[CAPITULO I – INTRODUCCIóN 1](#_Toc445764119)

[1.1 Antecedentes 1](#_Toc445764120)

[1.2 Definición del Problema 2](#_Toc445764121)

[1.3 Objetivo del Proyecto 2](#_Toc445764122)

[1.3.1 Objetivo General 2](#_Toc445764123)

[1.3.2 Objetivos Específicos 2](#_Toc445764124)

[1.4 Áreas Involucradas 2](#_Toc445764125)

[1.5 Justificación 2](#_Toc445764126)

[1.6 Alcance 3](#_Toc445764127)

[1.7 Metodología de trabajo 3](#_Toc445764128)

[1.7.1 Scrum 3](#_Toc445764129)

[1.7.1.1 Roles 4](#_Toc445764130)

[1.7.1.2 Artefactos 5](#_Toc445764131)

[1.7.1.3 Eventos o ceremonias 6](#_Toc445764132)

[CAPITULO II – SISTEMA WEB 8](#_Toc445764133)

[2.1 Sistema web 8](#_Toc445764134)

[2.1.1 Sistemas web y su diferencia con las aplicaciones de escritorio 9](#_Toc445764135)

[2.1.2 Ventajas 9](#_Toc445764136)

[2.1.3 Desventajas 10](#_Toc445764137)

[2.2 Arquitecturas Cliente/Servidor 10](#_Toc445764138)

[2.2.1 Separación de funciones 11](#_Toc445764139)

[2.2.2 Modelos de distribución en aplicaciones cliente / servidor 12](#_Toc445764140)

[2.2.2.1 Presentación distribuida 12](#_Toc445764141)

[2.2.2.2 Aplicación distribuida 12](#_Toc445764142)

[2.2.2.3 Datos distribuidos 12](#_Toc445764143)

[2.2.3 Arquitecturas de dos y tres niveles 12](#_Toc445764144)

[2.3 Elementos de un sistema web 12](#_Toc445764145)

[2.3.1 El cliente 12](#_Toc445764146)

[2.3.2 El Servidor 13](#_Toc445764147)

[2.3.3 Protocolo HTTP 13](#_Toc445764148)

[2.4 Transferencia de páginas web 13](#_Toc445764149)

[2.5 Entornos web 14](#_Toc445764150)

[2.5.1 Internet 14](#_Toc445764151)

[2.5.2 Intranet 14](#_Toc445764152)

[2.5.3 Extranet 14](#_Toc445764153)

[2.6 HTML Hypertext Transfer Protocol 15](#_Toc445764154)

[2.7 Cloud Computing 15](#_Toc445764155)

[2.7.1 Historia 15](#_Toc445764156)

[2.7.2 Características 16](#_Toc445764157)

[2.7.3 Arquitectura de la nube 16](#_Toc445764158)

[2.7.3.2 Plataforma como Servicio (PaaS) 16](#_Toc445764159)

[2.7.3.3 Infraestructura como Servicio (IaaS) 17](#_Toc445764160)

[2.7.4 Ventajas 17](#_Toc445764161)

[2.7.5 Desventajas 18](#_Toc445764162)

[2.8 Software como Servicio (SaaS) 18](#_Toc445764163)

[2.8.1 Características 19](#_Toc445764164)

[2.8.2 Aplicaciones de una sola página (SPA) 19](#_Toc445764165)

[2.8.2.1 Estructura de un SPA 19](#_Toc445764166)

[2.8.2.2 Comunicación con el Servidor 21](#_Toc445764167)

[2.8.3 Diseño web Adaptable 22](#_Toc445764168)

[CAPITULO III – TECNOLOGÍAS UTILIZADAS 23](#_Toc445764169)

[3.1 Plataforma de desarrollo 23](#_Toc445764170)

[3.1.1 Framework Angular JS 23](#_Toc445764171)

[3.1.1.1 ¿Que es Angular? 23](#_Toc445764172)

[3.1.1.2 Características 24](#_Toc445764173)

[3.1.2 Framework CodeIgniter REST-Server 24](#_Toc445764174)

[3.1.2.1 ¿Que es CodeIgniter? 25](#_Toc445764175)

[3.1.2.2 Características 25](#_Toc445764176)

[3.1.3 Framework Twitter Bootstrap 26](#_Toc445764177)

[3.1.3.1 ¿Que es Twitter Bootstrap? 26](#_Toc445764178)

[3.1.2.2 Características 26](#_Toc445764179)

[3.1.4 JavaScript 27](#_Toc445764180)

[3.1.4.1 Características 27](#_Toc445764181)

[3.1.5 PHP 27](#_Toc445764182)

[3.1.5.1 Características 28](#_Toc445764183)

[3.1.6 MYSQL 28](#_Toc445764184)

[3.1.6.1 Características 28](#_Toc445764185)

[3.1.7 Apache 29](#_Toc445764186)

[3.1.7.1 Características 29](#_Toc445764187)

[3.1.8 Plataforma de producción 29](#_Toc445764188)

[CAPITULO IV – ÁREA DE APLICACIÓN 30](#_Toc445764189)

[4.1 Instalación deportiva 30](#_Toc445764190)

[4.1.1 Espacio deportivo 31](#_Toc445764191)

[4.1.2 Complejo deportivo 31](#_Toc445764192)

[4.1.3 Espacios complementarios 31](#_Toc445764193)

[4.1.4 Servicios auxiliares 31](#_Toc445764194)

[4.1.5 Cancha 31](#_Toc445764195)

[4.1.6 Clasificación 31](#_Toc445764196)

[4.1.6.1 Espacios deportivos convencionales 31](#_Toc445764197)

[4.1.6.2 Espacios deportivos singulares 32](#_Toc445764198)

[4.1.6.3 Áreas de actividad deportiva 32](#_Toc445764199)

[4.2 Administración de Horarios y Reservas 33](#_Toc445764200)

[4.2.1 Administración de Horarios 33](#_Toc445764201)

[4.2.2 Reservas 36](#_Toc445764202)

[4.2.2.1 Reservas regulares 36](#_Toc445764203)

[4.2.2.2 Pre reservas 38](#_Toc445764204)

[4.2.2.3 Reservas procedentes de días de mantenimiento 38](#_Toc445764205)

[4.2.2.4 Reservas periódicas 40](#_Toc445764206)

[CAPITULO V – PILA DEL PRODUCTO 41](#_Toc445764207)

[5.1 Pila del producto 41](#_Toc445764208)

[5.2 Planificación del proyecto 51](#_Toc445764209)

[5.2.1 Identificación de actividades 51](#_Toc445764210)

[5.2.2 Estimación de costos 52](#_Toc445764211)

[5.2.2.1 Factor de ajuste 52](#_Toc445764212)

[5.2.2.2 Puntos de Función 53](#_Toc445764213)

[5.2.2.3 Esfuerzo, tiempo y número de personas 53](#_Toc445764214)

[Líneas de Código 54](#_Toc445764215)

[Esfuerzo 54](#_Toc445764216)

[Duración 54](#_Toc445764217)

[Personal 54](#_Toc445764218)

[5.2.2.4 Determinación del costo 54](#_Toc445764219)

[5.3.3 Planificación de entregables de incrementos de las iteraciones (Release Plan) 55](#_Toc445764220)

[CAPITULO VI – PRIMERA ITERACÍON 56](#_Toc445764221)

[6.1 Planificación de la iteración 56](#_Toc445764222)

[6.1.1 Pila de la Iteración 56](#_Toc445764223)

[6.1.2 Estimación de esfuerzo 56](#_Toc445764224)

[6.1.3 Definición de tareas 56](#_Toc445764225)

[6.3 Diseño 57](#_Toc445764226)

[6.5 Pruebas 57](#_Toc445764227)

[6.6 Revisión de la Iteración 59](#_Toc445764228)

[6.6.1 Grafico BurnDown 60](#_Toc445764229)

[CAPITULO VII – SEGUNDA ITERACÍON 61](#_Toc445764230)

[6.1 Planificación de la iteración 61](#_Toc445764231)

[6.1.1 Pila de la Iteración 61](#_Toc445764232)

[6.1.2 Estimación de esfuerzo 61](#_Toc445764233)

[6.1.3 Definición de tareas 61](#_Toc445764234)

[6.3 Diseño 62](#_Toc445764235)

[6.5 Pruebas 63](#_Toc445764236)

[6.6 Revisión de la Iteración 64](#_Toc445764237)

[6.6.1 Grafico BurnDown 65](#_Toc445764238)

[CAPITULO VIII – TERCERA ITERACÍON 67](#_Toc445764239)

[6.1 Planificación de la iteración 67](#_Toc445764240)

[6.1.1 Pila de la Iteración 67](#_Toc445764241)

[6.1.2 Estimación de esfuerzo 67](#_Toc445764242)

[6.1.3 Definición de tareas 68](#_Toc445764243)

[6.3 Diseño 68](#_Toc445764244)

[6.5 Pruebas 69](#_Toc445764245)

[6.6 Revisión de la Iteración 70](#_Toc445764246)

[6.6.1 Grafico BurnDown 72](#_Toc445764247)

[CAPITULO IX – CUARTA ITERACÍON 74](#_Toc445764248)

[6.1 Planificación de la iteración 74](#_Toc445764249)

[6.1.1 Pila de la Iteración 74](#_Toc445764250)

[6.1.2 Estimación de esfuerzo 74](#_Toc445764251)

[6.1.3 Definición de tareas 74](#_Toc445764252)

[6.3 Diseño 75](#_Toc445764253)

[6.5 Pruebas 76](#_Toc445764254)

[6.6 Revisión de la Iteración 76](#_Toc445764255)

[6.6.1 Grafico BurnDown 77](#_Toc445764256)

[CAPITULO X – PRUEBAS DE VALIDACION E IMPLANTACION 79](#_Toc445764257)

[10.1 Implantación 79](#_Toc445764258)

[10.1.1 Herramientas para el desplegado de la aplicación 79](#_Toc445764259)

[10.1.2 Servidor de producción 79](#_Toc445764260)

[10.1.2 Características del servidor de producción 79](#_Toc445764261)

[10.2 Pruebas de validación 79](#_Toc445764262)

[CAPITULO XI – CONCLUSIONES Y RECOMENDACIONES 80](#_Toc445764263)

[Bibliografía 82](#_Toc445764264)

# CAPITULO I – INTRODUCCIóN

## 1.1 Antecedentes

El problema de asignación y reservas de canchas en los complejos deportivos, asociaciones y clubes es bastante complejo, ya que se busca determinar que canchas de cada disciplina deben asignarse a que clientes en qué fecha y varios aspectos que determinan los diferentes complejos deportivos, además la gran cantidad de información con que cuentan para realizar reservas.

Actualmente la ciudad de Cochabamba cuenta con un número importante de áreas deportivas entre complejos, asociaciones y pequeñas empresas que se dedican al flete de canchas. Estas áreas deportivas ofrecen diferentes disciplinas, precios de flete y calidad.

Los sistemas de administración de horarios tienen como misión fundamental ayudar a planificar horarios de forma precisa, ahorrar tiempo en la elaboración de los mismos y reducir las posibilidades de error que podrían presentarse cuando las personas intentan hacerlo por sí mismos.

Los sistemas de administración de horarios son considerados como una buena herramienta para la toma de decisiones oportunas, confiables y efectivas en cuanto a técnicas de planificación con el fin de garantizar el éxito, limitar el riesgo, reducir costos y aumentar las ganancias. De la misma forma estos sistemas proporcionan una variedad considerable de reportes para la empresa, almacenando grandes cantidades de información.

Los sistemas web o también conocidos como “Aplicaciones web” son aquellos que se alojan en servidores conectados a internet o una intranet (red local) que permiten acceder a ellos desde cualquier tipo de navegador. Su aspecto es muy similar a las páginas web que vemos normalmente pero en realidad los sistemas web tienen funcionalidades muy potentes que brindan respuestas a casos particulares.

Por otra parte, los sistemas web se han convertido en un factor importante para la vida de una empresa, sea grande mediana o pequeña debido a que proporciona un medio libre de publicidad las 24 horas del día, que da como resultado el crecimiento de clientes y la buena imagen de la empresa. Asimismo llegan a reducir costos operativos.

El Cloud computing consiste en la posibilidad de ofrecer servicios a través de internet, de esta manera se puede tener todos los archivos e información en internet sin depender de la capacidad suficiente de almacenar información. Es una nueva forma de negocio más conocidos como e-buisness o negocios por internet .

Una aplicación de Cloud computing permite a los usuarios contar con la información en servidores con conexión a internet, de tal forma que el usuario solo necesita un navegador para poder acceder a dicha información, sin importar donde se encuentre.

## 1.2 Definición del Problema

Deficiente manejo de información de reservas de canchas deportivas debido a la carencia de una administración precisa, óptima y confiable de horarios.

## 1.3 Objetivo del Proyecto

De acuerdo a la definición del problema del presente trabajo de grado, se ha planteado el siguiente objetivo general.

### 1.3.1 Objetivo General

Mejorar el manejo de información de la administración de horarios y reservas de las canchas deportivas, a través del desarrollo de un sistema web para la administración de horarios y reservas para complejos, usando cloud computing.

### 1.3.2 Objetivos Específicos

A continuacion se presentan los objetivos especificos.

* Estudiar los procesos que comprenden en la administración de horarios y reservas de los complejos deportivos.
* Identificar las necesidades y requerimientos de los usuarios.
* Llevar a cabo la Implementación de la administración de horarios de complejos deportivos.
* Llevar a cabo la implementación de la administración de reservas.
* Permitir a los deportistas poder encontrar canchas a través del sistema web de planificación.
* Poner en productivo el sistema web en un servidor de la nube.

## 1.4 Áreas Involucradas

* Sistemas de información.
* Programación web.

## 1.5 Justificación

La revolución tecnológica que actualmente estamos viviendo, bien podría ser la más profunda de nuestra historia. Los servicios convergen y pasan del mundo físico al mundo digital, siendo accesibles desde cualquier dispositivo. Un hecho relevante es que nuestros datos ya no residen en nuestros ordenadores sino en una Internet Global que adquiere entidad propia y se convierte en mucho más que una simple infraestructura de conexión: es la plataforma que ofrece servicio a millones de dispositivos inteligentes conectados a la red (Torres, 2011).

Es lo que se conoce como Cloud Computing o informática en la nube de Internet, que permite que los consumidores, empresas o particulares, no se tengan que preocupar de cómo se provee el servicio que necesitan. Las empresas no podrán evitar este cambio si no quieren perder el tren del avance tecnológico, y esto implicará tomar decisiones sobre la dirección a seguir para mejorar sus negocios (Torres, 2011).

El sistema web a desarrollar, permitirá que los complejos cuenten con la información de horarios de manera más eficiente y rápida, de tal forma que optimizara el tiempo del proceso de reservas. Además, existiendo el riesgo de cometer errores en la planificación de reservas, el sistema las evitara de manera automática, eliminando la posibilidad de fallas.

Ya que en la actualidad contar con una página en internet representa una ventaja para las empresas, sin importar el tamaño de las mismas, un sistema de planificación de horarios y reservas vía web permitirá a los complejos deportivos contar con publicidad las 24 horas del día, beneficiando así a los complejos que no cuenten con estrategias de marketing.

En los países más desarrollados, está demostrado que la intensidad en la aplicación de nuevas tecnologías y la incorporación de un Sitio Web, está correlacionado de manera positiva con el incremento de las ventas, la productividad y el valor de mercado de las empresas. No importa el tamaño de la empresa, es de suma importancia el poder alcanzar a sus clientes de una manera masiva y sencilla para ellos.

Si bien es cierto que existen muchas redes sociales en las que te puedes anunciar de forma gratuita o pagada, el tener un Sitio Web eleva el prestigio propio de la empresa y permite a tus clientes incrementar el nivel de confianza hacia tu producto o servicio. Las redes sociales muestran resultados aleatorios que no siempre van de la mano con el comportamiento del consumidor final.

Dado que el sistema se encontrara en servidores web y disponible para su acceso, las personas podrán ver la disponibilidad de las canchas sin necesidad de ir en persona al complejo deportivo, representando así un beneficio para los deportistas ya que contaran con la información de canchas y horarios de varios complejos a través de internet.

## 1.6 Alcance

* La aplicación será desarrollada para que funcione vía web.
* Para el desarrollo del proyecto se hará uso de la metodología scrum, con las adaptaciones necesarias dado que el proyecto será desarrollado por una persona.
* El sistema funcionara en los navegadores de Google Chrome v44, Firefox v39 y versiones superiores.
* El sistema podrá visualizarse en los sistemas operativos de Windows 7 y versiones superiores.
* La cancelación de reservas no se las realizara vía internet, debido a que se debe pagar una licencia y a los bajos costos que algunos complejos deportivos piden por reservar una cancha.

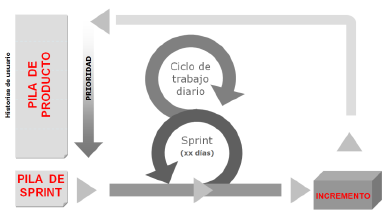
## 1.7 Metodología de trabajo

La metodología de desarrollo que se aplicara para la implementación del presente trabajo de grado, es la metodología scrum, con las adaptaciones necesarias, ya que scrum es una metodología que se aplica en equipos de desarrollo. A continuación, se hace la descripción de la metodología de trabajo que será usado.

### 1.7.1 Scrum

Scrum es un modelo de desarrollo ágil caracterizado por:

* Adoptar una estrategia de desarrollo incremental, en lugar de la planificación y ejecución completa del producto.
* Solapamiento de las diferentes fases del desarrollo, en lugar de realizar una tras otra en un ciclo secuencial o de cascada.



*Figura 1.1: Desarrollo de software con la Metodología Scrum (Palacio, 2014)*

El marco técnico de scrum está formado por: roles, artefactos y eventos o ceremonias.

#### 1.7.1.1 Roles

Un equipo Scrum tiene tres funciones: Propietario del producto, Scrum Master, y miembro del equipo.

* El propietario del producto es responsable por el valor empresarial de los productos y decide lo que se construye.
* El Scrum Master es principalmente un facilitador y ejecutor de las reglas de Scrum.
* El equipo es auto-organización y multi-funcional. El equipo decide cómo construir lo que el propietario del producto quiere y se va haciendo sobre él.

Obviamente, puesto que el trabajo de grado lo desarrolla una sola persona se va a hacer uso de los siguientes roles.

**El dueño del producto**: Esta función va a ser desempeñada por el Licenciado Valentin Laime Zapata, que figura como tutor del presente trabajo de grado.

El propietario del producto (*product owner)* es quien toma las decisiones del cliente. Su responsabilidad es el valor del producto.

Para simplificar la comunicación y toma de decisiones es necesario que este rol recaiga en una única persona.

Si el cliente es una organización grande, o con varios departamentos, puede adoptar la forma de comunicación interna que consideren oportuna, pero en el equipo de desarrollo sólo se integra una persona en representación del cliente, y ésta debe tener el conocimiento suficiente del producto y las atribuciones necesarias para tomar las decisiones que le corresponden.

En resumen, el propietario de producto es quien:

* Decide en última instancia cómo será el resultado final, y el orden en el que se van construyendo los sucesivos incrementos: qué se pone y qué se quita de la pila del producto, y cuál es la prioridad de las funcionalidades.
* Conoce el plan del producto, sus posibilidades y plan de inversión, así como del retorno esperado a la inversión realizada, y se responsabiliza sobre fechas y funcionalidades de las diferentes versiones del mismo.

Para ejercer este rol es necesario:

* ***Conocer perfectamente el entorno de negocio del cliente***, las necesidades y el objetivo que se persigue con el sistema que se está construyendo.
* Tener la ***visión del producto***, así como las necesidades concretas del proyecto, para poder priorizar eficientemente el trabajo.

**El desarrollador**: Esta función va a ser llevada a cabo por el autor del presente trabajo de grado.

El desarrollador es el que realizan un trabajo, con una asignación específica de tareas, responsabilidades y siguiendo un proceso o pautas de ejecución.

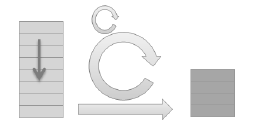
#### 1.7.1.2 Artefactos

Scrum cuenta con los siguientes artefactos: pila del producto, pila del sprint, el incremento y la iteración o sprint.

**Sprint**: Se denomina sprint a cada ciclo o iteración de trabajo que produce una parte del producto terminada y funcionalmente operativa (***incremento***). Comúnmente dura entre 2 a 4 semanas.

**Incremento**: El incremento es la parte de producto producida en un sprint, y tiene como característica el estar completamente terminada y operativa, en condiciones de ser entregada al cliente.

No se deben considerar como Incremento a prototipos, módulos o sub-módulos, ni partes pendientes de pruebas o integración.



*Figura 1.2: Incremento de iteración (Palacio, 2014)*

**Pila del producto**: (product backlog) lista de requisitos de usuario, que a partir de la visión inicial del producto crece y evoluciona durante el desarrollo. La pila del producto es el inventario de funcionalidades, mejoras, tecnología y corrección de errores que deben incorporarse al producto a través de las sucesivas iteraciones.

Representa todo aquello que esperan el cliente, los usuarios, y en general los interesados. Todo lo que suponga un trabajo que debe realizar el equipo debe estar reflejado en esta pila.

La pila de requisitos del producto nunca se da por completada; está en continuo crecimiento y evolución. Al comenzar el proyecto incluye los requisitos inicialmente conocidos y mejor entendidos, y conforme avanza el desarrollo, y evoluciona el entorno en el que será usado, se va desarrollando.

En definitiva su continuo dinamismo refleja aquello que el producto necesita incorporar para ser el más adecuado a las circunstancias, en todo momento.

Para comenzar el desarrollo se necesita la visión del objetivo de negocio que se quiere conseguir con el proyecto, comprendida y conocida por todo el equipo, y elementos suficientes en la pila para llevar a cabo el primer sprint.

**Pila del sprint**: La pila del sprint (*sprint Backlog*) es la lista que descompone las funcionalidades de la pila del producto (historias de usuario) en las tareas necesarias para construir un incremento: una parte completa y operativa del producto.

Se la realiza durante la planificación del sprint, auto asignando cada tarea a un miembro del equipo, e indicando en la misma lista cuánto tiempo o esfuerzo se prevé que falta para terminarla.

Son soportes habituales:

* Tablero físico o pared.
* Hoja de cálculo.
* Herramienta colaborativa o de gestión de proyectos.

Para llevar el control de la pila de la iteración, se va hacer uso de la herramienta “**pivotaltracker**”.

#### 1.7.1.3 Eventos o ceremonias

Los eventos que van a ser tomados en cuenta son: Planificación de las iteraciones y revisiones de las iteraciones.

**Planificación del sprint**: En esta reunión se toman como base las prioridades y necesidades de negocio del cliente, y se determinan cuáles y cómo van a ser las funcionalidades que se incorporarán al producto en el siguiente sprint.

Se trata de una reunión conducida por el responsable del funcionamiento del marco scrum (Scrum Master en scrum técnico) a la que deben asistir el propietario del producto y el equipo completo, y a la que también pueden asistir otros implicados en el proyecto.

La reunión puede durar una jornada de trabajo completa, cuando se trata de planificar un sprint largo (de un mes de duración) o un tiempo proporcional para planificar un sprint más breve.

Esta reunión debe dar respuesta a dos cuestiones:

* Qué se entregará al terminar el sprint.
* Cuál es el trabajo necesario para realizar el incremento previsto, y cómo lo llevará a cabo el equipo.
* Esta planificación debe dar como resultado lo siguiente:
* Pila del sprint.
* Duración del sprint y fecha de la reunión de revisión.
* Objetivo del sprint.

**Revisión de la iteración:** Reunión realizada al final del sprint para comprobar el incremento.

No debe durar más de 4 horas, en el caso de revisar sprints largos. Para sprints de una o dos semanas, con una o dos horas de duración debería ser suficiente.

Los objetivos de la revisión de la iteración son:

* El propietario del producto comprueba el progreso del sistema. Esta reunión marca, a intervalos regulares, el ritmo de construcción, y la trayectoria que va tomando la visión del producto.
* El propietario del producto identifica las funcionalidades que se pueden considerar “hechas” y las que no.
* Al ver y probar el incremento, el propietario del producto, y el equipo en general obtienen retroalimentación relevante para revisar la pila del producto.
* Otros ingenieros y programadores de la empresa también pueden asistir para conocer cómo trabaja la tecnología empleada.

Las precondiciones que deben cumplirse para realizar la revisión de la iteracion son:

* Se ha concluido el sprint.
* El incremento ha sido terminado
* Asiste todo el equipo de desarrollo, el propietario del producto, y todas las personas implicadas en el proyecto que lo deseen.

Es una reunión informativa. **Su misión no es la toma de decisiones ni la crítica del incremento**. Con la información obtenida, posteriormente el propietario del producto tratará las posibles modificaciones sobre la visión del producto.

Protocolo recomendado para la revisión de la iteración:

1.- El desarrollador expone el objetivo del sprint, la lista de funcionalidades que se incluían y las que se han desarrollado.

2.- El desarrollador hace una introducción general del sprint y demuestra el funcionamiento de las partes construidas.

3.- Se abre un turno de preguntas y sugerencias. Esta parte genera información valiosa para que el propietario del producto y el equipo en general, puedan mejorar la visión del producto.

4.- El desarrollador, de acuerdo con las agendas del propietario del producto y el desarrollador, cierra la fecha para la reunión de preparación del siguiente sprint.

# CAPITULO II – SISTEMA WEB

Este capítulo está dedicado a realizar una descripción completa de los elementos conceptuales que se manejan sobre sistemas web. Asimismo, se detallan las características principales de cada elemento. Una vez conceptualizado dichos elementos, se abordara el tema de los servicios en la nube, especialmente los servicios SAAS y una de las formas de implementarlas. Finalmente se describe acerca de las aplicaciones de una sola página o SPA y el diseño web adaptable.

## Sistema web

Un sistema web o también denominado aplicación web según Lujan (2002), es: “Una aplicación web (web-based application) es un tipo especial de aplicación cliente/servidor, donde tanto el cliente (el navegador, explorador o visualizador) como el servidor (el servidor web) y el protocolo mediante el que se comunican (HyperText Transfer Protocol (HTTP)) están estandarizados y no han de ser creados por el programador de aplicaciones”.

Estos tipos de aplicaciones son populares debido a lo práctico del navegador web, a la independencia del sistema operativo, así como a la facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software a miles de usuarios potenciales. Existen aplicaciones como los webmails, wikis, weblogs, tiendas en línea, etc. que son ejemplos bastante conocidos de aplicaciones web.

Una aplicación web contiene elementos que permiten una comunicación activa entre el usuario y la información. Esto permite que el usuario acceda a los datos de modo interactivo, gracias a que la aplicación responderá a cada una de sus acciones, como por ejemplo rellenar y enviar formularios, participar en juegos diversos y acceder a gestores de base de datos de todo tipo.

En vista de que algunas veces existen confusiones sobre páginas web estáticas y dinámicas, a continuación se hace una diferencia entre ellas.

Una **página web estática** tiene las siguientes propiedades:

* Realizadas principalmente en XHTML o HTML.
* Para modificación de contenidos es imprescindible acceder al servidor donde está alojada la página.
* El proceso de actualización es lento y manual.
* No ofrece bases de datos, interacción, etc.
* El usuario no puede seleccionar, ordenar o modificar los contenidos o el diseño de la página.
* Su diseño y desarrollo es más rápido y económico que el de una web dinámica.

Una **página web dinámica** tiene las siguientes propiedades:

* Muchas posibilidades en diseño y desarrollo.
* Utiliza varios lenguajes y técnicas de programación en su desarrollo.
* El proceso de actualización es muy sencillo y no requiere entrar en el servidor.
* Permite un gran número de funcionalidades tales como bases de datos, foros, contenido dinámico, etc.
* El usuario puede alterar el diseño, contenidos o presentación de la página a su gusto.
* Su diseño y desarrollo es más caro que el de una web estática, por su mayor complejidad y sus mayores prestaciones.

Por las características listadas anteriormente, se concluye que una aplicación web o sistema web es un tipo de página web dinámica, ya que concuerda con dichas características. Además, se considera que es un “sistema de información”, porque funciona como cualquier otra aplicación de escritorio, pudiendo almacenar grandes cantidades de información.

### Sistemas web y su diferencia con las aplicaciones de escritorio

En las aplicaciones de escritorio cada cliente debe instalar por separado el software. El cliente realiza peticiones a otro programa (el servidor) que le da respuesta. Una mejora en el servidor, como parte de la aplicación, requiere normalmente una mejora de los clientes instalados en cada computadora personal, añadiendo un coste de soporte técnico y disminuyendo la productividad.

A diferencia de lo anterior, las aplicaciones web generan dinámicamente una serie de páginas en un formato estándar, como HTML que es soportado por los navegadores web comunes. Se utilizan lenguajes interpretados en el lado del cliente, directamente o a través de extensiones tales como JavaScript, Java, Flash, etc., para añadir elementos dinámicos a la interfaz de usuario.

Generalmente cada página web en particular se envía al cliente como un documento estático, pero la secuencia de páginas ofrece al usuario una experiencia interactiva. Durante la sesión, el navegador web interpreta y muestra en pantalla las páginas, actuando como cliente para cualquier aplicación web.

### 2.1.2 Ventajas

A continuación, se muestra una lista de las ventajas que ofrecen este tipo de sistemas de información.

* **Ahorra tiempo:** se pueden realizar tareas sencillas sin necesidad de descargar ni instalar ningún programa.
* **No hay problemas de compatibilidad**: basta tener un navegador actualizado para poder utilizarlas.
* **No ocupan espacio en nuestro disco duro.**
* **Actualizaciones inmediatas**: como el software lo gestiona el propio desarrollador, cuando nos conectamos estamos usando siempre la última versión que haya lanzado.
* **Consumo de recursos bajo**: dado que toda (o gran parte) de la aplicación no se encuentra en nuestra computadora, muchas de las tareas que realiza el software no consumen recursos nuestros porque se realizan desde otra computadora.
* **Multiplataforma:** se pueden usar desde cualquier sistema operativo porque solamente es necesario tener un navegador.
* **Portables:** es independiente de la computadora donde se utilice (PC de sobremesa, portátil) porque se accede a través de una página web (solamente es necesario disponer de acceso a Internet). La reciente tendencia al acceso a las aplicaciones web a través de teléfonos móviles requiere sin embargo un diseño específico de los ficheros CSS para no dificultar el acceso de estos usuarios.
* **La disponibilidad** suele ser alta porque el servicio se ofrece desde múltiples localizaciones para asegurar la continuidad del mismo.
* **Los virus no dañan los datos** porque están guardados en el servidor de la aplicación.
* **Colaboración:** gracias a que el acceso al servicio se realiza desde una única ubicación es sencillo el acceso y compartición de datos por parte de varios usuarios. Tiene mucho sentido, por ejemplo, en aplicaciones en línea de calendarios u oficina.
* **Los navegadores ofrecen cada vez más y mejores funcionalidades** para crear "aplicaciones web enriquecidas" (Rich Internet application o RIA).

### 2.1.3 Desventajas

* **Habitualmente ofrecen menos funcionalidades que las aplicaciones de escritorio**. Se debe a que las funcionalidades que se pueden realizar desde un navegador son más limitadas que las que se pueden realizar desde el sistema operativo.
* **La disponibilidad depende de un tercero**. El proveedor de la conexión a internet o el que provee el enlace entre el servidor de la aplicación y el cliente. Así que la disponibilidad del servicio está supeditada al proveedor.

### Arquitecturas Cliente/Servidor

Un sistema web, tradicionalmente funciona bajo una arquitectura Cliente / Servidor, ya que se despliega en una red de computadoras. Según Lujan (2002), esta arquitectura se define de la siguiente manera.

“Cliente/servidor es una arquitectura de red en la que cada ordenador o proceso en la red es cliente o servidor. Normalmente, los servidores son ordenadores potentes dedicados a gestionar unidades de disco (servidor de ficheros), impresoras (servidor de impresoras), tráfico de red (servidor de red), datos (Servidor de bases de datos) o incluso aplicaciones (servidor de aplicaciones), mientras que los clientes son máquinas menos potentes y usan los recursos que ofrecen los servidores”.

Esta arquitectura implica la existencia de una relación entre procesos que solicitan servicios (clientes) y procesos que responden a estos servicios (servidores). Estos dos tipos de procesos pueden ejecutarse en el mismo procesador o en distintos.



*Figura 2.1: Arquitectura Cliente / Servidor (Elaboración propia, 2016)*

### Separación de funciones

La arquitectura cliente/servidor nos permite la separación de funciones en tres niveles.

* **Lógica de presentación.** La presentación de los datos es una función independiente del resto.
* **Lógica de negocio (o aplicación).** Los flujos de trabajo pueden cambiarse según las necesidades existentes de un procesador a otro.
* **Lógica de datos.** La gestión de los datos debe ser independiente para poder ser distribuida según las necesidades de la empresa en cada momento.

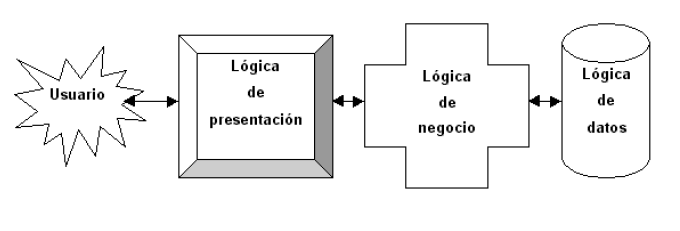


Figura 2.2: Separación de funciones en arquitectura cliente servidor (Lujan, 2002)

### 2.2.2 Modelos de distribución en aplicaciones cliente / servidor

Según como se distribuyan las tres funciones básicas de una aplicación (presentación, negocio y datos) entre el cliente y el servidor, podemos contemplar tres modelos: presentación distribuida, aplicación distribuida, datos distribuidos.

#### 2.2.2.1 Presentación distribuida

El cliente sólo mantiene la presentación, el resto de la aplicación se ejecuta remotamente. La presentación distribuida, en su forma más simple, es una interfaz gráfica de usuario, a la que se le pueden acoplar controles de validación de datos, para evitar la validación de los mismos en el servidor.

#### 2.2.2.2 Aplicación distribuida

Es el modelo que proporciona máxima flexibilidad, puesto que permite tanto a servidor como a cliente mantener la lógica de negocio realizando cada uno las funciones que le sean más propias, bien por organización, o bien por mejora en el rendimiento del sistema.

#### 2.2.2.3 Datos distribuidos

Los datos son los que se distribuyen, por lo que la lógica de datos es lo que queda separada del resto de la aplicación. Se puede dar de dos formas: ficheros distribuidos o bases de datos distribuidas.

### 2.2.3 Arquitecturas de dos y tres niveles

Una arquitectura de dos niveles está basada en un sistema gestor de bases de datos donde el cliente mantiene la lógica de la presentación, negocio, y de acceso a los datos, y el servidor únicamente gestiona los datos. Suelen ser aplicaciones cerradas que supeditan la lógica de los procesos cliente al gestor de base de datos que se está usando.

En las arquitecturas de tres niveles, la lógica de presentación, la lógica de negocio y la lógica de datos están separadas, de tal forma que mientras la lógica de presentación se ejecutará normalmente en la estación cliente, la lógica de negocio y la de datos pueden estar repartidas entre distintos procesadores.

El objetivo de aumentar el número de niveles en una aplicación distribuida es lograr una mayor independencia entre un nivel y otro, lo que facilita la portabilidad en entornos heterogéneos.

### Elementos de un sistema web

Los elementos más importantes de una aplicación web son: El cliente, el servidor y el protocolo http.

### El cliente

El cliente web es un programa con el que interacciona el usuario para solicitar a un servidor web el envío de los recursos que desea obtener mediante HTTP. La parte cliente de las aplicaciones web suele estar formada por el código HyperText Markup Language (HTML) que forma la página web más algo de código ejecutable realizado en lenguaje de script del navegador (JavaScript o VBScript) o mediante pequeños programas (applets) realizados en Java. También se suelen emplear plug-ins3 que permiten visualizar otros contenidos multimedia (como Flash4), aunque no se encuentran tan extendidos como las tecnologías anteriores y plantean problemas de incompatibilidad entre distintas plataformas. Por tanto, la misión del cliente web es interpretar las páginas HTML y los diferentes recursos que contienen (imágenes, sonidos, etc.).

### El Servidor

El servidor web es un programa que está esperando permanentemente las solicitudes de conexión mediante el protocolo HTTP por parte de los clientes web. En los sistemas Unix suele ser un demonio y en los sistemas Microsoft Windows un servicio.

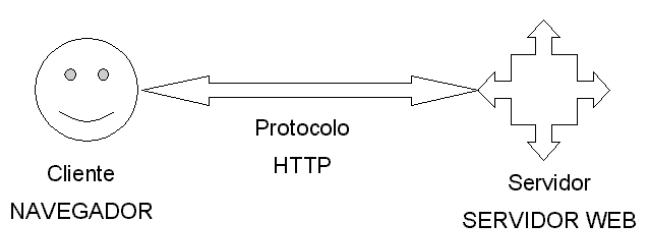
La parte servidor de las aplicaciones web está formada por páginas estáticas que siempre muestran el mismo contenido y por programas o scripts que son ejecutados por el servidor web cuando el navegador del cliente solicita algunas páginas dinámicas.

La programación en la parte del servidor se la realiza mediante tecnologías como ASP, ColdFusion, Java Server Pages (JSP), servlets, PHP, Javascript, etc. Todas ellas son muy similares, se basan en los mismos principios y ofrecen resultados equivalentes.

### Protocolo HTTP

El protocolo HTTP (Hypertext Transfer Protocol) Protocolo de transferencia de hipertexto, forma parte de la familia de protocolos de comunicaciones Transmission Control Protocol/Internet Protocol (TCP/IP), que son los empleados en Internet. Estos protocolos permiten la conexión de sistemas heterogéneos, lo que facilita el intercambio de información entre distintos ordenadores.

En la figura 2.3 se puede observar la relación entre los tres elementos de un sistema web.



*Figura 2.3: Relación entre los elementos de un sistema web (Elaboracion propia, 2016)*

## Transferencia de páginas web

El proceso completo, desde que el usuario solicita una página, hasta que el cliente web (navegador) se la muestra con el formato apropiado, es el siguiente:

* El usuario especifica en el cliente web la dirección de la página que desea consultar: el usuario escribe en el navegador la dirección (Universal Resource Locator (URL)) de la página que desea visitar.
* El cliente establece una conexión con el servidor web.
* El cliente solicita la página o el objeto deseado.
* El servidor envía dicha página u objeto (o, si no existe, devuelve un código de error).
* Si se trata de una página HTML, el cliente inicia sus labores de interpretación de los códigos HTML. Si el cliente web encuentra instrucciones que hacen referencia a otros objetos que se tienen que mostrar con la página (imágenes, sonidos, animaciones multimedia, etc.), establece automáticamente comunicación con el servidor web para solicitar dichos objetos.
* Se cierra la conexión entre el cliente y el servidor.
* Se muestra la página al usuario.

Obsérvese que siempre se libera la conexión, por lo que ésta sólo tiene la duración correspondiente a la transmisión de la página solicitada. Esto se hace así para no desperdiciar innecesariamente el ancho de banda de la red mientras el usuario lee la página recibida.

Cuando el usuario activa un enlace de la página, se establece una nueva conexión para recibir otra página o elemento multimedia. Por ello, el usuario tiene la sensación de que está disfrutando de una conexión permanente cuando realmente no es así.

Un detalle importante es que para casa objeto que se transfiere por la red se realiza una conexión independiente. Por ejemplo, si el cliente web solicita una página que contiene dos imágenes integradas, se realizan tres conexiones: una para el documento HTML y dos para los archivos de las imágenes.

## Entornos web

Los sistemas web se emplean en tres entornos informáticos muy similares que suelen confundirse entre sí: Internet, intranet y extranet.

### Internet

Internet es una red global que conecta millones de ordenadores por todo el mundo. Su nacimiento se sitúa en 1969, en un proyecto de investigación del Departamento de Defensa de Estados Unidos. En 1998, la Internet tenía más de 100 millones de usuarios en todo el mundo, en diciembre de 2000 unos 400 millones y el número sigue creciendo rápidamente. Más de 100 países están conectados a este nuevo medio para intercambiar todo tipo de información.

### Intranet

Una intranet es una red de ordenadores basada en los protocolos que gobiernan Internet (TCP/IP) que pertenece a una organización y que es accesible únicamente por los miembros de la organización, empleados u otras personas con autorización.

Una intranet puede estar o no conectada a Internet. Un sitio web en una intranet es y actúa como cualquier otro sitio web, pero los cortafuegos (firewall) lo protegen de accesos no autorizados.

### Extranet

Una extranet es una intranet a la que pueden acceder parcialmente personas autorizadas ajenas a la organización o empresa propietaria de la intranet.

Mientras que una intranet reside detrás de un cortafuego y sólo es accesible por las personas que forman parte de la organización propietaria de la intranet, una extranet proporciona diferentes niveles de acceso a personas que se encuentran en el exterior de la organización.

## HTML Hypertext Transfer Protocol

HTML es un lenguaje de marcas (etiquetas) que se emplea para dar formato a los documentos que se quieren publicar en la WWW. Los navegadores pueden interpretar las etiquetas y muestran los documentos con el formato deseado.

Las páginas web o páginas HTML son unos cheros escritos en el lenguaje HTML. El desarrollo de estas páginas abarca un amplio grupo de tecnologías, desde las páginas más sencillas que sólo usan el lenguaje HTML hasta las más complejas que usan Dynamic HTML (DHTML), JavaScript, applets realizados en Java o componentes ActiveX.

## Cloud Computing

Según la UAM (2011), cloud computing o computacion en la nube es: “Cuando se hace referencia a la nube, se está aludiendo a un termino con algunos años de historia y que es una forma metafórica de nombrar a Internet. Básicamente la computación en la nube consiste en los servicios ofrecidos a través de la red tales como correo electrónico, almacenamiento, uso de aplicaciones, etc., los cuales son normalmente accesibles mediante un navegador web”.

### Historia

El desarrollo de la computación en la nube comenzó a través de grandes empresas de servicios de

Internet como Google y Amazon los cuales construyeron su propia infraestructura. A partir de allí surgió una arquitectura: un sistema de recursos distribuidos de manera horizontal, introducidos como servicios virtuales de tecnologías de información (TI) escalados masivamente y manejados como recursos agrupados y configurados continuamente.

Aunque la implementación es reciente, la idea no es nueva ya que se ha discutido en el medio desde hace algunos años con distintos nombres tales como: “utility computing”, computación en demanda, computación elástica, o “grid computing” (no confundir con el procesamiento en red mencionado anteriormente). Haciendo una comparación de ideas y tecnologías entre las décadas de 1960 y 1970 con la época actual se tiene lo siguiente:

**Década 1960**

* Uso de “terminales tontas” que dependían de un sistema central más potente.
* La información se guardaba en el servidor.
* Se necesitaba conexión constante con el sistema central para funcionar correctamente.
* Imposibilidad de instalar aplicaciones.

**Siglo XXI**

* Terminales poco potentes pero autosuficientes(Netbooks, tablets, smartphones)
* La información se aloja en los servidores del proveedor de servicios aunque hay posibilidad de guardar información en la terminal del usuario.
* Se necesita conexión constante con el sistema central para hacer uso de todos los recursos.
* Dependiendo de la terminal, es posible instalar aplicaciones, aunque la idea es ejecutarlas a través de Internet.

### Características

* No es necesario disponer de un equipo potente, tan solo de un aparato con conexión a internet; esto debido a que el dispositivo del usuario no realizara ningún proceso complejo y los ficheros puede guardarse en la nube. Los servidores en donde se hallan los programas que se utilicen son los encargados de las tareas complicadas que antes se realizaba localmente.
* Con el uso del Cloud Computing no hay necesidad por parte del usuario de conocer la infraestructura detrás de esta, ya que pasa a ser una abstracción, “una nube” donde las aplicaciones y servicios pueden fácilmente crecer, funcionar rápido y con pocas fallas.
* Es auto reparable: En caso de surgir un fallo, el último respaldo (backup) de la aplicación se convierte automáticamente en la copia primaria y a partir de ´esta se genera uno nuevo.
* Posee un alto nivel de seguridad: El sistema esta creado de tal forma que permite a diferentes clientes compartir la infraestructura sin preocuparse de ello y sin comprometer su seguridad y privacidad; de esto se ocupa el sistema proveedor que se encarga de cifrar los datos.
* Disponibilidad de la información: No se hace necesario guardar los documentos editados por el usuario en su computadora o en medios físicos propios ya que la información radicara en Internet permitiendo su acceso desde cualquier dispositivo conectado a la red (con autorización requerida).

### Arquitectura de la nube

La computación en la nube basa su arquitectura haciendo una separación entre hardware, plataforma y aplicaciones quedando las siguientes capas:

**2.7.3.1 Software como servicio (SaaS)**

Se encuentra en la capa más alta y consiste en la entrega de aplicaciones completas como un servicio.

El proveedor de tecnologías de información y comunicación (TIC) ofrece el SaaS (Software as a Service).

Para ello dispone de una aplicación que se encarga de operar y mantener y que frecuentemente es desarrollada por ´el mismo. Con ella se encarga de dar servicio a multitud de clientes a través de la red, sin que estos tengan que instalar ningún software adicional.

Un ejemplo claro es la aplicación para el manejo del correo electrónico por medio de un navegador de Internet.

#### 2.7.3.2 Plataforma como Servicio (PaaS)

En orden descendente, PaaS (Platform as a Service) es la siguiente capa. Básicamente su objetivo se centra en un modelo en el que se proporciona un servicio de plataforma con todo lo necesario para dar soporte al ciclo de planteamiento, desarrollo y puesta en marcha de aplicaciones y servicios web a través de la misma. El proveedor es el encargado de escalar los recursos en caso de que la aplicación lo requiera, de que la plataforma tenga un rendimiento óptimo, de la seguridad de acceso, etc. Para desarrollar software se necesitan bases de datos, herramientas de desarrollo y en ocasiones servidores y redes.

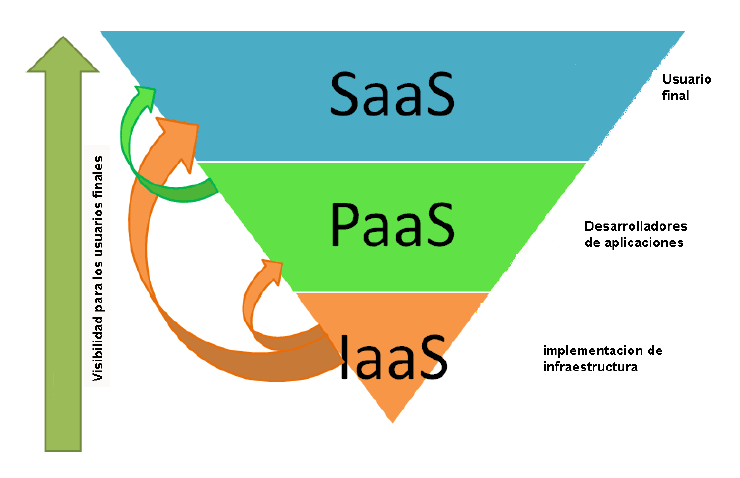
Con PaaS el cliente únicamente se enfoca en desarrollar, depurar y probar ya que la herramienta necesaria para el desarrollo de software es ofrecido a través de Internet, lo que teóricamente permite aumentar la productividad de los equipos de desarrollo.

Con el uso de PaaS, se abstrae del hardware físico al cliente, lo cual es interesante para muchos desarrolladores web, y es probable que llegue a reemplazar a las empresas de alojamiento tradicionales.

#### 2.7.3.3 Infraestructura como Servicio (IaaS)

IaaS (Infrastructure as a Service) corresponde a la capa más baja. La idea básica es la de hacer uso externo de servidores para espacio en disco, base de datos, ruteadores, swtiches así como tiempo de computo evitando de esta manera tener un servidor local y toda la infraestructura necesaria para la conectividad y mantenimiento dentro de una organizaron. Con una IaaS lo que se tiene es una solución en la que se paga por consumo de recursos solamente usados: espacio en disco utilizado, tiempo de CPU, espacio para base de datos, transferencia de datos, etc.

Las IaaS permiten desplazar al proveedor la mayor parte de los factores relacionados con la gestión de las maquinas con el ahorro de costos al pagar solo por lo consumido y olvidarse de tratar con máquinas y su mantenimiento. Por otro lado, IaaS puede permitir una escalabilidad automática o semiautomática, de forma que se puedan contratar más recursos según los se requieran5.



*Figura 2.4: Capas de computación en la nube (UAM, 2011)*

### 2.7.4 Ventajas

* **Prestación de servicios a nivel mundial.** Las infraestructuras de cloud computing proporcionan mayor capacidad de adaptación, recuperación completa de pérdida de datos (con copias de seguridad) y reducción al mínimo de los tiempos de inactividad.
* **Una infraestructura 100% de cloud computing** permite también al proveedor de contenidos o servicios en la nube prescindir de instalar cualquier tipo de software, ya que éste es provisto por el proveedor de la infraestructura o la plataforma en la nube. Un gran beneficio del cloud computing es la simplicidad y el hecho de que requiera mucha menor inversión para empezar a trabajar.
* **Implementación más rápida y con menos riesgos**, ya que se comienza a trabajar más rápido y no es necesaria una gran inversión. Las aplicaciones del cloud computing suelen estar disponibles en cuestión de días u horas en lugar de semanas o meses, incluso con un nivel considerable de personalización o integración.
* **Actualizaciones automáticas** que no afectan negativamente a los recursos de TI. Al actualizar a la última versión de las aplicaciones, el usuario se ve obligado a dedicar tiempo y recursos para volver a personalizar e integrar la aplicación. Con el cloud computing no hay que decidir entre actualizar y conservar el trabajo, dado que esas personalizaciones e integraciones se conservan automáticamente durante la actualización.
* **Contribuye al uso eficiente de la energía.** En este caso, a la energía requerida para el funcionamiento de la infraestructura. En los centros de datos tradicionales, los servidores consumen mucha más energía de la requerida realmente. En cambio, en las nubes, la energía consumida es sólo la necesaria, reduciendo notablemente el desperdicio.

### Desventajas

* **La disponibilidad de las aplicaciones** está sujeta a la disponibilidad de acceso a Internet.
* **Los datos "sensibles" del negocio no residen en las instalaciones de las empresas**, lo que podría generar un contexto de alta vulnerabilidad para la sustracción o robo de información.
* **La confiabilidad de los servicios depende de la "salud" tecnológica y financiera** de los proveedores de servicios en nube. Empresas emergentes o alianzas entre empresas podrían crear un ambiente propicio para el monopolio y el crecimiento exagerado en los servicios.
* **La disponibilidad de servicios altamente especializados** podría tardar meses o incluso años para que sean factibles de ser desplegados en la red.
* **La madurez funcional de las aplicaciones** hace que continuamente estén modificando sus interfaces, por lo cual la curva de aprendizaje en empresas de orientación no tecnológica tenga unas pendientes significativas, así como su consumo automático por aplicaciones.
* **Seguridad.** La información de la empresa debe recorrer diferentes nodos para llegar a su destino, cada uno de ellos (y sus canales) son un foco de inseguridad. Si se utilizan protocolos seguros, HTTPS por ejemplo, la velocidad total disminuye debido a la sobrecarga que éstos requieren.

## Software como Servicio (SaaS)

Software como Servicio (del inglés: Software as a Service, SaaS) es un modelo de distribución de software, en donde la compañía de tecnologías de información (IT) provee el servicio de mantenimiento, operación diaria y soporte del software usado por el cliente. Regularmente el software puede ser consultado en cualquier computador, esté presente en la empresa o no. Se deduce que la información, el procesamiento, los insumos y los resultados de la lógica de negocio del software están hospedados en la compañía de IT.

Las actividades son gestionadas desde alguna ubicación central, en lugar de hacerlo desde la sede de cada cliente, permitiendo a los clientes el acceso remoto a las aplicaciones mediante la web.

### 2.8.1 Características

Las características del software como servicio incluyen:

* Acceso y administración a través de una red.
* Actividades gestionadas desde ubicaciones centrales, en lugar de la sede de cada cliente, permitiéndoles tener acceso remoto a las aplicaciones a través de la web.
* La distribución de la aplicación es más cercana al modelo uno a muchos (una instancia con múltiples usuarios) que al modelo uno a uno, incluyendo arquitectura, precios, colaboración, y administración.
* Actualizaciones centralizadas, lo cual elimina la necesidad de descargar parches por parte de los usuarios finales.
* Multiplataforma, en donde el cliente puede ejecutar la aplicación desde diferentes sistemas operativos, incluso, diferentes tipos de dispositivos, tales como celulares, tabletas o computadores.

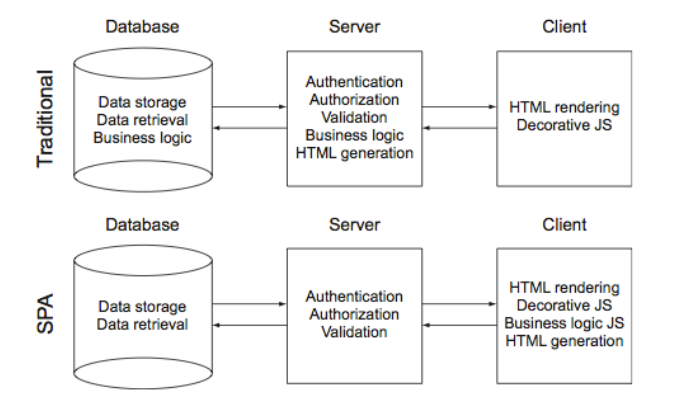
### 2.8.2 Aplicaciones de una sola página (SPA)

Una de las formas de implementar un servicio Saas (Software as a service), es desarrollando un SPA (Single page Application) en español, aplicación de una sola página, lo cual es una solución óptima para los servicios Saas.

Según (Hung, 2014) una aplicación de una sola página es: “Una aplicación de una sola página (SPA) es una aplicación web instalado en una sola página y hace que no vuelva a cargar la página durante su uso para proporcionar una mejor experiencia de usuario y aumentar el rendimiento”.

### 2.8.2.1 Estructura de un SPA

Cuando la aplicación web, se traslada al navegador, los requisitos para el rendimiento en el servidor son bastante importantes. La siguiente figura muestra cómo las SPA solucionan un problema de rendimiento con la lógica de negocio y el procesamiento de plantillas migran desde el servidor y el cliente.



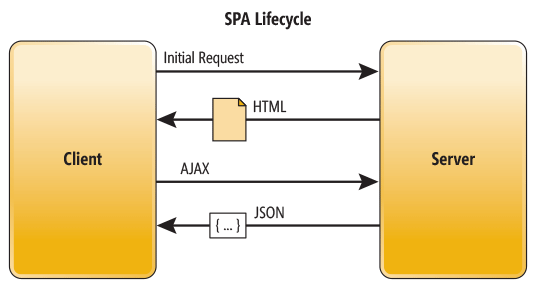
*Figura 2.5: Responsabilidades de la base de datos, servidor y cliente en una aplicación web tradicional y una aplicación de una sola página (SPA) (*HUNG, 2014*)*

Como se puede ver en el gráfico, con una SPA, toda la lógica de negocio de la base de datos y el servidor (Tradicional) se desplazan al lado del cliente (SPA) con el fin de mejorar el rendimiento de la Base de datos y servidor. Esto porque una de las principales características de un servicio SaaS es que puede tener múltiples usuarios.

Una aplicación web tradicional, como se puede ver en la figura, realiza trabajos que hacen que las solicitudes tomen mucho mas tiempo para ser resueltas, lo cual se traduce en una experiencia baja de usuario, sobretodo tratándose de un servicio SaaS.

La forma en que una SPA funciona es básicamente de la siguiente manera: cuando el usuario interactúa con la página realizando alguna acción (pulsar un botón, enlaces u otros eventos) se realiza una petición asíncrona (AJAX) al servidor, cuya respuesta (generalmente XML, JSON o similar) es interpretada por la aplicación que actualizará el DOM (Document Object Model) **“sin recargar el sitio”**, permitiendo visualizar los cambios con alta velocidad y sin desperdicio de ancho de banda. Las SPA delegan muchas operaciones que anteriormente se realizaban en el servidor al lado del cliente, reduciendo así la complejidad del sistema.

La figura siguiente muestra de una forma clara el proceso de funcionamiento de una SPA, realizando peticiones asíncronas mediante tecnología AJAX.



*Figura 2.6: Funcionamiento de un SPA, realizando peticiones asíncronas (*HUNG, 2014)

#### 2.8.2.2 Comunicación con el Servidor

Las peticiones que realiza una SPA, normalmente resultan en datos crudos (XML o JSON), o son plantillas HTML nuevos que están siendo regresados. En los casos donde el HTML regresa como resultado, JavaScript en el cliente actualiza el área parcial de DOM (Document Object Model). Cuando el dato crudo es regresado, el JavaScript de lado del cliente procesa el XML o el JSON para hacer la traducción de los datos crudos a un HTML la cual es usada para actualizar la información del área parcial del DOM.

Con todo lo descrito anteriormente sobre aplicaciones de una sola página (SPA), se concluye que la forma de comunicación entre el cliente (SPA) y el servidor, es mediante peticiones asíncronas, las cuales requieren únicamente de la URL del recurso solicitado. Razon por la cual en el lado del servidor se hace uso de los llamados “Servicios REST”.

**2.8.2.3 Transferencia de Estado Representacional (REST) API**

Según (Hung, 2014), un servicio REST es: “Transferencia de estado representacional (REST) es la arquitectura sin estado, basado en el protocolo HTTP para representar el modelo de cómo debería funcionar los sistemas web modernos. Los servicios REST utilizan la tecnología y protocolos de web existente para acceder y manipular los recursos que utilizan los verbos (métodos de solicitud HTTP).”

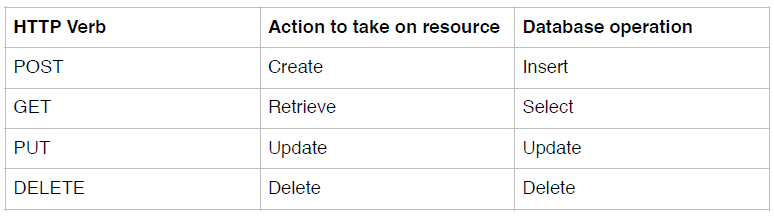


Tabla 2.7: Cómo las acciones de HTTP son relevantes para el funcionamiento de bases de datos (HUNG, 2014)

Hay una serie de verbos para el enfoque REST como GET, POST, PUT y DELETE para manipular los datos en la base de datos. Estos verbos corresponden a un CRUD tradicional (Crear, Leer, actualizar, eliminar) las operaciones en una base de datos. Además, REST también tiene 5 categorías de respuestas: información general, una petición correcta, redirigir, error en el cliente y el error en el lado del servidor. La respuesta depende del tipo de petición HTTP, como se indica en la siguiente tabla:

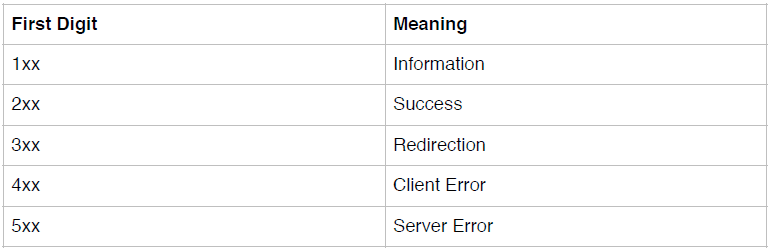


Tabla 2.8: Respuestas a peticiones comunes en un servicio REST (HUNG, 2014)

### 2.8.3 Diseño web Adaptable

# CAPITULO III – TECNOLOGÍAS UTILIZADAS

Este capítulo está dedicado a realizar una descripción completa de las herramientas tecnológicas, que se utilizaron para el desarrollo del presente trabajo de grado, tomando en cuenta características de una aplicación de una sola página (SPA), como se había descrito en el capítulo II.

## 3.1 Plataforma de desarrollo

La siguiente plataforma de desarrollo, ha sido elegida tomando en cuenta las características de un servicio SaaS (Software como Servicio) y una aplicación de una sola página (SPA).

El uso de un framework ayuda con la seguridad, reducción de tiempo de desarrollo de una aplicación. Además nos permite tener código optimizado, ordenado y entendible. Por tanto se ha hecho la elección de los siguientes frameworks: Angular Js, framework para JavaScript; CodeIgniter REST-Server, framework para PHP y Twitter Boostrap, framework para diseño web adaptable basado en CSS.

### 3.1.1 Framework Angular JS

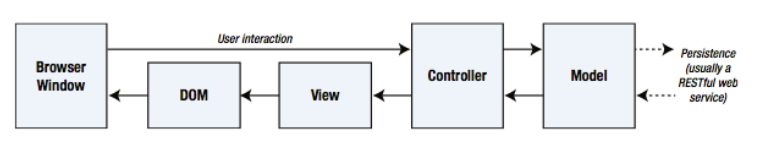
La razón principal por la que se ha elegido a Angular Js como framework de Javascript, es que ha sido desarrollado específicamente para aplicaciones web de una sola página (SPA).

#### 3.1.1.1 ¿Que es Angular?

Según la página oficial, Angular se define de la siguiente manera: “AngularJS es un marco estructural para aplicaciones web dinámicas. Permite utilizar HTML como lenguaje de plantillas y permite extender la sintaxis HTML para expresar los componentes de una aplicación clara y concisa. El enlace de datos de AngularJS y la inyección de dependencia elimina gran parte del código que de otro modo tendría que escribir. Y todo sucede dentro del navegador, lo que lo convierte en un socio ideal con cualquier tecnología de servidor.”

Angular es un framework javascript de codigo abierto, que implementa el patrón MVC (Model-View-Controller o Model-View-ViewModel). Ha sido desarrollado por Google.

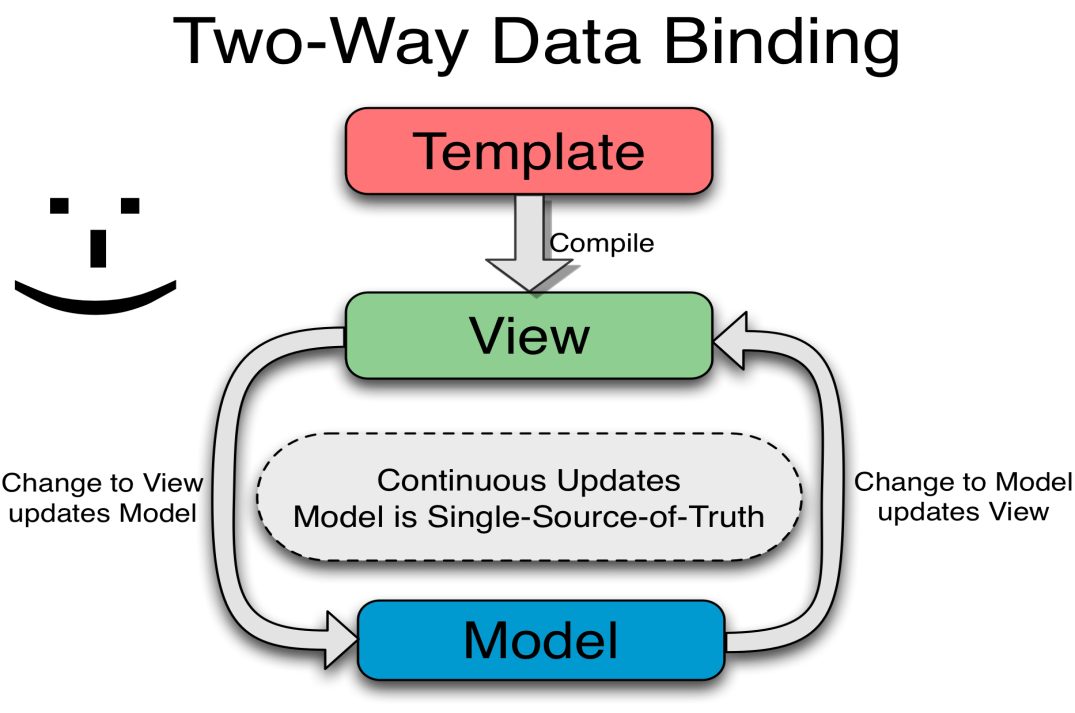
En la siguiente figura, se puede observar como Angular interactúa con los servidores web, según en patrón MVC (Modelo Vista Controlador).



*Figura 3.1: Funcionamiento del patrón MVC en Angular JS* *(HUNG, 2014)*

Como se puede ver en la figura superior, la aplicación del lado del cliente MVC obtiene datos de lado del servidor, por lo general a través de un servicio web REST. El objetivo del controlador y la vista es procesar los datos en el modelo con el fin de llevar a cabo la manipulación del DOM con el fin de crear y administrar los elementos HTML que el usuario puede interactuar.

Una de las características más influyentes en Angular, es el la implementación de “doble enlace de datos”, en inglés “two way data binding”. En la siguiente figura, se puede ver cómo funciona esta implementación.



*Figura 3.2: Funcionamiento del doble enlace de datos en Angular JS (Angular, 2016)*

AngularJS adopta un enfoque diferente para la combinación de datos de modelos y la vista. En lugar de utilizar de forma tradicional para combinar datos en una plantilla y luego vuelva a colocar un elemento DOM, AngularJS crea una vista mediante el uso de plantillas HTML en vivo. Cada componente de las vistas se interpola de forma dinámica. Esta característica es una de las características más importantes de AngularJS y se puede ver claramente en la figura.

#### 3.1.1.2 Características

* Doble enlace de datos entre los modelos y la vista.
* Sigue el patrón MVC.
* Modularización de aplicaciones.
* Sigue el patrón de inyección de dependencias.
* Construcción de componentes y modificación de comportamiento mediante directivas.
* Funciona correctamente con otras bibliotecas JavaScript.
* Documentación detallada y una comunidad de desarrolladores activa.

### 3.1.2 Framework CodeIgniter REST-Server

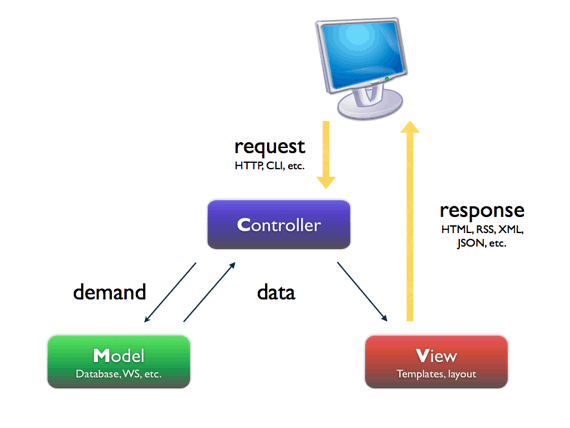
Las razones porque se ha elegido Codeigniter como framework PHP son:

* Comunicación con Base de Datos automatizada, validando todas las consultas y filtrando los datos variables para evitar inyección SQL.
* Disponer de componentes plenamente probados, con el objetivo de mejorar la productividad.
* La facilidad de implementar servicios REST, con la simple agregación de su módulo REST-Server.

#### 3.1.2.1 ¿Que es CodeIgniter?

Según la página oficial, Codeigniter tiene la siguiente definición: “CodeIgniter es un conjunto de herramientas para las personas que crean aplicaciones web usando PHP. Su objetivo es permitir el desarrollo de proyectos mucho más rápido de lo que podría si estuviera escribiendo código desde cero, al funcionar como un conjunto de librerías para tareas comunes necesarios, así como una interfaz simple y estructura lógica para acceder a estas bibliotecas. CodeIgniter le permite creativamente centrarse en su proyecto, reduciendo al mínimo la cantidad de código necesario para una tarea determinada”.

Codeigniter es un framework para el desarrollo de aplicaciones en PHP que utiliza el patrón **MVC**. En la siguiente figura se puede ver cómo funciona Codeigniter.



*Figura 3.3: Funcionamiento del patrón MVC en Codeigniter (Angular, 2016)*

* **Modelo**: es dónde se procesa y obtienen los datos, la conexión con la Base de datos.
* **Vista**: presenta los datos en pantalla, es donde va el código HTML.
* **Controlador**: controla los datos, dicho de forma rápida obtiene datos de un modelo, los procesa, y se los pasa a la vista.

#### 3.1.2.2 Características

* **La documentación** del framework es bastante extensa, hay mucha información y libros en la red sobre CI, además de que la guía que trae está muy muy completa.
* La curva de aprendizaje es muy rápida, por la sencillez del framework muy rápido le agarras la onda
* **Es flexible**, ya que no obliga a tener una determinada estructura de tablas, nombres de campos, ni adherirse a una forma de programar concreta como con otros frameworks.
* **Es ligero y compatible**, funciona en casi cualquier hosting, y no pide ningún requisito y se instala en un par de minutos (nomas es descomprimir el un archivo zip y listo).
* Las aplicaciones que se hacen con CI, son independientes del motor de BD que se utilice; usando la clase ActiveRecord que trae se puede migrar de motor de base de datos sin cambiar nada del código de la aplicación.

### 3.1.3 Framework Twitter Bootstrap

#### 3.1.3.1 ¿Que es Twitter Bootstrap?

Según la página oficial, Twitter bootstrap es: “Twitter Bootstrap es un framework o conjunto de herramientas de software libre, para el diseño de sitios y aplicaciones web adaptables o responsivos”.

Bootstrap fue desarrollado por Mark Otto y Jacob Thornton en Twitter. Fue lanzado como un producto de código abierto en agosto de 2011 en GitHub.

Este framework ofrece las siguientes funcionalidades:

* **Sistema de regillas**: Bootstrap proporciona una estructura básica con Sistema de rejilla, enlace, estilos, y fondos.
* **CSS**: Bootstrap viene con la característica de la configuración global de CSS, elementos HTML fundamentales con estilos predefinidos, clases extensibles y un sistema de regilla avanzada.
* **Componentes**: Bootstrap contiene más de una docena de componentes reutilizables construidos para proporcionar la iconografía, menús desplegables, navegación, alertas, pop-off, etc.
* **Extensiones JavaScript**: Bootstrap contiene más de una docena de extensiones jquery. Se puede incluir a todos componentes, o uno por uno.
* **Personalización**: Puede personalizar los componentes de archivos de inicio, como las variables y las extensiones jQuery para obtener una propia versión.

La principal característica de Bootstrap, es que las aplicaciones creadas con este framework, se ajustan a los ordenadores de escritorio, tabletas y móviles.



*Figura 3.4: Diseño web adaptable con Bootstrap (*Bootstrap, 2016)

#### 3.1.2.2 Características

**Enfoque de la filosofía “primero móviles”:** Consiste en diseñar primero la interfaz para los dispositivos móviles.

**Soporte para navegadores**: Soporta los navegadores populares como: Chrome, Firefox, Opera e Internet Explorer.

**Curva de aprendizaje**: Solo se necesita conocimiento sobre HTML y CSS. Además se tiene una buena documentación sobre sus componentes.

Componentes prediseñados: Contiene estilos predefinidos para sus componentes.

**Diseño web adaptable**: La interfaz se ajusta a distintos tamaños de dispositivos.

**Personalización**: También proporciona la personalización basada en la web.

### 3.1.4 JavaScript

Ya que se hace uso del framework Angular, el lenguaje de programación JavaScript es un requisito para desarrollar aplicaciones en dicho framework.

JavaScript(a veces abreviado como JS) es un lenguaje ligero e interpretado, orientado a objetos con funciones de primera clase, más conocido como el lenguaje de script para páginas web, pero también usado en muchos entornos sin navegador, tales como node.js o Apache CouchDB. Es un lenguaje script multi-paradigma, basado en prototipos, dinámico, soporta estilos de programación funcional, orientada a objetos e imperativa.

JavaScript no debe ser confundido con el lenguaje de programación Java. Java es una marca registrada de Oracle en Estados Unidos y otros países.

JavaScript es un lenguaje de programación dinámico que soporta construcción de objetos basado en prototipos. La sintaxis básica es similar a Java y C++ con la intención de reducir el número de nuevos conceptos necesarios para aprender el lenguaje. [6]

#### 3.1.4.1 Características

* Imperativo y estructurado.
* Tipado dinámico.
* Puede funcionar como lenguaje procedimental y como orientado a objetos.
* Evaluación del código en tiempo de ejecución.
* Funcional.
* Basado en prototipos.

### 3.1.5 PHP

“PHP es un lenguaje de programación para la creación de sitios Web dinámicos interactivos. Como regla general, los programas de PHP se ejecutan en un servidor Web y servir páginas web a los visitantes bajo petición. Una de las principales características de PHP es que se puede incrustar código PHP dentro de las páginas Web HTML, por lo que es simple crear contenido dinámico rápidamente”. [libro]

PHP es un lenguaje de script del lado del servidor, lo que significa que los scripts PHP, o programas, generalmente se ejecutan en un servidor Web. Por otra parte, PHP es un lenguaje interpretado, un script PHP es procesado por el motor de PHP cada vez que se ejecuta.

Al ser PHP un lenguaje que se ejecuta en el servidor no es necesario que su navegador lo soporte, es independiente del navegador, pero sin embargo, para que las páginas PHP funcionen, el servidor donde están alojadas debe soportar PHP.

El proceso de ejecución de un script PHP en un servidor Web es el siguiente:

1. Un visitante solicita una página Web presionando en un vínculo, o escribiendo la dirección URL en la barra de direcciones del navegador. El visitante también puede enviar datos al servidor Web al mismo tiempo, ya sea utilizando un formulario en una página web, o a través de AJAX (Asynchronous JavaScript And XML).

2. El servidor Web reconoce que la URL solicitada es un script PHP, e instruye al motor de PHP para procesar y ejecutar el script.

3. El script se ejecuta, y cuando se termina, normalmente envía una página HTML al navegador Web, que el visitante ve en su pantalla.

#### 3.1.5.1 Características

* Es un lenguaje multiplataforma.
* Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL.
* Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados extensiones).
* Posee una amplia documentación, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
* Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
* Permite las técnicas de Programación Orientada a Objetos.
* Biblioteca nativa de funciones sumamente amplia e incluida.
* No requiere definición de tipos de variables.
* Tiene manejo de excepciones (desde PHP5).

### 3.1.6 MYSQL

MySQL es un sistema gestor de bases de datos (SGBD, DBMS por sus siglas en inglés) muy conocido y ampliamente usado por su simplicidad y notable rendimiento. Aunque carece de algunas características avanzadas disponibles en otros SGBD del mercado, es una opción tanto para aplicaciones comerciales, como de entretenimiento precisamente por su facilidad de uso y tiempo reducido de puesta en marcha. Esto y su libre distribución en Internet bajo licencia GPL le otorgan como beneficios adicionales (no menos importantes) contar con un alto grado de estabilidad y un rápido desarrollo.

MySQL está disponible para múltiples plataformas, la seleccionada para los ejemplos de este libro es GNU/Linux. Sin embargo, las diferencias con cualquier otra plataforma son prácticamente nulas, ya que la herramienta utilizada en este caso es el cliente mysql-client, que permite interactuar con un servidor MySQL (local o remoto) en modo texto. De este modo es posible realizar todos los ejercicios sobre un servidor instalado localmente o, a través de Internet, sobre un servidor remoto.

#### 3.1.6.1 Características

* Está desarrollado en C/C++.
* Se distribuyen ejecutables para cerca de diecinueve plataformas diferentes.
* La API se encuentra disponible en C, C++, Eiffel , Java, Perl, PHP, Python, Ruby y TCL.
* Está optimizado para equipos de múltiples procesadores.
* Es muy destacable su velocidad de respuesta.
* Se puede utilizar como cliente-servidor o incrustado en aplicaciones.
* Cuenta con un rico conjunto de tipos de datos.
* Soporta múltiples métodos de almacenamiento de las tablas, con prestaciones y rendimiento diferentes para poder optimizar el SGBD a cada caso concreto.
* Su administración se basa en usuarios y privilegios.
* Se tiene constancia de casos en los que maneja cincuenta millones de registros, sesenta mil tablas y cinco millones de columnas.
* Sus opciones de conectividad abarcan TCP/IP, sockets UNIX y sockets NT, además de soportar completamente ODBC.
* Los mensajes de error pueden estar en español y hacer ordenaciones correctas con palabras acentuadas o con la letra ’ñ’.
* Es altamente confiable en cuanto a estabilidad se refiere.

### 3.1.7 Apache

Como servidor web local de pruebas se ha decidido por Apache web server, con el fin de coincidir con el servidor de producción.

#### 3.1.7.1 Características

### 3.1.8 Plataforma de producción

# CAPITULO IV – ÁREA DE APLICACIÓN

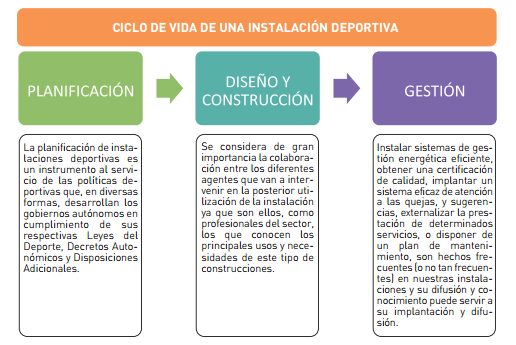
En este capítulo se realiza una definición sobre complejos deportivos, características y conpectos fundamentales que se manejan en la administracion de las reservas. Se hace una descripción completa de los procesos que se realizan en el departamento de deportes de la Universidad Mayor de San Simón como ser: registro de campos deportivos, realización de una reserva, etc., así como el diagrama de flujo de proceso para cada una de ellas.

## 4.1 Instalación deportiva

Una instalación deportiva es un recinto o una construcción provista de los medios necesarios para el aprendizaje, la práctica y la competición de uno o más deportes. Incluyen las áreas donde se realizan las actividades deportivas, los diferentes espacios complementarios y los de servicios auxiliares. Las instalaciones deportivas se componen de uno o más espacios deportivos específicos para un tipo de deporte (IBV, 2011).

Una instalación deportiva puede tener un solo espacio deportivo o varios, cada uno destinado a un deporte diferente. Así, por ejemplo, un estadio de fútbol puede contener una pista de atletismo, y albergar en el sótano un gimnasio y una sala polideportiva; en este caso se habla de una instalación deportiva con cuatro espacios deportivos diferentes (IBV, 2011).

Ejemplos de instalaciones deportivas son los estadios, los pabellones deportivos, velódromos, pistas de tenis, piscinas, canales de remo y piragüismo, marinas deportivas, estaciones de esquí, circuitos de bicicletas, campos de tiro, de hípica, de golf, etc.



*Figura 4.1: Ciclo de vida de una instalación deportiv* (IBV, 2011)

### 4.1.1 Espacio deportivo

Es el espacio donde se desarrolla una actividad deportiva específica, por ejemplo una sala de artes marciales, la pista de carreras en un estadio de atletismo, un campo de tiro al plato o una pista de snowboard.

### 4.1.2 Complejo deportivo

Se trata de dos o más instalaciones deportivas ubicadas en un recinto común y con fácil acceso entre cada una de sus partes; funcionan independientemente entre sí y se conocen generalmente bajo una misma denominación. El llamado Anillo Olímpico de Montjuïc, en Barcelona, es un complejo deportivo que incluye varias instalaciones: el Estadio Olímpico Lluís Companys (atletismo y fútbol), el Palau Sant Jordi (polideportivo), las Piscinas Bernat Picornell (natación), un campo de béisbol y otro de hockey hierba.

### 4.1.3 Espacios complementarios

Sirven para dar apoyo a las actividades deportivas desarrolladas en los diferentes espacios deportivos; el deporte no es realizado en estos espacios. Ejemplos: vestuarios, aseos, primeros auxilios, control antidopaje, almacenes de material deportivo, gradas, etc.

### 4.1.4 Servicios auxiliares

No están relacionados con las actividades deportivas. Pueden ser de diversa índole: cafeterías, bares, guarderías, tiendas, servicio médico, cuartos de máquinas, de calderas, etc.

### 4.1.5 Cancha

El concepto de “cancha” tiene origen quechua. Es posible, según los expertos en cuestiones lingüísticas, distinguir entre dos raíces etimológicas de la palabra: kamcha y kancha. En el primer caso, se refiere a las habas ya tostadas y al maíz que los habitantes de la región sudamericana acostumbran ingerir.

En cambio, la palabra cancha inspirada en kancha (traducida como “recinto”) posee una mayor amplitud de acepciones. El vocablo se utiliza para referirse al espacio reservado para la práctica de distintos deportes o para la organización de espectáculos. Entre los ejemplos de cancha tenemos: futbol-sala, futbol, bascket, etc.

### 4.1.6 Clasificación

Una instalación deportiva se clasifica de acuerdo a sus espacios deportivos. Estos pueden ser de tres tipos:

#### 4.1.6.1 Espacios deportivos convencionales

Dan servicio a las prácticas deportivas más comunes, y atienden a referentes reglamentados con dimensiones normalizadas, aunque no siempre se ajustan a ellas.

Hay seis tipos de espacios convencionales:

**Campos**: generalmente son de forma rectangular, al aire libre y con delimitación y marcación clara. Superan los 1500 m² de superficie. Ejemplos: campos polideportivos, de fútbol, fútbol 7, rugby, hockey sobre hierba, béisbol, zonas de lanzamiento de atletismo.

**Espacios longitudinales**: espacios en los que la actividad se realiza siguiendo un recorrido fijo y delimitado. Ejemplos: pistas de atletismo (de 200 m, 300 m o 400 m), rectas de saltos atléticos, velódromos.

**Pistas**: son como los campos, de forma rectangular y con delimitación y marcación clara, generalmente están al aire libre, aunque las hay cubiertas (en pabellones). Su superficie es menor a 1500 m². Ejemplos: pistas polideportivas, de baloncesto, voleibol, vóley playa, tenis, bádminton, patinaje sobre ruedas, patinaje sobre hielo, hockey sobre hielo, etc.

**Pistas con pared**: son como las pistas pero con una o más paredes que son necesarias para el desarrollo de la actividad deportiva. Ejemplos: pistas de frontón, squash, padel, etc.

**Salas**: son espacios cubiertos de poca superficie donde se pueden realizar diferentes actividades deportivas y que no tienen los requerimientos dimensionales y de marcación de las pistas. Ejemplos: las salas de artes marciales, esgrima, gimnasia, halterofilia, etc.

**Piscinas**: cuerpos cubiertos de agua para la práctica de deportes acuáticos, pueden estar al aire libre o cubiertas. Piscinas de natación (de 25 m, 33 m o 50 m), saltos, waterpolo.

#### 4.1.6.2 Espacios deportivos singulares

Son espacios más específicos que suelen estar sujetos a unos requerimientos espaciales.

Representativos de este tipo son:

**Campos de golf**: de minigolf, de 18 hoyos, de 9 hoyos.

**Campos de tiro**: para tiro con arco, tiro de precisión o tiro al plato.

**Canal de regatas**: para remo, piragüismo en aguas tranquilas o piragüismo en eslalon.

**Circuito de bicicleta**: circuitos cerrados y carril bici.

**Circuitos de carrera a pie**.

**Circuitos de motor**: circuitos de velocidad (para autos o motocicletas), de motocross, de kart

**Espacios de hípica**: pistas de doma, de saltos, hipódromos, campos de polo, etc.

**Estaciones de esquí**: para esquí de fondo, esquí alpino, snowboard, etc.

**Zonas de juegos populares o tradicionales**: boleras, billares, pistas de petanca, etc.

#### 4.1.6.3 Áreas de actividad deportiva

Se distinguen por la indefinición de sus límites y por el medio natural en el que la práctica físico deportiva tiene lugar: acuático, aéreo o terrestre.

Según el medio natural en el que la actividad deportiva es realizada, se distinguen tres tipos:

**Áreas de actividad acuática**: zonas de barranco (ráfting, piragüismo en aguas bravas), de actividades subacuáticas, surf, vela, esquí acuático, de remo y piragüismo (en embalses, lagos o bahías), etc.

**Áreas de actividad aéreas**: zonas de paracaidismo, aeromodelismo, vuelo sin motor, vuelo aerostático, parapente, ala delta, etc.

**Áreas de actividad terrestres**: zonas de escalada, espeleología, senderos, campo a través (atletismo, BTT, ecuestre, esquí), rutas ecuestres, campos de caza, etc.

## 4.2 Administración de Horarios y Reservas

El presente trabajo de grado tiene como caso de estudio al “**Complejo Deportivo San Simón**” que pertenece al departamento de deportes, cultura y recreación de la “**Universidad Mayor de San Simón**”. Por consiguiente, todos los procesos que son descritos sobre administración de horarios y reservas, están basados en dicho complejo.

El departamento de deportes de la Universidad mayor de San Simón se encarga de registrar, controlar y reservar los campos deportivos. Además, generar reportes diarios por reservas. Estos procesos se realizan manualmente, lo que representa un esfuerzo por parte de los encargados de este departamento, por que año tras año se genera más información por campo deportivo, lo que hace que al buscar datos y realizar reservas se cometan errores. Si se contara con un sistema de información para administrar dichos procesos, las tareas se realizarían en menor tiempo y además los errores ya no serían cometidos.

Actualmente el complejo deportivo San simón cuenta con los siguientes campos deportivos:

* Cancha de futbol 11.
* Cancha de futbol 8.
* Cancha de futbol 5.
* Cancha de frontón.
* Cancha de basquetbol.
* Cancha de futbol de salón.
* Cancha de boleibol.

A continuación, se hace la descripción de cómo se administran dichos campos deportivos para sus correspondientes reservas y control de horarios.

### 4.2.1 Administración de Horarios

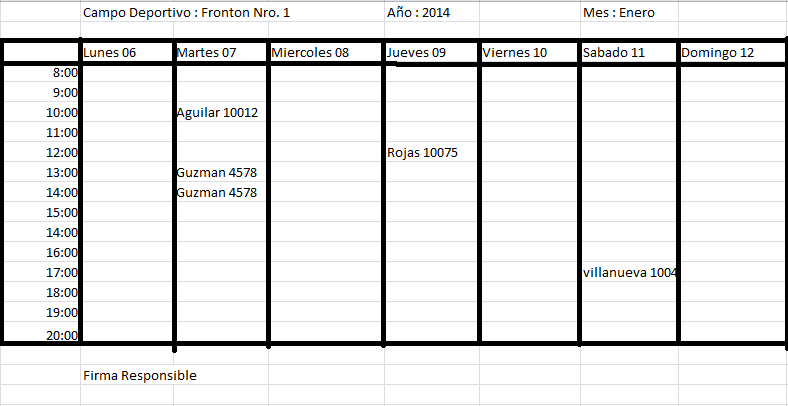
Administración de horarios es el proceso de desarrollo, mantenimiento y comunicación de los horarios de tiempo de los campos deportivos.

Para hacer esto posible, se realiza un registro de cada campo deportivo existente en las instalaciones. Este registro lo realiza la persona encargada de administrar, tanto los horarios como las reversas o fletes de los campos. Este registro, permite tener un control de horarios, ya que por cada campo deportivo, existe una planilla con los horarios disponibles.

El proceso que se lleva a cabo, para el registro de un campo deportivo es el siguiente:

* El administrador del complejo identifica a cada campo con un código, ya que pueden existir varios campos con la misma disciplina deportiva.
* Identifica las características del campo deportivo, como pueden ser: tipo de suelo, disciplina deportiva, superficie, nombre del campo si es que existiese otro con la misma disciplina. Una vez identificado, registra dichos datos.
* Registra el precio por hora de reserva del campo deportivo. Esta característica es una de las más importantes ya que es con la que se trabaja al momento de hacer las reservas.
* Se asigna un horario de atención, dicho horario de atención esta segmentado en horas, por ejemplo: 8:00 a 9:00, 9:00 a 10:00, 10:00 a 11:00, etc.
* Finalmente, es creado una planilla de horarios por cada campo deportivo, en donde se encuentran datos como: hora, día, mes y año. Esta planilla es donde se insertan datos de reserva.

En la siguiente figura, se puede observar un ejemplo de planilla de horarios por campo deportivo.



*Figura 4.2: Planilla de horarios por campo deportivo (Elaboración Propia, 2015)*

Como se puede ver en la planilla superior, es necesario tanto el nombre del campo deportivo, el año, el mes, el día y la hora por cada subdivisión de los horarios.

Otro dato importante, es que esta planilla es generada para cada semana de una gestión, por ejemplo, por mes se tiene que generar 4 planillas, entonces por año debería generarse 48 planillas. Por lo tanto, la cantidad de información que se administra sobre campos deportivos es demasiada, y poco eficiente administrarlo mediante hojas de papel impreso.

El diagrama de flujo del proceso de registro de campo deportivo se puede apreciar en la figura inferior.

C:\Users\Beimar\Downloads\flujo2 (1).png

*Figura 4.3: Diagrama de flujo para el registro de una campo deportivo (Elaboración propia, 2015)*

### 4.2.2 Reservas

Se diferencian tres tipos de reservas:

* Reservas regulares.
* Pre reservas realizadas por usuario.
* Reservas procedentes de días de mantenimiento.

Para la realización de cualquier tipo de reserva, es necesario que el proceso de registro de campo deportivo se encuentre completado, puesto que, la planilla de horarios por campo deportivo va a ser usado en este proceso.

#### 4.2.2.1 Reservas regulares

La característica común de estas reservas es que se originan cuando la empresa ha tenido beneficios. Si esto ha sucedido, la cuenta que expresa el beneficio tendrá un saldo acreedor y cuando se acuerde su traspaso a cualquier tipo de las reservas anteriormente citadas. En conclusión, este tipo de reservas generan ingresos para el complejo deportivo.

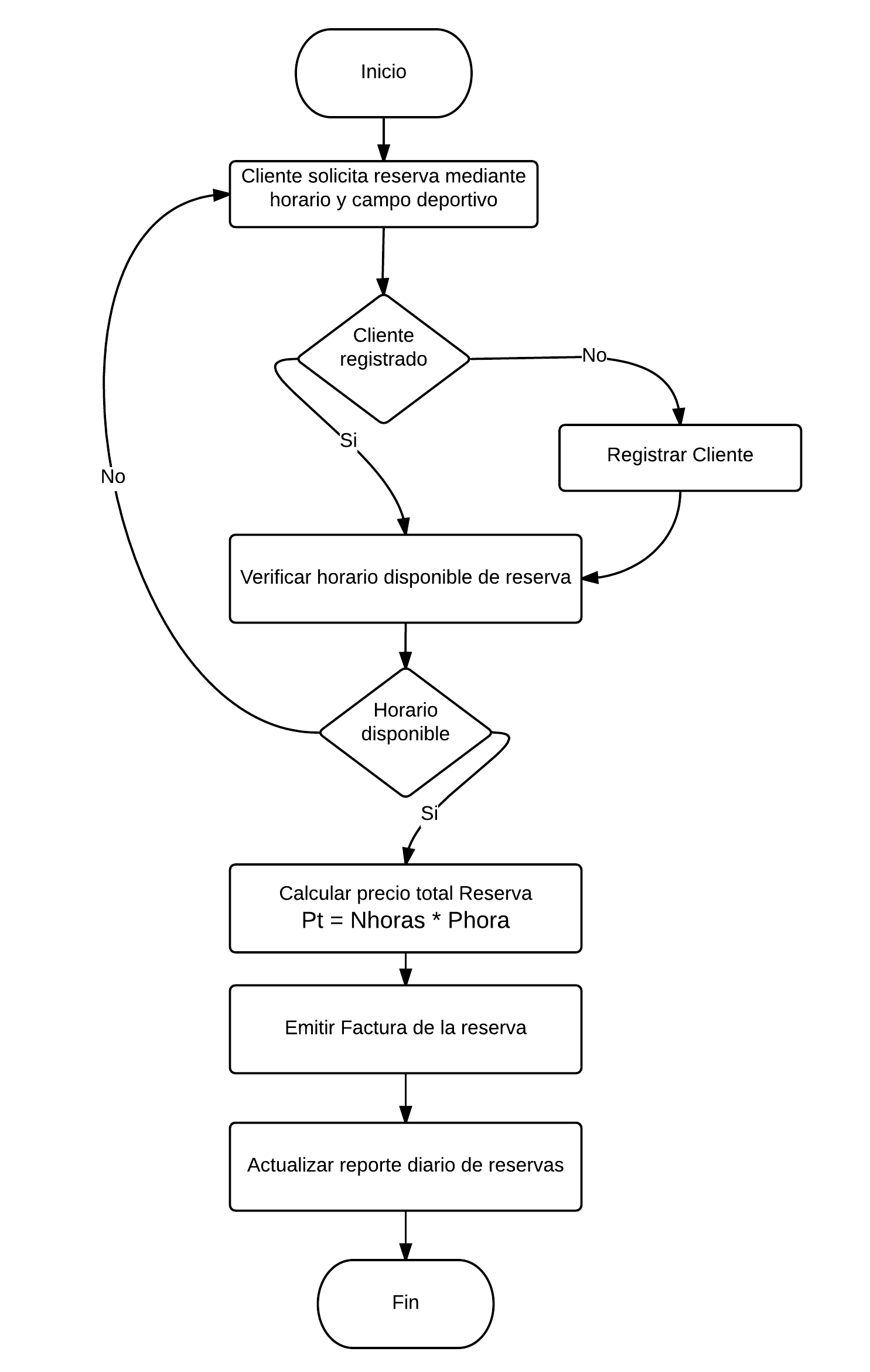
El encargado de aprobar este tipo de reserva es el administrador, ya que al momento de emitir el recibo, es necesaria la firma de dicho administrador.

El proceso que se sigue para realizar una reserva regular es el siguiente:

* Un cliente, ya sea que este registrado o no, solicita una reserva, aportando datos como la fecha y hora de la reserva y el campo deportivo que desea fletar.
* El encargado verifica si el cliente se encuentra registrado en el complejo, si está registrado sigue con la siguiente tarea, si no, hace el registro del cliente con los siguientes datos: nombre, apellidos, número de cedula de identidad y teléfono.
* El siguiente paso es, verificar si en la fecha y horario solicitado, el campo deportivo está libre de reservas.
* Si el campo deportivo no está libre de reservas, entonces el administrador pide al cliente que solicite una diferente fecha para la reserva.
* Si el campo deportivo está libre de reservas, entonces el administrador calcula el precio a pagar por el cliente mediante la siguiente formula:
* **Pt = Nhoras \* Phora**
* Luego, se emite el recibo correspondiente para la reserva, introduciendo los siguientes datos: Nombre Cliente, C.I. del cliente, nombre del administrador, numero de recibo, datos básicos del complejo deportivo, nombre del campo deportivo, hora de reserva, fecha de reserva y el precio de la reserva.
* Por último, el administrador actualiza el reporte diario de reservas, que es una planilla, donde se registran todas las reservas que se realizaron en el día.

El reporte diario mencionado contiene los siguientes datos: fecha del reporte, nombre del encargado, cedula de identidad del encargado, total ingreso por día, firma del administrador y una tabla detallada de todas las reservas realizadas por día.

El proceso de reservas regulares se observa en la *Figura 4.4.*

**

*Figura 4.4: Diagrama de flujo para una reserva (Elaboración propia, 2015)*

#### 4.2.2.2 Pre reservas

Las pre reservas son un tipo especial de reservas, que se caracterizan principalmente porque lo solicitan los clientes, pero esta reserva no es cancelada en el momento en el que se la hace, sino en un momento posterior. El complejo realiza este tipo de reservas, para mantener la fidelidad de los clientes.

El proceso de realizar una pre reserva, tiene dos pasos: el primero es registrar la pre reserva y el segundo es confirmar dicha pre reserva. Por otra parte, este tipo de reserva solo se las hace a clientes que están registrados en el complejo deportivo.

A continuación se describe los el proceso para el registro de una pre reserva:

**Paso 1: Registrar la pre reserva**

* Un cliente, ya sea que este registrado o no, solicita una pre reserva, aportando datos como la fecha y hora de la reserva y el campo deportivo que desea fletar.
* El encargado verifica, si el cliente se encuentra registrado en el complejo, si está registrado se procede al siguiente paso.
* El siguiente paso es, verificar si en la fecha y horario solicitado, el campo deportivo está libre de reservas.
* Si el campo deportivo no está libre de reservas, entonces el administrador pide al cliente que solicite una diferente fecha para la reserva.
* Si el campo deportivo está libre de reservas, entonces se registra la pre reserva.

**Paso 2: Confirmar la pre reserva**

* El cliente solicita confirmar una pre reserva.
* El administrador realiza una búsqueda entre las pre reservas registradas, lo cual toma un cierto tiempo.
* Una encontrado la pre reserva existente, el administrador calcula el precio a pagar por el cliente mediante la siguiente formula:

**Pt = Nhoras \* Phora**

* Luego, se emite el recibo correspondiente para la reserva, introduciendo los siguientes datos: Nombre Cliente, C.I. del cliente, nombre del administrador, numero de recibo, datos básicos del complejo deportivo, nombre del campo deportivo, hora de reserva, fecha de reserva y el precio de la reserva.
* Por último, el administrador actualiza el reporte diario de reservas, que es una planilla, donde se registran todas las reservas que se realizaron en el día.

#### 4.2.2.3 Reservas procedentes de días de mantenimiento

Este tipo de reservas, se dan cuando un campo deportivo necesita de un mantenimiento, ya sea de limpieza o refacción del campo deportivo.

Los casos más comunes de mantenimiento son:

* Riego para canchas de futbol.
* Lustre para canchas de futbol de salón con superficie de madera.
* Pintado de límites para canchas de futbol de salón.
* Etc.

La característica principal de este tipo de reserva, es que no genera ningún ingreso económico para el complejo, mas al contrario, es una pérdida económica, pero necesaria para mantener en buen estado el campo deportivo.

Esta reserva la autoriza el administrador del complejo deportivo y sigue el sigue el siguiente proceso:

* El administrador solicita mantenimiento para un campo deportivo, con los siguientes datos: fecha y hora de mantenimiento.
* Se verifica que en la fecha y hora solicitada, el campo deportivo se encuentre libre de reservas.
* Si está libre de reservas, se registra dicho mantenimiento en las planillas de horarios por campo deportivo.

Registrando en las planillas de horario por campo deportivo, se evita tener errores de reservas.

C:\Users\Beimar\Downloads\reservaespecial.png

*Figura 4.5: Proceso para realizar una reserva por días de mantenimiento (Elaboración propia, 2015)*

#### 4.2.2.4 Reservas periódicas

Es otro tipo de reservas, que sucede, aparece o se realiza con intervalos regulares de tiempo o con cierta frecuencia. Estas reservas se basan en las anteriores, porque tienen la misma forma, unas generan ingresos y otras no. Se podría decir, que es un conjunto de las anteriores reservas, que es solicitada por clientes en casos como por ejemplo: el cliente reserva el campo deportivo de futbol los días sábados a las 13:00 PM de la tarde, lo particular de este caso, es que el cliente fleta dicho campo deportivo a la misma hora, el mismo día.

Existen casos en los que el cliente, realiza la reserva de un campo deportivo por un mes, pero solo los días sábados a una determinada hora. A esta reserva se la conoce como reserva periódica.

# CAPITULO V – PILA DEL PRODUCTO

Ya que la metodologia scrum esta basado en historias de usuario, en este capítulo, se presentan las historias de usuario identificadas de acuerdo con las necesidades del administrador de un complejo deportivo.

## 5.1 Pila del producto

De acuerdo al capítulo de “Áreas de Aplicación”, donde se hace la descripción de los procesos que se llevan a cabo en un complejo deportivo, se han identificado los siguientes requerimientos funcionales del sistema, también conocidos como historias de usuario en la metodología scrum. De acuerdo a las historias de usuario identificadas se definirá la pila del producto.

A continuación, se describen las historias de usuario que forman parte de la pila del producto. Como se podra observar, las historias de usuario no estan estimadas y tampoco tienen asignado una iteracion especifica, esto es porque, deacuerdo a la metodologia, tanto la estimacion por puntos y la asignacion de iteracion se la realiza en la planificacion de cada iteracion.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 1** | **Usuario:** Administrador | |
| **Nombre historia:** Registro de campo deportivo | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  Alta |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Desarrollador responsable:** Beimar E. Huarachi Mamani | | |
| **Descripción:**  Yo como administrador me gustaría registrar campos deportivos para llevar un control sobre ellos, ingresando datos relevantes respecto a dicho campo, como por ejemplo: Nombre, número si es necesario, horario de atención, precio por hora de reserva, tipo de disciplina y tipo de superficie. | | |
| **Criterios de Aceptación:**   * Debo ser capaz de agregar información descriptiva de cada cancha datos: Nombre campo deportivo, horarios de atención, precio por hora de reserva, tipo de disciplina y tipo de superficie. * Todos los datos anteriormente citados deben ser introducidos obligatoriamente. * Debo ser capaz de agregar imágenes de las canchas. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 2** | **Usuario:** Administrador | |
| **Nombre historia:** Registro de reserva regular. | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  Alta |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Desarrollador responsable:** Beimar E. Huarachi Mamani | | |
| **Descripción:**  Yo como administrador quisiera registrar una reserva regular. Para registrar una reserva se necesita: nombre del cliente, teléfono, campo deportivo, hora, fecha. | | |
| **Criterios de Aceptación:**   * Debo ser capaz de agregar una reserva con los siguientes datos: nombre cliente, teléfono, campo deportivo, hora, fecha, cantidad de horas. * Debo ser capaz de ver todas las reservas ya realizadas. * No debe registrar una reserva con fecha menor a la actual. * No debe registrar una reserva con fecha actual y hora menor a la hora actual. * No debe registrar una reserva, si existe otra reserva para el mismo campo, el mismo día y exista colisión en horarios. * No debe registrar una reserva, que tenga horario inicio mayor o igual al horario final. * No debe registrar una reserva, cuya duración es menor a una hora. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 3** | **Usuario:** Administrador | |
| **Nombre historia:** Yo como administrador quisiera registrar clientes. | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  Alta |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Desarrollador responsable:** Beimar E. Huarachi Mamani | | |
| **Descripción:**  El administrador debe tener un registro de los clientes para realizar reservas de una manera más fácil. Registro de usuarios de la cancha con los siguientes datos: nombres, apellidos, teléfono, ci. | | |
| **Criterios de Aceptación:**   * Se debe poder registrar un cliente con los datos: Teléfono, Nombres, Apellidos, Número CI y deseable el correo electrónico. * El Número de C.I. debe ser unico para cada cliente. * Los nombres y apellidos deben obligatorios para cada cliente. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 4** | **Usuario:** Administrador | |
| **Nombre historia:** Yo como administrador me gustaria registrar reserva administrativa. | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  Media |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Desarrollador responsable:** Beimar E. Huarachi Mamani | | |
| **Descripción:**  Como administrador me gustaria registrar reservas administrativas como por ejemplo: Mantenimiento de canchas, paro, etc.  Estas reservas no tienen costo, pero si deben controlar que otras reservas no se la realicen en los horarios de las estas reservas.  Se necesita: Horario, fecha, campo deportivo. | | |
| **Criterios de Aceptación:**   * Debo ser capaz de resgistrar una reserva administrativa. * Debo ser capaz de ver todas las reservas ya registradas. * No debe registrar una reserva con fecha menor a la actual. * No debe registrar una reserva con fecha actual y hora menor a la hora actual. * No debe registrar una reserva, si existe otra reserva para el mismo campo, el mismo día y exista colisión en horarios. * No debe registrar una reserva, que tenga horario inicio mayor o igual al horario final. * No debe registrar una reserva, cuya duración es menor a una hora. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 5** | **Usuario:** Cliente | |
| **Nombre historia:** Como Cliente del complejo deportivo me gustaría realizar pre reservas. | | |
| **Prioridad en negocio:**  Media | | **Riesgo en desarrollo:**  Media |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Desarrollador responsable:** Beimar E. Huarachi Mamani | | |
| **Descripción:**  Como cliente del complejo deportivo me gustaria realizar pre reservas con los siguientes datos: fecha, horario, campo deportivo y el complejo deportivo | | |
| **Criterios de Aceptación:**   * No puedo colisionar con otras prereservas o reservas. * Avisar a cerca del Plazo para pagar precio prereserva a Admi. * El Cliente solo puede realizar una prereserva, y ademas haber solicitado reservas anteriormente al complejo * Datos necesarios iguales al de una reserva regular. * Las prereservas se realizan 48 antes de la fecha establecida | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 6** | **Usuario:** Administrador | |
| **Nombre historia:** Como administrador me gustaria realizar reservas periódicas con las Reservas Especiales y administrativas | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  Media |
| **Puntos estimados:** | | **Iteración asignada: 2** |
| **Desarrollador responsable:** Beimar E. Huarachi Mamani | | |
| **Descripción:**  Como administrador me gustaria registrar reservas periodicas , tanto de las reservas especiales y las reservas regulares. Las reservas deberian contener los siguintes datos: fecha, horario, campo deportivo y el nombre del cliente. | | |
| **Criterios de Aceptación:**   * Realiza la reserva si está o sus repeticiones no tienen choques con otras reservas y sus repeticiones respectivas y si sus datos son válidos. * Como maximo, se deben realizar solo 10 reservas por un cliente. * Se debe poder especificar de que tipo de reserva periodica se esta registrando. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 7** | **Usuario:** Administrador | |
| **Nombre historia:** Yo como administrador quisiera confirmar las pre reservas | | |
| **Prioridad en negocio:**  Media | | **Riesgo en desarrollo:**  Media |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Desarrollador responsable:** Beimar E. Huarachi Mamani | | |
| **Descripción:**  Yo como administrador quisiera confirmar las prereservas que realizan los clientes. | | |
| **Criterios de Aceptación:**   * Solo deben confirmarse las prereservas actuales. * Debo poder buscar el nombre del cliente que solicito la pre reserva. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 8** | **Usuario:** Administrador | |
| **Nombre historia:** Yo como administrador quisiera imprimir los recibos para las reservas. | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  Media |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Desarrollador responsable:** Beimar E. Huarachi Mamani | | |
| **Descripción:**  Como administrador me gustaria poder generar un recibo al realizar un reserva, o al confirmar una pre reserva con la siguiente informacion: Nombre del encargado, nombre del cliente, C.I. del cliente, numero de recibo, detalle de la reserva, nombre del campo deportivo y precio. El precio total debe indicarse tanto en letras como en cantidad en digitos numeros. | | |
| **Criterios de Aceptación:**   * El recibo debe contener la siguiente informacion: Nombre del encargado, nombre del cliente, C.I. del cliente, numero de recibo, detalle de la reserva, nombre del campo deportivo y precio. * El precio total debe indicarse tanto en letras como en cantidad en digitos numeros. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 9** | **Usuario:** Administrador | |
| **Nombre historia:** Yo como administrador quisiera poder imprimir los reportes diarios de ingresos por Reservas realizadas. | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  Media |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Desarrollador responsable:** Beimar E. Huarachi Mamani | | |
| **Descripción:**  Yo como adminstrador quiesiera poder imprimir los reportes diarios de ingresos por Reservas realizadas. Me gustaria que el registro diario se genere de forma automatica, ya que actualmente se realiza de forma manual. | | |
| **Criterios de Aceptación:**   * El documento debe tener los datos del complejo deportivo. * El documento debe tener el detalle de las reservas realizadas en el dia, datos : nombre del campo deportivo, el cliente que realizo la reserva, el precio de la reserva. * El documento debe tener un espacio para la firma del administrador. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 10** | **Usuario:** Administrador | |
| **Nombre historia:** Yo como administrador quisiera generar un reporte de los ingresos por reservas. | | |
| **Prioridad en negocio:**  Media | | **Riesgo en desarrollo:**  Media |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Desarrollador responsable:** Beimar E. Huarachi Mamani | | |
| **Descripción:**  Yo como administrador quisiera generar un reporte de los ingresos por reservas. Me gustaria generar un reporte de ingresos por gestion, donde me muestre los ingresos de cada mes. | | |
| **Criterios de Aceptación:**   * El reporte generado debe ser de alguna gestion especifica. * Deberia ser capaz de seleccionar la gestion del reporte. * El reporte deberia mostrar los ingresos de las reservas por cada mes de la gestion seleccionada. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 11** | **Usuario:** Cliente | |
| **Nombre historia:** Yo como usuario del sistema web me gustaría buscar campos deportivos | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  Media |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Desarrollador responsable:** Beimar E. Huarachi Mamani | | |
| **Descripción:**  Yo como usuario del sistema web me gustaria buscar campos deportivos por complejo y por disciplina deportiva | | |
| **Criterios de Aceptación:**   * Deberia ser capaz de buscar campos deportivos por disciplina. * Deberia ser capaz de buscar campos deportivos por complejo deportivo. * Al encontrar una cancha deberia poder ver la disponibilidad de sus horarios. * Al encontrar un campo deportivo deberia ser capaz de ver informacion del complejo al que pertenece. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 12** | **Usuario:** Administrador | |
| **Nombre historia:** Como administrador me gustaría poder modificar los precios por hora y las horas de atención de los campos deportivos. | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  Media |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Desarrollador responsable:** Beimar E. Huarachi Mamani | | |
| **Descripción:**  Como administrador me gustaria poder modificar los precios por hora y las horas de atencion de los campos deportivos. | | |
| **Criterios de Aceptación:**   * Los cambios que se realizen deben verse en la lista de campo deportivos registrados. * Debo ser capaz de seleccionar cualquier campo deportivo para actualizar los precios y horarios de atencion. * Debo recibir la notificacion de confirmacion, de que el campo deportivo se modifico. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 13** | **Usuario:** Administrador | |
| **Nombre historia:** Como administrador me gustaría tener notificaciones de las pre reservas que realizan los clientes. | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Desarrollador responsable:** Beimar E. Huarachi Mamani | | |
| **Descripción:**  Como administrador me gustaria recibir notificaciones de las prereservas que realizen los clientes del complejo deportivo. | | |
| **Criterios de Aceptación:**   * Me deben llegar notificaciones solo de la prereservas actuales. * Si un cliente realiza una pre reserva, debe existir una notificacion. * Debo tener siempre disponible la lista de prereservas. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 14** | **Usuario:** Administrador | |
| **Nombre historia:** Yo como administrador quisiera tener reporte de las canchas más utilizadas. | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Desarrollador responsable:** Beimar E. Huarachi Mamani | | |
| **Descripción:**  Yo como administrador quisiera tener reporte de las canchas mas utilizadas por los usuarios. | | |
| **Criterios de Aceptación:**   * El grafico debe estar de acuerdo al número de reservas por cada campo * Se debe ver el grafico de los campos más requeridos. * Debo ser capaz de identificar el campo más requerido. * Debo ser capaz de escoger la gestión para el reporte. * Si hago una reserva debe reflejarse en el gráfico. | | |

## 5.2 Planificación del proyecto

### 5.2.1 Identificación de actividades

Actualmente el proyecto se encuentra estimado para 5 iteraciones, dentro de las fechas establecidas en la tabla número uno. Bajo la siguiente descripción:

* Los días laborales a tomar en cuenta serán de lunes a viernes.
* Cada iteración durará 10 días laborales (2 semanas)
* El último sprint será dedicado a la realización de pruebas e implantación.
* Las reuniones con el dueño del producto se realizaran cada miércoles, donde se llevara a cabo demostraciones, en las fechas definidas en la tabla 5.2.

|  |  |  |  |
| --- | --- | --- | --- |
| Etapa | Inicio | Fin | Días |
| Primera Iteración | 10-Dic-13 | 18-Dic-13 | 10 |
| Segunda Iteración | 19-Dic-13 | 2-Oct-13 | 10 |
| Tercera Iteración | 3-Dic-13 | 16-Oct-13 | 10 |
| Cuarta iteración | 17-Oct-13 | 30-Oct-13 | 10 |
| Quinta iteración | 31-Oct-13 | 13-Nov-13 | 10 |
|  | | | 50 |

*Tabla 5.1: Duración de las iteraciones (Elaboración propia, 2015)*

|  |  |
| --- | --- |
| Sprint | Fecha |
| Primera Iteracion | 18-Sep-13 |
| Segunda Iteracion | 02-Oct-13 |
| Tercera Iteracion | 16-Oct-13 |
| Cuarta Iteracion | 30-Oct-13 |
| Quinta Iteracion | 13-Nov-13 |

*Tabla 5.2: Fechas de las reuniones programadas con el product owner (Elaboración propia, 2015)*

## 5.2.2 Estimación de costos

#### 5.2.2.1 Factor de ajuste

|  |  |  |
| --- | --- | --- |
| # | Característica General del Sistema | Valor |
| 1 | Comunicaciones de datos | 5 |
| 2 | Datos o procesamiento distribuidos | 0 |
| 3 | Objetivos de rendimiento | 3 |
| 4 | Configuración utilizada masivamente | 0 |
| 5 | Tasa de transacción | 5 |
| 6 | Entrada de datos on-line | 5 |
| 7 | Eficiencia para el usuario | 2 |
| 8 | Actualización on-line | 5 |
| 9 | Procesamiento complejo | 5 |
| 10 | Reutilización | 0 |
| 11 | Facilidad de instalación y conversión | 0 |
| 12 | Facilidad de operación | 5 |
| 13 | Puestos múltiples | 0 |
| 14 | Facilidad de cambio | 3 |
| Total | | 38 |

#### 5.2.2.2 Puntos de Función

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Parámetros de medida | Conteo | Simple | Medio | Complejo | Total | |
| # Entradas de usuario | 9 | 3 | 4 | 6 | 36 | |
| # Salidas de usuario | 6 | 4 | 5 | 7 | 30 | |
| # De consultas | 2 | 3 | 4 | 6 | 8 | |
| # De archivos | 6 | 7 | 10 | 15 | 60 | |
| # De interfaces externas | 0 | 5 | 7 | 10 | 0 | |
| Puntos de función no ajustados | | | | | | 134 |
| Factor de ajuste | | | | | | 38 |
| Puntos de función | | | | | | 138,02 |

#### 5.2.2.3 Esfuerzo, tiempo y número de personas

Basados en la técnica de estimación de COCOMO Orgánico se tienen las siguientes formulas:

Dónde:

E = Esfuerzo

D = Tiempo de desarrollo en meses

KLDC = es el número de miles de líneas de código estimado para el proyecto.

M = Factor de ajuste del esfuerzo, igual a 1 en COCOMO Orgánico.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tipo de Proyecto | a | b | c | d |
| Orgánico | 2,4 | 1,05 | 2,5 | 0,38 |

#### Líneas de Código

Para efectos de PHP cada punto de fusión corresponde a 67 líneas de código fuente, teniendo el resultado total:

#### Esfuerzo

Dadas las líneas de código estimadas, la estimación del esfuerzo será:

#### Duración

Utilizando la estimación del esfuerzo, la duración del proyecto resultara en:

#### Personal

El valor de la duración del proyecto permite recomendar un número de personas N para el proyecto:

#### 5.2.2.4 Determinación del costo

**Costo** **Programador**

**Costos Indirectos en desarrollo**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Equipamiento | Cantidad | Tiempo | Costo ($us/Mes) | Monto($us) |
| Computadoras | 1 | 3 Meses | 40 | 360 |
| Insumos |  |  |  | 300 |
| Material de Escritorio |  |  |  | 100 |
| Mantenimiento del Sistema |  |  |  | 400 |
| Total | | | | 1160 |

**Costo Total**

|  |  |
| --- | --- |
| Trabajo | Costo($us) |
| Costo del Software | 1436,8 |
| Costos Indirectos | 1160,0 |
| Total | 3596,8 |

### 5.3.3 Planificación de entregables de incrementos de las iteraciones (Release Plan)

|  |  |  |
| --- | --- | --- |
| Etapa | Fecha de Pago | Contenido / Incremento |
| Primera Iteración | 09-Dic-16 | * Diseño inicial de base de datos * Definición de historias de usuario iniciales con estimaciones * Diseño inicial página web * Configuración inicial en el servidor del cliente para posteriores integraciones |
| Segunda Iteración | 22-Dic-16 | * Entregable básico * Diseño de interfaz * Correcciones sugeridas * Pruebas |
| Tercera Iteración | 16-Ene-16 | * Entregable básico * Correcciones sugeridas de la iteración anterior * Pruebas |
| Cuarta Iteración | 13-Ene-16 | * Entregable básico * Correcciones sugeridas de la iteración anterior * Pruebas |
| Quinta Iteración | 6-Feb-16 | * Correcciones sugeridas del sprint anterior * Manuales de Usuario * Implantación final del software * Pruebas |

# CAPITULO VI – PRIMERA ITERACÍON

La metodologia scrum es iterativa e incremental, por lo tanto en este capitulo, se abarca todo lo relacionado al desarrollo de la primera iteracion.

## 6.1 Planificación de la iteración

La planificación de la iteración, tiene como puntos importantes: La pila de la iteración, estimación de esfuerzo necesario para el desarrollo de las historias de usuario y la definición de tareas por historia de usuario.

### 6.1.1 Pila de la Iteración

Para la primera iteración del proyecto, se ha tomado la decisión de desarrollar las siguientes historias de usuario, de acuerdo a su prioridad, ya que son altas. Dichas historias de usuario se pueden ver en la tabla 6.1.

|  |  |  |
| --- | --- | --- |
| **Numero Historia de Usuario** | **Nombre** | **Prioridad** |
| 1 | Registro de campo deportivo | Alta |
| 2 | Registro de reserva regular | Alta |

### 6.1.2 Estimación de esfuerzo

La estimación de las historias de usuario se ha realizado de cuerdo a la serie Fibonacci, tomando como 1 la menor complejidad y 8 mayor complejidad.

|  |  |  |
| --- | --- | --- |
| **Numero Historia de Usuario** | **Nombre** | **Estimación** |
| 1 | Registro de campo deportivo | 8 puntos |
| 2 | Registro de reserva regular | 8 puntos |

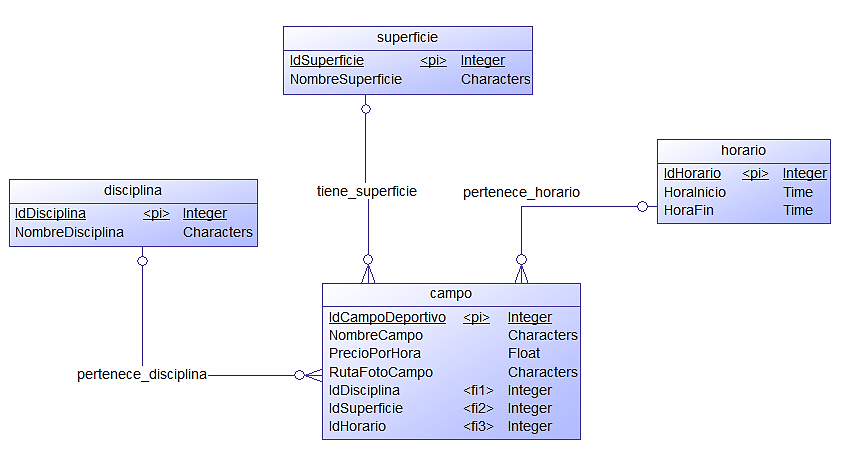
### 6.1.3 Definición de tareas

A continuación, se definen las tareas necesarias para cumplir el desarrollo de las historias de usuario.

|  |  |
| --- | --- |
| **Numero Historia de Usuario** | **Tareas** |
| 1 | * Desarrollar la interfaz de usuario para el registro. * Realizar un diseño inicial de la base de datos. * Implementar la lógica para el registro del campo deportivo. * Realizar las pruebas respectivas. |
| 2 | * Diseñar la tabla para realizar las reservas. * Agregar tablas a la base de datos para registro de reservas. * Implementar los servicios en Angular Js. * Implementar los servicios en CodeIgniter para el registro de reservas. * Implementar la integración de los servicios para la funcionalidad. * Realizar las pruebas correspondientes. |

## 6.3 Diseño

El diseño inicial de la base de datos que se ha realizado, ha tomado en cuenta todas las entidades que son necesarias para el desarrollo de las historias de usuario seleccionadas para la primera iteración del proyecto.



*Figura 6.1: Diseño inicial de la base de datos (Elaboracion propia, 2016)*

## 6.5 Pruebas

Para realizar el control de calidad del desarrollo, se ha utilizado las platillas que se muestran en la parte inferior, donde se observa a detalle los pasos que se deben seguir para los casos de prueba.

Las plantillas son para realizar el control de calidad de manera manual, ya que no se ha implementado pruebas automatizadas.

|  |  |
| --- | --- |
| Titulo | Verificar la notificación de registro de campos deportivos. |
| Descripción | Verificar que al realizar un registro de campo deportivo con los datos correspondientes, el sistema debe notificar que el campo deportivo se ha registrado correctamente. |
| Requerimientos | * Navegador web |
| Pasos | 1. Abrir el navegador. 2. Introducir la siguiente url: <http://localhost/compleapp/complejoapp/#/admin/registrarCampo>. 3. Verificar que el formulario de registro se encuentre cargado 4. Llenar el formulario de registro con los datos del campo deportivo. 5. Presionar el botón “Guardar”. 6. Verificar que la notificación de éxito se muestre. |
| Resultado Esperado | La notificación de éxito al registrar un campo deportivo debe mostrarse. |

|  |  |
| --- | --- |
| Titulo | No debería habilitarse el botón de registro, si los campos del formulario no están introducidos. |
| Descripción | El botón de registro de campos deportivos no debería habilitarse, si alguno de los campos del formulario de registro no han sido introducidos previamente. |
| Requerimientos | * Navegador web |
| Pasos | 1. Abrir el navegador. 2. Introducir la siguiente url: <http://localhost/compleapp/complejoapp/#/admin/registrarCampo>. 3. Verificar que el formulario de registro se encuentre cargado 4. No llenar el formulario de registro con los datos del campo deportivo. 5. Intentar presionar el botón “Guardar”. 6. Verificar que el botón “Guardar” no se encuentre habilitado. |
| Resultado Esperado | No se habilita el botón de “Guardar”. |

## 6.6 Revisión de la Iteración

A continuación, se muestra la tabla, en donde se puede observar las historias de usuario que se consideran como terminadas de la primera iteración y además, resultan como el incremento de la primera iteración.

Una observación importante, es que en la primera iteración, no se ha desarrollado una historia de usuario por completo, que estaba planificada para la iteración, esto es, porque ha sido la primera vez que se ha desarrollado con el framework Angular, por lo tanto, se ha tomado más tiempo para el desarrollo.

|  |  |  |  |
| --- | --- | --- | --- |
| **Numero H.U.** | **Criterios de Aceptación** | **Cumple** | **Historia Aceptada** |
| 1 | * Debo ser capaz de agregar información descriptiva de cada cancha datos: Nombre campo deportivo, horarios de atención, precio por hora de reserva, tipo de disciplina y tipo de superficie. | SI | SI |
| * Todos los datos anteriormente citados deben ser introducidos obligatoriamente. | SI |
| * Debo ser capaz de agregar imágenes de las canchas. | SI |
| 2 | * Debo ser capaz de agregar una reserva con los siguientes datos: nombre cliente, teléfono, campo deportivo, hora, fecha, cantidad de horas. | NO | NO |
| * Debo ser capaz de ver todas las reservas ya realizadas. | NO |
| * No debe registrar una reserva con fecha menor a la actual. | NO |
| * No debe registrar una reserva con fecha actual y hora menor a la hora actual. | NO |
| * No debe registrar una reserva, si existe otra reserva para el mismo campo, el mismo día y exista colisión en horarios. | NO |
| * No debe registrar una reserva, que tenga horario inicio mayor o igual al horario final, | NO |
| * No debe registrar una reserva, cuya duración es menor a una hora. | NO |

### 6.6.1 Grafico BurnDown

El siguiente grafico pertenece al burndown chart de la primera iteración. Realizando un análisis de este gráfico, se puede concluir que en la primera iteración, no se han cumplido los puntos estimados, ya que la línea de “Real” está por encima de lo estimado. Además, en esta iteración quedaron puntos de esfuerzo pendientes, que van a ser tomados en cuenta para la segunda iteración.

Los puntos no quemados pertenecen a la historia de usuario número dos.

# CAPITULO VII – SEGUNDA ITERACÍON

La metodologia scrum es iterativa e incremental, por lo tanto este capitulo, abarca todo lo relacionado al desarrollo de la segunda iteracion.

## 6.1 Planificación de la iteración

Para la planificación de la segunda iteración, se ha tomado en cuenta los puntos pendientes o la historia de usuario no aceptada en la primera iteración. Dicha historia de usuario fue introducida en la pila del producto de la segunda iteración.

### 6.1.1 Pila de la Iteración

Para el desarrollo de la segunda iteración del proyecto, se ha tomado la decisión de desarrollar las siguientes historias de usuario, tomando en cuenta la prioridad y complejidad. Dichas historias de usuario se pueden ver en la tabla 7.1.

|  |  |  |
| --- | --- | --- |
| **Numero Historia de Usuario** | **Nombre** | **Prioridad** |
| 2 | Registro de reserva regular | Alta |
| 4 | Yo como administrador me gustaria registrar reserva administrativa. | Alta |
| 6 | Como administrador me gustaria realizar reservas periódicas con las Reservas Especiales y administrativas | Alta |

### 6.1.2 Estimación de esfuerzo

La estimación de las historias de usuario se las realizo de cuerdo a la serie Fibonacci, tomando como 1 la menor complejidad y 8 mayor complejidad.

|  |  |  |
| --- | --- | --- |
| **Numero Historia de Usuario** | **Nombre de Historia de Usuario** | **Estimación** |
| 2 | Registro de reserva regular | 8 puntos |
| 4 | Yo como administrador me gustaria registrar reserva administrativa. | 5 puntos |
| 6 | Como administrador me gustaria realizar reservas periódicas con las Reservas Especiales y administrativas | 8 puntos |

### 6.1.3 Definición de tareas

A continuación, se definen las tareas necesarias para cumplir como las historias de usuario planificadas.

|  |  |
| --- | --- |
| **Numero Historia de Usuario** | **Tareas** |
| 2 | * Diseñar la base de datos para realizar las reservas. * Agregar tablas a la base de datos para registro de reservas. * Implementar los servicios en Angular Js. * Implementar los servicios en CodeIgniter para el registro de reservas. * Implementar la integración de los servicios para la funcionalidad. * Realizar las pruebas correspondientes. |
| 4 | * Modificar la base de datos para poder realizar las reservas administrativas. * Diseñar e implementar el formulario para el registro. * Implementar los servicios en Angular y Codeigniter. * Hacer la integración de la comunicación de cliente y servidor. * Realizar las pruebas. |
| 6 | * Modificar la implementación en los controladores de Angular en reservas regulares. * Modificar los servicios RESTFULL. * Modificar la interfaz para permitir el registro de reservas periódicas. * Implementar la integración entre cliente y servidor. * Realizar pruebas. |

## 6.3 Diseño

El diseño inicial de la base de datos ha sido modificado para la segunda iteración, se han agregado tablas para dar soporte a los registros de reservas regulares, administrativas y periódicas. Dichos cambios, se pueden observar en la figura 7.1.

C:\Users\Beimar\Desktop\segundo sprint.emf

*Figura 6.1: Diseño de la base de datos (Elaboracion propia, 2016)*

## 6.5 Pruebas

Para realizar el control de calidad del desarrollo, se ha utilizado las platillas que se muestran en la parte inferior, donde se observa a detalle los pasos que se deben seguir para los casos de prueba.

Han sido agregados los casos de prueba necesarios para cubrir los criterios de aceptación.

|  |  |
| --- | --- |
| Titulo | Verificar la notificación de registro de campos deportivos. |
| Descripción | Verificar que al realizar un registro de campo deportivo con los datos correspondientes, el sistema debe notificar que el campo deportivo se ha registrado correctamente. |
| Requerimientos | * Navegador web |
| Pasos | 1. Abrir el navegador. 2. Introducir la siguiente url: <http://localhost/compleapp/complejoapp/#/admin/registrarCampo>. 3. Verificar que el formulario de registro se encuentre cargado 4. Llenar el formulario de registro con los datos del campo deportivo. 5. Presionar el botón “Guardar”. 6. Verificar que la notificación de éxito se muestre. |
| Resultado Esperado | La notificación de éxito al registrar un campo deportivo debe mostrarse. |

|  |  |
| --- | --- |
| Titulo | No debería habilitarse el botón de registro, si los campos del formulario no están introducidos. |
| Descripción | El botón de registro de campos deportivos no debería habilitarse, si alguno de los campos del formulario de registro no han sido introducidos previamente. |
| Requerimientos | * Navegador web |
| Pasos | 1. Abrir el navegador. 2. Introducir la siguiente url: <http://localhost/compleapp/complejoapp/#/admin/registrarCampo>. 3. Verificar que el formulario de registro se encuentre cargado 4. No llenar el formulario de registro con los datos del campo deportivo. 5. Intentar presionar el botón “Guardar”. 6. Verificar que el botón “Guardar” no se encuentre habilitado. |
| Resultado Esperado | No se habilita el botón de “Guardar”. |

## 6.6 Revisión de la Iteración

A continuación, se muestra la tabla, en donde se puede observar las historias de usuario que se consideran como terminadas de la segunda iteración y además, resultan como el incremento de la segunda iteración.

Una observación importante, es que en esta segunda iteración ya se tiene un dominio sobre las tecnologías, que dan como resultado un mejor desempeño en esta iteración.

|  |  |  |  |
| --- | --- | --- | --- |
| **Numero H.U.** | **Criterios de Aceptación** | **Cumple** | **Historia Aceptada** |
| 2 | * Debo ser capaz de agregar una reserva con los siguientes datos: nombre cliente, teléfono, campo deportivo, hora, fecha, cantidad de horas. | SI | SI |
| * Debo ser capaz de ver todas las reservas ya realizadas. | SI |
| * No debe registrar una reserva con fecha menor a la actual. | SI |
| * No debe registrar una reserva con fecha actual y hora menor a la hora actual. | SI |
| * No debe registrar una reserva, si existe otra reserva para el mismo campo, el mismo día y exista colisión en horarios. | SI |
| * No debe registrar una reserva, que tenga horario inicio mayor o igual al horario final, | SI |
| * No debe registrar una reserva, cuya duración es menor a una hora. | SI |
| 4 | * Debo ser capaz de resgistrar una reserva administrativa. | SI | SI |
| * Debo ser capaz de ver todas las reservas ya registradas. | SI |
| * No debe registrar una reserva con fecha menor a la actual. | SI |
| * No debe registrar una reserva con fecha actual y hora menor a la hora actual. | SI |
| * No debe registrar una reserva, si existe otra reserva para el mismo campo, el mismo día y exista colisión en horarios. | SI |
| * No debe registrar una reserva, que tenga horario inicio mayor o igual al horario final. | SI |
| * No debe registrar una reserva, cuya duración es menor a una hora. | SI |
| 6 | * Realiza la reserva si está o sus repeticiones no tienen choques con otras reservas y sus repeticiones respectivas y si sus datos son válidos. | SI | SI |
| * Como maximo, se deben realizar solo 10 reservas por un cliente. | SI |
| * Se debe poder especificar de que tipo de reserva periodica se esta registrando. | SI |

### 6.6.1 Grafico BurnDown

Analizando el grafico burndown de la segunda iteración, se puede concluir que se han completado 21 puntos, lo cual indica que las historias de usuario planificadas para la iteración 2, han sido aceptadas satisfactoriamente. Haciendo una comparativa con la primera iteración, se puede observar que se ha aumentado la velocidad de desarrollo.

# CAPITULO VIII – TERCERA ITERACÍON

La metodologia scrum es iterativa e incremental, por lo tanto en este capitulo, se abarca todo lo relacionado al desarrollo de la tercera iteracion.

## 6.1 Planificación de la iteración

En la planificación de la tercera iteración se han tomado historias de usuario nuevas de la pila del producto, ya que en la segunda iteración, no quedaron puntos pendientes.

### 6.1.1 Pila de la Iteración

A continuación se detalla las historias de usuario que han sido planificadas para la tercera iteración.

|  |  |  |
| --- | --- | --- |
| **Numero Historia de Usuario** | **Nombre** | **Prioridad** |
| 3 | Yo como administrador quisiera registrar clientes | Alta |
| 5 | Como Cliente del complejo deportivo me gustaría realizar pre reservas. | Media |
| 7 | Yo como administrador quisiera confirmar las pre reservas. | Media |
| 13 | Como administrador me gustaría tener notificaciones de las pre reservas que realizan los clientes. | Baja |
| 12 | Como administrador me gustaría poder modificar los precios por hora y las horas de atención de los campos deportivos. | Media |
| 14 | Yo como administrador quisiera tener reporte de las canchas más utilizadas. | Baja |

### 6.1.2 Estimación de esfuerzo

El total de puntos que se estimaron para esta iteración es de 24. La información se detalla en la tabla 8.2.

|  |  |  |
| --- | --- | --- |
| **Numero Historia de Usuario** | **Nombre** | **Estimación** |
| 3 | Yo como administrador quisiera registrar clientes. | 3 puntos |
| 5 | Como Cliente del complejo deportivo me gustaría realizar pre reservas. | 5 puntos |
| 7 | Yo como administrador quisiera confirmar las pre reservas. | 3 puntos |
| 13 | Como administrador me gustaría tener notificaciones de las pre reservas que realizan los clientes. | 2 puntos |
| 12 | Como administrador me gustaría poder modificar los precios por hora y las horas de atención de los campos deportivos. | 3 puntos |
| 14 | Yo como administrador quisiera tener reporte de las canchas más utilizadas. | 8 puntos |

### 6.1.3 Definición de tareas

A continuación, se definen las tareas necesarias para cumplir el desarrollo de las historias de usuario.

|  |  |
| --- | --- |
| **Numero Historia de Usuario** | **Tareas** |
| 3 | * Diseñar e implementar el formulario de registro. * Implementar los servicios en Angular y CodeIgniter. * Modificar el modelo de la base de datos. * Realizar casos de pruebas. |
| 5 | * Diseñar una de usuario interfaz para los clientes. * Hacer una modificación del modelo de la base de datos. * Realizar casos de pruebas. |
| 7 | * Implementar las clases sobre la confirmación de pre reservas en Angular. * Realizar casos pruebas. |
| 13 | * Diseñar la presentación para las notificaciones. * Implementar las consultas necesarias a la base de datos para la recepción de notificaciones. * Implementar la integración con el registro de las pre reservas y confirmación. * Realizar casos pruebas. |
| 12 | * Implementar los verbos PUT y GET en CodeIgniter para realizar la edición de campos deportivos. * Modificar la interfaz de registro de campos deportivos, para permitir hacer la edición de los mismos. * Realizar casos de prueba. |
| 14 | * Realizar una investigación sobre Morris js, el cual permitirá realizar gráficos. * Implementar las consultas a la base de datos necesarias para generar el reporte. * Implementar los servicios en Angular y Codeigniter. * Diseñar e implementar la interfaz para la generación de los reportes. * Realizar casos de prueba. |

## 6.3 Diseño

Los cambios que se han hecho al modelo de la base de datos se puede observar en la figura 8.1.

C:\Users\Beimar\Desktop\tercero.emf

*Figura 6.1: Diseño de la base de datos para la iteración 3 (Elaboracion propia, 2016)*

## 6.5 Pruebas

Para realizar el control de calidad del desarrollo, se ha utilizado las platillas que se muestran en la parte inferior, donde se observa a detalle los pasos que se deben seguir para los casos de prueba.

Las plantillas son para realizar el control de calidad de manera manual, ya que no se ha implementado pruebas automatizadas.

|  |  |
| --- | --- |
| Titulo | Verificar la notificación de registro de campos deportivos. |
| Descripción | Verificar que al realizar un registro de campo deportivo con los datos correspondientes, el sistema debe notificar que el campo deportivo se ha registrado correctamente. |
| Requerimientos | * Navegador web |
| Pasos | 1. Abrir el navegador. 2. Introducir la siguiente url: <http://localhost/compleapp/complejoapp/#/admin/registrarCampo>. 3. Verificar que el formulario de registro se encuentre cargado 4. Llenar el formulario de registro con los datos del campo deportivo. 5. Presionar el botón “Guardar”. 6. Verificar que la notificación de éxito se muestre. |
| Resultado Esperado | La notificación de éxito al registrar un campo deportivo debe mostrarse. |

|  |  |
| --- | --- |
| Titulo | No debería habilitarse el botón de registro, si los campos del formulario no están introducidos. |
| Descripción | El botón de registro de campos deportivos no debería habilitarse, si alguno de los campos del formulario de registro no han sido introducidos previamente. |
| Requerimientos | * Navegador web |
| Pasos | 1. Abrir el navegador. 2. Introducir la siguiente url: <http://localhost/compleapp/complejoapp/#/admin/registrarCampo>. 3. Verificar que el formulario de registro se encuentre cargado 4. No llenar el formulario de registro con los datos del campo deportivo. 5. Intentar presionar el botón “Guardar”. 6. Verificar que el botón “Guardar” no se encuentre habilitado. |
| Resultado Esperado | No se habilita el botón de “Guardar”. |

## 6.6 Revisión de la Iteración

A continuación, se muestra la tabla, en donde se puede observar las historias de usuario que se consideran como terminadas de la primera iteración y además, resultan como el incremento de la primera iteración.

|  |  |  |  |
| --- | --- | --- | --- |
| **Numero H.U.** | **Criterios de Aceptación** | **Cumple** | **Historia Aceptada** |
| 3 | * Se debe poder registrar un cliente con los datos: Telefono, Nombres, Apellidos, Numero CI y deseable el correo electronico. | SI | SI |
| * El Numero de C.I. debe ser unico para cada cliente. | SI |
| * Los nombres y apellidos deben obligatorios para cada cliente. | SI |
| 5 | * No puedo colisionar con otras prereservas o reservas. | SI | SI |
| * Avisar a cerca del Plazo para pagar precio prereserva a Admi. | SI |
| * El Cliente solo puede realizar una prereserva, y ademas haber solicitado reservas anteriormente al complejo. | SI |
| * Datos necesarios iguales al de una reserva regular. | SI |
| * Las prereservas se realizan 48 antes de la fecha establecida. | SI |
| 7 | * Solo deben confirmarse las prereservas actuales. | SI | SI |
| * Debo poder buscar el nombre del cliente que solicito la pre reserva. | SI |
| 13 | * Me deben llegar notificaciones solo de la prereservas actuales. | SI | SI |
| * Si un cliente realiza una pre reserva, debe existir una notificacion. | SI |
| * Debo tener siempre disponible la lista de prereservas. | SI |
| 12 | * Los cambios que se realizen deben verse en la lista de campo deportivos registrados. | SI | SI |
| * Debo ser capaz de seleccionar cualquier campo deportivo para actualizar los precios y horarios de atencion. | SI |
| * Debo recibir la notificacion de confirmacion, de que el campo deportivo se modifico. | SI |
| 14 | * El grafico debe estar de acuerdo al número de reservas por cada campo. | SI | SI |
| * Se debe ver el grafico de los campos más requeridos. | SI |
| * Debo ser capaz de identificar el campo más requerido. | SI |
| * Debo ser capaz de escoger la gestión para el reporte. | SI |
| * Si hago una reserva debe reflejarse en el gráfico. | SI |

### 6.6.1 Grafico BurnDown

El siguiente grafico pertenece al burndown chart de la tercera iteración.

Dado este gráfico, se puede concluir que los 24 puntos estimados para la tercera iteración han sido cumplidos satisfactoriamente.

# CAPITULO IX – CUARTA ITERACÍON

La metodologia scrum es iterativa e incremental, por lo tanto en este capitulo, se abarca todo lo relacionado al desarrollo de la cuarta iteracion.

## 6.1 Planificación de la iteración

En esta iteración han sido tomados en cuenta historias de usuario nuevas para el desarrollo, porque en la tercera iteración, no se dejaron puntos pendientes por terminar.

### 6.1.1 Pila de la Iteración

La lista de historias de usuario se muestra en la tabla …...

|  |  |  |
| --- | --- | --- |
| **Numero Historia de Usuario** | **Nombre** | **Prioridad** |
| 8 | Yo como administrador quisiera imprimir los recibos para las reservas. | Alta |
| 10 | Yo como administrador quisiera generar un reporte de los ingresos por reservas. | Media |
| 9 | Yo como administrador quisiera poder imprimir los reportes diarios de ingresos por Reservas realizadas. | Media |
| 11 | Yo como usuario del sistema web me gustaría buscar campos deportivos | Baja |

### 6.1.2 Estimación de esfuerzo

La suma de puntos de esfuerzo para la cuarta iteración es 21 puntos y se detallan a continuación.

|  |  |  |
| --- | --- | --- |
| **Numero Historia de Usuario** | **Nombre** | **Estimación** |
| 8 | Yo como administrador quisiera imprimir los recibos para las reservas. | 8 puntos |
| 10 | Yo como administrador quisiera generar un reporte de los ingresos por reservas. | 5 puntos |
| 9 | Yo como administrador quisiera poder imprimir los reportes diarios de ingresos por Reservas realizadas. | 5 puntos |
| 11 | Yo como usuario del sistema web me gustaría buscar campos deportivos | 3 puntos |

### 6.1.3 Definición de tareas

A continuación, se definen las tareas necesarias para cumplir el desarrollo de las historias de usuario.

|  |  |
| --- | --- |
| **Numero Historia de Usuario** | **Tareas** |
| 8 | * Investigar sobre pdfmake, para la generación de archivos con formato pdf. * Diseñar el recibo de las reservas deacuerdo a los recibos originales del complejo San Simon. * Hacer las modificaciones necesarias al modelo de la base de datos. * Implementar la clase generadora de archivos con formato pdf. * Realizar los casos de prueba. |
| 10 | * Diseñar la interfaz para los reportes de ganancias. * Implementar las clases necesarias para la generación del grafico y tabla del reporte. * Implementar las consultas necesarias para la generación del reporte. * Implementar los servicios en CodeIgniter. * Implementar la integración de los servicios para la funcionalidad. * Realizar los casos de prueba. |
| 9 | * Diseñar el formato del documento para los reportes e implementar con pdfmake. * Implementar las consultas necesarias para generar el reporte diario. * Implementar las funciones para imprimir el documento. * Realizar los casos de prueba. |
| 11 | * Diseñar la interfaz para la búsqueda de campos deportivos. * Implementar las funciones para la búsqueda. * Implementar las interfaces para mostrar resultados de las búsquedas. * Realizar los casos de prueba. |

## 6.3 Diseño

La modificación al modelo de la base de datos se muestra en la figura …...

C:\Users\Beimar\Desktop\cuarto.emf

*Figura 6.1: Diseño de la base de datos para la cuarta iteracion (Elaboracion propia, 2016)*

## 6.5 Pruebas

Para realizar el control de calidad del desarrollo, se ha utilizado las platillas que se muestran en la parte inferior, donde se observa a detalle los pasos que se deben seguir para los casos de prueba.

Las plantillas son para realizar el control de calidad de manera manual, ya que no se ha implementado pruebas automatizadas.

## 6.6 Revisión de la Iteración

A continuación, se muestra la tabla, en donde se puede observar las historias de usuario que se consideran como terminadas de la primera iteración y además, resultan como el incremento de la primera iteración.

|  |  |  |  |
| --- | --- | --- | --- |
| **Numero H.U.** | **Criterios de Aceptación** | **Cumple** | **Historia Aceptada** |
| 8 | * El recibo debe contener la siguiente informacion: Nombre del encargado, nombre del cliente, C.I. del cliente, numero de recibo, detalle de la reserva, nombre del campo deportivo y precio. | SI | SI |
| * El precio total debe indicarse tanto en letras como en cantidad en digitos numeros. | SI |
| 10 | * El reporte generado debe ser de alguna gestion especifica. | SI | SI |
| * Deberia ser capaz de seleccionar la gestion del reporte. | SI |
| * El reporte deberia mostrar los ingresos de las reservas por cada mes de la gestion seleccionada. | SI |
| 9 | * El documento debe tener los datos del complejo deportivo. | SI | SI |
| * El documento debe tener el detalle de las reservas realizadas en el dia, datos : nombre del campo deportivo, el cliente que realizo la reserva, el precio de la reserva. | SI |
| * El documento debe tener un espacio para la firma del administrador. | SI |
| 11 | * Deberia ser capaz de buscar campos deportivos por disciplina. | SI | SI |
| * Deberia ser capaz de buscar campos deportivos por complejo deportivo. | SI |
| * Al encontrar una cancha deberia poder ver la disponibilidad de sus horarios. | SI |
| * Al encontrar un campo deportivo deberia ser capaz de ver informacion del complejo al que pertenece. | SI |

### 6.6.1 Grafico BurnDown

El siguiente grafico pertenece al burndown chart de la cuarta iteración. Para la cuarta iteración se estimó un total de 21 puntos de esfuerzo. Analizando el grafico se concluye que, en esta iteración se quemaron los puntos de esfuerzo hasta el noveno día, lo que indica que la estimación no fue eficiente.

# CAPITULO X – PRUEBAS DE VALIDACION E IMPLANTACION

Este capitulo corresponde a la quinta iteracion, pero dada las caracteristicas de esta iteracion, se la ha denominado “Pruebas de validación e implantación”.

## 10.1 Implantación

### 10.1.1 Herramientas para el desplegado de la aplicación

### 10.1.2 Servidor de producción

### 10.1.2 Características del servidor de producción

## 10.2 Pruebas de validación

Cuadro para las pruebas de validación.

# CAPITULO XI – CONCLUSIONES Y RECOMENDACIONES

El trabajo de grado, cuyo objetivo principal era el de:

“Mejorar el manejo de información de la administración de horarios y reservas de las canchas deportivas, a través del desarrollo de un sistema web para la administración de horarios y reservas para complejos, usando cloud computing” ha sido concluido satisfactoriamente, el producto obtenido, es un sistema web que ha sido probado para la administración de campos deportivos del complejo deportivo de la Universidad Mayor de San Simón.

A continuación, se describe las conclusiones:

* El proceso manual que comprende la administración de horarios y reservas de complejos deportivos, requiere de un tiempo considerable, ya que la información almacenada es abundante, en cambio con el sistema desarrollado, el tiempo de administración se reduce de una manera apreciable.
* Ya que el sistema se encuentra integrado con Google Maps, los usuarios del sistema web, podrán ubicar campos deportivos de una manera amigable y de acuerdo a su ubicación geográfica.
* Como el sistema web está implantado en un servidor de la nube, los administradores de complejos deportivos pueden acceder a la información desde cualquier lugar.
* Ya que el sistema web cumple con el diseño web adaptable, los usuarios de la aplicación, podrán hacer uso de la misma, desde cualquier dispositivo.

Sobre la tecnología y herramientas utilizadas:

* Hacer uso del framework Angular JS, ha reducido el tiempo de desarrollo y aumentado la productividad.

Sobre la metodología:

* El presente trabajo de grado ha sido desarrollado por una persona, por lo tanto, no se han usado las ceremonias de la metodología scrum, porque dichas ceremonias están dirigidas a equipos que conforman 2 o más personas. Además, los roles que han sido tomados en cuenta son: el desarrollador y el dueño del producto.

Para trabajos futuros se recomienda:

* Realizar pruebas en el navegador Internet Explorer, ya que cuenta con una cantidad considerable de usuarios.
* Implementar el pago con tarjeta de crédito vía internet, ya que con esta implementación, el proceso quedaría completamente automatizado y el tiempo en la administración de reservas, reduciría de una manera considerable.
* Migrar a la versión 2.0 de Angular, ya que permite una sintaxis orientada a objetos, haciendo el código entendible y de fácil mantenimiento.

# Bibliografía

DOYLE Matt (2010) Beginning PHP 5.3.Ed.Wiley Publishing, Inc.

ETHAN Marcotte (2011) Responsive web Design.Ed.Jeffrey Zeldman

HUNG Ho Ngoc (2014) Single Page Web Application with Restful API and AngularJS .Ed. Helsinki

Mozilla Foundation (s.f.).Javascript.Disponible en: <https://developer.mozilla.org/es/docs/Web/JavaScript>.Consultado en: 19 de febrero de 2016

MySql (s.f.).Mysql Documentation.Disponible en: <http://dev.mysql.com/doc/>.Consultado en: 19 de febrero de 2016

Angular Js (s.f.).Angular Js Documentation.Disponible en: <https://docs.angularjs.org/guide>.Consultado en: 19 de febrero de 2016

Bootstrap (s.f.).Getting Started Bootstrap.Disponible en: <http://getbootstrap.com/getting-started/>.consultado en: 19 de febrero de 2016

IBV, Instituto de biomecanica de Valencia (2011) De la planificacion a la gestion de instalaciones deportivas.Ed.CreaImpresion2000 S.L.U.

LUJAN Sergio (2002) Programación de aplicaciones web: historia, principios básicos y clientes web.Ed.Universidad de Alicante

PALACIO Juan (2014) Gestión de proyectos Scrum manager.Ed.lubaris4MediaS.L.

TORRES Jordi (2011) Empresas en la nube. Ventajas y retos del cloud computing.Ed. Libros de cabecera S.L.

UAM, Universidad Autonoma Metropolitana (2011) Computación en la nube