МІНІСТЕРСТВО ОСВІТИ І НАУКИ

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп'ютерної та програмної інженерії

Кафедра комп'ютерних інформаційних технологій

ДОПУСТИТИ ДО ЗАХИСТУ

Завідувач кафедри

_____(*Савченко А.С.*)

«_____»_____2021 р.

ДИПЛОМНИЙ ПРОЕКТ

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ

"БАКАЛАВР"

Тема: "Взаємодія мобільного додатку та бази даних"

Виконавець: *Цвид Владислав Васильович*

Керівник: **к.т.н., доцент** *Моденов Юрій Борисович*

Нормоконтролер: **ст. викл.** *Шевченко А. Т.*

Київ 2021

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп'ютерної та програмної інженерії

Кафедра Комп'ютерних інформаційних технологій

Освітній ступінь: Бакалавр

Галузь знань, спеціальність, спеціалізація: 12 "Інформаційні технології", 122 "Комп'ютерні науки", "Інформаційні управляючі системи та технології"

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ (*Савченко А. С.)*

«____» _____ 2021 р.

ЗАВДАННЯ

на виконання дипломного проекту студента

Цвида Владислава Васильовича

1. Тема дипломного проекту: « Взаємодія мобільного додатку та бази даних » затверджена наказом ректора від «22» квітня 2021 р. № 636/ст.
2. Термін виконання роботи (проекту): з 20.04.2021 до 11.06.2021.
3. Вихідні дані до роботи (проекту): аналіз існуючих веб-серверів, аналіз найпопулярніших і найрозвинутіших php-фреймворків, веб-додаток, написаний на мові php та на основі даних, зібраних в ході досліждення проблеми.
4. Зміст пояснювальної записки: Розділ 1. Мова програмування PHP. Розділ 2. Аналіз веб-серверів та php-фреймворків. Розділ 3. Можливості вибраної технології. Розділ 4. Розробка веб-додатку.
5. Перелік обов'язкового графічного (ілюстративного) матеріалу:

   Схема роботи Varnish. Схема роботи кешу. Схема роботи менеджера черг. Графіки тестування веб-серверів. Графіки тестування php-фреймворків.

КАЛЕНДАРНИЙ ПЛАН

№ пп.	Етапи виконання дипломної роботи	Термін виконання етапів	Примітка
1.	Підбір і вивчення літературних джерел	20.04.21-19.05.21	Виконав
2.	Обгрунтування необдності дослідження технології	20.05.21 – 21.05.21	Виконав
3.	Аналіз існуючих веб-серверів	22.05.21 – 26.05.21	Виконав
4.	Аналіз популярних php-фреймворків	27.05.21 – 30.05.21	Виконав
5.	Розробка веб-додатку	01.06.21 – 10.06.21	Виконав
6.	Написання пояснювальної записки	10.06.21 – 11.06.21	Виконав

Студент Цвид Владисла Васильович

Керівник дипломної роботи Моденов Юрій Борисович

РЕФЕРАТ

Пояснювальна записка до дипломного проекту “Взаємодія мобільного додатку та бази даних”: 70 сторінок, 51 рисунків, 2 таблиці, 10 використаних джерел.

Об’єкт дослідження – розробка веб-додатку, що слугуватиме зв’язком між веб-сервером там мобільним додатком.

Мета дипломного проекту – дослідження технологій розробки веб-додатків на мові PHP та розробка власного додатку на основі зібраної інформації.

Метод дослідження – збір та аналіз необхідної інформації про мову програмування систему PHP, існуючі засоби розробки додатків, вибір оптимального фреймворку для розробки та реалізації додатку за його допомогою на мові програмування PHP.

Встановлено, що існує багато способів розробки веб-додатків за допомогою мови програмування PHP, але оптимальним способом розробки веб-додатку є використання широко розвинутого фреймворка із дуже великою спільнотую користувачів через найкращу оптимізацію, зручність та швидкодію.

Матеріали даного дипломного проекту рекомендується використовувати при проведені досліджень методів програмування на мові PHP та в практичній діяльності фахівців по розробці веб-орієнтовного ПЗ.

PHP, SYMFONY2, MYSQL, NGINX, DOCTRINE2, SONATAADMIN, ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ, ПАТЕРНИ ПРОЕКТУВАННЯ.

ЗМІСТ

ВСТУП

Сьогодні на веб-додатках тримається наше життя. Люди кожен день заходять в інтернет в свої цілях: хтось почитати новини, хтось програти в гру, хтось написати листа, а хтось просто вбити час, але сенс у тому, що так чи інакше кожна людина пов'язани з інтернетом, а отже і з веб-додатками. Веб-додатки набули популярності у кінці 1990-х – початку 2000-х років. Зараз динамічний веб-додаток – невід'ємна частина інтернету. Навіть простий менеджер з продажу хотів би мати власний сайт.

Сьоодні, веб-додатки – самостійні гравці на арені Software. Окрім того, що веб-додаток може бути повністю самостійним, він теж може буди підтримуючою системою. Для прикладу, розроблюючи мобільний додаток, інженер все ж таки зрозуміє, що не вийде вмістити всі дані на мобільному, а навіть якщо і вийдет, то що ж тоді залишиться користувачеві на пісеньки та фотографії? l тут на допомогу приходять веб-додатки, але в своєму найпершому вигляді, а у вигляді мостів між сервером та мобільним додаток або будь якою іншою не серверною систему в якій немає достатніх інструментів для комфортного обіну даними із сервером.

Одна з популярних мов для програмування веб-додатків є PHP. Перша назва: Personal Home Page Tools —мова програмування, була створена для генерації HTML-сторінок на стороні веб-сервера. PHP є однією з найпоширеніших мов, що використовуються у сфері веб-розробок. PHP підтримується переважною більшістю хост-провайдерів. PHP — проект відкритого ПЗ.

PHP інтерпретується веб-сервером в HTML, який передається на сторону замовника чи клієнта. На відміну від скриптової мови JavaScript, користувач не бачить PHP-коду, бо браузер отримує готовий html-код. Саме цей фактор є перевагою з точки зору безпеки, але ось із-за цього потрібно змиритись зі швидкість сторінок. Але ніщо не забороняє використовувати PHP для генерування і JavaScript.

lсторія PHP починається з 1995 року, коли Расмус Лердорф (англ. Rasmus Lerdorf) створив простий застосунок мовою Perl, що аналізував відвідування

користувачами його резюме на веб-сайті. Потім, коли цим застосунком вже користувалися кілька чоловік, а число охочих одержати його постійно збільшувалося, Лердорф назвав творіння Особисті інструменти домашньої сторінки англ. Personal Home Page Tools версія 1 і виставив для вільного користування . З того часу PHP[1] і залишається популярною мовою.

РОЗДІЛ 1
МОВА ПРОГРАМУВАННЯ PHP

Щоб зрозуміти php буде проведенно порівняння її з іншими мовами, а теж виділення полюсів серед інших. На даний момент PHP найбіль широко використовується для розробки веб-орієнтовних додатків.

1.1. Порівняння з інши мовами програмування

PHP vs. ASP

ASP не є самостійною мовою програмування, але є акронімом для активних серверних сторінок; фактичні мови, що використовуються для програмування ASP включають Visual Basic, JScript, C# та інші. Найбільшим мінусом ASP є те, що це приватна система, яка використовується на платформі Microsoft (IIS). Це обмежує його доступність серверів на базі Win32. Існує декілька робочих проектів, що дозволяє ASP працювати з іншими середовищами і веб-серверами: » InstantASP » Halcyon (комерційний), Чилі Soft ASP від » Chili Soft (комерційний), і » Mono (з відкритим вихідним кодом) , ASP є більш повільним і великим ніж мова PHP, з меншою загальною стабільностю.

Однією з плюсів ASP є то, що, так як він в основному використовує VBScript, це те, що відносно не тяжко підібрати мову, якщо ви вже володієте Visual Basic. Підтримка ASP теж включена за замовчуванням в IIS, що дозволяє його не тяжко налаштувати і запустити. Але, компоненти, побудовані в ASP дуже обмежені, так що якщо вам потрібно вибрать "просунуті" функції (Для прикладу, взаємодія з FTP-серверів), ви повинні будете купити додаткові компоненти.

PHP vs. ColdFusion

PHP в більшості випадків, різкіше і ефективніше для тяжких завдань у програмуванні і випробуванні нових ідей, і, на думку багатьох, буде більш стабільним і менш ресурсномістким. є його відмінна пошукова система, але й багато аргументів припускають, що пошуковий двигун не те, що має бути включено в мову веб-скриптів. Крім того, PHP працює майже на всіх сьогоднішніх платформах, в той час як Cold Fusion доступний тільки на Windows, MacOS, AIX , Solaris, Linux. Cold Fusion має хорошу IDE і в загальному різкіше для нових розробників для досягнення результатів, для простих додатків, в той час як PHP на початку вимагає більше знань з кодування. Cold Fusion розроблений з не-програмістами, в той час як PHP орієнтований на програмістів.

PHP vs. Perl

Найбільшою перевагою PHP перед Perl є те, що PHP був запрограмований для кодування сценаріїв Інтернету, в той час як Perl був розроблений, щоб зробити набагато більше. Через то, Perl, є складнішим. Гнучкість/складність Perl може зробити важкою роботу програмістів різного рівня кваліфікації. PHP менш заплутаний і має більш стабільний формат без шкоди для гнучкості та використання. PHP теж не тяжко інтегрувати в існуючу HTML, ніж Perl. В значній мірі, PHP має все «хороше» від Perl - конструкції, синтаксис і так далі - не роблячи його так складним, як Perl може бути. Тим не менш, інтерпретатор командного рядка в PHP (CLI) є досить потужним, щоб виконувати завдання високого рівня багато в чому таким же чином як Perl.

1.1. Плюси та Пріорітети

Відкритий вихідний код

PHP є легко доступним для використання. Товариство програмістів PHP з відкритим вихідним кодом надає тех. підтримку та весь час покращує оновлення основних PHP функцій. PHP безкоштовний і пошірюється під PHP General Public License, і більшість з його асоціативно необхідного програмного забезпечення такого, як MySQL, текстові редактори і сервера Apache теж вільно доступні, так це робить дуже рентабельною розробку ПЗ.

Крос-платформеність

PHP обіцяє високу сумісність з провідними операційними системами і веб-серверами. Таким чином, це дає змогу бути просто розгорнутим в декількох різних платформах. Коди PHP можуть працювати в різних операційних системах, таких як OpenBSD, Mac OSX , Linux, Windows, Solaris і т.д., а теж забезпечити підтримку всіх основних веб-серверів, таких як Apache, IIS, IPLANET і т.д.

Потенціальність та Потужність

Окремі веб-завдання можуть тепер бути з легкістю виконані за допомогою PHP. Для прикладу, тепер можна розвиватися від невеликих сайтів до величезних систем та організаційних лендингів, форумах, платформ спілкування, CRM рішень, систем електронної комерції, громадських сайтів, інтернет-магазинів і великих баз даних сайтів.

Зручність

Програмований в комфортній для користувачів формі, PHP дає більшу гнучкість, ніж C, C ++ і ASP і в цілому допомагає у збільшенні зборі аудиторії на сайті.

Швидкодія

PHP створений , щоб добре працювати з вебом, мати доступ до GET і POST і роботи з HTML і URL-адресами, для швидкого програмування веб-додатків.

Розширення

Будучи мовою з відкритим вихідним, PHP має велику кількість фреймворків , бібліотек та розширень, щоб розширити свої основні функціональні, доступні для скачування. Вихідний код PHP може бути іншим, щоб включити користувацькі розширення та компоненти, тим самим збільшуючи кількість його можливостей.

Розгортання

Є багато веб компаній, які за кілька доларів на місяць, зможуть надати сервер під управлінням PHP, так що можна зробити сайт дійсно дуже легко.

Самостійне оновлення

На сьогодні програмування динамічних веб-сайтів має величезний попит через його магічні особливості, при цьому PHP можна не тяжко оновлювати не витрачаючу велику кількість сил та часу.

Товариство підримки

Великий плюс , який пропонує PHP є його співтовариство. Якщо ви шукаєте конкретний сюжет, різкіше за все, інший замовник чи користува вже зробив щось подібне. Перевірте в PHP спільноті серед доступних. Крім того, якщо ви створили функцію, якою інші можуть користуватися, не забудьте розмістити код для інших.

Інші інструменти

Якщо вам необхідний доступ до інструментів веб-інтерфейсів таких, як Google-карти, або будь-якого іншого, PHP дая можливість не тяжко отримати доступ.

Захист та безпека

PHP забезпечує безпеку, що допомагає запобігти зловмисним атакам. Ці рівні захисту можна регулювати в конфігураційному INI-файлі.

Не великий поріг входу

Ви можете найняти PHP розробників не тяжче, ніж будь-яких інших мовних програмістів, авжеж багато людей цікавляться та вивчають таку мову як ПХП.

1.2. PHP та Інструменти при роботі

Ця мова динамічно стає популярнішою з кожним роком. На сьогодні можна виділити декілька основних інструментів, що суттєво покращують процес програмування.

Laravel – як фреймворк (бібліотека) , Laravel виглядає добре, і багато PHP-програмістів обрали цей фреймворк рамках своєї праці.

Composer - Composer в PHP є схожим до RubyGems + Bundler у Рубі. Це робить використання пакетів більш організованим, дозволяє швидко керувати залежностями в проекті. За остадні роки в розробники та учні PHP мало справу з Pear, яке насправді не дало повної віддачі.

PHP веб-сервер – довга розробка з допомогою PHP на вашому комп'ютері означала, що треба було покладатися на зовнішні веб-сервери такі, як Apache. Багато програмістів-розробників в кінцевому підсумку користувалися стороннімі наборами , Для прикладу, MAMP. Починаючи з PHP 5.4, PHP тепер поставляється зі своїм власним командним рядком веб-сервера, який, що дивно, не тяжко запустити.

Codeception – фрейморк для тестування під назвою Codeception, що насправді виглядає гарно та солідно, підтримує такі речі, як Selenium і BDD-похожі тести.

HHVM - нобір з вільним вихідним кодом на чолі з Facebook, він бере PHP-код і компілює його в байт-код, що в першу чергу буде змінений в 64 машинний код і працює добре. Це дуже цікавий проект, який робить PHP масштабним.

1.3. Шкидкість у PHP

В програмування швидкість є важливим фактором, але налаштування починається пізно або не починається взагалі. В результаті, приймаючи на апгрейд швидкодії повільний проект перше, що приходить на думку – з чого взагалі начати? Спершу треба гарно розібратися із конфігурацію PHP. Найбільше проблем там , бо окрема система має бути налаштована по своєму. Ось саме це можна зробити

швидко, не згубивши багато ресурсів і отримати реальну віддачу. Шляхи Господні незвідані.

Бізнес-аналітика по швидкості від величезних проектів:

- Microsoft побачили, що Bing запити, повільніші 2 секунд призвели до зменшення вигоди з одного користувача на 4,5%;
- А коли Mozilla зменшив на 2,3 секунди швидкість скачування їх сторінки, завантаження інсталятора Firefox збільшилися на 16,4%;
- Shopzilla замітила, що коефіцієнт конверсії збільшиться на 8-13% в результаті їх оптимізації веб-продуктивності;
- зробивши веб-сайт президента америки на 50% різкіше, вийшло збільшити кількість конверсій пожертв на 15%.

Шляхи гарного налаштування:

- PHP Оновлення. Один з найлегших поліпшеннях, який ви можете використати , щоб поліпшити продуктивність і стабільність це оновлення версії PHP. PHP 5.3.x був випущений в 2009 р і якщо ви не блукали у PHP 5.5 або 5.6, то час прийшовю. Отож можна отримати не лише вигоду від чогось нового , а теж буде видно замітне збільшення швидкості;
- Використання кеш-коду операції , PHP є інтерпретованою мовою, це пояснює, що саме всі рази, коли запитується сторінка PHP, сервер буде інтерпретувати файл PHP і компілювати у те що машина може прочитати. Кеш-коду операції зберігає цей згенерований код в кеш, таким чином, що PHP має бути інтерпретованим на першу вимогу. Не використовувавши КЕШ операцій можна пропустити дуже легкий, але заміний приріст продуктивності. На сьогодні є багато різних кеш-менеджерів, на яких можна зупинитися: Eaccellerator, APC, Zend Optimizer, XCache. Рекомендується APC, написаний творцем PHP, Расмусом Лердорф;

- Автозавантаження. Велика кількість програмістів, які програмують на ООП , а саме створюють додатки один PHP файл для визначення класу. Одна з найбільших неприємностей у PHP у написанні довгого списку включень на початку кожного коду. PHP повторно включає необхідні класи і витрачає на це час при кожному виклику коду. Використання автозавантаження дозволяє удалити всі включення та получити шидвість від оновленої продуктивності;
- Оптимізація процессу сесії. Коли HTTP є особою без громадянства, більшість реального життя веб-додатка має вимогу управління обліковими даними. У PHP, стан програми управляється за допомогою процесу. Налаштування по стандарту для PHP є зберігання даних сесії в памяті. Цей процес повільний і не збільшується ніж один сервер. Найкраще рішення для зберігання якихось данних є сесії , це зберігання в базі та кешування з Memcached. Також необхідно зменшити розмір даних сеансу (4096 байт);
- Використання кешу даних . Дані зазвичай структуровані й організовані в базі даних. Залежно від набору та правил доступності саме це може бути ресурсоємним запитом. Легким рішенням є добавляти в кеш результат першого запиту в кеші даних (рис. 1.4.1), Для прикладу, Memcached або Redis. При зміні даних, робити недійсним минулий кеш і зробити ще один SQL запит, щоб отримати новий набір результатів з бази даних;

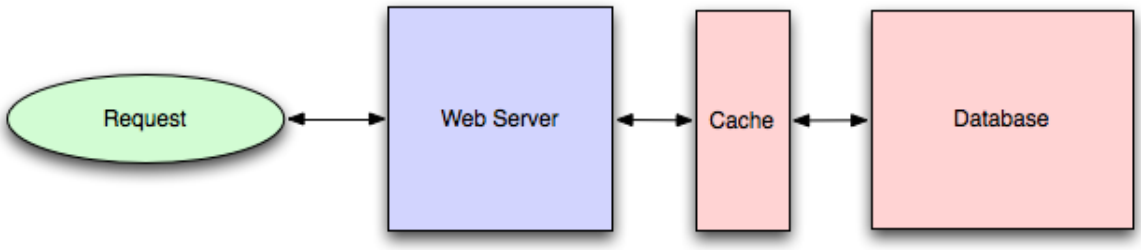


Рис. 1.4.1. схема роботи поліленого кеша

- Блокуючі задачі у фоновому режимі. Зазвичайно веб-додатки мають виконувати завдання, які займають час, щоб закінчитися. l ось найчастіше немає підстав, щоб заставити користувача на кінець. lдея полягає в створенні черги блокуючих задач для запуску в фонових завданнях;
- Фонові завдання завдання, які виконуються за кордоном основного потоку вашого проекту, та зазвичай обробляються з допомогою системи черг або повідомлень. Є багато гарних рішень, які допомагають виконувати фонові задачі (рис 1.4.2). Плюси приходять з точки зору роботи кінцевих користувачів і збільшення в письмовій формі і обробці довгих запущених завдань з черги. Є багато цікавих інструментів, які забезпечують систему чергування або повідомлень, та гарно працюють з PHP: Resque, Gearman, RabbitMQ, Beanstalkd, ZeroMQ, ActiveMQ;

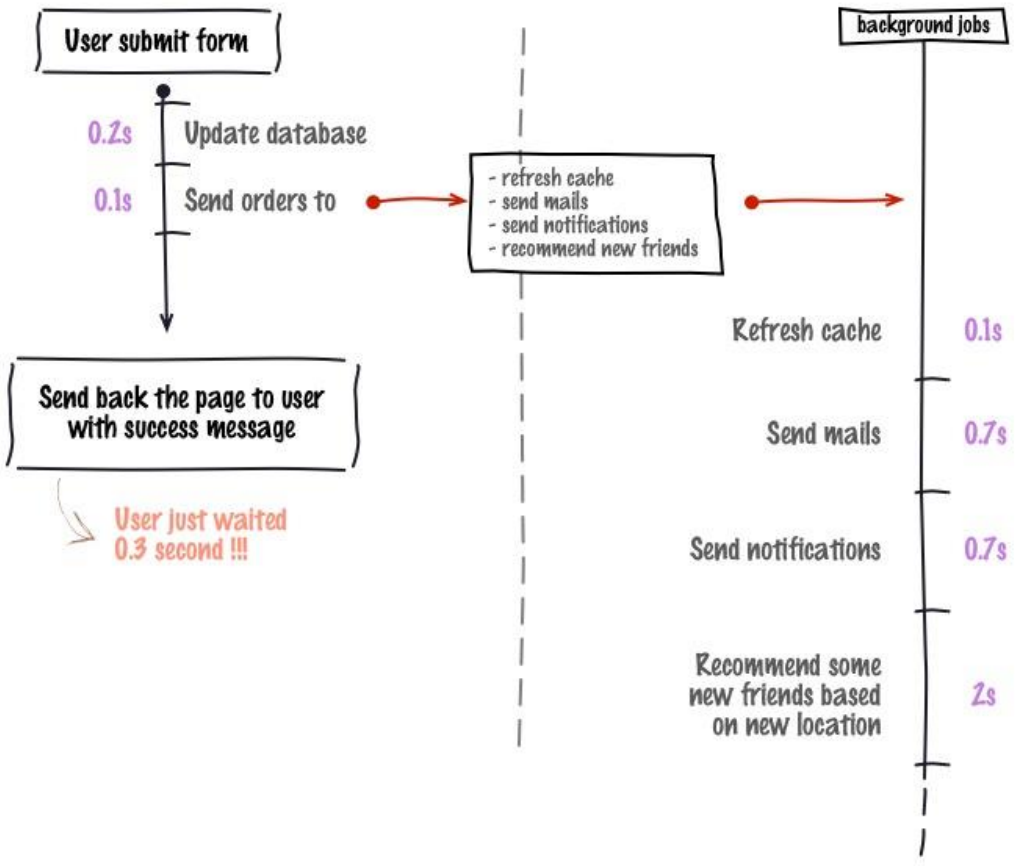


Рис. 1.4.2. схема роботи менеджера черг

- HTTP кеш. HTTP кеш є одним з найбільш заплутаних технологій в Інтернет просторі. Програміти вирішили всі ці проблеми проектування кешування приблизно 20 років тому. Суть закінченні я терміну дії або визнання недійсним і при правильному використанні може врятувати ваші сервери при високонавантажених додатках. Однією з рекомендацій є використання Varnish (рис. 1.4.3) в якості зворотного кеша проксі, щоб зменшити навантаження на сервери;

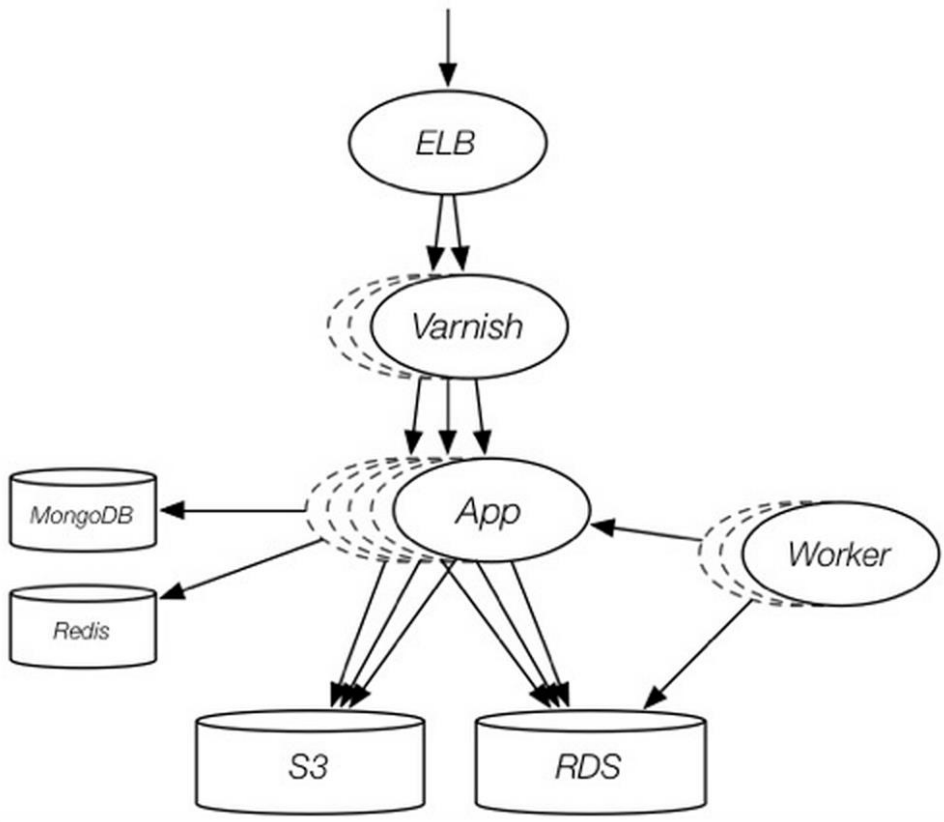


Рис. 1.4.3. схема роботи Varnish

- Оптимізація фреймворка. Працювавши з останньою стабільною версією фреймворка. Відключити все що, ви не використовуєте (безпека і т.д.). Включити функції кешу для представлення і результатів;

- Профілювання (рис. 1.4.4) PHP. Розширення PHP для потужного дебагу є Xdebug. Саме він підтримує стек і функціональні траси, профілювання і розподіл пам'яті, аналіз виконання коду. Це дозволяє програмістам не тяжко профілювати PHP

  код:

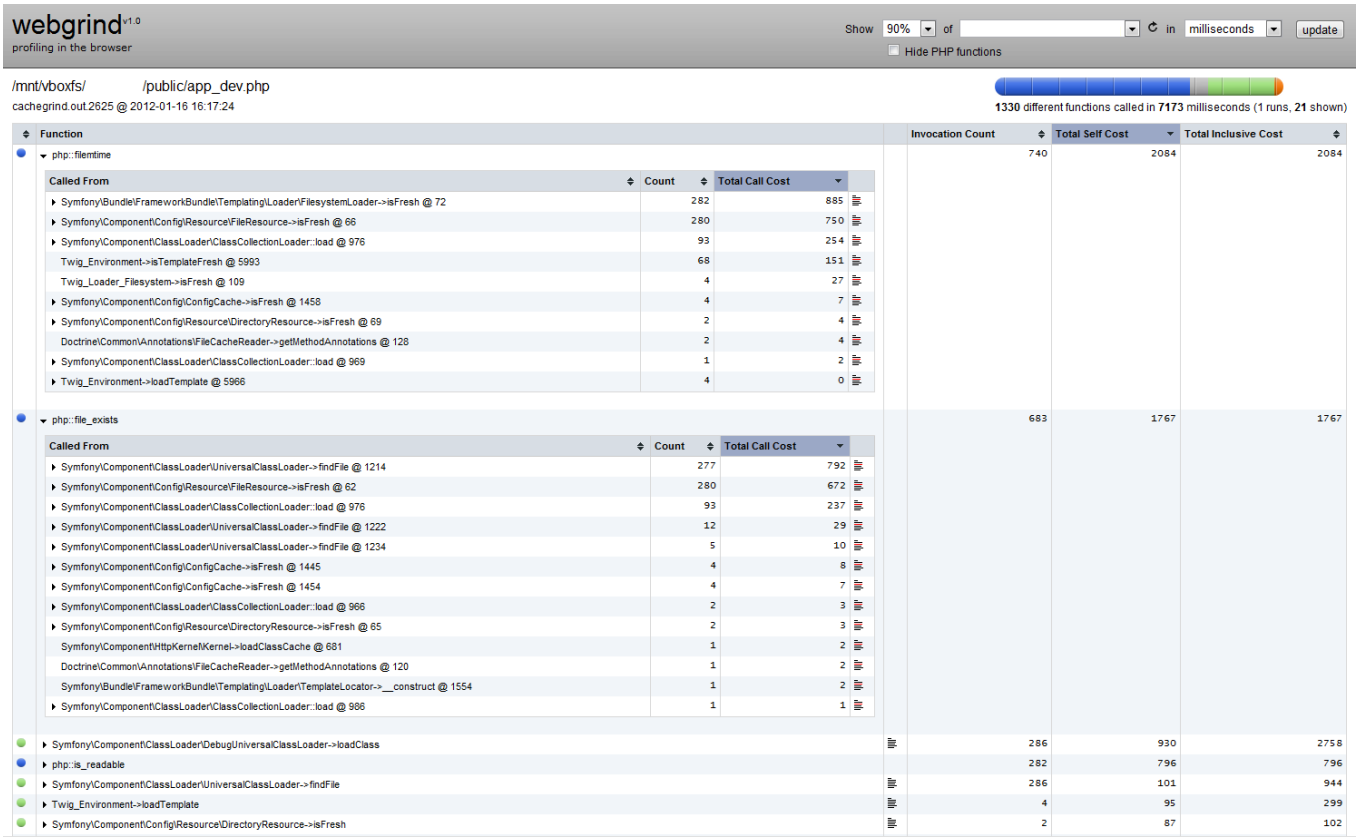


Рис. 1.4.4. робота плагіна для профілювання

XHprof є ієрархічнин профілюванням на рівні функцій для PHP. XHProf здатний репортувати на рівні функцій використання пам'яті, час центрального процесора і кількість викликів. Отож, воно підтримує можливість порівняти дві траси (ієрархічні звіти DIFF) або сукупні висновків з декількох серій;

РОЗДІЛ 2
ІСНУЮЧІ ВЕБ-СЕРВЕРИ , PHP-ФРЕЙМВОРКИ ТА ЇХ РОЗБІР

2.1. Розбір та порівняння веб-серверів

Ngnix веб-сервер

Nginx був створений у 2002 році в Росії програміст Ігор Сисоєв. Як і Apache це безкоштовний з відкритим та доступним кодом веб-сервер, що робить на Window , Linux, Unix, Mac OS X. Nginx підтримується округом користувачів, яка забезпечує широку особисту допомогу, навіть для початкових програмістів. Та саме тому, що Nginx новіше, тому менше документація та підтримка для цього, порівняно з Apache.

Основною причиною поширення Nginx є те, що це асинхронний веб-сервер. Ігор Сисоєв шукав шляхи оптимізації та вдосконалення архітектури веб-сервера без значного збільшення апаратних ресурсів. Їх рішенням була однопотокова, асинхронна та керована подіями архітектура, на відміну від синхронної моделі Apache: один потік на процес.

У традиційних моделях на основі потоків сервер створює окремий виділений потік для кожного підключення клієнта. Зі збільшенням кількості КЛІЄНТІВ це може призвести до проблем із блокуванням, оскільки це збільшує кількість процесів, яким може знадобитися чекати іншого процесу, щоб звільнити ресурси (пам’ять, центральний процесор), які зараз зайняті. Крім того, розробка окремих процесів для виробництва для обмеження апаратних ресурсів. З іншого боку, Nginx не створює окремий потік для кожного процесу. Натомість він використовує головний потік та робочі процеси в межах одного основного потоку. Замість приписування процесу шкірному клієнту робочому процесу присвоюється подія, тобто запит на ресурси від клієнта. Тому кожний робочий процес не може бути складним для обслуговування декількох сотень клієнтів, навіть з мінімальними обсягами пам'яті та центрального процесора, що робить Nginx дуже ефективним для великих сайтів.

Nginx обслуговує велику низку популярних у світі веб-сайтів з високою кісльстю людей, серед них є Netflix, Hulu, Pinterest, WordPress, Zynga і Airbnb.

Що стосується продуктивності, Nginx поєднує Apache з великим відривом. Незалежне тестування показало, що, слухаючи базову лінію (визначену як апаратне забезпечення та платформи операційної системи), Nginx може генерувати приблизно на 40% більше трафіку за 300% менше часу. Це робить Nginx у 4,2 рази більшим, ніж Apache.

Веб-сервер Apache

HTTP-сервер Apache - це веб-сервер з відкритим кодом, яким керує Apache Software Foundation. Серверне програмне забезпечення охоплює вільно, а ліцензія з відкритим кодом означає, що користувачі можуть редагувати вихідний код для підвищення продуктивності та сприяння подальшому розвитку програмного забезпечення. Підтримка, виправлення та розробка забезпечують обмін лояльними користувачами та координацію програми Software Foundation.

Хоча Apache буде працювати у всіх основних операційних системах (Linux, Windows, Unix, OSX, NetWare), ви також будете часто використовувати Linux. Вони,

у своїй спробі, з базовою базою даних MySQL та мовою програмування PHP, стануть більш популярними веб-версіями срібла.

Apache завжди був лідером на ринку з моменту свого розробка в 1996 році. Але як сказати стару приказку, коли ви знаходитесь на вершині єдиного шляху вниз. Одного разу можливі переговори Apache незручно перекривались протягом багатьох років з його постійно розвиваються конкурентами; Майкрософт IIS - майже номер 2, а Nginx створює першого з кількох молодих кандидатів на перше місце. Частка ринку Apache в 2010 році впала з 60% до 54% у червні 2013 року та 38% у лютому 2014 року. Однак Фонд програмного забезпечення Apache прагне підтримувати конкурентоспроможність своєї продукції, запускаючи додаткові компоненти для нових функцій, а також нові розширення для реалізації майстер-потоку, яке значно відрізняє Nginx від Apache.

Як зазначалося вище, Apache працює набагато повільніше, ніж Nginx. Але це ще не вся історія, оскільки вони є статистикою для статичних сторінок. Apache найкраще працює під час завантаження та оновлення динамічних сторінок. l Apache безперечно виграє, якщо поверне кількість функцій, що підтримуються веб-серверами. Багато функцій Apache реалізовані у вигляді загальних модулів, що розширюють функціональність. Деякі з цих функцій та модулів:

- підтримка мови програмування на сторонніх серверах;
- загальна підтримка мовних інтерфейсів (наприклад, Perl, Python, Tcl та PHP);
- модулі автентифікації: mod_access, mod_auth та mod_auth_digest;
- модуль підтримки SSL / TLS mod_ssl;
- проксі-модуль - mod_proxy;
- URL-адреса перезапису - mod_rewrite;
- файли журналу користувача - mod_log_config;
- підтримка фільтрів - mod_include та mod_ext_filter;
- Стискання веб-сторінок: mod_gzip.

Тестування

Було проведено тестування серверів Nginx та Apache. Для одночасних запитів від 150 користувачів, час відгуку сервера значно зріс, і на сторінках, які передають більше інформації, сервер видає помилки після очікування 70 секунд. Під час тестування Apache є швидким [2].

Таблиця №1

Тестування серверів

К-ть запитів	Nginx (час мс)	Apache (час мс)	Nginx (відмови)	Apache (відмови)
50	458	779	0	0
100	449	464	0	0
150	545	505	0	0
200	2085	1738	0	0
240	9686	5254	0	182
300	18126	8340	0	300
350	24059	11021	2	370
420	28460	13561	45	420
450	28866	15978	92	470
500	29668	17378	182	530

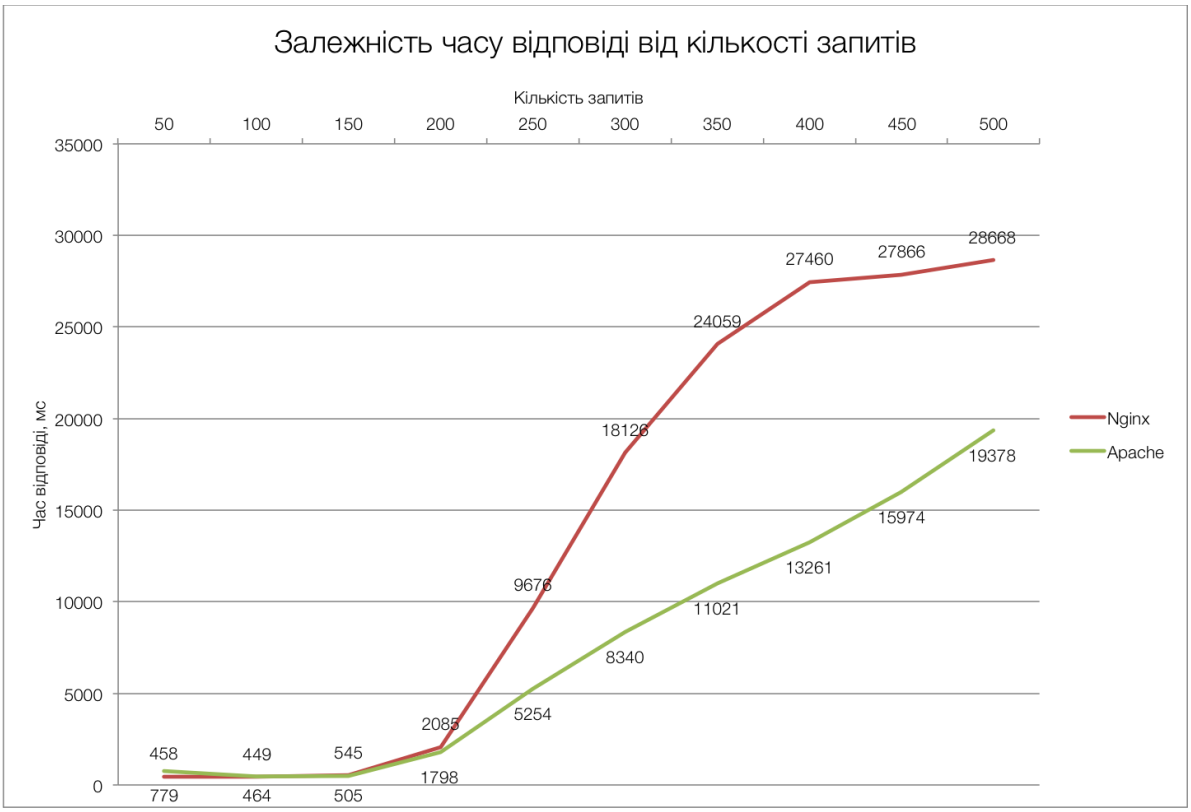


Рис. 2.1.1 – Залежність часу відповіді сервера від всіх запитів

Рис. 2.1.2. Залежність кількості відмов від кількості запитів

На графіках видно що, при великій частоті запитів швидкість видачі сторінок клієнтам сервера Apache швидче ніж Nginx, але для сервера Nginx дія відмови в сотні разів менша.

Порівняльний висновок

Apache та Nginx - найближчі конкуренти. Перевагою Nginx є висока швидкість і швидкість запису. Apache не можна порівняти з його роботою, але він повинен створити організаційну підтримку (Apache Software Foundation) і має більше можливостей (включаючи документацію), не створюючи ексклюзивного спільного користувача. lмпульс від Nginx, Apache підтримується в Windows.

Наприклад, ви можете використовувати Nginx Apache перед якістю зворотного проксі-сервера, щоб підтримувати постійний вміст для SSL або інших комбінацій або конфігурацій за потреби [3].

2.2. Порівняння фреймворків мови php

Розробники знають, що правильний фреймворк дає змогу робити додатки швидке та комфортніше. При обранні правильного фреймворку, найперше потрібно вирішити значення для надійності й успіху. Фреймворки PHP - супер корисні інструменти для веб-розробки, вони зменшують час для розробки , коли справа доходить до розробка і підтримки веб-сайту PHP. Склад Фреймворка:

- Набір інструментів - набір готових швидко інтегрованих програмних компонентів. Це означає менший ризик помилок і менше коду для написання. Це також означає, що ви будете більш продуктивними і зможете витрачати більше часу;
- Методологія - Застосування «схема складання». Структурований підхід на перший погляд може здатися обмеженням. Але насправді розробники можуть ефективно вирішувати найскладніші аспекти своїх завдань, а найкращі практики забезпечують стабільність, ремонтопридатність та модернізацію програми, що розробляється. На ринку існує так багато

апаратних PHP-фреймворків, що, мабуть, важко вибрати один. Кожен фреймворк має як сильні, так і слабкі сторони.

Laravel: фреймворк PHP для веб-майстрів

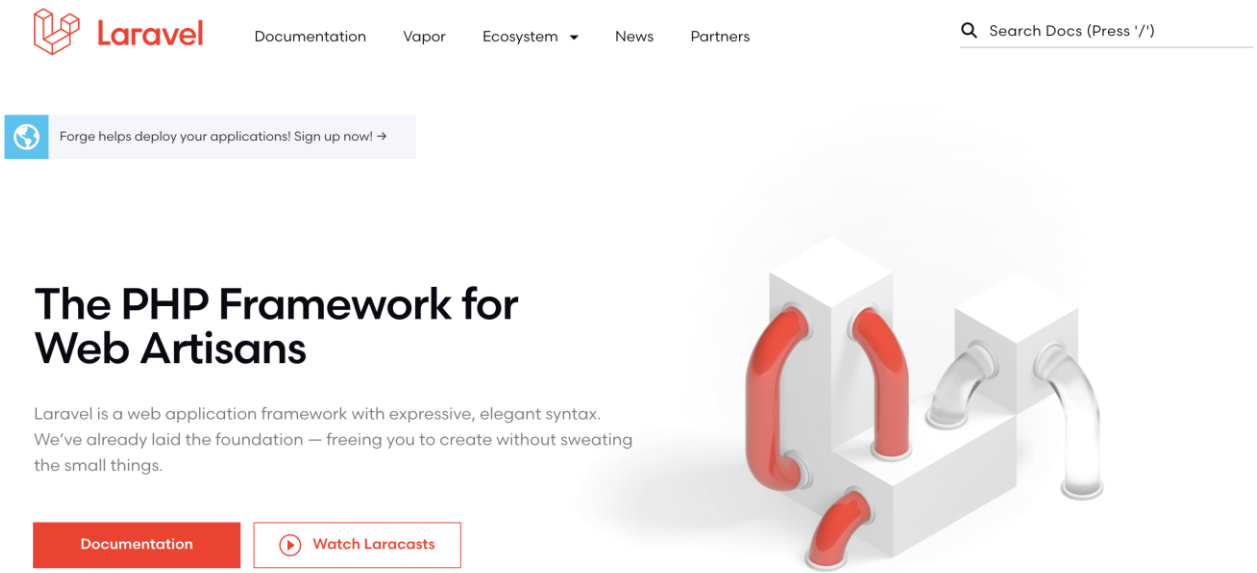


Рис. 2.2.1. знімок головною сторінки на сайті Laravel

Laravel - це безкоштовна платформа PHP з відкритим кодом, розроблена для розробки веб-додатків MVC. Вважається, що Laravel підняв фреймворк PHP на абсолютно новий рівень. Laravel допомагає створювати чудові програми з простим та виразним синтаксисом для полегшення загальних завдань, таких як автентифікація, маршрутизація, сеанси та кешування. Laravel легко читати та добре документувати, щоб пришвидшити написання коду. Composer дозволяє вам керувати всіма сторонніми пакетами вашої програми та добре працює з MySQL, Postgres, SQL Server та SQLite.

Phalcon

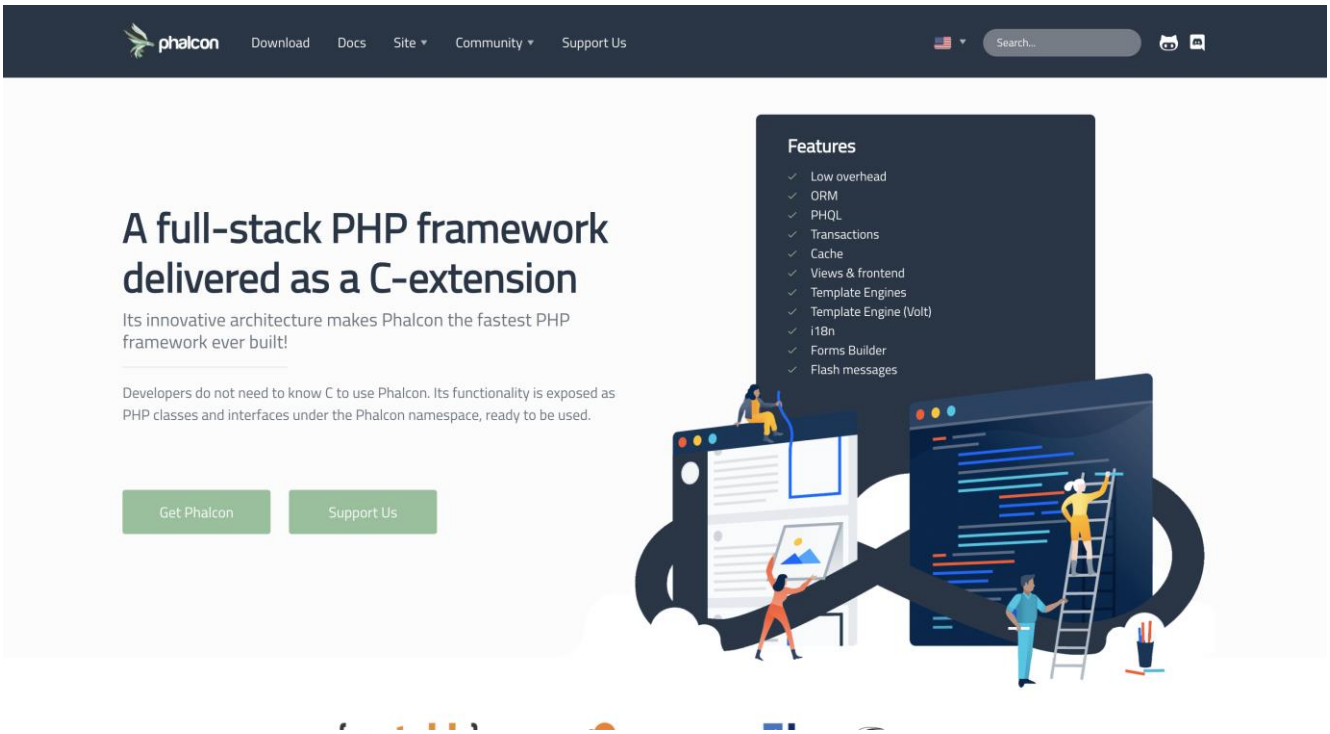


Рис. 2.2.2. Знімок дисплею головної сторінки Phalcon

Phalcon - це найшвидший фреймворк. Побудований на C, але наданий як розширення PHP, не існує компромісів у швидкості. Це забезпечує високу продуктивність та низьке використання ресурсів. Вам не потрібно вчитися чи використовувати C, ця функція доступна як готовий до використання клас PHP. Створюючи насичений повнофункціональний фреймворк, повністю написаний на мові C та упакований як розширення PHP, Phalcon може заощадити час процесора та покращити загальну продуктивність.

До цього часу продуктивність не вважалася головним пріоритетом для розробки веб-додатків. Нове обладнання може це компенсувати. Але коли Google почав враховувати швидкість рейтингу пошуку, продуктивність стала головним пріоритетом поряд із функціональністю. Це ще один спосіб покращення веб-ефективності може мати позитивний вплив на ваш веб-сайт.

Symfony 2

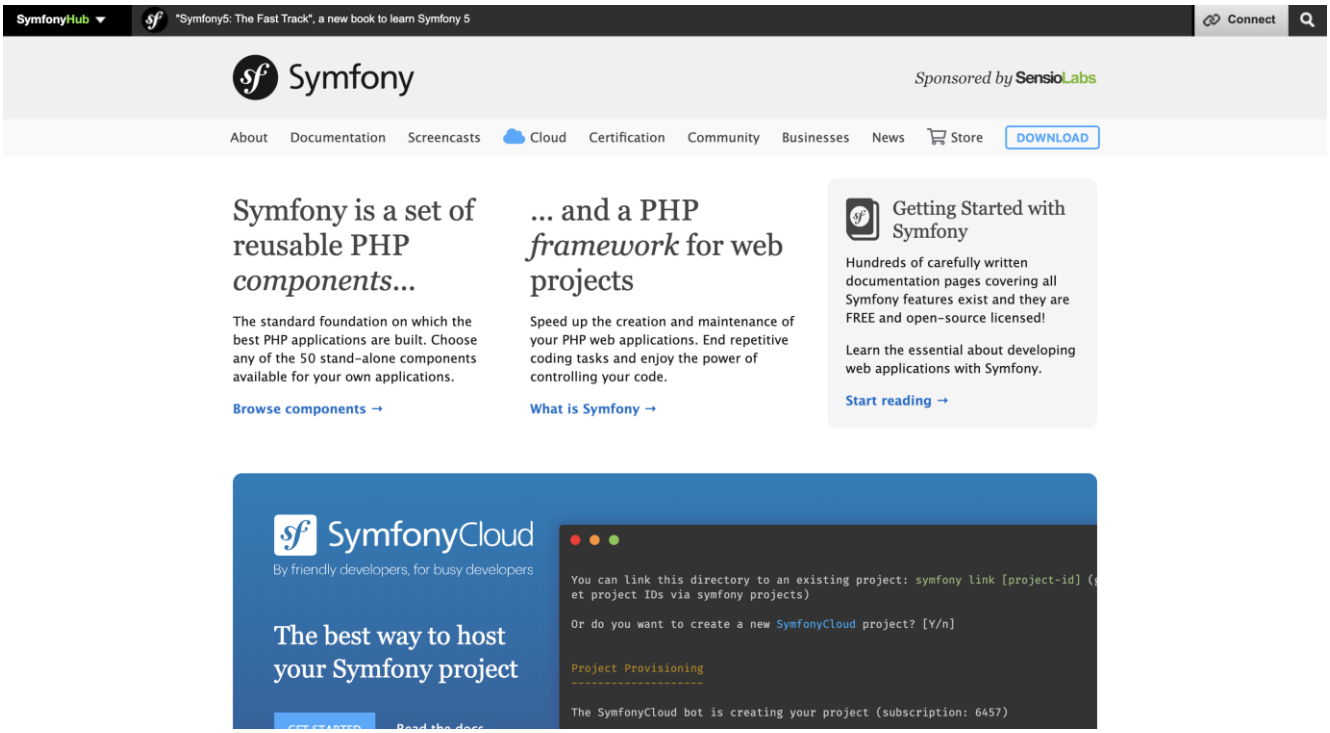


Рис. 2.2.3. Знімок дисплею головної сторінки Symfony 2

Symfony - це фреймворк PHP для веб-проектів. Це пришвидшує розробка та обслуговування веб-додатків PHP. Замініть повторювані завдання кодування потужністю, контролем та радістю.

Symfony - це набір багаторазових компонентів PHP та баз даних PHP для ваших веб-проектів. Це добре задокументована безкоштовна ліцензована структура MIT. Drupal, одна з найпопулярніших систем управління вмістом, та PHPBB, одна з найбільш часто використовуваних систем форуму, використовують Symfony.

Потужний, масштабований та гнучкий. Однак багато людей, особливо ті, хто не знайомий з рамками, вважають поріг входу високим.

Це певною мірою відповідає дійсності. На перший погляд, моделі, подання, контролери, установи, сховища, маршрути, шаблони тощо можуть бути дуже страшними та заплутаними.

Yii Framework

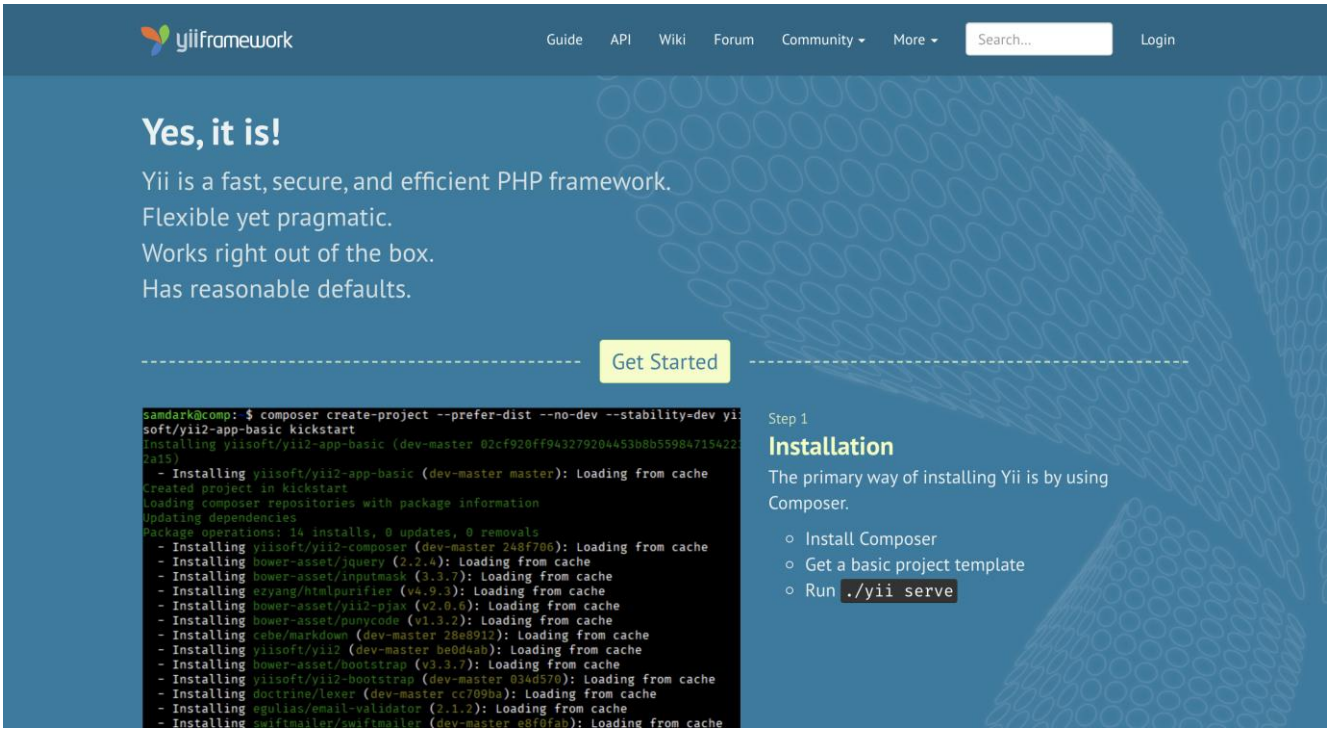


Рис. 2.2.4. Знімок дисплею головної сторінки Yii Framework

Yii - це швидкий та безпечний професійний фреймворк. Це високопродуктивний фреймворк PHP, ідеально підходить для розробки веб-додатків 2.0. Yii має безліч MVC, DAO / ActiveRecord, I18N, кешування, аутентифікації, контролю доступу на основі ролей, тестування, відкритого коду, високої продуктивності, об'єктно-орієнтованих, об'єктів доступу до бази даних, простої перевірки форми, підтримки веб-служб за замовчуванням та іншого Особливості включені. Більше Yii Framework ідеально підходить для розробки веб-сайтів у соціальних мережах, що значно скорочує час розробки.

CodeIgniter

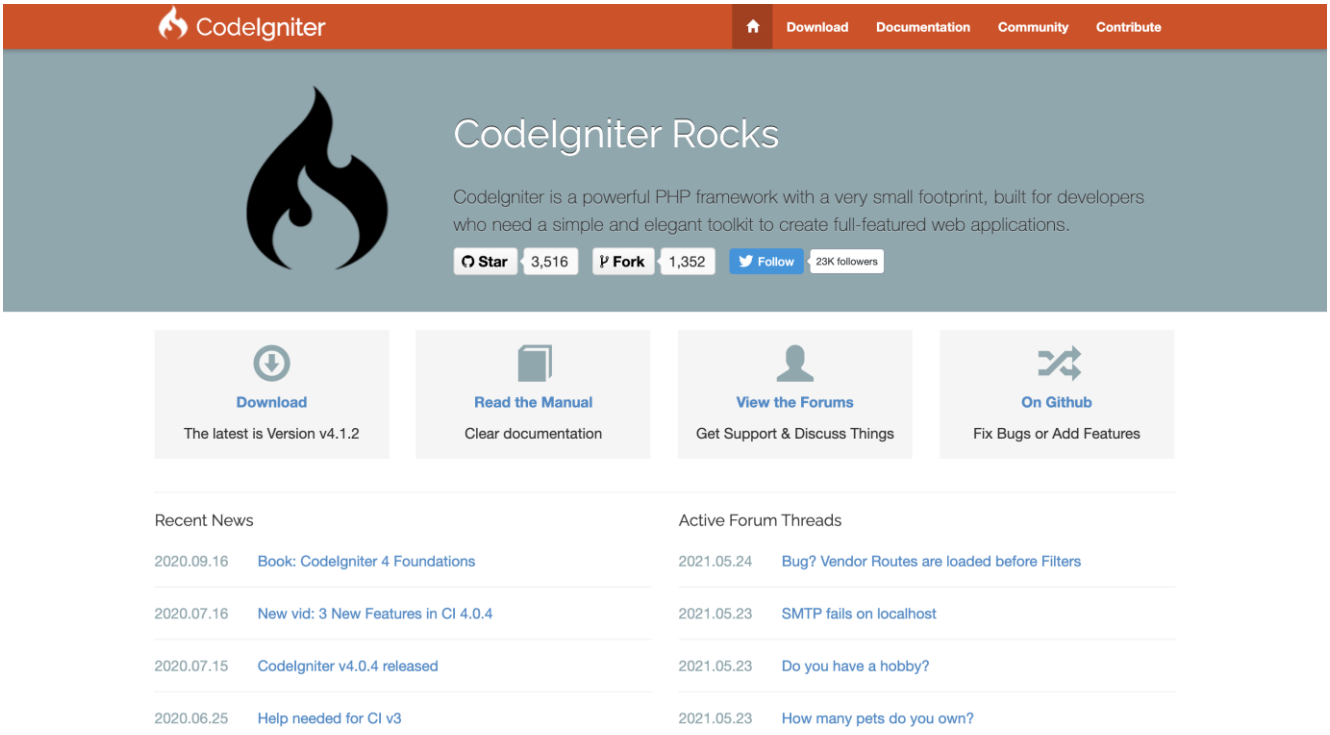


Рис. 2.2.5. Знімок дисплею головної сторінки CodeIgniter

CodeIgniter - це перевірений, невеликий, гнучкий і відкритий фреймворк. Це буде стимулювати веб-додатки наступного покоління. CodeIgniter - це дуже компактна і потужна інфраструктура PHP, створена для кодерів PHP, яким потрібен простий та елегантний набір інструментів для розробка повнофункціональних веб-додатків.

Cake PHP

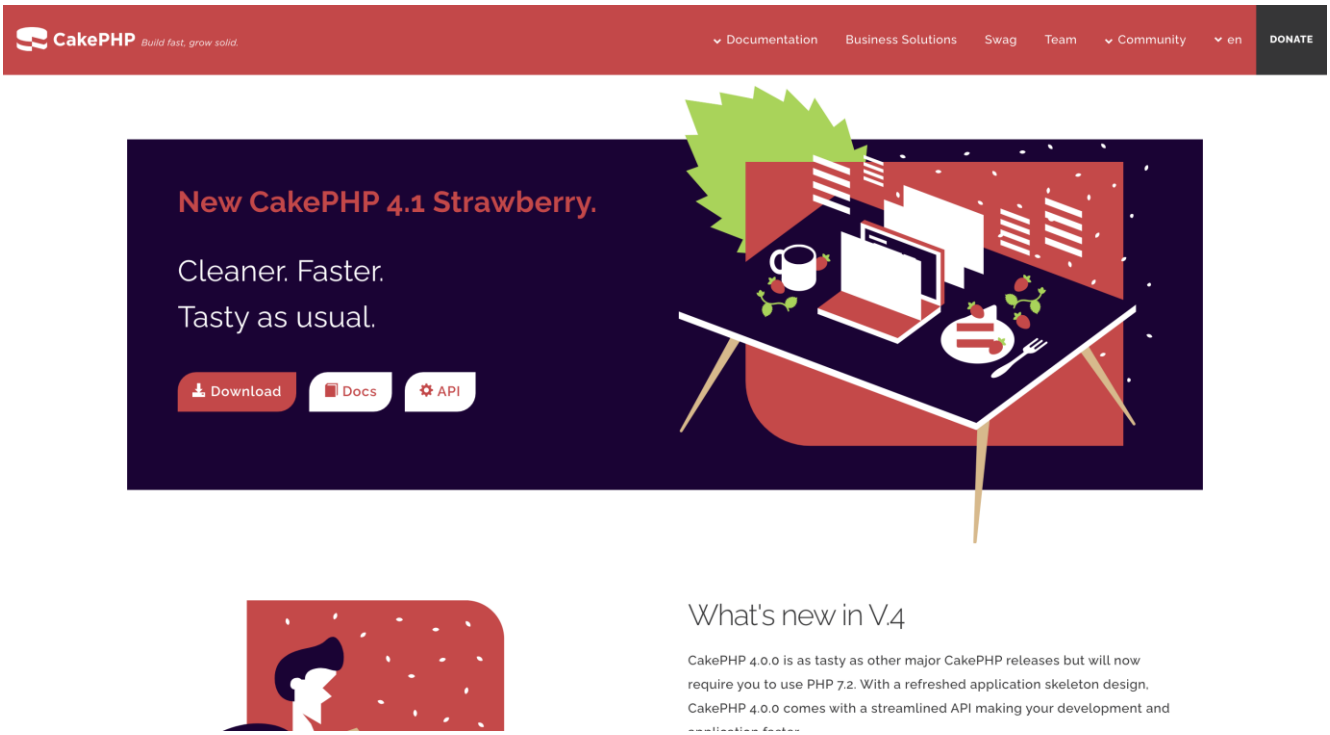


Рис. 2.2.6. Знімок дисплею головної сторінки CodeIgniter

CakePHP - одна з основ, яка сформувала мейнстрім у 2014 році. Вони планують випустити стабільну версію 3.0 для широкого загалу. CakePHP спрощує, чіткіше створює веб-програми та вимагає менше коду. Це дозволило йому швидко прототипувати, використовуючи можливості генерації коду та лісів. Не потрібно додаткових файлів конфігурації XML або YAML. Просто встановіть базу даних, і ви готові до випічки. CakePHP поширюється за ліцензією MIT, що робить його ідеальним для повсякденного використання. Те, що вам потрібно, вже вбудовано. Переклад, доступ до бази даних, кешування, перевірка, автентифікація тощо побудовані на одній з перших структур PHP MVC. CakePHP постачається із вбудованими інструментами для перевірки введення користувачем, захисту CSRF, запобігання фальсифікаціям, запобігання ін’єкціям SQL та запобігання XSS, щоб допомогти зберегти вашу програму надійною та безпечною.

Aura

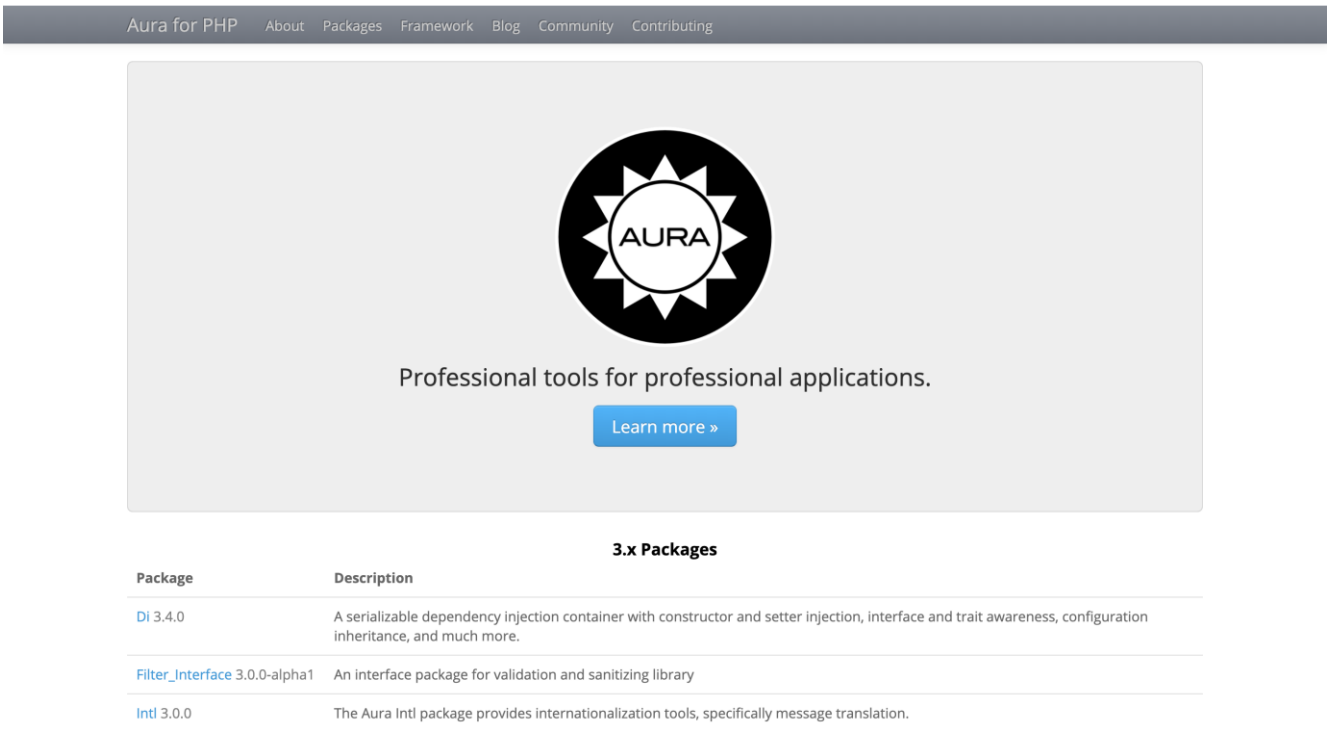


Рис. 2.2.7. Знімок дисплею головної сторінки Aura

Проект PHP Aura створений для людей, які люблять чистий код, повністю не пов'язані бібліотеки та справді незалежні пакети. Ця структура має досить багато користувачів, і її також використовує CakePHP. Завантажте один пакет і почніть використовувати його у своєму проекті вже сьогодні, не додаючи ніяких залежностей. Основна мета Aura - забезпечити високоякісну, добре перевірену, сумісну зі стандартами бібліотеку ізоляції, яку можна використовувати в будь-якій основі коду. Це означає, що ви можете використовувати стільки проектів, скільки вам потрібно.

Zend Framework

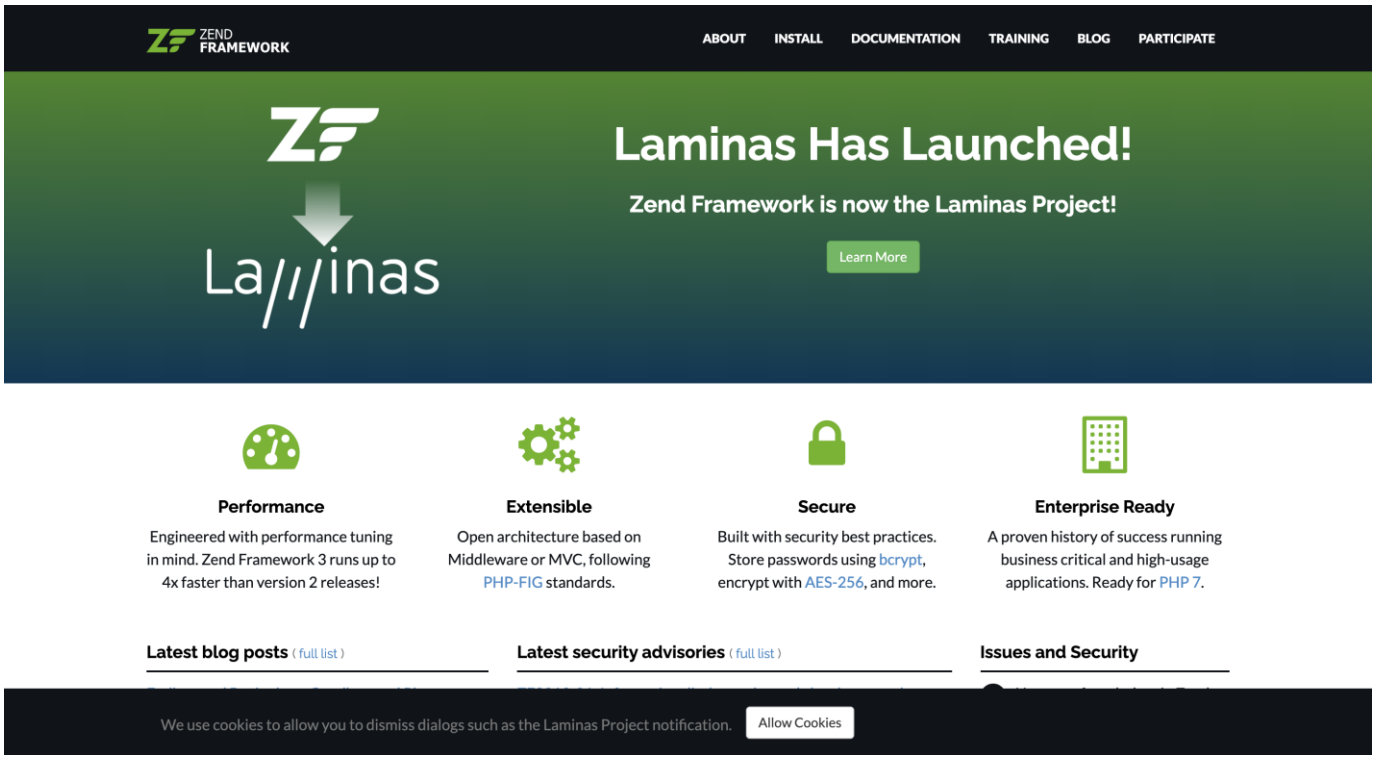


Рис. 2.2.8. Знімок дисплею головної сторінки Zend Framework

Протягом багатьох років це одна з провідних платформ із гнучкою архітектурою, придатною для сучасних веб-додатків.

Zend Framework розроблений з урахуванням простоти. Легкий, неважкий у налаштуванні, він зосереджений на найбільш загальних та необхідних функціях. Він побудований для значного прискорення навчального процесу, коли ви переходите до нових рамок. Широко використовується, він добре перевірений і безпечний.

Kohana

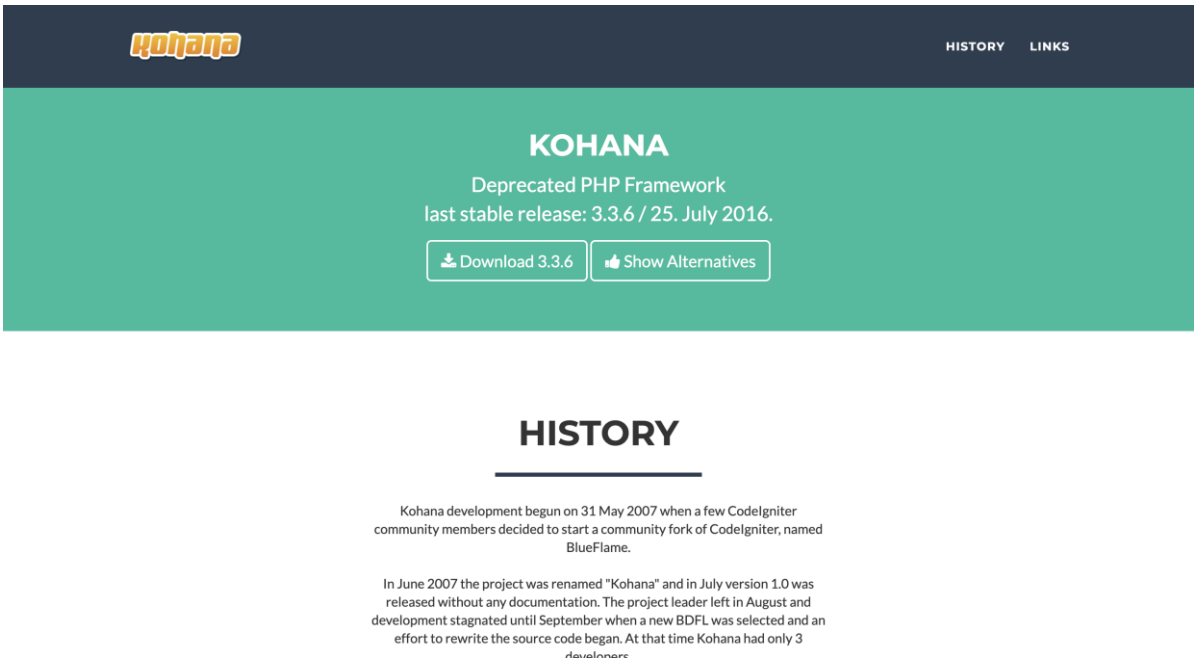


Рис. 2.2.9. Знімок дисплею головної сторінки Kohana

Kohana - це елегантний фреймворк з багатим набором функцій для розробка веб-додатків. Він включає безліч вбудованих компонентів, таких як засоби перекладу, доступ до бази даних, профілювання коду, шифрування та перевірку, завдяки чому ви можете швидко створювати веб-програми. Існують також хороші інструменти налагодження та профілювання, які можуть допомогти вам вирішити проблеми, які часто займають багато часу.

Kohana побудований як дуже швидкий фреймворк PHP і ретельно оптимізований для практичного використання. Однак деякі тести показали, що він повільніший, ніж CodeIgniter та інші основні гравці.

Це дуже сухий об’єктно-орієнтований фреймворк. Все це побудовано з використанням строгих класів та об’єктів PHP5. Це показує, що Kohana дотримується найкращих нових можливостей PHP і ретельно оптимізує свій основний код.

Slim

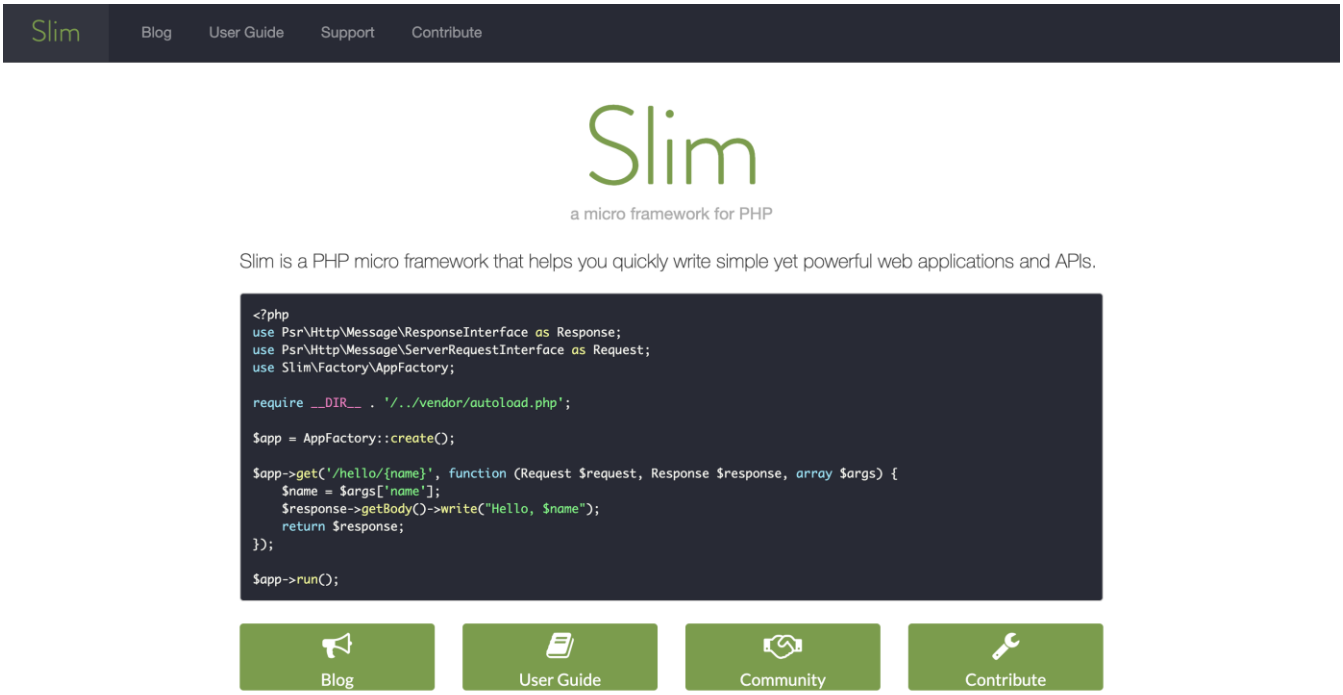


Рис. 2.2.10. Знімок дисплею головної сторінки Slim

Slim - фреймворк розроблений для того, щоб бути дуже легким і тонким. Це мікроструктура, яка дозволяє швидко створювати прості, але потужні веб-програми та API. Він має потужний маршрутизатор, шаблони візуалізації з перегляду користувача, флеш-повідомлення, захищені файли cookie з шифруванням AES-256, кешування HTTP, реєстрацію в журналах користувачів, обробку помилок та налагодження та просту конфігурацію.

Fuel PHP

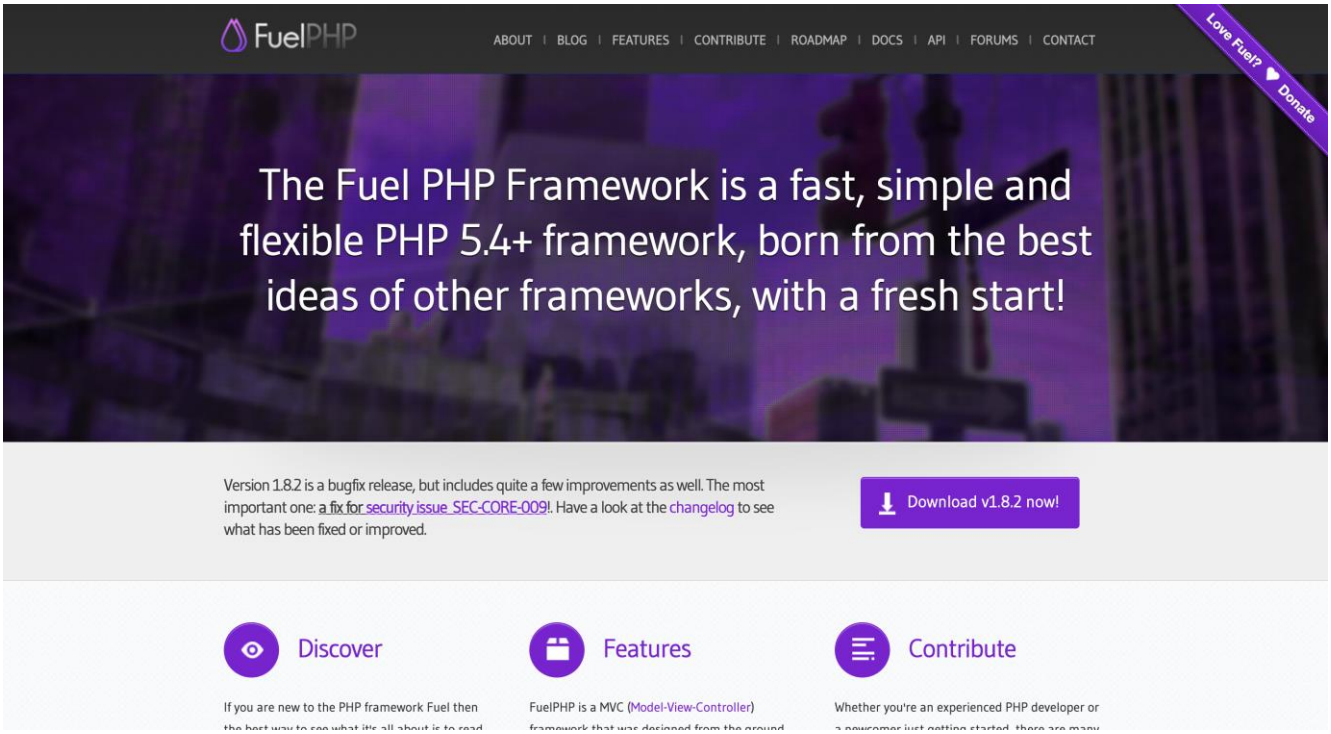


Рис. 2.2.11. Знімок дисплею головної сторінки Fuel PHP

Fuel - це проста, гнучка спільнота PHP 5.3+, заснована на найкращих ідеях інших платформ. Наразі випущена версія 2 фреймворку, яка зараз перебуває на стадії бета-тестування.

FuelPHP - це платформа MVC (Model-View-Controller), розроблена з нуля, щоб повністю підтримувати HMVC у своїй архітектурі. Але розробники на цьому не зупинились і додали до суміші ViewModels (також відомі як моделі перегляду). Це дозволяє додавати потужні шари між контролером та видом.

Flight

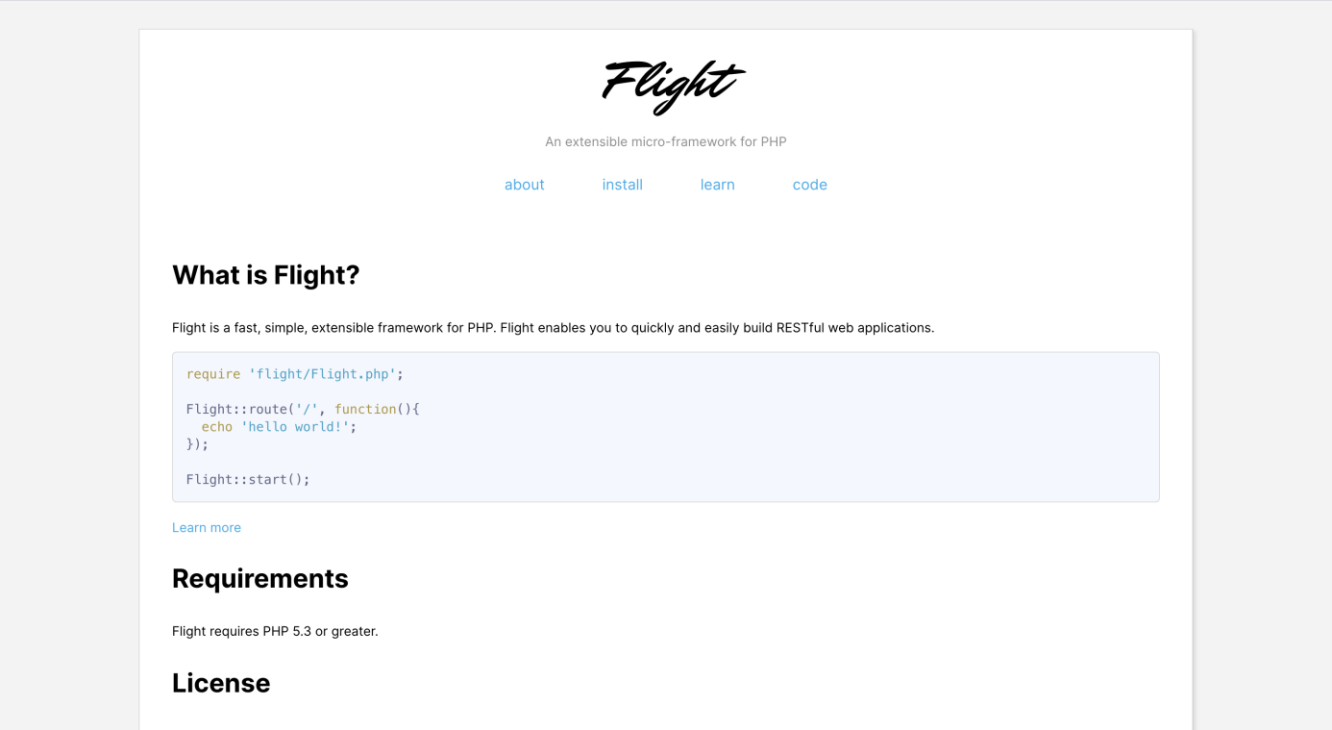


Рис. 2.2.12. Знімок дисплею головної сторінки Flight

Flight - це розгалужений мікрофреймворк для PHP. Flight - це швидкий, простий і розширюваний фреймворк для PHP. Політ дозволяє швидко і легко створювати RESTful веб-додатки. Потрібен PHP 5.3 + і випускається за ліцензією MIT.

Medoo

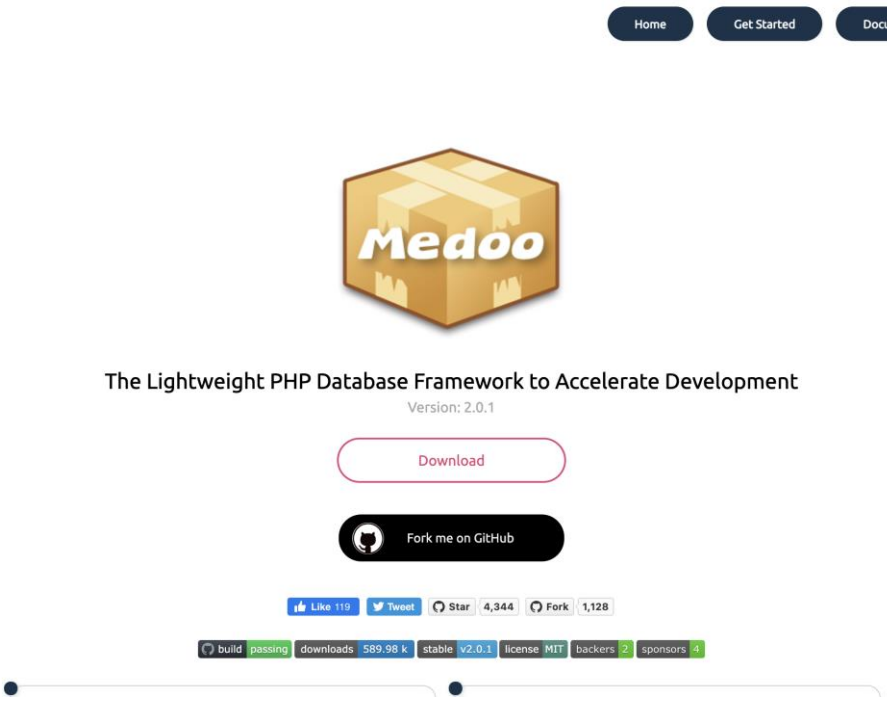


Рис. 2.2.13. Знімок дисплею головної сторінки Medoo

Medoo - це найлегший фреймворк бази даних PHP, призначений для прискорення розвитку. Такий самий легкий, загалом 13 КБ на файл. Це дуже просто створити та використовувати, і він сумісний з різними даними бази даних SQL, такими як MySQL, MSSQL, SQLite, MariaDB, Oracle, Sybase, PostgreSQL та ін.

PHPixie

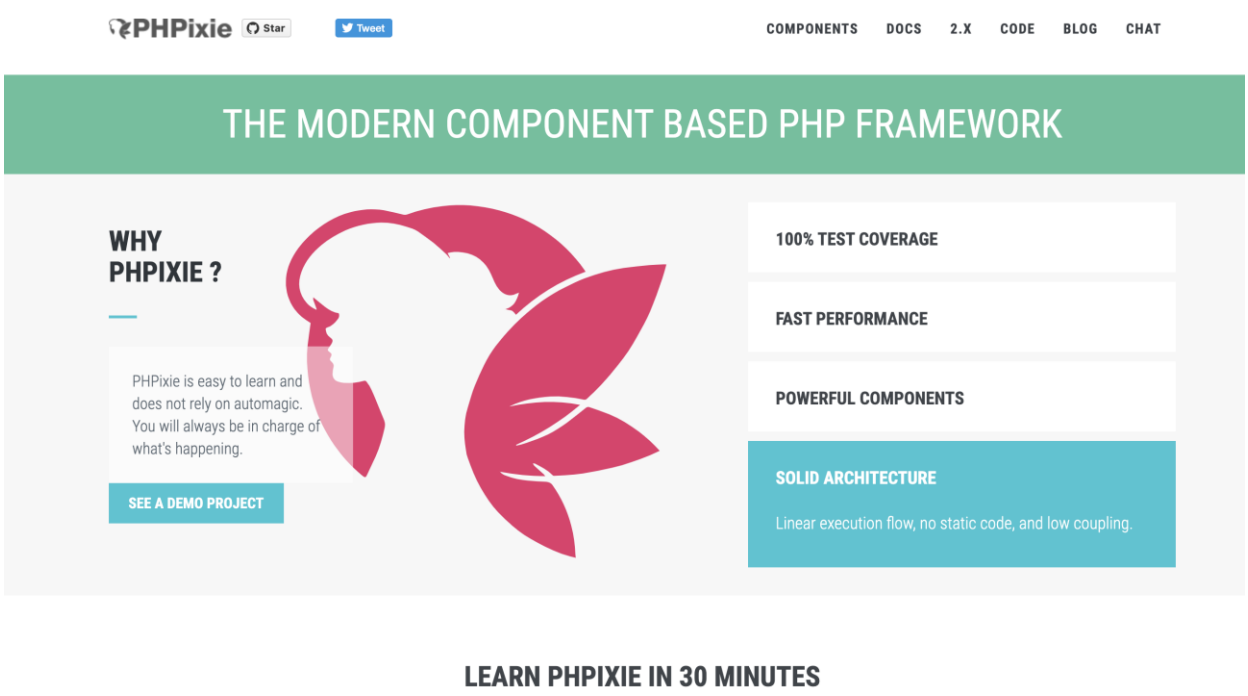


Рис. 2.2.14. Знімок дисплею головної сторінки PHPixie

PHPixie - це легкий фреймворк PHP MVC, розроблений для швидкого, простого в освоєнні та розробка надійної основи для розвитку. Він дуже легкий, добре задокументований і містить безліч правил таким чином, що принаймні потребує коригування.

PHPixie використовує спрощену реалізацію потоку запитів / відповідей, але все одно дозволяє використовувати більш складні структури, такі як HMVC. PHPixie обробляє набагато інакше, ніж повний фреймворк. Наприклад, більшість інших фреймворків, які використовують автозавантажувачі для завантаження класів, змушені використовувати простори імен. Тому вам потрібно розмістити клас у папці відповідно до імені класу, щоб ім'я класу в / driver / mysql / query.php було Driver_Mysql_Query. PHPixie змінює цей порядок так, що ім'я класу має значення Query_Mysql_Driver. Це полегшує читання, оскільки перше слово в назві класу насправді краще описує клас. Ця невелика зміна може значно покращити вихідний код презентацій для великих проектів.

Pop PHP

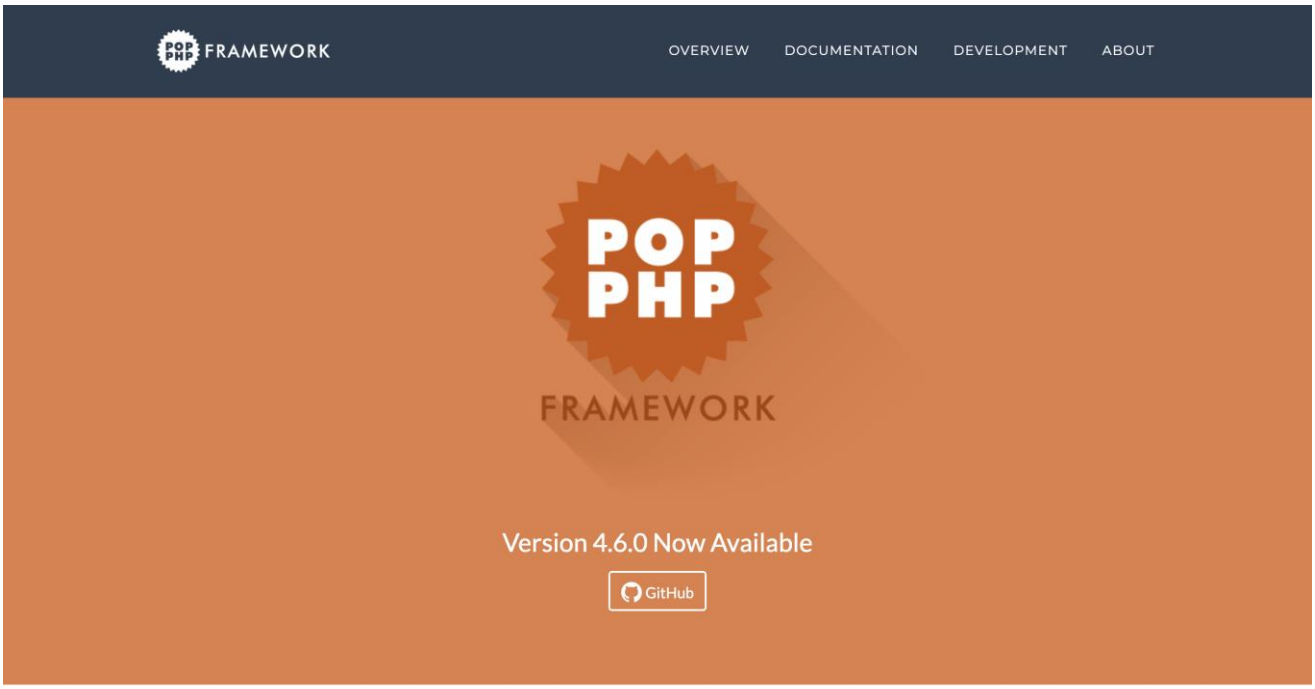


Рис. 2.2.15. Знімок дисплею головної сторінки Pop PHP

Pop PHP Framework - це простий у використанні PHP фреймворк з надійним і докладним API. Підтримує PHP 5.3+. Сьогодні Pop PHP Framework зберігає свою простоту і все ще дуже легкий. У нього також вбудовано багато нових функцій, але використання фреймворку як просто набору інструментів або як базового фреймворку для вашої програми поки не складно.

Simple MVC Framework

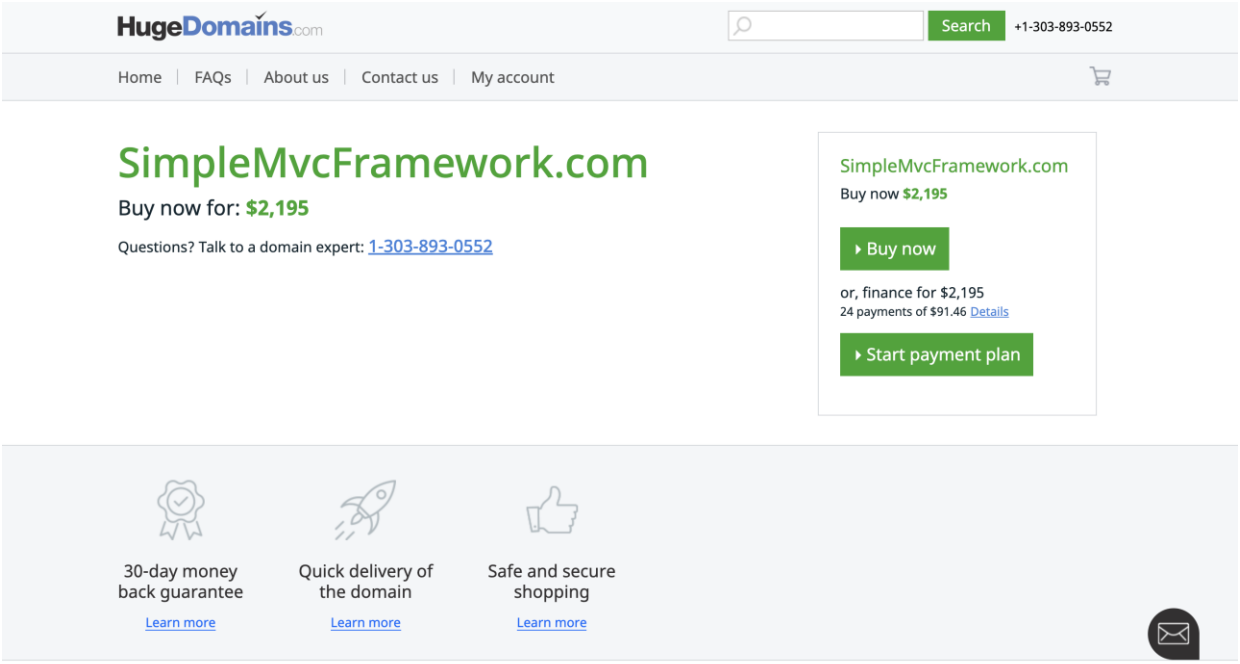


Рис. 2.2.16. Знімок дисплею головної сторінки Simple MVC Framework

Simple MVC Framework дуже простий у налаштуванні, має чітку структуру кодування та призначений для зворотного навчання. Фреймворк можна встановити лише вказавши шлях до сайту. Доступний простий файл теми, який дає вам повний контроль над зовнішнім виглядом теми, що дозволяє швидко змінити зовнішній вигляд вашої програми / сайту. Зв'язок MySQL забезпечує помічник PDO, але, звичайно, ви можете замінити його Mysqli, Medoo або іншою базою даних. Підтримка надається через різні джерела, Twitter, групи Facebook та спеціальні форуми. Легкий і менше 1 Мб

Typo3 Flow

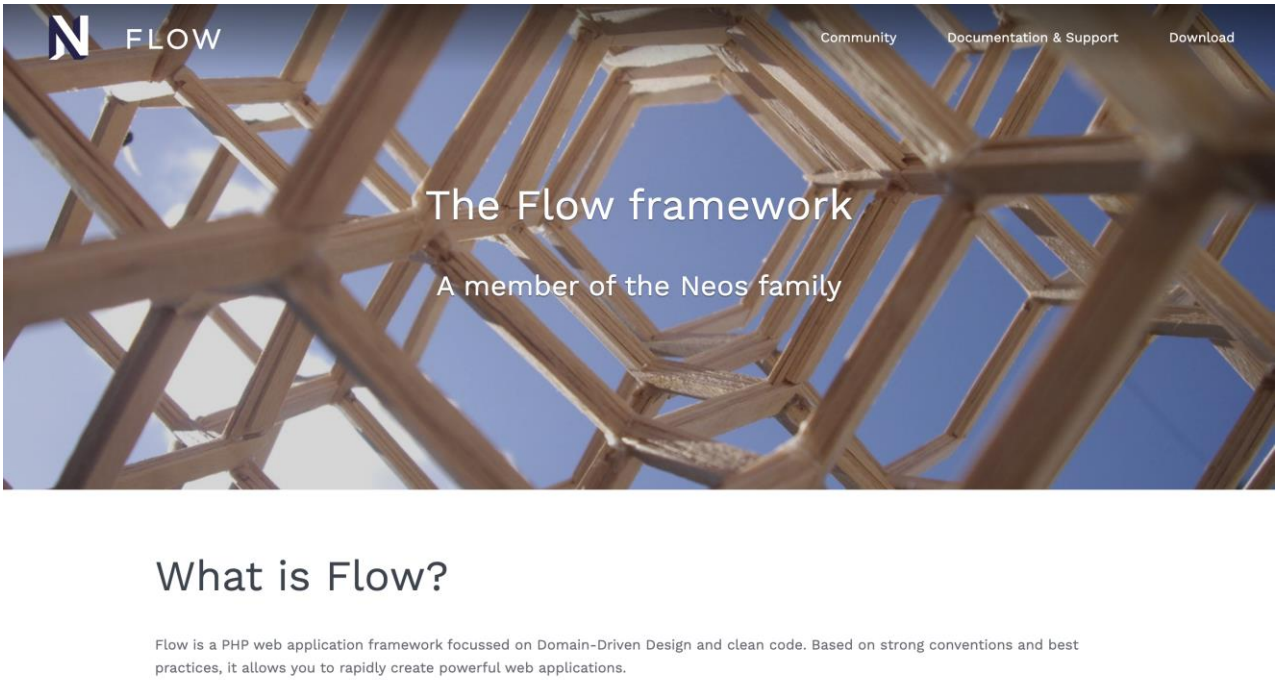


Рис. 2.2.17. Знімок дисплею головної сторінки Typo3 Flow

TYPO3 Flow - це платформа / веб-програма, яка дозволяє розробникам створювати чудові веб-рішення та повертати задоволення від кодування.

Це дає швидкі результати. Це міцна основа для складних застосувань. Потік TYPO3 спирається на TYPO3, одну з найбільших спільнот PHP.

Nette

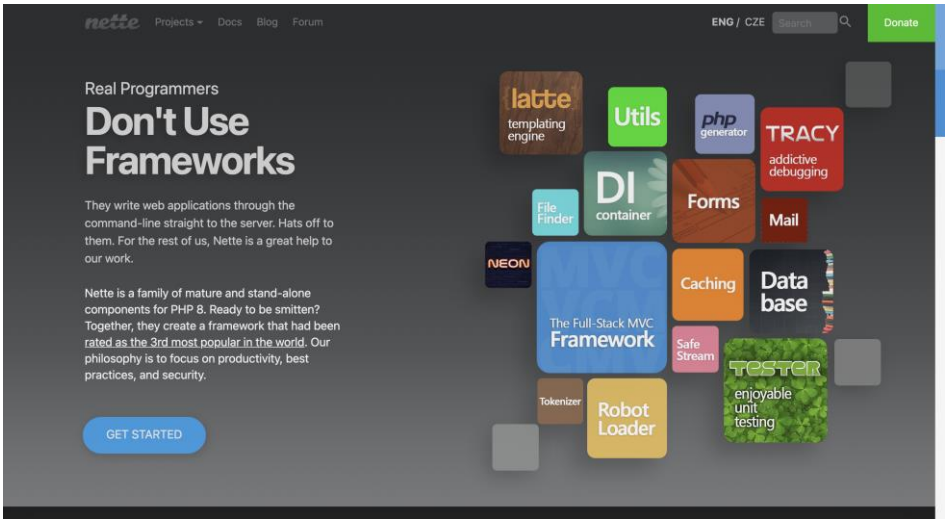


Рис. 2.2.18. Знімок дисплею головної сторінки Nette

Популярний інструмент для веб-розробки PHP. Він розроблений, щоб бути максимально комфортним і, можливо, одним із найбезпечніших. Він розмовляє вашою мовою та допомагає створювати найкращі веб-сайти. Nette використовує такі інноваційні технології, як XSS, CSRF, захоплення сеансу та блокування сеансу, щоб усунути прогалини в безпеці та неправильне використання. Він має відмінні інструменти налагодження, які допоможуть вам негайно знайти всі помилки. Nette - це сучасний фреймворк, який підтримує AJAX / AJAJ, Dependency Injection, SEO, DRY, KISS, MVC, Web 2.0 та круті URL-адреси. Широко підтримується для всіх передових технологій та концепцій.

Silex

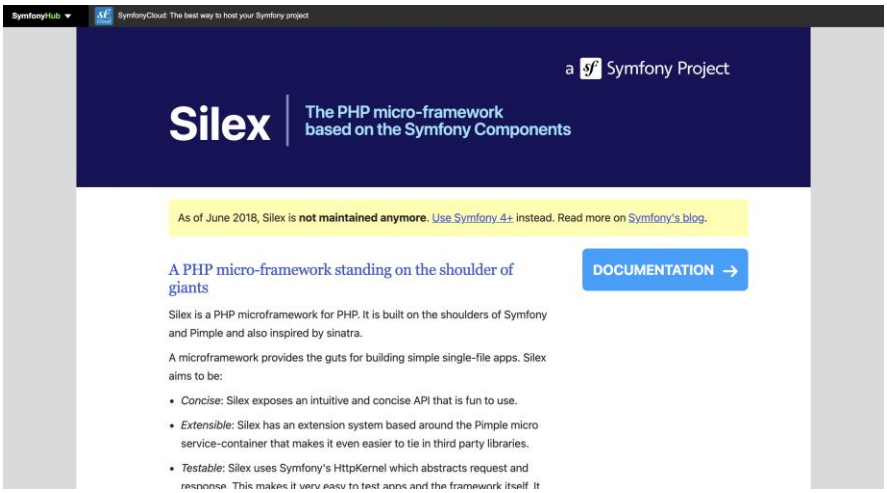


Рис. 2.2.20. Знімок дисплею головної сторінки Silex

Silex - це PHP-мікрофреймворк, заснований на компонентах Symfony2.

Silex - це мікрофреймворк PHP для PHP 5.3. Побудований на плечах Symfony2 та Pimple, він також натхненний Sinatra.

Мікрофреймворк надає можливість створювати прості однофайлові програми. Коротко: Silex надає простий у використанні, інтуїтивно зрозумілий та стислий API. Розширюваний: Silex має системні розширення для контейнерів мікропослуг Pimple, що робить його ще простішим для додавання до сторонніх бібліотек.

Перевірено: Silex використовує SymfonyHttpKernel, який абстрагує запити та відповіді. Це дозволяє дуже просто застосовувати і саму структуру. Він також поважає специфікацію HTTP та заохочує її належне використання [4].

Кращий тест продуктивності фреймворку

Існує багато припущень щодо ефективності різних систем. У цьому контексті ми часто читаємо та чуємо тверді думки щодо переваг X над Y. Деякі компанії пишуть нову структуру PHP з нуля, оскільки доступні рішення занадто повільні.

Представлення репрезентативної вибірки навантаження кожного фреймворку - завдання непросте. Існує кілька способів використання кожного. Для кожного випадку використання отримуються різні докази. Розглянемо маршрутизацію як приклад. Zend Framework 1 за замовчуванням не вимагає файлу маршрутизації. Ми

вже реалізували шаблон "/ контролер / дія". Symfony2, навпаки, поставляється з конфігурацією маршрутизації. Вам потрібно прочитати та проаналізувати файл. Очевидно, що є додаткові цикли процесора, чи означає це, що маршрутизація Symfony2 повільніша, ніж Zend Framework 1? Відповідь (очевидно) - ні.

Фреймворк (скелет) кожного фреймворку, який використовує Apache Benchmark, тестується. Усі віртуальні хости мали одне і те ж правило mod_rewrite. AllowOveride встановлено на None. Запити за секунду від тестів Apache C = 20 та N = 500.

Framework	Req/Sec
Phalcon	822.96
Slim	399.83
Kohana	217.34
Code Igniter	187.78
Silex	179.01
Laravel	135.9
YII	123.5
Fuel PHP	116.34
Hazaar MVC	103.53
Zend Framework 1	103.02
Cake PHP	54.97
Nette	53.48
Symfony2	39.22
Zend Framework 2	36.1

Рис. 2.2.1. максимальне навантаження у кількості запитів на секунду

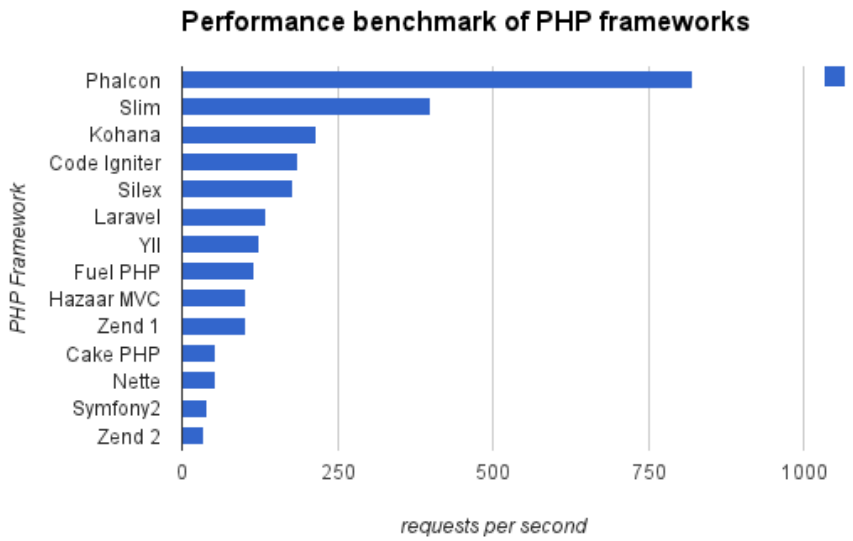


Рис. 2.2.21. Графік продуктивності фреймворків

Phalcon вражає всіх у списку, а Slim також є другим за швидкістю, оскільки це мікрофреймворк. У проекті швидкого запуску не використовувався чіткий для читання шаблон або макет. Zend Framework 1 вдвічі гостріший, ніж Symfony 2 та Zend Framework 2 [5].

Фреймворки повинні прискорити процес розробки, а продуктивність є другорядною. Zend Framework 2 і Symfony 2 можуть працювати краще, але це теж не погано. Є способи покращити ці цифри на виробничому сервері. Дослідіть і використовуйте саму структуру, а не винаходите колесо. Існує багато способів знайти баланс між продуктивністю та функціональністю.

На мою думку, першими п’ятьма фреймворками PHP сьогодні є Laravel, Phalcon, Symfony2, CodeIgniter та Yii.

Як показало те саме недавнє опитування на Sitepoint, на Laravel припадало понад 25%, на Phalcon - майже 17%, на Symfony2 - понад 10%, а на CodeIgniter та Yii 7,62%.

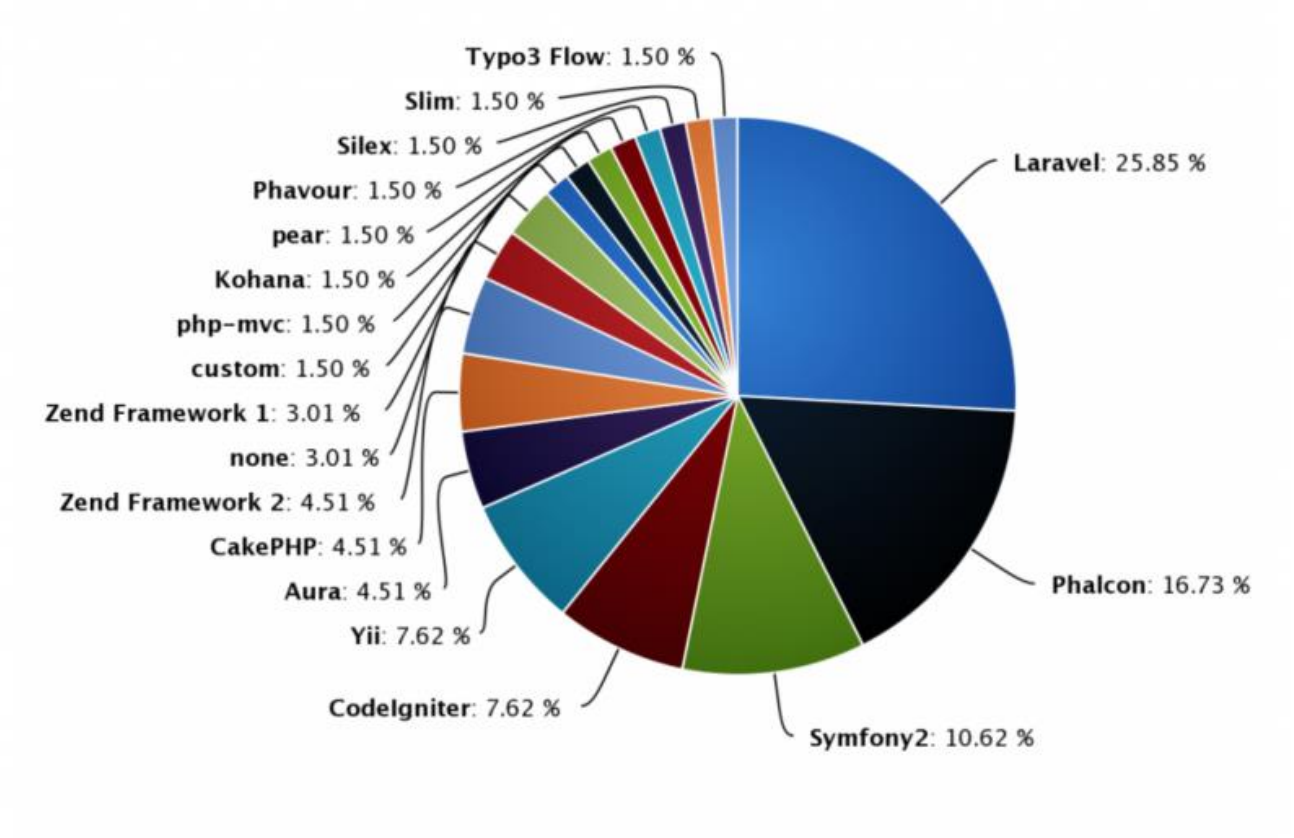


Рис. 2.2.22. Кругова діаграма популярності фреймворків у світі

РОЗДІЛ 3

Використання технології та їх можливості

3.1 Можливості фреймворку Symfony2

Symfony2 - це середовище, яке легко перевірити з наступних причин:

• Добре структурована, це дозволяє дуже легко знущатись над об’єктами, відокремлювати класи та забезпечувати „сірі” залежності, що робить модульне тестування дуже простим;

• Забезпечує перший рівень функціонального тестування (за допомогою PHPUnit). Будучи HTTP-орієнтованим фреймворком, він забезпечує базовий клас, який може імітувати HTTP-запити та перевіряти результати;

• lснує розширення Behat, яке дозволяє інтегрувати фреймворк із цим інструментом тестування поведінки.

Нарешті, ви можете побачити, як Symfony2 та навколишня його екосистема надають набір інструментів, придатних для тестування блоків, функцій та поведінки.

Symfony2 виходить за рамки попередника, забезпечуючи більш потужні та розширювані панелі та вбудовані профілі

Ви можете використовувати огляд бази даних, щоб зрозуміти, скільки запитів виконується, і побачити SQL для кожного запиту. Ви також можете отримати огляд часових витрат. Сам профайлер містить інформацію про кожен крок програми. Наприклад, Joomla, просто, ти бачиш, скільки часу пішло на розробка цього типу пам'яті і скільки він використовувався для виконання дій контролера?

Doctrine 2

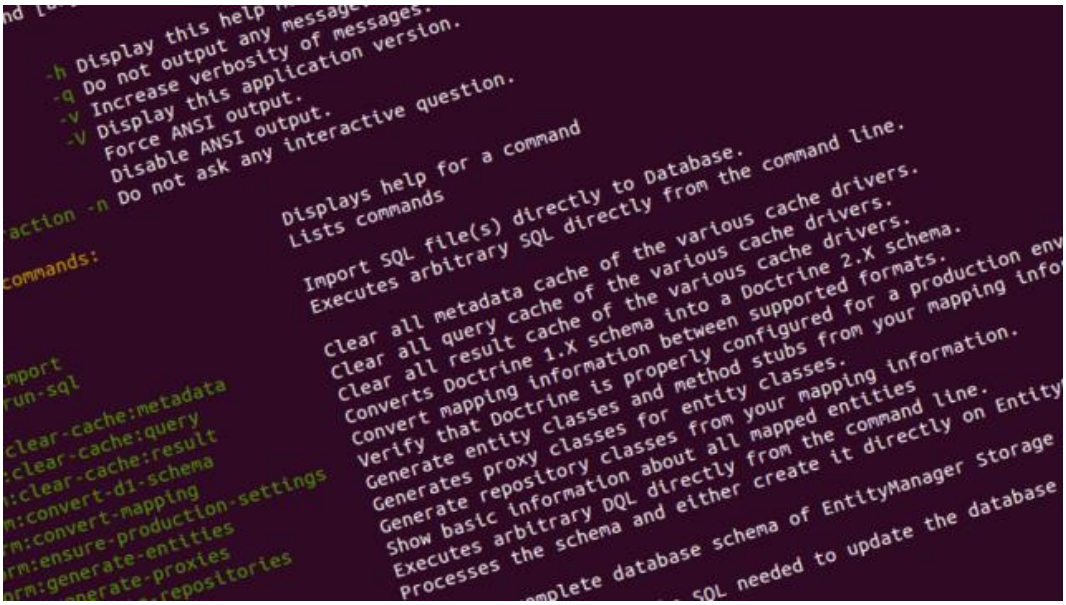


Рис. 3.1.2 – скріншот результату роботи консольною команди php app/console

Одним з емпіричних правил проектування SOA є можливість забезпечити доступ до одного і того ж джерела інформації з різних служб. Простіше кажучи, служби можуть отримувати доступ до даних, що зберігаються в інших місцях, замість того, щоб обмінюватися даними через веб-служби або черги повідомлень. Doctrine. 2 є важливим елементом, щоб десь отримати до неї доступ. По-перше, він забезпечує підтримку декількох з'єднань з базою даних та об'єктно-реляційних відображень. Ви можете безпечно використовувати цей інструмент разом із Symfony2 для обробки зчитування та запису в різні бази даних без забруднення. Модель предметної області для кожної послуги, що використовується Doctrine.

Розгортання

Symfony2 інтегрує коробку з Capistrano, найпопулярнішим інструментом автоматичного розгортання на ринку.

Це означає, що ви можете забути витратити свій час, гроші та енергію на розробку власного рішення для автоматизації розгортання будинку або, що ще гірше, покладаючись на ручні кроки, схильні до помилок.

Слідкування

Наприклад, припустимо, ви хочете використовувати Graylog2 для обробки журналів програм.

У мене немає сервера Graylog2 для підключення, оскільки я хочу тримати свій комп'ютер трохи чистішим під час розробки на моєму локальному комп'ютері. Може бути зручнішим читати локальний журнал із файлу у файловій системі або виводити його безпосередньо у ваш браузер.

Symfony2 дозволяє визначити реєстратора як послугу завдяки введенню залежностей.

Це середовище, створене для самої структури. Коли Symfony2 був випущений, жоден інший фреймворк не мав такого рівня якості, як продукти з відкритим кодом Sensio Labs.

Природним ефектом цього стало те, що найбільш відомі розробники PHP з відкритим кодом були задоволені фреймворком і вибрали його.

В основному Symfony2 - це фреймворк, вибраний спільнотою, тому ви можете скористатися усіма перевагами розробників різних операційних систем, починаючи від автоматичного розгортання і закінчуючи повністю інтегрованими ORM, тестування фреймворків і навчальних посібників. (Розроблено 100 найкращих веб-сайтів за рейтингом Alexa від Symfony2).

3.2. Архітектура додатку на Symfony2

Структура каталогу

Структура каталогів програми на Symfony2 дуже гнучка, але рекомендована структура:

- app /: конфігурація програми, шаблони та переклади;
- src /: код програми програми;
- Постачальник /: Додаткові залежності;
- web /: кореневий каталог, який дивиться на світ. Директорія web**.**

Кореневий каталог Інтернету - це розташування всіх доступних статичних файлів, таких як зображення, таблиці стилів та файли JavaScript. Крім того, розташування фронтального контролера, такого як контролер виробництва, є:

```
// web/app.php
require_once __DIR__.'/../app/bootstrap.php.cache';
require_once __DIR__.'/../app/AppKernel.php';

use Symfony\Component\HttpFoundation\Request;
$kernel = new AppKernel('prod', false);
$kernel->loadClassCache();
$request = Request::createFromGlobals();
$response = $kernel->handle($request);
$response->send();
```

Контролер спочатку завантажує програму, яка використовує клас ядра (у цьому випадку AppKernel). Потім створіть об'єкт Request за допомогою глобальних змінних PHP і передайте його ядру. Останній крок - відправити відповідь ядра користувачеві.

Директорія app **:** Клас AppKernel зберігається в додатку / каталозі, оскільки він є основною точкою входу в конфігурацію вашого додатка.

Цей клас повинен реалізувати два методи:

• registerBundles () повинен повернути масив усіх наборів, необхідних для запуску програми.

• registerContainerConfiguration () завантажує налаштування програми.

Автозапуск надає Composer, тому ви можете використовувати будь-який клас PHP. Усі залежності зберігаються у каталозі постачальника / каталогу, але це лише умовно. Їх можна зберігати глобально на сервері або локально в рамках проекту, де завгодно.

Bundle системи

У цьому розділі описується одна з найбільших і найпотужніших функцій Symfony - система пакетів.

Пачки - це як плагіни для інших програм. То чому його називають пакетом замість плагіна? Це пов’язано з тим, що все в Symfony, від фреймворка ядра до коду програми, є пакетом.

Весь код, який ви пишете для своєї програми, складається в комплекті. На думку Symfony, набір - це структурований набір файлів (PHP-файли, таблиці стилів, сценарії, зображення тощо), що реалізує одну функцію (блоги, форуми тощо) і може бути не складним у використанні. Є. Ще один проект.

Пачка - це першокласний громадянин Сімфоні. Це дозволяє використовувати заздалегідь створені функції, упаковані в сторонні пакети, або розповсюджувати власні пакети. Це дозволяє легко вибрати та вибрати компоненти, які ви хочете використовувати у своєму додатку, та оптимізувати їх за потреби.

Symfony вже включає AppBundle, за допомогою якого ви можете розпочати розробку вашої програми. Потім, якщо вам потрібно розділити вашу програму на компоненти, які можна багаторазово використовувати, ви можете створити власний пакет.

Реєстрація Bundle

Ця програма складається з наборів, визначених методом registerBundles () класу AppKernel. Кожен пакет - це каталог, що містить один клас пакета , яка описує його:

```
// app/AppKernel.php
public function
registerBundles()
{
   $bundles = array(
```

```
      new Symfony\Bundle\FrameworkBundle\FrameworkBundle(),
      new Symfony\Bundle\SecurityBundle\SecurityBundle(),
      new Symfony\Bundle\TwigBundle\TwigBundle(),
      new Symfony\Bundle\MonologBundle\MonologBundle(),
      new Symfony\Bundle\SwiftmailerBundle\SwiftmailerBundle(),
      new Symfony\Bundle\DoctrineBundle\DoctrineBundle(),
      new Symfony\Bundle\AsseticBundle\AsseticBundle(),
      new Sensio\Bundle\FrameworkExtraBundle\SensioFrameworkExtraBundle(),
      new AppBundle\AppBundle();
   );
   if (in_array($this->getEnvironment(), array('dev', 'test'))) {
      $bundles[] = new Symfony\Bundle\WebProfilerBundle\WebProfilerBundle();
      $bundles[] = new Sensio\Bundle\DistributionBundle\SensioDistributionBundle();
     $bundles[] = new Sensio\Bundle\GeneratorBundle\SensioGeneratorBundle();
   }
   return $bundles;
}
```

Зауважте, що на додаток до згаданих вище AppBundles, ядро також містить інші набори, які є частиною Symfony, такі як FrameworkBundle, DoctrineBundle, SwiftmailerBundle та AsseticBundle.

Налаштування Bundle

Кожен пакет можна налаштувати за допомогою файлу конфігурації, написаного у форматі YAML, XML або PHP. Зверніть увагу на цей шаблон конфігурації за замовчуванням:

```yaml
framework:
    #esi:             ~
    #translator:      { fallback: "%locale%" }
    secret:          "%secret%"
    router:
        resource: "%kernel.root_dir%/config/routing.yml"
        strict_requirements: "%kernel.debug%"
    form:            true
    csrf_protection: true
    validation:      { enable_annotations: true }
    templating:      { engines: ['twig'] }
    default_locale:  "%locale%"
    trusted_proxies: ~
    session:         ~
# Twig Configuration
twig:
    debug:          "%kernel.debug%"
    strict_variables: "%kernel.debug%"
# Swift Mailer Configuration
swiftmailer:
    transport: "%mailer_transport%"
    host:      "%mailer_host%"
    username:  "%mailer_user%"
    password:  "%mailer_password%"
    spool:     { type: memory }
# ...
```

Кожен запис першого рівня, такий як фрікворк, гілочка та швидкий посібник, визначає конфігурацію конкретного набору. Наприклад, фреймворк налаштовує FrameworkBundle, a swiftmailer - SwiftmailerBundle.

Кожне середовище може замінити конфігурацію за замовчуванням і надати певний файл конфігурації. Наприклад, середовище DEV завантажує файл config_dev.yml. Цей файл завантажує основну конфігурацію (тобто config.yml) і модифікує її, щоб додати деякі засоби налагодження.

```yaml
# app/config/config_dev.yml
imports:
    - { resource: config.yml }
framework:
    router:   { resource: "%kernel.root_dir%/config/routing_dev.yml" }
    profiler: { only_exceptions: false }
web_profiler:
    toolbar: true
    intercept_redirects: false
# ...
```

Розширення Bundle

Набір не тільки є чудовим способом упорядкувати та налаштувати свій код, але ви також можете розширити ще один комплект. Пакети успадкування дозволяють замінити існуючий пакет для налаштування контролера, шаблону або його файлу.

Логічні імена файлів

Посилаючись на файл у наборі, використовуйте такі позначення:

@ BUNDLE_NAME / path / from / file; symfony перетворює BUNDLE_NAME у фактичний метод. Наприклад, Symfony знає місце розташування AppBundle, тому

логічний шлях @ AppBundle / controller /DefaultController.php перетворюється на SRC / AppBundle / controller /DefaultController.php.

Логічниі імена контролерів

Для контролерів, необхідно посилатися на дії, використовуючи формат BUNDLE_NAME:CONTROLLER_NAME:ACTION_NAME. Для прикладу, AppBundle:Default:index буде шляхом до методу indexAction з класу AppBundle\Controller\DefaultController.

Перевизначення Bundle

Наприклад, ви можете створити пакет (NewBundle) і вказати, що він замінює AppBundle. Коли Symfony завантажує контролер AppBundle: Default: index: спочатку шукає клас контролера за замовчуванням у NewBundle і починає пошук у AppBundle, якщо він не існує. Це означає, що один комплект може скасувати майже всі частини іншого пакета.

Використання Vendors

Найгірше те, що ваша програма покладається на сторонні бібліотеки. Їх потрібно зберігати у постачальнику / каталозі. Ніколи не торкайтесь вмісту цього каталогу. Цей каталог уже містить посилання на бібліотеки Symfony, бібліотеки Swiftmailer, ORM Doctrine, систему шаблонів Twig та інші сторонні бібліотеки.

Загальна інформація про кеші та журнали

Додатки Symfony можуть містити десятки конфігураційних файлів, визначених у декількох форматах (YAML, XML, PHP тощо). Замість аналізу та злиття всіх цих файлів при кожному запиті Symfony використовує власну систему кешування. Насправді, структура програми аналізується лише у першому запиті та компілюється у звичайний PHP-код, що зберігається у каталозі app / cache /.

У середовищі розробки Symfony досить розумний, щоб оновити кеш при зміні файлів. Однак у виробничому середовищі користувач несе відповідальність за очищення кеш-пам’яті під час оновлення коду або зміни конфігурацій для прискорення процесу. Очистіть кеш середовища prod, виконавши таку команду:

```
$ php app/console cache:clear --env=prod
```

Під час розробки веб-програми все може піти не так, як багато. Файли журналів у каталозі app / logs / розкажуть вам все про запит та допоможуть більш чітко вирішити проблему [6].

РОЗДІЛ 4
РОЗРОБКА ДОДАТКУ

Розробка додатків складається з трьох етапів

1. Моделювання баз даних.
2. Програмування контролера.
3. Використовуйте.

На кожному етапі ви отримуєте дані та інструменти, які слід використовувати для наступного завдання. Повністю завершивши попередній етап, ви будете розумно використовувати час і людські ресурси на наступному етапі.

4.1. Моделювання баз даних

Процес проектування бази даних - один із найскладніших моментів у всьому циклі розробки програмного забезпечення. По-перше, якщо ви пишете програму на чистому PHP, найзручнішим інструментом є PHPMyAdmin. Це утиліта, написана на PHP для зручного використання бази даних, і немає драйвера для підключення мови програмування до бази даних. Я повноцінний. Інструмент для налаштування БД таблиць і рядків.

Структура Symfony2 спрощує процес моделювання баз даних та розробка інструментів управління об’єктами. Він має компонент Doctrine2 (рис. 4.1.1), який містить ряд консольних команд

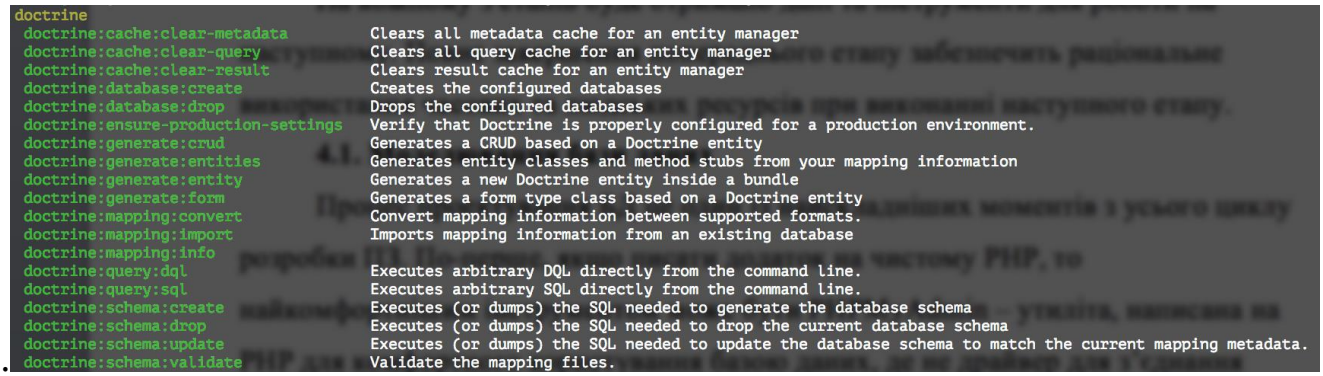


Рис. 4.1.1. Команди консольного рядка Doctrine2

- doctrine:cache:clear-result – очищення кешу результатів запитів менеджера сутностей;
- doctrine:cache:clear-query – очищення кеш запитів менеджера сутностей;
- doctrine:database:create – розробка сконфігурованною БД;
- doctrine:database:drop – видалення інснуючої БД, згідно конфігурації;
- doctrine:cache:clear-query – очищення кеш запитів менеджера сутностей;
- doctrine:ensure-production-settings – перевірка правильного налаштування Doctrine2 на виробничему середовищі;
- doctrine:generate:crud – розробка CRUD на основі сутностей Doctrine2;
- doctrine:generate:entities – розробка класів сутностей на базі конфігурацій;
- doctrine:generate:entity – розробка нової сутності у певному бандлі;
- doctrine:generate:form – розробка форми на базі певної сутності;
- doctrine:mapping:convert – конвертація конфігурацій сутностей у підтримуваних форматах;
- doctrine:mapping:import – імпорт конфігураційної інформації згідно вже існуючої БД;
- doctrine:mapping:info – відображення інформації по існуючим сутностям;
- doctrine:query:dql – виконання dql-запиту;
- doctrine:query:sql – виконання sql-запиту;

- doctrine:schema:create – виконання або дамп sql-запитів, необхідних для розробка схеми бази даних;
- doctrine:schema:drop – виконання або дамп sql-запитів, необхідних для знищення поточної БД;
- doctrine:schema:update – виконання або дамп sql-запитів, необхідних для оновлення схеми БД, згідно з оновленою конфігурацією сутностей Doctrine2;
- doctrine:schema:validate – валідація конфігураційних файлів сутностей Doctrine2.

Отже, з набору команд видно, що існує декілька варіантів для моделювання. БД. Сутності будуть створені командою doctrine:generate:entity (Рис. 4.1.2).

Для демонстрації можливостей буде створено сутність Test із полями:

- title (тип – рядок, довжина – 500 символів);
- description – (тип – текст);
- questions_quantity – (тип – smallint);
- created_at – (тип – datetime);
- active – (тип – булевий).

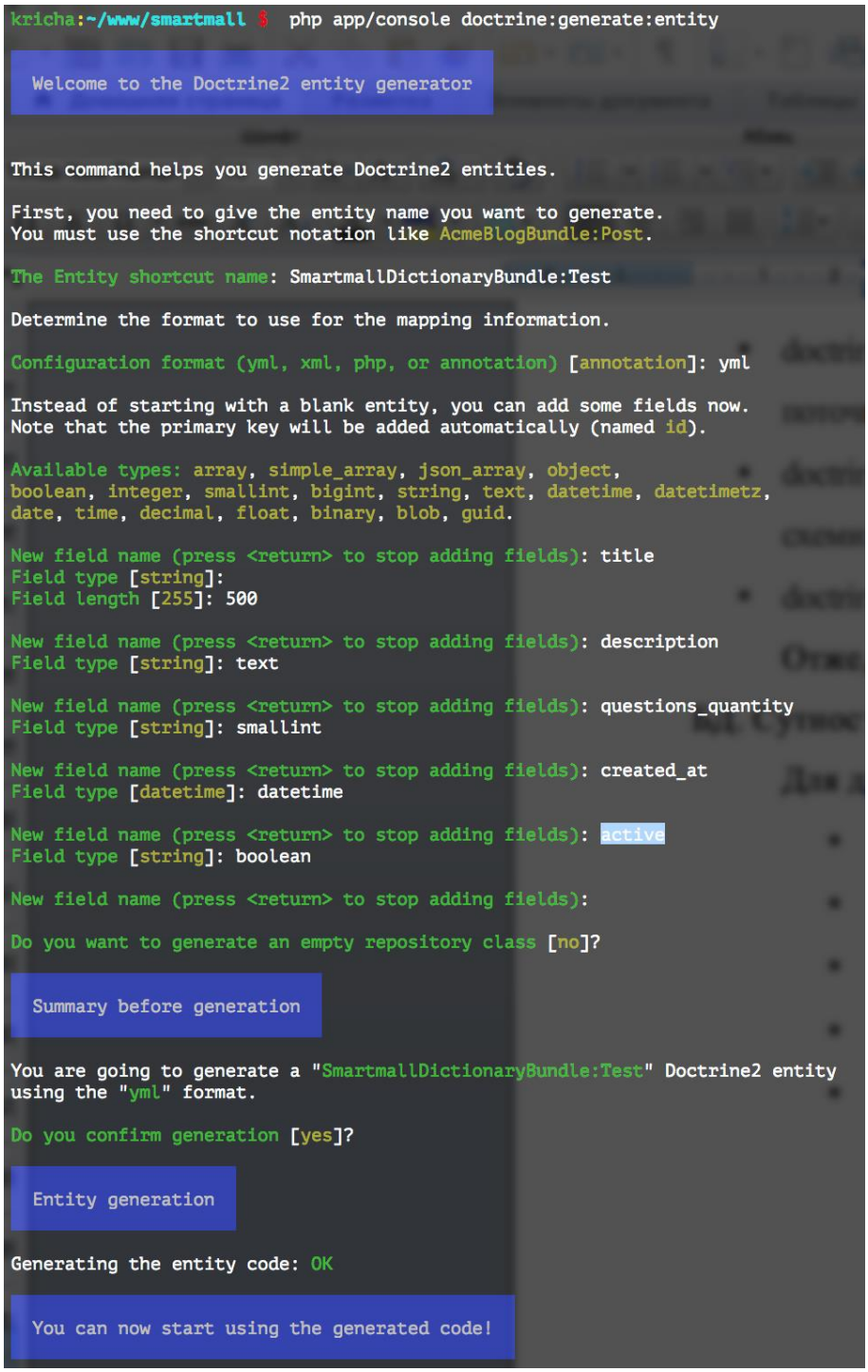


Рис. 4.1.2. результат роботи команди doctrine:generate:entity

Після отримання повідомлення “You can now start using the generated code!” буде створено сутність у форматі yml:

```yaml
Smartmall\DictionaryBundle\Entity\Test:
    type: entity
    table: null
    id:
        id:
            type: integer
            id: true
            generator:
                strategy: AUTO
    fields:


      title:
          type: string
          length: '500'
      description:
          type: text
      questionsQuantity:
          type: smallint
          column: questions_quantity
      createdAt:
          type: datetime
          column: created_at
      active:
          type: boolean
  lifecycleCallbacks: {  }
```

А для роботи з ним php-клас (рис. 4.1.3), до сутності додається необхідний ідентифікатор поля (первинний ключ), і всі методи, необхідні для комфортної роботи сутності, створюються в класі. Це показує це. Також важливо, щоб така автоматична генерація зберігала всі концепції ООП щодо інкапсуляції.

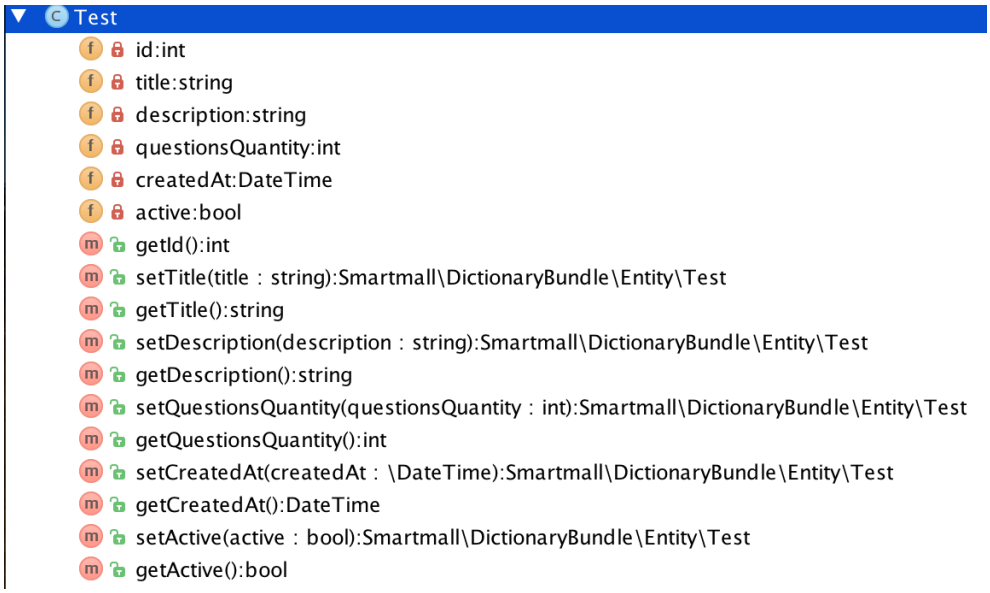


Рис. 4.1.3. Структура класу Test, створеного за допомогою команди

Єдиним недоліком є те, що ви не можете зв’язати інші існуючі об’єкти. Отже, для сутностей проекту це робиться в напівавтоматичному режимі, додаючи цей код до файлу конфігурації сутності:

```
#Smartmall\DictionaryBundle\Entity\Country:
    oneToMany:
        regions:
            targetEntity: Smartmall\DictionaryBundle\Entity\Region
            mappedBy: country

#Smartmall\DictionaryBundle\Entity\Region:
    manyToOne:
        country:
            targetEntity: Smartmall\DictionaryBundle\Entity\Country
            inversedBy: regions
```

Іншими словами, у файлі конфігурації для сутності "Країна" зазначалося, що між суттю "Регіон" та навпаки існує зв'язок "один до багатьох". Після запуску команди doctrine :gene: entity класи цих сутностей оновлюються методами, які маніпулюють взаємозв'язками між об'єктами на рівні php:

```php
<?php
namespace Smartmall\DictionaryBundle\Entity;
use Doctrine\ORM\Mapping as ORM;

class Country
{

    private $regions;

    public function __construct()
    {
        $this->regions = new
\Doctrine\Common\Collections\ArrayCollection();
    }

    public function
addRegion(\Smartmall\DictionaryBundle\Entity\Region $regions)
    {
        $this->regions[] = $regions;

        return $this;
    }

    public function
removeRegion(\Smartmall\DictionaryBundle\Entity\Region $regions)
    {
        $this->regions->removeElement($regions);
    }

    public function getRegions()
    {
        return $this->regions;
    }
}
```

Після таких змін об'єкт "Країна" зможе отримати доступ до об'єкта "Регіон" через область поля типу \ Doctrine \ Common \ Collections \ ArrayCollection після його створення, а об'єкт "Регіон" зможе доступ. Буде. Сутність "країна". Це створює структуру та сутність бази даних проекту трактату (рис. 4.1.4).

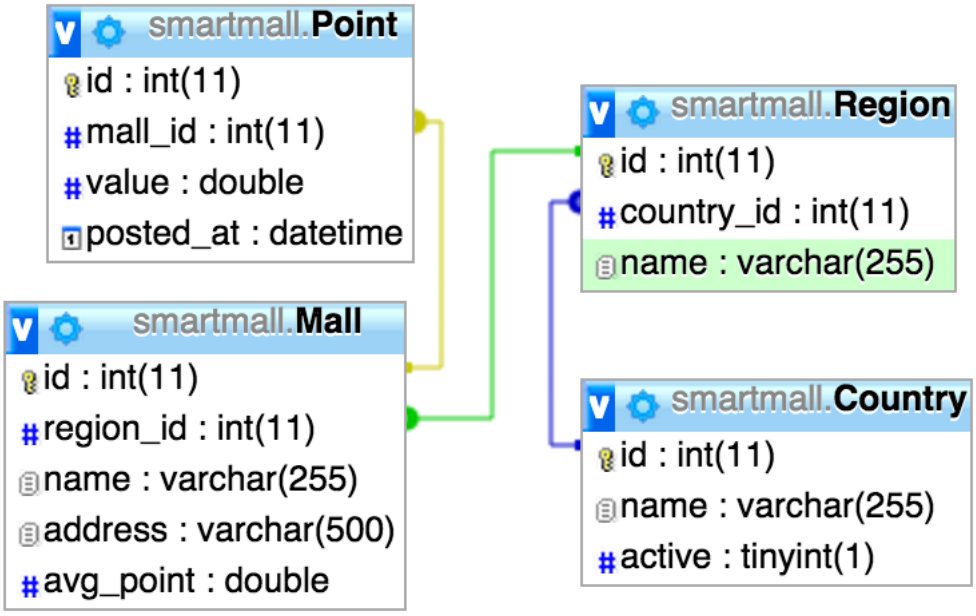


Рис. 4.1.4. структура бази даних дипломного проекта.

В кінці етапу "моделювання баз даних" є чотири таблиці:

- Країна - має ідентифікатори полів, імена та дії. Поле діяльності потрібно для тимчасової або остаточної деактивації певної країни з якихось причин;
- Регіон - має з'єднання "багато-до-одного" з ідентифікатором поля, назвою та країною таблиці;
- Торговий центр - існує взаємозв'язок "багато до одного" з ідентифікаторами полів, іменами, адресами, середніми балами та регіонами таблиці;
- Рейтингові зв’язки з ідентифікаторами полів, значеннями, датами та таблицею.

4.2. Кодування контроллерів

Контролери можна розробити кількома ручними та напівавтоматичними способами. Другий варіант включає використання інструментів командного рядка, а потім оновлення до ручного.

Щоб сформувати контролер, використовуйте доктрину командної консолі: generate: crud (рис. 4.2.1).

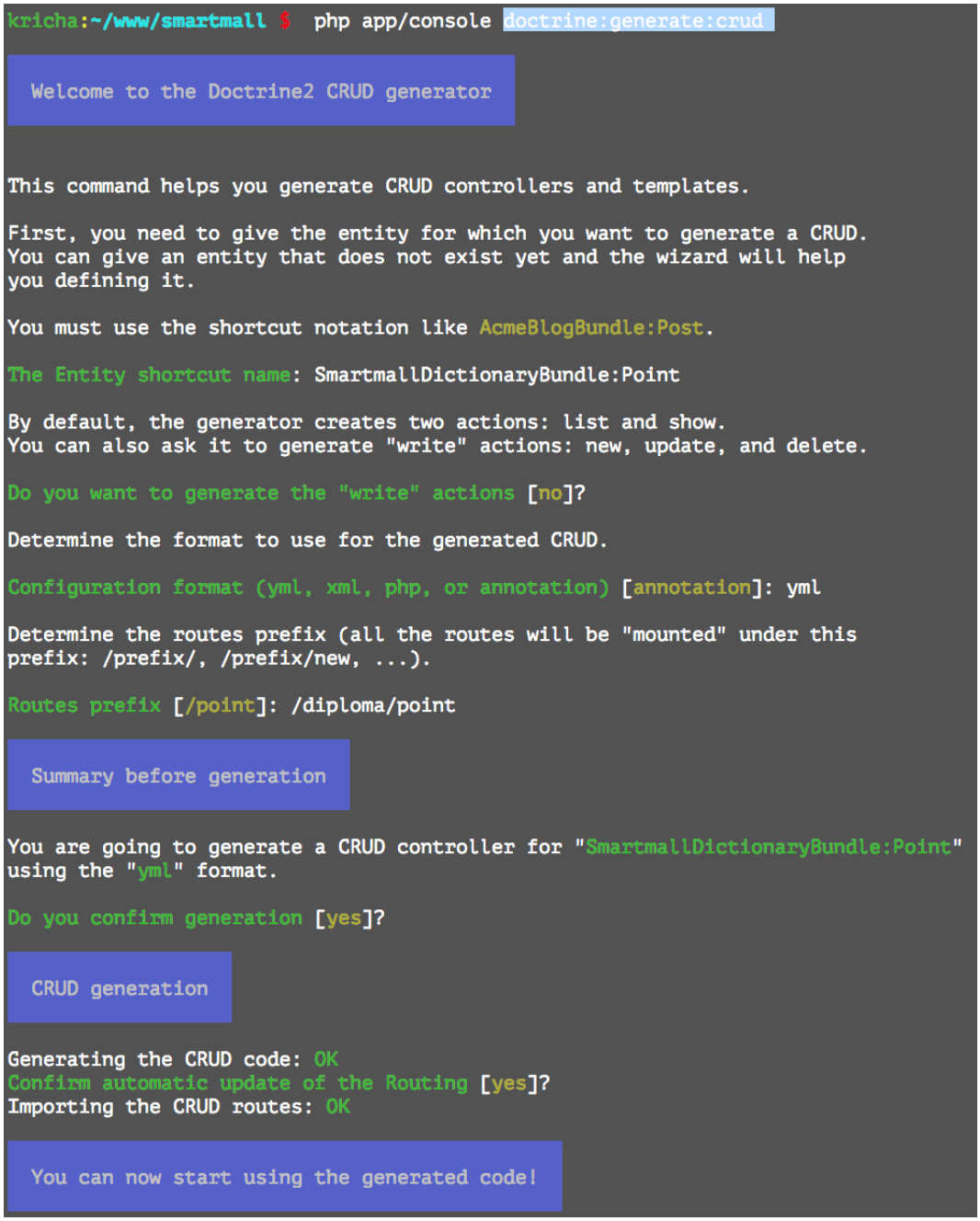


Рис. 4.2.1. виконання консольної команди doctrine:generate:crud

Під час запуску команди потрібно вказати деякі параметри:

- Суть, для якої створено контролер;

- Тип контролера (базовий або вдосконалений. Додаткові контролери включають методи та маршрути для методів запису, але базові контролери можуть відображати список усіх елементів сутності та одного конкретного елемента за ідентифікатором. Masu);
- Префікс маршрутизації;
- Формат файлу конфігурації, в якому генерується контролер;
- Чи потрібно оновлювати центральний маршрутизатор (підключити локальну маршрутизацію для цього контролера).

Результатом буде два файли. Конфігурація маршрутизатора:

```
diploma_point:
    path:     /
    defaults: { _controller: "SmartmallDictionaryBundle:Point:index" }

diploma_point_show:
    path:     /{id}/show
    defaults: { _controller: "SmartmallDictionaryBundle:Point:show" }
```

Та php-клас контроллера: Оскільки маршрутизатор базується на контролері, існує лише два правила diploma_point, які обробляються під час міграції посилання / диплом / точка (рис. 4.2.2), і викликається метод.

Викликаються SmartmallDictionaryBundle: Point: index та правило diploma_point_show, яке обробляється при переході до посилання / diploma / point / 1 / show (рис. 4.2.3), та метод SmartmallDictionaryBundle: Point: show.

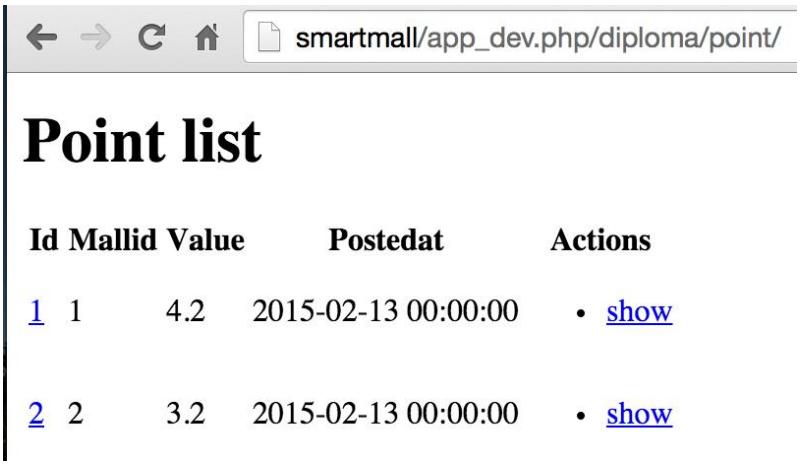


Рис. 4.2.2. результат переходу по посиланню /diploma/point

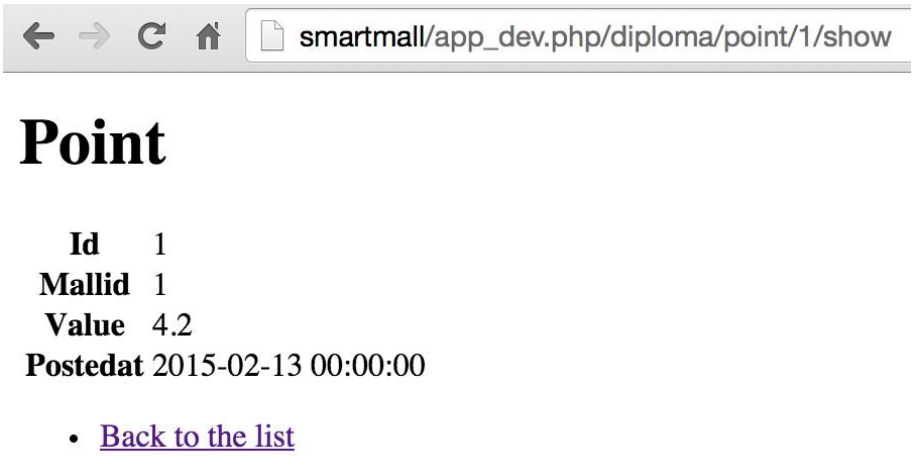
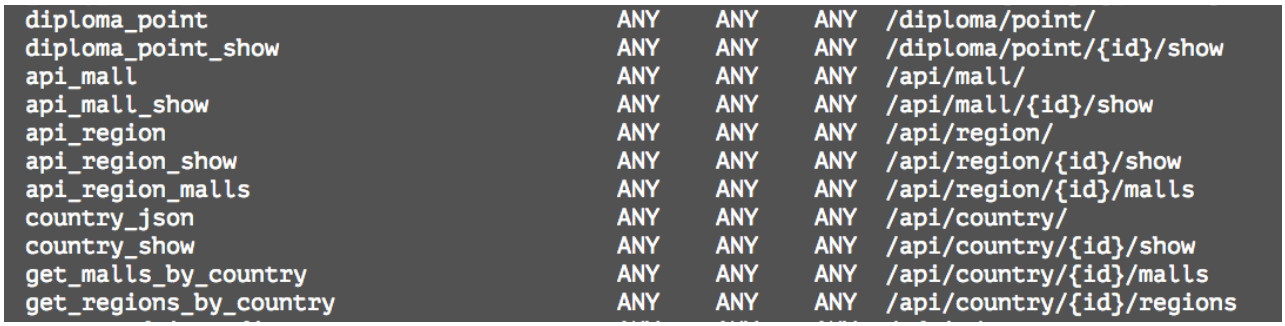


Рис. 4.2.3. результат переходу по посиланню /diploma/point

Під час розробки додатків великі проекти також мають велику кількість маршрутів, тому ви завжди можете нічого не пам'ятати і завжди швидко пам'ятати правильний маршрут і знаходити помилки при використанні там посилань. Є консольним маршрутизатором команд. налагодження (рис. 4.2.4) показує існуючі маршрути, їх назви, правила відповідності та інші параметри.



diploma_point	ANY	ANY	ANY	/diploma/point/
diploma_point_show	ANY	ANY	ANY	/diploma/point/{id}/show
api_mall	ANY	ANY	ANY	/api/mall/
api_mall_show	ANY	ANY	ANY	/api/mall/{id}/show
api_region	ANY	ANY	ANY	/api/region/
api_region_show	ANY	ANY	ANY	/api/region/{id}/show
api_region_malls	ANY	ANY	ANY	/api/region/{id}/malls
country_json	ANY	ANY	ANY	/api/country/
country_show	ANY	ANY	ANY	/api/country/{id}/show
get_malls_by_country	ANY	ANY	ANY	/api/country/{id}/malls
get_regions_by_country	ANY	ANY	ANY	/api/country/{id}/regions

Рис. 4.2.4. відображення роботи консольної команди router:debug

Оскільки додаток орієнтований на роботу в інших системах, в даному випадку на мобільних пристроях, необхідно надавати дані в конкретному умовному форматі. Формат json був обраний через його простоту, чіткий синтаксис та невелику вагу переданих даних порівняно, наприклад, з xml.

Щоб отримати такі дані, потрібно переписати основний контролер. Прикладом є один із методів контролера, який працює з суттю Country.

Після генерації базового контролера метод, який надає повний список, виглядає так:

```
public function indexAction()
{
    $em = $this->getDoctrine()->getManager();

    $entities = $em-
>getRepository('SmartmallDictionaryBundle:Country')->findAll();

    return $this-
>render('SmartmallDictionaryBundle:Country:index.html.twig', array(
        'entities' => $entities,
    ));
}
```

У методі отримувався доступність до папки сутності "Країна", забирав із бази усі записи і відображав у стандартному вигляді всі записи. Після модернізації метода, код змінився на такий:

```
public function index_jsonAction()
{
    $em = $this->getDoctrine();

    $entities = $em-
>getRepository('SmartmallDictionaryBundle:Country')-
>getAllInArray();

    return new JsonResponse($entities);
}
```

У цьому випадку ви також отримаєте доступ до сховища Country, але записи в базі даних будуть обрані по-іншому. Новий метод getAllInArray для вибору всіх країн з таблиці є нестандартним, програмованим розширенням сховища.

```
public function getAllInArray() {
    $qb = $this->getEntityManager()->createQueryBuilder();
    $countries = $qb->select('country')-
>from('SmartmallDictionaryBundle:Country', 'country')-
>where('country.isActive = 1')->getQuery()->getArrayResult();
    return $countries;
}
```

Цей метод використовує для побудови запиту до бази даних відповідно до певних правил. У цьому випадку вибираються лише країни зі значенням 1 у полі is_актив. Також результатом запиту буде набір сутностей Doctrine2, а не набір сутностей Doctrine2.

Після отримання даних з бази даних створюється об'єкт відповіді JsonResponse, масив, отриманий його конструктором, переноситься, перетворюється у відповідний формат і видається на вимогу.

Дані тестів часто потрібні при розробці програми. Щоб їх отримати, напишіть конкретний скрипт для генерації великого обсягу даних або скористайтеся власними інструментами. В даний час проекти не розглядаються з точки зору великих навантажень, тому їх можна запускати з невеликим обсягом даних. Написати сценарій для генерації було нерозумно, і всі інструменти, які я мав, виявились незручними, тому я вирішив підключити

компонент SonataAdminBundle. Це дозволяє швидко запустити адміністративну панель для управління об’єктами Doctrine2.

Сонати пишуться відповідно до встановлених правил Symfony2, тому для створення контролера адміністратора потрібно використовувати консольну команду sonata: admin: generate (Рисунок 4.2.5). Тут потрібно вказати суть, що використовується контролером управління. Сформовані та інші необхідні параметри.

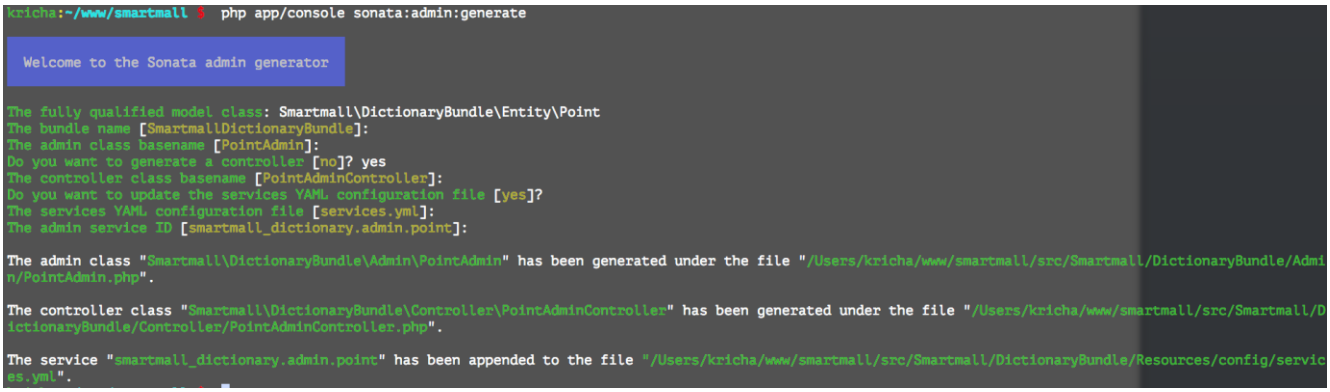


Рис. 4.2.5. виконання команди sonata:admin:generate

Результатом є два файли /Smartmall/DictionaryBundle/Admin/PointAdmin.php (основний конфігуратор файлів для полів, що використовуються в адміністративній панелі, що повинно бути показано, що слід приховати, як це показати) l де витягти дані):

```php
<?php

namespace Smartmall\DictionaryBundle\Admin;

use Sonata\AdminBundle\Admin\Admin;
use Sonata\AdminBundle\Datagrid\DatagridMapper;
use Sonata\AdminBundle\Datagrid\ListMapper;
use Sonata\AdminBundle\Form\FormMapper;
use Sonata\AdminBundle\Show\ShowMapper;

class PointAdmin extends Admin
{
    protected function configureDatagridFilters(DatagridMapper
$datagridMapper)
    {
        $datagridMapper
            ->add('id')
            ->add('mallId')
            ->add('value')
            ->add('postedAt')
        ;
    }
    protected function configureListFields(ListMapper $listMapper)
    {
        $listMapper
            ->add('id')
            ->add('mallId')
            ->add('value')
            ->add('postedAt')
            ->add('_action', 'actions', array(
                'actions' => array(
                    'show' => array(),
                    'edit' => array(),
                    'delete' => array(),
                )
            ))
        ;
    }
    protected function configureFormFields(FormMapper $formMapper)
    {
        $formMapper
            ->add('id')
            ->add('mallId')
            ->add('value')
            ->add('postedAt')
        ;
    }
    protected function configureShowFields(ShowMapper $showMapper)
    {
        $showMapper
            ->add('id')
            ->add('mallId')
            ->add('value')
            ->add('postedAt')
```

l/Smartmall/DictionaryBundle/Controller/PointAdminController.php контролює сутності, до яких вони належать. Після підключення контролер управління може переглядати, створювати, редагувати та видаляти записи з панелі управління (рис. 4.2.6-4.2.8).

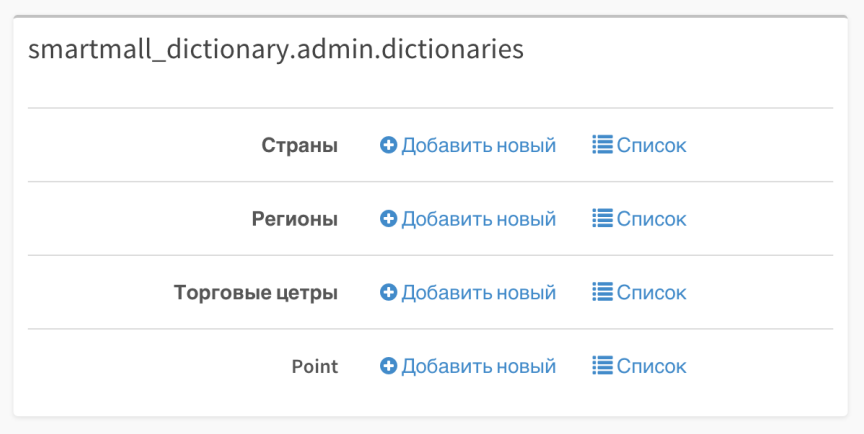


Рис. 4.2.6. список підключених адміністративних контроллерів у панелі

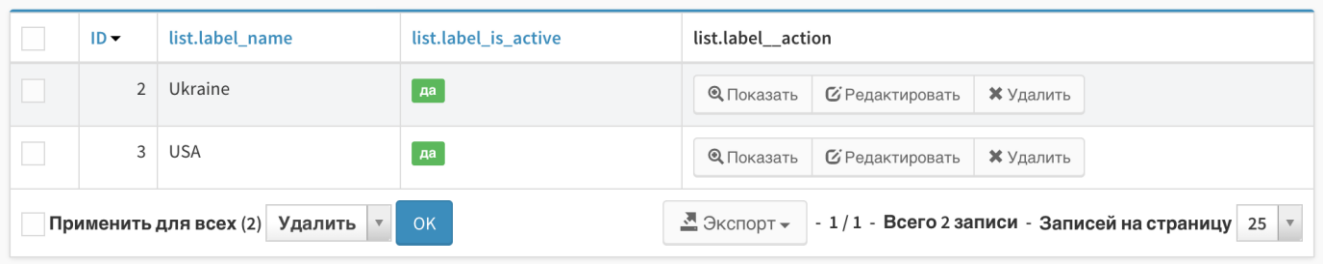


Рис. 4.2.7. список елементів сутності Країна у адміністративній панелі

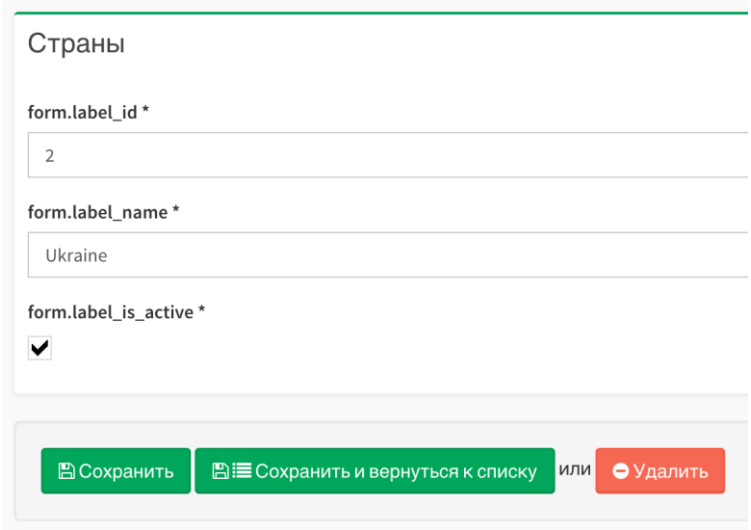


Рис. 4.2.8. редагування сутності Країна у адміністративній панелі

4.3. Використання api

Після завершення етапу попередньої розробки результатом став повнофункціональний API для отримання даних із веб-сервера за допомогою мобільного додатка. Робочий маршрут для отримання даних у форматі json такий:

- api_mall – отрмання списку молів (рис. 4.3.1);


```
[
  - {
        id: 1,
        name: "SkyMall",
        address: "Троещина, проспект генерала Ватутина",
        averagePoint: 4.4
    },
  - {
        id: 2,
        name: "Гуливер",
        address: "ст. м. Дворец Спорта, Эспланадная 33",
        averagePoint: 5
    }
]
```


Рис. 4.3.1. список молів у форматі json

- api_mall_show – детальна інформація по певному молу (рис. 4.3.2);


```
[
  - {
        id: 1,
        name: "SkyMall",
        address: "Троещина, проспект генерала Ватутина",
        averagePoint: 4.4,
      - region: {
            id: 1,
            name: "Kiev",
          - country: {
                id: 2,
                name: "Ukraine",
                isActive: true
            }
        }
    }
]
```


Рис. 4.3.2 – детальна інформація по певному молу у форматі json

- api_region - отрмання списку регіонів (рис. 4.3.3);


```
[
  - {
        id: 1,
        name: "Kiev"
    },
  - {
        id: 2,
        name: "Lviv"
    },
  - {
        id: 3,
        name: "California"
    },
  - {
        id: 4,
        name: "San-Francisco"
    },
  - {
        id: 5,
        name: "Харьков"
    },
  - {
        id: 6,
        name: "Днепропетровск"
    },
  - {
        id: 7,
        name: "Черновцы"
    },
  - {
        id: 8,
        name: "Чернигов"
    },
  - {
        id: 9,
        name: "Ужгород"
    }
]
```


Рис. 4.3.3. список регіонів у форматі json

- api_region_show - детальна інформація по певному регіону (рис. 4.3.4);


```
[
  - {
        id: 1,
        name: "Kiev"
    }
]
```


Рис. 4.3.4. детальна інформація по певному регіону у форматі json

- api_region_malls – список молів по регіону (рис. 4.3.5);

```
[
  - {
        id: 1,
        name: "SkyMall",
        address: "Троещина, проспект генерала Ватутина",
        averagePoint: 4.4
    },
  - {
        id: 2,
        name: "Гуливер",
        address: "ст. м. Дворец Спорта, Эспланадная 33",
        averagePoint: 5
    }
]
```


Рис. 4.3.5. список молів по регіону у форматі json

- country_json - список країн (рис. 4.3.6);


```
[
  - {
        id: 2,
        name: "Ukraine",
        isActive: true
    },
  - {
        id: 3,
        name: "USA",
        isActive: true
    }
]
```


Рис. 4.3.6. список країн у форматі json

- country_show – детальна інформація по певній країні (рис. 4.3.7);


```
[
  - {
        id: 2,
        name: "Ukraine",
        isActive: true
    }
]
```


Рис. 4.3.7. детальна інформація по певній країні у форматі json

- get_malls_by_country – список молів по країні (рис. 4.3.8);

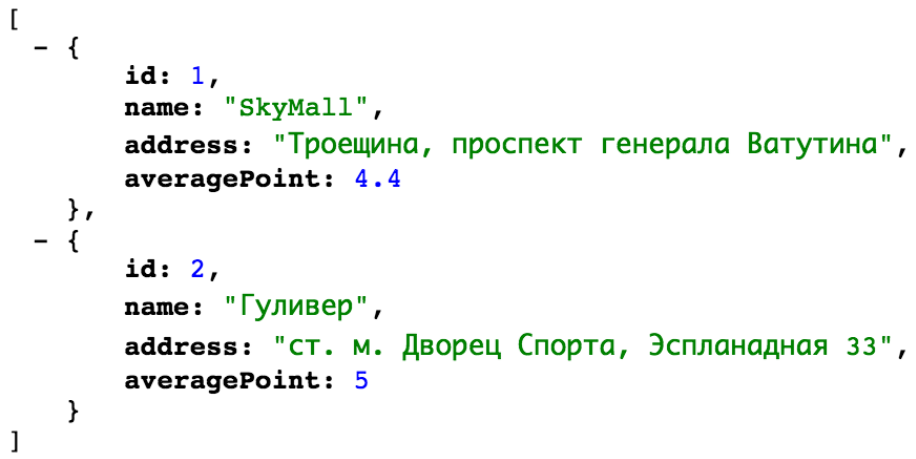


Рис. 4.3.8. список молів по країні у форматі json

- get_regions_by_country – список регіонів по країні (рис. 4.3.9);

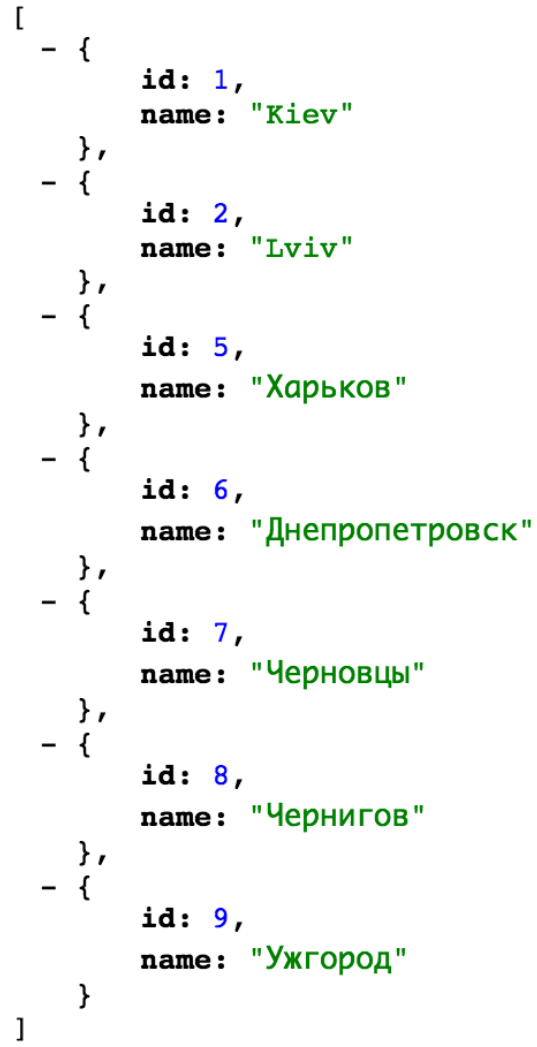


Рис. 4.3.9. список регіонів по країні у форматі json

ВИСНОВОК

В результаті трактату вивчено всі аспекти розробки веб-додатків мовою програмування PHP. Був проведений ретельний аналіз методів розробки програмного забезпечення для пошуку оптимальної структури програми. Було встановлено, що правильне використання php-framework symfony2 дасть хороші результати для роботи остаточного програмного продукту та допоможе вам розумно використовувати свій час та людські ресурси. symfony2 добре працював на веб-сервері nginx. Крім того, під час мого дослідження я виявив, що фреймворк symfony2 отримує значну кількість підтримки від сторонніх розробників, якщо тільки не врахована величезна підтримка автора.

Після детального аналізу та вибору інструменту розпочався та успішно завершився триетапний процес створення веб-програми. Він виступив мостом між сервером та мобільним додатком на iOS. Написаний API - це міжплатформене рішення, яке можна використовувати для розробки подібних мобільних додатків, але, наприклад, для платформи Android ви не витрачаєте час на адаптацію до нового середовища. На завершальному етапі своєї роботи використовуйте повноцінний REST API у форматі JSON. Це дозволило обмінюватися даними між мобільними додатками користувача за протоколом HTTP та отримувати веб-програму сервера.

СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Мова программування PHP [Електронний ресурс]. - Режим доступу: https://uk.wikipedia.org/wiki/PHP (дата звернення 19.05.21) - Назва з екрану.
2. Human-centred thinking Tech-centred doing [Електронний ресурс]. - Режим доступу: http://www.theorganicagency.com/ (дата звернення 20.05.21) - Назва з екрану.
3. WEB ресурси та фреймворки [Електронний ресурс]. - Режим доступу: http://codegeekz.com/ (дата звернення 21.05.21) - Назва з екрану.
4. Фреймворк Symfony PHP для веб-проектів [Електронний ресурс]. - Режим доступу: http://symfony.com/ (дата звернення 22.05.21) - Назва з екрану.
5. Pop PHP framework [Електронний ресурс]. - Режим доступу: https://www.popphp.org/ (дата звернення 23.05.21) - Назва з екрану.
6. Laravel: фреймворк PHP для веб-майстрів [Електронний ресурс]. - Режим доступу: https://laravel.com/ (дата звернення 23.05.21) - Назва з екрану.
7. Medoo framework library [Електронний ресурс]. - Режим доступу: https://medoo.in/ (дата звернення 23.05.21) - Назва з екрану.