DESARROLLO DE UN FRAMEWORK ORIENTADO A LA WEB BASADO EN LOS PATRONES DE DISEÑO MVC Y DAO EN EL LENGUAJE DE PROGRAMACIÓN PHP

JAIME LEONARDO RICO GUEVARA

JHON FREDDY RONDON BETANCOURT

TUTOR

LUIS EDUARDO PÉREZ PEREGRINO

CORPORACIÓN UNIVERSITARIA MINUTO DE DIOS

FACULTAD DE INGENIERÍA

TECNOLOGÍA EN INFORMÁTICA

BOGOTA

2013

DESARROLLO DE UN FRAMEWORK ORIENTADO A LA WEB BASADO EN LOS PATRONES DE DISEÑO MVC Y DAO EN EL LENGUAJE DE PROGRAMACIÓN PHP

JAIME LEONARDO RICO GUEVARA

JHON FREDDY RONDON BETANCOURT

Trabajo de Grado presentado como requisito para optar al título de

Tecnólogo en informática.

CORPORACIÓN UNIVERSITARIA MINUTO DE DIOS

FACULTAD DE INGENIERÍA

TECNOLOGÍA EN INFORMÁTICA

BOGOTA

2013

Nota de aceptación

_____

_____

_____

_____

_____

_____

_____

Firma del presidente del jurado

_____

Firma del jurado

_____

Firma del jurado

Bogotá D.C. Junio de 2013

DEDICATORIA

Dedicado a todas las personas cercanas, por su incondicional apoyo y confianza depositadas en nosotros. Todo este trabajo ha sido posible gracias a ellos, esperamos no defraudarlos y contar siempre con su valioso apoyo, sincero e incondicional.

.

AGRADECIMIENTOS LEONARDO RICO

Agradezco al desarrollo humano como sociedad, ya que gracias a la investigación y curiosidad innata ha sido posible avanzar no sólo como personas, sino como raza humana preocupada por comprender todo a su alrededor.

AGRADECIMIENTOS JHON RONDON

Agradezco a mi familia por su incondicional e inagotable apoyo no solamente durante esta etapa sino durante toda mi vida, por su extraordinaria calidez y unidad, por sus consejos y constante guía.

TABLA DE CONTENIDO

LISTA DE GRÁFICOS

Pág.

INTRODUCCIÓN

En el desarrollo de aplicaciones orientadas a la web es muy común contar con herramientas y extensiones de código libre, que permiten facilitar la construcción de proyectos a gran escala, la mayoría de Frameworks incorporan una estructura basada generalmente en el patrón de diseño MVC, sin embargo se especializan en las tecnologías abarcadas del lado del servidor también conocidas como Back-End, dejando con la responsabilidad al desarrollador encargarse de la parte Front del proyecto.

Generalmente estos Frameworks ofrecen al desarrollar una estructura jerárquica de directorios en la cual alojar los diferentes archivos que conforman la aplicación, como son los controladores, vistas y modelos. Un buen ejemplo es el Framework Codeigniter. No obstante el manejo que se les da a los modelos es muy limitado.

A partir de esto, el presente trabajo tiene como finalidad presentar un Framework que provea al desarrollador de una estructura ordenada y escalable para el proyecto, a través del patrón de diseño MVC, a su vez brindará herramientas que faciliten la construcción de código tanto del Back-End con el Front-End de la aplicación, minimizando los tiempos de desarrollo.

El Framework también implementa el patrón de diseño DAO para el acceso a datos en la capa de persistencia garantizando la escalabilidad del proyecto, permitiendo migrar la aplicación de un motor de base de datos a otro, este proceso se hace posible gracias al patrón DAO y FACTORY.

El Framework está construido bajo el lenguaje de programación PHP implementando los patrones arquitecturales MVC y DAO. A su vez hace uso del lenguaje de programación Javascript, y del conocido Framework JQuery.

A continuación se realiza una descripción más profunda acerca de las funcionalidades que tiene integrado el Framework.

1.1 Título del Proyecto

Desarrollo de un Framework Orientado a la web basado en los patrones de diseño MVC y DAO en el lenguaje de programación PHP.

1.2 Planteamiento del Problema

Hoy en día el desarrollo de aplicaciones, requieren una gran inversión de tiempo, esfuerzo y costos. A partir de este problema se han venido desarrollando herramientas de trabajo denominadas Frameworks que faciliten los procesos rutinarios de todo proyecto.

Un Framework se define como una aplicación o un conjunto de módulos que permiten el desarrollo ágil de aplicaciones mediante estructura, librerías y proveer al desarrollador funcionalidades integradas para hacer uso de ellas en cualquier momento y parte de la necesidad.

A partir de esto podemos encontrar una gran cantidad de Frameworks con distintos ámbitos específicos, ya sea para desarrollo de aplicaciones web, móviles o plataformas, en donde cada uno de ellos incorpora diferentes patrones de diseño según su enfoque, escalabilidad y rendimiento. Por ejemplo en cuanto al lenguaje de programación PHP los que más se destacan por sus múltiples características y rendimiento son: YII, CakePHP, Zend Framework, Codeigniter y Simphony.

Todos estos Frameworks tienen características especiales que los hacen únicos ya sea en la manera que se llevan a cabo los procesos en su estructura como puede ser el simple hecho de hacer una conexión a una base datos, como en la funcionalidad que incorporan y que brindan al desarrollador ya sea por medio de módulos de seguridad, de sesiones, etc. Sin embargo la mayoría de estos Frameworks presentan un problema común que está ligado a la curva de aprendizaje del desarrollador, ya que si bien es cierto que una de las

ventajas que genera usar un Framework es el ahorro considerable de tiempo para el proyecto, también se debe tener en cuenta que las primeras aplicaciones que se desarrollen en base al Framework seguramente se invertirá más tiempo ya que la estructura es nueva, la forma de usar sus componentes varia, etc., y todo esto influye en el resultado final.

Pensando en las dificultades nombradas anteriormente, se decide desarrollar un Framework como herramienta de trabajo que facilite los procesos rutinarios que se llevan a cabo en todo proyecto como son el control de sesiones, acceso a la capa de persistencia, seguridad, y dejando como única responsabilidad al desarrollador centrarse en el verdadero objetivo del proyecto. A la vez se necesita que el Framework sea sencillo en su forma de uso con lo cual se garantizara que su curva de aprendizaje sea relativamente corta.

1.3 Alcances y Justificación

El proyecto tiene como objetivo central generar una herramienta de trabajo (Framework), tanto para desarrolladores Back-End como Front-End, generando una curva de aprendizaje del mismo en un periodo de tiempo relativamente corto. Además busca ofrecer una completa solución de herramientas incorporadas facilitando llevar a cabo procesos para el desarrollo del aplicativo, a la vez permitirá al desarrollador realizar procesos que sin la herramienta pueden llegar a tomar tiempo o incluso llegando a ser complejos exponiendo la seguridad e integridad del proyecto.

El Framework permitirá llevar a cabo diferentes procedimientos a través de un módulo denominado “generator” el cual ofrece diferentes posibilidades para la creación de plantillas de código, que ahorran tiempo y proveen orden al sistema, teniendo en cuenta que el Framework abarca tanto la capa de Back-End como del Front-End del aplicativo, el modulo divide sus generadores en estas dos capas.

En la capa de Back-End el desarrollador tendrá la posibilidad de:

- Generar controladores, vistas, y modelos que representen las entidades de persistencia del sistema.

- Posibilidad de crear de forma sencilla y rápida los procesos correspondientes al CRUD (Create, Read, Update, Delete) de cada uno de los modelos, garantizando con esto que la aplicación contará con una sección estrictamente destinada a la administración de la información.

En cuanto a la capa de Front-End el desarrollador podrá:

- Diseñar visualmente de forma intuitiva el sitio, garantizando la compatibilidad con navegadores que cumplan los estándares de HTML, incorporando diseño adaptable.

El Framework inicialmente permitirá conectarse a dos motores gestores de bases de datos específicos a través del patrón DAO. El primero es un motor de base de datos relacional MySql, mientras que el otro es un motor de bases de datos NoSql MongoDB.

1.4 Objetivos

1.4.1 Objetivo general

Generar un Framework o herramienta de trabajo, orientada a la web, que permita crear proyectos de manera fiable, segura y controlada bajo un óptimo esquema basado en los patrones de diseño MVC, DAO.

1.4.2 Objetivos específicos

- Facilitar el desarrollo de Proyectos en la Web.
- Ahorrar el tiempo de desarrollo de un proyecto.

- Permitir la creación de Back-End y Front-End con auto generadores de código de forma práctica.
- Garantizar una estructura ordenada aplicando patrones de diseño manteniendo la escalabilidad e integridad del proyecto.

2. INGENIERÍA DEL PROYECTO

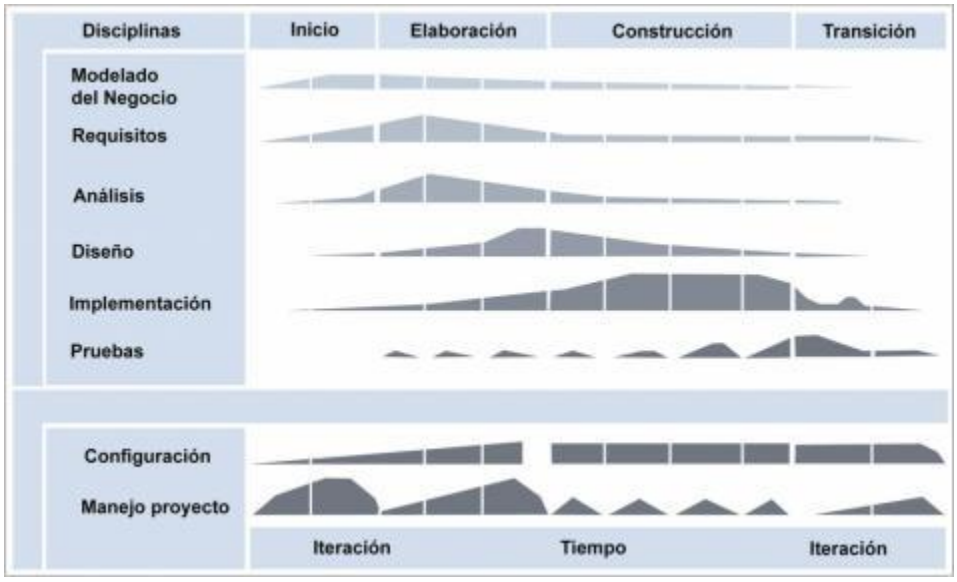
2.1 Modelo de Desarrollo

A continuación se detallará todo lo concerniente a la construcción del Sistema de Información propuesta. En un primer lugar se describirá la metodología de desarrollo a utilizar y luego los patrones de programación que se utilizaran durante la construcción de software.

La metodología a usar para el desarrollo de este proyecto es UP (Proceso Unificado), éste marco de desarrollo de software que se caracteriza por estar centrado en la arquitectura y por ser iterativo e incremental, esto resulta bastante útil para el proyecto, ya que es una aplicación evolutiva y su desarrollo debe evitar al máximo errores funcionales, esto permitirá continuar con otra fase de implementación.

Como se aprecia en la gráfica N° 1 el proceso Unificado divide el trabajo de desarrollo de Software en cuatro fases las cuales son Inicio, Elaboración, Construcción y Transición. Cada una de estas fases es a su vez dividida en una serie de iteraciones, estas iteraciones ofrecen como resultado un incremento del proyecto que añade o mejora las funcionalidades del sistema en desarrollo.

A la vez la figura representa cada una de las disciplinas utilizadas en el proceso de desarrollo de software y su nivel de participación en cada una de las fases definidas de UP.



Gráfica N° 1: Etapas de desarrollo del proyecto.

En la etapa inicial del proyecto, se establecen las necesidades que podrían tener los usuarios de la aplicación, en este caso serían los desarrolladores que usarán el Framework, con lo cual se generan los requerimientos del sistema, la parte gráfica del aplicativo basada en Wireframes, se definen los diferentes generadores de código que tendrá el Framework, y las librerías y funcionalidades que incorporará.

En la etapa de Elaboración, se determinaron los siguientes ítems:

- Se define el diseño del Framework a partir de la generación de los prototipos o Wireframes, teniendo en cuenta un diseño adaptable (Response Design).

- Se elaboran los diagramas UML para definir las clases, atributos, métodos, distribución, relaciones y funcionalidad que debe tener el sistema.

- Se elaboran los esquemas generales que muestran el comportamiento general del sistema.

- Se define el orden jerárquico en el cual estarán organizados los diferentes módulos y componentes que conforman el Framework.

En la etapa de Construcción, se realizaron los siguientes ítems:

- Se realizó la estructura jerárquica de directorios que conformaran el Framework basada en patrón de diseño MVC.

- Se desarrolló los principales módulos que integran el Framework como son el router, controladores, vistas y modelos.

- Se realizaron las pruebas correspondientes sobre los módulos, para comprobar su reutilización.

- Se integró como motor de plantillas la librería Smarty de PHP para el renderizado de la vistas.

- Para la capa de persistencia se implementó el patrón de diseño DAO, el cual facilita las gestiones correspondientes a esta capa, encargando del intercambio de información con el gestor de bases de datos.

- Se crearon los drives necesarios para la conexión a los motores de bases de datos MySql y MongoDB.

- Se desarrollaron los scripts encargados de generar las plantillas de código de las vistas, los controladores y los modelos.

- Se desarrolló e implemento el módulo encargado de generar las diferentes formas de visualización del sitio.

- Se integró un conjunto de librerías de código libre encargadas de facilitar tareas rutinarias, como son las sesiones, él envió de correo electrónico y creación de pdf.

- Se realizaron pruebas de errores para observar la funcionalidad del Framework y su rendimiento.

Durante la etapa de transición, se integraran más librerías, para generar más funcionalidades al sistema, siguiendo los mismos estándares, a través de pruebas de errores de la aplicación y en términos de tiempos de rendimiento y efectividad, observando la correcta funcionalidad para evitar falencias.

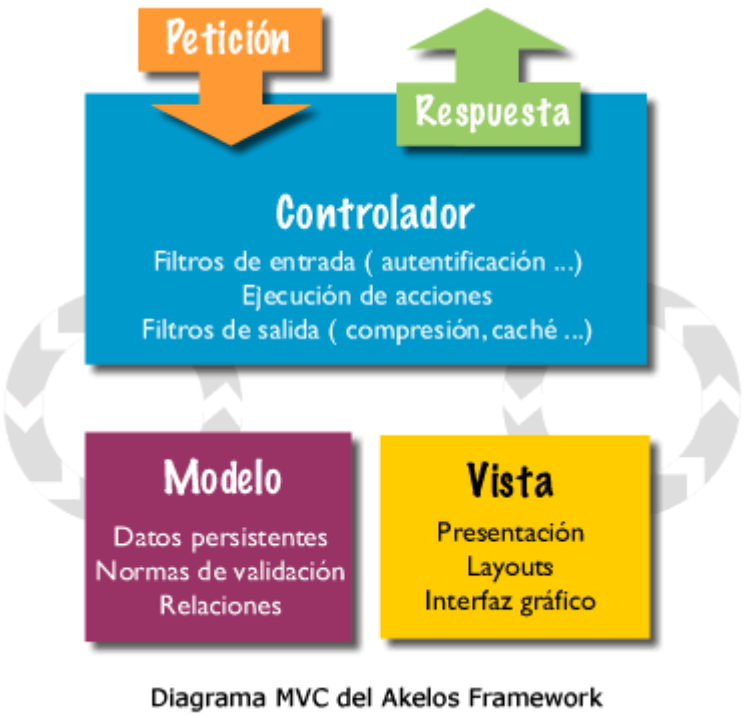
2.1.2 Patrones de Arquitectura

Un patrón es la solución a un problema de diseño de software y en la actualidad gracias a las características y ventajas que proporcionan los lenguajes de programación de alto nivel orientados a objetos los diseñadores implementan dichas características para crear soluciones frecuentes y reutilizables.

La arquitectura técnica del Framework está basado en el patrón de diseño MVC (Modelo Vista Controlador), sin embargo también integra otros patrones de diseño encargados de procesos específicos como por ejemplo el Patrón DAO (Objeto de acceso a datos) el cual actuará como nexo entre la lógica de negocio y la capa de persistencia (generalmente, una base de datos).

El patrón MVC ayuda a dividir la aplicación en capas lógicas, independientes unas de otras, las cuales dividen de manera eficaz la implementación, con lo cual el comportamiento de cada una de las capas es independiente a lo que pueda hacer la otra, con esto se garantiza que la escalabilidad de la plataforma.

Este patrón divide su estructura en 3 capas como se puede observar en el gráfico N° 2.



Gráfica N° 2: Patrón de Arquitectura Modelo Vista Controlador

- **Modelo**

Accede a la capa de almacenamiento de datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento, para esto se usará el patrón de diseño DAO.

- **Vista**

Es la parte encargada de la manipulación de los datos para representarlos gráficamente y mostrarlos como salida. El Framework incorporará una librería de código libre de PHP, que facilita el trato a datos en los templates de las vistas.

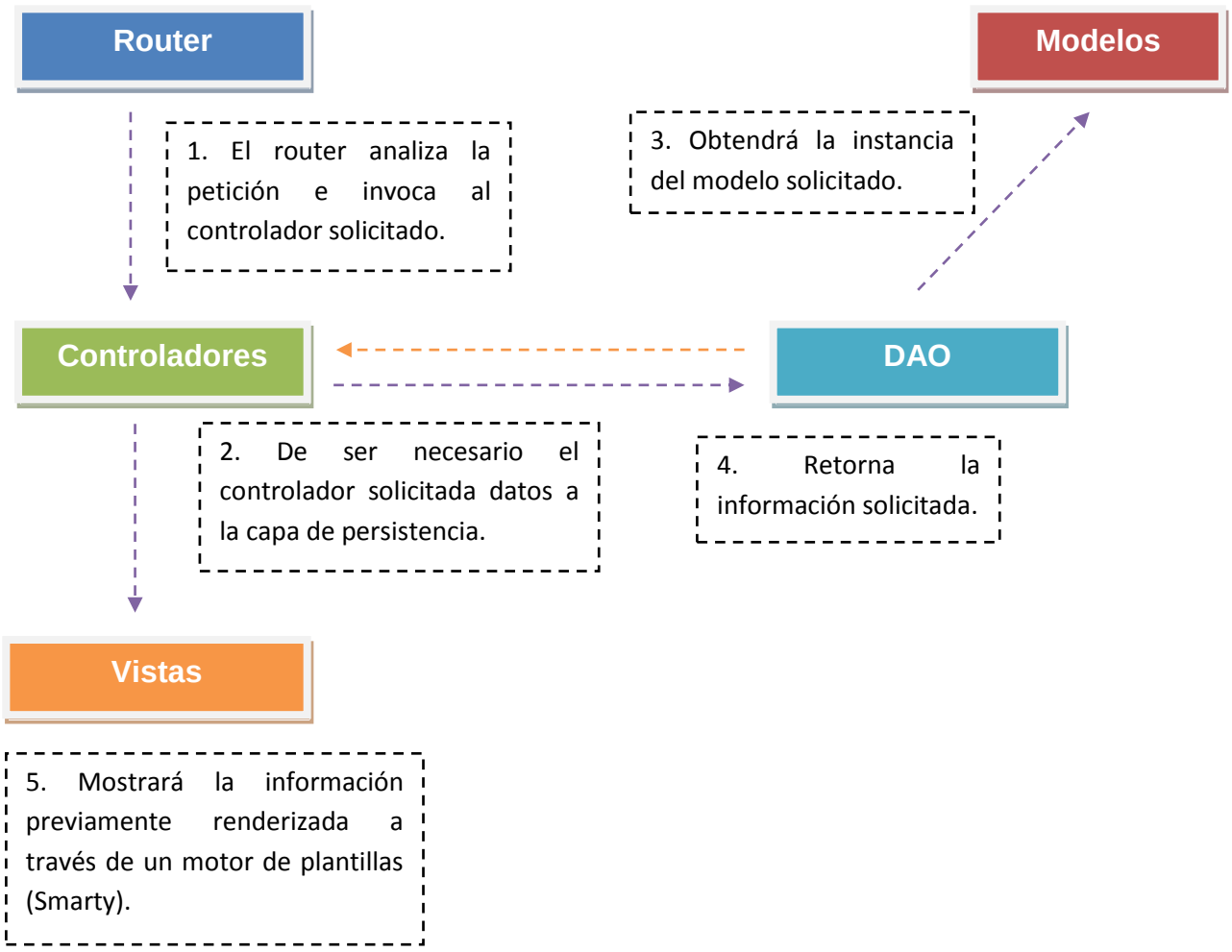
- **Controlador**

Interpreta las acciones del usuario informando al modelo y/o a la vista para cambiar apropiadamente sus estados.

2.1.3 Comportamiento del Sistema

A continuación veremos una descripción general del funcionamiento del proyecto:

En la gráfica N° 3 se podrá apreciar el proceso que realizará el Framework en base a una petición de un usuario. Primero el router recibe la url solicitada por el usuario, la analiza y elije el controlador encargado que coincide con dicha búsqueda. El controlador solicitará al modelo los datos solicitados que se encuentran en la base de datos de ser necesarios, una vez concluido el proceso el controlador pasara los datos a la vista para que esta finalmente los presente al usuario.



Gráfica N° 3: Comportamiento del Sistema

3. ANÁLISIS Y DISEÑO

El análisis y diseño es un método en el cual se realiza un modelo del sistema, mostrando sus interacciones a partir de los requerimientos solicitados y exigidos para llegar a una solución o satisfacer una necesidad, este proceso se lleva a cabo mediante la herramienta UML (Lenguaje Unificado de Modelado).

A partir de esto se hablará principalmente del análisis y diseño del Framework que es el objetivo principal de la realización de esta tesis. Se ilustrará claramente este proceso a través de diagramas de casos de uso y de clases entre otros utilizando el modelo UML. De igual manera se mostrará un diseño de la interfaz que se desarrolló a través de Wireframes. Por último también se podrá observar tanto los requerimientos funcionales y no funcionales dentro de la plataforma.

3.1 Definición de Requerimientos

Los requerimientos se definen como las actividades que el sistema deberá poder realizar los cuales se dividen en funcionales y no funcionales.

3.1.1 Requerimientos Funcionales

Los requerimientos funcionales se rigen en base a las características que el sistema deberá satisfacer y como deberá actuar el sistema frente a diferentes circunstancias o errores, a partir de esto los requerimientos funcionales establecidos para el Framework son los siguientes:

1. El Framework debe permitir generar plantillas de código de controladores, vistas y modelos a través del módulo “generator”.

2. El Framework debe ser capaz de generar los templates y scripts necesarios para las funciones del CRUD (*Create, Read, Update, Delete)* de cada tabla o entidad en la base de datos.

3. El Framework debe proveer al desarrollador la posibilidad de crear, modificar y eliminar sesiones de usuario.

4. El Framework debe permitir al desarrollador conectarse a un motor de base de datos ya sea relacional o NoSql, para este caso se tratarán sobre los motores MySql y MongoDB respectivamente.

5. El Framework debe soportar la creación de interfaces para las vistas de la aplicación.

6. El Framework debe permitir integrar librerías externas que faciliten el desarrollo de proyectos como por ejemplo PhpMailer, TCPDF, etc.

7. El Framework debe permitir configurar el proyecto de forma gráfica, especificando el nombre, ubicación, zona horaria, así como los datos de conexión a la base de datos.

8. El Framework debe permitir al desarrollador seleccionar los scripts y hojas de estilo (*CSS)* que se cargaran a la vista de forma dinámica.

9. El Framework debe proveer de funciones que faciliten los procesos llevados en el Front-End, como por ejemplo envió de información por medio de AJAX (*Asynchronous JavaScript And XML*).

10. El Framework deberá responder a las peticiones realizadas por el usuario a través de la URL, analizando e identificando que controlador es el solicitado.

11. El Framework debe permitir utilizar URL semánticas a través de un archivo .HTACCESS
12. El Framework debe permitir al desarrollador trabajar bajo el paradigma de programación orientada a objetos (*POO*) en lo que corresponde al manejo de la información con la base de datos, a través del Patrón DAO.

3.1.2 Requerimientos No Funcionales

Estos requerimientos definen restricciones adicionales con el que el sistema deberá contar, los requerimientos no funcionales de la plataforma son los siguientes:

1. El proyecto a desarrollar bajo el Framework debe estar montado en un servidor web o un servidor local como por ejemplo: XAMPP, WAMP, LAMP, etc.

2. El Framework debe proveer de una interfaz de usuario (GUI) para el módulo “generator”.

3. El Framework debe garantizar la escalabilidad del proyecto.

4. El Framework debe proveer al proyecto seguridad de su información, así como el control de acceso al proyecto.

5. El Framework debe manejar tiempos relativamente cortos de respuesta entre la solicitud y la visualización de la información.

6. El Framework estará diseñado para un fácil mantenimiento. Las funciones internas contarán con la documentación necesaria para realizar los cambios oportunos sin mayores complicaciones.

7. El sistema será fácil de usar, se le proporcionará al usuario mensajes de error y de advertencia.

3.2 Descripción del sistema propuesto

El Framework es una herramienta orientada a la web, que permite el desarrollo rápido de proyectos web, permitiendo que el desarrollador centre su trabajo en el verdadero problema del proyecto dejando a un lado los procesos rutinarios de todo proyecto. Esta construido bajo el patrón de diseño Modelo Vista Controlador (MVC), sin embargo también integra el patrón Objeto de Acceso a

Datos (DAO) que se encarga de la persistencia y acceso a la información y los patrones FACHADA y SINGLETON para obtener instancias de objetos únicas

El desarrollo del Framework está divido como toda aplicación web en dos capas Front-End, aquí las tecnologías de desarrollo y de diseño web utilizadas son HTML5, CSS, JAVASCRIPT y librerías que extienden a estos a estos como son JQuery.js, Undercore.js, Backbone.js; por otro lado el Back-End del Framework ha sido realizado con el lenguaje de programación PHP (*Hypertext Preprocessor*) que a pesar de no ser un lenguaje orientado a objetos, permite las técnicas de programación orientada a objetos.

El Framework será completamente libre y con alto grado de flexibilidad para el crecimiento del mismo, permitiendo a los desarrolladores la posibilidad de aumentar su funcionalidad dependiendo de sus propios requerimientos. Considerando estos factores la estructura del Framework se ha dividido de la siguiente manera:

- **application:**

Este directorio albergará toda la lógica de negocio del proyecto a tratar, en esta sección el desarrollador tendrá acceso a sus controladores, modelos, helpers y vistas. A su vez contará con 4 directorios los cuales brindaran un mayor orden y estructura al proyecto los cuales son:

  - **controllers**

  Directorio destinado a todos los controladores del proyecto, encargados de analizar y responder a las solicitudes del usuario ya sea por medio de la URL o una petición AJAX.

  - **helpers**

  En este directorio se alojaran todos los archivos que brindan funciones generalizadas que utilizan los controladores, con el fin de “No repetir código”.

  - **models**

  Por medio del Patrón DAO se llevará a cabo la organización de los modelos que representan las entidades de la base de datos, es así

como el directorio a su vez se dividirá en un conjunto de subdirectorios encargados de optimizar el manejo de la información sin importar el motor de base de datos utilizado. Los directorios son los siguientes:

- **DTO**

  Encapsula la información referente a las entidades de la base de datos.

- **DAO**

  A partir del DTO, extrae la información y construye la lógica necesaria para comunicarse con la fuente de datos.

- **Directorios dinámicos**

  En este punto el desarrollador deberá crear los directorios encargados del manejo de la información con los distintos motores de bases de datos que el proyecto necesite. Esta operación se facilitará a través del módulo “generator”.

- **views**

Albergará los layouts, templates y vistas presentes en el proyecto. En el caso de las vistas cada vista tendrá una estructura jerárquica de directorios relacionados únicamente a ella. Los directorios son los siguientes:

- **css**

  Directorio destinado a almacenar los archivos css de la vista.

- **js**

  Directorio que almacenará todos los archivos JavaScript que requiera específicamente la vista.

- **templates**

  Almacenará los diferentes templates creados con la extensión .tpl que utiliza la vista. Estos archivos son tratados por un motor de plantillas que tiene integrado el Framework llamado Smarty, el cual renderiza los contenidos de estor archivos a HTML.

- **core**

Como su nombre lo indica es el núcleo del Framework, el cual almacenará los principales archivos esenciales para correcto funcionamiento del Framework. Estos archivos se encargan de la configuración general del proyecto, control de sesiones, auto carga de clases, archivos base para los controladores y modelos. A la vez el core presenta subdirectorios encargados de configurar el funcionamiento del Framework, así como también un directorio encargado de archivar los errores presentados en el proyecto. Los directorios son los siguientes:

  - **log**

  Directorio que albergara los archivos de error presentados en ejecución.

  - **config**

  Almacena los archivos principales de configuración, como son las conexiones a la base de datos, configuración global del proyecto, y constantes predefinidas.

  - **dao**

  Mantiene los distintos drivers de conexión así como los archivos base, utilizados para los modelos del motor de base de datos.

  - **exceptions**

  Conjunto de clases que representa las diferentes excepciones que pueden lanzar los métodos en tiempo de ejecución.

  - **router**

  Conjunto de clases encarga de procesar las solicitudes realizadas por el cliente.

- **modules**

Directorio que contendrá los módulos del Framework, inicialmente el Framework contará con el módulo “generator”, sin embargo a partir de este se podrá crear un módulo encargado de la administración de las tablas para motores de base de datos relacionales denominado “administrator”.

- **generator**

  Directorio que contendrá la jerarquía de directorios y scripts necesarios para realizar las generaciones de código disponibles por el Framework como vistas, controladores, scaffold, etc.

- **libraries**

En este directorio se guardaran todas aquellas librerías importadas al Framework encargadas de tareas específicas como por ejemplo phpMailer, TCPDF, etc.

- **public**

Se almacenaran todos aquellos archivos correspondientes al Front-End del proyecto y que resultan ser generales para todas las vistas. La organización de directorios es la siguiente:

  - **css**

  Permitirá almacenar los archivos generales CSS para el proyecto, que por lo general son los encargados del diseño y maqueta del proyecto.

  - **js**

  Esta vez albergara archivos JavaScript que son generales para todo el proyecto.

  - **media**

  Su objetivo es albergar todos los archivos externos que las vistas necesiten como por ejemplo audio, imágenes, video, etc.

3.3 Diseño del sistema propuesto

En el diseño del Framework se ha optado por utilizar UML (Lenguaje unificado de Modelado), ya que es el lenguaje de modelado de sistemas que ofrece un estándar para describirlo, incluyendo aspectos conceptuales tales como procesos de negocio y funciones de la aplicación. A continuación se presenta el diseño general del Framework. A partir de esto se implementaran los siguientes diagramas que ayuden a apreciar la estructura y comportamiento del Framework.

3.3.1 Wireframes

Son la representación esquemática de un sitio o plataforma web sin elementos gráficos, que permiten visualizar el comportamiento del sitio, además actúan como herramienta de comunicación entre los programadores, diseñadores y clientes. A continuación se presentan los Wireframes realizados como diseño para el módulo “generator” del Framework.

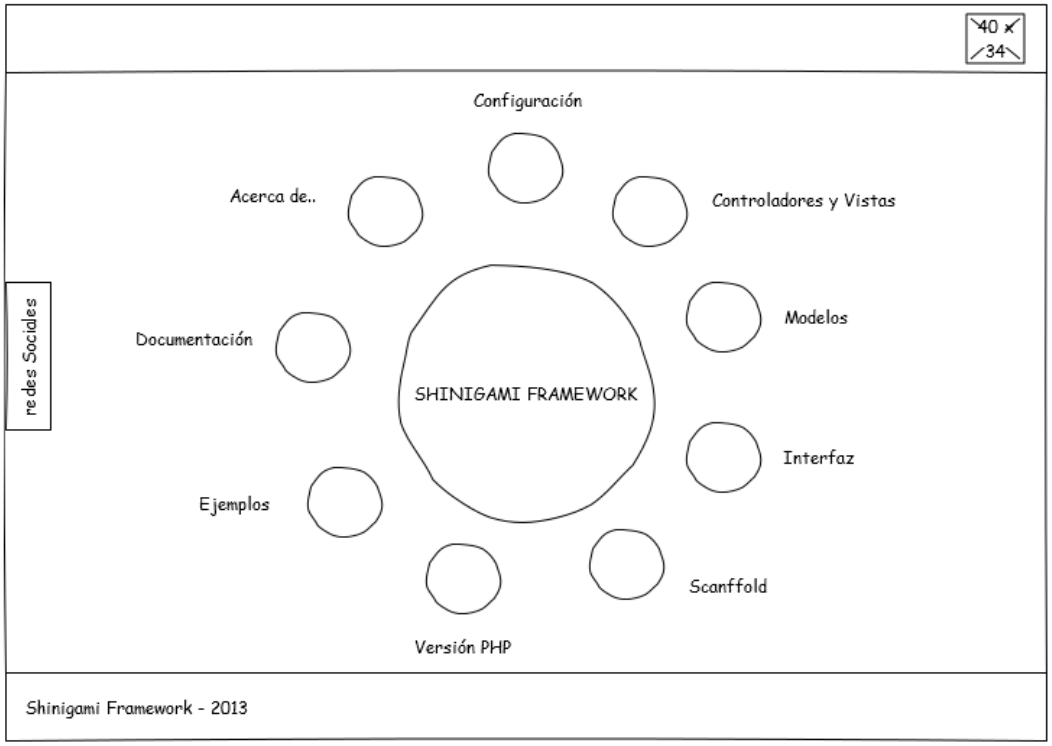


Gráfico N° 4: Diseño home del módulo “generator”

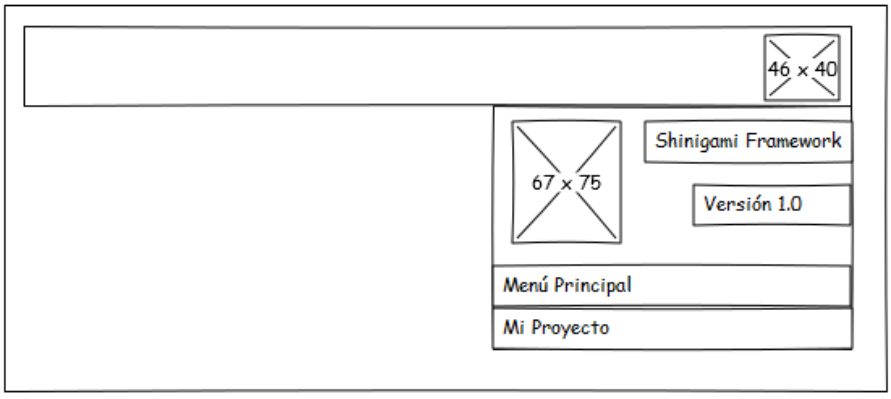


Gráfico N° 5: Diseño menú de inicio “generator”

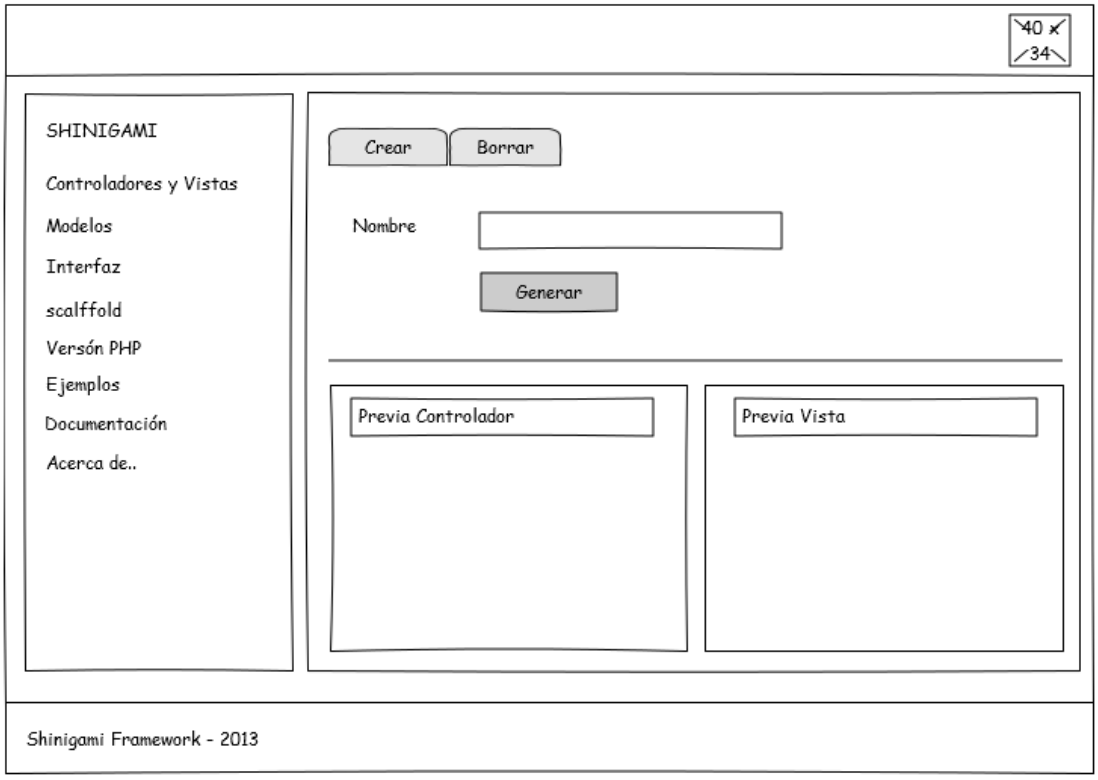


Gráfico N° 6: Diseño creación de vistas y controladores

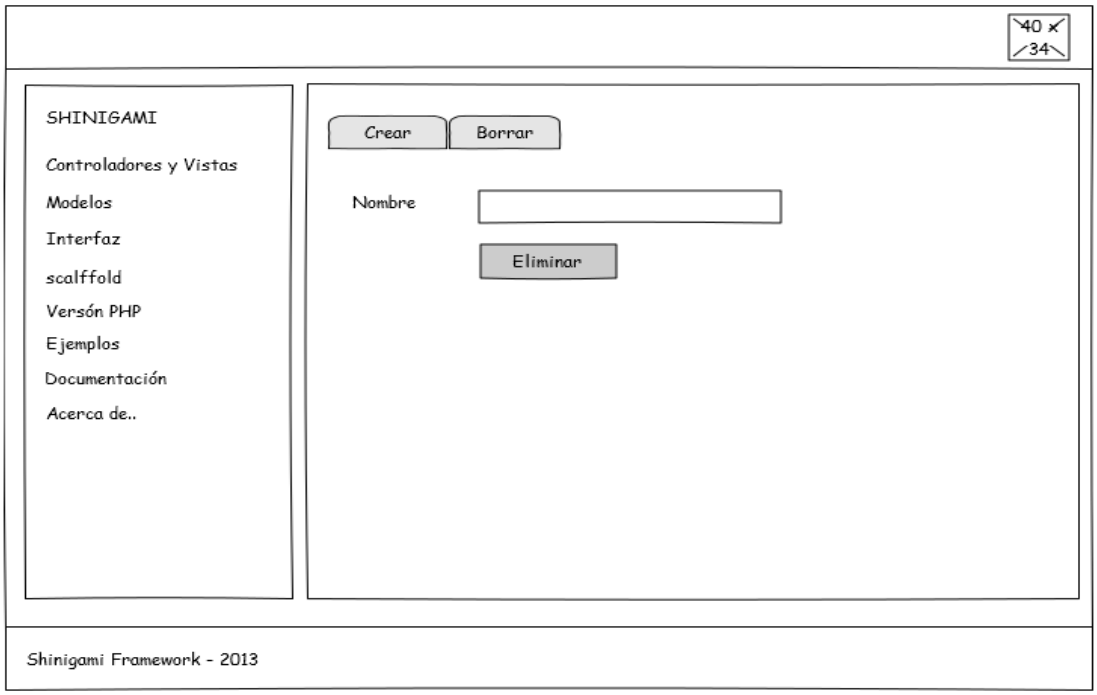


Gráfico N° 7: Diseño para eliminación de vistas y controladores

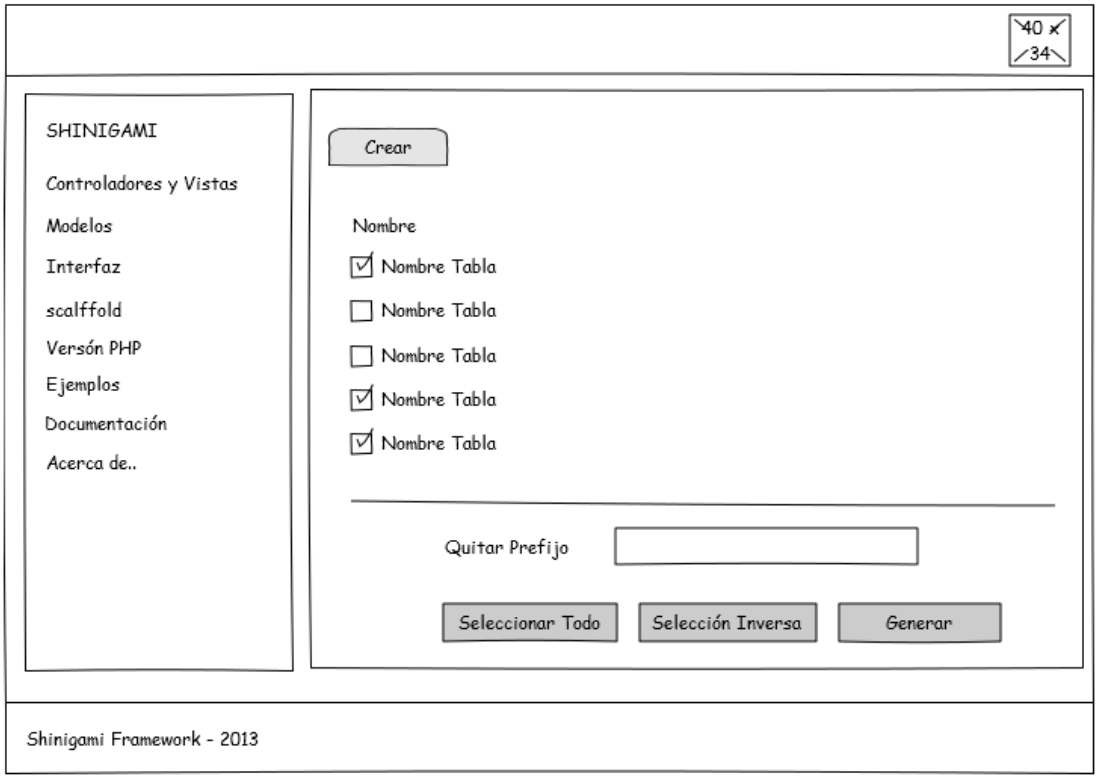


Gráfico N° 8: Diseño para la creación de modelos

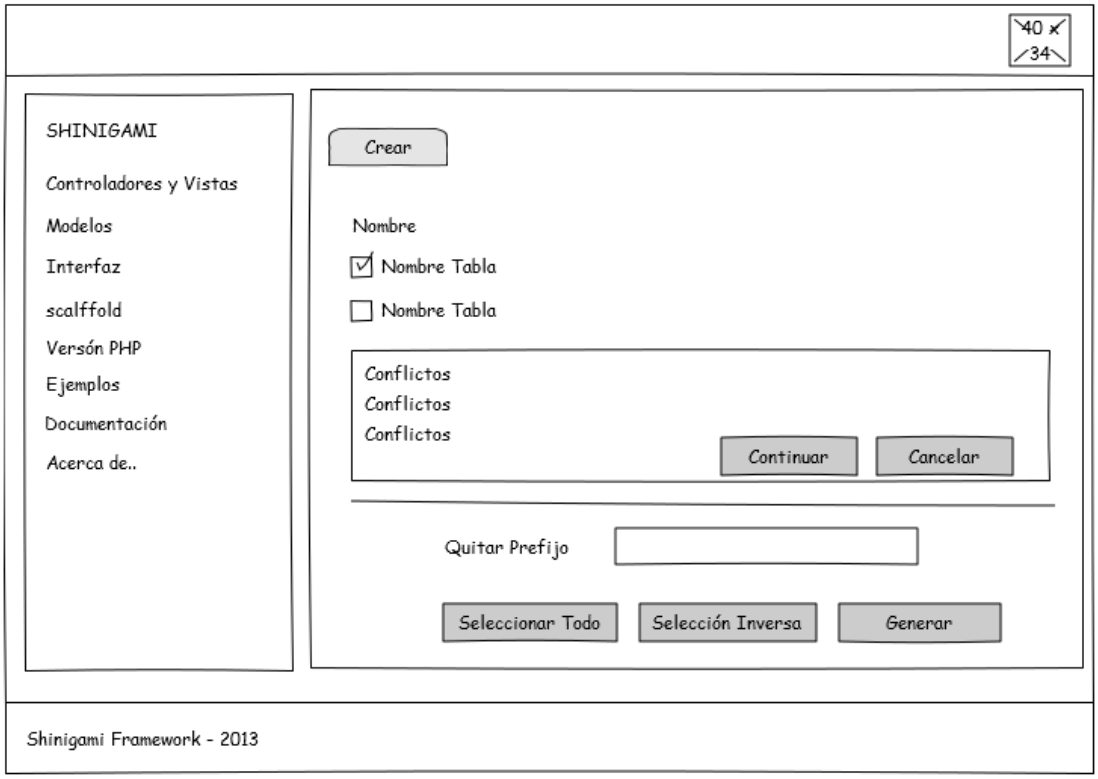


Gráfico N° 9: Diseño para la visualización de conflictos en la creación de modelos.

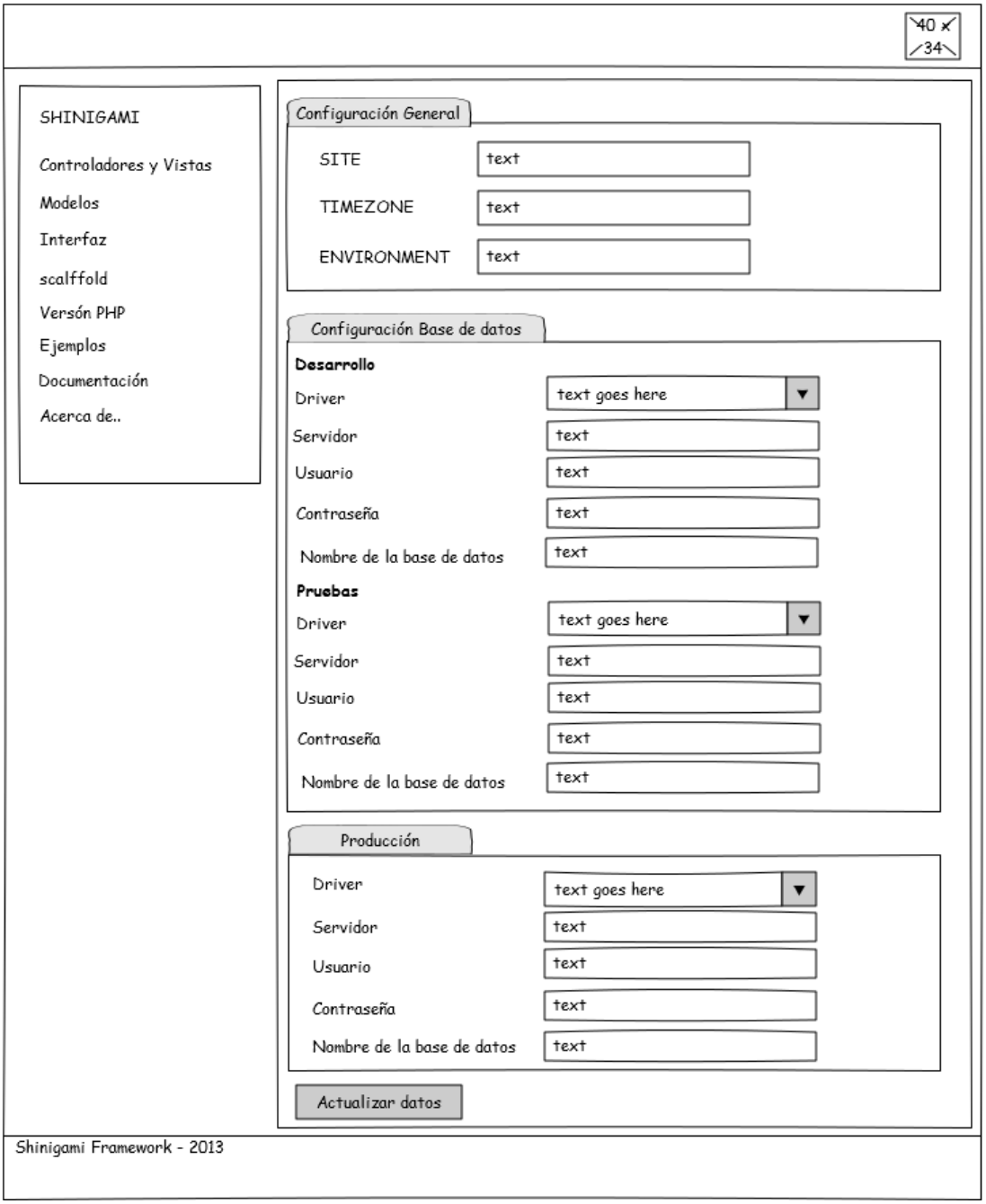


Gráfico N° 10: Diseño configuración del proyecto

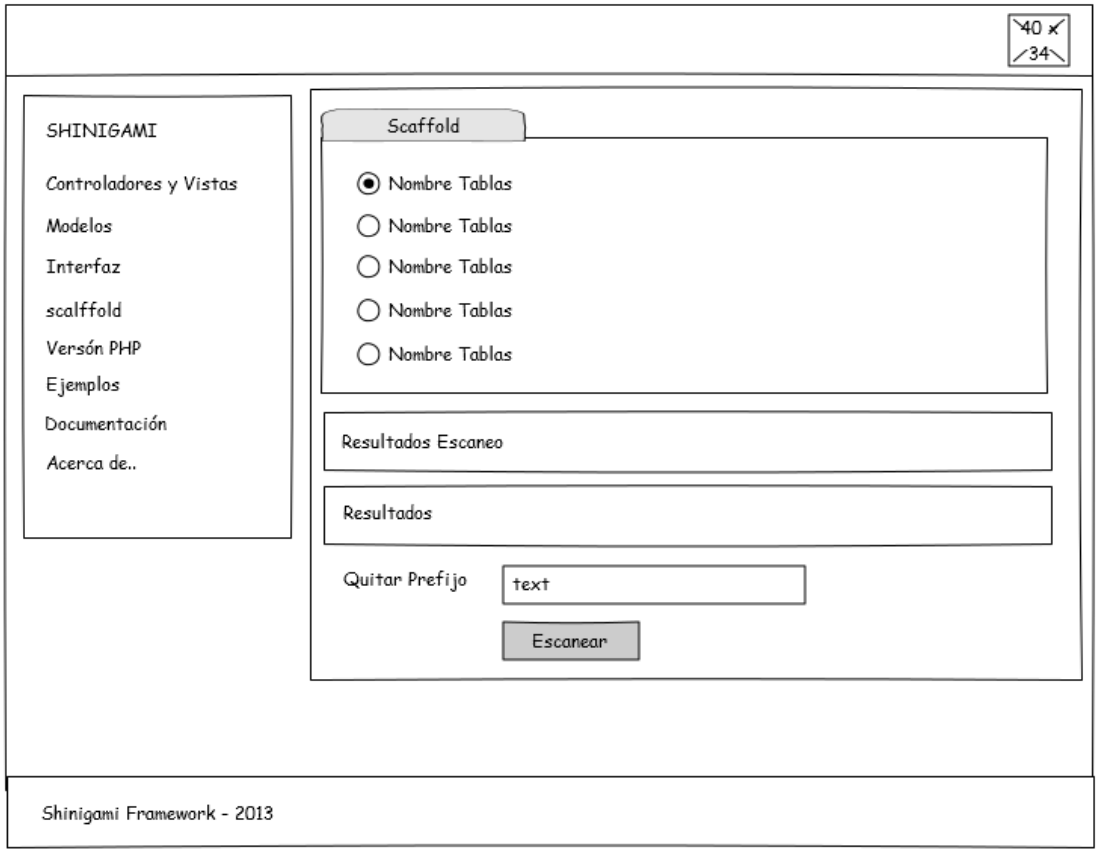
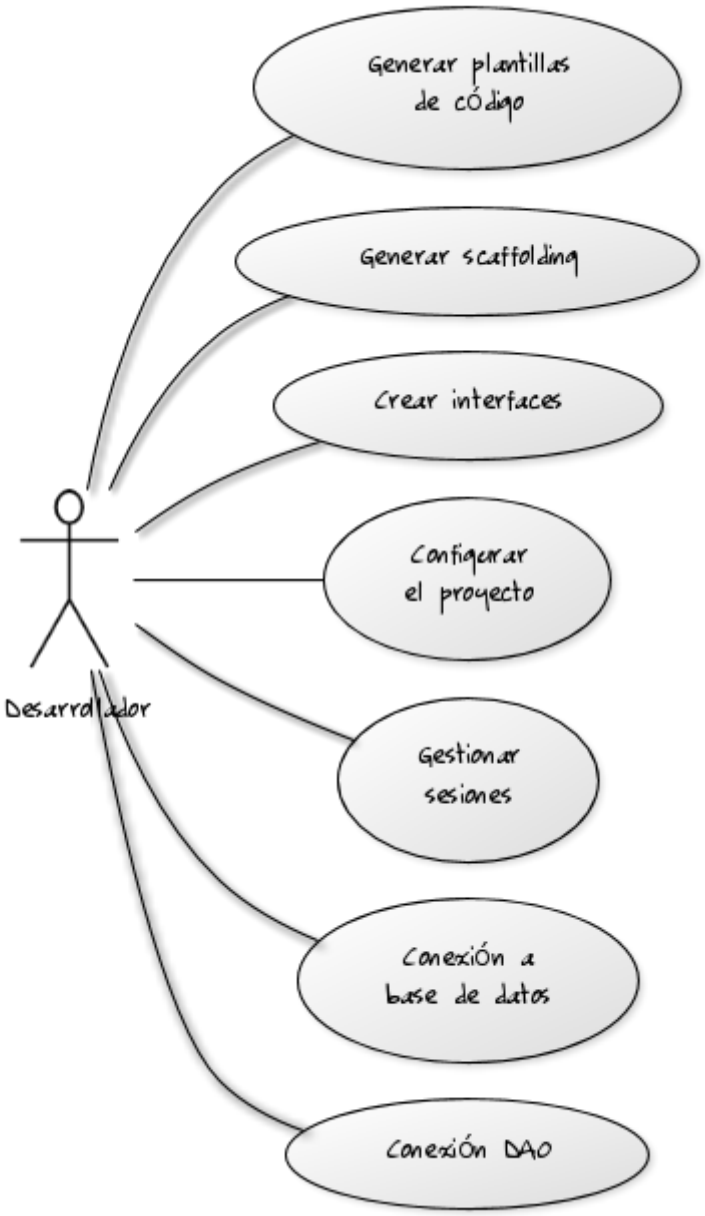


Gráfico N° 11: Diseño Scaffolding

3.3.2 Diagrama de casos de uso

Son diagramas que documentan el comportamiento de un sistema desde el punto de vista del usuario.

A continuación se presenta el diagrama de casos de uso que se desarrolló para el proyecto correspondiente al rol del desarrollador.



Gráfica N° 12: Diagrama de casos de uso, rol desarrollador

En este diagrama podemos observar que el actor es el desarrollador ya que es él quien interactuará directamente con el Framework, a la vez se puede observar las diferentes acciones que podrá realizar como es el caso de generar plantillas de código.

Caso de uso 001: Generar plantillas de Código

Identificación	CU-001	**Fecha de realización**	29/marzo./2013
Caso de Uso	Generar plantillas de Código		
Versión	0.1		
Autor	Leonardo Rico – Jhon Rondón		
Actores	Desarrollador		
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el desarrollador decida generar un plantilla de código, la cual podrá ser una vista, controlador, modelo.		
Pre-condición	El desarrollador debe haber ingresado al módulo “generator” del Framework.		
Post-condición	El desarrollador ha generado su plantilla de código y el Framework lo ha integrado al proyecto.		
Tipo	Primario		

Flujo normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1. El desarrollador ingresa al módulo “generator”. **3.** El desarrollador selecciona la opción que desea generar. **5.** El desarrollador ingresa los datos solicitados por el sistema y oprime en continuar.	**2.** El sistema muestra al desarrollador una lista de opciones para generar. **4.** El sistema solicita al desarrollador los datos necesarios para la generación del módulo. **6.** El sistema verifica que los datos ingresados sean correctos. **7.** El sistema genera el archivo e informa al usuario el estado del proceso.

Flujo alterno	
Línea 6	Si la información no sea válida informa el estado del proceso al desarrollador y termina el proceso.
Línea 7	Si el archivo a generar ya existe, el sistema pedirá confirmación al desarrollador de reemplazar el archivo, en caso de que la decisión del desarrollador sea negativa el proceso termina, en caso contrario continuar con el paso 7.

La generación de los modelos es un poco diferente, dado esto se tratará su flujo de eventos en otro caso de uso (Caso de uso 004).

Caso de uso 002: Generar Scaffolding

Identificación	CU-002	**Fecha de realización**	29/marzo./2013
Caso de Uso	Generar Scafolding		
Versión	0.1		
Autor	Leonardo Rico – Jhon Rondón		
Actores	Desarrollador		
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el desarrollador decida generar el Scaffolding de una entidad en la base de datos.		
Pre-condición	El desarrollador debe haber ingresado al módulo “generator” del Framework.		
Post-condición	El desarrollador ha generado el Scafolding de una entidad.		
Tipo	Primario		

Flujo normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1. El desarrollador ingresa al módulo “generator”.	**2.** El sistema muestra al desarrollador una lista de opciones.
3. El desarrollador selecciona la opción Scaffold.	**4.** El sistema solicita al desarrollador que seleccione la entidad a la cual desea realizar el Scaffolding.
5. El desarrollador selecciona la tabla.	**6.** El sistema busca la entidad en la base de datos.
8. El desarrollador ingresa los datos solicitados por el sistema.	**7.** El sistema solicita al desarrollador los datos necesarios para la generación del Scaffolding.
	9. El sistema validad que lo datos ingresados sean válidos.
	10. El sistema genera el Scaffolding de la entidad seleccionada.

Flujo alterno	
Línea 6	Si el sistema no encuentra la entidad solicitada, informa el estado del proceso al desarrollador y termina el proceso.
Línea 8	Si la información no sea válida informa el estado del proceso al desarrollador y termina el proceso.
Línea 9	Si el módulo a generar ya exista, el sistema reemplazará el módulo por el nuevo.

Caso de uso 003: Generar Interfaces

Identificación	CU-003	**Fecha de realización**	29/marzo./2013
Caso de Uso	Generar Interfaces		
Versión	0.1		
Autor	Leonardo Rico – Jhon Rondón		
Actores	Desarrollador		
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el desarrollador decida generar la interfaz de una vista.		
Pre-condición	El desarrollador debe haber ingresado al módulo “generate” del Framework.		
Post-condición	El desarrollador ha generado la interfaz de una vista dentro del Framework.		
Tipo	Primario		

Flujo normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1. El desarrollador ingresa al módulo “generator”.	**2.** El sistema muestra al desarrollador una lista de opciones para generar.
3. El desarrollador selecciona la opción de generar interfaz y diseño.	**4.** El sistema desplegará una lista de opciones disponibles para crear una interfaz.
5. El desarrollador crea de manera intuitiva como se verá la interfaz final.	**6.** El sistema valida las opciones generadas por el usuario.
	7. El sistema creará las plantillas en archivos físicos listos para ser integrados.

Flujo alterno	
Línea 6	El sistema evalúa las combinaciones de compatibilidad con los navegadores seleccionados indicando posibles conflictos.

Caso de uso 004: Generar DAO

Identificación	CU-004	**Fecha de realización**	29/marzo./2013
Caso de Uso	Generar DAO		
Versión	0.1		
Autor	Leonardo Rico – Jhon Rondón		
Actores	Desarrollador		
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el desarrollador decida realizar la generación de modelos mediante el patrón DAO.		
Pre-condición	El desarrollador debe haber ingresado al módulo “generator” del Framework.		
Post-condición	El desarrollador ha generado los modelos del proyecto a partir de la base de datos y por medio del patrón DAO.		
Tipo	Primario		

Flujo normal de los eventos	
Acción de los actores	**Respuesta del sistema**
1. El desarrollador ingresa al módulo “generator”.	**2.** El sistema muestra al desarrollador una lista de opciones para generar.
3. El desarrollador selecciona la opción modelos.	**4.** El sistema detecta el motor de bases de datos que se está utilizando y muestra al desarrollador las tablas de la base de datos.
5. El desarrollador selecciona las tablas que requiere generar.	**6.** El sistema busca los posibles conflictos entre archivos y los muestra al usuario.
8. El desarrollador confirma la generación de los archivos.	**7.** El sistema solicita al desarrollar la confirmación para generar los archivos.

	9. A partir de la selección del desarrollador el sistema genera los archivos ubicándolos en el directorio destinado para el proceso. **10.** El sistema informa al desarrollador el estado del proceso.
Flujo alterno	
Línea 8	Si el módulo a generar ya existe, el sistema pedirá confirmación al desarrollador de reemplazarlo, en caso de que la decisión del desarrollador sea negativa el proceso termina, en caso contrario continuar con el paso 9.

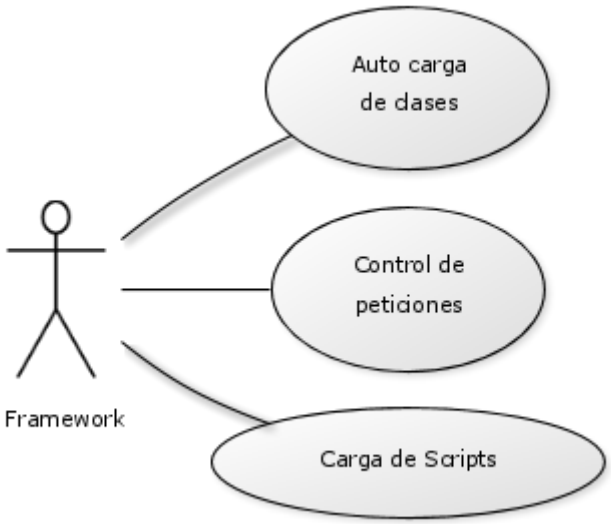
Caso de uso 005: Gestionar sesiones

Identificación	CU-005	**Fecha de realización**	29/marzo./2013
Caso de Uso	Gestionar sesiones		
Versión	0.1		
Autor	Leonardo Rico – Jhon Rondón		
Actores	Sistema		
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el desarrollador decida crear una sesión.		
Pre-condición	No presenta.		
Post-condición	El desarrollador ha creado la sesión.		
Tipo	Primario		
Flujo normal de los eventos			
Acción de los actores		**Respuesta del sistema**	
1. El desarrollador solicita al método encargado pasando por parámetro los datos a guardar en sesión.		**2.** El sistema valida que los datos sean válidos y los guarda los datos en sesión.	

Caso de uso 006: Conexión a base de datos

Identificación	CU-006	**Fecha de realización**	29/marzo./2013
Caso de Uso	Conexión a base de datos		
Versión	0.1		
Autor	Leonardo Rico – Jhon Rondón		
Actores	Sistema		
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando el desarrollador decida realizar una conexión a la base de datos.		
Pre-condición	El desarrollador tuvo que haber configurado los datos de conexión al motor de base de datos que requiere.		
Post-condición	El desarrollador ha creado una conexión con un motor de base de datos.		
Tipo	Primario		
Flujo normal de los eventos			
Acción de los actores		**Respuesta del sistema**	
1. El desarrollador invoca a algún método que se implícitamente necesite de una conexión a la base de datos previamente configurada.		**2.** El sistema al realizar la operación y a través de los datos de configuración establece la conexión.	
Flujo alterno			
Línea 2	Si los datos de conexión no han sido configurados o son erróneos el sistema arroja un error.		

A continuación se puede observar el diagrama de casos de uso destinado a las acciones llevadas a cabo por el sistema de manera implícita causadas por las peticiones del usuario o desarrollador.



Gráfica N° 13: Diagrama de casos de uso, rol Framework

Caso de uso 007: Auto carga de clases

Identificación	CU-007	**Fecha de realización**	29/marzo./2013
Caso de Uso	Auto carga de clases		
Versión	0.1		
Autor	Leonardo Rico – Jhon Rondón		
Actores	Sistema		
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se produzca algún evento o se realiza alguna petición por URL.		
Pre-condición	El usuario final realiza una solicitud al proyecto.		
Post-condición	El sistema ha importado todos los archivos necesarios que fueron invocados en tiempo de ejecución.		
Tipo	Primario		
Flujo normal de los eventos			
Acción de los actores		**Respuesta del sistema**	

1. Se produce un evento el cual necesita de un conjunto de clases que han sido instanciadas.	**2.** El sistema importa las clases invocadas y requeridas por los scripts en tiempo de ejecución.
Flujo alterno	
Línea 2	Si el sistema no puede cargar alguna clase requerida arrojara error.

Caso de uso 009: Control de peticiones

Identificación	CU-008	**Fecha de realización**	29/marzo./2013
Caso de Uso	Control de peticiones		
Versión	0.1		
Autor	Leonardo Rico – Jhon Rondón		
Actores	Sistema		
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se produzca algún evento o se realiza alguna petición por URL.		
Pre-condición	El usuario final realiza una petición al proyecto.		
Post-condición	El sistema ha analizado e invocado el controlador encargado de manejar la solicitud del usuario.		
Tipo	Primario		
Flujo normal de los eventos			
Acción de los actores		**Respuesta del sistema**	
1. Se produce una solicitud por el usuario a través de la URL.		**2.** El sistema analiza e invoca el controlador encargado de manejar la solicitud del usuario.	
Flujo alterno			
Línea 2	Si el sistema no encuentra el controlador encargado de dicha solicitud arroja error.		

Caso de uso 009: Carga de Scripts

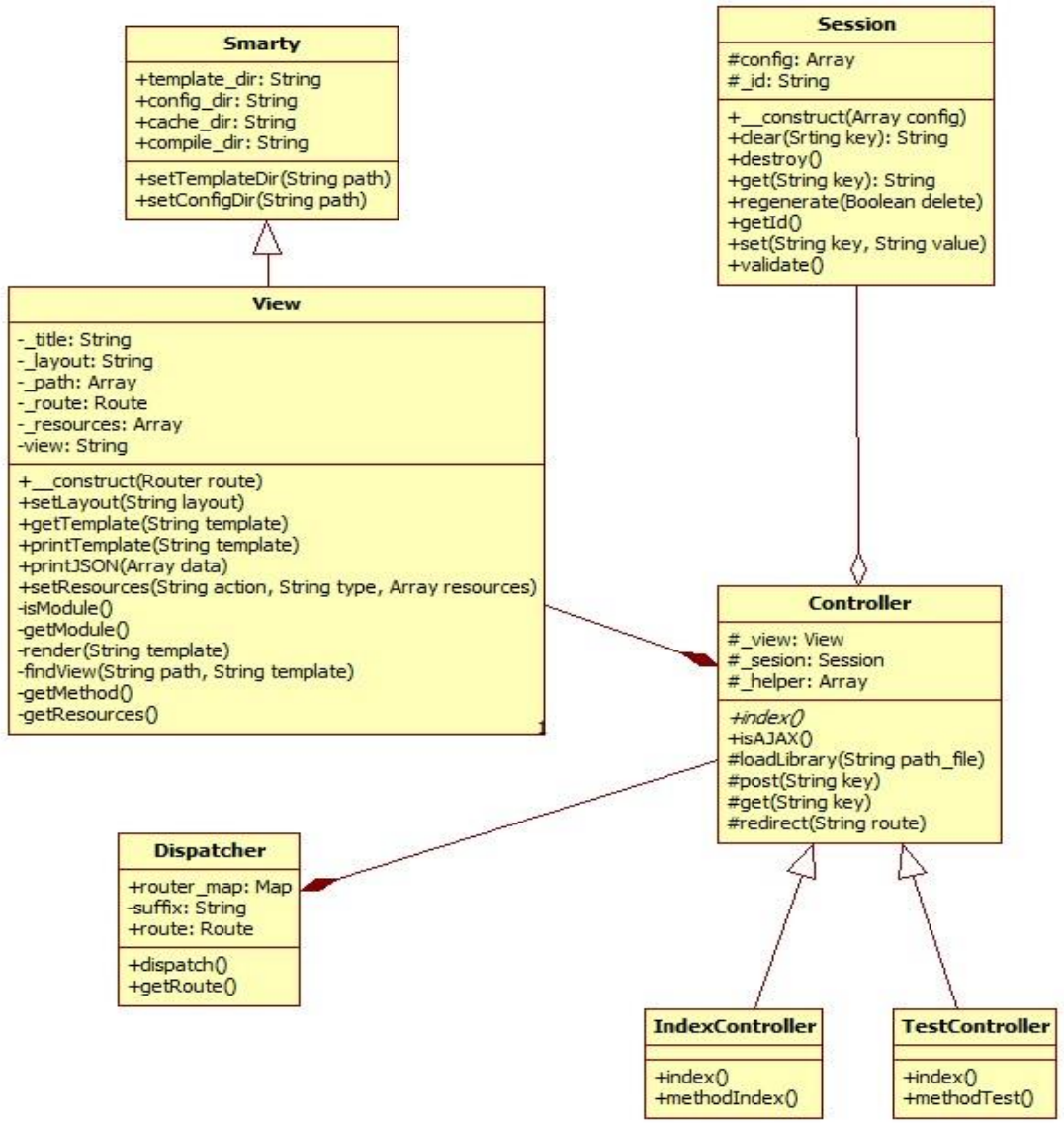
Se debe tener en cuenta que al cargarse una vista, automáticamente los archivos pertenecientes a la vista como son los JavaScript, CSS se cargan automáticamente, sin embargo el desarrollador puede restringir este proceso de ser necesario.

Identificación	CU-009	**Fecha de realización**	29/marzo./2013
Caso de Uso	Carga de Scripts		
Versión	0.1		
Autor	Leonardo Rico – Jhon Rondón		
Actores	Sistema		
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se renderiza una vista al navegador.		
Pre-condición	El usuario final realiza una petición a la plataforma.		
Post-condición	El sistema ha cargado automáticamente los scripts pertenecientes a la vista.		
Tipo	Primario		
Flujo normal de los eventos			
Acción de los actores		**Respuesta del sistema**	
1. Se produce una solicitud por el usuario.		**2.** El sistema analiza e invoca el controlador encargado de manejar la solicitud del usuario. **3.** El sistema invoca la vista encargada de procesar la información solicitada. **4.** El sistema verifica si hay restricciones al cargar automáticamente los scripts pertenecientes a la vista. **5.** El sistema carga dinámicamente los scripts.	
Flujo alterno			

Línea 2	Si el sistema no encuentra el controlador encargado de dicha solicitud arroja error.
Línea 4	Si el sistema encuentra que hay restricciones para cargar todos los archivos, omite el paso y continúa con el flujo normal.

3.3.2 Diagrama de clases

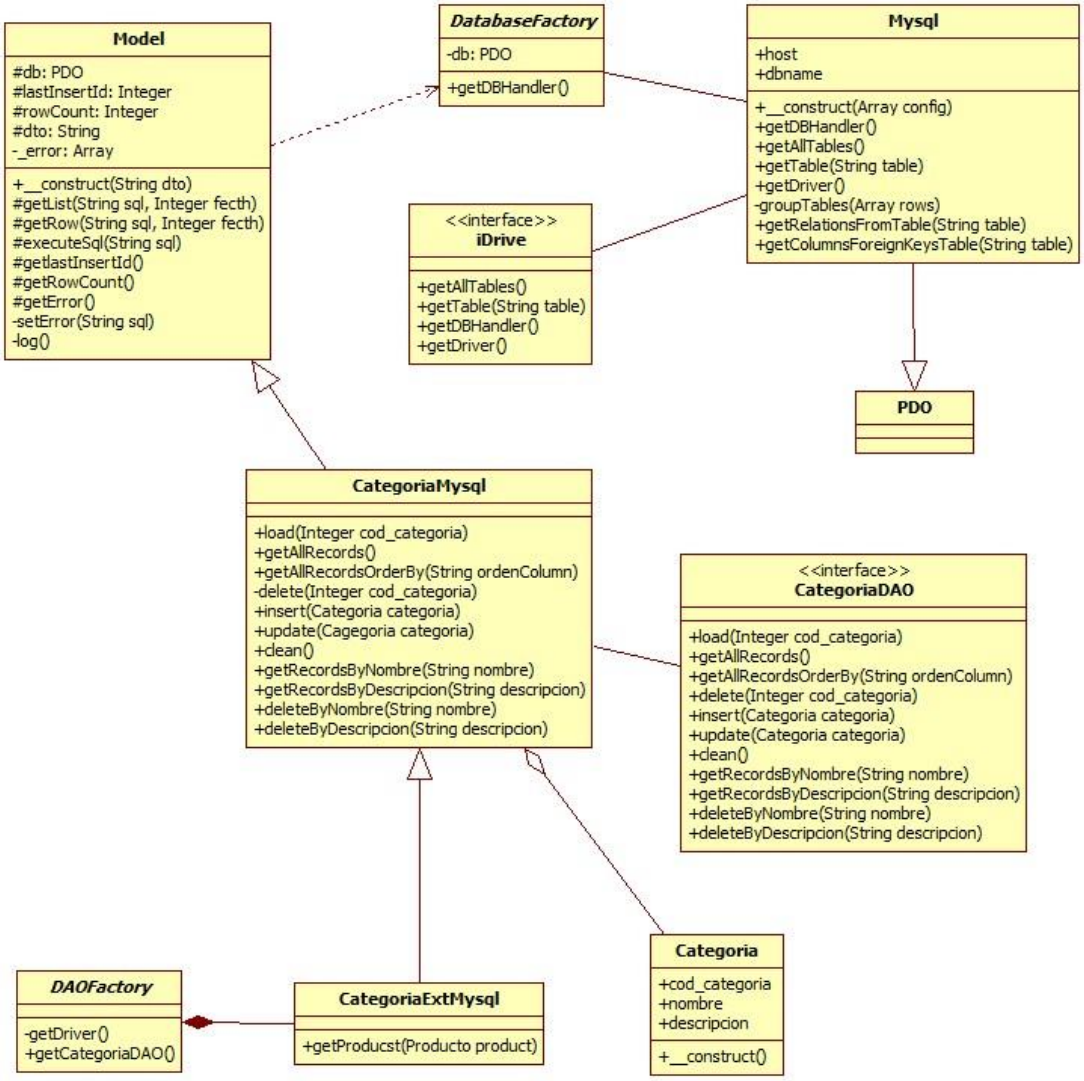
En los siguientes diagramas se encontrará la distribución de las clases para los módulos principales del Framework, como son la representación de los modelos a través del Patrón DAO, la vista y el controlador base del Framework presentando sus atributos, relaciones y el funcionamiento del sistema.



Gráfica N° 14: Diagrama de clases, representación de controladores

En el anterior diagrama de clases se representa la estructura que tendrían los controladores que se implementen en el proyecto, en este caso IndexController y TestController los cuales extienden las propiedades y métodos de su controlador padre.

El siguiente diagrama de clases permite visualizar las relaciones y propiedades cuando se implementa un modelo, en este caso Categoría quien representa a una tabla en el motor de bases de datos MySQL.



Gráfica N° 15: Diagrama de clases, representación de modelos de persistencia

4. DESARROLLO

Se deben tener dos aspectos de suma relevancia para la utilización del Framework, y son las especificaciones técnicas tanto de software como de hardware, para poder aprovechar y sacarle el máximo provecho a la aplicación.

4.1 Especificaciones técnicas

Esta es la etapa dónde se tendrá en cuenta todas las características en la cual interactuara el sistema, éste incluye capacidad de procesamiento, ya que se requieren ciertas exigencias para sacar un buen provecho y lograr de esta manera un rendimiento óptimo de la herramienta.

4.1.2 Software

Para garantizar el correcto funcionamiento del Framework se requiere:

- Sistemas operativos: Windows, Linux, MAC.
- Navegador Web compatible con el estándar HTML5, para MOZILLA FIREFOX desde su versión 3.6+, GOOGLE CHROME versión 3.0+, Opera versión 11+, Safari, MAXTHON.
- Servidor Apache, ya sea: XAMPP, WAMP, LAMP, MAMP o APPSERV.
- Motor de base de datos, MySql, MongoDB.

4.1.3 Hardware

Para un óptimo desempeño, el equipo o servidor deberá contar como mínimo con las siguientes especificaciones:

Requerimientos Mínimos:

- **Local:**
  - Procesador: Pentium 1000 MHz
  - Memoria: 128 MB de RAM

- Espacio libre en disco: 160 MB
- Accesos:

Clientes	Especificaciones
200 Accesos / Clientes	Intel Pentium 100MHz De un mínimo de32MB a 64 MB RAM 60MB de espacio en disco duro para la instalación. Mínimo de 250MB a 2GB de espacio libre en el disco duro para el Caché.
De200a 2000 Accesos/Clientes	Intel Pentium 133MHz 64MB RAMmínimo60MB de espacio en disco duro para la instalación. Mínimo de 2GBa4GB de espacio libre en el disco duro para el Caché.
másde2000 Accesos/Clientes	Intel Pentium 166MHz Mínimo64MB RAMmínimo60MB de espacio en disco duro para la instalación. Mínimo de 2GB a 6GB de espacio libre en el disco duro para el Caché.

Requerimientos Recomendados:

- **Local:**
  - Procesador: 2.4 GHz Pentium

- Memoria: 512 MB RAM
- Espacio libre en disco: 1Gb
- Internet de banda ancha
- Accesos:

másde2000/ Accesos/Clientes	Intel Pentium 800Mhz 256MB RAM 120MB de espacio en disco duro para la instalación. Mínimo de 2GB a 6GB de espacio libre en el disco duro para el Caché.

5. GLOSARIO

Apache: Es un servidor HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etc.), Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 [1] y la noción de sitio virtual.

Back-End: Es la parte que procesa la entrada desde el Front-End.

CSS: Hacen referencia a un lenguaje de hojas de estilos usado para describir la presentación semántica (el aspecto y formato) de un documento escrito en lenguaje de marcas.

Data Access Object (DAO): Es un componente de software que suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos, tales como una Base de datos o un archivo. El término se aplica frecuentemente al Patrón de diseño Object.

Diseño Web Adaptable: Es una técnica de diseño y desarrollo web que mediante el uso de estructuras e imágenes fluidas, así como de media-queries en la hoja de estilo CSS, consigue adaptar el sitio web al entorno del usuario.

Framework: Es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de *software* concretos, que puede servir de base para la organización y desarrollo de *software*. Típicamente, puede incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Front-End: Es la parte del software que interactúa con el o los usuarios.

JavaScript: Es un lenguaje de programación el cual es interpretado por el navegador web y es una extensión del lenguaje HTML.

Modelo Vista Controlador: Es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones.

Módulo: es una parte de un programa de ordenador los módulos suelen estar organizados jerárquicamente en niveles, de forma que hay un módulo superior que realiza las llamadas oportunas a los módulos del nivel inferior.

Patrón de diseño: Son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces.

PHP: Lenguaje de programación usado generalmente en la creación de contenidos para sitios web. Es un lenguaje interpretado especialmente usado para crear contenido dinámico web y aplicaciones para servidores.

Servidor web: Es un programa informático que procesa una aplicación del lado del servidor realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas con el cliente generando o cediendo una respuesta en cualquier lenguaje o Aplicación del lado del cliente.

Singleton: El patrón (instancia única) está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto.

6. CONCLUSIONES

Los objetivos propuestos se lograron, a partir de estos, del desarrollado del proyecto y el objeto de estudio se concluye:

- El uso del módulo “generator” del Framework ahorra tiempo al desarrollador al momento de crear vistas, controladores o modelos e interfaces, ya que éste los genera e integra dinámicamente al proyecto.

- La utilización general del Framework facilita el desarrollo de proyectos orientados al web, puesto que hace uso del patrón de diseño MVC permitiendo tener estructurado el proyecto y proveer al desarrollar herramientas que facilitan la construcción del mismo.

- El manejo del módulo Scaffolding para la creación de modelos garantiza una estructura escalable en los diferentes motores de bases de datos relacionales para el proyecto, ya que utiliza como base el patrón de diseño DAO.

- La implementación del patrón de diseño MVC permite un fácil mantenimiento del proyecto de software, dándole escalabilidad y madurez para el crecimiento.

- La interfaz por la cual está compuesta el Framework es organizada a nivel visual, posicionando los elementos de manera estratégica, con el objetivo de generar un ambiente cómodo que brinde al desarrollador un entorno totalmente amigable y entendible.

7. BIBLIOGRAFÍA

- Sitio oficial del proyecto Apache (1999). Disponible en URL: http://www.apache.org/

- Sitio oficial del servidor local XAMPP (2013), Documentación y descarga del servidor Disponible en URL: http://www.apachefriends.org/es/xampp.html

- Sitio oficial del lenguaje de programación PHP (1995), Documentación y ejemplos del lenguaje. Disponible en URL : http://www.php.net/

- Sitio oficial de la librería Smarty Template Engine, Documentación y descarga. Disponible URL : http://www.smarty.net/

- Sitio oficial de la librería Aura Router, Documentación y descarga Disponible URL : https://github.com/auraphp/Aura.Router

- Sitio oficial UML (2011), Disponible en URL: http://www.uml.org/

- MVC Y PHP, Información general de MVC en PHP, Disponible en URL: http://eugeniabahit.blogspot.com/2011/07/poo-y-mvc-en-php.html

- Información general Diagrama de Casos de uso (2009), Disponible en URL: http://www2.uah.es/jcaceres/uploaded/capsulas/DiagramaCasosDeUso.pdf

.

8. MANUALES

8.1 MANUAL TÉCNICO

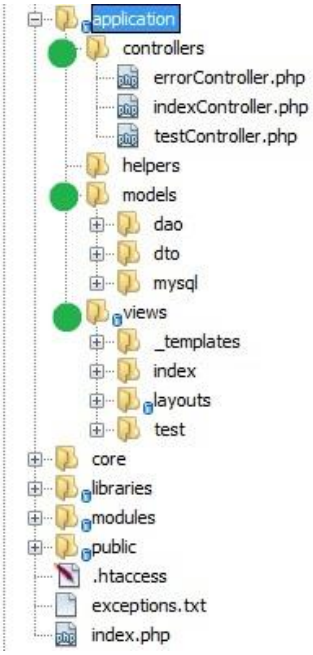
Tabla de contenido

INTRODUCCIÓN

Shinigami Framework es una herramienta que busca mejorar la manera de realizar proyectos en la web, mezclando una variedad de lenguajes de programación, librerías y recursos que facilitan el desarrollo de los mismos permitiendo crear múltiples plataformas con diferentes enfoques de manera rápida y simple y eficiente. Además éste Framework incorpora soporte completo tanto para el Back-End como para el Front-End, ya que existen muchos frameworks que prometen ejercer estas mismas tareas, pero la gran mayoría se concentran sólo en el Back-End dejando a un lado el Front-End. El presente manual de usuario pretende especificar características de comportamiento, lenguajes de programación utilizados, módulos y componentes integrados, estrategias y formas de ejecución y creación de proyectos dentro del Framework.

1.1 Definiciones previas

Shinigami Framework, tiene el core desarrollado con el lenguaje de programación PHP, y maneja patrones de diseño como MVC (Modelo, Vista, Controlador), el cual separa los datos y la lógica de negocio de la aplicación con la interfaz de usuario, y el módulo encargado de gestionar los eventos y las comunicaciones.



Gráfica N° 1: Estructura del Framework

2. ESTRUCTURA DEL FRAMEWORK

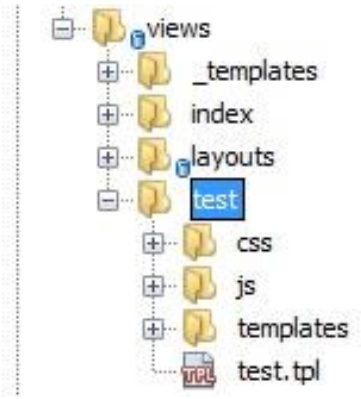
2.1 Application

Como se aprecia en el esquema anterior (Gráfico Nº 1), se puede ver la estructura básica del Framework, se sigue este estándar con el fin de adaptar y asimilar una organización pre establecida dentro del Framework.

En la carpeta con nombre “application” se aprecia la estructura del patrón de diseño MVC la cual está conformada dentro de la herramienta por las siguientes carpetas:

- **controllers:** Aquí se alojarán únicamente los controladores de la aplicación, quedando nombre{Controller}.php.
- **models:** Dependiendo el Motor de la base de datos previamente configurada por el usuario, se guardarán los modelos en la carpeta con el nombre del motor, quedando nombre{MOTOR}.php.
- **views:** Contiene las vistas de cada módulo y secciones que se usan en la aplicación, el template que contiene el código HTML tiene una

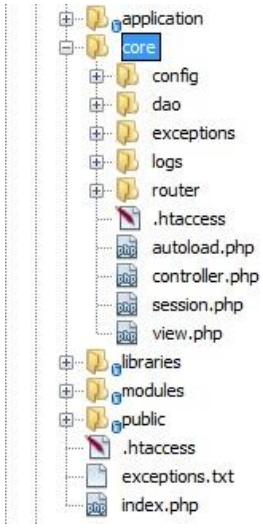
extensión “.tpl” y por defecto se cargará desde dónde sea llamado, para tener una idea más detalla podemos ver el siguiente gráfico:



Gráfica N° 2: Estructura de la carpeta views

Cada “Vista” alojada dentro de ésta carpeta tiene alojado en su interior otras carpetas las cuales se dividen en “css” que se encarga de almacenar las hojas de estilo específicas de la vista, “js” en ésta carpeta se alojan los scripts propios de la vista y por último está la carpeta “templates” en donde se guardaran plantillas en formato .tpl de sub secciones propios de la vista.

2.2. Core



Gráfica N° 3: Estructura del “core” del Framework

El “core” es el denominado núcleo del Framework, allí se encuentran las carpetas específicas únicamente del Framework, como se puede ver en el gráfico anterior (Gráfico Nº 3), la estructura está dada de la siguiente manera:

- **config:** Contiene los archivos básicos de configuración como la selección del motor de la Base de datos, contantes para la aplicación, y configuración de los routers.
- **dao:** Consta de los archivos de configuración para gestionar las conexiones y creación de las Bases de Datos.
- **exeptions:** Administración de excepciones.
- **logs:** Carpeta que almacena sucesos del Framework.
- **router:** Gestiona las peticiones de la url identificando el controlador indicado para cumplir con el proceso.
- **autoload.php:** Carga dinámicamente las clases utilizadas en la ejecución del script.
- **controller.php:** Contiene métodos e instancias de librerías y funciones específicas parar ser usadas en los controladores.
- **session.php:** Aloja métodos para administrar las sesiones, cookies.
- **view.php:** Maneja todos los métodos de las vistas y es la encargada de administrar la librería Smarty para renderizar los templates de la aplicación.

2.3. Libraries

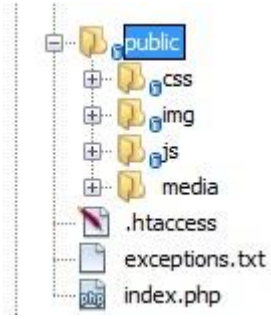
Esta carpeta es exclusiva para almacenar las librerías integradas dentro del Framework para ser utilizadas en el mismo. Allí se encuentra Smarty, y posteriormente se pueden agregar las que se requieran al momento de desarrollar un proyecto.

2.4. Modules

Gráfica N° 4: Modules del Framework

Allí se encuentra el “generador” encargado de crear dinámicamente contenidos requeridos para el proyecto a realizar, ya sean Controladores, Vistas, Modelos, interfaces, configuraciones, etc.

2.5 Public



Gráfica N° 5: Carpeta “public” del Framework

Es la carpeta Multimedia de todo el Framework, y puede ser accedida en busca de un recurso desde cualquier parte de la Herramienta, está conformada de los siguientes directorios:

- **css:** Contiene las hojas de estilo en cascada o (Cascading Style Sheets, o sus siglas CSS.
- **img:** Aloja todas las imágenes del aplicativo, el objetivo es que puedan ser utilizadas en cualquier momento y parte de la herramienta.
- **js:** almacena los archivos JavaScript para ser utilizados y compartidos en las diferentes secciones.
- **media:** Diseñado para guardar recursos adicionales, ya sean videos, presentaciones, etc.

3. MODO DE EJECUCIÓN

Para ejecutar de manera correcta el Framework y acceder a todos sus beneficios es necesario instalar los siguientes recursos:

- ✓ **Servidor Apache**

  Existe gran variedad de compilaciones de Apache que facilitan la instalación, entre los más utilizados podemos resaltar:

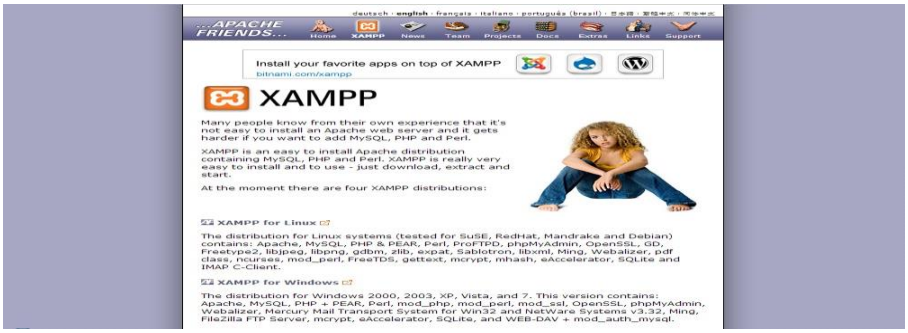
  - **XAMPP:** Es un servidor independiente de plataforma, software libre, que consiste principalmente en la base de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script: PHP y Perl. El nombre proviene del acrónimo de **X** (para cualquiera de los diferentes sistemas operativos), **A**pache, **M**ySQL, **P**HP, **P**erl.

    El programa está liberado bajo la licencia GNU y actúa como un servidor web libre, fácil de usar y capaz de interpretar páginas dinámicas. Actualmente XAMPP está disponible para Microsoft Windows, GNU/Linux, Solaris y MacOS X.

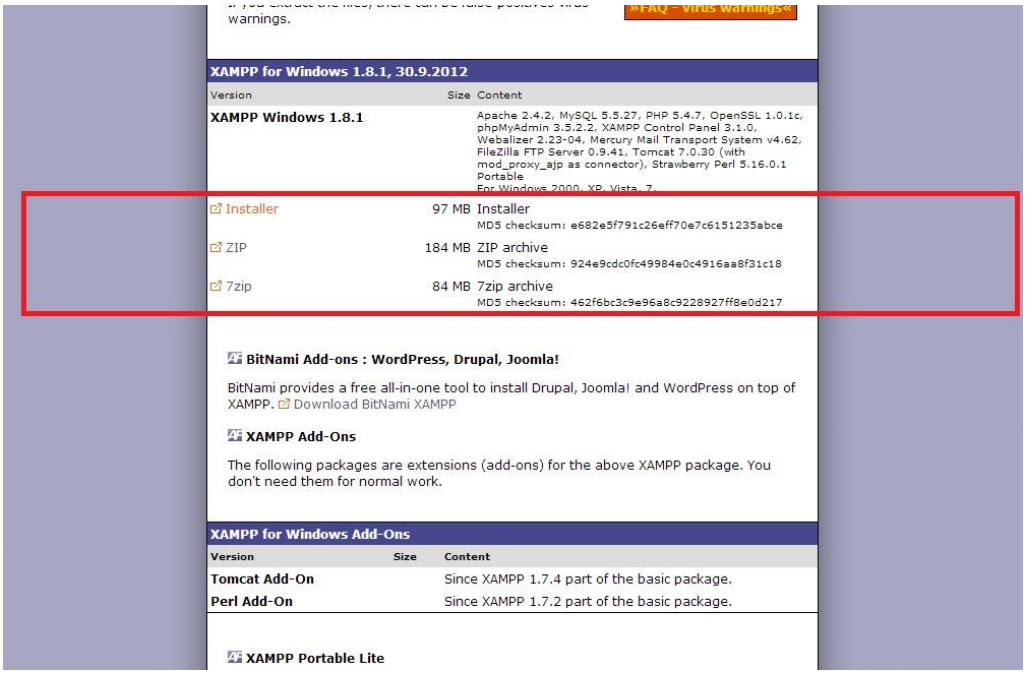
  - **WAMP:** WampServer es un entorno de desarrollo web de Windows. Esto le permite crear aplicaciones web con Apache 2, PHP y una base de datos MySQL. Además, PhpMyAdmin permite manejar fácilmente sus bases de datos.

Para el ejemplo en éste Manual se utilizará el servidor Apache XAMPP, los pasos a seguir son:

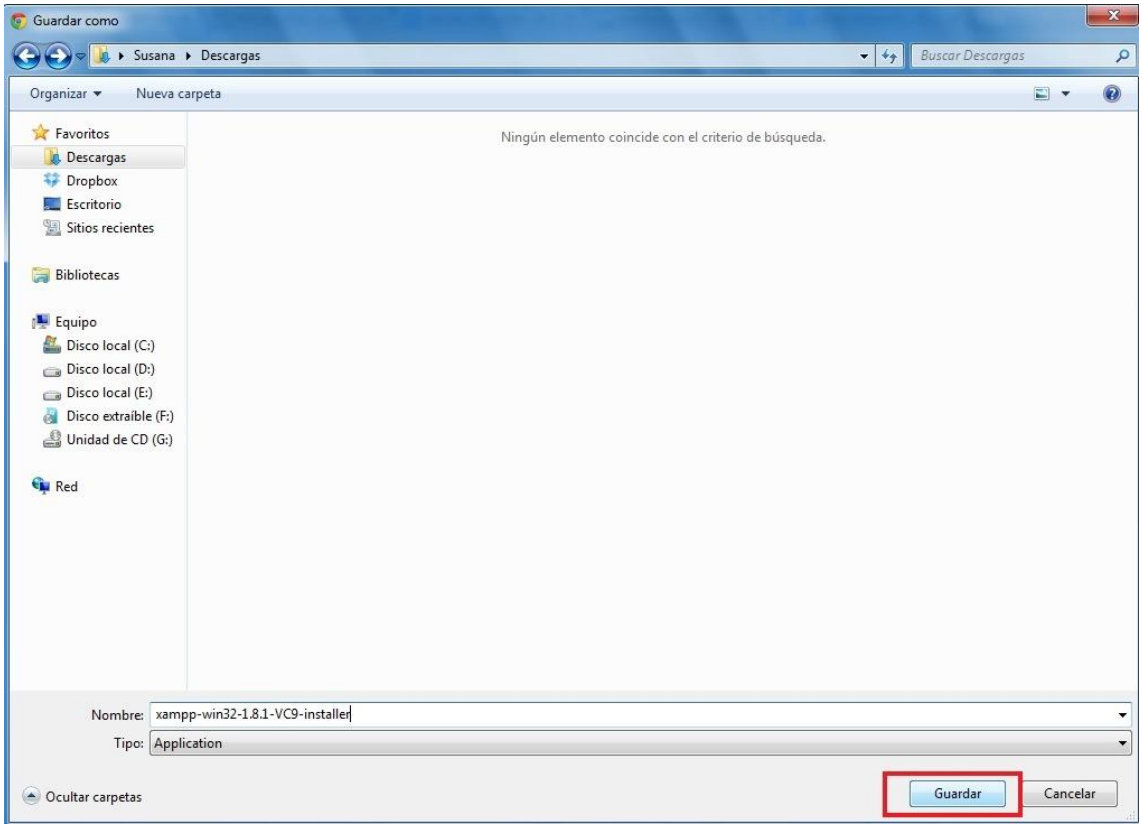
1. Ingresar a la página web de XAMPP para descargarlo y seleccione el sistema operativo correspondiente:
   http://www.apachefriends.org/en/xampp.html



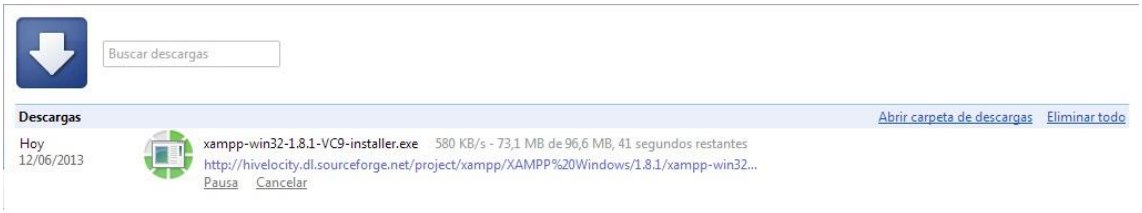
2. A continuación seleccione el tipo de instalador que desea descargar:



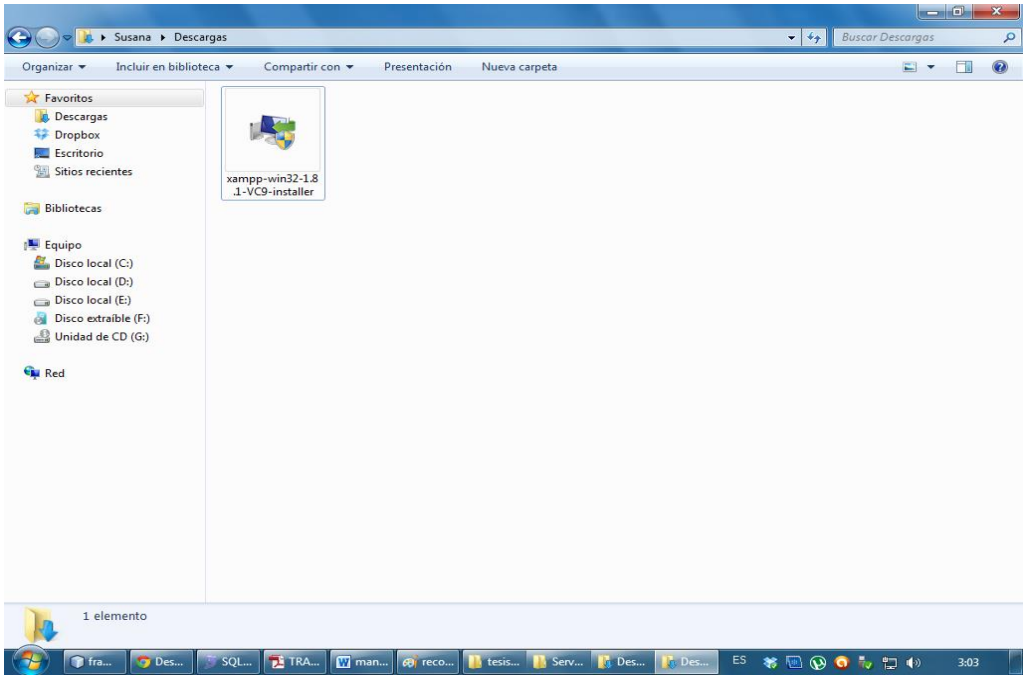
3. Seleccione la carpeta donde guardará el instalador:



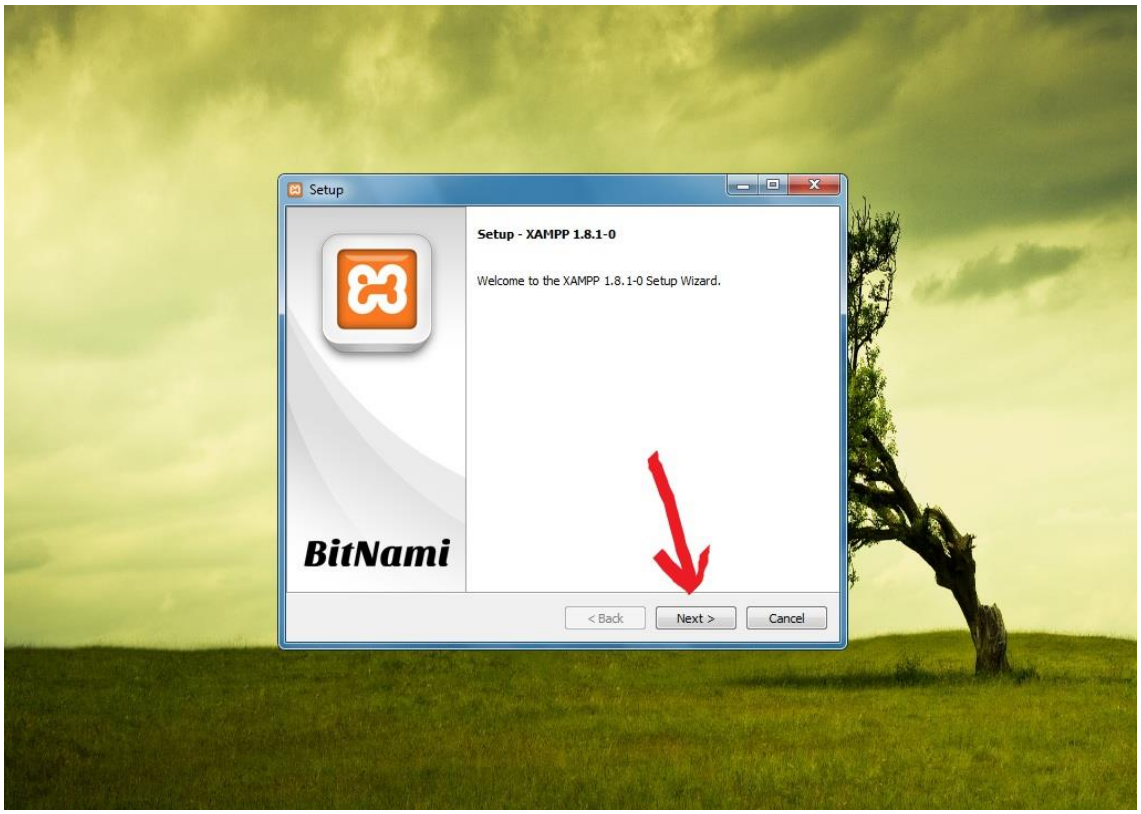
4. El Archivo comenzará a descargar, esperar hasta que se complete el 100% de la descarga:



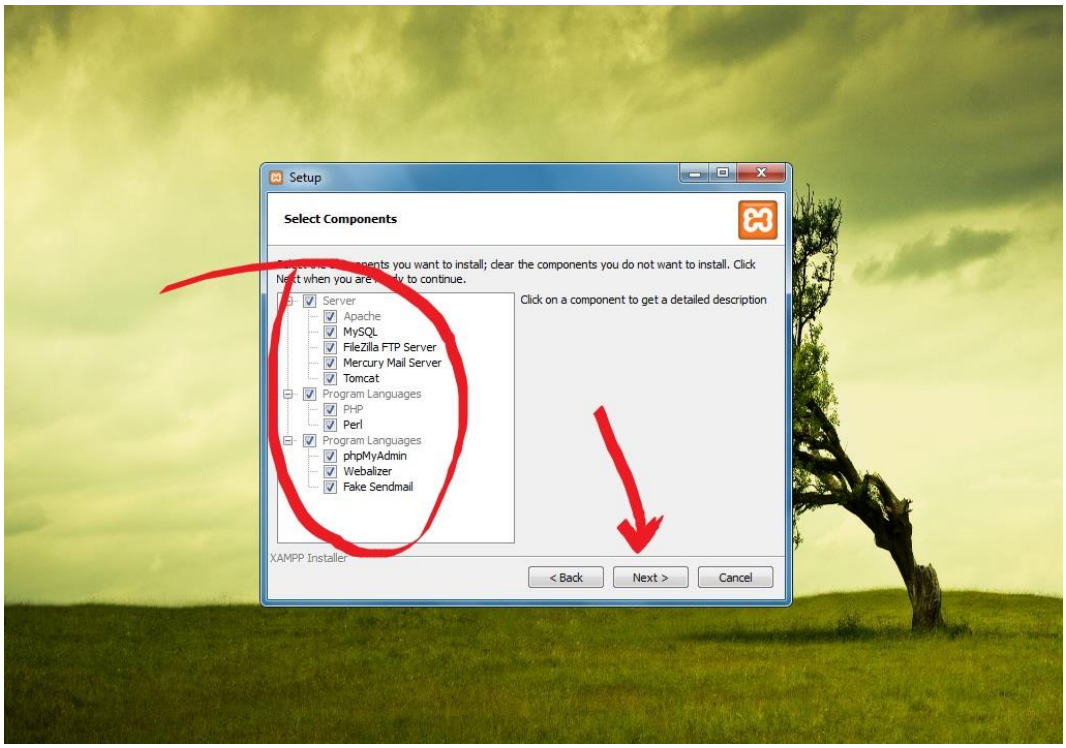
5. Una Vez completada la descarga, visualizaremos el archivos de la siguiente manera y lo ejecutamos:



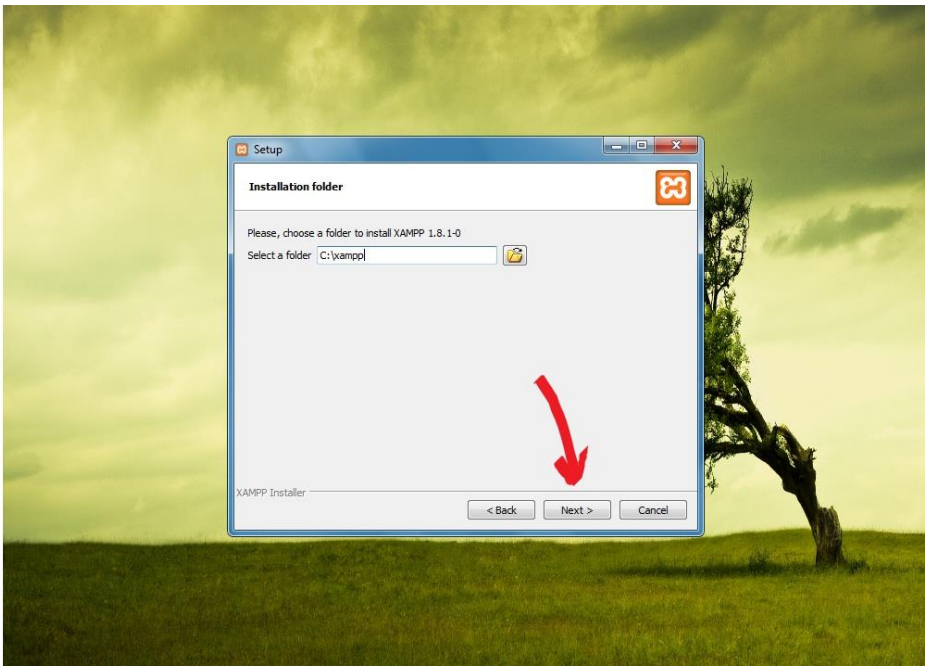
6. Al iniciar el instalador hacer Click en “Next”:



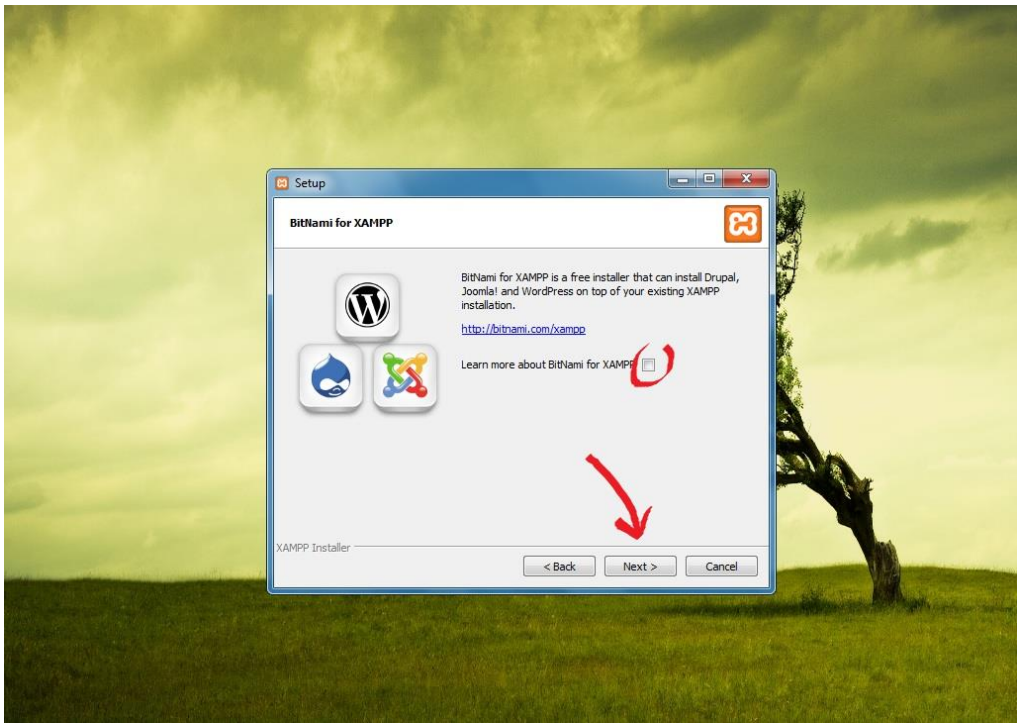
7. Seleccionar los servicios a instalar, por defecto se instalan los que ya predetermino la aplicación y click en “Next”:



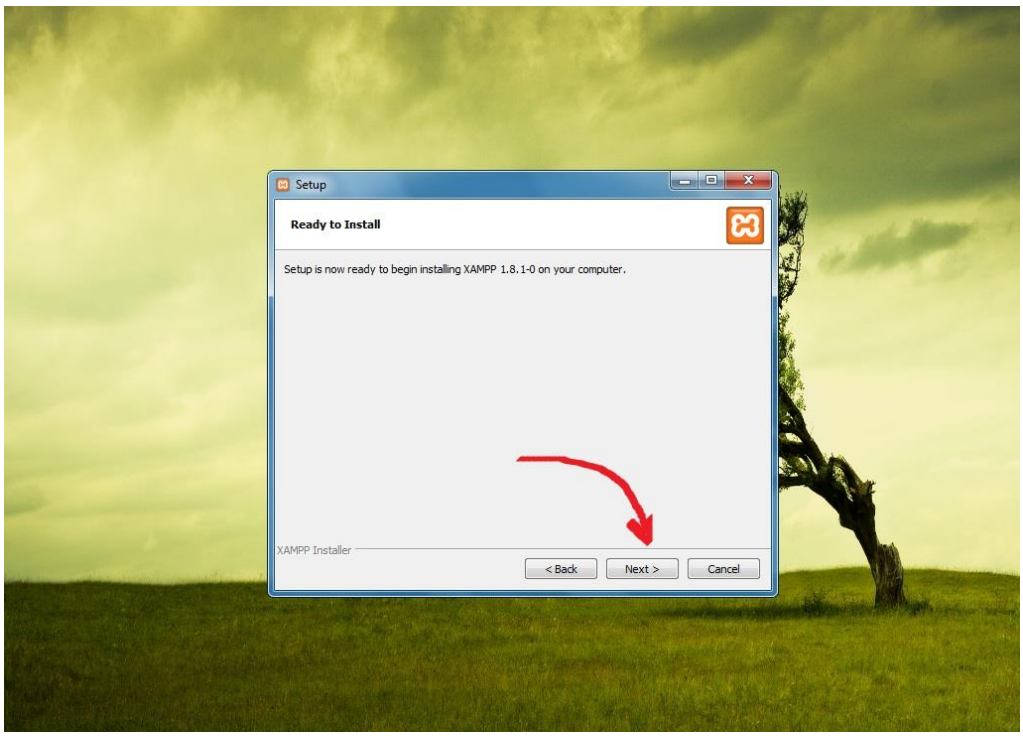
8. Seleccionar Carpeta dónde se instalara XAMPP, en éste ejemplo se dejó la que viene por Defautl es decir **c:\xampp.**



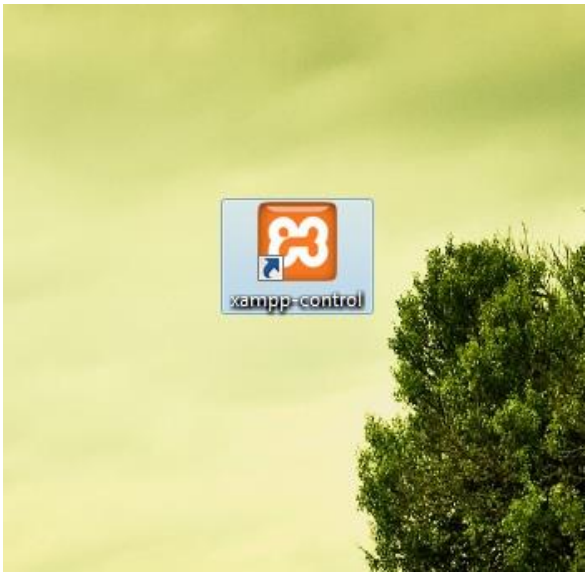
9. Desactivar la casilla como se ve en la ilustración y click en “Next”:



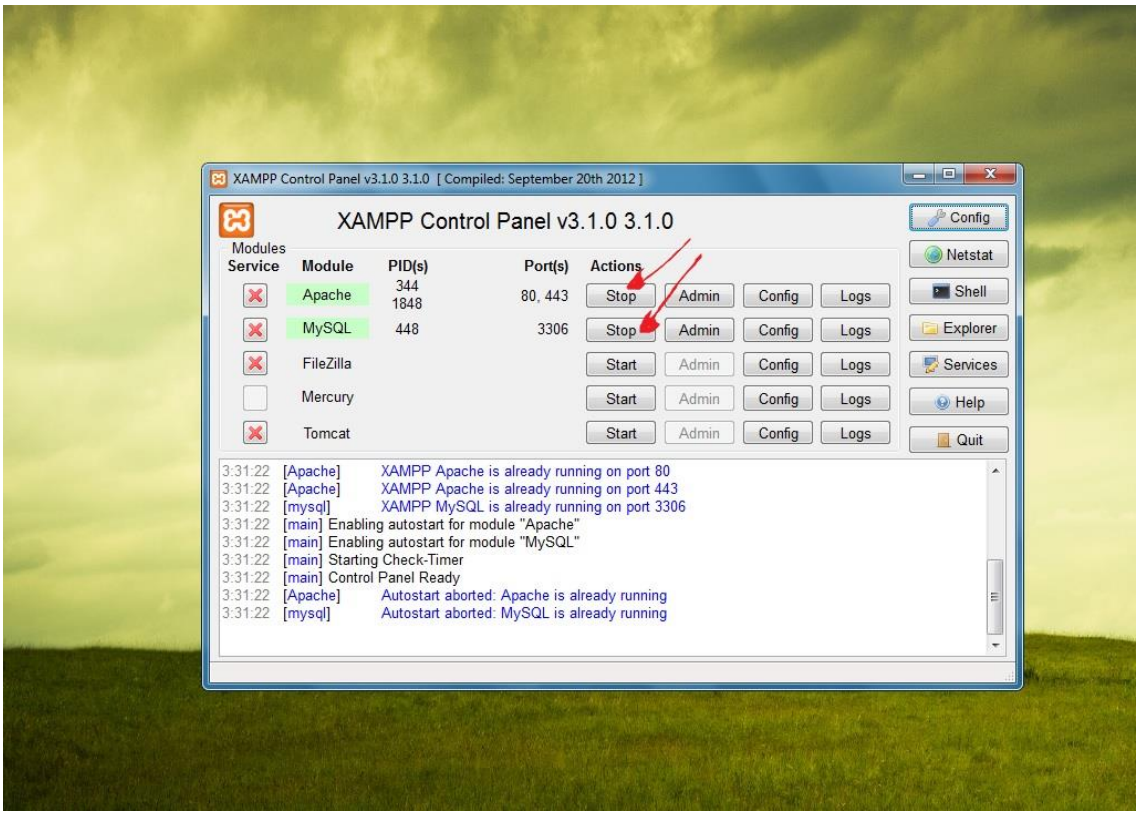
10. Por último Click en “Next” y esperar que complete la instalación.



11. Al finalizar creará un ícono como se muestra en la gráfica, y se ejecuta:



12. Al ejecutar el ícono XAMPP mostrará un Panel y se inician los servicios “Apache” y “MySQL” como se aprecia en la imagen.



13. En éste punto el servidor Apache ya se encuentra listo para ejecutar Shinigami Framework, solo basta con copiar el Framework en la ruta donde queda instalado XAMPP, para este ejemplo seria en C:\xampp\htdocs\. Posterior a este paso ahí que dirigirse a la siguiente ruta: a través del navegado http://localhost/shinigami/generator/welcome/ donde se podrá observar el menú principal del Framework.

8.2 MANUAL DE USUARIO

Tabla de contenido

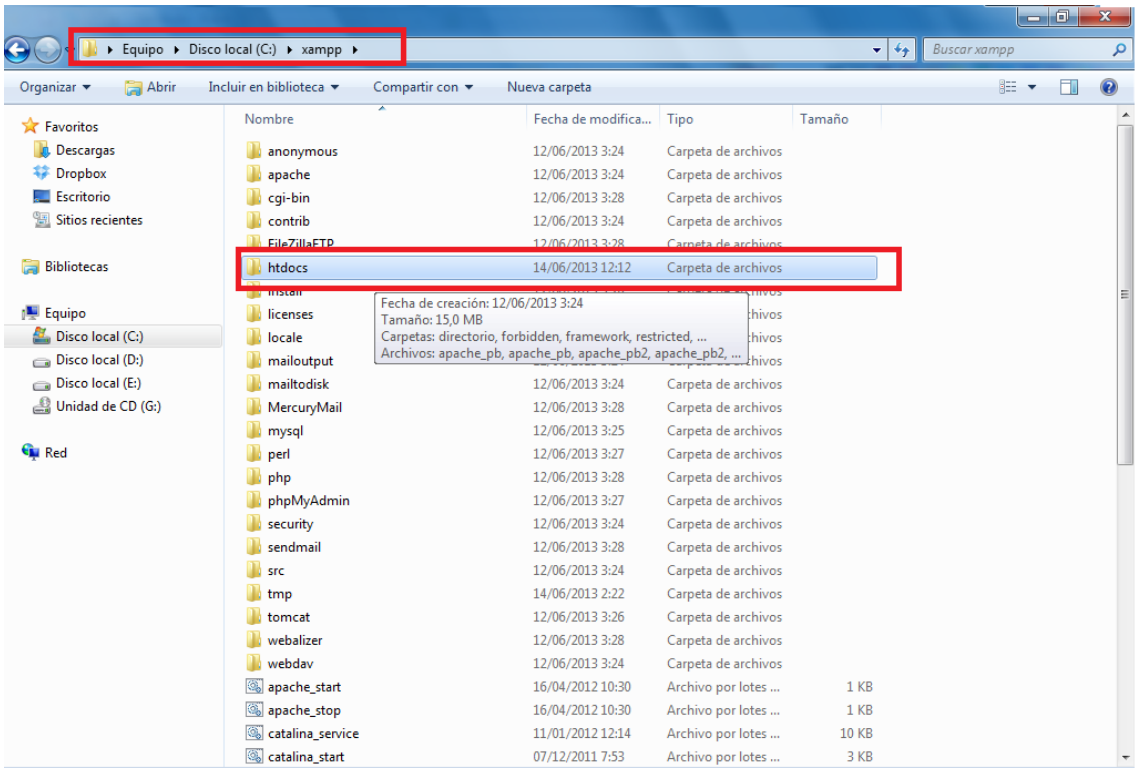
INDRODUCCIÓN

Este manual tiene como objetivo ser una guía en el uso del Framework, en su contenido se podrá observar los aspectos más esenciales para poder utilizar de una manera muy fácil dicha aplicación.

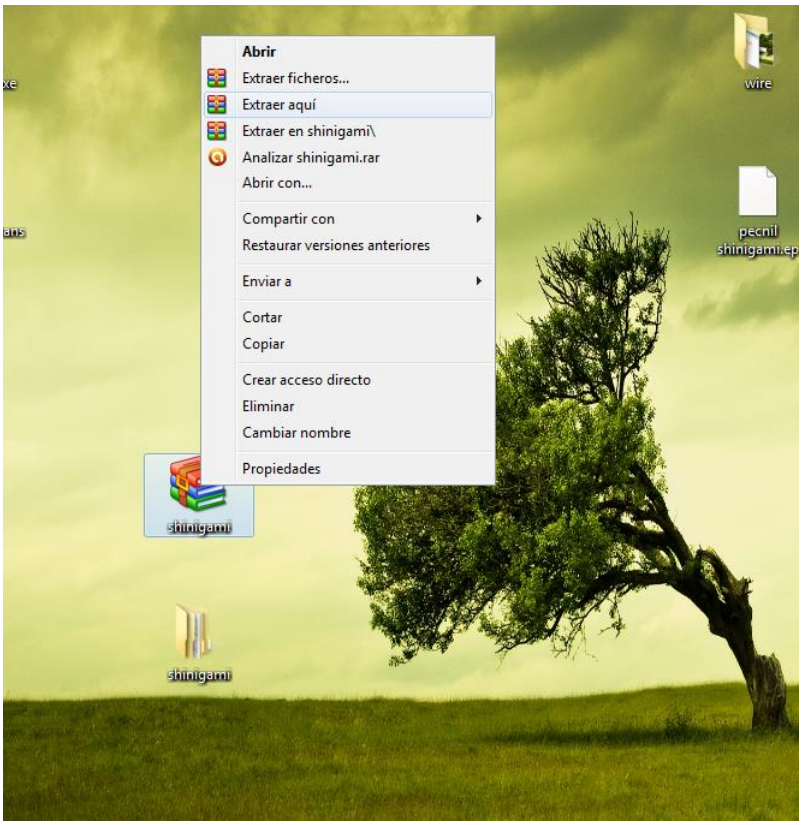
2. Instalación del Framework

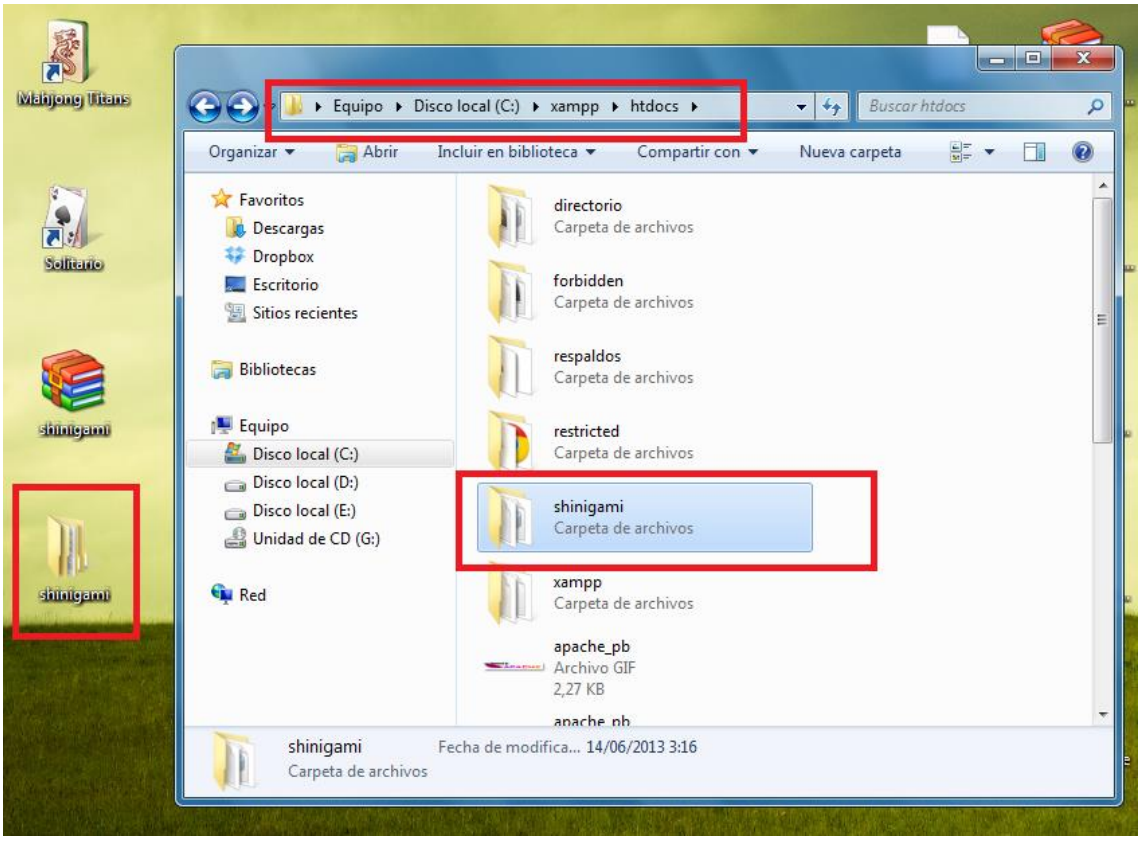
Se ha optado por utilizar un servidor local (XAMPP), que sirva como guía en este ejemplo para la instalación del Framework.

1. Ingresar al directorio en el cual se alojan los proyectos del servidor localhost, al utilizar XAMPP y para el presente ejemplo la ruta es la siguiente: C:\xampp\htdocs

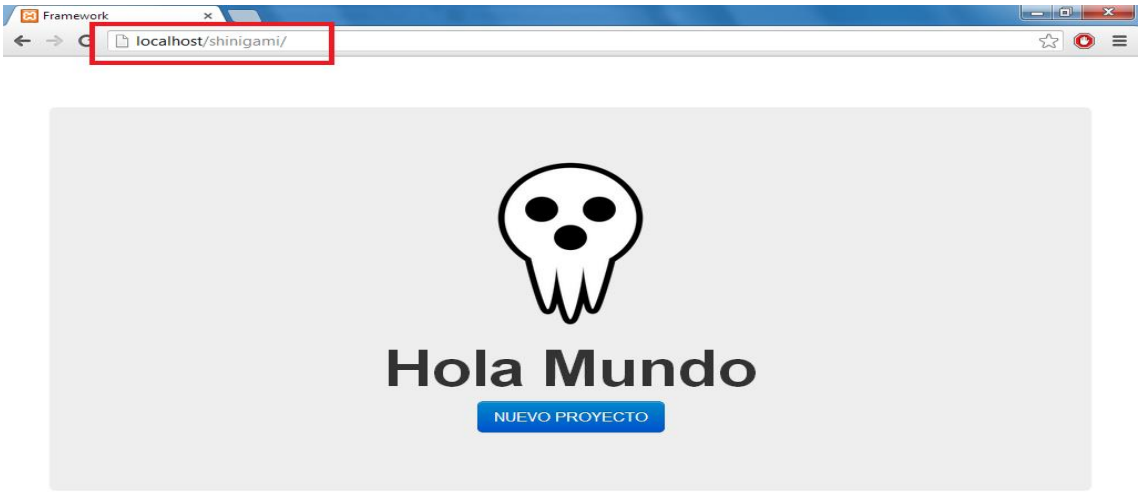


2. Copiar los archivos del Framework al directorio anteriormente nombrado.





3. Una vez realizado los pasos anteriores, el Framework ya quedaría instalado en el servidor local, para confirmar este proceso ingresar a http://localhost/shinigami/

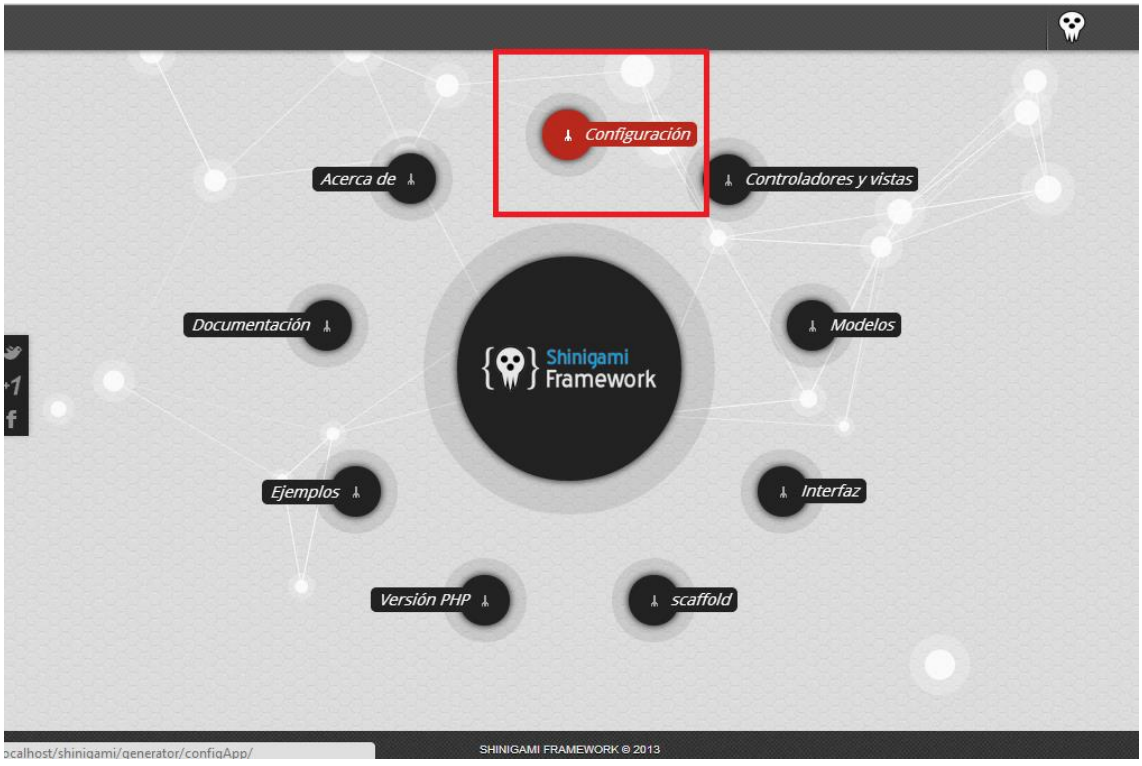


3. Especificaciones del módulo generator

En el Framework el módulo generator es el encargado de facilitar la construcción de código, ya que permite crear diversos templates como son vistas, controladores, interfaces, además que también provee al desarrollar herramientas para la generación de la estructura necesaria para implementar el patrón de diseño DAO en los procesos de conexión a bases de datos.

3.1 Configuración

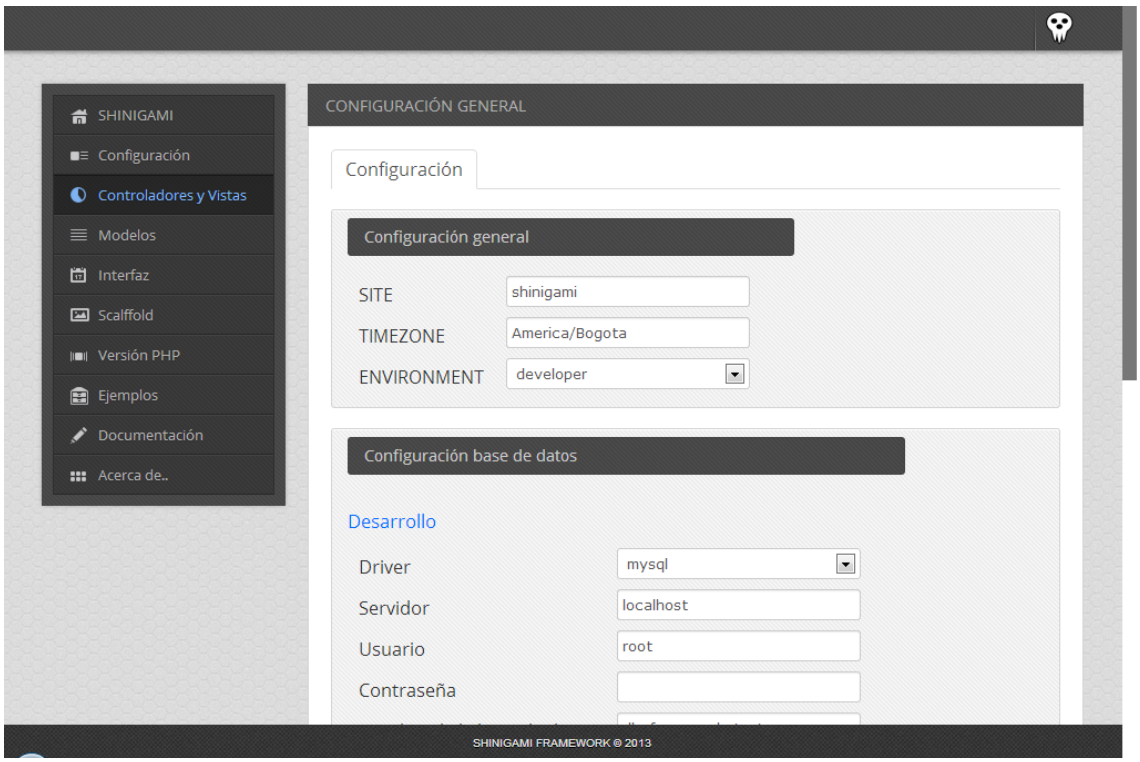
La sección de configuración está destinada a establecer los aspectos más generales del proyecto, como son el nombre, la zona horaria y el ambiente del proyecto (Producción, Desarrollo, Pruebas). A la vez se podrá configurar los datos de conexión con la base de datos. Para acceder a esta sección desde el menú principal de generator ingresar a “Configuración”



Gráfica N° 1: Menú inicio, sección Configuración

Para configurar los valores tan solo basta con actualizar los valores en sus respectivos campos de texto o lista desplegable y oprimir sobre el botón

“Actualizar“, se debe tener en cuenta que este proceso es irrevertible por lo tanto se pide la confirmación de la acción.

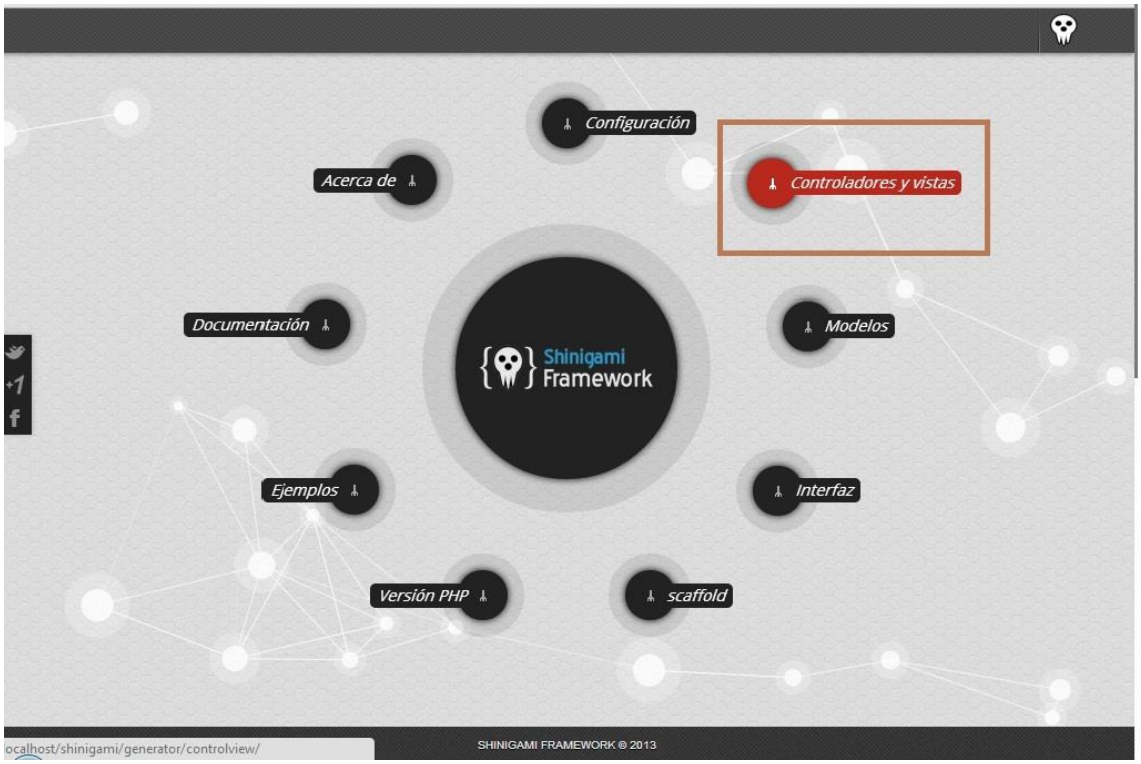


Gráfica N° 2: Configuración del proyecto

3.2 Controladores y Vistas

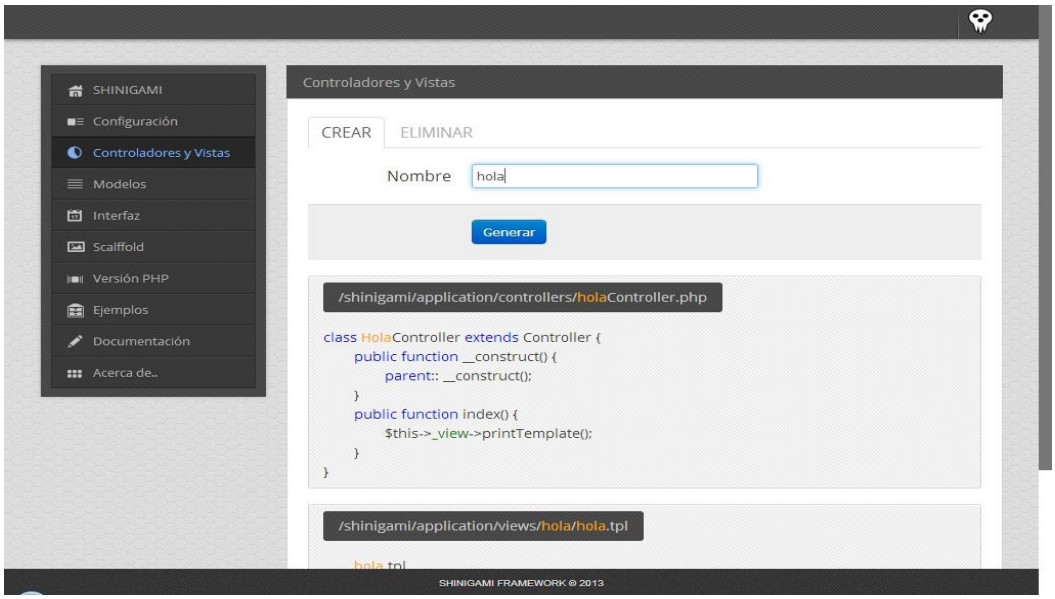
La estructura del Framework está basada en el patrón de diseño MVC, por lo tanto la creación de controladores y vistas es indispensable para correcto funcionamiento del Framework, esta tarea se puede realizar de forma manual o ingresando a la sección “Controladores y Vistas” del módulo generator, en esta sección se podrá tanto crear como eliminar vistas del proyecto.

Para acceder a esta sección desde el menú principal del Framework seleccione “Controladores y vistas”.



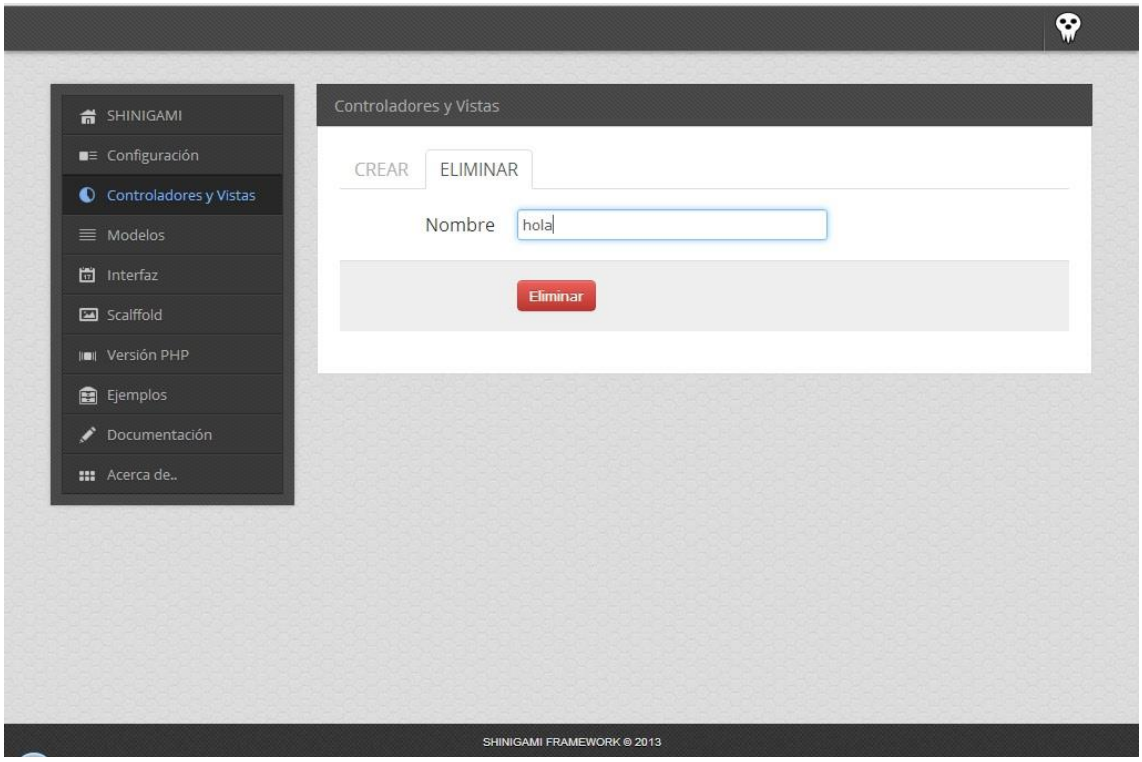
Gráfica N° 3: Menú inicio, sección Controladores y vistas

Para generarlos se debe ingresar el nombre del controlador en el campo de texto como se observa en la gráfica N° 4 y posteriormente oprimir sobre el botón “Generar”. A la vez se podrá visualizar la vista previa de cómo se generaría el controlador.



Gráfica N° 4: Creación de vistas y controladores

También se podrá eliminar un controlador, se debe tener en cuenta que esta acción implica que su vista también será borrada. Para llevar a cabo este proceso se debe ingresar a la pestaña “Eliminar” como se observa en la gráfica N° 5, una vez ingresado a esta sección de igual forma a como se creó el controlador se debe ingresar el nombre del controlador que se desea eliminar en el campo de texto y oprimir sobre el botón “Eliminar”.

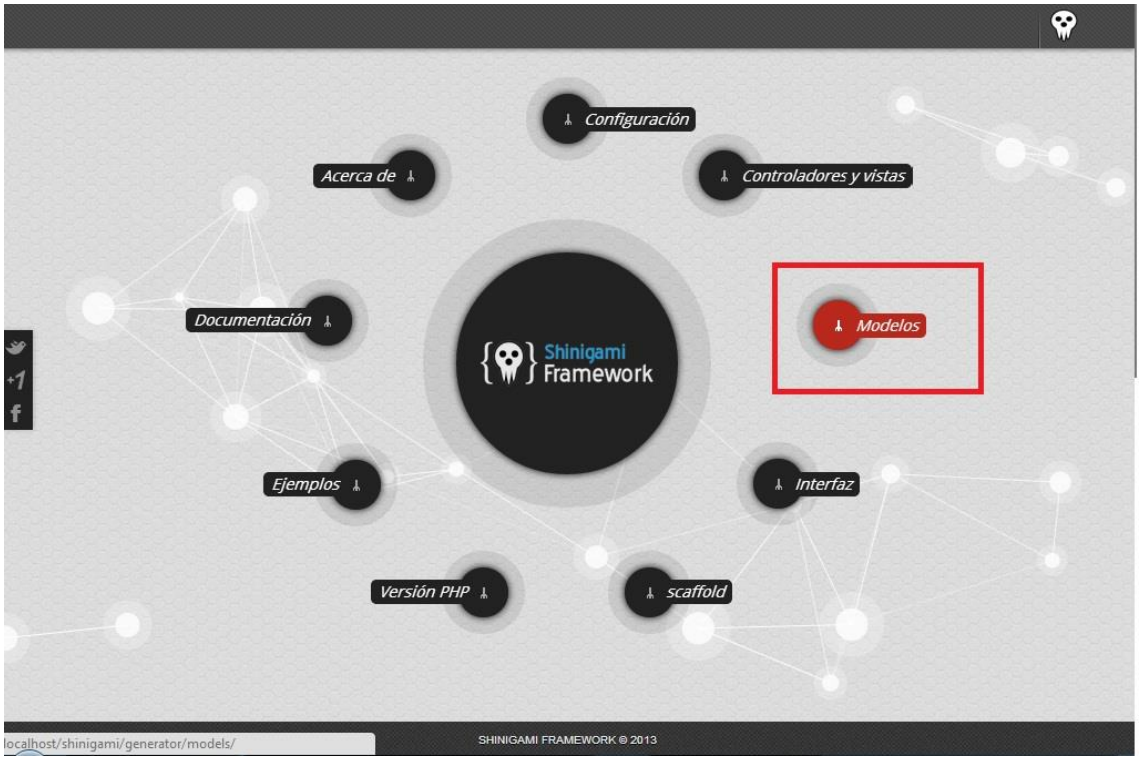


Gráfica N° 5: Pestaña eliminar módulo vistas y controladores

3.3 Modelos

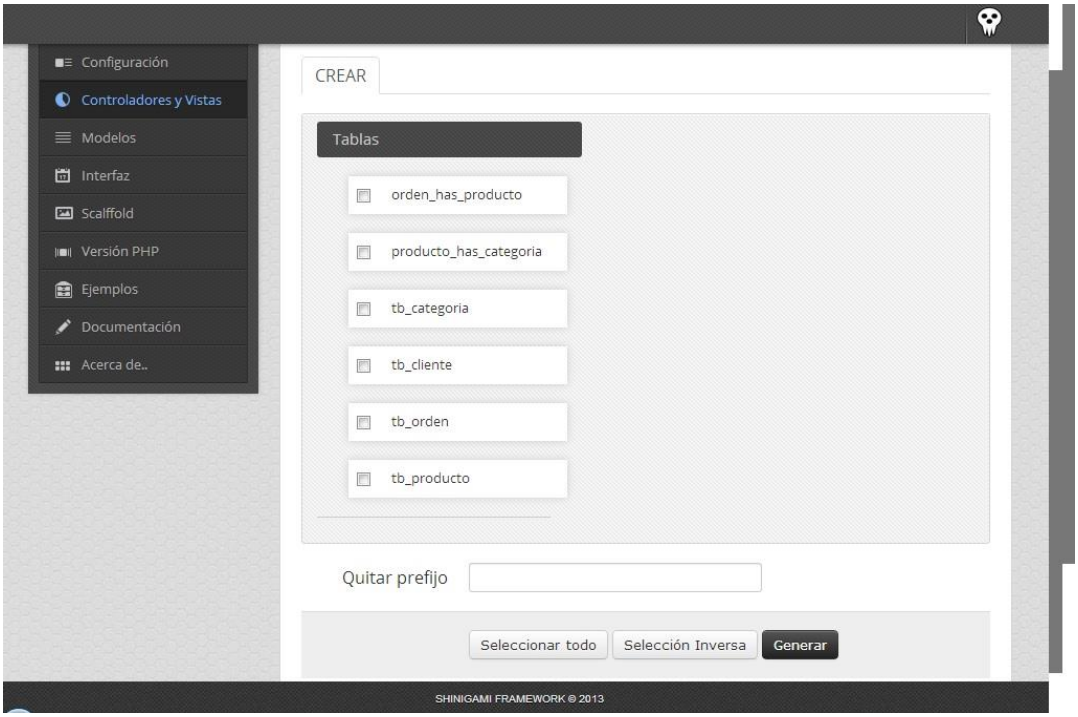
Los modelos en el Framework representan las entidades de las tablas de la base de datos. A través del módulo generator se puede construir la estructura para gestionar los procesos con la base datos implementando el patrón DAO, garantizando la escalabilidad del proyecto.

Para acceder a esta sección desde el menú principal del Framework seleccione “Modelos”.



Gráfica N° 6: Menú inicio, sección Modelos

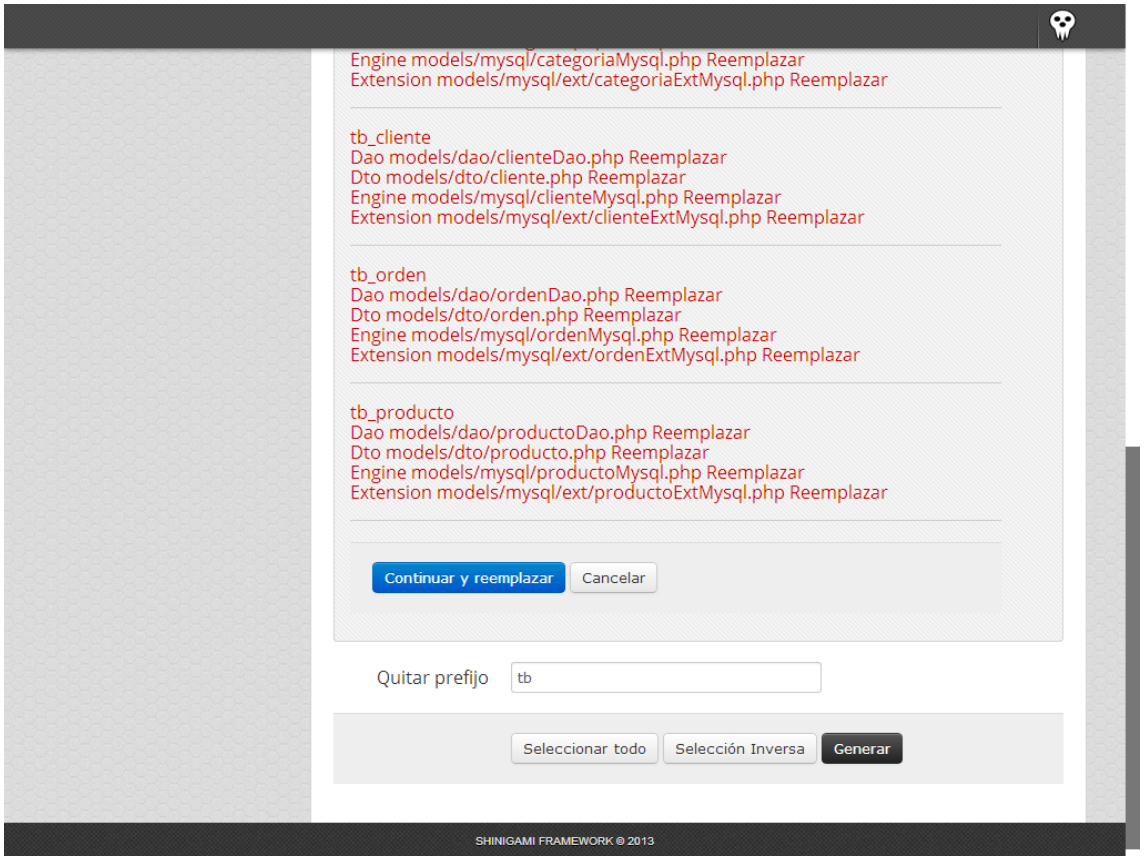
Al ingresar a la sección “modelos” dentro del módulo generator se podrá observar todas las tablas de la base de datos (Gráfica 7).



Gráfica N° 7: Ingreso a la sección modelos

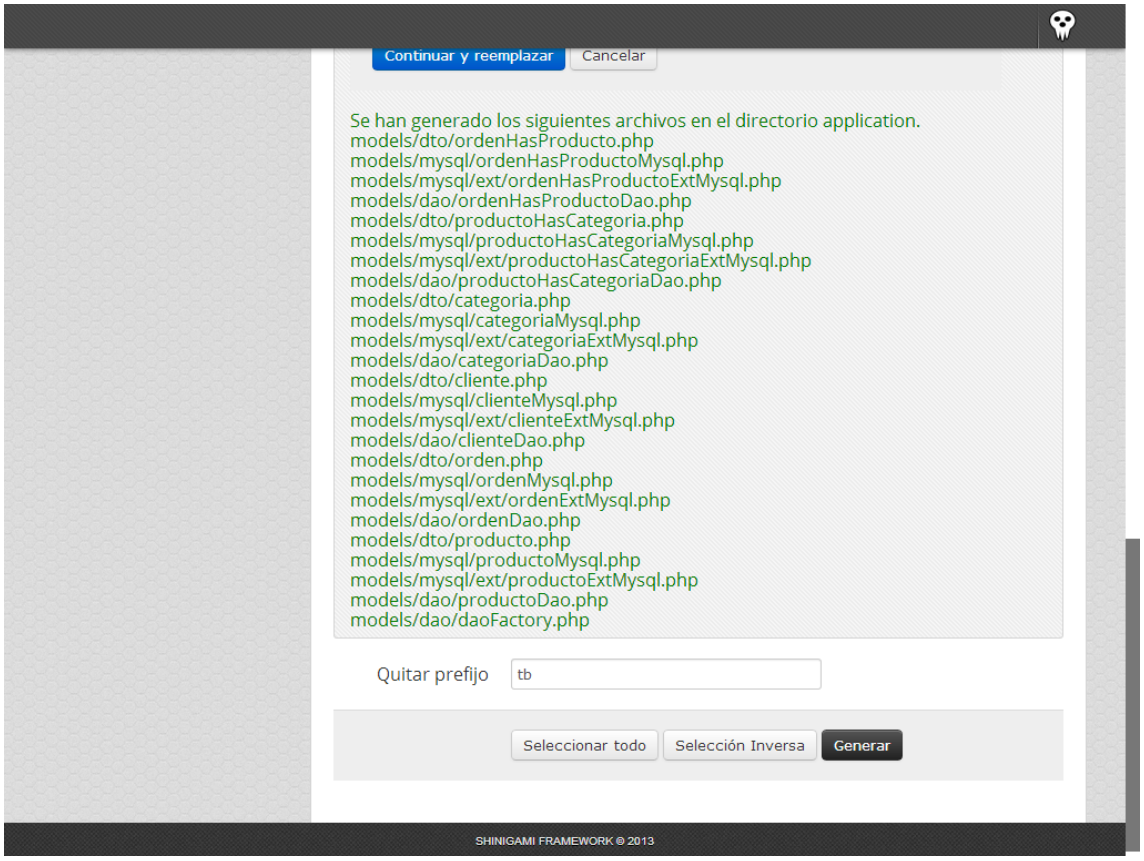
Para construir la estructura del patrón DAO en el Framework se deben seleccionar las tablas sobre las cuales se desea trabajar. Una vez seleccionadas las tablas el desarrollador podrá quitar los prefijos de las tablas, esto con el fin de que los archivos a generar no presenten dicho prefijo que por lo general suele ser “tb”.

Para continuar con el proceso se debe oprimir el botón “Generar”, esta acción buscará conflictos con archivos ya existentes e informará de ellos, en tal caso que existan. Gráfica 8.



Gráfica N° 8: Conflictos de archivos.

Se podrá decidir entre “Continuar y Reemplazar” y “Cancelar”. Si se continúa con el proceso se reemplazaran los archivos ya existentes y se creara la estructura de archivos necesaria para trabajar con la conexión a la base de datos. Al final de proceso se informa cuales archivos fueron creados. Grafico 9.

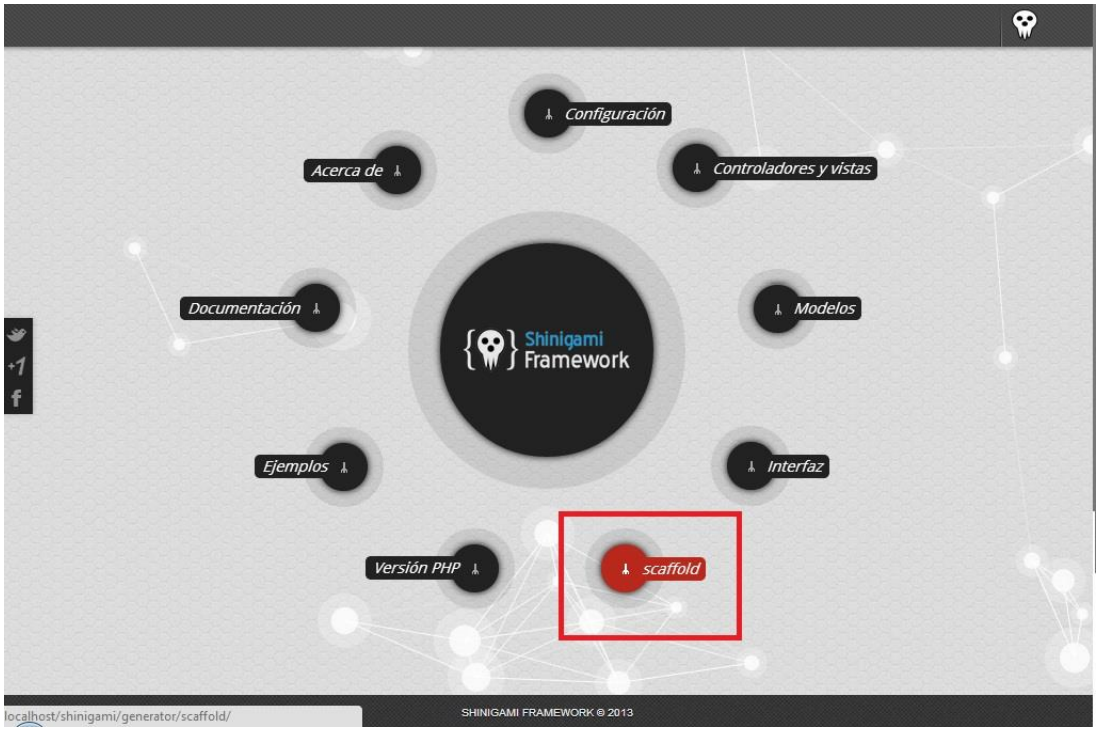


Gráfica N° 9: Archivos creados

3.4 Scaffolding

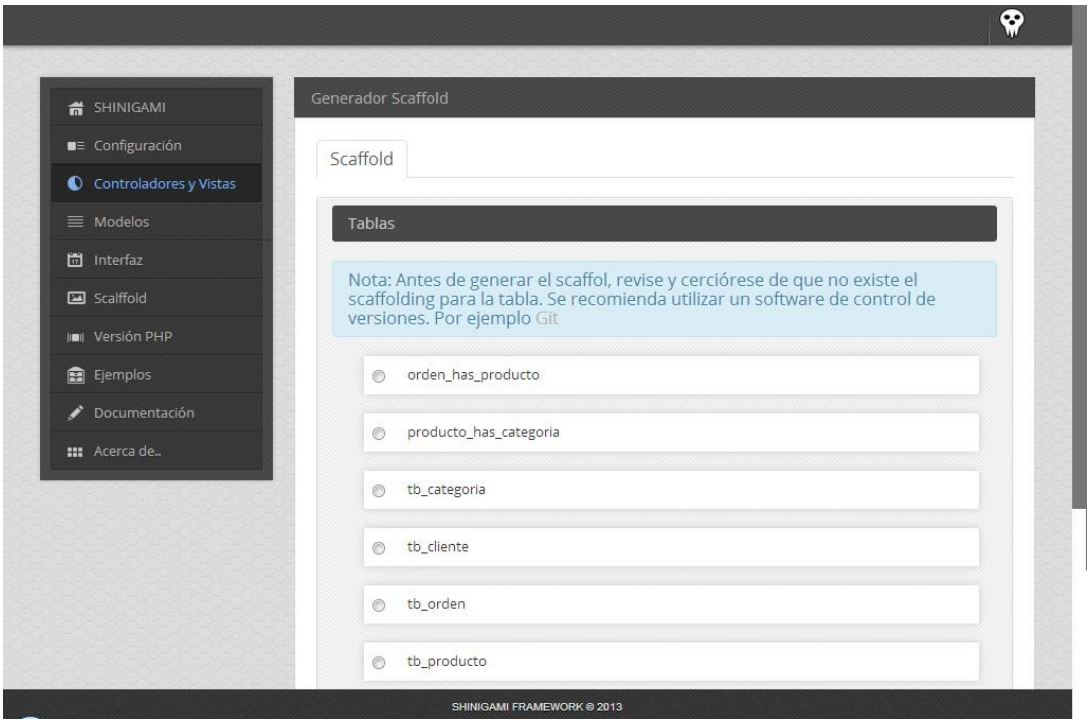
El término conocido como Scaffolding es un método para construir aplicaciones basadas en bases de datos, el cual genera los procesos del CRUD (Create, Read, Update y Delete) de una tabla.

Este método permite ahorrar tiempo en la construcción de vistas, controladores y gestión de procesos en la capa de persistencia. La sección Scaffolding del módulo generator se encarga de generar los componentes necesarios para construir el CRUD de una tabla especificada. Para acceder a esta sección desde el menú principal del Framework seleccione “Scaffold”.



Gráfica N° 10: Menú inicio, sección Scaffold

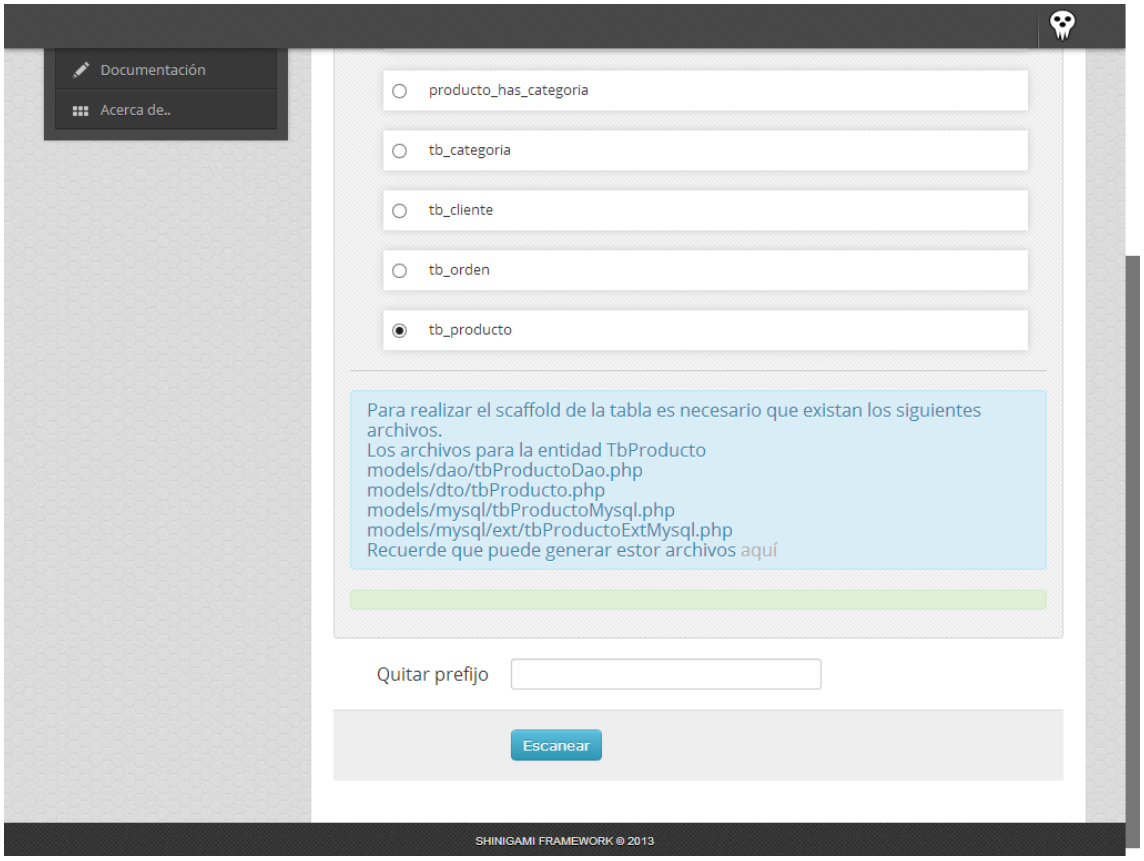
Al igual que en la sección modelos, al ingresar a Scaffolding se listaran las tablas de la base de datos previamente configurada. Grafica 11.



Gráfica N° 11: Ingreso a sección Scaffold

El acción para generar el Scaffolding es similar al de crear modelos, la diferencia radica en que en el Scaffolding solo se podrá seleccionar una tabla a la vez, pero de igual forma se podrá ingresar el prefijo a quitar de la generación. Una vez selecciona la tabla e ingresado el prefijo si se ha optado por quitarlo, se debe oprimir el botón “Escanear”.

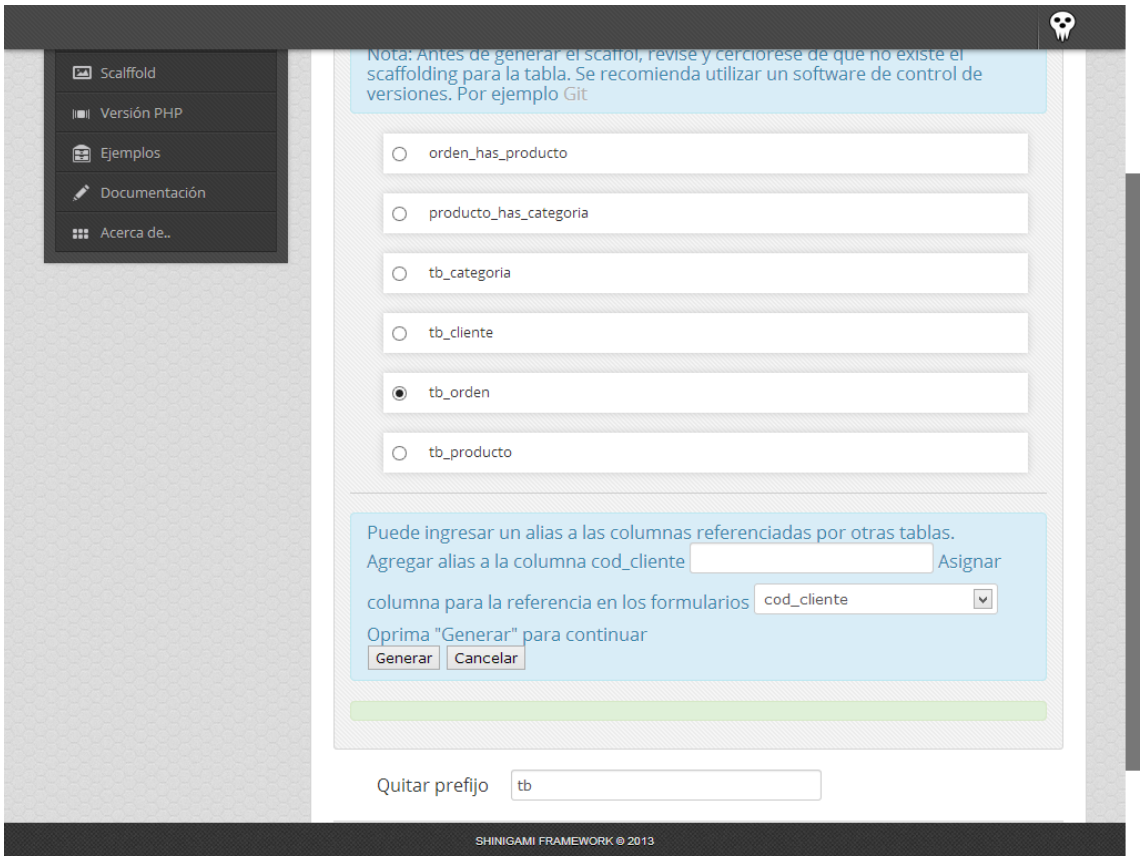
El proceso generado por esta acción, busca las posibles relaciones que pueda tener la tabla con otras dentro del sistema, a partir de esto analiza si ya existen los modelos que se necesitaran para la gestión del CRUD, en caso de que falte algún modelo o archivo necesario para construcción del Scaffolding se informará con mensaje de error. Grafica 12



Gráfica N° 12: Conflictos al escanear scaffold

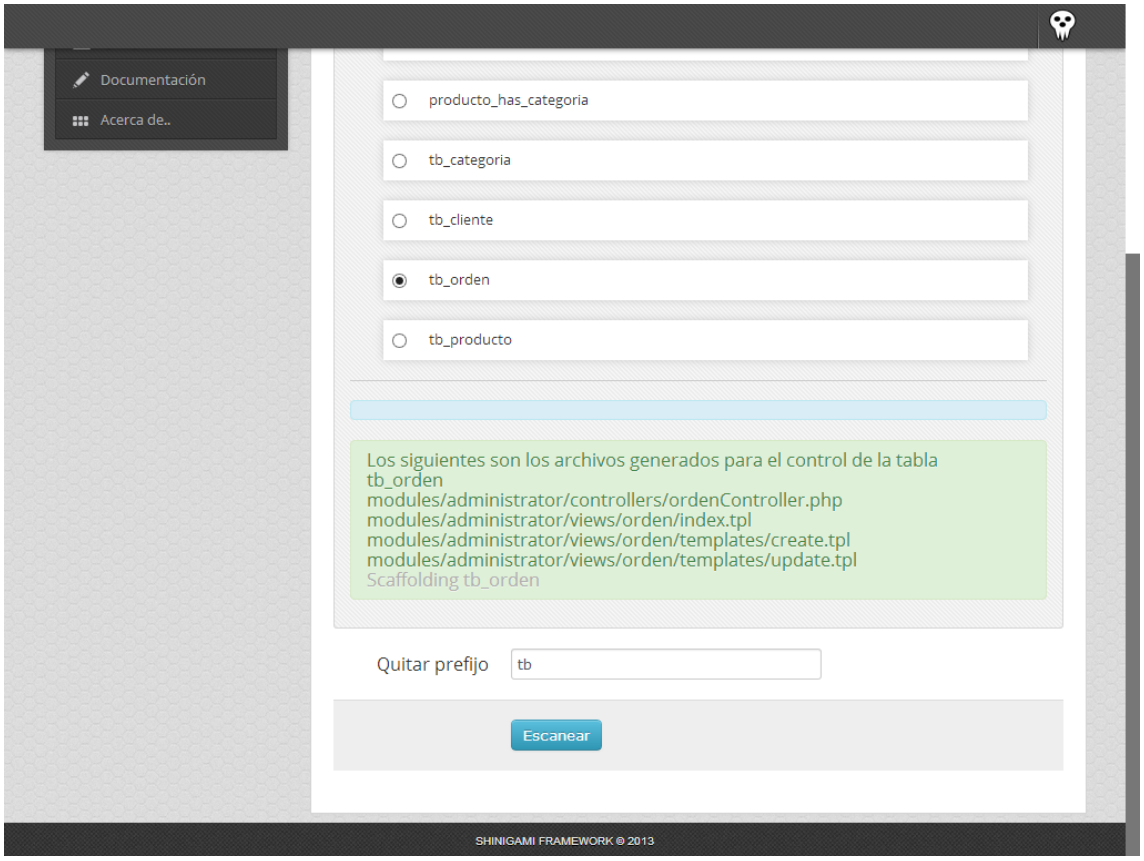
Si el proceso no encuentra conflictos mostrara las posibles opciones a establecer antes de generar, solo se podrán visualizar las opciones cuando la tabla a la cual se le generará el Scaffolding mantenga relaciones con otra

tablas, en tal caso se podrá dar un nombre en específico a la columna que representa una tabal externa. Grafica 13



Gráfica N° 13: Configuración Scaffolding

Una vez configuradas las opciones, cabe resaltar que estas son opcionales, se debe oprimir sobre el botón “Generar”, después del proceso se podrá visualizar los archivos que fueron credos y un enlace en donde quedo integrado el Scaffolding. Grafico 14.

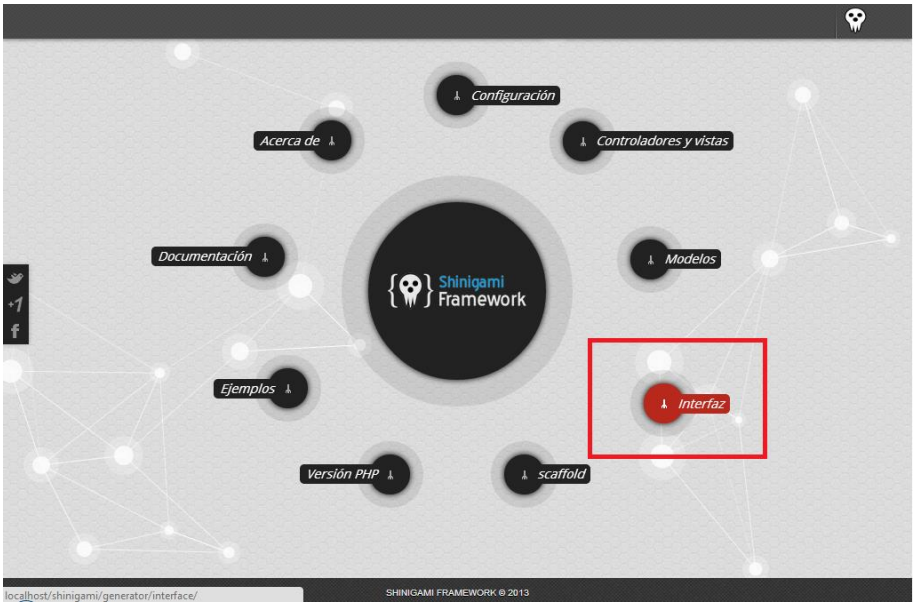


Gráfica N° 14: Finalización Scaffolding

3.5 Interfaz

Las interfaces cumplen un rol indispensable a la hora de navegar por una página web, no sólo por estética sino también desde el punto funcional hacia el usuario. En los últimos años el diseño, funcionalidad y experiencia de usuario en la web ha vuelto fundamental y ha evolucionado a tal punto de llegar a ser visualizada desde cualquier dispositivo que tenga un navegador web.

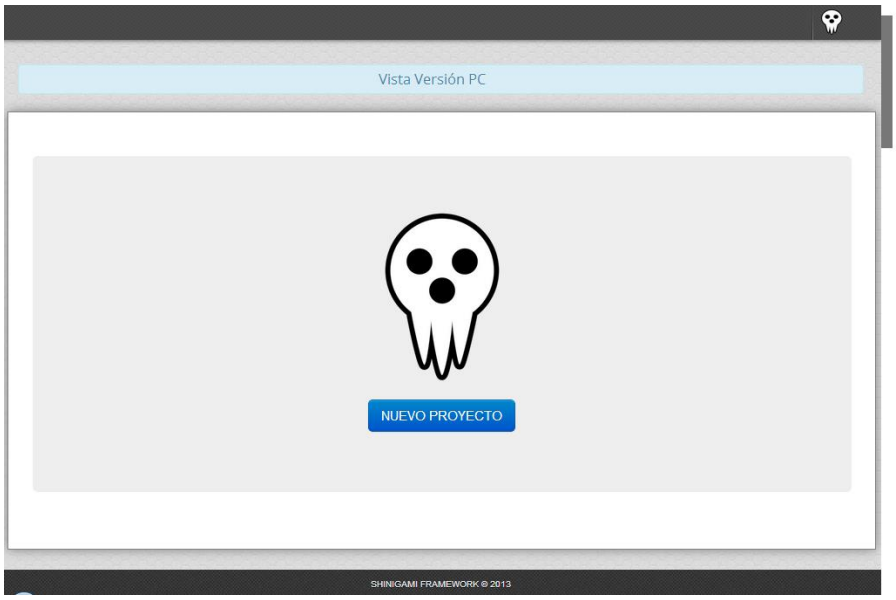
Shinigami Framework Incorpora un módulo para generar “interfaces”, para acceder a éste es necesario acceder a la opción “Interfaz” ubicado en el menú Principal (Gráfica Nº 15 Sección Interfaz).



Gráfica N° 15: Sección Interfaz

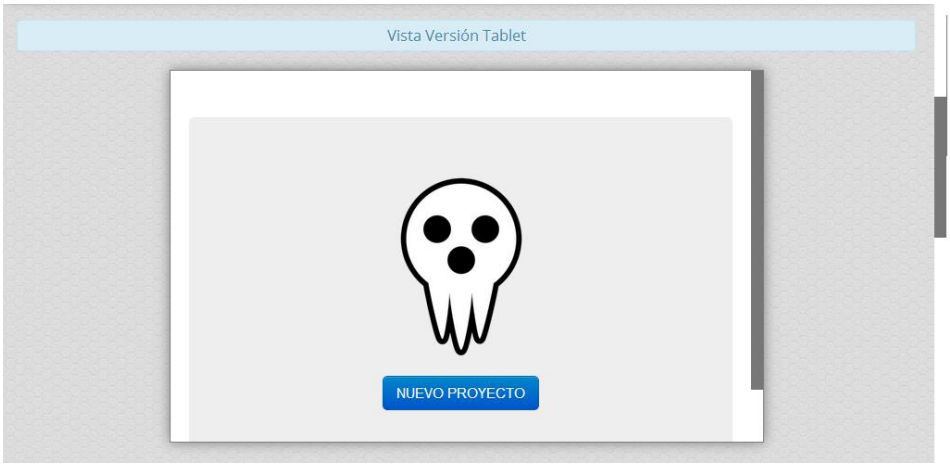
Al acceder a ésta opción, se podrá ver el proyecto en proceso de desarrollo de una manera visual, previamente organizado y clasificado en diferentes de tipos de pantalla dependiendo su resolución, a esto se le denominado diseño adaptable.

- Vista versión PC



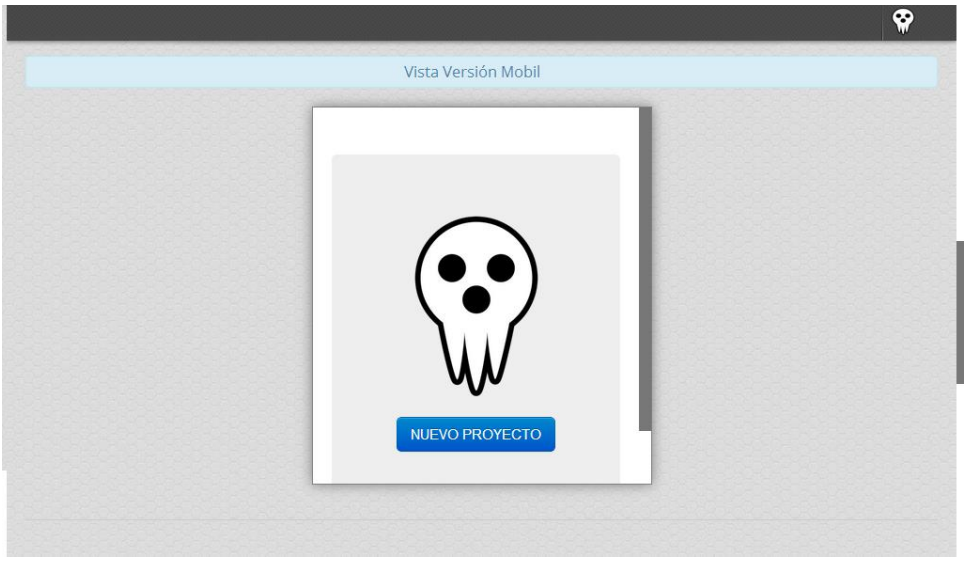
Gráfica N° 16: Diseño para PC

- Vista Versión Tablet



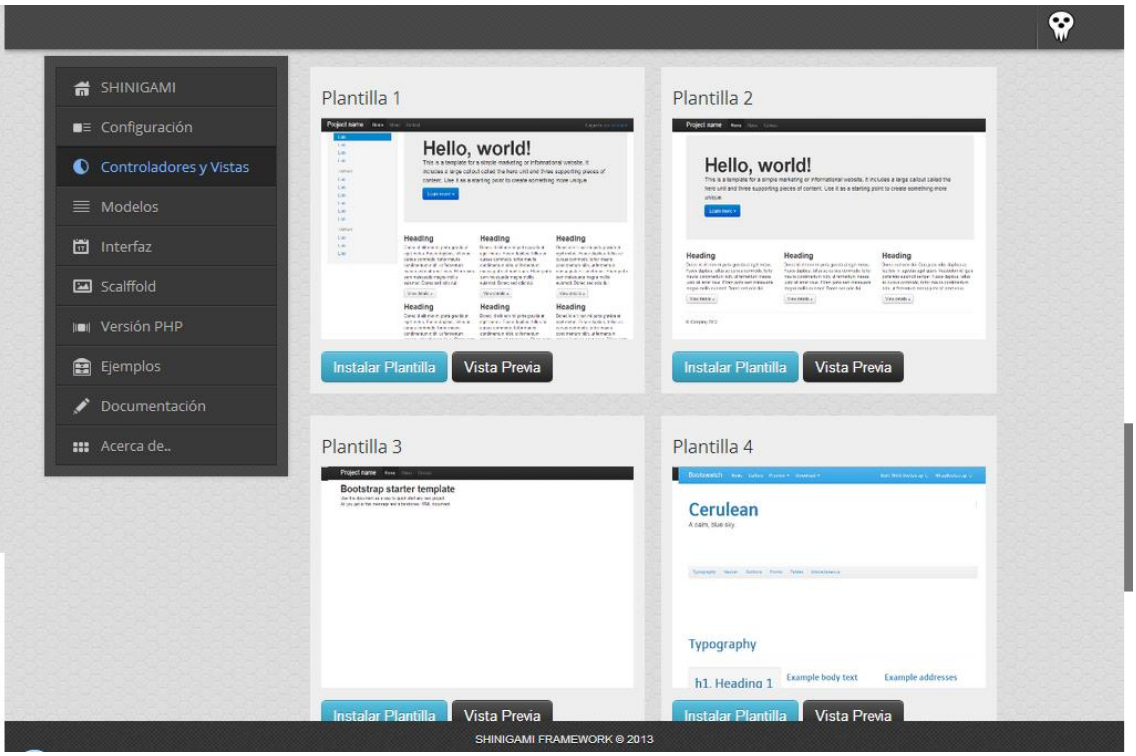
Gráfica N° 17: Diseño para Tablets

- Vista versión Mobil



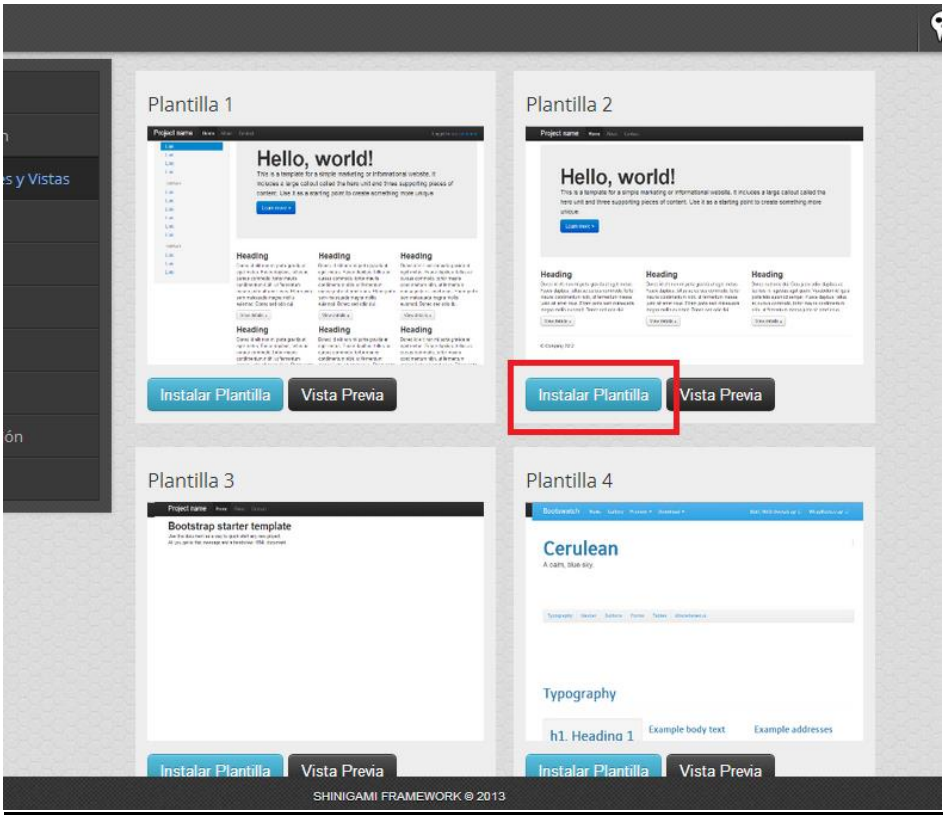
Gráfica N° 18/: Diseño para Mobiles

Enseguida se encuentra la sección de generar plantillas pre terminadas, éstas con el fin de ser utilizadas para el aplicativo en desarrollo. (Gráfica N° 18/: Plantillas pre terminadas). Cada una de éstas plantillas cuenta con 2 opciones, 1) Instalar plantilla, la cual instala automáticamente la plantilla en el proyecto, y 2) Vista previa, dónde es posible primero ver la plantilla antes de instalarla.



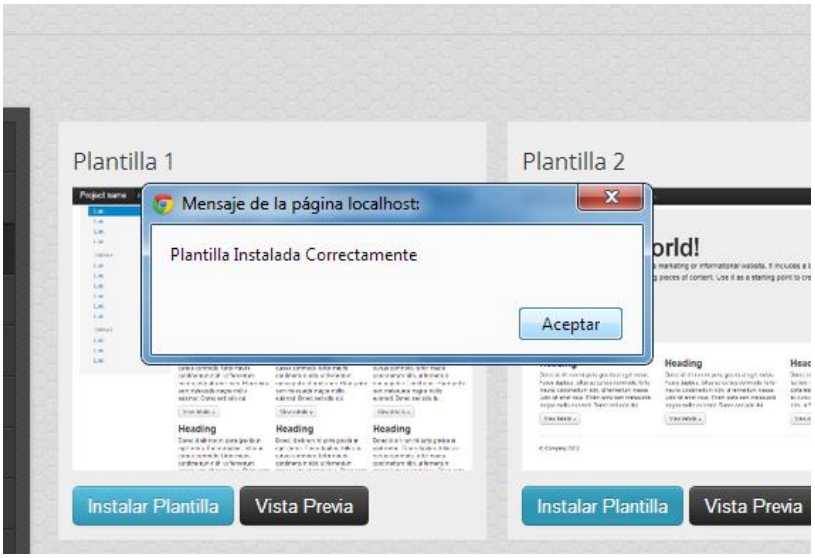
Gráfica N° 19/: Plantillas pretermindas

A continuación se mostrará de una manera detallada como llevar a cabo la instalación de una plantilla base, para ello se debe seleccionar una plantilla y darle click sobre la opción “Instalar Plantilla” (Gráfica Nº 20)



Gráfica N° 20/: Plantillas pretermindas

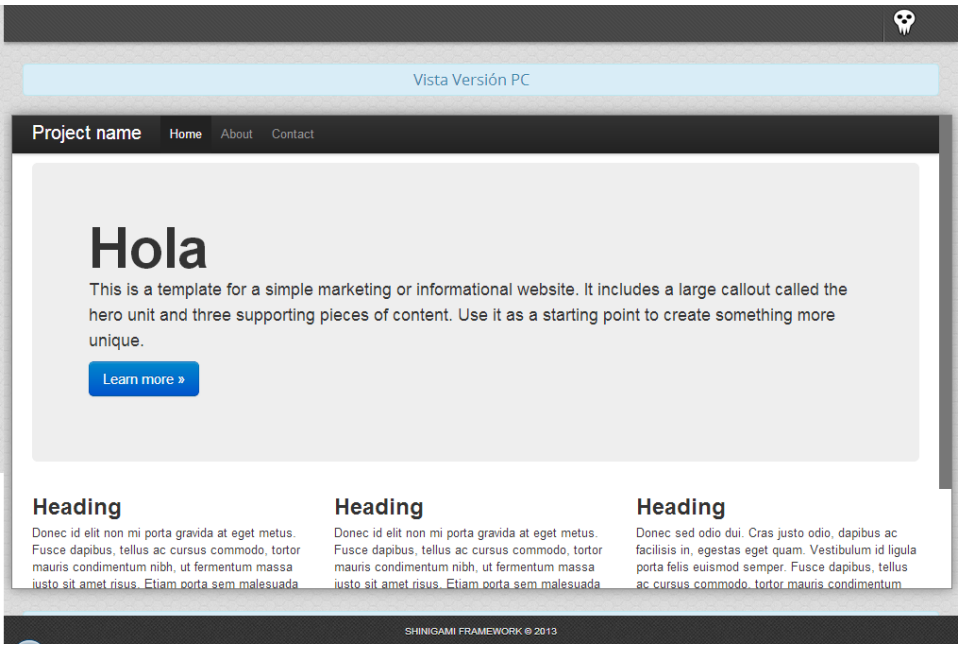
Una vez seleccionada la plantilla e instalada correctamente, se visualizara un mensaje (Gráfica Nº 21) y damos por hecho que la instalación fue satisfactoria.



Gráfica N° 21/: Plantilla instalada correctamente.

Al confirmar la instalación de la plantilla, automáticamente se cargará éste cambio en la parte superior para poder ser visualizada en las diferentes resoluciones:

- Vista Versión PC

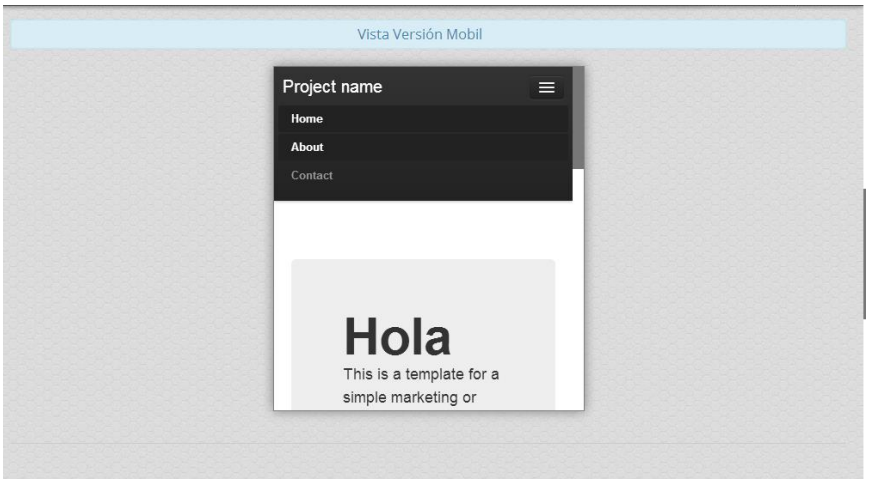


Gráfica N° 22/: Plantilla instalada correctamente.

- Vista Versión Tablet

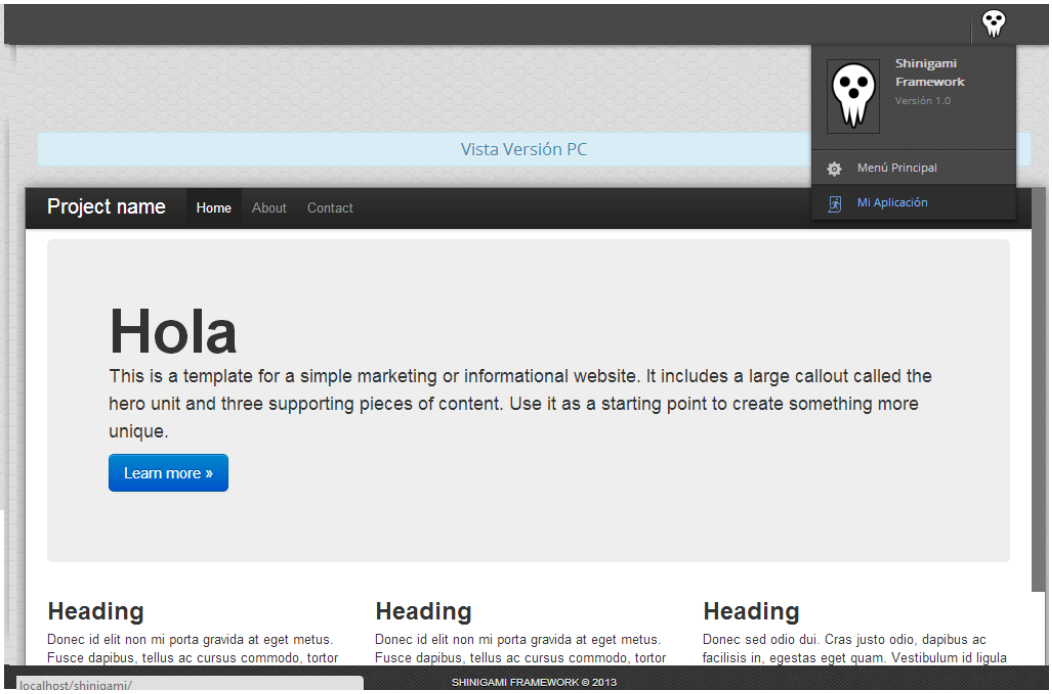
Gráfica N° 23/: Plantilla instalada correctamente.

- Vista Versión Mobil



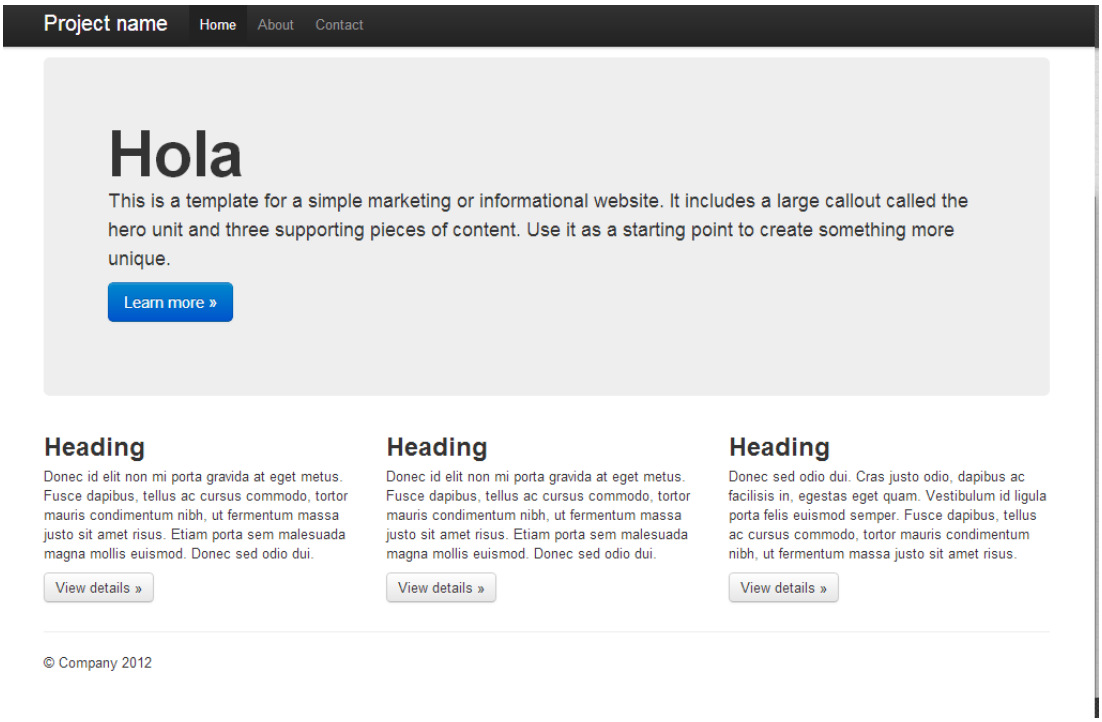
Gráfica N° 24/: Plantilla instalada correctamente.

Como paso final, Nos vamos a dirigir a la Aplicación en desarrollo para verificar dicha instalación, para ello nos dirigimos al menú ubicado en la parte superior derecha, y seleccionar “Mi Aplicación” (Gráfica Nº 25).



Gráfica N° 25/: Ir a la Aplicación en desarrollo

Llevado a cabo el paso anterior, Shinigami Framework se sitúa sobre la aplicación en desarrollo, mostrando la nueva interfaz previamente instalada. (Gráfica Nº 26)



Gráfica N° 26/: Plantilla correctamente instalada, visualizada desde la Aplicación