



الاسم/ضرار علي محمد عقلا م ٤-ت ١ مجموعة (B)

تكليف رقم (5) مقرر: هندسة برمجيات م/مالك المصنف

حظرت مع طلاب مجموعة (C/B1)

مقارنة بين طرق انشاء الفورم في البايثون (تحويل من Html الى python)

1. الربط مع الموديل (Model Binding)

تعتمد هذه الطريقة على إطار عمل (Framework) يوفر ميزة الربط التلقائي بين حقول الفورم في HTML والبيانات في Python، غالبًا ما يكون ذلك مرتبطًا بـ الموديل (Model) الذي يمثل جدولًا في قاعدة البيانات أو فئة (Class).

كيفية عملها:

- تُعرّف كلاس (Class) في Python يسمى (Model) يمثل هيكل البيانات الذي تريد استقباله (مثلًا، فئة User تحتوي على حقول name, email, password).
- يستخدم إطار العمل هذه الفئة لإنشاء فورم برمجياً (Form Class) تلقائياً.
- يتم تمرير هذا الفورم إلى قالب HTML ، ويقوم قالب HTML بتوليد الحقول المطلوبة تلقائياً بناءً على الموديل.
- عند استقبال الطلب، يقوم إطار العمل بإنشاء كائن (Object) من هذا الموديل ويملأه بالبيانات القادمة من الفورم تلقائياً.
- يتضمن هذا الكائن وظائف جاهزة للتحقق من صحة البيانات (Validation) وتحويلها.

لمميزات:

- كود نظيف ومختصر: يختصر الكثير من الكود المكرر، خاصة في التحقق من البيانات.
- صيانة أسهل: أي تغيير في الموديل (مثل إضافة حقل) سينعكس تلقائياً في الفورم دون الحاجة لتغيير كود آخر.
- تحقق من البيانات (Validation): يتم بشكل تلقائي وفعال، حيث يمكن تعريف قواعد التحقق مباشرة في الموديل.
- التكامل مع قاعدة البيانات: غالبًا ما تكون هذه الطريقة مرتبطة مباشرة بقاعدة البيانات، مما يسهل عمليات الحفظ والتحديث.

العيوب:

- تعقيد أكبر: تتطلب فهمًا أعمق لإطار العمل وكيفية عمل الموديلات والفورمات.
- مرونة أقل في الواجهة: قد يكون التحكم الكامل في شكل الفورم في HTML أصعب قليلًا، رغم أن معظم الأطر تسمح بالتخصيص.
- قد تكون غير ضرورية: إذا كان الفورم بسيطًا جدًا، قد لا تكون هذه الطريقة فعالة وتضيف تعقيدًا غير لازم.

التنفيذ:

كود دالة `views.read`

```
return render(request, 'home.html')
```

```
def read(request):
```

```
    students=Students.objects.all()
```

```
    students_count=Students.objects.all().count()
```

```
    return render(request, 'allStudents.html',{'students':students,'students_count':students_count})
```

داخل صفحة allstudent.html

التنفيذ:

The screenshot shows a web browser window displaying a page titled "قائمة الطلاب" (Students List). The page has a dark blue header with navigation links: "الرئيسية" (Home), "المعلمين" (Teachers), and "الطلاب" (Students). Below the header, there is a table with the following data:

الاسم الأول	الاسم الأخير	العمر	المرحلة الجامعية	المعدل	منتظم	ملاحظة	تحكم
Dherar	Ali	23	4	3.50	True	طالب مجتهد في كلية العلوم التطبيقية	<i></i> <i></i>
المجموع:							1

Below the browser window, a code editor shows the HTML code for the table. The code is as follows:

```
86 <td>
87 |
88 <a href=""><i class="fa-solid fa-pen-to-square"></i></a>
89 <a href="" class="text-danger"><i class="fa-solid fa-trash"></i></a>
90 </td>
```

2. المعالجة اليدوية للحقول (Manual Field Handling)

تعتبر هذه الطريقة هي الأساس، حيث تقوم بإنشاء فورم HTML بنفسك، ثم في جانب الخادم (Python)، تستقبل البيانات وتستخرج قيمة كل حقل على حدة.

كيفية عملها:

- في HTML، تُنشئ الفورم باستخدام وسوم `<form>`, `<input>`, `<label>`, `<textarea>`، وهكذا، وتُعطى لكل حقل اسمًا (name) فريدًا.
- في Python باستخدام إطار عمل مثل Flask أو Django، تستقبل الطلب (Request) الذي يحمل بيانات الفورم.
- تستخدم أدوات الإطار للحصول على قيمة كل حقل باستخدام اسمه، مثل `request.form['اسم_الحقل']`.
- بعد ذلك، تعالج هذه البيانات: تقوم بالتحقق من صحتها (Validation)، أو تخزينها في قاعدة البيانات، أو أي عملية أخرى.

المميزات:

- بساطة ومرونة عالية: لا تحتاج إلى أي مكتبات إضافية، وتتحكم بشكل كامل في كل تفاصيل الفورم.
- مناسبة للحالات البسيطة: إذا كان الفورم يحتوي على عدد قليل من الحقول ولا يتطلب تعقيدًا.
- فهم أعمق: تساعدك على فهم كيفية عمل بروتوكول HTTP واستقبال البيانات.

العيوب:

- تكرار الكود: قد يصبح الكود طويلًا ومكررًا عند التعامل مع عدد كبير من الحقول.
- صعوبة في الصيانة: إذا غيّرت اسم حقل في HTML، يجب أن تتذكر تغييره في كل مكان في كود Python.
- التحقق من البيانات (Validation): يجب أن تقوم بالتحقق من صحة كل حقل يدويًا، مما يزيد من التعقيد.

التنفيذ:

نعدل كود دالة `views.read` حيث يظهر نافذة بحث
استيراد مكتبة

```
from django.db.models import Q
```

```
def read(request):
    # فهذا يعني أن المستخدم أرسل بيانات البحث، POST إذا كان الطلب من نوع
    if request.method == 'POST':
        query = request.POST.get('query')
        if query:
            # (f_name أو l_name) للبحث في عدة حقول Q objects استخدام
            # lookup iendswith بالحرف الأخير هي للاستعلام
            students = Students.objects.filter(
                Q(f_name__icontains=query) | Q(l_name__icontains=query)
            ).order_by('f_name')
        else:
            students = Students.objects.all().order_by('f_name')
    # فهذا يعني أن المستخدم يطلب الصفحة لأول مرة، GET إذا كان الطلب من نوع
    else:
        students = Students.objects.all().order_by('f_name')

    students_count = students.count()
    return render(request, 'allStudents.html', {'students': students, 'students_count': students_count})
```

إضافة الى كود allstudent.html

```
{% csrf_token %}







<main>

    <!-- لعرض زر بحث وإنشاء فورم -->
    <form method="POST">
        {% csrf_token %}
        <div class="input-group mb-3">
            <input type="text" name="query" class="form-control" placeholder="... ابحث عن طالب">
            <button class="btn btn-outline-secondary" type="submit">بحث</button>
        </div>
    </form>

</html>
```

التنفيذ:

الرتبسية	المعلمين	الطلاب
قائمة الطلاب		

الاسم الأول	الاسم الأخير	العمر	المرحلة الجامعية	المعدل	منتظم	ملاحظة	تحكم
Dherar	Ali	23	4	98.99	True	طالب مجتهد في كلية العلوم التطبيقية	  
محمد	علي	21	3	80.23	True	جيد جداً	  
المجموع:		2					

طريقة أخرى باستخدام جملة الاستعلام:

استيراد مكتبة connections

```
from django.db import connections
```

نعدل كود دالة read:

```
def read(request):
    with connections['default'].cursor() as cursor:
        # students_students يدويًا لجلب جميع البيانات من جدول SQL كتابة استعلام
        # اسم التطبيق اسم المودل students_students اسم الجدول هو
        cursor.execute("SELECT * FROM students_students")
        # جلب جميع الصفوف من نتيجة الاستعلام
        rows = cursor.fetchall()

        # تحويل البيانات إلى قائمة من القواميس لتسهيل التعامل معها في القالب
        students = []
        for row in rows:
            student = {
                'f_name': row[1],
                'l_name': row[2],
                'age': row[3],
                'level': row[4],
                'gpa': row[5],
                'statust': row[6],
                'reporer': row[7]
            }
            students.append(student)
        # حساب عدد الطلاب
        students_count = len(students)
        # إرسال البيانات إلى القالب
        return render(request, 'allStudents.html', {'students': students, 'students_count': students_count})
# students list=[
```

التنفيذ:

الرئيسية المعلمين الطلاب

قائمة الطلاب

الاسم الأول	الاسم الأخير	العمر	المرحلة الجامعية	المعدل	منتظم	ملاحظة	تحكم
Dherar	Ali	23	4	98.99	1	طالب مجتهد في كلية العلوم التطبيقية	
محمد	علي	21	3	80.23	1	جيد جداً	
المجموع:						2	

اكتب هنا للبحث