**Отчёт по лабораторной работе №5 по курсу «Разработка интернет приложений»**

Тема: «Python. Работа с СУБД»

Выполнил: студент группы ИУ5-53

Андреев Алексей Андреевич

Дата: 24/12/2018 Подпись:\_\_\_\_\_\_\_

Проверил: Гапанюк Ю. Е.

Дата: \_\_\_\_\_\_ Подпись:\_\_\_\_\_\_\_

Москва, 2018

Задание и порядок выполнения

1. Скрипт с подключением к БД и несколькими запросами.
2. Набор классов вашей предметной области с привязкой к СУБД (класс должен уметь хотя бы получать нужные записи из БД и преобразовывать их в объекты этого класса)
3. Модели вашей предметной области
4. View для отображения списка ваших сущностей

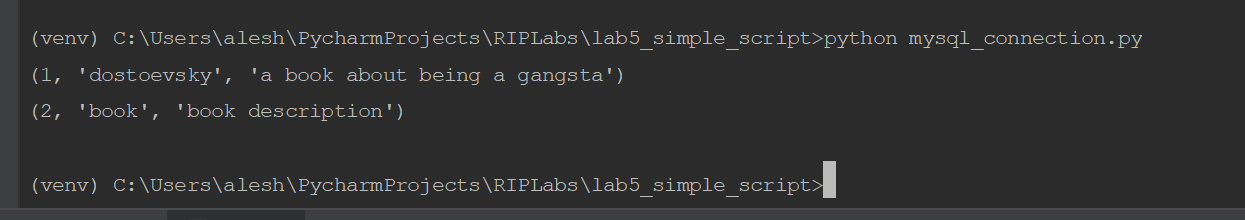
**Скрипт с подключением к БД и несколькими запросами**.

Исходный код

book\_example.py

import MySQLdb  
  
db = MySQLdb.connect(  
 host="localhost",  
 user="dbuser",  
 passwd="123",  
 db="exdb"  
)  
  
c = db.cursor();  
  
c.execute("INSERT INTO books (name, description) VALUES (%s, %s);", ('book', 'book description'))  
  
db.commit()  
  
c.execute("SELECT \* FROM books;")  
  
entries = c.fetchall()  
  
for e in entries:  
 print(e)  
  
c.close()  
db.close()

Результат работы



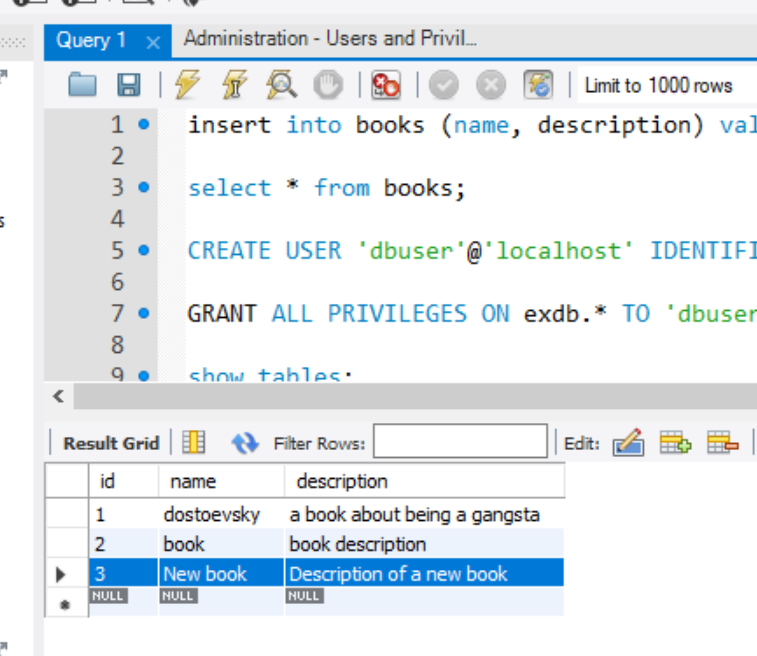
**Набор классов предметной области**.

Исходный код

book\_example.py

import MySQLdb  
  
  
class Connection:  
 def \_\_init\_\_(self, user, password, db, host='localhost'):  
 self.user = user  
 self.host = host  
 self.password = password  
 self.db = db  
 self.\_connection = None  
  
 @property  
 def connection(self):  
 return self.\_connection  
  
 def \_\_enter\_\_(self):  
 self.connect()  
  
 def \_\_exit\_\_(self, exc\_type, exc\_val, exc\_tb):  
 self.disconnect()  
  
 def connect(self):  
 if not self.\_connection:  
 self.\_connection = MySQLdb.connect(  
 host=self.host,  
 user=self.user,  
 passwd=self.password,  
 db=self.db  
 )  
  
 def disconnect(self):  
 if self.\_connection:  
 self.\_connection.close()  
  
  
class Book:  
 def \_\_init\_\_(self, db\_connection, name, description):  
 self.db\_connection = db\_connection.connection  
 self.name = name  
 self.description = description  
  
 def save(self):  
 c = self.db\_connection.cursor()  
 c.execute("INSERT INTO books (name, description) VALUES (%s, %s);",  
 (self.name, self.description))  
 self.db\_connection.commit()  
 c.close()  
  
  
con = Connection("dbuser", "123", "exdb")  
  
  
with con:  
 book = Book(con, "New book", "Description of a new book")  
 book.save()

Результат работы



**Модели моей предметной области**.

Исходный код

book\_example.py

from django.db import models  
  
  
class Programmer(models.Model):  
 username = models.CharField(max\_length=255, unique=True)  
 email = models.CharField(max\_length=255, unique=True, blank=True, null=True)  
 telephone = models.CharField(max\_length=12, unique=True, blank=True, null=True)  
 first\_name = models.CharField(max\_length=255)  
 last\_name = models.CharField(max\_length=255)  
 birthday = models.DateField(null=True, blank=True)  
 sex = models.CharField(max\_length=1, default='M')  
  
  
class Program(models.Model):  
 programName = models.CharField(max\_length=255, unique=True)  
 description = models.CharField(max\_length=255)  
 programmers = models.ManyToManyField(Programmer)  
  
  
class Site(models.Model):  
 url = models.CharField(max\_length=255, unique=True)  
 siteName = models.CharField(max\_length=255, blank=True)  
 programmers = models.ManyToManyField(Programmer)  
  
  
class Screenshot(models.Model):  
 filename = models.CharField(max\_length=255)  
 program = models.ForeignKey(Program, on\_delete=models.CASCADE)  
  
  
class Techfield(models.Model):  
 naming = models.CharField(max\_length=255)  
  
  
class Project(models.Model):  
 projectName = models.CharField(max\_length=255, unique=True)  
 description = models.CharField(max\_length=255)  
 techfield = models.ForeignKey(Techfield, on\_delete=models.CASCADE)  
 programs = models.ManyToManyField(Program)  
 site = models.ForeignKey(Site, on\_delete=models.CASCADE)  
  
  
class Milestone(models.Model):  
 date = models.DateField()  
 project = models.ForeignKey(Project, on\_delete=models.CASCADE)

**View для отображения списка моих сущностей**.

class IndexView(View):  
 def get(self, request):  
 return render(request, 'index.html')  
  
  
def getSubjectAreaInfo():  
  
 programmers = []  
 for programmer in Programmer.objects.all():  
 programmerinfo = {}  
 programmerinfo['firstname'] = programmer.first\_name  
 programmerinfo['username'] = programmer.username  
 programmerinfo['lastname'] = programmer.last\_name  
 programmerinfo['programs'] = []  
 for program in Program.objects.filter(programmers=programmer):  
 programmerinfo['programs'].append(program.programName)  
 programmers.append(programmerinfo)  
  
 subjectAreaDict = {'programmers' : programmers}  
 return subjectAreaDict  
  
class ModelView(View):  
 def get(self, request):  
 return render(request, 'myModels.html', getSubjectAreaInfo())

Результат работы

