**Django Forms**

Django provides a Form class which is used to create HTML forms. It describes a form and how it works and appears.

Each field of the form class map to the HTML form **<input>** element and each one is a class itself, it manages form data and performs validation while submitting the form.

**Put this code into the forms.py file.**

**from** django **import** forms  
**class** EmployeeForm(forms.Form):  
 name = forms.CharField(max\_length=50)  
 age = forms.IntegerField()  
 contact\_no = forms.IntegerField()  
 email = forms.CharField(max\_length=100)  
 username = forms.CharField(max\_length=50)  
 password = forms.CharField(max\_length=100)

**Instantiate the form in views.py file**

**from** django.shortcuts **import** render  
**from** formapp.forms **import** EmployeeForm  
**def** showIndex(request):  
 emp = EmployeeForm()  
 render(request,**"index.html"**,{**"form"**:emp})

**index.html**

<form action="//" method="post">  
 {% csrf\_token %}  
 {{ f.as\_p }}  
 <input type="submit" value="submit">  
</form>

**Example 1:**

**To Show a Simple Html form using forms.py File**

**forms.py**

from django import forms

class Regsiter(forms.Form):

idno = forms.IntegerField()

name = forms.CharField()

salary = forms.DecimalField()

**views.py**

def showIndex(request):

r = Regsiter()

return render(request,"index.html",{"form":r})

def display(request):

idno = request.POST.get("idno")

name = request.POST.get("name")

salary = request.POST.get("salary")

return HttpResponse(idno+name+salary)

**index.html**

<form action="/display/" method="post">

{% csrf\_token %}

<table align="center" border="2">

<tr><th colspan="2">Register</th></tr>

{{ form }}

<tr><th colspan="2">

<button type="submit">Register</button>

</th></tr>

</table>

</form>

**urls.py**

path('index/',views.showIndex),

path('display/',views.display),

**Example 2**

**HTML Form with labels and Help Text in forms.py File**

**forms.py**

**class** Register(forms.Form):  
 idno = forms.IntegerField(label=**"ID"**,help\_text=**"Only No's"**)  
 name = forms.CharField(max\_length=30,label=**"NAME"**, help\_text=**"Only char's"**)  
 contact = forms.IntegerField(label=**"CONTACT NO"**, help\_text=**"only no's"**)  
 email = forms.EmailField(max\_length=100,label=**"EMAIL"**, help\_text=**"Email Only"**)  
 password = forms.CharField(widget=forms.PasswordInput, max\_length=8,label=**"PASSWORD"**,help\_text=**"Must be 8Char's Only"**)

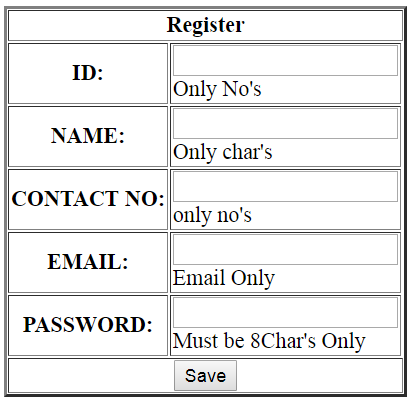
**views.py**

**def** showIndex(request):  
 r = Register()  
 **return** render(request,**"index.html"**,{**"f"**:r})  
  
**def** saveDetails(request):  
 idno = request.POST.get(**"idno"**)  
 name = request.POST.get(**"name"**)  
 contact = request.POST.get(**"contact"**)  
 email = request.POST.get(**"email"**)  
 password = request.POST.get(**"password"**)  
 **return** HttpResponse(idno+name+contact+email+password)

**index.html [Using form variable directly in Html]**

<form action="/savedetails/" method="post">  
 {% csrf\_token %}  
<table align="center" border="2">  
 <tr><th colspan="3">Register</th></tr>  
 {{ f }}  
<tr><th colspan="3">  
 <button type="submit">Save</button>  
 </th></tr>  
</table>  
</form>

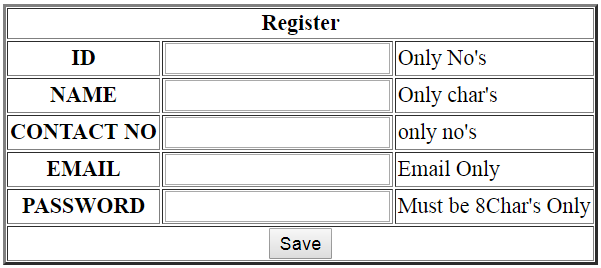
**Output Of Above Html Program**



**index.html [Using Form Fields Manually]**

<form action="/savedetails/" method="post">  
 {% csrf\_token %}  
<table align="center" border="2">  
 <tr><th colspan="3">Register</th></tr>  
 <tr>  
 <th>{{ f.idno.label }}</th>  
 <th>{{ f.idno }}</th>  
 <td>{{ f.idno.help\_text }}</td>  
 </tr>  
 <tr>  
 <th>{{ f.name.label }}</th>  
 <th>{{ f.name }}</th>  
 <td>{{ f.name.help\_text }}</td>  
 </tr>  
 <tr>  
 <th>{{ f.contact.label }}</th>  
 <th>{{ f.contact }}</th>  
 <td>{{ f.contact.help\_text }}</td>  
 </tr>  
 <tr>  
 <th>{{ f.email.label }}</th>  
 <th>{{ f.email }}</th>  
 <td>{{ f.email.help\_text }}</td>  
 </tr>  
 <tr>  
 <th>{{ f.password.label }}</th>  
 <th>{{ f.password }}</th>  
 <td>{{ f.password.help\_text }}</td>  
 </tr>  
 <tr><th colspan="3">  
 <button type="submit">Save</button>  
 </th></tr>  
</table>  
</form>

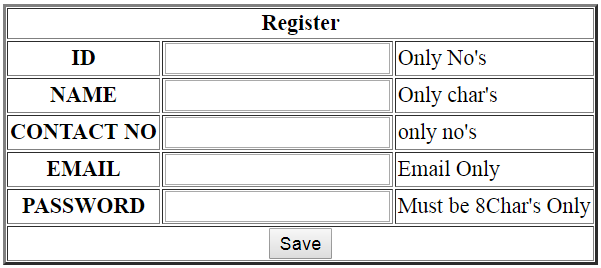
**Output Of Above Html Program**



**index.html [ Using for loop on form fields ]**

<form action="/savedetails/" method="post">  
 {% csrf\_token %}  
<table align="center" border="2">  
 <tr><th colspan="3">Register</th></tr>  
  
 {% for x in f %}  
 <tr>  
 <th>{{ x.label }}</th>  
 <td>{{ x }}</td>  
 <td>{{ x.help\_text }}</td>  
 </tr>  
 {% endfor %}  
  
 <tr><th colspan="3">  
 <button type="submit">Save</button>  
 </th></tr>  
</table>  
</form>

**Output Of Above Html Program**



**urls.py**

path(**''**, views.showIndex),  
path(**'savedetails/'**, views.saveDetails),

|  |  |  |
| --- | --- | --- |
| **Name** | **HTML Input** | **Empty value** |
| **BooleanField** | **CheckboxInput** | **False** |
| **CharField** | **TextInput** | **Whatever you've given as empty\_value.** |
| **ChoiceField** | **Select** | **'' (an empty string)** |
| **DateField** | **DateInput** | **None** |
| **DateTimeField** | **DateTimeInput** | **None** |
| **DecimalField** | **NumberInput** | **None** |
| **EmailField** | **EmailInput** | **'' (an empty string)** |
| **FileField** | **ClearableFileInput** | **None** |
| **ImageField** | **ClearableFileInput** | **None** |

**Django ModelForm**

ModelForm is a class which is used to create an HTML form by using the Model.

Django automatically does it for us to reduce the application development time.

For example, suppose we have a model containing various fields, we don't need to repeat the fields in the form file.

For this reason, Django provides a helper class which allows us to create a Form class from a Django model.

**models.py**

from django.db import models  
class Employee(models.Model):  
 name = models.CharField(max\_length=50)  
 age = models.IntegerField()  
 contactno = models.IntegerField(primary\_key=True)

**forms.py**

from django import forms  
from appmf.models import Employee  
class FormEmployee(forms.ModelForm):  
 class Meta:  
 model = Employee  
 fields = ["name","contactno"]

**views.py**

from django.shortcuts import render  
from appmf.forms import FormEmployee  
def showIndex(request):  
 fe = FormEmployee()  
 return render(request,"index.html",{"f":fe})

**index.html**

<form action="//" method="post">  
 {% csrf\_token %}  
 {{ f.as\_p }}  
 <input type="submit" value="submit">  
</form>

Commonly used fields and their details are given in the below table.

|  |  |  |
| --- | --- | --- |
| **Name** | **HTML Input** | **Empty value** |
| BooleanField | CheckboxInput | False |
| CharField | TextInput | Whatever you've given as empty\_value. |
| ChoiceField | Select | '' (an empty string) |
| DateField | DateInput | None |
| DateTimeField | DateTimeInput | None |
| DecimalField | NumberInput | None |
| EmailField | EmailInput | '' (an empty string) |
| FileField | ClearableFileInput | None |
| ImageField | ClearableFileInput | None |

There are other output options though for the **<label>**/**<input>** pairs:

* **{{ form.as\_table }}** will render them as table cells wrapped in **<tr>** tags
* **{{ form.as\_p }}** will render them wrapped in **<p>** tags
* **{{ form.as\_ul }}** will render them wrapped in **<li>** tags

Note that you’ll have to provide the surrounding **<table>** or **<ul>** elements yourself.

For Password Input

password = forms.CharField(widget=forms.PasswordInput)

For Number Input

contact = forms.CharField(widget=forms.NumberInput)

For Checkbox Input

select = forms.CharField(widget=forms.CheckboxInput)