS framework with built-in responsiveness. Some the features are :

Smaller and faster than any other CSS framework

Better cross-browser compatibility

Use standard CSS

Chapter 4

Design Interface

4.1 Database Design

Database Name:db_project497

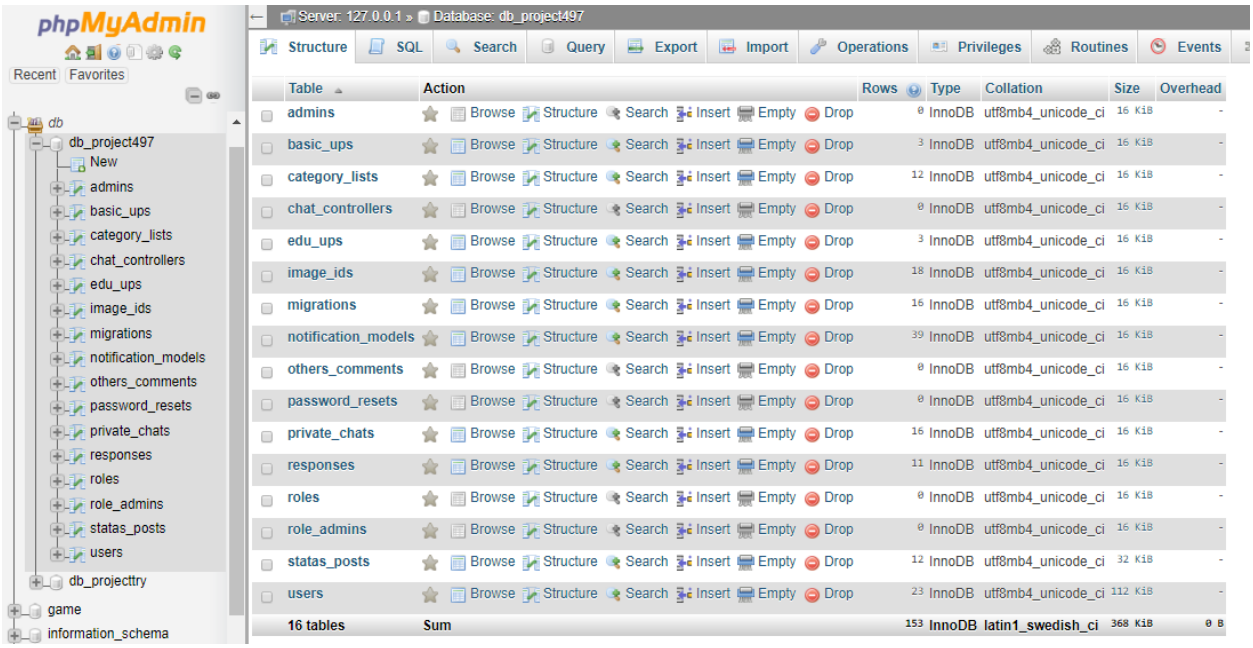


Figure 4.1.1:*All Tables*

Table Name 1:admins

Description: Attributes of table ‘admins’

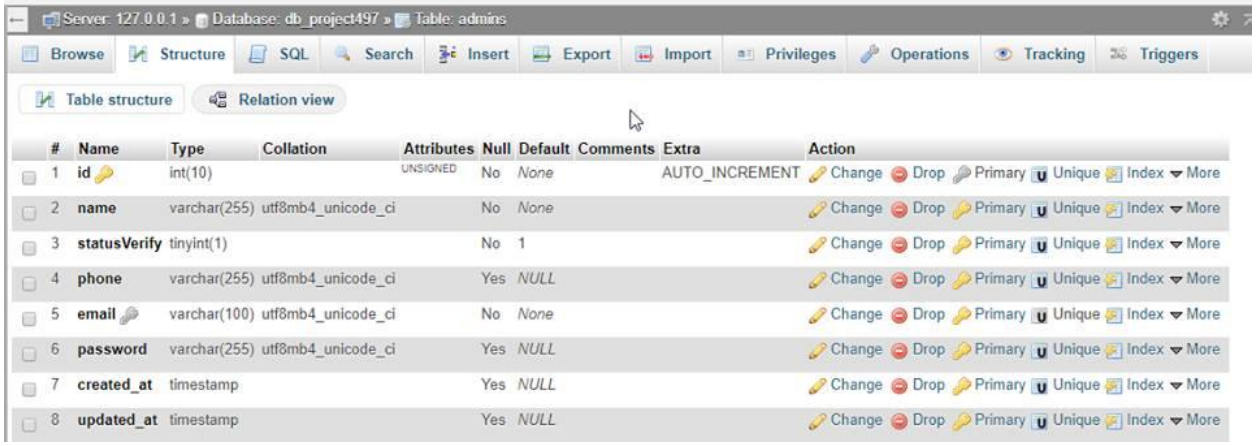


Figure 4.1.2: Table ‘admins’

Table Name2:basic_ups

Description: Attributes of table ‘basic_ups’

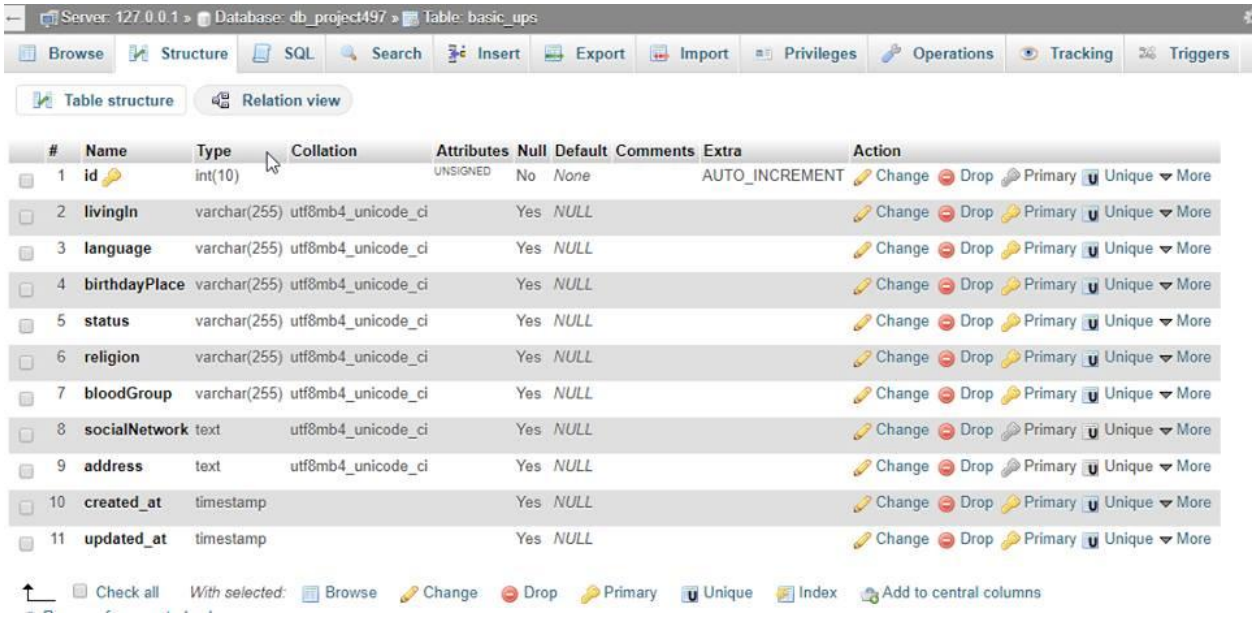


Figure 4.1.3: Table‘basic_ups’

Table Name 3:category_lists

Description: Attributes of table ‘category_lists’

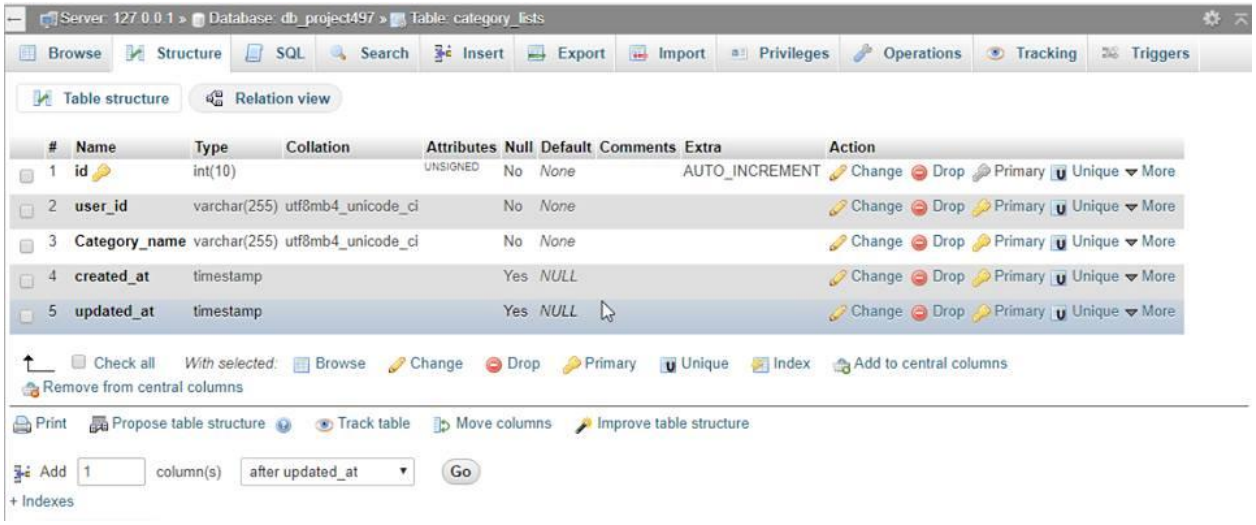


Figure 4.1.4: Table ‘category_lists’

Table Name 4:chat_controllers

Description: Attributes of table ‘chat_controllers ’

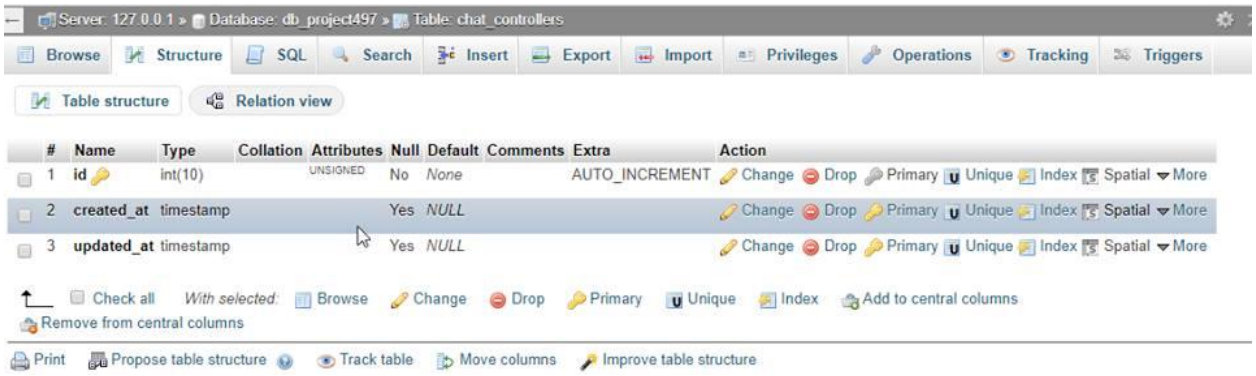


Figure 4.1.5: Table ‘chat_controllers’

TableName 5:edu_ups

Description: Attributes of table ‘edu_ups’

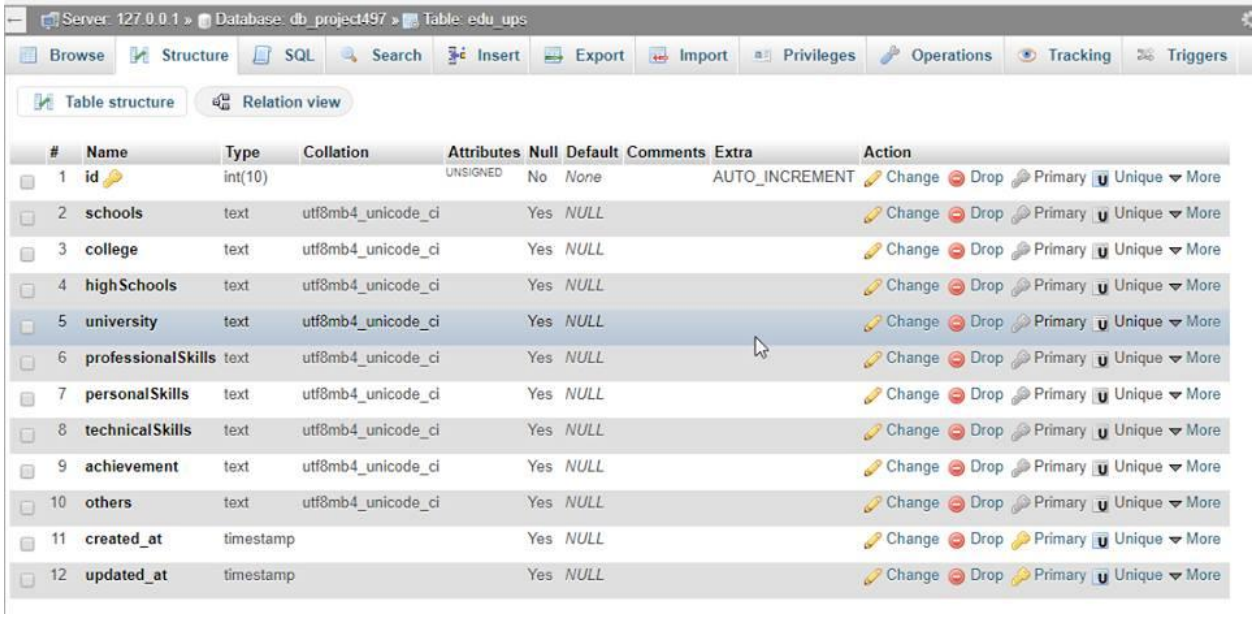


Figure 4.1.6: Table ‘edu_ups’

Table Name 6:image_ids

Description: Attributes of table ‘image_ids ’

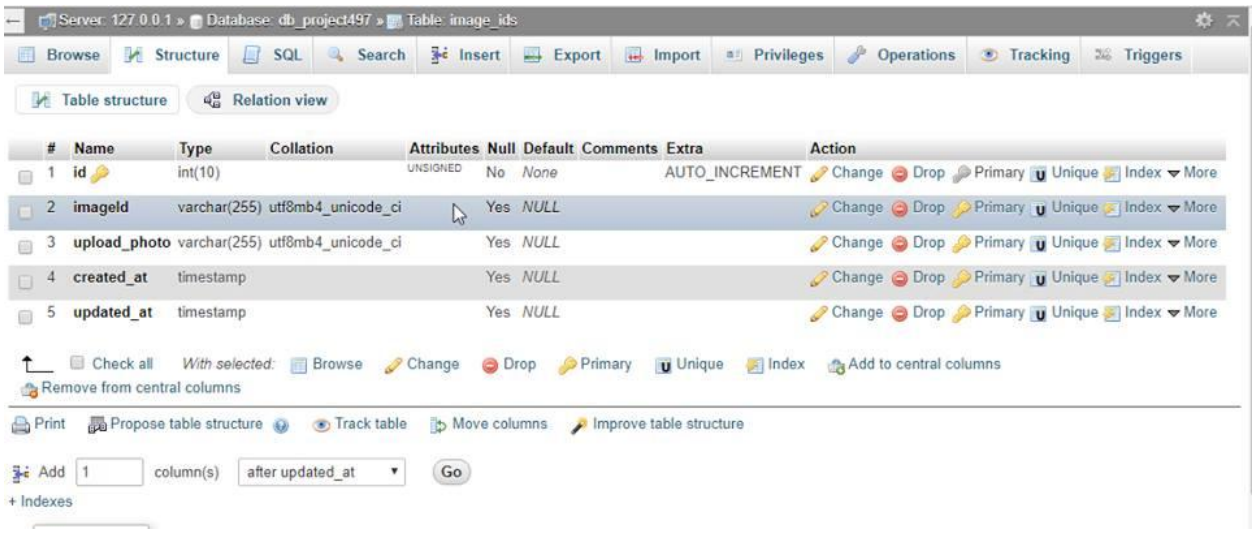


Figure 4.1.7: Table ‘image_ids’

TableName7:migrations

Description: Attributes of table ‘migrations’

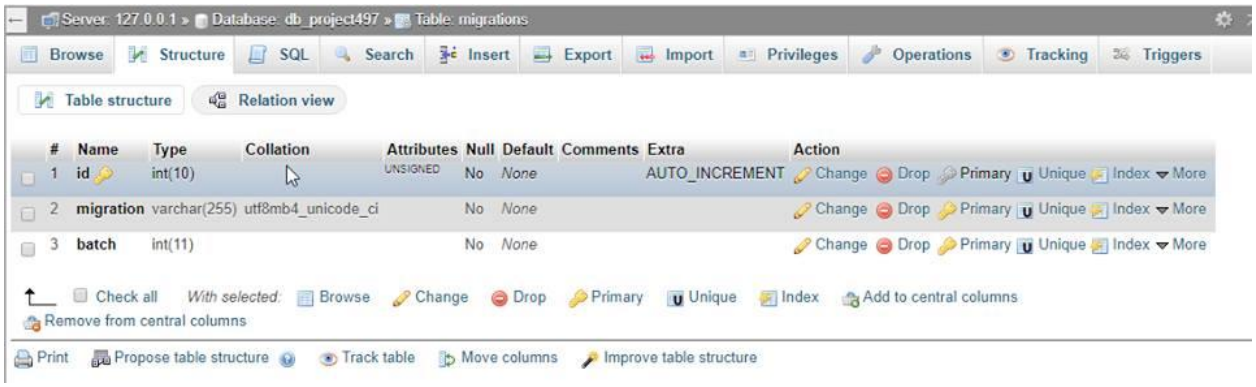


Figure 4.1.8: Table ‘migrations’

Table Name 8:password_resets

Description: Attributes of table ‘password_resets’

Figure 4.1.9: Table ‘password_resets’

Table Name 9:responses

Description: Attributes of table ‘responses’

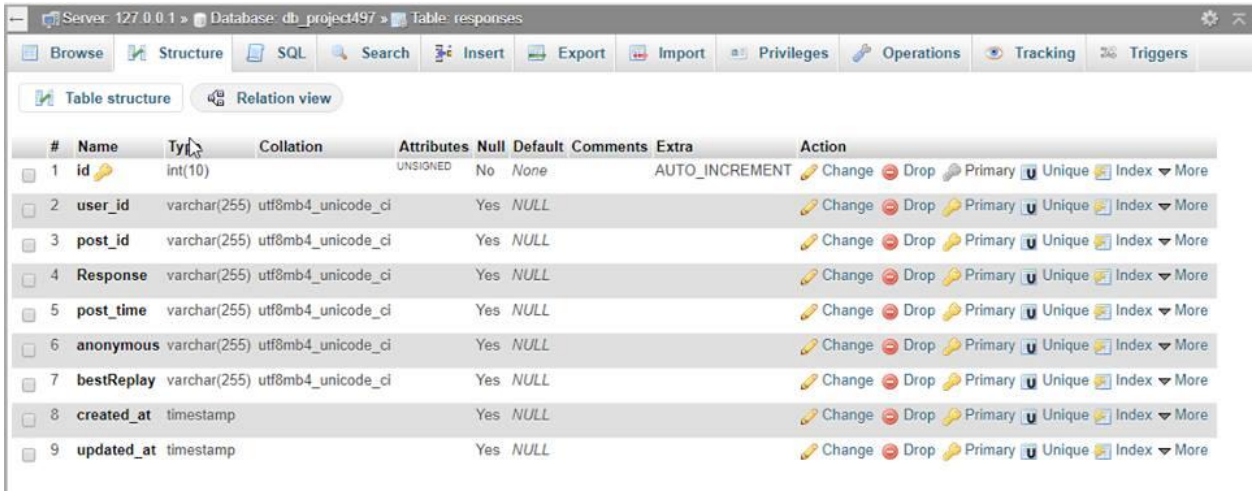


Figure 4.1.10: Table ‘responses’

Table Name 10:roles

Description: Attributes of table ‘roles’

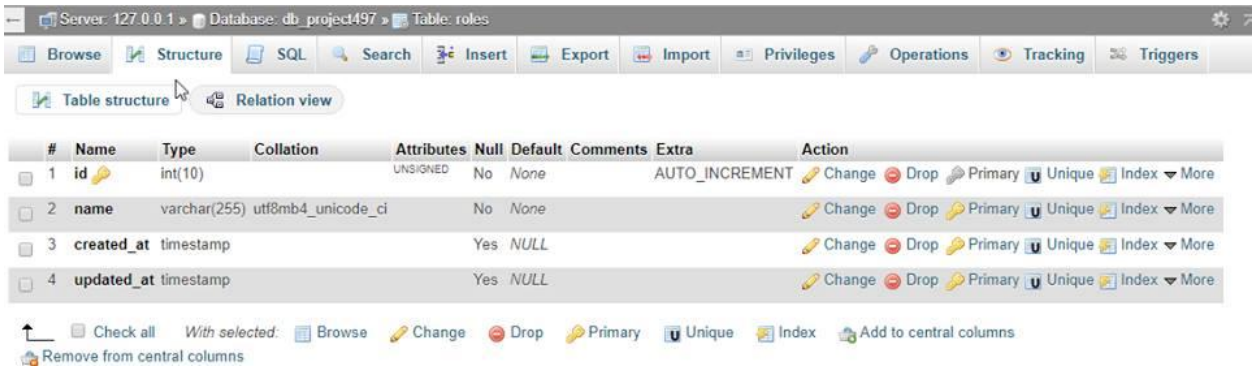


Figure 4.1.11: Table ‘roles’

Table Name 11:role_admins

Description: Attributes of table ‘role_admins’

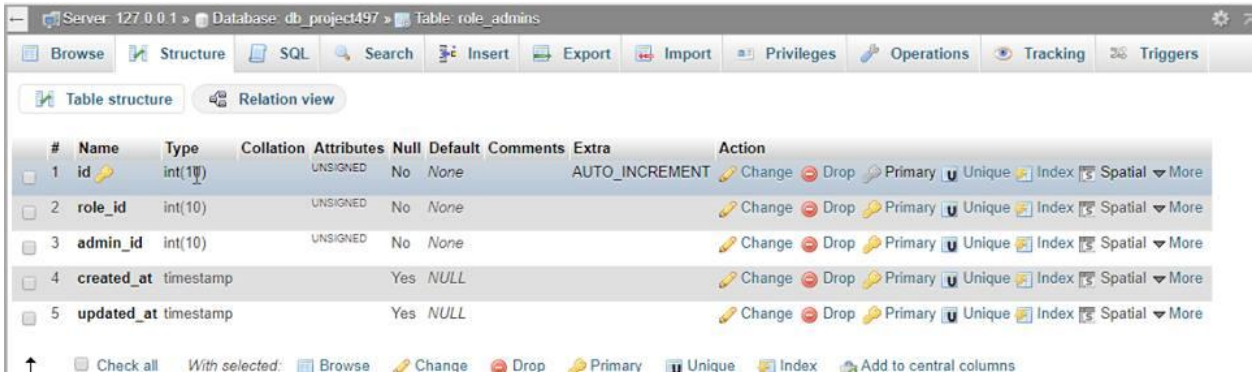


Figure 4.1.12: Table ‘role_admins’

Table Name 12:statas_posts

Description: Attributes of table ‘statas_posts’

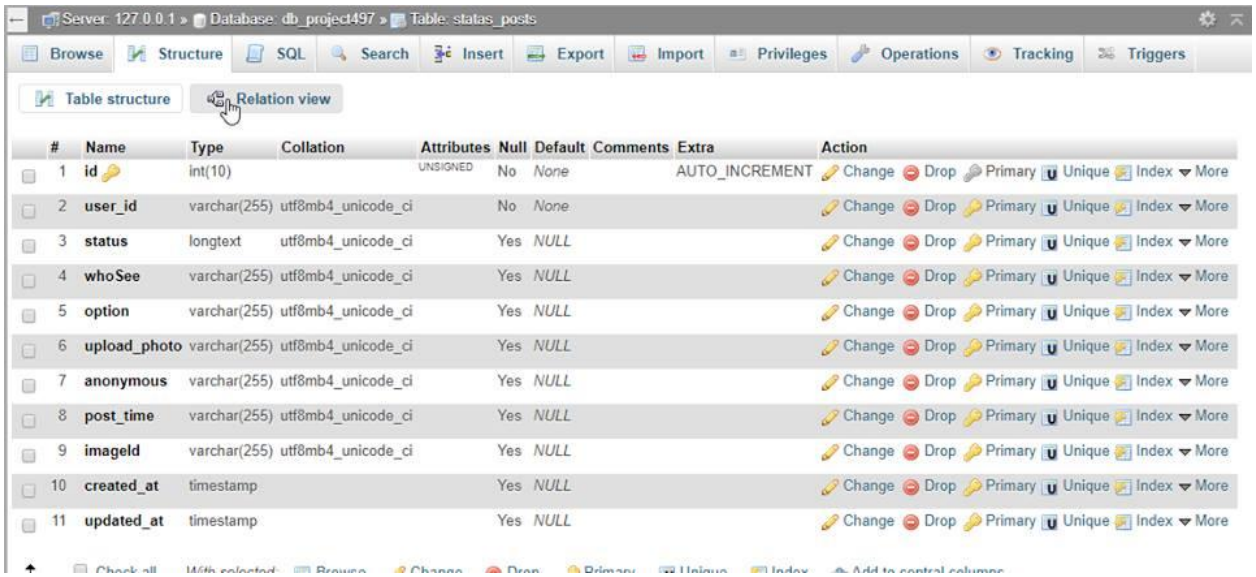
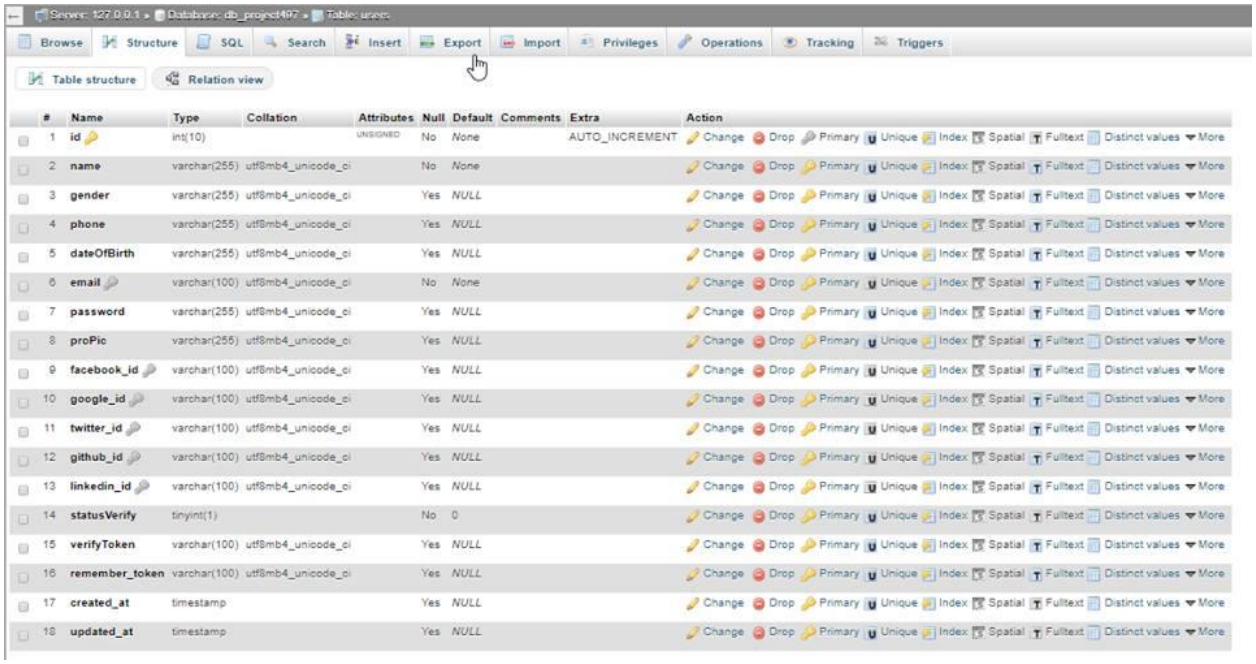


Figure 4.1.13: Table ‘statas_posts’

Table Name 13:users

Description: Attributes of table ‘users’

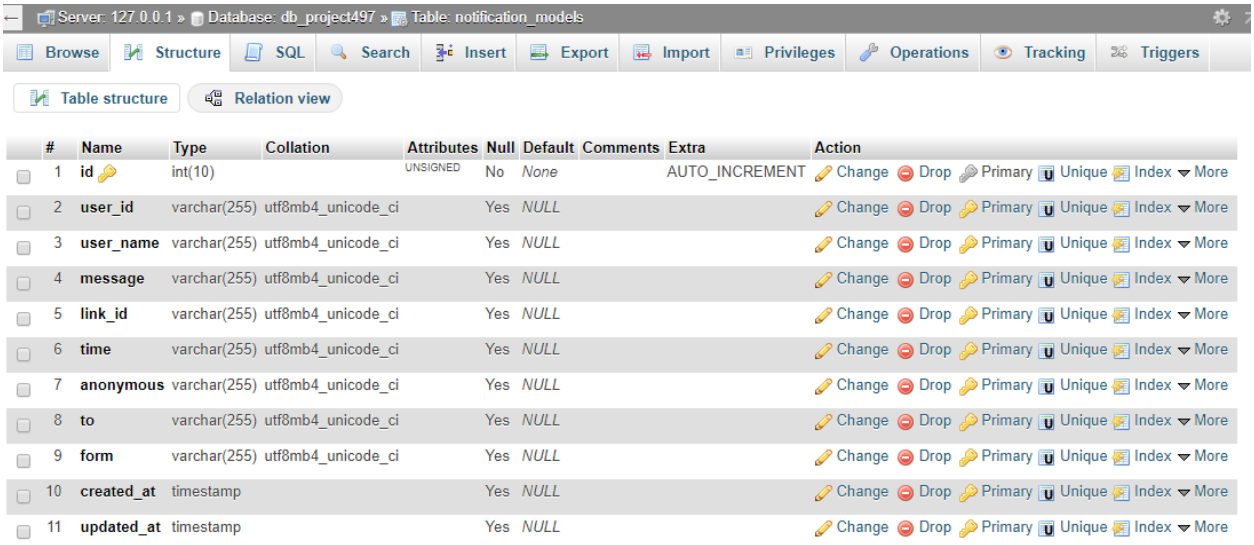


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(10)		UNSIGNED	No	None		AUTO_INCREMENT	Change Drop Primary Unique Index Spatial Fulltext Distinct values More
2	name	varchar(255)	utf8mb4_unicode_ci		No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
3	gender	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
4	phone	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
5	dateOfBirth	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
6	email	varchar(100)	utf8mb4_unicode_ci		No	None			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
7	password	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
8	proPic	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
9	facebook_id	varchar(100)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
10	google_id	varchar(100)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
11	twitter_id	varchar(100)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
12	github_id	varchar(100)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
13	linkedin_id	varchar(100)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
14	statusVerify	tinyint(1)			No	0			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
15	verifyToken	varchar(100)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
16	remember_token	varchar(100)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
17	created_at	timestamp			Yes	NULL			Change Drop Primary Unique Index Spatial Fulltext Distinct values More
18	updated_at	timestamp			Yes	NULL			Change Drop Primary Unique Index Spatial Fulltext Distinct values More

Figure 4.1.14: Table ‘users’

Table Name 14:notification_model

Description: Attributes of table ‘notification_model’



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(10)		UNSIGNED	No	None		AUTO_INCREMENT	Change Drop Primary Unique Index More
2	user_id	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index More
3	user_name	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index More
4	message	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index More
5	link_id	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index More
6	time	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index More
7	anonymous	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index More
8	to	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index More
9	form	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index More
10	created_at	timestamp			Yes	NULL			Change Drop Primary Unique Index More
11	updated_at	timestamp			Yes	NULL			Change Drop Primary Unique Index More

Figure 4.1.15: Table ‘notification_model’

Table Name 15:others_comments

Description: Attributes of table ‘others_comments’

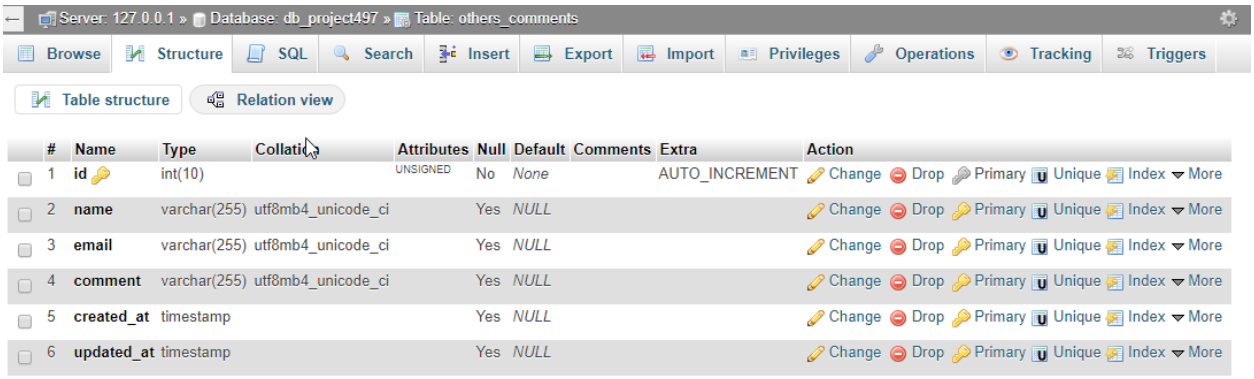
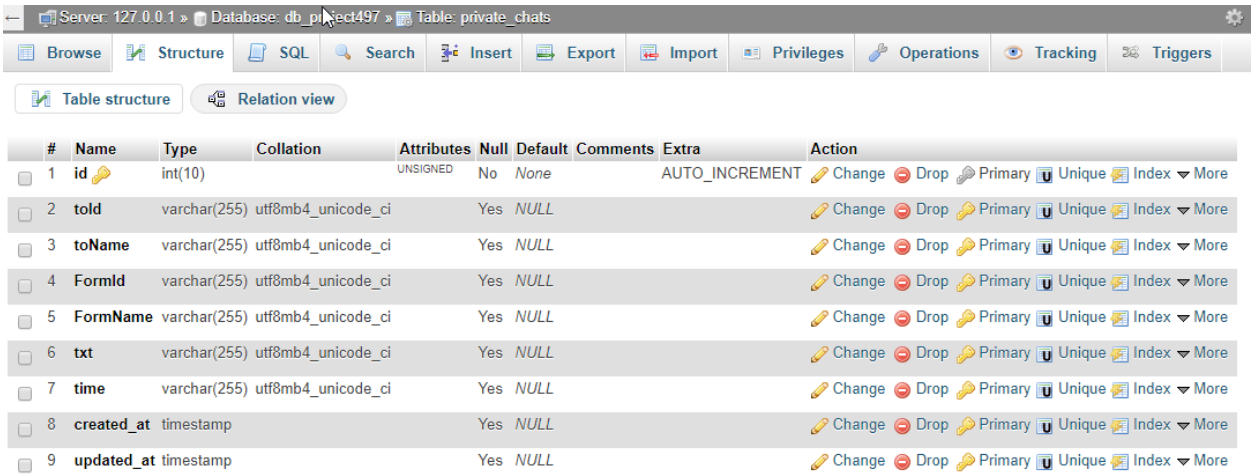


Figure 4.1.16: Table ‘others_comments’

Table Name 16:private_chat

Description: Attributes of table ‘private_chat’



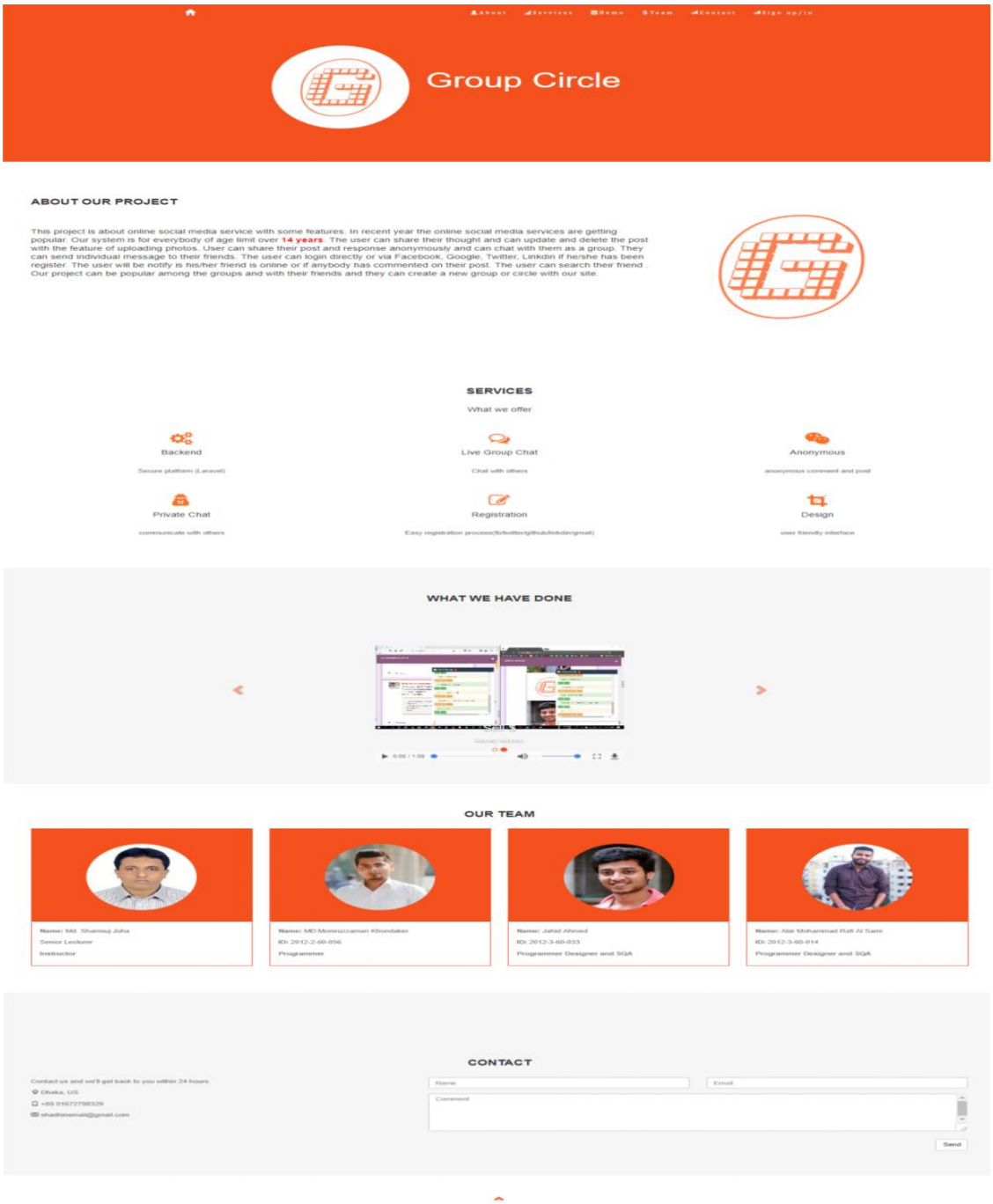
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(10)		UNSIGNED	No	None		AUTO_INCREMENT	Change Drop Primary Unique Index More
2	told	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index More
3	toName	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index More
4	FormId	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index More
5	FormName	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index More
6	txt	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index More
7	time	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop Primary Unique Index More
8	created_at	timestamp			Yes	NULL			Change Drop Primary Unique Index More
9	updated_at	timestamp			Yes	NULL			Change Drop Primary Unique Index More

Figure 4.1.17: Table ‘private_chat’

4.2User Interface

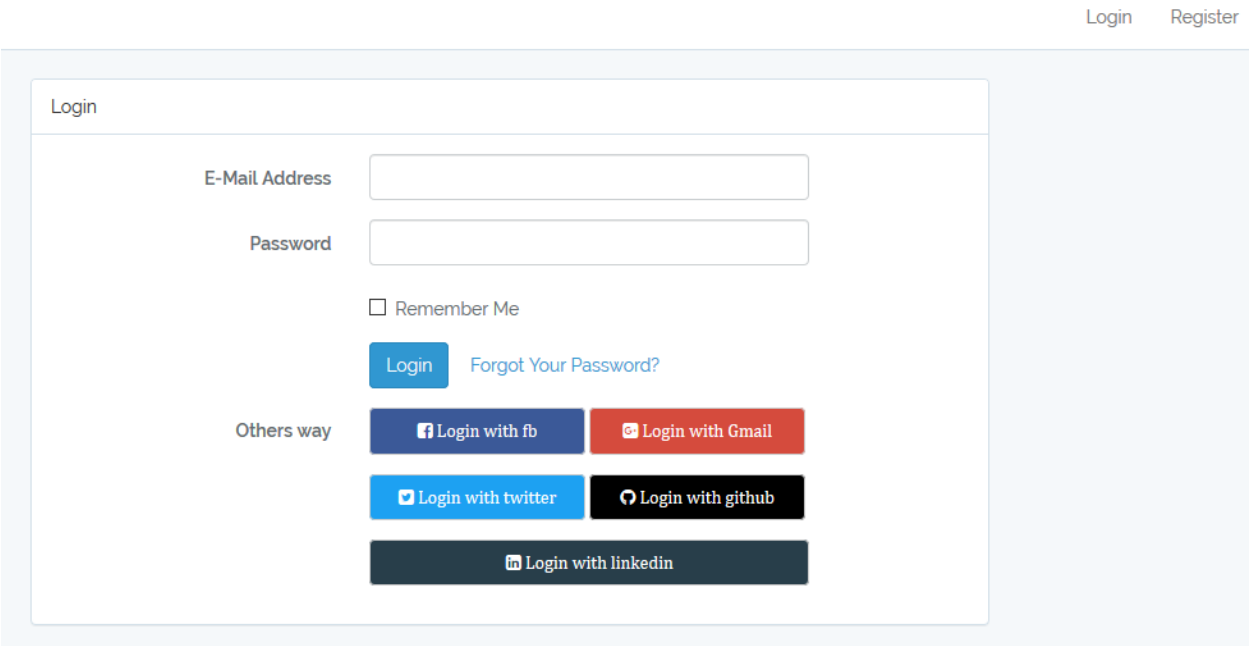
4.2.1 Home Page:

We have introduced one-page design in our homepage. User can know about the website and provide demo view of the website. There is menu for sign-in and sign-up and can contact us and know the details about team.



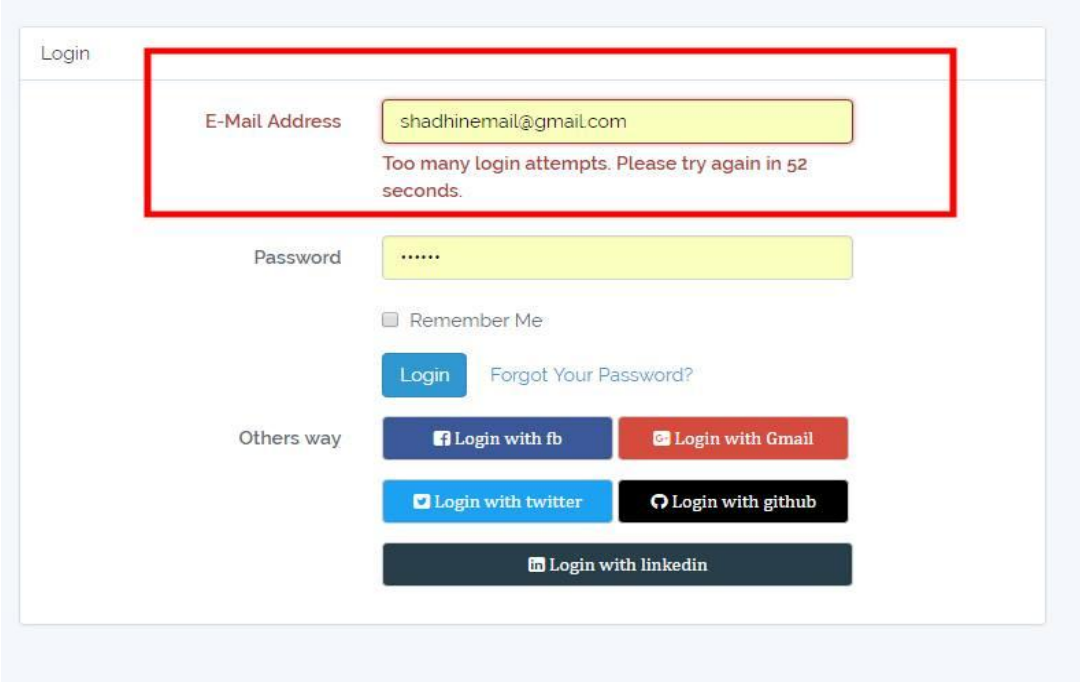
4.2.2 User Login:

The user can log into the system with registered email or the user can directly log into the system with facebook, gmail, twitter, github and linkedin with registration.



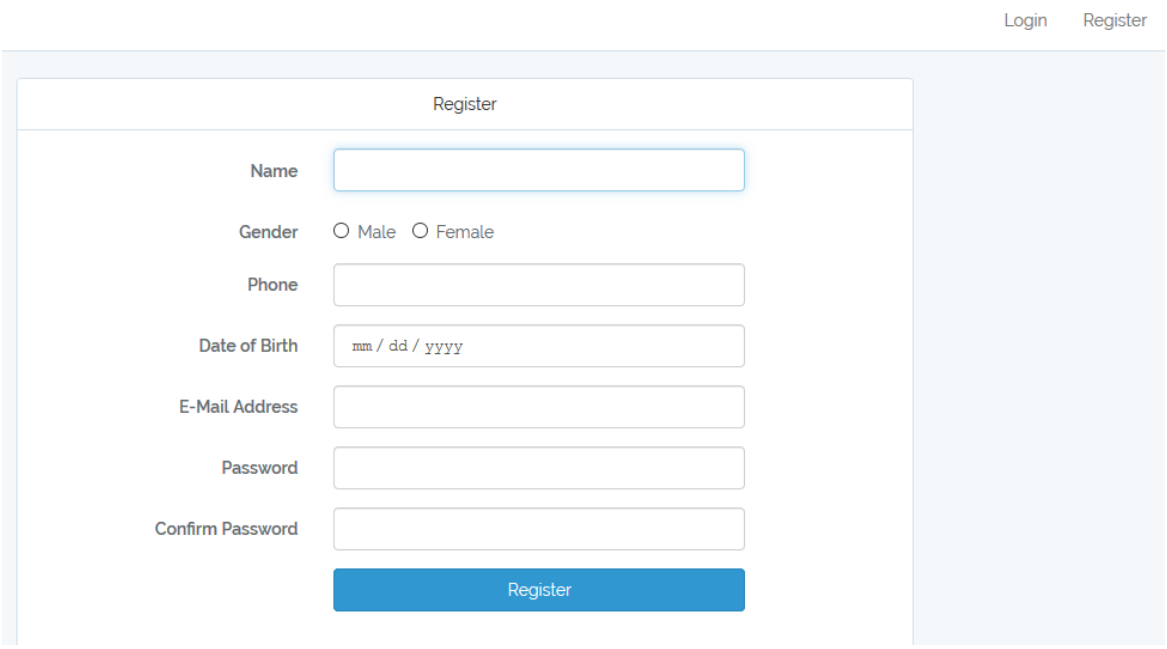
4.2.3 : Wrong Attempt Notification

When it is more than 3 time user have to wait for 1 min.



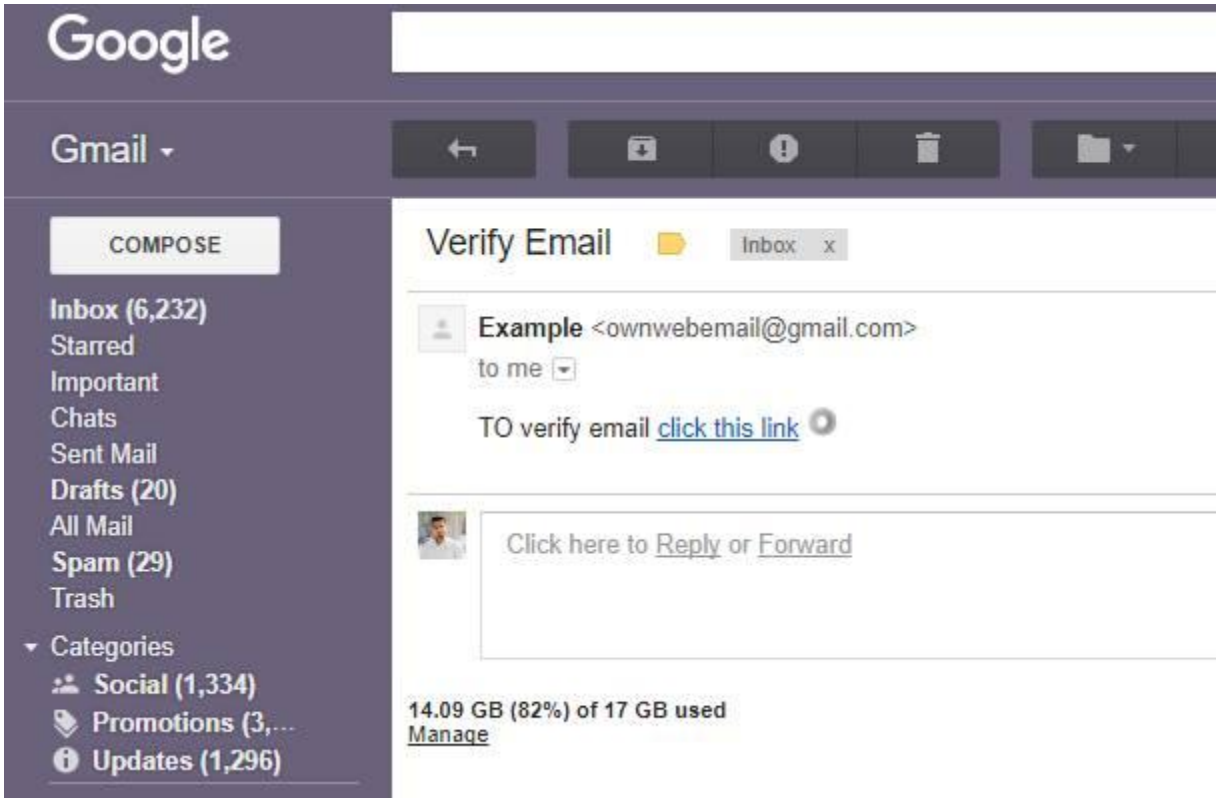
4.2.4 User Registration:

To register into the system the user must input name, gender, phone number, date of birth, email address and password. A verification link will be sent to the email address to verify the user.



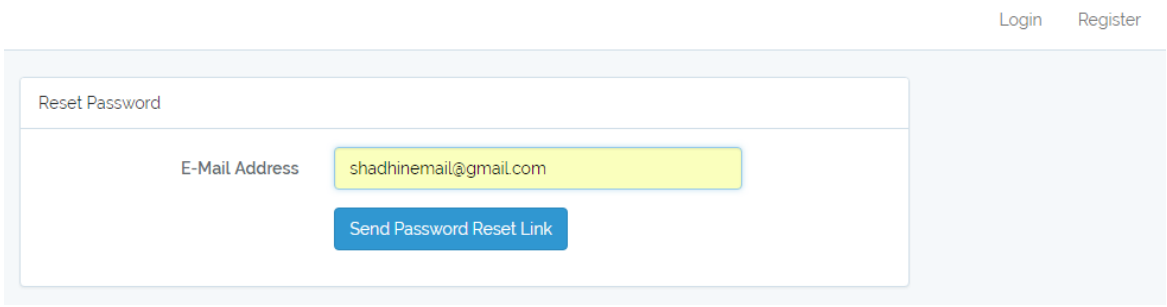
4.2.5 Verify Email:

If the password has been forgotten then the reset password link will be sent to the user email address to reset password.



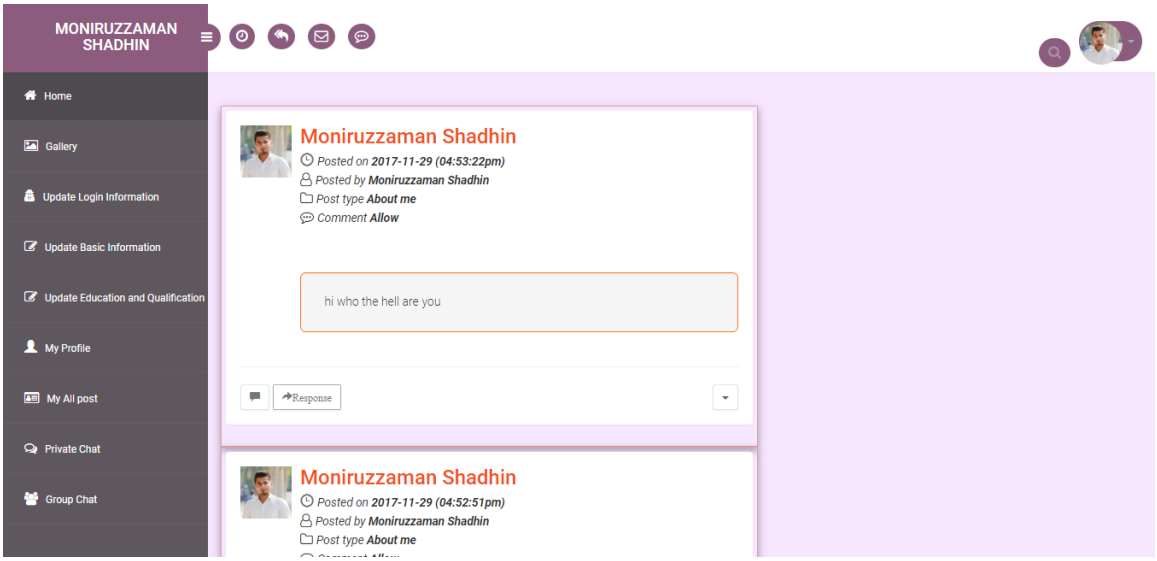
4.2.6 Reset Password:

If the password has been forgotten then the reset password link will be sent to the user email address to reset password.



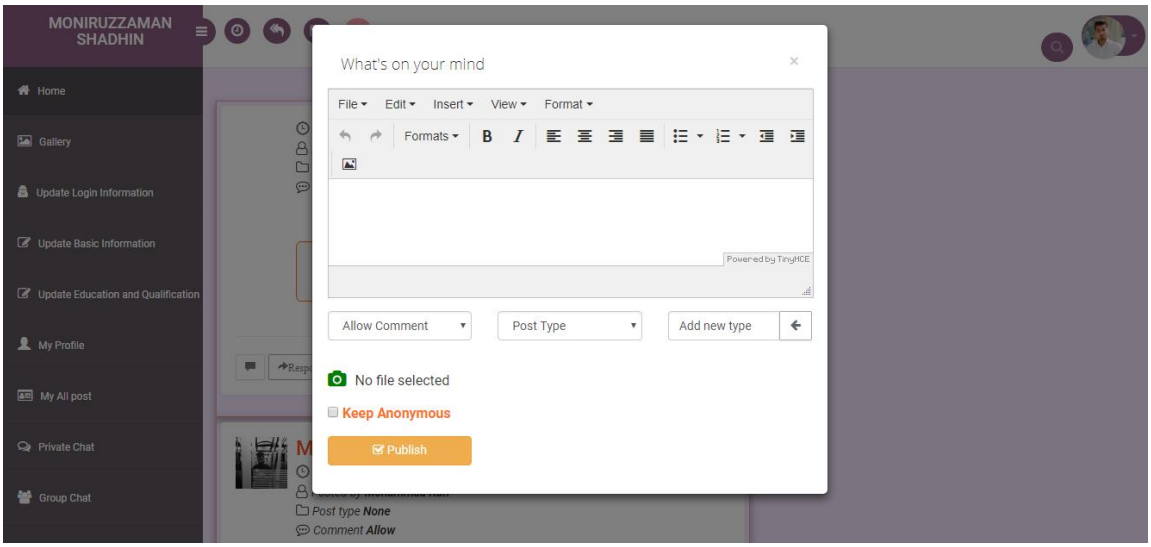
4.2.7 User Home Panel:

Once logged into the system, users can update their basic information, education and qualifications; they can have private chat with friends with and have group chat as well. They can update their status, upload photos and can see other's post and comment on their post.



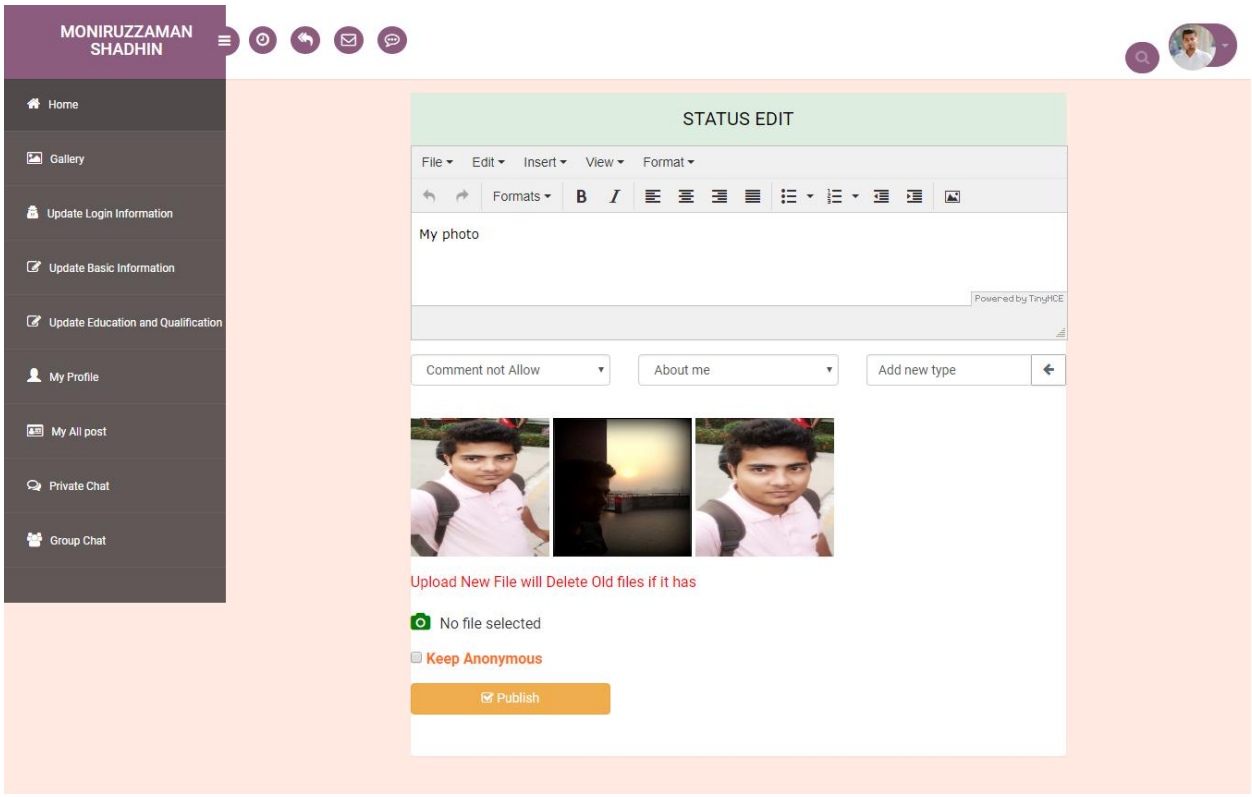
4.2.8 Posting Status:

User can format text and add custom 'post type' and give allowance to comment and upload multiple photos and post anonymously.



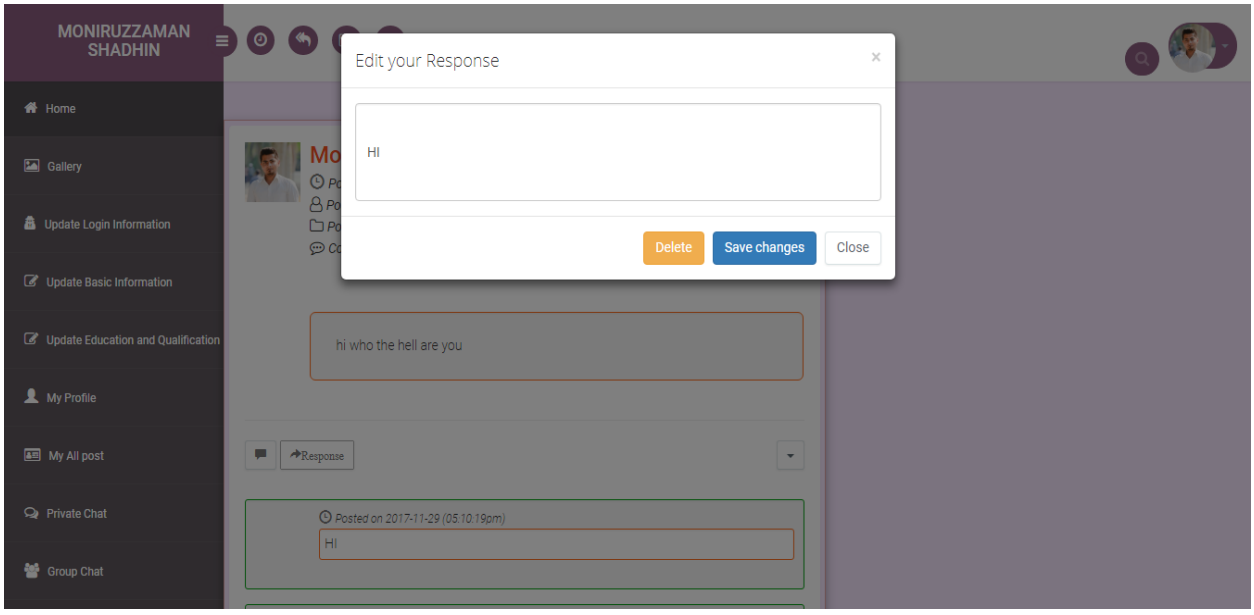
4.2.9 Edit Status:

User can modify statues later.



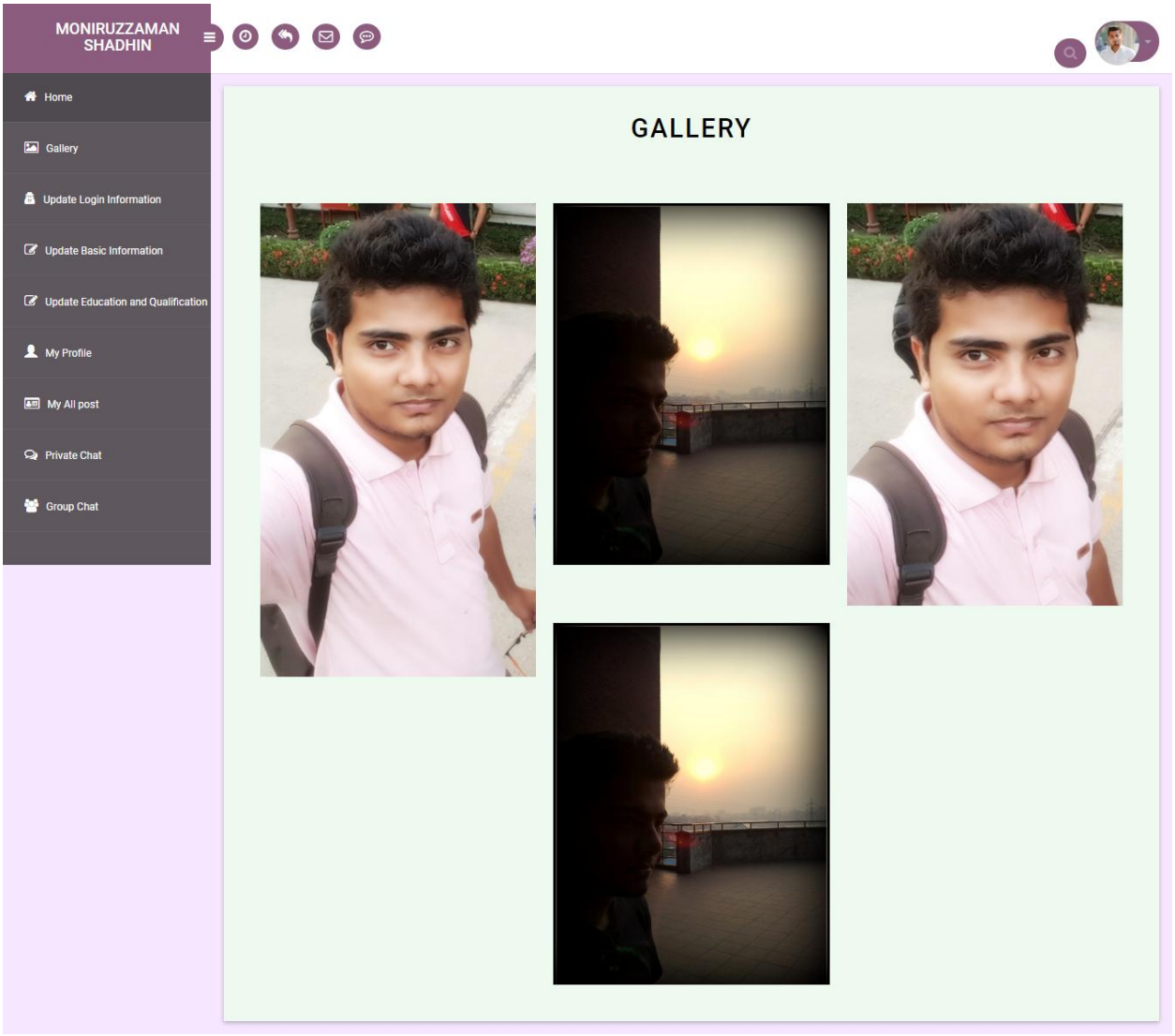
4.2.10 User Response:

User can responses to others post.



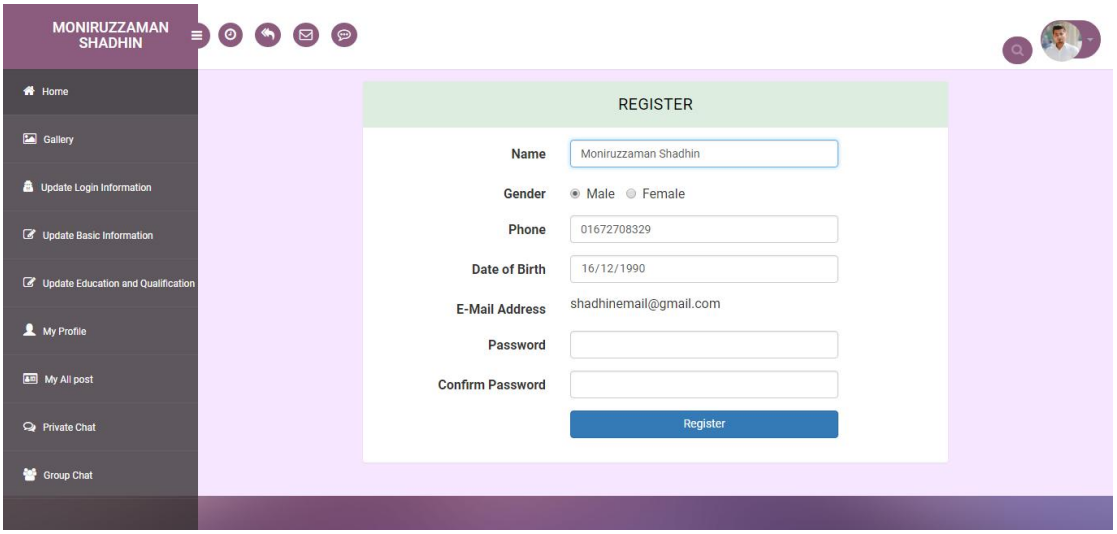
4.2.11 Gallery For Photos:

Photos uploaded by the users are stored in the gallery. The photos are sorted by recent upload.



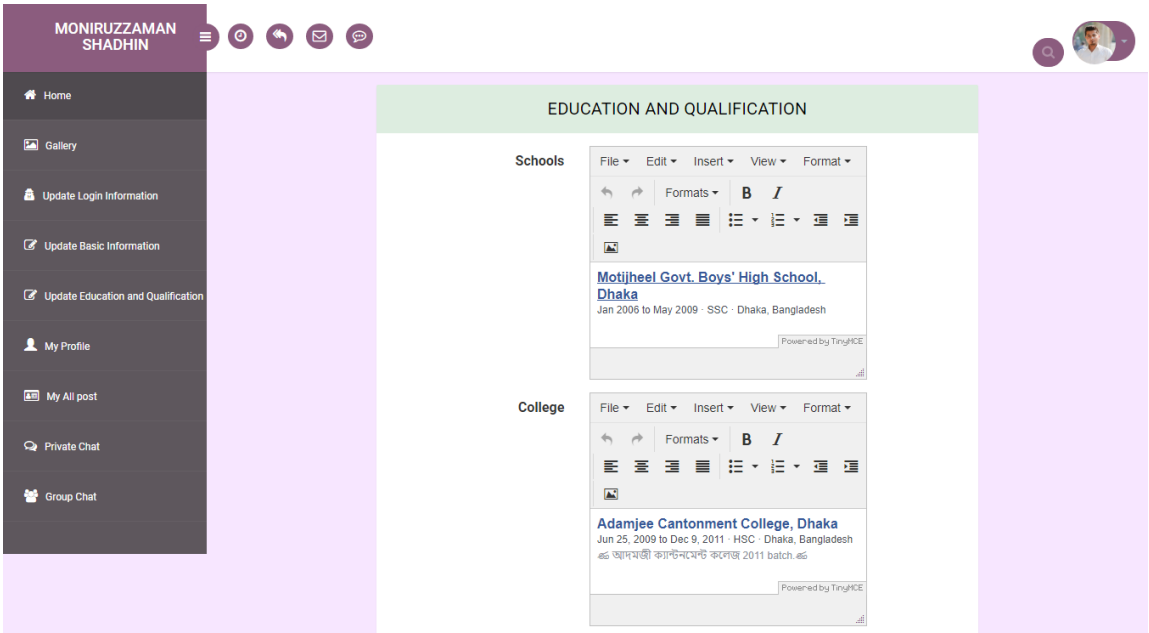
4.2.12Registration Info Setting Panel:

In this interface the user can change their name, gender, phone, date of birth and password except email.



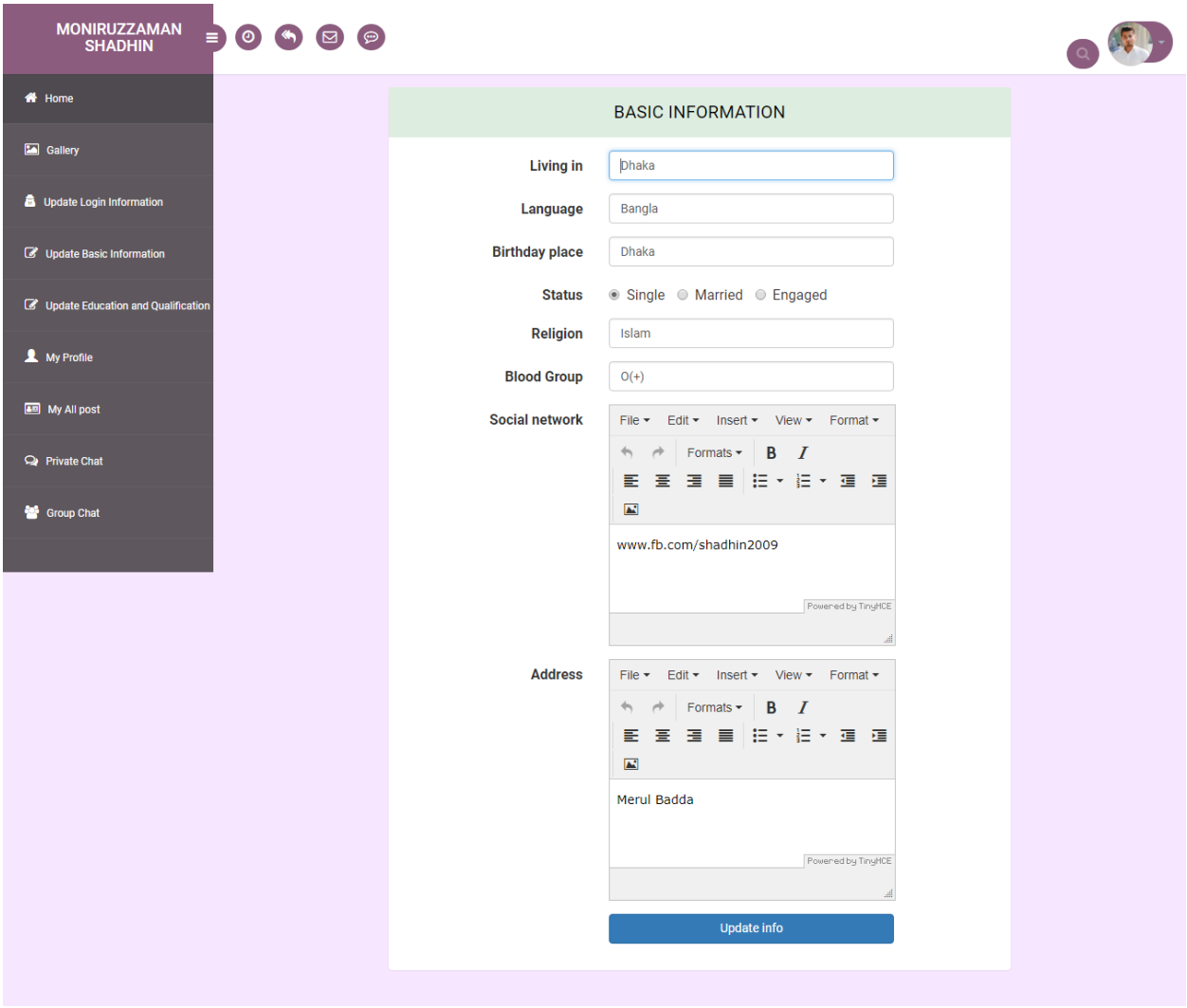
4.2.13Education and Qualification Setting Panel:

In this interface the users can update their education info and qualifications.



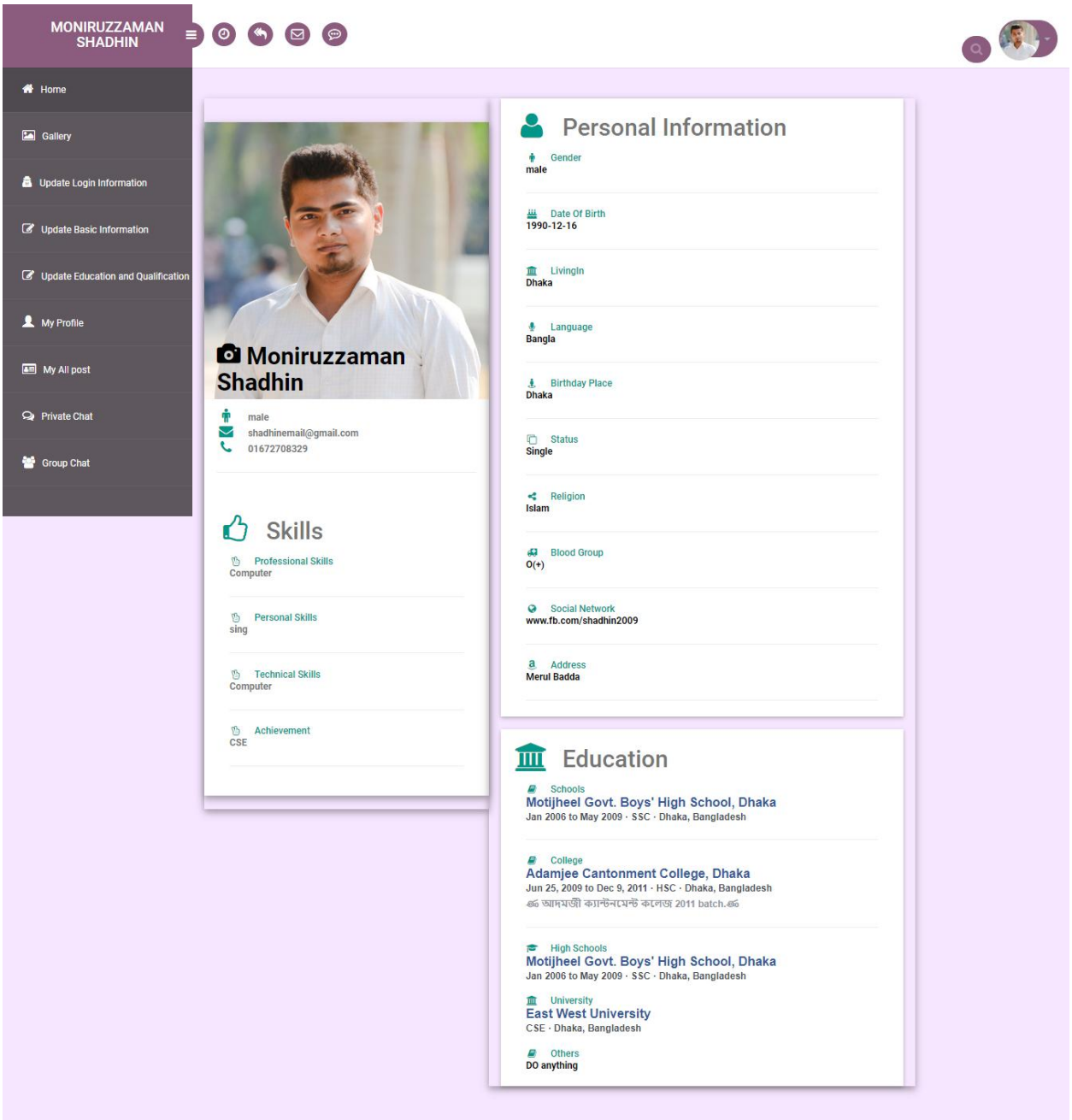
4.2.14Basic Information Setting Panel:

In this interface the users can update their basic information.



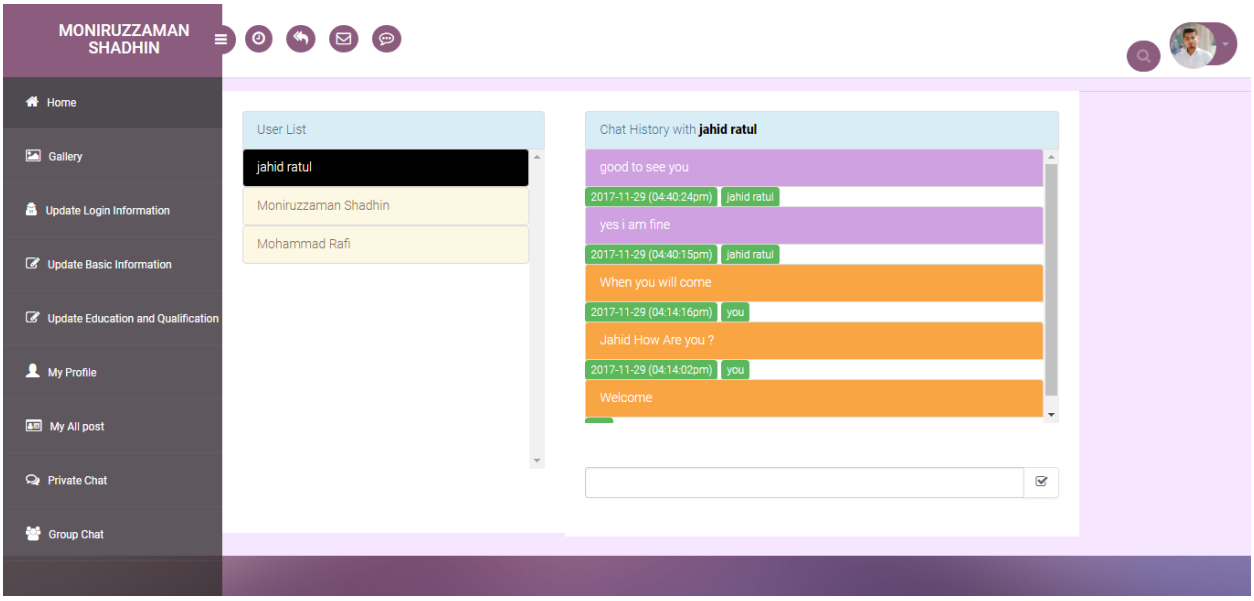
4.2.15 My Profile Panel:

In this interface the users can see their personal information.



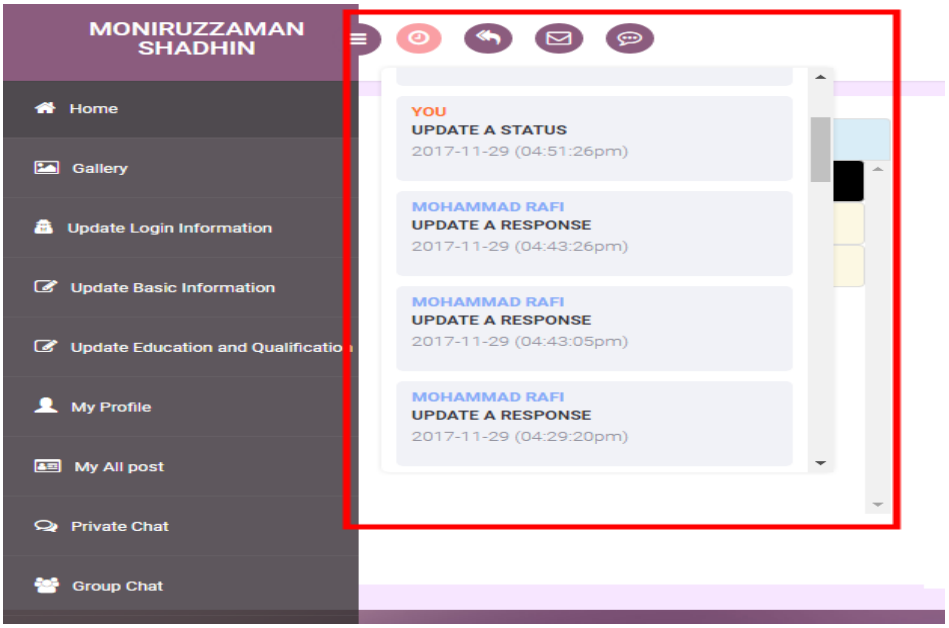
4.2.16Private Chat Interface:

The users can chat privately with their friends.



4.2.17Notification for Activity:

There is a notification alert when user's friend is doing any activities.

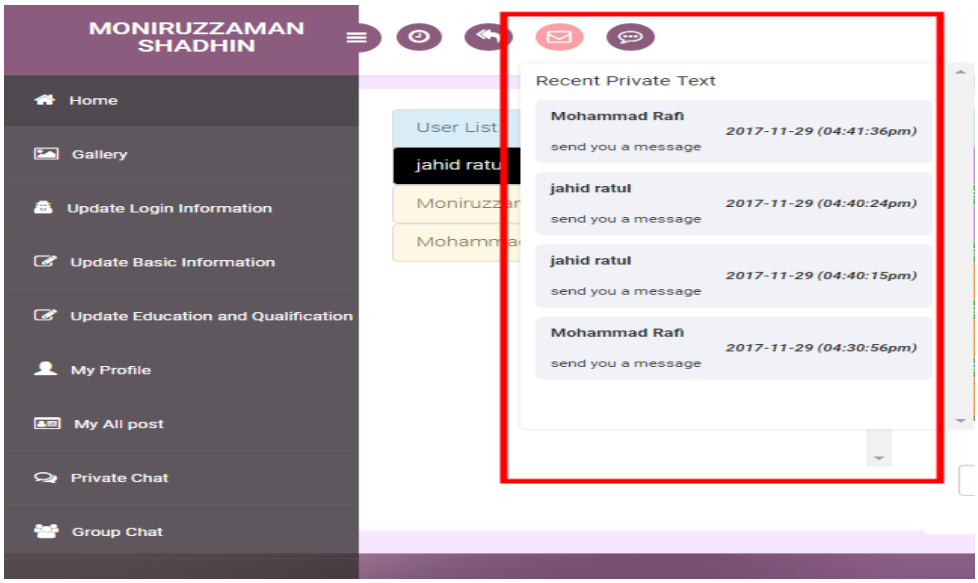


4.2.18My Notification:

The notification alert is for if anybody response in user's post.

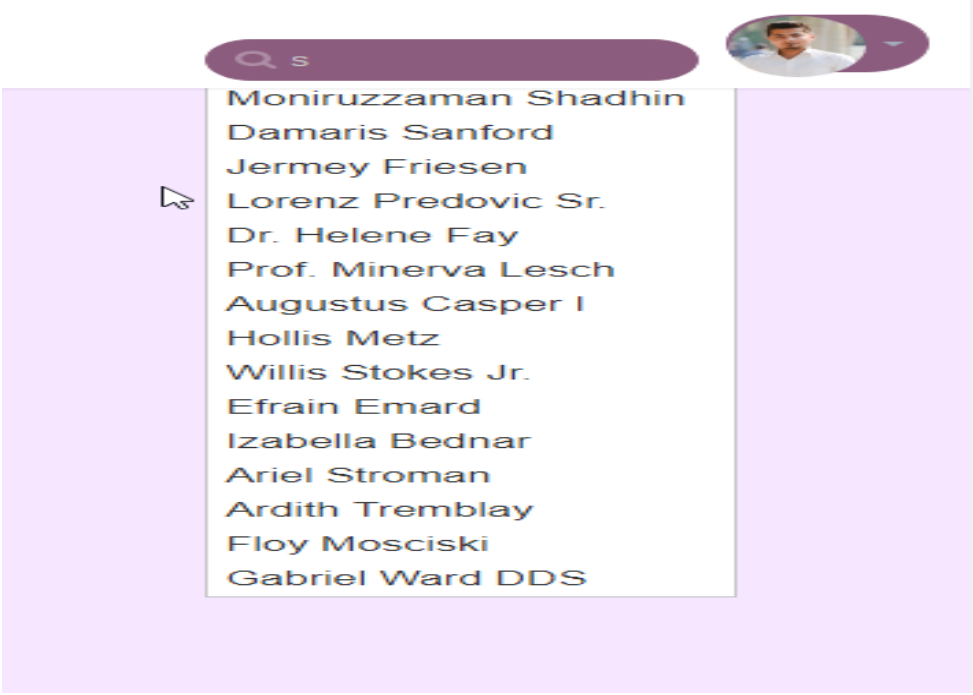
4.2.19 Message Box:

The user is notify for the private message.



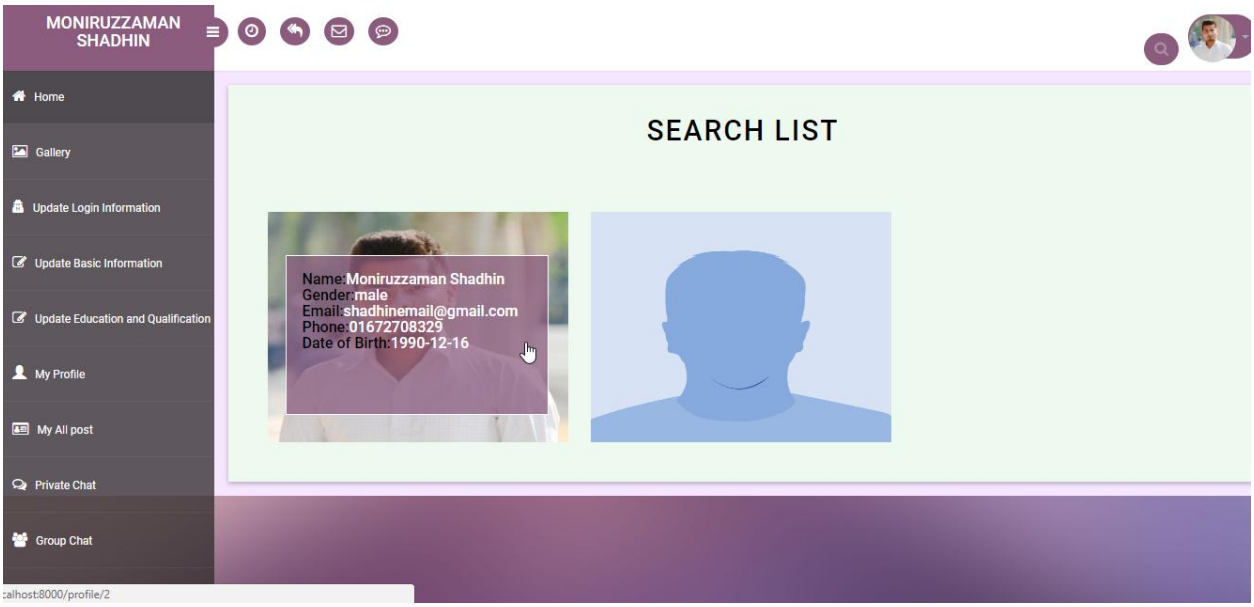
4.2.20Search Option:

In search option, user list is suggested when any character is inserted.



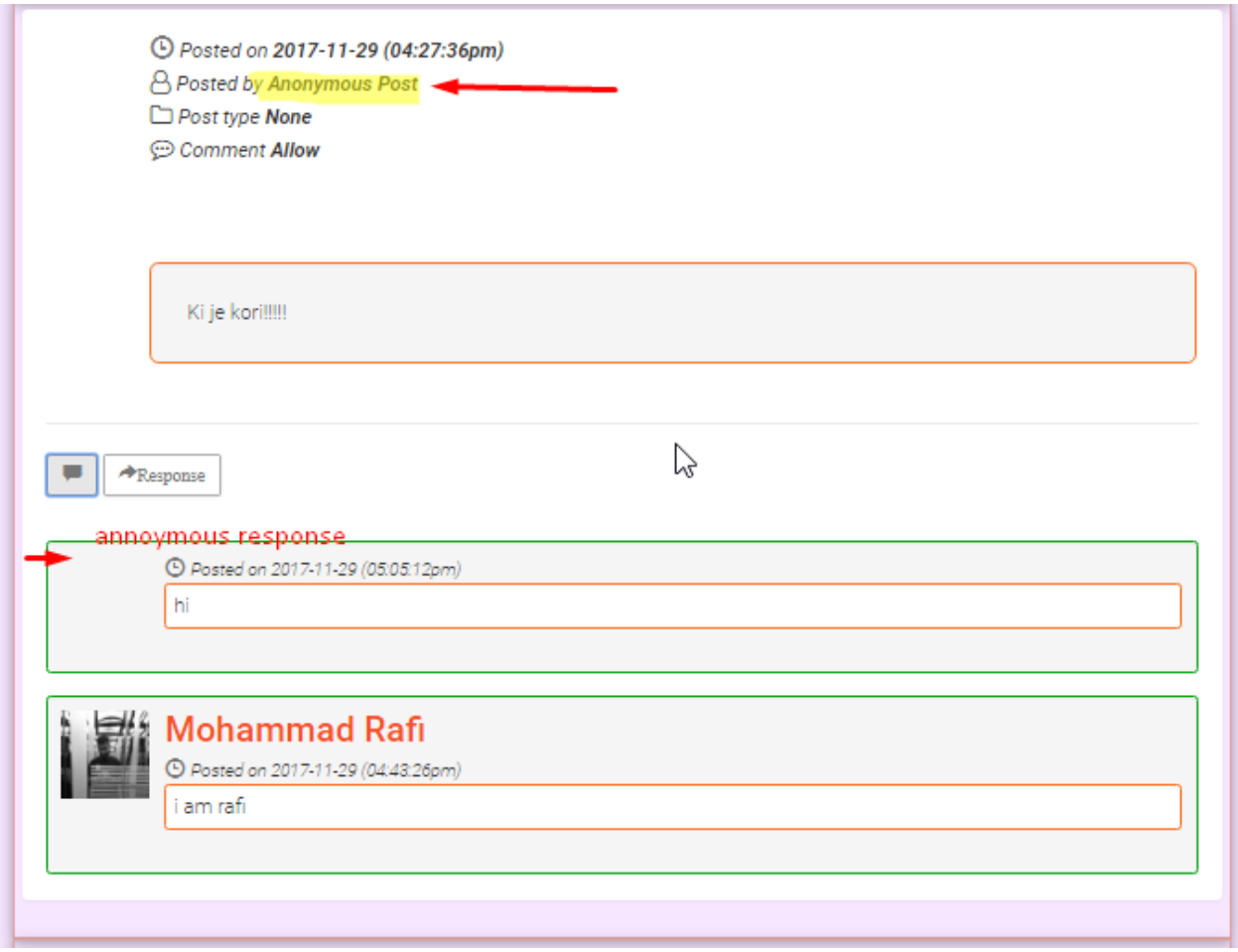
4.2.21Search Result:

After clicking on search list here it comes the search result.



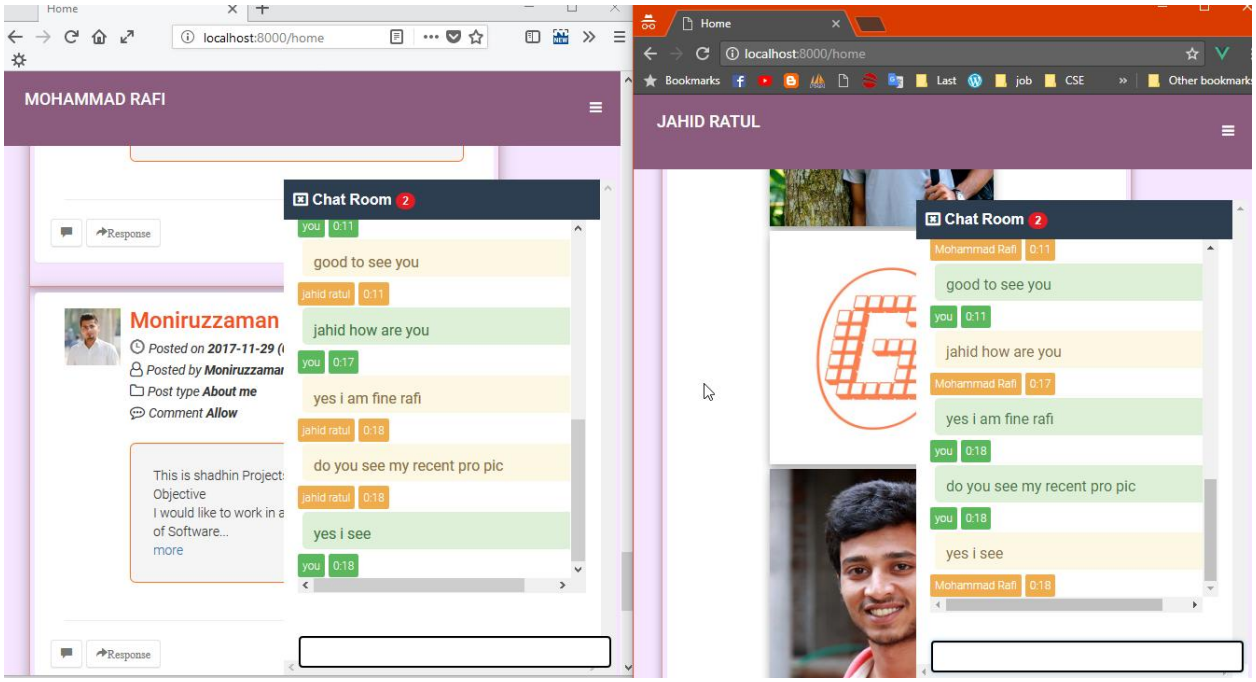
4.2.22 Anonymous Post:

User can post anonymous and identity will be hidden.



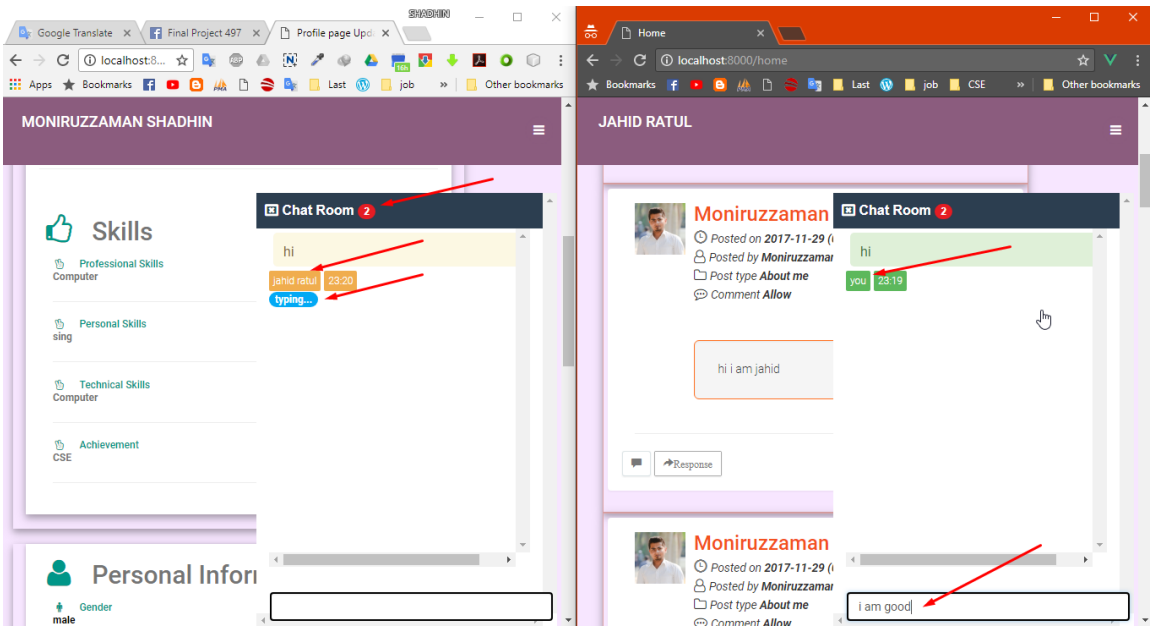
4.2.23 Group Chat:

User can chat with everyone any time.



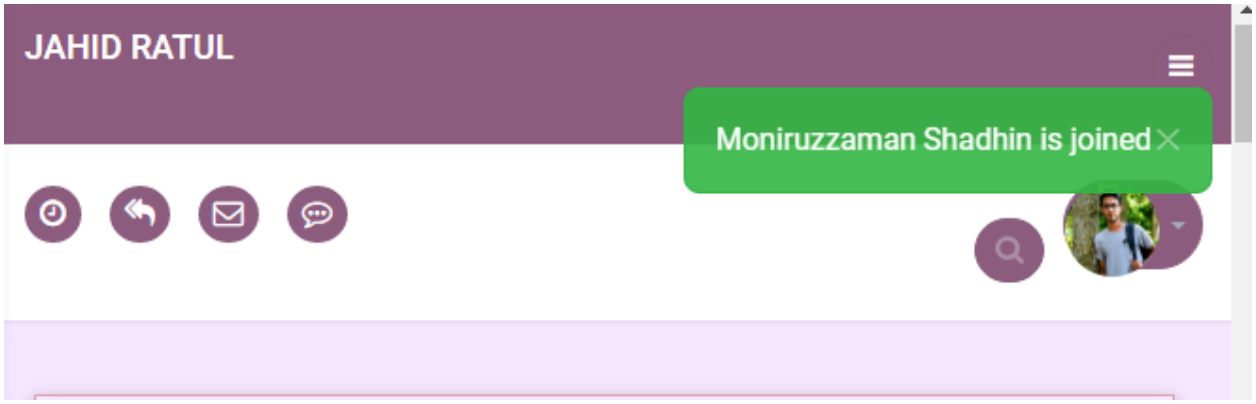
4.2.24 Live Typing Feature

User can see typing in the chat box while someone is writing to him/her.



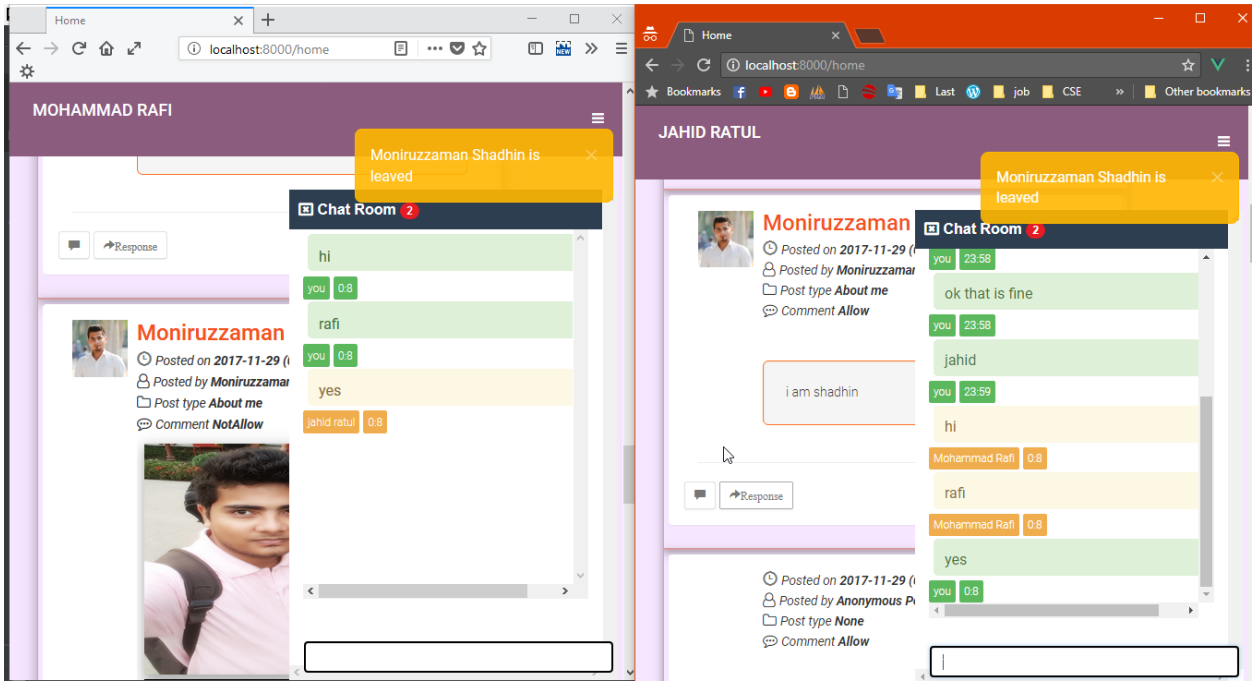
4.2.25 Online Notification

There is a notification alert when other user comes online.



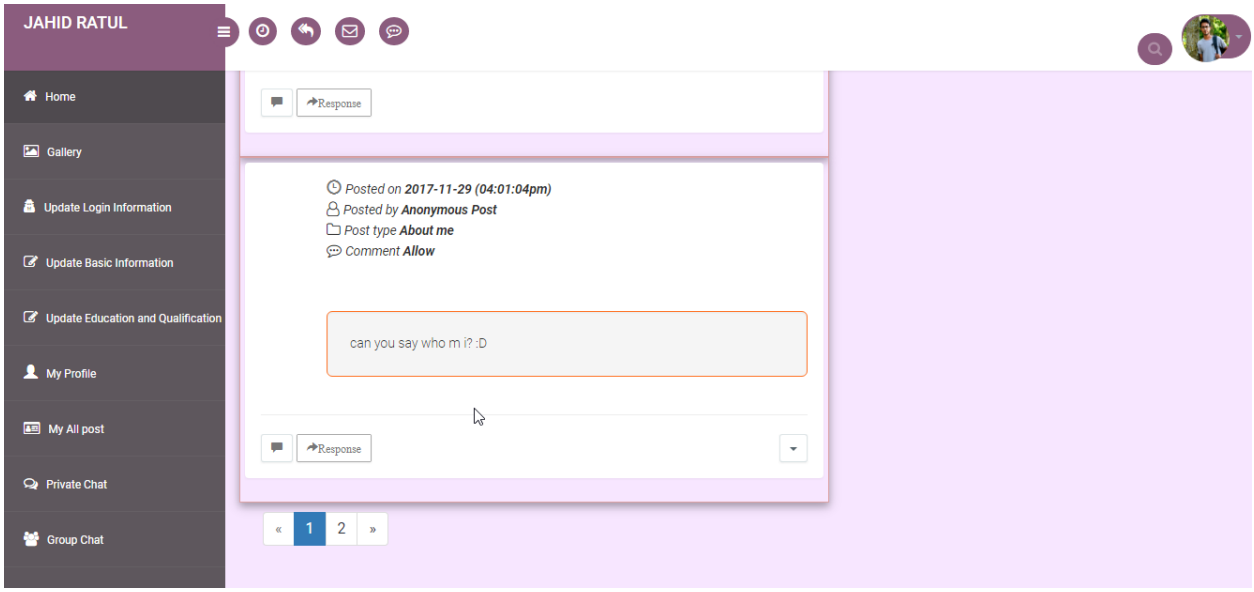
4.2.26 Offline Notification

There is a notification alert when other user goes offline.



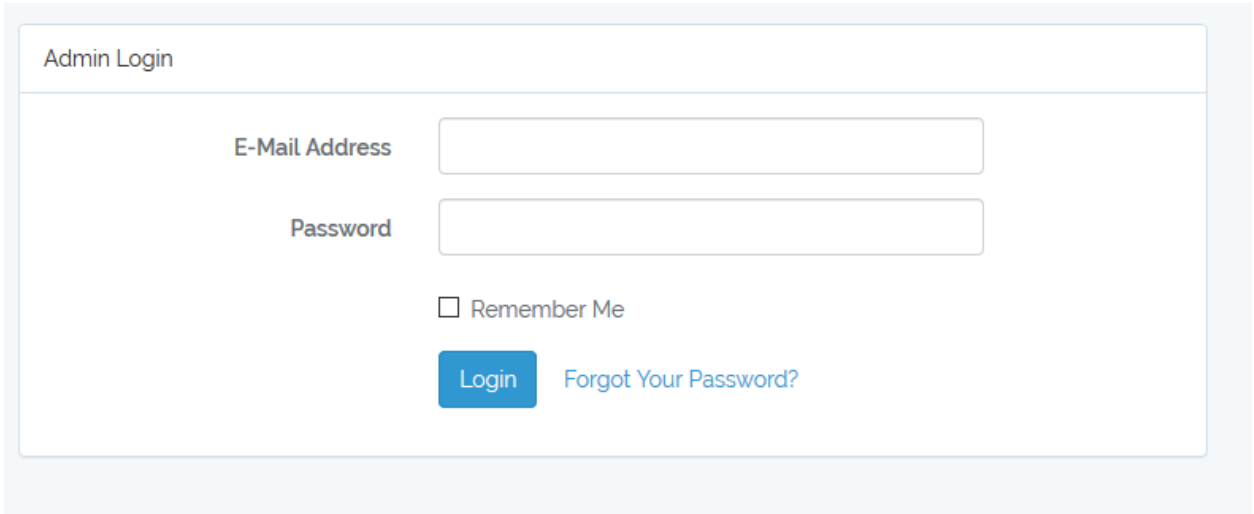
4.2.27 Pagination system

There is a pagination system for better experience.



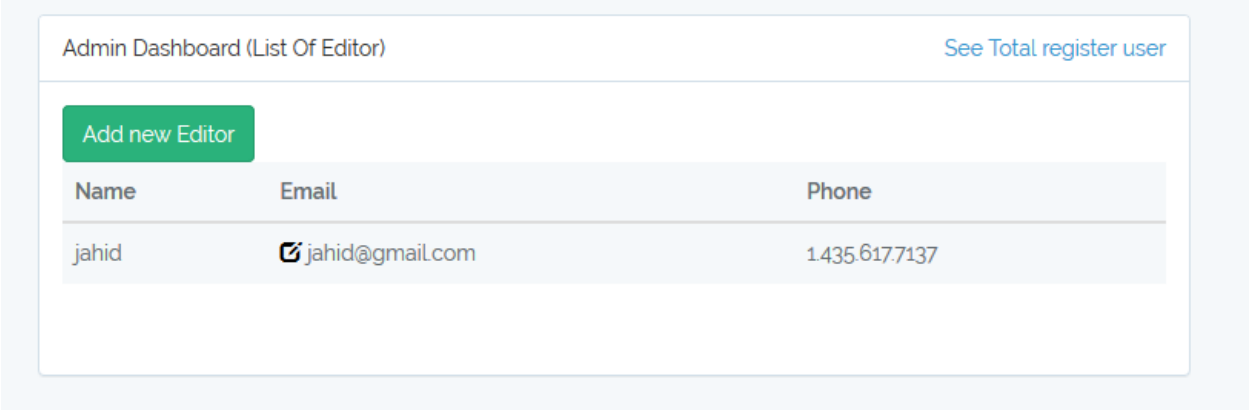
4.2.28 Admin Login:

Admin can directly log into the system.



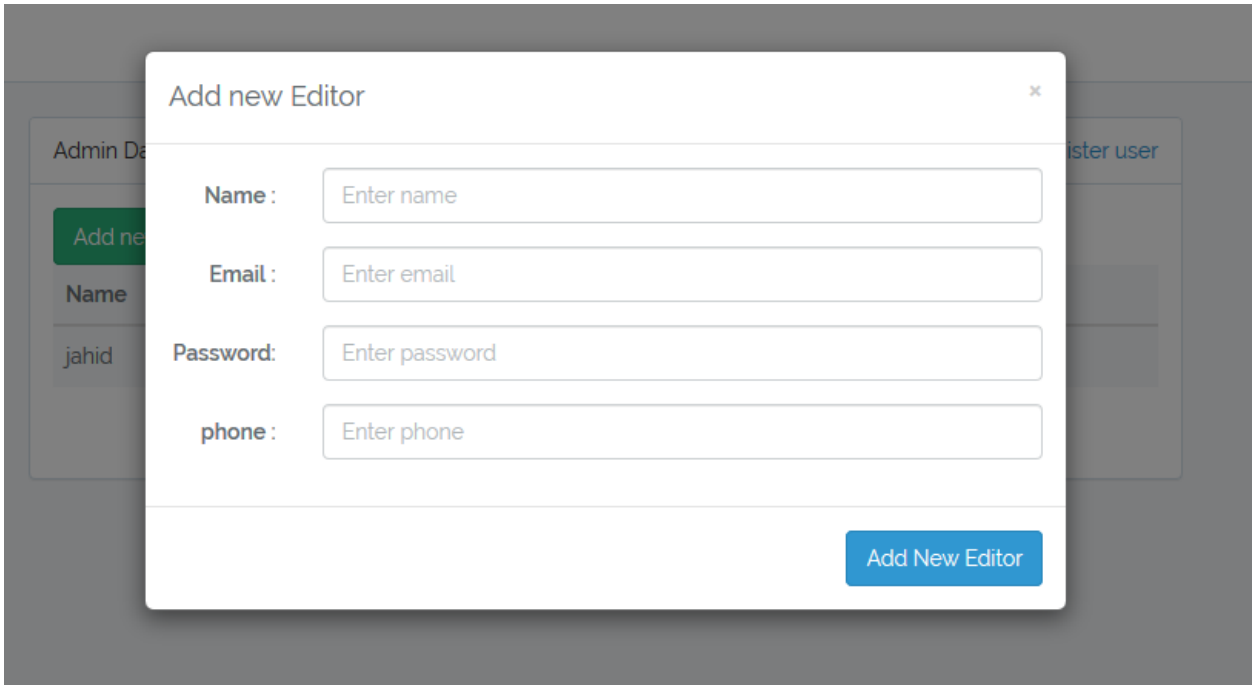
4.2.29 Admin Dashboard:

Admin can see total Editor and User.



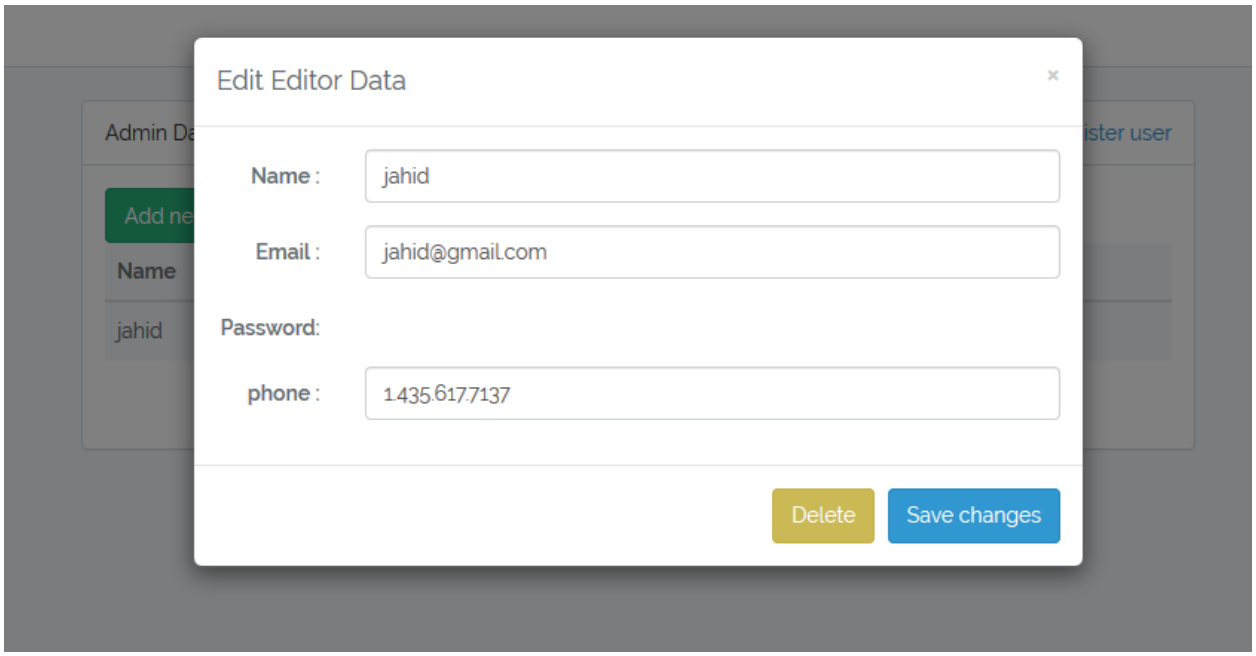
4.2.30 Add Editor:

Admin can add new Editor.



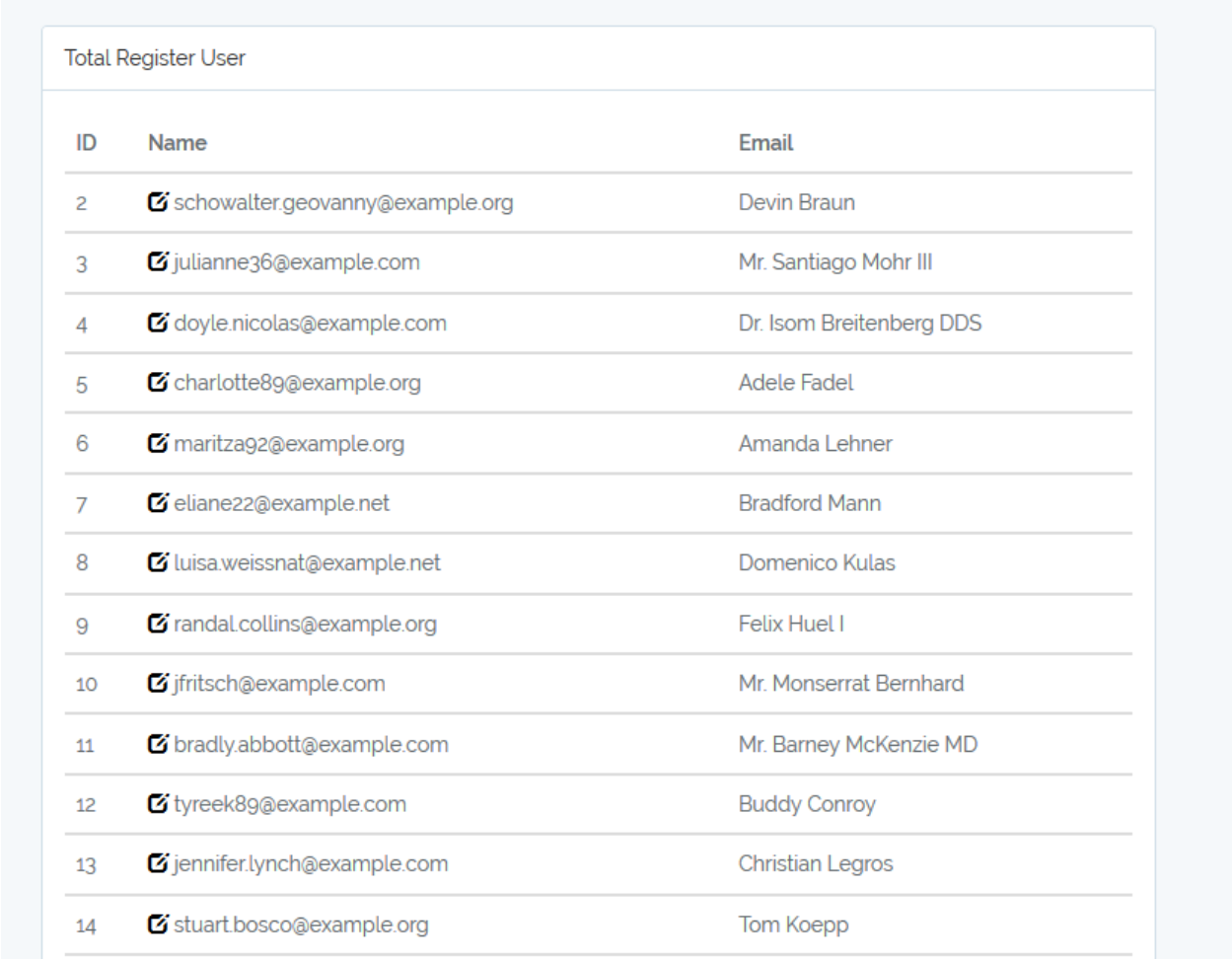
4.2.31 Edit Editor:

Admin can edit existing Editors data.



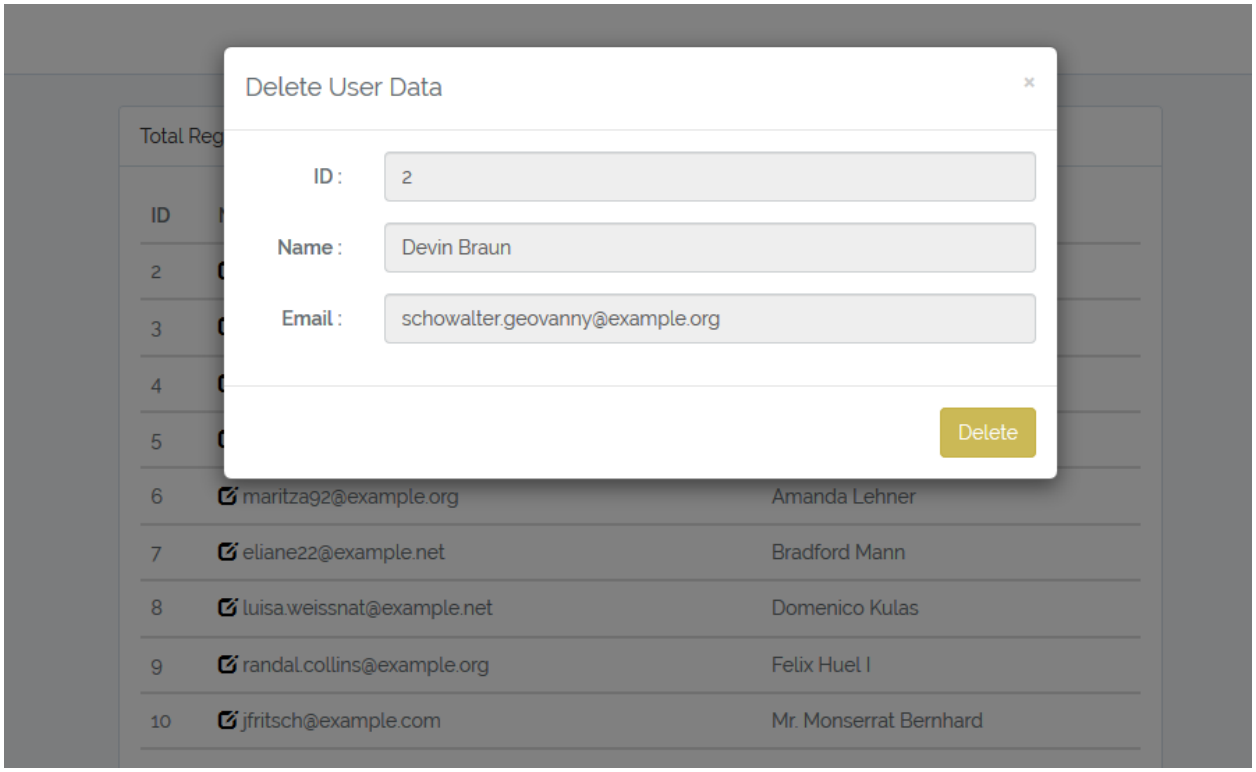
4.2.32 User List:

Total register User for Admin.



4.2.33 Delete User List:

Admin can delete User Data.



4.2.34 Not Found:

The web site hosting server will typically generate a "404 Not Found" web page when a user attempts to follow a broken or dead link.

Chapter 5

Conclusion

5.1 Conclusion

‘**Group Circle**’ is an online social media service where people can communicate with others very easily .This creates a group for friend those who are busy in real life and can’t spend their time with friends which create a communication gap. To revive those old memories of spending time with friends again ‘Group Circle’ is a great and secure platform.

In future we want to further works to develop the system introducing more interesting features and take this far more and making the user interface more interesting and improve the time complexity.

REFERENCES

1. Learn, Share, Build, Each month, over 50 million developers come to Stack Overflow to learn, share their knowledge, and build their careers, http://stackoverflow.com/ (04-07-2017)

2. We believe everyone should have easy, open access to information and that video is a powerful force for education, building understanding, and documenting world events, https://www.youtube.com/(07-07-2017)

3. Fast, flexible and pragmatic, PHP powers everything from your blog to the most popular websites in the world, http://php.net/(11-07-2017)

4. W3Schools is optimized for learning, testing, and training. Examples might be simplified to improve reading and basic understanding, http://www.w3schools.com/ (13-07-2017)

5. PHP is basically used for developing web based software applications. This tutorial helps you to build your base with PHP, http://www.tutorialspoint.com/php/(18-07-2017)

6. Learn practical, modern web development, through expert screen casts. Most video tutorials are boring. These aren't. Ready to binge, https://laracasts.com/(21-07-2017)

7. Love beautiful code? We do too. The PHP Framework for Web Artisans, https://laravel.com/(25-07-2017)

Appendix

Appendix

Source Code:

1. All Route File

```
Route::post('/OutCustomar', 'WelcomeController@OutCustomar');
Route::get('/', 'WelcomeController@welcome');
Route::get('/login', 'HomeIndexController@signInOut');
Auth::routes();
//vendor/laravel/framework/src/illuminate/routing/router.php
Route::get('/home', 'HomeController@index')->name('home');
/*Route::get('/profile/{id}', 'ProfileController@index')-
>middleware('CheckRegMiddleware');*/
Route::get('login/facebook', 'FbLoginControlle@redirectToProvider');
Route::get('login/facebook/callback',
'FbLoginControlle@handleProviderCallback');
Route::get('login/google',
'GmailLoginController@redirectToProvider');
Route::get('login/google/callback',
'GmailLoginController@handleProviderCallback');
Route::get('login/twitter',
'TwitterLoginController@redirectToProvider');
Route::get('login/twitter/callback',
'TwitterLoginController@handleProviderCallback');
Route::get('login/github',
'GitHubLoginController@redirectToProvider');
Route::get('login/github/callback',
'GitHubLoginController@handleProviderCallback');
Route::get('login/linkedin',
'linkedinLoginController@redirectToProvider');
Route::get('login/linkedin/callback',
'linkedinLoginController@handleProviderCallback');
/*Route::get('verifyEmailFirst',
'Auth\RegisterController@verifyEmailFirst')-
>name('verifyEmailFirst');*/
Route::get('verify/{email}/{verifyToken}',
'Auth\RegisterController@sendEmailDone')->name('sendEmailDone');
Route::group(['middleware' => ['CheckRegMiddleware']], function () {
/*Route::get('/timeline','HomeIndexController@homeIndex');*/
Route::get('/postById/{id}', 'HomeIndexController@PostById');
    Route::get('/allpostById/{id}',
'HomeIndexController@allPostById');
    Route::get('/search', 'HomeIndexController@search');
    Route::post('/searchResult',
'HomeIndexController@searchResult');
    Route::post('/pChat', 'ChatController@pChatUp');
```

```
    Route::get('/privateChat/{id}', 'ChatController@privateChat');
    Route::get('/chat', 'ChatController@Index');
    Route::post('/chat/send', 'ChatController@send');
    Route::post('saveToSession', 'ChatController@saveToSession');
    Route::post('getOldMessage', 'ChatController@getOldMessage');
    Route::get('/profile/{id}', 'ProfileController@index');
    Route::get('/gallery/{id}', 'ProfileController@gallery');
    Route::post('/profile/proup', 'ProfileController@proUp');
    Route::get('/update/login/{id}', 'UpdateController@upLogin');
    Route::post('/update/login/save',
'UpdateController@upLoginSave');
    Route::get('/update/basic/{id}', 'UpdateController@upBasic');
    Route::post('/update/save', 'UpdateController@upBasicSave');
    Route::get('/update/education/{id}',
'UpdateController@upEducation');
    Route::post('/update/save2', 'UpdateController@upEduSave');
    Route::post('/status', 'Status_postController@statusUp');
    Route::get('/status/delete/{id}',
'Status_postController@statusDelete');
    Route::get('/status/edit/{id}',
'Status_postController@statusEdit');
    Route::post('/status/update',
'Status_postController@statusUpdate');
    Route::post('/response/add',
'Status_postController@responseCreate');
    Route::post('/response/delete',
'Status_postController@responseDelete');
    Route::post('/response/update',
'Status_postController@responseUpdate');
});
Route::GET('admin/home', 'AdminController@index');
Route::GET('admin/editor', 'EditorController@index');
Route::GET('admin/share', 'EditorController@shareAdmin');
Route::GET('admin', 'Admin\LoginController@showLoginForm')-
>name('admin.login');
Route::POST('admin', 'Admin\LoginController@Login');
Route::POST('admin-password/email',
'Admin\ForgotPasswordController@sendResetLinkEmail')-
>name('admin.password.email');
Route::GET('admin-password/reset',
'Admin\ForgotPasswordController@showLinkRequestForm')-
>name('admin.password.request');
Route::POST('admin-password/reset',
'Admin\ResetPasswordController@reset');
Route::GET('admin-password/reset/{token}',
'Admin\ResetPasswordController@showResetForm')-
>name('admin.password.reset');
Route::POST('admin/add', 'AdminController@addAdmin');
Route::POST('admin/delete', 'AdminController@deleteAdmin');
Route::POST('admin/update', 'AdminController@updateAdmin');
Route::POST('user/delete', 'UpdateController@deleteUser');
```

2.

```php
<?php

namespace App\Http\Controllers;

use App\Admin;
use App\BasicUp;
use App\CategoryList;
use App\EduUp;
use App\imageId;
use App\Response;
use App\role_admin;
useApp\StatasPost;
use App\User;
use Auth;
use DB;
use Illuminate\Http\Request;

class AdminController  extends Controller
{
/**
     * Create a new controller instance.
     *
     * @return void
     */
public function __construct()
    {
$this->middleware('auth:admin');
$this->middleware('admin');
    }

/**
     * Show the application dashboard.
     *
     * @return \Illuminate\Http\Response
     */
public function index()
    {
$adminObj = DB::table('admins')
            ->join('role_admins', 'role_admins.admin_id', '=',
'admins.id')
            ->select('admins.*')
            ->where('role_id',2)
            ->get();


//$adminObj=Admin::all();

return view('admin.home',['adminAll' =>$adminObj]);
    }
public function addAdmin(Request $request){
```

```
$objAdmin = new Admin();
$objAdmin->name=$request->name;
$objAdmin->phone=$request->phone;
$objAdmin->email= $request->email;
$objAdmin->password=bcrypt($request->password);
$objAdmin->statusVerify=1;
$objAdmin->save();
$objAdmin1 = Admin::where('email',$request->email)->first();

$roleObj = new role_admin();
$roleObj->role_id=2;
$roleObj->admin_id=$objAdmin1->id;
$roleObj->save();
return 'Done';

    }
public function deleteAdmin(Request $request){
        Admin::where('id',$request->id)->delete();
        role_admin::where('admin_id',$request->id)->delete();

    }
public function updateAdmin(Request $request){
$adminObj = Admin::find($request->id);
$adminObj->name=$request->name;
$adminObj->email=$request->email;
$adminObj->phone=$request->phone;;
$adminObj->save();
    }

}
```

3.
```
<?php

namespace App\Http\Controllers;

use App\Events\ChatEvent;
use App\NotificationModel;
use App\PrivateChat;
use App\User;
use DB;

use function GuzzleHttp\Promise\all;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

class ChatController extends Controller
{
public function __construct()
    {
$this->middleware('auth');
```

```
    }

public function Index(){

return view('newPage.chat');
    }
public function chat(){
return view('newPage.chat');
    }
/*public function send(){
        $message='ami k';
        $user =User::find(Auth::id());
        event(new ChatEvent($message,$user));
    }*/
public function send(Request $request){
//return $request->all();
$user =User::find(Auth::id());
$this->saveToSession($request);

        event(new ChatEvent($request->message,$user));
    }
public function saveToSession(request $request)
    {
        session()->put('chat',$request->chat);
    }
public function getOldMessage()
    {
return session('chat');
    }
public function privateChat($id){

//$u= Auth::user()->id;
        //$chatData=PrivateChat::where('toId',$this->id)->get();
        //$chatData=PrivateChat::where('toId',$id)->get();
$userName=User::where('id',$id)->first();
$chatData= DB::table('private_chats')
            ->where('toId', '=',$id)
            ->where('FormId', '=',Auth::user()->id)
            ->orWhere('toId', '=',Auth::user()->id)
            ->Where('FormId', '=',$id)
            ->select('private_chats.*')
            ->orderBy('id', 'desc')
            ->get();
//return $chatData->all();

if(count($chatData)==0){
$userObject1=User::where('id',$id)->first();
$userObject2=User::where('id',Auth::user()->id)->first();
$cEnter=new PrivateChat();
$cEnter->toId=$userObject1->id;
$cEnter->toName=$userObject1->name;
$cEnter->FormId=$userObject2->id;
$cEnter->FormName=$userObject2->name;
$cEnter->txt='Welcome';
```

```
$cEnter->time='';
$cEnter->save();

$chatData= DB::table('private_chats')
                ->where('toId', '=',$id)
                ->where('FormId', '=',Auth::user()->id)
                ->orWhere('toId', '=',Auth::user()->id)
                ->Where('FormId', '=',$id)
                ->select('private_chats.*')
                ->orderBy('id', 'desc')
                ->get();

$toUser=User::where('id',$id)->first();
$userAll=User::all();
return
view('frontEnd.home.chat',['userAll'=>$userAll,'userName'=>$userName
,'toUser'=>$toUser,'chatData'=>$chatData,]);
        }
else{
$toUser=User::where('id',$id)->first();

$userAll=User::all();
return
view('frontEnd.home.chat',['userAll'=>$userAll,'userName'=>$userName
,'toUser'=>$toUser,'chatData'=>$chatData,]);

        }



    }
public function pChatUp(Request $request){
$Pchat=new PrivateChat();
$Pchat->toId=$request->toText;
$Pchat->toName=$request->toName;
$Pchat->FormName=$request->FormName;
$Pchat->FormId=$request->formText;
$Pchat->txt=$request->pCText;
$Pchat->time=date("Y-m-d (h:i:sa)");

$notificationObj=new NotificationModel();
$notificationObj->user_id=$request->toText;
$notificationObj->user_name=$request->FormName;
$notificationObj->message=' send you a message ';
$notificationObj->link_id=$request->formText;
$notificationObj->time=$Pchat->time;
$notificationObj->save();

$Pchat->save();

    }


}
```

4.
```php
<?php

namespace App\Http\Controllers;

use Illuminate\Foundation\Bus\DispatchesJobs;
use Illuminate\Routing\Controller as BaseController;
use Illuminate\Foundation\Validation\ValidatesRequests;
use Illuminate\Foundation\Auth\Access\AuthorizesRequests;

class Controller extends BaseController
{
use AuthorizesRequests, DispatchesJobs, ValidatesRequests;
}
```

5.
```php
<?php

namespace App\Http\Controllers;

use App\BasicUp;
use App\CategoryList;
use App\EduUp;
use App\Response;
use App\StatasPost;
use App\User;
use Illuminate\Http\Request;

class EditorController extends Controller
{    /**
 * Create a new controller instance.
 *
 * @return void
 */
public function __construct()
    {
$this->middleware('auth:admin');
$this->middleware('editor',['except'=>'shareAdmin']);
    }
public function index(){

return view('admin.editor');
    }
public function shareAdmin(){

$userObj=User::all();
return view('admin.share',['userAll'=>$userObj]);

    }
```

```
}
```

6. `<?php`

```
namespace App\Http\Controllers;
use Auth;
use App\User;
use Illuminate\Http\Request;
use Socialite;
class FbLoginControlle extends Controller
{
public function redirectToProvider()
    {
return Socialite::driver('facebook')->redirect();
    }
public function handleProviderCallback()
    {
try{
$socialUser = Socialite::driver('facebook')->user();
$user=User::where('facebook_id',$socialUser->getId())->first();
$user2=User::where('email',$socialUser->getEmail())->first();
/* $a=(string)$socialUser->getEmail();*/
if($user2==null &&$user==null){
$userObj = new User();
$userObj->facebook_id=$socialUser->getId();
$userObj->name=$socialUser->getName();
$userObj->email=$socialUser->getEmail();
$userObj->gender='';
$userObj->phone='';
$userObj->statusVerify='1';
$userObj->dateOfBirth='';
$userObj->password=bcrypt(123456);
$userObj->save();
                Auth::login($userObj);
return redirect()->intended('/home');
            }
/*elseif ($socialUser->getEmail().toString()==''){
                $userObj = new User();
                $userObj->facebook_id=$socialUser->getId();
                $userObj->name=$socialUser->getName();
                $userObj->email=$socialUser->getId().'@ownNote.com';
                $userObj->gender='';
                $userObj->phone='';
                $userObj->dateOfBirth='';
                $userObj->password=bcrypt(123456);
                $userObj->save();
                Auth::login($userObj);
                return redirect()->intended('/home');
            }*/
else{
```

```
                Auth::login($user2);
return redirect()->intended('/home');
            }

        }
catch (\Exception $e){
//return dd($socialUser);
return redirect('/login');
        }
//return dd($user);

}
}
```

7. `<?php`

```
namespace App\Http\Controllers;
use Auth;
use App\User;
use Illuminate\Http\Request;
use Socialite;
class GitHubLoginController extends Controller
{
public function redirectToProvider()
    {
return Socialite::driver('github')->redirect();
    }

/**
     * Obtain the user information from GitHub.
     *
     * @return Response
     */

public function handleProviderCallback()
    {
try{
$socialUser = Socialite::driver('github')->user();
$user=User::where('github_id',$socialUser->getId())->first();
$user2=User::where('email',$socialUser->getEmail())->first();
/* $a=(string)$socialUser->getEmail();*/
if($user2==null &&$user==null){
$userObj = new User();
$userObj->github_id=$socialUser->getId();
$userObj->name=$socialUser->getName();
$userObj->email=$socialUser->getEmail();
$userObj->gender='';
$userObj->phone='';
$userObj->statusVerify='1';;
$userObj->dateOfBirth='';
$userObj->password=bcrypt(123456);
$userObj->save();
```

```
                Auth::login($userObj);
return redirect()->intended('/home');
            }
/*elseif ($socialUser->getEmail().toString()==''){
                $userObj = new User();
                $userObj->facebook_id=$socialUser->getId();
                $userObj->name=$socialUser->getName();
                $userObj->email=$socialUser->getId().'@ownNote.com';
                $userObj->gender='';
                $userObj->phone='';
                $userObj->dateOfBirth='';
                $userObj->password=bcrypt(123456);
                $userObj->save();
                Auth::login($userObj);
                return redirect()->intended('/home');
            }*/
else{
                Auth::login($user2);
return redirect()->intended('/home');
            }

        }
catch (\Exception $e){
// return dd($socialUser);
return redirect('/login');
        }
//return dd($user);

}
}
```

8.
```
<?php

namespace App\Http\Controllers;
use Auth;
use App\User;
use Illuminate\Http\Request;
use Socialite;
class GmailLoginController extends Controller
{
public function redirectToProvider()
    {
return Socialite::driver('google')->redirect();
    }

/**
     * Obtain the user information from GitHub.
     *
     * @return Response
     */

public function handleProviderCallback()
```

```
    {
try{
$socialUser = Socialite::driver('google')->stateless()->user();
$user=User::where('google_id',$socialUser->getId())->first();
$user2=User::where('email',$socialUser->getEmail())->first();
/* $a=(string)$socialUser->getEmail();*/
if($user2==null &&$user==null){
$userObj = new User();
$userObj->google_id=$socialUser->getId();
$userObj->name=$socialUser->getName();
$userObj->email=$socialUser->getEmail();
$userObj->gender='';
$userObj->phone='';
$userObj->statusVerify='1';;
$userObj->dateOfBirth='';
$userObj->password=bcrypt(123456);
$userObj->save();
                Auth::login($userObj);
return redirect()->intended('/home');
            }
/*elseif ($socialUser->getEmail().toString()==''){
                $userObj = new User();
                $userObj->facebook_id=$socialUser->getId();
                $userObj->name=$socialUser->getName();
                $userObj->email=$socialUser->getId().'@ownNote.com';
                $userObj->gender='';
                $userObj->phone='';
                $userObj->dateOfBirth='';
                $userObj->password=bcrypt(123456);
                $userObj->save();
                Auth::login($userObj);
                return redirect()->intended('/home');
            }*/
else{
                Auth::login($user2);
return redirect()->intended('/home');
            }

        }
catch (\Exception $e){
// return dd($socialUser);
return redirect('/login');
        }
//return dd($user);

}
}
```

9.
```
<?php

namespace App\Http\Controllers;

use App\NotificationModel;
```

```
use App\Response;
use Auth;
use DB;
use App\imageId;
use Illuminate\Http\Request;

class HomeController extends Controller
{
/**
     * Create a new controller instance.
     *
     * @return void
     */
public function __construct()
    {
$this->middleware('auth');
    }

/**
     * Show the application dashboard.
     *
     * @return \Illuminate\Http\Response
     */
public function index()
    {
// $notificationObj=NotificationModel::all();


       // $responseObj=Response::all();
$responseObj = DB::table('responses')
            ->join('users', 'responses.user_id', '=', 'users.id')
            ->select('responses.*','users.name','users.proPic')
            ->orderBy('id', 'desc')
            ->get();

//return $responseObj;
       /*$userObject=User::where('id',$id)->first();
       return
view('frontEnd.update.loginUpdate',['userInfoById'=>$userObject]);*/
       /*
        *  echo '<pre>';
        print_r();
        exit();
       */
       //$imageid=DB::table('image_ids')
           /*->get();*/
$imageid=imageId::where('id','2')->first();
$imageid=imageId::all();
//return $imageid->imageId;
$statusData = DB::table('users')
            ->join('statas_posts', 'users.id', '=',
'statas_posts.user_id')
            ->select('statas_posts.*', 'users.name','users.proPic',
'users.email')
```

```
/*->where('id',$id)
                    ->first()*/
->orderBy('id', 'desc')
              ->paginate(10);
/* ->get();*/

         /* echo '<pre>';
          print_r($statusData);
          exit();*/
         //return $statusData;
return view('home', ['statusDataAlls' =>$statusData,'imageid'
=>$imageid,'responseId' =>$responseObj]);
    }
}
```

10.

```
<?php

namespace App\Http\Controllers;
use App\imageId;
use DB;
use App\User;
use Illuminate\Http\Request;

class HomeIndexController extends Controller
{
public function homeIndex(){
return view('frontEnd.home.HomeIndex');
    }public function signInOut(){
return view('frontEnd.home.login');
    }
public function search(Request $request ){
$term=$request->term;
$users=User::where('name','LIKE','%'.$term.'%')-
>orWhere('email','LIKE','%'.$term.'%')->get();
if(count($users)==0){
$searchItem[]='No item found';
        }else{
foreach ($users as $key=>$value){
$searchItem[]=$value->name;
            }
        }
return $searchItem;
    }
public function searchResult(Request $request){
$searchKey=$request->searchItem;
$users=User::where('name','LIKE','%'.$searchKey.'%')-
>orWhere('email','LIKE','%'.$searchKey.'%')->get();
//$users[]='No item found';
       //return $users->all();
return view('frontEnd.home.searchList', ['usersall' =>$users]);

    }
public function PostById($id){
```

```
$responseObj = DB::table('responses')
            ->join('users', 'responses.user_id', '=', 'users.id')
            ->select('responses.*','users.name','users.proPic')
            ->where('responses.post_id',$id)
            ->orderBy('id', 'desc')
            ->get();
// $imageid=imageId::where('id','2')->first();
$imageid=imageId::all();
//return $imageid->imageId;
$statusData = DB::table('users')
            ->join('statas_posts', 'users.id', '=',
'statas_posts.user_id')
            ->select('statas_posts.*', 'users.name','users.proPic',
'users.email')
            ->where('statas_posts.id',$id)
            ->first();

return view('frontEnd.home.PostById', ['statusDataAll'
=>$statusData,'imageid' =>$imageid,'responseId' =>$responseObj]);
    }
public function allPostById($id){
$responseObj = DB::table('responses')
            ->join('users', 'responses.user_id', '=', 'users.id')
            ->select('responses.*','users.name','users.proPic')
            ->orderBy('id', 'desc')
            ->get();

$imageid=imageId::all();
//return $imageid->imageId;
$statusData = DB::table('users')
            ->join('statas_posts', 'users.id', '=',
'statas_posts.user_id')
            ->select('statas_posts.*', 'users.name','users.proPic',
'users.email')
            ->where('user_id',$id)
            ->orderBy('id', 'desc')
            ->paginate(10);
return view('frontEnd.home.allPostById', ['statusDataAlls'
=>$statusData,'imageid' =>$imageid,'responseId' =>$responseObj]);


    }
}
```

11. `<?php`

```
namespace App\Http\Controllers;
use Auth;
use App\User;
use Illuminate\Http\Request;
use Socialite;
```

```
class linkedinLoginController extends Controller
{
public function redirectToProvider()
    {
return Socialite::driver('linkedin')->redirect();
    }

/**
     * Obtain the user information from GitHub.
     *
     * @return Response
     */

public function handleProviderCallback()
    {
try{
$socialUser = Socialite::driver('linkedin')->user();
$user=User::where('linkedin_id',$socialUser->getId())->first();
$user2=User::where('email',$socialUser->getEmail())->first();
/* $a=(string)$socialUser->getEmail();*/
if($user2==null &&$user==null){
$userObj = new User();
$userObj->linkedin_id=$socialUser->getId();
$userObj->name=$socialUser->getName();
$userObj->email=$socialUser->getEmail();
$userObj->gender='';
$userObj->phone='';
$userObj->statusVerify='1';;
$userObj->dateOfBirth='';
$userObj->password=bcrypt(123456);
$userObj->save();
                Auth::login($userObj);
return redirect()->intended('/home');
            }
/*elseif ($socialUser->getEmail().toString()==''){
                $userObj = new User();
                $userObj->facebook_id=$socialUser->getId();
                $userObj->name=$socialUser->getName();
                $userObj->email=$socialUser->getId().'@ownNote.com';
                $userObj->gender='';
                $userObj->phone='';
                $userObj->dateOfBirth='';
                $userObj->password=bcrypt(123456);
                $userObj->save();
                Auth::login($userObj);
                return redirect()->intended('/home');
            }*/
else{
                Auth::login($user2);
return redirect()->intended('/home');
            }

        }
catch (\Exception $e){
```

```
// return dd($socialUser);
return redirect('/login');
        }
//return dd($user);

}
}
```

12.

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\EduUp;
use App\BasicUp;
use App\User;
use DB;
class ProfileController extends Controller
{
public function index($id){
$userObject=User::where('id',$id)->first();
$BasicObject=BasicUp::where('id',$id)->first();
$EduObject=EduUp::where('id',$id)->first();
/* $eduBasicObject = EduUp::all();
        $basicBasicObject = BasicUp::all();
        $userBasicObject = User::all();*/
if( $BasicObject==null &&$EduObject==null){
$basicObj=new BasicUp();
$basicObj->id = $id;
$basicObj->livingIn = '';
$basicObj->language = '';
$basicObj->birthdayPlace = '';
$basicObj->status = '';
$basicObj->religion = '';
$basicObj->bloodGroup = '';
$basicObj->socialNetwork ='';
$basicObj->address = '';
$basicObj->save();

$EduObj=new EduUp();
$EduObj->id = $id;
$EduObj->schools = '';
$EduObj->college = '';
$EduObj->highSchools = '';
$EduObj->university = '';
$EduObj->professionalSkills = '';
$EduObj->personalSkills = '';
$EduObj->technicalSkills ='';
$EduObj->achievement = '';
$EduObj->others = '';
$EduObj->save();
return
```

```
view('frontEnd.update.profile',['eduinfo'=>$EduObj,'basicinfo'=>$bas
icObj,'userInfo'=>$userObject]);
        }
elseif( $BasicObject==null &&$EduObject!=null){
$basicObj=new BasicUp();
$basicObj->id = $id;
$basicObj->livingIn = '';
$basicObj->language = '';
$basicObj->birthdayPlace = '';
$basicObj->status = '';
$basicObj->religion = '';
$basicObj->bloodGroup = '';
$basicObj->socialNetwork ='';
$basicObj->address = '';
$basicObj->save();
return
view('frontEnd.update.profile',['eduinfo'=>$EduObject,'basicinfo'=>$
basicObj,'userInfo'=>$userObject]);
        }
elseif( $BasicObject!=null &&$EduObject==null){
$EduObj=new EduUp();
$EduObj->id = $id;
$EduObj->schools = '';
$EduObj->college = '';
$EduObj->highSchools = '';
$EduObj->university = '';
$EduObj->professionalSkills = '';
$EduObj->personalSkills = '';
$EduObj->technicalSkills ='';
$EduObj->achievement = '';
$EduObj->others = '';
$EduObj->save();
return
view('frontEnd.update.profile',['eduinfo'=>$EduObj,'basicinfo'=>$Bas
icObject,'userInfo'=>$userObject]);
        }
else
            return
view('frontEnd.update.profile',['eduinfo'=>$EduObject,'basicinfo'=>$
BasicObject,'userInfo'=>$userObject]);


}
public function proUp(Request $request){
if($request->hasFile('Prophoto')){
$userObj=User::find($request->userId);

$name = $request->Prophoto->getClientOriginalName();
$uploadPath = 'User/image/';
$request->Prophoto->move($uploadPath, $name);
$imageUrl = $uploadPath . $name;
$userObj->proPic= $imageUrl;
$userObj->save();
return redirect()->to('profile/'.$request->userId);
```

```
        }else{
return redirect()->to('profile/'.$request->userId);
        }

    }
public function gallery($id){
$imageData = DB::table('users')
            ->join('statas_posts', 'users.id', '=',
'statas_posts.user_id')
            ->join('image_ids', 'image_ids.imageId', '=',
'statas_posts.imageId')
            -
>select('image_ids.upload_photo','statas_posts.post_time','statas_po
sts.status')
            ->where('users.id',$id)
            ->distinct()
            ->get();
//return  $imageData;
return view('frontEnd.home.gallery',['imageDatas'=>$imageData]);

    }
}
```

13.

```
<?php

namespace App\Http\Controllers;

use App\CategoryList;
use App\imageId;
use App\NotificationModel;
use App\Response;
use App\User;
use Auth;
use DB;
use App\StatasPost;
use Illuminate\Http\Request;

class Status_postController extends Controller
{


public function statusUp(Request $request)
    {
$i=substr(md5(time()), 0, 11);

// return $request->all();

$this->validate($request, [
'upload_photo.*' =>'image|max:1024',
//'status'=>'required',
]);
```

```
if($request->hasFile('upload_photo'))
        {

foreach ($request->upload_photo as $file){
$filename=$file->getClientOriginalName();
print_r($filename.'<br>');
$name = $file->getClientOriginalName();
$uploadPath = 'User/image/';
$file->move($uploadPath, $name);
$imageUrl = $uploadPath . $name;
//$this->saveFunction($request, $imageUrl);

$imageIdObj=new imageId();
//$statusObj->imageId = $i;
$imageIdObj->upload_photo=$imageUrl;
$imageIdObj->imageId=$i;
$imageIdObj->save();
            }
$statusObj = new StatasPost();
$statusObj->user_id = Auth::user()->id;
$statusObj->status = $request->status;
$statusObj->whoSee = $request->whoSee;
$statusObj->option = $request->option;
$statusObj->upload_photo = '';
$statusObj->anonymous = $request->anonymous;
$statusObj->post_time = date("Y-m-d (h:i:sa)");
$statusObj->imageId = $i;
$statusObj->save();

$categoryListAddByuser = new CategoryList();
$categoryListAddByuser->user_id = Auth::user()->id;
$categoryListAddByuser->Category_name = $request->option;
$categoryListAddByuser->save();


$notificationObj=new NotificationModel();
//$uObj=User::find( Auth::user()->id )->first();
$uObj= DB::table('users')
                ->where('id', '=',Auth::user()->id)
                ->select('users.*')
                ->first();

$sPObj= DB::table('statas_posts')
                ->where('user_id', '=',Auth::user()->id)
                ->where('status', '=',$request->status)
                ->where('post_time', '=',$statusObj->post_time)
                ->select('statas_posts.*')
                ->first();


$notificationObj->user_id=Auth::user()->id;
$notificationObj->user_name=$uObj->name;
$notificationObj->message=' update a status ';
```

```
$notificationObj->link_id=$sPObj->id;
$notificationObj->anonymous=$request->anonymous;
$notificationObj->time=$statusObj->post_time;
$notificationObj->save();

return redirect('/home');




        }
else{

$imageUrl = '';
/*$this->saveFunction($request, $imageUrl);*/
$statusObj = new StatasPost();
$imageIdObj=new imageId();

$statusObj->user_id = Auth::user()->id;
$statusObj->status = $request->status;
$statusObj->whoSee = $request->whoSee;
$statusObj->option = $request->option;
$statusObj->upload_photo = $imageUrl;
$statusObj->anonymous = $request->anonymous;
$statusObj->post_time = date("Y-m-d (h:i:sa)");
//$statusObj->imageId = substr(md5(time()), 0, 11);
$statusObj->imageId =$i;
$imageIdObj->upload_photo=$statusObj->upload_photo;
$imageIdObj->imageId=$statusObj->imageId;
$statusObj->save();
$imageIdObj->save();

$categoryListAddByuser = new CategoryList();
$categoryListAddByuser->user_id = Auth::user()->id;
$categoryListAddByuser->Category_name = $request->option;
$categoryListAddByuser->save();
$notificationObj=new NotificationModel();
$uObj= DB::table('users')
                ->where('id', '=',Auth::user()->id)
                ->select('users.*')
                ->first();

$sPObj= DB::table('statas_posts')
                ->where('user_id', '=',Auth::user()->id)
                ->where('status', '=',$request->status)
                ->where('post_time', '=',$statusObj->post_time)
                ->select('statas_posts.*')
                ->first();


$notificationObj->user_id=Auth::user()->id;
$notificationObj->user_name=$uObj->name;
$notificationObj->message=' update a status ';
$notificationObj->link_id=$sPObj->id;
```

```
$notificationObj->anonymous=$request->anonymous;
$notificationObj->time=$statusObj->post_time;
$notificationObj->save();

return redirect('/home');

          }

/*$image = $request->file('upload_photo');
        if ($image == null) {
            $imageUrl = '';
        } else {
            $name = $image->getClientOriginalName();
            $uploadPath = 'User/image/';
            $image->move($uploadPath, $name);
            $imageUrl = $uploadPath . $name;
        }*/




}

/*protected function saveFunction($request, $imageUrl)
    {

        $statusObj = new StatasPost();
        $imageIdObj=new imageId();
        $statusObj->user_id = Auth::user()->id;
        $statusObj->status = $request->status;
        $statusObj->whoSee = $request->whoSee;
        $statusObj->option = $request->option;
        $statusObj->upload_photo = $imageUrl;
        $statusObj->anonymous = $request->anonymous;
        $statusObj->post_time = date("Y-m-d (h:i:sa)");
        //$statusObj->imageId = substr(md5(time()), 0, 11);
        $statusObj->imageId = $i;
        $imageIdObj->upload_photo=$statusObj->upload_photo;
        $imageIdObj->imageId=$statusObj->imageId;
        $statusObj->save();
        $imageIdObj->save();

    }*/

public function statusDelete($id)
    {

$statusObj = StatasPost::find($id);
$statusById1 = StatasPost::where('id',$id)->first();
$ImageId = $statusById1->imageId;
$imageidObj = imageId::where('imageId',$ImageId)->get();
$theDefaults = array();
foreach($imageidObj as $v) {
```

```
$theDefaults[$v->id] = $v->upload_photo;
        }
//return $theDefaults;
       /* print_r($theDefaults);
        exit();*/
        //$oldImageUrl1 = $statusById1->upload_photo;
if($imageidObj !=null){
try{
foreach ($theDefaults as $imageid ){
unlink($imageid);
                }
            }catch (\Exception $e){
$statusObj->delete();
                imageId::where('imageId', $ImageId)->delete();
return redirect('/home');
            }
        }

/*if($oldImageUrl1 !=null){
            try{
                unlink($oldImageUrl1);
            }catch (\Exception $e){
                $statusObj->delete();
                return redirect('/home');
            }

        }*/
imageId::where('imageId', $ImageId)->delete();
$statusObj->delete();
return redirect('/home');
    }

public function statusEdit($id)
    {

$statusObj = StatasPost::where('id', $id)->first();
$userId = $statusObj->user_id;
$imageId = $statusObj->imageId;
$userObj = User::where('id', $userId)->first();
$imageObject = imageId::where('imageId', $imageId)->get();
/*echo '<pre>';
        print_r($statusObj);
        exit();*/
return view('frontEnd.home.statusEdit', ['StatusById' =>$statusObj,
'StatusByUser' =>$userObj, 'StatusByImage' =>$imageObject,]);
    }

public function statusUpdate(Request $request)
    {
if($request->hasFile('upload_photo'))
        {
$statusById = StatasPost::where('id', $request->Statusid)->first();
$i=substr(md5(time()), 0, 11);
$this->validate($request, [
```

```
'upload_photo.*' =>'image|max:1024',
//'status'=>'required',
]);

$ImageId=$statusById->imageId;
$imageidObj = imageId::where('imageId',$ImageId)->get();
$theDefaults = array();
foreach($imageidObj as $v) {
$theDefaults[$v->id] = $v->upload_photo;
            }
if($imageidObj !=null){
try{
foreach ($theDefaults as $imageid ){
unlink($imageid);
                    }
                    imageId::where('imageId', $ImageId)->delete();
                }catch (\Exception $e){
                    imageId::where('imageId', $ImageId)->delete();
                }
            }


foreach ($request->upload_photo as $file){
$name = $file->getClientOriginalName();
$uploadPath = 'User/image/';
$file->move($uploadPath, $name);
$imageUrl = $uploadPath . $name;
$imageIdObj=new imageId();
$imageIdObj->upload_photo=$imageUrl;
$imageIdObj->imageId=$i;
$imageIdObj->save();
            }
$statusObj=StatasPost::find($request->Statusid);
$statusObj->user_id = $request->userId;
$statusObj->status = $request->status;
$statusObj->whoSee = $request->whoSee;
$statusObj->option = $request->option1;
$statusObj->imageId = $i;
$statusObj->anonymous = $request->anonymous;
$statusObj->post_time = date("Y-m-d (h:i:sa)");
$statusObj->save();



return redirect('/home');
        }
else{
$statusObj=StatasPost::find($request->Statusid);
$statusObj->user_id = $request->userId;
$statusObj->status = $request->status;
$statusObj->whoSee = $request->whoSee;
$statusObj->option = $request->option1;
$statusObj->anonymous = $request->anonymous;
```

```
$statusObj->post_time = date("Y-m-d (h:i:sa)");
$statusObj->save();



return redirect('/home');
        }

// $imageUrl = $this->imageExistStatus($request);


}

private function imageExistStatus($request)
    {/*


        if ($statusImage) {
            $oldImageUrl = $statusById->upload_photo;
            unlink($oldImageUrl);
            $name = $statusImage->getClientOriginalName();
            $uploadPath = 'User/image/';
            $statusImage->move($uploadPath, $name);

            $imageUrl = $uploadPath . $name;
        } else {
            $imageUrl = $statusById->upload_photo;
        }
        return $imageUrl;*/
}

public function responseCreate(Request $request){
$responseObj=new Response();
$responseObj->user_id=$request->user_id;
$responseObj->post_id=$request->post_id;
$responseObj->post_time=date("Y-m-d (h:i:sa)");
$responseObj->Response=$request->text;
$responseObj->anonymous=$request->responseAnonymous;
$responseObj->save();


$notificationObj=new NotificationModel();
//$statusPostObj=StatasPost::find($request->post_id)->first();
$statusPostObj= DB::table('statas_posts')
            ->where('id', '=',$request->post_id)
            ->select('statas_posts.*')
            ->first();


$uObj= DB::table('users')
            ->where('id', '=',$request->user_id)
            ->select('users.*')
            ->first();
```

```
$notificationObj->user_id=$request->user_id;
$notificationObj->user_name=$uObj->name;
$notificationObj->message=' update a response ';
$notificationObj->link_id=$request->post_id;
$notificationObj->anonymous=$request->responseAnonymous;
$notificationObj->time=$responseObj->post_time;

$notificationObj->form=$statusPostObj->user_id;
$notificationObj->save();


//return $request->all();
return 'Done';
    }
public function responseUpdate(Request $request){

$responseObj=Response::find($request->id);
$responseObj->Response=$request->value;
$responseObj->post_time=date("Y-m-d (h:i:sa)");
$responseObj->anonymous=$request->responseAnonymous;
$responseObj->save();
// return 'Done';
}

public function responseDelete(Request $request){
        Response::where('id',$request->id)->delete();
    }
}
```

14.

```
<?php

namespace App\Http\Controllers;
use Auth;
use App\User;
use Illuminate\Http\Request;
use Socialite;
class TwitterLoginController extends Controller
{
public function redirectToProvider()
    {
return Socialite::driver('twitter')->redirect();
    }

/**
     * Obtain the user information from GitHub.
     *
     * @return Response
     */

public function handleProviderCallback()
```

```
    {
try{
$socialUser = Socialite::driver('twitter')->user();
$user=User::where('twitter_id',$socialUser->getId())->first();
$user2=User::where('email',$socialUser->getEmail())->first();
/* $a=(string)$socialUser->getEmail();*/
if($user2==null &&$user==null){
$userObj = new User();
$userObj->twitter_id=$socialUser->getId();
$userObj->name=$socialUser->getName();
$userObj->email=$socialUser->getEmail();
$userObj->gender='';
$userObj->phone='';
$userObj->statusVerify='1';;
$userObj->dateOfBirth='';
$userObj->password=bcrypt(123456);
$userObj->save();
                Auth::login($userObj);
return redirect()->intended('/home');
            }
/*elseif ($socialUser->getEmail().toString()==''){
                $userObj = new User();
                $userObj->facebook_id=$socialUser->getId();
                $userObj->name=$socialUser->getName();
                $userObj->email=$socialUser->getId().'@ownNote.com';
                $userObj->gender='';
                $userObj->phone='';
                $userObj->dateOfBirth='';
                $userObj->password=bcrypt(123456);
                $userObj->save();
                Auth::login($userObj);
                return redirect()->intended('/home');
            }*/
else{
                Auth::login($user2);
return redirect()->intended('/home');
            }

        }
catch (\Exception $e){
// return dd($socialUser);
return redirect('/login');
        }
//return dd($user);

}
}
```

15.

```
<?php

namespace App\Http\Controllers;
use App\CategoryList;
use App\NotificationModel;
```

```
use App\Response;
use App\StatasPost;
use Illuminate\Support\Facades\Auth;
use Illuminate\Http\Request;
use App\BasicUp;
use App\User;
use App\EduUp;
use App\Notifications;
use DB;
class UpdateController extends Controller
{
public function upLogin($id){
// return $email;
$userObject=User::where('id',$id)->first();
//$userObject=User::where('id',$id)->get();
//        return $userObject;
return
view('frontEnd.update.loginUpdate',['userInfoById'=>$userObject]);
    }
public function upBasic($id){
$BasicObject=BasicUp::where('id',$id)->first();
if( $BasicObject==null){
$basicObj=new BasicUp();
$basicObj->id = $id;
$basicObj->livingIn = '';
$basicObj->language = '';
$basicObj->birthdayPlace = '';
$basicObj->status = '';
$basicObj->religion = '';
$basicObj->bloodGroup = '';
$basicObj->socialNetwork ='';
$basicObj->address = '';
$basicObj->save();
return
view('frontEnd.update.basicUpdate',['BasicObject'=>$basicObj]);
        }
return
view('frontEnd.update.basicUpdate',['BasicObject'=>$BasicObject]);
    }
public function upEducation($id){
$EduObject=EduUp::where('id',$id)->first();
if( $EduObject==null){
$EduObj=new EduUp();
$EduObj->id = $id;
$EduObj->schools = '';
$EduObj->college = '';
$EduObj->highSchools = '';
$EduObj->university = '';
$EduObj->professionalSkills = '';
$EduObj->personalSkills = '';
$EduObj->technicalSkills ='';
$EduObj->achievement = '';
$EduObj->others = '';
$EduObj->save();
```

```
return view('frontEnd.update.eduUpdate',['userInfoById'=>$EduObj]);
        }
return
view('frontEnd.update.eduUpdate',['userInfoById'=>$EduObject]);
    }
public function upBasicSave(Request $request){
// return $request->all();
    //return view('frontEnd.update.eduUpdate');
      //  $basicUpObject=new BasicUp();
$basicUpObject=BasicUp::find($request->id);
$basicUpObject->livingIn = $request->livingIn;
$basicUpObject->language = $request->language;
$basicUpObject->birthdayPlace = $request->birthdayPlace;
$basicUpObject->status = $request->status;
$basicUpObject->religion = $request->religion;
$basicUpObject->bloodGroup = $request->bloodGroup;
$basicUpObject->socialNetwork = $request->socialNetwork;
$basicUpObject->address = $request->address;
$basicUpObject->save();

$notificationObj=new NotificationModel();
//$uObj=User::find(Auth::user()->id)->first();
$uObj= DB::table('users')
            ->where('id', '=',Auth::user()->id)
            ->select('users.*')
            ->first();

$notificationObj->user_id=Auth::user()->id;
$notificationObj->user_name=$uObj->name;
$notificationObj->message=' update profile ';
$notificationObj->link_id=Auth::user()->id;
$notificationObj->save();

//return'Basic info Update successfully';
/*    return redirect('/update/basic')->with('message','Basic info
Update successfully');*/
return redirect('update/basic/'.$request->id)->with('message','Basic
info Update successfully');
    }
public function upEduSave(Request $request){
// return $request->all();
    //return view('frontEnd.update.eduUpdate');
    //return'Basic info Update successfully';
        /*DB::table('edu_ups')->insert([
            'schools'=>$request->schools,
            'college'=>$request->college,
            'highSchools'=>$request->highSchools,
            'university'=>$request->university,
            'professionalSkills'=>$request->professionalSkills,
            'personalSkills'=>$request->personalSkills,
            'technicalSkills'=>$request->technicalSkills,
            'achievement'=>$request->achievement,
            'others'=>$request->others,
        ]);*/
```

```
$EduBasicUp=EduUp::find($request->id);
$EduBasicUp->schools = $request->schools;
$EduBasicUp->college = $request->college;
$EduBasicUp->highSchools = $request->highSchools;
$EduBasicUp->university = $request->university;
$EduBasicUp->professionalSkills = $request->professionalSkills;
$EduBasicUp->personalSkills = $request->personalSkills;
$EduBasicUp->technicalSkills = $request->technicalSkills;
$EduBasicUp->achievement = $request->achievement;
$EduBasicUp->others = $request->others;
$EduBasicUp->save();
$notificationObj=new NotificationModel();
// $uObj=User::find(Auth::user()->id)->first();
$uObj= DB::table('users')
            ->where('id', '=',Auth::user()->id)
            ->select('users.*')
            ->first();

$notificationObj->user_id=Auth::user()->id;
$notificationObj->user_name=$uObj->name;
$notificationObj->message=' update profile ';
$notificationObj->link_id=Auth::user()->id;
$notificationObj->save();
/*    return redirect('/update/education')-
>with('message','Education info Update successfully');*/
return redirect('/update/education/'.$request->id)-
>with('message','Education info Update successfully');
    }
public function upLoginSave(Request $request){
// return $request->all();
    //return view('frontEnd.update.eduUpdate');
    //return'Basic info Update successfully';
       /* dd($request->all());*/
$userObject2=User::find($request->id);
$userObject2->name = $request->name;
$userObject2->email = $request->email;
$userObject2->gender = $request->gender;
$userObject2->phone = $request->phone;
$userObject2->dateOfBirth = $request->date;
$userObject2->password = bcrypt($request->password);
$userObject2->save();

$notificationObj=new NotificationModel();
//$uObj=User::find(Auth::user()->id)->first();
$uObj= DB::table('users')
            ->where('id', '=',Auth::user()->id)
            ->select('users.*')
            ->first();

$notificationObj->user_id=Auth::user()->id;
$notificationObj->user_name=$uObj->name;
$notificationObj->message=' update profile ';
$notificationObj->link_id=Auth::user()->id;
$notificationObj->save();
```

```
return redirect('/update/login/'.$request->id)->with('message','Info
Update successfully');;
    }

public function deleteUser(Request $request){

//$imageid=StatasPost::where('user_id',$request->id)->first();
        //imageId::where('imageId',$imageid->imageId)->delete();
BasicUp::where('id',$request->id)->delete();
        CategoryList::where('user_id',$request->id)->delete();
        EduUp::where('id',$request->id)->delete();
        Response::where('user_id',$request->id)->delete();
        StatasPost::where('user_id',$request->id)->delete();
        User::where('id',$request->id)->delete();
    }
}
```

16.

```
<?php

namespace App\Http\Controllers;

use App\OthersComment;
use Illuminate\Http\Request;

class WelcomeController extends Controller
{
public function welcome(){
return view('frontEnd.home.welcome');
    }
public function OutCustomar(Request $request){

$othersObj=new OthersComment();
$othersObj->name=$request->name;
$othersObj->email=$request->email;
$othersObj->comment=$request->comment;
$othersObj->save();
    }
}
```

17.

```
<?php

namespace App\Providers;

use Illuminate\Support\Facades\Schema;
use View;
use Auth;
use DB;
use App\StatasPost;
use App\CategoryList;
use Illuminate\Support\ServiceProvider;

class AppServiceProvider extends ServiceProvider
{
/**
     * Bootstrap any application services.
     *
     * @return void
     */
public function boot()
    {
//Schema::defaultStringLength(191);
View::share('shadhin', 'shadhin');


        View::composer('*', function ($view) {
/*$categoryObj=CategoryList::where('user_id',Auth::user()->id)->get();*/

try {
$notificationObj = DB::table('notification_models')
                    ->select('notification_models.*')
                    ->orderBy('id', 'desc')
                    ->get();

$id = Auth::user()->id;
$categoryObj = DB::table('category_lists')

/*->select('category_lists.*')*/
                    /*->where('id',$id)
                        ->first()*/

                    /*->orderBy('id', 'desc')*/
->distinct()
                    ->where('user_id', $id)
                    ->get(['Category_name']);
$view->with('CategoryListById', $categoryObj);
$view->with('notificationObj', $notificationObj);
            } catch (\Exception $e) {

return redirect('/home');
            }

        });
```

```
    }

/**
     * Register any application services.
     *
     * @return void
     */
public function register()
    {
//
}
}
```

18. App.js

```
/**
 * First we will load all of this project's JavaScript dependencies
which
 * includes Vue and other libraries. It is a great starting point
when
 * building robust, powerful web applications using Vue and Laravel.
 */

require('./bootstrap');

window.Vue = require('vue');
import Vue from 'vue'
import VueChatScroll from 'vue-chat-scroll'
Vue.use(VueChatScroll)

import Toaster from 'v-toaster'
import 'v-toaster/dist/v-toaster.css'
Vue.use(Toaster, {timeout: 5000})
/**
 * Next, we will create a fresh Vue application instance and attach
it to
 * the page. Then, you may begin adding components to this
application
 * or customize the JavaScript scaffolding to fit your unique needs.
 */

Vue.component('message', require('./components/message.vue'));

const app = new Vue({
el: '#app',
data:{
message:'',
chat:{
message:[],
user:[],
color:[],
time:[],
        },
```

```
typing:'',
nuberOfUsers: 0,

    },
watch:{
message(){
Echo.private('chat')
                .whisper('typing', {
name: this.message
});
        }
    },
methods:{
send(){
if(this.message.length!=0){
this.chat.message.push(this.message);
this.chat.color.push('success');
this.chat.user.push('you');
this.chat.time.push(this.getTime());

axios.post('chat/send', {
message:this.message,
chat:this.chat,

                })
                    .then(response => {
console.log(response);
this.message=''
})
                    .catch(error => {
console.log(error);
                    });


            }

        },
getTime(){
let time =new Date();
return time.getHours()+':'+time.getMinutes();
        },
getOldMessages(){
axios.post('/getOldMessage')
                .then(response =>{
console.log(response);
if(response.data !=''){
this.chat=response.data;
                    }

                })
                .catch(error=>{
console.log(error);
                });
```

```
        },
    },

mounted(){
this.getOldMessages();
Echo.private('chat')
            .listen('ChatEvent', (e) => {
this.chat.message.push(e.message);
this.chat.user.push(e.user);
this.chat.color.push('warning');
this.chat.time.push(this.getTime());

axios.post('/saveToSession',{
chat : this.chat
})
                    .then(response =>{

                    })
                    .catch(error=>{
console.log(error);
                    });

// console.log(e);

})
            .listenForWhisper('typing', (e) => {
if(e.name !=''){
this.typing ='typing...'
}
else {
this.typing =''
}

            })
Echo.join(`chat`)
            .here((users) => {
this.nuberOfUsers=users.length;
console.log(users);
            })
            .joining((user) => {
this.nuberOfUsers += 1;
this.$toaster.success(user.name+' is joined ')
console.log(user.name);
            })
            .leaving((user) => {

this.nuberOfUsers -= 1;
this.$toaster.warning(user.name+' is leaved ')

console.log(user.name);
            });
    }

});
```

19. Message.vue

```
<template>
<div><li style="list-style-type: none;  border-radius: 5px;margin: 5px
;padding: 12px;text-align: left;width: 320px;height: 40px"
:class="className"><slot></slot></li>
<small style=" border-radius: 2px;padding: 5px;" :class="badgeClass"
class=" float-right">{{ user }}</small>
<small style=" border-radius: 2px;padding: 5px;" :class="badgeClass"
class=" float-right">{{ time }}</small>
</div>

</template>

<script>
export default {
props:[
'color',
'user',
'time'
],
computed:{
className(){
return 'list-group-item-'+this.color;
            },
badgeClass(){
return 'label label-'+this.color;
            }
        },
mounted() {
console.log('Component mounted.')
        }
}
</script>
```