

Лабораторная работа №4: Создание системы авторизации в веб-приложении

1. Создание директории

- Открываем терминал
- Вводим следующие команды:

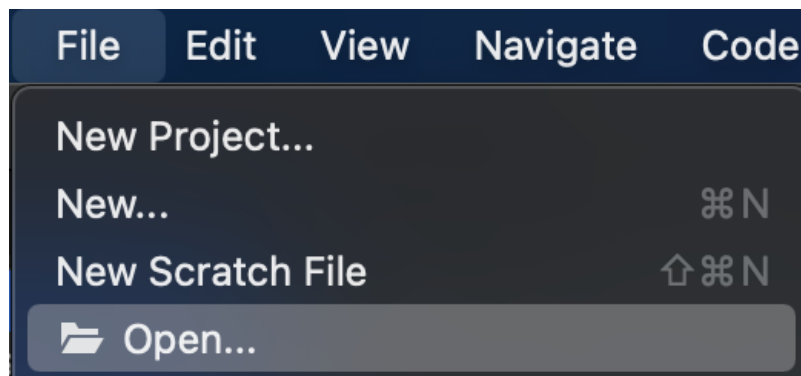
mkdir MyWebApp – создаем директорию с именем "MyWebApp"

cd MyWebApp – переходим в директорию

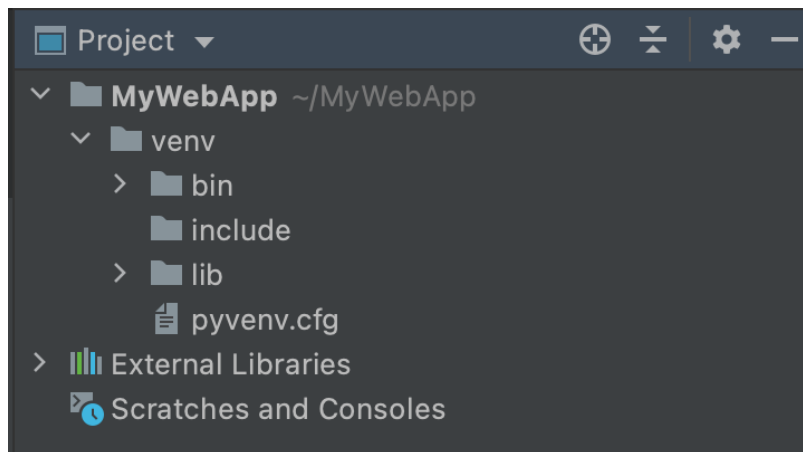
- После создания директории создаем и активируем виртуальную среду Python

1. Открываем директорию в PyCharm:

- Открываем нашу директорию:

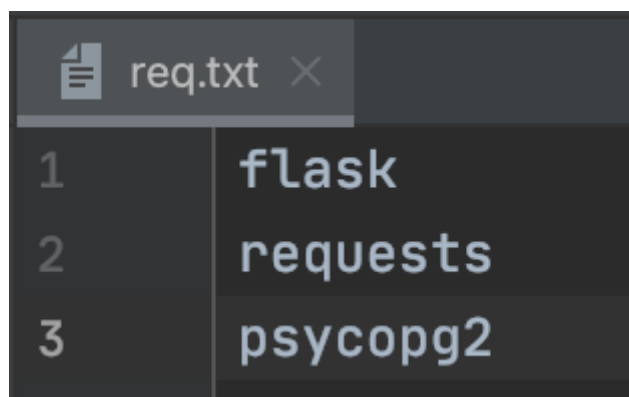


- Структура проекта на данном этапе выглядит следующим образом:



10. Устанавливаем необходимые инструменты для дальнейшей работы:

- Создаем файл req.txt
- Заполняем файл req.txt названиями необходимых нам инструментов



- Устанавливаем инструменты в терминале через менеджера пакетов Python pip3:

pip3 install -r req.txt

11. Создаем приложение:

- Создаем файл app.py
- Импортируем необходимые инструменты

```
import requests
from flask import Flask, render_template, request
import psycopg2
```

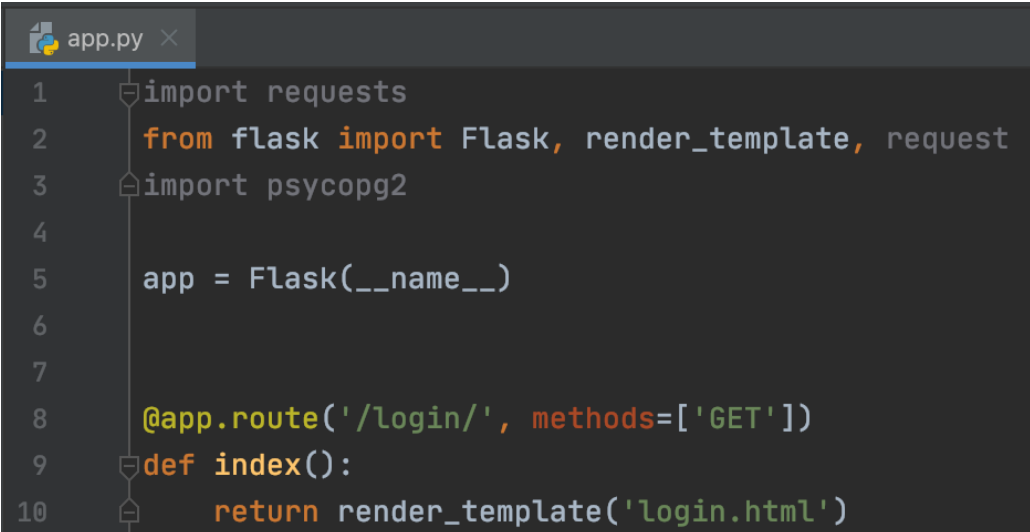
- Создаем приложение

```
app = Flask(__name__)
```

- Создаем первый декоратор

```
@app.route('/login/', methods=['GET'])
def index():
    return render_template('login.html')
```

На данном этапе файл app.py должен выглядеть следующим образом:



```
1 import requests
2 from flask import Flask, render_template, request
3 import psycpg2
4
5 app = Flask(__name__)
6
7
8 @app.route('/login/', methods=['GET'])
9 def index():
10     return render_template('login.html')
```

- Создаем директорию templates
- Внутри этой директории создаем файл login.html
- Удаляем содержимое файла login.html и вставляем текст разметки

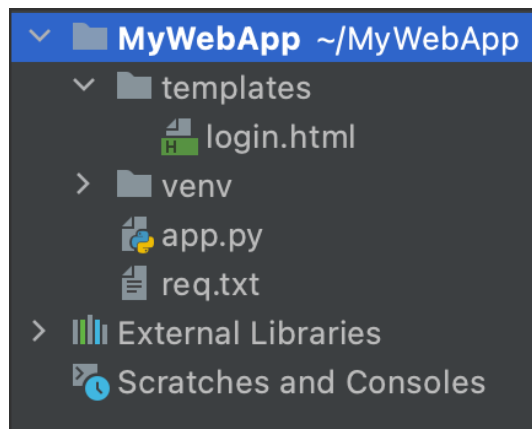
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Login</title>
</head>
<body>
    <form action="" method="post">
        <p>
            <label for="username">Username</label>
            <input type="text" name="username">
        </p>
        <p>
```

```
<label for="password">Password</label>
<input type="password" name="password">
</p>
<p>
  <input type="submit">
</p>
</form>

</body>
</html>
```

- Запускаем приложение

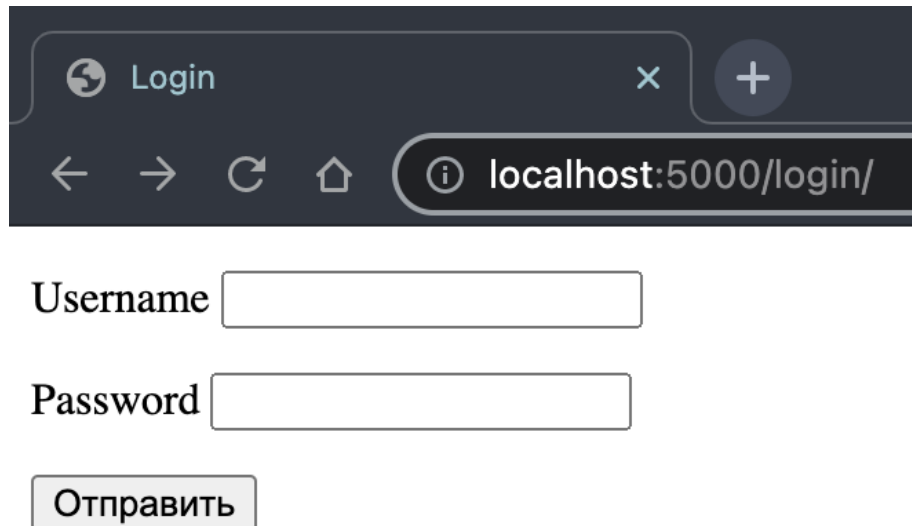
На данном этапе структура проекта выглядит следующим образом:



В терминале в папке MyWebApp запускаем команду

flask run

- Переходим по ссылке <http://localhost:5000/login/>



Username

Password

12. Устанавливаем PostgreSQL

- Обновляем список пакетов
→ **sudo apt update**
- Устанавливаем пакеты Postgres и contrib
→ **sudo apt install postgresql postgresql-contrib**
- Запускаем сервер PostgreSQL
→ **sudo -u postgres psql**

13. Создаем базу данных

- Создаем базу данных

CREATE DATABASE service_db;

- Подключаемся к базе данных

\c service_db

- Создаем схему

CREATE SCHEMA service;

- Создаем таблицу пользователей

CREATE TABLE service.users (id **SERIAL NOT NULL**, full_name **VARCHAR NOT NULL**, login **VARCHAR NOT NULL**, password **VARCHAR NOT NULL**);

- Заполняем таблицу пользователей

INSERT INTO service.users (full_name, login, password) **VALUES** ('<Полное имя пользователя>', '<логин>', '<пароль>');

- Проверяем заполнение таблицы

```
SELECT * FROM service.users;
```

```
id | full_name | login | password  
----+-----+-----+-----  
 1 | Ivanov Ivan | ivanov | 123456  
(1 row)
```

14. Модернизируем приложение:

- В файл app.py добавляем подключение к базе данных сразу после строки "app = Flask(__name__)"

```
conn = psycopg2.connect(database="service_db",  
    user="postgres",  
    password="пароль",  
    host="localhost",  
    port="5432")
```

-

- Добавляем курсор для обращения к базе данных

```
cursor = conn.cursor()
```

- Создаем еще один декоратор

```
@app.route('/login/', methods=['POST'])
def login():
    username = request.form.get('username')
    password = request.form.get('password')
    cursor.execute("SELECT * FROM service.users WHERE login=%s AND
password=%s", (str(username), str(password)))
    records = list(cursor.fetchall())

    return render_template('account.html', full_name=records[0][1])
```

- Создаем файл account.html

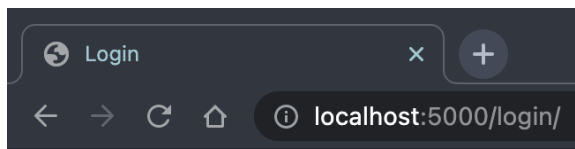
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <form action="" method="post">
        {% if full_name %}
        <p>Hello, {{full_name}}! </p>
        {% endif %}

        </p>

    </form>
</head>
<body>

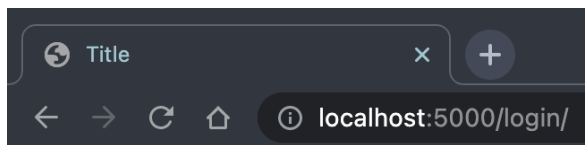
</body>
</html>
```

15. Посмотрим что получилось



Username

Password

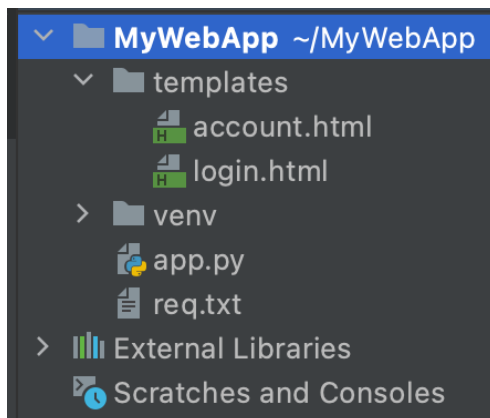


Hello, Ivanov Ivan!

16. Домашнее задание

- Дополнить таблицу users в базе данных до 10 пользователей
- Сделать обработку исключения на ввод пустого логина и пароля
- Сделать обработку исключения на отсутствие пользователя в базе данных
- Вывести на странице аккаунта помимо имени пользователя его логин и пароль

Структура проекта выглядит следующим образом:



Файл app.py выглядит следующим образом:

```
1 import requests
2 from flask import Flask, render_template, request
3 import psycopg2
4
5 app = Flask(__name__)
6
7 conn = psycopg2.connect(database="service_db",
8                         user="arshegor",
9                         password="",
10                        host="localhost",
11                        port="5432")
12
13 cursor = conn.cursor()
14
15
16 @app.route('/login/', methods=['GET'])
17 def index():
18     return render_template('login.html')
19
20
21 @app.route('/login/', methods=['POST'])
22 def login():
23     username = request.form.get('username') # запрос к данным формы
24     password = request.form.get('password')
25     cursor.execute("SELECT * FROM service.users WHERE login=%s AND password=%s", (str(username), str(password)))
26     records = list(cursor.fetchall())
27
28     return render_template('account.html', full_name=records[0][1])
```