98 Laravel interview questions to hire top engineers

Questions

1. What is Laravel and why would you choose it over plain PHP?

2. Explain the concept of 'artisan' in Laravel.

3. What are Laravel's key features?

4. Describe the purpose of a migration in Laravel.

5. What is the role of a 'Model' in Laravel's MVC architecture?

6. What is a 'Controller' in Laravel, and what does it do?

7. Explain the purpose of 'Routes' in a Laravel application.

8. What are Laravel 'Views', and how are they used?

9. Describe what 'Blade templating' is in Laravel.

10. How do you create a new Laravel project?

11. What is Composer, and why is it important for Laravel?

12. How do you configure a database connection in Laravel?

13. Explain how to retrieve data from a database using Laravel.

14. How do you pass data from a controller to a view?

15. What is the purpose of the .env file in Laravel?

16. How do you handle form submissions in Laravel?

17. Explain how to use Laravel's built-in authentication features.

18. What are 'seeders' in Laravel, and when would you use them?

19. Describe how you would implement basic input validation in Laravel.

20. What is the purpose of middleware in Laravel?

21. How do you define and use a custom middleware in Laravel?

22. Explain the concept of dependency injection in Laravel.

23. What are Service Providers in Laravel, and what do they do?

24. Describe how to use Laravel's Eloquent ORM to interact with a database.

25. How do you define relationships between Eloquent models?

26. Explain how to use Laravel's built-in pagination.

27. Describe the purpose and benefits of using Laravel Mix.

28. Explain the concept of Service Containers in Laravel and how they facilitate dependency injection.

29. Describe how you would implement a custom Artisan command that seeds data into the database based on a specific environment.

30. How would you go about optimizing database query performance in a Laravel application, considering eager loading and query caching?

31. What are the differences between `findOrFail()` and `firstOrFail()` methods in Eloquent, and when would you use each?

32. Explain the purpose and usage of Laravel's event system, and provide an example of how you might use events and listeners.

33. How would you implement a middleware to check if a user has a specific permission before accessing a route?

34. Describe the process of creating and using custom validation rules in Laravel, including how to handle error messages.

35. Explain how you would use Laravel's queue system to handle long-running tasks, and the benefits of doing so.

36. What are the different types of relationships in Eloquent (one-to-one, one-to-many, many-to-many), and how do you define them?

37. Describe how you would implement API authentication using Laravel Passport.

38. Explain the purpose of Laravel's service providers and how they are used to bootstrap application services.

39. How would you implement a feature that allows users to upload files to a specific storage location, including validation and security considerations?

40. Explain the difference between the `get()` and `value()` methods when retrieving configuration values in Laravel.

41. Describe how you would handle exceptions and errors in a Laravel application, including logging and custom error pages.

42. How would you implement a search functionality in a Laravel application, considering performance and relevance?

43. Explain the purpose and usage of route model binding in Laravel.

44. Describe how you would implement a multi-language (localization) feature in a Laravel application.

45. How do you use factories and seeders to create database records for testing or development purposes?

46. Explain the concept of route groups and their benefits.

47. Describe how you would implement a role-based access control (RBAC) system in a Laravel application.

48. How can you prevent SQL injection vulnerabilities in your Laravel application when using raw queries?

49. Explain how you would implement a real-time chat feature using Laravel and WebSockets.

50. Describe the process of creating and using custom Eloquent accessors and mutators.

51. How would you implement a scheduled task (cron job) in Laravel?

52. Explain Laravel's service container and how it facilitates dependency injection. Provide a practical example of its usage.

53. Describe the differences between Queues and Events in Laravel. When would you use one over the other?

54. How can you optimize database queries in Laravel applications? Discuss techniques like eager loading, query caching, and indexing.

55. Explain how Laravel's middleware works. Create a scenario where you'd implement custom middleware.

56. Describe the process of creating and using custom Artisan commands in Laravel. Give an example use case.

57. How does Laravel's encryption work? What methods are available for securing sensitive data?

58. Explain the purpose and usage of Laravel's Policies for authorization. Provide a basic policy example.

59. Describe the different types of relationships available in Laravel Eloquent ORM. How do you use them effectively?

60. How do you implement API authentication in Laravel? Discuss different approaches like Sanctum, Passport, or JWT.

61. Explain Laravel's broadcasting feature. How does it work, and what real-time functionalities does it enable?

62. What are Laravel's Service Providers and how are they used to bootstrap application components?

63. Describe the process of implementing localization in a Laravel application to support multiple languages.

64. How can you implement custom validation rules in Laravel? Give an example of a complex validation scenario.

65. Explain how to handle file uploads in Laravel, including validation, storage, and retrieval.

66. Describe the different methods for testing Laravel applications (unit, feature, integration). Give examples of each.

67. How can you use Laravel's events and listeners to decouple application components and handle side effects?

68. Explain the process of deploying a Laravel application to a production environment. What are some key considerations?

69. How do you handle exceptions and errors in Laravel? Discuss custom error pages and logging.

70. Describe how you can use Laravel's task scheduling feature to automate recurring tasks.

71. How can you use Laravel's Telescope to debug and monitor your application's performance?

72. How would you optimize a Laravel application dealing with millions of database records to ensure quick response times?

73. Describe your approach to building a highly scalable API using Laravel, considering rate limiting, versioning, and security.

74. Explain how you would implement a custom authentication guard in Laravel, detailing the steps and considerations involved.

75. How would you design and implement a complex queuing system in Laravel, capable of handling different types of jobs with varying priorities and retry strategies?

76. Discuss your experience with integrating Laravel with different front-end frameworks like React, Vue.js, or Angular, and how you managed the API interactions and state management.

77. Describe your strategy for handling transactions across multiple database connections in Laravel to ensure data consistency.

78. How would you implement a robust role-based access control (RBAC) system in Laravel, allowing fine-grained control over user permissions and resources?

79. Explain how you would approach testing a complex Laravel application, including unit, integration, and end-to-end tests, and the tools you would use.

80. How do you handle different environments such as development, staging, and production in a Laravel project and ensure consistency across them?

81. Describe your experience with debugging and profiling Laravel applications to identify and resolve performance bottlenecks.

82. How would you implement a feature flag system in Laravel to enable or disable features dynamically without deploying new code?

83. Explain how you would approach building a multi-tenant application using Laravel, considering data isolation and shared resources.

84. How would you secure a Laravel application against common web vulnerabilities like SQL injection, XSS, and CSRF attacks?

85. Describe your experience with using Laravel's event system to decouple components and improve the maintainability of a large application.

86. How do you implement and manage API versioning in a Laravel application to ensure backward compatibility?

87. Explain your approach to handling file uploads and storage in Laravel, considering different storage providers and security implications.

88. How would you implement a real-time notification system in Laravel using WebSockets or server-sent events (SSE)?

89. Describe your experience with contributing to open-source Laravel packages or creating your own, and the challenges you faced.

90. How would you implement a search functionality in Laravel that can handle large datasets and provide relevant results quickly?

91. Explain how you would handle localization (i18n) and internationalization (I10n) in a Laravel application to support multiple languages and regions.

92. How do you monitor the performance and health of a live Laravel application and respond to issues proactively?

93. Explain how you would implement two-factor authentication (2FA) in a Laravel application.

94. How would you design a system for handling scheduled tasks in Laravel, considering concurrency and error handling?

95. Describe your experience with using Laravel's collection pipeline to perform complex data transformations.

96. How do you ensure the quality of your Laravel code through code reviews, static analysis, and automated testing?

97. Explain how you would integrate a third-party payment gateway (e.g., Stripe, PayPal) into a Laravel application.