Real Estate Listing Portal

Team:

Srishti Patel
sp4917@nyu.edu

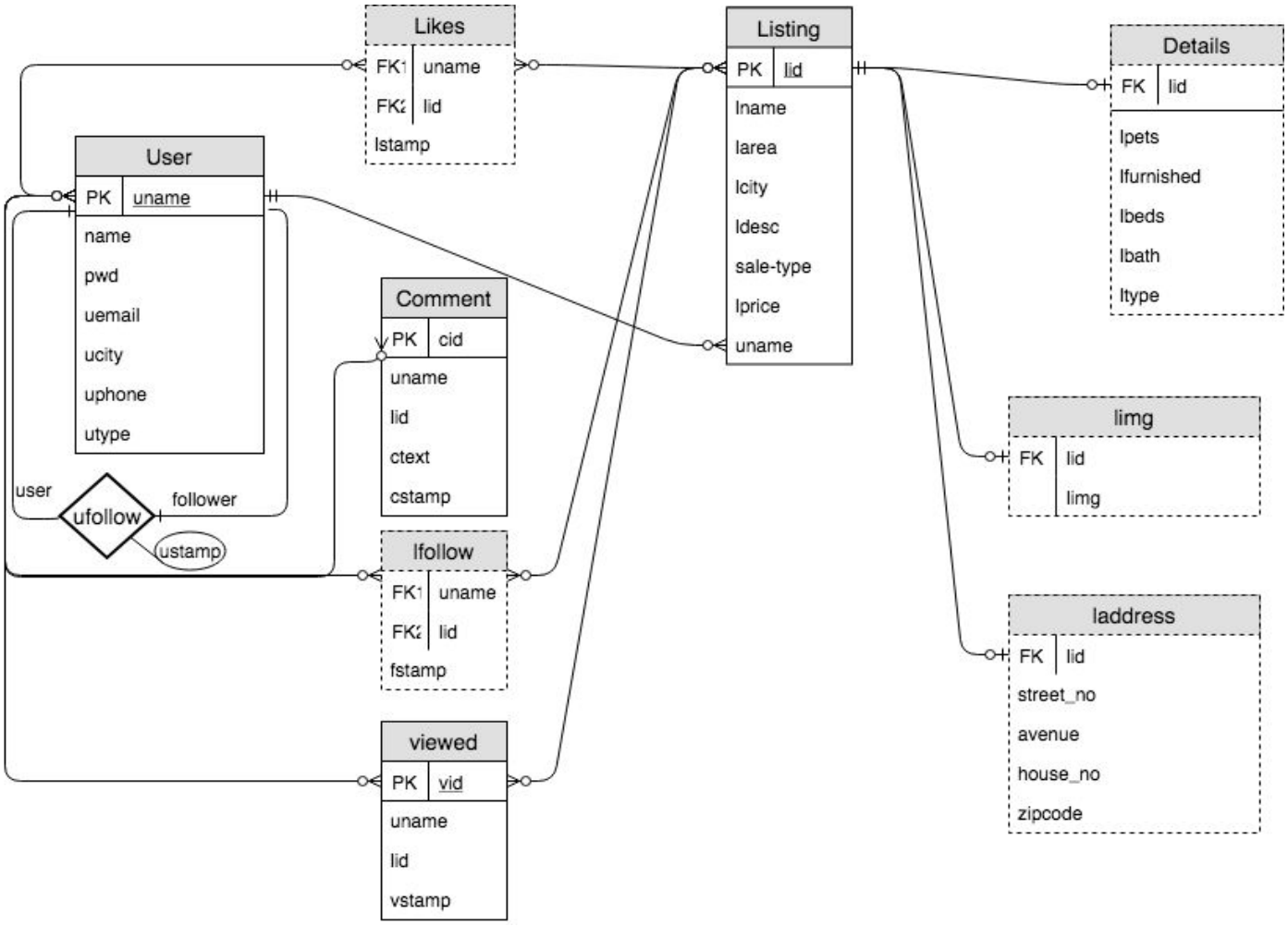
Yashashwini Gupta
yg1568@nyu.edu

INTRODUCTION

We have designed a system that can be used in the real estate market to sell,rent or buy houses.
Our system enables the users to post the housing details they want to sell or rent out along with the necessary details about the listing. The location of the listing uploaded, will be shown in maps.
In addition, there are various functionalities that are available to the user. A user can follow some other user. This would give the user, the privilige to get updates from the user he/she is following his/her own dashboard.
If a customer follows an agent, he/she will be updated about the agent's activity. If a customer follows a listings he/she will be updated about any changes to the listings. The users can also like or comments on the various listings. The listings had a detailed description of the various facilities in the apartment also multimedia for the user to see.

ER DIAGRAM

The ER-Diagram for the Real Estate Social Network:



The architecture of the system consists of 4 weak entities and 4 strong entities:

1) User- Strong Entity(uname,name,pwd,uemail,ucity,uphone,utype)

This contains the details of the User. The details stored are username (uname - pkey), password (pwd), name (name), email (uemail), city (ucity), phone number(uphone), type of user such as broker, agent, seller, buyer(utype). The password field in the test data contains random data but will be encrypted in the 2nd iteration. The possible functionalities from this table will be: create new user, update info, update password etc.

2) Listing- Strong Entity (lid,lname,larea,lcity,ldesc,sale-type,lprice,uname)

This contains the details of the listing. The details stored are identification number (lid - pkey), name of listing (lname) , area of location of listing(larea), city of listing(lcity), description about the listing(ldesc), type of listing can be sell or rent (sale-type), cost of sale (lprice), user who posted the listing(uname). The possible functionalities from this table will be: create new listing, update listing etc.

3) Comment- Strong Entity (cid,lid,uname,ctext, cstamp)

This contains the details of the comments posted by the users. The details stored are identification number (cid - pkey), username of the user commenting (uname) , listing id he commented on(lid), timestamp of comment(cstamp), ctext(text of the comment). The possible functionalities from this table will be: create new comment, update comment etc.

4) Viewed- Strong Entity (vid,lid,uname, vstamp)

This contains the details of the listings viewed by the users. The details stored are identification number (vid - pkey), username of the user viewing(uname) , listing id he viewed(lid), timestamp of view(vstamp). The possible functionalities from this table will be: looking at the recent views.

4) lfollow- Weak Entity (lid,uname, fstamp)

This contains the details of the listings followed by the users. The details stored are username of the user viewing(uname) , listing id he followed(lid), timestamp of follow(fstamp). The possible functionalities from this table will be: creating new follows, unfollowing etc.

5) likes- Weak Entity (lid,uname, lstamp)

This contains the details of the listings liked by the users. The details stored are username of the user viewing(uname) , listing id he liked(lid), timestamp of like(fstamp). The possible functionalities from this table will be: creating new likes, unliking etc.

6) ldetails- Weak Entity (lid,lpets,lfurnished,lbeds,lbath,ltype)

This contains the additional details of the listing. The details stored are identification number (lid - fkey), name of listing (lname), information about pets(lpets), information about furnishing(lfurnished), number of bedrooms(lbeds), number of baths(lbath), type of listings eg condo, villa, apartment (ltype). The possible functionalities from this table will be: add new details, update details etc.

7) Laddress- Weak Entity (lid,street_no,house_no,avenue,zipcode)

This contains the address of the listing. The details stored are identification number (lid - pkey), street number(street_no), house number(house_no), closest avenue(avenue), zipcode of house location(zipcode).The possible functionalities from this table will be: adding address, update address etc.

8) limg- Weak Entity (lid, limg)

This contains the images of the listings added by the user. The details stored are lid and limg. The possible functionalities from this table will be: adding images, update images etc.

RELATIONAL SCHEMA OF DATABASE

Schema:
User(uname,name,pwd,uemail,ucity,uphone,utype)
Listing(lid,lname,larea,lcity,ldesc,sale-type,lprice,uname)
Comment(cid,lid,uname,ctext, cstamp)
Viewed(vid,lid,uname, vstamp)
likes(lid,uname, lstamp)
ldetails(lid,lpets,lfurnished,lbeds,lbath,ltype)
laddress(lid,street_no,house_no,avenue,zipcode)
limg(lid, limg)
ufollow(uname1,uname2,ustamp)

Foreign key relationships:
listing(uname) references User(uname)
ufollow(uname1) references User(uname)
ufollow(uname2) references User(uname)
Comment(uname) references User(uname)
Comment(lid) references listing(lid)
Viewed(uname) references User(uname)
Viewed(lid) references listing(lid)
Likes(uname) references User(uname)
likes(lid) references listing(lid)
ldetails(lid) references listing(lid)
laddress(lid) references listing(lid)
limg(lid) references listing(lid)

DDL Statements:

```
create database real_estate;
Use real_estate;
drop table if exists user;
Create table user(
uname  varchar(100) primary key, pwd varchar(100), name varchar(100),  uemail  varchar(100) , ucity
varchar(100), uphone varchar(100), utype varchar(100)
);

Drop table if exists listings;
Create table listings(
Lid varchar(100) primary key, uname  varchar(100),lname varchar(100), larea varchar(100), lcity varchar(100),
ldesc varchar(100), sale_type varchar(100), lprice varchar(100) ,
Foreign key (uname) references user(uname)
```

```
On delete cascade
);

Drop table if exists ldeatils;
Create table ldetails(
lid  varchar(100) , lpets varchar(100), lfurnished varchar(100),  lbeds  varchar(100) , lbath varchar(100), ltype
varchar(100),
Foreign key (lid) references listings(lid),
Primary key(lid)
);

Drop table if exists laddress;
Create table laddress(
lid  varchar(100), street_no  varchar(100) , avenue varchar(100), zipcode int(10),  house_no  varchar(100) ,
Foreign key (lid) references listings(lid),
Primary key(lid)
);

Drop table if exists ufollow;
Create table ufollow(
uname1  varchar(100) , uname2 varchar(100), ustamp datetime,
Foreign key (uname1) references user(uname),
Foreign key (uname2) references user(uname)
);

Drop table if exists lfollow;
Create table lfollow(
uname  varchar(100) , lid varchar(100), fstamp datetime,
Foreign key (lid) references listings(lid),
Foreign key (uname) references user(uname)
);

Drop table if exists likes;
Create table likes(
uname  varchar(100) , lid varchar(100), lstamp datetime,
Foreign key (lid) references listings(lid),
Foreign key (uname) references user(uname)
Primary key(lid, uname)
);

Drop table if exists comment;
Create table comment(
Cid varchar(100) primary key, uname  varchar(100) , lid varchar(100), cstamp timestamp,
```

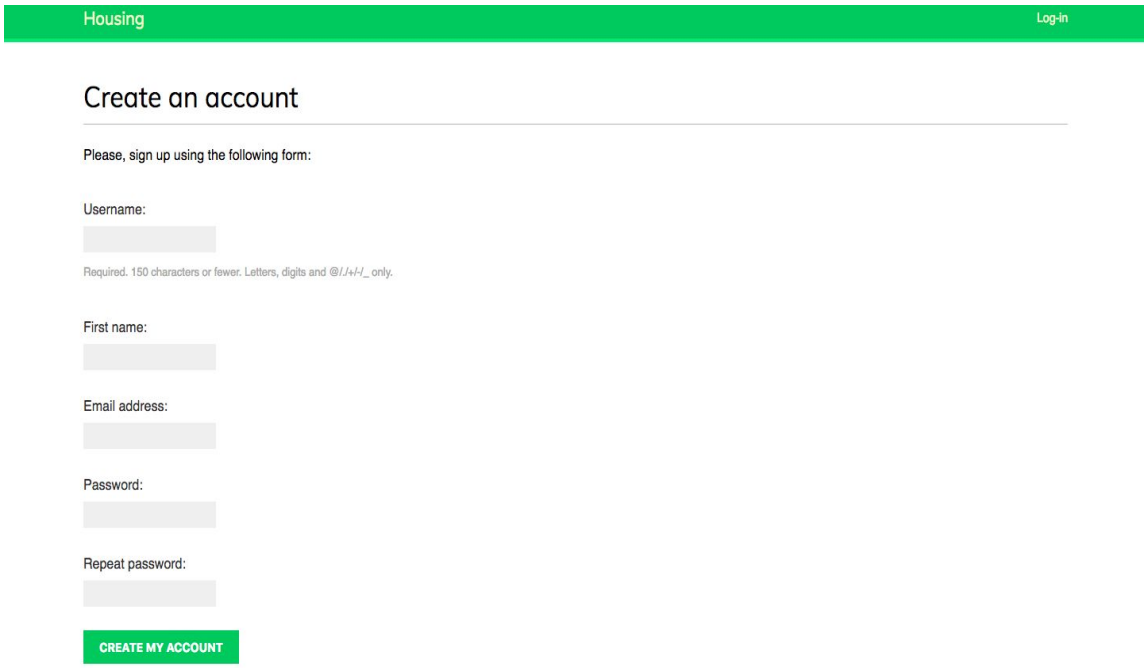
```
Foreign key (lid) references listings(lid),
Foreign key (uname) references user(uname)
);

Drop table if exists viewed;
Create table viewed(
uname  varchar(100) , lid varchar(100), rvstamp timestamp,
Foreign key (lid) references listings(lid),
Foreign key (uname) references user(uname)
);

Drop table if exists limg;
Create table limg(
lid varchar(100), img blob,
Foreign key (lid) references listings(lid),
);
```

USE CASES-

1) User Registration:

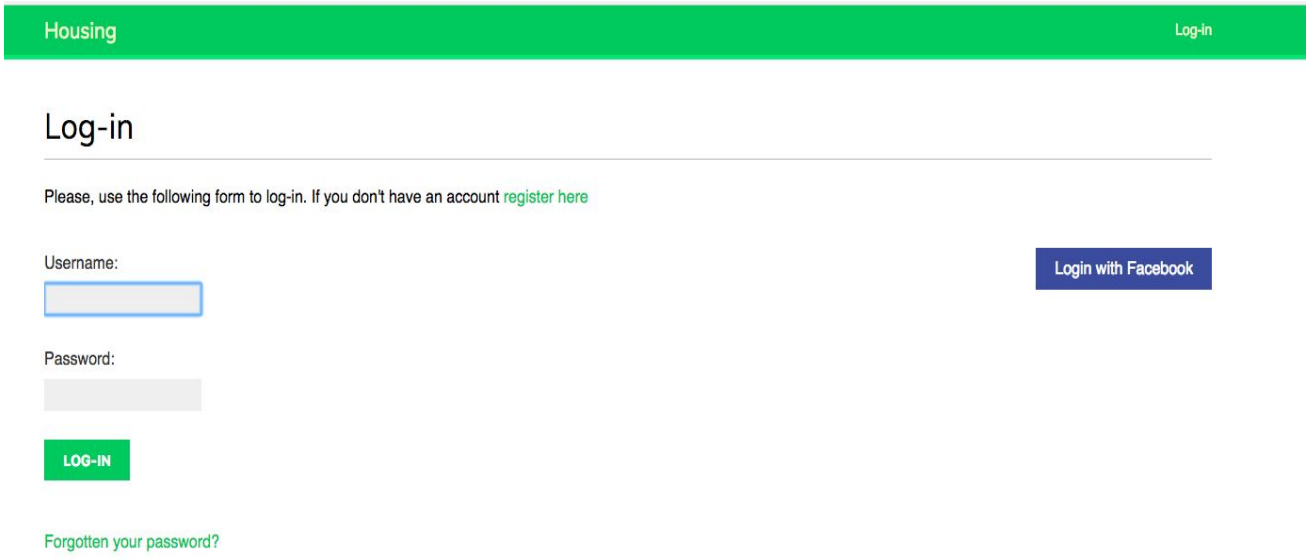
SignUp page:

- Go to project.com:8000/account/login.
- Select Register New Account.
- Add the details as shown in the form



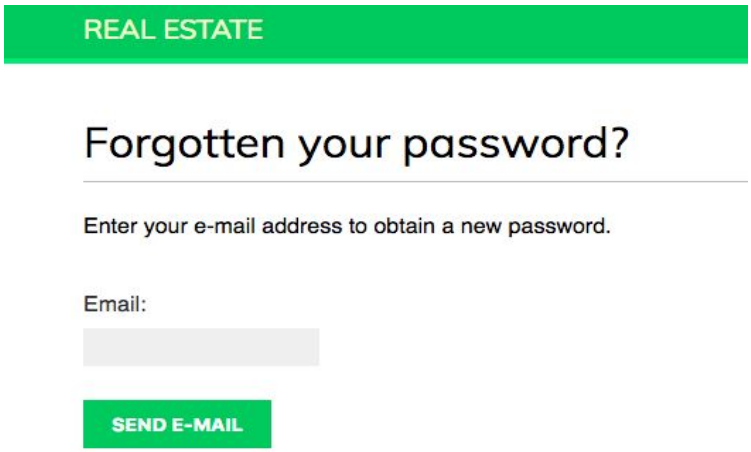
2)User Login Page:

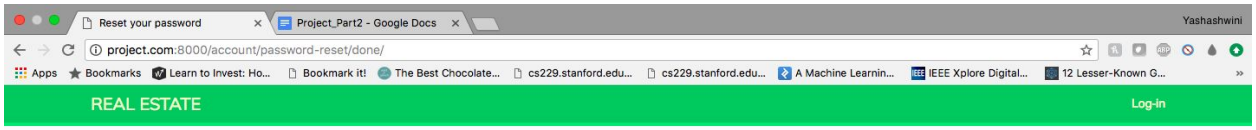
- Enter user-id
- Enter Password
- User can also login from facebook (However, because of facebook's updated API policy since March 2018, only https sites are enabled to use facebook authentication for logging in)
- You can choose forgot password if you do not remember the password



Forgotten Password:

If a user forgets the login password to access his or her account, it can be changed by following the forgotten my password link and then entering a valid email address.



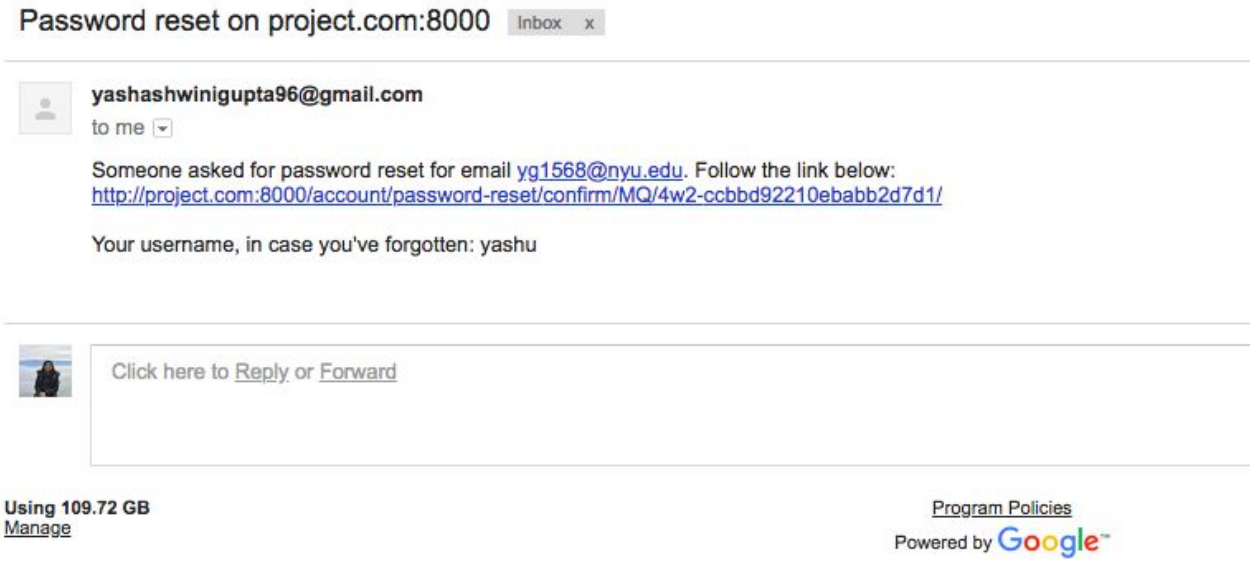


Reset your password

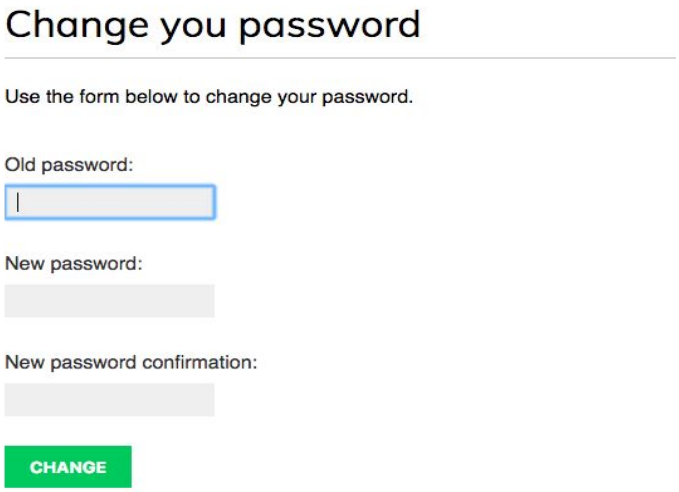
We've emailed you instructions for setting your password.

If you don't receive an email, please make sure you've entered the address you registered with.

You will then receive the following email to reset your password.

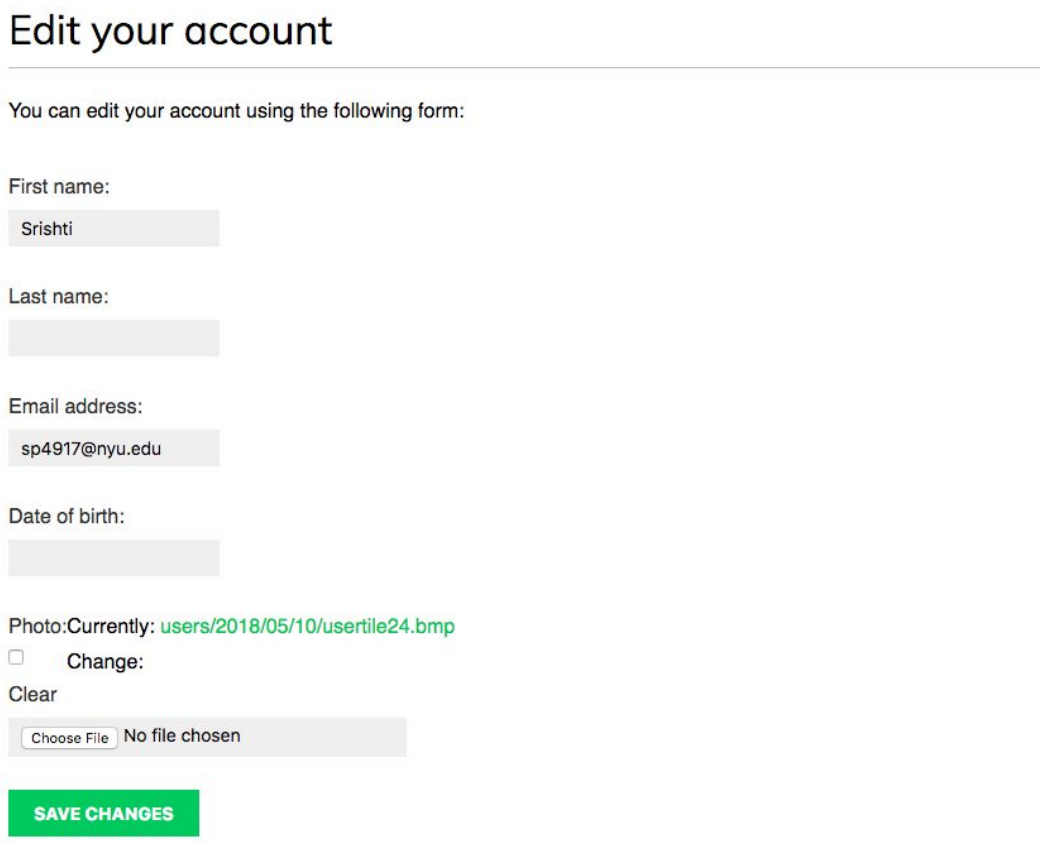


Change Password:



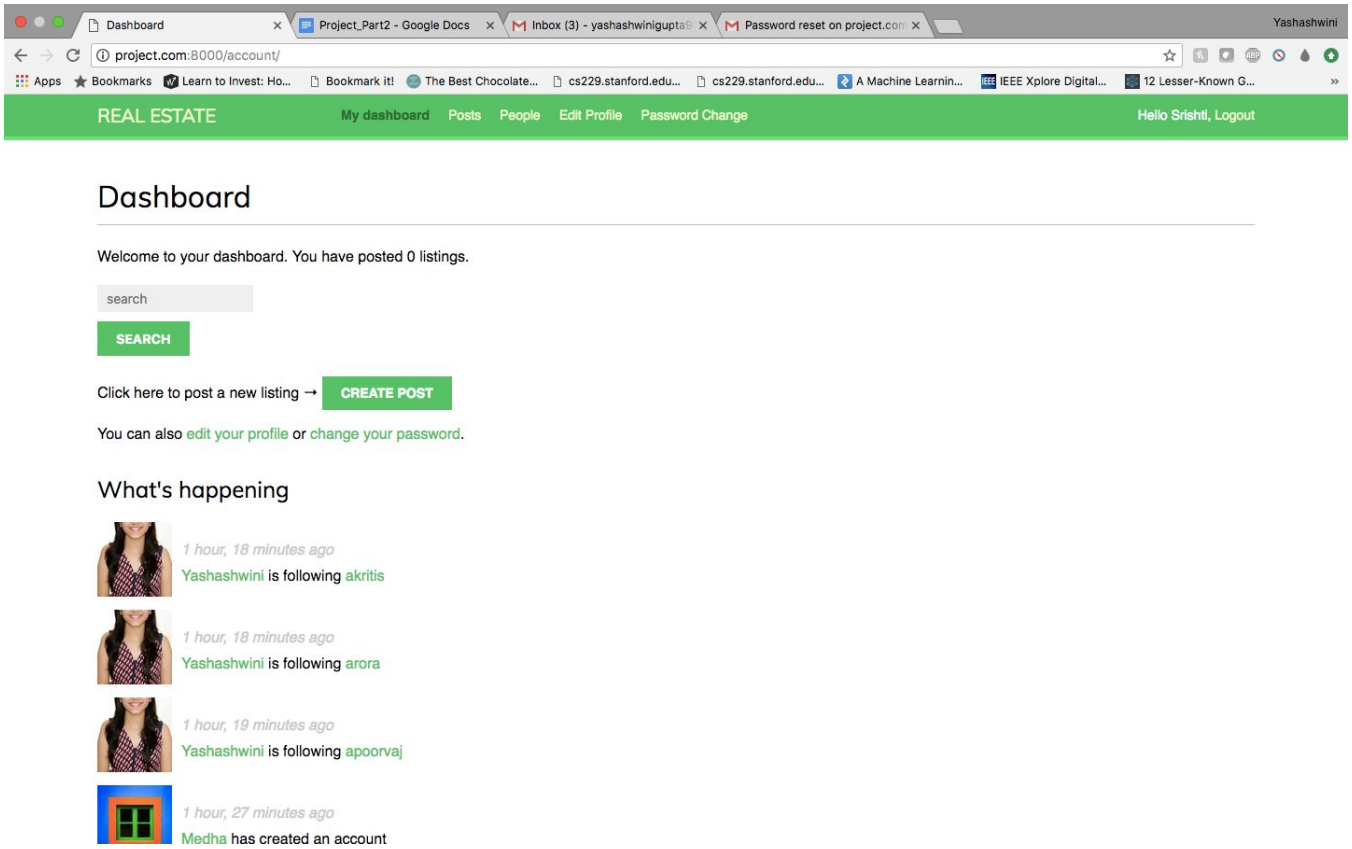
3)Edit Profile:

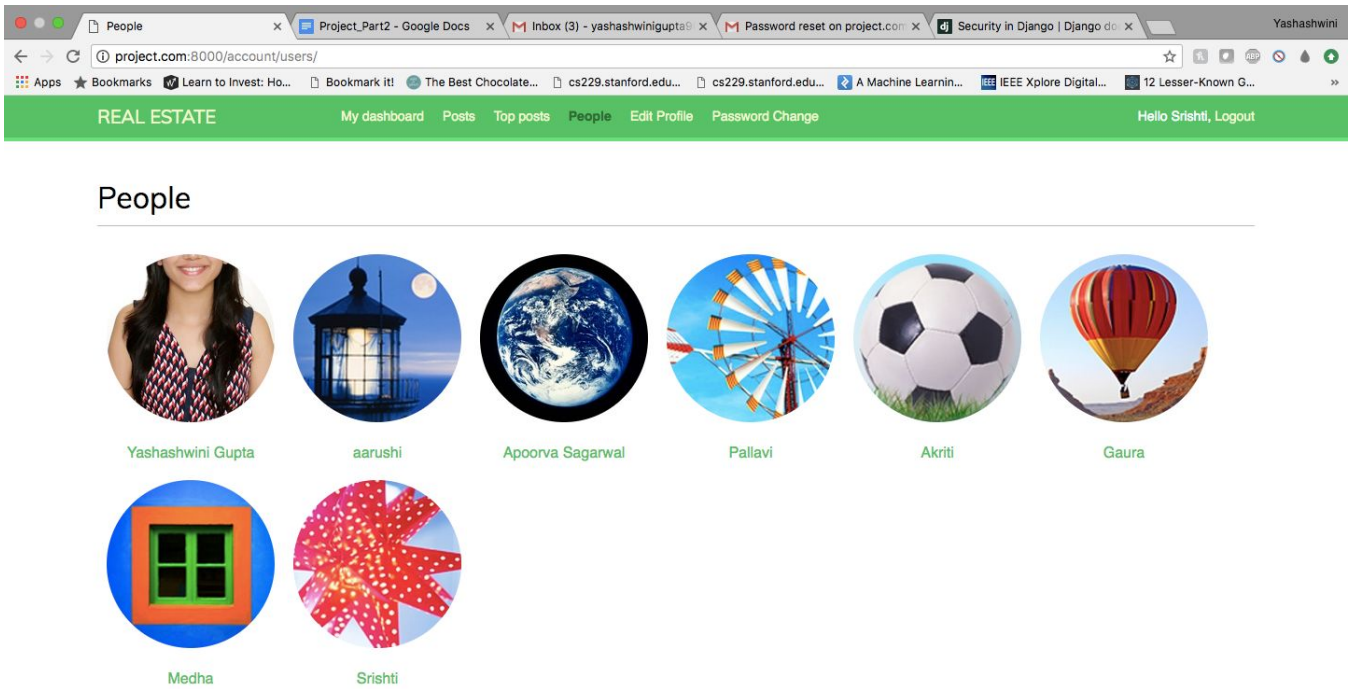
The user can edit his/her personal details by following the Edit profile link on the header.
The link will open to the following form which will update the database.

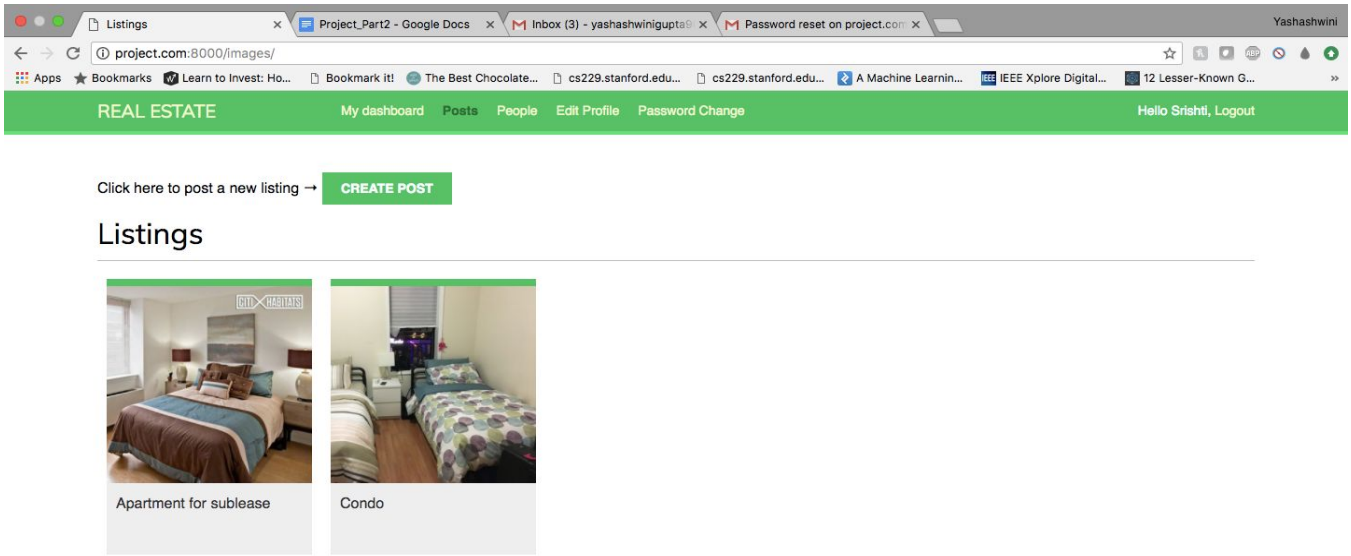


4)Activity Post:

The user can see the activity, along with the timestamp, from other user he/she is following.
One can also browse through the posts or a user by clicking on their name.
By following the Posts link on the header, we can view a list of all the listings posted on the website.
You can also see the number of listings posted by you.

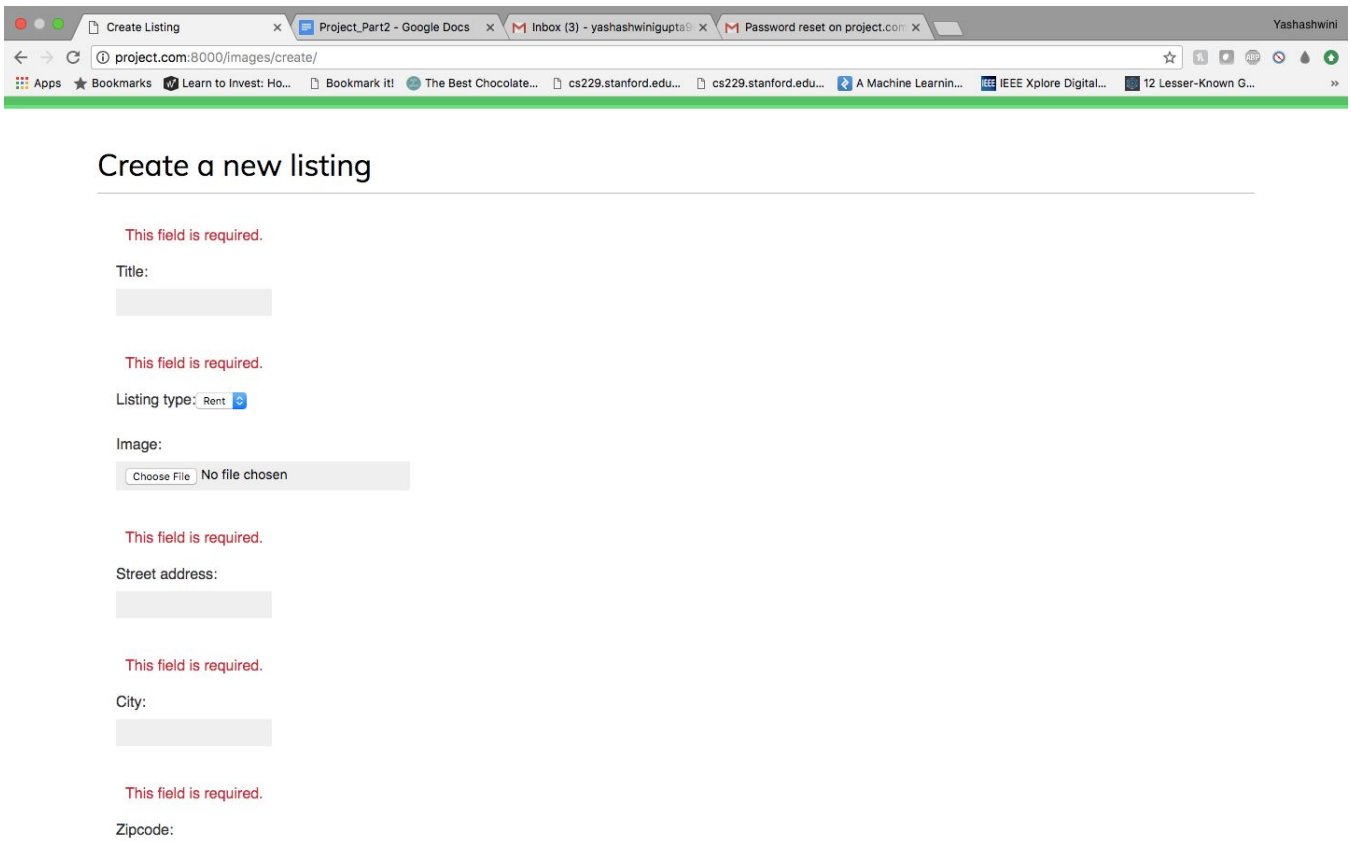

Dashboard
Project_Part2 - Google Docs
Inbox (3) - yashashwinigupta9
Password reset on project.com
Yashashwini
project.com:8000/account/
Apps
Bookmarks
Learn to Invest: Ho...
Bookmark it!
The Best Chocolate...
cs229.stanford.edu...
cs229.stanford.edu...
A Machine Learnin...
IEEE Xplore Digital...
12 Lesser-Known G...
REAL ESTATE
My dashboard
Posts
People
Edit Profile
Password Change
Hello Srishti, Logout
Dashboard
Welcome to your dashboard. You have posted 0 listings.
search
SEARCH
Click here to post a new listing →
CREATE POST
You can also edit your profile or change your password.
What's happening
1 hour, 18 minutes ago
Yashashwini is following akritis
1 hour, 18 minutes ago
Yashashwini is following arora
1 hour, 19 minutes ago
Yashashwini is following apoorvaj
1 hour, 27 minutes ago
Medha has created an account

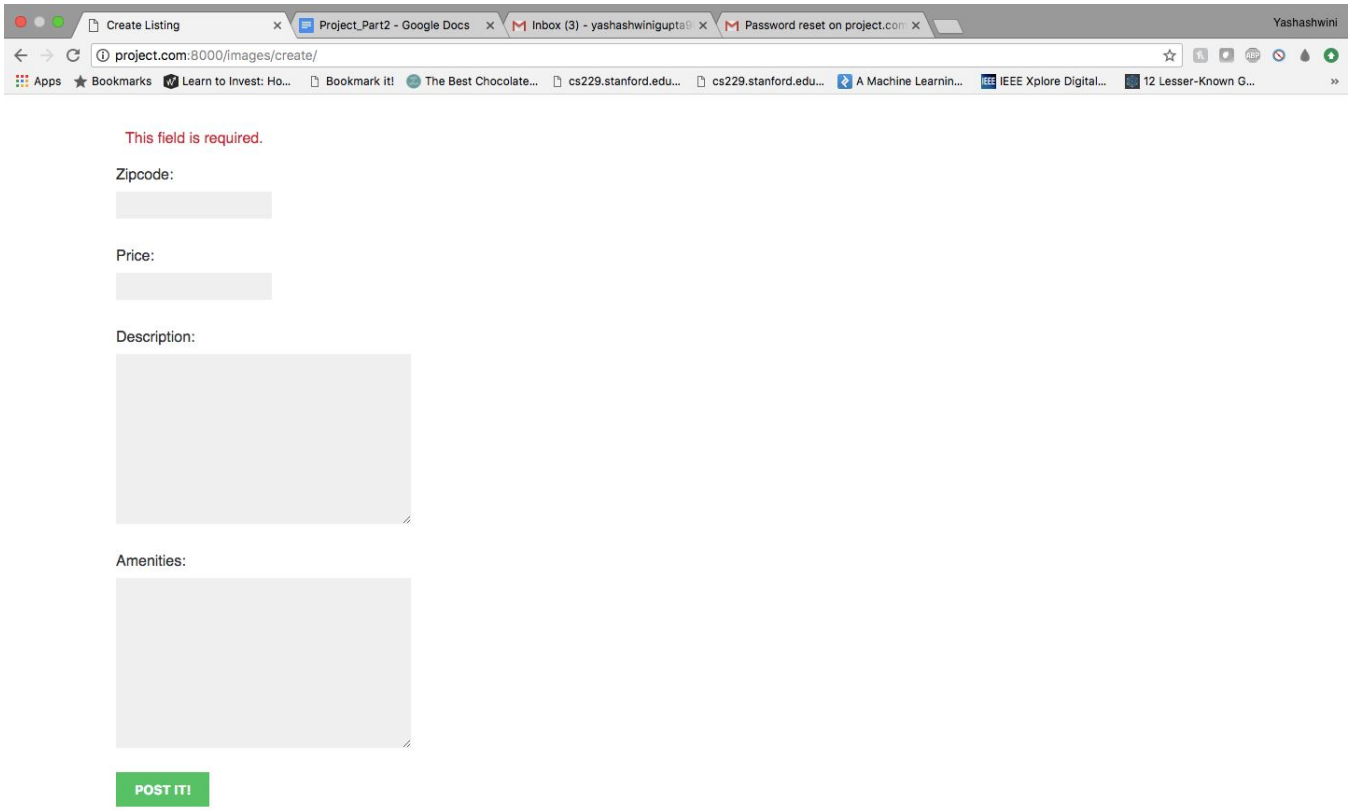

People
Project_Part2 - Google Docs
Inbox (3) - yashashwinigupta9
Password reset on project.com
Security in Django | Django do
Yashashwini
project.com:8000/account/users/
Apps
Bookmarks
Learn to Invest: Ho...
Bookmark it!
The Best Chocolate...
cs229.stanford.edu...
cs229.stanford.edu...
A Machine Learnin...
IEEE Xplore Digital...
12 Lesser-Known G...
REAL ESTATE
My dashboard
Posts
Top posts
People
Edit Profile
Password Change
Hello Srishti, Logout
People
Yashashwini Gupta
aarushi
Apoorva Sagarwal
Pallavi
Akriti
Gaura
Medha
Srishti



5)Diary Entry:

Apart from following other users, a user can also upload his/her own listing that he wants to sell or rent out. This can be done by clicking on the Create post button on the home page and then filling the form.

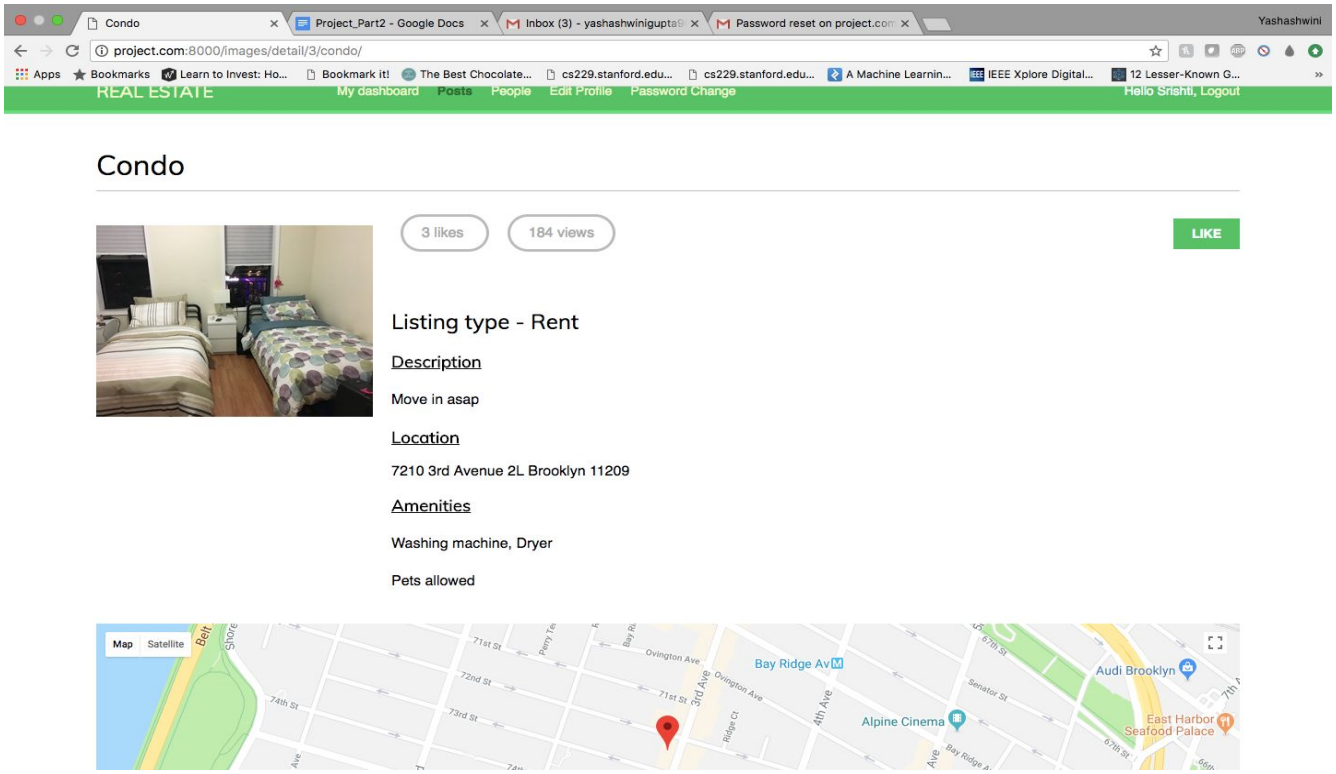


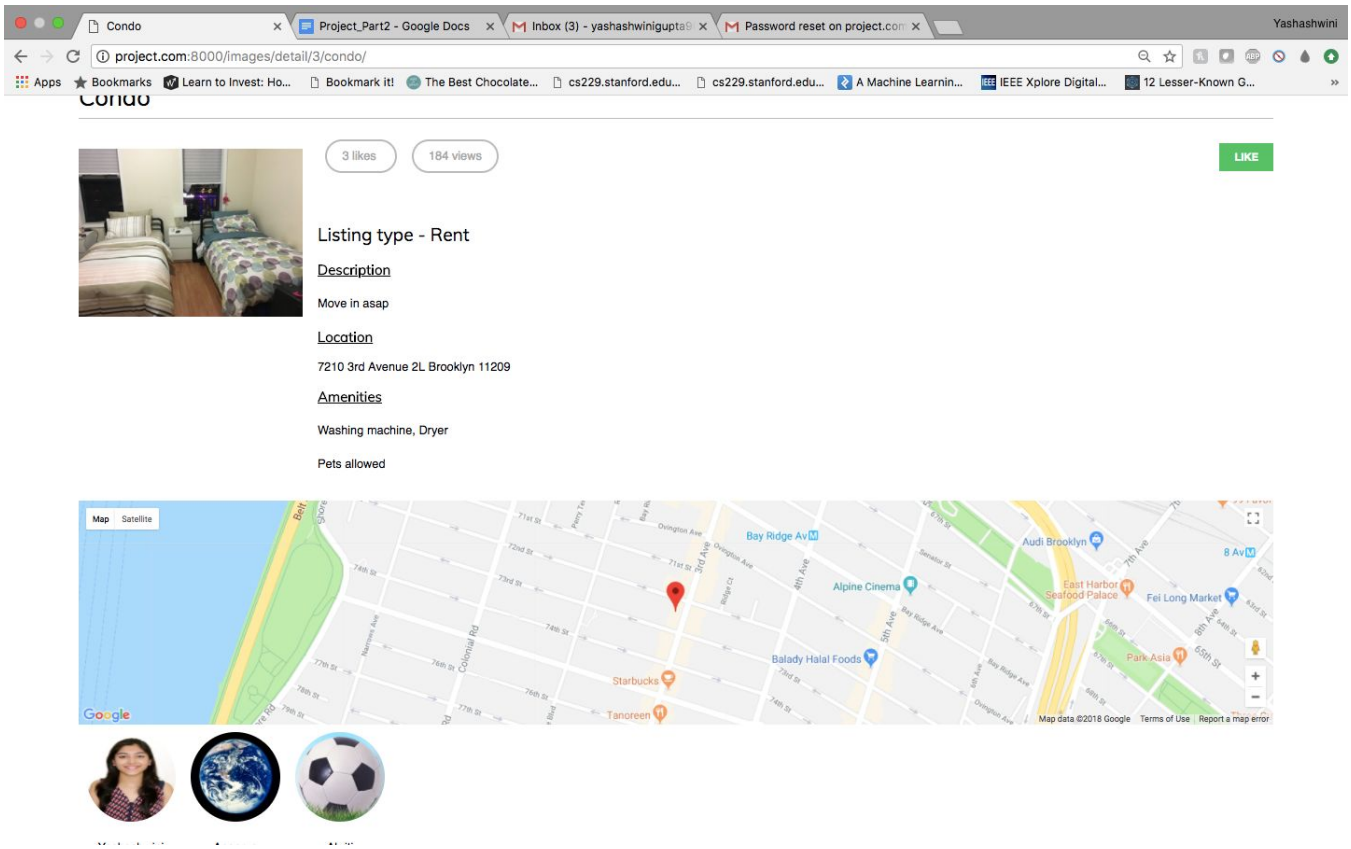


6) Location:

Steps used to add location marker to the template:

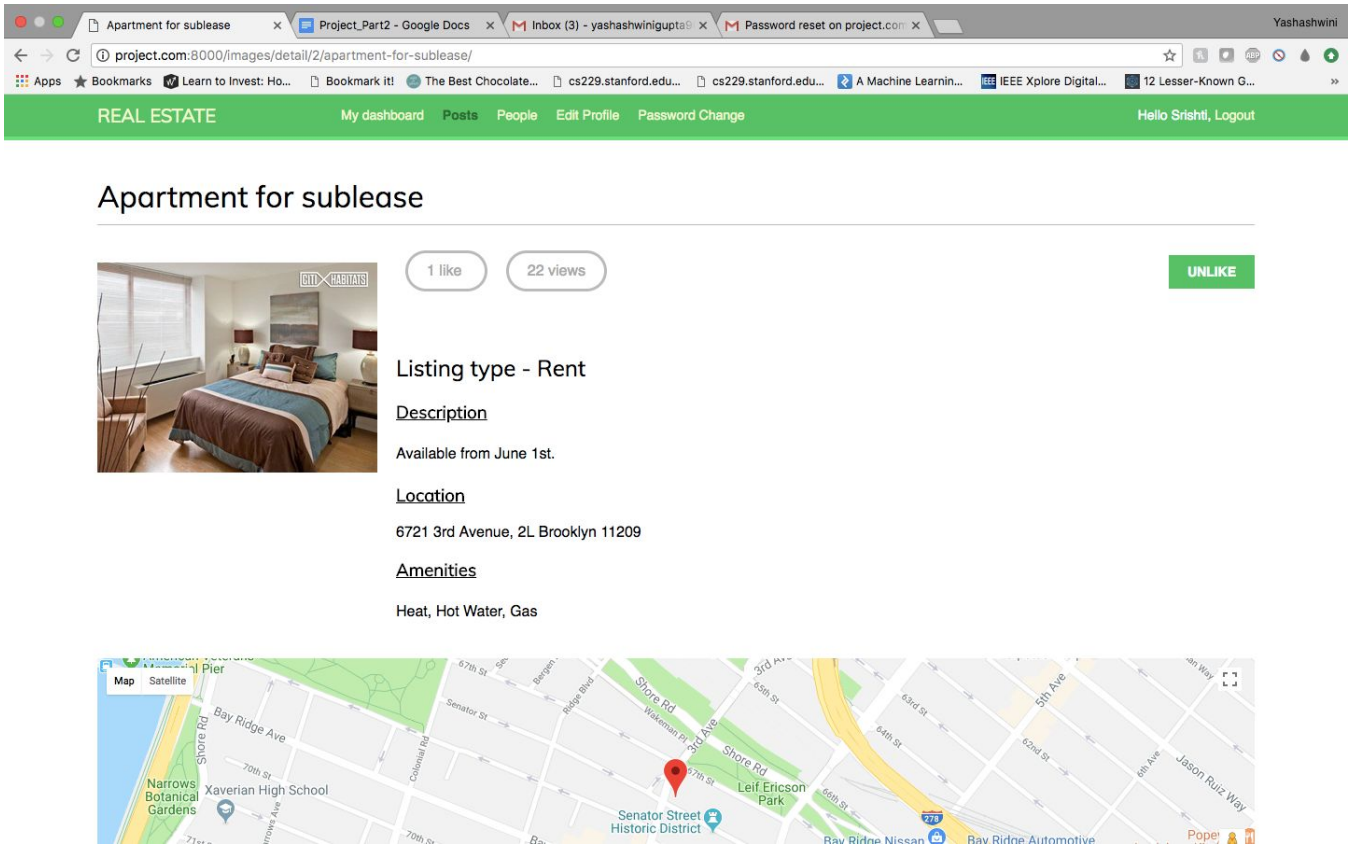
1. Obtained the address of the listing from the user through the create listing form.
2. Obtained API key from google maps.
3. Used google geocoding API to get json response for the given address.
4. Parsed json to obtain the latitude and longitude.
5. Rendered coordinates to the template and used asynchronous map to display the marker for each listing.


Condo
Project_Part2 - Google Docs
Inbox (3) - yashashwinigupta9
Password reset on project.com
Yashashwini
project.com:8000/images/detail/3/condo/
Apps
Bookmarks
Learn to Invest: Ho...
Bookmark it!
The Best Chocolate...
cs229.stanford.edu...
cs229.stanford.edu...
A Machine Learnin...
IEEE Xplore Digital...
12 Lesser-Known G...
REAL ESTATE
My dashboard
Posts
People
Edit Profile
Password Change
Hello Srishti, Logout
Condo
3 likes
184 views
LIKE
Listing type - Rent
Description
Move in asap
Location
7210 3rd Avenue 2L Brooklyn 11209
Amenities
Washing machine, Dryer
Pets allowed
Map
Satellite
Bay Ridge Av
Alpine Cinema
Audi Brooklyn
East Harbor Seafood Palace


Condo
Project_Part2 - Google Docs
Inbox (3) - yashashwinigupta9
Password reset on project.com
Yashashwini
project.com:8000/images/detail/3/condo/
Apps
Bookmarks
Learn to Invest: Ho...
Bookmark it!
The Best Chocolate...
cs229.stanford.edu...
cs229.stanford.edu...
A Machine Learnin...
IEEE Xplore Digital...
12 Lesser-Known G...
3 likes
184 views
LIKE
Listing type - Rent
Description
Move in asap
Location
7210 3rd Avenue 2L Brooklyn 11209
Amenities
Washing machine, Dryer
Pets allowed
Map
Satellite
Bay Ridge Av
Alpine Cinema
Audi Brooklyn
East Harbor Seafood Palace
Fei Long Market
Balady Halal Foods
Starbucks
Tanoreen
Park Asia
Google
Map data ©2018 Google
Terms of Use
Report a map error

7) LIKE / DISLIKE:

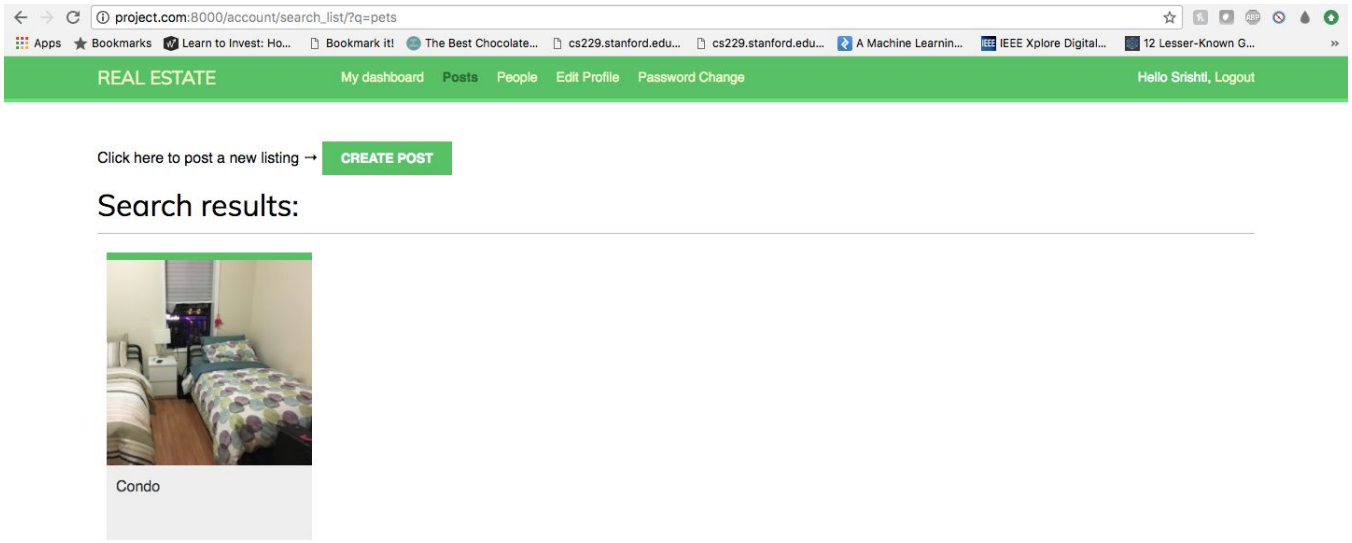
Used ajax and jquery to asynchronously update the like count.
Upon opening a particular post, a thumbnail of the profile picture of all the users who have liked that particular post appears at the bottom of that page.



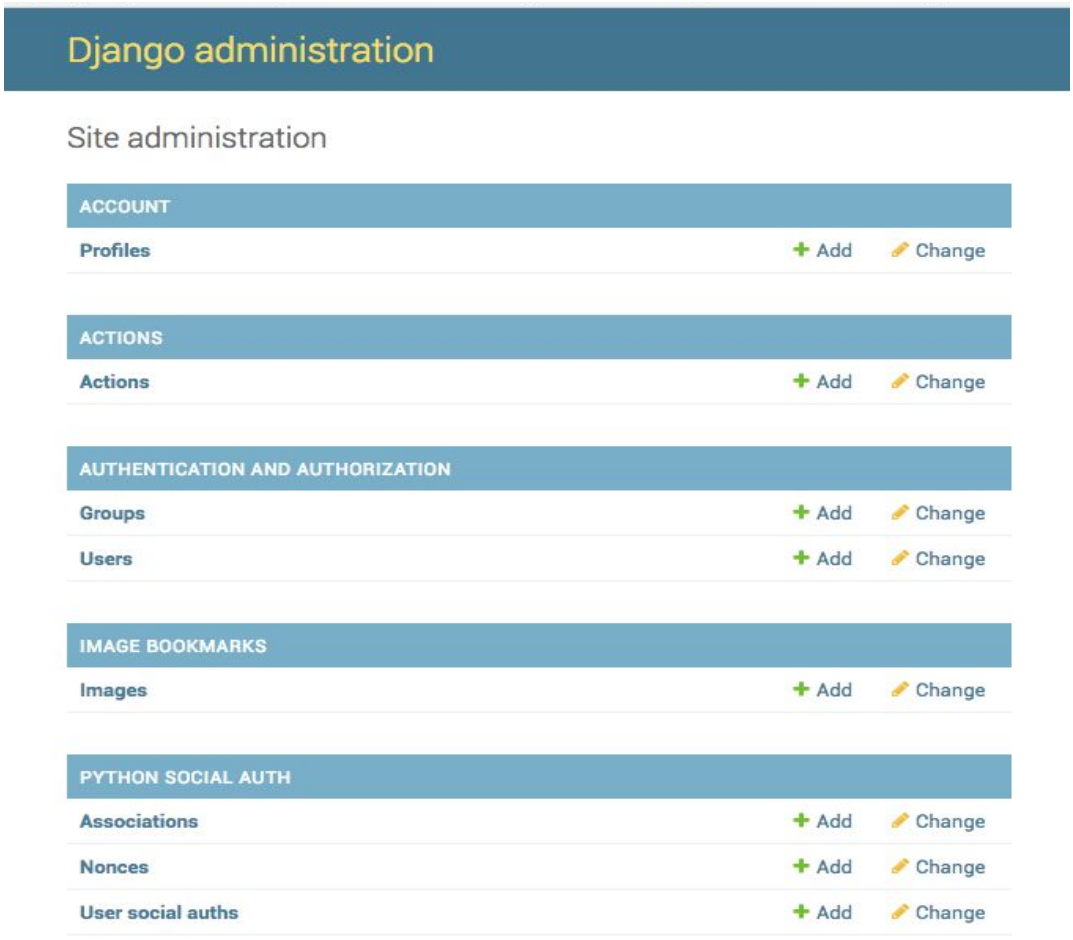
8) SEARCH:

_Implemented the following query to fetch listings containing the string searched by the user.

```
query = request.GET.get("q")
   if query=='':
      images = Image.objects.all()
   else:
      images =
Image.objects.filter(Q(amenities__icontains=query)|Q(title__icontains=query)|Q(city__icontains=query)|Q(price__icontains=query)|Q(street_address__icontains=query)|Q(description__icontains=query))
```



DATABASE TABLES:



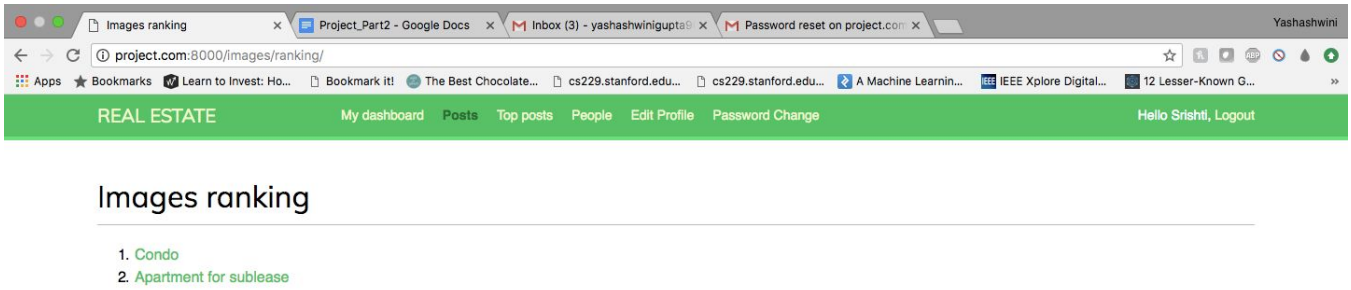
Extra Features:

Total views:

```
views.py — images | views.py — actions | Find Results | Find Results | views.py — account | models.py | forms.py | list.html | search_list.html

                                                    'form': form})

def image_detail(request, id, slug):
    image = get_object_or_404(Image, id=id, slug=slug)
    # increment total image views by 1
    total_views = r.incr('image:{}:views'.format(image.id))
    # increment image ranking by 1
    # html = urlopen("http://www.google.com/")
    map_url='https://maps.googleapis.com/maps/api/geocode/json?address='
    address=image.street_address+'+'+image.city+'+'+image.zipcode
    address=address.replace(' ', '+')
    # print(address)
    url=map_url+address+'&key=AIzaSyDdMyT_fvok7-onoNyQcEBhXUGqM_QPoZs'

    rs = requests.get(url)
    coords = rs.json()['results'][0]['geometry']['location']

    r.zincrby('image_ranking', image.id, 1)
    return render(request,
                  'images/image/detail.html',
                  {'section': 'images',
                   'image': image,
                   'coords': coords,
                   'total_views': total_views})


@ajax_required
@login_required
@require_POST
def image_like(request):
    image_id = request.POST.get('id')
    action = request.POST.get('action')
    if image_id and action:
        try:
            image = Image.objects.get(id=image_id)
            if action == 'like':
                image.users_like.add(request.user)
                create_action(request.user, 'likes', image)
            else:
                image.users_like.remove(request.user)
            return JsonResponse({'status':'ok'})
        except:
            pass
    return JsonResponse({'status':'ko'})
```

Ranking:

```
                create_action(request.user, 'likes', image)
            else:
                image.users_like.remove(request.user)
            return JsonResponse({'status':'ok'})
        except:
            pass
    return JsonResponse({'status':'ko'})


@login_required
def image_list(request):
    images = Image.objects.all()
    paginator = Paginator(images, 10)
    page = request.GET.get('page')
    try:
        images = paginator.page(page)
    except PageNotAnInteger:
        # If page is not an integer deliver the first page
        images = paginator.page(1)
    except EmptyPage:
        if request.is_ajax():
            # If the request is AJAX and the page is out of range return an empty page
            return HttpResponse('')
        # If page is out of range deliver last page of results
        images = paginator.page(paginator.num_pages)
    if request.is_ajax():
        return render(request,
                      'images/image/list_ajax.html',
                      {'section': 'images', 'images': images})
    return render(request,
                  'images/image/list.html',
                   {'section': 'images', 'images': images})


@login_required
def image_ranking(request):
    # get image ranking dictionary
    image_ranking = r.zrange('image_ranking', 0, -1, desc=True)[:10]
    image_ranking_ids = [int(id) for id in image_ranking]
    # get most viewed images
    most_viewed = list(Image.objects.filter(id__in=image_ranking_ids))
    most_viewed.sort(key=lambda x: image_ranking_ids.index(x.id))
    return render(request,
                  'images/image/ranking.html',
                  {'section': 'images',
                   'most_viewed': most_viewed})
```



FB Login:

Added facebook login through Social Authentication using FAcebook API to make logging in easier. (However, because of facebook's updated API policy since March 2018, only https sites are enabled to use facebook authentication for logging in)

Security:

Password checks
Added email verification for forget password sent through smtp server.
Ensured SQL Injection protection in databases

REQUIREMENTS:(TO RUN THE PROJECT)

Web Framework: Django
Database: sqlite
Backend: Python=3
Packages:

- Pillow
- Redis
- Sorl-thumbnail
- Social-django
- Python-social-auth

Additional:

- Email authentication backend settings(SMTP)
- Google_API_key
- Facebook_API_key