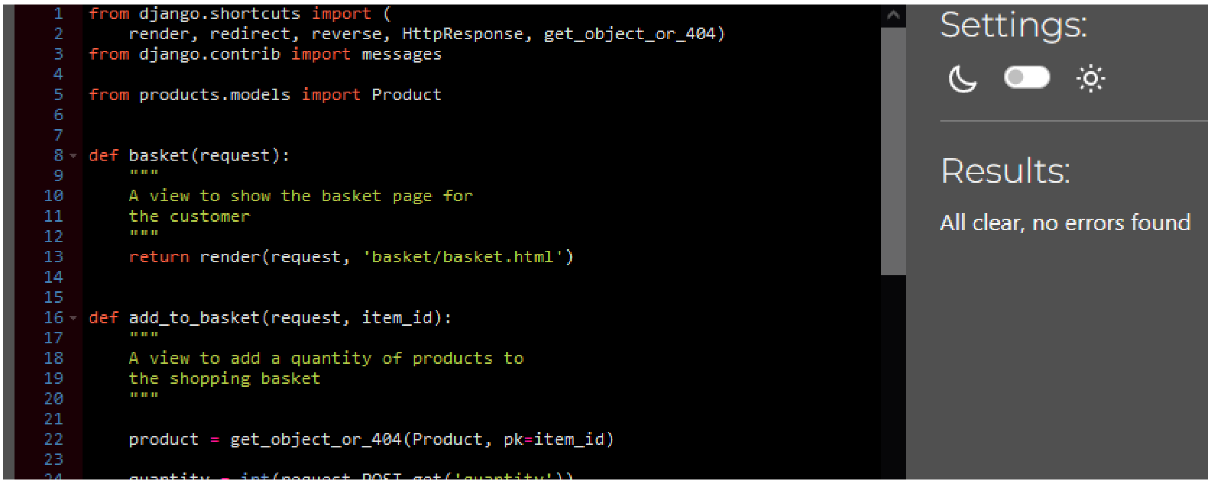
Basket

views.py

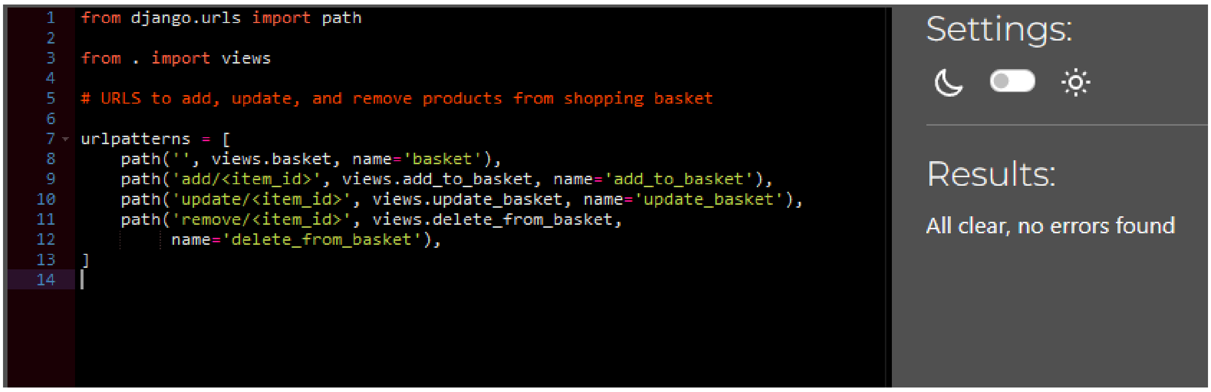


```python
from django.shortcuts import (
    render, redirect, reverse, HttpResponse, get_object_or_404)
from django.contrib import messages

from products.models import Product


def basket(request):
    """
    A view to show the basket page for
    the customer
    """
    return render(request, 'basket/basket.html')


def add_to_basket(request, item_id):
    """
    A view to add a quantity of products to
    the shopping basket
    """

    product = get_object_or_404(Product, pk=item_id)

    quantity = int(request.POST.get('quantity'))
```

Results:

All clear, no errors found

urls.py

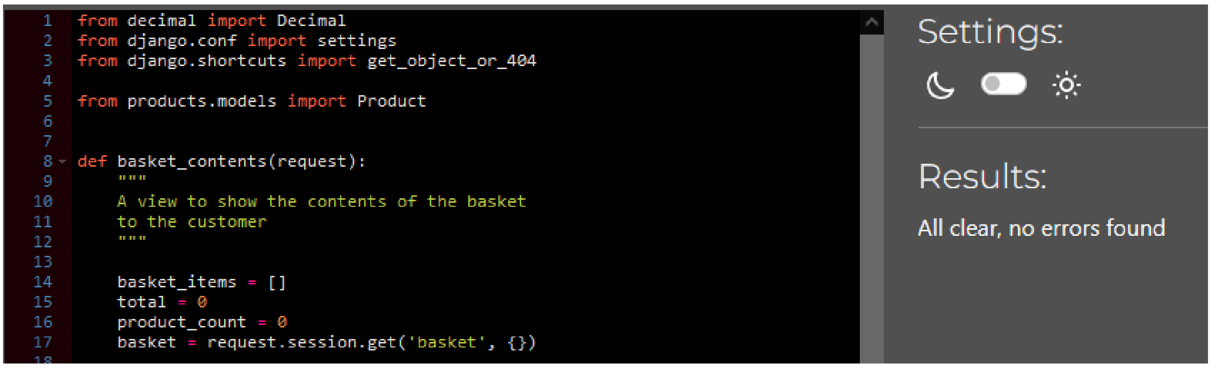


```python
from django.urls import path

from . import views

# URLS to add, update, and remove products from shopping basket

urlpatterns = [
    path('', views.basket, name='basket'),
    path('add/<item_id>', views.add_to_basket, name='add_to_basket'),
    path('update/<item_id>', views.update_basket, name='update_basket'),
    path('remove/<item_id>', views.delete_from_basket,
         name='delete_from_basket'),
]

```

Results:

All clear, no errors found

context.py



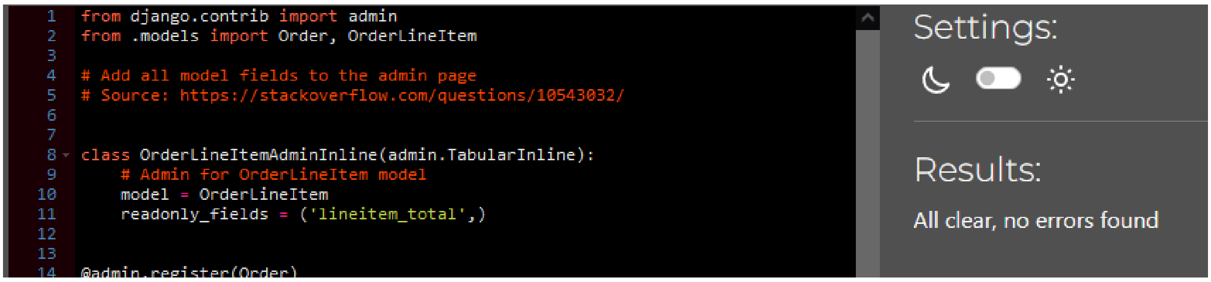
```python
from decimal import Decimal
from django.conf import settings
from django.shortcuts import get_object_or_404

from products.models import Product


def basket_contents(request):
    """
    A view to show the contents of the basket
    to the customer
    """

    basket_items = []
    total = 0
    product_count = 0
    basket = request.session.get('basket', {})
```

Results:

All clear, no errors found

Checkout

admin.py

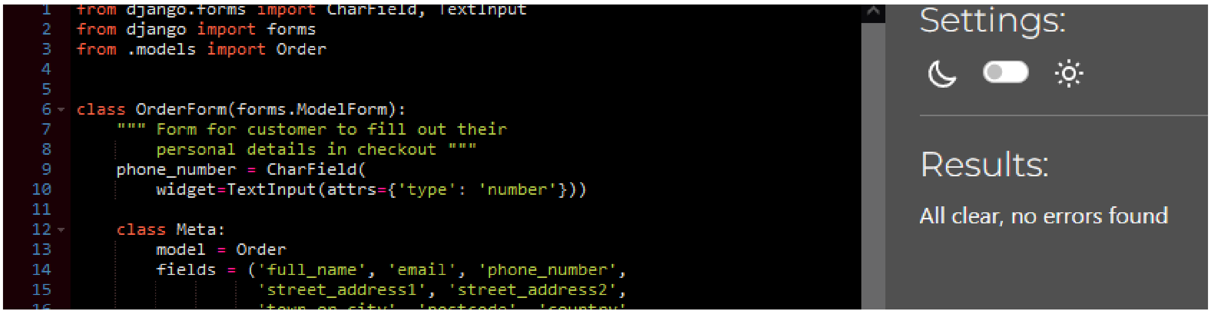


```python
from django.contrib import admin
from .models import Order, OrderLineItem

# Add all model fields to the admin page
# Source: https://stackoverflow.com/questions/10543032/


class OrderLineItemAdminInline(admin.TabularInline):
    # Admin for OrderLineItem model
    model = OrderLineItem
    readonly_fields = ('lineitem_total',)


@admin.register(Order)
```

Results:

All clear, no errors found

forms.py

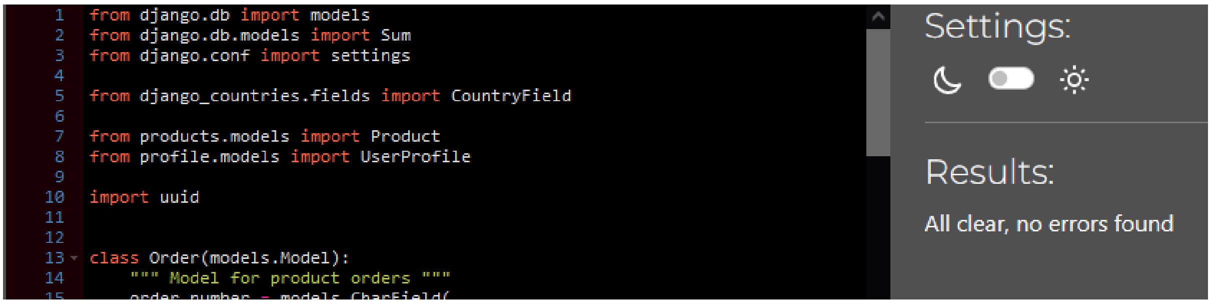


```python
from django.forms import CharField, TextInput
from django import forms
from .models import Order


class OrderForm(forms.ModelForm):
    """ Form for customer to fill out their
        personal details in checkout """
    phone_number = CharField(
        widget=TextInput(attrs={'type': 'number'}))

    class Meta:
        model = Order
        fields = ('full_name', 'email', 'phone_number',
                  'street_address1', 'street_address2',
                  'town_or_city', 'postcode', 'country',
```

Results:

All clear, no errors found

models.py

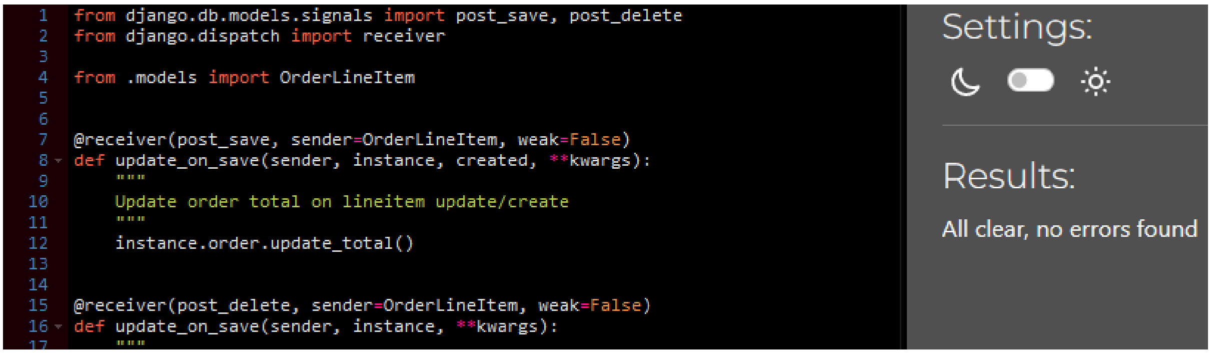


```python
from django.db import models
from django.db.models import Sum
from django.conf import settings

from django_countries.fields import CountryField

from products.models import Product
from profile.models import UserProfile

import uuid


class Order(models.Model):
    """ Model for product orders """
    order_number = models.CharField(
```

Results:

All clear, no errors found

signals.py

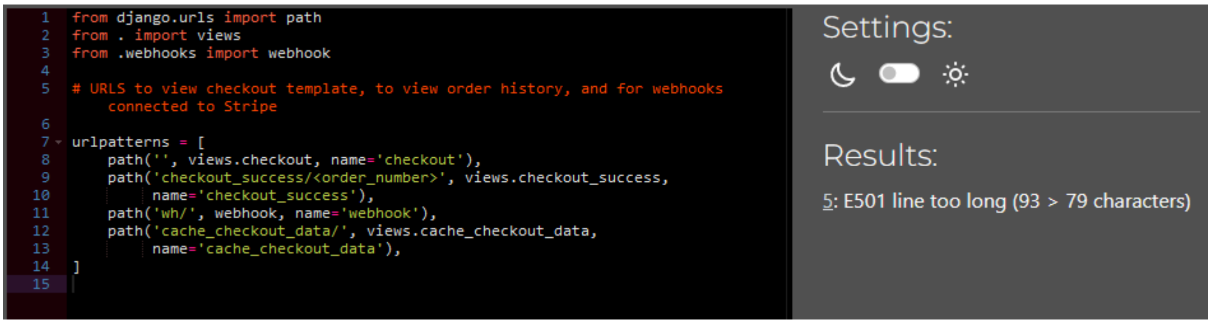


```python
from django.db.models.signals import post_save, post_delete
from django.dispatch import receiver

from .models import OrderLineItem


@receiver(post_save, sender=OrderLineItem, weak=False)
def update_on_save(sender, instance, created, **kwargs):
    """
    Update order total on lineitem update/create
    """
    instance.order.update_total()


@receiver(post_delete, sender=OrderLineItem, weak=False)
def update_on_save(sender, instance, **kwargs):
    """
```

Results:

All clear, no errors found

urls.py - Fixed

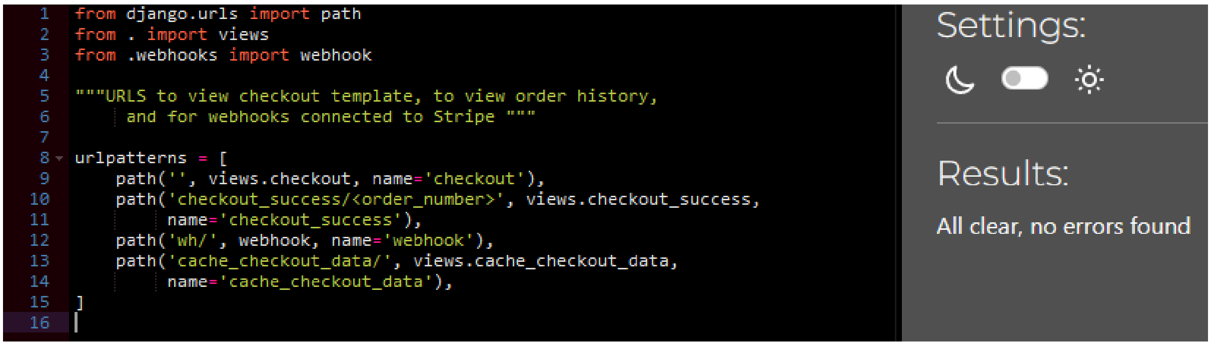


```
from django.urls import path
from . import views
from .webhooks import webhook

# URLS to view checkout template, to view order history, and for webhooks connected to Stripe

urlpatterns = [
    path('', views.checkout, name='checkout'),
    path('checkout_success/<order_number>', views.checkout_success,
         name='checkout_success'),
    path('wh/', webhook, name='webhook'),
    path('cache_checkout_data/', views.cache_checkout_data,
         name='cache_checkout_data'),
]
```

Settings:

Results:

5: E501 line too long (93 > 79 characters)



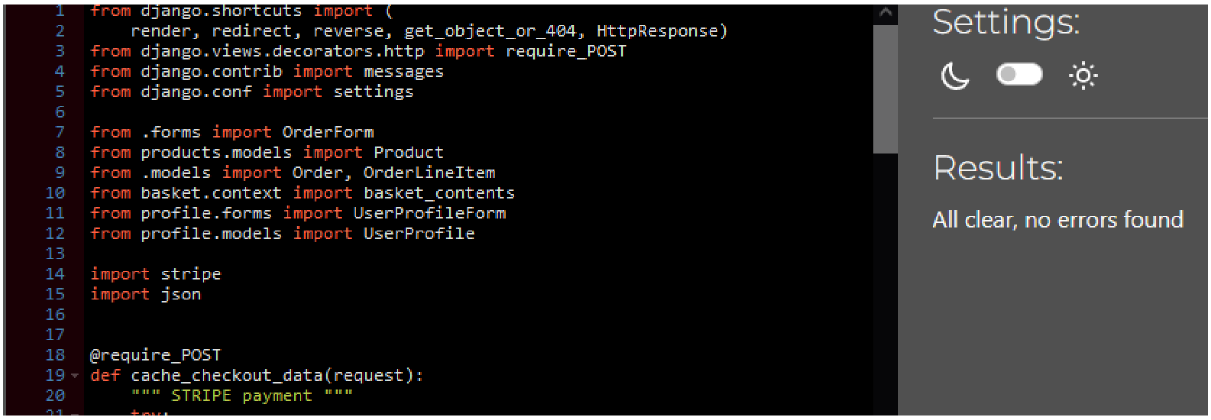
```
from django.urls import path
from . import views
from .webhooks import webhook

"""URLS to view checkout template, to view order history,
    and for webhooks connected to Stripe """

urlpatterns = [
    path('', views.checkout, name='checkout'),
    path('checkout_success/<order_number>', views.checkout_success,
         name='checkout_success'),
    path('wh/', webhook, name='webhook'),
    path('cache_checkout_data/', views.cache_checkout_data,
         name='cache_checkout_data'),
]
```

Settings:

Results:

All clear, no errors found

views.py



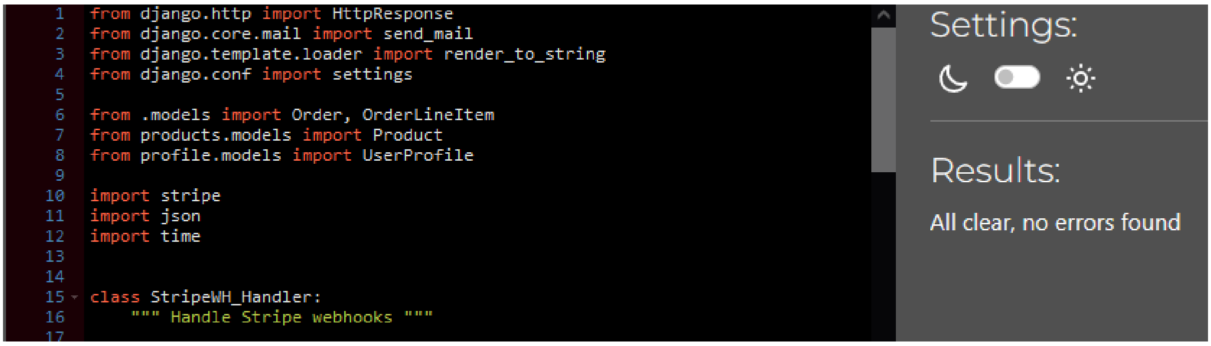
```
from django.shortcuts import (
    render, redirect, reverse, get_object_or_404, HttpResponse)
from django.views.decorators.http import require_POST
from django.contrib import messages
from django.conf import settings

from .forms import OrderForm
from products.models import Product
from .models import Order, OrderLineItem
from basket.context import basket_contents
from profile.forms import UserProfileForm
from profile.models import UserProfile

import stripe
import json


@require_POST
def cache_checkout_data(request):
    """ STRIPE payment """
    try:
```

Settings:

Results:

All clear, no errors found

webhook_handler.py



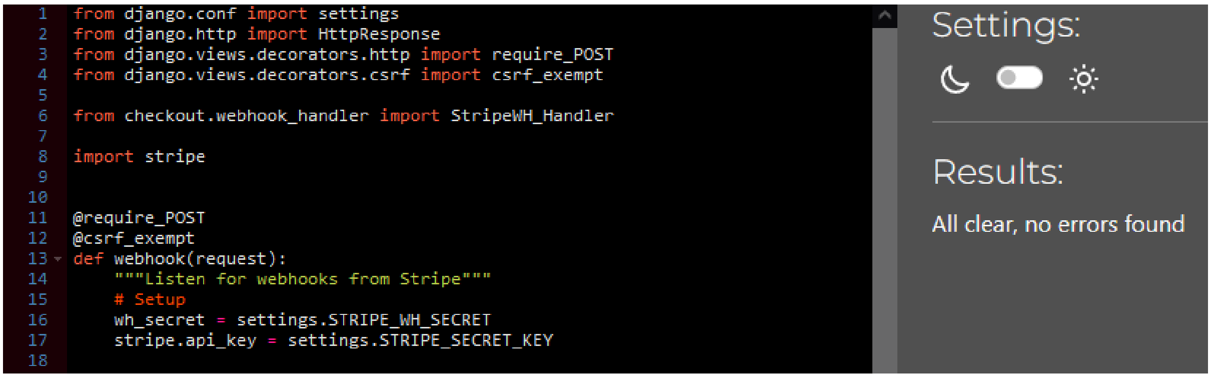
```
from django.http import HttpResponse
from django.core.mail import send_mail
from django.template.loader import render_to_string
from django.conf import settings

from .models import Order, OrderLineItem
from products.models import Product
from profile.models import UserProfile

import stripe
import json
import time


class StripeWH_Handler:
    """ Handle Stripe webhooks """
```

Settings:

Results:

All clear, no errors found

webhooks.py



```
from django.conf import settings
from django.http import HttpResponse
from django.views.decorators.http import require_POST
from django.views.decorators.csrf import csrf_exempt

from checkout.webhook_handler import StripeWH_Handler

import stripe


@require_POST
@csrf_exempt
def webhook(request):
    """Listen for webhooks from Stripe"""
    # Setup
    wh_secret = settings.STRIPE_WH_SECRET
    stripe.api_key = settings.STRIPE_SECRET_KEY

```

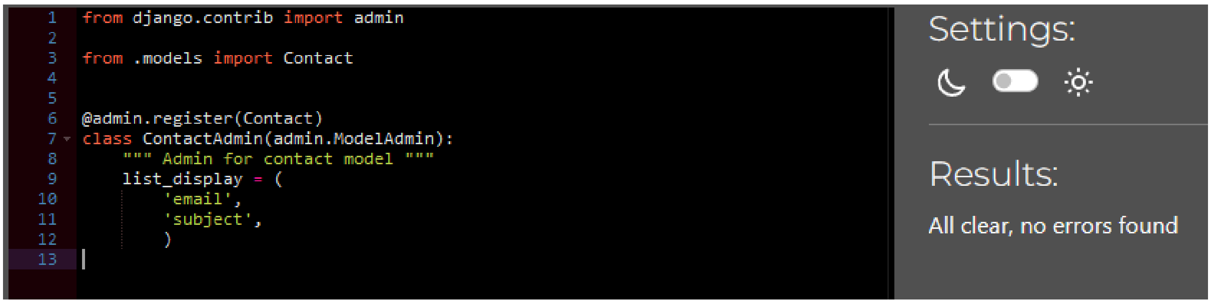
Settings:

Results:

All clear, no errors found

Company

admin.py



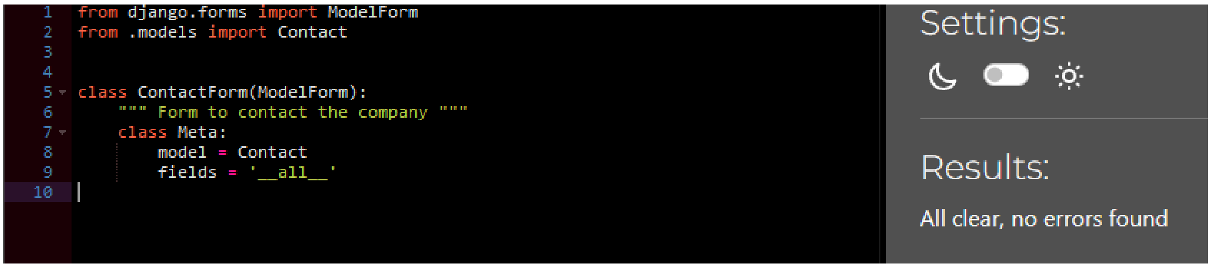
```
from django.contrib import admin

from .models import Contact


@admin.register(Contact)
class ContactAdmin(admin.ModelAdmin):
    """ Admin for contact model """
    list_display = (
        'email',
        'subject',
        )

```

Settings:

Results:

All clear, no errors found

forms.py



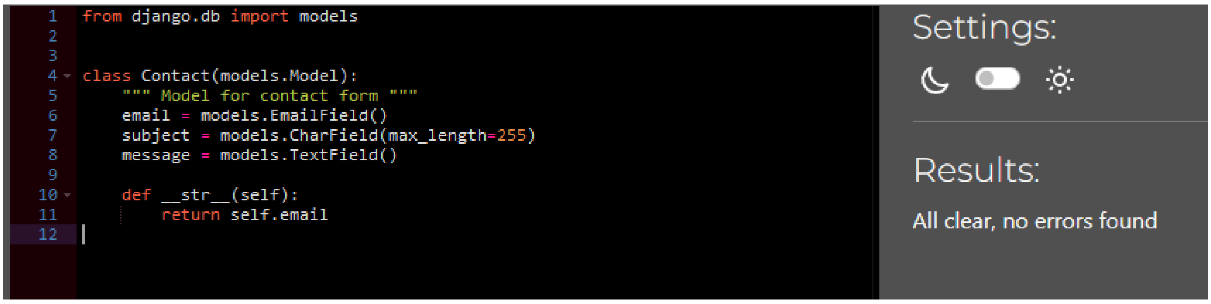
```
from django.forms import ModelForm
from .models import Contact


class ContactForm(ModelForm):
    """ Form to contact the company """
    class Meta:
        model = Contact
        fields = '__all__'

```

Settings:

Results:

All clear, no errors found

models.py



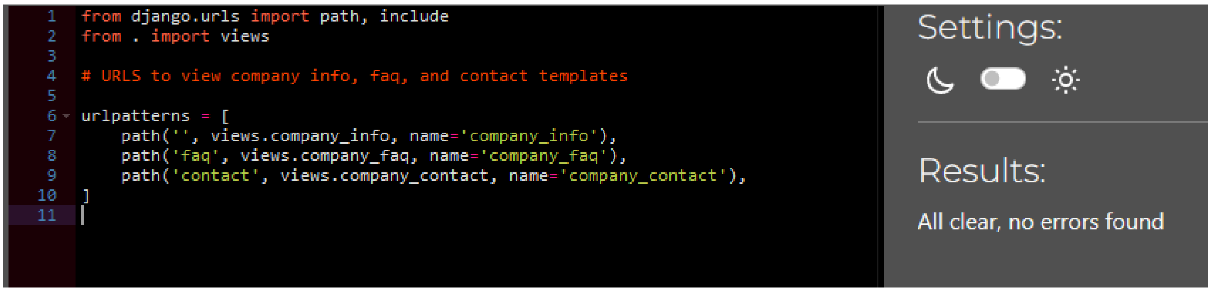
```
from django.db import models


class Contact(models.Model):
    """ Model for contact form """
    email = models.EmailField()
    subject = models.CharField(max_length=255)
    message = models.TextField()

    def __str__(self):
        return self.email

```

Settings:

Results:

All clear, no errors found

urls.py

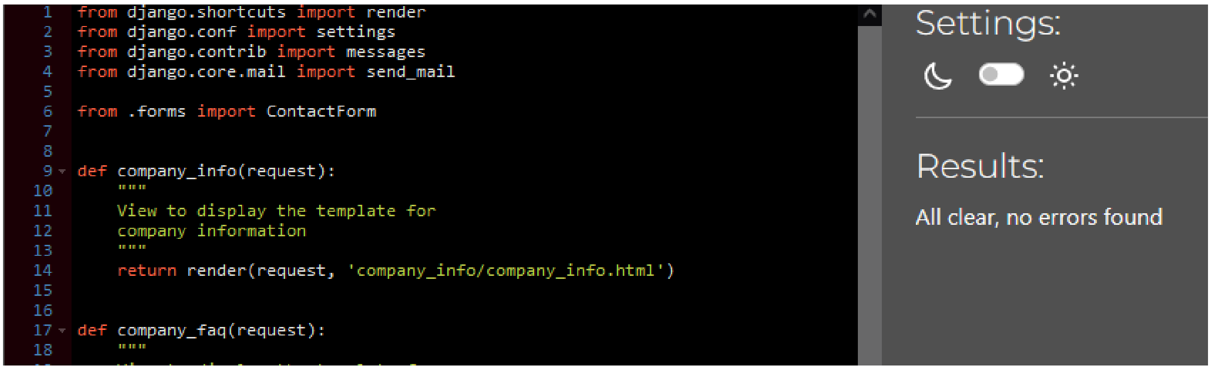


```python
from django.urls import path, include
from . import views

# URLS to view company info, faq, and contact templates

urlpatterns = [
    path('', views.company_info, name='company_info'),
    path('faq', views.company_faq, name='company_faq'),
    path('contact', views.company_contact, name='company_contact'),
]
```

Results:

All clear, no errors found

views.py



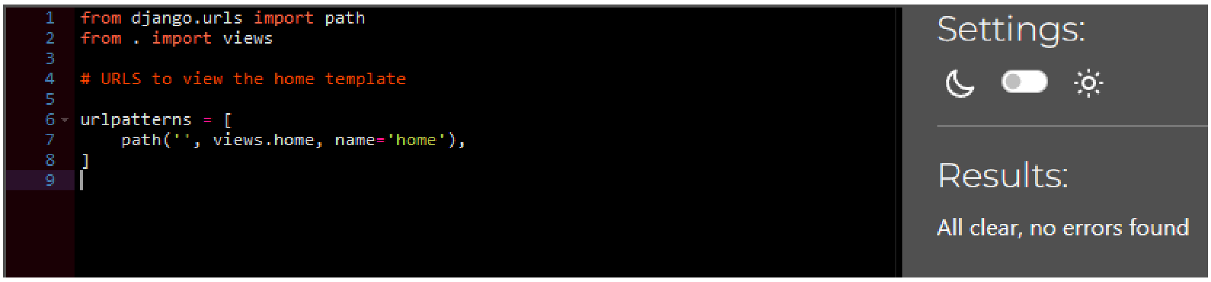
```python
from django.shortcuts import render
from django.conf import settings
from django.contrib import messages
from django.core.mail import send_mail

from .forms import ContactForm


def company_info(request):
    """
    View to display the template for
    company information
    """
    return render(request, 'company_info/company_info.html')


def company_faq(request):
    """
```

Results:

All clear, no errors found

Home

urls.py

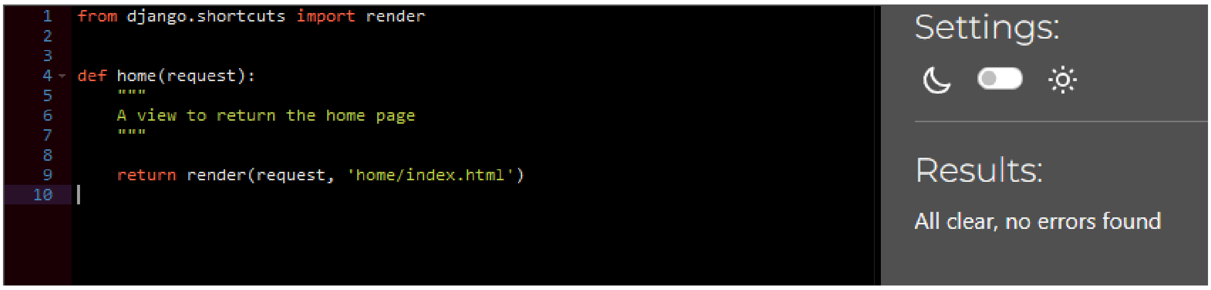


```python
from django.urls import path
from . import views

# URLS to view the home template

urlpatterns = [
    path('', views.home, name='home'),
]
```

Results:

All clear, no errors found

views.py



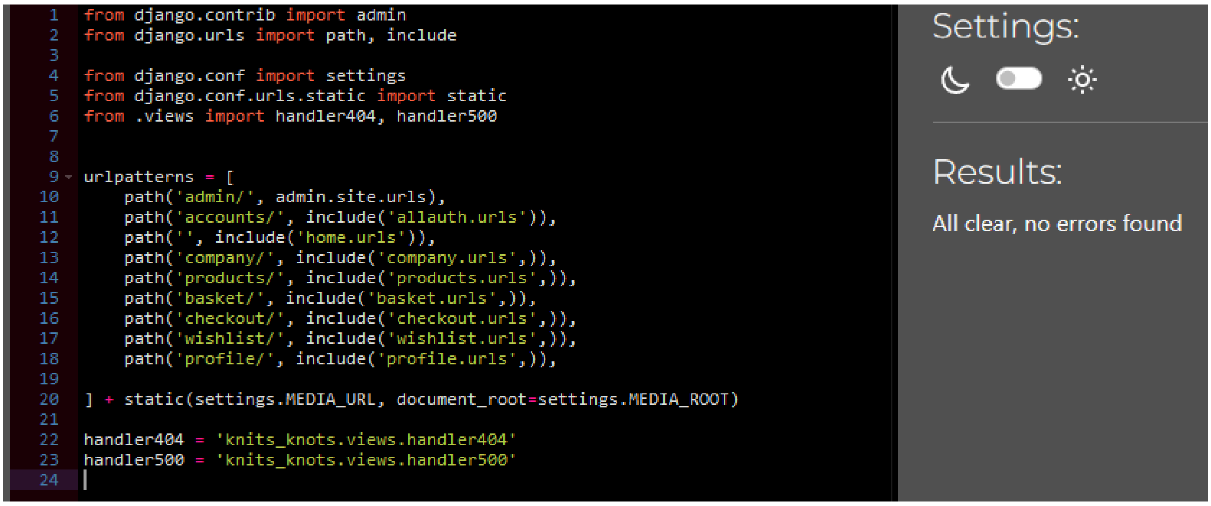
```python
from django.shortcuts import render


def home(request):
    """
    A view to return the home page
    """

    return render(request, 'home/index.html')
```

Results:

All clear, no errors found

Knits&Knots

urls.py

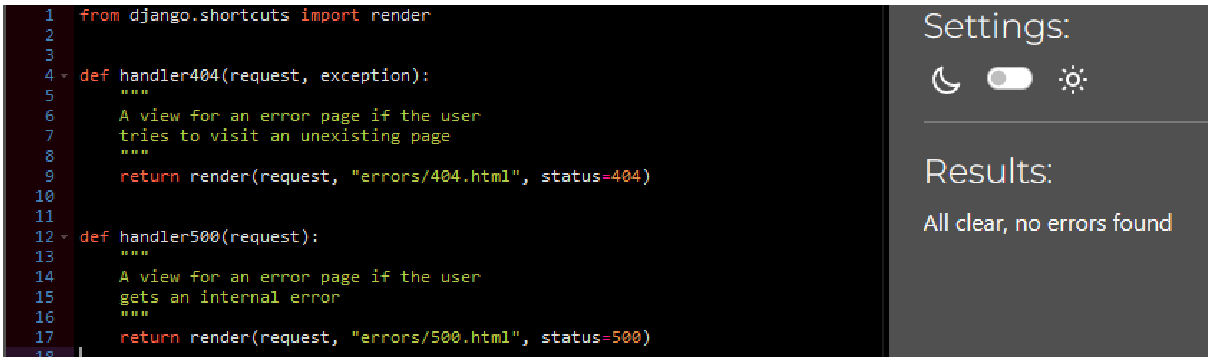


```python
from django.contrib import admin
from django.urls import path, include

from django.conf import settings
from django.conf.urls.static import static
from .views import handler404, handler500


urlpatterns = [
    path('admin/', admin.site.urls),
    path('accounts/', include('allauth.urls')),
    path('', include('home.urls')),
    path('company/', include('company.urls',)),
    path('products/', include('products.urls',)),
    path('basket/', include('basket.urls',)),
    path('checkout/', include('checkout.urls',)),
    path('wishlist/', include('wishlist.urls',)),
    path('profile/', include('profile.urls',)),

] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)

handler404 = 'knits_knots.views.handler404'
handler500 = 'knits_knots.views.handler500'

```

Results:

All clear, no errors found

views.py



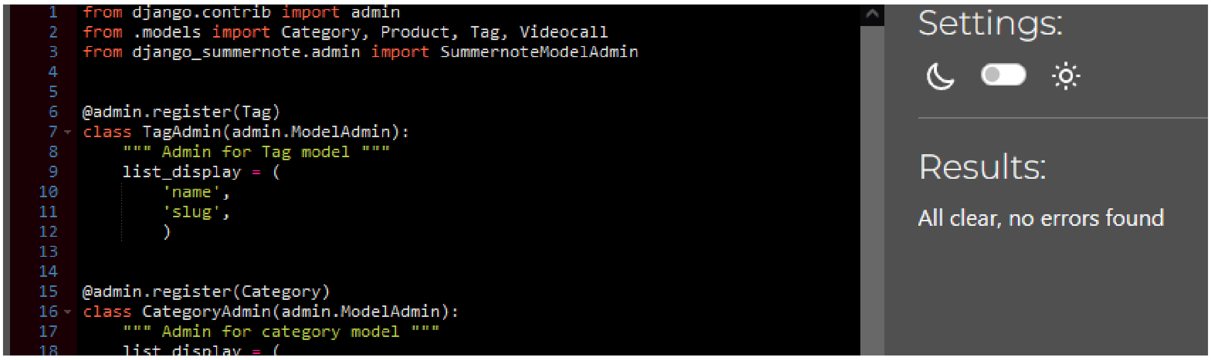
```python
from django.shortcuts import render


def handler404(request, exception):
    """
    A view for an error page if the user
    tries to visit an unexisting page
    """
    return render(request, "errors/404.html", status=404)


def handler500(request):
    """
    A view for an error page if the user
    gets an internal error
    """
    return render(request, "errors/500.html", status=500)
```

Results:

All clear, no errors found

Products

admin.py

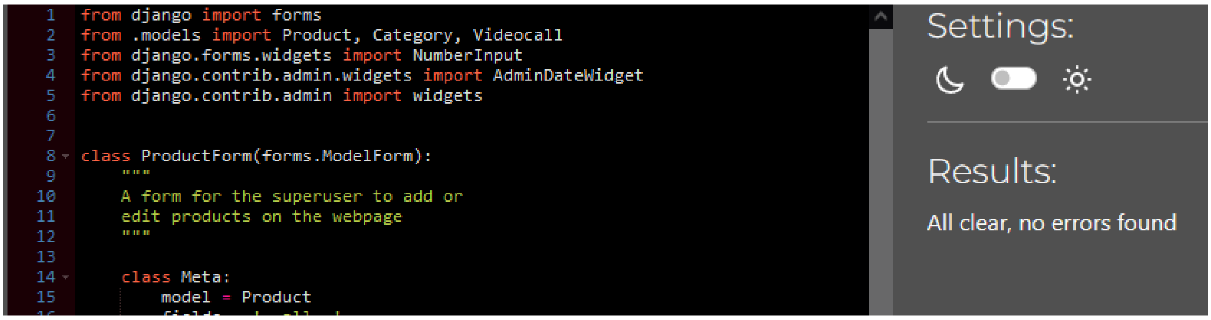


```python
from django.contrib import admin
from .models import Category, Product, Tag, Videocall
from django_summernote.admin import SummernoteModelAdmin


@admin.register(Tag)
class TagAdmin(admin.ModelAdmin):
    """ Admin for Tag model """
    list_display = (
        'name',
        'slug',
        )


@admin.register(Category)
class CategoryAdmin(admin.ModelAdmin):
    """ Admin for category model """
    list_display = (
```

Results:

All clear, no errors found

forms.py



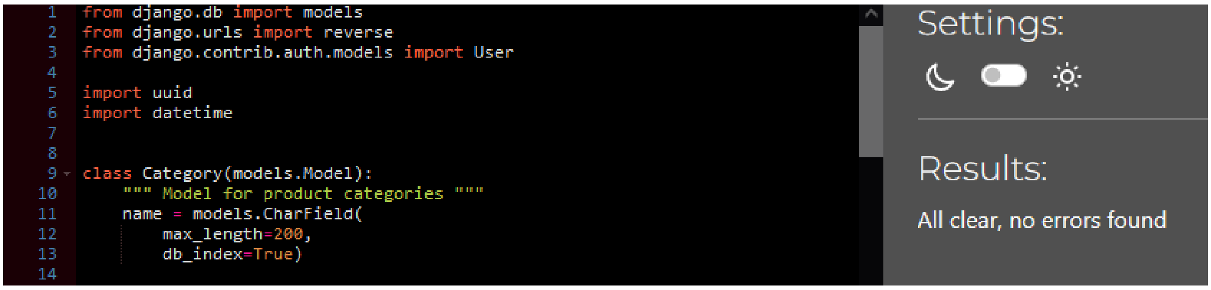
```
from django import forms
from .models import Product, Category, Videocall
from django.forms.widgets import NumberInput
from django.contrib.admin.widgets import AdminDateWidget
from django.contrib.admin import widgets


class ProductForm(forms.ModelForm):
    """
    A form for the superuser to add or
    edit products on the webpage
    """

    class Meta:
        model = Product
        fields = '__all__'
```

Settings:

Results:

All clear, no errors found

models.py



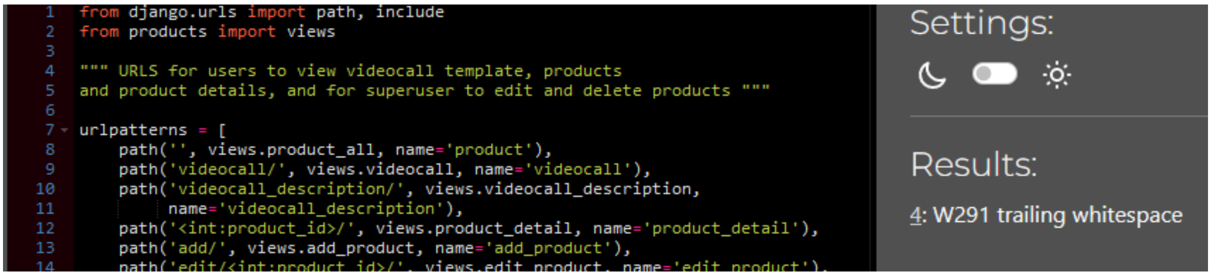
```
from django.db import models
from django.urls import reverse
from django.contrib.auth.models import User

import uuid
import datetime


class Category(models.Model):
    """ Model for product categories """
    name = models.CharField(
        max_length=200,
        db_index=True)

```

Settings:

Results:

All clear, no errors found

urls.py - Fixed

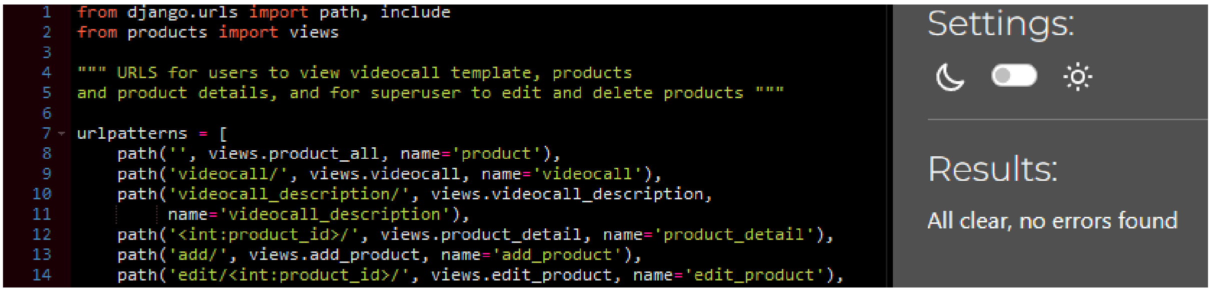


```
from django.urls import path, include
from products import views

""" URLS for users to view videocall template, products
and product details, and for superuser to edit and delete products """

urlpatterns = [
    path('', views.product_all, name='product'),
    path('videocall/', views.videocall, name='videocall'),
    path('videocall_description/', views.videocall_description,
         name='videocall_description'),
    path('<int:product_id>/', views.product_detail, name='product_detail'),
    path('add/', views.add_product, name='add_product'),
    path('edit/<int:product_id>/', views.edit_product, name='edit_product'),
```

Settings:

Results:

4: W291 trailing whitespace



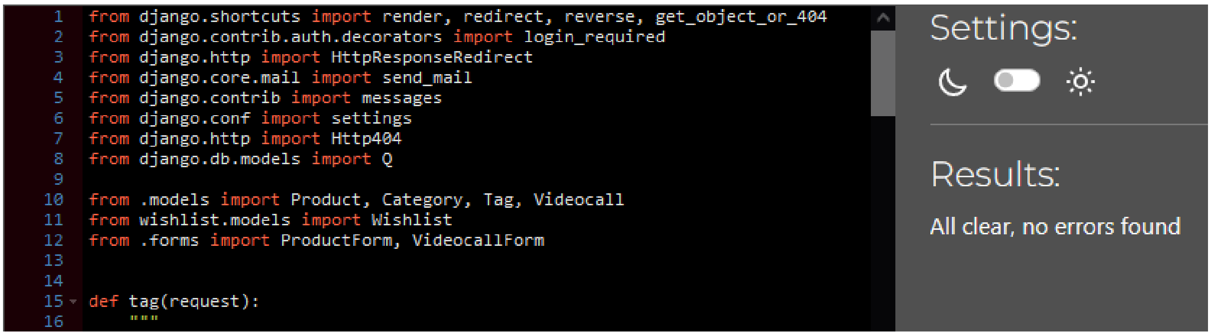
```
from django.urls import path, include
from products import views

""" URLS for users to view videocall template, products
and product details, and for superuser to edit and delete products """

urlpatterns = [
    path('', views.product_all, name='product'),
    path('videocall/', views.videocall, name='videocall'),
    path('videocall_description/', views.videocall_description,
         name='videocall_description'),
    path('<int:product_id>/', views.product_detail, name='product_detail'),
    path('add/', views.add_product, name='add_product'),
    path('edit/<int:product_id>/', views.edit_product, name='edit_product'),
```

Settings:

Results:

All clear, no errors found

views.py



```
from django.shortcuts import render, redirect, reverse, get_object_or_404
from django.contrib.auth.decorators import login_required
from django.http import HttpResponseRedirect
from django.core.mail import send_mail
from django.contrib import messages
from django.conf import settings
from django.http import Http404
from django.db.models import Q

from .models import Product, Category, Tag, Videocall
from wishlist.models import Wishlist
from .forms import ProductForm, VideocallForm


def tag(request):
    """
```

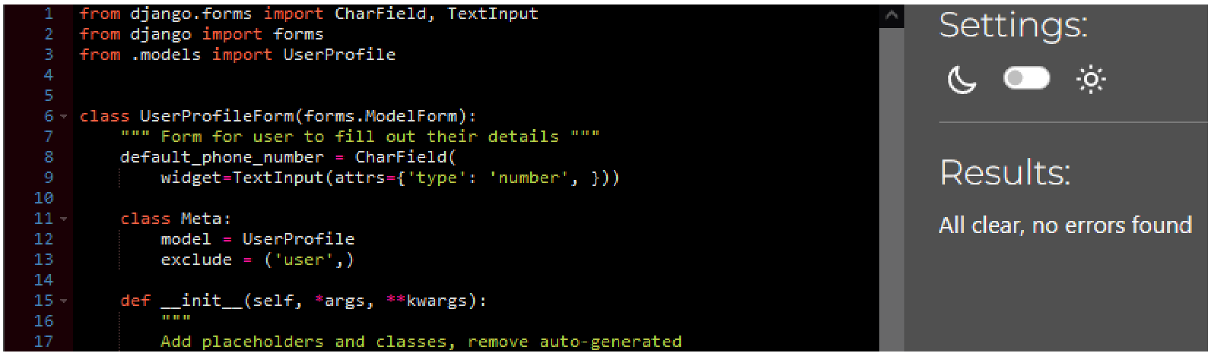
Settings:

Results:

All clear, no errors found

Profile

forms.py



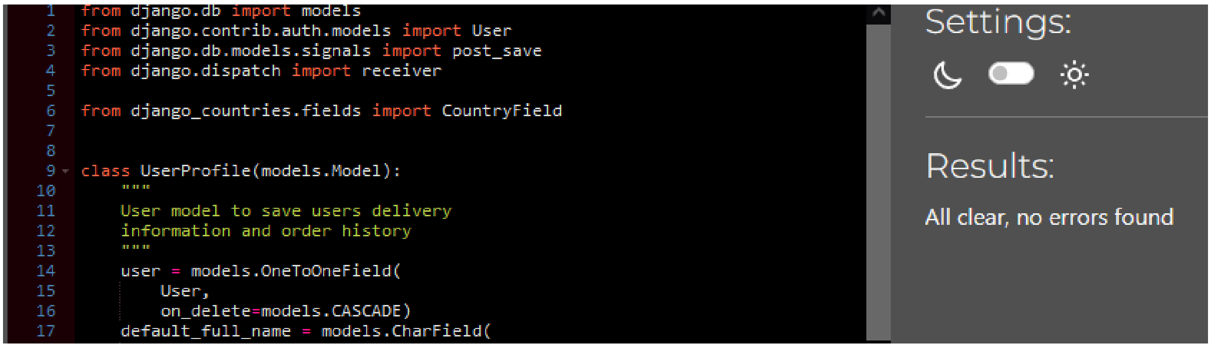
```
from django.forms import CharField, TextInput
from django import forms
from .models import UserProfile


class UserProfileForm(forms.ModelForm):
    """ Form for user to fill out their details """
    default_phone_number = CharField(
        widget=TextInput(attrs={'type': 'number', }))

    class Meta:
        model = UserProfile
        exclude = ('user',)

    def __init__(self, *args, **kwargs):
        """
        Add placeholders and classes, remove auto-generated
```

Settings:

Results:

All clear, no errors found

models.py



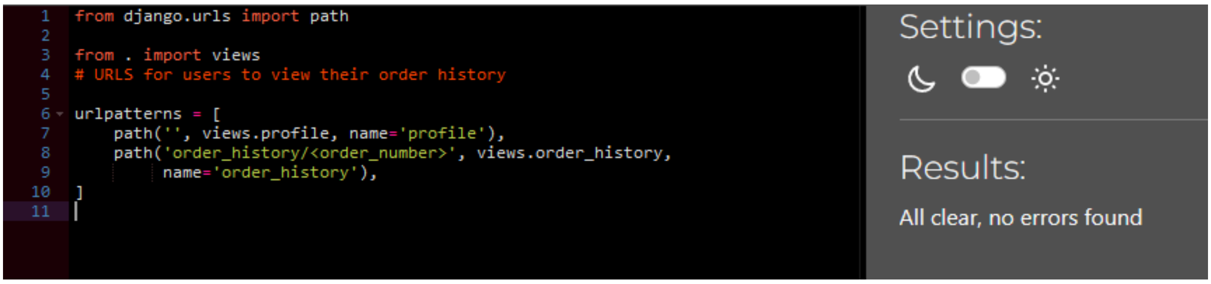
```
from django.db import models
from django.contrib.auth.models import User
from django.db.models.signals import post_save
from django.dispatch import receiver

from django_countries.fields import CountryField


class UserProfile(models.Model):
    """
    User model to save users delivery
    information and order history
    """
    user = models.OneToOneField(
        User,
        on_delete=models.CASCADE)
    default_full_name = models.CharField(
```

Settings:

Results:

All clear, no errors found

urls.py



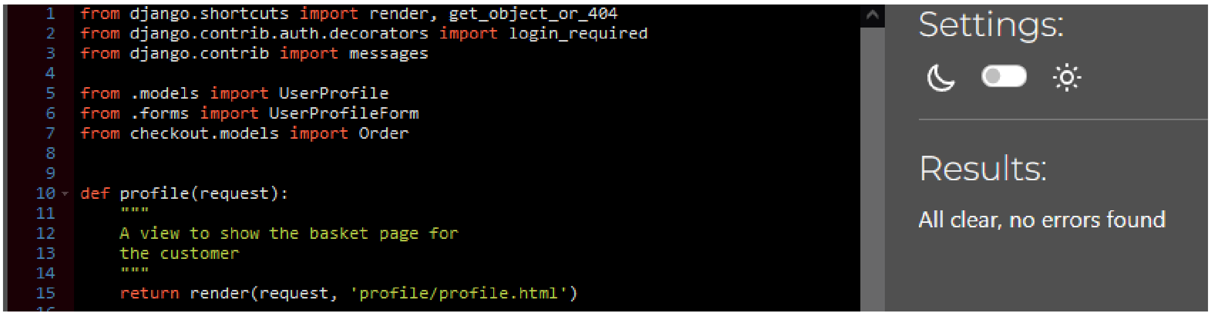
```
from django.urls import path

from . import views
# URLS for users to view their order history

urlpatterns = [
    path('', views.profile, name='profile'),
    path('order_history/<order_number>', views.order_history,
         name='order_history'),
]

```

Settings:

Results:

All clear, no errors found

views.py



```
from django.shortcuts import render, get_object_or_404
from django.contrib.auth.decorators import login_required
from django.contrib import messages

from .models import UserProfile
from .forms import UserProfileForm
from checkout.models import Order


def profile(request):
    """
    A view to show the basket page for
    the customer
    """
    return render(request, 'profile/profile.html')
```

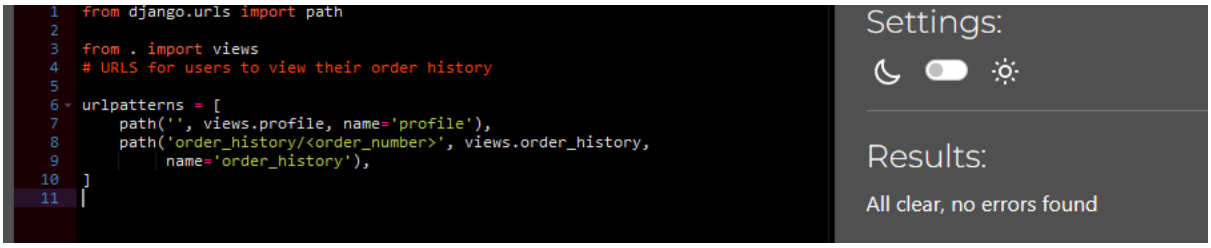
Settings:

Results:

All clear, no errors found

Wishlist

urls.py



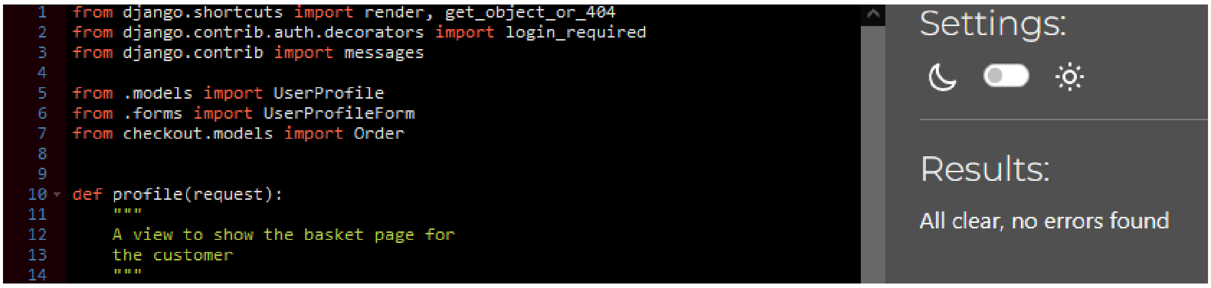
```
from django.urls import path

from . import views
# URLS for users to view their order history

urlpatterns = [
    path('', views.profile, name='profile'),
    path('order_history/<order_number>', views.order_history,
         name='order_history'),
]

```

Settings:

Results:

All clear, no errors found

views.py



```
from django.shortcuts import render, get_object_or_404
from django.contrib.auth.decorators import login_required
from django.contrib import messages

from .models import UserProfile
from .forms import UserProfileForm
from checkout.models import Order


def profile(request):
    """
    A view to show the basket page for
    the customer
    """
```

Settings:

Results:

All clear, no errors found