Web Forms Basics

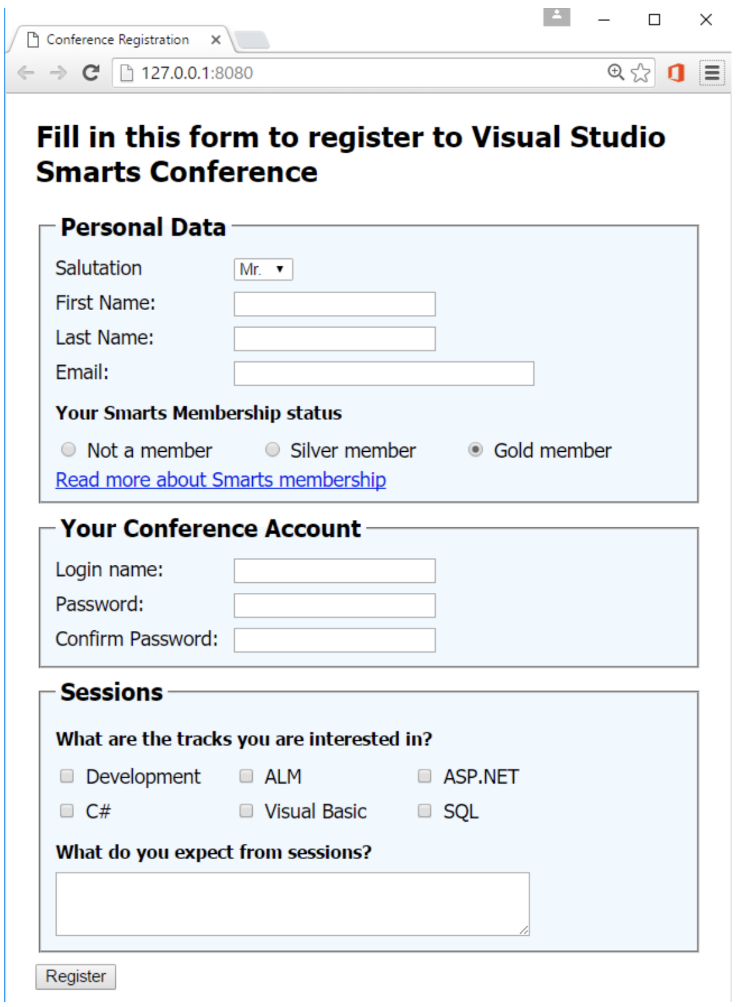
In this lesson, we'll learn the basic concept of web forms.
Let's begin!

Web pages are not just for reading documents and articles but are also for social and business applications. You can hardly imagine any kind of web app without **asking for some data from users**; just think about the most common functions like **login** and **registration**.

HTML forms have been a part of the markup since the earliest versions, and now, after *several tweaks* we have a refined model of forms in HTML5 that still works with older browsers.

How web forms work

The image below shows a web form in action which contains user interface components or **controls**. Not only are **textboxes**, **checkboxes**, and **radio buttons** controls in this form, but so are labels, the frames surrounding the sections, and the **Register button**.


Conference Registration

127.0.0.1:8080

Fill in this form to register to Visual Studio Smarts Conference

Personal Data

Salutation	Mr.
First Name:	
Last Name:	
Email:	

Your Smarts Membership status

Not a member / Silver member / [✓ Gold member]

Read more about Smarts membership

Your Conference Account

Login name:	
Password:	
Confirm Password:	

Sessions

What are the tracks you are interested in?

Development	ALM	ASP.NET
C#	Visual Basic	SQL

What do you expect from sessions?

Register

A web form in action

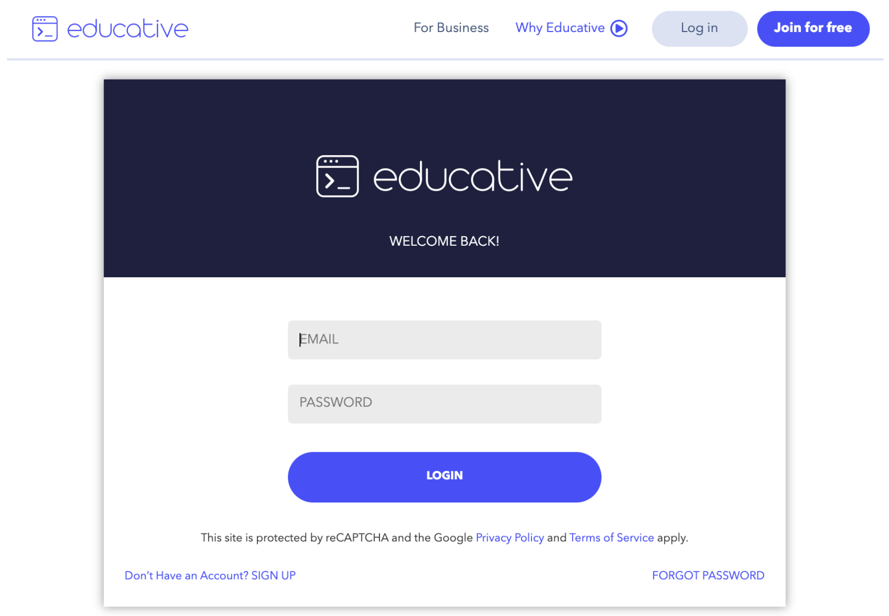
To design and code web forms, you need to understand how they work.

Here is a simplified overview:

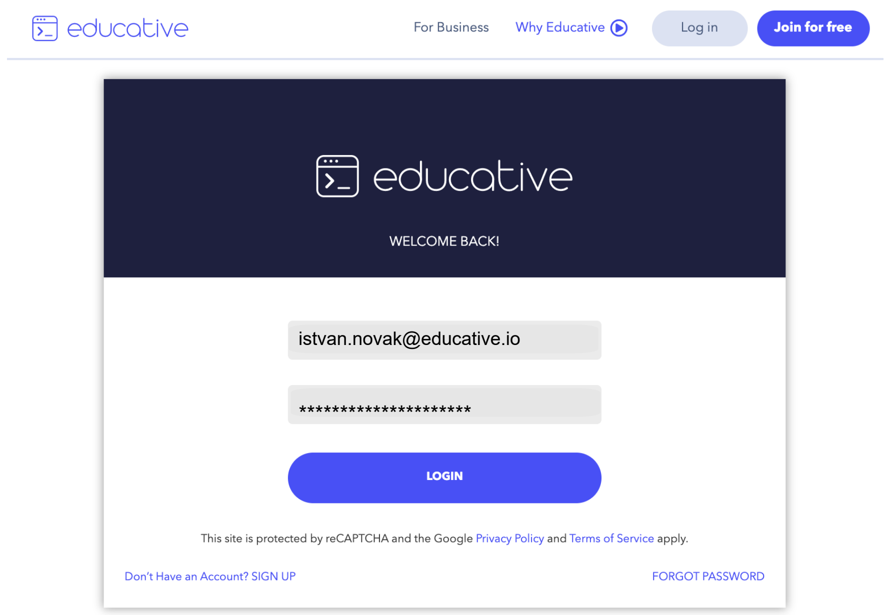
- When the browser displays a web form, it renders its controls just like any other element, such as text, menus, images, and so on.
- A user fills it out and then clicks a button that submits the form.
- The submission of a form means that the browser collects the data entered by the user, puts it into an HTTP/HTTPS request, and sends it to the server.
- When the server-side receives the request, the web-server finds the type of application or module to dispatch the request with the collected data.
- The entity that receives the request analyzes the information collected and decides what to do next. For example, if the request contains

registration data, the web server may store it in the database and send

back a message about the success of the registration. Or, if the data is invalid, for example, a mandatory field has not been filled out, the web-server might send back the original form with the data specified by the user, and with additional markup that summarizes the issues found by the server application.


educative
For Business
Why Educative
Log in
Join for free
educative
WELCOME BACK!
EMAIL
PASSWORD
LOGIN
This site is protected by reCAPTCHA and the Google Privacy Policy and Terms of Service apply.
Don't Have an Account? SIGN UP
FORGOT PASSWORD

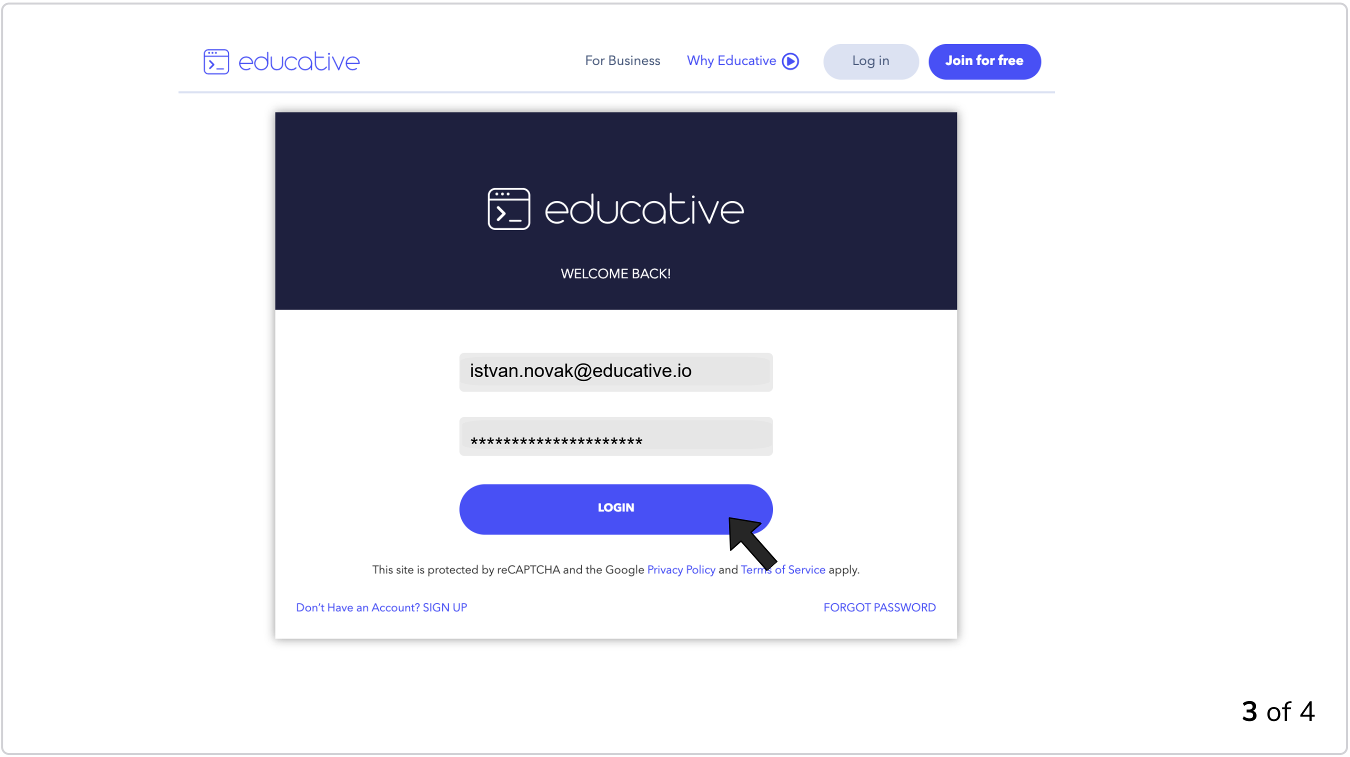
1 of 4


educative
For Business
Why Educative
Log in
Join for free
educative
WELCOME BACK!
LOGIN
This site is protected by reCAPTCHA and the Google Privacy Policy and Terms of Service apply.
Don't Have an Account? SIGN UP
FORGOT PASSWORD

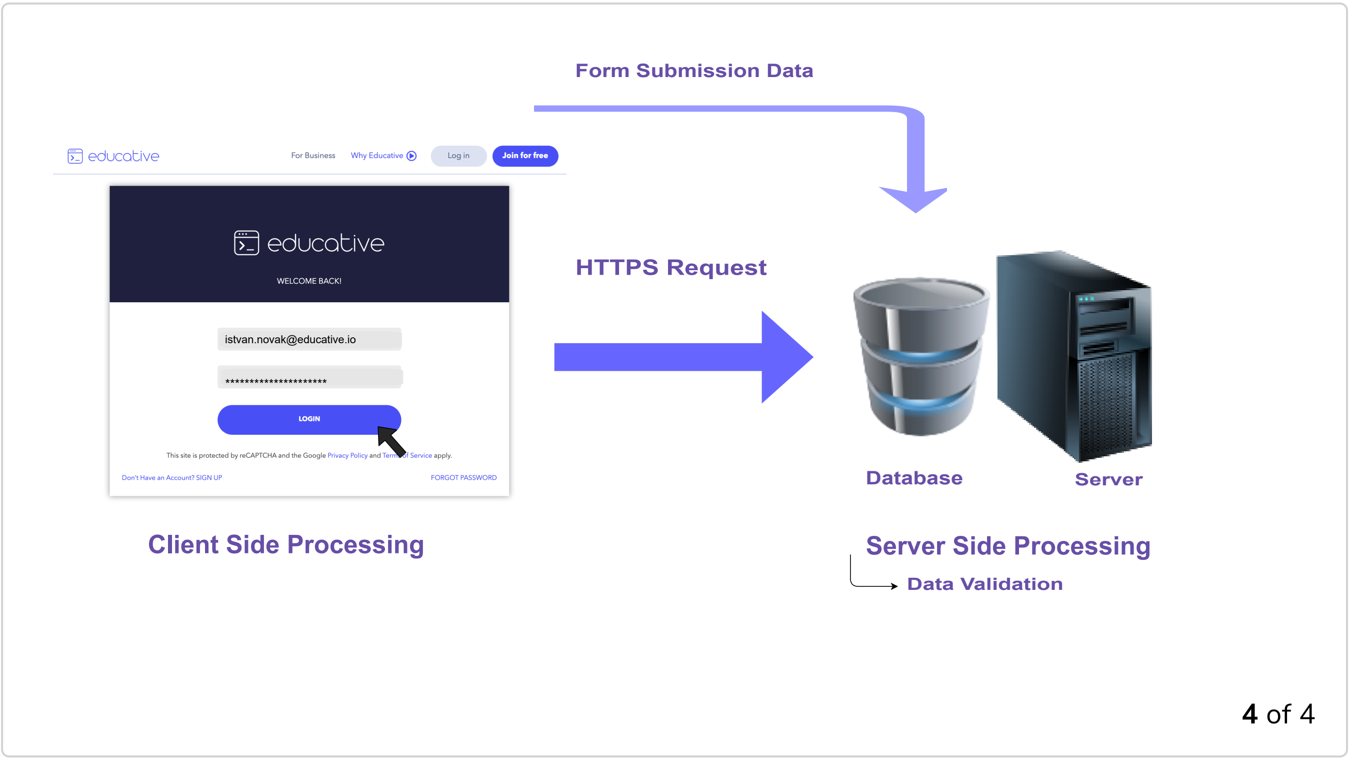
istvan.novak@educative.io

****

2 of 4


educative
For Business
Why Educative
Log in
Join for free
educative
WELCOME BACK!
istvan.novak@educative.io
****
LOGIN
This site is protected by reCAPTCHA and the Google Privacy Policy and Terms of Service apply.
Don't Have an Account? SIGN UP
FORGOT PASSWORD

3 of 4



4 of 4

The fundamental steps:

So, this process contains three fundamental steps:

1. The browser collects the form data and sends it to the web server.
2. The web server processes the data and sends the result back to the browser.
3. The browser visualizes the result which may be a web page, an error

3. The browser visualizes the result which may be a web page, an error
   message, or some other kind of information.

Of course, the most difficult part of this process is step two, which is carried out at the server side. There are many server technologies that can process web forms, but in this course, we won't dive into them, we will only scratch the surface.

As the web evolved and users expected fluent experience, new asynchronous technologies appeared, such as AJAX (Asynchronous JavaScript and XML). These do not expect the web server to retrieve a full web page, but only the part of the page affected by the changes in regard to the information the users specified.

This approach results in smaller network packages and less flickering on the screen. The main virtue of asynchronous technologies is that they can send data to and retrieve data from a server asynchronously in the background, without preventing user interaction.

In this chapter you will focus on building and using web forms at the client side.

In the *next lesson*, we'll learn how to represent web forms in HTML.

Stay tuned!