**WDL**

**EXPERIMENT NO 3**

**ADDING FORM VALIDATION**

**USING**

**JAVASCRIPT AND HTML5**

**SUBMITTED BY**

**NOEL JOE  
ROLL NO 38**

**BATCH B**

**Aim:-**To design the forms and develop client side validation using Client side Validation using

1. JavaScript and HTML5

**Objective:**-To provide students an overview of the concepts of client side programming JavaScript and Jquery

**Theory:-**

**Client side Validation**

When we go to any popular site or a registration form and you will notice that they provide feedback when you don’t enter data in the format they are expecting.You get messages such as

This field is required (You cant leave this field blank).

Please enter your mobile number in the format xxx-xxxxx

Please enter a valid email address.

Your password needs to be between 8 and 30 characters long and contain one uppercase letter one symbol and a number

This is called form validation.When you enter data the browser checks that data to see if its in the correct format and within the constraints set by the application.Validation done in the browser is called client-side validation,while if its done by the web server then its called server-side validation.

If the information is correctly formated then the application allows the data to be submitted to the server and usually saved in the database;if the information isnt correctly formatted,it gives the user an error message what needs to be corrected,and lets them try again.

WHY IS VALIDATION DONE???

We as the website developers want to get the right data in the right format

We want to protect our users data

We want to protect ourselves from malicious users

There are two types of client-side validation

Built in form validation uses HTML 5 form validation features.This validation generally dosent require much JavaScript.Built in form validation has much performance than Javascript but it is not as customizable.

Javascript validation is written in Javascript.It is customizable,but you need to create it all or use a library

Using built in form validation

HTML5 built in validation is used to validate almost all the user data without relying on JavaScript.This is done using inbuilt form validation functions like

Required :-Specifies whether a form field needs to be filled in before the form can be submitted/

Min length and max length:Specifies the minimum and maximum length of textual data(string)

Min and max:Specifies the minimum the maximum length of textual data

Type:Specifes whether the data needs to be a number an email address or some other kind of specific preset type.

Pattern:Specifies a regular expression that defines a pattern the entered data needs to follow.

You can also apply CSS attributes if in a certain field is corrected correctly or incorrectly.

(Also pseudo classes like (out of range can be applied depending on the error).

**VALIDATING** FORMS USING **JAVASCRIPT**

You must use Javascript if you want to take control over the look and feel of native error messages or to deal with legacy browsers that do not support HTML’s built in form validation.

The Constraint Validation API

Most browsers support the Constraint Validation API,which consists of a set of methods and properties available on the following form element DOM interfaces:

HTMLButtonElement (represents a <button> element)

HTMLFieldSetElement(represents a <fieldset>element)

HTMLInputElement(represents an <input> element)

HTMLOutputElement(represents an <output> element)

HTMLSelectElement(represents a <select> element)

HTMLTextAreaElement(represents a <textarea > element)

Contraint Validation API makes use of the following properties available on the above elements.

validationMessage:Returns a localized message describing the validation constraints that the control does not satisfy if any.If the control is not a candidate for constraint validation,this will return an empty string.

Validity:Returns a validityState object that conatins several properties describing the validity state of an element.The properties if the validity state are given below.

patternMismatch:Returns true if the vlaue does not match the required pattern and false if it does not match.

Rangeoverflow:Returns true if the value is longer than the maximum string or specified by the max sttribute,or false if its shorter than or equal to the maximum.If true it matches the required CSS class selectors.

Rangeunderflow:Returns true if the value is shorter than the required string.If true it matches the invalid and outofRange attribute.

Typemismatch:Returns true if the value is not in the required syntax.(When type is email or url), If its true it matches the :valid CSS pseudoclass otherwise.

valid:Returns true if element meets all its validation constraints and is therefore considered to be valid,or false if it fails any constraints.If true,the element matches the :valid CSS pseudo-class.

valueMissing:Returns true if the element has a required attribute,but no value or false otherwise.If the element is invalid,this method also fires an invalid event on the element.

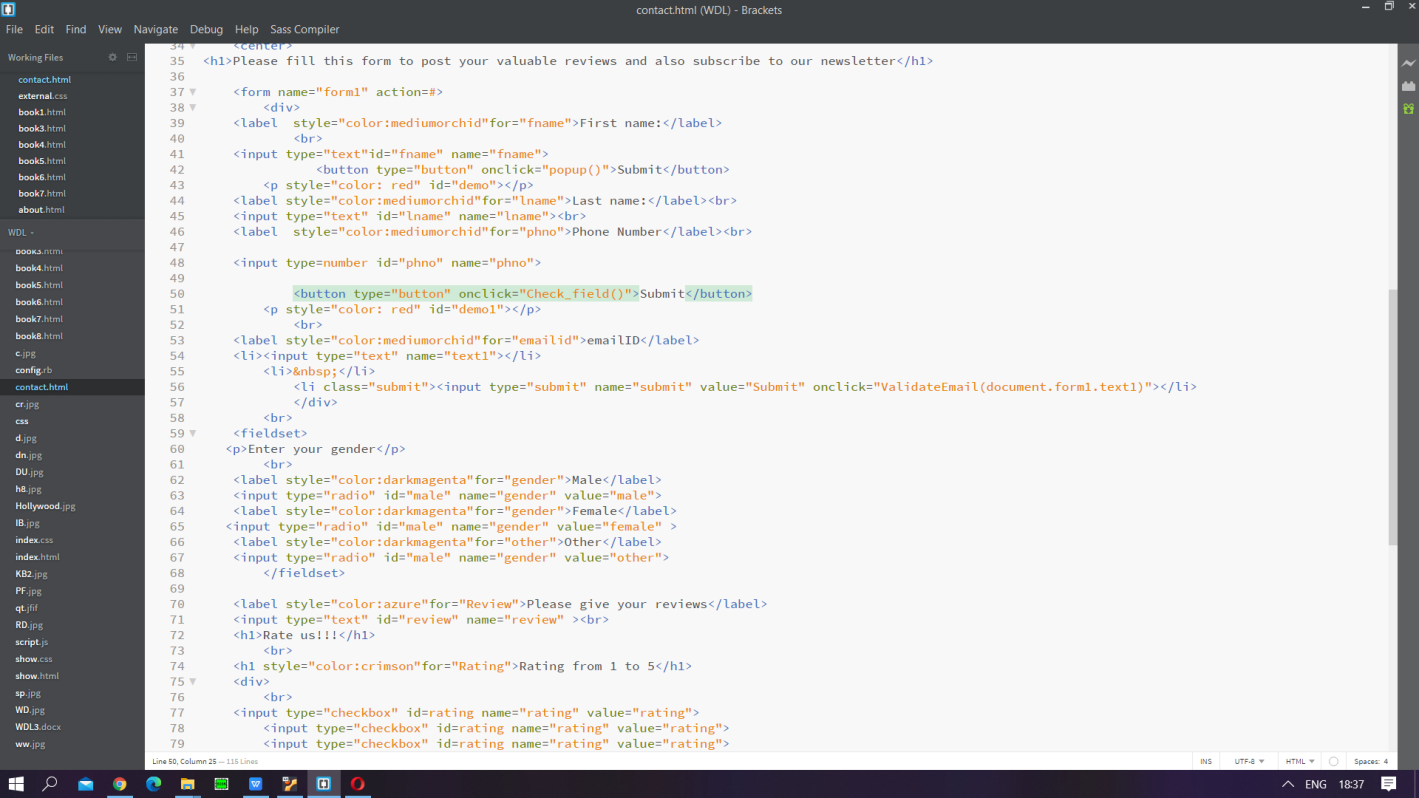
willValidate:Returns true if the element will be validated when the form is submitted,false otherwise.

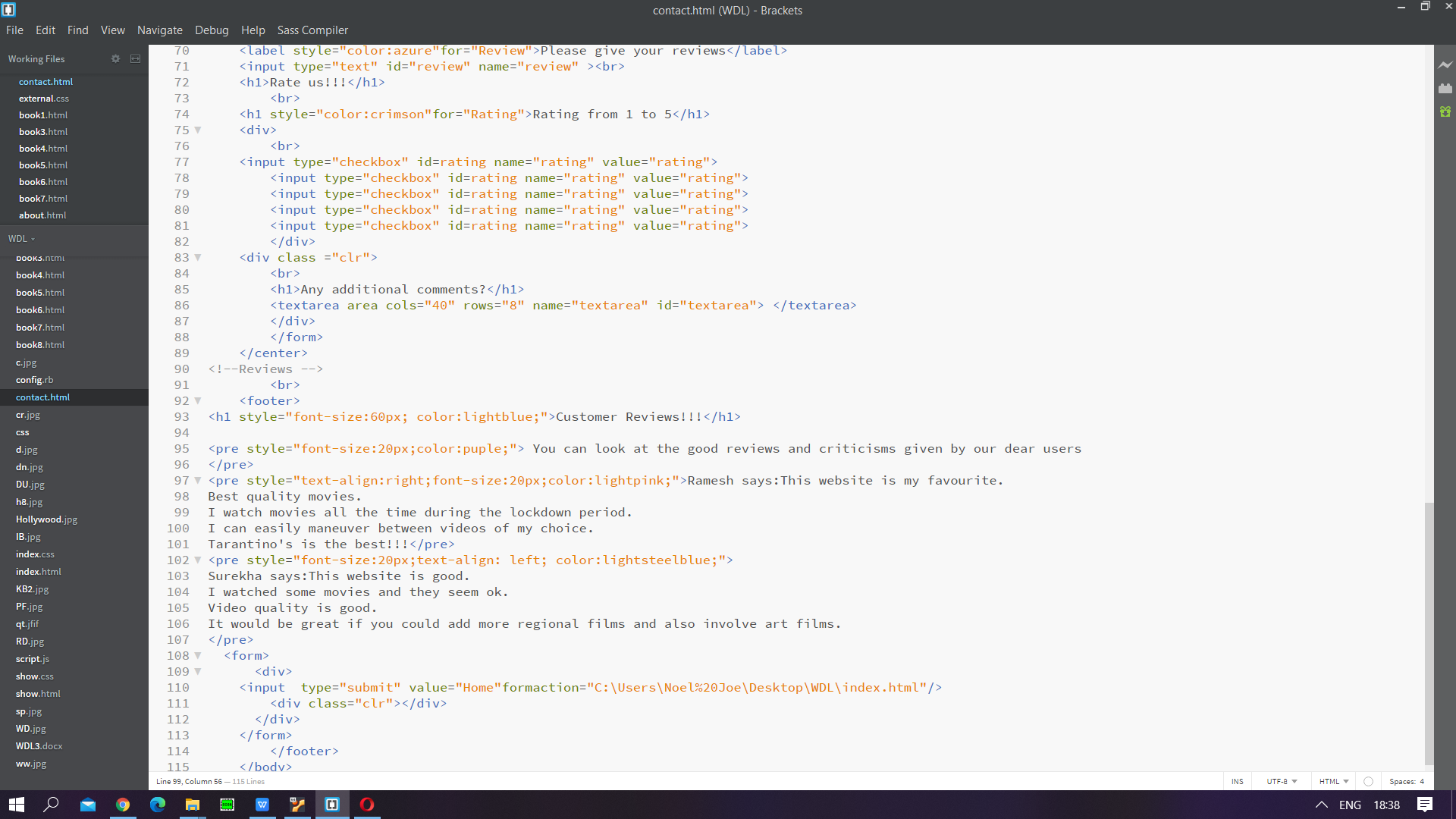
checkValidity():returns True if the element’s value has no validity problems,false otherwise.

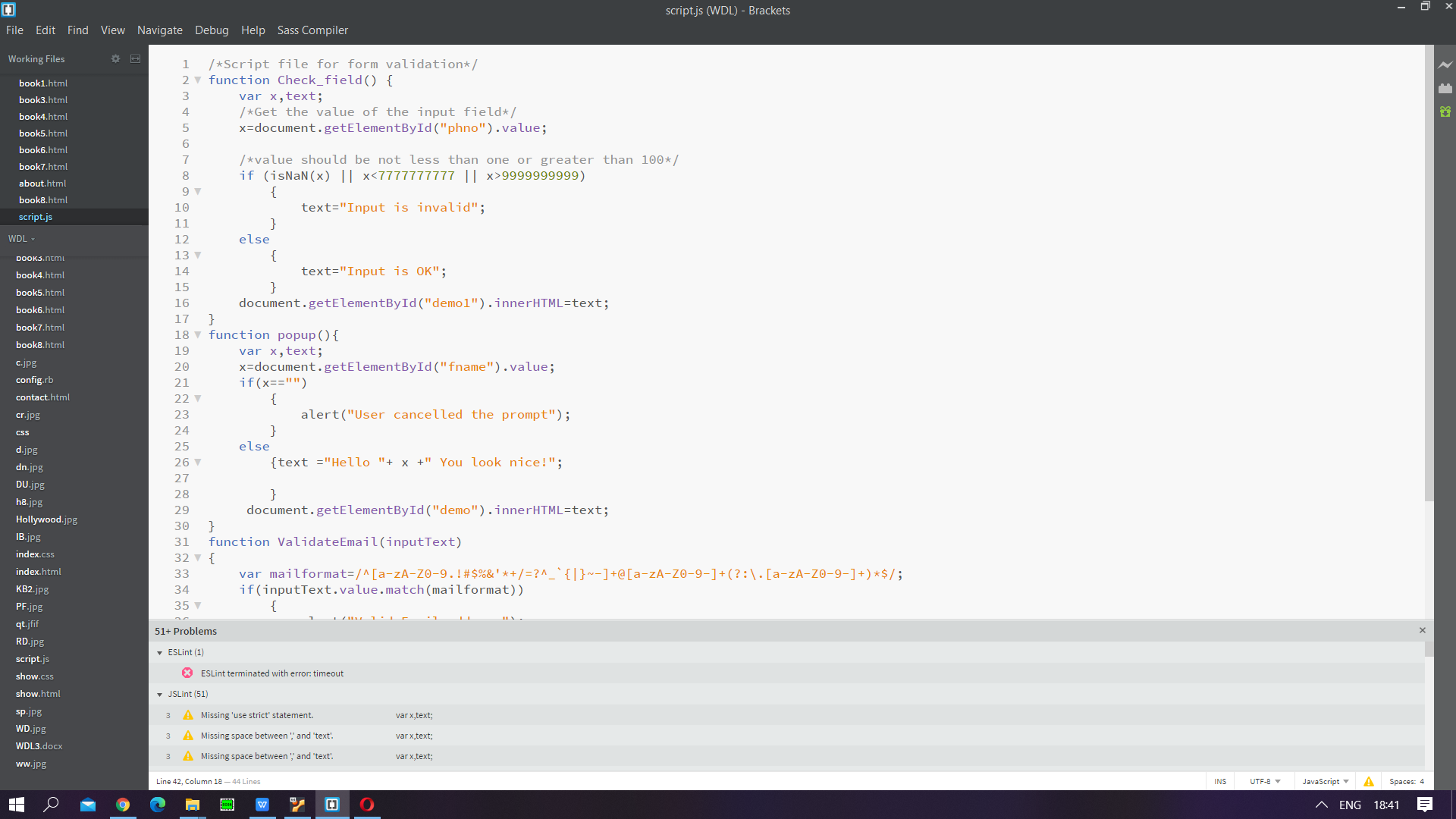
setCustomValidity(message):Adds a custom error message to the element; if you set a custom error mesage,the element is considered to be invalid,and the specified error is displayed.This lets you use JavaScript code to establish a validation failure other than other HTML 5 constraints.

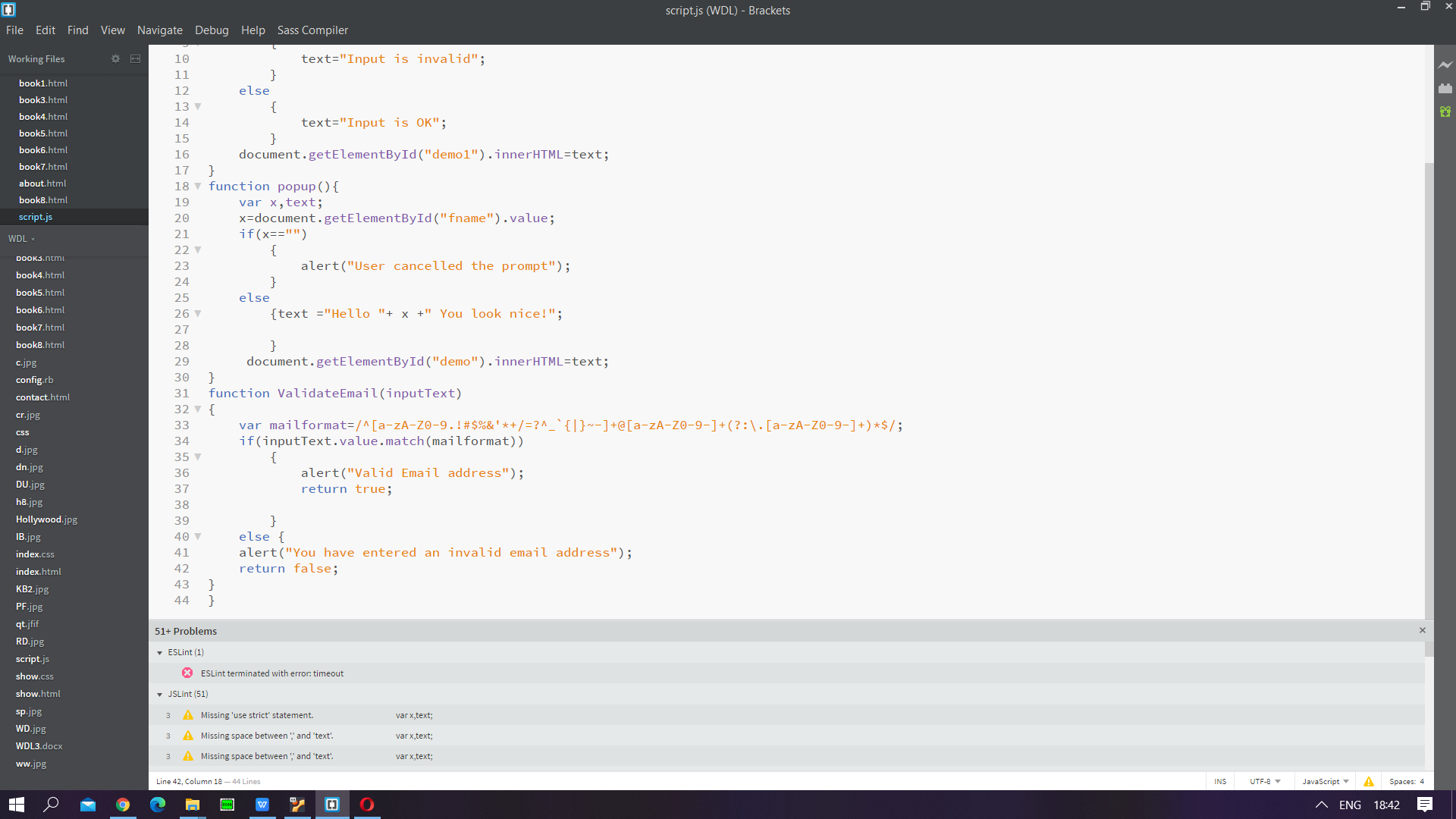
**Code and Output:-**

**HTML file**

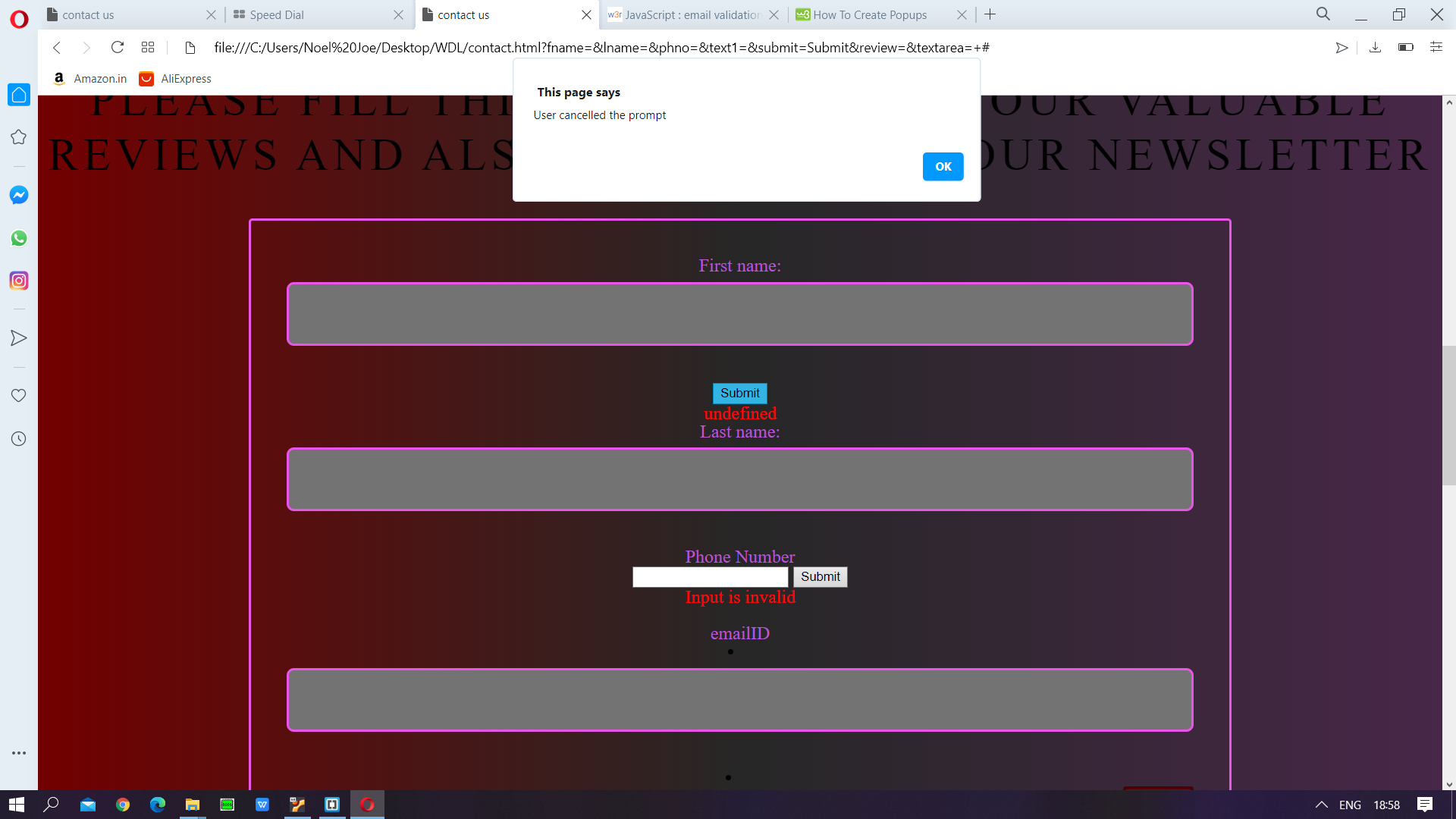


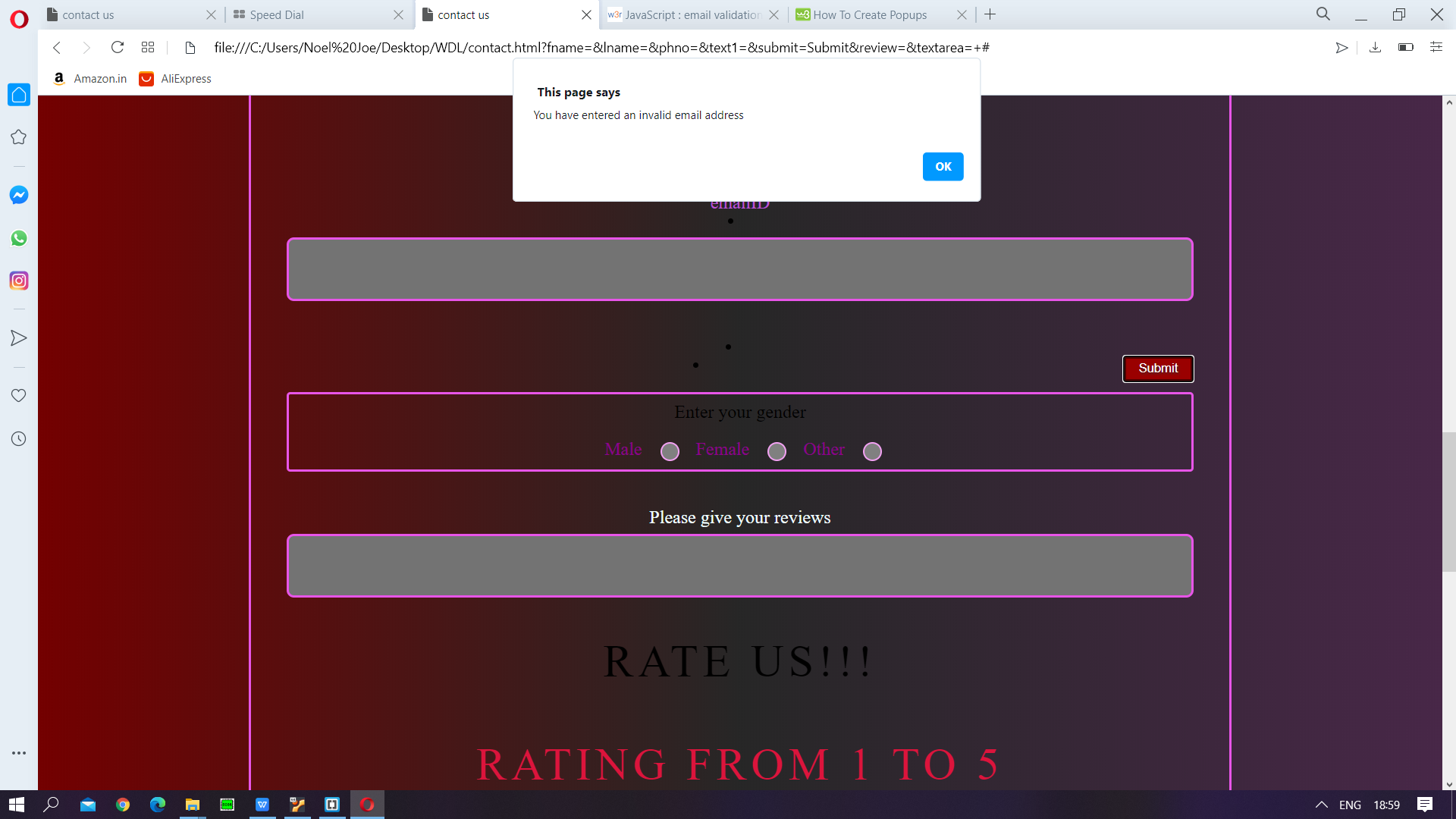


**Javascript FILE** 



**OUTPUT**





**Conclusion and Outcome:**- I was successfully able to design and validate forms with JavaScript.