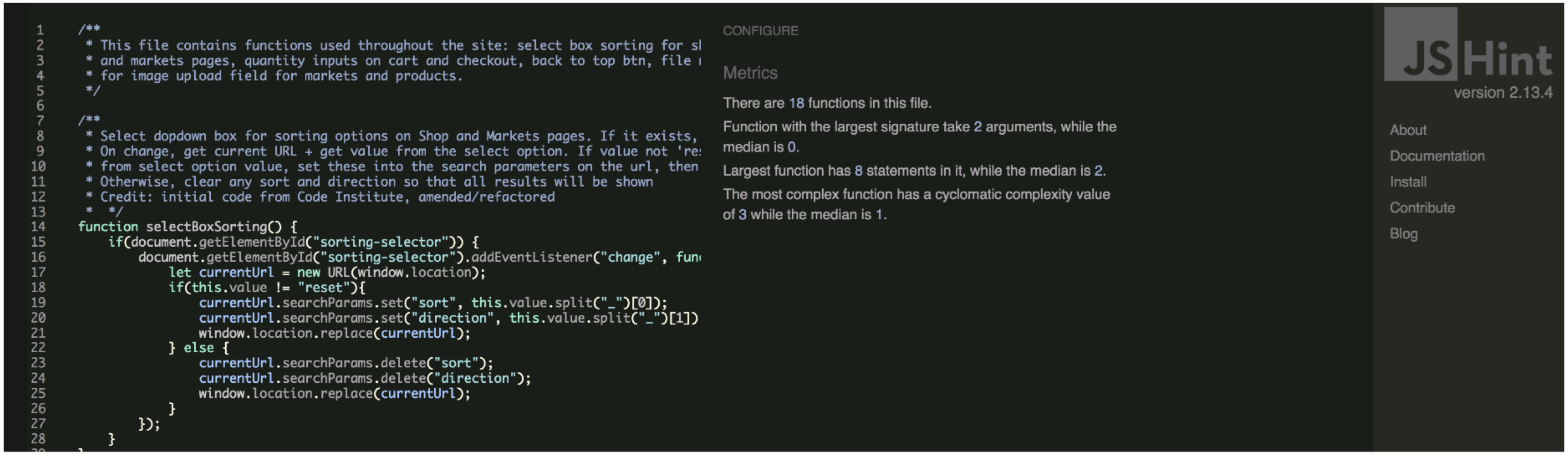
script.js



```
/**
 * This file contains functions used throughout the site: select box sorting for sl
 * and markets pages, quantity inputs on cart and checkout, back to top btn, file
 * for image upload field for markets and products.
 */

/**
 * Select dopdown box for sorting options on Shop and Markets pages. If it exists,
 * On change, get current URL + get value from the select option. If value not 're
 * from select option value, set these into the search parameters on the url, then
 * Otherwise, clear any sort and direction so that all results will be shown
 * Credit: initial code from Code Institute, amended/refactored
 *  */
function selectBoxSorting() {
    if(document.getElementById("sorting-selector")) {
        document.getElementById("sorting-selector").addEventListener("change", fun
            let currentUrl = new URL(window.location);
            if(this.value != "reset"){
                currentUrl.searchParams.set("sort", this.value.split("_")[0]);
                currentUrl.searchParams.set("direction", this.value.split("_")[1])
                window.location.replace(currentUrl);
            } else {
                currentUrl.searchParams.delete("sort");
                currentUrl.searchParams.delete("direction");
                window.location.replace(currentUrl);
            }
        });
    }
```

Metrics

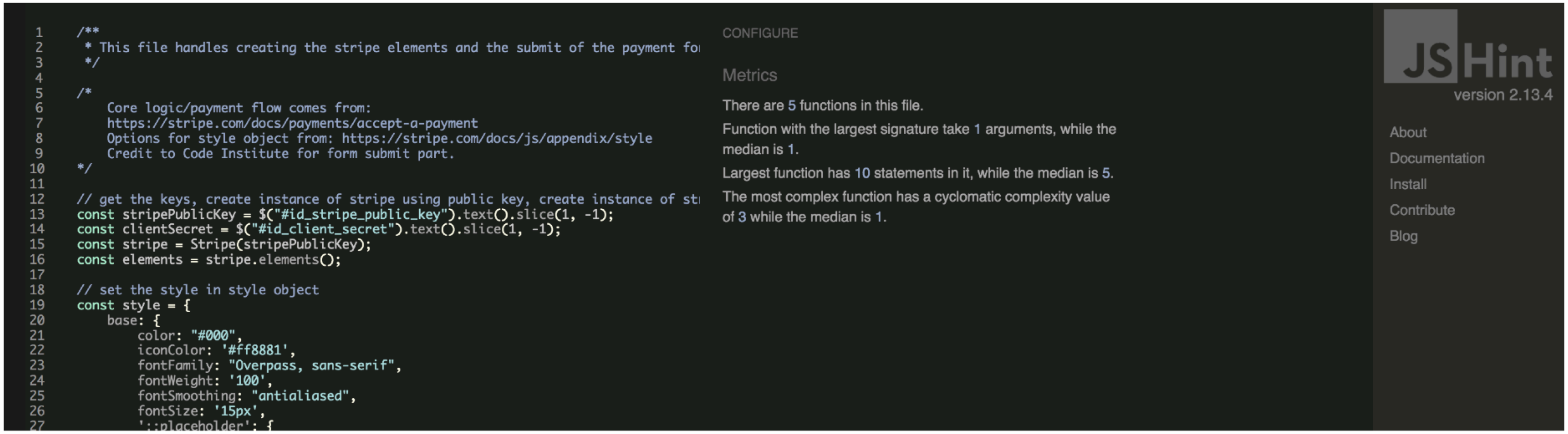
There are 18 functions in this file.

Function with the largest signature take 2 arguments, while the median is 0.

Largest function has 8 statements in it, while the median is 2.

The most complex function has a cyclomatic complexity value of 3 while the median is 1.

stripe_elements.js (checkout app)



```
/**
 * This file handles creating the stripe elements and the submit of the payment fo
 */

/*
    Core logic/payment flow comes from:
    https://stripe.com/docs/payments/accept-a-payment
    Options for style object from: https://stripe.com/docs/js/appendix/style
    Credit to Code Institute for form submit part.
*/

// get the keys, create instance of stripe using public key, create instance of st
const stripePublicKey = $("#id_stripe_public_key").text().slice(1, -1);
const clientSecret = $("#id_client_secret").text().slice(1, -1);
const stripe = Stripe(stripePublicKey);
const elements = stripe.elements();

// set the style in style object
const style = {
    base: {
        color: "#000",
        iconColor: '#ff8881',
        fontFamily: "Overpass, sans-serif",
        fontWeight: '100',
        fontSmoothing: "antialiased",
        fontSize: '15px',
        '::placeholder': {
```

Metrics

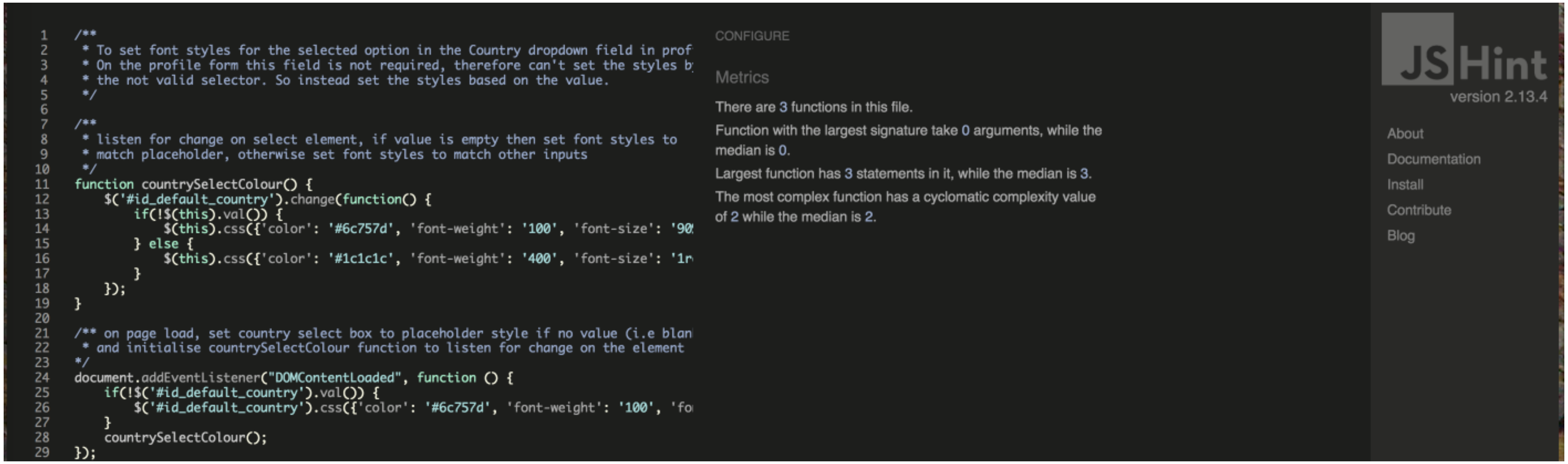
There are 5 functions in this file.

Function with the largest signature take 1 arguments, while the median is 1.

Largest function has 10 statements in it, while the median is 5.

The most complex function has a cyclomatic complexity value of 3 while the median is 1.

countryfield.js (profiles app)



```
/**
 * To set font styles for the selected option in the Country dropdown field in prof
 * On the profile form this field is not required, therefore can't set the styles b
 * the not valid selector. So instead set the styles based on the value.
 */

/**
 * listen for change on select element, if value is empty then set font styles to
 * match placeholder, otherwise set font styles to match other inputs
 */
function countrySelectColour() {
    $('#id_default_country').change(function() {
        if(!$(this).val()) {
            $(this).css({'color': '#6c757d', 'font-weight': '100', 'font-size': '90
        } else {
            $(this).css({'color': '#1c1c1c', 'font-weight': '400', 'font-size': '1r
        }
    });
}

/** on page load, set country select box to placeholder style if no value (i.e blan
 * and initialise countrySelectColour function to listen for change on the element
*/
document.addEventListener("DOMContentLoaded", function () {
    if(!$('#id_default_country').val()) {
        $('#id_default_country').css({'color': '#6c757d', 'font-weight': '100', 'fo
    }
    countrySelectColour();
});
```

Metrics

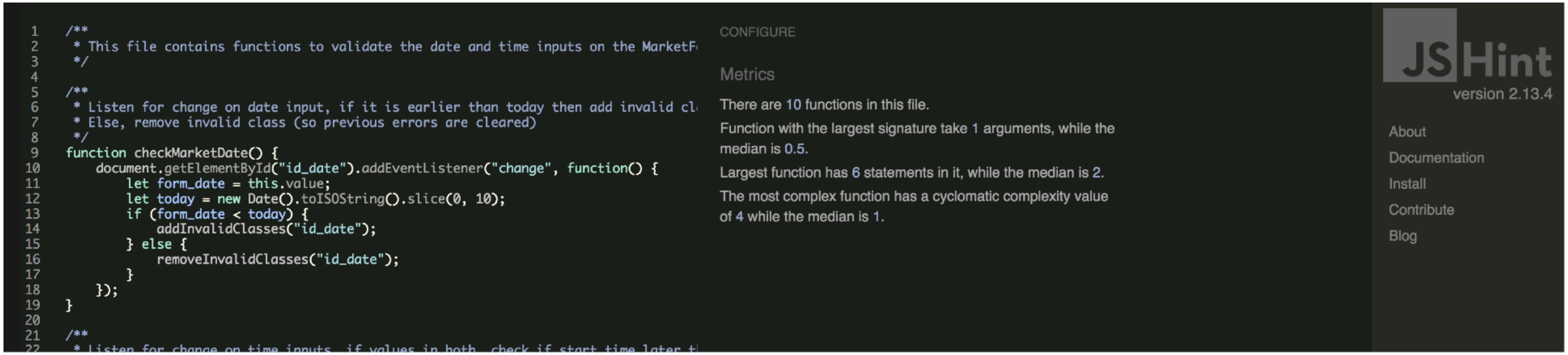
There are 3 functions in this file.

Function with the largest signature take 0 arguments, while the median is 0.

Largest function has 3 statements in it, while the median is 3.

The most complex function has a cyclomatic complexity value of 2 while the median is 2.

form_validation.js (markets app)



```
/**
 * This file contains functions to validate the date and time inputs on the MarketF
 */

/**
 * Listen for change on date input, if it is earlier than today then add invalid cl
 * Else, remove invalid class (so previous errors are cleared)
 */
function checkMarketDate() {
    document.getElementById("id_date").addEventListener("change", function() {
        let form_date = this.value;
        let today = new Date().toISOString().slice(0, 10);
        if (form_date < today) {
            addInvalidClasses("id_date");
        } else {
            removeInvalidClasses("id_date");
        }
    });
}

/**
 * Listen for change on time inputs, if values in both, check if start time later t
```

Metrics

There are 10 functions in this file.

Function with the largest signature take 1 arguments, while the median is 0.5.

Largest function has 6 statements in it, while the median is 2.

The most complex function has a cyclomatic complexity value of 4 while the median is 1.