MODAL FORM

1 - create forms.py

2 - in forms.py

```
from django import forms
from .models import Product

class ProductForm(forms.ModelForm):
    class Meta:
        model=Product
        fields=[
            'name',
            'description',
            'price'
        ]
```

3 - models.py (you don't have to do any thing extra)

```
class Product(models.Model):
    name = models.CharField(max_length=200)
    description = models.TextField()
    price = models.IntegerField(default=0)
```

4 - in view.py

```
from .models import Product

from .forms import ProductForm


def index(request):

    form=ProductForm(request.POST or None)

    if form.is_valid():
        print('form is valid')
        form.save()
        form=ProductForm  # reseting the form
    else:
        print(form.errors)

    context={
        "form": form
    }

    return render(request,'index.html',context)
```

5 - in index.html

```
<form method="post">
    {% csrf_token %}

    {{ form.as_p }}
    <input type="submit" value="submit"/>

</form>
```

FORM IN DJANGO

1 create form.py

2 - in forms.py

```
from django import forms

class ProductForm(forms.Form):
        name = forms.CharField()
        description = forms.CharField()
        price = forms.DecimalField()
```

3 - models.py (you don't have to do any thing extra)

```
class Product(models.Model):
    name = models.CharField(max_length=200)
    description = models.TextField()
    price = models.IntegerField(default=0)
```

4 - in view.py

```
from .forms import ProductForm
from .models import Product

    form = ProductForm()

    if request.method == 'POST':

        form=ProductForm(request.POST)

        if form.is_valid():

            Product.objects.create(**form.cleaned_data) # ** turn cleaned_data into arguments

            print(form.cleaned_data)

            form = ProductForm() #resting a form

        else:
            print(form.errors)

    context={
        'form':form
    }
    return render(request,'index.html',context)
```

5 - in index.html

```
<form method="post">
    {% csrf_token %}

    {{ form.as_p }}
    <input type="submit" value="submit"/>
```

```
</form>
```

widget for form

```
class ProductForm(forms.Form):

    name = forms.CharField(label='enter name',widget=forms.TextInput(
                                                attrs={
                                                    "class":"pname",
                                                    "placeholder":"enter your name"
                                                }

    ))
    description = forms.CharField(required=False,widget=forms.Textarea(
                                                  attrs={
                                                      "class":"pdes",
                                                      "id":"myid",
                                                      "row": 20,
                                                      "colums": 30
                                                  }
                                                )
                            )
    price = forms.DecimalField(initial='100')
```

widget for modal form

```
class ProductForm(forms.ModelForm):

    name = forms.CharField(label='enter name',widget=forms.TextInput(
                                                attrs={
                                                    "class":"pname",
                                                    "placeholder":"enter your name"
                                                }

    ))
    description = forms.CharField(required=False,widget=forms.Textarea(
                                                  attrs={
                                                      "class":"pdes",
                                                      "id":"myid",
                                                      "row": 20,
                                                      "colums": 30
                                                  }
                                                )
                            )
    price = forms.DecimalField(initial='100')

    class Meta:
        model=Product
        fields=[
            'name',
            'description',
            'price'
        ]
```

VALIDATION In FORM FIELDS

#If name has ‘nee’ in it than name is valid else forms.ValidationError will occur

```
class ProductModalForm(forms.ModelForm):

class Meta:
        model=Product
        fields=[
            'name',
            'description',
            'price'
        ]

    def clean_name(self):

        name=self.cleaned_data.get('name')
        if "nee" in name:
            return  name
        else:
            raise forms.ValidationError("name does not have a nee in it”)
```

instance (socialmedia—> blog/views—>edit_profile())

Instance are used when we need to autofill the form with data present in database

Ex when user click my profile then its saved data (username,first_name,email extra) will automatically shown in the form

Views.py

```
def edit_profile(request):

    if request.method =="POST":

            user_Edit_form = UserEditForm(data=request.POST or None , instance=request.user)
            Profile_Edit_form = ProfileEditForm(data=request.POST or None,
instance=request.user.profile, files=request.FILES)


            if user_Edit_form.is_valid() and Profile_Edit_form.is_valid():
                user_Edit_form.save()
                Profile_Edit_form.save()
                return redirect('edit_profile')

    else:

        user_Edit_form= UserEditForm(instance=request.user)
        Profile_Edit_form = ProfileEditForm(instance=request.user.profile)

        context={
            'user_form':user_Edit_form,
            'prfile_form':Profile_Edit_form,
        }

        return render(request,’blog/edit_profile.html’,context)
```

Modalform & form

Forms.py

```
class UserEditForm(forms.ModelForm):
    class Meta:
        model = User
        fields = (
            'username',
            'first_name',
            'last_name',
            'email',
        )

class ProfileEditForm(forms.ModelForm):
    class Meta:
        model=Profile
        fields=(
            'dob',
            'photo',
        )
```