

Laravel Route Model Binding

問題背景

傳統做法遇到的問題

在 Laravel 中，我們經常需要：

```
// ❌ 問題 1：每次都要手動查詢資料
Route::get('/tasks/{id}/edit', function ($id) {
    $task = Task::findOrFail($id); // 重複的查詢邏輯
    return view('edit', ['task' => $task]);
});
```


```
// ❌ 問題 2：驗證邏輯散落在各處
Route::post('/tasks', function (Request $request) {
    $data = $request->validate([ // 驗證邏輯重複
        'title' => 'required|max:255',
        'description' => 'required',
        'long_description' => 'required',
    ]);
});
```

Route Model Binding

什麼是 Route Model Binding ?

Laravel 會根據 URL 參數自動查詢對應的資料庫記錄。

基本用法

```
//  使用 Route Model Binding
Route::get('/tasks/{task}', function (Task $task) {
    return view('show', compact('task'));
});
```

工作原理

當訪問 `/tasks/1` 時：

Form Request

什麼是 Form Request ?

將驗證邏輯封裝在專門的類別中，讓控制器更簡潔。

建立 Form Request

```
php artisan make:request TaskRequest
```

定義驗證規則

```
// app/Http/Requests/TaskRequest.php  
class TaskRequest extends FormRequest  
{  
    public function authorize(): bool
```



validated() 方法

什麼是 validated() ?

只返回通過驗證的資料，過濾掉未驗證的欄位。

範例

```
// 假設表單提交了這些資料：  
[  
  'title' => '數學作業',  
  'description' => '完成第三章習題',  
  'long_description' => '詳細說明...',  
  'hack_attempt' => '惡意資料' // 未在驗證規則中定義  
]
```

```
// $request->validated() 只會返回：
```

Mass Assignment Protection

什麼是 Mass Assignment ?

Laravel 為了安全，預設不允許批量寫入資料庫。

錯誤範例

```
// 這樣會出錯：  
Task::create([  
    'title' => '作業',  
    'description' => '說明'  
]);  
// 錯誤：MassAssignmentException
```

解決方法

實際改造過程

步驟 1：建立 Form Request

```
php artisan make:request TaskRequest
```

步驟 2：設定驗證規則

```
// app/Http/Requests/TaskRequest.php
public function rules(): array
{
    return [
        'title' => 'required|max:255',
        'description' => 'required',
        'long_description' => 'required',
    ];
}
```



完整代碼對比

✗ 改造前

```
// 路由：需要手動查詢
Route::get('/tasks/{id}/edit', function ($id) {
    $task = Task::findOrFail($id);
    return view('edit', ['task' => $task]);
});

// 處理表單：需要手動驗證
Route::put('/tasks/{id}', function ($id, Request $request) {
    $data = $request->validate([
        'title' => 'required|max:255',
        'description' => 'required',
        'long_description' => 'required',
    ]);

    $task = Task::findOrFail($id);
    $task->update($data);
});
```


! 常見錯誤與解決

錯誤 1：Missing required parameter

錯誤訊息：

```
Missing required parameter for [Route: tasks.show] [URI: tasks/{task}] [Missing parameter: task].
```

原因：重定向時沒有正確傳遞參數

解決方法：

```
// ❌ 錯誤  
return redirect()->route('tasks.show', $task);
```

```
// ✅ 正確
```

小結

學到的技能

功能	改造前	改造後	好處
查詢資料	<code>Task::findOrFail(\$id)</code>	<code>Task \$task</code>	更簡潔、自動處理錯誤
驗證資料	在路由中手動驗證	<code>TaskRequest</code>	邏輯集中、可重用