Laravel Workshop

with MOHD AMIRUL ADLI

UTHM
Universiti Tun Hussein Onn Malaysia

About me

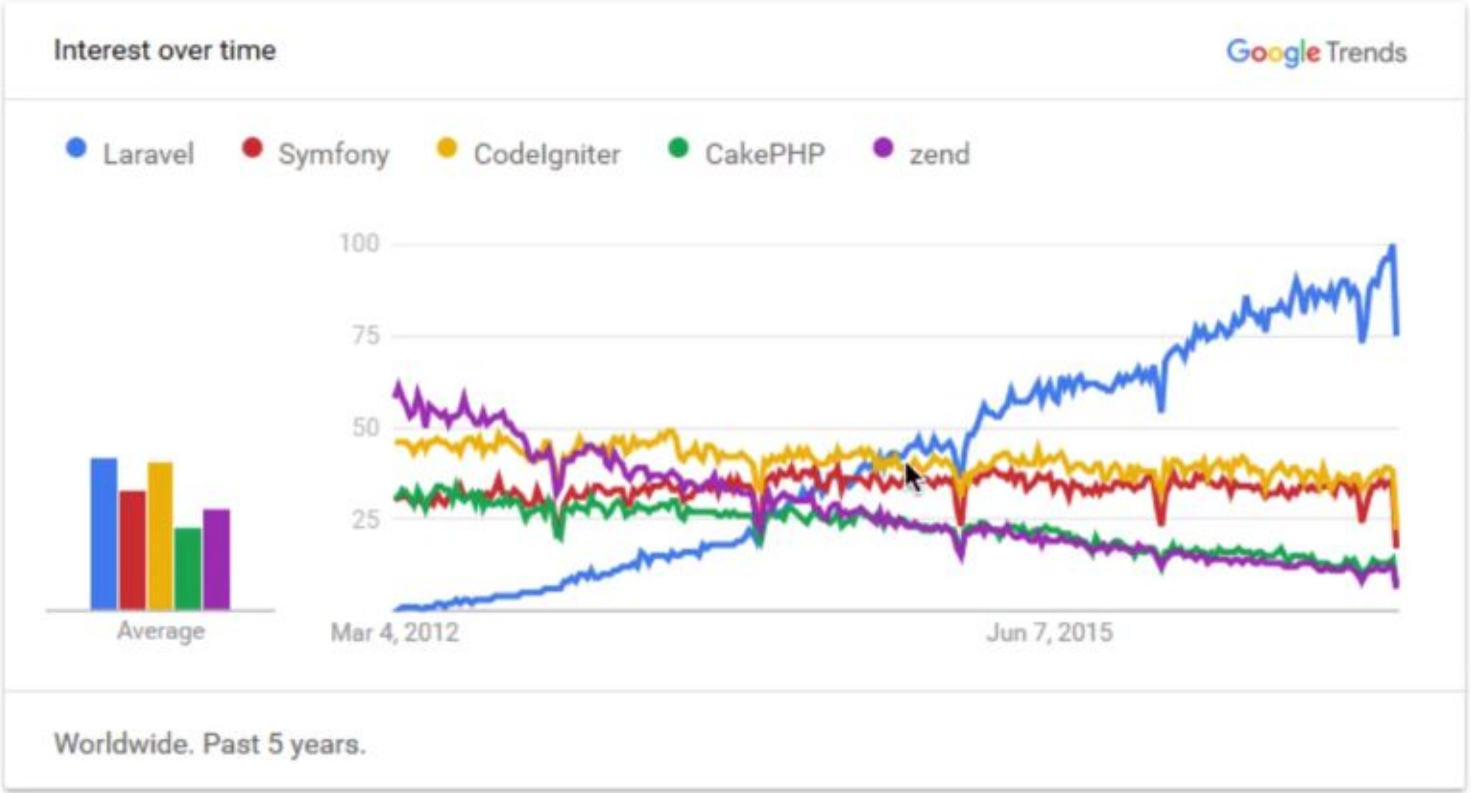
Amirul Adli

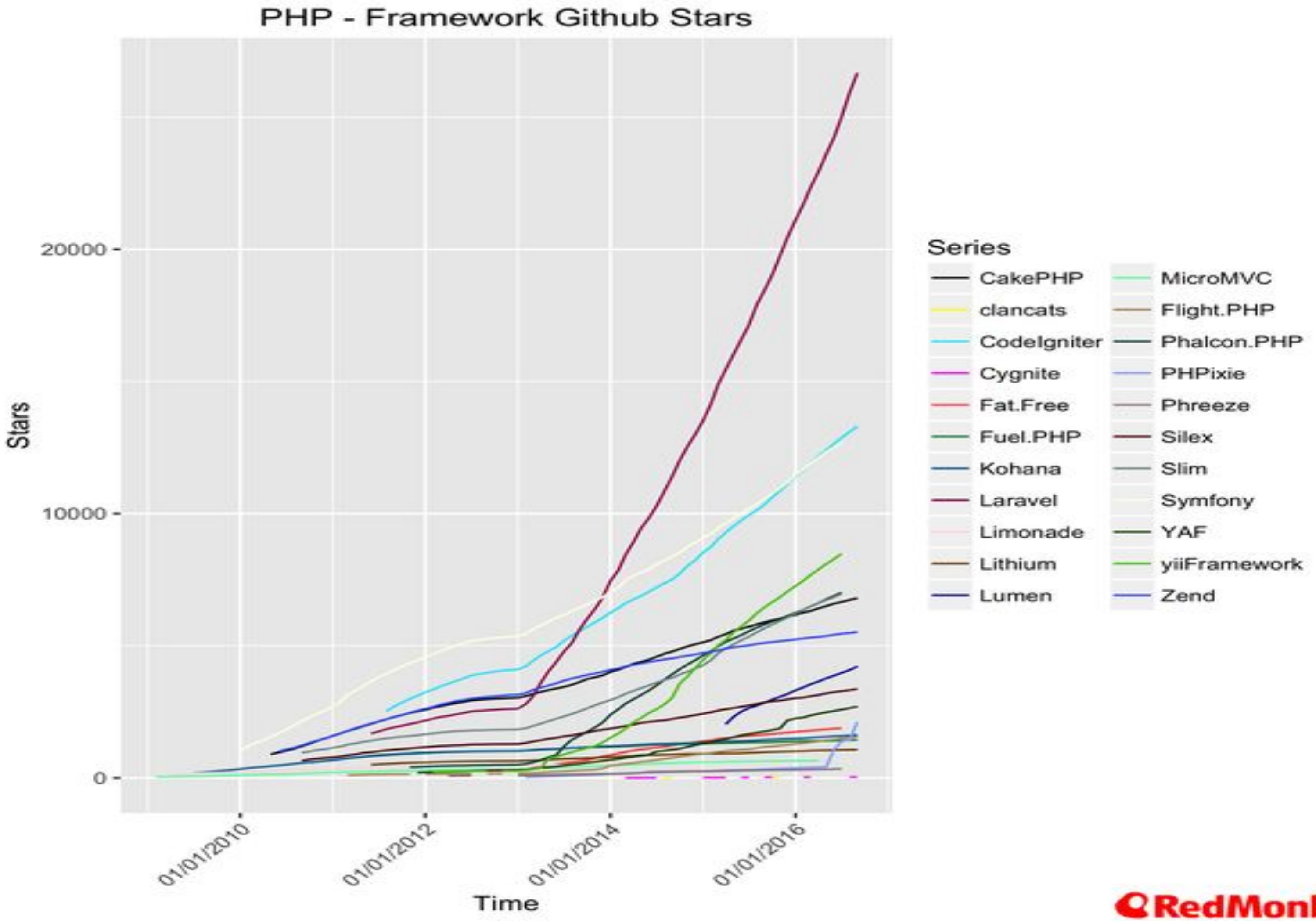
Full Stack Developer

Laravel enthusiast with 4 years experience and have used the framework for many projects.

Developer at **Vortechron** and **Baituljannah**

Why Laravel


Interest over time
Google Trends
Laravel
Symfony
CodeIgniter
CakePHP
zend
100
75
50
25
Average
Mar 4, 2012
Jun 7, 2015
Worldwide. Past 5 years.

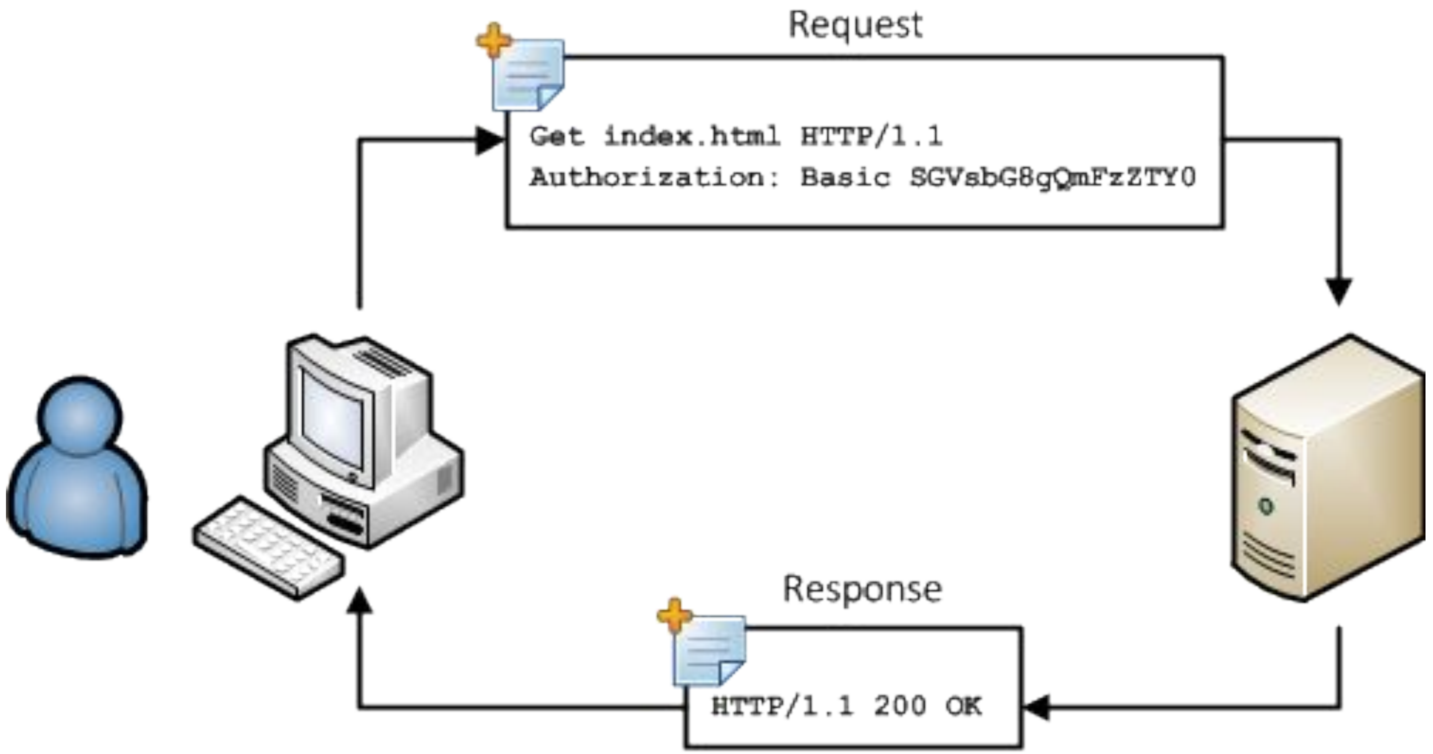

PHP - Framework Github Stars
Stars
20000
10000
0
01/01/2010
01/01/2012
01/01/2014
01/01/2016
Time
Series
CakePHP
clancats
CodeIgniter
Cygnite
Fat.Free
Fuel.PHP
Kohana
Laravel
Limonade
Lithium
Lumen
MicroMVC
Flight.PHP
Phalcon.PHP
PHPixie
Phreeze
Silex
Slim
Symfony
YAF
yiiFramework
Zend

RedMonk

Take cares of the following pain of a developer.

1. Building an Authentication and Authorization Systems
2. Integration with Mail Services
3. Integration with Tools for Making Web Applications Faster
4. Fixing the Most Common Technical Vulnerabilities
5. Configuration Error and Exception Handling
6. Automation Testing Work
7. URL Routing Configuration
8. Separation “Business Logic Code” from “Presentation Code”
9. Message Queue System (Delayed Delivery) Configuration
10. Scheduling Tasks Configuration and Management

How web works?



```
Get index.html HTTP/1.1
Authorization: Basic SGVsbG8gQmFzZTY0
```

```
HTTP/1.1 200 OK
```

Get php install

1. Install **WAMPP**/MAMP/XAMPP
2. Install **GitBash**
3. Check if everything up and running
   a. Run “php -v” in terminal
4. Goto: https://goo.gl/9Hk1wF

Code editor

1. Available Tools
    a. Sublime
    b. **VS Code** (recommended)
    c. Atom
    d. Bracket

Variables

- In PHP, a variable starts with the $ sign, followed by the name of the variable.
- Sample:
    a. $name

PHP AND HTML

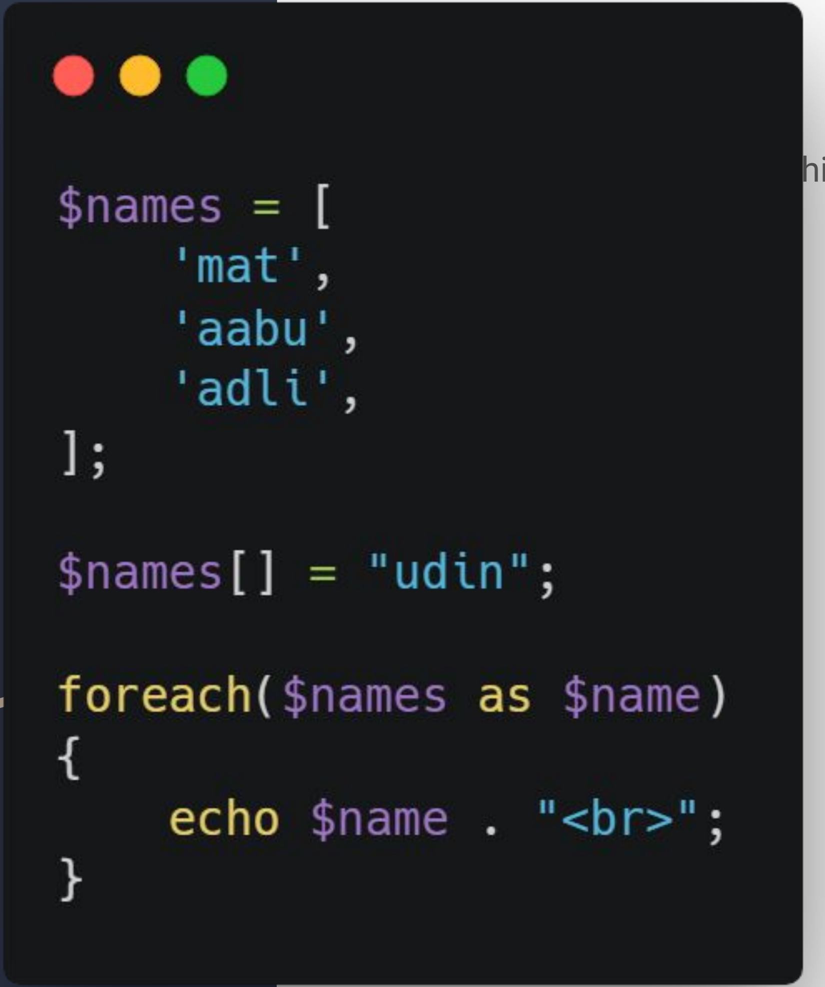
-> DEMO

Separate PHP Logic From Presentation

1. A **GOOD PRACTICE**
2. Avoid clutter

Array

hings (variable)



```
$names = [
    'mat',
    'aabu',
    'adli',
];

$names[] = "udin";

foreach($names as $name)
{
    echo $name . "<br>";
}
```

Boolean

1. Simplest type
2. A boolean expresses a truth value
3. It can be either **TRUE** or **FALSE**

Conditional statement

1. **if -**
   a. executes some code if one condition is true
2. **if...else -**
   a. executes some code if a condition is true and another code if that condition is false
3. **if...elseif....else -**
   a. executes different codes for more than two conditions
4. **switch -**
   a. selects one of many blocks of code to be executed

FUNCTIONS

1. 1000 built-in functions
2. Can be used repeatedly
3. You need to **call** the function
4. Sample:

```
function functionName() {

    code to be executed;

}
```

DATABASE

1. Available tools
    a. **MySQL**
    b. NoSQL
    c. SQLite
    d. MongoDB

CLASSES

1. Overwhelming topic
   http://php.net/manual/en/language.oop5.php
2. Focus
   a. Syntax
      i. **class** ClassName {}
      ii. **new** ClassName();
      iii. **$this**
   b. Naming class (Noun)

Forms and request

1. superglobals **$_GET** and **$_POST** are used to collect form-data
2. **GET**
   a. via the URL parameters.
   b. visible to everyone
   c. non-sensitive data
3. **POST**
   a. via the HTTP POST method
   b. invisible to others
   c. support for files uploading

Composer autoloading

1. Dependency management
2. Install https://getcomposer.org/download/
3. Can:
   a. Load PHP Class
   b. Use other people library

Class namespace

1. Group files/items/classes
2. To solve:
    a. Collision
    b. Ability to alias
3. Sample:
    a. **namespace** Database/User;
    b. **use** Database/User;

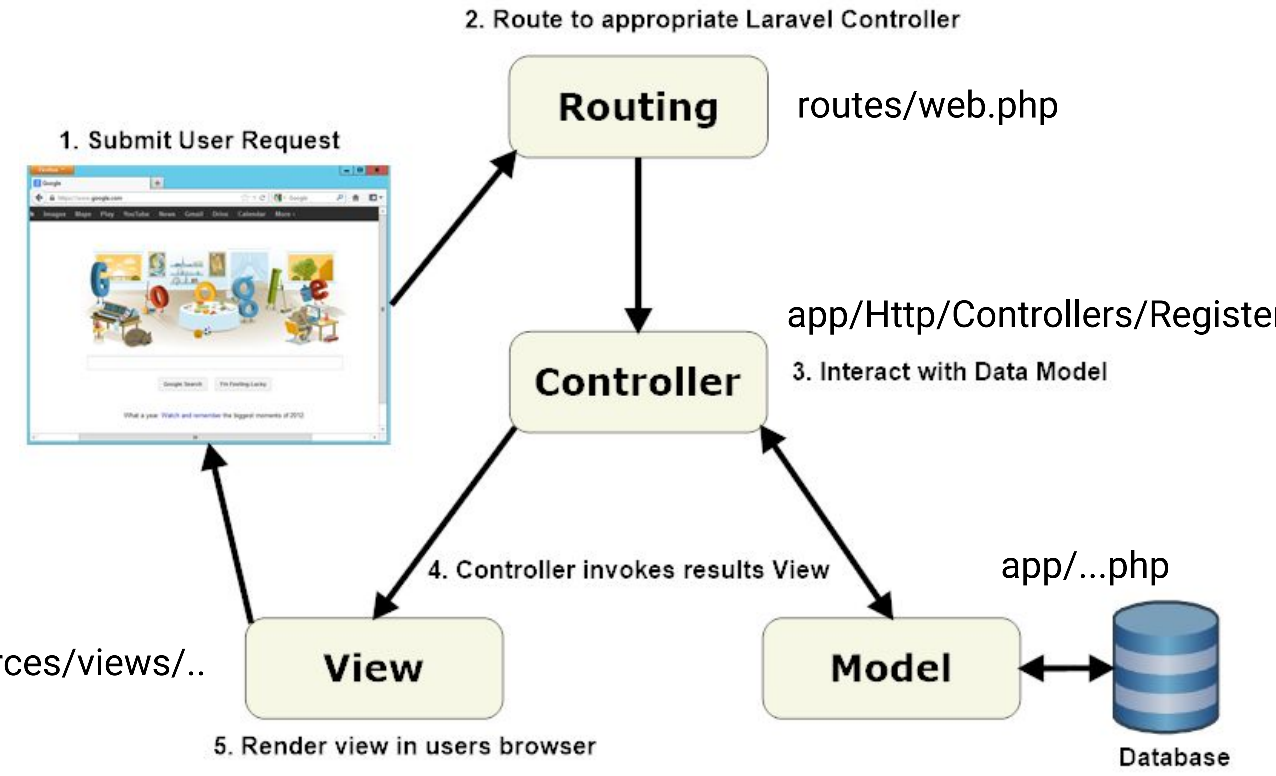
Framework (laravel) !!!

1. Web application framework with **expressive, elegant** syntax
2. Take the **pain out** of web development
3. **Combine the very best** of what we have seen in other web frameworks
    a. Ruby on Rails
    b. ASP.NET MVC
    c. Sinatra

Setup Laravel

https://laravel.com/docs/5.8

MVC?


2. Route to appropriate Laravel Controller
Routing
1. Submit User Request
Controller
3. Interact with Data Model
4. Controller invokes results View
View
Model
Database
5. Render view in users browser

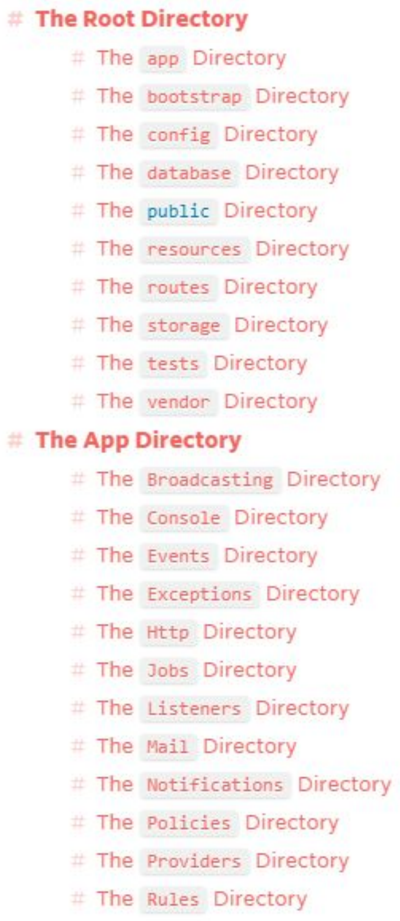
routes/web.php

app/Http/Controllers/RegisterController.php

app/...php

resources/views/..

Directory Structure



The Root Directory

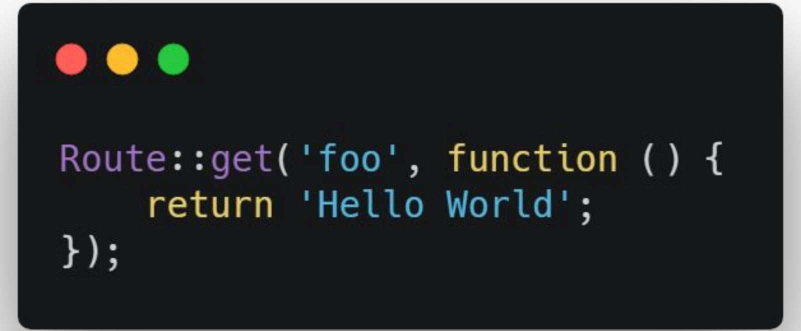
- # The `app` Directory
- # The `bootstrap` Directory
- # The `config` Directory
- # The `database` Directory
- # The `public` Directory
- # The `resources` Directory
- # The `routes` Directory
- # The `storage` Directory
- # The `tests` Directory
- # The `vendor` Directory

The App Directory

- # The `Broadcasting` Directory
- # The `Console` Directory
- # The `Events` Directory
- # The `Exceptions` Directory
- # The `Http` Directory
- # The `Jobs` Directory
- # The `Listeners` Directory
- # The `Mail` Directory
- # The `Notifications` Directory
- # The `Policies` Directory
- # The `Providers` Directory
- # The `Rules` Directory

Routing

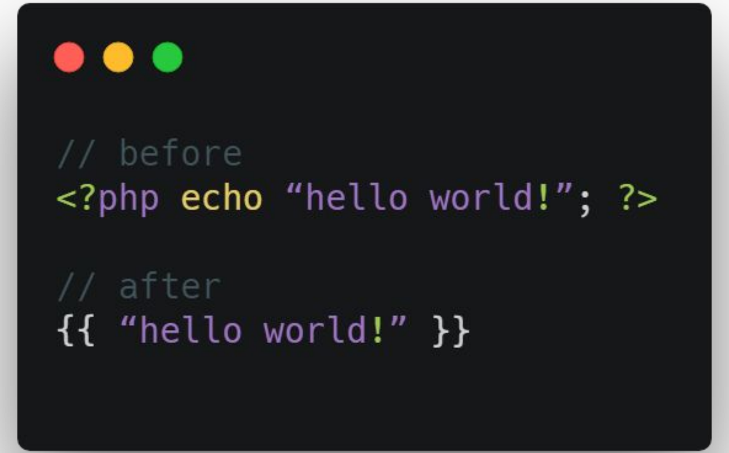
- Route all your application requests to its appropriate controller
    - routes/web.php
- Example:



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

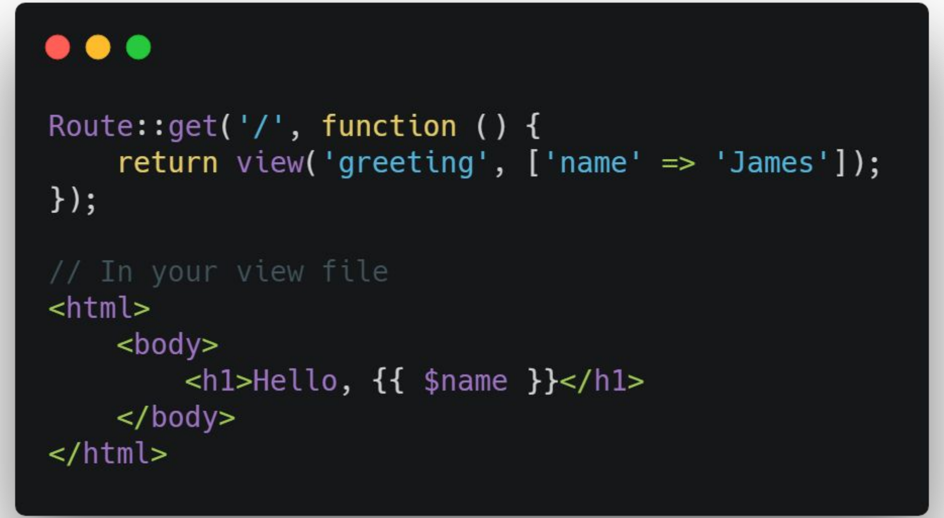
Blade

- Why? To reduce duplication and complexity.
- Example:



```
// before
<?php echo “hello world!”; ?>

// after
{{ “hello world!” }}
```

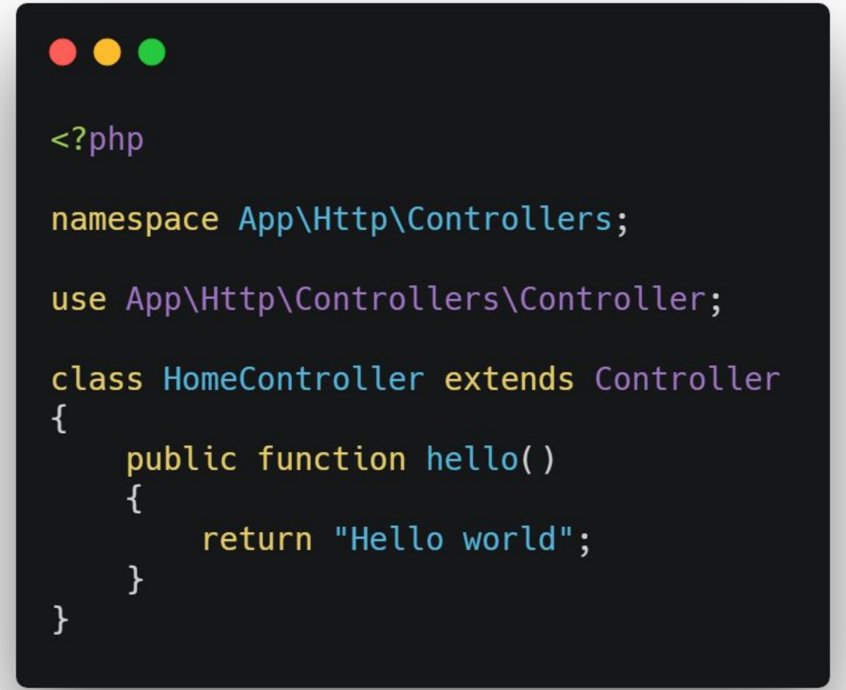
Send data to view



```
Route::get('/', function () {
    return view('greeting', ['name' => 'James']);
});

// In your view file
<html>
    <body>
        <h1>Hello, {{ $name }}</h1>
    </body>
</html>
```

Controller

- Group related request handling logic into a single class.



```php
<?php

namespace App\Http\Controllers;

use App\Http\Controllers\Controller;

class HomeController extends Controller
{
    public function hello()
    {
        return "Hello world";
    }
}
```

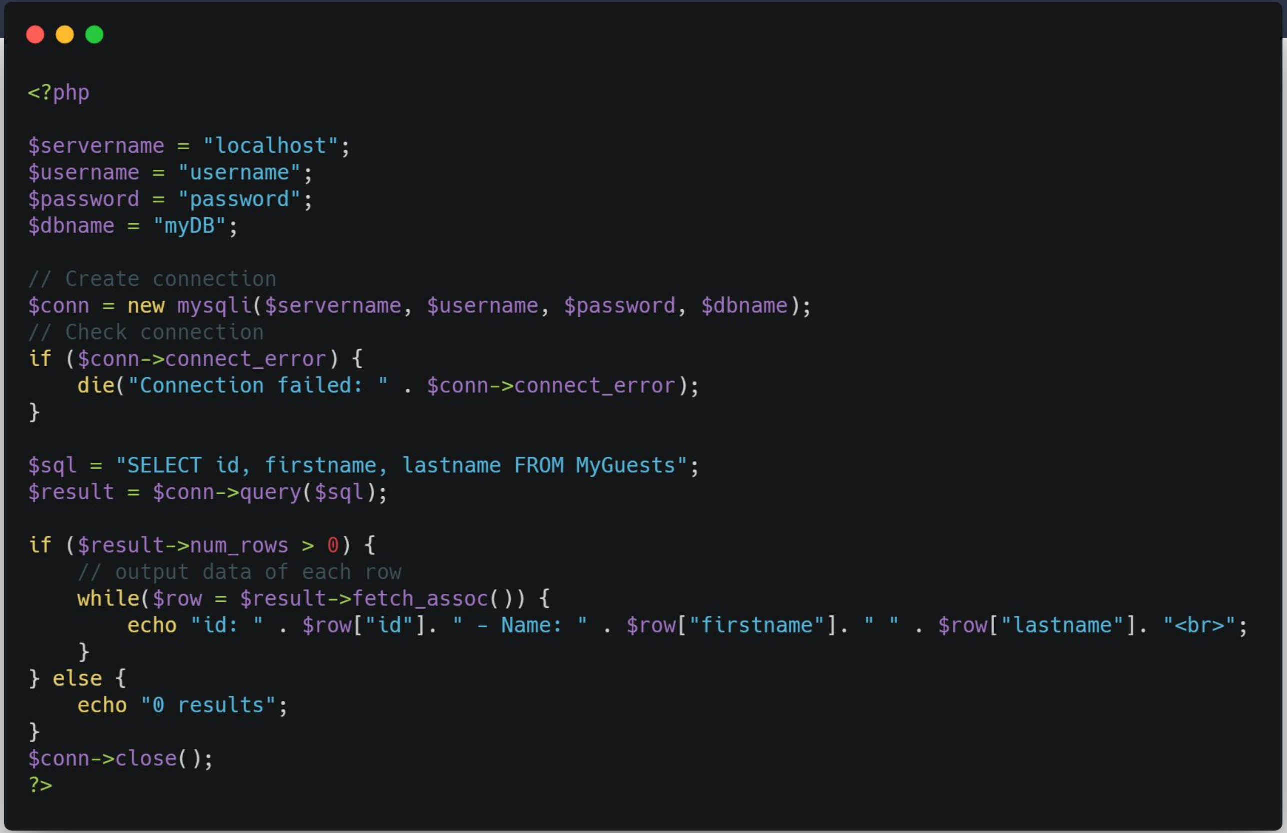
Database & Migration

- Eloquent ORM included with Laravel provides a beautiful, simple ActiveRecord implementation for working with your database. Each database table has a corresponding "Model" which is used to interact with that table.
- **Basic Command:**
    - Php artisan make:migration name_of_migration
    - Php artisan migrate
    - Php artisan migrate:rollback

Eloquent ?

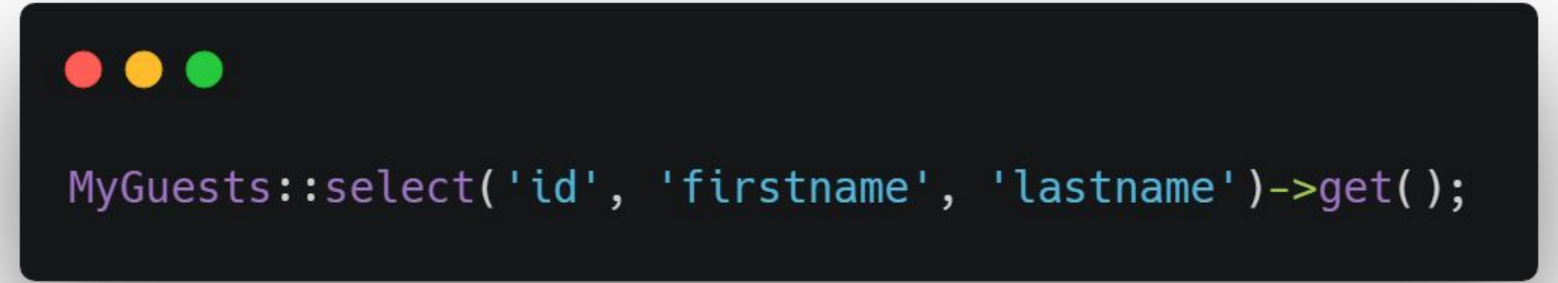
- Eloquent ORM included with Laravel provides a beautiful, simple ActiveRecord implementation for working with your database.

Before



```php
<?php

$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {
        echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"]. "<br>";
    }
} else {
    echo "0 results";
}
$conn->close();
?>
```

After



```
MyGuests::select('id', 'firstname', 'lastname')->get();
```

Form Handling and CSRF Protection

- Cross-site request forgeries are a type of malicious exploit whereby unauthorized commands are performed on behalf of an authenticated user.

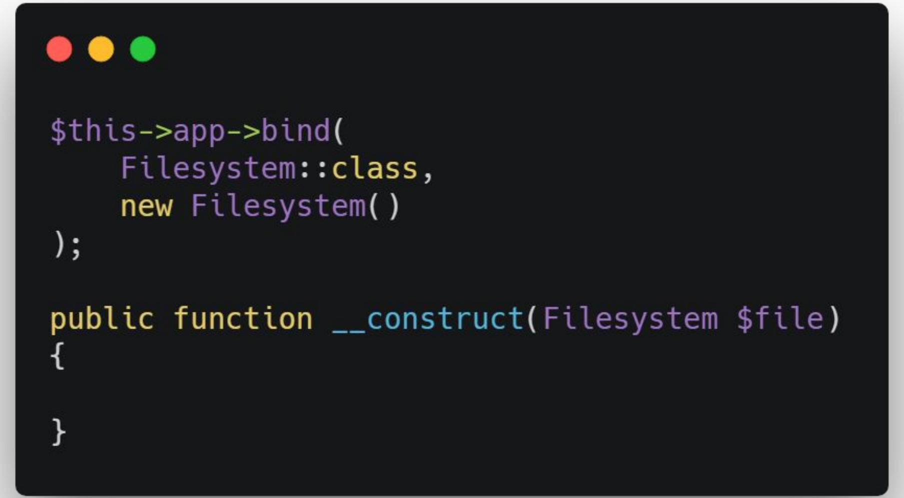
2 Layer Validation

- Client side
- Server side

Eloquent Relationship

Core Concepts: Service Container and Auto-Resolution

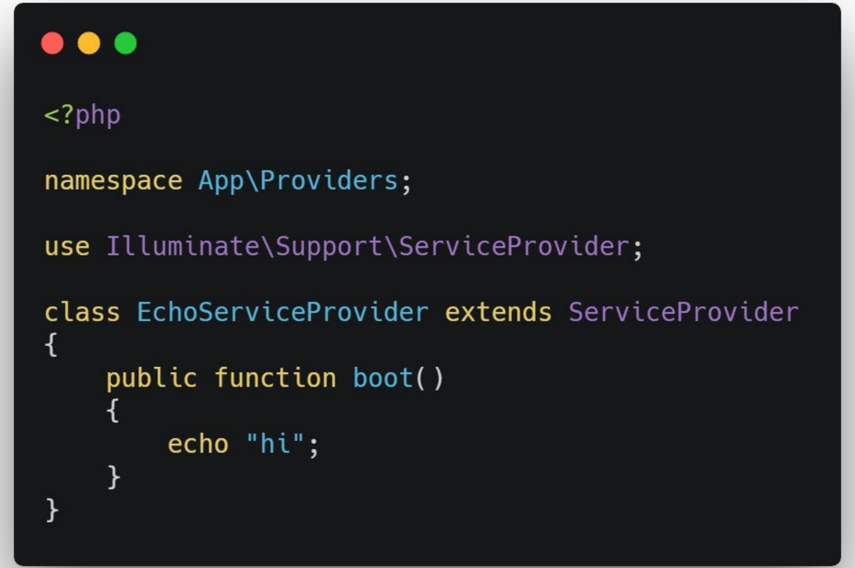
- Managing class dependencies and performing dependency injection.



```
$this->app->bind(
    Filesystem::class,
    new Filesystem()
);

public function __construct(Filesystem $file)
{

}
```

Core Concepts: Service Providers

- Central place of all Laravel application

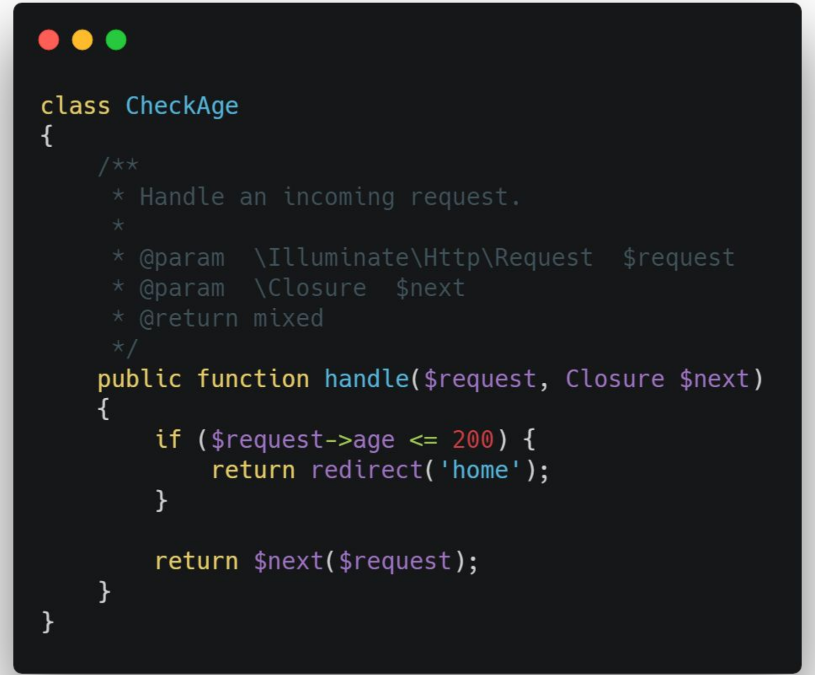


```php
<?php

namespace App\Providers;

use Illuminate\Support\ServiceProvider;

class EchoServiceProvider extends ServiceProvider
{
    public function boot()
    {
        echo "hi";
    }
}
```

Core Concepts: Configuration and Environments

Core Concepts: Middleware

- Filtering HTTP requests entering your application.



```
class CheckAge
{
    /**
     * Handle an incoming request.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \Closure  $next
     * @return mixed
     */
    public function handle($request, Closure $next)
    {
        if ($request->age <= 200) {
            return redirect('home');
        }

        return $next($request);
    }
}
```

Whats Next?

exercise

Create 3 table called: **customers, products, orders**

customers

- name
- address
- gender
- age

products

- name
- price
- quantity

orders

- product_name
- total
- quantity