UNIVERSIDAD TÉCNICA DE BABAHOYO

FACULTAD DE ADMINISTRACIÓN, FINANZAS INFORMÁTICA

F.A.F.I.

"*EXAMEN COMPLEXIVO DE GRADO O DE FIN DE CARRERA PRUEBA PRÁCTICA INGENIERÍA EN SISTEMAS"*

Tema:

ANÁLISIS COMPARATIVO DE TECNOLOGÍAS DJANGO JS VS LARAVEL, EN EL MODELO DE PROCESOS PARA EL DESARROLLO DE APLICACIONES WEB

Autor:

GUILINDRO MOREIRA YORGELIS WIMPER

Tutor:

ING. OMAR MONTECE MORENO

LOS RÍOS - BABAHOYO - ECUADOR

RESUMEN

En las últimas décadas el desarrollo web ha estado en crecimiento masivo ya que cada día se realizan más y más solicitudes a la web para brindar a los usuarios una mejor experiencia al hacerlo utilizar este medio, razón por la cual necesitamos mejores herramientas que permitan un despliegue rápido, eficiente y seguro de estas aplicaciones, más la capacidad de adaptarse a las necesidades de los desarrolladores.

El presente caso de estudio tiene como finalidad un análisis comparativo de tecnologías Django vs. Laravel, en el modelo de procesos para el desarrollo de aplicaciones web, se fundamenta en un análisis detallado donde se aplicarán el procedimiento inductivo, analítico, usando además técnicas estadísticas, documental y de campo para la recolección de información relacionado al asunto y a la problemática que existen al momento de desarrollar una solución analizando cuál de estos dos entornos es el mejor, evaluando diversos criterios para poder encontrar sus mejores características.

Para finalizar determinamos que Django tiene facilidad de aprendizaje, debido a que está basado en el lenguaje de programación de Python, sin embargo, demostramos al Framework Django que es más adecuado para el desarrollo de Aplicaciones web basada en criterios seleccionados características de rendimiento, usabilidad, portabilidad y seguridad.

Palabras Claves: Aplicaciones Webs, Frameworks, masivo, solicitudes, despliegue, desarrolladores

ABSTRACT

In recent decades, web development has been in massive growth since more and more requests are made to the web every day to provide users with a better experience by making them use this medium, which is why we need better tools that allow a deployment fast, efficient and secure of these applications, plus the ability to adapt to the needs of developers.

The purpose of this case study is a comparative analysis of Django vs. Laravel, in the process model for the development of web applications, is based on a detailed analysis where the inductive, analytical procedure will be applied, also using statistical, documentary and field techniques for the collection of information related to the issue and the problem that exist at the time of developing a solution by analyzing which of these two environments is the best, evaluating various criteria in order to find its best features.

Finally, we determine that Django is easy to learn, because it is based on the Python programming language, however, we demonstrate to the Django Framework that it is more suitable for the development of web applications based on selected criteria characteristics of performance, usability, portability and security.

Keywords: Web Applications, Frameworks, massive, requests, deployment, developers

INTRODUCCIÓN

Al día de hoy contamos con fuerte material de desarrollo permitiendo la reducción de fracciones horarias de un proyecto, dando facilidad de programación e implementación, dichas librerías se encuentran desarrollados en diferentes lenguajes de programación como C#, Java, JavaScript, Python, entre otros; pero existen herramientas orientadas al desarrollo de pequeños proyectos y otras más escalables, optimizadas y enfocadas a aplicaciones de negocio.

A medida que aumenta la escala de los sistemas de desarrollo, esto también requiere ser realizado con diferentes herramientas que faciliten su desarrollo, existe una variedad de lenguajes de programación y frameworks, a través de estas alternativas, que ayudarán a los desarrolladores web con ciertas dudas y problemas a la hora de elegir, ya que cada idioma tiene sus propios atributos, fortalezas y limitaciones que pueden afectar el plan que se desarrolla.

El estudio de caso está basado en la línea de investigación Sistemas de Información y Comunicación, Emprendimiento e Innovación, y la sublínea utilizada es la de Redes y Tecnologías Inteligentes de Software y Hardware, usamos una metodología de carácter descriptivo, que ayudará a la recolección de la información requerida, cuenta como técnica investigación el análisis documental, para eso usaremos referencias bibliografías conseguidas de sitios web, libros o documentos que aporten con entendimiento en nuestro asunto. La buena selección de los instrumentos de desarrollo influye para que un programa se desarrolle eficazmente y logre satisfacer las expectativas de los usuarios; es por esta razón que se realizará un análisis comparativo entre Frameworks primordiales como Django y Laravel, con el objeto de brindar un mejor entendimiento de conforme con sus propiedades y ventajas que todos pueden otorgar.

DESARROLLO

LARAVEL

Laravel es un framework de código abierto creado en 2011 con el fin de crear aplicaciones y servicios web con PHP 5, 7 y 8, su ideología era la de desarrollar código elegante y simple, evitando el código espagueti; en la categoría de Frameworks es influyente como Ruby on Rails, Sinatra y ASP.NET MVC. (Desarrolladores de Laravel, 2021).

Nos permite el desarrollo de aplicaciones web totalmente personalizadas de elevada calidad, uno de los frameworks más utilizados y de mayor comunidad en el mundo de Internet, resulta bastante moderno y ofrece muchas utilidades potentes a los desarrolladores, que permiten agilizar el desarrollo de las aplicaciones web. Su código es de calidad, la facilidad de mantenimiento y escalabilidad, lo que permite realizar proyectos desde pequeños a grandes o muy grandes. Además, permite y facilita el trabajo en equipo y promueve las mejores prácticas. (Desarrollo Web, 2021).

Su meta es ser un frameworks que posibilite la utilización de una sintaxis expresivas para establecer códigos de forma sencillas y permitiendo variedades de características. Además, se espera utilizar lo mejor de otros frameworks y emplear las propiedades de las últimas variantes de PHP, consta de dependencias, en especial de Symfony, esto incluye el desarrollo de Laravel, además del desarrollo de sus aportaciones. (Desarrolladores de Laravel, 2021).

Ilustración 1. Logo Laravel

Autor: www.laravel.com

Características

- Sistema de enrutamiento: Por medio de las cuales es simple generar y conservar toda clase de URLs amistosas a usuarios y buscadores, rutas de API, etc.

- Demandas Fluent: Construcción de colas de trabajo, debido a lo cual es viable mandar labores para ejecución en background e incrementar el rendimiento de las aplicaciones.

- Eloquent ORM: Un sistema de abstracción de base de datos, con un ORM potente empero sencillo de manejar, por medio de el que tenemos la posibilidad de intentar los datos de la base de datos como si fueran primordiales objetos.

- Abstracción del sistema de archivos: por medio del cual tenemos la posibilidad de redactar datos en proveedores cloud, y obviamente en el disco del servidor, con el mismo código.
- Soporte para el caché: Con todo lo primordial como recordatorios de clave, afirmación de cuentas, rememorar un cliente logueado, etc.

- Soporte para MVC: Labora con una arquitectura de carpetas avanzada, debido a lo cual promueve la división de los archivos con un orden conveniente y determinado, que guiará a todos los miembros del equipo de trabajo y va a ser un estándar durante los diversos proyectos.

- Usa elementos de Symfony: probabilidad de entrar a datos en realtime y recibir notificaciones una vez que éstos se alteran en la base de datos

Arquitectura

Laravel propone en el desarrollo usar 'Routes with Closures', en lugar de un MVC tradicional con el objetivo de hacer el código más claro. Aun así, permite el uso de MVC tradicional.

```
<?php


    //punto de entrada de la petición HTTP
    ...
});
```

Modelo:

Posee un sistema de mapeo de datos relacional llamado Eloquent ORM que facilita la creación de modelos, mismo que está basado en un patrón active récord, su uso es opcional, también dispone de otros recursos que facilitan la interacción con los datos, o específicamente la creación de modelos. A continuación, se mostrará la manera de crear Modelos en Laravel con Eloquent ORM:

```
use Illuminate\Database\Eloquent\Model;


class Libro extends Model {


   //defiendo el nombre de la tabla con la info de los libros
   protected $table = 'tb_libros';
}
```

El código del modelo puede ser más simple en el caso que el nombre de la tabla coincida con el nombre de la clase, ya que Laravel, al igual que Ruby on Rails, usa el paradigma de programación donde se favorece "la convención sobre la configuración". Si ahora necesitamos disponer un listado, en la ruta: https://miproyecto.com/libro/listar Entonces, solo bastaría crear la Ruta e interactuar con el modelo 'Libro' anteriormente creado, del modo siguiente:

```
Route::get('libro/listar', function() {
   $libros = Libro::all();
   return View::make('mi_vista', $libros);  //Muestra los datos
});
```

Vista

Incluye un sistema de procesamiento de plantillas de nombre Blade, dicho sistema de plantillas favorece un código mucho más limpio en las Vistas, además de incluir un sistema de Caché que lo hace mucho más rápido, también, permite una sintaxis mucho más reducida en su escritura. Por ejemplo, en vez pintar la vista usando el código PHP:

```
<?php echo $mi_nombre; ?>
```

En Blade se escribiría:

```
{{$mi_nombre}}
```

Lo cual no es una gran ventaja, siempre es posible usar una expresión resumida en PHP. No obstante, lo que sí es una gran ventaja, es el modo en que Blade maneja las plantillas.

Plantillas:

Las plantillas en Blade son archivos de texto sin formato que contienen todo el HTML de la página con etiquetas que representan los elementos o áreas que se incluirán en la plantilla, o vistas parciales como se conocen en otros marcos PHP, en Blade estos elementos incrustados se organizan en un archivo único, mejorando la organización y el rendimiento de las vistas, especialmente cuando las presentaciones se vuelven demasiado complejas incluso con elementos superpuestos. Al renderizar una vista completa en Laravel, se utilizan dos archivos: la plantilla que define el HTML general y las regiones que se incluirán. Aquí está el ejemplo de código HTML5:

```
<!DOCTYPE html>
<html lang="es">
   <head>
      <meta charset="UTF-8">
      <title>@yield('titulo')</title>
   </head>
   <body>
      @yield('navegacion')
   </body>
</html>
```

El código @yield() identifica al método donde como parámetro se indica el nombre de la zona desplegar, el código de la vista, donde se define la plantilla a usar y la información de las distintas zonas a desplegar:

```
<!-- identificando la plantilla a utilizar -->
@extends('template')
<!-- definiendo una zona -->
@section('titulo')
@endsection
<!-- definiendo otra zona -->
@section('navegacion')
@endsection
```

Controlador

Los controladores contienen la lógica de la aplicación y permiten organizar el código en clases sin tener que escribirlo todo en las rutas. Todos los controladores deben extenderse de la clase BaseController además de eso. Un ejemplo de un controlador en Laravel:

```
class UserController extends BaseController {
    public function mostrarPerfil($id)
    {
        $user = User::find($id);
        return View::make('user.profile', array('user' => $user));
    }
}
```

Estos pueden ser llamados en las rutas de diferentes maneras, pero la más común usando rutas es:

```
Route::get('user/{id}', [\App\Http\Controllers\UserController::class, 'mostrarPerfil']);
```

Ilustración 2. Modelo Vista Controlador

Autor: www.kiwop.com

VENTAJAS DE LARAVEL

- Existe documentación sencilla y completa, en diversos sitios incluyendo el sitio web oficial.

- Posee una comunidad atractiva y de gran tamaño donde brindar todo tipo de ayuda.

- Nos facilita el monitoreo de rutas de nuestro proyecto.

- Reduce los costos y el tiempo de desarrollo y mantenimiento de nuestras aplicaciones.

- Genera enlaces entendibles siendo así más fácil el mantenimiento del sitio web.

- Es OpenSource o de Código Abierto.

DESVENTAJAS DE LARAVEL

- Tienes que estar acostumbrado al manejo de consola para ejecutar ciertas funciones.

- Tendrás conocimiento de Composer con sus dependencias y Artisan de Laravel.

- Su instalación es mediante Composer no es tan fácil como Django.

- Su curva inicial de aprendizaje suele ser algo difícil si con anterioridad no has manejado Frameworks.

- Solo soporta Php a partir de la versión 5.4, diseñado para sus últimas versiones.

DJANGO

Es un frameworks de creación web de código abierto, escrito en Python, que respeta el gerente de diseño vista controlador (MVC), creado únicamente para regir sitios web orientados a noticias de la World Company de Lawrence, Kansas, y ha sido emitido al público bajo una licencia BSD en julio de 2005; el frameworks ha sido designado en referencia al guitarrista de jazz gitano Django Reinhardt. En junio de 2008 ha sido anunciado que la fundación Django Programa Foundation se realizaría cargo de Django en el futuro, su meta primordial es facilitar la construcción de sitios web complicados. Django pone hincapié en el reusó, la conectividad y extensibilidad de elementos, el desarrollo veloz y el inicio DRY perteneciente del inglés Don't Repeat Yourself, que significa No te repitas; este lenguaje Python es utilizado en todos los elementos del framework, inclusive en configuraciones, archivos, y en sus modelos de datos. (Python, 2021).

Ilustración 3. Logo Django

Autor: www.rescalante.com

Características

Al igual que Ruby on Rails, otro popular framework de código abierto, Django se usó en producción a lo largo de una época anterior a que se liberara al público; fue creado por Adrian Holovaty, Simon Willison, Jacob Kaplan-Moss y Wilson Miner a medida que trabajaban en World En línea, y originalmente se usó para regir 3 sitios web de noticias: The Lawrence Jornal-World, lawrence.com y KUsports.com.

Los principios de Django en la gestión de sitios de noticias son evidentes en su diseño, debido a que da una secuencia de propiedades que facilitan el desarrollo veloz de páginas orientadas a contenidos. Ejemplificando, en vez de solicitar que los desarrolladores escriban controladores y vistas para las zonas de gestión de la página, Django da una aplicación integrada para regir los contenidos, que puede incluirse como parte de cualquier página elaborada con Django y que puede regir numerosas páginas desde una misma instalación; la aplicación administrativa posibilita la construcción, actualización y supresión de objetos de contenido, llevando un registro de cada una de las ocupaciones llevadas a cabo sobre cada uno, y da una interfaz para regir los usuarios y los conjuntos de usuarios.

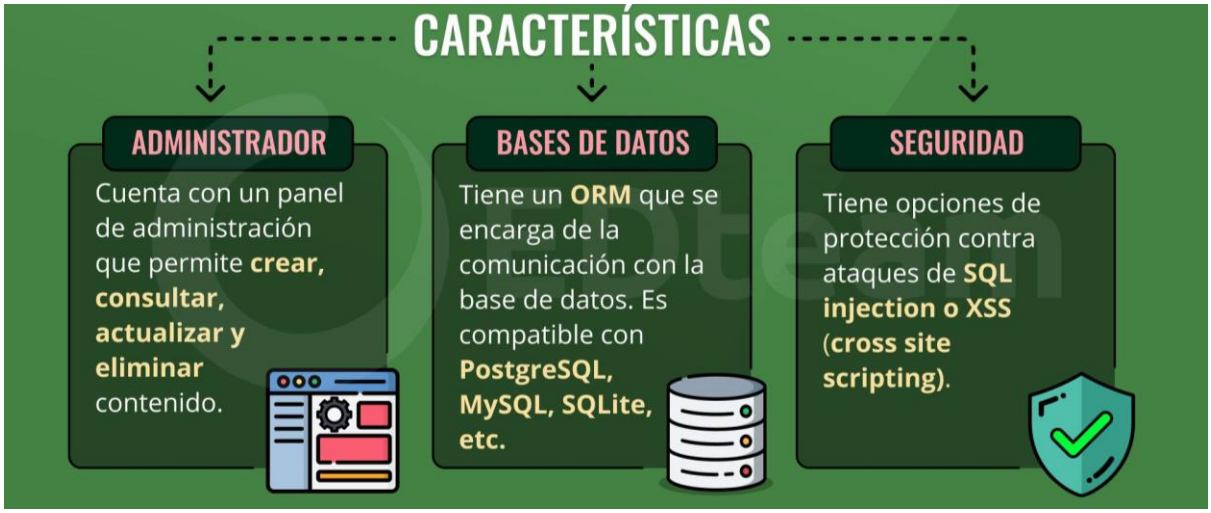

Ilustración 4. Características Django

Autor: www.ed.team

La asignación esencial de Django además reúne aplicaciones que otorgan un sistema de comentarios, herramientas para sindicar contenido vía RSS y/o Atom, "sitios planos" que concede la administración de los sitios de contenido sin necesidad de colocar controladores o vistas para esas páginas, y un sistema de redirección de URLs. Entre otras propiedades de Django encontraremos las siguientes:

- Un mapeador objeto-relacional.

- Aplicaciones "enchufables" que pueden instalarse en cualquier página gestionada con Django.
- Una API de base de datos robusta.
- Un sistema incorporado de "vistas genéricas" que ahorra tener que redactar la lógica de ciertas labores usuales.
- Un sistema extensible de plantillas con base en etiquetas, con herencia de plantillas.
- Un despachador de URLs basado en expresiones regulares.
- Un sistema "middleware" para desarrollar propiedades extras; ejemplificando, el reparto primordial de Django incluye elementos middleware que dan cacheo, compresión de la salida, normalización de URLs, custodia CSRF y soporte de sesiones.
- Soporte de internacionalización, incluyendo traducciones incorporadas de la interfaz de administración.
- Documentación integrada accesible a través de la aplicación administrativa (incluyendo documentación generada automáticamente de los modelos y las bibliotecas de plantillas añadidas por las aplicaciones).

Arquitectura

Aunque Django está poderosamente inspirado en la filosofía de creación Modelo Vista Controlador, sus desarrolladores mencionan públicamente que no se sienten en especial atados a mirar estrictamente ningún paradigma especial, y sin embargo prefieren hacer "lo cual les parece adecuado". Como consecuencia, ejemplificando, lo cual se llamaría "controlador" en un "verdadero" framework MVC se denomina en Django "vista", y lo cual se llamaría "vista" se denomina "plantilla". (Django, 2022).

Presentación: Aquí se maneja la relación entre el cliente y el computador. En Django, esta labor la hacen el motor de plantillas y el cargador de plantillas que toman la información y la

muestran al cliente (vía HTML, por ejemplo). El sistema de configuración de URLs es además parte de la capa de presentación.

Control: En esta capa se basa el programa o la lógica de aplicación en sí. En Django son representados por las vistas y los manipuladores. La capa de presentación es dependiente de esta y paralelamente esta es dependiente de la capa de dominio.

Mediator: Es el delegado de manejar la relación entre el subsistema Entity y foundation. Aquí se hace el mapeo objeto-relacional a cargo del motor de Django.

Entity: El subsistema entity maneja los objetos de comercio. El mapeo objeto-relacional de Django posibilita redactar objetos de tipo entity de una manera simple y estándar.

Foundation: La primordial labor del subsistema foundation es la de manejar a bajo grado el trabajo con la base de datos. Se provee soporte a grado de foundation para algunas bases de datos y otras permanecen en fase de prueba.

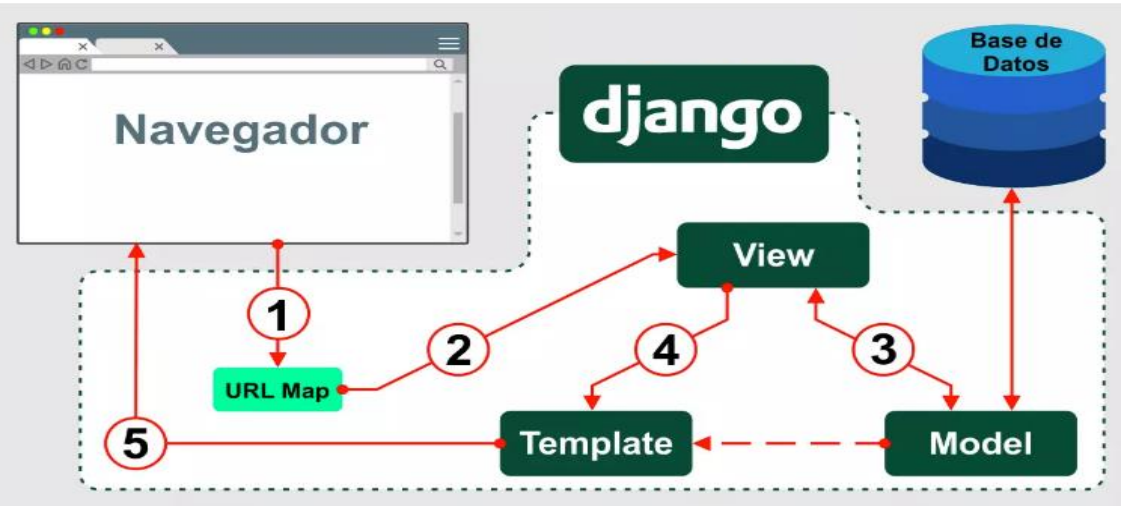

Ilustración 5. Arquitectura de Django

Autor: www.espifreelancer.com

VENTAJAS DE DJANGO

- posibilita separar el sistema.
- posibilita repartir correctamente las funciones el sistema.
- Nos facilita la intercomunicación de diversas tecnologías
- Es escalar.

DESVENTAJAS DE DJANGO

- Añade complejidad al stack
- Necesita mantenimiento
- Es dificil para debugear

TABLA COMPARATIVA DE CARACTERÍSTICAS DE FRAMEWORKS

Características	Laravel	Django
Tiempo de Ejecución de Operaciones CRUD	6.12 segundos	8.31 segundos
Uso del CPU	3.8%	2.7%
Confidencialidad y Autenticidad	8.89/10	8.89/10
Tiempo de Aprendizaje de Frameworks	41 horas	40 horas
Uso de Memoria RAM	3.8 megabytes/segundo	2.9 megabytes/segundo
Lenguaje Escrito	Php	Python
Lineas de Código de Aplicación de Prueba	2766	650

Tabla 1: Comparativa de características de Laravel vs. Django.

Fuente: Yorgelis Guilindro

En base a los resultados logrados desde la evaluación de los frameworks, se procedió a hacer la comprobación de la premisa que ha sido anteriormente fundada, obteniendo lo próximo: En la presente indagación se comprueba que "el estudio comparativo para la evaluación de los frameworks Laravel y Django permitió decidir el framework más correcto para el desarrollo de aplicaciones web". Esta confirmación es verdadera pues la investigación comparativo permitió entablar a Django como el framework óptimo para el desarrollo de aplicaciones web, esta información se puede verificar en la Tabla 1, donde se muestra que Django obtuvo un costo de 8.16, lo que representa el 81.6% frente al 7.29 que simboliza el 72.9% obtenido por Laravel en la evaluación de las propiedades establecidas para la comparación de ambos ámbitos de desarrollo web.

En el desarrollo del experimento se ha podido establecer que aunque Django es un ámbito bastante simple de aprender, ya que está con base en el lenguaje de programación Python, sin embargo, el desarrollo de la aplicación de prueba para la evaluación de los Frameworks ha sido terminada previamente con la utilización del Framework Laravel; esto se debió a que los alumnos que trabajaron con Laravel tenían más entendimiento y vivencia en el desarrollo de esta clase de aplicaciones y estaban más familiarizados con la implementación de la herramienta, de tal forma que esa aplicación ha sido terminada en 41 horas, en lo que la aplicación elaborada en Django ha sido terminada en una época de 40 horas.

BIBLIOGRAFÍA

Desarrolladores de Laravel. (8 de Marzo de 2021). *Introduccion - Documentation Laravel PHP Framework*. Obtenido de https://www.laravel.com

Desarrollo Web. (1 de Febrero de 2021). *Desarrollo Web*. Obtenido de https://desarrolloweb.com/home/laravel

Django. (03 de Agosto de 2022). *Django Project*. Obtenido de https://www.djangoproject.com/weblog/2022/aug/03/django-41-released/

Docs, L. (11 de Agosto de 2022). *Architecture of Laravel Applications - Laravel Book*. Obtenido de https://laravel.com/docs/9.x/cache

Molina Ríos, J. R., Loja Mora, N. M., Zea Ordóñez, M. P., & Loaiza Sojos, E. L. (23 de 09 de 2019). *Evaluación de los Frameworks en el Desarrollo de Aplicaciones Web con Python*. Obtenido de http://revistas.unla.edu.ar/software/article/view/1149

Python, M. D. (19 de Febrero de 2021). *Métodos File en Python: Creación y manipulación de archivos de texto*. Obtenido de https://pythondiario.com/2019/03/metodos-file-en-python-creacion-y.html

Vergara, R. C. (17 de Junio de 2020). *Biblioteca Digital Universidad de Alcalá*. Obtenido de Estudio del framework de desarrollo web Django: https://ebuah.uah.es/dspace/handle/10017/32018