ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

ПОЯСНЮВАЛЬНА ЗАПИСКА

до бакалаврської роботи

на ступінь вищої освіти бакалавр

на тему: «**РОЗРОБКА ФРЕЙМВОРКУ ДЛЯ WEB-ДОДАТКІВ МОВОЮ PHP**»

Виконав: студент 4 курсу, групи ПД-42

спеціальності

121 Інженерія програмного забезпечення

Ліщук І.В.

Керівник Дібрівний О.А

Рецензент

Нормоконтроль

Київ - 2021

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти – «Бакалавр»

Спеціальність – 121 Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
Негоденко О.В.
"____" _____ 2021 року

ЗАВДАННЯ
НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Ліщук Ігор Валерійович

1. Тема роботи: «Розробка фреймворку для WEB-додатків мовою PHP»
Керівник роботи Дібрівний Олесь Андрійович, доктор філософії
затверджені наказом вищого навчального закладу від «12» березня 2021 року № 65.

2. Строк подання студентом роботи 01.06.2021

3. Вихідні дані до роботи:
Методи створення WEB-фреймворків;
Порівняльні методи дослідження WEB-фреймворків;
Офіційна документація мови програмування PHP.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно

розробити)
4.1 Аналіз об’єкту програмування;
4.2 Опис програмних засобів;
4.3 Опис загальних відомостей про WEB-фреймворк;
4.4 Реалізація фреймворку;

5. Перелік графічного матеріалу
1. Титульний слайд;
2. Мета, об’єкт дослідження, предмет дослідження;
3. Аналіз існуючих аналогів;
4. Технічне завдання;
5. Засоби програмної реалізації;
6. Розроблений WEB-фреймворк;
7. UML діаграма послідовностей для архітектури MVC

6. Дата видачі завдання 19.04.2021

КАЛЕНДАРНИЙ ПЛАН

№ з /п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Вибір теми бакалаврської роботи		
2	Дослідження актуальності теми		
3	Розгляд аналогічних WEB-фреймворків		
4	Дослідження програмних засобів		
5	Розробка функціоналу WEB фреймворку		
6	Розробка додатку з використанням WEB-фреймворку		
7	Тестування стабільності роботи додатку		
8	Проходження нормоконтролю, отримання рецензії, подача роботи в ДЕК		
9	Захист дипломної роботи		

Студент _____ Ліщук І.В.

Керівник роботи _____ Дібрівний О.А.

РЕФЕРАТ

Текстова частина бакалаврської роботи с. 51, рис. 24, джерел 15.

Об`єкт дослідження – клієнт-серверні додатки та WEB сайти.

Предмет дослідження – WEB-фреймворк з використанням мови PHP.

Мета дослідження – прискорення часу розробки клієнт-серверних додатків, мінімізація витрат на готовий продукт, зменшення порогу входу для нових розробників, розбивка проекту на окремі модулі.

Методи дослідження – порівняльні методи дослідження фреймворків, методи створення WEB-фреймворків.

У дипломній роботі були проаналізовано актуальні схожі WEB-фреймворки, методи створення надійної архітектури WEB-фреймворку та розроблено тестовий додаток, котрий використовує розроблений WEB-фреймворк.

Розроблений WEB-фреймворк дозволяє створювати клієнт-серверні додатки різного направлення та надає можливості масштабування їх використовуючи його інструменти.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

WEB	World Wide WEB
MVC	Model View Controller
ORM	Object Role Model
REST	Representational State Transfer
API	Application Programming Interface
HTTP	Hypertext Transfer Protocol
PHP	Personal Home Page Tools
AR	Active Record
VPS	Virtual Private Server
URL	Uniform Resource Locator

ВСТУП

Так як WEB-технології стрімко розвиваються, у WEB-програмістів часто бракує часу для реалізації повсякденних завдань з нуля. В основному такими завданнями являються: дебаг, кешування, тестування, API, робота з БД, аутентифікація і т. д. Тому, знаючи цю проблему, були придумані WEB-фреймворки, котрі в свою чергу спрощують і прискорюють виконання таких завдань. WEB-фреймворк являється каркасом додатка. У ньому присутній набір інструментів, а також набір стандартів, дотримуючись яких можна створити якісний продукт.

Об`єкт сфери дослідження – клієнт-серверні додатки та WEB сайти.

Предмет дослідження – WEB-фреймворк з використанням мови PHP.

Мета дослідження – прискорення часу розробки клієнт-серверних додатків, мінімізація витрат на готовий продукт, зменшення порогу входу для нових розробників, розбивка проекту на окремі модулі.

Методи дослідження – порівняльні методи дослідження фреймворків, методи створення WEB-фреймворків.

Новизна проекту – удосконалення існуючих WEB-фреймворків, низький поріг входу для нових розробників, простота масштабованості проектів.

У дипломній роботі були проаналізовано актуальні схожі WEB-фреймворки, методи створення надійної архітектури WEB-фреймворку та розроблено тестовий додаток, котрий використовує розроблений WEB-фреймворк.

РОЗДІЛ 1. ТЕОРЕТИЧНА ЧАСТИНА

1.1. WEB-фреймворк

1.1.1. Визначення

WEB-фреймворк або фреймворк WEB-додатків – це програмний інструмент, призначений для створення та запуску WEB-програм, таких як WEB-служби, WEB-сервери, WEB-джерела, WEB-портали та WEB-API. В результаті програмістам не доводиться самостійно писати код і витрачати свій час та сили на пошук можливих помилок обчислення та багів. Насправді, різні провідні WEB-фреймворки та моделі пропонують бібліотеки для шаблонування, управління сеансами, статичного файлового сервера, інструменти тестування та доступу до баз даних. Саме з цих причин WEB-фреймворк часто схвалює перевантаження коду та повторне його використання.

Незважаючи на те, що WEB-програми є звичними явищами в Інтернеті і використовуються щодня більшістю людей, підключених до Інтернету, вони все ще перебувають у зародковому стані, коли йдеться про розробку тактики та інструментів. Ось чому так важливо мати WEB-фреймворк, який би виступав системою підтримки для розробки та розгортання таких додатків. Фреймворк WEB-додатків є одним із таких інструментів.

1.1.2. Користь

WEB-фреймворк – це бібліотека коду, яка робить WEB-розробку швидшою та простішою, надаючи основні шаблони для створення надійних, масштабованих та ремонтопридатних WEB-додатків. WEB-фреймворки існують, щоб спростити розробнику створення WEB-програми. Щоб запобігти необхідності переписувати цей код з нуля кожного разу, коли ви створюєте WEB-сайт або WEB-службу, WEB-фреймворк робить це за вас.

WEB-фреймворк відрізняється від WEB-сервера тим, що WEB-сервер насправді запускає WEB-додаток, тоді як WEB-фреймворк більше нагадує

віртуальну базу даних або бібліотеку, яка допомагає пришвидшити процес розробки та написання.

1.1.3. Загальні функції

Фреймворки надають функціональність у своєму коді або через розширення для завершення повсякденних операцій, необхідних для запуску WEB-додатків. Ці операції включають:

- маршрутизація URL-адрес;
- управління та перевірка форми вводу;
- налаштування HTML, XML, JSON та інших продуктів із механізмом шаблонування;
- конфігурація з'єднання з базою даних та рішуча маніпуляція даними за допомогою об'єктно-реляційного картографування (ORM);
- WEB-захист від підробки міжсайтових запитів (CSRF), SQL Injection, міжсайтових сценаріїв (XSS) та інших частих зловмисних атак;
- сховище сесії та пошук;

1.1.4. Аналіз аналогів

Laravel

Коли люди говорять про WEB-фреймворки, одне з імен, яке найчастіше з’являється – Laravel. Цей конкретний фреймворк славиться своїм елегантним синтаксисом, з яким легко зрозуміти і приємно працювати.

За допомогою Laravel можна швидко працювати над своїми проектами. Можна забути про виконання деяких рутинних задач, оскільки Laravel надає доступ до таких функцій, як автентифікація користувачів, управління сеансами та кешування. Загалом, Laravel має всі функції, необхідні для створення сучасного WEB-додатку, що говорить багато про що.

Ядро Laravel надійне з точки зору продуктивності, і ви можете розширити фреймворк, використовуючи безліч доповнень.

Laravel також чудово інтегрується з іншими сторонніми бібліотеками та платформами, такими як Amazon WEB Services (AWS), що дозволяє створювати високо масштабовані програми. Для довготривалих завдань ви можете поставити їх у чергу, щоб вони працювали асинхронно у фоновому режимі, що допомагає ще більше підвищити продуктивність.

Нарешті, Laravel може похвалитися високоактивною спільнотою, а це означає, що пошук допомоги чи навчальних посібників ніколи не буде проблемою. Якщо ви вперше використовуєте фреймворк, це робить Laravel ще кращим варіантом.

Переваги:

- управління чергою в Laravel є одним з найкращих серед усіх інших розробок PHP;
- фреймворк полегшує обробку кількох фонових завдань. Усі фонові дії реєструватимуться під час нормальної роботи на frontend частині проекту;
- вбудований composer Laravel полегшує додавання пакетів, а оскільки він також інтегрується з Elixir та Gulp, ssh може використовуватися для безпосереднього виклику будь-якого браузера та пакетів npn;
- Laravel спеціально обробляє структури NoSQL (наприклад, Redis та MongoDB) як шматок пирога;
- підручники Laravel Udemy, величезна підтримка спільноти та велика документація полегшують новим розробникам початок роботи з цією структурою;

Недоліки:

- велика частина функціоналу працює через фасади, тому системи IDE не бачать де знаходяться деякі методи та властивості у деяких класах, що призводить до помилок;
- вивчається трохи складніше Yii2;

- немає вбудованих генераторів інтерфейсів;

Поріг вступу: Досить знати ООП та бази даних.

Phalcon

Phalcon був побудований для оптимізації продуктивності. Продуктивність забезпечується за рахунок того, що його архітектура MVC побудована як розширення C. Через це Phalcon використовує дуже мало ресурсів, а отже HTTP-запити обробляються неймовірно швидко. Якщо вам потрібен WEB-сайт, який працює неймовірно швидко, то Phalcon може творити такі чудеса.

Цей фреймворк має величезне багатство компонентів, таких як ORM, автоматичне завантаження, кешування, зберігання даних PHQL, движки шаблонів, простота створення додатків, конструктори форм, картографування об'єктів MongoDB, а також відмінна документація.

Переваги:

- висока продуктивність;
- багатий функціонал;
- добре для високого навантаження;
- потрібно мінімум файлових операцій;
- споживає мало ресурсів;
- ви можете використовувати власну основу та окремі її елементи;
- технологія ORM взаємодіє з базами даних, що призводить до дуже високої продуктивності;
- всі процеси відбуваються досить швидко, завдяки прямому доступу фреймворку до внутрішніх структур PHP;
- ви можете використовувати власну основу та окремі її елементи;

Недоліки:

- вихідний код на мові C;
- ще не набув достатньої популярності;
- являється розширенням (не можна запускати на спільному хостингу);

- багато розробників досі не знають про існування Phalcon;

Поріг вступу: середній, потрібно знати ООП, розуміти закономірності проектування та мати практичний досвід роботи з проектами. Хоч його документація не на рівні з такими гігантами, як Laravel. Пошук допомоги щодо помилок також може зайняти більше часу. Однак Phalcon наближається до того, що є найкращим фреймворком PHP, який ви можете використовувати, особливо якщо продуктивність це те, що ви шукаєте.

Yii

Yii тепер переписано як Yii 2. Yii2 був стратегічно розроблений для оптимізації продуктивності WEB-сайту за допомогою своєї лінивої техніки завантаження. Завдяки цій техніці ініціалізація об’єкта буде відкладена доти, поки це не буде абсолютно необхідним. Оскільки Yii2 є виключно об'єктно-орієнтованою, він має логічну та чисту кодову базу. Крім свого динамічного генератора коду класу, Yii2 також інтегрований jQuery. Це означає, що ви не тільки насолоджуєтесь безліччю функцій AJAX, але також інтуїтивно зрозумілим механізмом тематизації та обробки.

Переваги:

- генератор візуального коду;
- інтуїтивна архітектура MVC;
- використовує стандартні методи вирішення проблем, які зменшують або усувають заплутаність коду;
- полегшує підтримку коду з використанням загальної архітектури та методів;
- він має активну спільноту розробників;
- невимогливий до ресурсів;
- різні варіанти кешування;
- проста інтеграція сторонніх бібліотек та класів;
- якісні засоби безпеки;

- підходить для проектів будь-якої складності та масштабу;

Недоліки:

- не дуже гнучке формування маршруту;
- поріг входу: низький, достатньо знати ООП та бази даних;

Поріг входу: низький, достатньо знати ООП та бази даних.

Symfony

Symfony – це фреймворк, який стоїть за такими іменами, як BBC, та великими проектами, такими як eZpublish та Drupal. Власне кажучи, кілька інших фреймворків, таких як Laravel, покладаються на Symfony. Це тому, що він був побудований з урахуванням стабільності. Він також забезпечує високу продуктивність і добре задокументований. Symfony додатково підтримується та підтримується, що робить його фантастичним фреймворком.

Елементи у Symfony мають форму багаторазових бібліотек PHP, званих Компонентами, які ідеально підходять для різних завдань, включаючи маршрутизацію, конфігурацію об'єкта, створення форми, шаблонування та автентифікацію.

Переваги:

- вбудована підтримка Codeception дозволяє писати функціональні та приймальні тести;
- YAML як фреймворкова перевага;
- велика спільнота розробників;
- безліч готових модулів (пучків);
- детальна та чітка документація;
- досить висока швидкість ядра;

Недоліки:

- Symfony використовує дуже ресурсномісткий ORM;
- перенасичення різного роду утворень;
- вбудований синтаксис анотацій

Поріг вступу: високий, ви повинні знати ООП, розуміти закономірності проектування та мати практичний досвід роботи з проектами в інших рамках.

Zend

За цією структурою є якась серйозна сила. У неї є відомі партнери, включаючи Microsoft, Adobe, IBM та Google. Zend дуже конфігурується і надійний, що робить його ідеальним для великих складних розробок PHP. Його безпека, розширюваність та продуктивність роблять його популярним для створення додатків на рівні підприємства.

Zend, як відомо, керується підприємством. Він має такі компоненти, як аутентифікація, форми, канали та інші послуги, що сприяють розвитку на рівні підприємства. Отже, це не є ідеальною базою для роботи над швидкою розробкою додатків.

Переваги:

- повністю об'єктно-орієнтований;
- ви можете інтегрувати все, що побажаєте;
- він має готове рішення для різноманітних завдань;
- локалізація додатків;
- підтримка спільноти розробників;
- якісна документація;

Недоліки:

- не підходить для швидкого розвитку;
- повільніше, ніж деякі інші рамки;
- забирає багато часу на навчання;
- ресурсоємність;

Поріг вступу: високий, ви повинні знати ООП, розуміти закономірності проектування та мати практичний досвід роботи з проектами на інших фреймворках.

1.2. Model View Controller

Model-View-Controller рис 1.1. (MVC) являється архітектурним шаблоном, який відокремлює додаток на три основні логічні компоненти: модель , вид і контролер. Кожен із цих компонентів створений для обробки конкретних аспектів розробки програми. MVC – одна з найбільш часто використовуваних галузевих стандартів WEB-розробки для створення масштабованих та розширюваних проектів.

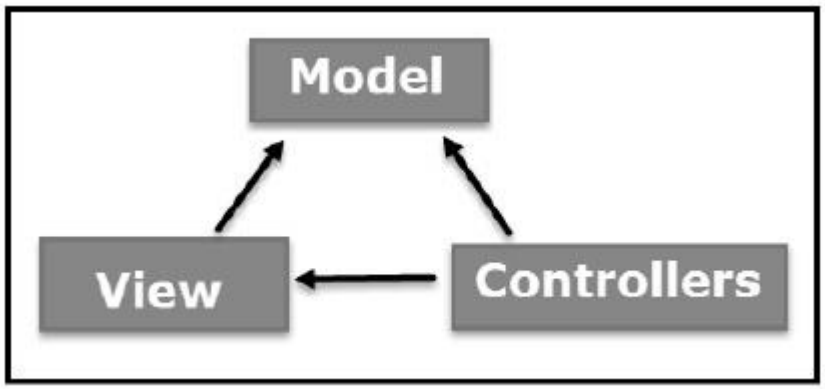


Рисунок 1.1 – Компоненти MVC.

Модель

Компонент Model відповідає всій логікі даних, з якою працює користувач. Це може представляти або дані, що передаються між компонентами View та Controller, або будь-які інші дані, пов’язані з бізнес-логікою. Наприклад, об'єкт "Клієнт" буде отримувати інформацію про клієнта з бази даних, маніпулювати нею та оновлювати дані назад у базу даних або використовувати її для рендерингу даних.

Вид

Компонент View використовується для всієї логіки інтерфейсу програми. Наприклад, подання Клієнт включатиме всі компоненти інтерфейсу, такі як текстові поля, випадаючі меню тощо, з якими взаємодіє кінцевий користувач.

Контролер

Контролери виступають інтерфейсом між компонентами Model і View для обробки всієї бізнес-логіки та вхідних запитів, маніпулювання даними за допомогою компонента Model та взаємодії з поданнями для надання кінцевого результату. Наприклад, контролер замовника буде обробляти всі взаємодії та входи з перегляду замовника та оновлювати базу даних за допомогою моделі замовника. Той самий контролер буде використовуватися для перегляду даних Клієнта.

1.3. Representational State Transfer

REST (Representational State Transfer) – це архітектурний стиль для розробки WEB-сервісів. REST популярний завдяки своїй простоті та тому, що він спирається на існуючі системи та особливості Інтернету. Протокол передачі гіпертексту (HTTP) для досягнення своїх цілей, на відміну від створення нових стандартів, рамок та технологій.

Переваги

Основною перевагою використання REST, як з точки зору клієнта, так і з боку сервера, є взаємодія на основі REST, що відбувається за допомогою конструкцій, знайомих кожному, хто звик використовувати HTTP в Інтернеті .

Прикладом цієї домовленості є взаємодія на основі REST, кожна з яких передає свій статус за допомогою стандартних кодів статусу HTTP. Отже, 404 означає, що запитаний ресурс не знайдено; код 401 означає, що запит не був авторизований; код 200 означає, що все в порядку; а 500 означає, що на сервері сталася невиправна помилка програми.

Подібним чином такі деталі, як шифрування та цілісність передачі даних, вирішуються не шляхом додавання нових фреймворків або технологій, а натомість, спираючись на добре відомі шифрування рівня захищених сокетів

(SSL) та захист транспортного рівня (TLS). Отже, вся архітектура REST побудована на концепціях, з якими більшість розробників вже знайомі.

REST – це також незалежний від мови архітектурний стиль. Додатки на базі REST можна писати будь-якою мовою, будь то Java , Kotlin, .NET, AngularJS або JavaScript. Поки мова програмування може робити WEB-запити за допомогою HTTP, цією мовою можна використовувати для виклику RESTful API або WEB-служби. Подібним чином WEB-сервіси RESTful можна писати будь-якою мовою, тому розробники, яким доручено впроваджувати такі сервіси, можуть вибрати технології, які найкраще підходять для їх ситуації.

Інша перевага використання REST – це його поширеність. На стороні сервера існує безліч фреймворків на основі REST, які допомагають розробникам створювати WEB-сервіси RESTful, включаючи RESTlet та Apache CXF. З боку клієнта, всі нові фреймворки JavaScript, такі як JQuery, Node.js, Angular та EmberJS, мають стандартні бібліотеки, вбудовані в їх API, які роблять виклик WEB-служб RESTful та споживають дані на основі XML або JSON, які вони повертають відносно прямолінійна діяльність.

Недоліки

Перевага REST з використанням конструкцій HTTP також створює обмеження. Багато обмежень HTTP також перетворюються на недоліки архітектурного стилю REST. Наприклад, HTTP не зберігає інформацію, засновану на стані, між циклами відповіді запиту, що означає, що додатки на базі REST мають бути без стану, а клієнт повинен виконувати будь-які завдання управління станом.

Подібним чином, оскільки HTTP не має жодного механізму для надсилання push-сповіщень із сервера на клієнта, важко реалізувати будь-який тип служб, де сервер оновлює клієнта без використання опитування сервера на стороні клієнта або деяких інших тип WEB-гачка.

З точки зору впровадження, загальною проблемою REST є той факт, що розробники не погоджуються з тим, що саме означає бути заснованим на REST.

Деякі розробники програмного забезпечення неправильно вважають RESTful те, що не базується на SOAP. Основною помилковою думкою про REST є той факт, що це архітектурний стиль, тому не існує еталонної реалізації або остаточного стандарту, який підтверджував би, чи є даний проект RESTful. В результаті виникає дискусія щодо того, чи відповідає даний API принципам REST.

Формати даних JSON та XML REST

```
{"employees":[
  { "firstName":"John", "lastName":"Doe" },
  { "firstName":"Anna", "lastName":"Smith" },
  { "firstName":"Peter", "lastName":"Jones" }
]}
```

Рисунок 1.2 – Приклад JSON.

Два найпоширеніші формати обміну даними – це JSON (рис 1.2) та XML (рис 3), і багато WEB-служб RESTful можуть використовувати обидва формати як взаємозамінні, доки клієнт може вимагати, щоб взаємодія відбувалася в XML або JSON.

```
<employees>
  <employee>
    <firstName>John</firstName> <lastName>Doe</lastName>
  </employee>
  <employee>
    <firstName>Anna</firstName> <lastName>Smith</lastName>
  </employee>
  <employee>
    <firstName>Peter</firstName> <lastName>Jones</lastName>
  </employee>
</employees>
```

Рисунок 1.3 – Приклад XML.

Потрібно зауважити, що хоча JSON і XML є популярними форматами обміну даними, сам REST не встановлює жодних обмежень щодо формату. Насправді, деякі WEB-служби RESTful обмінюються двійковими даними задля

ефективності. Це ще одна перевага для роботи з WEB-службами, що базуються на REST, оскільки архітектору програмного забезпечення надається велика свобода з точки зору того, як найкраще впровадити послугу.

REST та методи HTTP

Наведений вище приклад стосувався лише доступу до даних.

Операцією HTTP за замовчуванням є GET, яка призначена для використання при отриманні даних із сервера. Однак HTTP визначає низку інших методів, включаючи PUT, POST та DELETE.

Філософія REST стверджує, що для видалення чогось на сервері ви просто використовуєте URL-адресу ресурсу та вказуєте метод DELETE HTTP. Для збереження даних на сервері буде використана URL-адреса та метод PUT. Для операцій, які задіяні більше, ніж просто збереження, читання або видалення інформації, може бути використаний метод POST HTTP.

Альтернативи REST

Альтернативні технології для створення систем на основі SOA або створення API для виклику віддалених мікросервісів включають XML через HTTP (XML-RPC), CORBA, RMI через IIOP та Простий протокол доступу до об'єктів (SOAP).

Кожна технологія має свій власний набір переваг та недоліків, але переконливою особливістю REST, яка її відрізняє, є той факт, що замість того, щоб просити розробника працювати з набором користувацьких протоколів або створювати спеціальний формат даних для обміну повідомленнями між Клієнта та сервера, REST наполягає, що найкращим способом реалізації мережевої WEB-служби є просто використання базової конструкції самого мережевого протоколу, що у випадку Інтернету є HTTP.

Це важливий момент, оскільки REST не призначений лише для Інтернету; швидше, його принципи призначені для всіх протоколів, включаючи WEBDAV та FTP.

РОЗДІЛ 2. ОПИС ПРОГРАМНИХ ЗАСОБІВ

2.1. PhpStorm

Загальна інформація

PhpStorm являє собою спеціалізований засіб WEB-розробки, котрий орієнтований на WEB-додатки та інші види програм, які можна створювати за допомогою мови PHP та з використанням HTML, JavaScript і CSS. Рішення PhpStorm здійснює розгортання і синхронізацію проектів через протокол FTP. PhpStorm надає функції автоматичного завершення конструкцій мови PHP в коді, інспектування коду, різні алгоритми рефакторинг та зручну і швидку навігацію по коду.

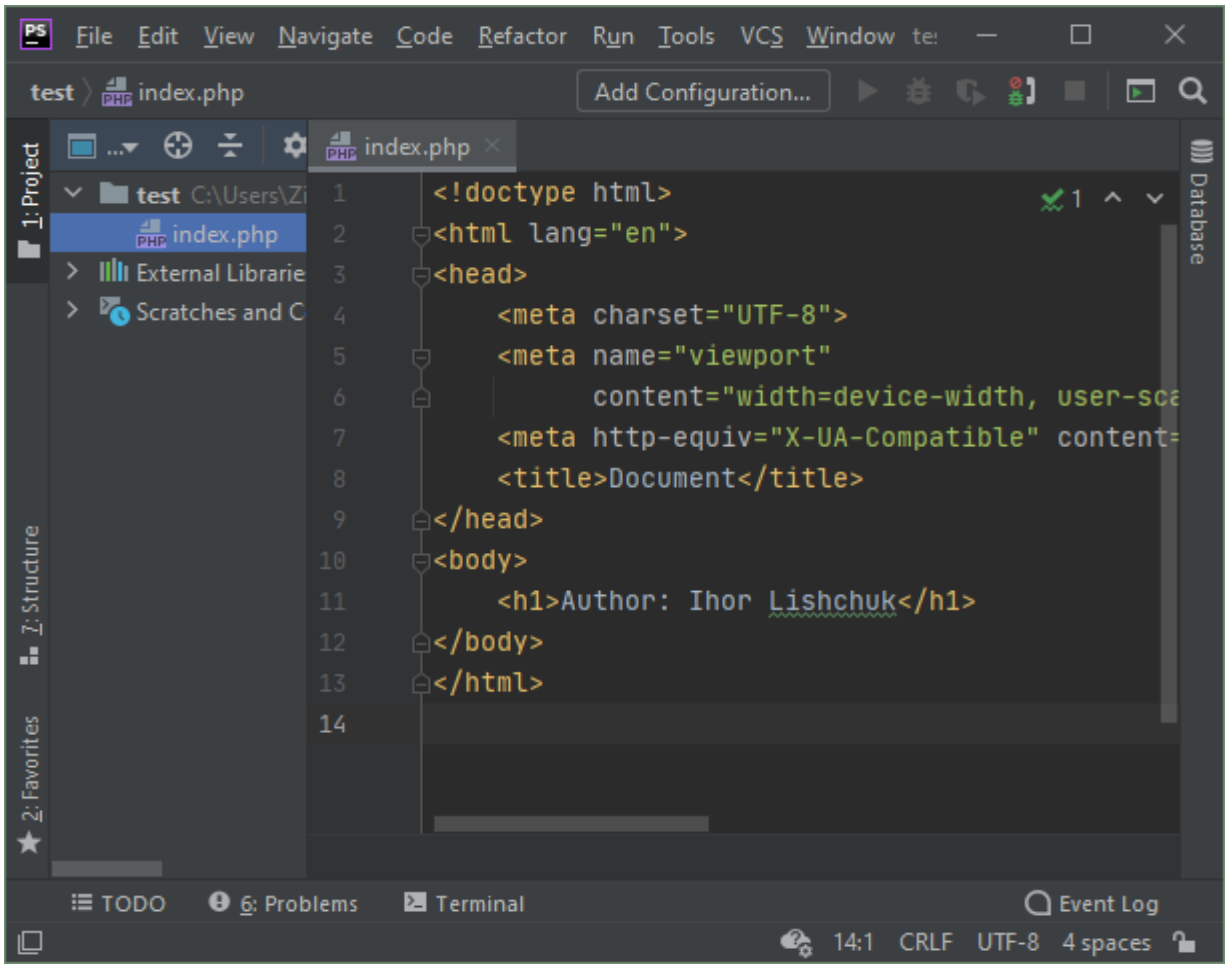


Рисунок 2.1 – Головний екран IDE PhpStorm.

Реалізований в PhpStorm графічний PHP-відладчик підтримує умовні точки зупину, відстеження значень і автоматизований вхід в налагодження окремих процедур. Для тестування додатків підтримується каркас тестових модулів PhpUnit і графічний інтерфейс для запуску тестів. При редагуванні коду виділяються конструкції синтаксису, здійснюється розширене форматування конфігурації, виявлення помилок в режимі реального часу і завершення коду. PhpStorm-редактор враховує коментарі до коду при його завершенні, автоматично вибираючи оптимальне рішення проблеми. PHP-рефакторинг і редагування шаблонів гарантує зміна проекту в найкоротші терміни. PhpStorm дозволяє візуалізувати код в ієрархічності вигляді і забезпечує швидку навігацію по всіх елементах.

Завдяки застосуванню PHP Unit-тестів можна швидко переглядати результати генерації коду окремих блоків або всього програми. Якщо тест був проведений невдало, продукт дозволяє переглядати окремі кодові рядки, в яких балу виявлена помилка. PhpStorm забезпечує налагодження коду JavaScript і надає широкий діапазон можливостей: знаходження точки зупину в HTML і JavaScript, настройка параметрів точки зупину, тестування синтаксису коду в режимі реального часу і т. Д.

Сотні інспекцій піклуються про верифікації коду, аналізуючи проект цілком під час розробки. Підтримка PHPDoc, code (re) arranger, форматування коду з конфігурацією стилю коду і інші можливості допомагають розробникам писати охайний і легко-підтримуваний код.

Підтримуються передові технології WEB-розробки, включаючи HTML5, CSS, Sass, SCSS, Less, Stylus, Compass, CoffeeScript, TypeScript, ECMAScript Harmony, шаблони Jade, Zen Coding, Emmet, і, звичайно ж, JavaScript.

PhpStorm включає в себе всю функціональність WEBStorm (HTML / CSS редактор, JavaScript редактор) і додає повнофункціональну підтримку PHP і баз даних / SQL.

2.2. Мова програмування PHP

2.2.1. Загальна інформація

PHP являється мовою сценаріїв з відкритим кодом, на стороні сервера, що використовується для розробки WEB-додатків. Під мовою сценаріїв мається на увазі програму на основі сценарію (рядки коду), написану для автоматизації завдань.

Рисунок 2.2 – Логотип PHP.

Що означає відкритий код? Подумайте, як виробник автомобілів робить секрет своїх дизайнерських моделей та технологічних інновацій доступними для всіх зацікавлених. Ці деталі дизайну та технології можуть бути перерозподілені, модифіковані та прийняті без побоювань будь-яких юридичних наслідків. Сьогоднішній світ міг розробити дивовижний суперкар!

WEB-сторінки можуть бути розроблені за допомогою HTML. За допомогою HTML виконання коду здійснюється в браузері користувача (на стороні клієнта). З іншого боку, з мовою сценаріїв на стороні сервера PHP, вона виконується на сервері, перш ніж потрапити до WEB-браузера користувача.

PHP може бути вбудований у HTML, і він добре підходить для WEB-розробки та створення динамічних WEB-сторінок для WEB-додатків, програм електронної комерції та баз даних. Вважається доброзичливою мовою, яка здатна легко підключатись до MySQL, Oracle та інших баз даних.

PHP являється найпопулярнішою мовою сценаріїв на стороні сервера. Він призначений для WEB-розробки та програмування загального призначення. Розроблено було у 1994 році Расмусом Лердорфом. Протягом більше двох десятиліть розвитку PHP бачив як максимуми, так і мінімуми. Зараз PHP управляється The PHP Group і знаходиться в постійному розвитку. PHP розшифровується як Hypertext Preprocessor, який було змінено з початкової назви "Персональна домашня сторінка".

PHP в основному використовується разом із HTML-кодом, системою управління WEB-вмістом, системами WEB-шаблонів та іншими популярними WEB-фреймворками. Мова PHP обробляється сервером або Common Gateway Interface (CGI). У будь-якому випадку, його можна використовувати для створення дивовижного WEB-додатку, де PHP-код завжди виконується на стороні сервера. Ви також можете виконати PHP-код за допомогою інтерфейсу командного рядка (CLI). Крім того, його також можна використовувати для реалізації графічного додатку, який є автономним за своєю суттю.

2.2.2. Історія

Історія PHP по-своєму унікальна. Все розпочалося, коли Расмус Лердорф почав експериментувати та працювати над інтерфейсом командного шлюзу (CGI) з акцентом на створенні власної домашньої сторінки. Він хотів розширити функціональність CGI на форми та забезпечити можливість взаємодії бланків із базою даних для передачі та зберігання інформації. Він досяг успіху в своїх спробах і назвав рішення PHP / FI або "Персональною домашньою сторінкою / перекладачем форм". Пізніше назву було змінено на "Гіпертекстовий препроцесор".

Його робота була зроблена в основному з Зеєвим Сураскі та Енді Гутманом. Обидва вони працювали над парсером, який використовувався в PHP 3.

Спочатку він використовувався для створення простого, але динамічного WEB-додатку. Щоб забезпечити належний ріст і основний розвиток PHP, Лердорф випустив PHP в дискусійній групі Usenet 8 червня 1995 р. Під версією PHP 1.0. Початковий випуск був потужним, і випуск PHP 2013 року несе основні функції першої версії. Пізніше додаються нові функції, включаючи змінні типу Per, вбудовування HTML, обробку форм. Що стосується синтаксису, він має схожість з Perl, але є простішим у порівнянні з ним.

PHP / FI можна використовувати для створення простих, динамічних WEB-додатків. Для пришвидшення повідомлення про помилки та вдосконалення коду Лердорф спочатку оголосив про випуск PHP / FI як "Інструменти особистої домашньої сторінки (Інструменти PHP) версії 1.0" в обговорюваній групі Usenet comp.infosystems.www.authoring.cgi 8 червня 1995 р. У цьому випуску вже була основна функціональність, якою володіє PHP станом на 2013 рік. Це включало Perl-подібні змінні, обробку форм та можливість вбудовувати HTML. Синтаксис нагадував синтаксис Perl, але був простішим, обмеженішим та менш послідовним.

PHP ніколи не задумувався як мова програмування, але він уже привернув увагу аудиторії, і органічний трафік зростав повільно. Щоб забезпечити зростання PHP у правильному напрямку, в середині 1997 року створена нова команда розробників. У листопаді 1997 року вийшла робоча версія мови програмування PHP, яка була відома як PHP / FI 2.

PHP ніколи не задумувався як мова програмування, але він уже привернув увагу аудиторії, і органічний трафік зростав повільно. Щоб забезпечити зростання PHP у правильному напрямку, в середині 1997 року створена нова команда розробників. У листопаді 1997 року вийшла робоча версія мови програмування PHP, яка була відома як PHP / FI 2.

PHP вмирає? Це одне з найбільш поширених питань в Інтернеті. Хоча багато людей вірять, що з PHP нічого не трапиться, але поганий початковий дизайн – це

те, що робить PHP таким поганим і не подобається спільноті. Є багато бібліотек низького рівня, які не відповідають умовам іменування, і для розробників може виникнути плутанина у підтримці та розробці за допомогою PHP.

2.2.3. Інструменти

На даний момент PHP – найпопулярніша WEB-мова програмування. Величезна популярність також означає, що є на вибір сотні і тисячі інструментів PHP. Інструменти допомагають вам впоратися з усією важливою частиною робочого процесу та дозволяють бути максимально продуктивними. Давайте ознайомимося з найкращими PHP – інструментами, які ви можете використовувати у своїй роботі.

DebugBar – це все, що вам потрібно. Це дуже простий інструмент, який дозволяє упорядковувати та збирати всі дані профілювання. Крім того, він не залежить від будь-якого іншого інструменту чи плагіна і може легко обробляти загальні колектори даних, запити Ajax та іншу форму інформації. Інформаційний рядок знаходиться у нижньому колонтитулі, щоб забезпечити легкий доступ.

Monsta FTP – це хмарне програмне забезпечення з відкритим кодом PHP / Ajax, яке надає керування файлами FTP прямо у вашому браузері, де завгодно та в будь-який час. Ви можете перетягувати файли у свій WEB-переглядач і спостерігати, як вони завантажуються, як магія. Він підтримує екранне редагування файлів. Також є багатомовна підтримка. Він був протестований на Chrome, Firefox, Internet Explorer і Safari. Він випускається під загальною публічною ліцензією GNU. Ви можете завантажити безкоштовно та встановити його на своєму власному сервері.

Рисунок 2.3 – Логотип Phalcon.

Phalcon PHP – це WEB-фреймворк, що постачається як розширення C, що забезпечує високу продуктивність та нижче споживання ресурсів. Phalcon PHP написаний на C з урахуванням незалежності платформи. Як результат, Phalcon PHP доступний у Microsoft Windows, GNU / Linux, Mac OS X. Ви можете або завантажити двійковий пакет для обраної вами системи, або створити його з джерел.

Pinba – це розумне програмне забезпечення, яке забезпечує статистику PHP-серверів у реальному часі. Він використовує MySQL як інтерфейс лише для читання.

CaseBox – це проста програма з відкритим кодом, яка дозволяє керувати завданнями, записами та файлами. Він побудований за допомогою PHP / MySQL. Крім того, він має настільний інтерфейс, що дозволяє працювати з легкістю. Також його можна використовувати для створення завдань із термінами. Інші ключові функції включають зберігання інформації про клієнта, швидкий пошук та багато іншого.

Munee – це багатоцільова бібліотека PHP, яка дозволяє робити багато речей. Він може використовуватися для маніпулювання ресурсами WEB-сайту та оптимізації додатків. Все це можна зробити завдяки потужному механізму кешування, який використовується як на сервері, так і на стороні клієнта. Крім того, його також можна використовувати для маніпуляцій та оптимізації зображень. Іншими завданнями оптимізації, з якими він добре справляється, є

мініфікація SCSS, LESS, CoffeeScript та інших мов програмування. Ви також можете використовувати плагін для комбінування файлу CSS + JS, зменшуючи розмір ресурсів.

PHPImageWorkshop – це бібліотека обробки зображень, яка дозволяє обробляти всю обробку зображень. Він настільки потужний, що може працювати подібно до GIMP та Photoshop. Наприклад, ви можете працювати з шарами зображень та іншими функціями, такими як обрізання, обертання тощо. Бібліотека є відкритим кодом і може використовуватися безкоштовно.

Рисунок 2.4 – Логотип Sylius.

Sylius – ще один інструмент з відкритим кодом для PHP. Це рішення для електронної комерції, яке пропонує комплексний менеджмент продуктів, управління категоріями тощо. Він також підтримує способи доставки, податкові ставки та платіжні шлюзи.

Pico – це невеликий додаток для управління вмістом, який дозволяє робити обмежені дії. І тому він популярний серед користувачів. Він використовує базу даних плоских файлів та збережені дані у файлах формату .md. Якщо ви любите знижку, вам також сподобається Піко.

phpMyFAQ – це ще один інструмент з відкритим вихідним кодом, який ви можете використовувати при розробці додатків для відповіді на поширені запитання щодо PHP. Його можна використовувати для створення приємних систем FAQ. Крім того, ви можете створити інтерфейс адміністратора і працювати

над відносно простими способами роботи. Він також пропонує вдосконалену систему пошуку, а для налаштування платформи можна використовувати інтерфейсні технології, такі як HTML-CSS. Ви також можете експортувати формат PDF, якщо це потрібно.

2.3. Що таке Phar

Для розробки WEB-додатків, якщо ви не отримаєте правильних інструментів, тому процес розробки може стати важким і болючим. При розробці програми на Java, там використовується файл Jar (Jar – це абревіатура від Java ARchive's). Додаток, включаючи всі доступні виконувані файли, упаковані у файл JAR, що робить процес розгортання дуже простим.

Рисунок 2.5 – Логотип Phar.

PHAR ("Php ARchive") схожий на PHP в одному з типів файлів JAR пакета. Якщо ви використовуєте PHP 5.3 або пізнішої версії, тоді розширення файлу Phar увімкнено за замовчуванням, що підтримується за замовчуванням, для його використання не потрібна додаткова установка.

Якщо ви раніше не використовували файл Phar, цей файл повинен представити деякі важливі функції цього файлу. Phar – це дуже корисна техніка, яка допоможе швидше розробити та розгорнути PHP та покращити досвід роботи.

За замовчуванням файл PHAR доступний лише для читання, використання файлу Phar не вимагає жодної конфігурації. Дуже легко розгорнути. Якщо потрібно створити власний файл Phar, то необхідно дозволити записувати файли Phar, для цего необхідно змінити параметр phar.readonly на 0 в файлі php.ini.

Переваги Phar в PHP

- PHAR легко встановлюється, а якщо точніше – він входить в стандартну поставку PHP 5.3;
- легко використовувати;
- зручно при копіюванні файлів на інший хост;
- легко розгортати – всього один файл;
- високий ступінь захисту PHP додатки – сигнатури, OpenSSL;
- має високу продуктивність;

2.4. XAMPP

XAMPP – це міжплатформенний WEB-сервер із відкритим кодом, який складається з WEB-сервера, механізму баз даних MySQL та PHP та Perl пакетів програмування. Він складається та підтримується Apache. Це дозволяє користувачам створювати WEB-сайти WordPress в Інтернеті за допомогою локального WEB-сервера на своєму комп'ютері. Він підтримує Windows, Linux та Mac.

Рисунок 2.6 – Логотип XAMPP.

2.3.1. Визначення

XAMPP компілюється та підтримується Apache. Абревіатура XAMPP розшифровується як;

- X – [крос-платформні операційні системи], що означає, що він може працювати на будь-якій ОС Mac OX, Windows, Linux тощо;
- A – Apache – це програмне забезпечення WEB-сервера;
- M – MySQL – База даних;
- P – PHP;
- P – Perl – мова сценаріїв;

2.3.2. Причина використання XAMPP

XAMPP забезпечує просту у використанні панель управління для управління Apache, MySQL та іншими програмами без використання команд. Щоб використовувати PHP, нам потрібно встановити Apache та MySQL. Встановити Apache та налаштувати його непросто, оскільки його потрібно, серед іншого, налаштувати та інтегрувати з PHP та Perl. XAMPP вирішує всю складність налаштування та інтеграції Apache з PHP та Perl.

2.3.3. Загальна конфігурація WEB-сервера

- htdocs – це WEB-кореневий каталог. Усі наші PHP-коди будуть розміщені в цьому каталозі;
- mysql – цей каталог містить всю інформацію, пов’язану з механізмом баз даних MySQL, за замовчуванням він працює на порту 3306;
- php – цей каталог містить інсталяційні файли PHP. Він містить важливий файл з назвою php.ini. Цей каталог використовується для налаштування поведінки PHP на вашому сервері;

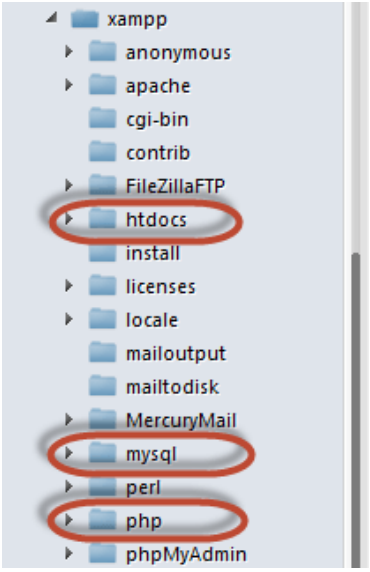


Рисунок 2.7 – Список основних каталогів.

За замовчуванням WEB-сервер Apache працює на порту 80. Якщо порт 80 бере інший WEB-сервер, ви можете використовувати інший номер порту. Примітка. Якщо ви використовуєте SKYPE, він використовує той самий порт. Закрийте Skype, якщо хочете використовувати XAMPP для PHP на порту 80.

2.3.4. Панель керування XAMPP

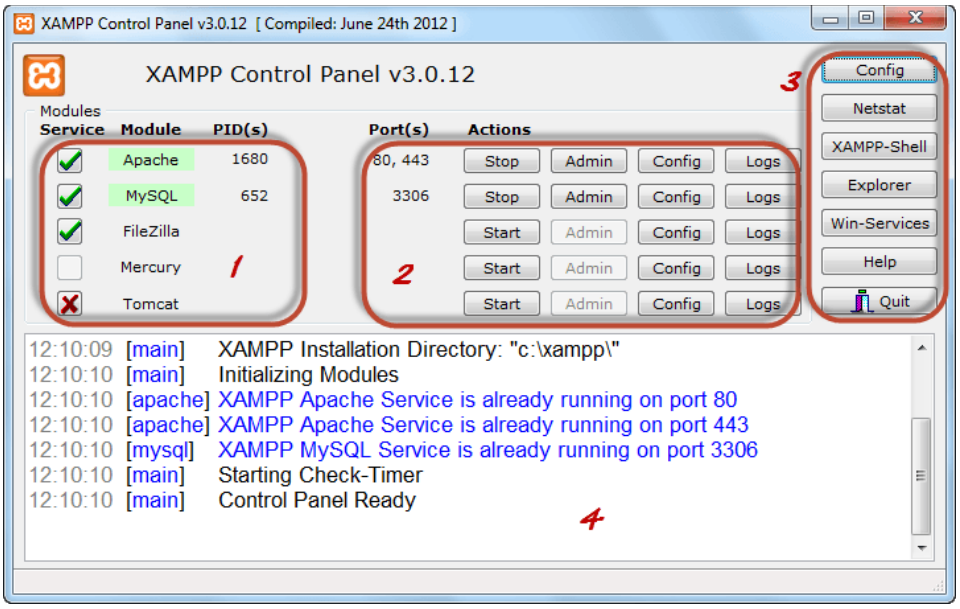


Рисунок 2.8 – Схема панелі керування.

1) У цьому розділі перелічені встановлені служби, модулі та ідентифікатори процесів (PID). Зелена галочка означає, що модуль встановлений як послуга. Червона позначка означає, що вона не була встановлена як послуга. Щоб встановити послугу, натисніть на червону позначку. Якщо кнопка відображає зелену галочку, і ви натискаєте на неї, панель управління запитає вас, чи хочете ви видалити систему.

2) У цьому розділі показані порти, пов'язані з модулями. Розділ дій призначений для:

- керування пусковими та зупиночними модулями;
- запуском адміністративних вікон для Apache та MySQL;
- відкриття файлу конфігурації для Apache, MySQL тощо, щоб внести зміни;
- перегляд файлів журналу для модулів;

3) Цей розділ містить корисні утиліти, такі як Netsat, ярлики служби Windows тощо.

4) Цей розділ відображає інформацію про стан модулів. Панель управління може бути використана для:

- встановлення та видалення модулів Apache, MySQL тощо, які встановлюються через XAMPP;
- запуск і зупинка послуг;
- відкривання конфігураційних файлів тощо;

2.5. Docker

2.4.1. Визначення

За своєю суттю Docker – це система контейнеризації, що включає багато корисних інструментів. Це дозволяє запускати програми в будь-якому місці, незалежно від того, як були створені ці програми або хост-система, на якій вони

запущені. Сюди входить програмне забезпечення, яке становить типовий WEB-стек, такий як Apache, MySQL та PHP.

"Це чудово, – можна сказати, – але чому це важливо?" Поки Drupal працює на підтримуваному WEB-стеку, яка різниця від цього?

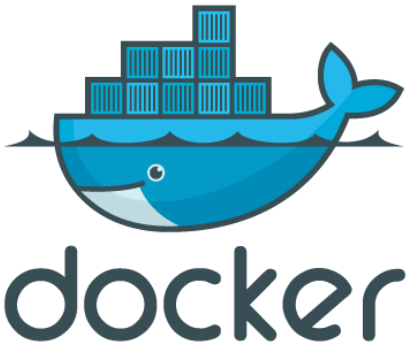


Рисунок 2.9 – Логотип Docker.

2.4.2. Проблема "Це працює в моїй системі"

Працюючи в команді, проблема часто з'являється у одного розробника, але не може бути відтворена в будь-якій іншій системі. Часто це пов'язано з невеликою різницею у конфігурації або версії програмного забезпечення сервера, що ускладнює пошук та налагодження.

У гіршому випадку це трапляється при розгортанні сайту на інфраструктурі, що працює. Відмінності у версіях серверного програмного забезпечення, конфігурації, доступності ресурсів або базовій операційній системі (ОС), яка запускає сервер, можуть створити несподівані та дорогі простої.

Рішення полягає в спробі максимально точно імітувати виробниче середовище. Але як?

2.4.3. Реплікаційні середовища

Якщо вам потрібно підтримати лише один проект на робочій станції, це не надто складна проблема. Отримавши робочі конфігураційні файли, потрібно завантажити та встановити конкретні версії для кожного серверного програмного

забезпечення. Це трудомісткий процес, але він може бути ефективним деякий час. Як тільки вам потрібно підтримати кілька проектів з різними вимогами до програмного забезпечення, стає важко керувати ними. Це може бути настільки просто, як різниця в конфігурації, але може бути настільки складним, як необхідність видалити та перевстановити різні версії серверного програмного забезпечення.

2.4.4. Все в одному середовищі WEB-розробки – не ідеально

Для вирішення цієї проблеми було створено все-в-одному середовища WEB-розробки. Такі продукти, як MAMP, WAMP та Acquia Dev Desktop, забезпечують зручний інтерфейс для створення декількох конфігурацій сайтів в одній системі. Кожен може підтримувати різні конфігурації та навіть різні версії серверного програмного забезпечення. Хоча це звучить як ідеальне рішення, вони мають кілька проблем.

Apache, PHP та MySQL ніколи не були повністю призначені для роботи разом з декількома версіями одного програмного забезпечення в одній системі. Якщо вам потрібен PHP 5.6 для одного сайту та PHP 7 для іншого, у центрі обробки даних ви б мали окремі сервери для кожного. Це неможливо під час роботи на ноутбуці чи локальній робочій станції. Середовища розробки «все в одному» часто вдаються до хитрощів, щоб паралельно запускати кілька версій та конфігурацій. Це може зробити конфігурацію схильною до поломок з часом.

Крім того, існує обмежена ізоляція між різними сайтами в одній системі. Якщо ви спробуєте змінити налаштування PHP для одного WEB-сайту, це може вплинути на кілька WEB-сайтів пізніше, що вимагатиме годин налагодження. Ця проблема посилюється під час запуску декількох версій одного програмного забезпечення в одній системі.

Прибирання – ще одна проблема. Багато з цих універсальних продуктів не відстежують кожен файл, пов’язаний із сайтом. Під час видалення одного в

інтерфейсі неможливо повністю видалити звисаючі файли конфігурації, глобальні зміни сервера та бази даних.

2.4.5. Переваги віртуальних машин

Усі ці проблеми призвели багатьох до альтернативи універсальним середовищам розробки – віртуальним машинам (ВМ). Віртуальні машини існують десятки років. Ви можете уявити це як тренажер польоту, лише для комп'ютерного обладнання. У симуляторі польоту ви отримуєте імітацію всіх індикаторів, циферблатів та елементів керування, необхідних для керування літаком. Аналогічним чином, віртуальна машина забезпечує віртуальний процесор, виділений обсяг пам'яті та віртуальний набір апаратних пристроїв. Замість пілотної версії віртуальна машина забезпечує основу для запуску цілої іншої операційної системи – як правило, Linux – поверх ОС вашого ноутбука, таких як macOS або Windows.

Віртуальні машини не є новими. Вони існують десятки років і є перевіреною технологією, яка приносить кілька переваг розробнику Drupal:

- Стандартизація. Віртуальну машину можна створити з точним серверним програмним забезпеченням та необхідними версіями або відтворити виробниче середовище аж до необхідної операційної системи.
- Спільний доступ. Знімок віртуальної машини можна зробити, коли створюється бажана конфігурація, а потім поділитися з усіма учасниками вашої команди. Це зменшує ймовірність неповторних проблем.
- Пісочниця. Ви можете мати різну віртуальну машину для кожного продукту, який розробляєте, кожен з різним набором серверного програмного забезпечення та конфігурацій. Усі файли у віртуальній машині відстежуються на віртуальному диску і можуть бути легко видалені, коли вони більше не потрібні.

2.4.6. Проблеми віртуальних машин

Віртуальні машини чудово звучать, але вони теж мають великий недолік: споживання ресурсів. Запуск іншої операційної системи вимагає нетривіального обсягу центрального процесора, пам'яті та диска. Навіть коли віртуальний диск не працює, віртуальний диск залишається на вашому жорсткому диску, забираючи гігабайти місця.

Пісочниця – це двосічний меч. Хоча це забезпечує ізоляцію між вашими проектами, це створює проблему дублювання. Якщо ви створюєте віртуальну машину для кожного проекту, над яким працюєте, деякі проекти будуть покладатися на одну і ту ж ОС, навіть на одне і те ж серверне програмне забезпечення. Більшість безкоштовних гіпервізорів із відкритим кодом – програмне забезпечення, яке запускає віртуальні машини – не дублює ці дані, що призводить до більш споживаного дискового простору.

Для багатьох віртуальні машини просто надмірні. Дійсно, вони віртуалізують і ізолюють все, але це часто більше, ніж те, що нам потрібно як розробникам Drupal.

2.4.7. Контейнери

Більшість гіпервізорів віртуалізують апаратне забезпечення, але переважна більшість WEB-сайтів Drupal працює на якійсь версії Linux. Різниця полягає лише у версіях та конфігурації програмного забезпечення WEB-сервера. Що робити, якщо замість віртуалізації обладнання для запуску певної ОС ми могли б віртуалізувати ОС і застосувати всі переваги віртуальної машини до запуску додатків?

Це те, що роблять контейнери. Контейнери – це свого роду віртуалізація операційної системи. Вони дозволяють запускати такі програми, як Apache, PHP та MySQL, у стандартизованому середовищі, яке можна розподілити та захистити від середовища. Замість того, щоб створювати абсолютно нову віртуальну

машину для кожного WEB-сайту, ви виділяєте лише контейнери, необхідні для підтримки вашого проекту.

Контейнери мають кілька великих переваг перед віртуальними машинами:

- швидкість. Набір контейнерів часто можна запустити і запустити за пару секунд, а не за хвилину і більше. Це дозволяє швидко перемикатися між проектами з різними наборами контейнерів;
- використання ресурсів. Оскільки контейнер повинен запускати лише одну програму, вони менші та потребують менше місця на диску та пам'яті порівняно з важкою ВМ;
- дедуплікація. Багато контейнерних систем також видаляють копії файлів, економлячи місце на диску;

Контейнери теж не нові. Вони є функцією багатьох операційних систем UNIX протягом багатьох років як альтернатива ВМ. Лише до Docker контейнери стали популярними серед розробників.

2.4.8. Що являє собою Docker?

Docker – це продукт, який дозволяє запускати контейнери, але він робить більше, ніж це, наступним чином:

- поширення. Docker пропонує легкодоступний і безкоштовний спосіб ділитися своїми контейнерами не лише зі своєю командою, але і з усім світом;
- інструменти розробки. Docker включає всі інструменти, необхідні для створення власних контейнерів з коробки. Після встановлення вам потрібен лише текстовий редактор та командний рядок;
- спрощені команди. Багато контейнерних систем досить складно налаштувати та використовувати, але Docker пропонує спрощену модель, яку можна використовувати лише за допомогою декількох команд;
- скупчення. Docker включає балансування навантаження навіть у безкоштовній версії продукту та з відкритим кодом. Ніщо не заважає

використовувати той самий час роботи Docker на своєму ноутбуці, що і на робочому WEB-сервері;

- стандартизація протягом життєвого циклу розвитку. За допомогою Docker на WEB-сервері ви можете використовувати ті самі контейнери – з тим самим програмним забезпеченням та конфігурацією – як і в процесі розробки. Це зменшує несподіванки запуску;

2.4.9. Навіщо використовувати контейнери?

Контейнери пропонують усі переваги віртуальних машин, включаючи ізоляцію додатків, економічну масштабованість та доступність. Але додатковий рівень абстракції (на рівні ОС) пропонує важливі додаткові переваги:

- менша вага: На відміну від віртуальних машин, контейнери не несуть корисного навантаження цілого екземпляра ОС - вони включають лише процеси ОС та залежності, необхідні для виконання коду;
- більша ефективність використання ресурсів: за допомогою контейнерів ви можете запустити в кілька разів більше копій програми на тому ж обладнанні, що і за допомогою віртуальних машин. Це може зменшити ваші хмарні витрати;
- покращена продуктивність розробника: порівняно з віртуальними машинами, контейнери швидше та простіше розгортати, надавати та перезапускати. Це робить їх ідеальними для використання в конвеєрах безперервної інтеграції та безперервної доставки (CI / CD), а також краще підходить для команд розробників, які застосовують практики Agile та DevOps;

РОЗДІЛ 3. ЗАГАЛЬНІ ВІДОМОСТІ WEB-ФРЕЙМВОРКУ

3.1. Розробка концепції

Цільова аудиторія WEB-фреймворку – програмісти, котрі займаються розробкою клієнт-серверних додатків (сайтів, API-сервісів і т. д.).

Короткий опис WEB-фреймворку:

Масштабована платформа для розробки клієнт-серверних WEB-додатків та сайтів, котра спрямована на полегшення та пришвидшення розробки програмістам за рахунок реалізації рутинних операцій та наданням необхідних інструментів для реалізації своїх проектів.

Переваги над іншими WEB-фреймворками:

Головною перевагою над іншими WEB-фреймворками являється те що даний WEB-фреймворк реалізовано у вигляді одного, стиснутого методом Phar, архіву. Програмістам не потрібно завантажувати велику кількість файлів (потрібно завантажити лише один файл і можна починати працювати), а потім дотримуватися структури папок обраного WEB-фреймворку. Структуру папок можна створити будь-яку. Також є можливість створити безліч проектів з різними точками входу. Наприклад API-сервер, CRON-задачу, PHP-demon, WEB-сайт і CLI-проект через одну точку входу (index.php). Так як WEB-фреймворк реалізовано у вигляді phar-архіву, то це дає додаткову безпеку для розроблених проектів, тому що через скрипти не можна буде відредагувати скрипти WEB-фреймворку. WEB-фреймворк не підключає велику кількість класів, котрі не завжди потрібні, натомість він підключає лише ті, які необхідно в певний момент в коді. Це реалізовано за допомогою автоматичних завантажувачів класів. Їх також можна редагувати, а саме додавати свої. Також є потужний Debug функціонал, за допомогою якого можна відслідковувати необхідні змінні в проекті. Без проблем можна додавати зовнішні бібліотеки через composer або схожий функціонал до WEB-фреймворку. Проекти, котрі будуть розроблятися

даним WEB-фреймворком, будуть використовувати, в основному, зручний ООП підхід розробки.

Розробка свого першого проекту:

Для розробки свого першого додатку необхідно:

- підключити phar-архів WEB-фреймворку через include;
- задати конфігурацію WEB-фреймворку;
- реалізувати клас «Точку входу» в свій проект, котрий буде унаслідуватися від спеціального інтерфейсу;
- додати клас, створений в пункті 3, в WEB-фреймворк через спеціально призначений для цього інструмент;
- запустити WEB-фреймворк;

3.2. Завдання WEB-фреймворку

Основні завдання WEB-фреймворку:

- пришвидшення розробки проектів;
- масштабованість проектів за рахунок можливості підключення абсолютно любих бібліотек до проекту;
- швидке перенесення проектів на різні хостинги за рахунок меншої кількості вихідних файлів;
- легка підтримка проектів;
- низький поріг входу для нових розробників;
- дотримування об'єктно-орієнтованого підходу розробки проектів;
- надання основного інструментарію для вирішення рутинних завдань;

Вхідні дані:

На вхід WEB-фреймворк отримує конфігурацію, перелік додатків, котрі можуть бути запущені, а також користувацькі скрипти.

Вихідні дані:

На вихід WEB-фреймворк видає результат роботи додатка, котрий був обраний як основний. Якщо такого додатку не було знайдено то буде запущено додаток за замовчанням. В тому випадку коли додатка за замовчанням не буде встановлено – виведеться помилка, котру можна буде побачити, включивши debug.

3.3. Структура WEB-фреймворку

Короткий загальний алгоритм точки входу (index.php файлу) майже кожного проекту зображено на рис. 3.1.

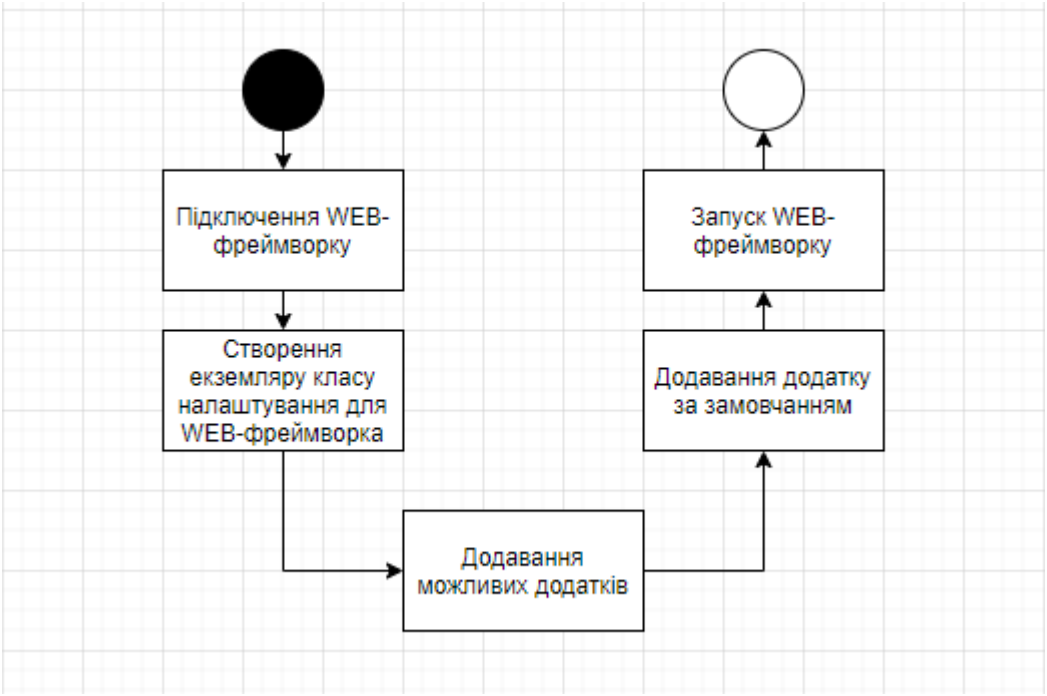


Рисунок 3.1 – Загальний алгоритм точки входу проекту.

Проте в залежності від ситуації він може розширятися різними допоміжними скриптами, котрі необхідні для роботи тих чи інших проектів. Такими скриптами можуть являтися оголошення необхідних констант, підключення залежних модулів та / або класів, додавання різних перевірок та інше.

Так як WEB-фреймворк запаковано в phar-архів, він все одно складається з різних файлів. Ці файли мають свою структуру (див. рис. 3.2).

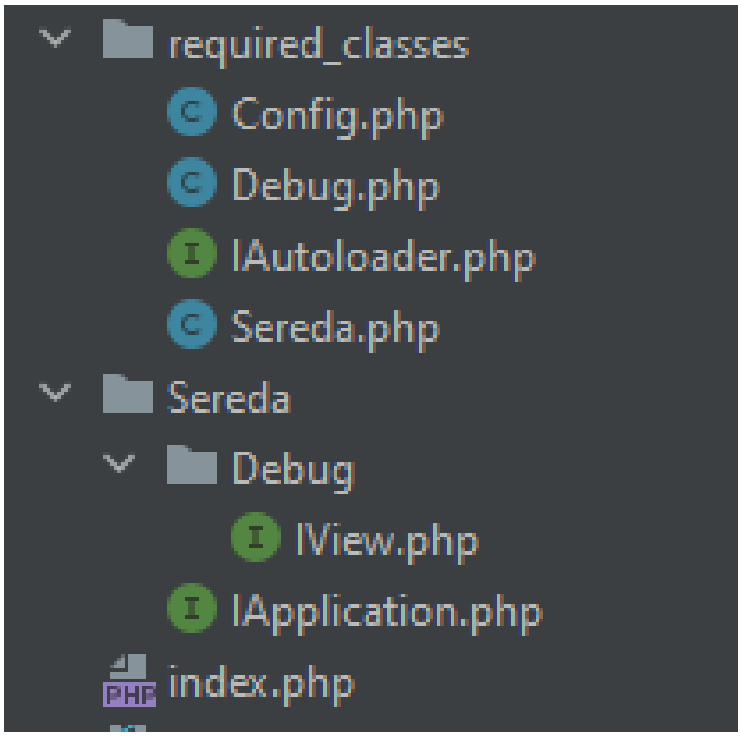


Рисунок 3.2 – Структура файлів WEB-фреймворка.

Пояснення щодо структури WEB-фреймворку:

- папка *required_classes* – папка, в якій знаходяться класи, котрі підключаються при запуску WEB-фреймворку;
- папка *Sereda* – папка, в котрій знаходяться допоміжні класи, які підключаються автоматично тоді, коли потрібно;
- файл *index.php* – скрипт, котрий виконує ініціалізацію WEB-фреймворку;

Скрипт ініціалізації WEB-фреймворку підключається через stub файл phar-архіву. Алгоритм скрипту зображено на рис. 3.3.

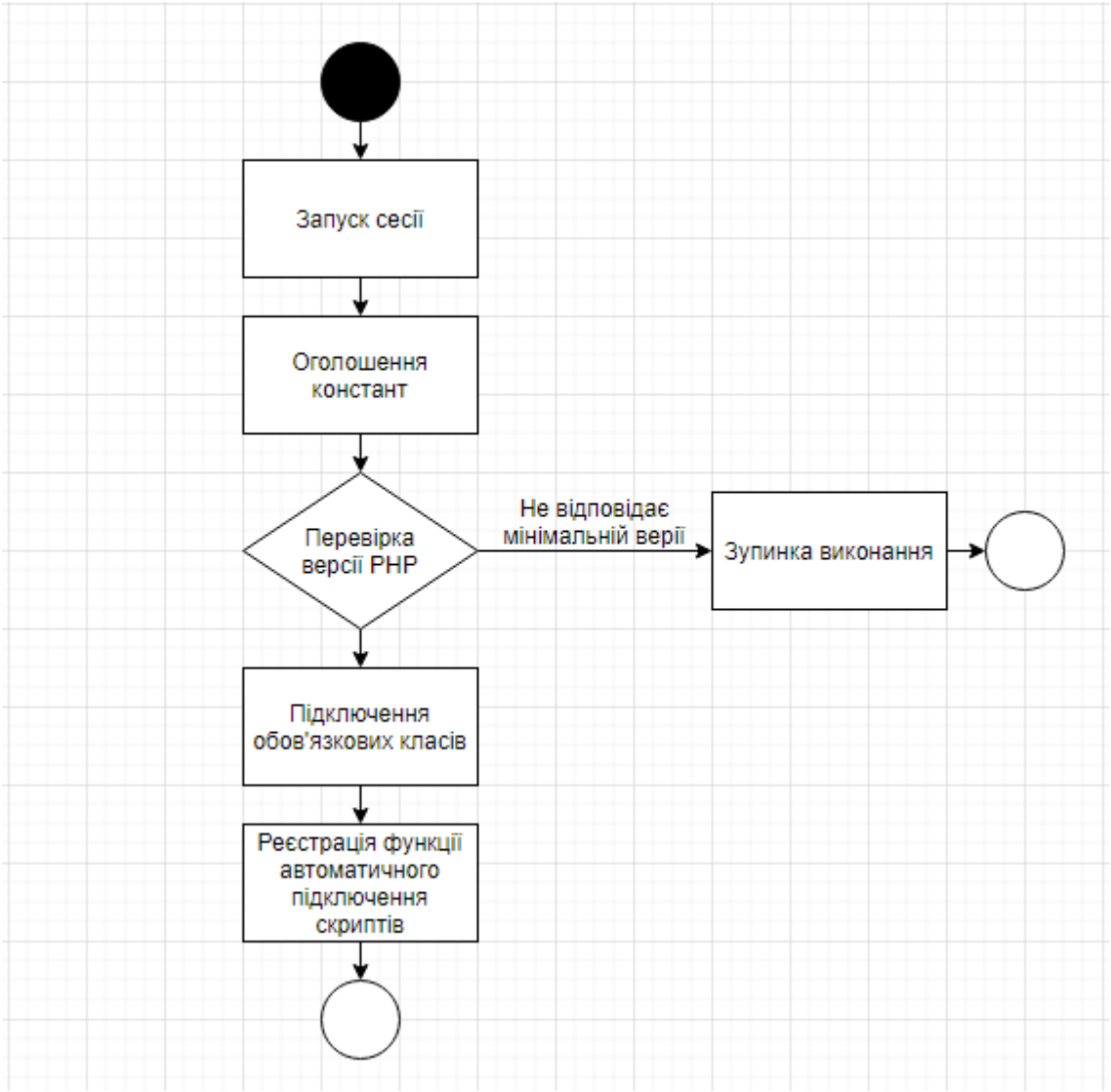


Рисунок 3.3 – Скрипт ініціалізації WEB-фреймворку.

Сесія запускається по замовчанню, так як вона потрібна в більшості проектів для реалізації внутрішнього функціоналу. Потім оголошуються дві константи: SEREDA_MINIMUM_PHP – мінімальна версія PHP, котра підтримується WEB-фреймворком, _SEREDA_PATH – шлях до папки з скриптами phar-архіву WEB-фреймворку.

Основний клас для запуску WEB-фреймворку це *Sereda* в ньому реалізовано метод start, котрий і запускає основний функціонал WEB-фреймворку після задання конфігурації. Його алгоритм зображено на рис. 3.4.

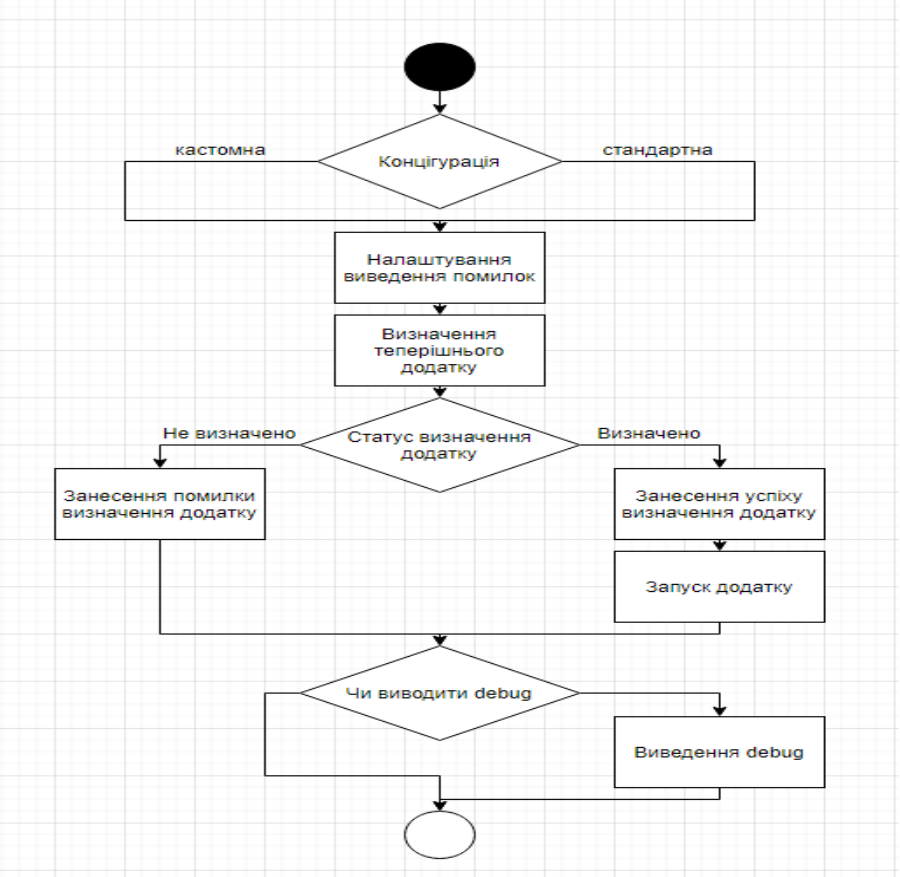


Рисунок 3.4 – Алгоритм методу start класу Sereda.

3.4. Основні особливості розробки WEB-фреймворку

WEB-фреймворк в результаті повинен упакуватися в phar-архів. Для того щоб упакувати проект в phar-архів можна налаштувати свій WEB-сервер для роботи з phar-архівом або використовувати сторонній інструмент.

Для того щоб налаштувати свій WEB-сервер необхідно мати версію PHP не менше 5.3 та в php.ini відредагувати phar.readonly на 0. Після цього можна створювати phar-архів штатними засобами WEB-сервера.

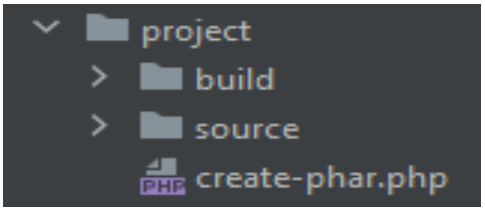


Рисунок 3.5 – Структура папок для створення phar-архіву.

Перед початком спочатку потрібно розділити папки для зручності (див. рис. 3.5). Наприклад build та source. В папці build буде збережено phar-архів і опціонально можна створити PHP-скрипт з підключенням phar-архіву. Папка source призначена для вихідних файлів проекту, котрі будуть поміщені в phar-архів. Наступним кроком буде створення скрипту для створення самого phar-архіву. Самий простий скрипт для створення phar-архіву зображено на рис. 3.6.

```php
<?php

$phar = new Phar( filename: 'build/project.phar');
$phar->startBuffering();
$phar->buildFromDirectory( directory: '/source');
$stub = "#!/usr/bin/env php \n{$phar->createDefaultStub( index: 'bin/stub.php')}";
$phar->setStub($stub);
$phar->stopBuffering();

```

Рисунок 3.6 – Скрипт для створення phar-архіву.

По коду видно що до архіву додається stub-файл (див. рис. 3.7), котрий викликається при підключені phar-архіву. В ньому зазвичай реалізується підключення скрипту ініціалізації.

```php
<?php
Phar::mapPhar();
require_once 'phar://' . __FILE__ . '/index.php';
__HALT_COMPILER(); ?>

```

Рисунок 3.7 – Скрипт stub-файлу.

Маленькою особливістю тут являється те що учитується назва phar-архіву, тобто назва phar-архіву може бути динамічною.

Також можна скористатися стороннім інструментом для створення phar-архіву. В моєму випадку таким являється сайт https://phar.scer.io/ (див. рис. 3.8). Він являється дуже простим та функціональним.

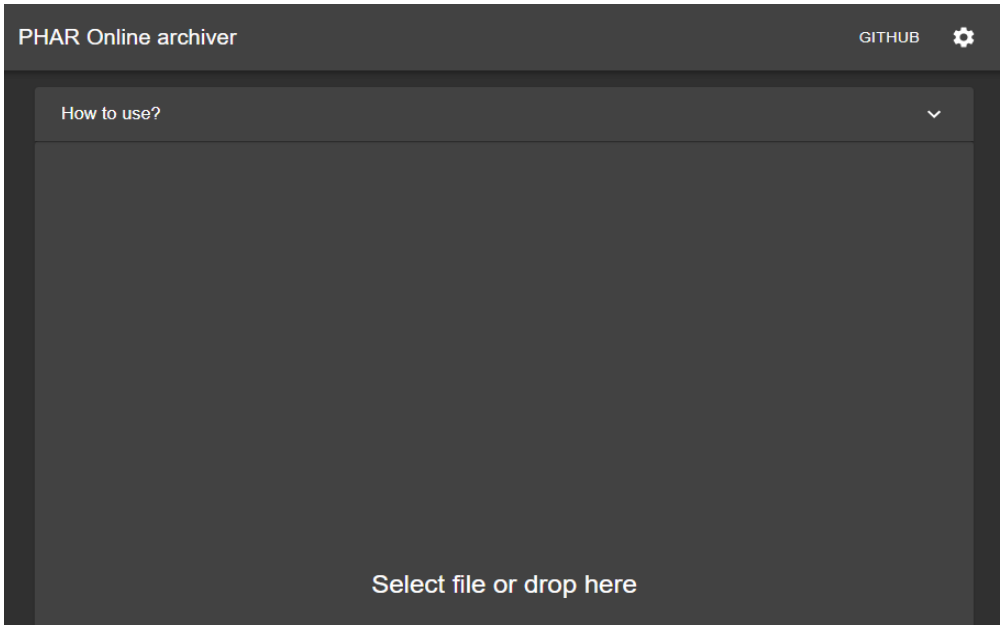


Рисунок 3.8 – Головний екран phar.scer.io.

Тут також можна налаштувати сигнатуру шифрування (із доступних MD5, SHA1, SHA256, SHA512), стиснення архіву і контент stub-файлу.

Рисунок 3.9 – Налаштування проекту phar.scer.io.

Для того щоб створити phar-архів необхідно помістити файли від проекту в zip-архів, після створення zip-архіву необхідно перемістити його (zip-архів) в область де написано «Select file or drop here». За лічені секунди створиться phar-архів, котрий можна буде використовувати в своїх проектах.

Щоб забезпечити керування автоматичним підключенням класів було реалізовано спеціальний метод в головному класі WEB-фреймворку та зареєстровано її через spl_autoload_register (див. рис. 3.10).

```
spl_autoload_register( callback: 'Sereda::autoload');
```

Рисунок 3.10 – Реєстрація метода підключення класів.

Після цього реалізований метод буде викликатися кожен раз коли буде викликатися не підключений клас. В методі (див. рис. 24) реалізований перебір доданих екземплярів класів та їх виклик. Зупиняється перебір тільки у випадку підключення потрібного класу. На вхід він приймає один аргумент – назву класу котрий викликається, але не являється підключеним.

```
public static function autoload($class_name)
{
    foreach (self::$autoload_functions as $autoloader_function) {
        if ($autoloader_function instanceof IAutoloader) {
            $autoloader_function->load($class_name);
            if (class_exists($class_name)) {
                Debug::success('sereda:autoload:loadClass', $class_name);
                return;
            }
        }
    }
}
```

Рисунок 3.11 – Лістинг зареєстрованого метода автоматичного підключення класів.

Також реалізовано кілька рівнів виведення помилок PHP в методі, котрий запускає WEB-фреймворк (див. рис. 3.11). Налаштовується він за допомогою конфігурації WEB-фреймворку, котра опціонально передається при запуску WEB-фреймворку. Список рівнів помилок включає:

- *default* – будуть показуватися стандартні налаштування виведення помилок PHP на WEB-сервері;
- *none* – не виводити взагалі помилки;
- *simple* – показувати не критичні помилки PHP. Сюди входять E_ERROR, E_WARNING або E_PARSE;
- *maximum* – виводити максимально доступні помилки визначені конфігурацією PHP на WEB-сервері;
- Development – примусово виводити абсолютно всі доступні помилки PHP на WEB-сервері;

Цей функціонал реалізований за допомогою методів PHP error_reporting та ini_set.

- error_reporting() – це метод, котрий заздалегідь визначений PHP. Це дозволяє контролювати кількість і помилки PHP, про які повідомлятиметься. Так як PHP має кілька рівнів помилок. Використання функції error_reporting() встановлює цей рівень протягом усього поточного сценарію;
- метод ini_set() дозволяє змінювати системні атрибути, що впливають на ваш скрипт виконується. Зміни стосуються лише поточного сценарію та повертаються назад, коли сценарій закінчується;

Лістинг цього функціоналу представлено на рис. 3.12.

```php
switch ($config->error_reporting) {
    case 'default':
        break;

    case 'none':
        error_reporting( error_level: 0);
        break;

    case 'simple':
        error_reporting( error_level: E_ERROR | E_WARNING | E_PARSE);
        ini_set( option: 'display_errors', value: 1);
        break;

    case 'maximum':
        error_reporting( error_level: E_ALL);
        ini_set( option: 'display_errors', value: 1);
        break;

    case 'development':
        error_reporting( error_level: -1);
        ini_set( option: 'display_errors', value: 1);
        break;

    default:
        error_reporting($config->error_reporting);
        ini_set( option: 'display_errors', value: 1);
        break;
}
```

Рисунок 3.12 – Лістинг скрипту встановлення рівня виведення помилок.

Функціонал Debug реалізовано у вигляді колекції, в котру додаються певні об’єкти. Список функцій для ведення логування:

- *error* – логування об’єктів, котрі являються помилками;
- *save* – метод призначений для логування свого типу об’єктів;
- *log* – логування об’єктів, котрі являються звичайними повідомленнями;
- *success* – логування об’єктів, котрі являються успішним виконанням;
- *warning* – логування об’єктів, котрі являються попередженнями;

Також є метод show, котрий вміє відображати список логованих об’єктів, а також розмальовувати кольорами типи об’єктив для зручності перегляду.

ВИСНОВКИ

У результаті виконання даної дипломної роботи було розроблено WEB-фреймворк для створення клієнт-серверних додатків різного спрямування. Робота містить головні принципи створення WEB-фреймворку.

- Під час роботи над дипломним проектом було проаналізовано основні найпопулярніші WEB-фреймворки, котрі доступні для завантаження кожному охочому. Також було розглянуто переваги та недоліки технологій, котрі використовуються при створенні WEB-фреймворку.
- Створено концепцію проекту та за допомогою WEB-сервера та IDE розроблено функціонал WEB-фреймворку і запаковано його в phar-архів для подальшого розповсюдження.

Було розроблено WEB-фреймворк з базовим функціоналом для розробки WEB-проектів, що включає надання необхідних інтерфейсів та реалізованих інструментів. Надана можливість переназначати своїми методами ключові методи стандартних бібліотек WEB-фреймворка. Для використання WEB-фреймворку необхідно мати WEB-сервер з версією PHP не менше ніж 7.0.

Методи розробки, що були використані при створенні WEB-фреймворку дозволяють нескінченно масштабувати майбутні проекти, постачати WEB-фреймворк різним набором інструментів. Від самого необхідного функціоналу до різних інструментів, котрі можуть бути використані для виконання різних задач та в різних ситуаціях.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ceruzzi, P.E. A history of modern computing / Ceruzzi P.E.. - M.: 2003. - 831 с.
2. Guo, Y. High Performance Data Mining: Scaling Algorithms, Applications and Systems / Guo Y., Grossman R. (eds.). - M.: 2002. - 219 с.
3. Karny, M. Probabilistic Advisory System / Karny M., Bohm J., Guy T.V.. - M.: 2003. - 532 с.
4. Weinberg, G.M. Psychology of computer programming / Weinberg G.M.. - M.: 1977. - 483 с.
5. Ye, N. The Handbook of Data Mining / Ye N.. - M.: [не указано], 2003. - 448 с.
6. Ceruzzi, P.E. A history of modern computing / Ceruzzi P.E.. - M.: 2003. - 864 с.
7. Cooper, R.B. Introduction to queueing theory / Cooper R.B.. - M.: 1981. - 399 с.
8. Cormen, T.H. Introduction to algorithms / Cormen T.H., Leiserson C.E., Rivest R.L.. - M.: 2001. - 979 с.
9. Crochemore, M. Jewels of stringology / Crochemore M., Rytter W.. - M.:, 2002. - 802 с.
10. Fokker, J. Functional Programming / Fokker J.. - M.: 1995. - 290 с.
11. Gacs, P. Complexity of Algorithms / Gacs P., Lovasz L.. - M.: 1999. - 838 с.
12. Gonnet, G.H. Handbook of algorithms and data structures / Gonnet G.H., Baeza-Yates R.. - M.: 1991. - 947 с.
13. Guo, Y. High Performance Data Mining: Scaling Algorithms, Applications and Systems / Guo Y., Grossman R. (eds.). - M.: 2002. - 766 с.
14. Herbrich, R. Learning Kernel Classifiers. Theory and Algorithms / Herbrich R.. - M.: 2002. - 207 с.
15. Hopcroft, J.E. Introduction to Automata Theory, Languages, and Computation / Hopcroft J.E., Motwani R., Ullman J.D.. - M.: 2001. - 566 с.

ДОДАТКИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

РОЗРОБКА ФРЕЙМВОРКУ ДЛЯ WEB-ДОДАТКІВ МОВОЮ PHP

Виконав студент 4 курсу
Групи ПД-42
Ліщук Ігор Валерійович
Керівник роботи
Дібрівний Олесь Андрійович

Київ – 2021

МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

- **Мета роботи** – прискорення часу розробки клієнт-серверних додатків, мінімізація витрат на готовий продукт, зменшення порогу входу для нових розробників, розбивка проекту на окремі модулі.
- **Об'єкт дослідження** – клієнт-серверні додатки та WEB сайти.
- **Предмет дослідження** – WEB-фреймворк з використанням мови PHP.

АНАЛОГИ

ТЕХНІЧНІ ЗАВДАННЯ

- *низький поріг входу;*
- *розробники самі визначають структуру проекту;*
- *весь фреймворк повинен являтися одним файлом;*
- *масштабованість проектів.*
- *перевизначення своїми методами ключові методи WEB-фреймворку для забезпечення більшої гнучкості проектів.*

ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

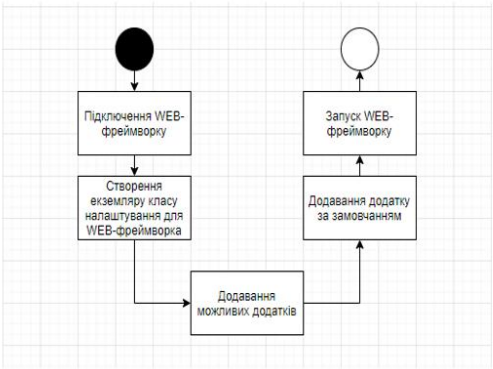
XAMPP

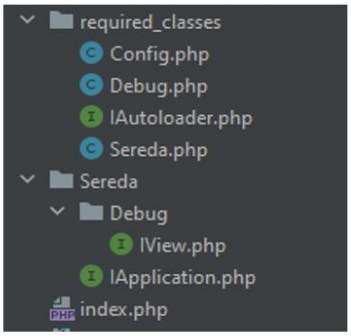
PhpStorm 2021.1

docker

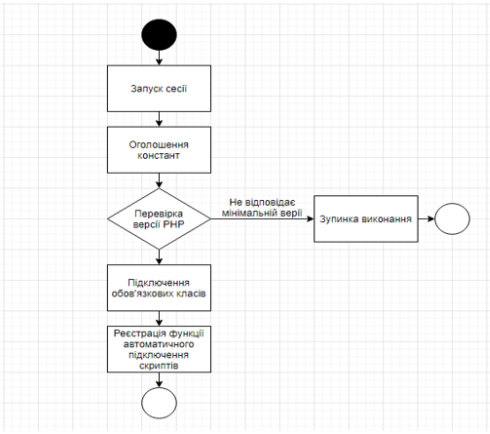
PHP

АЛГОРИТМИ ТА СТРУКТУРА





АЛГОРИТМИ ТА СТРУКТУРА



АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

- Ліщук І.В. Розробка WEB-фреймворку з використанням мови PHP / О.А. Дібрівний, І.В. Ліщук // Сучасний стан та перспективи розвитку IOT.
- Ліщук І.В. Розробка WEB-фреймворку для проектів на мові PHP / О.А. Дібрівний, І.В. Ліщук // Застосування програмного забезпечення в інфокомунікаційних технологій.

ВИСНОВКИ

- досліджено аналоги існуючих фреймворків;
- розроблено повністю функціонуючий фреймворк;
- розроблено функціонал, котрий пригодиться в більшості проектів.
- забезпечено підключення фреймворку через один файл (phar-архів).

Перспективи подальших досліджень

В майбутньому планується зробити не тільки програмний інтерфейс, а ще й графічний задля забезпечення меншого порогу входу та швидшої реалізації рутинних задач.

ДЯКУЮ ЗА УВАГУ!