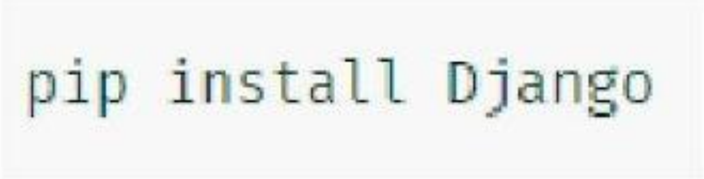
Forward School

Django
Part 2

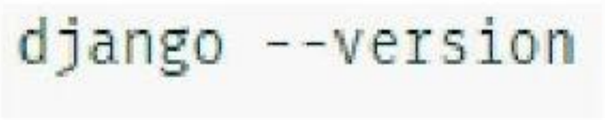
Pre-requisites

1. Install Visual Studio Code
2. Install Python (Should already come with your Anaconda installation)
3. Setup a Database.
   1. MYSQL, PostgreSQL, MongoDB, SQLite3
   2. Let's go with SQLite3 that we installed earlier lessons : *pip install sqlite3*
4. Create new env named *web*
5. Install Django in Anaconda Prompt



```
pip install Django
```

6. Test your Django installation



```
django --version
```

Step 1: Creating your First Django Project

1. django-admin startproject dashboard
2. cd dashboard

Step 2: Creating your First Django App

1. django-admin startapp users

Step 3: Defining User Models

```
from django.contrib.auth.models import User
```

Step 4: Creating User Registration and Login Forms

```
from django import forms
from django.contrib.auth.models import User
from django.contrib.auth.forms import UserCreationForm, AuthenticationForm

class RegisterForm(UserCreationForm):
    class Meta:
        model = User
        fields = ["username", "email", "password1", "password2"]

class LoginForm(AuthenticationForm):
    class Meta:
        fields = ["username", "password"]
```

`UserCreationForm` and `AuthenticationForm` are Django's built-in forms for user registration and login.

Step 5: Creating Views for Registration, Login, and Dashboard

```
from django.shortcuts import render, redirect
from .forms import RegisterForm, LoginForm
from django.contrib.auth import login, authenticate
from django.contrib.auth.decorators import login_required

def register(request):
    if request.method == "POST":
        form = RegisterForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('login')
    else:
        form = RegisterForm()
    return render(request, 'users/register.html', {'form': form})

def user_login(request):
    if request.method == 'POST':
        form = LoginForm(request, data=request.POST)
```

Step 6: Setting Up URLs

```
from django.urls import path
from . import views

urlpatterns = [
    path('register/', views.register, name='register'),
    path('login/', views.user_login, name='login'),
    path('dashboard/', views.dashboard, name='dashboard'),
]
```

Step 7: Creating Templates

Create the following templates in the templates/users directory:

register.html
login.html
dashboard.html

These HTML files should contain the form rendering and other HTML structures you desire.

Step 8: Run the server

```
python manage.py makemigrations
python manage.py migrate
python manage.py runserver
```

python manage.py makemigrations creates migration files that capture changes to your Django models, helping you maintain a record of database schema modifications.

python manage.py migrate applies these migration files to update your database schema, ensuring it aligns with your model changes.

python manage.py runserver starts a local development server, allowing you to test and interact with your Django web application while providing debugging information in the console.

Step 9: Create SuperAdmin

```
python manage.py createsuperuser
```

Step 10: Let’s create another App for Charts

1. django-admin startproject charts