JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 11 (2019) 125-130
Wysłane: 2018-11-28
Przyjęte: 2018-12-06

Porównanie możliwości tworzenia aplikacji PHP na przykładzie Yii2 i Laravel

Olena Sydorchuk*

Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W pracy przedstawiono wyniki porównania możliwości tworzenia aplikacji internetowych w technologii PHP na przykładzie popularnych platform programistycznych Yii2 i Laravel. Badania zostały zrealizowane poprzez implementację dwóch aplikacji o takiej samej funkcjonalności, korzystającej z tej samej bazy danych. W obu technologiach porównano strukturę aplikacji, elementy tworzenia interfejsu graficznego, wybrane metryki kodu oraz efektywność pracy z bazami danych. W podsumowaniu wskazano najważniejsze wady i zalety obu rozwiązań.

Słowa kluczowe: Yii2; Laravel;PHP; frameworki

*Autor do korespondencji.

Adres e-mail: helena.sydorchuk@gmail.com

Comparison of PHP applications development using the Yii2 and Laravel examples

Olena Sydorchuk*

Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The paper presents the results of comparison of the possibilities of creating web applications in PHP technology on the example of popular programming platforms Yii2 and Laravel. The research was carried out by implementing two applications with the same functionality using the same database. In both technologies, the application structure, elements of creating the graphical interface, selected code metrics and efficiency of working with databases were compared. The summary indicates the most important advantages and disadvantages of both solutions..

Keywords: Yii2; Laravel; PHP; frameworks.

*Corresponding author.

E-mail address: helena.sydorchuk@gmail.com

1. Wstęp

W warunkach szybkiego rozwoju technologii informacyjnych i szybkiej ewolucji systemów zarządzania informacją, najbardziej aktualne z zadań, które nieustannie występują przed twórcami aplikacji, to wybór podejścia do tworzenia interfejsu i przyspieszenia kodowania. Podczas tworzenia aplikacji, deweloperzy często projektują interfejs, za pomocą specjalnych narzędzi do projektowania, przeciągając zestaw składników lub elementów sterowania na powierzchnię projektową [1].

Ze względu na olbrzymią konkurencję pomiędzy aplikacjami o podobnym zakresie funkcjonalności, użytkownik wybierze tę, która będzie bardziej intuicyjna i przyjazna. A programista w prostszy sposób będzie w stanie zmienić interfejs oraz wspierać aplikację.

Do wyboru jest wiele różnych podejść, frameworków, bibliotek, narzędzi dla różnych języków programowania. Ich nowe wersje pojawiają się na rynku nawet kilka razy w ciągu roku.

2. Cel badań

Celem niniejszego artykułu jest porównanie dwóch, stosunkowo nowych, konkurencyjnych frameworków - Yii2 i Laravel. Obie platformy wspierają proces implementacji aplikacji webowych w PHP.

Postawiono następującą tezę:

Platforma programistyczna Laravel jest bardziej przyjazna dla początkującego programisty aplikacji webowych w stosunku do platformy Yii2.

3. Wybrane możliwości Yii2 i Laravel

Zaletą korzystania z dowolnego frameworka jest :

- korzystanie ze standardowego sposobu rozwiązania zadań, co minimalizuje nieczytelność kodu;
- stosowanie gotowych rozwiązań do tworzenia i walidacji formularzy oraz standardowe ich zabezpieczenia;
- łatwość pisania kodu z użyciem sprawdzonej architektury i metod;
- aktywna społeczność, która pomaga w rozwiązywaniu pojawiających się problemów;

Stosowanie frameworka nie jest jednak wolne od wad:

- pakiet frameworka, powoduje obciążenie serwera, co może stanowić poważny problem. Na szczęście dostępne są opcje buforowania, które zmniejszają wpływ przeciążenia, a dla aplikacji korporacyjnych można użyć natywnego kodu SQL, aby zminimalizować czas wykonania zapytania. Dlatego problem przeciążenia serwera nie uniemożliwia korzystanie z niego;
- praca z ogromną ilością generowanego automatycznie kodu, wymaga zrozumienia i doświadczenia w pracy z frameworkiem;

Pomimo wskazania kilku wad - o wiele trudniej jest napisać od zera własny system. Zastosowanie frameworka w celu zaimplementowania rozbudowanego projektu z modułem użytkownika i administratora – jest niewątpliwie dobrym rozwiązaniem.

W tabeli 1 przedstawiono podstawowe właściwości obu analizowanych platform programistycznych. W kwestii związanej z rozwojem i ciągłym unowocześnianiem frameworków, wyraźną przewagę ma Laravel. Framework ten jest na bieżąco aktualizowany. Niemalże każdego roku wychodzą nowe wersje, które mają nowe możliwości, zwiększające jego konkurencyjność[1].

Tabela 1. Właściwości frameworków

Właściwości	Yii2	Laravel
Aktualna wersja	Yii 2.0.15 (marzec, 2018)	Laravel 5.6 (luty, 2018)
Data wprowadzenia	12 października 2014	9 czerwca 2011
Strona domowa projektu	https://www.yiiframework.com/	https://laravel.com/
Twórca	QiangXue	Taylor Otwell
Język programowania	PHP	PHP
Środowisko programistyczne	NetBeans, PHP Storm, Zend Studio	NetBeans, PHP Storm, IntelliJ IDEA
Generacja widoku wspomagana przez IDE	Nie posiada takiej możliwości	Jest możliwość generacji widoku z kontrolera dla jego wybranej metody
Instalacja bibliotek	Możliwość podłączenia bibliotek innych firm	Możliwość podłączenia bibliotek innych firm
Społeczność (listy dyskusyjne, grupy wsparcia itp.)	https://forum.yiiframework.com; https://www.yiiframework.com;	http://laravel.org.pl; https://plus.google.com/communities/106838454591011616186 8; https://www.wykop.pl/tag/laravel/
Wielkość [MB]	30 MB	16 MB
Obsługiwane bazy danych	MySQL, PostgreSQL, SQLite, SQL Server i Oracle, MongoDB	MySQL, PostgreSQL, SQLite i SQL Server, MongoDB
Stosowany ORM	Active Record	*Eloquent*
Routing - przykład	use yii\helpers\Url; $url = Url::to(['site/view','id' => 1]);	Route::get('funckja', function () { return 'Witaj ';});
Architektura	MVC	MVC
Silniki szablonów	Twig, Smarty	Blade
Licencja	GFDL	MIT

W tabeli 2 przedstawiono podstawowe wady i zalety obu rozwiązań.

4. Aplikacja testowa

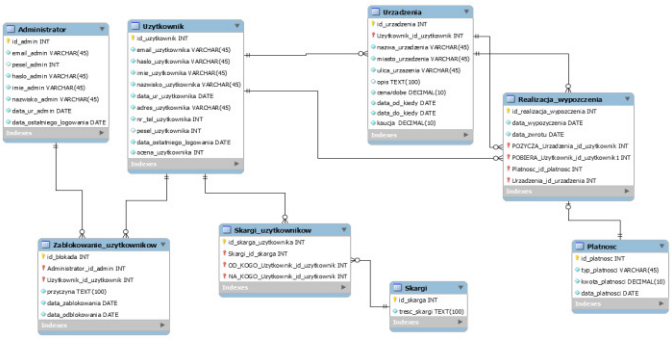
W celu przeprowadzenia porównania zostały utworzone dwie aplikacje testowe o tej samej funkcjonalności, współpracujące z tą samą bazą danych MySQL [2]. Aplikacja oferuje możliwość rejestracji i uwierzytelniania użytkownika w celu skorzystania z oferty wypożyczenia pojazdu dwukołowego (skuter, rower).

W panelu użytkownikawyświetlają się informacje o wiadomościach od innych użytkowników systemu, o czasie, na który jest wypożyczony pojazd itp.

Tabela 2. Najważniejsze wady i zalety frameworków

Właściwości	Yii2	Laravel
Zalety	Łatwość nauczenia się tworzenia prostego projektu;wiele wbudowanych rozwiązań dla interfejsów; doskonały generator modeli, kontrolerów i akcji CRUD;kompletny zestaw narzędzi do uproszczonego wdrażania RESTfull API.	Wbudowany kolektor skryptów i scss; wbudowany korektor szablonów Blade; bardzo elastycznyrouting;elastyczne możliwości pisania; interfejsu REST API; szybko rozwija się.
Wady	Mało elastyczny routing; bardzo rzadko pojawiają się nowe wersje; zbyt mało rozdzielonyfrontend i backend.	Duża liczba klas działa poprzez funkcje wspomagające; trudniejszy do nauki w porównaniu do Yii2 (subiektywna ocena autorki); brak oficjalnej dokumentacji w języku rosyjskim (z wiedzy autorki), brak wbudowanych generatorów interfejsów.

Aplikacja współpracuje z relacyjną bazą danych MySQL [2]. Model struktury bazy przedstawia rys. 1.



Rys. 1. Graficzna reprezentacja diagramu ERD

Strona główna aplikacji dla zalogowanego użytkownikaprzedstawiona jest na rysunku 2.

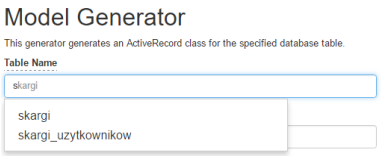
5. Elementy implementacjiaplikacji testowej

5.1. Yii

Aplikacja Yii jest zorganizowana zgodnie z wzorcem architektonicznym model-widok-kontroler (MVC). Modele reprezentują dane, zasady sprawdzania poprawności ilogikę biznesową, widoki odpowiadają za wyświetlanie informacji związanych z modelem, kontrolery przyjmują dane wejściowe i konwertują je na polecenia dla modeli i widoków.

W Yiido tworzenia modeli dla tabel wykorzystano generator kodu GII [7] (Rys. 3).

Rys. 2. Strona główna aplikacji testowej



Rys. 3. Generator kodu dodawania nazwy tabeli (Table_Name)

AccessControl jest filtrem akcji, która dopasowuje podaną ścieżkę do odpowiedniego kontrolera i jego metody (akcji). .Przykład 1 przedstawia podłączenie modeli w danym kontrolerze.

Przykład 1. Dodanie modeli do kontrolera

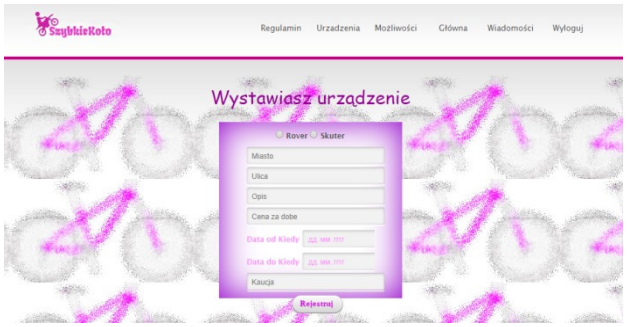
```
use app\models\ContactForm;
use app\models\Uzytkownik;
use app\models\LoginForm;
use app\models\MailerForm;
class SiteController extends Controller {

 public function actions() {
    return [
       'error' => [
          'class' => 'yii\web\ErrorAction',
       ],
      'captcha' => [
         'class' => 'yii\captcha\CaptchaAction',
          'fixedVerifyCode' => YII_ENV_TEST ? 'testme' : null,
],
    ];
 }
}
```

W aplikacjach internetowych, widoki są zwykle tworzone za pomocą szablonów, zarządzanych przez komponent widoku aplikacji. Formularze generowane jako widoki w Yii oferują prostą możliwość wprowadzenia walidacji za pośrednictwem modelu (Przykład 2).Rysunek 4 przedstawia widok strony z formularzem.

Przykład 2. Walidacja w modelu w Yii (Rys. 6)

```
public function rules()
{
 return [
   ['nazwa_urzadzenia', 'required', 'message' =>
         'Wybierz rodzaj urzadzenia'],
   ['miasto_urzadzenia','required', 'message' =>
         'Wpisz  miasto'],
   ['ulica_urzadzenia', 'required', 'message' =>
         'Wpiszulice'],
   ['cena_dobe', 'required', 'message' =>
      'Wpisz  cene za dobe'],
['data_od_kiedy','required', 'message' =>
      'Wpisz date od kiedy wypozyczasz'],
['data_do_kiedy', 'required', 'message' =>
      'Wpisz  date do kiedy wypozyczasz'],
['kaucja', 'required', 'message' =>
      'Wpiszkaucje'],
   ['opis', 'safe'],
  ];
}
```



Rys. 4. Widok strony z formularzem

Przykład 2 przedstawia walidacjęw modeludla danych z rysunku 4.

Nawigacja na stronie ma postać menu szybkiego dostępu (Przykład 3).

Listing 3. Fragment widoku z nawigacją w Yii

```
//strony, które zawsze będą widoczne
<li><a href="<?= Url::to(['/reservation/index']);
?>">Regulamin</a></li>
<li><a href="<?= Url::to(['/reservation/devices']);
?>">Urzadzenia</a></li>
//jeśli użytkownik nie jest zalogowany
<?phpif (Yii::$app->user->isGuest): ?>
<li><a href="<?= Url::to(['/reservation/exposure']);
?>">Wystawić</a></li>
<li><a href="<?= Url::to(['/site/register']);
?>">Rejestracja</a></li>
<li><a href="<?= Url::to(['/site/login']);
?>">Logowanie</a></li>
<?phpelse: ?>// jeżeli użytkownik jest zalogowany
<li class="menu-with-dropdown">
<a href="#">Możliwości</a>
<div class="dd-holder">
<div class="dd-t"></div>
<div class="dd">
<ul>
<li><a href="<?= Url::to
(['/reservation/rentinging']); ?>">Masz
wynajmowane</a></li>
<li><a href="<?= Url::to(['/reservation/devided']);
?>">Maszpozyczone</a></li>
```

Dodatkowy zasóbAssetsw Yii, jest to plik, który może być użyty na stronie internetowej. Może to być plik CSS, plik JavaScript, obraz bądź plik wideo itp. Rejestracja zestawu zasobów w widoku,powoduje automatyczne dołączenie do strony arkuszy CSS i JavaScript (Przykład 4).

Przykład 4. Dołączenie plików CSS i JavaScript w Yii

```
class AppAsset extends AssetBundle
{
   public $basePath = '@webroot';
   public $baseUrl = '@web';
   public $css = [
      'css/style.css',
```

```
        'css/ie6.css',
        'js/slick/slick.css',
        'js/slick/slick-theme.css',
    ];
    public $js = [
        'js/slick/slick.min.js',
        'js/js-func.js',
        'js/png-fix.js',
];
}
```

5.2. Laravel

Aplikacje Laravel są zgodne z tradycyjnym wzorcem projektowym MVC[3], w którym:

- kontrolery obsługują żądania użytkowników i pobierają dane za pośrednictwem modeli;
- modele są stosowane do interakcji z bazą danych i pobierania informacji o obiektach;
- widoki służą do renderowania stron.

Dodatkowo, trasy służą do mapowania adresów URL do wyznaczonych działań kontrolerów.Modele zazwyczaj znajdują się w folderze *app*, ale można je umieścić w dowolnym miejscu, co określone jest w pliku konfiguracyjnym *composer.json*. Wszystkie modele dziedziczą po klasie*Illuminate\Database\Eloquent\Model*.

Najłatwiejszym sposobem utworzenia szablonu modelu jest użycie w konsoli polecenia *artisanmake:model*.

Widoki stron w aplikacji testowej w przeglądarce wyglądają niemal identycznie (zastosowane zostały te same arkusze CSS i skrypty JS). Przykład 5 przedstawia walidacjęw modeluobsługującymiwypożyczenia.

Przykład5. Fragment modelu z walidacją w Laravel

```
protected function rules()
    {
        return [
          'nazwa_urzadzenia' => 'required',
          'miasto_urzadzenia' => 'required',
          'ulica_urzadzenia' => 'required',
          'opis' => 'required',
          'cena_dobe' => 'required',
          'data_od_kiedy' => 'required',
          'data_do_kiedy' => 'required',
          'kaucja' => 'required',
        ];
    }
}
```

Fragment kodu z elementem nawigacji w Laravel prezentuje przykład 6. Obiekty tworzące menu zostały dodane do pliku widoku w folderze*./views/layouts/app.blade.php*.

Przykład 6. Fragment widoku z nawigacją w Laravel

```
<div id="header">
<div class="shell">
<h1 id="logo" class="notext">
<a href="/">SzybkieKolo</a></h1>
<div id="navigation">
<ul>
<li><a
href="{{url('/reservation/index')}}">Regulamin</a></li>
<li><a
href="{{url('/reservation/devices')}}">Urzadzenia</a></li>
    @if(!Auth::user())
<li><a
href="{{url('/reservation/exposure')}}">Wystawić</a></li>
<li><a href="{{url('/register')}}">Rejestracja</a></li>
<li><a href="{{url('login')}}">Logowanie</a></li>
  @else
<li class="menu-with-dropdown">
<a href="#">Możliwości</a>
<div class="dd-holder">
<div class="dd-t"></div>
<div class="dd">
<ul>
<li><a href="{{url('/reservation/rentinging')}}">
Masz wynajmowane</a></li>
<li><a href="{{url('/reservation/devided')}}">
Masz pozyczone</a></li>
<li><a href="{{url('/reservation/exposure')}}">
Wystawić urządzenie</a></li>
```

W tym samym pliku, w celu dołączenia reguł CSS i JS należy dodać kilka wierszy kodu do nagłówka jak w przykładzie 7.

Przykład 7. Dołączenie CSS i JS w Laravel

```
<script src="{{ asset('js/slick/slick.min.js') }}"
defer></script>
<script src="{{ asset('js/js-func.js') }}" defer></script>
<script src="{{ asset('js/png-fix.js') }}" defer></script>
<!-- Fonts -->
<link rel="dns-prefetch" href="https://fonts.gstatic.com">
<link href="https://fonts.googleapis.com/css?family=Nunito"
rel="stylesheet" type="text/css">
<!-- Styles -->
<link href="{{ asset('css/bootstrap.css') }}"
                rel="stylesheet">
<link href="{{ asset('css/style.css') }}" rel="stylesheet">
<link href="{{ asset('css/ie6.css') }}" rel="stylesheet">
<link href="{{ asset('js/slick/slick.css') }}" rel="stylesheet">
<link href="{{ asset('js/slick/slick-theme.css') }}"
                rel="stylesheet">
```

6. Definicja routingu

6.1. Yii

Podstawą routingu i tworzenia adresów URL w Yii jest użycie menedżera URL, zarejestrowanego jako składnik aplikacji *urlManager*[3].

Menadżer udostępnia:

- metodę *parseRequest()*, która służy do analizy przychodzącego żądania z parametrami,
- metodę *createUrl()*, która tworzy adres URL.

Konfiguracja routingu (Przykład 8) realizowana jest w pliku konfiguracyjnym */config/web.php*.

Przykład 8. Plik z definicją routingu w Yii

```
<?php
$params = require(__DIR__ . '/params.php');
$config = [
    'id' => 'basic',
    'basePath' =>dirname(__DIR__),
    'bootstrap' => ['log'],
    'language' => 'pl-PL',
    'components' => [
        'request' => [
            'cookieValidationKey' =>
             'gNFp8ueOxFl90dSlsgLO5kBFGeD_0lP-',
        ],
        'cache' => [ 'class' => 'yii\caching\FileCache',  ],
        'user' => [
            'identityClass' => 'app\models\Uzytkownik',
            'enableAutoLogin' => true,
        ],
        'errorHandler' => [ 'errorAction' => 'site/error',  ],
        'mailer' => [ 'class' => 'yii\swiftmailer\Mailer',
```

```
  'useFileTransport' => true,        ],
            'mail' => [
       'class' => 'yii\swiftmailer\Mailer',
       'useFileTransport' => false,
       'transport' => [
          'class' => 'Swift_SmtpTransport',
          'host' => 'szybkie_kolo.com',
          'username' => 'szybkiekolo17@gmail.com',
          'password' =>
'$2y$13$Bz0H1y2VUjBQDjUUgXrysuPaj.E8ShLpKjs4WMTc33R
zB6L7XZK5S',
          'port' => '8080','encryption' => 'tls', ], ],
       'log' => [
          'traceLevel' => YII_DEBUG ? 3 : 0,
          'targets' => [
             [
                'class' => 'yii\log\FileTarget',
                'levels' => ['error', 'warning'],
             ],
          ],
       ],
       'db' => require(__DIR__ . '/db.php'),
    ],
    'params' => $params,
];
if (YII_ENV_DEV) {
    $config['bootstrap'][] = 'debug';
    $config['modules']['debug'] = [
       'class' => 'yii\debug\Module',
    ];

    $config['bootstrap'][] = 'gii';
    $config['modules']['gii'] = [
       'class' => 'yii\gii\Module',
    ];
}

return $config;
```

6.2. Laravel

Wszystkie trasy (ang. routes) w Laravel są zdefiniowane w specjalnych plikach, które znajdują się w katalogu *routes*. Te pliki są automatycznie ładowane przez framework.

Plik *routes/web.php* określa trasy dla interfejsu sieciowego (Przykład 9), które są częścią grupy tzw. brokerów internetowych iudostępniają funkcje obsługi sesji i ochronę np. przed atakiem CSRF. Trasy z pliku *routes/api.php*nie obsługują stanów i należą do grupy dystrybutorów api (Przykład 9).

Przykład 9. Definicja routingu w Laravel (plik web.php)

```
<?php
Route::get('/', 'HomeController@index')->name('home');
Route::get('reservation/exposure',
'ReservationController@reservation');
Route::post('reservation/exposure',
'ReservationController@create');
Route::get('wystaw_prom',
'ReservationController@wystaw_prom');
Route::get('reservation/devices',
'ReservationController@devices');
Route::get('reservation/advertisement/{id}',
'ReservationController@advertisement');
Route::get('reservation/message',
'ReservationController@messageForm');
Route::get('reservation/renting/{id}',
'ReservationController@renting');
Route::get('reservation/userlog',
'ReservationController@userLog');
Route::get('reservation/mydevises',
'ReservationController@mydevises');
Route::get('reservation/rating',
'ReservationController@rating');
Route::get('reservation/basked',
'ReservationController@basked');
Route::get('logout', function(){
Auth::logout();
   return redirect()->back();
});
Auth::routes();
```

Przykład 10. Definicja routingu w Laravel (plik api.php)

```
<?php
use Illuminate\Http\Request;
Route::middleware('auth:api')->get('/user',
                    function (Request $request) {
   return $request->user();
});
```

7. Metryki kodu

Tabela 2 przedstawiaporównanie podstawowych metryk kodu dla obu aplikacji.

Tabela 2. Wybrane metryki kodu obu aplikacji

	Yii2	**Laravel**
Liczba plików aplikacji	5 804	55 209
:iczba plików konfiguracyjnych	7	13
Liczba dołączanych bibliotek	20	45
Waga bibliotek[MB]	53	246
Przykładowy kontroler — liczba linii kodu	134	93
Przykładowy widok — liczba linii kodu	42	34
Przykładowy model — liczba linii kodu	62	57
Rozmiar projektu [MB]	60,5	294

Z wyników zestawionych w tabeli 2 wynika, ze aplikacja w Laravel zajmuje do 5 razy więcej pamięci, i wykorzystuje więcej bibliotek. Z kolei projekt Yii2 zajmuje więcej linii kodu. Tylko w przypadku przykładowego widoku w obu rozwiązaniach liczba linii kodu jest prawie jednakowa.

8. Wydajność aplikacji

Najważniejszym parametrem każdego programu komputerowego jest jego szybkość działania. Do zbadania wydajności aplikacji testowych skorzystano z narzędzia DevTools dostępnego wprzeglądarce GoogleChrome. Wyniki przedstawiono w tabeli 3.

Tabela 3. Wydajnoś aplikacji

	Yii2	**Laravel**
Czas wyszukiwania konkretnego pojazdu do rezerwacji [s]	2,72	1,1
Czas ładowania się strony głównej [s]	4,5	1,39

Czas ładowania strony był mniejszy dla aplikacji Laravel.Dla małej aplikacji, niewielkie różnice w czasie ładowania stron nie stanowią problemu, ale w wypadku

dużego projektu, czas ładowania aplikacji może już być bardzo istotny.

9. Ocena końcowa

W tabeli 4 przedstawiono ocenę punktową analizowanych frameworków (opinia autorki) w skali 1-5, gdzie 5 jest oceną najwyższą.

Tabela 4. Końcowa ocena frameworków

Kryterium oceny	Laravel	Yii
Przejrzystość tworzenia modelu aplikacji	4	5
Poziom trudności tworzenia aplikacji	3	3
Obsługiwane bazy danych	5	5
Dostępność materiałów	5	3
Wsparcie społeczności	5	3
Wymagany poziom znajomości frameworka	3	3
Przejrzystość struktury katalogów aplikacji	3	4
Wydajność wyświetlania rekordów	5	5
Końcowa ocena punktowa	**33**	**31**

10. Wnioski

W artykule przedstawiono wybrane elementy analizy porównawczejdwóch platform programistycznych dla PHP. Na podstawie przedstawionych wyników, można wnioskować, że Laravel jest bardziej wskazany dla mniej doświadczonego programisty. Oferuje możliwość automatycznego generowania layoutu, tworzenia widoków dla wybranego modelu oraz kodu akcji kontrolerów. Dodatkowo istnieje wiele materiałów pomocniczych udostępnianych na stronach Microsoft, blogach i forach dyskusyjnych.

Framework Yii2 jest silnym konkurentem Laravel, jednak oferuje zdecydowanie mniejsze wsparcie, jeśli chodzi o generowanie kodu aplikacji webowej. Jego wielką zaletą jest dużo mniejszy rozmiar gotowej aplikacji.

W wyniku przeprowadzonych badań stwierdzono, że Laravel jest bardziej przyjazny dla początkującego programisty aplikacji webowych w stosunku do Yii2.

Literatura

[1] Merkel Dirk. Chapter 6: PHP Frameworks // Expert PHP 5 Tools. Packt Publishing, 2010.

[2] Yii 2 Application Development Cookbook, Third Edition, Andrew Bogdanov, Dmitry Eliseev, November 2016.

[3] Dokumentacja Laravel, https://laravel.ru[16.10.2018]

[4] Dokumentacja Yii, https://www.yiiframework.com[19.10.2018]

[5] Surguy M.: History of Laravel PHP framework, Eloquence emerging. http://maxoffsky.com/code-blog/history-of-laravel-php-framework-eloquenceemerging/, [10.10.2018]

[6] Yii2 By Example, develop complete web applications from scratch through practical examples and tips for beginners and more advanced users, Fabrizio Caldarelli, 2015.