


admin.py

admin.py - abc-bookshop - Git

CI Python Linter

Administration panel | ABC Boo




pep8ci.herokuapp.com



CI Python Linter

```
26
27
28     "order_number",
29     "user_profile",
30     "date",
31     "full_name",
32     "email",
33     "phone_number",
34     "country",
35     "postcode",
36     "town_or_city",
37     "street_address1",
38     "street_address2",
39     "county",
40     "delivery_cost",
41     "order_total",
42     "grand_total",
43     "original_bag",
44     "stripe_pid",
45 )
46
47 list_display = (
48     "order_number",
49     "date",
50     "full_name",
51     "order_total",
52     "delivery_cost",
53     "grand_total",
54     "user_profile",
55 )
56
57 ordering = ("-date",)
58
59
60 admin.site.register(Order, OrderAdmin)
61
```

Settings:




Results:

All clear, no errors found

apps.py

apps.py - abc-bookshop - Gitp x CI Python Linter x Administration panel | ABC Boo x +

← → ↻ pep8ci.herokuapp.com



CI Python Linter

```
1 from django.apps import AppConfig
2
3
4 class CheckoutConfig(AppConfig):
5     name = 'checkout'
6
7     def ready(self):
8         import checkout.signals
9
```

Settings:

🌙 ☒ 🌞


Results:

All clear, no errors found

urls.py

urls.py - abc-bookshop - Gitpod x CI Python Linter x Administration panel | ABC Boo x +


← → ↻ pep8ci.herokuapp.com



CI Python Linter

```
1 from django.urls import path
2 from . import views
3 from .webhooks import webhook
4
5 urlpatterns = [
6     path('', views.checkout, name='checkout'),
7     path(
8         'checkout_success/<order_number>',
9         views.checkout_success, name='checkout_success'
10    ),
11     path(
12         'cache_checkout_data/',
13         views.cache_checkout_data,
14         name='cache_checkout_data'
15    ),
16     path('wh/', webhook, name='webhook'),
17 ]
18 |
```

Settings:

🌙 ☒ 

Results:

All clear, no errors found

views.py

views.py - abc-bookshop - Gitp x CI Python Linter x Administration panel | ABC Boo x +

pep8ci.herokuapp.com

code institute

CI Python Linter

```
145
146 if request.user.is_authenticated:
147     profile = UserProfile.objects.get(user=request.user)
148     # Attach the user's profile to the order
149     order.user_profile = profile
150     order.save()
151
152     # Save the user's info
153     if save_info:
154         profile_data = {
155             'default_phone_number': order.phone_number,
156             'default_country': order.country,
157             'default_postcode': order.postcode,
158             'default_town_or_city': order.town_or_city,
159             'default_street_address1': order.street_address1,
160             'default_street_address2': order.street_address2,
161             'default_county': order.county,
162         }
163         user_profile_form = UserProfileForm(profile_data, instance=profile)
164         if user_profile_form.is_valid():
165             user_profile_form.save()
166
167     messages.success(request, f'Order successfully processed! \
168         Your order number is {order.number}. A confirmation \
169         email will be sent to {order.email}.')
170
171     if 'bag' in request.session:
172         del request.session['bag']
173
174     template = 'checkout/checkout_success.html'
175     context = {
176         'order': order,
177     }
178
179     return render(request, template, context)
180
```

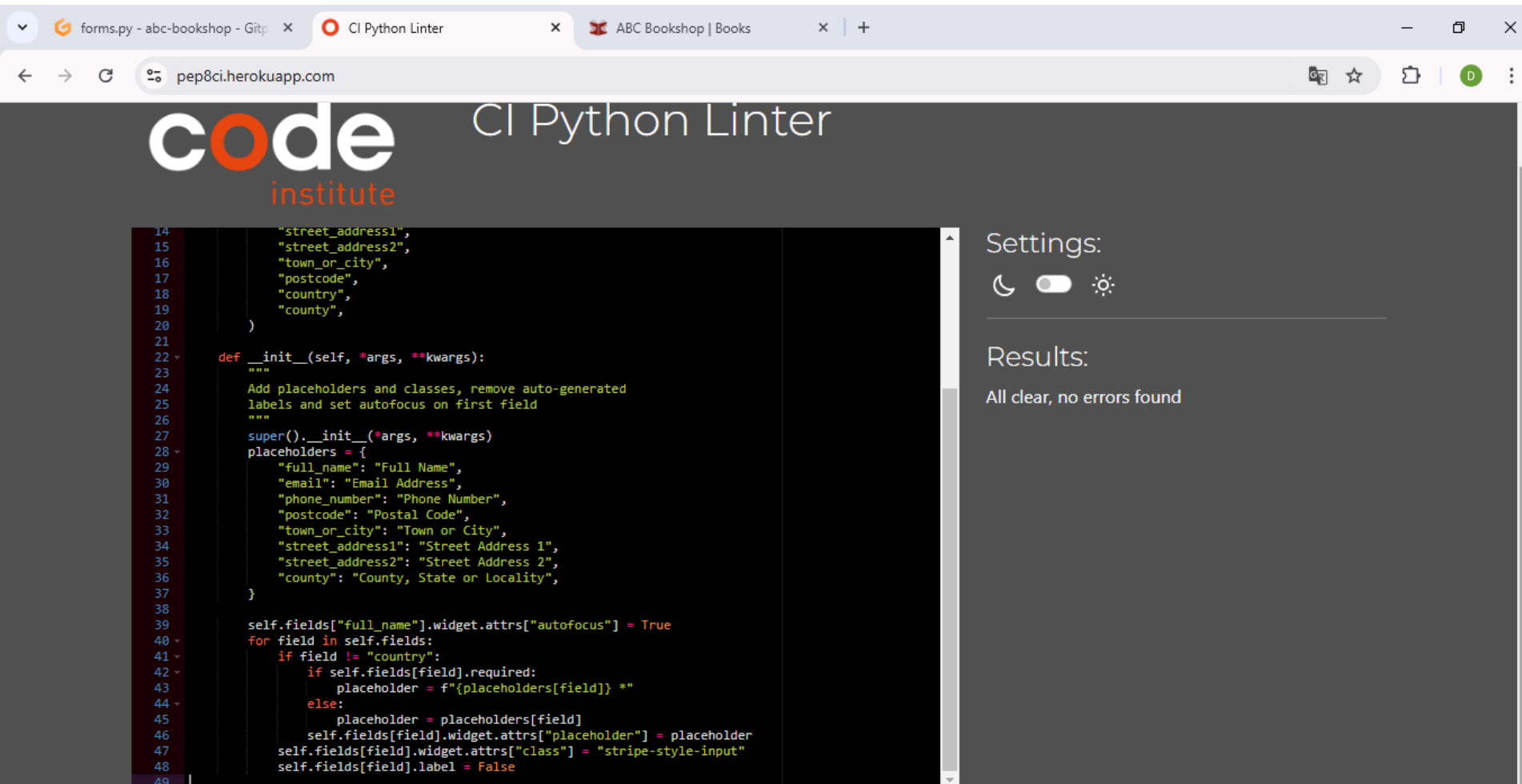
Settings:

☒ Dark mode

Results:

All clear, no errors found

forms.py



forms.py - abc-bookshop - Gitp x CI Python Linter x ABC Bookshop | Books x +

pep8ci.herokuapp.com

code institute

CI Python Linter

```
14         "street_address1",
15         "street_address2",
16         "town_or_city",
17         "postcode",
18         "country",
19         "county",
20     )
21
22     def __init__(self, *args, **kwargs):
23         """
24         Add placeholders and classes, remove auto-generated
25         labels and set autofocus on first field
26         """
27         super().__init__(*args, **kwargs)
28         placeholders = {
29             "full_name": "Full Name",
30             "email": "Email Address",
31             "phone_number": "Phone Number",
32             "postcode": "Postal Code",
33             "town_or_city": "Town or City",
34             "street_address1": "Street Address 1",
35             "street_address2": "Street Address 2",
36             "county": "County, State or Locality",
37         }
38
39         self.fields["full_name"].widget.attrs["autofocus"] = True
40         for field in self.fields:
41             if field != "country":
42                 if self.fields[field].required:
43                     placeholder = f"{placeholders[field]} *"
44                 else:
45                     placeholder = placeholders[field]
46                 self.fields[field].widget.attrs["placeholder"] = placeholder
47                 self.fields[field].widget.attrs["class"] = "stripe-style-input"
48                 self.fields[field].label = False
49
```

Settings:

☒ ☐ ☐

Results:

All clear, no errors found

models.py

The screenshot shows a web browser with the Code Institute CI Python Linter interface. The browser's address bar shows the URL `pep8ci.herokuapp.com`. The page has a dark theme. On the left, a code editor displays a Python file named `models.py` with the following content:

```
73         self.order_number = self._generate_order_number()
74         super().save(*args, **kwargs)
75
76     class Meta:
77         """ Order orders from newest to oldest """
78         ordering = ['-created_on']
79
80     def __str__(self):
81         return self.order_number
82
83
84     class OrderLineItem(models.Model):
85         """ Model for creating/managing Line Items """
86         order = models.ForeignKey(
87             Order, null=False,
88             blank=False, on_delete=models.CASCADE,
89             related_name='lineitems'
90         )
91         product = models.ForeignKey(
92             Product, null=False, blank=False, on_delete=models.CASCADE
93         )
94         quantity = models.IntegerField(null=False, blank=False, default=0)
95         lineitem_total = models.DecimalField(
96             max_digits=6, decimal_places=2, null=False, blank=False, editable=False
97         )
98
99     def save(self, *args, **kwargs):
100         """
101         Override the original save method to set the lineitem total
102         and update the order total.
103         """
104         self.lineitem_total = self.product.final_price * self.quantity
105         super().save(*args, **kwargs)
106
107     def __str__(self):
108         return f'EAN {self.product.ean} on order {self.order.order_number}'
```

On the right side of the interface, there are two sections:

- Settings:** Contains a dark mode toggle switch, which is currently turned on.
- Results:** Displays the message "All clear, no errors found".


signals.py

signals.py - abc-bookshop - Git

CI Python Linter

ABC Bookshop | Books




pep8ci.herokuapp.com



CI Python Linter

```
1 from django.db.models.signals import post_save, post_delete
2 from django.dispatch import receiver
3
4 from .models import OrderLineItem
5
6
7 @receiver(post_save, sender=OrderLineItem)
8 def update_on_save(sender, instance, created, **kwargs):
9     """
10     Update order total when lineitem is updated/created
11     """
12     instance.order.update_total()
13
14
15 @receiver(post_delete, sender=OrderLineItem)
16 def update_on_delete(sender, instance, **kwargs):
17     """
18     Update order total when a lineitem is deleted
19     """
20     instance.order.update_total()
21
```

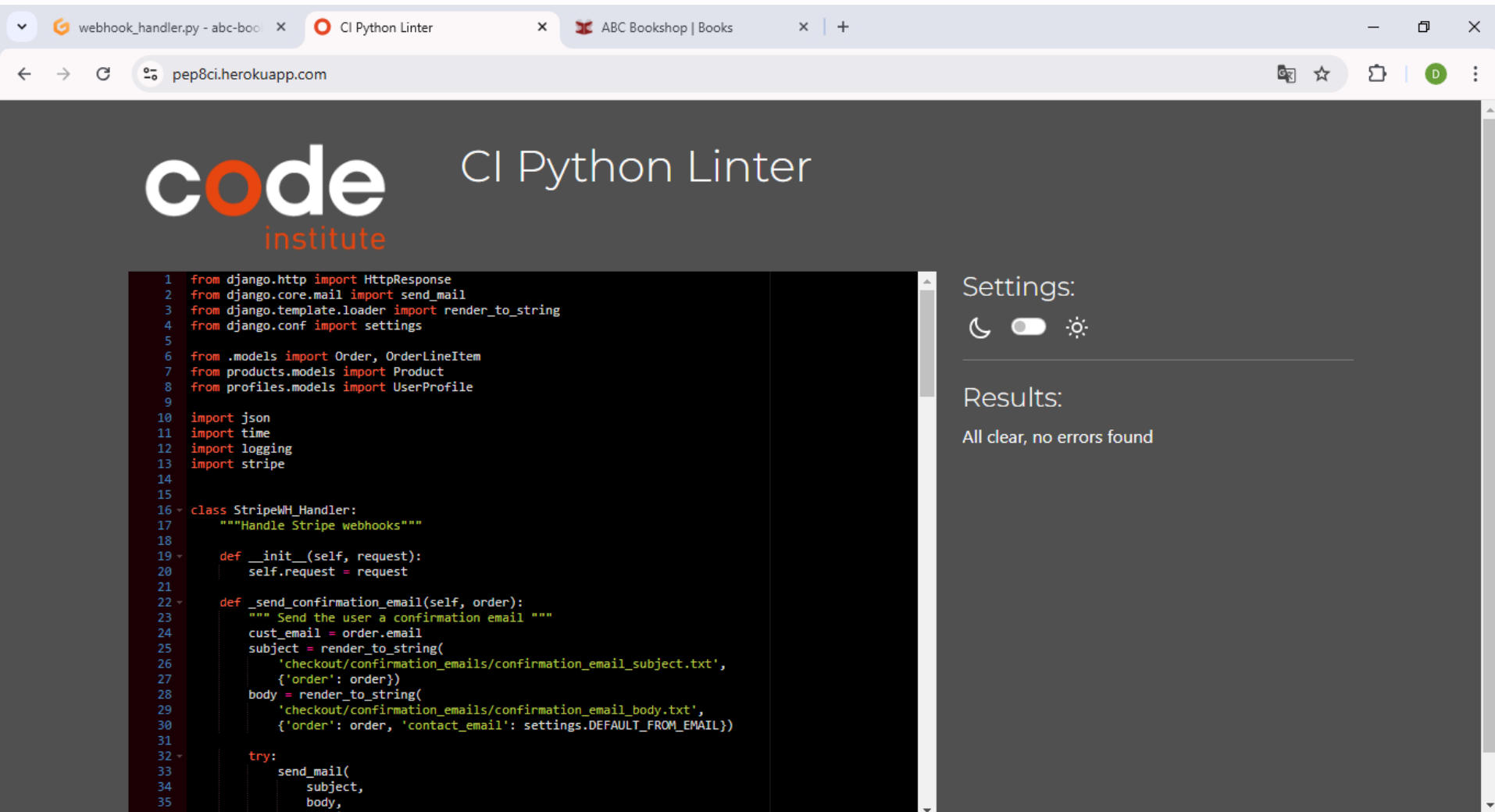
Settings:



Results:

All clear, no errors found

webhook_handler.py



The screenshot shows a web browser window with the URL `pep8ci.herokuapp.com`. The page title is "CI Python Linter" and the logo for "code institute" is visible. The main content area displays the Python code from `webhook_handler.py` in a dark-themed editor. The code includes imports for Django, JSON, time, logging, and Stripe, and defines a `StripeWH_Handler` class with methods for handling webhooks and sending confirmation emails. On the right side, there is a "Settings:" section with a toggle switch and a "Results:" section that states "All clear, no errors found".

```
1 from django.http import HttpResponse
2 from django.core.mail import send_mail
3 from django.template.loader import render_to_string
4 from django.conf import settings
5
6 from .models import Order, OrderLineItem
7 from products.models import Product
8 from profiles.models import UserProfile
9
10 import json
11 import time
12 import logging
13 import stripe
14
15
16 class StripeWH_Handler:
17     """Handle Stripe webhooks"""
18
19     def __init__(self, request):
20         self.request = request
21
22     def _send_confirmation_email(self, order):
23         """ Send the user a confirmation email """
24         cust_email = order.email
25         subject = render_to_string(
26             'checkout/confirmation_emails/confirmation_email_subject.txt',
27             {'order': order})
28         body = render_to_string(
29             'checkout/confirmation_emails/confirmation_email_body.txt',
30             {'order': order, 'contact_email': settings.DEFAULT_FROM_EMAIL})
31
32         try:
33             send_mail(
34                 subject,
35                 body,
```

Settings:

Results:

All clear, no errors found


webhooks.py

webhooks.py - abc-bookshop - x

CI Python Linter x

ABC Bookshop | Books x +



pep8ci.herokuapp.com



CI Python Linter

```
19 # Get the webhook data and verify its signature
20 payload = request.body
21 sig_header = request.META['HTTP_STRIPE_SIGNATURE']
22 event = None
23
24 try:
25     event = stripe.Webhook.construct_event(payload, sig_header, wh_secret)
26 except ValueError as e:
27     # Invalid payload
28     return HttpResponse(status=400)
29 except stripe.error.SignatureVerificationError as e:
30     # Invalid signature
31     return HttpResponse(status=400)
32 except Exception as e:
33     return HttpResponse(content=e, status=400)
34
35 # Set up a webhook handler
36 handler = StripeWH_Handler(request)
37
38 # Map webhook events to relevant handler functions
39 event_map = {
40     'payment_intent.succeeded': handler.handle_payment_intent_succeeded,
41     'payment_intent.payment_failed': handler.handle_payment_intent_payment_failed, # noqa
42 }
43
44 # Get the webhook type from Stripe
45 event_type = event['type']
46
47 # If there's a handler for it, get it from the event map
48 # Use the generic one by default
49 event_handler = event_map.get(event_type, handler.handle_event)
50
51 # Call the event handler with the event
52 response = event_handler(event)
53 return response
54
```

Settings:

 ☒ 

Results:

All clear, no errors found