***CASOS DE PRUEBA Y PROCEDIMIENTOS DE PRUEBA***

***“Biblioteca ITSL”***

***“Centro de Desarrollo de Aplicaciones WEB y Moviles del ITSL”***



***ISO/IEC 29110-4-1:2018***

***HISTORIAL DE VERSIONES***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***VERSIÓN*** | ***FECHA VIGENCIA*** | ***DETALLE DEL CAMBIO*** | ***SECCIÓN CAMBIADA*** | ***AUTOR*** | ***FECHA AUTORIZACIÓN*** |
| 1.0 | 09/05/2021 | Versión Inicial |  | MSc J. Jesús Ríos Acevedo | 09/11/2021 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

***CASOS DE PRUEBA Y PROCEDIMIENTOS DE PRUEBA***

***Los casos de prueba pueden incluir:***

* ***Identificación del caso de prueba.***
* ***Elementos a probar.***
* ***Especificaciones de entrada.***
* ***Especificaciones de salida.***
* ***Necesidades del entorno.***
* ***Requisitos de procedimientos especiales.***
* ***Dependencias de interfaz.***

***Los procedimientos de prueba pueden incluir:***

* ***Identificación: Nombre de la prueba, descripción de la prueba y la fecha de finalización de la prueba.***
* ***Identificación de posibles problemas de implementación.***
* ***Identificación de la persona que completó los procedimientos de prueba.***
* ***Identificación de los requisitos previos.***
* ***Identificación de los pasos del procedimiento. Incluyendo el número de paso, la acción requerida por el probador y los resultados esperados.***

|  |  |  |  |
| --- | --- | --- | --- |
| ***Identificación del caso de prueba.*** | ***Elementos a probar.*** | ***Especificaciones de entrada.*** | ***Especificaciones de salida.*** |
| AutorTest.php | *Flujo de control para ejecutar la función* ***store*** *en el modelo de datos.* | $values = [  'autor' => 'Jose'  ]; | Con los parámetros dados, el caso de prueba regresa:    PASS Tests\Unit\AutorTest  ✓ autor créate |
| CategoriaTest.php | *Flujo de control para ejecutar la función* ***create*** *en el modelo de datos.* | $values = [  'categoria' => 'texto'  ]; | Con los parámetros dados, el caso de prueba regresa:  PASS Tests\Unit\CategoriaTest  ✓ categoria create |
| EditorialTest.php | *Flujo de control para ejecutar la función* ***create*** *en el modelo de datos.* | $values = [  'editorial' => 'Trillas'  ]; | Con los parámetros dados, el caso de prueba regresa:    PASS Tests\Unit\EditorialTest  ✓ editorial create |
| LibroTest.php | *Flujo de control para ejecutar la función* ***create*** *en el modelo de datos.* | $values = [  'editorial' => 'Trillas'  ]; | Con los parámetros dados, el caso de prueba regresa:    PASS Tests\Unit\LibroTest  ✓ libro |
| LoginTest.php | *Flujo de control para ejecutar la función* ***create*** *en el modelo de datos.* | $values = [  'email' => 'tux\_rios@hotmail.com',  'password' => '12345'  ]; | Con los parámetros dados, el caso de prueba regresa:  PASS Tests\Unit\LoginTest  ✓ login |
| UserTest.php | *Flujo de control para ejecutar la función* ***create*** *en el modelo de datos.* | $values = [  'nombre' => 'Jose',  'apellidos' => 'Perez',  'username' => 'pp',  'email' => 'inventado@hotmail.com',  'password' => '12345'  ]; | Con los parámetros dados, el caso de prueba regresa:  PASS Tests\Unit\UserTest  ✓ autor create |
| AutorControllerTest.php | *Flujo de control para ejecutar la función* ***store*** *en el modelo de datos.* | $values = [  'autor' => 'Robert Martin'  ]; | Con los parámetros dados, el caso de prueba regresa:  PASS Tests\Feature\AutorControllerTest  ✓ autor store |
| CategoriaControllerTest.php | *Flujo de control para ejecutar la función* ***store*** *en el modelo de datos.* | $values = [  'categoria' => 'programacion'  ]; | Con los parámetros dados, el caso de prueba regresa:  PASS Tests\Feature\CategoriaControllerTest  ✓ categoria store |
| EditorialControllerTest.php | *Flujo de control para ejecutar la función* ***store*** *en el modelo de datos.* | $values = [  'editorial' => 'Alianza'  ]; | Con los parámetros dados, el caso de prueba regresa:  PASS Tests\Feature\EditorialControllerTest  ✓ editorial store |
| LibroControllerTest.php | *Flujo de control para ejecutar la función* ***store*** *en el modelo de datos.* | $values = [  'ISBN' => '9780132350884',  'titulo' => 'Clean Code: A Handbook of Agile Software Craftsmanship',  'fechaDeLanzamiento' => '2008-08-10',  'autor' => 'Robert Martin',  'categoria' => 'programacion',  'editorial' => 'Alianza',  'idioma' => 'espanol',  'pagina' => '431',  'descripcion' => 'Even bad code can function. But if code isnt clean, it can bring a development organization to its knees. Every year, countless hours and significant resources are lost because of poorly written code. But it doesnt have to be that way.'  ]; | Con los parámetros dados, el caso de prueba regresa:  PASS Tests\Feature\LibroControllerTest  ✓ libro store |
| UserControllerTest.php | *Flujo de control para ejecutar la función* ***store*** *en el modelo de datos.* | $values = [  'nombre' => 'Jesus',  'apellidos' => 'Rios',  'username' => 'jR',  'email' => 'tux\_rios@hotmail.com',  'password' => '12345'  ]; | Con los parámetros dados, el caso de prueba regresa:  PASS Tests\Feature\UserControllerTest  ✓ user store |

***Necesidades del entorno.***

* *PHP ver. 8.1.1*
* *Apache ver. 2.0*
* *Soporte de Base de Datos a Acces y MySQL (Ver 15.1 Distrib 10.4.21-MariaDB)*

***Requisitos de procedimientos especiales.***

* *Los archivos probados debe ser estructurado en funciones con paso de parámetros y retorno de valores.*
* *La pruebas de funciones de inserción sólo se puede ejecutar una vez con los parámetros especificados porque el valor de la llave primaria no se puede repetir.*

***Dependencias de interfaz.***

* Base de datos sicenetx.mdb en Access y base de datos Biblioteca en MySQL

***Procedimientos de prueba:***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ***Nombre de la prueba*** | ***Tipo de prueba*** | ***Descripción de la prueba*** | ***Fecha*** | ***Tester*** | ***Problema e encontrado*** | ***Pasos*** |
| AutorTest.php | ***Unidad, inicial*** | ***El sistema permite Insertar un nuevo autor desde el modelo de datos.***  ***Para tal caso la prueba debe responder:***  ***PASS Tests\Unit\AutorTest***  ***✓ autor create*** | ***01/02/2022*** | ***JRA*** | ***Ninguno*** | ***1. Inicializar las variables.***  ***2. Llamar la función con el método*** assertEquals  $this->assertEquals('Jose', $autor->autor); |
| CategoriaTest.php | ***Unidad, inicial*** | ***El sistema permite Insertar una nueva categoría desde el modelo de datos.***  ***Para tal caso la prueba debe responder:***  ***PASS Tests\Unit\CategoriaTest***  ***✓ categoria create*** | ***01/02/2022*** | ***JRA*** | ***Ninguno*** | ***1. Inicializar las variables.***  ***2. Llamar la función con el método*** assertEquals  $this->assertEquals('texto', $categoria->categoria); |
| EditorialTest.php | ***Unidad, inicial*** | ***El sistema permite Insertar una nueva editorial desde el modelo de datos.***  ***Para tal caso la prueba debe responder:***  ***PASS Tests\Unit\EditorialTest***  ***✓ editorial create*** | ***02/02/2022*** | ***JRA*** | ***Ninguno*** | ***1. Inicializar las variables.***  ***2. Llamar la función con el método*** assertEquals  $this->assertEquals('Trillas', $editorial->editorial); |
| LibroTest.php | ***Unidad, inicial*** | ***El sistema permite Insertar un nuevo libro desde el modelo de datos.***  ***Para tal caso la prueba debe responder:***  ***PASS Tests\Unit\LibroTest***  ***✓ libro*** | ***02/02/2022*** | ***JRA*** | ***Ninguno*** | ***1. Inicializar las variables.***  ***2. Llamar la función con el método***  assertStatus  $this->get($url)  ->assertStatus(200); |
| LoginTest.php | ***Unidad, inicial*** | ***El sistema permite Validar a un usuario para ingresar al sistema.***  ***Para tal caso la prueba debe responder:***  ***PASS Tests\Unit\LoginTest***  ***✓ login*** | ***03/02/2022*** | ***JRA*** | ***Ninguno*** | ***1. Inicializar las variables.***  ***2. Llamar la función con el método*** assertStatus  $this->post($url, $values)  ->assertStatus(302); |
| UserTest.php | ***Unidad, inicial*** | ***El sistema permite Insertar un nuevo usuario desde el modelo de datos.***  ***Para tal caso la prueba debe responder:***  ***PASS Tests\Unit\UserTest***  ***✓ autor create*** | ***03/02/2022*** | ***JRA*** | ***Ninguno*** | ***1. Inicializar las variables.***  ***2. Llamar la función con el método*** assertEquals  $this->assertEquals('pp', $user->username); |
| AutorControllerTest.php | ***Unidad, inicial*** | ***El sistema permite Insertar un nuevo usuario utilizando el método Store desde el controlador.***  ***Para tal caso la prueba debe responder:***  ***PASS Tests\Feature\AutorControllerTest***  ***✓ autor store*** | ***04/02/2022*** | ***JRA*** | ***Ninguno*** | ***1. Inicializar las variables.***  ***2. Llamar la función con el método***  $this->post($url, $values)  ->assertStatus(302);    $this>assertDatabaseHas(Autor::class, $values); |
| CategoriaControllerTest.php | ***Unidad, inicial*** | ***El sistema permite Insertar una nueva categoría utilizando el método Store desde el controlador.***  ***Para tal caso la prueba debe responder:***  ***PASS Tests\Feature\CategoriaControllerTest***  ***✓ categoria store*** | ***04/02/2022*** | ***JRA*** | ***Ninguno*** | ***1. Inicializar las variables.***  ***2. Llamar la función con el método*** assertEquals  $this->post($url, $values)  ->assertStatus(302);  $this->assertDatabaseHas(Categoria::class, $values); |
| EditorialControllerTest.php | ***Unidad, inicial*** | ***El sistema permite Insertar una nueva editorial utilizando el método Store desde el controlador.***  ***Para tal caso la prueba debe responder:***  ***PASS Tests\Feature\EditorialControllerTest***  ***✓ editorial store*** | ***04/02/2022*** | ***JRA*** | ***Ninguno*** | ***1. Inicializar las variables.***  ***2. Llamar la función con el método*** assertEquals  $this->post($url, $values)  ->assertStatus(302);  $this->assertDatabaseHas(Editorial::class, $values); |
| LibroControllerTest.php | ***Unidad, inicial*** | ***El sistema permite Insertar un nuevo libro utilizando el método Store desde el controlador.***  ***Para tal caso la prueba debe responder:***  ***PASS Tests\Feature\LibroControllerTest***  ***✓ libro store*** | ***05/02/2022*** | ***JRA*** | ***Ninguno*** | ***1. Inicializar las variables.***  ***2. Llamar la función con el método*** assertDatabaseHas  $this->assertDatabaseHas(Libro::class, $values); |
| UserControllerTest.php | ***Unidad, inicial*** | ***El sistema permite Insertar un nuevo usuario utilizando el método Store desde el controlador.***  ***Para tal caso la prueba debe responder:***  ***PASS Tests\Feature\UserControllerTest***  ***✓ user store*** | ***05/02/2022*** | ***JRA*** | ***Ninguno*** | ***1. Inicializar las variables.***  ***2. Llamar la función con el método*** assertStatus  $this->post($url, $values)  ->assertStatus(302); |