Implementación de Aplicación Web usando el Framework de Desarrollo Yii: MirafloresPark

Felix E. Huaroto Pachas
Nhalim Y. Iparraguirre Paredes
Jonathan E. Montalvan Serafin
Escuela Profesional de Ciencia de la Computacion
Universidad Nacional de Ingeniería
Lima-Peru

20 de diciembre de 2014

Resumen

En este artículo se trata de describir los fundamentos del desarrollo de aplicaciones web, haciendo uso de alguna herramienta o framework de desarrollo. Para el caso particular de este artículo, se implementará un sistema de control y cobro electrónico de un estacionamiento. Para la implementación de este sistema se hará uso del framework de desarrollo web Yii, el cual utiliza el patrón de arquitectura de software Modelo Vista Controlador(MVC de aquí en adelante). El objetivo del presente artículo es mostrar los pasos a seguir para utilizar correctamente el framework, y lograr una aplicación que se beneficie de la robustez del patrón MVC, y de las ventajas que ofrece el framework, obteniéndose así una aplicación web ordenada,legible,escalable y funcional.

1. Introducción

Actualmente el crecimiento en el desarrollo de aplicaciones web ha aumentado enormemente, no existe empresa en la actualidad que no cuente con una página web, cualquier empresa con aspiraciones de surgimiento, necesita tener una página web. Algunas estadísticas hasta diciembre del 2012 sobre internet son las siguientes:

- **634 millones**, de páginas web registradas.
- **51 millones**, de nuevos sitios web registrados.
- **43 %** del millón de sitios web más visitados están hosteados en EE.UU.
- **48 %** de los 100 mil blogs más visitados corren bajo WordPress.

- **75 %** de los 10 mil sitios web principales usan software de código abierto.
- **100 millones** fue la cantidad de nombres de dominio ".com" para el final de 2012.
- **14,1 millones** fue la cantidad de nombres de dominio ".net" para el final de 2012.
- **9.7 millones** fue la cantidad de nombres de dominio ".org" para el final de 2012.

Todo esto, sin contar el número de dispositivos móviles con acceso a internet con las que contamos. Estas estadísticas nos demuestran que el mercado de producción de aplicaciones web, sigue en constante aumento. Además los requerimientos de las aplicaciones web, son cada vez mayores; necesitamos que la página web esté activa 24 horas al día 7 días a la semana, necesitamos que la página web sea accesible desde cualquier dispositivo con acceso a internet.

Por último, las empresas requieren un tiempo de desarrollo extremadamente corto, que las aplicaciones web sean estandarizadas, de facil mantenimiento(hay que recordar que la aplicación tiene que estar activa en todo momento.), es por este motivo que los frameworks de desarrollo web, están en auge, ya que nos permiten reducir los tiempos de desarrollo y la estandarización de las aplicaciones web y el código, además de la modularización, y las librerías base que nos permiten un fácil mantenimiento de la aplicación.

La primera pregunta que uno se debe hacer al empezar siquiera a pensar en el diseño de una solución web a algún problema específico es: ¿Debería usar algún framework para mi proyecto web?. Sin embargo, esta simple pregunta nos lleva incluso a otra más básica aún: ¿Que es un framework?, y ésta aunque pudiera parecer una pregunta simple, no es tan sencilla de responder; no obstante cualquier persona con mínimos conocimientos técnicos y/o experiencia en el sector tecnológico sabe intuitivamente que es un framework.

2. Frameworks

Entonces, ¿Que es un framework?. Un framework por lo general se define como un conjunto de módulos que permiten, o tienen por objetivo, el desarrollo ágil de aplicaciones mediante la aportación de líbrerias, componentes y/o funcionalidades ya creadas para que nosotros las usemos directamente.

El objetivo de los frameworks es hacer que nos centremos en el verdadero problema, y no tengamos que preocuparnos por implementar funcionalidades que son de uso común en la mayoría de las aplicaciones, tales como el proceso de logueo de los usuarios, o la conexión y recuperación de información de la base

de datos.

Una de las principales ventajas de usar un framework es que la mayoría de estos suelen basarse en algun patrón de arquitectura de software, normalmente el patrón elegido es **MVC** (Model-View-Controller). Sin embargo este patrón de diseño puede ser implementado sin usar ningún framework, la diferencia radica en que el framework te guía y te obliga al mismo tiempo a implementar este patrón, lo que hace que la aplicación sea más robusta.

Ahora que ya conocemos que es un framework, y que la mayoría de frameworks implementan el **MVC**, la pregunta que nos estamos haciendo es: ¿Debo usar un framework, solo porque utiliza el patrón MVC?. He aquí algunas ventajas y desventajas adicionales:

- Ventajas Adicionales
  - **Convenciones sobre configuración:** Los framework suelen tenr una serie de reglas de convención. Una de las convenciones más necesarias es sobre la estructuración de los directorios. Así, si conocemos estas reglas, cuando tengamos que hacer una modificación al código, sabremos rápidamente donde encontrarlo, ya que tenemos montada una jerarquía de directorios, mucho mejor que cualquiera que pudieramos haber creado manualmente. Sin embargo, muchos frameworks permiten y dan total libertad para obviar estas convenciones.
  - El **código de las librerías base** que aportan los frameworks, han sido testeado muchas veces, con muchos casos de prueba. Obviamente nosotros podemos escribir una librería desde cero, pero no hay necesidad de reinventar la rueda, en tareas triviales y comunes a todos los proyectos como la conexión a base de datos. Este es un principio de ingeniería de software, y es una de las ventajas de usar patrones de desarrollo.
  - **La comunidad de Usuarios**. Todos los frameworks disponibles hoy, poseen una comunidad significativa de usuarios. Normalmente estas comunidades son de gran ayuda, ya que los usuarios comparten información, e incluso módulos o extensiones para el framework, desarrollado por ellos mismos, y el usuario solo acopla la extensión.
  - **Trabajo en Equipo**. Gracias a la distribución de los directorios y módulos, el trabajo en equipo se hace mucho más sencillo. Además es mucho más fácil incluir a más personal, basta que este conozca el framework, para que sepa donde está todo, y el acoplamiento es realmente rápido. Obviamente esta velocidad de acoplamiento no se puede lograr con un código que ha sido hecho enteramente por ti, sin usar ningún framework.
- Desventajas de los Frameworks

- **Curva de Aprendizaje**. La principal desventaja de los frameworks es la curva de aprendizaje. La primera aplicación que un programador desarrolle con un framework específico, seguramente le tomará más tiempo que desarrollarla sin un framework. Pero a mediano o largo plazo, el programador habrá ganado tiempo, ya que el framework una vez conociéndolo, te permitirá un mantenimiento mucho más agil de la aplicación, y te permitirá desarrollar otras aplicaciones en menos tiempo cada vez, además de que las aplicaciones que desarrolles serán mucho más robustas que si las desarrollarás sin ningún framework.

3. Modelo Vista Controlador (MVC):Uno de los patrones de diseño más populares

Este patrón de diseño nos indica que nuestra aplicación debe tener al menos las tres capas siguientes: *Modelo*, *Vista* y *Controlador*.

¿Qué son éstas capas? ¿Para que sirven?. Pues bien, La capa *Modelo* es la que se encarga de trabajar con los datos, por ejemplo mediante una base de datos, tanto con respecto a las consultas como con respecto a las actualizaciones, comúnmente se dice que aquí se implementan la **lógica de negocio**. Desde el modelo se envían a la vista la información que le es solicitada a cada momento para que sea mostrada. Las peticiones de acceso o manipulación de información llegan al modelo provenientes del controlador.

La capa *Controlador*, Responde a eventos y envía peticiones al *modelo*, el controlador actúa como una suerte de middleware entre el modelo y la vista, ya que recibe los eventos de la vista(acciones del usuario), y las envía al modelo, para que este haga lo referente al manejo de la información, y si es necesario envía los resultados a la vista para que los cambios efectuados sean mostrados mediante las vistas.

Por su parte la capa de Presentación o *Vista* se encarga de la presentación de los datos al usuario final, al cliente de nuestra aplicación en un formato adecuado para interactuar con el usuario

Este patrón se basa en los conceptos de reutilización de código y la separación de conceptos, para poder facilitar así el desarrollo y el posterior mantenimiento de la aplicación.

4. Yii Framework

Yii es uno de los frameworks que viene creciendo más ultimamente, utiliza como la mayoría de frameworks para desarrollo web, el patrón MVC y su nombre

viene del acrónimo: Yes It Is("Sí, lo es" en español), que según los desarrolladores de Yii es la respuesta a todas las preguntas sobre yii.

Yii además de implementar el patrón de arquitectura de software MVC, implementa algunos otros patrones y componentes, tales como Active Record. La idea es usar el concepto de reutilización del código, y no tener que estar escribiendo el mismo código cada vez que se necesite hacer una tarea trivial y archiconocida. En este aspecto los componentes y los patrones de diseño se usan mucho con este fin.

4.1. Yii Basado en Componentes

Desde la perspectiva de los componentes, su diseño debe incorporarse tratando de que puedan ser reutilizados globalmente, dentro de un mercado global de software. Esto supone seguir el concepto: "maximizando el re-uso se minimiza el uso", sin embargo es difícil diseñar un componente sin tener en cuenta el uso que se le va a dar en el futuro.

Sin embargo, el desarrollo de componentes no puede realizarse de forma arbitraria, sino de acuerdo a los modelos existentes, incorporando aquellos mecanismos que permitan a dichos componentes interoperar con otros. En este sentido el desarrollo de dichos componentes se ve altamente beneficiado por la existencia de dichos modelos. A un primer nivel, la interoperabilidad de componentes(casi independiente del modelo de componentes sobre el que se apoyen) podemos decir que está actualmente superada.

Esto quiere decir que el cliente no necesita saber sobre los funcionamientos internos del componente (su implementación) para hacer uso de ella. El cliente solo necesita saber, que hace ese componente, y además tener la seguridad de que funcionará con solo acoplarlo al framework de desarrollo.

Los componentes son el elemento clave en el framework Yii. Todo comienza con la clase CComponent de componente de Yii. La mayoría de las clases utilizadas en Yii son descendientes de la clase CComponent de componente base. Por ejemplo, el objeto de una aplicación web será de tipo CWebApplication. Esa clase se deriva de CComponent de componente (aunque hay otras clases en el medio). Los controladores son de tipo CController, que hereda de CBaseController, que hereda de CComponent de componente. CActiveRecord hereda de CModel, que hereda de CCable de componente, y la misma ruta de herencia se aplica a CFormModel.

4.2. Orientación

Para poder trabajar con un framework, se necesita cierto conocimiento sobre éste, de otra forma no se podrá sacar provecho del framework y es posible que el tiempo de desarrollo sea mayor en vez de acortarse. Una ventaja importante es conocer la estructura de directorios del framework, si la aplicación ha sido desarrollada bajo las convenciones del framework, cualquiera que busque alguna parte del código para darle mantenimiento o lo que sea, lo encontrará mucho

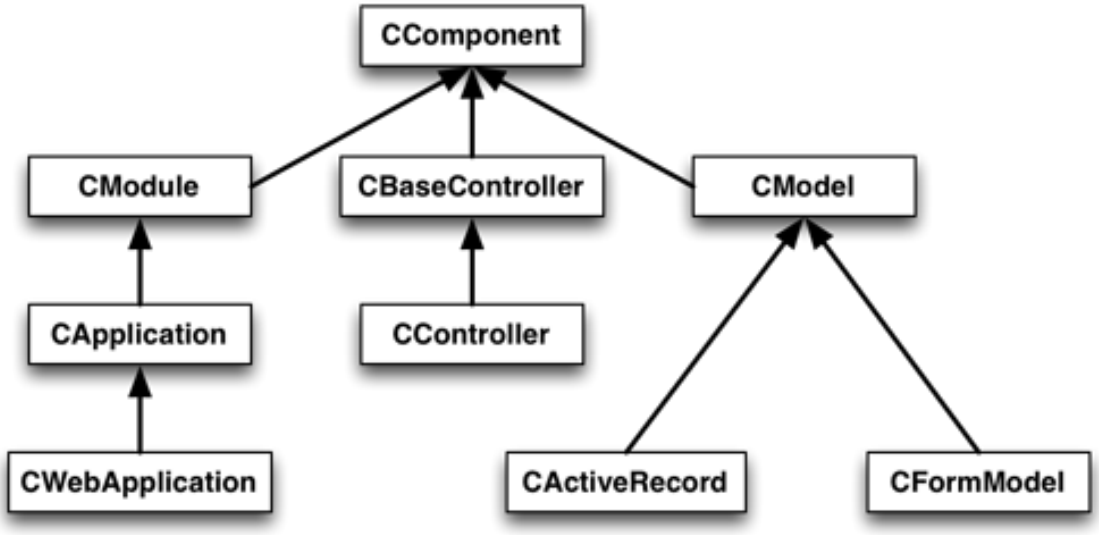

Figura 1: TArbol jerárquico de componentes de Yii

más rápido. De la misma forma, sabrá donde se encuentra cada parte de la aplicación. A continuación se muestra la estructura básica de directorios de Yii.

4.3. EStructura de directorios

```
Proyecto
index.php
/css
/protected
|_ home.xhtml
|_ Components
|_ /config
   |_ main.php
   |_ column1.php
   |_ ...
|_ /controllers
   |_ SiteController.php
   |_ ...
|_ /data
   |_ db.sql
|_ /models
|_ /views
   |_ layouts
```

4.4. Algunos patrones de diseño utilizados

• Table Data Gateway:
Sea un objeto que actúa como un Gateway a una tabla de base de datos. Una instancia de ella actuará podrá controlar todos las filas de dicha tabla

Por ejemplo para el Caso de una Orden de compra

```
<?php
$tabla = new OrdenGateway(new Conexion('...'));

$tabla->insert('XX123456789',358.80,'no pagada');

$tabla->update(42, 'XX123456789', 358.80, 'pagada');

$tabla->delete(42);
```

```
<?php
$tabla = new OrdenGateway(new Conexion('...'));

$tabla->insert('XX123456789',358.80,'no pagada');

$tabla->update(42, 'XX123456789', 358.80, 'pagada');

$tabla->delete(42);
```

• Active Record:
Un objeto que contiene la una fila de una tabla o vista de base de datos, encapsula el acceso a la base de datos y agrega la lógica de negocios a esta.

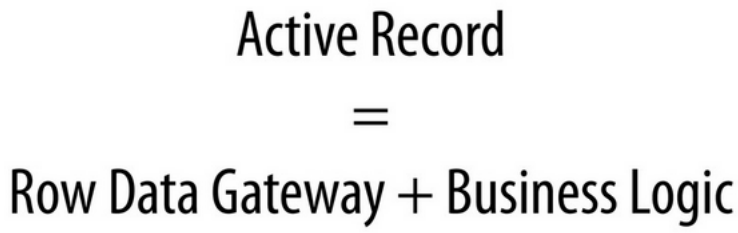

Figura 2:

5. MirafloresPark

Miraflorespark, está pensado como un aplicativo web, que permita controlar el sistema de parqueo de automóviles de un determinado estacionamiento. Como toda aplicación a desarrollar tiene requerimientos funcionales, los cuales se ven reflejados como las reglas de negocio, algunas de las cuales veremos a continuación.

5.1. Reglas de Negocio

- La playa de estacionamiento tiene un número limitado de lotes de estacionamiento.
- Cada lote de estacionamiento tiene un identificador y un sensor, que permite conocer si el estacionamiento está libre o desocupado.
- Existen dos tipos de lotes de estacionamiento, el primer tipo son los lotes contratados mensualmente, el otro tipo son los lotes que están disponibles para alquilarlos por horas.
- Los usuarios que tienen contratos mensuales, no necesitan tarjeta RFID, ya que el pago es mensual y su lote no se alquila a usuarios temporales. Sin embargo tienen un código especial para que su identificación sea más rápida. El usuario con contrato mensual puede optar por la tarjeta RFID, si desea que su salida sea más rapida.
- Los usuarios que requieran solo servicios temporales, podrán tener o no tarjeta RFID. Si la tienen el cobro se hace a través de esta y el usuario podrá "recargar la tarjeta en cualquier estacionamiento autorizado." Caso contrario el portero anota la placa del carro, y el sistema le indica que estacionamiento debe utilizar, generando un ticket con un código de barras, que guarde la información sobre el lote asignado y la hora de entrada.
- Al momento de salir, tenemos algunos casos: si el carro tiene un contrato mensual y cuenta con tarjeta RFID, simplemente espera a que el lector RFID constate su salida y levante la barrera. Si no tiene RFID brinda su código de usuario al portero, este lo introduce en el sistema, verifica que es un usuario con contrato mensual y levanta la barrera. Si el usuario es un usuario temporal y tiene una tarjeta RFID, espera que el lector RFID constate la hora de su salida, calcule el tiempo utilizado y facture el monto en la tarjeta RFID, y luego levanta la barrera. Por ultimo si el usuario es temporal y no tiene una tarjeta RFID, deberá proporcionar el ticket que le fue entregado al ingresar, para que el portero escanee el ticket, liquide, cobre y emita un comprobante y finalmente emita un comprobante.
- Para abrir un contrato mensual, el usuario(dueño del auto o compañia) deben dirigirse a un centro de atención del cliente(administrador), registrar los datos básicos del auto y suscribir el contrato. Los contratos mensuales

se pueden suscirbir en cualquier momento, pero estos comienzan solo los días 1 de cada mes.

- Las tarifas para los contratos mensuales y los usuarios temporales son definidas por el administrador, y están sujetos a cambios.
- El sistema debe guardar todos los servicios de parqueo de los últimos meses: Placa del carro, horas de entrada y salida, monto pagado.(usuarios con contrato mensual y usuarios temporales)
- El administrador debe poder obtener reportes por contrato, por empresa, o por persona natural. Si los reportes son pedidos por empresa, se debería poder diferenciar entre los diferentes autos que la empresa pudiera haber registrado.

Especificación del caso de Uso principal:Pagar el estacionamiento.

Caso de Uso	Pagar el estacionamiento
Actores	Usuario(Automóvil)
Tipo	Básico
Propósito	Permitir a un usuario, cancelar el monto producido por las horas que usó el estacionamiento.
Resumen	El caso de Uso es iniciado por el usuario(automóvil). Ofrece la funcionalidad de cancelar el monto establecido por el tiempo de uso del estacionamiento. El monto puede ser debitado de su tarjeta de débito o crédito o puede ser en efectivo.
PreCondiciones	Previamente el usuario debería haber ejecutado el caso de uso **activar sensor de entrada** y **registrar hora de entrada**.
Flujo Principal	El flujo principal inicia cuando el usuario, deja el estacionamiento que estaba utilizando, y el sensor ultrasónico notifica al portero. Pasado un tiempo el usuario debe llegar a la garita de salida y antes de ejecutar el caso de uso **activar sensor de salida**, se pueden dar los siguientes subflujos
SubFlujos	• Si el usuario cuenta con una tarjeta RFID ésta debió registrar la hora de entrada al momento que el usuario ingresa al estacionamiento. • Si el usuario no cuenta con una tarjeta RFID el portero debió registrar la información del usuario y la hora de entrada al momento que el usuario ingresó al estacionamiento. • En cualquiera de los dos casos el sistema tiene registrada la información de ingreso del usuario. De la misma forma el sistema obtiene la hora de salida(mediante el RFID, o a pedido manual del portero), y calcula el monto a cobrar de acuerdo a las tarifas establecidas por el estacionamiento. • Para efectuar el pago el lector de tarjetas deduce el costo calculado por el sistema de la tarjeta de débito o crédito del usuario asociado al automóvil. • En caso el usuario no cuente con saldo suficiente, el lector notifica al sistema para que no autorice la salida. Se notifica al portero y se notifica al usuario, para que pueda efectuar su pago en efectivo. • El usuario puede elegir pagar la tarifa en efectivo, en caso no cuente con tarjeta RFID.
Excepciones	E-1 Usuario inválido E-2 Información Incompleta. E-3 Monto insuficiente. E-4 Usuario sin tarjeta RFID.

Figura 3: Especificación de caso de uso principal

6. Metodología de Desarrollo

Para el desarrollo de la aplicación se escogió la metodología de desarrollo agil, ya que no existían requisitos estables en cuánto a las vistas de la aplica-

ción. La ventaja de este proceso adaptativo es que genera rápidamente prototipos de la aplicación que pueden ir siendo revisadas con el cliente, en nuestro caso particular, los clientes somos los mismos **stakeholders**, los programadores.

Se dividió el trabajo y se ponían a disposición del grupo los avances en cada uno de los módulos que se estaban trabajando, de tal forma que se pueda obtener retroalimentación de cada uno de los stakeholders, esto asegura que el producto final sea tal cual cada una de las partes esperaban y no hayan sorpresas de ultima hora. Adicionalmente como el nombre lo menciona, esto permite la fácil adaptación a nuevos requerimientos o la modificación de estos.

7. Conclusiones

- Podemos conluir que trabajar con un framework trae ventajas de acuerdo a los requerimientos del proyecto, en nuestro caso el framework de desarrollo Yii nos facilitó en buena medida el trabajo ya que traía implementado modelos y patrones que se ajustaban al proyecto.
- Usar un framework te permite y te anima a trabajar en equipo pues la estructura ordenada del framework permitía que todos los integrantes del equipo llevarán el control de los modulos que estaban desarrollando y éstos módulos eran facilmente integrables.
- EL uso de un framework reduce el tiempo de desarrollo, sin embargo es necesario cierto nivel de conocimiento del framework, la forma en la que opera y la jerarquía de directorios y convenciones usadas.
- Si bien es cierto que existen muchas herramientas y frameworks de desarrollo, cada uno tiene sus ventajas y desventajas, es cuestión de elegir con cuál se siente más comodo uno como desarrollador, al fin y al cabo debemos tener en cuenta, que ningún framework está desarrollado para una aplicación específica, si no que son más genéricos, lo que puede generar un problema de adaptación a nuestras necesidades específicas.
- Yii es un framework que se basa en componentes y extensiones, los cuáles pueden ser agregados o suprimidos de la versión base, de acuerdo a las necesidades de cada aplicación. Esta es una gran ventaja, ya que somos libres de adicionar funcionalidades que no vienen predefinidas a nuestras aplicaciones.

(**?**)

Referencias

[1] https://speakerdeck.com/hhamon/database-design-patterns-with-php-53

[2] http://martinfowler.com/

[3] https://speakerdeck.com/hhamon/database-design-patterns-with-php-53

[4] http://www.larryullman.com/2013/03/13/watching-for-model-events-in-the-yii-framework/

[5] http://www.larryullman.com/tag/yii/