

# PEP8 VALIDATION

# admin.py



## CI Python Linter

```
1 from django.contrib import admin
2 from .models import Category, Wallet
3
4 # Register your models here.
5 admin.site.register(Category)
6 admin.site.register(Wallet)
7 |
```

Settings:



Results:

All clear, no errors found

# Foms.py



## CI Python Linter

```
1 from django import forms
2 from django.contrib.auth.forms import UserCreationForm
3 from django.contrib.auth.models import User
4 from django.core.exceptions import ValidationError
5 from django.utils import formats, timezone
6
7 from .models import Transaction, Wallet
8
9
10 - class RegisterForm(UserCreationForm):
11 |     """Form for registering a new user with a username and password."""
12 -     class Meta:
13 |         model = User
14 |         fields = ["username", "password1", "password2"]
15
16
17 - class TransactionForm(forms.ModelForm):
18 |     """Form for creating a new transaction."""
19 -     class Meta:
20 |         model = Transaction
21 |         # fields = ['wallet', 'category', 'amount', 'date', 'is_income']
22 |         exclude = ["user"]
23 |         widgets = {
24 |             "date": forms.DateInput(
25 |                 format="%d-%m-%Y",
26 |                 attrs={
27 |                     "type": "date",
28 |                     "max": formats.date_format(timezone.now(), "Y-m-d"),
29 |                 },
30 |             )
31 |         }
32
33 -     def __init__(self, *args, **kwargs):
34 |         user = kwargs.pop("user", None)
35 |         super().__init__(*args, **kwargs) # Initialize the form
36
```

Settings:



Results:

All clear, no errors found

# Middleware.py



CI Python Linter

```
1 from django.http import Http404
2
3
4 class BlockAuthURLsMiddleware:
5     def __init__(self, get_response):
6         self.get_response = get_response
7         self.blocked_paths = [
8             "/password_change/",
9             "/password_change/done/",
10            "/password_reset/",
11            "/password_reset/done/",
12            "/reset/<uidb64>/<token>/",
13            "/reset/done/",
14        ]
15
16    def __call__(self, request):
17        if request.path in self.blocked_paths:
18            raise Http404("This page is not available.")
19        return self.get_response(request)
20
```

Settings:



Results:

All clear, no errors found

# Models.py



## CI Python Linter

```
47
48     name = models.CharField(
49         max_length=100,
50         unique=True,
51         help_text="The unique name of the category.",
52     ) # Unique category name
53
54     def __str__(self):
55         return self.name # For readable display
56
57     def delete(self, *args, **kwargs):
58         raise Exception("Deleting categories is not allowed.")
59
60
61 class Transaction(models.Model):
62     wallet = models.ForeignKey(
63         Wallet, on_delete=models.CASCADE
64     ) # One-to-many link to Wallet
65     category = models.ForeignKey(
66         Category, on_delete=models.SET_NULL, null=True
67     ) # Many-to-one link to Category
68     amount = models.DecimalField(
69         max_digits=10,
70         decimal_places=2,
71         validators=[MinValueValidator(Decimal("0.00"))],
72     ) # Transaction amount
73     date = models.DateField() # Transaction date
74     is_income = models.BooleanField(default=False)
75     user = models.ForeignKey(User, on_delete=models.CASCADE)
76
77     def __str__(self):
78         category_name = (
79             self.category.name if self.category else "Uncategorized"
80         )
81         return f"{self.amount} - {category_name}"
82
```

Settings:



Results:

All clear, no errors found

# URLS.py



## CI Python Linter

```
26 # path('personal/wallets', views.wallets, name="wallets"),
27 path(
28     "personal/transactions/<int:pk>/update",
29     views.UpdateTransaction.as_view(),
30     name="update_transaction",
31 ),
32 path(
33     "personal/transactions/<int:pk>/delete",
34     views.DeleteTransaction.as_view(),
35     name="delete_transaction",
36 ),
37 # Wallet
38 path(
39     "personal/wallet/create",
40     views.WalletCreateView.as_view(),
41     name="create_wallet",
42 ),
43 path("personal/wallets", views.listWallet.as_view(), name="list_wallet"),
44 path(
45     "personal/wallets/<int:pk>/update",
46     views.UpdateWallet.as_view(),
47     name="update_wallet",
48 ),
49 path(
50     "personal/wallets/<int:pk>/delete",
51     views.DeleteWallet.as_view(),
52     name="delete_wallet",
53 ),
54 ]
55
56 # Serve static files during development
57 if settings.DEBUG:
58     urlpatterns += static(
59         settings.STATIC_URL, document_root=settings.STATIC_ROOT
60     )
61
```

### Settings:



### Results:

All clear, no errors found

# Views.py



## CI Python Linter

```
324 -         if old_instance_data["is_income"]:
325 -             old_wallet.balance -= old_instance_data["amount"]
326 -         else:
327 -             old_wallet.balance += old_instance_data["amount"]
328 -         old_wallet.save()
329 -
330 -         # Apply the new transaction effect
331 -         if instance.is_income:
332 -             wallet.balance += instance.amount
333 -         else:
334 -             wallet.balance -= instance.amount
335 -         wallet.save()
336 -
337 -
338 - @receiver(post_save, sender=User)
339 - def create_default_wallet(sender, instance, created, **kwargs):
340 -     """Create a default wallet on account creation"""
341 -     if created:
342 -         Wallet.objects.create(user=instance, name="Your Wallet", balance=0.00)
343 -
344 -
345 - @receiver(pre_delete, sender=Transaction)
346 - def adjust_wallet_on_delete(sender, instance, **kwargs):
347 -     """Adjust the wallet balance when a transaction is deleted."""
348 -     wallet = instance.wallet
349 -     if instance.is_income:
350 -         wallet.balance -= instance.amount
351 -     else:
352 -         wallet.balance += instance.amount
353 -     wallet.save()
354 -
```

Settings:



Results:

All clear, no errors found