Laravel 치트시트

1. 설치 및 프로젝트 설정

- Composer를 통한 새 프로젝트 생성: `composer create-project laravel/laravel <project_name>`
- 개발 서버 실행: `php artisan serve`
- 환경 설정 파일: `.env`
- 애플리케이션 키 생성: `php artisan key:generate`

2. Artisan CLI

Laravel의 핵심 커맨드 라인 인터페이스.

- 명령어 목록 보기: `php artisan list`
- 컨트롤러 생성: `php artisan make:controller MyController`
- 모델 생성: `php artisan make:model Post -mfs`
  - `-m`: 마이그레이션 파일 생성
  - `-f`: 팩토리 파일 생성
  - `-s`: 시더 파일 생성
- 마이그레이션 실행: `php artisan migrate`
- 라우트 목록 보기: `php artisan route:list`
- Tinker (REPL): `php artisan tinker`

3. 라우팅 (Routing)

`routes/web.php` (웹) 및 `routes/api.php` (API) 파일에 정의.

- 기본 라우트:

```
use App\Http\Controllers\PostController;

Route::get('/posts', [PostController::class, 'index']);
Route::post('/posts', [PostController::class, 'store']);
```

- 리소스 컨트롤러 라우트: `Route::resource('photos', PhotoController::class);`
- 라우트 파라미터: `Route::get('/posts/{id}', function ($id) { ... });`
- 이름 있는 라우트: `Route::get('/', ...)->name('home');`

4. 컨트롤러 (Controllers)

`app/Http/Controllers` 디렉토리에 위치.

```
namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\Post;

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

    public function show(Post $post) // 라우트 모델 바인딩
    {
        return view('posts.show', ['post' => $post]);
    }
}
```

5. 블레이드 템플릿 (Blade)

Laravel의 강력한 템플릿 엔진.

- 변수 출력: `{{ $variable }}`
- 주석: `{{-- This is a comment. --}}`
- 제어 구조:
  - `@if`, `@elseif`, `@else`, `@endif`
  - `@foreach ($users as $user) ... @endforeach`
  - `@forelse ($users as $user) ... @empty ... @endforelse`
  - `@auth`, `@guest`
- 템플릿 상속:
  - `@extends('layouts.app')`
  - `@section('content') ... @endsection`
  - `@yield('title')`
- 컴포넌트 포함: `@include('partials.header')`

6. 엘로퀀트 ORM (Eloquent)

데이터베이스 작업을 위한 Active Record 구현체.

- 모델 정의 (`app/Models/Post.php`):

```
class Post extends Model
{
    use HasFactory;
    protected $fillable = ['title', 'content']; // 대량 할당 허용
}
```

- 모든 레코드 조회: `Post::all()`
- 레코드 찾기: `Post::find(1)`
- 조건부 조회: `Post::where('active', 1)->orderBy('created_at', 'desc')->get();`
- 레코드 생성: `Post::create(['title' => 'New', 'content' => 'Content']);`
- 레코드 업데이트:

```
$post = Post::find(1);
$post->title = 'Updated';
$post->save();
```

- 레코드 삭제: `$post->delete();`

7. 관계 (Relationships)

- 1:N (일대다): `hasMany()`, `belongsTo()`
- N:N (다대다): `belongsToMany()`
- 1:1 (일대일): `hasOne()`, `belongsTo()`

```
// User.php
public function posts()
{
    return $this->hasMany(Post::class);
}

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

- 관계 데이터 로딩:
  - Eager Loading: `User::with('posts')->get();`
  - Lazy Loading: `$user->posts`

8. 미들웨어 (Middleware)

HTTP 요청을 필터링하는 메커니즘.

- 미들웨어 생성: `php artisan make:middleware CheckAge`
- `app/Http/Kernel.php`에 미들웨어 등록.

9. 폼 요청 및 유효성 검사

- 폼 요청 생성: `php artisan make:request StorePostRequest`
- `app/Http/Requests/StorePostRequest.php`:

```
public function rules()
{
    return [
        'title' => 'required|unique:posts|max:255',
        'content' => 'required',
    ];
}
```

- 컨트롤러에서 사용:

```
public function store(StorePostRequest $request)
{
    // 유효성 검사 통과
    Post::create($request->validated());
    return redirect()->route('posts.index');
}
```

- 블레이드에서 에러 표시:

```
@error('title')
    <div class="alert alert-danger">{{ $message }}</div>
@enderror
```