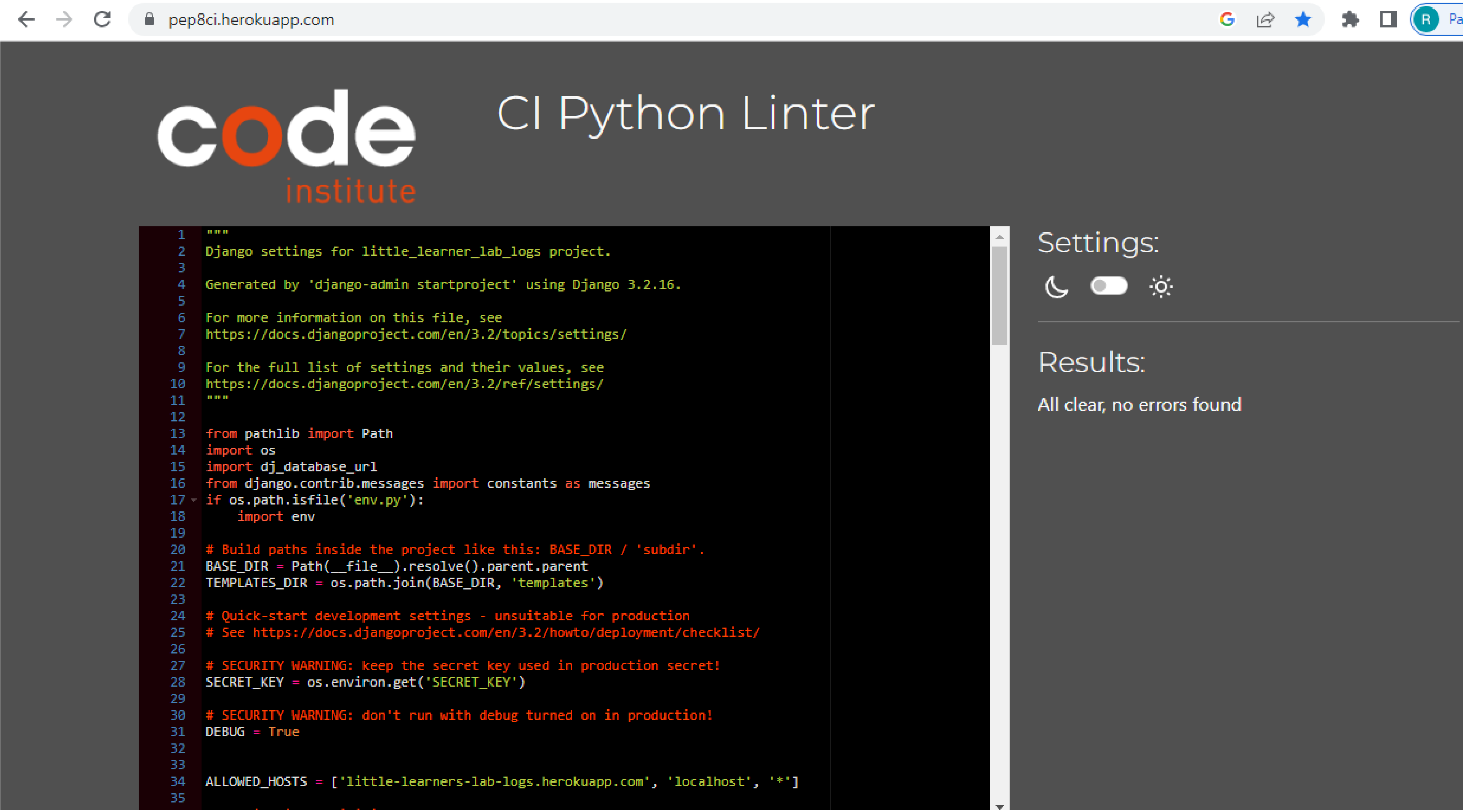
<little_learners_lab_logs> settings.py



CI Python Linter

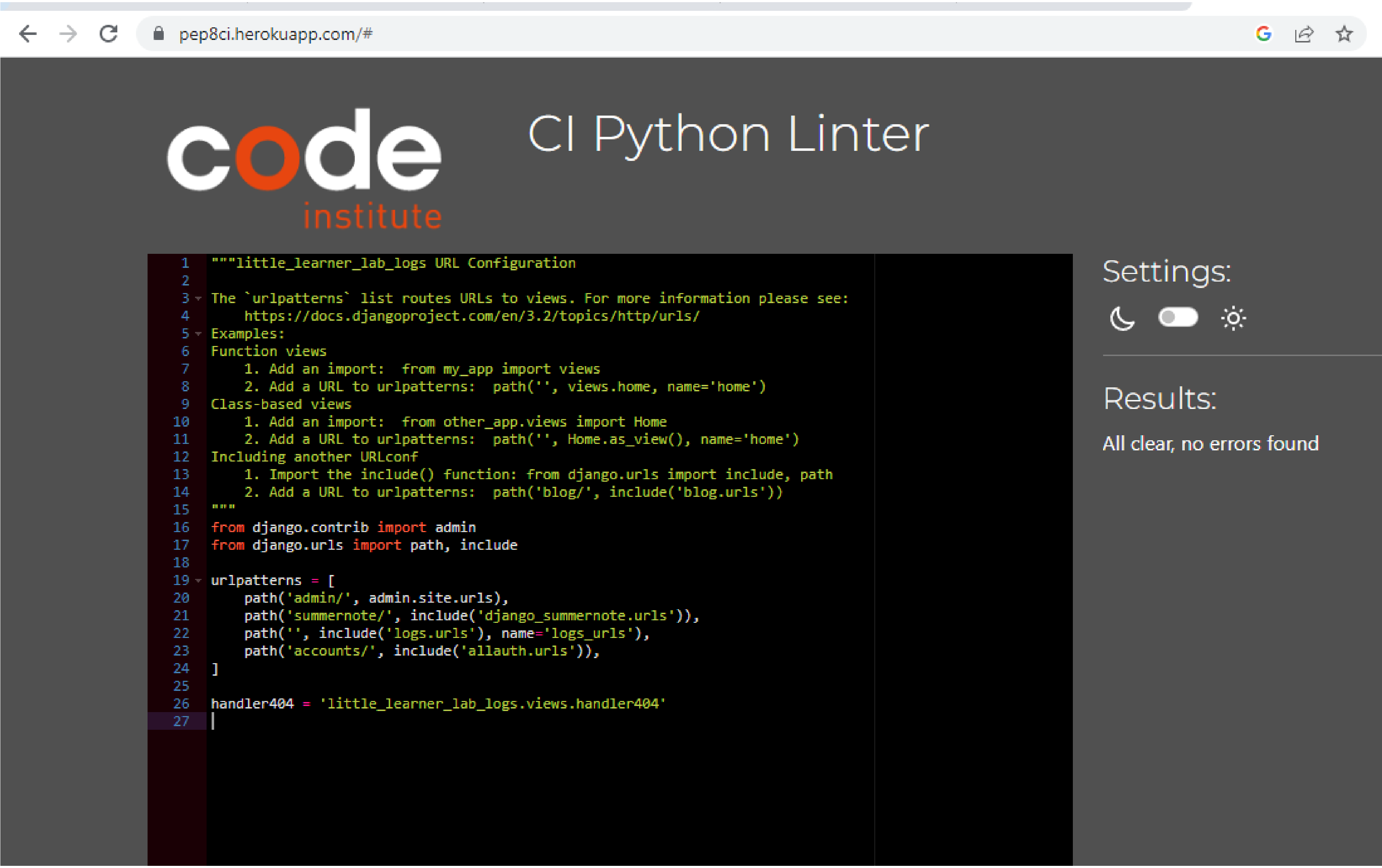
```
"""
Django settings for little_learner_lab_logs project.

Generated by 'django-admin startproject' using Django 3.2.16.

For more information on this file, see
https://docs.djangoproject.com/en/3.2/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/3.2/ref/settings/
"""

from pathlib import Path
import os
import dj_database_url
from django.contrib.messages import constants as messages
if os.path.isfile('env.py'):
    import env

# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent
TEMPLATES_DIR = os.path.join(BASE_DIR, 'templates')

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/3.2/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = os.environ.get('SECRET_KEY')

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True


ALLOWED_HOSTS = ['little-learners-lab-logs.herokuapp.com', 'localhost', '*']

```

Settings:

Results:

All clear, no errors found

<little_learners_lab_logs> urls.py

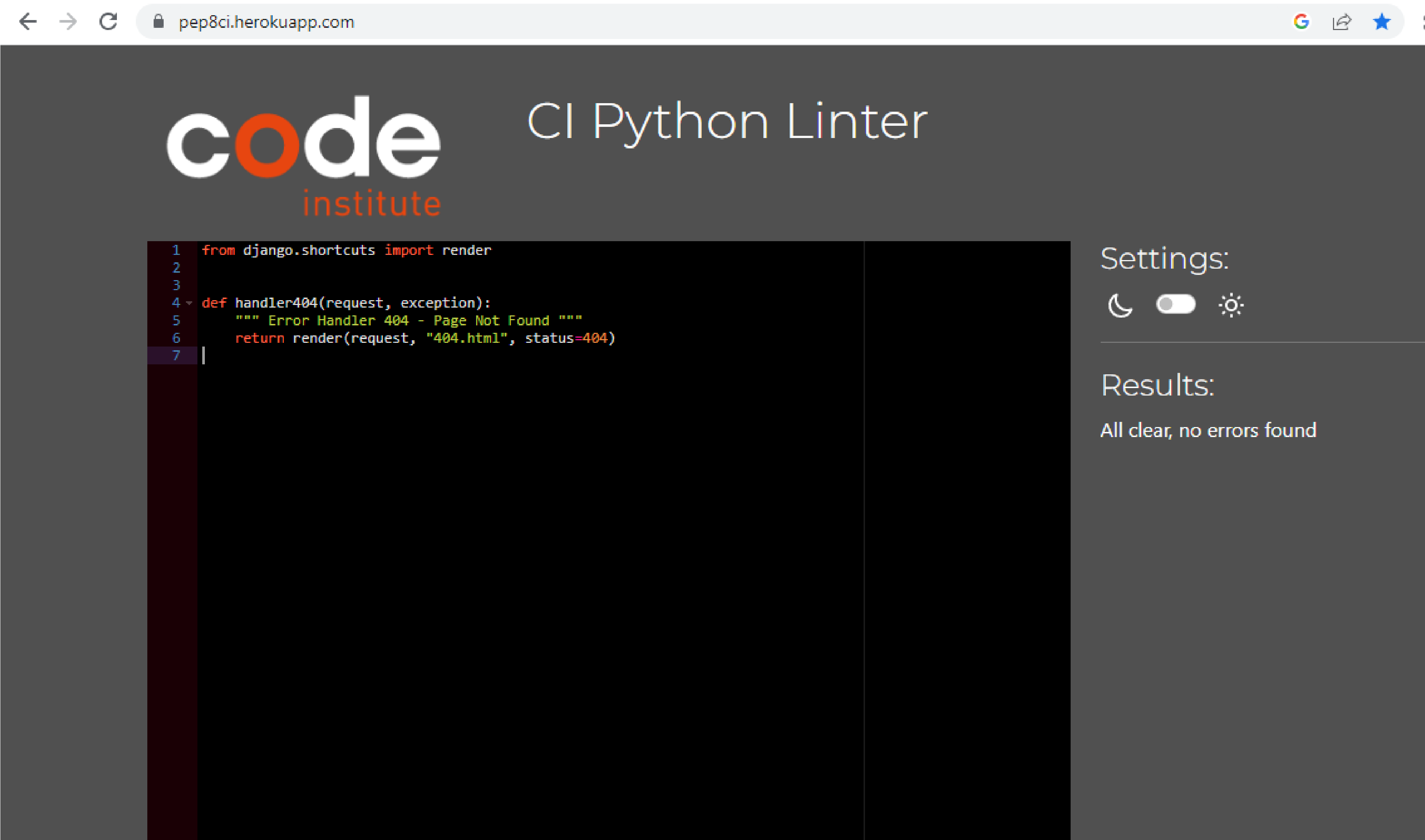


```
"""little_learner_lab_logs URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/3.2/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('summernote/', include('django_summernote.urls')),
    path('', include('logs.urls'), name='logs_urls'),
    path('accounts/', include('allauth.urls')),
]

handler404 = 'little_learner_lab_logs.views.handler404'
```

Results:

All clear, no errors found

<little_learners_lab_logs> views.py

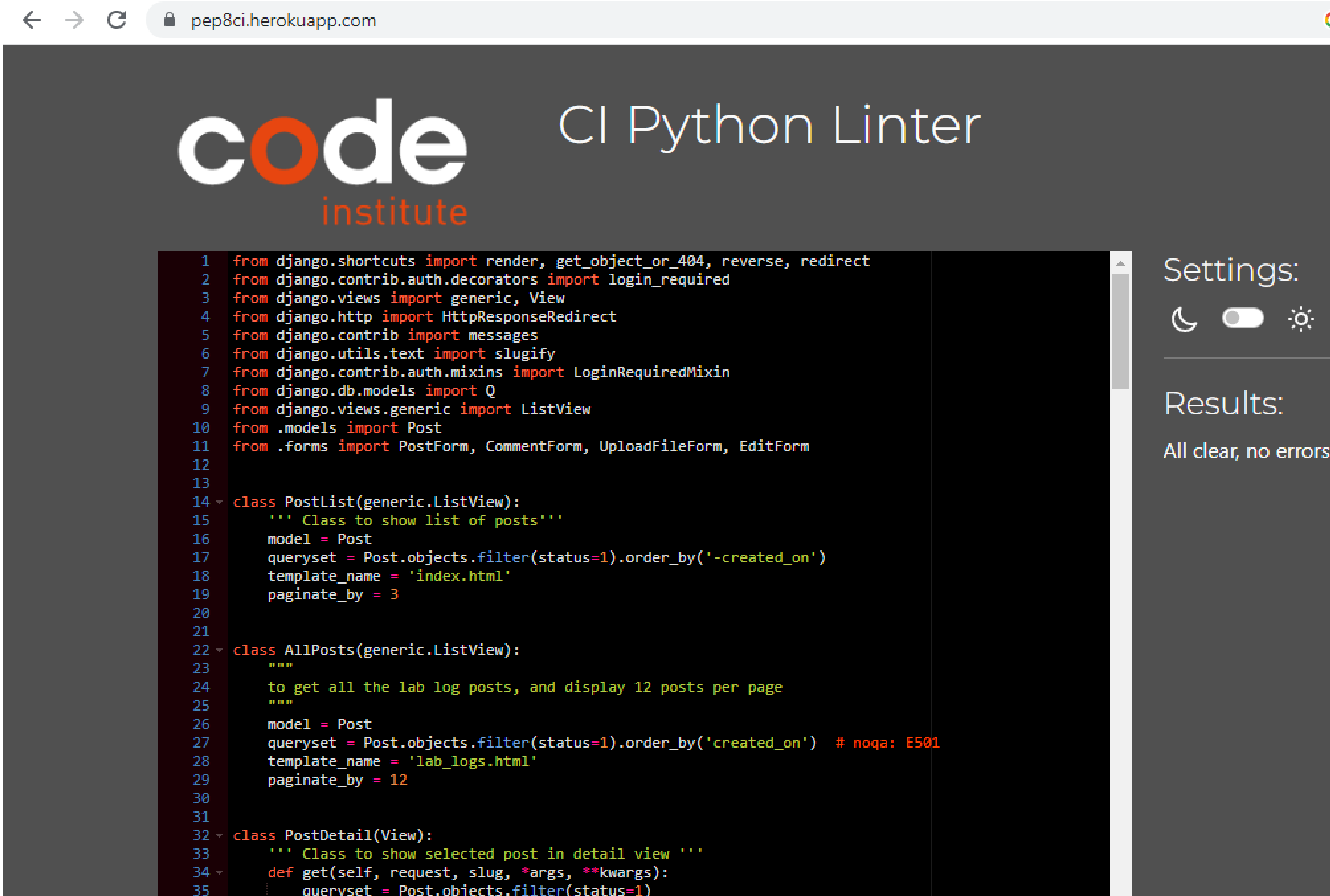


```
from django.shortcuts import render


def handler404(request, exception):
    """ Error Handler 404 - Page Not Found """
    return render(request, "404.html", status=404)
```

Results:

All clear, no errors found

views.py



CI Python Linter

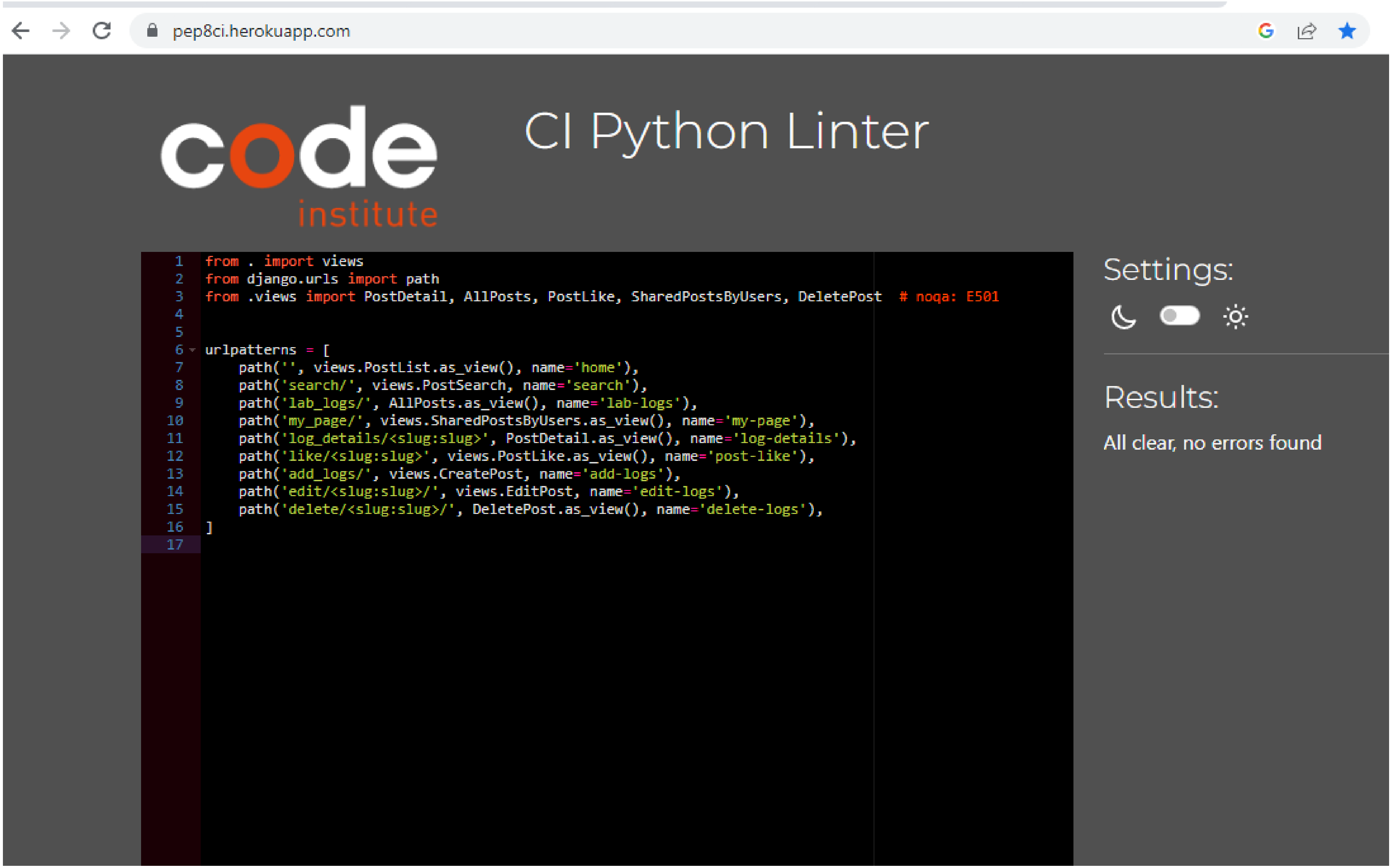
```
from django.shortcuts import render, get_object_or_404, reverse, redirect
from django.contrib.auth.decorators import login_required
from django.views import generic, View
from django.http import HttpResponseRedirect
from django.contrib import messages
from django.utils.text import slugify
from django.contrib.auth.mixins import LoginRequiredMixin
from django.db.models import Q
from django.views.generic import ListView
from .models import Post
from .forms import PostForm, CommentForm, UploadFileForm, EditForm


class PostList(generic.ListView):
    ''' Class to show list of posts'''
    model = Post
    queryset = Post.objects.filter(status=1).order_by('-created_on')
    template_name = 'index.html'
    paginate_by = 3


class AllPosts(generic.ListView):
    """
    to get all the lab log posts, and display 12 posts per page
    """
    model = Post
    queryset = Post.objects.filter(status=1).order_by('created_on')  # noqa: E501
    template_name = 'lab_logs.html'
    paginate_by = 12


class PostDetail(View):
    ''' Class to show selected post in detail view '''
    def get(self, request, slug, *args, **kwargs):
        queryset = Post.objects.filter(status=1)
```

Settings:

Results:

All clear, no errors

urls.py



CI Python Linter

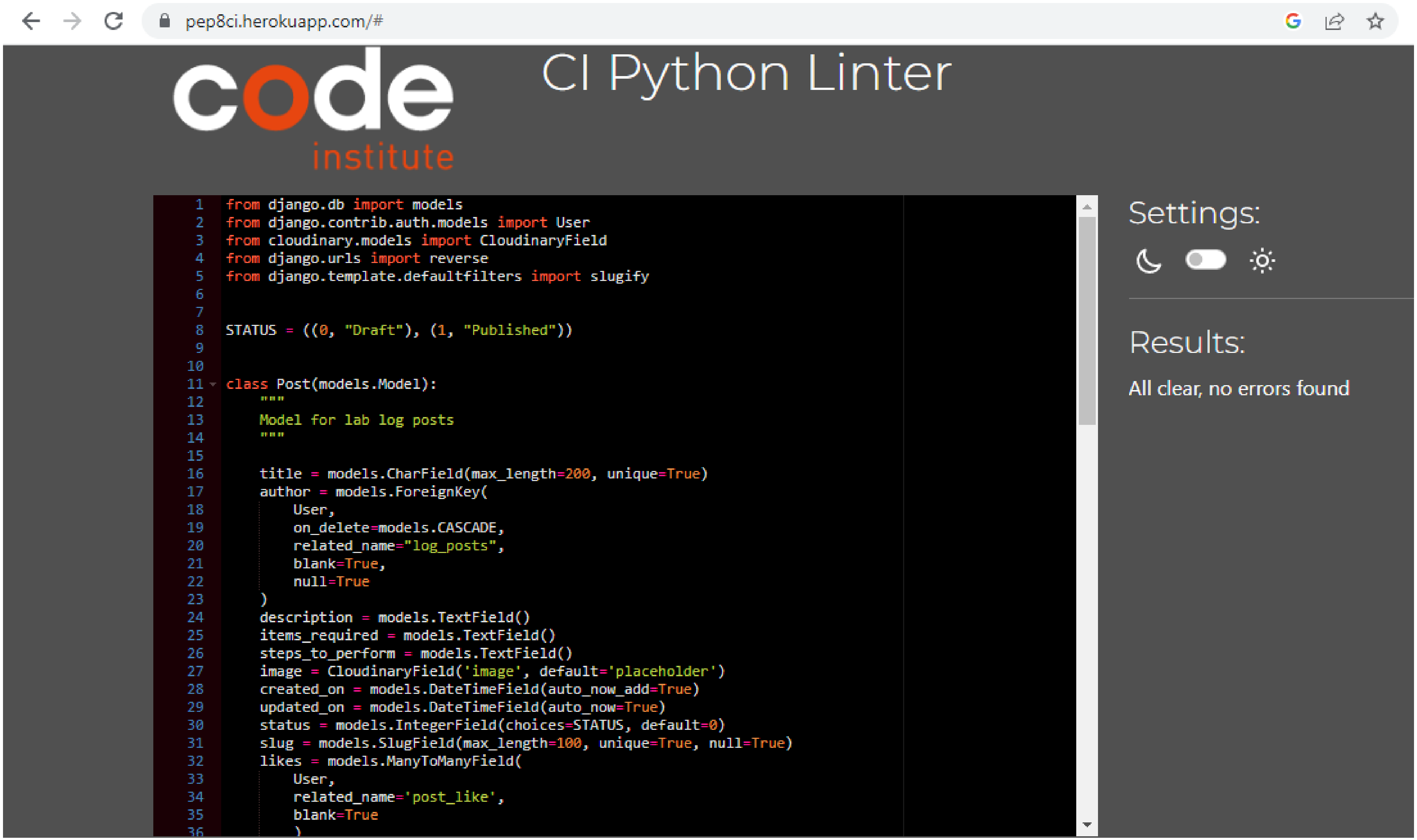
```
from . import views
from django.urls import path
from .views import PostDetail, AllPosts, PostLike, SharedPostsByUsers, DeletePost  # noqa: E501


urlpatterns = [
    path('', views.PostList.as_view(), name='home'),
    path('search/', views.PostSearch, name='search'),
    path('lab_logs/', AllPosts.as_view(), name='lab-logs'),
    path('my_page/', views.SharedPostsByUsers.as_view(), name='my-page'),
    path('log_details/<slug:slug>', PostDetail.as_view(), name='log-details'),
    path('like/<slug:slug>', views.PostLike.as_view(), name='post-like'),
    path('add_logs/', views.CreatePost, name='add-logs'),
    path('edit/<slug:slug>/', views.EditPost, name='edit-logs'),
    path('delete/<slug:slug>/', DeletePost.as_view(), name='delete-logs'),
]
```

Settings:

Results:

All clear, no errors found

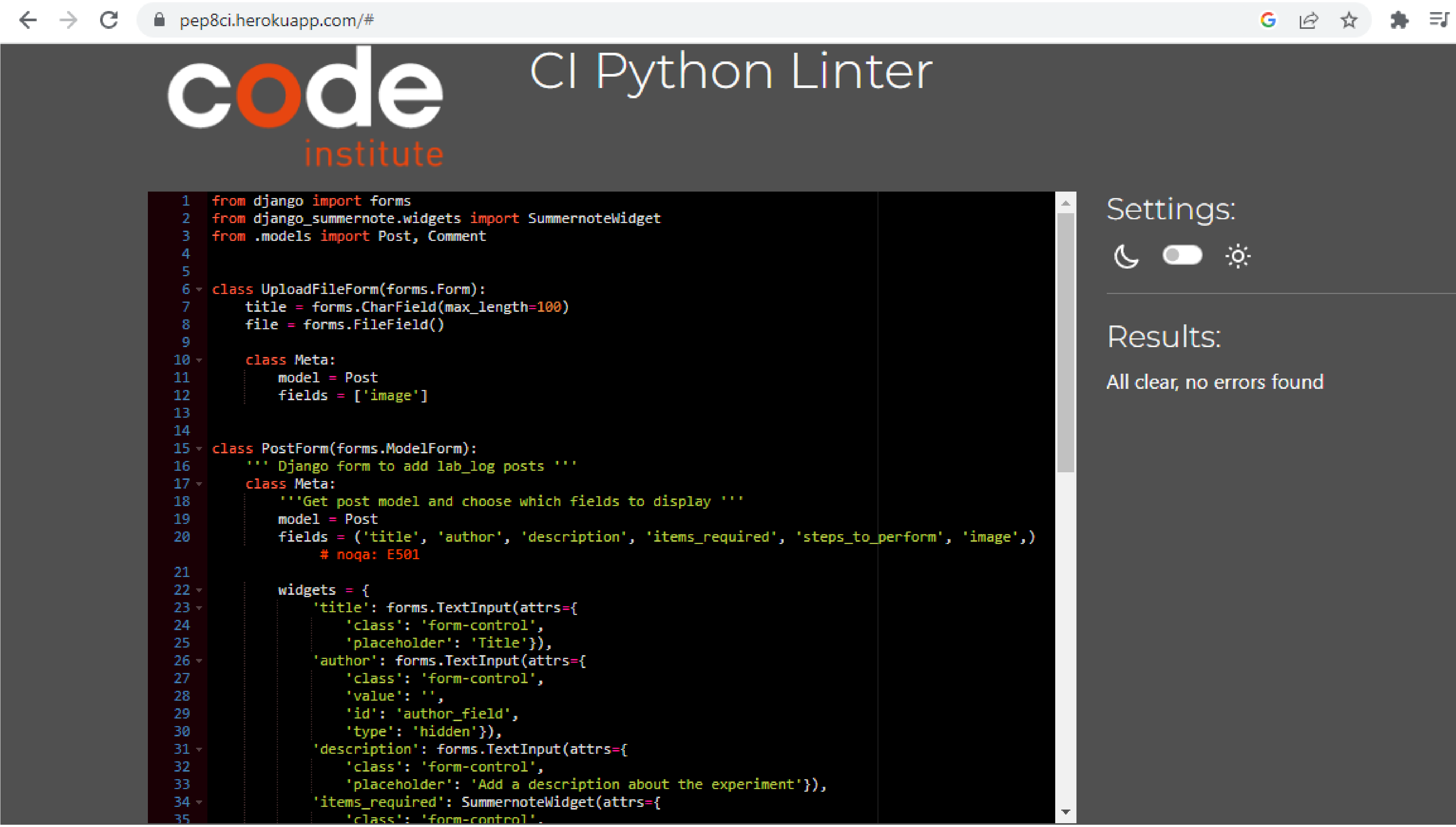
models.py



```
from django.db import models
from django.contrib.auth.models import User
from cloudinary.models import CloudinaryField
from django.urls import reverse
from django.template.defaultfilters import slugify


STATUS = ((0, "Draft"), (1, "Published"))


class Post(models.Model):
    """
    Model for lab log posts
    """

    title = models.CharField(max_length=200, unique=True)
    author = models.ForeignKey(
        User,
        on_delete=models.CASCADE,
        related_name="log_posts",
        blank=True,
        null=True
    )
    description = models.TextField()
    items_required = models.TextField()
    steps_to_perform = models.TextField()
    image = CloudinaryField('image', default='placeholder')
    created_on = models.DateTimeField(auto_now_add=True)
    updated_on = models.DateTimeField(auto_now=True)
    status = models.IntegerField(choices=STATUS, default=0)
    slug = models.SlugField(max_length=100, unique=True, null=True)
    likes = models.ManyToManyField(
        User,
        related_name='post_like',
        blank=True
        )
```

All clear, no errors found

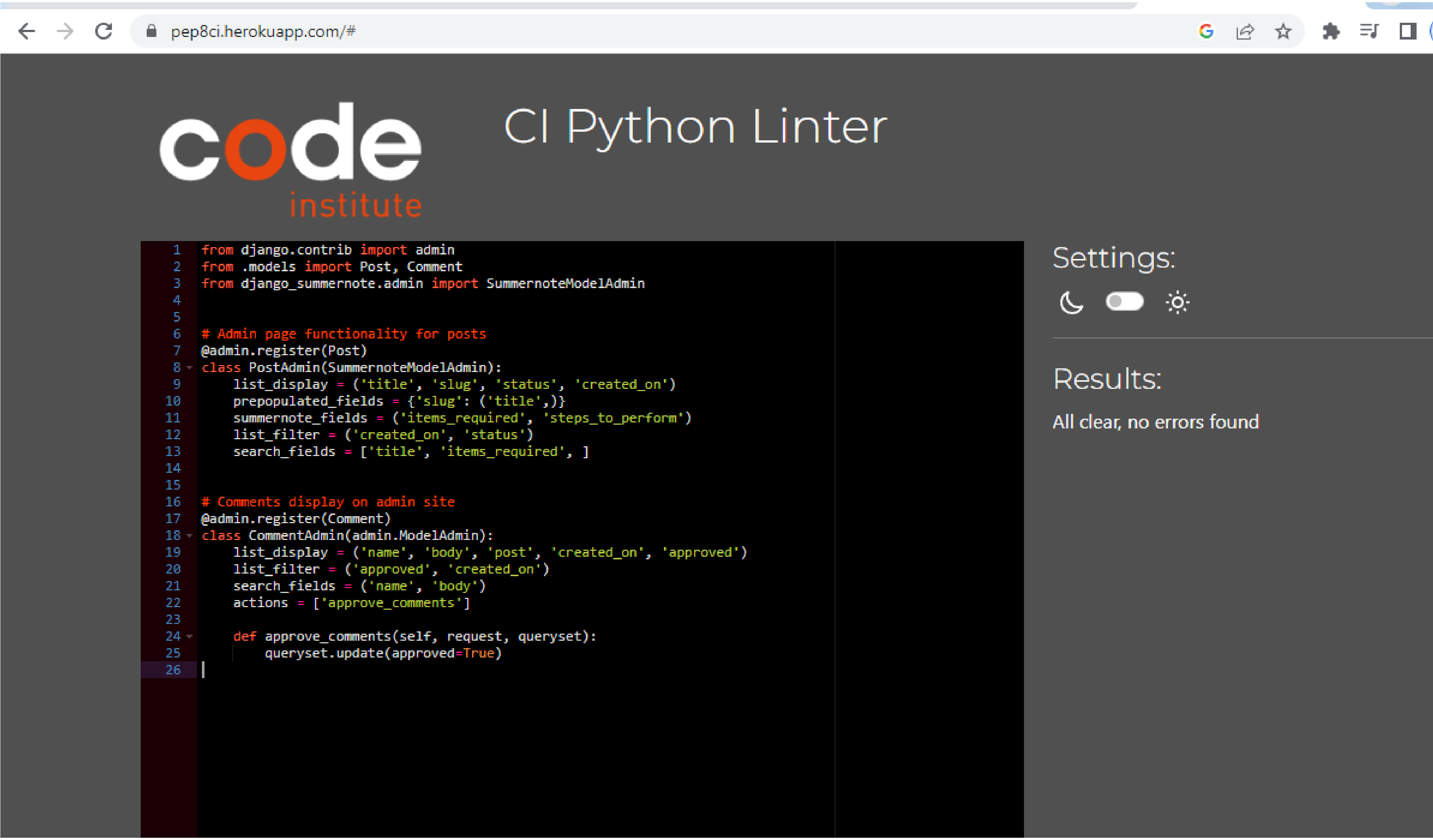
forms.py



```
from django import forms
from django_summernote.widgets import SummernoteWidget
from .models import Post, Comment


class UploadFileForm(forms.Form):
    title = forms.CharField(max_length=100)
    file = forms.FileField()

    class Meta:
        model = Post
        fields = ['image']


class PostForm(forms.ModelForm):
    ''' Django form to add lab_log posts '''
    class Meta:
        '''Get post model and choose which fields to display '''
        model = Post
        fields = ('title', 'author', 'description', 'items_required', 'steps_to_perform', 'image',) # noqa: E501

        widgets = {
            'title': forms.TextInput(attrs={
                'class': 'form-control',
                'placeholder': 'Title'}),
            'author': forms.TextInput(attrs={
                'class': 'form-control',
                'value': '',
                'id': 'author_field',
                'type': 'hidden'}),
            'description': forms.TextInput(attrs={
                'class': 'form-control',
                'placeholder': 'Add a description about the experiment'}),
            'items_required': SummernoteWidget(attrs={
                'class': 'form-control',
```

All clear, no errors found

admin.py

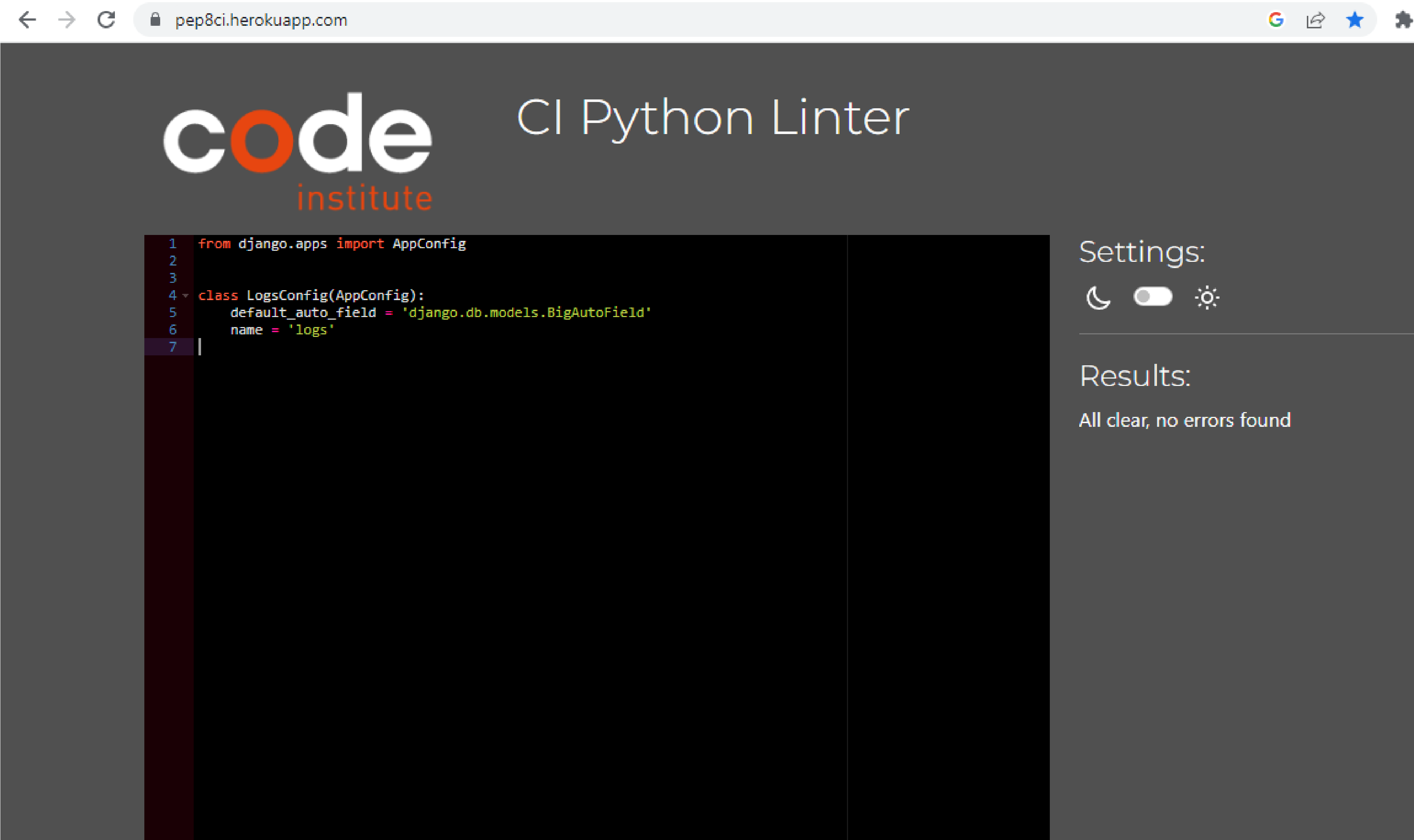


```python
from django.contrib import admin
from .models import Post, Comment
from django_summernote.admin import SummernoteModelAdmin


# Admin page functionality for posts
@admin.register(Post)
class PostAdmin(SummernoteModelAdmin):
    list_display = ('title', 'slug', 'status', 'created_on')
    prepopulated_fields = {'slug': ('title',)}
    summernote_fields = ('items_required', 'steps_to_perform')
    list_filter = ('created_on', 'status')
    search_fields = ['title', 'items_required', ]


# Comments display on admin site
@admin.register(Comment)
class CommentAdmin(admin.ModelAdmin):
    list_display = ('name', 'body', 'post', 'created_on', 'approved')
    list_filter = ('approved', 'created_on')
    search_fields = ('name', 'body')
    actions = ['approve_comments']

    def approve_comments(self, request, queryset):
        queryset.update(approved=True)
```

Results:

All clear, no errors found

apps.py



```
from django.apps import AppConfig


class LogsConfig(AppConfig):
    default_auto_field = 'django.db.models.BigAutoField'
    name = 'logs'
```

Results:

All clear, no errors found