**SAMPLE CODE:**

User side views:

# Create your views here.

from django.shortcuts import render, HttpResponse

from django.contrib import messages

from .forms import UserRegistrationForm

from .models import UserRegistrationModel, EncryptionImageModels

from django.conf import settings

from django.core.files.storage import FileSystemStorage

import numpy as np

import string

import random

import os

# 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('loginid')

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/UserHomePage.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/UserHomePage.html', {})

def UserShareImages(request):

if request.method == 'POST':

image\_file = request.FILES['file']

fromUser = request.session['loggeduser']

toUser = request.POST.get('usr')

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

# let's check if it is a csv file

# if not image\_file.name.endswith('.png'):

# messages.error(request, 'THIS IS NOT A PNG FILE')

fs = FileSystemStorage(location="media/originals/")

filename = fs.save(image\_file.name, image\_file)

outputName = filename.split(".")[0] + ".png"

# detect\_filename = fs.save(image\_file.name, image\_file)

input\_image = "/media/originals/" + filename # fs.url(filename)

output\_image = "/media/output/" + outputName

path1 = os.path.join(settings.MEDIA\_ROOT, 'originals', filename)

path2 = os.path.join(settings.MEDIA\_ROOT, 'output', outputName)

print("Image path ", input\_image)

print("Output Image ", output\_image)

# path1 = os.path.join()

secretKey = ''.join(random.choices(string.ascii\_uppercase + string.digits, k=10))

print("secret Key:", secretKey)

from cryptosteganography import CryptoSteganography

crypto\_steganography = CryptoSteganography(secretKey)

crypto\_steganography.hide(path1, path2, msg)

EncryptionImageModels.objects.create(fromName=fromUser, toName=toUser, secretKey=secretKey, path1=filename,

path2=outputName)

return render(request, "users/ShareImages\_success.html", {'path1': input\_image, 'path2': output\_image})

else:

from django.db.models import Q

loginid = request.session['loginid']

users = UserRegistrationModel.objects.filter(~Q(loginid=loginid), ~Q(status='waiting'))

# users = UserRegistrationModel.objects.exclude(loginid=loginid, status='active')

return render(request, "users/ShareImages.html", {'users': users})

def UserViewImages(request):

loginid = request.session['loginid']

data = EncryptionImageModels.objects.filter(toName=loginid)

return render(request, "users/ViewImages.html", {'data': data})

def getMessageFromImage(request):

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

data = EncryptionImageModels.objects.get(id=id)

outputImage = data.path2

path2 = os.path.join(settings.MEDIA\_ROOT,'output', outputImage)

from cryptosteganography import CryptoSteganography

secretKey = data.secretKey

crypto\_steganography = CryptoSteganography(secretKey)

secret = crypto\_steganography.retrieve(path2)

loginid = request.session['loginid']

data = EncryptionImageModels.objects.filter(toName=loginid)

return render(request, "users/ViewImages.html", {'data': data,'msg': secret})

Base.html:

{% load static %}

<!DOCTYPE html>

<html>

<head>

<!-- Basic -->

<meta charset="utf-8"/>

<meta http-equiv="X-UA-Compatible" content="IE=edge"/>

<!-- Mobile Metas -->

<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no"/>

<!-- Site Metas -->

<meta name="keywords" content=""/>

<meta name="description" content=""/>

<meta name="author" content=""/>

<title>Peculiar Image Encryption</title>

<!-- slider stylesheet -->

<link rel="stylesheet" type="text/css"

href="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.1.3/assets/owl.carousel.min.css"/>

<!-- bootstrap core css -->

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

<!-- fonts style -->

<link href="https://fonts.googleapis.com/css?family=Poppins:400,600,700&display=swap" rel="stylesheet">

<!-- Custom styles for this template -->

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

<!-- responsive style -->

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

</head>

<body class="sub\_page">

<div class="hero\_area">

<!-- header section strats -->

<header class="header\_section">

<div class="container">

<nav class="navbar navbar-expand-lg custom\_nav-container ">

<a class="navbar-brand" href="{% url 'index' %}">

<img src="{%static 'images/logo.png'%}" alt="">

<span>

PECULIAR IMAGE ENCRYPTION

</span>

</a>

<button class="navbar-toggler" type="button" data-toggle="collapse"

data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent"

aria-expanded="false" aria-label="Toggle navigation">

<span class="s-1"> </span>

<span class="s-2"> </span>

<span class="s-3"> </span>

</button>

<div class="collapse navbar-collapse" id="navbarSupportedContent">

<div class="d-flex ml-auto flex-column flex-lg-row align-items-center">

<ul class="navbar-nav ">

<li class="nav-item ">

<a class="nav-link" href="{% url 'index' %}">Home</a>

</li>

<li class="nav-item">

<a class="nav-link" href="{% url 'UserLogin' %}">User</a>

</li>

<li class="nav-item">

<a class="nav-link" href="{% url 'AdminLogin' %}">Admin</a>

</li>

<li class="nav-item">

<a class="nav-link" href="{% url 'UserRegister' %}">Register</a>

</li>

</ul>

</div>

</div>

</nav>

</div>

</header>

<!-- end header section -->

</div>

{%block contents%}

{%endblock%}

<!-- footer section -->

<footer class="container-fluid footer\_section">

<div class="container">

<div class="row">

<div class="col-lg-7 col-md-9 mx-auto">

<p>

&copy; 2024 All Rights Reserved By

<a href="#">Alex Corporations</a>

</p>

</div>

</div>

</div>

</footer>

<!-- footer section -->

<script src="{%static 'js/jquery-3.4.1.min.js'%}"></script>

<script src="{%static 'js/bootstrap.js'%}"></script>

</body>

</html>

Admin side views:

from django.shortcuts import render, HttpResponse

from django.contrib import messages

from users.models import UserRegistrationModel, EncryptionImageModels

# 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')

else:

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

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

def AdminHome(request):

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

def RegisterUsersView(request):

data = UserRegistrationModel.objects.all()

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

def ActivaUsers(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/viewregisterusers.html', {'data': data})

def DeleteUsers(request):

if request.method == 'GET':

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

print("Delete PID = ", id)

UserRegistrationModel.objects.filter(id=id).delete()

data = UserRegistrationModel.objects.all()

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

def BlockUsers(request):

if request.method == 'GET':

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

status = 'waiting'

print("PID = ", id, status)

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

data = UserRegistrationModel.objects.all()

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

def AdminViewResults(request):

data = EncryptionImageModels.objects.all()

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