ایده اصلی جلسه ۳۲

- قبلاً صفحه `show.php` فقط یادداشت رو نشون می‌داد.
- حالا یک دکمه "**حذف یادداشت** " اضافه می‌کنیم.
- وقتی کاربر روی این دکمه کلیک می‌کنه:
  ۱. یادداشت پیدا میشه (با بررسی اینکه مال کاربر هست یا نه)
  ۲. حذف میشه
  ۳. کاربر به لیست همه یادداشت‌ها (`/notes`) برمی‌گرده.

2.ساختار کلی show.php

```
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // حذف یادداشت
} else {
    // نمایش یادداشت
}
```

- شرط `$_SERVER["REQUEST_METHOD"] == "POST"` یعنی اگر درخواست حذف (از فرم) اومده.
- حالت `else` یعنی کاربر فقط خواسته یادداشت رو ببینه.

3.بخش حذف(POST)

```
    $db = new Database($config['database'], "root", "");

//پیدا کردن یادداشت برای اطمینان از وجود و مالکیت
$note = $db->query(
    "SELECT * FROM notes WHERE user_id = :user AND id =
:id",
    ["user" => 3, "id" => $_GET['id']]
)->findOrFail();
```

```
// بررسی مجوز (Authorize)
Authorize($note['user_id'] === 3);
```

//حذف یادداشت

```
$db->query(
    "DELETE FROM notes WHERE user_id = :user AND id =
:id",
    ["user" => 3, "id" => $_POST['id']]
)->fetch();
```

```
// هدایت به لیست یادداشت‌ها
header("Location: /laracast-php/public/notes");
exit();
```

نکات امنیتی:

- قبل از حذف، باید مطمئن بشیم یادداشت وجود داره (`findOrFail()` →
- باید مطمئن بشیم یادداشت مال کاربر فعلیه (`Authorize()` →
- حذف با `POST` انجام میشه نه `GET`، چون لینک حذف ناامن هست.

4.بخش نمایش(GET)

```
$db = new Database($config['database'], "root", "");
$note = $db->query(
    "SELECT * FROM notes WHERE user_id = :user AND id =
:id",
    ["user" => 3, "id" => $_GET['id']]
)->findOrFail();

$current_userid = 3;
Authorize($note['user_id'] === $current_userid);

view("note/show.view.php", ["note" => $note]);
```

- اینجا فقط داده رو می‌گیریم و توی ویو نمایش می‌دیم.
- مجوز مالکیت یادداشت رو چک می‌کنیم.

- `findOrFail()` تضمین می‌کنه که اگه یادداشت پیدا نشد، ۴۰۴ بده.

---

5. فرم حذف در `show.view.php`

```
<form method="POST">
    <input type="hidden" name="id" value="<?=
$note['id'] ?>">
    <button type="submit">حذف یادداشت</button>
</form>
```

چرا `hidden` داریم؟
چون نیاز داریم ID یادداشت در درخواست POST باشه تا بدونیم چی رو حذف کنیم.

---

6. فرق `findOrFail()` و `fetch()`

- `findOrFail()` → وقتی نیاز داری رکورد پیدا بشه و اگه پیدا نشد خطا بده (مناسب برای SELECT و نمایش).
- `fetch()` → وقتی query اجرا میشه ولی نتیجه‌ای برنگردونه یا نیاز به داده نداری (مثل DELETE).

---

7. ترتیب اجرای کد

۱. کاربر وارد `/notes/show?id=5` میشه → یادداشت نمایش داده میشه.
۲. کاربر روی دکمه حذف کلیک می‌کنه → فرم POST ارسال میشه.
۳. PHP دوباره `show.php` رو اجرا می‌کنه → شرط POST فعال میشه.
۴. یادداشت پیدا میشه → مجوز بررسی میشه → یادداشت حذف میشه.
۵. کاربر به `/notes` ریدایرکت میشه.

تمرین ۱

در کد زیر، چرا برای حذف یادداشت از `fetch()` استفاده شده و نه `findOrFail()`؟

پاسخ:
`fetch()` برای اجرای کوئری‌هایی استفاده می‌شود که نتیجه‌ی آن‌ها نیاز به گرفتن داده ندارد (مانند `DELETE` یا `UPDATE`).
`findOrFail()` مخصوص زمانی است که باید یک رکورد خاص را از دیتابیس دریافت کنیم و اگر وجود نداشت، خطا بدهد. در عملیات حذف، ما نیازی به دریافت رکورد بعد از حذف نداریم.

تمرین ۲

اگر در بخش حذف (`DELETE`) به جای `$_POST['id']` از `$_GET['id']` استفاده کنیم چه مشکلی ممکن است پیش بیاید؟

پاسخ:
این باعث می‌شود عملیات حذف از طریق آدرس URL هم قابل انجام باشد و امنیت کاهش پیدا کند. استفاده از `POST` برای عملیات حساس، جلوی حذف تصادفی یا حملات CSRF را بهتر می‌گیرد.

تمرین ۳

چرا قبل از حذف یادداشت، دوباره `SELECT` گرفته می‌شود؟

پاسخ:
برای اینکه بررسی کنیم یادداشت متعلق به کاربر جاری هست یا نه (`Authorize`). اگر بدون این بررسی حذف کنیم، هر کاربر می‌تواند یادداشت‌های بقیه را حذف کند.

تمرین ۴

اگر `Authorize($note['user_id'] === 3);` **حذف شود، چه خطری ایجاد می‌شود؟**

پاسخ:
هر کاربری می‌تواند با تغییر `id` **در آدرس، یادداشت‌های کاربران دیگر را ببیند یا حذف کند (دسترسی غیرمجاز).**

تمرین ۵

چرا بعد از `header("location:/laracast-php/public/notes");`**، باید** `exit();` **بگذاریم؟**

پاسخ:
برای جلوگیری از اجرای ادامه‌ی کد بعد از ریدایرکت. اگر `exit` **نباشد، اسکریپت ادامه پیدا می‌کند و ممکن است خطا یا پردازش ناخواسته انجام شود.**

تمرین ۶

در بخش نمایش(`GET`**) ، چرا از** `findOrFail()` **استفاده شده؟**

پاسخ:
چون نیاز داریم اطلاعات یادداشت را از دیتابیس بگیریم. اگر یادداشت وجود نداشت، `findOrFail` **باعث می‌شود اسکریپت متوقف شود و صفحه ۴۰۴ یا خطای مناسب نمایش داده شود.**

تمرین ۷

اگر بخواهیم کاربر بعد از حذف یادداشت به صفحه قبلی برگردد، باید چه کدی جایگزین `header()`کنیم؟

پاسخ:

```
header("Location: " . $_SERVER['HTTP_REFERER']);
exit();
```

این کاربر را به آدرس صفحه‌ای که از آن آمده برمی‌گرداند.

تمرین ۸

در این کد:

```
$db->query("DELETE FROM notes where user_id = :user and id=:id",["user"=>3,"id"=>$_POST['id']])->fetch();
```

اگر `fetch()` حذف شود، چه تغییری ایجاد می‌شود؟

پاسخ:
احتمالاً تغییری در عملکرد کلی حذف ایجاد نمی‌شود چون `DELETE` داده‌ای برنمی‌گرداند.
ولی ممکن است بعضی نسخه‌ها یا ساختار PDO نیاز به فراخوانی متد داشته باشند تا کوئری کامل اجرا شود. به جای `fetch()` می‌توان از `execute()` هم استفاده کرد.

تمرین ۹

اگر کاربر ID یادداشت را در `$_POST['id']` تغییر دهد و آن یادداشت متعلق به کاربر دیگری باشد، آیا کد فعلی جلوی حذف را می‌گیرد؟

پاسخ:
بله، چون قبل از حذف، ابتدا `SELECT`گرفته می‌شود و با `Authorize()`بررسی می‌کنیم که `user_id`برابر کاربر جاری باشد. اگر نباشد، حذف انجام نمی‌شود.

تمرین ۱۰

اگر بخواهیم حذف یادداشت به‌صورت Ajax انجام شود، چه تغییراتی باید دهیم؟

پاسخ:

۱. **دکمه حذف باید یک درخواست `fetch`یا `XMLHttpRequest`ارسال کند.**
۲. **در PHP ، به جای ریدایرکت(`header`) ، باید یک پاسخ JSON برگردانیم:**

```
echo json_encode(['status' => 'success']);
exit();
```

۳. **در سمتJavaScript ، بعد از موفقیت‌آمیز بودن حذف، یادداشت را از DOM حذف کنیم.**