Laravel Quick Reference

Artisan, routing, Eloquent, Blade, middleware, auth

Artisan

Common Commands

`php artisan serve`	Start development server
`php artisan make:model Name -m`	Create model with migration
`php artisan make:controller NameController`	Create controller class
`php artisan make:middleware Name`	Create middleware class
`php artisan migrate`	Run pending migrations
`php artisan migrate:rollback`	Rollback last migration batch
`php artisan db:seed`	Run database seeders
`php artisan tinker`	Interactive REPL for your app
`php artisan route:list`	List all registered routes
`php artisan cache:clear`	Clear the application cache
`php artisan config:clear`	Clear cached config
`php artisan queue:work`	Start processing queued jobs

Routing

Basic Routes

```
Route::get('/users', [UserController::class, 'index']);
Route::post('/users', [UserController::class, 'store']);
Route::put('/users/{id}', [UserController::class, 'update']);
Route::delete('/users/{id}', [UserController::class, 'destroy']);
```

Route Parameters & Groups

```
Route::get('/user/{id}', function (int $id) {
    return User::findOrFail($id);
});

Route::prefix('api')->middleware('auth')->group(function () {
    Route::get('/profile', [ProfileController::class, 'show']);
});
```

Route Features

`->name('route.name')`	Named route for URL generation
`->where('id', '[0-9]+')`	Regex constraint on parameter
`Route::resource()`	RESTful resource routes (7 routes)
`Route::apiResource()`	API resource (no create/edit views)
`Route::fallback()`	Catch-all for unmatched routes

Controllers

Resource Controller

```
class PostController extends Controller {
    public function index() {
        return view('posts.index', ['posts' => Post::all()]);
    }

    public function store(Request $request) {
        $validated = $request->validate(['title' => 'required|
max:255']);
        Post::create($validated);
        return redirect()->route('posts.index');
    }
}
```

Resource Methods

`index()`	GET /resource -- list all
`create()`	GET /resource/create -- show form
`store()`	POST /resource -- save new
`show($id)`	GET /resource/{id} -- display one
`edit($id)`	GET /resource/{id}/edit -- edit form
`update($id)`	PUT /resource/{id} -- update
`destroy($id)`	DELETE /resource/{id} -- remove

Blade Templates

Layout & Sections

```
{{-- layouts/app.blade.php --}}
<html><body>
  @yield('content')
</body></html>

{{-- pages/home.blade.php --}}
@extends('layouts.app')
@section('content')
  <h1>Home</h1>
@endsection
```

Directives

`{{ $var }}`	Echo with HTML escaping
`{!! $html !!}`	Echo raw (unescaped)
`@if / @elseif / @else`	Conditional blocks
`@foreach ($items as $item)`	Loop through collection
`@forelse / @empty`	Loop with empty fallback
`@include('partial')`	Include another Blade view
`@component / @slot`	Reusable Blade components
`@csrf`	CSRF token hidden field
`@auth / @guest`	Check authentication status
`@error('field')`	Display validation error

Eloquent ORM

Model Basics

```
class Post extends Model {
    protected $fillable = ['title', 'body', 'user_id'];

    public function user() {
        return $this->belongsTo(User::class);
    }
}
```

Querying

```
Post::all();                          // all records
Post::find(1);                        // by primary key
Post::where('status', 'published')->get();
Post::where('views', '>', 100)->orderBy('created_at', 'desc')-
>first();
```

CRUD Operations

```
$post = Post::create(['title' => 'New', 'body' => '...']);
$post->update(['title' => 'Updated']);
$post->delete();
Post::destroy([1, 2, 3]);   // delete by IDs
```

Relationships

`hasOne`	One-to-one (User -> Phone)
`hasMany`	One-to-many (Post -> Comments)
`belongsTo`	Inverse of hasOne/hasMany
`belongsToMany`	Many-to-many with pivot table
`hasManyThrough`	Has-many via intermediate model

Migrations

Creating Tables

```
Schema::create('posts', function (Blueprint $table) {
    $table->id();
    $table->foreignId('user_id')->constrained()->cascadeOnDelete();
    $table->string('title');
    $table->text('body')->nullable();
    $table->timestamps();
});
```

Column Types

`$table->id()`	Auto-increment BIGINT primary key
`$table->string('col', 100)`	VARCHAR with optional length
`$table->text('col')`	TEXT column
`$table->integer('col')`	INTEGER column
`$table->boolean('col')`	BOOLEAN column
`$table->json('col')`	JSON column
`$table->timestamp('col')`	TIMESTAMP column
`$table->timestamps()`	created_at and updated_at
`$table->softDeletes()`	deleted_at for soft deletes

Middleware

Custom Middleware

```
class EnsureAdmin {
    public function handle(Request $request, Closure $next) {
        if (! $request->user()?->is_admin) {
            abort(403);
        }
        return $next($request);
    }
}
```

Registering & Using

```
// bootstrap/app.php
->withMiddleware(function (Middleware $middleware) {
    $middleware->alias(['admin' => EnsureAdmin::class]);
})

// In routes
Route::get('/admin', fn() => '...')->middleware('admin');
```

Built-in Middleware

`auth`	Require authentication
`guest`	Redirect if authenticated
`throttle:60,1`	Rate limit (60 req/min)
`verified`	Require email verification
`signed`	Validate signed URL

Laravel Quick Reference

Authentication

Auth Helpers

```
Auth::check();           // is user logged in?
Auth::user();            // current User model
Auth::id();              // current user ID
Auth::attempt(['email' => $e, 'password' => $p]);
Auth::logout();
```

Starter Kits

Laravel Breeze	Minimal auth scaffolding (Blade or Inertia)
Laravel Jetstream	Full-featured (teams, 2FA, API tokens)
Sanctum	SPA / mobile API token authentication
Passport	Full OAuth2 server implementation

Protecting Routes

```
Route::middleware('auth')->group(function () {
    Route::get('/dashboard', [DashController::class, 'index']);
});
```

Validation

Controller Validation

```
$validated = $request->validate([
    'title' => 'required|string|max:255',
    'email' => 'required|email|unique:users',
    'age'   => 'nullable|integer|min:0',
]);
```

Form Request

```
class StorePostRequest extends FormRequest {
    public function rules(): array {
        return [
            'title' => 'required|max:255',
            'body'  => 'required|min:10',
        ];
    }
}
```

Common Rules

required	Field must be present and not empty
string | integer | boolean	Type validation
min:N | max:N	Min/max length or value
email	Valid email format
unique:table,column	Must be unique in DB table
exists:table,column	Must exist in DB table
in:a,b,c	Must be one of listed values
confirmed	Requires matching _confirmation field
date | after:date	Date validation

Common Patterns

API Response

```
return response()->json(['data' => $users], 200);
return response()->json(['error' => 'Not found'], 404);
```

Environment & Config

```
env('APP_KEY');             // read .env value
config('app.name');         // read config value
config(['app.debug' => true]); // set at runtime
```

Useful Helpers

route('name', $params)	Generate URL for named route
redirect()->route('name')	Redirect to named route
back()->withErrors()	Redirect back with validation errors
abort(404)	Throw HTTP exception
collect($array)	Create a Collection from array
now()	Current Carbon datetime
cache()->remember()	Cache a value with TTL