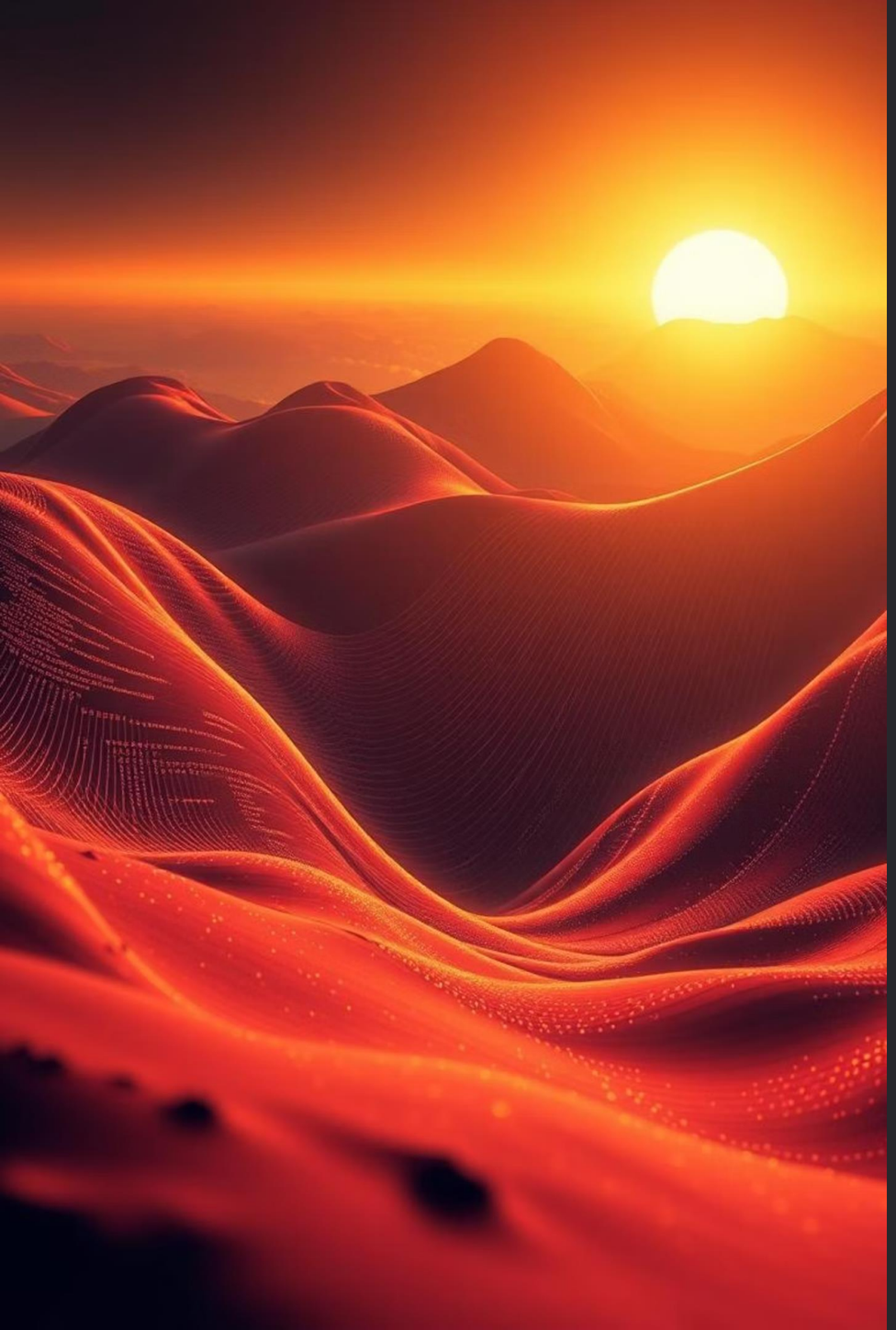
مقدم العرض

م /محمود اغا

الاستاذ م /مالك المصنف

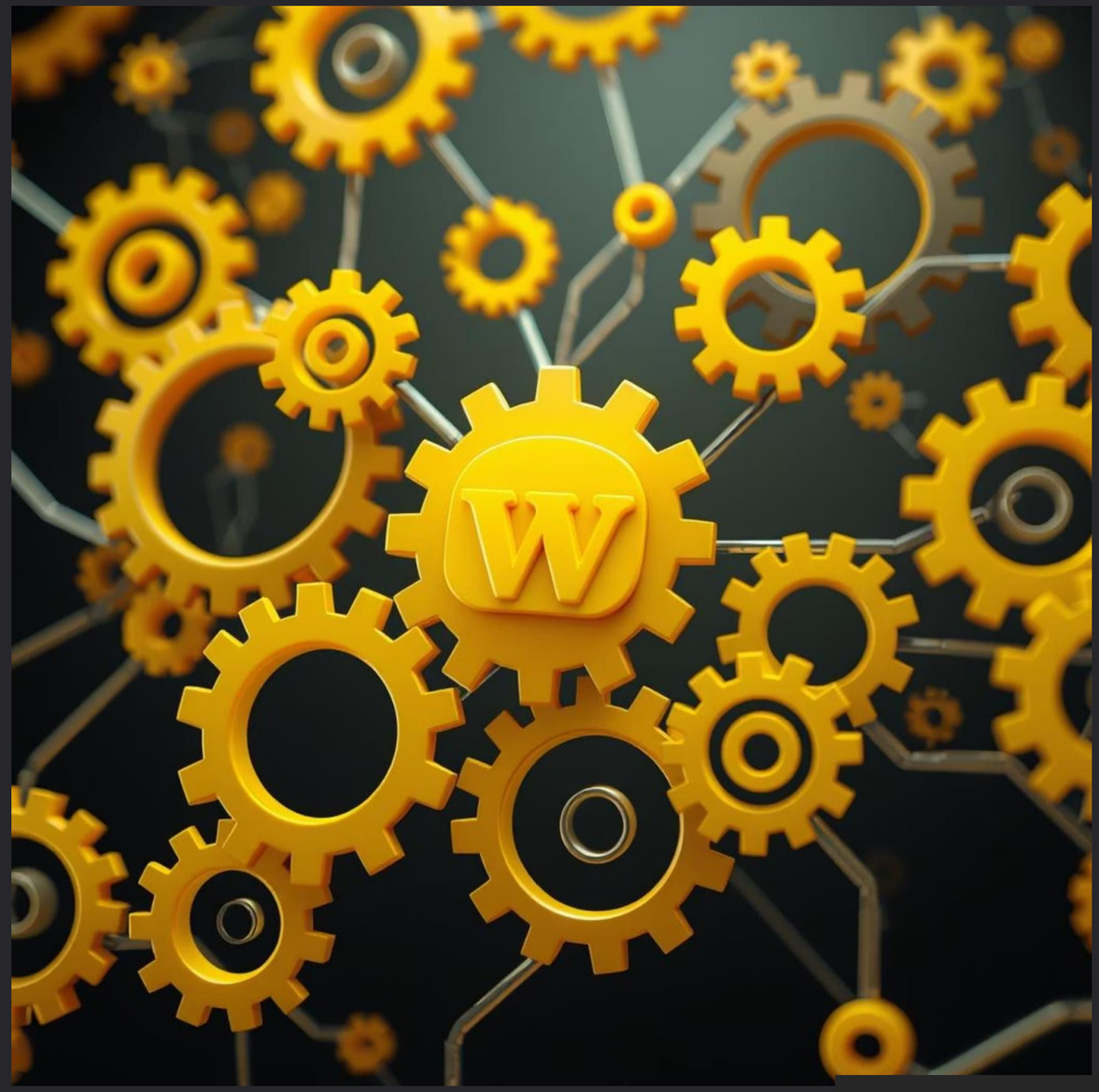
mahmoodaghe@gmail.com



محركات القوالب في Djangoو الفلاتر المخصصة

مرحباً بكم !سيكشف هذا العرض التقديمي عن آليات محركات القوالب في Django، مع التركيز على قوة ومرونة الفلاتر، وكيف يمكنها تحويل واجهات المستخدم الخاصة بكم إلى تجارب ديناميكية وتفاعلية.

فهم محركات القوالب



ما هو محرك القوالب؟

محرك القوالب هو أداة تُمكن المطورين من دمج البيانات الديناميكية مع قوالب HTML الثابتة لإنشاء صفحات ويب نهائية .في Django، يفصل هذا النهج منطق العرض عن منطق الأعمال، مما يجعل الكود أكثر قابلية للصيانة والتوسع.

أهميته في Django: يُعد محرك قوالب Django جزءًا لا يتجزأ من بيئة التطوير، حيث يوفر طريقة قوية ومرنة لإنشاء واجهات المستخدم، مع دعم للميزات مثل الوراثة (inheritance) والفلاتر (filters) والوسوم .(tags)

محركات القوالب: مقارنة سريعة

بينما يمتلك Djangoمحرك قوالب خاص به، هناك بدائل أخرى شائعة مثل Jinja2، والتي تشترك في مفاهيم أساسية ولكن تختلف في بعض التفاصيل والقدرات.

Jinja2	محرك قوالب Django (DTL)
مكتبة خارجية تحتاج إلى التثبيت.	مُضمن مع Django.
يسمح بمنطق أكثر تعقيدًا داخل القوالب.	تركيز على الفصل بين المنطق والعرض.
مرونة أعلى، مفضل للمشاريع الكبيرة والمعقدة.	يُفضل للتعقيد المنخفض والمتوسط.
بناء جملة مستوحاة من Python، مما يجعله مألوفًا للمطورين.	بناء جملة بسيطة ومحدودة.

يُعد محرك قوالب Djangoخيارًا ممتازًا لمعظم مشاريع Djangoبسبب تكامله وسهولة استخدامه.

الفلاتر الافتراضية في Django

توفر Djangoمجموعة غنية من الفلاتر الافتراضية التي تسمح بتعديل وعرض البيانات بسهولة داخل القوالب .دعونا نلقي نظرة على بعض من أهمها:

default

يعرض قيمة افتراضية إذا كانت القيمة فارغة أو غير موجودة.

```
{{ value|default:"لا يوجد" }}
```

length

يعيد طول القيمة)عدد الأحرف أو عناصر القائمة.(

```
{{ my_list|length }}
```

date

يعرض التاريخ بتنسيق معين.

```
{{ value|date:"Y-m-d" }}
```

join

يربط عناصر القائمة بسلسلة محددة.

```
{{ my_list|join:", " }}
```

upper

يحول جميع الأحرف إلى أحرف كبيرة.

```
{{ my_text|upper }}
```

الفلاتر الافتراضية الإضافية في Django

بالإضافة إلى الفلاتر التي ناقشناها، توفر Djangoالعديد من الفلاتر الافتراضية الأخرى التي تعزز من مرونة معالجة البيانات في قوالبك.

add

يضيف قيمة رقمية إلى متغير.

```
{{ value|add:10 }}
```

lower

يحول جميع الأحرف في النص إلى أحرف صغيرة.

```
{{ my_text|lower }}
```

truncatechars

يقوم بتقصير النص إلى عدد محدد من الأحرف، مع إضافة علامات حذف إذا تم تقصيره.

```
{{ value|truncatechars:20 }}
```

cut

يزيل جميع occurrences ofأحرف محددة من النص.

```
{{ my_string|cut:" " }}
```

striptags

يزيل جميع وسوم HTMLمن النص.

```
{{ html_content|striptags }}
```

تساعد هذه الفلاتر في معالجة البيانات وتنسيقها مباشرة داخل القوالب، مما يقلل من الحاجة إلى منطق إضافي في عروض View.

فلاتر Djangoالافتراضية : أمثلة إضافية

تستمر Djangoفي توفير مجموعة واسعة من الفلاتر المفيدة التي يمكن أن تبسط معالجة البيانات وتنسيقها في قوالبك.

first

يعيد أول عنصر في قائمة أو سلسلة نصية.

```
{{ my_list|first }}
```

last

يعيد آخر عنصر في قائمة أو سلسلة نصية.

```
{{ my_list|last }}
```

pluralize

يضيف لاحقة)عادة "s"للكلمات بناءً على عدد، مما يساعد في التعامل مع المفرد والجمع.

```
{{ num_items|pluralize:"s,es" }}
```

wordcount

يعيد عدد الكلمات في سلسلة نصية.

```
{{ my_text|wordcount }}
```

floatformat

ينسق رقمًا عشريًا إلى عدد محدد من المنازل العشرية.

```
{{ value|floatformat:2 }}
```

باستخدام هذه الفلاتر، يمكنك معالجة وعرض بياناتك بفعالية دون الحاجة إلى منطق معقد في ملفات Pythonالخاصة بك.

فلاتر Django الافتراضية: مجموعة شاملة

تستمر Djangoفي تقديم مجموعة واسعة من الفلاتر المدمجة التي تساعد في معالجة وعرض البيانات في قوالبك بفعالية وسهولة.

slugify

يحول النص إلى "slug"صديق لعنوان (URLمثل تحويل المسافات إلى شرطات.)

```
{{ value|slugify }}
```

linebreaks

يحول فواصل الأسطر في النص إلى وسوم HTML (<p>و
).

```
{{ value|linebreaks }}
```

removespace

يزيل جميع المسافات من النص.

```
{{ value|removespace }}
```

timesince

يعرض الفترة الزمنية منذ تاريخ معين)مثال" :قبل 4أيام.("

```
{{ value|timesince }}
```

capfirst

يحول الحرف الأول من النص إلى حرف كبير.

```
{{ value|capfirst }}
```

تُعد هذه الفلاتر أدوات قوية لتبسيط معالجة البيانات وتنسيقها مباشرة داخل القوالب، مما يقلل من الحاجة إلى منطق معقد في ملفات View.

تصميم فلتر Djangoمخصص

عندما لا تكون الفلاتر الافتراضية كافية، يمكنك إنشاء فلاتر مخصصة لتلبية احتياجاتك الخاصة.

01

إنشاء مجلد templatetags

داخل تطبيق Djangoالخاص بك، أنشئ مجلدًا جديدًا باسم `templatetags`بجوار ملف `views.py`أو `models.py`. تأكد من وجود ملف `__init__.py` بداخله لجعله حزمة Python.

02

كتابة ملف الفلتر

داخل مجلد `templatetags`، أنشئ ملف Python (على سبيل المثال `my_filters.py`) واكتب منطق الفلتر الخاص بك فيه:

03

تسجيل الفلتر

استخدم `@register.filter`لتسجيل الدالة الخاصة بك كفلتر.

04

تحميل الفلتر في القالب

في الجزء العلوي من قالب Django، استخدم (`{% load my_filters %}`مع استبدال `my_filters`باسم ملفك (لتحميل الفلاتر المخصصة.

05

استخدام الفلتر

الآن يمكنك استخدام الفلتر الخاص بك بنفس طريقة الفلاتر الافتراضية : `{{ value|my_custom_filter }}`.

مثال عملي: فلتر `status_icon`

لنفترض أن لديك قيمة منطقية (مثل "نعم" أو "لا") وتريد تحويلها إلى أيقونة بصرية جذابة (✅ أو ❌).

`my_filters.py`

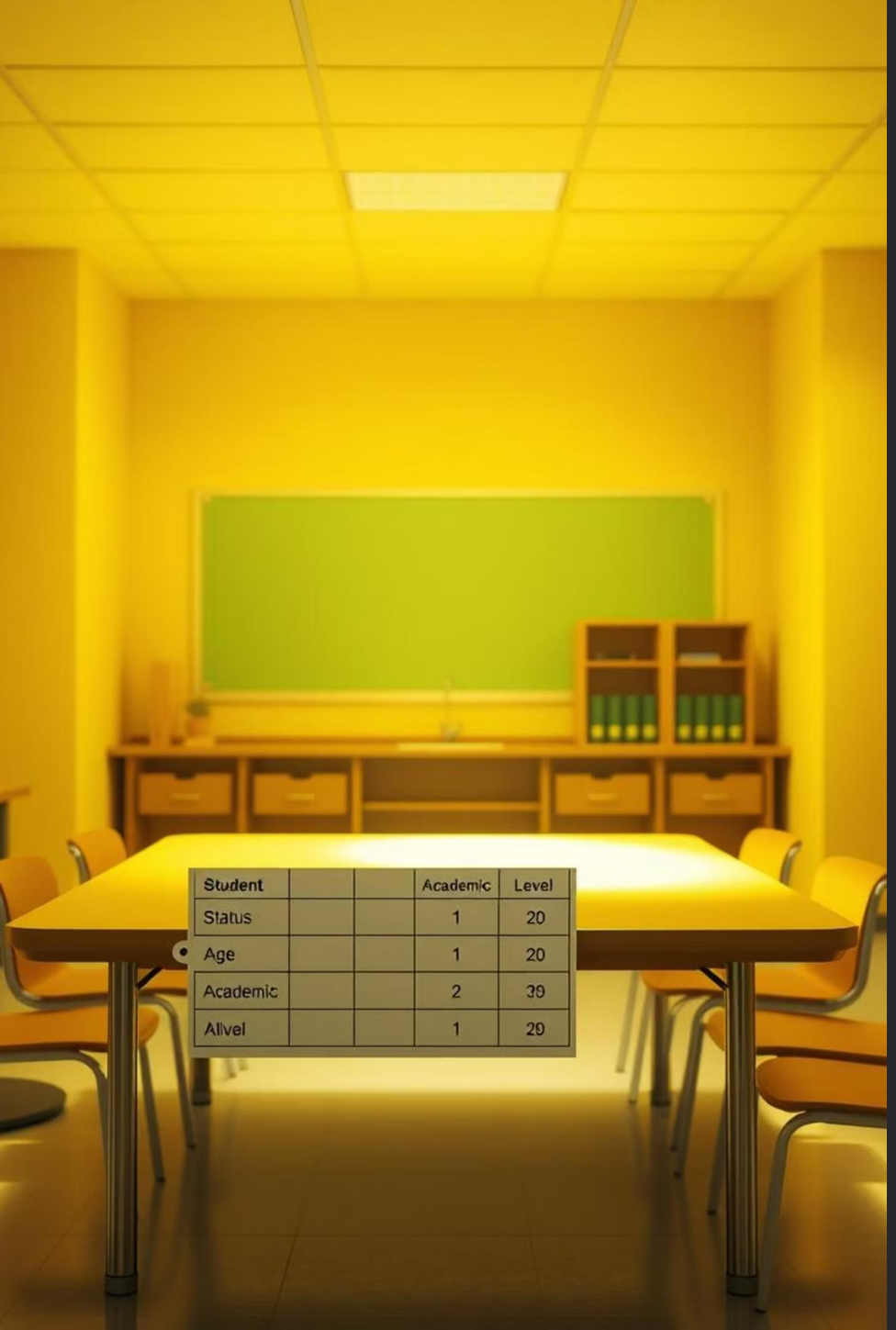
```
from django import templateregister = template.Library()@register.filterdef status_icon(value): if value == "نعم": return "✅"
return "❌"
```

الفلاتر الجاهزة مقابل الفلاتر المخصصة

فهم متى تستخدم كل نوع أمر بالغ الأهمية لتطوير Djangoالفعال.

الفلاتر المخصصة	الفلاتر الجاهزة
• التوفر:	• التوفر:
• الأداء:	• الأداء:
• التخصيص:	• التخصيص:
• متى تستخدم؟	• متى تستخدم؟

الخلاصة :ابدأ دائمًا بالجاهزة، وإذا لم تكن كافية، فقم بإنشاء فلتر مخصص.



تطبيق عملي: جدول الطلاب

دعونا نطبق ما تعلمناه على جدول افتراضي للطلاب لإظهار كيف يمكن للفلاتر تحسين عرض البيانات.

سنستخدم الفلاتر المخصصة التي أنشأناها لتلوين الصفوف بناءً على الحالة أو الجنس، وعرض رموز للحالة والعمر والمستوى الدراسي.

تلوين الصفوف والبيانات باستخدام الفلاتر

لنفترض أن لدينا بيانات الطلاب التالية)كمثال:(

```
'},    {'name': متوسط', 'level': 'ذكر', 'age': 20, 'gender': 'نعم', 'status': 'أحمدstudents = [    {'name': '
', نعم', 'status': 'علي'},    {'name': 'مبتدئ', 'level': 'أنثى', 'age': 15, 'gender': 'لا', 'status': 'سارة'
'},]  متقدم', 'level': 'ذكر'age': 25, 'gender': '
```

باستخدام فلاتر مخصصة (row_color_by_status، status_icon، age_group، level_abbr) يمكننا تحويل العرض:

```
{% load my_filters %}        {% for student in students %}                {% endfor %}
```

الاسم	الحالة	العمر	المستوى
{{ student.name }}	{{ student.status|status_icon }}	{{ student.age|age_group }}	{{ student.level|level_abbr }}

هذا يوضح كيف يمكن للفلاتر أن تجعل الجداول أكثر وضوحًا وجاذبية بصريًا.

الخلاصة والخطوات التالية

لقد استعرضنا كيف تُعد محركات القوالب والفلاتر في Django أدوات قوية لإنشاء واجهات ديناميكية ومرنة.

محركات القوالب

هي القلب الذي يدمج البيانات بالواجهة، مما يضمن فصلًا نظيفًا للمسؤوليات.

الفلاتر الافتراضية

توفر حلولًا سريعة وفعالة لمهام تنسيق البيانات الشائعة.

الفلاتر المخصصة

تطلق العنان لقدرات لا نهائية في تحويل البيانات لتناسب أي سيناريو خاص بتطبيقك.

الخطوات التالية: جرب إنشاء فلاتر مخصصة خاصة بك، واستكشف المزيد من الفلاتر الافتراضية المتاحة في توث