

Manual del Programador

Sistema de Transacciones
Académicas

Índice general

1	Acerca de este manual	5
1.1	Propósito y audiencia del manual	5
2	Estructura de los directorios	7
2.1	Estructura de los directorios	7
2.1.1	El directorio Config	7
2.1.2	El directorio Data	8
2.1.3	El directorio Public	10
2.1.4	El directorio Vendor	10
2.1.5	El directorio Module	11
3	Módulos del Sistema	13
3.1	Módulos del Sistema	13
3.1.1	El módulo Application	14
3.1.2	El módulo Debug	15
3.1.3	El módulo Solicitud	16
3.1.4	El módulo Usuarios	17
3.1.5	El módulo Visualize	18
3.2	Estructura de un directorio module	19
3.2.1	El directorio config	19
3.2.2	El archivo Module.php	19
3.2.3	El directorio View	20
3.2.4	El directorio src	21
4	El módulo Solicitud	23
4.1	El archivo module.config	24

4.2	El directorio src	24
4.2.1	El directorio Controller	25
4.2.2	El directorio Form	27
4.2.3	El directorio Model	28
4.2.4	El directorio Sapientia	29
4.2.5	El directorio Service	29

1. Acerca de este manual

1.1 Propósito y audiencia del manual

Este manual busca orientar al programador respecto al software implementado para facilitar el proceso de las transacciones académicas. La intención del mismo es guiar al programador de manera a que pueda entender la organización de los directorios y archivos que componen el software del sistema, de manera a que el mismo puede modificar, manipular, extender, mejorar o comprender el software. La estructura de directorios sigue el modelo común de organización de directorios del framework utilizado, Zend Framework 2.

La codificación del software del Sistema de Transacciones Académicas sigue las prácticas comunes de Zend Framework 2 de manera que si el lector tiene entendimiento de las convenciones y forma de funcionamiento de Zend y la arquitectura MVC, esta guía le permitirá entender cómo se implementó un sistema específico dentro del marco de funcionamiento general de Zend Framework 2.

Si el programador que lee este manual tiene conocimiento de Zend Framework 2, notará que el sistema es la puesta en práctica de las prácticas comunes generales de Zend Framework 2 y la arquitectura MVC para desarrollar un sistema específico. Esta guía busca por tanto brindar el entendimiento necesario para que el programador pueda mantener el sistema.

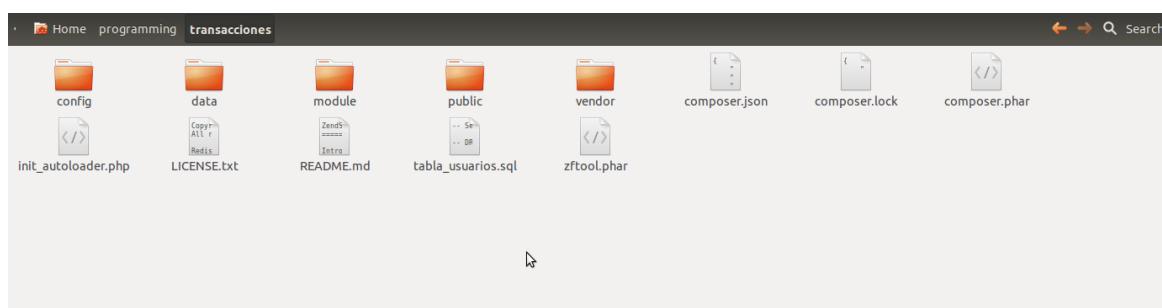
Estructura de los directorios

- El directorio Config
- El directorio Data
- El directorio Public
- El directorio Vendor
- El directorio Module

2. Estructura de los directorios

2.1 Estructura de los directorios

La organización de los directorios se distribuye de la siguiente manera:



2.1.1 El directorio Config

Contiene los siguientes archivos:

Archivo application.config

Especifica los módulos que se encuentran habilitados en el sistema. Este archivo puede ser modificado para editar los módulos que se ejecutarán en el sistema. Permite agregar módulos de manera a ampliar el sistema con nuevos módulos, o bien, deshabilitar módulos que se encuentran actualmente habilitados excluyéndolos de la lista de los módulos utilizados por el sistema.

El directorio autoload

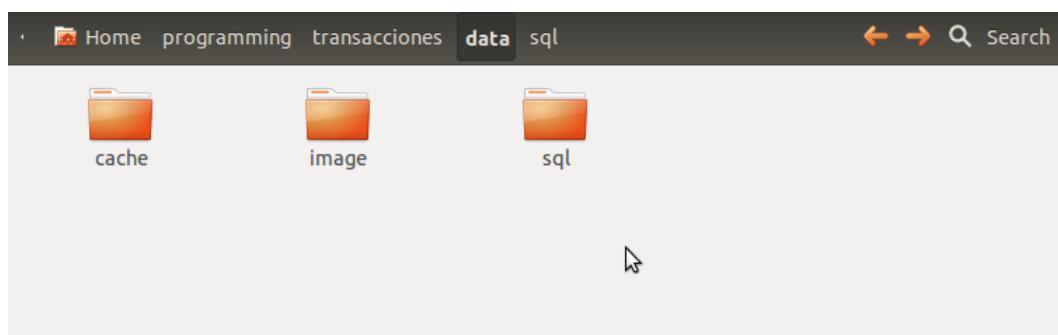
Contiene los archivos de configuración del sistema que son precargados antes de ejecutar cualquier otro código. Contiene declaraciones generales y variables globales del software.

El código que se encuentra en esta carpeta se ejecuta antes que cualquier otro código sea ejecutado. Este directorio es el lugar ideal para archivos de configuración globales que inicialicen módulos o tengan variables globales a ser utilizadas por los mismos. También se incluye en este directorio los parámetros de configuración de la base de datos que deben ser precargados, como puede ser el nombre de la base de datos, el usuario y contraseña de manera a que el adaptador de la base de datos pueda utilizar esta información para autenticar y acceder a la base de datos.

Podemos notar que todos los archivos incluidos en este directorio corresponden a inicialización de parámetros globales de los distintos módulos, necesarios para que los mismos puedan funcionar correctamente y tengan la configuración que permita el funcionamiento deseado por el programador. Las configuraciones incluyen parámetros de inicialización de módulos tanto propios como de terceros (*DOMPDFModule*, *ZfcBase*, *ZfcUser*, *ZfcAdmin*, *ZfcUserAdmin*, *GoalioMailService*, *GoalioForgotPassword*, *BjyAuthorize*, *RoleUserBridge*).

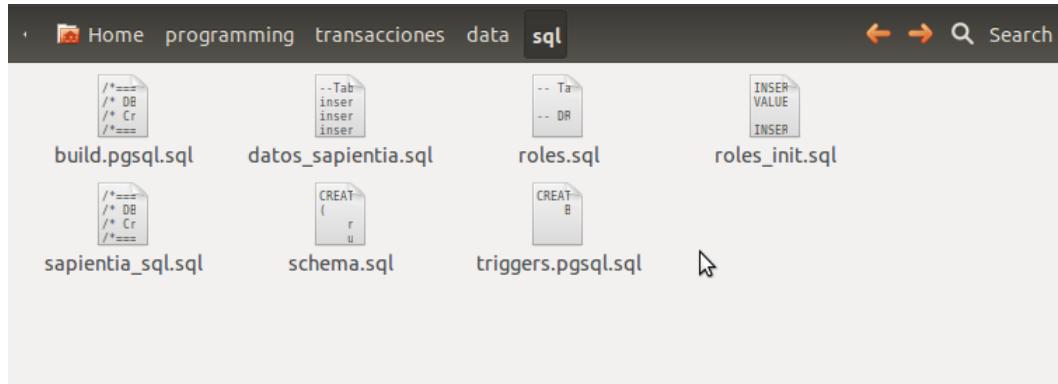
2.1.2 El directorio Data

Archivos en general a ser utilizados por el sistema.



Contiene los directorios:

- **Cache:** no se utiliza.
- **Image:** no se utiliza. Aquí podrían ir imágenes que se desean guardar del sistema, como por ejemplo, cuando el usuario desea adjuntar imágenes como archivos adjuntos, fotos de perfil.
- **Sql:** contiene los archivos SQL correspondientes a la creación de las bases de datos. Este directorio debe contener todos los archivos SQL correspondientes a la creación de tablas, creación de triggers de la base de datos.



El directorio Sql contiene los archivos que corresponden a la creación de la base de datos así como de su inicialización:

- **build.pgsql** el cual debe ejecutarse para la creación de la base de datos. Si se desea cambiar la base de datos, este archivo debe ser reemplazado por otro que contenga la nueva definición SQL de la base de datos.
- **triggers.pgsql.sql** Crea los triggers necesarios para el correcto funcionamiento de la base de datos con el resto del sistema.
- **sapientia_sql** el cual contiene la definición de la base de datos de Sapientia.



Importante: este archivo es una aproximación de la base de datos de Sapientia y debe ser eliminado en la práctica cuando se realice en forma efectiva la conexión con el sistema de Sapientia.

- **datos_sapientia** contiene datos de prueba de Sapientia. Contiene código SQL para insertar casos de prueba que permitan simular, emular los datos el sistema Sapientia.
- **roles.sql** contiene el SQL que permite la creación de tablas utilizadas por el módulo ByAuthorize de manera a otorgar roles a los diferentes usuarios.
- **roles_init.sql** contiene el SQL que al ser ejecutado inicializa las tablas creadas con el archivo roles.sql según los roles definidos y establecidos para el sistema actual de Transacciones Académicas. Este archivo permite crear los distintos roles para los diversos usuarios que serán utilizados para otorgar o denegar permisos en el sistema: administrador, guest, user, funcionarios y alumnos.
- **schema.sql** contiene el SQL que permite crear la tabla requerida y utilizada por el módulo GoalioForgotPassword que permite a un usuario realizar una solicitud de reestablecimiento de contraseña.

2.1.3 El directorio Public



El directorio public junto con todos sus archivos y subdirectorios serán las únicas entradas directamente accesibles desde el web browser. Este es el directorio donde se deben incluir archivos CSS, imágenes y archivos Javascript.

R Importante: No se debe incluir en este directorio ninguna información sensible tales como credenciales de usuario o números de tarjeta de crédito, contraseñas.

Este directorio contiene el archivo index.php el cual es el archivo principal que es consultado cuando el usuario ingresa a la página principal del sistema. Todas las peticiones y solicitudes del navegador web son redireccionadas a este archivo. Este archivo debe mantenerse tan ligero como sea posible ya que será ejecutado en cada solicitud de cualquier cliente. Es este el archivo donde el motor de Zend Framework 2 es cargado y ejecutado.

R Nota: No se recomienda realizar cambios en este archivo. Los cambios a la página web deben realizarse en los módulos, los cuales se encuentran en los directorios module y vendor.

2.1.4 El directorio Vendor

Este directorio contiene exclusivamente módulos de terceros, esto es, módulo pre-hechos por terceros. Estos módulos deben ser utilizados tal cual están, sin que el programador de la aplicación que los utiliza deba modificarlos. Las funcionalidades de estos módulos son utilizadas a través de su interfaz, realizando las configuraciones necesarias de estos módulos en el directorio config/autoload.

R Importante: En general, los archivos de los módulos de esta carpeta no deberían ser modificados. No se recomienda modificar los archivos de este directorio a menos que se tenga conocimiento claro de la implementación del módulo por parte de los terceros. Los módulos de terceros suelen tener sus archivos de configuración a ser modificados en el directorio config/autoload de manera a especificar parámetros de configuración a los mismos para conseguir el comportamiento deseado de todas las posibles configuraciones posibles ofrecidas por el módulo.

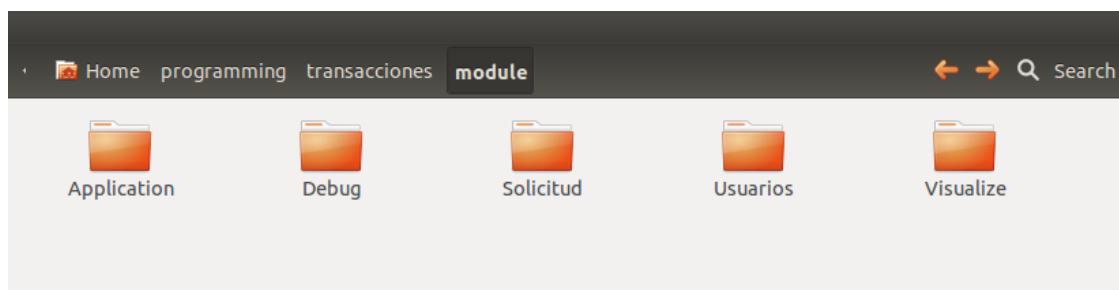
Como este directorio corresponde a módulos de terceros, reutilizados tal cual están hechos, y como los mismos no son hechos por el equipo de programación del Sistema de Transacciones Académicas, no se necesita dar más explicación sobre este directorio.

Entre los módulos de terceros utilizados se incluyen:

- **DOMPDFModule:** utilizado para la generación de PDF.
- **ZfcBase:** módulo requerido por los demás módulos desarrollado por ZF-Commons, la comunidad que brinda módulos estándar de Zend Framework 2. Contiene definiciones comunes necesarias para los otros módulos.
- **ZfcUser:** permite la gestión de usuarios del sistema como el registro, autenticación del usuario. Desarrollado por la comunidad de ZF-Commons.
- **ZfcAdmin, ZfcUserAdmin:** otorgan funcionalidades de administración del sistema incluida la gestión de los usuarios del sistema. Utilizado principalmente para implementar las funciones a cargo del administrador del sistema. Desarrollado por la comunidad de ZF-Commons.
- **GoalioMailService:** utilizado para el envío de emails. Se encarga de lidiar con los protocolos de email para brindar una interfaz de envío de emails.
- **GoalioForgotPassword:** módulo que permite la solicitud por parte del usuario de la recuperación de contraseña. Utiliza el módulo GoalioMailService para enviar el email de reestablecimiento de contraseña al usuario. Permite al usuario reestablecer una nueva contraseña tras ocurrir la pérdida de contraseña olvidada.
- **BjyAuthorize:** módulo que permite restringir los usuarios el acceso a las distintas páginas según los permisos asignados a su rol.
- **RoleUserBridge:** permite asignar el rol por defecto a un usuario. Utilizado para asignar automáticamente el rol por defecto a un usuario cuando el mismo se registra. El mismo es una extensión BjyAuthorize para la asignación automática de roles.
- **zendframework:** directorio que contiene la librería de Zend Framework 2.

2.1.5 El directorio Module

Contiene todos los módulos desarrollados por el equipo de desarrollo de software del Sistema de Transacciones Académicas.



Debido a que este directorio es el principal, será detallado en el siguiente capítulo.

Módulos del Sistema

El módulo Application

El módulo Debug

El módulo Solicitud

El módulo Usuarios

El módulo Visualize

Estructura de un directorio module

El directorio config

El archivo Module.php

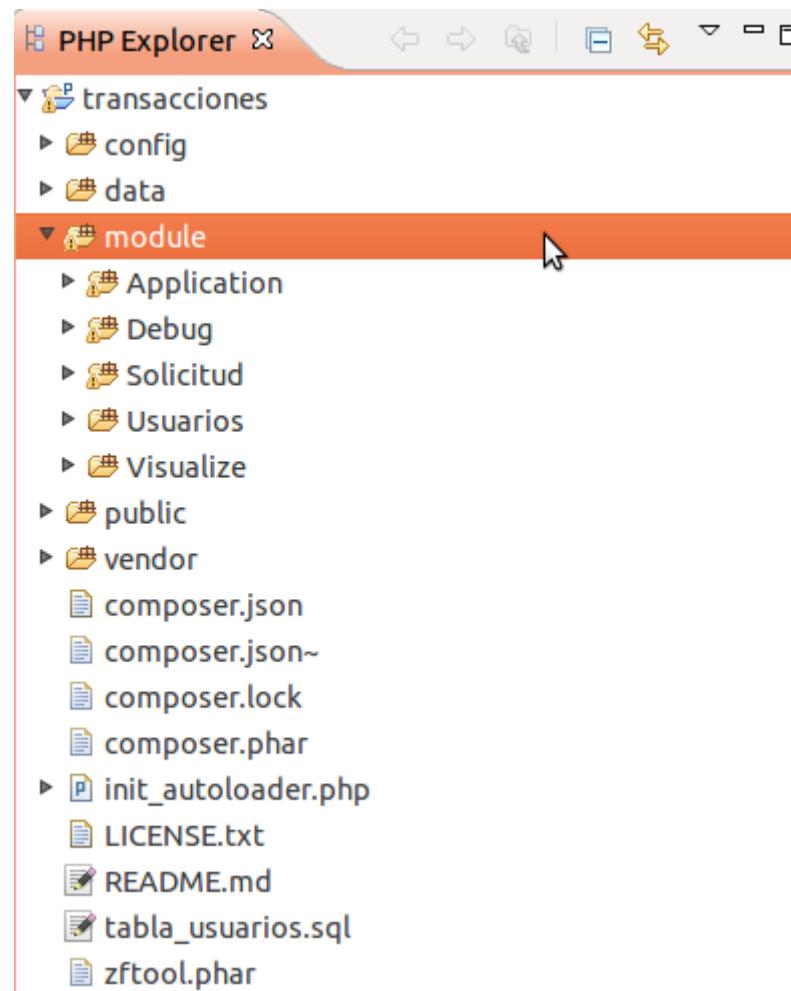
El directorio View

El directorio src

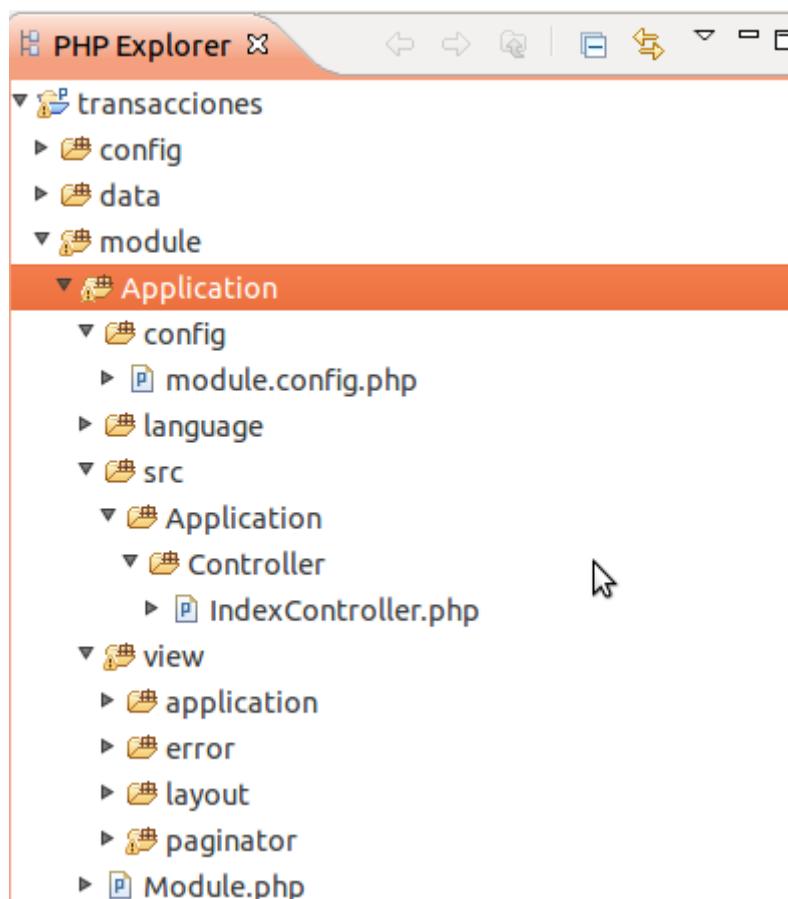
3. Módulos del Sistema

3.1 Módulos del Sistema

Los módulos del sistema implementados son los siguientes:



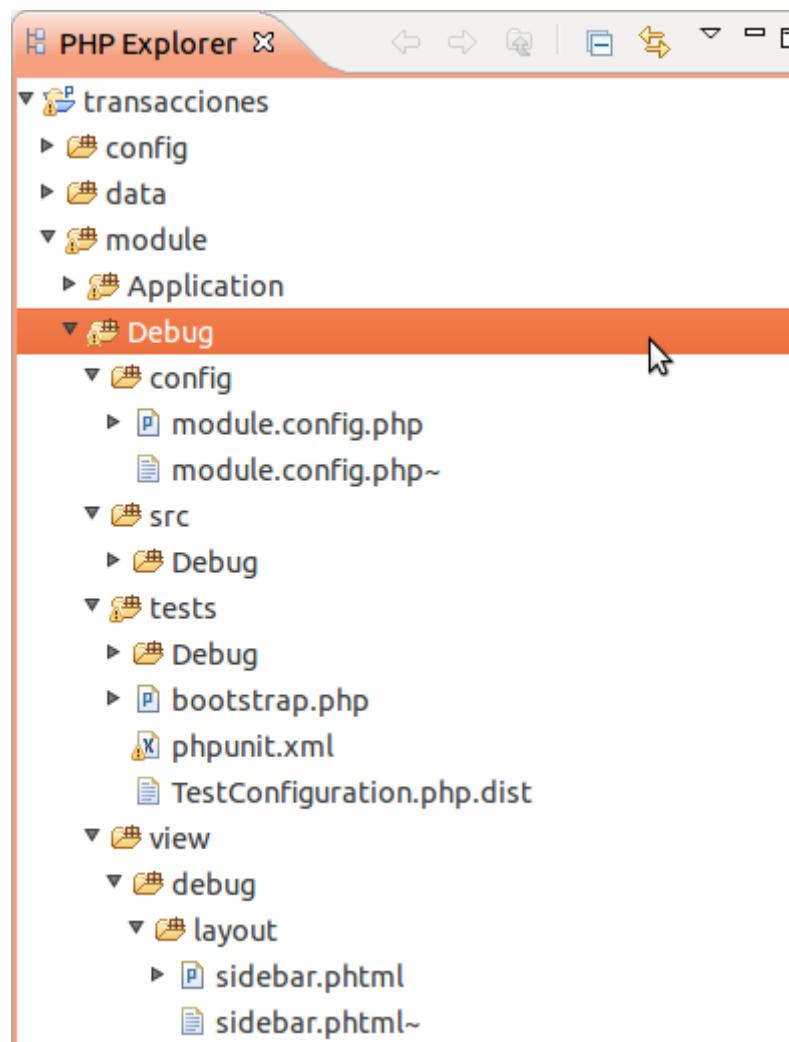
3.1.1 El módulo Application



Contiene las páginas y utilidades generales del sistema. Entre las páginas que incluye se encuentran el index, las páginas de ayuda y las páginas de descripción Acerca De del sistema. Contiene el template de la página de error por defecto a mostrarse cuando ocurra cualquier error.

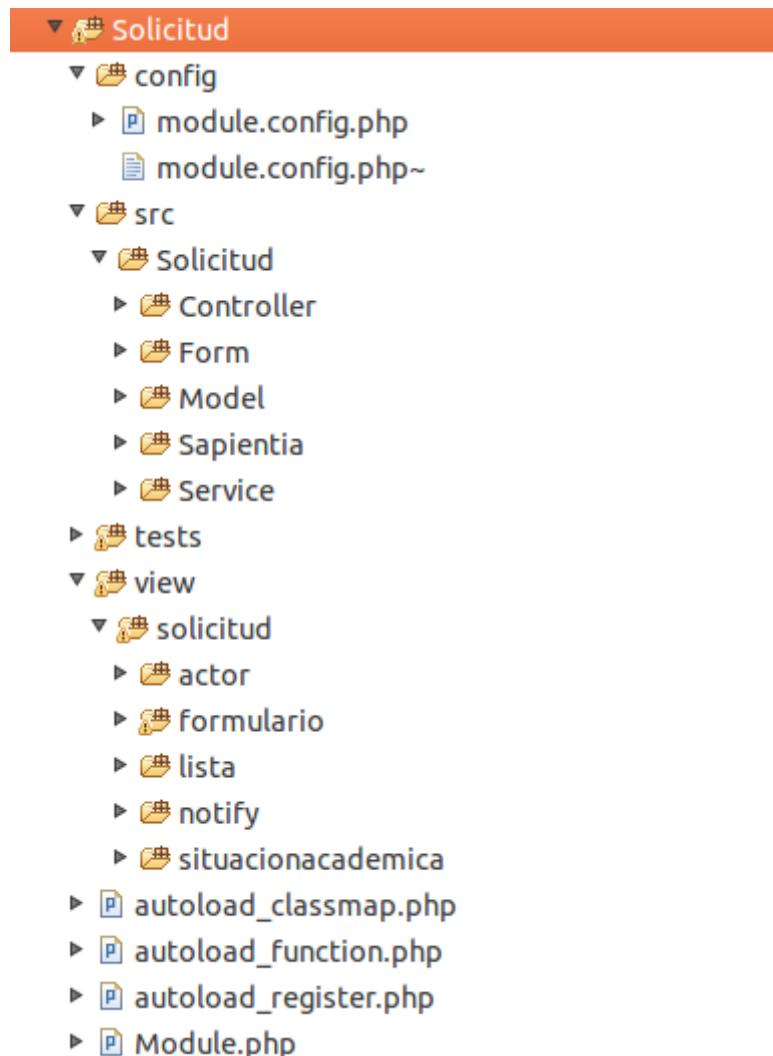
Entre las utilidades que provee se encuentra el template que corresponde a la vista del paginator el cual es el encargado de mostrar listas en pantallas con las opciones de "paginar", o sea, ir a la página de atrás, siguiente, primera o última. Este es utilizado para desplegar listas en varias partes del sistema. Este directorio también incluye el layout, es el recuadro que envuelve a cualquier página que se muestre al visitante de la página.

3.1.2 El módulo Debug



Contiene utilidades para crear logs del sistema que permitan realizar un análisis del funcionamiento del sistema. No es un módulo muy importante y se puede prescindir del mismo deshabilitándolo si se desea.

3.1.3 El módulo Solicitud

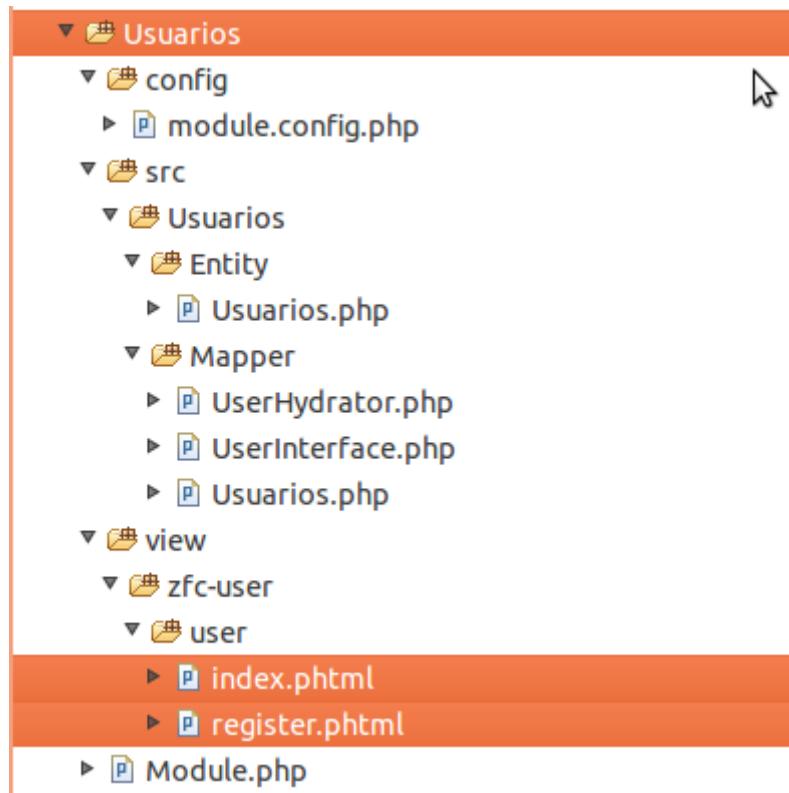


Contiene todas las páginas, lógica y vistas referentes a las solicitudes. Esto incluye los formularios de solicitudes a completar, las vistas que muestran las listas de solicitudes por actor como la lista de solicitudes hechas por el alumno y la lista de solicitudes a resolver por los funcionarios. También incluye la vista de una solicitud particular con los datos de la misma. En el caso del funcionario, puede visualizar una solicitud para poder tomar una acción o resolución a la misma.

En el caso del alumno, ver un resumen de una solicitud hecha junto con el estado actual de la misma. Este módulo incluye la visualización de la situación académica del alumno por parte del funcionario. También incluye la lógica que permite recabar datos del alumno alojados en el sistema Sapientia y que son accesibles a través de sus servicios publicados, junto con el correspondiente cliente de los servicios que permite la conexión y obtención de los mismos. Los cálculos que permiten la verificación automática de los requisitos se realizan en este módulo a partir de los datos obtenidos de Sapientia.

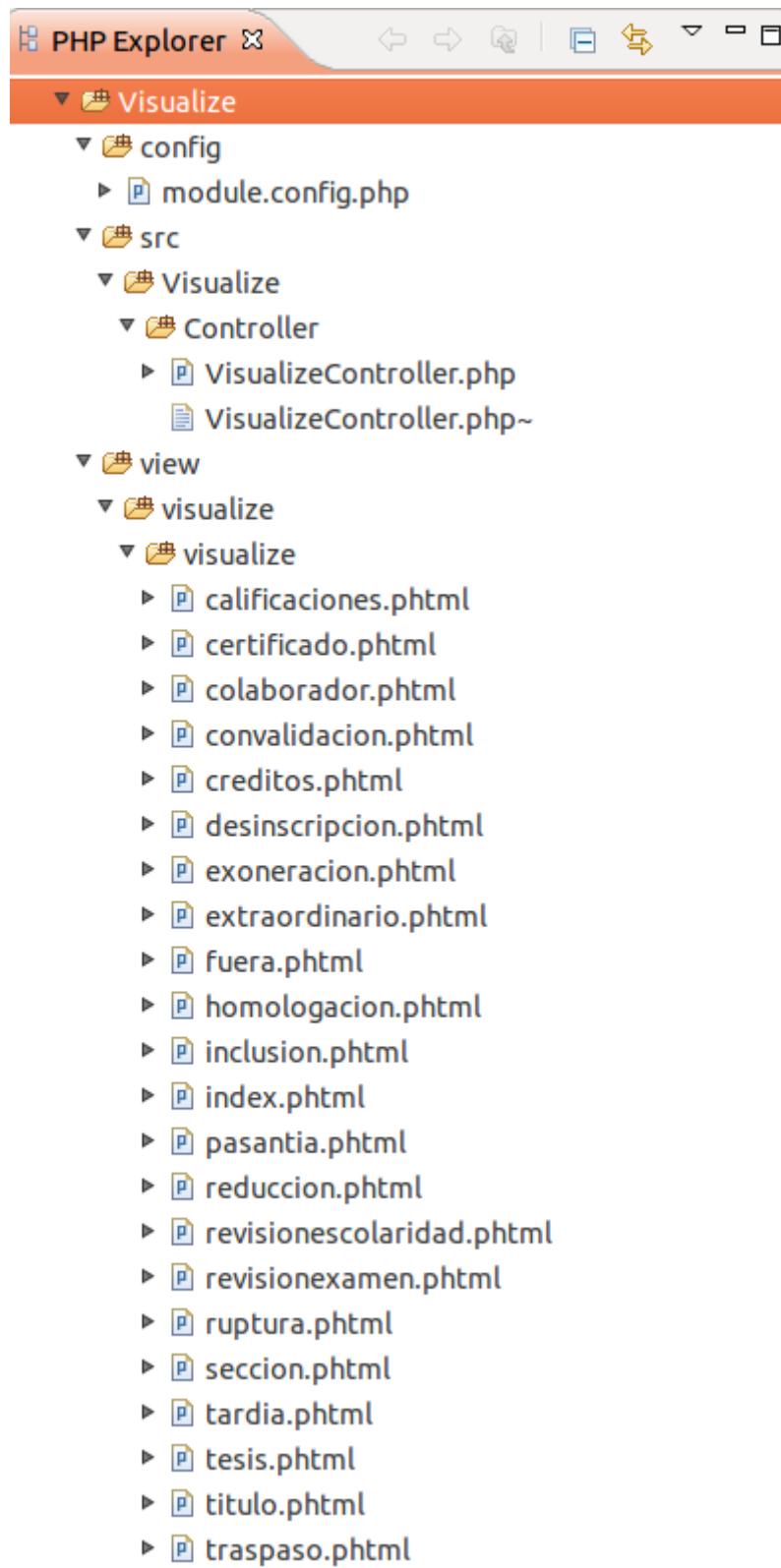
Debido a que es el módulo más importante, se verá en detalle en el siguiente capítulo.

3.1.4 El módulo Usuarios



Contiene las páginas referentes a las cuentas de los usuarios. Contiene la página de Registro del Usuario y la página principal del usuario, entre otros. Está muy relacionado y es dependiente del módulo ZfcUser, el cual es un módulo de terceros que se encuentra en el directorio Vendor del sistema, descrito más arriba. De hecho, solo sobreescribe ciertas páginas del módulo ZfcUser para adaptarlo al contexto del sistema de Transacciones Académicas, dejando lo demás intacto.

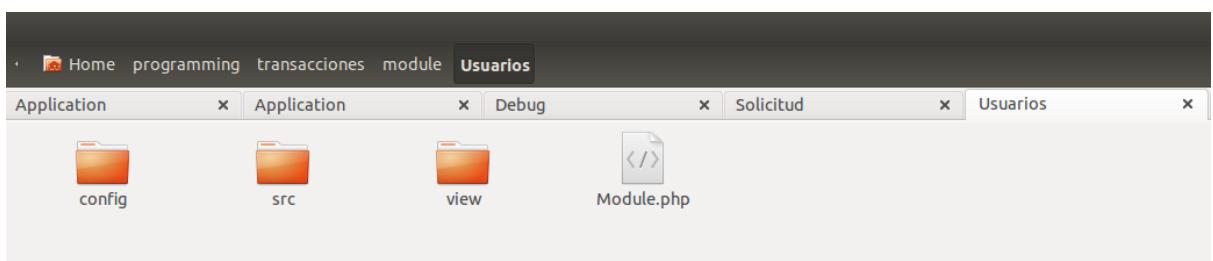
3.1.5 El módulo Visualize



Se utiliza para generar los PDF que corresponden a las distintas solicitudes. Estos PDF pueden imprimirse para obtener la versión en papel de las versiones digitales de las solicitudes de manera a que el solicitante y las autoridades puedan firmarlos en el proceso de resolución de las solicitudes académicas. Contiene páginas que reciben como parámetros datos de la solicitud que debe desplegar, y utilizan estos datos recibidos para llenar la información de la solicitud (completar datos en la plantilla modelo que se utiliza para generar el PDF, versión digital de la solicitud de papel) específica que se desea realizar.

3.2 Estructura de un directorio module

Cada directorio alojado en el directorio Module corresponde a un módulo del sistema. Cada módulo contiene en general los directorios y archivos de la siguiente figura



3.2.1 El directorio config

Este directorio contiene el archivo **module.config**, el cual es el archivo principal del configuración del módulo. En este archivo se definen las utilidades que el módulo provee como por ejemplo, servicios, vistas, routing, controllers y más. En este módulo también se puede especificar otros archivos de configuración. Estos describen lo que puede ser reemplazado en este módulo para mejorarlo según las necesidades del ambiente del application environment y la aplicación misma. Estos archivos adicionales de configuración se deben copiar al directorio config/autoload, los archivos que residen en este directorio servirán solo de documentación (que es muy importante para los programadores que utilicen este módulo).

3.2.2 El archivo Module.php

Cualquier módulo debe tener como requisito mínimo este archivo. En el mismo se especifica la configuración del módulo y la localización de las clases que pertenecen a este módulo para ser cargadas automáticamente por el framework.

Los métodos ubicados en este archivo responsables de obtener la configuración del módulo y de cargar automáticamente estas configuraciones se llaman **getConfig** y **getAutoloaderConfig** respectivamente. Normalmente estos códigos no suelen modificarse.

En algunos módulos suele añadirse un método llamado **onBootstrap** el cual es un método que se ejecuta tras cargarse todos los módulos, iniciarse la aplicación, realizarse todas las configuraciones y los módulos son arrancados o inicializados. Es extremadamente útil ya que se pueden añadir detectores de eventos que ocurren en el sistema que realicen acciones especiales cuando estos ocurren.

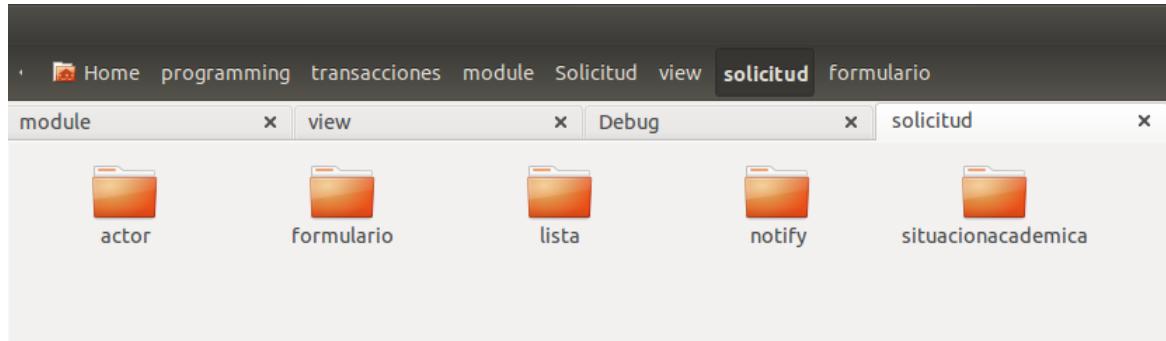
Los últimos dos directorios importantes que quedan por mencionar son el *src* y el *View*.

Zend utiliza la arquitectura MVC (Model View Controller). En la carpeta *src* suelen residir tanto los controllers como los models del sistema. En el directorio *View* se encuentran los archivos correspondientes a la parte View de la arquitectura MVC.

3.2.3 El directorio View

Contiene los archivos correspondientes a la presentación o vistas de las páginas. Los Views reciben como parámetros de los Controllers los datos a ser mostrados en el navegador y contienen la lógica que permite mostrar los mismos, contiene generalmente código CSS, PHP y HTML.

Por ejemplo, tomemos como caso de estudio representativo el view del módulo Solicitud el cual es el módulo más importante del sistema.



El mismo contiene los distintos directorios que alojan las vistas relacionadas para las distintas partes específicas del sistema.

Si tomamos como ejemplo, el directorio *actor*, podemos visualizar los archivos *visualizarResolucionSolicitud.phtml* y *visualizarSolicitud.phtml* los cuales son los responsables de las vistas del alumno de la resolución tomada a su solicitud y la visualización de la solicitud por parte de algún funcionario (decano, directores, secretarías, recepción) respectivamente.

Estos Views reciben los parámetros o variables que les serán útiles para desplegar información del controller *ActorController.php* el cual le pasa los mismos retornando un array asociativo (según la jerga del lenguaje PHP). El controller *ActorController.php* se encuentra en *transacciones/module/Solicitud/src/Solicitud/Controller* y será examinado en el contexto del módulo Solicitud más adelante.

3.2.4 El directorio src

Contiene directorios y carpetas correspondientes a los controllers y models del sistema. En este directorio se incluye todo el código fuente que se ocupe de implementar la lógica del sistema. Como este directorio es muy importante, se examinará en detalle con un caso representativo correspondiente al módulo Solicitud en el siguiente capítulo.

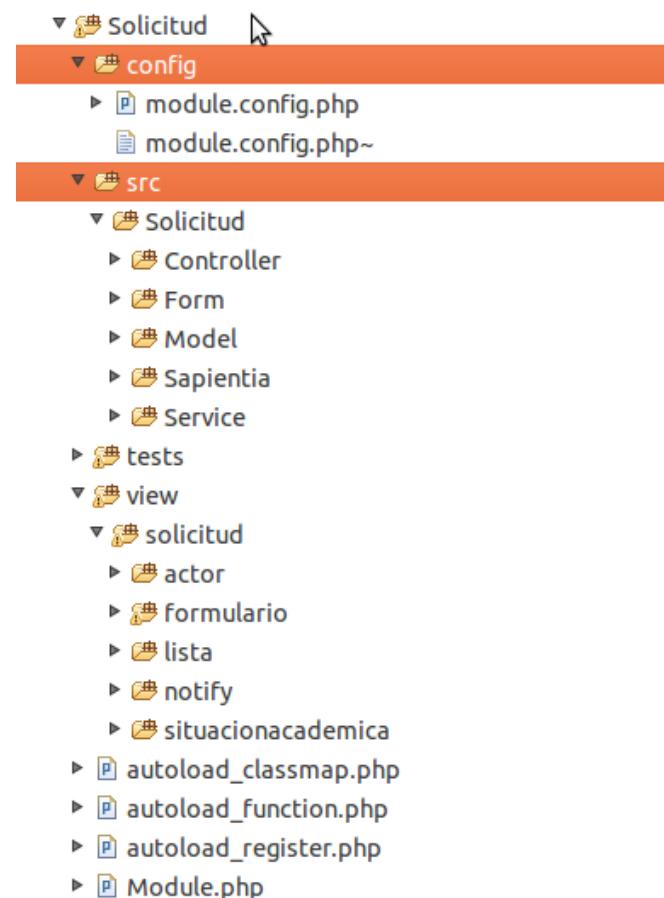
El archivo module.config

El directorio src

- El directorio Controller
- El directorio Form
- El directorio Model
- El directorio Sapientia
- El directorio Service

4. El módulo Solicitud

Como se mencionó en el capítulo anterior, cada módulo contiene mínimamente un archivo **Module.php** que se encarga de precargar el módulo y sus configuraciones. También contiene un directorio **config** el cual contiene la configuración del mismo. La definición del archivo **module.config** que se encuentra en el directorio config del módulo Solicitud es un ejemplo de configuración representativo.



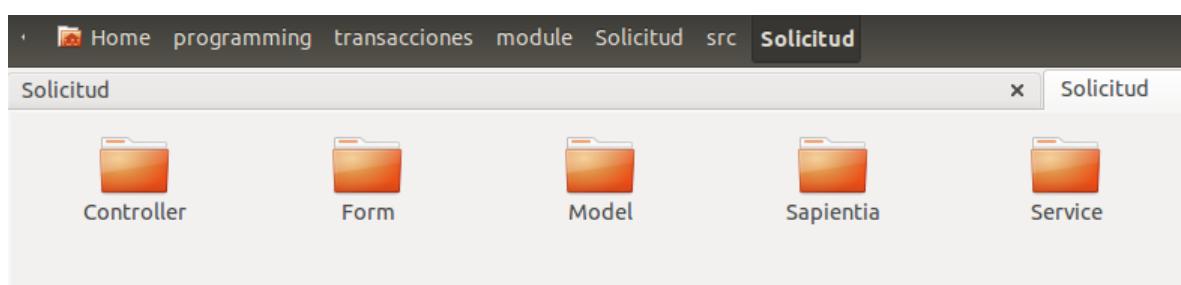
4.1 El archivo module.config

Está localizado en el directorio *config*. Contiene la configuración del módulo e incluye la definición de lo siguiente.

- **Controllers:** Especifica los controllers habilitados para el módulo.
- **Router:** Especifica las rutas de acceso para las distintas páginas que provee el módulo.
- **Service Manager:** Es un registro que permite describir los servicios que exporta el módulo así como los servicios que puede importar el módulo de manera que pueda invocarlos y utilizarlos. Es como un registro inteligente que permite otorgar como parámetro una cadena y recibir una instancia de una clase que fue definida en la aplicación para poder utilizar un servicio.
- **translator:** Servicio que permite traducir los mensajes de las páginas al español.
- **table-gateway:** Servicio que permite al instanciar una clase, obtener un objeto que representa una tabla de la base de datos, de manera que al manipular este objeto, se realicen operaciones CRUD de base de datos.
- **navigation:** Permite definir los menús que permiten navegar la página.

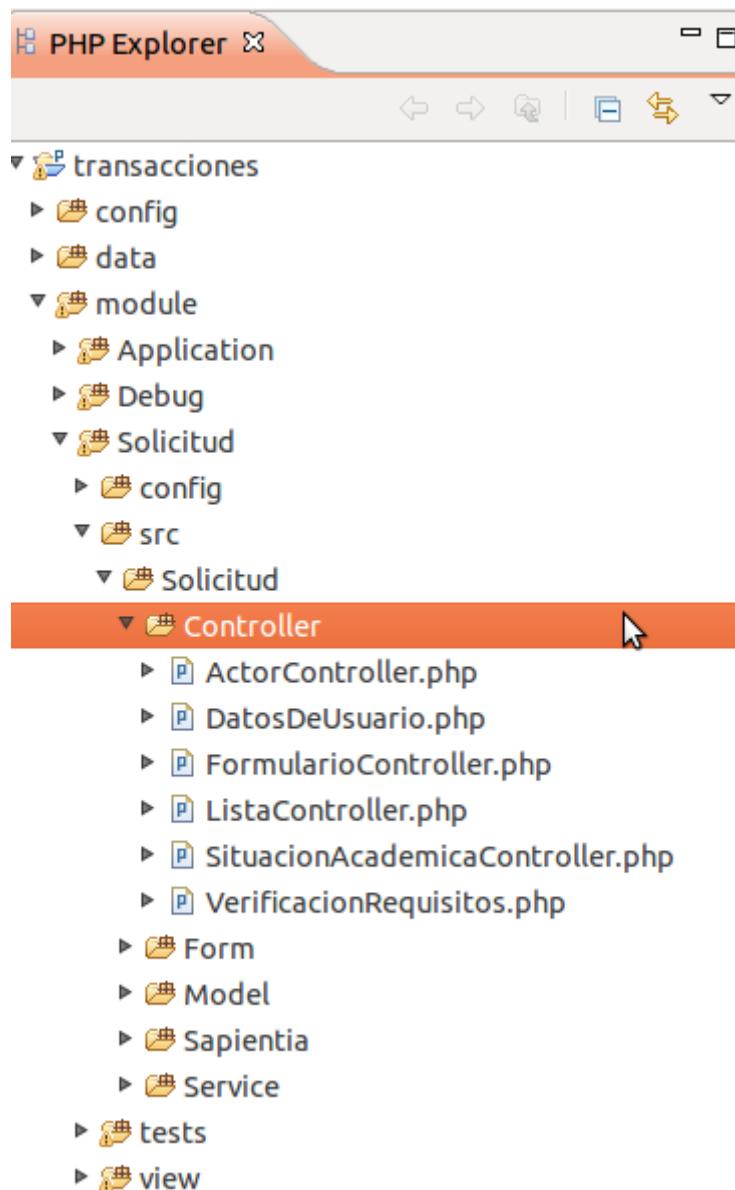
4.2 El directorio src

En el capítulo anterior se explicó el propósito general del directorio src. Ahora veremos el caso particular correspondiente al módulo Solicitud.



El mismo contiene los directorios Controller, Form, Model, Sapientia, Service.

4.2.1 El directorio Controller



Contiene los diversos Controllers del módulo Solicitud. Un Controller de la aplicación es el lugar desde el cual pasamos información como variables a las vistas para que las mismas las muestren y presenten. Un Controller agrupa un conjunto de Actions relacionados, los cuales se corresponden con páginas web, es decir, toda página del sitio web tiene un Action que contiene la lógica de la misma y un Action siempre está incluido en un Controller determinado.

El controller se encarga de recibir los datos ingresados por el usuario y de ejecutar toda la lógica en conjunto con el Model, realizar cálculos, y pasar la información procesada al View para que la despliegue al usuario. Básicamente la lógica de presentación corresponde al View, la lógica de los cálculos al Model y la lógica del flujo de la aplicación en el Controller.

Este directorio contiene a los siguientes archivos:

ActorController

Contiene la lógica correspondiente a las páginas de los diversos actores. Incluye la lógica correspondiente a las páginas de los diversos actores como los funcionarios (recepción, secretarías, directores y decano) y alumno.

FormularioController

Contiene la lógica correspondiente a las páginas de los formularios de las solicitudes a ser llenados por los usuarios. Este Controller agrupa los Actions que corresponden a cada una de las 22 solicitudes implementadas.

ListaController

Contiene la lógica correspondiente a las páginas que despliegan listas de solicitudes. Este Controller agrupa cada unas de las páginas que se encargan de listar todas las solicitudes o las solicitudes según su estado (nuevas, pendientes, aprobadas, rechazadas, anuladas). La lista de solicitudes a mostrar a un actor se determina según el rol del usuario de la sesión actual.

SituacionAcademicaController

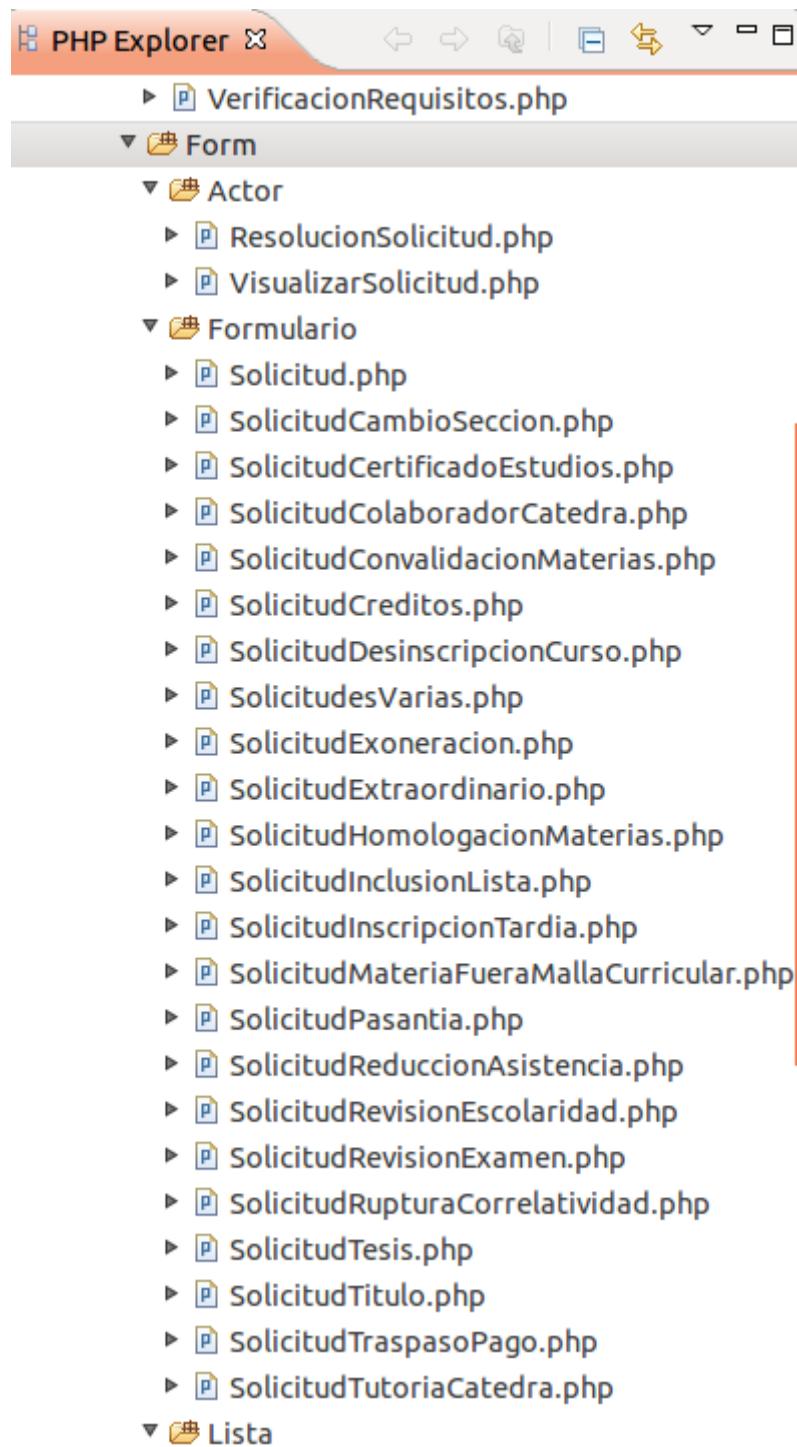
Contiene la lógica correspondiente a las páginas que despliegan la situación académica de un alumno determinado. Permite a los funcionarios visualizar los datos académicos del alumno solicitante (calificaciones, asistencias, horarios, materias en las que se encuentra inscripto, lista de materias que aprobó, materias cursadas actualmente por el alumno), de manera a facilitar el análisis que permita tomar una resolución a la solicitud.

Otros archivos

En este directorio también se incluyen dos archivos con funciones utilizadas ampliamente por los controllers:

- **DatosDeUsuario:** permite obtener los datos personales de un usuario.
- **VerificacionRequisitos:** contiene un conjunto de funciones que realizan los cálculos necesarios para comprobar si el usuario solicitante cumple o no los diferentes requisitos requeridos por una solicitud determinada para que la misma pueda ser aprobada.

4.2.2 El directorio Form



Contiene las clases que definen los formularios que serán instanciadas por los Controllers. Estos archivos definen clases que heredan de la clase Form y sirve para crear formularios HTML, los cuales son renderizados en el View al utilizar un View Helper propio de Zend Framework 2. Esta es la forma común de crear formularios en este framework.

Esta carpeta tiene los subdirectorios:

Actor

Contiene los formularios de ResolucionSolicitud que es utilizado en la vista del alumno de la resolución tomada a su solicitud hecha y de VisualizarSolicitud el cual es utilizado en la vista de la solicitud por parte de los funcionarios que les permite examinar los datos de la solicitud hecha por alumnos de manera a tomar una resolución a la misma.

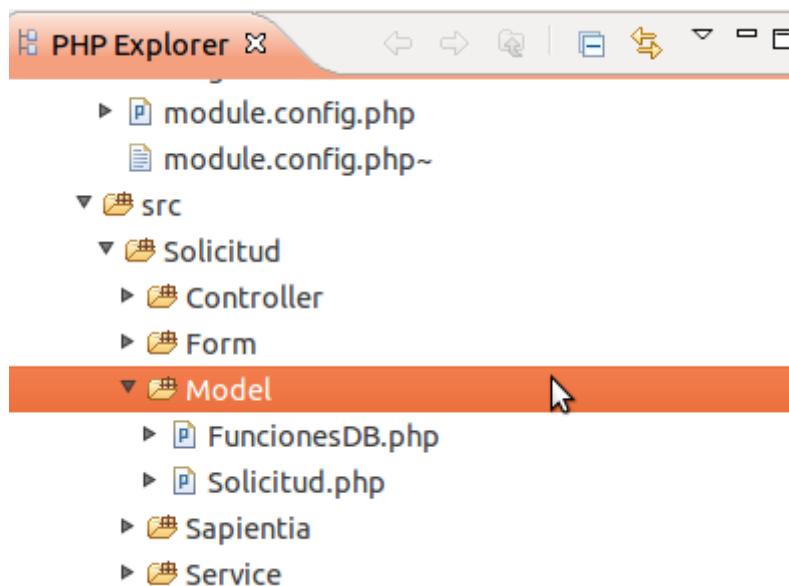
Formulario

Contiene los formularios que representan la versión digital de los formularios de las solicitudes. Estos formularios son las solicitudes que el alumno completa y envía con el objetivo de que sean aprobadas.

Lista

Contiene el formulario que se incluye al mostrar la lista de solicitudes que permite filtrar la lista empleando términos de búsqueda.

4.2.3 El directorio Model



Contiene los Models, los cuales están pensados para contener cualquier tipo de lógica, datos de aplicación y funciones. Proveen datos y funcionalidad que están agrupados en conjunto para un propósito común.

Este directorio es utilizado para que el programador implemente cualquier lógica que desea incluir o datos que desea proveer. Incluye los archivos:

- **FuncionesDB:** Que provee funciones de operaciones CRUD de la base de datos local. Provee una interfaz entre el sistema y la BD.
- **Solicitud:** contiene una clase que al ser instanciada resulta en un objeto que se utiliza en representación de la tabla de Solicitudes para realizar operaciones CRUD sobre la misma.

4.2.4 El directorio Sapientia

Provee una interfaz del sistema con los servicios de Sapientia. Incluye el archivo Sapientia-Client el cual contiene un conjunto de funciones que implementan la interfaz con el sistema de Sapientia. Estas funciones instancian clientes a los servicios ofrecidos por Sapientia.

4.2.5 El directorio Service

Provee servicios de base de datos como la clase que crea el adaptador a la base de datos local (base de datos del Sistema de Transacciones Académicas).

La creación de estos servicios se realiza según la especificación de Zend Framework.