# 1 INTRODUCTION

## 1.1 OVERVIEW

The social networking site concept not a new for all. We all aware and familiar with social networking concept in real word. Currently the Facebook, YouTube, twitter and many other popular social networking sites take place in each home. In this Python project post we try to develop social site similar to Facebook to communicate all people at one place.

Social networking site is a well-established web based final year project. This is developed to provide a website for people to communicate. people who wants to share their ideas and give their views for creating social awareness can use this application.

Social networking site allows users to create profiles, share the information with friends. they can create the growing network information. Users once registered can establish their views about social responsibilities and made publicly as well.

This site provides a common platform for people of Indian where they can perform many activities related social responsibilities. The site can provide various measures for fellow citizens to rate, control and monitor the social responsibility of individual.

## 1.2 OBJECTIVE

The objective of this Social Networking Site for Social Responsibility project is to explain and elaborate the concept of social networking by providing reliable and efficient communication among people on network. To provide attractive and secure access to the users. To make the application user-friendly.

## 1.3SCOPE

A social networking site is an online platform that allows people to create a public account and interact with other people on the website using internet. In this social networking site project users can create free account and communicate with other users by making friends.

There are different sections like Registration, search people, make friends, make picture gallery, share images using meme generator use all other facilities of our website.

**2.** PROBLEM STATEMENT

2.1PROBLEMINEXISTINGSYSTEM

* No use of Web Services and Remoting.
* Less Security.
* Lack of security of data.
* More man power.
* Time consuming.
* Consumes large volume of pare work.
* Needs manual calculations.
* No direct role for the higher officials.

## 2.2 SOLUTIONOFTHESEPROBLEMS

* The development of this system contains the following activities, which try to automate the entire process keeping in view of the database integration approach.
* User friendliness is provided in the application with various controls.
* The system makes the overall project management much easier and flexible.
* Readily upload the latest updates, allows user to view images.
* There is no risk of data mismanagement at any level while the project development is under process.
* It provides high level of security with different level of authentication.

# 3 PROJECT SIGNIFICANCE

## 3.1 STUDY OF THE SYSTEM

For the flexibility of the user the interface has been developed a graphics concept in mind, associated through a browser interface. The GUI’S at the top level have been categorized as

1. Administrative user interface
2. The operational or generic user interface

The administrative user interface concentrates on the consistent information that is practically, part of the web application activities and which needs proper authentication for the data collection. The interfaces help the administrations with all the transactional states like Data insertion, Data deletion and Date updating along with the extensive data search capabilities.

The operational or generic user interface helps the users upon the system in transactions through the existing data and required services.

**For This Application, we will provide following capabilities:**

Here, we will describe some modules or facilities of the social networking site.

* Admin
* Search Module
* Post Module
* Friend Module
* Gallery
* Account

1. **Admin Module**

The main functions of admin are as mentioned below.

* 1. Login
  2. Management

Admin can view users, delete users, delete Post, delete profile, like a profile user, follow and post picture. Admin can handle this website.

* 1. Logout

**(B)** **User Registration Module**

In this module when user fill-ups registration form user will get a username and in case their forget their password or username they will get on their email. After getting username, user will use his/her member id to login, and user can be part of the application.

1. **Search Module:** A registered user can search other user by name and send them friend request and view his profile and picture gallery.
2. **Post Module:** In this post facility user can share their thoughts to other user via writing a post and users can read and making reply by like post and also share the post in his account using share button. The post is a private shared message which can read by only our friends’ network.
3. **Friend Module:** A friend module is main base module for all social networking site. we all familiar with this word add friend in social networking world. The same functionality we have developed in our friend module, sending friend request to another user and request would be accepted or rejected by request received user.
4. **Gallery Module:** A gallery module allow user to create albums and upload images in album to create picture gallery. The user can also manage the gallery can edit, delete images and albums.
5. **User Account:** In Account section user can update his profile detail and change profile picture, change login password.

## 3.2 Data Flow Diagram

**Level 0**

Create A New ID & Password and Reset password

Login

User Login

Username Password

Username Password

User

Admin

ID Password

**Level 1**

user

Register

User Gallery

Gallery, search

Update

Follow Data, search

Admin

User Gallery

Gallery, search

Databases

Follow Data, search

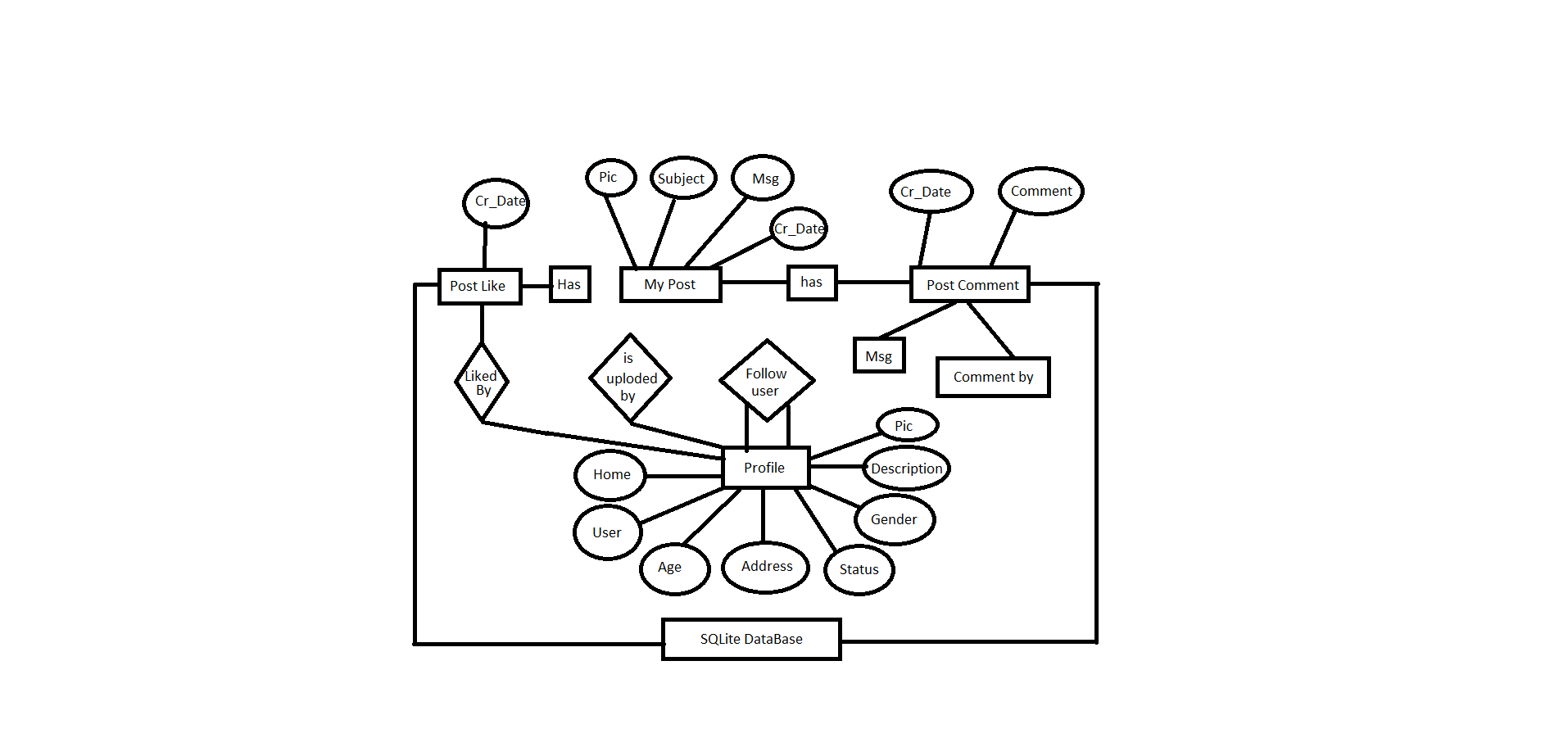
## 3.3 ER Diagram

User

Admin

Esabha

## 3.4 STRUCTURE OF DATABASE

****

# 4 SYSTEM SPECIFICATION

## 4.1 SOFTWRE SPECIFICATION

* **Operating System:** Microsoft windows 7, 8, 10
* **Browser Support:** Chrome, Opera, Microsoft Edge, Firefox above etc.
* **Software Use:** Eclipse, Visual studio Code, Subline, anaconda.
* **Front End:** HTML, CSS, JAVASCRIP, BOOTSTRAP.
* **Back End:** PYTHON, DJANGO, FRAMEWORK, SQLITE.

## 4.2 HARDWARE SPECIFICATION

* Pentium IV2.8 GHz Processor and Above
* RAM 1GBand Above
* HDD 20 GB Hard Disk Space and Above

# 5 PROPOSED SYSTEM

ESabha's purpose is to give people the power to build community and bring the world closer together. People use ESabha to stay connected with friends and family, to discover what's going on in the world, and to share and express what matters to them.

**NEED FOR COMPUTERIZATION**

* Minimizing the manual records kept at different Picture.
* There will be more data integrity.
* Facilitating desired information display, very quickly, by retrieving information from users.
* Facilitating various statistical information which helps in decision-making
* To reduce manual efforts in activities that involved repetitive work.
* Updating and deletion of such a huge amount of data will become easier.

**FUNCTIONAL FEATURES OF THE MODEL**

As far as the project is developed the functionality is simple, the objective of the proposal is to represent information about the area in an effective and better manner.

**For This Application, we will provide following capabilities:**

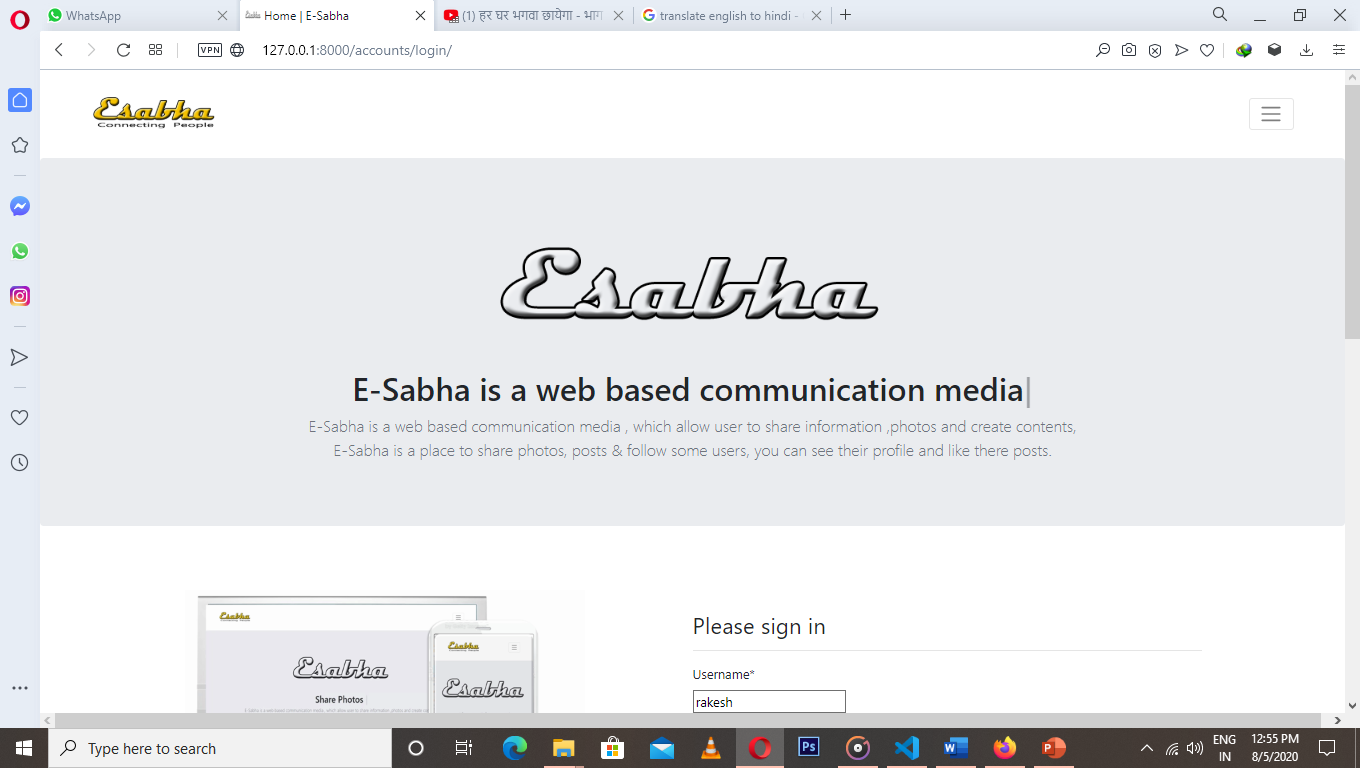
1. Sending messages
2. Voice Text Massage
3. Share photo

# 6 EXISITING SYSTEM

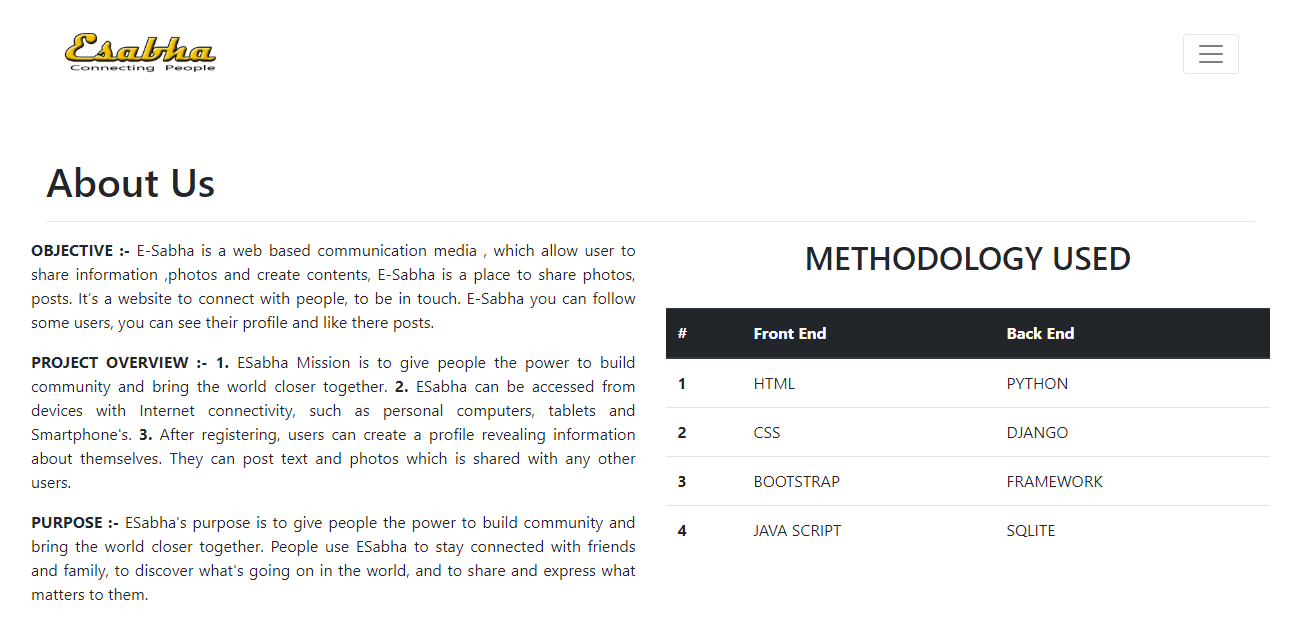
Existing system has no facility to know about the details. Slower processing means that some information that could be provided if computerized systems were used, will not be provided at all, because there is not time Risk of errors. Access to information is often restricted to one user at a time, paper files can easily be misled or buried in in-trays, in which case the information they contain is not available all Alterations. It is difficult to make corrections. If a manual document contains errors or need updating it is often necessary to recreate the whole documents from scratch, rather than just a new version with the relevant details changed. Poorly presented information may fail to communicate key points. Bulk Paper based systems are generally very bulky both to handle and to store, and office space is expensive.

# 7 SCREENSHOTS

**Homepage**

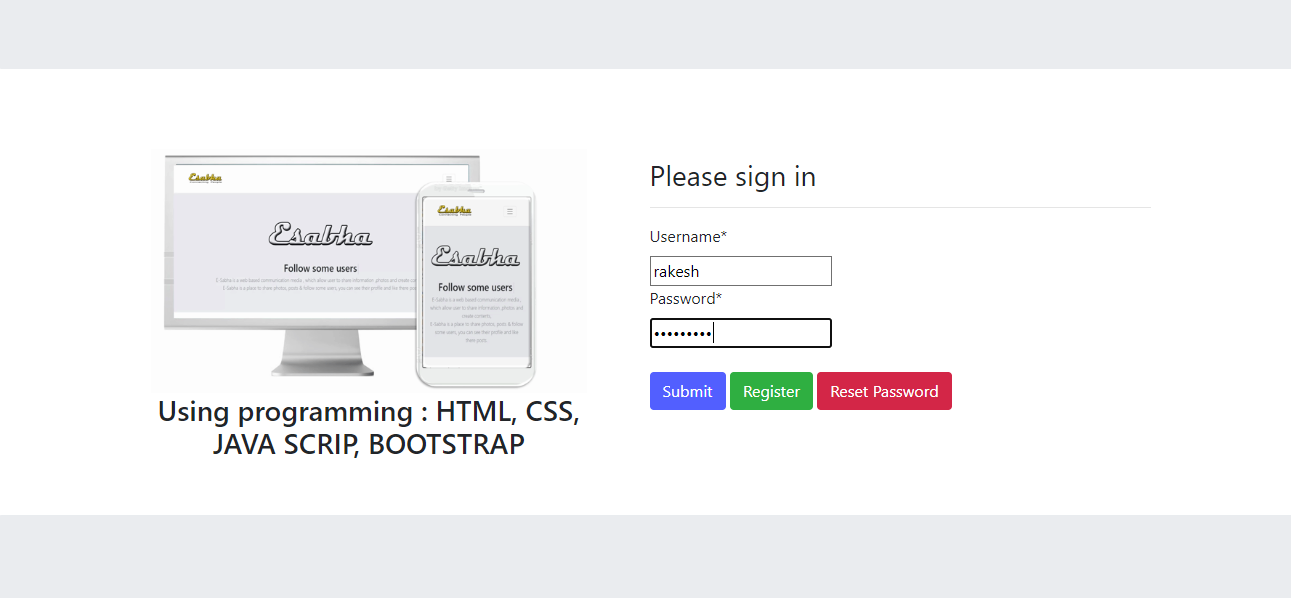


**About Us**

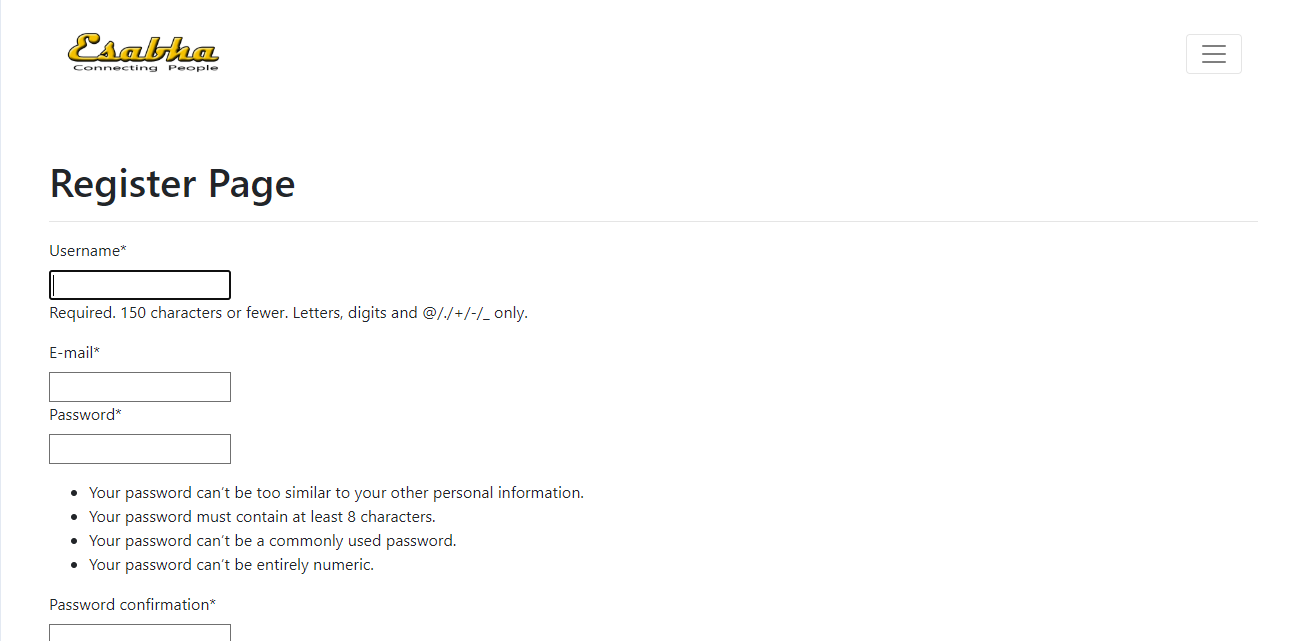


**User Site**

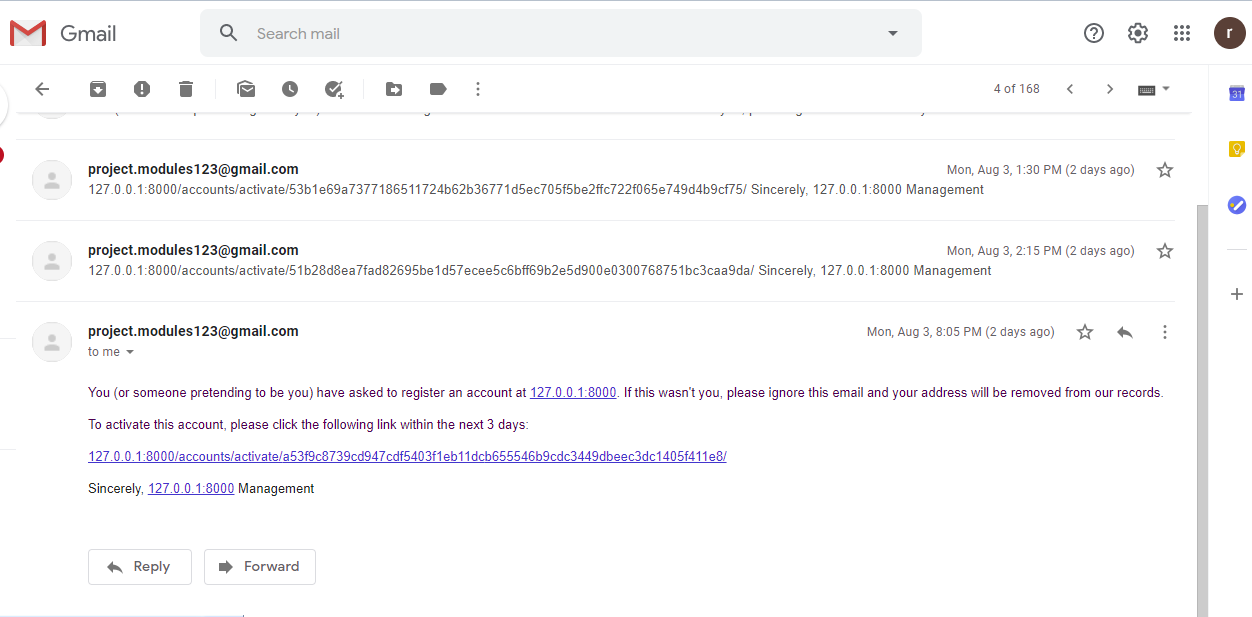
**User Login Page**

****

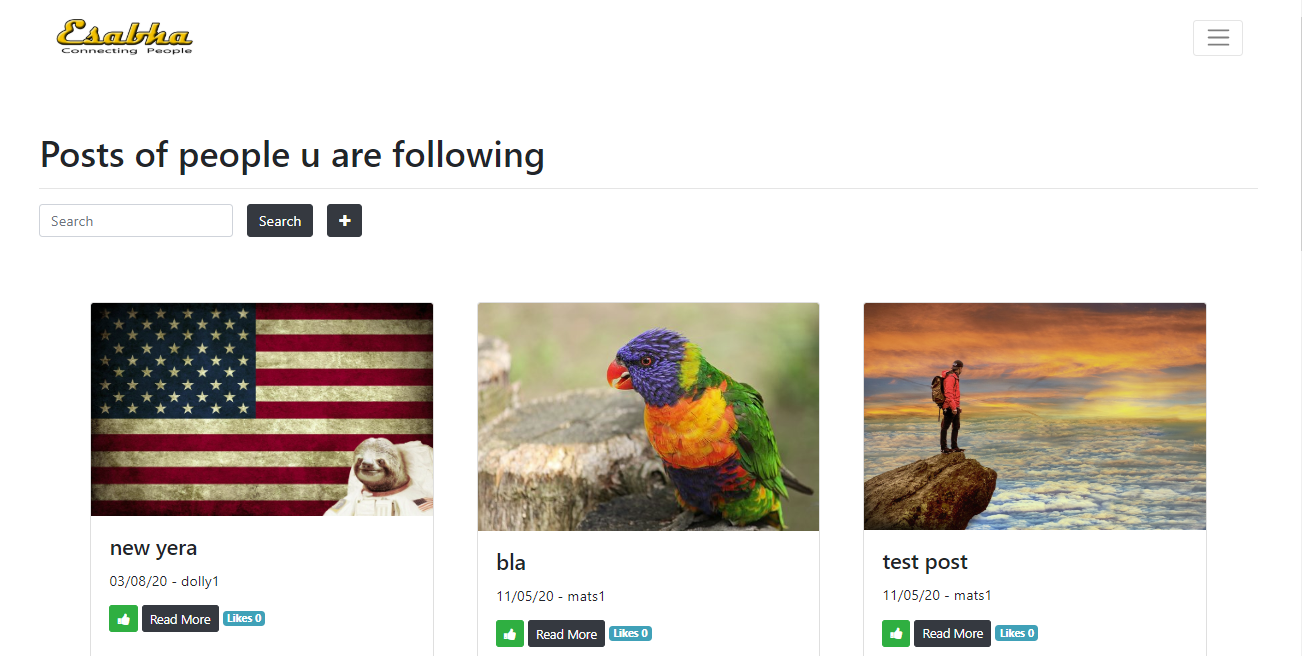
**User Register Page**

****

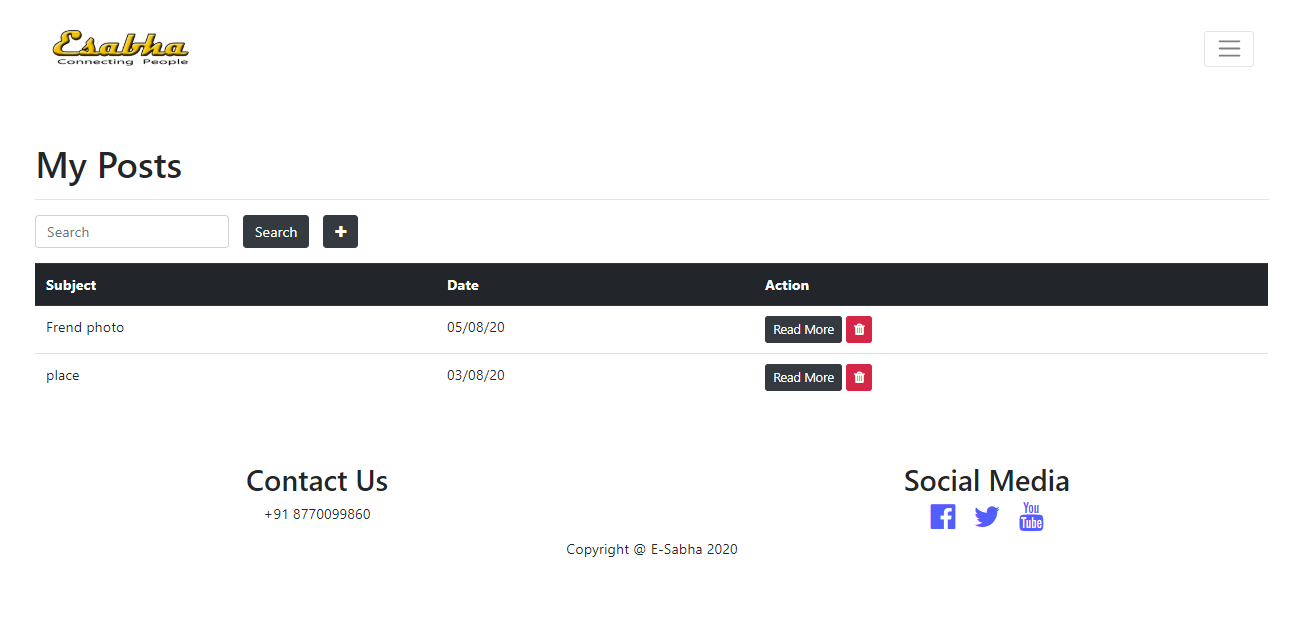
**Account Activation by email**

****

**Posts of people u are following (Login Home Page User)**



# My Posts List (User post photo List and Search And Delete)



User add photo

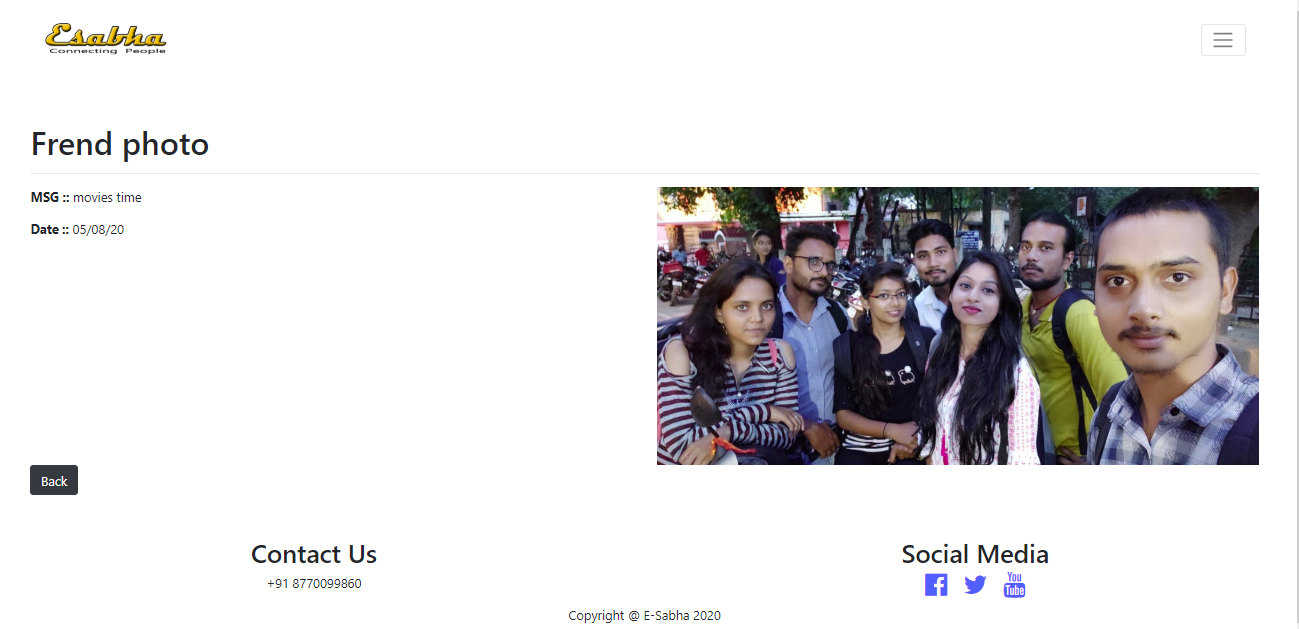
Delete post

Post Views

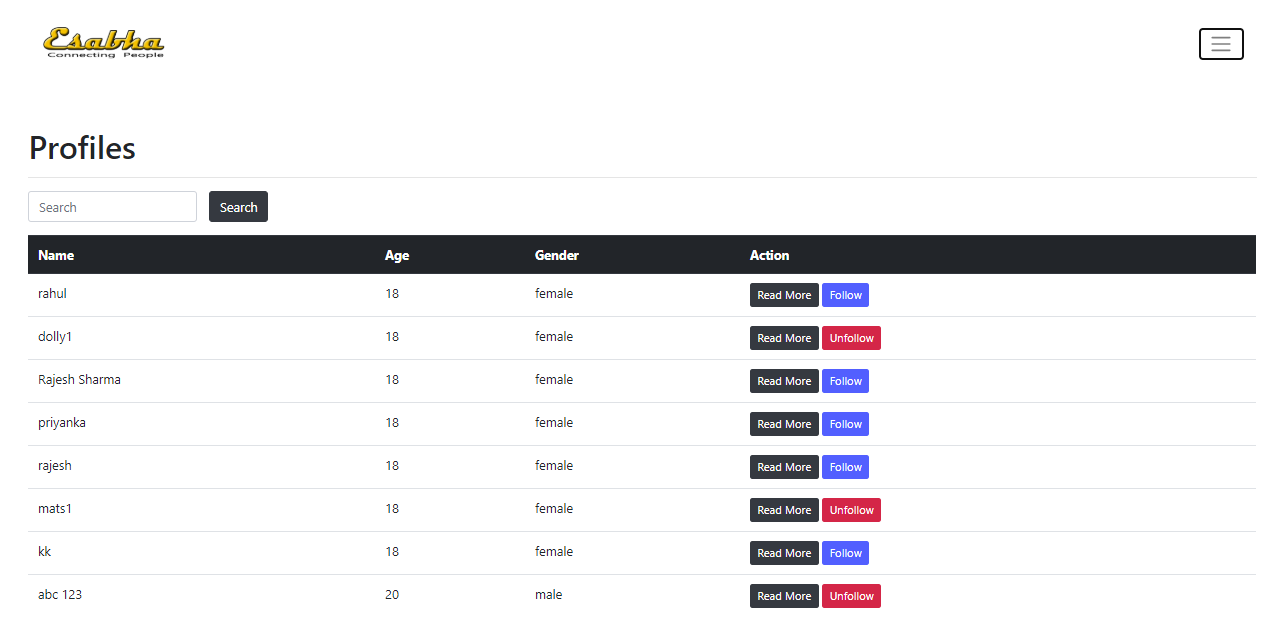
**User add photo**

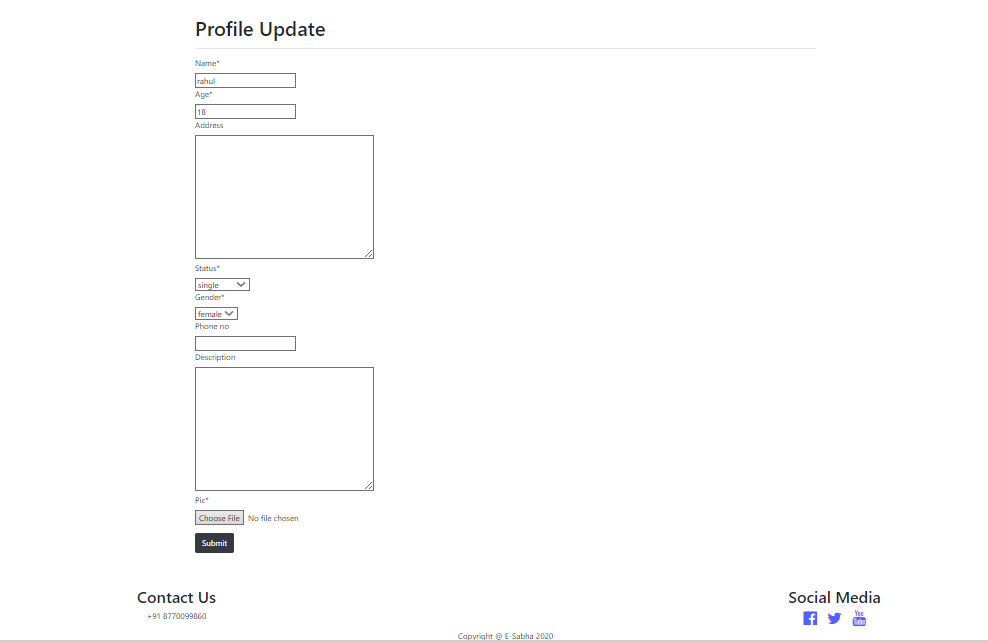


**User Post details**

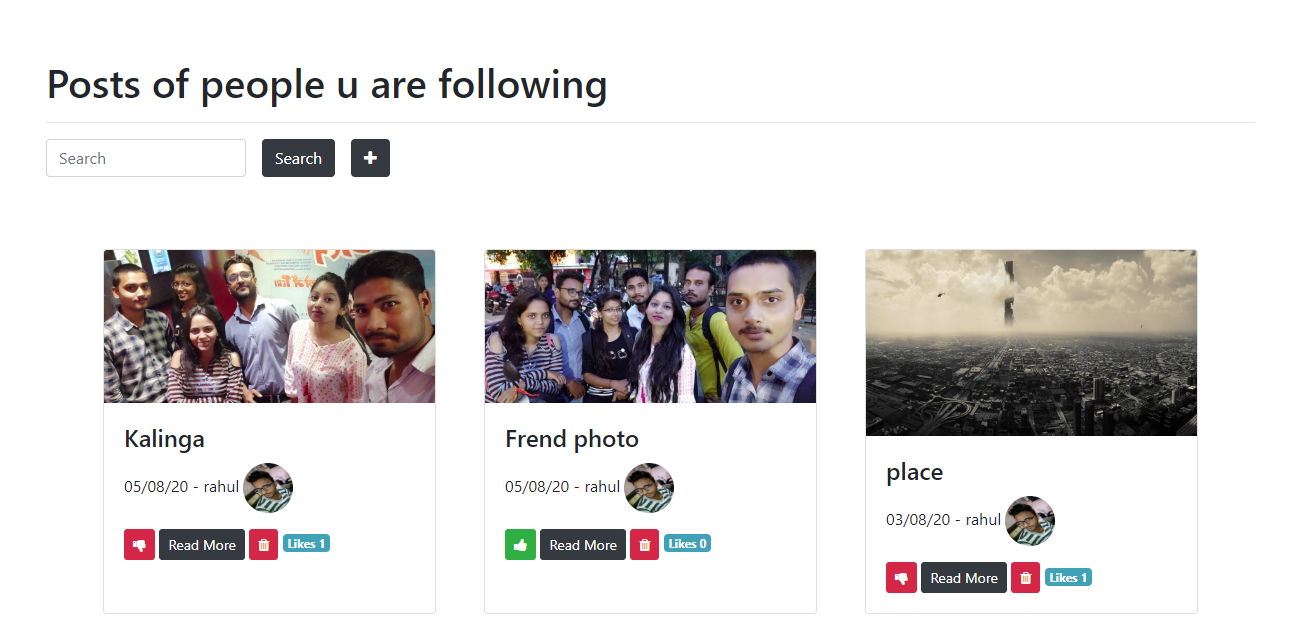


**Profiles (You can Follow and Unfollow other)**



**Edit Profile**

**Post of the people you are following**



Post pic

User Profile Pic

You Can Like

No of Like

# 8 CODING

Install in command

Install app

1. Pip install python
2. Pip install Django
3. Pip install Django-registrations-redux
4. Pip install pillow
5. Pip install –upgrade Django-crispy-form

Model.py (create a table )

from django.db import models

from django.contrib.auth.models import User

from django.db.models.deletion import CASCADE

from django.core.validators import MinValueValidator, RegexValidator

# Create your models here.

class MyProfile(models.Model):

    name = models.CharField(max\_length = 100)

    user = models.OneToOneField(to=User, on\_delete=CASCADE)

    age = models.IntegerField(default=18, validators=[MinValueValidator(18)])

    address = models.TextField(null=True, blank=True)

    status = models.CharField(max\_length=20, default="single", choices=(("single","single"), ("married","married"), ("widow","widow"), ("seprated","seprated"), ("commited","commited")))

    gender = models.CharField(max\_length=20, default="female", choices=(("male","male"), ("female","female")))

    phone\_no = models.CharField(validators=[RegexValidator("^0?[5-9]{1}\d{9}$")], max\_length=15, null=True, blank=True)

    description = models.TextField(null=True, blank=True)

    pic=models.ImageField(upload\_to = "images\\", null=True)

    def \_\_str\_\_(self):

        return "%s" % self.user

class MyPost(models.Model):

    pic=models.ImageField(upload\_to = "images\\", null=True)

    subject = models.CharField(max\_length = 200)

    msg = models.TextField(null=True, blank=True)

    cr\_date = models.DateTimeField(auto\_now\_add=True)

    uploaded\_by = models.ForeignKey(to=MyProfile, on\_delete=CASCADE, null=True, blank=True)

    def \_\_str\_\_(self):

        return "%s" % self.subject

class PostComment(models.Model):

    post = models.ForeignKey(to=MyPost, on\_delete=CASCADE)

    msg = models.TextField()

    commented\_by = models.ForeignKey(to=MyProfile, on\_delete=CASCADE)

    cr\_date = models.DateTimeField(auto\_now\_add=True)

    flag = models.CharField(max\_length=20, null=True, blank=True, choices=(("racist","racist"), ("abbusing","abbusing")))

    def \_\_str\_\_(self):

        return "%s" % self.msg

class PostLike(models.Model):

    post = models.ForeignKey(to=MyPost, on\_delete=CASCADE)

    liked\_by = models.ForeignKey(to=MyProfile, on\_delete=CASCADE)

    cr\_date = models.DateTimeField(auto\_now\_add=True)

    def \_\_str\_\_(self):

        return "%s" % self.liked\_by

class FollowUser(models.Model):

    profile = models.ForeignKey(to=MyProfile, on\_delete=CASCADE, related\_name="profile")

    followed\_by = models.ForeignKey(to=MyProfile, on\_delete=CASCADE, related\_name="followed\_by")

    def \_\_str\_\_(self):

        return "%s is followed by %s" % (self.profile, self.followed\_by)

View.py

from django.shortcuts import render, redirect

from django.views.generic.base import TemplateView

from django.utils.decorators import method\_decorator

from django.contrib.auth.decorators import login\_required

from django.views.generic.list import ListView

from social.models import FollowUser, MyPost, MyProfile, PostComment, PostLike

from django.views.generic.detail import DetailView

from django.db.models import Q

from django.views.generic.edit import UpdateView, CreateView, DeleteView

from django.http.response import HttpResponseRedirect

# Create your views here.

@method\_decorator(login\_required, name="dispatch")

class HomeView(TemplateView):

    template\_name = "social/home.html"

    def get\_context\_data(self, \*\*kwargs):

        context = TemplateView.get\_context\_data(self, \*\*kwargs)

        followedList = FollowUser.objects.filter(followed\_by = self.request.user.myprofile)

        followedList2 = []

        for e in followedList:

            followedList2.append(e.profile)

        postList = MyPost.objects.filter(uploaded\_by\_\_in = followedList2).order\_by("-id")

        for p1 in postList:

            p1.liked = False

            ob = PostLike.objects.filter(post = p1,liked\_by=self.request.user.myprofile)

            if ob:

                p1.liked = True

            obList = PostLike.objects.filter(post = p1)

            p1.likedno = obList.count()

        context["mypost\_list"] = postList

        return context;

class AboutView(TemplateView):

    template\_name = "social/about.html"

class ContactView(TemplateView):

    template\_name = "social/contact.html"

def follow(req, pk):

    user = MyProfile.objects.get(pk=pk)

    FollowUser.objects.create(profile=user, followed\_by = req.user.myprofile)

    return HttpResponseRedirect(redirect\_to="/social/myprofile")

def unfollow(req, pk):

    user = MyProfile.objects.get(pk=pk)

    FollowUser.objects.filter(profile=user, followed\_by = req.user.myprofile).delete()

    return HttpResponseRedirect(redirect\_to="/social/myprofile")

def like(req, pk):

    post = MyPost.objects.get(pk=pk)

    PostLike.objects.create(post=post, liked\_by = req.user.myprofile)

    return HttpResponseRedirect(redirect\_to="/social/home")

def unlike(req, pk):

    post = MyPost.objects.get(pk=pk)

    PostLike.objects.filter(post=post, liked\_by = req.user.myprofile).delete()

    return HttpResponseRedirect(redirect\_to="/social/home")

@method\_decorator(login\_required, name="dispatch")

class MyProfileUpdateView(UpdateView):

    model = MyProfile

    fields = ["name", "age", "address", "status", "gender", "phone\_no", "description", "pic"]

@method\_decorator(login\_required, name="dispatch")

class MyPostCreate(CreateView):

    model = MyPost

    fields = ["subject", "msg", "pic"]

    def form\_valid(self, form):

        self.object = form.save()

        self.object.uploaded\_by = self.request.user.myprofile

        self.object.save()

        return HttpResponseRedirect(self.get\_success\_url())

@method\_decorator(login\_required, name="dispatch")

class MyPostListView(ListView):

    model = MyPost

    def get\_queryset(self):

        si = self.request.GET.get("si")

        if si == None:

            si = ""

        return MyPost.objects.filter(Q(uploaded\_by = self.request.user.myprofile)).filter(Q(subject\_\_icontains = si) | Q(msg\_\_icontains = si)).order\_by("-id");

@method\_decorator(login\_required, name="dispatch")

class MyPostDetailView(DetailView):

    model = MyPost

@method\_decorator(login\_required, name="dispatch")

class MyPostDeleteView(DeleteView):

    model = MyPost

@method\_decorator(login\_required, name="dispatch")

class MyProfileListView(ListView):

    model = MyProfile

    def get\_queryset(self):

        si = self.request.GET.get("si")

        if si == None:

            si = ""

        profList = MyProfile.objects.filter(Q(name\_\_icontains = si) | Q(address\_\_icontains = si) | Q(gender\_\_icontains = si) | Q(status\_\_icontains = si)).order\_by("-id");

        for p1 in profList:

            p1.followed = False

            ob = FollowUser.objects.filter(profile = p1,followed\_by=self.request.user.myprofile)

            if ob:

                p1.followed = True

        return profList

@method\_decorator(login\_required, name="dispatch")

class MyProfileDetailView(DetailView):

    model = MyProfile

Url.py

from django.contrib import admin

from django.urls import path

from django.urls.conf import include

from social import views

from django.views.generic.base import RedirectView

urlpatterns = [

    path('home/', views.HomeView.as\_view()),

    path('about/', views.AboutView.as\_view()),

    path('contact/', views.ContactView.as\_view()),

    path('profile/edit/<int:pk>', views.MyProfileUpdateView.as\_view(success\_url="/social/home")),

    path('mypost/create/', views.MyPostCreate.as\_view(success\_url="/social/mypost")),

    path('mypost/delete/<int:pk>', views.MyPostDeleteView.as\_view(success\_url="/social/mypost")),

    path('mypost/', views.MyPostListView.as\_view()),

    path('mypost/<int:pk>', views.MyPostDetailView.as\_view()),

    path('mypost/like/<int:pk>', views.like),

    path('mypost/unlike/<int:pk>', views.unlike),

    path('myprofile/', views.MyProfileListView.as\_view()),

    path('myprofile/<int:pk>', views.MyProfileDetailView.as\_view()),

    path('myprofile/follow/<int:pk>', views.follow),

    path('myprofile/unfollow/<int:pk>', views.unfollow),

    path('', RedirectView.as\_view(url="home/")),

]

Login.html

{% extends 'base.html' %}

{% block title %}Home{% endblock title %}

{% block content %}

{% load crispy\_forms\_tags %}

<!-- Start : loging form -->

<section class="jumbotron text-center">

    <div class="container">

        <img class="img-fluid" src="/static/images/logo.png" alt="logo">

        <h1 class="jumbotron-heading"><span id="typed">Welcome to E-Sabha</span><span

                class="typed-cursor typed-cursor--blink"></span></h1>

        <p class="lead text-muted">E-Sabha is a web based communication media , which allow user to share information

            ,photos and create contents,

            <br>E-Sabha is a place to share photos, posts & follow some users, you can see their profile and like there

            posts.

        </p>

    </div>

</section>

<div class="container-fluid row featurette align-items-center justify-content-center d-flex ">

    <div class="col-md-5 order-md-2 ">

        <div class="row text-justify">

            <div class="col">

                <h1 class="h3 mb-3 font-weight-normal">Please sign in</h1>

                <hr>

            </div>

        </div>

        <form enctype="multipart/form-data" method="post">

            {% csrf\_token %}

            {{form | crispy}}<br>

            <input class="btn btn-primary " type="submit" />

            <a class="btn btn-success" href="/accounts/register/">Register</a>

            <a class="btn btn-danger" href="/accounts/password/reset/">Reset Password</a>

        </form><br><br>

    </div>

    <div class="col-md-5 order-md-1 p-sm-5">

        <img class=" img-fluid " src="/static/images/contain2.gif" width="500" height="500" alt="logo">

        <h3 class="text-center ">Using programming : HTML, CSS, JAVA SCRIP, BOOTSTRAP</h3>

    </div>

</div>

<!-- Start : Use Full Control -->

<section class="jumbotron text-center ">

    <div class="container">

        <h2 class="p-3">How To Work</h2>

    </div>

    <!-- Start : benifite -->

    <div class="album py-5 bg-light">

            <iframe width="100%" height="415" src="https://www.youtube.com/embed/o75Hp9okEvw" frameborder="0"

                allow="accelerometer; autoplay; encrypted-media; gyroscope; picture-in-picture"

                allowfullscreen></iframe>

    </div>

</section>

<!-- End : Use Full Control  -->

{% endblock %}

Registration.html

{% extends 'base.html' %}

{% block title %}Registration{% endblock title %}

{% block content %}

{% load crispy\_forms\_tags %}

<main class="p-5">

<div class="row text-justify">

    <div class="col">

    <h1 class="myhead2">Register Page</h1>

    <hr>

    </div>

</div>

<form enctype="multipart/form-data" method="post">

{% csrf\_token %}

{{form | crispy}}<br>

<input class="btn btn-primary" type="submit" />

</form>

</main>

{% endblock %}

About.html

{% extends 'base.html' %}

{% block title %}About{% endblock title %}

{% block content %}

<div class="row p-5">

  <div class="col">

    <h1 class="myhead2">About Us</h1>

    <hr>

  </div>

  <div class="row text-justify">

    <div class="col-sm-6">

      <p><b> OBJECTIVE :- </b>E-Sabha is a web based communication media , which allow user to share information ,photos

        and create contents,

        E-Sabha is a place to share photos, posts. It’s a website to connect with people, to be in touch.

        E-Sabha you can follow some users, you can see their profile and like there posts.

      <p><b>PROJECT OVERVIEW :- </b><b>1.</b> ESabha Mission is to give people the power to build community and bring

        the world closer together.

        <b> 2.</b> ESabha can be accessed from devices with Internet connectivity, such as personal computers, tablets

        and Smartphone's.

        <b>3.</b> After registering, users can create a profile revealing information about themselves. They can post

        text and photos which is shared with any other users.

      <p><b> PURPOSE :- </b>ESabha's purpose is to give people the power to build community and bring the world closer

        together. People use ESabha to stay connected with friends and family, to discover what's going on in the world,

        and to share and express what matters to them.

        <br>

    </div>

    <div class="col-sm-6">

      <h2 class=text-center>METHODOLOGY USED</h2><br>

      <table class="table">

        <thead class="thead-dark">

          <tr>

            <th scope="col">#</th>

            <th scope="col">Front End </th>

            <th scope="col">Back End </th>

          </tr>

        </thead>

        <tbody>

          <tr>

            <th scope="row">1</th>

            <td>HTML</td>

            <td>PYTHON</td>

          </tr>

          <tr>

            <th scope="row">2</th>

            <td>CSS</td>

            <td>DJANGO</td>

          </tr>

          <tr>

            <th scope="row">3</th>

            <td>BOOTSTRAP</td>

            <td>FRAMEWORK</td>

          </tr>

          <tr>

            <th scope="row">4</th>

            <td>JAVA SCRIPT</td>

            <td>SQLITE

            </td>

          </tr>

        </tbody>

      </table>

    </div>

    <div class="row my-5">

      <div class="col-sm-12 offset-sm-2">

        <h2 class="text-center">ADVANTAGE & LIMITATIONS</h2><br>

        <table class="table">

          <thead class="thead-dark">

            <tr>

              <th scope="col">#</th>

              <th scope="col">Advantages of E S

              </th>

              <th scope="col">Limitation of ESabha

              </th>

            </tr>

          </thead>

          <tbody>

            <tr>

              <th scope="row">1</th>

              <td>Social media can lead to more communication</td>

              <td>Lacks Emotional Connection</td>

            </tr>

            <tr>

              <th scope="row">2</th>

              <td>Students can use technology to form study groups</td>

              <td>Gives People a License to be Hurtful</td>

            </tr>

            <tr>

              <th scope="row">3</th>

              <td>Social media offers students a way to stay in touch with friends if they move</td>

              <td>Decreases Face-to-Face Communication Skills</td>

            </tr>

          </tbody>

        </table> <br><br><br>

        {% endblock %}

Home.html

{% extends 'base.html' %}

{% block title %}Profile{% endblock title %}

{% block content %}

<div class="p-5">

    <h1 class="myhead2">Posts of people u are following</h1>

    <hr>

    <form class="form-inline">

        <input class="form-control mr-3" placeholder="Search" value="{{ request.GET.si }}" type="text" name="si" />

        <input class="btn btn-dark mr-3" type="submit" value="Search" />

        <a class="btn btn-dark" href="/social/mypost/create"><span class="fa fa-plus"></span></a>

    </form>

    <div class="row p-5">

        {% for n1 in mypost\_list %}

        <div class="col-sm-4 p-4">

            <div class="card h-100">

                <img class="card-img-top" src="/media/{{n1.pic}}" alt="">

                <div class="card-body">

                    <h4 class="card-title">{{n1.subject}}</h4>

                    <p class="card-text">{{n1.cr\_date | date:"d/m/y"}} - {{n1.uploaded\_by}}

                        {% if n1.uploaded\_by.pic %}

                        <img style="width: 50px; height: 50px;" class="img-fluid rounded-circle"

                            src="/media/{{n1.uploaded\_by.pic}}" />

                        {% endif %}

                    </p>

                    {% if n1.liked %}

                    <a class="btn btn-danger btn-sm" href='/social/mypost/unlike/{{n1.id}}'><span

                            class="fa fa-thumbs-down"></span></a>

                    {% else %}

                    <a class="btn btn-success btn-sm" href='/social/mypost/like/{{n1.id}}'><span

                            class="fa fa-thumbs-up"></span></a>

                    {% endif %}

                    <a class="btn btn-dark btn-sm" href='/social/mypost/{{n1.id}}'>Read More</a>

                    {% if user.is\_superuser %}

                    <a class="btn btn-danger btn-sm" href='/social/mypost/delete/{{n1.id}}'><span

                            class="fa fa-trash"></span></a>

                    {% endif %}

                    <span class="badge badge-info">Likes {{n1.likedno}}</span>

                </div>

            </div>

        </div>

        {% endfor %}

    </div>

    {% endblock %}

mypost\_confirm\_delete.html

{% extends 'base.html' %}

{% block content %}

{% load crispy\_forms\_tags %}

<main class="p-5">

<div class="row">

    <div class="col-sm-6 offset-sm-3">

    <h1 class="myhead2">R u sure!</h1>

        <hr>

        <form enctype="multipart/form-data" method="post">

        {% csrf\_token %}

        {{form | crispy}}

        <input class="btn btn-danger my-3" type="submit" />

        </form>

    </div>

</div>

</main>

{% endblock %}

mypost\_detail.html

{% extends 'base.html' %}

{% block content %}

<div class="p-5">

    <h1 class="myhead2">{{mypost.subject}}</h1>

    <hr>

<div class="row">

    <div class="col-sm-6">

    <p><strong>MSG :: </strong> {{mypost.msg}}</p>

    <p><strong>Date :: </strong> {{mypost.cr\_date | date:"d/m/y"}}</p>

    </div>

    <div class="col-sm-6">

        <img class="img-fluid" src="/media/{{mypost.pic}}" />

    </div>

</div>

<a href="/social/mypost" class="btn btn-dark">Back</a>

</div>

{% endblock %}

mypost\_form.html

{% extends 'base.html' %}

{% block content %}

{% load crispy\_forms\_tags %}

<main class="p-5">

<div class="row">

    <div class="col-sm-6 offset-sm-3">

    <h1 class="myhead2">My Post</h1>

        <hr>

        <form enctype="multipart/form-data" method="post">

        {% csrf\_token %}

        {{form | crispy}}

        <input class="btn btn-dark my-3" type="submit" />

        </form>

    </div>

</div>

</main>

{% endblock %}

mypost\_list.html

{% extends 'base.html' %}

{% block title %}Mypost List{% endblock title %}

{% block content %}

<div class="p-5">

<h1 class="myhead2">My Posts </h1>

<hr>

<form class="form-inline">

<input class="form-control mr-3" placeholder="Search" value="{{ request.GET.si }}" type="text" name="si" />

<input class="btn btn-dark mr-3" type="submit" value="Search" />

<a class="btn btn-dark" href="/social/mypost/create"><span class="fa fa-plus"></span></a>

</form>

<table class="table my-3">

    <thead class="thead-dark">

        <tr>

            <th>Subject</th>

            <th>Date</th>

            <th>Action</th>

        </tr>

    </thead>

{% for n1 in mypost\_list %}

        <tr>

            <td>{{n1.subject}}</td>

            <td>{{n1.cr\_date | date:"d/m/y"}}</td>

            <td>

            <a class="btn btn-dark btn-sm" href='/social/mypost/{{n1.id}}'>Read More</a>

            <a class="btn btn-danger btn-sm" href='/social/mypost/delete/{{n1.id}}'><span class="fa fa-trash"></span></a>

            </td>

        </tr>

{% endfor %}

</table>

</div>

{% endblock %}

myprofile\_detail.html

{% extends 'base.html' %}

{% block title %}MyProfile Detials{% endblock title %}

{% block content %}

<div class="p-5">

    <h1 class="myhead2">{{myprofile.name}}</h1>

    <hr>

<div class="row">

    <div class="col-sm-6">

    <p><strong>Age :: </strong> {{myprofile.age}}</p>

    <p><strong>Gender :: </strong> {{myprofile.gender}}</p>

    <p><strong>Address :: </strong> {{myprofile.address}}</p>

    <p><strong>Status :: </strong> {{myprofile.status}}</p>

    <p><strong>Phone No :: </strong> {{myprofile.phone\_no}}</p>

    <p><strong>Description :: </strong> {{myprofile.description}}</p>

    </div>

    <div class="col-sm-6">

        <img class="img-fluid" src="/media/{{myprofile.pic}}" />

    </div>

</div>

<a href="/social/myprofile" class="btn btn-dark">Back</a>

</div>

{% endblock %}

myprofile\_form

{% extends 'base.html' %}

{% block title %}Myprofile form{% endblock title %}

{% block content %}

{% load crispy\_forms\_tags %}

<main class="p-5">

<div class="row">

    <div class="col-sm-6 offset-sm-3">

    <h1 class="myhead2">Profile Update</h1>

        <hr>

        <form enctype="multipart/form-data" method="post">

        {% csrf\_token %}

        {{form | crispy}}

        <input class="btn btn-dark my-3" type="submit" />

        </form>

    </div>

</div>

</main>

{% endblock %}

myprofile\_list.html

{% extends 'base.html' %}

{% block title %}Myprofile List{% endblock title %}

{% block content %}

<div class="p-5">

<h1 class="myhead2">Profiles </h1>

<hr>

<form class="form-inline">

<input class="form-control mr-3" placeholder="Search" value="{{ request.GET.si }}" type="text" name="si" />

<input class="btn btn-dark mr-3" type="submit" value="Search" />

</form>

<table class="table my-3">

    <thead class="thead-dark">

        <tr>

            <th>Name</th>

            <th>Age</th>

            <th>Gender</th>

            <th>Action</th>

        </tr>

    </thead>

{% for n1 in myprofile\_list %}

        <tr>

            <td>{{n1.name}}</td>

            <td>{{n1.age}}</td>

            <td>{{n1.gender}}</td>

            <td>

            <a class="btn btn-dark btn-sm" href='/social/myprofile/{{n1.id}}'>Read More</a>

            {% if n1.followed %}

            <a class="btn btn-danger btn-sm" href='/social/myprofile/unfollow/{{n1.id}}'>Unfollow</a>

            {% else %}

            <a class="btn btn-primary btn-sm" href='/social/myprofile/follow/{{n1.id}}'>Follow</a>

            {% endif %}

            </td>

        </tr>

{% endfor %}

</table>

</div>

{% endblock %}

Bases.html

<!DOCTYPE html>

<html>

<head>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <title>{% block title %} {% endblock title %} | E-Sabha</title>

    <!-- add icon link -->

    <link rel="icon" href="/static/images/logo.png" type="image/x-icon" />

    <link href="/static/vender/bs/css/bootstrap.min.css" rel="stylesheet" />

    <script src="/static/vender/jquery-3.3.1.min.js"></script>

    <script src="/static/vender/popper.min.js"></script>

    <script src="/static/vender/bs/js/bootstrap.min.js"></script>

    <link href="/static/vender/fa/css/font-awesome.min.css" rel="stylesheet" />

    <!--Typed.js-->

    <script type="text/javascript" src="https://cdn.jsdelivr.net/npm/typed.js"></script>

    <link href="/static/style.css" rel="stylesheet" />

</head>

<body>

    <header>

        <nav class="navbar navbar-light p-3 bg-light" style=" background-color: #ffffff!important;">

            <a class="navbar-brand mx-5" href="#"><img class="img-fluid" src="/static/images/logo2.png" alt="logo"

                    width="160" height="140"></a>

            <button class="navbar-toggler collapsed mx-5" type="button" data-toggle="collapse"

                data-target="#navbarsExample01" aria-controls="navbarsExample01" aria-expanded="false"

                aria-label="Toggle navigation">

                <span class="navbar-toggler-icon"></span>

            </button>

            <div class="navbar-collapse collapse" id="navbarsExample01" style="">

                <ul class="navbar-nav mr-auto">

                    <li class="nav-item active">

                        <a class="nav-link mx-5" href="/social/home">Home <span class="sr-only">(current)</span></a>

                    </li>

                    <li class="nav-item">

                        <a class="nav-link mx-5" href="/social/about">About</a>

                    </li>

                    <li class="nav-item">

                        <a class="nav-link mx-5" href="/social/contact">Contact</a>

                    </li>

                    {% if user.is\_authenticated %}

                    <li class="nav-item">

                        <a class="nav-link mx-5" href="/social/mypost">My Posts</a>

                    </li>

                    <li class="nav-item">

                        <a class="nav-link mx-5" href="/social/myprofile">Profiles</a>

                    </li>

                    <li class="nav-item">

                        <a class="nav-link mx-5" href="/accounts/logout">Logout</a>

                    </li>

                    {% else %}

                    <li class="nav-item">

                        <a class="nav-link mx-5" href="/accounts/login">Login</a>

                    </li>

                    {% endif %}

                    {% if user.is\_superuser %}

                    <li class="nav-item">

                        <a class="nav-link mx-5" href="/admin">Admin</a>

                    </li>

                    {% else %}

                    {% if user.is\_authenticated %}

                    <li class="nav-item">

                        <a class="nav-link mx-5" href="/social/profile/edit/{{ user.myprofile.id }}">Edit Profile</a>

                    </li>

                    {% endif %}

                    {% endif %}

            </div>

        </nav>

    </header>

    <!-- <div class="form-group"> -->

    <script>

        window.onload = function () {

            console.log("loaded")

            var typed = new Typed('#typed', {

                strings: ["Welcome to E-Sabha", "E-Sabha is a web based communication media ", "Share Photos & Posts", "Share Photos Like & Commands", "Follow some users",],

                backSpeed: 15,

                smartBackspace: true,

                backDelay: 1200,

                startDelay: 1000,

                typeSpeed: 25,

                loop: true,

            });

        };

    </script>

    {% block content %}

    {% endblock %}

    <footer class="myfooter text-center p-1">

        <div class="row">

            <div class="col-sm-6">

                <h2>Contact Us</h2>

                <p>+91 8770099860</p>

            </div>

            <div class="col-sm-6">

                <h2>Social Media</h2>

                <a href="https://www.facebook.com/EffconTech/"><span

                        class="fa fa-facebook-official fa-2x mx-2"></span></a>

                <a href="https://twitter.com/matsuniversity?lang=en"><span class="fa fa-twitter fa-2x  mx-2"></span></a>

                <a href="https://www.youtube.com/channel/UCMZwWDcN1q5qphqzRa44LKA"><span

                        class="fa fa-youtube fa-2x  mx-2"></span></a>

            </div>

        </div>

        <p>Copyright @ E-Sabha 2020</p>

    </footer>

</body>

</html>

# 9 Conclusion

Our portal want to connect all the people each other who have some information to share and who need it, to bring together people with different location so they can understand each other better, and to empower everyone to share their information with their loved one in same location or belonging to the rest of the world.

# 10 BIBLIOGRAPHY

**BOOKS:**

* **Python Crash Course: A Hands-On, Project-Based Introduction to Programming (2nd Edition)**
* **Head-First Python: A Brain-Friendly Guide (2nd Edition)**
* **Learn Python the Hard Way: 3rd Edition**
* [Build your first website with Django 2.1](https://amzn.to/2QWyeCR)
* Mastering Django: Core – The Complete Guide to Django 1.8 LTS
* The Definitive Guide to Django: Web Development Done Right

**Online Learn**

* [**https://www.djangoproject.com/**](https://www.djangoproject.com/)
* [**https://getbootstrap.com/docs/4.0/getting-started/introduction/**](https://getbootstrap.com/docs/4.0/getting-started/introduction/)

**WEB SITES**

FOR PYTHON AND DJANGO INSTALLATION

* Pip install python
* Pip install Django

**Free hosting Website**

* [**www.pythonanywere.com**](http://www.pythonanywere.com)

**rajesh007.pythonanywere.com**

**Index**

[1 INTRODUCTION 1](#_Toc47536599)

[1.1 OVERVIEW 2](#_Toc47536600)

[1.2 OBJECTIVE 2](#_Toc47536601)

[1.3SCOPE 2](#_Toc47536602)

[2.1 4](#_Toc47536603)

[2.2 SOLUTIONOFTHESEPROBLEMS 4](#_Toc47536604)

[3 PROJECT SIGNIFICANCE 5](#_Toc47536605)

[3.1 STUDY OF THE SYSTEM 6](#_Toc47536606)

[3.2 Data Flow Diagram 8](#_Toc47536607)

[3.3 ER Diagram 10](#_Toc47536608)

[3.4 STRUCTURE OF DATABASE 11](#_Toc47536609)

[4 SYSTEM SPECIFICATION 12](#_Toc47536610)

[4.1 SOFTWRE SPECIFICATION 13](#_Toc47536611)

[4.2 HARDWARE SPECIFICATION 13](#_Toc47536612)

[5 PROPOSED SYSTEM 14](#_Toc47536613)

[6 EXISITING SYSTEM 16](#_Toc47536614)

[7 SCREENSHOTS 18](#_Toc47536615)

[My Posts List (User post photo List and Search And Delete) 22](#_Toc47536616)

[8 CODING 25](#_Toc47536617)

[9 Conclusion 44](#_Toc47536618)

[10 BIBLIOGRAPHY 46](#_Toc47536619)