public function iShouldHavePiecesOfProduct($pieces, $product)

{

PHPUnit\_Framework\_Assert::assertArraySubset(

[intval($product) => intval($pieces)],

$this->cart->getItems()

);

}

private function resetCart()

{

$this->cart = new Cart(['storage' => new FakeStorage()]);

}

}

5. Also, in the features/bootstrap/StorageContext.php file, add the following:

<?php

use Behat\Behat\Context\SnippetAcceptingContext; use book\cart\Cart;

use book\cart\features\bootstrap\storage\FakeStorage; use book\cart\storage\SessionStorage; use yii\di\Container; use yii\web\Application;

require\_once DIR . '/bootstrap.php';

class StorageContext implements SnippetAcceptingContext {

/\*\*

\* @var SessionStorage

\* \*/

private $storage;

/\*\*

\* @Given there is a clean storage \*/

public function thereIsACleanStorage()

{

$this->mockApplication();

$this->storage = new SessionStorage(['key' => 'test']);

}

/\*\*

\* @When I save :pieces of :product to the storage \*/

public function iSavePiecesOfProductToTheStorage($pieces, $product)

{

$this->storage->save([$product => $pieces]);

}

/\*\*

\* @Then I should have empty storage \*/

public function iShouldHaveEmptyStorage()

{

PHPUnit\_Framework\_Assert::assertCount(

0,

$this->storage->load()

);

}

/\*\*

\* @Then I should have :pieces of :product in the storage \*/

public function iShouldHavePiecesOfProductInTheStorage($pieces, $product)