

## Лабораторная работа №5: Создание системы регистрации в веб-приложении

---

1. В файле app.py импортируем метод "redirect", отвечающий за перенаправление на другой путь:

```
from flask import Flask, render_template, request, redirect
```

2. Удалите существующие декораторы из файла app.py, чтобы сократить объем программы:

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

3. Модернизируем декоратор для пути "/login/":

```
@app.route('/login/', methods=['POST', 'GET'])
def login():
    if request.method == 'POST':
        if request.form.get("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])
        elif request.form.get("registration"):
            return redirect("/registration/")

    return render_template('login.html')
```

4. В папке templates изменяем файл login.html, чтобы он выглядел следующим образом.

```
<!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" value="login" name="login">
    <input type="submit" value="registration" name="registration">
  </p>

</form>

</body>
</html>

```

**5. В папке templates создадим файл registration.html и вставим в него следующий код:**

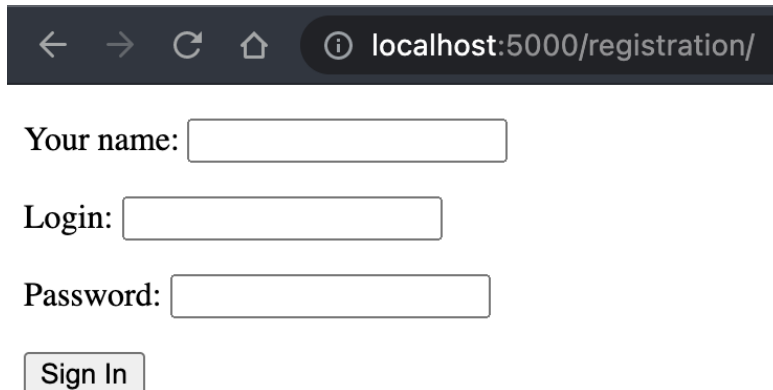
```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Registration</title>
</head>
<body>
  <form action="" method="post">
    <p>
      <label for="name">Your name:</label>
      <input type="text" name="name">
    </p>
    <p>
      <label for="login">Login:</label>
      <input type="text" name="login">
    </p>
    <p>
      <label for="password">Password:</label>
      <input type="password" name="password">
    </p>
    <p>
      <input type="submit" value="Sign In">
    </p>
  </form>

```

```
</body>
</html>
```

6. Запускаем приложение, на данном этапе оно должно выглядеть следующим образом:



The screenshot shows a web browser window with the address bar displaying 'localhost:5000/registration/'. Below the address bar, there is a registration form with the following elements:

- A label 'Your name:' followed by a text input field.
- A label 'Login:' followed by a text input field.
- A label 'Password:' followed by a text input field.
- A 'Sign In' button located below the password field.

7. Добавляем еще один декоратор, отвечающий за путь `"/registration/"`, в файл `app.py`:

```
@app.route('/registration/', methods=['POST', 'GET'])
def registration():
    if request.method == 'POST':
        name = request.form.get('name')
        login = request.form.get('login')
        password = request.form.get('password')

        cursor.execute('INSERT INTO service.users (full_name, login, password) VALUES
(%s, %s, %s);',
            (str(name), str(login), str(password)))
        conn.commit()

    return redirect('/login/')

return render_template('registration.html')
```

8. Проверяем таблицу `users` в базе данных на наличие нового пользователя в системе

**Домашнее задание:**

Реализовать обработку всех исключений при регистрации

## Лабораторная работа №6: Создание оконного приложения-калькулятора

---

1. **Создаем новый проект в PyCharm с названием Calculator**
2. **Устанавливаем библиотеку для создания оконных приложений PyQt5**  

```
pip3 install PyQt5
```
3. **В проекте создаем новый файл с названием calculator.py**
4. **Импортируем необходимые библиотеки для создания приложения**

```
import sys
from PyQt5.QtWidgets import QApplication, QWidget, QLineEdit, QHBoxLayout,
QVBoxLayout, QPushButton
```

Библиотека `sys` используется для получения информации об операционной системе.

Библиотека `PyQt5` используется для создания оконных приложений.

`PyQt5` используется в нашем проекте не полностью для уменьшения объема зависимостей приложения. Поэтому мы импортируем лишь некоторые классы из нее.

`QApplication` – управляет потоком управления и основными настройками приложения с графическим интерфейсом

`QWidget` – является базовым классом для всех объектов пользовательского интерфейса

`QLineEdit` – виджет, который разрешает вводить и редактировать одну строку текста

`QHBoxLayout` – выстраивает виджеты по горизонтали

`QVBoxLayout` – выстраивает виджеты по вертикали

`QPushButton` – кнопка, на которую можно нажимать

5. **Создаем класс `Calculator` и наследуем его от класса `QWidget`**

```
class Calculator(QWidget):
    def __init__(self):
```