UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

TEMA

ESTUDIO COMPARATIVO DE LOS FRAMEWORKS RUBY ON RAILS Y DJANGO PARA LA IMPLEMENTACIÓN DE UN SISTEMA INFORMÁTICO DE CONTROL Y ADMINISTRACIÓN DE NETWORK MARKETING.

AUTOR: Rubén Ignacio Guerrero Benalcázar

DIRECTOR: Ing. Omar Lara

Ibarra-Ecuador

2016

Ibarra, 12 de Julio de 2016

CERTIFICACIÓN

Certifico que la Tesis previa a la obtención del título de Ingeniería en Sistemas Computacionales con el tema "ESTUDIO COMPARATIVO DE LOS FRAMEWROKS RUBY ON RAIL Y DJANGO" con el aplicativo: "IMPLEMENTACION DE UN SISTEMA INFORMATICO DE CONTROL Y ADMINISTRACION DE NETWORK MARKETING.", ha sido realizada en su totalidad por la Sr. Rubén Ignacio Guerrero Benalcázar con C.C. 0401829767, bajo mi supervisión para lo cual firmo en constancia.

Atentamente,

Ing. Lenin Omar Lara

DIRECTOR DE TESIS

UNIVERSIDAD TÉCNICA DEL NORTE
BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

La Universidad Técnica del Norte dentro del proyecto Repositorio Digital Institucional, determinó la necesidad de disponer de textos completos en formato digital con la finalidad de apoyar los procesos de investigación, docencia y extensión de la Universidad.

Por medio del presente documento dejo sentada mi voluntad de participar en este proyecto, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	040182976 – 7		
APELLIDOS Y NOMBRES:	GUERRERO BENALCÁZAR RUBÉN IGNACIO		
DIRECCIÓN:	MANTA Y PELIKANO 7 – 05		
EMAIL:	rubenignacio1522@hotmail.com		
TELÉFONO FIJO:	(062) 609-391	**TELÉFONO MÓVIL:**	0996244233

DATOS DE LA OBRA	
TÍTULO:	"ESTUDIO COMPARATIVO DE LOS FRAMEWORKS RUBY ON RAILS Y DJANGO"
AUTOR :	GUERRERO BENALCÁZAR RUBÉN IGNACIO
FECHA: AAAAMMDD	2016 – 02 – 25
PROGRAMA:	☑ **PREGRADO** ☐ **POSGRADO**
TITULO POR EL QUE OPTA:	**INGENIERÍA EN CIENCIAS COMPUTACIONALES**
ASESOR /DIRECTOR:	**ING. OMAR LARA**

2. AUTORIZACIÓN DE USO A FAVOR DE LA UNIVERSIDAD

Yo, RUBÉN IGNACIO GUERRERO BENALCÁZAR, con cédula de identidad Nro. 040182976 - 7, en calidad de autor y titular de los derechos patrimoniales de la obra o trabajo de grado descrito anteriormente, hago entrega del ejemplar respectivo en formato digital y autorizo a la Universidad Técnica del Norte, la publicación de la obra en el Repositorio Digital Institucional y uso del archivo digital en la Biblioteca de la Universidad con fines académicos, para ampliar la disponibilidad del material y como apoyo a la educación, investigación y extensión; en concordancia con la Ley de Educación Superior Artículo 144.

3. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 22 días del mes de Julio de 2016.

EL AUTOR:

(Firma) .....

Nombre: Rubén Ignacio Guerrero Benalcázar

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE INVESTIGACIÓN

A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

Yo, RUBÉN IGNACIO GUERRERO BENALCÁZAR, con cédula de identidad Nro. 040182976-7, manifiesto mi voluntad de ceder a la Universidad Técnica del Norte los derechos patrimoniales consagrados en la ley de propiedad intelectual del Ecuador, artículos 4, 5 y 6, en calidad de autor del trabajo de grado denominado: "ESTUDIO COMPARATIVO DE LOS FRAMEWROKS RUBY ON RAIL Y DJANGO" con el aplicativo: "IMPLEMENTACION DE UN SISTEMA INFORMATICO DE CONTROL Y ADMINISTRACION DE NETWORK MARKETING.", que ha sido desarrollado para optar por el título de Ingeniería en Sistemas Computacionales en la Universidad Técnica del Norte, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente. En concordancia suscribo este documento en el documento que hago entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Técnica del Norte.

Ibarra, a los 22 días del mes de Julio de 2016

(Firma).....

Nombre: Rubén Ignacio Guerrero Benalcázar

Cédula: 040182976-7

DEDICATORIA

Este trabajo está dedicado a cada una de las personas que día a día lucharon por darme un futuro mejor mediante la educación, Mi Familia. Ellos que fueron los que día a día alentaban mi porvenir.

A mis padres que con su trabajo arduo lograron formarme de la mejor manera, a ellos que con su sacrificio, con su trabajo, con su esfuerzo y cariño alentaron a este servidor para que siga los estudios en la Universidad.

A mi esposa Janeth Enríquez que es la madre de mis hijas y la persona con la que luchamos juntos para salir adelante, además es la que siempre inculca los valores de responsabilidad y amor al hogar.

AGRADECIMIENTO

Agradezco infinitamente a Dios por regalarme el don de vivir y por todas las bendiciones en las que él se ha manifestado conmigo.

Un profundo agradecimiento a mi Familia que ha hecho hasta lo imposible por estar a mi lado apoyándome en el campo de formación profesional y moral.

A mi esposa por la paciencia, el amor, el cariño y el respeto que día con día me ha brindado.

A mis hijas que son mi primer impulso diario de querer ser el mejor en toda actividad que realice.

A la Universidad Técnica del Norte por abrirme las puertas para realizar mis estudios superiores en sus acogedoras aulas.

A todos mis maestros de la Universidad quienes compartieron conmigo sus conocimientos.

A mi director de Tesis Ing. Omar Lara por brindarme su apoyo incondicional para lograr esta meta.

ÍNDICE DE CONTENIDO

ÍNDICE DE GRÁFICOS

ÍNDICE DE TABLAS

RESUMEN

La presente Tesis tiene como objetivo, comparar las herramientas Ruby On Rails y Django, para obtener un framework ganador, con el cual se realizara una aplicación de network marketing.

Los capítulos por los que está conformada la presente Tesis se describen a continuación.

El capítulo I contiene los antecedentes y problemas que tiene la empresa o entidad en este momento y cuál será la solución ante este hecho.

El capítulo II está conformado por la investigación de cada una de las herramientas y sus características las cuales son estudiadas de acuerdo a lo que se crea conveniente, en este capítulo abarcará las arquitecturas de las herramientas entre otros aspectos, como características o que temas son los precisos para realizar una posterior comparación entre los frameworks.

El capítulo III contiene la valoración de los parámetros con los cuales se realizará una comparación entre herramientas para determinar un ganador, de acuerdo a la escala Likert con la cual se trabajó en este capítulo.

El capítulo IV contiene la implementación del sistema que dará solución a la administración de network marketing. Este capítulo contiene el desarrollo de las fases de la metodología con la que se trabajó para llevar a cabo el trabajo de diseñar el aplicativo.

Por último el capítulo V contiene las conclusiones y recomendaciones también el análisis de impacto que se tiene con este trabajo realizado.

SUMMARY

This thesis aims to compare the Ruby On Rails and Django tools to obtain a winning framework with which an application of network marketing is carried out.

Chapters which is shaped by this thesis are described below.

Chapter I contains the background and problems of the company or entity at this time and what will be the solution to this fact.

Chapter II consists of research of each of the tools and features which are studied according to what it sees fit, this chapter will cover architectures tools among other aspects, such as features or issues are accurate for a subsequent comparison between those named above frameworks.

Chapter III contains the assessment of the parameters with which a comparison tools to determine a winner, according to the Likert scale with which we worked in this chapter will take place.

Chapter IV contains the implementation of the system that will solve marketing management network. This chapter contains the development phase of the methodology with which we worked to carry out the work of designing the application.

Finally the chapter V contains the conclusions and recommendations also further impact analysis you have with this work.

CAPÍTULO 1

1. INTRODUCCIÓN

El presente capítulo detalla y a la vez informa el ámbito en el cual se va a desarrollar el aplicativo, el problema que existe actualmente, los antecedentes, objetivos y el alcance o solución que brindará dicho proyecto.

1.1 Problema

1.1.1 Antecedente

La existencia de herramientas y plataformas para desarrollo de aplicaciones web han evolucionado tanto que los usuarios se han quedado en las más conocidas, sin darse cuenta de las ventajas que ofrecen las nuevas y sofisticadas plataformas de desarrollo web.

1.1.2 Situación actual

Actualmente la empresa lleva los trabajos de administración de cada empresario y cliente a papel, teniendo como debilidad y riesgo la manipulación de la información, esta situación se viene dando por la falta de un sistema web que permita administrar y organizar la información de manera correcta.

1.1.3 Planteamiento del problema

No existe un sistema web que permita administrar de manera organizada y controlada la información de empresarios y clientes en la empresa.

1.1.4 Prospectiva

El estado en el que se encuentra la unidad de administración de la empresa impide que se lleve un control ordenado de los empresarios y sus clientes, por lo tanto a futuro se solucionará este problema con el aplicativo a desarrollarse en el cual se visualizará los empresarios y los clientes de manera clara y ordenada.

1.2 Objetivos

1.2.1 Objetivo general

Establecer un estudio comparativo entre los frameworks Ruby On Rails y Django para implementar un sistema de control y administración de Network Marketing.

1.2.2 Objetivos específicos

- ✓ Realizar un análisis actual de la administración de la empresa
- ✓ Determinar la métricas de comparación
- ✓ Diagnosticar la herramienta que se utilizará para la implementación del sistema.
- ✓ Emplear la herramienta seleccionada para proceder a realizar el sistema.

1.3 Alcance

Se compararán las siguientes herramientas de desarrollo de software Ruby On Rails (Rails, 2013) *y Django* (Apress, 2009) en base a parámetros y criterios de evaluación para así determinar la mejor herramienta que cubra todas las necesidades del usuario.

La implementación del Sistema beneficiará a la empresa Omnilife y a todos sus empresarios, ya que serán administradores de sus redes independientemente.

Los conocimientos adquiridos se aplicarán en el desarrollo de un sistema de administración y control orientado al Network Marketing (Robert Kiyosaki, 2012).

Este sistema tendrá la capacidad de:

- Tener un control sobre los clientes y valorar sus ganancias.
- Verificar las inscripciones que cada uno de los empresarios realice.
- Verificar el nivel en el cual se encuentran ubicados.

El sistema tendrá los siguientes módulos:

- **Administración de usuarios:** conllevará la creación, administración de cargos o roles y usuarios del sistema.
- **Catálogo de proyectos y actividades:** llevará un registro de las actividades y proyectos a ejecutarse en la empresa.
- **Bitácora:** se registrará los clientes que cada usuario inscriba bajo su código o su membresía.
- **Reportes**

Para la realización del sistema informático se utilizará la metodología en cascada que está basada en procesos secuenciales o lineales, esencialmente como un modelo lineal de desarrollo de software. Este modelo fluye secuencialmente desde el punto inicial al punto final. (Aristega, 2010) (Shari Lawrence Pfleeger, 2002)

Gráfico 1: Arquitectura del sistema organizacional de la Empresa
Fuente: Propia

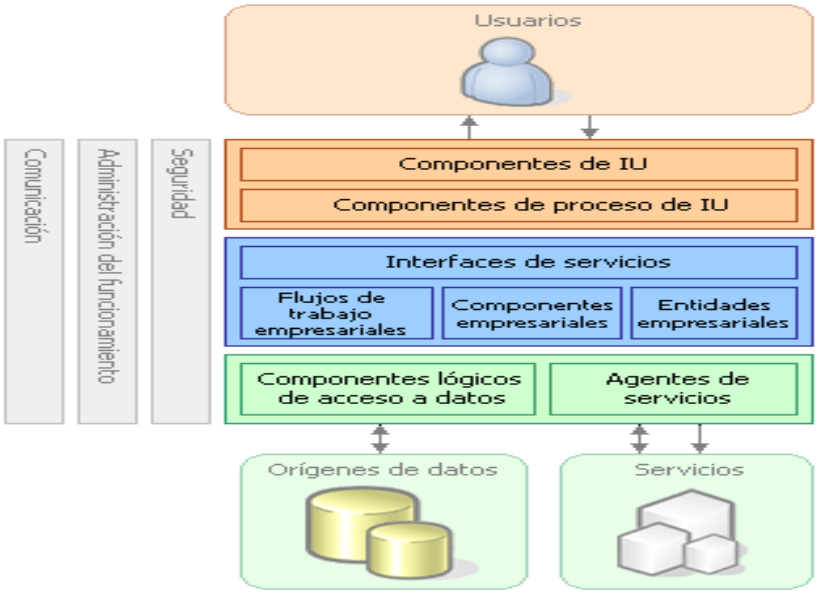


Gráfico 2: Arquitectura Técnica del Sistema.
Fuente: *Modelo n-capas*, recuperado de http://wiki-aplidaniel.wikispaces.com/file/view/n-capas.png/251847972/n-capas.png, 2015

1.4 Justificación

El avance tecnológico hoy en día ha hecho que las instituciones o empresas busquen la manera de automatizar sus campos o procesos para facilitar su desempeño. Con el estudio comparativo de las herramientas se busca facilitar la decisión de usar una herramienta para crear un sistema que permitirá agilizar los procesos de control y administración dentro del campo de Network Marketing (Cabrera, 2005).

El análisis de herramientas como proyecto de tesis, permitirá obtener una gran documentación misma que servirá como guía para docentes y estudiantes, además de profesionales en el ámbito de la informática, garantizando así un desarrollo de aplicaciones de excelente calidad.

Será un incentivo en los estudiantes a seguir realizando las debidas investigaciones de nuevas metodologías y herramientas tales como los frameworks para realizar de mejor manera el desarrollo de software.

CAPÍTULO 2

2. MARCO TEÓRICO

El presente capítulo contiene temas relacionados a la empresa Omnilife de la cual se dará un breve concepto de lo que es y qué negocio maneja o cuál es su actividad y a la vez contenidos de cada una de las herramientas o frameworks que se utilizará en los próximos capítulos para desarrollo del sistema aplicativo.

2.1 Empresa Omnilife

La empresa Omnilife o Grupo Omnilife como se le conoce, es una empresa que se encarga de realizar negocios de multinivel, relacionado esto con el network marketing o negocio del siglo XXI.

Gráfico 3: Logotipo de la Empresa Omnilife
Fuente: https://www.omnilife.com/boletin/images/22JUL15/boletin_220715_01_c.jpg

Esta empresa se encarga de elaborar productos multivitamínicos que son suplementos alimenticios que ayudan a mejorar las funciones del cuerpo humano, además de esto también produce productos de belleza. Su sede principal está ubicada en Guadalajara-México. Tiene más de 6000 empleados y más de 5.5 millones de empresarios independientes también conocidos como distribuidores que operan en más de 19 países.

El negocio Multinivel o Network marketing es la forma de comercializar ya sea productos o servicios mediante canales de distribución llamados redes. Este canal de distribución lo conforman personas las cuales invitan a otras a formar parte de la empresa y del negocio, por

esto los distribuidores o personas que invitan a otras reciben un porcentaje tanto de las ventas como también de los consumos o ventas que realicen las otras personas, esto es de manera eficaz y directa.

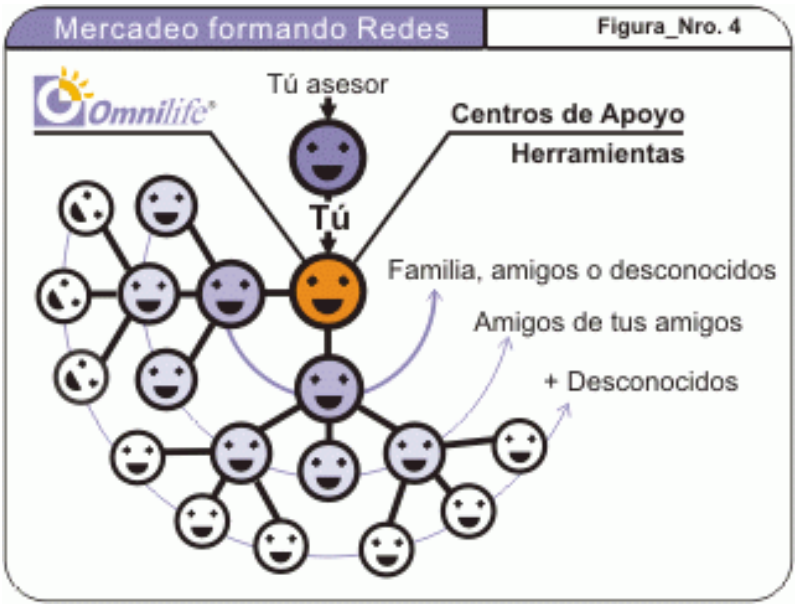


Gráfico 4: Modelo Multinivel

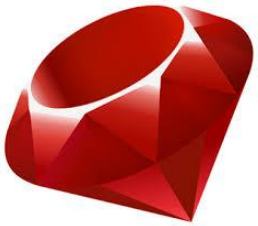
Fuente: http://2.bp.blogspot.com/-ilfUdbDPnl4/VQ3DezilnAI/AAAAAAAAAOw/96Chi89w2K0/s1600/mercadeo_redes.gif

2.2 Recopilación de información de las herramientas

A continuación se adentrará de manera breve en los temas relacionados con las herramientas o frameworks los cuales serán analizados y se sacará sus ventajas y desventajas de cada uno de ellos.

La información aquí obtenida servirá para verificar características de las herramientas Ruby on Rails y Django las cuales servirán para realizar el desarrollo de la solución ante el problema que se ha presentado y para el desarrollo del aplicativo.

2.3 Introducción a Ruby On Rails



Ruby es un lenguaje que creó el japonés Yukihiro Matz Matsumoto el 24 de febrero del año de 1993 y lo presentó públicamente en el año 1995.

Su nombre Ruby se le dió en son de broma aludiendo al lenguaje que en ese entonces era llamado Perl. Ruby On Rails es un lenguaje de programación creado por David Heinemier y lanzado al público en el año de 2004. Rails es un lenguaje se podría decir que es obstinado en decir que existen mejores cosas y poder lograrlas en cuanto a la programación se refiere, se ha puesto en las manos de programadores para facilitarle el trabajo ya que evita el tener que repetir código, ya que ésta es la filosofía de RoR (Ruby On Rails), Don't repeat Yourself.

El objetivo de Ruby on Rails, es mejorar la interactuación del usuario y el lenguaje de programación de tal manera que éste se divierta realizando un trabajo y no se frustre con un lenguaje complicado y difícil de entender, que por lo general ocurre cuando una persona o usuario busca una herramienta para realizar un programa o incluso aprender a programar.

El lenguaje se vuelve tan tedioso que las personas desisten en el intento. Pero con Ruby que lo que busca es evitar ese caos y mejorar en cierta forma el criterio de que la programación es divertida y se la puede realizar de mejores maneras, en el presente mismo.

Además se dice que hay que mejorar el ambiente del programador o usuario más que el de la máquina mismo, ya que el ser humano es el que ordena y el computador se encargar de realizar el trabajo.

2.4 Ruby On Rails

Rails es un framework de desarrollo de aplicaciones web escrito en el lenguaje Ruby. Está diseñado para hacer que las aplicaciones web de programación más fácil al hacer suposiciones acerca de lo que necesita cada desarrollador para empezar.

Permite escribir menos código durante el cumplimiento de tareas, más que muchos otros lenguajes y marcos. Experimentados desarrolladores Rails también informan que este framework hace que el desarrollo de aplicaciones web sea más divertida.[1]

Rails es un software obstinado. Esto hace que la suposición de que hay una "mejor" manera de hacer las cosas, se cumpla, y está diseñado para alentar esa manera - y en algunos casos para desalentar las alternativas. Si usted aprende "El Rails Way" es probable que descubra un enorme aumento en la productividad.

Si insiste en traer viejos hábitos de otros idiomas a su desarrollo Rails, y tratando de utilizar patrones que aprendió en otro lugar, usted puede tener una experiencia no tan agradable[2]. Ruby On Rails es por ende un innovador en cuanto al desarrollo en la programación.

La filosofía Rails incluye varios principios rectores:

DRY - "Do not Repeat Yourself" - sugiere que escribir el mismo código una y otra vez es un mal hábito.

[1] http://guides.rubyonrails.org/getting_started.html
[2] http://guides.rubyonrails.org/getting_started.html

Convención sobre configuración - significa que Rails hace suposiciones sobre lo que quiere hacer y cómo va a hacerlo, en lugar de exigir que especifique cada pequeña cosa a través de archivos de configuración sin fin. REST (Recurso de Ruby On Rails, que permite el uso de métodos para obtener información de las rutas, las cuales se pueden realizar de manera estática o manual), es el mejor patrón para aplicaciones web - la organización de su aplicación alrededor de los recursos y los verbos HTTP (Protocolo de Transferencia de Hipertexto) estándar es la manera más rápida de ir[3].

En unos sencillos ítems facilitaremos las características de Ruby On Rails.

- Es una tecnología útil, y se implementa de manera fácil y sencilla, con un nuevo modelo de programación evolucionando las ideas de un programador.
- Su planteamiento lo diferencia del resto de tecnologías de software libre.
- La rapidez de sus aplicaciones fueron las que lo llevaron al éxito.
- Su aprendizaje es mucho más sencillo que otros lenguajes de programación.
- Su objetivo es mejorar la relación entre el usuario y el lenguaje de tal manera que este se sienta atraído por la programación y no desista en su aprendizaje por las dificultades que representa aprender diferentes herramientas no tan amigables.

2.4.1 MVC en Ruby on Rails

Rails es fundamentalmente una implementación de un patrón de diseño de software conocido como MVC[4].

Este divide un sistema en tres componentes:

- El modelo de datos.

[3] http://guides.rubyonrails.org/getting_started.html
[4] Modelo Vista Controlador

- La vista o interfaz gráfica.
- El controlador que contiene la lógica del negocio.

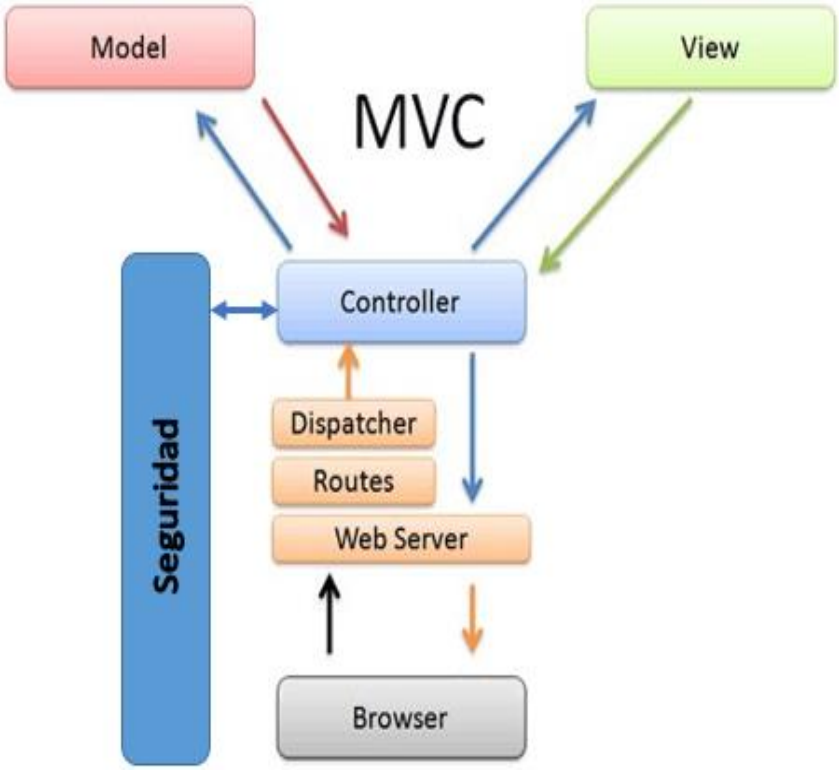


Gráfico 5: Arquitectura MVC de Ruby On Rails
Fuente: propia

2.4.2 Módulos en Rail

Ruby on Rails también tiene otros módulos para realizar algunas operaciones, las cuales son encargadas de mejorar el desarrollo en cuanto a la programación se refiere, todo esto es una ayuda para que el desarrollador de software no se hostigue como pasaba con herramientas antiguas que resultaba incómodo implementar el código sin alguna ayuda, o peor aún existían casos en los cuales no contaban por lo menos una documentación en la cual el usuario o desarrollador tome como base o guía y avance de manera más rápida y eficiente.

En la siguiente imagen se muestra gráficamente a todos y a continuación se describe brevemente a cada uno de ellos.

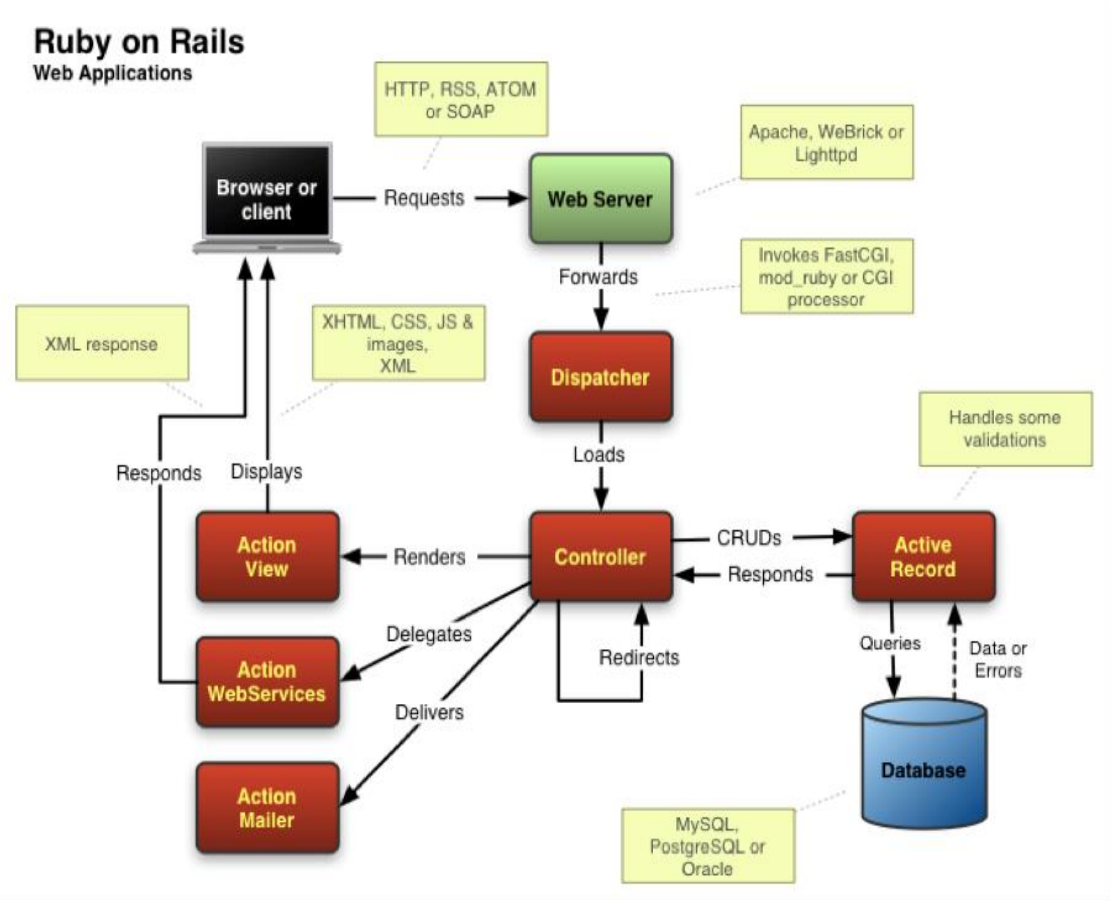


Gráfico 6: Módulos de Ruby On Rails en la Arquitectura
Fuente: http://binaryhash.com/assets/ruby-on-rails-88be263b5798172302f17d4862778f46.png

2.4.2.1 Action Pack

Action Pack es una herramienta que realiza el manejo y respuestas de las peticiones Web. Otra función es la de proporcionar un mecanismo para el enrutamiento en cuanto a la solicitud de mapeo o URLs a acciones.

El módulo de Action Pack proporciona capas la cuales son la de controlador y vista de los patrones MVC (Modelo Vista Controlador). Estas acciones o actividades definen la capa controlador y en otra instancia esas acciones harán una vista que se mostrará en el navegador

web la parte VC (Vista, Controlador) de MVC (Modelo Vista Controlador). Action Pack dispone de los módulos: Action Controller, Action Dispatch y Action view.

2.4.2.2 Action Controller

Action controller es el componente que gestiona los controladores en una aplicación Rails. Action Controller framework procesa todas las solicitudes que son entrantes a una aplicación Rails. Los servicios que presta Action Controller incluyen a su vez la gestión de sesiones, la representación plantilla y además la gestión a redirigir.

En resumen Action controller, proporciona una clase que permite aplicar filtros y acciones para el manejo de peticiones. También estará pendiente de la conexión o comunicación a la base de datos, como también con las acciones de CRUD (Create, Read, Update, Delete), cuando sea necesario.

2.4.2.3 Action View

Action view es responsable de compilar las respuestas ya que gestiona vistas de la aplicación Rails, ésta es llamada por el controlador de acción (action controller). Puede crear HTML[5] y XML[6] de salida automática o por defecto.

Además Action View gestiona diseños o plantillas de presentación, incluyendo también las plantillas que son anidadas o parciales, existen tres plantillas esquemas de Rails las cuales son: RHTML, RXML y RJS. El RHTML es el encargado de generar puntos de vista HTML a los usuarios con código de Ruby incrustado en HTML. El RXML es utilizado para construir

[5] HyperText Markup Language (lenguaje de marcas de hipertexto)
[6] Lenguaje de Marcas Extensible

documentos en forma XML (eXtensive Markup Languaje), usando Ruby y RJS, permite también la creación dinámica de código JavaScript que es muy útil en Ruby para implementar la funcionalidad AJAX.

2.4.2.4 Action Dispatch

Action Dispatch maneja el enrutamiento de solicitudes web, de acuerdo a lo definido por el usuario, esto quiere decir que es el encargado de analizar la información y los despacha ya sea para la misma aplicación u otra que sea Rack. Además Action Dispatch decodifica los parámetros POST, PUT y también realiza el manejo en cuanto a la lógica de almacenamiento en memoria caché. (HTTP, cookies y sesiones).

2.4.2.5 Active Record

Active Record es un patrón arquitectónico, que facilita el manejo o administración de los datos en las bases de datos relacionales, a través de objetos. El módulo Active Record en Ruby On Rails ofrece un mapeo eficiente referente a objeto-relacional de clases. Éste módulo se basa en la capa que conecta a las tablas de las bases de datos. Active Records se basa en una gran medida en las convenciones sobre cómo todas y cada una de las clases que se utiliza deben ser nombradas (Tablas, claves foráneas y claves primarias).

El módulo Active Record se utiliza para crear clases del modelo, las cuales contienen en sí lo que es la lógica del negocio, se encarga del manejo de las validaciones y las relaciones, así como también de los maps de forma automática a cada tabla y encapsula el acceso a datos, proporciona captadores y definidores, devoluciones de llamada y también es compatible con varias bases de datos que existen hoy por hoy.

2.4.2.6 Action Mailer

Action Mailer es una herramienta muy útil para la construcción de servicios de correo ya que éste módulo se encarga de prestar servicios de correo electrónico. Action Mailer se encuelve alrededor del Action Controler. Puede usarse esta opción para recibir y administrar el correo electrónico entrante y enviar texto o mensajes de correo electrónico de partes muy complejas basadas en ciertas plantillas flexibles de la misma manera que lo realiza Action View para hacer páginas Web. También tiene tareas comunes incorporadas, como el envío de contraseñas olvidadas, mensajes de bienvenida y el cumplimiento de necesidad de cualquier otra comunicación escrita.

2.4.2.7 Active Model

Active Model proporciona una interfaz clara y definida entre los servicios de la gema Action Pack y las gemas de Asignación Relación de objetos tales como Active Record. Active Model permite al software Rails que utilice otros marcos como ORM (Mapeo de Objeto Relacional) en lugar de Active Record si tuviera la necesidad la aplicación.

2.4.2.8 Active Resources

Active Resources proporciona la infraestructura necesaria para poder crear de manera sencilla y a la vez oportuna recursos **REST.**

Además proporciona un marco a la gestión de conexión entre cada uno de los objetos de negocio y servicios web RESTful. Implementa también una manera de asignar los recursos basados en la web a objetos locales con semántica CRUD (Create, Read, Update, Delete).

2.4.2.9 Active Support

Es un conjunto de clases de utilidad y es en sí la biblioteca de extensiones estándar que contiene el framework Ruby On Rails. Este conjunto es muy amplio en cuanto a herramientas se refiere debido a que incluye soporte para cadenas que son de múltiples bytes, internacionalización, zonas horarias, etc.

2.4.2.10 Railties

Railties es prácticamente el código del núcleo de Rails que es el que construye nuevas aplicaciones y colas en los distintos frameworks y plugins de Ruby on Rails juntos en los diferentes proyectos que se realice.

También se encarga de todo el proceso de bootstrapping, esto quiere decir que se encarga de la interfaz de línea de comandos y proporciona generadores de código de Rails. Rake in, uno de los comando de línea más usados para realizar las tareas de bases de datos, documentación, implementación, pruebas y limpiezas. Rails también incorpora cierto marco de pruebas de manera automática cuando se genera el código, proporciona lo que son llamadas las pruebas unitarias, pruebas funcionales para cada una de las vistas y los controladores, accesorios y datos de alimentación a la vez, utilizando YAML (Formato de serialización de datos legibles).

2.4.3 Arquitectura REST

REST significa Transferencia de Estado Representacional y es una alternativa a los servicios web. REST se basa prácticamente en el protocolo HTTP (protocolo de transferencia de hipertexto) para la realización de las operaciones CRUD (Crear, Leer, Actualizar, Borrar).

REST tiene la opción RESTful que son servicios Web, utilizados cuando los servicios Web son completamente sin estado o el ancho de banda es muy limitado, éste es muy útil para

dispositivos móviles debido a que no realiza sobrecarga como otros protocolos por nombrar a SOAP.

Es muy útil cuando los datos no se generan dinámicamente por lo que se almacena en la memoria caché y mejora el rendimiento y la comunicación entre el productor y el consumidor en este caso de servicios.

2.4.4 Ajax on Rail

Es la técnica utilizada para hacer uso de Javascript y XML (eXtensive Markup Languaje) para realizar procesos de un navegador web a un servidor web, que actúan de forma secundaria es decir sin abrir páginas web adicionales.

2.4.5 Soporte de Servidores Web

Ruby on Rails usa o utiliza los siguientes servidores web: Nginx, Mongrel, Apache, Lighttpd con FastCGI, o incluso llega a combinar algunos de estos para el desarrollo o prueba.

2.4.6 Arquitectura en tiempo de ejecución

Una visión en tiempo de ejecución de la arquitectura de un sistema se muestra a continuación, los componentes que existen y como hacen estos para comunicarse entre sí.

- **Connect and Component View**

Connect and Component View es la vista dinámica del sistema que presenta los componentes, las interfaces, conectores y sistemas. Esto proporciona una interfaz de representación compacta de cada una de las interfaces.

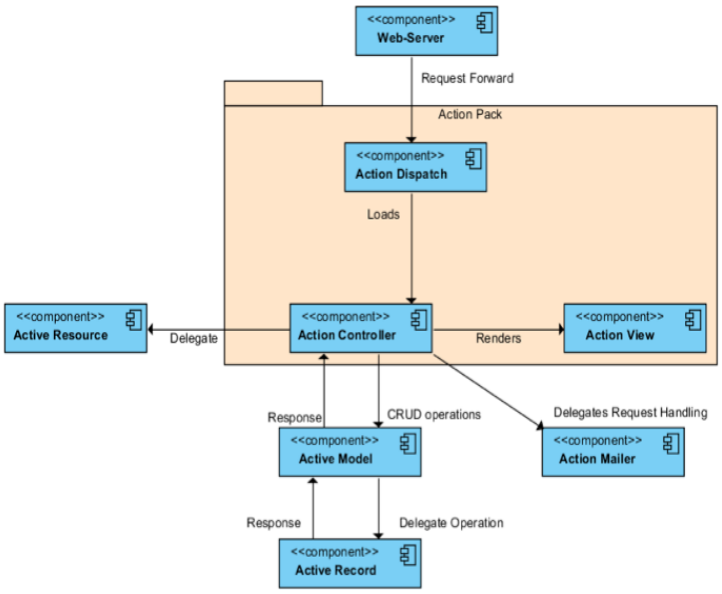


Gráfico 7: Vista Dinámica en tiempo de ejecución de Ruby On Rails
Fuente: http://adrianmejia.com/images/dynamic_view.png

- **Módulo View**

El módulo View muestra el código o la aplicación en forma de módulos y además presenta las diferentes interfaces y cada una de sus relaciones entre sí. El módulo View contiene: módulos, agregaciones, generalizaciones y dependencias. Cada uno de estos puntos son una forma adecuada se puede decir de mostrar una visión estática de la arquitectura.

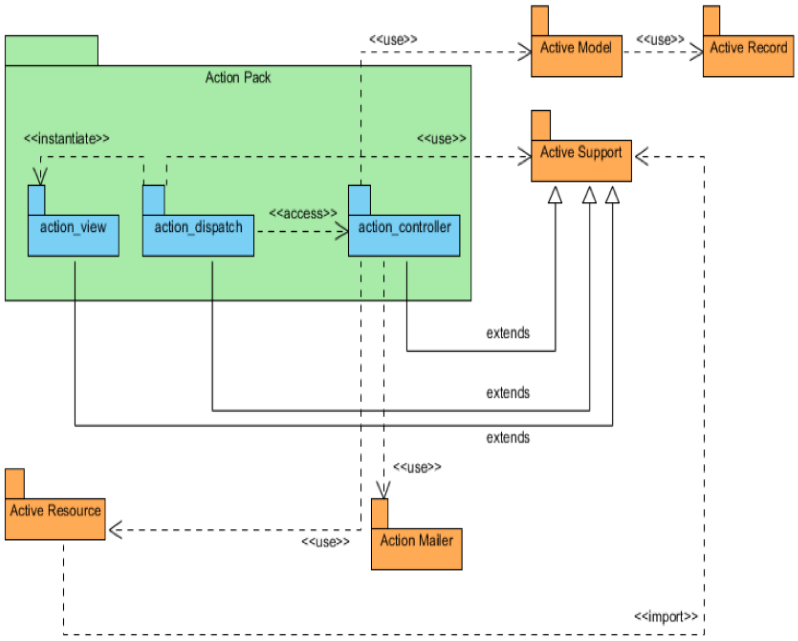


Gráfico 8: Vista Estática de Ruby On Rails
Fuente: http://adrianmejia.com/images/ror_static_view.png

2.4.7 Gemas

Una de las herramientas que usa Ruby son los plugins o códigos, que permiten tener nuevas funciones o funcionalidades como nuevos **create,** también cuenta con funciones predefinidas (como login para usuarios), una de las nuevas herramientas para el desarrollo son:

Haml es un template basado en Html pero sencilla y a la vez muy potente, y SASS (Hoja de estilo sintáctico impresionante) que es igual que Haml pero ésta en cambio se la usa para el caso de CSS.

2.4.8 Soporte a Base de Datos

La arquitectura de Ruby On Rails ayuda a realizar el uso de algunas bases de datos, pero se recomienda usar un SGBDR (Sistema Gestor de Base de Datos de Ruby) para almacenamiento de los datos. Esta plataforma Ruby On Rails, soporta la biblioteca SQLite que es por defecto.

El acceso que tiene a la base de datos es totalmente abstracto para el usuario (programador), esto quiere decir que es agnóstico a la base de datos. Rails realiza la gestión a la base de datos de forma automática, claro que se puede realizar consultas directamente en SQL. Nombramos algunos de los SGBDRs que soporta Rails, MySQL, PostgreSQL, SQLite, IBM DB2 y Oracle.

2.4.9 Scaffolding

Rails implementa también una técnica llamada scaffolding que genera una definición de un nuevo modelo y todo el código necesario para realizar cada una de las tareas CRUD (Crear, Leer, Actualizar, Borrar).También genera vistas o páginas necesarias para realizar operaciones desde un navegador y además una vista general que enlista a todos los objetos que se crearon a partir de ese modelo.

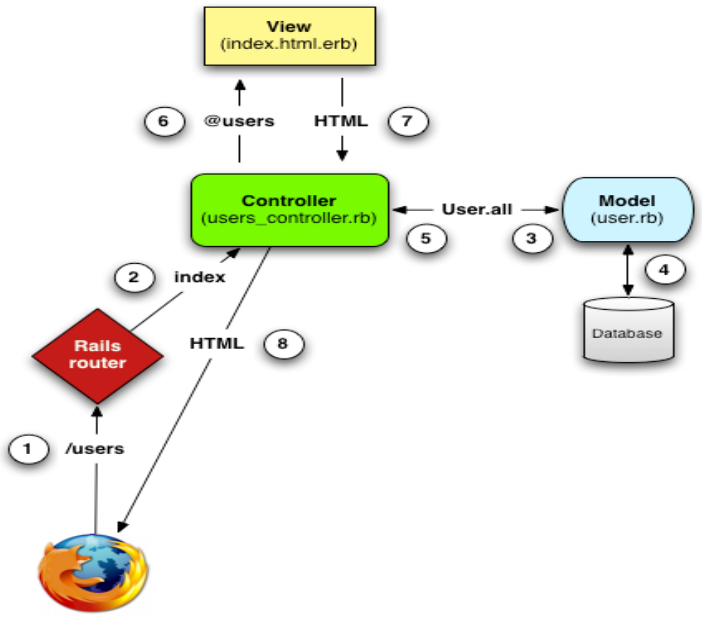


Gráfico 9: Modelo Detallado de MVC en Rails
Fuente: https://blog.chattyhive.com/wp-content/uploads/2014/01/mvc_detailed-full.png

2.4.10 Campo de desarrollo de Ruby On Rails

Existe una variedad en el campo de la programación y desarrollo para trabajar con este framework (RoR), tanto libres, de pago o gratuitas, hoy por hoy en el mundo del desarrollo Web.

La información de los campos donde se desarrolla este framework son muchos, pero vamos a recalcar algunos de ellos entre estos los más usados por desarrolladores y tomados también como ejemplo por los resultados positivos que ha dado Ruby on Rails en las herramientas de desarrollo, los principales son:

- **Aptana:** Multiplataforma. Fue un plugin del software llamado eclipse para edición y desarrollo web. Actualmente sigue existiendo como plugin, pero también existe de forma independiente. Cabe recalcar que hasta el momento Antana existe en versión 3 la cual es la versión estable.

- **Netbeans:** es uno de los que viene muy bien integrado con la herramienta JRuby pues es un programa de Oracle. Además Netbeans es uno de los más usados, también es libre y gratuito para todo usuario. Desde la versión de Netbeans 7.0, el soporte para Ruby y también para Ruby on Rails ya no se encuentra disponible en las versiones que son estándar del mismo.

- **TextMate:** En el ámbito de Rails es el más usado, este software es pagado, pero la potencia y el desarrollo que ofrece, favorece mucho a Rails. TextMate es solo para Mac, sería una limitación pero es un editor excelente y tiene una mayor aportación que otros editores de texto en programación.

- **Sublime Text:** fue creado en Phyton es un editor de texto y editor de código, desarrollado como una extensión de Vim originalmente, es en sí un editor fácil de manejar pero no tan potente, como lo desearía un programador experimentado.

2.4.11 Ruby On Rails y sus errores en la arquitectura

Rails se basa en el lenguaje Ruby ya por esta razón hereda la bondad y la debilidad de ese lenguaje, debido a que Ruby es un lenguaje dinámico de secuencias de comandos con una llamativa sintaxis ordenada y orientada al manejo de objetos. Pero debido a que es un lenguaje interpretado es mucho más lento que otros frameworks.

La velocidad no sería mucho el problema pero sí lo es cuando se trata de almacenamiento y se necesita interactuar con millones de usuarios esta empieza a perder velocidad y degradarse. Esto quiere decir que el lenguaje no es el adecuado para aplicaciones que tengan alta concurrencia.

Todo esto se debe a que fue diseñado para ser elegante, sencillo y de desarrollo rápido. Ruby al igual que otros frameworks tiene problemas para trabajar con procesos largos. Por otro lado existe también la debilidad de recolector de basura, esto quiere decir, que cada proceso requiere mucha más memoria.

En fin son ajustes del Lenguaje que se deberá tomar en cuenta para dar una pronta mejoría en las diferentes versiones que se vendrán más adelante, de esta manera se convertirá en un software poderoso, que ayudará al desarrollo de la programación.

2.5 Introducción a Django

Django es 100% Python es un framework de desarrollo Web que ahorra en primer lugar el tiempo y en segundo hace que la programación en este sea entretenida y divertida.

Con este framework se puede crear aplicaciones y a la vez mantenerlas siendo estas de alta calidad exigiendo un poco de esfuerzo.

El desarrollo web en este caso resulta ser entretenido y creativo, nada que ver con otras plataformas que resultan ser repetitivas y frustrantes.

El ambiente se vuelve amigable e interesante para todo desarrollador e incluso para usuarios que gustan aprender de nuevas tecnologías se interesan mucho en este lenguaje de nuevos diseños y su eficiencia al momento de mirar los resultados es satisfactoria.

Django permite un enfoque claro en la parte divertida además de estar dotado con herramientas que permiten realizar las tareas mucho más fácil porque provee un nivel alto en la abstracción

de patrones comunes en el ámbito de desarrollo Web, caminos cortos para tareas de programación. En julio del año 2005 fue anunciado al mundo bajo la licencia BSD, fue nombrado Django en alusión al guitarrista gitano de jazz llamado Django Reinhardt.

2.6 Django

Django es un framework que tiene por característica principal facilitar la tarea del programador o desarrollador de software. Este framework fue trabajado en la administración de páginas de noticias, tal es el caso que su diseño se hace evidente debido a que proporciona una serie de características que facilitan la creación o desarrollo ágil de páginas orientadas a contenidos.
Algunos autores mencionan que Django crea una página administrativa o aplicación incorporada para la administración de contenidos y páginas; ésta página permite crear, actualizar y eliminar objetos del contenido, pero sin duda lleva un registro de cada acción que realice, proporciona de igual forma una interfaz para administrar los usuarios y grupos de usuarios teniendo en cuenta los permisos que se puede asignar.

Django también cuenta con herramientas que permiten proporcionar un sistema de comentarios, herramientas de contenido vía RSS y Atom, cuenta también con páginas planas que sirven para gestionar páginas de contenido sin necesidad de realizar controladores o vistas para dichas páginas, y por último pero no menos importante cuenta con un sistema de redirección de URLs (Uniform Resource Locator o Localizador Uniforme de Recursos).

Este sistema en Django es muy fácil de usar y a la vez ayuda a obtener las rutas de cada vista de cada pantalla que el usuario usa en su aplicación mejorando de esta manera el uso de tiempo y orden al momento de diseñar las páginas web o vistas de los proyectos en desarrollo.

Algunas características de Django también son:

- Es Python su lenguaje base.
- Generar información ordenada automáticamente que sirve para administrar el proyecto o aplicación que se realice.
- Sistema extensible de plantillas teniendo como modelo etiquetas y además con herencia de plantillas.
- Una interfaz de programación de aplicaciones (API), robusta en cuanto se refiere a Base de datos.
- Un sistema de vistas genéricas incorporado, con esto permite el agilizar tareas comunes y que evita el codificar la lógica de las mismas.
- Aplicaciones que pueden instalarse en cualquiera de las páginas de Django llamadas enchufables.
- Un mapeador de tipo objeto relacional.
- Un sistema de tipo middleware que proporciona cacheo, compresión de salida, y además de todo ello permite la normalización de URLs.
- Documentación fácil de comprender para los programadores o usuarios del lenguaje de programación.
- La portabilidad para diferentes sistemas operativos como Windows, Linux, Mac.

2.6.1 Arquitectura Django

Django sigue la arquitectura Modelo Vista Controlador (MVC), aunque los que desarrollan sus proyectos con este framework hacen énfasis en que prefieren no basarse a esto sino que en Django lo que se llama controlador en un verdadero framework MVC aquí se lo denomina vista y la vista en cambio toma el nombre de plantilla. Debido a las capas que tiene el framework Django que son mediator y foundation, permite que los programadores se dediquen a construir

únicamente los objetos Entity y la lógica para mostrar o presentar su trabajo y también desde luego el control para ellos.

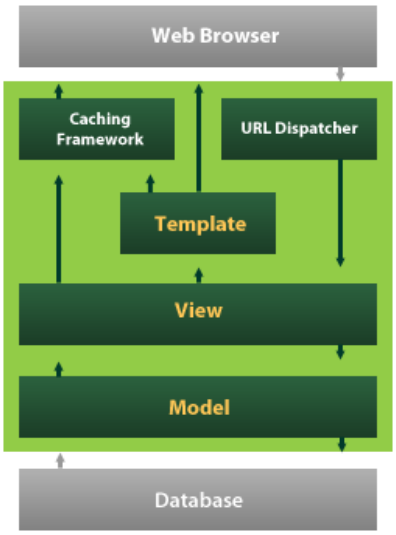


Gráfico 10: Arquitectura MVC en Django
Fuente: http://blog.chattyhive.com/wp-content/uploads/2014/01/Django_mvc.png

- **Lógica de presentación**

  Aquí se maneja la interacción del programador o usuario y su equipo de computación. (PC, Laptop). En El framework Django, ésta tarea es designada al *template* y al *template loader* que recopilan la información y la presentan al usuario.

  Dentro de esta capa de presentación también se encuentra el sistema de configuración de URLs[7].

- **Control**

  En la capa de control se encuentra la lógica de aplicación o el programa en sí. En el framework Django son presentados por los manipulators y las views, la capa de lógica de presentación depende de ésta capa y ésta depende de igual forma de la capa de dominio.

[7] Uniform Resource Locator

- **Mediator**

  Es el principal encargado de manejar la interacción entre los subsistemas Foundation y Entity. En esta parte se realiza el mapeo objeto-relacional que está a cargo del motor del framework Django.

- **Entity**

  Entity o subsistema Entity es el encargado de manejar los objetos de negocio, además tiene una cierta ventaja ya que por el mapeo objeto-relacional de Django permite escribir los objetos que son de tipo Entity de forma fácil y estándar.

- **Foundation**

  Foundation o subsistema foundation tiene la principal tarea de manejar el trabajo con las bases de datos a bajo nivel. Existe el soporte a nivel de foundation para algunas BDD[8] pero para otras aún está en prueba, para saber todo acerca del soporte.

2.6.2 Modelo de Django

En Django un modelo significa la única fuente la cual es la información definitiva sobre los datos. Esta información contiene lo que son los comportamientos de los datos los cuales son almacenados y además campos que son esenciales. Por lo general un modelo se asigna a una tabla, lo que significaría que cada modelo obtiene una sola tabla de la base de datos.

Django utiliza un modelo para ejecutar código SQL detrás de las escenas y retornar estructuras de datos convenientes en Python de esta manera representa las filas de las tablas de la BDD, también usa modelos para representar los conceptos que son de alto nivel que no necesariamente son o pueden ser manejados por SQL.

[8] Base de Datos

Django trabaja de cierta manera así por varias razones se puede decir, algunas son:

- Django necesita conocer la capa de la base de datos de cierta forma ya que la introspección requiere overhead y además es imperfecta. Por este motivo es que se describen de manera explícita los datos en Python, y también se realiza la introspección cuando se está en tiempo de ejecución para así poder determinar el modelo de BDD[9].
- Todo esto es de la introspección se ha venido dando de manera polémica ya que algunas bases de datos no guardan los suficientes metadatos para asegurarse que la introspección sea completa, por lo que no sería 100% confiable.
- Los cambios de contexto resultan un poco complicado, es decir: si se mantiene solo en el lenguaje de Python porque el entorno mentalidad será muy productivo, es lo que Django realiza, pero fuese un cambio de contexto como por ejemplo escribir en Python y luego SQL o SQL y luego Python resulta muchas veces perjudicial.
- SQL permite cierto nivel de metadatos. Hablando de sistema de base de datos, la mayoría, por ejemplo, no se dedica a proveer un tipo de datos especializados para la representación de una url o una dirección de email. Por otro lado la ventaja de Django es que este software si lo hace, lo cual hace que este sea de alto nivel y su productividad sea alta al igual que la reusabilidad de código.
- Una contra de toda esta aproximación, sin embargo, puede ser que el código que tiene Python quede fuera de sincronía respecto a lo que contiene a base de datos. Entonces si se hace cambios en Django, igual se deberá hacer en la base de datos para que su consistencia sea igual con el modelo que se tiene en Python.
- Django también posee una utilidad que genera los modelos, haciendo introspección sobre una base de datos que ya exista, todo esto resultaría de manera útil para trabajar de forma coordinada y eficaz.

[9] Base de Datos

En resumen lo básico de Django con respecto a los modelos sería:

- Un modelo es una clase Python
- Un atributo del modelo representará a un campo de la base de datos (BDD).
- Django proporciona una API[10] de BDD[11] de acceso la misma que es generada de manera automática.
- Una vez que se defina los modelos, es necesario informar al software Django que se va a utilizar dichos modelos, todo este proceso se lo realiza editando el archivo de configuración y modificar INSTALLED_APPS para incrustar el nombre del módulo que necesita.

- **Campos**

Sin duda los campos son una parte esencial de un modelo, estos son especificados como atributos de clase, se debe tener mucha cautela al momento de elegir sus nombres para que no entren en conflicto con el modelo API.

2.6.3 Soporte de Base de Datos (BDD)

Django soporta algunas bases de datos por nombrar algunas como son MySQL, SQLite3, PostgreSQL, Oracle y está trabajando en la actualidad en un adaptador para SQL Server. También le está permitido a Django extenderse para poder soportar otras bases de datos, aunque por lo general nunca lo ha usado. Para esto se modifica el diccionario DATABASES el cual se encuentra en el archivo settings.py de configuraciones.

Django proporciona una vez creados los data models una abstracción de la BDD (Base de Datos), a través de su API (Interfaz de Programación de Aplicaciones), permitiendo de esta

[10] Application Programming Interface
[11] Base de Datos

manera crear, actualizar, borrar y actualizar objetos. Django también permite realizar consultas SQL (Structured Query Language) directamente.

Al interactuar con una base de datos, estamos hablando de lógica arbitraria, porque detrás de cámaras se podría decir, un sitio web es impulsado por una base de datos y esta se conecta a un servidor, recuperando así, datos y mostrándolos en una agradable y vistosa página Web. Del mismo modo también puede proporcionar funcionalidad la que permite a los visitantes de sitio poblar la base de datos por cuenta propia.

A continuación se muestran las bases de datos que puede usar Django, teniendo en cuenta que cada una utiliza o usa un adaptador requerido, cada uno de ellos se encuentran libremente en la Web.

Configuración	Base de datos	Adaptador requerido
postgresql	PostgreSQL	psycopg versión 1.x
postgresql_psycopg2	PostgreSQL	psycopg versión 2.x
mysql	MySQL	MySQLdb
sqlite3	SQLite	No necesita adaptador si se usa Python 2.5+. En caso contrario, pysqlite
ado_mssql	Microsoft SQL Server	adodbapi version 2.0.1+
oracle	Oracle	cx_Oracle

Gráfico 11: Base de Datos y su Respectivo Adaptador
Fuente: http://librosweb.es/libro/django_1_0/capitulo_5/configuracion_de_la_base_de_datos.html

2.6.4 Soporte de Servidores Web

El framework Django como en otros incluye un servidor liviano, este sirve para la realización de pruebas y trabajar en un entorno de desarrollo. Es recomendado usar el servidor Apache 2 con mod_python, siempre que se esté en la etapa de producción; Django soporta también la

especificación WSGI[12], esto quiere decir que puede correr sobre algunos servidores como FastCGI[13] o SCGI[14]. En la siguiente figura se muestra la arquitectura de o modelo incluyendo sus servidores en Django.

2.6.5 Apps

En Django ninguno de estos archivos hacen un sitio dinámico funcional. Por lo tanto se necesita crear ciertas aplicaciones dentro de la misma aplicación, las cuales permitirán crear funcionalidad en el desarrollo Web. Se debe tener en cuenta que Django posee la filosofía de No repitas lo que ha hiciste, esto quiere decir entonces que las apps funcionan de la misma manera. Todo proyecto, programa o software que se desarrolle puede hacer uso de las apps, lo único que se realiza es la importación al trabajo que se esté desarrollando, y no se tendría que volver a implementar o volverse a codificar una app de la misma naturaleza para usarle.

2.6.6 Sistema de Plantillas Django

Una plantilla en Django es una cadena de texto la cual separa la parte de la presentación del documento que contiene sus datos. Ésta también define las etiquetas de pantalla esto quiere decir que regula el documento que se va mostrar.

Por lo general las plantillas son usadas para producir código HTML, pero en este framework las plantillas son capaces de crear un formato cualquiera basado en lo que es texto. Cada plantilla del software Django tiene acceso a varias etiquetas y a la vez a los filtros que son incorporados. Aunque cabe recalcar que se puede crear también los filtros y etiquetas.

[12] Web Server Gateway Interface
[13] Common Gateway Interface
[14] Simple Common Gateway Interface

Más que cualquier otro componente de las aplicaciones web, hay que tener en cuenta que la opinión de los programadores sobre el sistema de plantillas varía por lo general.
Python no sólo implementa algunas plantillas, sino cientos de lenguajes de plantillas que además son de código abierto. Para todo desarrollador es un rito el crear su propia plantilla.

Teniendo en cuenta todo lo anterior Django no obliga a usar sus plantillas, el hecho de que las tenga es para ser un framework completo, productivo, que sirva de ayuda al desarrollador, más no para obligarle a usar sus herramientas.
El sistema de plantillas es una forma de hacer ver que es la herramienta la cual se encarga de controlar la presentación y lógica de negocios en sí. Claro que habría que tener en cuenta que las plantillas no deben admitir ningún tipo de funcionalidad que quiera olvidar el concepto básico.
Por tal razón, nunca se puede llamar a código Python desde el interior de una plantilla de Django, esto es imposible.

Es posible realizar etiquetas personalizadas debido a que existen etiquetas que son limitadas en sus acciones, tal es la razón de poder llamar código desde una plantilla, que Django intencionalmente no permite que se realice esta acción y evita ejecutar código arbitrario del software Python.

2.6.7 Clases Middleware

Middleware es un conjunto de opciones en el proceso de petición-respuesta del framework Django. Es un light, de bajo nivel del sistema conocido como “plug-in” que sirve para alterar globalmente entrada o salida de Django.

Cada componente o middleware es responsable de hacer una función específica, como por ejemplo en este caso asociar a los usuarios con solicitudes utilizando sesiones para los mismos con el middleware AuthenticationMiddleware.

Se debe tener en cuenta que Django permite también realizar su propio middleware debido a que un middleware es una clase o un conjunto de comandos que realizan cierta operación.

Django viene implementado con una gran variedad de middleware para solución de problemas que se pueden presentar al momento de realizar alguna tarea de programación en dicho lenguaje a continuación se enlista brevemente un conjunto de middlewares las cuales son de gran ayuda para los usuarios:

- Middleware para soporte de autenticación
- Middleware Common
- Middleware de compresión
- Middleware Get (condicional)
- Middleware X-Forwarded-For para soporte del uso de proxy inverso
- Middleware para el soporte de sesiones
- Middleware de caché de todo el sitio Web
- Middleware para transacciones y,
- Middleware X-View

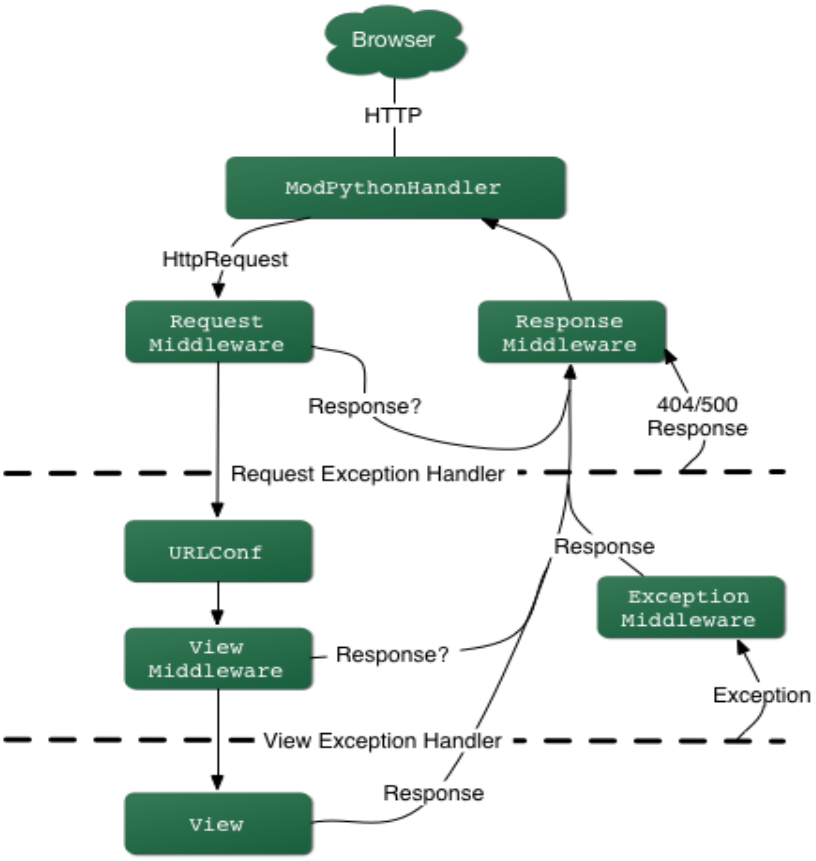


Gráfico 12: Arquitectura Django con Middlewars
Fuente: http://osl2.uca.es/wikiCE/images/2/2e/Get_response.png

2.6.8 Administración en Django

Gráfico 13: Pantalla de acceso al Administrador Django
Fuente: http://4.bp.blogspot.com/-C1ZJK8GgW4w/UBv7kXKLdpI/AAAAAAAAAB4/VeYZBUKSrkM/s320/admin01.png

Cuando se hace mención a los sitios Web o una interfaz que permita administrar, se hace referencia a que es una parte importante y esencial de la infraestructura, ésta interfaz debe ser

basada en la web, siempre limitada a los administradores que son autorizados los cuales tienen permiso de agregar, editar, y eliminar el contenido del sitio.

Algunos ejemplos de interfaces de administración son: la interfaz que se tiene en un blog para escribir, herramientas que los clientes utilizan para la actualización de los comunicados de prensa o noticias en la web, incluso los sitios que son privados en los cuales los editores moderan los comentarios de los lectores.

Para realizar una interfaz de administración resulta todo un lío, aparte que resulta ser aburrido construirlas, porque se vuelve algo repetitivo y cansón, el tener que autenticar usuarios, validar las entradas, mostrar formularios, mostrarlos y demás.

Pero Django tiene la solución perfecta porque él las realiza, es tan fácil construir estas interfaces ya que solo las realiza en un par de líneas y problema resuelto.
La interfaz de administración de Django es la característica más atractiva y la mayoría de las personas que usan este framework están en total acuerdo, pero no todo el que usa Django la necesita, esto por supuesto es algo opcional.

La interfaz de administración está diseñada para usuarios que son no técnicos y por lo tanto ésta debe ser clara de tal forma que el usuario la comprenda en su totalidad. Además tiene un mayor impacto cuando los usuarios no son técnicos y necesitan ser capaces de ingresar datos; en sí ese sería el objetivo o propósito detrás de esta característica, ya que la razón de ser de la interfaz de administración sería facilitar trabajos de simultáneos de los productores en este caso de contenidos y programadores.

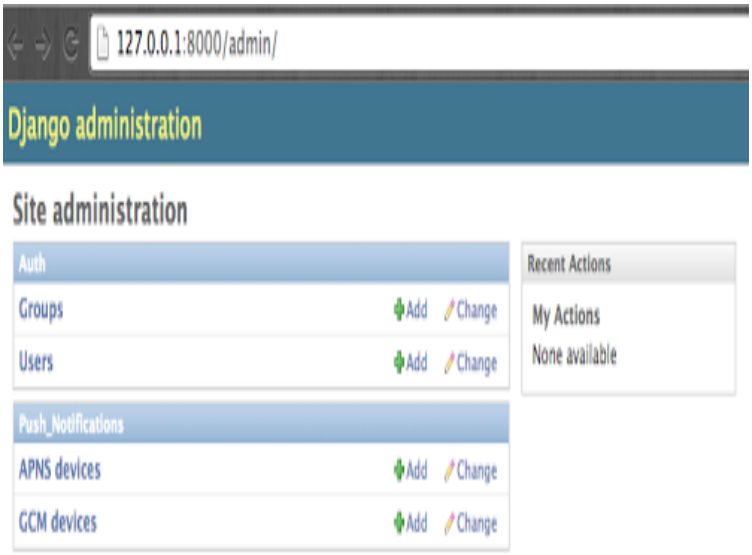


Gráfico 14: Interfaz de Administración de Django
Fuente: http://3.bp.blogspot.com/_51U5_NXg068/TNLDA_n3R5I/AAAAAAAAABg/vBlJUdvsakg/s1600/admin1.png

Esta interfaz es también útil en otros casos como por ejemplo: En la inspección de los modelos de datos, en la gestión de datos adquiridos, en fin es una de las herramientas amigables que hacen a Django un software amigable y que gusta a las personas que desarrollan aplicaciones en este lenguaje.

2.6.9 Vistas Genéricas

Django cubre otra necesidad en cuanto a la programación se refiere ya que evite el proceso complicado de realizar las vistas debido a que es un tema aburrido y a veces incluso monótono.

Este tema de generar las vistas Django lo ha solucionado, con unas vistas genéricas que recogen patrones comunes y ciertos estilos encontrados en el desarrollo de vistas y los abstraen de modo que puedas escribir rápidamente vistas comunes sin escribir demasiado código de hecho.

Casi la mayoría de las vistas pueden ser reescritas con la ayuda que brinda Django con sus vistas genéricas. Este tipo de vistas son encargadas de realizar tareas sencillas, como crear una

vista detallada, o presentar objetos que sean de tipo fecha y también permitir a los usuarios crear, actualizar y borrar objetos esto se puede realizar teniendo o no, la autorización correspondiente.

Las vistas genéricas se usan, creando diccionarios de configuración, esto se lo hace en los archivos URLconf. Este proceso sin duda se identifica como un arte de magia, una vista sin código, pero en realidad se toma información del diccionario de parámetros extra y usa la información para realizar la vista genérica.

2.6.10 Formularios

Django usa el concepto de un formulario perfecto en el cual se considera algunos aspectos como:

- La accesibilidad y la usabilidad que son importantes en este tema.
- Cada uno de los datos que son suministrados deberían ser sometidos a una extensiva evaluación que es una regla de seguridad en una aplicación web no sólo en Django sino a nivel general.
- En caso de que el usuario cometa una equivocación, el formulario debe estar listo para volverse a mostrar y detallar el error o la equivocación que se ha suscitado, no conforme con esto el formulario que aparece debe estar lleno ya con los campos que se ingresó para evitar que el usuario vuelva a completar la información que ya fue digitada.
- El formulario en caso de que los datos no se ingresen de manera correcta deberá seguirse mostrando una y otra vez, hasta que todos los campos estén llenos de manera correcta.

Construir un formulario de esta manera es demasiado trabajo, pero como siempre Django suele facilitar las cosas, el framework ya está diseñado para hacer la mayor parte del trabajo por el usuario. Se usa una simple plantilla y se le da la información requerida como reglas para la validación, descripción de los campos del formulario y Django lo pone en marcha. De esta manera se obtiene un formulario perfecto.

2.6.11 Seguridad

Hoy en día Internet es un lugar no seguro, es normal descubrir graves problemas de seguridad todos los días. Los virus, los hackers, sitios falsos, y más. Parte de la responsabilidad de un desarrollador o programador web al realizar una aplicación debe considerar la seguridad, aunque resulta difícil debido a que un atacante solo necesita encontrar una sola debilidad, para hacer daño.

Django intenta mitigar esta amenaza o dificultad y está diseñado para proteger al usuario automáticamente de muchos errores de seguridad que incluso son cometidos por los desarrolladores experimentados.

2.7 Postgresql

PostgreSQL es un potente sistema de base de datos, de código abierto objeto-relacional. Cuenta con más de 15 años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación por su fiabilidad, integridad de datos y corrección.

Se ejecuta en todos los sistemas operativos, incluyendo Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), y Windows. Es totalmente compatible con ACID, tiene soporte completo para claves foráneas, combinaciones, vistas, triggers y procedimientos

almacenados (en varios idiomas). Incluye más de SQL: 2008 tipos de datos, incluyendo INTEGER, numéricos, booleanos, CHAR, VARCHAR, DATE, INTERVALO y TIMESTAMP. También es compatible con el almacenamiento de grandes objetos binarios, incluyendo imágenes, sonidos o vídeos. Cuenta con interfaces de programación nativo de C / C ++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, entre otros, y la documentación excepcional.

Una base de datos de clase empresarial, PostgreSQL cuenta con características sofisticadas como Multi-Version Concurrency Control (MVCC), punto en el tiempo de recuperación, tablespaces, replicación asincrónica, transacciones anidadas (puntos de retorno), en línea copias de seguridad / calor, un sofisticado planificador de consulta / optimizador, y escribir por delante el registro para la tolerancia a fallos.

Es compatible con los juegos de caracteres internacionales, codificación de caracteres multibyte, Unicode, y es consciente de la configuración regional de la clasificación, mayúsculas y minúsculas, y el formato. Es altamente escalable tanto en la enorme cantidad de datos que puede manejar y en el número de usuarios simultáneos que puede acomodar.

Existen sistemas de PostgreSQL activos en entornos de producción que manejan más de 4 terabytes de datos.

Algunos límites generales o características de PostgreSQL se incluyen en la siguiente tabla, tomando en cuenta que serán los más importantes para realizar un proyecto o aplicación en cuanto a uso de software y hardware se refiere.

Tabla 1: *Características de PostgreSql.*

Límite	Valor
Máximo Base de Datos Tamaño	Ilimitado
Máximo Tamaño de la tabla	32 TB
Máxima Fila Tamaño	1.6 TB
Tamaño Máximo de Campo	1GB
Filas máximas por Tabla	Ilimitado
Columnas máximas por Tabla	250-1600 dependiendo de los tipos de columna
Los índices máximos por Tabla	Ilimitado

Fuente: Propia

PostgreSQL ha ganado los elogios de sus usuarios y de reconocimiento de la industria, incluyendo el Sistema de Base de Datos de Linux New Media Award a la mejor, y cinco veces ganador del Premio de los Editores de Linux Journal 'a la mejor DBMS.

CAPÍTULO 3

3. ANÁLISIS ANTES Y DESPUÉS DE IMPLEMENTAR EL SISTEMA

En este tiempo que la empresa ha ido generando herramientas, no lo ha hecho para uso de cada uno de los usuarios, sino de manera global.

Lo que se pretende hacer hoy, es el análisis de una herramienta que sea capaz de dar información al usuario de manera sencilla y dinámica. Para esto la idea que se tiene y se la va a plasmar es que cada uno de los usuarios tenga un software que les permita llevar un control de publicidad y de ingresos al Network marketing en esta empresa.

Con esta idea se hará que cada usuario cree su empresa y lleve el control de las personas que van a trabajar en su organización, a la vez es una herramienta publicitaria o de información, porque contendrá información de los servicios que brinda y de la forma de realizar el negocio.

3.1. Estudio Comparativo de Herramientas

Una correcta comparación respecto a las características más importantes en cuanto a los frameworks Ruby On Rails y Django, es el eje fundamental para seleccionar cuál de las herramientas será la mejor para el desarrollo del aplicativo, y la que genere una mejor solución al problema planteado. Estos puntos deben ser analizados de manera conveniente para obtener resultados positivos.

Los puntos que se toman en cuenta deberán ser valorados por una escala la cual deberá asumir los parámetros que son tomados de acuerdo a la necesidad que se ha planteado y para satisfacción de un mejor entorno de desarrollo en cuanto a la programación se refiere.

3.1.1 Escalas de valoración

Una escala de valoración es la que permite obtener resultados de un determinado tema a tratar. Con esto se podrá sacar conclusiones y determinar resultados sobre el tema que se elija. Una escala siempre contiene parámetros los cuales son las bases sobre las que se va a trabajar o también conocidos como condicionantes, variables, etc. Existe un sinnúmero de escalas las que permiten trabajar de manera efectiva para un análisis o investigación por el momento se usará la técnica o escala e Likert.

Bien para realizar la comparación se utilizará la escala de LIKERT como se ha dicho antes, la cual mide o valora las opciones o ítems que se propone para realizar un trabajo, en dicha escala se mide el criterio de las personas o de la información que se obtiene en una investigación, la cual puede manejar 4 o 5 o 7 parámetros todo esto en función de lo que estime conveniente él usuario. Para tener una idea de la escala de Likert se a cita un ejemplo, el cual indicará la manera que tiene para trabajar dicha escala:

Cuadro 2. Escala de evaluaciones

Valor	Significado
1	El atributo no es importante en la evaluación del desempeño del encuestado.
2	El atributo es ligeramente importante en la evaluación del desempeño del encuestado.
3	El atributo es regularmente importante en la evaluación del desempeño del encuestado.
4	El atributo es importante en la evaluación del desempeño del encuestado.
5	El atributo es muy importante en la evaluación del desempeño del encuestado.

Gráfico 15: Ejemplo de uso de Likert
Fuente: http://www.scielo.org.mx/img/revistas/peredu/v34n137/a9c2.jpg

3.1.2 Parámetros de evaluación

Los parámetros que se describen a continuación serán los pilares fundamentales para llegar a un resultado el cual se está buscando. Los parámetros son los siguientes:

- **Aprendizaje:** para ser una herramienta eficiente debe ser fácil su uso y su comprensión, para así de este modo interactuar de una manera clara o precisa ahorrando tiempo y evitando tener problemas al momento de realizar alguna aplicación o software.
- **Portabilidad:** este aspecto es un importante punto debido a que necesitamos que nuestra aplicación se puede acoplar a cualquier sistema o sea compatible con cualquier infraestructura.
- **Documentación:** en este punto hace referencia a la facilidad de encontrar documentación (libros), páginas web, sitios o blogs que sean de mucha ayuda con su información de las herramientas que se investigan en esta parte.
- **Soporte:** toda herramienta o a su vez toda aplicación debe tener soporte, esto quiere decir que debe existir una documentación en cuanto a su manejo, a sus ventajas, desventajas, blogs o libros en los cuales se hable del mantenimiento o uso correcto de la herramienta en este caso de los frameworks que se va a investigar.
- **Plantillas:** son las herramientas que tienen por defecto cada uno de los frameworks para facilitar al usuario la creación de modelos de páginas web o blogs.
- **Vistas Dinámicas:** este parámetro es una de las opciones que más llama la atención ya que son útiles para realizar actualizaciones inmediatas en una página web.
- **Seguridad:** el internet como todos sabemos es un peligro en cuanto a seguridad, pero los frameworks han trabajado mucho en eso para brindar una mejor protección de los datos e información.
- **Sesiones:** un sitio seguro debe tener una pantalla para realizar un inicio de sesión de usuario para protección y además según esto pueda realizar acciones según su categoría.

A continuación se explica el porqué de los parámetros a utilizarse:

Cada uno de estos parámetros, son importantes para obtener un resultado de un software que ayude a solucionar el problema pero que también tenga las garantías necesarias para que la información sea segura y evite tener conflictos en esta parte.

En la siguiente tabla se explica la valoración y cómo se va a trabajar para sacar el resultado final y la herramienta ganadora.

Se tendrá el nombre de la herramienta, su indicador o parámetro, su valoración la cual estará entre el rango de 1 a 5 y ésta significará lo siguiente.

Tabla 2: Puntaje de Indicadores Establecidos

Nombre del framework		
Características	Puntos	Porcentaje
Deficiente	1	20%
Poco eficiente	2	40%
Limitado	3	60%
Eficiente	4	80%
Muy Eficiente	5	100%

Fuente: propia

De esta manera se obtiene los indicadores que determinan cuál será la herramienta ganadora para dar solución al problema que se tiene actualmente. Estos parámetros serán esenciales al momento de calificar a los frameworks seleccionados para su estudio porque son las características de más importancia en los dos, según las investigaciones que se ha hecho.

3.1.3 Análisis de Indicadores

- **Parámetro Aprendizaje**

Tabla 3: Parámetro Aprendizaje

Parámetro Aprendizaje		
Herramienta	**Puntaje**	**Porcentaje %**
Ruby On Rails	4	80%
Django	5	100%

Fuente: Autor

Análisis:

Ruby On Rails maneja en cierta parte un aprendizaje un tanto cansado debido a que no existen muchas herramientas que faciliten el estado monótono que se genera a la hora de programar, en sí Ruby On Rails es un potente software pero también el aprendizaje de esta herramienta no es tan rápido, con esto no se quiere decir que sea complejo, al contrario, pero hay que tener en cuenta todas las cosas que se pueden ver en esta herramienta.

Por otro lado Django hace más divertida la programación y el hecho de generar vistas sin mucha codificación, gracias al sistema de plantillas que maneja para facilitar la realización de las mismas.

Por esta razón se otorga un punto más a Django, aunque su diferencia es mínima debido a que son herramientas muy poderosas al momento de usarlas para un desarrollo de trabajo sea sencillo o complicado. El mundo de estas poderosas herramientas es cada vez más equitativos pero competitivos entre sí.

- **Portabilidad**

Tabla 4: Parámetro Portabilidad

Parámetro Portabilidad		
Herramienta	**Puntaje**	**Porcentaje %**
Ruby On Rails	4	80%
Django	5	100%

Fuente: Autor

Análisis:

La portabilidad de cada framework es un parámetro que debe ser analizado de manera minuciosa, ya que existen varias teorías que hacen confundir a los desarrolladores en cuanto a esto se refiere. Para tener claro la portabilidad de una aplicación es que, si yo como programador o desarrollador tomo mi aplicación y la llevo a un lenguaje diferente como puede ser Linux, Mac, Windows, este debe funcionar de manera correcta.

Esto permite tener un punto de vista el cual menciona que es muy importante que todas las herramientas que se utilicen en el desarrollo de la aplicación o proyecto deberán ser compatibles para cada uno de los sistemas operativos que existen en la actualidad.

Por lo tanto no se debe confundir el tema de la portabilidad de un proyecto, es de manera sencilla para la comprensión de todos, si se necesita usar la aplicación en cualquier sistema operativo ésta deberá funcionar sin ningún inconveniente.

Así pues, en este caso se ha estudiado a cada una de las herramientas las cuales son: Ruby on Rails y Django, éstas son compatibles y existen los instaladores para cualquier sistema operativo, donde funcionan de manera correcta.

- **Parámetro Documentación**

Tabla 5: Parámetro Documentación

Parámetro Documentación		
Herramienta	**Puntaje**	**Porcentaje %**
Ruby On Rails	4	80%
Django	5	100%

Fuente: Autor

Análisis:

Un problema frecuente que se encuentra al momento de buscar información de Ruby on Rails es que no todas sus páginas, blogs existentes se encuentran en español, o incluso existen inconvenientes de no poder encontrar un determinado tema.

Django por lo general está en su tiempo y existen muchas páginas que hablan de su uso y de su poder que tiene en cuanto a desarrollo de software, existen tutoriales, blogs, etc. Respecto al lenguaje de programación. Este acontecimiento ha permitido el interés de personas amantes a la programación debido a que ha mejorado la manera de realizar ciertas características que resultaban tediosas como por ejemplo realizar las famosas vistas.

Con sus nuevas metodologías estas herramientas han logrado un cambio considerable para que las personas se dediquen a realizar softwares de calidad, por todas las facilidades que brindan al usuario.

La interfaz juega un papel muy importante a la hora de programar y estas herramientas han logrado solucionar ese tipo de problemas.

La documentación se la obtiene de manera fácil con el uso del internet, la única traba es el idioma en el que se encuentran los archivos, manuales, blogs, pero en un futuro no muy lejano se sabe que todos esos documentos se los podrá encontrar en cualquier lenguaje con el avance y uso de las herramientas, ya que las necesidades hacen que se solucione cualquier problema.

- **Parámetro Soporte**

Tabla 6: Parámetro Soporte

Parámetro Soporte		
Herramienta	**Puntaje**	**Porcentaje %**
Ruby On Rails	5	100%
Django	5	100%

Fuente: Autor

Análisis:

Cabe resalta que tanto Ruby On Rails y Django manejan un estricto cuidado en el aspecto del soporte, tanto en servidores web, como el soporte a base de datos. Cada framework pues sin duda maneja sus propias herramientas por mencionar servidores en Ruby On Rails: Apache, Nginx, Mongrel, y en cuanto a base de datos PostgreSQL, SQLite 3, MySQL, SQLServer y Oracle.

Servidores en Django: Apache 2 con mod_python, Nginx. Y en cuanto a base de datos: PostgreSQL, SQLite 3, MySQL y SQLServer. Como se puede apreciar cada uno de los frameworks es eficiente en el soporte que brinda para realizar una aplicación de esta manera igual hace que la programación no solo se use una base de datos sino que puede escoger la que sea compatible y esté mejor en cuanto a necesidades se refiere.

También se puede elegir la base de datos con la que el usuario tenga experiencia para acelerar el proceso de desarrollo de ahí que para cada una de ellas el framework a elegir pues se adapta de mejor manera.

- **Parámetro Plantillas**

Tabla 7: Parámetro Plantillas

Parámetro Plantillas		
Herramienta	**Puntaje**	**Porcentaje %**
Ruby On Rails	4	80%
Django	5	100%

Fuente: Autor

Análisis:

Para aplicar una plantilla en el framework Ruby On Rails es necesario proporcionar el generador Rails pero con la ubicación de la plantilla que se desea utilizar todo esto mediante la opción –m. La cual puede ser una ruta o dirección a un archivo cualquiera o incluso también puede ser una URL (Localizador de Recursos Uniforme).

Del mismo modo se puede aplicar una plantilla a una aplicación ya existente en Ruby On Rails con una tarea llamada rake *rails: template.* En todo caso o por lo general el uso de las plantillas API (Interfaz de Programación de Aplicaciones) en Ruby On Rails, es fácil de entender.

Por otro lado en el framework Django una plantilla es una cadena de texto la cual separa lo que es presentación del documento de todos sus datos. La plantilla se encarga de regular como se

lo debe mostrar el documento, por lo general y según conocimientos adquiridos una plantilla genera HTML pero en Django generan cualquier tipo de formato el cual se basa en textos.

Django provee una poderosa API (Interfaz de Programación de Aplicaciones), para cargar plantillas pero del disco, esto con el fin de evitar la redundancia en el uso de plantillas.
Para hacer uso de la API (Interfaz de Programación de Aplicaciones) para cargar las plantillas, primero se debe darle a conocer a la herramienta o framework, el lugar en donde se aloja o se guarda las plantillas, el lugar para especificar todo esto es en el archivo de configuraciones, se debe tener en cuenta evitar errores en los archivos de configuraciones.

Django posee la herencia de plantillas, para evitar la redundancia con la etiqueta {%*include%}* esta etiqueta se inserta dentro del HTML pero no siempre funciona de manera correcta, entonces para evitar todo tipo de problema es mejor usar la herencia de plantillas.

La esencia de la herencia de plantillas permite realizar una base la cual se denomina esqueleto que contiene todas las partes de un sitio y aparte de esto también define bloques que cada uno de los hijos pueden sobrescribir.

- **Parámetro Vistas Dinámicas**

Tabla 8: Parámetro Vistas Dinámicas

Parámetro Vistas Dinámicas		
Herramienta	**Puntaje**	**Porcentaje %**
Ruby On Rails	5	100%
Django	5	100%

Fuente: Autor

Análisis: El framework Ruby On Rails tiene una técnica sencilla pero a la vez poderosa para realizar las vistas dinámicas esta técnica se llama Scaffold, es muy útil al menos para desarrolladores que recién empiezan, esta técnica se encarga de proporcionar una sencilla interfaz a todos los datos de la base. Beneficios que brinda Scaffold:

- ✓ Genera el modelo.
- ✓ Genera cada una de las rutas de forma automática.
- ✓ Respeta y se apega estrictamente a los estándares de Rails.
- ✓ Genera las vistas y el controlador del CRUD (Crear, Actualizar, Editar y Eliminar).

En Django las vistas dinámicas se manejan con una simple función en Python que toma un *Web Request* y retorna una respuesta web (Response). Para realizar una vista dinámica en Django se debe tener en cuenta tres pasos muy importantes:

- ✓ Definir la vista
- ✓ Generar una plantilla (Template), y
- ✓ Realizar la conexión de las URLs (Interfaz de Programación de Aplicaciones), con las vistas.

Este código puede ir en cualquier lugar que se desee, siempre y cuando esté en el camino o ámbito de Python.

- **Parámetro Seguridad**

Tabla 9: Parámetro Seguridad

Parámetro Soporte		
Herramienta	**Puntaje**	**Porcentaje %**
Ruby On Rails	5	100%
Django	5	100%

Fuente: Autor

Análisis:

Ruby On Rails pone varias opciones para proteger tu sitio web, una de ellas es la autenticación mediante HTTP. Esto consiste en proteger el acceso que se tiene a algunas de las acciones que se tiene definidas en el controlador *ArticlesController.*

Esto hace que si el usuario no está autenticado, no puede acceder a las acciones que se definen en el archivo.

Para realizar todo eso se usa el archivo que lleva por nombre *http_basic_authenticate_with* de Ruby On Rails. La autenticación debe estar siempre al principio del controlador llamado *ArticlesController* para así de esta manera indicarle que queremos proteger todas las acciones salvao *index* y *show.*

Otra restricción básica es que los usuarios no puedan borrar lo que es comentarios para ello se deberá añadir en el archivo CommentController.rb *http_basic_authenticate_with name: "dhh", password: "secret", only: destroy.*

- **Parámetro Sesiones**

Tabla 10: Parámetro Soporte

Parámetro Soporte		
Herramienta	**Puntaje**	**Porcentaje %**
Ruby On Rails	5	100%
Django	5	100%

Fuente: Autor

Análisis:

Algo que es importante y que por lo general siempre queda expuesto a tener problemas en seguridad es el tema de las sesiones. Teniendo en cuenta que existen riesgos y que el tráfico en la red es demasiado concurrido, existe un riesgo con los usuarios anónimos y se abalanzan contra cualquier sitio o página que existe en el internet.

Una idea equivocada es no tener en cuenta que detrás de cada navegador existen personas y que no sirve para conectar máquinas sino para conectar personas. Así que si se desea desarrollar un sitio web realmente de competencia, se debe plantear cómo tratar a las personas que trabajan detrás de un navegador.

Una sesión suele constituirse en un hash el cual almacena valores y además un identificador de sesiones, que por lo general es una cadena de 32 caracteres que identifica al hash. Aquí entran las cookies las mismas que son enviadas al navegador incluyendo el identificador de cada sesión, el navegador por otro lado enviará al servidor cada petición que realice el cliente.

Ruby On Rails trata de realizar una máxima seguridad en cuanto a sesiones por lo que tiene los siguientes puntos que son los pilares fundamentales para realizar un buen manejo:

- **Sesión:** en este parámetro Ruby usa una cadena de 32 bits por el formato MD5 que hasta la actualidad no ha generado ningún caso de inseguridad, pero tiene un pequeño inconveniente el cual es que ha generado algunas colisiones, por lo cual se está pensando en otro tipo de entrada con el mismo valor hash.
- **Sesión secuestro:** un gran problema es el usar un usuario y contraseña como lo usan la mayoría de aplicaciones web y almacenar en un hash entonces cuando se realiza esto, a partir de ese momento la sesión ya es válida porque el identificador en la cookie

identifica la sesión sin necesidad de volver a pedir una identificación o autenticación. La cookie de un sitio web sirve como un archivo temporal y el problema sucita aquí debido a que si alguien se apodera de la cookie de otra persona, puede utilizar la aplicación o sistema como este usuario. Para evitar esto se debe forzar a una conexión segura a través de SSL en el archivo de configuración. El objetivo en usar o robar una cookie siempre se debe a querer realizar una estafa de parte de los atacantes. Los datos informan que se han realizado robos desde $10 a $1000 dependiendo la cantidad de los fondos de la víctima.

- **Directrices de sesiones:** no se debe guardar los objetos que sean de gran tamaño en una sesión. Se debe almacenarlos e una base de datos y lo que se debe guardar es únicamente la identificación en la sesión. Esto es lo más conveniente debido a que se puede modificar ya sea la estructura de algún objeto o versiones del sistema.
- **Almacenamiento Sesión:** el software Ruby On Rails tiene varios mecanismos que ayudan al almacenamiento para cada uno de los hash de sesión. Uno más importante viene hacer el ActionDispatch::Session::CookieStore. Este método guarda el hash de la sesión directamente en una cookie pero en el lado del cliente. Entonces el servidor recupera el hash sesión de la cookie y también elimina de esta manera la necesidad de un identificador de sesiones. De esta manera también aumenta la velocidad de la aplicación. Se debe tomar en cuenta que una cookie por lo general implica un tamaño de 4KB.
- **Fijación de sesión-contramedidas:** la contramedida es más eficaz si se realiza una emisión de un nuevo identificador de sesiones y de esta manera declarar al anterior como inválido si la sesión se inicia correctamente.
- **Sesión de caducidad:** una solución a la inseguridad es la caducidad debido a un tiempo estimado en el cual la cookie queda inválida o caduca.

Django define de manera esencial el uso de la protección en cuanto a cookies se refiere debido a que las personas pueden usar las cookies a favor o en contra. Toda la teoría y uso de las cookies se viene dando debido a que solo se podía manejar la seguridad por estados y esto a lo largo podría representar un gran problema para los desarrolladores web. Aspectos que se debe tomar en cuenta siempre se enlista más adelante:

- **¿Cómo se define las cookies y su uso?**

  En Django se usa las prestaciones de alto nivel, pero se debe tomar en cuenta cómo funcionan en bajo nivel vamos a tener una idea sobre el uso de las cookies y lo que realizan, para tener una mayor utilidad en caso de que tocase trabajar con cookies directamente. Para obtener los valores de las cookies ya está definida en Django, cada objeto que sea de tipo petición o request como se lo conoce, contiene un objeto cookies el mismo que se comporta como un diccionario, éste se usa para leer cualquier cookie que el navegador que se esté usando envié a la vista generada.

- **Cookies son de doble filo**

  El almacenamiento es opcional: cada navegador permite al usuario la definición o aceptación o rechazo de las cookies. A pesar de que las cookies son muy usadas también son un ejemplo de la situación no confiable que representan. Por lo que no se debe almacenar información importante en las cookies, por seguridad de que otros usuarios puedan usar la información o porque el mismo navegador borre las cookies sin razón alguna.

- **Las cookies no seguras**

  Las cookies bien pueden ser no seguras especialmente las que no son enviadas mediante un protocolo HTTPS, las cookies pueden ser leídas por terceras personas, a esto se le denomina ataques de tipo snooping (husmear, espiar). Por tal razón nunca se debe almacenar información confidencial en las cookies.La mayoría de sistemas web usan siempre algo como IsLoggedIn=1 en una cookie cuando el usuario es validado, pero no lleva más de unos

cuantos segundos hacerle creer al sistema que su seguridad es excelente cuando la realidad es otra. Este problema se ha solucionado gracias al uso de las cookies seguras que se analiza más adelante en el documento.

- **Seguridad con sesiones**

  Con todo lo anterior Django ha tomado la decisión de realizar las seguridades mediante sesiones las cuales protegen la información. El entorno de las sesiones permite almacenar y recuperar los datos sea cualquiera de ellos basándose en la sesión misma del usuario, debido a que almacena la información que es relevante solo en el servidor y también abstrae todo el problema del envío de las cookies como también de su recepción.

- **Activar Sesiones en Django**

  Las sesiones se implementan mediante un poco de middleware y un modelo Django. Para esto se debe editar el valor de MIDDLEWARE_CLASSES de forma que contenga 'django.contrib.sessions.middleware.SessionMiddleware' y también comprobar que 'django.contrib.sessions' esté incluido en el valor de INSTALLED_APPS.

- **Usar las sesiones en una vista**

  Cuando las sesiones están activas los objetos HttpRequest que son el primer argumento de las funciones sea cualquiera de éstas en una vista, llevará como atributo definido por defecto el llamado sesión, este atributo se comporta como un diccionario, porque, se puede leer y escribir en él de forma sencilla al igual que cualquier diccionario que sea normal.

- **Reglas para usar de manera eficaz las sesiones en Django**

  La primera.- debe usarse cadenas de textos normales solo que estas deben tener valores en clave en request.session, esto se usa en vez de los enteros o incluso objetos. La segunda.- es que los valores a usarse en las claves de una sesión y si estos son de carácter subrayado estén reservados para uso interno de Django.

3.1.4 Determinación del software ganador

Realizando un análisis de todos los indicadores para determinar el software ganador, se obtiene una tabla general de los puntajes que contiene cada uno de los ítems tomados como base con su respectiva calificación en porcentaje, para luego encontrar al mejor framework o herramienta para implementar el sistema, de acuerdo a la necesidad que actualmente se tiene, con esto se logrará dar una solución eficiente y de manera dinámica al problema que se ha suscitado.

A continuación se presenta la tabla que contiene a cada uno de los ítems tomados como base para su estudio y posteriormente para determinar un ganador:

Tabla 11: Resultados porcentuales de frameworks según sus indicadores

Parámetros	Ruby On Rails	Django
Aprendizaje	80%	100%
Portabilidad	80%	100%
Documentación	80%	100%
Soporte	100%	100%
Plantillas	80%	100%
Vistas Dinámicas	100%	100%
Seguridad	100%	100%
Sesiones	100%	100%
Total:	720%=90%	800%=100%

Fuente: Autor

Para una mayor apreciación se ha tomado el número de valoración que se le ha dado a cada parámetro para que sea de mayor comprensión para el lector en el cual se escribe el valor según lo estimado en la escala de 1 a 5 en la escala de Likert para valorar los parámetros que el autor lo requiera y de acuerdo a las necesidades que se haya tenido al momento de querer dar solución

al problema encontrado actualmente en la empresa, o a la vez en la necesidad que tenga el usuario.

Para ello a continuación se presenta la tabla que contiene dichos ítems ya con su calificación dada según la investigación que se ha realizado para obtener la mejor herramienta de las dos que se ha tomado en cuenta para su investigación, en esta se muestra la calificación en valores o números según la escala de Likert:

Tabla 12: Tabla de Resultados según la escala de Likert

Parámetros	Ruby On Rails	Django
Aprendizaje	4	5
Portabilidad	5	5
Documentación	4	5
Soporte	5	5
Plantillas	4	5
Vistas Dinámicas	5	5
Seguridad	5	5
Sesiones	5	5
Total:	37=90%	40=100%

Fuente: Propia

Así obtenemos el resultado de cada uno de los parámetros y por ende se logra la obtención de una herramienta ganadora en este caso como se puede observar el ganador el software de desarrollo Django, el cual obtiene un porcentaje de 100% de acuerdo a los ítems que se le ha planteado, según las necesidades que hacen referencia a lo que se necesita para el mejor desarrollo de un sistema de Network marketing.

CAPÍTULO 4

4. IMPLEMENTACIÓN DEL SISTEMA

Una vez realizado al análisis correspondiente de las herramientas y tomando en cuenta cada ítem que se le ha dado para saber su resultado frente a otra herramienta hemos obtenido un ganador y ahora se procede con la realización de la aplicación.

Para realizar la implementación del sistema se va usar la metodología Web llamada OOHDM (Método de Diseño de Hipermedia Orientado a Objetos).

Esta metodología está orientada a lo que es el diseño de aplicaciones hipermedia y para la Web, está convirtiéndose en la metodología más usada para el diseño de aplicaciones de diferentes tipos tanto como son las galerías que son interactivas, los famosos sitios web y las presentaciones multimedia, que se encuentran en auge total en las redes de internet.

La metodología OOHDM es inspirada en el modelo conocido como HDM (Modelo de Diseño de Hipermedia), el cual separa los aspectos globales y estructurales de una aplicación.

Pero se distingue por su orientación a objetos, además ésta metodología tiene 4 fases o etapas las cuales son:

- Diseño Conceptual
- Diseño Navegacional
- Diseño de Interfaces Abstractas
- Implementación

Cada una de las etapas o fases definen un cierto modelo o esquema específico en el cual se introducen las clases o los nuevos elementos.

Diseño Conceptual.- en esta etapa se realiza una construcción de un esquema conceptual el cual es representado por las clases de dominio llamados objetos y también las relaciones entre dichas clases.

En esta etapa lo más usado es un modelo de entidades y relaciones llamado modelo de datos semántico. El modelo de metodología OOHDM (Método de Diseño de Hipermedia Orientado a Objetos), propone un esquema basado en clases, relaciones y subsistemas, llamado también esquema conceptual.

Diseño Navegacional.- en esta etapa se define todas las clases navegacionales así como son los nodos y los enlaces. Por otro lado también se tiene las estructuras índices y visitas las cuales son guiadas.

Los enlaces no son sino los que se derivan de las relaciones que existen y los nodos son la representación de cada una de las ventanas lógicas o llamadas views. El diseñador realiza una descripción de la estructura navegacional de manera clara y con términos llamados de contexto navegacionales. OOHDM (Método de Diseño de Hipermedia Orientado a Objetos), propone un diseño o modelo muy enriquecido para el dominio de lo que es la aplicación.

El modelo de hipermedia sin embargo se define en un nivel llamado nivel de abstracción y consta de dos partes que son: las clases navegacionales y los contextos navegacionales.

Diseño de Interfaces Abstractas.- como su nombre lo indica esta interfaz está dedicada a la especificación de la interfaz abstracta.

Esta se define de una manera en la cual deberán aparecer los llamados contextos navegacionales. En esta etapa se realiza un tipo de descripción de los elementos u objetos que tiene la interfaz o view, y se los asociará con los objetos de navegación desde luego que deberían ir con cada una de las funcionalidades que tenga cada objeto.

Si se separa el diseño navegacional y lo que sería el diseño de interfaz abstracta permitirá construir dos interfaces diferentes para el modelo navegacional.

Implementación.- esta cuarta etapa es la dedicada a poner en práctica todos los objetos de interfaz con los que cuenta la aplicación.

Estos serían, sus relaciones entre esquemas siendo estos los conceptuales, navegacionales y los objetos que son de interfaz en el modelo OOHDM (Método de Diseño de Hipermedia Orientado a Objetos), que propone un diseño o modelo muy enriquecido para el dominio de lo que es la aplicación.

Este método es un modelo abierto porque los modelos de dominio no vienen impuestos y los de soporte permiten la especialización de las clases y los contextos navegacionales.

4.1 Fase de diseño conceptual

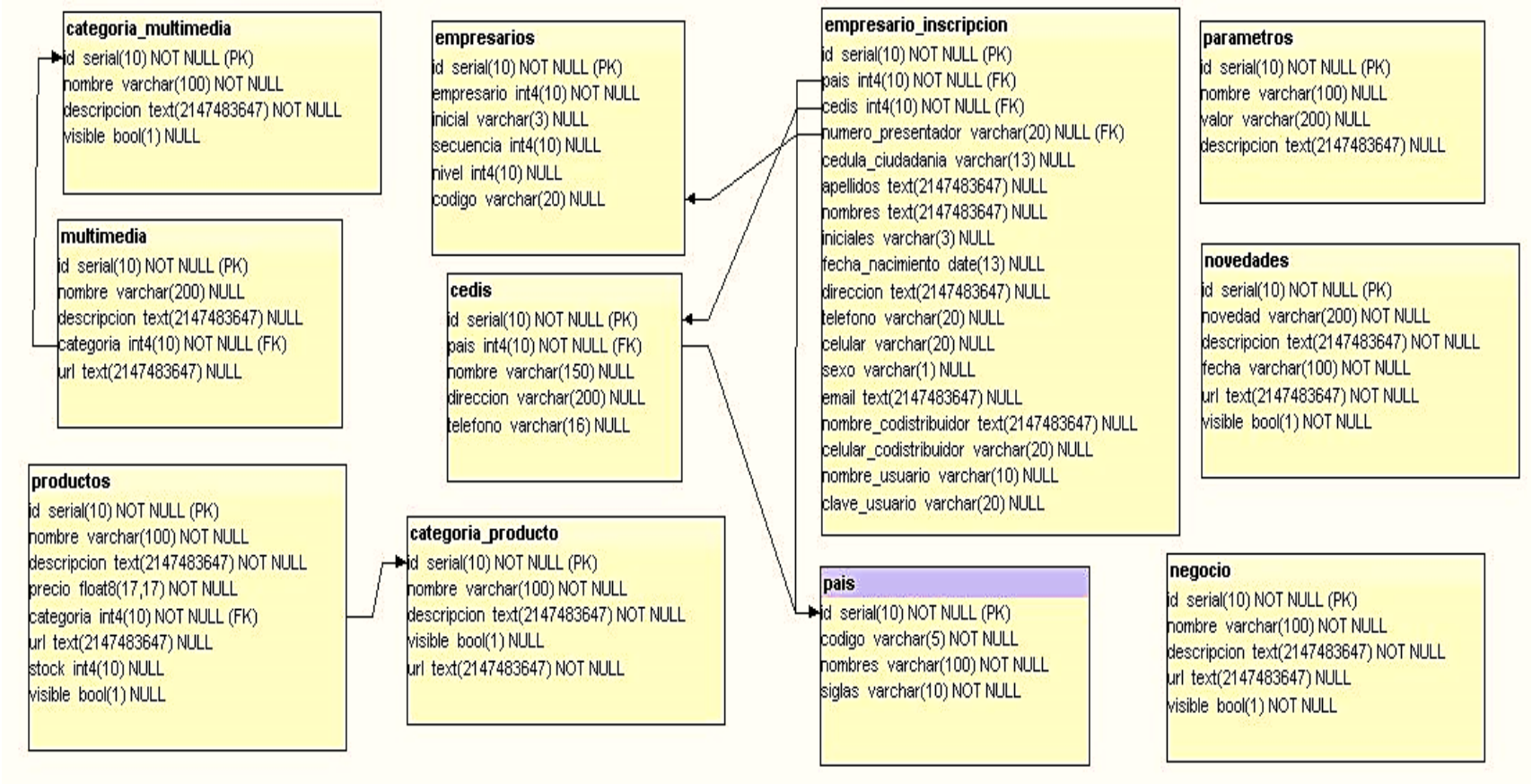


Gráfico 16: Diseño Conceptual de la Aplicación
Fuente: Propia

Explicación breve: en el modelo de la base de datos se refleja cada una de las clases conocidas también como objetos las cuales se relacionan entre sí para poder ayudar a la realización del software o aplicación de administración de Network Marketing.

4.2 Diseño Navegacional

Esquemas de clase navegacional

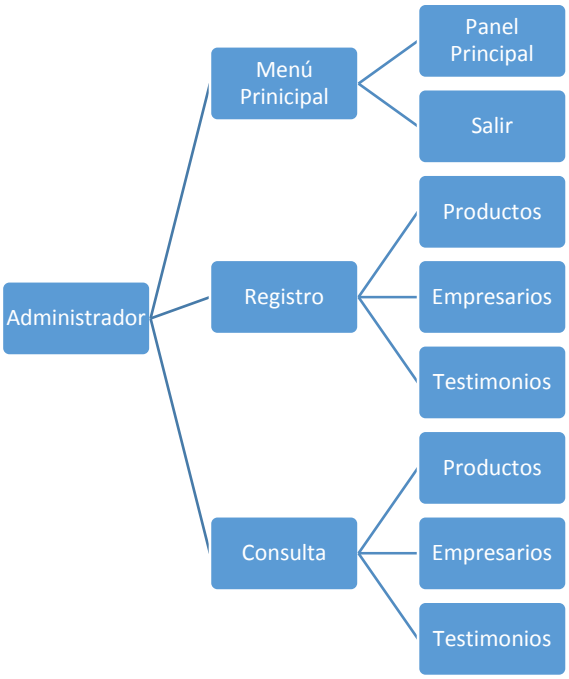


Gráfico 17: Modelo Navegacional del Administrador
Fuente: Propia

Explicación breve: el modelo navegacional no es más que el diseño gráfico de las funciones que puede realizar el administrador con respecto a la aplicación. Su administración, su funcionamiento, su modificación e incluso eliminación de la misma. Esto para una mayor comprensión del usuario o lector en este caso, para que posteriormente comprenda la nueva metodología Web.

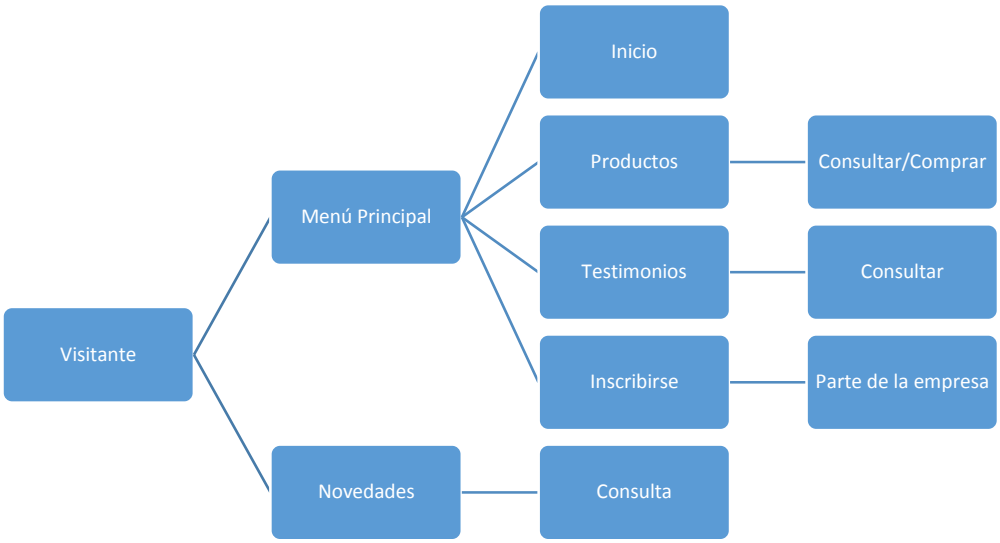


Gráfico 18: Modelo Navegacional del Cliente
Fuente: Propia

Explicación breve: el modelo navegacional del usuario o visitante explica las opciones que puede revisar, podrá hacerlo desde cualquier lugar, además de esto existe la opción de poder formar parte de la empresa, esto lo realiza en la opción de inscribirse la cual pedirá todos sus datos y explicará en que consiste dicha opción, así como también los beneficios de pertenecer a la empresa.

Esquema de Contextos Navegacionales

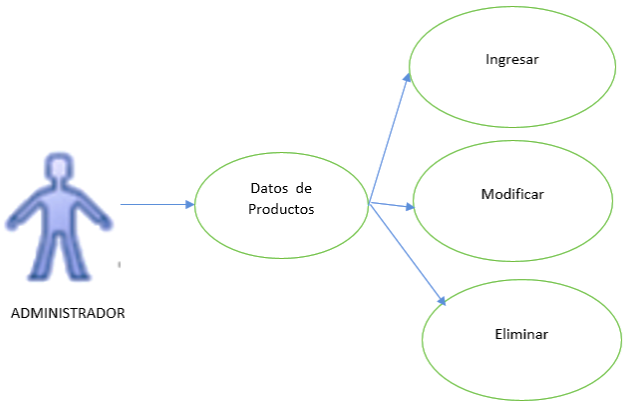


Gráfico 19: Diagrama de registro de Productos
Fuente: Propia

Explicación Breve: el diagrama de registro de productos, es donde el administrador del sistema web podrá ingresar, modificar, eliminar productos del registro de su base de datos, en caso de ser requerido. Por otro lado también son funciones que tienen ciertos privilegios no todos los usuarios podrán realizar dichas opciones, a menos que se les conceda por parte del administrador la opción de tener acceso a la administración en sí de todo el sistema web.

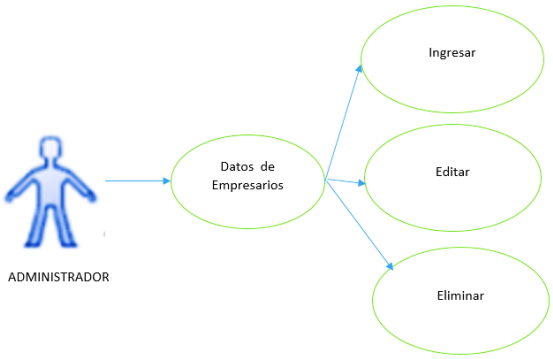


Gráfico 20: Diagrama de Datos de empresario
Fuente: Propia

Explicación Breve: el diagrama de registro de empresarios, es donde el administrador del sistema web podrá ingresar, modificar, eliminar empresarios del registro de su base de datos, en caso de ser requerido.

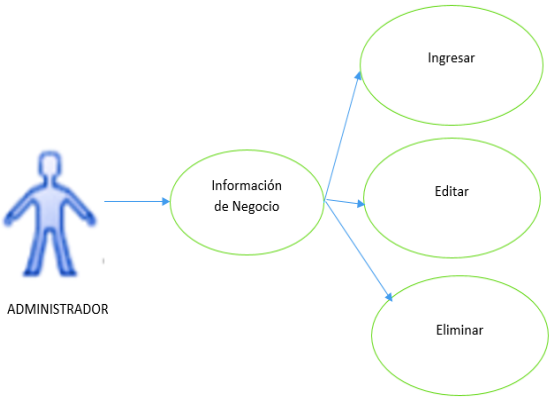


Gráfico 21: Diagrama de información de negocio
Fuente: Propia

Explicación breve: el diagrama de información de negocios nos indica que el administrador podrá ingresar, editar, o incluso eliminar información acerca del negocio, ya sea para actualizar

la información, acerca de los bonos o incluso de las modificaciones que lleve a cabo la empresa en cuanto a concursos se refiere y la manera de obtener las ganancias.

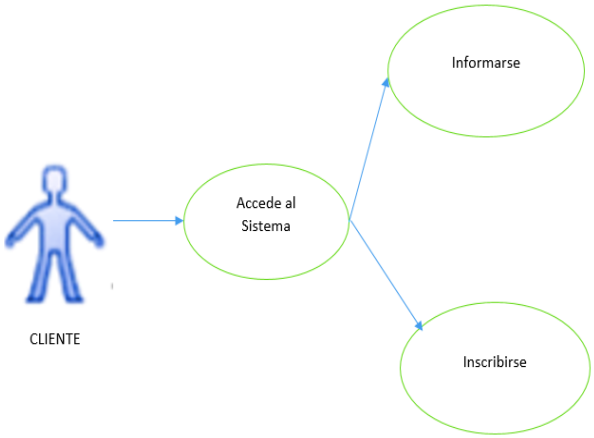


Gráfico 22: Diagrama de acceso al sistema como cliente
Fuente: Propia

Explicación breve: el cliente tendrá un acceso al sistema en el cual solo podrá informarse de lo que ofrece la empresa, además de esto también tendrá la opción de inscribirse y formar parte de la empresa. Todo esto sin ninguna obligación, si la persona o cliente que ingrese a la página y se informe de manera correcta decide pertenecer a la misma pues bienvenido sea.

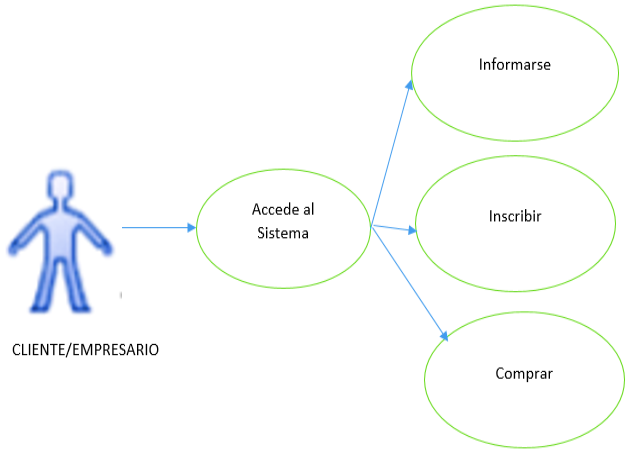


Gráfico 23: Diagrama de acceso al sistema como cliente Empresario
Fuente: Propia

Explicación breve: al momento de ya pertenecer a la empresa es un cliente Empresario, el cual podrá informarse de los productos y sus cantidades en stock, además de ello también podrá inscribir a nuevos empresarios o clientes que quieran formar parte de la empresa. Una vez que sea miembro de la empresa el cliente podrá realizar las compras de los productos que ofrece la misma.

4.3 Diseño Interfaz Abstracta

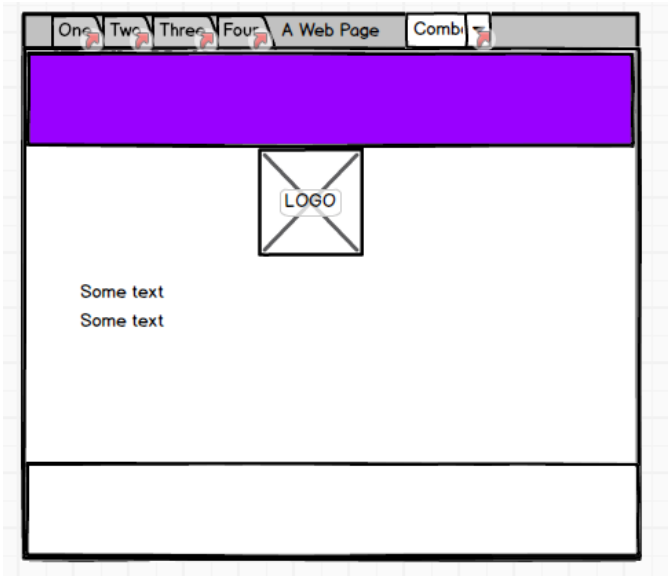


Gráfico 24: Fase de Implementación Pantalla de Inicio
Fuente: Propia

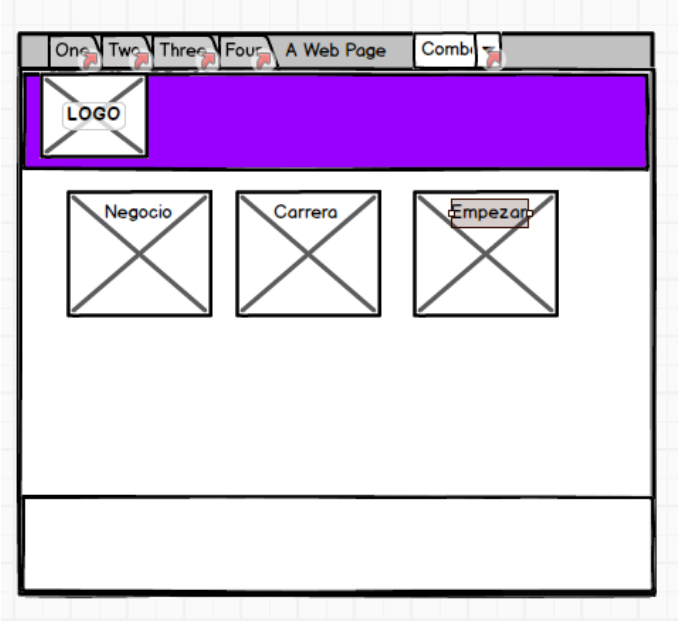


Gráfico 25: Fase de Implementación Abstracta Pantalla de Negocio
Fuente: Propia

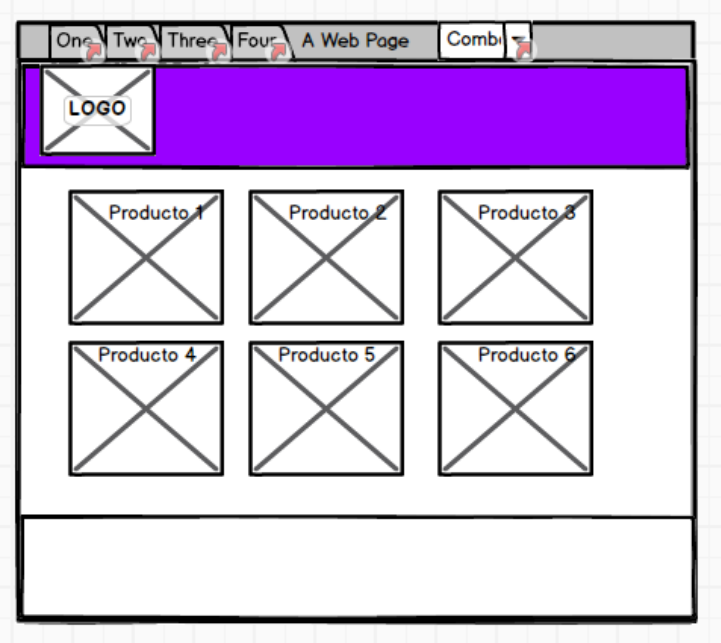


Gráfico 26: Fase de interfaz abstracta Pantalla de Productos
Fuente: Propia

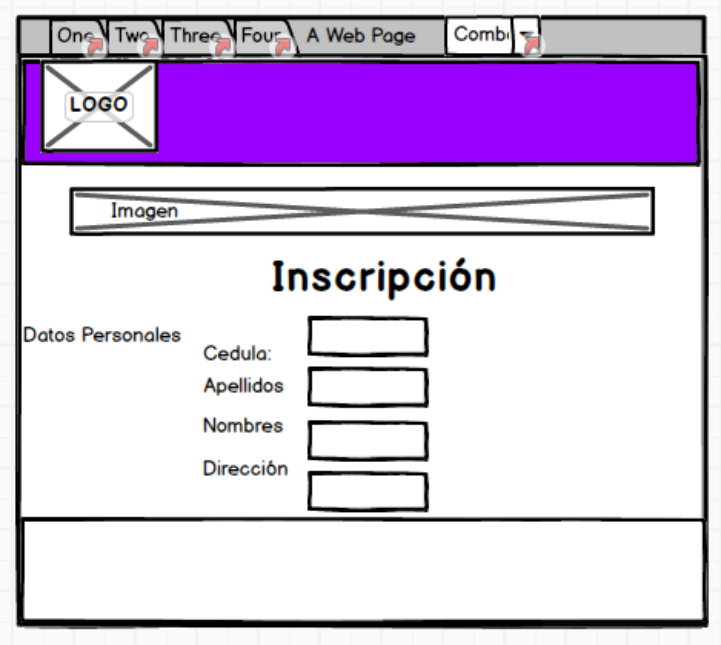


Gráfico 27: Fase de interfaz abstracta Pantalla de Inscripción
Fuente: Propia

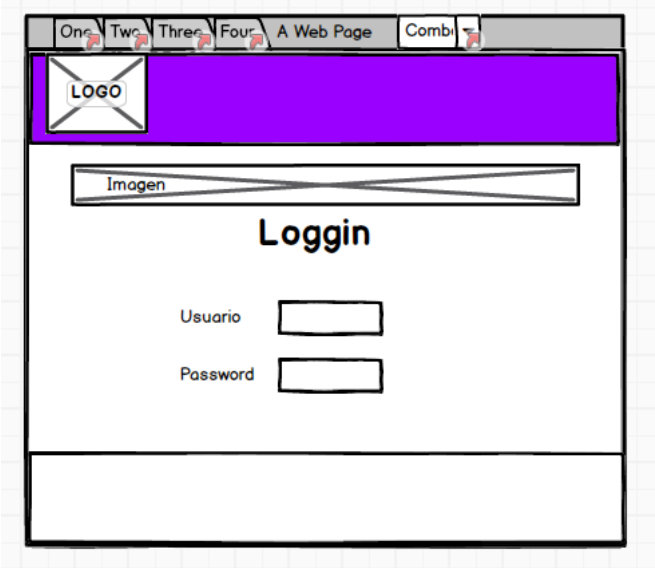


Gráfico 28: Fase de interfaz abstracta Pantalla de Ingreso de Usuarios.
Fuente: propia

Explicación Breve: en la fase de implementación abstracta se crea un bosquejo general de las pantallas que se va a usar en el sistema web, esto es una idea de cómo se puede presentar la vista para el usuario.

Este servirá de ayuda debido a que se debe tener una idea clara antes de realizar la implementación, encontrando la mejor manera de poder llegar al usuario y que sea de su completo agrado a la hora de visitar el sitio web.

Para ello pues se realiza una presentación en cuanto a los elementos se refiere, como se lo puede apreciar, así pues constará de una cabecera, de un contexto, imágenes, que servirán de información y comunicación con el usuario.

Todo esto de manera dinámica y entretenida dando una mejor presentación de la empresa a los usuarios o clientes.

4.4 Implementación

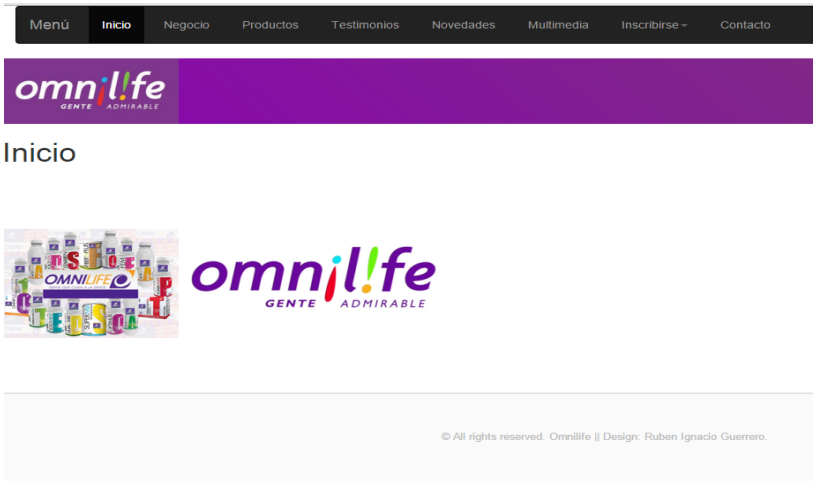


Gráfico 29: Pantalla de inicio de la Aplicación
Fuente: Propia

Expilcación Breve: La vista o pantalla de inicio presenta la información de lo que es la empresa su misión, visión y objetivos.

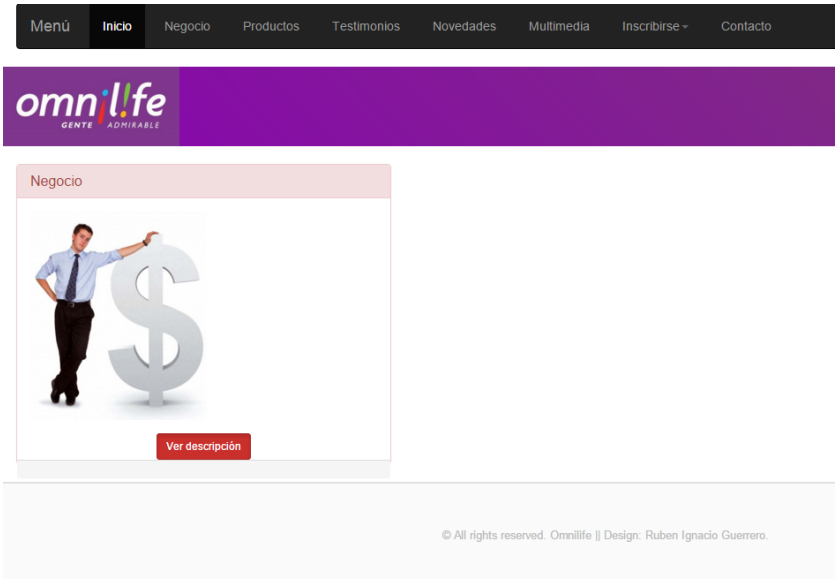


Gráfico 30: Vista explicativa del Negocio
Fuente: Propia

Explicación breve: la pantalla de negocio contiene toda la información de como se realiza las acciones de la empresa en cuanto a trabajo se refiere, aquí se explica la temática de como realizar el trabajo y como se obtiene las ganancias una vez que se pertenece a la entidad.

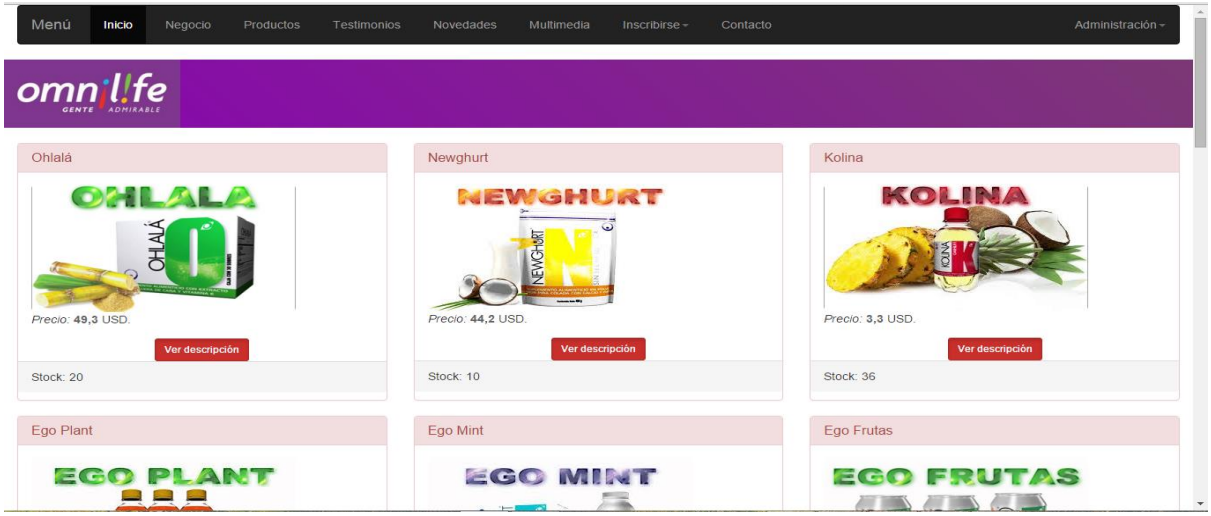


Gráfico 31: Vista de productos
Fuente: Propia

Explicación Breve: La vista de productos contendrá información clara de cada elemento para mejor información del usuario, en esta ventana existe también la opción de que una vez que se haya informado de lo que realiza cada producto lo pueda adquirir.

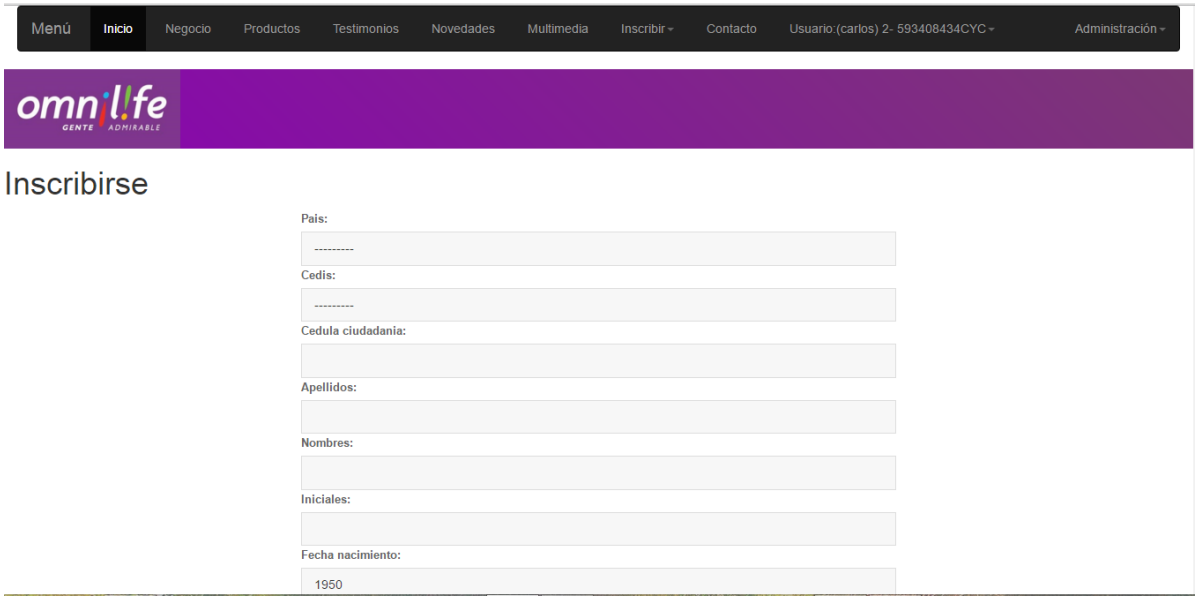


Gráfico 32: Vista de Inscripciones
Fuente: Propia

Explicación breve: La vista de inscripción será utilizada por los usuarios que formen parte ya de la empresa, para que puedan inscribir nuevos miembros a su red.

Gráfico 33: Vista de autenticación de usuarios
Fuente: Propia

Explicación Breve: La pantalla de Loggin se presentará luego de que el usuario o la persona haya decidido formar parte de la empresa, dando clic en la pestaña de inscripción y haya llenado los campos con sus datos. Una vez que realiza esos pasos, podrá obtener un usuario y una clave para administrar su sistema de Network Marketing.

Gráfico 34: Vista administrativa para cada Empresario
Fuente: propia

Explicación Breve: en la pantalla de administración una vez que el usuario pertenezca ya a la empresa, podrá ingresar datos de sus empresarios, de productos que el disponga, así como también modificarlos, o eliminarlos si es de su conveniencia.

CAPÍTULO 5

5. ANÁLISIS DE IMPACTOS, CONCLUSIONES Y RECOMENDACIONES

5.1 Análisis de Impactos

- **Área económica:** La empresa da a conocer a sus clientes los servicios que ofrece mediante el aplicativo realizado de esta manera mejora sus ingresos tanto en ventas, como en red. El ahorro y mejoramiento de inversión de capital en publicidad se verá mejor invertido, debido a que no se realizará gastos innecesarios en el tema de cómo llegar al público mediante impresiones de hojas volantes, anuncios en radio, comercio, ya que el aplicativo estará en línea las 24 horas del día, los 365 días del año.
- **Área del medio ambiente:** Un impacto muy fuerte es el tema de ahorrar, tiempo y espacio, ya que se usará solo las impresiones en casos necesarios, con esto se logrará obtener un ahorro de papel ya que en la actualidad lo que se pretende es evitar terminar con los pulmones de la tierra que en sí son los árboles, además evitar tener demasiados papeles que compliquen llevar un control de calidad de cada una de las opciones o tareas que realiza el aplicativo.
- **En el ámbito o área social:** Será una muy buena opción para generar trabajo, ya que toda persona puede llegar a formar parte de la empresa, con esto se estaría contribuyendo al desarrollo tanto intelectual, como también económico de la ciudadanía en general. En tal manera que las personas realizarán actividades que mejoren sus ingresos y su calidad de vida, porque también conocerán como llevar una vida llena de salud, contribuyendo también a mantener su desarrollo como ser humano, en una calidad que pocos realmente la conocen.

5.2 Conclusiones

- La realización e implementación del sistema de control de Network Marketing, permite de manera eficiente tener una exactitud en cuanto a ganancias se obtiene, para mayor superación de la empresa y además mejorará de forma notable la comunicación con los empresarios que estén dentro de la red.
- Una vez realizado este sistema y estar en funcionamiento en la Web, se obtendrá una ganancia en comunicación con el público debido a que estará disponible para todos, en cualquier momento para que se puedan informar acerca de los beneficios.
- La aplicación de una nueva metodología aplicada a la Web permite tener otro punto de vista en cuanto a tecnologías y metodologías se refiere, ya que se sale de los esquemas a los que se está acostumbrados. Esto ayudará a tener una mayor facilidad de comprensión de las aplicaciones Web y sus procedimientos.
- Concluida la aplicación resulta notorio el ahorro de ciertos recursos tales como: tiempo, dinero, papel y el trabajo de llegar a muchas personas. Esto es una manera de mejorar tanto la comunicación, los pedidos, las ventas y de esta manera se beneficiará la empresa ya que tiene la opción de realizar nuevas tareas de mejor manera.

5.3 Recomendaciones

- Mantener una adecuada infraestructura con respecto a los equipos informáticos, con su debido mantenimiento, para de esta manera evitar, que el sistema falle o se vuelva lento al realizar sus procesos.
- Realizar respaldos de manera periódica, de esta manera se evitará pérdida de información de la empresa, en cualquier situación hostil que se presente.
- Actualizar de acuerdo a las políticas de la empresa, los valores tanto de productos, como los sucesos importantes que se puedan presentar en el transcurso de existencia de la misma, de esta manera se logrará brindar un servicio de calidad a los usuarios.
- Tomar este tema de tesis para continuarlo y mejorarlo, orientado siempre a servir a la gente, de esta manera cuidando la salud y el bienestar económico de cada uno de los habitantes del país y a futuro del mundo entero. Con esto se logrará que las personas dejen de vivir cegadas por algo que se ha impuesto hoy por hoy en el mundo de la medicina.

GLOSARIO

Caché.- componente que almacena datos para que los futuros requerimientos puedan ser servidos con mayor velocidad.

Cookie.- Es una pequeña información enviada por un sitio web y almacenado en el navegador del usuario, de manera que el sitio web puede consultar la actividad previa del usuario.

Framework.- Es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado, el framework es un marco de aplicación o conjunto de bibliotecas orientadas a la reutilización de todos los componentes de software pero a gran escala de tal manera que el desarrollo de las aplicaciones sea eficaz.

Plugins.- Un plugin es un programa o incluso una aplicación que añade cierta funcionalidad al programa principal en donde este se encuentra hospedado.

Script.- El script es un conjunto de instrucciones que generalmente se encuentran almacenadas en un archivo de texto, estos son interpretados línea a línea en tiempo real para su ejecución.

URL.- Uniform Resource Locator (Localizador Uniforme de Recursos), es una dirección que permite acceder a un archivo o recurso tales como: páginas web ya sea su extensión .html, .php, .asp, o archivos de imágenes.

Template.- Es una plantilla que permite guiar, portar, o construir, un diseño o esquema que ya se encuentra predefinido, optimizando tiempo y recursos.

Clases.- Las clases en programación son un mecanismo diseñado para asemejarnos a los objetos de la vida real. En Python podemos decir que sirve para definir tipos por el usuario. Python define sus clases con la palabra reservada “class” seguida de su nombre.

BIBLIOGRAFÍA

Libros

Arias Ángel. *Aprende a Programar con Ruby on Rails.* (2014). RC Libros.

Hinojosa Gutierrez Angel Pablo. *Phyton paso a paso.* (2016). RA-MA.

Rerez Castaño Arnaldo. *Phyton Fácil.* (2016). Marcombo, S.A.

Knowlton Jim. *Phyton: Crear, Modoficar, Reutilizar.* (2009). Anaya Multimedia.

Buttu Marco. *El Gran Libro de Phyton.* (2016). Marcombo S.A. .

Summerfield Mark. *Phyton 3.* (2009) Anaya Multimedia.

González R Patricia. *Programacion en Ruby on Rails.* (2013). (L. RC, Ed.)

Realpe Rosero Christian Fernando. *Análisis y estudio de Tecnología Ruby on Rails con bases de datos Postgres para aplicaciones Web 2.0. Aplicativo: Implementación del Portal Web 2.0 para la Mancomunidad de la Cuenca del Río Mira.* (2013). Ibarra: [TESIS] UTN.

Kiyosaki Robert, D. T. *El toque de midas.* (2012). Sin Editorial.

Sam, Thomas, & Dave. *Desarrollo Web con Rails.* (2009). Anaya Multimedia.

Ponce Moreno Santiago. *Ruby on Rails. Desarrollo práctico de aplicaciones web.* (2013). RC Libros.

Ponce Moreno.Santiago*Ruby on Rails: Desarrollo práctico de aplicaciones web.* (2013). RC Libros.

Ponce Moreno Santiago. *Ruby on Rails: Desarrollo práctico de aplicaicones web.* (2013). RC libros.

Chazallet Sebastien. *Phyton3: Fundamentos del Lenguaje.* (2015). ENI.

Lawrence Pfleeger Shari, E. Q. *Ingeniería de software: teoría y práctica.* (2012).Pearson Education.

Sprenger, S., & Kieran Hayes. *Ruby on Rails 3.1 Profundidad.* (2012). Dpunkt.Verlag.

Stefan Tennigkeit Michael Voigt. *Ruby on Rails 3.* (2010). Entwickler.Press.

Tate, B. A., & Curt Hibbys. *Ruby on Rails.* (2007). Analaya Multimedia.

Deepak Vohra. *Ruby on Rails para PHP y Desarrolldores Java.* (2012). Springer.

Shaw Zed A. *Aprenda a Programar con Phyton.* (2014). Analaya Multimedia.

WEB

Apress. (2016). *La Guía Definitiva de Django.* Consultado el 15 de Agosto de `2015, de La Guía Definitiva de Django: http://www.academia.edu/7698037/Django_La_guia_Definitiva_Django_Software_Foundation

Cristalab (2012). *Python en la web con Django (VI): web de administración*. Consultado el 20 de Octubre de 2015, de Python en la web con Django (VI): web de administración: http://www.cristalab.com/tutoriales/python-en-la-web-con-django-vi-web-de-administracion-c104286l/

Alvarez Miguel Angel (2014). *Qué es MVC.* Consultado el 6 de Agosto de 2015, de Qué es MVC: http://www.desarrolloweb.com/articulos/que-es-mvc.html

Gaitán, F. (2013). *Ruby on Rails, parte 2: Modelo Vista Controlador*. Consultado el 29 de Octubre de 2015, de Ruby on Rails, parte 2: Modelo Vista Controlador: http://fernando-gaitan.com.ar/ruby-on-rails-parte-2-modelo-vista-controlador/

Guerrero, M. B. (2013). *Ruby on Rails desde cero: Primeros pasos*. Consultado el 20 de Mayo de 2015, de Ruby on Rails desde cero: Primeros pasos: http://html5facil.com/tutoriales/ruby-on-rails-desde-cero-primeros-pasos/

Lapuente, M. J. (2013). *Modelo OOHDM.* Consultado el 5 de Noviembre de 2015, de Modelo OOHDM: http://www.hipertexto.info/documentos/oohdm.htm

LibrosWeb. (2006-2016). *El libro de Django 1.0.* Consultado el 4 de Marzo de 2016, de El libro de Django 1.0: http://librosweb.es/libro/django_1_0/

LibrosWeb. (2006-2016). *Hola Rails.* Consultado el 20 de Septiembre de 2015, de Hola Rails: http://librosweb.es/libro/introduccion_rails/capitulo_4.html

LibrosWeb. (2006-2016). *Introducción a Ruby on Rails.* Consultado el 5 de enero de 2016, de Introducción a Ruby on Rails: http://librosweb.es/libro/introduccion_rails/

Picca Carlos. (2013). *Django desde Cero: Vistas Dinámicas*. Consultado el 15 de Diciembre de 2015, de Django desde Cero: Vistas Dinámicas: http://codehero.co/django-desde-cero-vistas-dinamicas/

Sin Autor. (2012). *Deshacer cambios en git*. Consultado el 17 de Mayo de 2015, de Deshacer cambios en git: https://mxrubyonrails.wordpress.com/page/2/

Llauradó Oriol. (2014). *La Escala de Likert: Qué es y cómo utillizarla*. Consultado el 25 de Diciembre de 2015, de La Escala de Likert: Qué es y cómo utillizarla: http://www.netquest.com/blog/es/la-escala-de-likert-que-es-y-como-utilizarla/

ANEXOS

ANEXOS

CONTENIDO
➢ Manual Técnico.
➢ Manual de Usuario.
➢ Scripts, entre otros.
DISPONIBLE EN EL CD