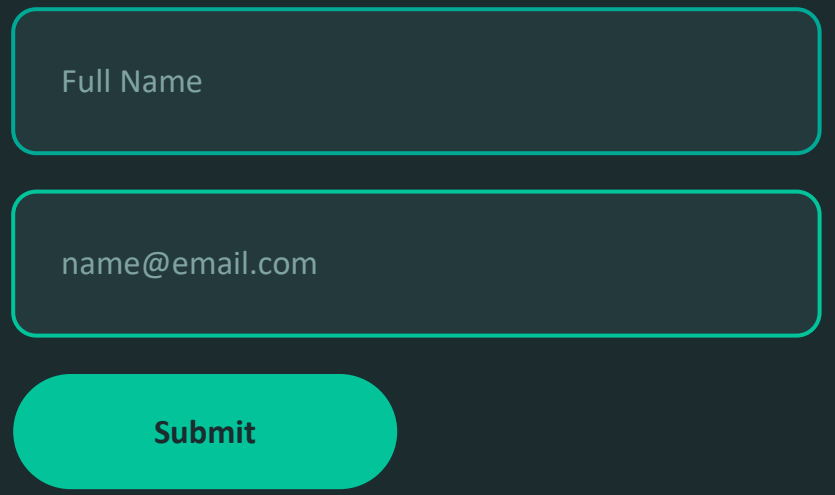
WEB DEVELOPMENT FUNDAMENTALS

HTML Forms

Collecting, structuring, and validating user input on the web

Front-End Fundamentals Track



AGENDA

What We'll Cover

01

Why Forms Matter

The role forms play in real web applications

02

Form Structure

The <form> element, action, and method

03

Input Types & Controls

Text, email, select, checkbox, radio, and more

04

Attributes & Validation

required, pattern, and built-in HTML5 checks

05

Best Practices

Labels, accessibility, and usability

06

Hands-On Exercise

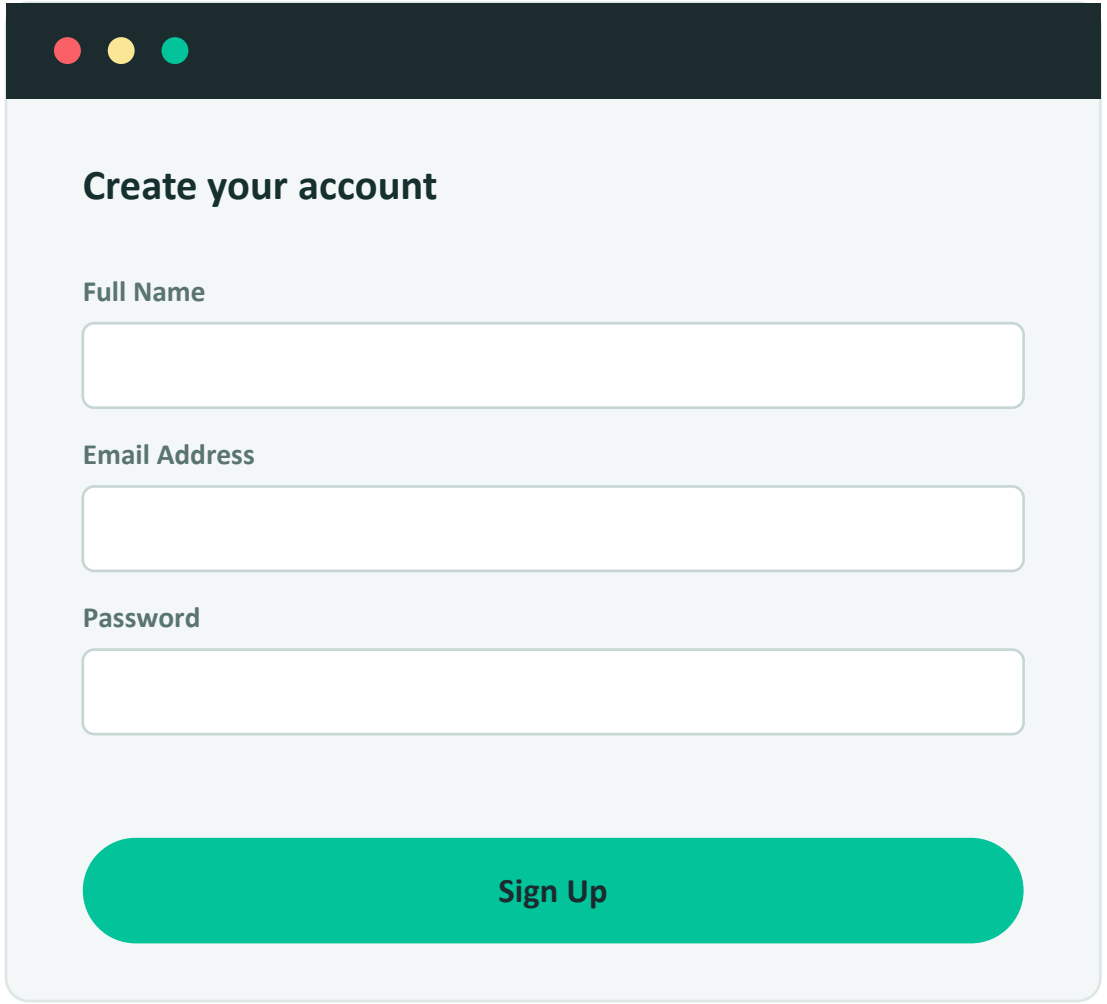
Build a working registration form

FOUNDATIONS

Why HTML Forms Matter

Forms are the primary way a website collects information from a visitor — and turns a page from something you read into something you use.

1. **Log In / Sign Up**
   Usernames, passwords, and account creation
2. **Search**
   Search bars across nearly every site
3. **Checkout**
   Shipping, billing, and payment details
4. **Feedback**
   Surveys, contact forms, and reviews



STRUCTURE

The <form> Element

Every form starts with a <form> tag. Its attributes tell the browser where to send the data and how to send it.

```
<form action="/register" method="post">
  <label for="email">Email</label>
  <input type="email" id="email" name="email">

  <label for="pwd">Password</label>
  <input type="password" id="pwd" name="pwd">

  <button type="submit">Sign Up</button>
</form>
```

action

URL that receives the submitted data

method

"get" appends data to the URL; "post" sends it in the request body

name

Identifies each field's value in the submitted data

CONTROLS

Common Input Types

`type="text"`

Free-form single-line text

`type="email"`

Validates an email pattern

`type="password"`

Masks characters as typed

`type="number"`

Numeric input, with min/max

`type="date"`

Native date picker

`type="checkbox"`

Toggle one or more options

`type="radio"`

Choose exactly one option

`type="file"`

Upload a file from device

type="text"

The default control — a single line of free-form text.

```
<input
  type="text"
  name="username"
  maxlength="20"
  placeholder="Username"
  value="">
```

maxlength

Caps how many characters can be entered

placeholder

Faint hint text shown only while the field is empty

value

Sets the field's starting (or current) content

pattern

Optional regex the entered text must match

type="email"

Same as text, but the browser checks it looks like an address.

```
<input
  type="email"
  name="email"
  required
  placeholder="you@example.com">
```

required

Blocks submission until a value is present

placeholder

Shows the expected format as a hint

multiple

Optional — allows a comma-separated list of addresses

(built in)

Browser checks for a basic name@host pattern automatically

type="password"

Masks each character as it's typed.

```
<input
  type="password"
  name="pwd"
  minlength="8"
  autocomplete="new-password">
```

minlength

Shortest number of characters accepted

autocomplete

"new-password" stops the browser suggesting a saved one

(security)

Masked on screen but sent as plain text — the connection itself needs HTTPS

type="number"

Numeric entry with a spinner and range limits.

```
<input
  type="number"
  name="age"
  min="16"
  max="99"
  step="1">
```

min / max

The lowest and highest accepted values

step

The increment the spinner arrows move by (default 1)

(built in)

Rejects non-numeric input automatically

type="date"

A native date picker, no JavaScript library needed.

```
<input
  type="date"
  name="dob"
  min="1900-01-01"
  max="2026-08-21">
```

min / max

Earliest and latest selectable dates (YYYY-MM-DD)

value

Pre-fills a specific date in the same format

(built in)

Renders a native picker in supporting browsers

type="checkbox"

An on/off toggle — one or several can be checked at once.

```
<input
  type="checkbox"
  name="newsletter"
  value="yes"
  checked>
```

value

The data sent ONLY if the box is checked — unchecked sends nothing

checked

Pre-selects the box when the page loads

(grouping)

Multiple boxes can share a name to submit a list of values

type="radio"

A group where exactly one option can be selected.

```
<input type="radio" name="level"
  value="beginner" checked>
<input type="radio" name="level"
  value="advanced">
```

name

MUST match across the group — it's what makes options mutually exclusive

value

The data sent for whichever option is selected

checked

Sets the default-selected option in the group

type="file"

Lets the person pick a file from their device to upload.

```
<input
  type="file"
  name="resume"
  accept=".pdf,.doc,.docx"
  multiple>
```

accept

Restricts which file types the picker shows/allows

multiple

Lets the person select more than one file at once

(form)

The <form> needs enctype="multipart/form-data" for uploads to work

CONTROLS

Labels, Choices & More

Beyond <input>, a handful of elements round out most forms.

`<label>`	Names a field and links to it via the for attribute — click the label to focus the field
`<select> / <option>`	A dropdown list of choices, e.g. a country or plan selector
`<textarea>`	Multi-line free text, such as a comment or message
`<fieldset> / <legend>`	Groups related fields together with a shared caption
`<button>`	Triggers the form — type="submit" sends it, type="reset" clears it

CONFIGURATION

Attributes That Do the Work

`required`

Blocks submission until the field has a value

`placeholder`

Faint hint text shown inside an empty field

`value`

Pre-fills or sets the current content of a field

`maxlength`

Caps how many characters can be entered

`min / max`

Sets bounds for number and date inputs

`pattern`

A regular expression the value must match

`disabled`

Greys out a field and excludes it from submission

`autofocus`

Puts the cursor in this field on page load

VALIDATION

Built-In HTML5 Validation

The browser checks these rules before the form is ever submitted — no JavaScript required.

```
<input
  type="email"
  name="email"
  required
>

<input
  type="text"
  pattern="[0-9]{5}"
  title="5-digit ZIP code"
>
```

- ✓ Invalid fields are flagged automatically with a browser tooltip
- ✓ required stops submission on empty fields
- ✓ type="email"/"url" check basic format on their own
- ✓ pattern lets you enforce a custom format with regex
- ✓ Always re-check on the server — client-side validation can be bypassed

BEST PRACTICES

Design Forms People Can Use

1

Label everything

Every input needs a visible, linked <label> — placeholders alone aren't enough for accessibility

2

Group related fields

Use <fieldset> for things like an address or a set of radio options

3

Show errors clearly

Pair browser validation with clear, specific error messages near the field

4

Keep it short

Only ask for what you need — every extra field lowers completion rates

HANDS-ON EXERCISE

Build a Sign-Up Form

Using only HTML, build a workshop registration form. No JavaScript or CSS required — this exercise is about structure and built-in validation.

- **Full Name**
  `Required text field`
- **Email Address**
  `type="email", required`
- **Password**
  `type="password", minlength="8"`
- **Age**
  `type="number", min="16", max="99"`
- **Preferred Track**
  `<select> with at least 3 <option> choices`
- **Experience Level**
  `Radio buttons: Beginner / Intermediate / Advanced`
- **Newsletter Opt-In**
  `A single checkbox`
- **Comments**
  `<textarea> for optional notes`

EXERCISE

Checklist & Hints

Before you consider it done

- [ ] The <form> wraps every field, with method="post"
- [ ] Every input has a matching <label for="...">
- [ ] Required fields actually block submission when empty
- [ ] The email field rejects an obviously invalid address
- [ ] The page still works if you press Submit with nothing filled in

Hints

```
<label for="track">Track</label>
<select id="track" name="track">
  <option value="frontend">Front-
End</option>
  <option value="backend">Back-End</option>
  <option value="fullstack">Full-
Stack</option>
</select>

<input type="radio" id="beg"
  name="level" value="beginner">
<label for="beg">Beginner</label>
```