

Sounds Good

Agnieszka Rudek, Maciej Zienkiewicz, Michał Ulewski

5 czerwca 2020

Prowadzący mgr. inż. Krzysztof Rewak



Spis treści

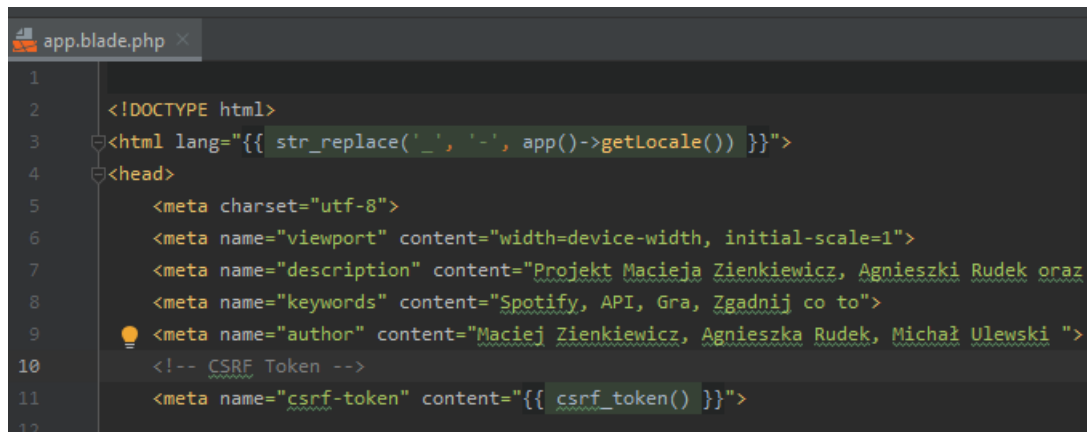
1	Opis funkcjonalny systemu	3
2	Zagadnienia zawarte w projekcie	3
2.1	HTML5	3
2.2	CSS3	3
2.3	Formularze	4
2.4	Baza danych	4
2.5	Router	5
2.6	Uwierzytelnianie	6
2.7	MVC	7
2.8	ORM	8
2.9	Konsumowanie API	8
2.10	Mail	9
2.11	Lokalizacja	10
2.12	RWD	11
2.13	System zarządzania zależnościami	13
2.14	Automatyzacja	13
2.15	SEO	13
3	Streszczenie opisu technologicznego	14
4	Instrukcja lokalnego uruchomienia systemu	14
5	Wnioski projektowe	14

1 Opis funkcjonalny systemu

Aplikacja internetowa **Sounds Good** została stworzona w celach rozrywkowych na zasadzie quizu. Gra oparta została na losowym utworze z API Spotify.

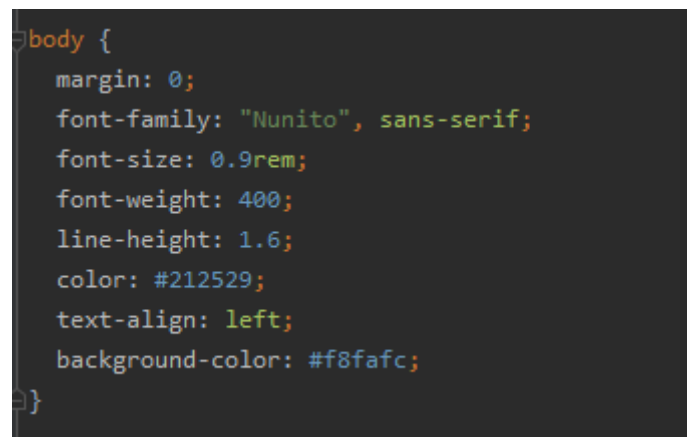
2 Zagadnienia zawarte w projekcie

2.1 HTML5



```
1
2 <!DOCTYPE html>
3 <html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
4 <head>
5     <meta charset="utf-8">
6     <meta name="viewport" content="width=device-width, initial-scale=1">
7     <meta name="description" content="Projekt Macieja Zienkiewicz, Agnieszki Rudek oraz
8     <meta name="keywords" content="Spotify, API, Gra, Zgadnij co to">
9     <meta name="author" content="Maciej Zienkiewicz, Agnieszka Rudek, Michał Ulewski ">
10    <!-- CSRF Token -->
11    <meta name="csrf-token" content="{{ csrf_token() }}">
12
```

2.2 CSS3



```
body {
    margin: 0;
    font-family: "Nunito", sans-serif;
    font-size: 0.9rem;
    font-weight: 400;
    line-height: 1.6;
    color: #212529;
    text-align: left;
    background-color: #f8fafc;
}
```

2.3 Formularze

Register

Name

E-Mail Address

Password

Confirm Password

Register

2.4 Baza danych

Tabela	Działanie
<input type="checkbox"/> leaderboards	★ Przeglądaj Struktura Szukaj Wstaw
<input type="checkbox"/> migrations	★ Przeglądaj Struktura Szukaj Wstaw
<input type="checkbox"/> password_resets	★ Przeglądaj Struktura Szukaj Wstaw
<input type="checkbox"/> users	★ Przeglądaj Struktura Szukaj Wstaw
4 tabel	Suma
<div><div></div><div><input type="checkbox"/> Zaznacz wszystko</div><div>Z zaznaczonymi: <div></div></div></div>	

2.5 Router

```
C:\laragon\www\SoundsGood (master)
λ php artisan route:list
```

Domain	Method	URI	Name	Action
	GET HEAD	/		App\Http\Controllers\HomeController@index
	GET HEAD	api/user		Closure
	GET HEAD	game/{points}/{wrong}/{temp}		App\Http\Controllers GameController@index
	GET HEAD	home	home	App\Http\Controllers HomeController@index
	GET HEAD	lang/{locale}		Closure
	GET HEAD	leaderboard		App\Http\Controllers LeaderBoardController@index
	GET HEAD	login	login	App\Http\Controllers AuthController@login
	POST	login		App\Http\Controllers AuthController@login
	POST	logout	logout	App\Http\Controllers AuthController@logout
	POST	password/email	password.email	App\Http\Controllers AuthController@passwordEmail
	GET HEAD	password/reset	password.request	App\Http\Controllers AuthController@passwordResetRequest
	POST	password/reset	password.update	App\Http\Controllers AuthController@passwordResetUpdate
	GET HEAD	password/reset/{token}	password.reset	App\Http\Controllers AuthController@passwordResetToken
	GET HEAD	points/{song_id}/{current_track}/{points}/{wrong}/{temp}		App\Http\Controllers GameController@points
	GET HEAD	register	register	App\Http\Controllers AuthController@register
	POST	register		App\Http\Controllers AuthController@register

```
Route::get( uri: '/', action: 'HomeController@index');
Auth::routes();
Route::get( uri: 'lang/{locale}', function ($locale){
    Session::put('locale', $locale);
    return redirect()->back();
});
Route::get( uri: '/game/{points}/{wrong}/{temp}', action: 'GameController@index');
Route::get( uri: '/home', action: 'HomeController@index')->name( name: 'home');
Route::get( uri: '/points/{song_id}/{current_track}/{points}/{wrong}/{temp}', action: 'GameController@points');
Route::get( uri: '/leaderboard', action: 'LeaderBoardController@index');
```

2.6 Uwierzytelnianie

Login

E-Mail Address

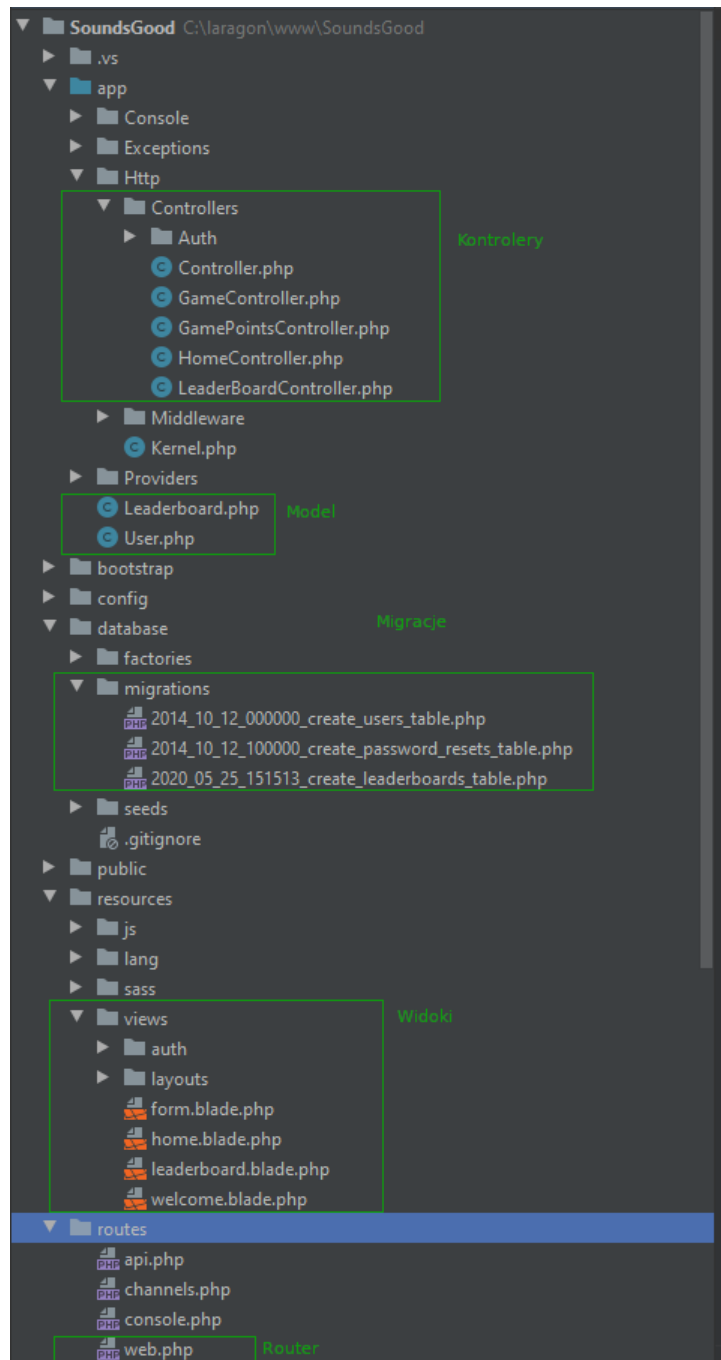
Password

☐ Remember Me

Login

[Forgot Your Password?](#)

2.7 MVC



2.8 ORM

Wykorzystanie Eloquenta

```
C:\laragon\www\SoundsGood (master)  
λ php artisan make:model Leaderboard|
```

```
class Leaderboard extends Model  
{  
    public static function LeaderSave($points){  
        $newpoints = new Leaderboard;  
        $newpoints->id_user = Auth::user()->id;  
        $newpoints->name = Auth::user()->name;  
        $newpoints->points = $points;  
        $newpoints->save();  
    }  
}
```

2.9 Konsumowanie API

```
$limit = 100;  
$offset = 100;  
$api = new Larafy( clientId: '00c8ff1427674a0e9d26895bb85ced9b', clientSecret: '59e6f9c975a94daab60ef101c3c52671');  
$api->setMarket( market: 'PL')->setLocale( locale: 'pl_PL');  
try {
```


2.10 Mail

```
use VerifiesEmails;

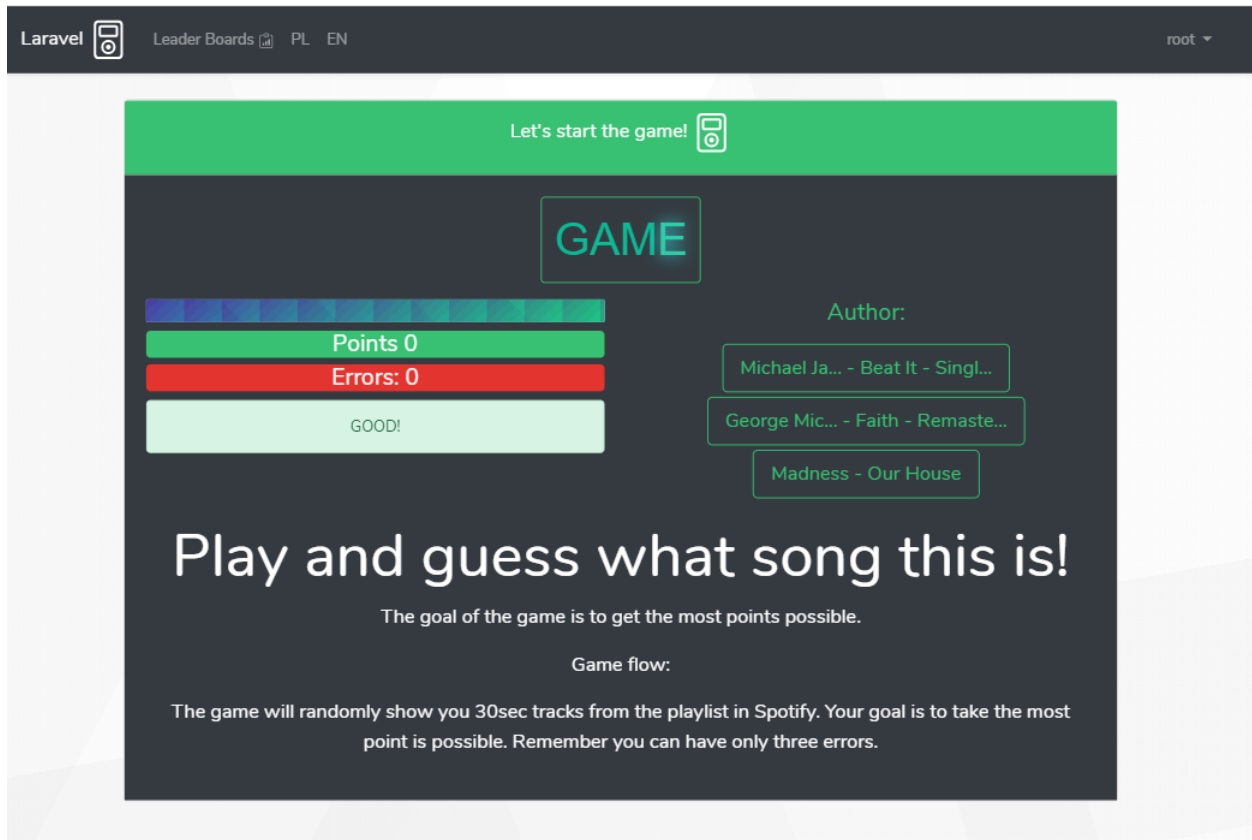
/**
 * Where to redirect users after verification.
 *
 * @var string
 */
protected $redirectTo = '/home';

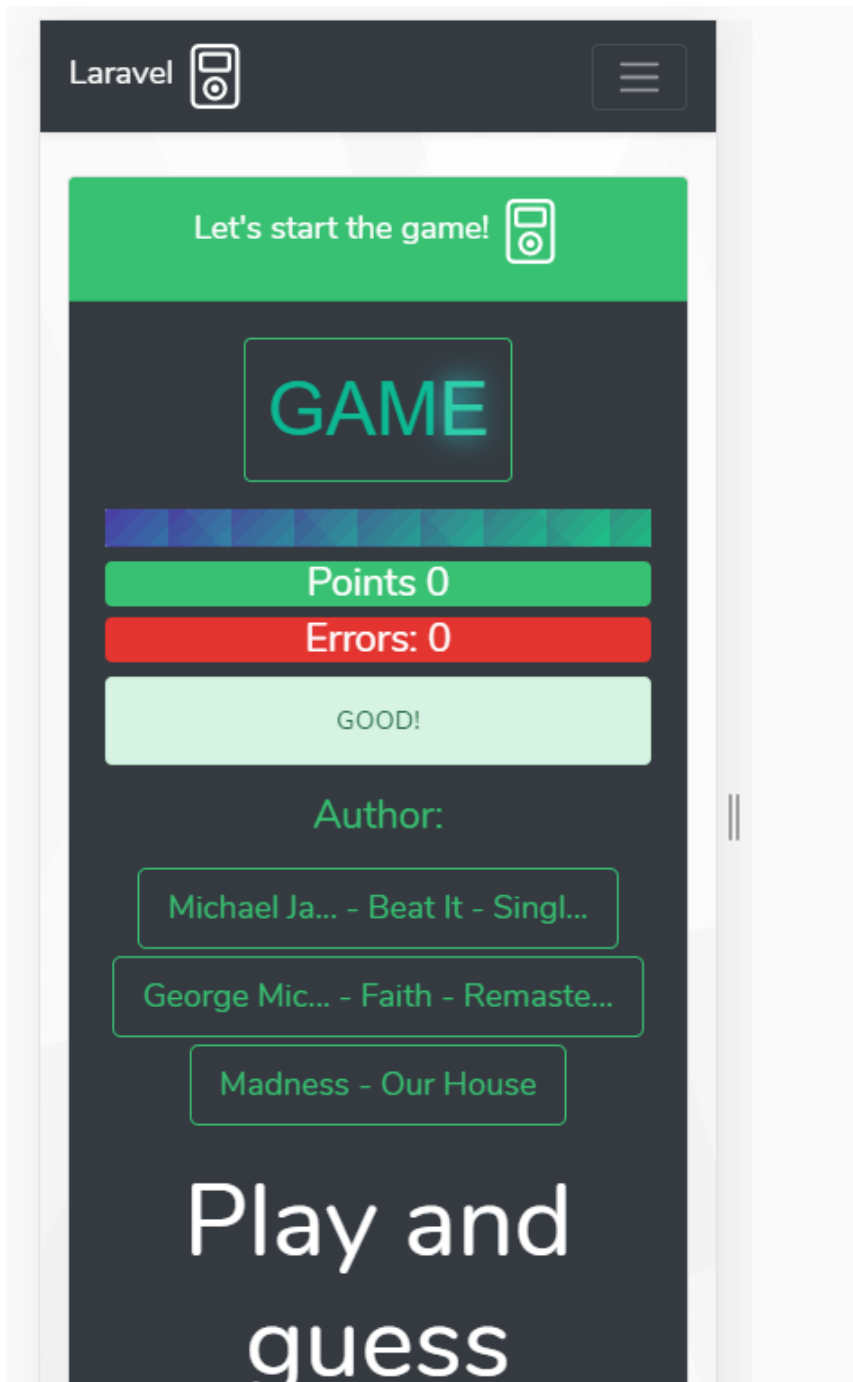
/**
 * Create a new controller instance.
 *
 * @return void
 */
public function __construct()
{
    $this->middleware( middleware: 'auth');
    $this->middleware( middleware: 'signed')->only( methods: 'verify');
    $this->middleware( middleware: 'throttle:6,1')->only( methods: 'verify', 'resend');
}
```

2.11 Lokalizacja

```
class Localization
{
    /**
     * Handle an incoming request.
     *
     * @param \Illuminate\Http\Request $request
     * @param \Closure $next
     * @return mixed
     */
    public function handle($request, Closure $next)
    {
        if(\Session::has('locale')){
            \App::setLocale(\Session::get('locale'));
        }
        return $next($request);
    }
}
```

2.12 RWD





2.13 System zarządzania zależnościami

Wykorzystanie Composera do utworzenia projektu

composer global require laravel/installer

composer create-project --prefer-dist laravel/laravel blog

2.14 Automatyzacja

Wykorzystanie Sass

```
@keyframes text-flicker {
  2% {
    color:$color47;
    text-shadow: $textshadow;
  }
  3% {
    color:$color47;
    text-shadow: none;
  }
  6% {
    color:$color47;
    text-shadow: $textshadow;
  }
  9% {
    color: $color47;
    text-shadow: none;
  }
  11% {
    color: $color34;
    text-shadow: $textshadow2;
  }
}
```

2.15 SEO

```
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<meta name="description" content="Projekt Macieja Zienkiewicz, Agnieszki Rudek oraz Michała
<meta name="keywords" content="Spotify, API, Gra, Zgadnij co to">
<meta name="author" content="Maciej Zienkiewicz, Agnieszka Rudek, Michał Ulewski ">
<!-- CSRF Token -->
<meta name="csrf-token" content="{ csrf_token() }">
```

3 Streszczenie opisu technologicznego

- IDE: PHP Storm
- Menadżer zależności dla PHP - Composer
- Baza danych MySQL i serwer Apache - Laragon
- Framework back-end PHP - Laravel
- Framework front-end PHP - Bootstrap
- Terminal Bash - CMDER

4 Instrukcja lokalnego uruchomienia systemu

Należy pobrać i zainstalować pakiet **Laragon**.

Utworzyć bazę danych w **phpMyAdmin** o nazwie **laravel**

Za pomocą terminala pobrać repozytorium z projektem za pomocą komendy

git clone https://www.github.com/pxmaciej/SoundsGood SoundsGood

W terminalu wpisać **cp .env example .env**

W pliku **.env** uzupełnić dane logowania do bazy danych

Wpisać komendę generującą nowy klucz: **php artisan key:generate**

Wpisać komendę migrującą dane do bazy danych: **php artisan migrate**

Wpisać komendę tworzącą domyślne obiekty w bazie danych: **php artisan db:seed**

Wpisać komendę tworzącą domyślne obiekty w bazie danych: **php artisan serve**

W przeglądarce wejść na adres <http://127.0.0.1:8000>

5 Wnioski projektowe

PhpStorm jest dobrym narzędziem, które pozwala nam na zachowanie przejrzystości podczas pracy. Posiada bardzo dużo przydatnych funkcji. Laravel oraz Bootstrap w bardzo dużym stopniu upraszczają pracę ze względu na gotową zawartość, natomiast to czego nam brakuje można bardzo łatwo doinstalować za pomocą Composera.