ESCUELA SUPERIOR POLICTÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

ESCUELA DE INGENIERÍA EN SISTEMAS

"ANÁLISIS DEL RENDIMIENTO DE FRAMEWOKS PHP PARA DESARROLLAR APLICACIONES WEB ÓPTIMAS. CASO PRÁCTICO: PORTAL ACADEMIA LINUX ESPOCH"

TESIS DE GRADO

Previa obtención del título de:

INGENIERO EN SISTEMAS INFORMÁTICOS

Presentado por:

EDWIN RICARDO SÁNCHEZ OSEJO
ITEL EDUARDO VERA CÁRDENAS

RIOBAMBA – ECUADOR
2011

AGRADECIMIENTO

Este trabajo agradezco primero a mi DIOS por darme la vida, a mi esposa Nuvia Esthela por estar a mi lado apoyándome, a mi hijo Edwin Sebastián quien a sus pocos días me supervisaba en mi trabajo, a mis padres Edwin y Elisa que me han dado toda la educación y su sabiduría, a mis hermanas Erika y Yessenia con las que compartí mi niñez y juventud, a todos y cada uno de mis amigos que me motivaron y ayudaron a seguir adelante y luchar por alcanzar estos anhelados sueños de ser un profesional.

Edwin Ricardo Sánchez Osejo

La vida está llena de retos, metas por cumplir, este trabajo en un escalón más para alcanzar solamente, una parte de mis tres ejes centrales, familia, amigos, trabajo, agradezco a mi padre que me protege desde lo desconocido para todos, y a mi madre que es la persona que cuando se piensa que todo está perdido te da una luz e incluso guía tu camino.

Itel Eduardo Vera Cárdenas

DEDICATORIA

A mi DIOS, a mi esposa Nuvia Esthela, a mi hijo Edwin Sebastián, a mis padres Edwin y Elisa, a mis hermanas Erika y Yessenia, a mi amigo Itel, a cada uno por su tiempo dado y comprensión de mis logros y caídas que estuvieron para alentarme.

Edwin Ricardo Sánchez Osejo

A mi padre que hubiera estado orgulloso y lleno de gozo, que me apoya desde el umbral incierto que todos debemos pasar para llegar al nuevo despertad de la vida.A mi madre que la tengo llena de vida y lucha siempre por sus hijos el mejor ejemplo de abnegación y entrega que sin ella no existo.

Itel Eduardo Vera Cárdenas

FIRMAS RESPONSABILIDAD Y NOTA

NOMBRE	FIRMA	FECHA
Ing. Iván Menes Camejo **DECANO DE LA FACULTAD DE INFORMÁTICA Y ELECTRONICA**	……..	……
Ing. Raúl Rosero **DIRECTOR DE LA ESCUELA DE INGENIERIA EN SISTEMAS**	……..	……
Ing. Danilo Pastor **DIRECTOR DE TESIS**	……..	……
Ing. Gloria Arcos **MIEMBRO DEL TRIBUNAL**	……..	……
Tlgo. Carlos Rodríguez **DIRECTOR DEL CENTRO DE DOCUMENTACIÓN**	……..	……

RESPONSABILIDAD DEL AUTOR

"Nosotros Edwin Ricardo Sánchez Osejo e Itel Eduardo Vera Cárdenas, somos los responsables de las ideas, doctrinas y resultados expuestos en esta Tesis de Grado, y el patrimonio intelectual de la misma pertenece a la Escuela Superior Politécnica de Chimborazo."

FIRMAS:

_____	_____
Edwin Ricardo Sánchez Osejo	Itel Eduardo Vera Cárdenas

INDICE DE ABREVIATURAS

A

ABE Ancho de Banda de Entrada

ABS Ancho de Banda de Salida

ACLAccessControlList

ACOAccesControlObject-Objetosdecontroldeacceso

AMAgileModeling–ModeladoAgile

AJAXAsynchronusJavascriptandXML–JavascriptyXMLAsíncrono

API AplicationProgrammingInterface.InterfazdeProgramacióndeAplicaciones

AROAccesRequestObject Objetosquedeseanaccederarecursos(usuariooogrupos)

B

BDBasedeDatos

BSDBerkeleySoftwareDistribution–DistribucióndeSoftwareBerkeley

C

CLR.CommonLanguageRuntime–LenguajeComúndeEjecución

CRCClaseResponsabilidadyColaboración

CSSCascadingStyleSheets–HojadeEstiloenCascada

CUxP Consumo de Usuarios por Prueba

CUxT Consumo de Usuarios por Transacción

D

DLLDynamicallyLinkedLibrary-LibreríadeEnlaceDinámico

F

FAQFrencuentlyAskedQuestion-PreguntasyRespuestasFrecuentes

FPDF.F-Portabledocumentformat.F-LibreríadePHPquetransformaenunformato dedocumentoportátil(PDF)

G

GPLGeneralPublicLicense-LicenciaPúblicaGeneral

GUIGraphicalUserInterface-Interfazgráficadeusuario **H**

HTMLHypertextMarkupLanguage-Lenguajedemarcadodehipertexto

HTTPHypertextTransferProtocol–ProtocolodetransferenciadeHipertexto

HWHardware

I

IBMInternationalBusinessMachines–MaquinasdeNegociosInternacional

IDEIntegratedDevelopmentEnvironment- EntornodeDesarrolloIntegrado

IISInternetInformationServer–ServidordeinformacióndeInternet

IDE: (Integrated Development environment – Entorno de desarrollo integrado)

IMAP: (Internet Message Access protocol – Protocolo de acceso a mensajes de internet)

ISAPIInternetServer(**API**)ApplicationProgrammingInterface-Interfazde programacióndeaplicacionesparaInternetInformationServer

M

MABR Mínimo Ancho de Banda Requerido

MD5 Message-DigestAlgorithm5

MPLMozillaPublicLicense

MVCModeloVistaControlador

N

NATNetworkAddressTranslation

O

ODBCOpenDataBaseConnectivity

OOOrientadosaobjetos

ORMObject-RecordMapper

P

P Promedio máximo

PHPPersonalHomePage

PHTMLPHPHypertextMarkupLanguage

POOProgramaciónOrientadaaObjetos

POP3: (Post Office Protocol – protocolo de oficina de correos)

R

RAD: (Rapid Application Development – Desarrollo Rápido de Aplicaciones)

S

SOAPSimpleObjectAccessProtocol–ProtocolodeAccesoaobjetossimples

SQLStructureQueryLanguage-LenguajedeConsultaEstructurado

U

URI: (Uniform Resource Identifier – Identificador uniforme del recurso)

URLUniformResourceLocator-LocalizadorUniformedeRecursos

UMLUnifiedModelingLanguage-Lenguajedemodeladounificado

V

VR Velocidad de la Red

VRTU Velocidad de Red Para el Total de Usuarios

X

XMLExtendedMarkupLanguage-Lenguajedemarcadoextendido

XPExtremeProgramming–Programaciónextrema

Z

ZF: (Zend Framework)

INDICE GENERAL

INDICE DE FIGURAS

INDICE DETABLAS

INTRODUCCIÓN

El presente trabajo de investigación de tesis previo a la obtención del título de Ingeniería en Sistemas Informáticos, trata "ANÁLISIS DEL RENDIMIENTO DE FRAMEWORKS PHP PARA DESARROLLAR APLICACIONES WEB ÓPTIMAS. CASO PRÁCTICO: PORTAL ACADEMIA LINUX ESPOCH".

Es un estudio y análisis investigativo a fondo del rendimiento de diferentes Frameworks para obtener como resultado el Frameworks másóptimo para el desarrollo de aplicaciones web.

Como precedente cuando se trata de desarrollar aplicaciones web en lenguaje Php, la mayoría de desarrolladores implementan sus proyectos en el Frameworks que tienen mejor conocimiento, sin realizar una investigación previa en búsqueda del Frameworks que se ajuste a sus necesidades a su vez ofrezca el mejor rendimiento y estabilidad.

En la actualidad el desarrollo de aplicaciones web requieren la mejor optimización de recursos de hardware y software que se encuentran conectados a la red de Internet, Aquí nace la preocupación de la ingeniería de software, una vez recopilados los requisitos del proyecto web, estandarizar la búsqueda del Frameworks que se acople a las necesidades del sistema.

El presente trabajo contiene los siguientes capítulos:
El en Capítulo 1 se presenta el planteamiento de la investigación, antecedentes, hipótesis, métodos y técnicas, es todo el marco referencial para el desarrollo de la tesis.

En el Capítulo 2 se detalla los aspectos teóricos motivo de la investigación, conceptos, terminologías, referenciadas al objeto de estudio.

El Capítulo 3 trata del desarrollo del análisis comparativo de los Frameworks, donde se han seleccionado 3 herramientas de desarrollo web para determinar cual tiene el mejor desempeño en el rendimiento, sometiendo a cada sistema web a diferentes escenarios de pruebas.
Además se han determinado los indicadores e índices que sirven para comparación de los Frameworks, finalizando con la demostración de la hipótesis.

En el Capítulo 4 se detalla la parte aplicativa de la tesis, contiene los requisitos de ingeniería de software, desarrollo rápido de aplicaciones web, estándares de desarrollo, todo referente al portal de la academia Linux-Espoch. El manejo de usuarios será mediante un control de acceso de Id y clave, se manejara artículos, archivos históricos de todas las publicaciones del portal, el administrador será el encargado de realizar publicaciones.

1. CAPÍTULO I:

MARCO REFERENCIAL

1.1. Antecedentes

En la actualidad el desarrollo de aplicaciones web se ha vuelto muy popular a nivel de todas las empresas que deseen ofertas sus productos y servicios.

En nuestro medio al no existir estudios previos de rendimiento de Frameworks que ayuden al desarrollador, ellos utilizan empíricamente un entorno de desarrollo integrado, en estos casos no existen criterios de desarrollo para el aseguramiento de la calidad del software.

El problema de desarrollar sistemas sin tomar en cuenta técnicas de la ingeniería de software para el proceso del desarrollo de software se vería afectado el diseño de la calidad con sistemas poco satisfactorios e improductivos en cuanto a su rendimiento y escalabilidad.

El Diseño de la arquitectura de software es la fase importante que no se debe saltar para la selección de las mejores herramientas de desarrollo que permitan crear un buen sistema garantizando que se mantendrá el rendimiento a pesar de las modificaciones que se realicen.

Con el refinamiento de la arquitectura de software podremos elegir el mejor Framework que se adapte a los requisitos de desarrollo del sistema.

Entre los Frameworks más populares de la actualidad tenemos los siguientes: Zend Framework, CodeIgniter, Kohana, CakePhp, Symfony, PHP Prado, Yii Framework entre otros.

El aseguramiento de la calidad se ha convertido en una necesidad de prioridad para las empresas que desarrollan software porque en la medida que avanza la tecnología, es más exigente la calidad del software requerida actualmente por las empresas encargadas de medir la calidad, y para satisfacer estos requerimientos de calidad uno de los métodos más utilizados por los desarrolladores es la selección del top Frameworks.

El portal web de la academia Linux tiene las opciones básicas que trabaja en tiempos y estándares aceptables, pero el desarrollo de este portal no está pensado para la reutilización del código y no permite la escalabilidad de mismo ya que no posee un Frameworks adecuado

1.2. Justificación del Proyecto de Tesis

1.2.1. Justificación Teórica

La clasificación de los Frameworks mediante estudios y comparaciones técnicas es parte vital para la selección de la herramienta que permita mejorar la calidad del software, ya que si no elegimos el Frameworks con el mejor rendimiento nuestros procesos tendrían tiempos de ejecución y respuesta lentos.

Al comparar el rendimiento de los Frameworks el desarrollador podrá seleccionar el entorno de desarrollo más óptimo acorde a sus requerimientos.

El desarrollo de este trabajo está encaminado a realizar pruebas de Laboratorio con el fin de mejorar y promover el óptimo rendimiento de nuestras aplicaciones en la web,

evaluando y poniendo a prueba todas las capacidades de los distintos Frameworks PHP más populares en la actualidad.

La selección de Frameworks óptimo ayudará a mejorar los tiempos de procesamiento de peticiones del portal web a desarrollarse.

Para el desarrollo del portal se seleccionaría un IDE que trabaje con el Frameworks PHP ganador para poder tener el óptimo rendimiento.

El alcance que se desea enfocar al Portal web es que los tiempos de respuestas de las peticiones que realicen los usuarios sean los mínimos optimizando el código Php para un excelente desempeño del sitio y sea visitado más a menudo por su eficiencia.

1.2.2. Justificación Práctica

Los escenarios de pruebas a desarrollarse para el análisis comparativo del rendimiento serán como mínimo tres y serán puestos a prueba en los 3 Frameworks seleccionados con los siguientes escenarios básicos:

- Módulo de interfaz
- Módulo de consultas SQL
- Módulo de funciones y procedimientos

El alcance de la aplicación contemplará el rediseño de la apariencia del portal web con el uso de menús de navegación.

La gestión de descargas, mediante este módulo se podrá gestionar todos los archivos que posteriormente los usuarios van a poder descargar del portal Web.

Gestión de usuarios contempla el rediseño del módulo para una navegación intuitiva y amigable a la vez.

Módulo de noticias, artículos se podrá crear cualquier contenido o información de forma sencilla.

Hemeroteca archiva en el histórico de contenidos las noticias que han pasado a un segundo plano de la actualidad para que siempre estén disponibles para sus visitantes.

Módulo de gestión de Inscripción de usuarios ayudará para dar privilegios del tipo de usuario para su posterior los usuarios pueda comentar sus opiniones o expresar sus inquietudes en la página de comentarios.

1.3. Objetivos

1.3.1. Objetivo General:

Realizar un análisis comparativo del rendimiento de Frameworks Php para desarrollar aplicaciones web óptimas.

1.3.2. Objetivos Específicos.

- Estudiar los conceptos de funcionamiento, arquitectura, y rendimiento del Frameworks web.

- Realizar escenarios de pruebas que permitan determinar cuantitativamente el rendimiento de los Frameworks

- Comparar los Frameworks Php y seleccionar el de mayores prestaciones de rendimiento.

- Desarrollar el portal web usando el Frameworks seleccionado para la Academia LINUX - ESPOCH.

1.4. Hipótesis

CodeIgniter es el Frameworks que ofrece el mejor rendimiento frente a Zend y CakePhp

1.5. Métodos y Técnicas

1.5.1. Métodos

El método utilizado como guía para la presente investigación es el método Científico, el cual contempla los siguientes puntos:

- El planteamiento del problema que es objeto principal de nuestro estudio.
- El apoyo del proceso previo a la formulación de la Hipótesis.
- Levantamiento de información necesaria.
- Análisis e interpretación de Resultados.
- Proceso de Comprobación de la Hipótesis, etc.

Para complementar la investigación se aplicará el método deductivo ya que parte de verdades previamente establecidas como principios generales, para luego aplicarlo a casos individuales y comprobar así su validez en el desarrollo del portal web de la Academia LINUX-ESPOCH.

1.5.2. Técnicas

Para la recopilación de la información necesaria que sustente este trabajo de investigación, se ha establecido como técnicas las siguientes:

- Revisión de Documentos
- Observación
- Técnica de Comprobación de Hipótesis
- Encuestas

2. CAPÍTULO II:

MARCO TEORICO

2.1. Definiciones

2.1.1. Frameworks

Un Framework, en el desarrollo de software es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto. [25]

2.1.2. PHP

El acrónimo de "*PHP: Hypertext Preprocessor*", es un lenguaje "Open Source" interpretado de alto nivel, especialmente pensado para desarrollos web y el cual puede ser incrustado en páginas HTML. La mayoría de su sintaxis es similar a C, Java y Perl y es fácil de aprender. La meta de este lenguaje es permitir escribir a los creadores de páginas web, páginas dinámicas de una manera rápida y fácil, aunque se pueda hacer mucho más con PHP. [17]

2.1.3. Base de Datos

Se define una base de datos como una serie de datos organizados y relacionados entre sí, los cuales son recolectados y explotados por los sistemas de información de una empresa o negocio en particular. La Base de datos es un conjunto de información almacenada en memoria auxiliar que permite acceso directo y un conjunto de programas que manipulan esos datos.

Una base de datos es un “almacén” que nos permite guardar grandes cantidades de información de forma organizada para que luego podamos encontrar y utilizar fácilmente. A continuación te presentamos una guía que te explicará el concepto y características de las bases de datos.

El término de bases de datos fue escuchado por primera vez en 1963, en un simposio celebrado en California, USA. Una base de datos se puede definir como un conjunto de información relacionada que se encuentra agrupada ó estructurada.

Desde el punto de vista informático, la base de datos es un sistema formado por un conjunto de datos almacenados en discos que permiten el acceso directo a ellos y un conjunto de programas que manipulen ese conjunto de datos.

Cada base de datos se compone de una o más tablas que guarda un conjunto de datos. Cada tabla tiene una o más columnas y filas. Las columnas guardan una parte de la información sobre cada elemento que queramos guardar en la tabla, cada fila de la tabla conforma un registro. [13]

2.1.4. Mapeo Objeto – Relacional (ORM)

Es un componente de software que me permite trabajar con los datos persistidos como si ellos fueran parte de una base de datos orientada a objetos (en este caso virtual). Debido a que lo estándar es trabajar con BD relacionales, se deben realizar operaciones que permitan transformar un registro en objeto y viceversa. [5]

2.1.5. Base de Datos Orientadas a Objetos

En una base de datos orientada a objetos, la información se representa mediante objetos como los presentes en la programación orientada a objetos. Cuando se integra las características de una base de datos con las de un lenguaje de programación orientado a objetos, el resultado es un sistema gestor de base de datos orientada a objetos (ODBMS, object database management system). Un ODBMS hace que los objetos de la base de datos aparezcan como objetos de un lenguaje de programación en uno o más lenguajes de programación a los que dé soporte. Un ODBMS extiende los lenguajes con datos persistentes de forma transparente, control de concurrencia, recuperación de datos, consultas asociativas y otras capacidades. [18]

2.1.6. Memoria Cache

Un caché es un sistema especial de almacenamiento de alta velocidad. Puede ser tanto un área reservada de la memoria principal como un dispositivo de almacenamiento de alta velocidad independiente. Hay dos tipos de caché frecuentemente usados en las computadoras personales: memoria caché y caché de disco. Una memoria caché, llamada también a veces almacenamiento caché o RAM caché, es una parte de memoria RAM estática de alta velocidad (SRAM) más que la lenta y barata RAM dinámica (DRAM) usada como memoria principal. La memoria caché es efectiva dado que los programas acceden una y otra vez a los mismos datos o instrucciones. Guardando esta información en SRAM, la computadora evita acceder a la lenta DRAM.

Cuando se encuentra un dato en la caché, se dice que se ha producido un acierto, siendo un caché juzgado por su tasa de aciertos (*hit rate*). Los sistemas de memoria caché usan una tecnología conocida por caché inteligente en la cual el sistema puede reconocer cierto tipo de datos usados frecuentemente. Las estrategias para determinar qué información debe de ser puesta en el caché constituyen uno de los problemas más interesantes en la ciencia de las computadoras. Algunas memorias caché

estánconstruidas en la arquitectura de los microprocesadores. Por ejemplo, el procesador Pentium II tiene una caché L2 de 512 Kilobytes.

La caché de disco trabaja sobre los mismos principios que la memoria caché, pero en lugar de usar SRAM de alta velocidad, usa la convencional memoria principal. Los datos más recientes del disco duro a los que se ha accedido (así como los sectores adyacentes) se almacenan en un buffer de memoria. Cuando el programa necesita acceder a datos del disco, lo primero que comprueba es la caché del disco para ver si los datos ya están ahí. La caché de disco puede mejorar drásticamente el rendimiento de las aplicaciones, dado que acceder a un byte de datos en RAM puede ser miles de veces más rápido que acceder a un byte del disco duro. [19]

2.1.7. Scaffold

La palabra Scaffold está en inglés y en español significa Andamio, pero en programación el scaffolding es un método para construir aplicaciones basadas en bases de datos, esta técnica está soportada por algunos Frameworks del tipo MVC en el cuál el programador escribe una especificación que describe cómo debe ser usada la base de datos. Luego el compilador utiliza esa especificación para generar el código que la aplicación usará para crear, leer, actualizar y eliminar registros de la base de datos, esto es conocido como CRUD (create, read, update, delete).El Scaffolding fue popularizado por el Frameworks Ruby on Rails y ahora es utilizado por otros Frameworks también cómo Zend, CakePHP, Symfony [20]

2.1.8. Modelo Vista Controlador

Es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El estilo de llamada y retorno MVC (según CMU), se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página. El modelo es el Sistema de Gestión de Base de Datos y la Lógica de negocio, y el controlador es el responsable de recibir los eventos de entrada desde la vista. [21]

2.1.9. Plantillas

Una plantilla es una forma de dispositivo que proporciona una separación entre la forma o estructura y el contenido. Es un medio o un aparato que permite guiar, portar o construir un diseño o esquema predefinido.

Una plantilla agiliza el trabajo de reproducción de muchas copias idénticas o casi idénticas (que no tiene que ser tan elaborado, sofisticado o personal). Si se quiere un trabajo más refinado, más creativo, la plantilla no es sino un punto de partida, un ejemplo, una idea aproximada de lo que se quiere hacer.

Las plantillas, como norma general, pueden ser utilizadas por personas o por sistemas automatizados. Se utilizan plantillas en todos los terrenos de la industria y la tecnología. Una plantilla puede servir como muestra base de una diversidad sobre la que comparten elementos comunes (patrón) y que en sí es lo que constituye la plantilla.

En relación con la edición o composición de textos o imágenes, se compone de cajas y líneas, con unos tamaños y márgenes, para facilitar la escritura de artículos o cartas, con títulos, fotos y diagramas. En relación con la mecánica, puede ser una forma específica de ángulos y medidas, tal que colocando las partes constituyentes en su sitio permite un ensamblaje calibrado y uniforme e identificar la carencia de algún elemento. [22]

2.2. Tipos de Frameworks

Existen varios tipos de Frameworks Web: orientados a la interfaz de usuario, como Java Server Faces, orientados a aplicaciones de publicación de documentos, como Coocon, orientados a la parte de control de eventos, como Struts y algunos que incluyen varios elementos como Tapestry.

La mayoría de Frameworks Web se encargan de ofrecer una capa de controladores de acuerdo con el patrón MVC o con el modelo 2 de Servlets y JSP, ofreciendo

mecanismos para facilitar la integración con otras herramientas para la implementación de las capas de negocio y presentación. [12]

2.2.1. Frameworks de Aplicación

Un Frameworks de aplicación encapsula una capa de funcionalidad horizontal que puede ser aplicada en la construcción de una gran variedad de programas.

2.2.2. Frameworks de Dominio

Un Frameworks de dominio implementa una capa de funcionalidad vertical, correspondiéndose con un dominio de aplicación o una línea de producto. Su evolución deberá ser también la más rápida, pues deben adaptarse a las áreas de negocio para las que están diseñados. [10]

2.2.3. Framework de Integración Middleware

Su utilidad es integrar aplicaciones y componentes distribuidos. Potencian la facilidad del software para ser dividido en módulos, reutilizado y fácilmente extendido. Ejemplos de este tipo de Frameworks incluyen middleware orientado a mensajes y bases de datos transaccionales.

2.2.4. Framework de Caja Blanca o Dirigidos por la Arquitectura

Poseen las características de la orientación a objetos, tales como la herencia o la ligadura dinámica.
El Frameworks se extiende por herencia de sus clases base o redefiniendo sus métodos.
Estos Frameworks definen interfaces para componentes que pueden integrarse mediante composición de objetos. Sin embargo, la dificultad del uso de este tipo de Frameworks reside en el hecho de que requieren un profundo conocimiento y comprensión de las clases a extender. Otro punto débil, propio de la herencia en general, es la dependencia

entre métodos, redefinir una operación puede requerir redefinir otra, y así sucesivamente.

2.2.5. Framework de Caja Negra o Dirigidos por Datos

Se estructuran utilizando composición de objetos y delegación en lugar de herencia. Enfatizan las relaciones dinámicas entre los objetos en lugar de las relaciones estáticas entre las clases. Se puede añadir nueva funcionalidad al Frameworks componiendo objetos existentes de diferentes formas, para reflejar el comportamiento de la aplicación. El usuario del Frameworks no tiene que conocer con profundidad los detalles de éste, sino sólo conocer cómo usar y combinar los objetos existentes. En general son más fáciles de utilizar que los Frameworks de caja blanca. Por el contrario, los Frameworks de caja negra son más difíciles de construir, porque sus interfaces y puntos de anclaje deben anticiparse a un conjunto muy amplio de posibilidades.

Debido a su flexibilidad predefinida, éstos son más rígidos dentro del dominio al que dan soporte. Un excesivo uso de la composición de objetos puede provocar diseños difíciles de comprender.

2.2.6. Framework de Caja Gris

Es una mezcla de los dos tipos anteriores, buscando evitar sus desventajas. Permiten la extensión mediante la herencia y la ligadura dinámica, y además tienen la propiedad de ocultar información innecesaria a los usuarios del Frameworks.

2.3. Arquitectura del Frameworks

Dentro de este aspecto, podemos basarnos en el modelo MVC (Controlador => Modelo => Vista) ya que debemos fragmentar nuestra programación. Tenemos que contemplar estos aspectos básicos en cuanto a la implementación de nuestro sistema.

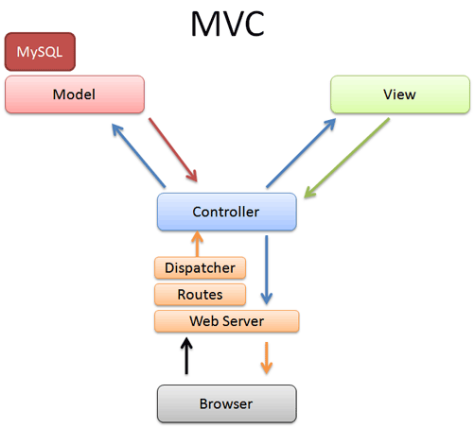


Ilustración 1 Arquitectura MVC

Fuente: http://xyox.info/2008/10/12/php-y-la-arquitectura-mvc/

- **Modelo:**

Este miembro del controlador maneja las operaciones lógicas, y de manejo de información (previamente enviada por su ancestro) para resultar de una forma explicable, y sin titubeos. Cada miembro debe ser meticulosamente llamado, en su correcto nombre y en principio, con su verdadera naturaleza: el manejo de información, su complementación directa.

- **Vista:**

Al final, a este miembro de la familia le corresponde dibujar, o expresar la última forma de los datos: la interfaz gráfica que interactúa con el usuario final del programa (GUI). Después de todo, a este miembro le toca evidenciar la información obtenida hasta hacerla llegar con el controlador. Solo (e inicialmente), nos espera demostrar la información.

- **Controlador:**

Con este apartado podemos controlar el acceso (incluso todo) a nuestra aplicación, esto pueden ser: archivos, scripts o programas; cualquier tipo de información que permita la interfaz. Así, podremos diversificar nuestro contenido de forma dinámica, y estática (a la vez); pues, sólo debemos controlar ciertos aspectos (como se ha mencionado antes). [21]

2.4. Rendimiento del Frameworks

En informática, medida o cuantificación de la velocidad/resultado con que se realiza una tarea o proceso. En una computadora, su rendimiento no depende sólo del microprocesador como suele pensarse, sino de la suma de sus componentes como la memoria, el bus, los diversos dispositivos, etc. y su software. [2]

2.4.1. Medida de Optimización

En general, la optimización es empleada para que una tarea se realice más rápidamente. Pero este no siempre es el caso; por ejemplo, en determinados casos lo más importante es que se consuma menos memoria, por lo tanto, se deben crear programas más lentos, pero que estén optimizados con respecto la memoria.

La optimización se hace siempre con respecto a uno o más recursos como ser: tiempo de ejecución, uso de memoria, espacio en disco, ancho de banda, consumo de energía, etc. Muchas veces la optimización de un recurso se hace a expensas de otros recursos.

- **Optimización de aplicaciones**

Optimizar una aplicación significa hacer los cambios pertinentes para que esta se ejecute y funcione más rápidamente, o para que ocupe menos memoria, o para que gaste menos batería (útil en computadoras portátiles). Por lo general la optimización de un programa se hace a través de otros programas, una mejor configuración o similares, pero siempre a nivel software.

De todas maneras, la mejor optimización que puede hacérsele a una aplicación es a nivel código fuente, es decir, cambios en las sentencias de programación. Mejorar los algoritmos resulta en una mejora del rendimiento general de la aplicación. También puede emplearse un compilador optimizador que ayude a crear una aplicación más optimizada.

Muchas veces, la optimización de los algoritmos resulta en códigos menos legibles para los programadores y más difíciles de mantener y expandir.En programación, la eficiencia es utilizada para describir varias propiedades deseables en los algoritmos. Generalmente la eficiencia puede medirse en:

* Velocidad: el tiempo que toma para completar una operación.
* Espacio: la memoria principal y/o secundaria empleada.

La optimización es el proceso de hacer un código (grupo de algoritmos) lo más eficientemente posible. A veces optimizar espacio implica una desmejora en la velocidad, o viceversa.

Muchas veces, la optimización de los algoritmos depende de las propiedades de la máquina en donde serán ejecutados.

- **Optimización de dispositivos**

La optimización de dispositivos o periféricos de una computadora puede hacerse a nivel hardware (cambio de piezas, cables, puertos, jumpers, etc.) o a nivel software (cambio en las configuraciones, controladores, aplicaciones, etc.). Pero también un dispositivo puede optimizarse a nivel uso, es decir, su rendimiento puede muchas veces estar relacionado al uso que el propio usuario le da.

- **Optimización de redes**

En tanto, para optimizar una red, puede hacerse a nivel software (configuración, programas, etc.) o a nivel hardware (antenas, cables, etc.).

Generalmente, el rendimiento de una red de computadoras es medido o cuantificado usando la velocidad de transmisión de datos. Es una medida concreta y de fácil cálculo, que permite saber si una red está funcionando en forma óptima.

Otras formas de medir el rendimiento en una red, es la cantidad de paquetes de datos que llegan de forma íntegra desde un nodo hacia otro en la red. En el camino, los paquetes de datos pueden alterarse (generalmente por interferencias en la conexión física). Un elevado porcentaje de paquetes íntegros significan un buen rendimiento de la red.

El tiempo de respuesta también es determinante en el rendimiento de una red. La velocidad en la transferencia de datos puede ser alta, pero puede ser lenta la velocidad que tarda en contactarse un nodo con otro. En algunas redes, el tiempo de respuesta es fundamental. Por ejemplo, imaginen un juego online en donde las reacciones de los jugadores demoren 2 o 3 segundos en llegar a destino. [3]

2.4.2. Medida de Eficiencia

Uso racional de los recursos con que se cuenta para alcanzar un objetivo predeterminado. A mayor eficiencia menor la cantidad de recursos que se emplearán, logrando mejor optimización y rendimiento. [4]

3. CAPÍTULO III:

ANALISIS COMPARATIVO DE LOS FRAMEWORK PHP

3.1. Determinación de los Frameworks a comparar

En la actualidad la existencia de varios Frameworks para PHP, se ha realizado una investigación previa de estos, publicados en el Internet se han elegido los tres mejores Frameworks por las razones que se explican a continuación.

- En el portal del TOP 10 de Ranking PHP Frameworks "www.phpframeworks.com" ha realizado una votación entre 10 Frameworks dando como resultado a los siguientes: [16]

    - Yii 106 votos
    - CodeIgniter 90 votos
    - CakePhp 73 votos
    - Zend 60 votos
    - Symfony 49 votos
    - PHPDevShell 38 votos
    - Prado 26 votos
    - Akelos 24 votos
    - QPHP 5 votos
    - ZooP 5 votos

- En el sitio Mundo Geek “www.mundogeek.net” sugiere los Frameworks más populares respondiendo a una pregunta planteada por nuestro lector Programador, os deja unas cuantas líneas con Mi opinión sobre algunos Frameworks para desarrollo de aplicaciones web con PHP más populares de la actualidad: [14]

    - Zend Framework
    - Symfony
    - CakePhp
    - CodeIgniter

- En el sitio Webmaster UM www.elwebmaster.com En los últimos años ha habido una explosión de Frameworks PHP. Existen grandes debates acerca de cuál es el mejor Frameworks, porque la realidad es que no todos los Frameworks, le sirven a cualquiera. Aquí hay una simple revisión acerca de 5 de las mejores y más populares opciones: [8]

    - Zend Framework
    - CakePhp
    - Symfony
    - CodeIgniter
    - Seagull

- En el sitio AdminWeb “www.admixweb.com” PHP es probablemente la lengua más popular en el mundo en lo que respecta a la flexibilidad y facilidad de uso con respecto a la construcción de aplicaciones web y los Frameworks. PHP tiene una curva de aprendizaje muy grande lo que significa que, independientemente de que haya venido de un fondo que no sea de programación, usted puede aprender fácilmente PHP. da los 15 mejores Frameworks que son fáciles y ligeros de utilizar: [1]

    - CakePhp
    - Vork
    - Simple PHP Framework

- Zend Framework
- Fat Free
- Akelos
- Flourish
- Konstrukt
- CodeIgniter
- LightVC
- Adroit
- Tekuna
- UltraLite
- Zephyr

➢ La encuesta que se realizó por medio de un entorno muy práctico como es el Facebook a estudiantes y programadores los resultados del sitio "www.facebook.com/Que_Framework_PHP_conoces_y_has_utilizado" fueron los siguientes:

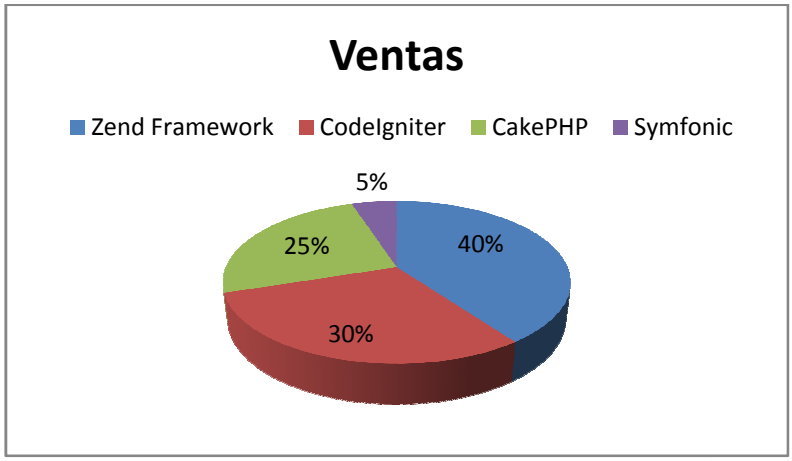


Ilustración 2 Valorización de la encuesta

- Zend Frameworks
- CodeIgniter
- CakePhp
- Symfonic

Para la realización de este análisis comparativo se han seleccionado los siguientes Frameworks Zend Framework, CakePhp y CodeIgniter por los siguientes motivos:

- Los tres Frameworks gozan de una popularidad entre los desarrolladores PHP por la publicación de diferentes sitios web.
- Los resultados de la encuesta que se realizó por medio del Facebook nos dio como veredicto a estos los Frameworks más usados por desarrolladores y estudiantes informáticos
- Tienen una excelente documentación en línea y comunidad de usuarios.
- La factibilidad de uso es muy buena ya que se basan en el modelo vista controlador.
- La optimización de recursos es fundamental para el mejor desempeño de los sitios web.
- El manejo y la abstracción de base datos es realmente bueno.

3.2. Análisis de los Frameworks seleccionados

A continuación se analizar más a cerca de cada una de las herramientas definidas anteriormente:

3.2.1. Zend Frameworks

Es un código abierto, el objeto marco de aplicación web, orientada a PHP 5. ZF a menudo se llama una biblioteca de componentes, ya que tiene muchos componentes débilmente acoplados que se pueden utilizar más o menos independientes. Pero ZF también proporciona un avanzado Model-View-Controller (MVC) la aplicación que se puede utilizar para establecer una estructura básica para el ZF. Una lista completa de los componentes de ZF, junto con una breve descripción se puede encontrar en la vista general de componentes. [26]

3.2.1.1. Historia.

Zend Framework fue concebido en el inicio de 2005 mientras muchos nuevos Frameworks, tales como Ruby on Rails y Spring Framework, estaban ganando popularidad en la comunidad web development. ZF fue públicamente anunciado por primera vez en la Zend Conference.A la vez, ningún Frameworks ampliamente usado ha sido disponible para la comunidad PHP para llenar completamente necesidades de desarrollo web similares. Los proyectistas del ZF buscaron combinar características de uso final y rapid application development (RAD) de esos nuevos Frameworks con la simplicidad, apertura y practicidad del mundo real que es altamente valorada en la comunidad PHP.

Típicamente, escenarios de uso de desarrollo específicos son implementados usando componentes de software más genéricos a través de configuración automática y/o generación de código. Nos da la comunidad ZF optó por el desarrollo completo y prueba de esos componentes esenciales antes de comenzar a trabajar en la simplificación de tareas de desarrollo tales como migraciones de banco de datos, generación de scaffolding, y configuración y creación de proyecto.

Esa práctica ha sido objeto de algunas críticas desde que algunas funcionalidades consideradas por muchos como necesarias para un release general para Frameworks de aplicación web modernos fueron deseadas para futuros releases del ZF. Muchos usuarios ZF, sin embargo, tienen hallazgo tales componentes de software genéricos más reusables y extensivos en la implementación de sus aplicaciones. ZF también busca promover las mejores prácticas de desarrollo web en la comunidad PHP; convenciones no son tan comúnmente usadas en el ZF como en muchos otros Frameworks. Apropiadamente las sugerencias son substituidas por patrones razonables de configuración que pueden ser sobrescritos por cada requisito específico de la aplicación ZF.

3.2.1.2. Licenciamiento

Zend Framework está licenciado bajo Open Source Initiative(OSI)- approved New BSD License, y todos los contribuidores de código deben firmar un Contributor License Agreement (CLA) basado en el Apache Software Foundation's CLA. El licenciamiento y las políticas de contribución fueron establecidos para frustrar cualesquier cuestiones de propiedad intelectual por usuarios ZF comerciales.

3.2.1.3. Características

- Todos los componentes son PHP 5 completamente orientados a objeto y tiene conformidad con Y STRICT
- Arquitectura úsela-gana con débil acoplamiento de componentes e interdependencias mínimas
- Implementación MVC extensible soportando esbozos y templates basados en PHP por patrón
- Implementación flexible de tablas Gateway para acensar datos de un banco de datos relacional en un ambiente orientado a objetos
- Soporte para múltiples sistemas de bancos de datos, incluyendo MySQL, Oracle, IBM DB2, Microsoft SQL Server, PostgreSQL,SQLite, e Informix Dynamic Server
- Autenticación y autorización basada en ACL usando una variedad de sistemas de backend
- Filtro de datos y validación para fortalecimiento de la seguridad de la aplicación
- Gestión de sesión
- Componente de configuración para promover una gestión de configuración consistente a través de Zend Framework y aplicaciones ZF
- Composición y entrega de email, recuperación vía mbox, maildir, POP3 e IMAP4
- Indexación y búsqueda que soporta el formato de archivo índice Lucene
- Internacionalización y localización.
- Creación de formularios usando PHP, archivos de configuración o XML.

- Tecnologías de Identity 2.0 tales como Microsoft InfoCard y OpenID.
- Múltiples formatos para web services, inclundo XML-RPC, REST, y Google GData.
- Subsistema de caching flexible con soporte para muchos tipos de backends, tales como memory o un sistema de archivos.
- Componente de logging simple inspirado por log4j
- Componente nativo PHP para lectura, actualización y creación de documentos PDF
- Señalización de estructuras de datos PHP para y de JSON de modo a facilitar el desarrollo AJAX
- API para consumir RSS y alimentadores Atom
- Bibliotecas cliente para muchos repositorios de web services. [9]

3.2.2. CodeIgniter

Es un potente Frameworks PHP con un tamaño muy pequeño, construido para programadores PHP que necesitan una herramienta simple y elegante para crear aplicaciones web con todas las funciones.

Las características en y de ellas son una forma muy pobre de juzgar una aplicación ya que no dicen nada acerca de la experiencia del usuario, o cuan intuitivamente o inteligentemente es diseñado. Las características no revelan nada acerca de la calidad del código, o la performance, o la atención a los detalles, o las prácticas de seguridad.

La única forma de realmente juzgar una aplicación es probarla y llegar a conocer el código. InstalandoCodeIgniter es un juego de niños así que alentamos que haga eso.

3.2.2.1. Historia

CodeIgniter es un potente Frameworks PHP con un tamaño muy pequeño, construido para programadores PHP que necesitan una herramienta simple y elegante para crear aplicaciones web con todas las funciones. Si usted es un programador que vive en el

mundo real de cuentas de alojamiento compartido y clientes con los plazos, y si usted está pesadamente marcos grandes y bien cansado de indocumentados.

Antes de comenzar EllisLab Rick Ellis fue un músico, ingeniero de audio, y diseñador de sonido en Los Ángeles trabajando en películas, televisión y proyectos interactivos. Especialidad de Rick fue de audio basado en computadora, que más tarde serviría como una programación de computadoras.

Cuando la web surgió a mediados de los 90 se enamoró con el nuevo medio e inmediatamente se sumergió en ella. Pronto fue la construcción de sitios para sí mismo, a continuación, para otros, como su afición se convirtió en una empresa de desarrollo web.

En 2001, mientras que en Japón mezcla algunos shows de Disney, que comenzó a desarrollar una aplicación que su cliente podría utilizar para mantener el contenido del sitio sin necesidad de conocimientos técnicos. Cuando regresó de Japón tuvo un prototipo de trabajo, que instalado en Nancy Sinatra (hija de Frank Sinatra), página web, uno de sus clientes web en el momento. A ella le encantó, y Rick decidió llamar a su programa "pMachine" (Editorial de la máquina) y ponerla a disposición del público.

En 2002 pMachine fue puesto en libertad. Dentro de unos meses apareció en revistas importantes como Mac Mundial y adicto Mac, y en un libro sobre blogs publicado por McRae Hill. Entusiasmado por el éxito de su programa estaba disfrutando, Rick dio un salto de una fe y dejó su carrera de audio para centrarse en el software de tiempo completo. Dentro de seis meses iba a contratar a su primer empleado.

En 2003 se reunió Rick Envíen Lyne, el propietario de una pequeña empresa de alojamiento que se especializa en la gestión de sitios web de alto tráfico. Dentro de unos meses se puso en marcha una empresa de servicio completo de alojamiento diseñados específicamente para las necesidades de sitios dinámicos, con bases de datos.

En 2004 lanzamos nuestro sistema de publicación de próxima generación, ExpressionEngine, ofreciendo la facilidad de uso de una herramienta de los blogs y el poder de una completa herramienta de gestión de contenidos del sistema.

En 2005 se lanzará un módulo de foro de discusión para ExpressionEngine, por lo que EE la plataforma perfecta para los sitios de la comunidad.

En 2006 lanzamos CodeIgniter, una aplicación web, plataforma de desarrollo para desarrolladores de PHP.

3.2.2.2. Licenciamiento

La Licencia Pública General GNU es una licencia libre, sin derechos para software y otro tipo de obras.

Las licencias de la mayoría del software y otros trabajos prácticos están diseñados para quitarle a usted la libertad de compartir y modificar esos trabajos. Por el contrario, la Licencia Pública General GNU pretende garantizarle la libertad de compartir y modificar todas las versiones de un programa - para asegurarse de que sigue siendo software libre para todos sus usuarios. Nosotros, la Free Software Foundation, utilicen la Licencia Pública General GNU para la mayoría de nuestro software, sino que también se aplica a cualquier trabajo realizado de esta manera por sus autores. Usted también puede aplicarla a sus propios programas.

Cuando hablamos de software libre, nos estamos refiriendo a la libertad, no de precio. Nuestras Licencias Públicas Generales están diseñadas para asegurarse de que usted tiene la libertad de distribuir copias de software libre (y cobrar por ellos si lo desea), que recibe el código fuente o que pueda conseguirlo si lo quiere, de que puede cambiar el software o piezas de uso de él en nuevos programas libres, y que usted sepa que puede hacer estas cosas.

Para proteger sus derechos, necesitamos evitar que otros le nieguen estos derechos o pedirle que renuncie a ellos. Por lo tanto, usted tiene ciertas responsabilidades si distribuye copias del software, o si lo modifica: responsabilidades de respetar la libertad de los demás.

Por ejemplo, si distribuye copias de tal programa, ya sea gratuitamente o cambio de una contraprestación, debe transmitir a los destinatarios de las mismas libertades que ha recibido. Usted debe asegurarse de que ellos también reciban o puedan conseguir el código fuente. Y debe mostrarles estas condiciones de forma que conozcan sus derechos.

Los desarrolladores que usen la GPL GNU protegen sus derechos con dos pasos: (1) imponen derechos al software, y (2) le ofrecemos esta licencia que le da permiso legal para copiar, distribuir y / o modificarlo.

Para los desarrolladores y autores de protección, la GPL expone claramente que no hay ninguna garantía para este software libre. Tanto para los usuarios y autores, la GPL exige que las versiones modificadas ser identificadas como tales, de modo que sus problemas no se atribuye erróneamente a los autores de versiones anteriores.

Algunos dispositivos están diseñados para denegar a los usuarios el acceso a instalar o ejecutar versiones modificadas del software dentro de ellos, a pesar de que el fabricante pueda hacerlo. Esto es fundamentalmente incompatible con el objetivo de proteger la libertad de los usuarios a cambiar el software. El patrón sistemático de tal abuso se produce en el área de productos para uso personal, que es precisamente donde es más inaceptable. Por lo tanto, hemos diseñado esta versión de la GPL para prohibir la práctica de dichos productos. Si apareciesen problemas similares en otros ámbitos, estamos dispuestos a extender esta disposición a los dominios en futuras versiones de la GPL, según sea necesario para proteger la libertad de los usuarios.

Por último, todo programa está constantemente amenazado por las patentes de software. Los Estados no deben permitir que las patentes restrinjan el desarrollo y uso

de software en ordenadores de propósito general, pero en aquellos que lo hacen, queremos evitar el peligro especial de que las patentes aplicadas a un programa libre puedan hacerlo propietario. Para evitar esto, la GPL asegura que las patentes no se pueden utilizar para hacer que el programa no-libre.Los términos exactos y las condiciones para la copia, distribución y modificación. (GNU)

3.2.2.3. Características

- Sistema Basado en Modelo-Vista-Controlador
- Compatible con PHP
- Extremadamente Liviano
- Clases de base de datos llenas de características con soporte para varias plataformas.
- Soporte de Active Record para Base de Datos
- Formulario y Validación de Datos
- Seguridad y Filtro XSS
- Manejo de Sesión
- Clase de Envío de Email. Soporta Archivos Adjuntos, email de texto/HTML, múltiples protocolos (sendmail, SMTP, and Mail) y más.
- Librería de Manipulación de Imagen (cortar, redimensionar, rotar, etc.). Soporta
- GD, ImageMagick, y NetPBM
- Clase de Carga (upload) de Archivo
- Clase de FTP
- Localización
- Paginación
- Encriptación de Datos
- Puntos de referencia
- Cacheo de páginas enteras
- Historial de Errores
- Perfilando la Aplicación
- Scaffolding
- Clase de Calendario

- Clase de Agente del Usuario
- Clase de Codificación Zip
- Clase de Motor de Plantillas
- Clase de Trackback
- Librería XML-RPC
- Clase de Prueba de Unidad
- URLs amigables a motores de búsqueda
- Ruteo de URI Flexible\
- Soporte para Ganchos, Extensiones de Clase y Plugins[7]

3.2.3. CakePhp

CakePhp es un marco de desarrollo rápido para PHP, libre, de código abierto. Se trata de una estructura que sirve de base a los programadores para que éstos puedan crear aplicaciones Web. Nuestro principal objetivo es que puedas trabajar de forma estructurada y rápida, sin pérdida de flexibilidad.

Con CakePhp el desarrollo web ya no es monótono porque ofrecemos las herramientas para que empieces a escribir el código que realmente necesitas: la lógica específica de tu aplicación. Consigue una copia de CakePhp, empieza con lo verdaderamente importante y no reinventes la rueda cada vez que te incorpores a un nuevo proyecto.
CakePhp tiene un equipo de desarrolladores y una comunidad activos, lo que añade valor al proyecto. Con CakePhp, además de no tener que reinventar la rueda, el núcleo de tu aplicación se mejora constantemente y está bien probado.

3.2.3.1. Historia

CakePhp se inició en 2005. La comunidad ha crecido desde entonces y ha generado varios sub-proyectos. CakePhp no es un puerto de Ruby on Rails para PHP, pero se apropia de muchos de sus conceptos útiles.
En octubre de 2009, el director del proyecto Garrett Woodworth y desarrollador Nate Abele abandonó el proyecto para centrarse en sus propios proyectos. El equipo de

desarrollo restante siguió centrándose en la hoja de ruta original que se ha definido previamente. [23]

3.2.3.2. Licenciamiento

El licenciamiento que posee CakePhp es MIT License. La licencia del MIT es una licencia de software libre con origen en el Instituto de Tecnología de Massachusetts (MIT). Se autoriza, de forma gratuita, a cualquier persona que ha obtenido una copia de este software y archivos asociados de documentación (el "Software"), para tratar en el Software sin restricción, incluyendo sin ninguna limitación en lo que concierne los derechos para usar, copiar, modificar, fusionar, publicar, distribuir, sublicenciar, y / o vender copias de este Software, y para permitir a las personas que usan el Software para hacerlo, con sujeción a las siguientes condiciones:

El aviso de copyright anterior y este aviso de permiso se incluirán en todas las copias o partes sustanciales de este Software.

El software se entrega "tal cual", sin garantía de ningún tipo, expresa o implícita, no limitado a la garantías de comercialización, capacidad de hacer y de no infracción de copyright. En ningún caso los autores o titulares del copyright serán responsables de ninguna reclamación, daños u otras responsabilidades, ya sea en un litigio, agravio o de otro modo, derivadas de, fuera de o en conexión con el software su utilización u otras operaciones en el software. (Open Source)

3.2.3.3. Características

- Comunidad activa y amistosa
- Licencia flexible
- Compatible con PHP4 y PHP5
- CRUD integrado para la interacción con la base de datos
- Soporte de aplicación [scaffolding]
- Generación de código

- Arquitectura Modelo Vista Controlador (MVC)
- Despachador de peticiones [dispatcher], con URLs y rutas personalizadas
- Validación integrada
- Plantillas rápidas y flexibles (sintaxis de PHP, con ayudantes[helpers])
- Ayudantes para AJAX, Javascript, formularios HTML y más
- Componentes de Email, Cookie, Seguridad, Sesión y Manejo de solicitudes
- Listas de control de acceso flexibles
- Caché flexible
- Localización[6]

3.3. Determinación de parámetros de comparación

Para la comparación de los Frameworks PHP, en su ámbito del rendimiento se debe considerar una serie de aspectos, siendo cada índice seleccionado importante para el cálculo del mejor rendimiento, se ha dividido en 5 indicadores cada uno con sus índices para determinar por separado sus potencialidades y debilidades, Se describe a continuación los parámetros de comparación para la determinación del mejor Frameworks.

- Ingeniería de Carga
  - CPU %
  - Memoria %
  - Ancho de banda de entrada
  - Ancho de banda de salida

- Línea Base
  - Consumo usuarios x prueba
  - Consumo usuarios por transacción
  - Velocidad de la red
  - Mínimo de ancho de banda requerido
  - Velocidad de la red para el total de usuarios.

- Carga Transaccional de Alta
    - Ingreso de carga transaccional
    - Duración transaccional

- Carga Transaccional de Baja
    - Baja de carga transaccional
    - Duración transaccional

- Integridad
    - User
    - Éxito
    - Fallas
    - Promedio transacción del paquete

3.3.1. Indicador 1: Ingeniería de Carga.

La sección de carga de los motores proporciona métricas sobre el rendimiento de los motores de carga durante la prueba.

Tabla I Descripción Indicador 1: Ingeniería de Carga

		Descripción
Indicador 1	**Ingeniería de carga**	La secciónde Ingeniería de carga delos motoresproporciona métricassobre el rendimientode los motoresde cargadurante la prueba
[índice 1.1]	CPU%	Describe el porcentaje de utilización del CPU al final del periodo de prueba.
[índice 1.2]	Memoria	Describe el porcentaje de utilización de la memoria al final del periodo de prueba.
[índice 1.3]	Ancho de banda entrada	Describe el total delos datos detasa de

		transferencia(en el motor de carga)durante elperíodo de la muestra
[índice 1.4]	Ancho de banda de salida	Describe el total delos datos detasa de transferencia(salida del motor de carga)durante elperíodo de la muestra

3.3.2. Indicador 2: Línea Base

Describe la baselas estimacionesde los requisitosde ancho de bandapara cadacaso de pruebaen la configuración decarga, así comoel ancho de bandatotal requeridopara la prueba.

Tabla II Descripción Indicador 2: Línea Base

		Descripción
Indicador 2	**Línea Base**	Describe la baselas estimacionesde los requisitosde ancho de bandapara cadacaso de pruebaen la configuración decarga, así comoel ancho de bandatotal requeridopara la prueba.
[índice 2.1]	Consumo Usuarios x prueba	Describe el mínimo ancho de banda de las escalas de necesidades, asociado al número de usuarios.
[índice 2.2]	Consumo Usuario x transacción	Describe el mínimo ancho de banda de las escalas de necesidades, asociado a un usuario
[índice 2.3]	Velocidad de la red	Describe la velocidad de conexión simulada por los usuarios virtuales
[índice 2.4]	Mínimo ancho de banda requerido(Estimado) Para el número de usuarios	El ancho de bandatotal que seconsumepor la combinacióndecasos de

		pruebaconfigurado. Esto supone quela redestá completamente saturaday el uso deancho de bandase distribuyede manera uniforme.
[índice 2.5]	Velocidad de red para el total de usuarios (configurado)	Lavelocidaddel usuario de redtotalse describe elancho de banda totaldisponible para los usuariossegún la configuración. Si los usuariosestánen una red localy deeste totalexcede la capacidad dela red, la configuraciónse debe ajustarsegún sea necesario parala estimación más precisa.

3.3.3. Indicador 3: Carga Transaccional de Alta

Describe el tiempo de duración de las cargas transaccionales de alta del test, dependiendo del número de cargas transaccionales

Tabla III Descripción Indicador 3: Carga Transaccional de Alta

		Descripción
Indicador 3	**Carga Transaccional de Alta**	Describe el tiempo de duración de las cargas transaccionales de alta del test, dependiendo del número de cargas transaccionales
[índice 3.1]	Ingreso de carga transaccional	Cargas transaccional por usuario de la prueba
[índice 3.2]	Duración transaccional	Identifica el tiempo de mostrar la solicitud enviada

3.3.4. Indicador 4: Carga Transaccional de Baja

Describe el tiempo de duración de las cargas transaccionales de baja del test, dependiendo del número de cargas transaccionales

Tabla IV Descripción Indicador 4: Carga Transaccional de Baja

		Descripción
Indicador 4	**Carga Transaccional de Baja**	Describe el tiempo de duración de las cargas transaccionales baja del test, dependiendo del número de cargas transaccionales
[índice 4.1]	Baja de carga transaccional	Cargas transaccional por usuario de la prueba
[índice 4.2]	Duración transaccional	Identifica el tiempo de mostrar la solicitud enviada

3.3.5. Indicador 5: Integridad

Describe el porcentaje de la información enviada, recibida, llego íntegramente y si hubo errores o fallas.

Tabla V Descripción Indicador 5: Integridad

		Descripción
Indicador 5	**Integridad**	Describe el porcentaje de la información enviada, recibida, llego íntegramente y si hubo errores o fallas.
[índice 5.1]	User	Describe la cantidad de usuarios que ingresaron al test en el periodo configurado
[índice 5.2]	Éxito	Describe la cantidad de transacciones finalizadas sin errores

[índice 5.3]	Fallas	Describe la cantidad de transacciones que no se completaron correctamente(se volvieron a reenviar)
[índice 5.4]	Promedio transacción del paquete	Describe el total delos datos detasa de transferencia(salida del motor de carga)durante elperíodo de la muestra

3.4. Descripción de los módulos de pruebas

Los módulos de prueba son escenarios que ayudan a verificar y obtener datos para determinar que Framework es el mejor en el rendimiento ante las diferentes configuraciones.

Los módulos que se desarrollarán serán implementados en los 3 Frameworks seleccionados cada uno con su manera particular de desarrollo siguiendo el mismo modelo de arquitectura de desarrollo llamado MVC (modelo vista, controlador),
En cada Framework se probara los mismos escenarios y se obtendrá los resultados mediante un software especial para calcular el rendimiento de hardware y software.

Para el desarrollo de la aplicación web, se tendrá 3 módulos que son:

- Módulo de Navegación de Interfaces
- Módulo de transacciones de alta y baja.
- Módulo de funciones.

A continuación se explicará cada uno de los módulos para su posterior desarrollo e implementación.

3.4.1. Módulo de Navegación de interfaces

Conocido también como capa de presentación.El módulo describe la manera en que el usuario interactúa con la aplicación y la información almacenada en la base de datos, con este módulo se verificara la velocidad de interpretación del código Php para su posterior presentación en código HTML en forma visual de interfaces.

Este móduloestá compuesto por todos los elementos de una interfaz tales como botones, navegación por tabulaciones, hipervínculos, código Php, llamadas a librerías de interfaz de los Frameworks.

3.4.2. Módulo de transacciones de alta y baja

En este módulo se encuentra la manera de acceder a la información desde y hacia una base de datos.

Con este módulo se probara el rendimiento y la capacidad de carga que puede soportar el Frameworks en cuanto a manejar grandes cantidades de transacciones al mismo tiempo, así como su integridad en cuanto a la transmisión de la información, es decir si llega a información con errores o falla en élenvió.

3.4.3. Módulo de funciones

En este módulo se encuentra la lógica del negocio, la programación de funciones y procedimientos para que los módulos realicen las tareas dinámicamente, ya sea generando interfaces o realizando transacciones como consultas, inserciones e eliminaciones.

Con la iteración en este módulo probaremos la capacidad y mejores prácticas de programación en las librerías de cada Frameworks al momento de llamar a sus funciones para generar las interfaces e información requerida por el usuario.

3.5. Desarrollo de los módulos de pruebas

Los distintos módulos dentro de la aplicación están programados bajo el lenguaje de programación Php y siguiendo la arquitectura MVC Modelo vista controlador que es un patrón de arquitectura de las aplicaciones software.

Todos los módulos siguen el mismo patrón de diseño MVC y cada módulo en diferente Frameworks tiene su manera estándar de realizar las acciones para llamar, crear, generar funciones,

Cada aplicación web desarrollada para cada Framework será implementada en la misma y realizaran idénticas acciones como funciones, para probar los escenarios planteados.

3.5.1. Zend Framework

El desarrollo de la aplicación web utilizando ZendFrameworksiguiendo la arquitectura Modelo Vista controlador, como indica la ilustración.Los principales componentes de Zend Frameworks dentro de su estructurason:

- Application
- Controller
- Models
- Views
- Docs
- Library
- public

Dentro de la estructura los tres componentes más importantes son

*Controller:*El controlador une las partes específicas del modelo y de la vista para asegurar que los datos correctos se muestren en la página.

*Modelo:*La parte del modelo de la aplicación es la que concierne a los datos específicosque se muestran en la interfaz, el modelo generalmente está relacionado con la parte lógica de negociación y concierne a la carga y guardado en las bases de datos.

Vista: consiste en segmentos de la aplicación relacionados con laInterfaz gráfica del usuario. Usualmente ésta se realiza en HTML.

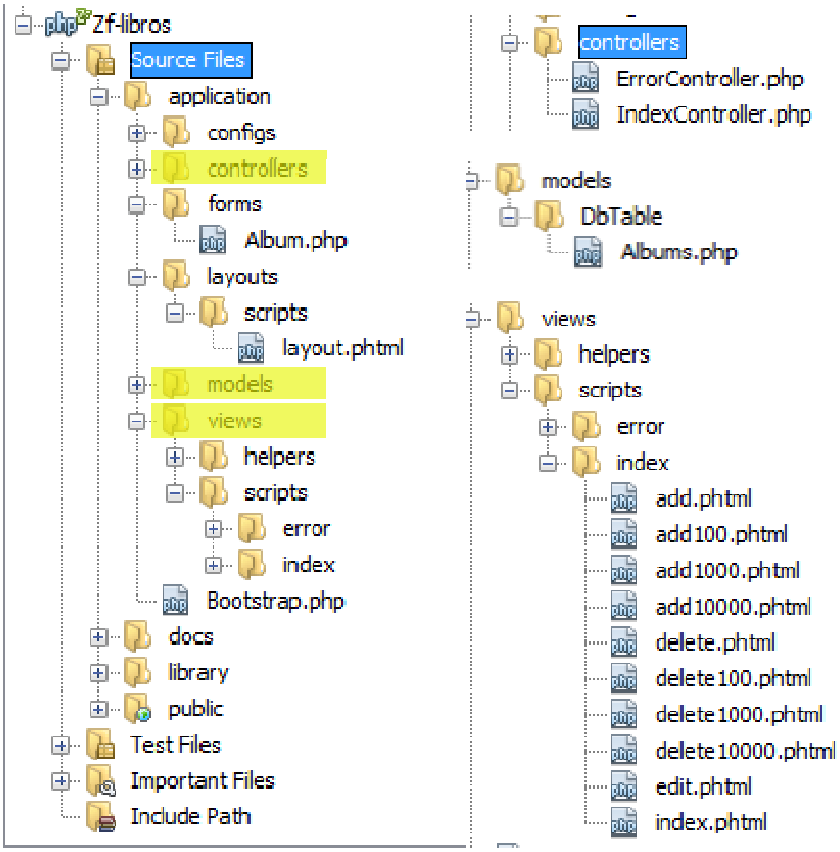


Ilustración 3 Estructura Detallada de Zend Framework MVC Ver Anexo 1

3.5.1.1. Módulo de Navegación de Interfaces

Dentro del módulo de interfaces del usuario, se ha realizado una interfaz amigable con funciones propias para una navegación intuitiva y rápida, estas funciones son paneles de navegación que nos permitirán interactuar en una página y otra, la interfaz nos permitirá capturar el tiempo que dura el proceso de iteración entre los elementos creados.

Ilustración 4 Interfaz Principal del Cliente Zend

Ilustración 5 Interfaz de Presentación de Información desde la Base de Datos, Zend

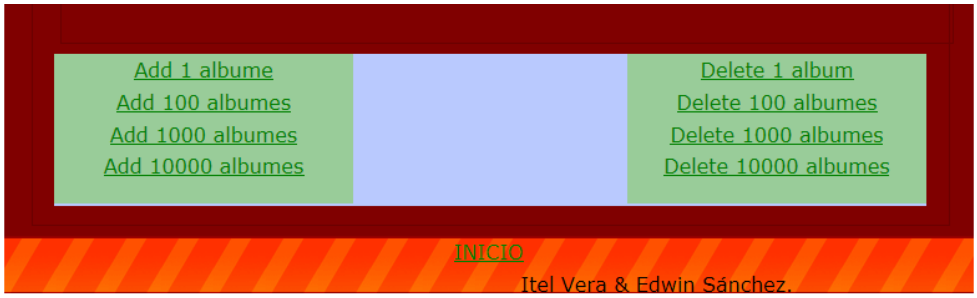


Ilustración 6 Interfaz de Links de Funciones de Carga Transaccional, Zend

3.5.1.2. Módulo de Funciones y transacciones de alta

Dentro de la arquitectura MVC se encuentra la estructura del modelo que es la manera de acceder a la base de datos, donde se implementa todas las funciones para el ingreso, actualización desde y hacia la base de datos.

Ilustración 7 Interfaz de Funciones de Ingreso y Actualización de la Base de Datos, Zend

3.5.1.3. Módulo de funciones y transacciones de baja

Dentro de la arquitectura MVC se encuentra la estructura del modelo que es la manera de acceder a la base de datos, donde se implementa todas las funciones para la eliminación, actualización desde y hacia la base de datos.

Ilustración 8 Interfaz de Funciones de Eliminación de la Base de Datos, Zend

3.5.2. CakePhp Framework

Estructura de datos del Patrón de diseño MVC, se detalla la manera de organización de clases y funciones dentro del Frameworks.

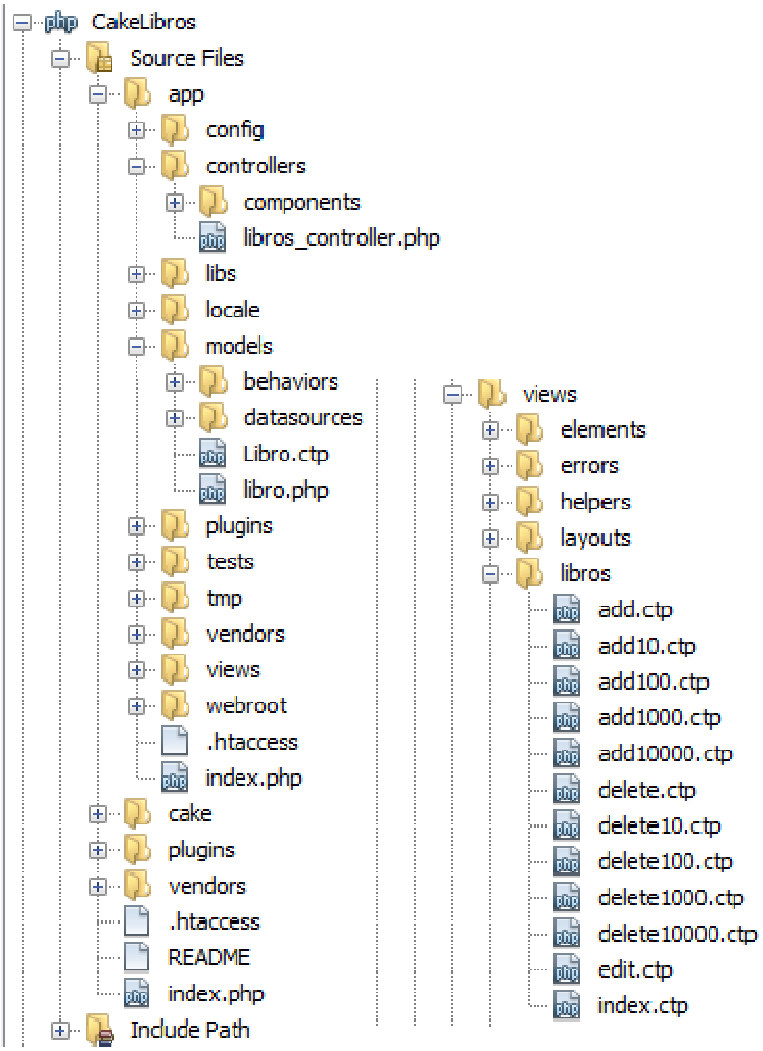


Ilustración 9 Estructura detallada de CakePhp Frameworks MVC Ver Anexo 1

3.5.2.1. Módulo de navegación de interfaces

Se nuestra la iteración entre los elementos principales de la interfaz, como son el Área de trabajo, los links, botones y demás elementos que forman una interfaz.

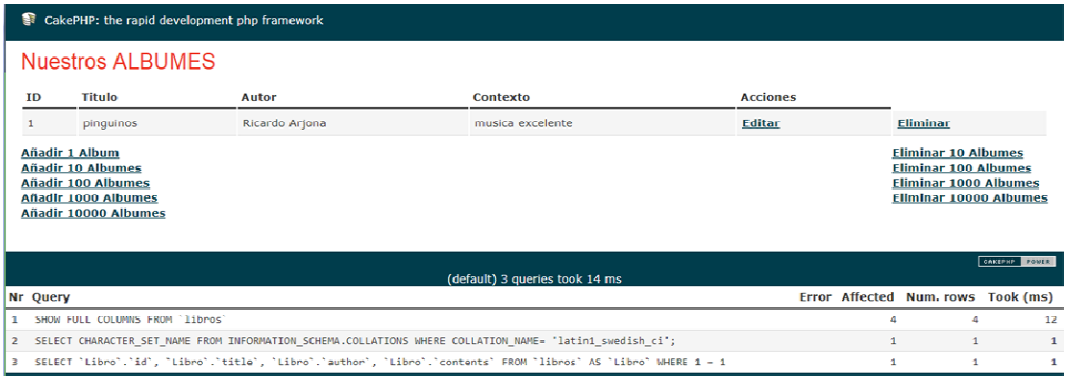


Ilustración 10Interfaz de Navegación de FuncionesCakePhp.

3.5.2.2. Módulo defunciones y transacciones de alta

Dentro del módulo de interfaz tenemos un modular dedicado al ingreso de la información, para ingreso y edición de la base de datos.

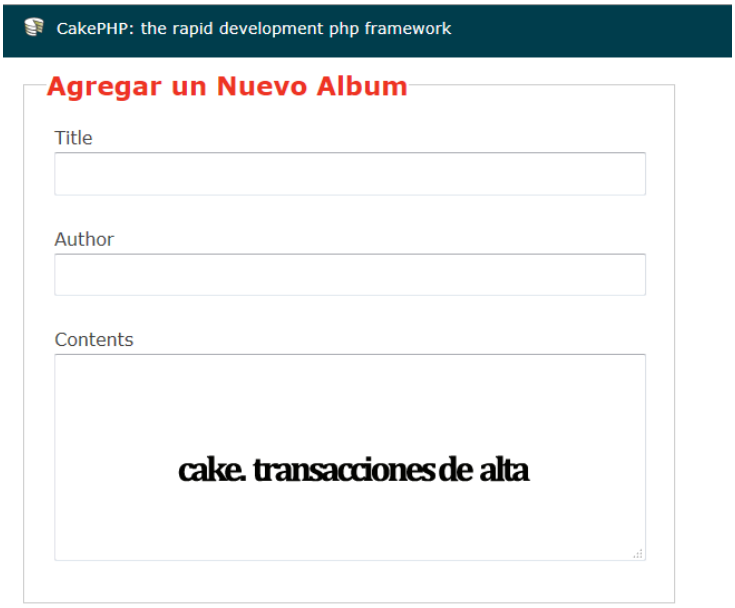


Ilustración 11Interfaz de Ingreso de Información a la Base de DatosCakePhp.

3.5.2.3. Módulo de funciones y transacciones de baja

Se muestra la captura del módulo de funciones, donde se detalla las funciones que realiza para capturar la información sobre las cargas transaccionales que realiza.

Ilustración 12 Interfaz de Funciones para Realizar Cargas TransaccionalesCakePhp.

3.5.3. CodeIgniter Framework

Patrón de diseño MVC, se detalla la estructura interna del FrameworksCodeIgniter, destacando sus tres carpetas principales que son:

- Controller
- Models
- Views.

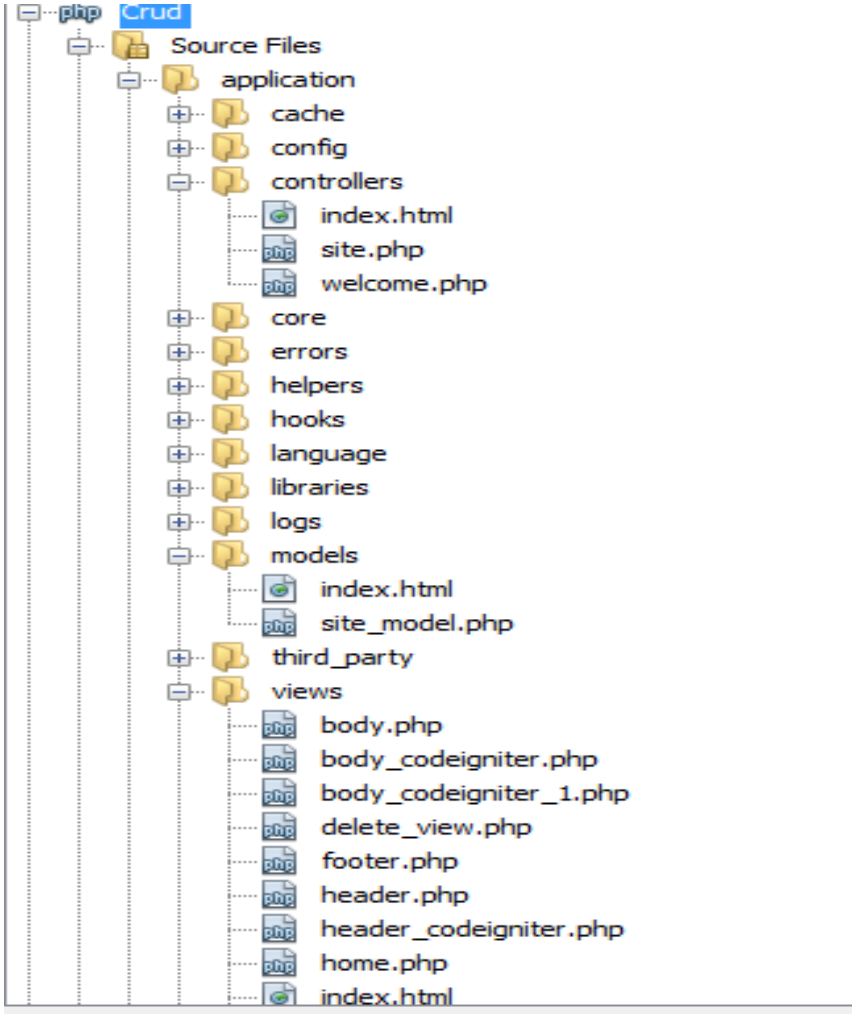


Ilustración 13 Estructura Detallada de CodeIgniter Framework MVC Ver Anexo 1.

3.5.3.1. Módulo de navegación de interfaces

El módulo desarrollado en CodeIgniter Frameworks muestra una interfaz amigable, siguiendo el estándar diseño de interfaces en cuanto a la presentación y navegabilidad.

Ilustración 14MóduloPrincipal de Navegación de Interfaces CodeIgniter

3.5.3.2. Módulo de funciones y transacciones de alta

En el módulo se muestra la manera de agregar información a la base de datos desde esta interfaz, que está creada con un formulario de relleno para agregar la información y luego de su ingreso a la base de datos, ser desplegada nuevamente en una tabla dinámica.

Ilustración 15 Módulo de Ingreso de la Información a la Base de Datos CodeIgniter.

3.5.3.3. Módulo de funciones y transacciones de baja

Modular desarrollado en CodeIgniter para la eliminación de información de la base de datos y su posterior presentación en una tabla dinámica, la consulta a las tablas de la base de datos son obtenidas mediante el método active record, que es una función que sin realizar consultas SQL obtienes toda la información de una tabla de la base de datos.

Ilustración 16 Interfaz de Funciones de Eliminación de la Base de Datos.

3.6. Análisis Comparativo

Los resultados de los indicadores con sus respectivos índices se realizan un cuadro comparativo de los tres Frameworks que fueron configurados con iguales parámetros carga transaccional para obtener resultados precisos de que Frameworks desempeño el mejor rendimiento.

3.6.1. Indicador 1:Ingeniería de Carga

3.6.1.1. Determinación del Indicador

La secciónde Ingeniería de carga posee 4 índices cada uno de los cuales mantuvo 4 pruebas diferentes. Sus resultados obtenidos serán evaluados tomando sus valores mínimos, máximos para cada índice, así realizar una escala de valorización y determinar los valores no satisfactorio, poco satisfactorio, satisfactorio y muy satisfactorio respectivamente.

Tabla VI Valorización para el Indicador 1: Ingeniería de Carga

Valor Cualitativo	Valor Cuantitativo	Valor Representativo
No Satisfactorio	1	♣
Poco Satisfactorio	2	♣♣
Satisfactorio	3	♣♣♣
Muy Satisfactorio	4	♣♣♣♣

- **Índice 1.1: CPU%:**En este índice se valorizará para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks).

Tabla VII Valorización del Índice 1.1: CPU

Resultados				Valor Cualitativo
Uno	Cien	Mil	Diez Mil	
37-42	69-78	54-62	86-100	No Satisfactorio
31-36	58-68	46-53	71-85	Poco Satisfactorio
26-30	49-57	36-45	56-70	Satisfactorio
21-25	40-48	24-35	39-55	Muy Satisfactorio

- **Índice 1.2: Memoria:**En este índice se valorizará para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks).

Tabla VIII Valorización del Índice 1.2: Memoria

Resultados				**Valor Cualitativo**
Uno	**Cien**	**Mil**	**Diez Mil**	
-	-	-	38	No Satisfactorio
-	33	-	37	Poco Satisfactorio
32	32	33	35	Satisfactorio
31	31	32	35	Muy Satisfactorio

- **Índice 1.3: Ancho de banda entrada:**En este índice se valorizará para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks).

Tabla IX Valorización del Índice 1.3: Ancho de banda Entrada

Resultados				**Valor Cualitativo**
Uno	**Cien**	**Mil**	**Diez Mil**	
-	5,2-5,8	7.9-8,4	9,1-10,6	No Satisfactorio
-	4,7-5.1	7,3-7,8	8,1-9	Poco Satisfactorio
-	4-4,6	6,8-7,2	6,5-8	Satisfactorio
1.1	3,3-4	6,4-6,7	4,8-6,4	Muy Satisfactorio

- **Índice 1.4: Ancho de banda de salida:** En este índice se valorizará para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks).

Tabla X Valorización del Índice 1.4: Ancho de banda de Salida

Resultados				**Valor Cualitativo**
Uno	**Cien**	**Mil**	**Diez Mil**	
14,6-16	11-13,5	8-8,5	-	No Satisfactorio

16,1-18	13,6-15	8,6-9	-	Poco Satisfactorio
18,1-19,5	15,1-17,5	9,1-9,6	7,9-8,6	Satisfactorio
19,6-21	17,6-20,5	9,7-10,3	8,7-9,4	Muy Satisfactorio

3.6.1.2. Valoraciones

Tabla XI Resultados del Indicador 1: Ingeniería de Carga, después de la obtención de la ejecución de las diferentes pruebas Ver Anexo 2.

	INGENIERIA DE CARGA											
	CAKEPHP				**CODEIGNITER**				**ZEND**			
	Un	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**Diez Mil**
CPU%	24	78	55	100	42	40	24	39	21	57	62	95
Memoria %	32	33	32	38	31	32	33	35	31	31	32	38
ABE	1.1	3.5	8.4	10.6	1.1	3.3	6.4	4.8	1.1	5,8	7,8	6,9
ABS	14.6	11	8	7,9	17.4	14.7	9.3	7.9	21	20.5	10.3	9.4

3.6.1.3. Calificación

Tabla XII Calificación del Indicador 1: Ingeniería de Carga, se basarán en la calificación de las Tablas (6-10)

	INGENIERIA DE CARGA											
	CAKEPHP				**CODEIGNITER**				**ZEND**			
	Un	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**Diez Mil**
CPU%	4	1	1	1	1	4	4	4	4	3	1	1
Memoria%	3	2	4	1	4	3	3	4	4	4	4	1
ABE	4	4	1	1	4	4	4	4	4	1	2	3
ABS	1	1	1	3	2	2	3	3	4	4	4	4

Para sacar los porcentajes de cada índice se utilizará las siguientes formulas.
Dónde:

X = Equivale a CakePhp.
Y =Equivale a CodeIgniter
Z = Equivale a Zend.

C1(X)= Puntaje acumulativo del CPU de CakePhp
C2(X)= Puntaje acumulativo de Memoria de CakePhp
C3(X)= Puntaje acumulativo ABE de CakePhp
C4(X)= Puntaje acumulativo ABS de CakePhp

C1 (Y)= Puntaje acumulativo del CPU de CodeIgniter
C2 (Y)= Puntaje acumulativo de Memoria de CodeIgniter
C3 (Y)= Puntaje acumulativo ABE de CodeIgniter
C4 (Y)= Puntaje acumulativo ABS de CodeIgniter

C1 (Z)= Puntaje acumulativo del CPU de Zend
C2 (Z)= Puntaje acumulativo de Memoria de Zend
C3 (Z)= Puntaje acumulativo ABE de Zend
C4 (Z)= Puntaje acumulativo ABS de Zend

Vi= Resultado de las pruebas (Un-Cien-Mil y Diez mil).

$$\boldsymbol{C1(X)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C2(X)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C3(X)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C4(X)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi}$$

$$\boldsymbol{C1(Y)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C2(Y)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C3(Y)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C4(Y)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi}$$

$$\boldsymbol{C1(Z)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C2(Z)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C3(Z)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C4(Z)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi}$$

Pa(X)= Valor acumulativo de CakePhp

Pa (Y)= Valor acumulativo de CodeIgniter

Pa (Z)= Valor acumulativo de Zend

$$\boldsymbol{Pa(X) = \sum C1(X) + C2(X) + C3(X) + C4(X)}$$

$$\boldsymbol{Pa(Y) = \sum C1(Y) + C2(Y) + C3(Y) + C4(Y)}$$

$$\boldsymbol{Pa(Z) = \sum C1(Z) + C2(Z) + C3(Z) + C4(Z)}$$

PpT (X)= Porcentaje Parcial Total de CakePhp

PpT (Y)= Porcentaje Parcial Total de CodeIgniter

PpT (Z)= Porcentaje Parcial Total de Zend

Vp= Valor máximos de índices es de 16

$$\boldsymbol{PpT(X) = \left(\frac{Pa(X)}{Vp}\right) * 100}$$

$$\boldsymbol{PpT(Y) = \left(\frac{Pa(Y)}{Vp}\right) * 100}$$

$$\boldsymbol{PpT(Z) = \left(\frac{Pa(Z)}{Vp}\right) * 100}$$

Tabla XIII Valores y Porcentajes del Indicador 1: Ingeniería de Carga, con sus respectivos Índices

	INGENIERIA DE CARGA					
	CakePhp		**CodeIgniter**		**Zend**	
	Valor(X)	% PpT	Valor(Y)	% PpT	Valor (Z)	% PpT
CPU %: C1	7	43,75	13	81,25	9	56,25
Memoria %: C2	10	62,5	14	87,5	13	81,25
Ancho de Banda Entrada : C3	10	62,5	16	100	10	62,5
Ancho de Banda de Salida : C4	6	37,5	10	62,5	16	100

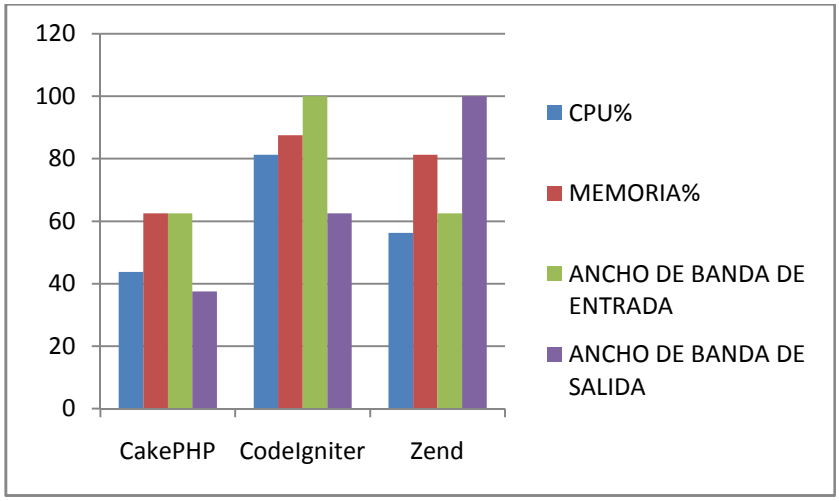


Ilustración 17Resultado por índice del Indicador 1: Ingeniería de Carga

Para sacar los porcentajes de cada índice se utilizará las siguientes formulas.

Dónde:

PT(X)= PorcentajeTotal de CakePhp

PT(Y)= Porcentaje Total de CodeIgniter

PT(Z)= Porcentaje Total de Zend

VA= Valor máximos de índices es de 64

$$\boldsymbol{PT(X) = \left(\frac{Pa(X)}{VA}\right) * 100}$$

$$\boldsymbol{PT(Y) = \left(\frac{Pa(Y)}{VA}\right) * 100}$$

$$\boldsymbol{PT(Z) = \left(\frac{Pa(Z)}{VA}\right) * 100}$$

Tabla XIV Valores y Porcentajes Finales del Indicador 1: Ingeniería de Carga.

	Ingeniería de Carga	
	Valor (Pa)	% (PT)
CakePhp : (X)	33	51,56
CodeIgniter : (Y)	53	82,81

Zend: (Z)	48	75

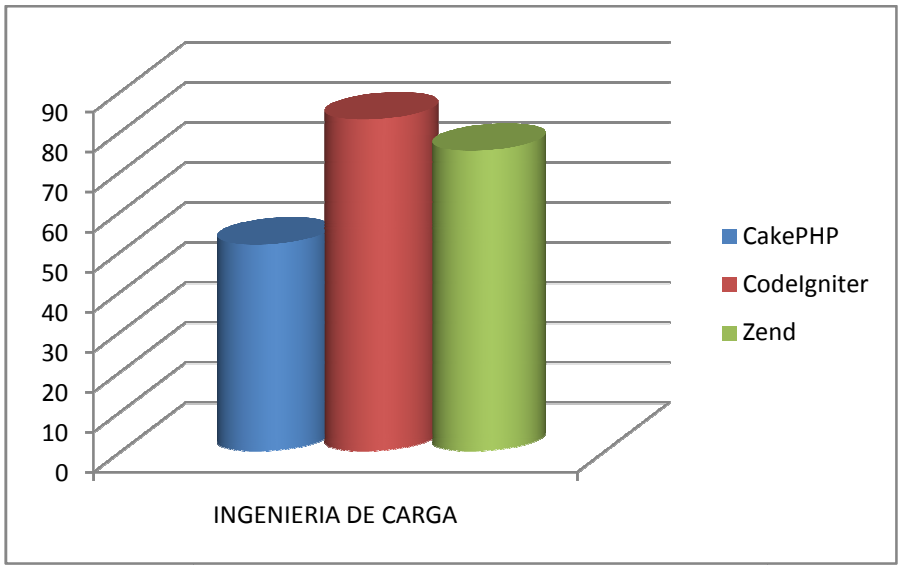


Ilustración 18 Resultado Finaldel Indicador 1: Ingeniería de Carga

3.6.1.4. Interpretación

Tabla XV Representación del Indicador 1: Ingeniería de Carga

	INGENIERIA DE CARGA											
	CAKEPHP				**CODEIGNITER**				**ZEND**			
	Un	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**Diez Mil**
CPU%	♣♣♣♣	♣	♣	♣	♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣	♣	♣
Memoria %	♣♣♣	♣♣	♣♣♣♣	♣	♣♣♣♣	♣♣♣	♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣
ABE	♣♣♣♣	♣♣♣♣	♣	♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣	♣♣	♣♣♣
ABS	♣	♣	♣	♣♣♣	♣♣	♣♣	♣♣♣	♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣

3.6.1.5. Descripción Resultados

Los resultados expuestos en la tabla XV “representación del Indicador 1: Ingeniería de carga” serán explicados por índices.

CPU%: El uso del CPU o también llamado procesador, en las pruebas realizas muestra que cake Frameworks mantiene una concordancia entre la aplicación y uso de recursos al obtener el mejor puntaje de uso de procesador en el ingreso de un único registro, seguido de Zend Frameworks.

En la utilización de procesador ejecutando la aplicación CodeIgniter supo mantener un equilibrio del uso del procesador, demostrando que tiene un alto rendimiento cuando se realiza ingresos de centenas, miles, y diez mil ingresos de carga.

Memoria%: Guiándose en los resultados de la tabla de referencia podemos afirmar que CodeIgniter tiene la mejor manera de administrar los recursos de memoria, una vez confirmado que al realizar las pruebas de ingreso de cargas, utilizo menos memoria en comparación con cake y Zend Frameworks.

Ancho de banda de entrada: los mejores puntajes obtiene CodeIgniter con 4 puntos al saber administrar los recursos de red de entrada, seguido muy de cerca de CakePhp y finalizando con Zend Frameworks, este último al poseer más controles de envió-recepción de datos utilizo más recursos de red de lo esperado, si bien esto Frameworks es ideal para el desarrollo de aplicaciones empresariales grandes.

Ancho de banda de salida: Destacando de manera sorprendente a Zend Frameworks por su utilización de la red en cuanto a la salida, ya que en la entrada de datos hizo todos los controles en la salida devuelve la información inmediatamente, seguido de cerca esta CodeIgniter que obtuvo un puntaje promedio en cuanto al uso de la red.

3.6.2. Indicador 2:Línea Base

3.6.2.1. Determinación del Indicador

La secciónde Línea Base posee 5 índicescada uno de los cuales mantuvo 4 pruebas diferentes. Sus resultados obtenidos serán evaluados tomando sus valores mínimos, máximos para cada índice, así realizar una escala de valorización y determinar los valores no satisfactorio, poco satisfactorio, satisfactorio y muy satisfactorio respectivamente.

Tabla XVI Valorización para el Indicador II: Línea Base

Valor Cualitativo	Valor Cuantitativo	Valor Representativo
No Satisfactorio	1	♣
Poco Satisfactorio	2	♣♣
Satisfactorio	3	♣♣♣
Muy Satisfactorio	4	♣♣♣♣

- **Índice 2.1: Consumo Usuarios x Prueba:**En este índice se valorizará para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks).

Tabla XVII Valorización del Índice 2.1: Consumo Usuarios x Prueba

Resultados				Valor Cualitativo
Uno	Cien	Mil	Diez Mil	
136-148	-	-	901-1000	No Satisfactorio
123-135	-	-	801-900	Poco Satisfactorio
111-122	12,1-13	96-104,9	701-800	Satisfactorio
98-110	11,3-12	85,7-95	642,6-700	Muy Satisfactorio

- **Índice 2.2: Consumo Usuario x Transacción:** En este índice se valorizará para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks).

Tabla XVIII Valorización del Índice 2.2: Consumo Usuarios x Transacción.

Resultados				**Valor Cualitativo**
Uno	**Cien**	**Mil**	**Diez Mil**	
-	-	-	-	No Satisfactorio
-	367-375	498,5-534	1,7-2,0 MB	Poco Satisfactorio
347,2-358	376-383	535-570	2,1-2,4 MB	Satisfactorio
359-368,2	384-392	571-603	2,5-2,7MB	Muy Satisfactorio

- **Índice 2.3: Velocidad de la red:** En este índice tendrá valorizaciones estándar para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks) por haber configurado previamente.

Tabla XIX Valorización del Índice 2.3: Velocidad de la Red

Resultados	**Valor Cualitativo**
-	No Satisfactorio
-	Poco Satisfactorio
-	Satisfactorio
1,5	Muy Satisfactorio

- **Índice 2.4: Mínimo ancho de banda requerido:** En este índice se valorizará para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks).

Tabla XX Valorización del Índice 2.4: Mínimo de ancho de Banda Requerido

Resultados				**Valor Cualitativo**
Uno	**Cien**	**Mil**	**Diez Mil**	
-	-	-	401-724,3	No Satisfactorio
133-148	126-132	91-104,9	121-400	Poco Satisfactorio
116-132	120-125	76-90	91-120	Satisfactorio
98-115	112,5-119	64,3-75	64,3-90	Muy Satisfactorio

- **Índice 2.5: Velocidad de red para el total de usuarios (configurado):** En este índice tendrá valorizaciones estándar para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks) por haber configurado previamente.

Tabla XXI Valorización del Índice 2.5: Velocidad de red para el total de Usuarios.

Resultados	Valor Cualitativo
-	No Satisfactorio
-	Poco Satisfactorio
-	Satisfactorio
1,5	Muy Satisfactorio

3.6.2.2. Valoraciones

Tabla XXII Resultados del Indicador 2: Línea Base después de la obtención de la ejecución de las diferentes pruebas Ver Anexo 2.

	LÍNEA BASE											
	CAKEPHP				**CODEIGNITER**				**ZEND**			
	Un	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**Diez Mil**
CUxP	98	11.3	85.7	724.3	148	12.5	104.9	642.6	128.3	13	103	992.1
CUxT	347.2	367	548.3	2.4 MB	365	378.4	498.5	1.7 MB	368.2	392	603	2.7 MB
VR	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5
MABR	98	112.5	85.7	724.3	148	125.3	104.9	64.3	128.3	132	103	99.2
VRTU	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5

3.6.2.3. Calificación

Tabla XXIII Calificación del Indicador 2: Línea Base, se basarán en la calificación de las Tablas (16-21)

	LÍNEA BASE											
	CAKEPHP				**CODEIGNITER**				**ZEND**			
	Un	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**Diez Mil**
CUxP	4	4	4	3	1	3	3	4	2	3	3	1
CUxT	3	2	3	3	4	3	2	2	4	4	4	4
VR	4	4	4	4	4	4	4	4	4	4	4	4
MABR	4	4	3	1	2	3	2	4	3	2	2	3
VRTU	4	4	4	4	4	4	4	4	4	4	4	4

Para sacar los porcentajes de cada índice se utilizará las siguientes formulas.
Dónde:

X = Equivale a CakePhp.
Y = Equivale a CodeIgniter
Z = Equivale a Zend.

C1(X)= Puntaje acumulativo del Consumo de Usuarios x PruebaCakePhp
C2(X)= Puntaje acumulativo de Consumo de Usuarios x Transacción CakePhp
C3(X)= Puntaje acumulativo Velocidad de la redCakePhp
C4(X)= Puntaje acumulativo Mínimo Ancho de Banda Requerido CakePhp
C5(X)= Puntaje acumulativo Velocidad de red para el total de usuarios CakePhp

C1(Y)= Puntaje acumulativo del Consumo de Usuarios x Prueba CodeIgniter
C2(Y)= Puntaje acumulativo de Consumo de Usuarios x Transacción CodeIgniter
C3(Y)= Puntaje acumulativo Velocidad de la red CodeIgniter
C4(Y)= Puntaje acumulativo Mínimo Ancho de Banda Requerido CodeIgniter

C5(Y)= Puntaje acumulativo Velocidad de red para el total de usuarios CodeIgniter

C1(Z)= Puntaje acumulativo del Consumo de Usuarios x Prueba Zend

C2(Z)= Puntaje acumulativo de Consumo de Usuarios x Transacción Zend

C3(Z)= Puntaje acumulativo Velocidad de la red Zend

C4(Z)= Puntaje acumulativo Mínimo Ancho de Banda Requerido Zend

C5(Z)= Puntaje acumulativo Velocidad de red para el total de usuarios Zend

Vi= Resultado de las pruebas (Un-Cien-Mil y Diez mil).

$$\boldsymbol{C1(X)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C2(X)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C3(X)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi}$$

$$\boldsymbol{C4(X)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi}; \;\; \boldsymbol{C5(X)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi}$$

$$\boldsymbol{C1(Y)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C2(Y)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C3(Y)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi}$$

$$\boldsymbol{C4(Y)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C5(Y)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi}$$

$$\boldsymbol{C1(Z)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C2(Z)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C3(Z)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi}$$

$$\boldsymbol{C4(Z)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C5(Z)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi}$$

Pa(X)= Valor acumulativo de CakePhp

Pa(Y)= Valor acumulativo de CodeIgniter

Pa(Z)= Valor acumulativo de Zend

$$\boldsymbol{Pa(X)} = \sum \boldsymbol{C1(X)} + \boldsymbol{C2(X)} + \boldsymbol{C3(X)} + \boldsymbol{C4(X)} + \boldsymbol{C5(X)}$$

$$\boldsymbol{Pa(Y)} = \sum \boldsymbol{C1(Y)} + \boldsymbol{C2(Y)} + \boldsymbol{C3(Y)} + \boldsymbol{C4(Y)} + \boldsymbol{C5(Y)}$$

$$\boldsymbol{Pa(Z) = \sum C1(Z) + C2(Z) + C3(Z) + C4(Z) + C5(Z)}$$

PpT (X)= Porcentaje Parcial Total de CakePhp

PpT (Y)= Porcentaje Parcial Total de CodeIgniter

PpT (Z)= Porcentaje Parcial Total de Zend

Vp= Valor máximos de índices es de 16

$$\boldsymbol{PpT(X) = \left(\frac{Pa(X)}{Vp}\right) * 100}$$

$$\boldsymbol{PpT(Y) = \left(\frac{Pa(Y)}{Vp}\right) * 100}$$

$$\boldsymbol{PpT(Z) = \left(\frac{Pa(Z)}{Vp}\right) * 100}$$

Tabla XXIV Valores y Porcentajes del Indicador 2: Línea Base con sus respectivos Índices

	LÍNEA BASE					
	CakePhp		**CodeIgniter**		**Zend**	
	Valor	%	Valor	%	Valor	%
Consumo de Usuarios x Prueba	15	93,75	11	68,75	9	56,75
Consumo de Usuarios x Transacción	11	68,75	11	68,75	16	100
Velocidad de la red	16	100	16	100	16	100
Mínimo Ancho de Banda requerido	12	75	11	68,75	10	62,5
Velocidad de red para el total de usuarios	16	100	16	100	16	100

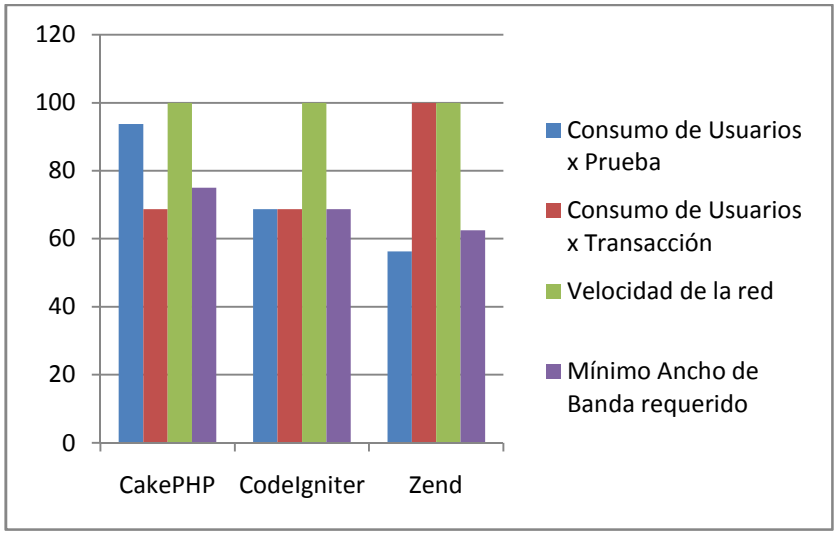


Ilustración 19 Resultado por índice del Indicador 2: Línea Base

Para sacar los porcentajes de cada índice se utilizará las siguientes formulas. Dónde:

PT(X)= Porcentaje Total de CakePhp

PT(Y)= Porcentaje Total de CodeIgniter

PT(Z)= Porcentaje Total de Zend

VA= Valor máximos de índices es de 80

$$\boldsymbol{PT(X) = \left(\frac{Pa(X)}{VA}\right) * 100}$$

$$\boldsymbol{PT(Y) = \left(\frac{Pa(Y)}{VA}\right) * 100}$$

$$\boldsymbol{PT(Z) = \left(\frac{Pa(Z)}{VA}\right) * 100}$$

Tabla XXV Valores y Porcentajes Finales del Indicador 2: Línea Base.

	Línea Base	
	Valor	%
CakePhp	70	87,5
CodeIgniter	65	81,25
Zend	67	83,75

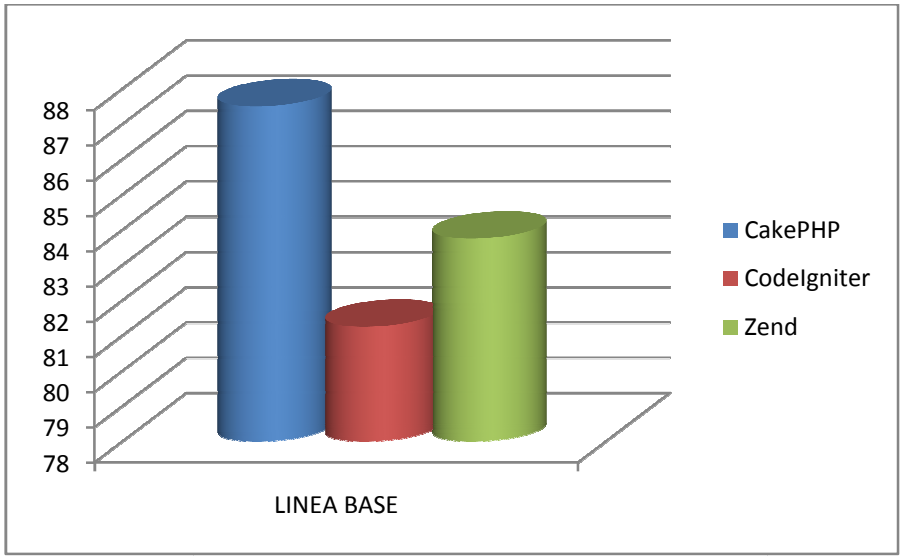


Ilustración 20 Resultado Final del Indicador 2: Línea Base

3.6.2.4. Interpretación

Tabla XXVI Representación del Indicador 2: Línea Base, con sus diferentes Índices

	LÍNEA BASE											
	CAKEPHP				CODEIGNITER				ZEND			
	Un	Cien	Mil	Diez Mil	Un	Cien	Mil	Diez Mil	Un	Cien	Mil	Diez Mil
CUxP	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣	♣	♣♣♣	♣♣♣	♣♣♣♣	♣♣	♣♣♣	♣♣♣	♣
CUxT	♣♣♣	♣♣	♣♣♣	♣♣♣	♣♣♣♣	♣♣♣	♣♣	♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣
VR	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣
MABR	♣♣♣♣	♣♣♣♣	♣♣♣	♣	♣♣	♣♣♣	♣♣	♣♣♣♣	♣♣♣	♣♣	♣♣	♣♣♣
VRTU	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣

3.6.2.5. Descripción Resultados

Los resultados expuestos en la tabla XXVI “Representación del Indicador 2: Línea Base” serán explicados por índices.

Consumo de Usuarios por prueba: CakePhp se destaca al realizar la evaluación total de consumo de recursos de red en forma global.

Consumo de Usuarios por transacción: CodeIgniter posee un mejor manejo al utilizar los recursos de manera adecuada e individual.

Velocidad de red: todos los Frameworks utilizan y administran de igual manera el ancho de banda de la red, por lo que usan el mismo estándar y por lo tanto en la administración de la velocidad de red coinciden igualitariamente todos.

Mínimo de ancho de banda requerido: no existe mucha diferencia en la utilización del menor valor de ancho de banda entre Frameworks, destacando a CakePhp al ocupar menos recursos enviando la misma información.

Velocidad de red para el total de usuarios: todos los Frameworks trabajaron con un ancho de banda configurado y todos aprovecharon al máximo la red y de manera óptima los resultados son igualitarios entre todos.

3.6.3. Indicador 3:Carga Transaccional de Alta

3.6.3.1. Determinación del Indicador

La secciónCarga Transaccional de Alta posee 2 índices cada uno de los cuales mantuvo 4 pruebas diferentes. Sus resultados obtenidos serán evaluados tomando sus valores mínimos, máximos para cada índice, así realizar una escala de valorización y determinar los valores no satisfactorio, poco satisfactorio, satisfactorio y muy satisfactorio respectivamente.

Tabla XXVII Valorización para el Indicador III: Carga Transaccional de Alta

Valor Cualitativo	Valor Cuantitativo	Valor Representativo
No Satisfactorio	1	♣
Poco Satisfactorio	2	♣♣
Satisfactorio	3	♣♣♣
Muy Satisfactorio	4	♣♣♣♣

- **Índice 3.1: Ingreso de Carga Transaccional:**En este índice se valorizará para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks).

Tabla XXVIII Valorización del Índice 3.1: Ingreso de Carga Transaccional

Resultados				Valor Cualitativo
Uno	Cien	Mil	Diez Mil	
-	2,21-2,31	-	4:02,325	No Satisfactorio
0,5-0,519	2,01-2,2	19,36-19,719	3:35,000	Poco Satisfactorio
0,46-0,49	1,91-2,0	19,1-19,35	3,09,286	Satisfactorio
0,415-0,45	1,844-1,9	18,876-19	3:08,717	Muy Satisfactorio

- **Índice 3.2: Duración Transaccional:** En este índice se valorizará para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks).

Tabla XXIX Valorización del Índice 3.2: Duración Transaccional

Resultados				Valor Cualitativo
Uno	**Cien**	**Mil**	**Diez Mil**	
24-25,14	-	43,01-46,91	-	No Satisfactorio
21-23	22,01-23,04	38,01-43	4:23,85	Poco Satisfactorio
18,1-20	21,01-22	35,01-38	4:00,00	Satisfactorio
15,64-18	19,67-21	33,42-35	3:32,69	Muy Satisfactorio

3.6.3.2. Valoraciones

Tabla XXX Resultados del Indicador 3: Carga Transaccional de Alta, después de la obtención de la ejecución de las diferentes pruebas Ver Anexo 2.

	CARGA TRANSACCIONAL DE ALTA											
	CAKEPHP				**CODEIGNITER**				**ZEND**			
	Un	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**DiezMil**
Ingreso de Carga Transaccional	0.435	2.310	19.719	4:02.325	0.415	1.844	18.876	3:08.717	0.519	2.129	19.388	3:09.286
Duración Transaccional	25.14	23.04	46.91	4:23.85	15.64	20.32	33.42	3:32.69	18.95	19.67	35.13	3:32:99

3.6.3.3. Calificación

Tabla XXXI Calificación del Indicador 3: Carga Transaccional de Alta, se basarán en la calificación de las Tablas (27-29)

	CARGA TRANSACCIONAL DE ALTA											
	CAKEPHP				CODEIGNITER				ZEND			
	Un	Cien	Mil	Diez Mil	Un	Cien	Mil	Diez Mil	Un	Cien	Mil	DiezMil
Ingreso de Carga Transaccional	4	1	2	1	4	4	4	4	2	2	2	3
Duración Transaccional	1	2	1	2	4	4	4	4	3	4	3	4

Para sacar los porcentajes de cada índice se utilizará las siguientes formulas.
Dónde:

X = Equivale a CakePhp.
Y = Equivale a CodeIgniter
Z = Equivale a Zend.

C1(X)= Puntaje acumulativo del Ingreso de Carga Transaccional CakePhp
C2(X)= Puntaje acumulativo de la Duración Transaccional CakePhp
C1(Y)= Puntaje acumulativo del Ingreso de Carga Transaccional CodeIgniter
C2(Y)= Puntaje acumulativo de la Duración Transaccional CodeIgniter
C1(Z)= Puntaje acumulativo del Ingreso de Carga Transaccional Zend
C2(Z)= Puntaje acumulativo de la Duración Transaccional Zend
Vi= Resultado de las pruebas (Un-Cien-Mil y Diez mil).

$$\boldsymbol{C1(X)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C2(X)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi}$$

$$C1(Y) = \sum_{i=1}^{n=4} vi \;\; ; \;\; C2(Y) = \sum_{i=1}^{n=4} vi$$

$$C1(Z) = \sum_{i=1}^{n=4} vi \;\; ; \;\; C2(Z) = \sum_{i=1}^{n=4} vi$$

Pa(X)= Valor acumulativo de CakePhp

Pa(Y)= Valor acumulativo de CodeIgniter

Pa(Z)= Valor acumulativo de Zend

$$Pa(X) = \sum C1(X) + C2(X)$$

$$Pa(Y) = \sum C1(Y) + C2(Y)$$

$$Pa(Z) = \sum C1(Z) + C2(Z)$$

PpT (X)= Porcentaje Parcial Total de CakePhp

PpT (Y)= Porcentaje Parcial Total de CodeIgniter

PpT (Z)= Porcentaje Parcial Total de Zend

Vp= Valor máximos de índices es de 16

$$PpT(X) = \left(\frac{Pa(X)}{Vp}\right) * 100$$

$$PpT(Y) = \left(\frac{Pa(Y)}{Vp}\right) * 100$$

$$PpT(Z) = \left(\frac{Pa(Z)}{Vp}\right) * 100$$

Tabla XXXII Valores y Porcentajes del Indicador 3: Carga Transaccional de Alta, con sus respectivos Índices

	CARGA TRANSACCIONAL DE ALTA					
	CakePhp		**CodeIgniter**		**Zend**	
	Valor	%	Valor	%	Valor	%
Ingreso de Carga Transaccional	8	50	16	100	9	56,25
Duración Transaccional	6	37,5	16	100	14	87,5

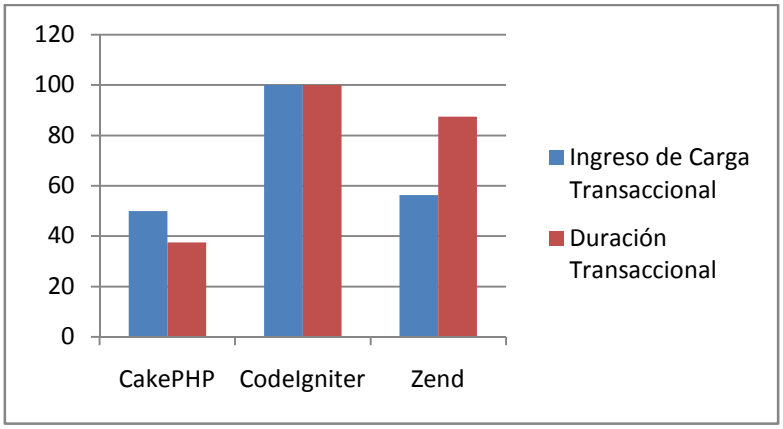


Ilustración 21Resultado por índice del Indicador 3: Carga Transaccional de Alta

Para sacar los porcentajes de cada índice se utilizará las siguientes formulas.

Dónde:

PT(X)= Porcentaje Total de CakePhp

PT(Y)= Porcentaje Total de CodeIgniter

PT(Z)= Porcentaje Total de Zend

VA= Valor máximos de índices es de 32

$$\boldsymbol{PT(X) = \left(\frac{Pa(X)}{VA}\right) * 100}$$

$$\boldsymbol{PT(Y) = \left(\frac{Pa(Y)}{VA}\right) * 100}$$

$$\boldsymbol{PT(Z) = \left(\frac{Pa(Z)}{VA}\right) * 100}$$

Tabla XXXIII Valores y Porcentajes Finales del Indicador 3: Carga Transaccional de Alta.

	CARGA TRANSACCIONAL DE ALTA	
	Valor	%
CakePhp	14	43,75
CodeIgniter	32	100
Zend	23	71,875

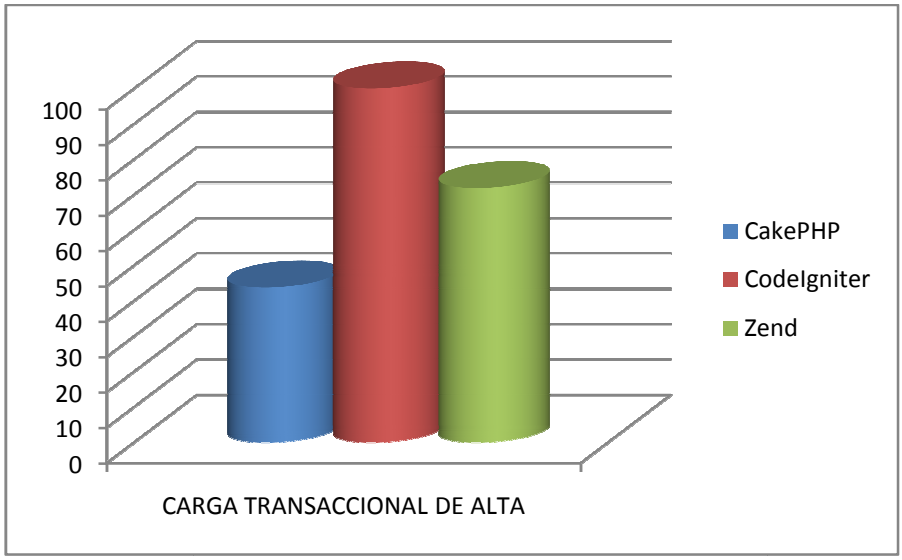


Ilustración 22 Resultado Final del Indicador 1: Carga Transaccional de Alta

3.6.3.4. Interpretación

Tabla XXXIV Representación del Indicador 3: Carga Transaccional de Alta, con sus diferentes Índices

	CARGA TRANSACCIONAL DE ALTA											
	CAKEPHP				CODEIGNITER				ZEND			
	Un	Cien	Mil	Diez Mil	Un	Cien	Mil	Diez Mil	Un	Cien	Mil	DiezMil
Ingreso de Carga Transaccional	♣♣♣♣	♣	♣♣	♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣	♣♣	♣♣	♣♣♣
Duración Transaccional	♣	♣♣	♣	♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣	♣♣♣♣	♣♣♣	♣♣♣♣

3.6.3.5. Descripción Resultados

Los resultados expuestos en la tabla XXXIV “Representación del Indicador 3: Carga Transaccional de Alta” serán explicados por índices.

Ingreso de carga transaccional: al realizar ingresos a la base de datos de 1 a 10000 datos los resultados son contundentes según datos obtenidos de la tabla anterior, CodeIgniter es el más rápido cuando se trata de interactuar con la base de datos y realizar ingresos y actualizaciones, CakePhp y Zend están por el promedio de velocidad de CodeIgniter.

Duración transaccional: Cuando se trata de optimizar tiempos globales realizando cargas transaccionales CodeIgniter siempre destaca al realizar sus tareas en periodos menores las mismas cargas que los 2 Frameworks competidores, seguido de cerca Zend que también obtuvo tiempos característicos que su principal CodeIgniter.

3.6.4. Indicador 4:Carga Transaccional de Baja

3.6.4.1. Determinación del Indicador

La secciónCarga Transaccional de Baja posee 2 índices cada uno de los cuales mantuvo 4 pruebas diferentes. Sus resultados obtenidos serán evaluados tomando sus valores mínimos, máximos para cada índice, así realizar una escala de valorización y determinar los valores no satisfactorio, poco satisfactorio, satisfactorio y muy satisfactorio respectivamente.

Tabla XXXV Valorización para el Indicador 4: Carga Transaccional de Baja

Valor Cualitativo	Valor Cuantitativo	Valor Representativo
No Satisfactorio	1	♣
Poco Satisfactorio	2	♣♣
Satisfactorio	3	♣♣♣
Muy Satisfactorio	4	♣♣♣♣

- **Índice 4.1: Baja de Carga Transaccional:**En este índice se valorizará para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks).

Tabla XXXVI Valorización del Índice 4.1: Baja de Carga Transaccional

Resultados				**Valor Cualitativo**
Uno	**Cien**	**Mil**	**Diez Mil**	
-	-	22-23.42	3:47.201	No Satisfactorio
0.31-0.388	2.52-2.684	21-22	2:30.01	Poco Satisfactorio
0.21-0.3	2.21-2.5	20-21	3:14.834	Satisfactorio
0.177-0.2	1.959-2.2	19.122-19.99	3:09481	Muy Satisfactorio

- **Índice 4.2: Duración Transaccional:** En este índice se valorizará para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks).

Tabla XXXVII Valorización del Índice 4.2: Duración Transaccional

Resultados				**Valor Cualitativo**
Uno	**Cien**	**Mil**	**Diez Mil**	
25.01-29.37	30.01-36.17	1:32.99	12:15.98	No Satisfactorio
20.01-25	26.01-30	1;15.00	9:00.00	Poco Satisfactorio
19.01-20	24.01-26	1:08.68	6:56.49	Satisfactorio
18.51-19	23.28-24	1:00.10	6:47.70	Muy Satisfactorio

3.6.4.2. Valoraciones

Tabla XXXVIII Resultados del Indicador 4: Carga Transaccional de Baja, después de la obtención de la ejecución de las diferentes pruebas Ver Anexo 2.

	CARGA TRANSACCIONAL DE BAJA											
	CAKEPHP				CODEIGNITER				ZEND			
	Un	Cien	Mil	Diez Mil	Un	Cien	Mil	Diez Mil	Un	Cien	Mil	DiezMil
Baja de Carga Transaccional	0.383	2.234	23.420	3:47.201	0.175	1.959	19.122	3:09.481	0.377	2.684	20.036	3:14.834
Duración Transaccional	29.37	36.17	1:32.99	12.15.98	19.24	23.28	1:00.10	6:56.49	18.51	24.34	1:08.68	6:47.70

3.6.4.3. Calificación

Tabla XXXIX Calificación del Indicador 4: Carga Transaccional de Baja, se basarán en la calificación de las Tablas (35-37)

	CARGA TRANSACCIONAL DE BAJA											
	CAKEPHP				CODEIGNITER				ZEND			
	Un	Cien	Mil	Diez Mil	Un	Cien	Mil	Diez Mil	Un	Cien	Mil	DiezMil
Baja de Carga Transaccional	2	3	1	1	4	4	4	4	2	2	3	3
Duración Transaccional	1	1	1	1	3	4	4	4	4	3	3	3

Para sacar los porcentajes de cada índice se utilizará las siguientes formulas.
Dónde:

X = Equivale a CakePhp.
Y = Equivale a CodeIgniter
Z = Equivale a Zend.

C1(X)= Puntaje acumulativo Baja de Carga Transaccional CakePhp
C2(X)= Puntaje acumulativo de la Duración Transaccional CakePhp
C1(Y)= Puntaje acumulativo Baja de Carga Transaccional CodeIgniter
C2(Y)= Puntaje acumulativo de la Duración Transaccional CodeIgniter
C1(Z)= Puntaje acumulativo Baja de Carga Transaccional Zend
C2(Z)= Puntaje acumulativo de la Duración Transaccional Zend

Vi= Resultado de las pruebas (Un-Cien-Mil y Diez mil).

$$\boldsymbol{C1(X) = \sum_{i=1}^{n=4} vi \;\; ; \;\; C2(X) = \sum_{i=1}^{n=4} vi}$$

$$\boldsymbol{C1(Y) = \sum_{i=1}^{n=4} vi \;\; ; \;\; C2(Y) = \sum_{i=1}^{n=4} vi}$$

$$\boldsymbol{C1(Z) = \sum_{i=1}^{n=4} vi \;\; ; \;\; C2(Z) = \sum_{i=1}^{n=4} vi}$$

Pa(X)= Valor acumulativo de CakePhp
Pa(Y)= Valor acumulativo de CodeIgniter
Pa(Z)= Valor acumulativo de Zend

$$\boldsymbol{Pa(X) = \sum C1(X) + C2(X)}$$

$$\boldsymbol{Pa(Y) = \sum C1(Y) + C2(Y)}$$

$$\boldsymbol{Pa(Z) = \sum C1(Z) + C2(Z)}$$

PpT (X)= Porcentaje Parcial Total de CakePhp

PpT (Y)= Porcentaje Parcial Total de CodeIgniter

PpT (Z)= Porcentaje Parcial Total de Zend

Vp= Valor máximos de índices es de 16

$$PpT(X) = \left(\frac{Pa(X)}{Vp}\right) * 100$$

$$PpT(Y) = \left(\frac{Pa(Y)}{Vp}\right) * 100$$

$$PpT(Z) = \left(\frac{Pa(Z)}{Vp}\right) * 100$$

Tabla XL Valores y Porcentajes del Indicador 4: Carga Transaccional de Baja, con sus respectivos Índices

	CARGA TRANSACCIONAL DE BAJA					
	CakePhp		**CodeIgniter**		**Zend**	
	Valor	%	Valor	%	Valor	%
Baja de Carga Transaccional	7	43,75	16	100	10	62,5
Duración Transaccional	4	25	15	93,75	13	81,25

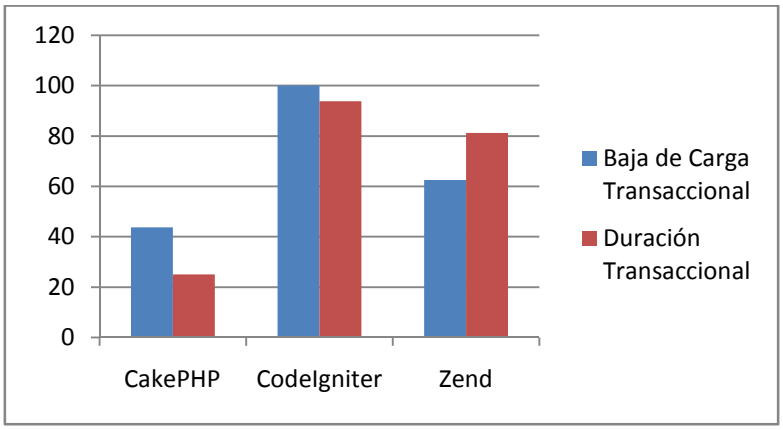


Ilustración 23Resultado por índice del Indicador 4: Carga Transaccional de Baja

Para sacar los porcentajes de cada índice se utilizará las siguientes formulas.

Dónde:

PT(X)= Porcentaje Total de CakePhp

PT(Y)= Porcentaje Total de CodeIgniter

PT(Z)= Porcentaje Total de Zend

VA= Valor máximos de índices es de 32

Tabla XLI Valores y Porcentajes Finales del Indicador 4: Carga Transaccional de Baja.

	CARGA TRANSACCIONADE BAJA	
	Valor	%
CakePhp	11	34,375
CodeIgniter	31	96,875
Zend	23	71,875

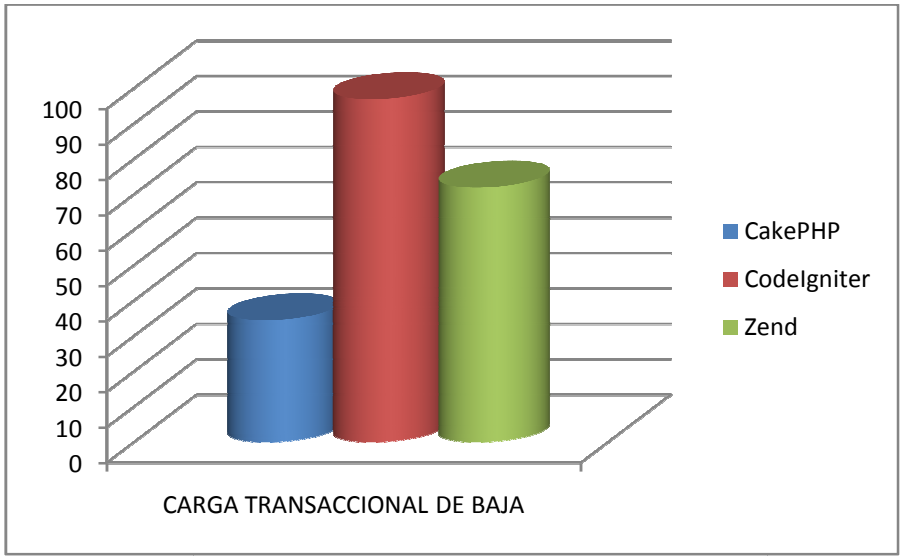


Ilustración 24Resultado Final del Indicador 4: Carga Transaccional de Baja

3.6.4.4. Interpretación

Tabla XLII Representación del Indicador 4: Carga Transaccional de Baja, con sus diferentes Índices

	CARGA TRANSACCIONAL DE ALTA											
	CAKEPHP				CODEIGNITER				ZEND			
	Un	Cien	Mil	Diez Mil	Un	Cien	Mil	Diez Mil	Un	Cien	Mil	DiezMil
Ingreso de Carga Transaccional	♣♣	♣♣♣	♣	♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣	♣♣	♣♣♣	♣♣♣
Duración Transaccional	♣	♣	♣	♣	♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣	♣♣♣	♣♣♣

3.6.4.5. Descripción de Resultados

Los resultados expuestos en la tabla XLII “Representación del Indicador 4: Carga Transaccional de Baja” serán explicados por índices.

Baja de carga transaccional: al momento de eliminar datos y presentarlos en una tabla dinámica CodeIgniter es el más rápido ya que su Frameworks es más liviano y posee código optimizado en cuanto a realizar estas tareas conocidas, Zend Frameworks también posee tiempos que se consideran buenos para tomar en cuenta en la realización de proyectos empresariales.

Duración transaccional: El tiempo es vital al momento de realizar miles de transacciones por miles de usuarios que realizan una misma acción, en este caso CodeIgniter en el manejo de procesos de Baja y alta se destaca de sus competidores, dando como resultados de la misma acción tiempos cortos, Zend una vez más ha

demostrado que es un buen competidor al terminar sus procesos con tiempos relativamente cortos a su mejor competidor CodeIgniter.

3.6.5. Indicador 5:Integridad

3.6.5.1. Determinación del Indicador

La secciónde Integridad posee 4 índices cada uno de los cuales mantuvo 4 pruebas diferentes. Sus resultados obtenidos serán evaluados tomando sus valores mínimos, máximos para cada índice, así realizar una escala de valorización y determinar los valores no satisfactorio, poco satisfactorio, satisfactorio y muy satisfactorio respectivamente.

Tabla XLIII Valorización para el Indicador 5: Integridad

Valor Cualitativo	Valor Cuantitativo	Valor Representativo
No Satisfactorio	1	♣
Poco Satisfactorio	2	♣♣
Satisfactorio	3	♣♣♣
Muy Satisfactorio	4	♣♣♣♣

- **Índice 5.1: User:** En este índice tendrá valorizaciones estándar para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks) por haber configurado previamente.

Tabla XLIV Valorización del Índice 5.1: Usuarios

Resultados	Valor Cualitativo
-	No Satisfactorio
-	Poco Satisfactorio
-	Satisfactorio
10	Muy Satisfactorio

- **Índice 5.2: Éxito:** En este índice se valorizará para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks).

Tabla XLV Valorización del Índice 5.2: Éxito

Resultados				**Valor Cualitativo**
Uno	**Cien**	**Mil**	**Diez Mil**	
-	-	-	-	No Satisfactorio
43-44	38-19	-	-	Poco Satisfactorio
45-48	40-41	18-19	-	Satisfactorio
49-53	42-44	20-21	10	Muy Satisfactorio

- **Índice 5.3: Fallas:** En este índice se valorizará para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks).

Tabla XLVI Valorización del Índice 5.3: Fallas

Resultados				**Valor Cualitativo**
Uno	**Cien**	**Mil**	**Diez Mil**	
-	-	-	-	No Satisfactorio
-	-	-	-	Poco Satisfactorio
-	-	-	-	Satisfactorio
0	0	0	0	Muy Satisfactorio

- **Índice 5.4: Promedio transacción del paquete:** En este índice se valorizará para cada una de las pruebas estás son (Uno, Cien, Mil y diez mil Ingresos por cada Frameworks).

Tabla XLVII Valorización del Índice 5.4: Promedio Transacción del Paquete

Resultados				**Valor Cualitativo**
Uno	**Cien**	**Mil**	**Diez Mil**	
-	-	238.1	2.4 MB	No Satisfactorio

4.4	24.5-27.4	205.5	2.1 MB	Poco Satisfactorio
4.2-4.3	20.1-24.4	170	1.7 MB	Satisfactorio
3.9-4.1	17.4-20	137,5	1.4 MB	Muy Satisfactorio

3.6.5.2. Valoraciones

Tabla XLVIII Resultados del Indicador 5: Integridad, después de la obtención de la ejecución de las diferentes pruebas Ver Anexo 2

	INTEGRIDAD											
	CAKEPHP				**CODEIGNITER**				**ZEND**			
	Un	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**Diez Mil**
Usuario	10	10	10	10	10	10	10	10	10	10	10	10
Éxito	44	38	18	10	53	41	21	10	43	44	19	10
Fallas	0	0	0	0	0	0	0	0	0	0	0	0
Promedio transacción	4.4	24.3	205.5	2.1 MB	3.9	17.4	137.5	1.4 MB	4.2	27.4	238.1	2.4 MB

3.6.5.3. Calificación

Tabla XLIX Calificación del Indicador 5: Integridad, se basarán en la calificación de las Tablas (43-47)

	INTEGRIDAD											
	CAKEPHP				**CODEIGNITER**				**ZEND**			
	Un	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**Diez Mil**	**Un**	**Cien**	**Mil**	**Diez Mil**
Usuario	4	4	4	4	4	4	4	4	4	4	4	4
Éxito	2	2	3	4	4	3	4	4	2	4	3	4
Fallas	4	4	4	4	4	4	4	4	4	4	4	4

Promedio transacción	2	3	3	3	4	4	4	4	3	2	1	1

Para sacar los porcentajes de cada índice se utilizará las siguientes formulas.
Dónde:

X = Equivale a CakePhp.
Y = Equivale a CodeIgniter
Z = Equivale a Zend.

C1(X)= Puntaje acumulativo del Usuario CakePhp
C2(X)= Puntaje acumulativo de Éxito de CakePhp
C3(X)= Puntaje acumulativo Fallas de CakePhp
C4(X)= Puntaje acumulativo Promedio de Transacción del Paquete de CakePhp

C1(Y)= Puntaje acumulativo del Usuario CodeIgniter
C2(Y)= Puntaje acumulativo de Éxito de CodeIgniter
C3(Y)= Puntaje acumulativo Fallas de CodeIgniter
C4(Y)= Puntaje acumulativo Promedio de Transacción del Paquete de CodeIgniter

C1(Z)= Puntaje acumulativo del Usuario Zend
C2(Z)= Puntaje acumulativo de Éxito de Zend
C3(Z)= Puntaje acumulativo Fallas de Zend
C4(Z)= Puntaje acumulativo Promedio de Transacción del Paquete de Zend

Vi= Resultado de las pruebas (Un-Cien-Mil y Diez mil).

$$\boldsymbol{C1(X)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C2(X)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C3(X)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C4(X)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi}$$

$$\boldsymbol{C1(Y)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C2(Y)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C3(Y)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi} \;\; ; \;\; \boldsymbol{C4(Y)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n=4}} \boldsymbol{vi}$$

$$C1(Z) = \sum_{i=1}^{n=4} vi \;\; ; \;\; C2(Z) = \sum_{i=1}^{n=4} vi \;\; ; \;\; C3(Z) = \sum_{i=1}^{n=4} vi \;\; ; \;\; C4(Z) = \sum_{i=1}^{n=4} vi$$

Pa(X)= Valor acumulativo de CakePhp

Pa(Y)= Valor acumulativo de CodeIgniter

Pa(Z)= Valor acumulativo de Zend

$$Pa(X) = \sum C1(X) + C2(X) + C3(X) + C4(X)$$
$$Pa(Y) = \sum C1(Y) + C2(Y) + C3(Y) + C4(Y)$$
$$Pa(Z) = \sum C1(Z) + C2(Z) + C3(Z) + C4(Z)$$

PpT (X)= Porcentaje Parcial Total de CakePhp

PpT (Y)= Porcentaje Parcial Total de CodeIgniter

PpT (Z)= Porcentaje Parcial Total de Zend

Vp= Valor máximos de índices es de 16

$$PpT(X) = \left(\frac{Pa(X)}{Vp}\right) * 100$$
$$PpT(Y) = \left(\frac{Pa(Y)}{Vp}\right) * 100$$
$$PpT(Z) = \left(\frac{Pa(Z)}{Vp}\right) * 100$$

Tabla L Valores y Porcentajes del Indicador 5: Integridad, con sus respectivos Índices

	INTEGRIDAD					
	CakePhp		**CodeIgniter**		**Zend**	
	Valor	%	Valor	%	Valor	%
Usuario	16	100	16	100	16	100
Éxito	11	68,75	15	93,75	13	81,25
Fallas	16	100	16	100	16	100
Promedio transacción del paquete	11	68,75	16	100	7	43,75

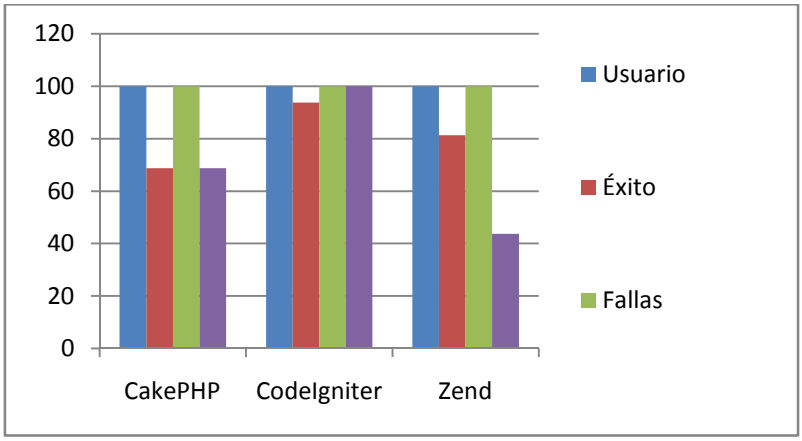


Ilustración 25Resultado por índice del Indicador 5: Integridad

Para sacar los porcentajes de cada índice se utilizará las siguientes formulas.

Dónde:

PT(X)= Porcentaje Total de CakePhp

PT(Y)= Porcentaje Total de CodeIgniter

PT(Z)= Porcentaje Total de Zend

VA= Valor máximos de índices es de 64

$$\boldsymbol{PT(X) = \left(\frac{Pa(X)}{VA}\right) * 100}$$

$$\boldsymbol{PT(Y) = \left(\frac{Pa(Y)}{VA}\right) * 100}$$

$$\boldsymbol{PT(Z) = \left(\frac{Pa(Z)}{VA}\right) * 100}$$

Tabla LI Valores y Porcentajes Finales del Indicador 5: Integridad

	INTEGRIDAD	
	Valor	%
CakePhp	54	84,375
CodeIgniter	63	98,457
Zend	52	81,25

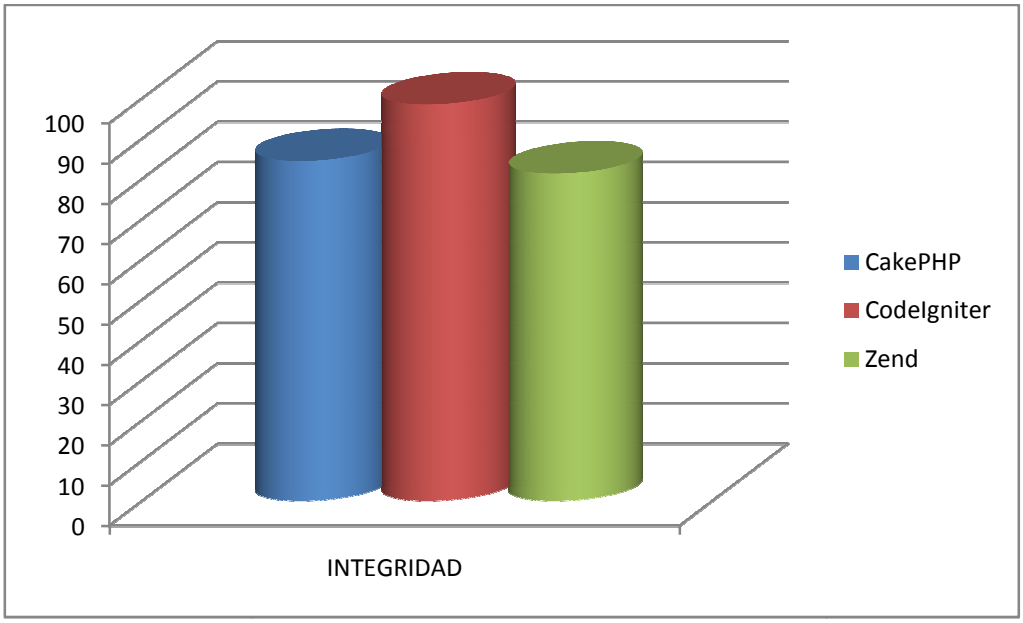


Ilustración 26 Resultado Final del Indicador 5: Integridad

3.6.5.4. Interpretación

Tabla LII Representación del Indicador 5: Integridad, con sus diferentes Índices

	INTEGRIDAD											
	CAKEPHP				CODEIGNITER				ZEND			
	Un	Cien	Mil	Diez Mil	Un	Cien	Mil	Diez Mil	Un	Cien	Mil	Diez Mil
Usuario	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣
Éxito	♣♣	♣♣	♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣	♣♣♣♣	♣♣♣♣	♣♣	♣♣♣♣	♣♣♣	♣♣♣♣
Fallas	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣
Promedio transacción del paquete	♣♣	♣♣♣	♣♣♣	♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣♣	♣♣♣	♣♣	♣	♣

3.6.5.5. Descripción de Resultados

Los resultados expuestos en la tabla LII “Representación del Indicador 5: Integridad” serán explicados por índices.

Usuario: en los escenarios planteados y realizada la simulacion con el web perfomace tester la cantidad de usuarios fue la misma para todos los framework y todos los usuarios realizron las misma tarea, por lo tanto todos los framework en este punto reciben el mismo puntaje igualitario al utilizar todos los usuarios disponibles.

Éxito: El triunfo de una transaccion no necesariamente es del mas rapido, tambien debe cumplir con llevar la carga, datos completa y sin errores, una vez mas Codeigniter se destanca al realizar las transaciones completamentey sin fallas, de igual manera Zend y CakePhp siguen los mismos principios.

Fallas: el manejo de errores al momento de realizar las transacciones es un punto fuerte de los 3 Framework ya que si falla alguna transacion lo vuelven a enviar y notificar, Codeigniter obtiene el mejor puntaje al mencionados eventos.

Promedio transacion del paquete: Dentro del area de redes, la carga util es la informacion que se transporta de un servidor a otro, obviando los controles e informacion de seguridad y capas que se realizan en el transporte de la informacion, los tres framework poseen un buen desempeño ya que todos trabajan sobre Tcp/Ip y manejan los mismos protocolos, Codeigniter tiene un mejor transporte de carga util, seguido de cerca cakephp y zend cuando se trata de cantidades de información alta en su cabecera de transporte lleva un porcentaje mayor de controles, esto no necesarimente es malo cuando se trata de informacion sumamente importante, los controles ayudan a que la informacion no sea capturada por otros usarios de la red.

3.7. Puntajes Alcanzados

Tabla LIII Cuadro Resultados de Indicador e Índice

INDICADORES	INDICES	FRAMEWORKS			
		Promedio	CakePhp	CodeIgniter	Zend
INGENIERIA DE CARGA	CPU %	16	7	13	9
	Memoria %	16	10	14	13
	Ancho de Banda Entrada	16	10	16	10
	Ancho de Banda de Salida	16	6	10	16
LINEA BASE	Consumo de Usuarios x Prueba	16	15	11	9
	Consumo de Usuarios x Transacción	16	11	11	16
	Velocidad de la Red	16	16	16	16
	Mínimo Ancho de Banda Requerido	16	12	11	10
	Velocidad de red para el total de usuarios	16	16	16	16
CARGA TRANSACCIONAL DE ALTA	Ingreso de Carga Transaccional	16	8	16	9
	Duración Transaccional	16	6	16	14
CARGA TRANSACCIONAL DE BAJA	Baja de Carga Transaccional	16	7	16	10
	Duración Transaccional	16	4	15	13
INTEGRIDAD	Usuario	16	16	16	16
	Éxito	16	11	15	13
	Fallas	16	16	16	16
	Promedio transacción del paquete	16	11	16	7
TOTAL		272	182	244	213

3.8. Interpretación

Las tabulaciones de los resultados obtenidos en las pruebas dan a conocer el comportamiento global de cada Frameworks en los diferentes escenarios expuestos en el desarrollo de este análisis comparativo.

La ingeniería de carga fue la encargada de probar la utilización de los recursos de hardware al momento de realizar todas las operaciones de los escenarios planteados anteriormente, donde se destaca el FrameworksCakePhp ya que su principal función es un desarrollo rápido de aplicaciones web sólidas.

La línea base fue el indicador encomendado a dar un punto base de partida de cada Frameworks es decir las características de consumo de recursos de las transacciones unitarias y globales, donde se destaca CodeIgniter seguido de cerca de CakePhp, ya que ambos poseen un Frameworks de peso extra ligero desarrollado en Modelo Vista controlador, cuentan con clases optimizadas y buen manejo interno de los recursos de hardware y software.

Las transacciones realizadas de alta y baja en los 3 Frameworks seleccionados denotan que CodeIgniter es el mejor en cuanto al uso de recursos y manejo de la base de datos en conjunto con la aplicación web, ya que los tiempos son menores en comparación conCakePhp y ZendFramework.

La integridad de los datos en las tres escenarios se mantuvo en un excelente nivel probando que los 3 Frameworks poseen solidez al momento de realizar todo tipo de transacciones, cabe destacar que CodeIgniter en los resultados fue el mejor ya que posee como una de sus características el manejo de errores.

Los resultados son claros que cada Frameworks fue desarrollado para casos diferentes ya que sus componentes y características están enfocados a ciertos tipos de aplicaciones, CodeIgniter y CakePhpestán altamente recomendado para el desarrollo de aplicaciones

de pequeña y mediana empresa en cambio Zend Frameworks para aplicaciones empresariales grandes ya que posee una cantidad extensa de clases y servicios más utilizados además de poseer clases desarrollados API para integrarse con aplicaciones web de terceros.

3.9. Análisis de Resultados

Luego de la interpretación de los resultados realizado anteriormente destaca el Frameworks CodeIgniter como el ideal para realizar el desarrollo del portal web de la academia Linux con los objetivos planteados al principio de la tesis, ya que CodeIgniter tiene una completa gama de clases para conexión de base de datos con soporte para varias plataformas. Formularios y validación de datos, manejo de sesiones y perfiles además posee la clase Benchmarking que ayuda a optimizar y comprobar el desempeña de nuestra aplicación web.

En conclusión por tener un dominio amplio sobre los demás Frameworks y contando con la experiencia como usuarios desarrolladores de haber utilizado los 3 Frameworks para el desarrollo de las aplicaciones web, concluimos que el Framework CodeIgniter es el más adecuado para el desarrollo de la aplicación web.

3.10. Comprobación de Hipótesis

La hipótesis planteada es:

H1: CodeIgniter es el Frameworks que ofrece el mejor rendimiento frente a Zend y CakePhp

Tabla LIVParámetros de Hipótesis

VARIABLE	INDICADOR	INDICE	TÉCNICA	HERRAMIENTA
VARIABLE DEPENDIENTE FRAMEWORKS	FRAMEWORKS	CAKEPHP CODEIGNITER ZEND	Encuesta, Cuestionario Observación Directa	Web, Facebook Tool, Web Performance load tester
	APLICACION	APLICACIÓN		
VARIABLE DEPENDIENTE RENDIMIENTO	INGENIERIA DE CARGA	CPU %	Monitoreo Observación de Laboratorio Test Experimento	Web Performance load tester. Monitoreo de Recursos. %CPU.
		Memoria %	Monitoreo Observación de Laboratorio Test Experimento	Web Performance load tester. Monitoreo de Recursos. %Memoria.
		Ancho de Banda Entrada	Monitoreo Observación de Laboratorio Test Experimento	Web Performance load tester. Monitoreo de Recursos. Actividad de red.
		Ancho de Banda de Salida	Monitoreo Observación de Laboratorio Test Experimento	Web Performance load tester. Monitoreo de Recursos. Actividad de red...

	LINEA BASE	Consumo de Usuarios x Prueba	Monitoreo Observación de Laboratorio Test Experimento	Web Performance load tester. Simulación.
		Consumo de Usuarios x Transacción	Monitoreo Observación de Laboratorio Test Experimento	Web Performance load tester. Simulación
		Velocidad de la Red	Monitoreo Observación de Laboratorio Test Experimento	Web Performance load tester. Simulación
		Mínimo Ancho de Banda Requerido	Monitoreo Observación de Laboratorio Test Experimento	Web Performance load tester. Simulación
		Velocidad de red para el total de usuarios	Monitoreo Observación de Laboratorio Test Experimento	Web Performance load tester. Simulación
	CARGA TRANSACCION	Ingreso de Carga	Monitoreo Observación	Web Performance load tester.

	AL DE ALTA	Transaccional	de Laboratorio Test Experimento	Simulación
		Duración Transaccional	Monitoreo Observación de Laboratorio Test Experimento	Web Performance load tester. Simulación
	CARGA TRANSACCIONAL DE BAJA	Baja de Carga Transaccional	Monitoreo Observación de Laboratorio Test Experimento	Web Performance load tester. Simulación
		Duración Transaccional	Monitoreo Observación de Laboratorio Test Experimento	Web Performance load tester. Simulación
	INTEGRIDAD	Usuario	Monitoreo	Web Performance load tester. Simulación
		Éxito	Monitoreo	Web Performance load tester.
		Fallas	Monitoreo	Web Performance load tester.
		Promedio	Test	Web Performance

		transacción del paquete		load tester.

Tabla LVResultado Finales

INDICADORES	INDICES	FRAMEWORKS			
		Promedio	CakePhp	CodeIgniter	Zend
INGENIERIA DE CARGA	CPU %	16	7	13	9
	Memoria %	16	10	14	13
	Ancho de Banda Entrada	16	10	16	10
	Ancho de Banda de Salida	16	6	10	16
LINEA BASE	Consumo de Usuarios x Prueba	16	15	11	9
	Consumo de Usuarios x Transacción	16	11	11	16
	Velocidad de la Red	16	16	16	16
	Mínimo Ancho de Banda Requerido	16	12	11	10
	Velocidad de red para el total de usuarios	16	16	16	16
CARGA TRANSACCIONAL DE ALTA	Ingreso de Carga Transaccional	16	8	16	9
	Duración Transaccional	16	6	16	14
CARGA TRANSACCIONAL DE BAJA	Baja de Carga Transaccional	16	7	16	10
	Duración Transaccional	16	4	15	13
INTEGRIDAD	Usuario	16	16	16	16
	Éxito	16	11	15	13
	Fallas	16	16	16	16
	Promedio transacción del paquete	16	11	16	7

TOTAL	272	182	244	213

Tabla LVI Valores y Porcentajes Finales

	TOTAL	
	Valor	Porcentaje
CakePhp	182	66,91
CodeIgniter	244	89,7
Zend	213	78,3

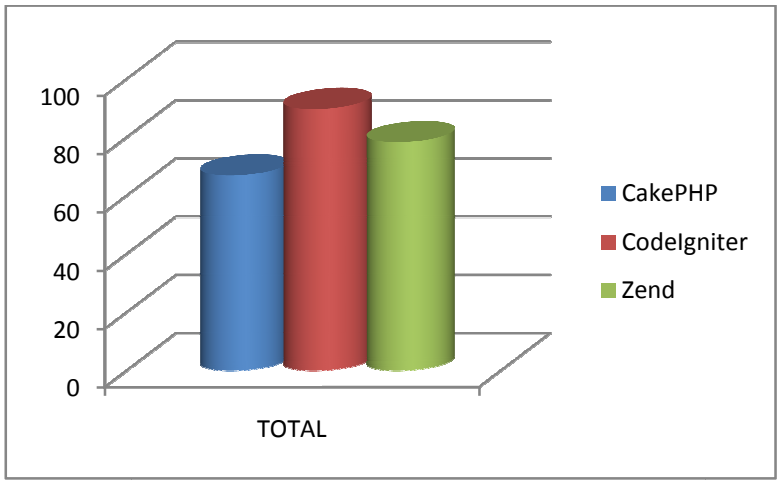


Ilustración 27 Resultado Final del Análisis

Haciendo referencia a la tabla “LVI” y por observación directa se concluye que en la hipótesis planteada: CodeIgniter posee el valor más alto 244 puntos sobre 272, valor máximo posible, superando en un 22,79% a la herramienta Cake PHP y en un 11.4% a la herramienta Zend, CodeIgniter tiene el nivel de desempeño del 89,7%/100. Por lo tanto se concluye que la hipótesis H1 es verdadera.

4. CAPÍTULO IV:

DESARROLLO DEL PORTAL

Al momento del desarrollo del sistema web para la academia Linux, se va a proceder a rediseñar completamente el sistema web, siguiendo los objetivos y alcances planteados en nuestra justificación práctica.

La aplicación contempla el rediseño de la apariencia web con el uso de menús amigables, gestión de usuarios como área de registro y de login, para compartir la información se tendrá la sección de soporte y descargas para mantener actualizados a los usuarios de las últimos driver y noticias sobre la Academia y sus cursos.

El sistemaarchivará todas las noticias y artículos publicados que se podrán revisar instantáneamente similar a una hemeroteca, la sección de registro de usuarios se crea para mantener niveles de acceso a la información muy importante para tener un control que garantice la privacidad y seguridad del sistema.

4.1. Ingeniería de la Información

La metodología usada para el desarrollo del portal web se la puede considerar como un conjunto de pasos y procedimientos que deben seguirse para el desarrollo del software.

Se ha utilizado una metodología híbrida, desarrollada para procesos de diseño de aplicaciones Web, denominada “Metodología para Creación de Sitios Web”, creada por la Ingeniera Dely Maybel Gil Álvarez del Instituto Universitariode Tecnología de Valencia (IUTVal).

Está basada principalmente en las fases parael desarrollo de sitios Web propuestas en la obra “Studio MX Creación desitios Web”, la creación de sitios Web está dividida en cuatro partes:Administración del sitio Web, Desarrollo del concepto del sitio, desarrollo delcontenido, Prueba y entrega del producto final.

La Metodología comprende las fases de:Análisis, Planificación, Contenido, Diseño, Programación, Testeo, Mercadeoy Publicidad.

4.1.1. Definición del Ámbito

Crear una solución informática que provea la automatización de los procesos que realiza el portal web de la Academia LINUX - ESPOCH, para facilitar sus servicios académicos.

Además se pretende generar la creciente interacción entre el usuario y el portal para que pueda ser más práctico con la finalidad de llevar un mejor control de los accesos a la documentación.

La solución que se pretende realizar se ve enfocada en el mejoramiento de los procesos que realiza el portal web de la Academia LINUX - ESPOCH, para mejor su calidad de

servicio y adaptarse a la tecnología actual, ya que la mayor parte de loma por falta de pasos intuitivos.

El modelo de desarrollo de software para el portal de la academia Linux se aplica el RAD (Desarrollo Rápido de Aplicaciones).

Tipos de Usuarios y Permisos de Acceso: Los Usuarios de sitios web linuxespoch.com pueden dividirse en dos categorías principales:

- Invitados
- Usuarios Registrados

Los Invitados son sencillamente usuarios delinuxespoch.com que han navegado hasta encontrar su sitio web. Dependiendo de cómo el administrador ha configurado el sitio, los invitados podrán navegar libremente por todo el contenido o tener restringido el acceso a cierto tipo de contenidos, reservados para usuarios registrados.

Los Usuarios Registrados están registrados en su sitio con un nombre de usuario y contraseña. Este nombre de usuario y contraseña les permite acceder al área restringida del sitio, recibiendo privilegios especiales no disponibles para los invitados. Los usuarios registrados se dividen en dos grupos:

Usuarios del Sitio (Front-end)

Registrado	Un Usuario Registrado no puede crear, editar o publicar contenido en el sitio. Puede enviar nuevos Enlaces Web para ser publicados y puede tener acceso a contenidos restringidos que no están disponibles para los invitados.
Autor	Los Autores pueden crear su propio contenido, especificar ciertos aspectos de cómo se presentará el contenido e indicar la fecha en la que debería publicarse el material.

Editor	Los Editores tienen todas las posibilidades de un Autor, y además la capacidad de editar el contenido de sus propios artículos y los de cualquier otro Autor.
Supervisor	Los Supervisores pueden ejecutar todas las tareas de los Autores y Editores, y además tienen la capacidad de publicar un artículo.

Usuarios del Administrador (Back-end)

Mánager	Un Manager tiene los mismos permisos que un Supervisor pero con acceso al panel de administración del Back-end. Los Managers tienen acceso, en el panel del administrador, a todos los controles asociados al contenido, pero no tienen capacidad para cambiar las plantillas, alterar el diseño de las páginas, o añadir o eliminar extensiones de Linuxespoch.com Los Managers tampoco tienen autoridad para añadir usuarios o alterar los perfiles de usuarios existentes.
Administrador	Los Administradores tienen un rango de acceso más amplio que los Managers. Además de todas las actividades relacionadas con el contenido que puede ejecutar un Mánager, los administradores pueden añadir o eliminar extensiones al sitio web, cambiar plantillas o alterar el diseño de las páginas, e incluso alterar los perfiles de usuario a un nivel igual o inferior al suyo. Lo que no pueden hacer los Administradores es editar los perfiles de Súper- Administradores o cambiar ciertas características globales del sitio web. De hecho, ni siquiera verán los usuarios de tipo ‘Super-Administrador’ en el Administrador de Usuarios.
Súper-	Los Súper-Administradores tienen el mismo poder que un

Administrador	‘root’ en un sistema tradicional *Linux* y disponen de posibilidades ilimitadas para ejecutar todas las funciones administrativas del Portal. Solo los Súper-Administradores tienen la capacidad de crear nuevos usuarios con permisos de Súper-Administrador, o asignar este permiso a usuarios ya existentes.

Módulos del portal web: Tendrá 3 módulos principales

Módulo de Certificaciones: contendrá toda la información referente a la Academia y acceso a ella. Se tendrá un menú con los siguientes ítems:Catalogo de Cursos, Requisitos, metodología de estudio, soporte y descargas, preguntas frecuentes.

Módulo de Foro: en esta sección se tendrá un foro abierto para realizar comentarios y poder crear temas para discutir entre la comunidad Linux.

Módulo de Usuario: la parte de control de acceso al usuario, podrá registrarse para poder acceder a la información clasificada, mediante un formulario de registro y si se encuentra registrado, ingresara normalmente con su Id y Password.

4.1.2. Requerimientos

4.1.2.1. Requerimientos Funcionales

Para solucionar los inconvenientes citados en el literal anterior se puede desarrollar un sistema el mismo que permitirá.

1. Ingresar, Consultar y Actualizar la información de los usuarios registrados
2. Ingresar, Consultar y Actualizar la información de los administradores del portal
3. Ingresar, Consultar y Actualizar la información de todo el portal dinámicamente
4. Ingresar, Consultar y Actualizar la información de las noticias publicadas.

4.1.2.2. Requerimientos No Funcionales

A continuación se muestran los requerimientos no funcionales más relevantes del software con sus respectivas características:

- **Rendimiento**

- Tiempos de respuesta al, abrir una página web para la autenticación será de 1s.
- Tasa esperada de velocidad de respuesta dada la tasa de clientes conectados será mucho más rápida.

- **Fiabilidad**

- Capacidad para tolerar errores en un 90%.
- Capacidad para tolerar sobrecargas en el volumen de información, de usuarios o de procesos en un 80%.

- **Disponibilidad**

- El sistema estará funcionando 24*7 o sea 24 h por 7 días.
- Empleo de sistemas de respaldo.

- **Seguridad**

- El sistema tendrá un formulario de autenticación.
- Uso de sesiones de usuario.

- **Portabilidad**

- Utilización de estándares para la presentación de información.

- **Mantenibilidad**

- Empleo del Modelo Microsoft Solution Framework para la metodología.
- Documentación del diseño y de la codificación de la solución.

- **Escalabilidad**

- Diseño de la arquitectura empleando MVC

- Empleo de NetBeans, y Frameworks para hacer al sistema compatible con otros sistemas.

- **Reusabilidad**

- Uso de estándares en los formatos para los datos.

- **Interfaces**

- Interfaces web para carga de archivos de entrada/salida.
- Intuitivas y amigables la interfaz

- **Amigabilidad (Usabilidad)**

- Facilidad de uso.
- Diseño de interfaz de usuario.

4.1.3. Estudio de Factibilidad

4.1.3.1. Factibilidad Técnica

- **Recurso Humano**

Existe el recurso humano capacitado para realizar la administración del sistema, obteniendo una operación garantizada y uso garantizado.

- Ing. Rogel Miguez (Administrador)

- **Recurso Hardware**

✓ Intel Core 2 Duo 2.1Ghz
✓ DVD-RW
✓ Teclado, Mouse, Monitor

- **Recurso Software**

✓ *Sistema Operativo:* Microsoft Windows Seven Ultimate / Linux Fedora
✓ *Base de Datos:* MySql 3.3.9
✓ *Servidor Web:* Apache
✓ *Tecnología de Desarrollo:* NetBeans 6.9.1 y CodeIgniter Frameworks

4.1.3.2. Factibilidad Operativa

- Ing. Rogel Miguez
- Usuarios de la Academia

4.1.3.3. Factibilidad Legal

Existe la autorización de las autoridades respectivas por lo que no existe ningún tipo de impedimento legal para el desarrollo del sistema

4.1.4. Planificación Temporal

Ver Anexo 3

4.2. Análisis del Sistema

4.2.1. Definir Casos de Uso esenciales en formato extendido

CASO DE USO AUTENTIFICACIÓN

Tabla LVII Caso de Uso Autenticación

IDENTIFICAR CASO DE USO:	CU_Autentificación
NOMBRE DEL CASO DE USO	Autentificación de Usuarios
ACTORES	Estudiantes _ Asesores (Usuarios)
PROPÓSITO	Realizar el proceso de identificación de los

<table>
<tr><td></td><td colspan="2">estudiantes que utilizan el portal web</td></tr>
<tr><td>VISIÓN GENERAL</td><td colspan="2">El usuario requiere autenticarse en el sistema; ingresa su nombre de sesión y Password respectivo y de acuerdo a la función que desempeñe el usuario se abrirá la pantalla correspondiente.</td></tr>
<tr><td>TIPO</td><td colspan="2">Primario, real y expandido</td></tr>
<tr><td>REFERENCIAS</td><td colspan="2">Requerimientos</td></tr>
<tr><td colspan="3">CURSO TÍPICO DE EVENTOS</td></tr>
<tr><td>ACCIÓN DE LOS ACTORES</td><td colspan="2">RESPUESTA DEL SISTEMA</td></tr>
<tr><td>1. Este caso de uso inicia cuando el usuario desea autenticarse para realizar la función que le compete</td><td colspan="2">2. Muestra pantalla para que el usuario ingrese los datos respectivos</td></tr>
<tr><td>3. El usuario ingresa su sesión y Password</td><td colspan="2">4. Valida los datos ingresados</td></tr>
<tr><td></td><td colspan="2">5. Presenta una ventana con un menú en el cual el usuario podrá realizar las funciones asignadas.</td></tr>
<tr><td colspan="3">CURSOS ALTERNATIVOS</td></tr>
<tr><td colspan="3">2.1.- El sistema no presenta pantalla para ingreso de datos</td></tr>
<tr><td colspan="3">Mensaje de error
4.1.- “Sesión Incorrecta”
4.2.- No existe el usuario con esa sesión
Mensaje de advertencia
4.3.- “Ingrese su sesión”
4.4.- “Ingresar Clave”
4.5.-“Clave Incorrecta”</td></tr>
</table>

CASO DE USO CREACION CUENTA

Tabla LVIII Caso de Uso Creación de Cuenta

IDENTIFICAR CASO DE USO:	CU_Solicitar_Cuenta
NOMBRE DEL CASO DE USO:	Solicitar_Cuenta
ACTORES:	Administrador, Usuario
PROPÓSITO:	Realizar el registro de una determinada cuenta al estudiante que lo solicita
VISIÓN GENERAL:	El proceso inicia cuando el estudiante solicita el registro de una determinada cuenta, para lo cual el administrador solicita se le entregue los datos correspondientes para proceder a registrar la cuenta.
TIPO:	Primario, real y expandido
REFERENCIAS:	Requerimientos
CURSO TÍPICO DE EVENTOS	
ACCIÓN DEL ACTOR	**RESPUESTAS DEL SISTEMA**
1.- Se inicia cuando el estudiante se acerca a Administrador a solicitar la matriculación de un curso.	
2.- Administrador solicita documentos para poder realizar la respectiva creación de cuenta.	
3.- Estudiante entrega documentos y entrega el Numero de Cedula.	4.-Muestra pantalla para ingresar el número de cedula del usuario
5.- Administrador ingresa número de cedula	6.- Valida número de cedula
	7.- Muestra pantalla con datos del Cliente verificado.
8.- Administrador Ingresa los datos	9.- Verifica los datos

	11.- Crea Cuenta
12.- Administrador escoge la Cuenta solicitada.	
13.- Administrador Registra Cuenta	15.- Almacena en la base de datos del sistema
	16.- Presenta un mensaje que informa que se ha realizado el registro correctamente.
CURSOS ALTERNATIVOS	
Mensajes de error : 6.1 “El número de cuenta no tiene la cantidad especificada” 6.2 “Ingrese correctamente el número de cedula”	
Mensajes de Advertencia: 10.1 “No se puede realizar la creación de dicha cuenta”	
Mensaje de Información: 16.1 “La creación de la cuenta se realizo correctamente”	

CASO DE USO CAMBIAR DATOS USUARIO

Tabla LIX Caso de Uso Cambiar Datos de Usuario

IDENTIFICAR CASO DE USO:	CU_Cambiar_Datos_Usuario
NOMBRE DEL CASO DE USO:	Cambiar _ Datos_Usuario
ACTORES:	Usuario, Administrador
PROPÓSITO:	Cambiar la información de algún usuario especifico
VISIÓN GENERAL:	Los usuarios podrán realiza el cambio de información excepto de sus datos principales
TIPO:	Primario, real y expandido
REFERENCIAS:	Requerimientos
CURSO TÍPICO DE EVENTOS	

ACCIÓN DEL ACTOR	RESPUESTAS DEL SISTEMA
1.- Se inicia cuando usuario se autentifica.	2.-Muestra pantalla para autentificación de usuario
3.-Ingresa sus datos (sesión y clave)	4.- Valida datos del usuario
	5- Muestra pantalla principal correspondiente a los permisos de dicho usuario
6.- Selecciona la opción Cambio de datos del usuario	
7.- Cambia datos	
8.-Registra cambio utilizando un botón de la interfaz asignado.	9.- Actualiza los cambios en la base de datos del sistema.
	10. Muestra mensaje confirmando que la actualización se ha realizado.
CURSOS ALTERNATIVOS	
Mensajes de error : 4.1 “No existe usuario” 4.2 “Ingrese correctamente los datos”.	
Mensaje de Información: 13.1 “Los datos fueron actualizados”	

CASOS DE USO UTILIZACION SISTEMA

Tabla LX Caso de Uso Utilización Sistema

IDENTIFICAR CASO DE USO:	CU_Utilizacion_Sistema
NOMBRE DEL CASO DE USO:	Utilización Sistema
ACTORES:	Usuario
PROPÓSITO:	Realizar las diferentes operaciones que puede realizar el usuario
VISIÓN GENERAL:	El proceso inicia cuando el usuario ingresa

	al portal y podrá ver noticias, comentar y descargar todo lo que exista en el portal.
TIPO:	Primario, real y expandido
REFERENCIAS:	Requerimentos
CURSO TÍPICO DE EVENTOS	
ACCIÓN DEL ACTOR	**RESPUESTAS DEL SISTEMA**
1.- Se inicia cuando usuario se autentifica.	2.-Muestra pantalla para autentificación de usuario
3.-Ingresa sus datos (sesión y clave)	4.- Valida datos del usuario
	5- Muestra pantalla principal correspondiente a los permisos de dicho usuario
6.- Comienza a navegar	
7.- Descarga archivos	8.-Muestra pantalla de descarga.
9.- Comenta en foros	9.- Guardan los cambios en la base de datos del sistema.
	10. Muestra mensaje confirmando que la actualización se ha realizado.
CURSOS ALTERNATIVOS	
Mensajes de error : 6.1 “No existe cliente” 6.2 “Ingrese correctamente el número de cuenta”	
Mensajes de Advertencia: 10.1 “No se puede realizar la descarga, no disponible”	
Mensaje de Información: 16.1 “El descarga se ha realizado correctamente”	

4.2.2. Definir los Casos de Uso

Esta actividad nos permite representar los casos de uso a partir de la nomenclatura definida en UML, tanto para la representación de actores, casos de uso, interacciones y relaciones.

DIAGRAMA DE CASO DE USO AUTENTIFICACIÓN

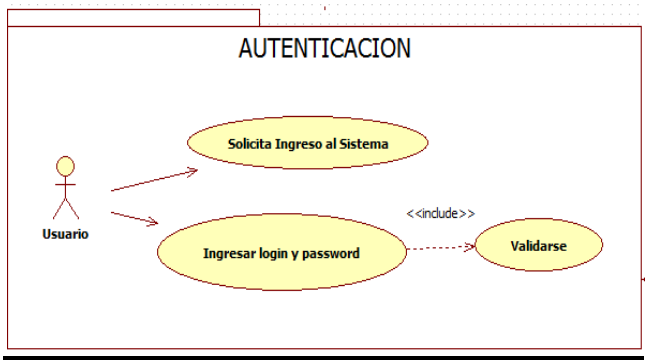


Ilustración 28 Diagrama de Caso de Uso Autenticación

DIAGRAMA DE CASOS DE USO CREACION DE CUENTA

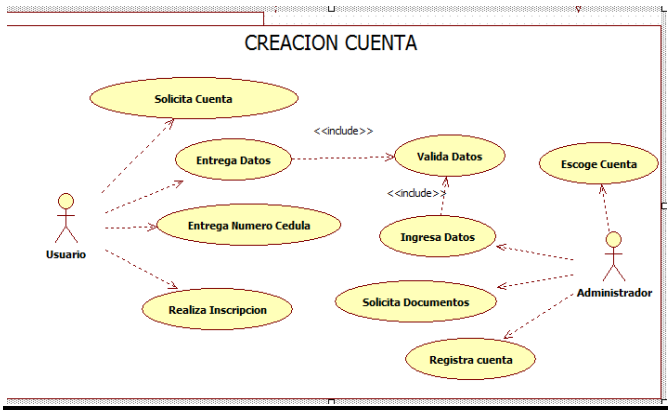


Ilustración 29 Diagrama de Caso de Uso Creación de Cuenta

DIAGRAMA DE CASO DE USO CAMBIAR DATOS USUARIO

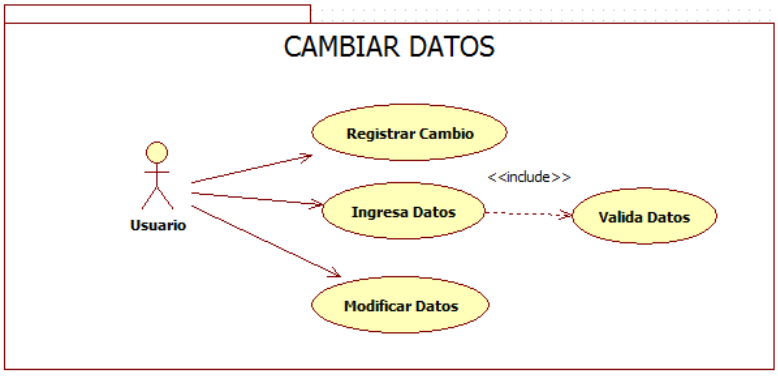


Ilustración 30 Diagrama de Caso de Cambiar Datos de Usuario

DIAGRAMA DE CASO DE USO UTILIZACION SISTEMA

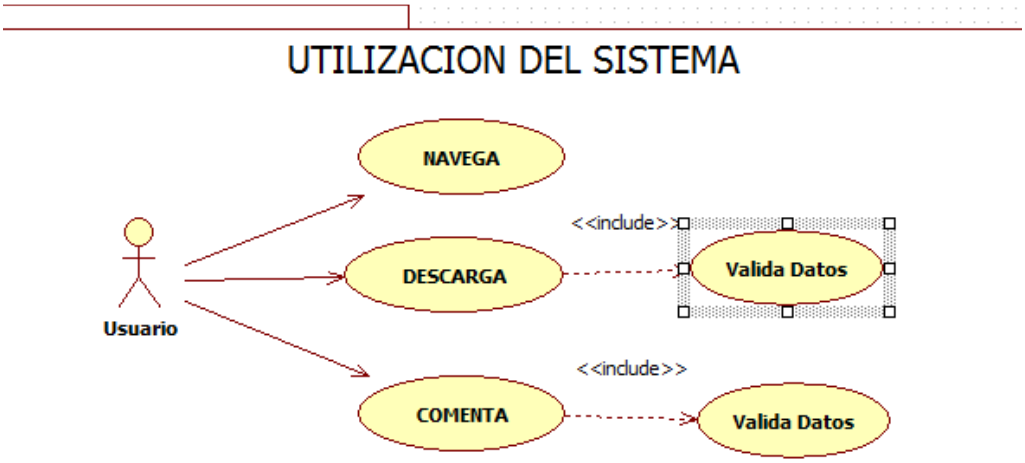


Ilustración 31 Diagrama de Caso de Uso Utilización Sistema

4.2.3. Diagrama de Secuencias

DIAGRAMA DE SECUENCIAS AUTENTIFICACIÓN

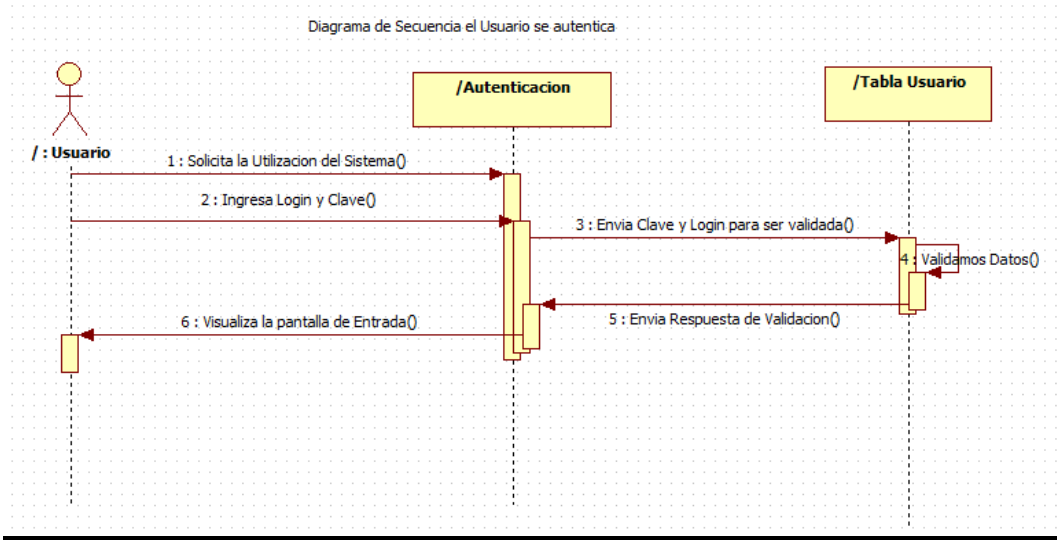


Ilustración 32 Diagrama de Secuencias Autenticación

DIAGRAMA DE SECUENCIAS CREACION DE CUENTA

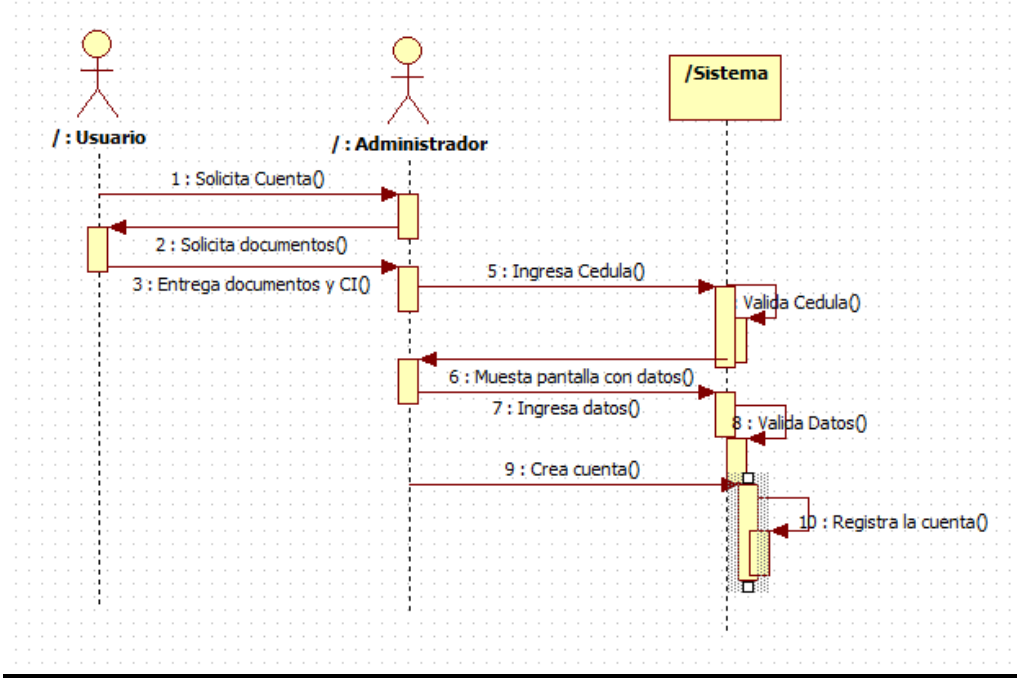


Ilustración 33 Diagrama de Secuencias Creación de Cuenta

DIAGRAMA DE SECUENCIAS CAMBIAR DATOS USUARIO

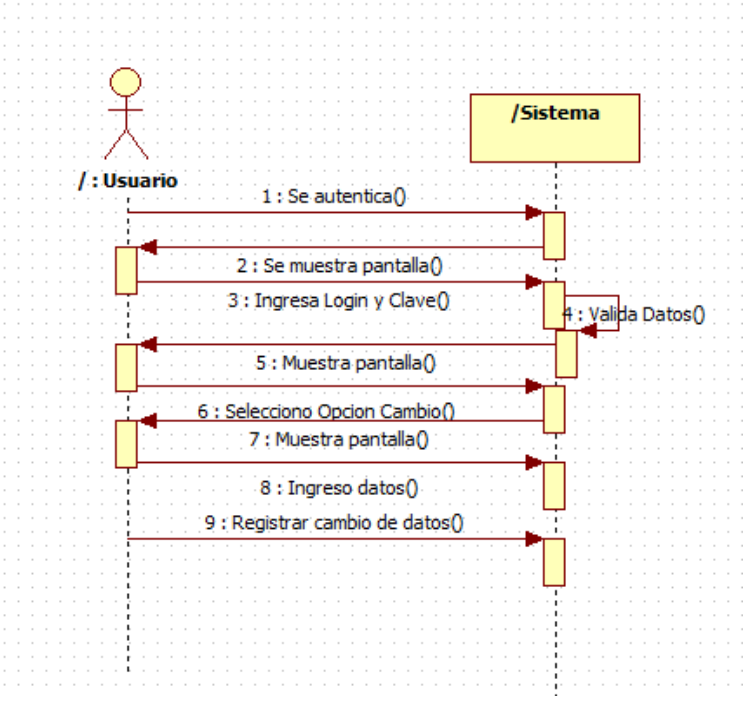


Ilustración 34 Diagrama de Secuencias Cambiar Datos Usuario

DIAGRAMA DE SECUENCIAS UTILIZACION SISTEMA

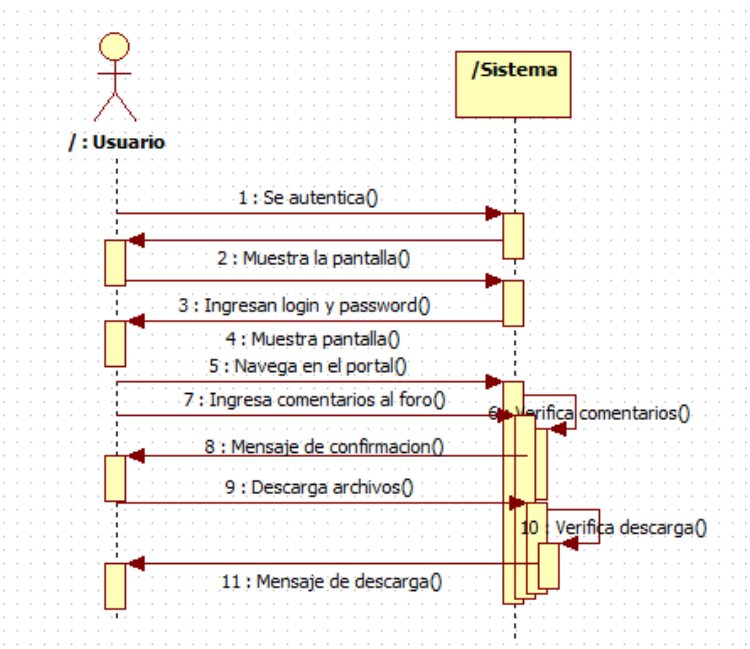


Ilustración 35 Diagrama de Secuencias Utilización Sistema

4.3. Diseño

4.3.1. Diagrama de Clases

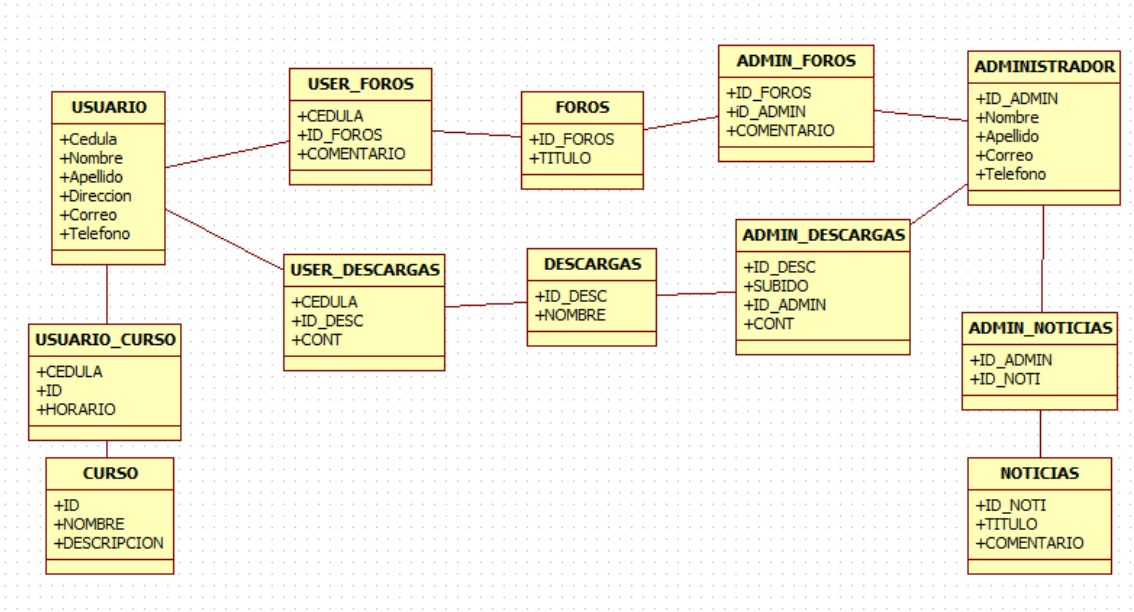


Ilustración 36 Diagrama de Clases

4.3.2. Diagrama de Componentes

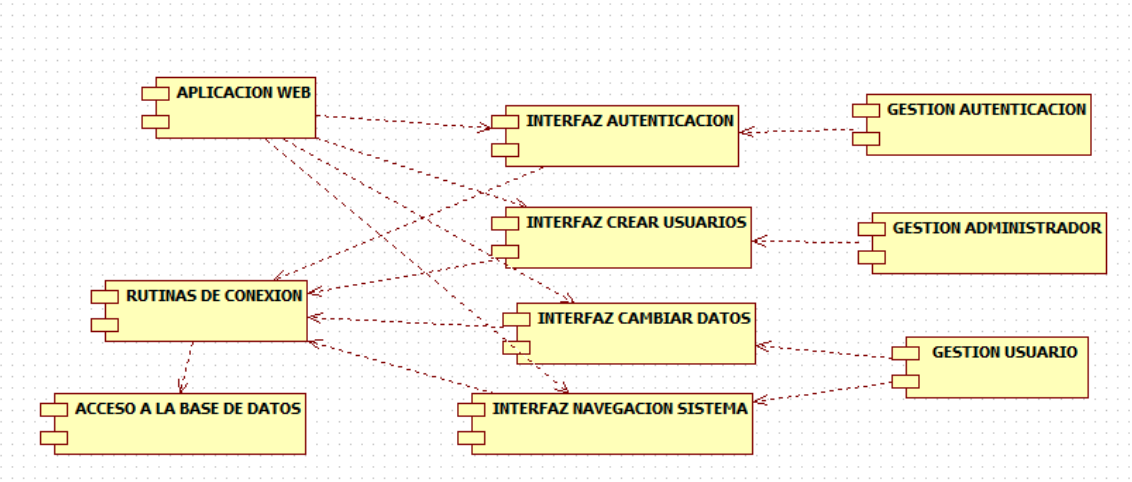


Ilustración 37 Diagrama de Componentes

4.3.3. Diagrama de Nodos

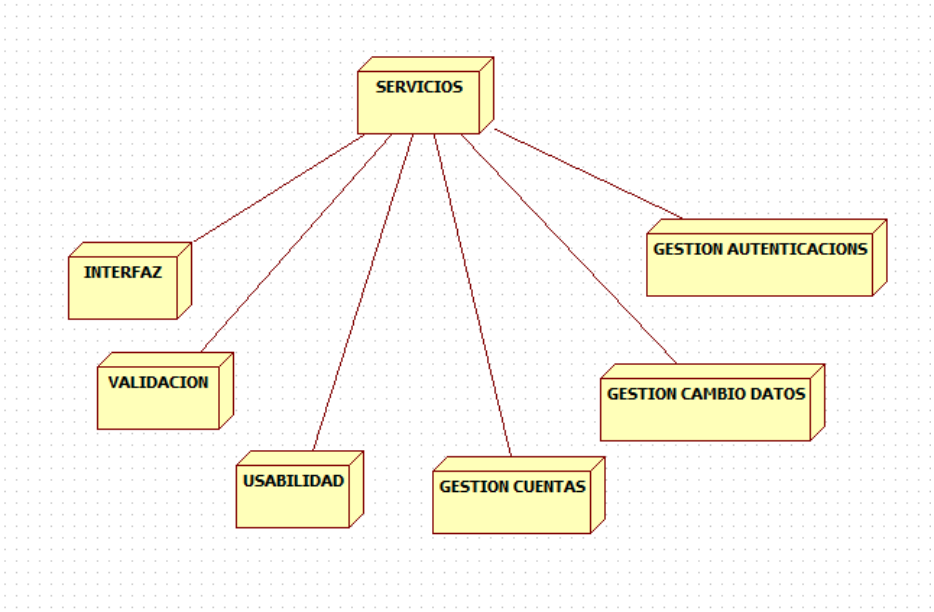


Ilustración 38 Diagrama de Nodos

4.4. Implementación y Pruebas

4.4.1. Definición de estándares de Programación

Para realizar la codificación del sistema se han definido los siguientes estándares:

- El nombre de las tablas se encuentra con letras minúsculas.
- El nombre de los campos empieza con una letra minúscula, seguido de una línea hacia abajo en el caso de más de una palabra.
- Los métodos que interactúan con la base de datos reciben como parámetros de entrada:
    - Connection: Representa la cadena de conexión hacia la base de datos.
    - Lista de parámetros en el caso de ser necesaria.

- Los métodos que interactúan con la base de datos en el caso de funciones de inserción, deben devolver como resultado el código del campo que ha sido insertado.

4.4.2. Pruebas Unitarias

Para asegurar el correcto funcionamiento del Sistema se han probado sus métodos de forma independiente, enviando datos de entrada desde el código, para luego obtener los a través de los diferentes métodos para realizar consultas.Se han probado especialmente todas las funciones para validaciones de datos.

4.4.3. Pruebas de Módulos y del Sistema

Las pruebas finales consistieron en verificar que la información ingresada se vea inmediatamente reflejada en las consultas del sistema, esto sirve para comprobar que la información se está registrando correctamente en la base de datos.

Se provocaron errores intencionales para verificar el correcto funcionamiento del sistema, así como de las funciones de validación de datos, como por ejemplo:
- Realizar consultas a tablas vacías
- Ingresar campos incorrectos

Tratar de ingresar información diferente al tipo de dato correcto, como tratar de ingresar texto en los campos que son numéricos, ingresar formatos de fechas diferentes, tratar de ingresar datos técnicos en servicios que no admiten esta información.

CONCLUSIONES

1. Mediante un adecuado estudio de los tres Frameworks seleccionados, se puede tener una mejor visión en el marco del rendimiento.

2. El estudio nos permitió determinar que el Framework más eficiente para el desarrollo en el rendimiento es CodeIgniter.

3. En el Análisis de Ingeniería de Carga, CodeIgniter posee un 82.81% de desempeño en la utilización de los recursos de hardware, superando con un 7.81% a la herramienta Zend que posee junto con CodeIgniter la mejor manera de optimizar los recursos de hardware, y superando con un 31.25% a Cakephp, demostrando que CodeIgniter como una de las mejores características es la optimización y utilización mínima de recursos de hardware como es CPU y memoria.

4. El indicador Línea Base es el punto promedio para determinar el nivel de partida y funcionamiento óptimo de la aplicación, Cake al utilizar el mínimo de recursos y por ser un Frameworks liviano supero en un 6.25% a CodeIgniter y en un 3.75 a Zend, se ha demostrado que los 3 Frameworks poseen características similares para obtener los requisitos mínimos de funcionamiento.

5. El manejo de cargas transaccionales de alta demanda de un aplicación robusta que maneje miles de transacciones en tiempos cortos, CodeIgniter supera en un 28.2% a Zend y en un 56.25 % a Cake en el manejo de transacciones en el menor tiempo posible, ya que posee librerías optimizadas que permiten transportar más datos que controles.

6. El manejo de la base de datos mediante el uso de Frameworks requiere de utilización de recursos de hardware y software al momento de realizar consultas a bases de datos, demostrando que Codeigniter posee las mejores librerias para

el manejo de bases de datos, superando ampliamente a Cake con un 62%, seguido a Zend con un 25% en el manejo de las transacciones.

7. No basta con ejecutar y procesar la información en el menor tiempo posible, sino también dicha información debe llegar a su destino sin errores e intacta, CodeIgniter demuestra que tiene control de errores al superar a Zend en un 14% y en un 17% a CakePhp, todos los Frameworks poseen control de errores, asegurando así que la información llegue integra a su destino.

8. La utilización de Frameworks para el desarrollo de aplicaciones nos facilita a la estandarización y la programación efectiva del desarrollo.

9. La investigación realizada sobre el rendimiento de los Frameworks facilitará a los desarrolladores a la optimización de recursos en la implementación de sus aplicaciones.

10. El manejo de recursos de hardware y software es complejo por lo tanto al momento de elegir el Frameworks para el desarrollo de la aplicación web se debe realizar un previo análisis obligatorio de diferentes Frameworks para elegir cual se adapta a las necesidades de la aplicación para obtener el mejor sistema web.

RECOMENDACIONES

1. Plantear correctamente con el usuario final todos los requisitos que desea para no perder tiempo en el desarrollo de la aplicación.

2. Revisar infraestructuras y sistemas informáticos existentes antes de decidir la utilización de una determinada tecnología, ya que esto nos ayudará a tener más claro el panorama.

3. Realizar un análisis previo de características de diferentes Frameworks para seleccionar el que mejor se preste para desarrollar los requisitos de software.

4. Investigar las librerías y funciones que cada Frameworks posee para el desarrollo del sistema web, ya que aquellos ayudarán al desarrollo rápido de aplicaciones en tiempos cortos.

5. Se debería implementar como requisito de software un análisis previo de Frameworks para realizar el sistema web, como una recomendación de mejores prácticas para el desarrollo de aplicaciones.

RESUMEN

Se realizó el análisis del rendimiento de los Frameworks Php para aplicaciones web óptimas. Caso Práctico el portal Academia Linux de la Escuela Superior Politécnica de Chimborazo.

El análisis comparativo se basó en el método empírico, con técnicas de observación directa, de laboratorio y experimento, para alcanzar los objetivos se utilizó las siguientes herramientas: Servidor Web Apache 2.2.17, MySQL Server 5.5.8), Php 5.3.5 (VC6 X86 32bit) lenguaje de programación web y NetBeans 6.9 como IDE de desarrollo.

Los Frameworks motivo del análisis investigativo fueron integrados al IDE NetBeans 6.9 para el desarrollo de sistemas de pruebas que permitieron determinar cuál posee el mejor rendimiento, el resultado cuantitativo obtenido mediante la comparación de Indicadores: Ingeniería de carga, Línea Base, Cargas transaccionales de alta, Cargas transaccionales de baja, Integridad, para evaluar el rendimiento en el desarrollo fue:66.91%(regular) para CakePhp; 89.70%(muy bueno) para Codeigniter; 78.3%(bueno) para Zend.

En conclusión la estructura de software que ofrece el mejor rendimiento para el desarrollo de aplicaciones web óptimas con un puntaje de 89.70% es CodeIgniter que supera en desempeño a sus competidores, ya que esta es una herramienta muy ligera y ayudará a las grandes empresas a responder el gran número de peticiones o solicitudes.

Como recomendaciónya que existen varias infraestructuras realizar un previo estudio de características de diferentes Frameworks para seleccionar el que mejor se preste para desarrollar los requisitos de software.

SUMMARY

The performance analysis of the Php Frameworks for optimum web applications, was carried out. Such is the case of the Academy Linux gate of the Chimborazo Higher Education Polytechnic School. The comparative analysis was basedon the empirical method, with direct observation, laboratory and experimentation techniques.
To reach the objectives the following tolls were used: Server Web Apache 2.2.17, MySql Server 5.5.8, Php 5.3.5(VC6 X86 32 bit) web programming language and Netbeans as a development IDE.

The investigation analysis frameworks were integrated to the IDE Netbeans 6.9 for the test system development which permitted to determine the best performance; the quantitative result obtained through indicator comparison were: Load engineering, base line, high transactional loads, low transactional loads and integrity to evaluate the development performance which was: 66.91(regular) for the Cakephp; 89.70%(very good) for the Codeigniter and 78.3% (good) for Zend.

As a conclusion, the software structure which offers the best performance for the development of optimum web applications with a score of 89.70% is the Codeigniter which surpasses it’s competitors in performance, as it is a very light tool and will help the large enterprises to meet a number of requests or applications.

It is recommended, as there are various infrastructures, to carry out a previous study of the characteristics of the different frameworks to select the best one to develop the software requirements.

BIBLIOGRAFÍA

(1.) Admixweb, Frameworks PHP.
http://www.admixweb.com/2010/10/06/15-best-ever-php-frameworks/
[Consulta 2011 01 03]

(2.) ALEGSA, Rendimiento
http://www.alegsa.com.ar/Dic/rendimiento.php
[Consulta 2011 06 02]

(3.) ALEGSA, Optimizacion de Recurso
http://www.alegsa.com.ar/Dic/optimizar.php
[Consulta 2011 06 02]

(4.) ALEGSA, Eficiencia del Framework
http://www.alegsa.com.ar/Dic/eficiencia.php
[Consulta 2011 01 03]

(5.) METODOLOGIAS DE SISTEMAS, Arquitecturas
http://metodologiasdesistemas.blogspot.com/2007/10/que-es-un-orm-object-relational-mapping.html
[Consulta 2011 04 10]

(6.) CakePHP, Historia, Caracteristicas del Framework
http://book.cakephp.org/es/view/8/What-is-CakePHP-Why-Use-it

[Consulta 2011 01 04]

(7.) Codeigniter, Historia, Caracteristicas del Framework
http://codeigniter.com/user_guide/overview/cheatsheets.html
[Consulta 2011 01 04]

(8.) El Web Master, Tops de los Frameworks Recomendados
http://www.elwebmaster.com/articulos/frameworks-php-recomendados-guia-para-principiantes
[Consulta 2011 01 03]

(9.) Enciclopedia Encydia,Caracteristicas de Zend Frameworks
http://es.encydia.com/pt/Zend_Framework
[Consulta 2011 01 06]

(10.) FernandoAsiap, Frameworks PHP más usados
http://www.asiap.org/jiap/presentaciones/bull.pps
[Consulta 2011 01 04]

(11.) GNU, Copileft de GNU
http://www.gnu.org/copyleft/gpl.html
[Consulta 2011 01 04]

(12.) GutiérrezLSI, Frameworks PHP
http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf
[Consulta 2011 01 04]

(13.) Maestros del Web, Que son las Base de Datos
http://www.maestrosdelweb.com/principiantes/%C2%BFque-son-las-bases-de-datos/

[Consulta 2011 04 10]

(14.) Mundo Geek, El mundo de los Frameworks
http://mundogeek.net/archivos/2010/10/30/frameworks-php/
[Consulta 2011 01 04]

(15.) Open Source, Licenciamiento GNU
http://www.opensource.org/licenses/mit-license.php
[Consulta 2011 01 04]

(16.) PHP Frameworks, Top 10 de Frameworks PHP<
http://www.phpframeworks.com/top-10-php-frameworks/
[Consulta 2011 01 04]

(17.) Prefacio.Rendimiento
http://www.php.net/manual/es/preface.php
[Consulta 2011 06 04]

(18.) Wikipedia, Base da Datos Orientadoa a Objetod
http://es.wikipedia.org/wiki/Base_de_datos_orientada_a_objetos
[Consulta 2011 0530]

(19.) Wikipedia. Cach
http://es.wikipedia.org/wiki/Cach%C3%A9
[Consulta 2011 06 02]

(20.) Wikipedia. Scaffold
http://en.wikipedia.org/wiki/Scaffold_%28programming%29
[Consulta 2011 06 02]

(21.) Wikipedia. modelo Vista Controlador
http://es.wikipedia.org/wiki/Modelo_Vista_Controlador
[Consulta 2011 01 04]

(22.) Wikipedia, Plantillas
http://es.wikipedia.org/wiki/Plantilla
[Consulta 2011 01 04]

(23.) Wikipedia, CakePHP
http://en.wikipedia.org/wiki/CakePHP
[Consulta 2011 01 04]

(24.) Wikipedia, Frameworks
http://es.wikipedia.org/wiki/Framework
[Consulta 2011 01 04]

(25.) Wikipedia.Zend framework
http://es.wikipedia.org/wiki/Framework
[Consulta 2011 01 04]

(26.) Zend Framework., Framework Zend Caracteristicas
http://framework.zend.com/manual/en/introduction.overview.html
[Consulta 2011 01 04]

ANEXOS

ANEXO 1

Todas la líneas de códigosque fueronProgramadas en los diferentes Frameworks estos son: CakePhp, CodeIgniter, Zend Framework.

1) CakePhp

- **Configuración de la Base de Datos en "Config-schema-database.php"**

```
class DATABASE_CONFIG {
        var $default = array(
                'driver' => 'mysql',
                'persistent' => false,
                'host' => 'localhost',
                'login' => 'root',
                'password' => '',
                'database' => 'ci_series',
                'prefix' => '',
        );
        var $test = array(
                'driver' => 'mysql',
                'persistent' => false,
                'host' => 'localhost',
                'login' => 'user',
                'password' => 'password',
                'database' => 'test_database_name',
                'prefix' => '',
```

```
        );
}
```

- **Configuración en el Contenedor “Models-libro.php”**

```
<?php
class Libro extends AppModel {
var $name = 'Libro';
}
?>
```

- **Configuración en el Contenedor “Controllers-libros_controller.php”**

```
<?php
class LibrosController extends AppController {
var $name = 'Libros';
var $helpers = array('html', 'Form');
function index() {
$this->set('libros', $this->Libro->find('all'));
}
function add() {
if (!empty($this->data)) {
$this->Libro->create();
if ($this->Libro->save($this->data)){
$this->Session->setFlash('El Libro ha sido salvada');
$this->redirect(array('action'=>'index'),null,true);
} else {
   $this->Session->SetFlash('Tarea no salvada. Prueba de nuevo');
}
}
}
```

```
function add10() {
if (!empty($this->data)) {
   for ($i = 0; $i < 10; $i++):
    $this->Libro->create();
if($this->Libro->save($this->data)){}
endfor;
$this->redirect(array('action'=>'index'),null,true);
}
}
function add100() {
if (!empty($this->data)) {
   for ($i = 0; $i < 100; $i++):
    $this->Libro->create();
if($this->Libro->save($this->data)){}
endfor;
$this->redirect(array('action'=>'index'),null,true);
}
}
function add1000() {
if (!empty($this->data)) {
   for ($i = 0; $i < 1000; $i++):
    $this->Libro->create();
if($this->Libro->save($this->data)){}
endfor;
$this->redirect(array('action'=>'index'),null,true);
}
}
function add10000() {
if (!empty($this->data)) {
   for ($i = 0; $i < 10000; $i++):
    $this->Libro->create();
if($this->Libro->save($this->data)){}
```

```
endfor;
$this->redirect(array('action'=>'index'),null,true);
}
}
function edit($id = null ){
if (!$id){
   $this->Session->setFlash('Tarea Inválida');
   $this->redirect(array('action'=>'index'),null,true);
}
if(empty($this->data)){
   $this->data = $this->Libro->find(array('id'=> $id));
}
else{
if ($this->Libro->save($this->data)){
   $this->Session->setFlash('Tarea ha sido salvada');
   $this->redirect(array('action'=>'index'),null,true);
} else {
$this->Session->setFlash('La Tarea no ha sido salvada. Inténtalo de nuevo');
}
}
}

function delete($id=null){
if (!$id){
   $this->Session->setFlash('Tarea Inválida');
   $this->redirect(array('action'=>'index'),null,true);
}
if(empty($this->data)){
   $this->data = $this->Libro->find(array('id'=> $id));
}
else{
if ($this->Libro->delete($id)){
```

```
    $this->redirect(array('action'=>'index'),null,true);
} else {
$this->Session->setFlash('La Tarea no ha sido salvada. Inténtalo de nuevo');
}
}
}

function delete10($id=null){
for ($i = 0; $i < 10; $i++):
   if ($this->Libro->delete($i)){}
endfor;
$this->redirect(array('action'=>'index'),null,true);
}
function delete100($id=null){
   for ($i = 0; $i < 100; $i++):
   if ($this->Libro->delete($i)){}
endfor;
$this->redirect(array('action'=>'index'),null,true);
}
function delete1000($id=null){
   for ($i = 0; $i < 1000; $i++):
   if ($this->Libro->delete($i)){}
endfor;
$this->redirect(array('action'=>'index'),null,true);
}
function delete10000($id=null){
   for ($i = 0; $i < 10000; $i++):
   if ($this->Libro->delete($i)){}
endfor;
$this->redirect(array('action'=>'index'),null,true);
}
}
```

?>

- **Configuración en el Contenedor “Views”**

“libro-intex.ctp”

```
<h2>Nuestros ALBUMES</h2>
<?php if (empty($libros)): ?>
No hay albumes para mostrar
<?php else: ?>
<table>
<tr>
<th>ID</th>
<th>Titulo</th>
<th>Autor</th>
<th>Contexto</th>
<th>Acciones</th>
</tr>
<?php foreach ($libros as $libro): ?>
<tr>
<td>
<?php echo $libro['Libro']['id'] ?>
</td>
<td>
<?php echo $libro['Libro'] ['title'] ?>
</td>
<td>
<?php echo $libro['Libro']['author'] ?>
</td>
<td>
```

```
<?php echo $libro['Libro']['contents'] ?>
</td>
<td>
<?php echo $html->link('Editar', array ('action'=>'edit',$libro['Libro']['id'])); ?>
</td>
<td>
<?php echo $html->link('Eliminar', array ('action'=>'delete',$libro['Libro']['id'])); ?>
</td>
</tr>
<?php endforeach; ?>
</table>
<?php endif; ?>
<div id="izquierda" style="float: left;">
<div><?php echo $html->link('Añadir 1 Album', array ('action'=>'add')); ?></div>
<div><?php echo $html->link('Añadir 10 Albumes', array ('action'=>'add10')); ?></div>
<div><?php echo $html->link('Añadir 100 Albumes', array ('action'=>'add100'));
?></div>
<div><?php echo $html->link('Añadir 1000 Albumes', array ('action'=>'add1000'));
?></div>
<div><?php echo $html->link('Añadir 10000 Albumes', array ('action'=>'add10000'));
?></div>
</div>
<div id="derecha" style="float: right;">
<div><?php echo $html->link('Eliminar 10 Albumes', array ('action'=>'delete10'));
?></div>
<div><?php echo $html->link('Eliminar 100 Albumes', array ('action'=>'delete100'));
?></div>
<div><?php echo $html->link('Eliminar 1000 Albumes', array ('action'=>'delete1000'));
?></div>
<div><?php echo $html->link('Eliminar 10000 Albumes', array
('action'=>'delete10000')); ?></div>
</div>
```

Para los ingresos usamos el mismo código cambiando solo las etiquetas “libro-add.ctp, add100.ctp, add1000.ctp,add10000.ctp”

```
<?php echo $form->create('Libro'); ?>
<fieldset>
<legend>Agregar un Nuevo Album</legend>
<?php
echo $form->input('title');
echo $form->input('author');
echo $form->input('contents');
?>
</fieldset>
<?php echo $form->end('Añadir Tarea'); ?>
<?php echo $html->link('Listar Todos los Albumes', array ('action'=>'index')); ?>
```

Para las eliminaciones usamos el mismo código cambiando solo las etiquetas “libro-delete.ctp, delete100.ctp, delete1000.ctp,delete10000.ctp”

```
<?php echo $form->create('Libro'); ?>
<fieldset>
<legend>Eliminar este Album</legend>
<?php
echo $form->input('title');
echo $form->input('author');
echo $form->input('contents');
?>
</fieldset>
<?php echo $form->end('Eliminar'); ?>
<?php echo $html->link('Listar Todos los Albumes', array ('action'=>'index')); ?>
```

Para modificar “libro-edit.ctp”

```
<?php echo $form->create('Libro'); ?>
<fieldset>
<legend>Editar Este Album</legend>
<?php
echo $form->input('title');
echo $form->input('author');
echo $form->input('contents');
?>
</fieldset>
<?php echo $form->end('Guardar'); ?>
<?php echo $html->link('Listar Todos los Albumes', array ('action'=>'index')); ?>
```

2) CodeIgniter

- **Configuración de la Base de Datos en “Config-database.php”**

```
$active_group = 'default';
$active_record = TRUE;
$db['default']['hostname'] = 'localhost';
$db['default']['username'] = 'root';
$db['default']['password'] = '';
$db['default']['database'] = 'ci_series';
$db['default']['dbdriver'] = 'mysql';
$db['default']['dbprefix'] = '';
$db['default']['pconnect'] = TRUE;
$db['default']['db_debug'] = TRUE;
$db['default']['cache_on'] = FALSE;
$db['default']['cachedir'] = '';
$db['default']['char_set'] = 'utf8';
```

```
$db['default']['dbcollat'] = 'utf8_general_ci';
$db['default']['swap_pre'] = '';
$db['default']['autoinit'] = TRUE;
$db['default']['stricton'] = FALSE;
```

- **Configuración en el Contenedor “Models-site_model.php”**

```
<?php
class Site_model extends CI_Model{
   function get_records()
   {
     $query=$this->db->get('data');
     return $query->result();
   }
   function add_record($data,$data1)
   {
      $i=1;
      $maxi=$data1['repetir'];
     while($i<=$maxi)
     {
      $this->db->insert('data',$data);
      $i++;
     }
      return;
   }
   function update_record()
   {
      $this->db->where('id', 1);
     $this->db->update('data',$data);
}
   function delete_row($data)
   {
```

```
      $i=1;
      $maxi=$data['cantidad'];
      while($i<=$maxi)
      {
       $this->db->where('id',$i);
       $this->db->delete('data');
       $i++;
      }
   }
   function cleartable()
   {
      $this->db->trans_start();
      $this->db->query('truncate `data`');
$this->db->trans_complete();
   }
}
?>
```

- **Configuración en el Contenedor “Controllers”**

“Controllers-welcome.php”

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');
class Welcome extends CI_Controller {
        function __construct()
        {
                parent::__construct();
        }
        function index()
        {
                $this->load->view('welcome_message');
        }
```

```
}
```

“Controllers-site.php”

```
<?php
class Site extends CI_Controller
{
    function index(){
       $this->load->view('home');
   }
   function vista1(){
       $this->load->view('options_view_1');
   }
   function show(){
       $data = array();
        $query = $this->site_model->get_records();
           $data['records']= $query;
       if ($data){
       $this->load->view('options_view',$data);
       }
       else
       {$this->load->view('options_view_1');}
   }
   function create(){
       $data= array(
           'title' => $this->input->post('title'),
           'author'=>$this->input->post('author'),
           'contents'=>$this->input->post('content'));
       $data1= array('repetir' => $this->input->post('repetir'));
      $this->site_model->add_record($data,$data1);
$this->show();
   }
```

```
    function delete(){
        $data= array('cantidad' => $this->input->post('cantidad'));
        $this->site_model->delete_row($data);
        $this->show();
    }
    function truncatetable() {
        $this->site_model->cleartable();
$this->show();
    }
}
?>
```

- **Configuración en el Contenedor “Views”**

“Views-index.html”

```
<html>
<head>
<title>403 Forbidden</title>
</head>
<body>
<p>Directory access is forbidden.</p>
</body>
</html>
```

“Views-body.php”

```
<body>
<div id="wrapper">
<div id="header">
<h1><span>C</span>OMPARACIÓN DE FRAMEWORKS</h1>
</div>
```

```
<?php $this->load->view('nav'); ?>
<div id="body">
<div id="content"><div class="i">
<a href="<?php echo base_url(); ?>site/show"> pagina php option_view</a>
<h2><strong>Bienvenidos</strong> Estudio de Frameworks</h2>
<p>esta portal Web esta dedicado al estudio de los 3 frameworks  seleccionados previo
        a la obtención del titulo de ¨Ingeniería en Sistemas Informaticos¨
</p>
        <p> tema: <br></br>
        “ANÁLISIS DEL RENDIMIENTO DE FRAMEWORKS PHP PARA DESARROLLAR APLICACIONES WEB ÓPTIMAS. CASO PRÁCTICO: PORTAL ACADEMIA LINUX-ESPOCH”</p>
<h2 class="h"><span class="htop">HERRAMIENTAS</span><span class="hproject">UTILIZADAS</span><span class="hof">DE</span><span class="h2006">BENCHMARK</span></h2>
<div id="picbox-left">
<img src="<?php echo base_url();?>/images/pic_1.jpg" width="130" height="63" alt="Pic 1" />
</div>
<p>If you're looking for beautiful and professionally made templates you can find them at <a href="http://www.templatebeauty.com">Template Beauty</a>.</p>
<p>Even more websites all about website templates on <a href="http://www.justwebtemplates.com">Just Web Templates</a>.</p>
<h2><strong>My</strong> gallery</h2>
<div id="gallery">
<div id="bigpic">
<a href="http://www.freewebsitetemplates.com"><img src="<?php echo base_url();?>/images/pic_2.jpg" width="180" height="112" alt="Pic 2" /></a>
</div>
<div class="smallpic" id="pica">
<a href="http://www.freewebsitetemplates.com"><img src="<?php echo base_url();?>/images/pic_3.jpg" width="63" height="43" alt="Pic 3" /></a>
```

```
</div>
<div class="smallpic" id="picb">
<a href="http://www.freewebsitetemplates.com"><img src="<?php echo
base_url();?>/images/pic_4.jpg" width="63" height="43" alt="Pic 4" /></a>
</div>
<div class="smallpic" id="picc">
<a href="http://www.freewebsitetemplates.com"><img src="<?php echo
base_url();?>/images/pic_5.jpg" width="63" height="43" alt="Pic 5" /></a>
</div>
<div class="smallpic" id="picd">
<a href="http://www.freewebsitetemplates.com"><img src="<?php echo
base_url();?>/images/pic_6.jpg" width="63" height="43" alt="Pic 6" /></a>
</div>
<div class="clear"></div>
</div>
</div>
</div>
<div id="copyright">
&copy; Copyright 2011.model agency.
</div>
</div>
<?php $this->load->view('footer'); ?>
</div>
</body>
```

“Views-body_codeigniter.php”

```
<body>
<div id="wrapper">
        <div id="header">
        <h1><span>C</span>OMPARACIÓN DE FRAMEWORKS</h1>
        </div>
```

```
<?php $this->load->view('nav'); ?>
<div id="body">
<div id="contenedor">
<div class="i">
<div>
<h2>INGRESO DE ARTISTAS</h2>
<?php echo form_open('site/create');?>
<p>
<label for="title">  Title:       </label>
<input type="text" name="title" id="title"/>
</p>
<p>
<label for="author">  Author:   </label>
<input type="text" name="author" id="author"></input>
</p>
<p>
<label for="content">  Content:  </label>
<input type="text" name="content" id="content"></input>
</p>
<p>
<label for="repetir">  # repeticiones:  </label>
<input type="text" name="repetir" value="1" id="repetir" size="13"></input>
</p>
<p>
<input type="submit" value="agregar"></input>
</p>
<?php echo form_close();?>
</div>
<div id="barravertical">
        <p>LIBRO DE ARTISTAS </p>
<table  id="box-table-a" summary="Artistas de siempre">
<thead>
```

```
        <tr>
<th scope="col">ID</th>
<th scope="col">TITLE</th>
<th scope="col">AUTHOR</th>
<th scope="col">CONTENTS</th>
</tr>
</thead>
<tbody>
<?php foreach ($records as $r): ?>
<tr>
<td><?php echo $r->id; ?></td>
<td><?php echo $r->title; ?></td>
<td><?php echo $r->author; ?></td>
<td><?php echo $r->contents; ?></td>
</tr>
<?php endforeach; ?>
</tbody>
</table>
</div>
<div>
<h3>Eliminar Albumes</h3>
<?php echo form_open('site/delete');?>
<p>
<label for="cantidad">  cantidad:  </label>
<input type="text" name="cantidad" id="cantidad" size="10"></input>
</p>
<p>
<input type="submit" value="eliminar" ></input>
</p>
<?php echo form_close();?>
</div>
<div><a href="<?php echo base_url();?>site/truncatetable">eliminar todo</a></div>
```

```
</div>
</div>
</div>
<?php $this->load->view('footer'); ?>
</div>
</body>
```

“Views-delete_view.php”

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="<?php echo base_url();?>/css/style.css" rel="stylesheet" type="text/css" />
<title>Artistas de ayer hoy y siempre</title>
</head>
<body>
<h2>DELETE</h2>
<?php echo form_open('site/delete');?>
<?php echo form_close();?>
<div id="container">
        <p>MIS ARTISTAS FAVORITOS QUE RECUERDO</p>
<table  id="box-table-a" summary="Employee Pay Sheet">
<thead>
        <tr>
<th scope="col">ID</th>
<th scope="col">TITLE</th>
<th scope="col">AUTHOR</th>
<th scope="col">CONTENTS</th>
</tr>
</thead>
<tbody>
</tbody>
```

```
</table>
</div>
</body>
</html>
```

“Views-footer.php”

```
<div id="footer">
<a  href="<?php echo base_url(); ?>site/"> INICIO</a>
<div id="copyright">
                        &copy; Itel Vera & Edwin Sánchez.
                </div>
</div>
```

“Views-header.php”

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en"></html>
<head>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
        <title>Frameworks1</title>
<link rel="stylesheet" href="<?php echo base_url(); ?>css/stylehome.css"
type="text/css" charset="utf-8" />
</head>
```

“Views-header_codeigniter.php”

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en"></html>
<head>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
        <title>Frameworks1</title>
<link rel="stylesheet" href="<?php echo base_url(); ?>css/stylehome_codeigniter.css"
type="text/css" charset="utf-8" />
```

```
<link rel="stylesheet" href="<?php echo base_url(); ?>css/style.css" type="text/css"
charset="utf-8" />
</head>
```

“Views-home.php”

```
<?php $this->load->view('header'); ?>
<?php $this->load->view('body'); ?>
```

“Views-nav.php”

```
<div id="nav">
                <ul>
                        <li class="first"><a href="#"><em>I</em>NICIO</a></li>
                        <li><a href="<?php echo
base_url();?>"><em>F</em>FRAMEWORKS</a></li>
                        <li><a
href="http://localhost/CakeLibros/libros/"><em>C</em>AKE PHP</a></li>
                        <li><a href="<?php echo
base_url();?>site/show"><em>C</em>ODEIGNITER</a></li>
                        <li><a href="http://localhost/Zf-
libros/public/"><em>Z</em>END</a></li>
                </ul>
        </div>
```

“Views-options_view.php”

```
<?php $this->load->view('header_codeigniter'); ?>
<?php $this->load->view('body_codeigniter'); ?>
```

“Views-options_view1.php”

```
<?php $this->load->view('header_codeigniter'); ?>
<?php $this->load->view('body_codeigniter_1'); ?>
```

“Views-welcome_message.php”

```
<!DOCTYPE html>
<html lang="en">
<head>
	<meta charset="utf-8">
	<title>Welcome to CodeIgniter</title>
<style type="text/css">
body {
 background-color: #fff;
 margin: 40px;
font-family: Lucida Grande, Verdana, Sans-serif;
font-size: 14px;
 color: #4F5155;
}
a {
 color: #003399;
 background-color: transparent;
 font-weight: normal;
}
h1 {
 color: #444;
 background-color: transparent;
 border-bottom: 1px solid #D0D0D0;
 font-size: 16px;
 font-weight: bold;
 margin: 24px 0 2px 0;
 padding: 5px 0 6px 0;
}
```

```
code {
 font-family: Monaco, Verdana, Sans-serif;
 font-size: 12px;
 background-color: #f9f9f9;
 border: 1px solid #D0D0D0;
 color: #002166;
 display: block;
 margin: 14px 0 14px 0;
 padding: 12px 10px 12px 10px;
}
</style>
</head>
<body>
<h1>Welcome to CodeIgniter!</h1>
<p>The page you are looking at is being generated dynamically by CodeIgniter.</p>
<p>If you would like to edit this page you'll find it located at:</p>
<code>application/views/welcome_message.php</code>
<p>The corresponding controller for this page is found at:</p>
<code>application/controllers/welcome.php</code>
<p>If you are exploring CodeIgniter for the very first time, you should start by reading
the <a href="user_guide/">User Guide</a>.</p>
<p><br />Page rendered in {elapsed_time} seconds</p>
</body>
</html>
```

3) Zend

- **Configuración de la Base de Datos en “Configs-application.ini”**

```
[production]
phpSettings.display_startup_errors = 0
phpSettings.display_errors = 0
includePaths.library = APPLICATION_PATH "/../library"
bootstrap.path = APPLICATION_PATH "/Bootstrap.php"
bootstrap.class = "Bootstrap"
appnamespace = "Application"
resources.frontController.controllerDirectory = APPLICATION_PATH "/controllers"
resources.frontController.params.displayExceptions = 0
resources.db.adapter= PDO_MYSQL
resources.db.params.host= localhost
resources.db.params.username= root
resources.db.params.password= ""
resources.db.params.dbname= ci_series
resources.layout.layoutPath = APPLICATION_PATH "/layouts/scripts/"
resources.view.doctype = "XHTML1_STRICT"
[staging : production]
phpSettings.date.timezone= "America/Guayaquil"
[testing : production]
phpSettings.display_startup_errors = 1
phpSettings.display_errors = 1
[development : production]
phpSettings.display_startup_errors = 1
phpSettings.display_errors = 1
resources.frontController.params.displayExceptions = 1
```

- **Configuración en el Contenedor “Models-DbTable-Albums.php”**

```
<?php
class Application_Model_DbTable_Albums extends Zend_Db_Table_Abstract
{
    protected $_name = 'album';
    public function getAlbum($id)
    {
        $id = (int)$id;
        $row = $this->fetchRow('id = ' . $id);
        if (!$row) {
            throw new Exception("Could not find row $id");
        }
        return $row->toArray();
    }
    public function addAlbum($artist, $title, $contents)
    {
        $data = array(
            'artist' => $artist,
            'title' => $title,
            'contents' => $contents,
        );
        $this->insert($data);
    }
    public function add100Album($artist, $title, $contents)
    {
        $data = array(
            'artist' => $artist,
            'title' => $title,
            'contents' => $contents,
        );
         $i=1;
       while($i<=100)
       {
```

```
    $this->insert($data);
    $i++;
   }
}
 public function add1000Album($artist, $title, $contents)
{
    $data = array(
        'artist' => $artist,
        'title' => $title,
        'contents' => $contents,
    );
     $i=1;
   while($i<=1000)
   {
    $this->insert($data);
    $i++;
   }
}
 public function add10000Album($artist, $title, $contents)
{
    $data = array(
        'artist' => $artist,
        'title' => $title,
        'contents' => $contents,
    );
     $i=1;
   while($i<=10000)
   {
    $this->insert($data);
    $i++;
   }
}
```

```
public function updateAlbum($id, $artist, $title, $contents)
{
    $data = array(
        'artist' => $artist,
        'title' => $title,
        'contents' => $contents
    );
    $this->update($data, 'id = '. (int)$id);
}
public function deleteAlbum($id)
{
    $this->delete('id =' . (int)$id);
}
public function delete1Album()
{   $i=1;
    while($i<=1)
   {
    $this->delete('id =' . (int)$i);
    $i++;
   }
}
public function delete100Album()
{   $i=1;
    while($i<=100)
   {
    $this->delete('id =' . (int)$i);
    $i++;
   }
}

 public function delete1000Album()
{   $i=1;
```

```
        while($i<=1000)
       {
        $this->delete('id =' . (int)$i);
        $i++;
       }
    }
     public function delete10000Album()
    {   $i=1;
        while($i<=10000)
       {
        $this->delete('id =' . (int)$i);
$i++;
       }
    }
}
```

- **Configuración en el Contenedor “Controllers”**

“IndexController.php”

```
<?php
class IndexController extends Zend_Controller_Action
{
    public function init()
    {
        /* Initialize action controller here */
    }
    public function indexAction()
    {
         $albums = new Application_Model_DbTable_Albums();
         $this->view->albums = $albums->fetchAll();
    }
```

```
public function addAction()
{
    $form = new Application_Form_Album();
     $form->submit->setLabel('Add');
    $this->view->form = $form;
    if ($this->getRequest()->isPost()) {
        $formData = $this->getRequest()->getPost();
        if ($form->isValid($formData)) {
            $artist = $form->getValue('artist');
            $title = $form->getValue('title');
            $contents = $form->getValue('contents');
            $albums = new Application_Model_DbTable_Albums();
            $albums->addAlbum($artist, $title, $contents);
            $this->_helper->redirector('index');
        } else {
            $form->populate($formData);
        }
    }
}
public function add100Action()
{
    $form = new Application_Form_Album();
     $form->submit->setLabel('Add');
    $this->view->form = $form;
    if ($this->getRequest()->isPost()) {
        $formData = $this->getRequest()->getPost();
        if ($form->isValid($formData)) {
            $artist = $form->getValue('artist');
            $title = $form->getValue('title');
            $contents = $form->getValue('contents');
            $albums = new Application_Model_DbTable_Albums();
            $albums->add100Album($artist, $title, $contents);
```

```
            $this->_helper->redirector('index');
        } else {
            $form->populate($formData);
        }
    }
}
public function add1000Action()
{
    // action body
    $form = new Application_Form_Album();
     $form->submit->setLabel('Add');
    $this->view->form = $form;
    if ($this->getRequest()->isPost()) {
        $formData = $this->getRequest()->getPost();
        if ($form->isValid($formData)) {
            $artist = $form->getValue('artist');
            $title = $form->getValue('title');
            $contents = $form->getValue('contents');
            $albums = new Application_Model_DbTable_Albums();
            $albums->add1000Album($artist, $title, $contents);
            $this->_helper->redirector('index');
        } else {
            $form->populate($formData);
        }
    }
}
public function add10000Action()
{
    $form = new Application_Form_Album();
     $form->submit->setLabel('Add');
    $this->view->form = $form;
    if ($this->getRequest()->isPost()) {
```

```
        $formData = $this->getRequest()->getPost();
        if ($form->isValid($formData)) {
            $artist = $form->getValue('artist');
            $title = $form->getValue('title');
            $contents = $form->getValue('contents');
            $albums = new Application_Model_DbTable_Albums();
            $albums->add10000Album($artist, $title, $contents);
            $this->_helper->redirector('index');
        } else {
            $form->populate($formData);
        }
    }
}
public function editAction()
{
    // action body
     $form = new Application_Form_Album();
    $form->submit->setLabel('Save');
    $this->view->form = $form;
    if ($this->getRequest()->isPost()) {
        $formData = $this->getRequest()->getPost();
        if ($form->isValid($formData)) {
            $id = (int)$form->getValue('id');
            $artist = $form->getValue('artist');
            $title = $form->getValue('title');
            $title = $form->getValue('contents');
            $albums = new Application_Model_DbTable_Albums();
            $albums->updateAlbum($id, $artist, $title, $contents);
            $this->_helper->redirector('index');
        } else {
            $form->populate($formData);
        }
```

```
    } else {
        $id = $this->_getParam('id', 0);
        if ($id > 0) {
            $albums = new Application_Model_DbTable_Albums();
            $form->populate($albums->getAlbum($id));
        }
    }
}
public function deleteAction()
{
    // action body
    if ($this->getRequest()->isPost()) {
        $del = $this->getRequest()->getPost('del');
        if ($del == 'Yes') {
            $id = $this->getRequest()->getPost('id');
            $albums = new Application_Model_DbTable_Albums();
            $albums->deleteAlbum($id);
        }
        $this->_helper->redirector('index');
    } else {
        $id = $this->_getParam('id', 0);
        $albums = new Application_Model_DbTable_Albums();
        $this->view->album = $albums->getAlbum($id);
    }
}
public function delete1Action()
{
            $albums = new Application_Model_DbTable_Albums();
            $albums->delete1Album();
            $this->_helper->redirector('index');
}
 public function delete100Action()
```

```
    {
            $albums = new Application_Model_DbTable_Albums();
            $albums->delete100Album();
            $this->_helper->redirector('index');
    }
  public function delete1000Action()
    {
             $albums = new Application_Model_DbTable_Albums();
             $albums->delete1000Album();
             $this->_helper->redirector('index');
    }
      public function delete10000Action()
    {
             $albums = new Application_Model_DbTable_Albums();
             $albums->delete10000Album();
             $this->_helper->redirector('index');
}
}
```

“ErrorController.php”

```
<?php
class ErrorController extends Zend_Controller_Action
{
    public function errorAction()
    {
        $errors = $this->_getParam('error_handler');
        if (!$errors || !$errors instanceof ArrayObject) {
            $this->view->message = 'You have reached the error page';
            return;
        }
```

```
        switch ($errors->type) {
            case Zend_Controller_Plugin_ErrorHandler::EXCEPTION_NO_ROUTE:
            case
Zend_Controller_Plugin_ErrorHandler::EXCEPTION_NO_CONTROLLER:
            case Zend_Controller_Plugin_ErrorHandler::EXCEPTION_NO_ACTION:
                // 404 error -- controller or action not found
                $this->getResponse()->setHttpResponseCode(404);
                $priority = Zend_Log::NOTICE;
                $this->view->message = 'Page not found';
                break;
            default:
                // application error
                $this->getResponse()->setHttpResponseCode(500);
                $priority = Zend_Log::CRIT;
                $this->view->message = 'Application error';
                break;
        }
        // Log exception, if logger available
        if ($log = $this->getLog()) {
            $log->log($this->view->message, $priority, $errors->exception);
            $log->log('Request Parameters', $priority, $errors->request->getParams());
        }
        if ($this->getInvokeArg('displayExceptions') == true) {
            $this->view->exception = $errors->exception;
        }
        $this->view->request   = $errors->request;
    }
    public function getLog()
    {
        $bootstrap = $this->getInvokeArg('bootstrap');
        if (!$bootstrap->hasResource('Log')) {
            return false;
```

```
        }
        $log = $bootstrap->getResource('Log');
        return $log;
}
}
```

- **Configuración en el Contenedor “Views-scripts”**

“Error.phtml”

```
<html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Zend Framework Default Application</title>
</head>
<body>
<h1>An error occurred</h1>
<h2><?php echo $this->message ?></h2>
<?php if (isset($this->exception)): ?>
<h3>Exception information:</h3>
<p>
<b>Message:</b><?php echo $this->exception->getMessage() ?>
</p>
<h3>Stack trace:</h3>
<pre><?php echo $this->exception->getTraceAsString() ?>
</pre>
<h3>Request Parameters:</h3>
<pre><?php echo $this->escape(var_export($this->request->getParams(), true)) ?>
</pre>
<?php endif ?>
</body>
</html>
```

“index.phtml”

```
<?php $this->title = "MIS ALBUMES";?>
<?$this->headTitle($this->title);?>
<div id="barravertical">
        <p>LIBRO DE ARTISTAS </p>
<table id="box-table-a" summary="Employee Pay Sheet">
<tr>
<th>Id</th>
<th>Title</th>
<th>Artist</th>
<th>Acciones </th>
</tr>
<?php foreach($this->albums as $album) : ?>
<tr>
<td><?php echo $this->escape($album->id);?></td>
<td><?php echo $this->escape($album->title);?></td>
<td><?php echo $this->escape($album->artist);?></td>
<td><?php echo $this->escape($album->contents);?></td>
<td>
<a href="<?php echo $this->url(array('controller'=>'index',
        'action'=>'edit', 'id'=>$album->id));?>">Edit</a>
<a href="<?php echo $this->url(array('controller'=>'index',
        'action'=>'delete', 'id'=>$album->id));?>">Delete</a>
</td>
</tr>
<?php endforeach; ?>
</table>
</div>
<div id="menug">
<div id="menua">
```

```
<div><a href="<?php echo $this->url(array('controller'=>'index',
      'action'=>'add'));?>">Add 1 albume</a></div>
<div><a href="<?php echo $this->url(array('controller'=>'index',
      'action'=>'add100'));?>">Add 100 albumes</a></div>
<div><a href="<?php echo $this->url(array('controller'=>'index',
      'action'=>'add1000'));?>">Add 1000 albumes </a></div>
<div><a href="<?php echo $this->url(array('controller'=>'index',
      'action'=>'add10000'));?>">Add 10000 albumes </a></div>
</div>
<div id="menub">
<div><a href="<?php echo $this->url(array('controller'=>'index',
      'action'=>'delete1'));?>">Delete 1 album</a></div>
<div><a href="<?php echo $this->url(array('controller'=>'index',
      'action'=>'delete100'));?>">Delete 100 albumes</a></div>
<div><a href="<?php echo $this->url(array('controller'=>'index',
      'action'=>'delete1000'));?>">Delete 1000 albumes</a></div>
<div><a href="<?php echo $this->url(array('controller'=>'index',
'action'=>'delete10000'));?>">Delete 10000 albumes </a></div>
</div>
</div>
```

“edit.phtml”

```
<?php
$this->title = "Edit album";
$this->headTitle($this->title);
echo $this->form ;
?>
```

“add.phtml”

```
<?php
$this->title = "Add new album";
$this->headTitle($this->title);
echo $this->form ;
?>
```

“delete.phtml”

```
<?php
$this->title = "Delete album";
$this->headTitle($this->title);
 ?>
<p>Are you sure that you want to delete </p>
<form action="<?php echo $this->url(array('action'=>'delete1')); ?>" method="post">
<div>
<input type="submit" name="del" value="Yes" />
<input type="submit" name="del" value="No" />
</div>
</form>
```

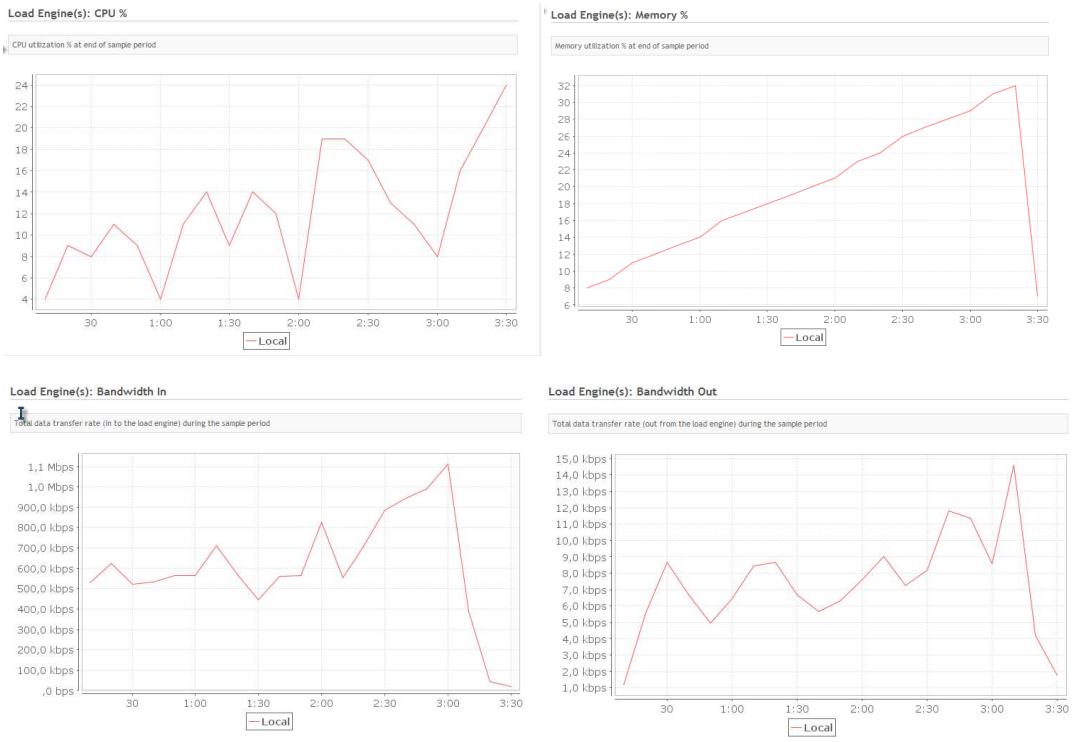
ANEXO 2

Se presentaran las pantallas de los indicadoressacados de cada uno de los escenarios realizadas a cada uno de los FrameworksCakePHP, CodeIgniter, Zend Framework con sus respectivas pruebas, Uno, Cien, Mil y Diez Mil

A) CAKEPHP

- **Indicador 1: Ingeniería de Carga**

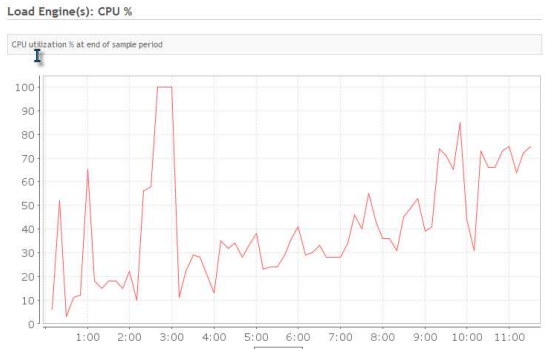
  - **Prueba 1.1 una transacción**

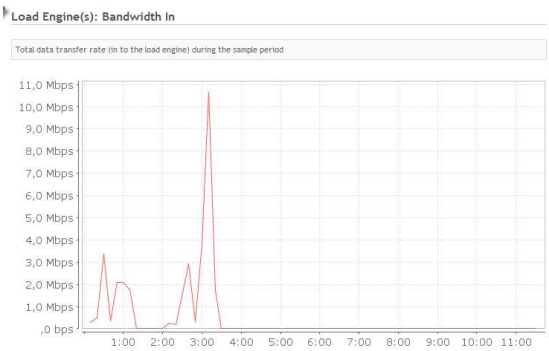


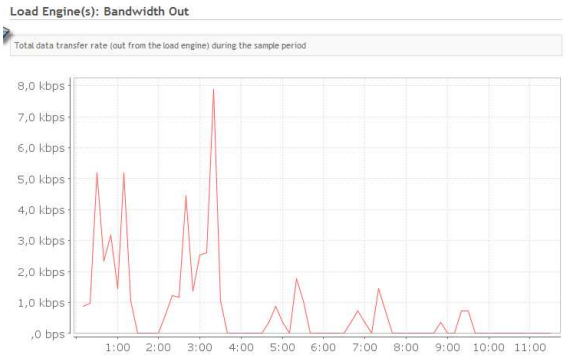
  - **Prueba 1.2 cien transacciones**

- **Prueba 1.3 mil transacciones**

- **Prueba 1.4 diez mil transacciones**







- **Indicador 2: Línea Base**

  - **Prueba 2.1 una transacción**

Minimum Bandwidth Requirement (estimated)

Testcase	Avg BandwidthConsumed	Users	Simulated User Network Speed
CakeAdd	98,0 kbps	1	1,5 Mbps (T1)

EXPORT

Per-testcase bandwidth consumption (per user)

Testcase	Avg BandwidthConsumed	Simulated User Network Speed	Size (bytes)	Estimated Duration (MM:SS)	Repeat Delay (MM:SS)
CakeAdd	98,0 kbps	1,5 Mbps (T1)	347,2 KB	29	1

EXPORT

  - **Prueba 2.2 cien transacciones**

Minimum Bandwidth Requirement (estimated)

Testcase	Avg BandwidthConsumed	Users	Simulated User Network Speed
CakeCien	11,3 Mbps	100	1,5 Mbps (T1)

EXPORT

Per-testcase bandwidth consumption (per user)

Testcase	Avg BandwidthConsumed	Simulated User Network Speed	Size (bytes)	Estimated Duration (MM:SS)	Repeat Delay (MM:SS)
CakeCien	112,5 kbps	1,5 Mbps (T1)	367,0 KB	26	1

EXPORT

- **Prueba 2.3 mil transacciones**

Minimum Bandwidth Requirement (estimated)

Testcase	Avg BandwidthConsumed	Users	Simulated User Network Speed
CakeMil	85,7 Mbps	1.000	1,5 Mbps (T1)

EXPORT

Per-testcase bandwidth consumption (per user)

Testcase	Avg BandwidthConsumed	Simulated User Network Speed	Size (bytes)	Estimated Duration (MM:SS)	Repeat Delay (MM:SS)
CakeMil	85,7 kbps	1,5 Mbps (T1)	548,3 KB	52	1

EXPORT

- **Prueba 2.4 diez mil transacciones**

Minimum Bandwidth Requirement (estimated)

Testcase	Avg BandwidthConsumed	Users	Simulated User Network Speed
CakeDiezMil	724,3 Mbps	10.000	1,5 Mbps (T1)

EXPORT

Per-testcase bandwidth consumption (per user)

Testcase	Avg BandwidthConsumed	Simulated User Network Speed	Size (bytes)	Estimated Duration (MM:SS)	Repeat Delay (MM:SS)
CakeDiezMil	72,4 kbps	1,5 Mbps (T1)	2,4 MB	4:40	1

EXPORT

- **Indicador 3: Carga Transaccional de Alta**

- **Prueba 3.1 La prueba de una transacción**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
liga1a.jpg	2.793	323,0 KB	
CakePHP: the rapid development php framework:Libros [1]	1.608	17,0 KB	
CakePHP: the rapid development php framework:Libros [2]	.151	2,8 KB	
CakePHP: the rapid development php framework:Libros [3]	.435	4,4 KB	

EXPORT

Summary

Summary information about the testcase.

Total Duration	25.14
Pages	4
URLs	10

- **Prueba 3.2 cien transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
liga1a.jpg	2.714	323,0 KB	
CakePHP: the rapid development php framework:Libros [1]	.574	17,0 KB	
CakePHP: the rapid development php framework:Libros [2]	.128	2,8 KB	
CakePHP: the rapid development php framework:Libros [3]	2.310	24,3 KB	

EXPORT

Summary

Summary information about the testcase.

Total Duration	23.04
Pages	4
URLs	10

- **Prueba 3.3 mil transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
liga1a.jpg	2.959	323,0 KB	
CakePHP: the rapid development php framework:Libros [1]	.609	17,0 KB	
CakePHP: the rapid development php framework:Libros [2]	.132	2,8 KB	
CakePHP: the rapid development php framework:Libros [3]	19.719	205,5 KB	Page Duration <= 4000ms

EXPORT

Summary

Summary information about the testcase.

Total Duration	46.91
Pages	4
URLs	10

- **Prueba 3.4 diez mil transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
liga1a.jpg	4.698	323,0 KB	Page Duration <= 4000ms
CakePHP: the rapid development php framework:Libros [1]	1.844	17,0 KB	
CakePHP: the rapid development php framework:Libros [2]	1.342	2,8 KB	
20minutos.es	5.667	35,0 KB	Page Duration <= 4000ms
CakePHP: the rapid development php framework:Libros [3]	4:02.325	2,1 MB	Page Duration <= 4000ms

EXPORT

Summary

Summary information about the testcase.

Total Duration	4:23.85
Pages	5
URLs	13

- **Indicador 4: Carga Transaccional de Baja**

- **Prueba 4.1 una transacción**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
<webpage> [1]	.599	4,3 KB	
<webpage> [2]	.000	0 B	
<webpage> [3]	.000	0 B	
<webpage> [4]	.000	0 B	
CakePHP: the rapid development php framework:Libros [1]	1.358	3,8 KB	
(srchdetect12.predictad.com) [1]	.000	0 B	
CakePHP: the rapid development php framework:Libros [2]	1.286	6,3 KB	
CakePHP: the rapid development php framework:Libros [3]	.383	4,7 KB	
(srchdetect12.predictad.com) [2]	.000	0 B	
CakePHP: the rapid development php framework:Libros [4]	1.981	7,1 KB	
CakePHP: the rapid development php framework:Libros [5]	.375	4,4 KB	

Summary

Summary information about the testcase.

Total Duration	29.37
Pages	11
URLs	16

- **Prueba 4.2 cien transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
<webpage>	.700	3,2 KB	
CakePHP: the rapid development php framework:Libros [1]	.496	3,8 KB	
(srchdetect16.predictad.com)	.000	0 B	
CakePHP: the rapid development php framework:Libros [2]	1.225	6,4 KB	
CakePHP: the rapid development php framework:Libros [3]	2.538	24,8 KB	
Web Performance Analyzer™: ERROR	.000	0 B	
CakePHP: the rapid development php framework:Libros [4]	2.234	4,4 KB	

Summary

Summary information about the testcase.

Total Duration	36.17
Pages	7
URLs	10

- **Prueba 4.3 mil transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
<webpage>	.674	3,2 KB	
CakePHP: the rapid development php framework:Libros [1]	.313	3,8 KB	
(srchdetect16.predictad.com)	.000	0 B	
CakePHP: the rapid development php framework:Libros [2]	1.204	6,4 KB	
CakePHP: the rapid development php framework:Libros [3]	23.420	209,7 KB	Page Duration <= 4000ms
Web Performance Analyzer™: ERROR	.000	0 B	
CakePHP: the rapid development php framework:Libros [4]	35.243	4,9 KB	Page Duration <= 4000ms

Summary

Summary information about the testcase.

Total Duration	1:32.99
Pages	7
URLs	10

- **Prueba 4.4 diez mil transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
<webpage> [1]	.682	4,3 KB	
CakePHP: the rapid development php framework:Libros [1]	.311	3,8 KB	
Web Performance Analyzer™: ERROR [1]	.000	0 B	
<webpage> [2]	.000	0 B	
CakePHP: the rapid development php framework:Libros [2]	2.645	2,9 KB	
<webpage> [3]	.000	0 B	
<webpage> [4]	2.950	3,4 KB	
CakePHP: the rapid development php framework:Libros [3]	3:47.201	2,0 MB	Page Duration <= 4000ms
(srchdetect12.predictad.com)	8:05.768	0 B	Page Duration <= 4000ms
CakePHP: the rapid development php framework:Libros [4]	8:00.827	4,9 KB	Page Duration <= 4000ms
Web Performance Analyzer™: ERROR [2]	.000	0 B	

EXPORT

Summary

Summary information about the testcase.

Total Duration	12:15.98
Pages	11
URLs	15

- **Indicador 5: Integridad**

- **Prueba 5.1 una transacción**

Page Analysis: CakePHP: the rapid development php framework:Libros [3]

Testcase	CakeAdd (1)
Page URL	http://localhost/CakeLibros/libros
Baseline Size	4,4 KB
URLs	2
Baseline Duration	.435
Performance Goal	4.000
Attempts	44
Completions	44
Successes	44
Failures	0
Average Duration	.373
Total Duration	16.418

- **Prueba 5.2 cien transacciones**

Page Analysis: CakePHP: the rapid development php framework:Libros [3]

Testcase	CakeCien (1)
Page URL	http://localhost/CakeLibros/libros
Baseline Size	24,3 KB
URLs	2
Baseline Duration	2.310
Performance Goal	4.000
Attempts	38
Completions	38
Successes	38
Failures	0
Average Duration	4.501
Total Duration	2:51.040

- **Prueba 5.3 mil transacciones**

Page Analysis: CakePHP: the rapid development php framework:Libros [3]

Testcase	CakeMil (1)
Page URL	http://localhost/CakeLibros/libros
Baseline Size	205,5 KB
URLs	2
Baseline Duration	19.719
Performance Goal	4.000
Attempts	18
Completions	18
Successes	18
Failures	0
Average Duration	51.371
Total Duration	15:24.692

- **Prueba 5.4 diez mil transacciones**

Page Analysis: CakePHP: the rapid development php framework:Libros [3]

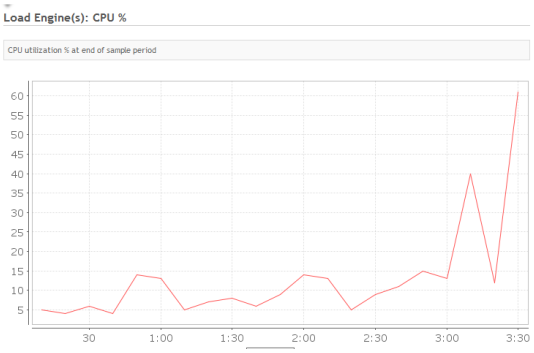
Testcase	CakeDiezMil (1)
Page URL	http://localhost/CakeLibros/libros
Baseline Size	2,1 MB
URLs	2
Baseline Duration	4:02.325
Performance Goal	4.000
Attempts	10
Completions	10
Successes	0
Failures	10
Average Duration	8:00.008
Total Duration	80:00.088

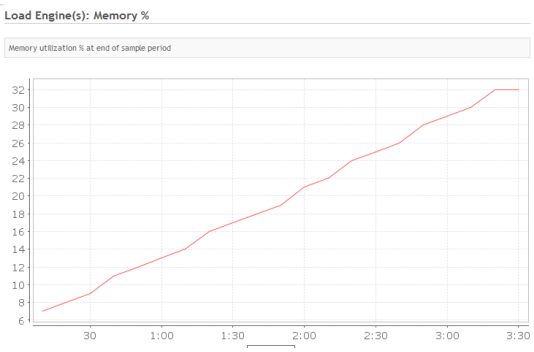
B) CODEIGNITER

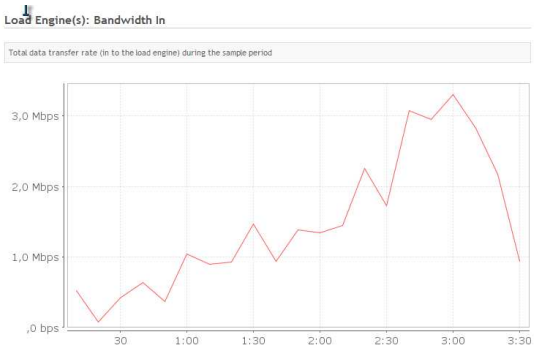
- **Indicador 1: Ingeniería de Carga**

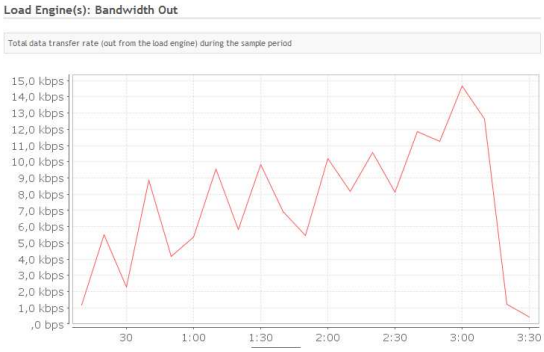
  - **Prueba 1.1 una transacción**

- **Prueba 1.2 cien transacciones**

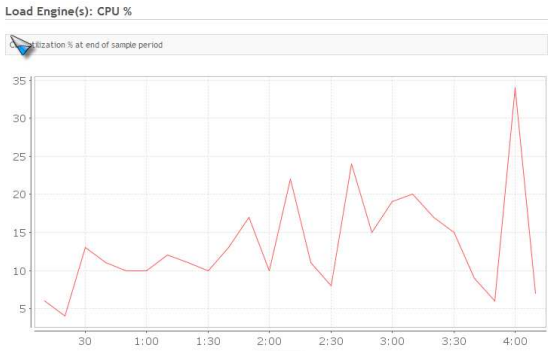


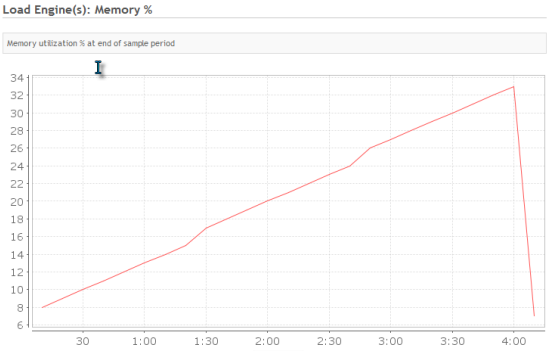


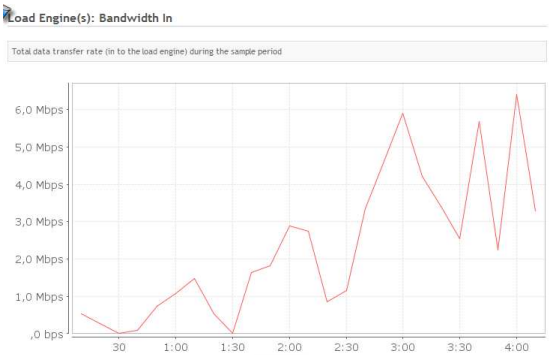




- **Prueba 1.3 mil transacciones**







- **Prueba 1.4 diez mil transacciones**

- **Indicador 2: Línea Base**

  - **Prueba 2.1 una transacción**

Minimum Bandwidth Requirement (estimated)

Testcase	Avg BandwidthConsumed	Users	Simulated User Network Speed
CodeIgniterAdd	147,8 kbps	1	1,5 Mbps (T1)

EXPORT

Per-testcase bandwidth consumption (per user)

Testcase	Avg BandwidthConsumed	Simulated User Network Speed	Size (bytes)	Estimated Duration (MM:SS)	Repeat Delay (MM:SS)
CodeIgniterAdd	147,8 kbps	1,5 Mbps (T1)	365,0 KB	20	1

EXPORT

- **Prueba 2.2 cien transacciones**

Minimum Bandwidth Requirement (estimated)

Testcase	Avg BandwidthConsumed	Users	Simulated User Network Speed
CodeIgniterCien	12,5 Mbps	100	1,5 Mbps (T1)

EXPORT

Per-testcase bandwidth consumption (per user)

Testcase	Avg BandwidthConsumed	Simulated User Network Speed	Size (bytes)	Estimated Duration (MM:SS)	Repeat Delay (MM:SS)
CodeIgniterCien	125,3 kbps	1,5 Mbps (T1)	378,4 KB	24	1

EXPORT

- **Prueba 2.3 mil transacciones**

Minimum Bandwidth Requirement (estimated)

Testcase	Avg BandwidthConsumed	Users	Simulated User Network Speed
CodeIgniterMil	104,9 Mbps	1.000	1,5 Mbps (T1)

EXPORT

Per-testcase bandwidth consumption (per user)

Testcase	Avg BandwidthConsumed	Simulated User Network Speed	Size (bytes)	Estimated Duration (MM:SS)	Repeat Delay (MM:SS)
CodeIgniterMil	104,9 kbps	1,5 Mbps (T1)	498,5 KB	38	1

EXPORT

- **Prueba 2.4 diez mil transacciones**

Minimum Bandwidth Requirement (estimated)

Testcase	Avg BandwidthConsumed	Users	Simulated User Network Speed
CodeIgniterDiezMil	642,6 Mbps	10.000	1,5 Mbps (T1)

EXPORT

Per-testcase bandwidth consumption (per user)

Testcase	Avg BandwidthConsumed	Simulated User Network Speed	Size (bytes)	Estimated Duration (MM:SS)	Repeat Delay (MM:SS)
CodeIgniterDiezMil	64,3 kbps	1,5 Mbps (T1)	1,7 MB	3:45	1

EXPORT

- **Indicador 3: Carga Transaccional de Alta**

  - **Prueba 3.1 La prueba de una transacción**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
liga1a.jpg	2.661	323,0 KB	
Frameworks1 [1]	.531	38,1 KB	
Frameworks1 [2]	.415	3,9 KB	

EXPORT

Summary

Summary information about the testcase.

Total Duration	15.64
Pages	3
URLs	12

  - **Prueba 3.2 cien transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
liga1a.jpg	2.646	322,9 KB	
Frameworks1 [1]	.369	38,1 KB	
Frameworks1 [2]	1.844	17,4 KB	

EXPORT

Summary

Summary information about the testcase.

Total Duration	20.32
Pages	3
URLs	12

- **Prueba 3.3 mil transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
liga1a.jpg	2.829	322,9 KB	
Frameworks1 [1]	.210	38,1 KB	
Frameworks1 [2]	18.876	137,5 KB	Page Duration <= 4000ms

EXPORT

Summary

Summary information about the testcase.

Total Duration	33.42
Pages	3
URLs	12

- **Prueba 3.4 diez mil transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
liga1a.jpg	3.165	323,0 KB	
Frameworks1 [1]	.412	38,1 KB	
Frameworks1 [2]	3:08.717	1,4 MB	Page Duration <= 4000ms

EXPORT

Summary

Summary information about the testcase.

Total Duration	3:32.69
Pages	3
URLs	12

- **Indicador 4: Carga Transaccional de Baja**

  - **Prueba 4.1 una transacción**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
<webpage> [1]	3.797	4,3 KB	
Frameworks1 [1]	.058	3,8 KB	
<webpage> [2]	.000	0 B	
<webpage> [3]	.000	0 B	
<webpage> [4]	.000	0 B	
<webpage> [5]	.298	3,0 KB	
Frameworks1 [2]	.175	4,1 KB	
Frameworks1 [3]	1.112	7,4 KB	

EXPORT

Summary

Summary information about the testcase.

Total Duration	19.24
Pages	8
URLs	10

  - **Prueba 4.2 cien transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
<webpage> [1]	.654	3,2 KB	
Frameworks1 [1]	.076	3,8 KB	
<webpage> [2]	.000	0 B	
Frameworks1 [2]	2.168	17,4 KB	
Frameworks1 [3]	3.859	7,4 KB	

EXPORT

Summary

Summary information about the testcase.

Total Duration	23.28
Pages	5
URLs	6

- **Prueba 4.3 mil transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
<text> [1]	.676	3,2 KB	
Frameworks1 [1]	.071	3,8 KB	
<text> [2]	.393	3,5 KB	
Frameworks1 [2]	19.122	142,5 KB	Page Duration <= 4000ms
<text> [3]	.980	3,5 KB	
Frameworks1 [3]	18.856	3,9 KB	Page Duration <= 4000ms

EXPORT

Summary

Summary information about the testcase.

Total Duration	1:00.10
Pages	6
URLs	6

- **Prueba 4.4 diez mil transacciones**

Summary

Summary information about the testcase.

Total Duration	6:56.49
Pages	8
URLs	10

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
<webpage> [1]	.584	4,3 KB	
Frameworks1 [1]	.102	3,8 KB	
<webpage> [2]	.000	0 B	
<webpage> [3]	.000	0 B	
<webpage> [4]	.397	3,5 KB	
Frameworks1 [2]	3:09.481	1,3 MB	Page Duration <= 4000ms
<text> [3]	.982	3,5 KB	
Frameworks1 [3]	3:10.163	3,9 KB	Page Duration <= 4000ms

EXPORT

- **Indicador 5: Integridad**

  - **Prueba 5.1 una transacción**

Page Analysis: Frameworks1 [2]

Testcase	CodeIgniterAdd (1)
Page URL	http://localhost/Crud/site/create
Baseline Size	3,9 KB
URLs	1
Baseline Duration	.415
Performance Goal	4.000
Attempts	53
Completions	53
Successes	53
Failures	0
Average Duration	.154
Total Duration	8.208

- **Prueba 5.2 cien transacciones**

Page Analysis: Frameworks1 [2]

Testcase	CodeIgniterCien (1)
Page URL	http://localhost/Crud/site/create
Baseline Size	17,4 KB
URLs	1
Baseline Duration	1.844
Performance Goal	4.000
Attempts	41
Completions	41
Successes	41
Failures	0
Average Duration	3.063
Total Duration	2:05.616

- **Prueba 5.3 mil transacciones**

Page Analysis: Frameworks1 [2]

Testcase	CodeIgniterMil (1)
Page URL	http://localhost/Crud/site/create
Baseline Size	137,5 KB
URLs	1
Baseline Duration	18.876
Performance Goal	4.000
Attempts	21
Completions	21
Successes	21
Failures	0
Average Duration	39.893
Total Duration	13:57.770

- **Prueba 5.4 diez mil transacciones**

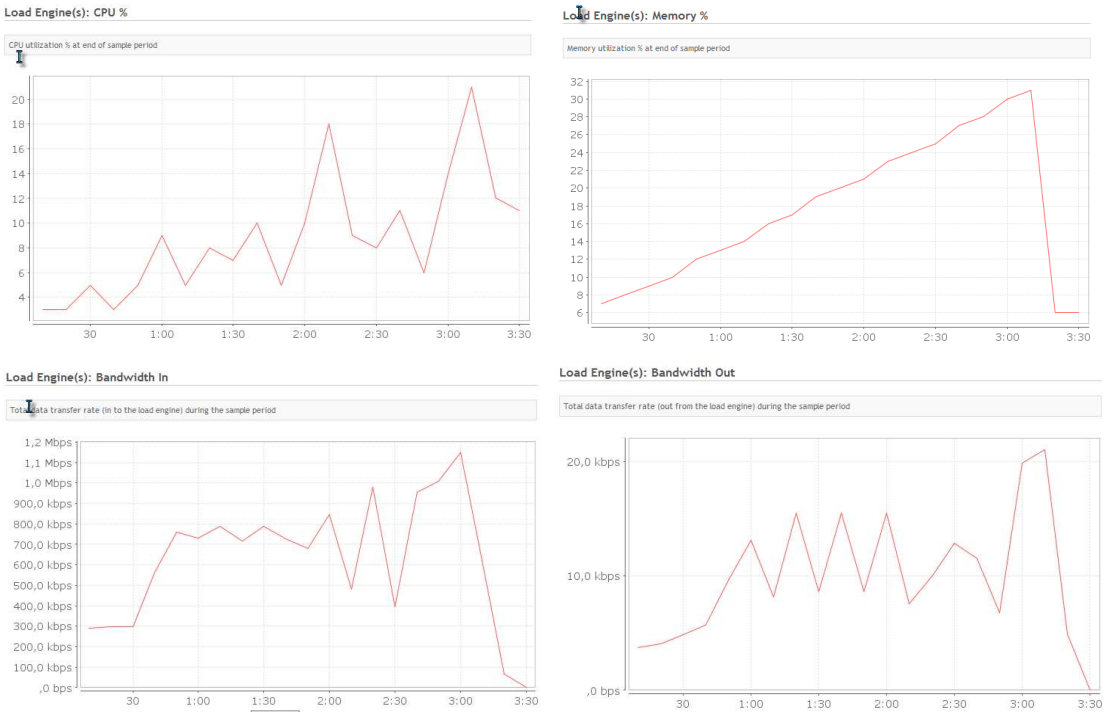
Page Analysis: Frameworks1 [2]

Testcase	CodeIgniterDiezMil (1)
Page URL	http://localhost/Crud/site/create
Baseline Size	1,4 MB
URLs	1
Baseline Duration	3:08.717
Performance Goal	4.000
Attempts	10
Completions	10
Successes	0
Failures	10
Average Duration	4:00.003
Total Duration	40:00.036

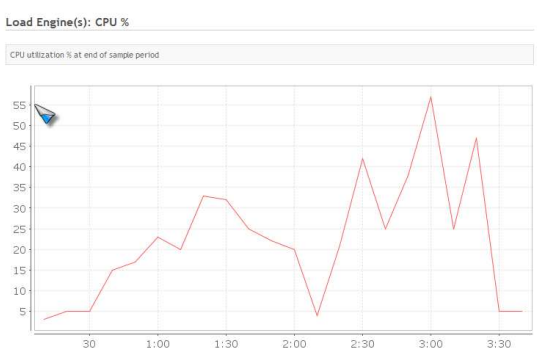
C) ZEND

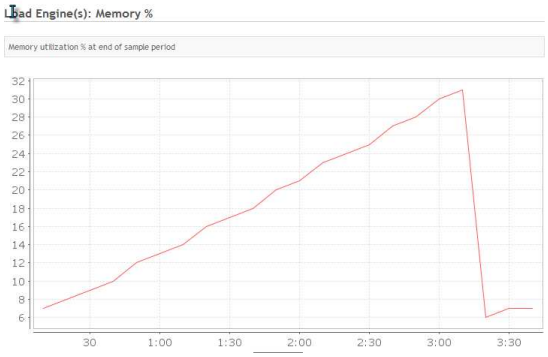
- **Indicador 1: Ingeniería de Carga**

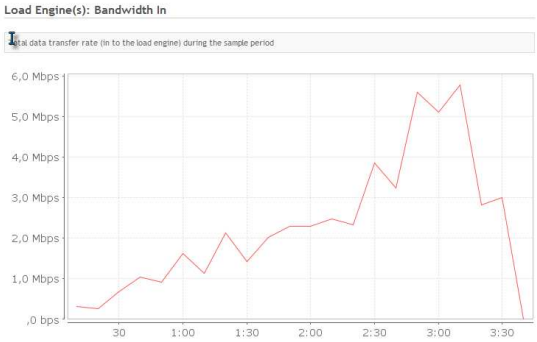
  - **Prueba 1.1 una transacción**

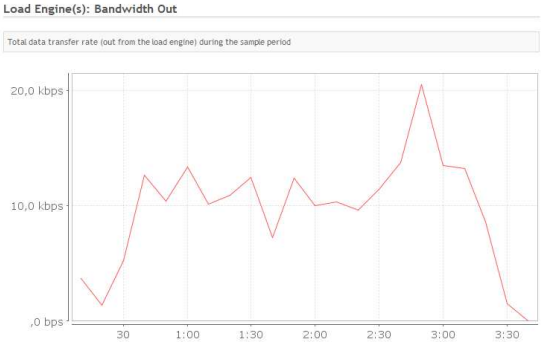


- **Prueba 1.2 cien transacciones**

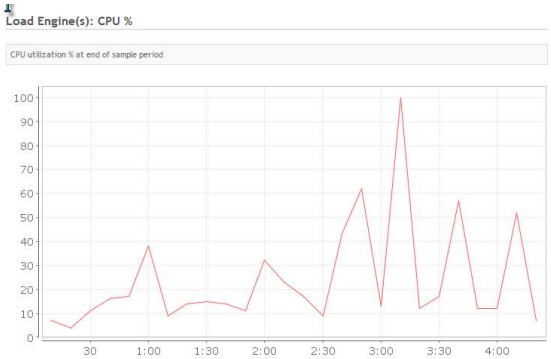


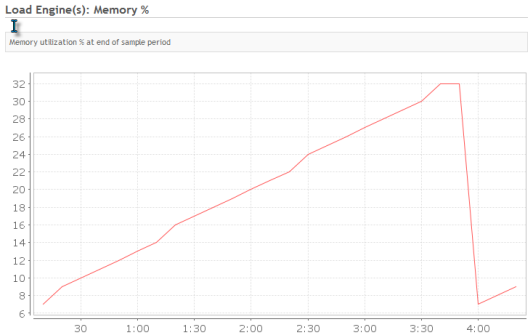


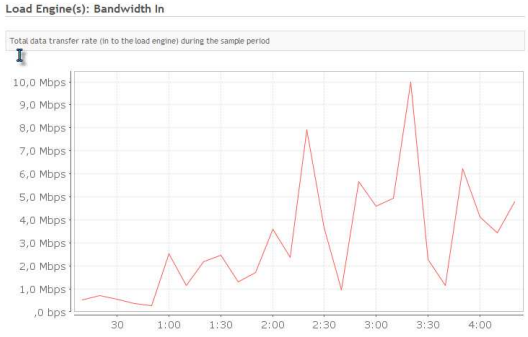


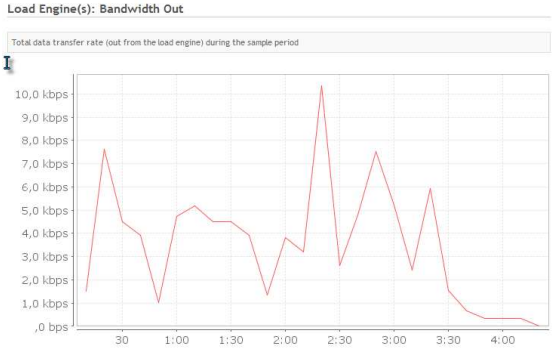


- **Prueba 1.3 mil transacciones**

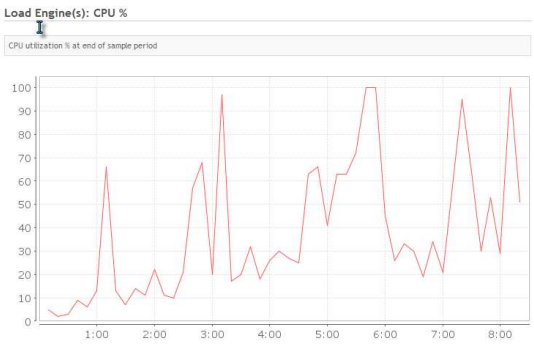


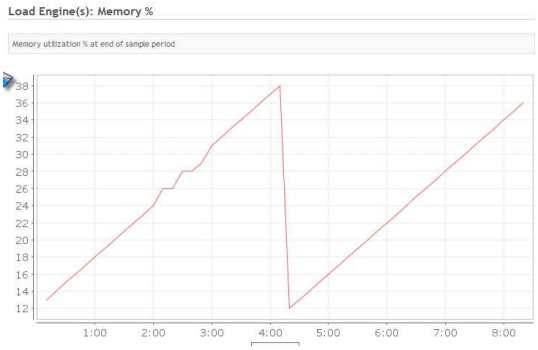


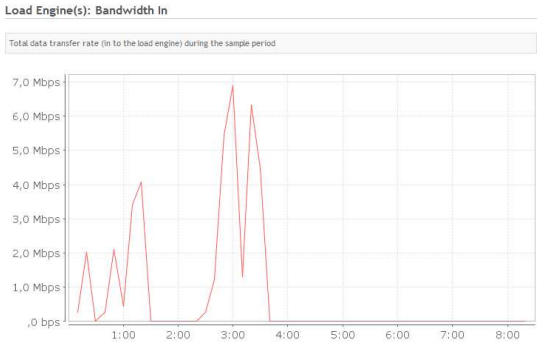


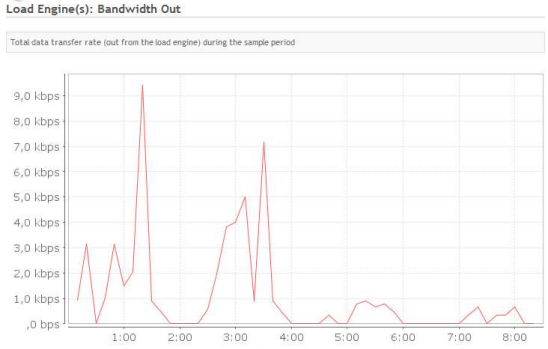


- **Prueba 1.4 diez mil transacciones**









- **Indicador 2: Línea Base**

- **Prueba 2.1 una transacción**

Minimum Bandwidth Requirement (estimated)

Testcase	Avg BandwidthConsumed	Users	Simulated User Network Speed
ZendAdd	128,3 kbps	1	1,5 Mbps (T1)

EXPORT

Per-testcase bandwidth consumption (per user)

Testcase	Avg BandwidthConsumed	Simulated User Network Speed	Size (bytes)	Estimated Duration (MM:SS)	Repeat Delay (MM:SS)
ZendAdd	128,3 kbps	1,5 Mbps (T1)	369,2 KB	23	1

EXPORT

- **Prueba 2.2 cien transacciones**

Minimum Bandwidth Requirement (estimated)

Testcase	Avg BandwidthConsumed	Users	Simulated User Network Speed
ZendCien	13,2 Mbps	100	1,5 Mbps (T1)

EXPORT

Per-testcase bandwidth consumption (per user)

Testcase	Avg BandwidthConsumed	Simulated User Network Speed	Size (bytes)	Estimated Duration (MM:SS)	Repeat Delay (MM:SS)
ZendCien	132,3 kbps	1,5 Mbps (T1)	392,4 KB	24	1

EXPORT

- **Prueba 2.3 mil transacciones**

Minimum Bandwidth Requirement (estimated)

Testcase	Avg BandwidthConsumed	Users	Simulated User Network Speed
ZendMil	120,9 Mbps	1.000	1,5 Mbps (T1)

EXPORT

Per-testcase bandwidth consumption (per user)

Testcase	Avg BandwidthConsumed	Simulated User Network Speed	Size (bytes)	Estimated Duration (MM:SS)	Repeat Delay (MM:SS)
ZendMil	120,9 kbps	1,5 Mbps (T1)	603,0 KB	40	1

EXPORT

- **Prueba 2.4 diez mil transacciones**

Minimum Bandwidth Requirement (estimated)

Testcase	Avg BandwidthConsumed	Users	Simulated User Network Speed
ZendDiezMil	992,1 Mbps	10.000	1,5 Mbps (T1)

EXPORT

Per-testcase bandwidth consumption (per user)

Testcase	Avg BandwidthConsumed	Simulated User Network Speed	Size (bytes)	Estimated Duration (MM:SS)	Repeat Delay (MM:SS)
ZendDiezMil	99,2 kbps	1,5 Mbps (T1)	2,7 MB	3:50	1

EXPORT

- **Indicador 3: Carga Transaccional de Alta**

  - **Prueba 3.1 La prueba de una transacción**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
liga1a.jpg	2.865	323,0 KB	
MIS ALBUMES - Zend Framework Albums [1]	.651	38,9 KB	
Add new album - Zend Framework Albums	.338	3,1 KB	
MIS ALBUMES - Zend Framework Albums [2]	.519	4,2 KB	

EXPORT

Summary

Summary information about the testcase.

Total Duration	18.95
Pages	4
URLs	15

  - **Prueba 3.2 cien transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
liga1a.jpg	2.850	323,0 KB	
MIS ALBUMES - Zend Framework Albums [1]	.546	38,9 KB	
Add new album - Zend Framework Albums	.201	3,1 KB	
MIS ALBUMES - Zend Framework Albums [2]	2.129	27,4 KB	

EXPORT

Summary

Summary information about the testcase.

Total Duration	19.67
Pages	4
URLs	15

- **Prueba 3.3 mil transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
liga1a.jpg	2.649	323,0 KB	
MIS ALBUMES - Zend Framework Albums [1]	.549	38,9 KB	
Add new album - Zend Framework Albums	.175	3,1 KB	
MIS ALBUMES - Zend Framework Albums [2]	19.388	238,1 KB	Page Duration <= 4000ms

EXPORT

Summary

Summary information about the testcase.

Total Duration	35.13
Pages	4
URLs	15

- **Prueba 3.4 diez mil transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
liga1a.jpg	2.866	323,0 KB	
MIS ALBUMES - Zend Framework Albums [1]	.549	38,9 KB	
Add new album - Zend Framework Albums	.219	3,1 KB	
MIS ALBUMES - Zend Framework Albums [2]	3:09.286	2,4 MB	Page Duration <= 4000ms

EXPORT

Summary

Summary information about the testcase.

Total Duration	3:32.99
Pages	4
URLs	15

- **Indicador 4: Carga Transaccional de Baja**

  - **Prueba 4.1 una transacción**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
<webpage> [1]	.471	4,3 KB	
MIS ALBUMES - Zend Framework Albums [1]	.140	3,3 KB	
Web Performance Analyzer™: ERROR [1]	.000	0 B	
Add new album - Zend Framework Albums	.683	3,2 KB	
<webpage> [2]	.000	0 B	
<webpage> [3]	5.115	3,3 KB	Page Duration <= 4000ms
<webpage> [4]	2.861	0 B	
MIS ALBUMES - Zend Framework Albums [2]	.453	4,5 KB	
(srchdetect12.predictad.com)	.000	0 B	
Delete album - Zend Framework Albums	.166	2,7 KB	
(srchdetect16.predictad.com)	1.748	0 B	
MIS ALBUMES - Zend Framework Albums [3]	.377	4,3 KB	
Web Performance Analyzer™: ERROR [2]	.000	0 B	

EXPORT

Summary

Summary information about the testcase.

Total Duration	18.51
Pages	13
URLs	17

- **Prueba 4.2 cien transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
<webpage>	.653	3,2 KB	
MIS ALBUMES - Zend Framework Albums [1]	.189	3,3 KB	
(srchdetect12.predictad.com)	.000	0 B	
Add new album - Zend Framework Albums	1.244	6,5 KB	
MIS ALBUMES - Zend Framework Albums [2]	2.210	27,8 KB	
Web Performance Analyzer™: ERROR	4.933	0 B	Page Duration <= 4000ms
MIS ALBUMES - Zend Framework Albums [3]	2.684	4,1 KB	

EXPORT

Summary

Summary information about the testcase.

Total Duration	24.34
Pages	7
URLs	10

- **Prueba 4.3 mil transacciones**

Summary

Summary information about the testcase.

Total Duration	1:08.68
Pages	11
URLs	15

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
<webpage> [1]	.740	4,3 KB	
MIS ALBUMES - Zend Framework Albums [1]	.157	3,3 KB	
<webpage> [2]	.000	0 B	
<webpage> [3]	.000	0 B	
(srchdetect12.predictad.com) [1]	.000	0 B	
<webpage> [4]	.000	0 B	
Add new album - Zend Framework Albums	3.833	6,5 KB	
MIS ALBUMES - Zend Framework Albums [2]	19.757	242,3 KB	Page Duration <= 4000ms
(srchdetect12.predictad.com) [2]	26.135	0 B	Page Duration <= 4000ms
MIS ALBUMES - Zend Framework Albums [3]	20.036	4,1 KB	Page Duration <= 4000ms
Web Performance Analyzer™: ERROR	.000	0 B	

EXPORT

- **Prueba 4.4 diez mil transacciones**

Page Performance Goals

The goals section shows the state of performance goals configured for pages in the testcase.

Page	Duration	Size	Failed Goals (as recorded)
<webpage> [1]	.529	4,3 KB	
MIS ALBUMES - Zend Framework Albums [1]	.203	3,3 KB	
(srchdetect12.predictad.com)	.000	0 B	
Add new album - Zend Framework Albums	1.239	6,5 KB	
<webpage> [2]	.000	0 B	
<webpage> [3]	.000	0 B	
<webpage> [4]	3:14.639	0 B	Page Duration <= 4000ms
MIS ALBUMES - Zend Framework Albums [2]	3:14.834	2,3 MB	Page Duration <= 4000ms
Web Performance Analyzer™: ERROR	.000	0 B	
MIS ALBUMES - Zend Framework Albums [3]	3:06.214	4,1 KB	Page Duration <= 4000ms

EXPORT

Summary

Summary information about the testcase.

Total Duration	6:47.70
Pages	10
URLs	14

- **Indicador 5: Integridad**

  - **Prueba 5.1 una transacción**

Page Analysis: MIS ALBUMES - Zend Framework Albums [2]

Testcase	ZendAdd (1)
Page URL	http://localhost/Zf-libros/public/
Baseline Size	4,2 KB
URLs	2
Baseline Duration	.519
Performance Goal	4.000
Attempts	43
Completions	43
Successes	43
Failures	0
Average Duration	.395
Total Duration	17.004

  - **Prueba 5.2 cien transacciones**

Page Analysis: MIS ALBUMES - Zend Framework Albums [2]

Testcase	ZendCien (1)
Page URL	http://localhost/Zf-libros/public/
Baseline Size	27,4 KB
URLs	2
Baseline Duration	2.129
Performance Goal	4.000
Attempts	44
Completions	44
Successes	44
Failures	0
Average Duration	4.006
Total Duration	2:56.305

- **Prueba 5.3 mil transacciones**

Page Analysis: MIS ALBUMES - Zend Framework Albums [2]

Testcase	ZendMil (1)
Page URL	http://localhost/Zf-libros/public/
Baseline Size	238,1 KB
URLs	2
Baseline Duration	19.388
Performance Goal	4.000
Attempts	19
Completions	19
Successes	19
Failures	0
Average Duration	46.032
Total Duration	14:34.619

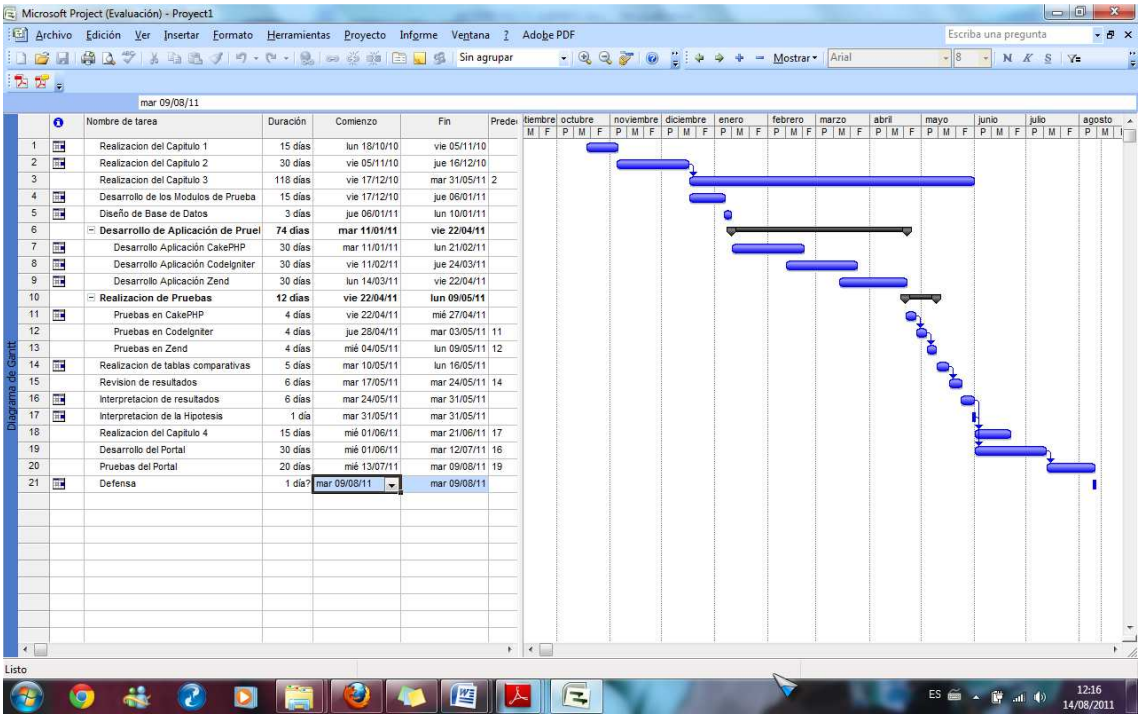
- **Prueba 5.4 diez mil transacciones**

Page Analysis: MIS ALBUMES - Zend Framework Albums [2]

Testcase	ZendDiezMil (1)
Page URL	http://localhost/Zf-libros/public/
Baseline Size	2,4 MB
URLs	2
Baseline Duration	3:09.286
Performance Goal	4.000
Attempts	10
Completions	10
Successes	0
Failures	10
Average Duration	4:21.707
Total Duration	43:37.071

ANEXO 3

Diagrama de Actividades



ANEXO 4

MANUAL DE USUARIO

PORTAL ACADEMIA LINUX ESPOCH

A. Introducción

El manual de usuario del portal web de la academia Linux ESPOCH le ofrece toda la información necesaria para que pueda operar con facilidad. Su estructura le permitirá identificar sencillamente cuáles son las operaciones que puede realizar de forma tal de aprovechar al máximo su funcionalidad.

Asimismo aconsejamos visitar regularmente la página WEB de la academia (http://www.linuxespoch.com) donde puede encontrar información actualizada que no se encuentra en el presente manual, puesto que en función de las consultas recibidas se efectuaran las correspondientes aclaraciones.

B. Requisitos mínimos para la Instalación

Los requerimientos mínimos que necesitamos para su correcto funcionamiento del portal web son muy importantes, por lo que sugeriremos el tipo de plataforma, librerías que deben estar instaladas previamente Tanto como al servidor y el cliente

- ❖ ***Requerimientos de Infraestructura.***

Hablaremos del Hardware que necesitamos como mínimo:

Servidor:

- Procesador: PIV o Superior
- Memoria: 512 MB o Superior
- CD-ROM: 52X
- Espacio Disco Duro: 2 GB o Superior
- Tarjeta de Red:10/100/1000

Cliente:

- Procesador: PII o Superior
- Memoria: 256 MB o Superior
- CD-ROM: 52X
- Espacio Disco Duro: 2 GB o Superior
- Tarjeta de Red:10/100/1000

- ❖ ***Requerimientos de Plataforma.***

Hablaremos del Software que necesitamos:

Servidor:

- Apache 2.2.13 o Superior
- PHP 5.2 o Superior
- MySQL 5.0 o Superior
- CodeIgniter 1.7.2
- NetBeans 6.9.1
- Conexión a Internet

Cliente:

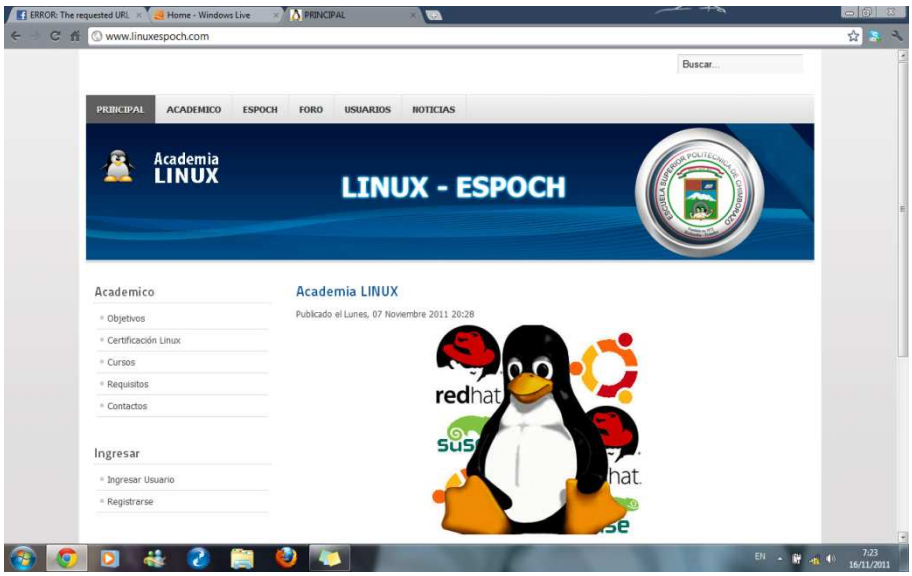
- Navegador IExplorer, Firefox, Chrome u otro
- Conexión a Internet
- Se recomienda para una óptima visualización, resolución de pantalla de 800x600 píxeles o superior.

C. Operatoria general del sistema

Aquí se detallara la forma correcta del funcionamiento del portal, como la administración en la parte de los usuarios:

❖ Ingreso al Sistema

En esta etapa se ingresa a la página principal del sistema, en donde se selecciona el perfil del operador.Abrir su navegador de internet, y escribir la siguiente dirección WEB:http://www.linuxespoch.com e ingresar según el vínculo que se presenta en la página WEB.



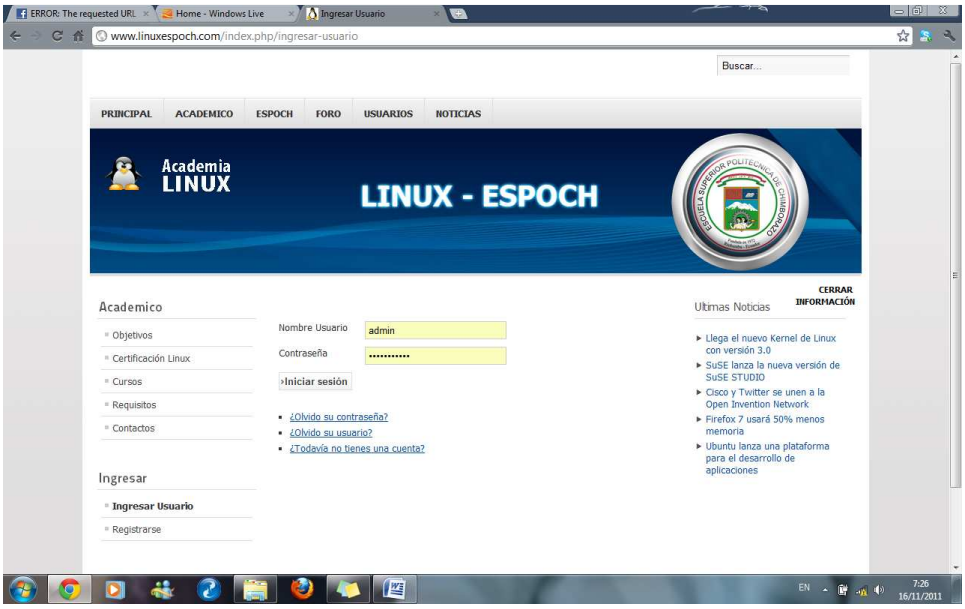
❖ Preinscripción al Sistema

En esta etapa se cargan los datos del operador para realizar la preinscripción por única vez al sistema y posterior autorización.

Complete todos los datos solicitados en el formulario.Tenga en cuenta algunos servidores de correo electrónico pueden llegar a tardar algún tiempo en recibir los emails, por lo que se aconseja esperar desde 24hs hasta 72hs por la confirmación de la preinscripción.

❖ Ingreso al Sistema para Operadores Registrados

Una vez que recibió el correo electrónico confirmando su preinscripción al sistema, junto con su cuenta y contraseña asignadas, puede ingresar al sistema.

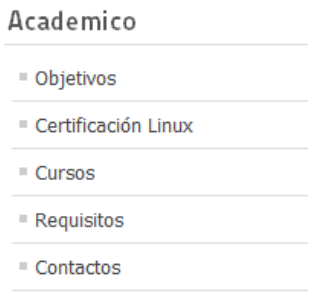


❖ **Menú de los usuarios**

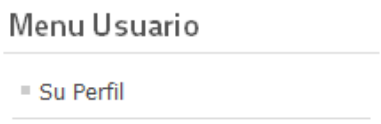
Las opciones del menú son:

- Login.-Aquí se puede es coger las opciones registrar o loguearse

- Menú Académico.- En este Menú se presenta Descripción de la academia con sus diferentes curso, Requisitos que es necesario para pertenecer, Metodología a emplear, Descarga de Información que existe en el portal, Preguntas Frecuentes



- Menú Usuario.- Un simple usuario podrá ver solo su perfil.



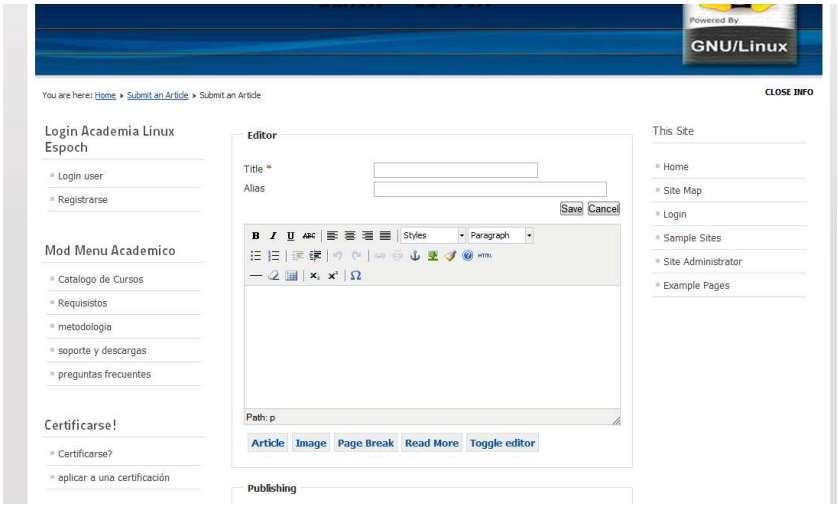
- Menú Subir Archivos.- Todos los usuarios registrados podrán subir archivos

- Menú Administrador.- Además de lo que tiene el usuario de ver perfil podrá agregar una publicación al portal y la administración de enlaces.

- Menú.-PRINCIPAL se mostrarán todas las noticias que se publiquen en el portal y se almacenaran para tener un registro de las misma, ACADEMICO nos da una información exacta del programa, ESPOCH un link directo al portal de la institución, FORO sector donde los usuarios pueden escribir sus inquietudes. USUARIOS donde podemos loguearnos, NOTICIAS son donde veremos publicaciones periódicas de temas relaciones con Linux.

❖ Ingresar, Consultar y Actualizar la información de las noticias publicadas.

Ingresaremos por Subir Articulo en el menú del administrador



- **Salir del Sistema**

Sale del sistema