Міністерство освіти і науки України

Харківський Національний економічний університет імені Семена Кузнеця

Лабораторна робота №7

з дисципліни “Програмування Інтернет”

за темою “Використання баз даних у веб-застосунках на базі фреймворку Django”

|  |  |
| --- | --- |
| Виконав: | студент 3 курсу  Радченко В. С. групи  6.04.121.012.16.1 факультету ЕІ |
| Прийняв: | доцент кафедри ІС к.т.н., с.н.с. Парфьонов Ю.Е. |
|  |  |

Харків 2019

Мета лабораторної роботи: набуття практичних навичок використання Django Database ORM. Удоскналення навичок роботи з інтегрованим середовищем Microsoft Visual Studio.

Завдання 1: використовуючи фреймворк Django, розробити «скелет» застосунку, призначенного для виконання CRUD – операцій з даними, що зберігаються в базі даних. Дані предменої області мають складатися не менш чим з двох сутностей, між якими є відношення «один до багатьох».

Завдання 2: розробити веб застосунок призначений для виконання CRUD – операцій з даними предметної області. Застосунок має бути побудовано відповідно до архітектурного шаблону Model-View-Template.

Код:

models.py

from django.db import models

from django.shortcuts import reverse

from django.utils.text import slugify

from time import time

def gen\_slug(s):

new\_slug = slugify(s)

return new\_slug + '-' + str(int(time()))

class Tracks(models.Model):

title = models.CharField(max\_length=150, db\_index=True)

slug = models.SlugField()

performer = models.CharField(max\_length=150, db\_index=True)

genres = models.ManyToManyField('Genre', blank=True, related\_name='tracks')

data\_pub = models.DateTimeField(auto\_now\_add=True)

def get\_absolute\_url(self):

return reverse('track\_detail\_url', kwargs={'slug': self.slug})

def get\_update\_url(self):

return reverse('track\_update\_url', kwargs={'slug': self.slug})

def get\_delete\_url(self):

return reverse('track\_delete\_url', kwargs={'slug': self.slug})

def save(self, \*args, \*\*kwargs):

if not self.id:

self.slug = gen\_slug(self.title)

super(Tracks, self).save(\*args, \*\*kwargs)

def \_\_str\_\_(self):

return self.title

class Meta:

verbose\_name\_plural = "Tracks"

class Genre(models.Model):

title = models.CharField(max\_length=30, unique=True)

slug = models.SlugField()

def get\_absolute\_url(self):

return reverse('genre\_detail\_url', kwargs={'slug': self.slug})

def get\_update\_url(self):

return reverse('genre\_update\_url', kwargs={'slug': self.slug})

def get\_delete\_url(self):

return reverse('genre\_delete\_url', kwargs={'slug': self.slug})

def save(self, \*args, \*\*kwargs):

if not self.id:

self.slug = slugify(self.title)

super(Genre, self).save(\*args, \*\*kwargs)

def \_\_str\_\_(self):

return '{}'.format(self.title)

class Meta:

verbose\_name\_plural = "Genres"

views.py

from django.shortcuts import render, get\_object\_or\_404, redirect, reverse

from .models import \*

from django.views.generic import View

from .utils import \*

from .forms import \*

def tracks\_list(request):

tracks = Tracks.objects.all().order\_by('-data\_pub')

genres = Genre.objects.all()[:10]

return render(request, 'tracks/index.html',

context={'tracks': tracks,

'genres': genres,

'manage\_panel': True,

'manage\_panel\_UD': False})

def top\_list(request):

tracks = Tracks.objects.all().order\_by('-data\_pub')[:2]

return render(request, 'tracks/home\_page.html',

context={'tracks': tracks,

'manage\_panel': True,

'manage\_panel\_UD': False})

class TrackDetail(DetailMixin, View):

model = Tracks

template = 'tracks/track\_detail.html'

class TrackCreate(ObjectCreateMixin, View):

model = TrackForm

template = 'tracks/track\_create.html'

class TrackUpdate(ObjectUpdateMixin, View):

model = Tracks

template = 'tracks/track\_update.html'

model\_form = TrackForm

class TrackDelete(ObjectDeleteMixin, View):

model = Tracks

template = 'tracks/track\_delete.html'

redirect\_url = 'tracks\_list\_url'

def genres\_list(request):

genres = Genre.objects.all()

return render(request, 'tracks/genres\_list.html',

context={'genres': genres,

'manage\_panel': True,

'manage\_panel\_UD': False})

class GenreDetail(DetailMixin, View):

model = Genre

template = 'tracks/genre\_detail.html'

class GenreCreate(ObjectCreateMixin, View):

model = GenreForm

template = 'tracks/genre\_create.html'

class GenreUpdate(ObjectUpdateMixin, View):

model = Genre

template = 'tracks/genre\_update.html'

model\_form = GenreForm

class GenreDelete(ObjectDeleteMixin, View):

model = Genre

template = 'tracks/genre\_delete.html'

redirect\_url = 'tracks\_list\_url'

urls.py

from django.urls import path

from .views import \*

urlpatterns = [

path('', top\_list, name='top\_list\_url'),

path('tracks/', tracks\_list, name='tracks\_list\_url'),

path('track/create/', TrackCreate.as\_view(), name='track\_create\_url'),

path('track/<slug>/', TrackDetail.as\_view(), name='track\_detail\_url'),

path('track/<slug>/update/', TrackUpdate.as\_view(), name='track\_update\_url'),

path('track/<slug>/delete/', TrackDelete.as\_view(), name='track\_delete\_url'),

path('genres/', genres\_list, name='genres\_list\_url'),

path('genre/create/', GenreCreate.as\_view(), name='genre\_create\_url'),

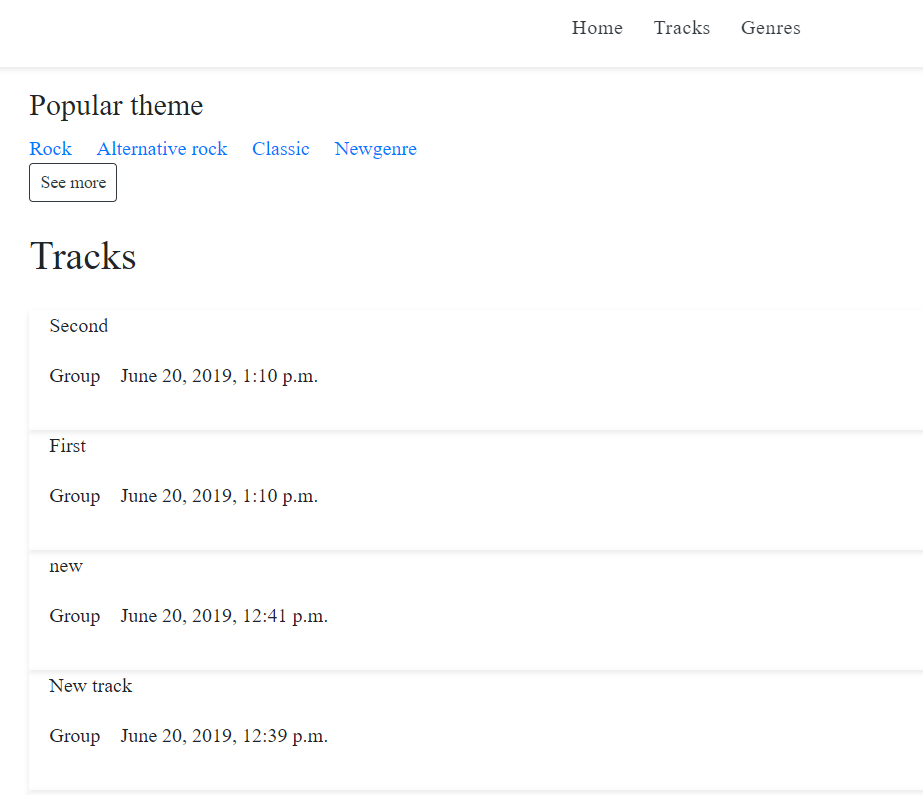
path('genre/<slug>/', GenreDetail.as\_view(), name='genre\_detail\_url'),

path('genre/<slug>/update/', GenreUpdate.as\_view(), name='genre\_update\_url'),

path('genre/<slug>/delete/', GenreDelete.as\_view(), name='genre\_delete\_url')

]

Приклад роботи програми:



ВИСНОВКИ

Під час виконання лабораторної роботи № 7 я отримала практичні навички використання Django Database ORM, а також удосконалила навички роботи з середовищем програмування Microsoft Visual Studio. Розроблені програми в основному відповідають постановці завдання.