1. **ما هو Laravel؟**
   - **الإجابة:** هو إطار عمل PHP مفتوح المصدر يتيح تطوير تطبيقات الويب بسهولة باستخدام أنماط البرمجة الحديثة.
   - **مثال:** يشبه Laravel صندوق الأدوات الذي يساعدك على بناء منزل (تطبيق ويب) بسرعة، حيث يوفر الأدوات الأساسية مثل المطرقة والمسامير.
2. **ما هي مميزات Laravel الرئيسية؟**
   - **الإجابة:**
     - التوجيه (Routing).
     - ORM مع Eloquent.
     - جدولة المهام (Task Scheduling).
     - Blade Template Engine.
   - **مثال:** التوجيه في Laravel يشبه خرائط جوجل، حيث يوجه المستخدمين إلى الصفحة الصحيحة بناءً على طلباتهم.
3. **ما الفرق بين Laravel و CodeIgniter؟**
   - **الإجابة:** Laravel يدعم ORM ونظام التوجيه المتقدم، بينما CodeIgniter يعتمد على بنية أكثر بساطة بدون ORM مدمج.
   - **مثال:** Laravel يشبه سيارة كهربائية متقدمة، بينما CodeIgniter يشبه دراجة بخارية.

أسئلة حول Eloquent ORM

4. **ما هو Eloquent؟**
   - **الإجابة:** هو ORM في Laravel يتيح التعامل مع قواعد البيانات باستخدام الكائنات بدلاً من SQL.
   - **مثال:** بدلاً من كتابة SQL مثل `SELECT * FROM users;`، يمكن استخدام `User::all()`.
5. **كيف تعمل العلاقات في Eloquent؟**
   - **الإجابة:** عبر التعريف بـ `hasOne`, `hasMany`, `belongsTo`, وغيرها.
   - **مثال:** علاقة `hasMany` مثل "الأب لديه عدة أطفال"، حيث الأب يمثل النموذج الأب، والأطفال يمثلون النموذج الفرعي.

أسئلة حول Blade Templates

6. **ما هو Blade في Laravel؟**
   - **الإجابة:** هو محرك قوالب مدمج يتيح كتابة HTML مع PHP بسهولة.
   - **مثال:** يشبه تصميم بطاقة دعوة حيث يمكنك استخدام قوالب جاهزة وملء التفاصيل.
7. **كيف تستخدم `@yield` و `@section` في Blade؟**
   - **الإجابة:** يتم استخدامهما لتحديد محتوى يمكن تخصيصه.
   - **مثال:** يشبه ذلك إعداد صفحة غلاف مجلة، حيث يحتوي القالب الأساسي على العنوان الرئيسي، ويمكن تغيير الصور والمقالات لكل إصدار.

أسئلة حول Middleware

8. **ما هو Middleware؟**
   - **الإجابة:** هو طبقة وسيطة تمرر الطلبات بين المستخدم والتطبيق.
   - **مثال:** يشبه حارس البوابة الذي يتحقق من هوية الزائر قبل دخوله.
9. **كيف تنشئ Middleware مخصص؟**
   - **الإجابة:** باستخدام `php artisan make:middleware MiddlewareName`.
   - **مثال:** إنشاء Middleware يتحقق مما إذا كان المستخدم يمتلك دور "مدير" قبل الوصول إلى لوحة التحكم.

أسئلة حول التوجيه (Routing)

10. **ما الفرق بين Route::get و Route::post؟**
    - **الإجابة:** `get` يستخدم لجلب البيانات، و`post` لإرسالها.
    - **مثال:** عند البحث عن كتاب في مكتبة، تستخدم `GET`. عند إضافة كتاب جديد، تستخدم `POST`.
11. **كيف تُمرر معلمات إلى Route؟**
    - **الإجابة:** باستخدام `{parameter}` في التوجيه.
    - **مثال:** رابط مثل `profile/{id}/` يشبه بطاقة هوية لكل مستخدم.

أسئلة حول قواعد البيانات (Database)

12. **ما هو Migration؟**
    - **الإجابة:** هو وسيلة لإدارة قواعد البيانات باستخدام ملفات PHP.
    - **مثال:** يشبه تخطيط بناء جديد حيث تحدد مواقع الغرف مسبقًا.
13. **كيف تُنشئ جدولًا جديدًا باستخدام Migration؟**
    - **الإجابة:**
      1. تنفيذ الأمر: `php artisan make:migration create_table_name`.
      2. تعديل الملف لإضافة الحقول.
      3. تنفيذ الأمر: `php artisan migrate`.
    - **مثال:** يشبه ذلك إعداد جدول في قاعدة البيانات مثل إعداد صفوف وطاولات في مطعم.

أسئلة حول RESTful APIs

14. **كيف تنشئ API في Laravel؟**
    - **الإجابة:** عبر استخدام `Route::apiResource` أو `Route::get` مع Controller.
    - **مثال:** API يشبه قائمة طعام رقمية يمكن للعملاء طلبها عبر تطبيق الجوال.
15. **ما الفرق بين API Resource و API Resource Collection؟**
    - **الإجابة:** Resource يمثل كائنًا واحدًا، بينما Collection يمثل مجموعة من الكائنات.
    - **مثال:** Resource يشبه بطاقة هوية واحدة، وCollection يشبه كومة بطاقات.

أسئلة حول Auth

16. **كيف يعمل نظام المصادقة (Authentication) في Laravel؟**
    - **الإجابة:** باستخدام مكتبة `Auth` المدمجة.
    - **مثال:** يشبه ذلك تسجيل الدخول إلى تطبيق باستخدام اسم المستخدم وكلمة المرور.
17. **كيف تُخصص عملية تسجيل الدخول؟**
    - **الإجابة:** عبر تعديل ملف `LoginController` أو إنشاء Guard مخصص.
    - **مثال:** تسجيل الدخول بشهادة بدلاً من كلمة مرور يشبه استخدام بطاقة هوية بدلاً من مفتاح.

أسئلة إضافية مع أمثلة حياتية

18. **ما هو Queue في Laravel؟**
    - **الإجابة:** هو نظام لإدارة المهام الخلفية.
    - **مثال:** يشبه صف العملاء في متجر ينتظرون دورهم.
19. **ما هو Event و Listener؟**
    - **الإجابة:** Event يمثل حدثًا معينًا، وListener يستجيب له.
    - **مثال:** عند الضغط على زر في المصعد (Event)، يتحرك المصعد (Listener).
20. **ما هي الفوائد الرئيسية لاستخدام Laravel؟**
    - **الإجابة:** تنظيم الكود، سرعة التطوير، دعم المجتمع، والأمان.
    - **مثال:** مثل استخدام سيارة حديثة تحتوي على كل الميزات الأساسية لتوفير الوقت والجهد.

---

21. **كيف يمكن تحسين أداء تطبيق Laravel؟**
    - **الإجابة:**
      - استخدام الكاش (Cache).
      - تقليل استعلامات قاعدة البيانات.
      - تحميل الملفات بشكل lazy.
      - تحسين وقت الاستجابة باستخدام CDN.
    - **مثال:** يشبه ذلك تقليل حركة المرور على الطريق باستخدام إشارات ذكية لتسريع تدفق السيارات.
22. **ما هو الكاش (Cache) في Laravel وكيف يمكن استخدامه؟**
    - **الإجابة:** الكاش هو نظام لتخزين البيانات مؤقتًا لتسريع الأداء. يمكن استخدامه عبر `Cache::put()` و `Cache::get()`.
    - **مثال:** يشبه الاحتفاظ بمفتاح الباب في جيبك بدلاً من البحث عنه في كل مرة.
23. **ما هي Artisan Commands وكيف تُنشئ أوامر مخصصة؟**
    - **الإجابة:** Artisan هو CLI مدمج في Laravel. لإنشاء أمر مخصص، استخدم `php artisan make:command`.
    - **مثال:** يشبه ذلك إعداد روبوت يقوم بمهمة معينة نيابة عنك.

أسئلة حول الأمان (Security)

24. **كيف يحمي Laravel من هجمات SQL Injection؟**
    - **الإجابة:** عبر استخدام Query Builder و Eloquent، حيث يتم تعقيم البيانات تلقائيًا.
    - **مثال:** يشبه استخدام قفل ذكي يتحقق من البصمة قبل السماح بالدخول.
25. **ما هو CSRF وكيف يتم الحماية منه؟**
    - **الإجابة:** CSRF هي هجمات تُجبر المستخدم على تنفيذ طلب غير مرغوب فيه. Laravel يحمي منها باستخدام توكن CSRF.
    - **مثال:** يشبه رمز PIN على بطاقة الصراف لحماية أموالك.
26. **كيف يمكن تشفير البيانات في Laravel؟**
    - **الإجابة:** باستخدام واجهة `Crypt` المدمجة.
    - **مثال:** مثل تخزين بياناتك المهمة في خزنة مع قفل رقمي.

أسئلة حول الحزم (Packages)

27. **كيف تضيف حزمة جديدة في Laravel؟**
    - **الإجابة:** باستخدام Composer مثل: `composer require package/name`.
    - **مثال:** يشبه ذلك إضافة تطبيق جديد على هاتفك لتحسين وظيفته.
28. **كيف تُنشئ حزمة مخصصة في Laravel؟**
    - **الإجابة:** عبر إنشاء مجلد في `/packages` وتكوين ملفات الحزمة.
    - **مثال:** يشبه ذلك تصميم أداة جديدة بنفسك بدلاً من شرائها.

أسئلة حول العلاقات (Relationships)

29. **ما الفرق بين `hasOne` و `hasMany`؟**
    - **الإجابة:**
        - `hasOne`: علاقة فردية.
        - `hasMany`: علاقة متعددة.
    - **مثال:**
        - `hasOne`: مثل شخص واحد يمتلك سيارة واحدة.
        - `hasMany`: مثل أم لديها عدة أطفال.
30. **كيف تُنشى علاقة Many-to-Many؟**
    - **الإجابة:** عبر استخدام جدول وسيط يحتوي على المفاتيح الخارجية.
    - **مثال:** علاقة الطلاب بالدورات، حيث يمكن لكل طالب الالتحاق بعدة دورات وكل دورة تحتوي عدة طلاب.

أسئلة حول الإشعارات (Notifications)

31. **ما هي الإشعارات في Laravel؟**
    - **الإجابة:** نظام لإرسال تنبيهات للمستخدمين عبر قنوات مثل البريد الإلكتروني أو الرسائل النصية.
    - **مثال:** إشعار من البنك عند إجراء معاملة مالية.
32. **كيف تُرسل إشعارًا بالبريد الإلكتروني؟**
    - **الإجابة:** باستخدام `Notification::send()` أو `notify()` في النموذج.
    - **مثال:** إرسال رسالة تهنئة بعيد ميلاد العميل.

أسئلة حول الأحداث (Events)

33. **ما الفرق بين الحدث (Event) والمستمع (Listener)؟**
    - **الإجابة:**
        - الحدث هو شيء يحدث في النظام.
        - المستمع هو الذي ينفذ مهمة بناءً على الحدث.
    - **مثال:** عند فتح الباب (Event)، يعمل الإنذار (Listener).
34. **كيف تُنشئ حدثًا مخصصًا؟**
    - **الإجابة:** باستخدام الأمر `php artisan make:event EventName`.
    - **مثال:** إنشاء حدث لإرسال بريد إلكتروني عند تسجيل مستخدم جديد.

أسئلة حول جدولة المهام (Task Scheduling)

35. **ما هي جدولة المهام في Laravel؟**
    - **الإجابة:** هي نظام لتشغيل المهام تلقائيًا في وقت محدد باستخدام الكرون (Cron Jobs).
    - **مثال:** إرسال تقارير يومية عبر البريد الإلكتروني.
36. **كيف تُضيف مهمة مجدولة؟**
    - **الإجابة:** عبر تعديل `app/Console/Kernel.php` وإضافة المهمة في `schedule()` method.
    - **مثال:** تشغيل مهمة تنظيف الملفات المؤقتة يوميًا في منتصف الليل.

أسئلة حول APIs

37. **كيف يمكن تأمين API في Laravel؟**
    - **الإجابة:** باستخدام JWT أو Sanctum للتوثيق.
    - **مثال:** يشبه طلب هوية شخصية عند دخول مبنى حكومي.
38. **ما الفرق بين Passport و Sanctum؟**
    - **الإجابة:**
        - Passport يُستخدم للتطبيقات الكبيرة مع OAuth2.
        - Sanctum يُستخدم للتطبيقات الصغيرة أو SPA.

- **مثال:** Passport مثل جواز السفر الدولي، وSanctum مثل بطاقة تعريف محلية.

أسئلة حول الحاويات (Containers)

39. **ما هي الحاويات في Laravel؟**
    - **الإجابة:** حاوية خدمة تساعد على إدارة التبعيات بين الكائنات.
    - **مثال:** مثل حقيبة تنظيم تحتوي كل أدواتك.
40. **كيف تعمل Dependency Injection في Laravel؟**
    - **الإجابة:** عبر تمرير الكائنات إلى الكلاسات بدلاً من إنشائها داخليًا.
    - **مثال:** مثل طلب وجبة جاهزة بدلاً من إعدادها بنفسك.

أسئلة متنوعة

41. **كيف تعمل ملفات Config في Laravel؟**
    - **الإجابة:** توفر إعدادات التطبيق وتُحمل تلقائيًا عند التشغيل.
    - **مثال:** مثل دفتر التعليمات الذي يحدد إعدادات جهاز جديد.
42. **كيف تُخصص رسالة تحقق الحقول؟**
    - **الإجابة:** باستخدام `messages()` method في قواعد التحقق.
    - **مثال:** تخصيص رسالة "حقل الاسم مطلوب" لتكون "الرجاء إدخال اسمك".

---

43. **ما هي الطريقة المفضلة للتحقق من الحقول في Laravel؟**
    - **الإجابة:** باستخدام `Form Request` للحصول على تحكم أكبر وتنظيم التحقق.
    - **مثال:** مثل فصل قائمة الطلبات في المطعم لتسهيل التنظيم.
44. **كيف تتحقق من أن القيمة رقمية؟**
    - **الإجابة:** باستخدام القاعدة `numeric`.
    - **مثال:** التحقق من عمر المستخدم بأنه رقم وليس نصًا.
45. **كيف تتحقق من أن النص يتكون من عدد معين من الأحرف؟**
    - **الإجابة:** باستخدام القاعدة `min` و`max` أو `between`.
    - **مثال:** التحقق من أن كلمة المرور تحتوي على 8-16 حرفًا.
46. **كيف تتحقق من أن القيمة موجودة في قاعدة البيانات؟**
    - **الإجابة:** باستخدام القاعدة `exists`.
    - **مثال:** التحقق من أن البريد الإلكتروني موجود بالفعل عند محاولة تسجيل الدخول.
47. **كيف تتحقق من أن القيمة فريدة؟**
    - **الإجابة:** باستخدام القاعدة `unique`.
    - **مثال:** التأكد من أن اسم المستخدم لم يُستخدم من قبل.

أسئلة حول النطاقات (Scopes)

48. **ما هي النطاقات (Scopes) في Eloquent؟**
    - **الإجابة:** هي طرق تُستخدم لتصفية الاستعلامات.
    - **مثال:** إنشاء نطاق لعرض المنتجات المتاحة فقط.
49. **ما الفرق بين Global Scope و Local Scope؟**
    - **الإجابة:**
        - Global Scope: يطبق على جميع الاستعلامات.
        - Local Scope: يتم استدعاؤه عند الحاجة.
    - **مثال:**
        - Global Scope: عرض المستخدمين الفعّالين دائمًا.
        - Local Scope: تصفية المستخدمين بناءً على العمر.

أسئلة حول الصفوف الوسيطة (Middleware)

50. **ما هو Middleware وكيف يعمل؟**
    - **الإجابة:** هو فلتر يعترض الطلبات قبل أن تصل إلى التطبيق.
    - **مثال:** مثل حارس أمن يفحص التصاريح قبل السماح بالدخول.
51. **كيف تُنشئ Middleware مخصص؟**
    - **الإجابة:** باستخدام الأمر `php artisan make:middleware MiddlewareName`.
    - **مثال:** إنشاء Middleware للتحقق من عمر المستخدم.

أسئلة حول المعالجات (Handlers)

52. **كيف تُدير الأخطاء في Laravel؟**
    - **الإجابة:** باستخدام ملفات `Handler` في `app/Exceptions`.
    - **مثال:** مثل وضع خطة طوارئ في حالة حدوث مشكلة.
53. **كيف تُخصص رسالة خطأ مخصصة؟**
    - **الإجابة:** عبر تعديل `report` أو `render` في `Handler.php`.
    - **مثال:** تخصيص رسالة "الصفحة غير موجودة" لتكون أكثر وضوحًا.

أسئلة حول تعدد اللغات (Localization)

54. **كيف يتم التعامل مع تعدد اللغات في Laravel؟**
    - **الإجابة:** باستخدام ملفات الترجمة في مجلد `resources/lang`.
    - **مثال:** عرض الموقع باللغتين الإنجليزية والعربية حسب تفضيل المستخدم.
55. **كيف تُغير لغة التطبيق؟**
    - **الإجابة:** باستخدام `App::setLocale('lang')`.

- **مثال:** تغيير لغة واجهة المستخدم إلى الفرنسية.

أسئلة حول وحدات الاختبار (Testing)

56. **كيف تُجري اختبارات في Laravel؟**
    - **الإجابة:** باستخدام PHPUnit أو Pest.
    - **مثال:** اختبار تسجيل الدخول للتأكد من أن النظام يعمل كما هو متوقع.
57. **ما هو الفرق بين Feature Tests و Unit Tests؟**
    - **الإجابة:**
      - Unit Tests: لاختبار وظائف صغيرة مثل الكلاسات.
      - Feature Tests: لاختبار سيناريوهات التطبيق الكاملة.
    - **مثال:**
      - Unit Test: اختبار إضافة منتج.
      - Feature Test: اختبار عملية الشراء بأكملها.

أسئلة حول التحميل (File Upload)

58. **كيف تُحمّل ملفًا في Laravel؟**
    - **الإجابة:** باستخدام `Storage` أو `request->file()`.
    - **مثال:** تحميل صورة الملف الشخصي للمستخدم.
59. **كيف تتحقق من نوع الملف قبل التحميل؟**
    - **الإجابة:** باستخدام قواعد التحقق `mimes` أو `mimetypes`.
    - **مثال:** التحقق من أن الملف هو صورة.
60. **كيف تُخزّن الملفات على Amazon S3؟**
    - **الإجابة:** إعداد S3 في ملف `filesystems.php` واستخدام `Storage::disk('s3')->put()`.
    - **مثال:** مثل حفظ مستندات هامة في خدمة تخزين سحابية.

أسئلة حول إدارة الجداول (Migrations)

61. **كيف تُنشئ جدولًا جديدًا؟**
    - **الإجابة:** باستخدام `php artisan make:migration`.
    - **مثال:** إنشاء جدول للمستخدمين.
62. **كيف تُعدل جدولًا موجودًا؟**
    - **الإجابة:** باستخدام Migration جديدة مع أوامر التعديل مثل `addColumn` أو `dropColumn`.
    - **مثال:** إضافة عمود العمر إلى جدول المستخدمين.

أسئلة حول الأحداث المجدولة (Broadcasting)

63. **ما هو Broadcasting في Laravel؟**
    - **الإجابة:** هو نظام لإرسال الأحداث إلى واجهة المستخدم.
    - **مثال:** إشعار مباشر عند وصول رسالة جديدة.
64. **كيف تُستخدم Pusher مع Laravel؟**
    - **الإجابة:** إعداد Pusher في `broadcasting.php` واستخدام `event()`.
    - **مثال:** مثل بث مباراة كرة القدم مباشرة للمشاهدين.

أسئلة حول الحاويات (Service Containers)

65. **ما هو مفهوم Dependency Injection؟**
    - **الإجابة:** هو توفير التبعيات بدلاً من إنشائها داخل الكلاس.
    - **مثال:** مثل استئجار سيارة بدلاً من شرائها.
66. **كيف تُسجل خدمة جديدة في Service Container؟**
    - **الإجابة:** باستخدام `bind` أو `singleton` في `AppServiceProvider`.
    - **مثال:** تسجيل خدمة للرسائل النصية.

أسئلة حول العلاقات المتقدمة

67. **كيف تُنشى علاقة polymorphic؟**
    - **الإجابة:** باستخدام الأعمدة `type` و`id` لتحديد الكيان المرتبط.
    - **مثال:** نموذج تعليق يمكن أن يرتبط بمنتج أو مقال.

---

68. ما الفرق بين `Observer` و `Event`؟

- **الإجابة:**
    - **Observer**: يُستخدم لمراقبة تغييرات النموذج (Model) مثل الإنشاء أو التحديث أو الحذف.
    - **Event**: يُستخدم لإرسال إشارات مخصصة يمكن أن تستجيب لها وظائف متعددة (Listeners).
- **مثال:**
    - **Observer**: عند إنشاء مستخدم جديد، يتم إرسال بريد الترحيب تلقائيًا.
    - **Event**: بث إشعار إلى جميع المستخدمين عند تحديث منتج.

---

69. كيف تُنشئ مصادقات متعددة باستخدام `guards`؟

- **الإجابة:**
  1. في ملف `config/auth.php`، أضف تعريفًا جديدًا في قسم `guards`.
  2. قم بإنشاء Middleware خاص بهذا الحارس.
- **مثال:** السماح للمسؤولين بالدخول باستخدام حارس خاص بهم، بينما يستخدم العملاء حارسًا آخر.

---

70. ما هي Facades وكيف تعمل؟

- **الإجابة:**
Facades هي واجهات مبسطة للوصول إلى الخدمات داخل الحاوية (Service Container).
- **مثال:**
عند استخدام `Cache::get()` لاسترداد بيانات من الكاش بدلاً من استدعاء الخدمة يدويًا.

---

71. كيف تُنشئ Binding في Service Container؟

- **الإجابة:**
باستخدام `bind` أو `singleton` في `AppServiceProvider`.

مثال:
php

```
$this->app->bind('PaymentService', function ($app) {
    return new PaymentService();
});
```

-

---

72. ما هو مفهوم Rate Limiting؟

- **الإجابة:**
هو تحديد عدد الطلبات التي يمكن للمستخدم إرسالها خلال فترة زمنية محددة.
- **مثال:** منع المستخدم من إرسال أكثر من 100 طلب في الدقيقة.

---

73. كيف تُخصص مكونًا في Blade؟

- **الإجابة:**
بإنشاء ملف مخصص في مجلد `resources/views/components` واستخدامه داخل القوالب.

مثال:
إنشاء مكون لإظهار زر:
php

```
</ "<x-button text="Submit
```

-

---

74. كيف تُنشئ جدول Pivot لعلاقات Many-to-Many؟

- **الإجابة:**
باستخدام Migration يحتوي على مفتاحين أجنبيين يمثلان النماذج المرتبطة.
- **مثال:**
جدول `user_role` لربط المستخدمين بالأدوار.

---

75. ما الفرق بين `hasOneThrough` و `hasManyThrough`؟

- **الإجابة:**
  - **hasOneThrough**: تُستخدم للوصول إلى سجل واحد عبر علاقة وسيطة.
  - **hasManyThrough**: تُستخدم للوصول إلى عدة سجلات عبر علاقة وسيطة.
- **مثال:**
  - `hasOneThrough`: الوصول إلى عنوان العميل عبر الطلب.
  - `hasManyThrough`: الوصول إلى المنتجات عبر الطلبات.

---

76. كيف تستخدم Redis في Laravel؟

- **الإجابة:**
قم بتثبيت Redis باستخدام Composer وأعد تهيئة الاتصال في ملف `.env`.
- **مثال:**
استخدام Redis لتخزين بيانات الكاش بسرعة.

---

77. ما هي مزايا Task Queues؟

- **الإجابة:**
تسريع التطبيق بتشغيل المهام الطويلة في الخلفية.
- **مثال:** إرسال رسائل البريد الإلكتروني أثناء تشغيل التطبيق.

78. كيف تُنشئ API Resource؟

- **الإجابة:**
باستخدام الأمر `php artisan make:resource`.
- **مثال:**
تنسيق بيانات المستخدم قبل إرسالها عبر API.

---

79. كيف تعمل المصفوفات في Laravel؟

- **الإجابة:**
باستخدام الـ `Collection` لمعالجة البيانات بسهولة.
- **مثال:**
تصفية قائمة المنتجات بناءً على السعر.

---

80. ما الفرق بين `Session` و `Cache`؟

- **الإجابة:**
  - **Session**: تُستخدم لتخزين البيانات المتعلقة بالمستخدمين مثل تسجيل الدخول.
  - **Cache**: تُستخدم لتخزين البيانات المؤقتة لتحسين الأداء.
- **مثال:**
  - Session: تخزين سلة مشتريات المستخدم.
  - Cache: تخزين نتائج استعلام ثقيل.

---

81. كيف تُنشئ Event Listeners مخصصين؟

- **الإجابة:**
باستخدام الأمر `php artisan make:listener`.
- **مثال:** الاستجابة لإرسال بريد عند إتمام الطلب.

---

82. ما هو مفهوم Stubs في Artisan؟

- **الإجابة:**
هي قوالب تُستخدم لإنشاء ملفات جديدة.
- **مثال:** تخصيص القالب الافتراضي عند إنشاء Controller.

83. كيف تعمل Mutators و Accessors؟

- **الإجابة:**
  - **Accessor**: لتنسيق البيانات عند قراءتها.
  - **Mutator**: لتنسيق البيانات عند كتابتها.
- **مثال:**
  - Accessor: إضافة "د.م" إلى سعر المنتج.
  - Mutator: تخزين الاسم بالأحرف الكبيرة.

---

84. ما هو مفهوم Service Providers؟

- **الإجابة:**
  هي المكان الذي تُسجل فيه الخدمات داخل التطبيق.
- **مثال:** تسجيل خدمة الدفع.

---

85. كيف تُدمج Vue.js مع Laravel؟

- **الإجابة:**
  عن طريق إعداد Webpack أو Vite لتجميع مكونات Vue.js.
- **مثال:** إنشاء واجهة تفاعلية لعربة التسوق.

---

86. ما الفرق بين `call` و `callSilent`؟

- **الإجابة:**
  - **call**: يعرض الإخراج في الطرفية.
  - **callSilent**: يُنفذ الأمر دون إظهار الإخراج.
- **مثال:** استخدام `callSilent` لتشغيل مهمة مجدولة بدون تسجيلها.

---

87. كيف تعمل ميزة Soft Delete؟

- **الإجابة:**
  بتضمين خاصية `SoftDeletes` في النموذج.
- **مثال:** حذف منتج مع إمكانية استرجاعه لاحقًا.

---

88. كيف تُخصص صفحة الخطأ 404؟

- **الإجابة:**
بإنشاء ملف مخصص `resources/views/errors/404.blade.php`.
- **مثال:** عرض رسالة ودية للمستخدم عند الخطأ.

---

89. ما هي الـ Collections؟

- **الإجابة:**
هي كائنات تُستخدم لمعالجة البيانات بطريقة مريحة.
- **مثال:** فرز قائمة المنتجات حسب السعر.

---

90. كيف تُستخدم التوكنات مع API؟

- **الإجابة:**
باستخدام Laravel Sanctum أو Passport لإدارة المصادقة.
- **مثال:** منح الوصول إلى API بناءً على التوكن.

---

91. كيف تعمل قنوات التحقق (Validation Channels)؟

- **الإجابة:**
تُحدد كيفية إرسال الأكواد (SMS، بريد إلكتروني).
- **مثال:** إرسال رمز تحقق عبر SMS.

---

92. كيف تُحدد مناطق التوقيت (Timezone)؟

- **الإجابة:**
بتحديث الإعدادات في ملف `.env`.
- **مثال:** تعيين توقيت السعودية كمنطقة زمنية افتراضية.

---

93. ما الفرق بين `public` و `storage`؟

- **الإجابة:**
  - **public**: يُستخدم للوصول إلى الملفات مباشرة من الويب.
  - **storage**: يُستخدم لتخزين الملفات بشكل آمن.

- **مثال:** تخزين الصور في `storage` واسترجاعها عبر رابط.

---

94. كيف تُضيف سياسة (Policy) جديدة؟

- **الإجابة:**
  باستخدام الأمر `php artisan make:policy`.
- **مثال:** تحديد صلاحيات المشرف على حذف المستخدمين.

---

95. كيف تُدير الحقول المشفرة (Encrypted Fields)؟

- **الإجابة:**
  باستخدام `Crypt` لتشفير وفك تشفير البيانات.
- **مثال:** تشفير كلمات المرور.

---

96. كيف تُخصص ملفات Log؟

- **الإجابة:**
  عبر تعديل إعدادات `logging.php`.
- **مثال:** فصل سجلات الأخطاء عن العمليات.

---

97. كيف تُنشئ محاكاة (Mocking) في الاختبارات؟

- **الإجابة:**
  باستخدام `Mockery` أو Laravel's `Mocking`.
- **مثال:** اختبار API بدون الاتصال بخدمة خارجية.

---

98. ما الفرق بين Queue و Jobs؟

- **الإجابة:**
  - **Queue**: نظام إدارة المهام المؤجلة.
  - **Jobs**: المهام الفعلية التي تُنفذ في الصفوف.
- **مثال:** إرسال رسائل البريد عبر Queue.

---

99. كيف تُضيف Websockets في Laravel؟

- **الإجابة:**
  باستخدام مكتبات مثل Pusher أو Laravel Echo.
- **مثال:** تحديث حالة الطلب مباشرة عند تغييره.

---

100. ما هو مفهوم Lazy Loading و Eager Loading؟

- **الإجابة:**
  - **Lazy Loading**: تحميل البيانات عند الحاجة.
  - **Eager Loading**: تحميل البيانات مقدمًا لتقليل عدد الاستعلامات.
- **مثال:**
  - Lazy: استدعاء المنتجات عند الحاجة فقط.
  - Eager: استدعاء المنتجات والفئات المرتبطة بها دفعة واحدة.