بسم الله الرحمن الرحيم

الاسم/ ضرار علي محمد عقلان م٤-ت١ مجموعة(B2)

تكليف رقم(4) مقرر: هندسة برمجيات م/مالك المصنف

تكليف: مقارنة بين أنظمة إدارة قواعد البيانات:

اولاً: ما هي أنظمة إدارة قواعد البيانات؟

أنظمة إدارة قواعد البيانات (Database Management Systems - DBMS) هي برامج تُستخدم لتخزين وإدارة واسترجاع البيانات من قواعد البيانات. تُقسم بشكل عام إلى فئتين رئيسيتين:

١. قواعد البيانات العلائقية :(Relational Databases) تُخزّن البيانات في جداول منظّمة مع علاقات محددة بينها. تستخدم لغة SQL (Structured Query Language) لإدارة البيانات.

٢. قواعد البيانات غير العلائقية :(Non-Relational Databases - NoSQL) تُخزّن البيانات بتنسيقات أكثر مرونة، مثل المستندات أو أزواج المفاتيح-القيمة، ولا تتطلب مخططًا ثابتًا. تُستخدم غالبًا في التطبيقات التي تحتاج إلى قابلية توسع عالية أو التعامل مع البيانات غير المهيكلة.

ثانياً: مقارنة بين أنظمة إدارة قواعد البيانات

سنقارن بين بعض أشهر الأنظمة العلائقية وغير العلائقية:

SQLite.1

- النوع :علائقية.
- المميزات:
  - خفيفة الوزن ومدمجة (تُخزّن في ملف واحد).
  - لا تتطلب خادمًا منفصلاً.
  - سهلة الإعداد والاستخدام، ومثالية للتطبيقات الصغيرة أو لأغراض التطوير والاختبار.
- العيوب:
  - لا تناسب التطبيقات الكبيرة التي تتطلب قابلية توسع أو تعدد المستخدمين.
  - أداء محدود في بيئات الإنتاج المعقدة.

PostgreSQL.2

- النوع :علائقية.

- المميزات:
  - **نظام قوي ومتقدم يدعم ميزات معقدة مثل أنواع البيانات المتقدمة، والفهرسة المتطورة، والبحث النصي الكامل.**
  - **يتميز بالموثوقية وسلامة البيانات.(ACID compliance)**
  - **مناسب للتطبيقات الكبيرة والمعقدة التي تتطلب موثوقية عالية.**
- **العيوب:**
  - **قد يكون إعداده وإدارته أكثر تعقيدًا من الأنظمة الأخرى.**

MySQL.3

- **النوع** :علائقية.
- **المميزات**:
  - **أكثر الأنظمة العلائقية شهرة واستخدامًا في تطبيقات الويب.**
  - **أداؤه سريع في عمليات القراءة المتكررة.**
  - **سهل الاستخدام وله مجتمع كبير يدعمه.**
- **العيوب**:
  - **قد لا يكون بنفس قوة PostgreSQL في التعامل مع بعض الميزات المتقدمة.**
  - **لا يتوافق دائمًا بشكل كامل مع معايير.SQL**

MongoDB.4

- **النوع** :غير علائقية.(NoSQL)
- **المميزات**:
  - **يخزن البيانات في مستندات شبيهة بـ.JSON**
  - **مرن جدًا ولا يتطلب مخططًا ثابتًا، مما يسهل التعامل مع البيانات غير المهيكلة.**
  - **قابل للتوسع الأفقي(horizontally scalable) ، مما يجعله مثاليًا للبيانات الضخمة والتطبيقات الموزعة.**
- **العيوب**:
  - **لا يدعم المعاملات المعقدة بنفس طريقة قواعد البيانات العلائقية.**
  - **قد يكون أداؤه أبطأ في عمليات البحث التي تتطلب علاقات معقدة.**
  - **غير مناسب للتطبيقات التي تتطلب مخططًا صارمًا.**

Microsoft SQL Server.5

- **النوع :علائقي.**
- **المميزات:**
  - **قوة وموثوقية :مصمم للتعامل مع كميات هائلة من البيانات والمعاملات المعقدة في بيئات الإنتاج الكبرى.**
  - **الأداء :يتميز بأداء عالٍ في معالجة الاستعلامات، خاصةً على أنظمة ويندوز.**
  - **التكامل مع منتجات مايكروسوفت :يتكامل بشكل ممتاز مع أدوات أخرى مثل Power BI وAzure، مما يجعله خيارًا مفضلاً للشركات التي تستخدم نظام.Microsoft**
  - **ميزات أمنية متقدمة :يوفر SQL Server ميزات أمان قوية لحماية البيانات.**
- **العيوب:**
  - **التكلفة :النسخ المدفوعة قد تكون باهظة الثمن.**
  - **التعقيد :إعداده وإدارته قد يكونان أكثر تعقيدًا مقارنةً بـ PostgreSQL أو.MySQL**
  - **التوافق :قد لا يكون سهل الاستخدام على أنظمة التشغيل غير ويندوز مقارنة بالأنظمة الأخرى.**

ثالثاً: طريقة استخدامها في (Django)

لتغيير قاعدة البيانات فيDjango ، يجب تعديل إعدادات المشروع في ملف .settings.py

1. **تثبيت الموصل :(Connector) يجب تثبيت مكتبة Python التي تربط Django بقاعدة البيانات.**
   - **لـPostgreSQL: pip install psycopg2**
   - **لـMySQL: pip install mysqlclient**
2. **تعديل ملف :settings.pyداخل قاموس DATABASES، قم بتحديد المحرك(ENGINE) ، واسم قاعدة البيانات(NAME) ، وبيانات الاعتمادUSER) ، PASSWORD، HOST، .(PORT**

مثال لربط:PostgreSQL

```
DATABASES = {
'default' : {
'ENGINE': 'django.db.backends.postgresql',
'NAME': 'your_db_name',
'USER': 'your_db_user',
'PASSWORD': 'your_db_password',
'HOST': 'localhost', # أو عنوان IP لخادم قاعدة البيانات
'PORT': '5432',
```

```
}
}
```

مثال لربط:MySQL

```
DATABASES = {
'default' : {
'ENGINE': 'django.db.backends.mysql',
'NAME': 'your_db_name',
'USER': 'your_db_user' ,
'PASSWORD': 'your_db_password',
'HOST': 'localhost' ,
'PORT': '3306',
}
}
```

تشغيل الهجرات :(Migrations)

بعد تعديل الإعدادات، قم بتشغيل الأوامر التالية لإنشاء الجداول في قاعدة البيانات الجديدة:

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

استخدام قواعد البيانات غير العلائقية(NoSQL)

لا يدعم Django قواعد بيانات NoSQL بشكل رسمي، ولكن يمكن استخدامها عبر حزم إضافية (third-party packages) مثل Djongo الذي يسمح بربط Django بـ MongoDB. هذه الحزم تعمل على تكييف ORM الخاص بـ Django للعمل مع NoSQL.

مثال لربط MongoDB باستخدام:Djongo

١. **تثبيت Djongo**: pip install djongo

٢. **تعديل settings.py**:

```
DATABASES ={
'default':{
```

'ENGINE': 'djongo',

'NAME': 'your_mongo_db_name',

}

{

استخدام SQL Server معDjango

طريقة الاستخدام

١. تثبيت الموصل :(Connector) أشهر حزمة لربط Django بـ SQL Server هي django-mssql-backend.

- تثبيت الحزمة:

pip install django-mssql-backend

تثبيت مكتبة الاتصال :ستحتاج أيضًا إلى تثبيت مكتبة للاتصال بقاعدة البيانات، مثل pyodbc، التي تُستخدم للاتصال بقواعد البيانات عبر ODBC.

pip install pyodbc

2. تعديل ملف :settings.py

بعد تثبيت الحزم، قم بتعديل قاموس DATABASES في ملف الإعدادات.

DATABASES ={

'default' : {

'ENGINE': 'sql_server.pyodbc', # اسم المحرك

'NAME': 'your_db_name',

'USER': 'your_db_user',

'PASSWORD': 'your_db_password',

'HOST': 'localhost', # أو عنوان IP لخادم SQL Server

'PORT': '1433', # المنفذ الافتراضي لـ SQL Server

'OPTIONS': {

'driver': 'ODBC Driver 17 for SQL Server', # قد تحتاج إلى تغيير اسم الـ driver

}

}

}

3. تشغيل الهجرات (Migrations):

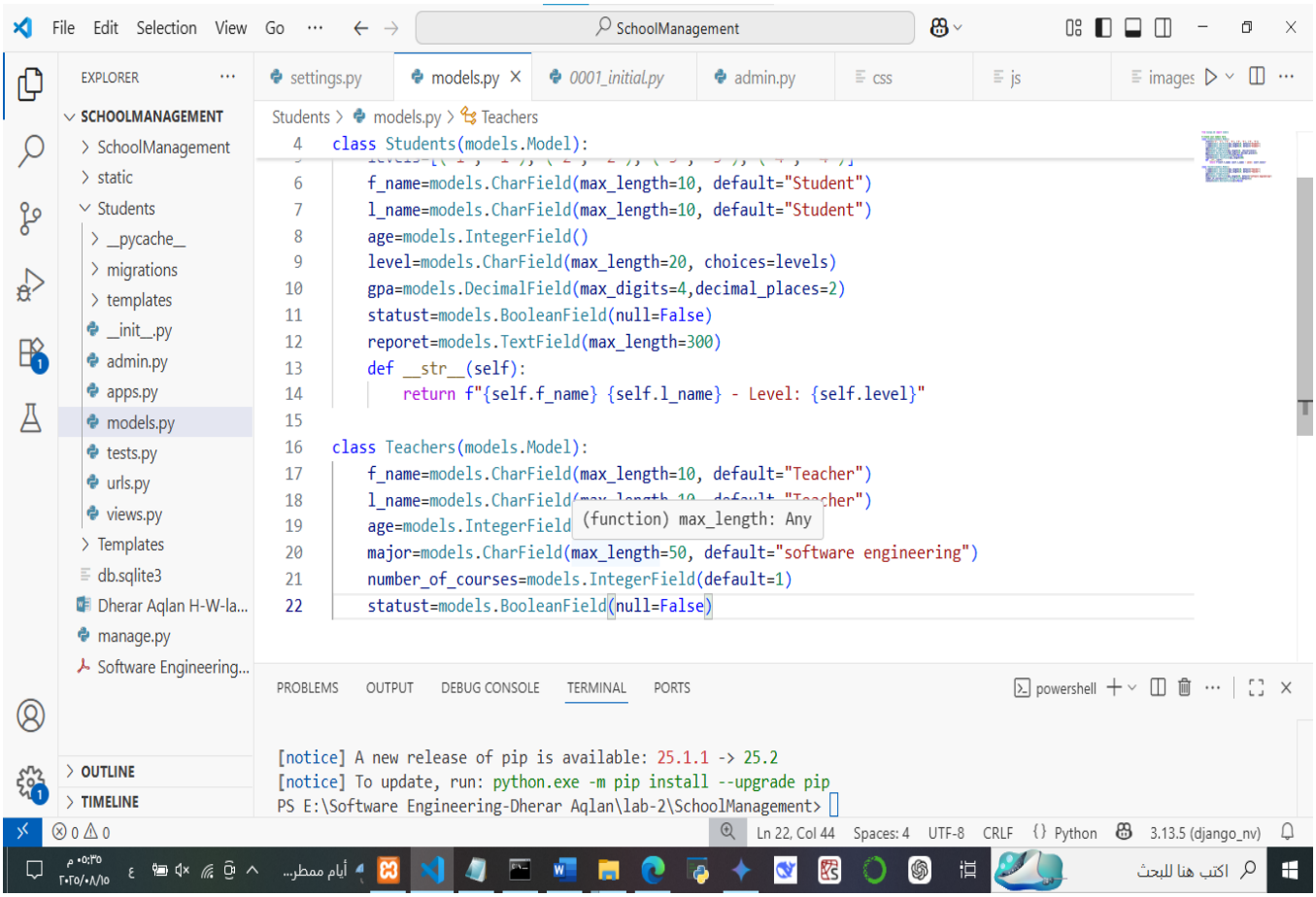
بعد تعديل الإعدادات، يمكنك تشغيل أوامر الهجرة لإنشاء الجداول:

```
python manage.py makemigrations
```

```
python manage.py migrate
```

تكليف٢ : إضافة جدول خاص بالمعلمين.

١.شكل MODEL جدول المعلمين.

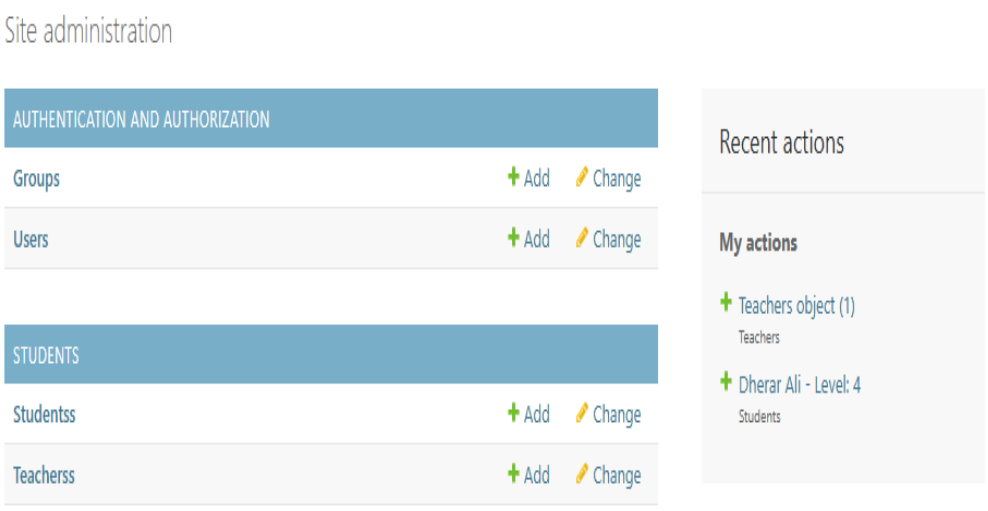


٢.ملف migrations

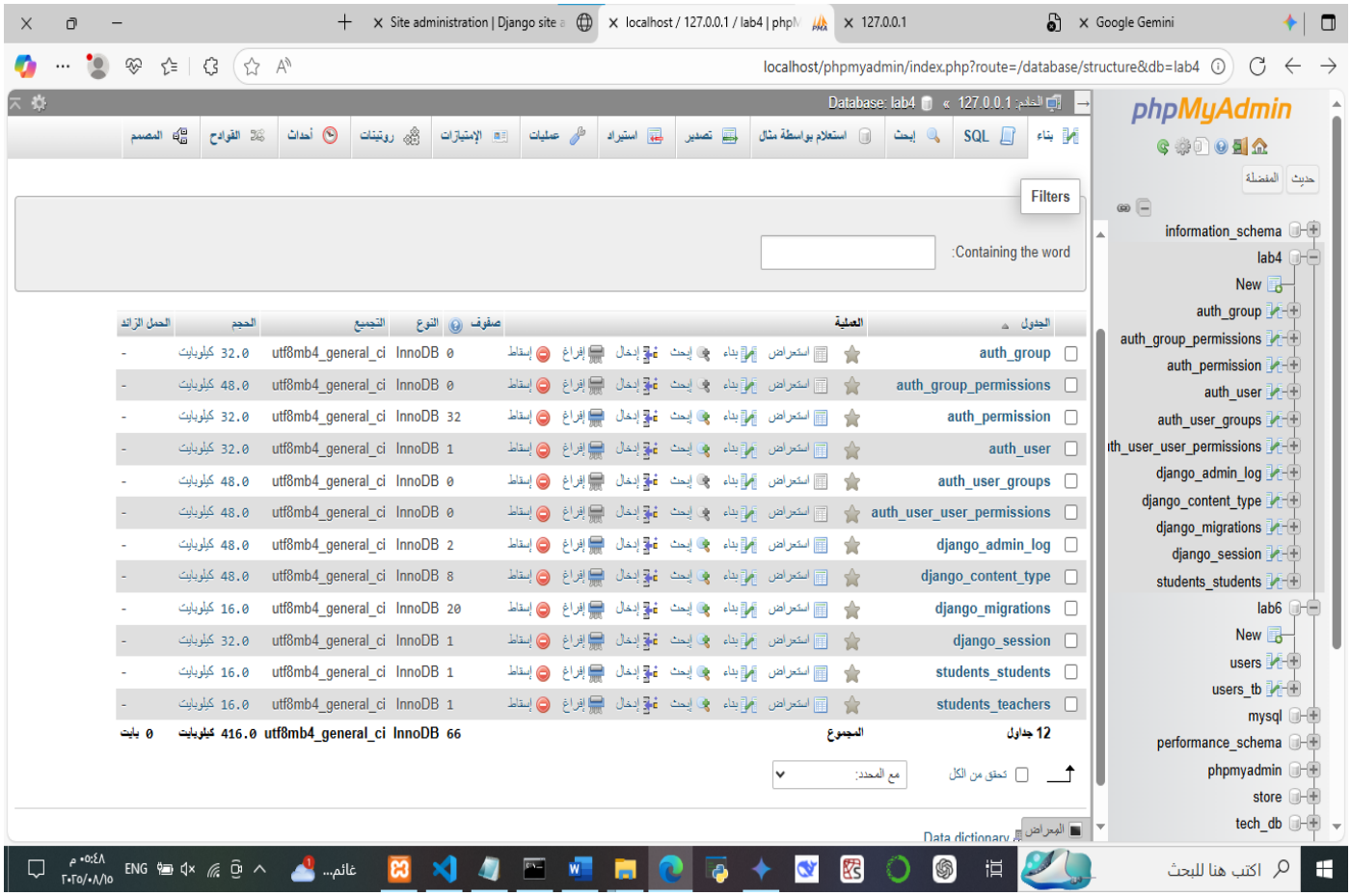
Students > migrations > 0002_teachers.py > ...

```python
class Migration(migrations.Migration):

    dependencies = [
        ('Students', '0001_initial'),
    ]

    operations = [
        migrations.CreateModel(
            name='Teachers',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False,
                ('f_name', models.CharField(default='Teacher', max_length=10)),
                ('l_name', models.CharField(default='Teacher', max_length=10)),
                ('age', models.IntegerField()),
                ('major', models.CharField(default='software engineering', max_length=50)),
                ('number_of_courses', models.IntegerField(default=1)),
                ('statust', models.BooleanField()),
            ],
        ),
```

٣.الجدول في admin



٤.الجدول في قاعدة البيانات.



٥.عرض بيانات المعلم المضاف في الجدول.