МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

Лабораторна робота № 2

з дисципліни “Бази даних”

тема “ Створення додатку бази даних, орієнтованого на взаємодію з СУБД PostgreSQL”

Виконав

студент 2-го курсу

групи КП-93

Торговських Олександр Олегович
(*прізвище, ім’я, по батькові*)

Перевірив

“____” “_____” 20___ р.

Асистент

Замекула Олексій Ігорович
(*прізвище, ім’я, по батькові*)

Варіант: Студент в університеті
(студент, група, факультет)

Київ 2020

Метою роботи є здобуття вмінь програмування прикладних додатків баз даних PostgreSQL.

Завдання роботи полягає у наступному:

1. Реалізувати функції внесення, редагування та вилучення даних у таблицях бази даних, створених у лабораторній роботі №1, засобами консольного інтерфейсу.
2. Передбачити автоматичне пакетне генерування «рандомізованих» даних у базі.
3. Забезпечити реалізацію пошуку за декількома атрибутами з двох та більше сутностей одночасно: для числових атрибутів – у рамках діапазону, для рядкових – як шаблон функції LIKE оператора SELECT SQL, для логічного типу – значення True/False, для дат – у рамках діапазону дат.
4. Програмний код виконати згідно шаблону MVC (модель-подання-контролер).

URL репозиторію - https://github.com/alex123411/Data-Base

Пункт 1:

Помилка при додаванні. Користувач хотів додати студента який має групу КП-99 але так як в базі груп такої групи не існує

	group_name [PK] character varying (111)	fac_name character varying (111)	head_of_group character varying (70)	student_count integer
1	КП-91	ФПМ	Цой	22
2	КП-93	ФПМ	Антон	32

користувачу вивело повідомлення про помилковий ввід його даних

```
input student`s id  222
input grou where this student is КП-99
input his name and surname Олександр
input his birthdate 08 04 2002
Failed to insert record into student table, student with such id already exists or there is no such group
Sorry, you couldn`t add this student. Student with such id already exists or there is no such group
```

Помилка при видаленні. В даному випадку при видаленні студента користувач ввів id за яким такого студента не існує тому йому вивело повідомлення що такого студента за таким id не існує

```
delete hear
Input student`s id to delete student
12
Student with such ID doesn`t exist
u have this commands:
 1 - add
 2 - delete
 3 - edit
 4 - exit
 5 - generate random data
 6 - filtrationsn
select a command u wanna use
```

Видалення користувачем групи. При видаленні користувачем групи також видаляються усі студенти що входили і цю групу.

	student_id [PK] integer	group_name character varying (10)	name_surname character varying (70)	birthdate character varying (15)
1	221	КП-91	МАша	2002
2	2213	КП-93	Андрій	2001

Після того як користувач видалить групу КП-93

```
delete hear
Input goup name to delete group
КП-93
1 Record deleted successfully
1 Record deleted successfully
u have this commands:
 1 - add
 2 - delete
 3 - edit
 4 - exit
 5 - generate random data
 6 - filtrationsn
select a command u wanna use
```

Можна побачити ‘1 Record deleted successfully’ двічі, адже користувач видалив групу в якій був один студент. І стало в базі.

	student_id [PK] integer	group_name character varying (10)	name_surname character varying (70)	birthdate character varying (15)
1	221	КП-91	МАша	2002

На одного студента менше який був в групі КП-93 і

	group_name [PK] character varying (111)	fac_name character varying (111)	head_of_group character varying (70)	student_count integer
1	КП-91	ФПМ	Цой	22

на одну групу менше.
При видаленні цілого факультету таким же чином видаляються всі групи та всі студенти що були приєднані до цього факультету.

Валідація даних:

Користувач постарався додати студента з id яке не является числом тому йому вивело помилку про неможливість даної дії

```
u work in table of students
Add new student
input student`s id  qw
input grou where this student is qw
input his name and surname qw
input his birthdate qw
Please make sure that student`s id is a number and try again

u have this commands:
 1 - add
 2 - delete
 3 - edit
 4 - exit
 5 - generate random data
 6 - filtrationsn
select a command u wanna use
```

Пункт 2:

Генерація факультетів:

Запит користувача:

```
Choose in which table u wanna generate
 1 - faculty
 2 - group
1
input number of random rows
20000
```

Результат: Утворилося 20000 нових записів у таблиці факультетів

1	0006c994ec80519ca135c1006a7...	17100	075e288c23a8f1a022a58cd...
2	00084a75d2d3313f72dde6ffad11...	4092	ee3eebd8fde9a89b57e63b...
3	000ef3f2126a0a35354e6b77fa96...	7218	f884279c850ee51663449d9...
4	000ef6cebece22300d45e08fc089...	5819	b4e82c651b03b2d4351801...
5	0013ef0316dcc71600e23134829...	14059	8b871f25ae433067f97a6b8...
6	0018e667ae5225bd7386bed9b42...	1230	b324c66f58b3b5be93385c3...
7	0019b1b4387ec01570bb4240ae4...	2055	19527fb84955591d295cc02...
8	00211dd04683dfc664b11899f59...	12444	1fb879a2d64d66504ef5116...
9	0022606fe8f76b61a3366cf4659f...	6915	2b5c884af7f39f0c77d8eb00...
10	002610929e85ba946eae5e2a255...	12325	5d865bae446bb00b4e4965...
11	002c2ad734edc42d9b737a6d369...	538	6cd5484d2a2b40b34af7ffa7...
12	002ca0948cb175b280434dfa623...	3642	d0bc904b8cec91beefaf84f5...
13	002df41c5beef5fe7fd4e570cee5...	18433	60c9974a4525fe5ae5d1911...
14	0032b752ba5a3648999f49144e7...	15559	8715d704b77c3ea0052d5a...
15	0033efd0f89419112a5f780ad355...	6794	bb77b92587d482b39261b1...
16	003605e2d6deaf9aea065b234fe...	7037	e9376fbb56db022d1010e18...
17	00386136b9a0a9e5323bfe58334...	18048	6e483b2d659207d2279ff1d...
18	003b8abfe7cfe479774e19659a9...	11313	196b9738701fd17b4480007...
19	003ec52a5ae0b89457387e1dc5f...	7949	68fe82a06d6e9857c99706e...
20	003ee0cbd77f397d2bdcde7f9f81...	8287	d05b9eb4a1824b27bbcbdb...
21	0042ee65d1c7181c21cababbd4d...	10637	ec349facc8b172dbf013060...
22	00483b9884853570f40a5487b01...	4958	defea16259c0072efd341b5...
23	005811287f33dcc7e94797fcfe3c...	4064	8ee9e693d2ac2cbaad70998...
24	005de6105dee296ae19673d4fc8...	13186	6e83319b51062feddad0c3d...
25	0062bd5225c9cec6faf8f5de38dc...	16644	bbbca96c60a502019190cb...

✔ Запрос выполнен успешно. Общее время выполнения: 142 msec. обработано строк: 20001.

Генерація груп:

Запит користувача:

```
Choose in which table u wanna generate
 1 - faculty
 2 - group
2
input number of random rows
20000
u have this commands:
 1 - add
 2 - delete
 3 - edit
 4 - exit
 5 - generate random data
```

Результат: Утворилося 20000 нових записів у таблиці груп

Результат | План выполнения | Сообщения | Notifications

	group_name [PK] character varying (111)	fac_name character varying (111)	head_of_group character varying (70)	student_count integer
1	000c9581f34d9340d9b77f765d...	13a4e180a0db5131b10c8...	685071c1555d45e2bed56de41e...	19443
2	000f19e2b6711460a8730269d...	13a4e180a0db5131b10c8...	bec0a5d8d8fbd38e514bc6ca108...	11052
3	00118909ec4abc6a850e648f1...	13a4e180a0db5131b10c8...	1e51c57df7f0c6276b64aca7390...	15382
4	0013fbfe4a11735ddcb8165e1f...	13a4e180a0db5131b10c8...	064a0a39188da7268659ce1c36...	15073
5	0016c0c14f71527a306399d95...	13a4e180a0db5131b10c8...	830b40bca09b1af4495d5c3ba39...	10942
6	0016e036e85a1d19bf68c26e8...	13a4e180a0db5131b10c8...	f46956a6d3d3ed7e88c662a51da...	19054
7	0018b50717e061e6f42274a06...	13a4e180a0db5131b10c8...	3a1b920f2ba117f9447deb75f8d...	981
8	0018e3458cfd3643a59b99a7d...	13a4e180a0db5131b10c8...	602be4f4bb497325dc7e853e4c1...	13127
9	001958a0e0a660bd9c52cbc23...	13a4e180a0db5131b10c8...	e5ddd832dbafb15f21702605451...	6243
10	001a1ce5c88b451df0e82bbe5...	13a4e180a0db5131b10c8...	2d3801162f3cb2a93248d23fe84...	2122
11	001c6f5fa40fdec4f589265bac5...	13a4e180a0db5131b10c8...	f2e16f3f0ee83252be5d15f92240...	2302
12	001f292acaf7e47317300378fc...	13a4e180a0db5131b10c8...	0992004a5ec59787adfacafd0d4...	

✔ Запрос выполнен успешно. Общее время выполнения: 163 msec. обработано строк: 20001.

В даному випадку при утворенні випадкових даних для груп так як атрибут fac_name являється зовнішнім ключем то запит бере випадкові дані з таблиці факультетів для запису атрибуту fac_name(FK)

SQL-запиити:
для генерування факультетів:

```
INSERT INTO public.faculty  (fac_name, foundation_year, dean) SELECT MD5(random()::text),
trunc(random()*%s)::int, MD5(random()::text) FROM generate_series(1,%s)"""
```

для генерування груп:

```
INSERT INTO public.group  (group_name, fac_name, head_of_group, student_count)
SELECT MD5(random()::text), (SELECT fac_name  FROM public.faculty ORDER BY random() LIMIT 1) ,
MD5(random()::text), trunc(random()*%s)::int FROM generate_series(1,%s)"""
```

Пункт 3:

В даному консольному додатку реалізовано три запити на фільтрацію:
1)Показати усіх студентів на факультеті

```
INput a faculty u are interested in ФПМ

Execution time
0.005011320114135742
--------------
Id =  5
Name =  Вона
birthdate  =  2001
--------------
Id =  2
Name =  Я
birthdate  =  2002
--------------
Id =  6
Name =  Воно
birthdate  =  2003
--------------
Id =  3
Name =  Ти
birthdate  =  2001
u have this commands:
 1 - add
 2 - delete
 3 - edit
 4 - exit
 5 - generate random data
 6 - filtrationsn
select a command u wanna use
```

sql-запит:

```
query = """with x as(SELECT group_name from public.group WHERE fac_name = %s)
select student_id , name_surname, birthdate from public.student h inner join(select* from x) m on
h.group_name = m.group_name"""
```

```
1  with x as(SELECT group_name from public.group WHERE fac_name = 'ФПМ')
2          select student_id , name_surname, birthdate from public.student h inner join(select* from x) m on
3          h.group_name = m.group_name;
```

Результат | План выполнения | Сообщения | Notifications

	student_id [PK] integer	name_surname character varying (70)	birthdate character varying (15)
1	2	Я	2002
2	3	Ти	2001
3	5	Вона	2001
4	6	Воно	2003

2)Показати усіх студентів у кого кількість студентів в групі більша за якесь число

```
INput a number of students 20

Execution time
0.007997751235961914
--------------
Id =  2
Name =  Я
birthdate  =  2002
numb of students in this group  =  22
--------------
Id =  3
Name =  Ти
birthdate  =  2001
numb of students in this group  =  30
--------------
Id =  4
Name =  Вігн
birthdate  =  2003
numb of students in this group  =  40
--------------
Id =  5
Name =  Вона
birthdate  =  2001
numb of students in this group  =  22
--------------
Id =  6
Name =  Воно
birthdate  =  2003
numb of students in this group  =  30
u have this commands:
 1 - add
 2 - delete
 3 - edit
 4 - exit
 5 - generate random data
 6 - filtrationsn
select a command u wanna use
```

sql-запит:

```
query = """with x as(SELECT group_name, student_count from public.group WHERE student_count > %s)
select student_id , name_surname, birthdate , student_count from public.student
h inner join(select* from x) m on h.group_name = m.group_name"""
```

```
1  with x as(SELECT group_name, student_count from public.group WHERE student_count > 2)
2          select student_id , name_surname, birthdate , student_count from public.student h inner join(select* from x)
3          m on h.group_name = m.group_name
```

Результат | План выполнения | Сообщения | Notifications

	student_id integer	name_surname character varying (70)	birthdate character varying (15)	student_count integer
1	2	Я	2002	22
2	3	Ти	2001	30
3	4	Вігн	2003	40
4	5	Вона	2001	22
5	6	Воно	2003	30

3)Дізнатись який декан відноситься до даної групи

```
INput a group name to find its dean КП-93

Execution time
0.0029916763305664062
dean  =  ('Дичка',)
u have this commands:
 1 - add
 2 - delete
 3 - edit
 4 - exit
 5 - generate random data
 6 - filtrationsn
select a command u wanna use
```

sql-запит:

```
query = """with x as(SELECT fac_name  from public.group WHERE group_name = %s)
select    dean from public.faculty h inner join(select* from x) m on h.fac_name = m.fac_name"""
```

```
1  with x as(SELECT fac_name  from public.group WHERE group_name = 'КМ-99')
2  select    dean from public.faculty h inner join(select* from x) m on h.fac_name = m.fac_name
```

Результат | План выполнения | Сообщения | Notifications

	dean character varying (70)
1	НеДичка

КОД:

program.py

```
import controller
controller.start()
```

controller.py

```
import view
import model

def start():
    print("START!")
    x = view.MENU()
    if x == 0:
        print("Goodbye!")

def addfaculty(x,y,z):
    y = int(y)
    model.add_faculty(x,y,z)

def addgroup(x,y,z,f):
    f = int(f)
    model.add_group(x,y,z,f)

def addstudent(x,y,z,f):
    try:
        int(x)
    except ValueError:
        print("Please make sure that student`s id is a number and try
again\n")
        return

    y = model.add_student(x,y,z,f)
    return y

def deletefaculty(fac_name):
    model.deletefaculty(fac_name)


def deletegroup(group_name):
    model.deletegroup(group_name)

def deletestudent(student_name):
    count  = model.deletestudent(student_name)
    return count

def editfaculty(k, new_fac_name, foundationyear, dean):
    model.editfaculty(k, new_fac_name, foundationyear, dean)
```

```
def editgroup(k, new_group_name, fac_name, head_of_group,
student_count):
    model.editgroup(k, new_group_name, fac_name, head_of_group,
student_count)

def editstudent(id_, gr_name, name_surname, birthdate):
    model.editstudent(id_, gr_name, name_surname, birthdate)

def generate_in_faculty(num):
    model.generate_in_faculty(num)

def generate_in_group(num):
    model.generate_in_group(num)

def find_all_students_on_faculty(fac_name):
    records = model.find_all_students_on_faculty(fac_name)
    return records

def find_all_students_whoose_group_count_is_more_then(num):
    records =
model.find_all_students_whoose_group_count_is_more_then(num)
    return records

def who_is_a_dean_4_yhis_group(group_name):
    records = model.who_is_a_dean_4_yhis_group(group_name)
    return records
```

model.py

```
import time
import psycopg2

def add_faculty(x,y,z):
    connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")
    cursor = connection.cursor()
    postgres_insert_query = """ INSERT INTO faculty (fac_name,
foundation_year, dean) VALUES (%s,%s,%s)"""
    try:
        record_to_insert = (x, y, z)
```

```
        cursor.execute(postgres_insert_query, record_to_insert)
        connection.commit()
        count = cursor.rowcount
        print (count, "Record inserted successfully into faculties
table\n")
    except (Exception, psycopg2.Error) as error :
        if(connection):
            print("Failed to insert record into faculties table, this
faculty already exists")

    cursor.close()
    connection.close()

def add_group(x,y,z,f):
    connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")
    cursor = connection.cursor()
    try:
        postgres_insert_query = """ INSERT INTO public.group
(group_name, fac_name, head_of_group, student_count) VALUES
(%s,%s,%s,%s)"""
        record_to_insert = (x, y, z, f)
        cursor.execute(postgres_insert_query, record_to_insert)
        connection.commit()
        count = cursor.rowcount
        print (count, "Record inserted successfully into groups
table\n")
    except (Exception, psycopg2.Error) as error :
        if(connection):
            print("Failed to insert record into groupss table, group
with such name already exists or there is no such faculty" , error)
    cursor.close()
    connection.close()

def add_student(x,y,z,f):
    connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")
    cursor = connection.cursor()
    postgres_insert_query = """ INSERT INTO public.student
(student_id, group_name, name_surname, birthdate) VALUES
(%s,%s,%s,%s)"""
    try:
```

```
        record_to_insert = (x, y, z, f)
        cursor.execute(postgres_insert_query, record_to_insert)
        connection.commit()
        count = cursor.rowcount
        print (count, "Record inserted successfully into students
table\n")
    except (Exception, psycopg2.Error) as error :
        if(connection):
            print("Failed to insert record into student table, student
with such id already exists or there is no such group")
            cursor.close()
            connection.close()
            return 0
    cursor.close()
    connection.close()
    return 1

def show_faculties():
    connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")
    cur = connection.cursor()
    cur.execute("SELECT fac_name, foundation_year, dean FROM
public.faculty")
    rows = cur.fetchall()
    print("faculty name    foundation year    Dean ")
    for row in rows:
        print(row[0] +' '+ str(row[1])+ ' ' + row[2])
    cur.close()
    connection.close()

def show_students():
    connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")
    cur = connection.cursor()
    cur.execute("SELECT student_id, group_name, name_surname,
birthdate FROM public.student")
    rows = cur.fetchall()
    print("student_id    group_name  name_surname   birthdate ")
    for row in rows:
        print( str(row[0]) +' '+ str(row[1])+ ' ' + row[2]+ ' ' +
str(row[3]))
    cur.close()
```

```
    connection.close()

def show_groups():
    connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")
    cur = connection.cursor()
    cur.execute("SELECT group_name, fac_name, head_of_group,
student_count FROM public.group")
    rows = cur.fetchall()
    print("group_name   its faculty    head_of_group     student_count
")
    for row in rows:
        print(row[0] +' '+ str(row[1])+ ' ' + row[2]+ ' ' +
str(row[3]))

    cur.close()
    connection.close()

def deletefaculty(fac_name):
    try:
        connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")

        cursor = connection.cursor()

        #deleting students
        sql_sel_query = """SELECT DISTINCT group_name FROM
public.group WHERE fac_name = %s"""
        cursor.execute(sql_sel_query, (fac_name, ))
        rows = cursor.fetchall()

        for row in rows:
            sql_delete_query = """Delete from student where group_name
= %s"""
            cursor.execute(sql_delete_query, (row[0], ))
            connection.commit()
            count = cursor.rowcount
            print(count, "Records were deleted ")

        #deleting groups
        sql_delete_query = """Delete from public.group where fac_name
= %s"""
```

```
        cursor.execute(sql_delete_query, (fac_name, ))
        connection.commit()
        count = cursor.rowcount
        print(count, "Records were deleted ")

        # Update single record now
        sql_delete_query = """Delete from public.faculty where
fac_name = %s"""
        cursor.execute(sql_delete_query, (fac_name, ))
        connection.commit()
        count = cursor.rowcount
        print(count, "Records were deleted ")

    except (Exception, psycopg2.Error) as error:
        print("Error in Delete operation", error)

    finally:
        # closing database connection.
        if (connection):
            cursor.close()
            connection.close()

def deletestudent(student_name):
    try:
        connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")

        cursor = connection.cursor()

        # Update single record now
        sql_delete_query = """Delete from public.student where
student_id = %s"""
        cursor.execute(sql_delete_query, (student_name, ))
        connection.commit()
        count = cursor.rowcount
        return count
    except (Exception, psycopg2.Error) as error:
        print("Error in Delete operation", error)

    finally:
        # closing database connection.
        if (connection):
```

```
            cursor.close()
            connection.close()

def deletegroup(group_name):
    try:
        connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")

        cursor = connection.cursor()
        sql_delete_query = """Delete from public.student where
group_name = %s"""
        cursor.execute(sql_delete_query, (group_name, ))
        connection.commit()
        count = cursor.rowcount

        # Update single record now
        sql_delete_query = """Delete from public.group where
group_name = %s"""
        cursor.execute(sql_delete_query, (group_name, ))
        connection.commit()
        count = cursor.rowcount


    except (Exception, psycopg2.Error) as error:
        print("Error in Delete operation", error)

    finally:
        # closing database connection.
        if (connection):
            cursor.close()
            connection.close()




def editstudent(id_, gr_name, name_surname, birthdate):
    try:
        connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")

        cursor = connection.cursor()
        # Update multiple records
```

```
        sql_update_query = """Update public.student set group_name =
%s, name_surname = %s, birthdate = %s where student_id = %s"""
        cursor.execute(sql_update_query, (gr_name, name_surname,
birthdate, id_))
        connection.commit()

        row_count = cursor.rowcount
        print(row_count, "Records Updated")

    except (Exception, psycopg2.Error) as error:
        print("Error while updating PostgreSQL table", error)

    finally:
        # closing database connection.
        if (connection):
            cursor.close()
            connection.close()


def editgroup(group_name, new_group_name, fac_name, head_of_group,
student_count):
    try:
        connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")

        cursor = connection.cursor()

        #create new group with new_group_name
        postgres_insert_query = """ INSERT INTO public.group
(group_name, fac_name, head_of_group, student_count) VALUES
(%s,%s,%s,%s)"""
        record_to_insert = (new_group_name, fac_name, head_of_group,
student_count)
        cursor.execute(postgres_insert_query, record_to_insert)
        connection.commit()

        #update all students where this grop is
        sql_update_query = """ Update public.student set group_name =
%s WHERE group_name = %s"""
        cursor.execute(sql_update_query, (new_group_name, group_name))
        connection.commit()
```

```
        #delete created group
        sql_delete_query = """Delete from public.group where
group_name = %s"""
        cursor.execute(sql_delete_query, (group_name, ))
        connection.commit()

    except (Exception, psycopg2.Error) as error:
        print("Error while updating PostgreSQL table", error)

    finally:
        # closing database connection.
        if (connection):
            cursor.close()
            connection.close()


def editfaculty(faculty_name, new_fac_name, foundationyear, dean):
    try:
        connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")

        cursor = connection.cursor()

        #create new faculty with new_fac_name
        postgres_insert_query = """ INSERT INTO public.faculty
(fac_name, foundation_year, dean) VALUES (%s,%s,%s)"""
        record_to_insert = (new_fac_name, foundationyear, dean)
        cursor.execute(postgres_insert_query, record_to_insert)
        connection.commit()

        #update all groups where this faculty is
        sql_update_query = """ Update public.group set fac_name = %s
WHERE fac_name = %s"""
        cursor.execute(sql_update_query, (new_fac_name, faculty_name))
        connection.commit()

        #delete faculty
        sql_delete_query = """Delete from public.faculty where
fac_name = %s"""
        cursor.execute(sql_delete_query, (faculty_name, ))
        connection.commit()
```

```
    except (Exception, psycopg2.Error) as error:
        print("Error while updating PostgreSQL table", error)

    finally:
        # closing database connection.
        if (connection):
            cursor.close()
            connection.close()




def generate_in_faculty(number):
    try:
        connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")

        cursor = connection.cursor()

        postgres_insert_query = """
        INSERT INTO public.faculty  (fac_name, foundation_year, dean)
SELECT MD5(random()::text), trunc(random()*%s)::int,
MD5(random()::text) FROM generate_series(1,%s)"""
        cursor.execute(postgres_insert_query, (number,number))
        connection.commit()
    except (Exception, psycopg2.Error) as error:
        print("Error while updating PostgreSQL table", error)

    finally:
        # closing database connection.
        if (connection):
            cursor.close()
            connection.close()


def generate_in_group(number):
    try:
        connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")

        cursor = connection.cursor()
```

```
        postgres_insert_query = """
        INSERT INTO public.group  (group_name, fac_name,
head_of_group, student_count)
        SELECT MD5(random()::text), (SELECT fac_name  FROM
public.faculty ORDER BY random() LIMIT 1) , MD5(random()::text),
trunc(random()*%s)::int FROM generate_series(1,%s)"""
        cursor.execute(postgres_insert_query, (number,number))
        connection.commit()
    except (Exception, psycopg2.Error) as error:
        print("Error while updating PostgreSQL table", error)

    finally:
        # closing database connection.
        if (connection):
            cursor.close()
            connection.close()

def find_all_students_on_faculty(s):
    try:
        print(" ")
        connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")

        cursor = connection.cursor()
        query = """with x as(SELECT group_name from public.group WHERE
fac_name = %s)
        select student_id , name_surname, birthdate from
public.student h inner join(select* from x) m on
        h.group_name = m.group_name"""
        start = time.time()
        cursor.execute(query, (s, ))
        connection.commit()
        recoreds = cursor.fetchall()
        finish = time.time()
        print ("Execution time ")
        print(finish-start)
        return recoreds
    except (Exception, psycopg2.Error) as error:
        print("Error while updating PostgreSQL table", error)
    finally:
        # closing database connection.
        if (connection):
```

```
            cursor.close()
            connection.close()

def find_all_students_whoose_group_count_is_more_then(i):
    try:
        print(" ")
        connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")
        cursor = connection.cursor()
        query = """with x as(SELECT group_name, student_count from
public.group WHERE student_count > %s)
        select student_id , name_surname, birthdate , student_count
from public.student h inner join(select* from x) m on h.group_name =
m.group_name"""
        start = time.time()
        cursor.execute(query, (i, ))
        connection.commit()
        recoreds = cursor.fetchall()
        finish = time.time()
        print ("Execution time ")
        print(finish-start)
        return recoreds
    except (Exception, psycopg2.Error) as error:
        print("Error while updating PostgreSQL table", error)
    finally:
        # closing database connection.
        if (connection):
            cursor.close()
            connection.close()

def who_is_a_dean_4_yhis_group(name):
    try:
        print(" ")
        connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")
        cursor = connection.cursor()
        query = """with x as(SELECT fac_name  from public.group WHERE
group_name = %s)
        select    dean from public.faculty h inner join(select* from
x) m on h.fac_name = m.fac_name"""
        start = time.time()
        cursor.execute(query, (name, ))
```

```
        connection.commit()
        recoreds = cursor.fetchall()
        finish = time.time()
        print ("Execution time ")
        print(finish-start)
        return recoreds
    except (Exception, psycopg2.Error) as error:
        print("Error while updating PostgreSQL table", error)
    finally:
        # closing database connection.
        if (connection):
            cursor.close()
            connection.close()
```

view.py

```
import psycopg2
import os
import controller

def MENU():
    connection = psycopg2.connect(host="localhost", port = 5432,
database="Lab1", user="postgres", password="1234")
    cursor = connection.cursor()

    while True:
        i = input("u have this commands:\n 1 - add\n 2 - delete\n 3 -
edit\n 4 - exit \n 5 - generate random data\n 6 - filtrationsn\nselect
a command u wanna use\n")
        i = int(i)
        os.system('cls||clear')
        if i == 1:
            l = input("u can add smthing in this tables:\n 1 -
faculties\n 2 - groups\n 3 - students\n 4 - cancel \nchoose the
table\n ")
            os.system('cls||clear')
            l = int(l)
            if l == 1:
                x = input("input faculty`s name ")
                y = input("input its foundation year ")
```

```
                z = input("input its dean name ")
                controller.addfaculty(x,y,z)

            if l == 2:
                print("u work in table of groups\nu already have
this\n")
                print("\n")
                x = input("input group`s name ")
                y = input("input its faculty name ")
                z = input("input its head name ")
                f = input("input count of students in this group ")
                controller.addgroup(x,y,z,f)

            if l == 3:
                print("u work in table of students\nAdd new student")
                x = input("input student`s id  ")
                y = input("input grou where this student is ")
                z = input("input his name and surname ")
                f = input("input his birthdate ")
                vv = controller.addstudent(x,y,z,f)
                if vv == 0:
                    print("Sorry, you couldn`t add this student.
Student with such id already exists or there is no such group\n")
            if l == 4:
                break
        if i == 2:
            v = input("u can delete smthing in this tables:\n 1 -
faculties\n 2 - groups\n 3 - students \n 4 - cancel\n")
            os.system('cls||clear')
            v = int(v)
            if v == 1:
                print("delete hear")
                k = input("Input name of faculty ot delete\n")
                controller.deletefaculty(k)
            if v == 2:
                print("delete hear")
                k = input("Input goup name to delete group\n")
                controller.deletegroup(k)
            if v == 3:
                print("delete hear")
                k = input("Input student`s id to delete student\n")
                count = controller.deletestudent(k)
```

```
                if count == 0:
                    print ("Student with such ID doesn`t exist")
            if v == 4:
                break


        if i == 3:

            b = input("\nu can edit smthing in this tables:\n 1 -
faculties\n 2 - groups\n 3 - students \n 4 - cancel\n")
            os.system('cls||clear')
            b = int(b)
            if b == 1:
                k = input("Choose name of faculty ot edit\n")
                new_fac_name = input("INput new name for this
faculty\n")
                foundationyear = input("input new foundation year\n")
                foundationyear= int(foundationyear)
                dean = input("input new dean`s name\n")
                controller.editfaculty(k, new_fac_name,
foundationyear, dean)
            if b == 2:
                k = input("Choose group name to edit group\n")
                new_group_name = input("Choose new group name\n")
                fac_name = input("input new faculty for this group\n")
                head_of_group = input("Input new head of this
group\n")
                student_count = input("Input new count of students\n")
                student_count = int(student_count)
                controller.editgroup(k, new_group_name, fac_name,
head_of_group, student_count)
            if b == 3:
                id_ = input("Choose student`s id to edit student\n")
                id_ = int(id_)
                gr_name = input("input new group name for this
student\n")
                gr_name = str(gr_name)
                name_surname = input("input new name and surname
student\n")
                name_surname = str(name_surname)
                birthdate = input("input new birthdate for this
student\n")
```

```
                birthdate = str(birthdate)
                controller.editstudent(id_, gr_name, name_surname,
birthdate)
            if b == 4:
                break

        if i == 4:
            return 0

        if i == 5:
            dd = input("Choose in which table u wanna generate\n 1 -
faculty\n 2 - group\n")
            dd = int(dd)

            if dd == 1:
                lp = input("input number of random rows\n")
                controller.generate_in_faculty(lp)
            if dd == 2:
                lp = input("input number of random rows\n")
                controller.generate_in_group(lp)
        if i == 6:
            cc = input("Choose which filtrarion u wanna use\n 1 - find
all students on this faculty\n 2 - find all students whoose group
count is more then\n 3 - find a dean for this group\n")
            cc = int(cc)
            os.system('cls||clear')
            if cc == 1:
                x = input("INput a faculty u are interested in ")
                records = controller.find_all_students_on_faculty(x)
                for row in records:
                    print("---------------")
                    print("Id = ", row[0], )
                    print("Name = ", row[1])
                    print("birthdate  = ", row[2])
            if cc == 2:
                x = input("INput a number of students ")
                records =
controller.find_all_students_whoose_group_count_is_more_then(x)
                for row in records:
                    print("---------------")
                    print("Id = ", row[0], )
                    print("Name = ", row[1])
```

```
                    print("birthdate  = ", row[2])
                    print("numb of students in this group  = ",
row[3])
            if cc == 3:
                x = input("INput a group name to find its dean ")
                records = controller.who_is_a_dean_4_yhis_group(x)


                print("dean  = ", records[0])
    cursor.close()
    connection.close()

    return 1
```