# Django

## Database setup

* Open terminal
* Go to project directory

1. *python manage.py migrate*

## Models

* Open music folder
* Open models.py

Code

from django.db import models

# Create your models here.

class Album(models.Model):

artist = models.CharField(max\_length=250)

album\_title = models.CharField(max\_length=500)

genre = models.CharField(max\_length=100)

album\_logo = models.CharField(max\_length=1000)

class Song(models.Model):

album = models.ForeignKey(Album, on\_delete=models.CASCADE)

file\_type = models.CharField(max\_length=10)

song\_title = models.CharField(max\_length=250)

* Open website/settings.py
* Include following code inside INSTALLED\_APPS

'music.apps.MusicConfig',

* Code

INSTALLED\_APPS = [

'music.apps.MusicConfig',

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

]

* Open terminal
* Goto project directory

1. *python manage.py makemigrations music*
2. *python manage.py sqlmigrate music 0001*
3. *python manage.py migrate*
4. *python manage.py runserver (Restart server)*

## Django Database API

* Open cmd
* Go to project directory

1. python manage.py shell

(python36) C:\Users\abhijith.m\Desktop\website>python manage.py shell

Python 3.6.8 |Anaconda, Inc.| (default, Feb 21 2019, 18:30:04) [MSC v.1916 64 bit (AMD64)]

Type 'copyright', 'credits' or 'license' for more information

IPython 7.6.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: from music.models import Album, Song

In [2]: Album.objects.all()

Out[2]: <QuerySet []>

In [3]: a = Album(artist="Taylor swift", album\_title="Red", genre="Country", album\_logo="https://previews.123rf.com/ima

...: ges/queezzard/queezzard1601/queezzard160100009/50339428-equalizer-icon-can-be-used-as-logo-to-music-album-dj-se

...: t-concert-banner-vector-illustration-.jpg")

In [4]: a.save()

In [5]: a.artist

Out[5]: 'Taylor swift'

In [6]: a.album\_title

Out[6]: 'Red'

In [7]: a.id

Out[7]: 1

In [8]: a.pk

Out[8]: 1

In [9]: b = Album()

In [10]: b.artist = "Myth"

In [11]: b.album\_title = "High School"

In [12]: b.genre = "Punk"

In [13]: b.album\_logo = "https://www.kidsdiscover.com/wp-content/uploads/2014/06/Kid\_Smiling-e1401721146279.

...: jpg"

In [14]: b.save()

In [15]: b.artist

Out[15]: 'Myth'

1. When check all objects in album. It will display like this

In [16]: Album.objects.all()

Out[16]: <QuerySet [<Album: Album object (1)>, <Album: Album object (2)>]>

* Open music/models.py
* Include this function inside Artist function

def \_\_str\_\_(self):

return self.album\_title + ' - ' +self.artist

* Full code

class Album(models.Model):

artist = models.CharField(max\_length=250)

album\_title = models.CharField(max\_length=500)

genre = models.CharField(max\_length=100)

album\_logo = models.CharField(max\_length=1000)

def \_\_str\_\_(self):

return self.album\_title + ' - ' +self.artist

1. *exit() # exit from shell*
2. *python manage.py shell*

In [1]: from music.models import Album, Song

In [2]: Album.objects.all()

Out[2]: <QuerySet [<Album: Red - Taylor swift>, <Album: High School - Myth>]>

Filter

In [3]: Album.objects.filter(id=1)

Out[3]: <QuerySet [<Album: Red - Taylor swift>]>

In [5]: Album.objects.filter(artist\_\_startswith='Taylor')

Out[5]: <QuerySet [<Album: Red - Taylor swift>]>

## Admin

* Create admin

(python36) C:\Users\abhijith.m\Desktop\website>python manage.py createsuperuser

Username (leave blank to use 'abhijith.m'): admin

Email address: admin@example.com

Password:

Password (again):

Superuser created successfully.

* Open browser
* Go to: <http://127.0.0.1:8000/admin/>
* Enter username and password
* Open music/admin.py

from django.contrib import admin

from .models import Album

admin.site.register(Album)

* Again Go to: <http://127.0.0.1:8000/admin/>

## Writing another view

* Open music/urls.py

### Connecting to db

* In music/views.py

*from .models import Album*

from .models import Album

def index(request):

#connecting to db

all\_albums = Album.objects.all()

html = ''

for album in all\_albums:

url = '/music/' + str(album.id) + '/'

html += '<a href="'+ url +'">' + album.album\_title + '</a><br>'

return HttpResponse(html)

## Templates

* Open music/views.py
* Create a folder ‘templates’ inside music: (music/template)
* Inside templates create another folder music. (music/templates/music)
* (music/templates/music) > create file index.html
* Index.html

<ul>

<li>

<a href="#">Album title</a>

</li>

</ul>

* (music/views.py) >

from django.http import HttpResponse

from .models import Album

from django.template import loader

def index(request):

#connecting to db

all\_albums = Album.objects.all()

template = loader.get\_template('music/index.html')

context = {

'all\_albums': all\_albums,

}

return HttpResponse(template.render(context, request))

* Update index.html
* (music/templates/music/index.html)>

{% if all\_albums %}

<ul>

{% for album in all\_albums %}

<li>

<a href="/music/{{ album.id }}/">{{ album.album\_title }}</a>

</li>

{% endfor %}

</ul>

{% else %}

<h3>You don't have any albums </h3>

{% endif %}

### Render template shortcut

* (music/views.py)>
  1. Delete

from django.template import loader

* code

from django.http import HttpResponse

from .models import Album

from django.shortcuts import render

def index(request):

#connecting to db

all\_albums = Album.objects.all()

context = {

'all\_albums': all\_albums,

}

return render(request, 'music/index.html', context)

## Raising 403 Error

* (music/templates/music/) > Create a deatil.html file
* Code

{{album}}

* (music/views.py)>
* Code

from .models import Album

from django.shortcuts import render

from django.http import Http404

def detail(request, album\_id):

try:

album = Album.objects.get(pk=album\_id)

except Album.DoesnotExist:

raise Http404("Album does not exist")

return render(request, 'music/detail.html', {'album':album})

## Adding songs to db

* (music/admin.py)

from django.contrib import admin

from .models import Album, Song

admin.site.register(Album)

admin.site.register(Song)

* (music/models.py) >

class Song(models.Model):

album = models.ForeignKey(Album, on\_delete=models.CASCADE)

file\_type = models.CharField(max\_length=10)

song\_title = models.CharField(max\_length=250)

def \_\_str\_\_(self):

return self.song\_title

* Open terminal

(python36) C:\Users\abhijith.m\Desktop\website>python manage.py shell

Python 3.6.8 |Anaconda, Inc.| (default, Feb 21 2019, 18:30:04) [MSC v.1916 64 bit (AMD64)]

Type 'copyright', 'credits' or 'license' for more information

IPython 7.6.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: from music.models import Album, Song

In [3]: album1 = Album.objects.get(pk=1)

In [4]: album1.artist

Out[4]: 'Taylor swift'

In [5]: song = Song()

In [6]: song.album = album1

In [7]: song.file\_type = 'mp3'

In [8]: song.song\_title = 'I hate my boyfriend'

In [9]: song.save()

* Open browser
* <http://127.0.0.1:8000/admin/music/song/>

In [10]: album1.song\_set.all()

Out[10]: <QuerySet [<Song: I hate my boyfriend>]>

In [11]: album1.song\_set.create(song\_title='I love bacon', file\_type='mp3')

Out[11]: <Song: I love bacon>

In [12]: album1.song\_set.create(song\_title='Bucky is lucky', file\_type='mp3')

Out[12]: <Song: Bucky is lucky>

In [13]: album1.song\_set.create(song\_title='Ice cream', file\_type='mp3')

Out[13]: <Song: Ice cream>

In [14]: song = album1.song\_set.create(song\_title='Ice cream', file\_type='mp3')

In [15]: song.song\_title

Out[15]: 'Ice cream'

In [16]: album1.song\_set.count()

Out[16]: 5

## Design detail template

* (music/templates/music/deatil.html)>

<img src="{{ album.album\_logo }}">

<h1>{{ album.album\_title }}</h1>

<h3> {{ album.artist }} </h3>

<ul>

{% for song in album.song\_set.all %}

<li> {{ song.song\_title }} - {{ song.file\_type }} </li>

{% endfor %}

</ul>

## Removing hardcoded URL

* (music/templates/music/index.html)> update the url like this

<a href="{% url 'detail' album.id %}">{{ album.album\_title }}</a>

## Namespace and HTTP 404 shortcut

404

* (music/views.py)> only need to import the following library. Delete other libraries

from .models import Album

from django.shortcuts import render, get\_object\_or\_404

* Update

def detail(request, album\_id):

# album = Album.objects.get(pk=album\_id)

album = get\_object\_or\_404(Album, pk=album\_id)

return render(request, 'music/detail.html', {'album':album})

## Simple Form

* (music/models.py) Add the following line to song model

is\_favorite = models.BooleanField(default=False)

* Full code

class Song(models.Model):

album = models.ForeignKey(Album, on\_delete=models.CASCADE)

file\_type = models.CharField(max\_length=10)

song\_title = models.CharField(max\_length=250)

is\_favorite = models.BooleanField(default=False)

def \_\_str\_\_(self):

return self.song\_title

* Open cmd
* *python manage.py makemigrations music*
* *python manage.py migrate*
* *python manage.py runserver (Restart server)*
* (music/urls.py)> include the path inside ‘urlpatterns’

path('<int:question\_id>/favorite/', views.favorite, name='favorite'),

* (music/templates/music/details.html) >

<img src="{{ album.album\_logo }}">

<h1>{{ album.album\_title }}</h1>

<h3> {{ album.artist }} </h3>

{% if error\_message %}

<p><strong>{{ error\_message }}</strong></p>

{% endif %}

<form action="{% url 'favorite' album.id %}" method="POST">

{% csrf\_token %}

{% for song in album.song\_set.all %}

<input type="radio" id="song{{ forloop.counter }}" name="song" value="{{ song.id }}">

<label for="song{{ forloop.counter }}">

{{ song.song\_title }}

{% if song.is\_favorite %}

<img href="http://icons.iconarchive.com/icons/icons-land/vista-elements/256/Favorites-icon.png"

width="15" />

{% endif %}

</label><br>

{% endfor %}

<input type="submit" value="Favorite" >

</form>

* (music/views.py)> import ‘Song’

from .models import Album, Song

* Create a new method
* (music/views.py) >

def favorite(request, album\_id):

album = get\_object\_or\_404(Album, pk=album\_id)

try:

selected\_song = album.song\_set.get(pk=request.POST['song'])

except (KeyError, Song.DoesNotExist):

return render(request, 'music/detail.html', {

'album':album,

'error\_message': "You did not select a valid song",

})

else:

selected\_song.is\_favorite = True

selected\_song.save()

return render(request, 'music/detail.html', {'album':album})

## Bootstrap And static frames

* (music)> Create folder ‘static’ inside music
* (music/static) create folder ‘music’
* (music/static/music) create folder ‘images’
* (music/static/music/images) > inside this folder save a ‘background.png’ image file
* (music/static/music/) > create ‘style.css’
* Code style.css

body{

background: white url("images/background.png");

}

.navbar{

border-radius: 0;

}

.navbar-brand{

font-family: 'sans-serif', cursive;

}

* Code index.html

{% load static %}

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

<!-- Latest compiled and minified CSS -->

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" crossorigin="anonymous">

<!-- Latest compiled and minified JavaScript -->

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js" integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa" crossorigin="anonymous"></script>

<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>

<nav class="navbar navbar-inverse">

<div class="container-fluid">

<!--logo-->

<div class="navbar-header">

<button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#topNavbar">

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

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

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

</button>

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

</div>

<!-- Items -->

<div class="collapse navbar-collapse">

<ul class="nav navbar-nav">

<li class="active">

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

<span class="glyphicon glyphicon-cd" aria-hidden="true"></span>&nbsp;

Albums

</a>

</li>

<li class="">

<a href="#">

<span class="glyphicon glyphicon-music" aria-hidden="true"></span>&nbsp;

Songs

</a>

</li>

</ul>

<form class="navbar-form navbar-left" role="search" method="GET" action="#">

<div class="form-group">

<input type="text" class="form-control" name="q" value="">

</div>

<button type="submit" class="btn btn-default">Search</button>

</form>

<ul class="nav navbar-nav navbar-right">

<li class="">

<a href="#">

<span class="glyphicon glyphicon-plus" aria-hidden="true"></span>&nbsp;

Add Album

</a>

</li>

<li class="">

<a href="#">

<span class="glyphicon glyphicon-off" aria-hidden="true"></span>&nbsp;

Logout

</a>

</li>

</ul>

</div>

</div>

</nav>

## Create a base template

* (music/templates/music) > create a new file ‘base.html’ inside music

<!DOCTYPE html>

<html>

<head>…

<title>{% block title %} {% endblock %}</title>

</head>

<body>

<nav class="navbar navbar-inverse">…

</nav>

{% block body %}

{% endblock %}

</body>

</html>

* (music/templates/music/index.html) >

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

{% block title %} Album title {% endblock %}

{% block body %}

{% if all\_albums %}…

{% endif %}

{% endblock %}

* (music/templates/music/details.html) > also like index.html

## Generic view

* (music/views.py)
* Delete all things in views.py

## generic view is not working…

## Model Form

* (music/models.py)>

from django.core.urlresolvers import reverse

* Add inside Album

def get\_absolute\_url(self):

return reverse('music:detail', kwargs={'pk': self.pk})

* Album

# Create your models here.

class Album(models.Model):

artist = models.CharField(max\_length=250)

album\_title = models.CharField(max\_length=500)

genre = models.CharField(max\_length=100)

album\_logo = models.CharField(max\_length=1000)

def get\_absolute\_url(self):

return reverse('music:detail', kwargs={'pk': self.pk})

def \_\_str\_\_(self):

return self.album\_title + ' - ' +self.artist

views.py

from django.http import Http404

from .models import Album, Song

from django.shortcuts import render

def index(request):

#connecting to db

all\_albums = Album.objects.all()

context = {

'all\_albums': all\_albums,

}

return render(request, 'music/index.html', context)

def favorite(request, pk):

try:

album = Album.objects.get(pk=pk)

except Album.DoesnotExist:

raise Http404("Album does not exist")

try:

selected\_song = album.song\_set.get(pk=request.POST['song'])

except (KeyError, Song.DoesNotExist):

return render(request, 'music/detail.html', {

'album':album,

'error\_message': "You did not select a valid song",

})

else:

selected\_song.is\_favorite = True

selected\_song.save()

return render(request, 'music/detail.html', {'album':album})

def detail(request, pk):

try:

album = Album.objects.get(pk=pk)

except Album.DoesnotExist:

raise Http404("Album does not exist")

return render(request, 'music/detail.html', {'album':album})

## Rest API

* In cmd
  1. *django-admin startproject api\_example*
  2. *cd api\_example*
  3. *python manage.py migrate*
  4. *python manage.py createsuperuser*

*Username (leave blank to use 'abhijith.m'): admin*

*Email address: admin@example.com*

*Password:*

*Password (again):*

*Superuser created successfully.*

1. python manage.py startapp languages

* (api\_example/settings.py)>
  1. Insert the following code inside ‘INSTALLED\_APPS’

'rest\_framework',

'languages'

* (api\_example/urls.py)> include the following code

from django.urls import path, include

* (api\_example/urls.py)> include the following code inside ‘urlpatterns’

path('', include('languages.urls'))

* (api\_example/languages)Create another file ‘urls.py’ inside ‘languages’ folder

from django.urls import path, include

urlpatterns = [ ]

* (api\_example/languages/models.py)>

from django.db import models

class Language(models.Model):

name = models.CharField(max\_length=50)

paradigm = models.CharField(max\_length=50)

* In cmd
  1. *python manage.py makemigrations*
  2. *python manage.py migrate*
  3. *python manage.py runserver*
* Open browser <http://127.0.0.1:8000/admin/>
* (api\_example/languages/admin.py)>

from .models import Language

admin.site.register(Language)

* Refresh browser <http://127.0.0.1:8000/admin/>
* (api\_example/languages/) create a file ‘serializers.py’

from rest\_framework import serializers

from .models import Language

class LanguageSerializer(serializers.ModelSerializer):

class Meta:

model = Language

fields = ('id', 'name', 'paradigm')

* (api\_example/languages/views.py)

from django.shortcuts import render

from rest\_framework import viewsets

from .models import Language

from .serializers import LanguageSerializer

class LanguageView(viewsets.ModelViewSet):

queryset = Language.objects.all()

serializer\_class = LanguageSerializer

* (api\_example/languages/urls.py)

from django.urls import path, include

from . import views

from rest\_framework import routers

router = routers.DefaultRouter()

router.register('languages', views.LanguageView)

urlpatterns = [

path("", include(router.urls))

]

* In cmd Restart server
  1. *python manage.py runserver*
* open browser <http://127.0.0.1:8000/>
* add language (java:OOPs, C: procedural)
* (api\_example/languages/manage.py)>

from django.db import models

class Language(models.Model):

name = models.CharField(max\_length=50)

paradigm = models.CharField(max\_length=50)

def \_\_str\_\_(self):

return self.name

* open <http://127.0.0.1:8000/admin/>

## Authentication

* Start new project
  1. *django-admin startproject authenticate*
  2. *cd authenticate*
  3. *python manage.py startapp user\_example*
  4. *python manage.py migrate*
  5. *python manage.py createsuperuser*

*Username (leave blank to use 'abhijith.m'): admin*

*Email address: admin@example.com*

*Password:*

*Password (again):*

*Superuser created successfully.*

* (authenticate/urls.py)> inside ‘urlpatterns’

path('admin/', admin.site.urls),

path('', include('user\_example.urls')),

path('accounts/', include('django.contrib.auth.urls'))

* (authenticate/settings.py)> inside ‘INSTALLED\_APPS’

'user\_example',

* (user\_example/)> Create another file ‘urls.py’

from django.urls import path

from . import views

urlpatterns = [

path('', views.index, name = 'index'),

]

* (user\_example/views.py)>

from django.shortcuts import render

def index(request):

return render(request, 'user\_example/index.html')

* (user\_example/templates/user\_example)> create another file ‘index.html’

<!DOCTYPE html>

<html>

<head>

</head>

<body>

<h1>This is my index page</h1>

</body>

</html>

* (user\_example/templates) create folder registration
* (user\_example/templates/registration) create file ‘login.html’

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

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

<title>Document</title>

</head>

<body>

{% if form.errors %}

<p>There's something wrong with what you entered!</p>

{% endif %}

{% if next %}

<p>Hey, you can't access that page.</p>

{% endif %}

<form action="{% url 'login' %}" method="post">

{% csrf\_token %}

<p> Username: {{ form.username }} </p>

<p> Password: {{ form.password }} </p>

<input type="submit" value="login">

<input type="hidden" name= value="{{ next }}">

</form>

</body>

</html>

* (authenticate/settings.py)> add ‘LOGIN\_REDIRECT\_URL’ end of the file

STATIC\_URL = '/static/'

LOGIN\_REDIRECT\_URL = '/'

* In browser <http://127.0.0.1:8000/accounts/login/> login with (username:admin password:\*\*\*\*\*)
* (user\_example/views.py)> add

from django.contrib.auth.forms import UserCreationForm

def register(request):

form = UserCreationForm()

context = {'form' : form}

return render(request, 'registration/register.html', context)

* (userexample/templates/registration/)> create file ‘’registration.html’’

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

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

<title>Register</title>

</head>

<body>

<form method="POST" action="{% url 'register' %}"></form>

{% csrf\_token %}

{% if form.errors %}

<p>There are errors in the form!</p>

{% endif %}

{{ form }}

<input type="submit" value="Register">

</body>

</html>

* (user\_example/urls.py)> add path

path('register', views.register, name = 'register'),

* In browser <http://127.0.0.1:8000/register>
* (user\_example/views.py)> complete code

from django.shortcuts import render, redirect

def index(request):

return render(request, 'user\_example/index.html')

from django.contrib.auth import authenticate, login

from django.contrib.auth.forms import UserCreationForm

def register(request):

if request.method == 'POST':

form = UserCreationForm(request.POST)

if form.is\_valid():

form.save()

username = form.cleaned\_data['username']

password = form.cleaned\_data['password1']

user = authenticate(username= username, password=password)

login(request, user)

return redirect('index')

context = {'form' : form}

return render(request, 'registration/register.html', context)

* (user\_example/templates/user\_example/index.html)> update

<!DOCTYPE html>

<html>

<head>

</head>

<body>

<h1>This is my index page</h1>

{% if user.is\_authenticated %}

<h2>Your name is: {{ user.username }}</h2>

{% else %}

<h2>You are not logged in.</h2>

{% endif %}

</body>

</html>

# Form

## HTML Form

* (testapp/views.py)>

def login\_page(request):

return render(request, "auth/login.html", {})

* (testapp/templates/auth/login.html)>

<form method="POST" >

{% csrf\_token %}

<input type="text" placeholder="Name" name="fullname">

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

</form>

* (urls.py)>

path('testapp/',views.login\_page),

* Update (views.py)>

def login\_page(request):

if request.method == "POST":

print(request.POST)

print(request.POST.get('fullname'))

return render(request, "auth/login.html", {})

* In browser <http://127.0.0.1:8000/testapp/>

## Django Form

* Create file (testapp/forms.py)

from django import forms

class ContactForm(forms.Form):

fullname = forms.Charfield()

email = forms.EmailField()

* In (views.py)>

from .forms import ContactForm

def login\_page(request):

contact\_form = ContactForm()

if request.method == "POST":

print(request.POST)

print(request.POST.get('fullname'))

context = {

'form':contact\_form,

}

return render(request, "auth/login.html", context)

* In (templates/auht/login.html)>

<form method="POST">

{% csrf\_token %}

{{ form }}

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

</form>

* In browser <http://127.0.0.1:8000/testapp/>

## Validation in email

* (Testapp/forms.py)>

from django import forms

class ContactForm(forms.Form):

fullname = forms.CharField(

widget=forms.TextInput(

attrs={

"id":"form\_full\_name",

"placeholder":"Your full name"

}

)

)

#fullname = forms.CharField(widget=forms.TextInput(attrs={"class":"form-control","id":"form\_full\_name", "placeholder":"Your full name"}))

email = forms.EmailField()

def clean\_email(self):

email = self.cleaned\_data.get("email")

if not "gmail.com" in email:

raise forms.ValidationError("Email has to be gmail.com")

return email

* (testapp/views.py)>

def login\_page(request):

contact\_form = ContactForm(request.POST or None)

if contact\_form.is\_valid():

print(contact\_form.cleaned\_data)

# if request.method == "POST":

# print(request.POST)

# print(request.POST.get('fullname'))

context = {

'form':contact\_form,

}

return render(request, "auth/login.html", context)

* (testapp/templates/auth/login.html)>

<form method="POST">

{% csrf\_token %}

{{ form }}

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

</form>

* In browser <http://127.0.0.1:8000/testapp/>
* Check email validation with and without ‘gmail.com’

## Login page

* (testapp/forms.py)>

class LoginForm(forms.Form):

username = forms.CharField()

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

* (testapp/views.py)>

from .forms import ContactForm, LoginForm

def login\_page(request):

login\_form = LoginForm(request.POST or None)

print(request.user.is\_authenticated)

if login\_form.is\_valid():

print(login\_form.cleaned\_data)

context = {

'form':login\_form,

}

return render(request, "auth/login.html", context)

* Create (templates/auth/login.html)>

<form method="POST">

{% csrf\_token %}

{{ form }}

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

</form>

* In browser <http://127.0.0.1:8000/login/>
* In (views.py)>

from django.contrib.auth import authenticate, login

from django.shortcuts import render, redirect

def login\_page(request):

login\_form = LoginForm(request.POST or None)

context = {

'form':login\_form,

}

print("\*" \* 10)

print(request.user.is\_authenticated)

if login\_form.is\_valid():

print(login\_form.cleaned\_data)

username = login\_form.cleaned\_data.get("username")

password = login\_form.cleaned\_data.get("password")

user = authenticate(request, username=username, password=password)

print(user)

if user is not None:

login(request, user)

#context['form'] = LoginForm()

return redirect("/login")

else:

print("Error")

return render(request, "auth/login.html", context)

* In cmd
  + python manage.py createsuperuser
* in browser <http://127.0.0.1:8000/login/>

## Check authenticated

* in (urls.py)>

path('home/',views.home\_page),

* (views.py>

def home\_page(request):

# if not request.user.is\_authenticated():

# return Login

print(request.session.get("first\_name", "unknown"))

print(request.session.get("user", "unknown"))

context = {}

if request.user.is\_authenticated:

context["premium\_content"] = "User logged in"

return render(request, "testapp/home.html",context)

* (testapp/templates/testapp/home.html)>

<h1>Cart</h1>

{% if request.user.is\_authenticated%}

<h1>Premium</h1>

{{ premium\_content }}

{% endif %}

* In browser <http://127.0.0.1:8000/home/>

## Register Form

* (forms.py)>

class RegisterForm(forms.Form):

username = forms.CharField()

email = forms.EmailField()

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

password2 = forms.CharField(label='Confirm password' , widget=forms.PasswordInput())

* In (views.py)>

from .forms import ContactForm, LoginForm, RegisterForm

def register\_page(request):

register\_form = RegisterForm(request.POST or None)

context = {

'form':register\_form,

}

if register\_form.is\_valid():

print(register\_form.cleaned\_data)

return render(request, "auth/register.html", context)

* Create file (testapp/templates/auth/register.html)>

<form method="POST">

{% csrf\_token %}

{{ form }}

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

</form>

* (urls.py)>

path('register/',views.register\_page),

* (forms.py)>

class RegisterForm(forms.Form):

username = forms.CharField()

email = forms.EmailField()

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

password2 = forms.CharField(label='Confirm password' , widget=forms.PasswordInput())

def clean(self):

data = self.cleaned\_data

password = self.cleaned\_data.get('password')

password2 = self.cleaned\_data.get('password2')

if password2 != password:

raise forms.ValidationError("password must match")

return data

* (views.py)>

from django.contrib.auth import authenticate, login, get\_user\_model

User = get\_user\_model()

def register\_page(request):

register\_form = RegisterForm(request.POST or None)

context = {

'form':register\_form,

}

if register\_form.is\_valid():

print(register\_form.cleaned\_data)

username = register\_form.cleaned\_data.get("username")

email = register\_form.cleaned\_data.get("email")

password = register\_form.cleaned\_data.get("password")

new\_user = User.objects.create\_user(username, email, password)

print(new\_user)

return render(request, "auth/register.html", context)

* (forms.py)>

from django.contrib.auth import get\_user\_model

User = get\_user\_model()

class RegisterForm(forms.Form):

username = forms.CharField()

email = forms.EmailField()

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

password2 = forms.CharField(label='Confirm password' , widget=forms.PasswordInput())

def clean\_username(self):

username = self.cleaned\_data.get("username")

qs = User.objects.filter(username = username)

if qs.exists():

raise forms.ValidationError("Username is taken")

return username

def clean(self):

data = self.cleaned\_data

password = self.cleaned\_data.get('password')

password2 = self.cleaned\_data.get('password2')

if password2 != password:

raise forms.ValidationError("password must match")

return data

# Session

1. django-admin startproject session
2. cd session
3. python manage.py startapp testapp

* (testapp) views.py >

from django.shortcuts import render

# Create your views here.

def cart\_home(request):

print(request.session)

print(dir(request.session))

return render(request, "testapp/home.html",{})

* Urls.py

from django.contrib import admin

from django.urls import path

from testapp import views

urlpatterns = [

path('admin/', admin.site.urls),

path('testapp/',views.cart\_home),

]

* Settings.py

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

'testapp',

]

* Create folder:- testapp/templates/home.html

<h1>Cart</h1>

## Session key

* Testapp/views.py

from django.shortcuts import render

# Create your views here.

def cart\_home(request):

# print(request.session)

# print(dir(request.session))

key = request.session.session\_key

print(key)

return render(request, "testapp/home.html",{})

* In cmd
  + python manage.py migrate
  + python manage.py runserver
* in (testapp/views.py)>

def cart\_home(request):

# print(request.session)

# print(dir(request.session))

# key = request.session.session\_key

# print(key)

request.session['first\_name'] = "Justin"

return render(request, "testapp/home.html",{})

def home\_page(request):

print(request.session.get("first\_name", "unknown"))

return render(request, "testapp/home.html",{})

* (urls.py)>

urlpatterns = [

path('admin/', admin.site.urls),

path('testapp/',views.cart\_home),

path('testapp1/',views.home\_page),

]

* Refresh browser >
* <http://127.0.0.1:8000/testapp/>
* <http://127.0.0.1:8000/testapp1/>

# Reference

1. <https://docs.djangoproject.com/en/2.2/>