## Django Social Media Project



*Framedoor Social Network*

|  |  |
| --- | --- |
| ***Project by Group 5*** | ***Supervised By*** |
| **Cristine Nordgren** **Danilo** **Mallika**  **Osama Chaban** | **Haithem** **Sowjanya Penmetsa** |

**Project documentation**

**Name of the project: Frame door**

**Features:**

* **User**

1. Mandatory Registration is needed to access the site.
2. Login (For registered users)
3. Logout (For registered users)
4. Edit Profile (for registered users)

1.FirstName

2.LastName

3.Username

4.Email

5.Bio

6.Date of Birth

7.Image

* **Admin (CRUD operations)**

1.Create post

2.Delete Post

3.Read post

4.Update post

* **User Profile**

1.Edit the profile

* **Posts**

1.All the posts

2.Add the post

3.Save the post

4.Like the post

5.Comment the post

* **Friend**

1.Send Friend Request

2.Unfollow

3.follow

4.Accept Friend Request will update both users friend list

5.See number of friends

6.Users can search for other users and send friend requests

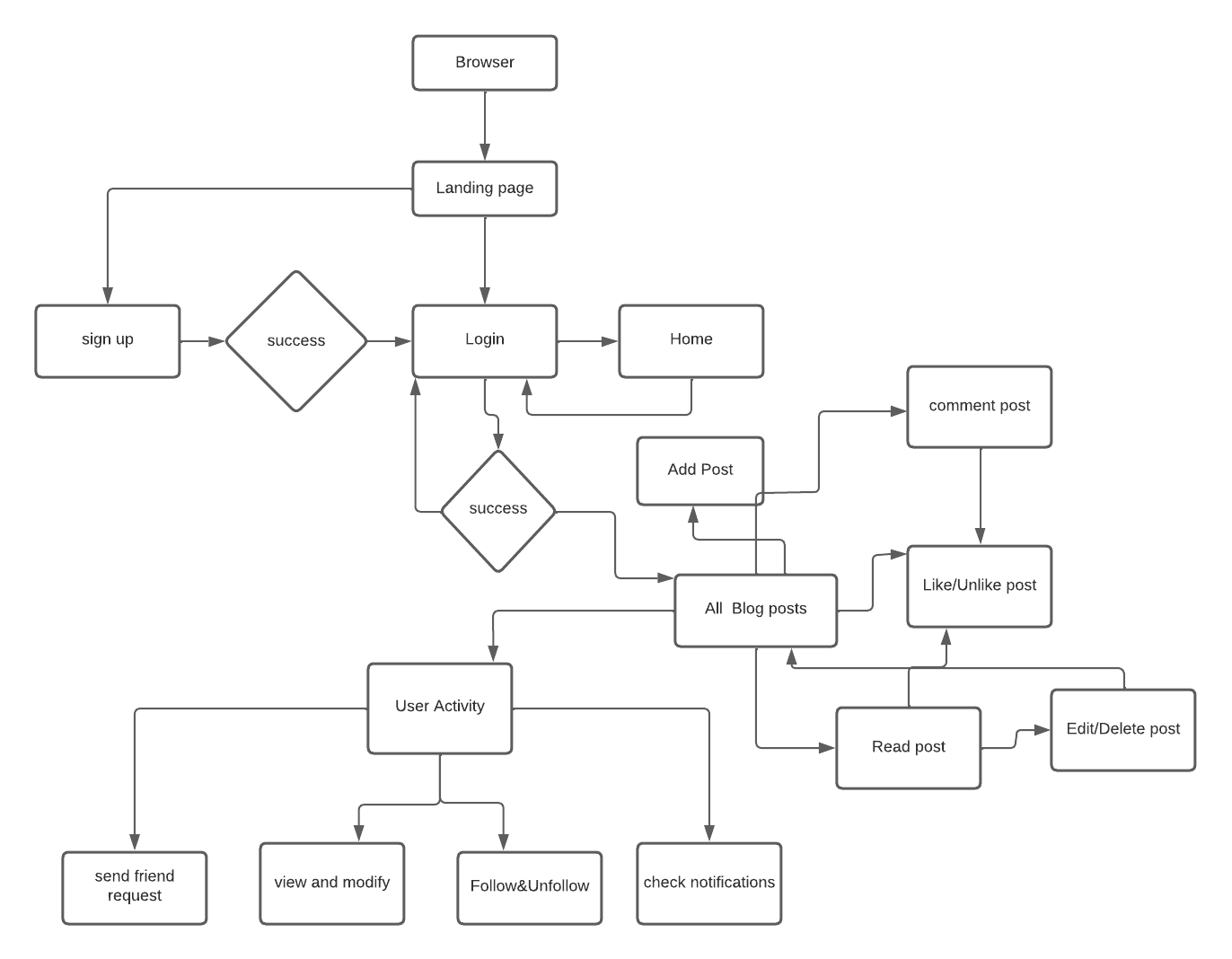
* **Notifications**

1.See all friend requests

2.see the liked posts

3.see the commented posts

* **Pages and Navigation**



* **Technologies Used**

1. Django
2. Html
3. Python=3.8
4. JavaScript
5. Bootstrap 4
6. Sqlite3

* **Apps used in the Project**

1. Social App
2. Friends
3. User

* **Django Models**

1. Profile
2. Relationship
3. Post
4. Post Comments
5. Notification
6. Friend List
7. Friend Request

* **Needed Django views and templates**
* **Views**

1. Home
2. Register
3. Login
4. First
5. Post of following Profiles
6. Like view
7. Save view
8. Like comment view
9. Post list view
10. User Post list view
11. Post detail view
12. Post Create view
13. Post Update View
14. Post Delete view
15. Search
16. All like view
17. All save view
18. Got online
19. Follow unfollow profile
20. Edit Profile
21. Profile Detail view
22. Profile
23. Public profile
24. Profile list view
25. Friend list
26. Friend request
27. Send friend request
28. Accept friend
29. Remove friend request
30. Accept friend request
31. Decline friend request
32. Show notifications

* **Templates**

1. First.html
2. Home.html
3. Index.html
4. Base.html
5. Login.html
6. Registration.html
7. Addpost.html
8. Deletepost.html
9. Editpost.html
10. Userposts.html
11. Savedposts.html
12. Comments.html
13. Feeds.html
14. Followfriend.html
15. Likesection.html
16. Likedpost.html
17. Navbar.html
18. Notifications.html
19. Postconfirmdelete.html
20. Postdetail.html
21. Postform.html
22. Savesection.html
23. Searchresults.html
24. Userposts.html
25. Allprofiles.html
26. Editprofile.html
27. Logout.html
28. Profile.html
29. Publicprofile.html
30. Userprofiles.html
31. Friendlist.html and friendrequests.html

**FrameDoor Documentation:**

Table of Contents

* General Details
* Requirements
* Configurations
* Navigation
* User
  1. Login
  2. Register
  3. View profile
  4. Edit profile
* Post
  1. Add post
  2. Save post
  3. Comment post
  4. Like post
  5. Delete post
* Friend
  1. View friend suggestion list
  2. Sending friend request
  3. View friend request list
  4. Accept friends
  5. followers
* Styling and UI components
  1. Custom CSS

**General Guidelines**

Frame door is a social networking app developed with Django 3.3.2 and Python 3.8. This web apps have smoother functionalities of adding friends to your profile from a suggested friend list. Also, it lets you maintain your social profile and sharing your thoughts in the timeline.

**Requirements**

To run the web app properly you need to follow the following requirements and have them installed in the virtual environment.

* Django =3.3.2
* Python= 3.8
* Bootstrap = 4
* SQLite

**Configurations**

As mentioned earlier the web app requires Django 3.3.2 and Python 3.8. We have used SQLite database for this application. The installed apps of the project is mentioned below.

INSTALLED\_APPS = [

    'crispy\_forms',

    'django\_cleanup',

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.sites',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'django.contrib.humanize',

    'ckeditor',

    'allauth',

    'allauth.account',

    'allauth.socialaccount',

    'allauth.socialaccount.providers.google',

    'allauth.socialaccount.providers.github',

    'social\_app.apps.SocialAppConfig',

    'users.apps.UsersConfig',

    'notification',

    'friend',

 ]

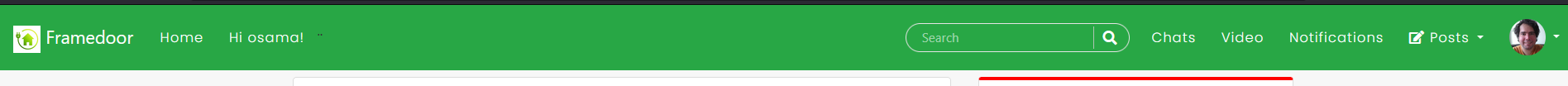
The media root of the package is set as follows:

MEDIA\_ROOT = MEDIA\_DIR

MEDIA\_URL = '/media/'

CRISPY\_TEMPLATE\_PACK = 'bootstrap4'

* **Navigation:**



At the very beginning the app will direct you to home page, where You can login to the app using your username and password. But it requires you to create an account in the apps first. So, you can access to the signup page from there to create an account. Then you can access to your profile after login. There you can also edit your profile. You can also add post and save them. You can then see the other users post and like them and comment. You can also comment on their posts and send them friend requests.

The URL pattern that is used in the social app looks as follows,

app\_name = "social\_app"

urlpatterns = [

    path('', views.first, name='firsthome'),

    path('home/', PostListView.as\_view(), name='blog-home'),

    path('feed/', posts\_of\_following\_profiles, name='posts-follow-view'),

    path('post/user/<str:username>/', UserPostListView.as\_view(), name='user-posts'),

    path('post/<int:pk>/', PostDetailView, name='post-detail'),

    path('post/<int:pk>/update/', PostUpdateView.as\_view(), name='post-update'),

    path('post/<int:pk>/delete/', PostDeleteView.as\_view(), name='post-delete'),

    path('post/new/', PostCreateView.as\_view(), name='post-create'),

    path('post/like/', LikeView, name='post-like'),

    path('liked-posts/', AllLikeView, name='all-like'),

    path('post/save/', SaveView, name='post-save'),

    path('saved-posts/', AllSaveView, name='all-save'),

    path('post/comment/like/', LikeCommentView, name='comment-like'),

    path('about/', views.about, name='blog-about'),

    path('search/', views.search, name='search'),

]

The URL is the users app looks as follows,

app\_name = 'users'

urlpatterns = [

    path('register/', user\_views.register, name='register'),

    path('profile/', user\_views.profile, name='profile'),

    path('login/', user\_views.user\_login, name='login'),

    path('logout/',user\_views.user\_logout, name='logout'),

    path('password-reset/', auth\_views.PasswordResetView.as\_view(template\_name='users/password\_reset.html'), name='password\_reset'),

    path('password-reset/done/', auth\_views.PasswordResetDoneView.as\_view(template\_name='users/password\_reset\_done.html'), name='password\_reset\_done'),

    path('password-reset-confirm/<uidb64>/<token>/', auth\_views.PasswordResetConfirmView.as\_view(template\_name='users/password\_reset\_confirm.html'), name='password\_reset\_confirm'),

    path('password-reset-complete/', auth\_views.PasswordResetCompleteView.as\_view(template\_name='users/password\_reset\_complete.html'), name='password\_reset\_complete'),

    path('all/', views.ProfileListView.as\_view(), name='profile-list-view'),

    path('follow/', views.follow\_unfollow\_profile, name='follow-unfollow-view'),

    path('<int:pk>/', views.ProfileDetailView.as\_view(), name='profile-detail-view'),

    path('public-profile/<str:username>/', views.public\_profile, name='public-profile'),

]

The URL pattern that is used in the friend looks as follows,

app\_name = "friend"

urlpatterns = [

    path('friend\_list/<int:pk>', friends\_list\_view, name='friend-list'),

    path('friend\_request/', send\_friend\_request, name="friend-request"),

    path('friend\_requests/<user\_id>/', friend\_requests, name='friend-requests'),

    path('friend\_request\_accept/<friend\_request\_id>/', accept\_friend\_request, name='friend-request-accept'),

    path('friend\_remove/', remove\_friend, name='remove-friend'),

    path('friend\_request\_decline/<friend\_request\_id>/', decline\_friend\_request, name='friend-request-decline'),

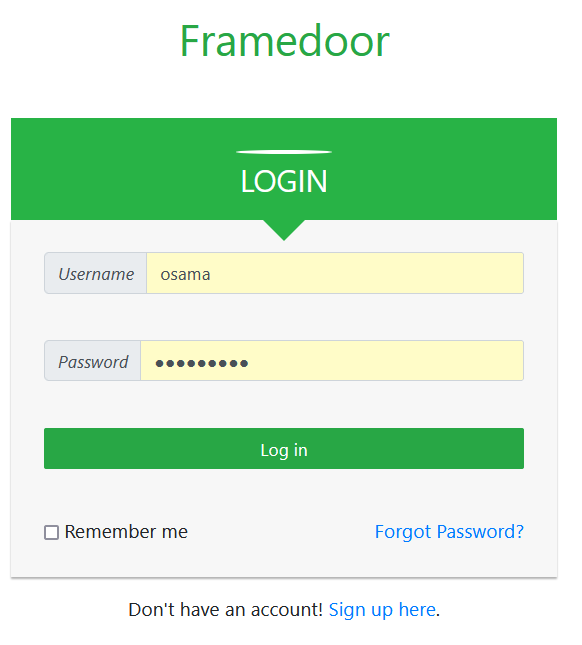
    path('friend\_request\_cancel/', cancel\_friend\_request, name='friend-request-cancel'),

]

**User**

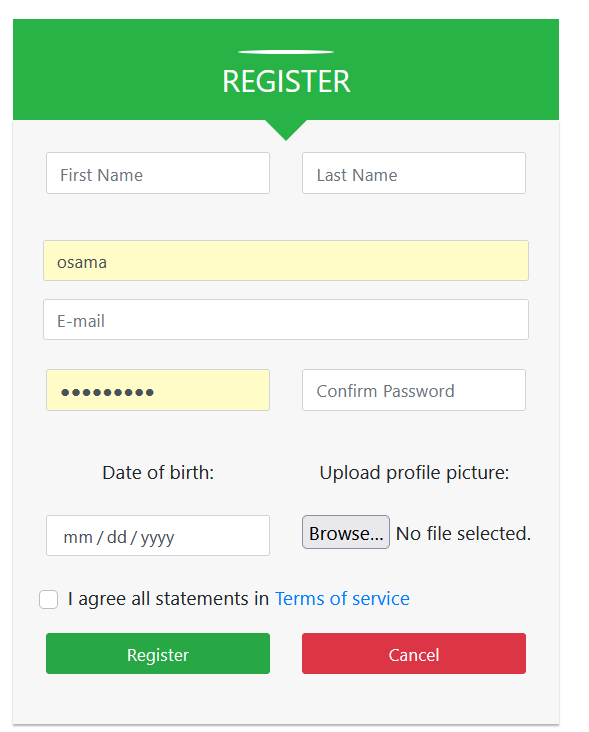
The user app has all the functionalities like login, signup, viewing profile and editing profileand user profile details The proper usage of the app comes with the initial usage of the app. The very beginning the app routes user to login in the app. The app used Django’s in built auth form in order to ensure the security of the user.

* **Login:**

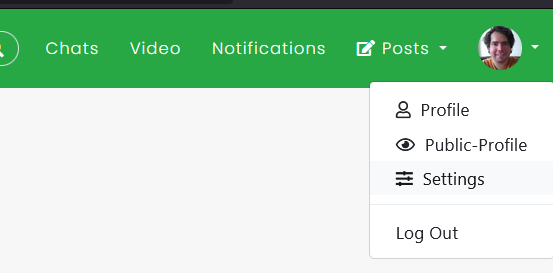


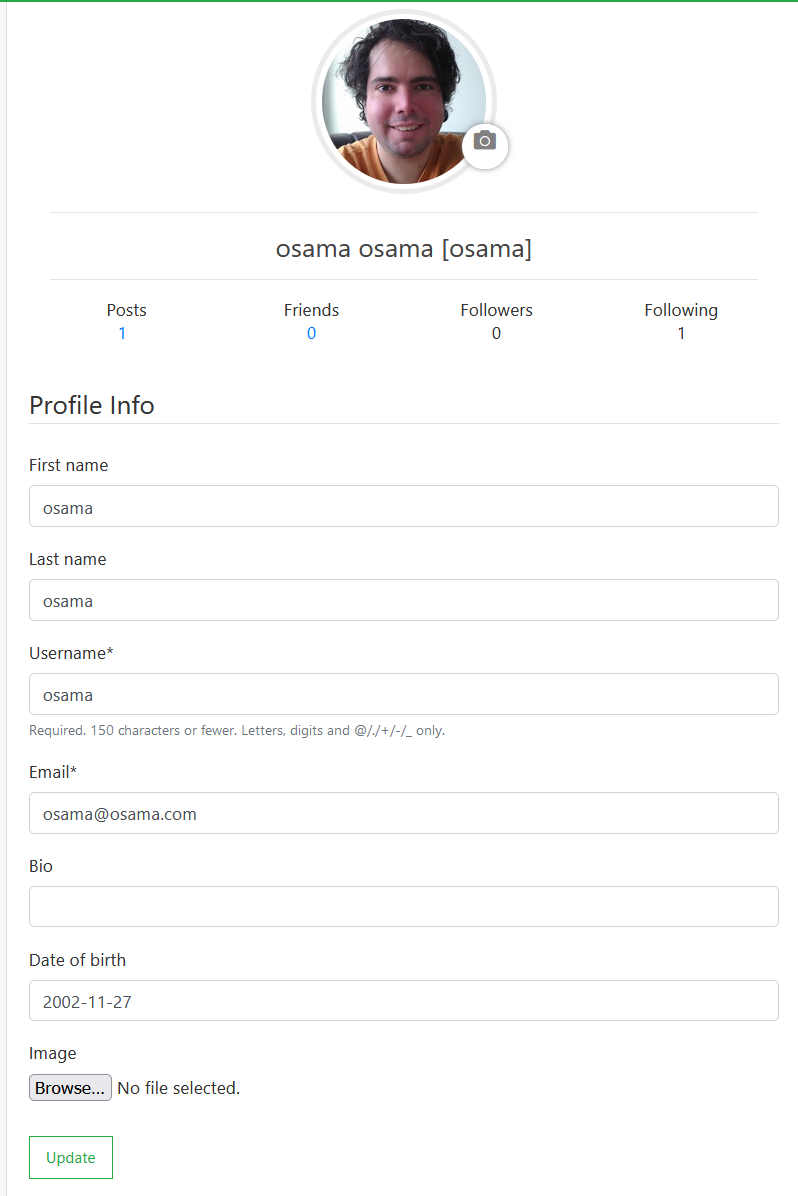
In the login page user have to give the user name and the password of the user in order to access to the web app. If the user does not have any account in the web app, they can access the sign-up page from there using the register button.

* **Register:**



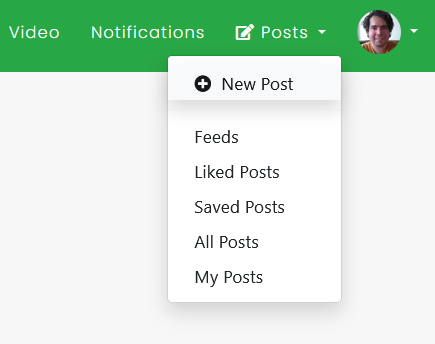
* + In the sign-up page the user has to register himself to the web app. For that the user have to give some information in the very beginning. The user must have to provide the following info.
    - First name
    - last name
    - Username
    - Password
    - Confirm password
    - Date of birth
    - Profile pic
* **Edit profile:**

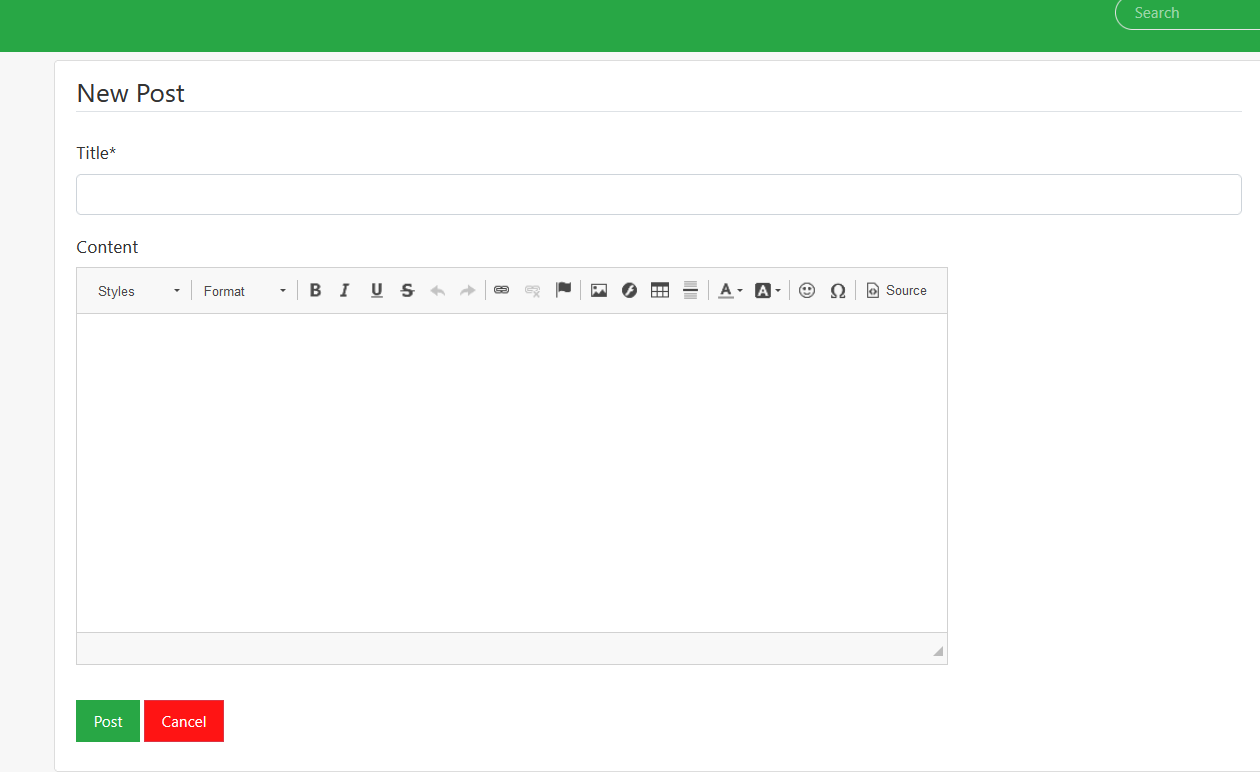




In the navigation bar, we have an option to edit profile, the user can update the information they have provided before. They can also edit their profile picture.

* **Posts**



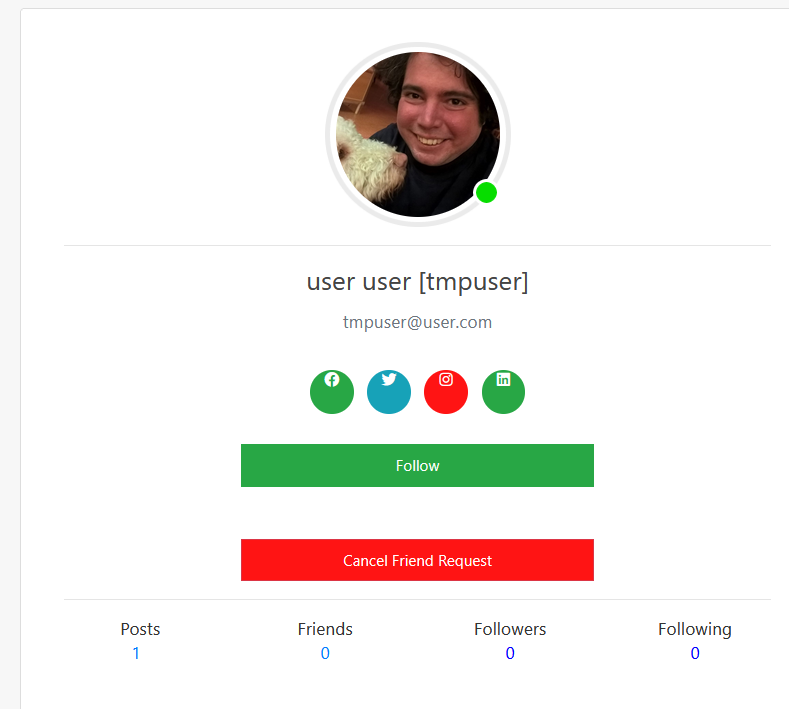
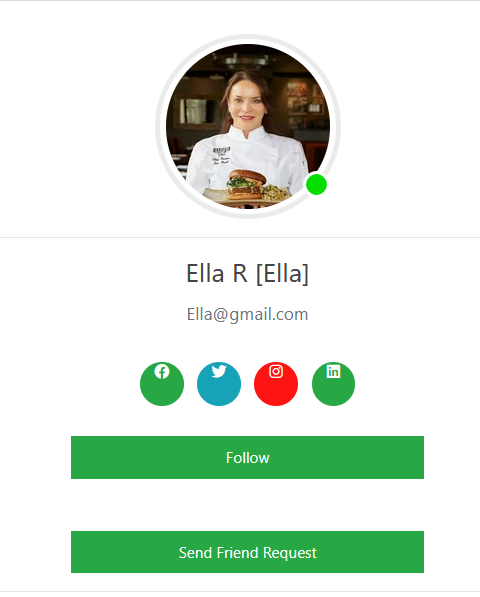


In this after logging into the web app, the user can see all the posts of the other users. Here, the user can add post and save it. He can also like the other users posts and comment them.

**Friend**

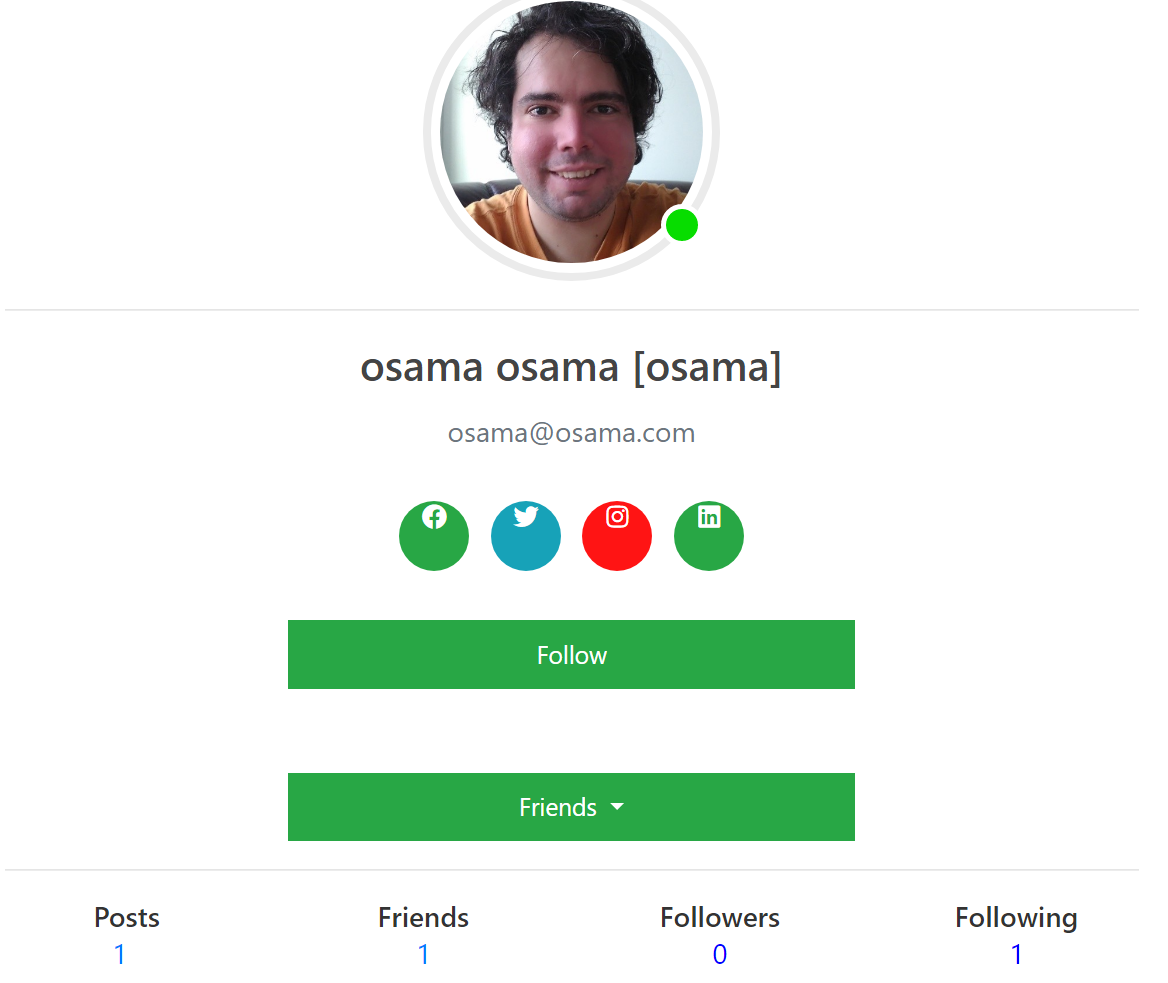
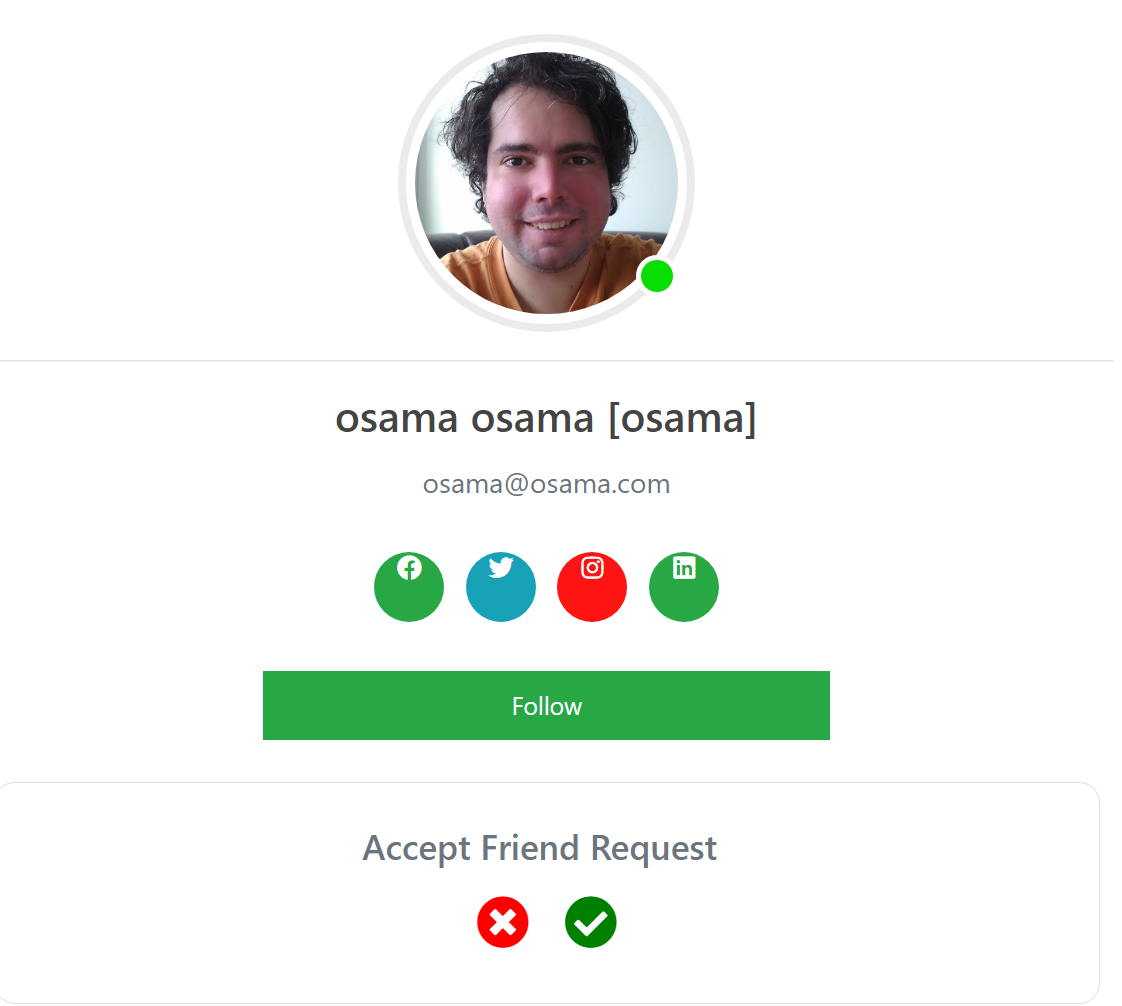
The friend feature is done with our “friends” app. The friend app has the functionalities like vi sending friend requests, viewing the friend request , accepting or rejecting friends and also viewing the user’s friend list. After a user has successfully logged in to the system, he or she will be able to make friends with the help of our friends app.

* Sending friend request



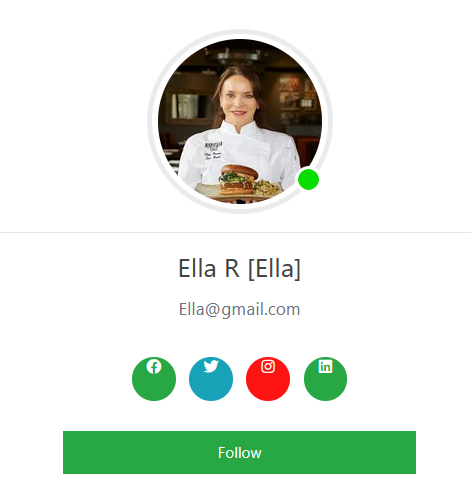
For sending a friend request a user needs to click the user profile and there he can click on the Add friend button and send the request to the desired user. You can also cancel the friend request if you want.

* Accept friends



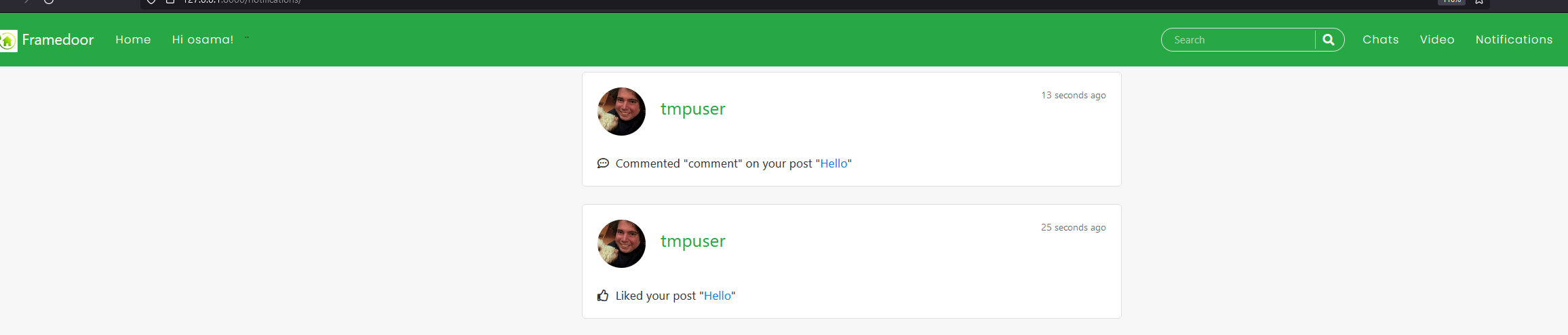
After a user finds his or incoming friend requests, he or she will be able to accept the friend requests. If the friend request is accepted, both the users friend lists get updated automatically.

* Following/Followers



Instead of freind request you can choose to follow users and give their posts a priority in you feed and keep up to date with their posts

* **Notifications:**



After the user accepts the friend request, he can click on the notifications and see whether they are friends or not. We can also see the status of the person in online. He can also see all the saved posts and liked posts. He has also an opportunity to look into only his posts.

**Styling and UI components**

For styling the web app we have used standard CSS that can be accessed in the basic style.CSS file. We have used the Bootstrap 4 UI components like button, navigation bar, form components.