**План тестирования**

Ваша ближайшая задача — протестировать проект, а для этого нужно спланировать, что именно нужно тестировать, а что не стоит.

Будем тестировать:

В файле *test\_routes.py*:

* Главная страница доступна анонимному пользователю.
* Страница отдельной новости доступна анонимному пользователю.
* Страницы удаления и редактирования комментария доступны автору комментария.
* При попытке перейти на страницу редактирования или удаления комментария анонимный пользователь перенаправляется на страницу авторизации.
* Авторизованный пользователь не может зайти на страницы редактирования или удаления чужих комментариев (возвращается ошибка 404).
* Страницы регистрации пользователей, входа в учётную запись и выхода из неё доступны анонимным пользователям.

В файле *test\_content.py*:

* Количество новостей на главной странице — не более 10.
* Новости отсортированы от самой свежей к самой старой. Свежие новости в начале списка.
* Комментарии на странице отдельной новости отсортированы от старых к новым: старые в начале списка, новые — в конце.
* Анонимному пользователю недоступна форма для отправки комментария на странице отдельной новости, а авторизованному доступна.

В файле *test\_logic.py*:

* Анонимный пользователь не может отправить комментарий.
* Авторизованный пользователь может отправить комментарий.
* Если комментарий содержит запрещённые слова, он не будет опубликован, а форма вернёт ошибку.
* Авторизованный пользователь может редактировать или удалять свои комментарии.
* Авторизованный пользователь не может редактировать или удалять чужие комментарии.

Не будем тестировать:

* Непосредственно регистрацию пользователей, процесс входа в учетную запись и выхода из неё.
* Всё, что связано с админ-зоной проекта.
* Абсолютные url-адреса; обращаться к адресам будем при помощи функции reverse('name').
* Какие шаблоны используются, и насколько корректно шаблон обрабатывает полученную от Django информацию. Например, если в шаблон передается список объектов, мы будем считать, что они выводятся все, а их сортировка не меняется.
* Всё то, что не попало в список «будем тестировать».

Отдельно стоит сказать про шаблоны. Чтобы проверить, как именно шаблон выводит информацию, зачастую в тестах надо исследовать HTML; не будем этим заниматься. Будем считать, что шаблон написан корректно и выводит то, что нужно, а разработчик проверил это на этапе работы над приложением.

Дополнительно в тестах мы исходим из того, что атрибуты объектов моделей передаются в контекст шаблона в неизменном виде. То есть в проекте нет кода, который «по пути» из моделей в шаблоны изменяет атрибуты объекта. Да, это можно было бы проверить, но это представляется излишним.

В тестах можно проверять множество мелочей, но надо соблюдать баланс и уметь вовремя остановиться. Важно проверить критичный для приложения функционал, логику, написанную самим разработчиком, и тот функционал, который может быть изменён.

Например, регистрация пользователей, авторизация и выход из аккаунта — это, по большей части, работа самого Django-фреймворка, и для этих действий тесты можно не писать: разработчики Django много раз всё это протестировали.

При работе над приложением можно вручную проверить, что это работает; если в дальнейшем этот функционал не будет меняться — то и проверять его автоматизированными тестами нет особой необходимости.