SOFTWARE DEVELOPMENT DOCUMENT

Industrial Training Report

On

"TWEET ME"

Submitted for the partial fulfillment of Bachelor of Engineering

Submitted by:

ROHINI T	**(1SI16CS086)**
SHRAVANI J	**(1SI16CS101)**
SHREELAKSHMI A N	**(1SI16CS102)**

Under the Guidance of:

Mr. VENKAT SUBRAMANIAN
Training Faculty

Department of Computer Science and Engineering

Siddaganga Institute of Technology, Tumakuru – 572103
(An Autonomous Institution, Affiliated to VTU, Belagavi& Recognized by AICTE, New Delhi)

2018 -2019

SIDDAGANGA INSTITUTE OF TECHNOLOGY,TUMAKURU-3

(An Autonomous Institution, Affiliated to VTU, Belagavi& Recognized by AICTE, New Delhi)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that that the Industrial Training Project entitled **"TWEET ME"** has been successfully carried out by **Rohini T (1SI16CS086), Shravani J(1SI16CS101) and Shreelakshmi A N (1SI16CS102)** of VI semester **Computer Science and Engineering**, **SIDDAGANGA INSTITUTE OF TECHNOLOGY** for the partial fulfillment of Bachelor of Engineering during the academic year 2018-2019.

Signature of the Guide
Mr. Venkat Subramanian
Training Faculty

Signature of the Convener
Mr. Deepak Singh
Co-Founder, Monkfox

Signature of the HOD

Dr. R. Sumathi

Prof. and Head, Dept. of CSE

ACKNOWLEDGEMENT

It’s our pleasure to thank all the individuals who have directly or indirectly helped and motivated us in the fulfillment of completion of the project work.

We express our gratitude to our respected Director, **Dr. M. N. Channabasappa** for his constant support in fulfilling our endeavors.

We would also like to thank our respected Principal, **Dr. K. P. Shivananda** for providing us with various facilities for carrying out this project.

We are also thankful to **Dr. R. Sumathi,** Professor and Head, Department of Computer Science and Engineering for giving us all the freedom**,** inspiration and encouragement to carry out this project successfully.

We thank our project guide **Mr. Venkat Subramanian.** Training Faculty, for providing the critical support on every step of this project development. His guidance gave us the environment to enhance our knowledge, skills and to reach the pinnacle with sheer determination, dedication and hard work. Without his aspiring support this project would not have been implemented.

We thank our project coordinator **Mr. Deepak Singh,** Co-Founder, Monkfox for allowing us to take up this idea as our mini project.

Shreelakshmi A N

Shravani J

Rohini T

ABSTRACT

Social media websites have emerged as one of the platforms to raise users' opinions and influence the way any business is commercialized. Opinion of people matters a lot to analyze how the propagation of information impacts the lives in a large-scale network like Twitter

This reports on the design of a sentiment analysis, extracting vast number of tweets. Results classify user's perception via tweets into positive and negative. Secondly, we discuss various techniques to carryout sentiment analysis on twitter data in detail.

INTRODUCTION

Tweet me is an online news and social networking service on which users post and interact with messages known as "tweets".

Registered users can post, like, and re-tweet tweets, but unregistered users can only read them. Users access Twitter through its website interface, through Short Message Service (SMS) or its mobile-device application software.

Twitter messages are public, but users can also send private "direct messages".Information about who has chosen to follow an account and who a user has chosen to follow is also public, though accounts can be changed to "protected" which limits this information to approved followers. Twitter collects personally identifiable information about its users and shares it with third parties as specified in its privacy policy. The service also reserves the right to sell this information as an asset if the company changes hands. While Twitter displays no advertising, advertisers can target users based on their history of tweets and may quote tweets in ads directed specifically to the user.

Tweet Me is also useful in the field of education, as an effective tool that can be used to encourage learning and idea, or knowledge sharing, in and outside the classroom.-By using or creating hash-tags, students and educators are able to communicate under specific categories of their choice, to enhance and promote education. Once teachers find someone they want to talk to, they can either direct message the person, or narrow down the hash-tag to make the topic of the conversation more specific.

Tweet-bots are capable of influencing public opinion about culture, products and political agendas by automatically generating mass amounts of tweets through imitating human communication.

SOFTWARE DEVELOPMENT DOCUMENT

REQUIREMENTS

- ✓ Click==7.0
- ✓ Django==2.1.4
- ✓ Djnago-crispy-forms==1.7.2
- ✓ djangorestframework==3.9.4
- ✓ SQLAlchemy==2.4.0
- ✓ itsdangerous==1.1.0
- ✓ Jinja2==2.10.1
- ✓ MarkupSafe==1.1.1
- ✓ mysqlclient==1.4.2. post1
- ✓ SQLAlchemy==1.3.5
- ✓ Werkzeug==0.15.4

SOURCE CODE

form.py

```
from django import forms
from django.contrib.auth import get_user_model

User = get_user_model()



class UserRegisterForm(forms.Form):
    username = forms.CharField()
    email = forms.CharField()
    password = forms.CharField(widget=forms.PasswordInput)
    password2 = forms.CharField(label='Confirm password', widget=forms.PasswordInput)

    def clean_password2(self):
        password = self.cleaned_data.get('password')
        password2 = self.cleaned_data.get('password2')
        if password != password2:
            raise forms.ValidationError('Password mush match')
        return password2

    def clean_username(self):
        username = self.cleaned_data.get('username')
        if User.objects.filter(username__icontains=username).exists():
            raise forms.ValidationError('This username exists')
        return username

    def clean_email(self):
        email = self.cleaned_data.get('email')
        if User.objects.filter(email__icontains=email).exists():
            raise forms.ValidationError('This email exists')
        return email
```

Views.py

```
from django.views import generic
from django.db import models
from django.http import HttpResponseRedirect
from django.shortcuts import render, get_object_or_404, redirect
from django.views import View
from django.contrib.auth import get_user_model
from .forms import UserRegisterForm

from .models import UserProfile
# Create your views here.
User = get_user_model()


class UserRegisterView(generic.FormView):
    form_class = UserRegisterForm
    template_name = 'accounts/user_register_form.html'
    success_url = '/login'

    def form_valid(self, form):
        username = form.cleaned_data.get('username')
        email = form.cleaned_data.get('email')
        password = form.cleaned_data.get('password')
        new_user = User.objects.create(username=username, email=email)
        new_user.set_password(password)
        new_user.save()
        return super(UserRegisterView, self).form_valid(form)



class UserDetailView(generic.DetailView):
    queryset = User.objects.all()
    template_name = 'accounts/user_detail.html'

    # def get_slug_field(self):
    #     return 'username'

    def get_object(self):
        return get_object_or_404(User, username__iexact=self.kwargs.get('username'))

    def get_context_data(self, *args, **kwargs):
        context = super(UserDetailView, self).get_context_data(*args, **kwargs)
        following = UserProfile.objects.is_following(self.request.user, self.get_object())
        context['following'] = following
        context['recommended'] = UserProfile.objects.recommended(self.request.user)
        return context



class UserFollowView(View):
    def get(self, request, username, *args, **kwargs):
        toggle_user = get_object_or_404(User, username__iexact=username)
        if request.user.is_authenticated:
            is_following = UserProfile.objects.toggle_follow(request.user, toggle_user)
        return redirect('profiles:detail', username=username)
        # url = reverse('profiles:detail', kwargs={"username": username})
        # HttpResponseRedirect(url)
```

Models.py

```
from django.db import models
from django.conf import settings
from django.db.models.signals import post_save
from django.urls import reverse_lazy
# Create your models here.


class UserProfileManager(models.Manager):
    use_for_related_fields = True
    def all(self):
        qs = self.get_queryset().all()
        try:
            if self.instance:
                qs = qs.exclude(user=self.instance)
        except:
            pass
        return qs

    def toggle_follow(self, user, to_toggle_user):
        user_profile, created = UserProfile.objects.get_or_create(user=user)  # (user_obj, true)
        if to_toggle_user in user_profile.following.all():
            user_profile.following.remove(to_toggle_user)
            added = False
        else:
            user_profile.following.add(to_toggle_user)
            added = True
        return added

    def is_following(self, user, followed_by_user):
        user_profile, created = UserProfile.objects.get_or_create(user=user)
        if created:
            return False
        if followed_by_user in user_profile.following.all():
            return True
        return False

    def recommended(self, user, limit_to=5):
        profile = user.profile
        following = profile.following.all()
        following = profile.get_following()
            qs = self.get_queryset().exclude(user__in=following).exclude(id=profile.id).order_by('?')
[:limit_to]
        return qs


class UserProfile(models.Model):
         user = models.OneToOneField(settings.AUTH_USER_MODEL, related_name='profile',
on_delete=models.CASCADE)
          following = models.ManyToManyField(settings.AUTH_USER_MODEL, blank=True,
related_name='followed_by')
    image = models.FileField(upload_to = 'profile_image',blank=True)
    # user.profile
    # user.profile.following -- users I follow
    # user.profile.followed_by -- users that follows me -- reverse relaationship

    objects = UserProfileManager() # UserProfile.objects.all()
    # abc = UserProfileManager() # UserProfile.abc.all()

    def _str_(self):
        return str(self.following.all().count())
    def get_following(self):
        users = self.following.all()  # User.objects.all().exclude(username=self.user.username)
        return users.exclude(username=self.user.username)

    def get_follow_url(self):
        return reverse_lazy('profiles:follow', kwargs={'username': self.user.username})

    def get_absolute_url(self):
        return reverse_lazy('profiles:detail', kwargs={'username': self.user.username})




# abc = User.objects.first()
# abc.save()

def post_save_user_receiver(sender, instance, created, *args, **kwargs):
    if created:
        new_profile = UserProfile.objects.get_or_create(user=instance)

post_save.connect(post_save_user_receiver, sender=settings.AUTH_USER_MODEL)
```

ROLES AND RESPONSIBILITIES

To make project successful, contribution of each team members is equally important. Hence each one has to take responsibilities to contribute towards work. Our project consists of three members where we divided work equally among ourselves such that pressure and work will reduce.

Team member 1: Took the responsibility of creating HTML files of different module which has to be connected to the backend database.

Team member 2: Took the responsibility of working on backend that is MYSQL database by creating tables, writing queries, insertion, deletion and updating of data into tables.

Team member 3: Took the responsibility of creating new.py python file which connects the HTML files and database.

CONCLUSION

This reports on the design of a sentiment analysis, extracting vast number of tweets. Results classify user's perception via tweets into positive and negative. Secondly, we discuss various techniques to carryout sentiment analysis on twitter data in detail.