



Larvel – ett fullskaligt
MVC-ramverk

Inlämningsuppgift 2

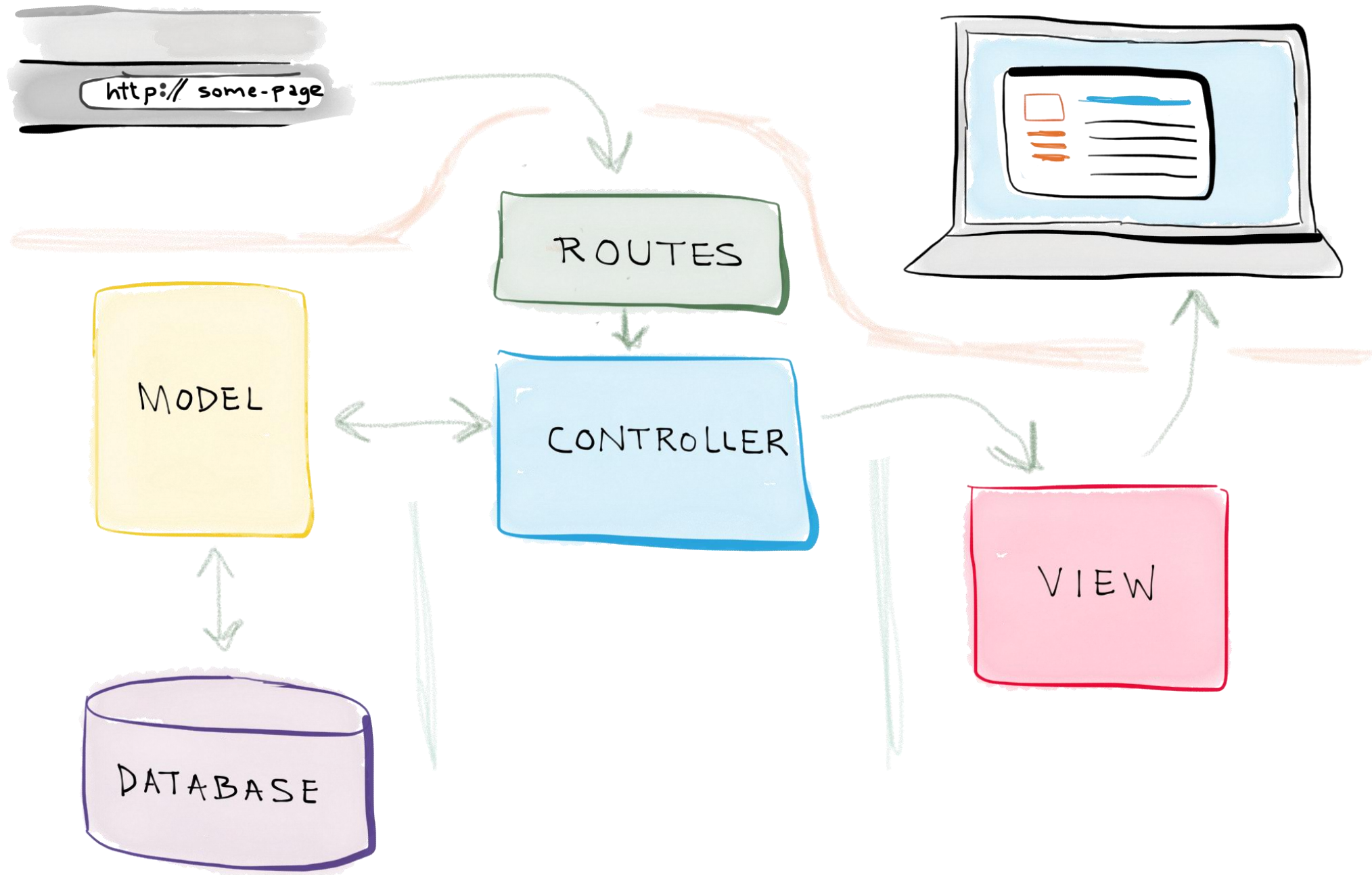
<https://github.com/mah-dv/da288a-vt18/blob/master/Assignments/2/assignment.md>

Dagens agenda

- Repetition – Vad har vi gjort hittills?
- Laravel vs. Lumen?
- Nya delar i laravel
 - Mer hjälp av PHP-artisan
 - Vyer (Views)
- Templates
- Att arbeta med HTML / JS / CSS i Laravel

Repetition

Lumen, etc.



demo

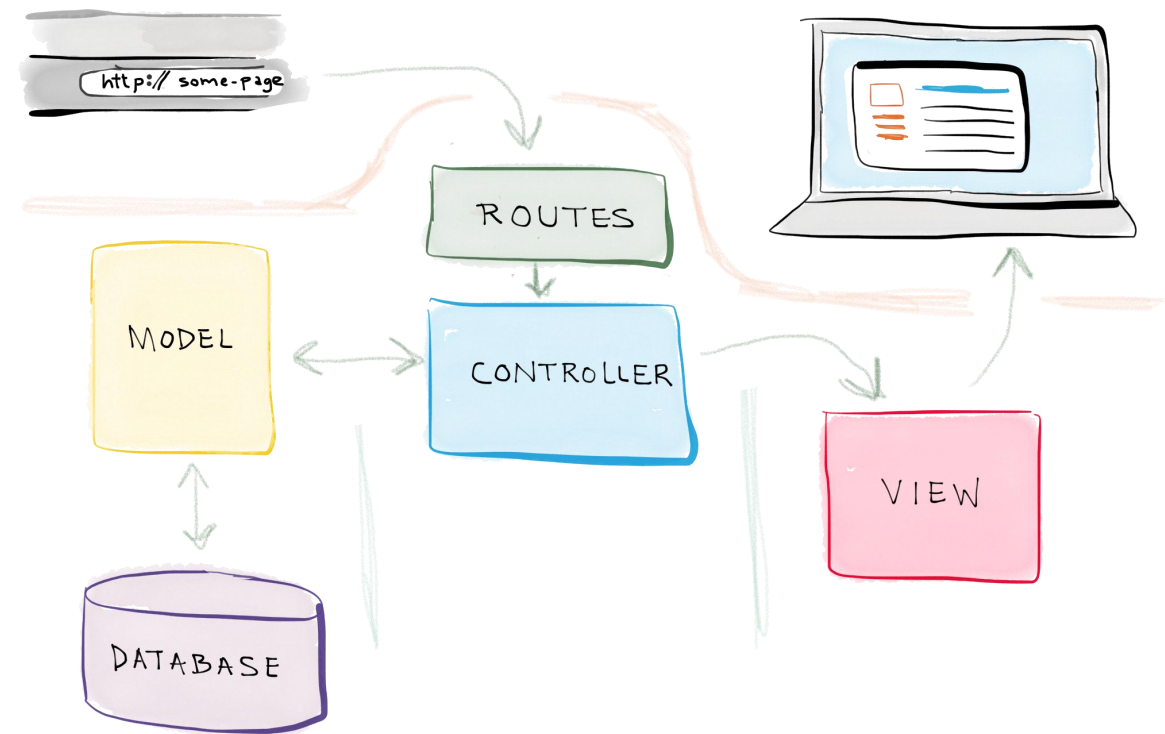
Arkiv Redigera Visa Gå Bokmärken Hjälp

Bakåt Framåt 50% Listvy

Platser student Projekt demo


Dator	Namn	Storlek	Typ	Ändringsdatum
student	app	8 objekt	mapp	tis 24 jan 2017 18:04:17
Skrivbord	Console	2 objekt	mapp	tis 24 jan 2017 18:04:17
Filsystem	Events	2 objekt	mapp	tis 24 jan 2017 18:04:17
Dokument	Exceptions	1 objekt	mapp	tis 24 jan 2017 18:04:17
Hämtningar	Http	2 objekt	mapp	tis 24 jan 2017 18:04:17
Musik	Controllers			
Bilder	Controller.php			
Video	ExampleController.php	236 byte	PHP-skript	tis 24 jan 2017 18:04:17
Papperskorg	Middleware	2 objekt	mapp	tis 24 jan 2017 18:04:17
Nätverk	Jobs	2 objekt	mapp	tis 24 jan 2017 18:04:17
Bläddra i nä...	Listeners	1 objekt	mapp	tis 24 jan 2017 18:04:17
	Providers	2 objekt	mapp	tis 24 jan 2017 18:04:17
	User.php			
	bootstrap			
	database	3 objekt	mapp	tis 24 jan 2017 18:04:17
	public	1 objekt	mapp	tis 24 jan 2017 18:04:17
	resources	1 objekt	mapp	tis 24 jan 2017 18:04:17
	routes	1 objekt	mapp	tis 24 jan 2017 18:04:17
	web.php			
	storage			
	tests	2 objekt	mapp	tis 24 jan 2017 18:04:17
	vendor	23 objekt	mapp	tor 6 apr 2017 15:58:53
	artisan	1,1 kB	PHP-skript	tis 24 jan 2017 18:04:17
	composer.json	817 byte	JSON-dokument	tis 24 jan 2017 18:04:17

14 objekt, ledigt utrymme: 1,4 GB



Routes => Controller => Response

```
1 <?php
2
3 use App\Http\Controllers\MoviesController;
4
5 $router->get('/', function () {
6     return "Welcome to my movie API!";
7 });
8
9 $router->get('/movies', 'MoviesController@index');
10 $router->post('/movies', 'MoviesController@create');
11 $router->get('/movies/{id}', 'MoviesController@show');
12 $router->put('/movies/{id}', 'MoviesController@update');
13 $router->delete('/movies/{id}', 'MoviesController@delete');
14
```



```
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6
7 class MoviesController extends Controller
8 {
9     /**
10      * Create a new controller instance.
11      *
12      * @return void
13      */
14     public $movies;
15
16     public function __construct()
17     {
18         $this->movies = json_decode(file_get_contents("../resources/movies.json"));
19     }
20
21     public function index() {
22         return response()->json($this->movies);
23     }
24
25     public function show($id) {
26         foreach($this->movies as $movie) {
27             if($movie->id == $id) {
28                 return response()->json($movie);
29             }
30         }
31     }
32
33     public function create(Request $request) {
34         $movie = [];
35         $movie['title'] = $request->input("title");
36         $movie['id'] = $request->input("id");
37         $movie['grade'] = $request->input("grade");
38
39         $movies = $this->movies;
40         $movies[] = $movie;
41
42         file_put_contents("../resources/movies.json", json_encode($movies));
43
44         return response()->json($movies);
45     }
46 }
```

En controllers uppgift

- Sköter logiken för vår applikation
- Tar emot anropet
 - Och ev. användardata
- Hanterar anropet
 - T.ex. hämta / skriva lämplig data i vår tjänst
 - Databas (MySQL, etc.)
 - Icke-relationella databaser
- Returnera ett svar
 - HTML-svar för en webbplats
 - JSON-svar för vårt API

```
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class MoviesController extends Controller
8  {
9      /**
10       * Create a new controller instance.
11       *
12       * @return void
13       */
14      public $movies;
15
16      public function __construct()
17      {
18          $this->movies = json_decode(file_get_contents("../resources/movies.json"));
19      }
20
21      public function index() {
22          return response()->json($this->movies);
23      }
24
25      public function show($id) {
26          foreach($this->movies as $movie) {
27              if($movie->id == $id) {
28                  return response()->json($movie);
29              }
30          }
31      }
32
33      public function create(Request $request) {
34          $movie = [];
35          $movie['title'] = $request->input("title");
36          $movie['id'] = $request->input("id");
37          $movie['grade'] = $request->input("grade");
38
39          $movies = $this->movies;
40          $movies[] = $movie;
41
42          file_put_contents("../resources/movies.json", json_encode($movies));
43
44          return response()->json($movies);
45      }
46  }
```


Versionshantering av databas

- Migrations

Mycket smidigt när man är många i projektet!

Konfigurera vår databas: .env



Vår konfiguration

```
Program Platser System
.env — ~/Skrivbord/Labs/db-test — Atom
File Edit View Selection Find Packages Help

db-test
├── app
├── bootstrap
├── database
├── public
├── resources
├── routes
├── storage
├── tests
├── vendor
├── .env
├── .env.example
├── .gitignore
├── artisan
├── composer.json
├── composer.lock
├── phpunit.xml
└── readme.md
```

```
1 APP_ENV=local
2 APP_DEBUG=true
3 APP_KEY=
4 APP_TIMEZONE=UTC
5
6 DB_CONNECTION=mysql
7 DB_HOST=127.0.0.1
8 DB_PORT=3306
9 DB_DATABASE=movies
10 DB_USERNAME=root
11 DB_PASSWORD=supersecret
12
13 CACHE_DRIVER=memcached
14 QUEUE_DRIVER=sync
15
```

php artisan

Ett hjälpmedel för att snabba upp utvecklingen av webbapplikationer

Skapa en migration

Migration-funktioner

```
student@php-dev: ~/Skrivbord/Labs/db-test
Arkiv Redigera Visa Sök Terminal Hjälp

Available commands:
help           Displays help for a command
list           Lists commands
migrate        Run the database migrations
auth
auth:clear-resets  Flush expired password reset tokens
cache
cache:clear      Flush the application cache
cache:forget     Remove an item from the cache
cache:table      Create a migration for the cache database table
db
db:seed         Seed the database with records
make
make:migration   Create a new migration file
make:seeder      Create a new seeder class
migrate
migrate:install  Create the migration repository
migrate:refresh  Reset and re-run all migrations
migrate:reset    Rollback all database migrations
migrate:rollback Rollback the last database migration
migrate:status   Show the status of each migration
queue
queue:failed     List all of the failed queue jobs
queue:failed-table Create a migration for the failed queue jobs database table
queue:flush      Flush all of the failed queue jobs
queue:forget     Delete a failed queue job
queue:listen     Listen to a given queue
queue:restart    Restart queue worker daemons after their current job
queue:retry      Retry a failed queue job
queue:table      Create a migration for the queue jobs database table
queue:work       Start processing jobs on the queue as a daemon
schedule
schedule:run     Run the scheduled commands
student@php-dev:~/Skrivbord/Labs/db-test$ php artisan
```

Vår migration-fil

```
> app
> bootstrap
▼ database
  > factories
  ▼ migrations
    .gitkeep
    2017_04_18_154231_create_movies_table.php
  > seeds
> public
> resources
> routes
> storage
> tests
> vendor
.env
.env.example
.gitignore
artisan
composer.json
composer.lock
phpunit.xml
readme.md
```

```
1  <?php
2
3  use Illuminate\Support\Facades\Schema;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Database\Migrations\Migration;
6
7  class CreateMoviesTable extends Migration
8  {
9      /**
10       * Run the migrations.
11       *
12       * @return void
13       */
14     public function up()
15     {
16         Schema::create('movies', function (Blueprint $table) {
17             $table->increments('id');
18             $table->text('title');
19             $table->integer('year');
20             $table->text('poster');
21             $table->timestamps();
22         });
23     }
24
25     /**
26      * Reverse the migrations.
27      *
28      * @return void
29      */
30     public function down()
31     {
32         Schema::drop('movies');
33     }
34 }
```

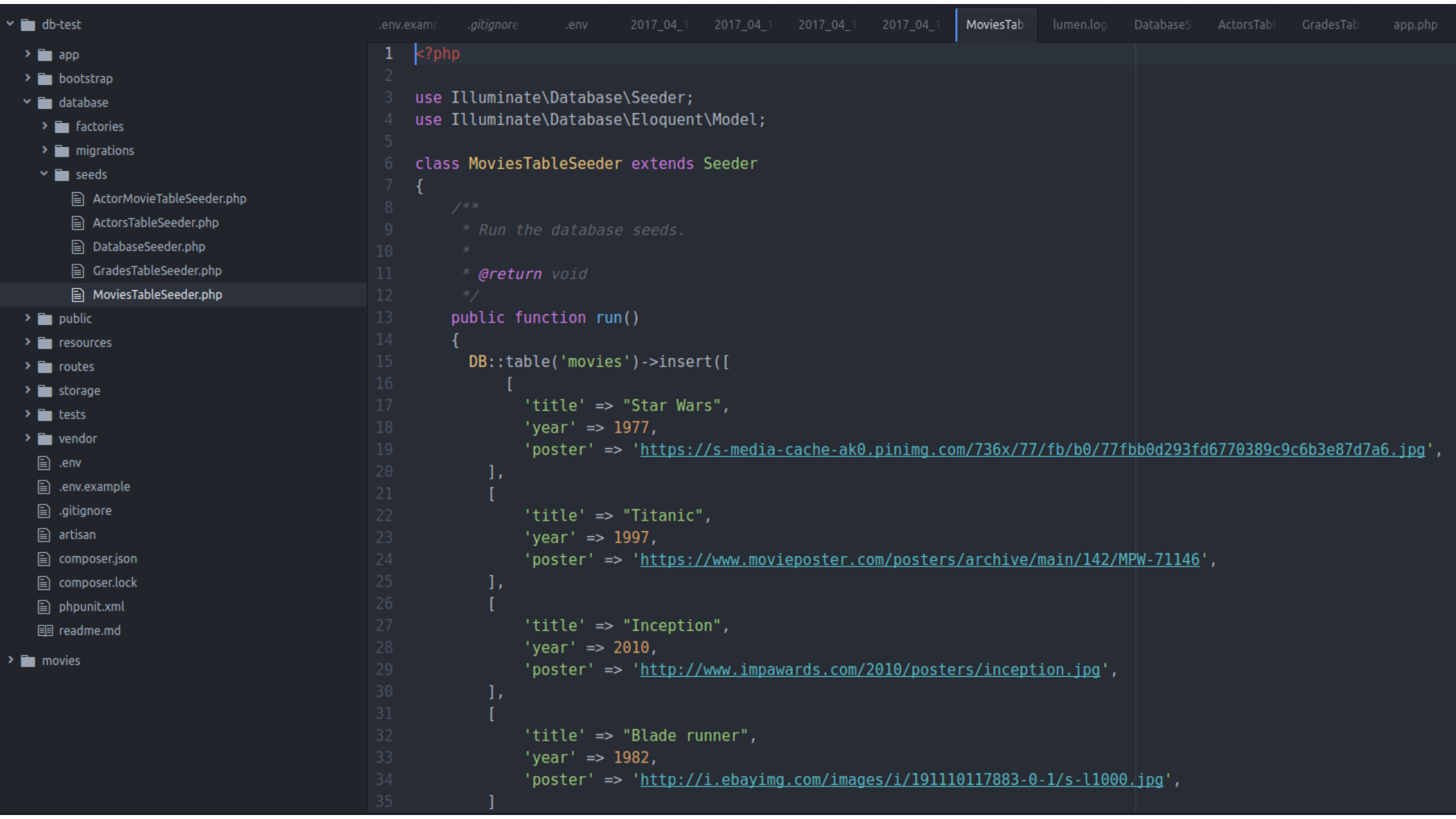
Seeds! Lägg in demodata i
databasen!

Lägg in seeds i db

Skapa en seeder

```
student@php-dev: ~/Skrivbord/Labs/db-test
Arkiv Redigera Visa Sök Terminal Hjälp

Available commands:
  help           Displays help for a command
  list           Lists commands
  migrate        Run the database migrations
  auth
  auth:clear-resets  Flush expired password reset tokens
  cache
  cache:clear     Flush the application cache
  cache:forget    Remove an item from the cache
  cache:table     Create a migration for the cache database table
  db
  db:seed         Seed the database with records
  make
  make:migration  Create a new migration file
  make:seeder     Create a new seeder class
  migrate
  migrate:install Create the migration repository
  migrate:refresh  Reset and re-run all migrations
  migrate:reset    Rollback all database migrations
  migrate:rollback Rollback the last database migration
  migrate:status   Show the status of each migration
  queue
  queue:failed     List all of the failed queue jobs
  queue:failed-table Create a migration for the failed queue jobs database table
  queue:flush      Flush all of the failed queue jobs
  queue:forget     Delete a failed queue job
  queue:listen     Listen to a given queue
  queue:restart    Restart queue worker daemons after their current job
  queue:retry      Retry a failed queue job
  queue:table      Create a migration for the queue jobs database table
  queue:work       Start processing jobs on the queue as a daemon
  schedule
  schedule:run     Run the scheduled commands
student@php-dev:~/Skrivbord/Labs/db-test$ php artisan
```

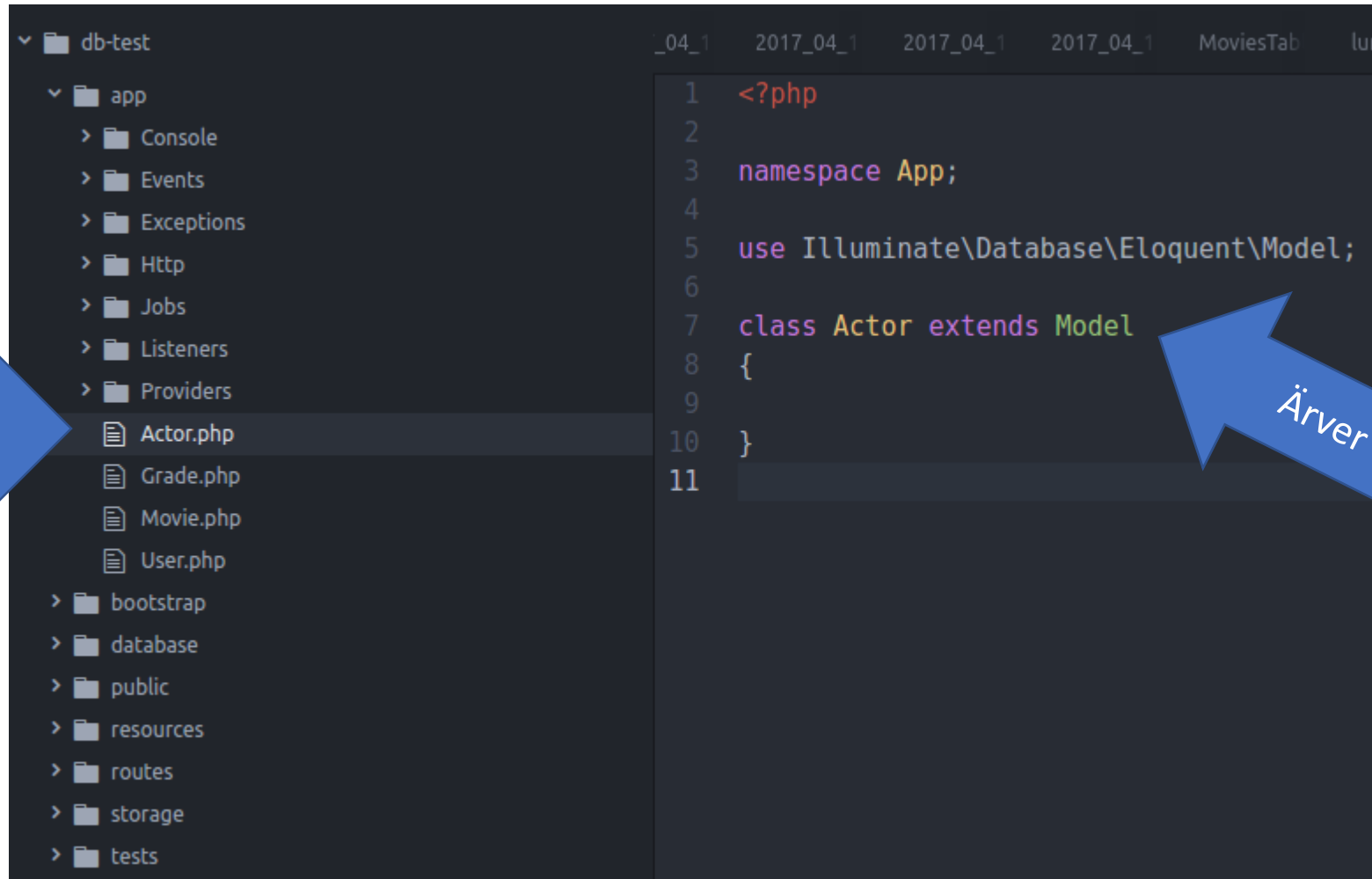


*”ORM, står för **object-relational mapping**, är ett objektorienterat system som konverterar databastabeller till klasser, tabellrader till objekt, samt celler till objekt-attribut”*

En model är en...

- Representation av en resurs
- I Lumen så är dessa väldigt ofta (men inte alltid) kopplade till en databastabell
- Alltså, eftersom vi har 3st tabeller kommer vi skapa 3st modeller
 - Movies
 - Actors
 - Grades
- Vi har även en många till många-relation som är en egen tabell & således en egen modell (om vi vill!)
 - ActorMovie

Var finns modellerna?



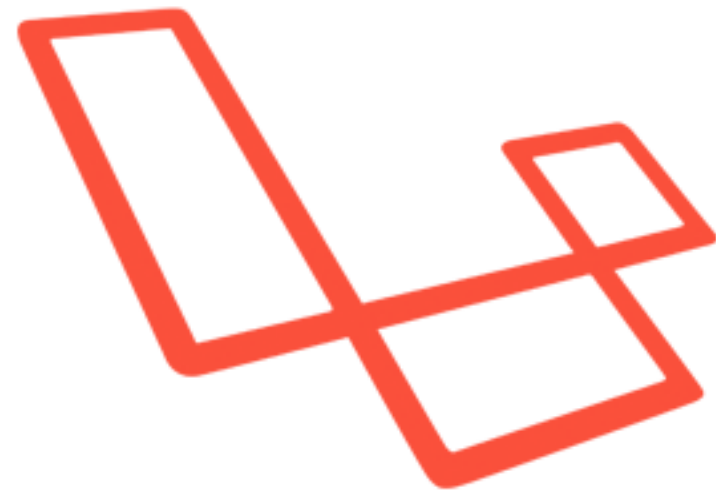
The screenshot shows a code editor with a dark theme. On the left, a file explorer displays the directory structure of a project named 'db-test'. The 'app' directory is expanded, showing subdirectories like 'Console', 'Events', 'Exceptions', 'Http', 'Jobs', 'Listeners', and 'Providers', as well as files 'Actor.php', 'Grade.php', 'Movie.php', and 'User.php'. A large blue arrow points from the left towards the 'Actor.php' file. On the right, the code editor shows the contents of 'Actor.php'. The code is as follows:

```
1 <?php
2
3 namespace App;
4
5 use Illuminate\Database\Eloquent\Model;
6
7 class Actor extends Model
8 {
9
10 }
11
```

A blue arrow points from the text 'Ärver av Model' (Inherits from Model) to the 'extends Model' part of the class declaration on line 7.

Bra att veta om modeller

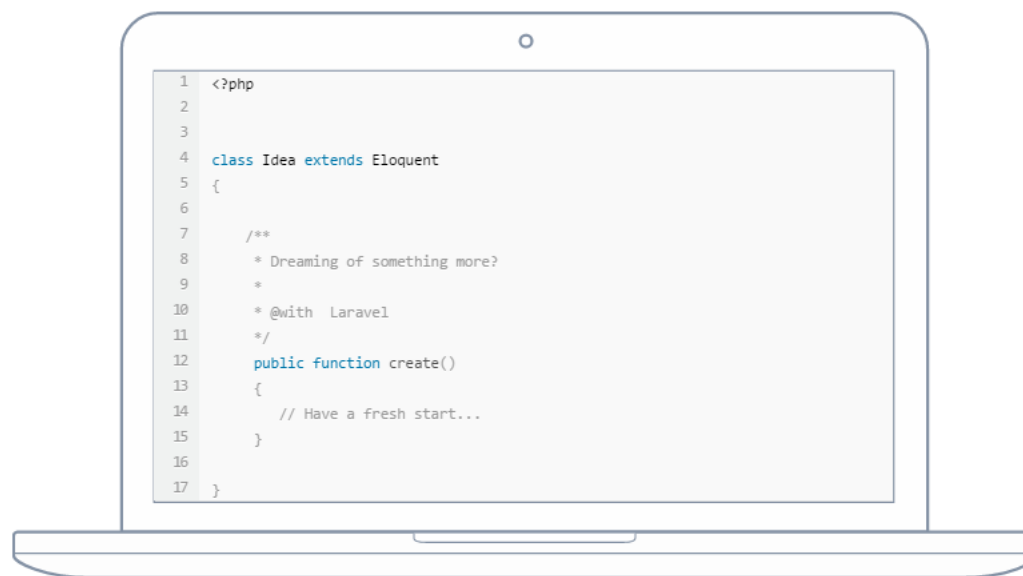
- Mappas mot en tabell i databasen, fast i plural. T.ex.
 - Modellen "Movie" mappas mot tabellen "movies" i databasen
 - Kan ändras genom egenskapen (i modellen):
 - `protected $table = 'my_movies';`
- Andra bra egenskaper kan ni läsa mer om här:
<https://laravel.com/docs/5.6/eloquent> , t.ex.
 - Vilka värden som får fyllas i
 - Vilken databasanslutning som gäller (om man har flera)
 - Om man ska använda "softDelete", m.m.



laravel

Love beautiful code? We do too.

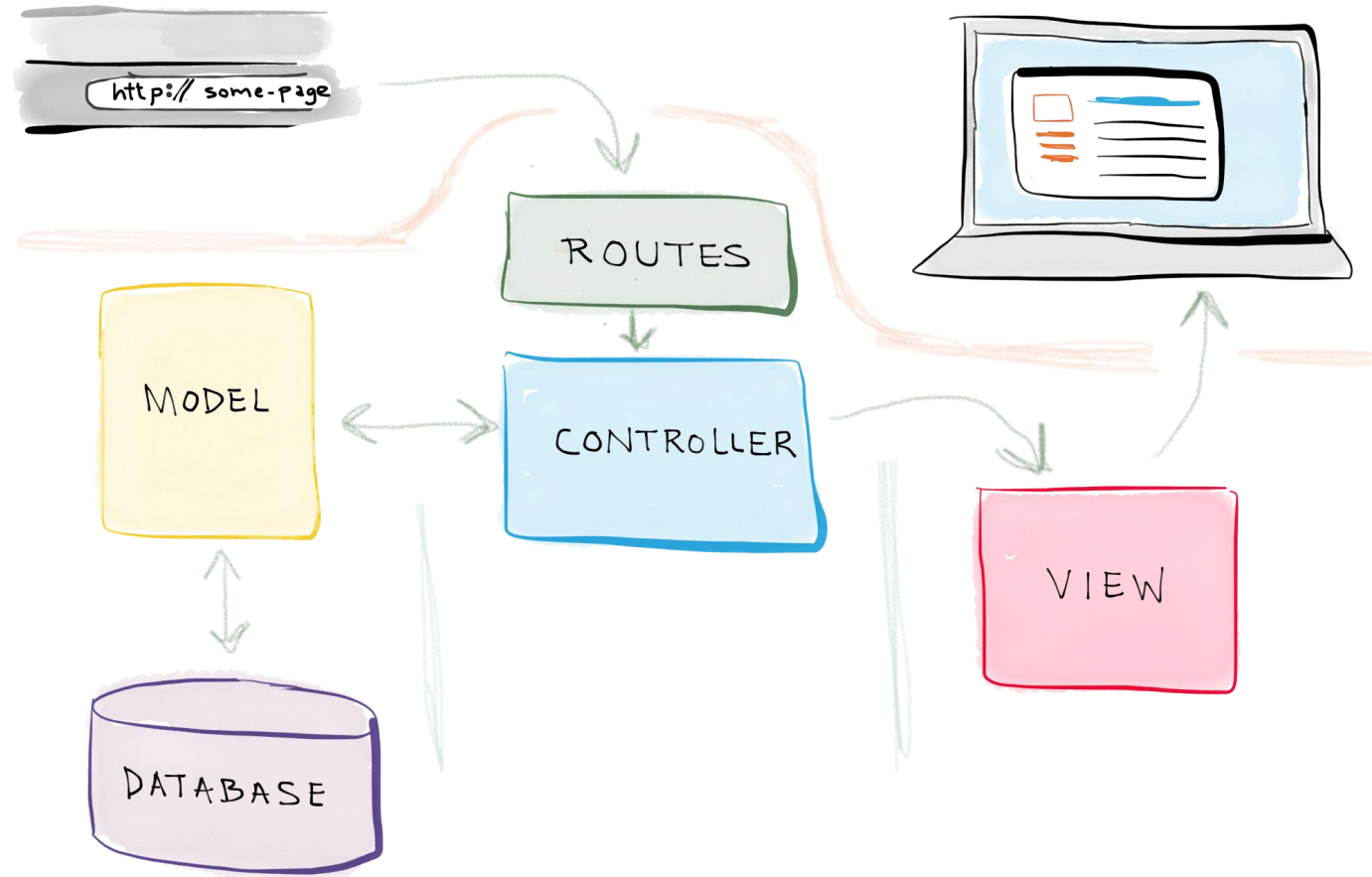
The PHP Framework For Web Artisans



SEE WHAT'S NEW!



<https://laravel.com/>



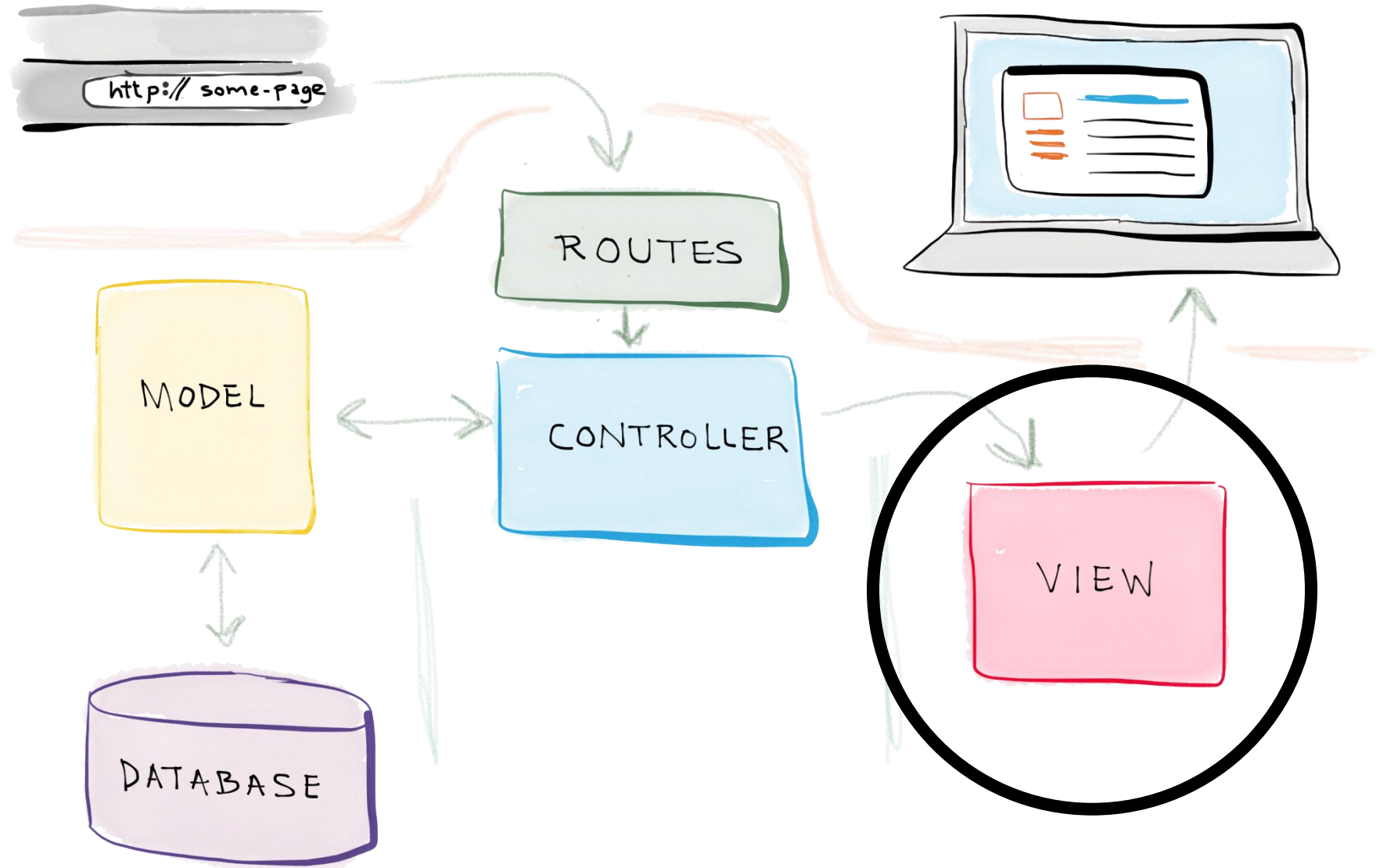
Laravel = Lumen + views

Eller nja, inte hela sanningen... men ett stort användningsområde

Skillnader mellan Lumen och Laravel

- Lumen är en "light"-version av Laravel
 - Saknar vyer
 - Saknar sessioner
 - Lumen fokuserar således på *tillståndslösa APIr* medan Laravel fokuserar på kompletta webbapplikationer
- Laravel är ett mycket stort, och ibland komplext ramverk
 - <https://laravel.com/docs/5.6>

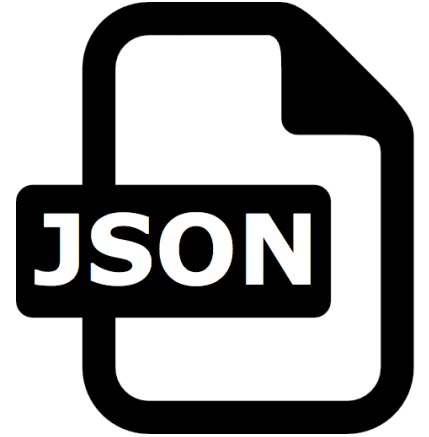
Vyer



Lumen =>

Routes
Controllers
Models
Database

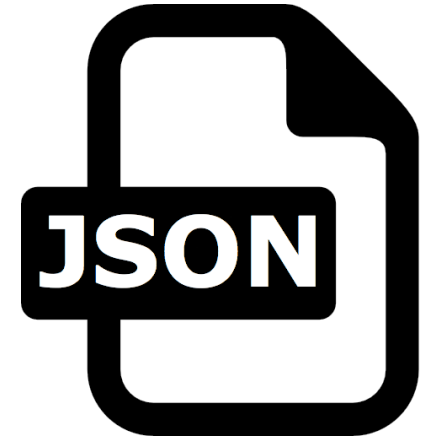
=> JSON-data



Lumen =>

Routes
Controllers
Models
Database

=> JSON-data



Laravel =>

Routes
Controllers
Models
Database

=> Webbbsida



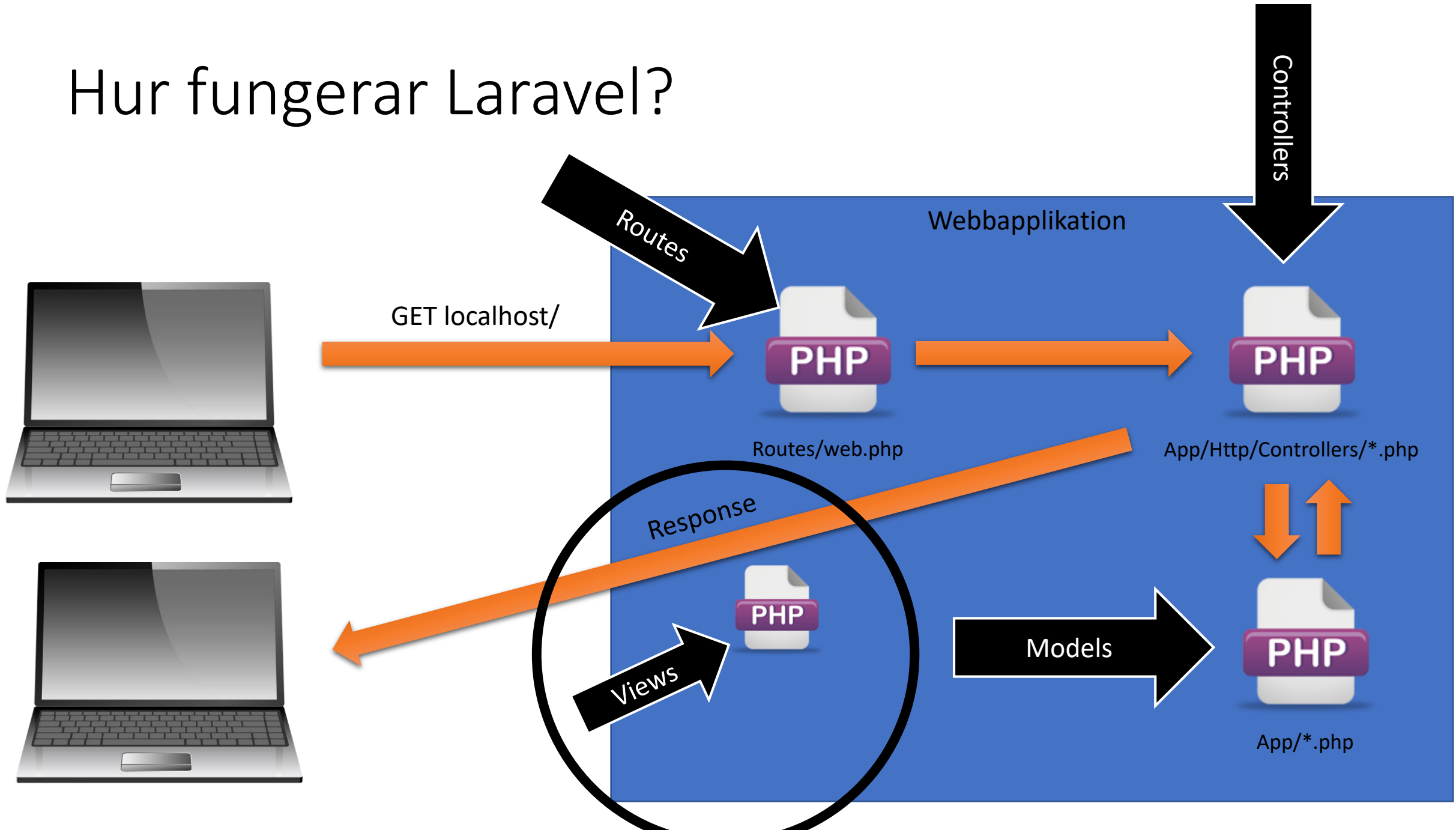
Lumen – Svar på ett anrop => JSON

```
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use App\Movie;
6  use Illuminate\Support\Facades\DB;
7
8  class MoviesController extends Controller
9  {
10     public function index(){
11         $movies = Movie::all();
12         return response()->json($movies);
13     }
14
15     public function show($id){
16         $movie = Movie::find($id);
17         $movie->actors = $movie->actors;
18         return response()->json($movie);
19     }
20 }
```

JSON-svar

JSON-svar

Hur fungerar Laravel?



En vy renderar ett
användargränssnitt en modell (eller
en kombination av modeller)

Routes i Laravel

```
1  <?php
2
3  /*
4  |-----|
5  | Web Routes
6  |-----|
7  |
8  | Here is where you can register web routes for your application. These
9  | routes are loaded by the RouteServiceProvider within a group which
10 | contains the "web" middleware group. Now create something great!
11 |
12 */
13
14 // Skickar vidare anropet till en controller
15 Route::get('/', 'MoviesController@index');
16
17 // Returnerar vyn "Welcome"
18 Route::get('/', function () {
19     return view('welcome');
20 });
21
```

Basic Routing

The most basic Laravel routes simply accept a URI and a `Closure`, providing a very simple and expressive method of defining routes:

```
Route::get('foo', function () {  
    return 'Hello World';  
});
```

The Default Route Files

All Laravel routes are defined in your route files, which are located in the `routes` directory. These files are automatically loaded by the framework. The `routes/web.php` file defines routes that are for your web interface. These routes are assigned the `web` middleware group, which provides features like session state and CSRF protection. The routes in `routes/api.php` are stateless and are assigned the `api` middleware group.

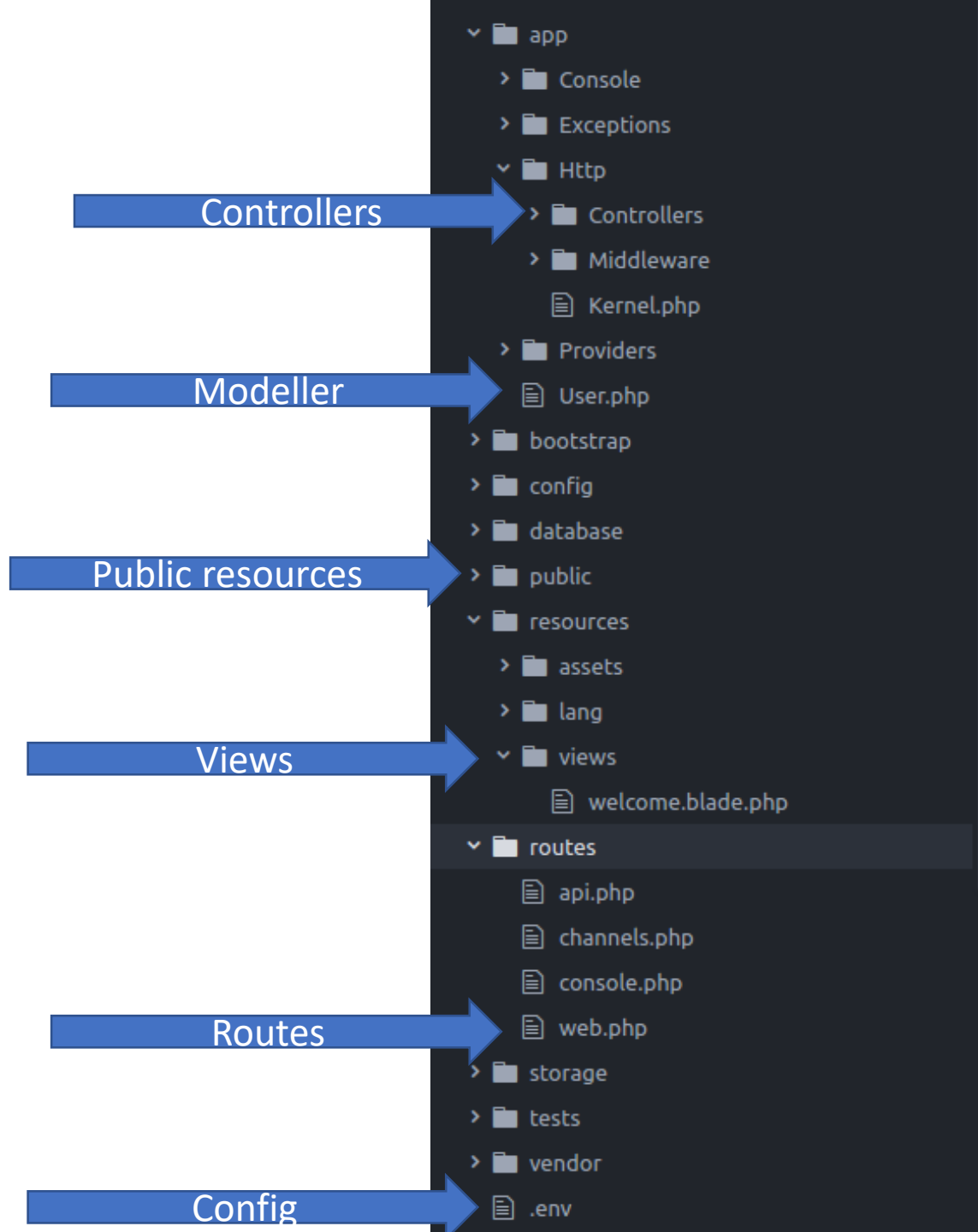
For most applications, you will begin by defining routes in your `routes/web.php` file.

Available Router Methods

The router allows you to register routes that respond to any HTTP verb:

```
Route::get($uri, $callback);  
Route::post($uri, $callback);  
Route::put($uri, $callback);  
Route::patch($uri, $callback);  
Route::delete($uri, $callback);  
Route::options($uri, $callback);
```

<https://laravel.com/docs/5.6/routing>



Dagens schema för "demo"

- Skapa ett nytt Laravelprojekt (<https://laravel.com/docs/5.6/installation>)
- Skapa migration & seeds (<https://laravel.com/docs/5.6/migrations> <https://laravel.com/docs/5.6/seeding>)
- Skapa routes vi behöver (<https://laravel.com/docs/5.6/routing>)
 - Enskilda routes
 - Routes till resurser
- Ska modeller & Controllers (<https://laravel.com/docs/5.6/controllers>)
- Göra vyer (templates) (<https://laravel.com/docs/5.6/blade>)
- Skicka med data till vyerna (<https://laravel.com/docs/5.6/blade#displaying-data>)
- Använda CSS / JS i våra vyer
- Vad innebär CSRF (Cross-site request forgery) (<https://laravel.com/docs/5.6/csrf>)
- Bygga vårt GUI
- Redirects inom vår applikation? (<https://laravel.com/docs/5.6/responses#redirects>)
- Klart!?
- Videos på ovan: <https://laracasts.com/>