WEB TECHNOLOGIES

& ALL THE OTHER STUFF THAT HAPPENS

BROWSER TECHNOLOGIES

- HTML (hypertext markup language)
  - XML Markup defining tags to structure your website
- CSS (cascading style sheet)
  - Text markup that selects element(s) and applies styles
  - Adjusting the visual representation of your HTML tags
- JavaScript
  - Advanced interactions using a simple programming language
  - Modify HTML/CSS properties dynamically
  - Handle clicks, dragging, form submissions, and more

THE BACKEND - SERVER PROGRAMMING

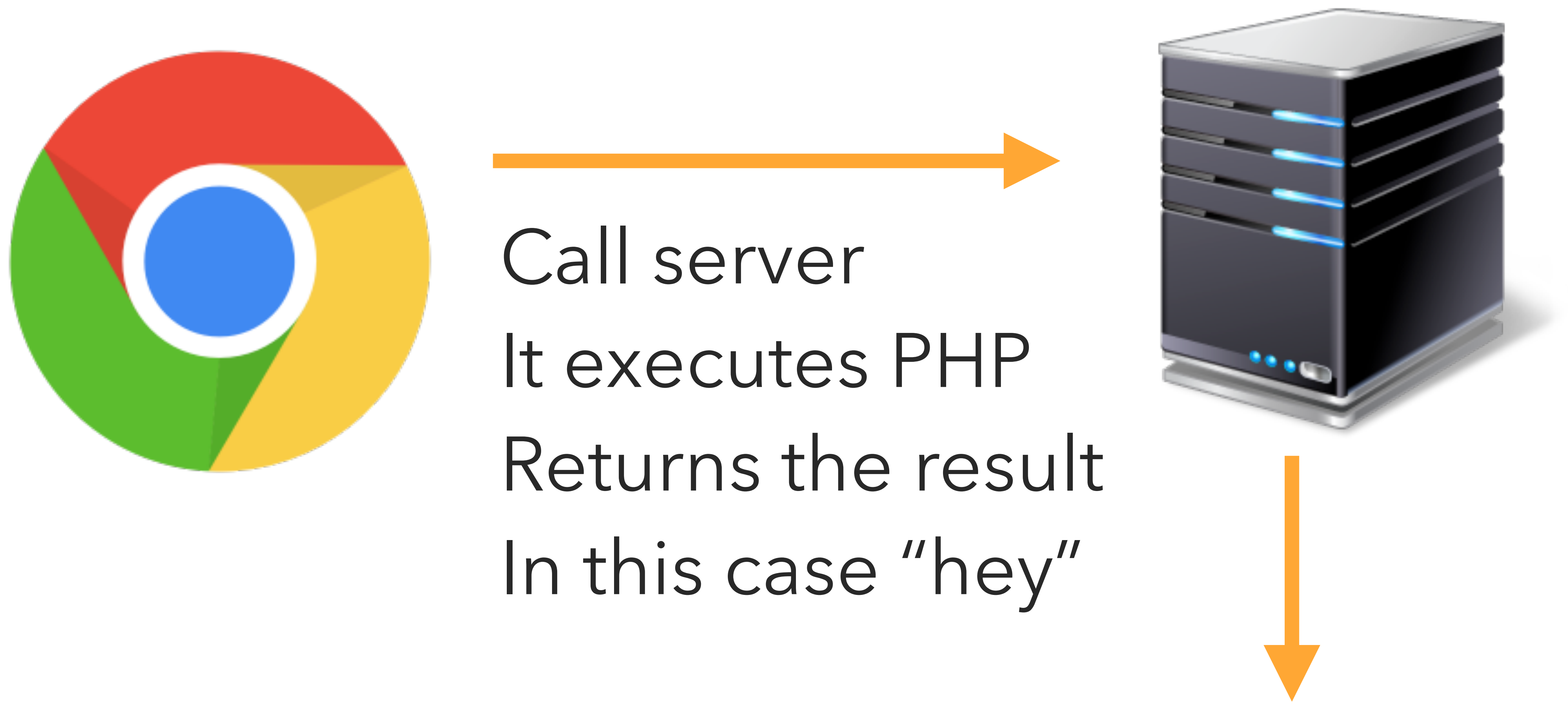
- A number of programming languages utilized by the web
  - PHP, Java, Python, and more. Differences in the way the language works and the syntax used (follow many common similarities)
- These languages are NOT executed in your browser, they are run in/by your server
- For example a simple test.php page
  - Inside it only has the following "<?php echo 'Welcome'; ?>"
  - You type http://mysite.com/test.php when the server sees the request it performs the code inside test.php
  - Then it returns the result to the browser as plain HTML (just Welcome)

THE BACKEND - SERVER PROGRAMMING

index.php

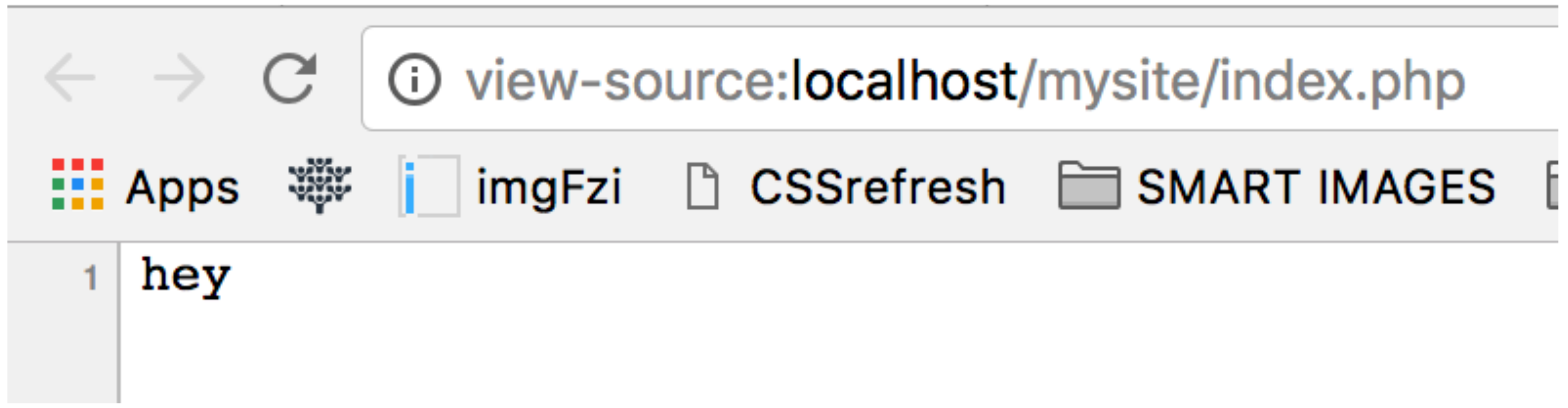
```
<?php
echo "hey";
?>
```

mysite.com/index.php



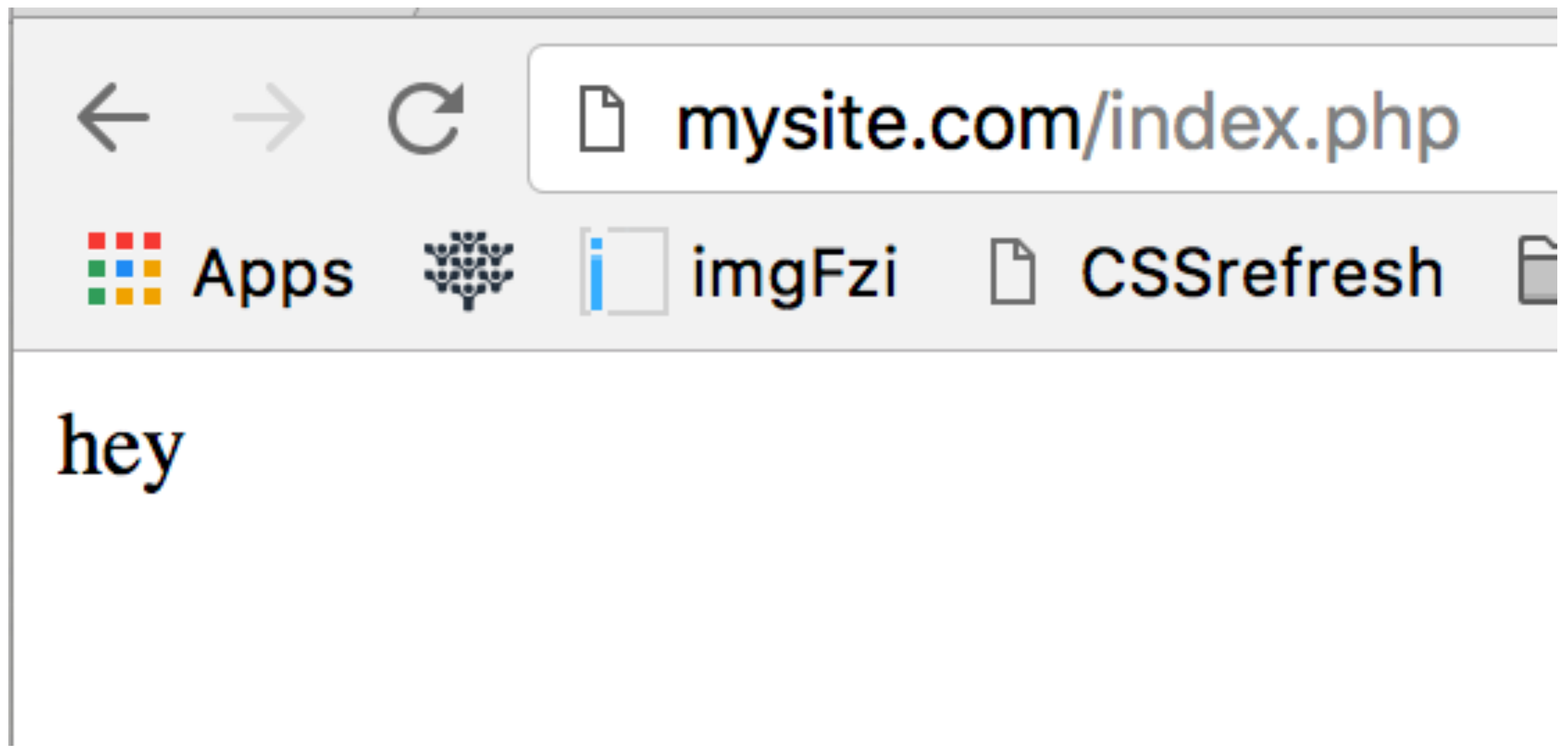
Call server
It executes PHP
Returns the result
In this case "hey"

View Page Source



view-source:localhost/mysite/index.php

```
hey
```



mysite.com/index.php

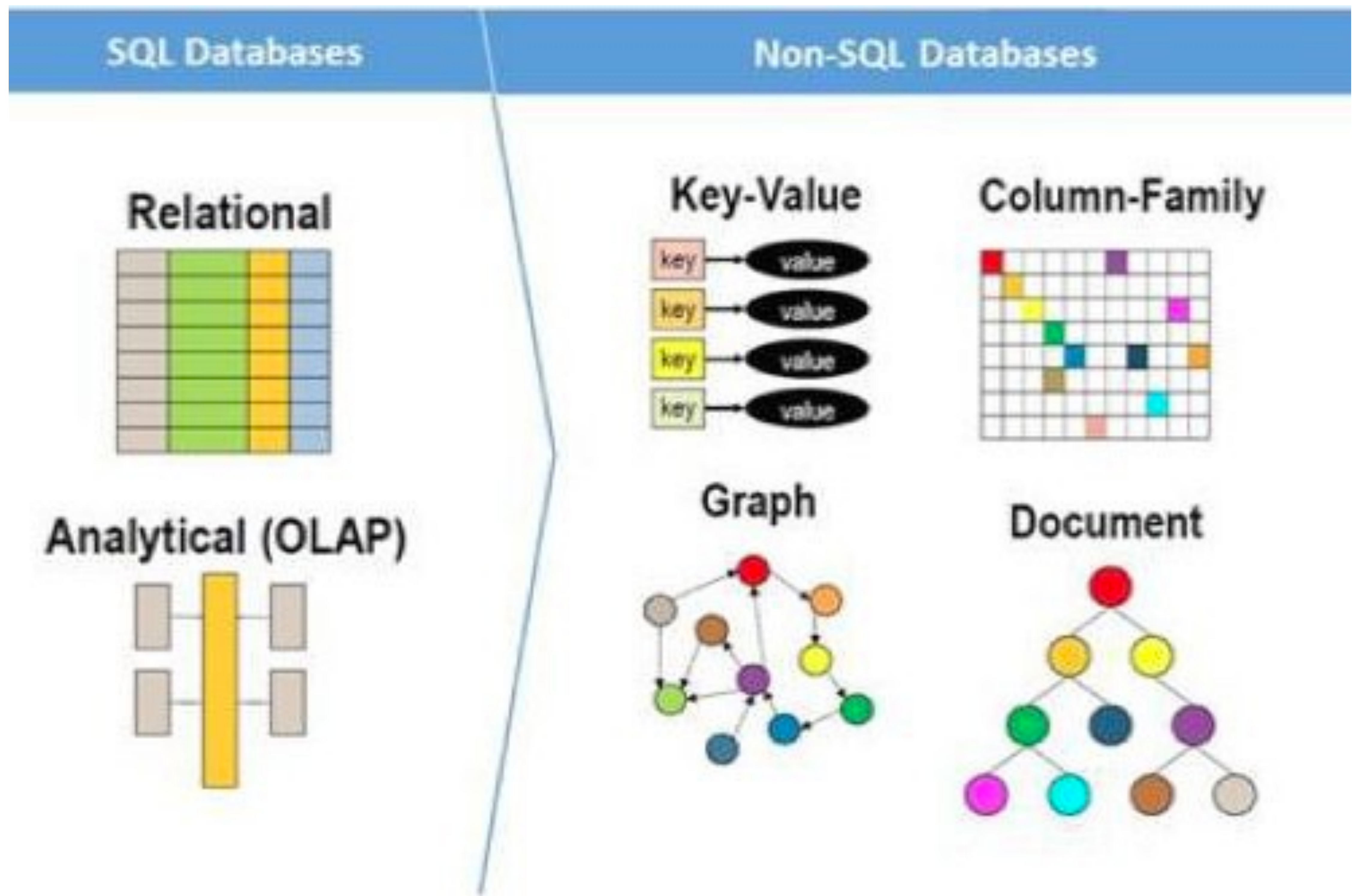
hey

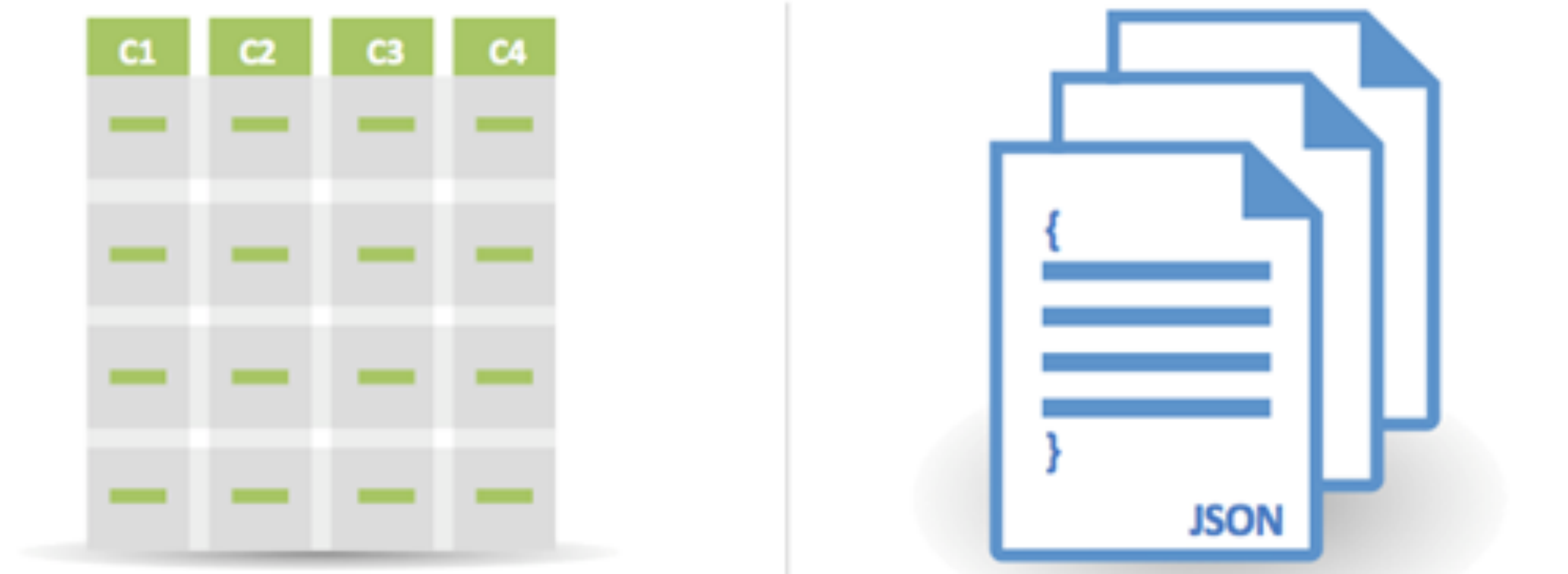
THE BACKEND - SERVER PROGRAMMING

- There are many important reasons you use server side languages
  - Keep your business logic private (you can’t read contents of test.php, just the result) meaning nobody can steal your app
  - Can access the database in the server and make changes
  - Uses advanced programming techniques not found in JavaScript

THE BACKEND - DATABASES

- Two main types
  - SQL (MySQL) - Similar to an excel spreadsheet
  - NoSQL - Uses a document data model





Relational data model

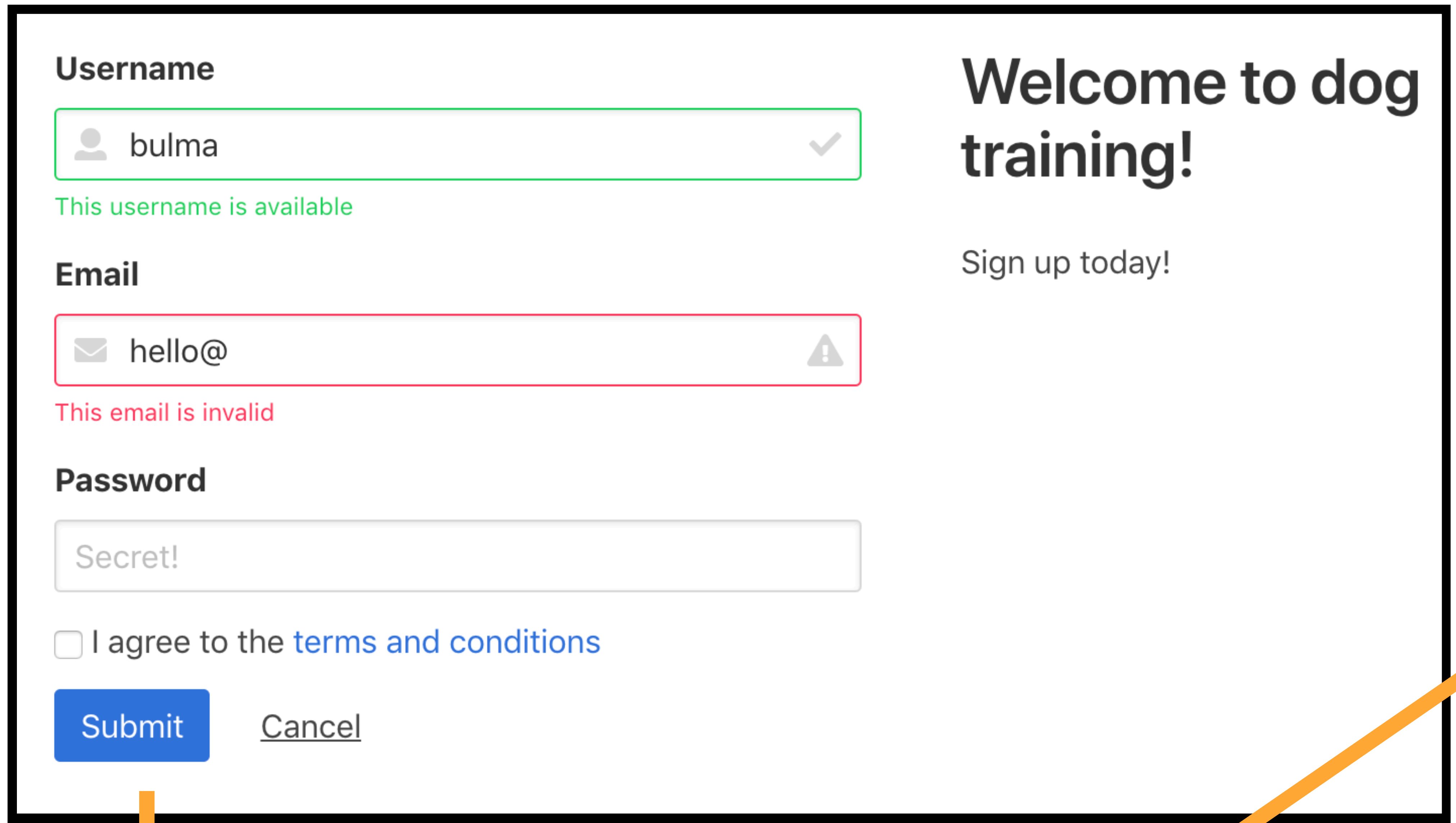
Highly-structured table organization with rigidly-defined data formats and record structure.

Document data model

Collection of complex documents with arbitrary, nested data formats and varying "record" format.

REAL WORLD BACKEND EXAMPLE

Visit dogtraining.com



Username

bulma

This username is available

Email

hello@

This email is invalid

Password

Secret!

I agree to the terms and conditions

Submit Cancel

Welcome to dog training!

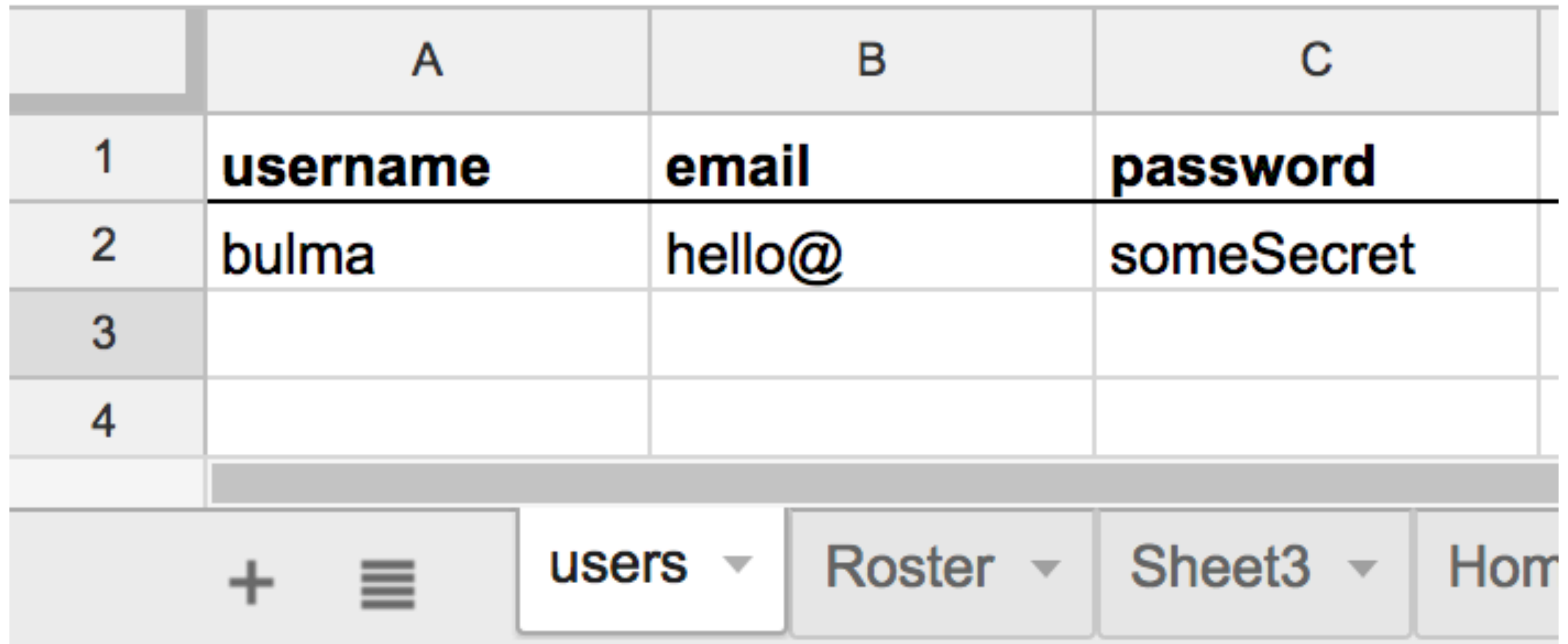
Sign up today!

Submit form which goes to /signup.php

We pass data:
Username = bulma
Email = hello@
Password = someSecret

signup.php

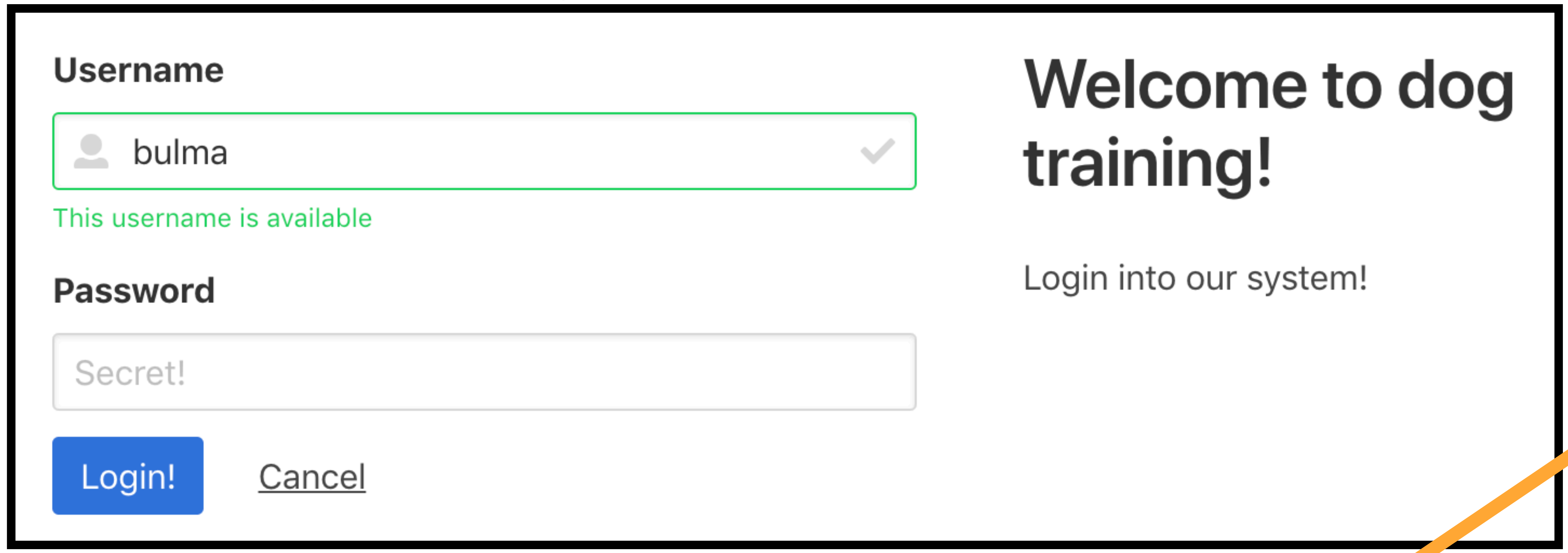
- Database check table users
  - users.email == $data.email
  - If user with email exists **return "Email already in system"**
- Add new row in table users
  - Add row (username=$data.username, email=$data.email, password=$data.password)
  - **Return "User added!"**



	A	B	C
1	**username**	**email**	**password**
2	bulma	hello@	someSecret
3			
4			

REAL WORLD BACKEND EXAMPLE

Visit dogtraining.com/login.html



Username

bulma

This username is available

Password

Secret!

Login! Cancel

Welcome to dog training!

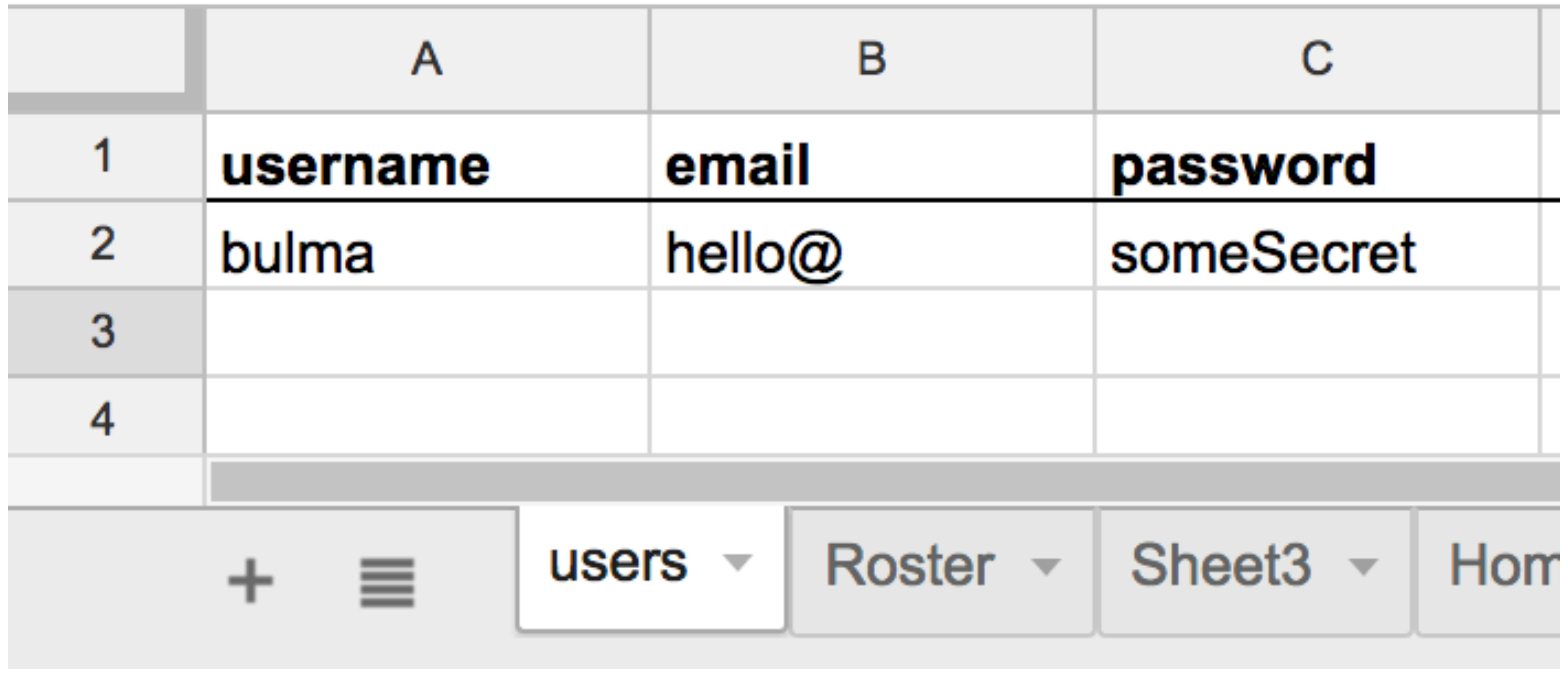
Login into our system!

Submit form which goes to /login.php

We pass data:
Username = bulma
Password = someSecret

login.php

- Database get row with user
  - $user = users.username == $data.username
- If $user.password == $data.password
  - **Return "User logged in!"**
- Else
  - **Return "Wrong password!"**



	A	B	C
1	**username**	**email**	**password**
2	bulma	hello@	someSecret
3			
4			

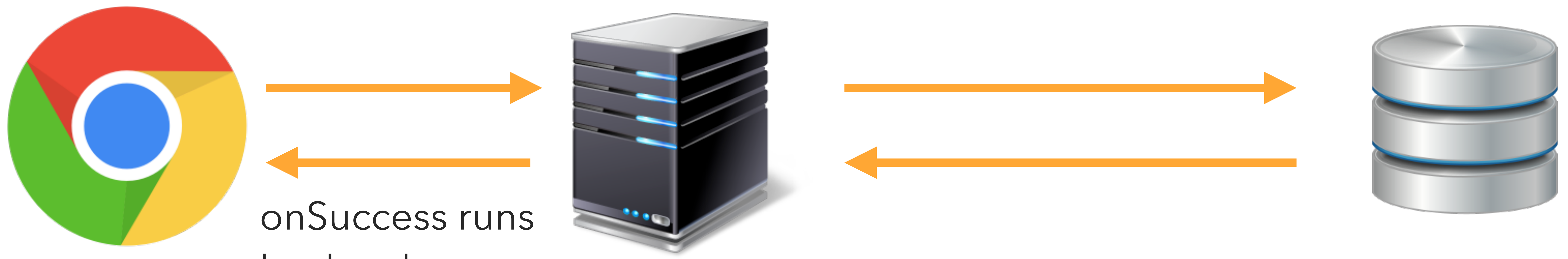
JAVASCRIPT - AJAX

How we interact with the server **without** reloading the page

mysite.com/login.html

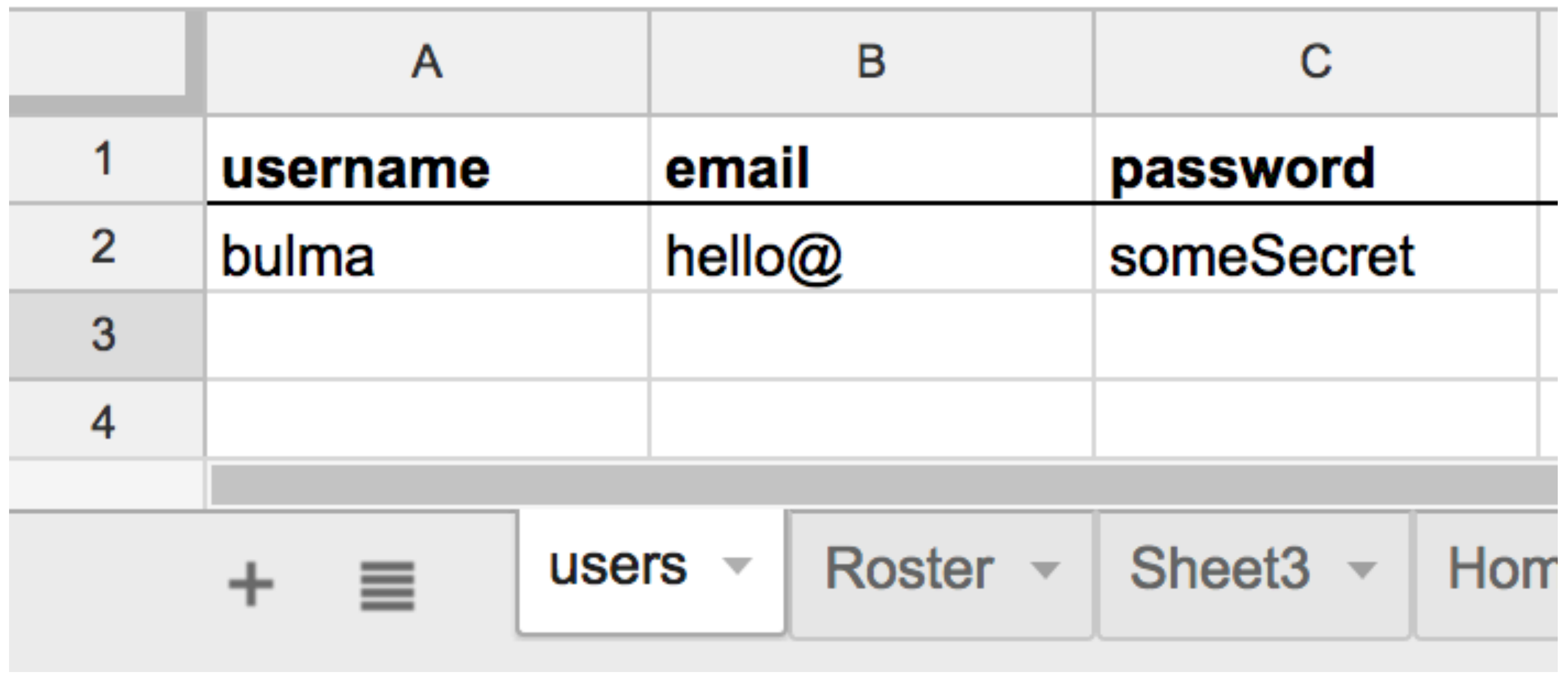
We use JavaScript feature AJAX to call login.php and pass user information to server

We run login.php on the server which calls our database to confirm credentials



onSuccess runs back to browser

```
$.ajax({
  url:"/login.php",
  data: [username,password].
  onSuccess: function(){
   alert("You're logged in"); }
 });
```



	A	B	C
1	**username**	**email**	**password**
2	bulma	hello@	someSecret
3			
4			

ORGANIZING YOUR PROJECTS

- Create one main folder for projects on your computer
  - Suggest to use **userFolder/projects**
- Create a new folder for each new project inside the main projects folder
  - **FinalProject**
  - **TestSite1**
- Inside your project folder create a folder for your stylesheets
  - **FinalProject/css/**
  - **FinalProject/css/style.css**

ORGANIZING YOUR PROJECTS

- Create an index.html file
- Include your stylesheet with this syntax

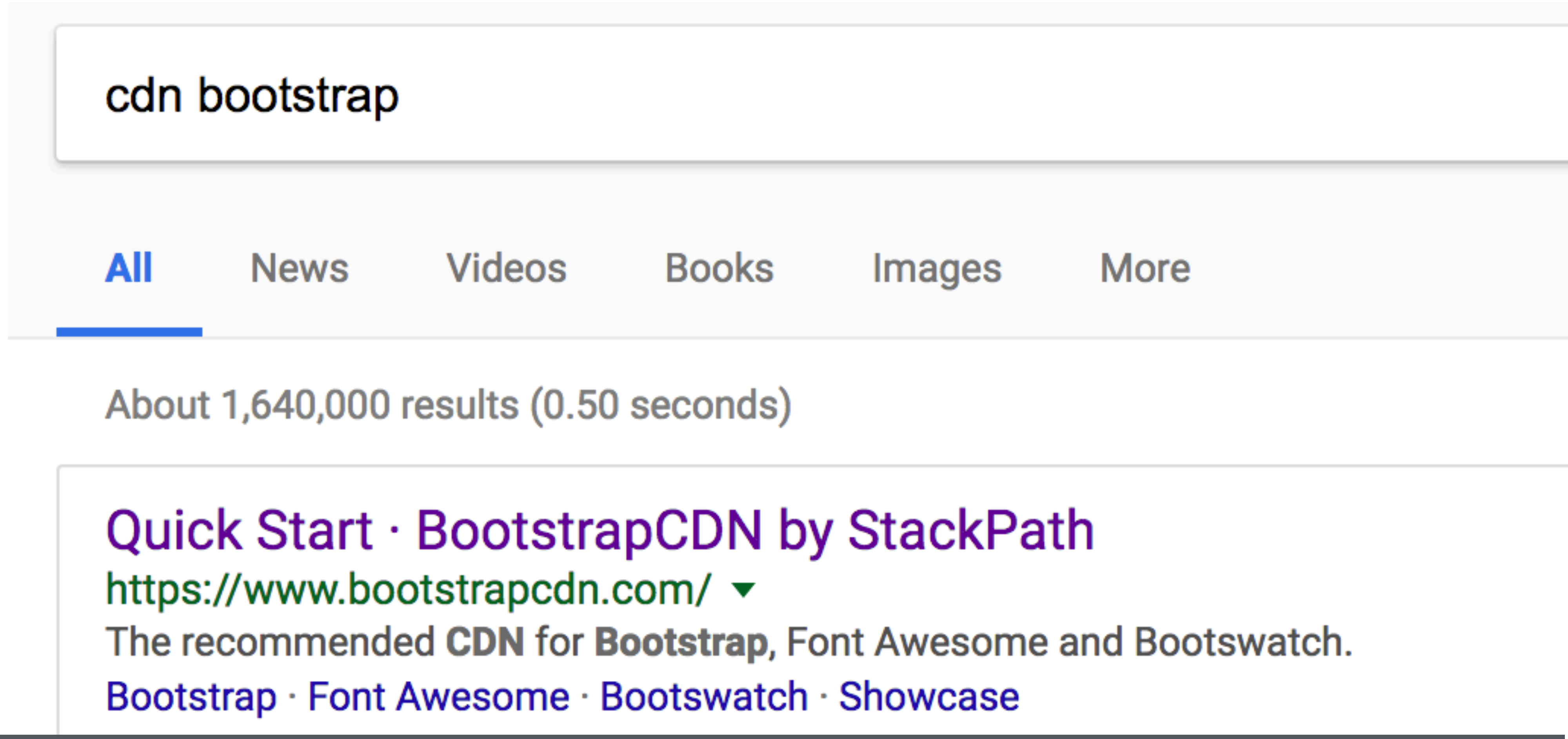
```
<head>
  <title>Welcome!</title>
  <link rel="stylesheet" href="css/style.css">
</head>
```

Tell the browser what type of link it is, in this case a stylesheet

Specify a location to your stylesheet

ADDING DEPENDENCIES WITH CDN

- Let’s add bootstrap to our project
- Quick find a cdn url for bootstrap by googling “bootstrap cdn”
- Include the cdn link from bootstrap like we did our main style



cdn bootstrap

All News Videos Books Images More

About 1,640,000 results (0.50 seconds)

Quick Start · BootstrapCDN by StackPath
https://www.bootstrapcdn.com/
The recommended **CDN** for **Bootstrap**, Font Awesome and Bootswatch.
Bootstrap · Font Awesome · Bootswatch · Showcase

```
<head>
  <title>Welcome!</title>
  <link rel=“stylesheet” href=“css/style.css”>
  <link rel=“stylesheet” href=“https://stackpath.bootstrapcdn.com/
  bootstrap/4.1.1/css/bootstrap.min.css”>
</head>
```

LETS BUILD A WEBSITE USING BOOTSTRAP

- Create a project following the steps I outlined
- Utilize the needed components from Bootstrap to build a full website
- Customize/extend bootstrap to fit your style!
  - Add custom font
  - Change colors
  - And more
- In class exercise - start building your final project using Bootstrap or Bulma!