**Apr 25**

**Implementing signup button functionality**

**-->**auth application having form class to provide login form.

**-->**But auth application does not contain any form class for signup functionality.

**-->**If a user signup, compulsory that information should be stored in database(user table)

**-->**Display form to signup and that information should be stored inside database directly. For such type of requirement it is highly recommended to go for model based form.

* **forms.py**

from django import forms

from django.contrib.auth.models import User

class SignUpForm(forms.ModelForm):

class Meta:

model = User

fields = ['username','password','email','first\_name','last\_name']

* **views.py**

from testapp.forms import SignUpForm

def signup\_view(request):

form = SignUpForm()

return render(request,'testapp/signup.html',{'form':form})

* **signup.html**

<!DOCTYPE html>

{% extends 'testapp/base.html' %}

{% block body\_block%}

<div class="jumbotron">

<div class="container">

<h1>Please SignUp To Write Exams....</h1>

<form method="post">

{{form.as\_p}}

{% csrf\_token %}

<input type="submit" name="" value="SignUp">

</form>

</div>

</div>

{% endblock %}

* **urls.py**

path('signup/',views.signup\_view)

* **base.html**

<a class="nav-link" href="/signup">Signup</a>

* **views.py**

from testapp.forms import SignUpForm

from django.http import HttpResponseRedirect

def signup\_view(request):

form = SignUpForm()

if request.method == 'POST':

form = SignUpForm(request.POST)

user = form.save()

user.set\_password(user.password)#to hash password

user.save()

return HttpResponseRedirect('/accounts/login')

return render(request,'testapp/signup.html',{'form':form})

**-->**In django auth application, User model, the password should not be saved directly. It should be hashed by using some security algorithm. The default password hashing algorithm:pbkdf2\_sha256

form.save():We are trying to save password in the plain text form.

Invalid password format or unknown hashing algorithm.

**Password hashers:**

The default password hasher:pbkdf2\_sha256

We can use other more secured password hashers also

like argon2, bcrypt etc....

pip install bcrypt

pip install django[argon2]

More secured algorithm is argon2 followed by bcrypt and then pbkdf2\_sha256.

In settings.py we have to configure password hashers as...........

PASSWORD\_HASHERS = [

'django.contrib.auth.hashers.Argon2PasswordHasher',

'django.contrib.auth.hashers.BCryptSHA256PasswordHasher',

'django.contrib.auth.hashers.BCryptPasswordHasher',

'django.contrib.auth.hashers.PBKDF2PasswordHasher',

'django.contrib.auth.hashers.PBKDF2SHA1PasswordHasher',

]

**CHAPTER-10**

Class Based Views and CRUD operations by using both CBVs and FBVs.

**CRUD Operations**

FBVs--->Function Based View

CBVs-->Class Based Views

Django ORM

C-->Create(Insert Operation)

R-->Retrieve/Read(select)

U-->Update(update)

D-->Delete(delete)

CRUD/CURD

**CRUD Operations on FBV's**

D:\Django\_20MAR\_7PM>django-admin startproject fbvcrudproject

D:\Django\_20MAR\_7PM>cd fbvcrudproject

D:\Django\_20MAR\_7PM\fbvcrudproject>py manage.py startapp testapp

-->Add app in settings.py

* **models.py**

class Employee(models.Model):

eno = models.IntegerField()

ename = models.CharField(max\_length=64)

esal = models.FloatField()

eaddr = models.CharField(max\_length=128)

-->makemigrations and migrate

* **admin.py**

from testapp.models import Employee

class EmployeeAdmin(admin.ModelAdmin):

list\_display = ['eno','ename','esal','eaddr']

admin.site.register(Employee, EmployeeAdmin)

* **populate.py**

import os

os.environ.setdefault('DJANGO\_SETTINGS\_MODULE', 'fbvcrudproject.settings')

import django

django.setup()

from testapp.models import Employee

from faker import Faker

from random import \*

faker = Faker()

def populate(n):

for i in range(n):

feno = randint(1001,9999)

fename = faker.name()

fesal = randint(10000,20000)

feaddr = faker.city()

emp\_record = Employee.objects.get\_or\_create(

eno = feno,

ename = fename,

esal = fesal,

eaddr = feaddr)

n = int(input('Enter number of employees:'))

populate(n)

print(f'{n} Records Inserted Successfully....')