**AUTOMATIZACIÓN DE PRUEBAS UNITARIAS DE INTERFAZ GRÁFICA DE USUARIO CON LAVAREL**



**JOSE UBALDO CARVAJAL**

**Estudiante maestría gestión y desarrollo de proyectos de software**

**Profesores**

**Sandra Victoria Hurtado Gil**

**Jorge Iván Meza Martinez**

**UNIVERSIDAD AUTÓNOMA DE MANIZALES**

**MAESTRIA EN GESTIÓN Y DESARROLLO DE PROYECTOS DE SOFTWARE**

**LINEA DE INVESTIGACIÓN EN CALIDAD DE SOFTWARE**

**ÉNFASIS EN DESARROLLO DE SOFTWARE**

**MANIZALES**

**2018**

Contenido

[Contenido 2](#_Toc522949844)

[1. INTRODUCCIÓN, PROPÓSITO DEL TRABAJO 5](#_Toc522949845)

[2. TECNOLOGÍAS/HERRAMIENTAS TÉCNICAS UTILIZADAS 6](#_Toc522949846)

[3. INTRODUCCIÓN LARAVEL DUSK 7](#_Toc522949847)

[4. DESCRIPCIÓN DEL PROYECTO A REALIZAR 9](#_Toc522949848)

[4.1. Historia de usuario “Matricular estudiante” 9](#_Toc522949849)

[4.2. Prototipo no funcional historia de usuario “Matricular estudiante” 10](#_Toc522949850)

[4.3. Modelo entidad relación sistema “Matrícula” 10](#_Toc522949851)

[5. PROCEDIMIENTO 11](#_Toc522949852)

[5.1. Instalar *Laravel*. 11](#_Toc522949853)

[5.2. Crear proyecto “matricula” con *Laravel*. 11](#_Toc522949854)

[5.3. Instalar Laravel Dusk para pruebas unitarias de GUI 13](#_Toc522949855)

[5.4. Ejecución de pruebas unitarias 14](#_Toc522949856)

[5.5. Creación de pruebas unitarias para el proyecto de matriculas 18](#_Toc522949857)

[5.5.1. Prueba unitaria para verificar que se carga el formulario de “Matricular estudiante” : 19](#_Toc522949858)

[5.5.2. Criterio aceptación 1: El nombre es requerido 20](#_Toc522949859)

[5.5.3. Criterio aceptación 2: La edad debe ser un valor entero positivo 21](#_Toc522949860)

[5.5.4. Criterio aceptación 3: Se filtran las ciudades según el departamento 22](#_Toc522949861)

[5.5.5. Criterio de aceptación 4: Se calcula descuento de matrícula según estrato del estudiante 23](#_Toc522949862)

[5.5.6. Criterio aceptación 5: Al guardar la matrícula se redirecciona al listado de matrículas registradas 25](#_Toc522949863)

[6. LISTA DE REFERENCIA 26](#_Toc522949864)

[7. ANEXOS 27](#_Toc522949865)

**Lista de Figuras**

Figura 1. GUI para historia de usuario “Matricula estudiante” 10

Figura 2. Modelo Entidad Relación para historia de usuario “Matricula estudiante” 10

Figura 3. Creación proyecto “matricula” con Laravel 11

Figura 4. Finalización en la creación proyecto “matricula” con Laravel 11

Figura 5. Estructura del proyecto “matricula” con Laravel 12

Figura 6. Inicio de servidor local en el proyecto “matricula” 12

Figura 7. Vista inicial del proyecto “matricula” 12

Figura 8. Instalación de Laravel Dusk. 13

Figura 9. Instalación de Laravel Dusk a nivel de proyecto. 13

Figura 10. Vista a probar 14

Figura 11. Prueba unitaria para página de bienvenida de un proyecto recién creado con Laravel. 14

Figura 12. Resultado de ejecución prueba unitaria para página de bienvenida de un proyecto recién creado con Laravel. 15

Figura 13. Configuración de parámetro APP\_URL para Dusk. 16

Figura 14. Resultado esperado de ejecución prueba unitaria para página de bienvenida de un proyecto recién creado con Laravel. 16

Figura 15. Instalación de “better PHPUnit” en Visual Studio Code. 17

Figura 16. Uso del plugin “Better PHPUnit” 17

Figura 17. Captura de GUI “Matricular estudiante” 18

Figura 18. Creación de clase de pruebas unitarias para “Matricular estudiante” 18

Figura 19. Prueba unitaria para verificar que se carga el formulario de “Matricular estudiante” 19

Figura 20. Prueba unitaria Criterio aceptación 1: El nombre es requerido. 20

Figura 21. Comportamiento GUI: Prueba unitaria Criterio aceptación 1: El nombre es requerido. 20

Figura 22. Prueba unitaria Criterio aceptación 2: La edad debe ser un valor entero positivo. 21

Figura 23. Comportamiento GUI: Prueba unitaria Criterio aceptación 2: La edad debe ser un valor entero positivo. 21

Figura 24. Prueba unitaria Criterio aceptación 3: Se filtran las ciudades según el departamento. 22

*Figura 25.* Comportamiento GUI: Prueba unitaria Criterio aceptación 3: Se filtran las ciudades según el departamento. 22

Figura 26. Prueba unitaria criterio de aceptación 4: Se calcula descuento según estrato del estudiante. 23

Figura 27. Comportamiento GUI: Prueba unitaria criterio de aceptación 4: Se calcula descuento según estrato del estudiante. 23

Figura 28. Prueba unitaria criterio de aceptación 4: Se muestra error para cálculos erróneos de descuentos en el valor de la matrícula. 24

Figura 29. Prueba unitaria Criterio aceptación 5: Se guarda la matrícula y se redirecciona a listado de matrículas registradas. 25

Figura 30. Comportamiento GUI: Prueba unitaria Criterio aceptación 5: Se guarda la matrícula y se redirecciona a listado de matrículas registradas. 25

**Lista de Tablas**

Tabla 1. Tecnologías/herramientas utilizadas. 6

Tabla 2. Historia de usuario “Matricular estudiante”. 9

1. INTRODUCCIÓN, PROPÓSITO DEL TRABAJO

En el siguiente trabajo se busca utilizar y documentar la herramienta *Dusk* que es una utilidad del framework *Laravel* para probar el comportamiento de *JavaScript*, esto permite automatizar las pruebas unitarias correspondientes al comportamiento de la interfaz gráfica de usuario.

PHPUnit ha sido la herramienta por preferencia para realizar test unitarios en PHP, no obstante, PHPUnit tiene la limitante que sólo puede probar comportamiento a nivel del Back-End y no se pueden realizar pruebas de interfaz gráfica de usuario (GUI) con PHPUnit. Para suplir este vacío, se creado Lavarel Dusk, que si permite probar aspectos de interactividad de usuario con la GUI, dado que está hecho para probar JavaScript en el navegador.

En este documento se registran los pasos necesarios para instalar y utilizar Laravel Dusk, adicionalmente, se implementó un proyecto que utiliza Laravel Dusk para automatizar las pruebas unitarias de una historia de usuario compuesta de 5 criterios de aceptación, por cada criterio de aceptación se implementó una prueba automatizada con Laravel Dusk y se documentó el proceso.

La implementación del proyecto sobre el que se aplicaron los conceptos y herramientas se puede encontrar en el repositorio Github: <https://github.com/joseucarvajal/matricula-laravel>

1. TECNOLOGÍAS/HERRAMIENTAS TÉCNICAS UTILIZADAS

|  |  |  |
| --- | --- | --- |
| **Tecnología/Herramienta** | **Versión** | **Fuente/Descarga** |
| PHP | 7.2.8 | <https://www.apachefriends.org/download.html> |
| XAMPP | 7.2.8 | <https://www.apachefriends.org/download.html> |
| Composer | 1.7.2 | https://getcomposer.org/download/ |
| Laravel | 5.6 | [https://laravel.com/docs/5.6/#server-requirements](https://laravel.com/docs/5.6/%23server-requirements) |
| Editor: Visual Studio Code | 1.26 | <https://code.visualstudio.com/> |
| Navegador: Google Chrome | 68.x | https://www.google.com/chrome/?brand=CHBD&gclid=EAIaIQobChMI5IKBh-n83AIVE6aaCh1W3wYJEAAYASAAEgKanvD\_BwE&gclsrc=aw.ds&dclid=CIqG-orp\_NwCFYxiwQodh2UAGw |

Tabla 1. Tecnologías/herramientas utilizadas.

1. INTRODUCCIÓN LARAVEL DUSK

Laravel Dusk es un framework que permite automatizar las pruebas del comportamiento de las aplicaciones en un navegador web. Esto ofrece la posibilidad probar la interactividad de los usuarios finales con la GUI (*Graphical User Interface*).

Laravel Dusk provee un API y un modelo de objetos para realizar pruebas sobre una instancia de un navegador web. El objeto principal para codificar las pruebas se denomina “$browser” y corresponde a una instancia de navegador que ejecuta Laravel Dusk para probar la interactividad con el usuario final. El objeto $browser permite interactuar con los elementos de la GUI:

1. public function test\_la\_edad\_del\_estudiante\_debe\_ser\_mayor\_a\_cero\_ok()
2. {
3. $this->browse(function (Browser $browser) {
4. $browser->visit('matricula/matriculas/create')
5. ->type("edad", "-1")
6. ->press("Guardar")
7. ->AssertSee("El campo edad debe ser mínimo 1");
8. });
9. }

El código anterior corresponde a una prueba unitaria que utiliza el objeto $browser de Laravel Dusk de la siguiente manera:

En la línea 3 se está inicializando una nueva instancia del objeto $browser.

En la línea 4 se utiliza el método visit(url) del objeto $browser para navegar hacia una URL específica.

En la línea 5 se utiliza el método type(nombre campo del formulario HTML, valor) para establecer el valor -1 en el campo del formulario HTML con name=”edad”.

En la línea 6 se utiliza el método press(nombre el botón) para presionar el botón con name=”Guardar” del formulario HTML.

Finalmente, en la línea 7, el método AssertSee(Texto a buscar en el DOM del navegador) verifica que en cualquier parte del DOM de la página cargada aparezca el texto “El campo edad debe ser mínimo -1”.

El objeto $browser también dispone del método AssertSeeIn(Selector CSS, Texto a buscar en el DOM del navegador). Este método permite realizar verificaciones más precisas de valores en el DOM, dado que no busca el string en todo el documento, sino solamente en el elemento HTML que coincida con el selector CSS. Ejemplo:

$browser-> AssertSeeIn("select[name=idCiudad]", "Zipaquirá")

En el ejemplo anterior, se busca el texto “Zipaquirá” en el elemento del formulario cuyo name=”ciudad”, esto permite realizar verificaciones más precisas, dado que el texto “Zipaquirá” podría estar en otra parte del DOM diferente al campo del formulario con name igual a ciudad. Si se utiliza el método AssertSee produciría un resultado erróneo, puesto que este último busca en todo el DOM.

1. DESCRIPCIÓN DEL PROYECTO A REALIZAR

Con fines ilustrativos, se creará un proyecto básico para aplicar las pruebas de interfaz de usuario con Dusk.

Para fines prácticos y académicos este proyecto consiste en un sistema de matrícula básico, con una sola historia de usuario que será objeto de las pruebas unitarias.

* 1. Historia de usuario “Matricular estudiante”

La siguiente historia de usuario se realiza sólo con efectos académicos y se adapta con el fin de aplicar pruebas unitarias automatizadas de GUI (*Graphical user interface*). La historia de usuario que se describe a continuación no corresponde a un proceso real de matrícula de estudiantes.

En la siguiente tabla se detalla el comportamiento de la historia de usuario. Las pruebas unitarias de la interfaz gráfica de usuario se obtendrán a partir de los criterios de aceptación especificados.

|  |  |
| --- | --- |
| **Matricular estudiante** | |
| **Descripción** | **Criterios de aceptación** |
| Un usuario se dispone a matricula un estudiante, para lo cual debe ingresar la siguiente información:   1. Nombre: Corresponde al nombre del estudiante. (Requerido). 2. Documento: Es el número de identificación del estudiante. (Requerido). 3. Edad: Corresponde a la edad del estudiante. (No Requerido). 4. Departamento: Departamento de donde es originario el estudiante. (Requerido). 5. Ciudad: Ciudad de donde es originario el estudiante. (Requerido). 6. Estrato: Corresponde al estrato (1, 2…6) del estudiante. | 1. El nombre es requerido. 2. La edad del estudiante no es requerida, sin embargo, si se ingresar un valor, se debe validar que este valor sea un número entero positivo. 3. La ciudad se debe filtrar y debe corresponder según el departamento seleccionado. Ej: si se escoge el departamento de Caldas, sólo deben mostrarse las ciudades correspondientes a este departamento. 4. Según el estrato seleccionado, el estudiante tendrá descuentos en el valor de la matrícula de la siguiente manera:    1. Estrato 1: 30% de descuento    2. Estrato 2: 20% de descuento    3. Estrato 3: 10% de descuento    4. Para los demás estratos, no se tendrá ningún descuento. 5. Al guardar una matrícula exitosamente, se debe redireccionar al listado de matrículas almacenadas. |

Tabla 2. Historia de usuario “Matricular estudiante”.

* 1. Prototipo no funcional historia de usuario “Matricular estudiante”

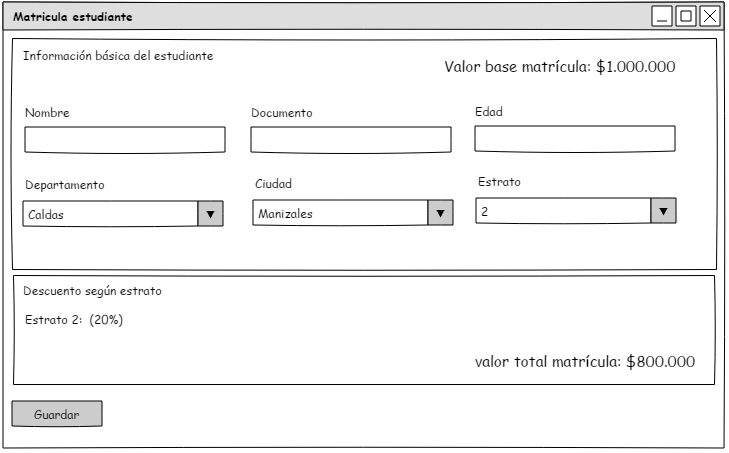


Figura 1. GUI para historia de usuario “Matricula estudiante”

* 1. Modelo entidad relación sistema “Matrícula”

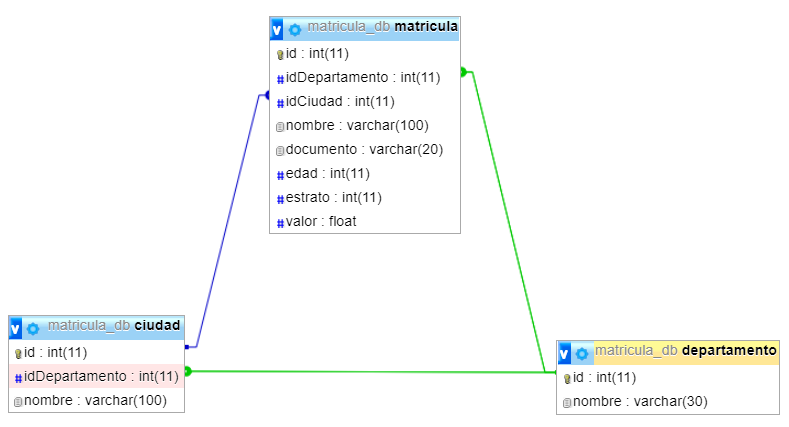


Figura 2. Modelo Entidad Relación para historia de usuario “Matricula estudiante”

1. PROCEDIMIENTO

Para automatizar las pruebas unitarias de interfaz de usuario en Lavarel, se siguió el siguiente procedimiento:

* 1. Instalar *Laravel*.

Para instalar *Laravel*, se utilizó la guía de instalación oficial: <https://laravel.com/docs/5.6/installation>.

* 1. Crear proyecto “matricula” con *Laravel*.

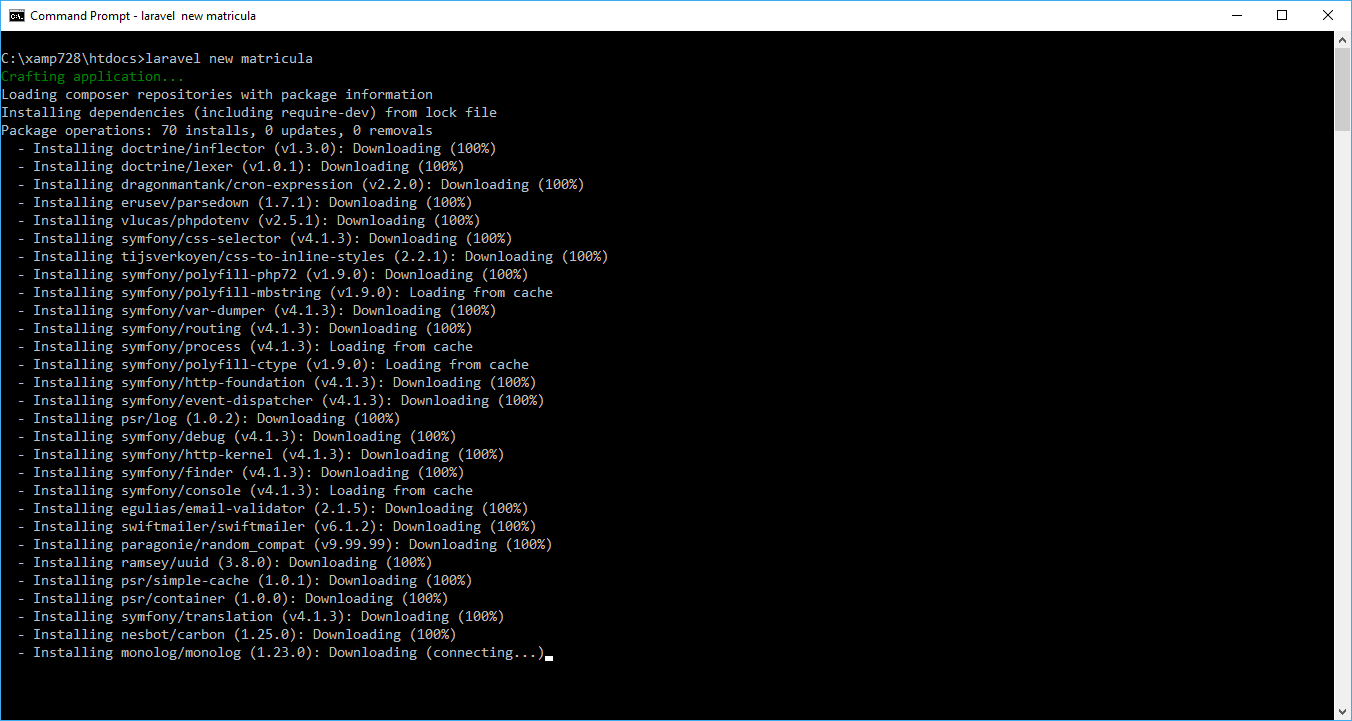


Figura 3. Creación proyecto “matricula” con Laravel

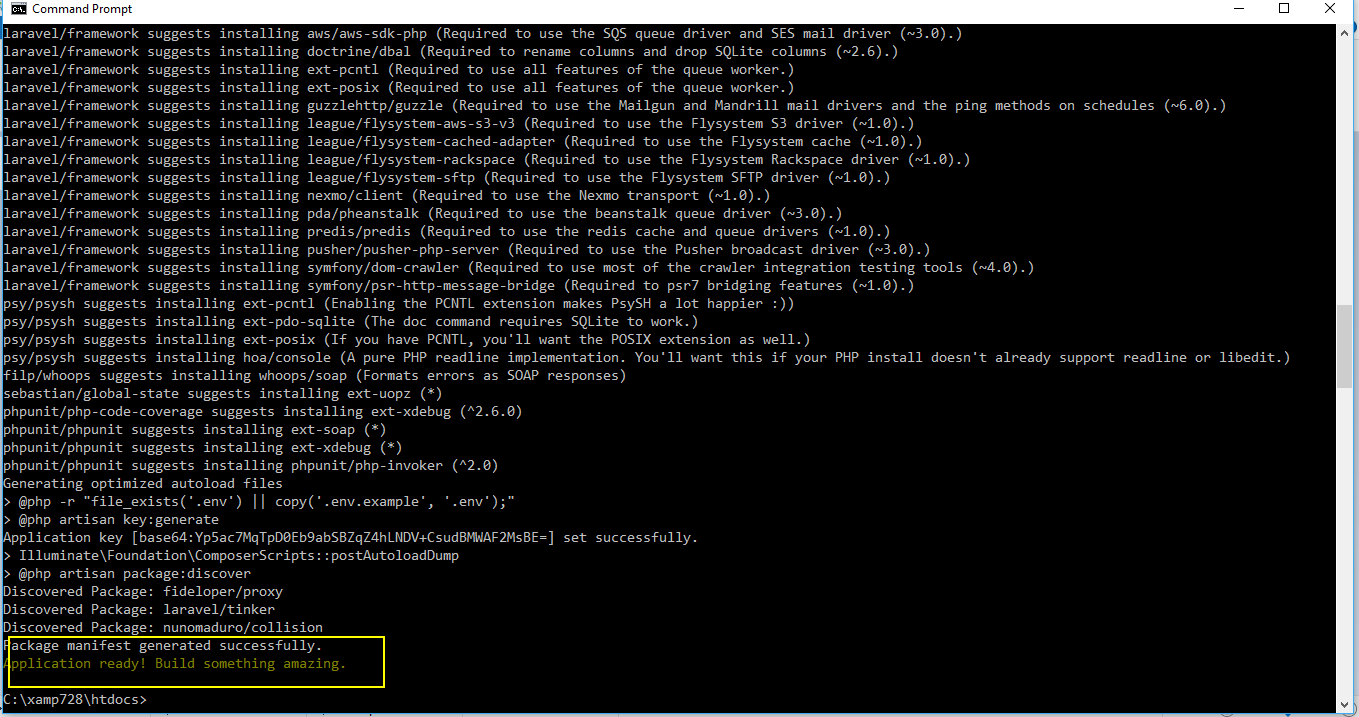


Figura 4. Finalización en la creación proyecto “matricula” con Laravel

Estructura del proyecto creado, Se editará con Visual Studio Code.

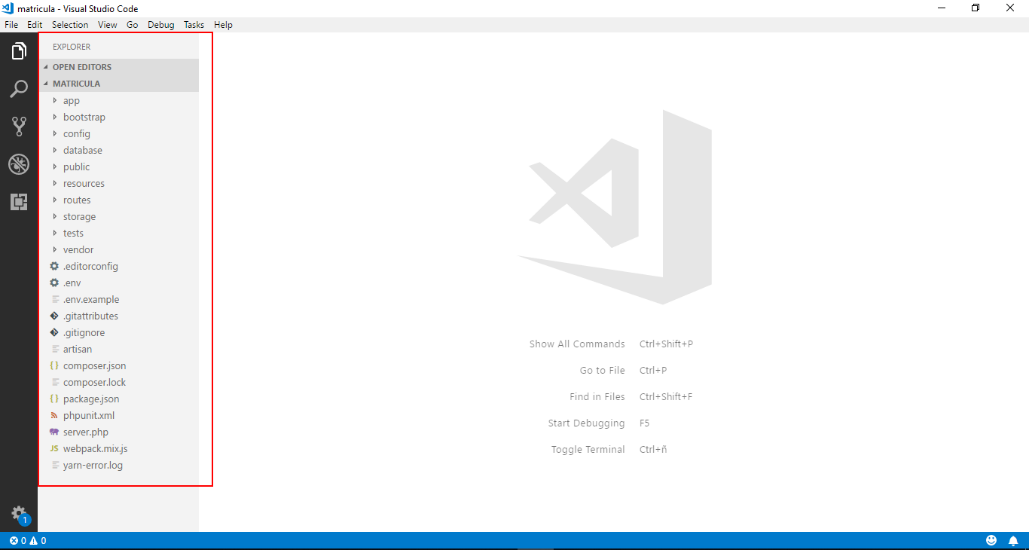


Figura 5. Estructura del proyecto “matricula” con Laravel

Una vez creado el proyecto, se inicia el servidor local del mismo con: **php artisan serve**

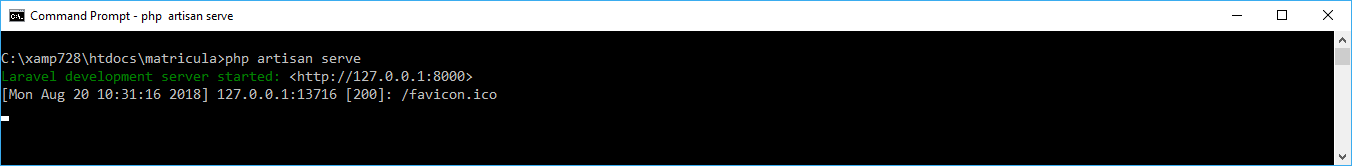


Figura 6. Inicio de servidor local en el proyecto “matricula”

Al navegar a <http://localhost:8000/>, el proyecto debería verse de la siguiente manera:

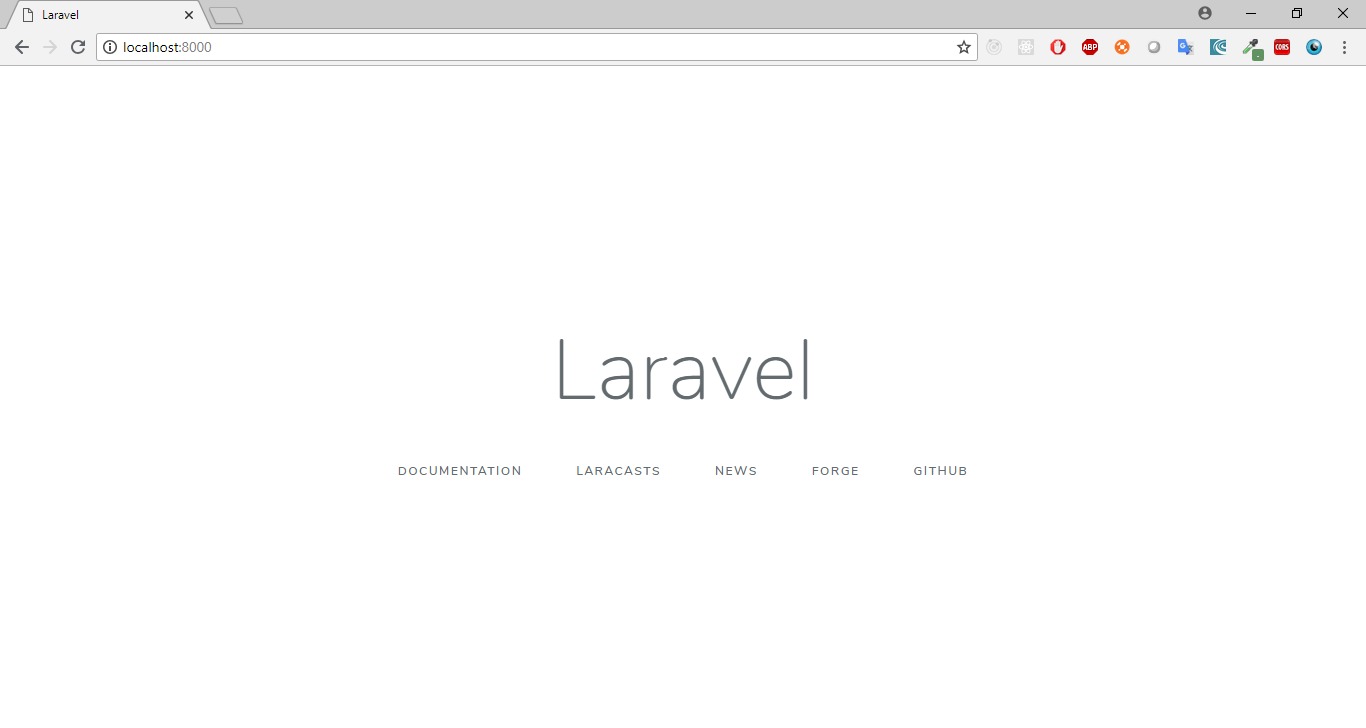


Figura 7. Vista inicial del proyecto “matricula”

* 1. Instalar Laravel Dusk para pruebas unitarias de GUI

Para instalar Laravel Dusk, se siguió la documentación disponible en: [https://laravel.com/docs/5.6/dusk#installation](https://laravel.com/docs/5.6/dusk%23installation). Desde el directorio del proyecto se ejecuta el comando: **composer require --dev laravel/dusk**

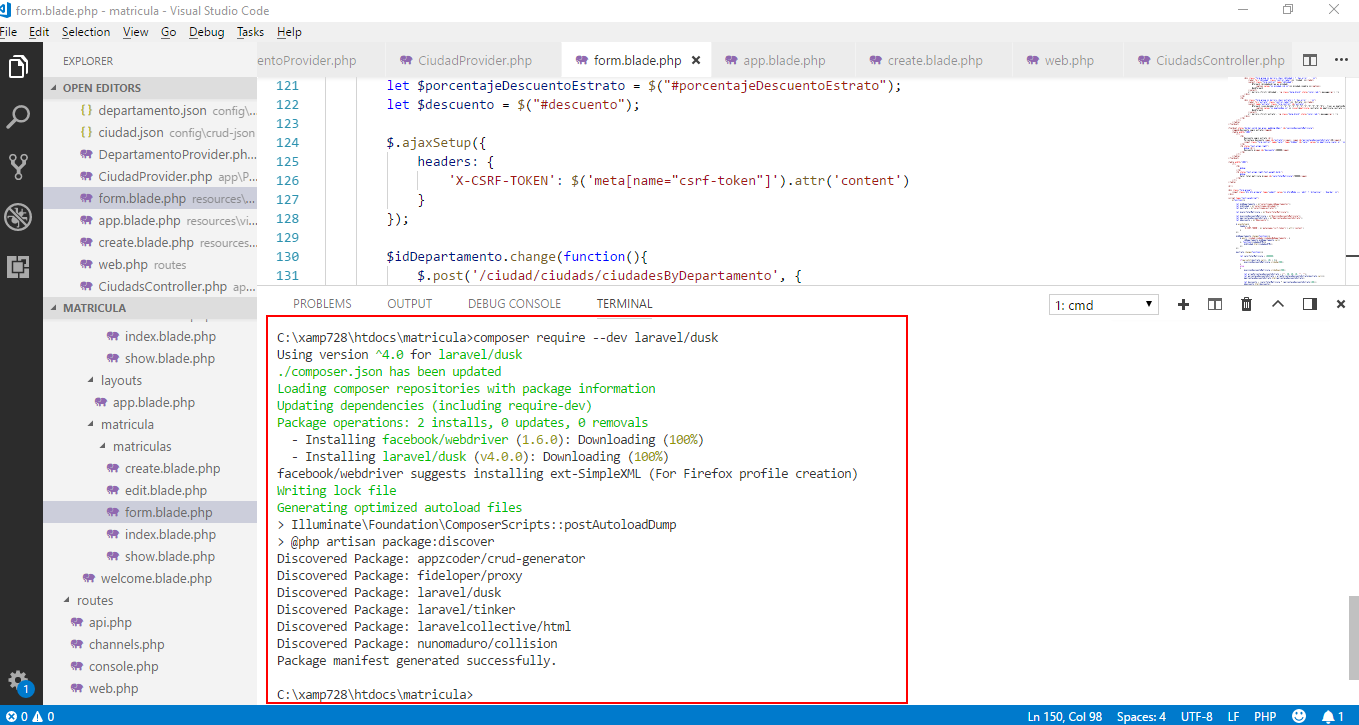


Figura 8. Instalación de Laravel Dusk.

Posteriormente se installa Dusk para el proyecto específicamente con el comando: **php artisan dusk:install**. Una vez instalado Dusk, este modifica la estructura del proyecto, creando la carpeta: tests/Browser y el archivo DuskTestCase.php.

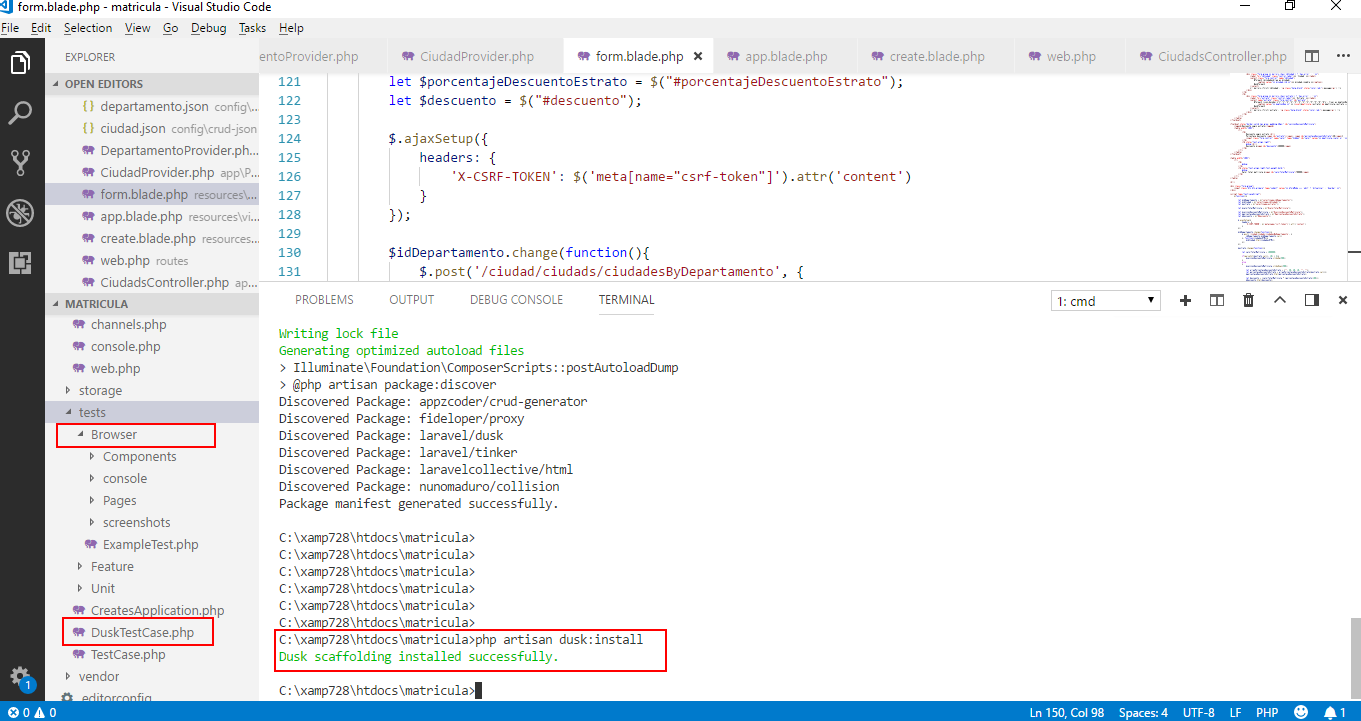


Figura 9. Instalación de Laravel Dusk a nivel de proyecto.

* 1. Ejecución de pruebas unitarias

Posterior a la instalación de Dusk, se proceden a crear y ejecutar los test unistarios.

Uno de los archivos que se crean al instalar Dusk en el proyecto es tests/Browser/ExampleTest.php. Este archivo contiene una prueba unitaria básica que verifica se imprima la palabra “Laravel” cuando se visita la página principal de un proyecto recién creado.

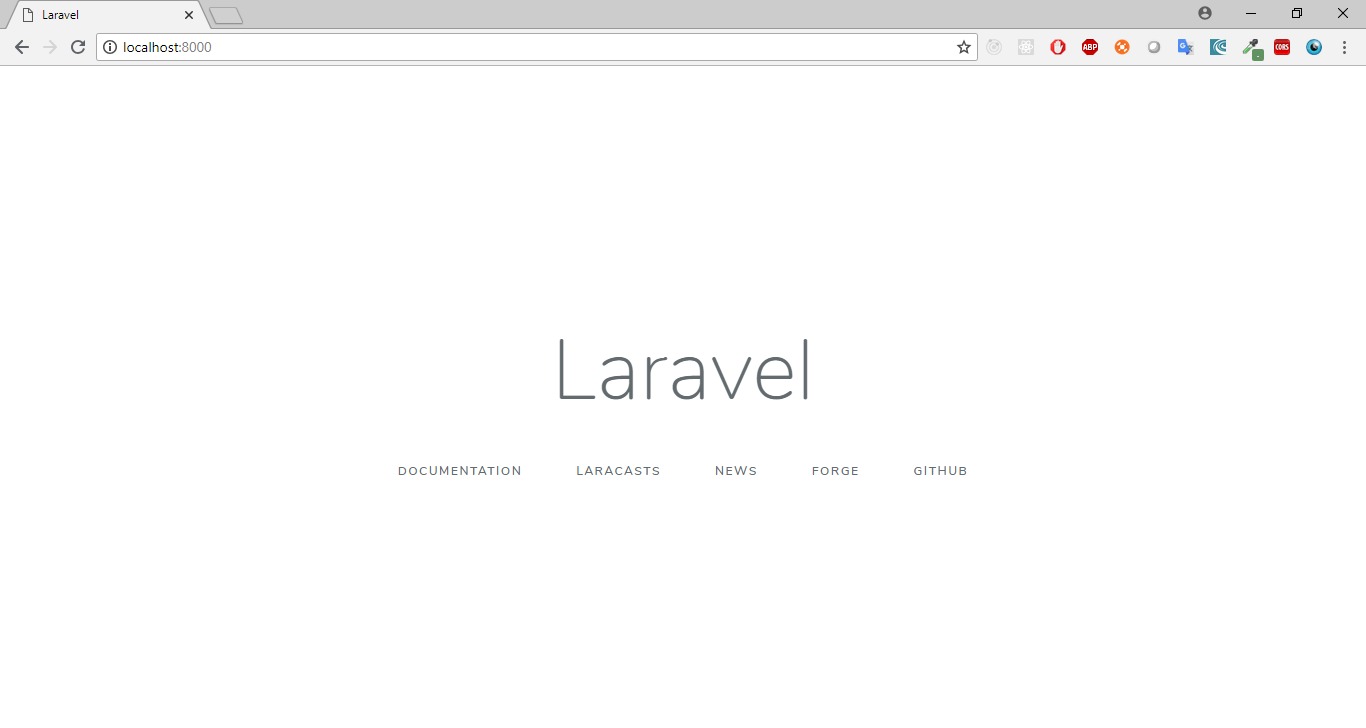


Figura 10. Vista a probar

Para probar la vista anterior, en el archivo tests/Browser/ExampleTest.php ha creado la siguiente prueba unitaria durante su instalación:

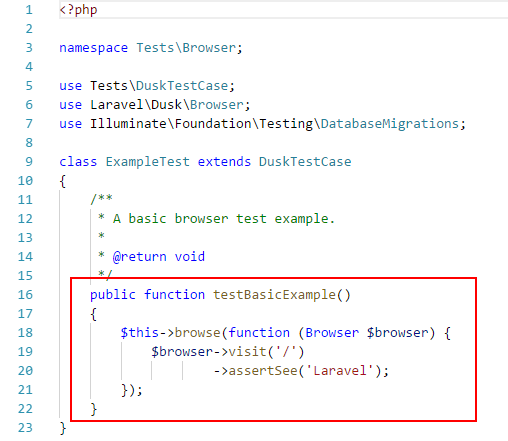


Figura 11. Prueba unitaria para página de bienvenida de un proyecto recién creado con Laravel.

En la prueba unitaria anterior, se está cargando la URL “/” que corresponde a la página de bienvenida de un proyecto recién creado con Laravel. La línea 20 verifica que exista el texto “Laravel” en la página visitada.

Para ejecutar la prueba anterior, se corre el comando: **php artisan dusk**. La ejecución de la prueba muestra el siguiente mensaje:

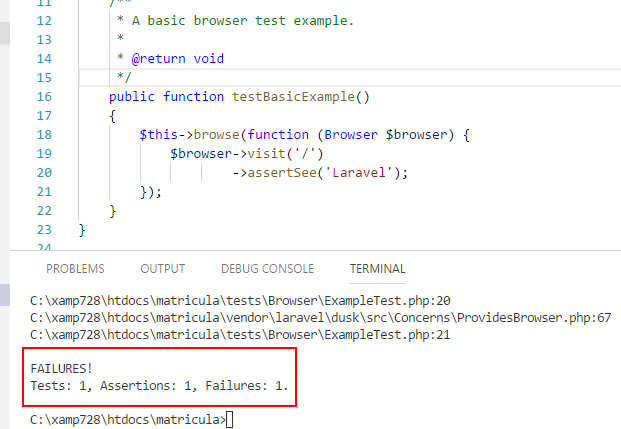


Figura 12. Resultado de ejecución prueba unitaria para página de bienvenida de un proyecto recién creado con Laravel.

El mensaje anterior indica que falta algo por configurar, dado que, al visitar la página de bienvenida, se obtiene el texto “Laravel”. Hay una razón para esto.

El comportamiento erróneo se obtiene porque Dusk requiere la instancia de un navegador para ejecutar las pruebas unitarias. Esta instancia debe ejecutarse en la misma URL y puerto que se ejecuta el proyecto Laravel. Para esto se debe editar el archivo .env con la URL exacta en que corre el servidor de desarrollo de Laravel.

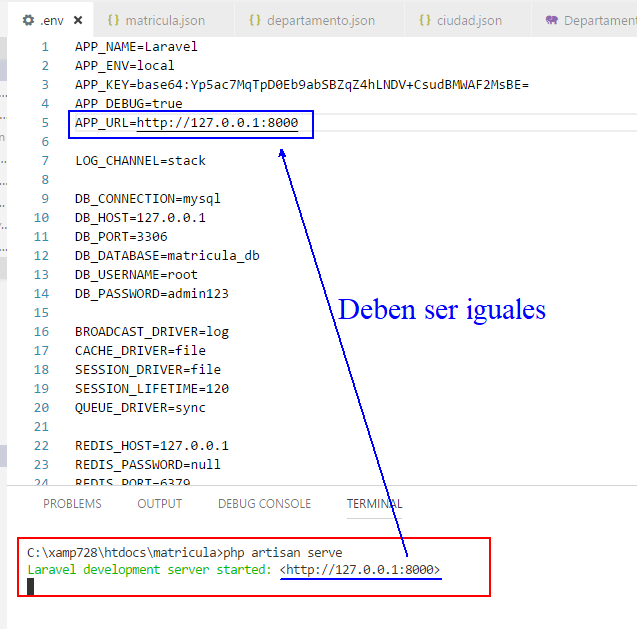


Figura 13. Configuración de parámetro APP\_URL para Dusk.

Al ejecutar de nuevo las pruebas unitarias mediante el comando **php artisan dusk** se obtiene el resultado esperado sin errores:

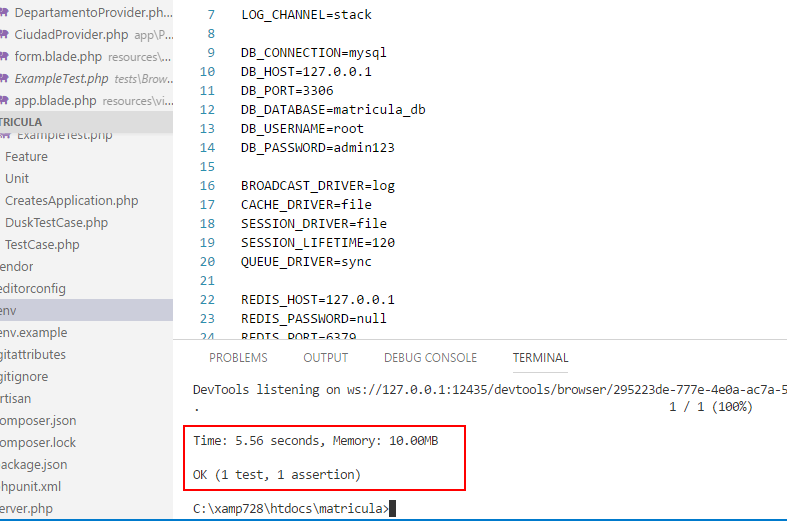


Figura 14. Resultado esperado de ejecución prueba unitaria para página de bienvenida de un proyecto recién creado con Laravel.

Una utilidad interesante para ahorrar tiempo a la hora de ejecutar pruebas es instalar la utilidad “better PHPUnit” para el editor Visual Studio Code:

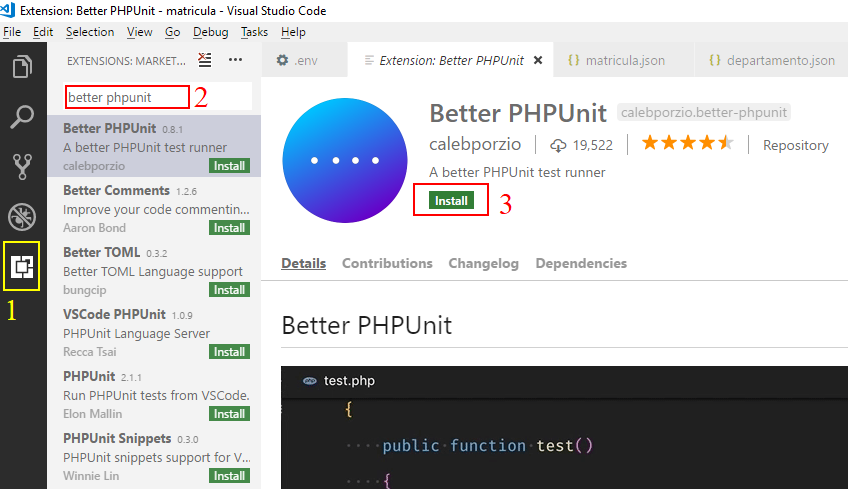


Figura 15. Instalación de “better PHPUnit” en Visual Studio Code.

Una vez instalado el plugin para Visual Studio Code, sólo es necesario presionar **Ctrl + Shift + P** con el cursor sobre el archivo de pruebas unitarias y luego se selecciona “Better PHPUnit: run” .

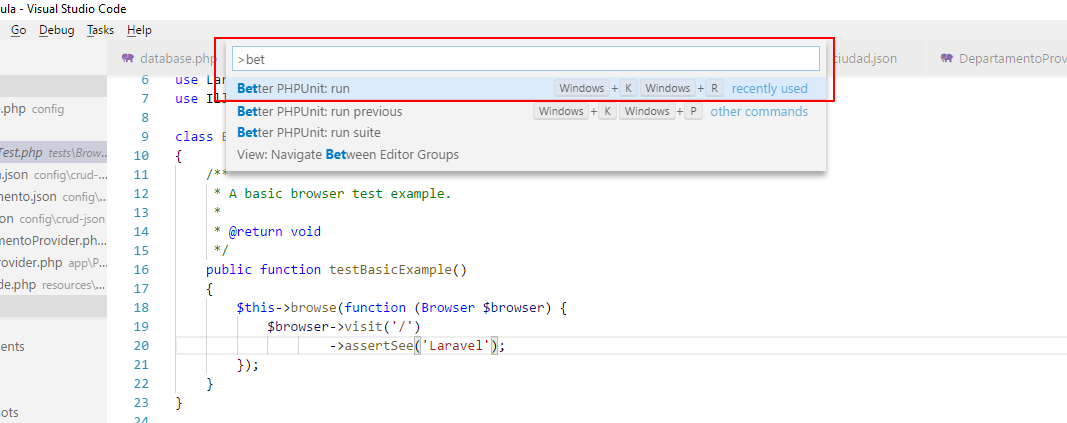


Figura 16. Uso del plugin “Better PHPUnit”

* 1. Creación de pruebas unitarias para el proyecto de matriculas

Ahora que se ha instalado Dusk y se han ejecutado las pruebas que vienen por defecto en la instalación, se procede a crear pruebas específicas para el proyecto.

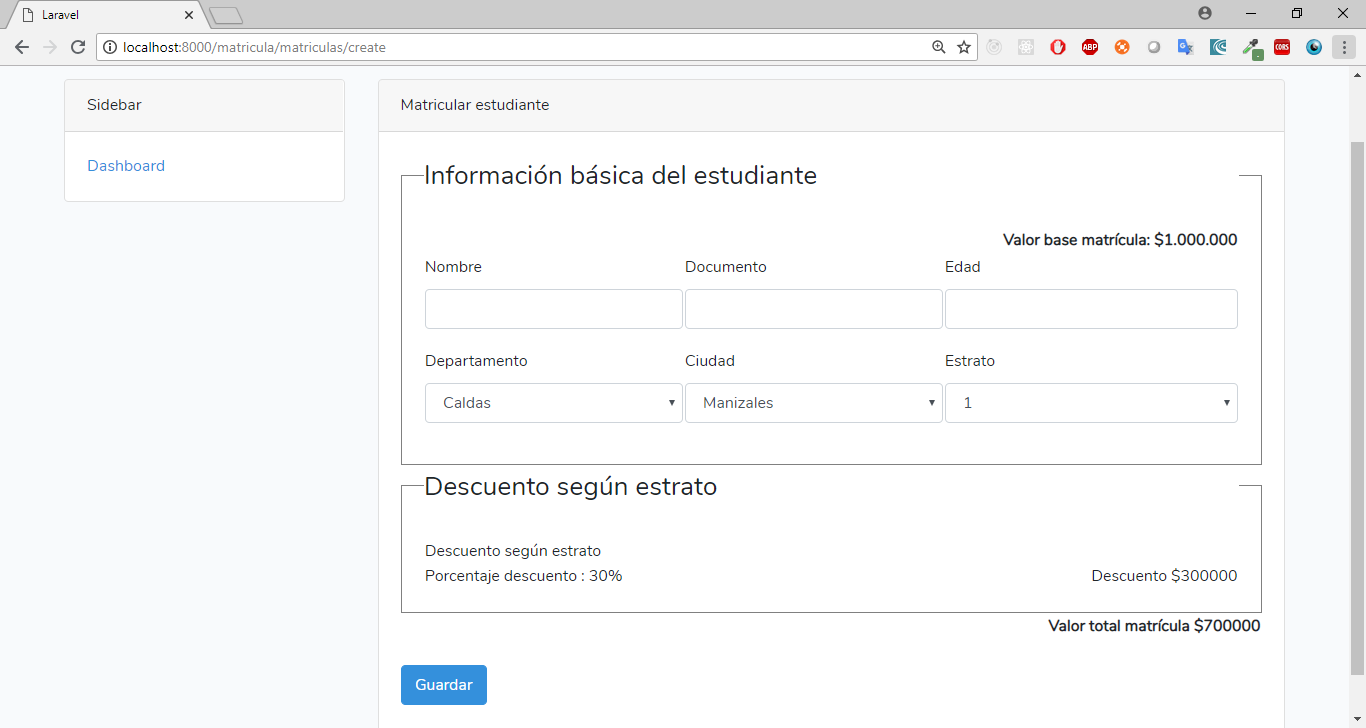


Figura 17. Captura de GUI “Matricular estudiante”

Una vez creada la funcionalidad especificada en el numeral 3.1 “Historia de usuario "Matricular estudiante"”, se proceden a crear las pruebas unitarias para asegurar los criterios de aceptación de la historia de usuario. Para crear la prueba unitaria se ejecuta el comando: **php artisan dusk:make [NombreClasePruebaUnitaria]**.

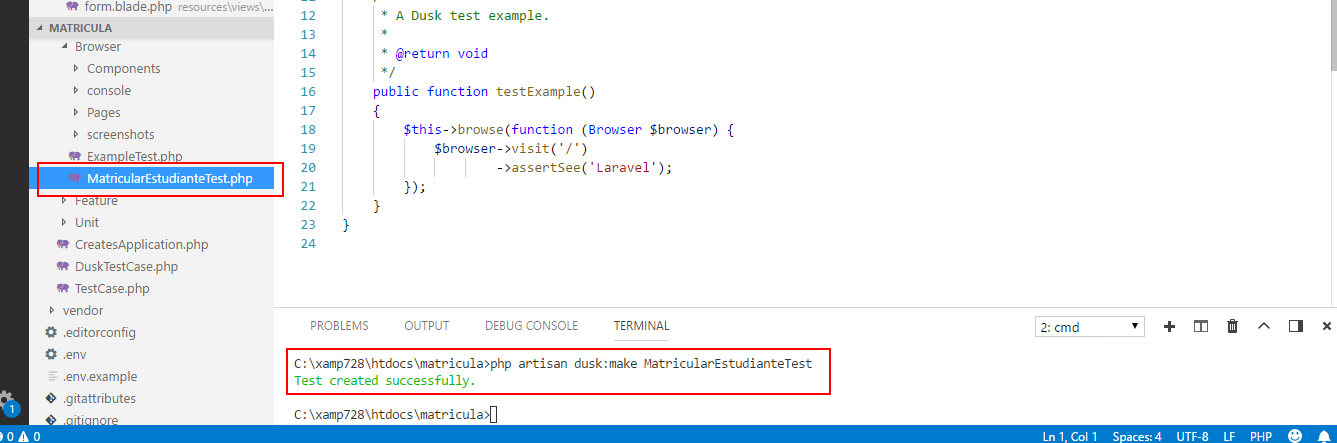


Figura 18. Creación de clase de pruebas unitarias para “Matricular estudiante”

**Nota**: Es importante aclarar que el nombre del método de prueba unitaria debe empezar por “**test**” para que Dusk lo tome con un método válido de prueba unitaria.

* + 1. Prueba unitaria para verificar que se carga el formulario de “Matricular estudiante” :

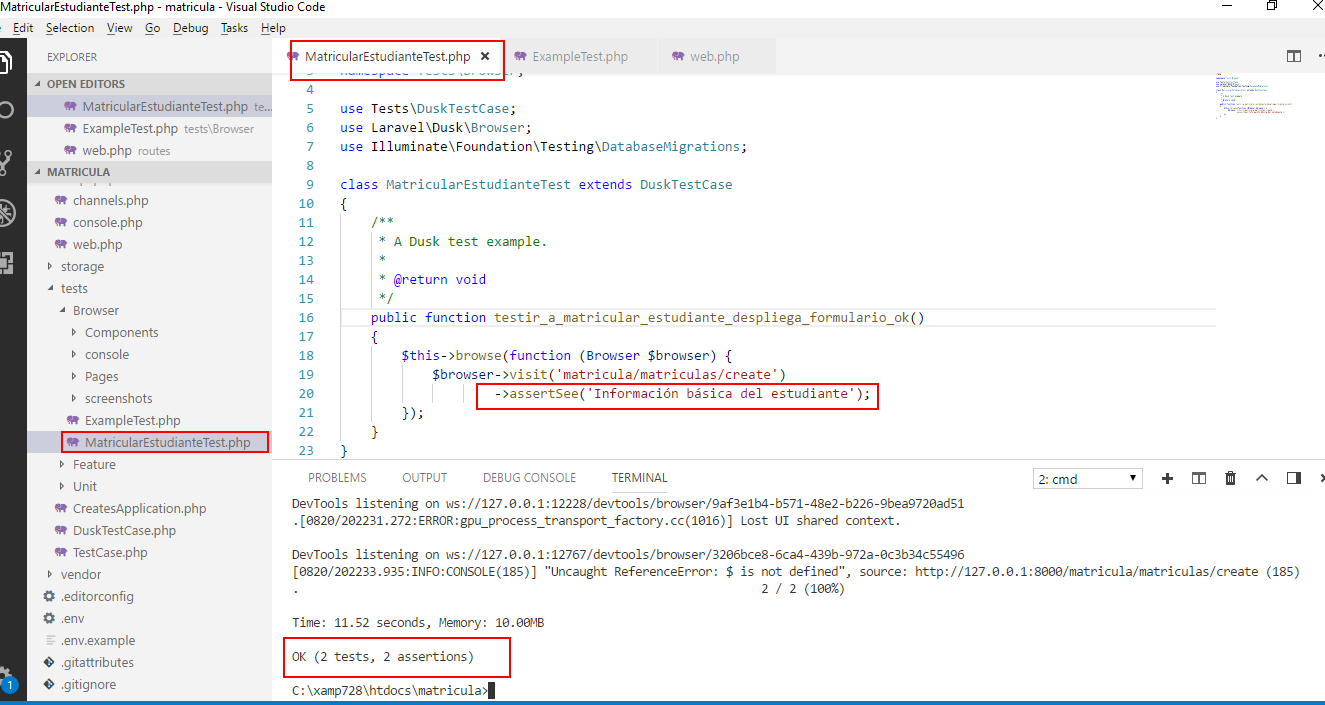


Figura 19. Prueba unitaria para verificar que se carga el formulario de “Matricular estudiante”

Como se puede observar en la prueba unitaria, al visitar la ruta “matricula/matriculas/create”, se verifica que cargue el texto “Información básica del estudiante”. El resultado de la prueba es (2 tests, 2 assertions), el primer test es la prueba unitaria que viene por defecto al instalar Dusk y el segundo test, fue el que recién se creó para matricular estudiante.

Ahora que se ha verificado mediante test se puede navegar a la opción de “Matricular estudiante”, se proceden a ingresar valores en los campos del formulario desde la prueba unitaria:

* + 1. Criterio aceptación 1: El nombre es requerido



Figura 20. Prueba unitaria Criterio aceptación 1: El nombre es requerido.

El método AssertSeeIn (línea 32), busca en un elemento con el selector CSS de clase: “**.help-block**”.

Comportamiento GUI:

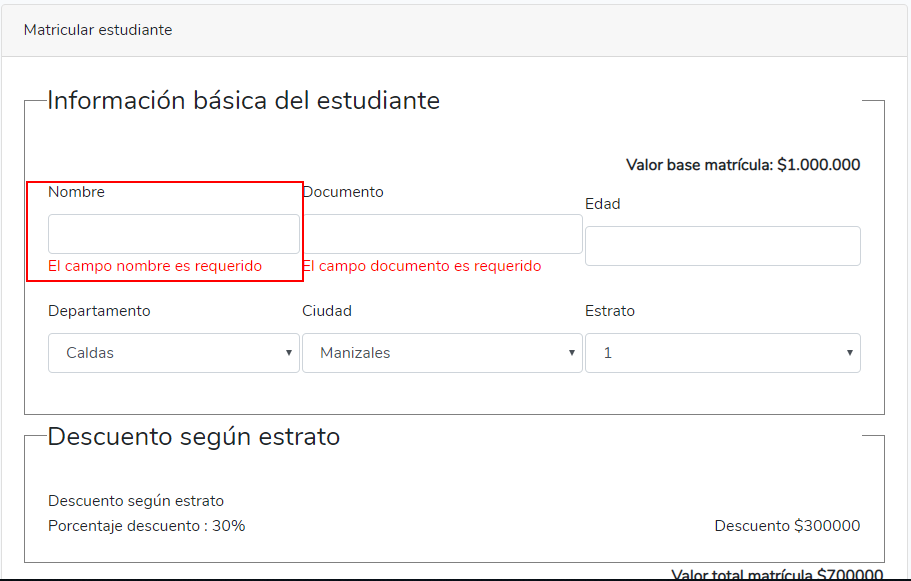


Figura 21. Comportamiento GUI: Prueba unitaria Criterio aceptación 1: El nombre es requerido.

* + 1. Criterio aceptación 2: La edad debe ser un valor entero positivo



Figura 22. Prueba unitaria Criterio aceptación 2: La edad debe ser un valor entero positivo.

Comportamiento GUI:

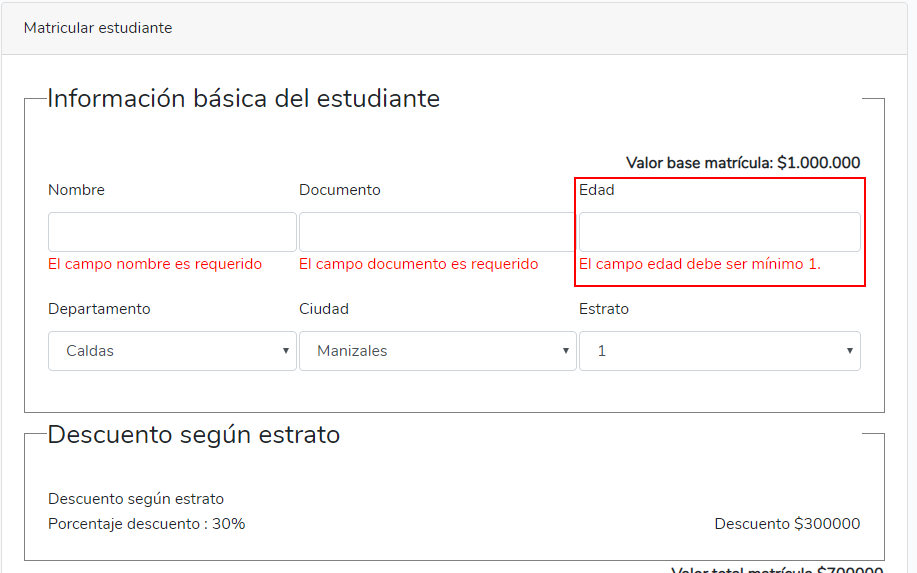


Figura 23. Comportamiento GUI: Prueba unitaria Criterio aceptación 2: La edad debe ser un valor entero positivo.

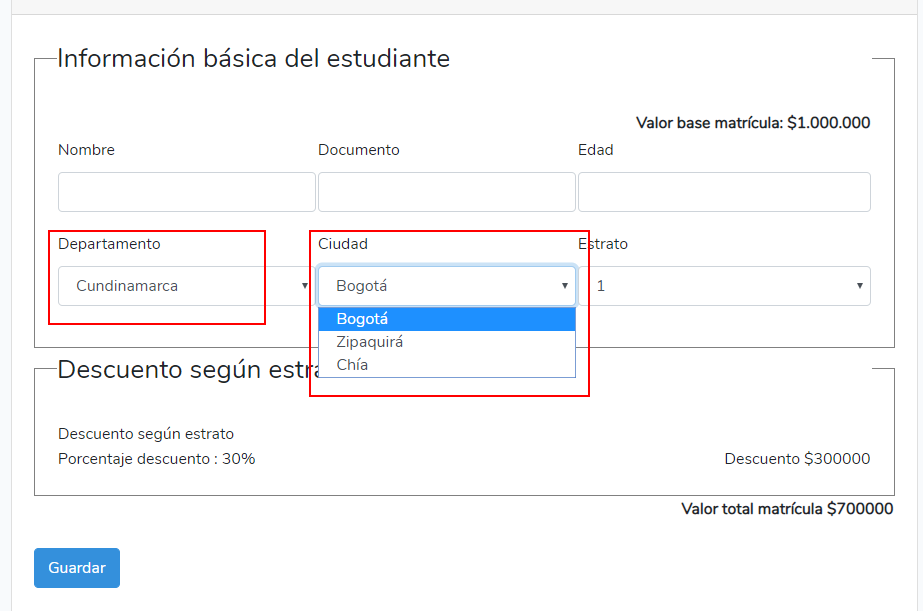
* + 1. Criterio aceptación 3: Se filtran las ciudades según el departamento



Figura 24. Prueba unitaria Criterio aceptación 3: Se filtran las ciudades según el departamento.

Al seleccionarse el departamento con id=2 (Cundinamarca), se muestran sólo sus ciudades. Esta prueba unitaria contiene 3 aserciones, una por cada ciudad a verificar .

Comportamiento GUI:



*Figura 25.* Comportamiento GUI: Prueba unitaria Criterio aceptación 3: Se filtran las ciudades según el departamento.

* + 1. Criterio de aceptación 4: Se calcula descuento de matrícula según estrato del estudiante



Figura 26. Prueba unitaria criterio de aceptación 4: Se calcula descuento según estrato del estudiante.

En esta prueba, se espera que la GUI se muestre el texto “Descuento $100000” que corresponde al 10% de la matrícula del estudiante de estrato 3. Al ejecutar esta secuencia manualmente en la GUI, se tiene

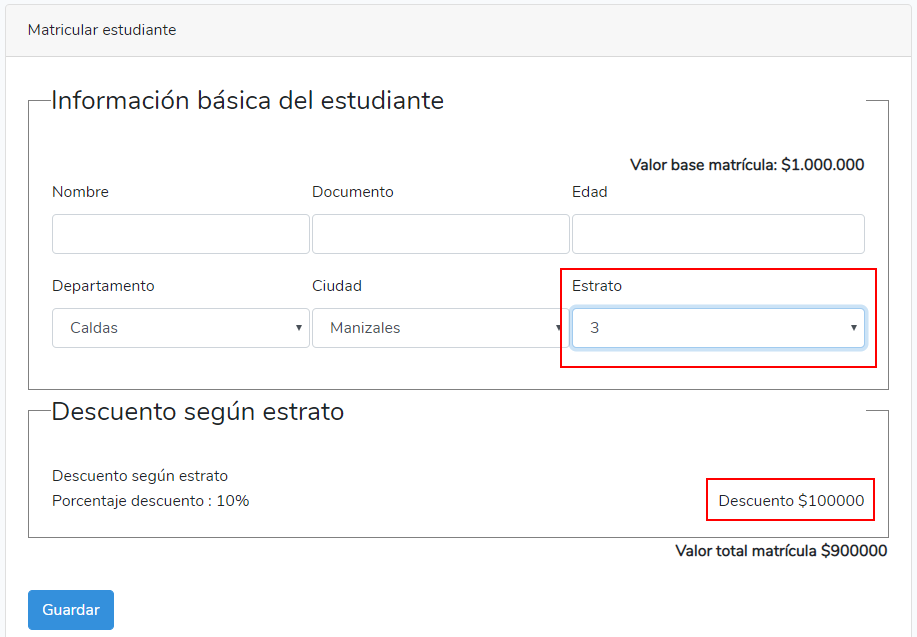


Figura 27. Comportamiento GUI: Prueba unitaria criterio de aceptación 4: Se calcula descuento según estrato del estudiante.

Si el resultado esperado fuese diferente a $100000 para el descuento, Dusk informaría el error:



Figura 28. Prueba unitaria criterio de aceptación 4: Se muestra error para cálculos erróneos de descuentos en el valor de la matrícula.

* + 1. Criterio aceptación 5: Al guardar la matrícula se redirecciona al listado de matrículas registradas



Figura 29. Prueba unitaria Criterio aceptación 5: Se guarda la matrícula y se redirecciona a listado de matrículas registradas.

Comportamiento GUI:

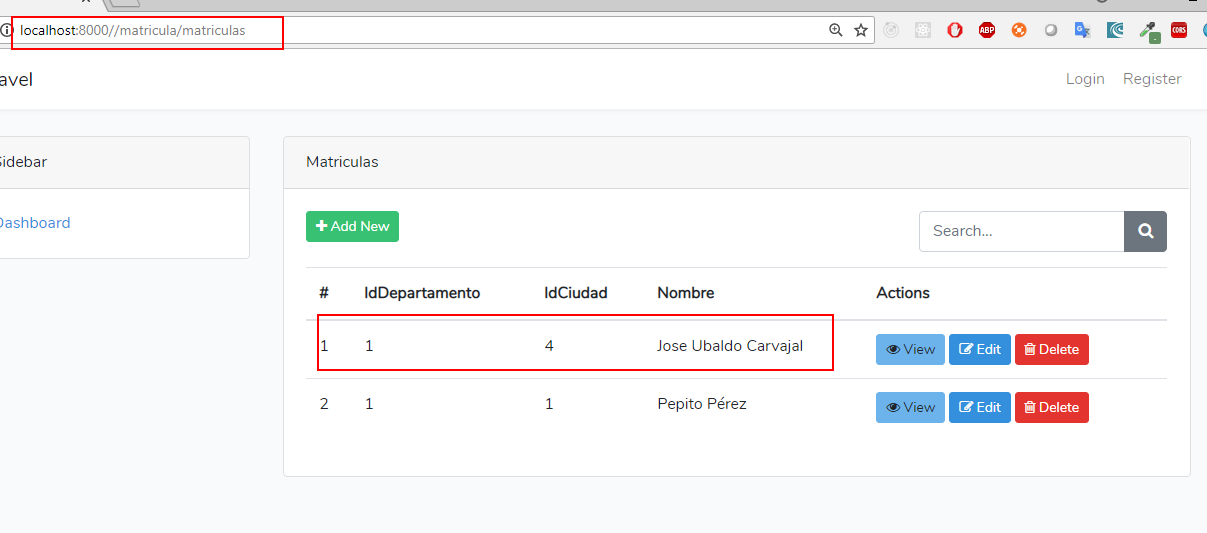


Figura 30. Comportamiento GUI: Prueba unitaria Criterio aceptación 5: Se guarda la matrícula y se redirecciona a listado de matrículas registradas.

1. LISTA DE REFERENCIA

Laravel 5.6 Official documentation. (2018). Installation. Recuperado de <https://laravel.com/docs/5.6/installation>.

Laravel 5.6 Official documentation. (2018). Browser Tests (Laravel Dusk). Recuperado de <https://laravel.com/docs/5.6/dusk>.

Laravel Dusk Tutorials (Youtube). (2018). Introduction and Installation #1. Recuperado de <https://www.youtube.com/watch?v=YjvSyWInHeE>.

Laravel Dusk Tutorials (Youtube). (2018). Setup Laravel Dusk with Database #2. Recuperado de <https://www.youtube.com/watch?v=7s6NY74m1pA>.

1. ANEXOS

ANEXO 1. Github de proyecto Laravel matrícula: <https://github.com/joseucarvajal/matricula-laravel>