Universidad del Azuay

FACULTAD DE CIENCIAS DE LA ADMINISTRACIÓN

ESCUELA DE INGENIERÍA DE SISTEMAS

ESTUDIO, ANÁLISIS Y DESARROLLO DE UNA APLICACIÓN WEB HÍBRIDA.

TRABAJO DE GRADUACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS

AUTORES: ANDRÉS CRIOLLO E.

DIEGO VINTIMILLA E.

DIRECTOR ING. RUBÉN ORTEGA LÓPEZ

CUENCA, ECUADOR

2016

Dedicatoria.

Dedico esta tesis a mis padres, que han sido mi apoyo en momentos difíciles, quienes gracias a sus consejos me han permitido cumplir mis metas, a mi esposa que ha sido mi inspiración y fortaleza, a Dios que bendice y guía mi camino.

Diego Ricardo Vintimilla Espinoza

Dedico este trabajo a mis padres Ricardo y Ligia que son los pilares principales de mi vida, que, con su apoyo de manera incondicional, confianza y sobre todo sabiduría han guiado mi camino de la mejor manera, ayudando a mantenerme firme en cada paso que doy. A mis hermanos, Sandra, Fernando, Cristian, Ronald, que han sido mi ejemplo y luz en las situaciones más adversas, convirtiéndose en mis amigos y consejeros. A la persona que me alentó a culminar con esta faceta de mi vida la Ingeniera Magali Chicaiza, que me acompaño y dio la fuerza para que no me diera por vencido cuando pensé hacerlo.

Andrés Patricio Criollo Espinoza

Agradecimientos

Agradecemos a Dios por todas sus bendiciones y darnos el conocimiento necesario para culminar esta tesis y alcanzar nuestras metas.

A la Universidad del Azuay y a cada uno de sus profesores que ciclo a ciclo nos brindaron los conocimientos necesarios para formarnos en profesionales en función de la sociedad.

Al Ing. Rubén Ortega quien ha sido nuestro guía, gracias a su ayuda, paciencia y experiencia las cuales han sido fundamentales para la culminación de este proyecto.

Al Ing. Francisco Salgado, quien ayudo a enfocar de manera clara los pasos y metodologías necesarias para la elaboración de este trabajo.

Contenido

Índice de imágenes

Índice de tablas

Resumen

Los desarrolladores de aplicaciones web en la actualidad, pueden simplificar su trabajo utilizando contenidos provenientes de componentes de software, conocidos como API's, los mismo que son desarrollados por terceros y son accesibles mediante internet de una manera libre o con algún aporte económico. La API's tienen como propósito, empaquetar funcionalidades y tareas independientes, que pueden ser integradas a otras aplicaciones sin tener que crearlas y probarlas. El presente trabajo pretende demostrar la utilidad funcional de las API's integrándolas en una aplicación, relacionada con apuestas dentro de la disciplina del futbol, que muestra la integración e interoperabilidad de varias interfaces de programación de aplicaciones.

ABSTRACT

Nowadays web applications developers can simplify their work using content from software components known as API's, which are developed by third parties and are accessible via Internet free of charge or by a financial contribution. The purpose of the API's is to package features and independent tasks, which can be integrated into other applications without having to create and test them. This paper aims to demonstrate the functional purpose of API's by integrating them in an application related to betting within soccer games, which shows the integration and interoperability of multiple Application Programming Interfaces.

UNIVERSIDAD DEL AZUAY
Dpto. Idiomas

Translated by,
Lic. Lourdes Crespo

Introducción

Desde la aparición de la Web, incontables usos se han ido integrando y usando de manera muy significativa, teniendo constante evolución en ámbitos, geográficos, estudiantiles, laborales y sociales. La web se va convirtiendo cada vez con mayor dominación en un espacio social, convirtiendo a esta era en una sociedad de conocimiento donde compartir información y comunicar conocimiento es muy habitual.

La Web 2.0 va evolucionando constantemente, y oportunamente se crean aplicaciones más poderosas donde la sencillez y eficacia de dicha aplicación es de suma importancia para el usuario final. Una de las metas de ingeniería de software dice que debemos tener muy en cuenta el principio de reusabilidad, con el objeto de conseguir objetivos más rápidamente, utilizando programación, plantillas, frameworks, servicios web, donde cada una de ellas ha nacido con el deseo de usar material ya elaborado en vez de construirlo.

Para llegar a esta nueva etapa evolutiva, debemos centrarnos en la llamada web semántica donde las ontologías para describir una nueva plataforma de cambio de información y conocimiento ha hecho surgir nuevas metodologías que tratan de cumplir los requerimientos actuales, una de estas tecnologías actuales se llama Aplicación Web híbrida (mashup).

Mashup se puede definir como una web híbrida donde se aloja varias aplicaciones elaboradas por terceros integrándolas con el fin de crear algo nuevo.

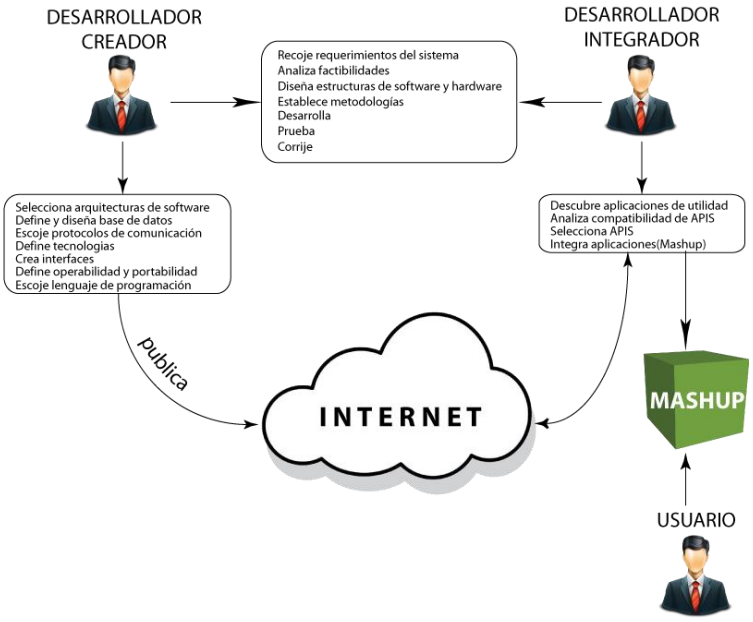


Fig. 1 Esquema de aplicación web híbrida

Fuente: Basado en (Florian Daniel, 2009)

Tipos de Aplicaciones Web Híbridas

Dependiendo del tipo de requerimientos o características las aplicaciones web híbridas se pueden clasificar en:

Consumidores: Combinan información de distintas aplicaciones desarrolladas por terceros y presentarlas en una interfaz gráfica sencilla.

Datos: Consumen información de varias Web feeds como Yahoo, Google y Twitter.

Empresariales: Combina información externa e interna a la organización dando valor agregado a los datos institucionales obteniendo como resultado una funcionalidad colaborativa y obteniendo una aplicación de negocio.

Capas de una aplicación web híbrida

Las capas que componen un mashup generalmente vienen compuestas de una estructura de 3 niveles, esta estructura puede ser extendida según necesidades de los usuarios, pero las citadas a continuación son la base para todo tipo de aplicación web híbrida:

Lógica de componentes. En la lógica de componentes se define el sitio en el cual van a estar alojados los datos, así como también el tipo de servidor que va a ser huésped de la información y como la información va a ser almacenada en el mismo: estructura de las tablas que componen la base de datos de la aplicación, los módulos de codificación. Los componentes de este nivel están alojados en el servidor web, al cual las demás capas tendrán acceso y podrán obtener lo necesario para sus procesos.

Componentes de datos. Middleware o capa intermedia cuya función es la de interactuar con el servidor recibiendo o enviando información entre, los datos alojados en el servidor y el usuario final. Estos datos tienen su propia ontología y la información detallada de cada uno (metadata) contenida dentro de sí, todos estos parámetros son fundamentales para la manipulación y el correcto intercambio de la información. Los formatos más utilizados actualmente para tratar los datos para la comunicación son: eXtensible Markup Language (XML), JavaScript Object Notation (JSON), Comma Separated Value.

Componentes de interfaz de usuario. Conjuntos de herramientas que ayudan a diseñar e implementar la parte frontal de la aplicación que sirve de medio para que el usuario interactúe con la aplicación. En esta interfaz se muestran los datos al usuario de forma que sean de fácil entendimiento, de una manera eficaz y en el momento que el usuario lo requiera, es decir que la información sea oportuna. En la actualidad se tiene a disposición herramientas web que nos permiten desarrollar el componente de interfaz, las cuales ayudan a los desarrolladores a trabajar sobre la parte visual de la aplicación, consiguiendo así, la conformidad del usuario al navegar y trabajar por y con el programa. Herramientas tales como *H*yper*T*ext *M*arkup *L*anguage (HTML) y *C*ascading

*S*tyle *Sheets* (CSS), brindan la facilidad a las personas encargadas de la creación de la interfaz, para elaborar diversas soluciones a los requerimientos del usuario.

1 Conceptos y definiciones

1.1 Aplicación web híbrida

El término mashup hoy en día es extensamente utilizado por las personas encargadas de generar productos software, ya sean estas aplicaciones móviles, web, empresariales, etc. Aunque no tiene aún una definición oficial que certifique lo que es, por ello se puede tener muchas interpretaciones distintas del concepto de mashup de lo que es y de lo que no. Pero todas estas interpretaciones tienen un punto en común, su concepto base, que ayuda a entender la definición de una aplicación web híbrida de mejor manera y esta es que: Una aplicación híbrida, selecciona aplicaciones web de las cuales puede obtener contenido y funcionalidades, con el objetivo de crear algo nuevo.

Las aplicaciones híbridas pueden ser formados por diversos contenidos: galerías de imágenes y videos, noticias, geolocalización de personas o cosas e incluso la de ayudar a determinar enfermedades en pacientes, tan solo con la obtención de datos como sus síntomas. Toda esta información debe ser combinada y presentada de forma que sea atractiva para el usuario además de eficaz.

(Bernal, 2009), afirma que la palabra "mashup" proviene de un término inglés vinculado a la música, que significa, crear una nueva canción a partir de la mezcla o unión de pedazos de canciones. De allí que el nuevo producto resultante posea cualidades propias de un mashup como son: el proceso de combinación, visualización del resultado, y agregación o reutilización del resultado en futuros proyectos.

Keith Chapman (2011) ejecutivo de la compañía WSO2 (Web Services Oxygen) define los mashups como: "aplicaciones compuestas que abstraen datos y/o funcionalidades de dos o más fuentes externas para combinarlas, obteniendo como resultado un nuevo servicio o un componente web". La mezcla de contenido que elaboran las aplicaciones híbridas, ayuda a mejorar la experiencia de los usuarios mediante una fácil y rápida

integración de APIs (Application Programming Interface) (Figura 2), feeds[1] y Web scraping[2], obteniendo resultados que tienen que ser superiores a los originales.

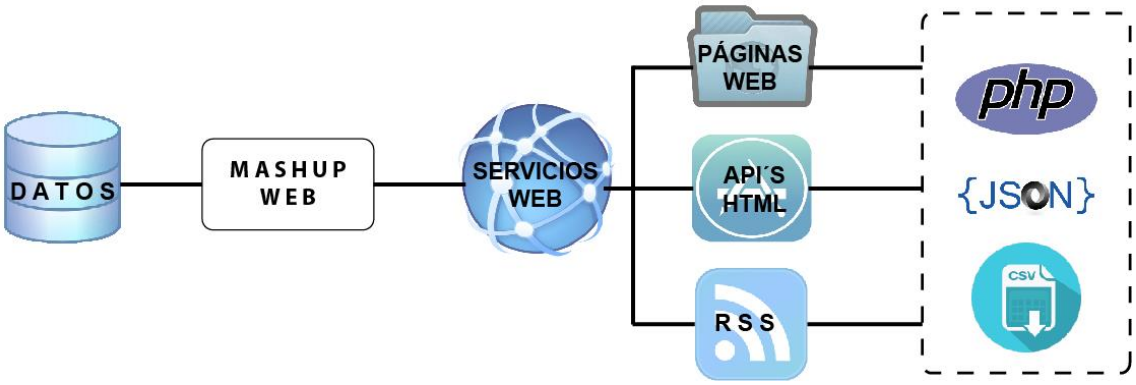


Fig. 2 Anatomía de los mashups

Basada en wso2.org

1.1.1 Tipos de Mashup

Por sus características, la clasificación de los mashups puede ser definida como: mashups consumidores, mashups de datos y mashups empresariales.

Los mashups consumidores, tienen la característica de ser lo más fáciles de entender y aplicar, por ende, son muy populares y conocidos. Su objetivo es combinar datos de fuentes internas o externas heterogéneas y presentarlas en una interfaz gráfica sencilla. (Young, 2009), engloba dentro de esta categoría a aquellas aplicaciones que incluyen las API de Google Maps y Google Earth.

Los mashups de datos, tienen como principal objetivo mezclar información similar, provenientes de distintas fuentes, mostrando el resultado en un nuevo entorno visual. Dentro de esta categoría se encuentran las aplicaciones que consumen datos provenientes de varios Web feeds como Google, Twitter, Yahoo, entre otros.

[1] Feeds: contenido sindicado
[2] Scraping: técnica para obtener información directamente de páginas Web

El mashup empresarial, combina la información que posee la organización, sea esta interna o externa, añadiendo un valor agregado a esa información, dependiendo del objetivo que se requiera cubrir. Esta combinación puede ser enfocada incluso a procesos o grupo de procesos, incrementando funcionalidades colaborativas y obteniendo como resultado una aplicación de negocio más eficaz y dirigido a lo que se pretende resolver. Inclusive pudiendo incluir otras aplicaciones híbridas que funcionen entre sí para lograr mayores y mejores resultados. Los mashups que suelen ser desarrollados dentro de otros se conocen como mashups monstruos.

La Tabla 1 muestra los API´s más usadas en portales web y aplicaciones móviles. De todas estas aplicaciones generadas muchas de ellas por algún funcionamiento favorable, son tomadas y se combinan con otras para formar un servicio mejorado y personalizado.

Apis Públicas	**Porcentaje utilización**
Google Maps	46%
Twitter	12%
Youtube	10%
Flickr	9%
Amazon	6%
Facebook	6%
Twilio	5%
LastFM	3%
EBay	3%
Google	2%

Tabla 1 Apis populares basado en http://www.programmableweb.com/apis/

Teniendo en cuenta que estas API´s pueden ser utilizadas por terceros, obtenemos el siguiente cuadro que expone en qué áreas son aplicadas y el porcentaje para cada área que se tiene actualmente.

Área	Porcentaje
Mapeado	27%
Mensajería	13%
Búsqueda	10%
Social	10%
Fotografía	7%
Compras	7%
Video	6%
Viajes	5%
Música	5%
Móviles	4%

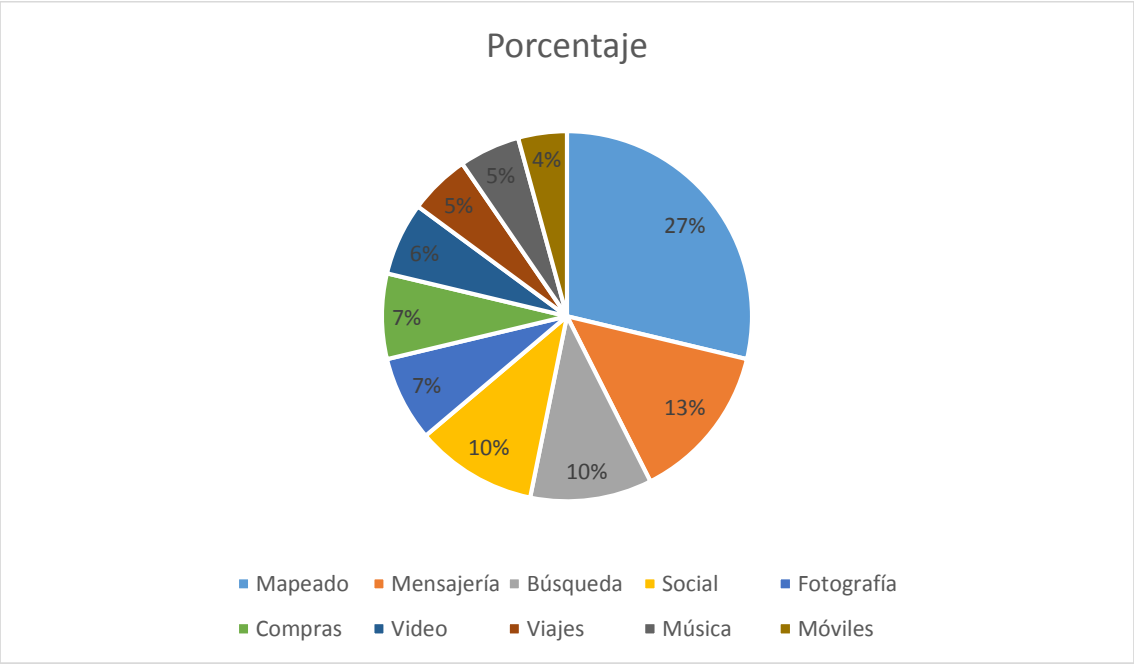


Tabla 2 Categorización de mashups web

Basado en http://www.programmableweb.com/apis/

El concepto de añadir información desde diversos sitios no es nuevo. Las empresas hoy en día, por el afán de aumentar la productividad de su personal, se ven en la necesidad de compartir sus aplicaciones y fuentes de información entre sí, permitiendo unir aplicaciones con información ya depurada en otra aplicación que solo la utilice como repositorio de datos. (Sarang, 2009), opina que las empresas para lograr objetivos de

manera conjunta utilizan portales empresariales, los que permiten acceder a múltiples aplicaciones que contienen información de distintos sistemas, los mismos que, pueden ser configurados y personalizados según los requerimientos de las empresas.

Incluir aplicaciones web híbridas dentro de la organización, permite a los usuarios el poder gestionar y personalizar la información que se desea visualizar, ayudando a cumplir con la satisfacción de requerimientos y necesidades de la empresa. En la actualidad, los proveedores de herramientas para el desarrollo de portales soportan la implementación de mashups (Sezov, 2011).

El continuo avance e implementación de soluciones web, ha permitido la creación de herramientas de programación confiables para desarrollas aplicaciones distribuidas. Tecnologías como Ajax, redes P2P, Google Maps, Google Earth, entre otras, ha incrementado exponencialmente la versatilidad del entorno Web, ayudando a que se consiga productos atractivos, dinámicos e interactivos que llaman mucho la atención del usuario. Estas herramientas han permitido que los desarrolladores sean capaces de crear fácilmente nuevos servicios basados en la integración de aplicaciones realizadas con la herramienta (reutilización) y en fuentes de datos situadas en la nube.
Las herramientas generalmente más utilizadas para el desarrollo de aplicaciones web híbridas son: Google Maps, Amazon Web Services, Del.icio.us, Flickr, Microsoft, Yahoo, Twitter, Youtube un gran número de ellos tienen recursos que aportan al diseño e implementación de los mashups.

1.1.2 Ventajas y desventajas de Aplicación Web Híbrida

Las aplicaciones web híbridas tienen varias ventajas, las cuales, facilitan al desarrollador la integración de funcionalidades de componentes pertenecientes a otros sistemas en sus productos software, pero tienen ciertas desventajas que vienen inmersas con las API´s que pretendemos utilizar.

1.1.2.1 Ventajas

Una de las ventajas de utilizar mashups es que, permite la reutilización de las aplicaciones existentes, en donde, en lugar de desarrollar componentes para un sitio web desde cero, se puede utilizar los disponibles en la nube, para implementar la característica deseada.

Para el desarrollo de un mashup, no se requiere que el usuario tenga habilidades de programación, sino simplemente, tiene que saber cómo implementar los componentes dentro de su propio sistema y acoplarlo según sus necesidades.

El costo de desarrollo de aplicaciones se reduce significativamente, pues se puede buscar y tener acceso a componentes preexistentes que son fáciles de reutilizar.

1.1.2.2 Desventajas

El usuario no tiene control sobre la calidad del componente que está integrando en su aplicación, es decir, el componente puede quedar obsoleto por falta de actualización de datos y la evolución del mismo a las tecnologías vigentes en ese momento.

El problema de escalabilidad, al no existir garantía de que el servicio a implementar, será capaz de gestionar el tráfico producido por el sitio web a medida que este incremente el número de peticiones, para la obtención de la información requerida.

La veracidad de los datos proporcionados por el servicio que se integra en la aplicación, puede producir grandes problemas de confianza por parte del usuario al usar el producto software.

1.1.3 Composición de un mashup

A pesar de que las investigaciones acerca de las webs híbridas aún sean escasos, ya han existido casos de estudio que profundizan en el desarrollo de la pila de un mashup;

todos ellos con diseños y nombres, pero todos tienen en común las mismas características básicas. En la conferencia internacional sobre ciencias de sistemas, dictada en Hawaii (Stefan Bitzer, 2009) proponen una arquitectura web híbrida compuesta por cinco capas que indican la jerarquía de procesos y recursos fundamentales para el desarrollo de mashups.

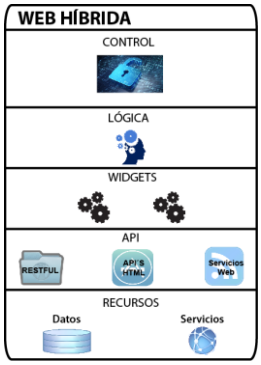


Fig. 3 Composición de una aplicación web híbrida

Basado en (Stefan Bitzer, 2009)

La capa más baja representa los recursos, los cuales son primordiales para los demás niveles de la jerarquía, los mismos que utilizarán y basarán su funcionamiento sobre esta primera capa. Aquí se encuentran los datos y los servicios web, base esencial de una web híbrida, estos recursos se pueden obtener de distintas fuentes ya sean internas o externas, por lo que los datos pueden tener varios formatos y ser expresados de distintas formas. Para la integración dentro de nuestra web híbrida los datos deben ser preparados para poder trabajar con ellos y mezclarlos entre sí, para obtener nueva y mejor información.

Se debe tener en cuenta que, en algunos casos particulares al integrar varias aplicaciones, podemos encontrarnos con sistemas heredados[3]. Una aplicación web híbrida es flexible y para su implementación se las tiene que aislar y trabajarlas como componentes sueltos, utilizando envoltorios (wrappers[4]) de contenido o visuales.

[3] Sistemas que se han desarrollado y con el tiempo dependen de tecnologías hardware y software obsoletas. (Somerville, 2005)

[4] Clases diseñadas para envolver los tipos primitivos en un objeto, consiguiendo con ello que puedan adquirir comportamientos y características que son solo reservados para los objetos.

El segundo nivel, es la capa de acceso, en ella se describen las características funcionales y no funcionales de los recursos, en esta capa existe un débil acoplamiento entre las aplicaciones que componen la web híbrida y es donde se definen los métodos para el intercambio de información. Bitzer define dos tipos de aplicaciones para la comucacion de los servicios web, ambas basadas en el protocolo HTTP, el que actua como un sobre que empaqueta los datos y los envía establaciendo la comunicación para el intercambio de informacion.

La tercera capa es donde se crean los widgets[5] que funcionan como módulos que pueden ayudar con la adquisición, manipulación y envió de los datos, así como también de operaciones de índole visual donde realizan funciones con el fin de mejorar la apariencia de la web híbrida. Estos componentes son opcionales sus funciones se las puede hacer de forma manual, es decir, colocar el contenido de lo que tendría el widget en cada sitio donde lo necesitemos.

La cuarta capa contiene la lógica de la aplicación web híbrida. Los diversos componentes que se disponga son combinados en un mashup, en donde se pueden comunicar entre sí e intercambiar información. En esta capa todos los datos están escritos en el mismo idioma, es decir, el lenguaje con el que se comunican es homogéneo, ayudando con esto a mostrar al usuario lo que necesita de una manera eficaz y eficiente.

En la quinta capa se definen seguridades y políticas, es decir, desde cómo va a ser lanzado la web híbrida, publicidad, costes, hasta temas de quien puede acceder a ciertos elementos y el porqué de los permisos, aquí se estructura el organigrama de la aplicación, dando a conocer todas las jerarquías existentes y quien pertenece a cada nivel.

[5] Componentes cuya función es la de realizar tareas individuales pequeñas.

1.2 Integración de datos

Los Mashups por su naturaleza, son aplicaciones compuestas las cuales crean un nuevo producto y/o servicio a través de la integración de los componentes existentes. La integración es un tema importante en gran medida investigado en ingeniería de software y gestión de datos. Un número cuantioso de aplicaciones se ven afectados por este problema, en donde influye de manera significativa la integración en áreas como: Integración de la información empresarial, Integración de aplicaciones empresariales y la composición de servicios. Los problemas centrales que subyace a estas áreas y al desarrollo en sí mismo del mashup, tienen que ver con la dificultad de saber cómo adquirir y reunir datos de diferentes fuentes e integrarlas en una sola a fin de obtener una visión mejor estructurada de ellos, otros de los problemas son al momento de interconectar y orquestar diferentes componentes de software, así como para obtener una lógica integrada al momento de ponerlo en ejecución.

El desarrollo de mashups gira principalmente a la integración de sus partes individuales en un solo producto por ello es importante conocer sus problemas más notables y soluciones para la integración de aplicaciones.

La integración de datos se refiere a un conjunto de métodos cuyo objetivo es el de combinar datos de diferentes fuentes para proporcionar una visión única y unificada. Las actividades de integración de datos cubren diferentes formas de reutilización de la información, tales como mover datos de una base de datos a otra, traducir mensajes para las transacciones de negocio a negocio, o proporcionar acceso a los datos y documentos estructurados a través de portales de Internet. La necesidad de integrarlos datos se deriva especialmente de la evolución de las tecnologías especialmente en la red, que tiene un papel importante en la forma de como los sistemas de información están diseñados hoy en día.

En la mayoría de casos de integración, sucede con frecuencia que los datos provenientes diferentes fuentes son heterogéneas con respecto a los formatos de datos adoptados por la aplicación contenedora, tales como: archivos de texto, páginas web, documentos

XML, bases de datos, esquemas y los métodos de acceso a la información. Ellos tienen como objetivo el de ofrecer un sistema integrado de los sistemas de los cuales se obtiene la información, sea cual sea el enfoque de integración adoptado se trata de ocultar a los usuarios la distribución y la heterogeneidad de las fuentes subyacentes y proporcionar una visión homogénea de los datos integrados.

1.3 Integración de aplicaciones

Si dejamos de lado la integración de fuentes de datos heterogéneas para conseguir la integración completa de sistemas de software que son diferentes en su naturaleza y funcionalidad, para estudiar la integración de aplicaciones. En el que radica el problema que se da al desarrollar una lógica de la aplicación integrada, y del cómo se comunican y orquestan los múltiples sistemas heterogéneos al estar en ejecución. Por lo tanto, el problema ya no es estático como en los datos, sino más bien de naturaleza dinámica ya que intervienen diferentes metodologías y arquitecturas para cada sistema por separado, que al tratar de unirlos debemos tomar en cuenta cada uno de los factores importantes al momento de integrarlos en un solo sistema haciéndolo homogéneo al usuario.

Los sistemas que integran múltiples componentes de software y hardware son conocidos como sistemas distribuidos. Tanenbaum (1995) define un sistema distribuido como una colección de equipos independientes que aparece a los usuarios como un sistema único y coherente.

Es común que los sistemas distribuidos comprendan no sólo las máquinas con hardware donde sus configuraciones sean diferentes, sino también máquinas con diferentes configuraciones de software, tales como diferentes sistemas operativos. Esta heterogeneidad es exactamente la razón por la cual existe la necesidad de un componente en el desarrollo de aplicaciones distribuidas que no es necesario en el desarrollo de aplicaciones convencionales o no distribuidas, este componente es un middleware. El middleware es un software que permite e interactúa como mediador entre la heterogeneidad de la red y las interacciones entre los componentes de una

aplicación distribuida. Es decir, middleware proporciona una capa de sistema distribuido que comprende hardware y sistemas operativos heterogéneos y brinda una visión abstracta de sus capacidades. Esta visión abstracta se proporciona a las aplicaciones distribuidas que se ejecutan en la parte superior, a fin de permitir y facilitar su desarrollo. Como tal, los servicios de middleware son normalmente utilizados por los desarrolladores que programan aplicaciones distribuidas. Middleware, por tanto, representa, por un lado, una infraestructura de computación sobre el cual distribuye aplicaciones se pueden desarrollar y, por otro lado, una abstracción de programación que facilita el desarrollo.

1.4 Tecnologías Web

Al desarrollar una aplicación web híbrida, se deben considerar las actuales tecnologías de lenguajes de programación y los estándares Web (W3C), en los que se sugieren los requerimientos que debe cumplir un proceso o producto para mejorar en características como: rendimiento, interoperabilidad, eficacia, robustez. La Web ha ido evolucionando mediante la introducción de capacidades de la lógica de negocio en el lado del cliente, y el uso de estas tecnologías promovió la aparición de los mashups. La adopción de prácticas de desarrollo ligero, basado en la integración de los recursos por lado del cliente, es uno de los aspectos que más diferencian mashups de otras prácticas de integración. Sin embargo, otras clases de mashups también pueden requerir tecnologías del lado del servidor. Esto sucede, por ejemplo, para mashups empresariales donde se tiene una necesidad de la integración de los activos de los datos de la empresa y recursos remotos, en conjunto con sistemas de flujo de trabajo complejos.

A continuación, se estudiarán las tecnologías Web más relevantes, sus ventajas y limitaciones que ofrecen al momento de desarrollar mashups. Ilustrando los componentes básicos de la Web, por ejemplo, el Protocolo de transferencia de hipertexto (HTTP) y el Hypertext Markup Language (HTML) y, las diferentes tecnologías del lado del cliente que en los últimos años han llevado al desarrollo de aplicaciones ricas e interactivas.

1.4.1 Internet

A pesar de su extraordinario crecimiento y gran éxito, el Internet no tiene tantos años; para encontrar sus raíces debemos ir a la década de los 50, durante la Guerra Fría, Estados Unidos da surgimiento la agencia de investigación ARPA (Advanced Research Projects Agency) para poder combatir al avance tecnológico que Rusia mostraba. El departamento ARPA identificó una debilidad en su infraestructura de comunicación, la cual fue construida en la red de telefonía pública, por esta razón, vulnerable a ataques militares. El objetivo del departamento ARPA consistía en desarrollar una infraestructura de red que fuese más robusta y mucho más difícil para ser vulnerada por las fuerzas enemigas. Al final de la década de los 60 nació una primera red prototipo que tenía las características necesarias para cumplir con los requerimientos descritos, a esta red se le llamo ARPANET.

Esta red permitía el intercambio de información a través de la conmutación de paquetes y técnicas de enrutamiento distribuido, en los que se especificaba además de los datos a enviar, la ruta detallada que debe seguir el paquete para llegar a su destino; lo que permitió que cada nodo de la red tenga la posibilidad de enviar y recibir paquetes, consiguiendo con esto una mejora considerable de la robustez de la red. En 1975, el protocolo de Transmisión Control Protocol / Internet Protocol (TCP/IP) con el tiempo también permitió establecer comunicaciones entre múltiples infraestructuras de red.

Hoy en día, el internet es indispensable en nuestras vidas, y su constante evolución permite que la comunicación entre personas que no ocupan físicamente el mismo lugar,sin embargo, se puedan ver; las compras por internet sin necesidad de salir de casa, la adquisición de conocimiento sin ningún tipo de barrera, en fin, el internet se ha transformado en la herramienta que hace nuestra vida mucho más práctica y cómoda. Todo el mundo puede ser parte de la Internet. Como Tanenbaum describe,
"Una máquina está en Internet si ejecuta procesos con protocolos TCP/IP, tiene una dirección IP, y puede enviar paquetes IP a todas las otras máquinas de la red." (Tanenbaum, 2002)

1.4.2 Protocolo TCP/IP

Toda la comunicación en Internet se implementa por los procesos del protocolo TCP/IP, y actualmente este se considera estándar para toda la Internet. En la siguiente figura se compara TCP / IP con el estándar Open Systems Interconnected (OSI) el cual se usa comúnmente como un modelo de referencia teórica, en la práctica la mayoría de las redes de ordenadores implementan el protocolo TCP / IP.

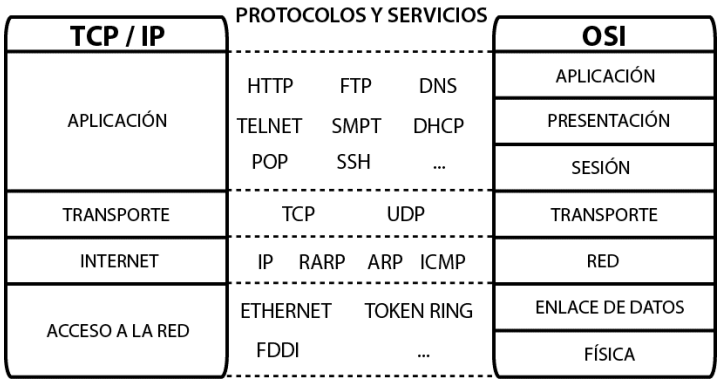


Fig. 4 Composición, protocolos y servicios de modelos TCP/IP y OSI

La capa inferior del modelo TCP / IP está representado por las capas física y de enlace de datos, del modelo OSI. Esta capa se encarga del enrutamiento, sincronización, formato, conversión analógica/digital de los datos, así como también de detectar errores en los paquetes entrantes en cada nodo.

El objetivo real del protocolo TCP/IP comienza con la capa de internet, la más importante, ya que es la encargada del intercambio de paquetes IP entre los diferentes nodos de la red. La capa de internet se preocupa del camino que deben tomar los paquetes para llegar a su destino, verificando sus cabeceras y escogiendo la mejor ruta para enviarlos a la capa de transporte.

La capa de transporte recibe los paquetes y comprueba que no contengan errores, ya sea en sus datos o en la dirección de destino, los ordena según su prioridad y de manera secuencial construye cada paquete para ser enviado a sus correspondientes destinos. En esta capa existen dos tipos de protocolos de transmisión de datos: Transmission Control Protocol (TCP) y User Datagram Protocol (UDP), el primero siendo un protocolo

orientado a conexión, es decir, una vez que se estableció la conexión el intercambio de información se lo puede hacer de manera bidireccional. Utilizado en la elaboración de aplicaciones donde cada paquete es importante, para la cual TCP nos brinda la facilidad de detectar errores en el intercambio de comunicación. UDP es un protocolo menos confiable y sin necesidad de conexiones, por ello es más fácil mandar grupos de paquetes al mismo tiempo, utilizado en aplicaciones donde la pérdida puede ser aceptable.

Tanto en el modelo OSI como en el protocolo TCP/IP tenemos la capa de aplicación, la cual contiene las funcionalidades de red, encargadas de establecer la comunicación mediante los niveles inferiores de la arquitectura. Esta capa tiene la responsabilidad identificar y establecer la disponibilidad del destino para poder establecer la conexión, así como determinar los recursos para que exista esa comunicación.

1.4.3 Protocolo Http

Un usuario mediante el uso de una aplicación como puede ser un navegador web y mediante el uso del protocolo HTTP, puede acceder a los recursos disponibles en un servidor remoto o servidor Web. Por lo general los recursos intercambiados a través de HTTP son páginas web, es decir, documentos codificados en HTML que muestran en pantalla: texto, objetos multimedia, y enlaces a través del cual el usuario puede navegar a otros documentos HTML.

Cuando el usuario introduce una dirección de página web en el navegador, la página de llamada URL (Uniform Resource Locator), o cuando el usuario hace clic en un vínculo de una página, el explorador emite una solicitud HTTP, la dirección URL del recurso solicitado, y la versión del protocolo. Cuando se recibe la solicitud, el servidor localiza el recurso solicitado y lo envía como una respuesta al cliente. El mensaje de respuesta incluye una línea de estado, es decir, información sobre la versión del protocolo y un código de estado numérico con su mensaje asociado (por ejemplo, "HTTP / 1.1 404 Not Found"), y un cuerpo de mensaje que lleva el recurso real para ser intercambiado.

El protocolo HTTP se basa en una serie de métodos que se puede llamar cuando una se emite la solicitud. Los más importantes son GET y POST. El método GET somete una solicitud normal para un recurso en el servidor Web, y permite al cliente enviar entradas simples a través de una cadena de consulta anexa a la URL. El método POST envía una petición que le permite al cliente enviar entradas complejas (por ejemplo, un texto largo o un archivo) para ser procesados por el servidor.

1.5 Componentes de una aplicación web híbrida

Cada vez es más frecuente desarrollar aplicaciones web utilizando las APIs existentes que se obtienen desde otros sitios, y que permiten generar nuevas aplicaciones. Un aspecto importante es la disponibilidad de los componentes de software para ser reutilizables.

Un componente Mashup es cualquier pedazo de datos, lógica de la aplicación, y / o interfaz de usuario que puede ser reutilizado y que es accesible de forma local o remota. (Florian Daniel, 2009)

1.6 Lógica de componentes

Para poder reconocer el tipo de componente, es necesario identificar en que podemos reutilizarlo, es decir, en que ámbitos de una aplicación lo podemos insertar.
La lógica de componentes proporciona acceso a la lógica del negocio, por ejemplo, un componente lógico de funcionalidad de pago o la reutilización de algún algoritmo.

1.6.1 Servicios Web

Es un componente de software que está disponible a través de la red accesible mediante protocolos que ayuda a la computación entre sistemas heterogéneos, accediendo a funcionalidades de un objeto. En un servicio web tenemos acceso a los métodos que

pueden ser de entrada y salida. El tipo de llamada más común es mediante SOAP (Simple Object Access Protocol).

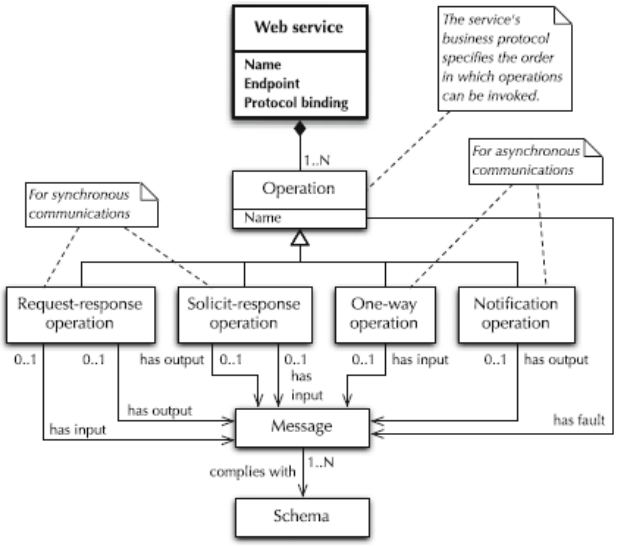


Fig. 5 Modelo conceptual de un servicio web

Fuente: (Florian Daniel, 2009)

1.6.2 RESTFul Web Service

RESTful (Transferencia de estado representacional) es un servicio web que no está basado en la orientación de objetos WSDL/SOAP, sino más bien tiene como objetivo proporcionar servicios directamente en la parte superior de HTTP de una arquitectura orientada a servicios y donde es formateada en JSON o XML.

El RESTful soporta 4 tipos de operaciones de mensajes-basados y está definida por el protocolo estándar HTTP.

- Get: Es empleada para obtener la información del recurso que está identificado por la url.
- Post: Crea un nuevo recurso, del recurso que está identificado por la url.
- Put: Actualiza un recurso existente identificado por la url.
- Delete: Elimina un recurso existente identificado por la url.

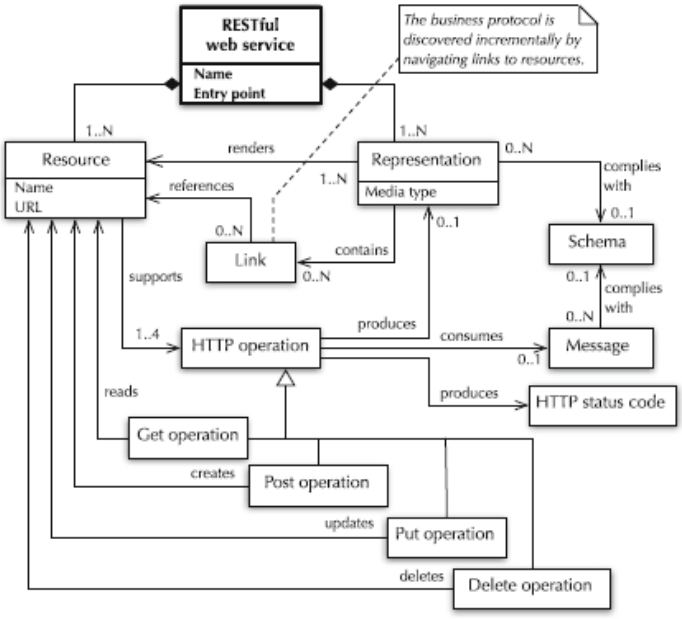


Fig. 6 Modelo Conceptual de un RESTFul Web Service

Fuente: (Florian Daniel, 2009)

1.6.3 JavaScript API

Permite reutilizar la lógica de negocios, como ejemplo podemos utilizar para almacenar datos a lado del cliente, con el XMLHttpRequest de manera asíncrona o síncrona.

Todos estos archivos que se pueden descargar y que proporcionan código reutilizable las llamamos bibliotecas, como ejemplo de biblioteca jquery que permite la manipulación del Modelo de Objetos del Documento (DOM) de manera sencilla. Una API tiene uno o varios objetos javascript donde tienen funciones que permite invocaciones. Las APIs de javascript aportan llamadas síncronas y asíncronas de transmisión de datos.

1.6.4 SOAP

SOAP es un protocolo simple y ligero basado en XML para intercambiar información estructurada y de tipos, el cual su diseño consiste en conseguir la más alta simplificación. Este protocolo define un marco para la mensajería que excluye del mensaje la semántica de la aplicación y el transporte.
SOAP consiste en 4 principales partes que son:

- Se define un sobre extensible para encapsular los datos el cual define un mensaje y método de intercambio entre mensajes.
- A continuación, las reglas de la codificación de los datos opcionales para representar tipos de datos definidos por cada aplicación y un modelo uniforme para datos no sintácticos.
- Se define el intercambio de mensajes, solicitud y respuesta donde cada mensaje es una transmisión unidireccional.
- Se define un enlace entre SOAP y HTTP y de esta forma se puede combinar SOAP con cualquier otro protocolo de transporte.

Fig. 7 Publicación de un servicio Web

(Vara, Juan, López, & Verde Marín, 2014)

1.6.5 Application Programming Interface (API)

Una API es una interfaz de programación de aplicaciones que a través de los métodos podemos comunicarnos con una aplicación externa. Las API dependen del lenguaje que este escrito, pero es mucho mejor que los métodos puedan ser llamados por una cantidad considerable de lenguajes.

Podemos llamar a una API, mediante una URL desde una página web, podremos acceder a sus métodos y ejecutar toda la funcionalidad, mostrando en nuestro navegador los resultados.

1.7 Componente de Datos

La forma de acceder a los datos provenientes de un servicio web puede ser: de manera estática, es decir mediante la lectura de un archivo RSS enviado como respuesta a la petición de la aplicación, o de manera dinámica, mediante consultas específicas enviadas al servicio web.

1.7.1 Really Simple Syndication

RSS o formato de sindicación de contenidos para la Web, es uno de los formatos más usados por los administradores de contenidos de los Sitios de internet, donde permite a los sitios compartir su contenido con otras aplicaciones. El formato se compone de un XML (eXtensible Markup Language) que utiliza una estructura sencilla, en el cual, los elementos de contenido que describen el articulo como el título, enlace y la descripción son requeridos, en cambio que fecha de publicación, imagen y similares son opcionales.

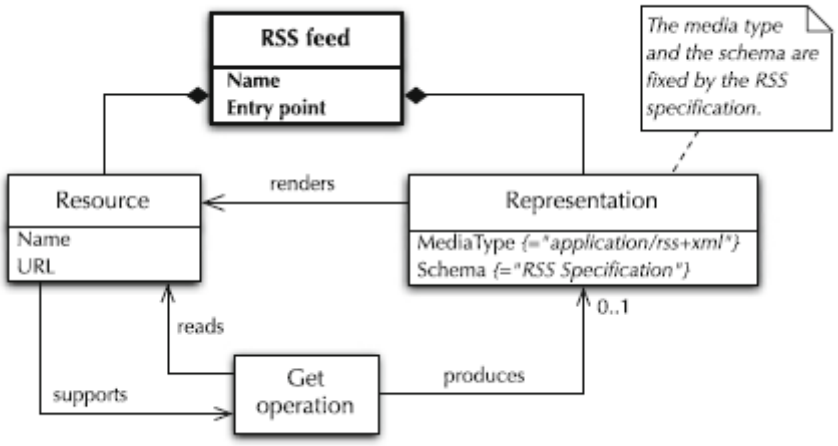


Fig. 8 Modelo Conceptual Formato de Sindicación de contenidos (RSS)

Fuente: (Florian Daniel, 2009)

1.7.2 ATOM

Atom Publication Protocol es un lenguaje XML usado por las páginas web que brindan contenido, y es muy similar al RSS, este permite a un software tener la capacidad de verificar actualizaciones de contenido de un sitio en específico. Nació con el objeto de reemplazar a RSS debido a su contenido limitado y por la gran incompatibilidad de versiones de RSS y una baja interoperabilidad.

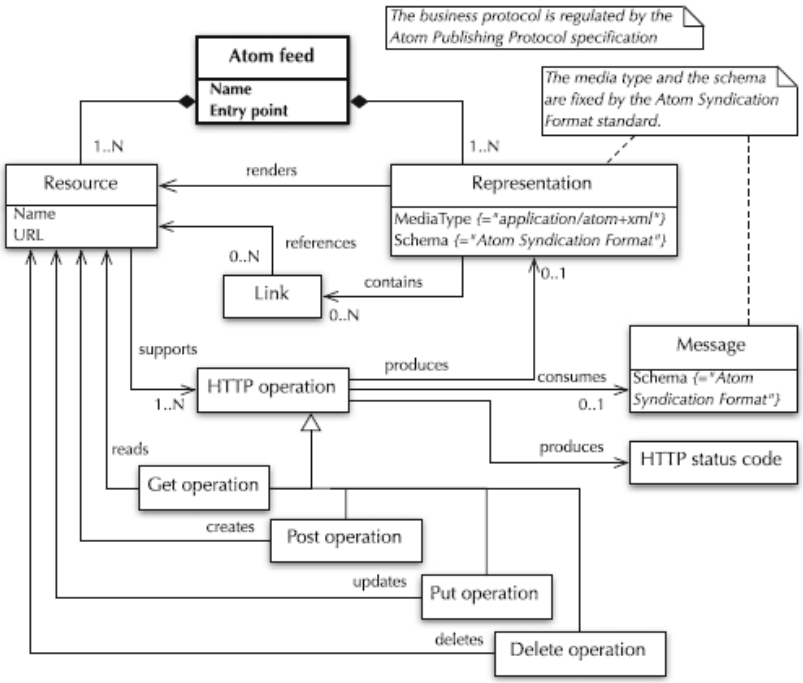


Fig. 9 Modelo Conceptual ATOM

Fuente: (Florian Daniel, 2009)

1.7.3 XML, JSON, CSV

Cuando los datos se encuentran en tránsito generalmente se encuentran estructurados en XML, JSON, CVS.

1.7.3.1 XML

Usa etiquetas con corchetes que pueden ser incrustadas con otras etiquetas que también puede crear una jerarquía de información. Es un lenguaje utilizado por documentos y esquemas de codificación. Los XML no tienen etiquetas definidas, es decir, cada etiqueta es establecida por el usuario, por lo tanto, podemos decir que XML es una estructura jerárquica de datos establecida por el usuario.

```
<encounter id='111'>
<vitals>
<weight units="lbs">140</
weight>
</vitals>
</encounter>
```

Fig. 10 Estructura de un documento XML

1.7.3.2 JSON (Javascript Object Notation)

Es un formato de intercambio de datos que es usado comúnmente por los servicios web.

```
{
"encounter": {
"id": "111",
"vitals": {
"weight": {
"units": "lbs",
"weight": "140"
} } } }
```

Fig. 11 Estructura de un documento JSON

1.7.3.3 CSV

Es un formato que esencialmente contiene datos representados como hojas de cálculo, donde las columnas están separas por comas y cada fila por saltos de línea.

```
(This "vitals" csv would be one of
several csv files needed to represent
these data.)
Encounter,weight,units
111,140,lbs
```

Fig. 12 Estructura de un documento CSV

1.7.3.4 Screen scraping (raspado de pantalla)

Es una técnica de programación que extrae la información de un sitio web mediante la utilización de ingeniera inversa, logrando conseguir todo el código fuente de la página,

es decir, su contenido HTML. El propietario de la página web, a la que se le aplica el raspado de pantalla, desconoce la ejecución del procedimiento para la obtención de información.

1.8 Componentes de Interfaz de usuario

Generalmente viene con su propia interfaz de usuario y probablemente de su lógica y datos donde muestra contextualmente contenido o funcionalidad de la interfaz de usuario. En las aplicaciones web el contenido se mezcla usando lenguaje del lado del cliente tales como JavaScript, CSS, HTML/XHTML.

1.8.1 Lenguaje HTML

El lenguaje HTML (Hiper Text Markup language / Lenguaje de marcado de hipertexto) es un estándar para el desarrollo de contenidos web el cual se basa en el uso de etiquetas. A diferencia de otros lenguajes los archivos HTML no son compilados sino interpretados por los navegadores

```
<!DOCTYPE html>
<html>
   <head>
     <title>(aquí va  el título de la página)</title>
   </head>
   <body>
        (aquí va el resto del código html)
   </body>
</html>
```

Fig. 13 Estructura de un documento HTML

1.8.2 CSS

El lenguaje CSS (Cascading Style Sheets / Hojas de Estilo en Cascada) son complementos de código añadidos al HTML, que tienen como función principal, separar el funcionamiento de una web de su contenido, es decir, la estructura de la parte visual.

```
h1
{
color:orange;
text-align:center;
}
p
{
font-family:"Times New Roman";
font-size:20px;
}
```

Fig. 14 Estructura de una Hoja de Estilo

1.8.3 JavaScript

Se presenta como un lenguaje muy sólido el cual está basado en objetos para el desarrollo de aplicaciones cliente / servidor.

Los bloques de código de JavaScript pueden estar directamente presentes en el documento HTML o escritos en un archivo js de manera de totalmente aparte.

```
<script language="javascript">
var i,j=0;
for (i=0;i<2;i++) {
alert("este es el valor de mi primer contador i: "+i);
    for (j=0;j<2;j++) {
    alert("este el valor de mi segundo contador j: "+j);
    }
)</script>
```

Fig. 15 Estructura de un documento JavaScript

1.8.4 Widget y Gadgets

Un widget es un programa pequeño con código reutilizable que se puede añadir a cualquier página web, un gadget al contrario solo funcionará en un determinado sitio o conjunto de sitios donde se instale el complemento. La disponibilidad de servicios web, APIs hace que sea fácil el desarrollo de estos componentes.

1.8.5 Widgets

Son envoltorios o elementos composicionales con los que ha de interactuar el usuario, y/o elementos dedicados a realizar tareas específicas (acceso instantáneo a la web 2.0) si hablamos de widgets html. Cada widget que componen una aplicación, tienen una funcionalidad establecida que es independiente de los demás, pero cuando se relacionan entre ellos el resultado incrementa de manera significativa. La World Wide Web Consortium (W3C) define los widgets como aplicaciones interactivas de propósito simple, las cuales muestran contenido de manera visual o actualizan los datos locales de la aplicación o del sitio web. (Kaar, 2007) dice en su publicación, que un widget esta empaquetado de tal forma que permita una descarga simple y se instale en la máquina del usuario o en un dispositivo móvil.

Para el desarrollo de estos componentes se utilizan tecnologías web estándar tales como: HTML, CSS, JavaScript, XML entre otras. Se ejecutan en un entorno de tiempo especial, ya que son ejecutados tras haber recibido alguna interacción por parte del usuario, por lo que para su desarrollo la tecnología por excelencia para su desarrollo es AJAX, ya que trabaja de manera asíncrona a la web o aplicación.

1.8.5.1 Gadgets

Los gadgets web por lo general son archivos de formato XML que se encuentran alojados en el internet. Este archivo contiene las instrucciones del cómo ejecutarse en el lugar en donde sea invocado o implementado. Un gadget por lo general, tiende a ser de proporciones pequeñas, son muy prácticos y novedosos. La aparición de los gadgets mostrando sus funcionalidades y vistosidad fue en el sistema operativo Windows 7, donde venían instalados y el tiempo donde este S.O fue el mejor de su empresa, el desarrollo de gadgets de escritorio parecía ir en incremento. Pero fue desapareciendo en los nuevos sistemas operativos creados por Windows.

1.8.5.2 Clasificación de los Widgets y Gadgets

En la actualidad se puede clasificar estos componentes como: de escritorio, web y móviles. Todos comparten las mismas características, sin embargo, los componentes web, pueden ser incrustadas en otros sitios o aplicaciones sin necesidad de descargar y/o instalar.

2 Análisis y definición de arquitectura para una aplicación web híbrida

2.1 Análisis de protocolos de comunicación

Las aplicaciones web híbridas, para acceder a la información necesaria para su funcionamiento, lo efectúa mediante una API en donde estos servicios son ofrecidos y consumidos a través del protocolo de comunicaciones HTTP.

Antes de introducirnos al lenguaje de desarrollo de una API tenemos que primero conocer sus protocolos, ya que la interoperabilidad depende del cómo el navegador interactúa con el sitio web. Estos protocolos se basan en una arquitectura llamado Representational State Transfer (REST).

La comunicación con los servicios web puede estar basados en el protocolo HTTP, pero no es una regla, es decir, pueden utilizar otros protocolos de comunicación pues esta capa, que es la de séptimo nivel en el modelo OSI y el cuarto en TCP/IP, es independiente de las otras capas.

2.1.1 Arquitectura aplicación web híbrida

La arquitectura de una aplicación web híbrida está formada por 3 partes: por el proveedor de contenidos, el servidor web de la aplicación y el navegador del cliente.

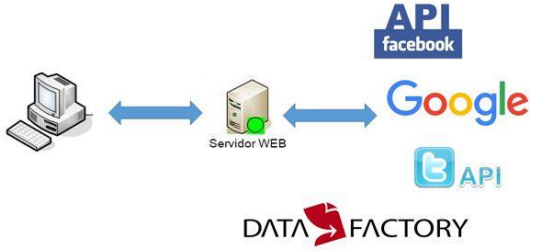


Fig. 16 Elementos de la arquitectura de una aplicación web híbrida.

En la figura anterior visualizamos las partes de una aplicación web híbrida donde a la derecha tenemos al proveedor del contenido (Facebook, Twitter, Google, DataFactory), a la izquierda tenemos el navegador de cliente que accede a la aplicación web híbrida, en el centro encontramos el servidor web de la aplicación web híbrida. Nuestro programa es como el sistema principal y las APIs externas son un subsistema entonces se produce un nuevo resultado.

La comunicación de una arquitectura en una aplicación web híbrida puede tener diferentes vías de comunicación en función del navegador del cliente donde puede o no pasar por el servidor web de la aplicación.

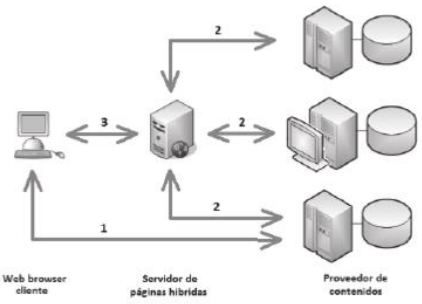


Fig. 17 Comunicación de elementos de una arquitectura de aplicación web híbrida *(Vara, Juan, López, & Verde Marín, 2014)*

Los casos de la figura anterior los detallamos a continuación.

1. El navegador cliente accede directamente a la información del proveedor de contenidos mediante JavaScript.

2. El servidor de aplicaciones obtiene los datos directamente del proveedor de contenidos, estos se mezclan en el servidor y son enviados al navegador del cliente.

3. El navegador del cliente accede al servidor web y extraer el contenido almacenado.

La estructura de una web híbrida, basa su funcionamiento en las arquitecturas orientadas a servicios (SOA), las cuales ayudan en la interoperabilidad de los componentes integrados. Estas arquitecturas dan fuerte énfasis en la integración entre sistemas heterogéneos. Los dos requisitos que una SOA debe tener son:

Los distintos componentes integrados en una web híbrida, deben tener una forma de comunicación, es decir, un lenguaje común y un protocolo común que permita el intercambio de información. Este medio debe permitir establecer y mantener el flujo correcto de intercambio de datos entre los servicios, permitiendo al servicio generado a partir de la integración, el poder comunicarse con cada servicio de manera individual o simultánea, ya que su arquitectura es común para cada elemento de datos proveniente de los servicios.

Para el diseño de aplicaciones híbridas lo podemos hacer de dos maneras:

Basados en la web. Utiliza el navegador web del usuario para obtener e integrar la información de los distintos servicios, y mostrar el resultado inmediatamente al usuario. Utiliza Javascript para lograr este fin. Toda la manipulación de los datos solo se hace del lado del cliente. Las tecnologías actuales de este tipo son:

- AJAX
- RSS, RDF, ATOM
- CSS

Basado en el servidor. La manipulación de los datos se lo realiza tanto en el navegador del cliente, como en el servidor web. En el servidor se integra la información y se envía a los usuarios mediante HTTP Request. En este enfoque, se aisla la integración del navegador, es decir, se integra en un servidor remoto y luego el resultado se envía para que lo visualice el cliente. Las tecnologías presentes son:

- SOAP, REST

- Lenguajes Script como Javascript, PHP, etc.
- Frameworks tales como Django, Jruby, etc.

2.2 Compatibilidad de componentes

Uno de los principales inconvenientes al trabajar con componentes software heterogéneos es el tratar de conseguir una separación clara entre las características de interacción (funcionalidad propia del software) y de computación (requerimientos de hardware), para, de esta manera permitir la reusabilidad de componentes, facilitando en gran medida el análisis completo y claro de la aplicación.

La Ingeniería del Software tiene como meta, elaborar software de calidad. El objetivo principal de las personas que trabajan en el desarrollo de software es que siempre sus aplicaciones sean rápidas a las interacciones de los usuarios, seguras y que mantengan la integridad de los datos, fáciles de usar, legibles para los usuarios, que posean programación modular y estructurada, etc. Propiedades como la facilidad de uso y rapidez del sistema de responder a necesidades, son detectadas principalmente por los usuarios del producto, incluyendo a los desarrolladores. Por otro lado, existen características que sólo son pueden ser percibidas por personas profesionales o con conocimientos en el área de la informática, estas propiedades pueden ser: modularidad o legibilidad. Esta diferencia para poder detectar la falta o presencia de una propiedad en el componente software, nos permite clasificar a los factores que definen la calidad del mismo en: factores externos e factores internos.

A continuación, se presenta una tabla abreviada, que contienen las características pertenecientes al ciclo de vida del software, estas métricas a tener en cuenta al momento de desarrollar.

Ciclo de vida	Atributos	Tipo de métrica

Idoneidad	• Cobertura • Integridad • Pre y post condiciones	Porcentual Porcentual Presencia
Interoperabilidad	• Compatibilidad de datos	Presencia
Cumplimiento	• Estandarización • Certificación	Presencia Presencia
Autosuficiencia	• Confiabilidad	Porcentual
Madurez	• Volatilidad • Eliminación de fallos	Valor Valor
Facilidad de compresión	• Documentación disponible • Calidad de la documentación	Presencia Presencia
Facilidad de aprendizaje	• Tiempo y esfuerzo para uso, configuración, administración, etc.	Valor
Operabilidad	• Nivel de complejidad • Interfaces proporcionadas • Interfaces requeridas • Esfuerzo operativo	Porcentual Valor Valor Presencia
Facilidad de modificación	• Extensibilidad • Personalización	Porcentual Presencia
Facilidad de testeo	• Batería de pruebas proporcionadas. • Pruebas extensivas.	Presencia Presencia

	• Pruebas en un entorno específico. • Pruebas formales	Presencia Presencia
Adaptabilidad	• Movilidad. • Capacidad de configuración	Presencia Porcentual
Facilidad de reemplazo	• Compatibilidad con versiones anteriores.	Presencia
Reusabilidad	• Nivel de abstracción del dominio. • Nivel del código transversal. • Compatibilidad de arquitectura. • Modularidad	Porcentual Porcentual Porcentual Porcentual

Tabla 3 Ciclo de vida del software

Al momento del desarrollo de aplicaciones web híbridas, los atributos más importantes a considerar son: un factor interno como es la reutilización y otro externo como la compatibilidad de los componentes. Cabe destacar que, todos los factores expuestos en el ciclo de vida de los componentes de software, se deben tomar en cuenta al momento del análisis, desarrollo e implementación.

La reutilización de un componente indica la capacidad que posee el mismo para ser utilizado nuevamente, ya sea, de manera total o parcial en otros productos software, con el fin de evitar que se genere trabajo redundante para la resolución de problemas ya solucionados con anterioridad. La programación de una aplicación se la debe realizar mediante la definición de módulos independientes, en donde cada módulo tenga la menor o nula dependencia de los demás para su correcto funcionamiento, y donde las

utilidades de las que hace uso la aplicación (funciones, validaciones, procedimientos, etc.), deben ser lo más generales y concisas como para ser integradas tal cual por otros productos.

2.3 Compatibilidad de datos

Al momento de integrar se tiene que considerar que los datos provenientes de las distintas fuentes externas e internas, no tengan ningún problema en su compatibilidad, es decir que debe existir armonía y adaptación entre la información requerida para integrar. La aplicación debe tener inherente un medio por el cual las diferentes aplicaciones depositen sus datos y en esta capa se traduzca de ser necesario a un lenguaje que pueda ser utilizado por todos los componentes, esta traducción puede ser unilateral o también se puede dar en ambos sentidos de ser necesario.

Al ser un ambiente web el empleado para este caso de estudio, la posibilidad de variación de lenguaje entre componentes es muy reducido, ya que, bajo el estándar web, hoy en día son pocos los lenguajes que se ejecutan bajo esta plataforma.

2.4 Arquitectura de integración e interoperabilidad.

En la actualidad existen dos tipos de modelos de aplicaciones web híbridas, del lado del cliente y del lado del servidor.

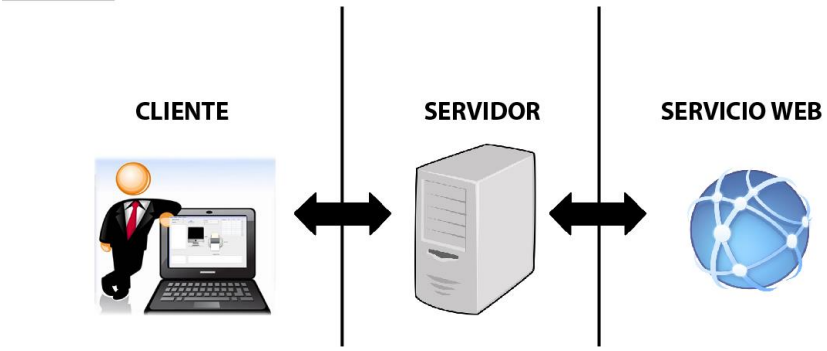


Fig. 18 Arquitectura de una aplicación web híbrida

Para el intercambio de información entre las diferentes capas de una web híbrida, se utilizan diferentes opciones tales como: RSS-Feeds, API´s, widgets, o interfaces web que nos proporcionan herramientas que ayudan a implementar e integrar varios servicios que se tiene disponibles en la nube. La información se encuentra alojada en servidores, y para acceder a ella, el cliente debe enviar peticiones a través de los métodos GET y POST, el servidor analiza las solicitudes e intercambia la información que necesita el cliente. En este proceso el servidor actúa como un proxy entre el navegador web del lado del cliente y los recursos del sitio que forma parte de la web híbrida, cliente – servidor – proveedor de servicios/contenidos.

2.4.1 Aplicaciones web híbridas del lado del servidor

Según (Brydon y Basler, 2009), estas aplicaciones mezclan el contenido en el servidor de la web y lo transfieren al cliente a través del protocolo HTTP (por ejemplo, RSS).

(Andreas Auinger, 2010), definen varios pasos que son realizados en las aplicaciones del lado del cliente y del servidor, los cuales son citados a continuación:

1. El usuario genera un evento, mediante la interacción con la aplicación, a través de un navegador web. El evento desencadena una función de JavaScript en el lado del cliente.

2. El cliente realiza una solicitud al servidor (proxy), el cual proporciona el contenido del sitio web. La solicitud es por lo general en AJAX, ya que permite obtener los datos alojados en el servidor de forma asíncrona, es decir, que obtiene los datos en segundo plano sin alterar con la visualización y el comportamiento de la página web.

3. Un componente web en el servidor recibe la petición y llama a un método que encapsula el código para conectar e interactuar con el otro sitio web en el mashup.

4. El servidor proxy establece una conexión con el sitio de la aplicación web híbrida.

5. El sitio mashup recibe la solicitud, la procesa y devuelve los datos al servidor proxy.

6. El servidor proxy recibe la respuesta y de ser necesario puede transformar los datos recibidos a un formato apropiado para los intereses del cliente.

7. El servidor proxy devuelve la respuesta al cliente.

8. La función de AJAX actualiza la página web del cliente con los resultados proporcionados por el servidor.

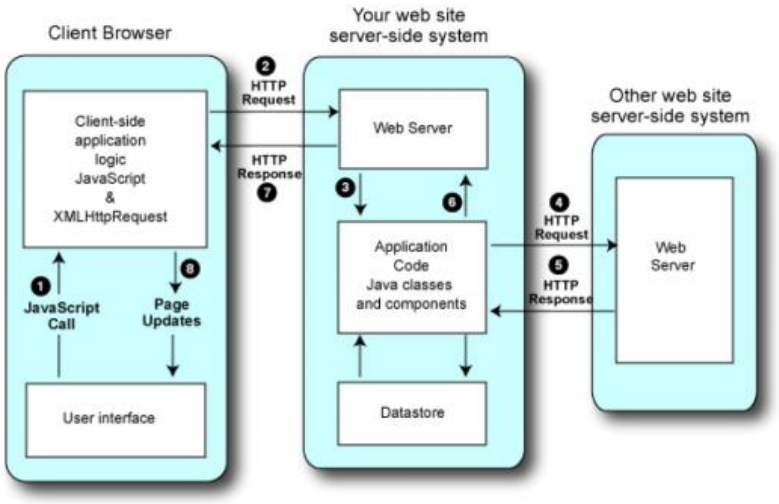


Fig. 19 Web híbrida del lado del servidor

2.4.2 Aplicaciones web híbridas del lado del cliente

En este tipo de aplicaciones los datos y componentes se integran de manera directa en el navegador web del cliente. En este modelo, para la comunicación entre el cliente y proveedor no es necesario la intervención de un servidor, sino que se la realiza de manera directa.

1. El navegador del cliente hace una petición al servidor mashup para la página web.

2. El servidor de la web híbrida carga la página en el cliente. La página incluye generalmente una biblioteca JavaScript para habilitar el uso de los servicios de contenidos/servicios.

3. Cualquier acción generada por el cliente en el navegador invoca a funciones JavaScript que retornan los contenidos/servicios.

4. Las funciones generadas son enviadas como petición a la web híbrida.

5. La web híbrida procesa las solicitudes y envía los datos en el formato solicitado.

6. La función de recuperación de datos, actualiza la página web del cliente con los resultados recibidos.

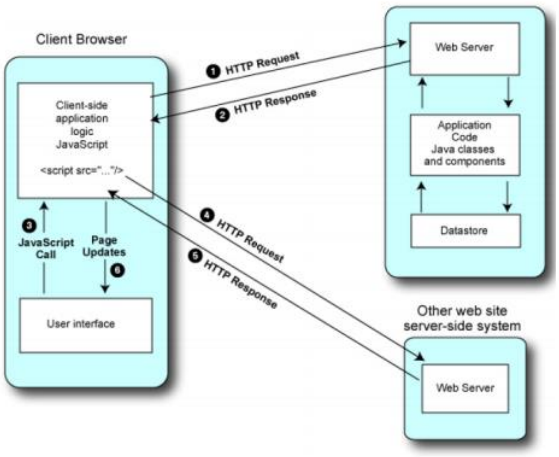


Fig. 20 Web híbrida del lado del cliente

2.5 Autentificación de aplicaciones

La cantidad en aumento de empresas y personas que desarrollan componentes que pueden ser utilizados como servicios web, de allí la importancia de las aplicaciones híbridas que tratan de integrarlos para generar un valor añadido a esos componentes que ni los propios desarrolladores imaginaron que podía llegar a alcanzar. Para esa integración se necesita actualmente una intervención manual considerable ya que para una orquestación entre los diferentes componentes que forman parte del mashup es necesario codificación extra ya que al no existir por el momento estándares para la creación de servicios web, los tipos de datos, arquitecturas y la estructura en sí de la aplicación, hace que los desarrolladores actuales tengan un desafío para encontrar el equilibrio donde el intercambio de información sea homogéneo.

La Generic Security Service Application Programming Interface (GSSAPI) proporciona servicios de seguridad para llamar a otras aplicaciones. Eso permite establecer una comunicación segura entre los componentes, para ello se pide al usuario que se autentifique con la aplicación asociada, para que se deleguen los derechos, manteniendo así confidencialidad e integridad de cada rutina de adquisición e intercambio de información.

Las aplicaciones que se comunican, establecen un conjunto de normas y seguridades proporcionadas al momento de llegar a un acuerdo mutuo. El intercambio de información pueden ser mensajes cifrados o un mensaje con una secuencia XML que solo pueda adquirir la aplicación autentificada. La aplicación que acepta los derechos de la que proporciona los servicios, debe mantenerlas e incluir las propias en el sistema final.

3 Implementación del Prototipo

En este capítulo se determinarán los requerimientos necesarios para el desarrollo e implementación de la aplicación web, el análisis y selección de componentes, la definición de las herramientas utilizadas para el desarrollo de la aplicación, así como la definición de los procesos y flujo de datos y los resultados obtenidos con pruebas unitarias y pruebas de interoperabilidad.

3.1 Casos de Uso

Los usuarios de esta aplicación, pueden ingresar al sistema mediante el botón "Registrar con Facebook" que se encuentra en la página principal, el cual, obtiene los datos del servicio externo de Facebook, y verifica esta información con la base de datos de la aplicación, si el usuario ya está registrado, se muestra su historial, caso contrario se almacena los datos de la persona y consecuentemente se permite el acceso al contenido de la aplicación.

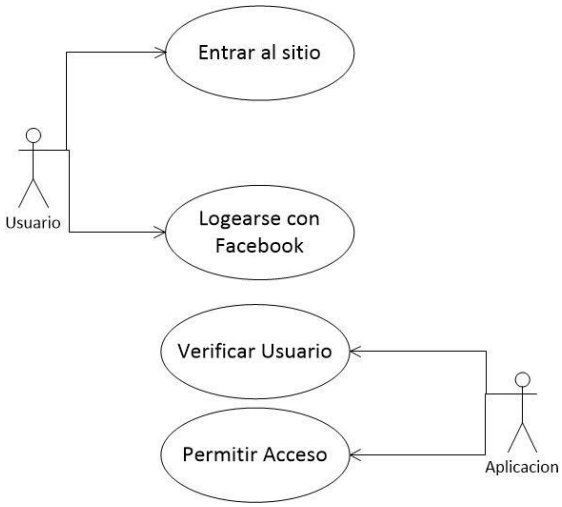


Fig. 21 Caso de uso para el registro del usuario.

El usuario registrado puede acceder a las opciones que presenta la aplicación, como: observar el calendario deportivo, realizar apuestas, visualizar historial de sus apuestas, etc.

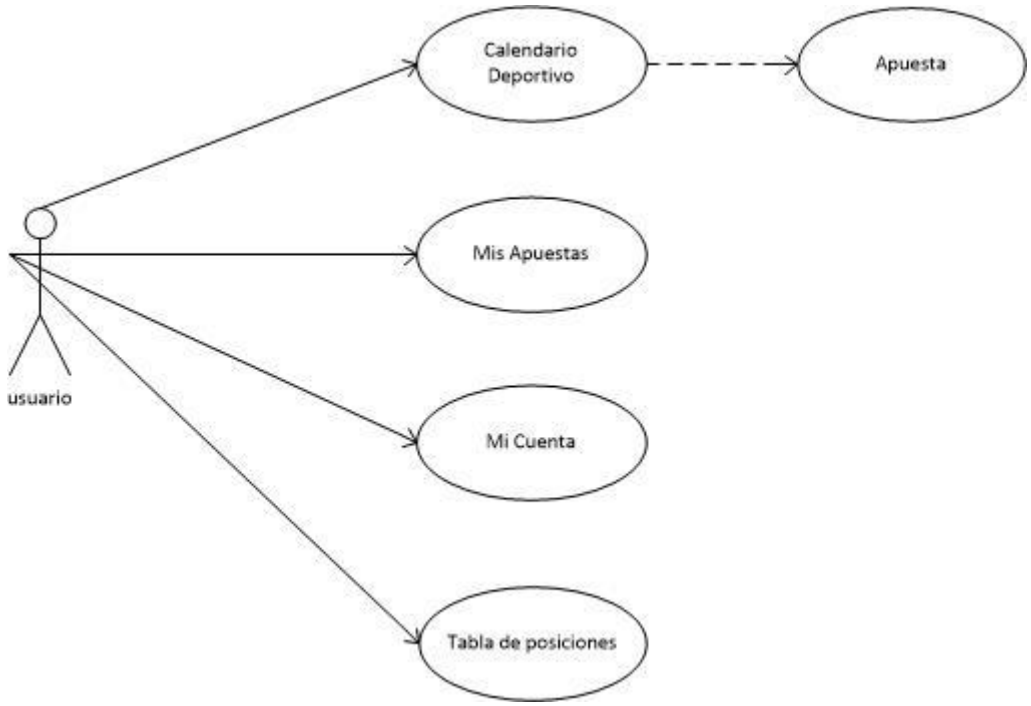


Fig. 22 Caso de uso del usuario registrado.

El administrador de la aplicación tras acceder al sistema, tiene las herramientas necesarias que le permiten gestionar la aplicación, el contenido presentado, y la información almacenada. Puede modificar o crear nuevos registros de las tablas que componen la base datos según sea necesario, así como también realizar consultas que pueden ayudar a verificar procesos y/o a tomar de decisiones estratégicas en la administración de la aplicación,

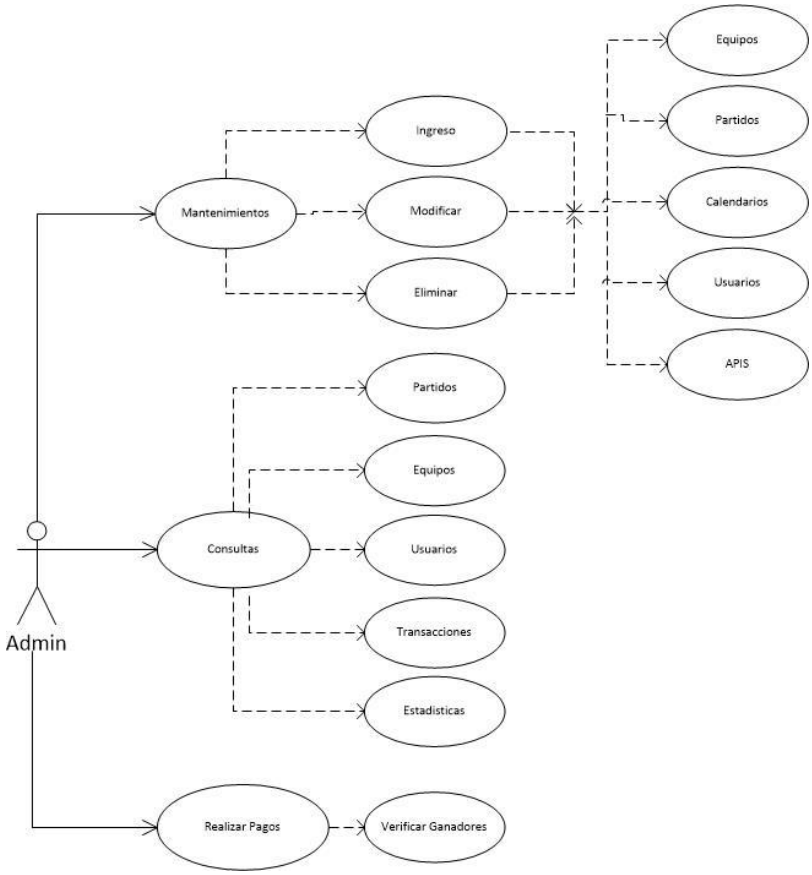


Fig. 23 Caso de uso para el usuario administrador de contenido y gestión del sitio.

3.2 Diagramas de Secuencias

El usuario realiza una petición para ingresar a la aplicación, la que, se conecta a la API's de Facebook y esta a su vez se encarga de dar respuesta a esa petición con todos los datos del usuario. La información obtenida se compara con la almacenada en la aplicación, de ser correcta se permite el acceso y se ingresa el registro del nuevo usuario en caso de serlo.

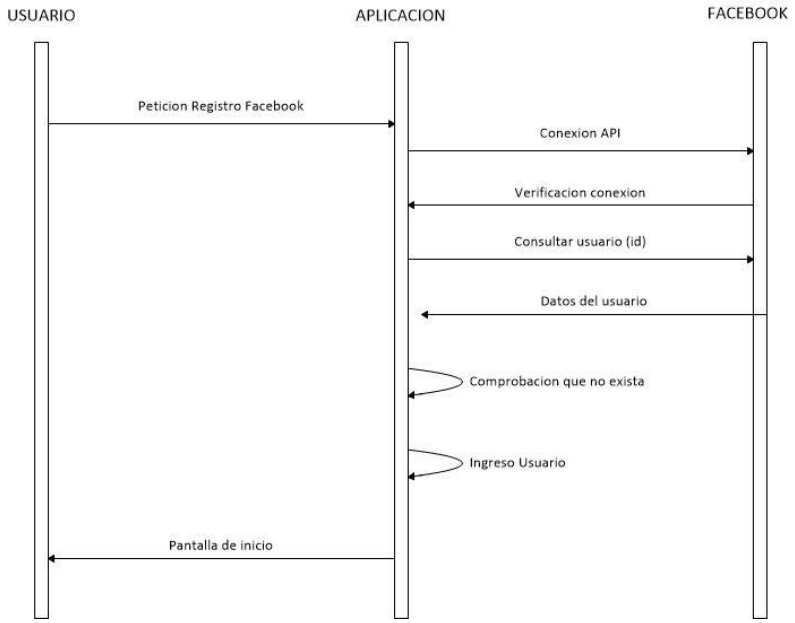


Fig. 24 Diagrama de secuencia de registro de usuario.

El usuario de la aplicación puede realizar consultas del calendario deportivo, para ello, la aplicación envía una petición a una API externa de resultados-futbol.com con la cual se esté trabajando, la que devuelve la información enviando un archivo generalmente XML, este archivo es tratado por la aplicación, convirtiendo el contenido en información legible para el usuario que lo visualizará en la pantalla de su dispositivo.

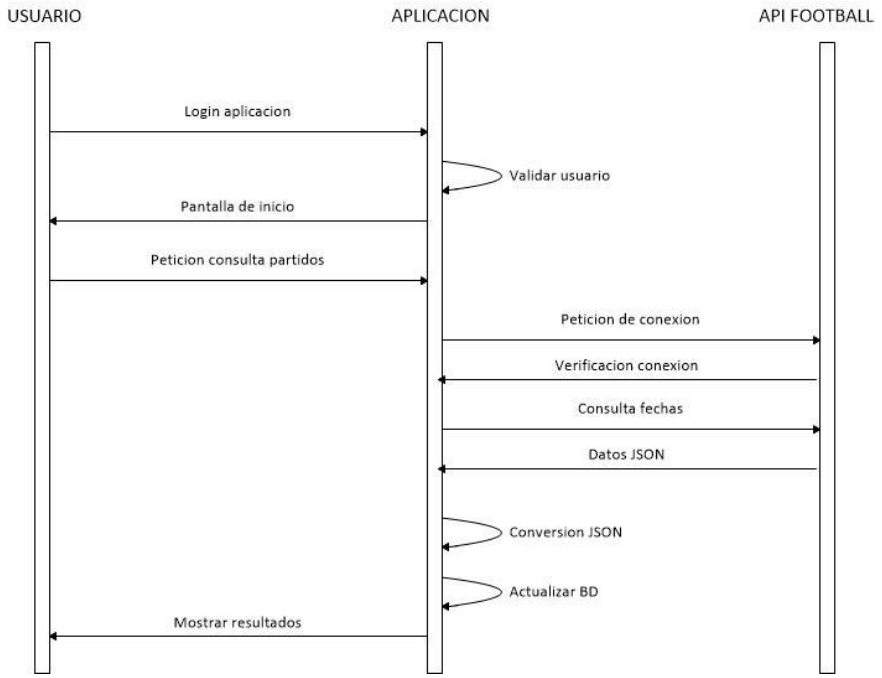


Fig. 25 Diagrama de secuencias de consulta de partidos.

En el calendario mostrado al usuario, se presenta toda la información correspondiente a la temporada actual del campeonato, así como la posibilidad de poder realizar una apuesta a un partido concreto o a una fecha que no ha sido jugada aún. Si el usuario desea ingresar una apuesta se abre una pasarela de pago, enviando una petición a la API con los datos necesarios para que se confirme la transacción, una vez que el usuario proceda a confirmar el pago, esta se registra tanto en la API externa, como en la base de datos de la aplicación.

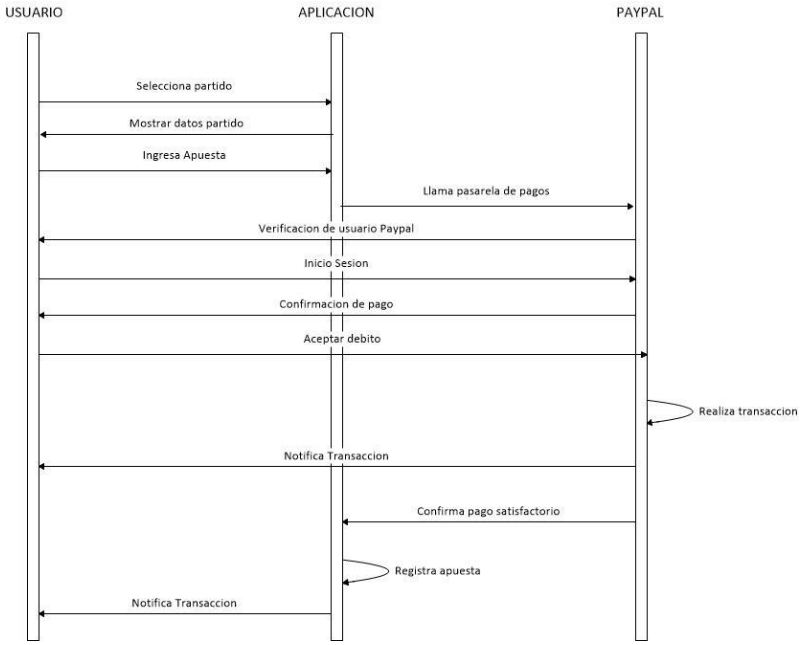


Fig. 26 Diagrama de secuencias de generación de apuestas

Para la ejecución del proceso de pago de apuestas, la aplicación envía peticiones programadas a la API, con el fin de mantener la información actualizada y comprobar la culminación de jornadas deportivas. El proceso de pago consulta la información de los partidos que han finalizado, comprueba los ganadores para cada encuentro y calcula el monto a pagar. El pago termina cuando la aplicación envía la transacción mediante la comunicación con la API, en este caso Paypal, y genera la orden de pago, con la cual, se envía el dinero a los ganadores y se actualizan las tablas de la base de datos de la aplicación.

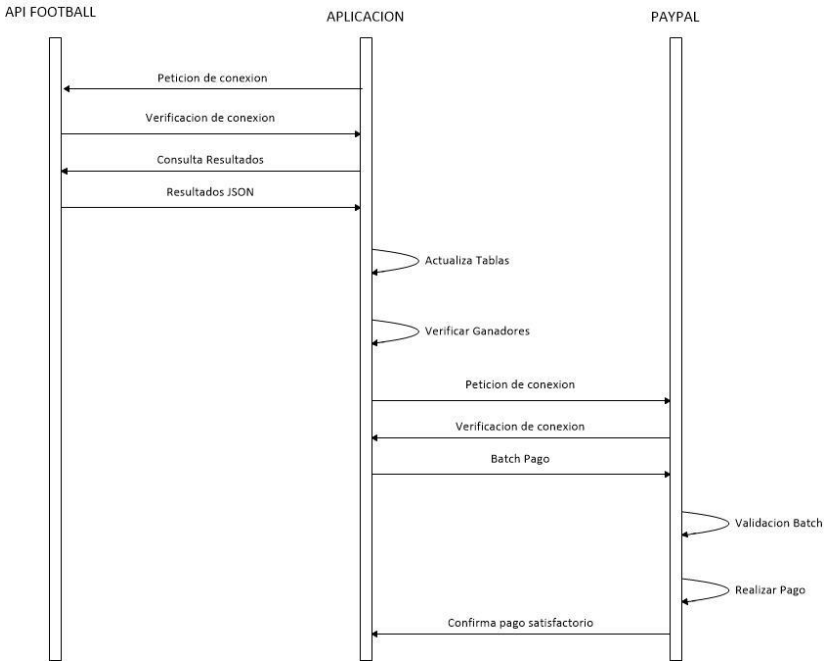


Fig. 27 Diagrama de secuencias de generación de pagos.

3.3 Diagrama de flujo de datos

La aplicación web híbrida presenta un diagrama de flujo de datos con 2 entidades externas el usuario que se registra para utilizar el sistema y las APIs que proporcionan la información para el almacén de datos.

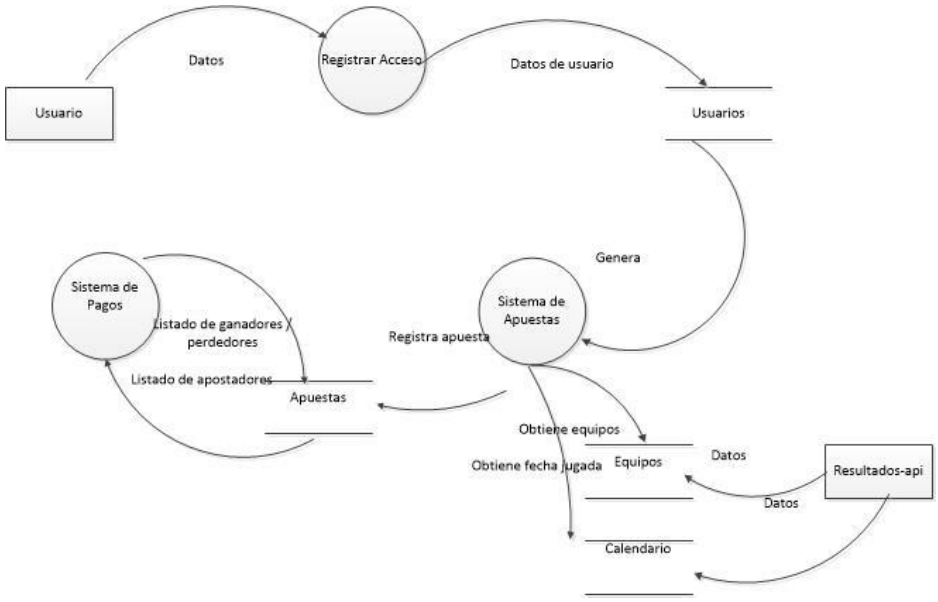


Fig. 28 Diagrama de flujo de la aplicación

3.4 Diseño de la base de datos

El diseño de la base de datos está estructurado para almacenar datos históricos y de esta manera eliminamos las peticiones de datos no necesarias a las APIs, tal es el caso del calendario deportivo.

Tablas:

app_apuestas: Almacena toda la información de los usuarios y los equipos, el valor de la apuesta y a que equipos apostaron.

app_fechas: Esta tabla almacena las fechas de todo el calendario deportivo del año actual y también indica la fecha actual.

app_partido: En esta tabla almacenamos todos los datos del calendario deportivo, incluyendo los partidos que ya fueron efectuados y los pendientes. Para la actualización de esta tabla existe un proceso en segundo plano que se encarga de alimentar de datos a esta tabla.

app_usuario: En esta tabla se almacenan los datos del usuario que se registraron al ingresar a la aplicación web.

app_equipo: Esta tabla almacena los equipos del calendario deportivo e indica con un estado los que pertenecen a la serie A.

app_metodo_pago: En esta tabla indica los métodos de pago que dispondrá la aplicación web para poder realizar el pago.

app_valor_apuesta: En esta tabla se almacena los valores, es decir, las cantidades que los usuarios podrán apostar.

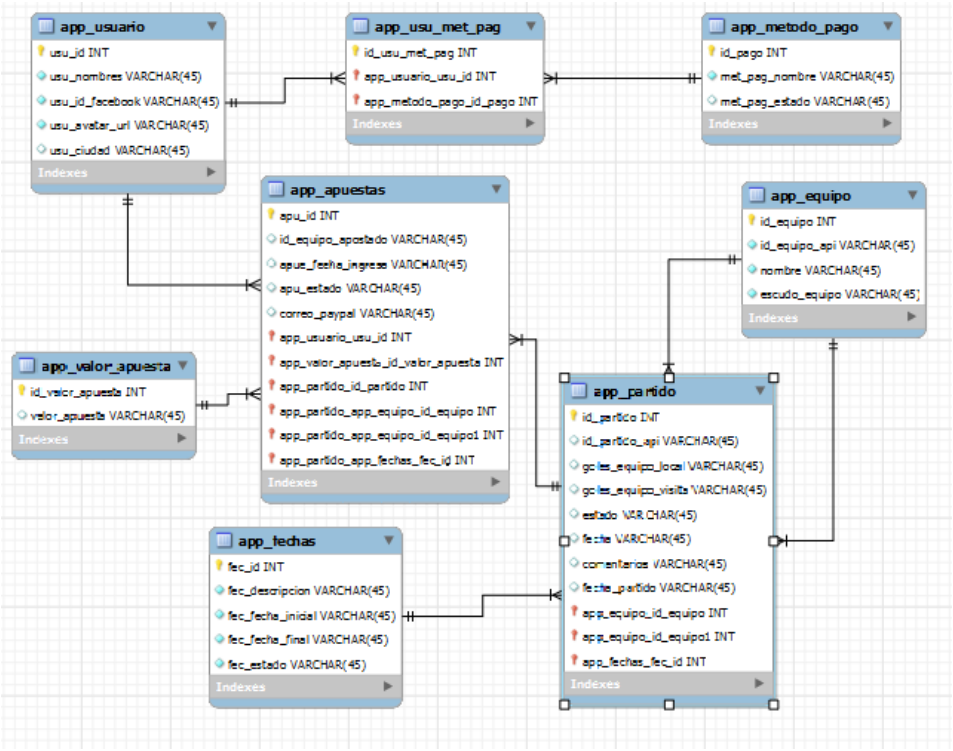


Fig. 29 Diseño de la base de datos de la aplicación web híbrida

3.5 Herramientas de programación

El conjunto y selección de herramientas que ayudan a las personas encargadas de la implementación de la aplicación, desempeña un papel importante, por lo que es importante tener conocimiento de estos productos que estarán involucrados directamente con el resultado final de la aplicación.

3.5.1 Gestor de base de datos

El lugar donde los datos de la aplicación estarán almacenados, debe ser estudiado, según los requerimientos y alcance del proyecto, los cuales tienen que ser tratados en la fase de análisis, la cual ayudará a la toma de decisiones, en general se recomienda tener en cuenta factores como: cantidad de usuarios activos, usuarios recurrentes, versión, documentación, número de registros máximos que permite almacenar, entre otras.

3.5.1.1 MySQL

MySQL es un sistema de administración de bases de datos relacionales rápido, sólido y flexible ideal para crear bases de datos con acceso desde páginas web dinámicas, para la creación de sistemas de transacciones on-line o para cualquier otra solución profesional que implique almacenar datos, teniendo la posibilidad de realizar múltiples y rápidas consultas. (Cobo, Gómez, & Pérez, 2007).

MySQL presenta las siguientes ventajas:

- Tiene una licencia pública.

- Utiliza el lenguaje SQL (Lenguaje de Consulta Estructurado) que es el lenguaje de consulta estandarizado de bases de datos relacionales.

- Es un sistema cliente/servidor, permitiendo trabajar como servidor multiusuario y cada vez que se establece una conexión con el servidor, el programa servidor crea un subproceso para manejar la solicitud del cliente, controlando el acceso simultáneo de un gran número de usuarios a los datos y asegurando el acceso solo a usuarios autorizados.

- Puede ser llevado a cualquier plataforma informática. MySQL está disponible en varios sistemas operativos incluyendo las distribuciones más usadas de Linux, sistema operativo Mac X, UNIX y Microsoft Windows.

3.5.1.2 MySQL Workbench

Esta herramienta es ideal para el modelado de la base datos (diagrama E/R), posee una serie de opciones como administración de la base de datos, monitoreo, exportación /importación, administrador de cuentas de usuario, etc. Además, permite la

administración de diferentes instancias de la base de datos localizados, en cualquier servidor.

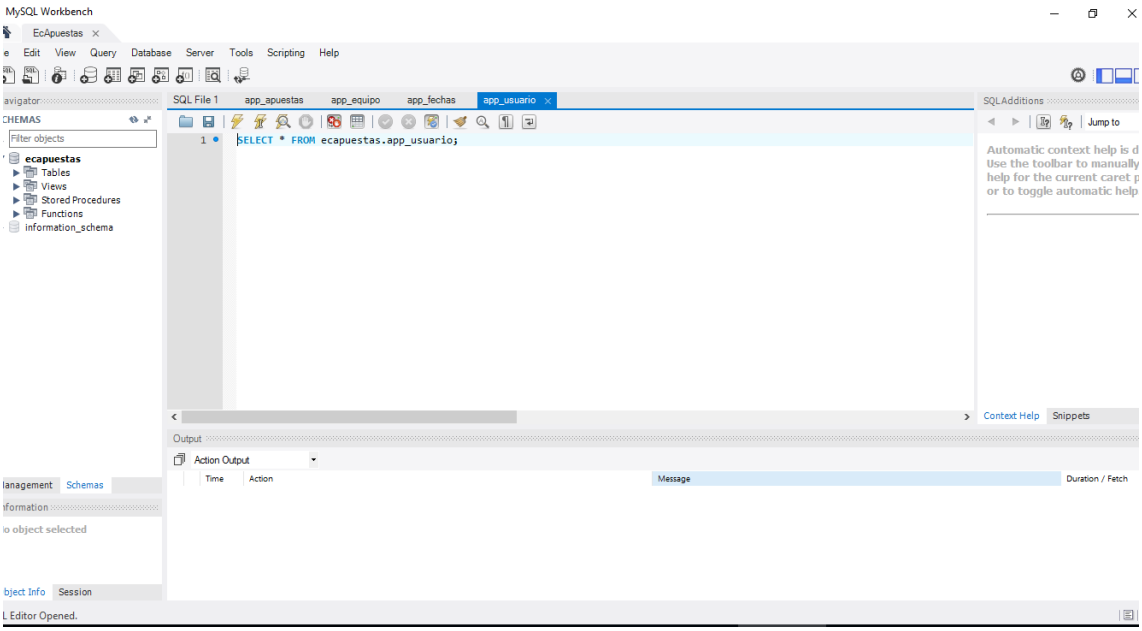


Fig. 30 Interfaz MySQL Workbench

3.5.2 IDE

Los entornos de desarrollo de software o entorno integrado de desarrollo es una combinación de herramientas que automatizan una gran cantidad de tareas o procesos de desarrollo. El objetivo principal de los entornos de desarrollo es simplificar la tarea de desarrollo de software. Un IDE ofrece uno o varios lenguajes de programación, para ello debemos definir qué lenguaje de programación vamos a utilizar e identificar que IDE nos ofrece.

3.5.3 NetBeans

Es un entorno profesional de desarrollo integrado libre que fue creado con la ayuda de Sun Microsystem adquirida ahora por Oracle Corporation. Provee soporte para la creación de aplicaciones Web con PHP 5 con depurador integrado y soporte para AJAX.

- Integración con Chrome
- Integración Webkit Browser

- Nuevo Editor JavaScript y depurador
- Consumo Servicio Web RESTful
- Versionamiento
- Integración con My SQL

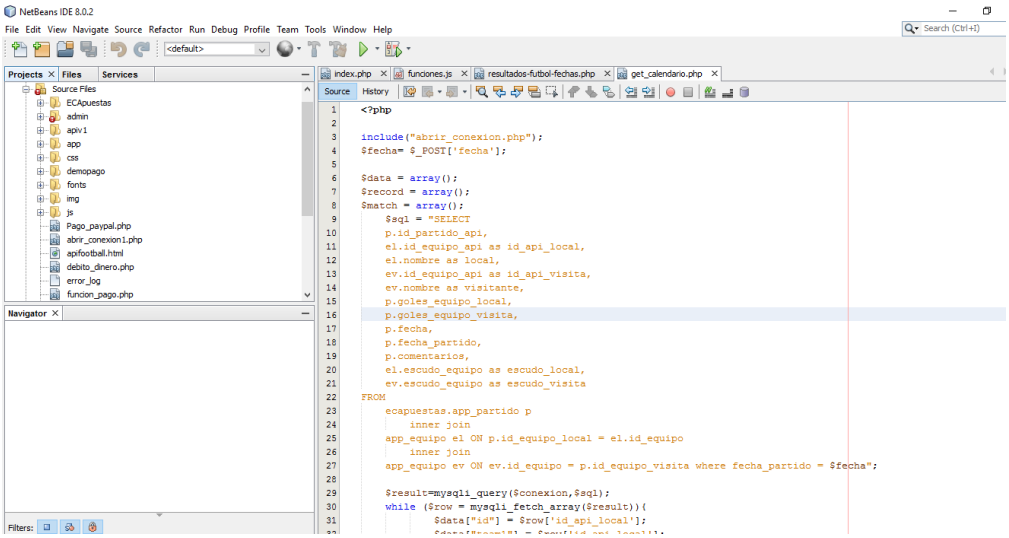


Fig. 31 Interfaz de Desarrollo NetBeans

3.5.4 Servidor Web

Un servidor web es un software que implementa el protocolo HTTP (HyperText Transfer Protocol); este protocolo está diseñado para transferir paginas HTML. Los servidores web se están ejecutando todo el tiempo y atienden las peticiones de los clientes que realizan desde los navegadores. (Camazon, 2014)

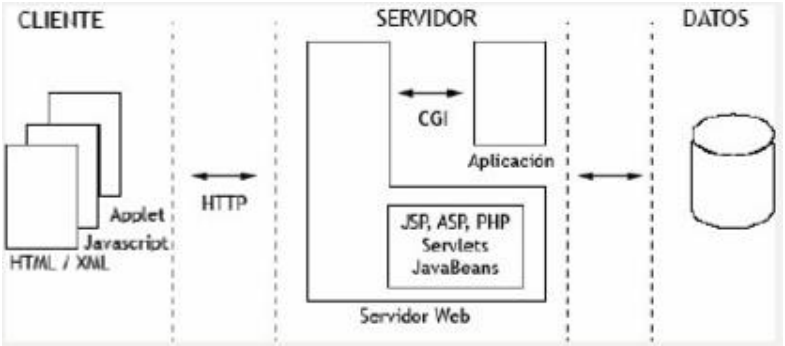


Fig. 32 Esquema de funcionamiento de un servidor web.

(Talón, 2012)

3.5.4.1 Servidor Web Apache

El Proyecto Apache HTTP Server es un esfuerzo de desarrollo de software de colaboración cuyo objetivo es crear un sistema robusto, de grado comercial, y de libre disposición del código fuente de un servidor HTTP (web). (Apache HTTP Server Poject, 2015)

3.6 Herramientas a ocupar en el desarrollo

El desarrollo de la aplicación web híbrida implica que previamente debemos seleccionar los componentes que convivirán con nuestro sistema, así como la preparación de las herramientas necesarias para la codificación, el almacenamiento de información y un servidor web el que aloje nuestro sitio. En nuestro caso hemos seleccionado las siguientes herramientas: NetBeans IDE, MySQL, MySQL Workbench, Hosting/Linux con Servicio Web Apache.

3.7 Análisis y selección de componentes

El análisis y selección de componentes es un proceso significativo mediante el cual obtendremos los datos que formarán el eje principal de la aplicación, para esto haremos una selección de componentes gratuitos y con licencia comercial que brinda ciertas funcionalidades como la posibilidad de realizar consultas más exactas, la cantidad de peticiones por hora, el formato de la estructura, y algunos puntos más.

Para buscar el componente utilizaremos un sitio web que tiene como objetivo ser un repositorio de APIs de algunas categorías.

El sitio web programmableweb.com contiene un repositorio bastante completo de APIs, esto facilita encontrar un componente con características específicas.

3.7.1 Análisis de componentes

3.7.1.1 DataFactory

Es una empresa establecida en Argentina de cobertura mundial, que ofrece contenidos en formato XML que cuenta con datos históricos e incidencias en vivo.

Su sitio web es http://www.datafactory.la.

Características:

- Partidos en tiempo real
- Incidencia de los partidos en tiempo real
- Tabla de posiciones
- Tabla de goleadores
- Ficha de jugadores por equipo

Arquitectura:

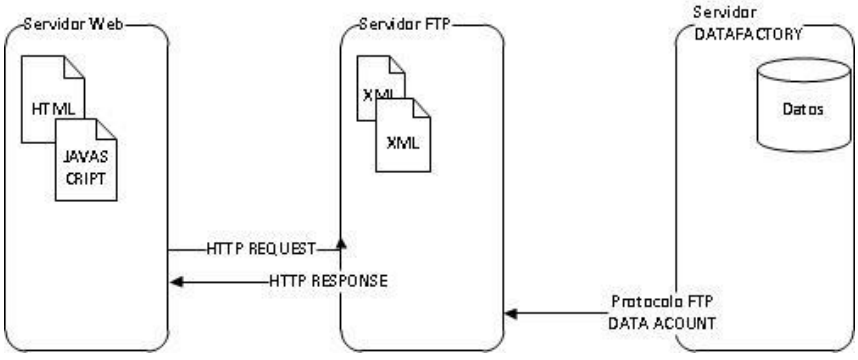


Fig. 33 . Esquema de funcionamiento del servicio DataFactory

Al momento de realizar una petición al servidor desde la aplicación, recibiremos un archivo XML, el cual contiene toda la información que requerimos del servicio web, para utilizarlo de manera conveniente.

Fig. 34 Formato XML entregado por DataFactory

3.7.1.2 Resultados Futbol

Posee una API's con información en tiempo real y estadísticas de futbol bastante completa. Posee una documentación bastante detallada y además cuenta con formato de intercambio de datos en JSON (JavaScript Object Notation) y XML.

3.7.1.3 Football – API

Es un servicio de datos de futbol en línea que incluye calendario, tabla de posiciones, comentarios con más de 300 competiciones. Actualmente su API's está documentada y se encuentra en la versión 2.0 con intercambio de datos en JSON (JavaScript Object Notation).

3.7.2 Selección de componentes

3.7.2.1 Implementación Paypal

Actualmente la mayoría de API´s poseen un entorno de prueba para desarrolladores, en donde podremos comprobar la integración de la Api con nuestra aplicación. Paypal no es la excepción y proporciona un **sandbox** que ayuda a desarrollar la pasarela de pagos, sin el temor de que al cometer errores pueda verse afectado dinero real. Debemos tener en cuenta que las cuentas creadas en Paypal no funcionan en el sandbox y viceversa.

3.7.2.1.1 Entorno de desarrollo Sandbox

Sandbox es el entorno para que los desarrolladores puedan hacer las pruebas pertinentes a sus aplicaciones correspondientes a la función de pagos sin la necesidad de utilizar transacciones de dinero real, pudiendo dentro de ella crear la cantidad de usuarios que el desarrollador considere necesario para evaluar los distintos escenarios que pudiese presentar una transacción real. Una vez satisfecho con los resultados y habiendo comprobado que la aplicación funcione como debería, el desarrollador puede poner en marcha lo desarrollado en el sandbox, haciendo que funcione en un entorno real, ocupando la url http://www.paypal.com/webscr.

El ingreso a esta plataforma de desarrollo se lo hace mediante la url https://developer.paypal.com/, en la cual podemos iniciar sesión si disponemos de una cuenta, o en su defecto proceder a crearla para utilizar las funcionalidades del sandbox.

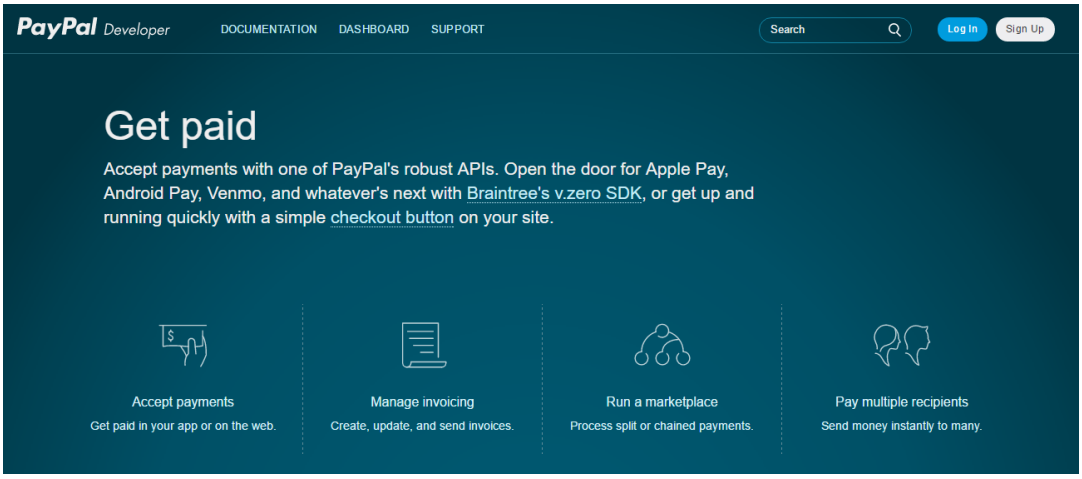


Fig. 35 Página de inicio del sandbox de Paypal

3.7.2.1.2 Creación de cuentas

Para poder utilizar el ambiente de desarrollo en nuestra aplicación debemos al menos crear dos cuentas con tipos diferentes, la que recibe el dinero, en este caso la casa de apuestas tiene que ser una cuenta Business y la cuenta o cuentas que actuarán como usuarios podrán ser una cuenta Personal o una Business.

Para crear cuentas se debe ingresar a la plataforma de desarrollo de Paypal, una vez dentro, seleccionar en el menú superior la opción Dashboard.

Fig. 36 Menú principal del sandbox de Paypal

Luego procedemos a seleccionar en el menú lateral izquierdo la opción Sandbox/Accounts.

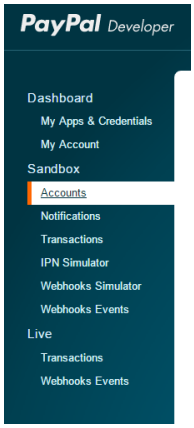


Fig. 37 Submenú lateral de la opción Dashboard

Una vez dentro de esa opción podremos ver que se despliega la lista de usuarios ya creados, por defecto estarán dos usuarios con los dos tipos de cuenta que necesitamos:

Business y Personal. Las cuales podemos ocuparlas si así lo queremos, nuestra recomendación es crear las cuentas ficticias desde cero para poder tener un control más exhaustivo de ellas y que sepamos con certeza porque fueron creadas.

La creación de una cuenta se lo hace pulsando el botón Create Account en la parte superior derecha de la pantalla de Cuentas

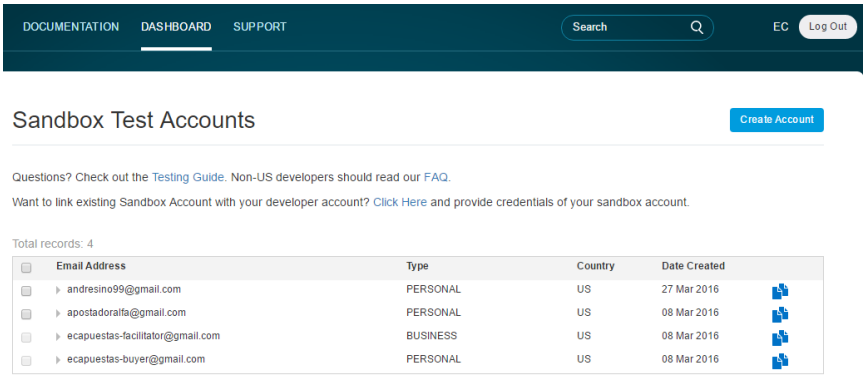


Fig. 38 Administrador de cuentas del sandbox Paypal

A continuación, se despliega un formulario con los campos necesarios para la nueva cuenta. Aquí solo debemos tener en consideración la información que seleccionaremos

al momento de elegir el tipo de cuenta (Business/Personal) y el monto de dinero que queremos darle a esa cuenta, pudiendo ser cero la cantidad, el resto de información puede ser ficticia completamente, ya que en el ambiente de desarrollador no se envían notificaciones a los correos, por lo que si vamos a colocar un correo ficticio debemos recordar bien la dirección que inventaremos y su contraseña, ya que estas serán las que nos permitan acceder a la información de cada una de ellas dentro del sandbox. Terminado de llenar la información necesaria, procedemos a crearla, si todo sea correcto, la nueva cuenta se visualizará en la pantalla de Cuentas.

3.7.2.1.3 Perfil de usuarios

Para acceder a la información de las cuentas creadas en el sandbox, debemos seleccionar la cuenta cuyo perfil deseamos ver el perfil y a continuación en el botón Profile.

Fig. 39 Menú del perfil de las cuentas

Se despliega una pantalla con varias opciones donde:

Profile: Proporciona la información que introdujimos al momento de la creación de la cuenta como el correo, el número telefónico, el tipo de cuenta, el país, también nos da la opción de poder cambiar la contraseña si es que la olvidamos o necesitamos hacerlo.

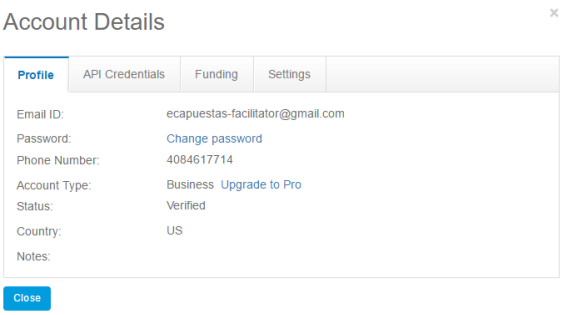


Fig. 40 Detalle general de la cuenta

API Credentials: Estos datos son necesarios para poder utilizar la API, integrándola en la aplicación. Más adelante hablaremos en detalle sobre cómo utilizarlos para poder comunicar la plataforma Paypal, ya sea en ambiente de desarrollo o real, con el producto software.

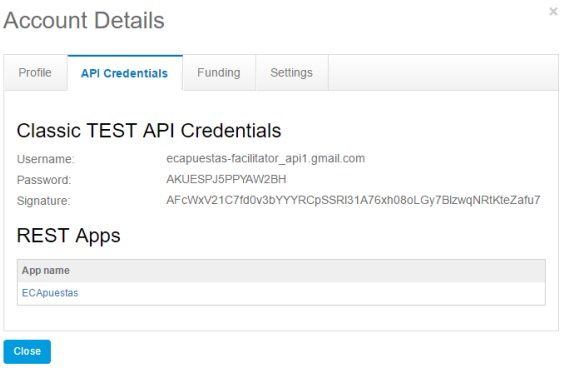


Fig. 41 Detalle de las credenciales de la cuenta

Funding: Muestra los fondos actuales de la cuenta, así como la información bancaria proporcionada al crearla.

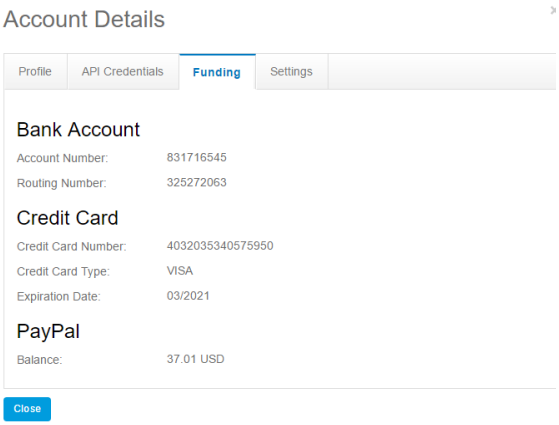


Fig. 42 Detalle de los fondos de la cuenta

Settings: Brinda la posibilidad de configurar la cuenta, con el fin de probar las transacciones antes de que se produzca el pago, y configurarla de modo que podamos recrear condiciones de error para un manejo oportuno de los mismos.

3.7.2.1.4 Creación API Paypal

Para la integración de PayPal en nuestra aplicación creamos un api que sirve de enlace entre ellas, proporcionando la comunicación, y los datos que serán enviados a la pasarela de pago paypal desde nuestra aplicación, para a continuación realizar las transacciones en tiempo real entre las cuentas negocio-cliente.

Para la creación de la Api vamos a la opción Dashboard/My Apps & Credentials y damos clic en el botón Create App.

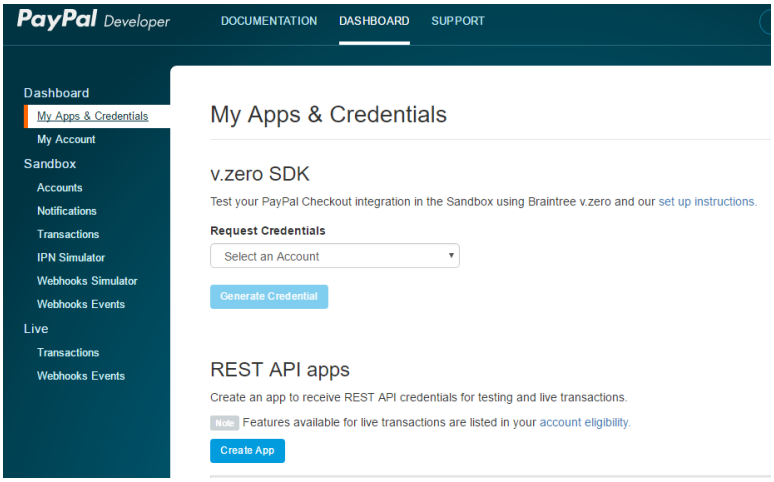


Fig. 43 Menú de credenciales de la API del sandbox de Paypal

En el formulario desplegado digitamos el nombre que deseamos darle al API, seguido de la cuenta a la que estará asociada. Las cuentas que pueden ser asociadas al API solo pueden ser Business ya que solo ellas poseen las configuraciones necesarias que un negocio requiere para controlar el pago de sus clientes.

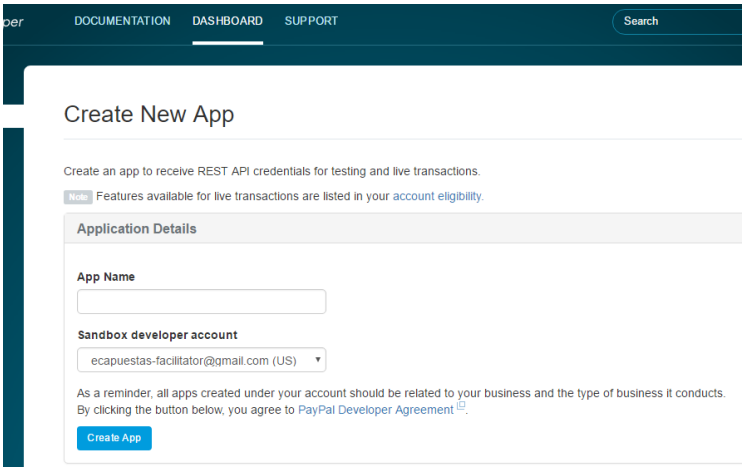


Fig. 44 Pantalla de creación de una API de pago

Una vez culminada con la creación de manera correcta, podremos verla en el listado de APIs que se tenga en esa cuenta de sandbox.

Fig. 45 Administrador de API´s creadas

3.7.2.1.5 Integración del api en la aplicación

La estructura básica que debemos colocar en nuestra aplicación para llamar a la pasarela de pago es la siguiente:

```
<html>
<head>
    <title>Ejemplo de pago mediante la API de PayPal</title>
</head>
<body>
    <!--<form action="https://www.paypal.com/cgi-bin/webscr" method="post"> -->
    <form action="https://www.sandbox.paypal.com/cgi-bin/webscr" method="post">
    <input type="hidden" name="cmd" value="_xclick">
    <input type="hidden" name="business" value="apostadornumerouno-facilitator@gmail.com">
    <input type="hidden" name="item_name" value="Emelec vs Barcelona">
    <input type="hidden" name="item_number" value="1">
    <input type="hidden" name="currency_code" value="USD">
    <input type="hidden" value="0" name="no_shipping"/>
    <input type="hidden" name="amount" value="99999">
    <input type="hidden" name="return" value="http://isbdevelopers.com/ecapuestas/gracias.html">  <!--Dond
    <input type="hidden" name="cancel_return" value="http://isbdevelopers.com/ecapuestas/app/"> <!--Donde
    <input type="hidden" name="notify_url" value="http://isbdevelopers.com/ecapuestas/debito_dinero.php">
    <input type="submit" value="Realizar pagos">
    </form>
</body>
</html>
```

Fig. 46 Archivo index.php que llama a la pasarela de Paypal

En donde, en el parámetro action del método form colocamos la url ya sea del paypal o del sandbox, en este caso la conexión es realizada al sandbox.

Los diferentes elementos corresponden al detalle de la transacción en donde:

Business: Es el correo de negocio que recibe las transacciones de pago de los clientes.
Item_name: Descripción del ítem a vender.
Item_number: Código del ítem a vender.
Currency_code: Tipo de moneda en la que se efectuará la transacción.
No_Shipping: Valor que se puede cobrarse en caso de haber envío.
Amount: Valor del ítem a vender.

Para conocer más acerca de que parámetros podemos utilizar y para qué sirven cada uno de ellos, se puede acceder a https://www.paypalobjects.com/webstatic/es_ES/developer/docs/pdf/pasarelaintegral_es.pdf

En cada transacción debe existir, las url donde debe ir al momento de confirmar la transacción, de retorno en caso de cancelar el pago y donde se realizan las operaciones de pago, es decir, el lugar en el cual se envía el formulario con los datos del pago y se realiza la petición al api de paypal para que efectué los movimientos de dinero y envíe las notificaciones del estado de la transacción. Esta última debe contener el siguiente código.

```php
<?php
$paypal_account = "apostadornumerouno-facilitator@gmail.com"; //Mi cuenta de paypal
$paypal_currency = "USD"; //La moneda con la que estamos trabajando
/*
En la siguiente linea formaremos el query para mandar al servidor de paypal y verificar el pago.
*/
$req = 'cmd=_notify-validate';
foreach ($_POST as $key => $value) {
     $value = urlencode(stripslashes($value));
     $req .= "&$key=$value";
}
$test = 'si'; //Si estamos usando el sandbox, lo cambiamos a "si", de lo contrario lo mantendras en "no"
if($test == 'si'){
    $url="https://www.sandbox.paypal.com/cgi-bin/webscr";
}else{
    $url="https://www.paypal.com/cgi-bin/webscr";
}

$item_name = $_POST['item_name']; //El nombre de nuestro artículo o producto.
$order_id = $_POST['item_number']; //El numero o ID de nuestro producto o invoice.
$payment_status = $_POST['payment_status']; //El estado del pago
$amount = $_POST['mc_gross']; //El monto total pagado
$payment_currency = $_POST['mc_currency']; //La moneda con que se ha hecho el pago
$transaction_id = $_POST['txn_id']; //EL ID o Código de transacción.
$receiver_email = $_POST['receiver_email']; //La cuenta que ha recibido el pago.
$customer_email = $_POST['payer_email']; //La cuenta que ha enviado el pago.
// Aqui verificamos si la cuenta que ha recibido el pago es nuestra cuenta.
if($paypal_account != $receiver_email){
   exit;
}
$res=file_get_contents($url."?".$req);
if (strcmp (trim($res), "VERIFIED") == 0) {
```

Fig. 47 Archivo Pago.php para la confirmación de la transacción.

Si ejecutamos el archivo index.php y damos clic en el botón enviando el formulario, obtendremos la página correspondiente según los datos en el parámetro action, en nuestro caso el entorno de desarrollo de Paypal en donde el usuario tiene que iniciar

sesión para poder confirmar el pago. En nuestro caso podemos manejar con las cuentas de usuario creadas en sandbox.

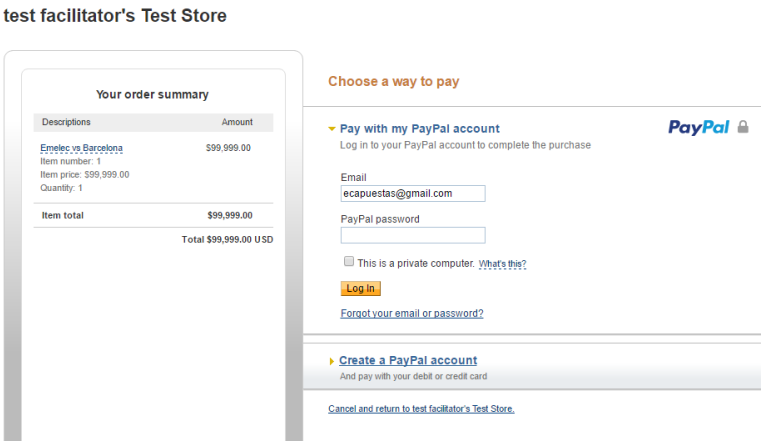


Fig. 48 Pasarela de pago del sandbox de Paypal

3.7.2.2 Implementación API de Facebook

Si bien, es posible mediante la utilización de plugins integrar contenidos sociales en nuestra web sin haber requerido la creación de una aplicación en Facebook, la mayoría de los casos y para lograr una mayor integración, es necesario tener una aplicación que de soporte a todas las peticiones que la aplicación va a realizar.

3.7.2.2.1 Entorno de desarrollo Facebook developer

Para utilizar la plataforma de desarrolladores, primero, debemos ser usuarios de Facebook, una vez creada una cuenta, debemos darnos de alta como desarrollador para poder crear una aplicación que ayude a administrar la comunicación y el envió de datos proporcionados por Facebook ante las peticiones que solicite la aplicación.

El ingreso a esta plataforma de desarrollo se lo hace mediante la url https://developers.facebook.com/ , en la cual podemos ingresar, previamente tras iniciada la sesión con nuestra cuenta de Facebook.

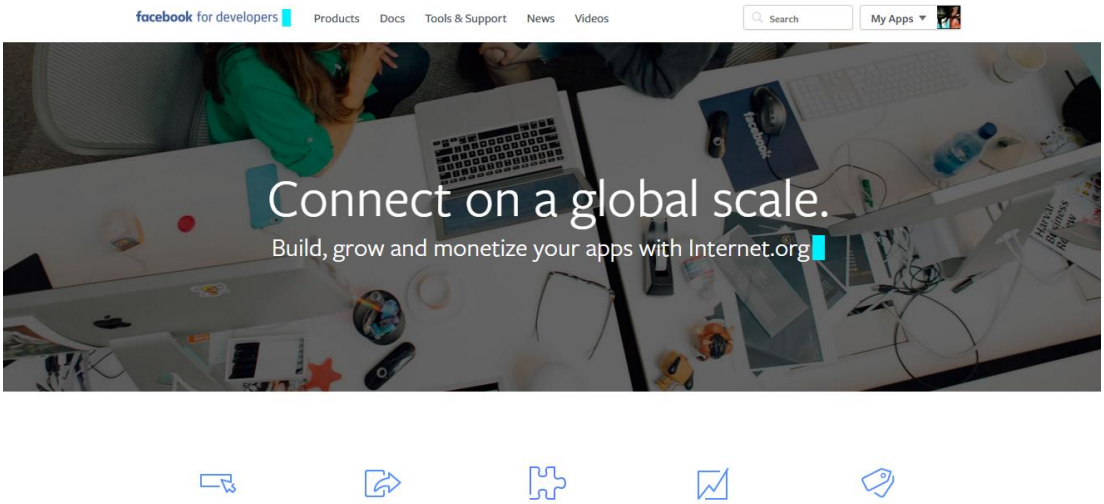


Fig. 49 Página de inicio de Facebook developer

3.7.2.2.2 Administrador de API´s de Facebook

Facebook presenta una opción en el menú principal de la página, donde se encuentra un apartado que dice “My Apps” en la parte superior derecha que nos permite administrar las API´s que se crean en el entorno de desarrollo.

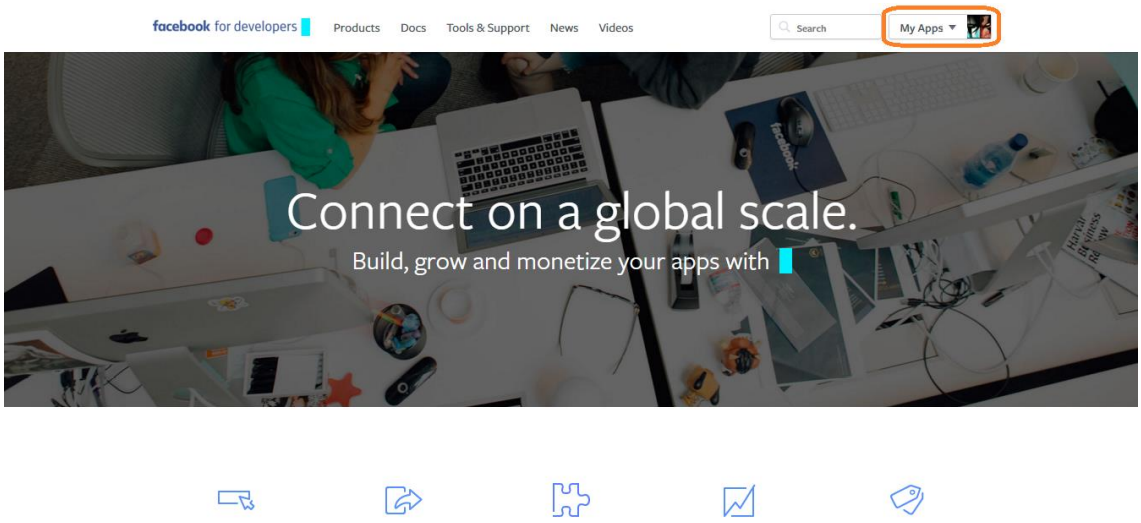


Fig. 50 Menú de administración de Apps

Una vez ingresado en la opción, se muestra el listado de aplicaciones que ese usuario de Facebook tenga creadas en el entorno para desarrolladores.

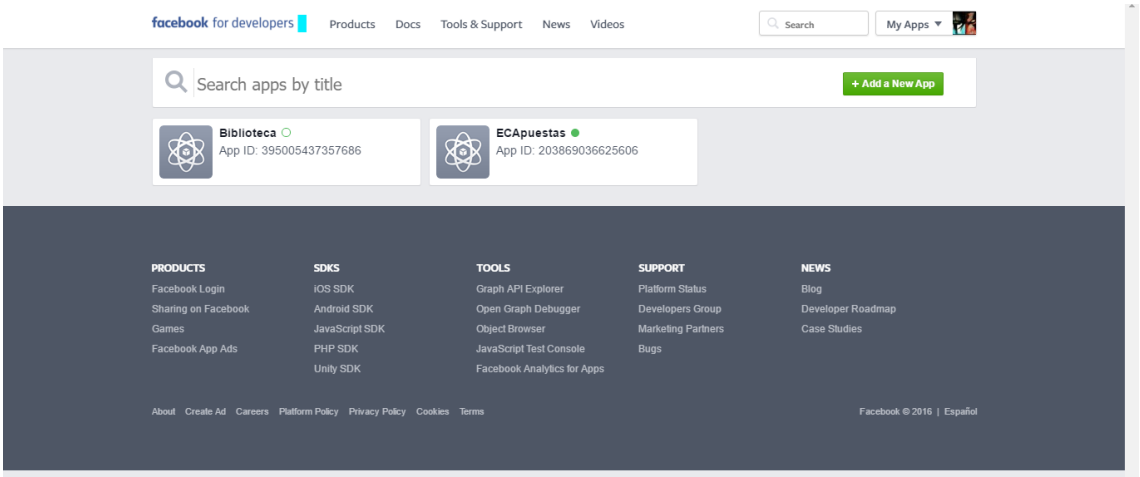


Fig. 51 Pantalla de creación y listado de Apps

3.7.2.2.3 Creación de una API de Facebook

Para trabajar con el API's de Facebook, debemos acceder en la opción "+ Add New App" en la parte superior derecha. Para proceder a llenar los datos necesarios que da como consecuencia, poseer una aplicación que podemos integrar en nuestro producto software.

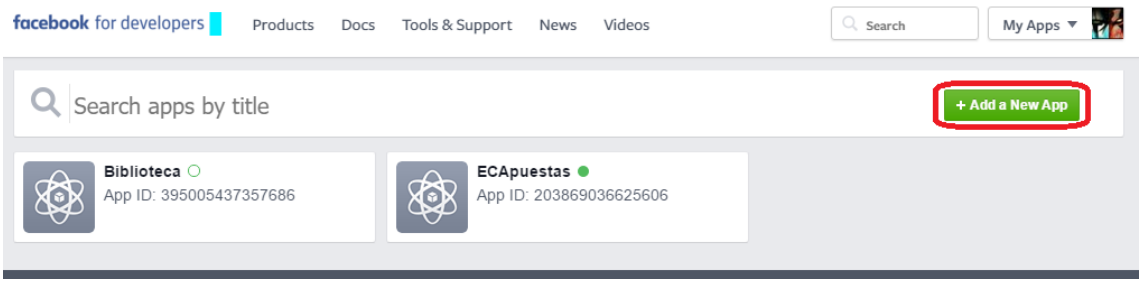


Fig. 52 Botón para crear nuevas App

Al crear una nueva aplicación en el entorno para desarrolladores, se muestra distintas plataformas en las que la API que va a ser creada puede ser integrada. Para el fin de la aplicación web híbrida en desarrollo, se seleccionará la opción para sitios web.

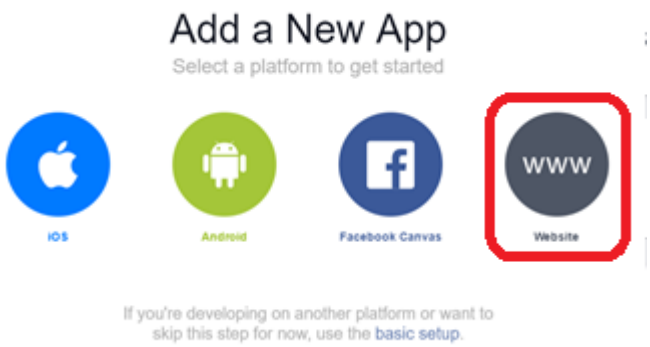


Fig. 53 Plataformas donde se integrará la app de Facebook

La selección de la opción WebSite, abre una pantalla en la que debemos ingresar el nombre de la nueva aplicación a crear.

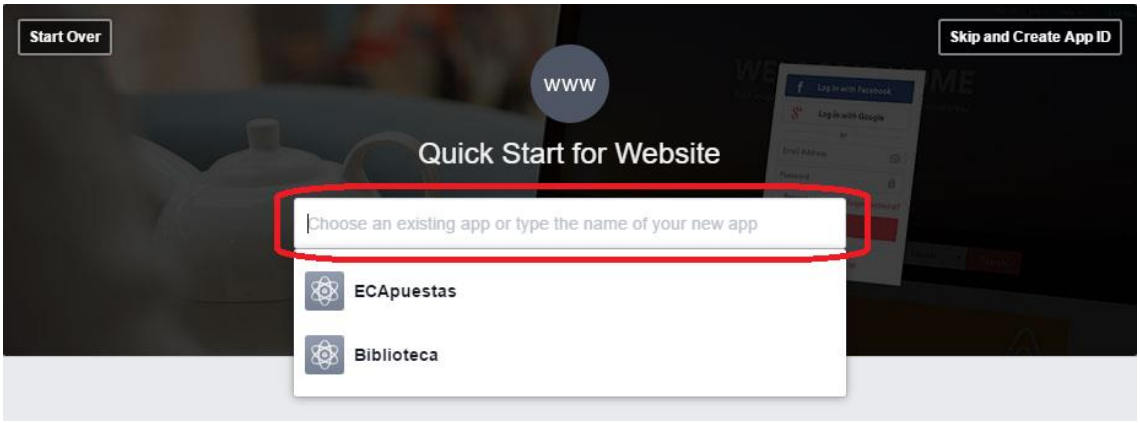


Fig. 54 Ingreso del nombre de la app

Una vez introducido el nombre correspondiente a la aplicación, que deseamos crear, debemos proceder a crearla pulsando “Create New Facebook App ID”.

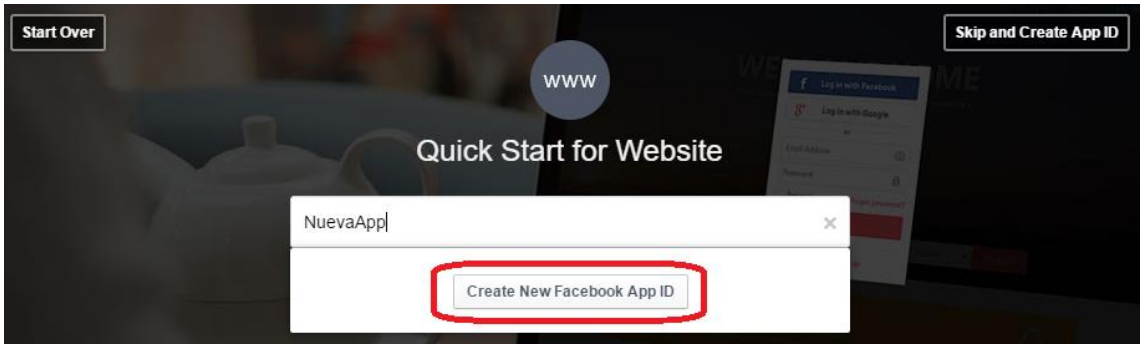


Fig. 54 Pantalla de creación de la app

Procedemos a llenar la información para completar la creación del API donde:

Create NombreApp App: Nos permite realizar pruebas de aplicaciones que tengamos creadas, realizando una copia de las configuraciones de la app, para posteriormente trabajar sobre la app clonada y configurarla según sea necesario.

Category, *Sub-Category:* Indicamos en que área se va a ocupar la aplicación creada y dependiendo de la seleccionada, puede pedir ingresar el sector de la rama donde integraremos la API. Estos datos son utilizados para generar estadísticas para Facebook de las áreas donde se utilizan sus apps.

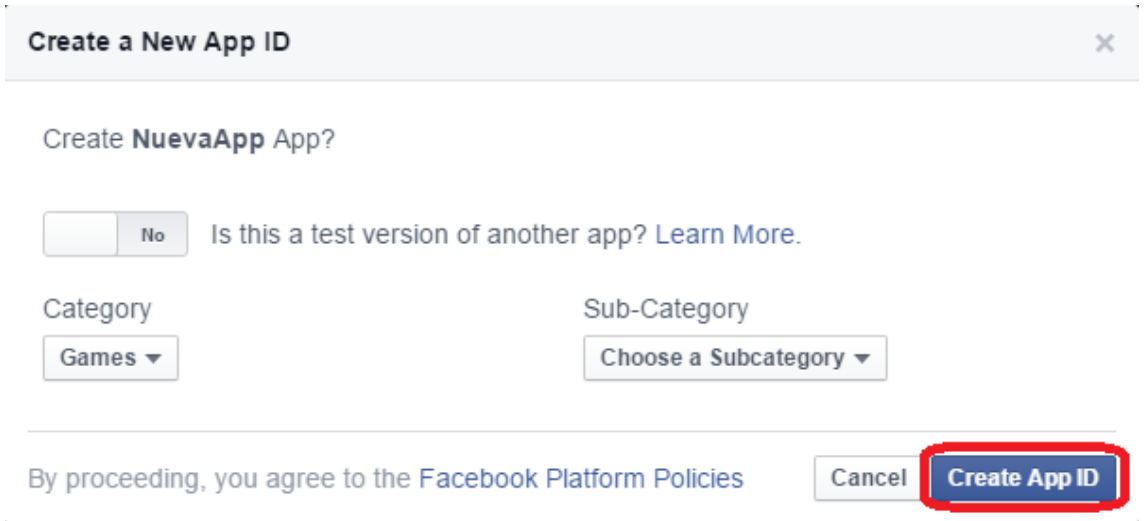


Fig. 55 Formulario de creación

3.7.2.2.4 Integración del api en la aplicación

Con la finalización del proceso de crear una nueva app de Facebook de manera satisfactoria, obtendremos un menú para dar las últimas configuraciones las cuales ayudarán en el proceso de integración en el que:

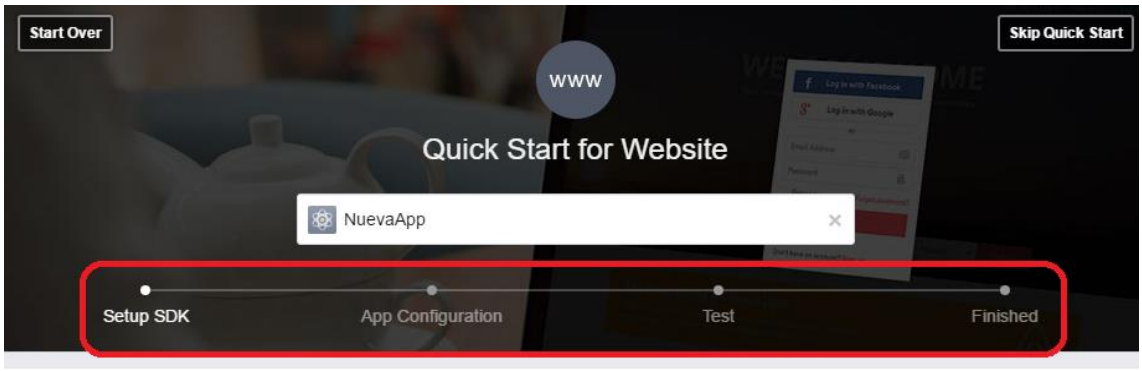


Fig. 56 . Proceso de creación

Setup SDK, Proporciona el código necesario para implementarlo en la aplicación, el cual, se puede incrustar en el sitio donde se desea invocar la app creada.

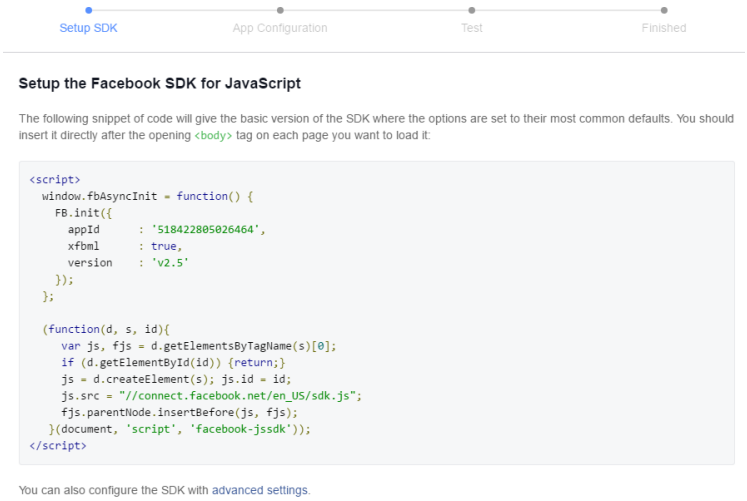


Fig. 57 Proceso de creación Setup SDK

App Configuration. Se tiene que especificar la URL del servidor, donde la app creada se va a ejecutar.

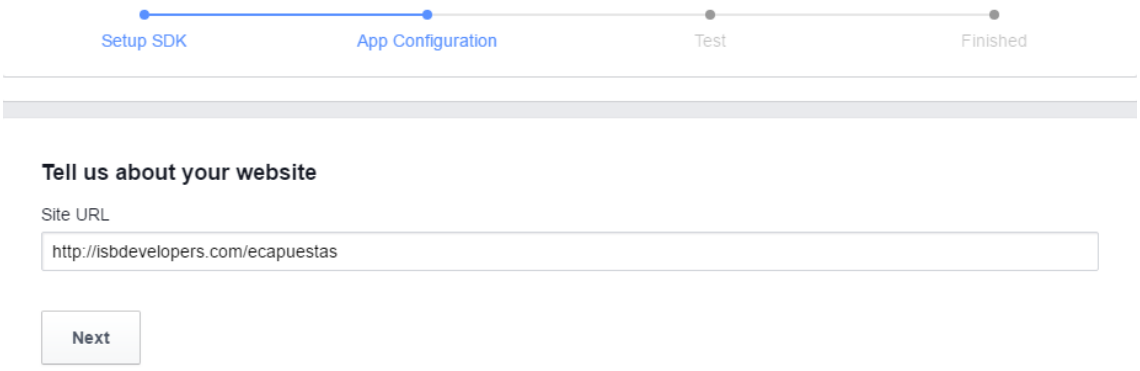


Fig. 58 Proceso de creación App Configuration

Test. Si la conexión a la URL proporcionada es satisfactoria, obtendremos una porción de código que nos permite añadir en la aplicación la funcionalidad de "Me gusta".

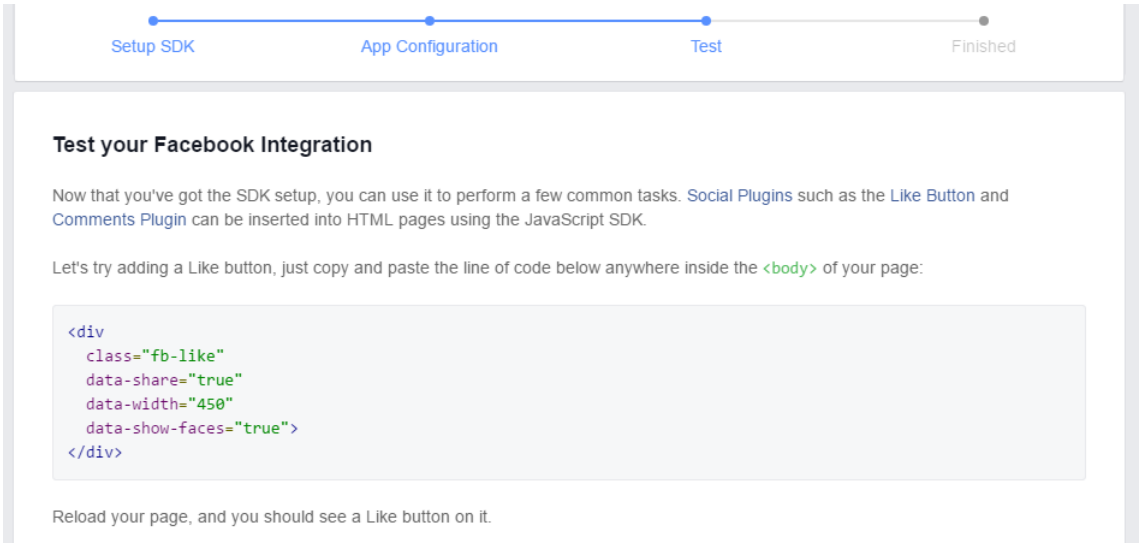


Fig. 59 Proceso de creación Test

Finished. Indica que la app creada está integrada en el servidor de la aplicación y puede ser utilizada. Para finalizar el proceso se debe seguir los enlaces de "Skip to Developer Dashboard" que nos ayuda a la gestión de nuestra API o "Documentation" que proporciona la documentación necesaria para los desarrolladores de la aplicación de Facebook.

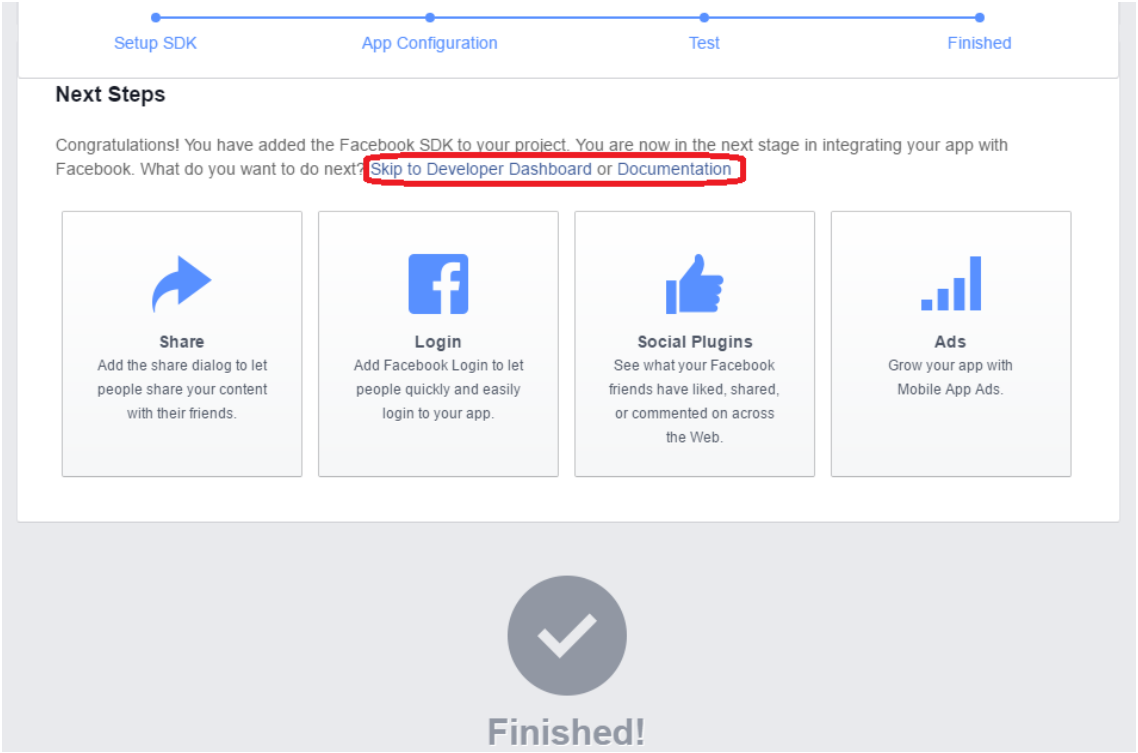


Fig. 60 Proceso de final creación

3.7.2.2.5 Perfil de la App

Al ingresar al dashboard de la app creada, podemos visualizar que la misma no se encuentra disponible para los usuarios, sino que solo está en modo para que los desarrolladores puedan trabajar. También en esta ventana se visualiza la versión de la API, y las credenciales (ID, Secret) necesarias para la implementación dentro de la aplicación.

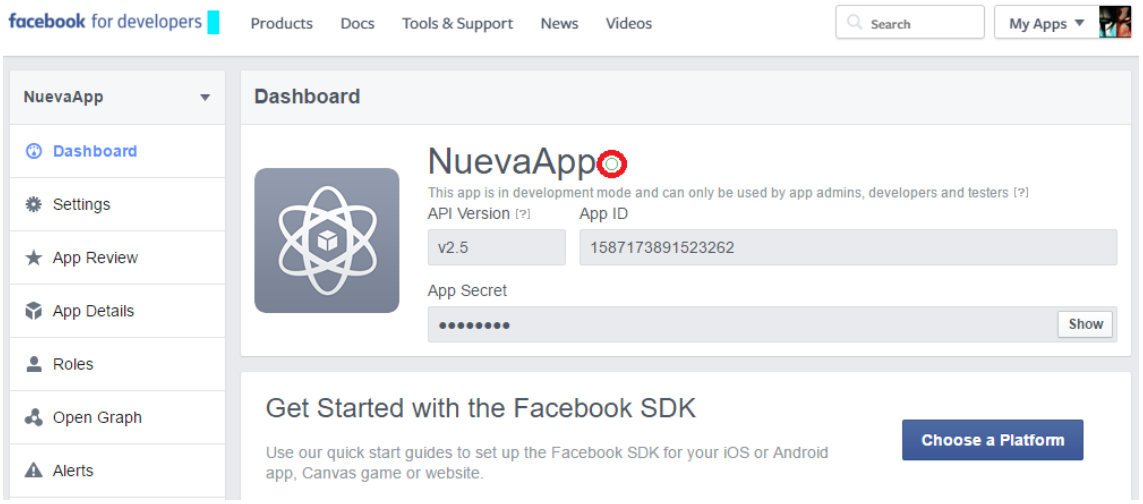


Fig. 61 Menú de administración de la app

3.7.2.2.6 Publicar una App

Para cambiar el estado de la app a disponible, debemos acceder a la opción "AppReview" del menú lateral del dashboard de la aplicación. La opción "Make NuevaApp public?", puede ser cambiada a: "Yes" para publicar la app y "No" para dejar de publicarla.

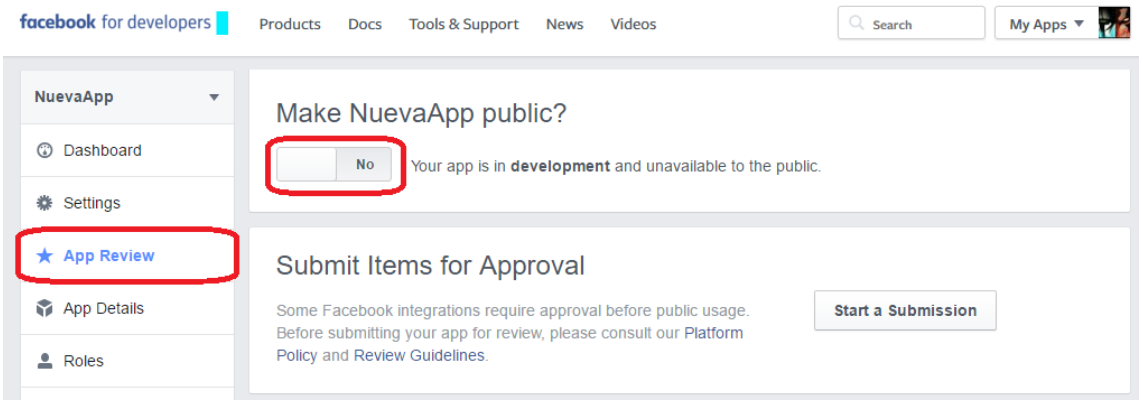


Fig. 62 Menú de revisión de estado de la app

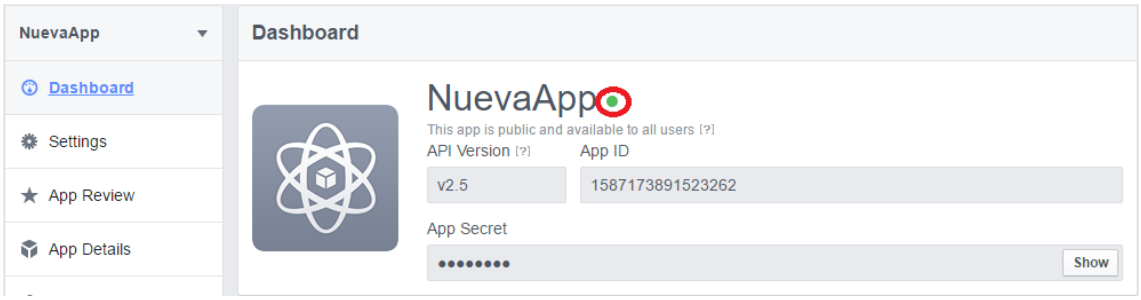


Fig. 63 Revisión del estado de la App

3.7.2.3 Implementación API Resultados Futbol

Los datos que son el motor de la aplicación web híbrida provienen de la API de resultados-futbol, a los que el producto software accede de manera gratuita a través del registro en http://www.resultados-futbol.com/ . Una vez completado el registro, resultados-futbol envía una notificación al correo electrónico asociado al momento de darse de alta en la URL especificada, con el token que nos permitirá hacer uso de la información del servicio web.

Fig. 64 Página de inicio de Resultados Futbol

Una vez dentro de la página iniciado sesión con nuestro usuario, podemos ver el token necesario para la comunicación de datos entre el servicio web y la aplicación, en la opción “API” del menú principal.

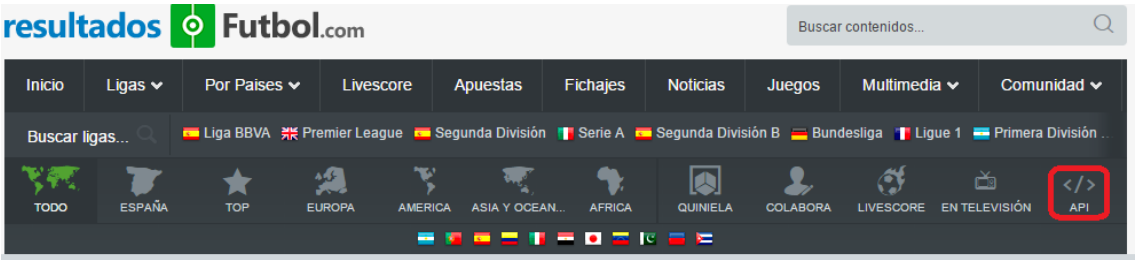


Fig. 65 Menú principal de Resultados Futbol

El menú mostrará una ventana con la información correspondiente a la cuenta, así como un menú, en el cual accedemos a la opción “Panel” que nos permite visualizar la key de la APIs para establecer la comunicación.

Fig. 66 Menú de usuario Resultados Futbol

Además de visualizar la key del api, también podemos ver la fecha en la que ingresamos al servicio, también el número de peticiones restantes que podemos realizar por día.

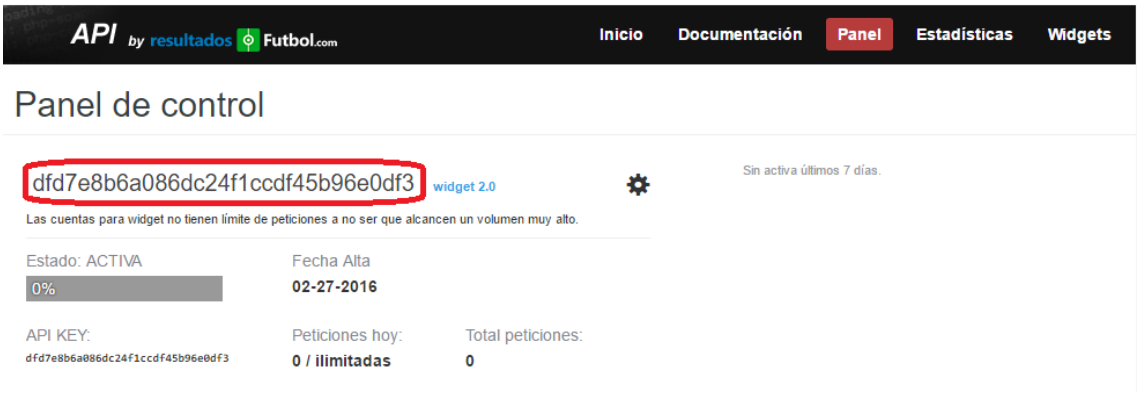


Fig. 67 . Panel de control de la cuenta

3.7.2.3.1 Integración del api en la aplicación

Para realizar peticiones desde la aplicación al servicio web, con el fin de obtener los datos de los campeonatos de futbol con los que cuenta este servicio, debemos disponer de manera obligatoria de una Key que establecerá la comunicación y permitirá a la aplicación híbrida obtener los datos de la API que considere necesarios.

A continuación, se muestra un ejemplo real de una petición que devuelve un listado con todas las ligas en formato JSON:

```
http://www.resultados-futbol.com/scripts/api/api.php?key=KEY_PROPORCIONADA&format=json&req=leagues
```

Fig. 68 Petición para la recuperación de información del servidor

Donde:

Key: Clave única que permite conectarse a la API.
Format: Los formatos disponibles en los que podemos recibir la información del servicio web son:

- XML.
- JSON.
- WIDGET.
- Ninguno. Si no se especifica este parámetro, la API envía un archivo plano.

- Req: El tipo de petición para obtener ligas, campeonatos, resultados, etc.

Para más información sobre los parámetros que podemos ocupar para la obtención de datos, se puede consultar http://www.resultados-futbol.com/api/documentacion

3.8 Diseño y creación de interfaces

Para la elaboración de las interfaces de este proyecto, se definieron 3 etapas, las cuales han servido de guía hasta el momento y de igual modo, hasta que el software salga de la fase de prototipo. Estas etapas fueron definidas de la siguiente manera: Bocetos, Implementación y Retroalimentación:

Etapa de bocetos. En esta etapa, se realizó una recolección de ideas, las cuales, tras acuerdo mutuo se fueron poco a poco cristalizando en figuras dibujadas primero en papel y luego llevadas a ordenador para que pueda ser visualizado de mejor manera. En esta etapa aún no se puede apreciar colores, letras, imágenes, etc. Más bien, sirve de ayuda para saber proporciones, donde va a estar ubicado cada elemento dentro de la página, etc.

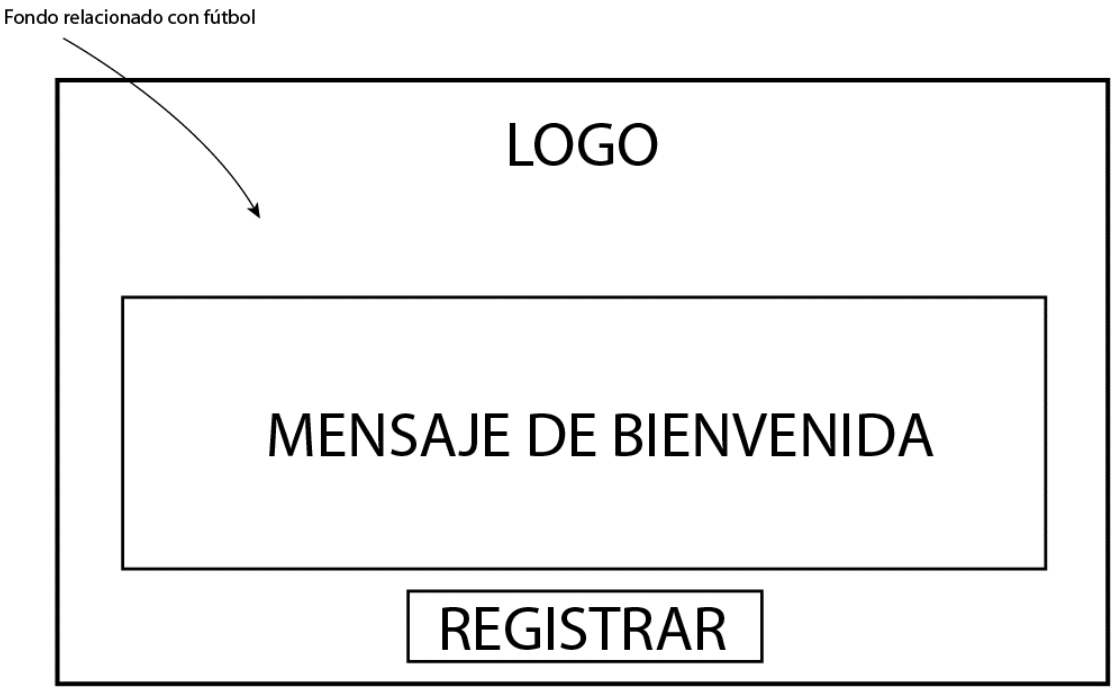


Fig. 69Boceto de la pantalla de registro

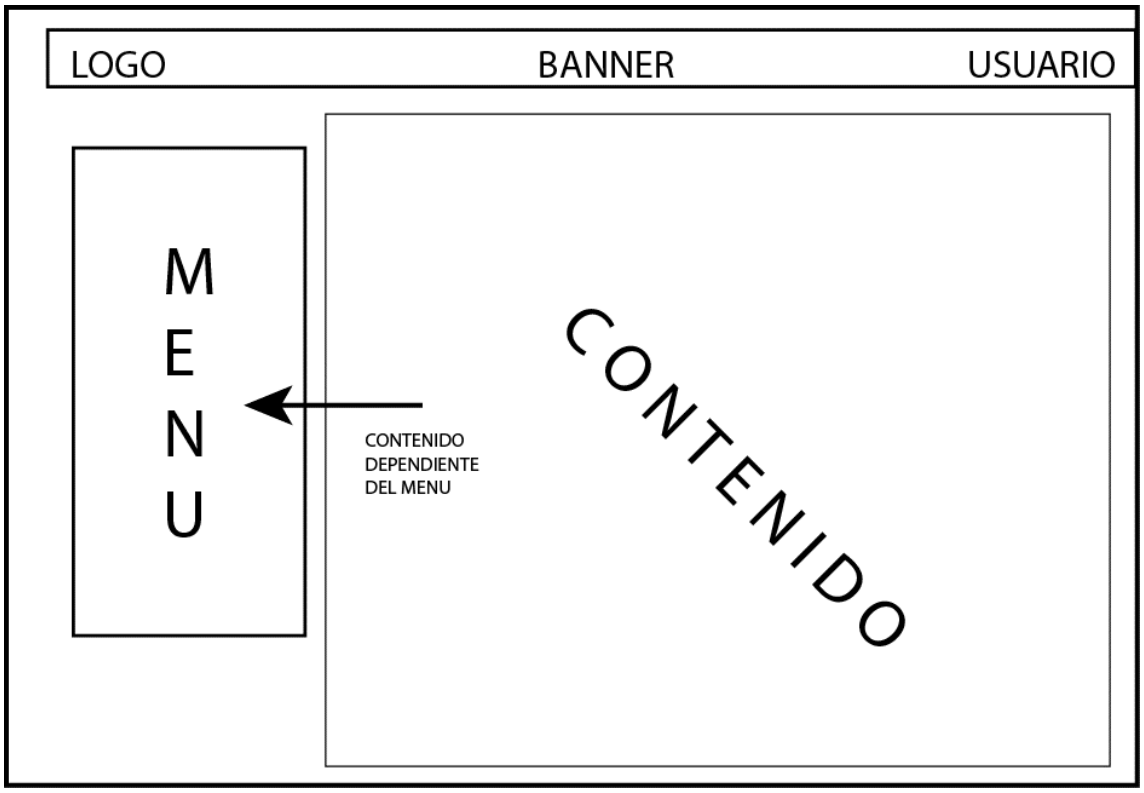


Fig. 70. Boceto de la página principal de la aplicación

Etapa de implementación. En esta fase, se trata de llevar los bocetos, a un trabajo más elaborado, decidiendo ya colores, textos, contenido, ítems del menú, etc. Cada uno de los bocetos ya detallados y aprobados son traducidos para el caso de este proyecto, con

la creación de las plantillas HTML y hojas de estilos. ECApuestas se encuentra actualmente en esta fase, al ser un prototipo se da más prioridad a la funcionalidad.

Fig. 71. Estado actual de la pantalla de registro

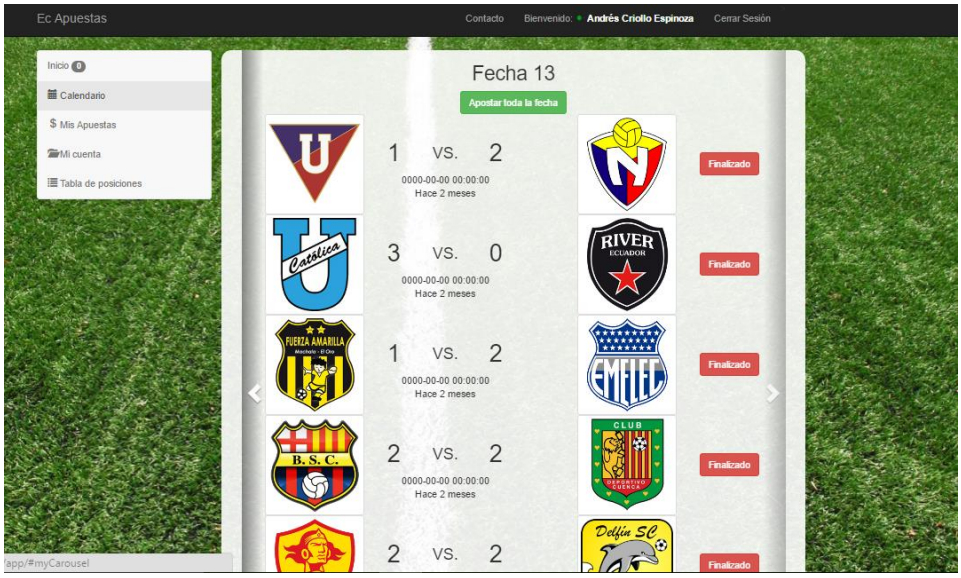


Fig. 72.Estado actual de la página principal de la aplicación.

Etapa de retroalimentación. En esta etapa se va puliendo los detalles de la presentación de la web, en caso de ser necesario se realizará las correcciones oportunas. Como hemos dicho ECApuestas se encuentra en la fase de prototipo, en la etapa de implementación, pero ya se ha pensado y acordado cambios que pueden ayudar a mejorar el sistema actual.

Fig. 73. Logotipos de ECApuestas.

Fig. 74. Nueva pantalla de la aplicación para invitados

Pantalla Registro

Fig. 75. Nueva pantalla de registro de la aplicación.

Fig. 76 Nueva pantalla principal de la aplicación.

3.9 Pruebas Unitarias

Las pruebas unitarias, ayudan a la verificación de funcionamiento, comunicación entre las diferentes API´s integradas con la web híbrida. Estas pruebas se recomiendan hacerlas para probar la correcta integración de los componentes de software que componen el sistema, y con más énfasis si los mismos son de terceras personas y/o heterogéneos, heredados.

En este trabajo utilizamos las API´s de Facebook, Paypal y de Resultados Futbol, que están incluidas para el registro y adquisición de datos del usuario, recepción y pago de apuestas y recolección en tiempo real de la información concerniente a los partidos del campeonato nacional ecuatoriano consecutivamente.

3.9.1 Pruebas unitarias de Facebook

Como se ha dicho con anterioridad en reiteradas ocasiones, esta API en la aplicación web híbrida es utilizada para el registro del usuario en EC Apuestas. Para ello es importante verificar que los datos obtenidos al momento de enviar una solicitud a Facebook, correspondan al usuario en cuestión y además que los datos proporcionen toda la información que la aplicación requiere para su funcionamiento.

Al entrar a la aplicación se ejecuta un fragmento de código que entrega la API de Facebook, el cual nos permite enlazarnos a un cuadro de diálogo en donde nos muestra iniciar la sesión o pedir acceso para obtener los datos.

```
FB.login(function(response) {
          if (response.authResponse) {
                    FB.api('/me', function(response) {
                              console.log(response);
                    });
          } else {
                    //console.log('User cancelled login or did not fully authorize.');
          }
          $("#loading").hide();
}, {scope: 'email,user_location,user_likes'});
```

Fig. 77 Fragmento de código javascript del SDK de Facebook.

Pasos para obtener autorización de Facebook.

1. Redirigir al usuario al cuadro de diálogo de autentificación
    a. Para autenticación del usuario: Si el usuario no ha iniciado sesión, se les pide que introduzcan sus credenciales.
    b. Autorización de aplicación: Después de que el usuario se autentica, se le pedirá al usuario que autorice la aplicación.

2. Una vez que se autorice la aplicación, se volverá a dirigir a la aplicación con un código de autorización llamado token.

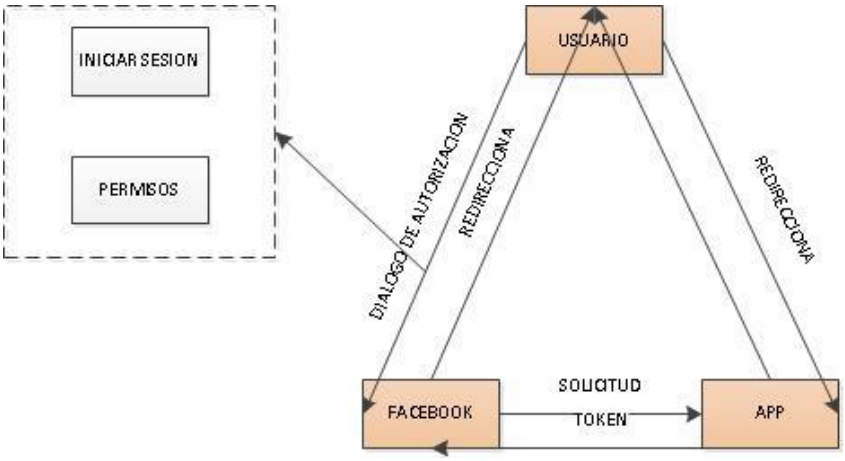


Fig. 78 Proceso de autorización a la API de Facebook

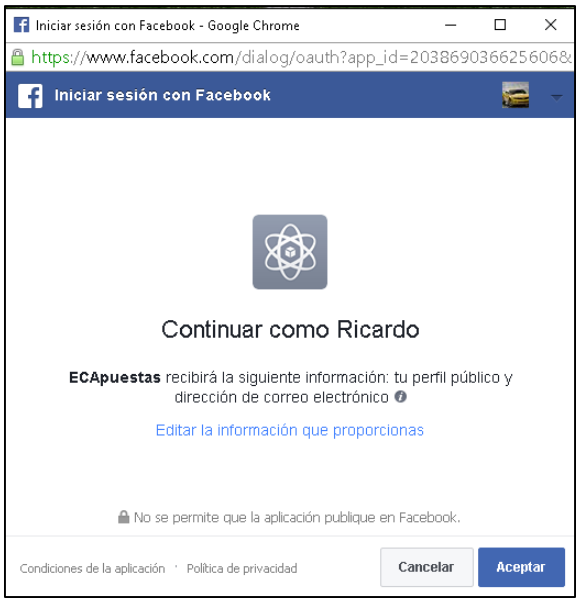


Fig. 79 Cuadro de dialogo de autorización de la API de Facebook.

El cuadro de dialogo de autorización de la API de Facebook se muestra solamente la primera vez que la aplicación solicita a Facebook permiso para acceder a los datos. Al aceptar las condiciones que indica el cuadro de dialogo, el SDK de la API de Facebook nos permite obtener los datos que hemos solicitado.

Fig. 80 Información obtenida de la API de Facebook en formato JSON.

.

Los datos que retornan son almacenados en la tabla app_usuario de la base de datos, en esta tabla se almacenan todos los usuarios que se registran en la aplicación.

```
SELECT *
FROM `app_usuario`
LIMIT 0 , 30
```

Perfilando [En línea] [Editar] [Explicar SQL] [C

Mostrar : Fila de inicio: 0 Número de filas: 30 Cabeceras cada 100 filas

Ordenar según la clave: Ninguna

+ Opciones

	usu_id	usu_nombres	usu_id_facebook	usu_avatar_url	usu_ciudad	app_usuariocol
Editar Copiar Borrar	1	AndrÃ©s Criollo Espinoza	735103949967590	NULL	NULL	NULL
Editar Copiar Borrar	2	Diego Vintimilla Espinoza	10207630875304110	NULL	NULL	NULL
Editar Copiar Borrar	3	Ricardo Espinoza	901182159995003	NULL	NULL	NULL
Editar Copiar Borrar	4	RubÃ©n Ortega LÃ³pez	449895951886103	NULL	NULL	NULL
Editar Copiar Borrar	5	AndrÃ©s SebastiÃ¡n Vintimilla	1679733348957606	NULL	NULL	NULL
Editar Copiar Borrar	6	Diany Monse Mendez	10153310345867282	NULL	NULL	NULL
Editar Copiar Borrar	7	Alejandro Vintimilla	1120429107996806	NULL	NULL	NULL

Fig. 81 Registros de la tabla app_usuario

3.9.2 Pruebas unitarias de Paypal

La aplicación ECApuestas realiza las transacciones monetarias mediante la utilización de la plataforma Paypal. Para comprobar su correcto desempeño es necesario probar su funcionamiento en los módulos donde se vea integrada la API, para nuestro caso en la recolección de dinero al momento que un usuario realiza una apuesta y el pago de las personas ganadores de dichas apuestas.

Para la creación de un pago mediante la plataforma de Paypal, se debe enviar una petición a la API con los datos que va a componer la transacción, esta solicitud se la realiza mediante la URL https://api.sandbox.paypal.com/v1/payments/payouts/, la cual debe contener una estructura similar a la de Fig. 84

Fig. 82 Estructura de solicitud de datos a la API Paypal.

En donde, se puede observar con claridad, los parámetros que deben ser enviados para la cabecera y el detalle referente del pago, los parámetros que definen esta estructura son:

- Cabecera
  - sender_batch_id: Corresponde al id del pago, generado de forma automática por la plataforma de Paypal, con el fin de poder mantener un control sobre todas las transacciones realizadas.
  - email_subject: Asunto que informa el motivo de la transacción al receptor.

- Detalle
  - recipient_type: Método utilizado por el cual se va a efectuar el pago, pudiendo ser: una cuenta de correo, una cuenta de Paypal, número telefónico.
  - note: Contenido del mensaje que será recibido en el cuerpo del correo.
  - Receiver: la dirección de la cuenta asociada al destinatario de la transacción, pudiendo ser un listado con varios beneficiarios.
  - Sender_item_id: id asociado al ítem o ítems involucrados en la transacción.
  - Amount: Este parámetro consta de dos valores, la cantidad en unidades monetarias y el tipo de moneda.

La petición enviada con esta estructura a la API de Paypal, genera como respuesta lo siguiente.

Response Object


```
{
    "batch_header": {
        "payout_batch_id": "74P8L5T26CBJS",
        "batch_status": "PENDING",
        "sender_batch_header": {
            "email_subject": "You have a payment",
            "sender_batch_id": "575b966896449"
        }
    },
    "links": [
        {
            "href": "https://api.sandbox.paypal.com/v1/payments/payouts/74P8L5T26CBJS",
            "rel": "self",
            "method": "GET"
        }
    ]
}
```


Fig. 83 Respuesta de la API Paypal con datos de la solicitud.

- Encabezado

  Los datos que nos retorna son los mismos que los que fueron enviados en la solicitud de la petición, a los que Paypal añade el parámetro:

  - Batch_status: Paypal al momento de recibir la solicitud, la coloca con el estado "PENDING", la cual cambia a "PAID OUT", cuando la transacción sea satisfactoria.

- Detalle

  Devuelve la URL donde podemos obtener en un futuro los datos de la transacción para procesos posteriores al pago.

  - Href: Url que contiene todos los datos de la transacción.
  - Rel: Abre la url en la misma ventana
  - Method: Método por el cual los datos son enviados.

3.9.3 Pruebas unitarias de Resultados Futbol

Esta API es el eje central de la web híbrida por lo que, la verificación de los datos obtenidos es esencial para garantizar la calidad de la aplicación. Para ello debemos ver como los datos son enviados por el servicio al momento de realizar una petición y ver que efectivamente estos datos puedan ser leídos, extrayendo de ellos la información que creamos pertinente.

Para obtener los partidos de una determinada fecha, debemos hacer una petición de datos a la API que brinda este servicio, en este caso es la API de resultados-futbol.com. Esta página tiene una amplia documentación ordenada y categorizada que utilizando ejemplos nos facilita la integración.

```
http://www.resultados-futbol.com/scripts/api/api.php?tz=Europe/Madrid&format=json&req=matchs
&key=ebef7538f246f0feb72bb2e071c2ed05&league=43&round=7&order=twin&twolegged=1&year=2016
```

Fig. 84 Consulta a la API de Resultados-futbol.com.

La API de resultados-futbol.com necesita de algunos parámetros para indicar que información estamos solicitando. En este ejemplo se hace una petición de datos de todos los partidos de la fecha 7 del campeonato ecuatoriano de futbol. Los parámetros requeridos son:

- Format: La estructura de datos a ser devuelta, puede ser JSON, XML. WIDGET
- Req: El tipo de petición que se requiere realizar, ligas, clasificación
- Key: Clave privada otorgada al activar la cuenta.
- League: La liga a consultar
    - 43: Ecuador
- Round: La fecha a consultar, este valor es número.

Postman es una aplicación que permite realizar peticiones a los servicios web, para este caso vamos a utilizar para hacer una petición al servidor.

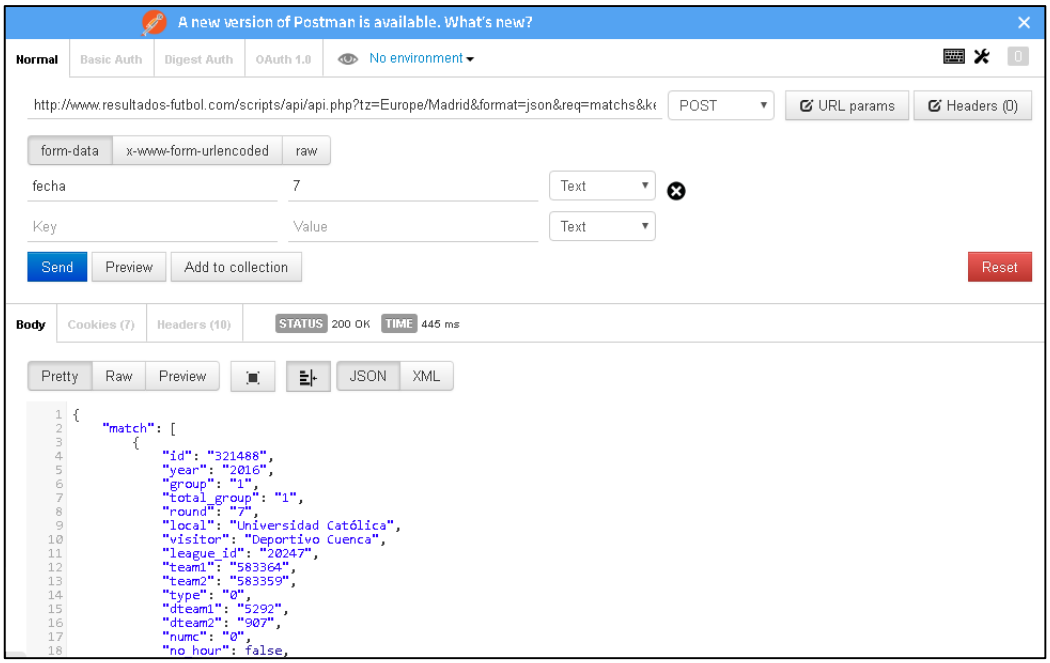


Fig. 85 Aplicación Postman Cliente Http

El resultado de la petición a la API de resultados-futbol.com es la estructura de datos en JSON, en este caso es el resultado de la fecha 7 del campeonato ecuatoriano de futbol.


```
{
    "match": [
        {
            "id": "321488",
            "year": "2016",
            "group": "1",
            "total_group": "1",
            "round": "7",
            "local": "Universidad Católica",
            "visitor": "Deportivo Cuenca",
            "league_id": "20247",
            "team1": "583364",
            "team2": "583359",
            "type": "0",
            "dteam1": "5292",
            "dteam2": "907",
            "numc": "0",
            "no_hour": false,
            "local_abbr": "UCE",
            "visitor_abbr": "CUE",
            "competition_name": "Serie A - Apertura Ecuador",
            "playoffs": false,
            "group_code": "1",
            "coef": "14.318",
            "local_shield": "http://thumb.resfu.com/img_data/escudos/medium/5292.jpg?size=60x&ext=png&lossy=1&1",
            "visitor_shield": "http://thumb.resfu.com/img_data/escudos/medium/907.jpg?size=60x&ext=png&lossy=1&1",
            "extraTxt": "Hace 3 meses",
            "schedule": "2016-03-12 18:00:00",
            "date": "2016/03/12",
            "hour": "18",
            "minute": "00",
            "local_goals": "2",
            "visitor_goals": "2",
            "result": "2-2",
            "live_minute": "",
            "status": 1,
            "channels": [],
            "winner": 0,
            "penaltis1": 0,
            "penaltis2": 0,
            "prorroga": false
        },
```


Fig. 86 Estructura de datos en JSON obtenidos del servicio web de resultados-futbol.com

3.10 Pruebas de interoperabilidad

3.10.1 Prueba de recepción de apuestas

Aquí se comprobará si al momento de realizar una apuesta utilizando el método de pago Paypal, efectivamente la transacción es satisfactoria, es decir, que la suma apostada se vea disminuida del usuario e incrementada en la cuenta de la aplicación EC Apuestas.

Para la visualización de las transacciones en cuentas de usuario y empresa, tal como fue indicado en el punto 3.7.2.1, para este prototipo se utiliza el entorno de desarrollo de Paypal (Sandbox), que nos facilita observar con claridad, si las transacciones se realizaron de forma satisfactoria en las cuentas involucradas.

3.10.1.1 Apuesta por partido

Para esta prueba vamos a realizar una apuesta a un partido cualquiera, con un monto de 6 dólares y verificar que las transacciones que se realicen sean las correctas. Debemos tener en cuenta que por Paypal lo que es enviado por la cuenta del apostador, no es lo que llega a la cuenta de la empresa, esto se da como se explicó por los valores de comisión que cobra la plataforma de pago que se está usando para realizar dichos movimientos.

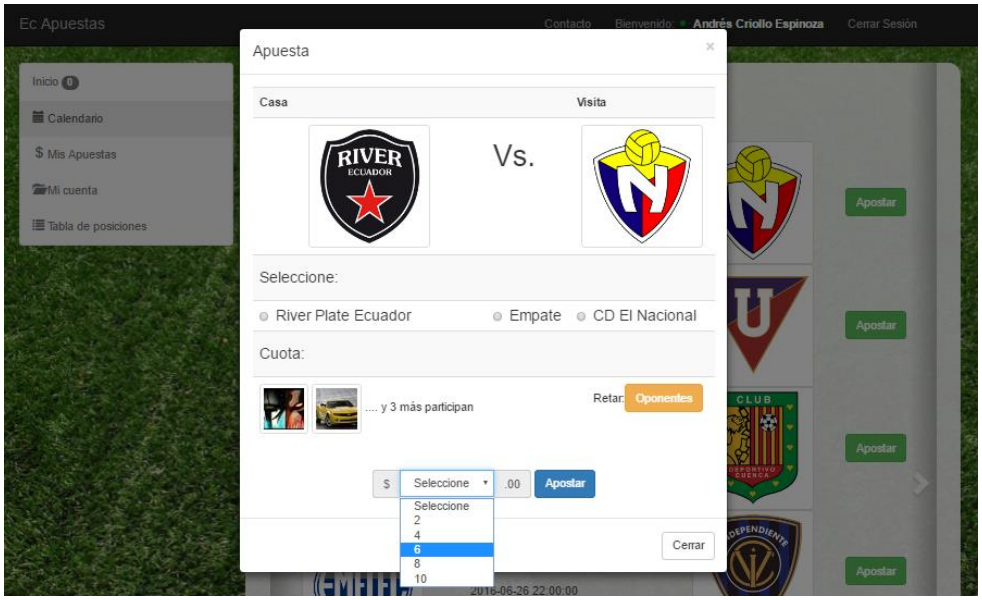


Fig. 88 Selección del equipo para apuesta para la aplicación

Se puede observar como la aplicación de manera satisfactoria abre la plataforma de Paypal, con el partido y la cantidad elegida en la apuesta.

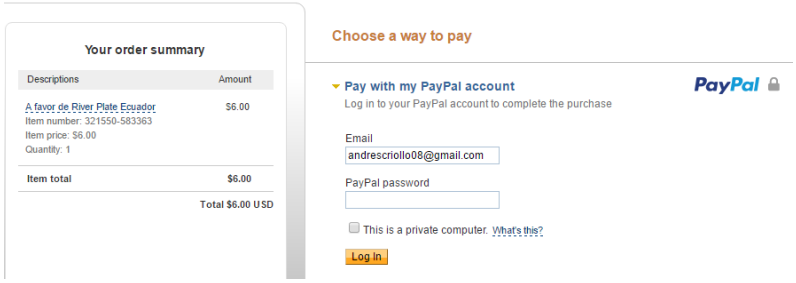


Fig. 87.Comprobación de información en Paypal.

Para poder conseguir la finalización de la transacción una vez iniciada la sesión con los datos de la cuenta, solo se debe proceder a confirmar el pago y comprobar que tanto la aplicación híbrida como la API Paypal, reflejen el proceso.

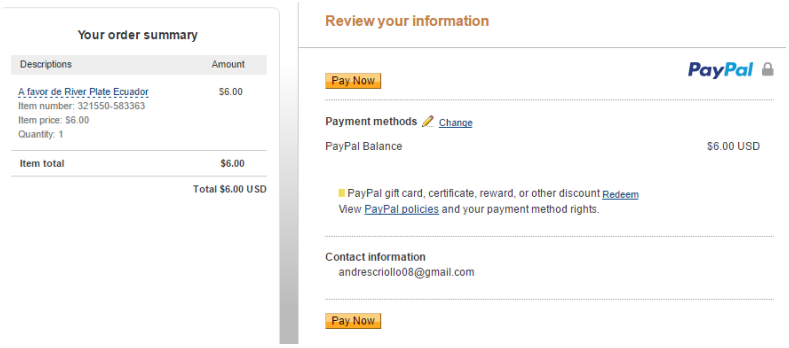


Fig. 90. Confirmación de pago de apuesta.

Al autorizar el pago una notificación se mostrará en pantalla, dando por terminado el proceso de pago.

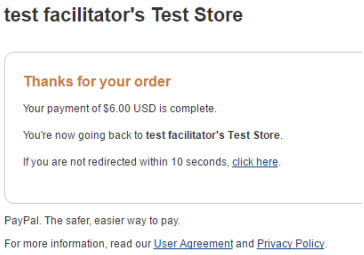


Fig. 91. Aviso de pago satisfactorio.

El usuario apostador una vez confirmado el pago, en su cuenta de Paypal recibirá una notificación, con los datos y valores seleccionados en su apuesta.

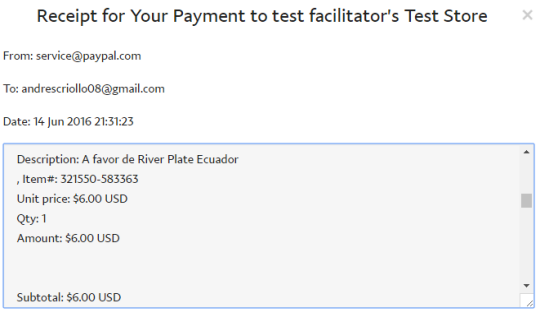


Fig. 92. Notificación detallada de la transacción

Una notificación similar recibe la cuenta asociada a la aplicación, con el aviso que ha sido acreditado un nuevo monto con la apuesta que ha sido realizada.

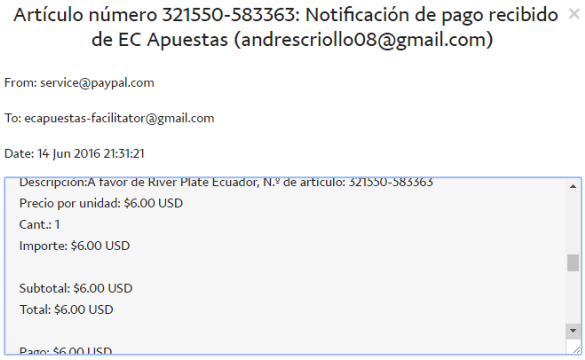
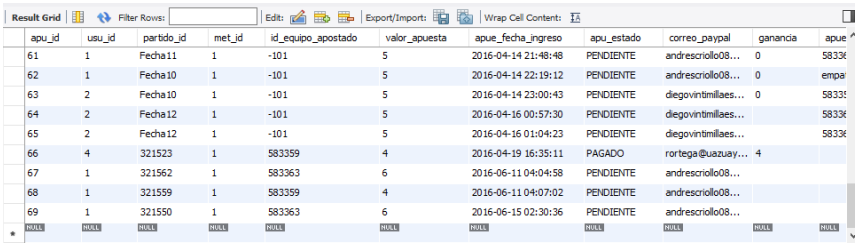


Fig. 93. Notificación recibida por ECApuestas la aplicación tras la realización de apuestas por el usuario

La transacción a más de guardarla en la plataforma de Paypal, es almacenada en la base de datos de la aplicación híbrida, con la finalidad de mantener un control de apuestas y estadisticas actualizadas de cada usuario.



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

apu_id	usu_id	partido_id	met_id	id_equipo_apostado	valor_apuesta	apue_fecha_ingreso	apu_estado	correo_paypal	ganancia	apue
61	1	Fecha11	1	-101	5	2016-04-14 21:48:48	PENDIENTE	andrescriollo08...	0	5833
62	1	Fecha10	1	-101	5	2016-04-14 22:19:12	PENDIENTE	andrescriollo08...	0	empa
63	2	Fecha10	1	-101	5	2016-04-14 23:00:43	PENDIENTE	diegovintimillaes...	0	5833
64	2	Fecha12	1	-101	5	2016-04-16 00:57:30	PENDIENTE	diegovintimillaes...		5833
65	2	Fecha12	1	-101	5	2016-04-16 01:04:23	PENDIENTE	diegovintimillaes...		5833
66	4	321523	1	583359	4	2016-04-19 16:35:11	PAGADO	rortega@uazuay...	4	
67	1	321562	1	583363	6	2016-06-11 04:04:58	PENDIENTE	andrescriollo08...		
68	1	321559	1	583359	4	2016-06-11 04:07:02	PENDIENTE	andrescriollo08...		
69	1	321550	1	583363	6	2016-06-15 02:30:36	PENDIENTE	andrescriollo08...		
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Fig. 94. Apuestas de los usuarios en la base de datos de ECApuestas

El usuario verá reflejado la nueva transacción en la seccion del menu “Mis Apuestas”

Fig. 95. Sección del menú Mis Apuestas

3.10.1.2 Prueba de pago de apuestas

Esta prueba tendrá como objetivo, la comprobación de dos elementos principales, el algoritmo de selección de las personas ganadoras y por ende el movimiento de dinero de la cuenta EC Apuestas a cada una de las cuentas de los usuarios resultantes del algoritmo.

3.10.1.2.1 Algoritmo de selección

Las lineas de código que se muestran en la Fig. 96, actualizan los datos correspondientes a los resultados provenientes de la API de resultados futbol, mediante un proceso que se ejecuta de manera automática en el servidor aproximandamente cada 12 horas.

```php
function actualizar_partidos($conexion){
    $sql = "select * from app_parametros where nombre='API FOOTBALL'";
    $consulta=mysqli_query($conexion,$sql);
    $registro = mysqli_fetch_array($consulta);
    $api_football= $registro['valor'];
    for($i=1;$i<50;$i++){
        $json = file_get_contents("http://www.resultados-futbol.com/scripts/api/api.php?tz=Europe/Ma
        $data = json_decode($json,true);
        $partidos= $data["match"];
        //print_r($partidos);
        $numero_partidos_fecha = count($partidos);
        for($j=0;$j<$numero_partidos_fecha;$j++) {
            $id_partido = $partidos[$j]["id"];
            if($id_partido=="")
                break 2;
            $partido_fecha = $partidos[$j]["schedule"];
            $ronda = $partidos[$j]["round"];
            $partido_estado = $partidos[$j]["status"];
            $equipo_local_id = $partidos[$j]["team1"];
            $equipo_visita_id = $partidos[$j]["team2"];
            //echo $equipo_local_id.'equipo1';
            $goles_local = $partidos[$j]["local_goals"];
            $goles_visita = $partidos[$j]["visitor_goals"];
            $comentarios = $partidos[$j]["extraTxt"];
            $sql = "select * from app_partido where id_partido_api='$id_partido'";
            $consulta=mysqli_query($conexion,$sql);
            $filas= mysqli_num_rows($consulta);
            echo $ronda;
            //comentario de porque se aplazo el partido
            if($filas<=0){
                //cambiar ids por las de las tablas propias
                $equipo_local_id = validar_equipo($equipo_local_id,$conexion);
                $equipo_visita_id = validar_equipo($equipo_visita_id,$conexion);
                //echo $equipo_local_id.'vs'.$equipo_visita_id.'<br>';
                $sql= "insert into app_partido(id_partido_api,id_equipo_local,id_equipo_visita,goles_
                $consulta=mysqli_query($conexion,$sql);
            }else if($filas>0){
                $sql = "update app_partido set goles_equipo_local=$goles_local,goles_equipo_visita='$
                $consulta=mysqli_query($conexion,$sql);
            }
        }
    }
}
```

Fig. 96. Algoritmo de selección de ganadores

Una vez se dispare la función de actualización la aplicación híbrida recorre la BD donde estan almacenadas las apuestas de los usuarios y realiza la selección de ganadores mediante las líneas de código de la Fig 97, donde primero filtra las apuestas en estado “PENDIENTE” y seguido a ello realiza las comprobaciones de las apuestas(equipo apostado contra equipo ganador) para verificar los ganadores y posteriormente realizar el pago de las mismas. Una vez ejecutado el pago, se dispara un evento tanto en la API de Paypal y en la aplicación que da como concluida la transacción tras la actualización de los estados de las apuestas.

```php
$sql= 'select * from app_apuestas a, app_partido p where a.apu_estado="PENDIENTE" and p.estado="1"
//echo $sql;
$consulta=mysqli_query($conexion,$sql);
$numero_filas_apuestas= mysqli_num_rows($consulta);
//echo $apiContext;
//require __DIR__ . '/../bootstrap.php';
$numero_ganadores= 0;
for($i=0;$i<$numero_filas_apuestas;$i++){
    $registro = mysqli_fetch_array($consulta);
    $id_partido = $registro['partido_id'];
    $id_apostado=$registro['id_equipo_apostado'];
    $valor_apuesta=$registro['valor_apuesta'];
    $id_apuesta = $registro['apu_id'];
    //comprobar partido y personas que apostaron en cada rango de apuesta
    $datos = estadistica_partido($id_partido,$id_apostado,$valor_apuesta,$conexion);
    //print_r($datos);
    if($datos['ganancia']>0){
       $correo = $registro['correo_paypal'];
       //echo $id_partido;
       realizar_pago($id_apuesta,$correo,$datos['asunto'],$datos['nota'],$datos['ganancia'],$nume
       $numero_ganadores++;
       $sql = 'update app_apuestas set apu_estado="PAGADO", ganancia="'.$datos['ganancia'].'" whe
       $consulta1=mysqli_query($conexion,$sql);
    }else{
        $sql = 'update app_apuestas set apu_estado="PERDIDO", ganancia="0" where apu_id='.$id_apu
       $consulta1=mysqli_query($conexion,$sql);
    }
}
?>
```

Fig. 97. Algoritmo de selección de los ganadores de las apuestas

3.10.1.2.2 Pago de Apuestas

Tras la selección de ganadores, el proceso correspondiente al pago de apuestas, se realiza con el código de la Fig. 98, que proporciona a la API de Paypal los datos de las cuentas de los usuarios a pagar, así como las cantidades que debe ser abonado a cada una de ellas. Para ello usamos el SDK proporcionado por Paypal, ocupando las librerías que ayudarán en el proceso del pago de apuestas.

Fig. 98. Algoritmo de pago de los ganadores de las apuestas

4 Conclusiones

La reutilización de componentes, ya sean estos de terceros o propios, para el desarrollo de aplicaciones cada vez más grandes y robustas, reduce significativamente el costo, tiempo y esfuerzo. En la actualidad aunque no existe una normativa oficial sobre el cómo se deben desarrollar las aplicaciones, con el fin de abstraer funcionalidades en API´s independientes, sin embargo, los programadores al momento de desarrollar sus aplicaciones intentan hacerlo de manera modular, tratando de dividir en procesos independientes las partes que componen su software, estas partes deben tener la capacidad de poder ser utilizados en cualquier sitio del sistema donde sean llamados o hasta en una aplicación distinta de donde fue creado, sin ningún tipo de dificultad, y una vez integrado pueda funcionar de forma correcta, sin la necesidad de mayores cambios en su estructura para una óptima adaptación. Esto favorece de forma positiva a la iniciativa de que cada vez se realice aplicaciones híbridas, motivando a los desarrolladores a crear aplicaciones más atractivas, tanto visualmente como de manera funcional.

Las principales funcionalidades de las aplicaciones web hibridas son: la combinación, con la cual, transformamos los datos existentes en otros más útiles, la visualización, donde se muestra al usuario la información necesaria, la que, se generó a partir de la integración de los datos y la agregación, que permite a las aplicaciones ser flexibles para si fuera necesario seguir integrando más Apis con la finalidad de aumentar el alcance del sistema híbrido. Esto nos permite complementar servicios existentes dando un valor agregado o generar nuevas soluciones más completas.

En la etapa de desarrollo de este proyecto se utilizaron 3 componentes: Facebook para el registro de usuarios, Paypal para el proceso de pago o desembolso, y resultados-futbol para la obtención de la información de los partidos del campeonato de fútbol ecuatoriano. Para llegar a los resultados esperados hemos probado varios servicios de información deportiva, para encontrar el componente se determinó:

- Calidad del componente
- Tiempo de respuesta
- Costo

- Numero de accesos
- Documentación

En la etapa de análisis de los componentes se determinó que la Api resultados-futbol obtuvo mejor tiempo de respuesta a las peticiones solicitadas por la web híbrida, así como también, brindaba la posibilidad de realizar un número de peticiones al día considerables para la realización del prototipo y cuenta con una excelente documentación la cual facilitó la implementación.

El diseño e implementación de este trabajo, fue bajo la premisa, de guía para el desarrollo de aplicaciones híbridas, ya que, al ser un tema no tan conocido en la actualidad, mostrar mediante el desarrollo de este mashup, conceptos, métodos, arquitecturas, lenguajes de programación, básicos que puedan orientar al programador para futuros proyectos.

Como se ha dicho anteriormente, al ser un concepto introducido en la actualidad, está siendo estudiado y explorado por muchos desarrolladores, que, como resultado, ayudan con pautas y métricas que pueden orientar al desarrollo de un consorcio internacional donde se pueda establecer estándares para la creación, comunicación y utilización de mashups. Por lo tanto, lo expresado en este documento, no es una norma, ni tampoco un modelo, más bien un manual, que pueda facilitar la comprensión del cómo funciona un sistema híbrido y poder definir según nuestro criterio formado en este tema a base de investigación, análisis y ejecución, el cómo llegar a implementarlo en nuestros proyectos. Para ello es de suma importancia, la parte conceptual de este trabajo, que trata de mostrar las grandes ventajas que la reutilización de componentes puede ofrecer para ayudar a los desarrolladores de componentes de software, a la investigación e implementación de cada vez más llamativas y versátiles API´s que puedan ser compartidas por el resto de la comunidad de programadores alrededor del mundo.

Con la finalización de este trabajo, tanto investigativo como práctico, podemos expresar según nuestra experiencia, que las aplicaciones híbridas orientadas a la Web, son los cimientos para la programación de sistemas, tanto en la nube como de escritorio. En un

mundo globalizado en el que contamos con el mayor repositorio como lo es el Internet, que permite compartir y adquirir componentes de software, para evitar trabajo innecesario, creando algo ya creado. Podemos garantizar que la reutilización de API´s, favorece mucho al momento de analizar, diseñar e implementar soluciones a diversos problemas, ya que, la mayoría de problemas a los que nos podemos enfrentar al momento de encontrar una solución a un conflicto, puede ser resuelto, tras un poco de investigación, con el objeto de encontrar una API que haya sido elaborada por alguien más, que se pueda adaptar a nuestras necesidades. Según nuestra experiencia, en Internet se encuentran componentes software con funcionalidades que pueden ayudar a la resolución, de casi cualquier eventualidad al momento de desarrollar.

Podemos encontrar diversas soluciones que nos puedan ayudar a resolver nuestras necesidades, pero debemos estar atentos al momento de escoger entre una de ellas e implementarlo, ya que, esta decisión puede ayudar o perjudicar el rendimiento de nuestro software. Por ello es importante, buscar alternativas, comprobar que sea una API que garantice confianza al implementarla, este factor puede ser estudiado tras visualizar el tiempo que lleva disponible, el número de versiones, comunidad que tiene detrás, número de conexiones para realizar peticiones, cómo la API maneja el tráfico de red, etc. El ver todas estas métricas antes de seleccionar una opción a utilizar, puede ayudarnos a encontrar la mejor alternativa, para la reutilización del mashup dentro de nuestra web híbrida.

5 Referencias

Apache HTTP Server Poject. (10 de 03 de 2015). Obtenido de Apache HTTP Server Poject: http://httpd.apache.org/ABOUT_APACHE.html

Camazon, J. N. (2014). Aplicaciones Web. En J. N. Camazon, *Aplicaciones Web* (pág. 42).

Cappielo, C., Florian, D., Matera, M., & Pautasso, C. (2010). Information Quality in Mashups. *IEEE Computer Society*, 31-32.

Cobo, Gómez, P., & Pérez, D. (2007). PHP y MySQL: tecnologías para el desarrollo de aplicaciones web. En Á. G. Cobo, *PHP y MySQL: tecnologías para el desarrollo de aplicaciones web* (págs. 339-340). Ediciones Díaz de Santos.

Florian Daniel, M. M. (2009). A Quality Model for Mashup Components. *Research Gate*, 16.

Talón, E. M. (2012). APACHE. En E. M. Talón, *APACHE* (pág. 4).

6 Bibliografía

Andreas Auinger, M. E. (2010). Mixing Content and Endless Collaboration – MashUps: Towards Future Personal Learning Environments. *Institute of Medical Informatics, Statistics and Documentation, Austria* , 3-6.

Kaar, C. (2007). An introduction to Widgets with particular emphasis on Mobile Widgets. *Mobile Computing*, 3-5.

Stefan Bitzer, S. R. (2009). Mashups as an Architecture for Knowledge Management Systems . *Proceedings of the 42nd Hawaii International Conference on System Sciences*, (pág. 10).

Vara, M., Juan, M., López, S., & Verde Marín, J. (2014). Desarrollo web en entorno servidor. En *Desarrollo web en entorno servidor* (págs. 262-268). España: RA-MA Editorial.

7 Anexos

Doctora Jenny Ríos Coello, Secretaria de la Facultad de Ciencias de la Administración de la Universidad del Azuay,

C E R T I F I C A:

Que, el Consejo de Facultad en sesión del 05 de noviembre de 2015, conoció la petición del (los) estudiante(s) **Andrés Patricio Criollo Espinoza y y Diego Ricardo Vintimilla Espinoza** con código(s) **44017 y 41113 respectivamente**, registrado(s) en la Unidad de Titulación Especial, quien(es) denuncia(n) su trabajo de titulación denominado: **"ESTUDIO, ANALISIS Y DESARROLLO DE UNA APLICACIÓN WEB HIBRIDA"**en la modalidad: Proyecto Integrador y presentado como requisito previo a la obtención del título de Ingeniero de Sistemas .-El Consejo de Facultad acoge el informe de la Junta Académica y aprueba la denuncia. Designa como Director(a) a Ing. Rubén Ortega López y como miembro del Tribunal Examinador a Ing. Francisco Salgado Arteaga. De conformidad con el cronograma de la Unidad de Titulación el (los) peticionario(s) debe presentar su trabajo de titulación hasta el 11 de marzo de 2016.

Cuenca, 06 de noviembre de 2015

Dra. Jenny Ríos Coello
Secretaria de la Facultad de Ciencias de la Administración

UNIVERSIDAD DEL AZUAY
FACULTAD DE ADMINISTRACION
SECRETARIA

Oficio Nro. 158-2015-DIST-UDA

Cuenca, 30 de Octubre de 2015

Señor Ingeniero
Xavier Ortega Vázquez
DECANO DE LA FACULTAD DE CIENCIAS DE LA ADMNISTRACIÓN
Presente.-

De nuestras consideraciones:

La Junta Académica de la Escuela de Ingeniería de Sistemas y Telemática, reunida el día 30 de octubre del 2015, recibió el proyecto de tesis titulado "Estudio, análisis y desarrollo de una aplicación Web híbrida", presentado por los estudiantes Andrés Patricio Criollo Espinoza y Diego Ricardo Vintimilla Espinoza, estudiantes de la Escuela de Ingeniería de Sistemas y Telemática, y revisado por el Ing. Rubén Ortega, previo a la obtención del título de Ingeniero de Sistemas.

Por lo expuesto, y de conformidad con el Reglamento de Graduación de la Facultad, recomienda como director y responsable de aplicar cualquier modificación al diseño del trabajo de graduación posterior a al Ing. Rubén Ortega y como miembro del Tribunal a Francisco Salgado Ph.D.

Atentamente,

Ing. Marcos Orellana Cordero
Director Escuela de Ingeniería de Sistemas y Telemática
Universidad del Azuay

Oficio Nro. 159-2015-DIST-UDA

Cuenca, 30 de Octubre de 2015

Señor Ingeniero
Xavier Ortega Vázquez
DECANO DE LA FACULTAD DE CIENCIAS DE LA ADMNISTRACIÓN
Presente.-

De mis consideraciones:

La Junta Académica de la Escuela de Ingeniería de Sistemas y Telemática, reunida el día 30 de octubre del 2015, revisó la documentación del proyecto de tesis denominado "Estudio, análisis y desarrollo de una aplicación Web híbrida", presentado por los estudiantes Andrés Patricio Criollo Espinoza y Diego Ricardo Vintimilla Espinoza, estudiantes de la Escuela de Ingeniería de Sistemas y Telemática, y revisado por el Ing. Rubén Ortega, previo a la obtención del título de Ingeniero de Sistemas.

La Junta considera que la documentación cumple con las normas legales y reglamentarias de la Universidad y de la Facultad de Ciencias de la Administración y avala la aprobación por parte del tribunal designado, así por su digno intermedio, el conocimiento y aprobación por parte del Consejo de Facultad.

Atentamente,

Ing. Marcos Orellana Cordero
Director Escuela de Ingeniería de Sistemas y Telemática
Universidad del Azuay

UNIVERSIDAD DEL AZUAY

Cuenca, 30 de Octubre del 2015

Señor Ingeniero

Xavier Ortega Vázquez

DECANO DE LA FACULTAD DE CIENCIAS DE LA ADMINNISTRACIÓN

Presente.-

De nuestras consideraciones:

Nosotros Andrés Patricio Criollo Espinoza con código 44017 junto con Diego Ricardo Vintimilla Espinoza con código 41113, solicitamos a usted se nos conceda la aprobación del proyecto de Titulación con el tema "Estudio y Desarrollo de una aplicación web híbrida"..

Por la atención que preste a nuestra solicitud le damos nuestros más sinceros agradecimientos.

Atentamente,

Andrés Criollo
ua044017
CI: 010463337-5

Diego Vintimilla
ua041113
CI: 010529367-4

Edición autorizada de 30.000 ejemplares
Del 678.501 al 708.500
Nº 0704236

CONVOCATORIA

Por disposición de la Junta Académica de Ingeniería de Sistemas y Telemática, se convoca a los Miembros del Tribunal Examinador, a la sustentación del Protocolo del Trabajo de Titulación: "Estudio, análisis y desarrollo de una aplicación Web híbrida" presentado por los estudiantes **Andrés Patricio Criollo Espinoza** con código 44017 y **Diego Ricardo Vintimilla Espinoza** con código 41113 previa a la obtención del grado de Ingeniero de Sistemas, para el día **VIERNRES 30 DE DE OCTUBRE DE 2015 A LAS 08h00.**

Cuenca, 28 de octubre de 2015

Dra. Jenny Ríos Coello
Secretaria de la Facultad

Ing. Rubén Ortega López .....

Ing. Francisco Salgado Arteaga Francisco Salgado .....

UNIVERSIDAD DEL AZUAY

ACTA

SUSTENTACIÓN DE PROTOCOLO/DENUNCIA DEL TRABAJO DE TITULACIÓN

1.1 **Nombre del estudiante:** Andrés Patricio Criollo Espinoza y Diego Ricardo Vintimilla Espinoza
1.2 Código 44017 y 41113
1.3 **Director sugerido**: Ing. Rubén Ortega López
1.4 **Codirector (opcional)**: _____
1.5 **Tribunal:** Ing. Francisco Salgado Arteaga
1.6 **Título propuesto:** "Estudio, análisis y desarrollo de una aplicación Web híbrida"
1.7 **Resolución**:

1.7.1 Aceptado sin modificaciones SI

1.7.2 Aceptado con las siguientes modificaciones:

_____

_____

_____

1.7.3 Responsable de dar seguimiento a las modificaciones: Ing. Rubén Ortega López
No aceptado
- Justificación:

_____
_____
_____

Tribunal

Ing. Rubén Ortega López

Ing. Francisco Salgado Arteaga

Sr. Andrés Criollo Espinoza

Sr. Diego Vintimilla Espinoza

Dra. Jenny Ríos Coello
Secretario de Facultad

Fecha de sustentación: Viernes 30 de octubre de 2015 al as 08h00

UNIVERSIDAD DEL AZUAY

RÚBRICA PARA LA EVALUACIÓN DEL PROTOCOLO DE TRABAJO DE TITULACIÓN

1.1 **Nombre del estudiante:** Andrés Patricio Criollo Espinoza y Diego Ricardo Vintimilla Espinoza

1.1.1 **Código** 44017 y 41113

1.1 Director sugerido: Ing. Rubén Ortega López

1.2 Codirector (opcional):.

1.3 Título propuesto: "Estudio, análisis y desarrollo de una aplicación Web híbrida"

1.4 Revisores (tribunal): Ing. Francisco Salgado Arteaga

1.5 Recomendaciones generales de la revisión:

	Cumple totalmente	Cumple parcialmente	No cumple	Observaciones (*)
Línea de investigación				
1. ¿El contenido se enmarca en la línea de investigación seleccionada?	✓			
Título Propuesto				
2. ¿Es informativo?	✓			
3. ¿Es conciso?	✓			
Estado del arte				
4. ¿Identifica claramente el contexto histórico, científico, global y regional del tema del trabajo?	✓			
5. ¿Describe la teoría en la que se enmarca el trabajo	✓			
6. ¿Describe los trabajos relacionados más relevantes?	✓			
7. ¿Utiliza citas bibliográficas?	✓			
Problemática y/o pregunta de investigación				
8. ¿Presenta una descripción precisa y clara?	✓			
9. ¿Tiene relevancia profesional y social?	✓			
Hipótesis (opcional)				
10.¿Se expresa de forma clara?				
11.¿Es factible de verificación?				
Objetivo general				
12.¿Concuerda con el problema formulado?	✓			
13.¿Se encuentra redactado en tiempo verbal infinitivo?	✓			
Objetivos específicos				

14.¿Concuerdan con el objetivo general?	✓			
15.¿Son comprobables cualitativa o cuantitativamente?	✓			
Metodología				
16.¿Se encuentran disponibles los datos y materiales mencionados?	✓			
17.¿Las actividades se presentan siguiendo una secuencia lógica?	✓			
18.¿Las actividades permitirán la consecución de los objetivos específicos planteados?	✓			
19.¿Los datos, materiales y actividades mencionadas son adecuados para resolver el problema formulado?	✓			
Resultados esperados				
20.¿Son relevantes para resolver o contribuir con el problema formulado?	✓			
21.¿Concuerdan con los objetivos específicos?	✓			
22.¿Se detalla la forma de presentación de los resultados?	✓			
23.¿Los resultados esperados son consecuencia, en todos los casos, de las actividades mencionadas?	✓			
Supuestos y riesgos				
24.¿Se mencionan los supuestos y riesgos más relevantes?	✓			
25.¿Es conveniente llevar a cabo el trabajo dado los supuestos y riesgos mencionados?	✓			
Presupuesto				
26.¿El presupuesto es razonable?	✓			
27.¿Se consideran los rubros más relevantes?	✓			
Cronograma				
28.¿Los plazos para las actividades son realistas?	✓			
Referencias				
29.¿Se siguen las recomendaciones de normas internacionales para citar?	✓			
Expresión escrita				
30.¿La redacción es clara y fácilmente comprensible?	✓			
31.¿El texto se encuentra libre de faltas ortográficas?	✓			

(*) Breve justificación, explicación o recomendación.

- Opcional cuando cumple totalmente,
- Obligatorio cuando cumple parcialmente y NO cumple.

.....

.....

.....

Ing. Rubén Ortega López

Ing. Francisco Salgado Arteaga

Doctora Jenny Ríos Coello, Secretaria de la Facultad de Ciencias de la Administración de la Universidad del Azuay

C E R T I F I C A:

Que el señor Decano con autorización amplia y suficiente concedida por el Consejo de Facultad en sesión del 25 de febrero de 2016, conoció la petición de los estudiantes **ANDRES PATRICIO CRIOLLO ESPINOZA** con código 44017 y **DIEGO RICARDO VINTIMILLA ESPINOZA** con código 41113, quienes solicitan prórroga para la presentación del trabajo de titulación "Estudio y Desarrollo de una aplicación web híbrida" previa a la obtención del título de Ingeniero, cuya fecha vence el 11 de marzo de 2016. *El señor Decano considerando el libera b) del Art. 6 del Instructivo para la Conformación y Funcionamiento de las Unidades de Titulación Especial en las Carreras de Grado de la Universidad del Azuay, aprobado por el Consejo Universitario el 5 de mayo de 2015, resuelve aprobar la solicitud y conceder una prórroga de seis meses, esto es hasta el* ***11 de septiembre de 2016.***

Cuenca, marzo 10 de 2016.

Dra. Jenny Ríos Coello
Secretaria de la Facultad

UNIVERSIDAD DEL AZUAY
FACULTAD DE ADMINISTRACIÓN
SECRETARÍA

UNIVERSIDAD DEL AZUAY

Universidad del Azuay

FACULTAD DE CIENCIAS DE LA ADMINISTRACIÓN

ESCUELA DE INGENIERIA DE SISTEMAS

UNIDAD DE TITULACIÓN ESPECIAL

Tema

Estudio, Análisis y Desarrollo de una Aplicación de Web Hibrida.

Responsables

Andrés Criollo E.

Diego Vintimilla E.

Director

Ing. Rubén Ortega López

Edición autorizada de 30.000 ejemplares
Del 678.501 al 708.500 Nº 0704215

GUIA PARA LA ELABORACIÓN Y PRESENTACIÓN DE LA DENUNCIA/PROTOCOLO DE TRABAJO DE TITULACIÓN

1. DATOS GENERALES

1.1 Nombres de estudiantes:

Vintimilla Espinoza Diego Ricardo,

Criollo Espinoza Andrés Patricio.

1.1.1 Código: 41113, 44017

1.1.2 Contacto: teléfonos, convencional, celular y correo

4083255

0984726984

dieguini866@gmail.com

4107489

0999874459

andrescriollo08@hotmail.com

1.2 Director sugerido: Ortega López Vicente Rubén, Ingeniero.

1.2.1 Contacto: rortega@uazuay,edu.ec

1.3 Co-Director sugerido: Erazo Garzón, Lenin Xavier Ingeniero.

1.3.1 Contacto: lerazo@uazuay.edu.ec

1.4 Asesor Metodológico:

Salgado Arteaga Francisco Rodrigo, Ingeniero Civil Msc

1.5 Tribunal designado.

1.6 Aprobación:

1.7 Línea de Investigación de la carrera: 1203 Informática de computadores

1.7.1 Programa: 1203.17

1.7.2 Tipo de trabajo: El trabajo está orientado a la parte metodológica donde se estudiarán nuevos conceptos sobre integración e interoperabilidad de aplicaciones actuales, y se complementara con el

UNIVERSIDAD DEL AZUAY

desarrollo de un mashup (Aplicación Web hibrida) utilizando APIs (Interfaz de programación de desarrollo - *Application Programming Interface*) para obtener un nuevo producto en el cual se aplique todos estos conceptos.

1.8 Área de estudio: El área al cual se enfoca este trabajo es a la ingeniería y calidad de software principalmente los conceptos referentes a integración de aplicaciones, reutilización de componentes, interoperabilidad e ingeniería web para facilitar la creación de nuevos servicios y/o productos software.

1.9 Titulo propuesto:

Estudio, Análisis y Desarrollo de una Aplicación de Web Hibrida.

1.10 Subtítulo: Integración e interoperabilidad de componentes web.

1.11 Estado del proyecto:

La propuesta es una adaptación tecnológica que consiste en la investigación e implementación de una aplicación web que integre los recursos web de terceros generando como resultado una aplicación que solucione un problema en específico. Esta aplicación a su vez estará diseñada e implementada de tal manera que pueda ser ocupado por terceros desarrolladores para su beneficio.

2. CONTENIDO

2.1 Motivación de la investigación: Debido a la evolución de la web y la aparición de la Web 2.0 con la interoperabilidad para el intercambio de información y conocimiento, esto hace que los usuarios generen gran cantidad de información, debido a ello se generan aplicaciones para brindar un nuevo servicio, encontrando una oportunidad para elaborar una aplicación vanguardista. La mayoría de las aplicaciones tienen algunas características únicas e interesantes que abstrayéndolas e integrándolas en un nuevo contexto puede dar como resultado una nueva aplicación.

Edición autorizada de 30.000 ejemplares
Del 678.501 al 708.500 Nº 0704214

2.2 Problemática:

En la actualidad las fuentes de información generadas por los usuarios al ingresar a la web, ha hecho que se creen aplicaciones que ocupen dicha información para brindar servicios mejores y personalizados. Este trabajo pretende, dar a conocer las formas en que se puede utilizar dicha información para generar un nuevo servicio a partir de aplicaciones creadas por terceros, generando una interfaz donde se integre dichos servicios, además de añadir nuevas funcionalidades según las necesidades del caso, consiguiendo con esto generar un nuevo producto software.

2.3 Resumen: Este trabajo pretende aplicar conceptos de integridad e interoperabilidad entre diferentes aplicaciones que tienen diferentes modelos de negocio y a través de estas generar una aplicación propia que adquiera lo más importante de cada una de estas aplicaciones, obteniendo una nuevo repositorio de información que dará origen a un nuevo servicio.

Al concluir este trabajo se obtendrá una aplicación web hibrida con una interfaz de usuario y que contenga información referente al ámbito deportivo, y teniendo como añadido la posibilidad de generar apuestas entre los usuarios de la aplicación. Para lograr esta meta se utilizarán los conceptos adquiridos a lo largo de nuestra formación universitaria además de diferentes componentes, cada una de ellas con su propia arquitectura y modelo de negocio, de todas ellas, se seleccionarán las más oportunas para nuestro caso de estudio y con ellas se va a generar un nuevo y propio modelo de negocio orientado al área deportiva.

De esta manera con este trabajo se pretende dar un producto que pueda ser usado por futuros desarrolladores. También nuestro propósito es el dar a conocer el uso de recursos web de servicios recurrentes por los usuarios y obtener información relevante y en si poder generar un nuevo servicio y generar una API para que otros servicios lo puedan utilizar. Esto permite a los desarrolladores poder crear nuevo software.

2.4 Indagación exploratoria y base conceptual:

Desde la aparición de la Web, incontables usos se han ido integrando y usando de manera muy significativa, teniendo constante evolución en ámbitos, geográficos, estudiantiles, laborales y sociales. La web se va convirtiendo cada vez con mayor dominación en un espacio social, convirtiendo a esta era en una sociedad de conocimiento donde compartir información y comunicar conocimiento es muy habitual.

La Web 2.0 va evolucionando constantemente, y oportunamente se crean aplicaciones más poderosas donde la sencillez y eficacia de dicha aplicación es de suma importancia para el usuario final. Una de las metas de ingeniería de software dice que debemos tener muy en cuenta el principio de reusabilidad, con el objeto de conseguir objetivos más rápido, utilizando programación, plantillas, frameworks, servicios web, donde cada una de ellas ha nacido con el deseo de usar material ya elaborado en vez de construirlo.

Para llegar a esta nueva etapa evolutiva, debemos centrarnos en la llamada web semántica donde las ontologías para describir una nueva plataforma de cambio de información y conocimiento ha hecho surgir nuevas metodologías que tratan de cumplir los requerimientos actuales, una de estas tecnologías actuales se llama Aplicación Web hibrida (mashup).

Mashup se puede definir como una web hibrida donde se aloja varias aplicaciones elaboradas por terceros integrándolas con el fin de crear algo nuevo.

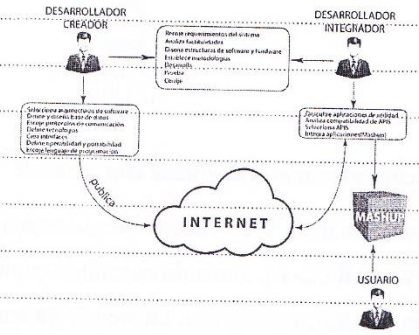


Fig. 1 Esquema de aplicación web hibrida

Fuente: Basado en (Florian Daniel, 2009)

Tipos de Aplicaciones Web Hibridas

Dependiendo del tipo de requerimientos o características las aplicaciones web hibridas se pueden clasificar en:

Consumidores: Combinan información de distintas aplicaciones desarrolladas por terceros y presentarlas en una interfaz gráfica sencilla.

Datos: Consumen información de varias Web feeds como Yahoo, Google y Twiter.

Empresariales: Combina información externa e interna a la organización dando valor agregado a los datos institucionales obteniendo como resultado una funcionalidad colaborativa y obteniendo una aplicación de negocio.

2.5 Objetivo general: Diseñar una aplicación web hibrida e implementarla en un caso de estudio.

2.6 Objetivos específicos:

- Analizar y seleccionar los diferentes componentes provenientes de fuentes externas.

- Comprobar compatibilidades de aplicaciones seleccionadas.
- Definir la arquitectura, protocolos, lenguaje de programación para la interoperabilidad e integración de la aplicación web hibrida.
- Diseño y creación de interfaces de usuario.
- Programar el componente software con los requerimientos del caso de estudio.
- Realizar pruebas de interoperabilidad entre las aplicaciones seleccionadas.
- Elaborar el documento final.

2.7 Metodología:

En este trabajo se va a utilizar el diseño modular para la solución del problema de nuestro caso de estudio, consiguiendo con ello profundizar en el análisis de cada uno de estos elementos por separado. Para ello se investigarán y analizarán trabajos que tengan estrecha relación con el desarrollo y uso de web hibridas. Al concluir con la fase de obtención de conocimiento, se diseñará un prototipo el cual brinde un nuevo servicio a partir la integración e interoperabilidad de componentes web de terceros, dando a conocer el beneficio de las web hibridas con un caso práctico.

2.8 Alcances y resultados esperados: El alcance con esta investigación y desarrollo es obtener un prototipo de una aplicación que contenga características propias, y a su vez generar nuestra propia web API para que otros desarrolladores puedan hacer uso.

2.9 Supuestos y riesgos: La investigación se muestra sólida en su viabilidad, pues se cuenta con las referencias bibliográficas para apoyar su desarrollo conceptual y metodológico. Algunas vulnerabilidades detectadas se encuentran al generar el

desarrollo del proyecto donde existen algunos casos puntuales que afectarían al proyecto.

SUPUESTOS Y RIESGOS	ALTERNATIVA
Compatibilidad de los componentes al momento de integrar en la aplicación hibrida.	Buscar un componente compatible en otras fuentes o comprar
Disponibilidad y calidad baja del componente.	Luego de la indagación se utilizara la mejor alternativa.
Escasa documentación del componente	Buscar fuentes especializadas
Alto costo de los componentes.	Por tratarse de un prototipo utilizaremos software libre

2.10 Presupuesto:

Rubro-Denominación	Costo USD (detalle)	Justificación ¿para qué?
Proveedor de internet ISP	$66	Para investigación y desarrollo.
Servidor web apache SSL	$110	Conexión segura Https
Libros y referencias	$100	Para la adquisición de fuentes de información relacionadas con el tema de estudio
Componentes software	$150	Componentes a integrar en la aplicación web hibrida
Impresiones	$50	Documentaciones finales y parciales para el director.

2.11 Financiamiento: Se financiará con fondos propios.

UNIVERSIDAD DEL AZUAY

2.12 Esquema tentativo:

Abstract: The development of this project is about the study and implementation of a Web Application Hybrid, where we seek to find a new service.

Introducción: El desarrollo de este proyecto se trata sobre el estudio e implementación de una Aplicación Web Hibrida, donde buscamos buscar un nuevo servicio.

- **Capítulo 1: Conceptos y Definiciones**
  - **a.** Concepto y Tipos de aplicaciones híbridas
  - **b.** Integración de datos
  - **c.** Integración de aplicaciones
  - **d.** Tecnologías web
    - Internet
    - Protocolos TCP/IP
    - Http Protocolo
    - Hypertext Markup Language
    - Formato de Datos
      - XML
      - JSON
  - **e.** Componentes de una aplicación web hibrida
  - **f.** Lógica de componentes
    - Web Service
    - RESTful Web Service
    - Java Script API
  - **g.** Componente de datos
    - ATOM
    - XML, JSON, CVS
    - Extracción de datos de la web
  - **h.** Componentes de interfaz de usuario
- **Capítulo 2:Análisis y definición de arquitectura aplicación web hibrida**
  - **a.** Análisis de protocolos de comunicación

b. Modelo Entidad-Relación (E-R)

c. Interfaces de usuario

d. Compatibilidad de componentes

e. Arquitectura de integración e interoperabilidad.

- **Capítulo 3: Implementación de prototipo**

a. Desarrollo de la aplicación web hibrida

b. Pruebas y correcciones

2.13 Cronograma:

Objetivo Específico	**Actividad**	**Resultado esperado**	**Tiempo (semanas)**
Analizar y seleccionar los diferentes componentes provenientes de fuentes externas.	Búsqueda y selección de las API´s que van integrar la aplicación web híbrida.	Listado de componentes a integrar y sus correspondientes alternativas.	**1**
Comprobar compatibilidades de aplicaciones seleccionadas.	Verificar compatibilidad de componentes software	Componentes software listos a integrar	**1**
Definir la arquitectura, protocolos, lenguaje de programación para la interoperabilidad e integración de la aplicación web hibrida.	Establecer requerimientos funcionales para el desarrollo de la aplicación web híbrida.	Lenguaje de programación, arquitectura, y modelo integrador sobre el cual se va a desarrollar el nuevo componente.	**1**

Diseño y creación de interfaces de usuario.	Elaboración del front end	Interfaces, botones, colores, tamaños, fuentes.	2
Programar el componente software con los requerimientos del caso de estudio.	Desarrollo de la web híbrida.	Mapa de navegación web, componentes software integrados.	3
Realizar pruebas de interoperabilidad entre las aplicaciones seleccionadas.	Pruebas de funcionalidad	Aplicación web híbrida	1
Elaboración del documento final	Culminación del contenido del documento	Libro con contenido final	1

2.14 Referencias:

- Rouse, Margaret "Open API" 01/01/2015 http://searchcloudapplications.techtarget.com/definition/open-API
- David Berlind ProgrammableWeb. 23/09/2013. http://www.programmableweb.com/news/programmablewebs-directory-hits-10000-apis.-and-counting./analysis/2013/09/23
- Adam DuVander, ProgrammableWeb. 03/01/2011. http://www.programmableweb.com/news/api-growth-doubles-2010-social-and-mobile-are-trends/2011/01/03\

- Duane, M. (2009). Mashups: The new breed of Web app. 10.
- Florian Daniel, M. M. (2009). A Quality Model for **Ma**shup Components. *Research Gate*, 16.

- Oliver, Y. (2008). The Mashup Opportunity. *Forrester*, 15.
- Rafael, F. (2008). Integracion de Aplicaciones. *Research Reporr*, 139.
- Saeed Aghaee, C. P. (2010). Mashup Development with HTML5. 8.
- Shyam, G. (2011). Itinerary Planner: Mashup Case Study. 13.

2.16 Firma del responsable

.....

Diego Ricardo Vintimilla Espinoza

.....

Andrés Patricio Criollo Espinoza

.....

Ing. Rubén Ortega

2.17 Fecha entrega:

Manual de usuario de EC Apuestas

La aplicación fue diseñada con el fin de realizar apuestas referentes al marco deportivo de nuestro país, concretamente en lo que al campeonato ecuatoriano de futbol se refiere. A continuación, se detallará las funcionalidades y aspectos que fueron desarrollados para la utilización de esta web híbrida.

Fig. 1. Logo de la aplicación EC Apuestas

1. Pantalla de Inicio

La primera pantalla que se muestra al usuario, después de darse de alta, o en su defecto al iniciar sesión, muestra las personas que ganaron en la fecha que fue jugada con anterioridad, así como la cantidad que el usuario acredito a su cuenta en la fecha.

Fig. 2. Pantalla inicial EC Apuestas

2. Registro

Para darnos de alta en EC Apuestas, lo tenemos que hacer mediante la API de Facebook, obteniendo la autorización del usuario, para tener acceso a sus datos de perfil como son: nombre, ubicación, contactos, foto.

Fig. 3. Pantalla de registro de la aplicación EC Apuestas

Para ello solo presionamos el botón “Registrar con Facebook” e iniciamos la sesión de la cuenta, para poder ingresar la aplicación.

Fig. 4. Pantalla de inicio de sesión con Facebook

3. Menú Principal

Aquí estarán disponibles las opciones de la aplicación para el usuario, entre las que encontramos las siguientes:

3.1. Calendario

El calendario muestra toda la información del campeonato actual en su totalidad, partidos jugados como partidos pendientes. En el cual podemos desplazarnos entre las distintas fechas, observando resultados y todos los encuentros correspondientes a cada jornada. También tenemos la posibilidad de apostar a los partidos ya sea de manera individual o de toda una fecha en concreto, las veces que queramos para cada partido y fecha aún no jugada del campeonato nacional.

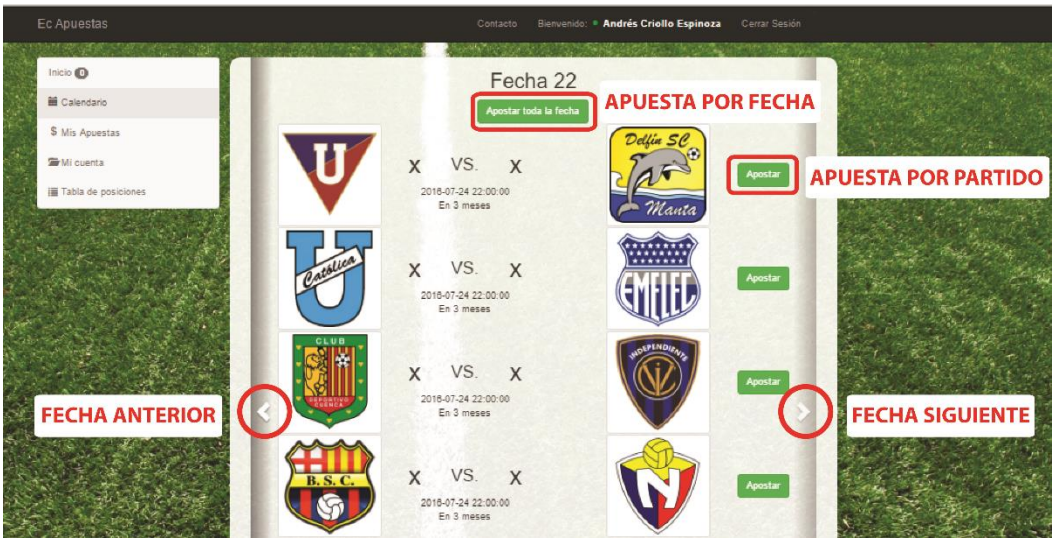


Fig.5. Calendario del campeonato nacional para apostar

3.1.1. Apuesta por partido

Podemos apostar a todos los partidos que deseemos que tengan el botón verde de "Apostar", al presionarlo, se mostrará la información del partido, los equipos a apostar, con empate como posibilidad, un listado con las cantidades disponibles para apostar.

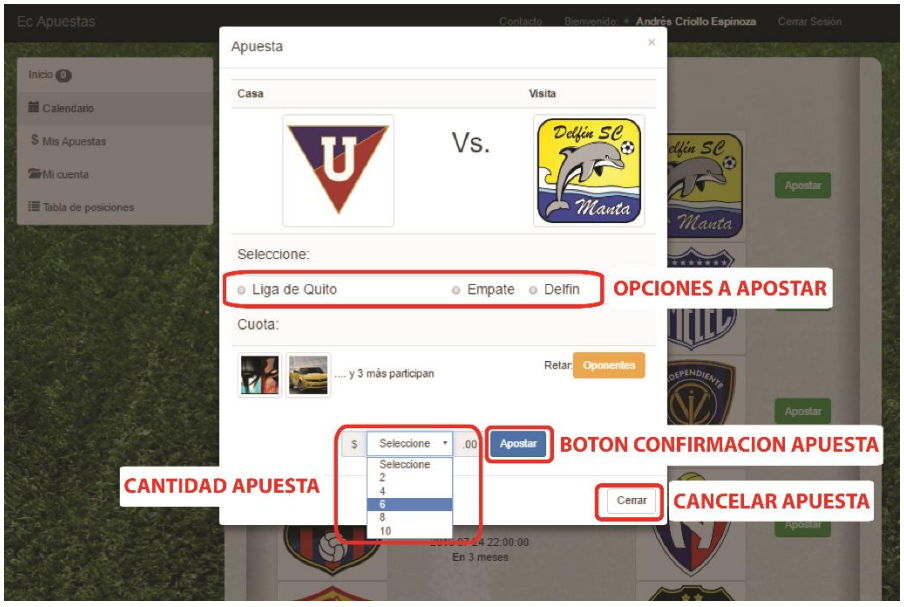


Fig. 6. Opción Apuesta por Partido

3.1.2. Apuesta por fecha

También EC Apuestas tiene la posibilidad de realizar apuestas a toda una fecha en concreto, para ello el usuario debe tener en cuenta, que solo es posible apostar a toda la fecha siempre y cuando ningún encuentro se haya disputado. Para ello tenemos que presionar el botón "Apostar toda la fecha", obteniendo una nueva ventana donde de manera similar a la apuesta por partido, debemos escoger de cada encuentro nuestra opción a apostar y a continuación seleccionar la cantidad de dinero a apostar. En esta modalidad de juego por el momento solo es posible apostar la suma de $5.

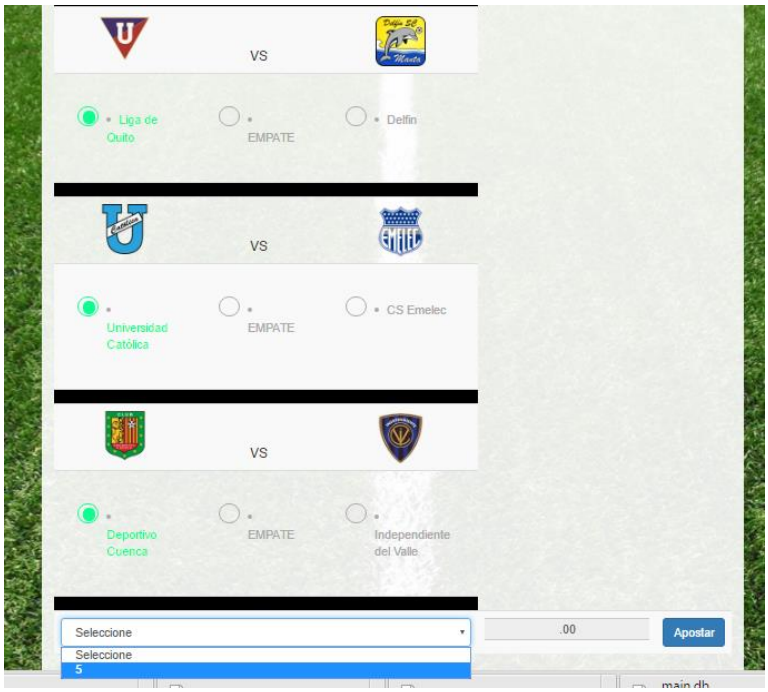


Fig. 7. Opción Apuesta por fecha

3.1.3. Cancelar Apuesta

La cancelación y cierre de la ventana apuesta, para poder seguir con la navegación por la aplicación, se lo puede realizar de dos formas, la primera, presionando el botón "Cancelar" que tanto la apuesta de partido y por fecha la tienen; la segunda, presionando con el mouse fuera de la ventana de la apuesta.

3.1.4. Confirmar Apuesta

Cuando el usuario este seguro de realizar una transacción en ECApuestas, solo debe darle al botón "Apostar" al final de cada tipo de apuesta. Esto abrirá una ventana de Paypal en el cual, el usuario debe iniciar sesión y seguir los pasos de confirmación del pago de la apuesta elegida.

Fig. 8. Página para pago por Paypal

3.2. Mis Apuestas

En la opción del menú “Mis Apuestas” podemos encontrar el historial de cada usuario referente a las apuestas realizadas, ya sean por partido o fecha. Aquí el usuario podrá visualizar estado de la apuesta, resultados, y comprobar si ha ganado. Una vez finalizado el partido ECApuestas puede tardar unas horas para hacer efectivo el pago a la cuenta de Paypal de cada ganador Fig. 9. Las apuestas por fecha tienen un botón “Ver Detalle” que abre una ventana con las apuestas realizadas en la fecha consultada Fig. 10

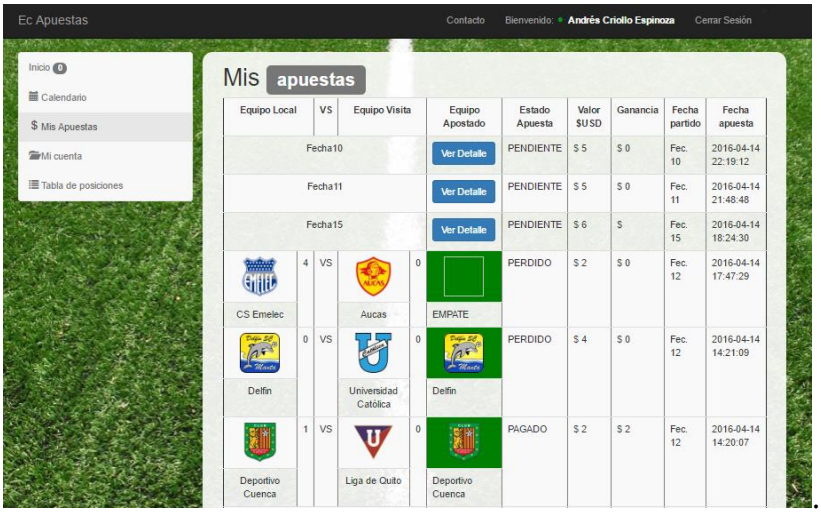


Fig. 9. Historial de apuestas del usuario en EC Apuestas

Equipo Local	VS	Equipo Visita	Equipo Apostado	Estado Apuesta	Valor $USD	Ganancia	Fecha partido	Fecha apuesta
2 Fuerza Amarilla	VS	0 Independiente del Valle	Independiente del Valle	PENDIENTE	$ 6	$	Fec. 15	2016-04-14 18:24:30
2 Universidad CatÃ'lica	VS	3 CD El Nacional	CD El Nacional	PENDIENTE	$ 6	$	Fec. 15	2016-04-14 18:24:30
1 Aucas	VS	2 Deportivo Cuenca	Deportivo Cuenca	PENDIENTE	$ 6	$	Fec. 15	2016-04-14 18:24:30
2 Mushuc Runa	VS	2 Liga de Quito	Liga de Quito	PENDIENTE	$ 6	$	Fec. 15	2016-04-14 18:24:30
5 Barcelona Guayaquil	VS	0 CS Emelec	EMPATE	PENDIENTE	$ 6	$	Fec. 15	2016-04-14 18:24:30
1 River Plate Ecuador	VS	2 Delfín	River Plate Ecuador	PENDIENTE	$ 6	$	Fec. 15	2016-04-14 18:24:30

Fig. 10. Revisión de una apuesta por fecha en concreto

3.3. Mi cuenta

Aquí encontraremos los datos del usuario con la información obtenida de la API de Facebook, datos a los que ECApuestas puede acceder por consentimiento de cada usuario al momento del registro. Así como también, un marcador que indica la cantidad de apuestas realizadas, número de apuestas perdidas y ganadas.

Fig. 11. Información de cuenta de usuario

3.4. Tabla de posiciones

Aquí el usuario encontrará in widget donde se muestra la tabla de posiciones en la actualidad, ayudando al usuario a tomar la mejor decisión al momento de apostar.

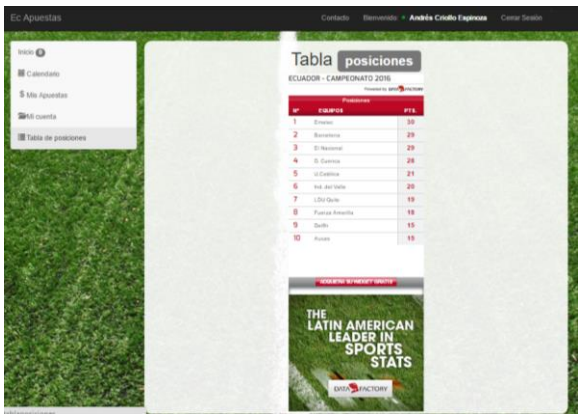


Fig. 12. Tabla de posiciones del campeonato de futbol