Pruebas unitarias

PHPunit

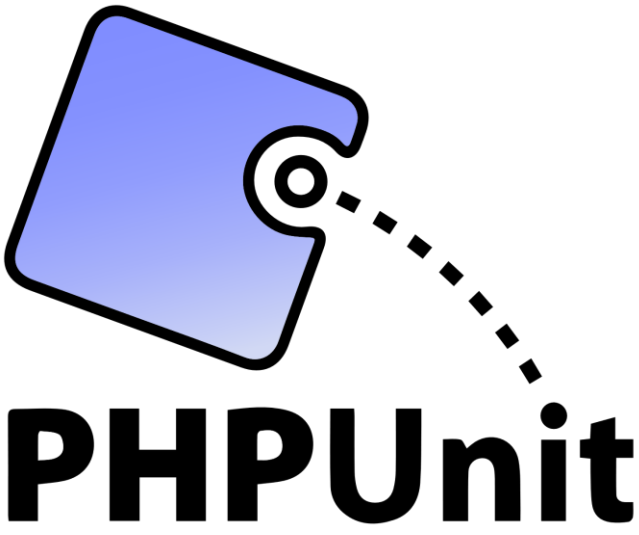

Tabla de contenido

Prueba de conexión a BD

En esta prueba deseamos saber si la conexión con la base de datos es satisfactoria ya que sin ella no podremos hacer los CRUD´S

```php
/** @test **/
public function conect() {
  $conex=new conexion;
  $conn=$conex->conect();
  $this->assertEquals(2, $conn);

}
```

```php
public function conect(){
  $conexion = mysqli_connect("localhost","root","","prueba");
  if(!$conexion){
    return 'mal';
  }else{
    return 2;
  }
}
```

Prueba	Condición	Parámetros	Resultado
#1	**conexion:** Método que retorna un booleano si la comparación es correcta o incorrecta, esto gracias al método **assertEquals(?,?)**	**conect():** función que nos devuelve un entero 2 si la conexión fue establecida correctamente	True

```
PS C:\xampp\htdocs\Vender> .\vendor\bin\phpunit
PHPUnit 5.7.27 by Sebastian Bergmann and contributors.

Runtime:       PHP 7.4.5
Configuration: C:\xampp\htdocs\Vender\phpunit.xml

......                                                              6 / 6 (100%)

Time: 225 ms, Memory: 4.00MB

OK (6 tests, 6 assertions)
PS C:\xampp\htdocs\Vender>
```

Todas las pruebas han sido realizadas sin ningún fallo, ni error

Prueba Consulta de datos

Verificamos que podemos hacer una consulta de los datos que se encuentran en nuestra base de datos

```php
/** @test **/
public function consulta() {
  $sql = "Select marca from productos where Nombre='Pan'";
  $conexion = mysqli_connect("localhost","root","","prueba");
  $result=mysqli_query($conexion,$sql);
  $fila = mysqli_fetch_row($result);
  $this->assertEquals('comapan', $fila[0]);

}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#2	**consulta():** Método que retorna un booleano si la comparación es correcta o incorrecta, esto gracias al método **assertEquals(?,?)**	**$sql:** consulta sql **$result:** variable en la que guardamos el resultado de nuestra consulta sql a la BD **$fila:** guarda el resultado de nuestra consulta por filas		True

```
PS C:\xampp\htdocs\Vender> .\vendor\bin\phpunit
PHPUnit 5.7.27 by Sebastian Bergmann and contributors.

Runtime:       PHP 7.4.5
Configuration: C:\xampp\htdocs\Vender\phpunit.xml

......                                                              6 / 6 (100%)

Time: 225 ms, Memory: 4.00MB

OK (6 tests, 6 assertions)
PS C:\xampp\htdocs\Vender>
```

Todas las pruebas han sido realizadas sin ningún fallo, ni error

Prueba Inserción de datos

Verificaremos que la inserción de los datos en la base de datos se realiza de manera satisfactoria

```php
/** @test **/
public function insercion() {
  $conex=new conexion;
  $conn=$conex->Addprod('Arepa','Doña lola','2022-01-29',3800,2);
  $this->assertEquals('Good', $conn);

}
```

```php
public function Addprod($Nombre,$marca,$date,$cost,$cant){
  $conexion = mysqli_connect("localhost","root","","prueba");
  $sql = "INSERT INTO productos(Nombre, marca, fecha_venc, costo, cantidad)
  VALUES ('$Nombre','$marca','$date',$cost,$cant)";
  $result=mysqli_query($conexion,$sql);
  if(!$result){
    return 'Producto unregister';
  }else{
    return 'Good';
  }
}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#3	**insercion():** Método que retorna un booleano si la comparación es correcta o incorrecta, esto gracias al método **assertEquals(?,?)**	**$sql:** consulta sql **$result:** variable en la que guardamos el resultado de nuestra consulta sql a la BD **$conn:** guarda el resultado del método **Addprod()** **Addprod():** método al que se le pasan los datos del producto	**Addprod(**'Arepa','Doña lola','2022-01-29',3800,2**);**	True

```
PS C:\xampp\htdocs\Vender> .\vendor\bin\phpunit
PHPUnit 5.7.27 by Sebastian Bergmann and contributors.

Runtime:       PHP 7.4.5
Configuration: C:\xampp\htdocs\Vender\phpunit.xml

......                                                              6 / 6 (100%)

Time: 225 ms, Memory: 4.00MB

OK (6 tests, 6 assertions)
PS C:\xampp\htdocs\Vender>
```

Todas las pruebas han sido realizadas sin ningún fallo, ni error

Inserción no vacía

```php
/** @test **/
public function Insercion_NEmp() {
    $conex=new conexion;
    $conn=$conex->Addprod('Chocolatina','Hershey','2024-01-29',4000,4);
    $this->assertNotEmpty($conn);
}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#4	**Inserción_NEmp():** Método que retorna un booleano, si el parámetro no está vacío	**$conn:** guarda el resultado del método **Addprod()** **Addprod():** método al que se le pasan los datos del producto	**Addprod(**Chocolatina,'Hershey', '2024-01-29',4000,4**)**;	True

```
PS C:\xampp\htdocs\Vender> .\vendor\bin\phpunit
PHPUnit 5.7.27 by Sebastian Bergmann and contributors.

Runtime:       PHP 7.4.5
Configuration: C:\xampp\htdocs\Vender\phpunit.xml

......                                                              6 / 6 (100%)

Time: 225 ms, Memory: 4.00MB

OK (6 tests, 6 assertions)
PS C:\xampp\htdocs\Vender>
```

Como el parámetro ingresado no se encuentra vacío la aserción podemos ver a posteriori que se realiza satisfactoriamente

Prueba Actualización de datos

Verificaremos que la actualización de los datos en la base de datos se realiza de manera satisfactoria

```php
/** @test **/
public function actualizacion() {
  $conex=new conexion;
  $conn=$conex->Updateprod('Arepa','Doña Petra','2023-01-29',3800,2);
  $this->assertEquals('Good', $conn);

}
```

```php
public function Updateprod($Nombre,$marca,$date,$cost,$cant){
  $conexion = mysqli_connect("localhost","root","","prueba");
  $sql = "UPDATE productos SET nombre='".$Nombre."', marca= '".$marca."',
  fecha_venc='".$date."', costo='".$cost."', Cantidad='".$cant."'
  WHERE id=12 ";
  $result=mysqli_query($conexion,$sql);
  if(!$result){
    return 'Producto nonupdate';
  }else{
    return 'Good';
  }
}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#5	**actualizacion():** Método que retorna un booleano si la comparación es correcta o incorrecta, esto gracias al método **assertEquals(?,?)**	**$sql:** consulta sql **$result:** variable en la que guardamos el resultado de nuestra consulta sql a la BD **$conn:** guarda el resultado del método **Updateprod()** **Updateprod():** método al que se le pasan los datos del producto	**Updateprod(**'Arepa','Doña Petra,'2023-01-29',3800,2**);**	True

```
PS C:\xampp\htdocs\Vender> .\vendor\bin\phpunit
PHPUnit 5.7.27 by Sebastian Bergmann and contributors.

Runtime:       PHP 7.4.5
Configuration: C:\xampp\htdocs\Vender\phpunit.xml

......                                                              6 / 6 (100%)

Time: 225 ms, Memory: 4.00MB

OK (6 tests, 6 assertions)
PS C:\xampp\htdocs\Vender>
```

Todas las pruebas han sido realizadas sin ningún fallo, ni error

Prueba Eliminación de datos

Verificaremos que la eliminación de los datos en la base de datos se realiza de manera satisfactoria

```php
/** @test **/
public function Eliminar() {
  $conex=new conexion;
  $conn=$conex->Deleteprod(14);
  $this->assertEquals('Good', $conn);

}
```

```php
public function Deleteprod($num){
  $conexion = mysqli_connect("localhost","root","","prueba");
  $sql = "DELETE FROM productos WHERE id=$num ";
  $result=mysqli_query($conexion,$sql);
  if(!$result){
    return 'Producto nondelete';
  }else{
    return 'Good';
  }
}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#6	**Eliminar():** Método que retorna un booleano si la comparación es correcta o incorrecta, esto gracias al método **assertEquals(?,?)**	**$sql:** consulta sql **$result:** variable en la que guardamos el resultado de nuestra consulta sql a la BD **$conn:** guarda el resultado del método **Deleteprod()** **Deleteprod():** se le pasa el id del producto a eliminar		True

```
PS C:\xampp\htdocs\Vender> .\vendor\bin\phpunit
PHPUnit 5.7.27 by Sebastian Bergmann and contributors.

Runtime:       PHP 7.4.5
Configuration: C:\xampp\htdocs\Vender\phpunit.xml

......                                                              6 / 6 (100%)

Time: 225 ms, Memory: 4.00MB

OK (6 tests, 6 assertions)
PS C:\xampp\htdocs\Vender>
```

Todas las pruebas han sido realizadas sin ningún fallo, ni error

Prueba Eliminar Same

```php
/** @test **/
public function Eliminar_Same() {
    $conex=new conexion;
    $conn=$conex->Deleteprod(14);
    $this->assertSame('Good',$conn);
}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#7	**Eliminar_Same():** Método que retorna un booleano si la comparación es correcta o incorrecta, esto gracias al método **assertSame(?,?)**	**assertSame():** Compara entre el resultado esperado en este caso 'Good' y el obtenido en la variable **$conn**		True

```
PS C:\xampp\htdocs\Vender> .\vendor\bin\phpunit
PHPUnit 5.7.27 by Sebastian Bergmann and contributors.

Runtime:       PHP 7.4.5
Configuration: C:\xampp\htdocs\Vender\phpunit.xml

...........................                                 27 / 27 (100%)

Time: 690 ms, Memory: 4.00MB

OK (27 tests, 29 assertions)
```

La prueba al igual que el **assertEquals** hace la eliminación de manera correcta sin ningún fallo

Prueba de Columnas de tabla producto

Se quiere saber cuantas columnas nos esta devolviendo la tabla producto al hacer una consulta de todo su contenido

```php
/** @test **/
public function Producto_columns_count(){
  $sql = "Select * from productos where dia='2020-05-12'";
  $conexion = mysqli_connect("localhost","root","","prueba");
  $result=mysqli_query($conexion,$sql);
  $fila = mysqli_fetch_row($result);
  $this->assertCount(7,$fila);
}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#8	**Producto_columns_count():** Método que retorna un booleano si la comparación es correcta o incorrecta, esto gracias al método **assertCount(?,?)**	**$sql:** consulta sql **$result:** guarda el resultado de la consulta sql **$fila:** guardamos el resultado de la consulta sql (**$result**) dividida por cada una de sus columnas **assertCount():** Compara entre el resultado esperado en este caso 7 y el obtenido en la variable **$fila**		True

```
PS C:\xampp\htdocs\Vender> .\vendor\bin\phpunit
PHPUnit 5.7.27 by Sebastian Bergmann and contributors.

Runtime:       PHP 7.4.5
Configuration: C:\xampp\htdocs\Vender\phpunit.xml

...........................                                 27 / 27 (100%)

Time: 805 ms, Memory: 4.00MB

OK (27 tests, 29 assertions)
```

Ninguna de las aserciones probadas tiene problema al momento de su ejecución

Prueba de cantidad no nula

Verificamos que el resultado del precio de un producto no sea nulo

```php
/** @test **/
public function Cantidad_not_null() {
  $sql = "Select costo*cantidad from productos where Nombre='Pan'";
  $conexion = mysqli_connect("localhost","root","","prueba");
  $result=mysqli_query($conexion,$sql);
  $this->assertNotNull(0);
}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#9	**Cantidad_not_null():** Método que retorna un booleano para el resultado de una consulta	**$sql:** consulta sql **$result:** guarda el resultado de la consulta sql **assertNotNull():** si el proceso anterior es nulo o no		True

Prueba de cantidad

Verificamos que el resultado de la suma de un producto sea igual al esperado

```php
/** @test **/
public function total_cntdd() {
  $sql = "Select costo*cantidad from productos where Nombre='Pan'";
  $conexion = mysqli_connect("localhost","root","","prueba");
  $result=mysqli_query($conexion,$sql);
  $fila = mysqli_fetch_row($result);
  $this->assertEquals(9000, $fila[0]);
}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#10	**Total_cntdd():** Método que retorna un booleano si la comparación es correcta o incorrecta, esto gracias al método **assertEquals(?,?)**	**$sql:** consulta sql **$result:** guarda el resultado de la consulta sql **$fila:** obtenemos el resultado separado por filas **$fila[0]:** especificamos que fila queremos		True

```
PS C:\xampp\htdocs\Vender> .\vendor\bin\phpunit
PHPUnit 5.7.27 by Sebastian Bergmann and contributors.

Runtime:       PHP 7.4.5
Configuration: C:\xampp\htdocs\Vender\phpunit.xml

......                                                              6 / 6 (100%)

Time: 225 ms, Memory: 4.00MB

OK (6 tests, 6 assertions)
PS C:\xampp\htdocs\Vender>
```

Todas las pruebas sin ningún fallo, ni error

Prueba ventas diarias

Verificamos el resultado de las ventas realizadas de un día en especifico

```php
/** @test **/
public function ventas_dia() {
  $sql = "Select Sum(costo*cantidad) from productos where dia='2020-05-12'";
  $conexion = mysqli_connect("localhost","root","","prueba");
  $result=mysqli_query($conexion,$sql);
  $fila = mysqli_fetch_row($result);
  $this->assertEquals(43500, $fila[0]);
}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#11	**Ventas_dia():** Método que retorna un booleano si la comparación es correcta o incorrecta, esto gracias al método **assertEquals(?,?)**	**$sql:** consulta sql **$result:** guarda el resultado de la consulta sql **$fila:** obtenemos el resultado separado por filas **$fila[0]:** especificamos que fila queremos		True

```
PS C:\xampp\htdocs\Vender> .\vendor\bin\phpunit
PHPUnit 5.7.27 by Sebastian Bergmann and contributors.

Runtime:       PHP 7.4.5
Configuration: C:\xampp\htdocs\Vender\phpunit.xml

.......                                                             7 / 7 (100%)

Time: 228 ms, Memory: 4.00MB

OK (7 tests, 7 assertions)
PS C:\xampp\htdocs\Vender>
```

Todas las pruebas siguen sin ningún fallo o error

Prueba Ventas menor que

Probamos que la suma de los productos vendidos no pasa el umbral esperado para la venta promedio de un día

```php
/** @test **/
public function Ventasdia_less() {
  $sql = "Select Sum(costo*cantidad) from productos where dia='2020-05-12'"
  $conexion = mysqli_connect("localhost","root","","prueba");
  $result=mysqli_query($conexion,$sql);
  $fila = mysqli_fetch_row($result);
  $this->assertLessThan(44000,$fila[0]);
}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#12	**Ventasdia_less():** Método que retorna un booleano si la comparación es correcta o incorrecta, esto gracias al método **assertLessThan(?,?)**	**$sql:** consulta sql **$result:** guarda el resultado de la consulta sql **$fila:** obtenemos el resultado separado por filas **$fila[0]:** especificamos que fila queremos **assertLessThan():** el resultado de nuestra subdivisión por filas(**$fila[0]**) debe ser menor en este caso a nuestro valor 44000		True

Prueba ventas menor o igual

Probamos que las ventas del día por lo menos alcancen el umbral propuesto

```php
/** @test **/
public function Ventasdia_lessthan() {
  $sql = "Select Sum(costo*cantidad) from productos where dia='2020-05-12'";
  $conexion = mysqli_connect("localhost","root","","prueba");
  $result=mysqli_query($conexion,$sql);
  $fila = mysqli_fetch_row($result);
  $this->assertLessThanOrEqual(43500,$fila[0]);
}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#13	**Ventasdia_lessthan():** Método que retorna un booleano si la comparación es correcta o incorrecta, esto gracias al método **assertLessThanOrEqual(?,?)**	**$sql:** consulta sql **$result:** guarda el resultado de la consulta sql **$fila:** obtenemos el resultado separado por filas **$fila[0]:** especificamos que fila queremos **assertLessThanOrEqual():** el resultado de nuestra subdivisión por filas(**$fila[0]**) debe ser menor o igual en este caso a nuestro valor 43500		True

```
PS C:\xampp\htdocs\Vender> .\vendor\bin\phpunit
PHPUnit 5.7.27 by Sebastian Bergmann and contributors.

Runtime:       PHP 7.4.5
Configuration: C:\xampp\htdocs\Vender\phpunit.xml

...........................                                27 / 27 (100%)

Time: 805 ms, Memory: 4.00MB

OK (27 tests, 29 assertions)
```

Ninguna de las aserciones probadas tiene problema al momento de su ejecución

Prueba Cantidad no nula

Comprobamos que el total de una consulta de precio de un producto no es nulo, para ello usamos la aserción **assertNotNull()**

```php
/** @test **/
public function Cantidad_not_null() {
  $sql = "Select costo*cantidad from productos where Nombre='Pan'";
  $conexion = mysqli_connect("localhost","root","","prueba");
  $result=mysqli_query($conexion,$sql);
  $this->assertNotNull($result);
}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#14	**Cantidad_not_null():** Método que retorna un booleano	**$result:** Guardamos el resultado de nuestra consulta **assertNotNull:** Como parámetro le pasamos el **$result,** para verificar que el resultado de esa consulta no sea nulo		True

```
PS C:\xampp\htdocs\Vender> .\vendor\bin\phpunit
PHPUnit 5.7.27 by Sebastian Bergmann and contributors.

Runtime:       PHP 7.4.5
Configuration: C:\xampp\htdocs\Vender\phpunit.xml

...........................                                27 / 27 (100%)

Time: 690 ms, Memory: 4.00MB

OK (27 tests, 29 assertions)
```

Observamos que todas las pruebas fueron procesadas correctamente

Prueba costo mayor

Probamos que el costo de un producto este por encima del valor sugerido de venta, para tener algo de ganancia

```php
/** @test **/
public function Costo_Greater(){
  $sql = "Select costo from productos where marca='Jet'";
  $conexion = mysqli_connect("localhost","root","","prueba");
  $result=mysqli_query($conexion,$sql);
  $fila = mysqli_fetch_row($result);
  $this->assertGreaterThan(1000,$fila[0]);
}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#15	**Costo_Greater():** Método que retorna un booleano	**$result:** Guardamos el resultado de nuestra consulta **$fila:** Guardamos el resultado de la consulta sql (**$result**), por filas **$fila[0]:** especificamos que fila queremos obtener **assertGreaterThan:** el resultado de nuestra subdivisión por filas(**$fila[0]**) debe ser mayor en este caso a nuestro valor 1000		True

Prueba costo mayor o igual

Probamos que el costo de un producto este por encima o es el mismo que el valor sugerido de venta , para tener algo de ganancia

```php
/** @test **/
public function Costo_GreaterOrEqual(){
  $sql = "Select costo from productos where marca='Jet'";
  $conexion = mysqli_connect("localhost","root","","prueba");
  $result=mysqli_query($conexion,$sql);
  $fila = mysqli_fetch_row($result);
  $this->assertGreaterThanOrEqual(1450,$fila[0]);
}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#16	**Costo_GreaterOrEqual():** Método que retorna un booleano	**$result:** Guardamos el resultado de nuestra consulta **$fila:** Guardamos el resultado de la consulta sql (**$result**), por filas **$fila[0]:** especificamos que fila queremos obtener **assertGreaterThanOrEqual:** el resultado de nuestra subdivisión por filas(**$fila[0]**) debe ser mayor o igual en este caso a nuestro valor 1450		True

```
PS C:\xampp\htdocs\Vender> .\vendor\bin\phpunit
PHPUnit 5.7.27 by Sebastian Bergmann and contributors.

Runtime:       PHP 7.4.5
Configuration: C:\xampp\htdocs\Vender\phpunit.xml

...........................                                 27 / 27 (100%)

Time: 805 ms, Memory: 4.00MB

OK (27 tests, 29 assertions)
```

Podemos observar que las pruebas del costo son ejecutadas perfectamente sin ningún contratiempo

Prueba Index existe

Verificamos que el archivo este creado en nuestra carpeta del proyecto

```php
public function Exist_Index(){
    $this->assertFileExists('C:xampp/htdocs/Vender/index.php');
}
```

Prueba	Condición	Parámetros	Objeto	Resultado
#17	**Exist_Index():** Método que retorna un booleano	**assertFileExists:** donde vamos a pasar la ruta en la cual debe estar nuestro archivo		True

```
PS C:\xampp\htdocs\Vender> .\vendor\bin\phpunit
PHPUnit 5.7.27 by Sebastian Bergmann and contributors.

Runtime:       PHP 7.4.5
Configuration: C:\xampp\htdocs\Vender\phpunit.xml

.........                                                           9 / 9 (100%)

Time: 210 ms, Memory: 4.00MB

OK (9 tests, 9 assertions)
PS C:\xampp\htdocs\Vender>
```

La prueba realizada es ejecutada sin ningún problema