Лабораторная работа №6

Django предоставляет систему аутентификации и авторизации пользователя, реализованную на основе фреймворка работы с сессиями. Система аутентификации и авторизации позволяет вам проверять полномочия пользователей и определять какие кому соответствуют. Код данного приложения находится в пакете django.contrib.auth.

В соответствии с идеологией Django система аутентификации является очень общей и не предоставляет некоторые возможности, которые присутствуют в других системах веб-аутентификации. Решениями некоторых общих задач занимаются пакеты сторонних разработчиков, например, защита от подбора пароля (через стороннюю библиотеку OAuth).

Создание формы, шаблона и представления для регистрации

Модель пользователя находится в модуле django.contrib.auth.models.User, можете ознакомиться с исходным кодом модели. Для регистрации пользователя достаточно воспользоваться методом create_user (метод можно посмотреть в объявлении модели User в исходном коде фреймворка). Пример использования функции:

```
from django.contrib.auth.models import User
User.objects.create_user("user1", "user@mail.ru", "user_password")
```

В функцию передается 3 параметра: логин, email и пароль. Для того, чтобы пользователь смог отправить эти данные, необходимо создать страницу с формой регистрации и обработать вводимые значения. Не забудьте выполнять проверку на предмет того, имеется ли уже в базе данных пользователь с таким логином. Чтобы это проверить, достаточно попробовать найти его в базе данных с помощью метода get, который вызовет исключение, если объекта не существует:

```
try:
    User.objects.get(username=username)
# если пользователь существует, то ошибки не произойдет и программа # удачно доберется до следующей строчки
    print "Пользователь с таким именем уже есть"
except User.DoesNotExist:
    print "Этот логин свободен"
```

Задание:

- Создайте шаблон и настройте адрес для отображения формы регистрации;

- 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;
- Создайте стили, подключив CSS-файл к шаблону;
- 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).
- Измените отображение ссылок в шапке сайта в соответствие с состоянием пользователя. Например, если пользователь находится под своей учётной записью, то в шапке должны находиться ссылки: “Создать статью”, “Выход из аккаунта” и приветственная надпись с именем пользователя, иначе “Авторизация”, “Регистрация”. Можно использовать следующую конструкцию в шаблоне:

```
{% if request.user.is_authenticated %}
{% else %}
{% endif %}
```

Создание формы, шаблона и представления для авторизации

Для того, чтобы пользователь мог войти в систему, он должен пройти процесс аутентификации (проверка подлинности предъявленного пользователем идентификатора, т.е. имеется ли в базе данных такая пара логин – пароль) и авторизации (предоставление определённому лицу прав на выполнение определённых действий, процесс проверки (подтверждения) данных прав при попытке выполнения этих действий). Для этого можно воспользоваться функцией authenticate из модуля django.contrib.auth. Данная функция принимает два параметра: логин и пароль и сверяет их с базой данных, если такая пара логин – пароль существует, то метод возвращает объект User, иначе возвратит None:

```
from django.contrib.auth import authenticate
user = authenticate(username="user1", password="user_pass")
```

После удачной аутентификации, можно авторизовать пользователя функцией **login** из модуля **django.contrib.auth**. Функция принимает два параметра: текущий объект запроса и объект пользователя):

```
from django.contrib.auth import login
login(request, user)
```

Иначе, если аутентификация оказалось неудачной и вместо объекта User вернулся None, то пользователю нужно снова вернуть форму входа в систему, при этом вывив сообщение о том, что такого аккаунта не существует.

Задание:

- Создайте шаблон и настройте адрес для отображения формы авторизации;
- Сделайте недоступными для авторизованных пользователей страницы с регистрацией и авторизацией.
- Создайте представление, которое будет обрабатывать поступающие запросы и авторизировать 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;
- Создайте стили, подключив CSS-файл к шаблону;
- Загрузите ваш проект на любой гит-репозиторий (GitHub, GitLab, Google Code, Bitbucket и т.п.).