**Install Django**

**>> pip install django**

**You can check django’s version with this command**

**>> django-admin version**

**or**

**>> django-admin –version**

**Virtual Environment**

**>> pip install pipenv**

**>> pipenv shell**

**This command will activatevirtual environment. Then you will need to install django**

**>> pipenv install django**

**Check if it is installed  
>> pip freeze**

**or**

**>> python -m venv env**

**Open bash command line**

**>> source env/Scripts/activate**

**‘env’ can be any name**

**If you want to exit from virtual environment you can run this code**

**>> deactivate**

**Create project**

**>> django-admin startproject myproject**

**‘myproject’ can be any name**

**After creating project we need to create tables that is come with django as default. In order to do this we need to run this command**

**>> python manage.py migrate**

**Note: Go to folder that is manage.py is located**

**Create application**

**Move to project folder**

**>> cd myproject**

**Create new application in myproject folder**

**>> python manage.py startapp myapp**

**‘myapp’ can be any name**

**When we create new app, we need to show it in settings.py**

INSTALLED\_APPS = [

    …

    'myapp.apps.MyappConfig'

    …

]

**or**

INSTALLED\_APPS = [

    …

    'myapp'

    …

]

**First is more proper way.**

**Run server**

**>> python manage.py runserver**

**It will open your project at**

[**http://127.0.0.01:8000**](http://127.0.0.01:8000)

**or**

[**http://localhost:8000**](http://localhost:8000)

**Create model and add to database**

**In models.py create a model**

from django.db import models

from django.core.exceptions import ValidationError

def validate\_positive(value):

    if value <= 0:

        raise ValidationError("Value must be greater than 0")

# Create your models here.

class Tour(models.Model):

    origin\_country = models.CharField(max\_length=64)

    number\_of\_night = models.PositiveIntegerField()

    destination\_country = models.CharField(max\_length=64)

    price = models.DecimalField(max\_digits=7, decimal\_places=3, validators=[validate\_positive])

    def \_\_str\_\_(self) -> str:

        return f"Tour from {self.origin\_country} to {self.destination\_country}"

**In order to add this model to database we need to create migration file. Run this command in order to achieve this goal.**

**>> python manage.py makemigrations**

**This will create new file in migrations folder. Folder name is usually is ‘0001\_initial.py’.**

**To create model in database you need to run this command**

**>> python manage.py migrate**

**This will create model in db**

**Models and Fields (Detailed explanation)**

from django.db import models

# Create your models here.

class Article(models.Model):

    author = models.ForeignKey(

        'auth.User',

        on\_delete=models.CASCADE,

        verbose\_name='AUTHOR')

    title = models.CharField(

        max\_length=100,

        verbose\_name='TITLE')

    content = models.TextField(verbose\_name='CONTENT')

    created\_date = models.DateTimeField(

        auto\_now\_add=True,

        verbose\_name='CREATED DATE')

    def \_\_str\_\_(self) -> str:

        return f"'{self.title}' by {str(self.author)}"

**\* on\_delete=models.CASCADE**

**When we delete user, articles of that user will be automatically deleted.**

**\* auto\_now\_add=True**

**Date will be now as default.**

**\* verbose\_name=‘AUTHOR’**

**Label to inputs**

**Django shell**

**>> python manage.py shell**

**If you want to use interactive shell you need to install ipython**

**>> pipenv install ipython**

**Then you can run above command to enter shell**

**Open shell**

**ORM (Object-relational mapping)**

class Tour(models.Model):

    origin\_country = models.CharField(max\_length=64)

    number\_of\_night = models.PositiveIntegerField()

    destination\_country = models.CharField(max\_length=64)

    price = models.DecimalField(max\_digits=7, decimal\_places=3, validators=[validate\_positive])

**Will use this model**

**Open django shell**

**Add Tour class or model from asiatouragancy app**

**>> from \_\_\_.models import \_\_\_**

**>> from asiatouragancy.models import Tour**

**Create new tour**

**>> to1 = Tour(origin\_country = “Azerbaijan”, number\_of\_night = 10, destination\_country = “Turkey”, price = 2342)**

**Save tour to database**

**>> to1.save()**

class Article(models.Model):

    author = models.ForeignKey(

        'auth.User',

        on\_delete=models.CASCADE,

        verbose\_name='AUTHOR')

    title = models.CharField(

        max\_length=100,

        verbose\_name='TITLE')

    content = models.TextField(verbose\_name='CONTENT')

    created\_date = models.DateTimeField(

        auto\_now\_add=True,

        verbose\_name='CREATED DATE')

**Continue with this model**

**>> from article.models import Article**

**>> from django.contrib.auth.models import User**

**Import models**

**>> user = User(username=“Ali”, password=“123”)**

**Create new user**

**>> user.save()**

**Save that user**

**Note: with this method password is stored without an encryption**

**>> user2 = User(username=“Ali2”)**

**>> user2.set\_password(“123”)**

**>> user2.save()**

**Save with encrypted password**

**>> article = Article.objects.create(title=“Welcome to Django”, content=“Some text”, author=user)**

**Save model without save method**

**Note: You don’t need to equal to a variable**

**>> User.objects.all()**

**Get all users**

**>> User.objects.all().values()**

**Get all users with columns**

**>> User.objects.get(username=“Ali”)**

**Used for to get user with username “Ali”. It can get only one user at a time. If there are more than one user with “Ali” title, then it will raise an error**

**If there is not any user with username “Ali” then Django will raise Article.DoesNotExist exception.**

**>> Article.objects.filter(title\_\_contains=“one”)**

**Get all articles that their titles contain “one”**

**>> Article.objects.filter(title=“Hello”)**

**Get all articles that their titles are equal to “Hello”**

**>> Article.objects.all().delete()**

**Delete all articles**

**>> Article.objects.get(title=“Welcome to Django”).delete()**

**Delete article with title “Welcome to Django”**

**Note: You can equal it to a variable**

**>> article = Article.objects.get(title=“Title 2”)**

**>> article.delete()**

**Admin panel and create superadmin**

**>> python manage.py createsuperuser**

**This will create new superuser**

**In order to show new created app in Admin panel. We need to register it in admin.py**

from django.contrib import admin

from . import models

# Register your models here.

admin.site.register(models.Article)

**If we want to modify admin panel we need to these steps.**

from django.contrib import admin

from . import models

# Register your models here.

# admin.site.register(models.Article) # Default

@admin.register(models.Article)

class ArticleAdmin(admin.ModelAdmin):

    # fields = ["author"]

    list\_display = ["author", "title", "created\_date"]

    list\_display\_links = ["author", "title"]

    list\_filter = ["created\_date"]

    search\_fields = ["title"]

**\* fields = [“author”]**

**Fields that will be shown when we need to create or update**

**\* list\_display = [“author”, “title”, “created\_date”]**

**Fields that will be shown when in admin panel**

**\* list\_display\_links = [“author”, “title”]**

**Fields that is links**

**\* list\_filter = [“created\_date”]**

**Fields that we can filter with**

**\* search\_fields = [“title”]**

**Fields that we can search with**

from django.contrib import admin

from . import models

# Register your models here.

class ArticleAdmin(admin.ModelAdmin):

    # fields = ["author"]

    list\_display = ["author", "title", "created\_date"]

    list\_display\_links = ["author", "title"]

    list\_filter = ["created\_date"]

    search\_fields = ["title"]

admin.site.register(models.Article, ArticleAdmin)

**Without decorator**

**Templates, urls and view**

**First create “templates” folder and “index.html” in it.**

**Folder structure should look like this**

**\* Project\_name**

**\* app\_name**

**\* views.py**

**\* urls.py**

**\* templates**

**\* index.html**

**\* project\_name**

**\* urls.py**

**In “views.py” file**

from django.shortcuts import render

# Create your views here.

def house(request):

    return render(request=request, template\_name='house.html')

**Create “urls.py” in “app\_name” project.**

**In “urls.py” file**

from django.urls import path

from . import views

urlpatterns = [

    path(route='', view=views.house, name='house'),

]

**“urls.py” in “project\_name” folder**

from django.contrib import admin

from django.urls import path, include

urlpatterns = [

    path('admin/', admin.site.urls),

    path('house/', include('article.urls'))

]

**Template inheritance (“extends”, “include”, “url”)**

**Use “extends”**

**Create two html file in ‘templates’ folder. ‘layout.html’ and ‘index.html’**

**In ‘layout.html’ write this**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

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

</head>

<body>

    <div>

        {% block body %}

        <h1>Layout page</h1>

        {% endblock %}

    </div>

</body>

</html>

**In ‘index.html’ write this**

{% extends "layout.html" %}

{% block title %} Main {% endblock %}

{% block body %}

<h1>Main Page</h1>

{% endblock %}

**‘index.html’ takes ‘layout.html’ as blueprint**

**Use “include”**

**First create ‘assets’ folder and ‘navbar.html’ file in it**

**In ‘navbar.html’ write this**

<ul>

    <li><a href="">Main</a></li>

    <li><a href="">About</a></li>

    <li><a href="">Contact us</a></li>

    <li><a href="">Tutorial</a></li>

</ul>

**Include it in ‘layout.html’. In ‘layout.html’ write this**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

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

</head>

<body>

    {% include 'assets/navbar.html' %}

    <div>

        {% block body %}

        <h1>Layout page</h1>

        {% endblock %}

    </div>

</body>

</html>

**Use “url”**

**Create ‘about.html’ file. In ‘about.html’ file write this**

{% extends 'layout.html' %}

{% block title %}About{% endblock %}

{% block body %}

<h1>About page</h1>

{% endblock %}

**In ‘navbar.html’ write this**

<ul>

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

    <li><a href="{% url 'about' %}">About</a></li>

    <li><a href="">Contact us</a></li>

    <li><a href="">Tutorial</a></li>

</ul>

**\* {% url ‘index’ %}**

**\* {% url ‘about’ %}**

**Here ‘index’ and ‘about’ are names of views in ‘urls.py’ file**

from django.urls import path

from . import views

urlpatterns = [

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

    path(route='about/', view=views.about, name='about')

]

**Use “context”**

**In ‘views.py’ file write this**

from django.shortcuts import render

def index(request):

    context = {

        "name": "Ali",

        "surname": "Abasgulizada",

        "age": 20

    }

    return render(request=request, template\_name='index.html',  context=context)

**In ‘index.html’ file write this**

{% extends "layout.html" %}

{% block title %} Main {% endblock %}

{% block body %}

<h1>Main Page</h1>

<p>Hello my name is {{name}} {{surname}} and I am {{age}} years old</p>

{% endblock %}

**We can use ‘name’, ‘surname’ and ‘age’ here.**

**Use “if, elif, else”**

**In ‘index.html’ file write this**

{% extends "layout.html" %}

{% block title %} Main {% endblock %}

{% block body %}

<h1>Main Page</h1>

{% if name == "King" %}

<p>He is king</p>

{% elif name == "Queen" %}

<p>She is queen</p>

{% else %}

<p>I dont know this person</p>

{% endif %}

{% endblock %}

**Use “for”**

**In ‘index.html’ file write this.**

{% extends "layout.html" %}

{% block title %} Main {% endblock %}

{% block body %}

<h1>Main Page</h1>

<ul>

    {% for number in array %}

    <li>{{number}}</li>

    {% endfor %}

</ul>

{% endblock %}

**Use “Dynamic urls”**

**Create ‘detail.html’. We will use it later.**

**In ‘urls.py’ file write this**

from django.urls import path

from . import views

urlpatterns = [

    path(route='detail/<int:id>', view=views.detail, name='detail')

]

**If we write string instead of integer in url (not in ‘url.py’. don’t confuse) it will raise an error**

**In ‘views.py’ file write this**

from django.shortcuts import render

def detail(request, id):

    context = {

        "id": id

    }

    return render(request=request, template\_name='detail.html', context=context)

**In ‘detail.html’ file write this**

{% extends 'layout.html' %}

{% block title %}Detail{% endblock %}

{% block body %}

<h1>Detail</h1>

<p>Id is {{id}}</p>

{% endblock %}

**If you want to use this in ‘a’ tag. Write like this**

{% for article in articles %}

<a href="{% url 'detail' id=article.id %}">{{article.title}}</a>

{% endfor %}

**‘detail’ here is name of view**

**Static folders -> css, js and images**

**Create “static” folder in “app\_name” folder. Also creare “css”, “js” and “images” folders in “static” folder**

**Folder structure should look like this**

**\* Project\_name**

**\* app\_name**

**\* templates**

**\* index.html**

**\* static**

**\* css**

**\* style.css**

**\* js**

**\* style.**

**\* images**

**\* image.jpg**

**\* project\_name**

**\* settings.py**

**First in “settings.py” write this.**

import os

STATICFILES\_DIRS = [

    os.path.join(BASE\_DIR, 'static')

]

**This will show folder’s place to django**

**Then in “index.html” write this**

{% load static %}

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>House</title>

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

</head>

<body>

    <h1>What's up, my old friend.</h1>

    <img src="{% static 'images/image.jpg' %}" alt="image">

</body>

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

</html>

**\* {% load static %}**

**This will load static to html file**

**\* <link rel=“stylesheet” href=“{% static ‘css/style.css’ %}”>**

**\* <img src=“{% static ‘images/image.jpg’ %}” alt= “image”>**

**\* <script src=“{% static ‘js/script.js’ %}”></script>**

**Static folders for Deployment**

**When you want to deploy your web to internet. You need add all templates to one folder. Django offers you to make this process by itself**

**First in ‘settings.py’ file you need write this**

import os

STATIC\_ROOT = os.path.join(BASE\_DIR, 'staticfiles')

**Don’t forget to delete this(If you have)**

STATICFILES\_DIRS = [

    os.path.join(BASE\_DIR, 'static')

]

**Then run this command**

**>> python manage.py collectstatic**

**This will create ‘staticfiles’ folder**

**Forms**

**Register an User**

**First we need to create ‘forms.py’ and ‘register.html’**

**In ‘forms.py’ file write this**

from django import forms

from django.contrib.auth.models import User

class RegisterForm(forms.ModelForm):

    username = forms.CharField(

        max\_length=100,

        label="Please enter your username"

    )

    email = forms.EmailField(

        label="Please enter your email"

    )

    password = forms.CharField(

        max\_length=50,

        label="Please enter your password",

        widget=forms.PasswordInput

    )

    confirm = forms.CharField(

        label="Please confirm your password",

        widget=forms.PasswordInput

    )

    class Meta:

        model = User

        fields = ['username', 'email', 'password']

    def clean(self):

        cleaned\_data = super().clean()

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

        confirm = cleaned\_data.get('confirm')

        if password and confirm and password != confirm:

            raise forms.ValidationError("Passwords don't match")

        return cleaned\_data

    def save(self, commit=True):

        user = super().save(commit=False)

        user.username = self.cleaned\_data['username']

        user.email = self.cleaned\_data['email']

        user.set\_password(self.cleaned\_data['password'])

        if commit:

            try:

                user.save()

            except ValueError as e:

                raise e

        return user

**\* ‘RegisterForm’ class inherit from ‘forms.ModelForm’ class because we use ‘model=User’. We will talk about later.**

**\* class Meta**

**Is used for to do some modifications by ourself**

**\* model = User**

**\* fields = [‘username’, ‘email’, ‘password’]**

**Use fileds of User in our form.**

**\* def clean(self)**

**When we write ‘form.is\_valid()’ in ‘views.py’ file ‘clean’ function will be called**

**\* cleaned\_data = super().clean()**

**Call ‘cleaned\_data’ from parent class. Here parent class is ‘ModelForm’ class**

**\* def save(self, commit=True)**

**Used for save User. Commit is true as default**

**\* user = super().save(commit=False)**

**Call ‘save()’ function from parent class. Here parent class in ‘ModelForm’ class**

**In ‘views.py’ file write this**

from django.shortcuts import render, redirect

from . import forms

from django.contrib.auth import login

from django.contrib import messages

def register(request):

    form = forms.RegisterForm(request.POST or None)

    if form.is\_valid():

        user = form.save()

        login(request, user)

        messages.success(request, "You have registered successfully!")

        return redirect('index')

    context = {

        "form": form

    }

    return render(request=request, template\_name='register.html', context=context)

**\* form = forms.RegisterForm(request.POST or None)**

**If method is ‘POST’ method then form will be**

**\* form = forms.RegisterForm(request.POST)**

**Any other methods form will be**

**\* form = forms.RegisterForm()**

**\* form.is\_valid()**

**This will call ‘clean()’ function from ‘forms.py’ file**

**\* login(request, user)**

**This will login user to website**

**\* messages**

**Go to messages documentation**

**\* redirect(‘index’)**

**This will return view with ‘index’ in urls.py**

from django.urls import path

from . import views

urlpatterns = [

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

]

**In this case page is this view**

**In ‘register.html’ file write this**

{% extends "layout.html" %}

{% block title %}Register{% endblock %}

{% block body %}

<h1>Register</h1>

<form method="post" novalidate>

    {% csrf\_token %}

    {{ form.as\_p }}

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

</form>

{% endblock %}

**\* novalidate**

**It is used for to deactivate default validations of form**

**\* {% csrf\_token %}**

**Is used for protect form from ‘Cross-Site Request Forgery (CSRF)’ attack**

**\* {{ form.as\_p }} -> Is used to show form**

**Django Messages**

**First call ‘messages’ from ‘django.contrib’**

from django.contrib import messages

**Then Use it.**

messages.success(request, "You have registered successfully!")

**\* messages.debug**

**\* messages.info**

**\* messages.success**

**\* messages.warning**

**\* messages.error**

**These are types(tags) of messages**

**If you want it to use in any html file write this. Usually write it in ‘layout.html’**

**In ‘layout.html’ file write this**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

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

</head>

<body>

    <div>

        {% if messages %}

            {% for message in messages %}

            <div {% if message.tags %} class="{{ message.tags }}" {% endif %}>

                {{ message }}

            </div>

            {% endfor %}

        {% endif %}

        {% block body %}

        <h1>Layout page</h1>

        {% endblock %}

    </div>

</body>

</html>

**Login an User**

**Create ‘login.html’ file**

**First in ‘forms.py’ file write this**

from django import forms

class LoginForm(forms.Form):

    username = forms.CharField(label="Please enter your username: ")

    password = forms.CharField(

        label="Please enter your password",

        widget=forms.PasswordInput

    )

**In ‘views.py’ file write this**

from django.shortcuts import render, redirect

from . import forms

from django.contrib.auth import login, authenticate

from django.contrib import messages

def loginUser(request):

    form = forms.LoginForm(request.POST or None)

    if form.is\_valid():

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

        password = form.cleaned\_data.get("password")

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

        if user:

            login(request, user)

            messages.success(request, f"Welcome {username}")

            return redirect('index')

        messages.error(request, "Username of password is invalid")

    context = {

        "form": form

    }

    return render(request=request, template\_name='login.html', context=context)

**\* form.is\_valid()**

**We didn’t create ‘clean’ in ‘LoginForm’ class but we can call ‘is\_valid()’ function. That is because it call that function form ‘Form’ class**

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

**‘authenticate’ function checks that user if exist or not. If user is not exists then ‘authenticate’ function will give ‘None’**

**In ‘login.html’ file write this**

{% extends "layout.html" %}

{% block title %}Register{% endblock %}

{% block body %}

<h1>Login</h1>

<form method="post">

    {% csrf\_token %}

    {{ form.as\_p }}

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

</form>

{% endblock %}

**@login\_required**

**What if we enter a restricted page without authentication. In order to prevent this problem we need use ‘@login\_required’ decorator.**

**First write this in ‘views.py’ file**

from django.contrib.auth.decorators import login\_required

**Then use this**

@login\_required()

def dashboard(request):

    articles = models.Article.objects.filter(author = request.user)

    context = {

        "articles": articles

    }

    return render(request, 'article/dashboard.html', context)

**‘@login\_required’ will prevent to access to restricted page without authentication**

**With this if we try to access restricted page that will give 404 error. If you want to redirect to one particular page you can use this.**

**In ‘settings.py’ file write this**

LOGIN\_URL = '/login/'

**This is where it will go**

**Note: I recommend you to use this**

return redirect(request.POST.get('next') or request.GET.get('next') or 'index')

**Because this will send you to previous page that you want to access. Write this in ‘login’**

def loginUser(request):

    form = forms.LoginForm(request.POST or None)

    if form.is\_valid():

        username = form.cleaned\_data.get('username')

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

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

        if user:

            login(request, user)

            messages.success(request, f"Welcome back {username}")

            return redirect(request.POST.get('next') or request.GET.get('next') or 'index')

        messages.error(request, "Username or Password is invalid")

    context = {

        "form": form

    }

    return render(request, 'user/login.html', context)

**NOTE: ‘instance’**

**When we want to give form that info is in it. Then we use ‘instance’**

article = models.Article.objects.get(id=id)

form = forms.ArticleForm(request.POST or None, instance=article)

**Logout an User**

**In ‘views.py’ file write this**

from django.shortcuts import redirect

from django.contrib.auth import logout

from django.contrib import messages

def logoutUser(request):

    logout(request)

    messages.success(request, "Bye")

    return redirect('index')

**\* logout(request)**

**Is used for to logout user**

**Show links based on authentication**

**In ‘navbar.html’ file write this**

<ul>

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

    <li><a href="{% url 'about' %}">About</a></li>

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

    <li><a href="">Control Panel</a></li>

    <li><a href="{% url 'logout' %}">Logout</a></li>

    {% else %}

    <li><a href="{% url 'login' %}">Login</a></li>

    <li><a href="{% url 'register' %}">Register</a></li>

    {% endif %}

</ul>

**In ‘index.html’ file write this**

{% extends "layout.html" %}

{% block title %} Main {% endblock %}

{% block body %}

<h1>Main Page</h1>

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

<p>Salam {{ request.user.username }}</p>

{% else %}

<p>I don’t know you</p>

{% endif %}

{% endblock %}