**CodeIgniter 4**

**Техническа документация**

**Христо Владимиров 1701681020**

**Динчо Костадинов 1701681017**

**Съдържание**:

1. **Въведение**
   1. Обща информация
   2. Предимства и недостатъци
   3. В практиката
2. **Инсталация и употреба**
   1. Инсталиране
   2. Ръчна инсталация
   3. Инсталция чрез Composer
   4. Разработка
   5. Стартиране
3. **Технология**
4. **Компоненти**
   1. Маршрутизиране
   2. Контролери
   3. Entity/Model
   4. Локализация
   5. Филтри
   6. Email
   7. Сигурност
5. **Примерен проект**

**I. Въведение**

* 1. Обща информация: CodeIgniter е MVC(Model-View-Controller) фреймуърк за бързо изграждане на уеб приложения базиран на PHP. Известен е със своето бързодействие и с това че е по-лек от другите алтернативи написани на PHP.

Първата версия на CodeIgniter излиза през 2006 а последната (CodeIgniter) на 24фев 2020. Тази последна версия е базирана само на PHP7 и няма backwards compatibility и добавя опростено начално конфигуриране на проекта. Вгражда се и употребата на Entities. Новата версия е цялостно е по-оптимизирана, с което става осезаемо по-бърза. Също се подновява файловата и система.

2. Предимства и недостатъци: Codeigniter е лек, бърз, лесен за инсталация и за разширяване фреймуърк с добри вградени функционалности за сигурност, но за сметка на това са осезаемо по-малко на брой наличните библиотеки за разширяване особено в директно сравняване с други алтернативи.

3. В практиката: CodeIgniter се използва в много правителствени сайтове в САЩ. Сайта на часовниците GShok също е базиран на CodeIgniter.

**III. Инсталация и употреба**

1. Инсталиране: За да работи CodeIgniter e нужна съществуваща PHP версия 7.2 или по-висока. След като се уверим че имаме инсталиран PHP има ръчен и автоматичен начин на инсталация.

2. Ръчна инсталация:

1. Изтегля се zip-файл, който съдържа необходимите файлове за CodeIgniter проект.
2. Файлът се разархивира
3. (По избор) Създадената папка се преименува на името на проекта.

3. Инсталация чрез Composer:

1. Трябва да се позволи свалянето на разширения, като се редактира конфигурационния файл php.ini, който се намира в инсталационната директория на PHP:

extension=intl.dll;

extension=mbstring.dll;

или

extension=php\_intl.dll;

extension=php\_mbstring.dll;

просто се премахва знакът „;“ в края на на реда на една от двете двойки.

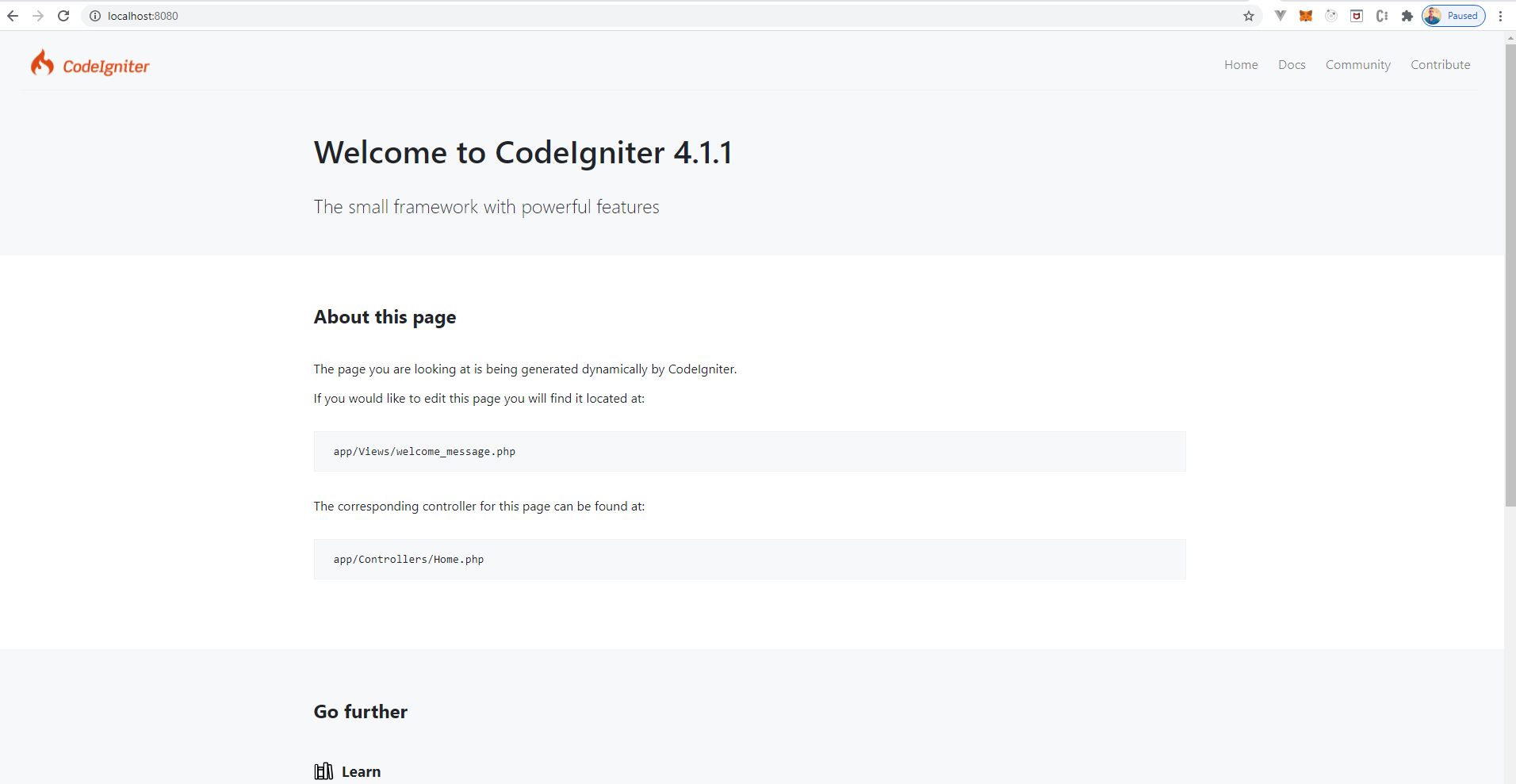
1. Изпълнява се командата `composer create-project codeigniter4/appstarter project-root` като ‘project-root’ трябва да бъде заменено с името на проекта

След като необходимите файлове са вече инсталирани, проектът е готов за разработване с инструментите на фреймуърка.

3. Разработване: Всички файлове, които се използват за разработване на приложение на рамката CodeIgniter, се намират в папката /app. В нея са поместени множество подпапки, предназначение да съдържат различните групи от файлове. Основната част от разработването е редактирането или добавянето на файлове в тези папки. Най-важните директории и техните предназначения са:

1. Config - Съдържа файлове, отговарящи за глобалните настройки на приложението и на модулите, които то използва
2. Controllers - Съдържа контролерите
3. Database - Съдържа история на основни операции за изграждането и попълването на базата данни
4. Entities - Съдържа класове, които представляват дадени таблици от базата данни
5. Filters - Съдържа класове, които дефинират дадена логика, която се изпълнява преди и/или след извикването на метод в контролерите
6. Language - Съдържа файловете, отговарящи за локализацията(превеждането) на приложението
7. Models - Съдържа класове, които осъществяват връзката между Entity клас и дадена таблица в базата данни и предоставят възможните операции с нея
8. Views - Съдържа, файлове, представляващи визуалната част на приложението

4. Стартиране: С командата ‘php spark serve’ се стартира проектът, след което се отваря браузъра с адрес: ‘http://localhost:8080’. Ако всичко е наред трябва да се покаже началната страница на CodeIgniter с линкове към документацията.



**III Технологии:** CodeIgniter е предназначен за изграждане на проекти на базата на шаблона MVC. MVC шаблонът се характеризира с разделянето на логиката на три слоя: Model – Централната част на шаблона. Представлява структурата на динамичните данни в приложението. Тази част е независима от потребителския интерфейс. В този слой се намира бизнес логиката на приложението, където се извършват операциите върху данните. View – Визуалната част на приложението. В този слой се съдържа логиката и форматът на информацията, която ще бъде представена на крайния потребител. Controller – Отговаря за получаването на входни данни/заявки от потребителя, валидирането им и преобразуването им в команди към моделния или визуалния слой. По-общо казано, когато потребителят отвори дадена страница или попълни някаква форма, се изпраща заявка към съответния контролер, който отговаря за тази страница(един контролер може да отговаря и за няколко страници), която може да съдържа входни данни. Контролерът от своя страна обработва заявката, валидира параметрите и ако всичко е наред, изготвя команда за моделът и я предава на моделният слой. Там се извършват необходимите операции спрямо командата - въвеждане, промяна, изтриване или извеждане на данни от дадена база данни и обработка на данните, ако е необходимо. След това моделът връща резултат на контролерът, на базата на който контролерът приготвя команда за визуализационния слой. Визуализационния слой приема командата от контролера, попълва данните, които е 6 получил, ако има такива, и приготвя крайния изглед в указан формат(най-често HTML). Обработеният изглед се връща на браузърът на потребителя, където бива визуализиран. CodeIgniter също така може да бъде конфигуриран да използва HMVC(Hierarchical MVC), който се отличава от MVC с това, че вместо приложението да разделено на три слоя, приложението е разделено на йерархия от модули, всеки от които съдържа трите MVC слоя.

**IV. Компоненти:**

1. Маршрутизирането по подразбиране е настроено да следва шаблонът /. Но доста често се налага да се конфигурира друг начин за дефиниране на маршрути, например, когато искаме да вземем някакъв продукт по идентификатор и трябва пътищата ни да изглеждат по следния начин:

/products/1

/products/2

/products/3

За дефинирането на маршрути в други формати, трябва да се въведат конфигурации в app/Config/Routes.php. За дефинирането на параметри в самите маршрути се използват заместители.

1. Контролерите като съставна част от MVC архитектурата обикновено стоят поместени в /app/Controllers/ папката. Те се грижат за приемането на команди и входни данни от браузъра на крайния потребител и координирането на тези команди и данни с моделите и визуализацията. За лесно изграждане на такъв контролер, CodeIgniter предоставя класа BaseController. Важно е да се спомене, че името на файла, съдържащ даден контролер винаги трябва да започва с главна буква и това да е единствената главна буква в цялото име. Всеки метод в контролера е предназначен за отделен маршрут в приложението. Например, ако имаме метод my() в контролер Notes.php, то този метод ще бъде повикан при поискването на маршрут /notes/my. Метод, който идва наследен от BaseController е index() и той ще бъде повикан когато се поискат маршрутите /notes и /notes/index, т.е. това е метод по подразбиране. Обекти, наследени от BaseController са:

* request - обект, съдържащ всички данни за получената заявка - метод(GET, POST, PUT, PATCH, DELETE и т.н.), параметри, хедери, данни за потребителската сесия и др.
* response - данни за обектът, който ще бъде върнат на браузърът като отговор на заявката - код за статус, тяло и др.
* logger - обект, използван за изграждане на логове
* forceHTTPS() - метод, който принуждава браузърът да праща само HTTPS заявки
* validator и validate() - за валидиране на данните е предоставена функцията validate(), с която може да се конфигурират кои полета какво изискват и каква грешка да бъде върната, ако изискванията не са спазени. Грешките се съхраняват в обекта validator.

1. Entity & Model: Entity класовете в CodeIgniter представляват дадена таблица в базата данни. Полетата на един такъв клас, всъщност представляват полетата на един ред от съответната таблица. Методите на класа служат за изграждането на бизнес логиката свързана със съответния ред.
2. Локализацията служи за превеждане на един сайт на различни езици и CodeIgniter предоставя много удобни готови функционалности за това.
3. Филтрите са предназначени за извършване на дадени действия непосредствено преди или след изпълнението на даден метод в контролер. Примерни действия, които могат да бъдат извършени с филтри са:

* извършването на CSRF защита на пристигащите заявки
* ограничаване на зони от приложението спрямо потребителска роля
* ограничаване на броя заявки към даден маршрут
* показване на страница “Извършва се профилактика”

За създаването на филтър трябва да се създаде клас, който имплементира интерфейсът CodeIgniter\Filters\FilterInterface. Така всеки филтър трябва да съдържа методите before() и after(), но е позволено някой от методите да остане празен, ако не е необходимо да се извършват действия преди или след дадения метод.

1. Email: Вграденият Email клас се грижи за лесно изпращане на имейли и поддържа следните функционалности: 11

* Множество протоколи: Mail, Sendmail и SMTP
* TLS и SSL криптиране за SMTP
* Множество получатели
* CC и BCCs
* HTML формат и чист текст
* Прикачени файлове
* Приоритети
* Инструменти за дебъгване на имейли

Изпращането на имейл е изключително лесно, като нужната конфигурация може да се декларира глобално в /app/Config/Email.php или в кода преди самото изпращане.

1. Сигурност: Най-важната функционалност, която е предоставена от Security модула на CodeIgniter е CSRF филтърът. Той подсигурява заявките срещу Cross Site Request Forgery, т.е. заявки, които не пристигат от браузъра на потребителя, а от друг източник, което е потенциална хакерска атака. CSRF защитата става, като се генерира жетон за всяка форма при всяко зареждане и след изпращане на формата, жетонът се валидира. Тази валидация се изпълнява автоматично от CSRF филтъра, който трябва да се конфигурира глобално под псевдонима ‘csrf’.

**V. Примерен проект.** В официалния сайт на CodeIgniter има туториъл за инсталация на рамката: <https://codeigniter.com/user_guide/tutorial/index.html>. Следвайки стъпките оказани там създадохме темплейтен проект и го стартирахме на локал хост на порт 8080.