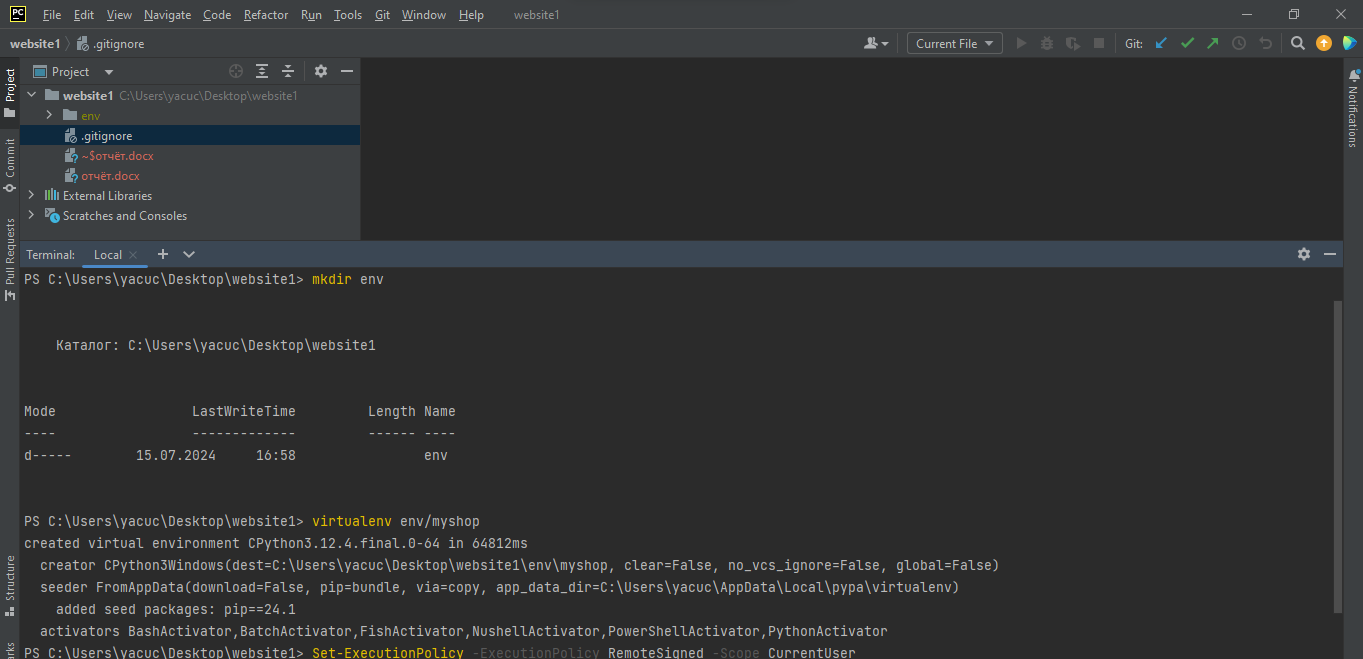
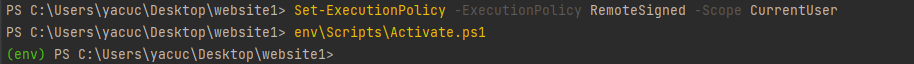
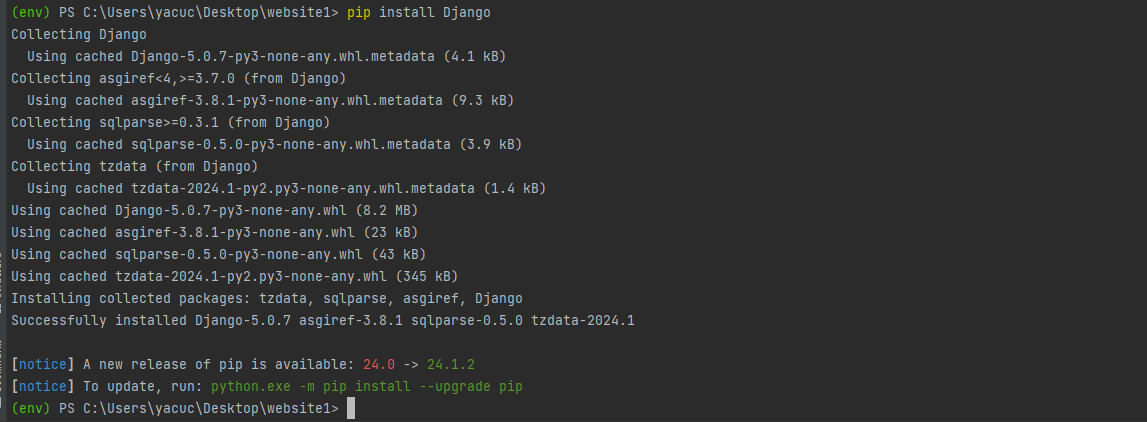
Создаём виртуальную среду для нового проекта и активируем её следующим образом:

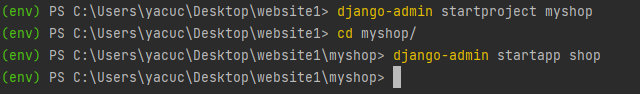




Установим Django в виртуальной среде:



Начнём новый проект под названием myshop с приложением, называемым shop:

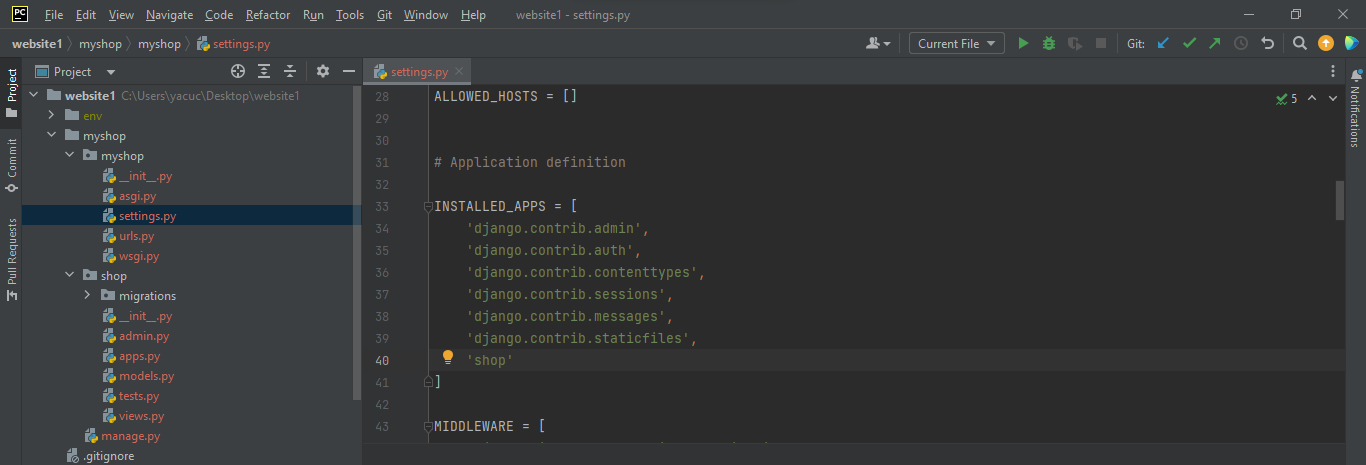


Изменим файл settings.py проекта и добавим приложение к настройкам INSTALLED\_APPS: INSTALLED\_APPS =(

# ...

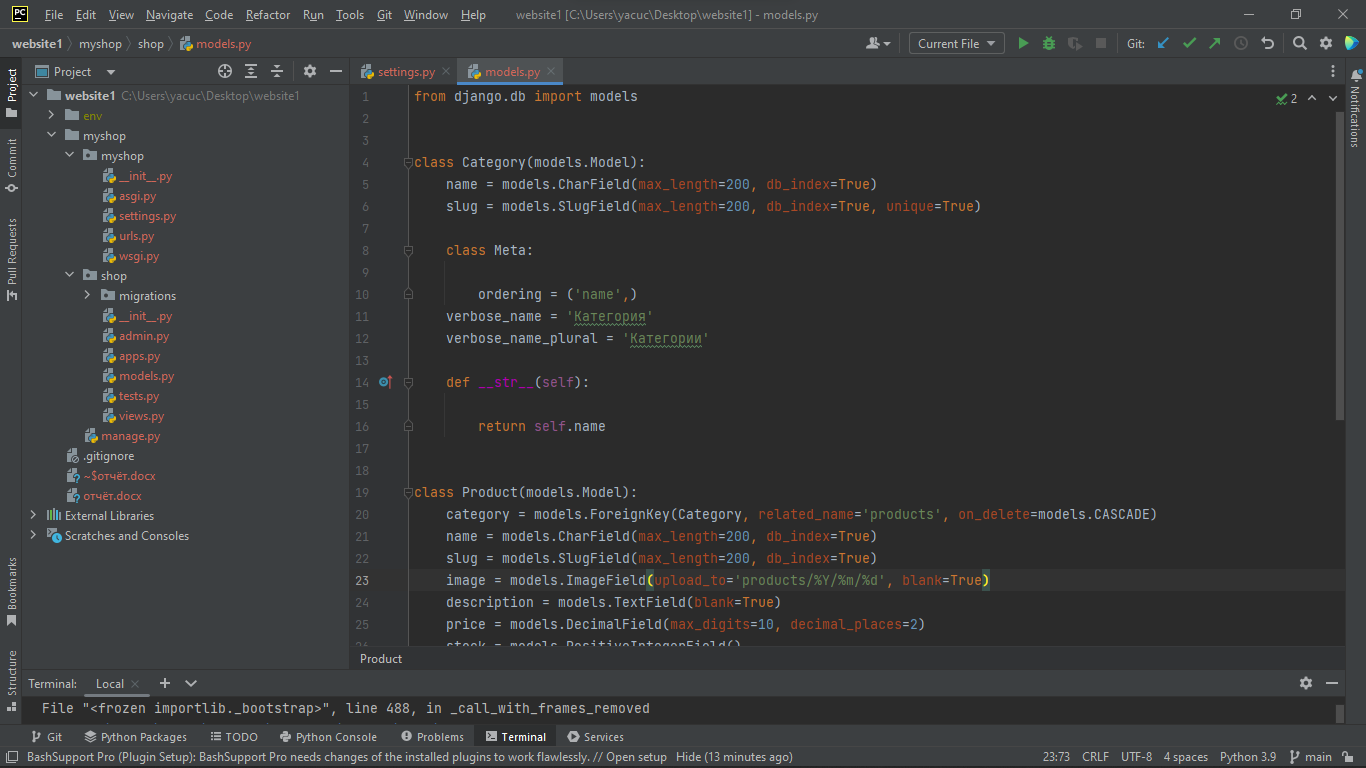
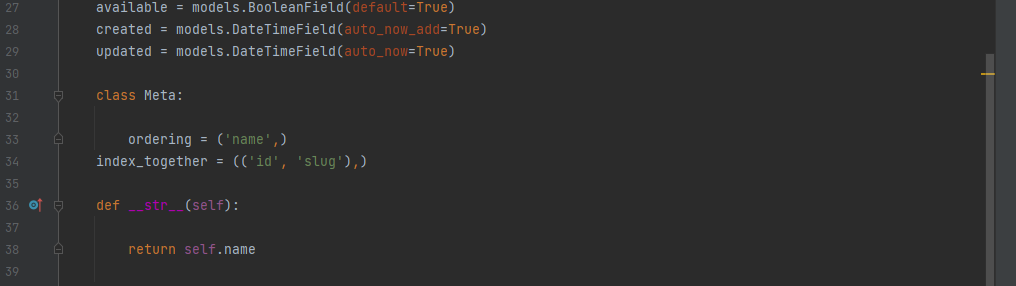
'shop',

)

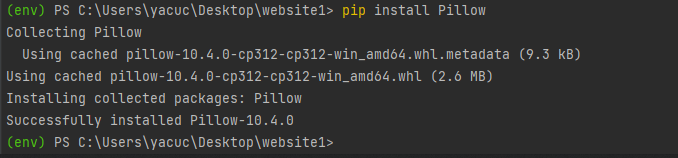


Теперь приложение активно для данного проекта. Определим модели для каталога продуктов.

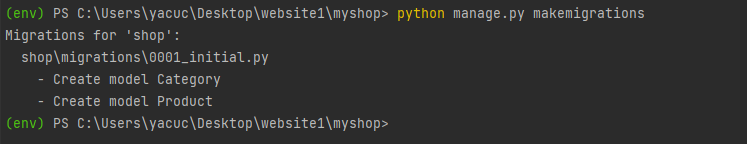
Каталог нашего магазина будет состоять из продуктов, сгруппированных по разным категориям. Каждый продукт будет иметь имя, необязательное описание, необязательное изображение, цену и доступный запас. Отредактируем файл models.py только что созданного приложения shop:

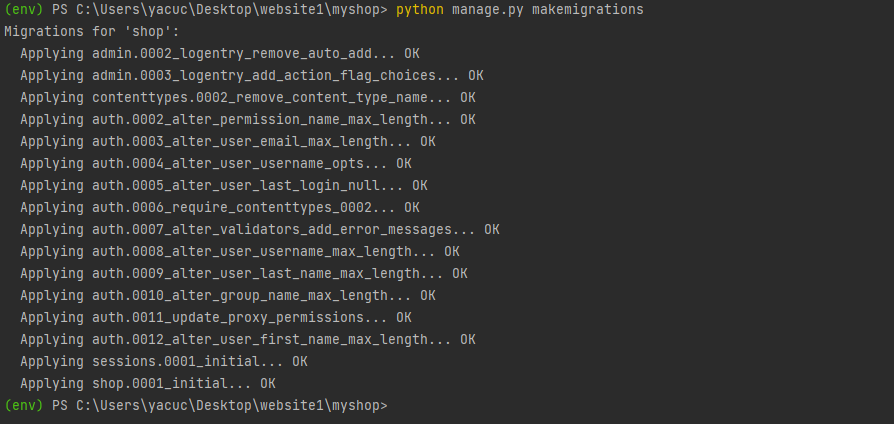
Поскольку мы собираемся использовать изображения в наших моделях, установим Pillow:



Теперь выполним следующую команду, чтобы создать начальные миграции для проекта:

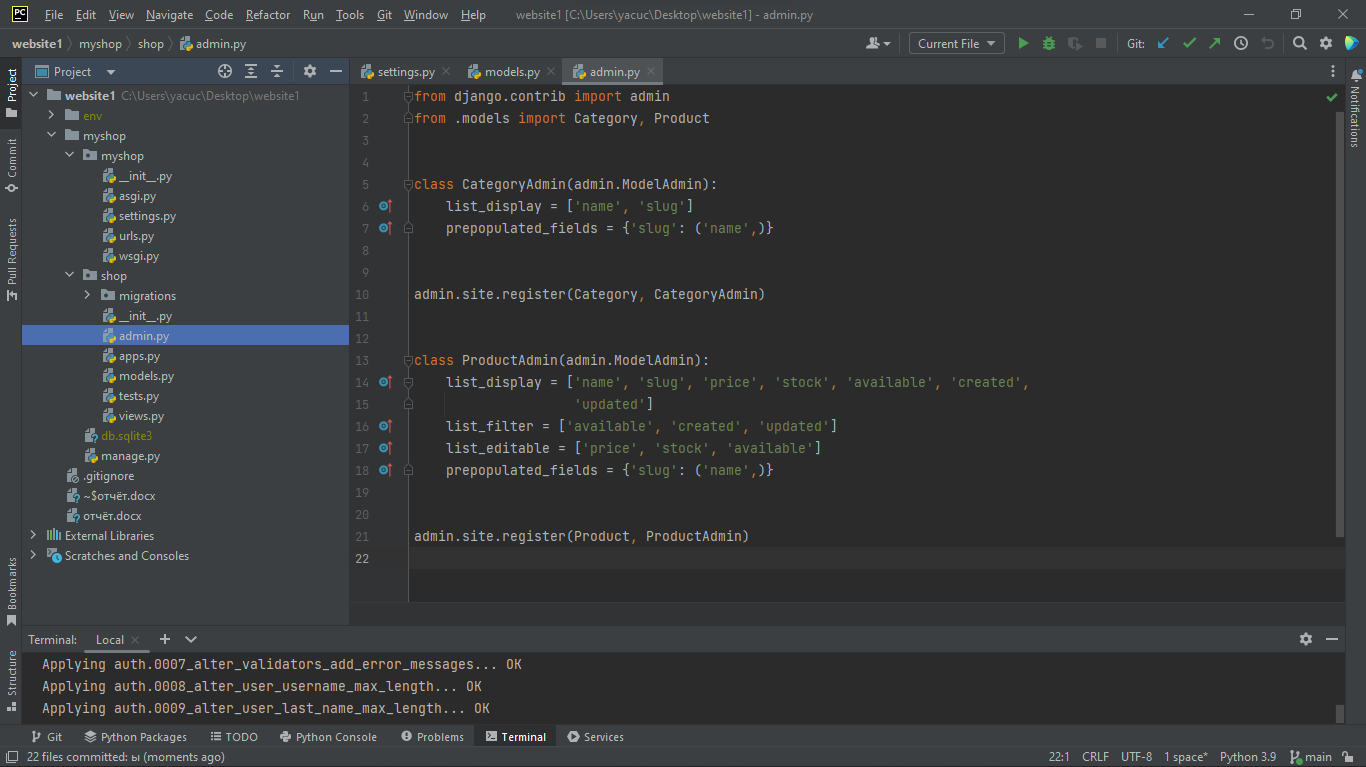


Выполним следующую команду для синхронизации базы данных:

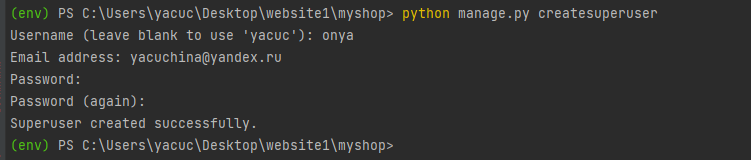


Теперь база данных синхронизирована с моделями.

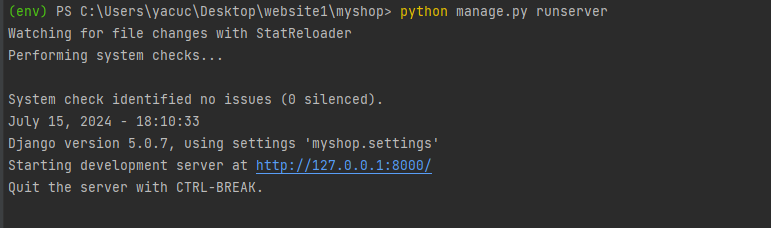
Добавим наши модели на сайт администрирования, чтобы мы могли легко управлять категориями и продуктами. Изменим файл admin.py приложения shop:

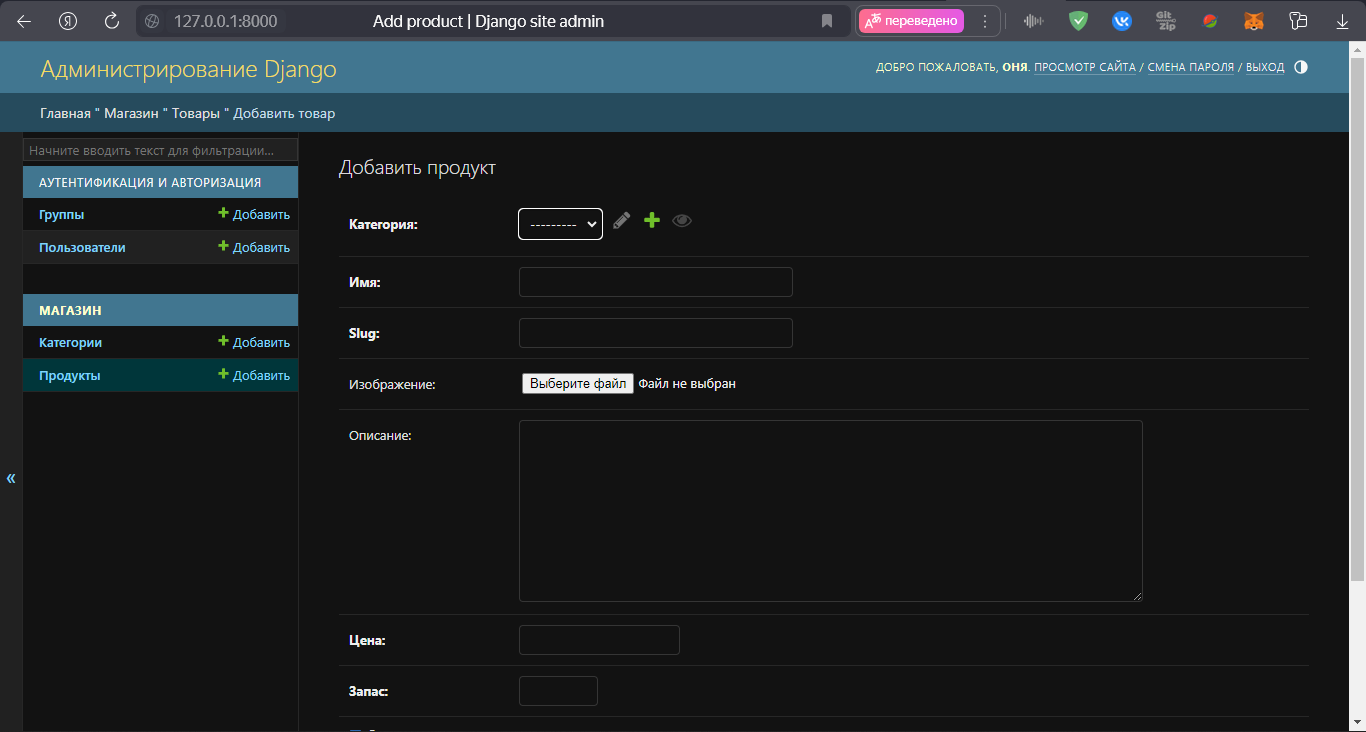


Теперь создадим суперпользователя:

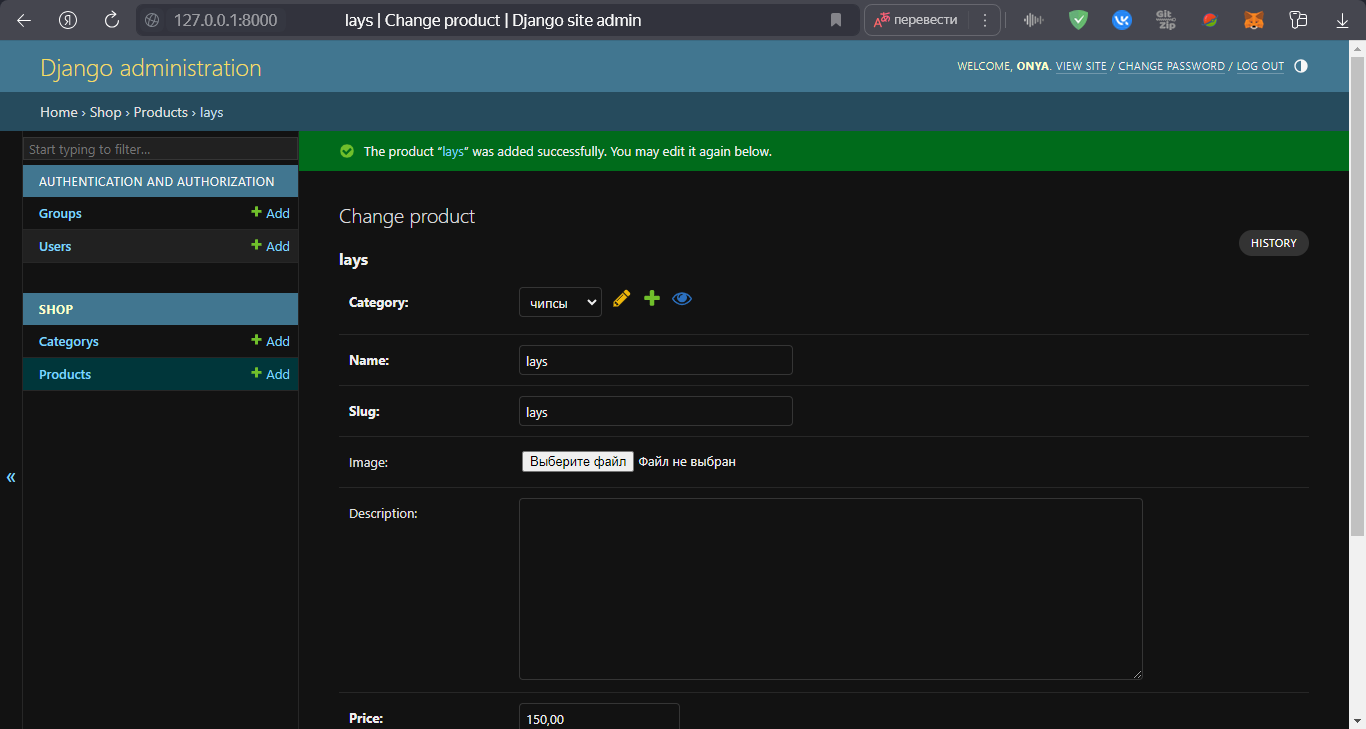
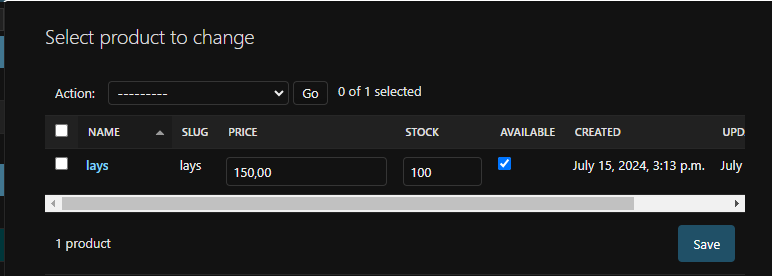


Запустим сервер разработки:





Добавим новую категорию и продукт с помощью интерфейса администрирования:

Теперь необходимо создать шаблоны для списка товаров и одного товара. Создадим следующую структуру файлов в каталоге приложения shop:

templates/

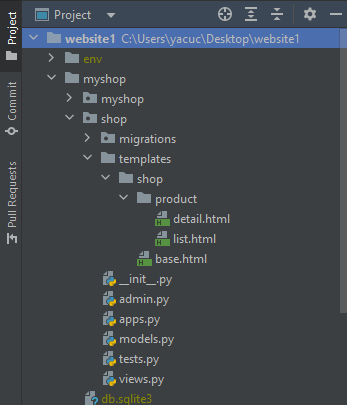
shop/

base.html

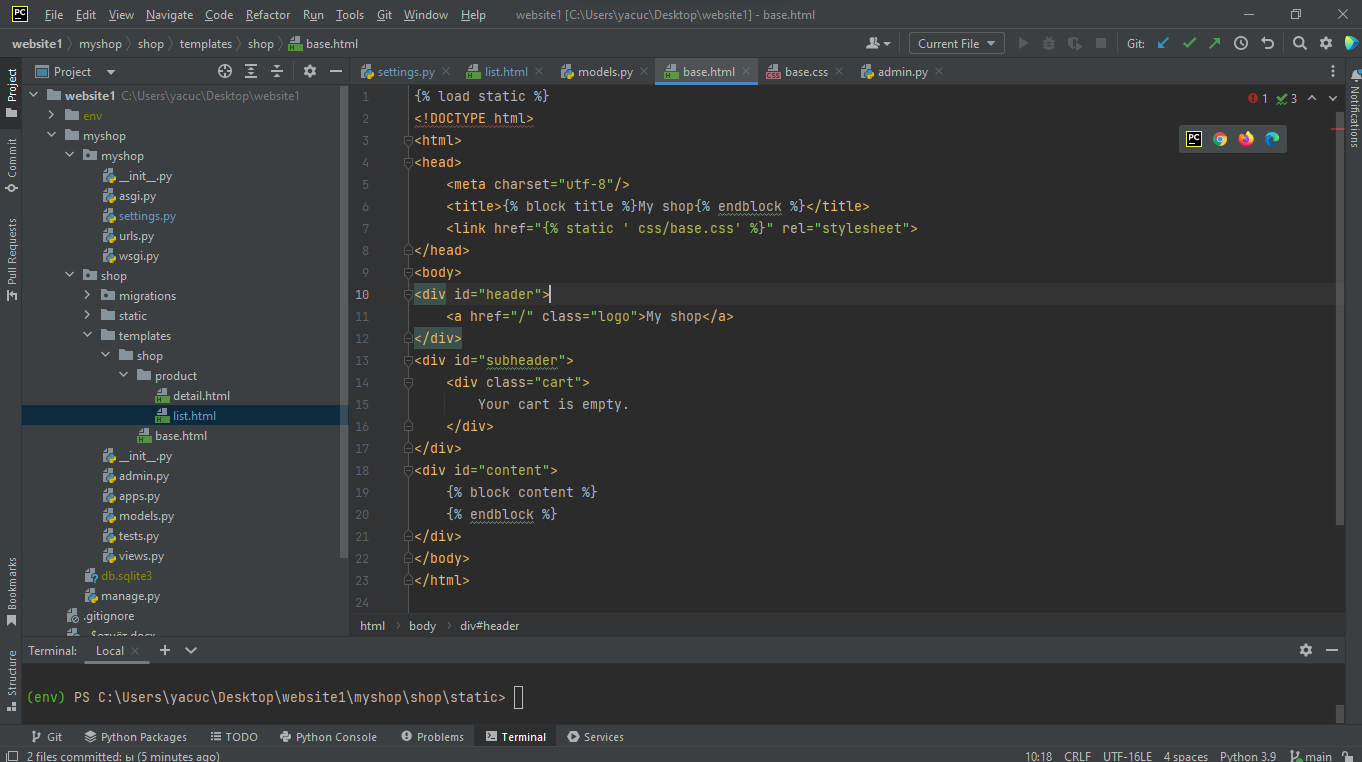
product/

list.html

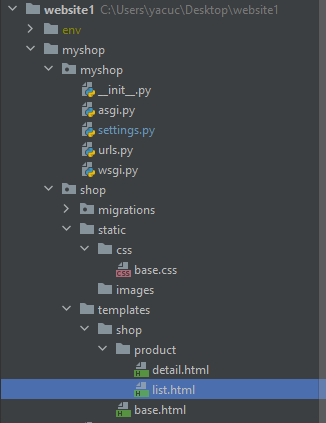
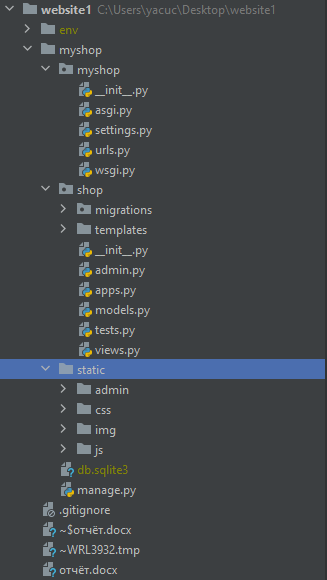
detail.html



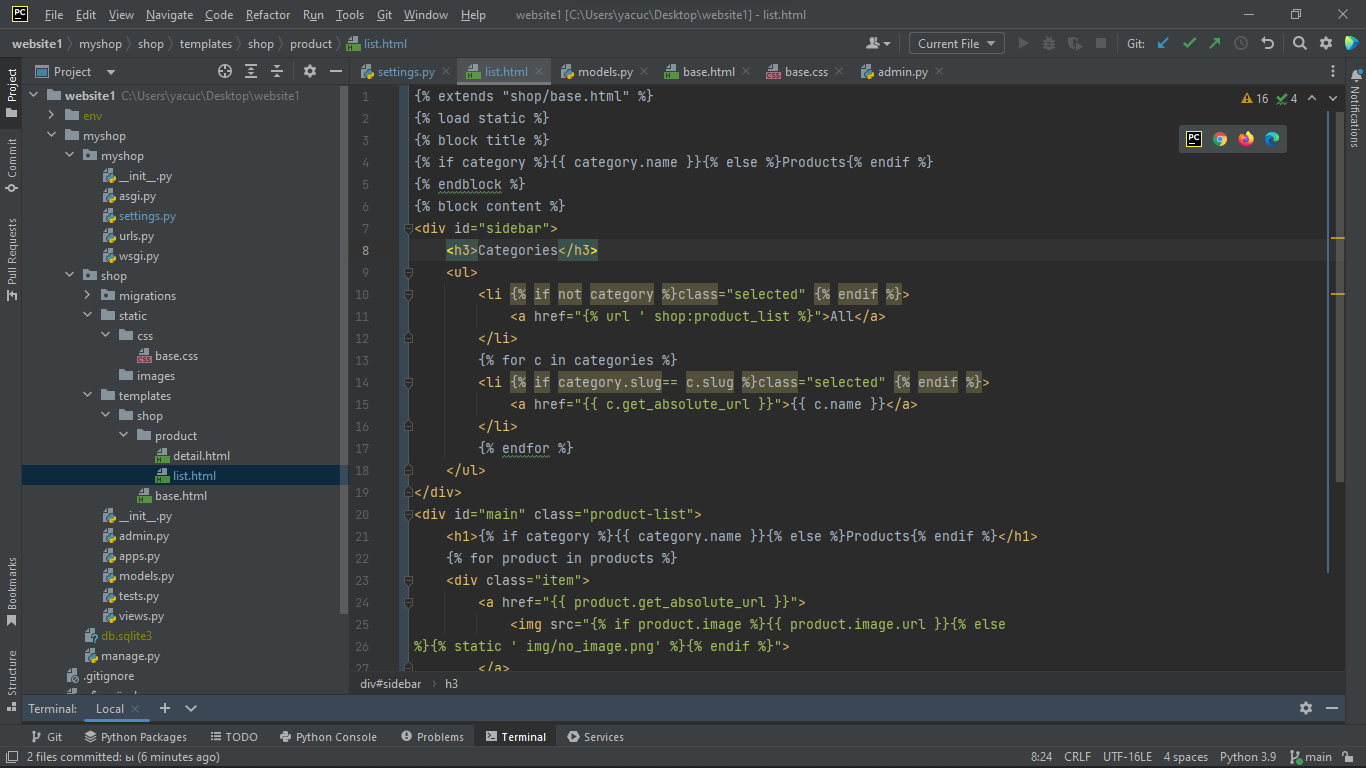
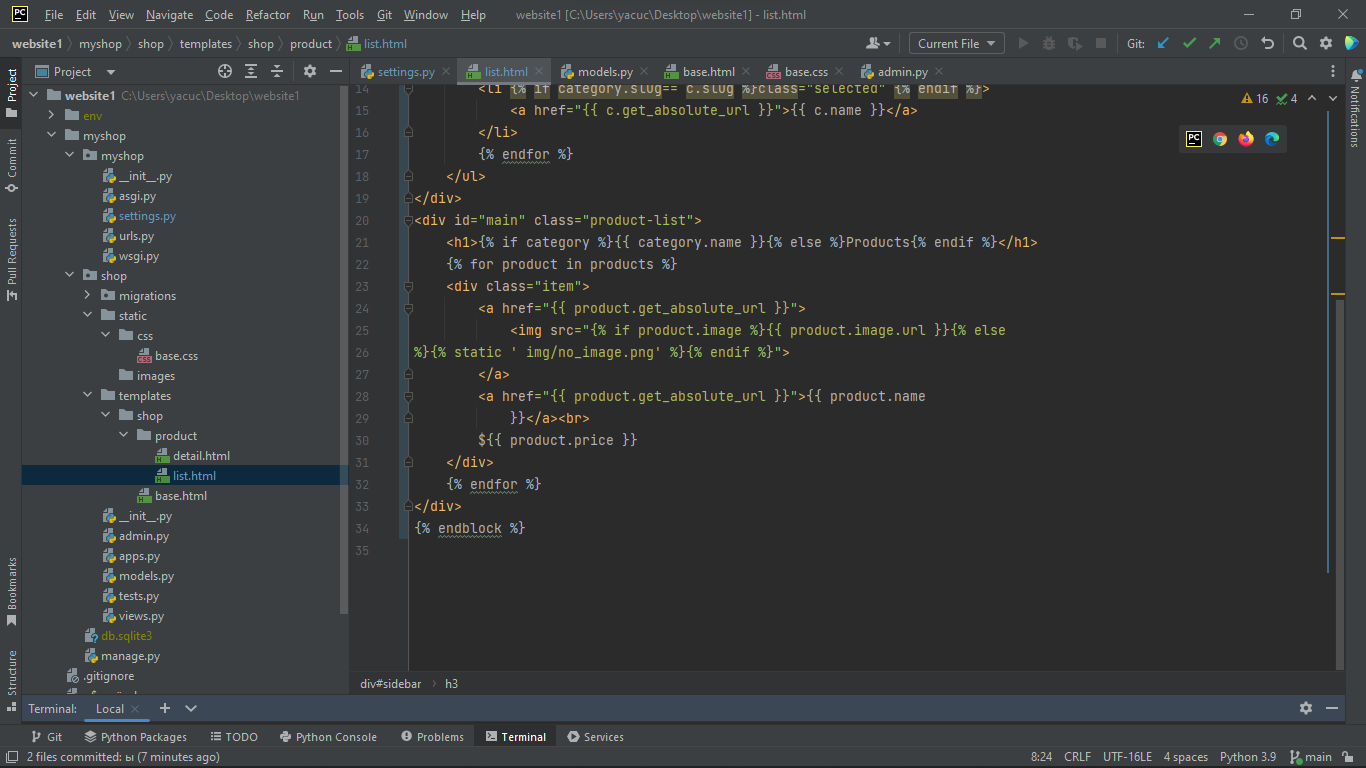
Необходимо определить базовый шаблон, а затем расширить его в list и в шаблоне detail. Отредактируем шаблон shop/base.html:



Чтобы включить требуемые стили CSS и изображения, используемые шаблонами, необходимо скопировать статические файлы, которые входят в эту главу, расположенную в каталоге static/ приложения shop. Скопируем их в ту же папку проекта.

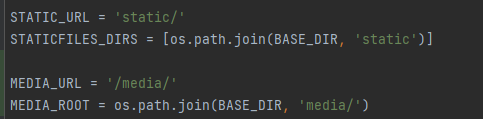
 

Изменим шаблон shop/product/list.html:

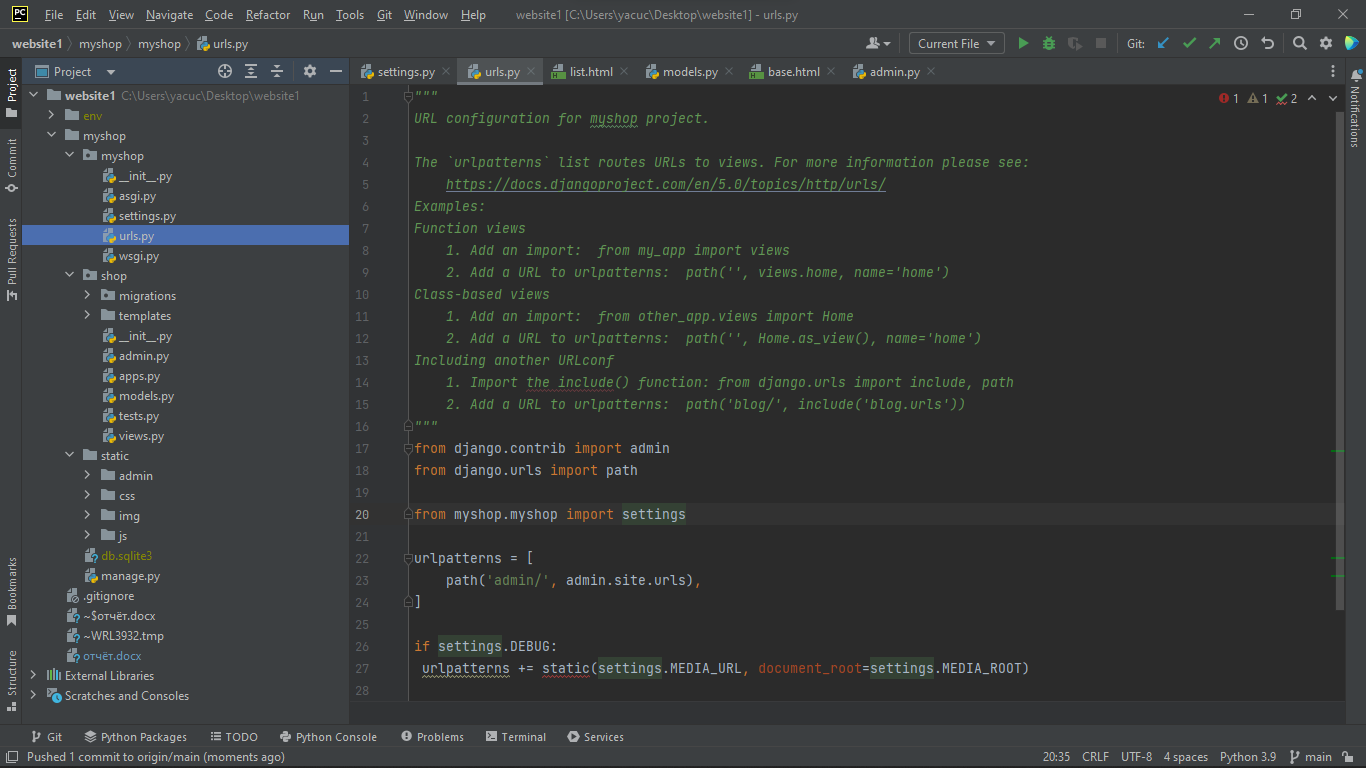
  


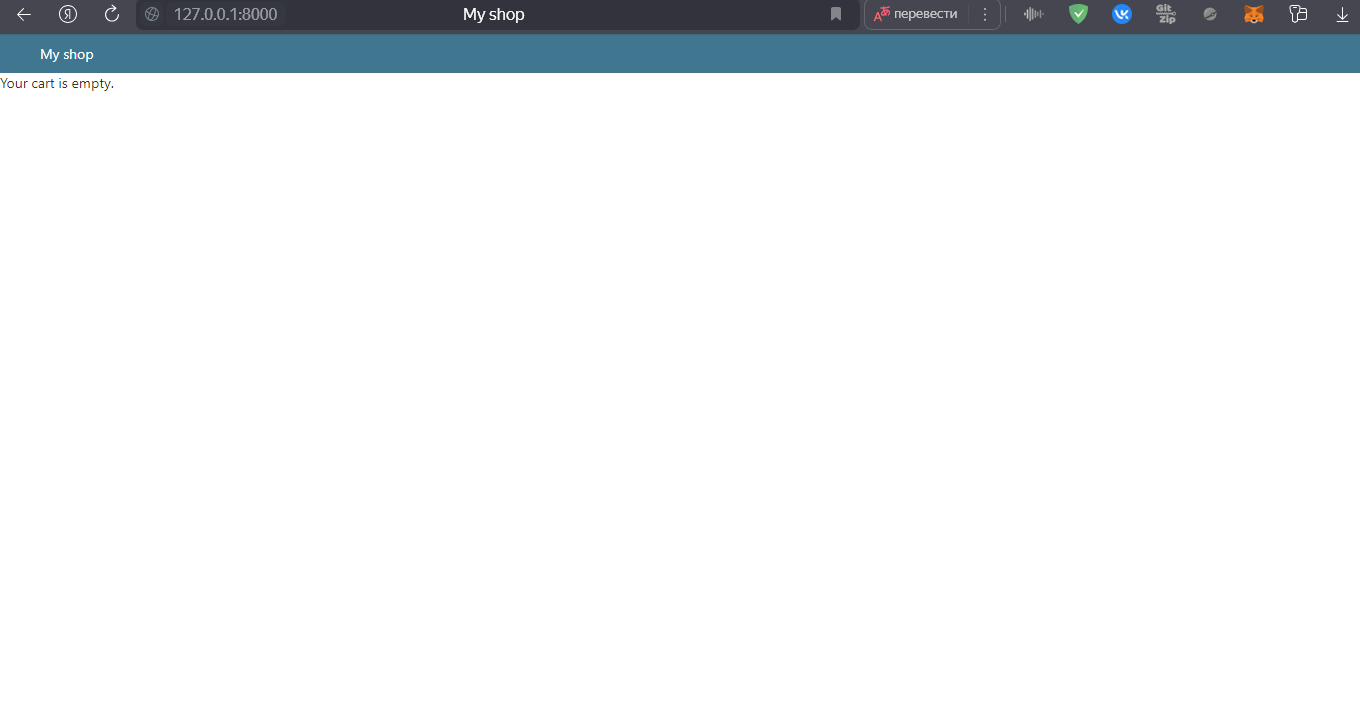
Это шаблон списка продуктов. Он расширяет шаблон shop/base.html и использует переменную контекста категорий для отображения всех категорий на боковой панели и продуктов для отображения продуктов текущей страницы. Один и тот же шаблон используется для обоих типов: список всех доступных продуктов и список продуктов, отфильтрованных по категориям. Поскольку поле изображения модели продукта может быть пустым, мы должны предоставить изображение по умолчанию для продуктов, у которых нет изображения. Изображение находится в каталоге статических файлов с относительным путем img/no\_image.png.

Отредактируем settings.py:



Чтобы Джанго обрабатывал загруженные мультимедийные файлы с помощью сервера разработки, отредактируем файл urls.py:



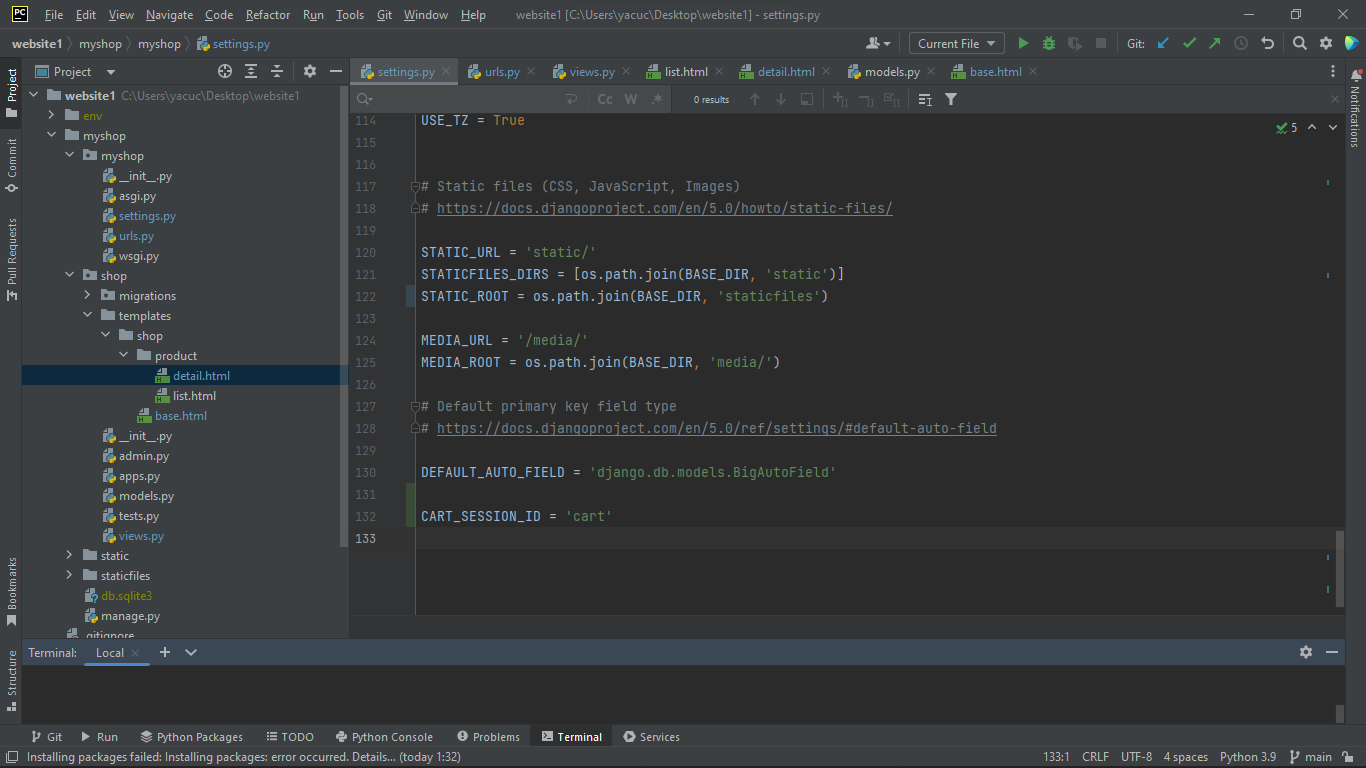
base.html будет выводить следующую страницу в соответствии с конструктором:  


Добавим в магазин несколько продуктов с помощью сайта администрирования:

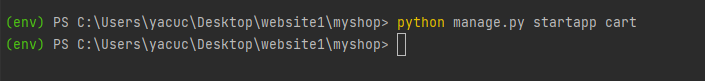
&&&&

Создаём корзину

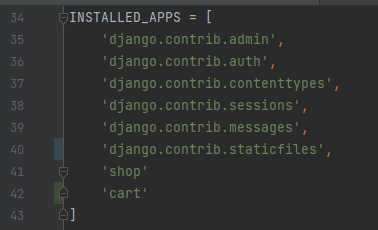
Изменим файл settings.py проекта:



Создадим приложение для управления корзинами покупок:



Добавим "cart" к параметру INSTALLED\_APPS:



Создадим новый файл в каталоге приложения cart: