**service container**

تعد حاوية الخدمة، والمعروفة أيضًا باسم حاوية IoC (انعكاس التحكم)، أداة قوية لإدارة تبعيات الفئة وتنفيذ حقن التبعيات. إنه جزء مهم من بنية Laravel ويوفر طريقة لربط الفئات في الحاوية وحلها عند الحاجة. يتم استخدام حاوية الخدمة لتحقيق تنظيم أفضل للتعليمات البرمجية وقابلية الصيانة وقابلية الاختبار.

**validation**

يعد التحقق من الصحة جانبًا مهمًا في معالجة النماذج ومعالجة البيانات. يوفر Laravel نظام تحقق شامل يجعل من السهل التحقق من صحة البيانات الواردة من مصادر مختلفة مثل طلبات HTTP أو النماذج أو استدعاءات API. عادةً ما يتم إجراء التحقق من الصحة في وحدة التحكم قبل معالجة البيانات.

**responses**

تُستخدم الاستجابات لإرسال المخرجات مرة أخرى إلى العميل، سواء كان متصفح ويب أو تطبيق جوال أو أي نوع آخر من العملاء. يوفر Laravel طريقة مريحة ومعبرة لإنشاء أنواع مختلفة من الاستجابات

**file upload with laravel**

1)Create a Form

2) Handle the Upload in the Controller

3) Configure File Storage

4) Display the Uploaded File

**Eloquent**

Eloquent هو نظام رسم الخرائط الارتباطية للكائنات (ORM) من Laravel، والذي يوفر طريقة مريحة ومعبرة للتفاعل مع قاعدة البيانات الخاصة بك. يسمح لك بتنفيذ عمليات قاعدة البيانات باستخدام بناء جملة موجه للكائنات بدلاً من كتابة استعلامات SQL الأولية. يتبع Eloquent نمط Active Record، حيث يكون لكل جدول قاعدة بيانات نموذج مطابق في تطبيقك.

**API resources in Laravel**

1. Creating API Resources

2. Defining Resource Structure

3. Using API Resources in Controllers

4. Resource Collections

5. Using Resource Collections

6. Conditional Transformation

**Vlidator class to apply validation in controller**

تسمح لك فئة Validator بإجراء التحقق مباشرة في وحدات التحكم الخاصة بك. يمكنك استخدامه للتحقق من صحة بيانات الطلبات الواردة، مثل مدخلات النماذج أو حمولات واجهة برمجة التطبيقات

1)Import the Validator Class

2)Perform Validation

3)Customizing Error Messages

4)Validate and Redirect (For Web Applications)