SAMPLE CODE:

**User side views:**

from django.shortcuts import render, HttpResponse

from django.contrib import messages

from .forms import UserRegistrationForm

from .models import UserRegistrationModel

from django.conf import settings

import os

import pickle

# Create your views here.

def UserRegisterActions(request):

if request.method == 'POST':

form = UserRegistrationForm(request.POST)

if form.is\_valid():

print('Data is Valid')

form.save()

messages.success(request, 'You have been successfully registered')

form = UserRegistrationForm()

return render(request, 'UserRegistrations.html', {'form': form})

else:

messages.success(request, 'Email or Mobile Already Existed')

print("Invalid form")

else:

form = UserRegistrationForm()

return render(request, 'UserRegistrations.html', {'form': form})

def UserLoginCheck(request):

if request.method == "POST":

loginid = request.POST.get('loginname')

pswd = request.POST.get('pswd')

print("Login ID = ", loginid, ' Password = ', pswd)

try:

check = UserRegistrationModel.objects.get(loginid=loginid, password=pswd)

status = check.status

print('Status is = ', status)

if status == "activated":

request.session['id'] = check.id

request.session['loggeduser'] = check.name

request.session['loginid'] = loginid

request.session['email'] = check.email

print("User id At", check.id, status)

return render(request, 'users/UserHome.html', {})

else:

messages.success(request, 'Your Account Not at activated')

return render(request, 'UserLogin.html')

except Exception as e:

print('Exception is ', str(e))

pass

messages.success(request, 'Invalid Login id and password')

return render(request, 'UserLogin.html', {})

def UserHome(request):

return render(request, 'users/UserHome.html', {})

def usersViewDataset(request):

dataset = os.path.join(settings.MEDIA\_ROOT, 'EmmergencyDataset.csv')

import pandas as pd

df = pd.read\_csv(dataset)

df = df.to\_html(index=None)

return render(request, 'users/viewData.html', {'data': df})

def userClassificationResults(request):

import pandas as pd

from .utility import EmmergencyClassi

rf\_report = EmmergencyClassi.process\_randomForest()

dt\_report = EmmergencyClassi.process\_decesionTree()

nb\_report = EmmergencyClassi.process\_naiveBayes()

gb\_report = EmmergencyClassi.process\_knn()

lg\_report = EmmergencyClassi.process\_LogisticRegression()

svm\_report = EmmergencyClassi.process\_SVM()

rf\_report = pd.DataFrame(rf\_report).transpose()

rf\_report = pd.DataFrame(rf\_report)

dt\_report = pd.DataFrame(dt\_report).transpose()

dt\_report = pd.DataFrame(dt\_report)

nb\_report = pd.DataFrame(nb\_report).transpose()

nb\_report = pd.DataFrame(nb\_report)

gb\_report = pd.DataFrame(gb\_report).transpose()

gb\_report = pd.DataFrame(gb\_report)

lg\_report = pd.DataFrame(lg\_report).transpose()

lg\_report = pd.DataFrame(lg\_report)

svm\_report = pd.DataFrame(svm\_report).transpose()

svm\_report = pd.DataFrame(svm\_report)

# report\_df.to\_csv("rf\_report.csv")

return render(request, 'users/ml\_results.html',

{'lg': lg\_report.to\_html, 'svm': svm\_report.to\_html, 'rf': rf\_report.to\_html, 'dt': dt\_report.to\_html,

'nb': nb\_report.to\_html, 'gb': gb\_report.to\_html})

def UserPredictions(request):

if request.method == 'POST':

age = int(request.POST.get('age'))

gender = int(request.POST.get('gender'))

pulse = int(request.POST.get('pulse'))

systolicBloodPressure = int(request.POST.get('systolicBloodPressure'))

diastolicBloodPressure = int(request.POST.get('diastolicBloodPressure'))

respiratoryRate = int(request.POST.get('respiratoryRate'))

spo2 = float(request.POST.get('spo2'))

randomBloodSugar = int(request.POST.get('randomBloodSugar'))

temperature = float(request.POST.get('temperature'))

test\_data = [age, gender, pulse,systolicBloodPressure,diastolicBloodPressure,respiratoryRate,spo2,randomBloodSugar,temperature] # noqa: E501

model\_path = os.path.join(settings.MEDIA\_ROOT, 'alexmodel.pkl')

model = pickle.load(open(model\_path, 'rb'))

result = model.predict([test\_data])

if result[0] == 0:

msg = 'not Needed'

else:

msg = 'Needed'

print("Result=", result)

return render(request, "users/predictForm.html", {'result': msg})

else:

return render(request, "users/predictForm.html", {})

**Base.html:**

{% load static %}

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, maximum-scale=1">

<title>Machine Learning Based Patient Classification In Emergency Department</title>

<link rel="icon" href="{%static 'favicon.png'%}" type="image/png">

<link href="{%static 'css/bootstrap.min.css'%}" rel="stylesheet" type="text/css">

<link href="{%static 'css/style.css'%}" rel="stylesheet" type="text/css">

<link href="{%static 'css/font-awesome.css'%}" rel="stylesheet" type="text/css">

<link href="{%static 'css/animate.css'%}" rel="stylesheet" type="text/css">

</head>

<body>

<!--Header\_section-->

<header id="header\_wrapper">

<div class="container">

<div class="header\_box">

<div class="logo"><a href="#"><h2>Emergency Classification</h2></a></div>

<nav class="navbar navbar-inverse" role="navigation">

<div class="navbar-header">

<button type="button" id="nav-toggle" class="navbar-toggle" data-toggle="collapse"

data-target="#main-nav"><span class="sr-only">Toggle navigation</span> <span

class="icon-bar"></span> <span class="icon-bar"></span> <span class="icon-bar"></span>

</button>

</div>

<div id="main-nav" class="collapse navbar-collapse navStyle">

<ul class="nav navbar-nav" id="mainNav">

<li><a href="{%url 'index'%}">Home</a></li>

<li><a href="{%url 'UserLogin'%}">Login</a></li>

<li><a href="{%url 'AdminLogin'%}">Admin</a></li>

<li><a href="{%url 'UserRegister'%}">Registrations</a></li>

</ul>

</div>

</nav>

</div>

</div>

</header>

<!--Header\_section-->

{%block contents%}

<!--Aboutus-->

{%endblock%}

<!--Footer-->

<footer class="footer\_wrapper" id="contact">

<div class="container">

<div class="footer\_bottom"><span>Copyright © 2024, Template by <a

href="#">Alex Corporations</a>. </span></div>

</div>

</footer>

<script type="text/javascript" src="{%static 'js/jquery-1.11.0.min.js'%}"></script>

<script type="text/javascript" src="{%static 'js/bootstrap.min.js'%}"></script>

<script type="text/javascript" src="{%static 'js/jquery-scrolltofixed.js'%}"></script>

<script type="text/javascript" src="{%static 'js/jquery.nav.js'%}"></script>

<script type="text/javascript" src="{%static 'js/jquery.easing.1.3.js'%}"></script>

<script type="text/javascript" src="{%static 'js/jquery.isotope.js'%}"></script>

<script type="text/javascript" src="{%static 'js/wow.js'%}"></script>

<script type="text/javascript" src="{%static 'js/custom.js'%}"></script>

</body>

</html>

**Admin side views:**

from django.shortcuts import render,HttpResponse

from django.contrib import messages

from users.models import UserRegistrationModel

# Create your views here.

def AdminLoginCheck(request):

if request.method == 'POST':

usrid = request.POST.get('loginid')

pswd = request.POST.get('pswd')

print("User ID is = ", usrid)

if usrid == 'admin' and pswd == 'admin':

return render(request, 'admins/AdminHome.html')

elif usrid == 'Admin' and pswd == 'Admin':

return render(request, 'admins/AdminHome.html')

else:

messages.success(request, 'Please Check Your Login Details')

return render(request, 'AdminLogin.html', {})

def AdminHome(request):

return render(request, 'admins/AdminHome.html')

def ViewRegisteredUsers(request):

data = UserRegistrationModel.objects.all()

return render(request, 'admins/RegisteredUsers.html', {'data': data})

def AdminActivaUsers(request):

if request.method == 'GET':

id = request.GET.get('uid')

status = 'activated'

print("PID = ", id, status)

UserRegistrationModel.objects.filter(id=id).update(status=status)

data = UserRegistrationModel.objects.all()

return render(request, 'admins/RegisteredUsers.html', {'data': data})

def adminResults(request):

import pandas as pd

from users.utility import EmmergencyClassi

rf\_report = EmmergencyClassi.process\_randomForest()

dt\_report = EmmergencyClassi.process\_decesionTree()

nb\_report = EmmergencyClassi.process\_naiveBayes()

gb\_report = EmmergencyClassi.process\_knn()

lg\_report = EmmergencyClassi.process\_LogisticRegression()

svm\_report = EmmergencyClassi.process\_SVM()

rf\_report = pd.DataFrame(rf\_report).transpose()

rf\_report = pd.DataFrame(rf\_report)

dt\_report = pd.DataFrame(dt\_report).transpose()

dt\_report = pd.DataFrame(dt\_report)

nb\_report = pd.DataFrame(nb\_report).transpose()

nb\_report = pd.DataFrame(nb\_report)

gb\_report = pd.DataFrame(gb\_report).transpose()

gb\_report = pd.DataFrame(gb\_report)

lg\_report = pd.DataFrame(lg\_report).transpose()

lg\_report = pd.DataFrame(lg\_report)

svm\_report = pd.DataFrame(svm\_report).transpose()

svm\_report = pd.DataFrame(svm\_report)

# report\_df.to\_csv("rf\_report.csv")

return render(request, 'admins/results.html',

{'lg': lg\_report.to\_html, 'svm': svm\_report.to\_html, 'rf': rf\_report.to\_html, 'dt': dt\_report.to\_html,

'nb': nb\_report.to\_html, 'gb': gb\_report.to\_html})