
GETTING STARTED WITCH
CODEIGNITER 4
ANDRÉS CRUZ YORIS
WEB DEVELOPER

Getting started with CodeIgniter 4, master the basics of the PHP framework for beginners

Here begins your roadmap in the use of PHPs frameworks

Andres Cruz Yoris

This version was published: 2025-02-26

Post about the book!

Please help promote this book.
The suggested post for this book is:
I just bought the book "First steps with CodeIgniter 4" from @LibreDesarrollo! get your copy at:

https://www.desarrollolibre.net/libros/primeros-pasos-codeigniter-4

About the Author

This book was prepared by Andres Cruz Yoris, Bachelor of Computer Science, with more than 10 years of experience in the development of web applications in general; I work with PHP, Python and client-side technologies such as HTML, JavaScript, CSS, Vue among others; and on the server side like Laravel, Flask, Django and of course Codeigniter since version 2.

I put at your disposal part of my learning reflected in each of the words that make up this book, my first book, on web development; I hope it is of your interest and you can enjoy it.

Copyright

No part of this book may be reproduced or transmitted in any way; that is, electronically or by photocopies without the permission of the author.

Foreword

CodeIgniter is a fascinating framework, and it is ideal when you are entering the world of programming, you already know how to program in PHP and what this language and the technologies that revolve around it, such as HTML, CSS and JavaScript, and want to go a step further, you want to start developing in a more professional, efficient and scalable way over time; although the fact of using CodeIgniter or another framework is not a guarantee that you will be able to take that step, since the most important thing is that you know how to use it, and that is what this book is for, so that you have a progressive advance, from scratch and with a curve of moderate learning; taking the first steps with CodeIgniter explaining what each component works for and, at once, doing practice, programming your first application that will surely be the first of many more.

Although I mentioned earlier that CodeIgniter is ideal for a beginner, it does not necessarily have to be used as a simple stepping stone to reach higher and work with more complete and complex frameworks such as Laravel or Symfony.

CodeIgniter is a fairly powerful framework that would be more than enough for most of the applications that we create in our day to day; It brings functionalities to work in an organized way through MVC, a routes layer, easy creation of Rest Api, validations, form processing, filters to intercept requests and a lot more functions that will surely help you when programming your applications.

Codeigniter, once learned, will help you get to know larger frameworks like Laravel, as many of its features exist in Laravel.

Who is this book for

This book is for anyone who wants to build their first applications in CodeIgniter 4, this writing offers a step-by-step introduction to the framework, knowing the most relevant aspects of it and is focused above all on practice; It is assumed that the reader has knowledge and has developed in PHP and related technologies of the language, such as JavaScript, HTML and CSS, since, remember that to use any framework, you have to have the bases that support it, that is, your programming language.

The book is aimed at those people who want to learn something new, learn about a framework that has very little documentation, who want to improve a skill in web development, who want to grow as a developer and who want to continue scaling their path with other frameworks superior to this one. As long as you identifies at least some of the above considerations, this book is for you.

Introduction

This guide is intended to get you started with CodeIgniter 4; with this, we are going to establish two things:
We do not pretend that the person gets to know 100% Codeigniter from zero to expert, since it would be too big an objective for the scope of what is written here, if not, to know its ecosystem, what it offers us and how it works based on several examples and/or small applications with limited scope.
The reader is expected to have knowledge of JavaScript, HTML and CSS as they are necessary to be able to develop in any PHP framework, or server-side web you want to learn; on my website desarrollolibre.net I have multiple resources that may interest you to take the first steps with various technologies for free; just like on my YouTube channel, you have a lot of free, unique and quality material.

To follow this book all you need is a Windows, Linux or MacOS computer and a willingness to learn!

I strongly recommend that you follow me on at least one of my social networks; and that is, so future updates on the book, you'll be the first to know.

Considerations

Remember that when you see the $ symbol it is to indicate commands in the terminal; you do not have to type this symbol in your terminal, it is a convention that helps you know that you are executing a command.

At the end of each chapter, you have the link to the source code so you can compare it with your code.

The use of bold in paragraphs has two functions:
If we place a letter in bold it is to highlight some code such as the name of variables, tables or the like.
Highlight important parts or ideas.

To display tips we use the following layout:

important tips

For the code snippets:

```
sudo apt install mysql-server
```

When using the *** or //***

It means that we are indicating that in the code there are fragments that we presented earlier.

The official page of the framework has been:

https://codeigniter.com/

Which is essential to follow this book and to get the latest updates and details on features of the framework.

As the official page indicates, Codeigniter is a small and lightweight framework (just over a megabyte) with great features for its size and how easy it is to start in this type of framework compared to others like Laravel, which although they are more larger and complete, they are heavier and have a higher learning curve.

It is difficult to fragment the framework into isolated pieces to explain each of its features in chapters, since, in a few words, a framework is many operations working together; so, do not despair when we start to mention aspects that we have not yet explained throughout the book, since, many times when introducing a topic we are forced to mention functions of something that we have not yet dealt with in detail, and that is due to the interconnection between different functions that the framework has; If so, and you don't have clear knowledge, don't worry, we'll see it later in the book; therefore, in

these cases I recommend that you leave a note, a pending return to that page where what caused you doubt is pointed out. This mix of topics happens especially in the first chapter of the book where we see the first steps with Codeigniter 4.

As a recommendation, use Visual Studio Code as an editor, since it is an excellent editor, with many customization options, extensions, intuitive, light and that you can develop in a lot of platforms, technologies, frameworks and programming languages; so overall Visual Studio Code will be a great companion for you; but, if you prefer other editors like Sublime Text, PHPStorm or similar, you can use it without any problem.

As a web browser, I recommend Google Chrome; although it is true that when developing web technologies, it is advisable to use more than one browser; by developing specifically for the server side and not focusing on client-side development; It doesn't make much sense for this book that you use multiple browsers; that said, you can use any other browser in case Google Chrome is not to your liking.

Errata and comments

If you have any questions, recommendations or found any errors, you can let them know at the following link:

https://www.desarrollolibre.net/contacto/create

By my email:

desarrollolibre.net@gmail.com

Or by Discord on the CodeIgniter 4 channel:

https://discord.gg/sg85Zgwz96

As a recommendation, before reporting a possible problem, check the version to which you refer and add it in your comment; it is found on the second page of the book.

The book was originally written in spanish by the author; any problem with the translation, you can please communicate it through the channels presented

Map

Chapter 1: This chapter is intended to explain the environment needed to develop in CodeIgniter 4.

Chapter 2: We take the first steps with the frameworks, we will know the official website of the framework, installation forms and advantages of each one of them, we install the framework, we will know the initial aspects on how to run the framework, configure a database, create our first components, the use of migrations for table management and we will see several cases on how to work with them, we will work with the MVC and create each of these components that define it from scratch, routing and of course, finishing the chapter with the traditional CRUD for one of our entities.

Chapter 3: It is a practice for you, in which you have to create another CRUD; putting into practice everything seen in the first chapter.

Chapter 4: You will know how to use the routes, grouped routes, their options and the different types.

Chapter 5: We will know the use of the session and also of the flash session to save data and present it to the user.

Chapter 6: You will learn to manage views in a reusable way.

Chapter 7: We will see how to use a more modular scheme when defining controllers for each type of module.

Chapter 8: We will know how to work with HTML forms and apply validations from the server side in CodeIgniter, knowing the types of validations, creating them and applying validations from the controller; to then finally show the errors by the view; we will also learn how to use helper functions in our forms.

Chapter 9: We will learn more about models, some properties that we can define in them, the use of functions and common example cases.

Chapter 10: We will learn to use filters to intercept requests; as a practical case, we will create an authentication module with login for our application in which we will protect the management module or dashboard.

Chapter 11: In this chapter we build a Rest Api type CRUD that can be consumed with JSON or XML; being CRUD type operations and working mainly from the server side; it will be a process similar to the CRUD that we created in Chapter 2, but with some variations that are typical of this type of structure.

Chapter 12: In this chapter you will learn to generate test data with seeders which are ideal for when we are starting to develop.

Chapter 13: In this chapter we will learn how to handle the relational schema of the database in CodeIgniter; we will look at one-to-many and many-to-many relationships.

Chapter 14: In this chapter we are going to work with uploading files in our application, to upload images to our movie model.

Chapter 15: This chapter provides an introduction to the use of the libraries and help functions and how to use them.

Chapter 16: This chapter we will adapt Bootstrap 5 components in the application.

Chapter 17: This chapter we will create an end user module with list and detail page.

Chapter 18: This chapter we will make some changes in the Rest Api created previously to define additional methods such as upload, pagination, among others.

Chapter 19: In this chapter we will learn how to integrate the PayPal platform in CodeIgniter, in order to process payments directly from the application.

Chapter 20: We will build an application in Vue with which to consume the Rest API created previously using Oruga Ui and Tailwind 4.

Chapter 21: This chapter we will know how to integrate CodeIgniter shield to handle authentication and authorization; also, we will create a small project with which we will learn to manage groups and permissions.

Index

- **Chapter 1: Required Software and Installing the Tools**
  - Windows
  - Mac
  - Linux
  - Composer

- **Chapter 2: Knowing general aspects of Codeigniter 4**
  - Organization of a project
  - MVC: Model, View and Controller
  - Spark, the command line
  - Download and install CodeIgniter
    - Manual
    - Using Composer
  - Run the app
    - Laragon
    - From localhost, if you use Apache
    - Optional, if you use Apache
  - Getting started with CodeIgniter 4
    - Enable developer mode
    - Configure the database
    - Routes
    - Arguments in routes:
    - Working with routes
    - Resource type routes
    - Automatic routes
    - Routes and controllers
    - Controllers and views
    - Work with the database
    - Models
    - Reuse views
    - Conclusion

- **Chapter 3: CRUD for categories**
  - Controller
  - Views
  - Models
  - Routes

- **Chapter 4: Routes and redirects**
  - Group routes
  - Redirects
    - Case study
  - Named routes
  - Redirects and named routes
    - Types of redirects
  - Limit resource paths or presenter

- **Chapter 5: Session and flash messages**
  - Flash messages
    - Case study
  - Messages session

- **Chapter 6: Views and layouts**
  - Views and layouts
    - Case study

- **Chapter 7: Controllers**
  - Reorder controllers
    - Case study
  - Private methods

- **Chapter 8: Forms**
  - Validations
    - Case study
  - Previous value (old)
    - Case study

- **Chapter 9: Models**
  - Model Properties
  - Changes to return types

        - Case study

- **Chapter 10: Filters**
    - Create a filter
        - Case study

- **Chapter 11: Rest Api CRUD**
    - Case study
        - Postman
    - Return json in controllers

- **Chapter 12: Seeders**
    - Generate a seeder:
        - Case study
    - Nested seeders

- **Chapter 13: Relationships**
    - One-to-many relationships
        - Case study
    - Many-to-many relationships
        - Case study
    - Nested relations
        - Case study

- **Chapter 14: File Upload**
    - Important functions in uploading files
    - Access the image using a controller
    - CRUD
    - Delete image
    - Download image
    - Optional: Delete image, variant

- **Chapter 15: Libraries and helper functions**
    - Case study
        - Helper: filesystem
        - List Pagination
    - Exception Handling

- **Chapter 16: Bootstrap 5**

- Instalation
  - Bootstrap components

- **Chapter 17: Web module**
  - Paginated list
  - Detail page
  - Filters for the list
    - Case study

- **Chapter 18: Rest api relationships custom methods**
  - Paginate
  - Search with filters
  - List of movies by category
  - Movies by tag
  - Create and update movie: Foreign relations
  - Detail of a movie
  - Upload files
  - CRUD for tags

- **Chapter 19: PayPal**
  - Create application in CodeIgniter and base structure
  - PayPal Developers Site
  - Integrate with CodeIgniter 4
    - JavaScript: Generate the order
    - Create PayPal Button
    - CodeIgniter/PHP: Generate access token
    - CodeIgniter/PHP: Process (Capture) the order
  - Final details

- **Chapter 20: Consuming the Rest Api from a Vue app**
  - Prepare the environment
  - Folder structure
  - Consuming a resource from the Rest Api
    - CORS
    - then(), resolve promise functions
    - Axios replacing fetch
  - Components for the API
    - List
    - v-for

        - Reactivity in Vue
        - Async and await
        - Functions
    - Set up project in Vue 3 with Oruga UI
    - Install Material Design Icons
    - Routing with Vue Router
        - Install
        - Define routes
        - Parent component for component rendering
        - Set the routes
        - Component to create and edit post
        - Create navigation links
        - Tailwind.css 4 in Vue project with Oruga UI
        - Grouping routes
        - Navbar: Links
        - Redirect on 404
    - CORS CodeIgniter 4 the right way

- **Chapter 21: CodeIgniter Shield**
    - Install and configure
    - Shield pages
    - Customize Shield
        - Redirects
        - Views
    - Authentication
    - Authorization
        - Groups
        - Permissions
        - Assign permissions to groups
    - Groups and Permissions
        - Test users
        - User list
        - User detail: Groups and permissions
        - Groups and permissions in practice
        - Classify permissions by groups
        - Assign/remove permissions with one click
        - Assign/remove groups with one click
        - Iterate array
    - Protect controllers using groups and permissions:

- Users: SuperAdmin and Administrator
- Dashboard module with authentication required
- Example controllers

Preview

This is a Preview of the book content and it is completely FREE, you can get the full book from: **desarrollolibre.net/libros**.

Chapter 1: Required Software and Installing the Tools

Being CodeIgniter a PHP framework, we are going to need the technologies that support this tool, in addition to others that we can use; specifically:

1. Apache
2. PHP
3. MySql

The usual trilogy, which is used as the basis for working on any web application with PHP and, of course, frameworks such as CodeIgniter, also use it.

So, we have several paths, since, in each operating system, we have several options to install these tools; as long as you have the mentioned technologies in your pc and as Codeigniter is a framework with relatively few requirements, you should not have many problems:

https://codeigniter.com/user_guide/intro/requirements.html

Remember that the objective of this book is not to install the necessary tools or to deal with topics that are not directly the framework; as a developer who wants to learn to use a framework like CodeIgniter 4, you should know the bases that support it (Apache, PHP, MySQL, HTML, CSS and a bit of JavaScript) and with this, have them installed on your computer. These are the basics of CodeIgniter, just like any other PHP framework; however, some recommendations and summaries of the available or most used configurations will be given.

Windows

In the case of Windows, the solutions "next, next next..." can be WAMP and XAMPP among other similar ones; although, if you don't have any environment installed, Laragon is an excellent solution as it is the most complete, scalable and easy to work with:

https://laragon.org/

In its "Full" version

As you can see on the page, the purposes of Laragon is to allow the development of Laravel applications, since CodeIgniter is a simpler framework with fewer requirements than Laravel, you can use this environment perfectly.

...

Chapter 2: Knowing general aspects of Codeigniter 4

CodeIgniter is a small framework (barely 1.2 Mb) but robust and with great features, it is easy to learn and has an organization similar to that offered by Laravel but substantially simpler; it is a framework that by having the environment correctly established (LAMP) requires practically zero configuration except for the database; in this book we will use MySQL or its free version MariaDB.

CodeIgniter offers us a simple and easy to follow official documentation, although sometimes it feels a bit sparse and generic.

For this chapter, I recommend that you reread it several times as you go through the book; since, we talked about how the framework is formed and what its main elements are for; elements that are the ones that we will address in the book; therefore, if you have not worked with similar frameworks, it is likely that you will get lost in the definition of these components. The purpose of this chapter is to get a basic understanding of each of these components, but don't worry if you get a little lost; I recommend that you write down those definitions that you have not understood and wait for their subsequent explanation in detail in the following chapters.

Organization of a project

A project in CodeIgniter 4 has the following organization which we will explain in a general way the purpose of each folder and file:

app folder

Inside the app folder, it is where all the code of the application resides and where we are going to spend most of the time; this folder comes with a default file and folder structure that works well for many applications. The following folders make up the content:

1. /Config Stores the configuration files, they are varied, for example, from the database, form validations, emails, filters, among others.
2. /Controllers Controllers determine the flow of the program; they are one of the three elements of our MVC and it is the one that communicates the view with the model according to the rules that we define.
3. /Database Stores database migrations and seed files (seeds) we talked about in another chapter.
4. /Filters Stores filter type classes that can be executed before and after handlers you specify.

5. /Helpers Helper files that store collections of independent functions that we can easily reuse throughout the application.
6. /Language Here we define files to handle the languages of the application.
7. /Libraries Useful classes that don't fit into another category; they are more specific schemes than helpers, in which we can use classes to create functions that we can reuse.
8. /Models Models are nothing more than classes that have a direct relationship between the tables with the classes and the columns with the properties of the classes; here we define formats, structures and in essence the communication with the database; this is one of the three elements that correspond to our MVC.
9. /ThirdParty Third party libraries that can be used in applications.
10. /Views Views make up the HTML that is displayed to the client; and this is another layer of our MVC; it is assigned from the controller.

system folder

This folder stores the files that make up the framework itself; all the functionalities of our framework reside here, therefore the files in the system folder should never be modified. Instead of this, you can add your functionalities via helpers (files with helper functions) or create new classes to provide the desired functionality.

All files in this folder are in the "CodeIgniter" namespace.

public folder

The public folder contains the browser-accessible part of your web application; it is the only folder that can be accessed, and this is for security reasons; this is an important change that we have with version 3 and follows the same steps that other frameworks such as Laravel have used, which also have this same structure.

Placing a single public accessible folder prevents direct access to its source code. Contains the file .htaccess main, index.php, and any app resources you add, such as CSS, JavaScript, and images.

This folder is meant to be the "web root" of your site, and your web server would be configured to point to it.

writable folder

This folder contains all the folders and files that might need to be written to in the course of an application's life. This includes files and folders that the framework itself generates, as well as folders for storing cache files, logs, and any payloads a user might submit. This allows you to keep your other parent folders unwritable as an added security measure to prevent unwanted code from being loaded and executed.

tests folder

This folder is set to contain your test files.

You do not need to transfer this folder to your production servers.

docs folder

This folder may or may not be there; depends on how you get your copy of the framework, it contains a local copy of the CodeIgniter 4 user guide so you know what to do; it is only part of the CodeIgniter website (documentation section) but in a pdf version.

MVC: Model, View and Controller

The MVC pattern is one of the many software design patterns that exist today, one of the most widely used and from which numerous variants emerged such as MTV, MVP, MVA, MVVM and many more; as you can imagine, it is a scalable, adaptable pattern and if it has so many variants it means that it has great potential; frameworks like Laravel and of course CodeIgniter use it to organize our code, make it easy to maintain, adapt changes, maintain it and other aspects related to the guidelines that we must follow to develop any type of application.

CodeIgniter uses the Model, View, Controller (MVC) pattern; this keeps each layer as separate parts, but works together as a whole:

1. Models manage the application's data and help enforce any special business rules that the application might need.
2. Views are simple files, with little or no logic, that display information to the user; the views are composed of HTML for the static part and PHP for the dynamic part; aside from CSS and JavaScript.
3. Controllers act like a sticky code, sorting data between the view (or the user viewing it) and the data store, i.e. the model; this component is where we usually spend more time (along with the view) since we have to organize everything that we are going to see in the view, apply validations and other rules according to the logic that we program in our application.

In their most basic form, controllers and models are simply classes that have a specific job that we noted above; but there are always processes that we can reuse and we do this by defining other classes such as services, help files, etc; that can be part of the core, that is, part of the framework, or can be defined by you, as part of the application you are creating or by third parties, installed or copied into the framework.

In Codeigniter we have folders designated for a particular task; as we explained above; we as developers are going to spend most of our time in the /app call

If you want more information about it:

https://codeigniter.com/user_guide/concepts/mvc.html

Spark, the command line

CodeIgniter has a very easy to use command line interface (CLI) known as spark; spark is nothing more than a file that is located in the root of our project with the name of "spark" and allows us to execute a series of pre-established commands; of course, we can extend the commands offered by the framework itself by programming our own commands; but this is another topic; in short, we can divide the commands that we can use into three groups:

Commands to generate files:

1. Create migrations, we will talk about this in another chapter, but they are nothing more than files that save the structure of a table that the framework will map to the database.
2. Generate seeds for test data.

Commands to handle processes:

1. Set up a development server, in case you don't want to use Apache or other servers supported by the framework.
2. Run or return migrations.
3. Clear cache.
4. Manage the database.
5. Run the migrations and seeds.

Commands to get project information:

1. Command list.
2. List of project routes.

Among other commands you can see running at the project level:

```
$ php spark
```

Most used commands

So you have a list of the commands; I recommend that you copy and read it a few times a day and become familiar with these commands that are the most used when developing in CodeIgniter:

1. php spark serve: To start a development server.
2. php spark make:migration: To generate a migration file.
3. php spark migrate: To generate a migration and related as the rollback to return the migrations.
4. php spark routes: To see the routes of the application.

Do not worry if you do not understand the purpose of these functions, later we will see in detail the operation of each of these elements.

Download and install CodeIgniter

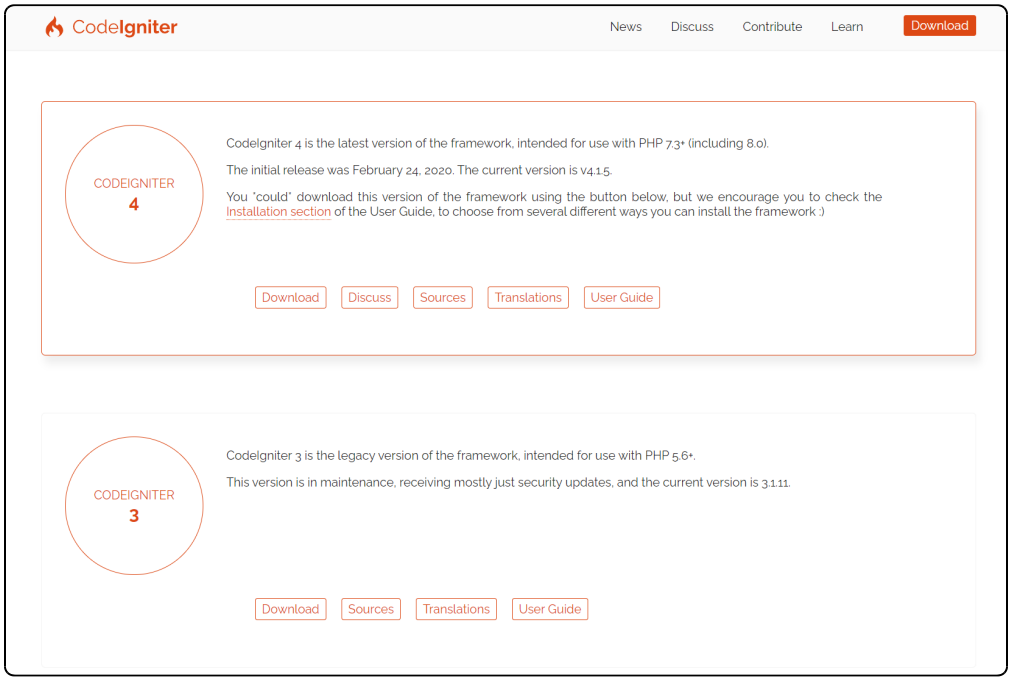
Working with CodeIgniter 4, we have two ways to install the same:

Manual

As with its predecessor, CodeIgniter 3, we can use the manual installation of the framework, which consists of downloading a compressed file from the official page:

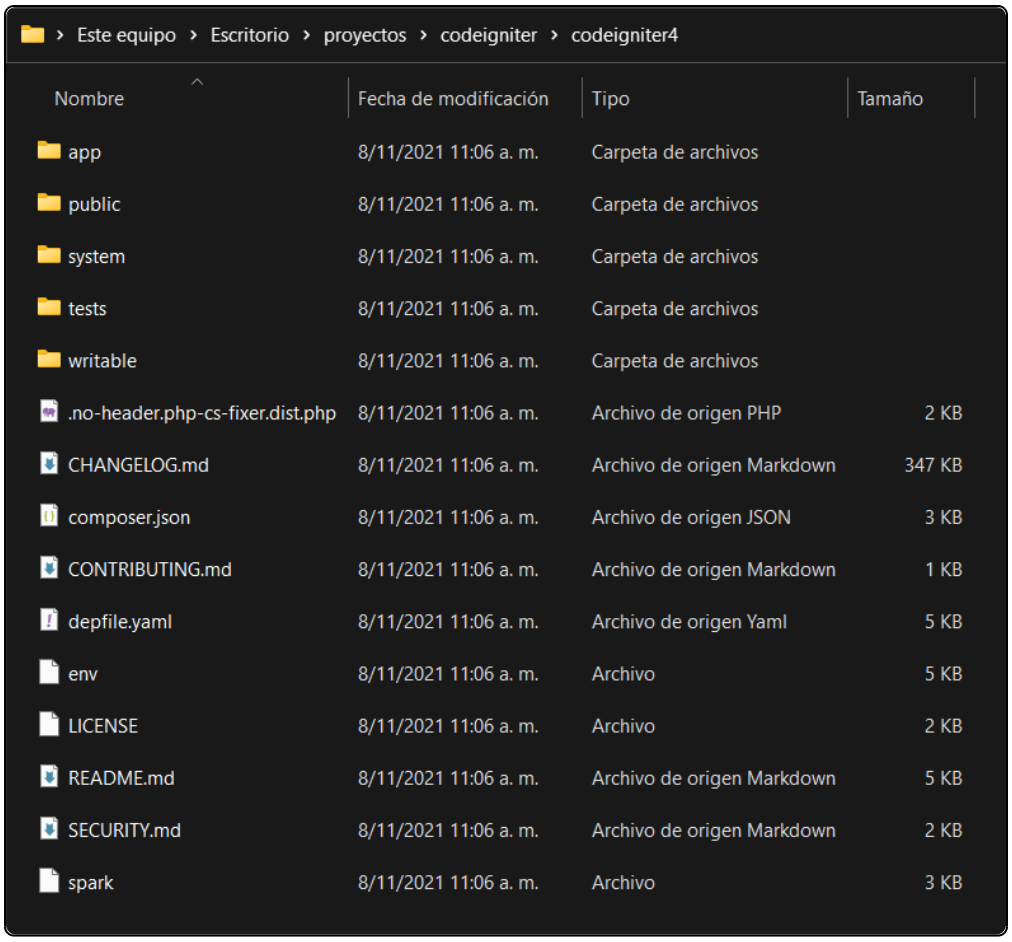
https://codeigniter.com/

In the download section:



Click where it says "Download" for CodeIgniter 4; this will start the download process of the framework which is in a compressed .zip. We unzip it, give the folder a better name and place it anywhere, in this book it will be in a folder on the desktop:

C:\Users\acy\Desktop\projects\codeigniter\codeigniter4



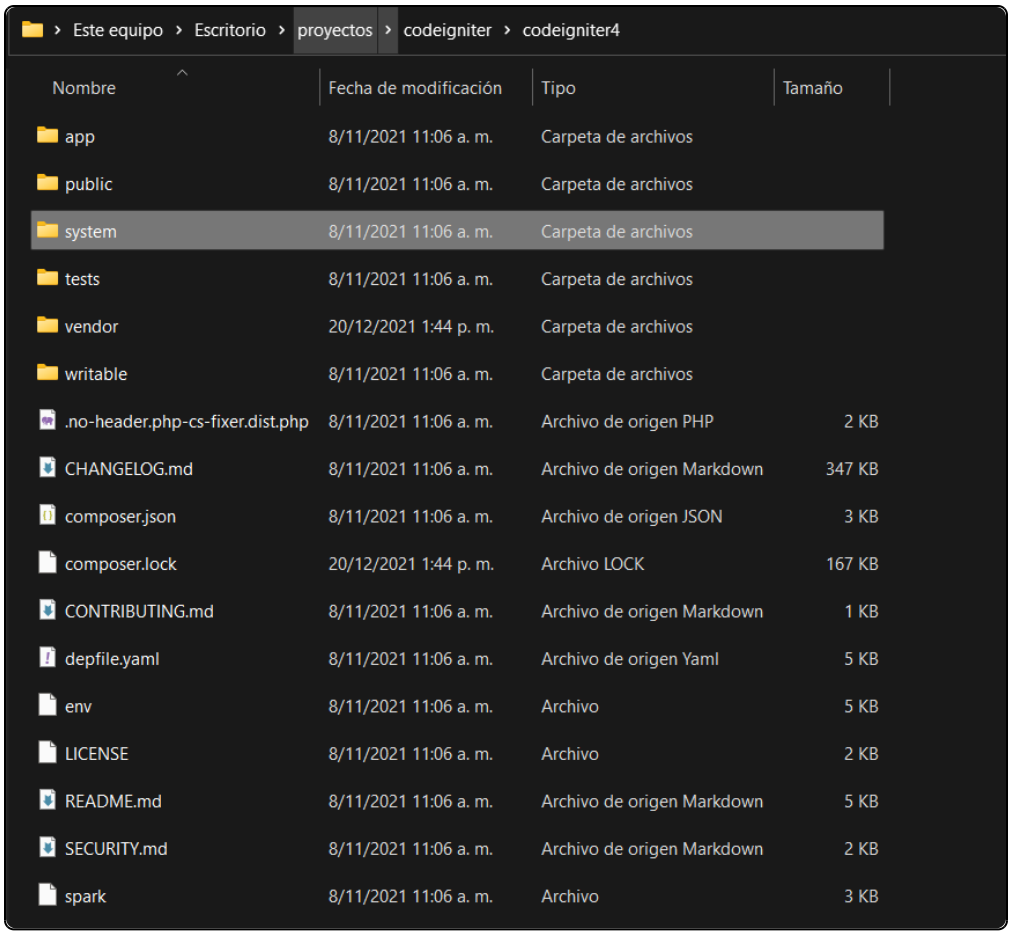
The project will be called "codeigniter4" in the book, and that's the structure we have.

This option is ideal for doing some testing with the framework; but in the end, if we get used to working with Composer, we can both create the project and update it and perform the various tasks that Composer allows us. With Composer, we have a more direct, fewer-step way to do the tasks outlined above.

As you can see in the previous image, in this project we already have Composer enabled to be able to work; however, by default the project does not work with Composer...

I know this may seem confusing, but when you compare it with the project that we will create in the next section with Composer, you will see that a folder is missing... for this option, the manual installation option, you will see that there is a folder called system, which is where find the core of our framework; even if we run Composer commands like:

```
$ composer update
```



vendor folder is generated, with all the dependencies, our project will still point to the system folder; therefore, we would have to make extra adaptations to our project so that it works through Composer and with this, all the advantages that this manager offers us; but it would not make any sense, since, if we want to use Composer, the best we can do is create the project from scratch with Composer.

In conclusion, this option is recommended to do some tests with the framework and nothing else, since we do not have a direct way to work with Composer and what this implies, impossibility -or in the best of cases- complications to update and install PHP dependencies. Since, if we want to extend the framework with PHP libraries or update the framework, we are greatly limited and both limitations can be easily solved if we use Composer.

Composer can be used both to install dependencies and update them, in our case, the latter includes the framework as such.

Using Composer

This is the preferred option for working with CodeIgniter 4 just like any modern PHP framework; being the most direct and the one that offers the best scope due to the potential of being able to use Composer in the development of applications. We are going to make use of Composer that we installed in the previous section to create a project in CodeIgniter 4, we have the following command that we take from the official page:

```
$ composer create-project codeigniter4/appstarter ProjectName
```

Where ProjectName is the name of our project, in our case it will be movies:

```
$ composer create-project codeigniter4/appstarter movies
```

Run the app

We have several ways to work with CodeIgniter 4, the most common is to run it directly from the server in Apache:

Laragon

If you use Windows and Laragon, you will have noticed that Laragon automatically generates a clean url through the virtual host to access the application:

```
<VirtualHost *:80>
    DocumentRoot "C:/laragon/www/peliculas/public"
    ServerName peliculas.test
    ServerAlias *.peliculas.test
    <Directory "C:/laragon/www/peliculas/public">
        AllowOverride All
        Require all granted
    </Directory>
</VirtualHost>
```

This is a virtual host that Laragon autogenerates for us and since they are autogenerated they are managed internally but you can see which ones you have created from the Laragon application at:

Menu - Apache - site enabled

Of course, if you use Apache on another operating system like Linux or MacOS, or simply use another server on Windows, you can create your virtual host.

Another option, which is the one we are going to use because it is the most direct, is to directly use the spark server that we have in CodeIgniter 4 when installing in the framework:

```
$ php spark serve
```

We have to execute this command from the root of the project; in other words, if our project is called as movies; then we have to place ourselves from our terminal in:

C:\laragon\www\peliculas

important to note that if you're using spark you don't need Apache, and for purposes of following this book, you'll only need PHP and MySQL.

Also remember to restart your server so Laragon will detect the new project and create the virtual host.

From localhost, if you use Apache

The last one would be to access the entire route from Apache:

http://localhost/movies/public/

Which is not highly recommended, since it can bring you complications with the handling of routes and referencing the base URL and it is a bit tedious to work with this type of URLs.

Optional, if you use Apache

httpd.conf file. For more information, you can see this link:

https://www.desarrollolibre.net/blog/apache/que-son-y-como-emploar-los-virtualhost-en-apache

Getting started with CodeIgniter 4

CodeIgniter as a framework that contains many functionalities that we must know in order to start creating our applications that go through the MVC, placing one more layer that has been that of the routes, which is nothing more than the mapping between a URL to a controller, such as the basic configurations that in this case would be of the database and enable the developer mode to be able to see the errors that are occurring.

This chapter is fundamental, to start creating our first applications; in this chapter it is assumed that you already know in a basic way what each folder and file in the framework is for; as well as the creation of applications in CodeIgniter 4 using Composer and running them. All of this was covered in the previous chapter.

Enable developer mode

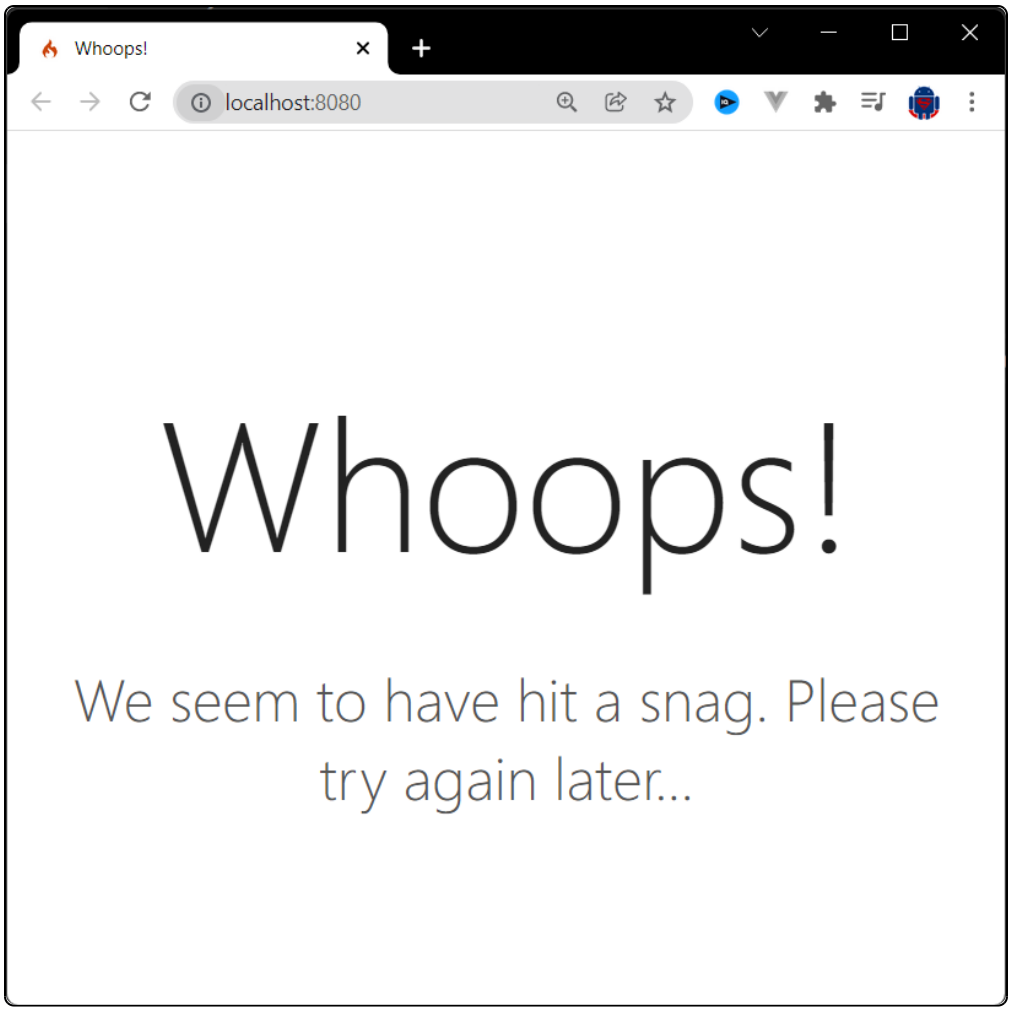
Let's start with the most important point in this chapter, or at least the most basic, that of being able to see the errors that are going to happen when we develop the application. Before this, to understand the importance of enabling this mode, we are going to cause a syntax error in our application to see what happens; in the file:

App\Controllers\Home.php

What we have:

```
<?php
namespace App\Controllers;
class Home extends BaseController
{
    public function index()
    {
        return view('welcome_message');
    }
}
```

Let's remove a ";" for the view('welcome_message') and when going to our app:



You will see that it tells us that an error occurred, but not where; to see what is happening, we would have to consult a log file that is in:

\movies\writable\logs

See the last log, and detail the first lines to know what is happening:

CRITICAL - 2022-01-15 15:07:15 --> syntax error, unexpected '}', expecting ';'

In this case, the error that we intentionally caused; obviously, consulting this file every time we have an error is a little headache, ideally the error should appear on the screen when we are in development mode.

Seeing the errors in the log file is ideal for managing the app when we are in a production environment, since the details of the errors are left to us, the developers, and with this we avoid giving details about the structure of our application to a user who could take advantage of the error detail to exploit a vulnerability in the application.

To enable developer mode, simply rename the file called "env" found at the root of our project to ".env", that is, add a dot in front:

.env

Go into that file, and uncomment:

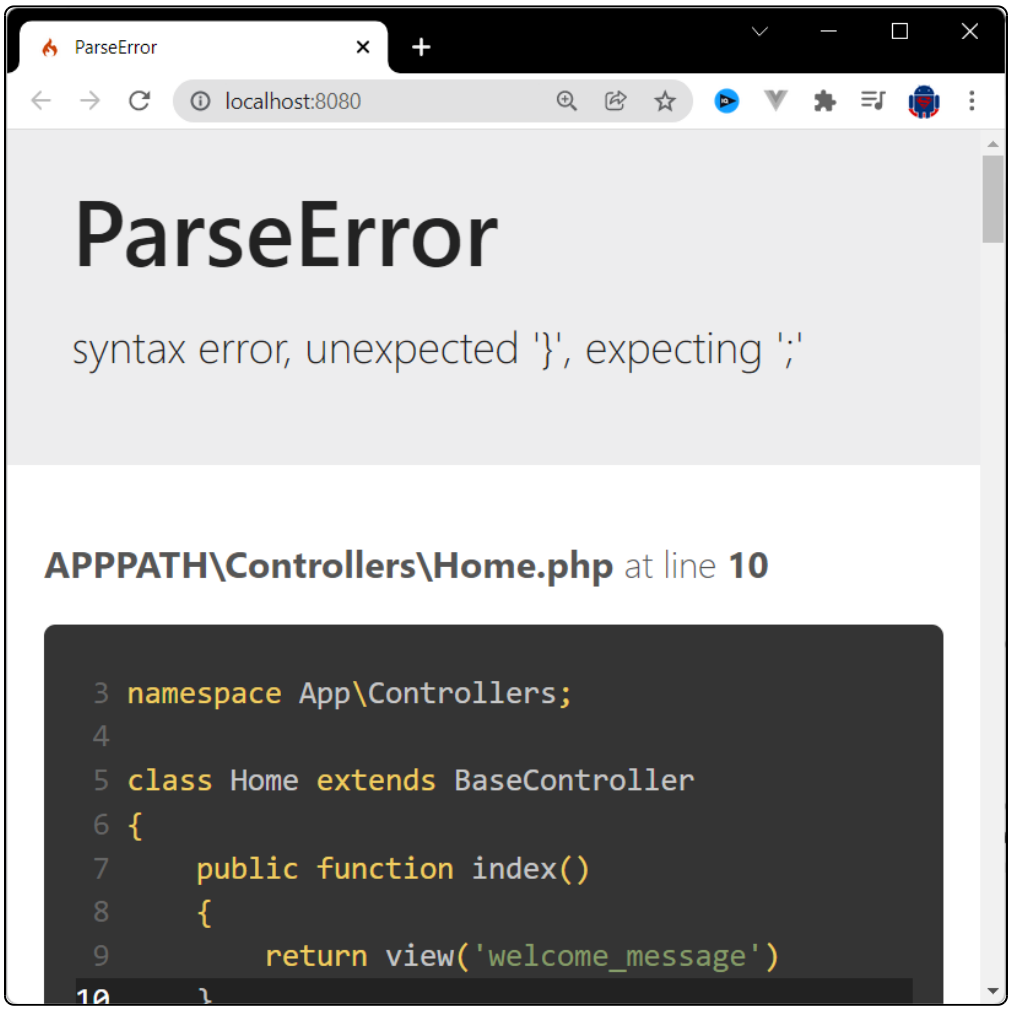
```
# CI_ENVIRONMENT = production
```

for:

```
CI_ENVIRONMENT = development
```

And remember to change "production" to "development"

With this, if we go to the page with the error, we will see the detail on the screen:



.env files are used to create project environment variables; standard configurations such as the database, base url, among others of the project and customized by you (we can create custom variables that do not exist in the project, for example, for PayPal, Stripe, etc. credentials).

However, you should keep in mind that standard configurations like the ones mentioned above, which have their equivalent in configuration files, when using the production application, must be removed from the .env to use the corresponding configuration file; in case it is necessary for your project, the .env file can also exist in the production environment (or other environments determined for your project).

In the end, environment variables are dynamic variables that arise from the behavior of the system and can exist in different environments.

Remember to revert the error we placed intentionally.

Configure the database

As we indicated initially, we are going to use a MySQL or MariaDB database in our project; for this, the first thing we have to do is create a database in MySQL/MariaDB (we will use the term MySQL or MariaDB equivalently throughout the book).

...