1.Setting up your database.php

**'connections'** => [  
 **'sqlite\_testing'** => [  
 **'driver'** => **'sqlite'**,  
 **'database'** => **':memory:'**,  
 **'prefix'** => **''**,  
 ],

2.Add in your composer

**"codeception/codeception"**: **"2.0.\*@dev"**

**3.Setting up in your testcase.php**

**<?php  
  
use** Illuminate\Support\Facades\Artisan;  
**class** TestCase **extends** Illuminate\Foundation\Testing\TestCase  
{  
 */\*\*  
 \* The base URL to use while testing the application.  
 \*  
 \** ***@var*** *string  
 \*/* **protected $baseUrl** = **'http://kurio.co.id.app:8000'**;  
  
 */\*\*  
 \* Creates the application.  
 \*  
 \** ***@return*** *\Illuminate\Foundation\Application  
 \*/* **public function** createApplication()  
 {  
 *putenv*(**'DB\_DEFAULT=sqlite\_testing'**);  
 $app = **require *\_\_DIR\_\_***.**'/../bootstrap/app.php'**;  
  
 $app->make(Illuminate\Contracts\Console\Kernel::***class***)->bootstrap();  
  
 **return** $app;  
 }  
  
 **public function** setUp()  
 {  
 **parent**::*setUp*();  
 Artisan::*call*(**'migrate'**);  
 }  
  
 **public function** tearDown()  
 {  
 Artisan::*call*(**'migrate:reset'**);  
 **parent**::*tearDown*();  
 }  
}

**4. Making Model Factory for your table**

**<?php***/\*  
|--------------------------------------------------------------------------  
| Model Factories  
|--------------------------------------------------------------------------  
|  
| Here you may define all of your model factories. Model factories give  
| you a convenient way to create models for testing and seeding your  
| database. Just tell the factory how a default model should look.  
|  
\*/***use** Kurio\Web\Models\Job;  
  
$factory->define(Job::***class***, **function** (Faker\Generator $faker) {  
 **return** [  
 **'id'** => 1,  
 **'job\_title'** => $faker->**sentence**,  
 **'job\_desc'** => $faker->**paragraph**,  
 **'created\_by'** => 1,  
 **'created\_at'** => $faker->**date**(),  
 **'updated\_at'** => $faker->**date**()  
 ];  
});

**5.Set up your controller(in this case jobController)**

**<?php  
  
use** Illuminate\Foundation\Testing\WithoutMiddleware;  
**use** Illuminate\Foundation\Testing\DatabaseMigrations;  
**use** Illuminate\Foundation\Testing\DatabaseTransactions;  
**use** Kurio\Web\Repositories\Job\JobRepository;  
**use** Kurio\Web\Http\Controllers\JobController;  
**use** Kurio\Web\Models\Job;  
**use** Mockery **as** m ;  
  
**class** JobControllerTest **extends** TestCase  
{  
 **private $assets\_path** = **'assets/'**;  
 **public function** setUp()  
 {  
 **parent**::*setUp*();  
  
 $this->**data\_** = app()->make(**'Kurio\Web\Http\Controllers\JobController'**);  
 $this->**job**= app()->make(**'Kurio\Web\Repositories\Job\JobRepository'**);  
 $this->**job\_repo** = factory(Job::***class***);  
 }  
  
 **public function** testJobs()  
 {  
 $this->visit(**'/'**)  
 ->click(**'Jobs'**)  
 ->seePageIs(**'/jobs'**);  
 $this->assertResponseStatus(200);  
 }  
*/\* public function testIndex()  
 {  
 $data\_['html\_title'] = 'Job | Kurio - Jobs at Kurio';  
 $data\_['assets\_path'] = $this->assets\_path;  
 $data\_ = $this->job->getList();  
 $data\_['email\_copy'] = 'If this all sounds like you, then we look forward to your application. Please send us your resume and portfolio to ';  
 $data\_['email'] = 'join@kurio.co.id';  
  
 $this->get('/jobs')->assertViewHas('data\_', $data\_);  
 }  
\*/* **public function** testUpdate()  
 {  
 $model = $this->**job\_repo**->make();  
  
 $data = [  
 **'job\_title'** => **'job title'**,  
 **'job\_desc'** => **'job description'**,  
 **'created\_by'** => 1,  
 **'created\_at'** => *date*(**'Y-m-d H:i:s'**),  
 **'updated\_at'** => *date*(**'Y-m-d H:i:s'**),  
 ];  
  
 $model->**job\_title** = $data[**'job\_title'**];  
 $model->**job\_desc** = $data[**'job\_desc'**];  
 $model->**created\_at**=$data[**'created\_at'**];  
 $model->**created\_by** = $data[**'created\_by'**];  
 $model->**updated\_at**=$data[**'updated\_at'**];  
  
 $return = $model->save();  
  
 $expected = [  
 **'id'** => $model->**id**,  
 **'job\_title'** => $data[**'job\_title'**],  
 **'job\_desc'** => $data[**'job\_desc'**],  
 **'created\_by'** => $data[**'created\_by'**],  
 **'created\_at'** => $data[**'created\_at'**],  
 **'updated\_at'** => $data[**'updated\_at'**]  
 ];  
  
 $this->assertInstanceOf(Illuminate\Database\Eloquent\Model::***class***, $model);  
 $this->assertTrue($return);  
 $this->assertEquals($expected, $model->toArray());  
  
 }  
  
}

**If you have any kind of question can you comment! Viva**

# Code to test call with parameter.

public function testLoginPost(){

Session::start();

$response = $this->call('POST', '/login', [

'email' => 'badUsername[@gmail.com](https://laracasts.com/@gmail.com)',

'password' => 'badPass',

'\_token' => csrf\_token()

]);

$this->assertEquals(200, $response->getStatusCode());

$this->assertEquals('auth.login', $response->original->name());

}