هل يمكن تصميم هذا النظام عن طريق django ?

نعم، يمكن تصميم منصة مثل "منصة التقارير الموحدة" باستخدام **Django**. Django هو إطار عمل قوي لتطوير تطبيقات الويب، ويتمتع بميزات تساعد في تحقيق المتطلبات المطروحة في الوثيقة، مثل إدارة المستخدمين، وتقييد الصلاحيات، وتوفير الواجهات التفاعلية. إليك كيف يمكن استخدام Django لتحقيق النظام:

**1. إدارة المستخدمين والصلاحيات**

* Django يوفر نظامًا متقدمًا لإدارة المستخدمين.
* يمكن إنشاء **مجموعات وصلاحيات** للمستخدمين مثل:
  + مسؤول لكل بلدية.
  + موظفو وكالة التطوير.
  + مديرون لمتابعة الأداء.

**2. إدارة البيانات والجداول**

* استخدام **Django Models** لإنشاء قواعد بيانات ديناميكية تدعم:
  + إنشاء جداول بيانات جديدة.
  + إدخال وتصدير البيانات.
  + تطبيق الفلاتر والبحث في الجداول.

**3. إدارة المهام والمبادرات**

* يمكن تطوير نظام لمتابعة المهام باستخدام:
  + **Task Management Models**: لتعقب تقدم المهام.
  + واجهة تتيح إنشاء المهام الفرعية وربطها بالمهام الرئيسية.
  + إرسال الإشعارات عند التحديث.

**4. الاجتماعات التشغيلية**

* بناء نظام يدير الاجتماعات وملاحظاتها:
  + **Meeting Management**: مع تفاصيل عن الوقت والمسؤوليات.
  + إمكانية عرض المؤقت لكل عرض تشغيلي.

**5. إدارة المؤشرات والبرامج**

* Django يسمح ببناء لوحة قياس مؤشرات مع رسوم بيانية:
  + **Django Chart Libraries** (مثل Plotly أو Matplotlib).
  + عرض التقدم الشهري وربط المؤشرات بالمهام والمبادرات.

**6. الواجهات التفاعلية**

* استخدام **Django Templates** لإنشاء واجهات المستخدم.
* ربط Django بـ **Figma Designs** لتحويل التصاميم إلى واجهات تفاعلية.

**7. الإشعارات والتحديثات**

* يمكن إضافة نظام إشعارات باستخدام:
  + **WebSockets** مع Django Channels.
  + إشعارات فورية عند حدوث تغييرات.

**8. الأمان وحفظ البيانات**

* Django يقدم مصادقة متقدمة لحماية المستخدمين.
* يمكن حفظ البيانات بشكل مشفر باستخدام أدوات مثل **Django Encrypted Fields**.

**المزايا التقنية لـ Django:**

* **مرونة** في التعامل مع مختلف المتطلبات.
* دعم قوي لقواعد البيانات المختلفة مثل PostgreSQL.
* **إدارة المشاريع** بشكل منظم وسريع.
* **توسع سلس** لاستيعاب عدد كبير من المستخدمين.

إذا كنت مهتمًا، يمكنني مساعدتك بوضع تصميم أولي لهيكل النظام باستخدام Django.

------------------------------------------------------------------------------------------------------------------------------------------

اريد منك توفير الخطوات و الكود لبناء هذا النظام

لإنشاء نظام "منصة التقارير الموحدة" باستخدام **Django**، سأوفر لك الخطوات مع الكود المطلوب لكل خطوة. النظام يتكون من عدة مكونات، وسنعمل عليها تدريجيًا:

**1. إعداد بيئة العمل**

1. قم بتثبيت Python وDjango:
2. pip install django
3. أنشئ مشروع Django جديد:
4. django-admin startproject unified\_reports
5. cd unified\_reports
6. أنشئ تطبيقًا داخل المشروع لإدارة الوظائف:
7. python manage.py startapp core
8. أضف التطبيق إلى ملف الإعدادات settings.py:
9. INSTALLED\_APPS = [
10. ...
11. 'core',
12. ]

**2. إنشاء نموذج المستخدم والصلاحيات**

**أ. نموذج المستخدم**

في ملف models.py الخاص بالتطبيق core:

from django.contrib.auth.models import AbstractUser

from django.db import models

class CustomUser(AbstractUser):

ROLE\_CHOICES = [

('admin', 'Admin'),

('municipality', 'Municipality User'),

('agency', 'Agency Employee'),

]

role = models.CharField(max\_length=20, choices=ROLE\_CHOICES)

قم بتحديث ملف settings.py:

AUTH\_USER\_MODEL = 'core.CustomUser'

**ب. إدارة الصلاحيات**

استخدم مجموعات وصلاحيات Django:

python manage.py createsuperuser

**3. إنشاء نماذج البيانات (Tables, Tasks, Reports)**

**أ. نموذج الجداول**

في ملف models.py:

class DataTable(models.Model):

name = models.CharField(max\_length=255)

created\_by = models.ForeignKey(CustomUser, on\_delete=models.CASCADE)

created\_at = models.DateTimeField(auto\_now\_add=True)

**ب. نموذج المهام**

class Task(models.Model):

title = models.CharField(max\_length=255)

description = models.TextField()

assigned\_to = models.ForeignKey(CustomUser, on\_delete=models.CASCADE)

deadline = models.DateTimeField()

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

created\_at = models.DateTimeField(auto\_now\_add=True)

**ج. نموذج التقارير**

class Report(models.Model):

name = models.CharField(max\_length=255)

related\_table = models.ForeignKey(DataTable, on\_delete=models.CASCADE)

created\_at = models.DateTimeField(auto\_now\_add=True)

content = models.TextField()

**4. إعداد واجهات الإدارة**

1. قم بإضافة النماذج إلى لوحة الإدارة: في ملف admin.py:
2. from django.contrib import admin
3. from .models import CustomUser, DataTable, Task, Report
4. admin.site.register(CustomUser)
5. admin.site.register(DataTable)
6. admin.site.register(Task)
7. admin.site.register(Report)
8. قم بتشغيل قاعدة البيانات:
9. python manage.py makemigrations
10. python manage.py migrate

**5. إنشاء واجهات المستخدم**

استخدم **Django Templates** و**Views** لعرض البيانات:

1. أضف ملف views.py:
2. from django.shortcuts import render
3. from .models import DataTable, Task
4. def dashboard(request):
5. tables = DataTable.objects.all()
6. tasks = Task.objects.filter(assigned\_to=request.user)
7. return render(request, 'dashboard.html', {'tables': tables, 'tasks': tasks})
8. أضف عنوان URL في ملف urls.py:
9. from django.urls import path
10. from . import views
11. urlpatterns = [
12. path('', views.dashboard, name='dashboard'),
13. ]
14. أضف قالب HTML بسيطًا:
    * ملف: templates/dashboard.html:
    * <!DOCTYPE html>
    * <html>
    * <head>
    * <title>Dashboard</title>
    * </head>
    * <body>
    * <h1>Welcome, {{ user.username }}</h1>
    * <h2>Tables</h2>
    * <ul>
    * {% for table in tables %}
    * <li>{{ table.name }}</li>
    * {% endfor %}
    * </ul>
    * <h2>Tasks</h2>
    * <ul>
    * {% for task in tasks %}
    * <li>{{ task.title }} - {{ task.deadline }}</li>
    * {% endfor %}
    * </ul>
    * </body>
    * </html>

**6. إضافة واجهات تفاعلية (Forms)**

**أ. نموذج إدخال جدول جديد**

في ملف forms.py:

from django import forms

from .models import DataTable

class DataTableForm(forms.ModelForm):

class Meta:

model = DataTable

fields = ['name']

**ب. عرض النموذج في الواجهة**

في ملف views.py:

from .forms import DataTableForm

def create\_table(request):

if request.method == 'POST':

form = DataTableForm(request.POST)

if form.is\_valid():

form.save()

else:

form = DataTableForm()

return render(request, 'create\_table.html', {'form': form})

**7. الخطوات المتبقية**

1. إضافة **AJAX** لجعل المنصة أكثر ديناميكية.
2. استخدام مكتبات مثل **Chart.js** لعرض المؤشرات.
3. إعداد نظام إشعارات باستخدام **Django Channels**.

إذا كنت بحاجة إلى المزيد من التفاصيل أو الكود لأي جزء من المشروع، سأكون سعيدًا بمساعدتك!