DJANGO PROJECT TUTORIAL

Learn django

GET STARTED

The tutorial focuses on the initial steps you'll need to start a new web application. To complete it, you'll need to have Python installed and understand how to work with virtual environments .

By the end of the tutorial you will know how to:

- Set up a **virtual environment**
- **Install** Django
- Pin your project **dependencies**
- Set up a Django **project**
- Start a Django **app**

PREPARE YOUR ENVIRONMENT PREPARE YOUR ENVIROMENT

When you're ready to start your new Django web application, create a new folder and navigate into it. In this folder, you'll set up a new virtual environment using your command line:

```
python3_-m_venv_env
```

This command sets up a new virtual environment named `env` in your current working directory. Once the process is complete, you also need to activate the virtual environment:

'name of the virtual env'\Scripts\activate

If the activation is successful, then you'll see the name of your virtual environment, ie `(env)`, at the beginning of your command prompt. This means that your environment setup is complete.

INSTALL DJANGO AND PIN THE DEPENDENCIES

Once you've created and activated your Python virtual environment, you can install Django into this dedicated development workspace:

And to install it you run the following command in the command line :

Pip install django

CREATING A DJANGO APP

You already created your project so, it's time for creating the Django app which should be inside of the project. For app creation you need to run following command in a terminal, while you are inside the project folder. (I am using 'Spare_parts' as my app, you can use whatever you want.):

```
python manage.py startapp spare_parts
```

REGESTERING THE APP

Now you have to register the created app a project. To do that you will have to open my `settings.py` file (which is inside of the project folder). And I scroll down it to `INSTALLED_APPS` section, and add my app name `Spare_parts` inside it (in the single quotes and then i put a commars outside the quotes) and then leave the rest of the code the way is.

INSTALLED APPS[

*

*

*

‘spare_parts’,

]

CREATE STATIC AND TEMPLATES FOLDER

At this point you have to create two very important folders where the the static and templates files will store.

Static Folder will contain all of my static files (like images, videos, text files, CSS etc...).

Templates Folder will contain your dynamic files (like all html necessary files).

In the project folder inside you create `static` and `templates` folders. We need no commands for this since we can simply create them normally .

SETTING DIRECTORIES

After creating the above folders now you go back to the code editor and set their directories. First you have to set static directory.
In `settings.py` file. At very beginning where all other libraries are imported let import `os` module in it by typing...

```
import os
```

scroll `settings.py` down to the end and type following code in it.

```
STATICFILES_DIRS = [

os.path.join(BASE_DIR, 'static')

]
```

SETTING TEMPLATE DIRECTORY

Still in `settings.py` file, find `TEMPLATES` section and look for `'DIRS':[],` and in those empty brackets type the following code inside and leave the other code as it is. You just have to edit

```
TEMPLATES = [{

+

'DIRS': [os.path.join(BASE_DIR,
'templates')],

+

+

+

}]
```

CREATE TEMPLATES

Templates are the web pages like `index.html` , `about.html` or `contact.html`. Though you need to extend the templates on your web pages using `{%extends 'index.html'%}` . And for including styles or scripts in your templates from static folder you have to use following code.

```
<link rel="stylesheet"
href="/static/css/styles.css">
```

```
<script
src="/static/js/scripts.js"></sc
ript>
```

DEFINE AND RENDER URLS

At this point now you have to render the URLs in the `views.py` file by Opening it and writing the following code:

```
from django.shortcuts import render

def index(request):

return render(request, 'index.html', {})
```

CREATING A SUPER USER

Django offers a built-in admin panel but to access it I simply have to create a `superuser` which will help me access every feature of that admin panel.

For creating `superuser` in Django I have to open my terminal and type the following command;

```
python manage.py createsuperuser admin

Email:

Password:

(Re-type) Password:
```

For `Email` you can either do it or not it's optional. And for the password its a must and **Remember**: when you type in a password it won't show it to you so, just type it and hit `Enter` key, and Retype it. After this you might see a message so, just enter `y` and hit `Enter` key. But you must use a password that you shall easily remember because incase you forget it you won't be able to use it any more.

At this point my `superuser` is just created, I can now open my admin panel by visiting http://127.0.0.1:8000/admin/ link. (But first I must run the server with `runserver` command) .

And now if you try to access the Project URL you might run into an error because of not migrating.
So, you will run migrations the project in order
to recognise the changes made.

In a terminal you will run:

```
python manage.py makemigrations
```

```
Python manage.py migrate
```

RUN MIGRATIONS

RUN THE PROJECT

Finally you just finished all the important steps of your Django project setup. So, at the end it's time to run our project and see how it looks using the following command in your terminal...

```
python manage.py runserver
```

If all clears, I will have to visit http://127.0.0.1:8000/ link and open my Django web application. At this point I can visit my admin panel using http://127.0.0.1:8000/admin/