20 best practices for Laravel development

1. Follow the MVC Pattern: Organize your code following the Model-View-Controller (MVC) pattern to separate concerns and make the codebase more maintainable.
2. Use Route Caching: Cache your routes to improve application performance by running the php artisan route:cache command.
3. Utilize Eloquent Relationships: Leverage Eloquent relationships to efficiently query related data and avoid excessive database queries.
4. Optimize Database Queries: Use eager loading, query optimization techniques, and indexes to improve database performance.
5. Implement Validation: Always validate user input to prevent security vulnerabilities and ensure data integrity.
6. Apply Middleware: Use middleware for cross-cutting concerns such as authentication, authorization, and request/response modifications.
7. Use Queues for Heavy Tasks: Offload time-consuming tasks to queues using Laravel's built-in queue system to enhance application responsiveness.
8. Sanitize User Input: Sanitize user input to prevent XSS and other security vulnerabilities.
9. Optimize Views: Cache views and use Blade template inheritance to reduce view rendering time.
10. Keep Controllers Slim: Keep your controllers focused on handling HTTP requests and delegate business logic to services or models.
11. Implement Exception Handling: Handle exceptions gracefully and log errors to facilitate debugging.
12. Use Named Routes: Define named routes to improve code readability and maintainability.
13. Utilize Route Model Binding: Use route model binding to automatically inject model instances into controller methods.
14. Implement Database Migrations: Use migrations to version and manage database schema changes effectively.
15. Apply Dependency Injection: Use dependency injection to decouple components and improve testability.
16. Write Unit Tests: Write comprehensive unit tests to ensure code correctness and maintainability.
17. Avoid Fat Controllers and Models: Keep your controllers and models focused and avoid having them grow too large.
18. Secure Your Application: Protect your application against common security threats by implementing CSRF protection, rate limiting, etc.
19. Use Environment Variables: Store sensitive information and configuration settings in environment variables.
20. Follow PSR Standards: Adhere to PSR coding standards to ensure code consistency and collaboration with the wider PHP community

Adopting these best practices will help you build more robust, scalable, and maintainable Laravel applications.

Happy coding

Revision #2
Created 23 May 2022 06:53:22
Updated 25 July 2023 07:48:50 by Milad majd