



## Rest Api for shop page

Steps:

1. Open postman tool
2. Choose get method and paste the provided url
3. for authentication choose Header tab and type in parameters

Key -> Authorization

value-> **Token f716397ecea3f7405ef35af149bedf8d4a0c99b8**

4. Click send
- 

### Task Completed:

- Product Image
- Name
- Price
- Discount Price
- Quantity
- Stock

### Created 2 category

- Whole Grain flour (with 3 product)
- Whole wheat flour ( with 3 product)

### Provide PUT, GET, POST, PATCH method

To retrieve list of products

- <url>/products

To retrieve specific products from id

- <url>/products/1

Same as followed with category

----- **Code setting.py** -----

```
INSTALLED_APPS = [  
    'shopwebapp',  
    'rest_framework',  
    'rest_framework.authtoken',  
]
```

> Add app name in settings.py as shopwebapp is my app name

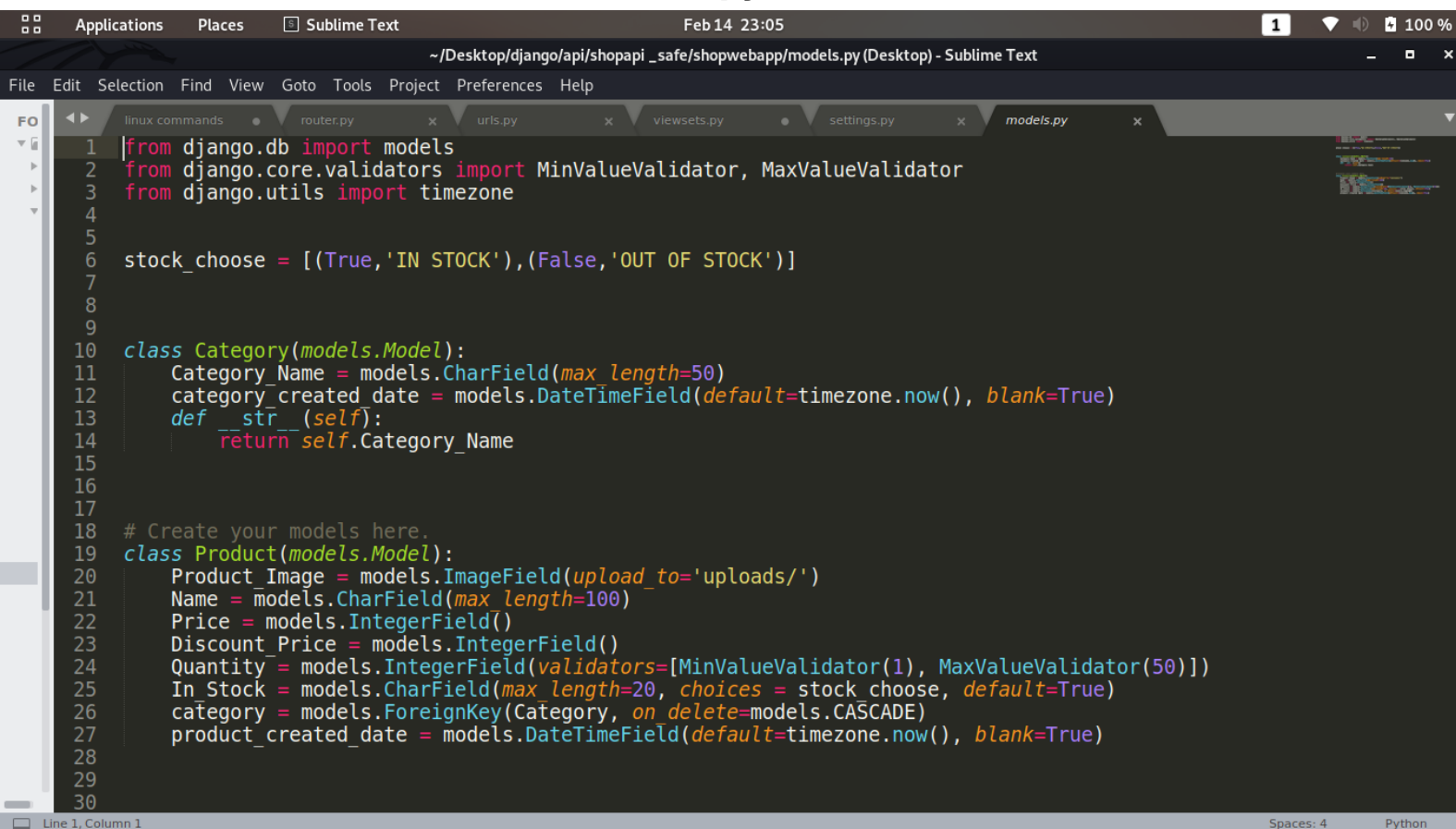
> at the end add below code for authentication

```
REST_FRAMEWORK = {
    'DEFAULT_AUTHENTICATION_CLASSES':[
        'rest_framework.authentication.TokenAuthentication',
    ],
    'DEFAULT_PERMISSION_CLASSES':[
        'rest_framework.permissions.IsAuthenticated',
    ]
}
```

---

> migrate settings

----- models.py -----

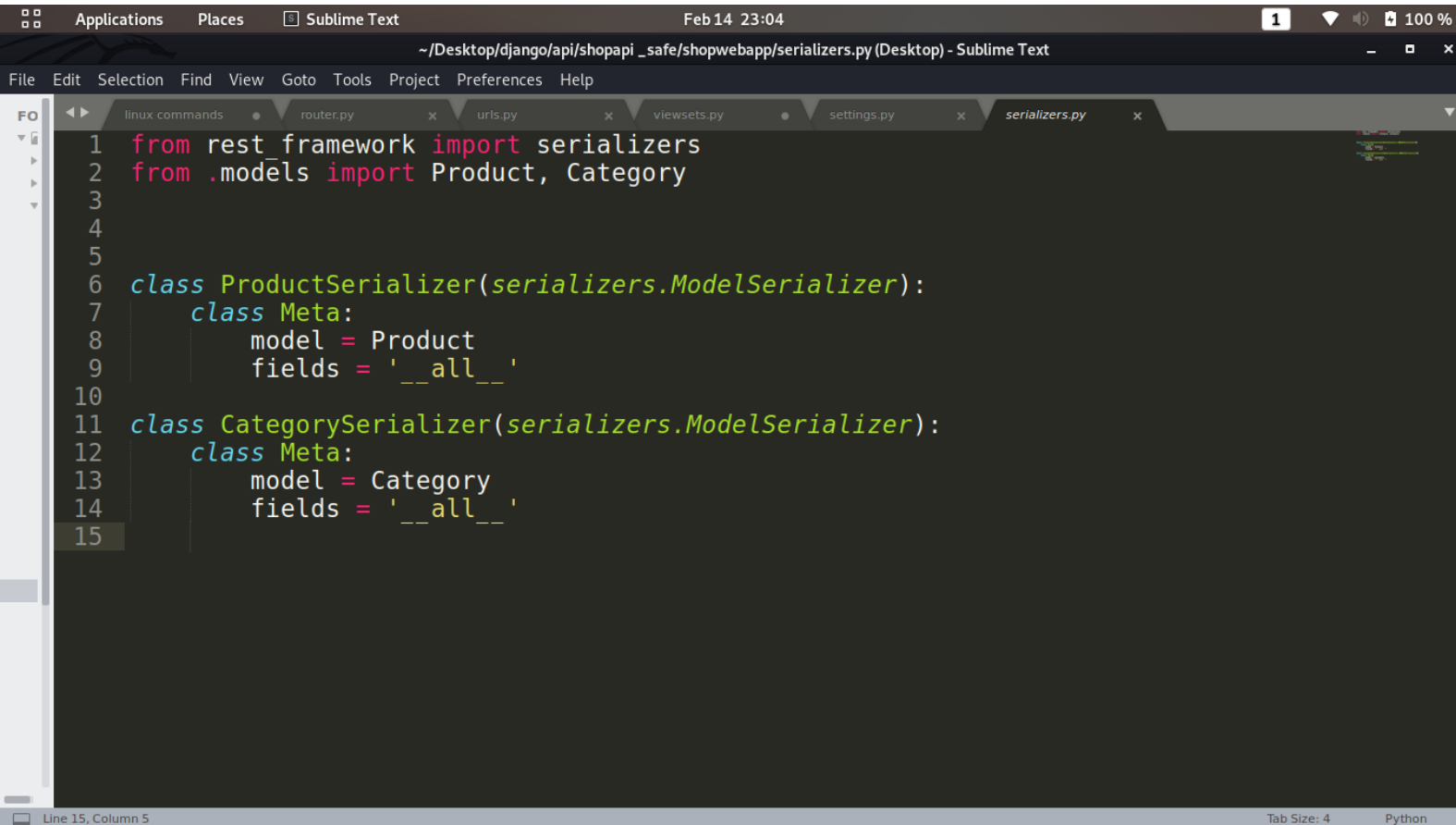


```
1 from django.db import models
2 from django.core.validators import MinValueValidator, MaxValueValidator
3 from django.utils import timezone
4
5
6 stock_choose = [(True, 'IN STOCK'), (False, 'OUT OF STOCK')]
7
8
9
10 class Category(models.Model):
11     Category_Name = models.CharField(max_length=50)
12     category_created_date = models.DateTimeField(default=timezone.now(), blank=True)
13     def __str__(self):
14         return self.Category_Name
15
16
17
18 # Create your models here.
19 class Product(models.Model):
20     Product Image = models.ImageField(upload_to='uploads/')
21     Name = models.CharField(max_length=100)
22     Price = models.IntegerField()
23     Discount_Price = models.IntegerField()
24     Quantity = models.IntegerField(validators=[MinValueValidator(1), MaxValueValidator(50)])
25     In_Stock = models.CharField(max_length=20, choices = stock_choose, default=True)
26     category = models.ForeignKey(Category, on_delete=models.CASCADE)
27     product_created_date = models.DateTimeField(default=timezone.now(), blank=True)
28
29
30
```

> Serializers used in Django rest framework to convert django model objects into json format

> router from rest framework supports to route the url automatically and quickly

## ----- Create serializers.py file -----

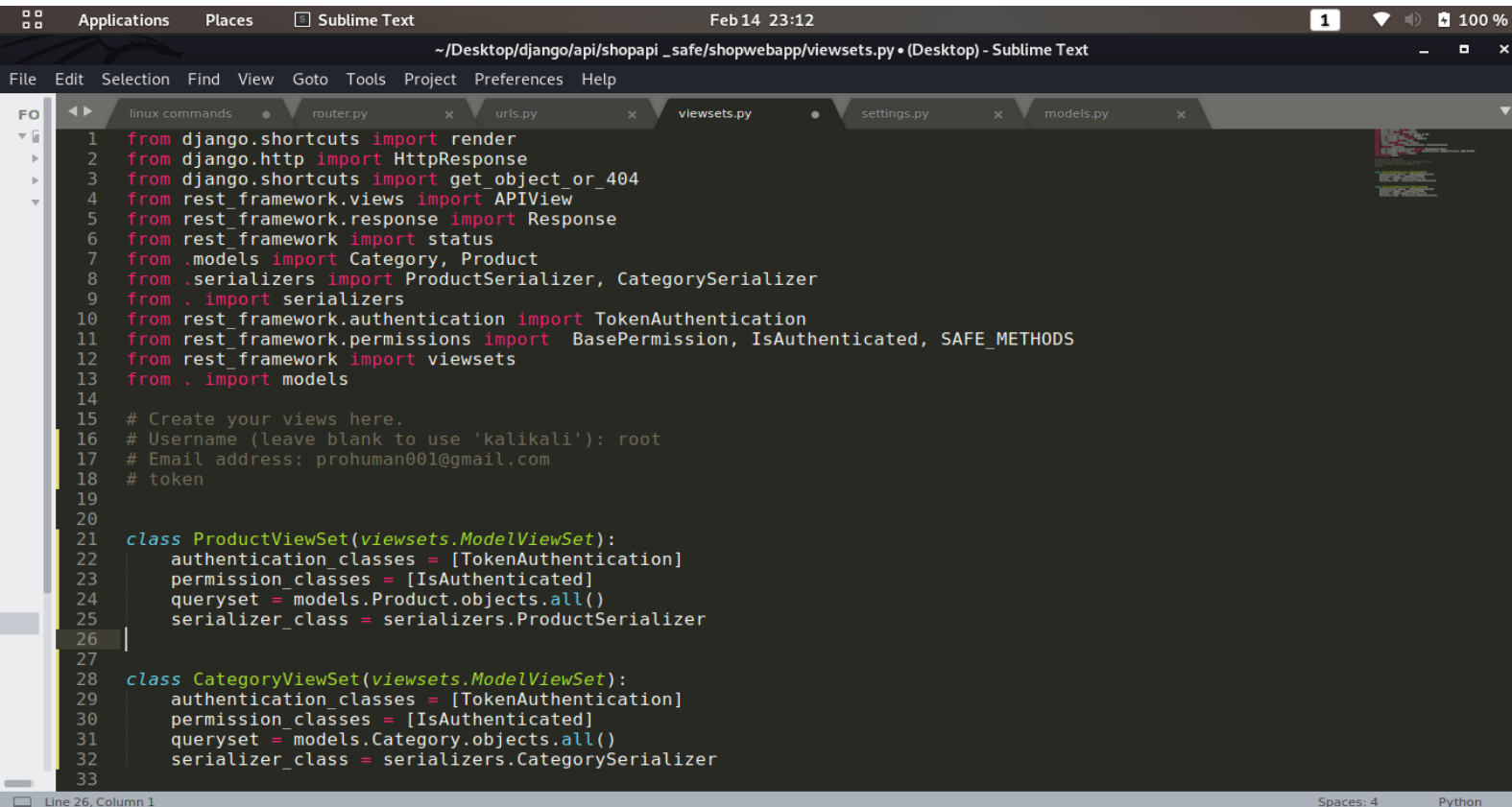


The screenshot shows the Sublime Text editor with the file `serializers.py` open. The code defines two serializers: `ProductSerializer` and `CategorySerializer`, both inheriting from `serializers.ModelSerializer`. The `ProductSerializer` is associated with the `Product` model, and the `CategorySerializer` is associated with the `Category` model. Both serializers use the `'__all__'` field list.

```
1 from rest_framework import serializers
2 from .models import Product, Category
3
4
5
6 class ProductSerializer(serializers.ModelSerializer):
7     class Meta:
8         model = Product
9         fields = '__all__'
10
11 class CategorySerializer(serializers.ModelSerializer):
12     class Meta:
13         model = Category
14         fields = '__all__'
15
```

Line 15, Column 5

## -----views.py or viewsets.py -----

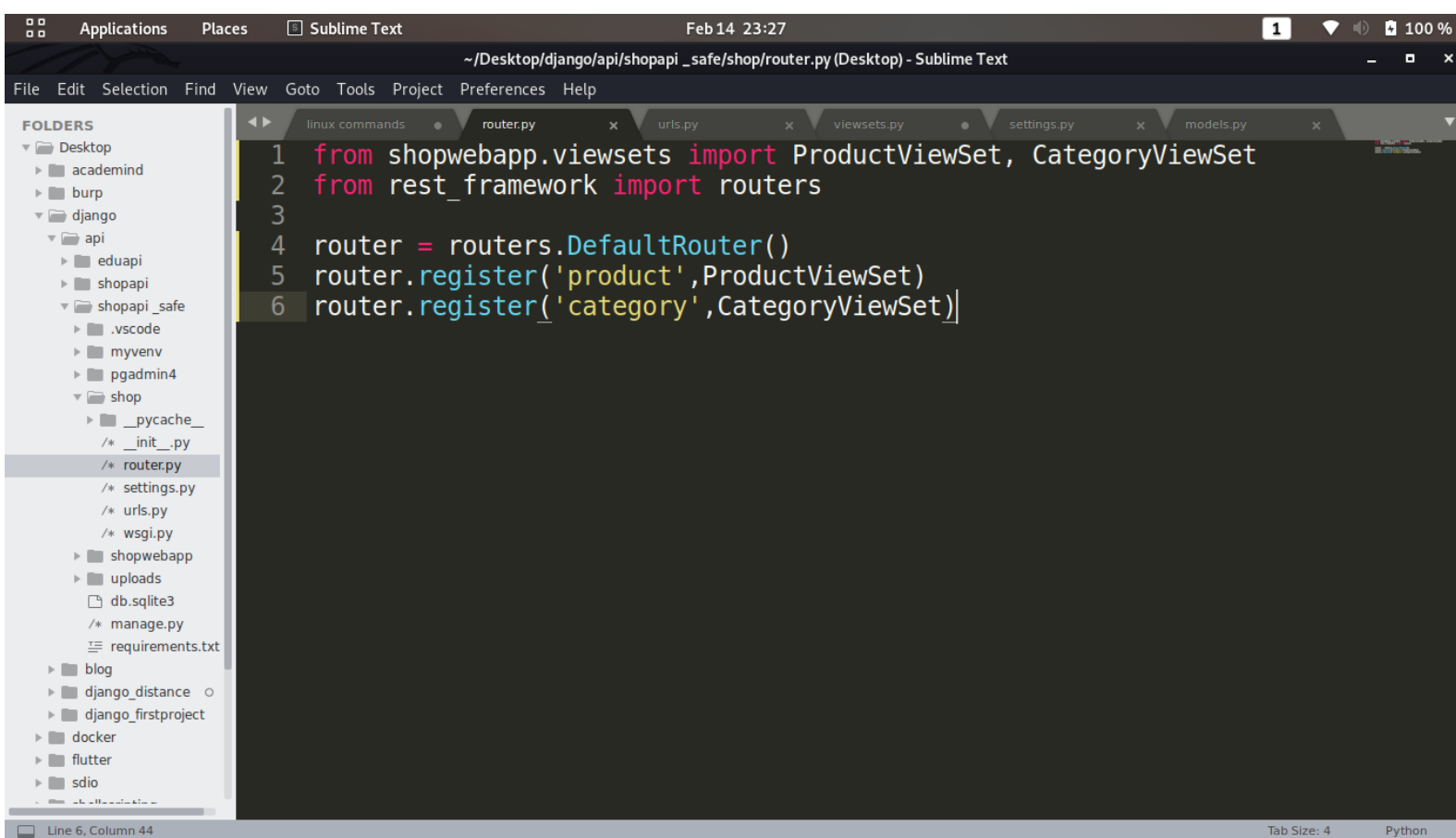


The screenshot shows the Sublime Text editor with the file `viewsets.py` open. The code imports various Django and REST framework modules, including `render`, `HttpResponse`, `APIView`, `Response`, `status`, `TokenAuthentication`, `BasePermission`, `IsAuthenticated`, `SAFE_METHODS`, and `viewsets`. It then defines two viewsets: `ProductViewSet` and `CategoryViewSet`, both inheriting from `viewsets.ModelViewSet`. The `ProductViewSet` is associated with the `Product` model and uses `ProductSerializer`. The `CategoryViewSet` is associated with the `Category` model and uses `CategorySerializer`. Both viewsets use `TokenAuthentication` and `IsAuthenticated` for authentication and permissions.

```
1 from django.shortcuts import render
2 from django.http import HttpResponse
3 from django.shortcuts import get_object_or_404
4 from rest_framework.views import APIView
5 from rest_framework.response import Response
6 from rest_framework import status
7 from .models import Category, Product
8 from .serializers import ProductSerializer, CategorySerializer
9 from . import serializers
10 from rest_framework.authentication import TokenAuthentication
11 from rest_framework.permissions import BasePermission, IsAuthenticated, SAFE_METHODS
12 from rest_framework import viewsets
13 from . import models
14
15 # Create your views here.
16 # Username (leave blank to use 'kalikali'): root
17 # Email address: prohuman001@gmail.com
18 # token
19
20
21 class ProductViewSet(viewsets.ModelViewSet):
22     authentication_classes = [TokenAuthentication]
23     permission_classes = [IsAuthenticated]
24     queryset = models.Product.objects.all()
25     serializer_class = serializers.ProductSerializer
26
27
28 class CategoryViewSet(viewsets.ModelViewSet):
29     authentication_classes = [TokenAuthentication]
30     permission_classes = [IsAuthenticated]
31     queryset = models.Category.objects.all()
32     serializer_class = serializers.CategorySerializer
33
```

Line 26, Column 1

## ----- create router.py file -----



The screenshot shows the Sublime Text editor with the file `~/Desktop/django/api/shopapi_safe/shop/router.py` open. The left sidebar displays a folder tree with the following structure:

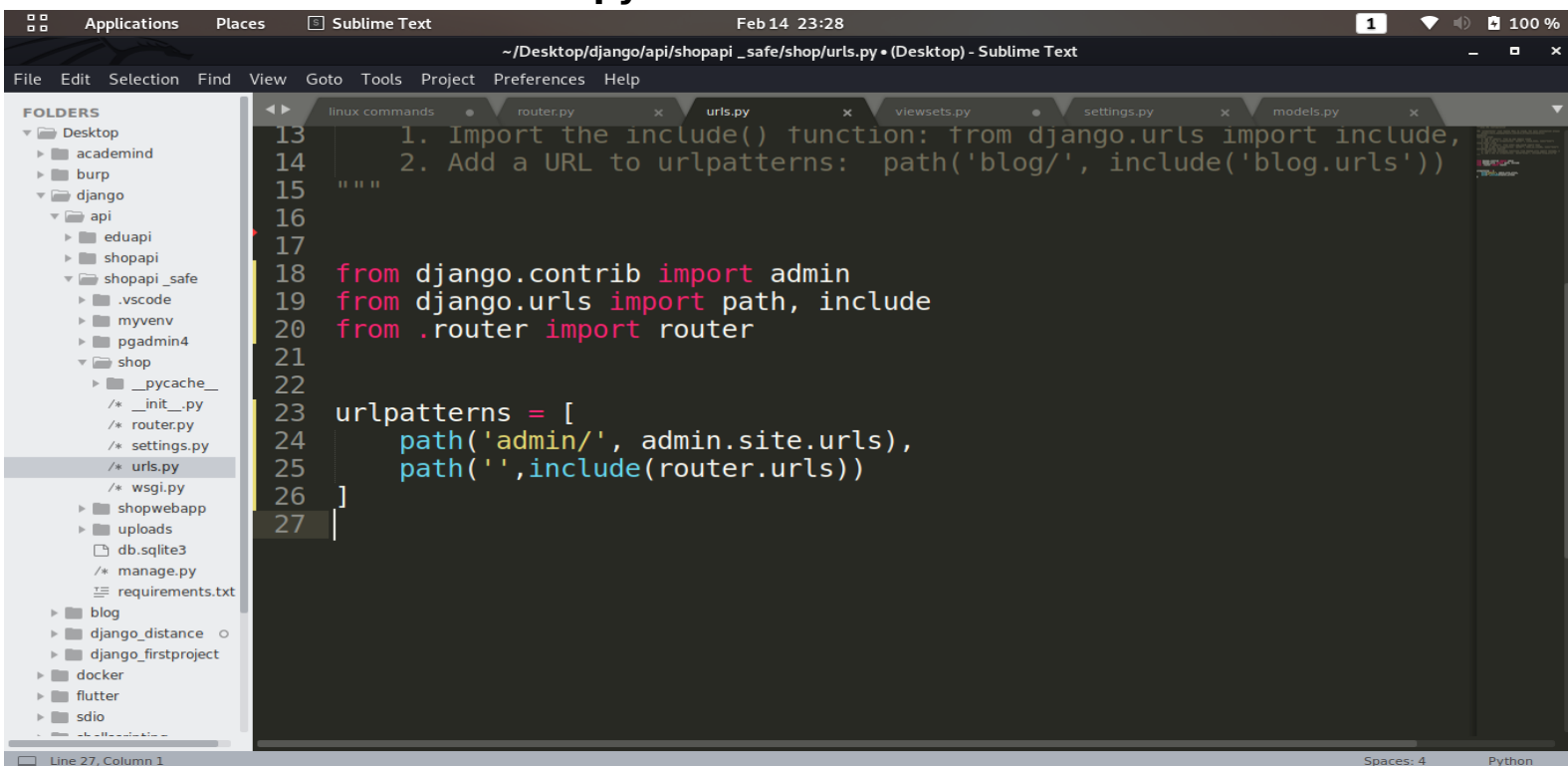
- Desktop
  - academind
  - burp
  - django
    - api
      - eduapi
      - shopapi
        - shopapi\_safe
          - .vscode
          - myenv
          - pgadmin4
          - shop
            - \_\_pycache\_\_
            - \_\_init\_\_.py
            - router.py
            - settings.py
            - urls.py
            - wsgi.py
          - shopwebapp
          - uploads
          - db.sqlite3
          - manage.py
          - requirements.txt
        - blog
        - django\_distance
        - django\_firstproject
      - docker
      - flutter
      - sdio

The main editor area shows the following Python code in `router.py`:

```
1 from shopwebapp.viewsets import ProductViewSet, CategoryViewSet
2 from rest_framework import routers
3
4 router = routers.DefaultRouter()
5 router.register('product', ProductViewSet)
6 router.register('category', CategoryViewSet)
```

The status bar at the bottom indicates "Line 6, Column 44", "Tab Size: 4", and "Python".

## ----- urls.py file-----



The screenshot shows the Sublime Text editor with the file `~/Desktop/django/api/shopapi_safe/shop/urls.py` open. The left sidebar displays a folder tree with the following structure:

- Desktop
  - academind
  - burp
  - django
    - api
      - eduapi
      - shopapi
        - shopapi\_safe
          - .vscode
          - myenv
          - pgadmin4
          - shop
            - \_\_pycache\_\_
            - \_\_init\_\_.py
            - router.py
            - settings.py
            - urls.py
            - wsgi.py
          - shopwebapp
          - uploads
          - db.sqlite3
          - manage.py
          - requirements.txt
        - blog
        - django\_distance
        - django\_firstproject
      - docker
      - flutter
      - sdio

The main editor area shows the following Python code in `urls.py`:

```
13 1. Import the include() function: from django.urls import include,
14 2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
15 """
16
17
18 from django.contrib import admin
19 from django.urls import path, include
20 from .router import router
21
22
23 urlpatterns = [
24     path('admin/', admin.site.urls),
25     path('', include(router.urls))
26 ]
27
```

The status bar at the bottom indicates "Line 27, Column 1", "Spaces: 4", and "Python".