PUESTA EN PRODUCCIÓN

Tarea 1

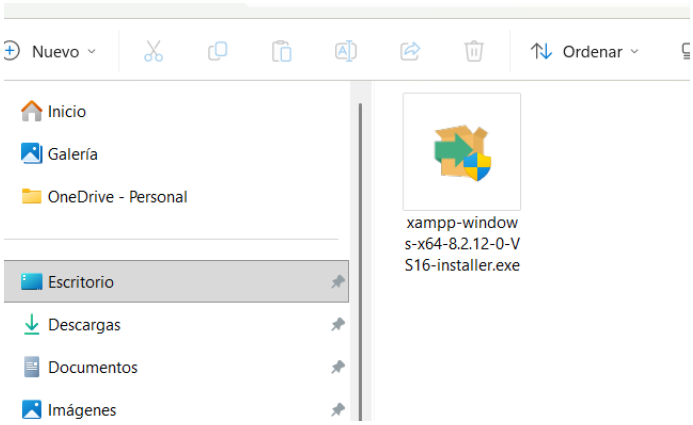
ESTHER CARRILLO GÁLVEZ

Contenido

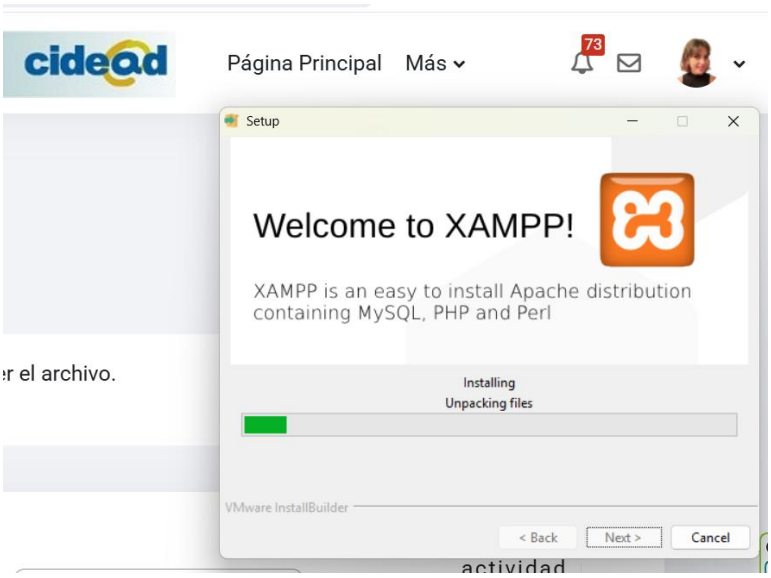
Apartado 1: Instalación de herramientas utilizadas

1.1. Instalación de entorno XAMPP

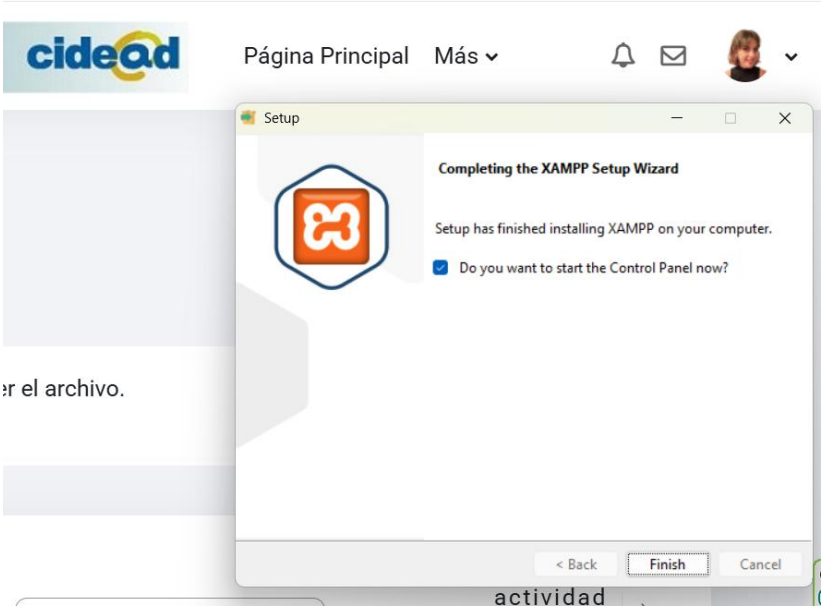
El primer paso del proceso es instalar el entorno XAMPP desde el sitio oficial https://www.apachefriends.org/es/index.html. En cada uno de los pasos anteriores hasta el último, se ha seleccionado la configuración por defecto pulsando el botón *Next*, así hasta llegar a *Finish*.



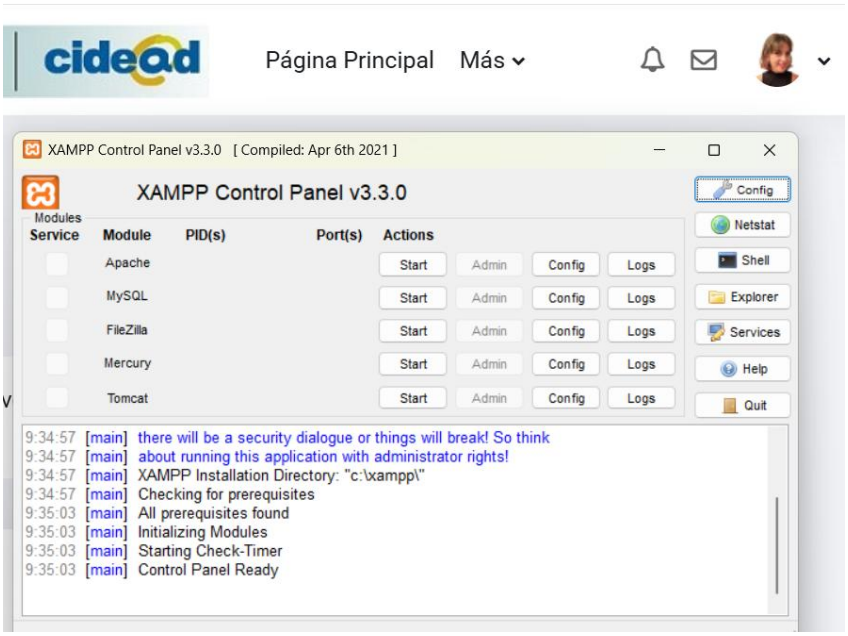
Instalación en progreso. Este es el penúltimo paso antes de la finalización.



Se deja seleccionado el botón de ejecutar el Panel de control y se finaliza.

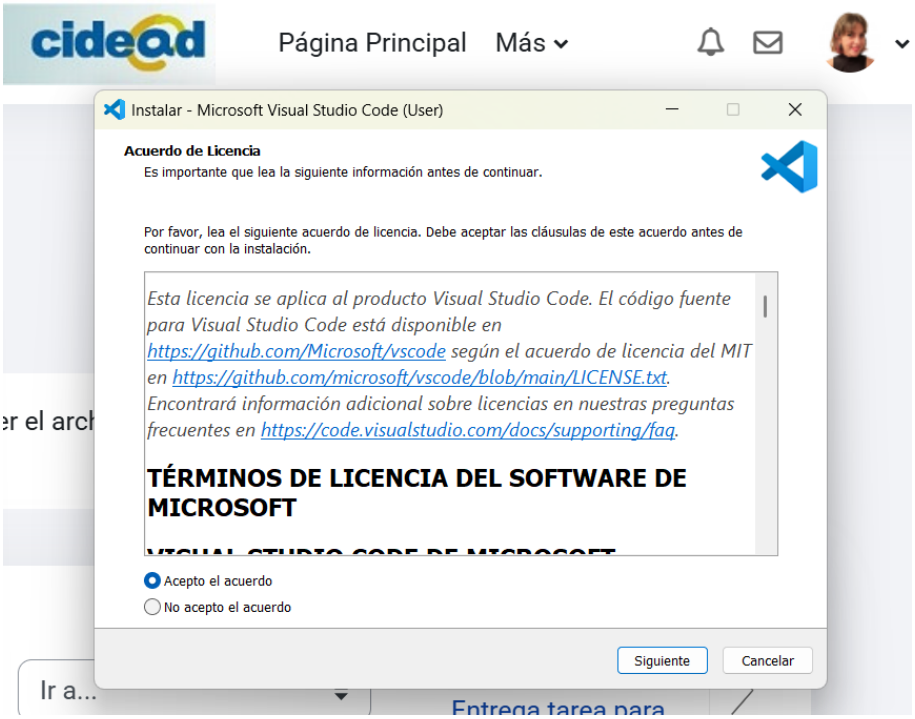


Una vez finalizado, se abrirá este panel que es la herramienta central para gestionar tu entorno de servidor local. La columna Module muestra los servicios que XAMPP instala. Es más importante y el que se va a usar para esta práctica es el primero.

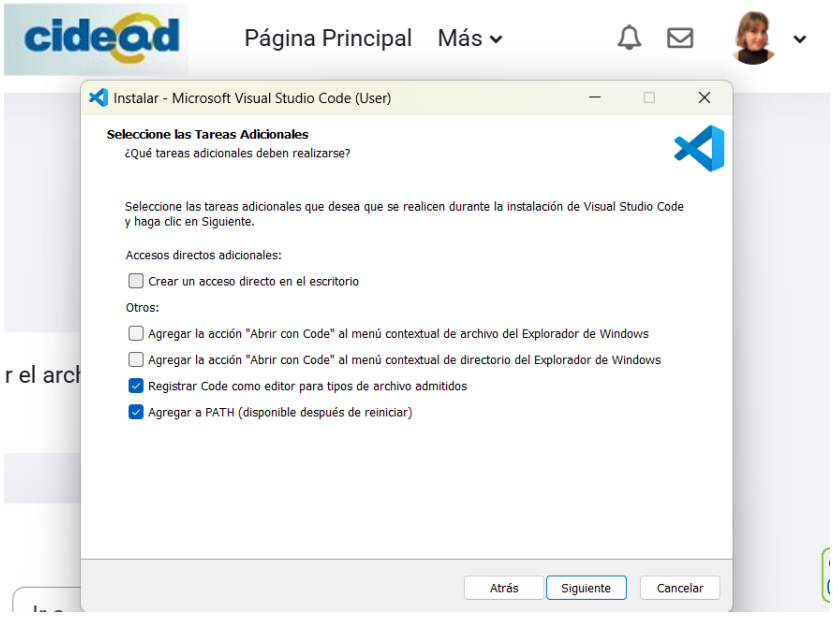


1.2. Instalación Visual Studio Code

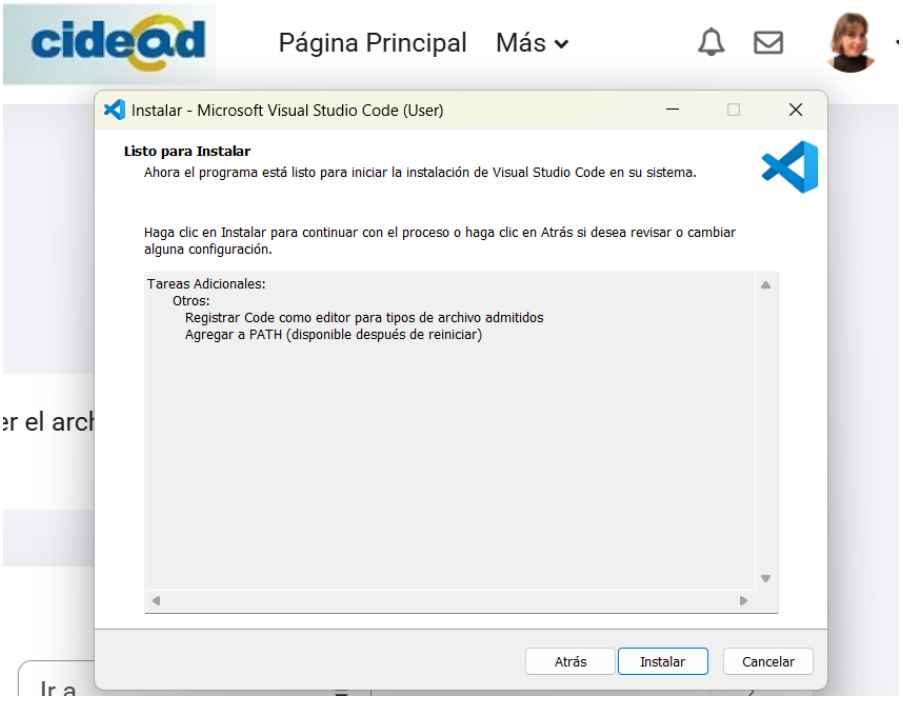
A continuación, se instala Visual Studio Code desde Visual Studio Code - The open source AI code editor. En primer lugar, se aceptan los términos de licencia del software.



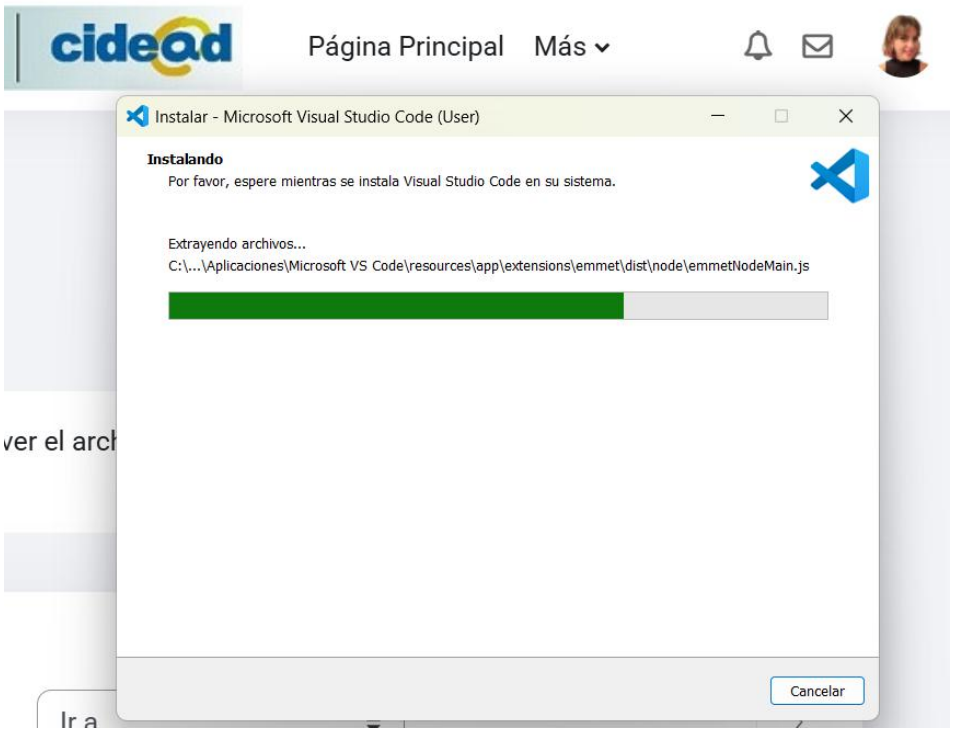
Se deben seleccionar las tareas adicionales. En este caso, se dejan las que vienen por defecto.



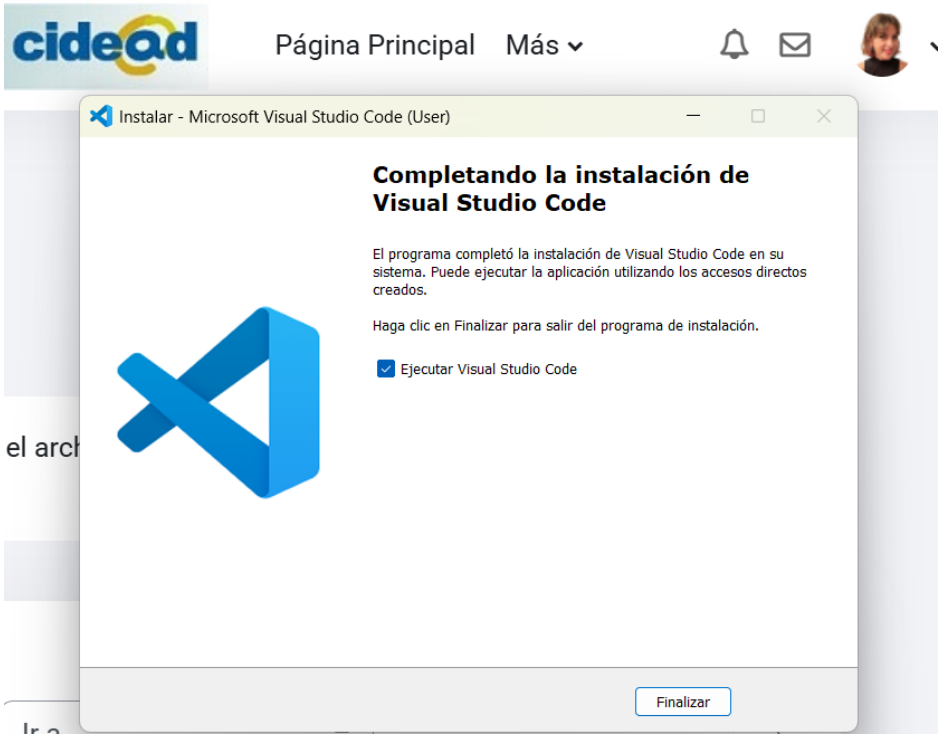
En el paso “Listo para instalar”, se muestra el resumen de la configuración.



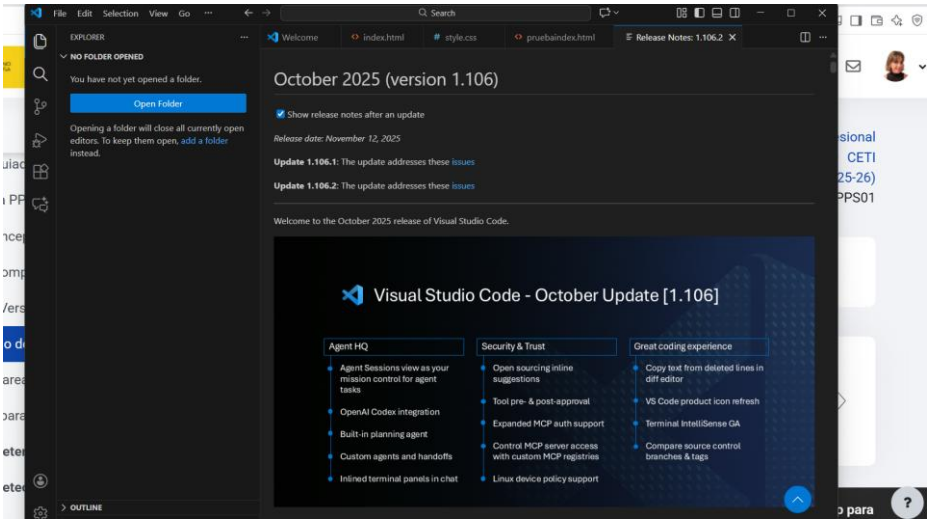
Instalación en progreso. Este es el penúltimo paso antes de la finalización.



Estado de la instalación completado. Se deja seleccionado “ejecutar Visual Studio Code”.



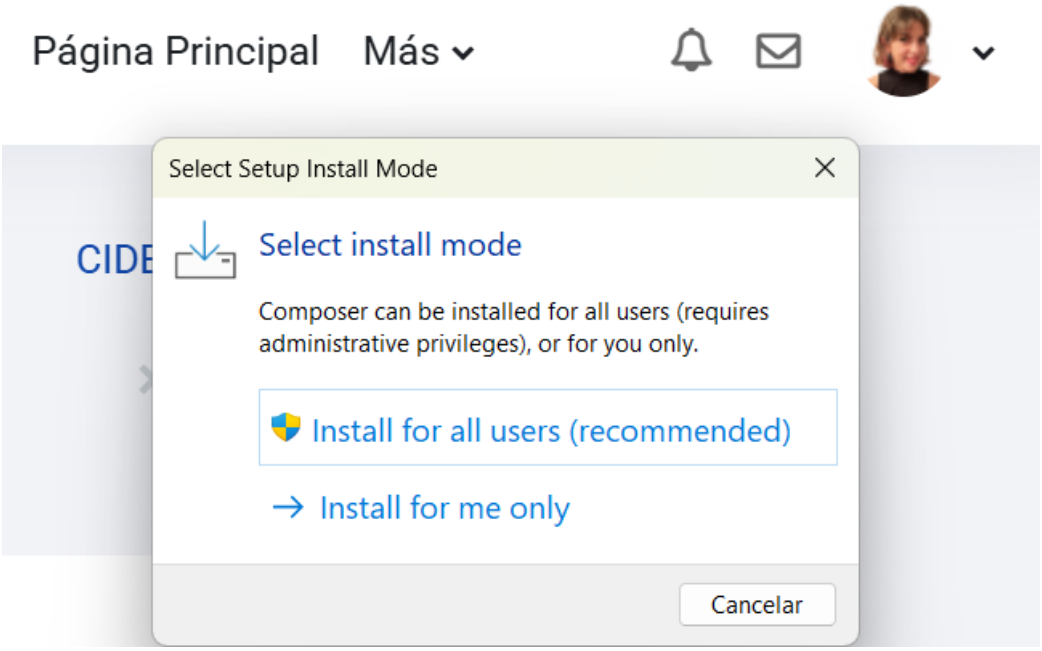
Finalmente, se ejecutará Visual Studio.



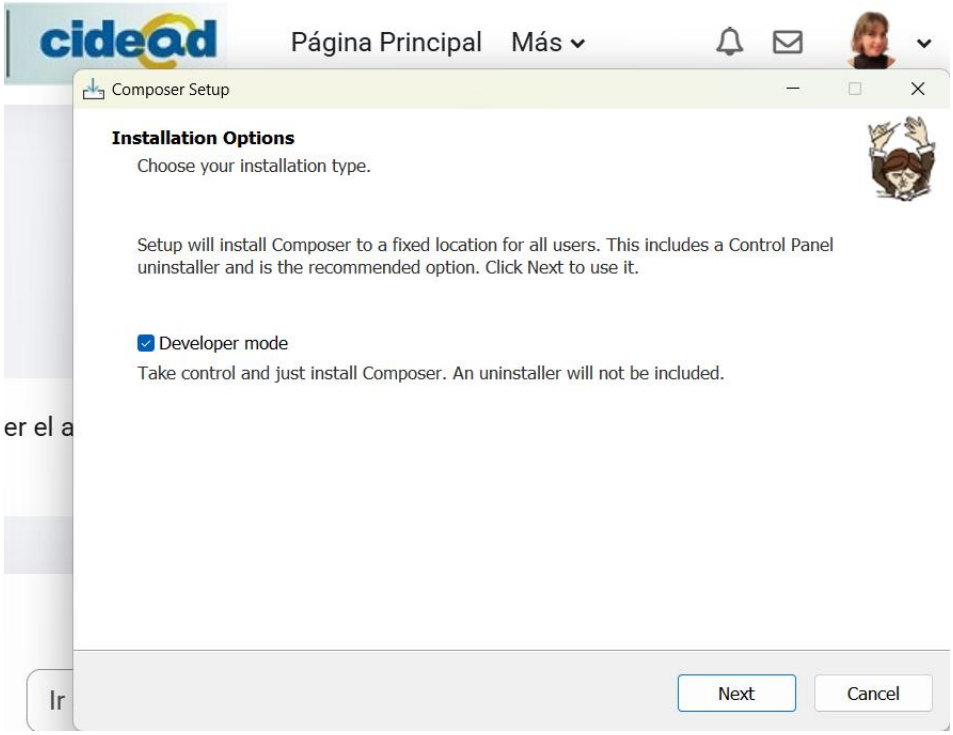
1.3. Instalación *Composer* para PHPUnit

Se debe instalar *Composer* antes que PHPUnit porque PHPUnit es una dependencia (o paquete de software) que se recomienda instalar y gestionar a través de *Composer*.

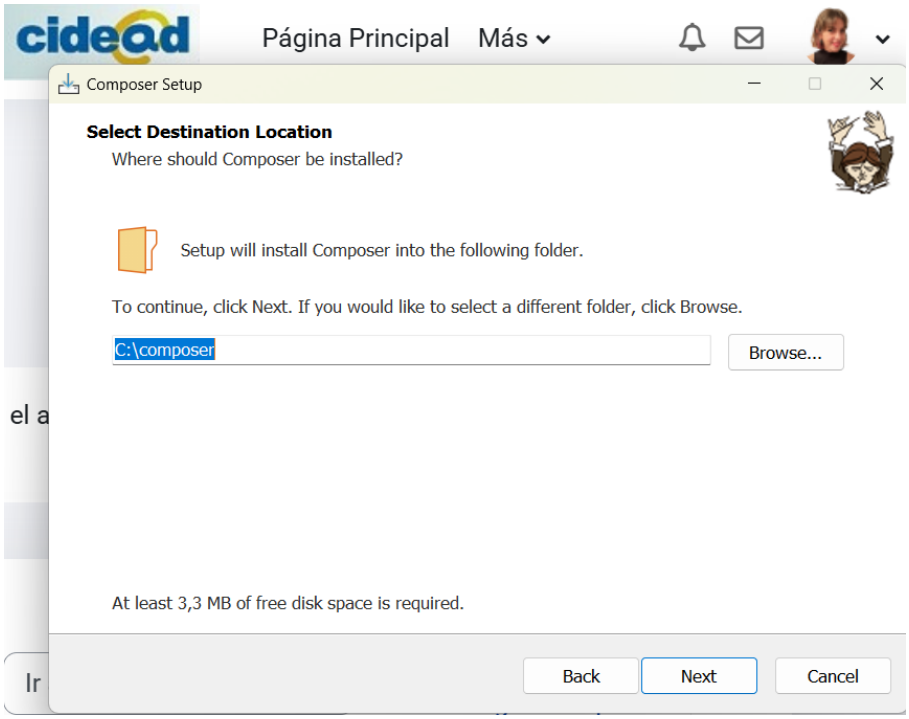
En el primer paso se debe seleccionar el modo de instalación, en este caso se instala para todos los usuarios.



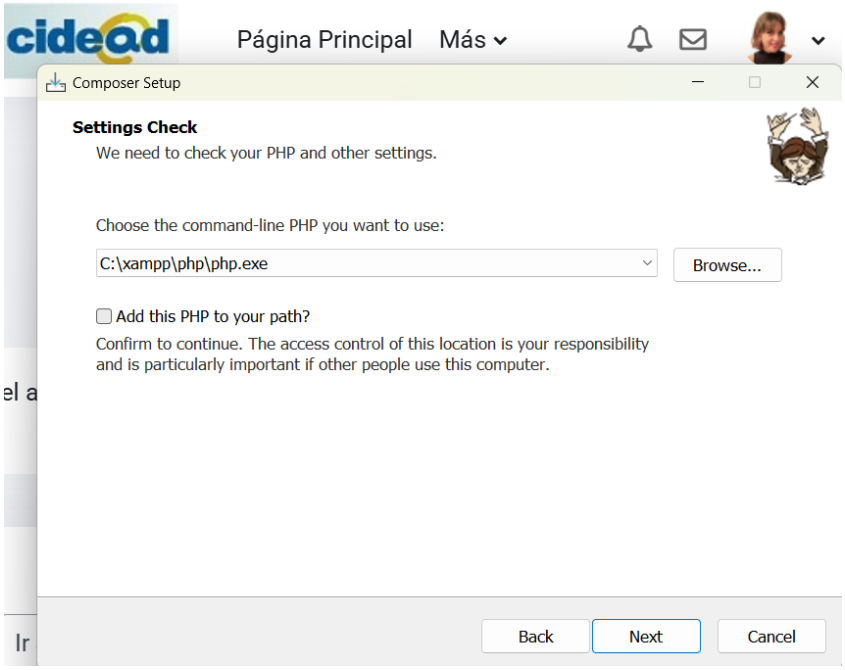
En las opciones de instalación, se marca el modo de desarrollador/a para una instalación más directa y menos "interferida".



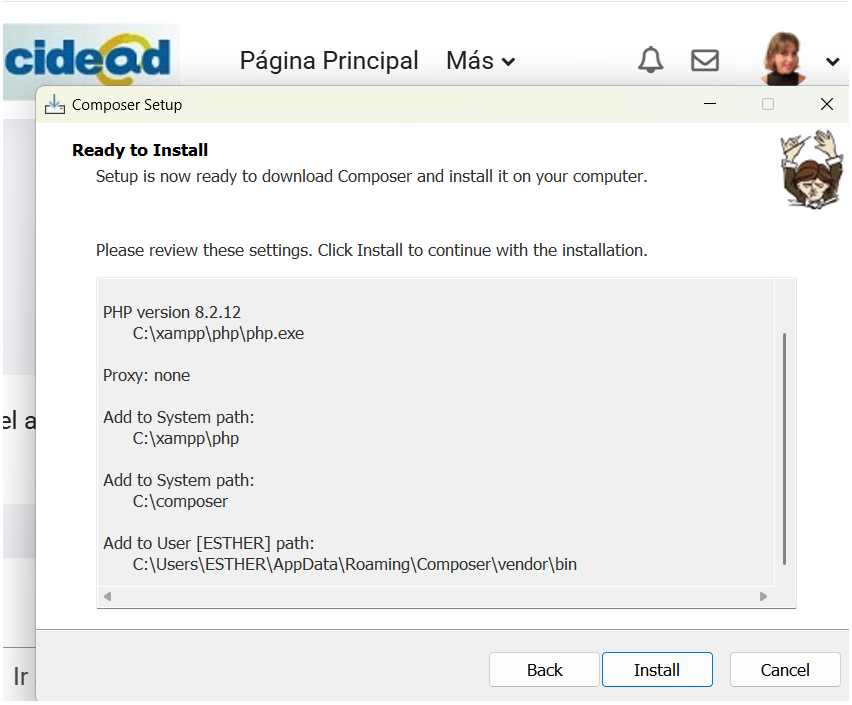
En el siguiente paso se deja por defecto el directorio para la instalación.



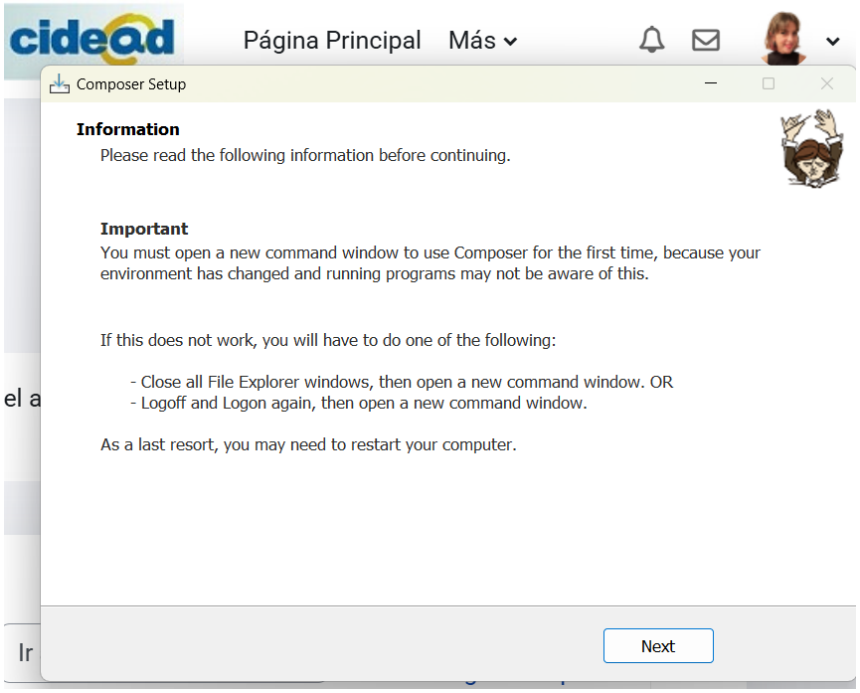
Se deja seleccionado la ruta donde se encuentra *Composer*.



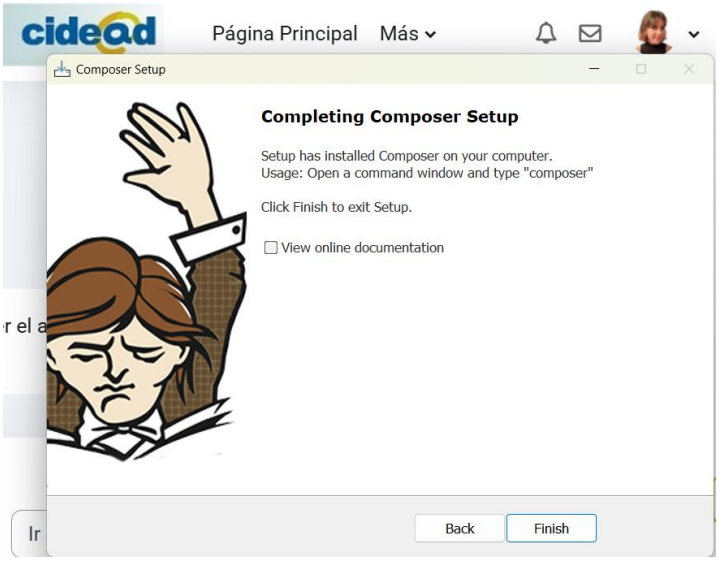
Este paso es el resumen de la configuración actual para el intérprete de PHPUnit. Pulsar el botón *install* para proceder a la instalación.



Este mensaje es un aviso sobre cómo empezar a usar *Composer* después de la instalación.



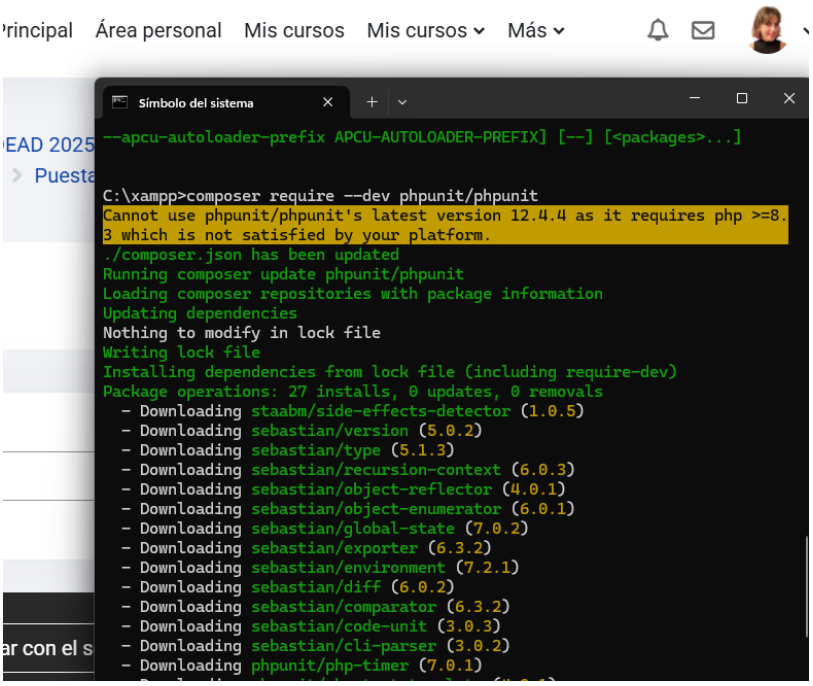
Esta simpática ilustración, indica que la instalación ha sido completada.



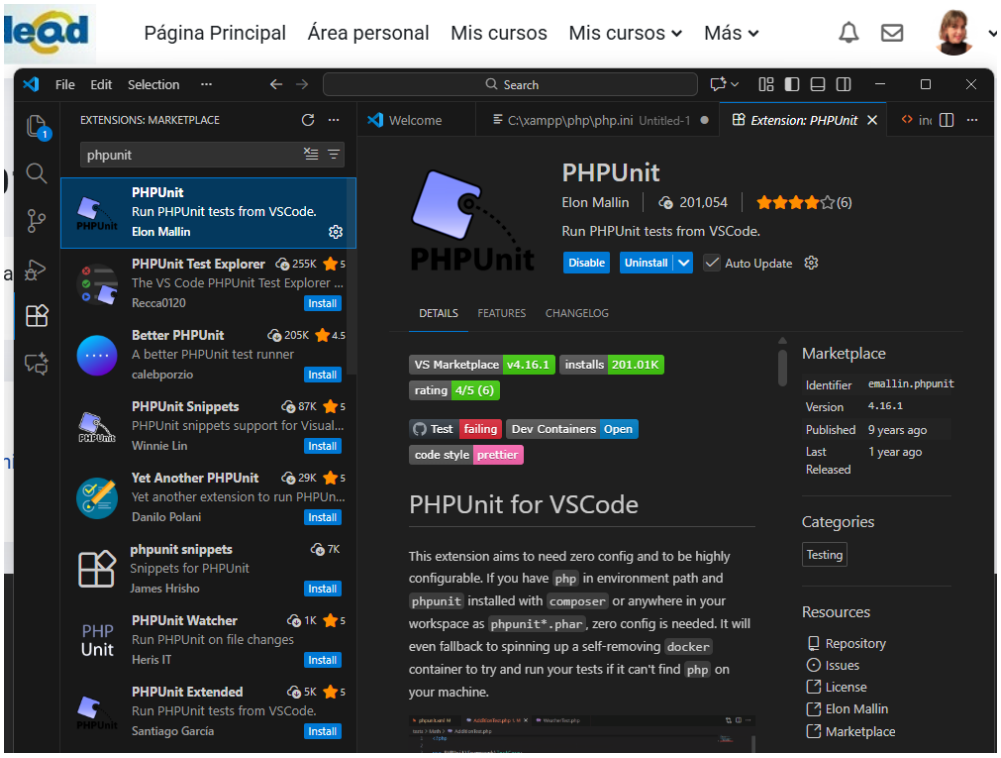
1.4. Instalación PHPUnit

Se abre la línea de comandos y se ejecuta el comando *composer require --dev phpunit/phpunit* para instalar PHPUnit es el directorio raíz del proyecto PHP.

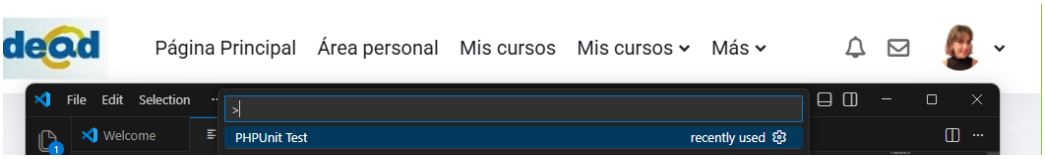
Composer advierte que no usaría la versión más nueva de PHPUnit, pero logra instalar correctamente una versión anterior que es compatible con la versión de PHP que está disponible en XAMPP.



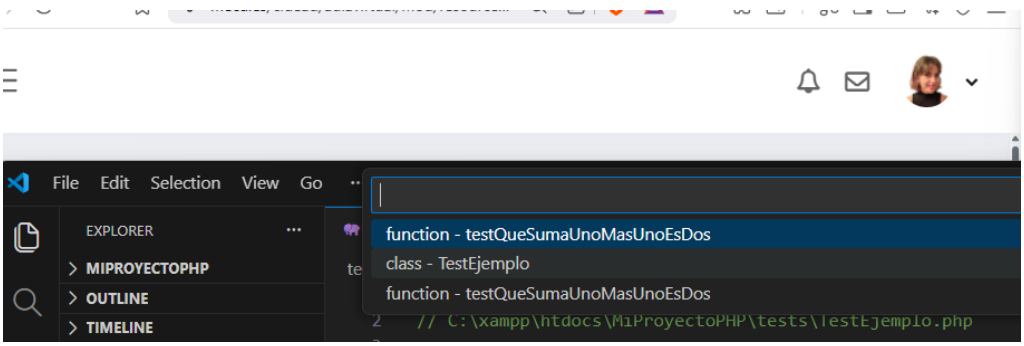
El siguiente paso es instalar la extensión de PHPUnit en Visual Studio Code.



A continuación, se configura el entorno para poder trabajar con PHP y poder realizar las pruebas unitarias en el mismo entorno. Se pulsa la combinación de teclas *Ctrl + Shift + P* para abrir una barra de búsqueda en la parte superior del editor.



Se escoge una clase o función de las que se despliegan.

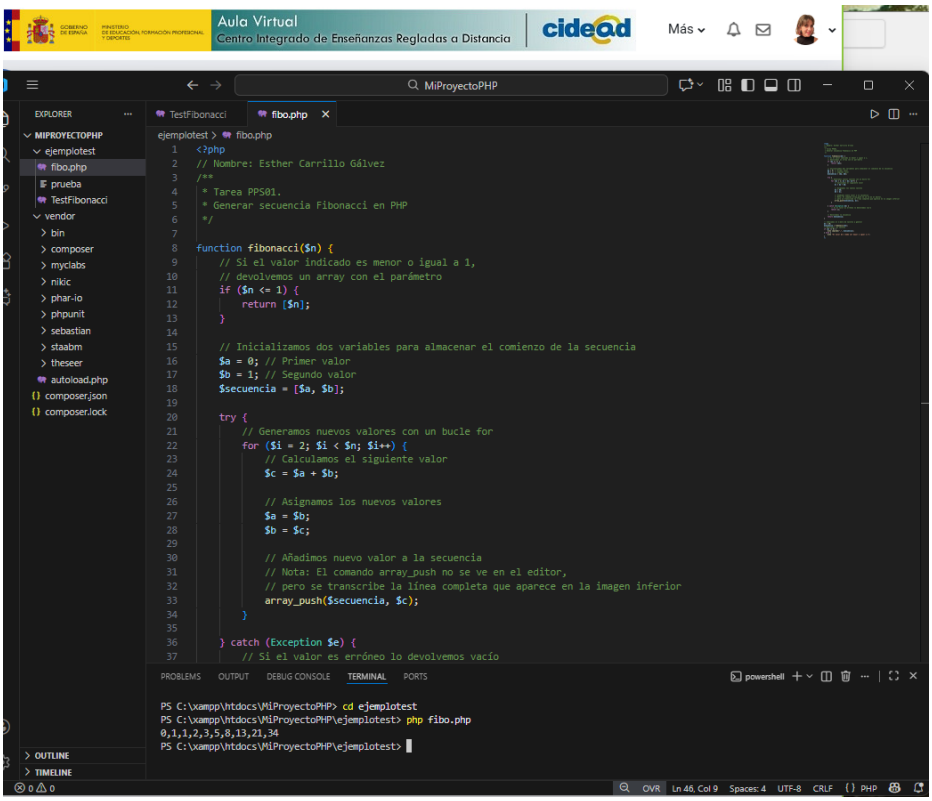


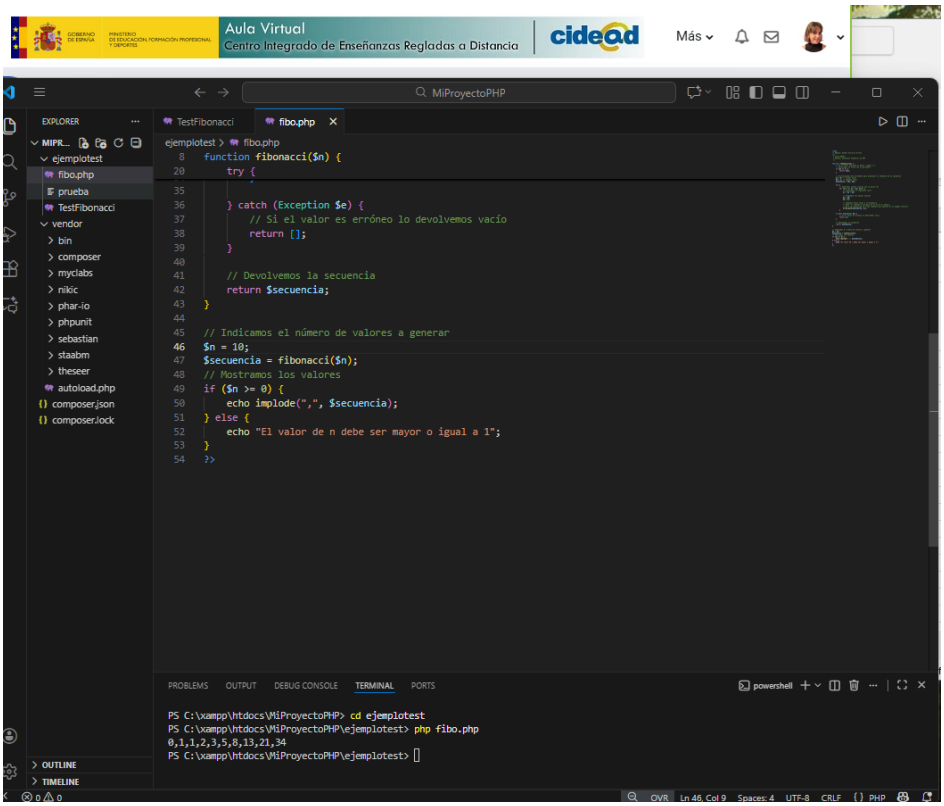
Apartado 2: Rellenar tabla con el código del programa de pruebas

PREGUNTA	RESPUESTA
Indica el número de las líneas donde hay un comentario	3: // Importa el archivo con la función a probar 5: // utiliza la librería con PHPUnit 7: // crea una clase que hereda de TestCase 9 : // crea un método para realizar las pruebas
Indica el número de las líneas donde hay comandos de pre-procesamiento	2 : require 'ejemplo.php'; 4: use PHPUnit\Framework\TestCase;
Indica el nombre de las variables utilizadas	10: $n 11: $resultado_esperado 12: $resultado_obtenido
Indica el nombre de los métodos/funciones utilizadas	12: obtenerPrimerosNumeros() 13: assertEquals()
Indica qué hace la línea 13	13: $this->assertEquals($resultado_esperado, $resultado_obtenido); Esta línea realiza una aserción. Compara el valor esperado ($resultado_esperado) con el valor obtenido ($resultado_obtenido) $this indica que el métido pertenece a la clase de prueba EjemploTest.

Apartado 3: Crear un programa PHP que genere la secuencia de Fibonacci.

Código php que lanza la sucesión de Fibonacci.


Aula Virtual
Centro Integrado de Enseñanzas Regladas a Distancia
cidead
Más
MiProyectoPHP
EXPLORER
MIPROYECTOPHP
ejemplotest
fibo.php
prueba
TestFibonacci
vendor
bin
composer
myclabs
nikic
phar-io
phpunit
sebastian
staabm
theseer
autoload.php
composer.json
composer.lock
TestFibonacci
fibo.php
ejemplotest > fibo.php
<?php
// Nombre: Esther Carrillo Gálvez
/**
* Tarea PPS01.
* Generar secuencia Fibonacci en PHP
*/
function fibonacci($n) {
// Si el valor indicado es menor o igual a 1,
// devolvemos un array con el parámetro
if ($n <= 1) {
return [$n];
}
// Inicializamos dos variables para almacenar el comienzo de la secuencia
$a = 0; // Primer valor
$b = 1; // Segundo valor
$secuencia = [$a, $b];
try {
// Generamos nuevos valores con un bucle for
for ($i = 2; $i < $n; $i++) {
// Calculamos el siguiente valor
$c = $a + $b;
// Asignamos los nuevos valores
$a = $b;
$b = $c;
// Añadimos nuevo valor a la secuencia
// Nota: El comando array_push no se ve en el editor,
// pero se transcribe la línea completa que aparece en la imagen inferior
array_push($secuencia, $c);
}
} catch (Exception $e) {
// Si el valor es erróneo lo devolvemos vacío
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
powershell
PS C:\xampp\htdocs\MiProyectoPHP> cd ejemplotest
PS C:\xampp\htdocs\MiProyectoPHP\ejemplotest> php fibo.php
0,1,1,2,3,5,8,13,21,34
PS C:\xampp\htdocs\MiProyectoPHP\ejemplotest>
OUTLINE
TIMELINE
Ln 46, Col 9
Spaces: 4
UTF-8
CRLF
PHP

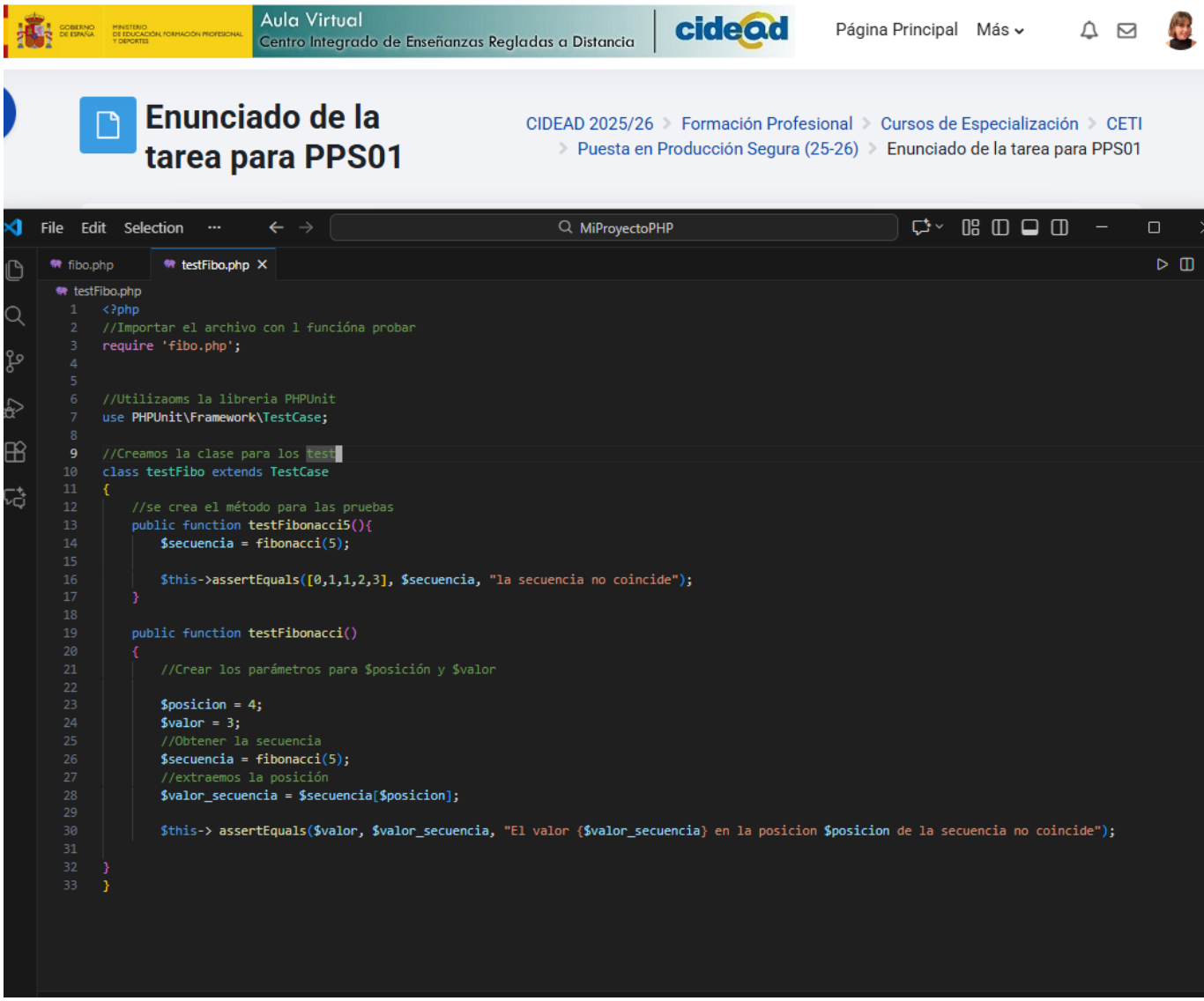

Aula Virtual
Centro Integrado de Enseñanzas Regladas a Distancia
cidead
Más
MiProyectoPHP
EXPLORER
ejemplotest
fibo.php
prueba
TestFibonacci
vendor
bin
composer
myclabs
nikic
phar-io
phpunit
sebastian
staabm
theseer
autoload.php
composer.json
composer.lock
TestFibonacci
fibo.php
ejemplotest > fibo.php
function fibonacci($n) {
try {
} catch (Exception $e) {
// Si el valor es erróneo lo devolvemos vacío
return [];
}
// Devolvemos la secuencia
return $secuencia;
}
// Indicamos el número de valores a generar
$n = 10;
$secuencia = fibonacci($n);
// Mostramos los valores
if ($n >= 0) {
echo implode(",", $secuencia);
} else {
echo "El valor de n debe ser mayor o igual a 1";
}
?>
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
powershell
PS C:\xampp\htdocs\MiProyectoPHP> cd ejemplotest
PS C:\xampp\htdocs\MiProyectoPHP\ejemplotest> php fibo.php
0,1,1,2,3,5,8,13,21,34
PS C:\xampp\htdocs\MiProyectoPHP\ejemplotest>
OUTLINE
TIMELINE
Ln 46, Col 9
Spaces: 4
UTF-8
CRLF
PHP

Código php completo.

```php
<?php
// Nombre: Esther Carrillo Gálvez
/**
 * Tarea PPS01.
 * Generar secuencia Fibonacci en PHP
 */

function fibonacci($n) {
    // Si el valor indicado es menor o igual a 1,
    // devolvemos un array con el parámetro
    if ($n <= 1) {
        return [$n];
    }

    // Inicializamos dos variables para almacenar el comienzo de la secuencia
    $a = 0; // Primer valor
    $b = 1; // Segundo valor
    $secuencia = [$a, $b];

    try {
        // Generamos nuevos valores con un bucle for
        for ($i = 2; $i < $n; $i++) {
            // Calculamos el siguiente valor
            $c = $a + $b;

            // Asignamos los nuevos valores
            $a = $b;
            $b = $c;

            // Añadimos nuevo valor a la secuencia
            // Nota: El comando array_push no se ve en el editor,
            // pero se transcribe la línea completa que aparece en la imagen inferio
            array_push($secuencia, $c);
        }

    } catch (Exception $e) {
        // Si el valor es erróneo lo devolvemos vacío
        return [];
    }

    // Devolvemos la secuencia
    return $secuencia;
}

// Indicamos el número de valores a generar
$n = 10;
$secuencia = fibonacci($n);
// Mostramos los valores
if ($n >= 0) {
    echo implode(",", $secuencia);
} else {
    echo "El valor de n debe ser mayor o igual a 1";
}
?>
```

Apartado 4: Creación del programa de pruebas unitarias

Código PHP para realizar la prueba unitaria sobre el fichero *fibo.php*.

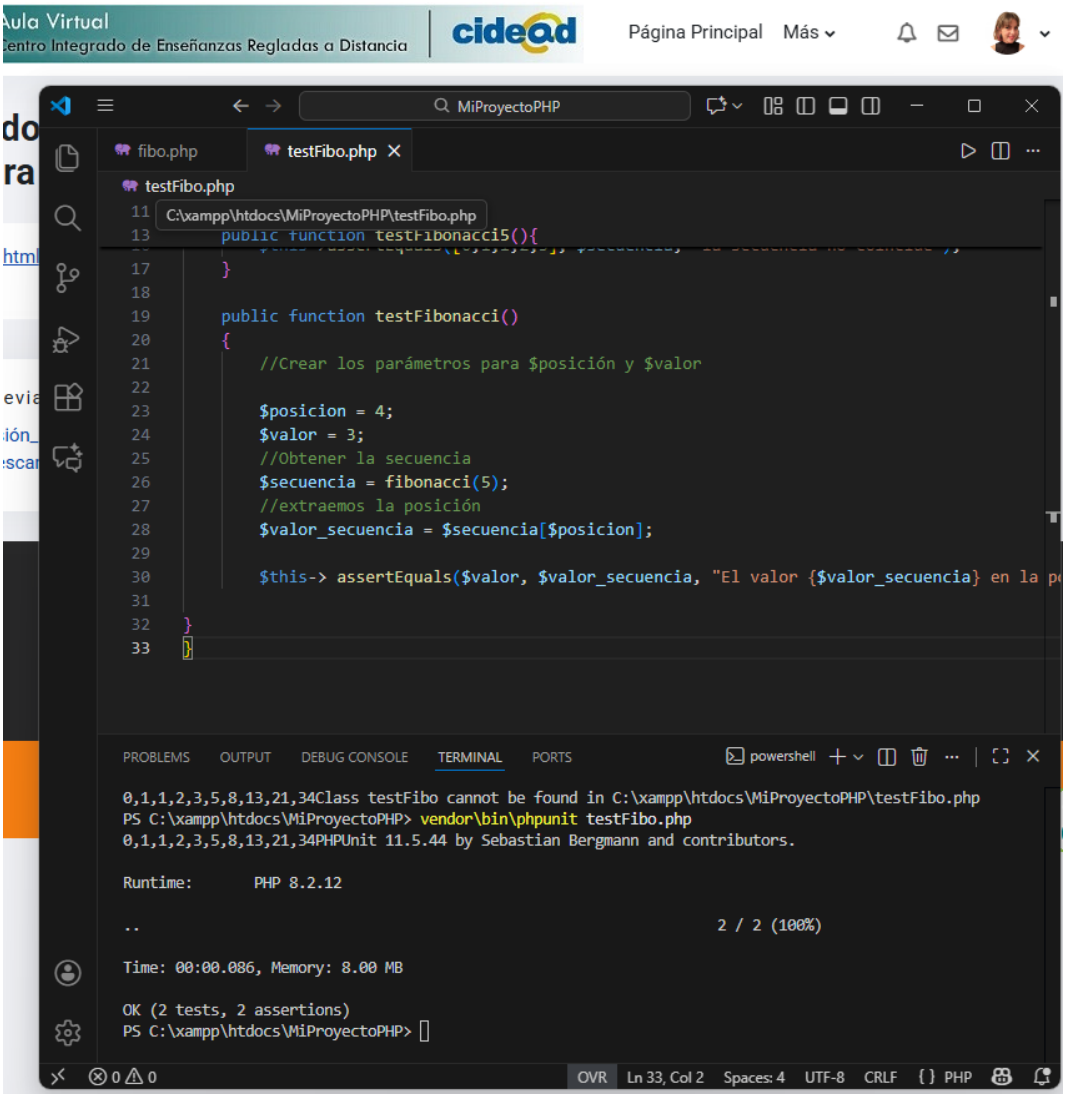


Código php completo.

```php
<?php
//Importar el archivo con l funcióna probar
require 'fibo.php';


//Utilizaoms la libreria PHPUnit
use PHPUnit\Framework\TestCase;

//Creamos la clase para los test
class testFibo extends TestCase
{
    //se crea el método para las pruebas
    public function testFibonacci5(){
        $secuencia = fibonacci(5);

        $this->assertEquals([0,1,1,2,3], $secuencia, "la secuencia no coincide");
    }

    public function testFibonacci()
    {
        //Crear los parámetros para $posición y $valor

        $posicion = 4;
        $valor = 3;
        //Obtener la secuencia
        $secuencia = fibonacci(5);
        //extraemos la posición
        $valor_secuencia = $secuencia[$posicion];

        $this-> assertEquals($valor, $valor_secuencia, "El valor {$valor_secuencia} en la posicion $posicion de la secuencia no coincide");

}
}
```

Apartado 5: Verificación de software

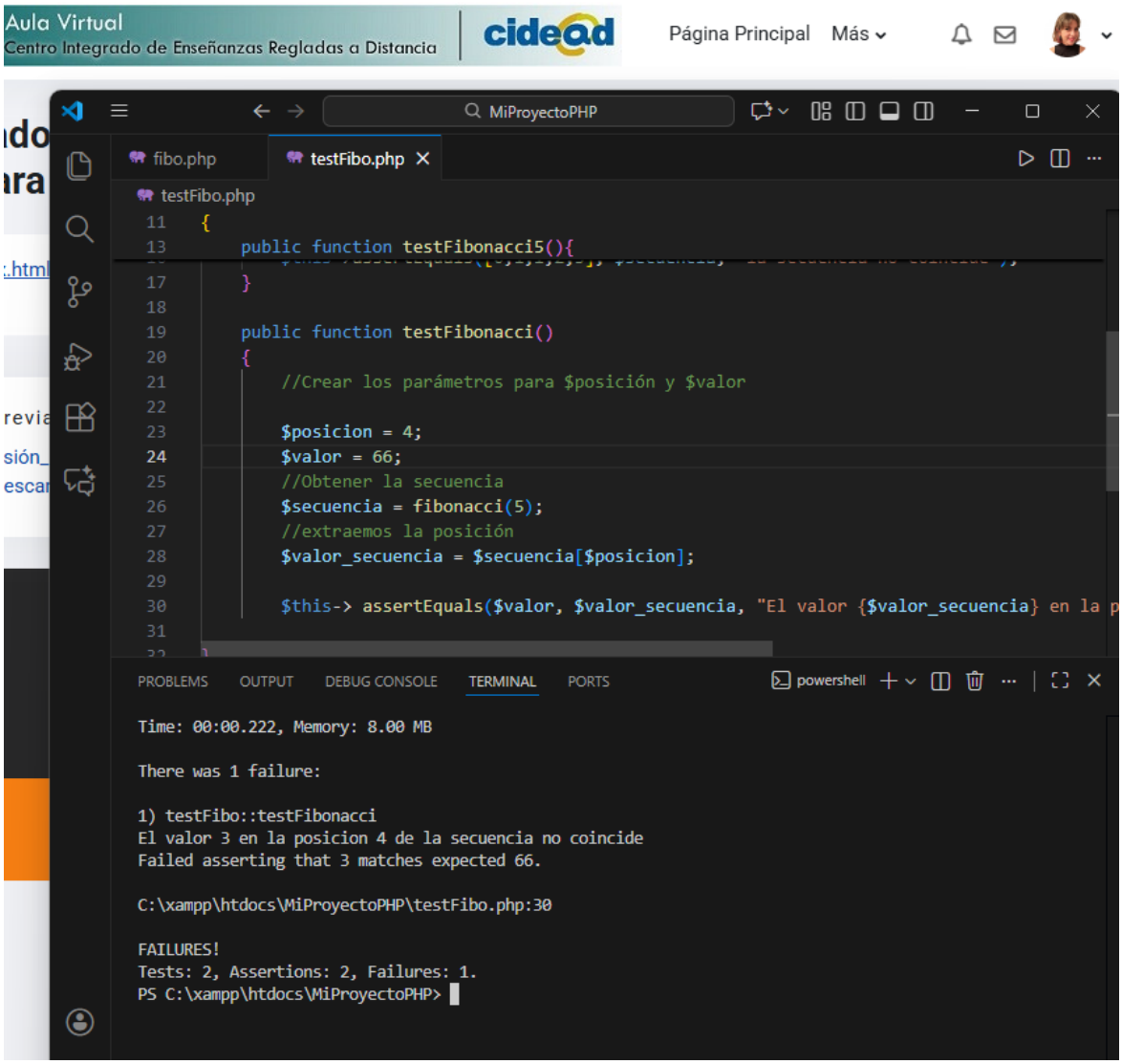
Se ejecuta el comando *vendor\bin\phpunit testFibo.php* para correr todas las pruebas contenidas en el archivo. Es exitoso.



Comprobación de la capacidad del programa de detectar errores mediante el cambio del valor 3 por el valor 66 en línea 24.

· El mensaje final del test**: FAILURES! Tests: 1, Assertions: 1, Failures: 1.**

La aparición del mensaje de error es la comprobación de que el test detecta errores.


Aula Virtual
Centro Integrado de Enseñanzas Regladas a Distancia
cidead
Página Principal
Más
MiProyectoPHP
fibo.php
testFibo.php
testFibo.php
11 {
13 public function testFibonacci5(){
17 }
19 public function testFibonacci()
20 {
21 //Crear los parámetros para $posición y $valor
23 $posicion = 4;
24 $valor = 66;
25 //Obtener la secuencia
26 $secuencia = fibonacci(5);
27 //extraemos la posición
28 $valor_secuencia = $secuencia[$posicion];
30 $this-> assertEquals($valor, $valor_secuencia, "El valor {$valor_secuencia} en la p
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
powershell
Time: 00:00.222, Memory: 8.00 MB
There was 1 failure:
1) testFibo::testFibonacci
El valor 3 en la posicion 4 de la secuencia no coincide
Failed asserting that 3 matches expected 66.
C:\xampp\htdocs\MiProyectoPHP\testFibo.php:30
FAILURES!
Tests: 2, Assertions: 2, Failures: 1.
PS C:\xampp\htdocs\MiProyectoPHP>

Apartado 6: Responder a la pregunta

Esta vulnerabilidad se presenta cuando las credenciales están escritas en el código fuente de la aplicación. Lo que ocurre en el código de ejemplo es que un atacante podría vulnerar el sistema si tuviera el código fuente, ya que las líneas 3, 4, 5 y 6 revelan información muy sensible. En estas líneas se leen cuatro variables relacionadas con el nombre del servidor (*$servername = "db1"*), el nombre de la base de datos (*$database = "databasename"*), y las credenciales de usuario *($username = "username"; $password = "password"*).

```
<?php
require_once 'pdoconfig.php';
$servername = "db1";
$database = "databasename";
$username = "username";
$password = "password";

try {
        $conn = new PDO("mysql:host=$servername;dbname=$database", $username, $password);
        echo "Conectado a $database de $servername con exito.";
} catch (PDOException $pe) {
        die("No es posible la conexión a  $database :" . $pe->getMessage());
}
?>
```

Para resolver el problema de credenciales fijas, la práctica recomendada es almacenar las credenciales en un gestor Secrets Manager como Azure Key Vault. Además de cargar las credenciales en un archivo de configuración local (.env, config.ini) fuera del directorio web (htdocs).

Bibliografía

Apache Friends. (s. f.). *XAMPP*. Recuperado el 1 de diciembre de 2025, de https://www.apachefriends.org/es/index.html

SonarSource. (s. f.). *Rule RSPEC-2068: Passwords should not be hardcoded*. Recuperado el 1 de diciembre de 2025, de https://rules.sonarsource.com/typescript/tag/cwe/RSPEC-2068/

Visual Studio Code. (s. f.). *Code editing. Redefined*. Recuperado el 1 de diciembre de 2025, de https://code.visualstudio.com/