

INTRODUCTION

What is form validation?

Go to any popular site with a registration form, and you will notice that they provide feedback when you don't enter your data in the format they are expecting. You'll get messages such as:

- "This field is required" (You can't leave this field blank).
- "Please enter your phone number in the format xxx-xxxx" (A specific data format is required for it to be considered valid).
- "Please enter a valid email address" (the data you entered is not in the right format).
- "Your password needs to be between 8 and 30 characters long and contain one uppercase letter, one symbol, and a number." (A very specific data format is required for your data).

This is called form validation. When you enter data, the browser and/or the web server will check to see that the data is in the correct format and within the constraints set by the application. Validation done in the browser is called client-side validation, while validation done on the server is called server-side validation. In this except, we are focusing on client-side validation.

If the information is correctly formatted, the application allows the data to be submitted to the server and (usually) saved in a database; if the information is not correctly formatted, it gives the user an error message explaining what needs to be corrected and lets them try again.

We want to make filling out web forms as easy as possible. So why do we insist on validating our forms?

There are three main reasons:

- We want to get the right data, in the right format. Our applications will not work properly if our users' data is stored in the wrong format, is incorrect, or is omitted altogether.
- We want to protect our users' data. Forcing our users to enter secure passwords makes it easier to protect their account information.
- We want to protect ourselves. There are many ways that malicious users can misuse unprotected forms to damage the application (see [Website security](#)).

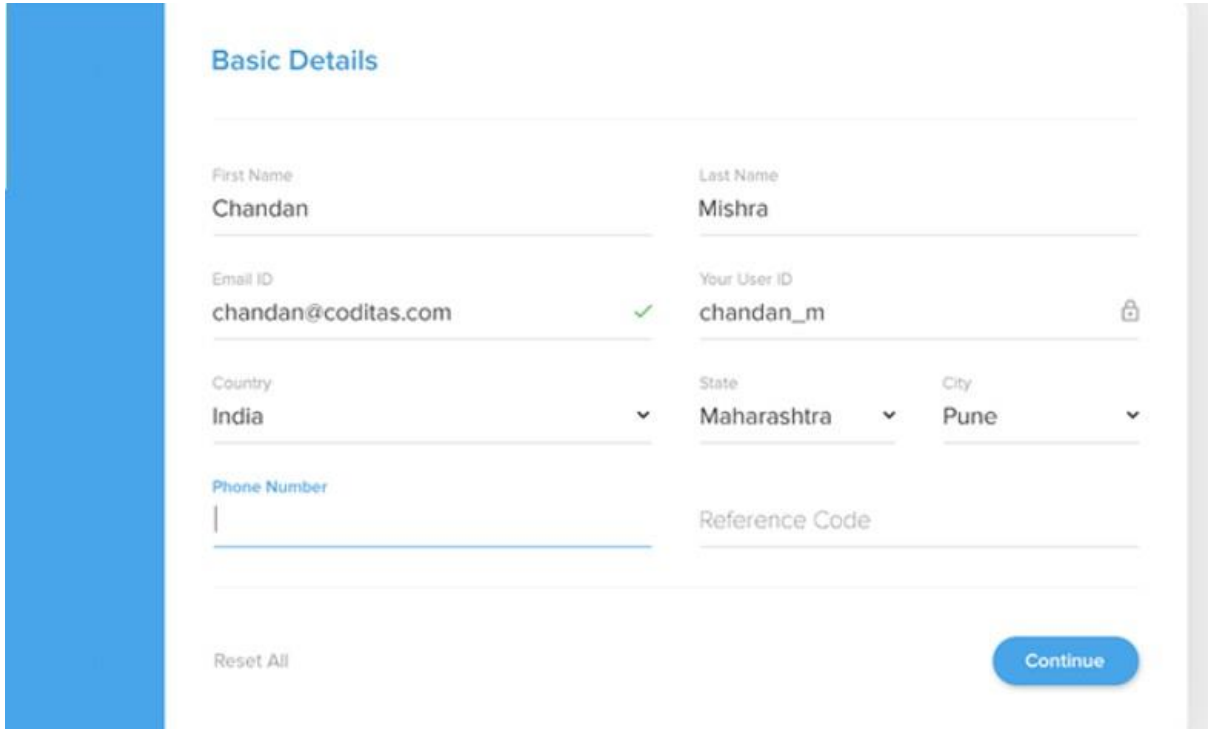
THE REQUIREMENTS

This project will give you practice creating HTML forms and using JavaScript to process their contents to generate interesting or useful outputs. You will demonstrate the ability to:

- Create HTML forms with input elements and a button,
- Write JavaScript functions that are used as event handlers,
- Write statements that gather the information from the input fields, store the values into variables, and perform some operations with them,
- Incorporate conditional execution of code to perform file size calculations, and display results to the user, by dynamically changing the HTML page.

Requirements

See the form below. You are required to recreate the form exactly as displayed and refer to the three reasons explained in the introduction to validate the form for invalid input. Note that the Phone Number field currently has focus in the image, hence the blue bottom border.



Basic Details

First Name	Last Name	
Chandan	Mishra	
Email ID	Your User ID	
chandan@coditas.com ✓	chandan_m 🔒	
Country	State	City
India ▼	Maharashtra ▼	Pune ▼
Phone Number	Reference Code	

Reset All Continue

CONSTRAINTS:

- All the fields are required.
- Validation must occur as soon as the user moves focus from one field to the next. This applies to text-based fields of entry (Name, Last Name, etc).
- *First Name* and *Last Name* must be letters only.
- The *user ID* and *Reference Code* can be any valid string.
- All drop downs must have at least three valid options each, as well as a *default* invalid option.
- Use icons to enhance the visual aspects of the validation. See the *Email ID* field.
- Create other visual effects to display invalid fields. You are strongly encouraged to do a bit of research in this regard.
- On submission, the form should redisplay all the options back to the user if there are invalid fields in the original fields of entry. A form that behaves this way is called a sticky form. The invalid fields should be indicated as well.
- If all the fields are valid, do not show the form again after submission. Instead, you should display a summary of the information that was entered by the user.

WEIGHTED SCORING

Aspect	Weight
<i>Form Elements</i>	<i>10</i>
<i>Formatting</i>	<i>20</i>
<i>Validation</i>	<i>70</i>
TOTAL	100

ADDITIONAL INFORMATION

- Late assignments will not be marked.
- Plagiarized assignments will not be marked.
- See the attached spreadsheet. Please indicate your groups and the names of your group members.
- There must be one submission per group. It is very important for you to nominate the person who will be responsible for submitting the assignment.
- The submitted assignment must include the names of the group members, as well as the group number. You can add this detail in the header area on your HTML page that contains the form.