

admin.py

The screenshot shows a web browser with three tabs: 'admin.py - abc-bookshop - Git', 'CI Python Linter', and 'ABC Bookshop | Books'. The address bar shows 'pep8ci.herokuapp.com'. The main content area has a dark background with the 'code institute' logo and the text 'CI Python Linter'. On the left, a code editor displays the contents of 'admin.py' with line numbers 4 through 38. The code defines two Django Admin classes, 'ProductAdmin' and 'CategoryAdmin', and registers them with the Django Admin site. On the right, the 'Settings' section shows a dark mode toggle which is currently turned on. The 'Results' section displays the message 'All clear, no errors found'.

```
4
5 class ProductAdmin(admin.ModelAdmin):
6     """ Product model in Admin """
7     list_display = (
8         'ean',
9         'title',
10        'author',
11        'category',
12        'by_age',
13        'price',
14        'sale_price',
15        'rating',
16        'number_of_pages',
17        'new_arrival',
18        'is_sale',
19        'image',
20    )
21
22    ordering = ('ean',)
23
24
25 class CategoryAdmin(admin.ModelAdmin):
26     list_display = (
27         'friendly_name',
28         'name',
29     )
30
31
32 admin.site.register(Product, ProductAdmin)
33 admin.site.register(Category, CategoryAdmin)
34
35 """ Custom header naming """
36 admin.site.site_title = "ABC Bookshop"
37 admin.site.index_title = "Administration panel"
38 admin.site.site_header = "ABC Bookshop administrator panel"
39
```

Settings:

Results:

All clear, no errors found


apps.py

apps.py - abc-bookshop - Gitp

CI Python Linter

ABC Bookshop | Books



← → ↻ 🔍 pep8ci.herokuapp.com



CI Python Linter

```
1 from django.apps import AppConfig
2
3
4 class ProductsConfig(AppConfig):
5     default_auto_field = 'django.db.models.BigAutoField'
6     name = 'products'
7
```

Settings:

 ☒ 

Results:

All clear, no errors found


urls.py

urls.py - abc-bookshop - Gitpod

CI Python Linter



ABC Bookshop | Books

← → ↻ pep8ci.herokuapp.com

 CI Python Linter

```
1 from django.urls import path
2 from . import views
3
4 urlpatterns = [
5     path('', views.all_products, name='products'),
6     path('<int:product_id>/', views.product_detail, name='product_detail'),
7     path('add/', views.add_product, name='add_product'),
8     path('edit/<int:product_id>/', views.edit_product, name='edit_product'),
9     path(
10         'delete/<int:product_id>/',
11         views.delete_product,
12         name='delete_product'),
13     path('add_review/<int:product_id>/', views.add_review, name='add_review'),
14 ]
15
```

Settings:

 ☒ 

Results:

All clear, no errors found


views.py

views.py - abc-bookshop - Gitp

CI Python Linter

ABC Bookshop | Books




pep8ci.herokuapp.com



CI Python Linter

```
276     else:
277         messages.error(
278             request, "Failed to update book. Please check the form inputs."
279         )
280     else:
281         form = ProductForm(instance=product)
282         messages.info(request, f"You are editing {product.title}")
283
284     template = "products/edit_product.html"
285     context = {
286         "form": form,
287         "product": product,
288         "on_page": True,
289     }
290
291     return render(request, template, context)
292
293
294 @login_required
295 def delete_product(request, product_id):
296     """Delete a book from the store offer"""
297
298     if not request.user.is_superuser:
299         messages.error(request, "Sorry, only store owner can delete books.")
300         return redirect(reverse("products"))
301
302     product = get_object_or_404(Product, pk=product_id)
303     product.delete()
304     messages.success(request, "Book was deleted.")
305
306     context = {
307         "on_page": True,
308     }
309
310     return redirect(reverse("products"), context)
311
```

Settings:



Results:

All clear, no errors found


forms.py

forms.py - abc-bookshop - Gitp

CI Python Linter

ABC Bookshop | Books



pep8ci.herokuapp.com



CI Python Linter

```
1 from django import forms
2 from .widgets import CustomClearableFileInput
3 from .models import Product, Category
4
5
6 class ProductForm(forms.ModelForm):
7     """ Product form for managing books """
8     class Meta:
9         model = Product
10        fields = '__all__'
11
12        image = forms.ImageField(
13            label='Image', required=False, widget=CustomClearableFileInput
14        )
15
16    def __init__(self, *args, **kwargs):
17        super().__init__(*args, **kwargs)
18        categories = Category.objects.all()
19        friendly_names = [(c.id, c.get_friendly_name()) for c in categories]
20
21        self.fields['category'].choices = friendly_names
22        for field_name, field in self.fields.items():
23            field.widget.attrs['class'] = 'border-black rounded-0'
24
```

Settings:

☒ 

Results:

All clear, no errors found

models.py

models.py - abc-bookshop - Git x

CI Python Linter x

ABC Bookshop | Books x +

← → ↻ pep8ci.herokuapp.com

code

institute

CI Python Linter

```
52         max_digits=6, decimal_places=2, blank=True, null=True
53     )
54     is_sale = models.BooleanField()
55     sale_price = models.DecimalField(
56         max_digits=6, decimal_places=2, null=True, blank=True, validators=[
57             MinValueValidator(0.0, message=None)]
58     )
59     by_age = models.CharField(
60         max_length=40,
61         null=False,
62         blank=False,
63         choices=[
64             ("schoolchildren", "schoolchildren"),
65             ("students", "students"),
66             ("professionals", "professionals"),
67         ],
68     )
69     image = CloudinaryField('image', null=True, blank=True)
70
71     # To be able to user have final_price as model attribute
72     # without having to add a new model field
73     @property
74     def final_price(self):
75         """ If product is on sale, set it's sales price to final price """
76         products = Product.objects.all()
77         for product in products:
78             try:
79                 if self.is_sale and self.sale_price < self.price:
80                     return self.sale_price
81                 else:
82                     return self.price
83             except (TypeError, ValueError):
84                 return None
85
86     def __str__(self):
87         return self.title
```

Settings:

☐ ☒ ☐

Results:

All clear, no errors found


widgets.py

widgets.py - abc-bookshop - G

CI Python Linter

ABC Bookshop | Books



pep8ci.herokuapp.com



CI Python Linter

```
1 from django.forms.widgets import ClearableFileInput
2 from django.utils.translation import gettext_lazy as _
3
4
5 class CustomClearableFileInput(ClearableFileInput):
6     """Improve image field on the form"""
7
8     clear_checkbox_label = _("Remove")
9     initial_text = _("Current Image")
10    input_text = _("")
11    template_name = (
12        "products/custom_widget_templates/"
13        "custom_clearable_file_input.html")
14
```

Settings:

 ☒ 

Results:

All clear, no errors found