CERTIFICATE

This is to certify that the project report entitled

CHAT APPLICATION

Submitted to Shree Sarasswathi Vidhyaah Mandheer

Sr Secondary School in fulfillment of the requirements

For completing the CBSE Board Practical Examination

2019 – 2020,is a bonafide work done by

DHARANEESH G G

Place:

Date :

Signature Of The Teacher

Signature Of The Principal

Signature Of Internal Examiner

Signature Of External Examiner

ACKNOWLEDGEMENT

My deep sense of gratitude to **Mrs. Surya K, M.Sc. , B.Ed., Computer Science Teacher** ,Shree Sarasswathi Vidhyaah Mandheer Sr Secondary School, Mettupalayam for suggesting me this project and for her sincere and able guidance and constructive criticism throughout this project work.

I express my humble gratitude to **Dr. Manimekalai Mohan** Managing Trustee, and **Shri S.Mohandoss**, Trustee of our school for their encouragement which ensured successful completion of this project.

I also express my sincere gratitude **to Mrs. Shanthakumari M.Sc., M.Phil., B.Ed., PGDCA.**, Principal, for the timely and tireless help and motivation.

Above all. I solemnly express my gratitude and humble thanks to my beloved parents and my brother for their pleasing advice and help through out this project.

TABLE OF CONTENT

MODULES USED

1. django

2. datetime

3. socket

And various Models within Django

WORKING DESCRIPTION

This project "Chat Application" is developed in Python, Django which are Object Oriented Programming Language.

Chatting is talking to other people through exchange of typed -in messages. It has grown popular because of the increasing demand for privacy and collaboration in this society where the public sharing domain dominates.

The specific purpose of this project is to make users allowing chat with their friend without internet.But the computers must be connected to same network.

The Output Of The Mentioned Functions Are Attached At The End Of The Document . . .

SOURCE CODE

1. urls.py:

```
from django.contrib import admin
from django.urls import path
from titans import views
from django.contrib import admin
from django.urls import path

urlpatterns = [
    path('admin/', admin.site.urls),
    path('',views.login),
    path('index',views.index),
    path("login",views.login),
    path("forget",views.forget),
    path("re_passcode",views.re_passcode),
    path("passchange", views.passchange),
    path("create",views.create),
    path('add',views.add),
    path('groupchat1',views.groupchat1),
    path("privatechat1",views.privatechat1),
    path('logout',views.groupchat1),
    path('dupindex',views.dupindex),
    path('privatechat2',views.privatechat2),
    path('privatechat3',views.privatechat3),
    path('bio',views.bio)
]
```

2. models.py:

```
from django.db import models

# Create your models here.
class usernamereg(models.Model):
    username = models.CharField(max_length=250,unique=True)
    question = models.CharField(max_length=250)
    answer = models.CharField(max_length=100)
    password = models.CharField(max_length=250)

    def __str__(self):
        return self.usernamereg

class Post(models.Model):
    username = models.CharField(max_length=200,unique=True)
    password = models.CharField(max_length=200)
    question = models.CharField(max_length=200)
    answers = models.CharField(max_length=200)
    email = models.CharField(max_length=200)
    anything = models.CharField(max_length=200)

    def __str__(self):
        return self.username
```

3. admin.py

```
from django.contrib import admin
from .models import Post
# Register your models here.
admin.site.register(Post)
# Register your models here.
```

4. views.py

```
from django.shortcuts import render
from django.contrib.auth.models import auth,User
from django.views.decorators.csrf import csrf_exempt
from .models import Post
import datetime
import socket

# Create your views here.

#Pre - Defined Variables
online=[]
username=''
message=''
messages=[" ADMIN : HI . . . everyone.","ADMIN : WELCOME "]
pri_online=[]
pri_message=[]
pri_dict={}


#Requesting Sign In page
@csrf_exempt
def login(request):
   return render(request , "login.html")

#Requesting forget password page
@csrf_exempt
def forget(request):
   return render(request , "forget.html")

#Requesting create account page
@csrf_exempt
def create(request):
   return render(request , "create.html")

#Sign In
@csrf_exempt
def index(request):
   hostname = socket.gethostname()
   IPAddr = socket.gethostbyname(hostname)
   username = str(request.POST.get("username"))
   password = str(request.POST.get("pass"))

   time = str(datetime.datetime.now())
```

```
 try:
       a = Post.objects.get(username=username)
       if (a):
          try:
             b=a.password
             if (b == password):
                f = open("records/signin", "a")
                f.write('USERNAME : ' + username)
                f.write('\n')
                f.write("PASSWORD : " + password)
                f.write('\n')
                f.write("TIME : " + time)
                f.write('\n')
                f.write("IP : " + IPAddr)
                f.write('\n')
                f.write("HOSTNAME : " + hostname)
                f.write('\n')
                f.write('\n')
                f.close()
                return render(request, "index.html", {'user': username})
             else:
                return render(request, "passerror.html")
          except:
             pass
    except:
       return render(request, "error.html")

#Creating account
@csrf_exempt
def add(request):
    username = request.POST.get('username')
    question = request.POST.get('question')
    answer = request.POST.get('answer')
    password = request.POST.get('password2')
    email = request.POST.get('email')
    bio = request.POST.get('bio')
    try:
       if Post.objects.get(username = username):
          return render(request,'create.html',{'data': 'User Name Exists'})
    except:
       a = Post(username=username,password=password,question=question,answers=answer,email=email,
anything=bio)
       a.save()
       return render(request , "login.html")


#Password change.Getting informations for change
@csrf_exempt
def re_passcode(request):
    username = request.POST.get('username')
    question = request.POST.get('question')
    answer = request.POST.get('answer')
```

```
    try:
        a = Post.objects.get(username=username)
        if (a):
            try:
                b = a.question
                if (b==question):
                    try:
                        c=a.answers
                        if (c==answer):
                            return render(request,'passchange.html', {'user':username})
                    except:
                        return render(request,"error.html")
            except:
                return render(request, "error.html")
    except:
        return render(request , 'error.html')

#Password Changing
@csrf_exempt
def passchange(request):
    username = request.POST.get('username')

    password = request.POST.get("passcode2")
    try:
        if(Post.objects.get(username=username)):

            a=Post.objects.get(username=username)
            a.password = password
            a.save()
    except:
        pass
    return render(request , "login.html")

#Bio page of user
@csrf_exempt
def bio(request):
    username = request.POST.get('username')
    a = Post.objects.get(username=username)
    b=a.email
    c=a.anything
    return render(request,'contact.html',{'user':username,'email':b,'bio':c})


#Requesting Home page
@csrf_exempt
def dupindex(request):
    username = request.POST.get('username')
    return render(request , 'index.html',{'user':username})

#group chat
@csrf_exempt
def groupchat1(request):
    username = request.POST.get('username')
```

```
    logout = request.POST.get('choice')

    message = request.POST.get("message")
    if (message != None):
        message = username + " : " + message
    if (username not in online):
        if (username != None):
            if (username != 'None'):
                online.append(username)
    else:
        pass
    if (message not in messages):
        f = open('records/groupcon.txt', "a+")
        if (message != None):
            f.write(message)
            f.write('\n')
            messages.append(message)
        f.close()
    if (logout == 'yes' or logout == 'YES' or logout == 'Yes'):
        online.remove(username)
        pri_online.remove(username)
        return render(request, 'login.html')
    else:
        pass
    return render(request, 'groupchat1.html', {'user': username, 'online': online, 'messages': messages})

#Displaying name in online client
@csrf_exempt
def privatechat1(request):
    username = request.POST.get('username')
    if (username not in pri_online):
        if (username != None):
            if(username !='None'):
                pri_online.append(username)
    else:
        pass

    return render(request , 'privatechat1.html',{'user':username,'pri_online':pri_online})


#Selecting User for private chat
@csrf_exempt
def privatechat2(request):
    username = request.POST.get('username')
    rec = request.POST.get("rec")

    if (username not in pri_online):
        if (username != None):
            if (username != 'None'):
                pri_online.append(username)
    else:
        pass
```

```
    if(rec not in pri_online):

        return render(request , 'notonline.html')

    if(rec == username):
        return render(request , 'notonline.html')

    return render(request, 'privatechat2.html', {'user': username, 'pri_online': pri_online, "rec": rec})

#Chatting with selected user
@csrf_exempt
def privatechat3(request):
    username = request.POST.get('username')
    rec = request.POST.get("rec")
    message = request.POST.get("message")
    if (message != None):
        message = username + " : " + message

    if (rec in pri_online):
        a = ''
        b = ''
        if (username > rec):
            a = rec + username
        else:
            b = username + rec

        if (a in pri_dict):
            if (message not in pri_dict[a]):
                pri_dict[a].append(message)

        if (b in pri_dict):
            if (message not in pri_dict[b]):
                pri_dict[b].append(message)

        if (a not in pri_dict and b not in pri_dict):
            if (username > rec):
                pri_dict[rec + username] = []
                pri_dict[a].append(message)
            else:
                pri_dict[username + rec] = []
                pri_dict[b].append(message)
        for i in pri_dict:
            if (username in i):
                if (rec in i):
                    z = pri_dict[i]
                    for j in range(z.count(None)):
                        z.remove(None)

                    return render(request, "privatechat2.html", {'user': username, 'rec': rec, 'list': z})

    if (rec not in pri_online):
        return render(request, 'notonline.html')
# - -  THE END  - -
```