Министерство образования и науки Российской Федерации Федеральное агентство по образованию

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Вятский государственный университет»

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Связывание приложения на Python с базой данных под управлением PostgreSQL

Лабораторная работа №5

по курсу «Базы данных»

Выполнил студент группы ИВТб-31 /Категов А.Д/ Проверил преподаватель /Клюкин В.Л./

Киров 2024

**Цели лабораторной работы:**

1. Познакомиться c библиотекой в Python для связывания приложения с БД;
2. Изучить некоторые шаблоны проектирования, связанные с работой с БД;
3. Освоить на практике основы взаимодействия с БД под управлением PostgreSQL в приложении на Python.

**Задание на лабораторную работу:**

* Выводить строки выбранной таблицы;
* Предоставлять возможность добавить новую строку в таблицу;
* Предоставлять возможность удалить строку из таблицы.

**Ход работы:**

1. Реализация подключения к базе данных.

def connect(self, config):

try:

self.conn = psycopg2.connect(

dbname=config["db"]["dbname"],

host=config["db"]["host"],

user=config["db"]["user"],

password=config["db"]["password"],

port=config["db"]["port"]

)

except:

return False

return True

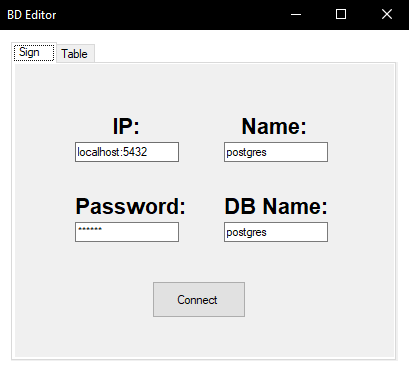


Рисунок 1 – Экранная форма

2. Класс управления базой данных

class DataBase:

def \_\_init\_\_(self):

self.conn = None

def connect(self, config):

try:

self.conn = psycopg2.connect(

dbname=config["db"]["dbname"],

host=config["db"]["host"],

user=config["db"]["user"],

password=config["db"]["password"],

port=config["db"]["port"]

)

except:

return False

return True

def select(self, query, values=None):

if self.conn:

data = []

try:

cursor = self.conn.cursor()

cursor.execute(query, values)

for row in cursor.fetchall():

data.append(row);

cursor.close()

except BaseException as e:

print(e)

return ([], False)

return (data, True)

def execute(self, query, data=None):

if self.conn:

try:

cursor = self.conn.cursor()

cursor.execute(query, data)

self.conn.commit()

cursor.close()

except:

return False

return True

def close(self):

if self.conn:

self.conn.close()

3. Функция добавления элемента

def add\_subscription(self):

self.view.window.error\_label["text"] = ""

name = self.view.window.form.name\_input.get().strip()

cost = self.view.window.form.cost\_input.get().strip()

num\_trainings = self.view.window.form.trainings\_input.get().strip()

gym = self.view.window.form.gyms.get().strip()

subscription = (name, cost, num\_trainings, gym)

if self.check\_subscription(subscription):

subscription = (name, cost, num\_trainings, self.gyms\_id[gym])

if self.model.add\_subscription(subscription):

self.clear\_table()

self.fill\_table()

self.show\_table()

else:

self.view.window.error\_label["text"] = "Ошибка при добавлении абонемента"

4. Функция удаления элемента

def delete\_subscription(self):

self.view.window.error\_label["text"] = ""

if askyesno(title="Подтвержение операции", message="Удалить запись?"):

for item in self.view.window.table\_area.table.get\_selection():

if not self.model.delete\_subscription(item[0]):

self.view.window.error\_label["text"] = "Ошибка при удалении"

break

self.clear\_table()

self.fill\_table()

**Вывод:** в ходе выполнения данной лабораторной работы были ознакомлены с библиотекой в Python для связывания приложения с БД. Изучены некоторые шаблоны проектирования, связанные с работой с БД. Освоены на практике основы взаимодействия с БД под управлением PostgreSQL в приложении на Python.