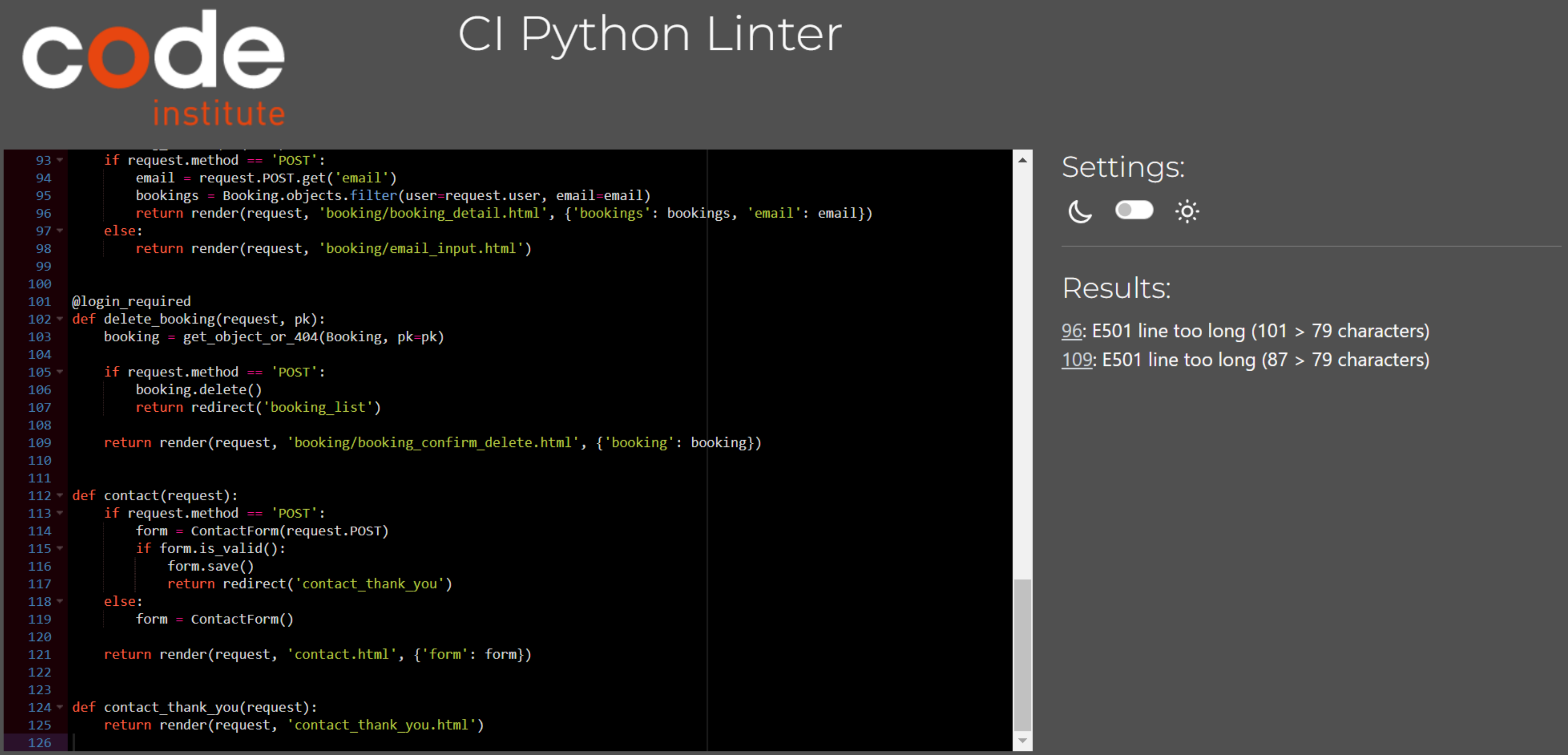


code institute

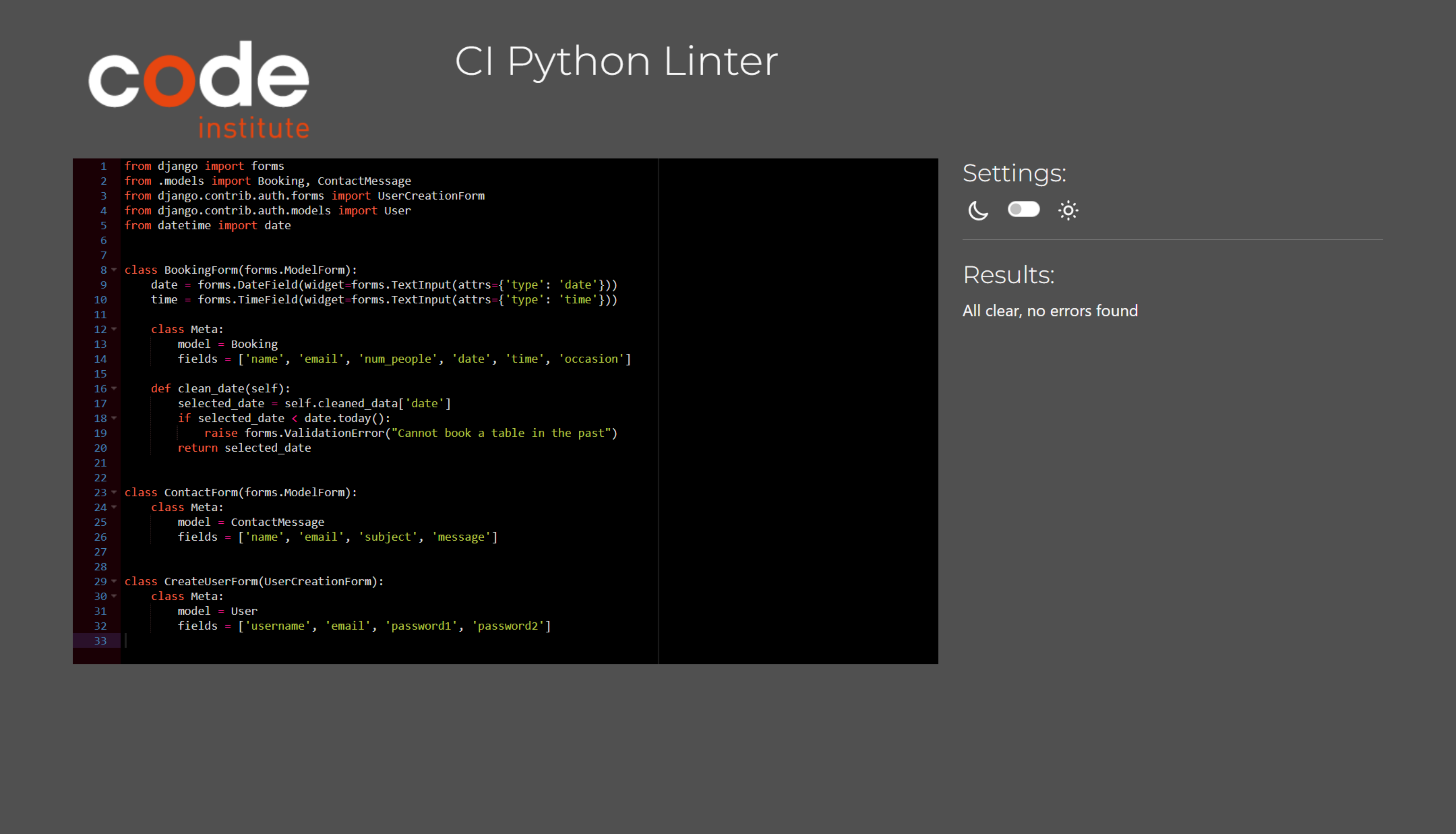
CI Python Linter

```
 93      if request.method == 'POST':
 94          email = request.POST.get('email')
 95          bookings = Booking.objects.filter(user=request.user, email=email)
 96          return render(request, 'booking/booking_detail.html', {'bookings': bookings, 'email': email})
 97      else:
 98          return render(request, 'booking/email_input.html')
 99
100
101  @login_required
102  def delete_booking(request, pk):
103      booking = get_object_or_404(Booking, pk=pk)
104
105      if request.method == 'POST':
106          booking.delete()
107          return redirect('booking_list')
108
109      return render(request, 'booking/booking_confirm_delete.html', {'booking': booking})
110
111
112  def contact(request):
113      if request.method == 'POST':
114          form = ContactForm(request.POST)
115          if form.is_valid():
116              form.save()
117              return redirect('contact_thank_you')
118      else:
119          form = ContactForm()
120
121      return render(request, 'contact.html', {'form': form})
122
123
124  def contact_thank_you(request):
125      return render(request, 'contact_thank_you.html')
126
```

Settings:

Results:

96: E501 line too long (101 > 79 characters)
109: E501 line too long (87 > 79 characters)



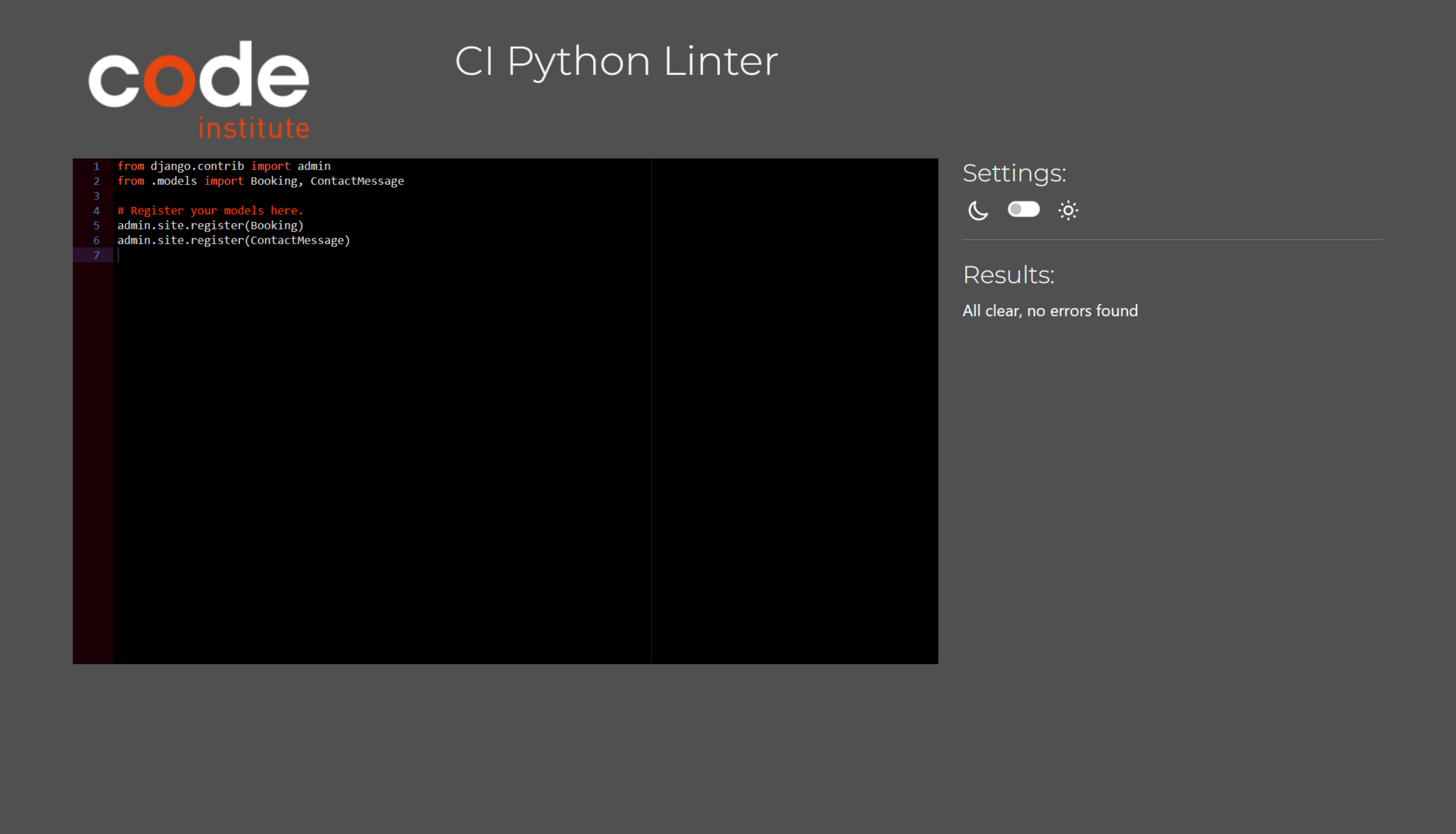
CI Python Linter

```python
from django import forms
from .models import Booking, ContactMessage
from django.contrib.auth.forms import UserCreationForm
from django.contrib.auth.models import User
from datetime import date


class BookingForm(forms.ModelForm):
    date = forms.DateField(widget=forms.TextInput(attrs={'type': 'date'}))
    time = forms.TimeField(widget=forms.TextInput(attrs={'type': 'time'}))

    class Meta:
        model = Booking
        fields = ['name', 'email', 'num_people', 'date', 'time', 'occasion']

    def clean_date(self):
        selected_date = self.cleaned_data['date']
        if selected_date < date.today():
            raise forms.ValidationError("Cannot book a table in the past")
        return selected_date


class ContactForm(forms.ModelForm):
    class Meta:
        model = ContactMessage
        fields = ['name', 'email', 'subject', 'message']


class CreateUserForm(UserCreationForm):
    class Meta:
        model = User
        fields = ['username', 'email', 'password1', 'password2']
```

Settings:

Results:

All clear, no errors found



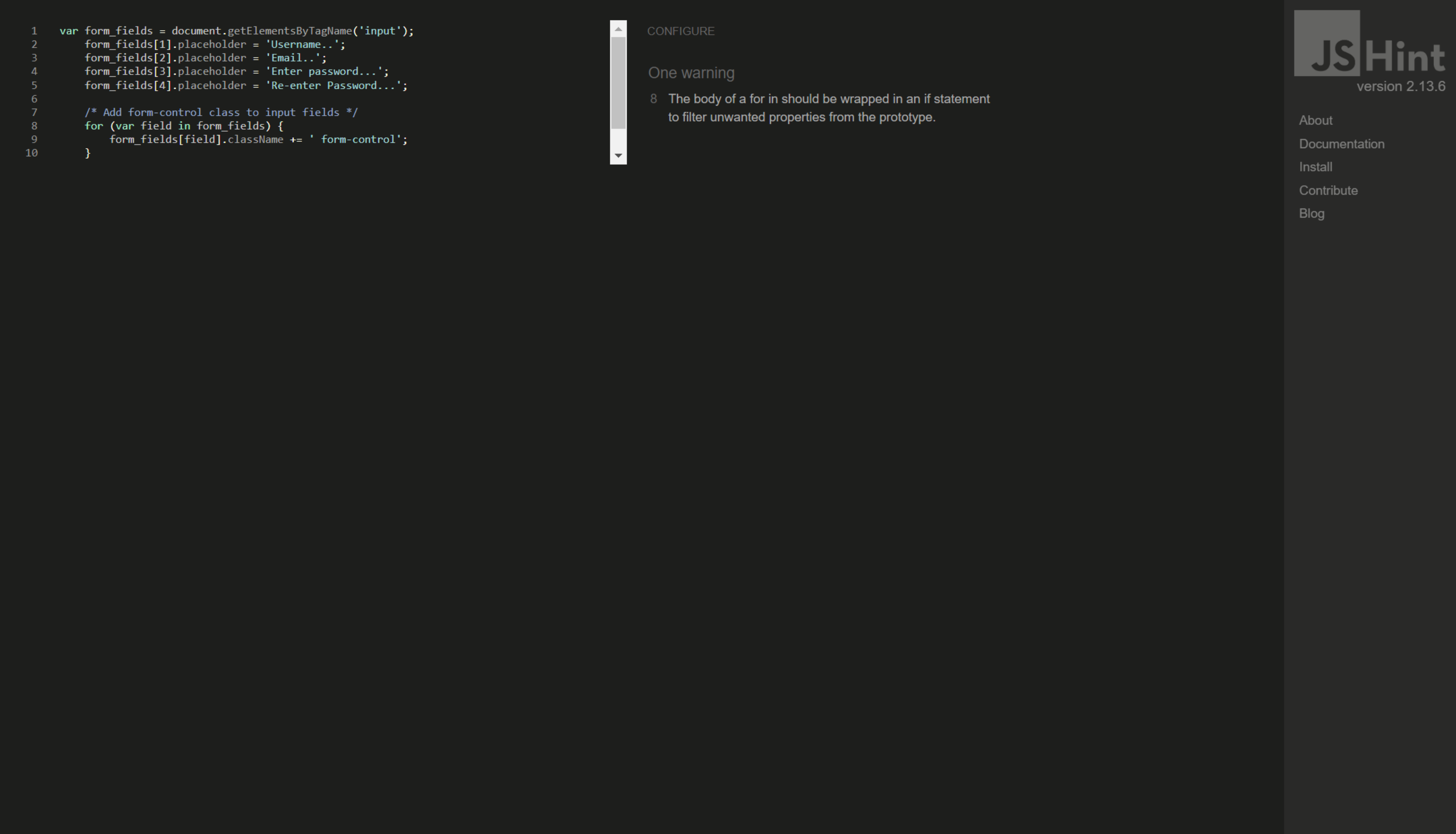
CI Python Linter

```
from django.contrib import admin
from .models import Booking, ContactMessage

# Register your models here.
admin.site.register(Booking)
admin.site.register(ContactMessage)
```

Settings:

Results:

All clear, no errors found

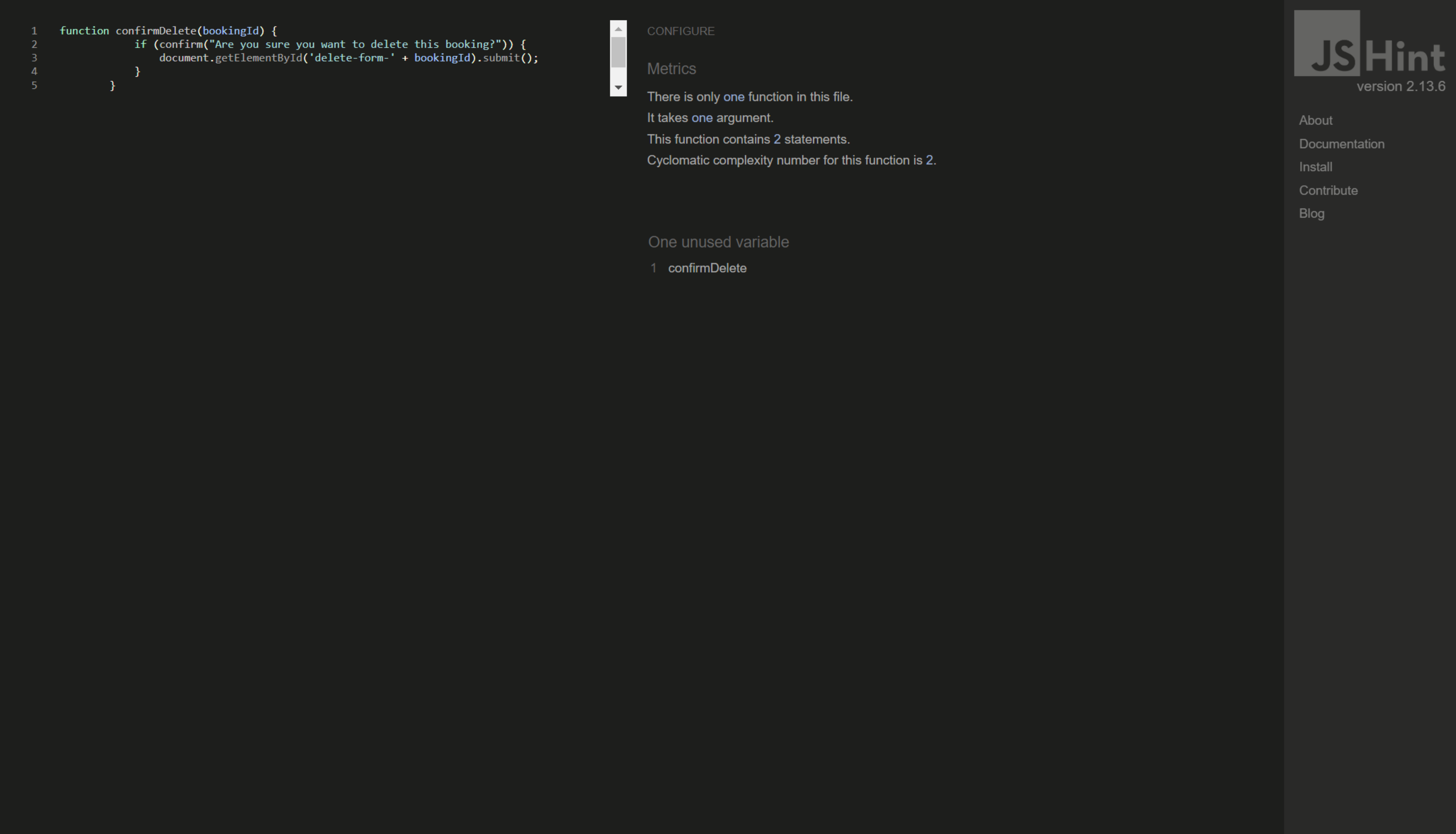


```
 1  var form_fields = document.getElementsByTagName('input');
 2      form_fields[1].placeholder = 'Username..';
 3      form_fields[2].placeholder = 'Email..';
 4      form_fields[3].placeholder = 'Enter password...';
 5      form_fields[4].placeholder = 'Re-enter Password...';
 6  
 7      /* Add form-control class to input fields */
 8      for (var field in form_fields) {
 9          form_fields[field].className += ' form-control';
10      }
```

CONFIGURE

One warning

8 The body of a for in should be wrapped in an if statement to filter unwanted properties from the prototype.



```javascript
function confirmDelete(bookingId) {
            if (confirm("Are you sure you want to delete this booking?")) {
                document.getElementById('delete-form-' + bookingId).submit();
            }
        }
```

CONFIGURE

Metrics

There is only one function in this file.

It takes one argument.

This function contains 2 statements.

Cyclomatic complexity number for this function is 2.

One unused variable

```
1  confirmDelete
```