The Little Lemon restaurant

wants to expand its service by

taking online orders to

deliver food to customers at home.

To do this, they need their customers

to set up an account on their website.

In this video, you will learn how to create

a simple signup form for

user accounts on the Little Lemon website.

I've set up my basic HTML document structure

in a file named, signup.html.

I start off by adding the form element.

The restaurant requires that

the customer data is sent securely.

I will set the method attribute to post so that

is submitted using an HTTP post request.

You will learn more about

different methods of form submission later.

For now, you just need to know that if you send

a form via an HTTP post request,

it improves the data security of the form.

The restaurant wants to collect

the customer's first name,

last name, and email address.

Of course, the form also needs a field for the password,

and another to confirm the password.

I need to create a form with five fields in total.

I'll start by adding the input elements,

also known as the form fields.

First, I'll set up six div elements,

one for each input element,

and one for the sign up button.

The div elements will cause each field of

the form to display in its own CSS block.

Now, I'll start adding the input elements.

I add an input element for

the first name to the first div element.

Since the user will input

plain texts for this form field,

I set the type attribute to text.

I also set the name and id attributes to user\_first\_name.

Remember that the name attribute sets

the key for the value submitted to the web server.

Later in this video,

I'll explain why I set the ID attribute.

Next, I add an input element for the last name.

Like the first name element,

I set the type attribute to text,

and the name and ID attributes to user\_last\_name.

I then add an input element for the email address.

To make sure users enter

a valid email address in this field,

I set the type attribute to email.

Earlier you learned that doing so

establishes HTML's client-side validation.

I also set the name and ID attributes to user\_email.

Next, I add the input element for the password.

I want the password to be hidden

on screen as the user types it,

so I set the type attribute to password.

When an input field is set to password,

each character that a user enters

will display as a placeholder character,

such as the bullet point.

The content in the field will contain the password

that the user enters, but it will be hidden.

I then set the name and ID attributes to user\_password.

Finally, I add the input element

for confirming the password.

Again, I set the type to password to

hide the content of the input field on screen.

I then set the name and

ID attributes to user\_confirm\_password.

Now that our input fields are set up,

I save the file and open

the Live Preview to check the webpage.

Everything displays correctly.

However, I'd like to improve the user experience

by informing the user what each field represents.

To do this, I'll add a label element

before each input element.

Now I want to explain why I set

the ID attribute for

each input field earlier in the video.

The reason for this,

is that it allows me to

associate a label with each input element.

Labels improve the user experience and

assist accessibility software and understanding the form.

I now add a label element above the first input element.

Inside the label element,

I add the descriptive text for the input field.

In this case, the text description will be first name.

Then, to associate the label with the input field,

I add the four attribute and set its value to match

the value of the ID attribute of the input element.

Finally, I'll add a line break tag after

the label element so that

the input field displays below the label.

I then repeat the same process for each input element.

I save the file again and open the Live Preview.

Oh, yes, that's much better.

The form labels now display above the input fields.

To finalize the form,

I need to add one more thing, the submit button.

Inside the last div element,

I add a button element.

I set the button type attribute to submit.

Then inside the button element,

I add the text, sign up.

I save the file again and open the Live Preview.

The form is now ready to use.

Once the backend is set up by

a backend developer to

accept the data submitted via the form,

users will be able to create their accounts.

In this video, you learned how to create

a basic form with several different input fields,

each in its own CSS block,

you covered how to set the method

attribute to submit the form in a secure way,

and you also now know how to associate a label

with an input field to improve the user experience.

Good luck with creating your own signup forms.

You are asked to help create a signup form. What can you do to improve the user experience of the form? Select all that apply.



Ensure that the correct input types are used.

Correct

That's right! Using specific input types makes it easier to complete a form correctly because users are warned if they enter an incorrect response.



Only use server-side validation.



Add labels for each input element.

Correct

That's right! Labels will help users understand what they need to add in each field. It is important to set the ID attribute of each input field so that the corresponding label can be associated with it.

Solution to Creating Form

**Create and test a form (solution)**

The following code is an example solution for the previous exercise.

This code is placed inside the <main> element of index.html.





<form>

    <div>

        <label for="username">Username</label>

        <input type="text" id="username" required minlength="2">

    </div>

    <div>

        <label for="password">Password</label>

        <input type="password" id="password" required minlength="2">

    </div>

    <button type="submit">Log In</button>

</form>

While reviewing the code, note the following items:

* The label *for* attribute value matches the value of the *id* attribute on the corresponding input element
* The *type* attribute is set to *text* for the username input
* The *type* attribute is set to *password* for the password input
* Each field has a *required* attribute to enable client-side validation which checks that the form is filled out by the user
* Each field has a *minlength* attribute with its value set to 2. This enables client-side validation which will prompt the user if the content of the field is less than 2 characters.