İlk Django Modelinizi Oluşturun

Gerekli tahmini süre: 15 dakika

Öğrenme Hedefleri

Bu laboratuvarı tamamladıktan sonra şunları yapabileceksiniz:

- Theia IDE ve Django ile tanışmak
- Bağımsız bir Django ORM uygulaması kurmak
- İlk Django projenizi ve uygulamanızı oluşturmak
- İlk Django modelinizi oluşturup test etmek

Bulut IDE'de Dosyalarla Çalışma

Bulut IDE'ye yeniyseniz, bu bölüm size projenizin bir parçası olan dosyaları nasıl oluşturup düzenleyeceğinizi gösterecektir.

Bulut IDE içinde dosyalarınızı ve dizinlerinizi görüntülemek için bu dosya simgesine tıklayın.

`Yeni`ye tıklayın, ardından `Yeni Terminal` seçeneğine tıklayın.

Bu, komutlarınızı çalıştırabileceğiniz yeni bir terminal açacaktır.

Laboratuvarda Ele Alınan Kavramlar

1. `Django ORM`: Her Django modelinin bir veritabanı tablosuna eşlendiği Django web uygulama çerçevesinin bir Python ORM bileşeni.
2. `PostgreSQL` veya `Postgres`: Django tarafından kullanılan açık kaynaklı ilişkisel veritabanı yönetim sistemi.
3. `Psycopg`: Django'nun PostgreSQL ile çalışmak için kullandığı bir arayüz.
4. `PGAdmin`: PostgreSQL sunucusunu yönetmek için kullanılan bir yönetim ve geliştirme aracı.
5. `Database migrations`: İlişkisel veritabanı şemalarına yapılan, versiyon kontrolü altında, artımlı ve geri alınabilir değişikliklerin yönetimi; şemayı daha yeni veya daha eski bir sürüme güncellemek veya geri almak için.

Theia'da PostgreSQL Başlatma

`PostgreSQL`, diğer adıyla `Postgres`, açık kaynaklı bir ilişkisel veritabanı yönetim sistemidir ve Django'nun kullandığı ana veritabanlarından biridir.

Eğer Skills Network Labs tarafından barındırılan Theia ortamını kullanıyorsanız, sizin için önceden yüklenmiş bir PostgreSQL örneği sağlanmaktadır.

PostgreSQL'i, sol menü çubuğundaki SkillsNetwork simgesini bularak ve DATABASES menü öğesinden PostgreSQL'i seçerek UI'dan başlatabilirsiniz:

PostgreSQL başlatıldıktan sonra, sunucu bağlantı bilgilerini UI'dan kontrol edebilirsiniz. Django uygulamanızın bu veritabanına bağlanabilmesi için kullanılacak `username`, `password` ve `host` gibi bağlantı bilgilerini markdown formatında not edin.

- Postgres'e erişim ortamını kurmadan önce bu gerekli paketleri yükleyin.

```
pip install --upgrade distro-info
pip3 install --upgrade pip==23.2.1
```

- Ayrıca, sizin için bir `pgAdmin` örneği kuruldu ve başlatıldı. Bu, PostgreSQL sunucunuzu etkileşimli olarak yönetmek için popüler bir PostgreSQL yönetim ve geliştirme aracıdır.

İlk Django Uygulamanızı Kurun

Eğer terminal açık değilse, `Terminal > Yeni Terminal` yolunu izleyin ve mevcut Theia dizininizin `/home/project` olduğundan emin olun.

- Bu laboratuvar için bir kod şablonu indirmek üzere aşağıdaki komut satırlarını çalıştırın.

```
wget "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-CD0251EN-SkillsNetwork/labs/m3_django_orm/lab1_template.zip"
```

```
unzip lab1_template.zip
rm lab1_template.zip
```

Son `rm` komutunu çalıştırmak için `Enter` tuşuna basmanız gerekebilir.

İndirme ve sıkıştırma işlemi tamamlandıktan sonra, sol menüdeki `Explorer` butonuna (ilk buton) tıklayın.

İlk Django projeniz aşağıdaki gibi görünmelidir:

Sonraki adımda, Django uygulaması için uygun bir çalışma ortamı kurmamız gerekiyor.

- Terminal açık değilse, `Terminal > New Terminal` yolunu izleyin ve proje klasörüne `cd` komutuyla gidin.

```
cd lab1_template
```

Tüm ihtiyaç duyduğumuz paketleri içerecek bir sanal ortam oluşturalım.

```
pip install virtualenv
virtualenv djangoenv
source djangoenv/bin/activate
```

```
pip install -r requirements.txt
```

`requirements.txt` dosyası, bu laboratuvarı çalıştırmak için gerekli olan tüm Python paketlerini içerir.

Oluşturulan projede bazı önemli proje dosyalarını bulabilirsiniz:

- `manage.py`, Django projenizle etkileşimde bulunmanızı ve yönetmenizi sağlayan bir komut satırı arayüzüdür.
- `settings.py`, veritabanları veya kurulu Django uygulamaları gibi bu projeye ait ayar bilgilerini içerir.
- `orm` klasörü, bağımsız bir Django ORM uygulaması için bir konteynırdır.
- `orm/models.py`, model tanımlarını içerir.

Bu dosyalar hakkında daha fazla detayı sonraki öğrenme modüllerinde ve laboratuvarlarda öğreneceksiniz.

Şimdi Django projemizi başlattığımız `PostgreSQL` ile bağlayalım.

- `settings.py` dosyasını açın ve `DATABASES` bölümüne gidin.

```
# Postgre SQL
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql_psycopg2',
        'NAME': 'postgres',
        'USER': 'postgres',
        'PASSWORD': 'Place it with your password saved in Step 1',
        'HOST': 'postgres',
        'PORT': '5432',
```

```
        }
    }
```

- `PASSWORD` değerini Adım 1'de oluşturulan `PostgreSQL` parolası ile değiştirin.

Bundan sonra, `settings.py` dosyanız aşağıdaki gibi görünmelidir:

- Tamam, şimdi ilk Django `orm` uygulamanız hazır ve bir sonraki adımda ilk Django modelinizi tanımlamaya başlayabilirsiniz.

İlk Django Modelinizi Tanımlayın

- `lab1_template/orm/` klasöründeki `orm/models.py` dosyasını açın ve aşağıdaki parçayı yorumun altına kopyalayın / yapıştırın

`# İlk modelinizi buradan tanımlayın:` bir `User` modeli tanımlamak için

```
    class User(models.Model):
        # CharField for user's first name
        first_name = models.CharField(null=False, max_length=30, default='john')
        # CharField for user's last name
        last_name = models.CharField(null=False, max_length=30, default='doe')
        # CharField for user's date for birth
        dob = models.DateField(null=True)
```

Artık yalnızca `first_name` ve `last_name` alanlarını `CharField` olarak ve `dob` alanını `DateField` olarak içeren çok basit bir `User` modeli tanımladınız.

Kullanıcı Modelini Aktifleştir

`User` modeli tanımlandıktan sonra, Django karşılık gelen bir veritabanı tablosu oluşturacaktır
`orm_user` adında. İlk kısım `orm` uygulama adınız ve ikinci kısım `user` model adıdır.

Modellerinizde yeni modeller oluşturma veya mevcut modelleri değiştirme gibi değişiklikler yaptığınızda, veritabanı göçleri gerçekleştirmeniz gerekir. Django, göçleri gerçekleştirmenize yardımcı olmak için `manage.py` arayüzü aracılığıyla yardımcı araçlar sağlar.

- Eğer mevcut çalışma dizininiz `/home/project/lab1_template` değilse, proje klasörüne `cd` komutunu kullanarak geçin.

  ```
  cd /home/project/lab1_template
  ```

- Öncelikle, `orm` uygulaması için göç betikleri oluşturmanız gerekecek.

  ```
  python3 manage.py makemigrations orm
  ```

ve terminalde aşağıdaki sonucu görmelisiniz:

```
    Migrations for 'orm':
      orm/migrations/0001_initial.py
        - Create model User
```

- `orm/migrations` klasörü, Django'nun modellerinizdeki değişiklikleri sakladığı yerdir ve Django'nun model göçleriniz için hangi SQL ifadelerini oluşturduğunu merak edebilirsiniz.

SQL ifadelerini kontrol etmek için şunu çalıştırabilirsiniz:

```
python3 manage.py sqlmigrate orm 0001
```

`orm_user` tablosunun oluşturulma SQL ifadesini sizin için yazdırır.

```
BEGIN;
--
-- Create model User
--
CREATE TABLE "orm_user" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "first_name" varchar(30) NOT NULL, "last_name" varchar(30) NOT NULL, "
COMMIT;
```

Çoğu durumda, Django Model ORM bileşeni olarak çalıştığında, SQL kısmı hakkında endişelenmenize gerek yoktur. Veritabanlarında veri sorgulamak/manipüle etmek için sağlanan Django Model API'lerini kullanabilirsiniz.

- Sonra, `orm_user` tablosunu oluşturmak için göç işlemini gerçekleştirebilirsiniz:

```
python3 manage.py migrate
```

Django, `orm` uygulaması da dahil olmak üzere, yüklü olan tüm uygulamalar için göç işlemleri gerçekleştirecektir.

```
Operations to perform:
  Apply all migrations: orm
Running migrations:
  Applying orm.0001_initial... OK
```

- Sol taraftaki **Skills Network** butonuna tıklayarak **Skills Network Toolbox**'u açın. Ardından **Database**'e, ardından **PostgreSQL**'e tıklayın ve sonra **pgAdmin**'e tıklayın. Tarayıcınızda açılacaktır.
  Not: Devam etmeden önce PostgreSQL'in Aktif durumda olduğundan emin olun.

- `pgAdmin` yeni bir tarayıcı sekmesinde başlatıldığında, `Servers`'ı seçin, 1. Adımda oluşturulan şifreyi girin ve `OK`'e tıklayın.

- Ardından `Databases->postgres->Schemas->public->Tables`'ı genişletin, önceki göç adımında oluşturulan `orm_user` tablosunu görmelisiniz.

Modeli Test Et

- `test.py` dosyasını açın ve bir mockup `user` nesnesini kaydetmek için `test_setup()` metodunu bulabilirsiniz ve kullanıcı nesnesinin başarılı bir şekilde kaydedilip kaydedilmediğini kontrol etmeye çalışın.
- Modelinizi test etmek için `test.py` dosyasını çalıştırabilirsiniz:

  ```
  python3 test.py
  ```

Görmeniz gereken bir mesaj olmalı,

> **Django Model kurulumu tamamlandı.**

İşte bu kadar, ilk Django ORM uygulamanızı ve ilk Django modelinizi kurdunuz.

Özet

Bu laboratuvarda, ilk Django projenizi ve uygulamanızı kurdunuz. Ayrıca, ilk Django modelinizi oluşturup test ettiniz.

Bir sonraki laboratuvarda, daha gerçek Django Modelleri oluşturmaya çalışacak ve bunlar üzerinde Oluşturma, Okuma, Güncelleme ve Silme (CRUD) işlemleri gerçekleştireceksiniz.

Author(s)

Yan Luo