

**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И
МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ
Ордена Трудового Красного Знамени федеральное государственное
бюджетное образовательное учреждение высшего образования
«Московский технический университет связи и информатики»**

Лабораторная работа №6
«Создание формы, шаблона и представления для регистрации и для
авторизации»
по дисциплине:
«Web-программирование»

Выполнил:
Студент группы БФИ2102
Шаинян С.А.
Проверил:
Фатхулин Т.Д.

Москва, 2023

Задание на лабораторную работу

1. В директорию lab6 скопируйте ваш прошлый проект blog.
2. Создайте шаблон и настройте адрес для отображения формы регистрации;
3. Создайте представление, которое будет обрабатывать поступающие запросы и регистрировать новых пользователей. Не забудьте сделать проверку на то, что отправленные поля не являются пустыми, а введенное имя пользователя уникально;
4. Создайте стили, подключив CSS-файл к шаблону;
5. Добавьте в шапку страницы всех записей и страницы для определенных статей, ссылку на регистрацию в верхнем правом углу (стиль ссылки сделать такой же, как у ссылки “Все статьи” на собственных страницах постов в предыдущих работах).
6. Создайте шаблон и настройте адрес для отображения формы авторизации;
7. Создайте представление, которое будет обрабатывать поступающие запросы и авторизовывать зарегистрированных пользователей. Не забудьте сделать проверку на то, что отправленные поля не являются пустыми, а введенные имя пользователя и пароль соответствуют одному из зарегистрированных аккаунтов;
8. Создайте стили, подключив CSS-файл к шаблону;
9. Загрузите ваш проект на любой гит-репозиторий (GitHub, GitLab, Google Code, Bitbucket и т.п.).

Ход лабораторной работы.

За основу был взят проект из лабораторной работы №5. Добавил в файл `urls.py` новые пути, которые продемонстрированы на рисунке 1.

```
from django.contrib import admin
from django.urls import path
from articles import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', views.archive, name="Articles"),
    path('article/<int:article_id>', views.get_article, name='get_article'),
    path('article/new/', views.create_post, name='create_post'),
    path('registration', views.registred, name='regPage'),
    path('logIn', views.logIn, name='logIn'),
    path('logOut', views.logoutFunc, name='logOut')
]
```

Рисунок 1 – Файл `urls.py`

Создадим в файле `views.py` новую функцию `registred`, которая должна будет отображать форму с регистрацией пользователя, и переводить его на форму авторизации в случае успешной регистрации. (рисунок 2)

```
def registred(request):
    if request.method == "POST":
        form = {
            'username': request.POST["username"],
            'email': request.POST["email"],
            'password': request.POST["password"]
        }
        if form["username"] and form["email"] and form["password"]:
            try:
                User.objects.get(username=request.POST["username"])
                form['errors'] = u"Пользователь с таким именем уже существует"
                return render(request, 'articles/registrationpage.html', {'form': form})
            except User.DoesNotExist:
                User.objects.create_user(username=request.POST["username"],
                                          email=request.POST["username"],
                                          password=request.POST["password"])
                return redirect('logIn')
        else:
            form['errors'] = u"Не все поля заполнены"
            return render(request, 'articles/registrationpage.html', {'form': form})
    else:
        return render(request, 'articles/registrationpage.html', {})
```

Рисунок 2 – Файл `views.py`

Далее создал файл registrationpage.html, программный код которого представлен на рисунке 3.

```
{% load static %}

<!DOCTYPE html>
<html lang="en">

<head>
  <link rel="stylesheet" href="{% static 'article.css' %}">
  <meta charset="UTF-8">
  <title>Регистрация</title>
</head>

<body>
  <div class="content">
    <h1>Страница регистрации</h1>
    <form method="POST">
      {% csrf_token %}
      <input type="text" name="username" placeholder="Имя пользователя" value="{{ form.title }}">
      <input type="text" name="email" placeholder="email пользователя" value="{{ form.title }}">
      <input type="password" name="password" placeholder="пароль" value="{{ form.title }}">
      <input type="submit" value="Сохранить">
    </form>
    {{ form.errors }}
  </div>
</body>

</html>
```

Рисунок 3 – Файл registrationpage.html

Затем добавил функцию login, которая будет выполнять авторизацию пользователя. (рисунок 4)

```
def login(request):
    if request.method == "POST":
        form = {
            'username': request.POST["username"],
            'password': request.POST["password"]
        }
        if form["username"] and form["password"]:
            user = authenticate(request, username=request.POST["username"], password=request.POST["password"])
            if user is not None:
                login(request, user)
                return redirect('home')
            else:
                form['errors'] = u"Введенный пользователь не существует"
                return render(request, 'signinpage.html', {'form': form})
        else:
            form['errors'] = u"Не все поля заполнены"
            return render(request, 'signinpage.html', {'form': form})
    else:
        return render(request, 'signinpage.html', {})
```

Рисунок 4 – Файл views.py

Далее создал файл signinpage.html, программный код которого представлен на рисунке 5.

```
{% load static %}

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="{% static 'article.css' %}">
    <title>Авторизация</title>
</head>

<body>
    <div class="content">
        <h1>Страница авторизации</h1>
        <form method="POST">
            {% csrf_token %}
            <input type="text" name="username" placeholder="Имя пользователя" value="{{ form.title }}">
            <input type="password" name="password" placeholder="Пароль" value="{{ form.title }}">
            <input type="submit" value="Сохранить">
        </form>
        {{ form.errors }}
    </div>
</body>

</html>
```

Рисунок 5 – Файл singInPage.html

На рисунке 6 представлен программный код файла article.css со стилями для страниц.

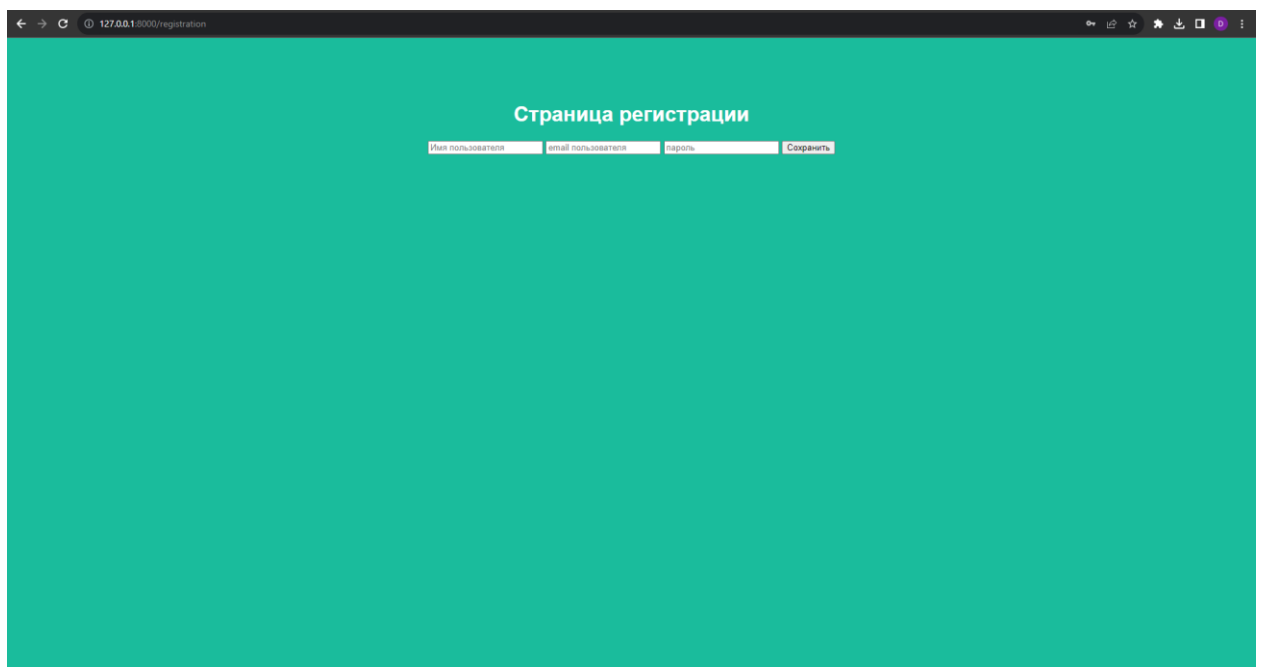
```

body{
  background: #1abc9c;
  font-family: Tahoma, Arial, sans-serif;
  color: #ffffff}
img {
  display: block;
  width: 318px;
  margin-left: auto;
  margin-right: auto;}
.archive {
  width: 960px;
  margin-left: auto;
  margin-right: auto;}
.post-title a {
  color: #ffffff;}
.article-author {
  width: 50%;
  float: left;}
.article-created-date {
  text-align: right;}
.article-image{
  display: block;
  width: 318px;
  margin-left: 0;}
.link {
  color: white;
  font-weight: bold;
  position: absolute;
  right: 470px;
  top: 180px;}
.article-border p{
  text-align: right;}
.article-text{
  width: 960px;
  text-align: justify;}
.article-created-data{
  text-align: right;}
.content{
  text-align: center;
  padding-top: 70px;}
input[name="title"]{
  padding: 5px;
  margin-bottom: 10px;
  border: 1px solid #888;
  outline: none;
  -moz-appearance: none;
  width: 200px;
  text-align: center;
  border-radius: 40px;}
textarea[name="text"]{
  padding: 25px;
  margin-bottom: 10px;
  border: 1px solid #888;
  outline: none;
  -moz-appearance: none;
  width: 650px;
  height: 350px;
  resize: none;
  border-radius: 40px;
  scrollbar-width: thin;}
.create{
  display: flex;
  flex-direction: column;
  align-items: center;}
.save_button{
  padding: 10px;
  width: 150px;
  background-color: white;
  border: none;
  border-radius: 40px;
  color: #1abc9c;
  font-weight: bold;
  letter-spacing: 0.06em;
  margin-top: 10px;}
.save_button:hover{
  color: white;
  background-color: #1abc9c;
  box-shadow: 1px 1px 10px 10px;
  transition-duration: 0.3s;}

```

Рисунок 6 – Файл article.css

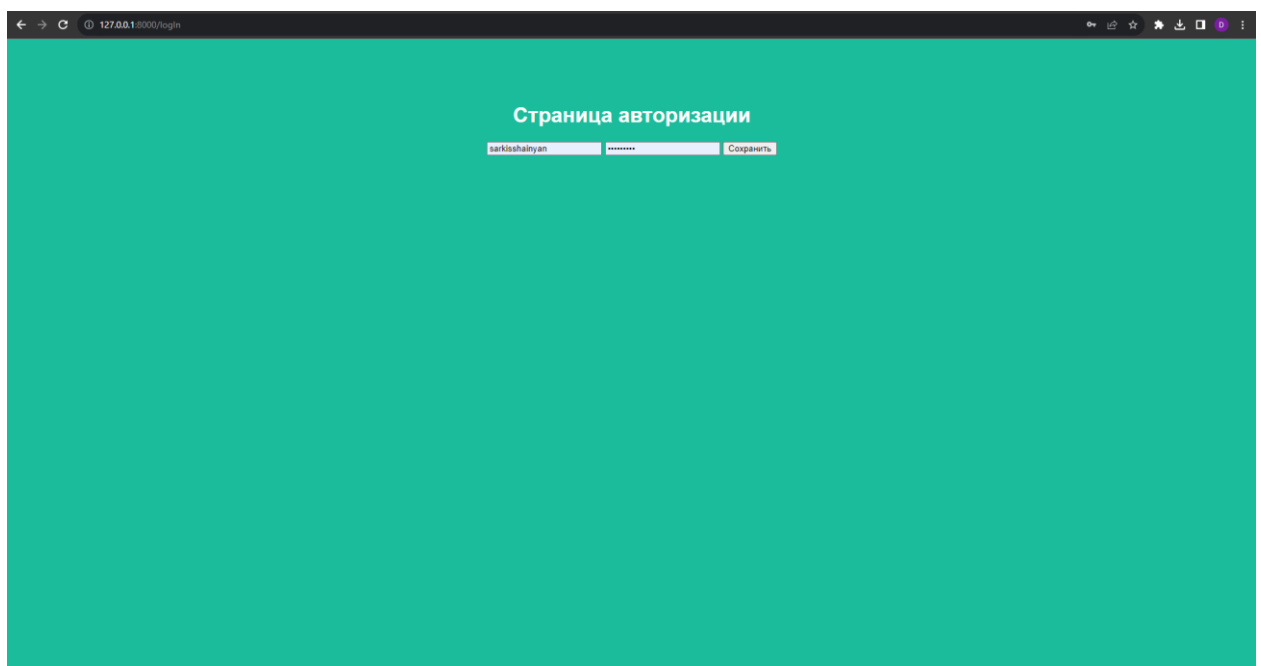
На рисунках 7-8 представлены страницы авторизации и регистрации



127.0.0.1:8000/registration

Страница регистрации

Рисунок 1 – Страница регистрации



127.0.0.1:8000/login

Страница авторизации

Рисунок 2 – Страница авторизации

Вывод: в данной лабораторной работе я научился создавать страницу регистрации и входа.