Laravel Coding Standards

Naming Conventions

- Use PascalCase for class names (e.g., UserController, OrderService).
- Use camelCase for variable and method names.
- Database tables must use snake_case and be plural (e.g., users, order_items).
- Route names should be kebab-case (e.g., user-profile, order-history).
- Eloquent model names must be singular and match the table name (User -> users).
- Event and Listener classes must use past tense and descriptive names (e.g., UserRegistered).

Security Practices

- Always use CSRF tokens (enabled by default).
- Use route model binding to prevent ID manipulation.
- Escape all user inputs using Blade's {{ }} syntax.
- Store secrets and credentials in .env file only.
- Hash passwords using bcrypt or Argon2 (via Hash facade).
- Use Policies and Gates for authorization, never rely on front-end checks.
- Validate all inputs using Form Requests or Validator.
- Sanitize file uploads and limit file types.

Syntax Structure

- Follow PSR-12 and Laravel naming conventions strictly.
- Use namespaces at the top, followed by use statements, then class definitions.
- Use constructor injection for dependencies.
- Always define return types and parameter types where applicable.
- Use short array syntax [].
- Group related middleware and controllers logically.

Error Handling

- Use try-catch blocks only when necessary, otherwise use global exception handler (Handler.php).
- Log errors using the Log facade or Monolog channels.
- Do not expose sensitive error details in production.
- Use custom exception classes for domain-specific errors.
- Use report() and render() methods in Handler for advanced handling.

Statements & Control Flow

- Always use braces for control structures even for single statements.
- Use early returns to reduce nesting.
- Prefer collection methods over raw loops.
- Use strict comparisons (===) for conditional checks.
- Avoid complex conditions by extracting logic into helper methods.

File Organization

- Controllers go in app/Http/Controllers.
- Models go in app/Models.
- Routes defined in routes/web.php or routes/api.php.
- Service classes go in app/Services, repository pattern in app/Repositories.
- Helper functions go in app/Helpers and loaded via composer.json.
- Configuration files go in config/ and should be environment agnostic.

Comments & Documentation

- Use PHPDoc blocks for all classes, methods, and properties.
- Include @param, @return, @throws annotations as needed.
- Keep inline comments brief and meaningful.
- Document business logic and complex queries.
- Update documentation when logic changes.

Indentation & Formatting

- Use 4 spaces for indentation.
- Max line length of 120 characters.
- Leave one blank line between class methods.
- Use consistent spacing in arrays and function calls.
- Remove trailing whitespace and unnecessary blank lines.

Best Practices

- Keep controllers thin — delegate logic to services or jobs.
- Use queues for long-running tasks.
- Leverage dependency injection, avoid facades in business logic.
- Keep environment-specific configs in .env files only.
- Use migrations and seeders for database changes.
- Version your APIs and use Resource classes for responses.