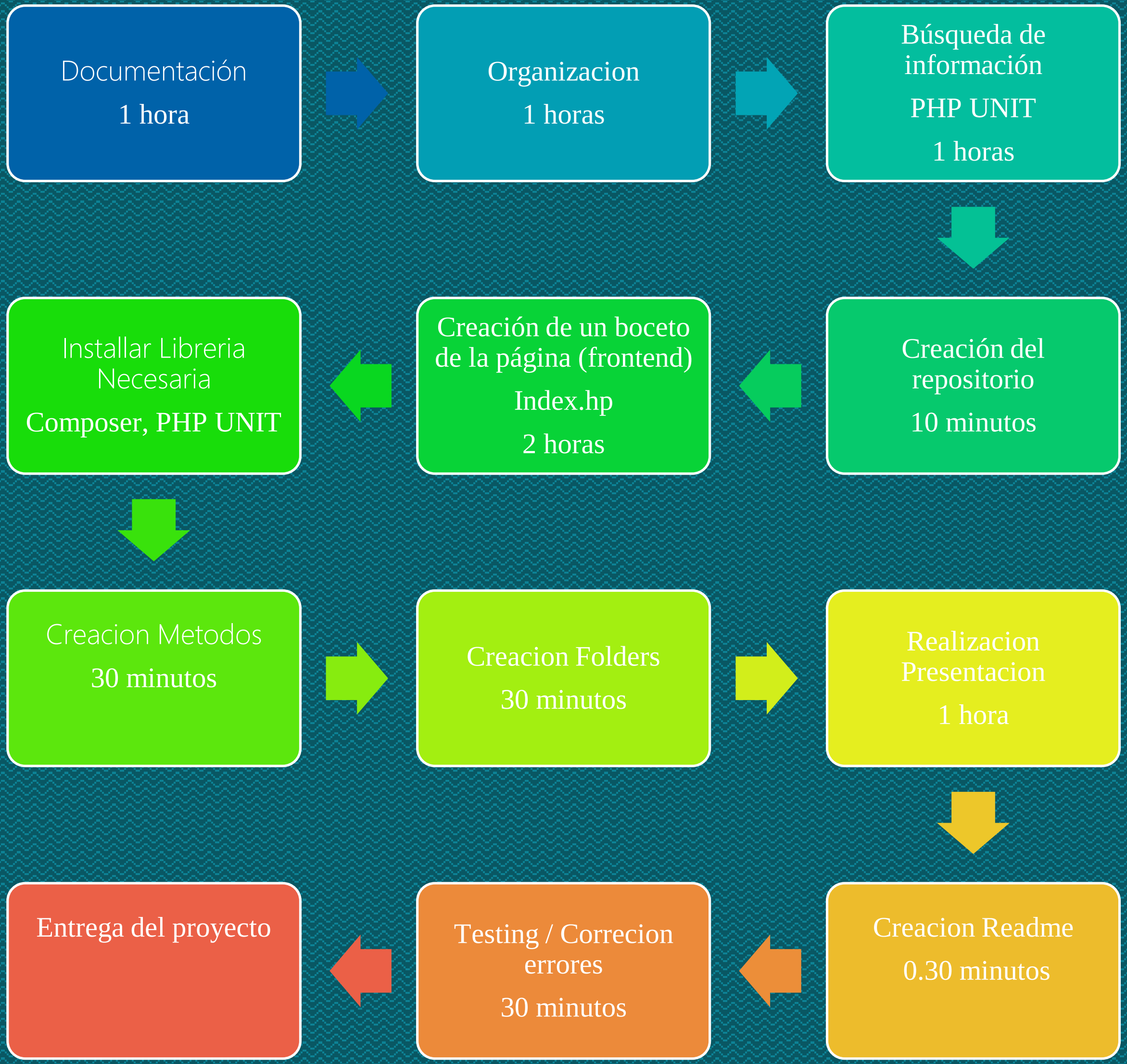
PHP UNIT

Robert Salazar Arias

Organización de las tareas



Documentación

1 hora

Organizacion

1 horas

Búsqueda de información

PHP UNIT

1 horas

Creación del repositorio

10 minutos

Creación de un boceto de la página (frontend)

Index.hp

2 horas

Installar Libreria Necesaria

Composer, PHP UNIT

Creacion Metodos

30 minutos

Creacion Folders

30 minutos

Realizacion Presentacion

1 hora

Creacion Readme

0.30 minutos

Testing / Correcion errores

30 minutos

Entrega del proyecto

Conocimientos

It's a tool that detects errors and won't let you move forward until you correct it.
It is a test-driven development methodology
It's a bug-driven development
To use php unit we have to use this route
use PHPUnit\Framework\TestCase;

Dificultades

In the distribution of the folders that the correct rendering is done

Caracteristicas

Class tests are in the Test class
Tests are public methods with a name for test
They can be documented using annotation @test
Within the test methods you enter the assert methods

Funcionalidades

Tests for a Class class go within a ClassTest class
ClassTest hereda PHPUnit\Framework\TestCase
Tests are public methods and take as their name
Then we'll perform two functions
AssertNotNull: Verifies that the reference is not null.
AssertEquals: Verifies that the two references passed as an argument are the same.

Uso del fichero *phpunit.xml*

It is used to define the configuration that will run PHPUnit tests and is loaded by default when we run PHPUnit without any other parameters.

The first parameter tells PHPUnit that we want to generate a report on the coverage of our suites. The second indicates in which destination folder we want that report to be exported in html format.

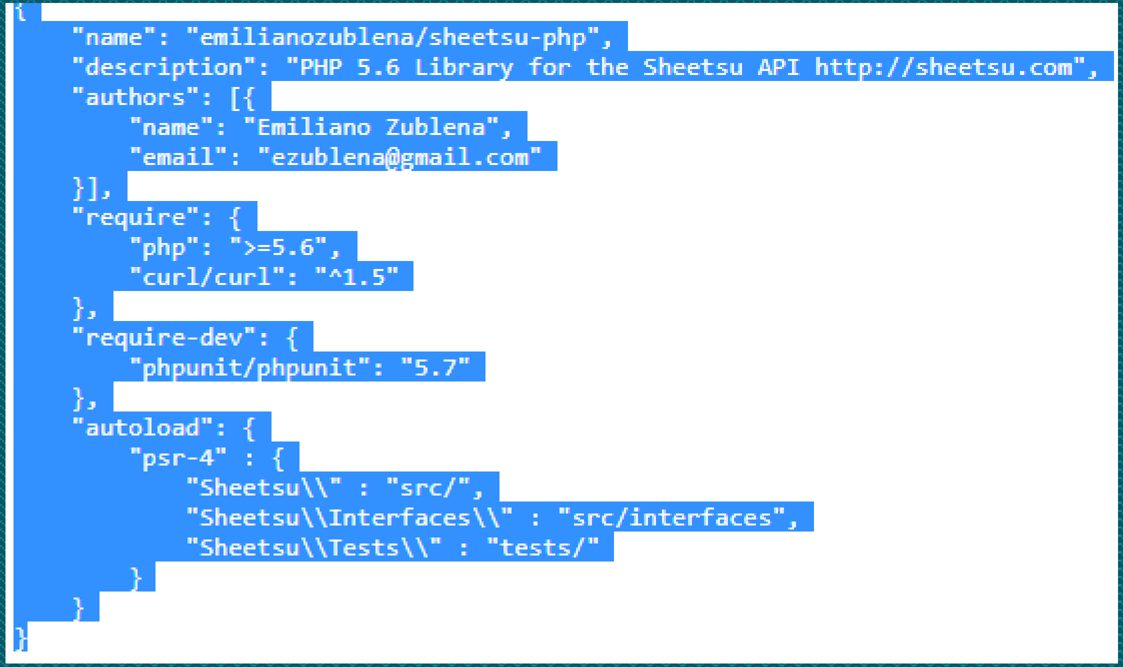
The configuration of PHPUnit is extensive and here you can have a complete reference but the two important points here are 2, the first one we mentioned is where we indicate our test suite and its location in the directory:

to xml

<phpunit colors="true">

Convencion de rutas

As a convention, PHPUnit suggests that all our unit tests be separated from the code, it is advisable to place them all inside a tests folder at the root of our project.
In addition to creating the folder, we have to remember to tell Composer the namespaces that we choose, so that when we generate the autoloaders for the classes everything works in order.



```
{
    "name": "emilianozublena/sheetsu-php",
    "description": "PHP 5.6 Library for the Sheetsu API http://sheetsu.com",
    "authors": [{
        "name": "Emiliano Zublena",
        "email": "ezublena@gmail.com"
    }],
    "require": {
        "php": ">=5.6",
        "curl/curl": "^1.5"
    },
    "require-dev": {
        "phpunit/phpunit": "5.7"
    },
    "autoload": {
        "psr-4" : {
            "Sheetsu\\" : "src/",
            "Sheetsu\\Interfaces\\" : "src/interfaces",
            "Sheetsu\\Tests\\" : "tests/"
        }
    }
}
```

Aserciones

The most commonly used assertions

Valores numéricos

assertEquals –If the two variables $expected and $actual are equal

assertSame – if the two variables do not have the same type and value text alores

assertIsString –If you check for type string

Working with array

assertIsArraySubset – used to verify the identity of objects within the array

assertIsArray – identifies if type Array

assertClassNotHasStaticAttribute –If the class attribute does not exist

Check directory

assertDirectoryNotExists –If the specified folder exists

Json

assertJsonStringEqualsJsonFile –If the value does not match the value $expectedFile