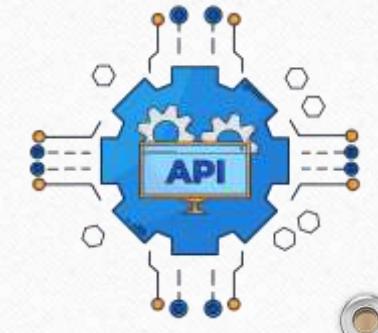
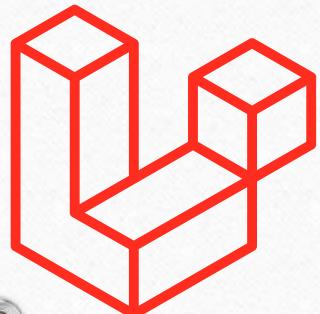


# Quick Recap of Strings + Introduction to MySQL Basics



## Day 7: Introduction to Full stack track with php

Eng. Ahmed Mohamed Abu-Bakr

Full stack web developer Laravel-react



FRONT END



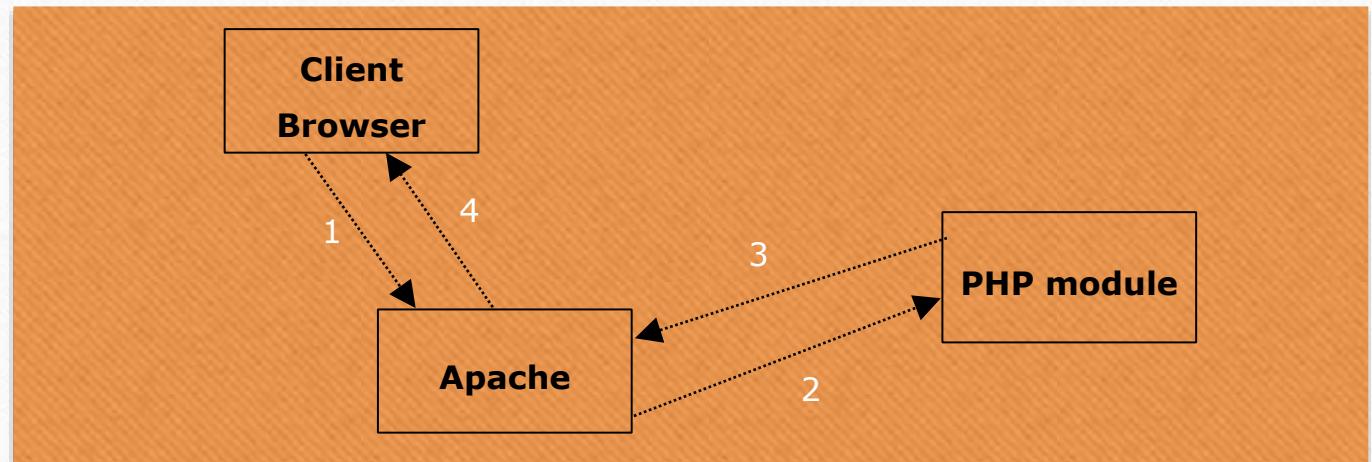
BACK END



comic.browserling.com

# Day 7: How Client Can open PHP File

```
C:\Users\CS\Desktop\test>php -S 127.0.0.1:5500
[Sun Jul 13 07:50:09 2025] PHP 7.4.29 Development Server (http://127.0.0.1:5500) started
[Sun Jul 13 07:51:07 2025] 127.0.0.1:64680 Accepted
[Sun Jul 13 07:51:07 2025] 127.0.0.1:64681 Accepted
[Sun Jul 13 07:51:07 2025] 127.0.0.1:64680 [200]: GET /
[Sun Jul 13 07:51:07 2025] 127.0.0.1:64680 Closing
[Sun Jul 13 07:51:25 2025] 127.0.0.1:64681 [200]: GET /
```



A screenshot of a web browser window showing the URL `127.0.0.1:5500`. The page content displays "Hello World!". Below the browser is a developer tools interface, specifically the "Elements" tab of the Chrome DevTools. The DOM tree shows the following structure:

```
<html>
  <head> ...
  <body> Hello World! </body>
</html>
```

The "Elements" tab also shows the selected element is the `body` tag.

# Day 7: Local Development Environment



XAMPP Control Panel v3.3.0 [ Compiled: Apr 6th 2021 ]

## XAMPP Control Panel v3.3.0

Service	Module	PID(s)	Port(s)	Actions
	Apache			<button>Start</button> <button>Admin</button> <button>Config</button> <button>Logs</button>
	MySQL			<button>Start</button> <button>Admin</button> <button>Config</button> <button>Logs</button>
	FileZilla			<button>Start</button> <button>Admin</button> <button>Config</button> <button>Logs</button>
	Mercury			<button>Start</button> <button>Admin</button> <button>Config</button> <button>Logs</button>
	Tomcat			<button>Start</button> <button>Admin</button> <button>Config</button> <button>Logs</button>

Config Netstat Shell Explorer Services Help Quit

```
2:13:18 PM [main] Initializing Control Panel
2:13:18 PM [main] Windows Version: Enterprise 64-bit
2:13:18 PM [main] XAMPP Version: 7.4.29
2:13:18 PM [main] Control Panel Version: 3.3.0 [ Compiled: Apr 6th 2021 ]
2:13:18 PM [main] You are not running with administrator rights! This will work for
most application stuff but whenever you do something with services
there will be a security dialogue or things will break! So think
about running this application with administrator rights!
2:13:18 PM [main] XAMPP Installation Directory: "c:\xampp\"
2:13:18 PM [main] Checking for prerequisites
2:13:19 PM [main] All prerequisites found
2:13:19 PM [main] Initializing Modules
2:13:19 PM [main] Starting Check-Timer
2:13:19 PM [main] Control Panel Ready
```

حالياً الجهاز يقدر يتعامل مع لغة  
**PHP** بس اين نضع ملفات  
الموقع

**Open C:\  
And watch what  
is inside it**



## Day 7: String Built-in Functions Recap



`trim($text)` .1

يحذف المسافات من البداية والنهاية

`strtoupper() / strtolower() / ucfirst()` .2

تحويل النص إلى شكل معين (كامل كبير - كامل صغير - أول حرف كبير)

`substr()` .3

تحديد جزء معين من النص.

`str_replace()` .4

استبدال كلمة بأخرى داخل النص.

```
<?php  
$text = " welcome to php world " ;  
echo strtoupper(trim($text)); // WELCOME TO PHP WORLD
```



## Day 7: Session Use Case

- ✓ الشرح:
- ✓ الخطوات الأساسية:
- ✓ المستخدم يملأ النموذج
- ✓ يتم التحقق من البيانات
- ✓ إذا كانت صحيحة → نحفظ الاسم في SESSION\_ \$
- ✓ يتم نقله إلى صفحة الترحيب

```
<?php
session_start();
if ($_POST['password'] === '123') {
    $_SESSION['user'] = $_POST['username'];
    header("Location: welcome.php");
}
```

isset(\$var) بترجم  
في حالة المتغير Boolean  
مش مترعرف أساسا ببقي false

```
<?php if (isset($_SESSION['error'])): ?>
    <div class="alert alert-danger alert-dismissible fade show" role="alert">
        <?php echo $_SESSION['error']; ?>
        <button type="button" class="btn-close" data-bs-dismiss="alert"></button>
    </div>
    <?php unset($_SESSION['error']); ?>
<?php endif; ?>
```

## Day 7: SESSION



- ✓ تستخدم لتخزين بيانات المستخدم مؤقتاً (مثلاً: حالة تسجيل الدخول)
- ✓ يتم حفظها في السيرفر
- ✓ تحتاج إلى session\_start() في أول كل ملف

```
session_start();
$_SESSION['username'] = 'Ahmed';
$_SESSION['useremail'] = 'ahmed@g.c';
var_dump($_SESSION);
```

← → ⌂ Not secure 192.168.235.115/PHP\_training/Day1/PHP\_Simple/supergoals.php

```
array(2) { ["username"]=> string(5) "Ahmed" ["useremail"]=> string(9) "ahmed@g.c" }
```

← → ⌂ Not secure 192.168.235.115/PHP\_training/Day1/PHP\_Simple/supergoals.php

Ahmed

```
19 echo $_SESSION['username'];
20
```



## Day 7: User Sessions and Visit Counter



الشرح: الجلسات (Sessions) بتساعدنا نخزن بيانات المستخدم طول فترة تصفحه للموقع.  
Task: كل ما المستخدم يعمل Refresh ، زوّد عداد الزيارات، واظهره داخل Badge من Bootstrap.

```
<?php
session_start();
$_SESSION['visits'] = ($_SESSION['visits'] ?? 0) + 1;
?>
<p class="m-3">Page Visits: <span class="badge bg-primary"><?= $_SESSION['visits'] ?></span></p>
```

ⓘ localhost/PHP\_training/Day5/PHP\_Session\_Examples/user\_count.php

Page Visits: 7

# Day 7: Task 1

## Task 1

اعمل Form فيه الاسم والإيميل، وخرزنهما في Session. استخدم تنسيق Bootstrap.

ahmed mohamed

john.doe@student.com

Save

clear Session

remove last

No users!

**user name**

**user email**

Name

Email

Save

clear Session

remove last

**user name**

ahmed mohamed

ahmed mohamed

**user email**

john.doe@student.com

abubakr3800@g.c



# Day 7: PHP File System (From Scratch)

## Overview of Common PHP File System Functions

قبل ما نبدأ في التاسكات العملية، تعالوا نستعرض أهم دوال نظام الملفات (File System)

Function	Description	Function	Description
fopen()	فتح ملف للقراءة أو الكتابة	file_exists()	التحقق من وجود الملف أو المجلد
fwrite()	كتابه نص داخل ملف مفتوح بـ ()	unlink()	حذف ملف
fclose()	إغلاق الملف بعد الانتهاء من الكتابة	glob()	قراءة كل الملفات المطابقة لنمط معين داخل مجلد
file_put_contents()	كتابة نص كامل داخل ملف (طريقة مختصرة)	basename()	استخراج اسم الملف من المسار الكامل
file_get_contents()	قراءة محتوى ملف كنص واحد	pathinfo()	استخراج معلومات عن الملف (الامتداد، الاسم...)
file()	قراءة ملف كسطر لكل عنصر في array	filesize()	الحصول على حجم الملف بالبايت
mkdir()	إنشاء مجلد جديد	move_uploaded_file()	نقل الملف المرفوع من مكانه المؤقت إلى مكان دائم
is_dir()	التحقق هل المسار مجلد	fputcsv()	كتابة صف من البيانات داخل ملف CSV



# Day 7: PHP File System (From Scratch)



## Examples

```
// logs/visits.txt  
  
Visit restored at 2025-07-17 18:30:00
```

```
<?php  
// تسجيل زيارة المستخدم إلى الموقع في ملف نصي  
$visitLog = fopen("logs/visits.txt", "a"); // فتح الملف بنمط الإضافة  
fwrite($visitLog, "New Visit at " . date("Y-m-d H:i:s") . "\n"); // كتابة التاريخ  
fclose($visitLog); // إغلاق الملف
```

```
ex > logs >  logs/visits.txt  
1 Visit restored at 2025-07-17 18:30:00  
2 New Visit at 2025-07-19 01:27:00  
3 New Visit at 2025-07-19 01:47:04  
4 New Visit at 2025-07-19 01:47:04  
5 New Visit at 2025-07-19 01:47:07  
6 New Visit at 2025-07-19 01:47:07  
7
```



# Day 7: PHP File System (From Scratch)



## Examples

```
// logs/visits.txt

Visit restored at 2025-07-17 18:30:00
New Visit at 2025-07-19 01:27:00
New Visit at 2025-07-19 01:47:04
New Visit at 2025-07-19 01:47:04
New Visit at 2025-07-19 01:47:07
New Visit at 2025-07-19 01:47:07
```

```
<?php
// قراءة محتوى ملف الزوارات كسطور مفصولة
$entries = file("logs/visits.txt");
echo "<pre>";
var_dump($entries);
echo "</pre>";
foreach ($entries as $index => $line) {
    echo "Visit #".($index + 1).":\n";
    $line<br>";
```

← → ⌂ ⓘ localhost/PHP\_training/Day6/ex/file-ex2.php

```
array(6) {
    [0]=>
    string(39) "Visit restored at 2025-07-17 18:30:00
"
    [1]=>
    string(34) "New Visit at 2025-07-19 01:27:00
"
    [2]=>
    string(34) "New Visit at 2025-07-19 01:47:04
"
    [3]=>
    string(34) "New Visit at 2025-07-19 01:47:04
"
    [4]=>
    string(34) "New Visit at 2025-07-19 01:47:07
"
    [5]=>
    string(34) "New Visit at 2025-07-19 01:47:07
"
}
```

```
Visit #1: Visit restored at 2025-07-17 18:30:00
Visit #2: New Visit at 2025-07-19 01:27:00
Visit #3: New Visit at 2025-07-19 01:47:04
Visit #4: New Visit at 2025-07-19 01:47:04
Visit #5: New Visit at 2025-07-19 01:47:07
Visit #6: New Visit at 2025-07-19 01:47:07
```

# Day 7: PHP Sourcing Files

## Include & Require in PHP

تُستخدم لاستدعاء ملفات PHP خارجية داخل ملف حالي لإعادة استخدام الكود (modularity)

`include "file.php"; .1` 

تحاول تضمين الملف.

لو الملف مش موجود: يظهر Warning، ويُكمل تنفيذ باقي السكريبت.

`require "file.php"; .2` 

تجبر على وجود الملف.

لو الملف غير موجود: يظهر Fatal Error ويتوقف السكريبت.

`require_once` و `include_once` .3 

الفرق: تمنع تضمين نفس الملف مرتين حتى لو تم استدعاؤه أكثر من مرة.



## Day 7: PHP Sourcing Files



### Include & Require in PHP

تُستخدم لاستدعاء ملفات خارجية داخل ملف حالي لإعادة استخدام الكود (modularity)



#### Common Use Cases

النوع	الوظيفة
header.php	Navigation Bar أو HTML
footer.php	نهاية الصفحة
db.php	إعداد اتصال قاعدة البيانات
config.php	إعدادات ثابتة مثل اسم الموقع أو التشفير
functions.php	مجموعة دوال عامة

أفضل الممارسات:  
استعمل `require_once` مع ملفات أساسية مثل الاتصال بقاعدة البيانات.

استعمل `include` مع ملفات غير حيوية (footer). لا تكرر إدراج نفس الملف.  
تأكد أن المسارات صحيحة (relative or absolute).



## Day 7: PHP Sourcing Files

### Include & Require in PHP

مثال عملي (مشروع صغير):  
ملفات المشروع:

```
project/
  └── index.php
  └── header.php
  └── footer.php
  └── config.php
```

```
<?php
    require_once "config.php";
    include "header.php";
    include "footer.php";
?>
```

```
<!DOCTYPE html>
<html>
<head><title>My App</title></head>
<body>
<nav class="navbar">Welcome</nav>
```

```
<footer>© 2025 My App</footer>
</body>
</html>
```



## Day 7: PHP Sourcing Files

### Magic Constants in PHP

تُستخدم لاستدعاء ملفات خارجية داخل ملف حالي لإعادة استخدام الكود (modularity)

Magic Constant	Description
<code>__LINE__</code>	رقم السطر الحالي في الملف
<code>__FILE__</code>	المسار الكامل للملف الحالي
<code>__DIR__</code>	اسم مجلد الملف الحالي
<code>__FUNCTION__</code>	اسم الدالة الحالية

ملاحظات مهمة:

كلهم بيترجعوا ك `string` مفيدين جداً في `logging`, `debugging`, `__FILE__`, `__DIR__` من أكثرهم استخداماً



## Day 7: 🧠 Why Databases? Why Not Excel?



الناس زمان كانت بتسجل بيانات في ملفات Excel.  
طيب لو عندي مئات المستخدمين؟

صعب أبحث - مفيش علاقات - في مشاكل بالتكرار.

قواعد البيانات = هيكل منظم لتخزين البيانات + سرعة + علاقات + حماية.

مثال سريع:

Excel = جدول واحد.

Database = عدة جداول بينهم علاقات (مثلاً: طلاب، مواد، درجات).

## Day 7: 💡 Types of Databases

النوع	الوصف	أمثلة
Relational (SQL)	الجداول مترابطة بمفاتيح	MySQL, PostgreSQL, SQL Server
Non-Relational (NoSQL)	بيانات غير منظمة، مرنة جدًا	MongoDB, Firebase

نقطة مهمة:

SQL = Structured Query Language (لغة للاستعلام المنظم).

NoSQL = Not Only Structured Query Language سريع، مرن، مناسب للبيانات الضخمة والموزعة.



## Day 7: 🧠 MySQL in Web Development



أكتر قاعدة بيانات مشهورة مع  PHP.

سهلة - مجانية - مدرومة في كل سيرفر.

من أشهر المشاريع اللي بتستخدم MySQL: WordPress Facebook Laravel with MySQL-:

## Day 7: 🧠 Where Databases Are Used

تسجيل دخول

تخزين منتجات في متجر إلكتروني

تتبع الطلاب في منصة تعليمية

رفع صور + وصفها

تتبع الطلبات والشحن



## Day 7: 🧠 Database Concepts



قاعدة بيانات: هي مجموعة من الجداول.

جدول: يشبه صفحة Excel — فيه أعمدة (Columns) وصفوف (Rows)

### ✓ Column Types:

### (indexing)

النوع	الاستخدام	ملاحظات
INT	رقم صحيح	أرقام فقط
VARCHAR	نص بطول محدد سريع + يفضل للحجم الصغير	طيء نسبياً لكن غير محدود الطول
TEXT	نص طويل	لحفظ التاريخ
DATE	تاريخ	

### ✓ Primary Key

هو العمود الأساسي الذي يميز كل صف لازم يكون Unique و Not Null مثل: Primary Key هو id في جدول الطلاب.

## Day 7: 🧠 Table Relationships

قاعدة بيانات: هي مجموعة من الجداول.

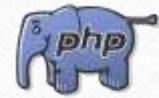
جدول: يشبه صفحة Excel فيه أعمدة (Columns) وصفوف (Rows)

العلاقة	الوصف	مثال
One to One	صف يرتبط بصف واحد فقط	كل موظف له ملف شخصي واحد فقط
One to Many	صف مرتبط بعده صفات أخرى	كل مستخدم له عدة طلبات
Many to Many	يتيم من خلال جدول وسيط	طالب يدرس عدة مواد، والمادة يدرسها عدة طلاب

# Day 7: 🧠 Table Relationships

أنواع الأعمدة (Columns) في MySQL وأفضل استخدام لكل نوع

النوع	مثال مدخل	استخدامه	ملاحظات
INT	1, 20, 1500	أرقام صحيحة (ID, العمر...)	سهل البحث والترتيب
VARCHAR(n)	"Ali", "Ahmed123"	نص قصير (اسم، بريد...)	أسرع من TEXT لكن محدود بطول n
TEXT	فقرات طويلة	وصف، محتوى مقال، تعليق طويل	لا يمكن فهرسته بسهولة – أبطأ
DATE	"2025-07-21"	تاریخ (تاريخ ميلاد، تسجيل)	يدعم عمليات مقارنة وتصفية
DATETIME	"2025-07-21 10:30:00"	تاریخ مع وقت	مناسب للطوابع الزمنية
BOOLEAN	0 أو 1	true/false (نشط؟ مدير؟...)	سهل التحقق منه
FLOAT / DECIMAL	99.99, 5.25	أرقام عشرية (سعر، نسبة...)	أدق للفلوس DECIMAL



# Day 7: 🎈 Table Relationships



لیه بنسخدم DATE بدل ما نخزن التاریخ کنص ؟(TEXT

لأنك تقدر تعمل على الأعمدة من نوع DATE العمليات التالية: ✓

العملية	تعمل على DATE	تعمل على TEXT
مقارنة تاريخين	<input checked="" type="checkbox"/>	<input type="checkbox"/>
ترتيب حسب التاريخ	<input checked="" type="checkbox"/>	<input type="checkbox"/>
استخراج الشهر / السنة باستخدام دوال	<input checked="" type="checkbox"/>	لازم parsing يدوي
البحث بالتاريخ	<input checked="" type="checkbox"/>	أبطأ و معقد

لو خزنت التاريخ كنص:

- ✓ تحتاج تتأكد من التنسيق يدوياً.
  - ✓ الترتيب هيكون بالحروف مش بالوقت.
  - ✓ فقدت قوة قاعدة البيانات في تحليل التواريخ



## Day 7: 🧠 Table Relationships

ماذا لو خزنت كل البيانات كنص TEXT؟

السرعة: TEXT أبطأ لأن النظام لا يستطيع استخدامه للفهرسة بكفاءة.

الحجم: TEXT يشغل مساحة أكبر على القرص.

فقدان المعنى: لو سجلت العمر كنص، مش هتقدر تقول "هات اللي عمرهم < 20".

مشاكل التحقق من البيانات: ممكن المستخدم يدخل "ahmed" بدل رقم أو تاريخ بدون ما تفهم القاعدة إنها مشكلة.

القاعدة الذهبية:

استخدم كل نوع في مكانه الصحيح، عشان الأداء، والوضوح، والتكامل.



## Day 7: 🧠 Data Base Design



إزاي أبدأ تصميم قاعدة بيانات لتطبيق معين؟

مثال: تطبيق "نظام طلاب"

١. حدد (Entities) العناصر الرئيسية اللي هتعامل معاها وه يكون لها دور في

السيستم

□ طلاب (Students)

□ كورسات (Courses)

□ درجات (Grades)



## Day 7: 🧠 Data Base Design

إزاي أبدأ تصميم قاعدة بيانات لتطبيق معين؟

مثال: تطبيق "نظام طلاب"

2. كل Entity : حدد الأعمدة الأساسية

### 📌 Students

	الاسم	النوع
id	INT + PRIMARY KEY + AUTO_INCREMENT	
name	VARCHAR(100)	
email	VARCHAR(150)	
birth_date	DATE	

### 📌 Courses

	الاسم	النوع
id	INT + PRIMARY KEY	
title	VARCHAR(100)	
hours	INT	

### 📌 Grades

	الاسم	النوع
student_id	INT (FOREIGN KEY to students.id)	
course_id	INT (FOREIGN KEY to courses.id)	
grade	FLOAT	



## Day 7: 🧠 Data Base Design

إزاي أبدأ تصميم قاعدة بيانات لتطبيق معين؟

مثال: تطبيق "نظام طلاب"

3. حدد العلاقات:

طالب ← له أكثر من كورس (Many to Many)  
نعمل جدول وسيط grades (intermediate table)

خلاصة

- لا نستخدم TEXT إلا لما يكون المحتوى فعلاً طويلاً زي وصف منتج.
- استخدم INT, DATE, BOOLEAN, VARCHAR حسب نوع الداتا.
- تصميم قواعد البيانات يبدأ بتحليل الكيانات وال العلاقات. MySQL قوية لأنها بتفهم نوع الداتا و بتساعدك تبني أنظمة قوية وآمنة.

# Day 7: 🧠 Data Base Design

## (Database Design Process)

رقم	الخطوة	الوصف	مثال عملي
1	تحليل المتطلبات Requirements Analysis	فهم النظام المطلوب واحتياجات المستخدم.	"نريد نظام لتسجيل الطلاب في كورسات"
2	تحديد الكيانات Identify Entities	تحديد الكائنات الأساسية (مثل: طالب، كورس).	Student, Course
3	تحديد الخصائص Define Attributes	تحديد خصائص كل كيان.	Student لديه name, email, birthdate
4	تحديد العلاقات Define Relationships	توضيح كيف ترتبط الكيانات بعضها.	Many-to-Many علاقة Student ← Course ↗ مستطيل = كيان، خط = علاقه، بيضاوي = خاصية
5	رسم ERD (Entity Relationship Diagram)	رسم تخطيطي للكيانات والعلاقات بينها.	student.id = PK, enrollment.student_id = FK
6	تحديد المفاتيح Choose Primary & Foreign Keys	اختيار مفاتيح فريدة وأجنبية للعلاقات.	فصل بيانات الكورسات عن الطلاب، إنشاء جدول enrollments
7	تطبيق التطبيع Apply Normalization	إزالة التكرار وتحسين التصميم.	students (id INT, name VARCHAR(100), email VARCHAR(150), ...)
8	إنشاء المخطط المنطقي Logical Schema	كتابة تعريف الجداول (columns + datatypes).	
9	تحويله إلى قاعدة فعلية Create DB Physically	تنفيذ المخطط باستخدام phpMyAdmin أو SQL.	MySQL في CREATE TABLE students (...)
10	اختبار النموذج Test & Validate	إدخال بيانات تجريبية واختبار العلاقات والاستعلامات.	إدخال طالب وكورس، التحقق من التسجيل، تجربة SELECT, JOIN



## Day 7: 🧠 Data Base Design



### ✿ ملاحظات مهمة:

الـ ERD مش مجرد رسم، ده بيساعدك تشواف الأخطاء قبل ما تكتب كود.

لو تخطيت خطوة التطبيع، ممكن يحصل تكرار كبير في البيانات، يؤدي لمشاكل لاحقاً.

Primary Key = الهوية،

Foreign Key = الربط.



## Day 7: 🧠 What is Normalization



Normalization هو عملية تنظيم البيانات داخل قاعدة البيانات بطريقة تقلل التكرار وتمنع التعارض (inconsistency).

### ✓ Why Normalize?

تقليل استهلاك المساحة.

منع التكرار المزعج (redundancy)

سهولة التعديل والتحديث.

تحسين أداء الاستعلامات.



### Normalization Levels (Up to 3NF)

#### ◆ 1NF – First Normal Form

"كل خلية تحتوي على قيمة واحدة فقط"

#### ◆ 2NF – Second Normal Form

"كل عمود يعتمد فقط على الـ Primary Key"

#### ◆ 3NF – Third Normal Form

"ما فيش عمود بيعتمد على عمود غير المفتاح الأساسي"

## Day 7: 🌟 Before DB creation

### 💡 Lt's Create a Database Design Flow

الخطوة 1: فهم النظام – أسأل نفسك

"أنا ببني نظام بيعمل إيه؟"



مثال عملي:-

- ✓ منصة كورسات أونلайн
- نظام بيسمح:-
  - تسجيل الطلاب
  - عرض كورسات
  - كل طالب ممكن يسجل في أكثر من كورس
  - كل كورس ممكن يكون فيه أكثر من طالب
  - ممكن أحتج أسجل درجات أو تقييم



## Day 7: 🐛 Before DB creation



💡 Let's Create a Database Design Flow

الخطوة 2: 🎯

تحديد الكيانات الأساسية (Entities)

من الفهم السابق أستنتج:

>User

Course

(العلاقة اللي بينهم)

### Student

- id (PK)
- Name
- Email
- Date-of-birth
- phone

### Course

- id (PK)
- Title
- Description
- Hours
- price

### Enrollment

- student\_id (FK → Student)
- course\_id (FK → Course)
- Grade
- enrollment\_date



# Day 7: 🌟 Before DB creation



## 💡 Let's Choose Data Type of each columns

الجدول	العمود	النوع المناسب	السبب
students	id	INT (PK)	مفتاح أساسى للتعريف
	name	VARCHAR(100)	نص قصير
	email	VARCHAR(150)	بريد - محدود الحجم
	phone	VARCHAR(20)	أرقام قد تبدأ بصفر أو علامة +
	date_of_birth	DATE	علشان نعمل حساب عمر
courses	id	INT (PK)	مفتاح أساسى
	title	VARCHAR(100)	عنوان بسيط
	description	TEXT	شرح طويل
	hours	DECIMAL(5,2)	عدد ساعات
	price	DECIMAL(8,2)	رقم عشرى للفلوس
enrollments	id	INT (PK)	مفتاح
	student_id	INT (FK)	ربط بالطلاب
	course_id	INT (FK)	ربط بالكورسات
	grade	FLOAT	ممكن تكون 95.5
	enrollment_date	DATETIME	وقت التسجيل بدقة



## Day 7: 🌐 Before DB creation



### 💡 Let's Normalize Data

#### 1NF:

كل خلية تحتوي على قيمة واحدة فقط ✓

ما فيش كورسات داخل خلية واحدة – كل كورس صف مستقل.

#### 2NF:

كل البيانات تعتمد فقط على الـ PK ✓

مفيش email داخل enrollments – هي موجودة فقط عند الطلاب.

#### 3NF:

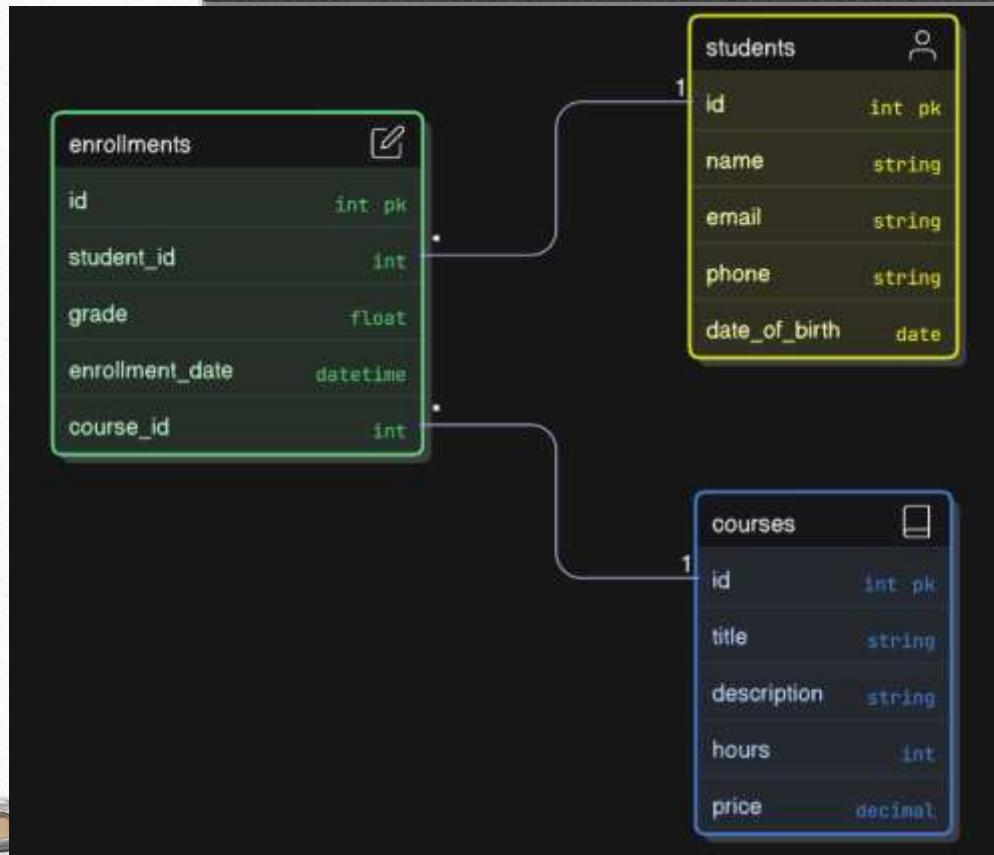
الأعمدة لا تعتمد على أعمدة أخرى غير الـ PK ✓

السعر والشرح موجودين في جدول الكورسات، مش في enrollments.



# Day 7: 🐛 Before DB creation

✳️ Let's create ERD



الخطوة 7: ✓

تحويل إلى قاعدة بيانات فعلية

نروح على phpMyAdmin ⬇️

نبدأ بإنشاء قاعدة البيانات training\_system

ثم ننفذ الجداول السابقة يدوياً أو بـ SQL

✳️ ملخص الأسئلة التي أسلّها قبل تصميم أي قاعدة بيانات:

إيه الكيانات في النظام ده؟ كل كيان فيه إيه؟ (صفاته) إيه العلاقات بينهم؟ إيه

نوع كل عمود؟ هل في أي تكرار أقدر أخلص منه؟ إزاي أرسم العلاقة

(ERD)؟ إزاي أحول دا إلى SQL وأبدأ تنفيذ فعلي؟

# Day 7: 🧠 phpMyAdmin

💡 Let's Create a database in phpMyAdmin

🎯 السيناريو: نظام تسجيل كورسات للطلاب

1. فتح phpMyAdmin:  
اكتب في المتصفح: <http://localhost/phpMyAdmin>

2. إنشاء قاعدة بيانات جديدة   
اضغط "New"  
اسم القاعدة: training\_system  
Charset: utf8mb4\_general\_ci  
اضغط "Create"



## Day 7: 🧠 phpMyAdmin



💡 Let's Create a database in phpMyAdmin

SQL = Structured Query Language  
هي اللغة المستخدمة للتعامل مع قواعد البيانات العلائقية (مثل MySQL – PostgreSQL)

💡 **تصنيفات أوامر SQL الأساسية:**

التصنيف	الوظيفة الأساسية	أشهر الأوامر
❖ DML	Data Manipulation Language	SELECT, INSERT, UPDATE, DELETE
❖ DDL	Data Definition Language	CREATE, ALTER, DROP, RENAME
❖ DCL	Data Control Language	GRANT, REVOKE
❖ TCL	Transaction Control Language	COMMIT, ROLLBACK, SAVEPOINT



## Day 7: 🧠 phpMyAdmin

💡 Let's Create a database in phpMyAdmin

SQL = Structured Query Language  
هي اللغة المستخدمة للتعامل مع قواعد البيانات العلائقية (مثل MySQL – PostgreSQL)

النوع

ال코드

الوصف

🔍 **SELECT** SELECT \* FROM students

استعلام عن كل البيانات

➕ **INSERT** INSERT INTO students (name, email) VALUES ('Ali', 'ali@mail.com')

إدخال بيانات جديدة

✎ **UPDATE** UPDATE students SET name = 'Omar' WHERE id = 1

تعديل بيانات

✖ **DELETE** DELETE FROM students WHERE id = 1

حذف بيانات



## Day 7: 🧠 phpMyAdmin

💡 Let's Create a database in phpMyAdmin

SQL = Structured Query Language  
هي اللغة المستخدمة للتعامل مع قواعد البيانات العلائقية (مثل MySQL – PostgreSQL)

الوصف	النوع	الكود
إنشاء جدول جديد	CREATE	CREATE TABLE students (...)
تعديل هيكل جدول	ALTER	ALTER TABLE students ADD phone VARCHAR(20)
حذف جدول بالكامل	DROP	DROP TABLE students
الربط بين جداولين	JOIN	SELECT * FROM students JOIN courses ON ...
شرط للتصفيية	WHERE	SELECT * FROM users WHERE email LIKE '%@gmail.com'
ترتيب النتائج	ORDER BY	SELECT * FROM courses ORDER BY price DESC
عرض عدد معين فقط	LIMIT	SELECT * FROM students LIMIT 5



# Day 7: 🧠 phpMyAdmin

## 💡 Let's Create a database in phpMyAdmin

```
CREATE DATABASE IF NOT EXISTS training_system
CHARACTER SET utf8mb4
COLLATE utf8mb4_general_ci;

USE training_system;

CREATE TABLE students (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(150) UNIQUE,
    phone VARCHAR(20),
    date_of_birth DATE
);

CREATE TABLE courses (
    id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(100) NOT NULL,
    description TEXT,
    hours DECIMAL(4,2),
    price DECIMAL(8,2)
);

CREATE TABLE enrollments (
    id INT AUTO_INCREMENT PRIMARY KEY,
    student_id INT NOT NULL,
    course_id INT NOT NULL,
    grade FLOAT DEFAULT NULL,
    enrollment_date DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (student_id) REFERENCES students(id) ON DELETE CASCADE,
    FOREIGN KEY (course_id) REFERENCES courses(id) ON DELETE CASCADE
);
```

```
USE training_system;

INSERT INTO students (name, email, phone, date_of_birth)
VALUES
    ('Ahmed Ali', 'ahmed@example.com', '01111111111', '2000-05-10'),
    ('Sara Youssef', 'sara@example.com', '01222222222', '2001-01-20');

INSERT INTO courses (title, description, hours, price)
VALUES
    ('Web Development', 'Intro to HTML, CSS, PHP', 40.00, 1500.00),
    ('MySQL Basics', 'Database fundamentals', 20.00, 1000.00);

INSERT INTO enrollments (student_id, course_id, grade)
VALUES
    (1, 1, 85.5),
    (2, 2, 90.0);
```



# Day 7: 🧠 phpMyAdmin



💡 Let's Connect PHP to MySQL + Print Student Table

```
<?php
$host = "localhost";
$user = "root";
$pass = "";
$dbname = "training_system";

// إنشاء الاتصال
$conn = new mysqli($host, $user, $pass, $dbname);

// التحقق من الاتصال
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
?>
```

```
<?php include "db.php";
$sql = "SELECT * FROM students";
$result = $conn->query($sql);

if ($result && $result->num_rows > 0):
    while ($row = $result->fetch_assoc()):
?>
    <tr>
        <td><?= $row['id'] ?></td> <td><?= $row['name'] ?></td> <td><?=
$row['email'] ?></td> <td><?= $row['phone'] ?></td> <td><?=
$row['date_of_birth'] ?></td>
    </tr>
<?php
    endwhile;
else:
    echo "<tr><td colspan='5'>No students found.</td></tr>";
endif;
?>
```

# Day 7: 🧠 phpMyAdmin

💡 Let's Connect PHP to MySQL + Print Student Table

الدالة

`mysqli_connect()`

`mysqli_query()`

`mysqli_fetch_assoc()`

`mysqli_num_rows()`

`mysqli_error()`

`mysqli_close()`

الوظيفة

الاتصال بقاعدة البيانات

تنفيذ استعلام

جلب النتائج كمصفوفة مفتاحية

عدد الصفوف المرتجعة

يعرض آخر خطأ

إغلاق الاتصال



## Day 7: Task



### Task

أنشئ نموذجاً لإضافة كورسات جديدة (العنوان إجباري، باقي الحقول اختيارية).

أضف فورم لتسجيل طالب في كورس باستخدام `student_id, course_id, grade`  
استخدام `INNER JOIN` بين `enrollments, students, courses` و

طباعة جدول فيه:

- اسم الطالب
- بريده الإلكتروني
- اسم الكورس
- الدرجة
- تاريخ التسجيل

أنشئ ملف `navigation.php` وأضفه إلى جميع الصفحات باستخدام `include`.

المطلوب :

توصيل ملف PHP بقاعدة البيانات

طباعة جدول الطلاب داخل Bootstrap table

إنشاء نموذج (form) يحتوي على:

- الاسم (مطلوب)
- البريد الإلكتروني (مطلوب)
- رقم الهاتف (اختياري)
- تاريخ الميلاد (مطلوب)
- التحقق من البيانات (HTML5 + PHP)
- إدخالها إلى قاعدة البيانات باستخدام `INSERT INTO`

## Day 7: Task 1

### Task 1

الصفحة

students.php

add\_student.php

add\_course.php

enroll\_student.php

enrollments\_list.php

الوظيفة

عرض جميع الطلاب

إضافة طالب جديد

إضافة كورس

تسجيل طالب في كورس

عرض كل الطلاب المسجلين بالكورسات

# The End Of Day 7



Feel free to contact me any time

[linktr.ee/ahmed\\_abubakr](https://linktr.ee/ahmed_abubakr)