**10.6 Практическая работа**

Цели практической работы

* Написать автотесты для view-функций Django.
* Осуществить проверку view, статуса кода, контекста, шаблона.
* Применить фикстуры.

Что нужно сделать

Воспользуйтесь кодовой базой из пройденных модулей или файлами из репозитория с практической работой.

Напишите тест для OrderDetailView:

1. Объявите класс тестов OrderDetailViewTestCase.
2. В setUpClass создайте пользователя и добавьте пользователю необходимые разрешения для просмотра заказа (“shopapp.view\_order”).
3. Добавьте удаление пользователя в tearDownClass.
4. При помощи метода setUp сделайте вход пользователя и создание заказа для дальнейшего теста.
5. В методе tearDown сделайте удаление заказа.
6. Объявите метод test\_order\_details для проверки получения заказа:
   * убедитесь, что в теле ответа есть адрес заказа;
   * убедитесь, что в теле ответа есть промокод;
   * убедитесь, что в контексте ответа тот же заказ, который был создан перед тестом (сравните по первичному ключу).

Создайте новый view и тест для этого view. Если хотите выполнить задание со звёздочкой, сделайте тест по TDD. Реализуйте view для выгрузки заказов в JSON-виде и напишите для этого тест:

1. Реализуйте view-функцию/класс для выгрузки информации по заказам:
   1. В теле JSON по ключу orders должен возвращаться список с данными по всем заказам.
   2. В информации по каждому заказу должны быть ID заказа, адрес, промокод, ID пользователя, список ID продуктов.
   3. Доступ к этому view должен быть только у пользователей с уровнем доступа is\_staff (для этого нужно использовать проверку через user passes test).
2. Объявите класс тестов OrdersExportTestCase.
3. В setUpClass задайте создание пользователя с необходимым уровнем доступа.
4. В tearDownClass сделайте удаление пользователя.
5. В setUp выполните вход пользователя.
6. Напишите тест, который проверяет получение списка заказов:
   1. Статус кода-ответа — 200.
   2. В JSON-теле ответа должны быть ожидаемые значения.
7. Создайте фикстуры при помощи команды dumpdata. Выполните создание фикстур для следующих сущностей:
   1. Пользователи.
   2. Продукты.
   3. Заказы.
8. Используйте эти фикстуры, чтобы проверять экспорт заказов (заказы должны быть связаны с пользователями и продуктами).

Что оценивается

* Сделан тест для OrderDetailView. В классе теста:
  + перед тестами происходит создание пользователя;
  + пользователю выдаются необходимые права для просмотра заказа;
  + перед тестом выполняется вход пользователя;
  + перед тестом выполняется создание заказа, который будет проверен далее в тесте;
  + после теста заказ удаляется;
  + после тестов выполняется удаление пользователя.
* Выполнена проверка, чтобы убедиться, что:
  + в теле ответа есть адрес заказа;
  + в теле ответа есть промокод;
  + в контексте ответа тот же заказ, который был создан перед тестом (проверка по первичному ключу).
* Создан OrdersExportView для экспорта заказов:
  + доступ только для is\_staff (проверка через user passes test);
  + все заказы запрашиваются с подгрузкой пользователей и продуктов;
  + в теле по каждому заказу должны быть поля с ID заказа, адрес, промокод, ID пользователя, список ID продуктов.
* Тест для OrdersExportView сделан, в классе теста:
  + использованы фикстуры, чтобы наполнять базу данных информацией по заказам;
  + создан пользователь с необходимым уровнем доступа;
  + после прохождения тестов пользователь удалён;
  + перед тестами выполняется вход пользователя.
* Выполнена проверка:
  + статус кода-ответа — 200;
  + тело совпадает с ожидаемым.

Как отправить работу на проверку

Сдайте практическую работу через Skillbox GitLab. В поле сдачи практической работы напишите «Сделано» и пришлите ссылку на репозиторий через форму ниже.