Laravel API-Only Project Structure

Pure API backend with Sanctum auth and versioned routes. No Blade views.

#laravel #php #api #backend #sanctum #rest

PNG PDF Copy Prompt

Project Directory

```
myproject/
    artisan
    composer.json
    .env
    .env.example
  app/
    Http/
      Controllers/
        Controller.php
        Api/ Version namespa…
          V1/ API v1 controll…
            AuthController.php
            UserController.php
            ProductController.php
      Requests/
        Api/
          LoginRequest.php
          StoreUserRequest.php
      Resources/ API response tr…
        UserResource.php
        ProductResource.php
        UserCollection.php
      Middleware/
        ForceJsonResponse.php Always return J…
    Models/
      User.php HasApiTokens tr…
      Product.php
    Services/
      AuthService.php
      UserService.php
    Providers/
    Exceptions/
      Handler.php JSON error resp…
  bootstrap/
  config/
    sanctum.php Token auth conf…
    cors.php CORS settings
  database/
    migrations/
    seeders/
    factories/
  routes/
    api/ Versioned API r…
      v1.php /api/v1/*
    api.php Route loader
  tests/
    Feature/
      Api/
        AuthTest.php
        UserTest.php
    Unit/
  public/
  storage/
```

Why This Structure?

This structure strips Laravel down to a pure API backend. No Blade views, no frontend assets. Sanctum handles token authentication, API Resources transform responses, and versioned routes (`/api/v1/`) allow backward-compatible evolution.

Key Directories

app/Http/Controllers/Api/V1/ - Versioned API controllers

app/Http/Resources/ - Transform Eloquent models to JSON

routes/api/v1.php - V1 API routes, loaded in api.php

config/sanctum.php - Token expiration, domains, guards

app/Exceptions/Handler.php - Custom JSON error responses

Versioned API Routes

```
// routes/api.php
Route::prefix('v1')->group(base_path('routes/api/v1.php'))

// routes/api/v1.php
Route::post('login', [AuthController::class, 'login']);
Route::middleware('auth:sanctum')->group(function () {
    Route::apiResource('users', UserController::class);
});
```

Getting Started

1. `composer create-project laravel/laravel myproject`
2. `composer require laravel/sanctum`
3. `php artisan vendor:publish --provider="Laravel\Sanctum\SanctumServiceProvider"`
4. `php artisan migrate`
5. Add `HasApiTokens` trait to User model
6. `php artisan serve`

When To Use This

- Backend for SPA (React, Vue, mobile apps)
- Microservices architecture
- Third-party API consumption
- Mobile app backends
- Headless CMS backends

Trade-offs

No SSR - Need separate frontend deployment

More complexity - CORS, token management, API versioning

Testing overhead - Must test API contracts explicitly

Best Practices

- Always use API Resources for consistent responses
- Version your API from day one (`/api/v1/`)
- Use Form Requests for validation, not controller logic
- Return proper HTTP status codes (201, 204, 422)
- Implement rate limiting via `throttle` middleware