A PROJECT REPORT
ON

SOCIAL MEDIA WEBSITE

BY

Mandalia Jaynish (CE064) 19CEUBS100
Matroja Seturaj (CE067) 19CEUOS104

Bachelors of Technology
Semester IV

Subject: Software Engineering

Guided By:

Dr. Brijesh S. Bhatt Professor
Prof. Jigar M. Pandya Assistant Professor
Prof. Ankit P. Vaishnav Assistant Professor
Prof. Pinkal C. Chauhan Assistant Professor
Dept. of Computer Engg.

Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University

Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University

CERTIFICATE

This is to certify that the practical/term work carried out in the subject of Software Engineering and recorded in the journal is the bona fide work of
Mandalia Jaynish (CE064) 19CEUBS100
Matroja Seturaj (CE067) 19CEUOS104
Of B.Tech Semester IV in branch of Computer Engineering
during the academic year 2020-2021

Prof. Pinkal C. Chavda
Assistant Professor,
Dept. Of Computer Engg.
Faculty of technology
Dharmsinh Desai University,
Nadiad

Dr. C. K. Bhensdadia
Head,
Dept. of Computer Engg.
Faculty of technology
Dharmsinh Desai University,
Nadiad

Table of Contents

1. Abstract and Introduction

Abstract

Social Media provides easier and accessible opportunities to interact with each other's. We can interact with someone who is staying far away from us.

Many websites and applications are there in the market to connect with our friends and family. This websites/Apps are very effective and efficient way to communicate to any person.

Introduction

Social Media provides easier and accessible opportunities to interact with each other's. We can interact with someone who is staying far away from us.

Many websites and applications are there in the market to connect with our friends and family. This websites/Apps are very effective and efficient way to communicate to any person.

In our website mainly two user are there. Admin and all other user which are taking service of our website.

Our website can be used by anybody. It is a simple social media website by which we can see our friends post, status and we can be connected with them.

The new user first has to register in our website and then he/she can login to use the website. He/She can post their pictures in our website. They can also delete them after posting it and can update also.
The user can also make friends with other users registered on website. They can see each other's photos, like, comment on them also. All user can logout also. Information of users will be stored at local database. And all the data is safe like no one can see other's password (Even admin can't see the user's password).

Technologies/Tools Used:

- Platform used: Visual Studio code 2021 for Admin side, Local development server
- Technology: Django
- Languages: Python, HTML, CSS, JS, BootStrap
- Database: Django database

2. Software Requirement Specifications

Social Media Website

Manage User:

Sign-up:
I/p: New to website? Please create your account

O/p: Your account is created now you are able to login

Log-in:

I/p: Enter Email address and Password
O/p: You are redirected to home page (with no feed)

Profile:

Check-Profile:

I/p: Click on the display picture
O/p: Redirected to Profile

Update-Profile:

I/p: Make some changes to profile (Like upload new DP) and click update
O/p: Profile page changed

Post:

Upload-Post:

I/p: You have to choose Picture from Device
O/p: Post will be uploaded

Friends:

Friends-List:

I/p: Check the list in navigation bar
O/p: Redirected to list

Friends-Request:

I/p: Can send the request
O/p: Request sended

Friends-Request-Accept:

I/p: You will be notified when request will arrive chose accept to be friend
O/p: You will become friend or you might not (if you chose reject)

Like/Comments:

Check-Likes:

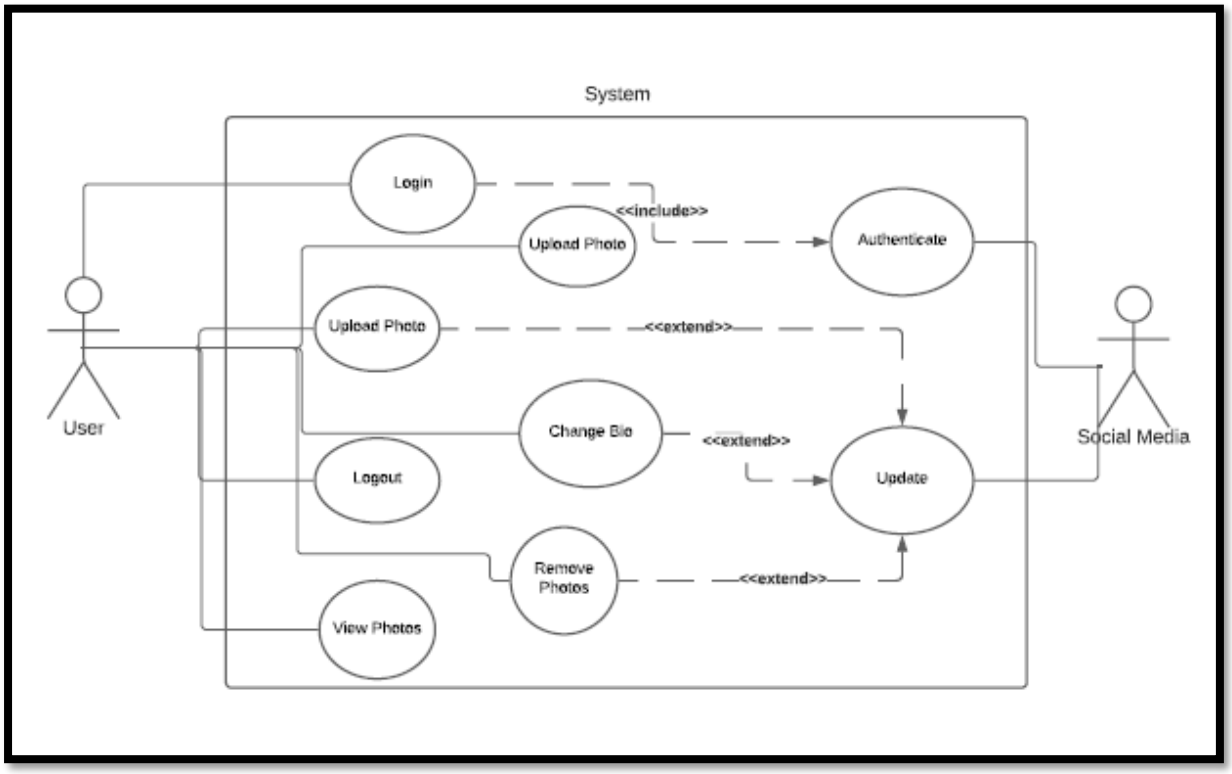
I/p: Like the picture
O/p: Like count will incremented

Check-comments:

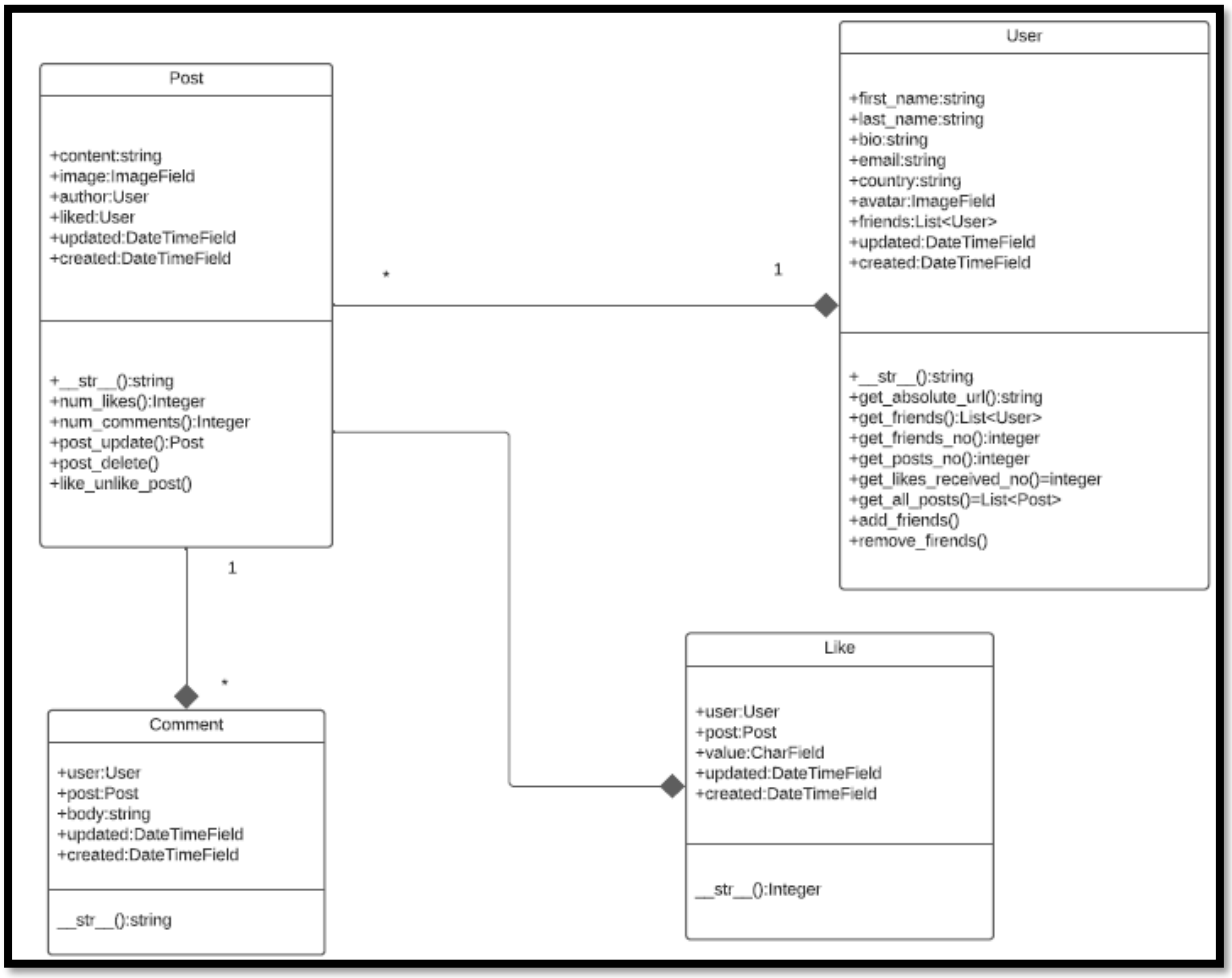
Read Someone's comment. And can reply also.

3. Design Documents:

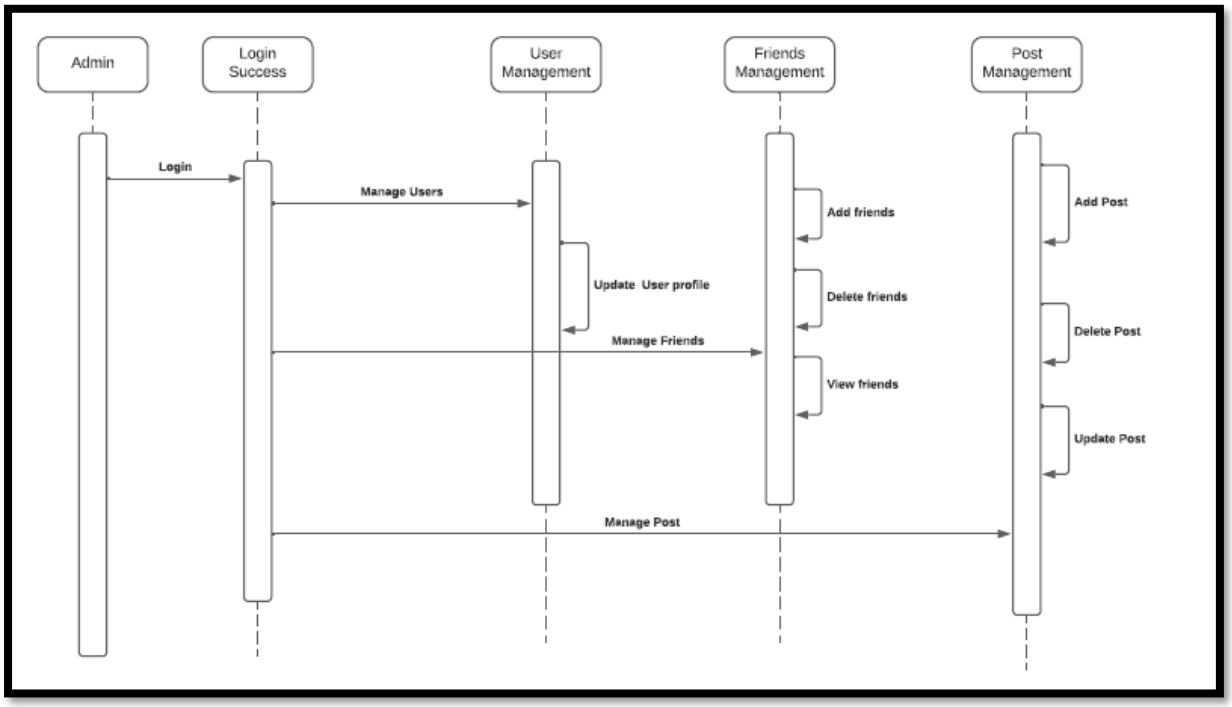
Use Case Diagram:



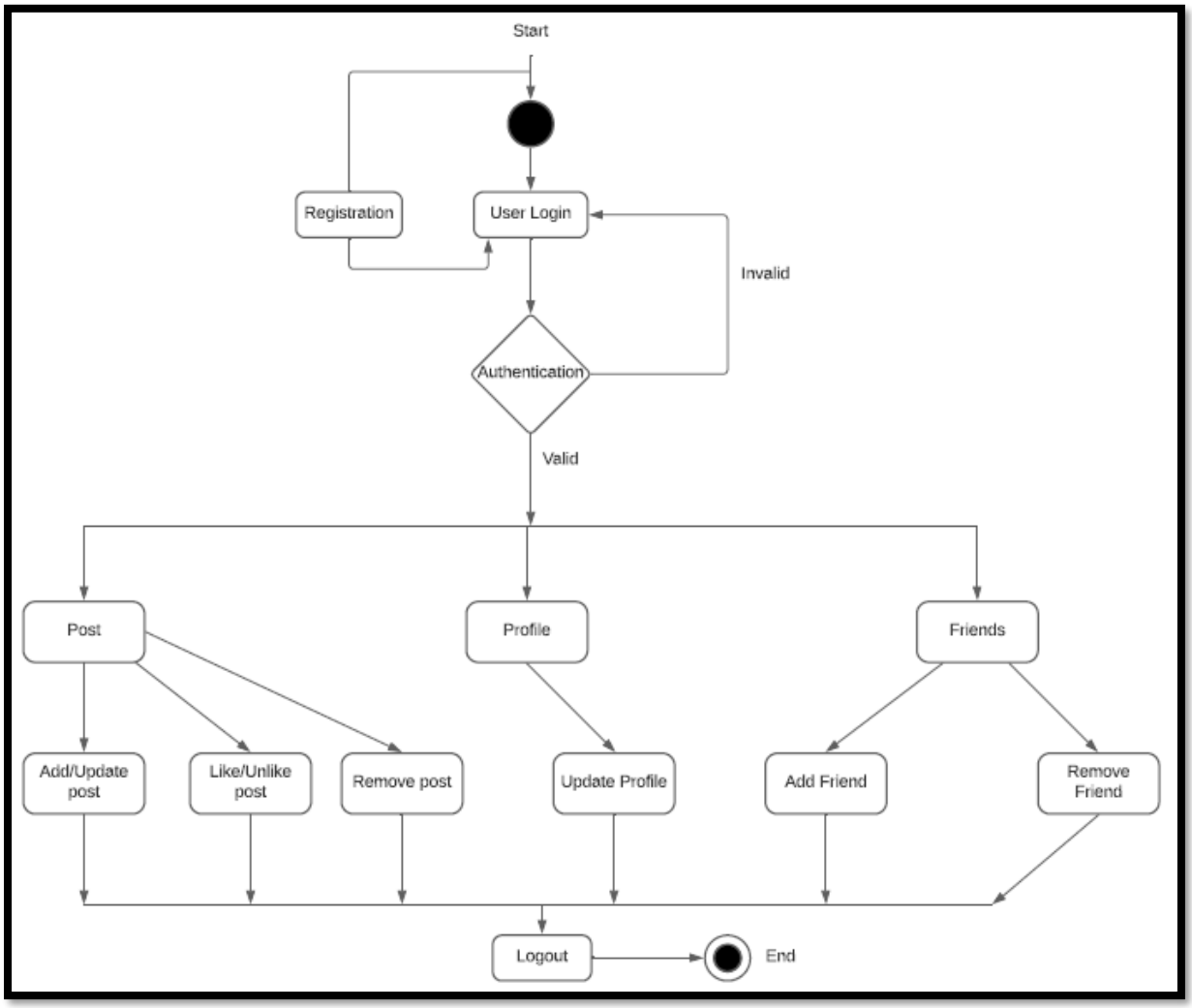
Class Diagram:



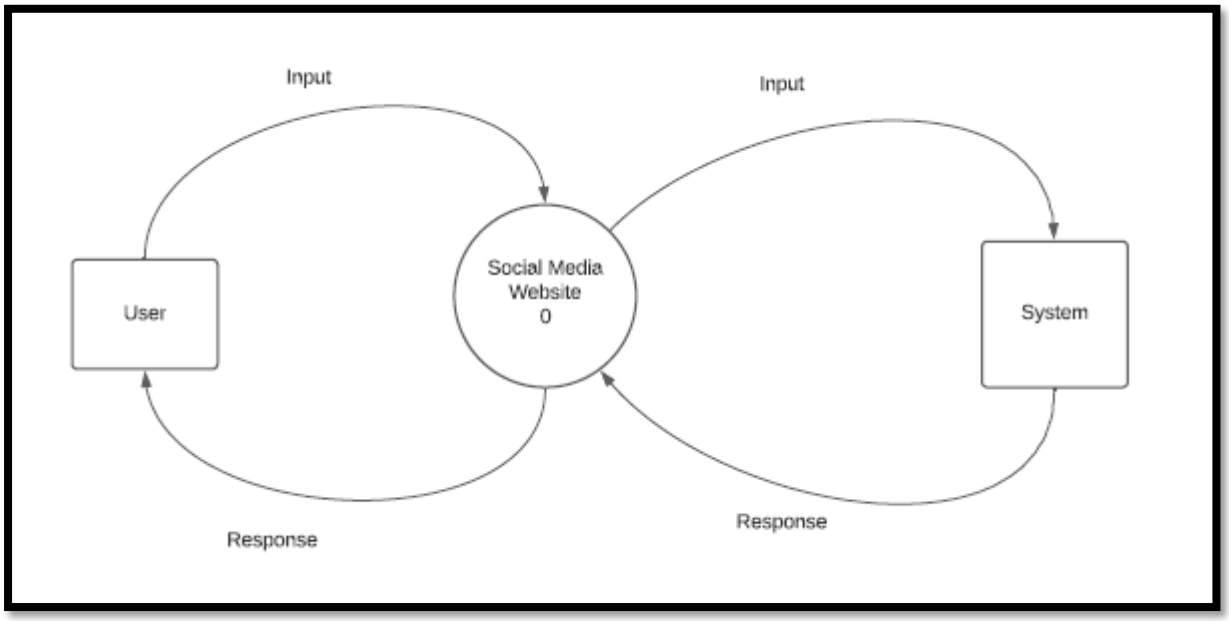
Sequence Diagram:



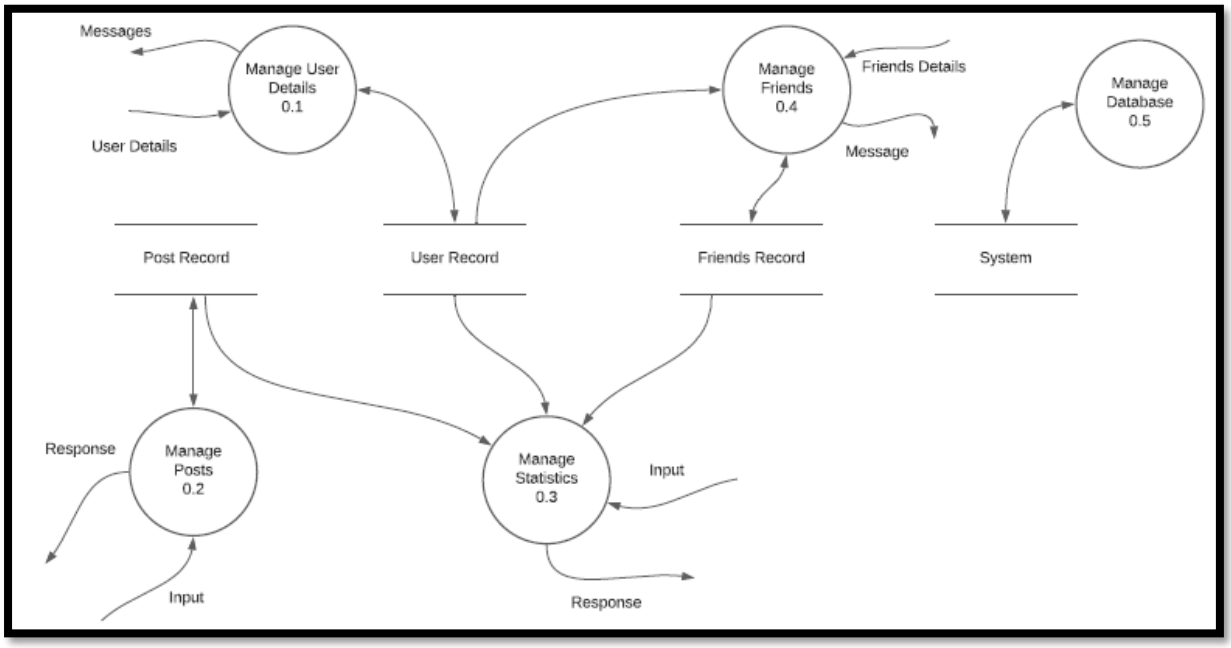
Activity Diagram:



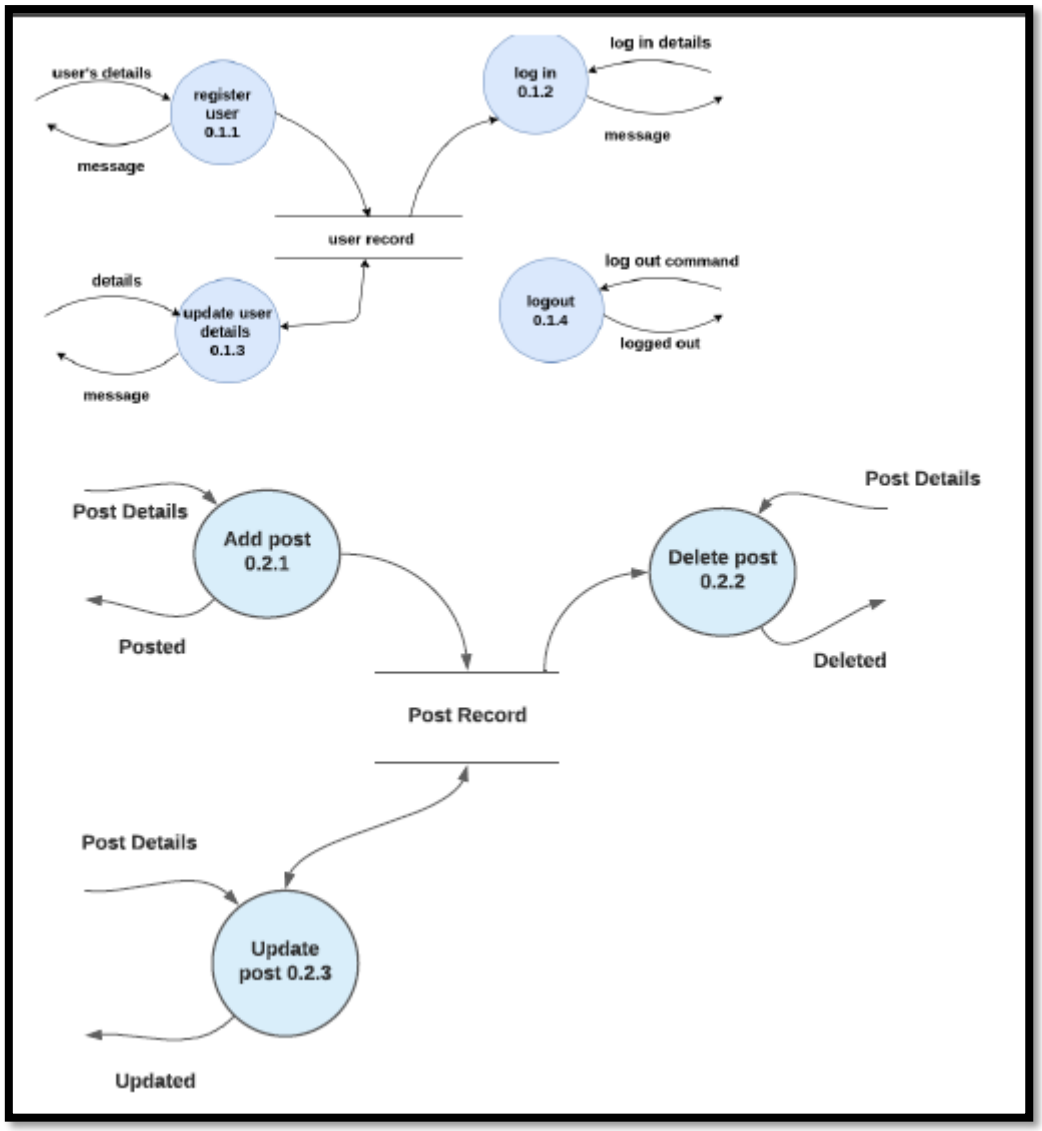
DFD Diagram:

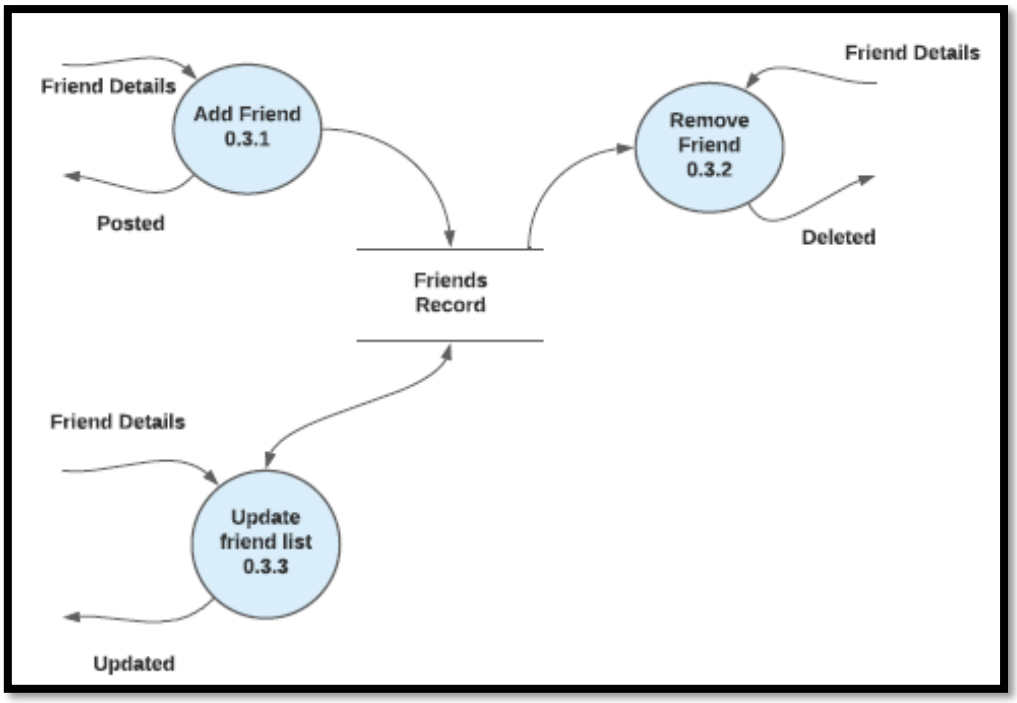


Level 0 (Context diagram)



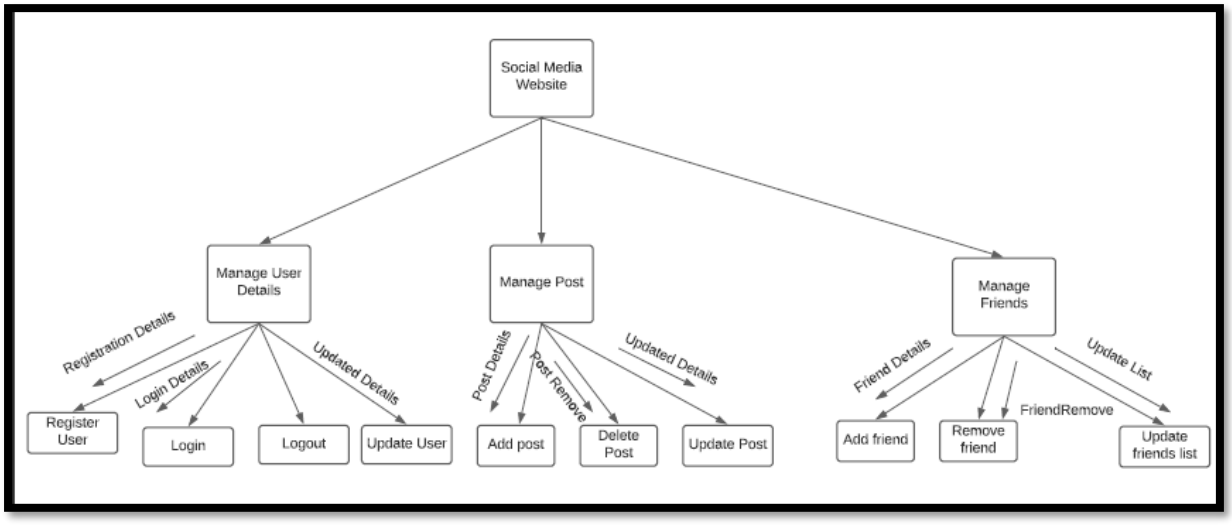
Level 1


user's details
register user 0.1.1
message
log in 0.1.2
log in details
message
user record
details
update user details 0.1.3
message
logout 0.1.4
log out command
logged out
Post Details
Add post 0.2.1
Posted
Delete post 0.2.2
Post Details
Deleted
Post Record
Post Details
Update post 0.2.3
Updated



Level 2

Structure Chart:



Data Dictionary:

User								
Sr no.	Field Name	Data Type	Width	Unique	Default	Primary Key/ Foreign Key	Referred Table	Description
1	Id	Int	255	Yes	None	Yes	-	Auto_Increment
2	First_name	varchar	200	No	None	No	-	-
3	Last_name	varchar	200	No	None	No	-	-
4	Bio	varchar	300	No	No Bio	No	-	-
5	email	varchar	200	Yes	None	No	-	-
6	country	varchar	200	No	None	No	-	-
7	avatar	image	-	No	No image	No	-	-
8	friends	list<user>	-	No	No Friends	No	-	-
9	updated	DateTime	-	No	Current timestamp	No	-	-
10	created	DateTime	-	No	Current timestamp	No	-	-

relationship								
Sr no.	Field Name	Data Type	width	Unique	Default	Primary Key/ Foreign Key	Referred Table	Description
1	sender	list <user>	-	No	None	Yes	User	-
2	receiver	list <user>	-	No	None	Yes	User	-
3	status	char	8	No	Pending	-	-	-
4	updated	DateTime	-	No	Current Timestamp	-	-	-
5	created	DateTime	-	No	Current Timestamp	-	-	-

post								
Sr no.	Field Name	Data Type	width	Unique	Default	Primary Key/ Foreign Key	Referred Table	Description
1	content	char	-	No	None	-	-	-
2	image	image	-	No	None	-	-	upload_to= ‘posts’
3	liked	DateTime	-	No	Not liked	-	-	-
4	updated	DateTime	-	No	Current Timestamp	-	-	-
5	created	DateTime	-	No	Current Timestamp	-	-	-
6	author	User	-	No	-	Yes	User	-

comment								
Sr no.	Field Name	Data Type	width	Unique	Default	Primary Key/ Foreign Key	Referred Table	Description
1	user	User	-	No	None	Yes	User	-
2	post	image	-	No	None	Yes	Post	-
3	body	char	-	No	Not liked	-	-	-
4	updated	DateTime	-	No	Current Timestamp	-	-	-
5	created	DateTime	-	No	Current Timestamp	-	-	-

like								
Sr no.	Field Name	Data Type	width	Unique	Default	Primary Key/ Foreign Key	Referred Table	Description
1	user	User	-	No	None	Yes	User	-
2	post	image	-	No	None	Yes	Post	-
3	value	char	8	No	0	-	-	-
4	updated	DateTime	-	No	Current Timestamp	-	-	-
5	created	DateTime	-	No	Current Timestamp	-	-	-

4. Implementation Details

The system consists of 3 major modules:

1. User Module
2. Friends Module
3. Posts Module

Each module consists of major methods to implement the required functionality. The implementation is done using Django and the database used is Django database.

User Module

This module is for the login and logout purpose. This module is also used in creating profile for the new users and for updating also.

Friends Module

This module handles the friend request part where the users can add or remove friends. They can also see the list of friends. Furthermore, they can also remove the friends after.

Post Module

This module handles the post part where the user can add his/her post including caption and see their friends

post also. After uploading post, the user can delete or update it. The friends can comment and like on the post. The user can also reply to those comments by commenting on the post.

Function Prototypes

```
@login_required
def send_invatation(request):
    if request.method=='POST':
        pk = request.POST.get('profile_pk')
        user = request.user
        sender = Profile.objects.get(user=user)
        receiver = Profile.objects.get(pk=pk)

        rel = Relationship.objects.create(sender=sender, receiver=receiver, status='send')

        return redirect(request.META.get('HTTP_REFERER'))
    return redirect('profiles:my-profile-view')
```

Send Friend Requests

```python
@login_required
def accept_invatation(request):
    if request.method=="POST":
        pk = request.POST.get('profile_pk')
        sender = Profile.objects.get(pk=pk)
        receiver = Profile.objects.get(user=request.user)
        rel = get_object_or_404(Relationship, sender=sender, receiver=receiver)
        if rel.status == 'send':
            rel.status = 'accepted'
            rel.save()
    return redirect('profiles:my-invites-view')

@login_required
def reject_invatation(request):
    if request.method=="POST":
        pk = request.POST.get('profile_pk')
        receiver = Profile.objects.get(user=request.user)
        sender = Profile.objects.get(pk=pk)
        rel = get_object_or_404(Relationship, sender=sender, receiver=receiver)
        rel.delete()
    return redirect('profiles:my-invites-view')
```

Accept and Reject Friend Requests

```python
@login_required
def remove_from_friends(request):
    if request.method=='POST':
        pk = request.POST.get('profile_pk')
        user = request.user
        sender = Profile.objects.get(user=user)
        receiver = Profile.objects.get(pk=pk)

        rel = Relationship.objects.get(
            (Q(sender=sender) & Q(receiver=receiver)) | (Q(sender=receiver) & Q(receiver=sender))
        )
        rel.delete()
        return redirect(request.META.get('HTTP_REFERER'))
    return redirect('profiles:my-profile-view')
```

```python
@receiver(pre_delete, sender=Relationship)
def pre_delete_remove_from_friends(sender, instance, **kwargs):
    sender = instance.sender
    receiver = instance.receiver
    sender.friends.remove(receiver.user)
    receiver.friends.remove(sender.user)
    sender.save()
    receiver.save()
```

Remove Friends

```python
@login_required
def post_comment_create_and_list_view(request):
    qs = Post.objects.all()
    profile = Profile.objects.get(user=request.user)
    # initials
    p_form = PostModelForm()
    c_form = CommentModelForm()
    post_added = False

    profile = Profile.objects.get(user=request.user)

    if 'submit_p_form' in request.POST:
        print(request.POST)
        p_form = PostModelForm(request.POST, request.FILES)
        if p_form.is_valid():
            instance = p_form.save(commit=False)
            instance.author = profile
            instance.save()
            p_form = PostModelForm()
            post_added = True

    if 'submit_c_form' in request.POST:
        c_form = CommentModelForm(request.POST)
        if c_form.is_valid():
            instance = c_form.save(commit=False)
            instance.user = profile
            instance.post = Post.objects.get(id=request.POST.get('post_id'))
            instance.save()
            c_form = CommentModelForm()
    context = {
        'qs': qs,
        'profile': profile,
        'p_form': p_form,
        'c_form': c_form,
        'post_added': post_added,
    }
    return render(request, 'posts/main.html', context)
```

Add new post and comment

```python
@login_required
def like_unlike_post(request):
    user = request.user
    if request.method == 'POST':
        post_id = request.POST.get('post_id')
        post_obj = Post.objects.get(id=post_id)
        profile = Profile.objects.get(user=user)

        if profile in post_obj.liked.all():
            post_obj.liked.remove(profile)
        else:
            post_obj.liked.add(profile)

        like, created = Like.objects.get_or_create(user=profile, post_id=post_id)

        if not created:
            if like.value=='Like':
                like.value='Unlike'
            else:
                like.value='Like'
        else:
            like.value='Like'

            post_obj.save()
            like.save()

        # data = {
        #     'value': like.value,
        #     'likes': post_obj.liked.all().count()
        # }

        # return JsonResponse(data, safe=False)
    return redirect('posts:main-post-view')
```

Like and unlike the post

5. Testing

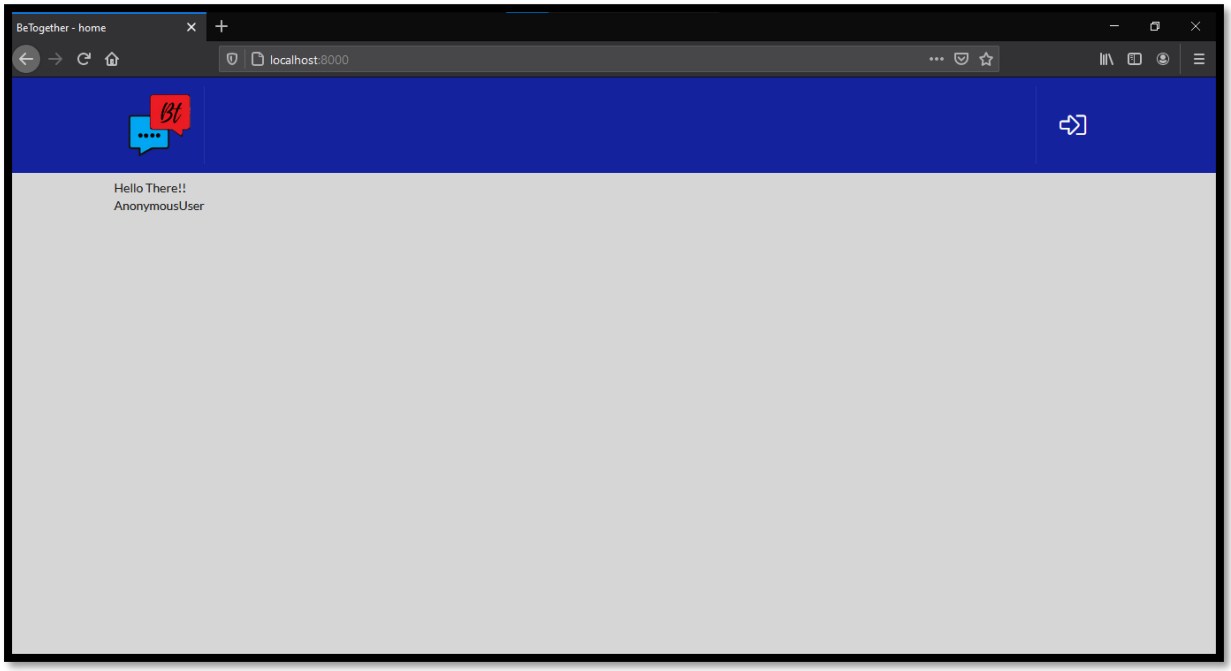
Manual testing was performed in order to find and fix the bugs in development process.

Testing Method: Manual Testing

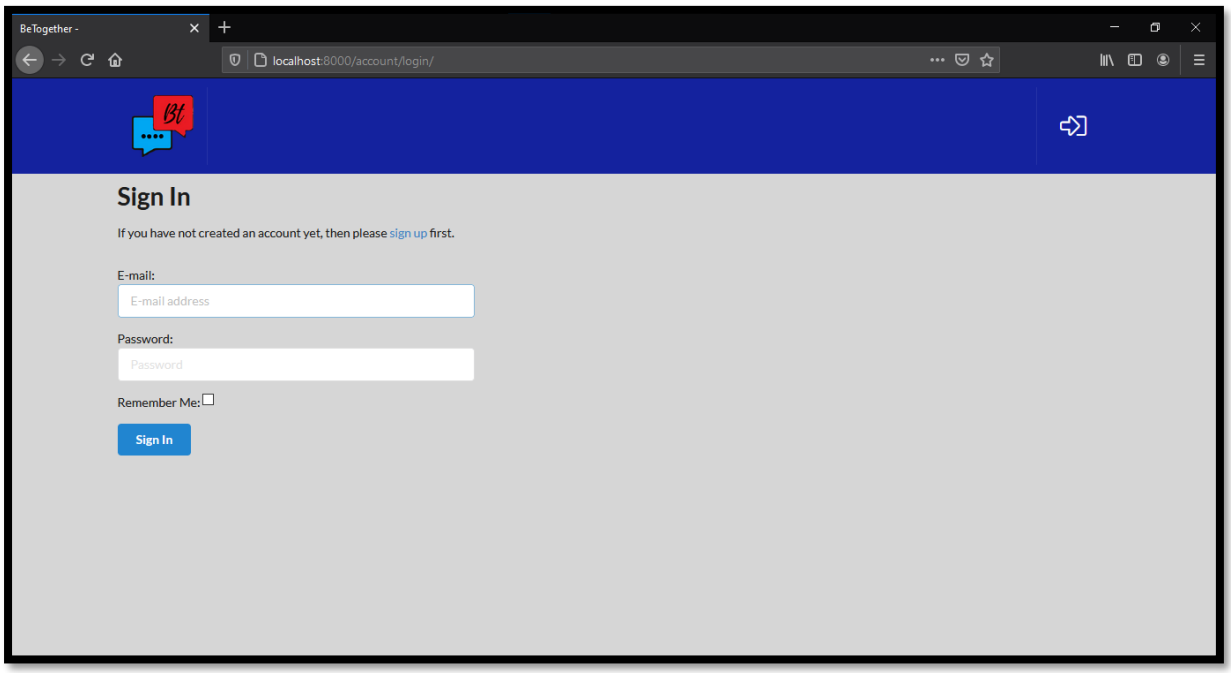
Sr No.	Test Scenario	Expected Result	Actual Result	Status
1	Login with incorrect credentials	Invalid Credentials, please try again	Invalid Credentials, please try again	Success
2	Login with correct credentials	You are Successfully logged in	You are Successfully logged in	Success
3	Confirmation of password in Register	Confirm your password again	Confirm your password again	Success
4	Registering with existing username	Please select unique username	Please select unique username	Success
5	Registering with existing Email	Please select unique Email address	Please select unique Email Address	Success
6	Registering with valid	Redirected to Home	Redirected to Home page	Success

	Credentials	page		
7	Logout	User should be logged out and redirected to Hello Page.	User is successfully logged out of the system	Success
8	Update Profile	Can update profile whenever you want	Profile updated successfully	Success
9	Add new post	Can add new image	Image will be there in home page	Success
10	Delete post	Can delete post	Image will be deleted from home page and account also	Success
11	Give comments	Can add Comment	Comment is visible to both, current user and friend	Success

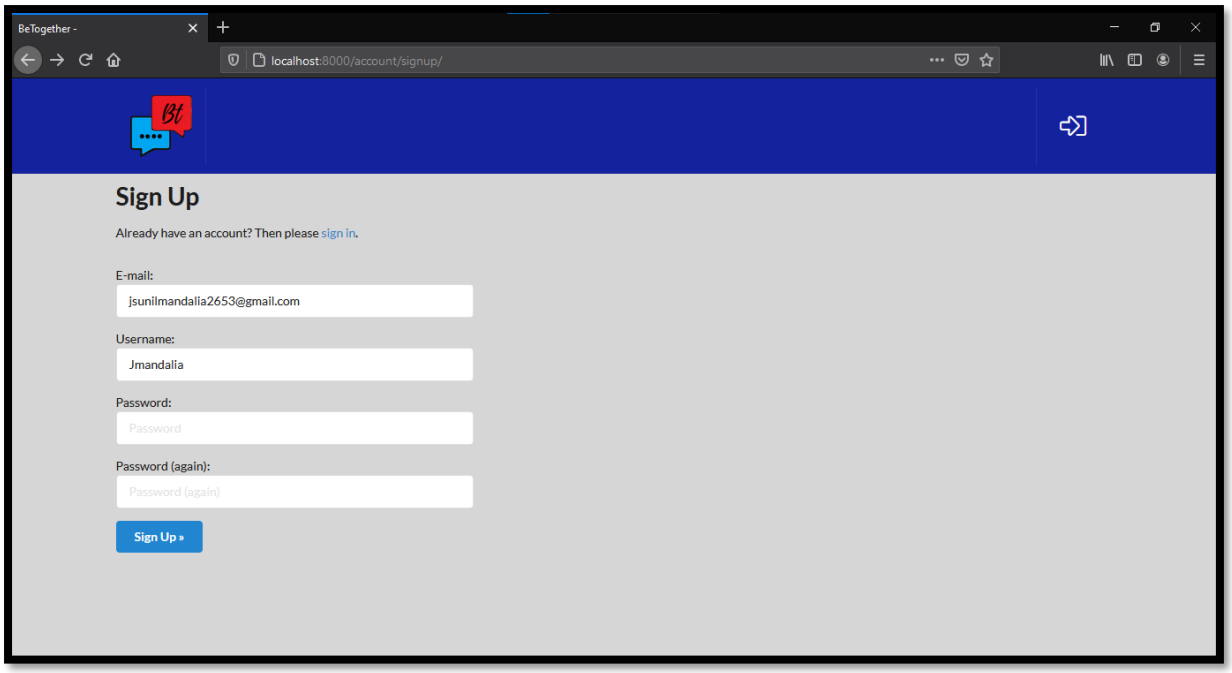
6. Screenshots



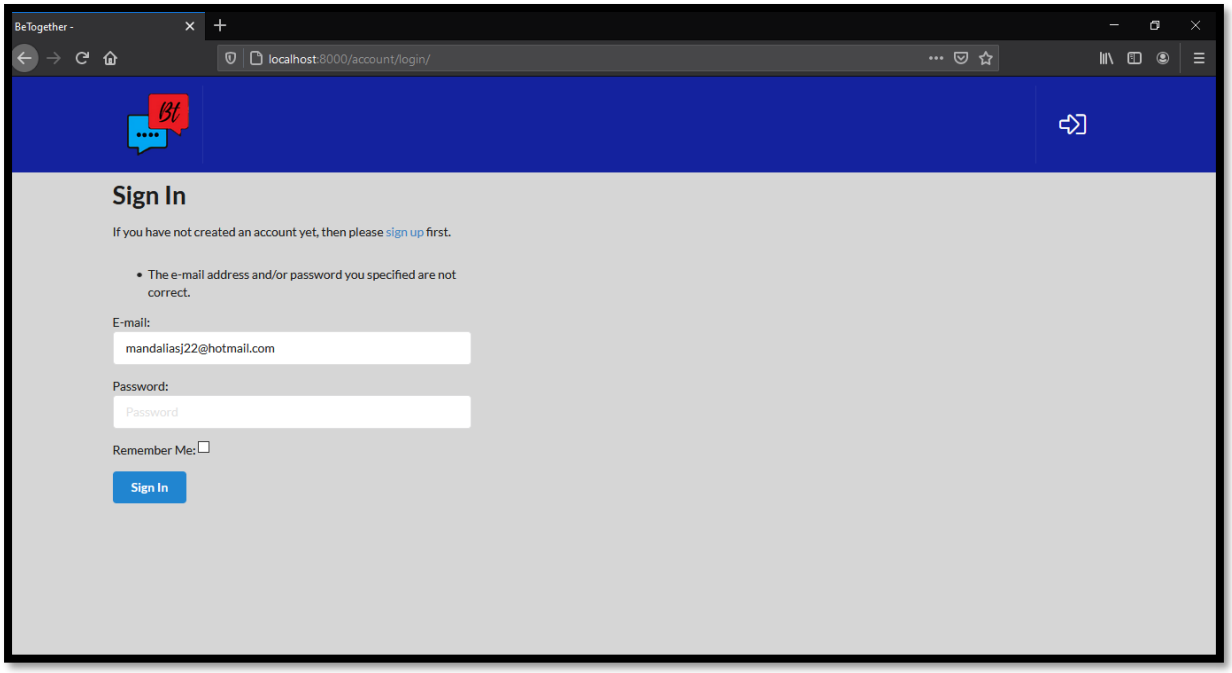
Welcome Page



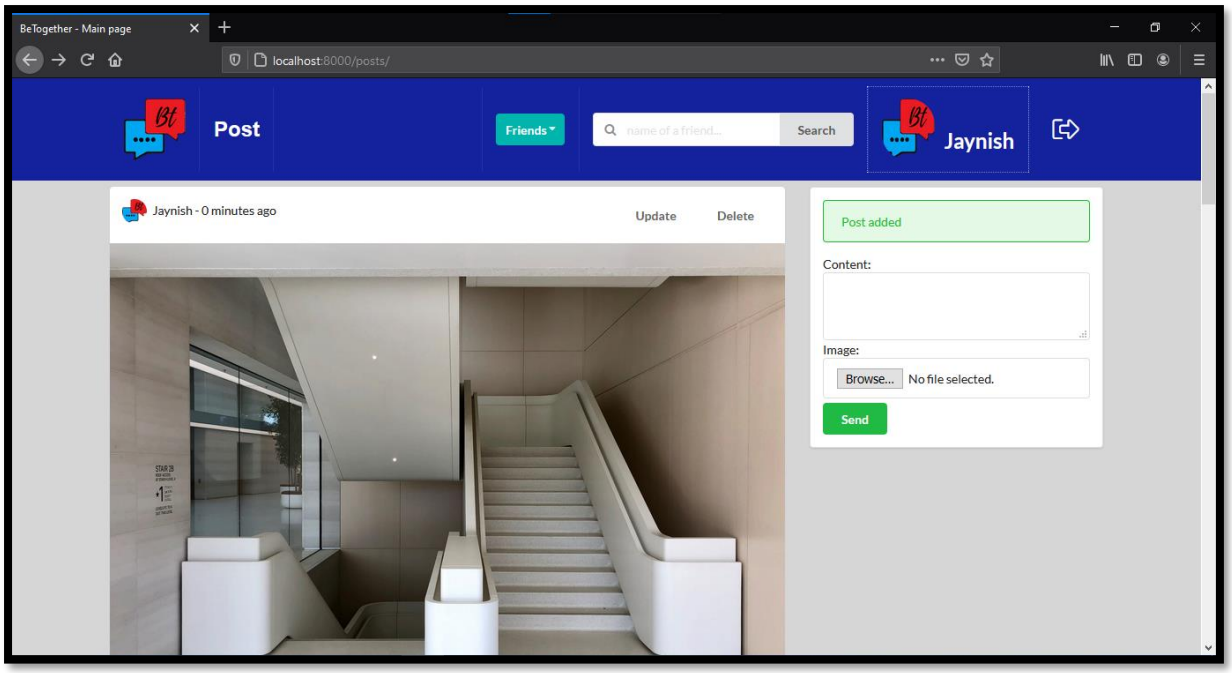
Sign In Page



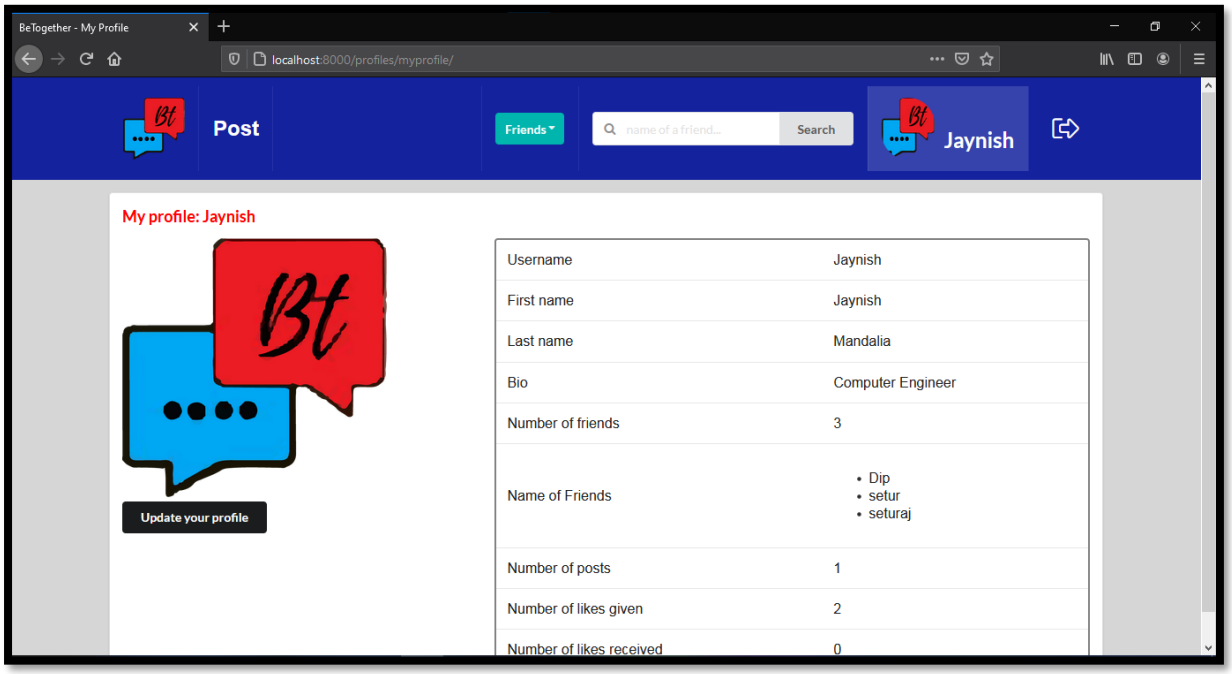
Sign Up page



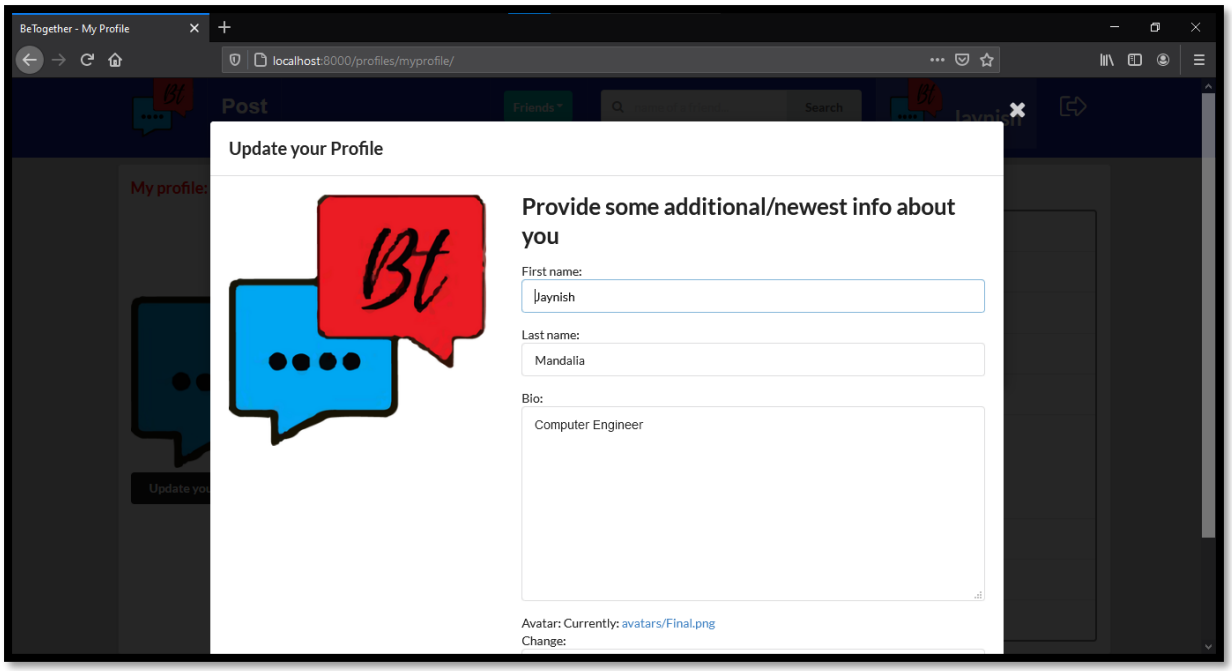
Invalid ID/Password notification



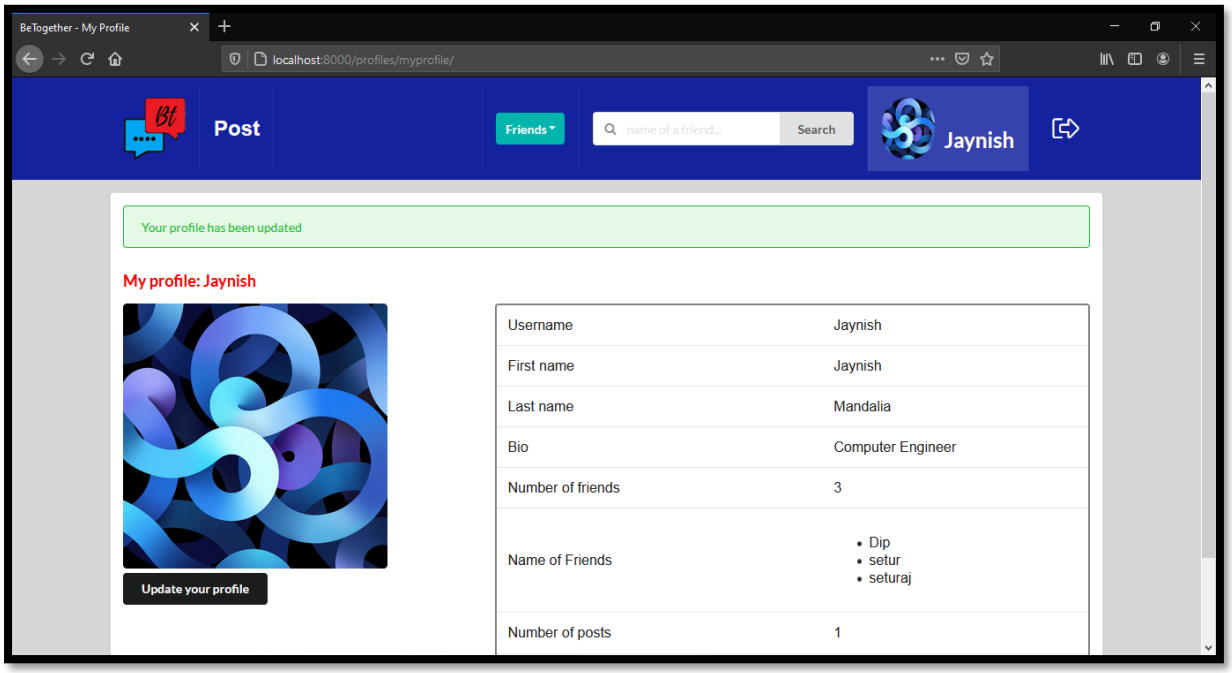
Post Uploaded at home page



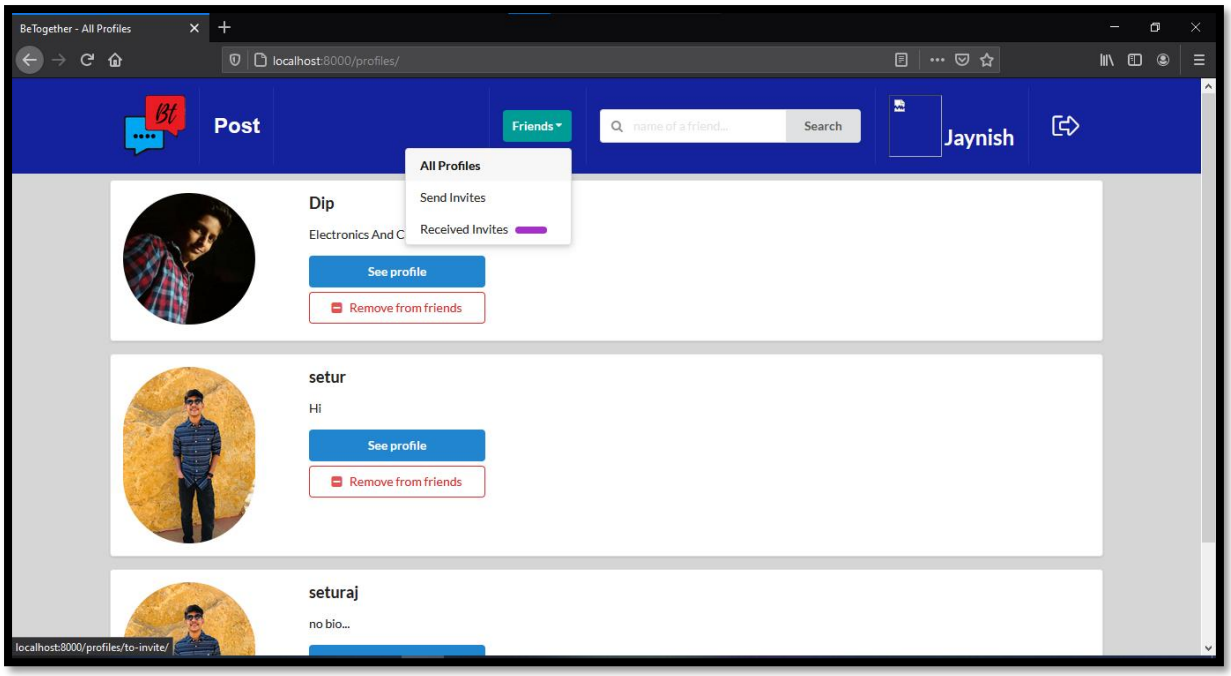
Profile page



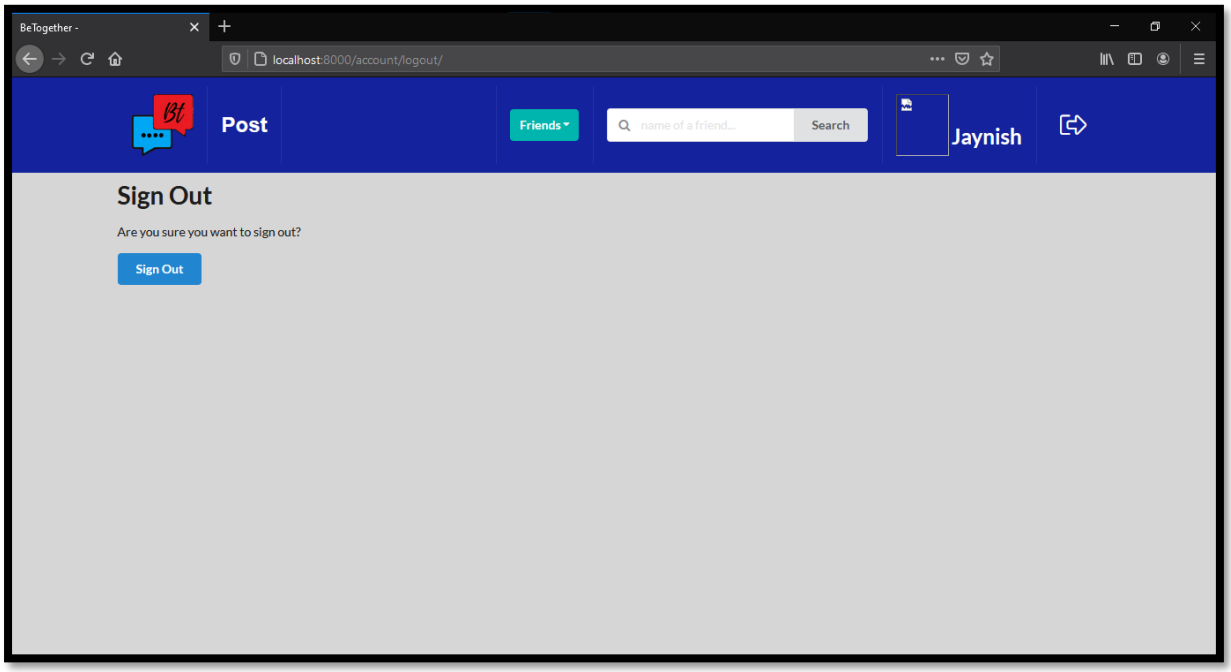
Update Profile



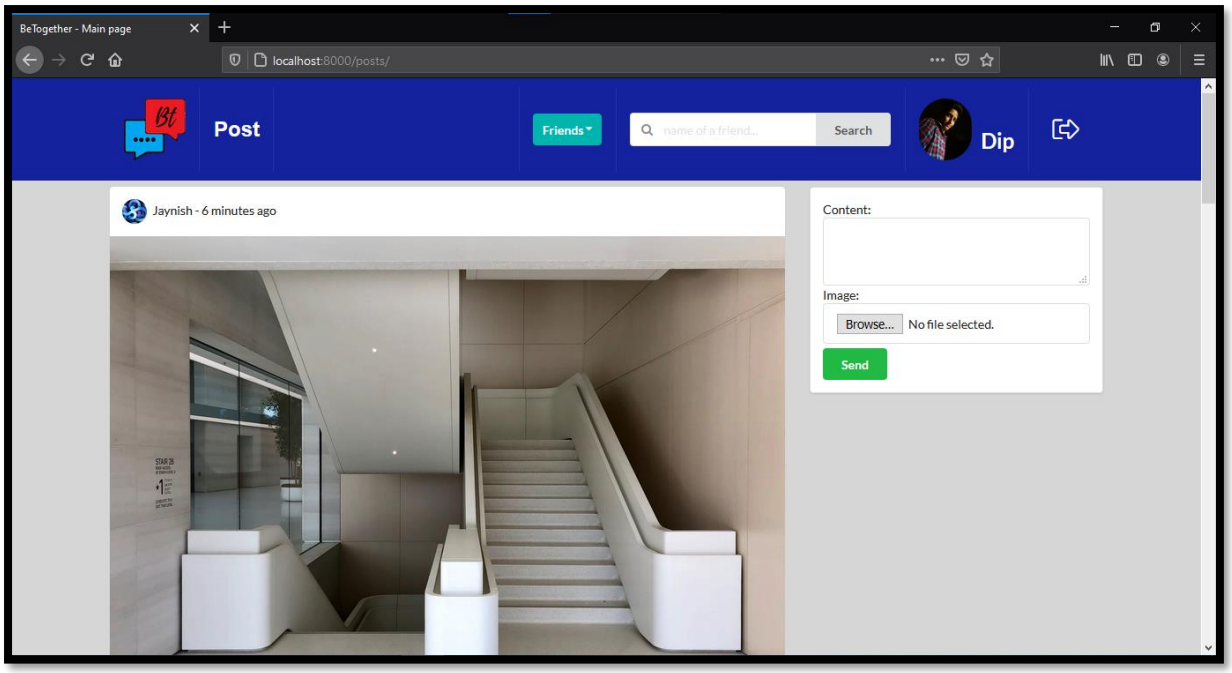
Updated Profile



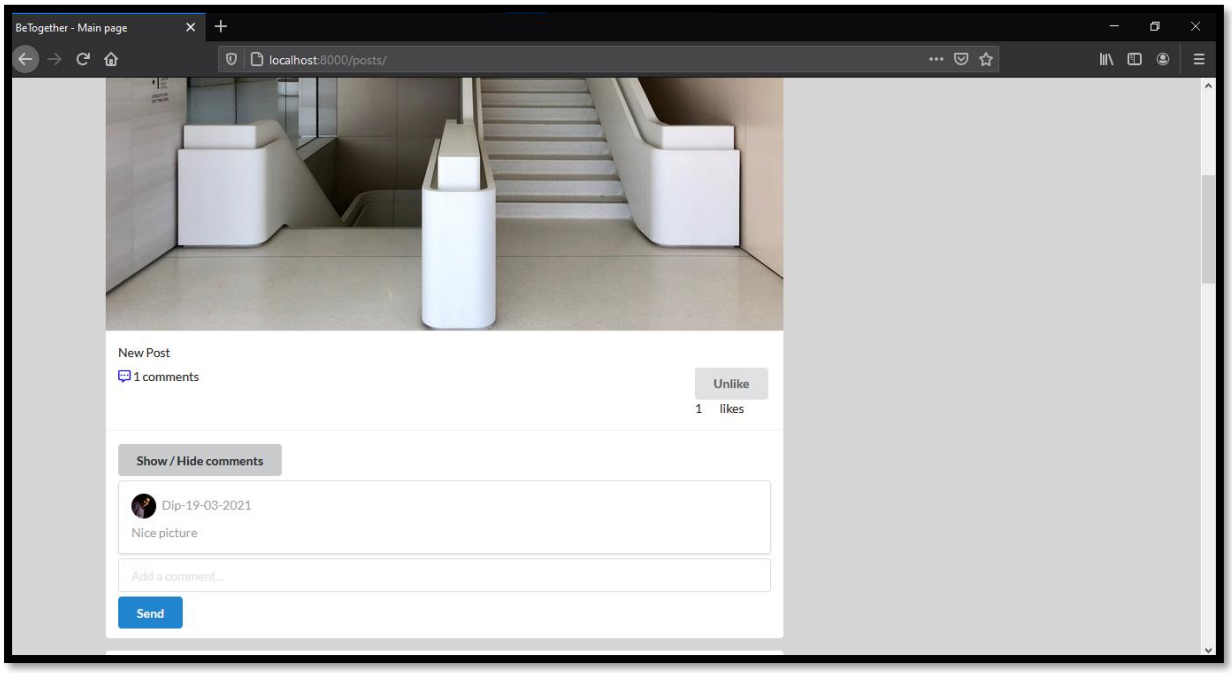
All friends list



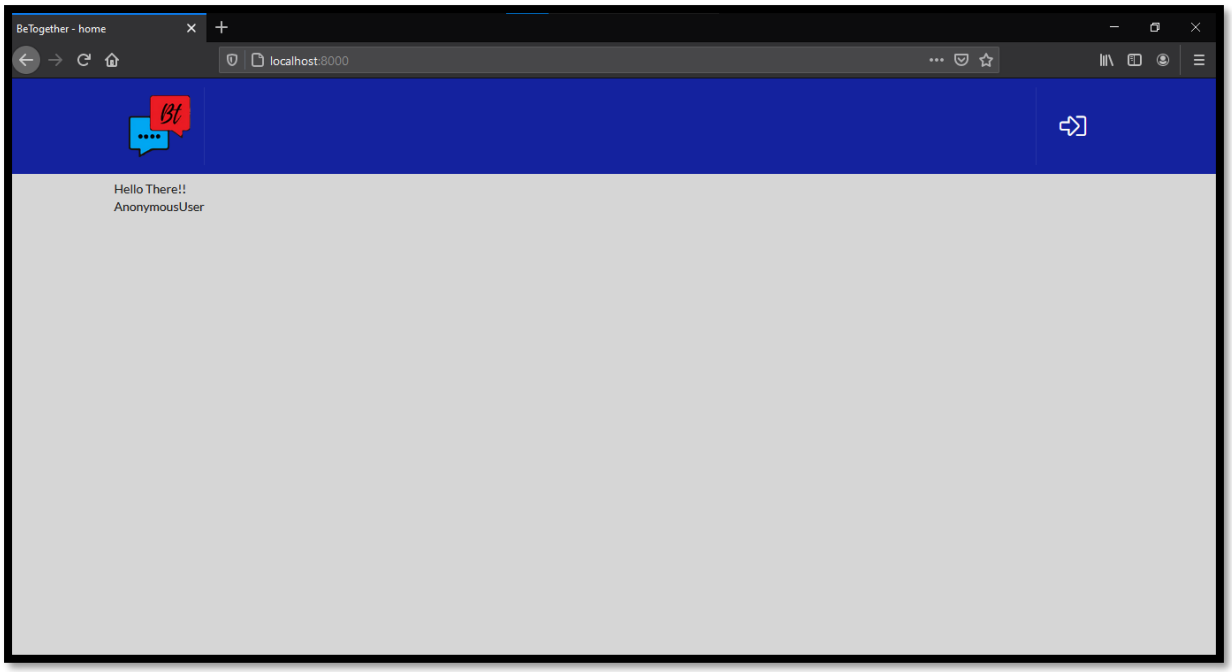
Sign out page



Friends account login



Like and comment



Logout and Back to welcome page

7. Conclusion

The functionalities are implemented in system after understanding all the system modules according to the requirements. Functionalities that are successfully implemented in the system are:

- Signup/ Registration
- Login/Sign-in
- User Validation
- Logout
- Comment

- Like
- Remove post
- Add post
- Send request to friend
- Accept request

After the implementation and coding of system, comprehensive testing was performed on the system to determine the errors and possible flaws in the system.

8. Limitations and Future Enhancements

We are able to implement some of the functionality of all modules. We aim to complete all the functionality of all modules and make this product ready to be used practically in all scenarios. Currently, the project runs completely fine if all the inputs / selections are given within proper criteria but it doesn't cover all the corner cases.

We have not created chat box where we can chat with the friends, we are also not able to implement the functionality to share the image/post to new friend.

We will extend this further more to create those functionality and try to make the project which runs anywhere (Not just in localhost).

9. Reference / Bibliography

Following links and websites were referred during the development of this project:

- getbootstrap.com
- stackoverflow.com
- docs.djangoproject.com
- w3schools.com
- github.com
- youtube.com

Project Git Repository Link:

https://github.com/JaynishMandalia/SocialMediaApp.git