بسم اللّه الرحمن الرحيم

التكليف الخامس

الطالب : اديب محمد قاسم محمد

اشراف الاستاذ: مالك المصنف

مقارنة يوضح أنواع إنشاء الـ Forms في Django (الطرق المختلفة) وما يميز كل واحدة عن الثانية

الطريقة	الوصف	المميزات	العيوب	متى أستخدمها
HTML Form عادي	إنشاء الفورم يدويًا في ملف HTML باستخدام <form>, <input>, <select>> إلخ ...	- تحكم كامل في التصميم - خفيف وبسيط - لا يحتاج إعدادات إضافية	- لا يوجد ربط تلقائي مع الموديل أو الفاليديشن - تحتاج تكتب كود للتحقق (validation) بنفسك	لو عندك فورم بسيط جدًا أو واجهة مخصصة بالكامل
Django Form (forms.Form)	إنشاء كلاس في forms.py يمثل الفورم، وتحديد الحقول يدويًا	- يوفر فاليديشن جاهز - سهل الاستخدام - يمكن إعادة استخدامه في عدة Views	- غير مرتبط مباشرة مع قاعدة البيانات - تحتاج تكتب كل الحقول بنفسك	للفورم اللي مش مرتبط مباشرة مع الموديل (زي فورم تسجيل دخول)
ModelForm (forms.ModelForm)	يعتمد على موديل محدد في models.py وينشئ فورم بشكل تلقائي من الحقول	- يقلل التكرار (DRY) - فاليديشن مرتبط بالموديل - سهل التعديل على الحقول	- أقل مرونة إذا أردت فورم مخصص بالكامل	الأفضل في CRUD (إضافة/تعديل بيانات) لأنها مرتبطة بقاعدة البيانات

Formsets	مجموعة من الـ Forms تتعامل معها دفعة واحدة (مثلاً إضافة عدة Teachers في صفحة واحدة)	- تسهل التعامل مع أكثر من فورم دفعة واحدة - تدعم إضافة/حذف فورمز ديناميكيًا	- أعقد من استخدام Form واحد - يحتاج كتابة View أعقد	لما تحتاج إدخال أو تعديل **قائمة بيانات** مرة واحدة
Third-party Forms (مثل django-crispy-forms)	مكتبات خارجية تساعد على تحسين تصميم الفورم	- تصميم جاهز وأنيق - يقلل وقت كتابة HTML	- يحتاج مكتبة إضافية - قد يقيدك أحيانًا	لما تحتاج واجهة أنيقة بسرعة

جدول البحث في Django ORM

مثال	الوصف	الأمر
Student.objects.filter(name__exact="Ali")	تطابق كامل	field__exact="..."
Student.objects.filter(name__iexact="ali")	تطابق كامل بدون حساسية لحالة الأحرف	field__iexact="..."
Student.objects.filter(name__contains="Ali")	يحتوي النص	field__contains="..."
Student.objects.filter(name__icontains="ali")	يحتوي النص (غير حساس لحالة الأحرف)	field__icontains="..."
Student.objects.filter(name__startswith="A")	يبدأ بـ	field__startswith="..."
Student.objects.filter(name__istartswith="a")	يبدأ بـ (غير حساس لحالة الأحرف)	field__istartswith="..."
Student.objects.filter(name__endswith="n")	ينتهي بـ	field__endswith="..."
Student.objects.filter(name__iendswith="N")	ينتهي بـ (غير حساس)	field__iendswith="..."
Student.objects.filter(name__regex=r'^[A-Z]')	البحث باستخدام Regular Expression	field__regex="..."
Student.objects.filter(name__iregex=r'^[a-z]')	البحث بـ Regex غير حساس للأحرف	field__iregex="..."

مثال

```
# البحث عن الطلاب اللي اسمهم "Ali" بالضبط
```

```
Student.objects.filter(name__exact="Ali")
```

```
# البحث عن الطلاب اللي اسمهم يحتوي كلمة "ah"
Student.objects.filter(name__icontains="ah")
```

```
# البحث عن الطلاب اللي أسماؤهم تبدأ بحرف "M"
Student.objects.filter(name__startswith="M")
```

```
# البحث باستخدام regex (مثلاً أسماء من 3 حروف فقط)
Student.objects.filter(name__regex=r'^\w{3}$')
```

جدول الاستعلامات في Django ORM

الأمر	الوصف	مثال
Model.objects.all()	جلب جميع السجلات	Student.objects.all()
Model.objects.get(pk=1)	جلب سجل واحد فقط (لو ما لقى → خطأ)	Student.objects.get(id=1)
Model.objects.filter()	جلب سجلات بشرط	Student.objects.filter(age=20)
Model.objects.exclude()	استبعاد سجلات بشرط	Student.objects.exclude(age=20)
Model.objects.order_by()	ترتيب النتائج	Student.objects.order_by('name')
Model.objects.order_by('-field')	ترتيب تنازلي	Student.objects.order_by('-age')
Model.objects.first()	أول عنصر	Student.objects.first()
Model.objects.last()	آخر عنصر	Student.objects.last()
Model.objects.count()	عدد السجلات	Student.objects.count()
Model.objects.exists()	التحقق من وجود بيانات	Student.objects.filter(name="Ali").exists()
Model.objects.values()	إرجاع Dictionary بدلاً من object	Student.objects.values('name', 'age')
Model.objects.values_list()	إرجاع Tuple من الحقول	Student.objects.values_list('name', flat=True)
Model.objects.create()	إنشاء سجل جديد وحفظه	Student.objects.create(name="Ali", age=20)
obj.save()	حفظ التعديلات على سجل	student.age = 25; student.save()

student.delete()	حذف سجل	obj.delete()
Student.objects.bulk_create([Student(name="A"), Student(name="B")])	إدخال مجموعة سجلات دفعة واحدة	Model.objects.bulk_create()
Student.objects.aggregate(Avg('age'))	إحصائيات (مثل المجموع والمتوسط)	Model.objects.aggregate()
Student.objects.annotate(age_plus_10=F('age')+10)	إضافة عمود محسوب	Model.objects.annotate()