Django and Database

Building an Airlines App.

1. Creating django project by typing the following command:

django-admin startproject

2. starting airlines app by typing this command:

Python manage.py startapp flights

3. go to settings.py and type in the section of the Installed apps: add ‘flights’

```
# Application definition

INSTALLED_APPS = [
    'flights',
    'flights1',
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
]
```

4. in URLs.Py writing the following code in order to map the URL for the project:

```
16  from django.contrib import admin
17  from django.urls import include, path
18
19  urlpatterns = [
20      path('admin/', admin.site.urls),
21      path('flights/', include("flights.urls")),
22      path('flights1/', include("flights1.urls")),
23  ]
```

5. in the flights Folder-> add new urls.py file in order to map the url for the app:

```
flights1 > urls.py > ...
1   from django.urls import path
2   from . import views
3   urlpatterns=[
4       path('', views.index, name="index"),
5   ]
```

هي طريقة لإنشاء فئة بيثون سيتم تمثيلهاالبيانات التي سيبدأها
django
داخل قاعدة البيانات

6. Let's start by creating a model, a model is a way of creating a python class that is going to represent data that django is going to start inside a database.

So, when a model is created, django is going to figure out what SQL syntax it is going to use to create the tables and how to manipulate that table by selecting, updating, inserting or deleting any time a changes are made to those models.

لذلك ، عندما يتم إنشاء نموذج ، فإن
django
سوف يكتشف بناء جملة
SQL
الذي سيستخدمه لإنشاء الجداول وكيفية التعامل مع هذا الجدول عن طريق تحديد
أو تحديث أو إدراج أو حذف في أي وقت يتم إجراء تغييرات على هذه النماذج.

Go flights Folder->models.py:

This is where we are going to define the models for our application.

Every model is going to be a python class.

```
flights1 > models.py > ...
1   from django.db import models
2
3   # Create your models here.
4   class Flight1(models.Model):
5       origin=models.CharField(max_length=64)
6       destination=models.CharField(max_length=64)
7       duration=models.IntegerField()
```

لإخبار
django
هو كيفية تحديث قاعدة البيانات لتضمين معلومات حول النموذج الذي تم
إنشاؤه وهذه هي العملية المشار إليها في
django
وتسمى بشكل عام "الترحيل".

To tell django is how you should update the database to include information about the model that is been created and this is the process referred in django and more generally called '*migration*'.

I created a migration to say here is some changes that I would like to apply to the database and then I migrate them to tell django to take these changes and actually apply them to the database in two step-process:

لقد أنشأت ترحيلاً لأقول هنا بعض التغييرات التي أود تطبيقها على قاعدة البيانات ثم أقوم بترحيلها لإخبار
django
بأخذ هذه التغييرات وتطبيقها فعليًا على قاعدة البيانات في عمليتين مرحلتين:

1. Creating the migration, the instructions how to manipulate the database تعليمات كيفية التعامل مع قاعدة البيانات

To create migration type the following command:

Python manage.py makemigrations

```
workspace@Brian-MBP airline % python manage.py makemigrations
Migrations for 'flights':
  flights/migrations/0001_initial.py
    - Create model Flight
```

A file has been created called 0001_initional.py under the migration folder

This file is a set of instructions to django to how to manipulate the database to reflect the changes to the model.

هذا الملف عبارة عن مجموعة من التعليمات إلى django حول كيفية معالجة قاعدة البيانات لتعكس التغييرات على النموذج.

2. to take the migrations and apply them to the underlying database

لأخذ عمليات الترحيل وتطبيقها على قاعدة البيانات الأساسية

Type the following command:

Python manage.py migrate

```
workspace@Brian-MBP airline % python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, flights,
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... O
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages...
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying flights.0001_initial... OK
```

7. Enter django shell to work on the database and variables, by typing this command:

أدخل django shell للعمل على قاعدة البيانات والمتغيرات ، عن طريق كتابة هذا الأمر:

Python manage.py shell

And you can start typing a python commands that can be executed on the web application:

ويمكنك البدء في كتابة أوامر بيثون يمكن تنفيذها على تطبيق الويب:

```
HP@DESKTOP-JEO3C1K MINGW64 ~/lecture7
$ python manage.py shell
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>> from flights1.models import Flight1
>>> f=Flight1(origin="New York", destination="London", duration=415)
>>> f.save()
```

So, this is the way of how to insert data into the database

When running this command django knows that it has to run insert command under the sqlite

عند تشغيل هذا الأمر ، يعرف django أنه يجب تشغيل الأمر insert تحت sqlite

8. Getting information from Flight

```
>>> Flight.objects.all()
<QuerySet [<Flight: Flight object (1)>]>
>>>
```

This command is equivalent to sql command: select * from Flight1

To change the representation of the extracted record we can add the function : لتغيير تمثيل السجل المستخرج يمكننا إضافة الوظيفة:

def __str__(self):

this function return string representation of a particular object in the models.py file:

flights1 > models.py > ...

```python
from django.db import models

# Create your models here.
class Flight1(models.Model):
    origin=models.CharField(max_length=64)
    destination=models.CharField(max_length=64)
    duration=models.IntegerField()
    def __str__(self):
        return f"{self.id}: {self.origin} to {self.destination}"

```