**REGISTRATION FORM**

* The vh unit stands for viewport height, and is relative to 1% of the height of the viewport.
* The method attribute specifies how to send form-data to the URL specified in the action attribute. The form-data can be sent via a GET request as URL parameters (with method="get") or via a POST request as data in the request body (with method="post").
* As the form will have three distinct sections,add  fieldset elements within the form element.
* Add label elements as per required.
* The rem unit stands for root em, and is relative to the font size of the html element.
* As label elements are inline by default, they are all displayed side by side on the same line, making their text hard to read. To make them appear on separate lines, add display: block to the label element, and add a margin of 0.5rem 0, to separate them from each other.
* label {

  display: block;

  margin: 0.5rem 0;

}

* Following accessibility best practices, link the input elements and the label elements together using the for attribute.
* Specifying the type attribute of a form element is important for the browser to know what kind of data it should expect. If the type is not specified, the browser will default to text.
* Give the first two input elements a type attribute of text, the third a type attribute of email, and the fourth a type attribute of password.
* The email type only allows emails with a @ and a . in the domain. The password type obscures the input, and warns if the site does not use HTTPS.
* To handle the form submission, after the last fieldset element add an input element with the type attribute set to submit and the value attribute set to Submit.
* o make the form more interactive, add the required attribute to the input elements in the first fieldset.
* Now, if you try to submit the form without filling in the required fields, you will see an error message.
* Certain type attribute values come with built-in form validation. For example, type="email" requires that the value be a valid email address.
* Add custom validation to the password input element, by adding a minlength attribute with a value of 8. Doing so prevents inputs of less than 8 characters being submitted.
* <label for="new-password">Create a New Password: <input id="new-password" type="password" minlength="8" required /></label>
* With type="password" you can use the pattern attribute to define a regular expression that the password must match to be considered valid.
* Add a pattern attribute to the password input element to require the input match: [a-z0-5]{8,}
* The above is a regular expression which matches eight or more lowercase letters or the digits 0 to 5.
* <fieldset>

        <label><input type="radio" /> Personal Account</label>

        <label><input type="radio" /> Business Account

</label>

        <label><input type="checkbox" required /> I accept the terms and conditions

</label>

      </fieldset>

* To relate the radio inputs, give them the same name attribute with a value of account-type. Now, it is not possible to select both radio inputs at the same time.
* <label><input name="account-type" type="radio" /> Personal Account</label>

        <label><input  name="account-type"type="radio" /> Business Account</label>

* What if you wanted to allow a user to upload a profile picture?
* Well, the input type file allows just that. Add a label with the text Upload a profile picture: , and add an input accepting a file upload.
* <fieldset>

<label><input type="file" />Upload a profile picture: </label>

      </fieldset>

* Adding a dropdown to the form is easy with the select element. The select element is a container for a group of option elements, and the option element acts as a label for each dropdown option. Both elements require closing tags.
* The dropdown options are currently empty. To give them content, add the following text to each subsequent option element:
* The textarea element acts like an input element of type text, but comes with the added benefit of being able to receive multi-line text, and an initial number of text rows and columns.
* The textarea appears too small. To give it an initial size, you can add the rows and cols attributes.
* The placeholder accepts a text value, which is displayed until the user starts typing.
* The border of the last fieldset element looks a little out of place. You can select the last element of a specific type using the last-of-type CSS pseudo-class, like this:

p:last-of-type { }

That will select the last p element.

* If you want the input and label text to appear on the same line,start, by giving the input elements in the second fieldset a class of inline.
* Instead of giving margin top,margin bottom,margin left, margin right,give margin:0,0,0,0 for T,R,B,L
* To style the submit button, you can use an attribute selector, which selects an element based on the given attribute value. Here is an example:

input[name="password"]

The above selects input elements with a name attribute value of password.

input[type="submit"] {

  display: block;

  width: 60%;

  margin:0 auto;

}