Федеральное агентство связи

Орден Трудового Красного Знамени федерального государственного бюджетного образовательного учреждения высшего образования «Московский технический университет связи и информатики»

Кафедра «Математической кибернетики и информационных технологий»

Лабораторные работы по дисциплине «Кроссплатформенные технологии программирования»

Лабораторная работа №3

«Исследование административного интерфейса django»

Выполнил студент:

Группы БСТ1701

Ли Эдуард

Москва 2019

**Среда разработки: Visual Studio Code**

**Задание №1:**

Напишите функцию фильтрации студентов по средней оценке (так чтобы функция возвращала всех студентов выше заданного в параметрах функции среднего балла). Примерная схема работы функции: создание пустого массива, куда будут добавляться все студенты, прошедшие фильтрацию; запуск цикла, в каждой итерации которого необходимо считать среднюю оценку текущего студента и сравнивать с тем значением, что передано в качестве параметра.

Пример работы программы (Рисунок 1):

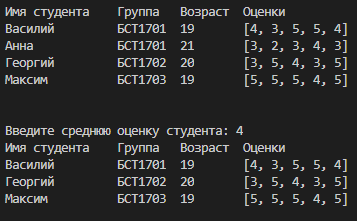


Рисунок 1 – Результат работы первой программы

Код программы №1:

#coding:utf-8

groupmates = [

{

"name": u"Василий",

"group": "БСТ1701",

"age": 19,

"marks": [4, 3, 5, 5, 4]

},

{

"name": u"Анна",

"group": "БСТ1701",

"age": 21,

"marks": [3, 2, 3, 4, 3]

},

{

"name": u"Георгий",

"group": "БСТ1702",

"age": 20,

"marks": [3, 5, 4, 3, 5]

},

{

"name": u"Максим",

"group": "БСТ1703",

"age": 19,

"marks": [5, 5, 5, 4, 5]

}

]

def print\_students(students):

print (u"Имя студента".ljust(15), \

u"Группа".ljust(8), \

u"Возраст".ljust(8), \

u"Оценки".ljust(20))

for student in students:

print (student["name"].ljust(15), \

student["group"].ljust(8), \

str(student["age"]).ljust(8), \

str(student["marks"]).ljust(20))

print ("\n")

print\_students(groupmates)

ave = float(input("Введите среднюю оценку студента: "))

average = []

for student in groupmates:

if float(sum(student["marks"])) / len(student["marks"]) >= ave:

average.append(student)

print\_students(average)

**Задание №2:**

* Откройте файл settings.py через редактор исходного кода IDLE (он уже установлен вместе с интерпретатором Python, как его открыть, объяснялось выше). Внимательно изучите все существующие там переменные и переведите описание каждой из них. Зубрить их нет необходимости, но примерно представлять совокупность всех настроек по умолчанию вам нужно.
* В файле settings.py настройте два параметра для базы данных. В переменной-словаре DATABASES есть ключ default, по которому содержится ещё один внутренний словарь с ключами ENGINE и NAME. Установите следующие значения этим параметрам: ◦ ENGINE: 'django.db.backends.sqlite3' ◦ NAME: 'db\_admin\_learning
* Теперь для вашего проекта в качестве базы данных подключена sqlite3, а название новосозданной базы будет “db\_admin\_learning”.

Код выполнения в файле settings.py:

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.sqlite3',

'NAME': 'db\_admin\_learning',

}

}

**Задание №3:**

* изучите интерфейс административного приложения django;
* через интерфейс административного приложения создайте нового пользователя с правами суперюзера;
* через интерфейс административного приложения создайте нового пользователя без прав суперюзера;
* через интерфейс административного приложения «забаньте» одного из пользователей (сделайте пользователя «неактивным»).

Пример выполнения задания (Рисунки №2,3,4,5):

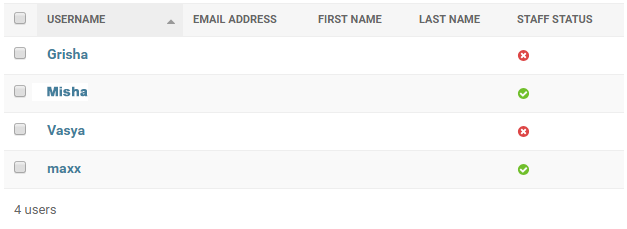


Рисунок 2 – Все пользователи

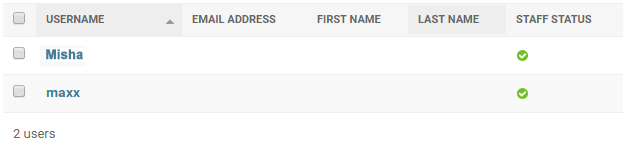


Рисунок – Сортировка по staff status

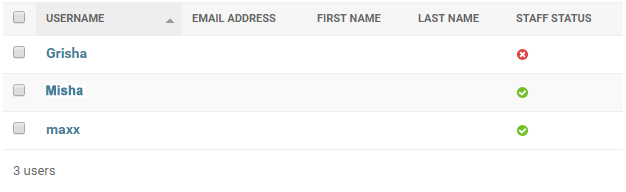


Рисунок – Сортировка по superuser status

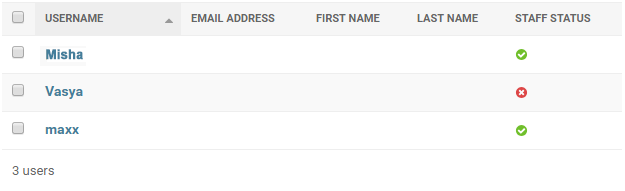


Рисунок – Сортировка по active users